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
2017

Advisor:

Professor Frank Massey, Chair


THE UNIVERSITY OF MICHIGAN
ARTES · SCIENTIA · VERITAS
1817

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 is 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

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

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

by

Prabaharan Sivashanmugam

Advisor: Professor Frank Massey

Color Chaos is a model between random-walk model and harmonic model introduced by the Prof. Ping Chen in the paper [2] published in Studies in Nonlinear Dynamics and Econometrics quarterly journal. By detail study of the paper [2], the color chaos model generates irregular oscillations with a narrow frequency band. The characteristic frequency is calculated from the Wigner decomposed distribution series. Standard and Poor stock price indexes and NASDAQ price indexes are used to study the color chaos model and detrended by using Hodrick-Prescott (HP) filters.

CHAPTER I

Introduction

1.0.1 Objective of the project

The object of the project is to study the color chaos model that was paper published by Prof. Ping Chen in the paper [2]. The approach of this paper is to understand the mathematical concepts of color chaos, time frequency analysis - Wigner distribution and Gabor transformation. In the paper [2], the S&P 500 composite monthly index price was taken for the analysis, and in this project the color chaos model is studied for both the S&P 500 and NASDAQ indices.

1.0.2 Organization of Project

This report has been organized into five chapters. Chapter 1 outlines the entire project giving an introduction to time series forecasting using stochastic models. Chapter 2 introduces Fourier transformation, discrete Fourier transformation, Gabor transformation and introduction to the joint time-frequency analysis. Chapter 3 provides an overview of Wigner distribution. Chapter 4 introduces the Time-frequency distribution series (TFDS) and chapter 5 introduces color chaos and conclusion of the study.

1.0.3 Data Source:

The Color chaos model is evaluated for two indices in this study and the sources of the data for the indices are given below in table 1.1.

Symbol	Description	Source	Frequency	Duration
FSPCOM	S&P 500 Price Composite index	Citibase	Monthly	1942-1992
NASDAQ	NASDAQ Composite index	Yahoo Finance	Monthly	1970-2010

Table 1.1: Meta data on FSPCOM(S&P 500) & NASDAQ

1.0.4 Time series forecasting using stochastic models

In general, models for time series data can have many forms and represent different stochastic processes. The most widely used linear time series models are Auto Regressive (AR) and Moving Average (MA) models. Combining these two, the Autoregressive and Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) are also used. The Autoregressive Fractionally Integrated Moving Average (ARFIMA) model generalized ARMA and ARIMA models. For seasonal time series forecasting, a variation of ARIMA, the Seasonal Autoregressive Integrated Moving Average (SARIMA) model is used.

Linear models have drawn much attention due to their relative simplicity in understanding and implementation. Many practical time series show non-linear patterns. Non-linear models are appropriate for predicting the volatility changes in economic and financial time series. Considering these facts, various non linear models have been proposed over the years. A few widely used non-linear models are Autoregressive Conditional Heteroskedasticity (ARCH) model, its variations like Generalized ARCH (GARCH), Exponential GARCH (EGARCH), the Threshold Autoregressive (TAR), the Non-linear AutoRegressive (NAR), the Non-linear Moving Average (NMA) model and others.

The Autoregressive Moving Average (ARMA) models: An ARMA(p,q) model is a combination of AR(p) and MA(q) models and is suitable for univariate time series modeling. In an AR(p) model the future value of a variable is assumed to be a linear combination of p past observations and a random error together with a constant term.

Mathematically the AR(p) model can be expressed as:

$$y_t = c + \sum_{i=1}^{p} \varphi_i y_{t-i} + \epsilon_t \tag{1.1}$$

Here y_t and ϵ_t are the actual value and random error respectively at time period t, $\varphi_i (i = 1, 2, 3, ..., p)$ are model parameters and c is a constant. The integer constant p is known as the order of the model. Sometimes the constant term is omitted for simplicity. A AR(p) model regresses against past values of the series; an MA(q) model uses past errors as the explanatory variables. The MA(q) model is given by

$$y_t = \mu + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{1.2}$$

Here μ is the mean of the series, $\theta_j (j = 1, 2, 3, ...q)$ are the model parameters and q is the order of the model. The random shocks are assumed to be a white noise process, i.e. a sequence of independent and identically distributed (i.i.d) random variables with zero mean and a constant variance σ^2. Generally, the random shocks are assumed to follow a normal distribution. Conceptually a moving average model is a linear regression of the current observation of the time series against the random shocks of one or more prior observations. Fitting an MA model to a time series is more complicated than fitting an AR model because in the former one the random error terms are not fore-seeable. Autoregressive (AR) and moving average (MA) models can be effectively combined together to form a general and useful class of time series models, known as ARMA models. Mathematically an ARMA(p, q) model is represented as:

$$y_t = c + \epsilon_t + \sum_{i=1}^{p} \varphi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} \tag{1.3}$$

Here the model orders p, q refer to p autoregressive and q moving average terms.

1.0.5 Difference Stationary

Loosely speaking a stationary process is one whose statistical properties do not change over time. More formally, a strictly stationary stochastic process is one where given $t_1, ...t_l$ the joint statistical distribution of $X_{t_1}, ..., X_{t_l}$ is the same as the joint statistical distribution of $X_{t_l+\tau}$ for all l and τ. It means that all moments of all degrees (expectations, variances, third order and higher) of the process anywhere are the same. It also means that the joint distribution of (X_t, X_s) is the same as $(X_{t+\tau}, X_{s+\tau})$.

A stochastic process is said to be stationary if its mean and variance are constant over time. i.e. time invariant. A stationary process will not drift too far away from its mean value because of the finite variance.

Difference Stationary: Sometimes even de-trending is not sufficient to make a time series stationary, in such case, it may be necessary to transform it into a series of period-to-period and/or season-to-season differences to make the series stationary. Such a series is said to be difference-stationary.

If the trend in a time series is a deterministic function of time, such as t or t^2, we call it a deterministic (predictable) trend. If it is not predictable, we have a stochastic trend.

Consider the following model.

$$Y_t = \alpha + \beta_1 t + \beta_2 Y_{t-1} + u_t \tag{1.4}$$

where u_t is white noise.

Pure Random Walk: $\alpha = 0$, β_1=0, and β_2=1. This is non stationary as we get $Y_t = Y_{t-1} + u_t$. The mean is constant over the time, but the variance is increasing linearly with time. If we find the difference, we get $\Delta Y_t = u_t$. Note that differenced series is stationary (DS) because $E(\Delta Y_t) = E(u_t) = 0$ and $Var(\Delta Y_t) = Var(u_t) = \sigma^2$. Both are time invariant. Hence, a random walk without a drift is difference-stationary(DS).

Random Walk with a drift: $\alpha \neq 0, \beta_1 = 0$, and β_2=1. This is non stationary as we get $Y_t = Y_{t-1} + u_t + \alpha$. The mean and variance are both increasing linearly with time. If we find the difference, we get $\Delta Y_t = \alpha + u_t$. Note that differenced series is stationary (DS) because $E(\Delta Y_t) = E(\alpha + u_t) = \alpha$ and $Var(\Delta Y_t) = Var(u_t) = \sigma^2$. Both are time invariant. Hence, a random walk with a drift is also difference-stationary(DS). Y_t is trending upward or downward depending on the sign of the drift (α).

Stationary Test: There are several tests of stationary available and we used Dickey Fuller test (DF test) to test stationary of the first difference for both indices. The $p-$value of DF test for both indices is 0.001 suggests that both indices are difference stationary. **Deterministic Trend:** $\alpha \neq 0, \beta_1 \neq 0$, and $\beta_2 = 0$. $Y_t = \alpha + \beta_1 t + \mu_t$. Note that the mean of the series, $E(Y_t) = E(\alpha + \beta_1 t) = \alpha + \beta_1 t$, which is time-varying but its variance, $Var(\Delta Y_t) = Var(\alpha + \beta_1 + u_t) = \sigma^2$ which is time-invariant. Still, the series with a deterministic trend is non-stationary. Once we know the values of α and β_1, we can subtract the mean from the series (detrending) and create a detrended series which is stationary.

Randow walk with drift and deterministic trend: $\alpha \neq 0, \beta_1 \neq 0$, and $\beta_2 = 1$. We get $Y_t = \alpha + \beta_1 t + Y_{t-1} + u_t$. Note that the difference series, $\Delta Y_t = \alpha + \beta_1 t + u_t$ is still time varying and hence, the mean of the differenced series is nonstationary. Detrending is still necessary on the differenced series to make it stationary.

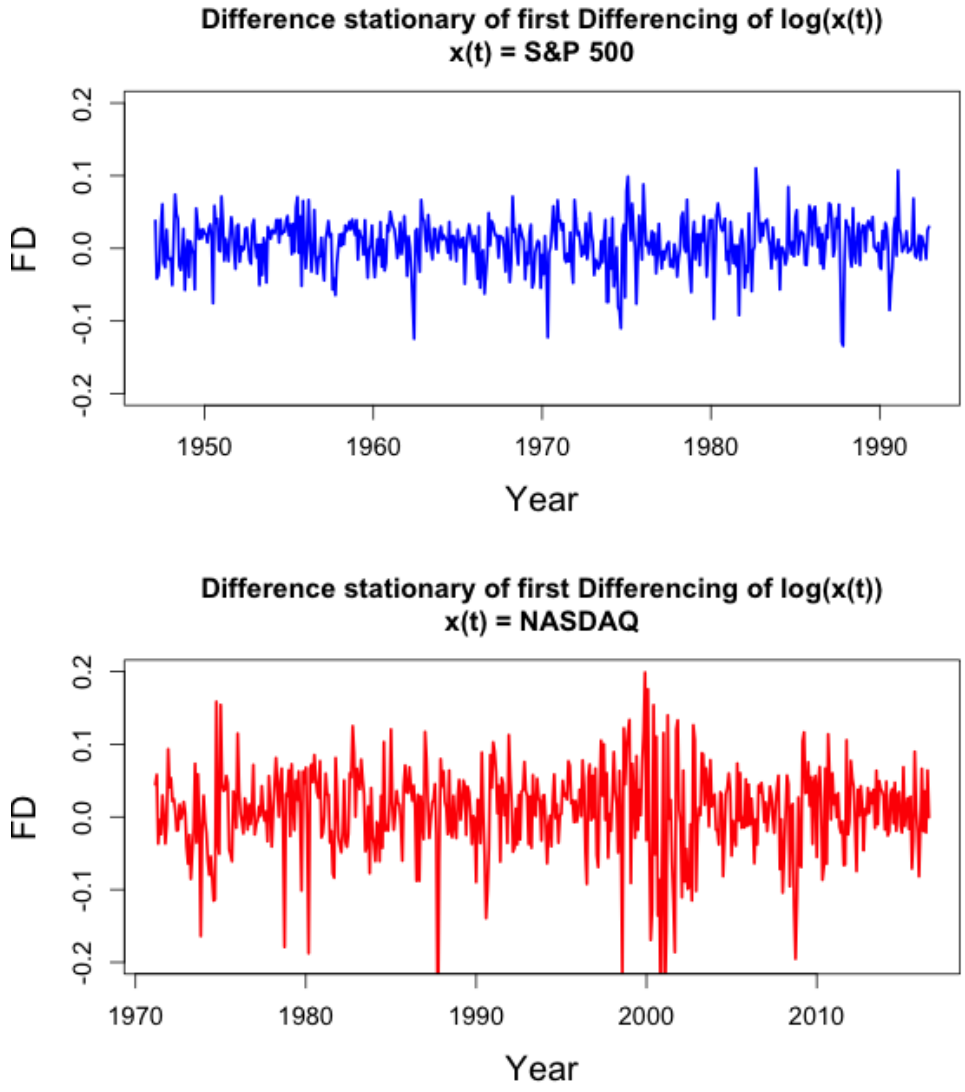

Figure 1.1: Difference stationary of natural log a) SP500 b) NASDAQ.The difference stationary [sp500(t2)-sp500(t1) and nasdaq(t2)-nasdaq(t1)] provides insights on the variation of signal. The R program fdplot.R used to generate the graph is given in Appendix A

1.0.6 Seasonal Trend Decomposition Procedure based on LOESS (STL)

STL is a filtering procedure for decomposing a time series into trend, seasonal, and remainder components. STL has a simple design that consists of a sequence of applications of the LOESS (LOcal regrESSion) smoother; the simplicity allows analysis of the properties of the procedure and allows fast computation, even for a

long time series and large amount of trend and seasonal smoothing. Other features of STL are specification of amounts of seasonal and trend smoothing that range, in a nearly continuous way, from a very small amount of smoothing to a very large amount; robust estimates of the trend and seasonal components that are not distorted by divergent behavior in the data.

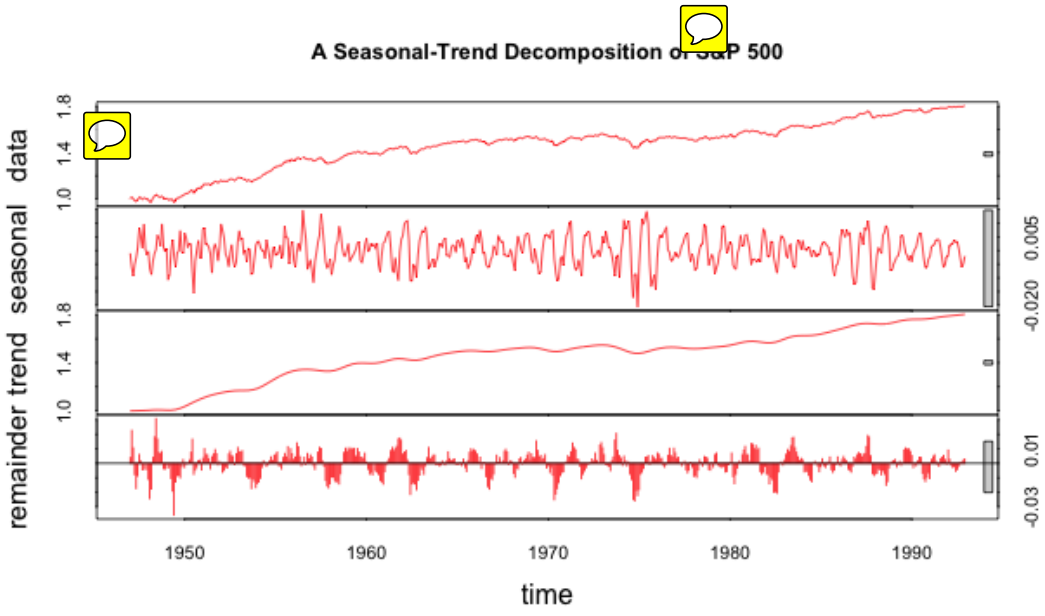

Figure 1.2: STL of Log SP500. a) The natural log of the SP500 b) The cyclical (or called seasonal) pattern c) The business trend of log SP500 d) Noise (remainder) data. The graph was created using the R program named as llt.R and it is attached in the Appendix A.

$$Y_t = f(S_t, T_t, E_t) \tag{1.5}$$

where Y_t is time series data at time t, S_t is seasonal component at time t, T_t is trend component at time t and E_t is remainder (or error or irregular) component of data at time t. For an additive decomposition by

$$Y_t = S_t + T_t + E_t \tag{1.6}$$

The user controls the variations on the trend and seasonal components.

In figure 1.2, trend, seasonal and noise or error remainder data of SP500 are extracted using the seasonal trend decomposition procedure. The seasonal window, in this case the value is done as 5, is used to control the variation on seasonal component.

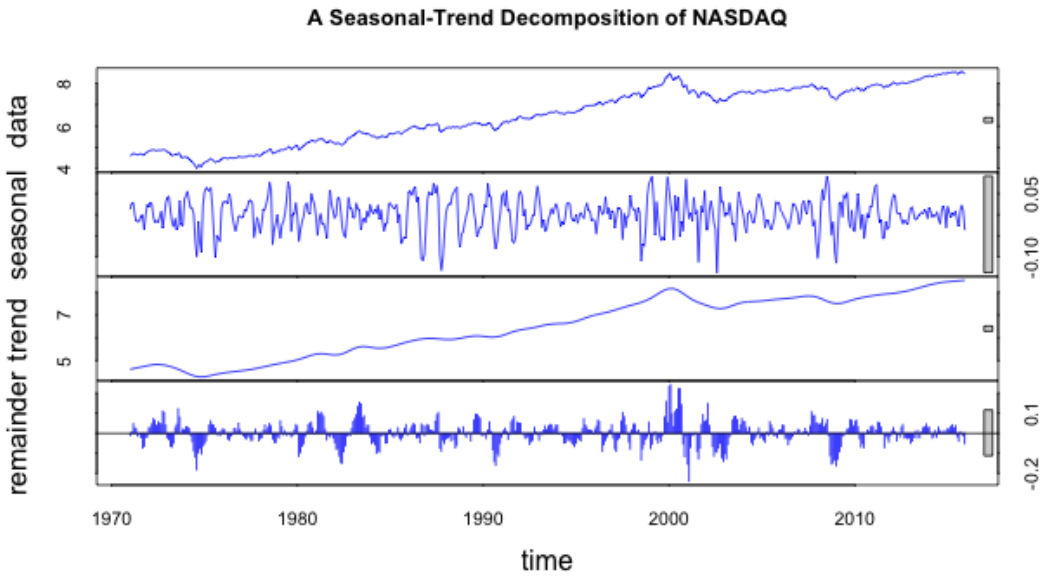

Figure 1.3: STL of NASDAQ. a) The natural log of the NASDAQ b) The cyclical (or called seasonal) pattern c) The business trend of log NASDAQ d) Noise (remainder) data. The graph was created using the R program named as llt.R and it is attached in the Appendix A

In the figure 1.3, trend, seasonal and noise or error remainder data of NASDAQ are extracted using the seasonal trend decomposition procedure. The seasonal window ($value = 5$) used to control the variation on seasonal component.

1.0.7 Log-Linear Method

When natural log values used for the dependent variable and independent variable in its original scale, those models are called log linear models.

The following model is of the value in a savings fund that depends on initial investment, growth rate and time duration in which the funds are invested.

$$Y_t = Y_0(1+r)^t \tag{1.7}$$

where Y_t represents the value of the fund at the time t, Y_0 is the initial investment in the saving fund, and r is the growth rate.

