

# **Study of color chaos model and Time frequency analysis of S&P 500 and NASDAQ indexes**

by

Prabaharan Sivashanmugam

A Project submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science  
(Department of Mathematics and Statistics)  
in The University of Michigan, Dearborn  
2016

Advisor:

Professor Frank Massey, Chair



© Prabakaran Sivashanmugam 2016

---

All Rights Reserved

Dedicated to my wife Raji Praba, my sons, Deepak Praba and Darshan Praba

## ACKNOWLEDGEMENTS

Immeasurable appreciation and deepest gratitude for the help and support are extended to the following persons who in one way or another have contributed in making this study possible.

Prof. Paul Watta, University of Michigan - Dearborn, for introducing Gabor transformation concept to me and provided base foundational insights on Gabor applications in engineering field.

Prof. Frank Massey, University of Michigan - Dearborn, for his guidance and coaching to complete the project.

# TABLE OF CONTENTS

<b>DEDICATION</b> . . . . .	ii
<b>ACKNOWLEDGEMENTS</b> . . . . .	iii
<b>LIST OF FIGURES</b> . . . . .	vi
<b>LIST OF TABLES</b> . . . . .	xii
<b>LIST OF APPENDICES</b> . . . . .	xiii
<b>LIST OF ABBREVIATIONS</b> . . . . .	xiv
<b>ABSTRACT</b> . . . . .	xv
<b>CHAPTER</b>	
<b>I. Wigner Distribution</b> . . . . .	1
1.1 Wigner Distribution . . . . .	1
1.2 Wigner-Ville Distribution . . . . .	3
1.3 Discrete Wigner Distribution . . . . .	3
<b>II. Time Frequency Distribution Series</b> . . . . .	7
2.1 Time Frequency Distribution Series . . . . .	7
<b>APPENDICES</b> . . . . .	13
C.1 Gabor Elementary function . . . . .	43
C.1.1 Proof: GEF has minimum uncertainty in the time- frequency domain . . . . .	44
<b>BIBLIOGRAPHY</b> . . . . .	49

# LIST OF FIGURES

## Figure

1.1	Wigner Distribution transform for $\log(s)$ ; where $s$ denotes the sp500. Top row is the $\log(s)$ , Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of $\log(s)$ with $\lambda = 1600$ and its Wigner Distribution transform. The graph was created using Matlab code mywvd.m and it is attached in the Appendix B . . . . .	5
1.2	Wigner Distribution transform for $\log(s)$ ; where $s$ denotes the NASDAQ index. Top row is the $\log(s)$ , Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of $\log(s)$ with $\lambda = 1600$ and its Wigner Distribution transform. The graph was created using Matlab code mywvd.m and it is attached in the Appendix B . . . . .	6
2.1	Fig in the top represents the Gaussian function $g(t)$ as defined above with $\sigma = 0.1$ . Fig in the bottom represents the Wigner Ville Distribution $WVD_g$ as defined above. The graph is created using the mywvdgauss.m program attached in the Appendix. $WVD$ values are created by using the HOSA (Higher Order Spectral Analysis) Matlab toolbox. . . . .	8

## LIST OF TABLES

Table



## LIST OF APPENDICES

### Appendix

A.	R code that are used to do analysis . . . . .	14
B.	Matlab code that are used to do analysis . . . . .	22
C.	Mathematical Proof . . . . .	43

## LIST OF ABBREVIATIONS

# ABSTRACT

Study of color chaos model and Time frequency analysis of S&P 500 and NASDAQ indexes

by

Prabaharan Sivashanmugam

Advisor: Professor Frank Massey

Change the abstract based on the finding of the report.. as of now, it is replication of the Chen's abstract Time frequency model and random walk model are two polar models in linear systems. Color chaos is a model that is in between these models which generates irregular oscillation with a narrow frequency band. The deterministic component from noisy data can be recovered by time variant filter in Gabor space. The characteristic frequency is calculated by Wigner decomposed distribution series. It is found that 7% of the detrended by HP filter can be explained by the deterministic color chaos. The existence of persistent chaotic cycle reveals a new perspective of market resilience and new sources of economic uncertainties. The nonlinear pattern in the stock market may not be wiped out by the market competition under non-equilibrium situations with trend evolution and frequency shifts.

# CHAPTER I

## Introduction

## CHAPTER II

### Gabor Transformation

## CHAPTER III

### Wigner Distribution

#### 3.1 Wigner Distribution

The Wigner Distribution (WD) introduced by Wigner (Wigner 1932) as a phase representation in Quantum Mechanics gives a simultaneous representation of a signal in space and spatial-frequency variables. WD belongs to a large class of bilinear distributions known as the Cohen's class, in which each member can be obtained choosing a different kernel in the generalized bilinear distribution definition.

Let's suppose  $f(t)$  is continuous, integrable and complex function. The symmetric definition of the Wigner distribution  $W_f(t, \omega)$  is given by

$$W_f(t, \omega) = \int_{-\infty}^{\infty} f\left(t + \frac{\tau}{2}\right) f^*\left(t - \frac{\tau}{2}\right) e^{-j\omega\tau} d\tau \quad (3.1)$$

where  $t$  and  $\tau$  are spatial variables,  $\omega$  is the spatial frequency variable and  $f^*$  is the complex conjugate of  $f$ . The product function  $r_f(t, \tau)$  is given by

$$r_f(t, \omega) = f\left(t + \frac{\tau}{2}\right) f^*\left(t - \frac{\tau}{2}\right) \quad (3.2)$$

The auto-Wigner distribution gives a generalized auto convolution at non-zero frequency. From  $W_f(t, \omega)$ , it can be observed that the Wigner Distribution is the Fourier transformation, for a given point  $\tau$ , of the product  $f f^*$ . It may also be obtained from

the Fourier transform,  $F$  of  $f$  by

$$W_F(\omega, t) = \int_{-\infty}^{\infty} F(\omega + \frac{\phi}{2}) F^*(\omega - \frac{\phi}{2}) e^{j\phi t} d\phi \quad (3.3)$$

where  $F^*$  is the complex conjugate of  $F$ . Connecting to  $W_f(t, \omega)$  and  $W_F(\omega, t)$  the following relation is observed,

$$W_f(t, \omega) = W_F(\omega, t) \quad (3.4)$$

which shows the symmetry between the two conjugate domains. Among various properties of the WD, the interference and inversion properties are most relevant for the current study. The WD computation introduces a spurious "auto terms" due to its intrinsic bi-linearity. The WD of sum of two signals  $f(t) + f'(t)$  is given by

$$W_{f+f'}(t, \omega) = W_f(t, \omega) + W_{f'}(t, \omega) + 2Re[W_{f,f'}(t, \omega)] \quad (3.5)$$

Inversion of the original signal  $f(t)$  from the Wigner Distribution is given by

$$f(t + \frac{\tau}{2}) f^*(t - \frac{\tau}{2}) = \int_{-\infty}^{\infty} W_f(t, \omega) e^{j\omega\tau} d\omega \quad (3.6)$$

Let  $t = \frac{\tau}{2}$  and then setting  $\tau = t$ , we have

$$f(t) f^*(0) = \int_{-\infty}^{\infty} W_f(\frac{t}{2}, \omega) e^{j\omega t} d\omega \quad (3.7)$$

$$f(t) = \frac{1}{f^*(0)} \int_{-\infty}^{\infty} W_f(\frac{t}{2}, \omega) e^{j\omega t} d\omega \quad (3.8)$$

### 3.2 Wigner-Ville Distribution

The Wigner-Ville distribution (WVD) is defined for a signal  $f(t)$  as follows:

$$W_f(t, \omega) = \int_{-\infty}^{\infty} z(t + \frac{\tau}{2}) z^*(t - \frac{\tau}{2}) e^{-j\omega\tau} d\tau \quad (3.9)$$

where  $z(t)$  is the analytic associative of  $f(t)$ . The Wigner distribution is simply defined when the real signal  $f(t)$  is used instead of the analytic one  $z(t)$ . A signal  $f(t)$  is said to be analytical if and only if

$$F(\omega) = 0 \quad (3.10)$$

for all  $\omega < 0$ , where  $F(\omega)$  is Fourier transform of  $f(t)$ . In other words, an analytical signal contains no negative frequencies. It may have a spectral component at zero frequency.

### 3.3 Discrete Wigner Distribution

One of the main disadvantages of the discrete definition is that not all the properties of the continuous WD are preserved by discretization due to aliasing effects. Several alternative definitions have been proposed in the literature in order to overcome this problem (Chan 1982), (Claasen 1983), (Brenner 1983), (Day 1983), (Peyrin 1986). The discrete WD of a sampled function  $f(t)$  is defined by

$$W_f(n, m) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} f(n+k) f^*(n-k) e^{-j(\frac{2\pi mk}{N+1})} \quad (3.11)$$

where  $n$  and  $m$  are the spatial and frequency variables respectively.

The discrete WD definition given above retain the basic properties of the continuous WD, however, in the main differences comes from the inversion property.



Inversion property is the ability to extract the time domain signal from distribution, up to a constant phase factor and proposed distribution satisfies this property as

$$f(n+k)f^*(n-k) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} W_f(n, m) e^{j(\frac{2\pi mk}{N+1})} \quad (3.12)$$

In the case of discrete signals, inserting  $k = n$  in the above equation allows one to write

$$f(2n)f^*(0) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} W_f(n, m) e^{j(\frac{2\pi mn}{N+1})} \quad (3.13)$$

From the above equation only the even samples can be recovered. Inserting  $k-1 = n$  in the discrete WD of sampled function  $f(t)$ ,

$$f(2n-1)f^*(1) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} W_f(n, m) e^{j(\frac{2\pi mn}{N+1})} \quad (3.14)$$

leads to the recovering of the odd samples.

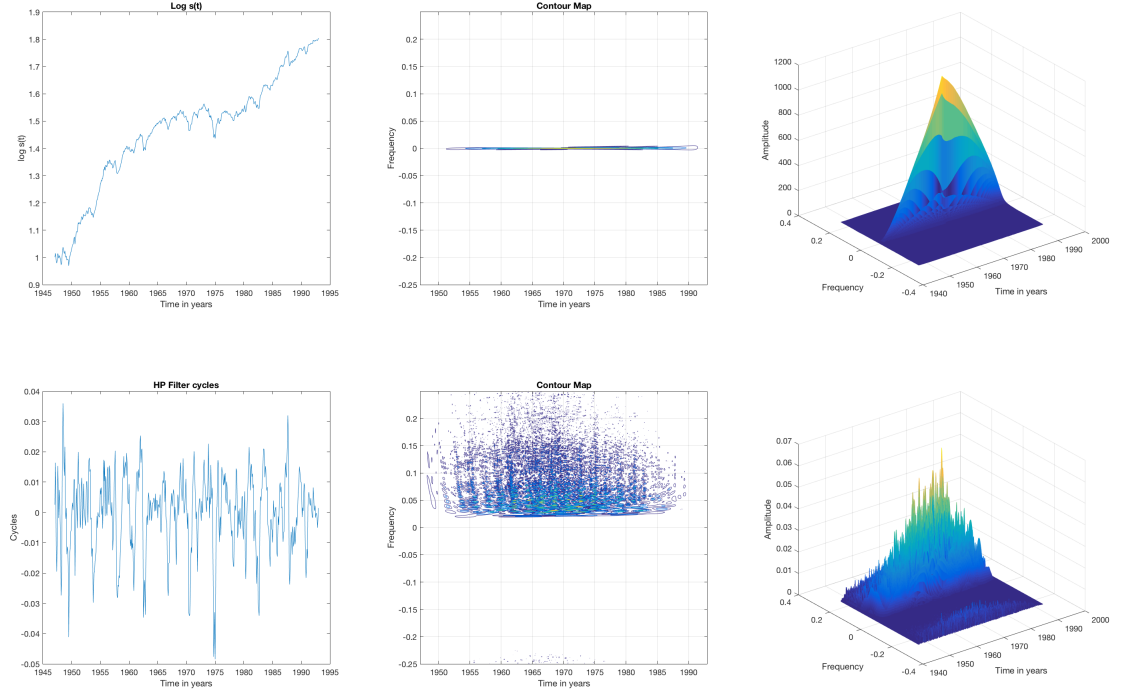


Figure 3.1: Wigner Distribution transform for  $\log(s)$ ; where  $s$  denotes the sp500. Top row is the  $\log(s)$ , Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of  $\log(s)$  with  $\lambda = 1600$  and its Wigner Distribution transform. The graph was created using Matlab code `mywvd.m` and it is attached in the Appendix B

