Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Baghdad
College of Education for Pure Science (Ibn al-Haitham)
Department of Mathematics

Nonparametric Estimation of Hazard Function by Using Wavelet Transformation

A Thesis

Submitted to College of Education for Pure Science (Ibn al-Haitham)

University of Baghdad

In Partial Fulfillment of Requirements for the Degree of Doctor of Philosophy in mathematics

By

Ali Talib Mohammed

Supervised by

Prof. Dr. Iden Hasan Hussein

2019 *1440*

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

إِنَّا فَتَحْنَا لَكَ فَتْحًا مُّبِينًا﴿١﴾لِّيَغْفِرَ لَكَ اللَّهُ مَا تَقَدَّمَ مِن
ذَنبِكَ وَمَا تَأَخَّرَ وَيُتِمَّ نِعْمَتَهُ عَلَيْكَ وَيَهْدِيَكَ صِرَاطًا
مُّسْتَقِيمًا﴿٢﴾وَيَنصُرَكَ اللَّهُ نَصْرًا عَزِيزًا﴿٣﴾هُوَ الَّذِي أَنزَلَ
السَّكِينَةَ فِي قُلُوبِ الْمُؤْمِنِينَ لِيَزْدَادُوا إِيمَانًا مَّعَ إِيمَانِهِمْ ۗ
وَلِلَّهِ جُنُودُ السَّمَاوَاتِ وَالْأَرْضِ ۚ وَكَانَ اللَّهُ عَلِيمًا حَكِيمًا
﴿٤﴾

صدق اللّٰه العَظيم

سورة الفتح

الآيات:(٤٣٢١)

الاهداء

الى وطني.....

إلى من أرسى لديَّ قواعد الخلق الكريم، وكيفية كبح زمام النفس... الحاضر الغائب أبي

إلى من علَّمتني الصبر والجدَّ والاجتهاد في كافة مناحي الحياة... أُمِّي الحبيبة.

الى من كان الحب تاجها والطيبة عنوانها الحاضرة الغائبة..... د.رجاء النجار.

إلى رفيقة الدرب ومن كانت نعم السَّند في رحلتي العلمية والبحثية، ولم تدَّخر جُهدًا في مُساعدتي، بكل الحب.....زوجتي الغالية.

الى سندي وعزوتي في الحياة أخي.....أحمد.

الى وردتي الجميلة حبيبتي.....أختي.

الى قلبي وروحي اولادي.....مريم، محمد، موسى.

Acknowledgments

First of all, I wish to thank **Allah**, my creator, for his help and acceptance of my prayers that make the accomplishment of this work more than a dream after all what happened. I would like to express my deep thanks and gratitude to my supervisor **Prof. Dr. Iden Hasan Hussein** for his valuable guidance and encouragement during my work.

Also, I'm very grateful to the faculty members of mathematics Department in college of education for pure science - Ibn Al-Haitham.

Finally, the words are unable on stating clearly the extent of my thanks for my family, especially my mother, my wife, my brother, and my sister for their supported and encouraged me in all times and help me to obtain my ambition.

Supervisor Certification

I certify that this dissertation was prepared under my supervision at the Department of Mathematics, College of Education for Pure Sciences, Ibn Al-Haitham, University of Baghdad as partial fulfillment of the requirements of the Degree of Doctor of Sciences in Mathematics.

Signature:

Name: Prof. Dr. Iden Hasan Hussein

Date: / / 2019

In view of the available recommendation, I forward this dissertation for debate by the examination committee.

Signature:

Name: Assist. Prof. Dr. Yousif Yaqoub Yousif

Head of the Department of Mathematics.

Date: / / 2019

Committee Certification

We certify that we have read this dissertation entitle **"Nonparametric Estimation of Hazard Function by Using Wavelet"** and as examining committee, we examined the student **(Ali Talib Mohammed)** in its contains and what is connected with it, and that in our opinion it is adequate for the partial fulfillment of the requirement for the degree of Doctor of Science in Mathematics.

(Chairman)
Signature:
Name: Dr. Luma Naji Mohammed
Title: Prof.
Data: / / 2019

(Member)
Signature:
Name: Dr. Oday Taha Raheem
Title: Assist. Prof.
Data: / / 2019

(Member)
Signature:
Name: Dr. Saad Naji Al-Azzawi
Title: Prof.
Data: / / 2019

(Member)
Signature:
Name: Dr. Suhaila Najm Abdullah
Title: Assist. Prof.
Data: / / 2019

(Member)
Signature:
Name: Dr. Hatam Yahya Khalaf
Title: Assist. Prof.
Data: / / 2019

(Member and Supervisor)
Signature:
Name: Dr. Iden Hasan Hussein
Title: Prof.
Data: / / 2019

Approved by the University Committee of graduate studies

Signature:

Name: Prof. Dr. Hasan Ahmed Hasan

Behalf: the Dean of College of Education for Pure Science / Ibn Al- Haitham, University of Baghdad.

Data: / / 2019

ABSTRACT

The main objective of the study in this thesis is to enrich the non-parametric estimation of the hazard rate function using the linear wavelet estimation for right randomly censoring data. The strategy of the estimation is based on the use of the wavelet projection of the father function $\left\{\varphi_{J,k}(x), J = 1,2, \dots, 0 \leq k \leq 2^{J-1}\right\}$ on the resolution subspace $\left(V_J\right)$ of the square integrable function $\left(L^2(R)\right)$, with the Breslow estimator of the cumulative function. The technique to estimate of the hazard function is built to find the ratio between the linear wavelet estimator of the probability density function (f(x)) and the empirical estimation to find the survival function {S(x)=1-F(x)}.

Applications are based on two types of study. First application, the simulation study to generate random variables for two experiments, the first experiment using the Gamm distribution G(5,1) for lifetimes samples of size n=100,200, and second experiment using bimodal distribution of two Normal distributions (X=0.4X1~N(5,1)+0.6X2~N(3,0.5)) for lifetimes samples of size n=400,600. Moreover, to generate the censoring times for the two experiments we using Exponential distribution (EXP(6)). The second application, three real application data are applied which are liver metastases data, nursing home data, and Stanford heart transplant data.

Comparing has been made using MSE between estimation results and actual results for two simulation experiments, while for the real data, the MSE has been made between the estimation results that calculated by wavelets estimation and the kernel estimator.

CONTENTS

An explanatory table of the meaning of the symbols used in this thesis

i.i.d	Independent and identically distributed
$L^{\mathbf{2}}(R)$	Square-integrable function space (Lebesgue integrable)
DWT	Discrete wavelet transformation
CWT	Continuous wavelet transformation
$\varphi_{J,k}(x)$	Father wavelet function
$\psi_{J,k}(x)$	Mother wavelet function
V_J	Multiresolution subspace of index J
$\boxed{?}_J$	Difference Multiresolution subspace of index J
MSE	Mean square error
$\Lambda(x)$	Cumulative hazard function

$\lambda(x)$	Hazard rate function
δ,I,or 1	Indicator function
$b_{n,j}(D,x)$	Bandwidth function of kernel estimator
⍰$_{2N}$	Discrete orthogonal Haar transformation matrix
W_{dbn}	Daubechies discrete wavelet transformation matrix
⍰$_{COif4}$	Coiflet discrete wavelet transformation matrix
$\omega_{\varphi}(j_0,k)$	Approximation wavelet coefficients
$\omega_{\psi}(j,k)$	Detail wavelet coefficients
$\mathcal{P}\left(f_{\hat{\jmath}}(x)\right)$	The wavelet projection of $f(x)$ in V_J
$\omega_{\psi}^{\partial}(J,k)$	Thresholding mother wavelet coefficient
$\hat{f}^{p}(x)$	Partial probability density function estimation
$\hat{S}^{p}(x)$	Partial survival function estimation

€$_{\varphi}$	The father wavelet projection coefficient
$\boldsymbol{\Phi}$	Father wavelet matrix
⍰$_{n\times n}$	Lower triangular matrix

LIST OF TABLES

LIST OF FIGURES

LIST OF DISSERTATION RELATED PUBLICATIONS

➢ First paper

Paper Title:	Density Estimation for Right Censored Data Using Hybrid Breslow and Semi-Symmetric Wavelet
Authors:	Ali Talib Mohammed and Iden Hasan Hussein
Journal Name:	Journal of Engineering and Applied Sciences
ISSN	1816-949X
Volume	14(1), p:242-246, 2019

➢ Second paper

Paper Title:	Estimate Hazard Rate Function by Wavelet Procedures
Authors:	Ali Talib Mohammed and Iden Hasan Hussein
Journal Name:	Journal of Advanced Research in Dynamical and Control Systems – JARDCS
ISSN	1943-023X
Volume	10, No. 10, 2018

➢ Third paper

- **Paper Title:** Nonparametric Estimation for Hazard Rate Function by Wavelet Procedures with Simulation
- **Authors:** Ali Talib Mohammed and Iden Hasan Hussein
- **Acceptation:** This paper is accepted in the Second International conference of Kerbala University, College of Education for Pure Sciences (ICCEPS)
- **Journal Name:** IOP Conf. Series: Materials Science and Engineering 571 (2019) 012013
- **Doi:** 10.1088/1757-899X/571/1/012013
- **Publishing** IOP-Publishing

INTRODUCTION

The survival analysis in the branches of statistics have a great importance in the study and analysis of events that occur according to time, especially analysis of the expected time of occurrence of one or more events according to a specific period or indefinite period. Survival and reliability analysis deals with many topics, including death in biological organisms and failure in mechanical and electronic devices.

There are many questions that the survival and reliability analysis tries to give answers for it. For example, how long will be expected that mechanical devices will work according to certain conditions? The expected time of living a population group in a controlled society by influencing factors? What are the factors influencing in death or failure? And is it possible to control these factors and reduce those that do not help?. In these studies it is very important to identify the lifetime and death (failure) contexts, although death in biological organisms is considering as an event while in mechanical and electronic, failure is also an event but it may not actually occur because it can be caused by a partial failure of the machine or device.

One important concept in survival analysis is the hazard rate function, which deals with calculating the incidence of a particular event such as death or failure at a certain point in time. The hazard function can be used in many areas, such as engineering, medicine, and economic studies. This common use is due to how it is calculated based on the probability density function and the survival function, as well as the availability of valuable information on failure rates.

In statistical studies the methods of data collection takes two different approaches. The first approach depends on the collection of data which is related to certain hypotheses that enable the researcher to link the data to one of the statistics distributions and called (parametric data) while the second approach takes a method of non-hypotheses or restrictions but is free to collect data, (non-parametric data). One of the major challenges faced by researchers in statistical and other studies is dealing with non-parametric data. The important objective is how to estimate functions of these data without any information about these functions. In statistical studies, non-parametric data are one of these challenges in terms of estimating density and hazard functions. Most of the studies that investigated this aspect were based on the analysis and study of data properties. Non-parametric data include several types, one of which is the censoring data, which in turn includes types such as left censoring, right censoring, interval censoring, randomly (progressively)censoring, type I censoring, and type II censoring.

A significant property is that $X_1, X_2, \dots, X_n$ for n samples are to be independent. Kernel and nearest neighbors technical are the most popular methods to non-parametric density function. Wavelet series could be considered as good approach for function estimator since the utilize of wavelet in solving this problem gives an important advantage because it build up functions belong to $L^2(R)$. Wavelets are considering as a new class of functions that are well localized in time and frequency. Moreover, the wavelet is rapidly decaying wave like oscillation that has zero mean and it exists for the finite duration. The wavelet transformations could be used in two types of discrete wavelet transformation (DWT) and continuous wavelet transformation (CWT). The method that will be displayed focus on (DWT).

Approximation and estimation of functions is one of the important and good uses of wavelets.

The content of this thesis includes five chapters, chapter one contains the literatures review, the aims of this study, and some basic concepts in statistics such as statistical functions and properties. Chapter two contains Fourier series and transforms, definitions of wavelets with types and characteristics, and methods of building wavelets. The content of chapter three will address the proposed method in this study in order to estimate the hazard function using wavelet, the application takes two types in this thesis simulation applications and real, all this includes in chapter four and the final one is chapter five contain the conclusion and recommendations.

CHAPTER

ONE

INTRODUCTION AND BASIC CONCEPT

1.1 Introduction

In statistic, the importance of estimators in the information provided and the calculation of the estimates based on measurements and information in the sample subject of the test. In censoring data, especially with the type of randomly censoring data which is considered to have that have wide applications in the medical fields. This chapter describes and surveys the aims of this study and literature review as well as presents some basic concepts, for example hazard rate function, types of censoring data, and some nonparametric estimation methods such as: Kaplan–Meier, Nelson-Aalen, Breslow, and Kernel estimators.

1.2 The aims of this thesis

The main objective of this thesis is estimating the Hazard rate function using linear wavelet transformation, where the estimation algorithm consists of two main parts, including the first estimate of the probability density function f(x) and the second estimate of the survival function s(x). The structure of the algorithm will rely on the projection property of the father wavelet function $\{\Phi_{J,k}(x)\}, 0 \leq k \leq 2^J - 1, J \geq 0$ on the subspace V_J of Lebesgue spaces $L^2(R)$. The data type is an important factor in the estimation process so data will be used is randomly right censoring data. Real and simulation applications will be used; in fact three real and two simulation applications are going to use to estimate the hazard rate function. Finally, the global main error (MSE) will be applicate to comparing between the result values.

1.3 Literature review

In the previous century especially in the 1990s many researchers widespread use of wavelets in the fields of statistical estimates. First used of the wavelets in statistics were introduced by a collection of articles such as Donoho et al. (1993), Kerkyacharian and Picard (1993), and Donoho and Johnstone (1994, 1995).

In 1990, Antoniadis, Grégoire and Nason [7] estimated hazard function based on divided the time into time intervals and then calculate the events in each interval separately with the survival function by linear wavelet estimator, then got the hazard function based on the ratio.

In 2003, Wu and Wells [59] proposed an estimator of hazard rate with nonparametric based on wavelets of increases cumulative Nielsen-Aalen estimator of hazard rate function, this estimate based on using of non-linear wavelet. Moreover, the introduced a new form of the mean integrated squared error (MISE).

In 2005, Brunel-Piccinini and Comte [10] presented a projection wavelet pressure to estimated density and hazard rate functions, two types to estimate the hazard rate function were applicate, first one was two-steps as a ratio of density and survival function, second type found estimated the hazard rate directly with using in both types penalized linear wavelet projection.

In 2005, Liang, Mammitzsch and Steinebach [41] used a non-linear wavelet to estimate density and hazard rate functions with assumption that hazard function belongs to Besov space B_{pq}^{s}, for independent and identically distributed (i.i.d) data in complete case.

In 2005, Bezandry, Bonney and Gannoun [9] considered a linear wavelet method to estimate the density function and hazard rate with established the weak uniform consistency of the density and hazard rate estimators. They were proved that there was uniform convergence in probability of the wavelet estimators of the density and hazard rate functions.

In 2007, Angers and MacGibbon [4] used a monotone wavelet to develop a non-parametric Bayesian functional estimation method to estimate the hazard function of censoring data of type randomly right.

In 2009, Doosti [25] used linear wavelet survival function estimator for independent and identically distributed (i.i.d) of real random variables to estimate the survival function that assumed it belong to Besov space.

In 2009, Plancade [48] presented an estimator of hazard rate function in censoring case based on unknown quantities in two models of estimators Non adaptive and adaptive, where these estimators predicated on a regression-type.

In 2010, Kim *et al.* [38] proposed local linear estimation of conditional hazard function in right censored data, the base of estimator used the kernel methods with positive optimal bandwidths h_1 and h_2.

In 2010, Chaubey *et al.* [13] introduced an estimate of the derivative of density function using linear wavelet under right randomly censoring data and extended the results regards to the asymptotic convergence rates.

In 2012, Ahmadi *et al.* [3] used Haar wavelet smoothing and Kernel smoothing methods to estimate hazard rate function for gastric cancer patients in Fayazbakhsh hospital in Tehran.

In 2012, Abbaszadeh, Chesneau and Doosti [1] estimated the density function under bias and multiplicative censoring using two pressures which are linear and nonlinear (hard) wavelet methods of the Daubechies wavelets db2N.

In 2014, Afshari and Tahmasebi [2] estimated the hazard rate function in two steps, using the linear wavelet to estimator the probability density function, and then used Haar transformation integrate to find the survival function and the ratio between them given hazard rate function, and evaluate of mean integral square error with convergence ratio.

In 2014, Salha, Ahmed and Alhoubi [52] used a new kernel estimated based on Weibull kernel to estimate the density and hazard functions, and showed that the smoothing parameter (h) was The influencing factor on the bias which is closed to zero when $h \to 0$.

In 2016, Chesneau and Doosti [15] developed a new adaptive estimator g(x, m) based on wavelet methods of multivariate discrete and continuous density function, prove its good theoretical performance by determining sharp rates of convergence under the L_p risk with p ≥ 1 1for a wide class of unknown conditional density.

In 2017, Comte, Samson and Stirnemann [20] proposed a non-parametric estimation of the hazard function depended on regression contrast minimized in a finite dimensional functional space generated by splines bases.

In 2017[30], Grez and Vidakovic [30] estimated the density function based on linear orthogonal projection periodic wavelet estimator onto a multiresolution space V_J using empirical wavelet coefficients related to Kaplan-Meier estimator, for randomly right censoring data.

1.4 Hazard rate function and its properties

In this section we will discuss and define one of the functions of survival analysis, which is the hazard rate function, which describes the risk rate of the random variable.

Before giving the definition of hazard rate function, some functions that describe lifetime must be presented such as:

- The cumulative distribution functions $F(x)$ or knows as lifetime distribution function and abbreviated written as CDF.

$$F(x) = \Pr(X \leq x) = \int_0^x f(x)dx$$

- The probability density functions $f(x)$ or known as failure density and abbreviated written as PDF.

$$f(x) = \frac{d\big(F(x)\big)}{dx} = F'(x)$$

- The survival functions $S(x)$ or knows as complementary cumulative distribution.

$$S(x) = Pr(X > x) = 1 - F(x)$$

1.4.1 Hazard rate function [2]

In survival analysis studies, the hazard rate function also known as failure rate function is defined as the conditional probabilistic condition of the individual to test a particular event (death or relapse) within a certain time period or predetermined.

Mathematically hazard rate function of non-negative random variable X defines as a ratio between the probability function $f(x)$ and survival function $S(x)$, and denoted by $h(x)$ or $\lambda(x)$.

$$\lambda(x) = \lim_{dx \to 0} \frac{Pr(x \leq X < x + dx)}{dx \cdot S(x)} = \frac{F'(x)}{S(x)} = \frac{f(x)}{S(x)} \qquad (1\text{-}1)$$

It is possible to form the hazard rate function according to the survival function as follows:

$$\lambda(x) = -\frac{d}{dx}\left[\ln\left(S(x)\right)\right] \qquad (1\text{-}2)$$

If the random variable X has distribution function $F(x)$ and density function $f(x)$, X is said to be have an increasing (decreasing) hazard rate function $\lambda(x)$, if $\lambda(x)$ is increasing (decreasing) for all x.

1.4.2 Cumulative hazard function [50]

The total number of failures, deaths or relapses over a period of time $[0, x]$ is define as cumulative hazard function denoted by $\Lambda(x)$, and It is an integral part of the hazard rate function, according that for $x \geq 0$:

$$\Lambda(x) = \int_0^x \lambda(v)dv \tag{1-3}$$

$$\Lambda(x) = -\ln\big(1 - F(x)\big) = -\ln\big(S(x)\big) \tag{1-4}$$

$$S(x) = e^{-\Lambda(x)} \tag{1-5}$$

$$f(x) = \lambda(x)e^{-\Lambda(x)} \tag{1-6}$$

1.5 Censoring data [49]

The general concept of censoring in statistical data can be defined as information about the time of occurrence of the event, but the exact amount of time is not known.

There are three main types of censoring data:

- Type I censoring data:

In this type the numbers of items or individuals which inter the experiment are random variables, but the lifetimes (failure times) which enter the experiment are fixed.

Type I occurs when determining the time it takes to experiment in advance and then follow the individuals of the experiment to the end of time and then observe the occurrence of the event or not for the individuals.

- Type II censoring data:

In this type the numbers of items or individuals which inter the experiment are specified and determined, but the lifetimes (failure times) which enter the experiment are random variables.

Type II occurs when the event occurrence of a number of predetermined individuals and then a failure observing whether or not for all individuals depending on the type of observation (right, left).

- Randomly (Progressively) censoring data:

In this type the numbers of items or individuals which inter the experiment are random variables, and the lifetimes (failure times) which enter the experiment are random variables too.

This type occurs when each individual has a specific time that statistically independent of the individual failure time, and the observed value is the minimum (maximum) of the censoring and failure times according to the right (left) censoring state.

Note: It should be noted that the type of data to be studied is of random type, so the next definition is a detailed definition.

1.5.1 Randomly right censoring data [4]

Let $X_1, X_2, \dots, X_n$ be an independent and identically distributed (i.i.d.) survival times with unknown density and cumulative functions f and F respectively. Let $C_1, C_2, \dots, C_n$ be i.i.d. censoring times unknown density and cumulative functions g and G respectively, it is presumed that for

$i = 1,2, \dots, n \; X_i and \; C_i$ are typically statistical independence. The observing is for $\{Z_i , \delta_i\}$ which is an i.i.d. sequences, such that:

$$Z_i = \min\{X_i, C_i\}$$

The indicator function (δ) defines:

$$\delta_i = \begin{cases} 1 \;\; if \; X_i \leq \; C_i \\ 0 \;\; if \; X_i > C_i \end{cases}$$

1.5.2 <u>Hazard rate function in censoring case [7],[30]</u>

Because of the observing in the case or randomly right censoring data is the pair $\{Z_i , \delta_i\}_{i=1}^{n}$, the target here is to get the joint distribution of the pair $\{Z , \delta\}$, it is possible to get:

$$\Pr(Z \leq z, \delta = 1) = \Pr(\min\{X, C\}, X \leq C)$$

$$= \int_0^z \Pr(C \geq x) \, f(x) dx$$

$$\Pr(Z \leq z, \delta = 1) = \int_0^z (1 - G(x)) f(x) dx \qquad (1\text{-}7)$$

$$\Pr(Z \leq z, \delta = 0) = \Pr(\min\{X, C\}, X > C)$$

$$= \int_0^\infty \Pr(C \leq min\{x, z\}) f(x) dx$$

$$= \int_0^z \Pr(C \leq x) f(x) dx + \int_z^\infty \Pr(C \leq z) f(x) dx$$

$$= \int_0^z G(x)f(x)dx + G(z)\int_z^\infty f(x)dx$$

$$\Pr(Z \le z, \delta = 0) = \int_0^z G(x)f(x)dx + G(z)\big(1 - F(z)\big) \qquad (1\text{-}8)$$

From (1-7) and (1-8) one can get:

$$f_{Z,\delta}(z,\delta) = \big(f(z)\big)^{\delta}\big(1 - G(z)\big)^{\delta}\big(g(z)\big)^{1-\delta}\big(1 - F(z)\big)^{1-\delta} \qquad (1\text{-}9)$$

Similarly, from (1-9) it’s possible to find the marginal density function of complete data Z as follows:

$$f_Z(z) = f_X(z)\big(1 - G_C(z)\big) + g_C(z)\big(1 - F_X(z)\big) \qquad (1\text{-}10)$$

Now let, $L(z) = \Pr(Z \le z)$

It could be got from that the following:

$$1 - L(z) = \big(1 - F_X(z)\big)\big(1 - G_C(z)\big) \qquad (1\text{-}11)$$

Finally, the hazard rate function of complete data Z can be formed as follows:

$$\lambda_Z(z) = \frac{f_Z(z)}{1 - L(z)} \qquad (1\text{-}12)$$

1.6 Non-parametric estimation methods

Non-parametric estimator is a statistical method that provides access to a functional form that is appropriate for non-parametric data. This section will review four of them which are: Kaplan–Meier, Nelson-Aalen, Breslow, and Kernel estimators.

1.6.1 Kaplan–Meier estimator [37]

It can also be called as product limit estimator, which can be used to estimate the survival function $S(x)$ for non-parametric data. The mathematical form to estimate $S(x)$ using Kaplan-Meier estimator is given by:

$$S(x) = \prod_{i:x_i \leq x} \left[1 - \frac{d_i}{n_i}\right]$$

That is, each term in the product is the conditional probability of survival beyond time x_i, meaning the probability of surviving beyond time x_i, given the subject has survived up to time x.

1.6.2 Nelson-Aalen estimator [49]

The Nelson-Aalen estimator is a non-parametric statistical method to estimate the cumulative hazard function $\Lambda(x)$ for non-parametric censoring data. The mathematical form to estimate $\Lambda(x)$ using Nelson-Aalen estimator is given by:

$$\widehat{\Lambda}(x) = \sum_{i:x_i \leq x} \frac{d_i}{n_i}$$

d_i: The number of events occurs at x_i.

n_i: The total individuals at risk at x_i.

1.6.3 Breslow estimator [50]

The Breslow estimator is considering as a method to estimate the baseline survival function $\hat{S}(x)$. The mathematical form to estimate $\hat{S}(x)$ using Breslow estimator is given by:

$$\hat{S}(x) = Exp\left(-\widehat{\Lambda}(x)\right)$$

Where, $\widehat{\Lambda}(x)$ is the cumulative hazard function.

1.6.4 Kernel estimator [49]

The general formula of kernel estimation method of hazard rate function for censored data case given as follows:

$$\hat{\lambda}(x) = \sum_{j=1}^{n} \left(\frac{\delta_j}{n-j+1}\right)\left(\frac{1}{b_{n,j}(D,x)}\right) \cdot K\left(\frac{x-X_j}{b_{n,j}(D,x)}\right), \qquad x \geq 0$$

Where $b_{n,j}(D,x) > 0$ which represents the bandwidth functions, and δ_j denoted the censoring indicator.

CHAPTER TWO

FOURIER AND WAVELET TRANSFORMATIONS

2.1 Introduction

Wavelet is one of the most important mathematical and statistical tools in the past 20 years and its applications are increasing day after day, it is used in a wide range of applications. The most important of these applications are: the description of non-linear signals, non-parametric estimations, data smoothing and decomposition of images, and other applications. The statistical alphabet has always been that the best input to the wavelet is by describing the Fourier Series (FS), clarifying its types, introducing its transformations and inverses of these transformations, as will be explained in the following sections.

This chapter will contain two main sections: section one consist general introduction on Fourier series and Fourier transform with a description of the matrix and inverse matrix (IFT) transforms and its mathematical formulas. Section two will provide a general introduction to the wavelets with continuous (CWT) and discrete (DWT) wavelet transforms

2.2 Fourier series [18]

The Fourier series, presented by French scientist Joseph Fourier, is one of the most important mathematical tools used in the different fields of mathematics, statistics and engineering. The benefit of the Fourier series is its ability to transform and rewriting any mathematical function in terms (to be addressed later) through infinite sum the sines and cosines.

For any periodic function $f \in L^2\ [-\pi, \pi]$ with period 2π, the general form of Fourier series is given by:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k cos(kx) + b_k sin(kx) \right) \qquad (2\text{-}1)$$

Where a_0, a_k, and b_k can be found as follows:

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) cos(kx) dx$$

$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) sin(kx) dx$$

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) dx$$

Note that: $L^{\mathbf{2}}(R) = \left\{ f : R \to R, \int_{-\infty}^{\infty} |f(x)|^2 dx < \infty \right\}$

2.3 Some special formulae for the Fourier series [47]

This section presents some special formulae of the Fourier series.

2.3.1 Even Fourier series

In statistic, this function is called harmonic cosine function and it is also symmetric on y axis, which takes the form as follows:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k cos(kx) \right) \qquad (2\text{-}2)$$

where: $a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) cos(kx) dx$

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) dx$$

2.3.2 Odd Fourier series

In statistic, this function is called harmonic sine function and it is also symmetric on X axis, which takes the form as follows:

$$f(x) = \sum_{k=1}^{\infty} \left(b_k sin(kx)\right) \qquad (2\text{-}3)$$

where:

$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) sin(kx) dx$$

2.3.3 Even harmonic Fourier series

This function takes the formula as follows:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_{2k} cos(2kx) + b_{2k} sin(2kx)\right) \qquad (2\text{-}4)$$

where a_0, a_{2k}, and b_{2k} can be found as follows:

$$a_{2k} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) cos(2kx) dx$$

$$b_{2k} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) sin(2kx) dx$$

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) dx$$

2.3.4 Odd harmonic Fourier series

This function takes the formula as follows:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_{(2k-1)} cos\big((2k-1)x\big) + b_{(2k-1)} sin\big((2k-1)x\big) \right) \quad \text{(2-5)}$$

Where a_0, a_{2k}, and b_{2k} can be found as follows:

$$a_{(2k-1)} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) cos\big((2k-1)x\big) dx$$

$$b_{(2k-1)} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) sin\big((2k-1)x\big) dx$$

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) dx$$

2.4 Fourier transformation [18],[29]

A Fourier transform can be defined as a tool by which the length or waveform (function or signal) can be changed to an alternative mathematical appearance can be represented by sine and cosine functions. Mathematical alphabets have always identified this transformation from time domain to frequency domain.

2.4.1 Time and Frequency domain

Mathematically, the time domain defined as the mathematical analysis of the different functions taking into account the real time of the function. In other words, the time domain can be considered as the real function domain. The frequency domain is defined as the mathematical analysis of the different functions taking into account the real frequency of the function. In other words, the frequency domain is the real domain of the sine and cosine functions.

2.4.2 Dirichlet conditions

Dirichlet put four sufficient conditions for any function f(x) over periodic T that can be expanded as Fourier series, and they are as follows:

I. f(x) is a signal value function and absolutely integrable function.

II. In each periodic T, f(x) has a finite number of discontinuous points.

III. In each periodic T, f(x) has a finite number of maximum and minimum points.

IV. $\int_0^T |f(x)|^2 < \infty$.

2.4.3 The continuous Fourier transform

For any continuous and integrable function f(x), it is able to find the continuous Fourier transform for f(x), as follows:

$$\digamma(u) = \int_{-\infty}^{\infty} f(x) \cdot EXP(-2i\pi ux)dx \qquad (2\text{-}6)$$

The results of this transform is a function in f or $\digamma(u)$ and called the function of the spectrum of the function f.

In order to find f(x) from $Ƒ(u)$, it could be used the inverse continuous Fourier transforms, as follows:

$$f(x) = \int_{-\infty}^{\infty} Ƒ(u) \cdot EXP(2i\pi ux)du \qquad (2\text{-}7)$$

It is worth to know that:

- $Ƒ(u)$ is complex function, and can be written as: $Ƒ(u) = R(u) + i\ Im(u)$
- The Fourier spectrum of f(x) is $\|Ƒ(u)\| = \sqrt{\left(R(u)\right)^2 + \left(Im(u)\right)^2}$.
- The power Fourier spectrum is $\|Ƒ(u)\|^2 = \left(R(u)\right)^2 + \left(Im(u)\right)^2$
- The phase angle is $\theta = \tan^{-1}\left(\frac{Im(u)}{R(u)}\right)$.

2.4.4 The discrete Fourier transform

The discrete Fourier transform of continuous function f(x) is defined as:

$$Ƒ(u) = \sum_{x=0}^{N-1} f(x) \cdot EXP\left(\frac{-2i\pi ux}{N}\right) \qquad (2\text{-}8)$$

where, $x = 1,2, \dots, N-1$ and $N = 2^n, n \in \mathbb{N}$.

In order to find f(x) from $Ƒ(u)$, it could be used the inverse discrete Fourier transforms, as follows:

$$f(x) = \sum_{u=0}^{N-1} \mathrm{F}(u) \cdot EXP\left(\frac{2i\pi ux}{N}\right) \tag{2-9}$$

2.4.5 The n-point discrete Fourier transform

The n-point discrete Fourier transform is define as $(N \times N)$ matrix of complex numbers denoted by (F_N) and define as follows:

$$\mathrm{F}_N = \left[(Z_N)^{jk}\right]_{0\le j,k\le N-1} = \begin{bmatrix} 1 & 1 & & 1 \\ 1 & Z_N^1 & \cdots & Z_N^{(N-1)} \\ \vdots & & \ddots & \vdots \\ 1 & Z_N^{(N-1)} & \cdots & Z_N^{(N-1)(N-1)} \end{bmatrix} \tag{2-10}$$

where, $N \ge 2, N \in \mathbb{N}$ and $Z_n = EXP\left(\frac{-2i\pi}{N}\right)$.

2.4.6 The inverse n-point discrete Fourier transform

The inverse n-point discrete Fourier transform is denoted by F_N^{-1} and define as:

$$\mathrm{F}_N^{-1} = \frac{1}{N}\,\mathrm{F}_N^{*} \tag{2-11}$$

where, F_N^{*} is the adjoint (transpose and complex conjugate) of F_N.

2.5 Wavelet and Multiresolution analyses [36]

This section will be presented in some detail to the mathematical construction of wavelets as well as the continuous and discrete wavelet transformations, and to access these subjects we will first give a detailed view of the space $L^2(R)$ from the perspective of the multiresolution analysis.

2.5.1 The multiresolution of $L^2(R)$ [51],[42]

The multiresolution is a tool to describe the construction of spaces and to give an analytical description of the components and bases of these spaces. Let's first give the definition of a square integrable function or known as the space of Lebesgue measurable functions, which denoted as $L^2(R)$ and defined:

$$L^2(R) = \left\{ f: R \to R;\ \int_{-\infty}^{\infty} |f(x)|^2 < \infty \right\}$$

The multiresolution analysis of the space $L^2(R)$ is defined as nested sequences of closed (resolution) subspaces $\{V_j\}, j \in \mathbb{Z}$, and have the following properties:

- $V_j \subset V_{j+1}, j \in \mathbb{Z}$.
- $f(x) \in V_j$ if and only if $f(2x) \in V_{j+1}$.
- $f(x) \in V_0$ if and only if $f(x - \square) \in V_0, \square \in \mathbb{Z}$.
- $\overline{\cup_{J \in Z} V_j} = L^{\mathbf{2}}(R)$.
- $\cap_{J \in Z} V_j = 0$.
- $V_{-\infty} = \{0\}$ and $V_{\infty} = L^{\mathbf{2}}(R)$.

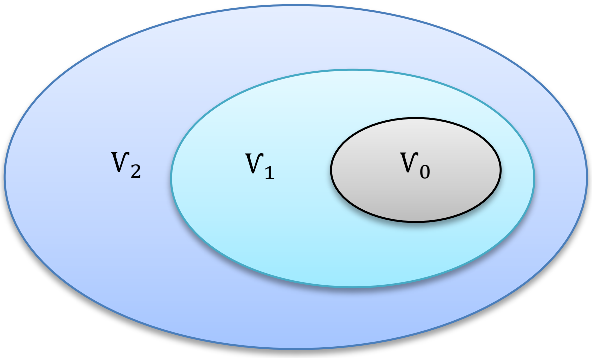


Figure (2-1): multiresolution subspaces

Now, define another subspace of $L^{\mathbf{2}}(R)$ which denoted as $⍰_j, j \in \mathbb{Z}$ and define as:

$$⍰_j = V_{j+1} \ominus V_j$$

Also, $⍰_j$ known as difference subspaces, generally we have the following:

- $V_1 = V_0 \oplus ⍰_0$
- $V_2 = V_1 \oplus ⍰_1 = V_0 \oplus ⍰_0 \oplus ⍰_1$
- $V_j = V_0 \oplus \left(\oplus_{\boldsymbol{i=0}}^{\boldsymbol{j-1}} ⍰_i \right)$

2.5.2 Wavelet functions [33]

The wavelet is one of the types of the mathematical functions used to divide the given function to different frequency compound and study each compound with an appropriate solution at each measurement. Mathematically, the wavelet is defined as a real value function on the real axis and oscillates up and down regularly around zero.