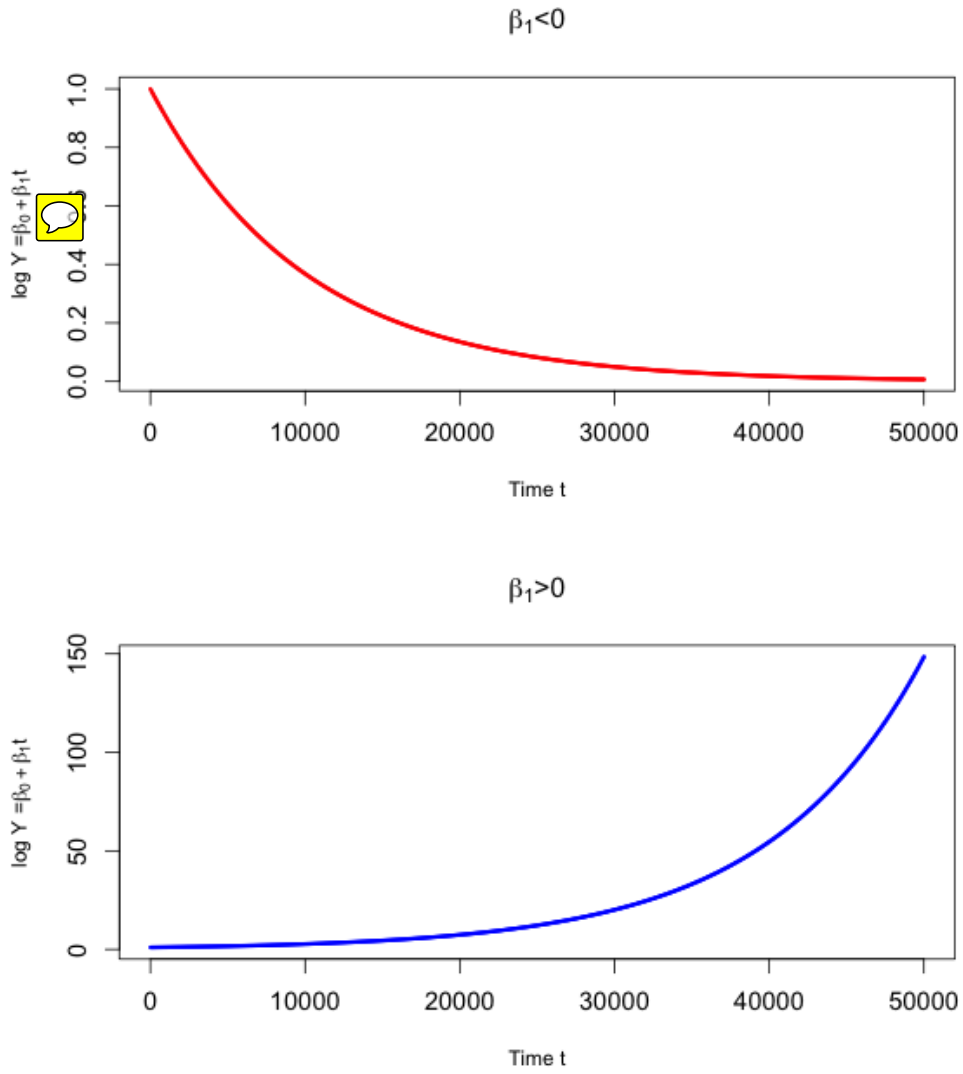

Figure 1.4: Log Linear Model when $\beta_1 < 0$ and when $\beta_1 > 0$. The graph was generated using loglinear.R, and it is attached in the Appendices A

Taking log on both side, we obtain the following.

$$log Y_t = logY_0 + tlog(1+r) \tag{1.8}$$

Here $logY_0$ is a constant and is taken to be β_0. Then $log(1+r)$ be β_1 and $logY_t$ given below:

$$logY_t = \beta_0 + \beta_1 t \tag{1.9}$$

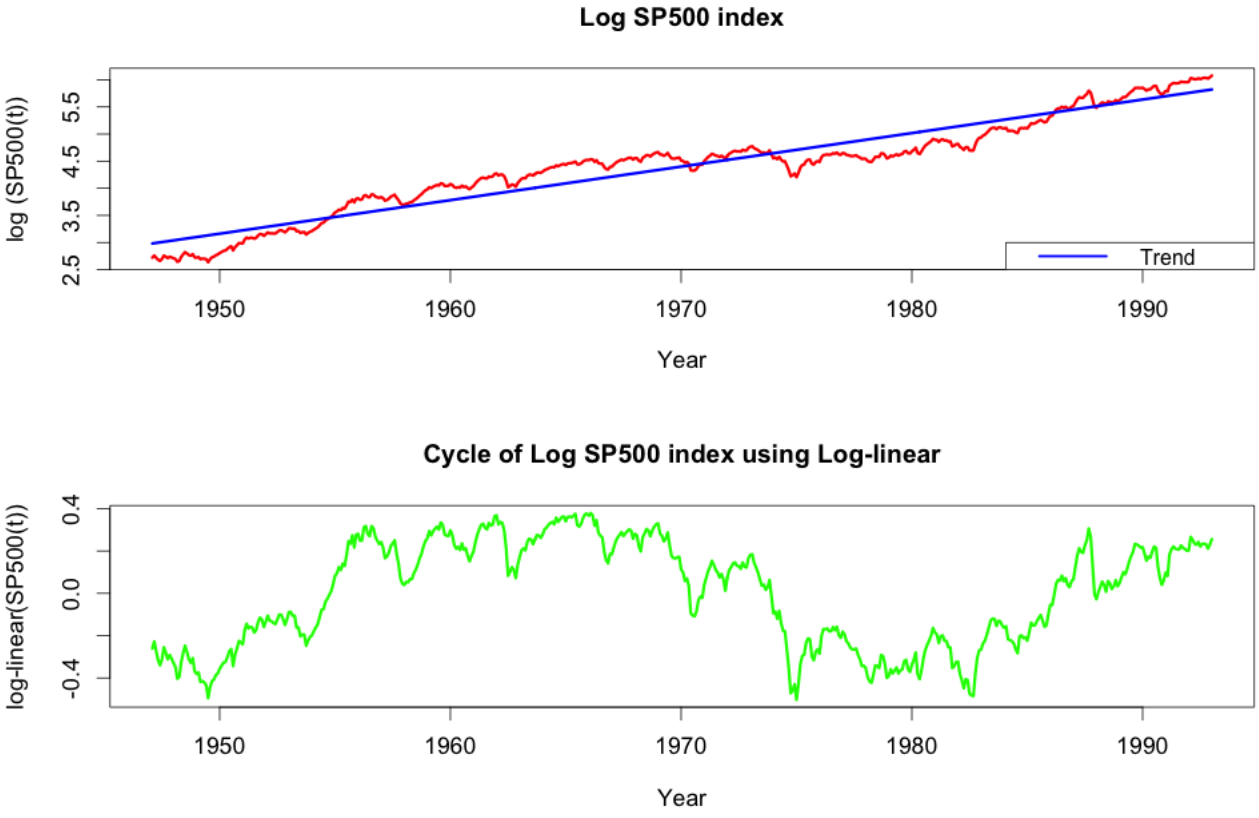

Figure 1.5: Log linear of trend and cycle for log SP500 a) Log SP500 and trend. The trend is calculated based on estimation of slope and y-intersect b)It is cycle of log-linear of SP500 and it is the variance of original log(sp500) and estimated trend. AutoCorrelation.R is the R program used to created the graph and it is attached in the Appendix A

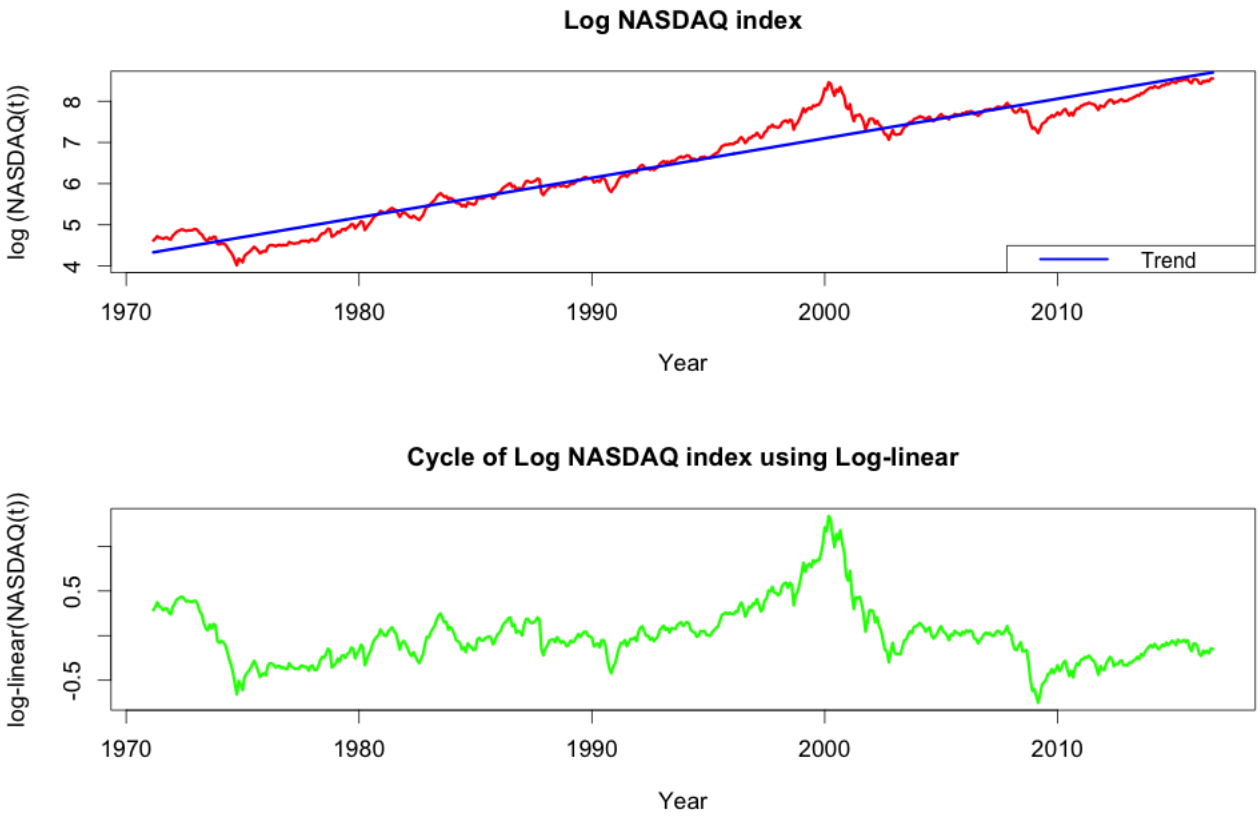

Figure 1.6: Log linear of trend and cycle for log NASDAQ a) Log NASDAQ and trend. The trend is calculated based on estimation of slope and y-intersect b)It is cycle of log-linear of SP500 and it is the difference between the original log(NASDAQ) and estimated trend. AutoCorrelation.R is the R program used to created the graph and it is attached in the Appendix A

After estimation of the log-linear model, the coefficients can be used to determine the impact of the independent variable (t) on the dependent variable (Y). The coefficient in a log-linear model represent the estimated percent change in the dependent variable for a unit change in independent variable. The coefficient β_1 provides the instantaneous rate of growth. The regression coefficients in log-linear model does not represent the slope.

When $\beta_1 > 0$, the log-linear function illustrates a positive impact from the independent variable and when, $\beta_1 < 0$, the log-linear function depicts a negative impact

from the independent variable.

1.0.8 Auto correlation Functions

An autoregressive model is when a value from a time series is regressed on previous value from that same time series. For exampe, y_t on y_{t-1}:

$$y_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t \tag{1.10}$$

In this regression model, the response variable in the previous time period has become the predictor and the errors have the same assumptions about errors in a simple linear regression model. The order of an autoregression is the number of immediate preceding values in the series that are used to predict the value at the present time. So, the preceding model is a first-order autoregression, written as $AR(1)$.

Let us say, if we want to predict the temperature y this year (y_t) using measurement of global temperature in the previous two years (y_{t-1}, y_{t-2}). Then the autoregressive model for doing so would be:

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \epsilon_t \tag{1.11}$$

The above model is a second-order autoregression, written as $AR(2)$, since the value at time t is predicted from the values at times $t-1$ and $t-2$. More generally, k^{th} order autoregression, written as $AR(k)$, is a multiple linear regression in which the value of the series at any time t is a linear function of the values at times $t-1, t-2, ...t-k$.

The cofficient of correlation between two values in a time series is called the **autocorrection function** (ACF). Let y_t be a sample, $t = 1, 2, .., n$ from an ARMA process of possibly unknown order, then the j^{th} order auto correlation $\rho(j)$ can be

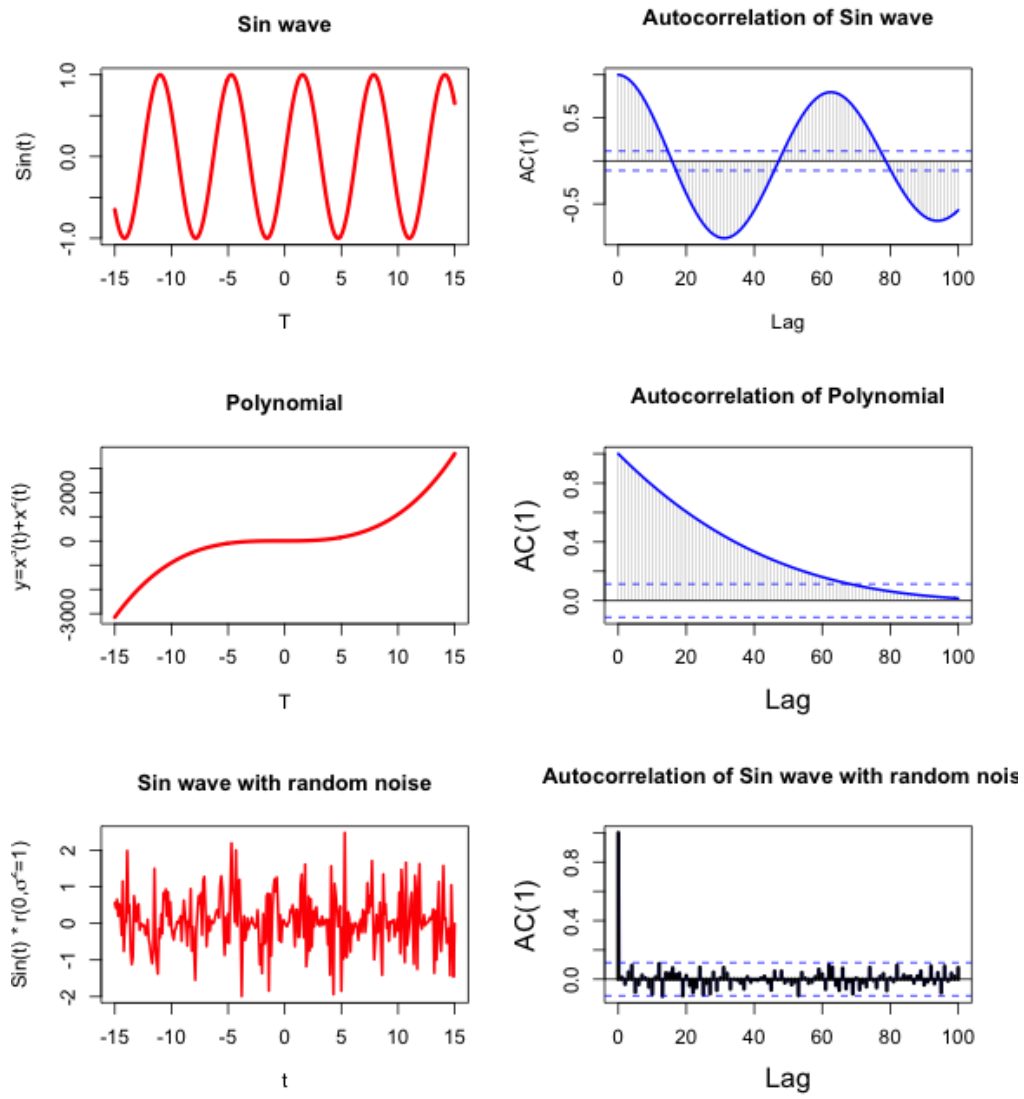

Figure 1.7: Auto correlation for a) Sin wave b) Polynomial equation c) Sin wave with random noise. AutoCorrelation.R is the R program used to created the graph and it is attached in the Appendix A

esimated by using the formula

$$\rho(j) = \frac{Cov(y_t, y_{t-j})}{Var(y_t)} \tag{1.12}$$

where

$$Cov(y_t, y_{t-j}) = \frac{1}{n-1} \sum_{t=j+1}^{n} (y_t - y_\mu)(y_{t-j} - y_\mu) \tag{1.13}$$

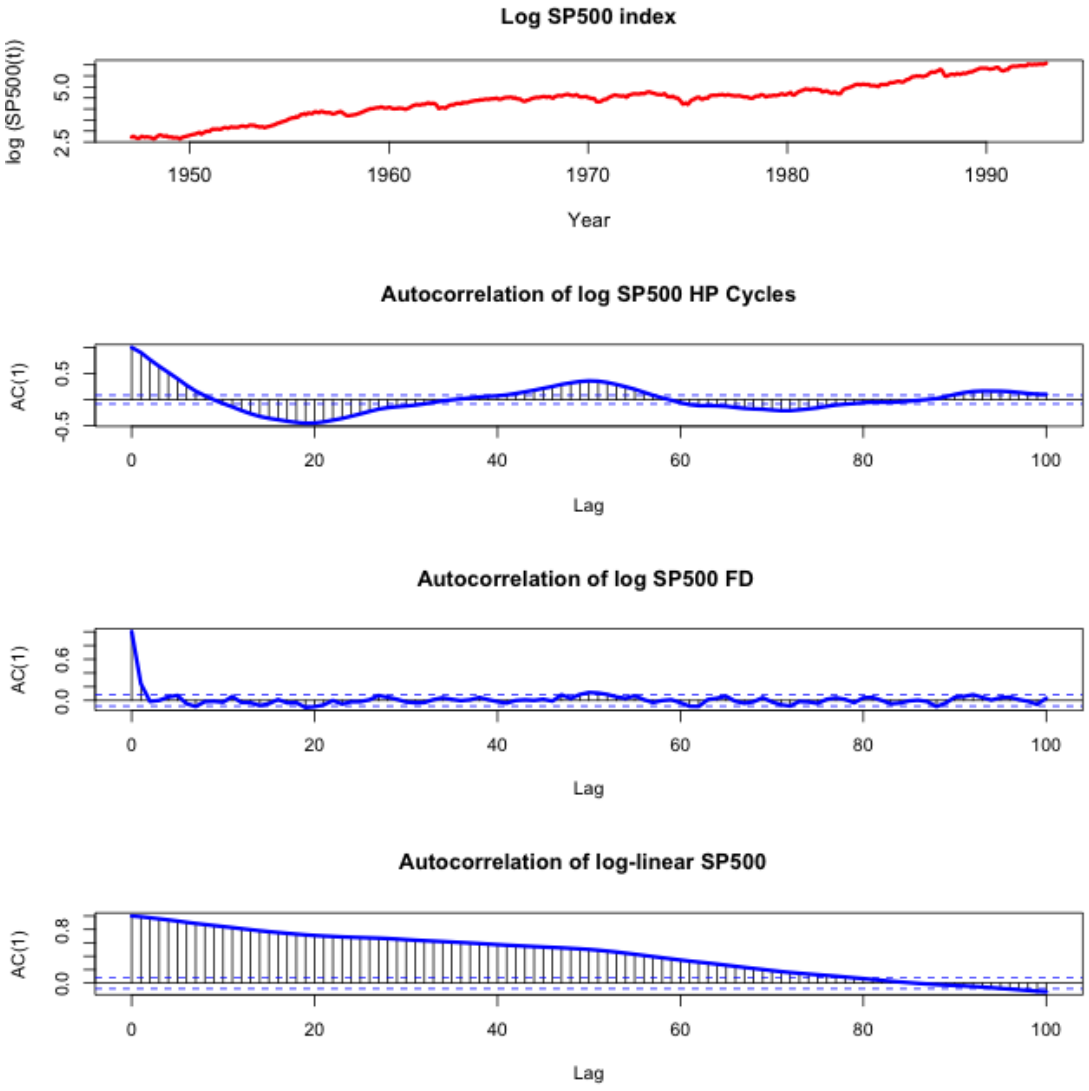

Figure 1.8: Auto correlation for log SP500 a) Log SP500 b)AutoCorrelation of HP cycle c) Autocorrelation of First Differencing d) Autocorrelation of log-linear SP500. AutoCorrelation.R is the R program used to created the graph and it is attached in the Appendix A

$$Var(y_t) = \frac{1}{n-1}\sum_{t=1}^{n}(y_t - y_\mu)^2 \tag{1.14}$$

y_μ is the mean of the y_t.