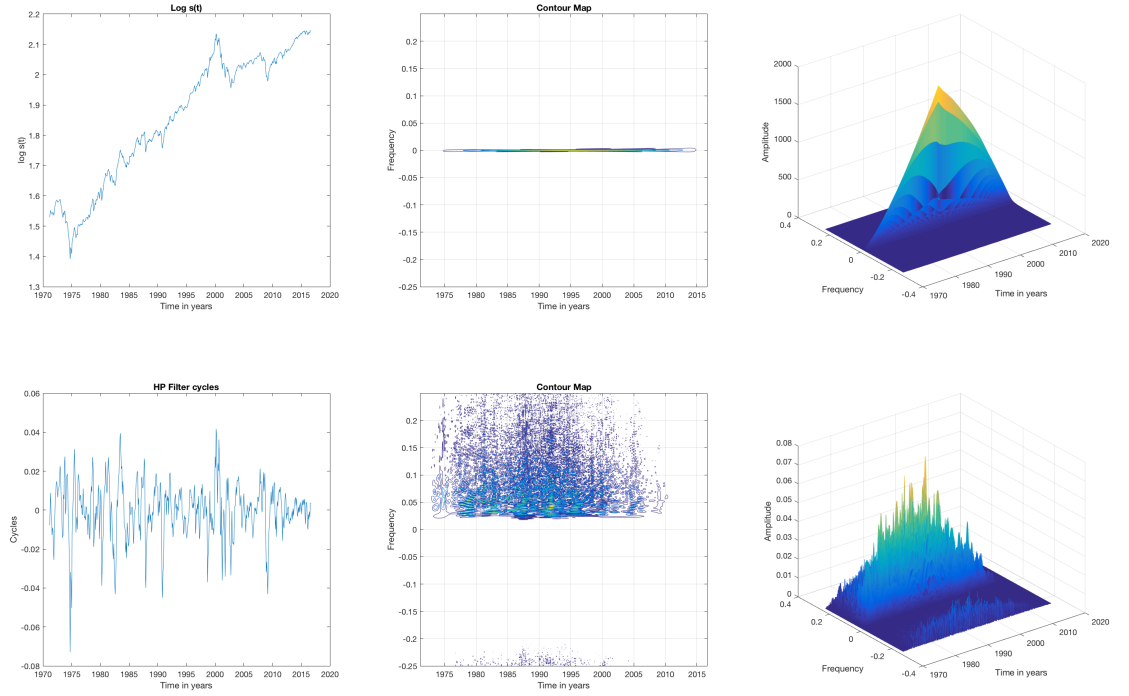


Figure 3.2: Wigner Distribution transform for  $\log(s)$ ; where  $s$  denotes the NASDAQ index. Top row is the  $\log(s)$ , Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of  $\log(s)$  with  $\lambda = 1600$  and its Wigner Distribution transform. The graph was created using Matlab code `mywvd.m` and it is attached in the Appendix B

## CHAPTER IV

### Time Frequency Distribution Series

#### 4.1 Time Frequency Distribution Series

The main drawback of the Wigner Ville distribution is cross term interference and the cross term oscillates and is localized.

Time Frequency Distribution Series (TFDS) was introduced by Chen and Qian [2] as the decomposition of the Wigner Ville distribution via the orthogonal like Gabor expansion. Let me walk through each step to attain the time frequency distribution series.

Let  $g(t)$  be a normalized Gaussian function which is defined as follows.

$$g(t) = \frac{1}{(\pi\sigma^2)^{0.25}} e^{-\frac{t^2}{2\sigma^2}} \quad (4.1)$$

The corresponding Wigner Ville Distribution (WVD) is given below.

$$WVD_g(t, \omega) = 2e^{-\left(\frac{t^2}{\sigma^2} + \sigma^2\omega^2\right)} \quad (4.2)$$

The  $WVD_g(t, \omega)$  is centered at origin and it decays exponentially in both time and frequency domains. The contour plot of the  $WVD_g(t, \omega)$  consists of concentric ellipses and it is given below.

WVD is time and frequency-shift invariant. Let

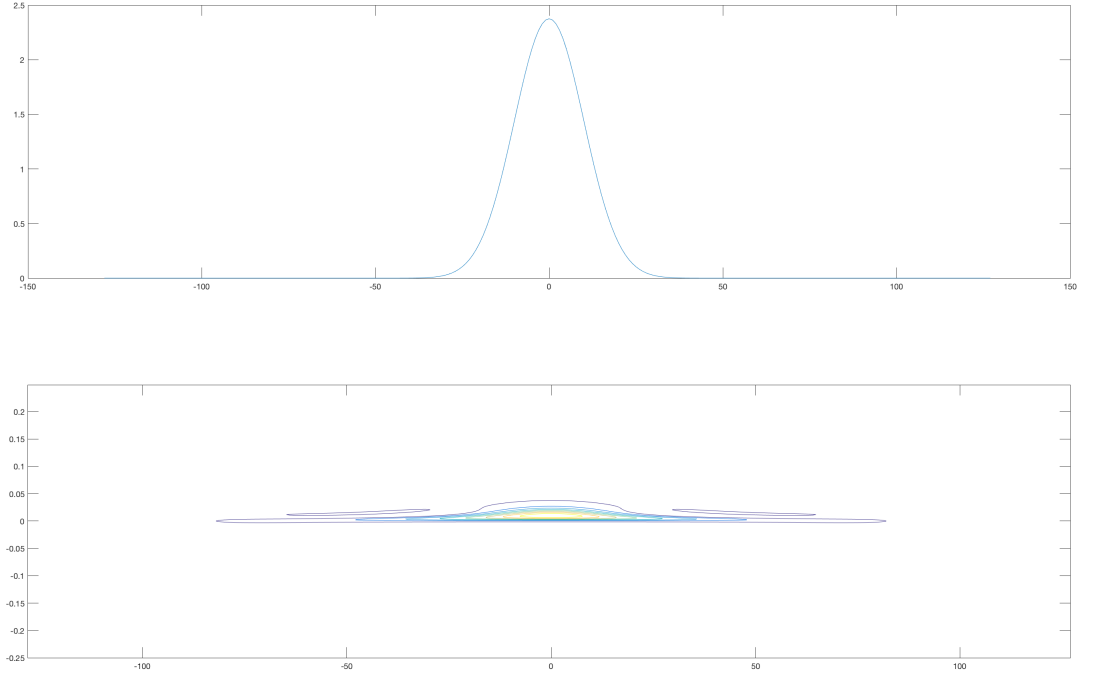


Figure 4.1: Fig in the top represents the Gaussian function  $g(t)$  as defined above with  $\sigma = 0.1$ . Fig in the bottom represents the Wigner Ville Distribution  $WVD_g$  as defined above. The graph is created using the `mywvdgauss.m` program attached in the Appendix.  $WVD$  values are created by using the HOSA (Higher Order Spectral Analysis) Matlab toolbox.

$$h(t) = g(t - mT)e^{-jn\Omega t} \quad (4.3)$$

where,  $\Omega$  and  $T$  are the frequency and time sampling steps respectively. The WVD of  $h(t)$  is given by,

$$WVD_h(t, \omega) = 2e^{-[\frac{(t-mT)^2}{\sigma^2} + [\sigma(\omega - n\Omega)]^2]} \quad (4.4)$$

$$WVD_h(t, \omega) = WVD_g(t - mT, \omega - n\Omega) \quad (4.5)$$

Let  $s(t) = h(t) + h'(t)$ , then  $WVD_s(t, \omega)$  is given as

$$WVD_s(t, \omega) = WVD_h(t, \omega) + WVD_{h'}(t, \omega) + 2Re[WVD_{h,h'}(t, \omega)] \quad (4.6)$$

where  $WVD_{h,h'}(t, \omega)$  is given by

$$WVD_{h,h'}(t, \omega) = e^{j\omega_d t_\mu} H(t - t_\mu, \omega - \omega_\mu) \quad (4.7)$$

where

$$H(t, \omega) = 2e^{-[\frac{t^2}{\sigma^2} + (\sigma\omega)^2]} e^{-j[t_d\omega - \omega_d t]} \quad (4.8)$$

$$t_\mu = \frac{m + m'}{2}T, \omega_\mu = \frac{n + n'}{2}\Omega, t_d = (m - m')T, \omega_d = (n - n')\Omega \quad (4.9)$$

The  $WVD_{h,h'}(t, \omega)$  has the same envelop as the  $WVD_g(t, \omega)$  but is oscillating with  $\omega_d$  in the time domain and  $t_d$  in the frequency domain. The location of  $WVD_{h,h'}(t, \omega)$  is halfway between  $h$  and  $h'$ . The  $2Re[WVD_{h,h'}(t, \omega)]$  is the cross term. When a signal  $s(t)$  can be decomposed as a linear combination of some elementary functions  $h(t)$  then the cross-terms can be controlled.

Recall from the previous chapter on the Gabor Expansion, for a given signal  $s(t)$ , the orthogonal-like Gabor expansion is defined as follows.

$$s(t) = \frac{1}{\sqrt{2\pi}} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} C_{m,n} g(t - mT) e^{jn\Omega t} \quad (4.10)$$

The Gabor coefficients  $C_{m,n}$  are determined by

$$C_{m,n} = \int s(t) \gamma_{m,n}^*(t) dt = \int s(t) \gamma^*(t - mT) e^{-jn\Omega t} dt = STFT(mT, n\Omega) \quad (4.11)$$

Using the Wigner-Ville distribution of  $s(t)$  from the above equation yields,

$$WVD_s(t, \omega) = \sum_{m,n} \sum_{m',n'} C_{m,n} C_{m',n'} WVD_{h,h'}(t, \omega) \quad (4.12)$$

where

$$WVD_{h,h'}(t, \omega) = e^{j\omega_d t_\mu} 2e^{-[\frac{t^2}{\sigma^2} + (\sigma\omega)^2]} e^{-j[t_d\omega - \omega_d t]} \quad (4.13)$$

$$WVD_s(t, \omega) = \sum_{m,n} \sum_{m',n'} C_{m,n} C_{m',n'} e^{j\omega_d t_\mu} 2e^{-[\frac{t^2}{\sigma^2} + (\sigma\omega)^2]} e^{-j[t_d\omega - \omega_d t]} \quad (4.14)$$

where  $t_d$  and  $\omega_d$  reflect the degree of oscillation.

When  $m = m'$  and  $n = n'$ ,

$$C_{m,n} C_{m',n'}^* WVD_{h,h'}(t, \omega) = 2|C_{m,n}|^2 e^{-[\frac{(t-mT)^2}{\sigma^2} + \sigma^2(\omega - n\Omega)^2]} \quad (4.15)$$

When  $m \neq m'$  or  $n \neq n'$

$$C_{m,n} C_{m',n'}^* WVD_{h,h'}(t, \omega) = C_{m,n} C_{m',n'}^* e^{j\omega_d t_\mu} H(t - t_\mu, \omega - \omega_\mu) \quad (4.16)$$

Based on the decomposition of the Wigner-Ville distribution defined above, the Time Frequency Distribution Series (TFDS) is defined as follows.

$$TFDS_D(t, \omega) = \sum_{d=0}^D P_d(t, \omega) \quad (4.17)$$

Here  $P_d(t, \omega)$  is the sum of a sequence of  $WVD_{h,h'}(t, \omega)$  which have a similar contribution to the useful properties and similar influence to the cross terms in which  $|m - m'| + |n - n'| = d$

$$P_d(t, \omega) = \sum_{|m-m'|+|n-n'|=d} C_{m,n} C_{m',n'}^* WVD_{h,h'}(t, \omega) \quad (4.18)$$

Substituting the value of  $WVD_{h,h'}(t, \omega)$  in the above equation, we get

$$P_d(t, \omega) = \sum_{|m-m'|+|n-n'|=d} C_{m,n} C_{m',n'}^* e^{j\omega_d t_\mu} 2e^{-[\frac{t^2}{\sigma^2} + (\sigma\omega)^2]} e^{-j[t_d\omega - \omega_d t]} \quad (4.19)$$

Substituting the value of  $t_d$ ,  $\omega_d$ ,  $t_\mu$  and  $\omega_\mu$ , we get

$$P_d(t, \omega) = \sum_{|m-m'|+|n-n'|=d} C_{m,n} C_{m',n'}^* e^{j\frac{n+n'}{2}\Omega(m-m')T} 2e^{-[\frac{t^2}{\sigma^2} + (\sigma\omega)^2]} e^{-j[(m-m')T\omega - (n-n')\Omega t]} \quad (4.20)$$

Both the SP500 and NASDAQ indices are discrete signals used for analysis. The  $P_d$  is further simplified for programming convenience.