Any wavelet function $\left(ʄ(u)\right)$ must satisfy the following three conditions:

1. $\int_{-\infty}^{\infty} \mathfrak{f}(u)du = 0$, this condition ensures that the wavelet function vibrations must balance higher and lower zero.
2. $\int_{-\infty}^{\infty} \big(\mathfrak{f}(u)\big)^2 du = 1$, this property is called unit energy, and for any $0 < \mathcal{E} < 1$ there exist $(-M, M)$ such that: $\int_{-M}^{M} \big(\mathfrak{f}(u)\big)^2 du > 1 - \mathcal{E}$. This condition ensures that most of waving's contained in the portion of interval and for specific width.
3. The wavelet function $\mathfrak{f}(u)$ has ability to admissible.

2.5.3 Wavelet transformations [36]

The wavelet expression indicates the meaning of the little wave, which contains the least oscillations, and will fast decay to zero in both positive and negative directions, this is the admissibility condition for the function that requires the wavelet transformation. The first definition for wavelet was introduced by Jean Morlat in 1980.

The wavelet group is used to approximate the signal in order to find a set of wavelet subsets which will be built from expansion or compression and shifting of the original wavelet, which represent the signal or data to be analyzed. This process is in short of the transformation from large measurements to accurate measurements by paving these data or signals. This is exactly the same as the kernel methods, which was discussed by many statisticians in the second half of the 20th century and beyond. The main outcome of the transformation process is the mother wavelet function, which is described below as:

$$\Psi_{a,b}(x) = \frac{1}{\sqrt{a}} \Psi\left(\frac{x-b}{a}\right) \tag{2-12}$$

There are two types of wavelet transformations, which are continuous and discrete wavelet transformations.

2.5.3.1 Continuous wavelet transformation (CWT) [21],[42]

The mathematical concept of CWT is to divide the continuous function in time range into a set of wavelets. This transformation will have the possibility to build a common signal representation (data) between the time band and the frequency band, which will give a prelude to data in both time and frequency range. The mathematical formula for this transformation is:

$$\text{⍰}_c\{\text{Ƒ}(a,b)\} = \int_{-\infty}^{\infty} \text{Ƒ}(x)\frac{1}{\sqrt{a}}\Psi\left(\frac{x-b}{a}\right)d(x) \qquad (2\text{-}13)$$

$\Psi(\cdot)$ represents a continuous function in the time and frequency range together and is also known as the mother wavelet function.

the mathematical formula of the inverse continuous wavelet transformation (ICWT) will be as follows:

$$\text{Ƒ}(x) = (\text{Ƈ}_{\Psi})^{-1}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\text{⍰}_c\{\text{Ƒ}(a,b)\}\frac{1}{\sqrt{a}}\breve{\Psi}\left(\frac{x-b}{a}\right)\frac{db\,da}{a^2} \qquad (2\text{-}14)$$

Where, $\breve{\Psi}(\cdot)$ is dual function and Ƈ_{Ψ} could be found as follows:

$$\text{Ƈ}_{\Psi} = \int_{-\infty}^{\infty}\frac{\overline{\left(\breve{\Psi}(t)\right)}}{|t|}\left(\widehat{\breve{\Psi}(t)}\right)dt \qquad (2\text{-}15)$$

$\left(\widehat{\breve{\Psi}(t)}\right)$ is indicates the Fourier transform that was previously mentioned.

The two most famous wavelet functions of are compatible with the CWT are:

1. Mexican Hat wavelet function

$$\Psi(t) = (1 - t^2) \cdot EXP\left(\frac{-t^2}{2}\right)$$

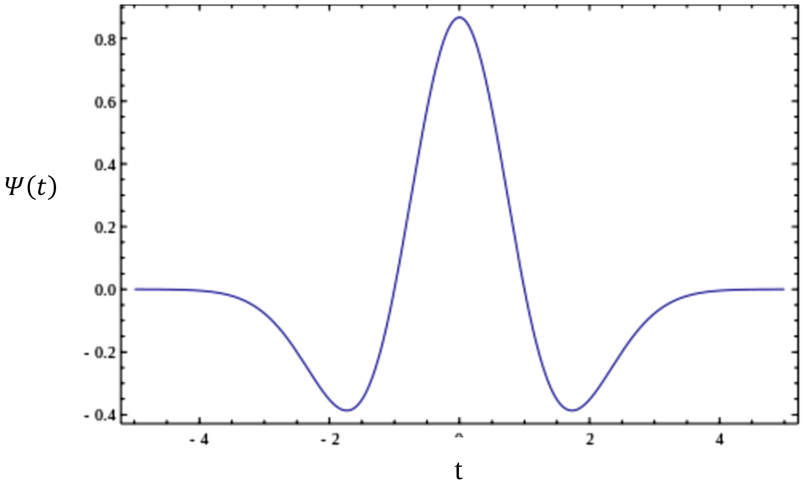


Figure (2-2): Mexican Hat wavelet function

2. Morlet wavelet function

$$\Psi(t) = EXP(i\varpi_0 t) \cdot EXP\left(\frac{-t^2}{2\sigma_0^2}\right)$$

Such that ϖ_0 is the frequency and $\sigma_0$is a measure of the support.

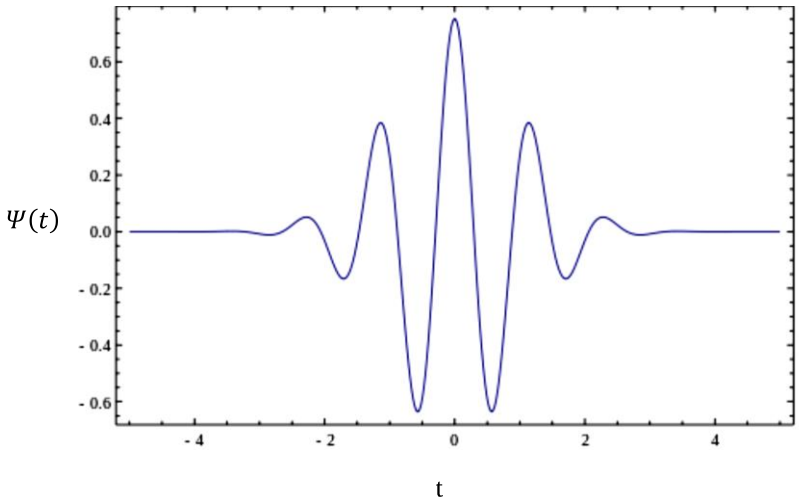


Figure (2-3): Marlet wavelet function

2.5.3.2 Discrete wavelet transformation (DWT) [27],[46],[42]

DWT is considered one of the most famous and application of the wavelet transformation tools in various fields of engineering, mathematics, statistics and other applications. The input and output of this transformation is discrete data and simulates the discrete Fourier transformation in procedure. Where the data is transform from the time range (original data field) to wavelet domain where the results will be vector-shaped and of the same original vector size. DWT can be expressed in linear equations and also by matrices as see below: (Van Fleet, 2011)[27].

Let Ƒ(x) be a function defines on equally-spaced observations, the DWT can be calculated using the following relationship:

$$\square_{D}\{Ƒ(x)\} = \sum_{k\in\mathbb{Z}} \square_{\Phi}(j_0,k)\Phi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_{k\in\mathbb{Z}} \square_{\Psi}(j,k)\,\Psi_{j,k}(x) \qquad (2\text{-}16)$$

Where $j < j_0$, $j, j_0 \in \mathbb{Z}$, and:

⍰$_{\Phi}(j_0, k) = \frac{1}{n}\sum_x \text{Ƒ}(x)\ \Phi_{j_0,k}(x)$ is called "Approximation" coefficients.

⍰$_{\Psi}(j, k) = \frac{1}{n}\sum_x \text{Ƒ}(x)\ \Psi_{j,k}(x)$ is called "Detail" coefficients.

The functions $\Phi(\cdot)$ and $\Psi(\cdot)$ are known as father and mother wavelet functions, and define as:

$$\Phi_{j,k}(x) = 2^{j/2}\Phi\left(2^j x - k\right) \qquad (2\text{-}17)$$

$$\Psi_{j,k}(x) = 2^{j/2}\Psi\left(2^j x - k\right) \qquad (2\text{-}18)$$

In matrices form, the difference here about the Fourier transform is to describe the matrix of the wavelet transform that will not be a single constant matrix as in Fourier transform where transform matrices will vary according to their associated functions, and we will explain that with three examples as follows:

1. <u>Haar wavelet [39]</u>

Haar wavelet considering as the simplest and oldest that used in wavelet transformation, it is based on Haar series which was presented by Alfred Haar in 1909. The mother and father Haar wavelet functions are defined as:

$$\Psi(x) = \begin{cases} 1 \quad if \quad 0 \le x < {}^{1}\!/_{2} \\ -1 \quad if \quad {}^{1}\!/_{2} \le x < 1 \\ 0 \qquad otherwise \end{cases} \qquad (2\text{-}19)$$

$$\Phi(x) = \begin{cases} 1 & if \quad 0 \le x < 1 \\ 0 & otherwise \end{cases} \qquad (2\text{-}20)$$

The orthogonal Haar matrix for DWT is taking the form:

$$\square_{2N} = \frac{1}{2} \cdot \begin{bmatrix} \mathrm{H}_N \otimes & [1,1] \\ I_N \otimes & [\sqrt{2}, -\sqrt{2}] \end{bmatrix} \qquad (2\text{-}21)$$

The symbol$\otimes$ is representing the Kronecker product.

Where I_N is the identity matrix of degree $(N \times N)$, and H_N is define as follows:

$$\mathrm{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

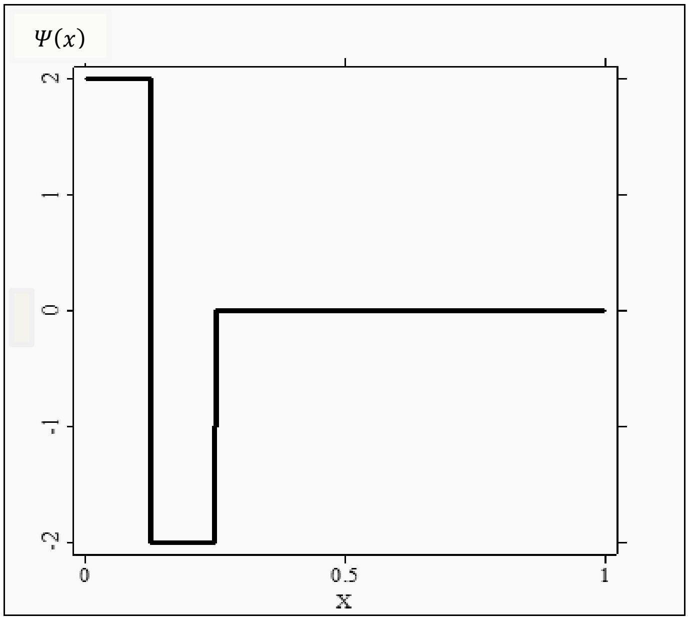


Figure (2-4): Haar mother function

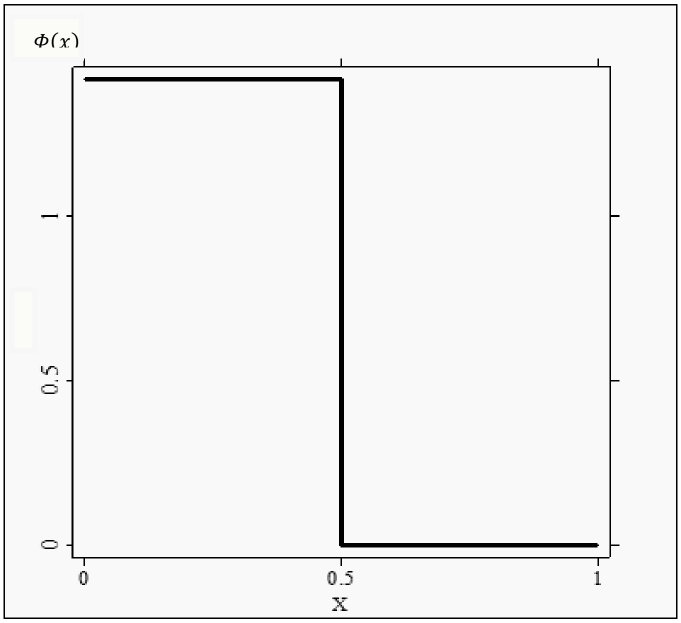
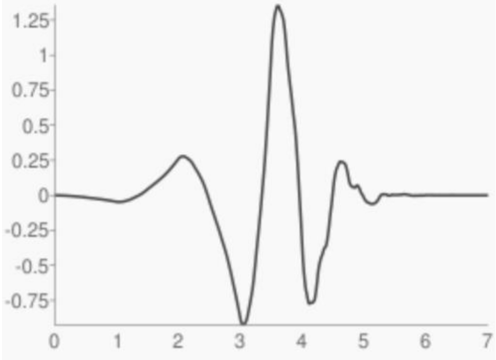
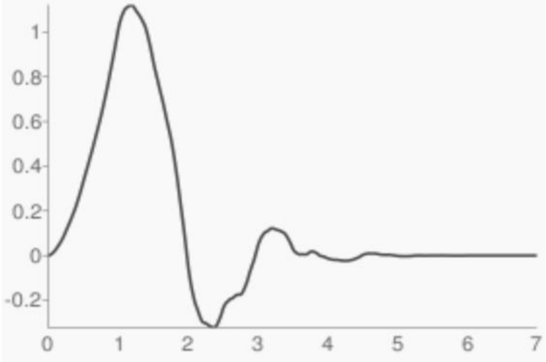
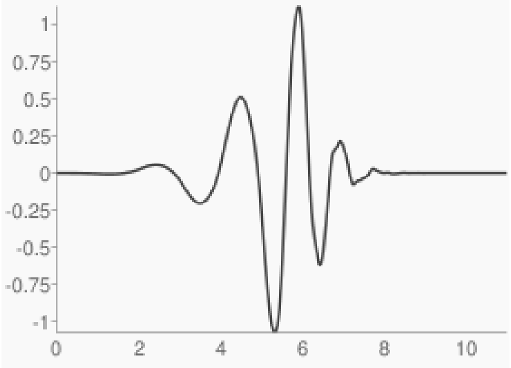
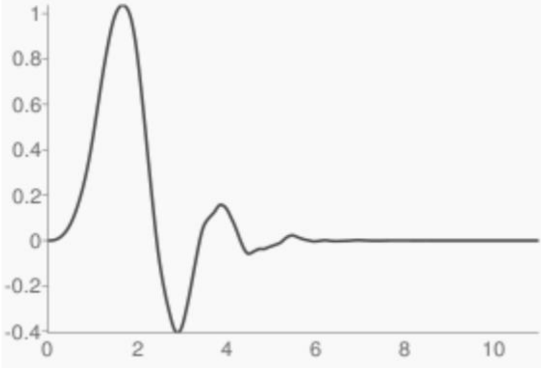
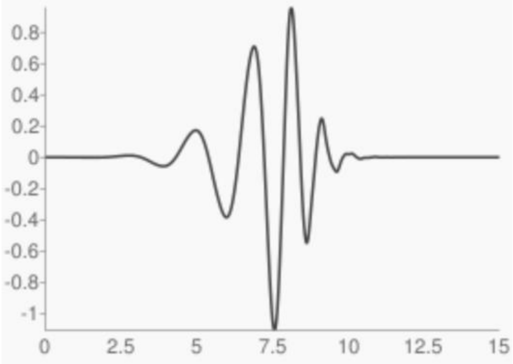
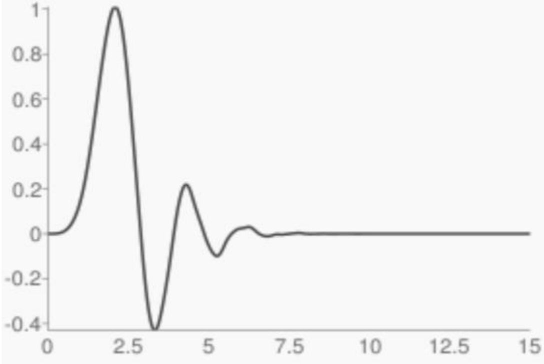
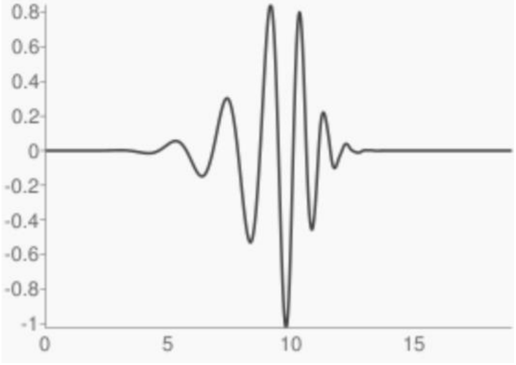
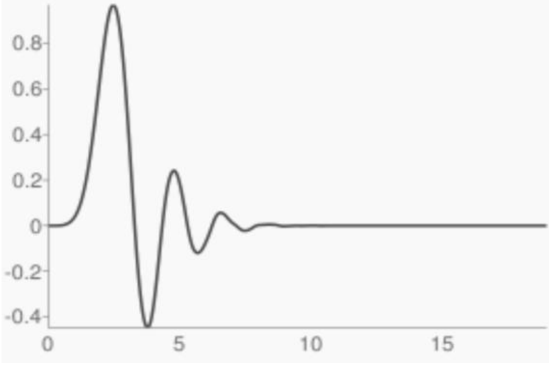


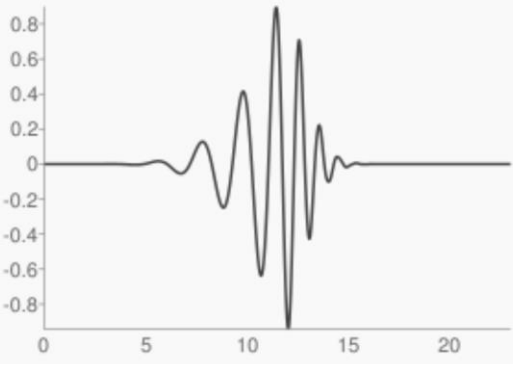
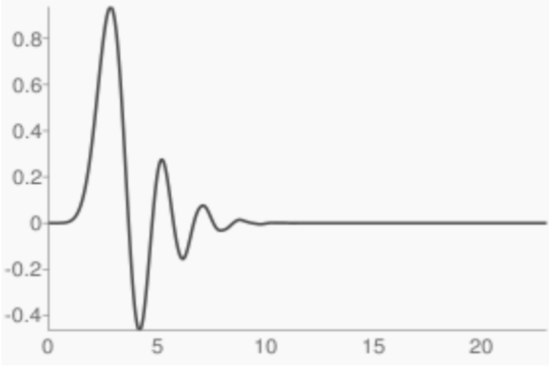
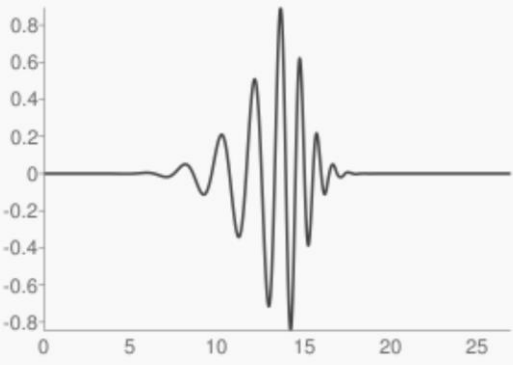
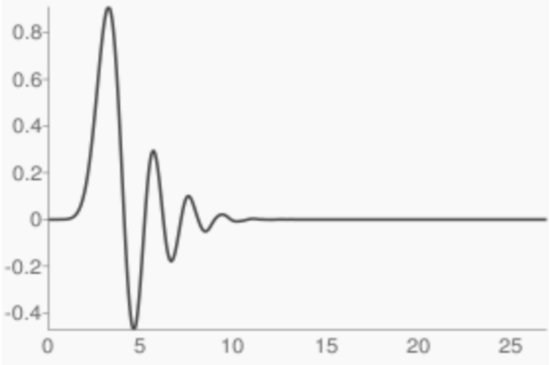
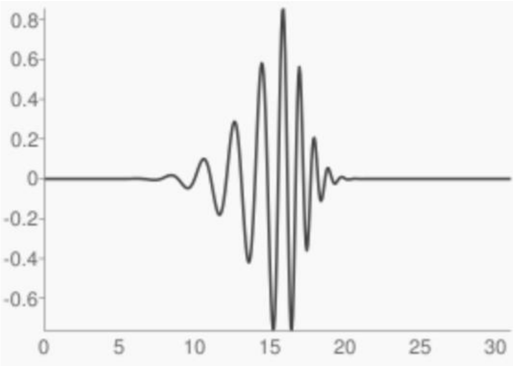
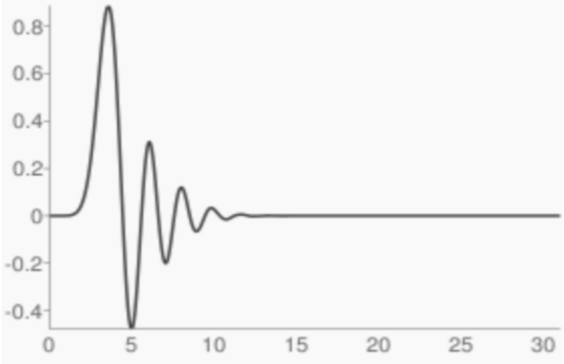
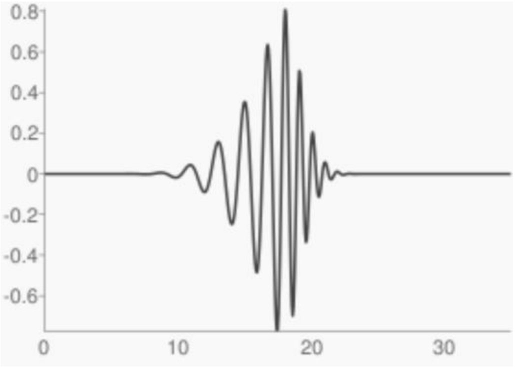
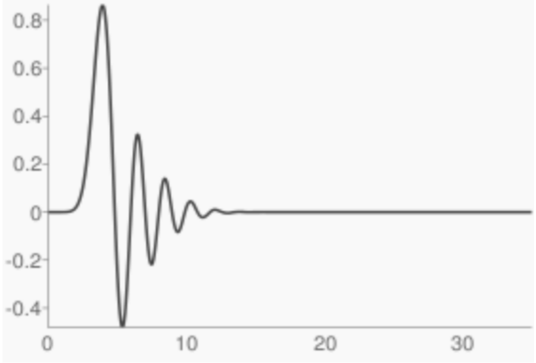
Figure (2-5): Haar father function

2. Daubechies wavelet[22]

Daubechies wavelet is an orthogonal discrete wavelet and was presented by Belgian mathematician Ingrid Daubechies in (Daubechies, 1992)[22] as a coronation for her advanced research on wavelets that have vanishing points. The Daubechies wavelet functions denoted according to how many vanishing points its own it, and denoted by db2,db4, … , db50.

It is worth mentioning the wavelet function $\Psi(\cdot)$ is said to have a (m) number of vanishing points if $\int x^m \,\Psi(x)dx = 0$, each wavelet has a number of zero moments, vanishing points or vanishing moments equal to half the number of coefficients. For example, D2 (the Haar wavelet) has one vanishing moment, D4 has two, etc. A vanishing moment limits the wavelets ability to represent polynomial behavior or information in a signal.

	Mother daubechies wavelet $\Psi(x)$	Father daubechies wavelet $\Phi(x)$
db4		
db6		
db8		
db10		

db12		
db14		
db16		
db18		

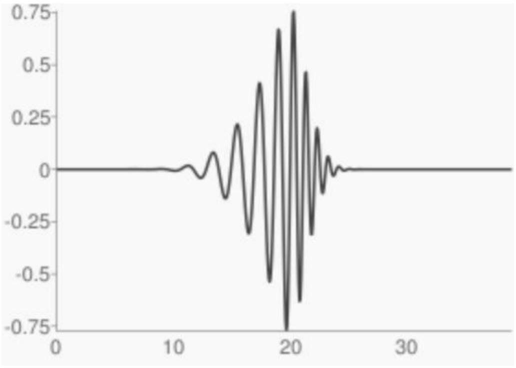
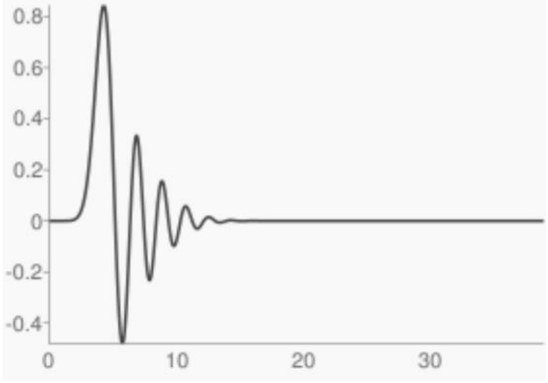
db20		

Figure (2-6): Mother and father Daubechies wavelet functions

Ingrid Daubechies has shown several mathematical equations, from which we will need the wavelet transformations matrix, and we will describe two of them, db4 and db6, where the number of parameters in db4 and db6 are only eight and six respectively.

The db4 parameters are:

$$\alpha_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, \alpha_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}, \alpha_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, \alpha_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}$$

$$\beta_0 = \alpha_3, \beta_1 = -\alpha_2, \beta_2 = \alpha_1, \beta_3 = -\alpha_0$$

The db4 discrete wavelet transformation denoted by W_{db4} ,

$$W_{db4} = \begin{bmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & 0 & 0 & 0 & & 0 & 0 & 0 \\ \beta_0 & \beta_1 & \beta_2 & \beta_3 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & 0 & & 0 & 0 & 0 \\ 0 & 0 & \beta_0 & \beta_1 & \beta_2 & \beta_3 & 0 & & 0 & 0 & 0 \\ & & & \vdots & & & & \ddots & & \vdots & \\ \alpha_2 & \alpha_3 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & \alpha_0 & \alpha_1 \\ \beta_2 & \beta_3 & 0 & 0 & 0 & 0 & 0 & & 0 & \beta_0 & \beta_1 \end{bmatrix} \qquad (2\text{-}22)$$

And the inverse of W_{db4} is:

$$(W_{db4})^{-1} = \begin{bmatrix} \alpha_2 & \beta_2 & \alpha_0 & \beta_0 & 0 & 0 & 0 & & 0 & 0 & 0 \\ \alpha_3 & \beta_3 & \alpha_1 & \beta_1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & \beta_2 & \alpha_0 & \beta_0 & 0 & & 0 & 0 & 0 \\ 0 & 0 & \alpha_3 & \beta_3 & \alpha_1 & \beta_1 & 0 & & 0 & 0 & 0 \\ & & & \vdots & & & & \ddots & & \vdots & \\ \alpha_0 & \beta_0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & \alpha_2 & \beta_2 \\ \alpha_1 & \beta_1 & 0 & 0 & 0 & 0 & 0 & & 0 & \alpha_3 & \beta_3 \end{bmatrix} \qquad (2\text{-}23)$$

The matrix W_{db4} is an orthogonal and the parameters satisfying:

- $\alpha_0^2 + \alpha_1^2 + \alpha_2^2 + \alpha_3^2 = 1.$
- $\beta_0^2 + \beta_1^2 + \beta_2^2 + \beta_3^2 = 1.$

The db6 parameters are:

$$\alpha_0 = \frac{1+\gamma_1+\gamma_2}{16\sqrt{2}}, \alpha_1 = \frac{5+\gamma_1+3\gamma_2}{16\sqrt{2}}, \alpha_2 = \frac{10-2\gamma_1+\gamma_2}{16\sqrt{2}},$$

$$\alpha_3 = \frac{10-2\gamma_1-2\gamma_2}{16\sqrt{2}}, \alpha_4 = \frac{5+\gamma_1-3\gamma_2}{16\sqrt{2}}, \alpha_5 = \frac{1+\gamma_1-\gamma_2}{16\sqrt{2}}.$$

$$\gamma_1 = \sqrt{10},\ \gamma_2 = \sqrt{5+2\sqrt{10}}$$

Therefore, the db6 discrete wavelet transformation of degree (8×8) is:

$$W_{db6} = \begin{bmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 0 & 0 \\ \alpha_5 & -\alpha_4 & \alpha_3 & -\alpha_2 & \alpha_1 & -\alpha_0 & 0 & 0 \\ 0 & 0 & \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \\ 0 & 0 & \alpha_5 & -\alpha_4 & \alpha_3 & -\alpha_2 & \alpha_1 & -\alpha_0 \\ \alpha_4 & \alpha_5 & 0 & 0 & \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 \\ \alpha_1 & -\alpha_0 & 0 & 0 & \alpha_5 & -\alpha_4 & \alpha_3 & -\alpha_2 \\ \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 0 & 0 & \alpha_0 & \alpha_1 \\ \alpha_3 & -\alpha_3 & \alpha_1 & -\alpha_0 & 0 & 0 & \alpha_5 & -\alpha_4 \end{bmatrix} \qquad (2\text{-}24)$$

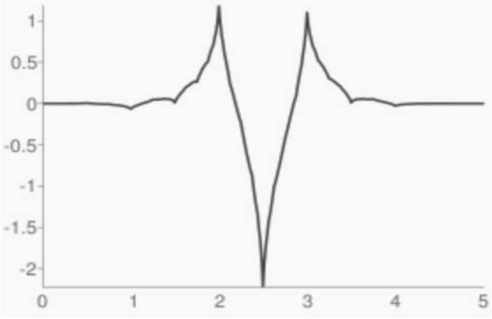
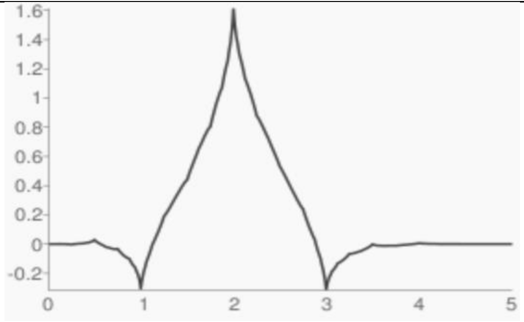
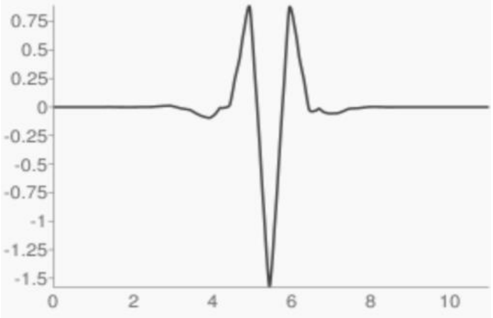
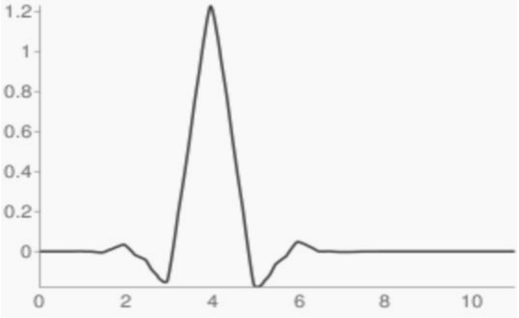
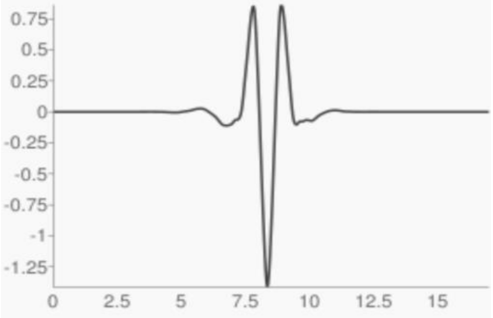
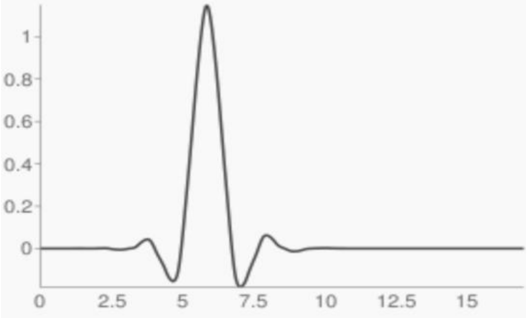
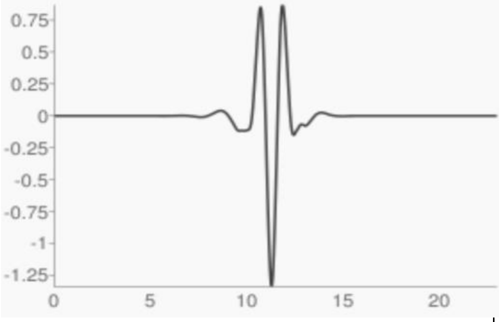
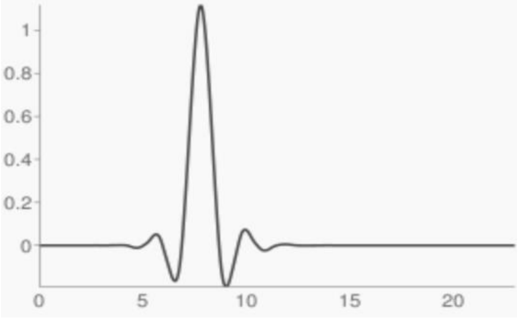
Note: the db2 wavelet transformation matrix is the same Haar wavelet.

3. Coiflet wavelet [27]

There are several advantages for the vanishing points in wavelets and we will mention two of them for their importance in mathematical and statistical studies.

- The smoother of the function $\Psi(\cdot)$ increases as the number of vanishing points increases.
- Whenever the wavelet functions have a lot of vanishing points, the wavelet series of the estimator function will converges fast to the original function.

When Ingrid Daubechies presented daubechies wavelets, only the mother wavelets $\Psi(\cdot)$ have the vanishing points. A mathematician at Yale University Ronald Raphael Coifman suggests her that the father (scaling) wavelet functions to have vanishing points. Because of that, she come out with new wavelets both father (scaling) and mother (wavelet) functions have (N/3)-1 and N/3 vanishing points respectively. The new wavelet is called Coiflet and it is semi-symmetric wavelets, and denoted by Coif1,Coif2,…,Coif9.