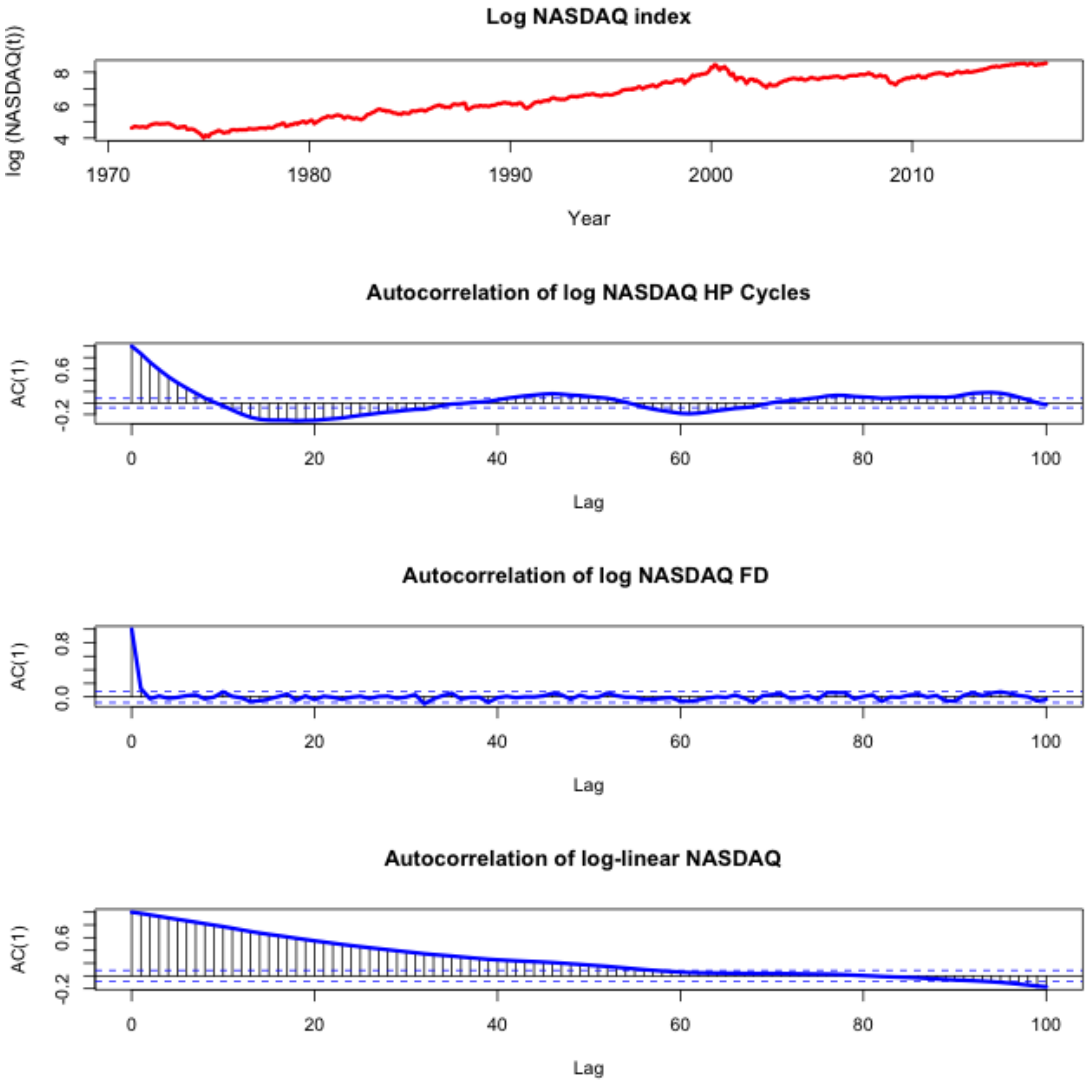

Figure 1.9: Auto correlation for log NASDAQ a) Log NASDAQ b)AutoCorrelation of HP cycle c) Autocorrelation of First Differencing d) Autocorrelation of log-linear NASDAQ. AutoCorrelation.R is the R program used to created the graph and it is attached in the Appendix A

1.0.9 Hodrick and Prescott (HP) filter

Hodrick and Prescott proposed the HP filter to decompose a macroeconomic time series into a non-stationary trend component and a stationary cyclical residual component. The filter has become popular in applied macro economics in the last 15 years. Given an observed series y_i, let $y_i = x_i + c_i$, with $y^T = (y_1, y_2..., y_N), x^T =$

$(x_1, x_2, ...x_N)$ and $c^T = (c_1, c_2...., c_N)$ where x_t denotes the unobserved trend component at time t and c_t the unobserved cyclical residual at time t. The HP trend $\hat{x}$ can be obtained as the solution to the following convex minimization problems:

$$\min_{[x_t]_{t=1}^N} \left[\sum_{t=1}^{N} (y_t - x_t)^2 + \lambda \sum_{t=2}^{N-1} ((x_{t+1} - x_t) - (x_t - x_{t-1}))^2 \right] \tag{1.15}$$

Here, λ is usually known as the smoothing parameter. As λ becomes larger, the HP estimated trend curve becomes smoother. The term being squared in the second sum of the equation, $(x_{t+1} - x_t) - (x_t - x_{t-1})$, or $\triangle^2 x_i$, is an approximation to the second derivate of x at time t. There are two opposing forces in the HP minimization problem. One force is attempting to minimize the sum of squared cyclical residuals and the other force is attempting to minimize the sum of squared $\triangle^2 x_i$. The smoothing parameter, λ, gives relative weight to these two opposing forces.

Detrending	Mean	SD	Variance	T_0 (month)	P_{dc} (year)
FD	0.011	0.1123	0.0126	1.94	00.7
HP	0.008	0.2686	0.0722	8.93	02.9
LLD	0.426	0.3265	0.1066	85.6	28.5

Table 1.2: Detrend Statistics on S&P 500. The value for the above table was created by using the Matlab program DetrendStatistics.m attached in the appendix.

Detrending	Mean	SD	Variance	T_0 (month)	P_{dc} (year)
FD	0.008	0.1072	0.0115	1.81	00.6
HP	0.022	0.2256	0.0509	9.22	03.0
LLD	0.258	0.3159	0.0992	80.5	26.8

Table 1.3: Detrend Statistics on NASDAQ. The value for the above table was created by using the Matlab program DetrendStatistics.m attached in the appendix.

The λ parameter determines the smoothness of the trend component. The larger the value of λ, the higher the penalty in the second term. An empirical study indicates that a 5% cyclical component is moderately large, as is a 1/8th of 1% change in the

growth rate in a quarter. In the paper [3], λ should be adjusted by multiplying it with the fourth power of the observation frequency ratios. This yields an HP parameter value of 6.25 for annual data given a value of 1600 for quarterly data.

1.0.9.1 First-order conditions of the HP minimization problem

The following HP first order conditions are derived by setting the gradient vector of the above minimization equation equal to zero. The first-order conditions are:

$$
\begin{aligned}
c_1 &= \lambda(x_1 - 2x_2 + x_3) \\
c_2 &= \lambda(-2x_1 + 5x_2 - 4x_3 + x_4) \\
c_t &= \lambda(x_{t-2} - 4x_{t-1} + 6x_t - 4x_{t+1} + x_{t+2}), t = 3, 4, 5, ..., N-2 \\
c_{N-1} &= \lambda(x_{N-3} - 4x_{N-2} + 5x_{N-1} - 2x_N) \\
c_N &= \lambda(x_{N-2} - 2x_{N-1} + x_N)
\end{aligned}
$$

or more compactly,

$$\boldsymbol{c = \lambda F x}$$

where F =

$$\begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & \cdots & \cdots & \cdots & & 0 \\ -2 & 5 & -4 & 1 & 0 & \cdots & \cdots & & & 0 \\ 1 & -4 & 6 & -4 & 1 & 0 & \cdots & & & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & \cdots & & 0 \\ 0 & 0 & 1 & -4 & 6 & -4 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & & & & & \vdots \\ \vdots & & & & & & & & & \vdots \\ \vdots & & & & & & & & & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & -4 & 6 & -4 & 1 & 0 \\ 0 & \cdots & & \cdots & 0 & 1 & -4 & 6 & -4 & 1 \\ 0 & \cdots & \cdots & & & 0 & 1 & -4 & 5 & -2 \\ 0 & \cdots & \cdots & \cdots & & & 0 & 1 & -2 & 1 \end{bmatrix}$$

which implies that

$$\boldsymbol{y} = (\boldsymbol{\lambda F} + \boldsymbol{I})\boldsymbol{x}$$

Thus, the HP trend is given by:

$$\hat{\boldsymbol{x}} = (\boldsymbol{\lambda F} + \boldsymbol{I})^{-1}\boldsymbol{y}$$

and

$$\hat{\boldsymbol{c}} = \boldsymbol{y} - \hat{\boldsymbol{x}}$$

In the figure 1.10, the λ value is 80 and the trend and cyclical information is separated from the natural log SP500 index and the trend data is useful for the long term investors like pension fund manager. The long term fund managers require the projection of the performance of an index to strategize the investment to meet the client's retirement objective.

In the figure 1.11, the SP500 trend analysis was performed using different λ values. The trend looks very similar when the λ is above 800 value and there is no significance

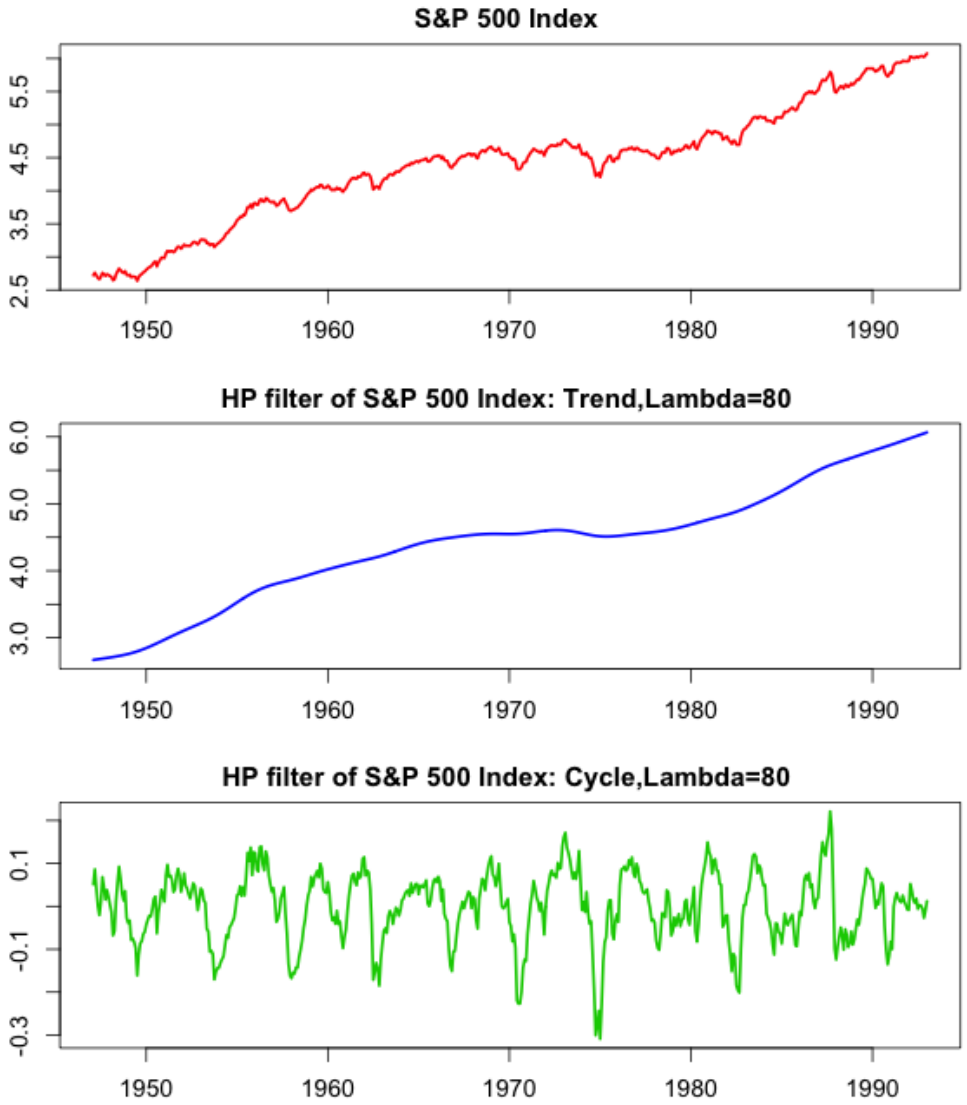

Figure 1.10: The trend and cycle separation from SP500 using HP filter with $\lambda = 80$. a) Natural Log of SP500 index b) Trend extracted from Log of SP500 index using HP filter c) Cyclical pattern extracted from SP500 index using HP filter

when the λ values are at 1600, 14400. The above figure 1.11, indicates that when λ is above 800 or above the trend in the HP filter is very close to least square linear regression line.

In the figure 1.12, the λ value is 800 and a similar analysis was performed.

In the figure 1.13, the λ value is 80 and the trend and cyclical information is separated from the natural log NASDAQ index.

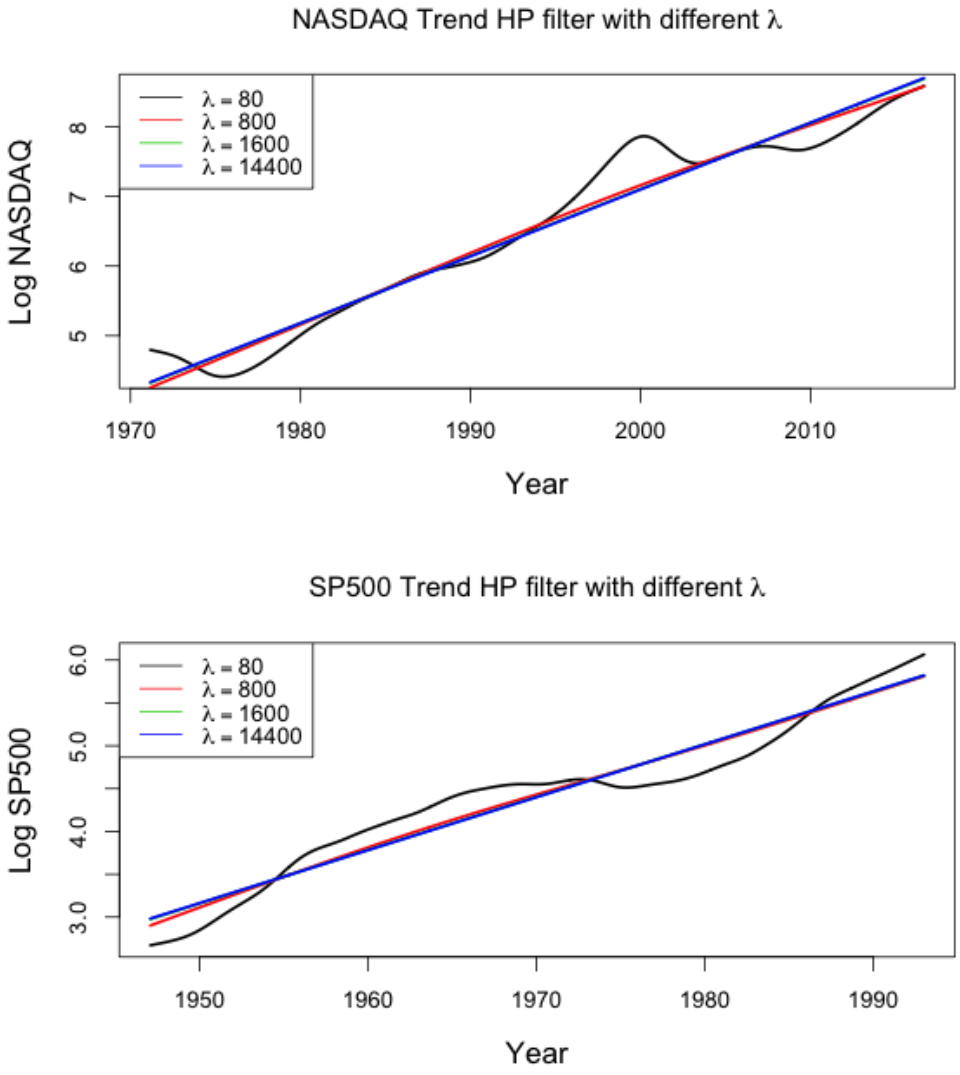

Figure 1.11: HP filter with different value for λ a) NASDAQ b) S&P 500

In the figure 1.11, the NASDAQ trend analysis was performed using different λ values. The trend looks very similar when the λ is above 800 value and there is no signficance when the λ values are at 1600, 14400.

The HP filters is one of the most heavily used econometric methods for measuring business cycles and potential output in empirical research. It is also a smoothing method that belongs to a very general class of nonparametric graduation procedures that depend on a tuning parameter governing the properties of the smoother. The

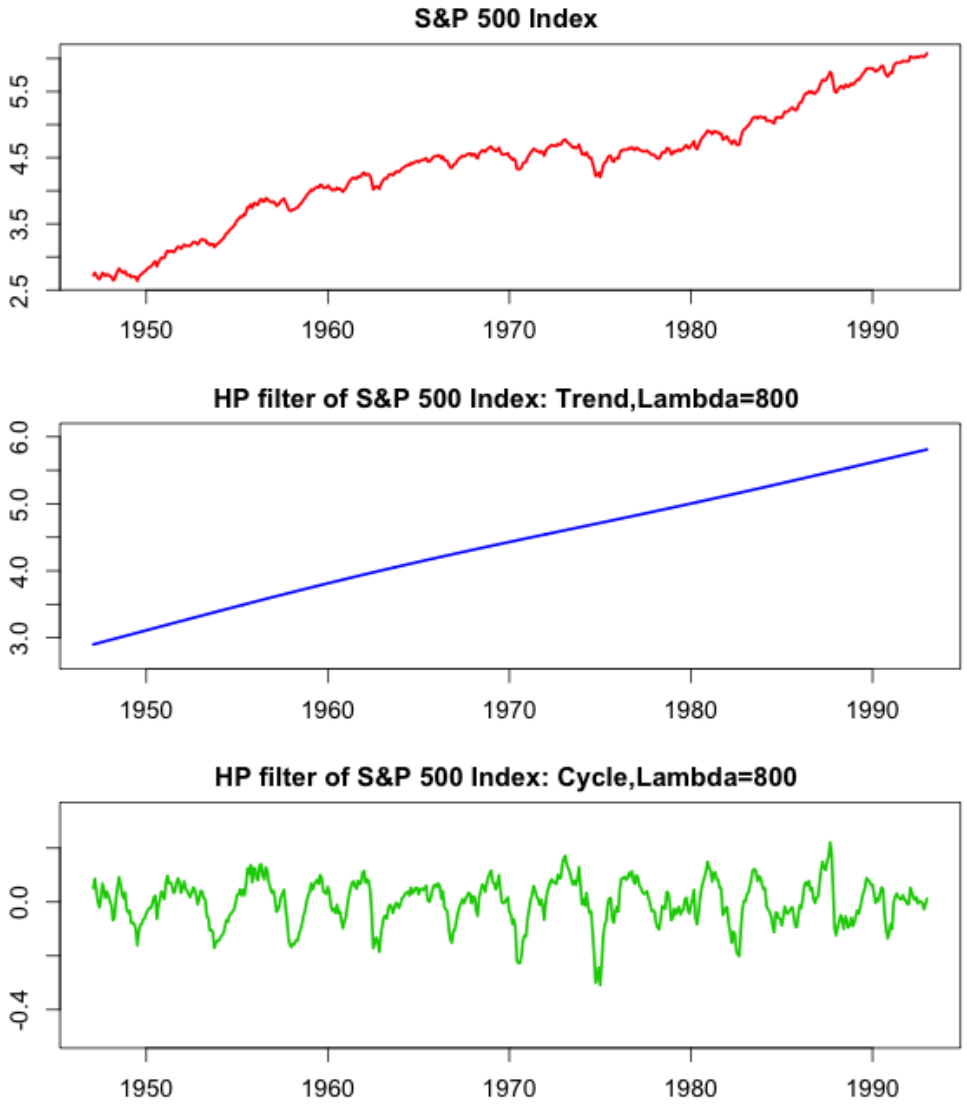

Figure 1.12: The trend and cycle separation from SP500 using HP filter with $\lambda = 800$. a) Natural Log of SP500 index b) Trend extracted from Log of SP500 index using HP filter c) Cyclical pattern extracted from SP500 index using HP filter

long run potential output of an economy can be substantially influenced by great recessions and depressions, which may sufficiently divert resources to impact long run trend components of output. The HP filter has the advantage that, depending on the smoothing parameter (λ) choice, it can encompass long run behavior that encompasses a vast range of possibilities -from a deterministic linear trend, to a smooth Gaussian process, through to stochastic trends and combination of stochastic