The discrete Time Frequency Distribution Series is defined as follows.

$$DTFDS_D[i, k] = TFDSD_d(t, \omega)|_{t=i\Delta t, \omega=\frac{2\pi k}{L\Delta t}} \quad (4.21)$$

for  $-\frac{L}{2} \leq k < \frac{L}{2}$

where  $\frac{1}{\Delta t}$  denotes the sampling frequency.  $L$  denotes the length of the signal. The discrete time frequency distribution series can be summarized as

$$TFDSD_d(i, k) = \sum_{d=0}^D P_d[i, k] \quad (4.22)$$

where

$$P_d[i, k] = \sum_{|m-m'|+|n-n'|=d} C_{m,n} C_{m',n'}^* WVD_{h,h'}[i, k] \quad (4.23)$$

The  $TFDSD_D[i, k]$  is the sum of all  $WVD_{h,h'}[i, k]$  in which the distance of the



corresponding Gabor elementary functions  $h_{m,n}[i]$  and  $h_{m',n'}[i]$  is less than or equal to D.  $WVD[i, k]$  is defined as a sampled Wigner-Ville distribution.

$$WVD_s[i, k] = WVD_s(t, \omega)|_{t=i\Delta t, \omega=\frac{2\pi k}{L\Delta t}} \quad (4.24)$$

where  $\Delta t$  denotes the sampling interval. For the Gaussian functions, WVD is obtained by sampling the formula.

$$WVD_{h,h'}[i, k] = 2e^{-\sigma(i-\frac{m+m'}{2}\Delta M)^2 - \frac{1}{\sigma}(k-\frac{n+n'}{2}\Delta N)^2} e^{j\frac{2\pi}{L}[(m-m')\Delta M k + (n-n')\Delta N i - \frac{n+n'}{2}\Delta N(m-m')\Delta M]} \quad (4.25)$$

We assume  $\Delta t = 1$ .  $WVD_{h,h'}[i, k]$  in ?? is completely determined by the parameters of the Gabor expansion, such as  $\Delta M, \Delta N, L$  and  $\sigma$  which are independent of the analyzed signal. Therefore, once  $\Delta M, \Delta N, L$  and  $\sigma$  are determined,  $WVD_{h,h'}[i, k]$  can be precomputed and saved in a table.

## CHAPTER V

### Color Chaos Model

## APPENDICES

## APPENDIX A

### R code that are used to do analysis

```
1 fdplot <-function()
2 {
3   #Program used to create the Difference stationary for NASDAQ & SP500 indexes
4   # Praba Siva
5   # praba@umich.edu
6   # @prabasiva
7   layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
8   setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
9   fspcom=read.table('fspcom.dat')
10  dat = log(fspcom[,5])
11  year=fspcom[,2]+1/12*fspcom[,3]
12  t1=dat[1:length(dat)-1]
13  t2=dat[2:length(dat)]
14  plot(year[1:length(year)-1],t2-t1,type='l',
15        main="Difference stationary of first Differencing of log(x(t))\nx(t) = S&P 500",
16        xlab="Year",ylab="FD",col='blue',
17        ylim=c(-.2,.2),cex.axis=1.1,cex.lab=1.5,lwd=2.2)
18  setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
19  dat <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
20  year=dat[,1]+1/12*dat[,2]
21  dat=log(dat[,3])
22  t1=dat[1:length(dat)-1]
23  t2=dat[2:length(dat)]
24  plot(year[1:length(year)-1],t2-t1,type='l',
25        main="Difference stationary of first Differencing of log(x(t))\nx(t) = NASDAQ",
26        xlab="Year",ylab="FD",col='red',
27        ylim=c(-.2,.2),cex.axis=1.1,cex.lab=1.5,lwd=2.2)
28
29 }
```

```

1  llt <-function ()
2  {
3    #Program used to Log linear trend and cycles for SP500 & NASDAQ index
4    # Praba Siva
5    # praba@umich.edu
6    # @prabasiva
7    setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
8    fspcom=read.table('fspcom.dat')
9    year=fspcom[,2]
10   tsfspcom=ts(log(fspcom[,5]),start=year[1],
11               end=c(year[length(year)],12),frequency=12)
12   loglinear=stl(log(tsfspcom),s.window=5)
13   plot(loglinear,main="A Seasonal-Trend Decomposition of S&P 500",
14        col='red')
15   setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
16   dat <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
17   year=dat[,1]
18   dat=dat[,3]
19   tsnasdaq=ts(dat,start=year[1],
20              end=c(year[length(year)]-1,12),frequency=12)
21   nloglinear=stl(log(tsnasdaq),s.window=5)
22   plot(nloglinear,main="A Seasonal-Trend Decomposition of NASDAQ",
23        col="blue")
24   strend=loglinear$time.series[,2]
25   ntrend=nloglinear$time.series[,2]
26   plot(year[1:length(strend)],strend[1:length(strend)],
27        col='blue',type='l',ylim=range(strend,ntrend))
28   lines(year[1:length(ntrend)],ntrend[1:length(ntrend)],
29         col='red',type='l')
30 }

```

```

1
2  hpfilt <- function()
3  {
4    #Program used to create the HP filter for lambda 80 & 800 for S&P 500 index
5    #Load the file and invoke hpfilt()
6    #Filename: hpfilter_slave.R
7    # Praba Siva;praba@umich.edu; @prabasiva
8    library(mFilter)
9    library(latex2exp)
10   opar <- par(no.readonly=TRUE)
11   setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
12   fspcom=read.table('fspcom.dat')
13   dat = fspcom[,5]
14   syear=fspcom[,2]+1/12*fspcom[,3]
15   ldat=log(dat)
16   dat=ldat
17   dat.hp1 <- hpfilter(dat, freq=80,type="frequency",drift=FALSE)
18   dat.hp2 <- hpfilter(dat, freq=800,type="frequency",drift=FALSE)
19   par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
20   plot(syear,dat, ylim=range(dat),
21        main="S&P 500 Index ",

```

```

22     col=2, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
23 plot(syear,dat.hp1$trend, ylim=range(dat.hp1$trend),
24      main="HP filter of S&P 500 Index: Trend,Lambda=80 ",
25      col=4, xlab='praba siva', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
26 plot(syear,dat.hp1$cycle, ylim=range(dat.hp1$cycle),
27      main="HP filter of S&P 500 Index: Cycle,Lambda=80 ",
28      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
29 par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
30 plot(syear,dat, ylim=range(dat),
31      main="S&P 500 Index ",
32      col=2, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
33 plot(syear,dat.hp2$trend, ylim=range(dat.hp1$trend),
34      main="HP filter of S&P 500 Index: Trend,Lambda=800 ",
35      col=4, xlab='praba siva', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
36 plot(syear,dat.hp1$cycle, ylim=range(dat.hp2$cycle),
37      main="HP filter of S&P 500 Index: Cycle,Lambda=800 ",
38      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
39 par(opar)
40 opar <- par(no.readonly=TRUE)
41 setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
42 nasda <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
43 nyear=nasda[,1]+1/12*nasda[,2]
44 nasda=nasda[,3]
45 lnasda=log(nasda)
46 nasda=lnasda
47 nasda.hp1 <- hpfiler(nasda, freq=80,type="frequency",drift=FALSE)
48 nasda.hp2 <- hpfiler(nasda, freq=800,type="frequency",drift=FALSE)
49 par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
50 plot(nyear,nasda, xlab="Year",ylab="log s(t)",ylim=range(nasda),
51      main="NASDAQ Index ",
52      col=2, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
53 plot(nyear,nasda.hp1$trend, xlab='Year', ylim=range(nasda.hp1$trend),
54      main="HP filter of NASDAQ Index: Trend,Lambda=80 ",
55      col=4, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
56 plot(nyear,nasda.hp1$cycle, ylim=range(nasda.hp1$cycle), xlab="Year",
57      main="HP filter of NASDAQ Index: Cycle,Lambda=80 ",
58      col=3, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
59 par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
60 plot(nyear,nasda, ylim=range(nasda),
61      main="NASDAQ Index ",
62      col=2, ylab="log s(t)",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
63 plot(nyear,nasda.hp2$trend, ylim=range(nasda.hp1$trend),
64      main="HP filter of NASDAQ Index: Trend,Lambda=800 ",
65      col=4, xlab='Year', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
66 plot(nyear,nasda.hp1$cycle, ylim=range(nasda.hp2$cycle),
67      main="HP filter of NASDAQ Index: Cycle,Lambda=800 ",
68      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
69 par(opar)
70 nasda.hp3 <- hpfiler(nasda, freq=1600,type="frequency",drift=FALSE)
71 nasda.hp4 <- hpfiler(nasda, freq=14400,type="frequency",drift=FALSE)
72 lambda=c(80,800,1600,14400);
73 c=1:4;
74 layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
75 plot(nyear, nasda.hp1$trend,ylab='Log NASDAQ',
76      main=TeX('NASDAQ Trend HP filter with different $\lambda$'),
77      xlab='Year',col=c(1),type='l',cex.axis=1,cex.lab=1.3,lwd=2.2);

```

```

78 lines(nyear, nasda.hp2$trend, main=TeX('NASDAQ Trend HP filter with different  $\lambda$ '),
79       col=c(2), type='l', cex.axis=1, cex.lab=1.1, lwd=2.2);
80 lines(nyear, nasda.hp3$trend, main=TeX('NASDAQ Trend HP filterwith different  $\lambda$ '),
81       col=c(3), type='l', cex.axis=1, cex.lab=1.1, lwd=2.2);
82 lines(nyear, nasda.hp4$trend, main=TeX('NASDAQ Trend HP filterwith different  $\lambda$ '),
83       col=c(4), type='l', cex.axis=1, cex.lab=1.1, lwd=2.2);
84 legend('topleft', legend=TeX(sprintf(" $\lambda = %d$ ", lambda)), lwd=1, col=c)
85 dat.hp3 <- hpfiler(dat, freq=1600, type="frequency", drift=FALSE)
86 dat.hp4 <- hpfiler(dat, freq=14400, type="frequency", drift=FALSE)
87 lambda=c(80, 800, 1600, 14400);
88 c=1:4;
89 plot(syear, dat.hp1$trend, ylab='Log SP500', main=TeX('SP500 Trend HP filter with different  $\lambda$ '),
90       xlab='Year', col=c(1), type='l', cex.axis=1, cex.lab=1.3, lwd=2.2);
91 lines(syear, dat.hp2$trend, main=TeX('SP500 Trend HP filter with different  $\lambda$ '),
92       col=c(2), type='l', cex.axis=1.1, cex.lab=1, lwd=2.2);
93 lines(syear, dat.hp3$trend, main=TeX('SP500 Trend HP filterwith different  $\lambda$ '),
94       col=c(3), type='l', cex.axis=1.1, cex.lab=1, lwd=2.2);
95 lines(syear, dat.hp4$trend, main=TeX('SP500 Trend HP filterwith different  $\lambda$ '),
96       col=c(4), type='l', cex.axis=1.1, cex.lab=1, lwd=2.2);
97 legend('topleft', legend=TeX(sprintf(" $\lambda = %d$ ", lambda)), lwd=1, col=c)
98
99 #Statistics of the HP Detrending
100 print('SP500 ')
101 print('HP Filter with Lambda = 80')
102 sprintf("Mean = %5f", mean(dat.hp1$cycle))
103 sprintf("SD = %5f", sd(dat.hp1$cycle))
104 sprintf("Variance = %5f", var(dat.hp1$cycle))
105 print('HP Filter with Lambda = 800')
106 sprintf("Mean = %5f", mean(dat.hp2$cycle))
107 sprintf("SD = %5f", sd(dat.hp2$cycle))
108 sprintf("Variance = %5f", var(dat.hp2$cycle))
109 print('HP Filter with Lambda = 1600')
110 sprintf("Mean = %5f", mean(dat.hp3$cycle))
111 sprintf("SD = %5f", sd(dat.hp3$cycle))
112 sprintf("Variance = %5f", var(dat.hp3$cycle))
113 print('HP Filter with Lambda = 14400')
114 sprintf("Mean = %5f", mean(dat.hp4$cycle))
115 sprintf("SD = %5f", sd(dat.hp4$cycle))
116 sprintf("Variance = %5f", var(dat.hp4$cycle))
117
118 print('NASDAQ ')
119 print('HP Filter with Lambda = 80')
120 xm=sprintf("Mean = %5f", mean(nasda.hp1$cycle))
121 xs=sprintf("SD = %5f", sd(nasda.hp1$cycle))
122 xv=sprintf("Variance = %5f", var(nasda.hp1$cycle))
123
124 print('HP Filter with Lambda = 800')
125 sprintf("Mean = %5f", mean(nasda.hp2$cycle))
126 sprintf("SD = %5f", sd(nasda.hp2$cycle))
127 sprintf("Variance = %5f", var(nasda.hp2$cycle))
128
129 print('HP Filter with Lambda = 1600')
130 sprintf("Mean = %5f", mean(nasda.hp3$cycle))
131 sprintf("SD = %5f", sd(nasda.hp3$cycle))
132 sprintf("Variance = %5f", var(nasda.hp3$cycle))

