

151237

DIGITAL COMPUTATION OF THE FRACTIONAL MELLIN TRANSFORM WITH APPLICATIONS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylül University
In Partial Fulfillment of the Requirements for
the Degree of Master of Science in Electrical and Electronics Engineering

151237

by
Edip BİNER

July, 2004

İZMİR

Ms.Sc. THESIS EXAMINATION RESULT FORM

We certify that we have read the thesis, entitled “**DIGITAL COMPUTATION OF THE FRACTIONAL MELLIN TRANSFORM WITH APPLICATIONS**” completed by **Edip BİNER** under supervision of **Asst. Prof. Dr. Olcay AKAY** and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.


Asst. Prof. Dr. Olcay AKAY

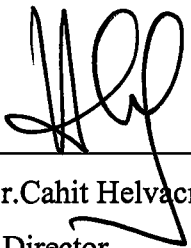
Supervisor


Asst. Prof. Dr. Hakkı T. YALAZAN

Committee Member


Committee Member

Approved by the
Graduate School of Natural and Applied Sciences


Prof. Dr. Cahit Helvacı
Director

ACKNOWLEDGMENTS

I would like to thank my supervisor Asst. Prof. Dr. Olcay AKAY for his valuable guidance and patient support during the course of this thesis. I also wish to express my sincere appreciation to my friend Erten ERÖZDEN for the valuable insight he has provided.

Edip BİNER



ABSTRACT

Scalings of physical variables, such as in the magnification of an image or in the variation of the frequency of a moving wave source with respect to a stationary observer, are facts of everyday life. In signal analysis, the Mellin transform is used to analyze signals with respect to their scale content. Much like the fractional Fourier transform is a generalization of the classical Fourier transform onto the time-frequency plane, the fractional Mellin transform is defined as a generalization of the ordinary Mellin transform onto the scaled time-frequency plane. In this thesis, algorithms are developed to compute the fractional Mellin transform digitally using the MATLAB numeric analysis software package. Performances of the various algorithms are compared through different simulations.

Keywords : Mellin transform, fractional Mellin transform, fractional Fourier transform, time–frequency analysis, time –scale analysis



ÖZET

Bir görüntünün büyütülmesindeki veya hareketli bir dalga kaynağının frekansının sabit bir gözlemciye göre değişmesindeki gibi fiziksel değişkenlerin ölçeklenmeleri, günlük hayatta karşılaşılabilecek olaylardandır. Sinyal analizinde kullanılan Mellin dönüşümü, sinyalleri ölçek içeriklerine göre incelemek için kullanılabilir. Klasik Fourier dönüşümünün zaman-frekans düzlemine genellemesi olarak kesirli Fourier dönüşümü tanımlanabildiği gibi, ölçeklenmiş zaman-frekans düzlemi üzerinde de klasik Mellin dönüşümünün genellemesi olarak kesirli Mellin dönüşümü tanımlanabilir. Bu tezde, kesirli Mellin dönüşümünün MATLAB nümerik analiz yazılım paketi üzerinde sayısal olarak hesaplanabilmesi için algoritmalar geliştirildi. Algoritmaların farklı versiyonlarının performansları da simülasyonlar yoluyla karşılaştırılmıştır.

Anahtar sözcükler : Mellin dönüşümü, kesirli Mellin dönüşümü, kesirli Fourier dönüşümü, zaman-frekans analizi, zaman-ölçek analizi

CONTENTS

	Page
Contents.....	IV
List of Tables	VII
List of Figures	VIII

Chapter One INTRODUCTION

Introduction.....	1
-------------------	---

Chapter Two INTRODUCTION TO FRACTIONAL TRANSFORMATIONS

2.1 Time and Scale Periodicity, Fourier and Scale Transforms	4
2.2 Operator Methods.....	6
2.2.1 Unitary and Hermitian Operators	6
2.2.2 Operator Associations	8
2.2.3 Stone's Theorem	8
2.2.4 Generalized Fourier Transforms	9
2.2.5 Unitary Equivalence as a Coordinate Transformation	11
2.2.6 Logarithmic Time - Axis Warping.....	13
2.3 Fractional Operators and the Fractional Fourier Transform	14
2.4 Fractional Mellin transform	18

Chapter Three

DEVELOPMENT OF THE LOGARITHMIC WARPING ALGORITHM

3.1 Logarithmic Warping	23
3.2 Warping as a Signal Processing Tool.....	24
3.3 Logarithmic Warping with A Priori Knowledge of Time.....	29
3.4 Logarithmic Warping without A Priori Knowledge of Time.....	32
3.5 Further Complications and Modifications on the Algorithm.....	36

Chapter Four

DIGITAL COMPUTATION OF THE FRACTIONAL MELLIN TRANSFORM

4.1 The Fractional Mellin Transform with A Priori Knowledge of Time.....	39
4.2 The Fractional Mellin Transform without A Priori Knowledge of Time	52
4.2.1 FrMT with <i>logwarp.m</i>	52
4.2.2 FrMT with <i>logwarp3.m</i>	57

Chapter Five

APPLICATIONS OF THE FRACTIONAL MELLIN TRANSFORM

5.1 FrMT of the Hypbolic Chirp Signal.....	64
5.2 A Simple Application – Filtering in the Fractional Scale Domain	69
5.3 An Example of FrFT versus FrMT	695

Chapter Six

CONCLUSIONS

Conclusions.....	78
------------------	----

References.....	80
-----------------	----

Appendix A

SOURCE CODE FOR THE ALGORITHMS USED IN THE THESIS

A.1 The Function Ulog(),.....	69
A.2 Testulog.m.....	69
A.3 Logwarp.m – First version	69
A.4 Logwarptest3.m – used to test logwarp.m.....	69
A.5 Logwarptest4.m – used to test logwarp.m.....	69
A.6 Sample.m – used in logwarptest3 and logwarptest4	69
A.7 Fracf.m – The Script Used for Taking the Fractional Fourier Transform.....	69
A.8 Corefr.m – Helper function to Fracf.m	69
A.9 Logwarp3.m – Logarithmic Warping Compliant with the Fractional Fourier Algorithm	69
A.10 Nearest.m – helper function used in Logwarp3.m.....	69
A.11 Fracfsamples.m – Helper function used by logwarp3.m	69
A.12 Tstfrmulog.m – Test file for obtaining the fractional Mellin Transform with Ulog.m.....	69
A.13 Pltcplx.m – helper function to tstfrmulog.m	100
A.14 Tstfrmlw.m – – Test file for obtaining the fractional Mellin Transform with logwarp.m.....	101
A.15 Tstfrmlw3.m – Used for Testing the Script Logwarp3.m	104
A.16 Hypchirptest.m – Used for Finding the FrMT of the Hyperbolic Chirp using Logwarp3.m	108

LIST OF TABLES

Page

Table 2. 1 Eigenfunctions Associated with Operators	10
Table 2. 2 Scaled Versions of Time and Frequency Operators	13
Table 2. 3 Definitions of Scaled Time and Frequency Operators	14



LIST OF FIGURES

	Page
Figure 2.1 Time, Frequency and Fractional domains and the operators associated with them.....	15
Figure 2.2 Operators used in the calculation of the FrFT	17
Figure 2.3 Block diagram representation of the FrFT	18
Figure 2.4 Time, Frequency and Fractional Mellin domains and the operators associated with them	19
Figure 2.5 Block diagram representation of FrMT	21
Figure 3. 1 Logarithmically modulated sawtooth wave via inverse logarithmic warping, U_{\log}^{-1} ; $(U_{\log}^{-1}h)(t) = \frac{1}{\sqrt{t}} \sum_{k=1}^5 \frac{(-1)^k}{k} \sin(2\pi k f_0 \ln(t))$	26
Figure 3. 2 Inverse logarithmically warped signal corrupted by noise, $s(t) = (U_{\log}^{-1}h)(t) + n(t)$	26
Figure 3. 3 The noisy sawtooth signal obtained after applying U_{\log} to the analyzed signal $s(t)$, $(U_{\log}s)(t) = h(t) + n'(t)$	27
Figure 3. 4 Fourier transform of the logarithmically warped signal in Figure 3.3	27
Figure 3. 5 The spectrum obtained after multiplication of the spectrum in Figure 3.4 with a comb filter mask	28
Figure 3. 6 The recovered signal as obtained through an inverse logarithmic warping operation of the Fourier domain masked signal having the spectrum in Figure 3.5.....	28

Figure 3. 7 Comparison of the recovered warped sawtooth wave (dashed line) with the original analyzed signal $(U_{\log}^{-1}h)(t)$ (solid line) in Figure 3.1.....	29
Figure 3. 8 Inverse logarithmic warping of a sine signal $\sin(2\pi f_0 t)$ (top), and the logarithmic warp of $\frac{1}{\sqrt{t}} \sin(2\pi f_0 (\ln t))$ (bottom)	31
Figure 3. 9 Inverse logarithmic warping of a real-valued chirp signal $\sin(\pi m_0 t^2)$ (top), and the logarithmic warp of $\frac{1}{\sqrt{t}} \sin(\pi m_0 (\ln t)^2)$ (bottom).....	31
Figure 3. 10 Graph illustrating $\ln(t)$ vs. $\ln^2(t)$	32
Figure 3. 11 Logarithmic warping of $\sin(2\pi f_0 t)$ using <i>logwarp.m</i> ($f_0=0.25$ Hz).....	35
Figure 3. 12 Logarithmic warping of a square wave of frequency 0.25 Hz using <i>logwarp.m</i>	35
Figure 3. 13 Logarithmic warping of $\sin(2\pi f_0 t)$ using <i>logwarp3.m</i> ($f_0=0.25$ Hz)....	38
Figure 3. 14 Logarithmic warping of a square wave of 0.25 Hz using <i>logwarp3.m</i>	38
Figure 4.1 Real part of the complex exponential $e^{j2\pi f_0 t}$ with $f_0=1.5$ Hz	41
Figure 4.2 Real part of the inverse logarithmic warping of $e^{j2\pi f_0 t}$	41
Figure 4.3 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{j2\pi f_0 t}$	42
Figure 4.4 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{j2\pi f_0 t})$	42
Figure 4.5 Comparison of the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ and the theoretical FrFT of $e^{j2\pi f_0 t}$ at $\phi=\pi/5$	43
Figure 4.6 Real part of the chirp signal $e^{j\pi m_0 t^2}$ with $m_0=0.05$	43
Figure 4.7 Real part of the inverse logarithmic warping of $e^{j\pi m_0 t^2}$	44
Figure 4.8 Absolute value of the FrFT of $e^{j\pi m_0 t^2}$ computed at the matching angle $\phi=\text{atan}(m_0)+\pi/2$	44
Figure 4.9 Absolute value of the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ computed at the matching angle $\phi=\text{atan}(m_0)+\pi/2$	45

Figure 4.10 Comparison of the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ and the theoretical FrFT of $e^{j\pi m_0 t^2}$	45
Figure 4.11 The Gaussian signal $e^{-\pi t^2}$	47
Figure 4.12 Real part of the inverse logarithmic warping of $e^{-\pi t^2}$	47
Figure 4.13 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{-\pi t^2}$	48
Figure 4.14 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{-\pi t^2})$	48
Figure 4.15 Comparison of the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ and the theoretical FrFT of $e^{-\pi t^2}$	49
Figure 4.16 The constant signal $c=5$	49
Figure 4.17 Real part of the inverse logarithmic warping of $c=5$	50
Figure 4.18 Absolute value of the FrFT at $\phi=\pi/5$ of $c=5$	50
Figure 4.19 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(c)$	51
Figure 4.20 Comparison of the FrMT of $U_{\log}^{-1}(c)$ and the theoretical FrFT of c	51
Figure 4.21 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{j2\pi f_0 t}$	53
Figure 4.22 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ using <i>logwarp.m</i>	53
Figure 4.23 Absolute value of the FrFT of a chirp $e^{-j\pi m_0 t^2}$ computed at the matching angle $\phi=\text{atan}(m_0)+\pi/2$ ($m_0=0.1$)	54
Figure 4.24 Absolute value of the FrMT of $U_{\log}^{-1}(e^{-j\pi m_0 t^2})$ calculated at the matching angle $\phi=\text{atan}(m_0)+\pi/2$ using <i>logwarp.m</i> ($m_0=0.1$)	54
Figure 4.25 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{-\pi t^2}$	55
Figure 4.26 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{-\pi t^2})$ using <i>logwarp.m</i> ..	55
Figure 4.27 Absolute value of the FrFT at $\phi=\pi/5$ of $c=5$	56
Figure 4.28 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(c)$ using <i>logwarp.m</i>	56
Figure 4.29 Comparison of the practical FrFT at $\phi=\pi/5$ of the complex exponential $e^{j2\pi f_0 t}$ with the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ computed using <i>logwarp3.m</i>	59

Figure 4.30 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the complex exponential $e^{j2\pi f_0 t}$ with the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ computed using <i>logwarp3.m</i>	59
Figure 4.31 Comparison of the practical FrFT of the chirp $e^{j\pi m_0 t^2}$ at the matching angle $\phi=\text{atan}(m_0)+\pi/2$ with the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ at the same angle computed using <i>logwarp3.m</i>	60
Figure 4.32 Comparison of the theoretical FrFT of the chirp $e^{j\pi m_0 t^2}$ at the matching angle $\phi=\text{atan}(m_0)+\pi/2$ with the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ at the same angle calculated using <i>logwarp3.m</i>	60
Figure 4.33 Comparison of the practical FrFT at $\phi=\pi/5$ of the Gaussian signal $e^{-\pi t^2}$ with the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ computed using <i>logwarp3.m</i>	61
Figure 4.34 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the Gaussian signal $e^{-\pi t^2}$ with the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ calculated using <i>logwarp3.m</i>	61
Figure 4.35 Comparison of the practical FrFT at $\phi=\pi/5$ of the constant signal $c=5$ with the FrMT of $U_{\log}^{-1}(c)$ computed using <i>logwarp3.m</i>	62
Figure 4.36 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the constant signal $c=5$ with the FrMT of $U_{\log}^{-1}(c)$ calculated using <i>logwarp3.m</i>	62
Figure 5.1 The real part of the hyperbolic chirp of Eq. (5.1) for $\phi=\pi/(2.1)$ and $r_0=1$	65
Figure 5.2 Absolute value of the FrFT of the theoretically logarithmic warped hyperbolic chirp computed at the matching angle $\phi=\pi/2.1$	67
Figure 5.3 Absolute value of the FrMT of the hyperbolic chirp calculated at the matching angle $\phi=\pi/2.1$ using <i>logwarp3.m</i>	67
Figure 5.4 Absolute value of the FrFT of the theoretically logarithmic warped hyperbolic chirp calculated at the mismatched angle $\phi=\pi/3$	68
Figure 5.5 Absolute value of the FrMT of the hyperbolic chirp computed at the mismatched angle $\phi=\pi/3$ using <i>logwarp3.m</i>	68

Figure 5.6 The hyperbolic chirp of chirp rate $\cot(\pi/(2.1))$ (top), theoretical logarithmic warping of the hyperbolic chirp (middle) and the practical logarithmic warping of the hyperbolic chirp using <i>logwarp3.m</i> (bottom)	69
Figure 5.7 The hyperbolic chirp (top) of chirp rate $\cot(\pi/5)$, theoretical logarithmic warping of the hyperbolic chirp (middle) and the practical logarithmic warping of the hyperbolic chirp using <i>logwarp3.m</i> (bottom)	69
Figure 5.8 Graph of $(U_{\log}^{-1})(e^{-\pi t^2})$	69
Figure 5.9 Graph of $(U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]}), f_0 = 6, m_0 = -1$	69
Figure 5.10 Graph of $s(t) = (U_{\log}^{-1})(e^{-\pi t^2}) + (U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]})$	69
Figure 5.11 FrMT of $s(t) = (U_{\log}^{-1})(e^{-\pi t^2}) + (U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]})$ computed at the matching angle $\phi = \text{atan}(m_0) + \pi/2$	69
Figure 5.12 The isolated Gaussian in the fractional scale domain obtained after a simple masking of the FrMT signal in Figure 5.11	69
Figure 5.13 Inverse fractional Fourier transform at the matching angle ϕ of the filtered signal in Figure 5.12	69
Figure 5.14 The recovered warped Gaussian signal obtained after inverse logarithmic warping of the signal in Figure 5.13	69
Figure 5.15 Comparison of the recovered warped Gaussian (solid line) with the original starting signal $(U_{\log}^{-1})(e^{-\pi t^2})$ (dashed line)	69
Figure 5.16 The rectangular pulse (dashed) and its logarithmically warped	69
Figure 5.17 The FrFT (top) and the FrMT (bottom) of a rectangular pulse at different angles from 0 to $\pi/2$ radians increasing in steps of $\pi/10$ (all figures show the magnitude plots of the transforms)	69

CHAPTER ONE

INTRODUCTION

The concept of scale is common throughout everyday life. A first example that comes to mind is magnification, which can be thought of as a positive scaling of the physical coordinate axes. The Doppler effect, where the frequency of a moving wave source is perceived differently by stationary sources in front of and behind the wave source, can be regarded as the scaling of the frequency variable of the vibration source.

In spite of this ubiquity, we have a rather limited set of tools to analyze this variable. The Mellin transform established by Robert Hjalmar Mellin (1854 - 1933) was initially intended as a tool to analyze number theoretic functions such as the gamma function, hypergeometric functions, and Riemann zeta function. Its use was later extended to the solution of partial differential equations and the generation of asymptotic expansions (WEB_1, 2004).

The use of the Mellin transform as a signal processing tool is fairly recent. In (Casasent et. al., 1977), Psaltis and Casasent use the Mellin transform to achieve scale invariant autocorrelation and cross-correlations for optical pattern recognition. Altes used the Mellin transform to simulate mammalian hearing (Altes, 1978). In another paper, Zwicke and Kiss used the Mellin transform to classify ships from their radar profiles (Zwicke & Kiss, 1983).

In this thesis, we develop a discrete algorithm in order to implement the fractional Mellin transform (FrMT) which is a generalization of the Mellin transform into the scaled time frequency plane. The FrMT itself is motivated by the fractional Fourier

transform (FrFT) which can be thought of as a rotation of the signal in the time-frequency plane. Being a mathematical generalization of the ordinary Fourier transform, the FrFT gained attention as a signal processing tool thanks to Almeida's paper on the subject (Almeida, 1994). In that paper, properties of the FrFT are derived and its relationships with time-frequency representations are established. After the appearance of Almeida's paper, Ozaktas and his colleagues developed a fast approximate discrete FrFT algorithm (Ozaktas et. al., 1996). Motivated by Baraniuk and Jones' work on unitary equivalence (Baraniuk & Jones, 1995), and Sayeed and Jones' papers on integral transforms covariant to unitary operators (Sayeed & Jones, 1996), Akay defined the FrFT and FrMT in terms of unitary fractional shift operators (Akay & Boudreaux-Bartels, December 1998), (Akay & Boudreaux-Bartels, August 1998). In this thesis, Akay's work is followed with the aim of developing an algorithm for computation of the FrMT.

In Chapter 2, a theoretical study of operator methods is given. After providing a brief introduction to the concept of scale as a signal variable, unitary and Hermitian operators, their associations with physical variables, and their equivalence relations are presented. Generalized Fourier transforms are defined as being the inner product of the given signal and the eigenfunctions of the operators. Within this framework, the FrFT and the FrMT are defined using the newly defined unitary fractional shift and unitary fractional scale shift operators.

Chapter 3 introduces the logarithmic warping algorithms used in various stages of the development. These algorithms are devised depending on the existence or the absence of the knowledge of the complete time orientation of the analyzed signal.

In Chapter 4, the warping algorithms are used together with the discrete FrFT code to realize the discrete FrMT. Simulations comparing various algorithms are presented.

In Chapter 5, the FrMT algorithm was demonstrated on several examples, namely the detection of a hyperbolic chirp, separation of a logarithmically warped Gaussian signal from a logarithmically warped chirp, and the comparison of the FrFT and FrMT of a rectangular pulse through different angles.

Finally, our conclusions are given in Chapter 6.



CHAPTER TWO

INTRODUCTION TO FRACTIONAL TRANSFORMATIONS

In this chapter, the tools and methods that form the background of this thesis are briefly reviewed. In the first section, the scale variable is introduced. In section two, operator methods are studied within the subject of signal analysis. Then, in sections three and four, the topics of time-frequency analysis and scale are combined to introduce the fractional Fourier and fractional Mellin transforms.

2.1 Time and Scale Periodicity, Fourier and Scale Transforms

A function $g(t)$ is *periodic* in time if

$$g(t) = g(t + T_0), \quad \forall t \in \mathbb{R} \quad \text{Eq. (2.1)}$$

where T_0 is the fundamental period. Fourier analysis began as an attempt to represent such signals using sinusoids. For periodic functions, a discrete set of Fourier coefficients can be calculated such that $g(t)$ can be written as a sum of complex sinusoids

$$g(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk\omega_0 t}, \quad a_k = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} g(t) e^{-jk\omega_0 t} dt \quad \text{Eq. (2.2)}$$

where $\omega_0 = \frac{2\pi}{T_0}$ is called the fundamental frequency. For aperiodic signals, the

coefficients a_k are replaced by a continuous function $G(\omega)$ given by

$$G(\omega) = \int_{-\infty}^{\infty} g(t) e^{-j\omega t} dt. \quad \text{Eq. (2.3)}$$

Another class of signals are known as “*scale periodic*” if they satisfy the following condition

$$f(t) = \sqrt{\tau} f(t\tau), \quad \forall \tau > 0 \quad \text{Eq. (2.4)}$$

where τ corresponds to the scale period. Note that ordinary periodic signals assume the same value at time points that form an arithmetic sequence whereas scale periodic signals assume scaled versions of the initial value at time points that form a geometric (or exponential) sequence.

An example of such a scale periodic signal would be of the form $\frac{e^{jC \ln t}}{\sqrt{t}}$. To check if this signal satisfies the scale periodicity property above, we perform the following;

$$\begin{aligned} f(t) &= \frac{e^{jC \ln t}}{\sqrt{t}} \\ f(t\tau) &= \frac{e^{jC \ln(t\tau)}}{\sqrt{t\tau}} = \frac{e^{jC \ln(\tau)} \cdot e^{jC \ln(t)}}{\sqrt{\tau} \cdot \sqrt{t}} = \frac{e^{jC \ln(\tau)}}{\sqrt{\tau}} \left(\frac{e^{jC \ln(t)}}{\sqrt{t}} \right) = \frac{e^{jC \ln(\tau)}}{\sqrt{\tau}} f(t). \end{aligned} \quad \text{Eq. (2.5)}$$

For the condition in Eq. (2.4) to hold, the term $e^{jC \ln(\tau)}$ should be equal to unity which requires

$$C \ln(\tau) = 2\pi k, \quad k \in \mathbb{Z}. \quad \text{Eq. (2.6)}$$

For $k = 1$, the fundamental scale, C_0 , associated with the function $f(t)$ is defined to be

$$C_0 \ln(\tau) = 2\pi. \quad \text{Eq. (2.7)}$$

Within the set of functions $\left\{ \frac{e^{jC \ln t}}{\sqrt{t}} \right\}$, only those functions for which $C = kC_0$ are

scale periodic with period τ . This can be thought as analogous to the fundamental frequency ω_0 of a harmonic oscillation where only the exponentials with frequencies $k\omega_0$ are periodic with period T_0 . In a manner similar to the Fourier series, one can define the *scale series* as

$$f(t) = \sum_{k=-\infty}^{+\infty} b_k \frac{e^{jkC_0 \ln(t)}}{\sqrt{t}}, \quad b_k = \frac{1}{\ln \tau} \int_a^b f(t) \frac{e^{-jkC_0 \ln(t)}}{\sqrt{t}} dt, \quad b/a = \tau \quad \text{Eq. (2.8)}$$

where integration from a to b corresponds to integration over one “scale period” τ . For functions which are not scale periodic, τ is let to approach infinity, C_0 approaches zero, and the product kC_0 in the integral in Eq. (2.8) can be treated as a continuous variable. One then defines the *Scale transform* (Sundaram et al., 1997) as

$$M(c) = \frac{1}{\sqrt{2\pi}} \int_0^{\infty} f(t) \frac{e^{-jc \ln(t)}}{\sqrt{t}} dt. \quad \text{Eq. (2.9)}$$

This is analogous to the Fourier Transform generalizing the Fourier Series for aperiodic functions. However, this progression from the scale periodic functions to the scale transform is not the historical development of the scale transform. Scale transform was previously developed in an operator theoretic manner by Cohen (Cohen, 1993). It is in that paper that Cohen defines the scale operators along with their eigenfunctions which form the kernel of the Scale transform.

2.2 Operator Methods

2.2.1 Unitary and Hermitian Operators

A function $f(t)$ is a mapping from the real time domain to the real range space. In finite dimensional spaces, a vector in a given n dimensional space \mathbb{R}^n has n components and is typically expressed as $\vec{x} \triangleq (x_1, x_2, \dots, x_n)$. The function $f(t)$, with its infinitely many points, can be thought of as an infinite dimensional vector; $\vec{f} \triangleq (f_1, f_2, \dots, f_{\infty})$. With the appropriate choice of norm, inner and outer product functions, vector analysis methods can be easily modified to serve the needs of signal processing. In general, our aim is to use the notion of a linear transformation in signal processing so that applying a transformation to a time domain signal maps it into such a domain that extracting information on the nature of the signal in that domain is much easier.

Operators are mathematical objects that can be used to denote either systems or transformations. Historically, operator methods have been used extensively in

quantum theory, prior to their adoption in signal processing. Thus, much of the jargon related to operator methods have roots in quantum theory.

There are two fundamental uses of operators. The first is to change the underlying basis of a signal $s(t)$ by transforming it into another domain (Sayeed & Jones, 1996). We can express this type of usage of an operator U as

$$s \mapsto Us. \quad \text{Eq. (2.10)}$$

The other is to change the underlying basis of a given operator A . This can be accomplished by applying a unitary operator U as (Baraniuk & Jones, 1995)

$$A \mapsto U^{-1}AU. \quad \text{Eq. (2.11)}$$

Two of the most commonly used operator types are *unitary* and *Hermitian* operators. Unitary operators preserve inner products, leaving the angle between the coordinate axes unchanged. That is, the inner product between two signals $g(t)$ and $h(t)$ is the same as the inner product between their transformed versions Ag and Ah

$$\langle Ag, Ah \rangle = \langle g, h \rangle. \quad \text{Eq. (2.12)}$$

In Eq. (2.12), the notation $\langle g, h \rangle$ denotes the inner product of two signals $g(t)$ and $h(t)$, which is defined by

$$\langle g, h \rangle = \int_{-\infty}^{\infty} g(t)h^*(t)dt \quad \text{Eq. (2.13)}$$

where $*$ represents complex conjugation. On the other hand, Hermitian operators satisfy the property;

$$\langle Ag, h \rangle = \langle g, Ah \rangle \quad \text{Eq. (2.14)}$$

for two signals $g(t)$ and $h(t)$. Note that in the above, we denote unitary operators with bold capital letters such as A and U whereas Hermitian operators are denoted by capital calligraphic letters like \mathcal{A} and \mathcal{B} .

2.2.2 Operator Associations

Traditionally, Hermitian operators have been associated with physical variables such as time, frequency and scale because the averages of Hermitian operators are always real valued (Cohen, 1995). Measurement of the associated quantity can be done by projecting the given signal onto the eigenfunctions of the Hermitian operator. The Hermitian operators associated with time, frequency and dilation (log modulation), as defined in the time domain, are given as (Baraniuk & Jones, 1995)

$$\begin{aligned}
 \text{Time: } (\mathcal{T}g)(t) &\equiv tg(t) \\
 \text{Frequency: } (\mathcal{F}g)(t) &\equiv \frac{1}{j2\pi} \frac{d}{dt} g(t) \\
 \text{Log Modulation: } (\mathcal{H}g)(t) &\equiv \left(\frac{\mathcal{TF} + \mathcal{FT}}{2} g \right)(t).
 \end{aligned} \tag{2.15}$$

Similarly, we can define the unitary operators associated with the fundamental variables of time, frequency and scale as (Baraniuk & Jones, 1995)

$$\begin{aligned}
 \text{Time Shift: } (\mathbf{T}_\tau g)(t) &\equiv g(t - \tau) \\
 \text{Frequency Shift: } (\mathbf{F}_\nu g)(t) &\equiv e^{j2\pi\nu t} g(t) \\
 \text{Dilation: } (\mathbf{D}_\gamma g)(t) &\equiv e^{-\gamma/2} g(e^{-\gamma} t).
 \end{aligned} \tag{2.16}$$

The log modulation and dilation operators mentioned above are the operators associated with the concept of scale.

2.2.3 Stone's Theorem

Using Stone's theorem from functional analysis, one can show that unitary and Hermitian operators are equivalent; i.e. given a Hermitian operator \mathcal{B} one can create a unitary operator \mathbf{A}_a associated with the physical variable a , by exponentiating \mathcal{B} via (Baraniuk & Jones, 1995) (Sayeed & Jones, 1996)

$$\mathbf{A}_a = e^{j2\pi a \mathcal{B}} = \sum_{n=0}^{\infty} \frac{(j2\pi a \mathcal{B})^n}{n!}. \tag{2.17}$$

Conversely, \mathcal{B} can be obtained from the unitary operator \mathbf{A}_a using the following relation

$$\mathcal{B} = \frac{1}{j2\pi} \lim_{a \rightarrow 0} \frac{\mathbf{A}_a - \mathbf{I}}{a}. \quad \text{Eq. (2.18)}$$

As a result of Stone's theorem, the equivalence of unitary and Hermitian time, frequency and scale (dilation and log modulation) operators can be expressed as follows (Baraniuk & Jones, 1995),

$$\mathbf{T}_{-t} = e^{j2\pi t \mathcal{F}}, \quad \mathbf{F}_f = e^{j2\pi f \mathcal{T}}, \quad \mathbf{D}_d = e^{j2\pi d \mathcal{H}}. \quad \text{Eq. (2.19)}$$

Why do we then need both unitary and Hermitian operators if they are equivalent? The answer lies in the fact that eigenfunctions of unitary and Hermitian operators define signal transformations with different properties as we will see in the next section.

2.2.4 Generalized Fourier Transforms

Given a unitary operator \mathbf{A}_a , we define the \mathbf{A}_a -Fourier transform, $\mathbb{F}_{\mathbf{A}}$, as the expansion of the time domain signal $s(t)$ onto the eigenfunctions $\mathbf{u}_\alpha^{\mathbf{A}}(t)$ of \mathbf{A}_a :

$$\hat{s}(\alpha) = (\mathbb{F}_{\mathbf{A}} s)(\alpha) \equiv \langle s(t), \mathbf{u}_\alpha^{\mathbf{A}}(t) \rangle = \int s(t) \mathbf{u}_\alpha^{\mathbf{A}*}(t) dt. \quad \text{Eq. (2.20)}$$

The inverse \mathbf{A} -Fourier transform is obtained by the expansion of the transformed signal onto the conjugate of the eigenfunctions of \mathbf{A}_a ,

$$s(t) = (\mathbb{F}_{\mathbf{A}}^{-1} \hat{s})(t) \equiv \langle \hat{s}(\alpha), \mathbf{u}_\alpha^{\mathbf{A}*}(t) \rangle = \int \hat{s}(\alpha) \mathbf{u}_\alpha^{\mathbf{A}}(t) d\alpha. \quad \text{Eq. (2.21)}$$

An \mathcal{A} -Fourier transform, associated with the Hermitian operator \mathcal{A} , is defined similarly in terms of the eigenfunctions $\mathbf{u}_\alpha^{\mathcal{A}}(t)$ of the \mathcal{A} .