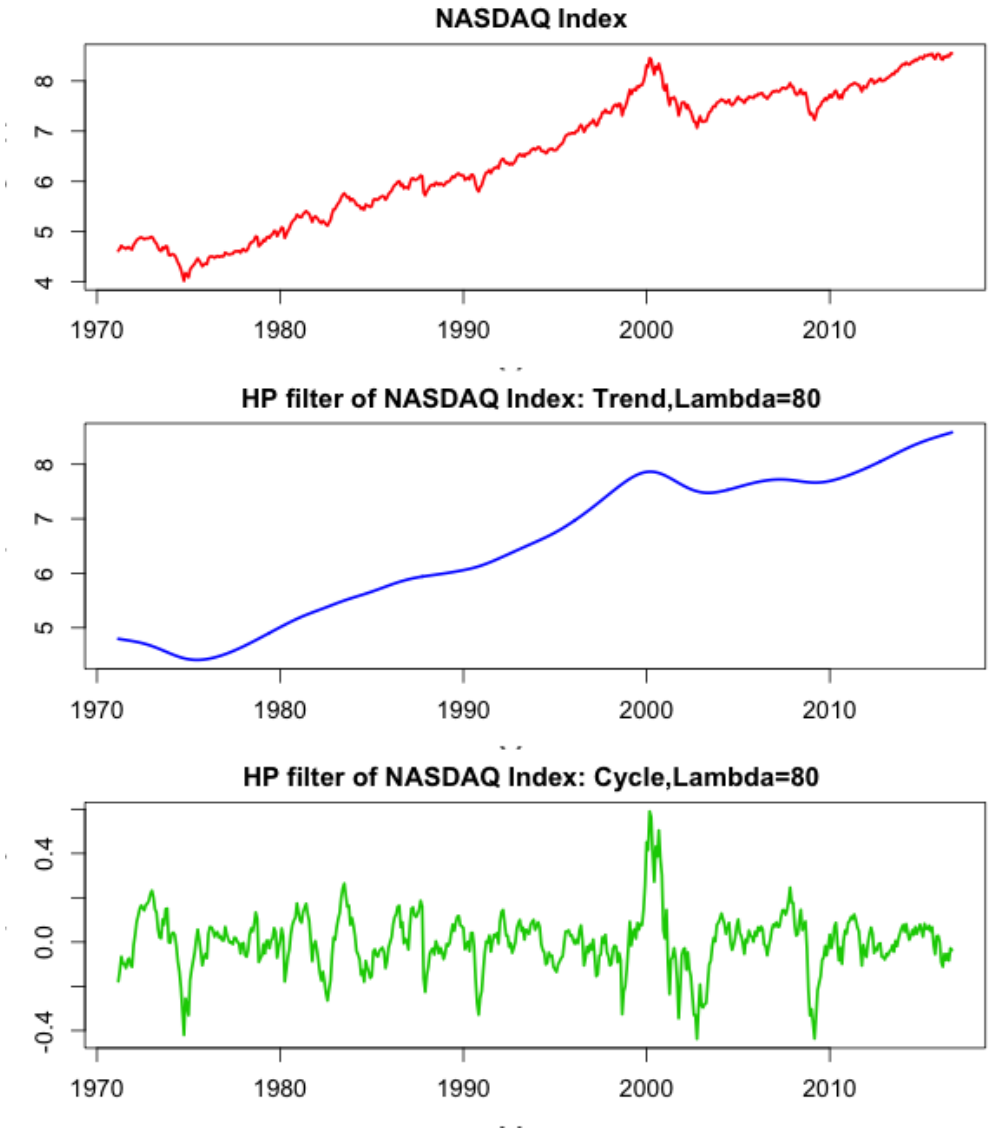

Figure 1.13: The trend and cycle separation from NASDAQ using HP filter with $\lambda = 80$. a) Natural Log of NASDAQ index b) Trend extracted from Log of NASDAQ index using HP filter c) Cyclical pattern extracted from NASDAQ index using HP filter

trends and deterministic trends that even include trend breaks.

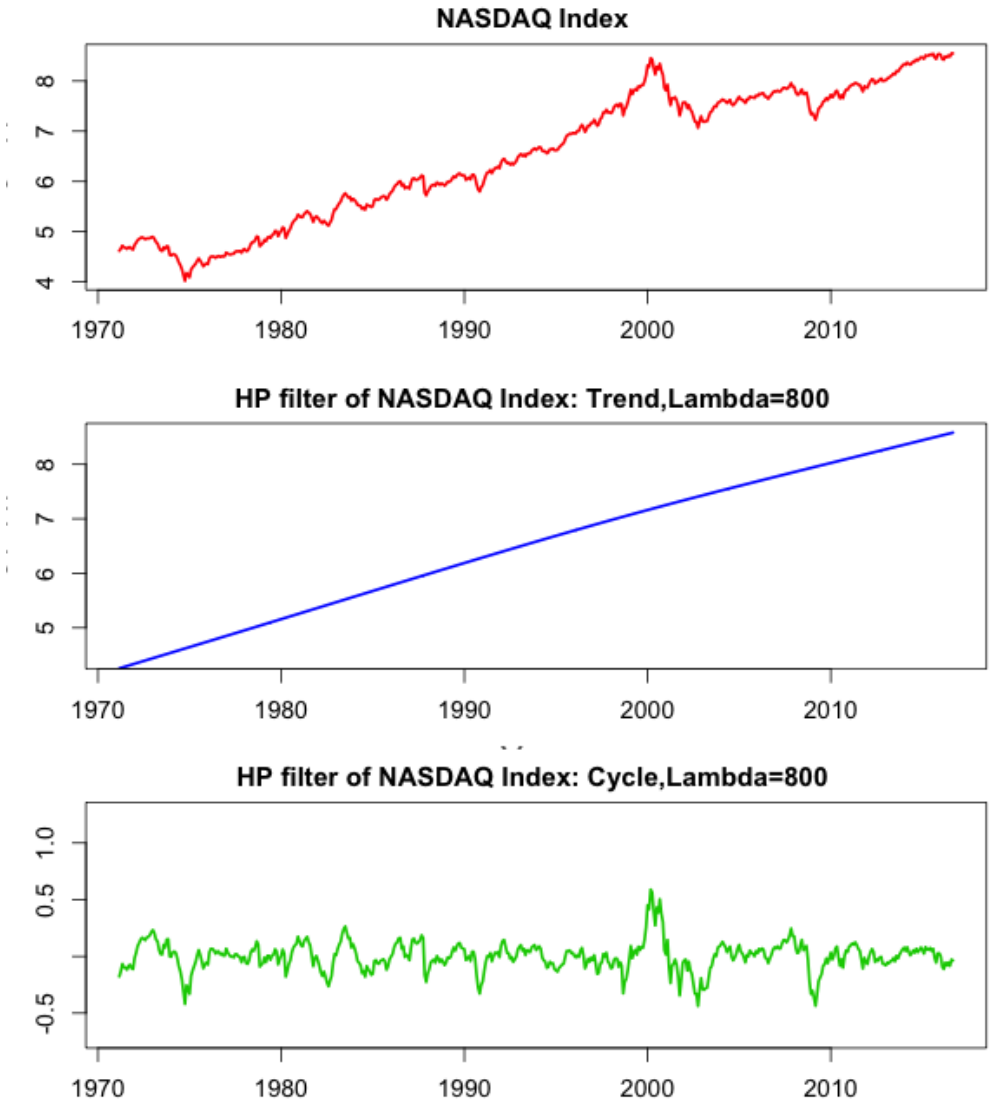

Figure 1.14: The trend and cycle separation from NASDAQ using HP filter with λ = 800. a) Natural Log of NASDAQ index b) Trend extracted from Log of NASDAQ index using HP filter c) Cyclical pattern extracted from NASDAQ index using HP filter

CHAPTER II

Gabor Transformation

2.1 Signal Processing

An arbitrary signal given by a function $f(t)$ can be represented by many forms for better understanding of the signal. The representation of the signal $f(t)$ in a different form depends on the type of application the user is interested in. In few cases, the original signals are perfectly fine as is for a given application.

In signal processing there are two major field  study.

- Signal synthesis - Construction of a signal
- Signal analysis - Study of a signal

Gabor in his original paper - Theory of communication [1], stated that a time function $f(t)$ in the time interval $\tau = \{s : t_1 < s < t_2\}$ contains an infinite amount of data. There are an infinite number of ways to represent $f(t)$ in the interval τ and one of the ways is to represent $f(t)$ in the interval τ by a polynomial of order N. For example, one can fit $f(t)$ as closely as possible by the method of least squares and take the coefficients of the polynomial as data. The polynomial coefficients of the curve is the data that represents the curve in the interval. It is equivalent to specifying the polynomial in such a way that its first $N + 1$ moments M_i shall be

equal to those of $f(t)$.

$$M_0 = \int_{t_1}^{t_2} f(t)dt;$$

$$M_1 = \int_{t_1}^{t_2} tf(t)dt;$$

$$M_2 = \int_{t_1}^{t_2} t^2 f(t)dt;$$

$$..$$

$$M_{N-1} = \int_{t_1}^{t_2} t^{N-1} f(t)dt$$

$$M_N = \int_{t_1}^{t_2} t^N f(t)dt$$

Instead of the polynomial coefficients, the moments are considered as the specific data. If the purpose is to transmit the signal, then the moments (equivalent of polynomial coefficients) can be transmitted and the signal can be reconstructed at the other end. Instead of representing the function $f(t)$ in the interval τ in terms of powers of time functions, it can be represented in terms of orthogonal functions $\phi_k(t)$ in the interval τ. How close the fit will be depends on the set of orthogonal functions selected and type of applications.

2.1.1 Sampling Theorem

The Sampling theorem states that for an accurate representation of a signal $f(t)$ by its time samples $f(nT)$, two conditions must be met.

- The signal $f(t)$ must be band limited, that is, its frequency spectrum must be limited to contain frequencies up to some maximum frequency, say S_{max}, and no frequencies beyond that.

- The sampling rate S_s must be chosen to be at least twice the maximum frequency S_{max}, that is, $S_s \geq 2S_{max}$ or, in terms of the sampling time interval: $T \leq \frac{1}{2S_{max}}$.

The minimum sampling rate allowed by the sampling theorem, that is, $S_s = 2S_{max}$, is called the **N***yquist* rate. For arbitrary values of S_s, the quantity $\frac{S_s}{2}$ is called the **N***yquist* frequency. It defines the endpoints of the Nyquist frequency interval $[-\frac{S_s}{2}, \frac{S_s}{2}]$.

2.1.2 Rayleigh Frequency

The lowest resolvable frequency interval is given by the inverse of the length of the analysis window and is denoted by the **R***ayleigh* frequency, $S_r = \frac{1}{L}$. The frequency resolution of any signal to be analyzed depends on the Rayleigh frequency of the data.

2.2 Fourier Transformation

If the orthogonal function set is simple harmonic functions sine and cosine in the interval extending from $-\infty$ to ∞, then the given signal is presented in the frequency domain and it is called the Fourier transformation.

$$F(s) = \int_{-\infty}^{\infty} f(t)e^{-j2\pi ts}dt \tag{2.1}$$

$F(s)$ is the Fourier transform of $f(t)$. The inverse Fourier transform function $F(s)$ provides the function $f(t)$ and the equation for this is given below.

$$f(t) = \int_{-\infty}^{\infty} F(s)e^{j2\pi ts}ds \tag{2.2}$$

Fourier transformation is a tool to translate a signal from time domain to frequency domain and vice versa. In fact, it is an apt tool to analyze a signal in the

frequency domain given the signal does not evolve over time, but, most of the practical signals like music, seismic and others evolve over time. It is challenging to study and understand the insights of those signals in the frequency domain by just using Fourier transformation. In recent years, there are many ideas proposed to overcome that challenge and gain more insights of the signals in other domains.

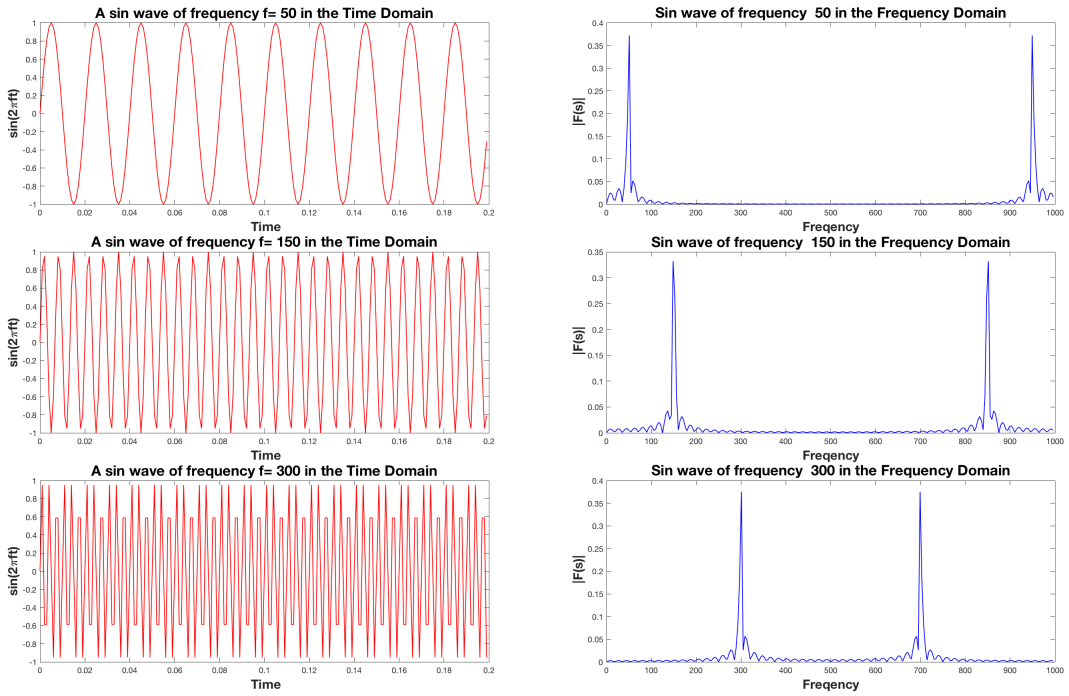

Figure 2.1: Fourier Transforms for discretized time signal sine waves with three different frequencies and respective discrete Fourier transforms are given above. The sampling period used is 0.001 and total length of signal (L) is 200. The graphs were created using Matlab code DrawSinFourierGraph.m and it is attached in the Appendix B

In the figure 2.1, a sine wave with different frequencies $f = 50, 150, 300$ is transformed into the frequency domain using the discrete Fourier transform. Please note high amplitude in the frequency domain for their respective frequency in each graph. $\omega = 2\pi f$.

$$f(t) = \sin(\omega t) \tag{2.3}$$

Three types of sine wave are created with frequencies $f = 50, 150, 300$. The Fourier transform of the $f(t)$ is given by

$$\begin{aligned} F(s) &= \int_{-\infty}^{\infty} \sin(2\pi f t) e^{-j2\pi t s} dt \\ \int_{-\infty}^{\infty} \sin(2\pi f t) e^{-j2\pi t s} dt &= \frac{\delta(t - 2\pi f) - \delta(t + 2\pi f)}{2j} \end{aligned} \tag{2.4}$$

Where $\delta(s)$ is the Dirac Delta function. The Dirac Delta function is defined as

$$\delta(t) = \begin{cases} +\infty & \text{if } t = 0 \\ 0 & \text{if } t \neq 0 \end{cases}$$

$$\int_{-\infty}^{\infty} \delta(t) dt = 1 \tag{2.5}$$

It follow that

$$\int_{-\infty}^{\infty} a\delta(t) dt = a \tag{2.6}$$

where a is a constant.

For a function $f(t)$, being integrable, we have that

$$\int_{-\infty}^{\infty} f(t)\delta(t) dt = f(0) \tag{2.7}$$

It denotes that the integral of any function multiplied by a δ-function located about zero is just the value of the function at zero. This concept can be extended to give

the shifting property, again for a function $f(t)$, giving,

$$\int_{-\infty}^{\infty} f(t)\delta(t-a)dt = f(a) \tag{2.8}$$

where $\delta(t-a)$ is just a δ-function located at $t=a$. In two dimensions, for a function $f(t,w)$, we have that,

$$\int\int f(t,w)\delta(t-a,w-b)dtdw = f(a,b) \tag{2.9}$$

where $\delta(t-a,w-b)$ is a δ-function located at position a,b. This property is central to the idea of convolution, which is used extensively in image formation theory, and in digital image processing. The Fourier transform of a δ function can be formed by direct integration using the definition of the Fourier transform, and the shift property. We get,

$$\begin{aligned} F(\delta(t)) &= \int_{-\infty}^{\infty} \delta(t)e^{-j2\pi st}dt \\ &= e^{0} \\ &= 1 \end{aligned} \tag{2.10}$$

2.2.1 Shifting Theorem

Let $f(t)$ be a function and its Fourier transform is given by $F(s)$. Fourier transform of a function $f(t-t_0)$ is given $e^{-j2\pi st_0}F(s)$. The proof of the shift theorem is given below. $F[f(t-t_0)](s) = e^{-j2\pi st_0}F(s)$

$$F[f(t-t_0)](s) = \int_{-\infty}^{\infty} f(t-t_0)e^{-j2\pi st}dt \tag{2.11}$$

Multiply $e^{j2\pi st_0}e^{-j2\pi st_0}$ on right hand side of the above equation.

$$\begin{aligned} F[f(t-t_0)](s) &= \int_{-\infty}^{\infty} f(t-t_0)e^{-j2\pi st}e^{j2\pi st_0}e^{-j2\pi st_0}dt \\ &= e^{-j2\pi st_0}\int_{-\infty}^{\infty} f(t-t_0)e^{-j2\pi s(t-t_0)}dt \end{aligned} \tag{2.12}$$

Let $u = t - t_0$

$$\begin{aligned} F[f(t-t_0)](s) &= \int_{-\infty}^{\infty} f(u)e^{-j2\pi su}du \\ &= e^{-j2\pi st_0}F(s) \end{aligned} \tag{2.13}$$

By applying the shifting theorem in the equation 2.10,we get that,

$$F(\delta(t-a)) = e^{-j2\pi sa} \tag{2.14}$$

so that the Fourier transform of a shifted δ function is given by a phase ramp. The modulus squared of above equation is

$$\begin{aligned} |F(\delta(t-a))|^2 &= |e^{-j2\pi sa}|^2 \\ &= 1 \end{aligned} \tag{2.15}$$

The power spectrum of a δ function is constant independent of its location in real space. Noting that the Fourier transform is a linear operator, consider two δ functions

located at $\pm a$. Then the Fourier transform of $\delta(t-a)+\delta(t+a)$ is given by

$$\begin{aligned} F(\delta(t-a)+\delta(t+a)) &= e^{-j2\pi as}+e^{j2\pi as} \\ &= 2\cos(2\pi as) \\ \int_{-\infty}^{\infty}\cos(2\pi at)e^{-j2\pi ts} &= \frac{\delta(t-a)+\delta(t+a)}{2} \end{aligned} \tag{2.16}$$

If we take the δ-function at $t=-a$ as negative, the Fourier transform is given below

$$\begin{aligned} F(\delta(t-a)-\delta(t+a)) &= e^{-j2\pi as}-e^{j2\pi as} \\ &= 2j\sin(2\pi as) \\ \int_{-\infty}^{\infty}\sin(2\pi at)e^{-j2\pi ts} &= \frac{\delta(t-a)-\delta(t+a)}{2j} \end{aligned} \tag{2.17}$$

In the figure 2.2, a signal with zero mean random noise, signal with zero mean with no random noise, Gaussian curve, *log* sp500, *log* Nasdaq are also transformed into a frequency domain using the discrete Fourier transform. $|F(s)|$ is an even function and so only $s>0$ is shown in figure 2.2.