```

```

133
134 print('HP Filter with Lambda = 14400')
135 sprintf("Mean = %5f",mean(nasda.hp4$cycle))
136 sprintf("SD = %5f",sd(nasda.hp4$cycle))
137 sprintf("Variance = %5f",var(nasda.hp4$cycle))
138
139 }

```

```

1 loglinear <-function()
2 {
3   #Program used to Log linear example
4   #Filename:loglinear.R
5   # Praba Siva
6   # praba@umich.edu
7   # @prabasiva
8   layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
9   t=0:50000;
10  plot(t,exp(t*-.0001),main=TeX('$\\beta_1 < 0$'),
11        xlab='Time t', ylab=TeX('log Y = $\\beta_0 + \\beta_1 t$'),
12        type='l',cex.axis=1.1,cex.lab=.9,lwd=3,col='red')
13  plot(t,exp(t*.0001),main=TeX('$\\beta_1 > 0$'),
14        xlab='Time t', ylab=TeX('log Y = $\\beta_0 + \\beta_1 t$'),type='l',
15        cex.axis=1.1,cex.lab=.9,lwd=3,col='blue')
16  par(opar)
17 }

```

```

1 ac<-function()
2 {
3   #Program used to create AutoCorrelation Analysis for sample, SP500 & NASDAQ
4   #Filename:AutoCorrelation.R
5   # Praba Siva
6   # praba@umich.edu
7   # @prabasiva
8
9   library(mFilter);
10  library(latex2exp)
11  setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
12  fspcom=read.table('fspcom.dat')
13  dat = (fspcom[,5])
14  mort=log(dat)
15  year=fspcom[,2]+1/12*fspcom[,3]
16  le=length(dat)
17  x=mort[2:le]
18  y=mort[1:le-1]
19  diffxy=x-y
20  #plot(diffxy,type='l')
21  dur=1:length(year)
22  lmr=lm(mort~dur)
23  intercept=coef(lmr)[1]
24  slope=coef(lmr)[2]

```



```

25 dftrend=intercept+slope*dur
26 dfcycle=mort-dftrend
27 dfacf=acf(dfcycle, plot=FALSE, 100);
28 hpf=hpfilter(mort, freq=14400)
29 layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
30 color={'blue'}
31 ac1=acf(hpf$cycle, ci.type = "ma", plot=FALSE, 100)
32 plot(year, mort, main='Log SP500 index ',
33       xlab='Year', ylab=TeX('log (SP500(t))'),
34       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='red');
35 bc1=acf(diffxy, ci.type="ma", plot=FALSE, 100)
36 plot(ac1, main='Autocorrelation of log SP500 HP Cycles '
37       , xlab='Lag', ylab='AC(1)')
38 lines(ac1$lag, ac1$acf, main='Autocorrelation of log SP500 HP Cycles ',
39       xlab='Lag', ylab='AC(1)', type='l',
40       col='blue', cex.axis=1.1, cex.lab=1.1, lwd=3)
41 plot(bc1, main='Autocorrelation of log SP500 FD ',
42       xlab='Lag', ylab='AC(1)')
43 lines(bc1$lag, bc1$acf, main='Autocorrelation of log SP500 FD ',
44       xlab='Lag', ylab='AC(1)', type='l', col='blue', lwd=3)
45 plot(dfacf, main='Autocorrelation of log-linear SP500 ',
46       xlab='Lag', ylab='AC(1)')
47 lines(dfacf$lag, dfacf$acf, main='Autocorrelation of log-linear SP500 ',
48       xlab='Lag', ylab='AC(1)', type='l',
49       col='blue', lwd=3)
50 layout(matrix(c(1,2), 2,1, byrow = TRUE))
51 plot(year, mort, main='Log SP500 index ',
52       xlab='Year', ylab=TeX('log (SP500(t))'),
53       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='red');
54 lines(year, dftrend, main='Trend of Log SP500 index using Log-linear ',
55       xlab='Year', ylab=TeX('log-linear(SP500(t))'),
56       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='blue');
57 legend("bottomright", c("Trend"), lty=c(1), lwd=c(2.5), col=c("blue"))
58 plot(year, dfcycle, main='Cycle of Log SP500 index using Log-linear ',
59       xlab='Year', ylab=TeX('log-linear(SP500(t))'),
60       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='green');
61 layout(matrix(c(1,2), 2,1, byrow = TRUE))
62 plot(year, mort, main='Log SP500 index ',
63       xlab='Year', ylab=TeX('log (SP500(t))'),
64       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='red');
65 plot(year[1:length(diffxy)], diffxy,
66       main='Cycle of Log SP500 index using Log-linear trend ',
67       xlab='Year', ylab=TeX('log-linear(SP500(t))'), type='l',
68       cex.axis=1.1, cex.lab=1.1, lwd=3, col='green');
69 sta.sp500=list(mean(dfacf$acf), sd(dfacf$acf), var(dfacf$acf), corlength(dfacf),
70               mean(ac1$acf), sd(ac1$acf), var(ac1$acf), corlength(ac1),
71               mean(bc1$acf), sd(bc1$acf), var(bc1$acf), corlength(bc1));
72 layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
73 setwd("/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program")
74 dat <- read.csv(file="nasdaq_ready.csv", head=TRUE, sep=",")
75 year=dat[,1]+1/12*dat[,2]
76 dat=dat[,3]
77 mort=log(dat)
78 le=length(dat)
79 x=mort[2:le]
80 y=mort[1:le-1]

```

```

81 diffxy=x-y
82 dur=1:length(year)
83 lmr=lm(mort~dur)
84 intercept=coef(lmr)[1]
85 slope=coef(lmr)[2]
86 dftrend=intercept+slope*dur
87 dfcycle=mort-dftrend
88 dfacf=acf(dfcycle, plot=FALSE, 100);
89 hpf=hpfilter(mort, freq=14400)
90 ac1=acf(hpf$cycle, ci.type = "ma", plot=FALSE, 100)
91 bc1=acf(diffxy, ci.type="ma", plot=FALSE, 100)
92 layout(matrix(c(1,2), 2,1, byrow = TRUE))
93 plot(year, mort, main='Log NASDAQ index ',
94       xlab='Year', ylab=TeX('log (NASDAQ(t)) '),
95       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='red');
96 lines(year, dftrend, main='Trend of Log NASDAQ index using Log-linear ',
97       xlab='Year', ylab=TeX('log-linear (NASDAQ(t)) '),
98       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='blue');
99 legend("bottomright", c("Trend"), lty=c(1), lwd=c(2.5), col=c("blue"))
100 plot(year, dfcycle, main='Cycle of Log NASDAQ index using Log-linear ',
101       xlab='Year', ylab=TeX('log-linear (NASDAQ(t)) '),
102       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='green');
103 layout(matrix(c(1,2), 2,1, byrow = TRUE))
104 plot(year, mort, main='Log NASDAQ index ',
105       xlab='Year', ylab=TeX('log (NASDAQ(t)) '),
106       type='l', cex.axis=1.1, cex.lab=1.1, lwd=3, col='red');
107 plot(year[1:length(diffxy)], diffxy,
108       main='Cycle of Log NASDAQ index using Log-linear trend ',
109       xlab='Year', ylab=TeX('log-linear (NASDAQ(t)) '), type='l',
110       cex.axis=1.1, cex.lab=1.1, lwd=3, col='green');
111 layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
112 plot(year, mort, main='Log NASDAQ index ',
113       xlab='Year', ylab=TeX('log (NASDAQ(t)) '), type='l',
114       col='red', cex.axis=1.1, cex.lab=1.1, lwd=3);
115 plot(ac1, main='Autocorrelation of log NASDAQ HP Cycles ',
116       xlab='Lag', ylab='AC(1) ')
117 lines(ac1$lag, ac1$acf, main='Autocorrelation of log NASDAQ HP Cycles ',
118       xlab='Lag', ylab='AC(1) ', type='l',
119       col='blue', cex.axis=1.1, cex.lab=1.1, lwd=3)
120 plot(bc1, main='Autocorrelation of log NASDAQ FD ',
121       xlab='Lag', ylab='AC(1) ')
122 lines(bc1$lag, bc1$acf, main='Autocorrelation of log NASDAQ FD ',
123       xlab='Lag', ylab='AC(1) ', type='l',
124       col='blue', cex.axis=1.1, cex.lab=1.1, lwd=3)
125 plot(dfacf, main='Autocorrelation of log-linear NASDAQ ',
126       xlab='Lag', ylab='AC(1) ')
127 lines(dfacf$lag, dfacf$acf, main='Autocorrelation of log-linear NASDAQ ',
128       xlab='Lag', ylab='AC(1) ', type='l', col='blue', lwd=3)
129 layout(matrix(c(1,2,3,4,5,6), 3, 2, byrow = TRUE))
130 #par(mfrow=c(2,1), mar=c(3,3,2,1), cex=.8)
131 x=seq(-15,15,.1);
132 y=sin(x)
133 ac1=acf(y, lag.max=100, plot=FALSE);
134 plot(x,y, main='Sin wave', xlab='T', ylab='Sin(t)', type='l',
135       col='red', cex.axis=1.1, cex.lab=1.1, lwd=3)
136 plot(ac1, main='Autocorrelation of Sin wave', xlab='Lag', ylab='AC(1) ',

```

```

137     cex.axis=1.1 , cex.lab=1.1 , lwd=.2)
138 lines(ac1$lag , ac1$acf , type='l' , col='blue' , lwd=2)
139 x=seq(-15,15,.1);
140 y=x^2+x^3
141 ac1=acf(y, lag.max=100, plot=FALSE);
142 plot(x,y, main='Polynomial' ,
143       xlab='T' , ylab=TeX('y=x^3(t)+x^2(t)') , type='l' , col='red' ,
144       cex.axis=1.1 , cex.lab=1.1 , lwd=3)
145 plot(ac1, main='Autocorrelation of Polynomial' ,
146       xlab='Lag' , ylab='AC(1)' , cex.axis=1.1 , cex.lab=1.5 , lwd=.2)
147 lines(ac1$lag , ac1$acf , type='l' , col='blue' , lwd=2)
148 x=seq(-15,15,.1);
149 y=sin(x)*rnorm(length(x) , mean=0, sd=1)
150 ac1=acf(y, lag.max=100, plot=FALSE);
151 plot(x,y, main='Sin wave with random noise' ,
152       xlab='t' ,
153       ylab=TeX('Sin(t) * r(\mu=0 , \sigma^2=1)') , type='l' , col='red' ,
154       cex.axis=1.1 , cex.lab=1.1 , lwd=2)
155 plot(ac1, main='Autocorrelation of Sin wave with random noise' ,
156       xlab='Lag' , ylab='AC(1)' , cex.axis=1.1 , cex.lab=1.5 , lwd=3)
157 lines(ac1$lag , ac1$acf , type='l' , col='blue' , lwd=.2)
158 corlength(ac1)
159 sta.nasdaq=list(mean(dfacf$acf) , sd(dfacf$acf) , var(dfacf$acf) , corlength(dfacf) ,
160                 mean(ac1$acf) , sd(ac1$acf) , var(ac1$acf) , corlength(ac1) ,
161                 mean(bc1$acf) , sd(bc1$acf) , var(bc1$acf) , corlength(bc1))
162 print("Dtrend statistics for SP500")
163 print(matrix(sta.sp500 , nrow=4))
164 print("Dtrend statistics for NASDAQ")
165 print(matrix(sta.nasdaq , nrow=4))
166 }
167
168 corlength <- function(acfvector)
169 {
170 {
171
172     ind=min(which(acfvector$acf < 0));
173
174     return . . .
175         ((abs(acfvector$acf[ind])+abs(acfvector$acf[ind-1])/10)*(abs(acfvector$acf[ind-1]))+ind-1)
176 }