<table>
<tr><th></th><th>Mother coiflet wavelet $\Psi(x)$</th><th>Father coiflet wavelet $\Phi(x)$</th></tr>
<tr><td>Coif1</td><td>
</td><td>
</td></tr>
<tr><td>Coif2</td><td>
</td><td>
</td></tr>
<tr><td>Coif3</td><td>
</td><td>
</td></tr>
<tr><td>Coif4</td><td>
</td><td>
</td></tr>
</table>

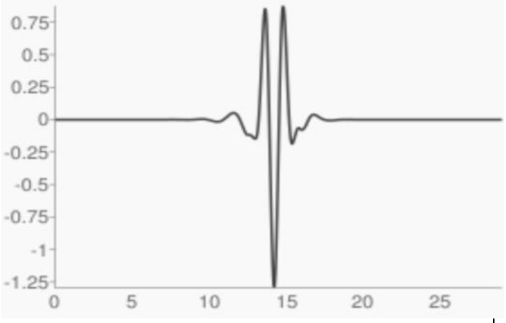
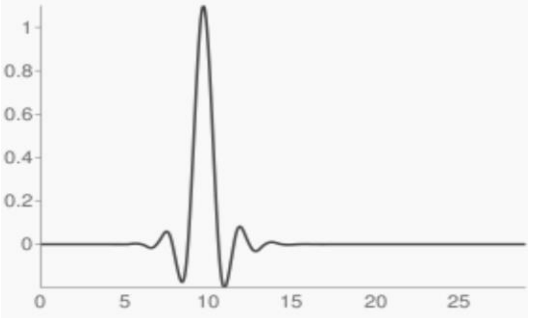
Coif5		

Figure (2-7): Mother and father Coiflet wavelet functions

For more information about Coiflet wavelets, we will give a description of Cof4 discrete wavelet transformation matrix as follows:

Cof4 has the following parameters:

$$h_{-2} = \frac{-\sqrt{2}a_0}{16}, h_{-1} = \frac{\sqrt{2}(4 - a_1)}{16}, h_0 = \frac{-\sqrt{2}(8 + a_0)}{16}$$

$$h_1 = \frac{\sqrt{2}(4 + 2a_1)}{16}, h_2 = \frac{-\sqrt{2}a_0}{16}, h_3 = \frac{-\sqrt{2}a_1}{16}$$

$$a_0 = \begin{cases} 0.5\left(\sqrt{7} - 1\right) \\ -0.5\left(\sqrt{7} + 1\right) \end{cases}, \; a_1 = \begin{cases} 0.5\left(3 - \sqrt{7}\right) \\ 0.5\left(3 + \sqrt{7}\right) \end{cases}$$

These parameters have the properties:

$$h_{-2}^2 + h_{-1}^2 + h_0^2 + h_1^2 + h_2^2 + h_3^2 = 1$$

$$h_{-2}h_0 + h_{-1}h_1 + h_0h_2 + h_1h_3 = 0$$

$$h_{-2}h_2 + h_{-1}h_3 = 0$$

Then the Cof4 discrete wavelet transformation matrix can be written as:

$$\square_{COif4} = \begin{bmatrix} h_3 & h_2 & h_1 & h_0 & h_{-1} & h_{-2} & 0 & 0 & & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_3 & h_2 & h_1 & h_0 & h_{-1} & h_{-2} & \cdots & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & \vdots & & & & & & & & \vdots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & h_3 & h_2 & h_1 & h_0 & h_{-1} & h_{-2} \\ h_{-1} & h_{-2} & 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & h_3 & h_2 & h_1 & h_0 \\ h_1 & h_0 & h_{-1} & h_{-2} & 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 & h_3 & h_2 \\ -h_{.} & h_{-1} & -h_0 & h_1 & -h_2 & h_3 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -h_{-2} & h_{-1} & -h_0 & h_1 & -h_2 & h_3 & & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & \vdots & & & & & & & & \vdots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & -h_{-2} & h_{-1} & -h_0 & h_1 & -h_2 & h_3 \\ -h_{;} & h_3 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -h_{-2} & h_{-1} & -h_0 & h_1 \\ -h_{(} & h_1 & -h_2 & h_3 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 & 0 & -h_{-2} & h_{-1} \end{bmatrix} \tag{2-25}$$

The two solutions can be compensated to find two matrices that are both orthogonal.

4. Symmlet wavelet

Symmlets are also wavelets within a minimum size support for a given number of vanishing moments, but they are as symmetrical as possible, as opposed to the Daubechies filters which are highly asymmetrical. General characteristics: Compactly supported wavelets with least asymmetry highest number of vanishing moment for a given support width. Associated scaling filters are near linear-phase filters.

2.6 Differences between wavelet and Fourier Transformations

Some differences can be identified between Fourier and Wavelet transformations according to the formulas and procedures in which the transform procedure is performed. From our point of view four important differences can be identified as follows:

1. Fourier transformations can be done appropriately for stable signals and data, while wavelet transformations can be performed appropriately for stable and unstable signals (data).

2. Fourier transformations do not have solutions for a function in the time domain and have solutions for the function in the frequency domain, comparing to wavelet transformations that have solutions in both time and frequency domains.

3. The Fourier transformations essentially transform the signals or the functions to waves of the sines and cosines in a different capacity, but wavelet transformations transform the signals or the function to a different measurement of mother and father wavelet functions.

4. In Fourier transformations, the inputs could be real or complex functions, while the outputs are complex functions. In wavelet transformations, both inputs and outputs could be real or complex functions.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This chapter introduces the methodology used in this thesis in terms of details for each step in it to estimate the function of the hazard. In addition, present some of the basic properties wavelets to utilizing the estimation the functions and the discrete wavelet transformation (DWT). Where this estimation have two types nonlinear and linear wavelet transformations, then introducing the mathematical structure of each one, it worth to know that the linear wavelet transformation is be used method in this thesis.

3.2 How wavelet approximate functions

In multiresolution analysis (as explained in chapter two), the space of all square integrable functions $L^2(\mathbb{R})$ contains two kinds of subspaces resolution and different subspaces and denoted ae V_J and $W_J, J \in \mathbb{Z}$ respectively. The sequence of functions $\{\varphi_{J,k}(x)\}\ and\ \{\psi_{J,k}(x)\}, 0 \leq k \leq 2^J - 1, J \geq 0$ are two basis for the subspaces $V_J\ and\ W_J$ respectively, where $\varphi_{J,k}(x) = 2^{\frac{J}{2}}\varphi(x-k)$ which is called the father wavelet function, and $\psi_{J,k}(x) = 2^{\frac{J}{2}}\psi(x-k)$ which is called the mother wavelet function.

It's possible that for any function $f \in \boldsymbol{L^2}(\mathbb{R})$ could be approximated using

$\{\varphi_{J,k}(x)\}\ and\ \{\psi_{J,k}(x)\}$ sequences with $j = 1,2,\dots,k \in \mathbb{Z}$, j_0 is an arbitrary starting scale, and $j_0 \leq j$.

There are two techniques to estimate the functions by wavelet nonlinear and linear wavelet estimations.

3.2.1 linear wavelet estimation [13],[30]

The linear wavelet estimation for any function $f \in L^2(\mathbb{R})$ is defined as follows:

$$f(x) = \sum_{k\in\mathbb{Z}} \omega_\varphi(j_0,k)\cdot\varphi_{j_0,k}(x) + \sum_{J=j_0}^{\infty}\sum_{k\in\mathbb{Z}} \omega_\psi(J,k)\cdot\psi_{j,k}(x) \qquad (3\text{-}1)$$

$$\omega_\varphi(j_0,k) = E[\![\varphi_{j_0,k}(x)]\!] \qquad (3\text{-}2)$$

$$\omega_\psi(J,k) = E[\![\psi_{j,k}(x)]\!] \qquad (3\text{-}3)$$

From (3-2) and (3-3) the coefficients ω_φ and ω_ψ can be written as:

- $\omega_\varphi(j_0,k) = \frac{1}{n}\sum_x f(x)\ \varphi_{j_0,k}(x)$ is called “Approximation” coefficients.
- $\omega_\psi(j,k) = \frac{1}{n}\sum_x f(x)\ \psi_{j,k}(x)$ is called “Detail” coefficients.

From (3-1) showing that j is start j_0 and end with infinity. Based on that, f(x) could be approximated from j_0 to Ĵ. The maximum value of scale index $\hat{J} = log_2(n/log_{10}(n))$ and k is belong to $\{0,1,\dots,2^{\hat{J}}-1\}$. Therefore, equation (3-1) reformulate as follows:

$$f(x) = \sum_{k} \left(\frac{1}{n} \sum_{x} f(x)\, \varphi_{j_0,k}(x) \right) \varphi_{j_0,k}(x) + \sum_{J=j_0}^{\hat{J}} \sum_{k} \left(\frac{1}{n} \sum_{x} f(x)\, \psi_{J,k}(x) \right) \psi_{J,k}(x) \tag{3-4}$$

For periodic wavelets and constant periodic value h, it can be defining the father and mother wavelets in [0,1] as:

$$\hat{\varphi}_{j_0,k}(x) = \sum_{h \in Z} \varphi(x-h) \tag{3-5}$$

$$\hat{\psi}_{J,k}(x) = \sum_{h \in Z} \psi(x-h) \tag{3-6}$$

Base on (equ.s. (3-5) and (3-6)), it is possible to rewrite (equ. (3-1)) as:

$$f(x) = \sum_{k \in \mathbb{Z}} \omega_{\hat{\varphi}}(j_0,k) \cdot \hat{\varphi}_{j_0,k}(x) + \sum_{J=j_0}^{\hat{J}} \sum_{k \in \mathbb{Z}} \omega_{\hat{\psi}}(J,k) \cdot \hat{\psi}_{J,k}(x) \tag{3-7}$$

The function $f(x)$ in (3-7) is an approximation result for using periodic wavelets with the two coefficients $\omega_{\hat{\varphi}}(j_0,k)$ and $\omega_{\hat{\psi}}(J,k), 0 \leq k \leq 2^J - 1, J \geq 0$.

And according to orthogonality projection of $f(x)$ on the subspaces V_J and W_J, then $\omega_{\hat{\varphi}}(j_0,k)$ and $\omega_{\hat{\psi}}(J,k)$ could be written as follows:

$$\omega_{\hat{\varphi}}(j_0, k) \ = \int_0^1 f(x)\,\hat{\varphi}_{j_0,k}(x)dx = \langle f(x), \hat{\varphi}_{j_0,k}(x)\rangle \tag{3-8}$$

$$\omega_{\hat{\psi}}(J, k) = \int_0^1 f(x)\,\hat{\psi}_{J,k}(x)dx = \langle f(x), \hat{\psi}_{J,k}(x)\rangle \tag{3-9}$$

Because of $\left\{\hat{\varphi}_{J,k}(x)\right\}, 0 \le k \le 2^J - 1, J \ge 0,$ is an orthogonal basis of the subspace V_{J+1}, it's possible to write any function $f(x) \in V_{J+1}$ in [0,1] as follows:

$$f(x) = \sum_{J\ge0} \sum_{k=0}^{2^J-1} \langle f(x), \hat{\varphi}_{J,k}(x)\rangle \cdot \ \hat{\varphi}_{J,k}(x) \tag{3-10}$$

One of the good advantages for wavelet is to estimate any function f ∊ $L^2(\mathbb{R})$ which approximate this function based on orthogonality projection. Basically, for fixed scale Ĵ the orthogonal projection of $f_{\hat{\jmath}}(x)$ onto the subspace $V_{\hat{\jmath}}$ is denoted $\mathcal{P}\left(f_{\hat{\jmath}}(x)\right)$.

Generally, fixed $J = \hat{J}$ and rewrite equation (3-7) as a projection of $f(x)$ in V_J and represented as:

$$\mathcal{P}\left(f_{\hat{\jmath}}(x)\right) = \sum_{k=0}^{2^{\hat{\jmath}}-1} \langle f(x), \hat{\varphi}_{J,k}(x)\rangle\, \hat{\varphi}_{J,k}(x) \tag{3-11}$$

Moreover, from periodic wavelet it could be shown that

$$\left\|\mathcal{P}\left(f_{\hat{\jmath}}(x)\right) - f(x)\right\|_2 \to 0 \ as\, J \to \infty$$

$$\left\|\mathcal{P}\left(f_{\hat{\jmath}}(x)\right)-f(x)\right\|_{\infty} \to 0\ as\ J \to \infty.$$

3.2.2 Nonlinear wavelet estimation [12],[43],[54]

The nonlinear wavelet estimation for any function $f \in L^2(\mathbb{R})$ is defined as follows:

$$f(x)=\sum_{k\in\mathbb{Z}}\omega_{\varphi}(j_0,k)\cdot\varphi_{j_0,k}(x)+\sum_{J=j_0}^{\infty}\sum_{k\in\mathbb{Z}}\omega_{\psi}^{\partial}(J,k)\cdot\psi_{j,k}(x)$$

Where $\partial > 0$ is known as wavelet threshold parameter and considering as one of the fundamentals of WT if it is discrete (DWT) or continuous (CWT), $\omega_{\psi}^{\partial}(J,k)$ is defined as thresholding mother wavelet coefficient and has two forms according to the wavelet threshold parameter (∂) either hard or soft thresholding rule known as follows:

$$\omega_{\psi}^{\partial}(J,k)=\begin{cases}\omega_{\psi}^{H}(J,k)\ if\ Hard\ t\square res\square old\\ \omega_{\psi}^{S}(J,k)\ if\ soft\ t\square res\square old\end{cases}$$

Where $\omega_{\psi}^{H}(J,k)$ and $\omega_{\psi}^{S}(J,k)$ are given as follows:

$$\omega_{\psi}^{H}(J,k)=\begin{cases}\omega_{\psi}(J,k) & if\ \ \left|\omega_{\psi}(J,k)\right|>\partial\\ 0 & if\ \ \left|\omega_{\psi}(J,k)\right|\le\partial\end{cases}$$

$$\omega_{\psi}^{S}(J,k)=\begin{cases}\omega_{\psi}(J,k)+\partial & if\ \ \omega_{\psi}(J,k)<-\partial\\ \omega_{\psi}(J,k)-\partial & if\ \ \omega_{\psi}(J,k)>\partial\\ 0 & if\ \left|\omega_{\psi}(J,k)\right|\le\partial\end{cases}$$

The wavelet threshold parameter (∂) will be used to remove any interference in the signal data after transmitting. The wavelet threshold will take the role of a preapprehension parameter in the functions where the increase or decrease will affect the amount of data preapprehension. There are two types wavelet threshold rules according to the method of data preapprehension, defined as follows:

- **Hard wavelet threshold**

The hard threshold will be used to make data whose values are less than the threshold value equal to zero.

$$\left(\hat{X}_i\right)_H = \begin{cases} X_i \; if \; |X_i| > \partial \\ 0 \; if \; |X_i| \leq \partial \end{cases}$$

- **Soft wavelet threshold**

$$\left(\hat{X}_i\right)_S = \begin{cases} X_i - \partial \; if \; |X_i| > \partial \\ \quad 0 \qquad if \;\; X_i \leq \partial \\ X_i + \partial \; if \; X_i < -\partial \end{cases}$$

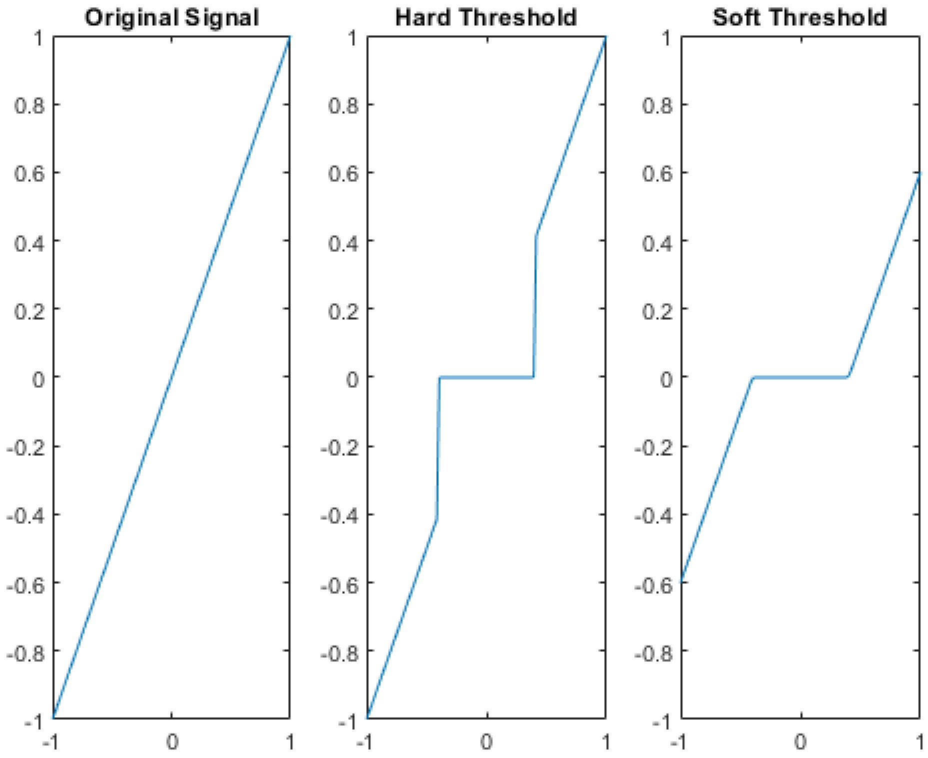


Figure (3-1): hard and soft thresholds of data curves.

There are many methods to determine the value of wavelet threshold parameter (∂) depend on the way of collection:

- Universal thresholding.
- Sub band adaptive thresholding.
- Spatially adaptive thresholding.
- Cross Validation.
- Sure thresholding.
- Bayesian methods.
- False-Discovery rate.

- Exact minimax.

For clarification, we will present the formula given by (Donoho *et al.*, 1996) [24] as follows:

$$\partial = \sigma\sqrt{2\ln(n)}$$

$$\sigma = \frac{median(|X_i - median(X_i)|)}{0.6745}$$

Where, E(M)= 0.6745 and M is the absolute vale of independent Gaussian random variable with zero mean and variance σ_0^2.

In order to get more details you can check the reference of (Luo and Zhang, 2012) [43].

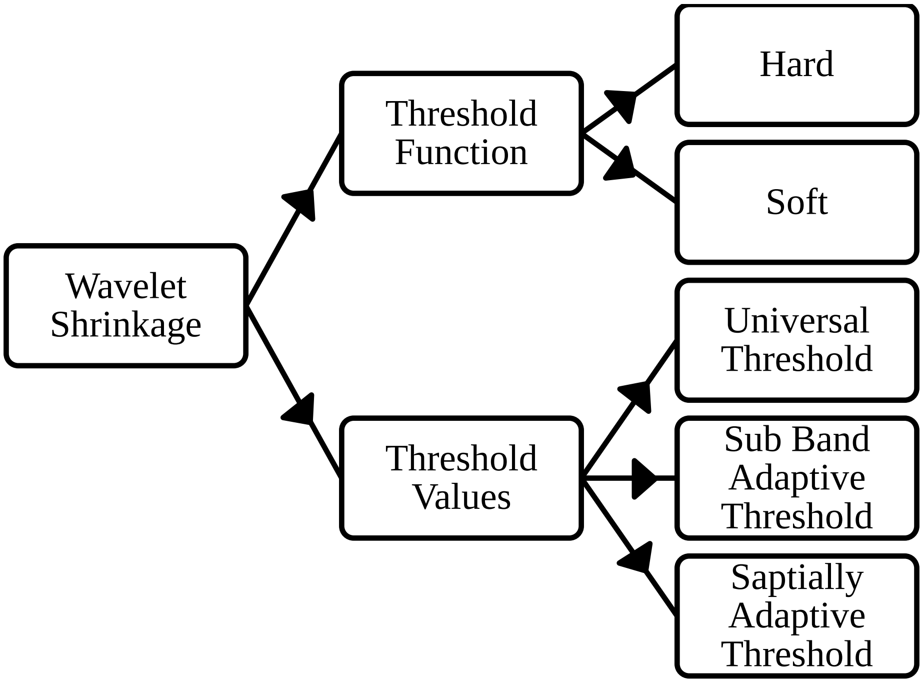


Figure (3-2): diagram of wavelet shrinkage and own threshold values.

3.3 Model-up and Hazard rate function [7],[49]

In this thesis we will study the samples which are randomly right censoring data and follows the assumptions:

- Lifetimes: let $X_1, X_2, \dots, X_n$ be a non-negative i.i.d distributed with continuous cumulative (F) and density (f) functions.
- Censoring times: let $C_1, C_2, \dots, C_n$ be non-negative i.i.d distributed with continuous cumulative (G) and density (g) functions.
- Independence includes both Lifetimes and Censoring times.

Let $Z_i, = \min\{X_i, C_i\}, i = 1,2, \dots, n$ be the survival times (observed times) with the indicator function $\delta_i = 1_{X_i \leq C_i}$ and 0 otherwise, so there is censoring for i$^{\text{th}}$ observed time if $\delta_i = 1$.

Hazard function known as failure rate function and denoted by $h(t) \; or \; \lambda(t)$,

$$\lambda(t) = \frac{f(t)}{1 - F(t)}$$

Hazard rate function has a special form in the censored case with $F(t) < 1 \; and \; G(t) < 1$ as:

$$\lambda(t) = \lambda_F(t) + \lambda_G(t)$$

which usually defined as:

$$\lambda(t) = \frac{f(t)}{\left(1 - F(t)\right)} + \frac{g(t)}{\left(1 - G(t)\right)}$$

And as a direct result of above, we get:

$$\lambda(t) = \frac{f(t)\left(1 - G(t)\right) + g(t)\left(1 - F(t)\right)}{\left(1 - F(t)\right)\left(1 - G(t)\right)}$$

Now, let that,

$$L(t) = P(Z_i \leq t) = 1 - P(Z_i > t)$$

$$= 1 - P(X_i > t, C_i > t)$$

$$= 1 - \left(1 - F(t)\right)\left(1 - G(t)\right)$$

Rewrite the hazard rate function with assuming that

$$f^*(t) = f(t)\left(1 - G(t)\right) + g(t)\left(1 - F(t)\right)$$

And

$$S^*(t) = 1 - L(t)$$

$f^*(t)$ and $S^*(t)$ be the density and survival functions, then:

$$\lambda(t) = \frac{f^*(t)}{S^*(t)} \qquad (3\text{-}12)$$

Before start estimating the hazard rate function, there are some details that are important to know.

Assuming that $\beta = max\{Z_i, i = 1,2, \dots, n\}$ and to make sure that all observed times Z_i belong to [0,1], putting all observing in normalized form, such that $\hat{Z}_i = \frac{1}{\beta} Z_i$ and $\left\{\hat{Z}_{(i)}, \delta_{(i)}\right\}$ be the ranked of $\left\{\hat{Z}_i, \delta_i\right\}$.

$$T_F = sup\{z; F(z) < 1\}$$

$$T_G = sup\{z; G(z) < 1\}$$

$$T_L = sup\{t; L(z) < 1\} = min\{T_F\,, T_G\}$$

The estimator is on the $[0, \beta]$, it's clear that $\beta < T_L$ and $Z_{(n)} \rightarrow T_L \; as \; n \rightarrow \infty$. Depending on what was mentioned above, suppose that $\beta = Z_{(n)}$.

3.4 Estimation of the hazard function

Our strategy to estimate Hazard function follows partially estimation, first estimate the probability density function denoted as$\left(\hat{f}^p(x)\right)$ and then estimate survival function denoted as$\left(\hat{S}^p(x) = 1 - \hat{F}(x)\right)$.

3.4.1 Estimation of density function $\left(\hat{f}^p(x)\right)$

In order to estimate $\left(\hat{f}^p(x)\right)$, The wavelet projection method previously referred to as (3-11). It will be followed by the creation of a hybrid between the wavelet and the Breslow estimate.

$$\hat{f}^p(x) = \sum_{k=0}^{2^{\hat{J}}-1} \langle f(x), \hat{\varphi}_{J,k}(x) \rangle \, \hat{\varphi}_{J,k}(x) \qquad (3\text{-}13)$$

Based on equ. (3-13), need to find the coefficient$\langle f(x), \hat{\varphi}_{j_0,k}(x)\rangle$, so let first denoted it as $€_{\varphi}$. Moreover, since $f(x)$ is unknown density function, for that use the cumulative functions (cdf) F and G to collect $€_{\varphi}$.

From the observed data $\{Z_i, \delta_i\}$ $i = 1,2, \dots, n$, the joint distribution of (Z, δ) is:

$$P(Z \leq z, \delta = 1) = \int_{-\infty}^{z} \big(1 - G(x)\big) f(x) dx \dots (14) \qquad (3\text{-}14)$$

$$P(Z \leq z, \delta = 0) = \int_{-\infty}^{z} G(x) f(x) dx + \int_{z}^{\infty} G(z) f(x) dx$$

$$P(Z \leq z, \delta = 0) = \int_{-\infty}^{z} G(x) f(x) dx + G(z)\big(1 - F(z)\big) \qquad (3\text{-}15)$$

Dependent on equations (3-14) and (3-15):

$$f_Z(z) = f_X(z)\big(1 - G_C(z)\big) + g_C(z)\big(1 - F_X(z)\big) \qquad (3\text{-}16)$$

As a result for equation (3-16):

$$f_X(z) = \frac{f_Z(z)}{1 - G_C(z)} - \frac{g_C(z)\big(1 - F_X(z)\big)}{1 - G_C(z)} \qquad (3\text{-}17)$$

From (3-17) it possible to express and formed $€_{\varphi}$=$\langle f(x), \varphi_{j_0,k}(x)\rangle$, as

$$€_{\varphi} = \int_{0}^{1} \left[\frac{f_Z(z)}{1 - G_C(z)} \; - \; \frac{g_C(z)\big(1 - F_X(z)\big)}{1 - G_C(z)}\right] \cdot \hat{\varphi}_{j_0,k}(x) \; d(x)$$

$$\text{€}_{\varphi}=E\left\llbracket\frac{\hat{\varphi}_{j_0,k}(Z)}{1-G(Z)}\right\rrbracket-E\left\llbracket\frac{\big(1-F(Z)\big)\cdot\hat{\varphi}_{j_0,k}(Z)}{1-G(Z)}\right\rrbracket \qquad (3\text{-}18)$$

Using the approach $\left(\omega_{\varphi}(j_0,k)=\frac{1}{n}\sum_{x}f(x)\,\varphi_{j_0,k}(x)\right)$ for $0\leq G(Z_i)<1$, $i=1,2,\dots,n$:

$$\begin{aligned}\text{€}_{\varphi}=n^{-1}&\sum\nolimits_{i=1}^{n}\frac{\hat{\varphi}_{j_0,k}(Z_i)}{\big(1-G(Z_i)\big)}\\&-n^{-1}\sum\nolimits_{i=1}^{n}\frac{I_{\delta_i=0}(1-F(Z_i))\cdot\hat{\varphi}_{j_0,k}(Z_i)}{\big(1-G(Z_i)\big)}\end{aligned} \qquad (3\text{-}19)$$

Now, $F\big(\hat{Z}_{(i)}\big)\ and\ G(\hat{Z}_{(i)})$ for $i=1,2,\dots,n$ can be estimated using Breslow estimator for survival function as follows:

$$\hat{F}\big(\hat{Z}_{(i)}\big)=\sum_{r=1}^{i}\frac{\delta_{(r)}}{n-r+1}\ e^{-\left(\sum_{s=1}^{r-1}\frac{\delta_{(s)}}{n-s+1}\right)} \qquad (3\text{-}20)$$

$$\hat{G}\big(\hat{Z}_{(i)}\big)=\sum_{r=1}^{i}\frac{1-\delta_{(r)}}{n-r+1}\ e^{-\left(\sum_{s=1}^{r-1}\frac{1-\delta_{(s)}}{n-s+1}\right)} \qquad (3\text{-}21)$$

$$\eta_i=\frac{1-I_{\delta_i=0}\left(1-\tilde{F}\big(\hat{Z}_{(i)}\big)\right)}{\big(1-\tilde{G}(\hat{Z}_{(i)})\big)} \qquad (3\text{-}22)$$

Rewrite equation (3-19) as follows:

$$\text{€}_{\varphi}=n^{-1}\sum_{i=1}^{n}\eta_i\cdot\hat{\varphi}_{\hat{j},k}\big(\hat{Z}_i\big) \qquad (3\text{-}23)$$

Finally, the estimate of density function $\hat{f}^{p}(x)$ for chosen scale index $\left(\hat{\jmath}\right)$ can be formed as:

$$\hat{f}^{p}(x)=\sum_{k=0}^{2^{\hat{\jmath}}-1}€_{\Phi}\,\hat{\varphi}_{\hat{\jmath},k}\left(\hat{Z}_{(i)}\right) \tag{3-24}$$

3.4.2 Estimation of survival function $\left(\hat{\boldsymbol{S}}^{\boldsymbol{p}}(\boldsymbol{x})\right)$[10]

The strategy of finding and estimating the survival function $\left(\hat{\boldsymbol{S}}^{\boldsymbol{p}}(\boldsymbol{x})\right)$ which is depend on using the empirical cumulative function (Ecdf). The empirical distribution function has a lot of statistical applications especially in nonparametric studies such as survival and reliability analyses, resampling methods, nonparametric regression models, sequential testing, and many other applications.

Mathematically, suppose that $X_1, X_2, \dots, X_n$ for some integer index n, be random variables with unknown distribution function F. the empirical distribution function denoted as $\left(\hat{F}(x)\right)$ and define as follows:

$$\hat{F}(x)=(n^{-1})\sum_{i=1}^{n}I(X_i\leq x) \tag{3-25}$$

Where $\left(I(X_i\leq x)\right)$ is known as the indicator function and defined as:

$$I(X_i\leq x)=\begin{cases}1 & if\ X_i\leq x\\ 0 & otherwise\end{cases} \tag{3-26}$$

To be ensure that the empirical function and hence the survival function will be belong to the interval (0,1), $\hat{F}(x)$ take the form:

$$\hat{F}(x) = ((n+1)^{-1}) \cdot \sum_{i=1}^{n} I(X_i \leq x) \qquad (3\text{-}27)$$

Now, the survival function defines as:

$$\hat{S}^p(x) = 1 - \hat{F}(x) \qquad (3\text{-}28)$$

Then:

$$\hat{S}^p(x) = 1 - \left(((n+1)^{-1}) \cdot \sum_{i=1}^{n} I(X_i \leq x) \right) \qquad (3\text{-}29)$$

Then finally, the hazard rate function defines as:

$$\hat{\lambda}(x) = \frac{\hat{f}^p(x)}{\hat{S}^p(x)} \qquad (3\text{-}30)$$

3.5 Algorithm implementation of estimation hazard rate function

This section introduces the algorithm implement of estimation the hazard rate function using wavelet. The algorithm presents a sequence steps with the order of finding the following:

- Determinate the sample size (n).
- The projection multiresolution space of index $(\hat{\jmath})$.

- Father wavelet matrix.
- Breslow estimator of $\hat{F}\left(\hat{Z}_{(i)}\right)$ and $\hat{G}\left(\hat{Z}_{(i)}\right)$.
- The coefficient €$_{\Phi}$.
- The probability density function $\hat{f}^{p}(x)$.
- The empirical survival function $\hat{S}^{p}(x)$.
- The hazard rate function $\hat{\lambda}(x)$.

➢ **Step 1**

Determinate the sample size (n) and normalized the data by divide all data samples by $\beta = max\{Z_i, i = 1,2,\dots,n\}$.

➢ **Step 2**

Collect the maximum value of the projection multiresolution space index

$$\hat{J} = log_2(n) - \left(log_2\left(log(n)\right)\right)$$

➢ **Step 3**

Constriction the father wavelet matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n\times 2^{\hat{J}}}$ for $k = 0,1,\dots,2^{\hat{J}} - 1$, and $i = 1,2,\dots,n$, as follows:

$$\boldsymbol{\Phi} = \begin{bmatrix} \hat{\varphi}_{\hat{J},0}\left(\hat{Z}_{(1)}\right) & \hat{\varphi}_{\hat{J},2}\left(\hat{Z}_{(1)}\right) & \cdots & \hat{\varphi}_{\hat{J},2^{\hat{J}}-1}\left(\hat{Z}_{(1)}\right) \\ \hat{\varphi}_{\hat{J},0}\left(\hat{Z}_{(2)}\right) & \hat{\varphi}_{\hat{J},1}\left(\hat{Z}_{(2)}\right) & & \hat{\varphi}_{\hat{J},2^{\hat{J}}-1}\left(\hat{Z}_{(2)}\right) \\ \vdots & & \ddots & \vdots \\ \hat{\varphi}_{\hat{J},n}\left(\hat{Z}_{(n)}\right) & \hat{\varphi}_{\hat{J},n}\left(\hat{Z}_{(n)}\right) & \cdots & \hat{\varphi}_{\hat{J},2^{\hat{J}}-1}\left(\hat{Z}_{(n)}\right) \end{bmatrix}$$

➢ Step 4

For $i = 1,2,\dots,n$ collect $\hat{F}\left(\hat{Z}_{(i)}\right)$ and $\hat{G}\left(\hat{Z}_{(i)}\right)$ using Breslow estimator, then find the following:

- the vector matrices$\left[1_{n\times1} - \hat{G}\left(\hat{Z}_{(i)}\right)\right]_{n\times1}$ and $\left[1_{n\times1} - \hat{F}\left(\hat{Z}_{(i)}\right)\right]_{n\times1}$
- $\left(\left[1_{n\times1} - \hat{G}\left(\hat{Z}_{(i)}\right)\right]_{n\times1}\right)^{inv} = \begin{bmatrix} \frac{1}{1-\hat{G}\left(\hat{Z}_{(1)}\right)} \\ \frac{1}{1-\hat{G}\left(\hat{Z}_{(2)}\right)} \\ \vdots \\ \frac{1}{1-\hat{G}\left(\hat{Z}_{(n)}\right)} \end{bmatrix}$
- $A_1 = \left(\frac{1}{1-\hat{G}\left(\hat{Z}_{(1)}\right)} \quad \frac{1}{1-\hat{G}\left(\hat{Z}_{(2)}\right)} \quad \cdots \quad \frac{1}{1-\hat{G}\left(\hat{Z}_{(n)}\right)}\right)$
- $A_2 = \left[1_{n\times1} - \hat{F}\left(\hat{Z}_{(i)}\right)\right]_{n\times1}$

➢ Step 5

This is collection of the coefficient $€_{\Phi}$, but first must find the values of

$\eta_i, i = 1,2,\dots,n$ using the values $A_1$and A_2 from step (4):

$$(A_3)_{1\times n} = A_1 \times \left[diag(A_2) \times diag\left([1_{n\times1}] - \left[I_{\delta_i=0}\right]\right)\right]$$

$$\eta = (n^{-1}) \cdot [A_1 - A_3]$$

Now for $k = 0,1, \dots, 2^{\hat{J}} - 1$ find the coefficient $€_{\Phi}$:

$$€_{\Phi} = \eta \times [\boldsymbol{\Phi}_k]$$

- **Step 6**

Compute the probability density function $\hat{f}^p(x)$ for $k = 0,1, \dots, 2^{\hat{J}} - 1$

$$\hat{f}^p(x) = (€_{\Phi})_{1\times k} \cdot [\boldsymbol{\Phi}_k]$$

- **Step 7**

Compute the survival function $\hat{S}^p(x)$:

- Construction lower triangular matrix $\text{⍰}_{n\times n}$
- For $i = 1,2, \dots, n$ compute $\text{⍰}_i = sum(\text{⍰}_{i\times n})$
- For i= $1,2, \dots, n$ the survival function computing as follows:

$$\hat{S}^p(x) = [1_{n\times 1}] - ((n+1)^{-1}[\text{⍰}_i])$$

- **Step 8**

Finally, for $i = 1,2, \dots, n$ compute the hazard rate function:

$$\hat{\lambda}(x) = \frac{\hat{f}^p(x)}{\hat{S}^p(x)}$$

CHAPTER FOUR

APPLICATIONS

4.1 Introductions

The content that included in this chapter is the application which determine the quality and reliability of the proposed method. The application is divided into two parts, including the first part including the simulation techniques and the second part the real data application.