2.2.2 Discrete Fourier Transform

The Discrete Fourier Transform (DFT) is the equivalent of the continuous Fourier Transform for signals known only at N instants separated by sample times T (i.e a finite sequence of data). Let $f(k)$ be a continuous signal and the N samples be denoted by $f(0), f(1), f(2)...f(k), ...f(N-1)$. Each sample $f(k)$ is regarded as an impulse having area $f(k)$ and the integrand exists only at the sample points. Since there are only a finite number of input data points, the DFT treats the data as if it were periodic i.e $f(N)$ to $f(2N-1)$ is the same as $f(0)$ to $f(N-1)$. In general, the

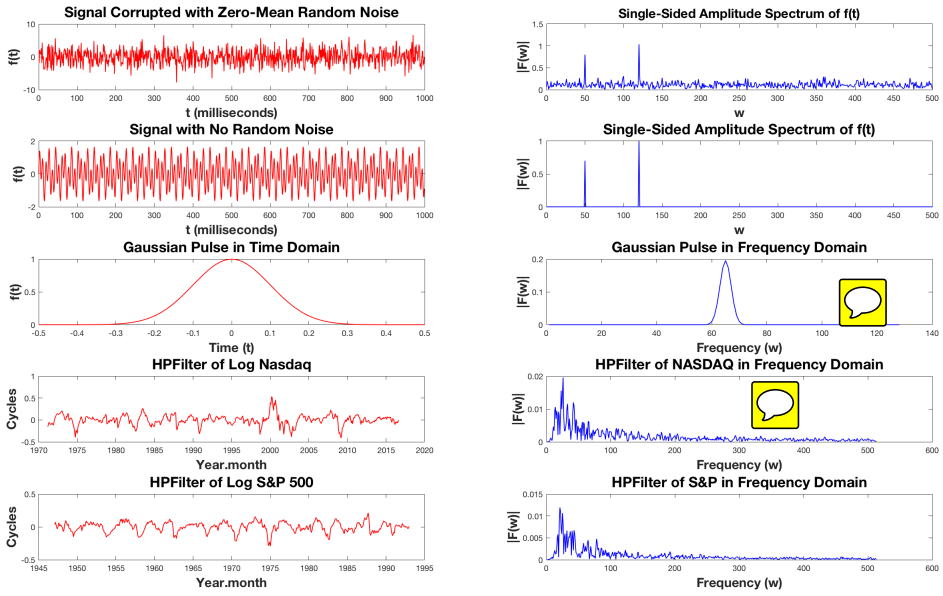

Figure 2.2: Fourier transform for a) Zero mean signal with random noise, b) Zero mean signal $f(t) = 0.7sin(2\pi 50t) + sin(2\pi 120t)$, c) Gaussian - Frequency are shifted (using fftshift function in matlab) in the frequency domain to show that the Fourier transform of Gaussian looks like Gaussian in the frequency domain, d) Log NASDAQ, e) Log S&P 500. The program used to generate the graph is SPNASDAQFourier.m and stored in Appendix

discrete Fourier Transform is given by

$$F(s) = \sum_{k=0}^{N-1} f(k) e^{-j\frac{2\pi}{N}sk} \tag{2.18}$$

$s = 0, 1, .., N-1$.

$$e^{j\frac{2\pi}{N}} = \cos(\frac{2\pi}{N}) + j\sin(\frac{2\pi}{N}) \tag{2.19}$$

is the principal N-th root of unity. Since $e^{j\frac{2\pi s}{N}}$ has a period of N, the DFT coefficients $F(s)$ are periodic with period N when s is taken outside the range $s = 0, 1, 2, ...N-1$. $F(s)$ is the Discrete Fourier Transform of the sequence $f(t)$. The matrix-vector format of the Discrete Fourier Transform are given by

$$\underbrace{\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ \vdots \\ \vdots \\ F(N-1) \end{bmatrix}}_{\text{DFT Vector}} = \underbrace{\begin{bmatrix} 1 & 1 & \cdots & \cdots & 1 \\ 1 & e^{-j(\frac{2\pi}{N})} & e^{-j(\frac{4\pi}{N})}\cdots & \cdots & e^{-j\frac{2(N-1)\pi}{N}} \\ 1 & e^{-j(\frac{4\pi}{N})} & e^{-j(\frac{8\pi}{N})}\cdots & \cdots & e^{-j\frac{4(N-1)\pi}{N}} \\ \vdots & & & \vdots & \\ \vdots & & & \vdots & \\ 1 & e^{-j(\frac{2(N-1)\pi}{N})} & e^{-j(\frac{4(N-1)\pi}{N})}\cdots & \cdots & e^{-j\frac{2(N-1)(N-1)\pi}{N}} \end{bmatrix}}_{\text{DFT Matrix:F}} \underbrace{\begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ \vdots \\ f(N-1) \end{bmatrix}}_{\text{Input Signal}}$$

The above matrix multiplication can also be used to compute the Discrete Fourier Transform of a given signal $f(t)$. The Discrete Fourier Transform matrix F can be precomputed as it is independent of the input signal.

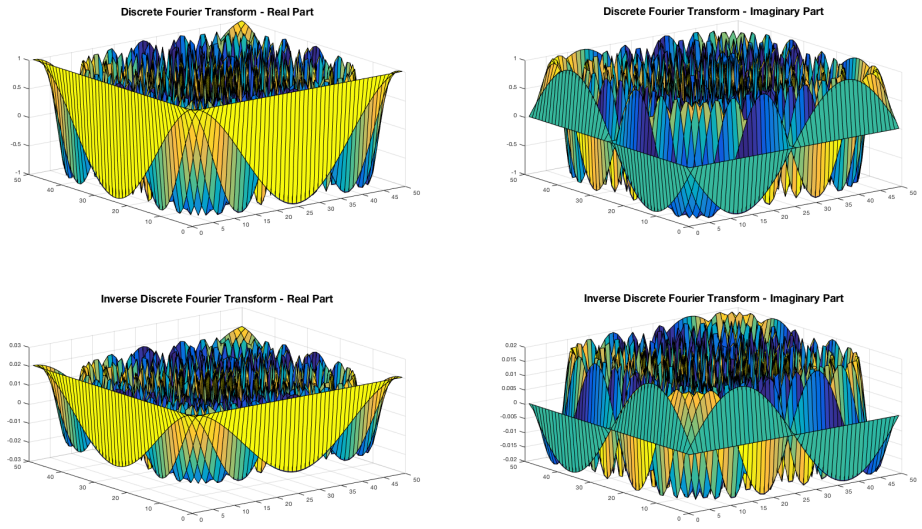

Figure 2.3: Discrete Fourier Transform & Inverse DFT real and imaginary part. The above surface created using the matrix F. The graph is created using Matlab program named mydft.m attached in the Appendix

2.2.2.1 Inverse Discrete Fourier Transform

The inverse transform of Discrete Fourier Transform is given by

$$f(t) = \frac{1}{N}\sum_{k=0}^{N-1} F(k)e^{j\frac{2\pi}{N}tk} \tag{2.20}$$

The matrix-vector format of the Inverse Discrete Fourier Transform are given by

$$\underbrace{\begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ \vdots \\ f(N-1) \end{bmatrix}}_{\text{Original Signal}} = \frac{1}{N} \underbrace{\begin{bmatrix} 1 & 1 & \cdots & \cdots & 1 \\ 1 & e^{j(\frac{2\pi}{N})} & e^{j(\frac{4\pi}{N})}\cdots & \cdots & e^{j(\frac{2(N-1)\pi}{N}} \\ 1 & e^{j(\frac{4\pi}{N})} & e^{j(\frac{8\pi}{N})}\cdots & \cdots & e^{j(\frac{4(N-1)\pi}{N}} \\ \vdots & & & \vdots & \\ \vdots & & & \vdots & \\ 1 & e^{j(\frac{2(N-1)\pi}{N})} & e^{j(\frac{4(N-1)\pi}{N})}\cdots & \cdots & e^{j(\frac{2(N-1)(N-1)\pi}{N}}) \end{bmatrix}}_{\text{Inverse DFT Matrix:G}} \underbrace{\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ \vdots \\ \vdots \\ F(N-1) \end{bmatrix}}_{\text{DFT Vector}}$$

The inverse DFT matrix G is inverse of matrix F and scaled by a factor of $\frac{1}{N}$. $G * F = F * G = I$, where as I is an identify matrix of order N.

2.2.3 Short Time Fourier Transformation

One of the ideas is to chop the signal into smaller pieces and perform Fourier transformation for each piece. This technique is called short time Fourier Transform. The smaller pieces in the signal can be chosen by a window function $w(t-\tau)$

$$F(\tau, s) = \int_{-\infty}^{\infty} f(t)w(t-\tau)e^{-j2\pi ts}dt \tag{2.21}$$

$w(t)$ represents a window function and there are multiple window functions available to choose.

2.2.3.1 Gaussian Window

The Gaussian Window is defined as

$$w(n) = e^{\frac{-(n-m)^2}{2(\sigma N)^2}} \tag{2.22}$$

for $n = 0, 1, 2, ...N-1$,where N is the length of the window, $m = \frac{(N-1)}{2.0}$, and the σ is the standard deviation of the Gaussian window. The Gaussian window is useful for time-frequency analysis because the Fourier transform and the derivate of a Gaussian window both are Gaussian function.

2.2.3.2 Hamming Window

The Hamming window is defined as

$$w(n) = \alpha - \beta \cos(\frac{2\pi n}{N-1}) \tag{2.23}$$

where the constants α and β are approximations to the values of $\frac{25}{46}$ and $\frac{21}{46}$ respectively. The values of α and β are given as 0.54 and 0.46 respectively.

$$w(n) = 0.54 - 0.46 \cos(\frac{2\pi n}{N-1}), 0 \leq n \leq N \tag{2.24}$$

The window length is $L = N + 1$.

The short time Fourier transform is performed for several signals using Hamming and Gaussian windows and the results are given below.

$$f(t) = \begin{cases} sin(2\pi 300t) & \text{if} \quad 0 < t < \dfrac{T}{3} \\ sin(2\pi 200t) & \text{if} \quad \dfrac{T}{3} + 1 < t < \dfrac{2T}{3} \\ sin(2\pi 100t) & \text{if} \quad \dfrac{2T}{3} + 1 < t < T \end{cases} \tag{2.25}$$

Where T is the complete duration of $f(t)$. Operations like Fourier Transformation, Short Time Fourier Transform (STFT), STFT with Gaussian and Hamming windows are performed for the function given in 2.25 and results of it is compared with similar operations for the SP500 Cycles, NASDAQ Cycles and results are given

Simultaneous analysis of signals in both time and frequency domains provides better understanding of the signal and the short time Fourier transform laid the

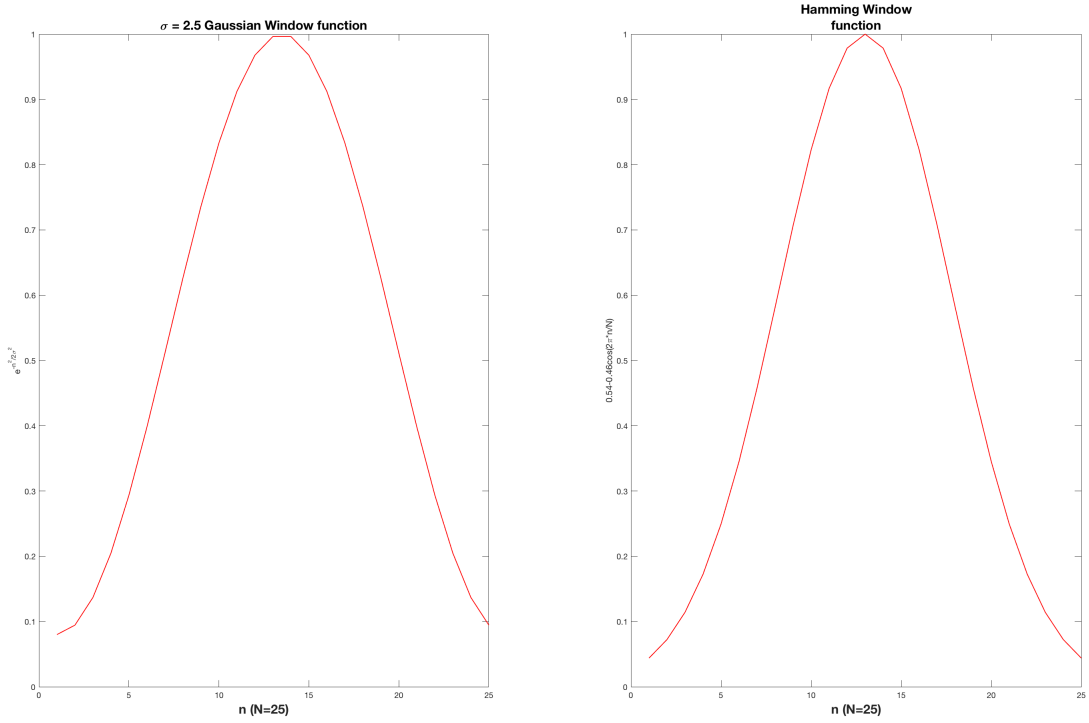

Figure 2.4: Window function (window length =25) for a) Gaussian b) Hamming. The graph was created using Matlab program ghamwin.m and attached in Appendix B

foundation for the joint time-frequency analysis.

2.3 Introduction

The Gabor transformation or expansion in signal synthesis uses the Gabor elementary function (GEF) as the base function (equivalent of simple harmonic function in the Fourier transform) and the idea was influenced by Heisenberg's uncertainty principle.

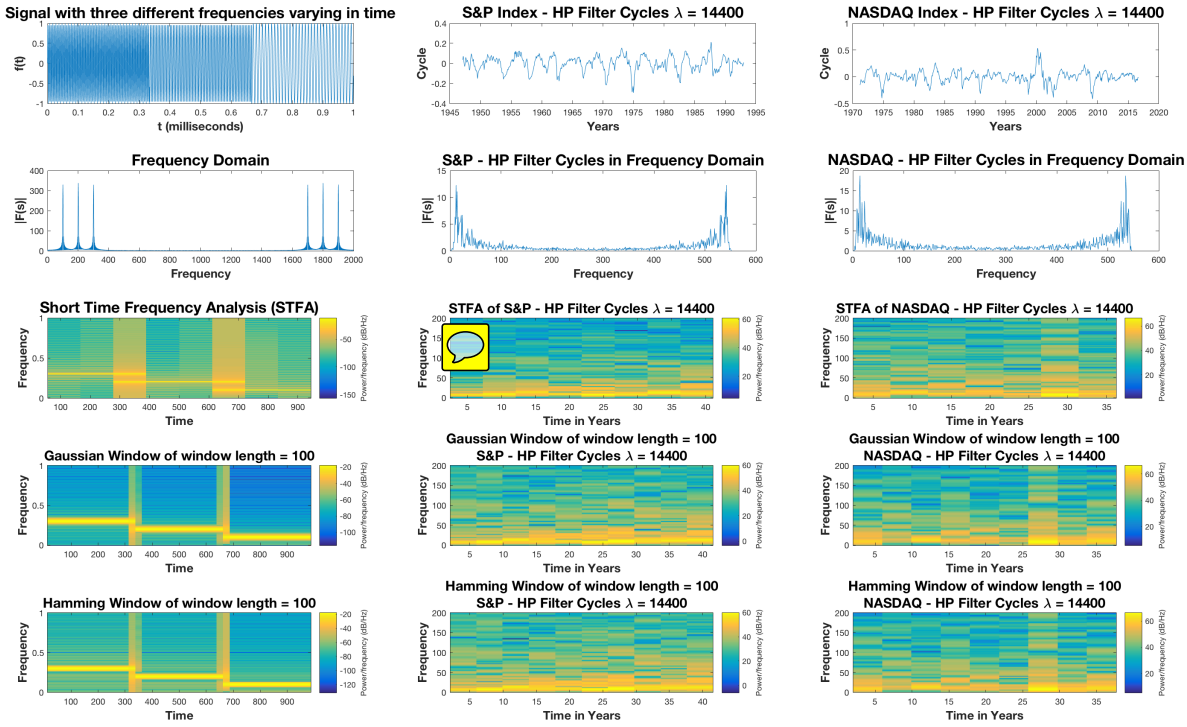

Figure 2.5: In the first column, The signal $f(t)$ as defined in 2.25 b) Fourier Transform of $f(t)$ c) Short Time Fourier Transfo $f(t)$ d) STFT using Gaussian window e) STFT using Hamming window. Second column is for SP500 and third column is for NASDAQ indexes. The graph was created using Matlab program spectrumAnalysis3.m and attached in Appendix B

2.3.1 Heisenberg's uncertainty principle

In quantum mechanics, simultaneously, both the position of a particle and the momentum of the particle can not be measured preciously. Let x be position and p be momentum of the particle. The standard deviation of x and p are denoted by Δx and Δp respectively. The uncertainty principle states that product of the standard deviations of position and momentum is greater than or equal to $\frac{\hbar}{2}$

$$\Delta x \Delta p \geq \frac{\hbar}{2}$$

$$\Delta x = \sqrt{\langle (x - \langle x \rangle)^2 \rangle}$$

$$\Delta p = \sqrt{\langle (p - \langle p \rangle)^2 \rangle}$$

2.3.2 Gabor Transformation

Gabor had the insight that the base function with minimum uncertainty in both time and frequency domains best captures temporal information during the frequency analysis.

Gabor in his original study proposed an elementary function in complex form which occupies minimum uncertainty and it is a product of harmonic oscillator of any frequency and a Gaussian function. The area occupied by the following elementary function in the joint time frequency domain is equal to the minimum uncertainty.

$$\psi(t) = \underbrace{e^{-\alpha^2(t)^2}}_{v} \overbrace{e^{j2\pi f_0 t + \phi}}^{w} \tag{2.26}$$

An arbitrary function $f(t)$ can be represented by a series of elementary functions which are constructed by translation in both the time and frequency domains. The function $f(t)$ is synthesized by the combination of GEFs.

$$f(t) = \sum_{m,n \in Z} C_{m,n} \psi_{m,n}(t) \tag{2.27}$$

where the $C_{m,n}$ are Gabor coefficients and the $\psi_{m,n}(t)$ are the bases functions called Gabor elementary functions. A GEF translated by 'na' is denoted by $\psi_{m,n}$ and given by

$$\psi_n(t) = \psi(t - na)$$

The translation in the frequency domain is also called modulation. A GEF is modu-

lated by using simple harmonic functions.

$$\psi_{m,n}(t) = \psi(t - na)e^{j2\pi mbt}$$

The original Gabor paper suggested that ψ is Gaussian. GEFs were studied by using other functions for ψ like a rectangle. a and b are time and frequency shift parameters and a,b > 0. $\psi_{m,n}$ is obtained by shifting it by lattice point $\times$ mb in the time-frequency plane.

ΔM, ΔN are time and frequency sampling intervals respectively. M, N are the number of sampling points in time and frequency domains.

The GEFs could be a set of Gaussian functions modulated by simple harmonics. GEF generated in 1D by combining Gaussian and simple harmonic functions. The Gaussian function remained the same for both GEF and only the sinusoidal functions changed between the two GEFs. Note the change in the frequency between the GEFs. A set of GEFs can be created by varying na and mb.

$$\psi_{m,n}(t) = e^{-\frac{(t-\mu-na)^2}{2\sigma^2}} e^{j2\pi mbt}$$

If the GEF ψ and its Fourier transform (Frequency representation) Ψ are localized at the origin then $\psi_{m,n}$ is localized at (na,mb) in the joint time-frequency domain. Each GEF occupies a region in time-frequency plane and associated $C_{m,n}$ represents a quantum of information.