```

## APPENDIX B

### Matlab code that are used to do analysis

```
1 function DrawSinFourierGraph()
2 Fs = 1000; % Sampling frequency
3 T = 1/Fs; % Sampling period
4 L = 200; % Length of signal
5 t = (0:L-1)*T; % Time vect
6 f = [50,150,300];
7 x1 = sin(2*pi*f(1)*t); % First row wave
8 x2 = sin(2*pi*f(2)*t); % Second row wave
9 x3 = sin(2*pi*f(3)*t); % Third row wave
10
11 X = [x1; x2; x3;];
12
13 figure;
14
15 for i = 1:3
16     g=subplot(3,2,i+i-1)
17     plot(t(1:L),X(i,1:L),'r','LineWidth',1)
18     ylabel('sin(2\pift)','FontSize',16,'FontWeight','bold')
19     xlabel('\fontname{Helvetica} Time','FontSize',16,'FontWeight','bold')
20     title(['A sin wave of frequency f= ',num2str(f(i)),' in the Time ...
           Domain'],'FontSize',18,'FontWeight','bold')
21     p=get(g,'position');
22     p(1)=.7*p(1);
23     p(4)=1.1*p(4);
24     set(g,'position',p);
25 end
26
27
28 n = 2^nextpow2(L);
29 dim = 2;
30 Y = fft(X,n,dim);
31 P2 = abs(Y/n);
```

```

32 P1 = P2(:,1:n/2+1);
33 P1(:,2:end-1) = 2*P1(:,2:end-1);
34
35
36 for i=1:3
37     g=subplot(3,2,i*2);
38     plot(0:(Fs/n):(Fs-Fs/n),P2(i,1:n),'b','LineWidth',1)
39     title(['Sin wave of frequency ',num2str(f(i)), ' in the Frequency ...
           Domain'], 'FontSize',18,'FontWeight','bold')
40     ylabel(' |F(s)| ', 'FontSize',16,'FontWeight','bold')
41     xlabel(' Frequency ', 'FontSize',16,'FontWeight','bold')
42     p=get(g,'position');
43     p(1)=.9*p(1);
44     p(4)=1.1*p(4);
45     set(g,'position',p);
46 end

```

```

1 function SPNASDAQFourier()
2 Fs = 1000;           % Sampling frequency
3 T = 1/Fs;           % Sampling period
4 L = 1000;           % Length of signal
5 t = (0:L-1)*T;      % Time vector
6 S = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
7 X = S + 2*randn(size(t));
8 g=subplot(5,2,1);
9 plot(1000*t(1:50),X(1:50),'r','LineWidth',1);
10 title('Signal Corrupted with Zero-Mean Random Noise','FontSize',18,'FontWeight','bold')
11 xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
12 ylabel('f(t)','FontSize',16,'FontWeight','bold')
13 Y = fft(X);
14 P2 = abs(Y/L);
15 P1 = P2(1:L/2+1);
16 P1(2:end-1) = 2*P1(2:end-1);
17 f = Fs*(0:(L/2))/L;
18 g=subplot(5,2,2);
19 plot(f,P1,'b','LineWidth',1)
20 title('Single-Sided Amplitude Spectrum of f(t)','FontSize',16,'FontWeight','bold')
21 xlabel('w','FontSize',16,'FontWeight','bold')
22 ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
23 g=subplot(5,2,3);
24 plot(1000*t(1:50),S(1:50),'r','LineWidth',1);
25 title('Signal with Zero-Mean Random Noise','FontSize',18,'FontWeight','bold')
26 xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
27 ylabel('f(t)','FontSize',16,'FontWeight','bold')
28 Y = fft(S);
29 P2 = abs(Y/L);
30 P1 = P2(1:L/2+1);
31 P1(2:end-1) = 2*P1(2:end-1);
32 f = Fs*(0:(L/2))/L;
33 g=subplot(5,2,4);
34 plot(f,P1,'b','LineWidth',1)
35 title('Single-Sided Amplitude Spectrum of f(t)','FontSize',18,'FontWeight','bold')
36 xlabel('w','FontSize',16,'FontWeight','bold')

```

```

37 ylabel(' |F(w)| ', 'FontSize', 16, 'FontWeight', 'bold')
38 Fs = 100; % Sampling frequency
39 t = -0.5:1/Fs:0.5; % Time vector
40 L = length(t); % Signal length
41 X = 1/(4*sqrt(2*pi*0.01))*(exp(-t.^2/(2*0.01)));
42 g=subplot(5,2,5)
43 plot(t,X,'r','LineWidth',1)
44 title('Gaussian Pulse in Time Domain', 'FontSize', 18, 'FontWeight', 'bold')
45 xlabel('Time (t)', 'FontSize', 16, 'FontWeight', 'bold')
46 ylabel('f(t)', 'FontSize', 16, 'FontWeight', 'bold')
47 n = 2^nextpow2(L);
48 Y = fft(X,n);
49 f = Fs*((n/2)/n):(n/2)/n;
50 P = abs(Y/n);
51 g=subplot(5,2,6)
52 plot(f,P(1:n/2+1), 'b', 'LineWidth', 1)
53 title('Gaussian Pulse in Frequency Domain', 'FontSize', 18, 'FontWeight', 'bold')
54 xlabel('Frequency (w)', 'FontSize', 16, 'FontWeight', 'bold')
55 ylabel(' |F(s)| ', 'FontSize', 16, 'FontWeight', 'bold')
56 dat=csvread('/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program/nasdaq-mat.csv');
57 year=dat(:,1);
58 month=dat(:,2);
59 naq=dat(:,3);
60 naq=log(naq);
61 year=year+month/12;
62 g=subplot(5,2,7)
63 plot(year,naq,'r','LineWidth',1);
64 title('Log Nasdaq', 'FontSize', 18, 'FontWeight', 'bold');
65 xlabel('Year.month', 'FontSize', 16, 'FontWeight', 'bold');
66 ylabel('Log Nasdaq', 'FontSize', 16, 'FontWeight', 'bold');
67 Fs = 1000;
68 [L, tp]=size(naq)
69 n = 2^nextpow2(L);
70 Y = fft(naq,n);
71 f = Fs*((n/2)/n):(n/2)/n;
72 P = abs(Y/n);
73 g=subplot(5,2,8)
74 %plot(f,P(1:n/2+1), 'b', 'LineWidth', 1)
75 plot(1:50,P(1:50), 'b', 'LineWidth', 1)
76 title('Nasdaq in Frequency Domain', 'FontSize', 18, 'FontWeight', 'bold')
77 xlabel('Frequency (w)', 'FontSize', 16, 'FontWeight', 'bold')
78 ylabel(' |F(w)| ', 'FontSize', 16, 'FontWeight', 'bold')
79 dat=readtable('/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program/fspcom.dat');
80 [maxx,maxy]=size(dat);
81 sp500=table2array(dat(1:maxx,5));
82 year=(table2array(dat(1:maxx,2)));
83 month=(table2array(dat(1:maxx,3)));
84 year=year+month/12
85 sp500=log(sp500)
86 g=subplot(5,2,9)
87 plot(year,sp500,'r','LineWidth',1);
88 title('Log S&P 500', 'FontSize', 18, 'FontWeight', 'bold');
89 xlabel('Year.month', 'FontSize', 16, 'FontWeight', 'bold');
90 ylabel('Log S&P 500', 'FontSize', 16, 'FontWeight', 'bold');
91 [L, tp]=size(sp500)
92 n = 2^nextpow2(L);

```

```

93 Y = fft(sp500,n);
94 f = Fs*((n/2)/n):(n/2)/n;
95 P = abs(Y/n);
96 g=subplot(5,2,10)
97 %plot(f,P(1:n/2+1),'b','LineWidth',1);
98 plot(1:50,P(1:50),'b','LineWidth',1);
99 title('S&P in Frequency Domain','FontSize',16,'FontWeight','bold')
100 xlabel('Frequency (w)','FontSize',16,'FontWeight','bold')
101 ylabel('|F(w)|','FontSize',16,'FontWeight','bold')

```

```

1 function ghamwin()
2 s={'Gaussian Window' 'Hamming Window'};
3 wlen=25;
4 figure;
5 for fla=0:1
6     if fla<1
7         % form a periodic hamming window
8         win = hamming(wlen, 'periodic');
9     else
10        win=gausswin(wlen)
11    end
12    g=subplot(1,2,1+fla);
13    plot(abs(win),'r','LineWidth',1);
14    xlabel('n (N=25)','FontSize',16,'FontWeight','bold')
15    if fla ==0
16        title('\sigma = 2.5 Gaussian Window ...
17              function','interpreter','Tex','FontSize',16,'FontWeight','bold')
18        ylabel('e^{-n^2/2\sigma^2}','interpreter','Tex','FontSize',20)
19    else
20        title([s(fla+1),'function'],'FontSize',16,'FontWeight','bold')
21        ylabel('0.54-0.46cos(2\pi*n/N)','interpreter','Tex','FontSize',20)
22    end
23 end

```

```

1 function drawSTFTEg()
2 s={'Gaussian Window' 'Hamming Window'};
3 wlen=25;
4 hopsize=25;
5 retno =1;
6 Ff = 500;
7 for fla=0:1
8     Fs = 1000;           % Sampling frequency
9     T = 1/Fs;           % Sampling period
10    L = 1000;            % Length of signal
11    t = (0:L-1)*T;       % Time vector
12    S = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
13    X = S + 2*randn(size(t));
14    figure;
15    g=subplot(2,2,1);
16    plot(1000*t(1:100),X(1:100),'r','LineWidth',1);

```

```

17     L = length(X);
18     title('Signal Corrupted with Zero-Mean Random Noise','FontSize',18,'FontWeight','bold')
19     xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
20     ylabel('f(t)','FontSize',16,'FontWeight','bold')
21     [Y,ft,tt]=stft2(X,wlen,hopsize,retno,Ff,fla);
22     %Y=fftshift(Y);
23     P=abs(Y/L);
24     g=subplot(2,2,2);
25     plot(tt,P,'b','LineWidth',1);
26     title(['STFT using ',s(fla+1)], 'FontSize',16,'FontWeight','bold')
27     xlabel('\omega','FontSize',16,'FontWeight','bold')
28     ylabel('|F(\omega,\tau)|','interpreter','Tex','FontSize',16,'FontWeight','bold')
29     g=subplot(2,2,3);
30     plot(1000*t(1:100),S(1:100),'r','LineWidth',1);
31     L = length(X);
32     title('Signal Corrupted with Zero-Mean Random Noise','FontSize',18,'FontWeight','bold')
33     xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
34     ylabel('f(t)','FontSize',16,'FontWeight','bold')
35     [Y,ft,tt]=stft2(S,wlen,hopsize,retno,Ff,fla);
36     %Y=fftshift(Y);
37     P=abs(Y/L);
38     g=subplot(2,2,4);
39     plot(tt,P,'b','LineWidth',1);
40     title(['STFT using ',s(fla+1)], 'FontSize',16,'FontWeight','bold')
41     xlabel('\omega','interpreter','Tex','FontSize',16,'FontWeight','bold')
42     ylabel('|F(\omega,\tau)|','interpreter','Tex','FontSize',16,'FontWeight','bold')
43
44 end

```

```

1 function [stft, f, t] = stft2(x, wlen, h, nfft, fs, flag)
2 % function: [stft, f, t] = stft(x, wlen, h, nfft, fs)
3 % x - signal in the time domain
4 % wlen - length of the hamming window
5 % h - hop size
6 % nfft - number of FFT points
7 % flag = 1 for Gaussian window or 0 for Hamming window
8 % fs - sampling frequency, Hz
9 % f - frequency vector, Hz
10 % t - time vector, s
11 % stft - STFT matrix (only unique points, time across columns, freq across rows)
12 % represent x as column-vector if it is not
13 if size(x, 2) > 1
14     x = x';
15 end
16 % length of the signal
17 xlen = length(x);
18 if flag<1
19     % form a periodic hamming window
20     win = hamming(wlen, 'periodic');
21 else
22     win=gausswin(wlen)
23 end
24 % form the stft matrix