4.2 Simulation study

Simulation is one of the most important techniques in statistics, it is provides the possibility of generating samples that can be studied and analyzed in case they are difficult to obtain from reality. The advantage of simulations, especially in the case of right randomly censoring data, is the knowledge of the actual values of X and C. In addition, it is possible to identify the approaches of the statistical functions (probability density function, the hazard rate function) which provide great potential for comparison between them and the estimators.

It should be noted that the estimation process in simulation studies follows the experimental method, in the sense of clarity and more detailed, that there are some constraints to be taken into account. These constrains can be listed as follows:

1. The samples size.
2. The wavelet type.
3. The number of vanishing points.
4. The projection multiresolution space index.

In the simulation studies and because of the number of repetitions, the process of compatibility between the constraints takes a course of difficulty, so in order to reach a consensus between constraints as soon as possible, we have taken partial samples from the experiment so that the experiment is divided into four parts and then from each part we chose a set of samples for experimentation. According to all those reasons, the estimation of hazard rate function in simulations studies will be done with one type of wavelet.

4.2.1 First simulation experiment

First simulation experiment is generated data that distributed as Gamma distribution for lifetimes $\{X_i\}_{i=1}^{n}$ with two parameters, shape parameter equal to 5 and scale parameter equal to 1. The independent censoring times $\{C_i\}_{i=1}^{n}$ are generated using exponential distribution with one parameter equal to 6. The aim of choosing parameters with X_i~G(5,1) and C_i~EXP(6) is to have simulation data with 50% censoring. For data generation, n = 100, 200 were selected. The following algorithm steps showing how generating and preparing the database for n=100,200 using Gamma distribution (G(5,1)) and Exponential distribution (Exp(6)).

- **Step 1**

The first step is important since it determines the data size (n=100,200) and all next steps depend on it.

- **Step 2**

Generate random samples for the lifetime's data $\{X_i\}_{i=1}^{n}$ use G(5,1) in Matlab 2018a programing with order (`gamrnd(A,B))`.

- **Step 3**

Generate random samples The independent censoring times $\{C_i\}_{i=1}^{n}$ use Exp(6) in Matlab 2018a programing with order (`exprnd(mu)`).

- **Step 4**

Finding indicator function for i=1,2,..n, where:

$$\delta_i = \begin{cases} 1 \; if \; X_i \leq C_i \\ 0 \; if \; X_i > C_i \end{cases}$$

- **Step 5**

Finding the failure times vectors Z, such that $Z_i = min\{X_i, C_i\}$ for i=1,2,…,n.

- **Step 6**

Organizes the data-matrix as $(n \times 2)$ matrix with Z_i's as first column and δ_i's as the second column for i=1,2,..n.

- **Step 7**

Applying the algorithm of finding the actual density function and hazard rate function which are defined in chapter three (section 3.2).

- **Step 8**

Finally, applying the data-matrix from the step 6 in algorithm that introduced in chapter three (section 3.4) to estimate the density function and hazard rate function.

4.2.1.1 Discussion and results of First simulation experiment

Before reviewing the results it is important to review some important details that play a role in the estimation process.

The process of determining the wavelet resolution index $(\hat{J})$ and the wavelet type is considered the basis of the estimation process of the hazard rate function by wavelets, and many have been trying to derive a general method for finding the appropriate wavelet resolution index $(\hat{J})$ for any set of data, but the researcher's experience is the most important and decisive in this aspect.

New, to determine the maximum wavelet resolution index $(\hat{J})$, set $\hat{J} = log_2(n/log_{10}(n))$, with using the experimentation process can determine the type of wavelet, wavelet resolution index $(\hat{J})$, and number of vanishing points as follows: the Daubecheis (db50) is the wavelet type with $\hat{J} = 3$. For comparing between the actual and estimation of hazard rate function, use the global error measurement with the form as follows:

$$MSE = R^{-1} \sum_{r=1}^{R} n^{-1} \sum_{i=1}^{n} \left(\lambda(t_i) - \hat{\lambda}_{n,r}(t_i) \right)^2$$

Where R =200 is the number to repeat the experience.

Table (4-1) showing the results of MSE for the first simulation experiment of size sample n=100,200, in figures (4-1) and (4-2) the red curves represent the wavelet estimation of hazard rate functions and the blue curves represent the actual hazard rate functions of data generating by Gamma distribution and the independent censoring times generating by Exponential distribution for n=100,200.

n	MSE_Hazard
100	0.273224908
200	0.201609502

Table (4-1): MSE of hazard rate function for first simulation experiment

The table (4-1) showing that the MSE for the hazard rate estimation using wavelet (dN50 and $\hat{J} = 3$) and MSE is minimized especially for n=200 and R=200.

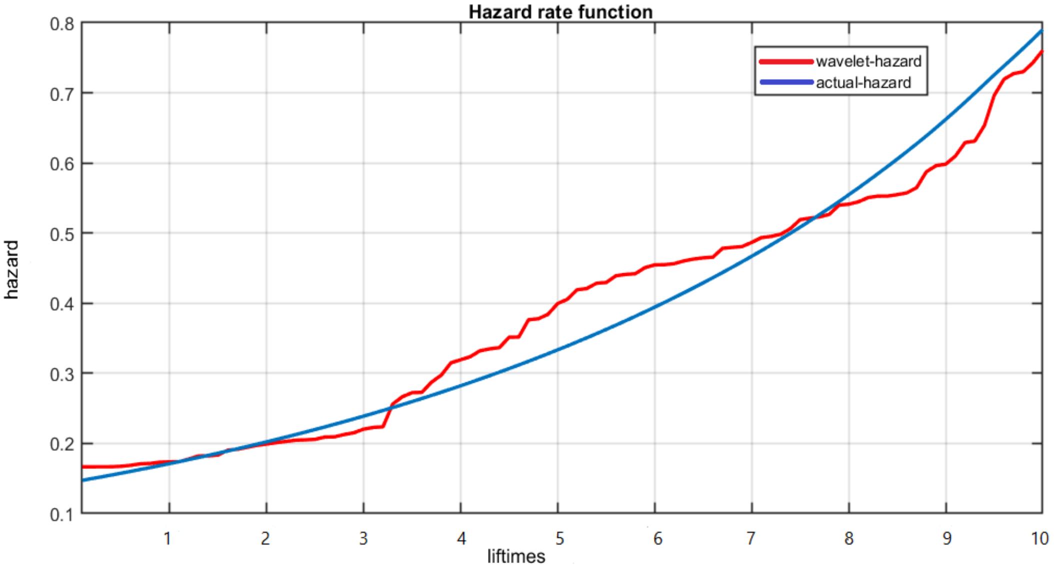


Figure (4-1): Estimation of hazard rate function of simulation experiment given in (4.2.1) with n=100.

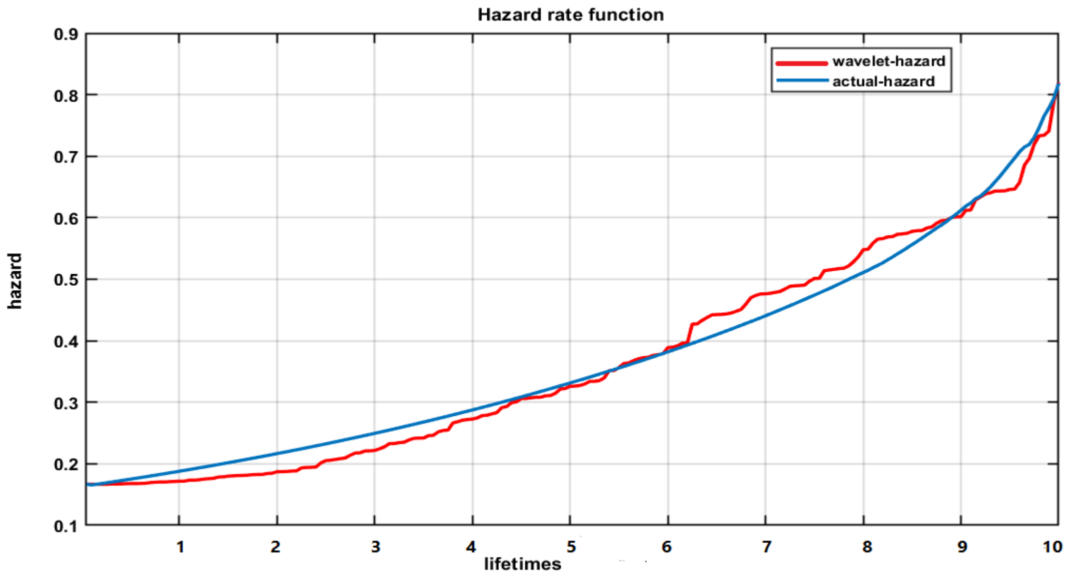


Figure (4-2): Estimation of hazard rate function of simulation experiment given in (4.2.1) with n=200.

From figure (4-1) noting that the estimation of hazard rate functions are convergent to actual hazard rat functions when n=100, but in figure (4-2) noting that the estimation of hazard rate functions are more convergent to actual hazard rat functions when n=200.

Table (4-2) showing the results of MSE for the first simulation experiment of size sample n=100,200, in figures (4-3) and (4-4) the red curves represent the wavelet estimation of density functions and the blue curves represent the actual density functions of data generating by Gamma distribution and the independent censoring times generating by Exponential distribution for n=100,200.

n	MSE_Density
100	0.323315
200	0.214655

Table (4-2): MSE of density function for first simulation experiment

The table (4-2) showing that the MSE of probability density function estimation using wavelet (dN50 and $\hat{J} = 3$) and MSE is minimized especially for n=200 and R=200.

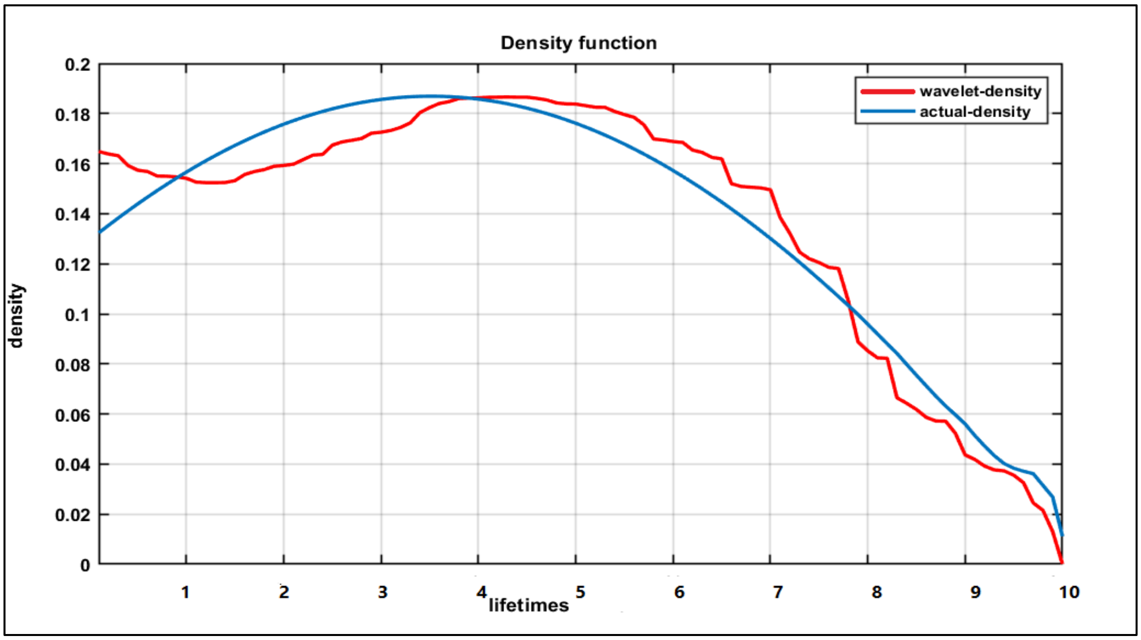


Figure (4-3): Estimation of density function of simulation experiment given in (4.2.1) with n=100.

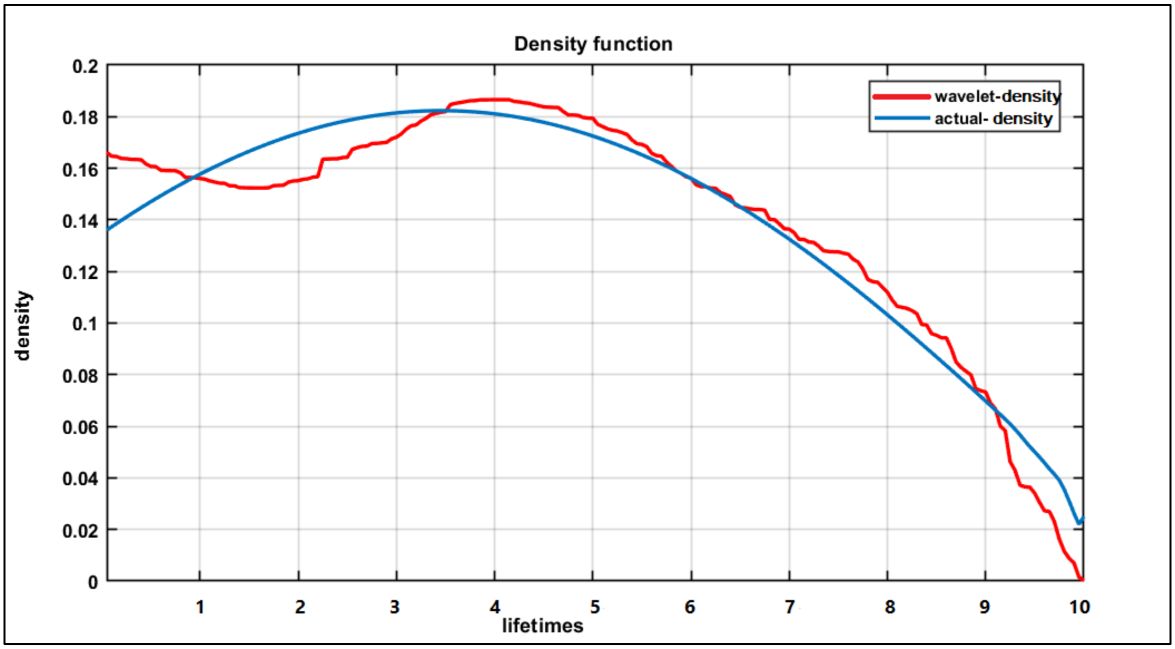


Figure (4-4): Estimation of density function of simulation experiment given in (4.2.1) of with n=200.

From figure (4-3) noting that the estimation of probability density functions are convergent to actual probability density functions when n=100, but in figure (4-4) noting that the estimation of probability density functions are more convergent to actual probability density functions when n=200.

4.2.2 Second simulation experiment

Second simulation data is generated using bimodal distribution for lifetimes$\{X_i\}_{i=1}^{n}$, such that $f(X_i) = 0.4X_1 + 0.6X_2$ where $X_1 \sim N(5,1)$ and $X_2 \sim N(3,0.5)$. The independent censoring times $\{C_i\}_{i=1}^{n}$ are generated using exponential distribution with one parameter equal to 6. The aim of choosing parameters for both distributions is to have simulation data with an average 45% censoring. For data generation, n = 400, 600 were selected. The following algorithm steps showing how generating and preparing the database for n=400,600 using bimodal distribution and Exponential distribution.

- **Step 1**

The first step is important since it is determined the data size (n=100,200) and all next steps dependent on it.

- **Step 2**

Generate random samples for the lifetime's data $\{X_i\}_{i=1}^{n}$ with following Matlab 2018a programing:

- First, generate a bimodal variable, which 40% of the time (on average) will be equal to 1, and 60% of the time (on average) will be equal to zero, with order:

```
(frac = rand(n,1) < 0.4;)
```

- Generate the two normal distributions to sample from with orders

```
    norm1=5+randn(n,500);
  norm2=3+0.5*randn(n,500);}
```

- Generate lifetimes $\{X_i\}_{i=1}^{n}$ with order :

```
   X= frac.*norm1 + (1-frac).*norm2;
```

- **Step 3**

Generate random samples the independent censoring times $\{C_i\}_{i=1}^{n}$ use Exp(6) in Matlab 2018a programing with order (`exprnd(mu))`.

- **Step 4**

Finding indicator function for i=1,2,..n, where:

$$\delta_i = \begin{cases} 1 \; if \; X_i \leq C_i \\ 0 \; if \; X_i > C_i \end{cases}$$

- **Step 5**

Finding the failure times vectors Z, such that $Z_i = min\{X_i, C_i\}$ for i=1,2,…,n.

- **Step 6**

Organizes the data-matrix as $(n \times 2)$ matrix with Z_i's as first column and δ_i's as the second column for i=1,2,..n.

- **Step 7**

Applying the algorithm of finding the actual density function and hazard rate function which are defined in chapter three (section 3.2).

- **Step 8**

Finally, applying the data-matrix from the step 6 in algorithm that introduced in chapter three (section 3.4) to estimate the density function and hazard rate function.

4.2.2.1 Discussion and results of second simulation experiment

As mentioned earlier in the subsection (4.2.1.1), the process of determining the level and type of wavelet is considered the most important process in the estimation. In this subsection, the data generation process is considered more complex than the first one, because there are two normal distributions in the process of data generation (lifetimes) which also includes the weight (factor) controls the process of generating, so the previous process in finding the level is not useful here, but it determines the upper limits and through experimentation we can determine the level required. So the resolution $(\hat{J})$ is choosing here and equal to (3). Furthermore, the wavelet type is choosing to be the Daubechies (dN20).

Table (4-3) showing the results of MSE for the bimodal simulation experiment of size sample n=400,600 with repetition R=500, in figures (4-5) and (4-6) the red curves represent the wavelet estimation of hazard rate functions and the blue curves represent the actual hazard rate functions of data generating by bimodal distribution and the independent censoring times generating by Exponential distribution for n=400,600.

n	MSE_Hazard
400	0.632627407
600	0.541041413

Table (4-3): MSE of hazard rate function for second simulation experiment

The table (4-3) showing that the MSE for the hazard rate estimation using wavelet (dN20 and $\hat{J} = 3$) and MSE is minimized especially for n=600, and R=500.

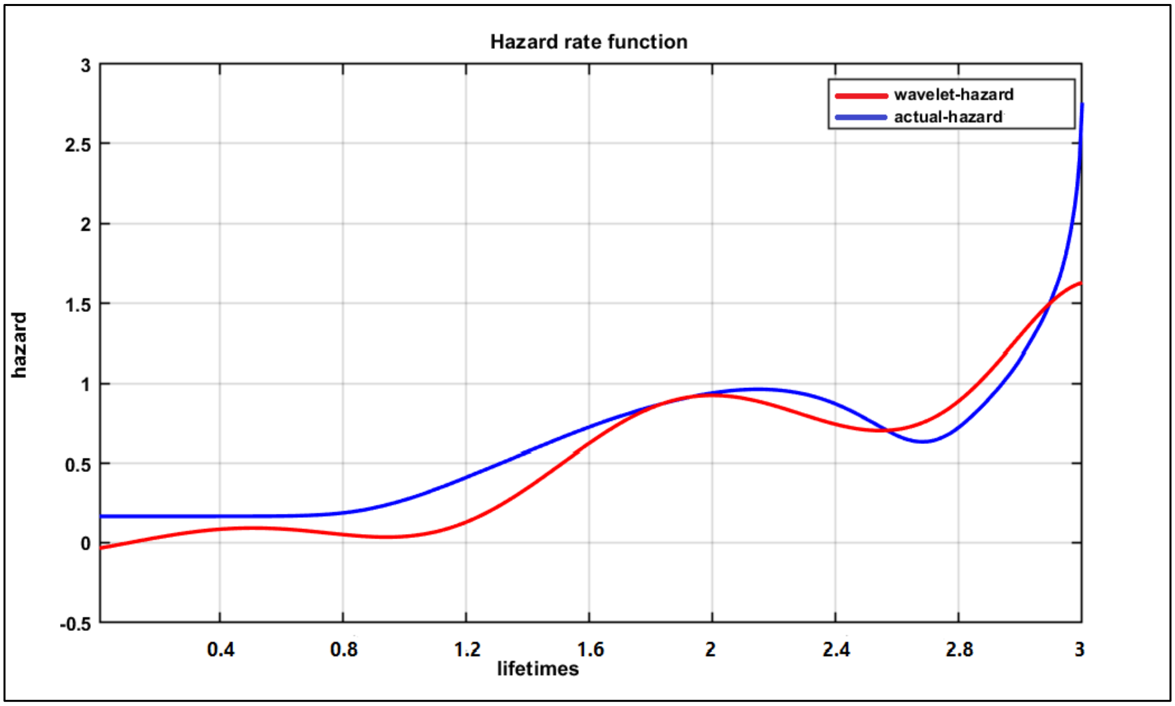


Figure (4-5): Estimation of hazard rate function of simulation experiment given in (4.2.2) with n=400.

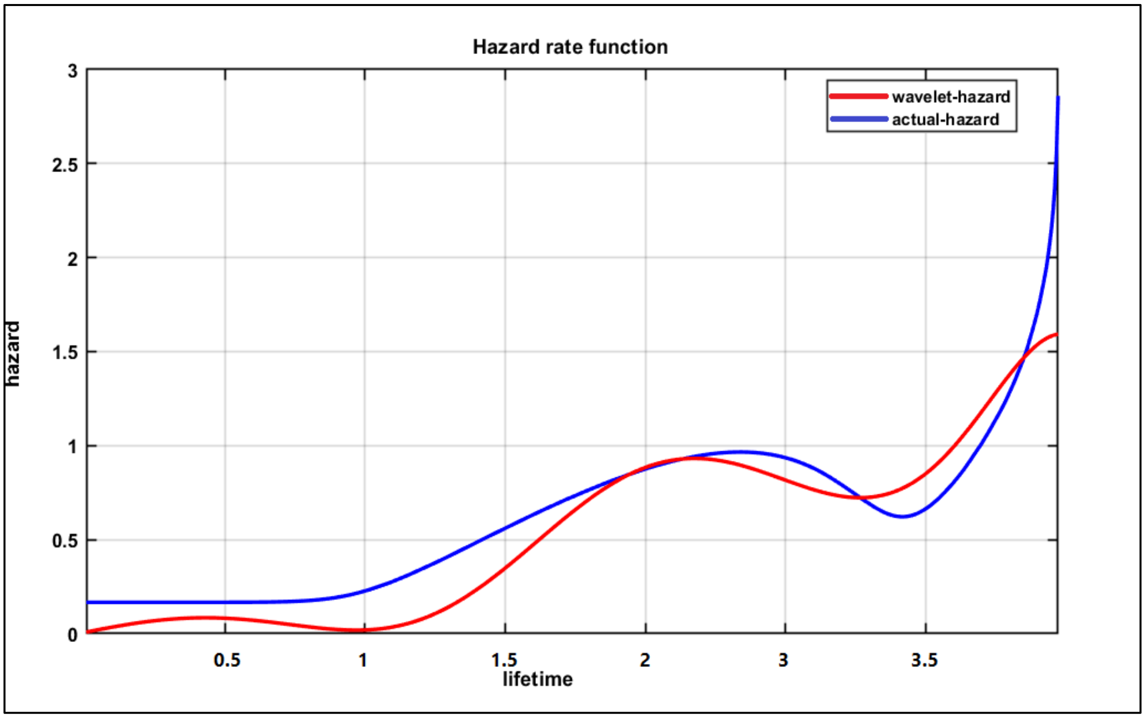


Figure (4-6): Estimation of hazard rate function of simulation experiment given in (4.2.2) with n=600.

From figure (4-5) noting that the estimation of hazard rate functions are convergent to actual hazard rat functions when n=400, but in figure (4-6) noting that the estimation of hazard rate functions are more convergent to actual hazard rat functions when n=600.

Table (4-4) showing the results of MSE for the bimodal simulation experiment of size sample n=400,600 with repetition R=500, in figures (4-7) and (4-8) the red curves represent the wavelet estimation of density functions and the blue curves represent the actual density functions of data generating by bimodal distribution and the independent censoring times generating by Exponential distribution for n=400,600.

n	MSE_Density
400	0.761276695
600	0.710094605

Table (4-4): MSE of density function for second simulation experiment

Table (4-4) showing that the MSE of probability density function estimation using wavelet (dN20 and $\hat{J} = 3$) and MSE is minimized especially for n=600 and R=500.

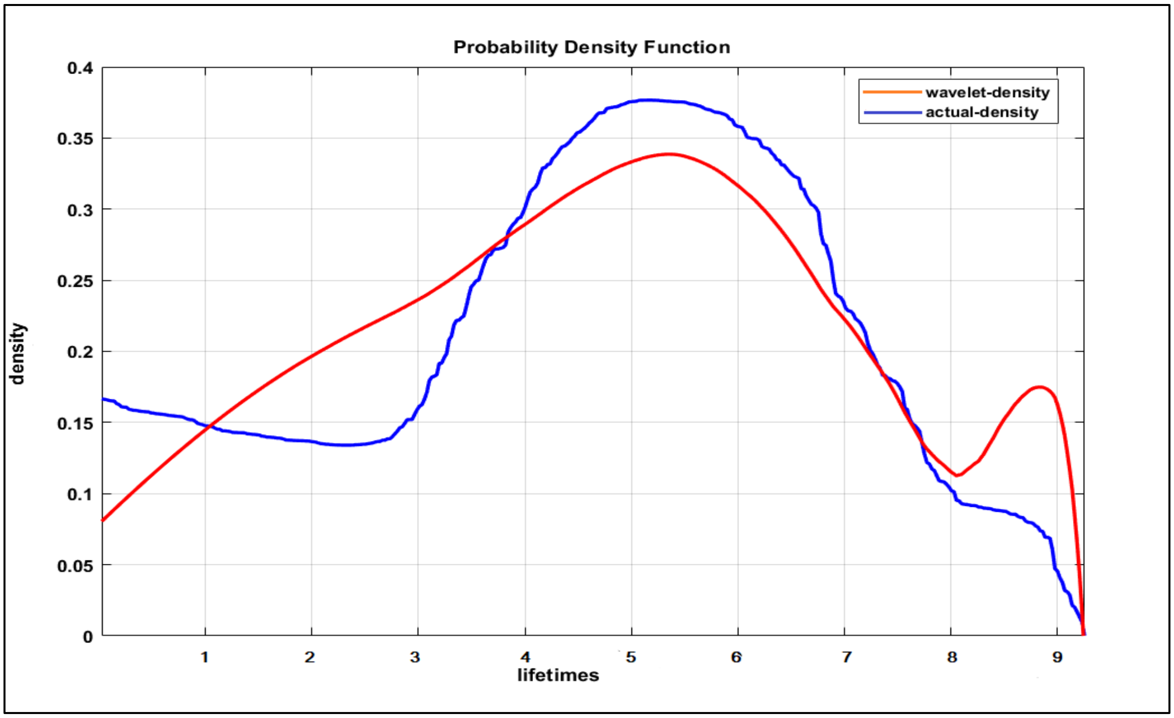


Figure (4-7): Estimation of density function of simulation experiment given in (4.2.2) with n=400.

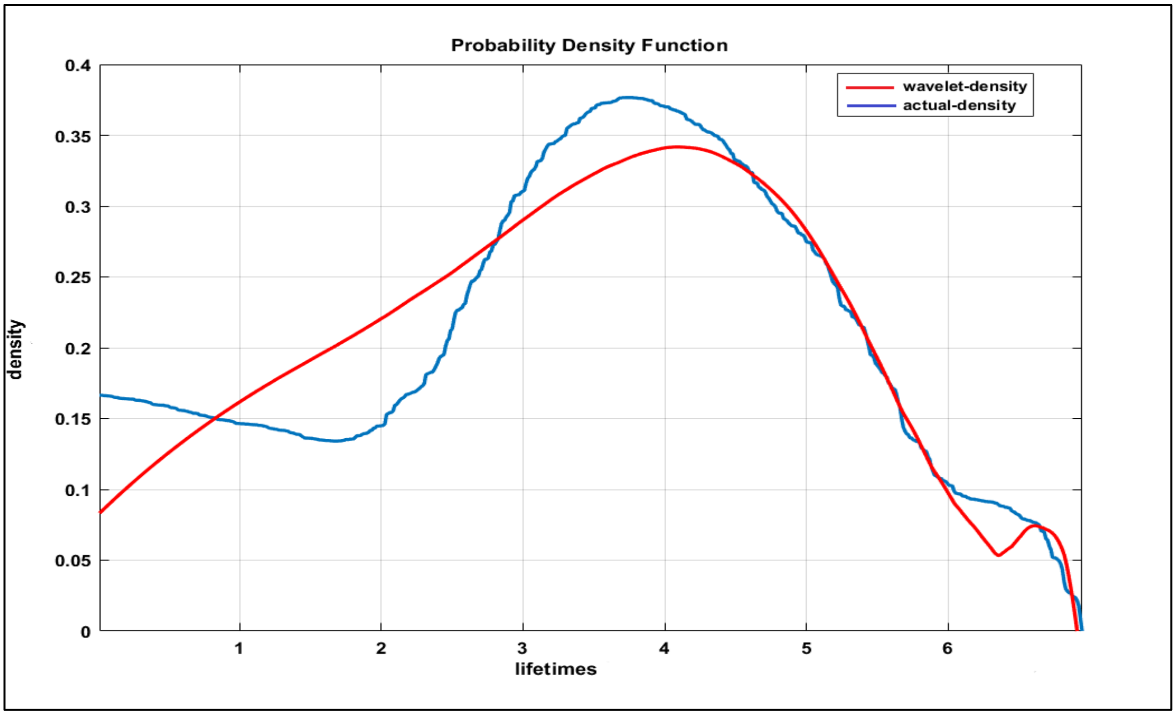


Figure (4-8): Estimation of density function of simulation experiment given in (4.2.2) with n=600.

From figure (4-7) noting that the estimation of probability density functions are convergent to actual hazard rat functions when n=400, but in figure (4-8) noting that the estimation of hazard rate functions are more convergent to actual hazard rat functions when n=600.

4.3 Real application data

This section will include three real applications to find the hazard rate function which are liver metastases data, nursing home data, and Stanford heart transplant data. Data were selected with different sample sizes for diversification to give options in terms of estimation.

As explained previously in Section (2.3), the estimation process is constrained by constraints that determine how to deal with the data in order to have the best compatibility between the constraints. Dealing with the real data in general is easier to deal with simulation data because it simply consists of one experiment (column), while simulation Because of repetitions consists of several experiments (columns).

For all this reasons here we can choose more than one type of wavelet in addition to choosing more than one value of the number of vanishing points for each type of wavelets. This provided an opportunity to make comparisons between the values of the estimates for each type of wavelet as well as the comparison between wavelet types.

4.3.1 Liver Metastases Data

The data is of 622 patients survival times suffering from liver metastases from a colorectal primary tumor collected by Haupt and Mansmann (1995). The survivals times of patients collected in months with 259 censored samples (41.62%). Moreover, the data is available in one of R program packages called locfit. We estimated the hazard function of the data using the wavelet method dependent on the wavelet level (Ĵ=3). Then the results were then compared with the results obtained from kernel smoothing estimation as shown in Figures (4-9), (4-10), and (4-11) for daubechies type with sizes of vanishing points (db30,db32, and db34), and figures (4-12), (4-13), and (4-14) for coilet type of sizes of vanishing points (coif1, coif2, and coif3). The blue curve represents the wavelet estimation, the red curve is kernel smoothing estimation, and the green curve represents the square error between wavelet and kernel estimations. In order to add more information about the estimation method, the MSE was calculated and shown in table (4-5).

The Square Error is formal is given as:

$$SE = \left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

The Mean Square Error is given as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

Wavelet type	MSE
db30	0.071486331
db32	0.012129634
db34	0.000693777
coif1	0.000506636
coif2	0.00610061
coif3	0.00285396

Table (4-5): MSE of kernel smoothing and wavelet estimations of hazard function for liver metastases data.

The table (4-5) showing that the MSE for the types of Daubechies wavelets are minimized especially for the type (db34) which equal to 0.000693777 and for Coiflet wavelets are minimized especially for the type (coif1) which equal to 0.000506636.