Let $\psi(t)$ be a GEF. The GEF in the frequency domain is given by the Fourier transform

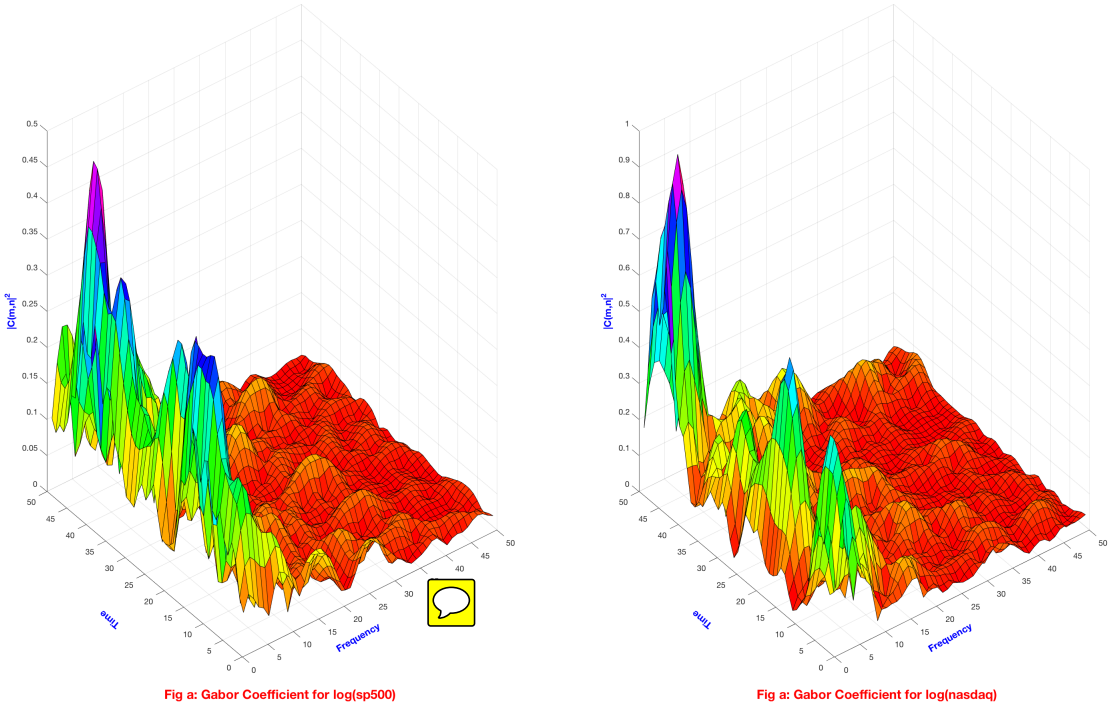

Figure 2.6: Fig a is the time section of Gabor distribution of HP Filter cycles log(SP500) and Fig b is the time section of Gabor distribution of for HP Filter cycles log(NASDAQ). $\Delta M = 8; \Delta N = 4; m = 50; n = 100; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Threshold is calculated as $max(|c(m,n)| - min(|c(m,n)|))/2$. The program used to create the graph is mygaborfilt.m and it is attached in the appendix.

$$\Psi(f) = \int_{-\infty}^{\infty} \psi(t) e^{-j2\pi ft}\, dt$$

These functions have a special property that relates to Heisenberg's uncertainty principle. The product of the standard deviation in time Δt and variance in frequency Δf is always greater than or equal to a certain quantity.

$$\Delta f \Delta t \geq \frac{1}{4\pi} \tag{2.28}$$

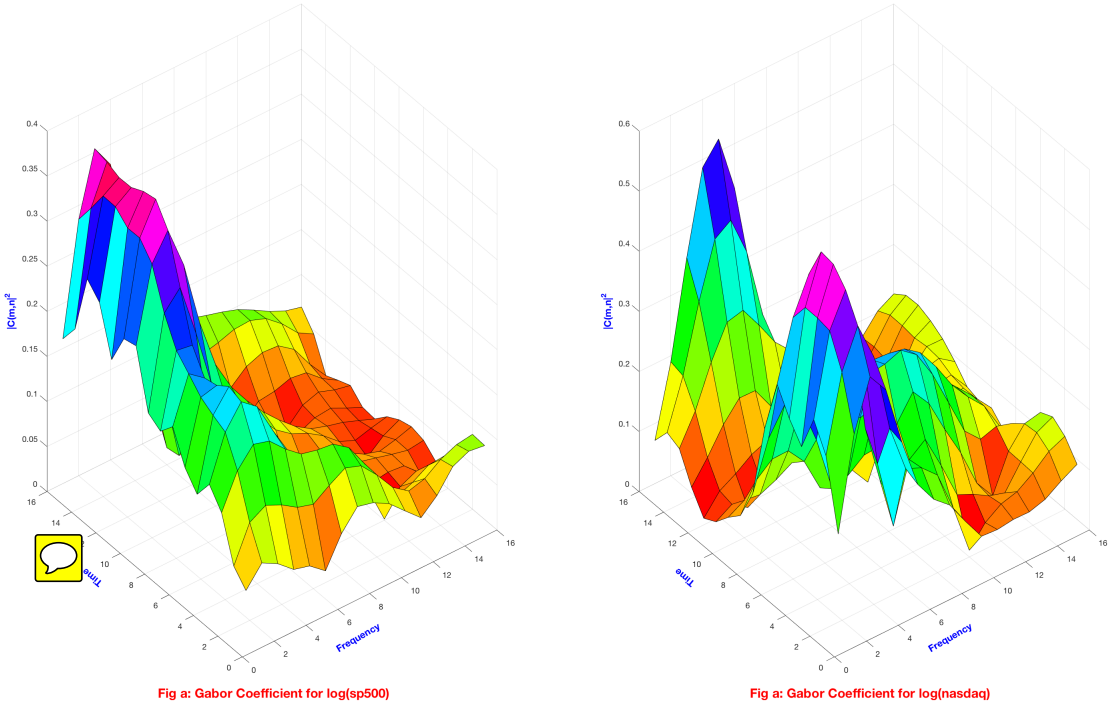

Figure 2.7: Fig a is the time section of Gabor distribution of HP Filter cycles log(SP500) and Fig b is the time section of Gabor distribution of for HP Filter cycles log(NASDAQ). $\Delta M = 8; \Delta N = 4; m = 16; n = 32; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Threshold is calculated as $max(|c(m,n)| - min(|c(m,n)|))/2$. The program used to create the graph is mygabor.m and it is attached in the appendix.

The time standard deviation or effective duration and frequency variance or effective frequency width can be calculated by the root mean square (RMS) deviation of the signal from the mean. The effective duration (Δt) and effective frequency width (Δf) are given by

$$\Delta t = \sqrt{\frac{\int_{-\infty}^{\infty} \psi(t)(\mu_t - t)^2 \psi^*(t) dt}{\int_{-\infty}^{\infty} \psi(t)\psi^*(t) dt}}; \Delta f = \sqrt{\frac{\int_{-\infty}^{\infty} \Psi(f)(\mu_f - f)^2 \Psi^*(f) df}{\int_{-\infty}^{\infty} \Psi(f)\Psi^*(f) df}}$$

where μ_t and μ_f are mean time and mean frequency and it is given by

$$\mu_t = \frac{\int_{-\infty}^{\infty} \psi(t) t \psi^*(t)}{\int_{-\infty}^{\infty} \psi(t)\psi^*(t)}; \mu_f = \frac{\int_{-\infty}^{\infty} \Psi(f) f \Psi^*(f)}{\int_{-\infty}^{\infty} \Psi(f)\Psi^*(f)}$$

2.3.3 Mask Operator

In general, random noise tends to spread evenly in the joint time-frequency domain, while the signal itself concentrates in a relatively small range. Consequently, by joint time-frequency representation, the signal-to-noise ratio could be substantially improved. Once the interesting signal has been identified, we can apply mask operator to eliminate the background noise. Typical procedure of the time-variant filter is first to take the Gabor transform and then apply the mask operator to eliminate the background noise to obtain noiseless Gabor coefficients.

$$\Delta t \Delta f = \sqrt{\frac{\int_{-\infty}^{\infty} \psi(t)(\mu_t - t)^2 \psi^*(t) dt}{\int_{-\infty}^{\infty} \psi(t)\psi^*(t) dt} \frac{\int_{-\infty}^{\infty} \Psi(f)(\mu_f - f)^2 \Psi^*(f) df}{\int_{-\infty}^{\infty} \Psi(f)\Psi^*(f) df}} \geq \frac{1}{4\pi}$$

There are three possibilities for above equation is given below in table 2.1 .

$\Delta t\ \Delta f$	**S**ampling	Remarks
$= \frac{1}{4\pi}$	Critical	Special functions called GEF
$> \frac{1}{4\pi}$	Over	–
$< \frac{1}{4\pi}$	Under	This case is not possible

Table 2.1: Possible values for Δt Δf and special case for Gabor function

Analogous to 1D uncertainty principle, there are two 2D uncertainty principles constraining the effective width (Δx) and the effective length (Δy) of a signal $f(x, y)$ and the effective width (Δu) and the effective length (Δv) of its 2D Fourier transform $F(u, v)$. 2D Gabor transformation has wide application in image processing domain and the 2D analysis is out of scope for this project.

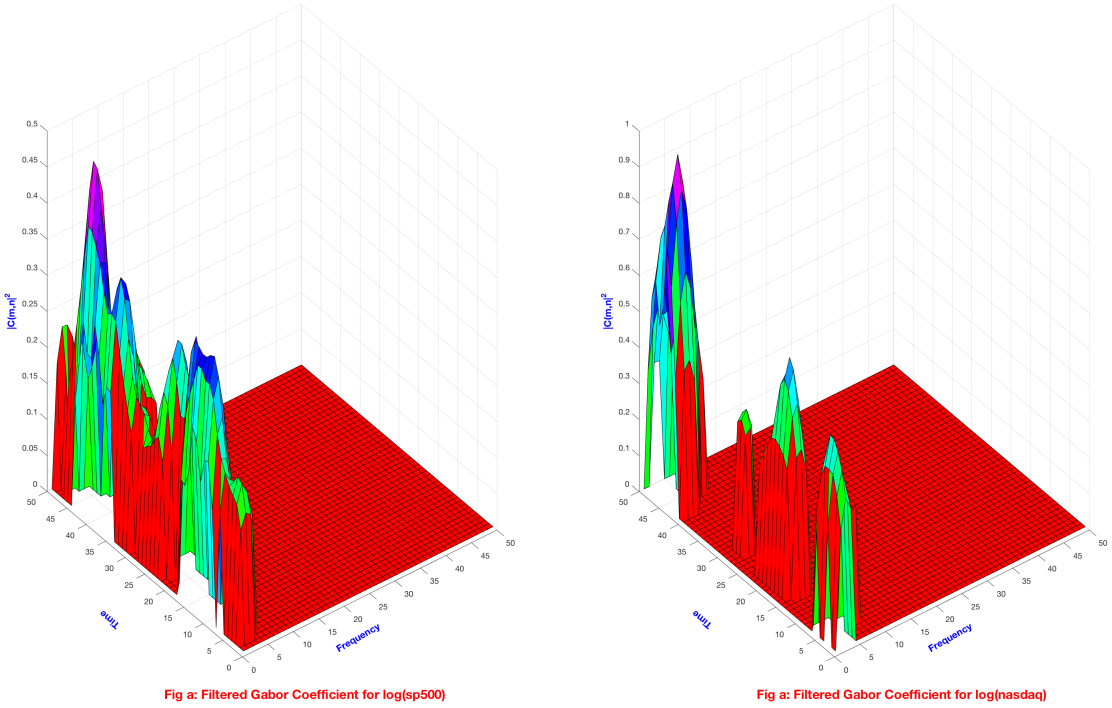

Figure 2.8: Fig a is the Filtered Gabor Coefficient for HP Filter cycles log(SP500) and Fig b is the Filtered Gabor Coefficient for HP Filter cycles log(NASDAQ).$\Delta M = 8; \Delta N = 4; m = 50; n = 100; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. The mask operator is created based on the threshold of a peak distribution. The program used to create the graph is myfiltgabor.m and it is attached in the appendix.

$$\Delta x \Delta u = \sqrt{\frac{\int_{-\infty}^{\infty} \psi(x,y)(\mu_x - x)^2 \psi^*(x,y) dxdy}{\int_{-\infty}^{\infty} \psi(x,y)\psi^*(x,y)dxdy} \frac{\int_{-\infty}^{\infty} \Psi(u,v)(\mu_u - u)^2 \Psi^*(u,v) dudv}{\int_{-\infty}^{\infty} \Psi(u,v)\Psi^*(u,v)dudv}} \geq \frac{1}{4\pi}$$

$$\Delta y \Delta v = \sqrt{\frac{\int_{-\infty}^{\infty} \psi(x,y)(\mu_y - y)^2 \psi^*(x,y) dxdy}{\int_{-\infty}^{\infty} \psi(x,y)\psi^*(x,y)dxdy} \frac{\int_{-\infty}^{\infty} \Psi(u,v)(\mu_v - v)^2 \Psi^*(u,v) dudv}{\int_{-\infty}^{\infty} \Psi(u,v)\Psi^*(u,v)dudv}} \geq \frac{1}{4\pi}$$

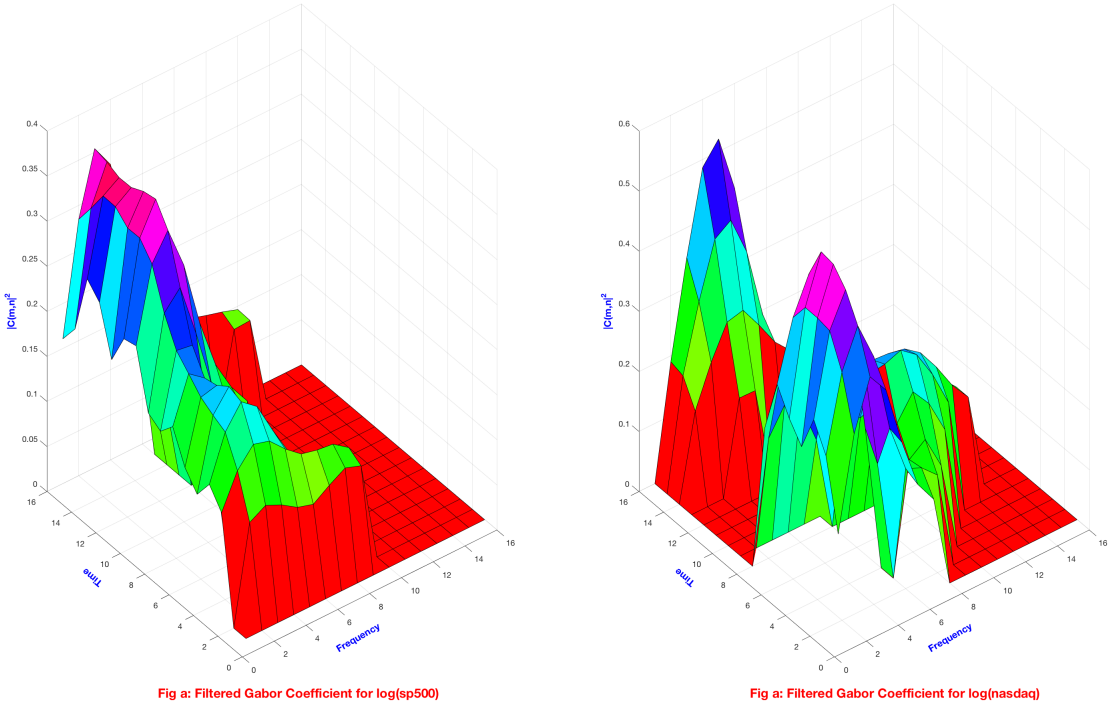

Figure 2.9: Fig a is the Filtered Gabor Coefficient for HP Filter cycles log(SP500) and Fig b is the Filtered Gabor Coefficient for HP Filter cycles log(NASDAQ).$\Delta M = 8; \Delta N = 4; m = 16; n = 32 \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. The mask operator is created based on the threshold of a peak distribution. The program used to create the graph is myfiltgabor.m and it is attached in the appendix.

$$\Delta x \Delta u \Delta y \Delta v \geq \frac{1}{16\pi^2}$$

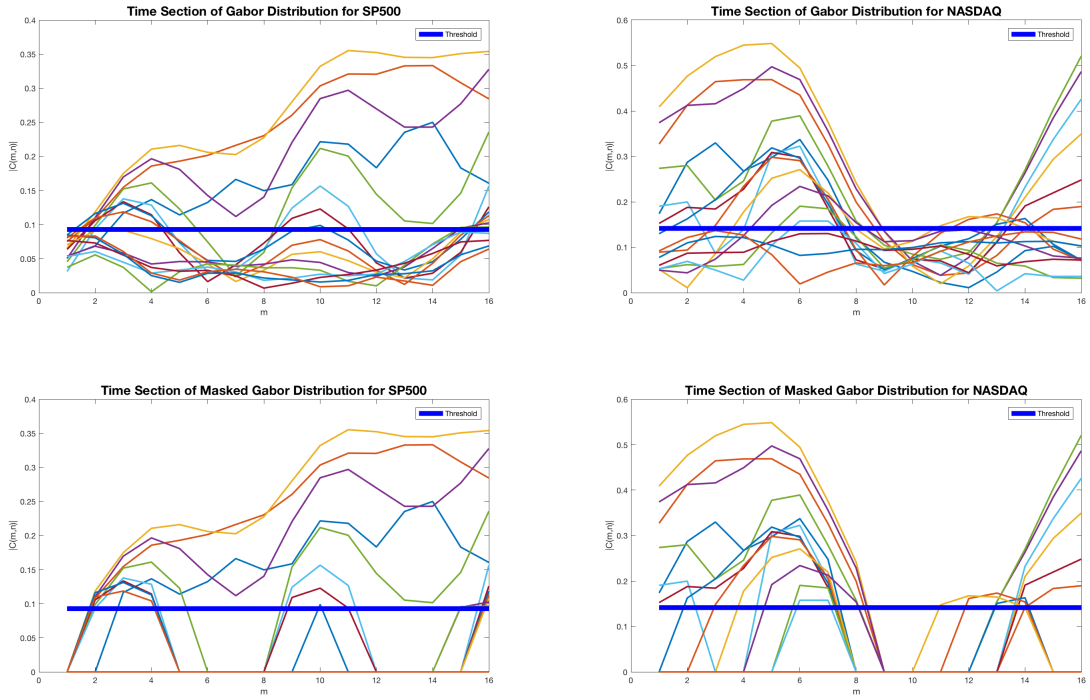

Figure 2.10: Fig a is the time section of Gabor distribution of HP Filter cycles log(SP500) and Fig b is the time section of Gabor distribution of for HP Filter cycles log(NASDAQ). $\Delta M = 8; \Delta N = 4; m = 16; n = 32; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Threshold is calculated as $max(|c(m,n)| - min(|c(m,n)|))/2$. Mask operator used to eliminate values below threshold. The program used to create the graph is mygaborfilt.m and it is attached in the appendix.