Eigenfunctions of unitary and Hermitian operators associated with the same physical variable a are usually different. Thus, they lead to different generalized Fourier transforms;

$$\mathbf{u}_\alpha^{\mathbf{A}} \neq \mathbf{u}_\alpha^{\mathcal{A}} \Rightarrow (\mathbb{F}_{\mathbf{A}} s)(\alpha) \neq (\mathbb{F}_{\mathcal{A}} s)(\alpha). \quad \text{Eq. (2.22)}$$

\mathbf{A}_a and \mathcal{A} Fourier transforms behave differently when a unitary transformation \mathbf{A}_a is applied to the signal before the generalized Fourier transform. The \mathbf{A}_a - Fourier transform, \mathbb{F}_A , is *invariant* to \mathbf{A}_a , i.e. the effect of \mathbf{A}_a is not reflected in the magnitude of the transform (but is included as a phase factor),

$$|(\mathbb{F}_A \mathbf{A}_a s)(\alpha)| = |(\mathbb{F}_A s)(\alpha)|. \quad \text{Eq. (2.23)}$$

The \mathcal{A} - Fourier transform, on the other hand, is *covariant* to \mathbf{A}_a ; i.e. changes due to \mathbf{A}_a are reflected by \mathbb{F}_A as a time shift

$$(\mathbb{F}_A \mathbf{A}_a s)(\alpha) = (\mathbb{F}_A s)(\alpha - a). \quad \text{Eq. (2.24)}$$

This can also be interpreted as \mathbb{F}_A measuring the a - content of the signal $s(t)$.

Table 2. 1 Eigenfunctions Associated with Operators

Operators	Eigenfunctions
\mathbf{T}, \mathcal{F}	$\mathbf{u}_k^T(t) = \mathbf{u}_k^F(t) = e^{j2\pi kt}$
\mathbf{F}, \mathcal{T}	$\mathbf{u}_k^F(t) = \mathbf{u}_k^T(t) = \delta(t - k)$
\mathbf{D}, \mathcal{H}	$\mathbf{u}_k^D(t) = \mathbf{u}_k^H(t) = \frac{1}{\sqrt{t}} e^{j2\pi k \log(t)}, \quad t > 0$

Traditional time, frequency and scale operators and their associated eigenfunctions are given in Table 2.1 above. We can write the generalized Fourier transforms associated with these variables as follows;

$$\text{Fourier Tr.: } \mathbb{F}_F = \mathbb{F}_T = \left\langle s(t), e^{j2\pi kt} \right\rangle = \int_{-\infty}^{\infty} s(t) e^{-j2\pi kt} dt = S(k)$$

$$\text{Identity Tr.: } \mathbb{F}_T = \mathbb{F}_F = \left\langle s(t), \delta(t - k) \right\rangle = \int_{-\infty}^{\infty} s(t) \delta(t - k) dt = s(k) \quad \text{Eq. (2.25)}$$

$$\text{Mellin Tr.: } \mathbb{F}_H = \mathbb{F}_D = \left\langle s(t), \frac{1}{\sqrt{t}} e^{j2\pi k \log t} \right\rangle = \int_0^{\infty} s(t) \frac{1}{\sqrt{t}} e^{-j2\pi k \log t} dt.$$

The Fourier transform $\mathbb{F}_\mathcal{F}$, is invariant to time shifts and covariant to frequency shifts. Thus, it measures the frequency content of the signal. The identity transform $\mathbb{F}_\mathcal{T}$, is invariant to frequency shifts and covariant to time shifts. Consequently, it measures the time content of the signal. Finally, the Mellin transform is invariant to scale changes (dilations), and it measures the “logarithmic modulation” content of the signal $s(t)$.

2.2.5 Unitary Equivalence as a Coordinate Transformation

Two unitary operators $\tilde{\mathbf{A}}$ and \mathbf{A} are *unitarily equivalent* if they are related as

$$\tilde{\mathbf{A}} = \mathbf{U}^{-1}\mathbf{A}\mathbf{U} \quad \text{Eq. (2.26)}$$

where \mathbf{U} is another unitary operator. This relation corresponds to the operation of change of basis in linear algebra. Thus, we represent the unitary operator \mathbf{A} in terms of a new set of eigenfunctions $\tilde{\mathbf{u}}_\alpha^A(t)$. While the operator \mathbf{A} represents a single physical quantity a , the operator $\mathbf{U}^{-1}\mathbf{A}\mathbf{U}$ can represent an infinite spectrum of different physical quantities each corresponding to a particular unitary transformation \mathbf{U} (Baraniuk & Jones, 1995).

It can be easily shown that the unitary time shift operator \mathbf{T}_k and unitary frequency shift operator \mathbf{F}_k are unitarily equivalent with the conventional Fourier transform providing the unitary link;

$$\mathbf{F}_k = \mathbb{F}^{-1}\mathbf{T}_k\mathbb{F} \quad \text{Eq. (2.27)}$$

where \mathbb{F} represents the classical Fourier Transform. Thus, if a time shift is applied to a Fourier transformed signal and then the inverse Fourier transform is calculated, the net result is a shift in the frequency domain.

Using the principle we have just introduced, we can readily define two more unitary operators $\tilde{\mathbf{T}}_k$ and $\tilde{\mathbf{F}}_k$ by

$$\tilde{\mathbf{T}}_k = \mathbf{U}^{-1}\mathbf{T}_k\mathbf{U} \quad \text{Eq. (2.28)}$$

$$\tilde{\mathbf{F}}_k = \mathbf{U}^{-1} \mathbf{F}_k \mathbf{U}. \quad \text{Eq. (2.29)}$$

These new operators can be shown to be equivalent by using Eq. (2.27), Eq. (2.28) and Eq. (2.29) via the relation

$$\tilde{\mathbf{F}}_k = (\mathbf{U}^{-1} \mathbf{F} \mathbf{U})^{-1} \tilde{\mathbf{T}}_k (\mathbf{U}^{-1} \mathbf{F} \mathbf{U}). \quad \text{Eq. (2.30)}$$

Similar results can be derived for Hermitian time and frequency operators since they are equivalent to their unitary counterparts via Stone's theorem.

An important result of unitary equivalence relations is the generation of new generalized Fourier transforms since the eigenfunctions of the transformed operators are different as follows;

$$\tilde{\mathbf{T}}_k \mathbf{e}_k^{\tilde{\mathbf{T}}_k} = k \mathbf{e}_k^{\tilde{\mathbf{T}}_k} \quad \text{Eq. (2.31)}$$

$$\mathbf{U}^{-1} \mathbf{T}_k \mathbf{U} \mathbf{e}_k^{\tilde{\mathbf{T}}_k} = k \mathbf{e}_k^{\tilde{\mathbf{T}}_k} \quad \text{Eq. (2.32)}$$

$$\mathbf{T}_k (\mathbf{U} \mathbf{e}_k^{\tilde{\mathbf{T}}_k}) = k (\mathbf{U} \mathbf{e}_k^{\tilde{\mathbf{T}}_k}). \quad \text{Eq. (2.33)}$$

$$\mathbf{U} \mathbf{e}_k^{\tilde{\mathbf{T}}_k} = \mathbf{e}_k^{\mathbf{T}_k} \rightarrow \mathbf{e}_k^{\tilde{\mathbf{T}}_k} = \mathbf{U}^{-1} \mathbf{e}_k^{\mathbf{T}_k}. \quad \text{Eq. (2.34)}$$

If we want to find the $\tilde{\mathbf{F}}$ -covariant, $\tilde{\mathbf{T}}$ -invariant Fourier transform $\mathbb{F}_{\tilde{\mathbf{T}}}$, we need to take the inner product of the eigenfunctions of $\tilde{\mathbf{T}}$ and the signal $s(t)$;

$$\mathbb{F}_{\tilde{\mathbf{T}}} = \mathbb{F}_{\tilde{\mathbf{T}}} = \langle s(t), \mathbf{U}^{-1} \mathbf{e}_k^{\mathbf{T}_k} \rangle = \langle \mathbf{U} s(t), \mathbf{U} \mathbf{U}^{-1} \mathbf{e}_k^{\mathbf{T}_k} \rangle = \langle \mathbf{U} s(t), \mathbf{e}_k^{\mathbf{T}_k} \rangle = \mathbb{F}_{\mathbf{T}} \mathbf{U} = \mathbb{F}_{\mathcal{F}} \mathbf{U} = \mathbb{F} \mathbf{U} \quad \text{Eq. (2.35)}$$

where \mathbb{F} represents the classical Fourier transformation. Note that, in the above equations, the eigenfunctions of the corresponding operators are denoted by \mathbf{e} instead of the usual \mathbf{u} , to avoid confusion with the unitary operator \mathbf{U} . Eq. (2.35) uses the fact that applying a unitary transformation \mathbf{U} preserves inner products. Thus, the new transformation $\mathbb{F}_{\tilde{\mathbf{T}}}$ is $\tilde{\mathbf{F}}$ -covariant and $\tilde{\mathbf{T}}$ -invariant.

In a similar fashion, $\tilde{\mathbf{T}}$ -covariant, $\tilde{\mathbf{F}}$ -invariant transform can be formulated as

$$\mathbb{F}_{\tilde{\mathbf{T}}} = \mathbb{F}_{\tilde{\mathbf{F}}} = \mathbb{F}_{\tilde{\mathbf{F}}} \mathbf{U} = \mathbb{I} \mathbf{U} = \mathbf{U} \quad \text{Eq. (2.36)}$$

where \mathbb{I} represents the identity transform. Thus, $\mathbb{F}_{\tilde{T}}$ is \tilde{T} -covariant and $\tilde{\mathbb{F}}$ -invariant.

It is seen that once the unitary link has been decided upon, it is straightforward to calculate the associated generalized Fourier transforms in terms of the unitary link and the original transforms.

2.2.6 Logarithmic Time - Axis Warping

Until now we have mostly explored the time and frequency operators. We need to define a *unitary warping operator* in order to carry over time-frequency plane concepts into scale space. This can be done using the unitary warping operator U_{\log} defined as

$$(U_{\log}s)(t) \equiv e^{t/2}s(e^t). \quad \text{Eq. (2.37)}$$

This operator takes signals in $L^2(\mathbb{R}_+)$ and stretches them into signals in $L^2(\mathbb{R})$ (Baraniuk & Jones, 1995). Note that the operator scales both the argument and the signal itself. With this unitary link, we can define the unitary and Hermitian scaled time and scaled frequency operators as given in Table 2.2 and Table 2.3.

Table 2. 2 Scaled Versions of Time and Frequency Operators

	Unitary	Hermitian
Dilation (Scaled Time)	$U_{\log}^{-1} \mathbf{T}_k U_{\log} = \mathbf{D}_k$	$U_{\log}^{-1} \mathcal{T} U_{\log} = \mathcal{D}$
Log Modulation (Scaled Frequency)	$U_{\log}^{-1} \mathbf{F}_k U_{\log} = \mathbf{H}_k$	$U_{\log}^{-1} \mathcal{F} U_{\log} = \mathcal{H}$

The operators \mathbf{D} and \mathcal{D} are called unitary and Hermitian “dilation” operators, respectively. By dilation, we mean scaling of the time variable. The operators \mathbf{H} and \mathcal{H} are called unitary and Hermitian “log modulation” operators. Scaling of frequency in an exponential manner can be termed as logarithmic modulation.

Table 2. 3 Definitions of Scaled Time and Frequency Operators

	Unitary	Hermitian
Dilation (Scaled Time)	$(\mathbf{D}_d g)(t) \equiv e^{-d/2} g(e^{-d} t)$	$(\mathcal{D}g)(t) \equiv (\log t)g(t)$
Log Modulation (Scaled Frequency)	$(\mathbf{H}_h g)(t) \equiv e^{j2\pi h \log t} g(t)$	$(\mathcal{H}g)(t) \equiv \left(\frac{\mathcal{TF} + \mathcal{FT}}{2} g \right)(t)$

The scale-invariant, log-modulation covariant transform is given by

$$\mathbb{F}_{\mathcal{H}} = \mathbb{F}_{\mathbf{D}} = \mathbb{F}_{\mathcal{F}} \mathbf{U}_{\log} = \mathbb{F} \mathbf{U}_{\log}. \quad \text{Eq. (2.38)}$$

Similarly, log-modulation invariant, scale-covariant transform (Baraniuk, 1993) is given by

$$\mathbb{F}_{\mathcal{D}} = \mathbb{F}_{\mathbf{H}} = \mathbb{F}_{\mathcal{T}} \mathbf{U}_{\log} = \mathbb{I} \mathbf{U}_{\log} = \mathbf{U}_{\log}. \quad \text{Eq. (2.39)}$$

2.3 Fractional Operators and the Fractional Fourier Transform

Switching from the time domain to the frequency domain via Fourier transform reveals valuable information about the spectral content of the signal. However, the spectra of real world signals usually change with time. Thus, the need arises to represent signals in time and frequency domains simultaneously. Two most common examples of time-frequency representations (TFRs) are the Wigner distribution and the short time Fourier transform. Many such representations were devised over the years to satisfy different constraints arising from different application requirements (Hlawatsch & Bartels, 1992). In the following sections, a generalization of the Fourier transform into the time-frequency plane, the fractional Fourier transform (FrFT), and its scaled time-frequency counterpart, the fractional Mellin transform (FrMT), are introduced from an operator point of view. The FrFT, although not a TFR by itself, can be shown to be implicitly related to many TFRs. As the main subject of this thesis, the FrMT is an attempt to generalize the Mellin transform into a scaled TF plane.

Conventional time and frequency domains are considered to be orthogonal and the Fourier transform can be thought of as a mapping from the time domain, t , to the frequency domain, f , of the time-frequency plane as shown in Figure 2.1(a).

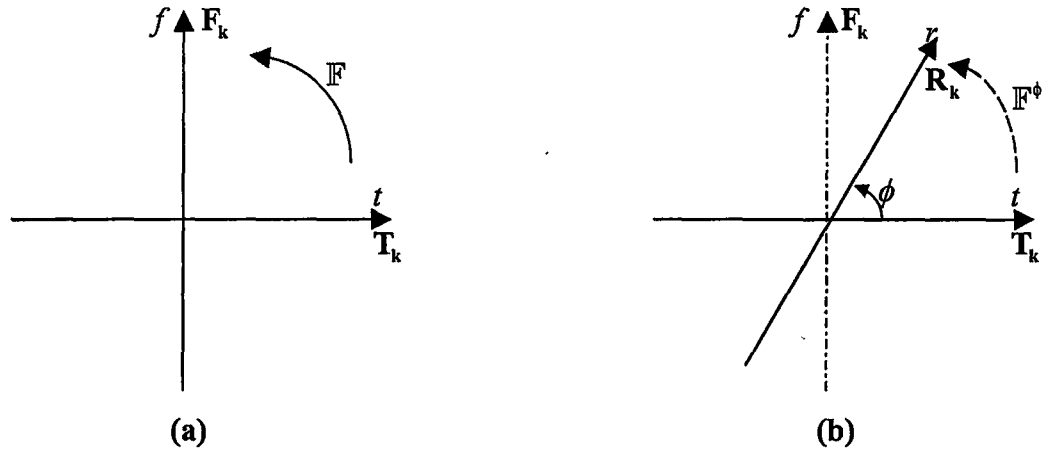


Figure 2. 1 Time, Frequency and Fractional domains and the operators associated with them

Now, let us consider a new signal domain in between the orthogonal time and frequency axes of the TF plane – the *fractional domain*, r , associated with the angle ϕ . Mapping of a signal onto this fractional domain can be thought of as the counterclockwise rotation of the signal by an angle ϕ from the positive time axis as in Figure 2.1(b). A rotation of $\pi/2$ radians would clearly amount to computing the classical Fourier transform of the signal. Analogously, a rotation of ϕ radians corresponds to calculating the fractional Fourier transform (FrFT) of the signal.

Just as the unitary time and frequency shift operators are unitarily equivalent via the Fourier transform as in Eq. (2.27), the unitary fractional shift operator R_p^ϕ associated with the fractional domain r is unitarily equivalent with the unitary time shift operator T_k , with the fractional Fourier transform \mathbb{F}^ϕ providing the unitary link (Akay, 2000)

$$R_p^\phi = \mathbb{F}^{-\phi} T_p \mathbb{F}^\phi. \quad \text{Eq. (2.40)}$$

The fractional shift operator corresponds to the operation of shifting the signal by an amount ρ along the fractional axis r that is at angle ϕ with the positive time axis t and is defined explicitly as (Akay & Boudreaux-Bartels, December 1998), (Akay, 2000)

$$(\mathbf{R}_\rho^\phi s)(t) = s(t - \rho \cos \phi) e^{-j\pi \rho^2 \cos \phi \sin \phi + j2\pi t \rho \sin \phi}. \quad \text{Eq. (2.41)}$$

Recall from Table 2.1 and Eq. (2.25) that to find the \mathcal{A} - Fourier transform that transforms the given signal into the 'a' domain associated with the Hermitian operator \mathcal{A} , one needs to know either the eigenfunctions of \mathcal{A} itself or the eigenfunctions of the equivalent unitary shift operator \mathbf{B}_b . Observing Eq. (2.19) and Table 2.1 carefully, one readily realizes that physical variables represented by unitary and Hermitian operators connected via Stone's theorem are orthogonal. For example; time and frequency are orthogonal quantities. Similarly dilation and log modulation are also orthogonal variables. From Eq. (2.25) it can be seen that to find the Fourier transform $\mathbb{F}_{\mathcal{F}}$ of a time domain signal one needs to know the eigenfunctions of the unitary time shift operator \mathbf{T}_k which are also the eigenfunctions of the Hermitian operator \mathcal{F} . In other words, to map a signal from a given domain into the orthogonal domain $\pi/2$ radians ahead, one either needs to know the eigenfunctions of the Hermitian operator associated with the target domain or equivalently the eigenfunctions of the unitary shift operator associated with the original domain. Therefore, to arrive at the fractional domain associated with the Hermitian fractional operator \mathcal{R}^ϕ via the fractional Fourier transform \mathbb{F}^ϕ , one needs to know the eigenfunctions of either \mathcal{R}^ϕ or $\mathbf{R}_\rho^{\phi-\pi/2}$ as illustrated in Figure 2.2.

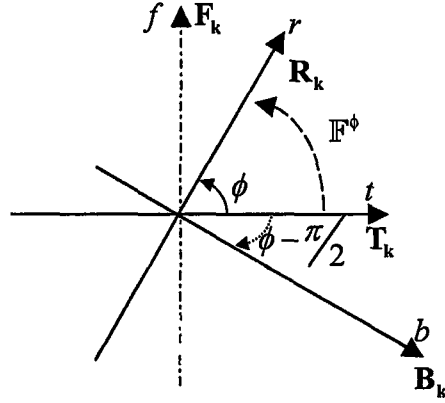


Figure 2. 2 Operators used in the calculation of the FrFT

The dual Hermitian fractional operator \mathcal{B}^ϕ is computed directly by substituting Eq. (2.40) in Eq. (2.18);

$$\begin{aligned}
 (\mathcal{B}^\phi s)(t) &= \frac{j}{2\pi} \lim_{\rho \rightarrow 0} \frac{(\mathbf{R}_\rho^\phi s)(t) - s(t)}{\rho} \\
 &= -\sin \phi [ts(t)] + \cos \phi \left[\frac{-j}{2\pi} \frac{d}{dt} s(t) \right] \\
 &= -(\sin \phi)T + (\cos \phi)\mathcal{F} = \mathcal{R}^{\phi+\pi/2}.
 \end{aligned}
 \tag{Eq. (2.42)}$$

The calculation of $\mathbf{B}_\beta^\phi = \mathbf{R}_\beta^{\phi-\pi/2}$ is not so straightforward however and is beyond the scope of this work (Akay, 2000). Using \mathbf{B}_β^ϕ , the Hermitian fractional operator \mathcal{R}^ϕ can be computed as

$$\begin{aligned}
 (\mathcal{R}^\phi s)(t) &= \frac{j}{2\pi} \lim_{\beta \rightarrow 0} \frac{(\mathbf{B}_\beta^\phi s)(t) - s(t)}{\beta} \\
 &= (\cos \phi)T + (\sin \phi)\mathcal{F}.
 \end{aligned}
 \tag{Eq. (2.43)}$$

Once the eigenfunctions $\mathbf{u}_k^{\mathcal{R}^\phi}(t, r) = \mathbf{u}_k^{\mathbf{R}_\beta^{\phi-\pi/2}}(t, r)$ of \mathcal{R}^ϕ are calculated (Akay, 2000) the calculation of the FrFT is straightforward as follows;

$$\begin{aligned}
(\mathbb{F}^\phi s)(r) &= \left\langle s(t), \sqrt{1+j \cot \phi} e^{-j2\pi \frac{(t^2+r^2)}{2} \cot \phi} e^{j2\pi r t \csc \phi} \right\rangle \\
&= \begin{cases} \sqrt{1-j \cot \phi} e^{j2\pi \frac{(r^2)}{2} \cot \phi} \int_{-\infty}^{\infty} s(t) e^{j2\pi \frac{(t^2)}{2} \cot \phi} e^{-j2\pi r t \csc \phi} dt & \phi \neq n\pi, n \in \mathbb{Z} \\ s(r) & \phi = (2n)\pi \\ s(-r) & \phi \neq (2n+1)\pi. \end{cases} \quad \text{Eq. (2.44)}
\end{aligned}$$

The process of calculating the FrFT can be summarized via the block diagram shown in Figure 2.3. The signal $s(t)$ is first multiplied with a linear chirp $e^{j\pi t^2 \cot \phi}$. Then, the classical Fourier transform of $s(t)e^{j\pi t^2 \cot \phi}$ is computed with the output variable scaled as $\frac{r}{\sin \phi}$. This provides the mapping into the fractional domain. For the transformation to be completed, the result is multiplied first with another chirp in the fractional domain followed by a final multiplication with the constant term $\sqrt{1-j \cot \phi}$. This constant term serves the purpose of making the FrFT an energy-preserving signal transform.

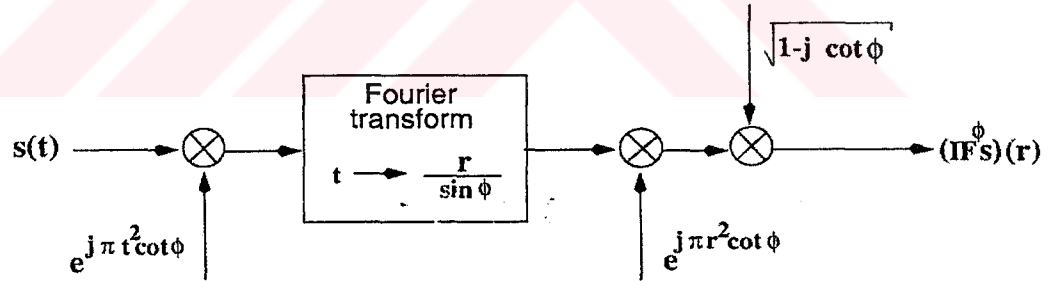


Figure 2.3 Block diagram representation of the FrFT

2.4 Fractional Mellin transform

Using the *unitary warping operator* \mathbf{U}_{\log} given in Eq. (2.37) as the unitary link, unitary scaled time shift (dilation) operator \mathbf{D}_k and the unitary scaled frequency shift (log modulation) operator \mathbf{H}_k were defined in Section 2.2. \mathbf{D}_k and \mathbf{H}_k are unitarily equivalent to \mathbf{T}_k and \mathbf{F}_k , respectively, via the unitary warping operator \mathbf{U}_{\log} . Similarly,

a unitary scaled fractional shift operator, M_p^ϕ , can be defined as unitarily equivalent to the unitary fractional shift operator R_p^ϕ via the unitary link U_{\log} (Akay, 2000)

$$M_p^\phi = U_{\log}^{-1} R_p^\phi U_{\log}. \quad \text{Eq. (2.45)}$$

The explicit definition can be written as (Akay, 2000)

$$(M_p^\phi s)(t) = \frac{1}{\sqrt{t}} s(e^{\ln t - \rho \cos \phi}) e^{\frac{\ln t - \rho \cos \phi}{2}} e^{-j\pi \rho^2 \cos \phi \sin \phi} e^{j2\pi \ln t \rho \sin \phi}.$$

The unitary warping operator U_{\log} can be thought of as warping the whole time-frequency plane into a scaled time-frequency plane as depicted in Figure 2.4. The unitary scaled fractional shift operator M_p^ϕ corresponds to the operation of shifting the signal by an amount ρ along the scaled fractional axis m that is at angle ϕ with the positive scaled time (dilation) axis, d . Just as the FrFT, by finding the eigenfunctions of M^ϕ , the Hermitian operator associated with the fractional scaled domain m , we can define the fractional Mellin transform (FrMT) as a generalization of the Mellin transform into the scaled TF plane.

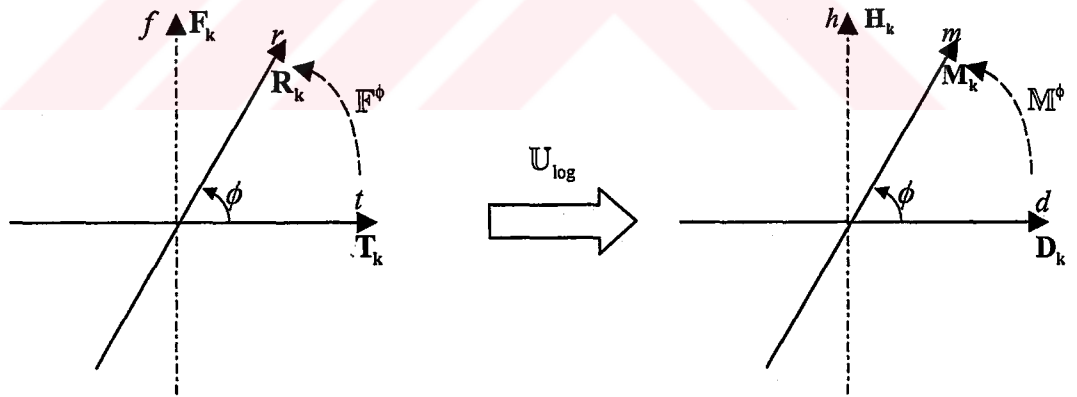


Figure 2. 4 Time, Frequency and Fractional Mellin domains and the operators associated with them

The Hermitian fractional scale operator, M^ϕ , is unitarily equivalent to the Hermitian fractional operator R^ϕ via U_{\log}

$$M^\phi = U_{\log}^{-1} R^\phi U_{\log}. \quad \text{Eq. (2.46)}$$

By substituting \mathcal{R}^ϕ in Eq. (2.43) we obtain

$$\begin{aligned}\mathcal{M}^\phi &= \mathbf{U}_{\log}^{-1} \mathcal{R}^\phi \mathbf{U}_{\log} = \mathbf{U}_{\log}^{-1} ((\cos \phi) \mathcal{T} + (\sin \phi) \mathcal{F}) \mathbf{U}_{\log} \\ &= (\cos \phi) (\mathbf{U}_{\log}^{-1} \mathcal{T} \mathbf{U}_{\log}) + (\sin \phi) (\mathbf{U}_{\log}^{-1} \mathcal{F} \mathbf{U}_{\log}) \\ &= (\cos \phi) \mathcal{D} + (\sin \phi) \mathcal{H}.\end{aligned}\tag{Eq. (2.47)}$$

As a result of the unitary equivalence relation in Eq. (2.45), the FrMT can be calculated, without resorting to any lengthy eigenfunction calculations (Akay, 2000). This is simply done by first applying the unitary warping operator \mathbf{U}_{\log} and then computing the FrFT, \mathbb{F}^ϕ , by virtue of the following relations

$$\mathbb{F}_{\mathcal{M}^\phi} = \mathbb{F}_{\mathbf{M}_p^{\frac{1}{2}}} = \mathbb{F}_{\mathbf{R}_p^{\frac{1}{2}}} \mathbf{U}_{\log} = \mathbb{F}^\phi \mathbf{U}_{\log} \equiv \mathbb{M}^\phi.\tag{Eq. (2.48)}$$

It can be easily seen that Eq. (2.48) is a straightforward application of Eq. (2.35) where \mathbf{U}_{\log} is the unitary link between the fractional shift operator \mathbf{R}_p^ϕ and the scaled fractional shift operator \mathbf{M}_p^ϕ .

Consequently, the explicit formula of the FrMT can be given as (Akay & Boudreaux-Bartels, August 1998)

$$(\mathbb{M}^\phi s)(m) = \begin{cases} \sqrt{1 - j \cot \phi} e^{j\pi m^2 \cot \phi} \int_0^\infty s(t) e^{j\pi (\ln t)^2 \cot \phi - j2\pi (\ln t) m \csc \phi} \frac{dt}{\sqrt{t}}, & \phi \neq n\pi \\ e^{\frac{m}{2}} s(e^m), & \phi = (2n)\pi \\ e^{-\frac{m}{2}} s(e^{-m}), & \phi = (2n+1)\pi \end{cases}\tag{Eq. (2.49)}$$

where we recognize the eigenfunctions as

$$u_{\mathcal{M}^\phi}(t, m) = \frac{\sqrt{1 + j \cot \phi}}{\sqrt{t}} e^{-j2\pi \frac{(\ln t)^2 + m^2}{2} \cot \phi} e^{j2\pi (\ln t) m \csc \phi}, \quad \phi \neq n\pi.\tag{Eq. (2.50)}$$

For $\phi=0$, the FrMT in Eq. (2.49) reduces to the scale-covariant transform in Eq. (2.39), which can more explicitly be expressed as

$$(\mathbb{M}^0 s)(m) = e^{\frac{m}{2}} s(e^m).\tag{Eq. (2.51)}$$

This transform was termed as the scale covariant transform in (Baraniuk, 1993). When $\phi=\pi/2$, the FrMT reduces to the ordinary Mellin transform of Eq. (2.38)

$$(\mathbb{M}^{\pi/2}s)(m) = \int_0^{\infty} s(t) e^{-j2\pi(\ln t)m} \frac{dt}{\sqrt{t}}. \quad \text{Eq. (2.52)}$$

This transform was first defined by Cohen as the scale transform (Cohen, 1993) (Cohen, 1995). Similar to the way the FrFT can be decomposed into separate blocks, we can think of the FrMT as a series of operations on the original signal as shown in Figure 2.5.

The signal $s(t)$ is first multiplied by the scaled time chirp $e^{j\pi(\ln t)^2 \cot \phi}$. Then, the Mellin transform of the resulting signal is computed with the output variable scaled as $\frac{m}{\sin \phi}$. This operation provides the transition into the scaled time-frequency plane. Lastly, the signal is multiplied by a scaled fractional domain chirp $e^{j\pi m^2 \cot \phi}$ and an amplitude factor $\sqrt{1-j \cot \phi}$.

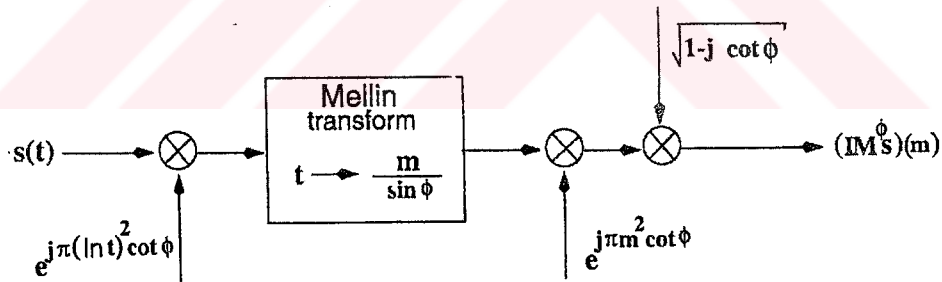


Figure 2. 5 Block diagram representation of FrMT

The FrMT given by $\mathbb{M}^\phi \equiv \mathbb{F}_{\mathcal{M}^\phi} = \mathbb{F}^\phi \mathbb{U}_{\log}$ constitutes the basis of our work. It will be the norm against which we test the correctness of the discrete FrMT algorithm that will be developed in the remainder of this thesis.