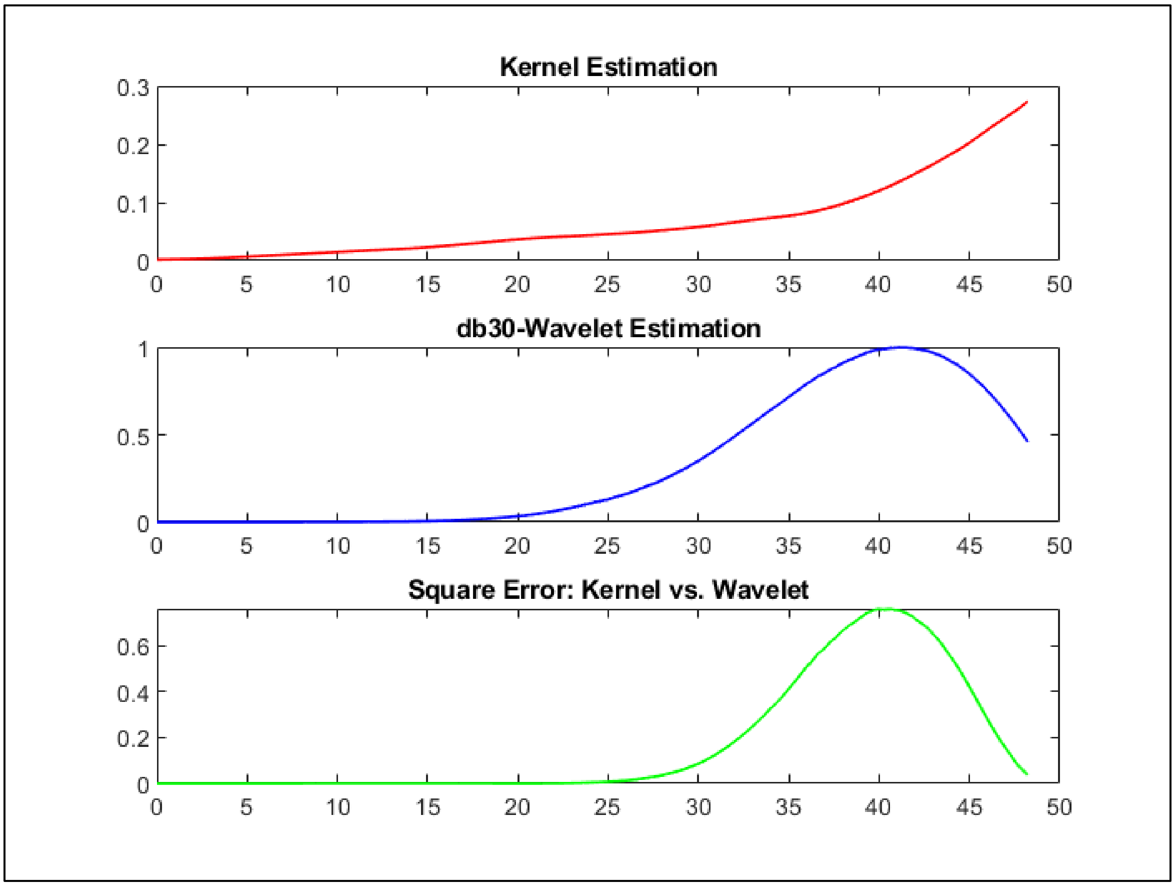


Figure (4-9): db30_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

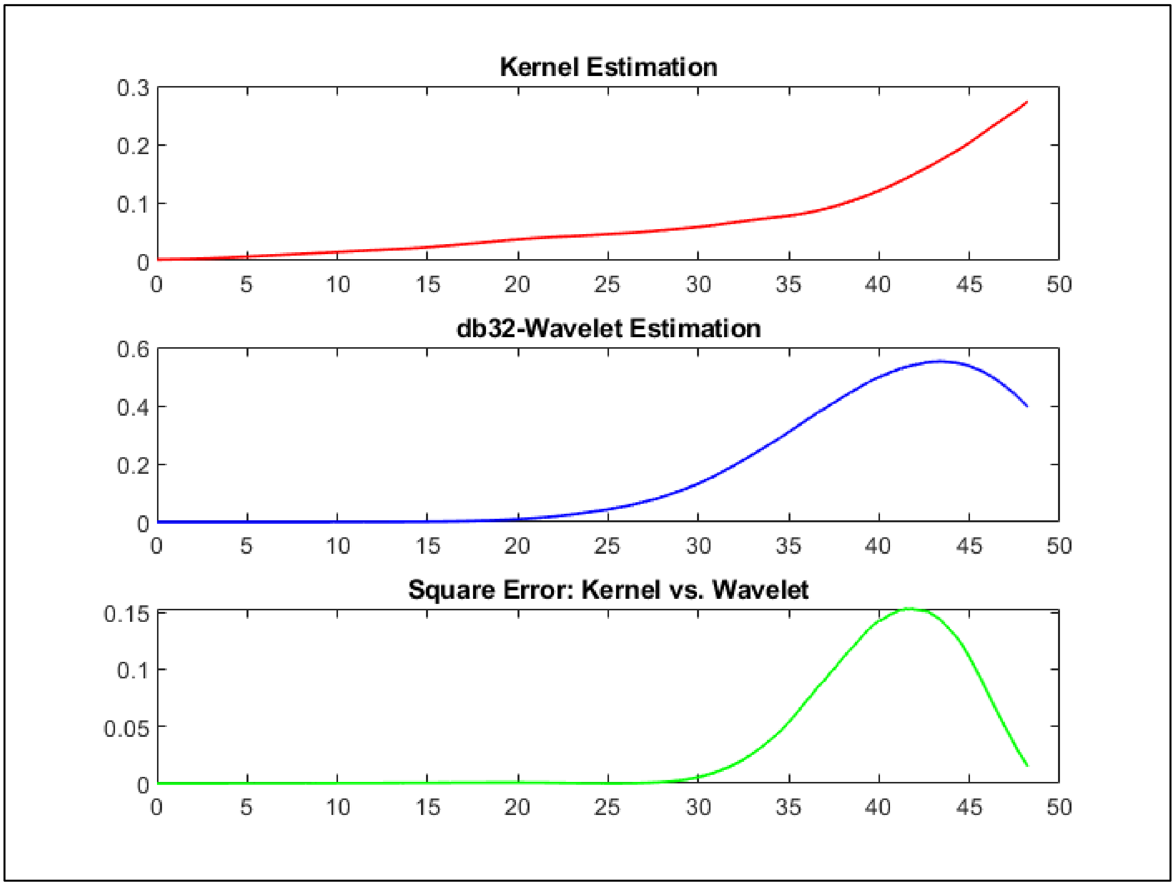


Figure (4-10): db32_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

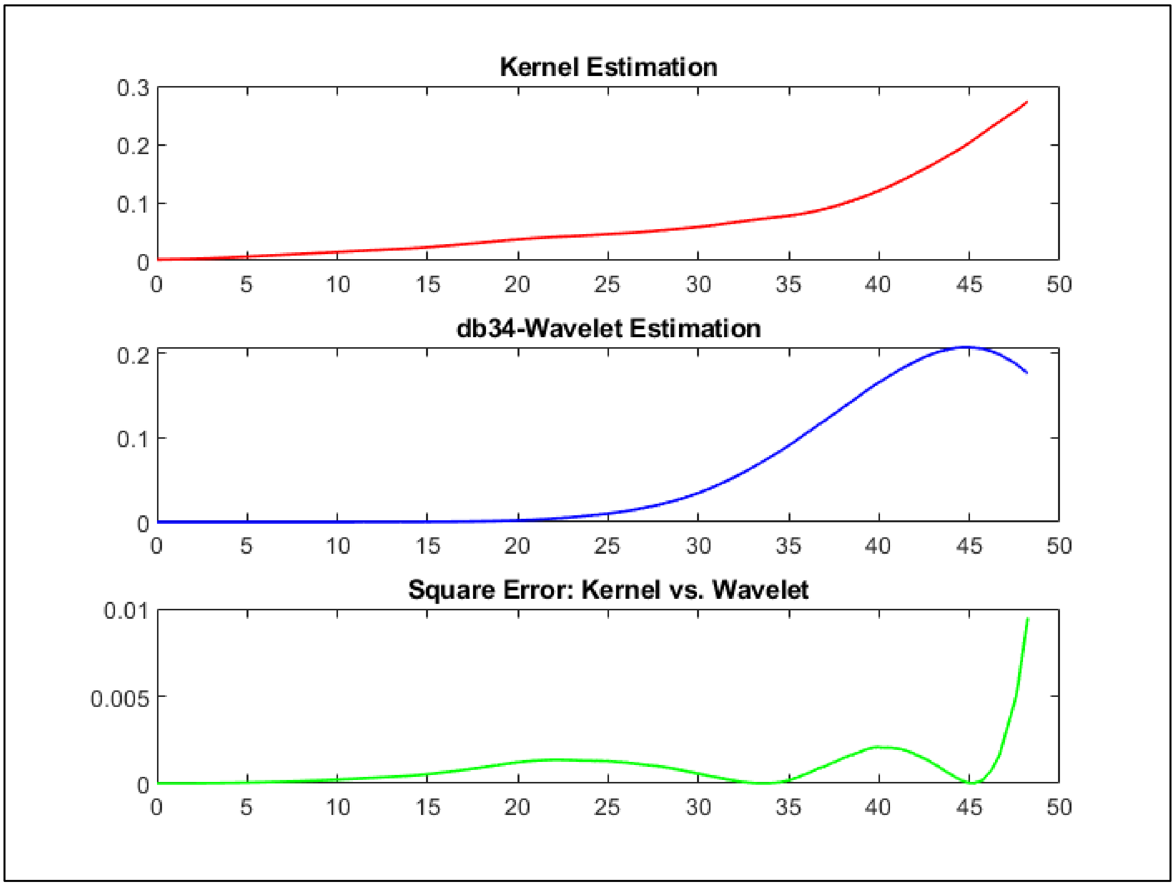


Figure (4-11): db34_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

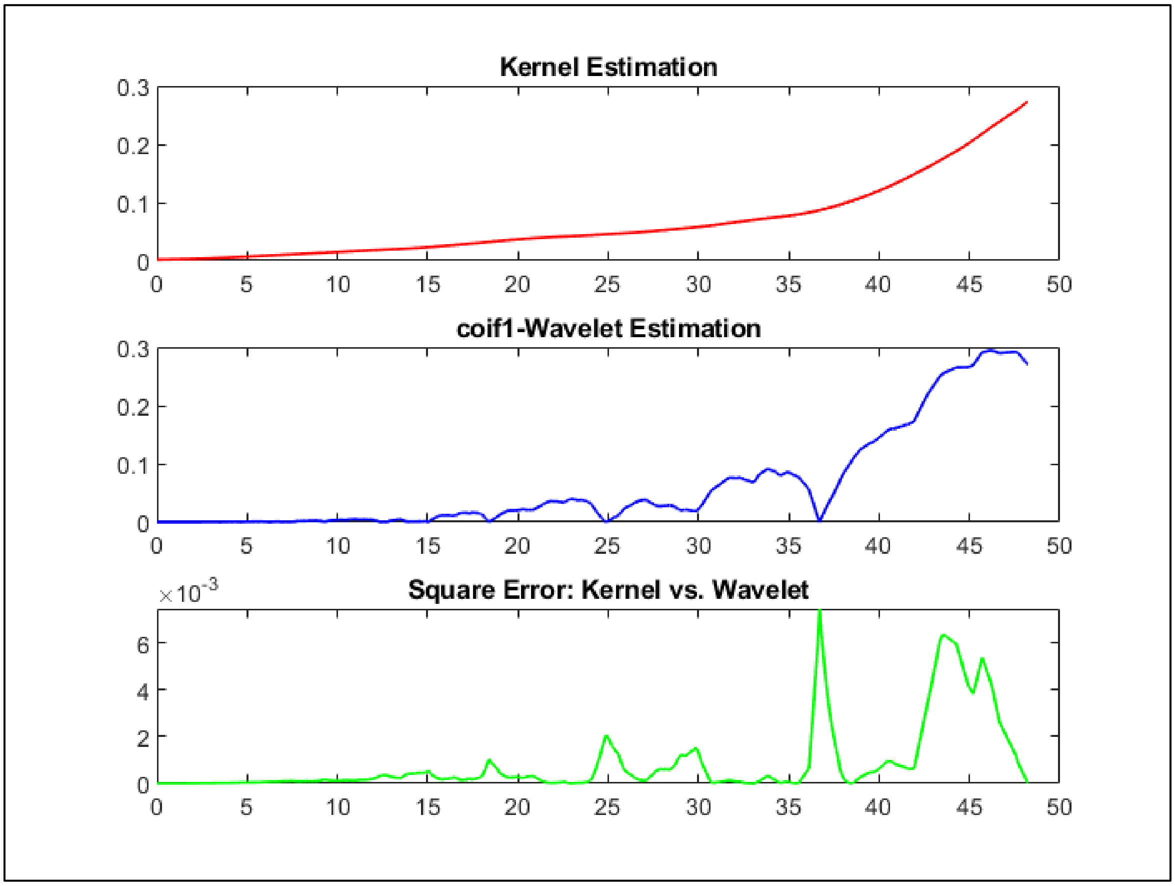


Figure (4-12): coif1_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

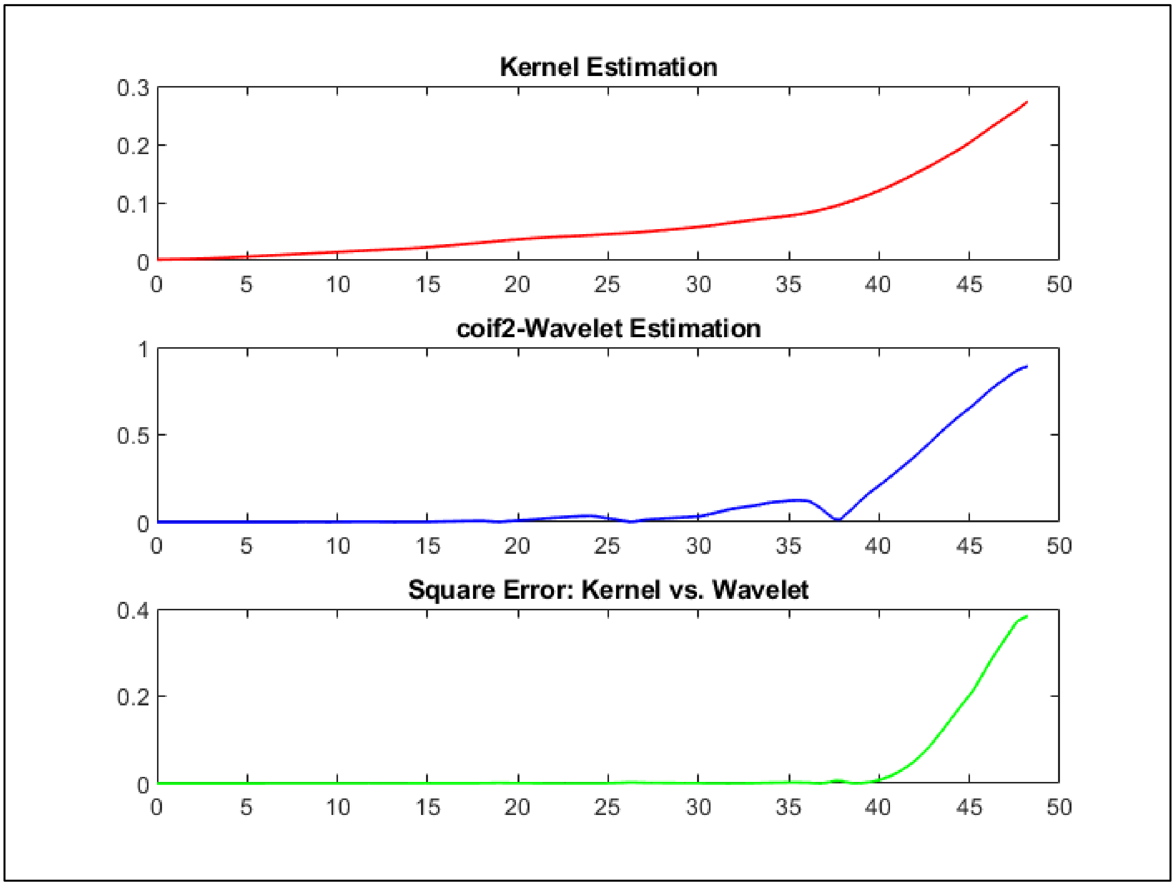


Figure (4-13): coif2_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

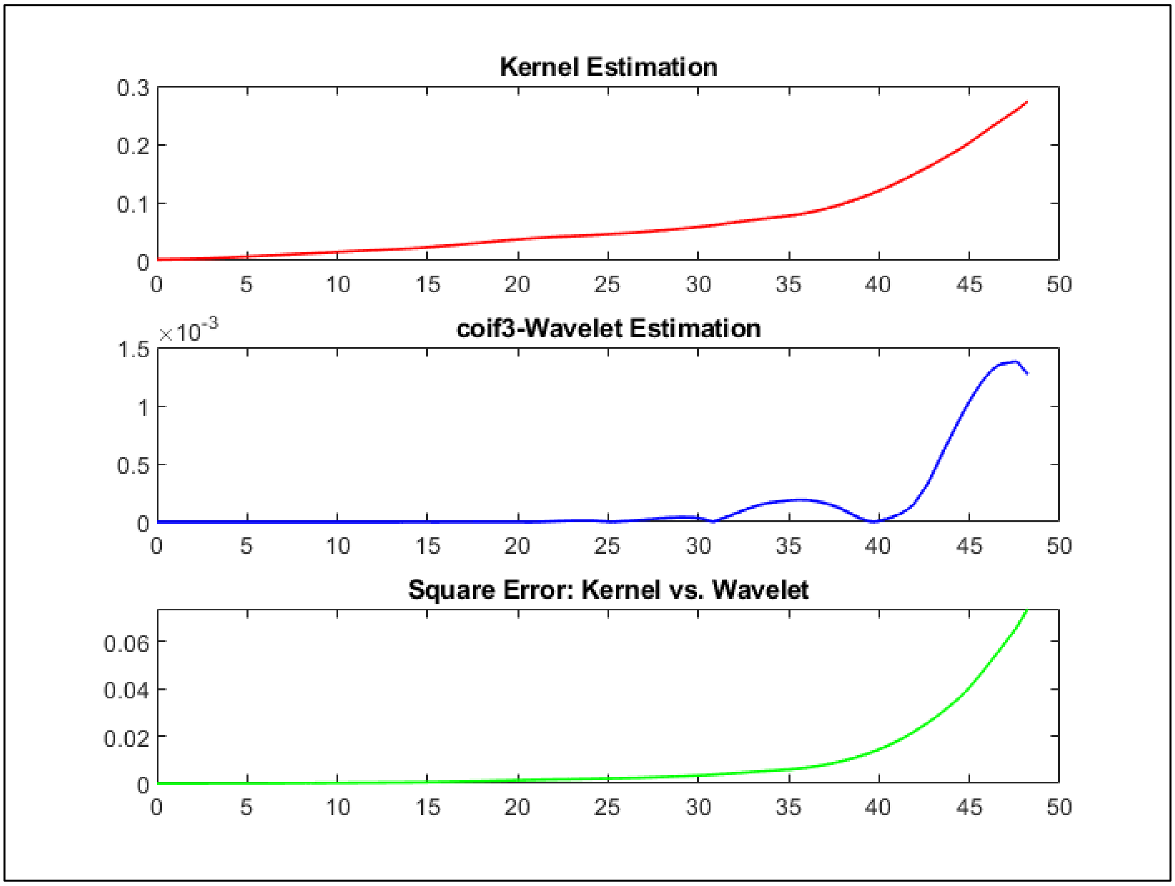


Figure (4-14): coif3_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 622 liver metastases data.

In the first three figures (4-9), (4,10), and (4-11), the using of the Daubechies wavelets showing the remarkable effect of the wavelet type in the estimation by observing the change in the estimation curves (blue curves) when the value of the vanishing points has changed, as well as the approximation of the estimation curve to kernel curve, which shows the effectiveness of the wavelets to obtain estimations as close as possible.

The last three figures (4-12), (4-13), and (4.14) of the using of Coiflet wavelets shows a slightly different approach from their predecessors due to a kind of oscillation in the values of the estimates which can be observed by the estimation curves (blue curves), oscillation can be observed more clearly in the square error curves (green curves), although the MSE appears to be of lower value when used the Coiflet wavelets.

4.3.2 Nursing Home Data

For data application, the data named is nursing home data, which was first introduced by Morris, Norton, and Zhou (1994). Data were collected for patients in a nursing home for the elderly between (1980-1982). The original study contains 1601 patients of home nursing and collected by the National Center for Health Services. For application using a subset of original data (n=214).

The approach with this data differs from its predecessor in terms of comparison where the kernel smoothing estimation method was used to estimate the hazard function. While, in wavelet estimation for hazard rate function, the type of wavelet was used is Daubechies with different seven sizes of vanishing points (db38,db40,db42,db44,db46,db48, and db50).

The following table (4-6) shows the Mean Square Error (MSE) for home nursing data between the kernel smoothing and wavelet estimation methods.

The Square Error is formal is given as:

$$SE = \left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

The Mean Square Error is given as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

Dbaubechies wavelets	MSE
db38	0.062838135
db40	0.029342693
db42	0.01081316
db44	0.003186177
db46	0.000739611
db48	0.00012561
db50	1.68099E-05

Table (4-6): MSE of kernel smoothing and wavelet estimations of hazard function for nursing home data.

The figures (4-15), (4-16), (4-17), (4-18), (4-19), (4-20), and (4-21) shows the curves or kernel smoothing estimation, wavelet estimation, and the square errors of nursing home data. Where the red curve represents the wavelet estimation, the blue curve represents the kernel smoothing estimation, and the green curve represents the square errors.