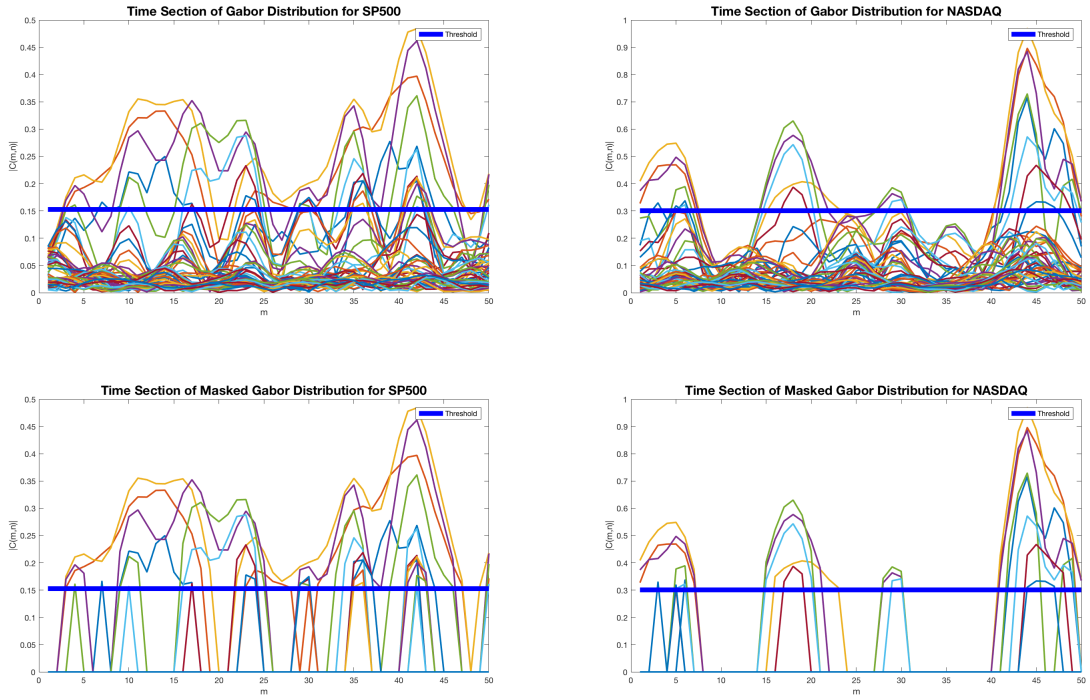

Figure 2.11: Fig a is the time section of Gabor distribution of HP Filter cycles log(SP500) and Fig b is the time section of Gabor distribution of for HP Filter cycles log(NASDAQ). $\Delta M = 8; \Delta N = 4; m = 50; n = 100; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Threshold is calculated as $max(|c(m,n)| - min(|c(m,n)|))/2$. Mask operator used to eliminate values below threshold. The program used to create the graph is mygaborfilt.m and it is attached in the appendix.

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. The WD belongs to a large class of bilinear distributions known as the Cohen's class, in which each member can be obtained by choosing a different kernel in the generalized bilinear distribution definition.

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

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

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

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

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 τ, of the product ff^*. It may also be obtained from

the Fourier transform F of f by

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

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

$$W_f(t, \omega) = W_F(\omega, t) \tag{3.4}$$

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 spurious "auto terms" due to its intrinsic bi-linearity. The WD of the sum of two signals $f(t) + f'(t)$ is given by

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

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

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

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

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

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

3.2 Wigner-Ville Distribution

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

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

where $z(t)$ is the analytical 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

$$F(\omega) = 0 \tag{3.10}$$

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

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

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

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

inversion property is the ability to extract the time domain signal from the distribution up to a constant phase factor. The distribution that satisfies this property is given as

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

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

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

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

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

leads to the recovering of the odd samples.

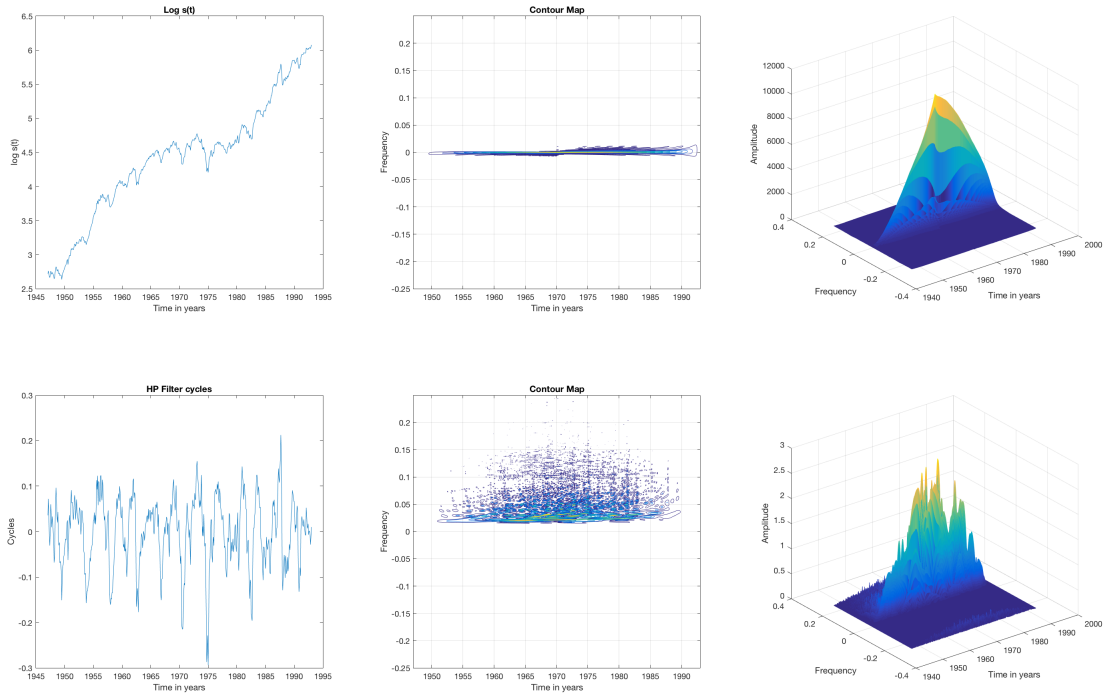

Figure 3.1: Wigner Distribution transform for log(s); where s denotes the sp500. Top row is the log(s) and its Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of log(s) with $\lambda = 14400$ and its Wigner Distribution transform. The graph was created using Matlab code mywvd.m and it is attached in the Appendix B

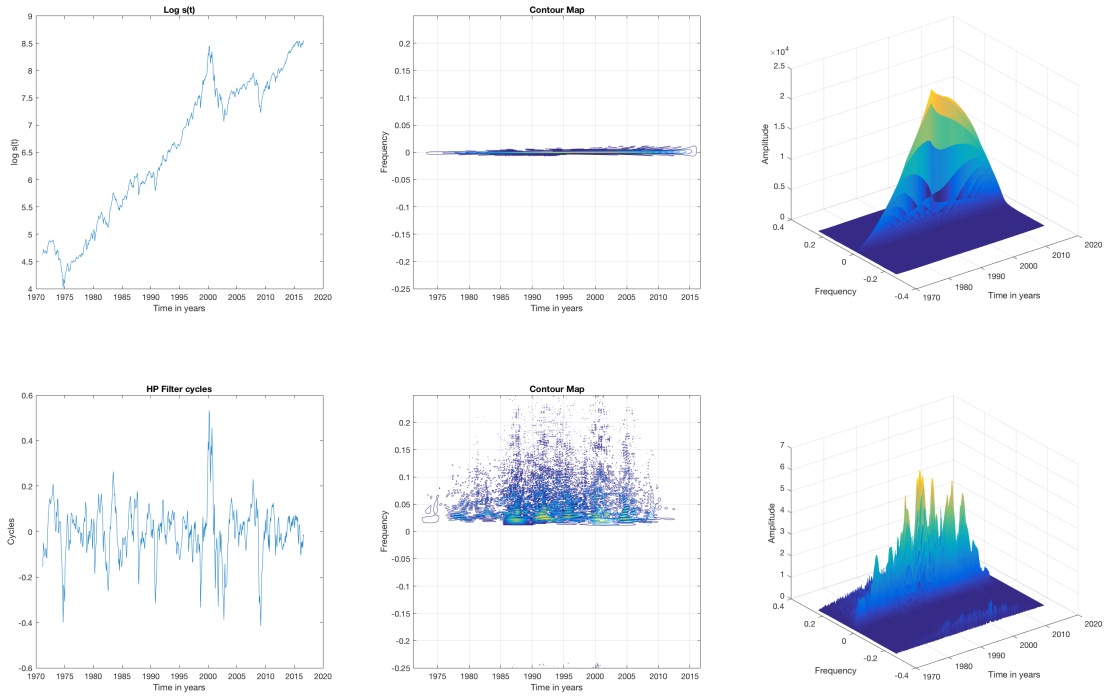

Figure 3.2: Wigner Distribution transform for log(s); where s denotes the NASDAQ index. Top row is the log(s), Wigner Distribution transform presentation in the contour and three dimensional mesh. Bottom row is the HP filter cycle of log(s) with $\lambda = 14400$ and its Wigner Distribution transform. 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.

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

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

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

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

The WVD is time and frequency-shift invariant. Let

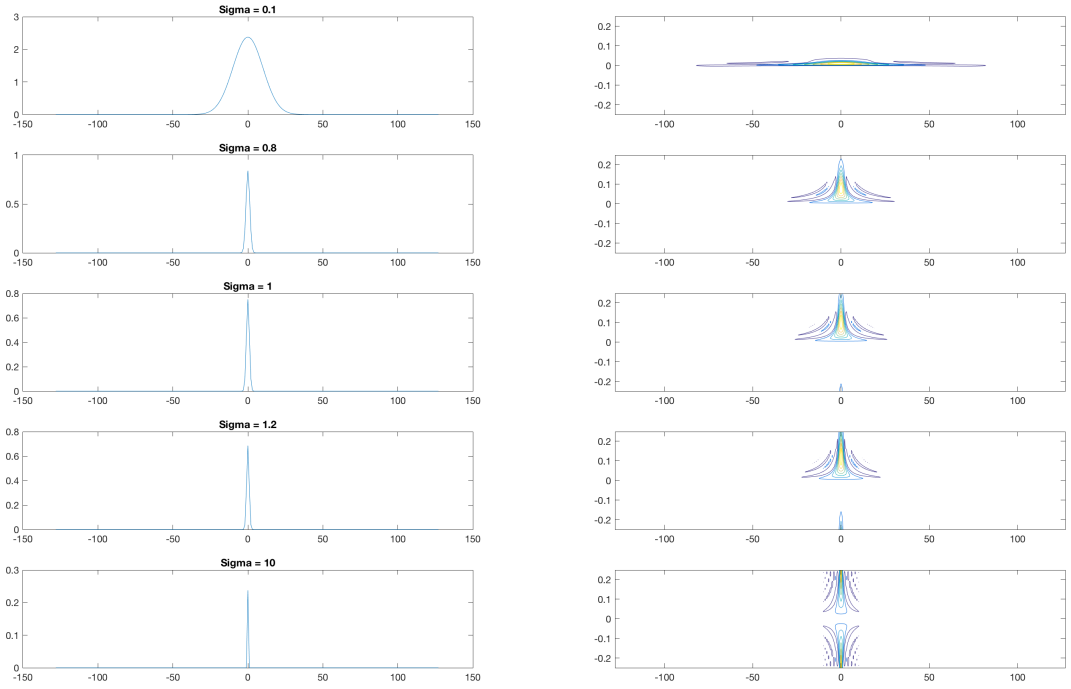

Figure 4.1: Fig in the left hand side represents the Gaussian function $g(t)$ as defined above for various σ values. Fig in the right land side represents the Wigner Ville Distribution WVD_g as defined above. The graph is created using the mywvdgauss.m program attached in the Appendix. WVD values are created by using the HOSA (Higher Order Spectral Analysis) Matlab toolbox.

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

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

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

$$WVD_h(t, \omega) = WVD_g(t - mT, \omega - n\Omega) \tag{4.5}$$

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

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

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

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

where

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

$$\begin{aligned} t_\mu &= \frac{m + m'}{2} T \\ \omega_\mu &= \frac{n + n'}{2} \Omega \\ t_d &= (m - m')T \\ \omega_d &= (n - n')\Omega \end{aligned} \tag{4.9}$$

The $WVD_{h,h'}(t,\omega)$ has the same envelope as the $WVD_g(t,\omega)$ but is oscillating with frequency ω_d in the time domain and t_d in the frequency domain. The location of $WVD_{h,h'}(t - t_\mu, \omega - \omega_\mu)$ 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.

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

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

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

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

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

where

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

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

where t_d and ω_d reflect the degree of oscillation.

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

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

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

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

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

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

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 $|m-m'|+|n-n'|=d$

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

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

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

Substituting the value of t_d, ω_d, t_μ and ω_μ, we get

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

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.

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

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

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

where

$$P_d[i,k] = \sum_{|m-m'|+|n-n'|=d} C_{m,n}C_{m'n'}WVD_{h,h'}[i,k] \tag{4.23}$$

The $TFDS_D[i,k]$ 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.

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

where Δt denotes the sampling interval. For the Gaussian functions, WVD is obtained by sampling the formula.

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

We assume $\Delta t = 1$. $WVD_{h,h'}[i,k]$ in 4.25 is completely determined by the parameters of the Gabor expansion, such as $\Delta M, \Delta N, L$ and σ which are independent of the analyzed signal. Therefore, once $\Delta M, \Delta N, L$ and σ are determined, $WVD_{h,h'}[i,k]$ can be precomputed and saved in a table.

CHAPTER V

Color Chaos Model

5.1 Color Chaos

Chen claimed in [2], that correlation analysis and spectral analysis are complementary tools in the stationary time-series analysis. Based on the detail study of the paper [2],**Color Chaos** is defined as a time frequency representation as a non-parametric approach for generalized spectral analysis for evolutionary time series. In color chaos, the HP filter is applied for trend-cycle decomposition and time-variant filters in Gabor space for pattern recognition. Discrete-time white chaos generalized by nonlinear difference equations is tractable analytically and from the needs of empirical analysis, the continuous-time color chaos generated by nonlinear differential equations is more capable of describing business cycles than white chaos, since fluctuations and recurrent patterns can be characterized by nonlinear oscillations with irregular amplitude and a narrow frequency band in the spectrum. Chen claimed in [2], the newly decoded deterministic signals from persistent business cycles reveal new sources of market uncertainty, such as changing growth-trend and shifting business cycles.

5.1.1 Role of time scale & reference trends in representation of business cycles

A distinctive problem in economic analysis is how to deal with growing trends in an aggregate economic time series. Both level and rate information are important when correlations are not short during business cycles.

Time Scale: Chaos theory in nonlinear dynamics emphasizes the role of history, because a nonlinear deterministic system is sensitive to its initial condition. The martingale theory of the stock market ignores the path-dependent information in the stock market. The challenges faced are:

1. Choosing an appropriate time sampling rate is often ignored in econometric analysis. Chaotic cycles in continuous time may look like noise if the sampling time internal is not small compared to its fundamental period of cycle. For example, annual economic data are not capable of revealing the frequency pattern of business cycle.

2. Numerically, a large time unit such as the annual time series can easily obscure a cyclic pattern in the correlation analysis of business cycles.

Reference Trend: How to choose a reference trend or a proper transformation to simplify the empirical pattern of business cycles? The core problem in economic analysis is not noise-smoothing but trend-defining in economic observation and decision making. There are two criteria in choosing the proper mathematical representation.

1. Mathematical reliability

2. Empirical verifiability.

Unlike experimental economics, macroeconomic time series are not reproducible in history. Traditional tests in econometric analysis have limited power in studies of an evolutionary economy containing deterministic components. A good fit of past data

does not guarantee the ability for better future predictions. There are two extreme approaches in econometric analysis: the trend-stationary (TS) approach of log-linear detrending (LLD) and the difference-stationary (DS) approach of first differencing (FD). A compromise between these two extremes is the Hodrick and Prescott (HP) filter. In principle, a choice of observation reference is associated with a theory of economic dynamics. Log-linear detrending implies a constant exponential growth as shown in the figure 1.4.

The FD detrending produces a noisy picture that is predicted by the random-walk model with a constant drift (or the so-called unit-root model in econometric literature). Economically speaking, the FD detrending in econometrics implies that the level information in price indicators can be ignored in economic behavior. This assertion may conflict with many economic practices, since traders constantly watch economic trends, and no one will make an investment decision based only on the current rate of price changes. The error-correction model in econometrics tried to remedy the problem by addition some lagged-level information, such as using a one-year-before level as an approximation of the long-run equilibrium. Then comes the problem of what is the long run equilibrium in the empirical sense. A proper decomposition of trend and cycles may find an appropriate scheme to weigh the short-term and long-run impacts of economic movements in economic dynamics. The essence of trend-cycle decomposition is finding an appropriate time window, or equivalently, a proper frequency window, for observing time-dependent movements. Log-linear detrending is a low-pass filter or wave detector, while first differencing is a high-pass filter or noise amplifier.

The main drawback of LLD detrending is its over-dependence on historical boundaries, while the DS series is to erratic from amplifying high-frequency noise.

HP filter has two advantages.

1. It is localized approach in detrending, with the problem of boundary depen-

dence.

2. Frequency response is in the range of business cycles.

5.1.2 Dimensionality

5.1.3 Correlation Dimension

The correlation dimension provides a tool to quantify self-similarity. A larger correlation dimension corresponds to a larger degree of complexity and less self-similarity. The most frequently used procedure to estimate the correlation dimension was introduced by Grassberger and Procaccia (1983). They defined the correlation sum for a collection of points X_i $(i = 1, 2, 3, ..N)$ in some phase space to be the fraction of all possible pairs of points which are closer than a given distance ϵ in a particular norm:

To compute the correlation integral $C(m, \epsilon)$ and the correlation dimension D_c, first delay-coordinate embed the signal as follow as:

$X(t) = x(t), x(t + \tau), x(t + 2\tau), .., x(t + (m - 1)\tau)$

where τ is the delay time, and m is the embedding dimension.

$$C(m, \epsilon) = \frac{2}{(N - m)(N - m - 1)} \sum_{i=m}^{N} \sum_{j=i+1}^{N} \Theta(\epsilon - \|X_i - X_j\|) \tag{5.1}$$

Where Θ is Heaviside step function, $\Theta(x) = 0$ if $x \leq 0$ and $\Theta(x) = 1$ for $x > 0$. Thus equation 5.1 counts the pairs (X_i, X_j) whose distance is smaller than ϵ. When $N \to \infty$, for small values of ϵ, C follows a power law;

$$C(\epsilon) \propto \epsilon^{D_c} \tag{5.2}$$

Where D_c is correlation dimension. Therefore, D_c is defined as

$$D_c = \lim_{\epsilon \to 0} \lim_{N \to \infty} \frac{\partial log_{10} C(\epsilon)}{\partial log_{10}(\epsilon)} \tag{5.3}$$

Correlation dimension is estimated by computing the slope of the straight line by using least square fit in a plot of $log_{10}C(\epsilon)$ vs $log_{10}(\epsilon)$.