Fractional Mellin transform (FrMT) in Eq. (2.49) was proposed by Akay and Boudreaux-Bartels in (Akay & Boudreaux-Bartels, August 1998), (Akay, 2000).

However, to our knowledge, its discrete implementation has not been considered up to this point. In the rest of this thesis, using MATLAB computing software, we develop algorithms to implement the logarithmic warping, U_{\log} , combined with a previously proposed (Ozaktas et. al., 1996) discrete FrFT algorithm for the purpose of implementing the FrMT efficiently.



CHAPTER THREE

DEVELOPMENT OF THE LOGARITHMIC WARPING ALGORITHM

As we have seen in Section 2.4, the fractional Mellin transform (FrMT) corresponds to first applying the unitary warping operator \mathbf{U}_{\log} to the input signal and then computing the fractional Fourier transform, \mathbb{F}^ϕ , of the logarithmically warped input signal as

$$\mathbf{M}^\phi \equiv \mathbb{F}_{\mathcal{M}^\phi} = \mathbb{F}^\phi \mathbf{U}_{\log}. \quad \text{Eq. (3.1)}$$

It is therefore necessary to construct a discrete algorithm that performs the warping \mathbf{U}_{\log} on as large a set of input signals as possible. In this section, development of the logarithmic warping algorithm is described.

3.1 Logarithmic Warping

The scale transform given in Eq. (2.9) is a special case of the more general classical Mellin transform, which can be defined as (Cohen, 1993),

$$\mathcal{M}[f(t); s] \equiv F(s) = \int_0^\infty f(t) t^{s-1} dt \quad \text{Eq. (3.2)}$$

where $s \in \mathbb{C}$. It can be shown by a simple algebraic manipulation that the scale transform corresponds to the Mellin transform with the complex parameter s taken to be $s = -jc + 1/2$ (Cohen 1993). This can be seen via

$$\frac{e^{-jc \ln(t)}}{\sqrt{t}} = \frac{(e^{\ln(t)})^{-jc}}{t^{1/2}} = t^{-jc-1/2}. \quad \text{Eq. (3.3)}$$

To show the connection between the Mellin transform and the Fourier transform, one must first perform the following change of variables

$$t = e^{-x} \quad dt = -e^{-x} dx. \quad \text{Eq. (3.4)}$$

Substituting in Eq. (3.2) we find

$$F(s) = \int_{-\infty}^{\infty} f(e^{-x}) e^{-x(s-1)} e^{-x} dx = \int_{-\infty}^{\infty} f(e^{-x}) e^{-sx} dx = \mathcal{L}[f(e^{-x})]. \quad \text{Eq. (3.5)}$$

Thus, the Mellin transform is equivalent to calculating the two-sided Laplace transform of $f(e^{-x})$. Expressing the parameter s explicitly as $s = \sigma + j\omega$ helps us see the Fourier transform connection;

$$F(s) = \int_{-\infty}^{\infty} f(e^{-x}) e^{-(\sigma + j\omega)x} dx = \int_{-\infty}^{\infty} [f(e^{-x}) e^{-\sigma x}] e^{-j\omega x} dx = \mathcal{F}[f(e^{-x}) e^{-\sigma x}]. \quad \text{Eq. (3.6)}$$

In Eq. (3.6), it can be seen that the Mellin transform of $f(t)$ is equivalent to the Fourier transform of the function warped according to $f(e^{-x}) e^{-\sigma x}$. Comparing with Eq. (2.37) we see that these two formulations agree to a great degree (except for the negative signs of the exponents) if we assign $\sigma = \frac{1}{2}$.

Now that we see the warping requirement to be able to compute the fractional Mellin transform, we first must devise a discrete algorithm to perform the warping operation. In the following sections, development of the various versions of the warping algorithm will be discussed.

3.2 Warping as a Signal Processing Tool

As a simple example of the usage of warping, we simulate the recovery of a logarithmically modulated harmonic signal in a noisy environment. Let us first define the signal

$$s(t) = (U_{\log}^{-1} h)(t) + n(t) = \frac{-1}{\sqrt{t}} \sum_{k=1}^S \frac{(-1)^k}{k} \sin(2\pi k f_0 \ln(t)) + n(t) \quad \text{Eq. (3.7)}$$

where $h(t)$ is a finite Fourier series representation of a periodic sawtooth wave with $f_0=2$ Hz. and $n(t)$ is the additive white Gaussian noise component with an SNR of 10

dB. Only five components of the Fourier series representation of the sawtooth wave is used. The signals $h(t)$ and $s(t)$ are illustrated in Figures 3.1 and 3.2, respectively.

The finite Fourier series representation of the sawtooth wave $h(t)$ is given as

$$h(t) = -\sum_{k=1}^5 \frac{(-1)^k}{k} \sin(2\pi k f_0 t). \quad \text{Eq. (3.8)}$$

Our purpose is to recover the inverse logarithmically warped sawtooth wave $(U_{\log}^{-1}h)(t)$ by suppressing the noise as much as possible. Recovery of such a harmonic signal requires the application of a set of narrowband filters. To obtain the signal as accurately as possible, time-invariant filters would have to include the entire modulation bandwidth which is quite large due to the logarithmic modulation. Using larger passband selections would also allow passing of more noise components, thus making the attempt futile.

An alternative solution, more suitable for such signals, would be to first warp the time axis of the signal via the warping operation $(U_{\log}s)(t) \equiv e^{t/2}s(e^t)$. After warping, the signal $s(t)$ is once again harmonic with respect to time axis (Figure 3.3). Thus, we can now apply Fourier domain narrowband filtering efficiently (Figures 3.4 and 3.5). For the sake of simplicity, a simple comb filter whose passband regions are centered at the integer multiples the fundamental harmonic f_0 is applied by a simple masking operation in the Fourier domain. After filtering, the time axis is once again warped with the inverse logarithmic warping operator $(U_{\log}^{-1}s)(t) \equiv \frac{1}{\sqrt{t}}s(\ln(t))$ which takes the signal back to its original time domain (Figures 3.6 and 3.7).

The advantage warping gained us in this example is the ease of filtering of an otherwise wideband signal by mapping it to a domain where it is represented as a band limited signal. Thus, the noise component of the signal could be filtered, without resorting to complicated time-frequency methods.

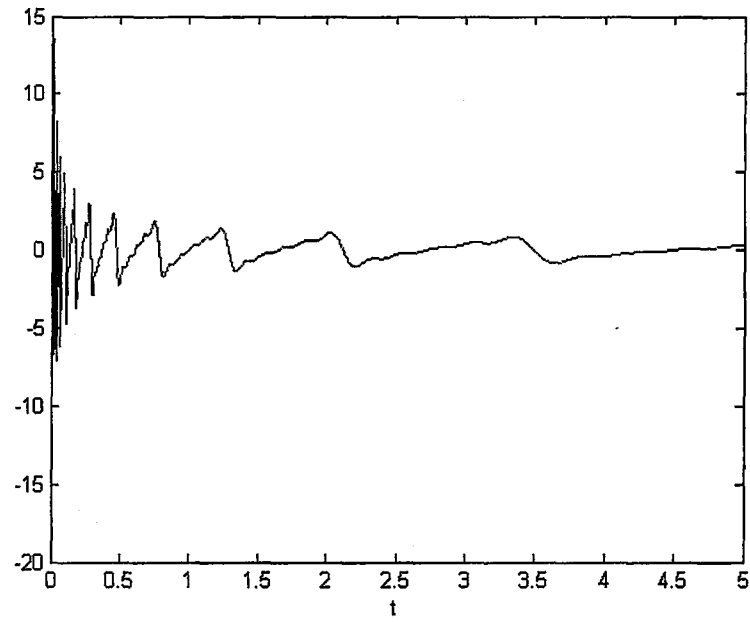


Figure 3. 1 Logarithmically modulated sawtooth wave via inverse logarithmic

warping, U_{\log}^{-1} ; $(U_{\log}^{-1}h)(t) = \frac{-1}{\sqrt{t}} \sum_{k=1}^5 \frac{(-1)^k}{k} \sin(2\pi k f_0 \ln(t))$

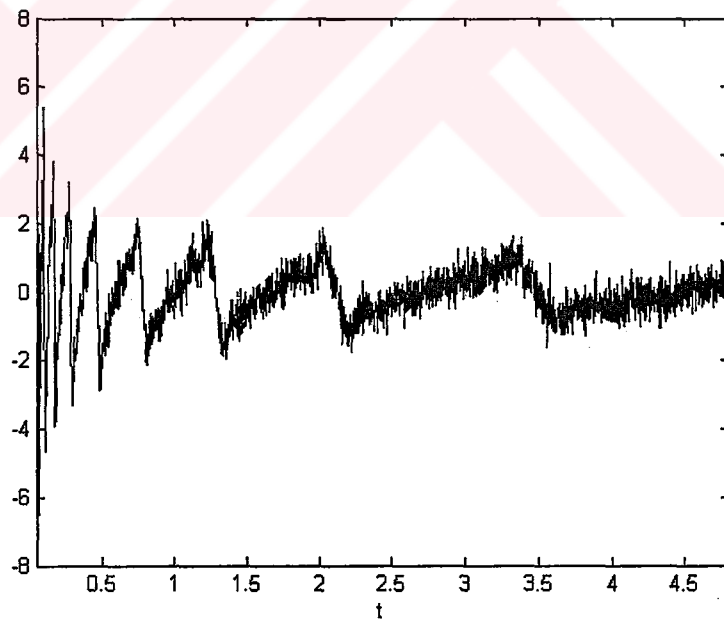


Figure 3. 2 Inverse logarithmically warped signal corrupted by noise,

$$s(t) = (U_{\log}^{-1}h)(t) + n(t)$$

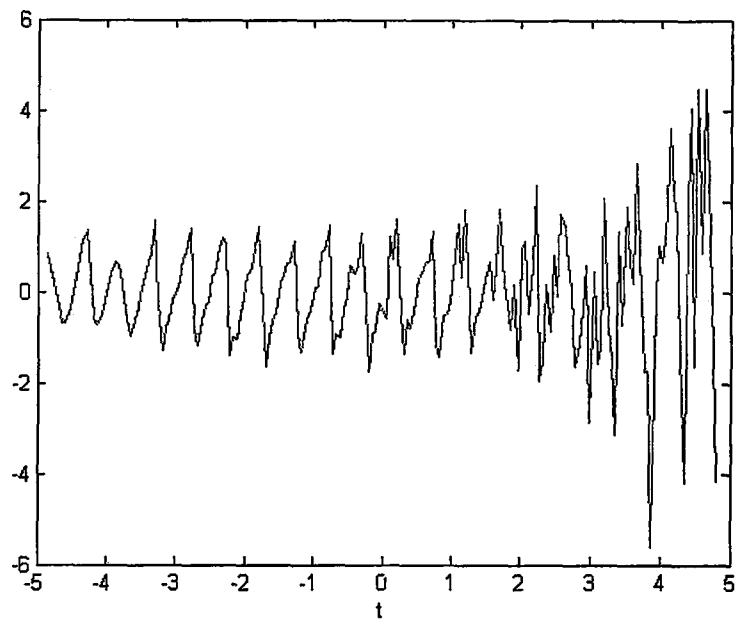


Figure 3. 3 The noisy sawtooth signal obtained after applying U_{\log} to the analyzed signal $s(t)$, $(U_{\log}s)(t) = h(t) + n'(t)$

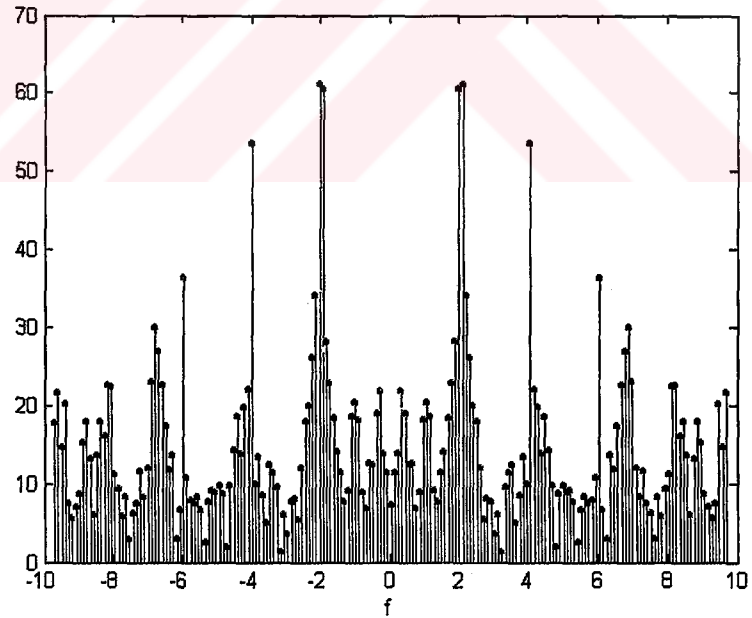


Figure 3. 4 Fourier transform of the logarithmically warped signal in Figure 3.3

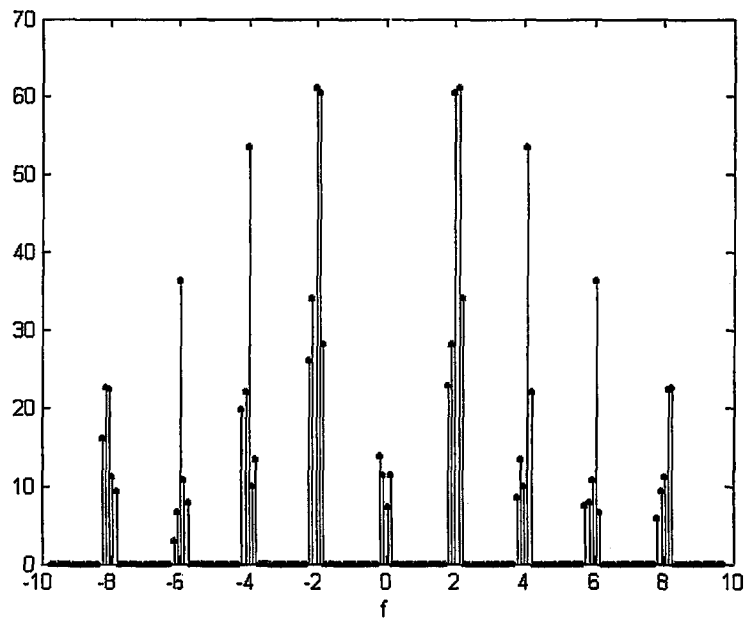


Figure 3. 5 The spectrum obtained after multiplication of the spectrum in Figure 3.4 with a comb filter mask.

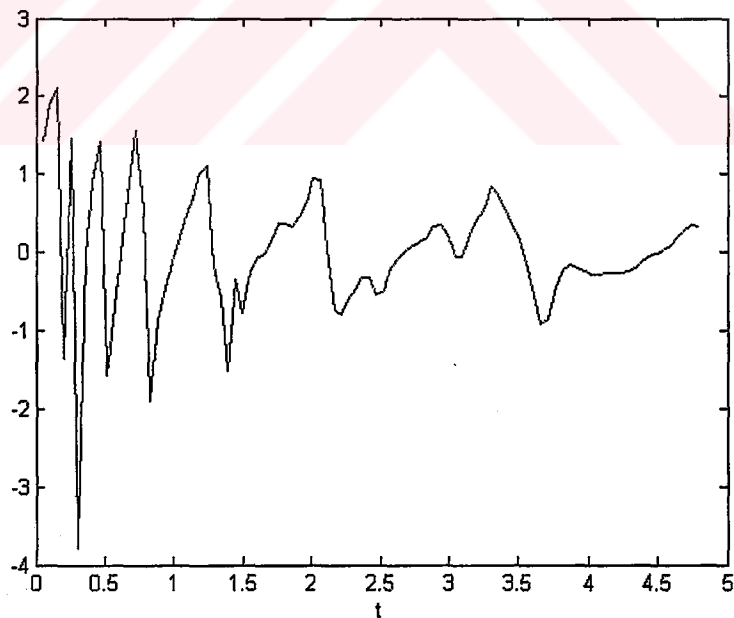


Figure 3. 6 The recovered signal as obtained through an inverse logarithmic warping operation of the Fourier domain masked signal having the spectrum in Figure 3.5.

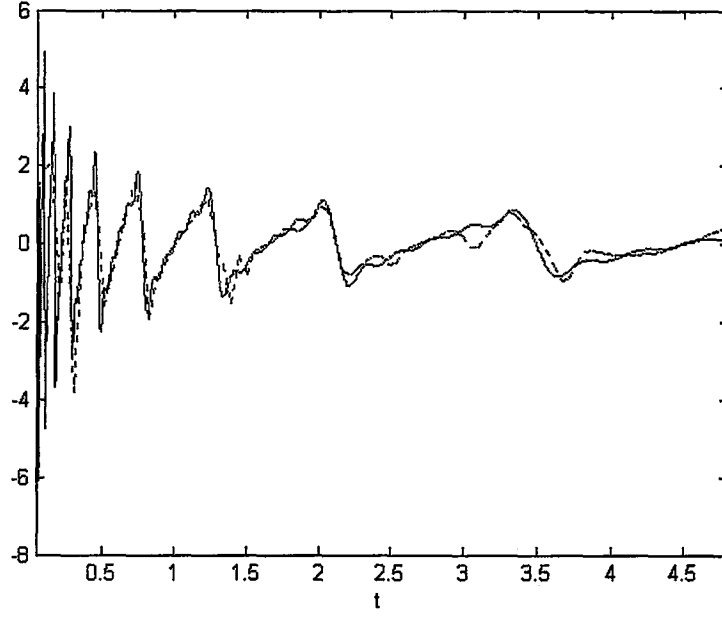


Figure 3. 7 Comparison of the recovered warped sawtooth wave (dashed line) with the original analyzed signal $(U_{\log}^{-1}h)(t)$ (solid line) in Figure 3.1.

3.3 Logarithmic Warping with A Priori Knowledge of Time

Logarithmic warping, as can be seen from the formulation,

$$(U_{\log} s)(t) \equiv e^{t/2} s(e^t). \quad \text{Eq. (3.9)}$$

is a time dependent process. Hence, to warp a signal logarithmically, one must know the time instances as well as the values the signal assumes at those instances. The first version of the logarithmic warping script as given in Appendix A.1 presumes that the initial and final time instants are known. Using this script several test functions were warped. To be able to verify our results, we must also know the inverse warping operator, U_{\log}^{-1} , which is given as

$$(U_{\log}^{-1} s)(t) = \frac{1}{\sqrt{t}} s(\ln t). \quad \text{Eq. (3.10)}$$

We confirm the proper operation of the script by applying our warping algorithm to the inverse warped signals to recover the unwarped original signals back. In our

simulations, we used sinusoidal and linear FM (chirp) signals. The sinusoidal signal $s(t)$ can be defined as

$$s(t) = \sin(2\pi f_0 t). \quad \text{Eq. (3.11)}$$

If we apply the inverse logarithmic warping in Eq. (3.10), we obtain

$$(U_{\log}^{-1}s)(t) = \frac{1}{\sqrt{t}} \sin(2\pi f_0 (\ln t)). \quad \text{Eq. (3.12)}$$

Similarly, the real-valued chirp signal and its inverse logarithmic warped versions are given as

$$\begin{aligned} s(t) &= \sin(\pi m_0 t^2) \\ (U_{\log}^{-1}s)(t) &= \frac{1}{\sqrt{t}} \sin(\pi m_0 (\ln t)^2). \end{aligned} \quad \text{Eq. (3.13)}$$

The script given in Appendix A.2 is used to test the logarithmic warping function *Ulog.m* in Appendix A.1. The results are given in Figures 3.8 and 3.9 below. Figure 3.8 shows the inverse logarithmically warped sine given in Eq. (3.12) and its warped version which corresponds to the sine function of frequency $f_0=0.5$ Hz. Similarly, Figure 3.9 shows the inverse logarithmically warped chirp signal given in Eq. (3.13) and its warped version which is a real-valued chirp signal with a chirp rate parameter of $m_0=2$. In both figures it can be observed that the signals begin with rather high frequencies and end with relatively lower frequencies. This is due to the fact that, via the inverse warping U_{\log}^{-1} , the independent variable t is replaced with $\ln(t)$. It is illustrated in Figure 3.10 that the natural logarithm function starts with bigger rates of change at $t=0^+$ and continues with slower rates of change towards $t \rightarrow \infty$.

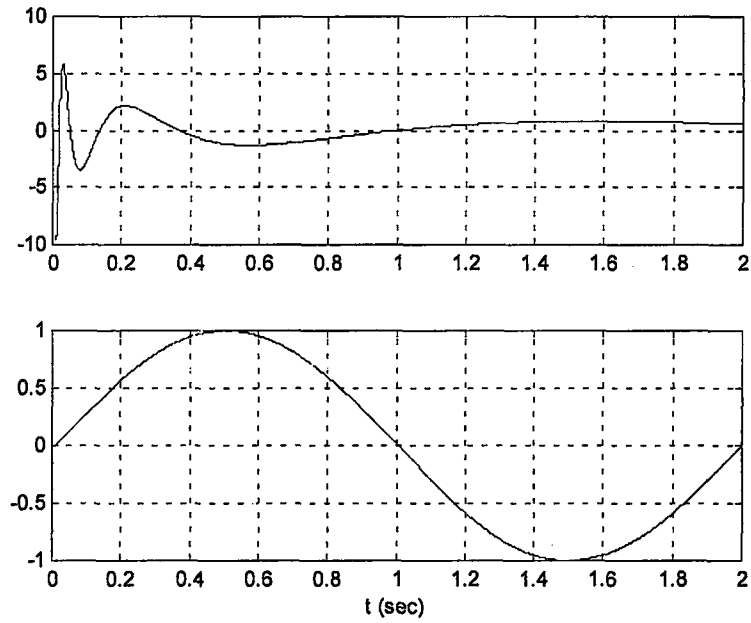


Figure 3. 8 Inverse logarithmic warping of a sine signal $\sin(2\pi f_0 t)$ (top), and the logarithmic warp of $\frac{1}{\sqrt{t}} \sin(2\pi f_0 (\ln t))$ (bottom)

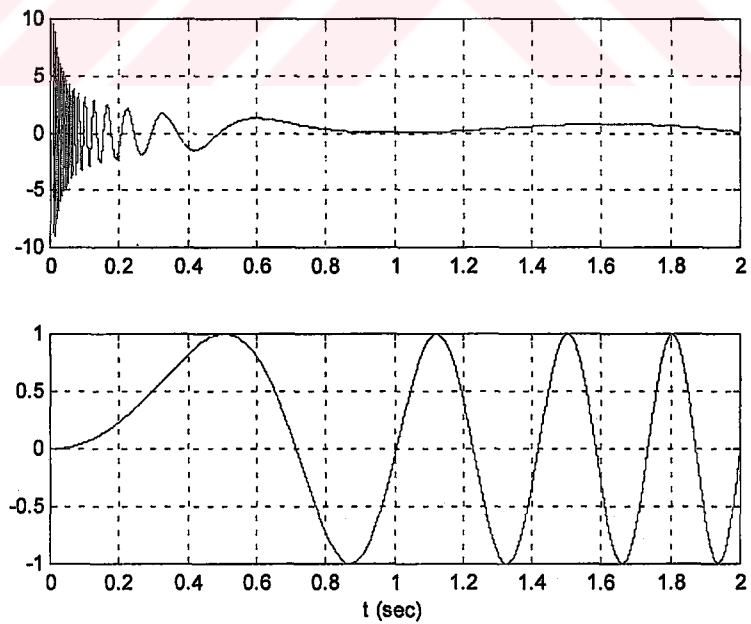


Figure 3. 9 Inverse logarithmic warping of a real-valued chirp signal $\sin(\pi m_0 t^2)$ (top), and the logarithmic warp of $\frac{1}{\sqrt{t}} \sin(\pi m_0 (\ln t)^2)$ (bottom)

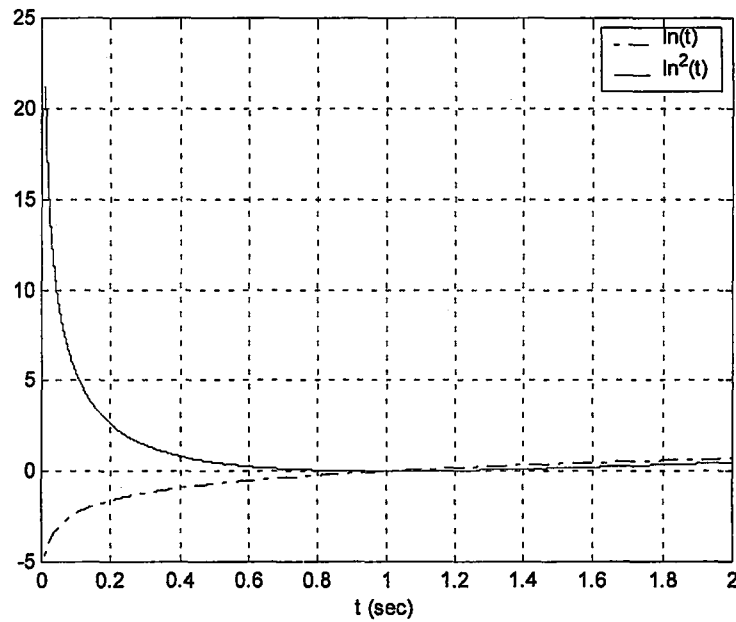


Figure 3. 10 Graph illustrating $\ln(t)$ vs. $\ln^2(t)$

It is really straightforward to work with warping if one knows the correct timing information of the signal. However, in some applications such information might not exist and a way must be devised to be able to implement the warping algorithm without the timing information.

3.4 Logarithmic Warping without A Priori Knowledge of Time

To overcome the time dependence problem of the logarithmic warping, we suggest to use interpolation. Our idea of utilizing interpolation in the implementation of the logarithmic warping was inspired by a paper by Canfield and Jones (Canfield & Jones, 1993). A similar application was also found on (Zalubas & Williams, 1995) and the simulation examples we perform closely follow the examples in their paper.

The full source code of the first version of this script, *logwarp.m* is given in Appendix A.3. The test files used to test this script are also given in Appendices A.4 and A.5.

Our simulation results can be seen in Figures 3.11 and 3.12. In Figure 3.11, a single period of a sine wave is shown with the dashed line. The solid line shows the theoretically warped version of the same sine wave. The bold solid line shows the warping of the sine using the algorithm in Appendix A.3. The same experiment is repeated for a single period of a square wave in Figure 3.12.

In this version of the logarithmic warping algorithm, we assume that only the initial time, t_0 , and the sampling frequency, f_s , of the signal are known. If not, their default values are taken to be 0 seconds and 8192 Hz. Using these values, the final time instant, t_f , is calculated and a time index vector t is formed. Then, another time index vector t_{os} is constructed with 10 times the sampling rate of the original signal. This is required to increase the resolution during interpolation. Interpolation of the input vector x is accomplished over the elements of the oversampled time index vector t_{os} to form the interpolated signal vector x_{int} . For each time instant $t(n)$ within vector t , the point that is nearest to $\exp(t(n))$ is found in t_{os} and assigned to $t_{nearest}$. The value in x_{int} that corresponds to $t_{nearest}$ is $x(\exp(t(n)))$ and is assigned to $x_{nearest}$. Finally, warping of the instant $t(n)$ is concluded by multiplying with the constant $\exp(t(n)/2)$.

The inherent difficulty in the implementation of any logarithmic warping algorithm is the unavoidable loss of data. Even if we know all the time instances for which the elements of a given input vector x is calculated, we cannot produce all the elements of the warped signal, since we only know the original signal between the time instances t_0 and t_f . However, for warping one needs the values of x at exponentiated time instants $\exp(t(n))$. Since we are working in a limited time window, sooner or later $\exp(t(n))$ will be exceeded. Thus, we can warp only until

$$\begin{aligned} e^{t(n)} &= t_f \\ t(n) &= \ln(t_f). \end{aligned} \tag{Eq. (3.14)}$$

This is the reason why the bold lines formed by the algorithm do not trace the whole length of the theoretically warped signals in Figures 3.11 and 3.12, but go only

as far as $\ln(2)=0.69$ seconds. The number of samples N in the original signal is found by

$$N = (t_f - t_0) f_s. \quad \text{Eq. (3.15)}$$

Substituting $\ln(t_f)$ for t_f in Eq. (3.15) gives us the number of samples obtained after the logarithmic warping

$$N' = (\ln(t_f) - t_0) f_s. \quad \text{Eq. (3.16)}$$

Using Eq. (3.15) and Eq. (3.16) we can calculate the percentage of samples that can be retained after the logarithmic warping through the ratio

$$\frac{N'}{N} = \frac{(\ln(t_f) - t_0)}{(t_f - t_0)}. \quad \text{Eq. (3.17)}$$

Substituting $t_0=-2$ and $t_f=2$ in Eq. (3.17) shows that for the simulations in Figures 3.11 and 3.12 only 67% of the samples could be retained. For a signal with symmetric time support $[-T/2 \ T/2]$ around the origin, Eq. (3.17) takes the form

$$\frac{N'}{N} = \frac{(\ln(\frac{T}{2}) - (-\frac{T}{2}))}{T} = \frac{(\ln(\frac{T}{2}) + \frac{T}{2})}{T} \quad \text{Eq. (3.18)}$$

which approaches 0.5 as T is let to approach infinity. In other words, as longer duration signals are taken, only half of the samples will be retained after performing the logarithmic warping with this algorithm.