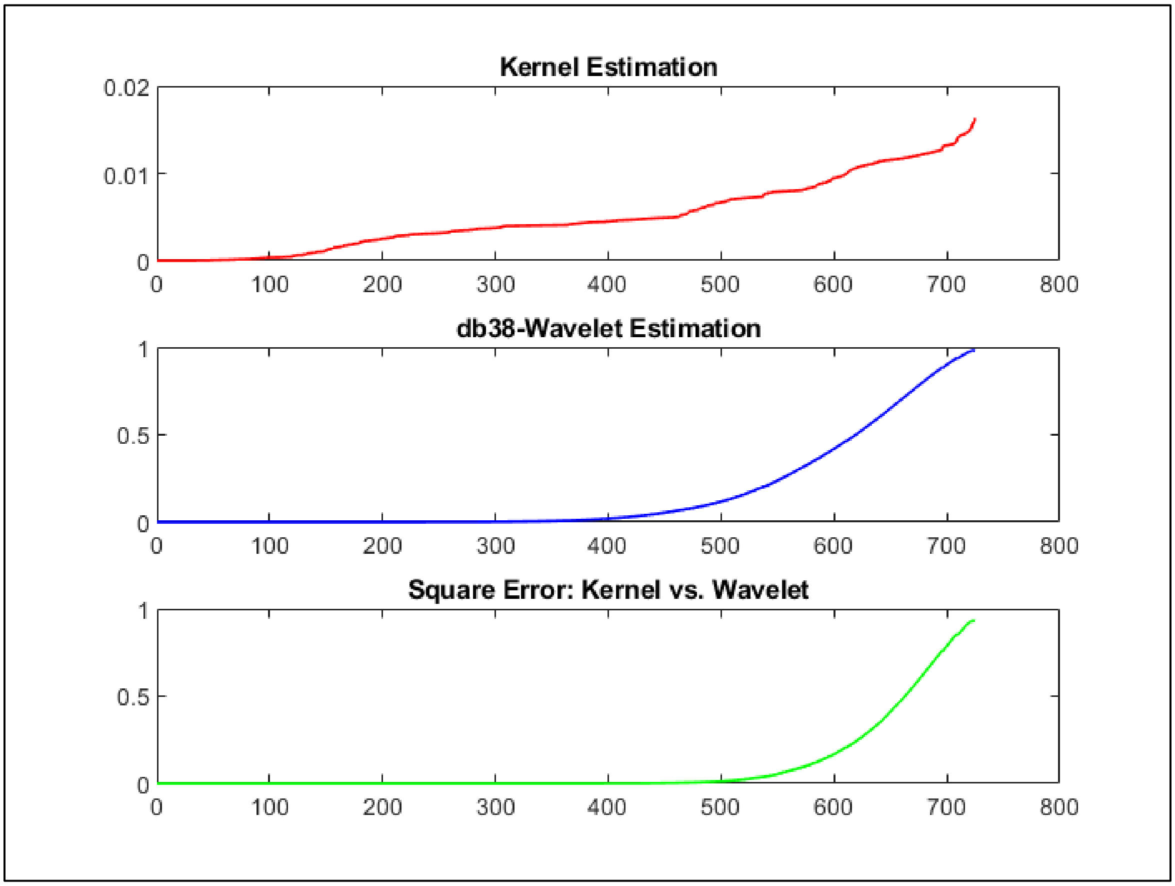


Figure (4-15): db38_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

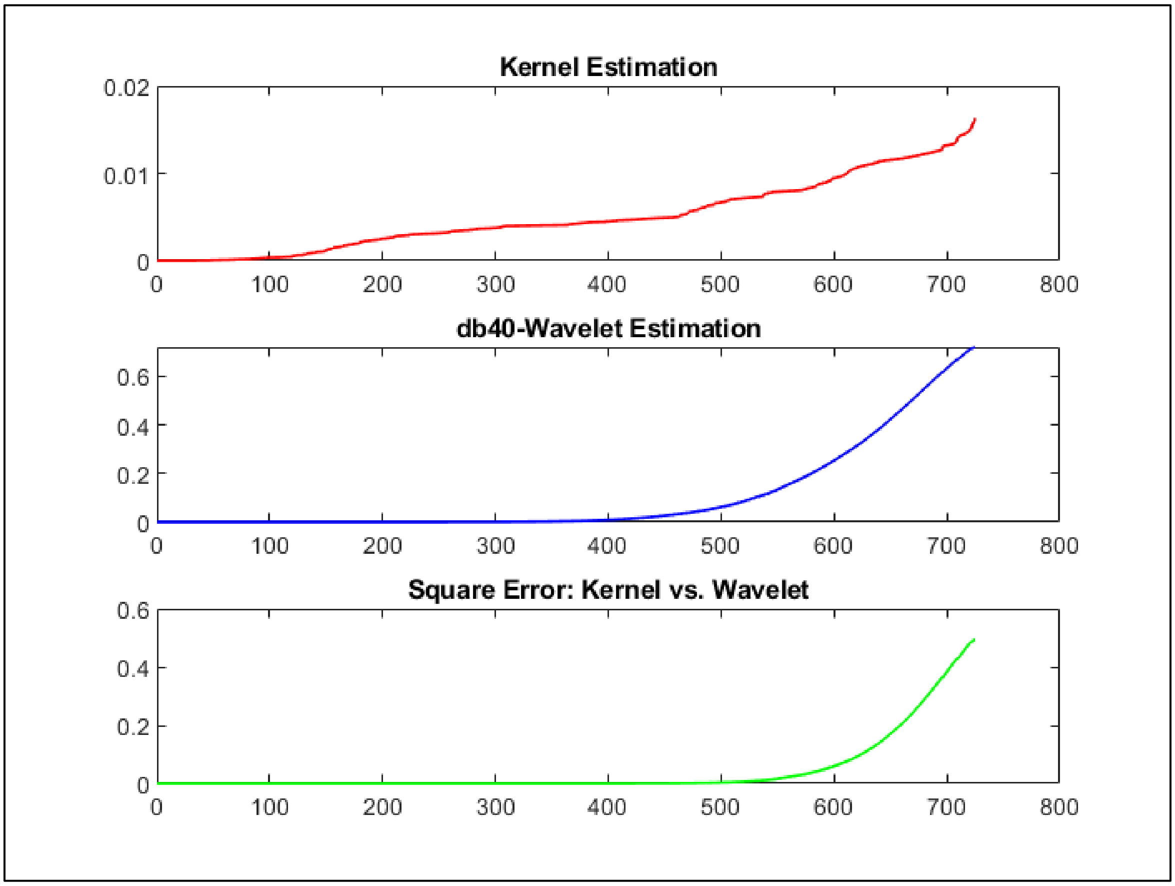


Figure (4-16): db40_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

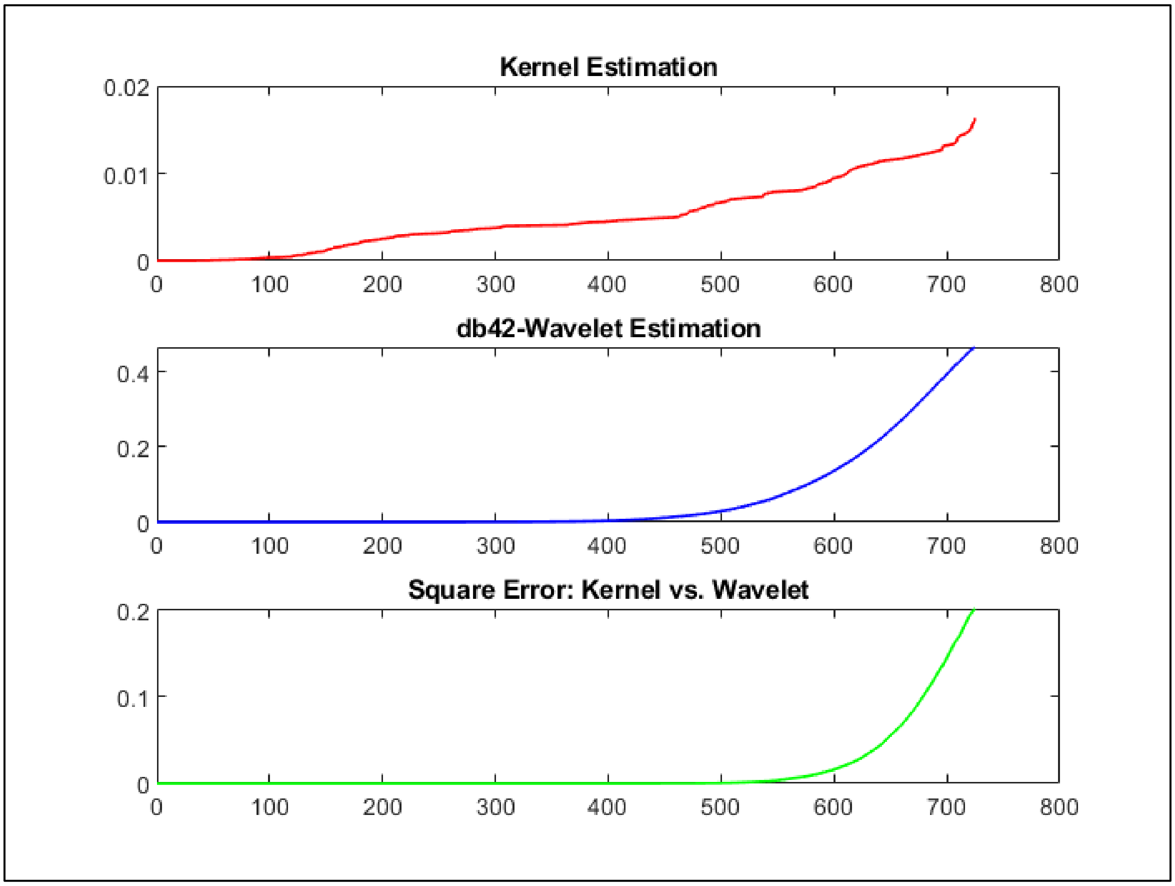


Figure (4-17): db42_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

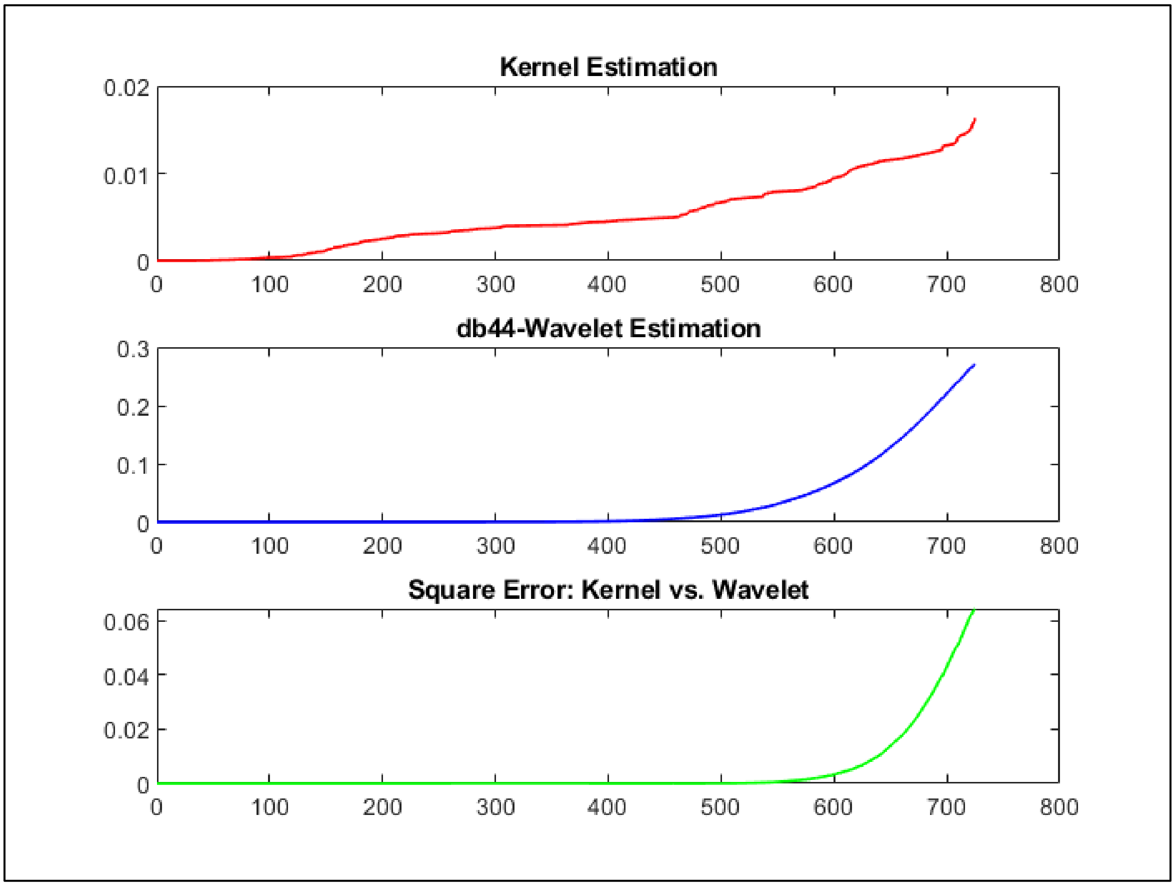


Figure (4-18): db44_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

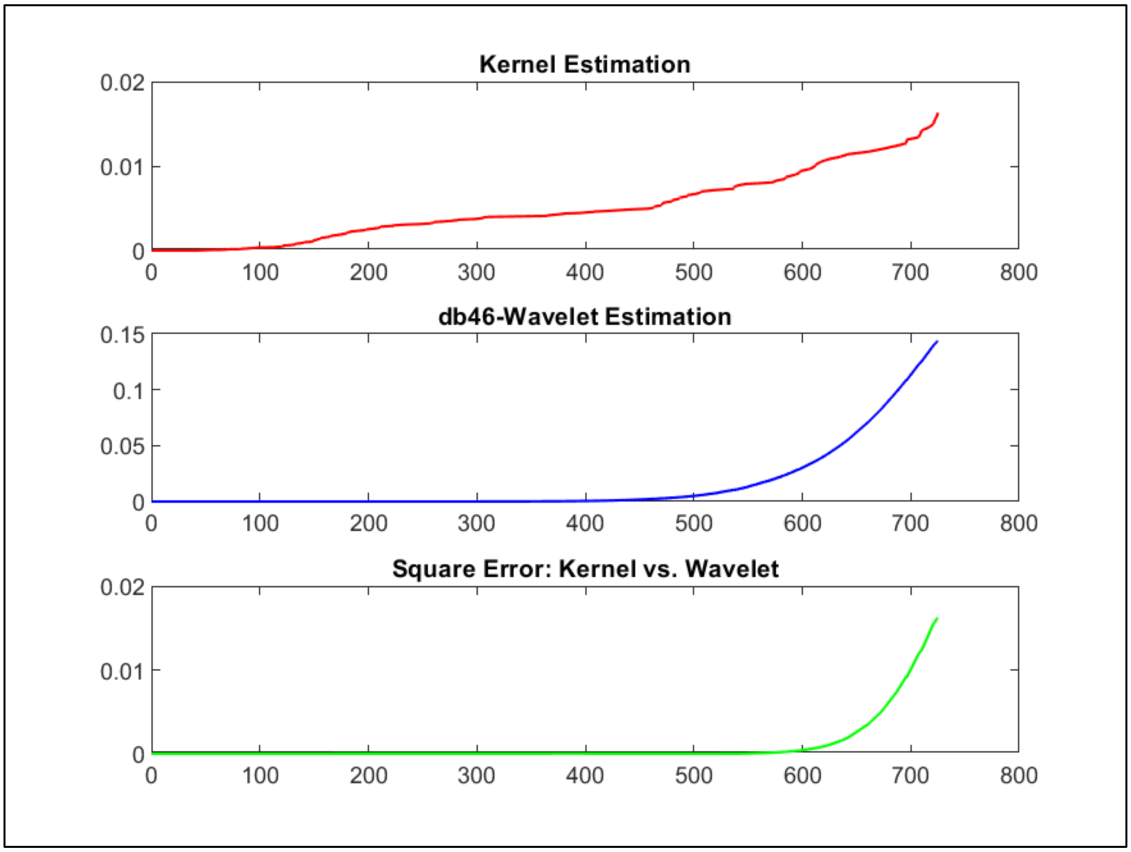


Figure (4-19): db46_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

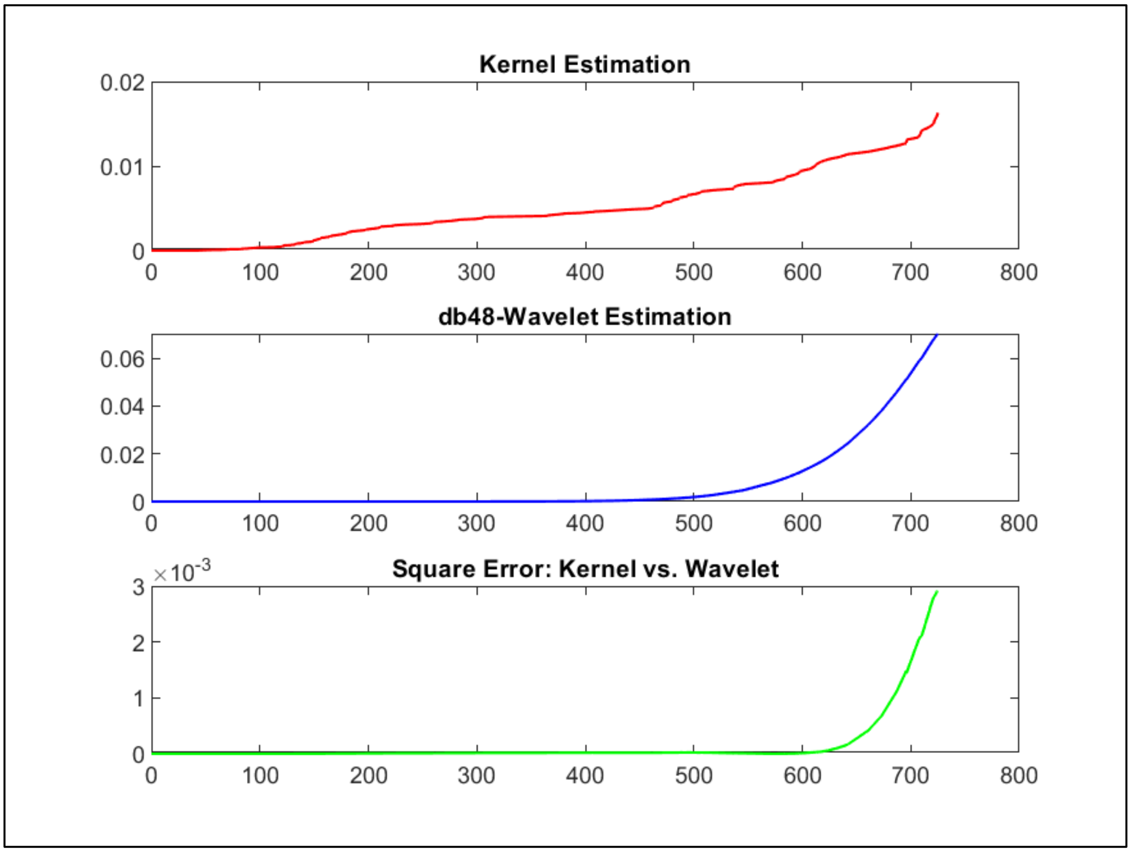


Figure (4-20): db48_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

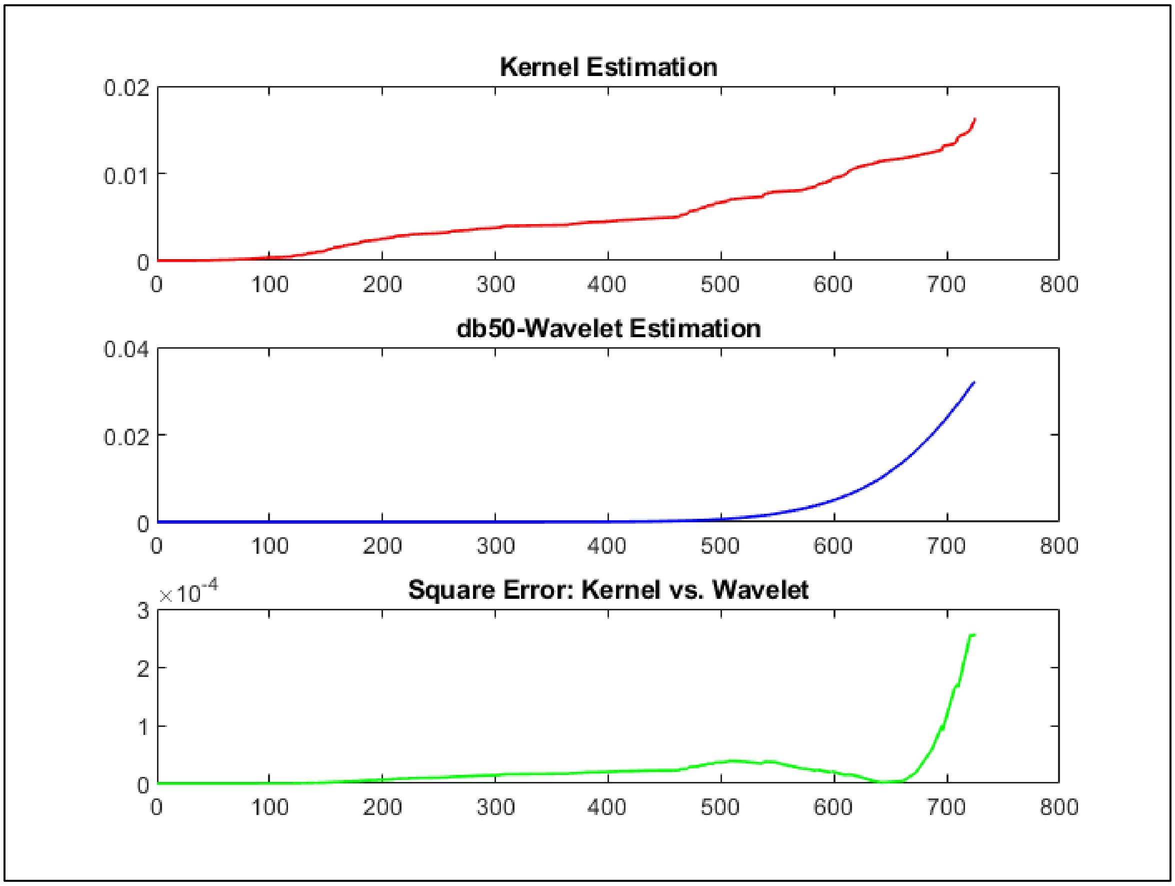


Figure (4-21): db50_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 214 nursing home data.

In this application, we try to show the change in the values of the estimates when using one type of wavelet with the possibility of a change in the number of vanishing points. This is an opportunity given by the lower sample size compared to the first application. The change in the number of vanishing points was serialized from 38 to 50, where the curve of the estimates showed an approach that takes approximation whenever the greater the level in the vanishing points, and all that was shown in figures (4-15), (4-16), (4-17), (4-18), (4-19), (4-20), and (4-21). This is confirmed by the value of MSE where it was the lowest possible when used of (db50) and equal to 1.68099E-05.

4.3.3 Stanford Heart Transplant data

For the application will use here real dataset called (Stanford Heart Transplant data), which he classifies them and analyzed (Crowley and Hu in 1977). The study was conducted between 1970 and 1974 and provides a survival time for 103 patients who need heart transplants. During this study, 67% (69 patients) received heart transfusion while the rest did not, at the end of the study in April 1974 73% (75 patients) died. Data representation includes the survival time in days, which is the difference between the last time the patient was seen and the time he entered the study. The survival status (censored status) formed as alive=0 and died=1. For more details about data, you can find it in (http://lib.stat.cmu.edu/datasets/stanford). In order to determine the appropriateness of the proposed method for estimating the hazard function, we will compare here the proposed method with a classical estimation method, which is kernel smoothing estimation, in order to collect the hazard rate function. Moreover, using the Mean Square Error (MSE) to compare between values of the proposed method and the kernel smoothing estimation.

The Square Error is formal given as:

$$SE = \left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

The Mean Square Error is given as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\lambda}(x_i) - \lambda(x_i)\right)^2$$

As mentioned before, the method used to choose the wavelet type and the value of vanishing points is an experimental method and since one of the constraints to be taken into consideration in the use of wavelets is the size of the sample in an attempt to use the most number of wavelet types with a different number of vanishing points and this is provided by this application It is possible to use three types of wavelets

The wavelet method used to estimate the hazard rate function was performed using three types of wavelets (Daubechies, Coiflit, and Symmlet) to compare the results. Each type of wavelets used with three sizes of vanishing points, for Daubechies (db40, db42, and db44), for Coiflet (coif2, coif3, and coif4), and for Symmlet (symm6, symm8, and symm10).

The following table (4-7) showed the MSE's values to denote each user type of wavelets compared with kernel smoothing estimation.

Wavelet type	MSE
db40	0.000123733
db42	7.45457E-06
db44	7.18966E-07
coif2	7.34425E-05
coif3	1.66059E-06
coif4	1.77922E-06
symm6	1.53041E-05
symm8	1.77525E-06
symm10	1.77932E-06

Table (4-7): MSE of kernel smoothing and wavelet estimations of hazard function for Stanford Heart Transplant data.

The figures (4-22), (4-23), (4-24) shows the curves of using daubechies, coiflet, and symmlet waveletes of smallest MSE values comparing to the kernel smoothing estimation, wavelet estimation, and the square errors of Stanford heart transplant data. Where the red curve represents the wavelet estimation, the blue curve represents the kernel smoothing estimation, and the green curve represents the square errors.

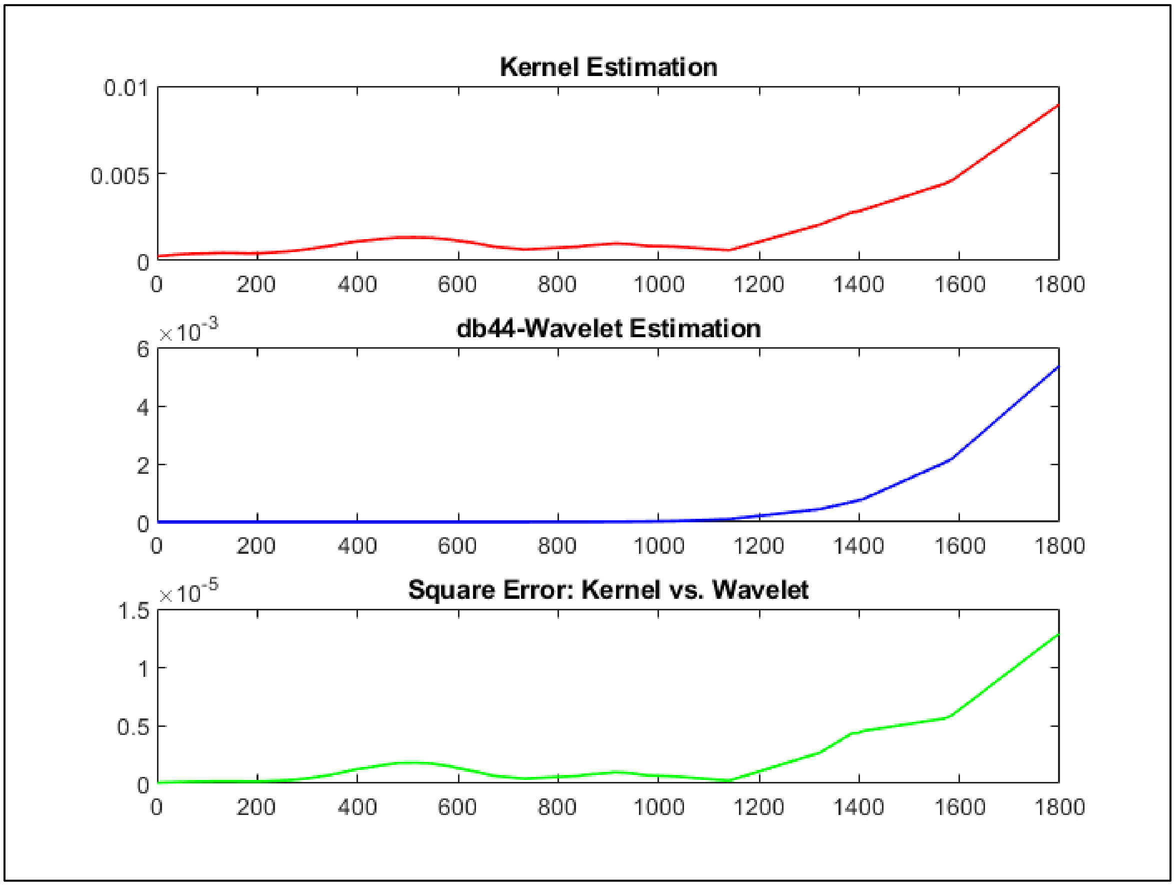


Figure (4-22): db44_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 103 Stanford heart transplant data.

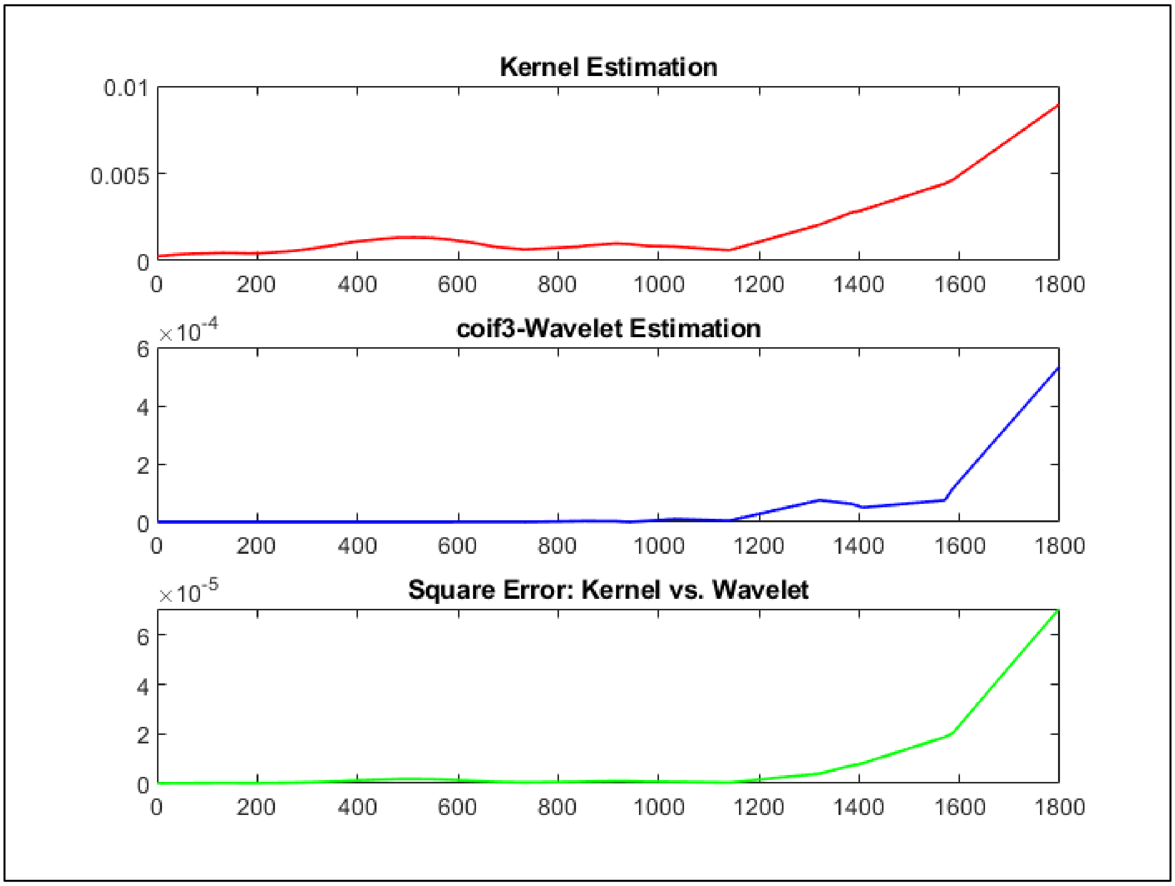


Figure (4-23): coif3_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 103 Stanford heart transplant data.

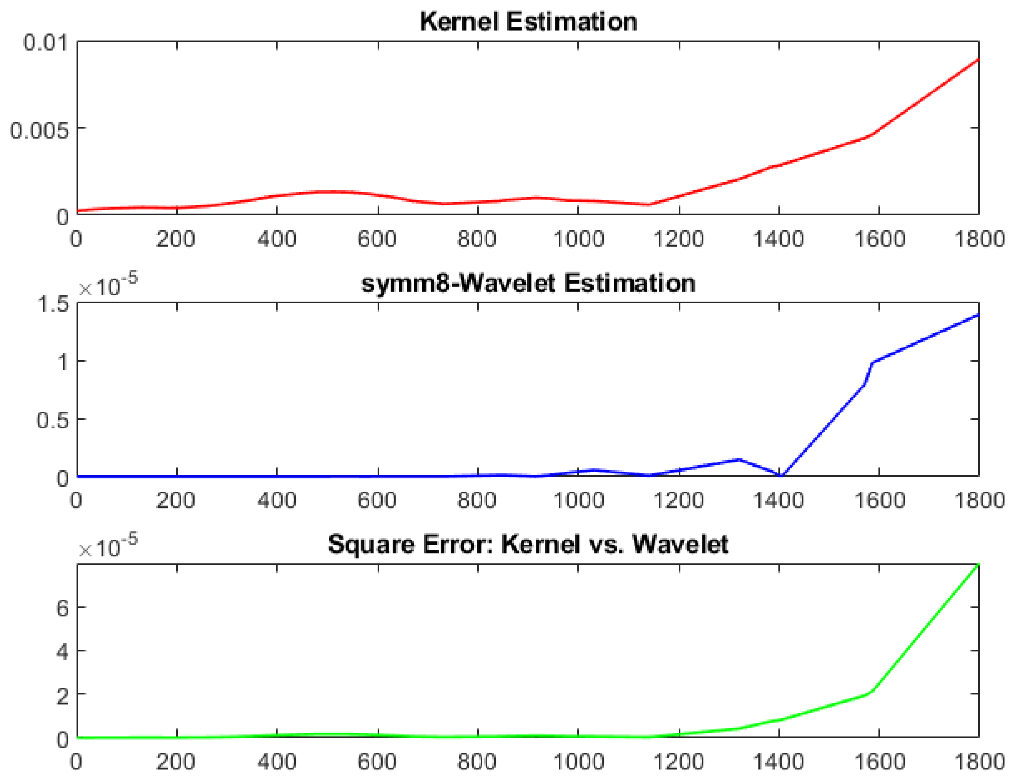


Figure (4-24): symm8_wavelet estimation (blue), kernel smoothing estimation (red), and square errors (green) of 103 Stanford heart transplant data.

In figures (4-22), (4-23), and (4-24), can be noting that the results of the estimations take a kind of stability when using a wavelet (db44) compared to the results that come with the use of (coif3) and (symm8). In addition, this stability and convergence between the values of the estimates proved by the smaller values of the Mean Square Error which equal to:

$$MSE_{db44} = 7.18966\mathrm{E} - 07$$

CHAPTER FIVE

CONCLUSION AND FUTURE STUDIES

5.1 Conclusion

1. The main object of this thesis is to study the nonparametric estimation of the hazard rate function for randomly right censored data using linear wavelet estimation.
2. The strategy to estimate hazard function follows partially estimation, at first estimate the probability density function denoted as$\left(\hat{f}^p(x)\right)$ and then estimate survival function denoted as$\left(\hat{S}^p(x) = 1 - \hat{F}(x)\right)$.
3. In order to estimate $\left(\hat{f}^p(x)\right)$, The wavelet projection method previously referred in equation (3-23) using the projection property of the father wavelet function $\left\{\Phi_{J,k}(x)\right\}, 0 \leq k \leq 2^J - 1, J \geq 0$ on the subspace V_J of Lebesgue spaces $L^2(R)$, followed by the creation of a hybrid between the wavelet and the Breslow estimate.
4. The survival function $\left(\hat{S}^p(x)\right)$ found using the empirical distribution function formula.
5. There are a set of constraints that control the estimation process which is the size of the sample, the wavelet type, the length of the wavelet, and the value of the resolution index (Ĵ).
6. Two types of applications were used, the first is the simulation study and the second is a real application.
7. First simulation generated of n=100, 200 samples used Gamma distribution (G(5,1)) for lifetimes $\{X_i\}_{i=1}^{n}$, and Exponential distribution (Exp(6)) for censoring times $\{C_i\}_{i=1}^{n}$. The type of wavelet was used in this simulation is Daubechies (db50).

8. Second simulation generated of n=400, 600 samples used bimodal distribution for lifetimes $\{X_i\}_{i=1}^{n}$, and Exponential distribution (Exp(6)) for censoring times $\{C_i\}_{i=1}^{n}$. The resolution index $(\hat{J})$ is choosing here and equal to (2 and 3). Furthermore, the wavelet type is choosing to be the Daubechies (dN20).
9. Three real applications used to study the estimation of hazard rate function which is liver metastases data, nursing home data, and stanford heart transplant data. Moreover, in real application for comparing with our method we used the nonparametric kernel smoothing estimation.
10. Because of the ease of dealing with real data, unlike the simulation data, the estimation process was done using three types of wavelets (Daubecheis, Coiflet, and Symmlet) with different wavelengths in order to know the effect of wavelets in the estimation process and make comparisons to reach the preferred result
11. In order to compare and determine the effectiveness of the method used in the estimation we used the Mean Square Error (MSE).
12. According to MSE results, in the simulation study we note that the sample size has an effect on the values of the results and the size of the larger sample gives the best results as we noted in tables (4-1) and (4-3), while the effect in the real data applications depends on the type of wavelet and wavelength in giving the best results as we noted in the tables (4-5), (4-6), and (4-7).
13. Finally, we use Matlab (17b and 18a) to program the proposal method of estimation the hazard rate function.

14. It is worth noting that the method used to reach to the best compatibility between the constraints (sample size, wavelet type, vanishing points, resolution index), which gives us an estimation results as best as possible, is an experimental method.

5.2 Future studies

1. Estimation the hazard rate function using non-linear wavelet estimation for randomly censored data.
2. Study the nonparametric estimation of hazard rate function using wavelets for different types of censored data such as left, interval, type I, and type II.
3. Hybrid between the wavelets estimation and the others nonparametric estimation such as Kaplan-Meier, Nalson-Aalen, and Kernel estimations.
4. Designation of a new mathematical style or system that allows for the best compatibility between the constraints (sample size, wavelet type, vanishing points, resolution index), which makes the estimation results to be the best as possible as it could

REFERENCES

1. Abbaszadeh, M., Chesneau, C. and Doosti, H. (2012) 'Nonparametric estimation of density under bias and multiplicative censoring via wavelet methods', *Statistics & Probability Letters*. Elsevier, 82(5), pp. 932–941.
2. Afshari, M. and Tahmasebi, S. (2014) 'Estimation of hazard function for censoring random variable by using wavelet decomposition and evaluation of MISE, AMSE with simulation', *Journal of Data Analysis and Information Processing*. Scientific Research Publishing, 2(01), p. 1.
3. Ahmadi, A. *et al.* (2012) 'Estimation of hazard function and its associated factors in gastric cancer patients using wavelet and kernel smoothing methods', *Asian Pacific Journal of Cancer Prevention*. Asian Pacific Organization for Cancer Prevetion, 13(11), pp. 5643–5646.
4. Angers, J.-F. and MacGibbon, B. (2007) *Bayesian estimation of the hazard function with randomly right censored data*. Wiley Online Library.
5. Antoniadis, A., Bigot, J. and Sapatinas, T. (2001) 'Wavelet estimators in nonparametric regression: a comparative simulation study', *Journal of statistical software*, 6, p. pp-1.
6. Antoniadis, A., Gregoire, G. and McKeague, I. W. (1994) 'Wavelet methods for curve estimation', *Journal of the American Statistical Association*, 89(428), pp. 1340–1353. doi:

10.1080/01621459.1994.10476873.

7. Antoniadis, A., Grégoire, G. and Nason, G. (1999) ‘Density and hazard rate estimation for right-censored data by using wavelet methods’, *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 61(1), pp. 63–84. doi: 10.1111/1467-9868.00163.
8. Bagkavos, D. (2011) ‘Local linear hazard rate estimation and bandwidth selection’, *Annals of the Institute of Statistical Mathematics*. Springer, 63(5), pp. 1019–1046.
9. Bezandry, P. H., Bonney, G. E. and Gannoun, A. (2005) ‘Consistent estimation of the density and hazard rate functions for censored data via the wavelet method’, *Statistics and Probability Letters*, 74(4), pp. 366–372. doi: 10.1016/j.spl.2005.04.058.
10. Brunel-Piccinini, E. and Comte, F. (2005) ‘Penalized contrast estimation of density and hazard rate with censored data’, *Sankhyā: The Indian Journal of Statistics*. JSTOR, pp. 441–475.
11. Brunel, E. and Comte, F. (2009) ‘Cumulative distribution function estimation under interval censoring case 1’, *Electronic Journal of Statistics*, 3, pp. 1–24. doi: 10.1214/08-EJS209.
12. CAI, J.-J. and LIANG, H.-Y. (2011) ‘Nonlinear Wavelet Density Estimation for Truncated and Dependent Observations’, *International Journal of Wavelets, Multiresolution and Information Processing*, 09(04), pp. 587–609. doi: 10.1142/s0219691311004237.
13. Chaubey, Y. P. *et al.* (2010) ‘Linear wavelet-based estimation for derivative of a density under random censorship’. JOURNAL OF THE IRANIAN STATISTICAL SOCIETY (JIRSS).
14. Chesneau, C., Dewan, I. and Doosti, H. (2016) ‘Nonparametric estimation of a quantile density function by wavelet methods’,

Computational Statistics and Data Analysis, 94, pp. 161–174. doi: 10.1016/j.csda.2015.08.006.

15. Chesneau, C. and Doosti, H. (2016) ‘A note on the adaptive estimation of a conditional continuous-discrete multivariate density by wavelet methods’, *Chinese Journal of Mathematics*. Hindawi, 2016.
16. Chesneau, C., Doosti, H. and Stone, L. (2018) ‘Adaptive wavelet estimation of a function from an m-dependent process with possibly unbounded m’, *Communications in Statistics - Theory and Methods*. Taylor & Francis, 0926, pp. 1–13. doi: 10.1080/03610926.2018.1423700.
17. Chesneau, C. and Willer, T. (2015) ‘Estimation of a cumulative distribution function under interval censoring “case 1” via warped wavelets’, *Communications in Statistics-Theory and Methods*. Taylor & Francis, 44(17), pp. 3680–3702.
18. Chui, C. K. and Jiang, Q. (2013) ‘Applied Mathematics’, *Data Compression. Spectral Methods, Fourier Analysis, Wavelets and Applications*. Springer.
19. Comte, F. *et al.* (2015) ‘Hazard estimation for censored data contaminated with additive measurement error : application to length of pregnancy’, (1), pp. 1–19.
20. Comte, F., Samson, A. and Stirnemann, J. J. (2017) ‘Hazard estimation with censoring and measurement error: application to length of pregnancy’, *TEST*. Springer, pp. 1–22.
21. Conraria, L. A. and Soares, M. J. (2011) ‘The continuous wavelet transform: A primer’, *NIPE Working Paper*. Universidade do Minho. Núcleo de Investigação em Políticas Económicas (NIPE), 16, pp. 1–43.

22. Daubechies, I. (1992) *Ten lectures on wavelets*. Siam.

23. Dixit, A. and Sharma, P. (2014) 'A Comparative Study of Wavelet Thresholding for Image Denoising', *IJ Image, Graphics and Signal Processing,* 12, pp. 39–46.

24. Donoho, D. L. *et al.* (1996) 'Density estimation by wavelet thresholding', *The Annals of Statistics*. JSTOR, pp. 508–539.

25. Doosti, H. (2009) 'A new wavelet linear survival function estimator', *World Applied Sciences Journal*, 6(2), pp. 191–195.

26. Farhadian, M. *et al.* (2014) 'Supervised wavelet method to predict patient survival from gene expression data', *The Scientific World Journal*. Hindawi, 2014.

27. Van Fleet, P. J. (2011) *Discrete wavelet transformations: An elementary approach with applications*. John Wiley & Sons.

28. Gaul, W. A. and Locarek-Junge, H. (2012) *Classification in the Information Age: Proceedings of the 22nd Annual GfKl Conference, Dresden, March 4–6, 1998*. Springer Science & Business Media.

29. Gomes, J. and Velho, L. (2015) *From fourier analysis to wavelets*. Springer.

30. Grez, G. A. S. and Vidakovic, B. (2017) 'An Empirical approach to Survival Density Estimation for randomly-censored data using Wavelets', 1(1997), pp. 1–31. Available at: http://arxiv.org/abs/1709.09298.

31. Grez, G. A. S. and Vidakovic, B. (2018) 'Least Squares Wavelet-based Estimation for Additive Regression Models using Non Equally-Spaced Designs', pp. 1–38. Available at: http://arxiv.org/abs/1804.03015.

32. Herrick, D. R. M., Nason, G. P. and Silverman, B. W. (2001) 'Some

new methods for wavelet density estimation', *Sankhyā: The Indian Journal of Statistics, Series A*. JSTOR, pp. 394–411.

33. In, F. and Kim, S. (2013) 'An introduction to wavelet theory in finance: a wavelet multiscale approach'. World scientific.

34. Jawerth, B. and Sweldens, W. (1994) 'An overview of wavelet based multiresolution analyses', *SIAM review*. SIAM, 36(3), pp. 377–412.

35. Jiang, Y. (2011) 'Estimation of hazard function for right truncated data'. Availbale at: https://scholarworks.gsu.edu/cgi/viewcontent.cgi?article=1095&context=math_theses

36. Kaiser, G. (2010) *A friendly guide to wavelets*. Springer Science & Business Media.

37. Kim, C. *et al.* (2005) 'Non-parametric hazard function estimation using the Kaplan-Meier estimator', *Journal of Nonparametric Statistics*, 17(8), pp. 937–948. doi: 10.1080/10485250500337138.

38. Kim, C. *et al.* (2010) 'A local linear estimation of conditional hazard function in censored data', *Journal of the Korean Statistical Society*. Elsevier, 39(3), pp. 347–355.

39. Lepik, Ü. and Hein, H. (2014) 'Haar wavelets', *Mathematical Engineering*, 1(9783319042947), pp. 7–20. doi: 10.1007/978-3-319-04295-4_2.

40. Li, L. (2002) 'Hazard rate estimation for censored data by wavelet methods', *Communications in Statistics - Theory and Methods*, 31(6), pp. 943–960. doi: 10.1081/STA-120004191.

41. Liang, H.-Y., Mammitzsch, V. and Steinebach, J. (2005) 'Nonlinear wavelet density and hazard rate estimation for censored data under dependent observations', *Statistics & Decisions*. Oldenbourg

Wissenschaftsverlag GmbH, 23(3/2005), pp. 161–180.

42. Liu, C.-L. (2010) 'A tutorial of the wavelet transform', *NTUEE, Taiwan*.

43. Luo, G. and Zhang, D. (2012) 'Wavelet denoising', in *Advances in wavelet theory and their applications in engineering, physics and technology*. IntechOpen.

44. Misiti, M. *et al.* (2014) *Wavelet Toolbox^{TM} User's Guide R 2014 b, MATLAB version*.

45. Naono, K. (1995) 'Comparative computations of non-parametric density estimation between some kernel method and the wavelet method *', *Monte Carlo Methods and Applications*, 1(2), pp. 147–163. doi: 10.1515/mcma.1995.1.2.147.

46. Olkkonou, H. (2011) 'Discrete wavelet transforms: Algorithms and applications'. Intech.

47. Pinsky, M. A. (2008) *Introduction to Fourier analysis and wavelets*. American Mathematical Soc.

48. Plancade, S. (2009) 'Non parametric estimation of hazard rate in presence of censoring'.

49. Rinne, H. (2014) *The Hazard rate: Theory and inference (with supplementary MATLAB-Programs)*.

50. Rodrıguez, G. (2005) 'Non-parametric estimation in survival models'. Available at: https://data.princeton.edu/pop509/NonParametricSurvival.pdf.

51. Ruch, D. K. and Van Fleet, P. J. (2011) *Wavelet theory: an elementary approach with applications*. John Wiley & Sons.

52. Salha, R. B., Ahmed, H. I. E. S. and Alhoubi, I. M. (2014) 'Hazard Rate Function Estimation Using Weibull Kernel', *Open Journal of*

Statistics. Scientific Research Publishing, 4(08), p. 650.

53. Schlossnagle, G., Restrepo, J. M. and Leaf, G. K. (no date) 'TECHNICAL REPORT ANL-93/34 PERIODIZED WAVELETS'.

54. Shirazi, E. *et al.* (2012) 'Wavelet based estimation for the derivative of a density by block thresholding under random censorship', *Journal of the Korean Statistical Society*. Elsevier, 41(2), pp. 199–211.

55. Sifuzzaman, M., Islam, M. R. and Ali, M. Z. (2009) 'Application of Wavelet Transform and its Advantages Compared to Fourier Transform', *Journal of Physical Sciences*, 13, pp. 121–134. doi: 10.5194/hess-15-1835-2011.

56. Vannucci, M. (1995) *Nonparametric density estimation using wavelets*. Institute of Statistics & Decision Sciences, Duke University.

57. Varanis, M. and Pederiva, R. (2017) 'The influence of the wavelet filter in the parameters extraction for signal classification: An experimental study.', *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, 5(1).

58. Weng, Y.-P. and Wong, K.-F. (2007) 'Baseline survival function estimators under proportional hazards assumption', *Institute of Statistics, National University of Kaohsiung, Kaohsiung, Taiwan*, 811.

59. Wu, S. S. and Wells, M. T. (2003) 'Nonparametric estimation of hazard functions by wavelet methods', *Journal of Nonparametric Statistics*, 15(2), pp. 187–203. doi: 10.1080/1048525031000089301.

60. Zhang, Z. *et al.* (2014) 'Achieving complex discrete wavelet transform by lifting scheme using Meyer wavelet', *International Conference on Wavelet Analysis and Pattern Recognition*, 2014-Janua, pp. 170–175. doi: 10.1109/ICWAPR.2014.6961310.