```



```

25 rown = ceil((1+nfft)/2);           % calculate the total number of rows
26 coln = 1+fix((xlen-wlen)/h);       % calculate the total number of columns
27 stft = zeros(rown, coln);          % form the stft matrix
28
29 % initialize the indexes
30 indx = 0;
31 col = 1;
32
33 % perform STFT
34 while indx + wlen ≤ xlen
35     % windowing
36     xw = x(indx+1:indx+wlen).*win;
37
38     % FFT
39     X = fft(xw, nfft);
40
41     % update the stft matrix
42     stft(:, col) = X(1:rown);
43
44     % update the indexes
45     indx = indx + h;
46     col = col + 1;
47 end
48
49 % calculate the time and frequency vectors
50 t = (wlen/2:h:wlen/2+(coln-1)*h)/fs;
51 f = (0:rown-1)*fs/nfft;
52
53 end

```

```

1 function [index,year]= getData(flag)
2 if flag == 1
3     dat=readtable('/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program/fspcom.dat');
4     [maxx,mxy]=size(dat);
5     index=table2array(dat(1:maxx,5));
6     year=(table2array(dat(1:maxx,2)));
7     month=(table2array(dat(1:maxx,3)));
8     index=log(index);
9     year=year+month/12;
10 else
11     dat=csvread('/Users/sivaspl/Documents/2016/Personal/Praba/MATH599/program/nasdaq_mat.csv');
12     year=dat(:,1);
13     month=dat(:,2);
14     index=dat(:,3);
15     index=log(index);
16     year=year+month/12;
17 end

```

```

1 function mywvd()
2

```

```

3  %Program used to Wigner Distribution for NASDAQ & SP500 indexes
4  % Praba Siva
5  % praba@umich.edu
6  % @prabasiva
7  % Filename: mywvd.m
8
9  close all;
10 clear all;
11 [sp500,syear]=getData(1);
12 sp500=log(sp500);
13 [naq,nyear]=getData(2);
14 naq=log(naq);
15
16 for step = 1:2
17
18     if step == 2
19         sp500=naq;
20         syear=nyear;
21     end;
22
23     figure;
24
25     [s2,s1]=hpfiler(sp500,1600);
26     subplot(2,3,1);
27     plot(syear,sp500);
28     xlabel('Time in years');
29     ylabel('log s(t)');
30     title('Log s(t)');
31
32     [wd,freq]=wig2(sp500);
33     subplot(2,3,2);
34     contour(syear,freq,abs(wd'),8), grid on
35     xlabel('Time in years');
36     ylabel('Frequency');
37     title('Contour Map');
38
39     subplot(2,3,3);
40     mesh(syear,freq,abs(wd')));
41     xlabel('Time in years');
42     ylabel('Frequency');
43     zlabel('Amplitude');
44
45     subplot(2,3,4);
46     plot(syear,s1);
47     xlabel('Time in years');
48     ylabel('Cycles');
49     title('HP Filter cycles');
50
51     [wd,freq]=wig2(s1);
52     subplot(2,3,5);
53     contour(syear,freq,abs(wd'),8), grid on
54     xlabel('Time in years');
55     ylabel('Frequency');
56     title('Contour Map');
57
58     subplot(2,3,6);

```

```

59     mesh(syear , freq , abs(wd')) ;
60     xlabel( 'Time in years' );
61     ylabel( 'Frequency' );
62     zlabel( 'Amplitude' );
63
64 end;

```

```

1  function mywvdgauss()
2
3  %Program used to Wigner Ville Distribution for Gaussian function
4  % Praba Siva
5  % praba@umich.edu
6  % @prabasiva
7  % Filename: mywvdgauss.m
8
9  close all;
10 clear all;
11 t=-128:127;
12 sigma=.1;
13 coef=nthroot(pi*sigma*sigma,-4);
14 expo=(t.*t)/2*sigma*sigma;
15 g=coef*exp(-expo);
16 subplot(2,1,1);
17 plot(t,g);
18 [wd,freq]=wig2(g);
19 subplot(2,1,2)
20 contour(t,freq,wd',8)

```

```

1  function mygabor()
2  %Program used to Gabor Coefficients for SP500 & NASDAQ
3  % Praba Siva
4  % praba@umich.edu
5  % @prabasiva
6  % Filename: mygabor.m
7  close all;
8  clear all;
9  [sp500,syear]=getData(1);
10 [naq,nyear]=getData(2);
11     Δn = 8;
12     %M=16;
13     M=50
14     Δn=4;
15     %nn=32;
16     nn=100;
17
18
19     [s2,s1]=hpfiler(sp500,1600);
20
21
22     s1=s1';

```

```

23     L=length(s1);
24     t=1:L;
25     N=L/2;
26     nn2=nn/2;
27     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
28     c=nthroot(pi*sigma*sigma,-4);
29     h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
30     h = @(ii) h0(mod(ii + N, L)-N);
31     for m = 1:M
32         for n = 1:nn2
33             c1(m, n)= sum(s1.*h(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
34         end
35     end
36
37
38     [s2,s1]=hpfiltfilt(naq,1600);
39     s1=s1';
40     L=length(s1);
41     t=1:L;
42     N=L/2;
43     nn2=nn/2;
44     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
45     c=nthroot(pi*sigma*sigma,-4);
46     h1 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
47     h2 = @(ii) h1(mod(ii + N, L)-N);
48     for m = 1:M
49         for n = 1:nn2
50             c2(m, n)= sum(s1.*h2(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
51         end
52     end
53     subplot(1,2,1)
54     surf(abs(c1));
55     %Change the text location based on the m & n.
56     %For m=n=16, text(-5,0.4...
57     %For m=n=50 text
58     if M==16
59         text(-5, 0.4, 'Fig a: Gabor Coefficient for ...
60             log(sp500)', 'FontSize',16,'FontWeight','bold','Color','r')
61     else
62         text(-15, 0.4, 'Fig a: Gabor Coefficient for ...
63             log(sp500)', 'FontSize',16,'FontWeight','bold','Color','r')
64     end;
65     xlabel('Time','FontSize',12,'FontWeight','bold','Color','b')
66     h=get(gca,'xlabel');
67     set(h,'rotation',30)
68     ylabel('Frequency','FontSize',12,'FontWeight','bold','Color','b')
69     h=get(gca,'ylabel');
70     set(h,'Position',get(h,'Position')+[2 4 0])
71     set(h,'rotation',140)
72     zlabel('|C(m,n)|^2','FontSize',12,'FontWeight','bold','Color','b')
73
74     subplot(1,2,2);
75     surf(abs(c2));
76     colormap hsv;
77     if M==16
78         text(-5, 0.4, 'Fig a: Gabor Coefficient for ...

```

```

        log(nasdaq)', 'FontSize', 16, 'FontWeight', 'bold', 'Color', 'r')
77     else
78         text(-15, 0.4, 'Fig a: Gabor Coefficient for ...
        log(nasdaq)', 'FontSize', 16, 'FontWeight', 'bold', 'Color', 'r')
79     end;
80     xlabel('Time', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
81     h=get(gca, 'xlabel');
82     set(h, 'rotation', 30)
83     ylabel('Frequency', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
84     h=get(gca, 'ylabel');
85     set(h, 'Position', get(h, 'Position') + [2 4 0])
86     set(h, 'rotation', 140)
87     zlabel('|C(m,n)|^2', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
88 end

```

```

1  function myfiltgabor()
2  %Program used to Gabor Coefficients for SP500 & NASDAQ
3  % Praba Siva
4  % praba@umich.edu
5  % @prabasiva
6  % Filename: myfiltgabor.m
7  close all;
8  clear all;
9  [sp500, syeaer]=getData(1);
10 [naq, nyear]=getData(2);
11     Δm = 8;
12     M=16;
13     %M=50;
14     Δn=4;
15     nn=32;
16     %nn=100;
17     thrcont=3;
18     [s2, s1]=hpfiler(sp500, 1600);
19     s1=s1';
20     L=length(s1);
21     t=1:L;
22     N=L/2;
23     nn2=nn/2;
24     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
25     c=nthroot(pi*sigma*sigma, -4);
26     h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
27     h = @(ii) h0(mod(ii + N, L)-N);
28     for m = 1:M
29         for n = 1:nn2
30             c1(m, n)= sum(s1.*(mod(t - m*Δm, L)).*exp(-2*pi*i*Δn*n*t/L));
31         end
32     end
33
34     [s2, s1]=hpfiler(naq, 1600);
35     s1=s1';
36     L=length(s1);
37     t=1:L;
38     N=L/2;

```

```

39     nn2=nn/2;
40     sigma=sqrt((Δn*L)/(Δn * 2 * pi));
41     c=nthroot(pi*sigma*sigma,-4);
42     h1 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
43     h2 = @(ii) h1(mod(ii + N, L)-N);
44     for m = 1:M
45         for n = 1:nn2
46             c2(m, n)=sum(s1.*h2(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
47         end
48     end
49
50     [m,n]=size(c1);
51     thr=max(abs(c1));
52     spmask = (max(thr)-min(thr))/thrcont;
53     for k = 1:m
54         for j = 1:n
55             if abs(c1(k,j)) < spmask
56                 sp500maskmatrix(k,j)=0;
57             else
58                 sp500maskmatrix(k,j)=1;
59             end;
60         end;
61     end;
62
63     %DRAW THRESHOLD PRESENTATION & MASK OPERATOR
64     figure;
65     subplot(2,1,1);
66     plot(abs(c1),'LineWidth',2);
67     xlabel('m');
68     ylabel(' |C(m,n)| ');
69     title('Time Section of Gabor Distribution for SP500');
70     hold on;
71     li(1:m)=spmask;
72     p1=plot(li,'LineWidth',6,'Color','b');
73     legend(p1,'Threshold');
74
75     [m,n]=size(c2);
76     thr(1:m)=max(abs(c2(1:m,:)));
77
78     nasmask = (max(thr)-min(thr))/thrcont;
79     for k = 1:m
80         for j = 1:n
81             if abs(c2(k,j)) < nasmask
82                 nasmaskmatrix(k,j)=0;
83             else
84                 nasmaskmatrix(k,j)=1;
85             end;
86         end;
87     end;
88     subplot(2,1,2);
89     plot(abs(c2),'LineWidth',2);
90     xlabel('m');
91     ylabel(' |C(m,n)| ');
92     title('Time Section of Gabor Distribution for NASDAQ');
93     hold on;
94     li(1:m)=nasmask;

```

```

95  pl=plot(li, 'LineWidth',6, 'Color', 'b');
96  legend(pl, 'Threshold');
97  c1=c1.*sp500maskmatrix;
98  c2=c2.*nasmaskmatrix;
99  figure;
100 subplot(2,1,1);
101 plot(abs(c1), 'LineWidth',2);
102 xlabel('m');
103 ylabel('|C(m,n)|');
104 title('Time Section of Masked Gabor Distribution for SP500 ');
105 hold on;
106 li(1:m)=spmask;
107 pl=plot(li, 'LineWidth',6, 'Color', 'b');
108 legend(pl, 'Threshold');
109
110 subplot(2,1,2);
111 plot(abs(c2), 'LineWidth',2);
112 xlabel('m');
113 ylabel('|C(m,n)|');
114 title('Time Section of Masked Gabor Distribution for NASDAQ');
115 hold on;
116 li(1:m)=nasmask;
117 pl=plot(li, 'LineWidth',6, 'Color', 'b');
118 legend(pl, 'Threshold');
119
120 figure;
121
122     subplot(1,2,1)
123     surf(abs(c1));
124     colormap hsv;
125     %Change the text location based on the m & n.
126     %For m=n=16, text(-5,0.4...
127     %For m=n=50 text
128     if M==16
129         text(-5, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
130             log(sp500)', 'FontSize',16, 'FontWeight', 'bold', 'Color', 'r')
131     else
132         text(-15, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
133             log(sp500)', 'FontSize',16, 'FontWeight', 'bold', 'Color', 'r')
134     end;
135     xlabel('Time', 'FontSize',12, 'FontWeight', 'bold', 'Color', 'b')
136     h=get(gca, 'xlabel');
137     set(h, 'rotation',30)
138     ylabel('Frequency', 'FontSize',12, 'FontWeight', 'bold', 'Color', 'b')
139     h=get(gca, 'ylabel');
140     set(h, 'Position', get(h, 'Position') +[2 4 0])
141     set(h, 'rotation',140)
142     zlabel('|C(m,n)|^2', 'FontSize',12, 'FontWeight', 'bold', 'Color', 'b')
143
144     subplot(1,2,2);
145     surf(abs(c2));
146     colormap hsv;
147     if M==16
148         text(-5, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
149             log(nasdaq)', 'FontSize',16, 'FontWeight', 'bold', 'Color', 'r')
150     else