It should be noted that the cancellation of the sampling frequency f_s in the numerator and the denominator of Eq. (3.17) does not make the expression independent of f_s since the very value of t_f is calculated from Eq. (3.15) as

$$t_f = t_0 + \frac{N}{f_s}. \quad \text{Eq. (3.19)}$$

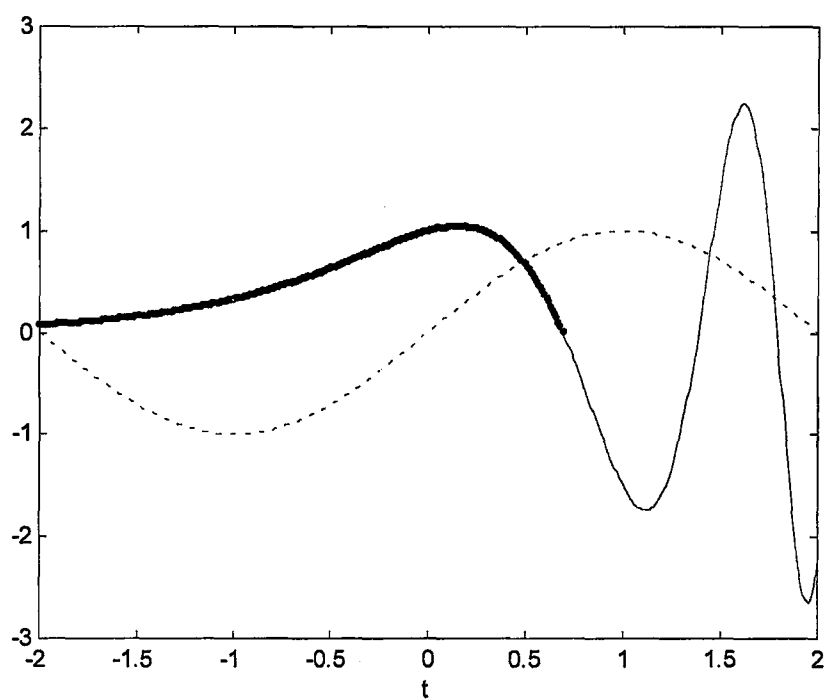


Figure 3. 11 Logarithmic warping of $\sin(2\pi f_0 t)$ using *logwarp.m* ($f_0=0.25$ Hz)

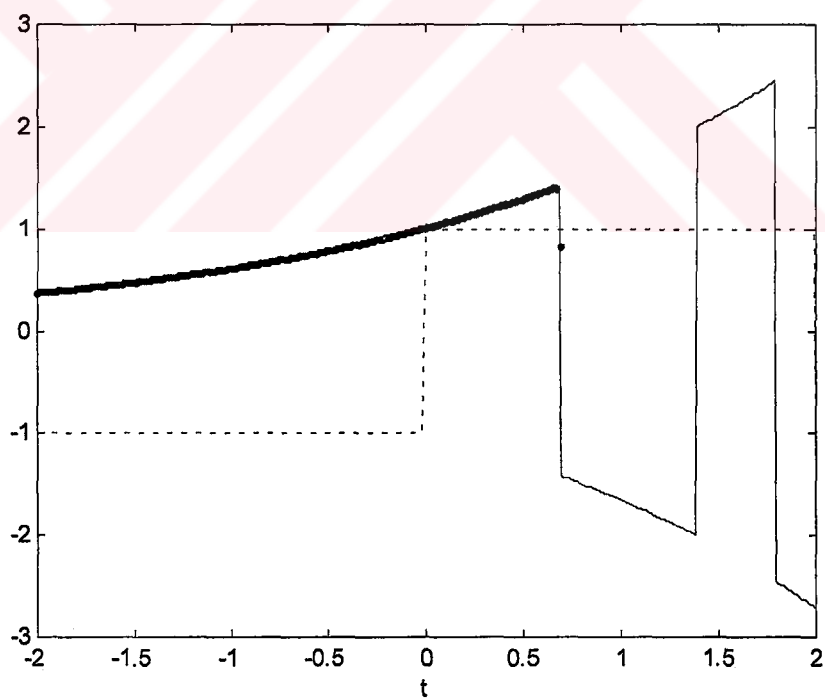


Figure 3. 12 Logarithmic warping of a square wave of frequency 0.25 Hz using *logwarp.m*

3.5 Further Complications and Modifications on the Algorithm

The logarithmic warping algorithm in Appendix A.3 works well by itself. However, at first try, it did not perform adequately when used together with the discrete fractional Fourier transform (FrFT) algorithm proposed by Ozaktas and his colleagues (Ozaktas et al., 1996). Recall that since our final aim is to implement the FrMT efficiently, we should be able to combine our warping algorithm with the discrete FrFT algorithm. Thus, another version of the logarithmic warping algorithm, named *logwarp3.m*, that interpolates over a symmetric portion of the middle of the signal, had to be written. That new algorithm obeys certain constraints forced by the FrFT algorithm. These constraints are discussed later in Section 4.2.2.

The results produced by the warping algorithm *logwarp3* can be observed in Figures 3.13 and 3.14. In both figures, the dashed lines correspond to the original unwarped signals of the sine and the square wave. The solid lines represent the theoretically warped signals found analytically. The heavy dots are the results of computation using the algorithm *logwarp3*.

Contrasting with Figures 3.11 and 3.12, we can see that the output of *logwarp3* produces far less number of samples than *logwarp*. While *logwarp* generated 270 samples from an input signal of size 400, *logwarp3* produced only 14 samples from an input of size 128. The percentage of samples retained after the third logarithmic warping algorithm can be found by

$$\frac{N'}{N} = \frac{4 \ln^2(T/2)}{T^2}. \quad \text{Eq. (3.20)}$$

As wider signal supports are taken around the origin, the number of samples produced by the *logwarp3* algorithm is given by $8 \ln^2(T/2)$ (See Section 4.2.2 for details). However, the ratio in Eq. (3.20) approaches 0 as T is let to approach infinity.

It can be observed from Figures 3.13 and 3.14 that the generated samples follow the theoretical values closely. By employing inputs of larger sizes, more samples of the logarithmically warped signals can be calculated.

In the next chapter, we will combine our logarithmic warping algorithms with the discrete FrFT code in order to calculate the FrMT digitally.



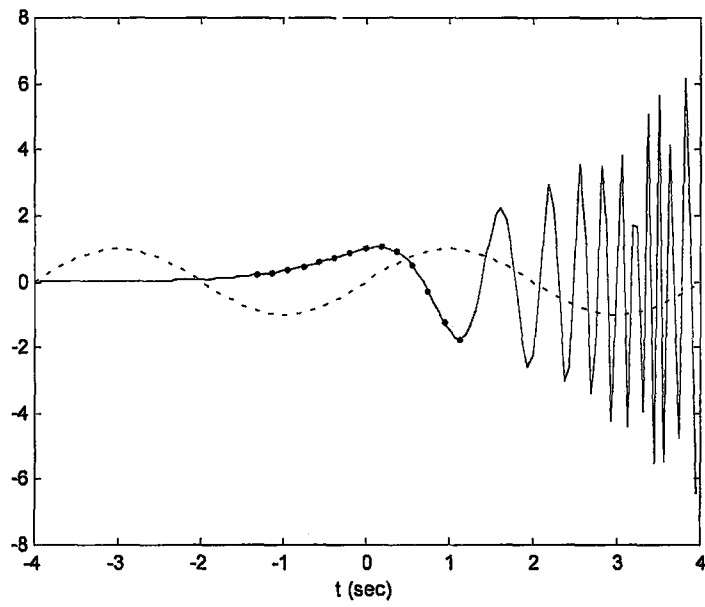


Figure 3. 13 Logarithmic warping of $\sin(2\pi f_0 t)$ using *logwarp3.m* ($f_0=0.25$ Hz)

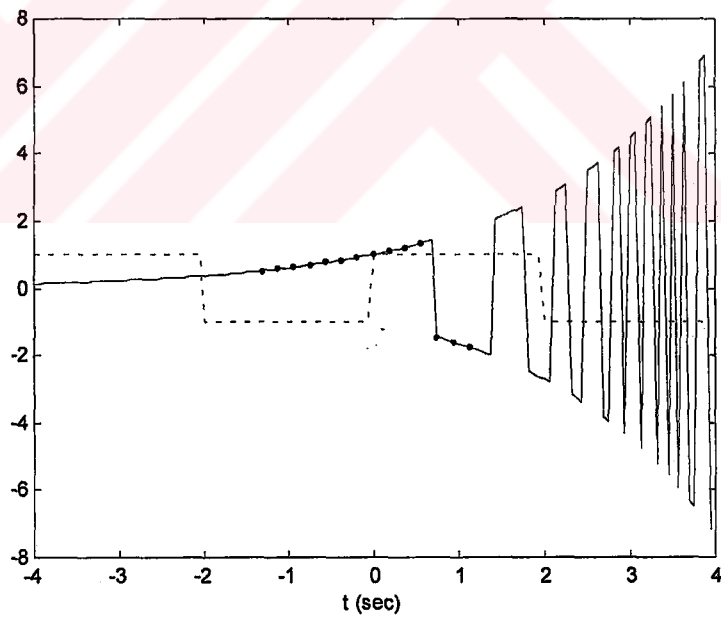


Figure 3. 14 Logarithmic warping of a square wave of 0.25 Hz using *logwarp3.m*

CHAPTER FOUR

DIGITAL COMPUTATION OF THE FRACTIONAL MELLIN TRANSFORM

In this chapter, the logarithmic warping algorithms developed in the previous chapter are combined with the discrete fractional Fourier transform code to compute the fractional Mellin transform. The results of different warping schemes are examined in the following sections.

4.1 The Fractional Mellin Transform with A Priori Knowledge of Time

The first warping algorithm used was *Ulog.m* given in Appendix A.1. Using this algorithm together with the fractional Fourier transform code given in Appendix A.7 produced the results in Figures 4.1 through 4.20. The script file for obtaining these figures is given in Appendix A.12.

Figures 4.1 through 4.5 give the results of the fractional Mellin transformation for the inverse logarithmically warped complex exponential of frequency $f_0=1.5$ Hz. The real part of the original exponential signal is plotted in Figure 4.1. The inverse logarithmic warping of this signal is found as

$$U_{\log}^{-1}(e^{j2\pi f_0 t}) = \frac{1}{\sqrt{t}} e^{j2\pi f_0 (\log t)} = \frac{1}{\sqrt{t}} (e^{(\log t)})^{j2\pi f_0} = t^{j2\pi f_0} t^{-1/2} = t^{j2\pi f_0 - 1/2} \quad \text{Eq. (4.1)}$$

and is plotted in Figure 4.2. The FrFT at angle $\phi=\pi/5$ of the complex exponential can be seen in Figure 4.3. Figure 4.4 shows the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ computed at $\phi=\pi/5$ and superimposed on the FrFT of the complex exponential. The data points indicated by the circles are the result of the logarithmic warping with *Ulog.m* followed by the

computation of the FrFT. The two results match by virtue of Eq. (2.46). That is, if we apply the FrMT to the inverse logarithmic warped version of a signal $(U_{\log}^{-1}s)(t)$, we obtain the FrFT of the original signal, $s(t)$. This is how we check the correctness of the discrete FrMT algorithm. This validation procedure can be represented by the following expression

$$(\mathbb{M}^\phi s)(t) = (\mathbb{F}^\phi U_{\log} s)(t) \quad \text{Eq. (4.2)}$$

$$(\mathbb{M}^\phi U_{\log}^{-1} s)(t) = (\mathbb{F}^\phi U_{\log} U_{\log}^{-1} s)(t) = (\mathbb{F}^\phi s)(t). \quad \text{Eq. (4.3)}$$

Figure 4.5 shows the FrMT at angle $\phi=\pi/5$ of $U_{\log}^{-1}(e^{j2\pi f_0 t})$, superimposed on the theoretical FrFT of $e^{j2\pi f_0 t}$. This time the results do not match perfectly, especially near the ends. This situation can be explained by considering the theoretical FrFT of the complex exponential signal given as (Akay, 2000)

$$\mathbb{F}^\phi(e^{j2\pi f_0 t}) = \sqrt{1 + j \tan \phi} e^{-j\pi(r^2 + f_0^2) \tan \phi} e^{j2\pi r f_0 \sec \phi}. \quad \text{Eq. (4.4)}$$

This signal cannot be perfectly created by the FrFT algorithm of Appendix A.7. As stated in (Ozaktas et.al, 1996), the FrFT algorithm is not an exact computation, but an approximation of the continuous formulation of the FrFT. Therefore, we expect the practical computations not to comply with the theoretical computations exactly.

Figures 4.6 through 4.10 contain the results of computing the FrMT of an inverse logarithmically warped complex chirp signal. Figure 4.6 shows the real part of the chirp signal with chirp rate $m_0=0.05$. In Figure 4.7, the inverse logarithmic warping of the chirp signal, given by,

$$U_{\log}^{-1}(e^{j\pi m_0 t^2}) = \frac{1}{\sqrt{t}} e^{j\pi m_0 (\log t)^2}, \quad \text{Eq. (4.5)}$$

is illustrated. Figure 4.8 shows the FrFT at the “matching” angle, ($\phi=\text{atan}(m_0)+\pi/2=1.6208$ rad=92.86 degrees), of the chirp signal. In Figure 4.9, the FrFT of the chirp $e^{j\pi m_0 t^2}$ is superimposed with the FrMT at the matching angle of the warped chirp signal $U_{\log}^{-1}(e^{j\pi m_0 t^2})$. The circles represent the data points produced by

the FrMT. The theoretical FrFT at angle ϕ of a chirp signal is another chirp given by (Akay, 2000)

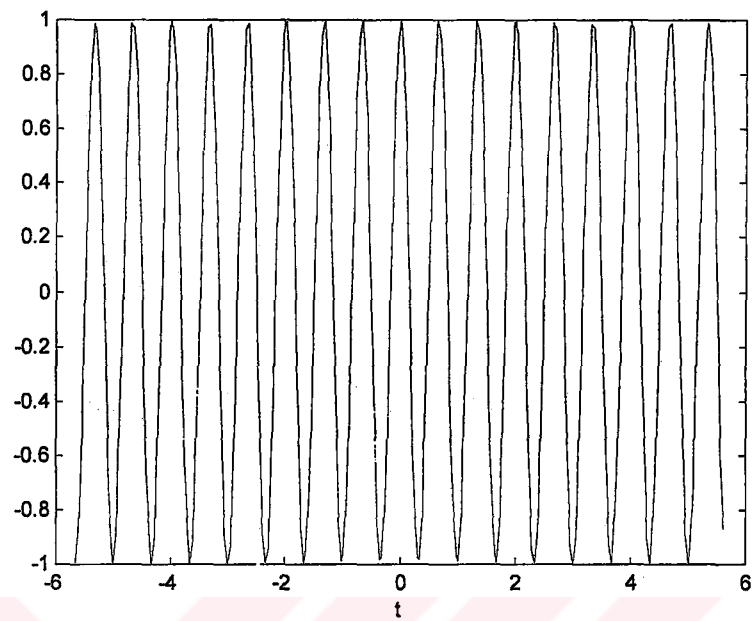


Figure 4. 1 Real part of the complex exponential $e^{j2\pi f_0 t}$ with $f_0=1.5\text{Hz}$

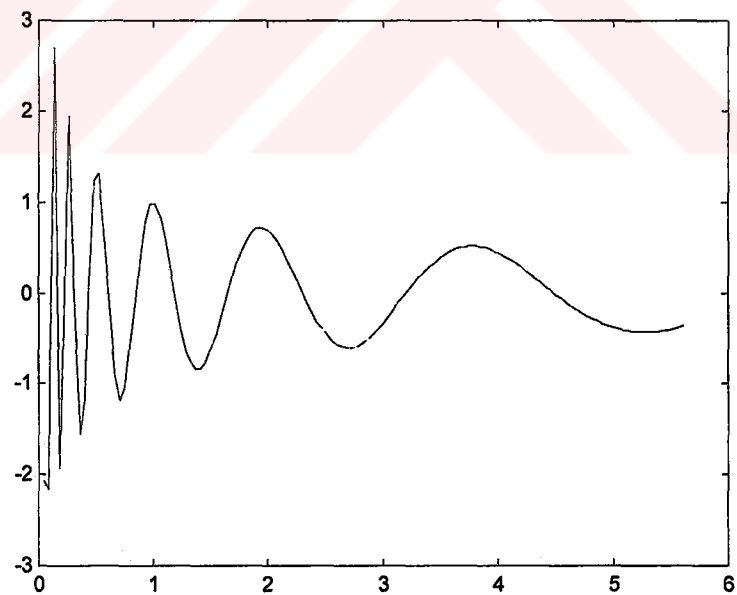


Figure 4. 2 Real part of the inverse logarithmic warping of $e^{j2\pi f_0 t}$

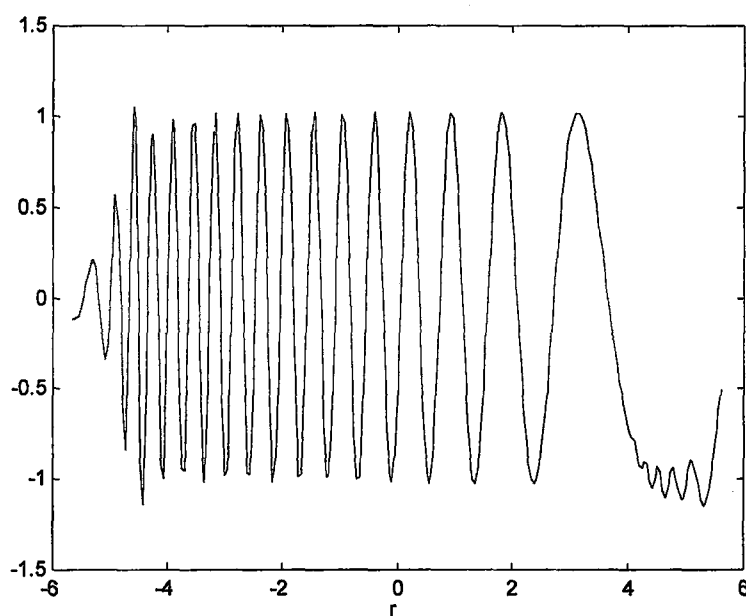


Figure 4. 3 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{j2\pi f_0 t}$

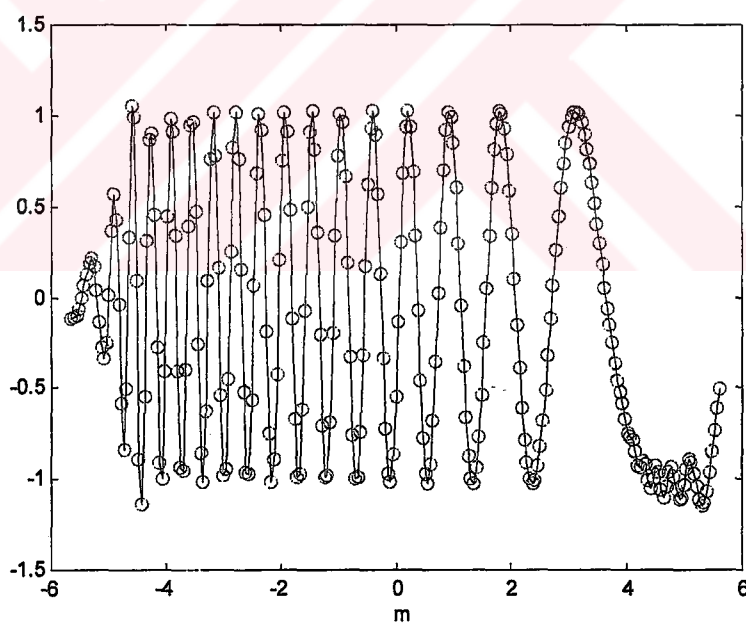


Figure 4. 4 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{j2\pi f_0 t})$

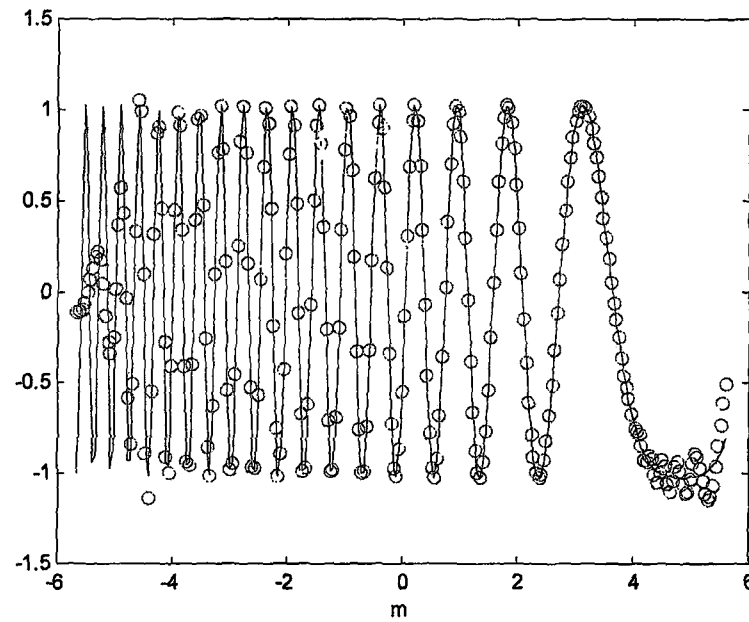


Figure 4. 5 Comparison of the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ and the theoretical FrFT of $e^{j2\pi f_0 t}$ at $\phi=\pi/5$

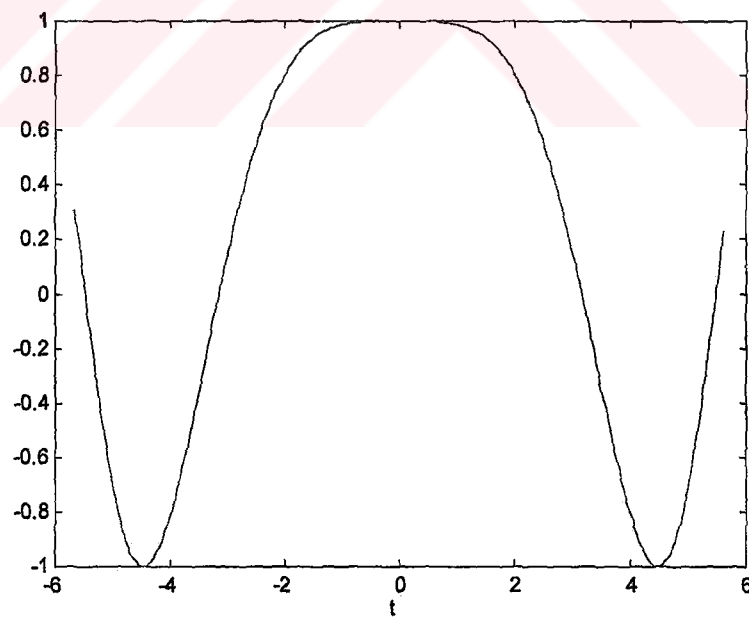


Figure 4. 6 Real part of the chirp signal $e^{j\pi m_0 t^2}$ with $m_0=0.05$

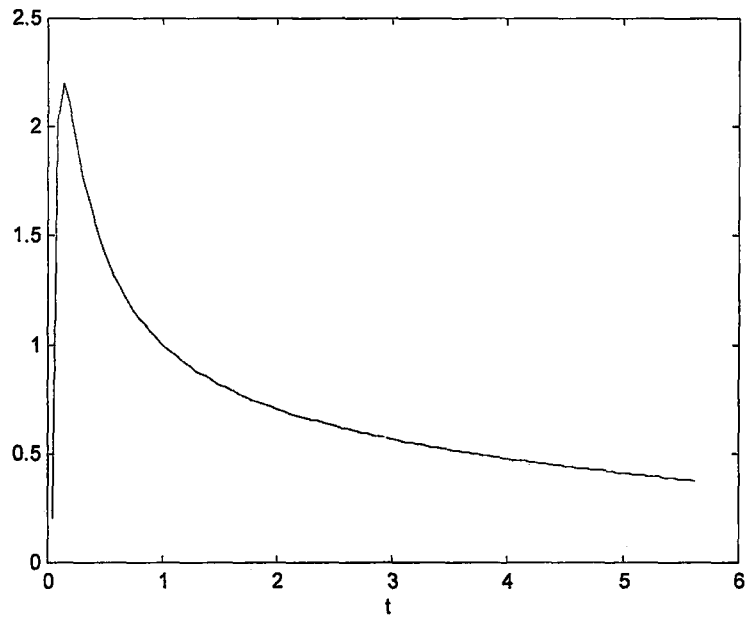


Figure 4. 7 Real part of the inverse logarithmic warping of $e^{j\pi m_0 t^2}$

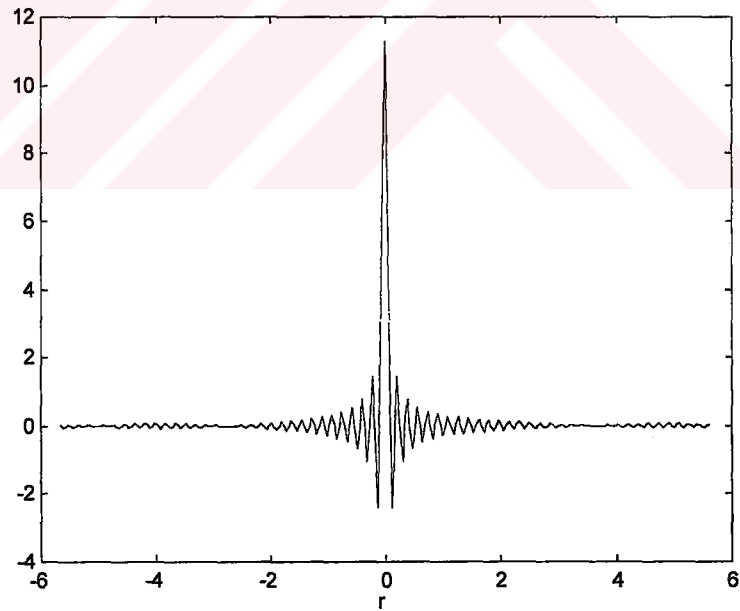


Figure 4. 8 Absolute value of the FrFT of $e^{j\pi m_0 t^2}$ computed at the matching angle $\phi = \text{atan}(m_0) + \pi/2$

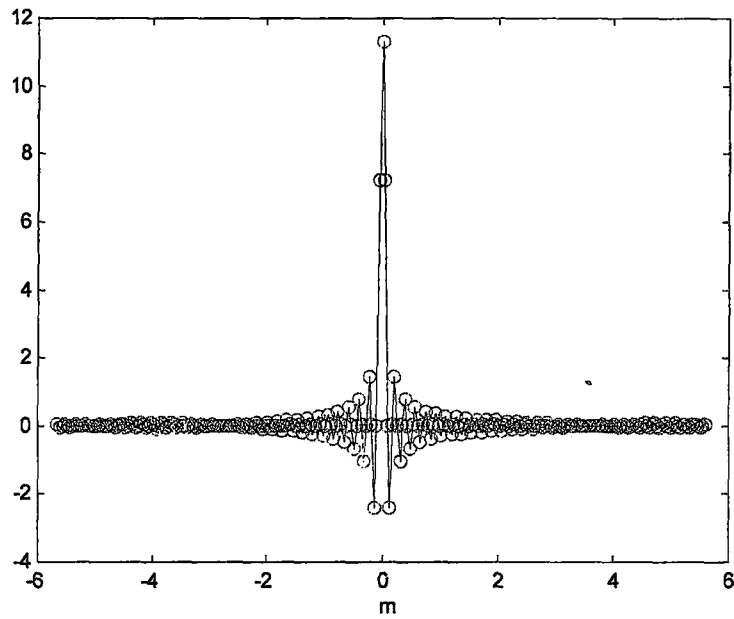


Figure 4. 9 Absolute value of the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ computed at the matching angle $\phi = \text{atan}(m_0) + \pi/2$

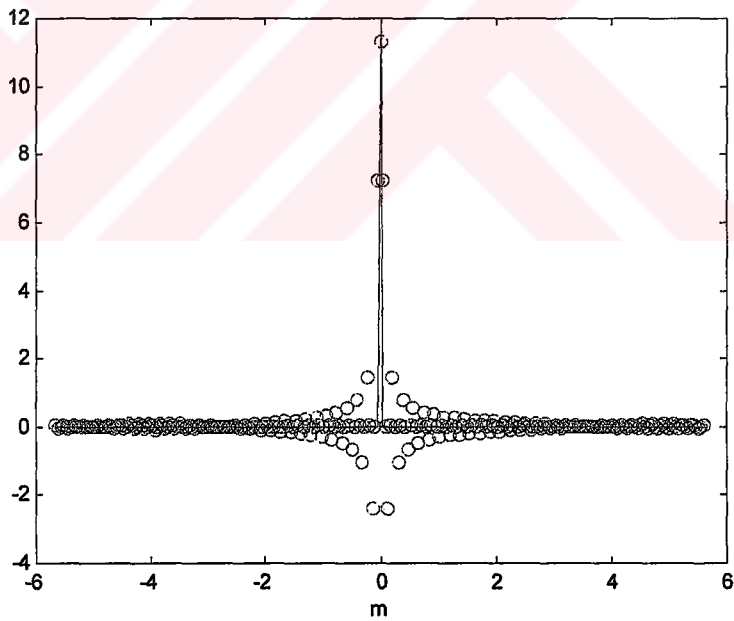


Figure 4. 10 Comparison of the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ and the theoretical FrFT of $e^{j\pi m_0 t^2}$

$$\mathbb{F}^\phi(e^{j\pi m_0 t^2}) = \sqrt{\frac{1+j\tan\phi}{1+m_0\tan\phi}} e^{-j\pi r^2 \frac{m_0-\tan\phi}{1+m_0\tan\phi}}. \quad \text{Eq. (4.6)}$$

When a chirp signal with a matching chirp rate parameter value of $m_0 = \tan(\phi - \pi/2)$ is transformed with an FrFT at angle ϕ , the chirp rate of the transformed signal given in Eq. (4.6) approaches infinity. The FrFT code of Appendix A.7 (Ozaktas et.al., 1996) implements the chirp multiplication and FT decomposition of the FrFT (displayed in Figure 2.3) as follows

$$S\left(\frac{m}{2\Delta r}\right) = \frac{\sqrt{1-j\cot\phi}}{2\Delta r} e^{j\pi(\cot\phi-\csc\phi)\left(\frac{m}{2\Delta r}\right)^2} \sum_{n=-N}^{N-1} e^{j\pi\csc\phi\left(\frac{m-n}{2\Delta r}\right)^2} e^{j\pi(\cot\phi-\csc\phi)\left(\frac{n}{2\Delta r}\right)^2} s\left(\frac{n}{2\Delta r}\right). \quad \text{Eq. (4.7)}$$

This formula is an approximation of the FrFT integral in Eq. (2.42). Note the $1/2\Delta r$ terms in the denominator of the coefficient term. When the matching chirp pulls the chirp rate of Eq.(4.6) to infinity, the $1/2\Delta r$ terms in Eq.(4.7) pull the value of the overall expression down as in

$$\lim_{\varepsilon \rightarrow 0} \frac{\sin\left(\frac{x}{\varepsilon}\right)}{\pi x} \rightarrow \delta(x) \quad \text{Eq. (4.8)}$$

and produce an impulse signal. This is expected if we think of the impulse function as a special degenerate chirp function with infinite slope.

In Figures 4.11 through 4.15, the FrMT computed at angle $\phi = \pi/5$ of an inverse logarithmically warped Gaussian signal of the form $e^{-\pi t^2}$ is illustrated. Figures 4.11 and 4.12 depict the Gaussian signal and its inverse logarithmic warped versions, respectively. Figures 4.13 and 4.14 show the FrFT at $\phi = \pi/5$ of the Gaussian signal and the FrMT at $\phi = \pi/5$ of the inverse logarithmic warped Gaussian given by;

$$U_{\log}^{-1}(e^{-\pi t^2}) = \frac{1}{\sqrt{t}} e^{-\pi(\log t)^2}. \quad \text{Eq. (4.9)}$$

This time the practical and theoretical results match perfectly because the Gaussian signal is an eigenfunction of the FrFT operator \mathbb{F}^ϕ defined in Eq. (2.44) (Akay, 2000).