APPENDIX

PROGRAMING

Estimation of hazard rate function using wavelet

```
n=…;                         % samples size
J=3;                         % default coarsest
level
tt=0;sd=0;sd1=0;sd2=0;sd3=0;
%------father Wavelet function My_pHjk-------
filter = MakeONFilterExt('symmlet',11);  % wavelet
filters
%filter = coifwavf('coif2');
%filter=symwavf('sym6');
%filter=dbwavf('db5');
%filter=symwavf('bior1.4');

y=x(:,1);                     % column of lifetimes
observations
d=x(:,2);            % column of censoring and
uncensoring cases
M=zeros(n,2^J);      % father wavelet function
matrix
for k = 0:(2^J-1)
    for i = 1:n
       M(i,k+1)=Phijk(y(i),J,k,filter,20);
    end
end
% -------berslow_estimayor-----

b=zeros(n,1);

%--------estimation of G(Yi)------
for i1=1:n
   for j=1:i1-1
       b(j)=(1-d(j))/(n-j+1);
        c(i1)=b(j);
    end
  end
```

```
 for k1=1:n
     sd=sd+c(k1);
     u(k1)=sd;
  for f=1:k1
       a(f)=(1-d(f))/(n-f+1);
  end
   m(k1)=a(k1)*exp(-u(k1));
   sd1=sd1+m(k1);
   v(k1)=sd1;
 end

 %------ estimation of F(Yi)------
 for i6=1:n
   for j1=1:i6-1
       b1(j1)=(d(j1))/(n-j1+1);
        c1(i6)=b1(j1);
    end
  end
 for k2=1:n
     sd2=sd2+c1(k2);
     u1(k2)=sd2;
  for f1=1:k2
       a1(f1)=(d(f1))/(n-f1+1);
  end
   m1(k2)=a1(k2)*exp(-u1(k2));
   sd3=sd3+m1(k2);
   v1(k2)=sd3;
 end
 %------phij cofficient (Cjk)-----
 for i2=1:n
   A1(i2)=1/(1-v(i2));
   A2(i2)=1-v1(i2);

 end
  A3=(A1)*(diag(A2)*diag(1-d));
A=(1/n)*(A1-A3);
A;
for i3=1:2^J
    cjk(i3)=(A)*(M(:,i3));
end
```

```
cjk;

%---------density function estimator------
for i4=1:2^J
    f2(:,i4)=cjk(i4)*M(:,i4);
end
f2;
for j2=1:2^J
    tt=tt+f2(:,j2);
end
ff=(tt);
ff;
% ------ empirical estimation of S(Yi)-----
R=ones(n);
F=tril(R);
for r=1:n
    Q(r)=1-((1/(n+1))*sum(F(1:n,r)));
End

%--------hazard estimator-----
for i5=1:n
FF(i5)=ff(i5)/Q(i5);
end
F1=abs(FF');

%--------------------------------------------------
```

Gamma simulation random samples

```
m=…;                %---number of Repetitions---
n=…;                %---size of samples --------
% ------generates random samples from the gamma
distribution with shape parameters =5 and scale
parameters =1%------
x=gamrnd(5,1,[n m]);
%-----generates random samples from the exponential
distribution with mean parameter =6%-------
c=exprnd(6,[n m]);
 for j=1:m
        for i=1:n
```

```
          if x(i,j)<=c(i,j)
              d(i,j)=1;
          else
              d(i,j)=0;
          end
          y(i,j)=min(x(i,j),c(i,j));
        end
 end
%--------------------------------------------------
```

Hazard rate function for Gamma distribution

```
n=…;                     %----size of samples --------
m=…;                     %----number of Repetitions---
for j=1:m
    for i=1:n
den(i,j)=((pdf('gamma',yy(i,j),5,1))*(1cdf('exponen
tial',yy(i,j),6)))+((pdf('exponential',yy(i,j),6))*
(1-cdf('gamma',yy(i,j),5,1)));
sur(i,j)=((1cdf('gamma',yy(i,j),5,1)))*(1cdf('expon
ential',yy(i,j),6));
    end
end
for jj=1:m
    for ii=1:n
         haz(ii,jj)=den(ii,jj)/sur(ii,jj);
    end
end
%--------------------------------------------------
```

Bimodal simulation random samples

```
% The number of random samples to generate
N =…;
%---number of Repetitions---
M=…;
% First, generate a binomial variable, which 40% of
the time (on average) will be equal to 1,
% and 60% of the time (on average) will be equal to
zero.
```

```
frac = rand(N,1) < 0.4;
% Generate the two normal distributions to sample
from
norm1 = 5 + randn(N,M);
norm2 = 3 + 0.5*randn(N,M);
% If "frac" is equal to 1, then the choice will be
from the first normal.
% If "frac" is equal to 0, then the choice will be
from the second normal.
d = frac.*norm1 + (1-frac).*norm2;
% Plot the resulting distribution
figure
histogram(d)
c=exprnd(6,[N,M]);
%---------------------------------------------------
```

Hazard rate function for bimodal distribution

```
alpha=0.4;           %----weight factor-----------
n=…;                 %----size of samples --------
m=…;                 %----number of Repetitions---
for k=1:m
    for t=1:n
density(t,k)=(alpha*(pdf('normal',yy(t,k),5,1)))+((
1-alpha)*(pdf('normal',yy(t,k),3,0.5)));
Cdf(t,k)=(alpha*(cdf('normal',yy(t,k),5,1)))+((1-
alpha)*(cdf('normal',yy(t,k),3,0.5)));
hazard(t,k)=density(t,k)/(1-Cdf(t,k));
    end
end
%for j=1:m
   % for i=1:n
for j=1:m
    for i=1:n
den(i,j)=((density(i,j))*((1-
cdf('exponential',yy(i,j),6))))
+((pdf('exponential',yy(i,j),6))*(1-Cdf(i,j)));
sur(i,j)=(1-Cdf(i,j))*(1-
cdf('exponential',yy(i,j),6));
```

```
haz(i,j)=den(i,j)/sur(i,j);
    end
end
%--------------------------------------------------
```

Indicator function

```
n=…;                  %----size of samples --------
m=…;                  %----number of Repetitions---
for j=1:m
    for i=1:n
        y(i,j)=min(x(i,j),c(i,j));
    end
end
for j1=1:m
for i1=1:n
if x(i1,j1)<=c(i1,j1)
d(i1,j1)=1;
else d(i1,j1)=0;
    end
  end
end
%--------------------------------------------------
```

Father wavelet function (Phijk)

```
 function yy = Phijk(z, j, k, filter, n)
%--------------------------------------------------
--------
 %  yy=Phijk(z, j, k, filter, n)
 %  Evaluation of the scaling function
 corresponding to an Orthogonal
 %  inputs:  z -- the argument
 %           j - scale
 %           k - shift
 %           filter -- ON finite wavelet filter,
might be an
 %                     output of WaveLab's:
 MakeONFilter
 %           n -- precision of approximation
 maesured by the       number (n=20)
 %  output:  yy -- value of father wavelet (j,k)
coresponding to
 %                  'filter' at z.
%--------------------------------------------------
    “if (nargin == 4)”
        n=20;
    end
     daun=length(filter)/2;
     N=length(filter)-1;
     x=(2^j)*z-k;
     if(x<=0||x>=N) yy=0;
else
     int=floor(x);
     dec=x-int;
     dy=dec2bin(dec,n);
     t0=t0(filter);
     t1=t1(filter);
     prod=eye(N);
     for i=1:n
             if dy(i)==1 prod=prod*t1;
             else prod=prod*t0;
            end
    end
     y=2^(j/2)*prod;
       yyy = mean(y');
       yy = yyy(int+1);
     end

  %----------functions ----------
 function a = dec2bin(x,n)
            a=[];
             for i = 1:n
              if(x <= 0.5) a=[a 0]; x=2*x;
               else a=[a 1]; x=2*x-1;
             end
            end
 %-----------
  function t0 = t0(filter
%
 n = length(filter);
 nn = n - 1;
 t0 = zeros(nn);
 for i = 1:nn
     for j= 1:nn
         if (2*i - j > 0 & 2*i - j <= n)
            t0(i,j) = sqrt(2) * filter( 2*i - j );
        end
    end
end
%------------------
 function t1 = t1(filter)
 n = length(filter);
 nn = n - 1;”
 t1 = zeros(nn);
 for i = 1:nn
     for j= 1:nn
         if (2*i -j+1 > 0 & 2*i - j+1 <= n)
          t1(i,j) = sqrt(2) * filter( 2*i - j+1 );
        end
    end
end
%--------------------------------------------------
```