```

```

148         text(-15, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
            log(nasdaq)', 'FontSize', 16, 'FontWeight', 'bold', 'Color', 'r')
149     end;
150     xlabel('Time', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
151     h=get(gca, 'xlabel');
152     set(h, 'rotation', 30)
153     ylabel('Frequency', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
154     h=get(gca, 'ylabel');
155     set(h, 'Position', get(h, 'Position') + [2 4 0])
156     set(h, 'rotation', 140)
157     zlabel('|C(m,n)|^2', 'FontSize', 12, 'FontWeight', 'bold', 'Color', 'b')
158
159 end

```

```

1  function myreconsfromgabor()
2  %Program used to Gabor Coefficients for SP500 & NASDAQ
3  % Praba Siva
4  % praba@umich.edu
5  % @prabasiva
6  % Filename: myreconsfromgabor.m
7
8      close all;
9      clear all;
10     [sp500, syeaer]=getData(1);
11     % sp500=log(sp500);
12     [s2, s1]=hpfiltet(sp500, 1600);
13
14
15     Δm = 8;
16     M=17;
17     % M=50;
18     Δn=4;
19     nn=84;
20     %nn=100;
21     thrcont=3;
22
23     s1=s1';
24     L=length(s1);
25     t=1:L;
26     N=L/2;
27     nn2=nn/2;
28     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
29     c=nthroot(pi*sigma*sigma, -4);
30     h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
31     h = @(ii) h0(mod(ii + N, L)-N);
32     for m = 1:M
33         for n = 1:nn2
34             c1(m, n)= sum(s1.*h(mod(t - m*Δm, L)).*exp(-2*pi*i*Δn*n*t/L));
35         end
36
37     end
38
39

```



```

40     [m,n]=size(c1);
41
42     H=0.5;
43     %thr=max(abs(c1));
44     %spmask = (max(thr)-min(thr))/thrcont;
45     spmask=mean(c1)+H*std(c1);
46     for k = 1:m
47         for j = 1:n
48             if abs(c1(k,j)) < abs(spmask(k))
49                 sp500maskmatrix(k,j)=0;
50             else
51                 sp500maskmatrix(k,j)=1;
52             end;
53         end;
54     end;
55
56     for t = 1:L
57         temp=0;
58         for m = 1:M
59             for n = 1:nn2
60                 temp= temp+c1(m,n).*h(mod(t - m*Δm,L)).*exp(2*pi*i*Δn*n*t/L);
61             end
62         end
63         sg(t)=temp;
64
65     end;
66     sg=sg/(2*pi);
67     c1=c1.*sp500maskmatrix;
68
69     for t = 1:L
70         temp=0;
71         for m = 1:M
72             for n = 1:nn2
73                 temp= temp+c1(m,n).*h(mod(t - m*Δm,M)).*exp(2*pi*i*Δn*n*t/L);
74             end
75         end
76         sg2(t)=temp;
77
78     end;
79     sg2=sg2/(2*pi);
80     % plot(real(sg),'LineWidth',3);
81     subplot(2,1,2);
82     hold on;
83     autocorr(s1,100);
84     [c1,c2,c3]=autocorr(s1,100);
85     hold on;
86     p1=plot(c2,c1,'LineWidth',2,'Color','c');
87     hold on;
88     autocorr(real(sg2),100);
89     [d1,d2,d3]=autocorr(real(sg2),100);
90     hold on;
91     p2=plot(d2,d1,'LineWidth',2,'Color','g');
92     legend([p1,p2], 'HP Original', 'HP Filtered');
93     title('Original & Filtered HP Cycles');
94     stdratio=std(real(sg2),1)/std(s1,1)
95     vratio=var(real(sg2),1)/var(s1,1) * 100

```

```

96         ccgo=corrcoef(real(sg2),s1)
97
98         subplot(2,1,1);
99         %figure;
100
101         p1=plot(real(sg2), 'LineWidth',.5, 'DisplayName', 'Constructued signal');
102         hold on;
103         p2=plot(s1, 'DisplayName', 'Original Signal');
104         legend('show');
105         ylabel('S(t)');
106         xlabel('Time in year');
107         title('log(SP500) original & Filtered HP Cycles');
108
109         figure;
110         clear all;
111
112         [naq,nyear]=getData(2);
113         naq=log(naq);
114         [s2,s1]=hpfilter(naq,1600);
115         Δn = 8;
116         % M=16;
117         M=50;
118         Δn=4;
119         %nn=32;
120         nn=100;
121         thrcont=3;
122
123         s1=s1';
124         L=length(s1);
125         t=1:L;
126         N=L/2;
127         nn2=nn/2;
128         sigma=sqrt((Δn*L)/(Δn * 2 * pi));
129         c=nthroot(pi*sigma*sigma,-4);
130         h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
131         h = @(ii) h0(mod(ii + N, L)-N);
132         for m = 1:M
133             for n = 1:nn2
134                 c1(m, n)= sum(s1.*h(mod(t - m*Δn,L)).*exp(-2*pi*i*Δn*n*t/L));
135             end
136         end
137
138
139
140         [m,n]=size(c1);
141
142         H=0.5;
143         %thr=max(abs(c1));
144         %spmask = (max(thr)-min(thr))/thrcont;
145         spmask=mean(c1)+H*std(c1);
146         for k = 1:m
147             for j = 1:n
148                 if abs(c1(k,j)) < abs(spmask(k))
149                     sp500maskmatrix(k,j)=0;
150                 else
151                     sp500maskmatrix(k,j)=1;

```

```

152         end;
153     end;
154 end;
155
156     for t = 1:L
157         temp=0;
158         for m = 1:M
159             for n = 1:nn2
160                 temp= temp+c1(m,n).*h(mod(t - m*Δm,L)).*exp(2*pi*i*Δn*n*t/L);
161             end
162         end
163         sg(t)=temp;
164
165     end;
166     sg=sg/(2*pi);
167     c1=c1.*sp500maskmatrix;
168
169     for t = 1:L
170         temp=0;
171         for m = 1:M
172             for n = 1:nn2
173                 temp= temp+c1(m,n).*h(mod(t - m*Δm,M)).*exp(2*pi*i*Δn*n*t/L);
174             end
175         end
176         sg2(t)=temp;
177
178     end;
179     sg2=sg2/(2*pi);
180     % plot(real(sg),'LineWidth',3);
181     subplot(2,1,2);
182     hold on;
183     autocorr(s1,100);
184     [c1,c2,c3]=autocorr(s1,100);
185     hold on;
186     p1=plot(c2,c1,'LineWidth',2,'Color','c');
187     hold on;
188     autocorr(real(sg2),100);
189     [d1,d2,d3]=autocorr(real(sg2),100);
190     hold on;
191     p2=plot(d2,d1,'LineWidth',2,'Color','g');
192     legend([p1,p2], 'HP Original', 'HP Filtered');
193     title('Original & Filtered HP Cycles');
194     stdratio=std(real(sg2),1)/std(s1,1)
195     vratio=var(real(sg2),1)/var(s1,1) * 100
196     ccgo=corrcoef(real(sg2),s1)
197
198     subplot(2,1,1);
199     %figure;
200
201     p1=plot(real(sg2),'LineWidth',.5,'DisplayName','Constructued signal');
202     hold on;
203     p2=plot(s1,'DisplayName','Original Signal');
204     legend('show');
205     ylabel('S(t)');
206     xlabel('Time in year');
207     title('log(NASDAQ) original & Filtered HP Cycles');

```

```

1  %Program used to Gabor Coefficients for SP500 & NASDAQ
2  % Praba Siva
3  % praba@umich.edu
4  % @prabasiva
5  % Filename: myAttractor.m
6
7      close all;
8      clear all;
9      [sp500,syear]=getData(1);
10     % sp500=log(sp500);
11     [s2,s1]=hpfiter(sp500,1600);
12
13
14     Δm = 12;
15     M=17;
16     %M=50;
17     Δn=4;
18     nn=84;
19     %nn=100;
20     thrcont=1;
21     s1=s1';
22     L=length(s1);
23     t=1:L;
24     N=L/2;
25     nn2=nn/2;
26     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
27     c=nthroot(pi*sigma*sigma,-4);
28     h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
29     h = @(ii) h0(mod(ii + N, L)-N);
30     for m = 1:M
31         for n = 1:nn2
32             c1(m,n)=sum(s1.*h(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
33         end
34     end
35
36     [m,n]=size(c1);
37     H=0.5;
38     %thr=max(abs(c1));
39     %spmask = (max(thr)-min(thr))/thrcont;
40     spmask=mean(c1)+H*std(c1);
41     for k = 1:m
42         for j = 1:n
43             if abs(c1(k,j)) < abs(spmask(k))
44                 % if abs(c1(k,j)) < spmask
45
46                 sp500maskmatrix(k,j)=0;
47             else
48                 sp500maskmatrix(k,j)=1;
49             end;
50         end;
51     end;

```

```

52
53     for t = 1:L
54         temp=0;
55         for m = 1:M
56             for n = 1:nn2
57                 temp= temp+c1(m,n).*h(mod(t - m*Δm,L)).*exp(2*pi*i*Δn*n*t/L);
58             end
59         end
60         sg(t)=temp;
61
62     end;
63     sg=sg/(2*pi);
64     c1=c1.*sp500maskmatrix;
65     for t = 1:L
66         temp=0;
67         for m = 1:M
68             for n = 1:nn2
69                 temp= temp+c1(m,n).*h(mod(t - m*Δm,M)).*exp(2*pi*i*Δn*n*t/L);
70             end
71         end
72         sg2(t)=temp;
73     end;
74     Δ=60;
75     % subplot(2,1,1);
76
77     % x=real(s1(1:length(s1)-Δ));
78     % y=real(s1(Δ+1:length(s1)));
79     % plot(x,y);
80     % subplot(2,1,2);
81     figure;
82     x=real(sg2(1:length(sg2)-Δ));
83     y=real(sg2(Δ+1:length(sg2)));
84     plot(x,y,'LineWidth',.7,'Color','r');
85     xlabel('x(t)','FontSize',12,'FontWeight','bold','Color','b');
86     ylabel('x(t+T)','FontSize',12,'FontWeight','bold','Color','b');
87     title('SP500 Filtered HP cycles','FontSize',12,'FontWeight','bold','Color','b');
88
89
90     sum(sum(sp500maskmatrix))
91     m*n
92
93
94     clear all;
95     [nas,syear]=getData(2);
96     nas=log(nas);
97     [s2,s1]=hpfilter(nas,1600);
98
99
100     Δm = 8;
101     M=16;
102     % M=50;
103     Δn=4;
104     nn=32;
105     %nn=100;
106     thrcont=1;
107     s1=s1';

```

```

108     L=length(s1);
109     t=1:L;
110     N=L/2;
111     nn2=nn/2;
112     sigma=sqrt((Δm*L)/(Δn * 2 * pi));
113     c=nthroot(pi*sigma*sigma,-4);
114     h0 = @(b) c*exp(-(b.*b)/(2*sigma*sigma));
115     h = @(ii) h0(mod(ii + N, L)-N);
116     for m = 1:M
117         for n = 1:nn2
118             c1(m, n)= sum(s1.*h(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
119         end
120
121     end
122     [m,n]=size(c1);
123     H=0.5;
124     %thr=max(abs(c1));
125     %spmask = (max(thr)-min(thr))/thrcont;
126     spmask=mean(c1)+H*std(c1);
127     for k = 1:m
128         for j = 1:n
129             if abs(c1(k,j)) < abs(spmask(k))
130                 % if abs(c1(k,j)) < spmask
131
132                 sp500maskmatrix(k,j)=0;
133             else
134                 sp500maskmatrix(k,j)=1;
135             end;
136         end;
137     end;
138
139     for t = 1:L
140         temp=0;
141         for m = 1:M
142             for n = 1:nn2
143                 temp= temp+c1(m,n).*h(mod(t - m*Δm,L)).*exp(2*pi*i*Δn*n*t/L);
144             end
145         end
146         sg(t)=temp;
147
148     end;
149     sg=sg/(2*pi);
150     c1=c1.*sp500maskmatrix;
151     for t = 1:L
152         temp=0;
153         for m = 1:M
154             for n = 1:nn2
155                 temp= temp+c1(m,n).*h(mod(t - m*Δm,M)).*exp(2*pi*i*Δn*n*t/L);
156             end
157         end
158         sg2(t)=temp;
159     end;
160     Δ=60;
161     figure;
162     x=real(sg2(1:length(sg2)-Δ));
163     y=real(sg2(Δ+1:length(sg2)));