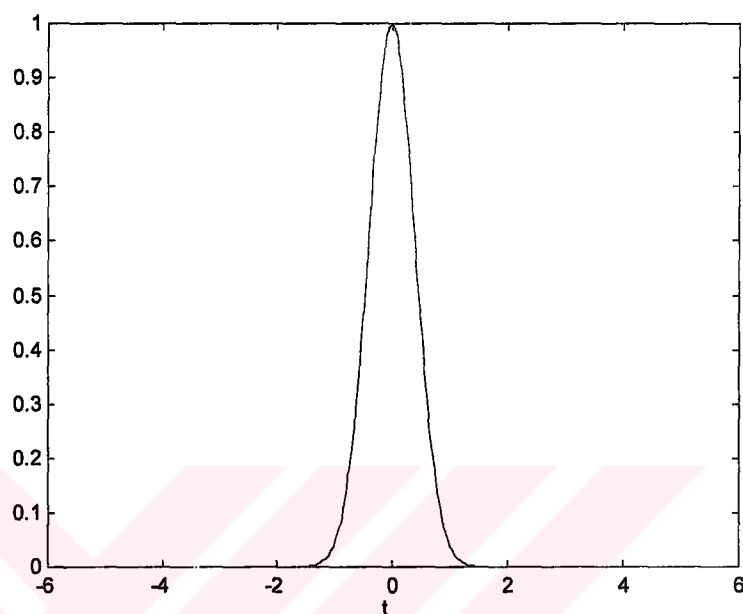


Figure 4. 11 The Gaussian signal $e^{-\pi t^2}$

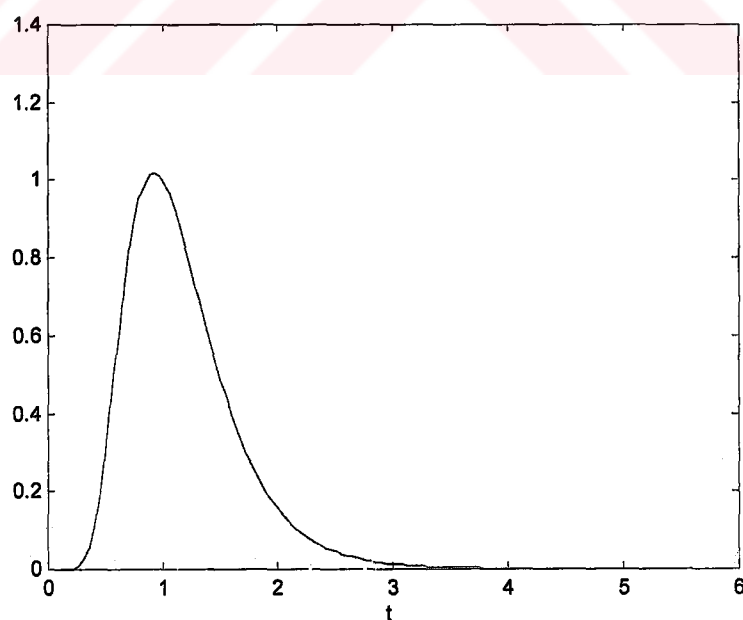


Figure 4. 12 Real part of the inverse logarithmic warping of $e^{-\pi t^2}$

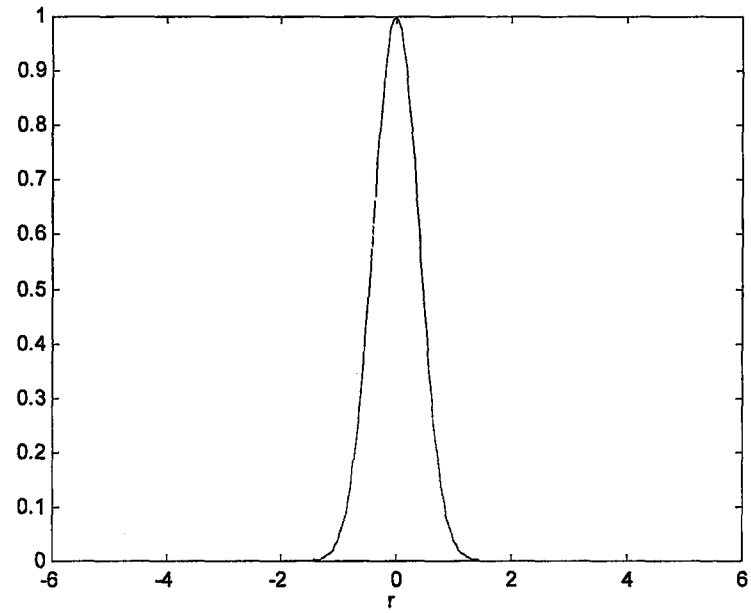


Figure 4.13 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{-\pi t^2}$

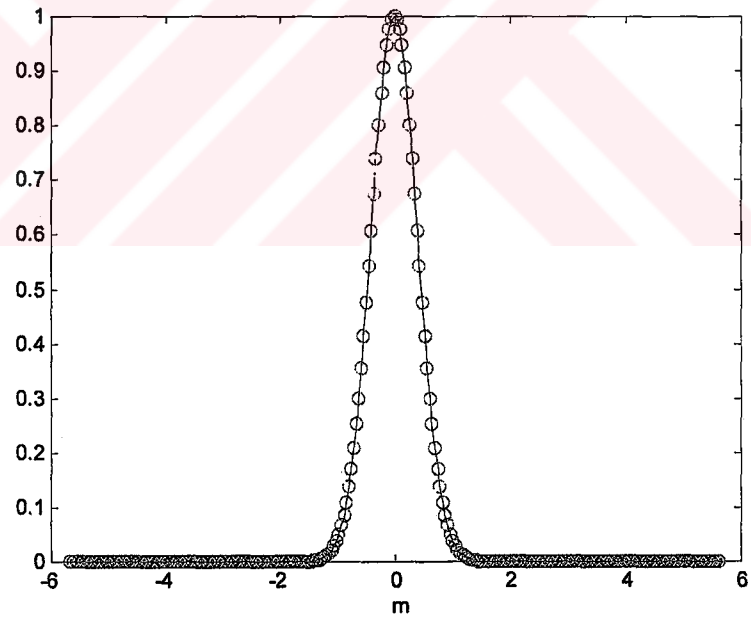


Figure 4.14 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{-\pi t^2})$

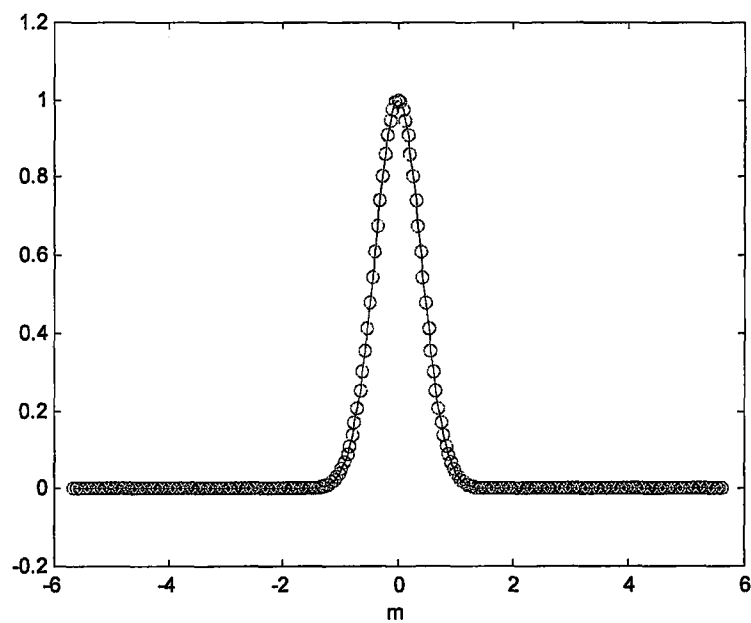


Figure 4. 15 Comparison of the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ and the theoretical FrFT of $e^{-\pi t^2}$

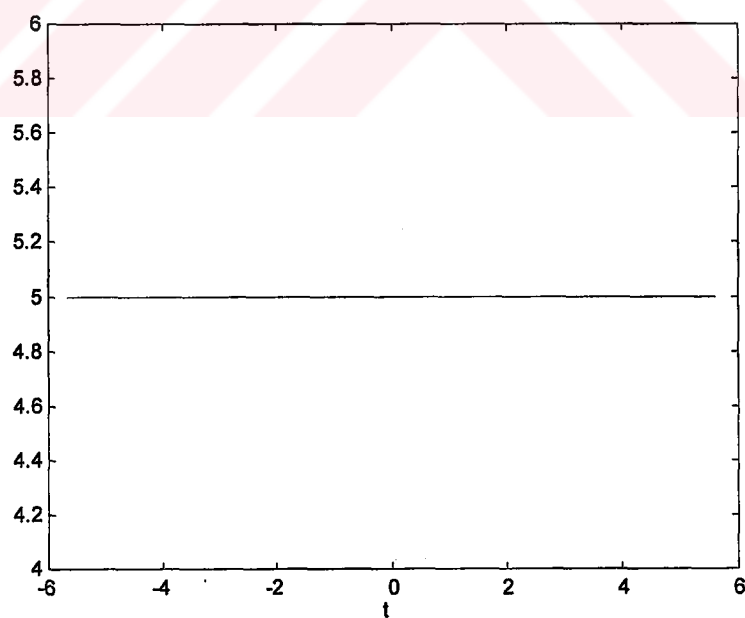


Figure 4. 16 The constant signal $c=5$

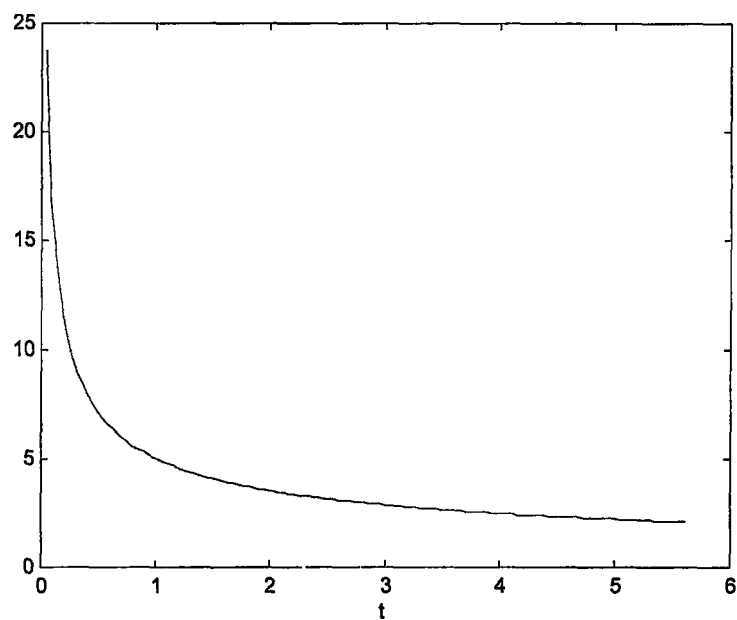


Figure 4. 17 Real part of the inverse logarithmic warping of $c=5$

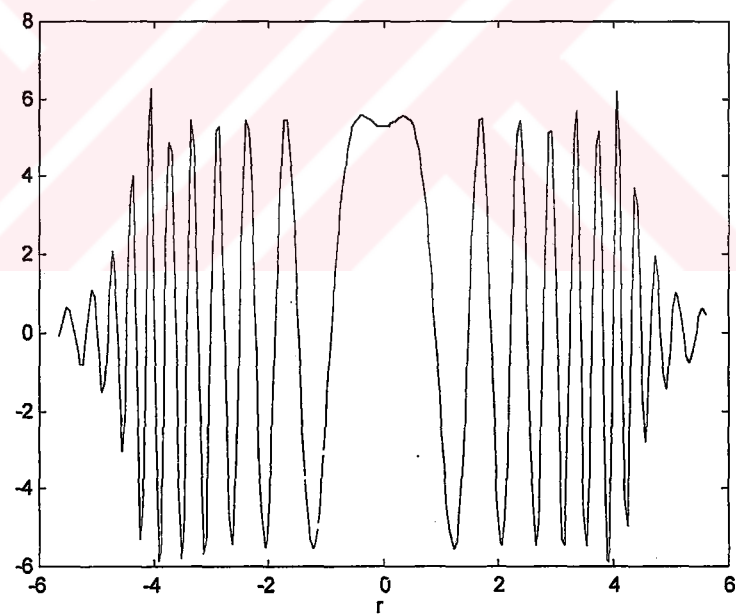


Figure 4. 18 Absolute value of the FrFT at $\phi=\pi/5$ of $c=5$

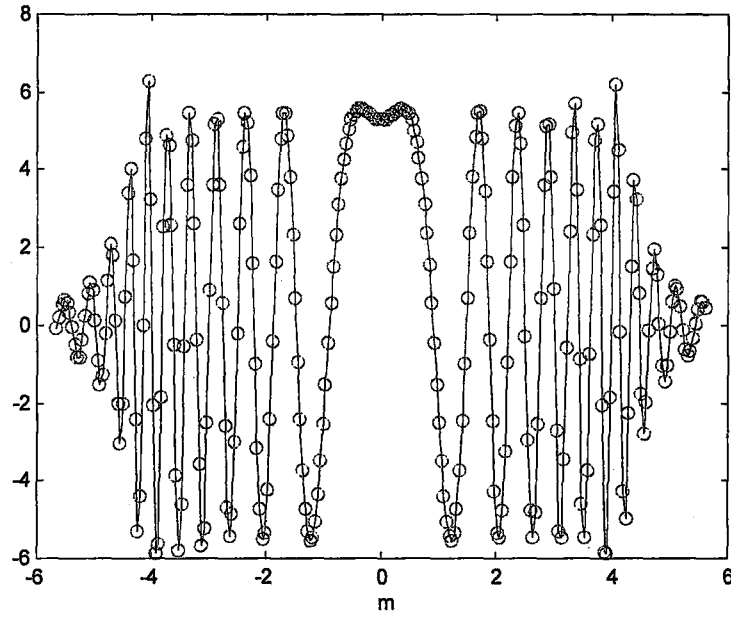


Figure 4. 19 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(c)$

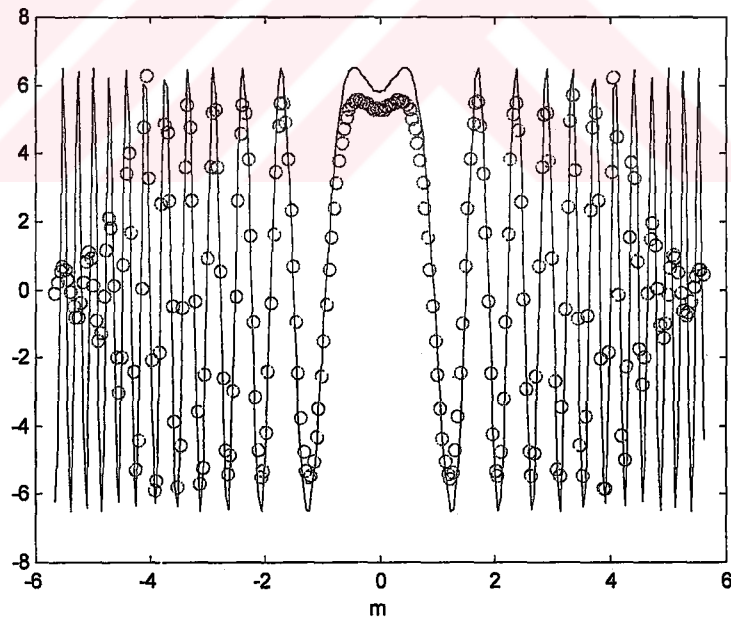


Figure 4. 20 Comparison of the FrMT of $U_{\log}^{-1}(c)$ and the theoretical FrFT of c

As a last example, Figures 4.16 through 4.20 show the fractional Mellin transform of an inverse logarithmically warped constant signal of value $c=5$. It can be seen in Figure 4.20 that the theoretical and practical calculations do not perfectly match, as

opposed to the results of practical FrFT and FrMT calculations in Figure 4.19 which follow each other fairly closely. The inconsistency between the theoretical and practical results is, again, due to the fact that the FrFT algorithm we use is not an exact computation, but an approximation of the continuous transform.

4.2 The Fractional Mellin Transform without A Priori Knowledge of Time

4.2.1 FrMT with *logwarp.m*

One cannot practically know the initial instant of everyday signals, any instant can be assigned to zero. Even if one knew perfectly the time localization of the signal, it would not be possible to know the mathematical formula of the signal or the one-to-one mapping between the signal and the time values just by looking at the data set. However, the algorithm *Ulog.m* in Appendix A.1, requires complete time orientation knowledge about the analyzed signal. Since that knowledge is not always practically available, one must devise other means to perform time warping of a given arbitrary discrete time signal.

An idea inspired by (Zalubas & Williams, 1995) and (Canfield & Jones, 1993) led to the implementation of the logarithmic warping operation using interpolation. Interpolation allows us to calculate the intermediate sample points of a signal from a given data set. One of the most common interpolation methods is accomplished by assuming that the given N points belong to a polynomial of order $N-1$ and calculating the intermediate points by first finding the appropriate polynomial coefficients, and then evaluating the polynomial at those given data points. A modified version, called the *cubic spline* interpolation, was used in the logarithmic warping function in Appendix A.3. In cubic spline interpolation, each two data points are supposed to be connected with a cubic polynomial. In addition, the function, its derivative, and its second derivative are assumed to be continuous at the data points and the polynomial coefficients for each region between two data points are calculated accordingly.

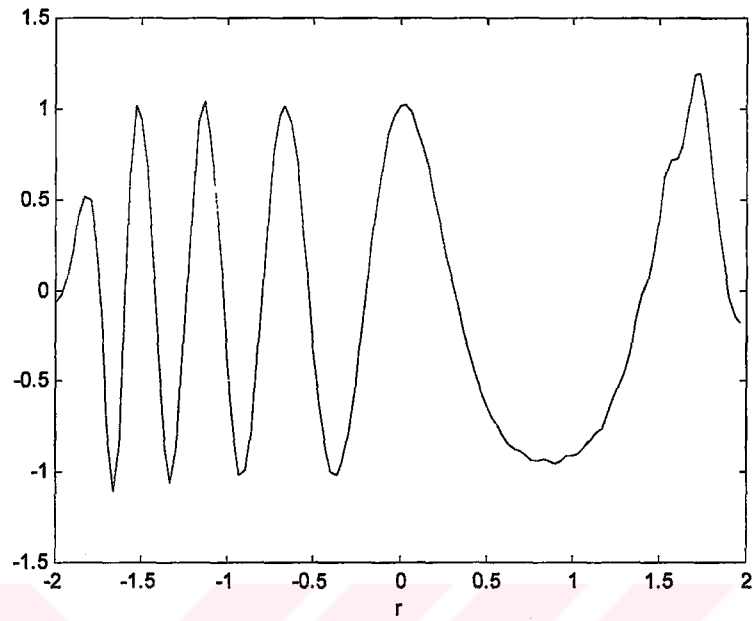


Figure 4. 21 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{j2\pi f_0 t}$

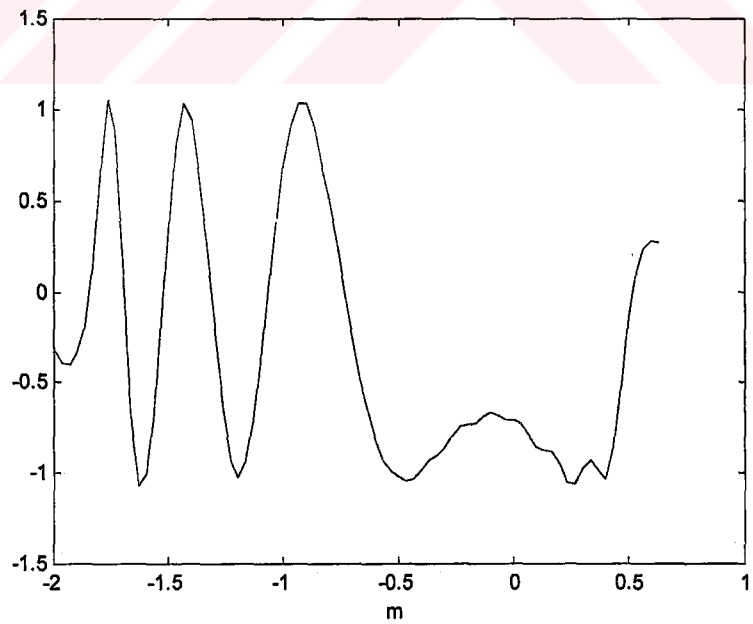


Figure 4. 22 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ using *logwarp.m*

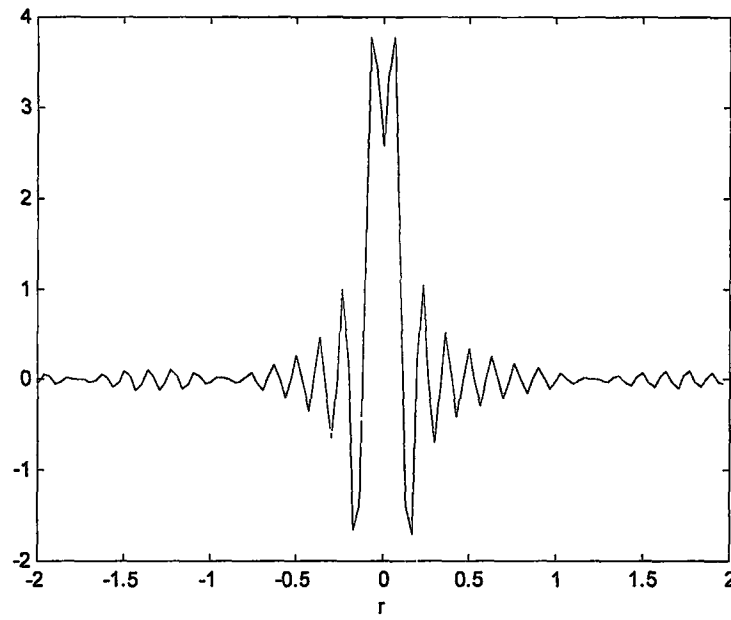


Figure 4. 23 Absolute value of the FrFT of a chirp $e^{-j\pi m_0 t^2}$ computed at the matching angle $\phi = \text{atan}(m_0) + \pi/2$ ($m_0 = 0.1$)

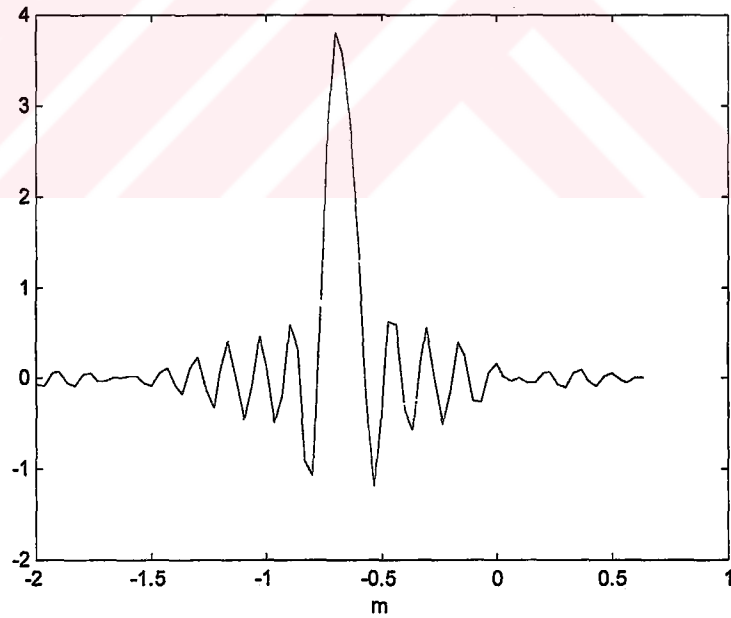


Figure 4. 24 Absolute value of the FrMT of $U_{\log}^{-1}(e^{-j\pi m_0 t^2})$ calculated at the matching angle $\phi = \text{atan}(m_0) + \pi/2$ using *logwarp.m* ($m_0 = 0.1$)

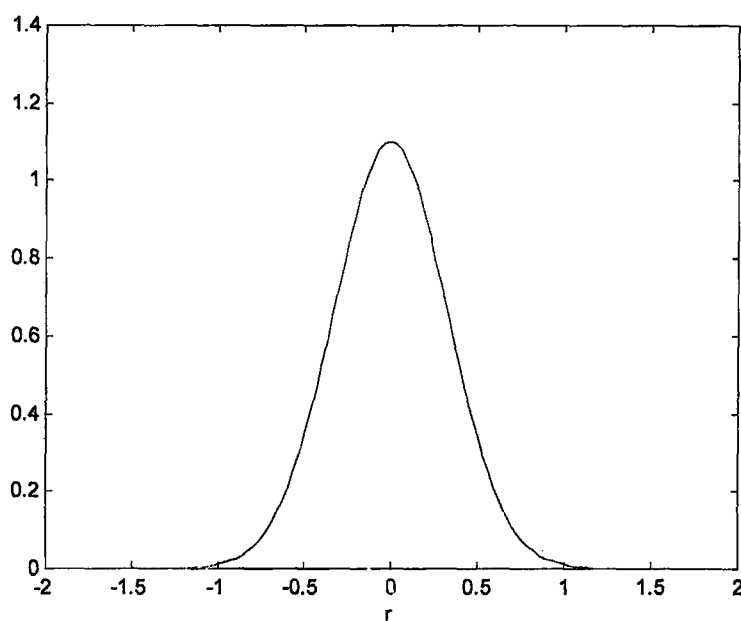


Figure 4. 25 Absolute value of the FrFT at $\phi=\pi/5$ of $e^{-\pi t^2}$

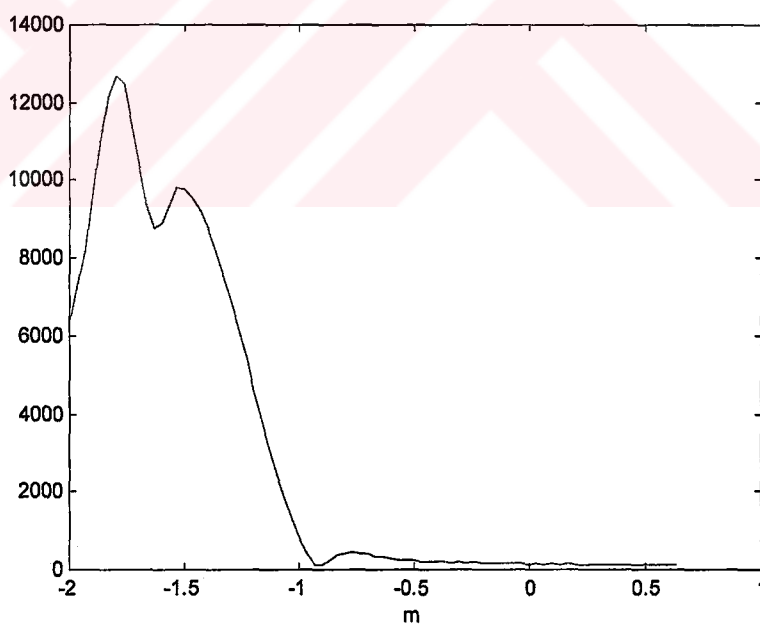


Figure 4. 26 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(e^{-\pi t^2})$ using *logwarp.m*

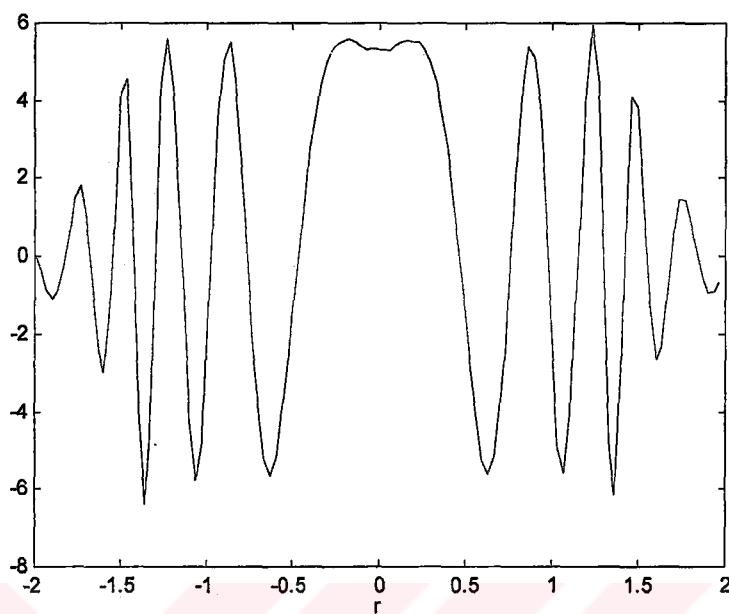


Figure 4. 27 Absolute value of the FrFT at $\phi=\pi/5$ of $c=5$

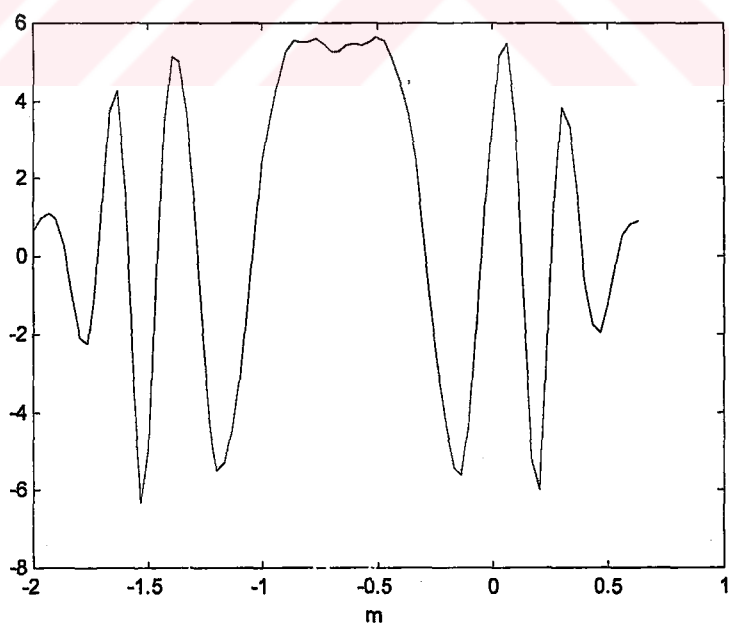


Figure 4. 28 Absolute value of the FrMT at $\phi=\pi/5$ of $U_{\log}^{-1}(c)$ using *logwarp.m*

The results of computing the FrMT of aforementioned signals using *logwarp.m* can be seen in Figures 4.21 through 4.28. Figures 4.21, 4.25 and 4.27 are the plots of the FrFT at $\phi=\pi/5$ of the complex exponential, Gaussian pulse, and the constant signal, respectively. In Figures 4.22, 4.26, and 4.28, we plotted the FrMT at $\phi=\pi/5$ of the inverse logarithmic warped versions of the same signals. In Figures 4.23 and 4.24, we see the FrFT of the chirp signal and the FrMT of the inverse logarithmic warped version of the same chirp, calculated at the matching angle value of $\phi=\text{atan}(m_0)+\pi/2$, respectively. The script used in obtaining these figures is given in Appendix A.14.

The results are far from satisfactory. Although similarities can be observed, the differences are not negligibly small. Similarity is the most in the case of the constant function, because the value of the constant function is independent of the time orientation information which reduces the interpolation errors.

4.2.2 FrMT with *logwarp3.m*

The results of Section 4.2.1 necessitate the fine tuning of the warping algorithm. Upon a reinspection of the process, it was realized that the function *fracf.m* given in Appendix A.7 is very sensitive to the sampling of its input signal. It requires that the input signal is a time-bandwidth product limited signal. It operates on an input vector x which is assumed to be the samples of a signal f , obtained at a rate $2\Delta_x$ where the Wigner distribution support region of the signal f is confined to a circle of diameter Δ_x around the origin of the time-frequency plane (Ozaktas et. al, 1996). Thus $(\Delta_x)^2$ becomes the time-bandwidth product of the signal f . For a signal of duration Δ_x , starting at $-\Delta_x/2$ and ending at $\Delta_x/2$, the total number of samples are calculated as

$$\begin{aligned}
 T &= \Delta_x \\
 f_s &= 2\Delta_x \\
 T_s &= \frac{1}{2\Delta_x} \\
 N &= \frac{T}{T_s} = 2\Delta_x^2.
 \end{aligned}
 \tag{4.10}$$

It is more convenient to express the above quantities of signal duration T , sampling rate f_s , and the sampling period T_s in terms of the number of samples N . Thus, we have

$$\begin{aligned}
 N &= \frac{T}{T_s} = 2\Delta_x^2 \\
 \Delta_x &= \sqrt{\frac{N}{2}} \\
 T &= \Delta_x = \sqrt{\frac{N}{2}} \\
 f_s &= 2\Delta_x = 2\sqrt{\frac{N}{2}} = \sqrt{2N} \\
 T_s &= \frac{1}{2\Delta_x} = \frac{1}{\sqrt{2N}}.
 \end{aligned} \tag{4.11}$$

If the input signal does not satisfy these conditions, results produced by the fractional Fourier transform algorithm might be erroneous.