Wavelet filters

```
 function f = MakeONFilterExt(Type,Par)
%-----------------------------------------------------%
 if strncmpi(Type,'Haar',2),
      f = [1 1];
end

 if strncmpi(Type,'Beylkin',2),
       f = [      .099305765374,    .424215360813,    .699825214057
      ...
                  .449718251149,   -.110927598348,   -.264497231446
      ...
                  .026900308804,    .155538731877,   -.017520746267
      ...
                 -.088543630623,    .019679866044,    .042916387274
      ...
                 -.017460408696,   -.014365807969,    .010040411845
      ...
```

```
                  .001484234782,    -.002736031626,   .000640485329
      ];
end

 if strncmpi(Type,'Coiflet',3),
      if Par==1,
             f = [      .038580777748     -.126969125396    -
.077161555496     ...
                        .607491641386     .745687558934
      .226584265197     ];
      end

     if Par==2,
    f = [   0.016387336463203640427491273259,    -
0.041464936786871774009717085107,...
            -0.06737255472372559380457322478,
0.38611006682276285041905816429, ...
            0.8127236354494134953442245286,
0.4170051844232390480477852360, ...
            -0.07648859907828075427761388352,    -
0.05943441864643108730683975853, ...
            0.023680171946847768805923800248,
0.005611434819368834245631844722, ...
            -0.0018232088709110320946081317474,  -
0.0007205494455203469950740392992  ];
    end

       if Par==3,
       f = [      -0.00379351286438080167548512785,
0.00778259642567274575655572678, ...
            0.0234526961420771662427503659,       -
0.0657719112814693671835016013, ...
            -0.0611233900029725412769257018,
0.405176902409118199272476181, ...
            0.793777222626087174791808049,
0.428483476377369981014779057, ...
            -0.071799821619154834013236764,       -
0.0823019271062998184866387774, ...
            0.0345550275732977330127285765,
0.0158805448636694509418667488, ...
            -0.00900797613673062389869059132,     -
0.00257451768813679701027860332,...
            0.001117518770830630223506747086,
0.000466216959820402869469087739, ...
            -0.0000709833025063790056111913832,   -
0.0000345997731972727738834567238    ];
      end
       if Par==4,
       f = [      0.0008923139025370029644343566865134121737383, ...
            -0.0016294924252267858123213544087120913158161, ...
            -0.007346167936268049768871523349607132118705,...
            0.01606894713157502651287763071920848497, ...
            0.02668230466960483260703486399047107194, ...
            -0.08126671024919372334475952281575884847,...
            -0.05607731960356925565970517606545448541,...
            0.4153084270006822731294692964046182739,...
            0.78223893442428258982647576237494028015, ...
            0.43438603311435654244291809279435858752, ...
            -0.06662747236681715660425633300924304564,...
            -0.09622042453595263696014466746929391735,...
            0.03933442260558914633132668418129075104, ...
            0.02508225333794960681821191769918797238, ...
            -0.01521172818769721159723577657069625373, ...
            -0.0056582838001308837068552084504470416875092,...
            0.003751434697146086349179148345354230545201, ...
            0.0012665610789256602060212989509419260626456, ...
            -0.000589020224633216477985278808337632907002, ...
            -0.0002599743371222568031968013258330909296978, ...
            0.0000623388543127871811259366755429554393222, ...
            0.0000312298615991952653049475548892070819932, ...
            -3.259647940030750678302345930717926596838865*10^-6, ...
            -1.78499091449334668126754831733930940802101*10^-6    ];
      end
      if Par==5,
    f = [   -0.0002120818620674939996481902944136410414047, ...
            0.0003585777411617576912682212568176855692809, ...
            0.0021782943778456947603953983474818156341871, ...
            -0.00415931262757863965550073295937883585, ...
            -0.01013158484690027491468212287890872587, ...
            0.02340832211892778307799159801168871895, ...
            0.02816974427053235189367516327102039599, ...
            -0.09192158806008608329572694314371482072, ...
            -0.05204667025355475665111819740782967482, ...
            0.42157126673075435177306894142751870946, ...
            0.77429362286032745160292654060366328807, ...
            0.43798230665916331792682237681050656764, ...
            -0.06203775157498195089253405688027513479, ...
            -0.10556315130733722646960604243367891954, ...
            0.04128753047211783146902068483243193175, ...
            0.03267479946705735095365614221278235227, ...
            -0.01975839160096546513890474068990811885, ...
            -0.00915950733867616299494402122283339758, ...
            0.00676152022062041680244768896723500979, ...
            0.00243157544253828849057946143095553203, ...
            -0.00166162730392987877456474532860529682, ...
            -0.0006375589261258811091711893522661460267423, ...
            0.0003018579416682447498632500676391103210754, ...
            0.0001403563281237324269902880098227066632525, ...
            -0.0000412198619242655021970132036154651202458, ...
            -0.0000212702216725156138191910478670808515531, ...
            3.7007277113394795164497963176580918909*10^-6, ...
            2.061220398578878156703380849619661817616506*10^-6, ...
            -1.623799517204833517470973891374972010523*10^-7, ...
            -9.6040101127678921250276359452964329236689*10^-8     ];
      end

    if Par==6,
    f = [   0.0000507754878363405645519850534163840843165, ...
            -0.0000811700262678483995046181490233613985991, ...
            -0.0006246130439256835305179694992775427, ...
            0.00109162471232590294442838718434499367, ...
            0.00353901987154099797676760943833152463, ...
            -0.0070294063910027282793426873312339407, ...
            -0.0122315777900379124123273915243790564, ...
```

```
        0.0296457728913238384799204963927926239, ...
```

```
          -0.0018015372833304248757700982439011759, ...
          0.00143474185665241227759908616270387381, ...
          0.00057949944823409528234503848853150457, ...
          -0.00036906682873489535803766645994529048, ...
          -0.00016781721215484972023899538482096261, ...
          0.00007971050025993866031131537354202557,...
          0.00004043048241714020221822226691074551, ...
          -0.0000142356369784515012815941734288435861855, ...
          -7.7712435473118614835496973116889804922*10^-6, ...
          2.0020780498554181311877559696846812794*10^-6, ...
          1.1579769069489572558429154565596032359*10^-6, ...
          -2.069320524393852523963248525127217416*10^-7, ...
          -1.2550913190794570507641433432157*10^-7, ...
          1.3935103885216451572958098811601798785366*10^-8, ...
          8.796593384856986494818762397984045253733*10^-9, ...
          -4.57833406779295070172598474467844779766*10^-10,...
          -2.990566231736865818869246704034628*10^-10   ];
  end

if Par==8,
   f = [   2.9543365214148866341203791795460856498*10^-6, ...
          -4.3682648203200749764069604318588208253*10^-6, ...
          -0.00004829631521409294057518294825489293, ...
          0.0000754736783816503960229241373762259,...
          0.00037129499560741243768356918203220247, ...
          -0.000623560447457940255198422634539581, ...
          -0.0017832600085971970719359953695042521, ...
          0.00330082501061611042248776517113013568, ...
          0.00599484919215588549270790383484069611, ...
          -0.012742370632719795227140215081485, ...
          -0.014978462081708433651627145924743, ...
          0.039372037877979845479596171012811, ...
          0.0288286217592880052319436606959, ...
          -0.110169976983470152889118086934849, ...
          -0.043718983365945585858147809196713, ...
          0.431209815555087578822264464482984, ...
          0.760113302017940493745162524428594, ...
          0.443442549841526025692112583885702, ...
          -0.051860743161188676553989081315791, ...
          -0.121211168231496469284998600458647, ...
          0.041185806676256539334644934268549, ...
          0.048252371085682255135905588083649, ...
          -0.026656710542648603662636753258155,...
          -0.018985244695254866708424157255454, ...
          0.014117470077618781611698797024736, ...
          0.007065827011035095989298841611896, ...
          -0.006156659548258420573069188973804, ...
          -0.00254400371024527345236589153040776921, ...
          0.00223564942204810318055673180460398378, ...
          0.00089677606307967974957481016931868077, ...
          -0.00068717164334800449242695373185778567, ...
          -0.00029777893219563999563276515831158818, ...
          0.00018169287648431022077202247294296659, ...
          0.00008754452091843061802876156718818633, ...
          -0.00004147478606916181410858912360731484, ...
          -0.0000218020007670103510682515219710687, ...
          8.03150299544078598193257904121228*10^-6, ...
```

```
        4.4969364435793914273785796464354612539*10^-6, ...
        -1.2754542996407563978284360515425311001*10^-6, ...
        -7.5150215588863255769483315109443069 58*10^-7, ...
        1.5893517221530648274246147211837102 04*10^-7, ...
        9.7724185083677984177279165533463654 3*10^-8, ...
        -1.4540008533753526575612426644 1*10^-8, ...
        -9.2712055915462966940488825258 2*10^-9, ...
        8.6699950823387109524774958829012322 1829*10^-10, ...
        5.7048103339097356584913552593031473 8442*10^-10, ...
        -2.5254234938854568377149207991422 6*10^-11, ...
        -1.7079895947055483349354981618256 2*10^-11  ];
end
 if Par==9,
f = [   -7.1649204312478856537140373597529303 3*10^-7, ...
        1.0293200668945786770405655113884578169*10^-6, ...
        0.0000131588856454253290749791041 38, ...
        -0.0000197872044324624803704064394167 9284, ...
        -0.0001144339527859028293441861809192 7519, ...
        0.0001822848596634226029710141874275 3238, ...
        0.0006264730321397159611910701142046 3634, ...
        -0.0010754582727412379717895037402806 6576, ...
        -0.0024212416736516484133615686341 98, ...
        0.0045970564249205383748272748299 98, ...
        0.0070223404601962370760429115511 57, ...
        -0.0153766496297187638823043729400 94, ...
        -0.0158602238947929078219400824375 59, ...
        0.0431817276082504488961690588529 39, ...
        0.0285726675569492854687006891661 21, ...
        -0.113883508190045080334707288292 53, ...
        -0.041726110205852796928777503053 57, ...
        0.433026751103154194437395990290928, ...
        0.757045523384378918975393007517394, ...
        0.444457893176447943769221898235793, ...
        -0.0493488662936291675797058772885 89, ...
        -0.1243455895392906223482465742549 61, ...
        0.0404737674557289601407844477156 82, ...
        0.0518446156862473157797054460146 27, ...
        -0.0276612394986804629697729848002 19, ...
        -0.0217545535109488448422245565180 26, ...
        0.0158187158159250589965521935532 75, ...
        0.008702757446229182592717517625 46, ...
        -0.0076140424482589170347255778989 24, ...
        -0.003357674526586578587155963179 76, ...
        0.003113227788384303748628662551 515, ...
        0.001269690925135339902432735868 585, ...
        -0.0010957456279526069055973235760 61, ...
        -0.0004627290855053042594485852435 03, ...
        0.000336913869284827093900804675073 65391, ...
        0.000155413521266738851215717324273 54289, ...
        -0.0000913559550874647589464752293 334731, ...
        -0.0000461370819846249313001059672 3498088, ...
        0.0000217763964100290262230136777 6630784, ...
        0.0000118144094515786942327400946 6836094, ...
        -4.4881114751527640989442835285661 3*10^-6, ...
        -2.5723835744866872047576035075266140512*10^-6, ...
        7.8024803293708842770202454094890758 14*10^-7, ...
        4.6795847694542984563228051988566252 3*10^-7, ...
```

```
            -1.1096670180879422876251209697474633337*10^-7, ...
            -6.9165470412180374984933161686610 2242*10^-8, ...
            1.2375256619810124319095047734335 9362*10^-8, ...
            7.9740058868468296718526928424325241*10^-9, ...
            -1.0136275688170465755191268077623618*10^-9, ...
            -6.7234644148859835638965986089839 36*10^-10, ...
            5.4171009642830378310660859592890 93026*10^-11, ...
            3.6861797364451787327421599467079620161*10^-11, ...
            -1.4162735509185840698015637867461918 19*10^-12, ...
            -9.8584372612370775947496469706643 1047*10^-13    ];
    end
end

 if strncmpi(Type,'Daubechies',3),
     if Par==2,
        “f = [ 0.70710678118654752440084436210 4849,
0.70710678118654752440084436210 4849 ];
    end
       if Par==4,
             f = [      0.48296291314453414337487 16,
0.83651630373780790557529 4, ...
                0.22414386804201338102597 3, -
0.12940952255126038117444 94   ];
      end
      if Par==6,
            f = [ 0.33267055295008261599851 2,
0.80689150931109257649449 4, ...
                0.45987750211849157009515 2, -
0.13501102001025458869639 0, ...
               -0.08544127388202666169281 9,
0.03522629188570953660274 07   ];
      end
       if Par==8,
             f = [      0.2303778133088965,     0.7148465705529158, ...
                0.630880767929859,     -0.0279837694168 6011, ...
               -0.1870348117190932,      0.0308413818355608, ...
                0.03288301166688522,   -0.01059740178506904       ];
      end
       if Par==10,
             f = [      0.1601023979741924,     0.6038292697971881, ...
                0.7243085284377715,     0.1384281459013217, ...
               -0.2422948870663802,    -0.0322448695846 3778, ...
                0.07757149384004565,   -0.006241490212798174, ...
               -0.01258075199908194,    0.003335725285473757      ];
      end
       if Par==12,
            f = [ 0.11154074335011,       0.4946238903984554, ...
                0.7511339080210982,     0.315250351709197, ...
               -0.2262646939654429,    -0.1297668675672638,...
                0.0975016055873231,     0.02752286553030565, ...
               -0.03158203931748625,    0.0005538422011615105, ...
                0.004777257510945544,  -0.001077301085308486      ];
      end
       if Par==14,
            “f = [      0.07785205408500813,    0.3965393194819123, ...
                0.7291320908462274,     0.4697822874051917, ...
               -0.1439060039285563,    -0.2240361849938672, ...
```

```
                0.07130921926683042,    0.080612609151082, ...
               -02993693501439,   -0.016574541630667, ...
                0.01255099855609955,    0.0004295779729213739, ...
               -0.001801640704047446,   0.0003537137999745171
      ];”
      end
       if Par==16,
       f = [      0.05441584224310704,    0.3128715909143165,  ...
                0.6756307362973218,     0.5853546836542239, ...
               -0.01582910525637238,   -0.2840155429615815, ...
                0.0004724845739030209,  0.1287474266204823, ...
               -0.01736930100181088,   -0.04408825393079791, ...
                0.01398102791739956,    0.008746094047406, ...
               -0.004870352993451852,  -0.000391740373376942, ...
                0.0006754494064506183, -0.0001174767841247786
      ];
      end
       if Par==18,
             f = [      07794736388813,        0.2438346746126514, ...
                0.6048231236902548,         0.6572880780514298, ...
                0.1331973858249681,        -0.2932737832793372, ...
               -0.0968407832230689,         0.148540749338104, ...
                0.03072568147931585,       -0.06763282906135907, ...
                0.0002509471148277948,      0.0223616621236839, ...
               -0.004723204757752752,      -0.004281503682464633, ...
                0.001847646883056686,       0.0002303857635232296, ...
               -0.0002519631889427889,      0.00003934732031628112
      ];
      end
       if Par==20,
       f = [      0.02667005790054869,        0.188176800077648, ...
                0.527201188931628,          0.6884590394535462, ...
                0.2811723436606982,        -0.2498464243271048, ...
               -0.1959462743773243,         0.127369340335694, ...
                0.0930573646035142,        -0.07139414716638016, ...
               -0.0294575368218849,         0.03321267405931551, ...
                0.003606553566951515,      -0.0107331754833277, ...
                0.001395351747051327,       0.001992405295184184, ...
               -0.0006858566949593225,     -0.0001164668551292262, ...
                0.0000935886703200315,     -0.00001326420289451403
      ];
      end
     if Par==22,
        f = [    0.01869429776144806,        0.1440670211504498, ...
                0.4498997643555165,         0.6856867749154562, ...
                0.4119643689476272,        -0.1622752450269621, ...
               -0.2742308468172826,         0.06604358819685894, ...
                0.1498120124663909,        -0.04647995511648684, ...
               -0.06643878569486228,        0.03133509021904213, ...
                0.02084090436017028,       -0.01536482090617611, ...
               -0.003340858873009247,       0.004928417656058, ...
               -0.0003085928588149355,     -0.00089302325066525, ...
                0.0002491525235524301,      0.00005443907469928305, ...
               -0.00003463498418694142,     0.000004494274277230458
];
    end
         if Par==24,
        f = [    0.01311225795736534,        0.1095662728222715, ...
                0.3773551352176745,         0.657198722584349, ...
                0.5158864784293156,        -0.04476388565908393, ...
               -0.3161784537592869,        -0.02377925725693821, ...
                0.182478605929806 9,        0.00535956967427179, ...
               -0.0964321200976865,         0.0108491302560784, ...
                0.04154627749559747,       -0.01221864906995923, ...
               -0.01284082519846823,        0.00671149900888981, ...
                0.002248607241020708,      -0.002179503618657147, ...
                0.000006545128213682533,    0.0003886530628261407, ...
               -0.0000885041092094801,     -0.00002424154575734139, ...
                0.00001277695221955214,    -0.000001529071758089919
];
    end
         if Par==26,
        f = [    0.00920213353936357,        0.082861243876398, ...
                0.3119963221728867,         0.6110558511805082, ...
                0.5888895704451372,         0.0869857261666496, ...
               -0.314972907739053,         -0.124576730762086, ...
                0.1794760794355785,         0.07294893365742099, ...
               -0.1058076181950538,        -0.02648840647689916, ...
                0.05613947710301562,        0.002379972253836755, ...
               -0.02383142071161908,        0.003923941449079961, ...
                0.007255589402002825,      -0.002761911234808676, ...
               -0.001315673911943637,       0.000932326130928484, ...
                0.00004925152513188404,    -0.0001651289885636495, ...
                0.00003067853758174376,     0.00001044193057207212, ...
               -0.000004700416479607929 ,   0.0000005220035098765021
];
    end
         if Par==28,
       f = [    0.006461153459818989,       0.0623647588469322, ...
                0.2548502677833766,         0.5543056179241174, ...
                0.6311878490950694,         0.2186706877760189, ...
                -0.2716885522429336,        -0.2180335299738394, ...
                0.138395213856541,          0.1399890165735457, ...
                -0.0867484115685856,        -0.07154895550625034, ...
                0.05523712625188016,        0.02698140830446938, ...
                -0.0301853515397028,        -0.005615049530747707, ...
                0.01278949326524909,        -0.000746218989436958, ...
                -0.003849638867994312,      0.001061691085418039, ...
                0.0007080211541344865,      -0.0003868319473184179, ...
                -0.00004177724577935138,    0.00006875504251988474, ...
                -0.00001033720918460207,    -0.000004389704901652653,
...
                0.000001724994675254821,    -0.0000001787139968209 58
];
    end
         if Par==30,
       “f = [    0.004538537356680069,       0.0467433948433292, ...
                0.2060238637760462,         0.4926317712332494, ...
                0.6458131398235114,         0.339002535383428, ...
                -0.19320413905893,          -0.2888825960016258,  ...
                0.06528295291444258,        0.1901467139017971,  ...
                -0.03966617641454303,       -0.1111209358626346, ...
                0.03387714389352461,        0.05478055052762776, ...
                -0.0257670072911817,        -0.02081005014572826,  ...
                0.01508391800773139,         0.005101000354434229, ...
                -0.006487734552531616,       -0.0002417564910950625,
...
                0.001943323977748212,        -0.0003734823537271217,
...
                -0.0003595652439869339,      0.0001558964896924794,  ...
                0.00002579269911910246,      -0.00002813329623232866,
...
                0.000003362987176654478,     0.000001811270405641324,
...
                -0.0000006316882317817563,  0.00000006133359905269254
];”
    end

          if Par==32,
        f = [    0.003189220905181802,        0.0349077141074775,  ...
                0.1650642824989111,          0.4303127204089899,  ...
                0.6373563289234388,          0.4402902557886062, ...
                -0.0897510867287953,         -0.3270633068118058,  ...
                -0.02791820715372535,        0.2111906930487478,  ...
                0.02734026408611786,         -0.1323883043443139, ...
                -0.006239722263724492,       0.07592423555847598,  ...
                -0.007588974252298305,       -0.03688839741760147,  ...
                0.01029765955546528,         0.01399376876290007, ...
                -0.006990014507518413,       -0.003644279596729619,  ...
                0.003128023357662664,        0.0004078969789133 64,  ...
                -0.0009410217421877 43,      0.0001142415191130 91,  ...
                0.0001747872440135933,       -0.00006103596571228747,
...
                -0.00001394566888488284,     0.00001133660857799308, ...
                -0.000001043571333041443,    -0.0000007363656730469882,
...
                0.0000002308784069376313,    -0.00000002109339613774 45
];”
    end
          if Par==34,
        f = [    0.002241806968367765,        0.02598539333038641,  ...
                0.1312149014643511,          0.3703507191428474,  ...
                0.6109966080619875,          0.5183157592365552, ...
                0.02731497388861195,         -0.3283207398752789,  ...
                -0.1265997478695799,         0.1973105883690036,  ...
                0.1011354893285621,          -0.1268156885448092, ...
                -0.05709141812622551,        0.081105985705437,  ...
                0.02231233608959475,         -0.04692243752178137,  ...
                -0.003270955473782776,       0.02273367623263168, ...
                -0.003042989911563062,       -0.008602921379753 92,  ...
                0.002967996640915282,        0.002301205207197428,  ...
                -0.001436845280352317,       -0.0003281325149411173,
...
                0.0004394654201169656,       -0.00002561010931458864,
...
                -0.00008204803088019 88,     0.00002318681330990614, ...
                0.000006990600842366534,     -0.000004505942411707292,
...
                0.0000003016549532645506,    0.0000002957700881589635,
...
```

```
                -0.0000000842394830828037,  0.0000000072674928439190 08
];
    end
           if Par==36,
        f = [    0.001576310332632241,        0.01928853309434481,  ...
                0.1035884729715391,          0.3146789620466176,  ...
                0.571826841995251,           0.5718016803655575, ...
                0.147223099399332,           -0.2936540837163994,  ...
                -0.2164809618743174,         0.1495339814252923,  ...
                0.1670813196471977,          -0.0923318969776604, ...
                -0.1067522571200224,         0.0648872212223416,  ...
                0.05705125157931265,         -0.04452614611490133,  ...
                -0.02373321210978654,        0.02667070832113655, ...
                0.006262168357742094,        -0.01305148206344844,  ...
                0.0001186301071328846,       0.004943344018360076,  ...
                -0.001118732786346494,       -0.001340596411265555,
...
                0.0006284657384942994,       0.0002135815764103265,
...
                -0.000198648570821057,       -0.0000001535916342659 62,
...
                0.00003741238184339052,      -0.00000852060341054129,
...
                -0.000003332634770075 13,     0.0000017687131374864 3,
...
                -0.0000000769163364021 7469,  -0.0000001176098869880 653,
...
                0.0000000306883613712 2469,   -
0.000000002507934683892356     ];
    end
         if Par==38,
       f = [    0.001108669779715294,         0.01428109865333334,  ...
                0.081278114333354,            0.2643884347822977,  ...
                0.5244363819574067,           0.6017045501513535, ...
                0.2608949440110274,           -0.2280914100170829,  ...
                -0.285838641929714,           0.07465227262054114,  ...
                0.2123497512548378,           -0.03351853842979753, ...
                -0.1427856935054576,          0.02758435493215239,  ...
                0.0869067594236619,           -0.02650123589611068,
...
                -0.04567422669495623,         0.02162376812192859, ...
                0.01937555029280247,          -0.01398838901012597,
...
                -0.005866922391341 82,        0.007040747519198927,
...
                0.0007689543646753964,        -0.002687551858597481,
...
                0.0003418086639330359,        0.0007358025360798398,
...
                -0.0002606761416764582,       -0.0001246007941078683,
...
                0.0000871127066319985,        0.000005105950548947162,
...
                -0.00001664017665533139,      0.000003010964385934741,
...
                0.000001531931507655374,      -
0.0000006862755810090276, ...
```

```
                0.00000001447088339408005,
0.00000004636937873589416,  ...
                -0.000000011164020912898,
0.000000000866684902796269    ];
    end
         if Par==40,
       f = [    0.0007799530020084384,        0.0105493864101072,  ...
                0.06342373157542249,          0.2199419467839922,  ...
                0.4726958375631425,           0.6104928215175741, ...
                0.3615021297395791,           -0.139211825416023,  ...
                -0.3267863905078842,          -0.01672694530514085,
...
                0.2282909876975237,           0.03985032729018178, ...
                -0.1554585361790331,          -0.02471674917392653,
...
                0.1022916746204368,           0.005632268726873665,
...
                -0.06172283526148656,         0.005874682288534986, ...
                0.03229427583633914,          -0.00878931595226129,
...
                -0.01381051445886118,         0.006721621652169426,
...
                0.004420538864131319,         -0.003581491222634283,
...
                -0.00083156152944895,         0.001392558453825609,
...
                -0.00005349753868856166,      -0.0003851044297986765,
...
                0.0001015328014373285,        0.00006774275277093538,
...
                -0.00003710583043522718,      -0.000004376140493506968,
...
                0.000007241242222701708,      -0.000001011993125412585,
...
                -0.0000006847073928591012,    0.0000002633921999175421,
...
                0.0000000002014328820034285,  -
0.0000000181484172957345, ...
                0.000000004056123630675098,   -
0.0000000002998833944499773     ];
    end
         if Par==42,
        f = [    0.0005488240399453808,        0.007776660464348811,
...
                0.04924790475876491,          0.1813601028599902,  ...
                0.419688998145241,            0.6015074510688103, ...
                0.4445910837993439,           -0.03572381948901234,
...
                -0.33566575122537,            -0.1123978514710653,  ...
                0.2115648260162405,           0.1152333439473735, ...
                -0.1399410472763452,          -0.08177625782428998,
...
                0.09660066710664022,          0.04572352417673011,  ...
                -0.06497770623152748,         -0.01865389796875268, ...
                0.03972696757220106,          0.003357765554657301,
...
```

```
                -0.02089211624987374,        0.002403482102825579,
...
                0.008988852342563074,        -0.002891344156898007,
...
                -0.002958382842307337,       0.001716612683276365,
...
                0.0006394203289590759,       -0.0006906733219030776,
...
                -0.00003196410553726866,     0.0001936652571660039,
...
                -0.0000363553295677002,      -0.00003499676704742804,
...
                0.00001535487521020741,      2.79033850314008*10^-6,
...
                -3.090027001911602*10^-6,    3.16610662424439*10^-7,
...
                2.99214595113828*10^-7,      -1.000404119487493*10^-7,
...
                -2.254019869522092*10^-9,    7.058055911572644*10^-9,
...
                -1.471958939283684*10^-9,    1.038808947669207*10^-10
];
    end
         if Par==44,
        f = [    0.0003862673246197253,          0.005721914066631049,
...
                0.03807032187691932,           0.1483689789282081,
...
                0.3677320057234413,            0.5784372354311235, ...
                0.5079033273631367,            0.07372115020105462,
...
                -0.3127333476121842,           -0.2005720141344328,
...
                0.1640948426591233,            0.179974931810035, ...
                -0.0971123372197599,           -0.1317696149504392,
...
                0.06807740848784511,           0.08455839833964807,
...
                -0.05136497255398131,          -0.04653131832736136,
...
                0.03697137276735332,           0.02058693268949487,
...
                -0.02348031395539096,          -0.006213835918293782,
...
                0.01256489065516637,           0.0003001305020824184,
...
                -0.005455761185358356,         0.001044278408986017,
...
                0.001827032986409597,          -0.000770702101944467,
...
                -0.0004237923063271874,        0.0003286138886837352,
...
                0.000043459369254213 9,         -
0.00009405347080647135,  ...
                0.00001137454223403893,        0.00001737397675279249,
...
```

```
                -6.166816318076451*10^-6,       -1.565197277819435*10^-
6,  ...
                1.295199441207159*10^-6,        -8.78003044824892*10^-
8,  ...
                -1.283352833826032*10^-7,       3.761280659022215*10^-
8,  ...
                1.680187679020641*10^-9,        -2.729659356918619*10^-
9,  ...
                5.33601149622179*10^-10,        -3.60216327759258*10^-
11   ];
    end
         if Par==46,
        f = [    0.0002719278182602901,          0.004203109552950134,
...
                0.02931247643736339,            0.1205254471036576,
...
                0.3184759568589838,             0.5449708209347766, ...
                0.5510501337055957,             0.1813841378320262,
...
                -0.2614398761995617,            -0.2714429864972958,
...
                0.0921245749243952,             0.2235864349031235, ...
                -0.03304774793732929,           -0.164030308293076,
...
                0.02028436820991752,            0.1123069840244809,
...
                -0.0211292480280753,            -0.07021415427385447,
...
                0.02176834655240395,            0.03849895908078205,
...
                -0.01852549112315692,           -0.01753870400805271,
...
                0.01275326613768589,            0.006032371159860696,
...
                -0.00707603267773538,           -0.001134947880346942,
...
                0.003123184807392083,           -0.000246537026777104,
...
                -0.001061334361043996,          0.000319454992361999,
...
                0.0002567865998070605,          -0.0001500373078747796,
...
                -0.00003379188332733358,        0.00004426515179248939,
...
                -2.635561787093299*10^-6,       -8.348692795439998*10^-
6,  ...
                2.397822036092728*10^-6,        8.14834303822415З*10^-
7,  ...
                -5.339546450998099*10^-7,       1.853340229309923*10^-
8,  ...
                5.418084825798256*10^-8,        -1.400079829615052*10^-
8,  ...
                -9.473736128438874*10^-10,      1.050551729533758*10^-
9, ...
                -1.93260193304542*10^-10,       1.250331739337031*10^-
11    ];
    end
```

```
         if Par==48,
        f = [    0.0001914240079776934,          0.003081894336144903,   
...
                0.02248099723913652,            0.09725657409395711,   
...
                0.272893661713225,              0.5043448957614517, ...
                0.5749146829767083,             0.2809851510053765, ...
                -0.1872418464658568,            -0.3179111538203686,   
...
                0.004781510762825361,           0.2392258659829295, ...
                0.042531243536347,              -0.1711600617797226,   
...
                -0.03877318682438014,           0.1210092088290207,   
...
                0.02098022912439134,            -0.08215538086453539,   
...
                -0.004578395730450242,          0.05129798128535279,   
...
                -0.004944235600686442,          -0.02821125709939177,   
...
                0.007661004281903052,           0.01304905186620713,   
...
                -0.006290964935213451,          -0.004746267936383896,   
...
                0.00373576397589871,            0.001153694353296646,   
...
                -0.001696334910033699,          -
0.00004416435334971148,  ...
                0.0005860851561798487,          -0.000118113728929818,   
...
                -0.0001459980983446589,         0.00006558881863639525,   
...
                0.00002183096348720674,         -
0.00002022741617379432,  ...
                1.337052417608915*10^-8,        3.900821594914755*10^-
6,  ...
                -8.979550384702172*10^-7,       -4.032228084773544*10^-
7,  ...
                2.166180932866001*10^-7,        -5.054643465620961*10^-
10,  ...
                -2.255577015054618*10^-8,       5.157391468496204*10^-
9, ...
                4.748066278754132*10^-10,       -4.024365393060184*10^-
10,  ...
                6.991284124010881*10^-11,       -4.342457865150871*10^-
12    ];
    end
         if Par==50,
        f = [    0.0001349140099190347,          0.002258811421928896,   
...
                0.01720076826024011,            0.07809905498823061,   
...
                0.2318787664012299,             0.4600423810287615, ...
                0.5820659400767751,             0.3680932821030804,   
...
                -0.0973914656330337,            -0.3369053502250008,   
...
```

```
                -0.08780821918435377,           0.2245876085890194, ...
                0.118127995249711,              -0.1507901382415503,
...
                -0.09866413500669811,           0.1066759749914026,
...
                0.06677659879664328,            -0.07716829308409851,
...
                -0.03721412413495515,           0.05365935822485196,
...
                0.01555326324900022,            -0.03407222558276132,
...
                -0.003082422156835764,          0.01893883135517686,
...
                -0.001991310900912701,          -0.00886823166517666,
...
                0.002729324227909382,           0.003325513663082694,
...
                -0.001844076742849583,          -0.0009007311640167205,
...
                0.0008780108981336599,          0.0001154141401669138,
...
                -0.0003101446827543741,         0.00003546902687895924,
...
                0.00007911359754985811,         -
0.00002735413947239519,  ...
                -0.00001278271963847119,        8.998380584459695*10^-
6,  ...
                5.236962340729562*10^-7,        -1.780721007596221*10^-
6,  ...
                3.214832263002398*10^-7,        1.924439055598693*10^-
7,  ...
                -8.66437988387495*10^-8,        -2.613650299345624*10^-
9, ...
                9.287154489811888*10^-9,        -1.882036245600092*10^-
9,  ...
                -2.230357347052905*10^-10,      1.537216399287396*10^-
10,  ...
                -2.529795306330366*10^-11,      1.510990558374748*10^-
12     ] ;
    end
end

 if strncmpi(Type,'Symmlet',3),
     if Par==1,
         f = [ 0.70710678118654752440084436210484 9,
0.70710678118654752440084436210484 9 ];
    end
       if Par==2,
             f = [      0.48296291314453414337487 16,
0.83651630373780790557529 4, ...
                0.22414386804201338102597 3, -
0.12940952255126038117444 94   ];
      end
       if Par==3,
            f = [ 0.33267055295008261599851 2,
0.80689150931109257649449 4, ...
```

```
                0.459877502118491570095152, -
0.135011020010254588696390, ...
               -0.085441273882026661692819,
0.035226291885709536602740 7   ];
    end
       if Par==4,
             f = [   -0.07576571478950221,       -0.029635527646002493,
...
                0.497618667632775,          0.8037387518051321,   ...
                0.29785779560530606,        -0.09921954357663353,   ...
                -0.012603967262031304,      0.032223100604051466  ];
      end
       if Par==5,
             f = [   0.019538882735249826 8,      -
0.021101834024689041 0,   ...
                -0.17532808990805622 4,      0.016602105764510848,   ...
                0.63397896345679206,        0.72340769040404079,   ...
                0.199397533976855597,       -0.039134249302313844,
...
                0.029519490925706261 3,      0.027333068344998768 8
      ];
      end
       if Par==6,
             f = [   0.01540410932704474,        0.003490712084221531,
...
                -0.1179901111485212,        -0.048311742585697 89,   ...
                0.4910559419279768,         0.787641141028653 6,  ...
                0.3379294217281644,         -0.07263752278637825,   ...
                -0.02106029251237119,       0.04472490177078142,   ...
                0.001767711864253766,       -0.007800708325032496 ];
      end
       if Par==7,
             f = [   0.01026817670846495,        0.004010244871523197,
...
                -0.1078082377032895,        -0.1400472404429405,   ...
                0.2886296317506303,         0.7677643170048699,  ...
                0.5361019170905749,         0.01744125508685128,   ...
                -0.04955283493703385,       0.06789269350122353,   ...
                0.03051551316588014,        -0.01263630340323927,  ...
                -0.001047384888679668,      0.002681814568260057
            ];
      end
       if Par==8,
            “f = [       -0.003382415951003908,     -
0.000542132331797018,   ...
                 0.03169508781151886,       0.007607487324948 97,   ...
                 -0.1432942383512576,       -0.06127335906765891,   ...
                 0.4813596512592537,        0.7771857516996492,   ...
                 0.3644418948360139,        -0.05194583810802026,   ...
                 -0.02721902991713553,      0.04913717967372511,   ...
                 0.003808752013880463,      -0.01495225833706814,   ...
                 -0.0003029205147226741,    0.001889950332768561
      ];
      end
       if Par==9,
             f = [       0.001069490032908175,      -
0.0004731544986808867,   ...
```

```
                 -0.01026406402762793,      0.00885926749341011 7,   ...
                 0.06207778930285313,       -0.01823377077946773,   ...
                 -0.1915508312971598,       0.03527248803579076,   ...
                 0.6173384491414731,        0.7178970827644066,   ...
                 0.2387609146068536,        -0.05456895843120489,   ...
                 0.0005834627459892242,     0.03022487885821281,   ...
                 -0.01152821020772933,      -0.01327196778183437,   ...
                 0.0006197808889867399,     0.001400915525915921  ];
      end
       if Par==10,
             f = [       0.0007701598091036597,
0.00009563267068491565,   ...
                 -0.008641299277002591,     -0.001465382581138532,
...
                 0.04592723923095083,       0.0116098939028464,   ...
                 -0.1594942788849671,       -0.07088053578057 98,   ...
                 0.4716906669415791,        0.7695100370206782,   ...
                 0.3838267610640166,        -0.03553674047551473,   ...
                 -0.03199005688220715,      0.04999497207760673,   ...
                 0.005764912033412411,      -0.02035493981234203,   ...
                 -0.0008043589319389408,    0.004593173585320195,   ...
                 0.00005703608359777954,    -0.0004593294210107238
            ];
      end
     if Par==11,
         f = [    0.0004892636102790465,     0.00011053509770077,   ...
                 -0.006389603666537886,     -0.002003471900538333,
...
                 0.04300019068196203,       0.03526675956730489,   ...
                 -0.1446023437042145,       -0.2046547945050104,   ...
                 0.2376899090326669,        0.7303435490812422,   ...
                 0.5720229780188006,        0.09719839447055164,   ...
                 -0.02283265101793916,      0.06997679961196318,   ...
                 0.03703741598066749,       -0.0240808415947161,   ...
                 -0.009857934828835874,     0.006512495674629366,   ...
                 0.0005883527354548924,     -0.001734366267274675,
...
                 -0.00003879565575380471,   0.0001717219506928879
];
      end
     if Par==12,
         f = [    -0.0001790665869786187,     -0.00001815807887734 71,
...
                 0.002350297614165271,       0.0003076477963025531,
...
                 -0.0145898364492100 9,       -0.002604391031185636,
...
                 0.05780417944546282,        0.01530174062149447,   ...
                 -0.1703706972388913,        -0.07833262231005749,
...
                 0.4627410312313846,         0.7634790977904264,   ...
                 0.3988859723844853,         -0.0221623061807925,   ...
                 -0.03584883074255768,       0.04917931829833128,   ...
                 0.007553780610861577,       -0.02422072267559388,
...
                 -0.001408909244210085,      0.007414965517868044,
...
```

```
                 0.0001802140900854918,      -0.001349755755614803,
...
                 -0.00001135392805049379,    0.0001119671942470856
];
      end
     if Par==13,
         f = [    0.00007042986709788876,     0.00003690537416474083,
...
                 -0.0007213643852104347,     0.0004132611973679777,
...
                 0.00567485376954048,        -0.00149244724795732,
...
                 -0.02074968632748119,       0.01761829684571489,   ...
                 0.09292603099190611,        0.008819757923922775,  ...
                 -0.1404900930989444,        0.1102302225796636,   ...
                 0.6445643835707201,         0.6957391508420829,   ...
                 0.1977048192269691,         -0.1243624606980946,   ...
                 -0.05975062792828035,       0.01386249731469475,   ...
                 -0.01721164274779766,       -0.02021676815629033,
...
                 0.005296359721916584,       0.007526225395916087,
...
                 -0.0001709428497111897,     -0.001136063437095249,
...
                 -0.0000357386241733562,     0.00006820325245288671
];
      end
     if Par==14,
         f = [    0.00004461898110644152,     0.00001932902684197359,
...
                 -0.0006057602055992672,     -0.00007321430367811753,
...
                 0.004532677588409982,       0.001013142476182283,  ...
                 -0.01943931472230284,       -0.002365051066227485,
...
                 0.06982761641982026,        0.02589859164319225,  ...
                 -0.1599974161449017,        -0.05811184934484923,  ...
                 0.4753357348650867,         0.7599762436030552,  ...
                 0.3932015487235067,         -0.03531809075139569,  ...
                 -0.05763449302747868,       0.03743308903888159,  ...
                 0.004280522331795536,       -0.02919621738508546,  ...
                 -0.002753775776578359,      0.01003769335863697,   ...
                 0.000366476770515625,       -0.002579441672422145,
...
                 -0.00006286548683867455,    0.0003984356519092697,
...
                 0.00001121086996816579,     -0.00002587908845615303
];
      end
     if Par==15,
         f = [    0.000028660707399511618,    0.00002171788180828788,
...
                 -0.00040216855725720875,    -0.00010815440716833184,
...
                 0.0034810287335767247,      0.0015261379816940614,
...
```

```
                 -0.01717125379613366,       -0.008744790172337504,
...
                 0.06796982852319061,        0.06839330814883558,    ...
                 -0.13405630898495416,       -0.19662638205515043,
...
                 0.24396267904975416,        0.7218430182315695,   ...
                 0.5786404260768322,         0.11153371778633057,    ...
                 -0.04108264654576916,       0.040735492211698726,
...
                 0.021937649685129356,       -0.03887671343231794,
...
                 -0.019405010502865788,      0.010079976851620261,
...
                 0.0034234504500168204,      -0.003590165561744339,
...
                 -0.0002673165193090753,     0.0010705671659500386,
...
                 0.00005512253152945822,     -0.00016066186440056O3,
...
                 -7.359664544811822*10^-6,   9.712420111966456*10^-6
];
      end
     if Par==16,
        f = [    -0.000010797985444060825,   -5.3964960594326874*10^-6,
...
                 0.0001654568107575125,      0.000036566003188144 29,
...
                 -0.001338720814110487,      -0.00022211728990382076,
...
                 0.006937761277326152,       0.001359846701344l263,
...
                 -0.024952761177797576,      -0.00351028784332523,
...
                 0.07803784806236971,        0.030721150497025267,
...
                 -0.15959221677565338,       -0.05404070849544753,
...
                 0.4753426930640303,         0.7565249925494576,   ...
                 0.3971230417468278,         -0.034574136859023705,
...
                 -0.06698302069147749,       0.03233309920319836,    ...
                 0.004869285298401547,       -0.031051198943356957,
...
                 -0.003126520186702648,      0.012666730821216037,
...
                 0.0007182134482533401,      -0.0038809118096674354,
...
                 -0.00010844597239036113,    0.0008523546022172409,
...
                 0.00002807864421317797 5,    -0.00010943145921454754,
...
                 -3.1135618941800743*10^-6,  6.230004737262791*10^-6
];
      end
     if Par==17,
         f = [    4.2973506629986956*10^-6,   2.7801688647505625*10^-6,
...
```

```
                 -0.00006293696990392763,    -0.000013506396803548528,
...
                 0.00047599681001429773,     -0.00013864025450273018,
...
                 -0.002741673289197601,      0.0008567721508596517,
...
                 0.010482379074328828,       -0.004819175075474626,
...
                 -0.03329133163029172,       0.017904007873400175,
...
                 0.10475473990614559,        0.017271442186243685,
...
                 -0.1185666208590123,        0.1423985355331832,    ...
                 0.6507166482523769,         0.6814888952815437,    ...
                 0.18053935219589717,        -0.15507633780169236,
...
                 -0.0860711490606599,        0.016158692741874165,
...
                 -0.0072616521550352585,     -0.01803889816207702,
...
                 0.009952983267225308,       0.012396993805630462,
...
                 -0.0019054056198378236,     -0.003932327920107898,
...
                 0.00005839844001157818,     0.0007198270212964605,
...
                 0.000025208218893204052,    -0.00007607121225156968,
...
                 -2.452745060439947*10^-6,   3.791246551426962*10^-6
];
      end
     if Par==18,
         f = [    2.6140187028148005*10^-6,   1.3597256411120972*10^-6,
...
                 -0.000045260099151022276,   -0.000014073709856766986,
...
                 0.0003962647703053168,      0.00007059203923948904,
...
                 -0.002314325309127317,      -0.00041350784379068383,
...
                 0.009503146264507371,       0.001648937806891975,
...
                 -0.03033018760379904,       -0.00510180660253373,
...
                 0.08421406918773965,        0.034024019305781264,
...
                 -0.15998146275864744,       -0.05223243824458295,
...
                 0.47375599169059157,        0.7536405256423095,    ...
                 0.4016913034776937,         -0.0323059809889792,    ...
                 -0.07374895242999176,       0.028536771134301076,
...
                 0.006292693377765397,       -0.03170873328122627,
...
                 -0.0032688666608867664,     0.01500986633652236,    ...
                 0.0010909833425781458,      -0.005238814002156449,
...
```

```
                 -0.0001897773119942463,     0.001427781305076276 5,
...
                 0.00004765078627565525 4,    -0.0002657505546476236,
...
                 -9.890224232299607*10^-6,   0.0000295454573529232 2,
...
                 7.866685781899692*10^-7,    -1.5123391551795506*10^-6
];
      end
     if Par==19,
         f = [    1.6822101374310987*10^-6,   1.5736233900352247*10^-6,
...
                 -0.000029084130082422266,   -0.000016290552278010628,
...
                 0.0002753742621134243 3,     0.0001045592173367842 4,
...
                 -0.0017655593234062173,     -0.0006524693932405551,
...
                 0.008240792309156801,       0.003903523253914178,
...
                 -0.028979337236055096,      -0.01858176253516738,
...
                 0.08265001836815583,        0.09052455442917737,   ...
                 -0.12553103109717628,       -0.193302466610985 7,   ...
                 0.24104659627671224,        0.7125038682183499,   ...
                 0.5857258835982894,         0.12559052557160855,   ...
                 -0.051417687498444706,      0.019505347664693754,
...
                 0.012438665386547568,       -0.04432391883451796,
...
                 -0.022155886952494982,      0.015464312015532047,
...
                 0.0076454965207025565,      -0.005198489304476271,
...
                 -0.0011616402526739716,     0.0021058625600138578,
...
                 0.0001566090506025432 8,     -0.0006245052918345595,
...
                 -0.000041063447425587 38,    0.0001139524500214609,
...
                 7.487464336562946*10^-6,    -0.000011966487988902881,
...
                 -5.34322686486931*10^-7,    5.711932378486931*10^-7
];
      end
     if Par==20,
        f = [    -6.299929076393514*10^-7,   -1.788381654723433*10^-7,
...
                 0.000013267792274307285,    5.715455163913879*10^-6,
...
                 -0.00012638480147829343,    -0.000060607704301728 46,
...
                 0.0007316472152793887,      0.00026095442734033774,
...
                 -0.003170803376848161,      -0.00043732670748901353,
...
```

```
                 0.01209058957678 6489,        0.001921167634207 2246,
...
                 -0.03663262584755033,         -0.013703513480057003,
...
                 0.07424185717192522,          0.02171405425391 0882,
...
                 -0.1609037041111548,          -0.03440166000144136,  ...
                 0.4894006264128903,           0.7507502982136451,   ...
                 0.3886194419202543,           -0.04521583665884769,
...
                 -0.07854049760146452,         0.03848275740152 7116,
...
                 0.02170214282821 3884,        -0.023834246491111964,
...
                 -0.0005786000109801912,       0.01717848841075394,   ...
                 0.0006324619136888966,        -0.007342682291732647,
...
                 -0.0005183408270565881,       0.002204695518355125,
...
                 0.00017752923953843423,       -0.00048232152518013383,
...
                 -0.000034343957912744 26,     0.000074065490501176 18,
...
                 3.2530365938639585*10^-6,     -7.413187876923192*10^-6,
...
                 -1.0539262586695314*10^-7,    3.7126642825501105*10^-7
];
      end
     if Par==21,
         f = [    2.4666687655533695*10^-7,     7.914248389737127*10^-8,
...
                 -5.098478772975259*10^-6,     -1.8332525199414118*10^-6,
...
                 0.000046948429858743 22,      6.088937039539296*10^-6,
...
                 -0.00028648787462784116,      0.00006202285932098 85,
...
                 0.0014213616519049683,        -0.000432897763752902,
...
                 -0.005353988694277162,        0.0017655806658982696,
...
                 0.015402169297438343,         -0.008988082355110115,
...
                 -0.04716066845359608,         0.01067117877581 5944,
...
                 0.10149319625091788,          0.00738855413219 6435,
...
                 -0.12355469595545877,         0.1441113106030 8363,   ...
                 0.6461733765248382,           0.6815923286559533,   ...
                 0.19203957652810966,          -0.1513192612796282,   ...
                 -0.08711940361369748,         0.02790507569203459,   ...
                 0.00625184607285568,          -0.015083074728218489,
...
                 0.011187152328141188,         0.01450803878862172,   ...
                 -0.004094954089873776,        -0.006749289429682182,
...
```

```
                 0.0006955275549391147,      0.002031175814309811,
...
                 -0.00002051693327729468,    -0.00041348833873143716,
...
                 -0.000010572800078769339,   0.00005820430382793364 4,
...
                 1.8409309631189882*10^-6,   -5.161165596805668*10^-6,
...
                 -7.41575536253381*10^-8,    2.3113011977605634*10^-7
];
      end
     if Par==22,
         f = [    1.5469892096783629*10^-7,   5.5743916934275646*10^-8,
...
                 -3.4968422679833534*10^-6,  -1.4370363121605532*10^-6,
...
                 0.000037380396807580654,    0.000017580940554227 16,
...
                 -0.00024483364534030704,    -0.00010585417623386842,
...
                 0.0011466583097184356,      0.0003226285920630637,
...
                 -0.004398948577001492,      -0.0005350221246800008,
...
                 0.015051009425518232,       0.003114197433178795,
...
                 -0.03983484859283483,       -0.01431630999875656 3,
...
                 0.07823387878992308,        0.023785426002814004,
...
                 -0.1593358890186262,        -0.029991258060954408,
...
                 0.4905406567562972,         0.7486689893282149,    ...
                 0.38946084846091966,        -0.04646770047156702,
...
                 -0.08505017544453836,       0.03554141495048075,    ...
                 0.023467779250349607,       -0.02426659585361798,
...
                 -0.0019440502814989976,     0.01780655171188534,    ...
                 0.0003877388938834633,      -0.008837354866518892,
...
                 -0.0006023488806178174,     0.0030309320519889987,
...
                 0.00025194457519336455,     -0.0007945748785723063,
...
                 -0.00006608462189094136,    0.0001549398752161714 4,
...
                 0.000010202851313391099,    -0.000021749524061348802,
...
                 -8.277216867302219*10^-7,   2.011485781107503*10^-6,
...
                 3.24106151598103*10^-8,     -8.994488134257862*10^-8
];
      end
     if Par==23,
         f = [    -6.088466993156109*10^-8,    -1.661520430459547 2*10^-
8,   ...
```

```
                 1.4054134729420521*10^-6,   5.065545930484757*10^-7,
...
                 -0.000014639159377583806,   -3.935849604669885*10^-6,
...
                 0.00009610455472740267,     -3.235179654211356*10^-7,
...
                 -0.0004912392092331126,     0.00011586281878255 32,
...
                 0.0020794420582160053,      -0.0005397239075479521,
...
                 -0.006723311075642093,      0.0023801024676674445,
...
                 0.018427107312302106,       -0.009708672972735352,
...
                 -0.052966594269723166,      0.006024328570134602,
...
                 0.09868192677853799,        0.0026702039747034177,
...
                 -0.1299766448268073,        0.13550672099058436,    ...
                 0.6387382588927035,         0.6863848550710899,    ...
                 0.20649267004039606,        -0.14369715652684245,
...
                 -0.08651297991591521,       0.03276374191323 8865,
...
                 0.012533684345771838,       -0.014070855072454728,
...
                 0.01062794083550241,        0.015024638946099386,
...
                 -0.004866076235991217,      -0.007967029103383086,
...
                 0.0010722016128121928,      0.0028093755854430584,
...
                 -0.00008313779295501525,    -0.0006970208534264938,
...
                 -0.00001326126189449906,    0.00012603730932265127,
...
                 4.446291783441227*10^-6,    -0.000016139269667555554,
...
                 -4.77553521844591*10^-7,    1.3365100765666046*10^-6,
...
                 1.523804807305629*10^-8,    -5.5838351515000746*10^-8
];
      end
     if Par==24,
         f = [    -5.2761865436863985*10^-8,  -8.690427954503197*10^-8,
...
                 1.0423208461508862*10^-6,   1.3969259517138812*10^-6,
...
                 -0.000011118982675005505,   -0.000012657478427953705,
...
                 0.0000773048286026619,      0.00007342863308434417,
...
                 -0.00039834902286264043,    -0.0003016488762115482,
...
                 0.00163359910950814 9,       0.0009177057961552579,
...
```

```
                 -0.005702626293286814,      -0.002525604130488947,
...
                 0.017026080554935406,       0.007861303957162372,
...
                 -0.040982600612933254,      -0.021283796275817276,
...
                 0.08372034358295327,        0.05437314304210483,   ...
                 -0.12818178218050752,       -0.035646434246813574,
...
                 0.46244307077589863,        0.7501914811592109,   ...
                 0.4182481983661332,         -0.04590755369598393,
...
                 -0.1204529195912711,        0.010585497932626633,
...
                 0.023513400494410315,       -0.022539152872931466,
...
                 -0.005021587977305239 4,    0.01833813846077915,   ...
                 0.0021823244607757,         -0.009954852329009839,
...
                 -0.0015175250406422927,     0.003835747174495 1996,
...
                 0.0007028294509160195,      -0.0011116911888874527,
...
                 -0.00021118156525038934,    0.0002507142923455971,
...
                 0.0000442170717380261,      -0.000043237491892894634,
...
                 -6.45525681719476*10^-6,    5.343044886829498*10^-6,
...
                 5.953891366605391*10^-7,    -4.1947862803617735*10^-7,
...
                 -2.5954357139211734*10^-8,  1.5757584826821534*10^-8
];
      end
     if Par==25,
         f = [   1.3559364540611227*10^-8,    -9.795737094326415*10^-
10,   ...
                 -3.5324698866183274*10^-7,  -2.864726752719305*10^-8,
...
                 4.22203278860423 3*10^-6,    5.49693105547489*10^-7,
...
                 -0.00003092830508017534,    -4.212835064605567*10^-7,
...
                 0.00016479057910264333,     -0.000037508744144871775,
...
                 -0.0007330652888770748,     0.0002658102686466399,
...
                 0.002714164795025192,       -0.0009683220303068 76,
...
                 -0.007856355179516 13,       0.003796912205844 2175,
...
                 0.02059795248498291,        -0.012308050256422173,
...
                 -0.05640015507726106,       0.007187328235056668,
...
                 0.0956457447512976,         -0.005802788490189 4556,
...
```

```
                 -0.12231754226725715,       0.1631514710968921,   ...
                 0.6543719294101015,         0.6715904122541644,   ...
                 0.18132650835646258,        -0.15599473760485902,
...
                 -0.08362502581802364,       0.041304200470714036,
...
                 0.017768444024218653,       -0.013548122888083406,
...
                 0.010296151957318115,       0.014633468106540926,
...
                 -0.006440853043345421,      -0.008929653602321046,
...
                 0.0019327781450455884,      0.0036421681089588136,
...
                 -0.00033935744667514006,    -0.0010791779886949856,
...
                 0.000027173893640607642,    0.00023869852577104 5,
...
                 1.0500229544977402*10^-6,   -0.000039832648663228424,
...
                 -5.410954871718931*10^-7,   4.768328003833541*10^-6,
...
                 3.284183959667723*10^-8,    -3.759486167421689*10^-7,
...
                 1.085292734525874*10^-9,    1.5021122420804858*10^-8
];
      end

end

 if strncmpi(Type,'Vaidyanathan',3),
       f = [      -.000062906118    .000343631905     -.000453956620
      ...
                  -.000944897136    .002843834547     .000708137504
      ...
                  -.008839103409    .003153847056     .019687215010
      ...
                  -.014853448005    -.035470398607    .038742619293
      ...
                  .055892523691     -.077709750902    -.083928884366
      ...
                  .131971661417     .135084227129     -.194450471766
      ...
                  -.263494802488    .201612161775     .635601059872
      ...
                  .572797793211     .250184129505     .045799334111
      ];
end

 if strncmpi(Type,'Battle',3),
      if Par == 1,
      g = [0.578163    0.280931    -0.0488618   -0.0367309 ...
          0.012003    0.00706442  -0.00274588  -0.00155701 ...
          0.000652922 0.000361781 -0.000158601 -0.0000867523
          ];
      end
```

```
       if Par == 3,

       g = [0.541736    0.30683      -0.035498    -0.0778079 ...
         0.0226846   0.0297468    -0.0121455   -0.0127154 ...
         0.00614143  0.00579932   -0.00307863  -0.00274529 ...
         0.00154624  0.00133086   -0.000780468 -0.00065562 ...
           0.000395946 0.000326749  -0.000201818 -0.000164264 ...
         0.000103307
          ];
      end

       if Par == 5,
       g = [0.528374    0.312869    -0.0261771   -0.0914068 ...
         0.0208414   0.0433544   -0.0148537   -0.0229951  ...
         0.00990635  0.0128754   -0.00639886  -0.00746848 ...
         0.00407882  0.00444002  -0.00258816  -0.00268646 ...
         0.00164132  0.00164659  -0.00104207  -0.00101912 ...
           0.000662836 0.000635563 -0.000422485 -0.000398759 ...
           0.000269842 0.000251419 -0.000172685 -0.000159168 ...
           0.000110709 0.000101113
          ];
      end
         l = length(g);
         f = zeros(1,2*l-1);
         f(l:2*l-1) = g;
         f(1:l-1) = reverse(g(2:l));
end
 if strncmpi(Type,'Lemarie',3),
     “f = [  3.0620193e-005  3.8631712e-005 -5.9878645e-005 ...
-7.4997553e-005 1.1762624e-004 1.4604017e-004 ...
-2.3234459e-004 -2.8538040e-004 4.6210986e-004  ...
 5.5993014e-004 -9.2718796e-004 -1.1037270e-003 ...
 1.8821116e-003 2.1866870e-003 -3.8823801e-003 ...
-4.3537889e-003  8.2013819e-003  8.6851939e-003 ...
-1.7982112e-002 -1.7176116e-002 4.2067863e-002 ...
 3.2080525e-002 -1.1003574e-001 -5.0201141e-002 ...
 4.3391759e-001 7.6612115e-001 4.3391759e-001...
 -5.0201141e-002 -1.1003574e-001  3.2080525e-002  ...
4.2067863e-002 -1.7176116e-002 -1.7982112e-002 ...
8.6851939e-003 8.2013819e-003 -4.3537889e-003 ...
-3.8823801e-003 2.1866870e-003 1.8821116e-003 ...
-1.1037270e-003 -9.2718796e-004  5.5993014e-004 ...
 4.6210986e-004  -2.8538040e-004 -2.3234459e-004  ...
1.4604017e-004  1.1762624e-004 -7.4997553e-005 ...
-5.9878645e-005 ];
end
 if strncmpi(Type,'Pollen',3),
      s = 2 * sqrt(2);
      if length(Par) == 1
          phi = Par(1);
  f =[ (1 + cos(phi) - sin(phi))/s ...
      (1 + cos(phi) + sin(phi))/s ...
      (1 - cos(phi) + sin(phi))/s ...
      (1 - cos(phi) - sin(phi))/s];
     end
      if length(Par)==2
     phi1 = Par(1);
     phi2 = Par(2);
     s = 2 * sqrt(2);
 f=[( 1+cos(phi1) - cos( phi2 ) - cos( phi1) * cos( phi2) ...
       + sin( phi1 ) - cos( phi2) * sin( phi1) - sin( phi2 )  ...
       + cos( phi1) * sin( phi2) - sin( phi1) * sin( phi2 ) )/ (2*s)
...
   ( 1-cos( phi1) + cos( phi2) - cos( phi1) * cos( phi2) ...
       + sin( phi1) + cos( phi2) * sin( phi1) - sin( phi2) ...
       - cos( phi1) * sin( phi2) - sin( phi1) * sin( phi2) )/ (2*s) ...
   ( 1 + cos( phi1) * cos( phi2) + cos( phi2) * sin( phi1) ...
       - cos( phi1) * sin( phi2 ) + sin( phi1) *sin( phi2 ) )/s ...
   ( 1 + cos( phi1 ) * cos( phi2 ) - cos( phi2 ) * sin( phi1 ) ...
       + cos( phi1 )* sin( phi2 ) + sin( phi1)*  sin( phi2) )/s ...
   ( 1-cos( phi1) + cos( phi2 )- cos( phi1 )* cos( phi2 ) ...
       - sin( phi1 ) - cos( phi2 ) * sin( phi1 ) + sin( phi2 ) ...
       + cos( phi1 )* sin( phi2 )- sin( phi1 )* sin( phi2) ) /(2*s)...
   ( 1+cos( phi1 )- cos( phi2 )- cos( phi1 )* cos( phi2 ) ...
       - sin( phi1 ) + cos( phi2 ) * sin( phi1 ) + sin( phi2 ) ...
       - cos( phi1 ) * sin( phi2 ) - sin( phi1 )*  sin( phi2) )/
(2*s)];
     end
end
 f = f ./ norm(f);
```

المستخلص

لهدف الرئيسي لهذه الأطروحة هو دراسة التقدير غير المعلمي لدالة معدل الخطر باستخدام تقدير المويجة الخطية لبيانات الرقابة العشوائية من اليمين. تعتمد استراتيجية التقدير على استخدام الإسقاط المويجي لدالة الأب $\{\varphi_{J,k}(x), J = 1,2, \dots , 0 \le k \le 2^{J-1}\}$ على الفضاء الجزئي (V_J) من فضاء الدوال المربعة L ^ 2 (R) ، مع مقدّر برسلو للدالة التراكمية.

تم تصميم تقنية تقدير دالة الخطر بإيجاد النسبة بين مقدر المويجة الخطية لدالة الكثافة الاحتمالية (f(x)) والتقدير التجريبي لدالة البقاء {S(x)=1-F(x)}.

تعتمد التطبيقات على نوعين من الدراسة. التطبيق الأول ، دراسة المحاكاة لتوليد متغيرات عشوائية لتجربتين ، التجربة الاولى باستخدام توزيع كاما G(5 ، 1)لعينات بالحجم n = 100،200 ، والتجربة الثانية باستخدام التوزيع الثنائي الأبعاد لتوزيعتين طبيعيين (X=0.4X1~N(5,1)+0.6X2~N(3,0.5)) لعينات بالحجم n=400,600. علاوة على ذلك ، لتوليد أوقات الرقابة للتجربتين نستخدم التوزيع الأسي (EXP (6)). التطبيق الثاني ، يتم تطبيق ثلاث بيانات حقيقية وهي بيانات تليف الكبد الوبائي، وبيانات التمريض المنزلي، وبيانات ستانفورد لزراعة القلب.

تمت المقارنة باستخدام MSE بين نتائج التقدير والنتائج الفعلية لتجربتي المحاكاة ، اما بالنسبة للبيانات الحقيقية ، تم إجراء MSE بين نتائج التقدير التي تم حسابها بواسطة تقدير المويجات ومقدر النواة.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بغداد
كلية التربية للعلوم الصرفة (ابن الهيثم)
قسم الرياضيات

التقدير اللامعلمي لدالة الخطورة باستخدام تحويل المويجة

أطروحة

مقدمة الى كلية التربية للعلوم الصرفة (ابن الهيثم)

جامعة بغداد

وهي جزء من متطلبات نيل شهادة الدكتوراه في فلسفة الرياضيات

من قبل

علي طالب محمد

بأشراف

أ.د. أيدن حسن حسين

1440 *2019*