```

```

164         plot(x,y,'LineWidth',.7,'Color','b');
165         xlabel('x(t)','FontSize',12,'FontWeight','bold','Color','r')
166         ylabel('x(t+T)','FontSize',12,'FontWeight','bold','Color','r');
167         title('NASDAQ Filtered HP cycles','FontSize',12,'FontWeight','bold','Color','r');
168
169         sum(sum(sp500maskmatrix))
170         m*n

```

```

1  function myFDfilter()
2  %Program used to Gabor Coefficients for SP500 & NASDAQ
3  % Praba Siva
4  % praba@umich.edu
5  % @prabasiva
6  % Filename: myFDfilter.m
7  close all;
8  clear all;
9  [sp500,syear]=getData(1);
10 process(sp500,1);
11 [naq,nyear]=getData(2);
12 process(naq,2);
13
14
15 function process(sp500,flag)
16
17 % First differencing
18 % Graph looks identical as shown in the P.Chen's paper
19 % P.Chen's paper says it is log of sp500 data but he used sp500
20 % The graph in his paper shows it all.
21
22 len=size(sp500);
23 t1=sp500(1:len-1);
24 t2=sp500(2:len);
25 fd=t2-t1;
26 lenfd=size(fd);
27 Δ=40;
28 x=fd(1:lenfd-Δ);
29 y=fd(Δ+1:lenfd);
30 figure;
31 subplot(2,1,1);
32 scatter(x,y,'b');
33 xlabel('X(t)','FontSize',20);
34 ylabel('X(t+T)', 'FontSize', 20);
35 if flag ==1
36     title('FD Series for SP500 Index','FontSize',20);
37 else
38     title('FD Series for NASDAQ Index','FontSize',20);
39 end;
40 y=fd;
41 n=size(y);
42 % C and Phi values are given in the P.Chen's paper.
43
44 c = 0.006*0.002;
45 phi = [0.000265*0.043, -0.81*0.043];

```

```

46
47     % White noise created based on the standard deviation value
48     noise = 0.033*randn(n(1),1);
49
50     for t=3:n
51         x(t-2) = c + phi(1)*y(t-1) + phi(2)*y(t-2) +noise(t-2);
52     end
53     lenfd=size(x);
54     Δ=5;
55     x1=x(1:lenfd-Δ);
56     y1=x(Δ+1:lenfd);
57     subplot(2,1,2);
58     scatter(x1,y1,'r');
59     xlabel('X(I)','FontSize',20);
60     ylabel('X(I+T)', 'FontSize', 20);
61     if flag ==1
62         title('AR(2) for FDSeries of SP500 index','FontSize',20);
63     else
64         title('AR(2) for FDSeries of NASDAQ index','FontSize',20);
65     end;
66 end
67 end

```



## APPENDIX C

### Mathematical Proof

#### C.1 Gabor Elementary function

$$\psi(t) = \underbrace{e^{-\alpha^2(t-t_0)^2}}_v \overbrace{e^{j2\pi f_0 t + \phi}}^w \quad (\text{C.1})$$

$v$  represents the probability function and  $w$  represents simple harmonic oscillator.  $\Psi(f)$  is the GEF in the frequency domain. The GEF in the frequency domain is attained by taking the Fourier transform of the GEF.

$$\Psi(f) = \int_{-\infty}^{\infty} \psi(t) e^{-j2\pi f t} dt; \Psi(f) = \int_{-\infty}^{\infty} e^{-\alpha^2(t-t_0)^2} e^{j2\pi f_0 t + \phi} e^{-j2\pi f t} dt$$

$$\Psi(f) = \int_{-\infty}^{\infty} e^{-\alpha^2(t-t_0)^2} e^{j2\pi t(f_0 - f) + \phi} dt$$

$$\Psi(f) = e^{\phi} \int_{-\infty}^{\infty} e^{-\alpha^2(t-t_0)^2} e^{j2\pi t(f_0 - f)} dt$$

when  $t_0$  is 0, then

$$\Psi(f) = e^{\phi} \int_{-\infty}^{\infty} e^{-\alpha^2 t^2} e^{j2\pi t(f_0 - f)} dt \quad (\text{C.2})$$

This is of the form.

$$\int_{-\infty}^{\infty} e^{2bx - ax^2} dx = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{a}}$$

where  $b = j\pi(f_0 - f)$  and  $a = \alpha^2$

$$\Psi(f) = \sqrt{\frac{\pi}{\alpha^2}} e^{\frac{(j\pi(f_0 - f))^2}{\alpha^2}} e^{\phi}$$

$$\Psi(f) = \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 (f_0 - f)^2 + \phi}$$

$\alpha$  is connecting the GEF between time and frequency domain.  $\psi(t)$  and  $\Psi(f)$  occupies the minimum uncertainty in time and frequency domain.

### C.1.1 Proof: GEF has minimum uncertainty in the time-frequency domain

I believe, we will better understand physical or mathematical concept by performing a step wise derivation. Let me do a step wise derivation to prove that GEF has a minimum uncertainty for a special case. Let me simplify the GEF by taking GEF at zero frequency,  $t_0 = 0$  and  $\phi = 0$ , the Gabor elementary function and Fourier transform of GEF are given by

$$\psi(t) = e^{-\alpha^2 t^2}$$

$$\Psi(f) = \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2}$$

Effective duration  $\Delta t$  is given by:

$$\Delta t = \sqrt{\frac{\int_{-\infty}^{\infty} e^{-\alpha^2 t^2} t^2 e^{-\alpha^2 t^2} dt}{\int_{-\infty}^{\infty} e^{-\alpha^2 t^2} e^{-\alpha^2 t^2} dt}};$$

Let me take the denominator first

$$\int_{-\infty}^{\infty} e^{-\alpha^2 t^2} e^{-\alpha^2 t^2} dt = \int_{-\infty}^{\infty} e^{-2\alpha^2 t^2} dt$$

The above equation is of the form and it only applies when  $a > 0$

$$\int_{-\infty}^{\infty} e^{-ax^2} dx = \sqrt{\frac{\pi}{a}}$$

where  $a = 2\alpha^2$

$$\int_{-\infty}^{\infty} e^{-2\alpha^2 t^2} dt = \sqrt{\frac{\pi}{2\alpha^2}}$$

Let me take the numerator now,

$$\int_{-\infty}^{\infty} e^{-\alpha^2 t^2} t^2 e^{-\alpha^2 t^2} dt = \int_{-\infty}^{\infty} t^2 e^{-2\alpha^2 t^2} dt$$

The above equation is of the form.

$$\int_{-\infty}^{\infty} x^2 e^{-ax^2} dx = \frac{1}{2} \sqrt{\frac{\pi}{a^3}}$$

where  $a = 2\alpha^2$

$$\int_{-\infty}^{\infty} t^2 e^{-2\alpha^2 t^2} dt = \frac{1}{2} \sqrt{\frac{\pi}{(2\alpha^2)^3}} = \frac{\sqrt{\pi}}{4\sqrt{2}\alpha^3}$$

Let me apply both numerator and denominator value to get the effective duration

$\Delta t$

$$\Delta t = \sqrt{\frac{\frac{\sqrt{\pi}}{4\sqrt{2}\alpha^3}}{\sqrt{\frac{\pi}{2\alpha^2}}}};$$

Straight forward steps to simplify the value of  $\Delta t$

$$\Delta t = \sqrt{\frac{\sqrt{\pi}}{4\sqrt{2}\alpha^3} \sqrt{\frac{2\alpha^2}{\pi}}};$$

$$\Delta t = \sqrt{\frac{\sqrt{\pi}}{4\sqrt{2}\alpha^3} \sqrt{\frac{2\alpha^2}{\pi}}};$$

$$\Delta t = \sqrt{\frac{1}{4\alpha^2}};$$

$$\Delta t = \frac{1}{2\alpha} \tag{C.3}$$

Let me do the similar steps to calculate the effective frequency  $\Delta f$ . The frequency representation of the GEF is given by,

$$\Psi(f) = \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2}$$

Effective frequency  $\Delta f$  is given by,

$$\Delta f = \sqrt{\frac{\int_{-\infty}^{\infty} \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} f^2 \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} df}{\int_{-\infty}^{\infty} \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} df}};$$

Let me take the denominator first.

$$\int_{-\infty}^{\infty} \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} \sqrt{\frac{\pi}{\alpha^2}} e^{-(\frac{\pi}{\alpha})^2 f^2} df = \frac{\pi}{\alpha^2} \int_{-\infty}^{\infty} e^{-2(\frac{\pi}{\alpha})^2 f^2} df$$

The above equation is of the form and it only applies when  $a > 0$

$$\int_{-\infty}^{\infty} e^{-ax^2} dx = \sqrt{\frac{\pi}{a}}$$

where  $a = 2\left(\frac{\pi}{\alpha}\right)^2$

$$\frac{\pi}{\alpha^2} \int_{-\infty}^{\infty} e^{-2\left(\frac{\pi}{\alpha}\right)^2 f^2} df = \frac{\pi}{\alpha^2} \sqrt{\frac{\pi}{2\left(\frac{\pi}{\alpha}\right)^2}}$$

Let  $\beta = \frac{\pi}{\alpha}$

$$\frac{\pi}{\alpha^2} \int_{-\infty}^{\infty} e^{-2\left(\frac{\pi}{\alpha}\right)^2 f^2} df = \frac{\beta}{\alpha} \sqrt{\frac{\pi}{2\beta^2}} = \frac{1}{\alpha} \sqrt{\frac{\pi}{2}}$$

Let me take the numerator now.

$$\begin{aligned} & \int_{-\infty}^{\infty} \sqrt{\frac{\pi}{\alpha^2}} e^{-\left(\frac{\pi}{\alpha}\right)^2 f^2} f^2 \sqrt{\frac{\pi}{\alpha^2}} e^{-\left(\frac{\pi}{\alpha}\right)^2 f^2} df \\ &= \frac{\pi}{\alpha^2} \int_{-\infty}^{\infty} f^2 e^{-2\left(\frac{\pi}{\alpha}\right)^2 f^2} df \end{aligned}$$

Substitute  $\beta$  in above equation.

$$= \frac{\beta}{\alpha} \int_{-\infty}^{\infty} f^2 e^{-2\beta^2 f^2} df$$

The above equation is of the form.

$$\int_{-\infty}^{\infty} x^2 e^{-ax^2} dx = \frac{1}{2} \sqrt{\frac{\pi}{a^3}}$$

where  $a = 2\beta^2$

$$= \frac{\beta}{\alpha} \frac{1}{2} \sqrt{\frac{\pi}{8\beta^6}}$$

$$= \frac{\beta}{\alpha} \frac{\sqrt{\pi}}{4\sqrt{2}\beta^3}$$

Substitute the value of  $\beta$

$$= \frac{1}{\alpha} \frac{\sqrt{\pi}}{4\sqrt{2}\beta^2} = \frac{1}{\alpha} \frac{\sqrt{\pi}\alpha^2}{4\sqrt{2}\pi^2} = \frac{\sqrt{\pi}\alpha}{4\sqrt{2}\pi^2}$$

Apply the value of numerator and denominator of  $\Delta f$

$$\Delta f = \sqrt{\frac{\frac{\sqrt{\pi}\alpha}{4\sqrt{2}\pi^2}}{\frac{1}{\alpha} \sqrt{\frac{\pi}{2}}}} = \sqrt{\frac{\sqrt{\pi}\alpha^2}{4\sqrt{2}\pi^2} \sqrt{\frac{2}{\pi}}}$$

Step wise simplification steps to get the value of  $\Delta f$

$$\Delta f = \sqrt{\frac{\alpha^2}{4\pi^2}}$$

$$\Delta f = \frac{\alpha}{2\pi} \tag{C.4}$$

Apply both the value of  $\Delta f$  and  $\Delta t$  from equation (8) and equation (9)

$$\Delta t \Delta f = \frac{\alpha}{2\pi} \frac{1}{2\alpha}$$

$$\boxed{\Delta t \Delta f = \frac{1}{4\pi}}$$

Hence the proof.

## BIBLIOGRAPHY

## BIBLIOGRAPHY

- [1] D. Gabor. Theory of Communication. *J. IEE*, 93(26):429–457, November 1946.
- [2] Ping Chen. A Random-Walk or Color-Chaos on the Stock Market? - Time-Frequency Analysis of S&P Indexes. *Studies in Nonlinear Dynamics & Econometrics*, 1(A9):87–103, July 1996.
- [3] Morten O. Ravn and Harald Uhlig. On Adjusting the Hodrick-Prescott filter for the frequency of observation. *Notes*, 1996.