The input signal of the FrFT algorithm must be a vector positioned around the origin of the time-frequency plane. Accordingly, the output of the logarithmic warping algorithm has to be reshaped so as to correspond to the logarithmic warped values of a signal concentrated around the origin. Since the farthest we can proceed and warp is $\ln(t_f)$ as shown in Eq. (3.14), which corresponds to $\ln(T/2)$ in our case, the smallest time instance value that can be taken to have a symmetric time support around the origin is $-\ln(T/2)$.

The script *logwarp3.m* in Appendix A.9, uses the helper function *nearast.m* in Appendix A.10 to find the greatest number of samples N_2 , less than the number of samples N of the input signal x , that would have a duration of $2\ln(T/2)$ and satisfy the rules given in Eq. (4.11). This number N_2 is used with the script *fracfsamples.m* given in Appendix A.11 to form a time vector t_2 , spanning $[-\ln(T/2) \ln(T/2)]$ and satisfying the sampling condition $f_s = \sqrt{2N}$. This new time vector t_2 , now forms the basis of the interpolation instead of t_{os} that was introduced in Section 3.4.

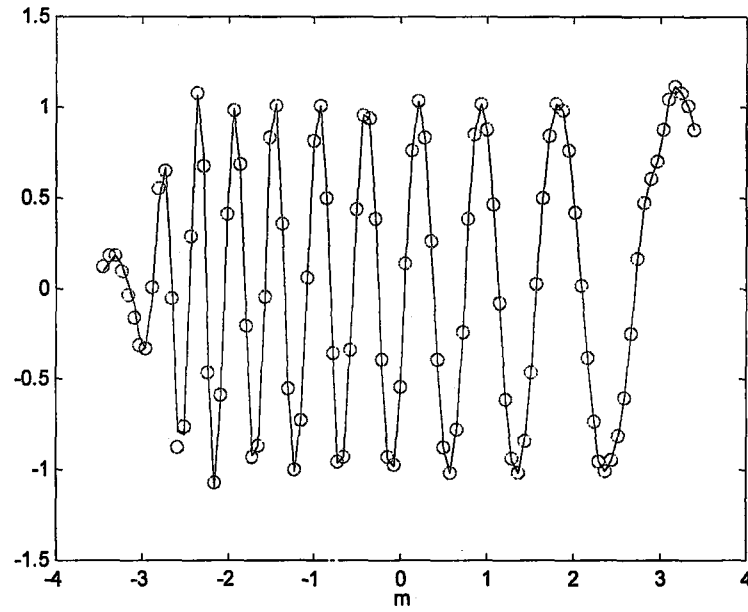


Figure 4. 29 Comparison of the practical FrFT at $\phi=\pi/5$ of the complex exponential $e^{j2\pi f_0 t}$ with the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ computed using *logwarp3.m*

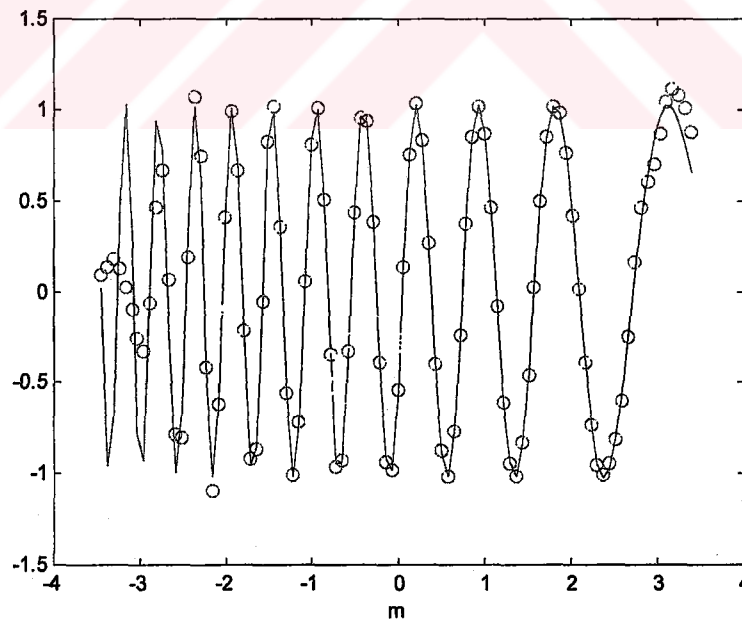


Figure 4. 30 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the complex exponential $e^{j2\pi f_0 t}$ with the FrMT of $U_{\log}^{-1}(e^{j2\pi f_0 t})$ computed using *logwarp3.m*

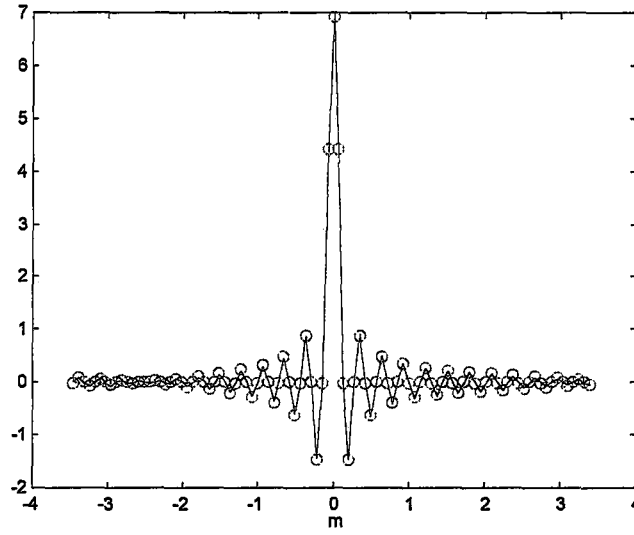


Figure 4. 31 Comparison of the practical FrFT of the chirp $e^{j\pi m_0 t^2}$ at the matching angle $\phi = \text{atan}(m_0) + \pi/2$ with the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ at the same angle computed using *logwarp3.m*

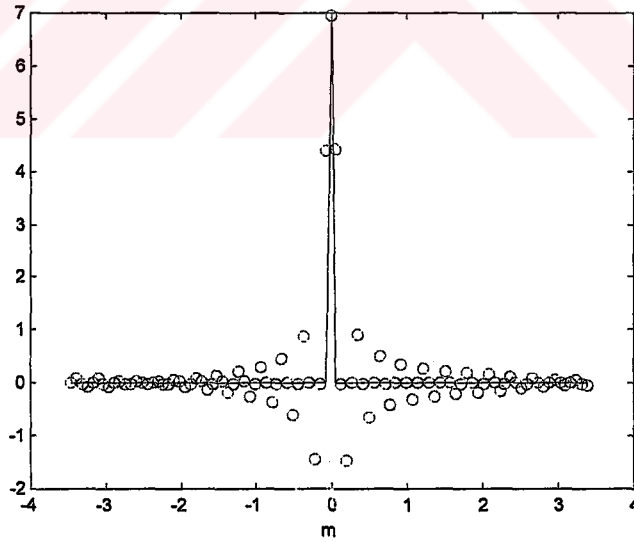


Figure 4. 32 Comparison of the theoretical FrFT of the chirp $e^{j\pi m_0 t^2}$ at the matching angle $\phi = \text{atan}(m_0) + \pi/2$ with the FrMT of $U_{\log}^{-1}(e^{j\pi m_0 t^2})$ at the same angle calculated using *logwarp3.m*

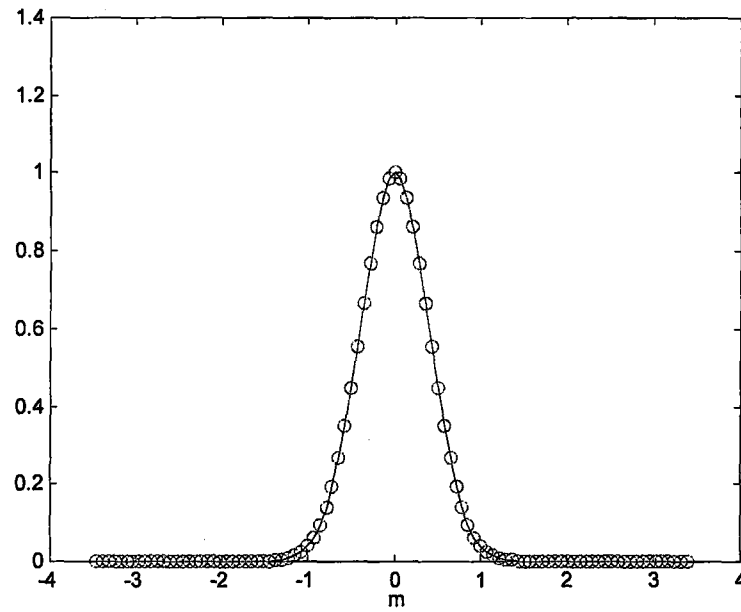


Figure 4.33 Comparison of the practical FrFT at $\phi=\pi/5$ of the Gaussian signal

$e^{-\pi t^2}$ with the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ computed using *logwarp3.m*

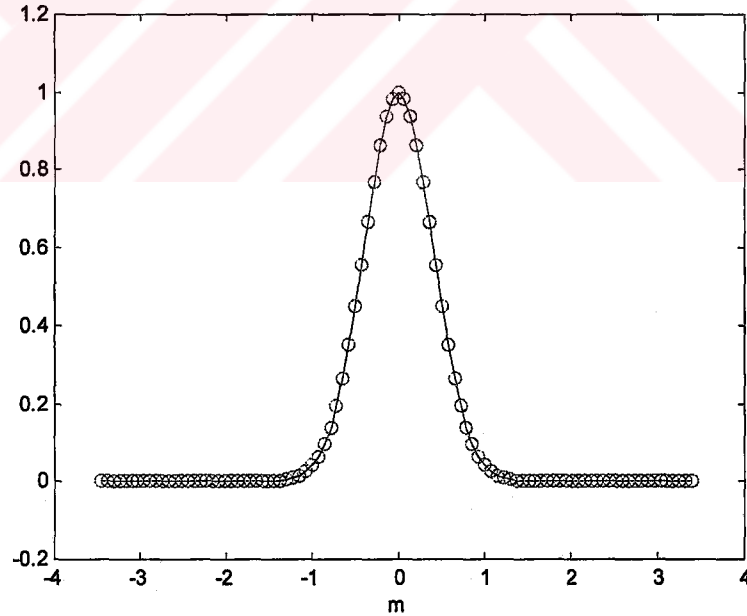


Figure 4.34 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the Gaussian signal

$e^{-\pi t^2}$ with the FrMT of $U_{\log}^{-1}(e^{-\pi t^2})$ calculated using *logwarp3.m*

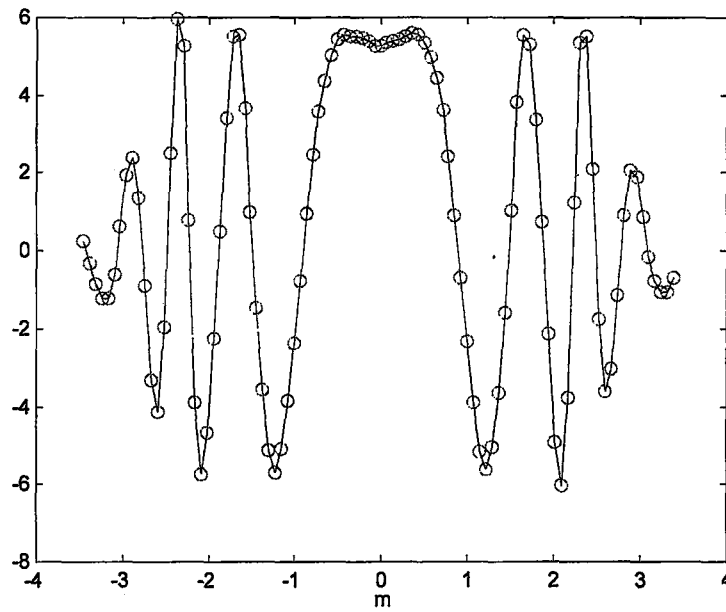


Figure 4. 35 Comparison of the practical FrFT at $\phi=\pi/5$ of the constant signal $c=5$ with the FrMT of $U_{\log}^{-1}(c)$ computed using *logwarp3.m*

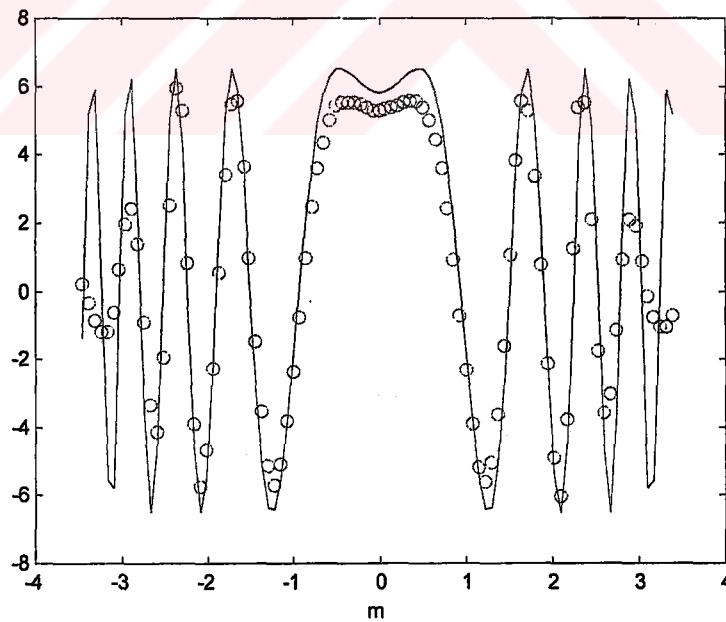


Figure 4. 36 Comparison of the theoretical FrFT at $\phi=\pi/5$ of the constant signal $c=5$ with the FrMT of $U_{\log}^{-1}(c)$ calculated using *logwarp3.m*

Figures 4.29 through 4.36 show the results of computing the FrMT using the *logwarp3.m* algorithm. The signals that are used are the complex exponential, Gaussian and constant signals, respectively. Figures 4.29, 4.33 and 4.35 compare the FrFTs at $\phi=\pi/5$ of the original signals with the FrMTs at the same angle of the inverse logarithmically warped versions of the signals. In those figures, the circles represent the data points calculated with the FrMT. Figures 4.30, 4.34 and 4.36 compare the theoretical FrFTs for $\phi=\pi/5$ of the above mentioned signals with the FrMTs of the inverse logarithmically warped signals. In Figures 4.31 and 4.32, the FrFT of the chirp signal is compared with the FrMT of the inverse logarithmically warped chirp signal at the matching transform angle of $\phi = \text{atan}(m_0)+\pi/2$. The script used for these simulations is given in Appendix A.15.

This time, the practical computations match perfectly. Theoretical computations can not be reproduced exactly because of the approximate nature of the FrFT algorithm (Ozaktas et. al., 1996). Theoretical and practical results match perfectly in case of the Gaussian signal, fit less for the complex exponential, chirp and the constant signal examples.

CHAPTER FIVE

APPLICATIONS OF THE FRACTIONAL MELLIN TRANSFORM

5.1 FrMT of the Hypbolic Chirp Signal

As a first application, the detection of the hyperbolic chirp signal

$$s_{\phi}(t) = \frac{1}{\sqrt{t}} e^{-j\pi(\cot\phi)(\log t)^2 + j2\pi\frac{r_0}{\sin\phi}\log t} \quad \text{Eq. (5.1)}$$

defined in (Akay 2000) was experimented with the FrMT algorithm. This signal, when logarithmically warped, reduces to a regular chirp as

$$\hat{s}_{\phi}(t) = e^{-j\pi(\cot\phi)t^2 + j2\pi\frac{r_0}{\sin\phi}t} \quad \text{Eq. (5.2)}$$

Here, r_0 is called the initial frequency parameter. Notice that the chirp rate parameter is given as $m_0 = -\cot(\phi)$. We know that, when computed at the matching angle of $\phi + \pi/2$, the FrFT of Eq. (5.2) should produce an impulse.

The hyperbolic chirp is plotted in Figure 5.1. It begins with a high frequency oscillation whose frequency diminishes logarithmically as time progresses. Aliasing towards the origin is inevitable since the logarithm function tends to minus infinity as its argument tends to zero. The *logwarp3.m* and the FrFT algorithms assume that the support of the signal is a symmetric region around the origin. When ϕ is taken to be $\pi/(2.1)$ and r_0 is chosen as 1, the theoretical FrMT calculated at the matching angle $\phi = \pi/(2.1)$ of the hyperbolic chirp signal should concentrate as an impulse at around $m=1$.

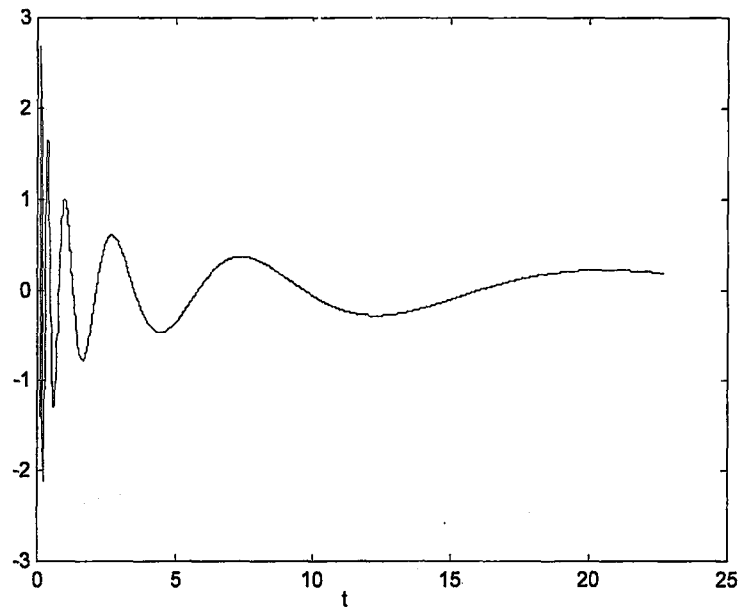


Figure 5. 1 The real part of the hyperbolic chirp of Eq. (5.1) for $\phi=\pi/(2.1)$ and $r_0=1$

The theoretical FrMT of Eq. (5.1) is illustrated in Figure 5.2 where the impulse is located at $m=1$. When Eq. (5.1) is first logarithmically warped using the *logwarp3.m* algorithm followed by an FrFT at an angle $\phi=\pi/(2.1)$, the result is not exactly an impulse, but rather a broader peak correctly localized at $r=1$ (see Figure 5.3). Figures 5.4 and 5.5 show the results of the same experiment for transform angles $\phi=\pi/3$. This time the two figures obtained bear less resemblance. It should be noted that logarithmic warping by *logwarp3.m* produces much less number of interpolated samples, which may affect the correctness of the results.

In Figure 5.6 the hyperbolic chirp and its warped versions are illustrated together. The top figure window shows the hyperbolic chirp. In the middle figure window, its theoretically warped version which corresponds to Eq. (5.2) is plotted. The bottom figure window shows the logarithmic warping carried out by the warping algorithm *logwarp3.m*. The theoretically and practically warped versions are similar but not exactly equal.

In the above simulation, a hyperbolic chirp signal having the chirp rate angle value of $\phi=\pi/(2.1)$, which corresponds to a very low chirp rate parameter of approximately 0.0749, was used. We observed that experimenting with signals having higher chirp rate angles causes aliasing and prevents proper interpolation of the signal when implementing *logwarp3.m*. Due to the insufficient representative nature of aliased data, sample points corresponding to aliased values are incorrectly interpolated as lower frequency components. This distortion of the signal leads to an erroneous FrMT result. Figure 5.7 illustrates this situation with an hyperbolic chirp signal having a chirp rate angle $\phi=\pi/5$ which corresponds to a chirp rate parameter value of approximately 1.3764. It can be seen that the high frequency components that are aliased in the middle figure window can only be represented by lower frequency exponentials in the bottom figure window. This is a drawback of interpolation when working with chirp signals. Unless we are employing some form of variable bit rate sampling, it seems to be extremely hard to achieve perfect frequency resolution for a chirp signal. Thus, from the time when the signal starts to become aliased, the interpolated values are no longer correct. The interpolation algorithm performs independent of the frequency content of the analyzed signal and it tries to create the smoothest possible curve with the given data points.

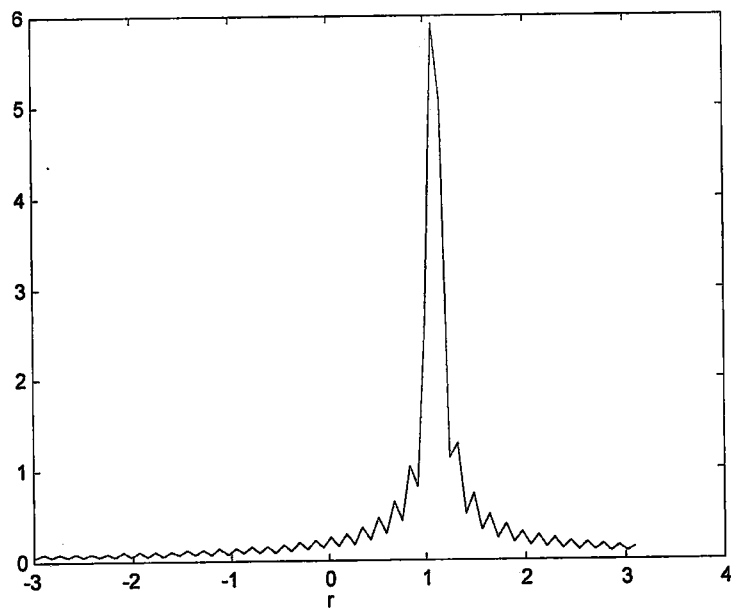


Figure 5. 2 Absolute value of the FrFT of the theoretically logarithmic warped hyperbolic chirp computed at the matching angle $\phi=\pi/2.1$

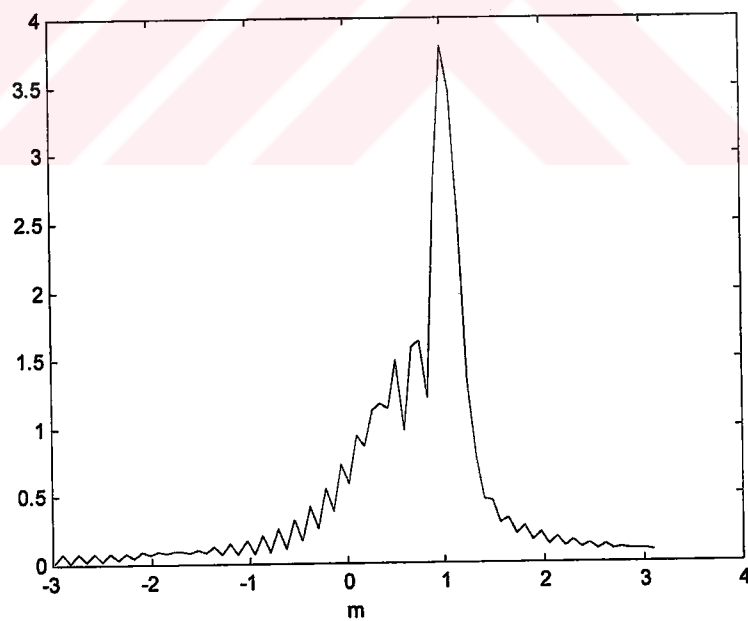


Figure 5. 3 Absolute value of the FrMT of the hyperbolic chirp calculated at the matching angle $\phi=\pi/2.1$ using *logwarp3.m*

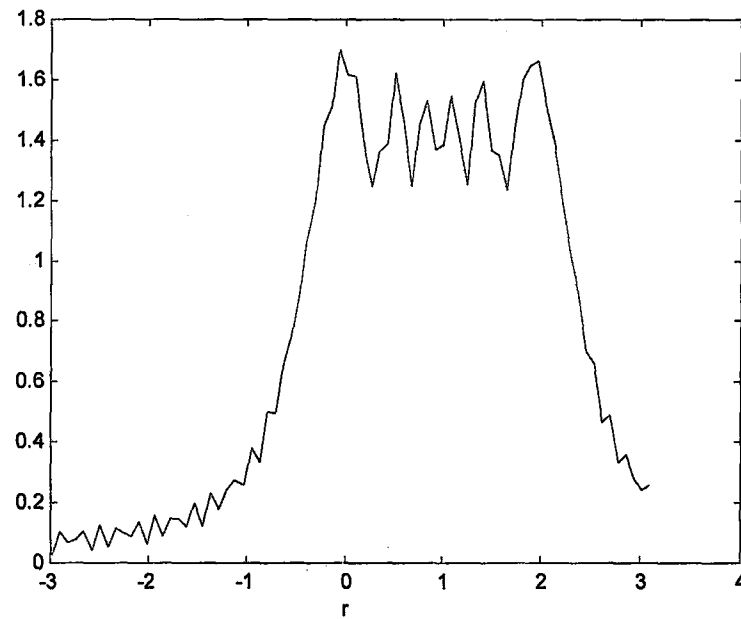


Figure 5. 4 Absolute value of the FrFT of the theoretically logarithmic warped hyperbolic chirp calculated at the mismatched angle $\phi=\pi/3$

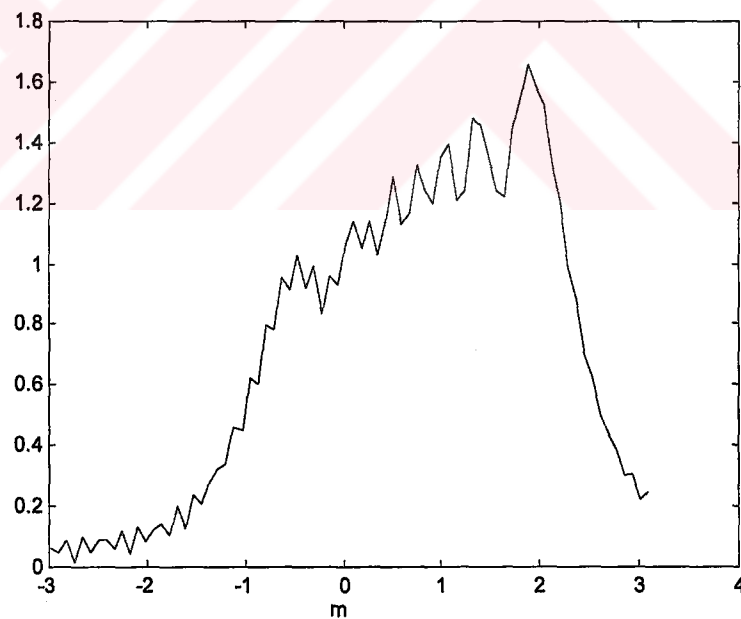


Figure 5. 5 Absolute value of the FrMT of the hyperbolic chirp computed at the mismatched angle $\phi=\pi/3$ using *logwarp3.m*

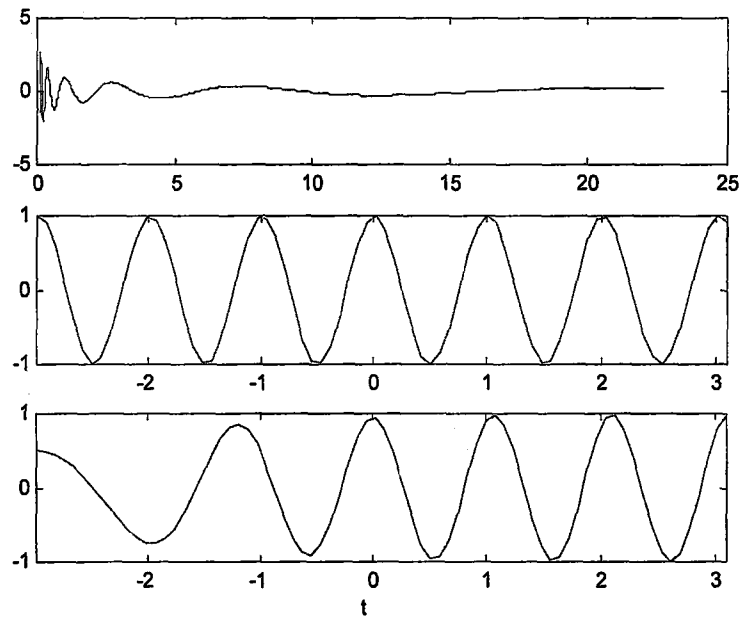


Figure 5. 6 The hyperbolic chirp of chirp rate $\cot(\pi/(2.1))$ (top), theoretical logarithmic warping of the hyperbolic chirp (middle) and the practical logarithmic warping of the hyperbolic chirp using *logwarp3.m* (bottom)

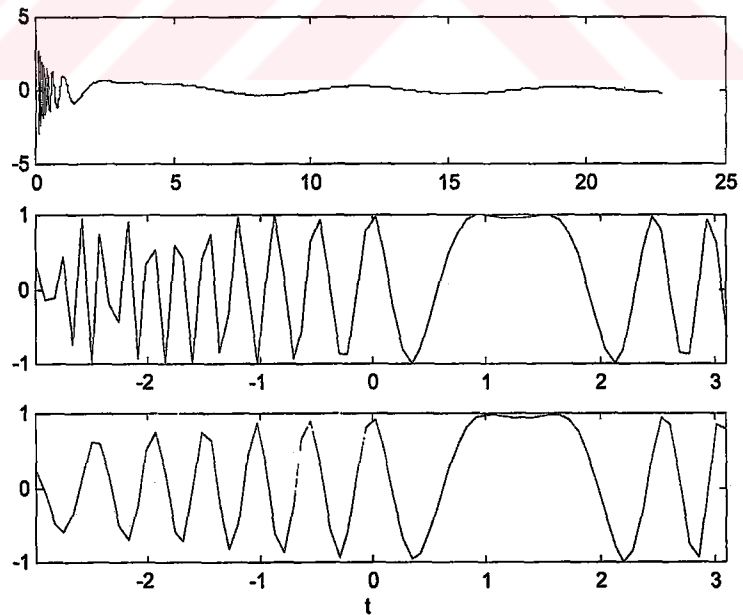


Figure 5. 7 The hyperbolic chirp (top) of chirp rate $\cot(\pi/5)$, theoretical logarithmic warping of the hyperbolic chirp (middle) and the practical logarithmic warping of the hyperbolic chirp using *logwarp3.m* (bottom)

5.2 A Simple Application – Filtering in the Fractional Scale Domain

In this application, we use the following signal

$$s(t) = \frac{1}{\sqrt{t}} e^{-\pi(\ln(t))^2} + \frac{1}{\sqrt{t}} e^{j[\pi m_0 (\ln(t))^2 + 2\pi f_0 \ln(t)]} \quad \text{Eq. (5.3)}$$

which is just the inverse logarithmically warped version of a Gaussian plus a chirp signal. Thus, it can alternatively be expressed using the operator U_{\log}^{-1} as

$$s(t) = (U_{\log}^{-1})(e^{-\pi t^2}) + (U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]}). \quad \text{Eq. (5.4)}$$

Observing Figures 5.8, 5.9 and 5.10, we can see that the warped Gaussian signal is dominated by the warped chirp in $s(t)$. To separate the two signals which are known to be logarithmically warped, we will use the fractional Mellin transform (FrMT). In the simulations, we use the parameter values $m_0=-1$ and $f_0=6$. To be able to represent the warped chirp signal with parameter m_0 as an impulse, we have to compute the FrMT at the matching angle of $\phi = \text{atan}(m_0) + \pi/2$. Thus, for the value of $m_0=-1$, the matching angle is found to be $\pi/4$. When we calculate the FrMT at the matching angle of $\pi/4$ radians, the warped chirp signal becomes concentrated as an impulse at $\frac{f_0}{\sin \phi} = \frac{6}{\sqrt{2}} \approx 4.24$ as depicted in Figure 5.11. By applying a simple ideal fractional scale lowpass filter with cut off scale of $m=1$, we isolate the Gaussian signal as shown in Figure 5.12.

To transform the signal back to the time domain, we need to compute the inverse FrMT, given by

$$(\mathbb{M}^\phi)^{-1} = (\mathbb{F}^\phi U_{\log})^{-1} = U_{\log}^{-1} \mathbb{F}^{-\phi}. \quad \text{Eq. (5.5)}$$

Notice that, as opposed to the inverse FrFT, the inverse FrMT at angle ϕ is not equal to the FrMT of the signal at the negative angle $-\phi$. According to Eq. (5.5), the first operation that must be performed is the calculation of the inverse FrFT of the filtered signal at angle $-\phi$ which is illustrated in Figure 5.13.

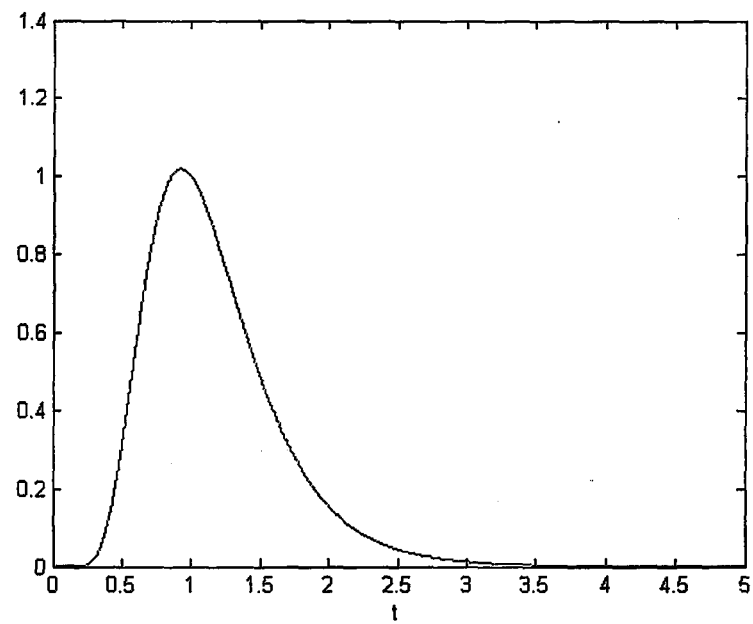


Figure 5. 8 Graph of $(U_{\log}^{-1})(e^{-\pi t^2})$

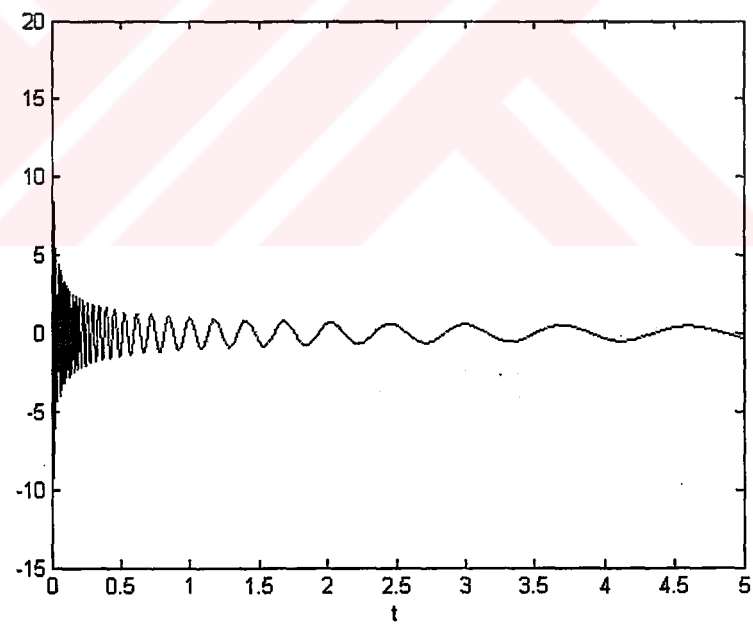


Figure 5. 9 Graph of $(U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]}), f_0 = 6, m_0 = -1$

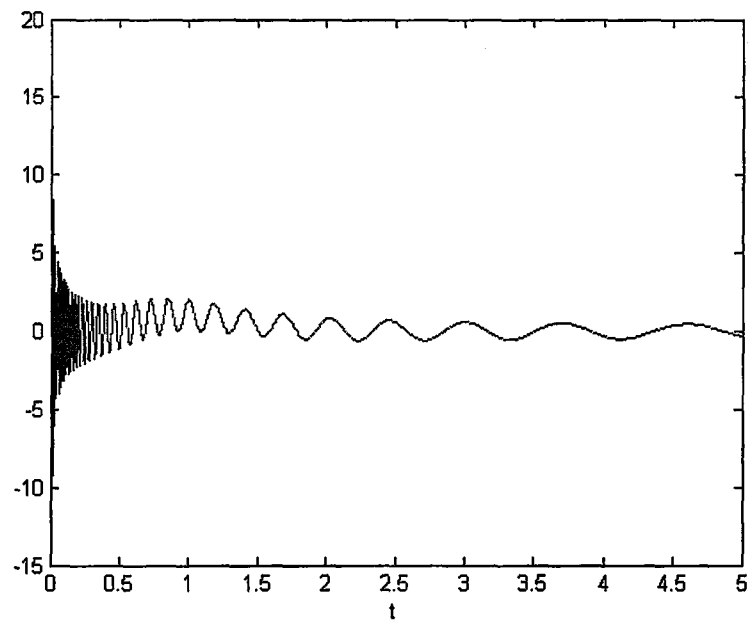


Figure 5. 10 Graph of $s(t) = (U_{\log}^{-1})(e^{-\pi t^2}) + (U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]})$

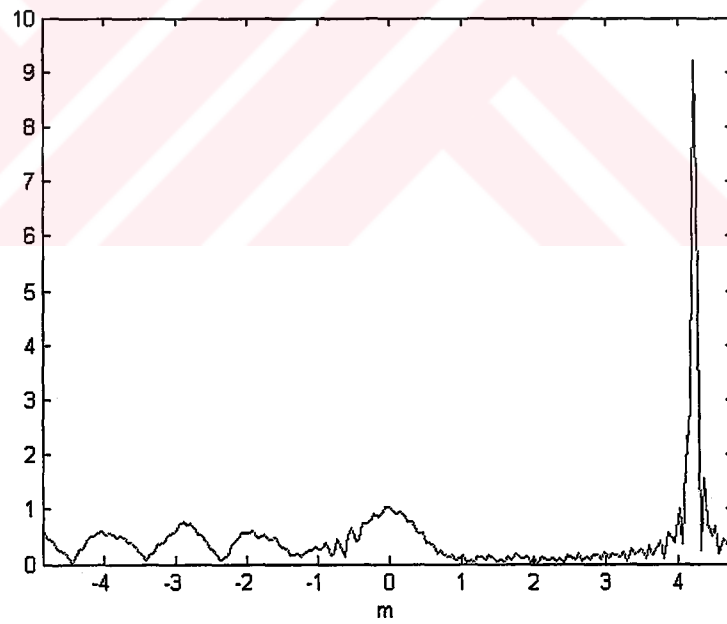


Figure 5. 11 FrMT of $s(t) = (U_{\log}^{-1})(e^{-\pi t^2}) + (U_{\log}^{-1})(e^{j[\pi m_0 t^2 + 2\pi f_0 t]})$ computed at the matching angle $\phi = \text{atan}(m_0) + \pi/2$

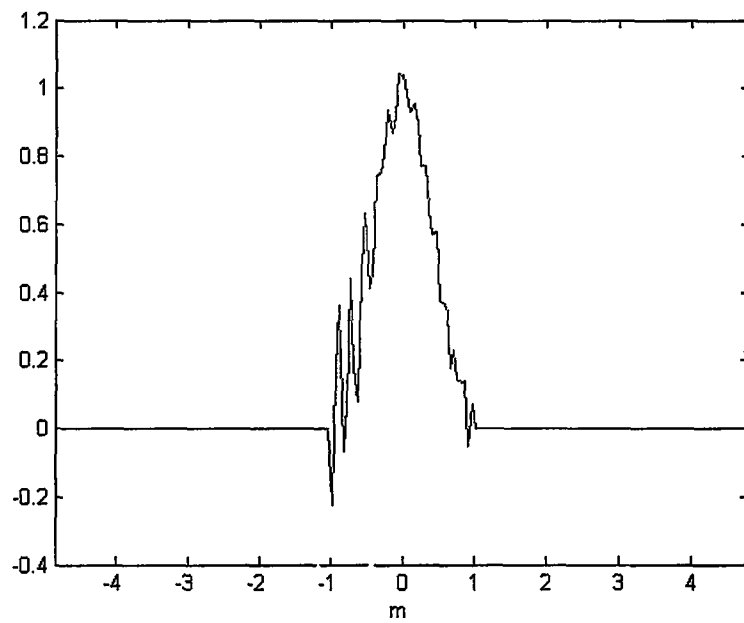


Figure 5. 12 The isolated Gaussian in the fractional scale domain obtained after a simple masking of the FrMT signal in Figure 5.11

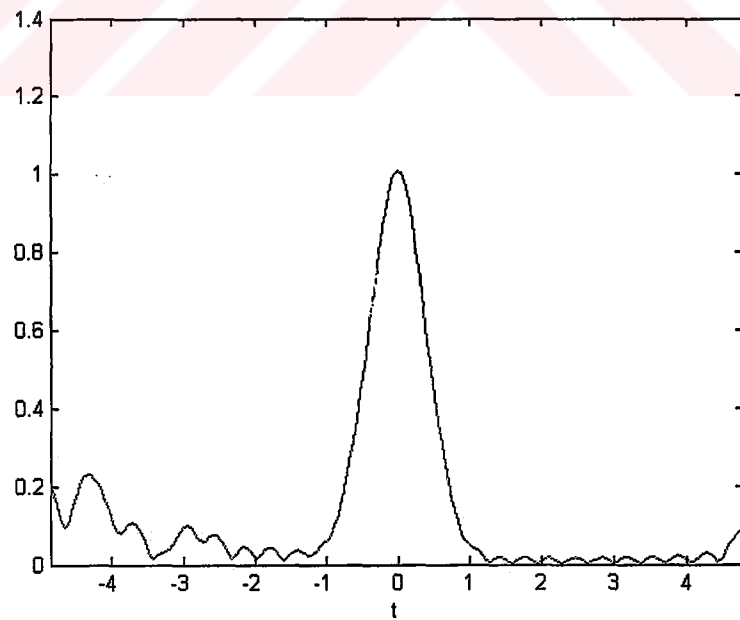


Figure 5. 13 Inverse fractional Fourier transform at the matching angle ϕ of the filtered signal in Figure 5.12

As the final step, the inverse logarithmic warping defined by

$$(\mathbf{U}_{\log}^{-1}s)(t) \equiv \frac{1}{\sqrt{t}}s(\ln t) \quad \text{Eq. (5.6)}$$

should be applied. The third warping algorithm *logwarp3.m*, developed in Chapter 3, was modified to implement the inverse logarithmic warping operation in Eq. (5.6). Applying this inverse logarithmic warping algorithm to the signal in Figure 5.13, we recover back the warped Gaussian signal as seen in Figure 5.14. When compared with the original warped Gaussian as in Figure 5.15, it can be seen that both original and recovered signals match closely, the only disagreement being towards the origin of the time axis. In our opinion, this discrepancy is inevitable due to the effect of the logarithmic modulation which introduces high frequency oscillations near the time origin.

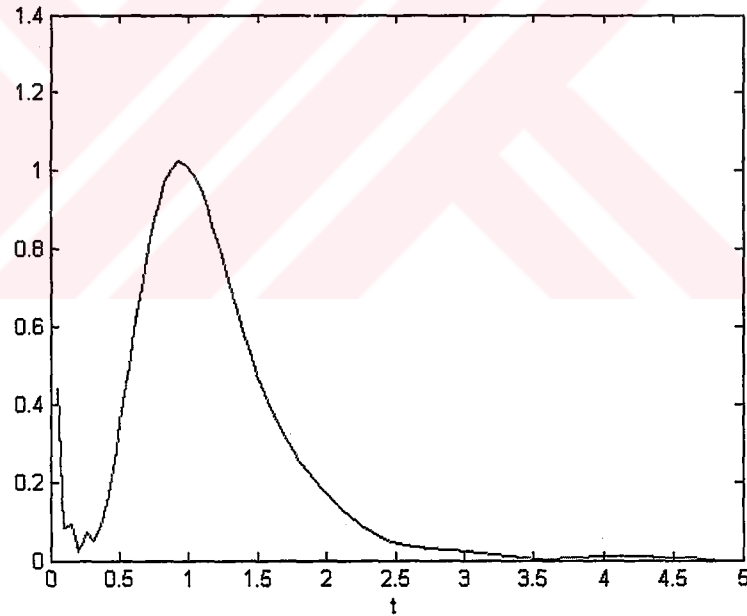


Figure 5. 14 The recovered warped Gaussian signal obtained after inverse logarithmic warping of the signal in Figure 5.13

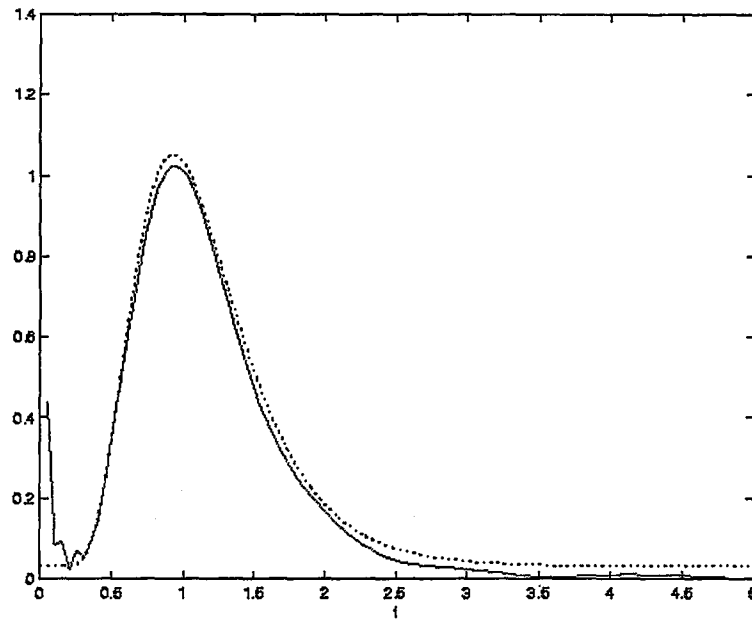


Figure 5. 15 Comparison of the recovered warped Gaussian (solid line) with the original starting signal $(U_{\log}^{-1})(e^{-\pi t^2})$ (dashed line).

5.3 An Example of FrFT versus FrMT

As a final demonstration, we compute the FrFT and FrMT of a square pulse of width 2, centered at $t=4$ as seen in Figure 5.16. We compute the fractional transforms at six different angles between 0 to $\pi/2$ inclusive.

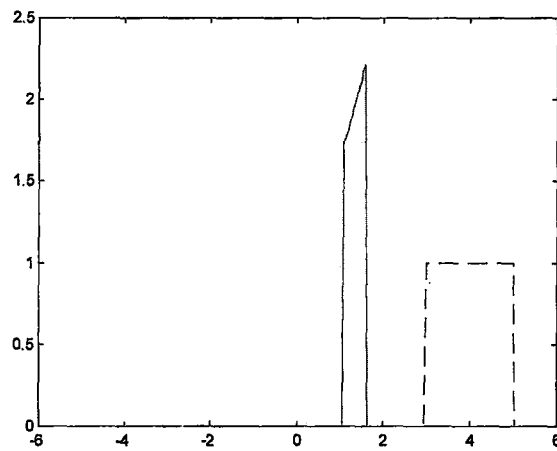


Figure 5. 16 The rectangular pulse (dashed) and its logarithmically warped (via U_{\log}) version (solid)

Figure 5.16 displays the analyzed rectangular pulse signal as plotted using dashed line. Its logarithmically warped version is also plotted using solid line on the same figure. The magnitude plots of the FrFT and FrMT of the rectangular pulse at varying angles from 0 to $\pi/2$ are shown in Figure 5.17. It is interesting to note that although both transformations start out quite differently, at $\pi/2$ they both assume the familiar sinc function appearance.

It is also interesting to observe the amplitude changes of the FrFT and FrMT signals. In addition, as the angle value of the transforms change towards $\pi/2$, the support regions move towards the origin. Finally at $\phi=\pi/2$, both transforms are centered at the origin.

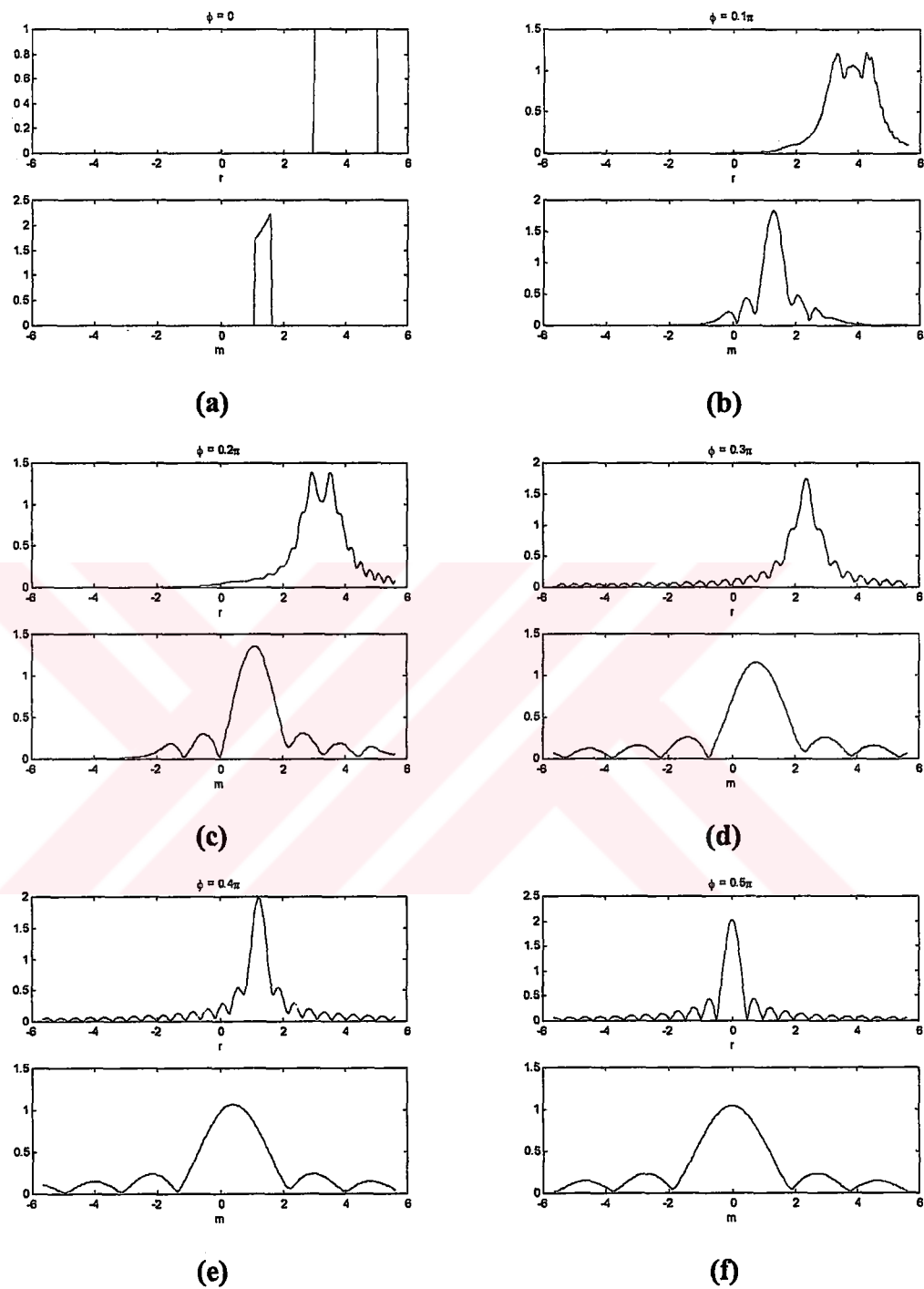


Figure 5. 17 The FrFT (top) and the FrMT (bottom) of a rectangular pulse at different angles between 0 to $\pi/2$ radians increasing in steps of $\pi/10$ (all figures show the magnitude plots of the transforms)

CHAPTER SIX

CONCLUSIONS

Our aim in this thesis was to realize the digital computation of the fractional Mellin transform. We adopted a two stage approach instead of building the algorithm from scratch. We used the fact that computing the FrMT of a signal is equivalent to first logarithmically warping the given signal and then computing the FrFT (Akay, 2000). There already was a discrete FrFT algorithm developed by Ozaktas and his colleagues (Ozaktas et. al., 1996). There remained only to implement the logarithmic warping in a discrete fashion, before combining it with the discrete FrFT algorithm.

Three types of warping algorithms were developed and compared in this thesis. The first algorithm required complete knowledge about the signal, that is, the mathematical formulation and the complete time orientation. Then, the logarithmic warping formula

$$(U_{\log} s)(t) = e'^{1/2} s(e')$$

was applied by direct substitution of e' in place of the independent time variable t followed by a multiplication by the factor $e'^{1/2}$. The second warping algorithm required only the initial time instant and the sampling rate. Logarithmic warping was then accomplished by first constructing the base time vector from the given initial time and the sampling rate values. The warped values $s(e'^{(n)})$ corresponding to each time instant $t(n)$ in the base time vector was computed by interpolation. The third algorithm operated directly on the input signal complying with the constraints of time localization and sampling rate imposed by the discrete FrFT algorithm. Warping was again done by interpolation.

The best results were obtained by the first warping algorithm. However, the amount of a priori information required makes it unsuitable for practical use. The second warping algorithm was designed to be less dependent on a priori signal information. This time however, limitations of the discrete FrFT algorithm were encountered. The discrete FrFT algorithm assumes that the Wigner distribution of the analyzed signal is confined to a circle of diameter Δx around the origin of the time-frequency plane. Thus, allowing arbitrary signal localization on the time axis becomes impossible. Moreover, the sampling rate of the analyzed signal must also be taken as $2\Delta x$. If the analyzed signal is not already discretized satisfying these constraints, the results produced by the discrete FrFT algorithm are prone to errors. Our second warping algorithm did not comply with these constraints and the results produced were quite different than those produced by the first warping algorithm.

To minimize the errors, the third logarithmic warping algorithm was developed so as to comply with the limitations of the discrete FrFT algorithm. The input signal was assumed to be centered around $t=0$ and the interpolation points were spaced so as to satisfy the sampling frequency of $2\Delta x$. This time the results agreed with their theoretical counterparts more closely. The difficulty encountered, however, was the loss of data due to warping. For a signal having the time support from $-T/2$ to $T/2$, only that portion of the signal from $-T/2$ to $\ln(T/2)$ can be warped since no values corresponding to time instants greater than $e^{\ln(T/2)} = T/2$ are available.

In light of the above discussion, a complete, stand-alone discrete implementation of the FrMT requires two main components; a logarithmic warping algorithm that can process all of the input data and produce the corresponding output and a discrete FrFT implementation that can work completely on discrete data without taking the sampling frequency into account. Otherwise, the discrete FrMT requires quite large and carefully conditioned data sets, which are not always available in signal processing.

REFERENCES

- Altes, R.A., (January 1978) The Fourier-Mellin transform and mammalian hearing. Journal of Acoustical Society of America, 63, 173-184
- Akay, O. (2000). Unitary and Hermitian Fractional Operators and Their Extensions. Ph.D. thesis. University of Rhode Island
- Akay, O., & Boudreaux – Bartels, G. F. (December 1998). Unitary and Hermitian fractional operators and their relation to the fractional Fourier transform. IEEE Signal Processing Letters, 5, 312-314
- Akay, O., & Boudreaux – Bartels, G. F. (August 1998). Fractional Mellin transform: An extension of fractional frequency concept for scale. 8th Digital Signal Processing (DSP) Workshop (in CD-ROM)
- Almeida, L.B. (November 1994) The fractional Fourier transform and time-frequency representations. IEEE Transactions on Signal Processing, 42, 3084-3091.
- Baraniuk, R., (September 1993). Signal transform covariant to scale changes. IEE Electronics Letters, 29, 1675-1676
- Baraniuk, R., & Jones, D. (October 1995). Unitary equivalence: A new twist in signal processing. IEEE Transactions on Signal Processing, 42, 2269-2282

Canfield, K.G., & Jones, D.L. (November 1993). Implementing time-frequency representations for non-Cohen classes. Proceedings of the Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, 1993, 2, 1464-1468

Casasent, D., & Psaltis, D. (January 1997) New optical transforms for pattern recognition. Proceedings of the IEEE, 65, 77-84

Cohen, L. (December 1993). The scale representation. IEEE Transactions on Signal Processing, 41, 3275–3292

Cohen, L. (1995). Time – Frequency Analysis. Englewood Cliffs, NJ: Prentice - Hall

Hlawatsch, F., & Bourdeaux-Bartels, G. F. (Apr. 1992). Linear and quadratic time-frequency signal representations. IEEE Signal Processing Magazine, 9, 21-67

Ozaktas, H.M, Arikan, O., Kutay, M.A., & Bozdagi, G. (September 1996) Digital computation of the fractional Fourier transform. IEEE Transactions on Signal Processing, 44, 2141-2150

Sayeed, A.M., R., & Jones, D. (June 1996). Integral transforms covariant to unitary operators and their implications for joint signal representations. IEEE Transactions on Signal Processing, 44, 1365-1377

Sundaram, H., Joshi, S.D.,& Bhatt, R.K.P. (July 1997). Scale periodicity and its sampling theorem. IEEE Transactions on Signal Processing, 45, 1862-1865

WEB_1, (2004). The MacTutor History of Mathematics Archive, <http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Mellin.html>, 21/7/2004

Zalubas, E.J. & Williams W.J. (1995) Discrete scale transform for signal analysis. (ICASSP-95) 1995 International Conference on Acoustics, Speech, and Signal Processing, 3, 1557-1560

Zwicke, P.E., & Kiss, I. Jr. (March 1983) A new implementation of the Mellin transform and its application to radar classification of ships. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-5, 191-199



APPENDIX A

SOURCE CODE FOR THE ALGORITHMS USED IN THIS THESIS

A.1 The Function Ulog(),

This function performs logarithmic warping defined by Eq. (2.36). It assumes previous knowledge of the mathematical representation of the signal, initial, final instances of the signal and the number of samples taken.

```
function [WarpedTimeSignal] = ...
    Ulog(InputFunction, Ti, Tf, N)
% The function that calculates the logarithmically
% warped version of the function given by the
% string 'InputFunction'. The string 'InputFunction'
% should obey the rules of operation dictated
% by matlab programming conventions, i.e. a '*'
% means a vector or matrix multiplication. To
% define a multiplication similar to the continuous
% sin(t)*cos(t) one must use the '.*'. The initial
% and final times Ti and Tf are necessary to
% determine at which points the function and the
% resulting warped version are evaluated. N is
% assumed to be the sufficient number of samples
% including fun(Ti) and fun(Tf)
fun = inline(InputFunction);
t = linspace(Ti, Tf, N);
WarpedTimeSignal = exp(t/2) .* feval(fun, exp(t));
```

A.2 Testulog.m

This is the script that tests Ulog.m script in A.1

```

t = linspace(0.01, 2, 10001);
yinvw = 1 ./ sqrt(t) .* sin( 2 * pi * 0.5 * log(t));
yw = ulog('1./sqrt(t).* sin( 2 * pi * 0.5 * log(t))'...
    , 0, 2, 10001);
figure(1)
subplot(2,1,1), plot(t, yinvw),
title(['\bfU_{log}^{\{-1\}}\rm sin(2*pi*f*t)',...
    ' = 1/sqrt(t)*sin(2*pi*f*ln(t))'])
grid on
subplot(2,1,2), plot(t, yw),
title(['\bfU_{log}\rm (1/sqrt(t)*sin(2*pi*f*ln(t))=','...
    'sin(2*pi*f*t)'])
grid on
yinvw = 1 ./ sqrt(t) .* sin(pi * 2 * log(t).^2);
yw = ulog(['1 ./ sqrt(t) .* sin( pi * 2 * ',...
    'log(t).^2)'], 0, 2, 10001);
figure(2)
subplot(2,1,1), plot(t, yinvw),
title(['\bfU_{log}^{\{-1\}}\rm sin(pi*m*t^2) =','...
    ' 1/sqrt(t)*sin(pi*m*(ln(t))^2)'])
grid on
subplot(2,1,2), plot(t, yw),
title(['\bfU_{log}\rm (1/sqrt(t)*sin(pi*m*(ln(t))^2)',...
    '=sin(pi*m*t^2)'])
grid on
figure(3)
plot(t, log(t), 'r-.', t, log(t).^2),
legend('ln(t)', 'ln^2(t)')
grid on

```

A.3 Logwarp.m – First version

```

function y = logwarp ( x, t0, fs )
% logarithmic warp of the input vector x
% t0 : beginning instant, default 0
% fs : sampling rate, default 8192

if nargin == 2
    fs = 8192;
elseif nargin ==1
    fs = 8192;
    t0 = 0;
end

[m, n] = size ( x );
if m*n ~= length (x)
    error('Please enter a vector');
end

if m ~= 1
    x = x.' ;
end

N = length ( x );

% (tf - t0 ) * fs + 1 = N
tf = ( N - 1 ) / fs + t0 ;
if exp(t0) > tf
    error('Not enough temporal information to warp.')
end

t = linspace ( t0, tf, N );
t_warped = exp ( t ) ; % warped t

% in order to interpolate correctly we need to
oversample

```

```

% choose fs = 10 * fs
N_os = (tf - t0) * (10 * fs) + 1 ;
% no of samples in the oversampled version
t_os = linspace ( t0, tf, N_os ); % oversampled t

if (find(isnan(x))==1)
    x(1)=x(2);
elseif (find(isnan(x))==length(x))
    x(end) = x(end-1);
else
    x(isnan(x)) = (x(find(isnan(x))+1) + x(find(isnan(x))-
1))./2;
end

x_int = spline (t, x, t_os) ;
% interpolated x using the spline method
% we will use this to find the intermediate values
corresponding to exp(ti) s

% for any ti find the element in t_os that is closest
to exp(ti)
t_max = log(tf);
t_max_in_t = t(abs(t - t_max )==min( abs( t - t_max ) )
) ;
N_max = find(t == t_max_in_t);
for n = 1 : N_max
    t_nearest = t_os(abs(t_os - exp( t(n) ) )==min( abs(
t_os - exp ( t(n) ) ) ) );
    x_nearest = x_int ( t_os == t_nearest ) ;
    y( n ) = exp ( t( n ) / 2) * x_nearest ;
end

```

A.4 Logwarptest3.m – used to test logwarp.m

```

% logwarp test file

```

```

% yw : warped single sine
% yuw : unwarped single sine
% as in Zalubas' paper

ti = -2;
tf = 2;
t = sample( ti,tf, 100);
et = exp(t);
f = 1/4;
yw = exp(t/2) .* sin ( 2* pi * f * exp(t));
yuw = sin ( 2* pi * f * t);
yuwz = [yuw];

plot(t,yw)
hold on
plot(t,yuwz,':')
t2 = sample(ti,log(tf),100);
ywlz = logwarp(yuwz,ti,100); plot(t2,ywlz,'.')
hold off
xlabel('t')

```

A.5 Logwarptest4.m – used to test logwarp.m

```

% logwarp test file
% yw : warped single sine
% yuw : unwarped single sine
% as in Zalubas' paper

ti = -2;
tf = 2;
t = sample( ti,tf, 100);
et = exp(t);
f = 1/4;
yw = exp(t/2) .* square ( 2* pi * f * exp(t));
yuw = square ( 2* pi * f * t);

```

```

yuwz = [yuw];

plot(t,yw)
hold on
plot(t,yuwz,':')
t2 = sample(ti,log(tf),100);
ywlz = logwarp(yuwz,ti,100); plot(t2,ywlz,'.')
hold off
xlabel('t')

```

A.6 Sample.m – used in logwarptest3 and logwarptest4

```

function t = sample ( t0, tf, fs )
% function t = sample ( t0, tf, fs )
% creates a time vector t from t0 to tf with
% sampling rate fs

t = linspace(t0, tf, (tf - t0) * fs + 1 );

```

A.7 Frac.m – The Script Used for Taking the Fractional Fourier Transform

```

function [res] = fracF(fc,a)
% This function operates on the vector fc which is
assumed to
% be the samples of a function, obtained at a rate 2
deltax
% where the Wigner distribution of the function f is
confined
% to a circle of diameter of deltax around origin.
deltax is merely
% the time-bandwidth product of the function f. Also fc
% is assumed to have even number of elements.
% This function maps the fc to its fractional Fourier
transform vector
% whose elements are the samples of the fractional

```

```

Fourier transform
% of the function f. Lengths of the input and output
vectors are same
% if the input vector has even number of elements.
% Operating interval:  $-2 \leq a \leq 2$ 
% It uses the core function corefr.m

flag = 0;

if (a>0) & (a<0.5)
    flag = 1;
    a = a-1;
end
if (a>-0.5) & (a<0)
    flag = 2;
    a = a+1;
end

if (a>1.5) & (a<2)
    flag = 3;
    a = a-1;
end

if (a>-2) & (a<-1.5)
    flag = 4;
    a = a+1;
end

if (a==0)
    res = fc;
else
    if (a==2) | (a==-2)
        res = fliplr(fc);
    else
        res = corefr(fc,a);
    end
end

```



```

end
end

if (flag==1) | (flag==3)
    res = corefr(res,1);
end

if (flag==2) | (flag==4)
    res = corefr(res,-1);
end

```

A.8 Corefr.m – Helper function to Fracf.m

```

function [res] = corefr(fc,a)
% Core funtion for computing fractional Fourier
transform.
% valid only when 0.5 <= abs(a) <= 1.5
% Decomposition: chirp mutiplication - chirp
convolution - chirp mutiplication
% See Digital Computation of frac. Four. transform
paper.

deltax = sqrt(length(fc)/2);

phi= a*pi/2;
N = fix(length(fc)/2);
deltax1 = 2*deltax;
alpha = 1/tan(phi);
beta = 1/sin(phi);

x = [-N:N-1]/deltax1;
fc = fc(:);
fc = fc(1:2*N);
f1 = exp(-i*pi*tan(phi/2)*x.*x); %multiplication by
chirp!

```

```

f1 = f1(:);
fc      = fc.*f1;
x  = x(:);
clear x;
t  = [-2*N+1:2*N-1]/deltax1;
hlptc = exp(i*pi*beta*t.*t);
clear t;
hlptc = hlptc(:);

N2 = length(hlptc);
N3 = 2^(ceil(log(N2+2*N-1)/log(2)));
hlptcz = [hlptc;zeros(N3-N2,1)];
fcz= [fc;zeros(N3-2*N,1)];
Hcfft = ifft(fft(fcz).*fft(hlptcz));
% convolution with chirp

clear hlptcz;
clear fcz;
Hc = Hcfft(2*N:4*N-1);
clear Hcfft;
clear hlptc;

Aphi = exp(-i*(pi*sign(sin(phi))/4- ...
phi/2))/sqrt(abs(sin(phi)));
xx = [-N:N-1]/deltax1;
f1 = f1(:);
res= (Aphi*f1.*Hc)/deltax1; % multiplication by chirp!

clear f1
clear Hc

```

A.9 Logwarp3.m – Logarithmic Warping Compliant with the Fractional Fourier Algorithm

```

function [y, t, t2] = logwarp3 ( x )
% logarithmic warp of the input vector x
% x is assumed to be situated around the
% origin over the interval -T/2 T/2 with
% sampling rate fs=sqrt(2*N)

[m, n] = size ( x );
if m*n ~= length (x)
    error('Please enter a vector');
end

if m ~= 1
    x = x.' ;
end

N = length ( x );
T = sqrt(N/2);
fs = 2 * T;
t = fracfsamples(N);
ti = t(1);
tf = t(end);

if (exp(ti) > tf)
    error('Not enough temporal information to warp. ');
end

N2 = nearest(N);
t2 = fracfsamples(N2);
% t2 is the largest warppable vector around 0 for which
the
% required data values exist in x(t)

```

```

t_warped = exp ( t2 ) ; % warped t

% in order to interpolate correctly we need to
oversample
% choose fs = 10 * fs
N_os = (tf - ti) * (10 * fs ) + 1 ;
% no of samples in the oversampled version
t_os = linspace ( ti, tf, N_os ); % oversampled t

x_int = spline (t, x, t_os) ;
% interpolated x using the spline method
% we will use this to find the intermediate values
corresponding to exp(ti) s

% for any ti find the element in t_os that is closest
to exp(ti)

for n = 1 : N2
    t_nearest = t_os(abs(t_os - exp( t2(n) ) )==min(
abs( t_os - exp ( t2(n) ) ) ) );
    x_nearest = x_int ( t_os == t_nearest ) ;
    y( n ) = exp ( t2( n ) / 2) * x_nearest ;
end

```

A.10 Nearest.m – helper function used in Logwarp3.m

```

function [N2mod, N2] = nearest(N)
% for a vector of length N which is assumed to be
sampled at
% fs=sqrt(2*N) and spanning the interval -T/2 T/2
around the
% origin where T = sqrt(N/2), this function finds the
next
% largest vector of length N2 < N that spans 2*(T/2 -

```

```

1) or
% smaller that can comply with the sampling rate
rule
% fs2 = sqrt(2*N2)

fs = sqrt( 2 * N );
T = sqrt ( N / 2 );
To2 = T / 2;
% T over 2, half the duration

T2o2 = log(To2);
T2 = T2o2 * 2;
N2 = 2 * (T2)^2;

N2mod = 2 * fix(N2 / 2);
T2mod = sqrt ( N2mod / 2 );
s = ['N = ',num2str(N),' T = ',num2str(T),' --> '];
s = [s, 'N2 = ',num2str(N2),' T2 = ',num2str(T2),' -->
'];
s = [s, 'N2mod = ',num2str(N2mod),' T2mod =
',num2str(T2mod)];
disp(s)

```

A.11 Fracfsamples.m – Helper function used by logwarp3.m

```

function [t, fs, Ts] = fracfsamples(no_of_samples)
% Usage :
% [t, fs, Ts] = fracfsamples(no_of_samples)
%
% a function to provide the correct number of samples
and duration
% for reliable operation of the FracF algorithm
% sampling period Ts and number of samples N satisfy
% Ts = 1 / sqrt(2 * N)
% no_of_samples : must be an even number

```

```

% t      : time vector spanning -1 / 4Ts, 1 / 4Ts
% fs     : resulting sampling frequency
% Ts     : resulting sampling period

if mod(no_of_samples,2) ~= 0
    error('You should supply an even number of
samples');
end

N = no_of_samples;
Ts = 1 / sqrt(2 * N);
fs = 1 / Ts;

T = 1 / (2 * Ts);
t = linspace(-T / 2, T / 2, N + 1);
t = t(1 : end - 1);

```

A.12 Tstfrmulog.m – Test file for obtaining the fractional Mellin Transform with Ulog.m

```

n=8;
N = 2^n
N2 = nearest(N);
t = fracfsamples(N);
ti = t(1);
tf = t(end);
fs = sqrt(2*N);

T = sqrt(N/2); % duration of the signal with N samples
and sampling rate fs = sqrt(2*N)

f = 1.5;
s1 = exp(j * 2 * pi * f * t);

```

```

figure(4), plot(t, real(s1)), title('\ite^{j2\pift}'),
    xlabel('t')
U_inv_log_s1 = t .^ (j * 2 * pi * f - 1/2);
figure(5), pltcplx(t, (U_inv_log_s1)),
figure(5), title('\bfU^{-1}_{log} \rm \ite^{j2\pift}'),

% take FracFT at an arbitrary angle pi/5
phi = pi / 10;
a = phi * 2 / pi;

% FracFT of s1
FracFT_s1 = fracf(s1,a);
figure(1), plot(t,real(FracFT_s1))
title('FracFT of \ite^{j2\pift}')

%FracMT of U_inv_log_s1
FracMT_U_inv_s1 = fracf(ilog('t .^ (j * 2 * pi * 1.5 -
1/2)',ti,tf,N),a);
figure(2), plot(t, real(FracMT_U_inv_s1))
hold on, plot(t, real(FracFT_s1),'ro'), hold off

title('FracMT of \bfU^{-1}_{log} \rm \ite^{j2\pift}'))
FracFT_s1_th = sqrt(1 + j * tan(phi)) .* exp(-j * pi *
((t).^2 + f^2) * tan(phi)) .* exp(j * 2 * pi * t * f *
sec(phi));
figure(3), plot(t, real(FracFT_s1_th));
title('Theoretical FracFT of
\ite^{j2\pift}');xlabel('t')
hold on, plot(t, FracMT_U_inv_s1,'ro'),hold off

FracFT_s1_th = sqrt(1 + j * tan(phi)) .* exp(-j * pi *
((t).^2 + f^2) * tan(phi)) .* exp(j * 2 * pi * t * f *
sec(phi));
figure(6), plot(t, real(FracFT_s1_th));

```

```

    title('Theoretical FracFT of
\ite^{j2\pift}');xlabel('t')
    hold on, plot(t, FracMT_U_inv_s1,'ro'),hold off

    pause

    %3 ----- linear chirp -----
    alpha = 0.05;
    s3 = exp(j * pi * alpha * (t).^2);
    figure(4), plot(t, real(s3)),
title('\ite^{j\pi\alphan^2}')
    U_inv_log_s3 = 1 ./ sqrt(t) .* exp(j * pi * alpha *
(log(t)).^2);
    figure(5), pltcplx(t, (U_inv_log_s3)),
title(' \bfU^{-1}_{log} \rm \ite^{j\pi\alphan^2} ')

    phi = pi / 5;
    p = 10;
    a = 1 + (alpha) * 2 / pi;
    phi = atan(alpha);

    % FracFT of s2
    FracFT_s3 = fracf(s3,a);
    figure(1), plot(t, real(FracFT_s3))
    title('FracFT of \ite^{j\pi\alphan^2}')
    %FracMT of U_inv_log_s3
    FracMT_U_inv_s3 = fracf(ulog('1 ./ sqrt(t) .* exp(j *
pi * 0.05 * (log(t)).^2)',ti,tf,N),a);
    figure(2), plot(t, real(FracMT_U_inv_s3))
    hold on, plot(t, real(FracFT_s3),'ro'), hold off
    title('FracMT of \bfU^{-1}_{log} \rm of
\ite^{j\pi\alphan^2} ')
    FracFT_s3_th = sqrt((1 + j* tan(phi))/(1 + alpha *
tan(phi) )) .* exp(j * pi * ((t).^2) * ((alpha -
tan(phi))/(1 + alpha * tan(phi) )));

```



```

figure(3), plot(t, real(FracFT_s3_th));
title('Theoretical FracFT of
\ite^{j\pi\alphat^2}');xlabel('t')
hold on, plot(t, real(FracMT_U_inv_s3),'ro'),hold off

FracFT_s3_th = sqrt((1 + j* tan(phi))/(1 + alpha *
tan(phi) )) .* exp(j * pi * ((t).^2) * ((alpha -
tan(phi))/(1 + alpha * tan(phi) )));
figure(6), plot(t, real(FracFT_s3_th));
title('Theoretical FracFT of
\ite^{j\pi\alphat^2}');xlabel('t')
hold on, plot(t, FracMT_U_inv_s3,'ro'),hold off
pause

%4 ----- Gaussian signal -----
s4 = exp(-pi * (t).^2);
figure(4), plot(t, (s4)), title(' \ite^{-\pit^2}')

U_inv_log_s4 = 1 ./ sqrt(t) .* exp(-pi * (log(t)).^2);

figure(5), pltcplx(t,(U_inv_log_s4)), title('\bfU^{\log} \ite^{-\pit^2}')

phi = pi /5;
a = phi * 2 / pi;

% FracFT of s4
FracFT_s4 = fracf(s4,a);
figure(1), plot(t, real(FracFT_s4))
title('FracFT of \ite^{-\pit^2}')
%FracMT of U_inv_log_s4
FracMT_U_inv_s4 = fracf(ulog('1 ./ sqrt(t) .* exp(-pi *
(log(t)).^2)',ti,tf,N),a);
figure(2), plot(t,real(FracMT_U_inv_s4))

```

```

hold on, plot(t, real(FracFT_s4),'ro'), hold off
title('FracMT of  $\mathbf{U}^{-1}_{\log}$  \rm of  $\mathbf{ite}^{-\pi t^2}$ ')
FracFT_s4_th = exp(-pi * (t).^2);
figure(3), plot(t,abs(FracFT_s4_th));
title('Theoretical FracFT of  $\mathbf{ite}^{-\pi t^2}$ ');xlabel('t')
hold on, plot(t, FracMT_U_inv_s4,'ro'),hold off

FracFT_s4_th = exp(-pi * (t).^2);
figure(6), plot(t,abs(FracFT_s4_th));
title('Theoretical FracFT of  $\mathbf{ite}^{-\pi t^2}$ ');xlabel('t')
hold on, plot(t, abs(FracMT_U_inv_s4),'ro'),hold off
pause

%2 ----- constant term -----

c = 5;
s2 = c * ones(size(t));
figure(4), plot(t, real(s2)), title('constant c')
U_inv_log_s2 = c ./ sqrt(t);
figure(5), pltcplx(t,(U_inv_log_s2)), title('\mathbf{U}^{-1}_{\log} \rm \itc')

phi = pi /5;
a = phi * 2 / pi;

% FracFT of s2
FracFT_s2 = fracf(s2,a);
figure(1), plot(t,real(FracFT_s2))
title('FracFT of \itc')
%FracMT of U_inv_log_s2
FracMT_U_inv_s2 = fracf(ulong('5 ./
sqrt(t)',ti,tf,N),a);

```

```

figure(2), plot(t,real(FracMT_U_inv_s2))
hold on, plot(t, real(FracFT_s2),'ro'), hold off
title('FracMT of \bfU^{-1}_{log} \rm of \itc')
FracFT_s2_th = c .* sqrt(1 + j * cot(phi)) .* exp(-j *
pi * ((t).^2) * tan(phi));
figure(3), plot(t, real(FracFT_s2_th));xlabel('t')
title('Theoretical FracFT of \itc');
hold on, plot(t, FracMT_U_inv_s2,'ro'),hold off

FracFT_s2_th = c .* sqrt(1 + j * cot(phi)) .* exp(-j *
pi * ((t).^2) * tan(phi));
figure(6), plot(t, real(FracFT_s2_th));
title('Theoretical FracFT of \itc');xlabel('t')
hold on, plot(t, FracMT_U_inv_s2,'ro'),hold off

pause

```

A.13 Pltcplx.m – helper function to tstfrmulog.m

```

function [] = pltcplx(t,z)
% A function to quickly plot real, imaginary and
absolute
% values of complex valued vectors.
% Usage : pltcplx(t,z)

figure(gcf);
zs = inputname(2);
subplot(3, 1, 1), plot(t,real(z)), xlabel(['Real
part']);
subplot(3, 1, 2), plot(t,imag(z)), xlabel(['Imaginary
part']);
subplot(3, 1, 3), plot(t,abs(z)), xlabel(['Absolute
value']);

```

```
subplot(3, 1, 1)
```

A.14 Tstfrmlw.m -- Test file for obtaining the fractional Mellin Transform with logwarp.m

```
% test fractional mellin using the logwarp that does not use
% the time function of the signal
%

%1 ----- complex exponential -----
fs=30;
t = sample(-2,2,fs);t=t(1:end-1);
t2=sample(-2,log(t(end)),fs);t2=t2(1:end-1);
Ti=t(1);

f = 1;
s1 = exp(j * 2 * pi * f * t);
U_inv_log_s1 = t .^ (j * 2 * pi * f - 1/2);

% take FracFT at an arbitrary angle pi/5
phi = pi / 10;
a = phi * 2 / pi;

% FracFT of s1
FracFT_s1 = fracf(s1,a);
figure(1), plot(t,real(FracFT_s1)),xlabel('t')
title('FracFT of \ite^{j2\pift}')

%FracMT of U_inv_log_s1
FracMT_U_inv_s1 = fracf(logwarp(U_inv_log_s1,Ti,fs),a);
figure(2), plot(t2,real(FracMT_U_inv_s1)),xlabel('t')
title('FracMT of \bfU^{-1}_{log} \rm \ite^{j2\pift}')
FracFT_s1_th = sqrt(1 + j * tan(phi)) .* exp(-j * pi * (t.^2
+ f^2) * tan(phi)) .* exp(j * 2 * pi * t * f * sec(phi));
figure(3), plot(real(FracFT_s1_th)),xlabel('t')
title('Theoretical FracFT of \ite^{j2\pift}');

pause
```

```

%3 ----- linear chirp -----
alpha = 0.1;
s3 = exp(j * pi * alpha * t.^2);
U_inv_log_s3 = 1 ./ sqrt(t) .* exp(j * pi * alpha *
(log(t)).^2);

phi = pi /5;

a = 1+ (alpha) * 2 / pi;

% FracFT of s2
FracFT_s3 = fracf(s3,a);
figure(1), plot(t,real(FracFT_s3)),xlabel('t')
title('FracFT of \ite^{j\pi\alphat^2}')
%FracMT of U_inv_log_s3
FracMT_U_inv_s3 = fracf(logwarp(U_inv_log_s3,Ti,fs),a);
figure(2), plot(t2,real(FracMT_U_inv_s3)),xlabel('t')
title('FracMT of \bfU^{-1}_{log} \rm of
\ite^{j\pi\alphat^2}')
FracFT_s3_th = sqrt((1 + j* tan(phi))/(1 + alpha * tan(phi)
)) .* exp(j * pi * (t.^2) * ((alpha - tan(phi))/(1 + alpha *
tan(phi) )));
figure(3), plot(real(FracFT_s3_th),xlabel('t'))
title('Theoretical FracFT of \ite^{j\pi\alphat^2}');

pause

%4 ----- Gaussian signal -----
s4 = exp(-pi * (t).^2);
U_inv_log_s4 = 1 ./ sqrt(t) .* exp(-pi * (log(t)).^2);

phi = pi /5;
a = phi * 2 / pi;

```

```
% FracFT of s4
FracFT_s4 = fracf(s4,a);
figure(1), plot(t,real(FracFT_s4)),xlabel('t')
title('FracFT of \ite^{\-\pit^2}')
%FracMT of U_inv_log_s4
FracMT_U_inv_s4 = fracf(logwarp(U_inv_log_s4,Ti,fs),a);
figure(2), plot(t2,real(FracMT_U_inv_s4)),xlabel('t')
title('FracMT of \bfU^{-1}_{\log} \rm of \ite^{\-\pit^2}')
FracFT_s4_th = exp(-pi * t.^2);
figure(3), plot(real(FracFT_s4_th)),xlabel('t')
title('Theoretical FracFT of ite^{\-\pit^2}');
```

```
pause
```

```
%2 ----- constant term -----
```

```
Ti = t(1);
```

```
c = 5;
```

```
s2 = c * ones(size(t));
```

```
U_inv_log_s2 = c ./ sqrt(t);
```

```
phi = pi /5;
```

```
a = phi * 2 / pi;
```

```
% FracFT of s2
```

```
FracFT_s2 = fracf(s2,a);
```

```
figure(1), plot(t,real(FracFT_s2)),xlabel('t')
```

```
title('FracFT of \itc')
```

```
%FracMT of U_inv_log_s2
```

```
FracMT_U_inv_s2 = fracf(logwarp(U_inv_log_s2,Ti,fs),a);
```

```
figure(2), plot(t2,real(FracMT_U_inv_s2)),xlabel('t')
```

```
title('FracMT of \bfU^{-1}_{\log} \rm of \itc')
```

```
FracFT_s2_th = c .* sqrt(1 + j * cot(phi)) .* exp(-j * pi *
(t.^2) * tan(phi));
```

```
figure(3), plot(real(FracFT_s2_th)),xlabel('t')
```

```
title('Theoretical FracFT of \itc');
```

pause

A.15 Tstfrmlw3.m – Used for Testing the Script Logwarp3.m

```

N = 2^10
N2 = nearest(N);
t = fracfsamples(N) ;
ti = t(1);
tf = t(end);
fs = sqrt(2*N);
t2 = fracfsamples(N2);
T = sqrt(N/2); % duration of the signal with N samples and
sampling rate fs = sqrt(2*N)

T2=sqrt(N2 / 2);

% 1. complex exponential
f = 1.5;
s1 = exp(j * 2 * pi * f * t2);
figure(4), plot(t2, real(s1)). title('s1')
U_inv_log_s1 = t .^ (j * 2 * pi * f - 1/2);
figure(5), pltcplx(t, (U_inv_log_s1)),
figure(5), title('\bfU^{-1}_{log} \rm \ite^{j2\pift}'),

% take FracFT at an arbitrary angle pi/5
phi = pi / 10;
a = phi * 2 / pi;

% FracFT of s1
FracFT_s1 = fracf(s1,a);
figure(1), plot(t2,real(FracFT_s1))
title('FracFT of \ite^{j2\pift}')

%FracMT of U_inv_log_s1
FracMT_U_inv_s1 = fracf(logwarp3(U_inv_log_s1),a);
figure(2), plot(t2, real(FracMT_U_inv_s1))

```

```

hold on, plot(t2, real(FracFT_s1),'ro'), hold off

title('FracMT of \bfU^{-1}_{\log} \rm \ite^{j2\pift}')
FracFT_s1_th = sqrt(1 + j * tan(phi)) .* exp(-j * pi *
((t2).^2 + f^2) * tan(phi)) .* exp(j * 2 * pi * t2 * f *
sec(phi));
figure(3), plot(t2, real(FracFT_s1_th));
title('Theoretical FracFT of \ite^{j2\pift}');xlabel('t2')
hold on, plot(t2, FracMT_U_inv_s1,'ro'),hold off

FracFT_s1_th = sqrt(1 + j * tan(phi)) .* exp(-j * pi *
((t).^2 + f^2) * tan(phi)) .* exp(j * 2 * pi * t * f *
sec(phi));
figure(6), plot(t, real(FracFT_s1_th));
title('Theoretical FracFT of \ite^{j2\pift}');xlabel('t')
hold on, plot(t2, FracMT_U_inv_s1,'ro'),hold off

pause

%3 ----- linear chirp -----
alpha = 0.05;
s3 = exp(j * pi * alpha * (t2).^2);
figure(4), plot(t2, real(s3)), title('s3')
U_inv_log_s3 = 1 ./ sqrt(t) .* exp(j * pi * alpha *
(log(t)).^2);
figure(5), pltcplx(t, (U_inv_log_s3)),
title(' \bfU^{-1}_{\log} \rm \ite^{j\pi\alphan^2}')

phi = pi /5;
p = 10;
a = 1 + (alpha) * 2 / pi;
phi = atan(alpha);

% FracFT of s2
FracFT_s3 = fracf(s3,a);
figure(1), plot(t2, real(FracFT_s3))
title('FracFT of \ite^{j\pi\alphan^2}')

```



```

%FracMT of U_inv_log_s3
FracMT_U_inv_s3 = fracf(logwarp3(U_inv_log_s3),a);
figure(2), plot(t2, real(FracMT_U_inv_s3))
hold on, plot(t2, real(FracFT_s3),'ro'), hold off
title('FracMT of \bfU^{-1}_{log} \rm of
\ite^{j\pi\alphat^2}')
FracFT_s3_th = sqrt((1 + j* tan(phi))/(1 + alpha * tan(phi)
)) .* exp(j * pi * ((t2).^2) * ((alpha - tan(phi))/(1 + alpha
* tan(phi) )));
figure(3), plot(t2, real(FracFT_s3_th));
title('Theoretical FracFT of
\ite^{j\pi\alphat^2}');xlabel('t2')
hold on, plot(t2, FracMT_U_inv_s3,'ro'),hold off

FracFT_s3_th = sqrt((1 + j* tan(phi))/(1 + alpha * tan(phi)
)) .* exp(j * pi * ((t).^2) * ((alpha - tan(phi))/(1 + alpha *
tan(phi) )));
figure(6), plot(t, real(FracFT_s3_th));
title('Theoretical FracFT of
\ite^{j\pi\alphat^2}');xlabel('t')
hold on, plot(t2, FracMT_U_inv_s3,'ro'),hold off
pause

%4 ----- Gaussian signal -----

s4 = exp(-pi * (t2).^2);
figure(4), plot(t2, (s4)), title('s4')
U_inv_log_s4 = 1 ./ sqrt(t) .* exp(-pi * (log(t)).^2);

figure(5), pltcplx(t,(U_inv_log_s4)), title('\bfU^{-1}_{log}
\ite^{-\pit^2}')

phi = pi /5;
a = phi * 2 / pi;

% FracFT of s4

```

```

FracFT_s4 = fracf(s4,a);
figure(1), plot(t2, abs(FracFT_s4))
title('FracFT of  $\text{ite}^{-\pi t^2}$ ')
%FracMT of U_inv_log_s4
FracMT_U_inv_s4 = fracf(logwarp3(U_inv_log_s4),a);
figure(2), plot(t2,abs(FracMT_U_inv_s4))
hold on, plot(t2, real(FracFT_s4),'ro'), hold off
title('FracMT of  $\text{bfU}^{-1}_{\log}$   $\text{rm}$  of  $\text{ite}^{-\pi t^2}$ ')
FracFT_s4_th = exp(-pi * (t2).^2);
figure(3), plot(t2,abs(FracFT_s4_th));
title('Theoretical FracFT of  $\text{ite}^{-\pi t^2}$ ');xlabel('t2')
hold on, plot(t2, FracMT_U_inv_s4,'ro'),hold off

FracFT_s4_th = exp(-pi * (t).^2);
figure(6), plot(t,abs(FracFT_s4_th));
title('Theoretical FracFT of  $\text{ite}^{-\pi t^2}$ ');xlabel('t')
hold on, plot(t2, abs(FracMT_U_inv_s4),'ro'),hold off
pause

%2 ----- constant term -----

c = 5;
s2 = c * ones(size(t2));
figure(4), plot(t2, real(s2)), title('s2')
U_inv_log_s2 = c ./ sqrt(t);
figure(5), pltcplx(t,(U_inv_log_s2)), title('\bfU^{-1}_{\log} \rm \itc')

phi = pi /5;
a = phi * 2 / pi;

% FracFT of s2
FracFT_s2 = fracf(s2,a);
figure(1), plot(t2,real(FracFT_s2))
title('FracFT of \itc')
%FracMT of U_inv_log_s2
FracMT_U_inv_s2 = fracf(logwarp3(U_inv_log_s2),a);

```

```

figure(2), plot(t2,real(FracMT_U_inv_s2))
hold on, plot(t2, real(FracFT_s2),'ro'), hold off
title('FracMT of \bfU^{-1}_{\log} \rm of \itc')
FracFT_s2_th = c .* sqrt(1 + j * cot(phi)) .* exp(-j * pi *
((t2).^2) * tan(phi));
figure(3), plot(t2, real(FracFT_s2_th));xlabel('t2')
title('Theoretical FracFT of \itc');
hold on, plot(t2, FracMT_U_inv_s2,'ro'),hold off

FracFT_s2_th = c .* sqrt(1 + j * cot(phi)) .* exp(-j * pi *
((t).^2) * tan(phi));
figure(6), plot(t, real(FracFT_s2_th));
title('Theoretical FracFT of \itc');xlabel('t')
hold on, plot(t2, FracMT_U_inv_s2,'ro'),hold off

```

A.16 Hypchirptest.m – Used for Finding the FrMT of the Hyperbolic Chirp using Logwarp3.m

```

N = 2^12;

% n2 = nearest(N) = 2*(log(N/8))^2 %

phi = pi / 2.1;
% phi = acot(0.05);
r = 1;
t = fracfsamples(N) + 1e-1;
%t = t - t(1) + 1e-1;
t2 = fracfsamples(nearest(N)) + 1e-1;
%t2 = t2 - t2(1) + 1e-1;
% t=sample(0,1,512);
nzi = N/2+1 : N; %nonzero index
mt = -pi * cot(phi) * (log(t)).^2;
b = 2 * pi * r / sin(phi) * log(t);
sphi = 1./sqrt(t) .* exp(j*mt + j * b);
lw3sphi = logwarp3(sphi);

```

```

    figure(1),          plot(t(nzi),real(sphi(nzi))),          %
    title('real(sphi)')

    figure(2),          plot(t2,abs(fracf(lw3sphi,phi*2/pi))),  %
    title(['fracf logwarp sphi, \phi'...    ,'\pi/5']);

    ulog_sphi_th        =      exp(-j*pi*cot(phi).*(t2).^2      +
    j*2*pi*r/sin(phi)*t2);

    figure(5),          plot(t2,abs(fracf(ulog_sphi_th,phi*2/pi))); %
    title(['fracf ulog\_sphi\_th \phi={\pi}/{5}'])

    figure(6),          plot(t2,abs(fracf(ulog_sphi_th,pi/3*2/pi))); %
    title(['fracf ulog\_sphi\_th \phi={\pi}/{3}'])

    figure(10),         plot(t2,abs(fracf(lw3sphi,pi/3*2/pi))), %
    title(['fracf logwarp sphi, \phi'...    ,'\pi/3']);

    %figure(7),          plot(t,abs(fracf(sphi,phi*2/pi)));      %
    title(['fracf sphi \phi={\pi}/{5}'])

    %figure(8), plot(t,abs(fracf(sphi,1))); % title(['fracf sphi
    \phi={\pi}/{2}'])

    figure(9)
    subplot(3,1,1), plot(t(nzi),real(sphi(nzi)))
    subplot(3,1,2), plot(t2,real(ulog_sphi_th)), xlim([t2(1)
    t2(end)])
    subplot(3,1,3), plot(t2, real(lw3sphi)), xlim([t2(1)
    t2(end)])

```