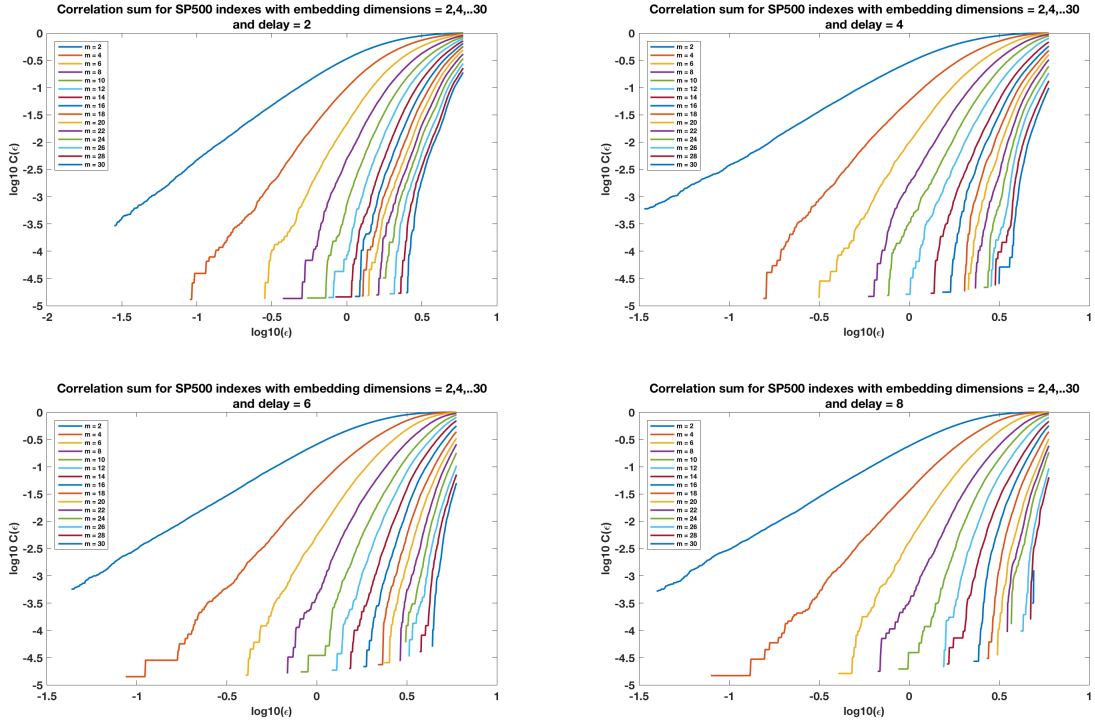

Figure 5.1: Correlation Dimension for NASDAQ. The graph was created by using drawCorrDim.m and it is attached in the Appendix

5.1.4 Lyapunov exponent

The Lyapunov exponent is a quantity that describes the rate of separation of two neighbouring trajectories. The divergence of two trajectories at time t with initial separation δZ_0 at $t = 0$, is expressed as:

$$|\delta Z(t)| \approx e^{\lambda t}|\delta Z_0| \tag{5.4}$$

where λ is the Lyapunov exponent. Different orientations of initial separation vectors would result in different rate of separation and thus, a whole spectrum of the

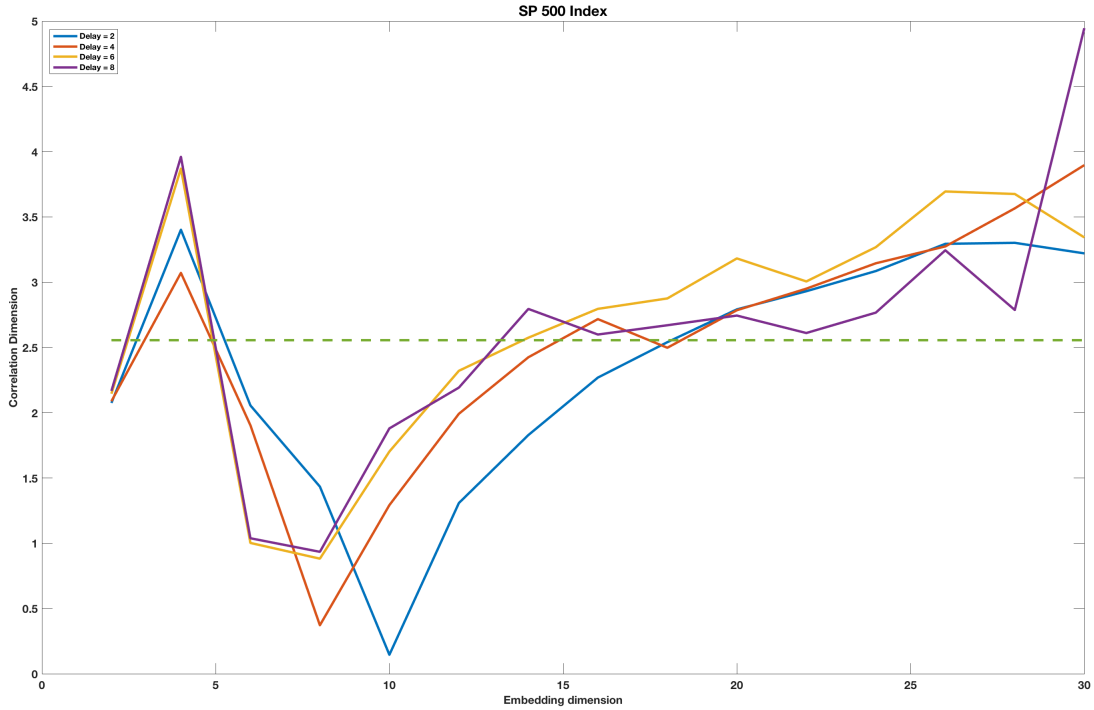

Figure 5.2: Correlation Dimension for NASDAQ. The graph was created by using drawCorrDim.m and it is attached in the Appendix

Lyapunov exponents. The Largest λ is called the Maximum Lyapunov Exponent:

$$D_c = \lim_{\epsilon \to 0} \lim_{N \to \infty} \frac{\partial log_{10} C(\epsilon)}{\partial log_{10}(\epsilon)} \tag{5.5}$$

$\lambda gt0$ is usually considered as indicator of chaos, $\lambda lt0$ is usually considered an indicator of a mean reverting behavior and $\lambda = 0$ is a characteristic of cyclic behavior.

5.1.5 Instantaneous Autocorrelations and instantaneous frequency in Time-frequency representation

The concepts of instantaneous autocorrelation and instantaneous frequency are important in developing generalized spectral analysis. A symmetric window in a localized time interval is introduced in the instantaneous autocorrelation function of the

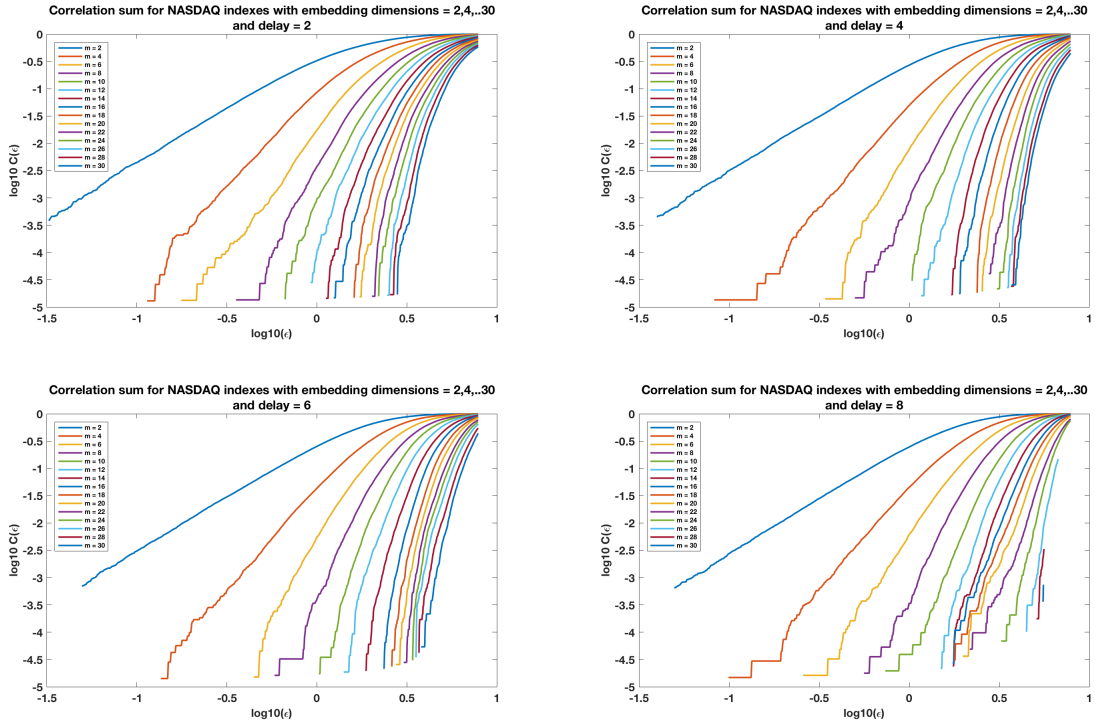

Figure 5.3: Correlation Dimension for NASDAQ. The graph was created by using drawCorrDim.m and it is attached in the Appendix

bilinear Wigner distribution (WD); the corresponding time-dependent frequency (or simply time frequency) can be defined by the Fourier spectrum of its autocorrelations.

In the figure 5.7 the chaos of the SP500 is shown and in the figure 5.5 the chaos of the SP500 is shown.

Data	η	$\nu(\%)$	CCgo	P_c	$\phi(\%)$	P_{dc}	λ^{-1}	μ
SP500	1.032	106.6	0.4141	NA	NA	NA	NA	
NASDAQ	1.0116	102.33	0.4735	NA	NA	NA	NA	

Table 5.1: Detrend Statistics on S&P 500

5.1.6 Conclusion

Color chaos model uncovers the business cycle in the stock market indices SP500 and NASDAQ and existence of persistent cycles reveals a new perspective of market.

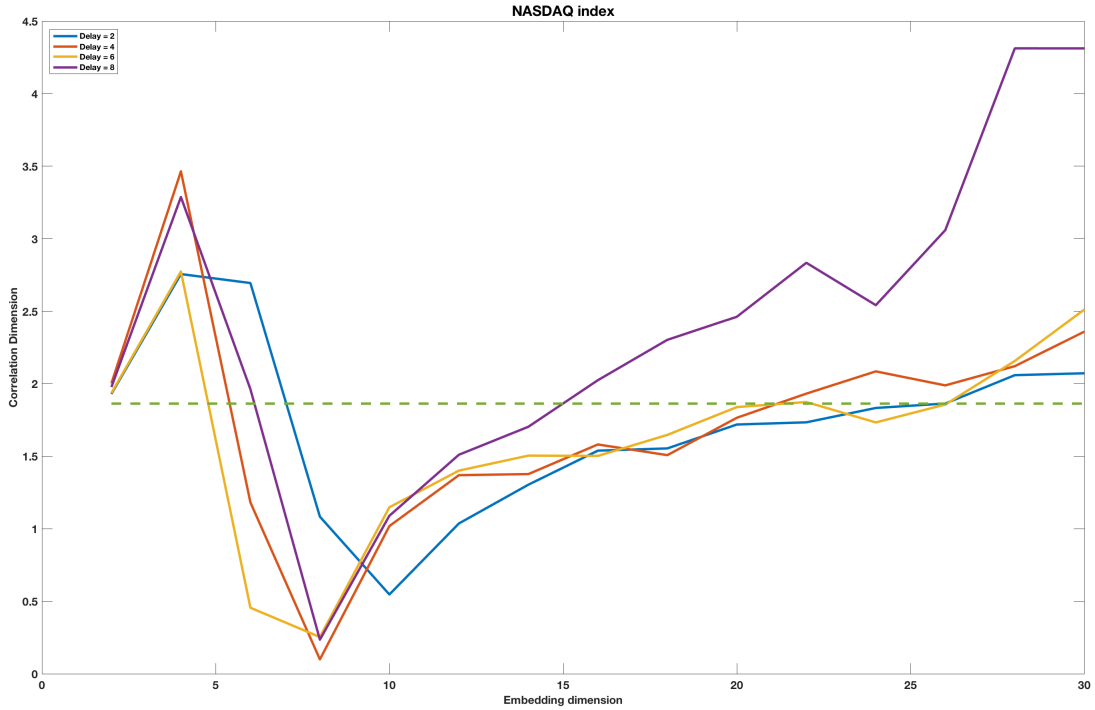

Figure 5.4: Correlation Dimension for NASDAQ. The graph was created by using drawCorrDim.m and it is attached in the Appendix

The study provides additional clarity on the opportunity in studying stock market movements using the joint time frequency analysis. A new study was performed by rearranging the indices value randomly instead of being time dependent, similar color chaos models was applied. The initial results of this new study revealed a similar cycles in the stock market. The results of this new study is not attached to this report but provides an opportunity to continue the study market behavior by converting the market indices value time independent. Further study may reveal more deeper understanding of the market behavior.

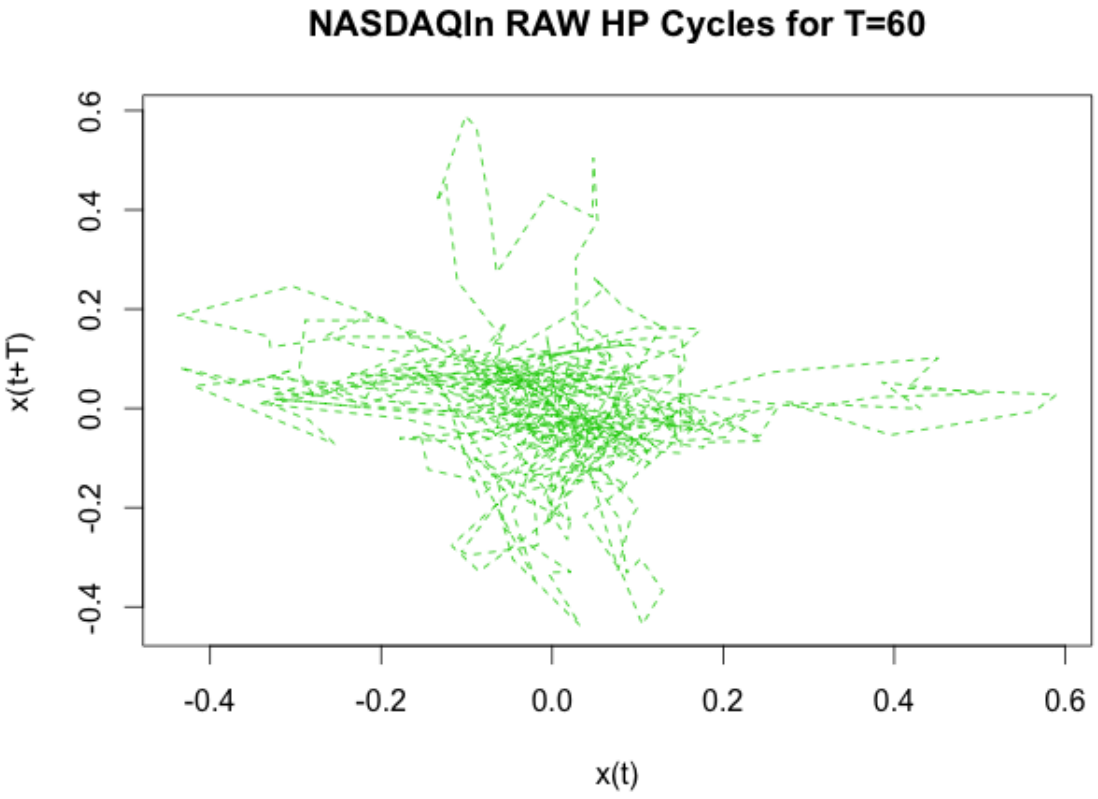

Figure 5.5: Phase Portrait for log(NASDAQ) unfiltered series $T = 60$. The graph was created by using Attractor.R (R program) and it is attached in the Appendix A

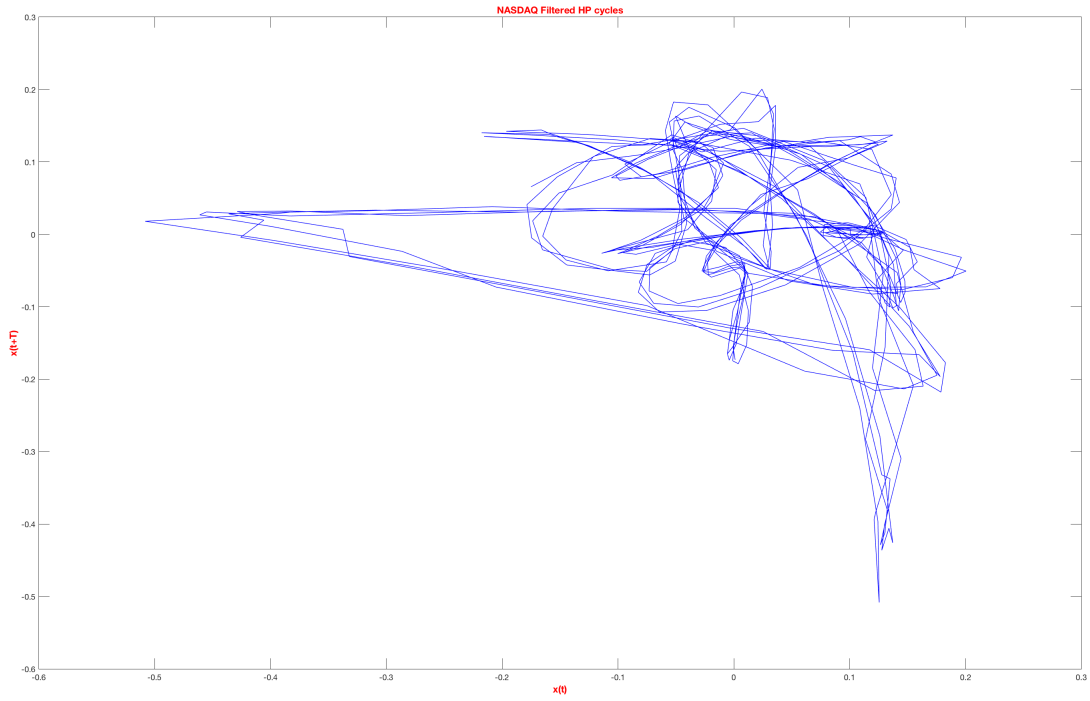

Figure 5.6: Phase Portrait for log(NASDAQ) unfiltered series $T = 60$. A pattern of strange attractor can be observed in the graph. The threshold value that depends on H ($H = 0.5$) has less significant impact to the pattern emergence whereas the size of m, n in $C(m, n)$, the Gabor Coefficient has significant impact to the pattern of strange attractor. The $16x16$ Gabor Coefficient was used. The graph was created by using myAttractor.m (Matlab program) and it is attached in the Appendix B.

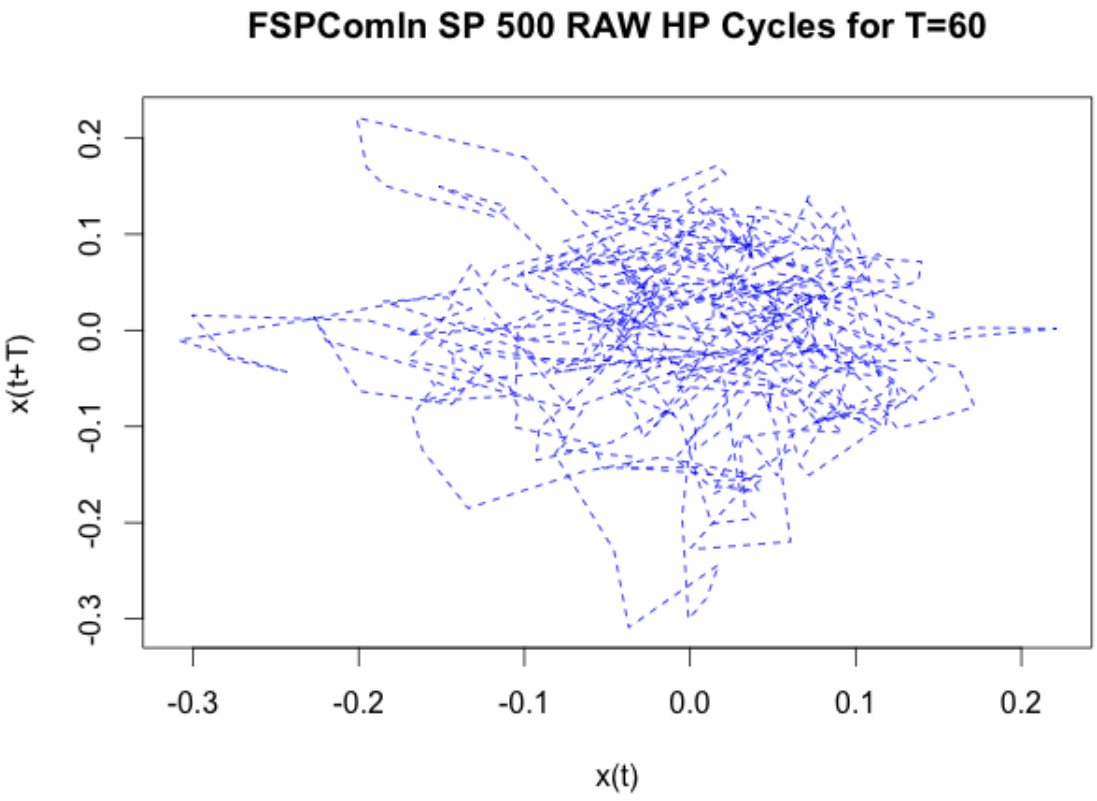

Figure 5.7: Phase Portrait for log(SP500) unfiltered series $T = 60$. The graph was created by using Attractor.R (R program) and it is attached in the Appendix A.

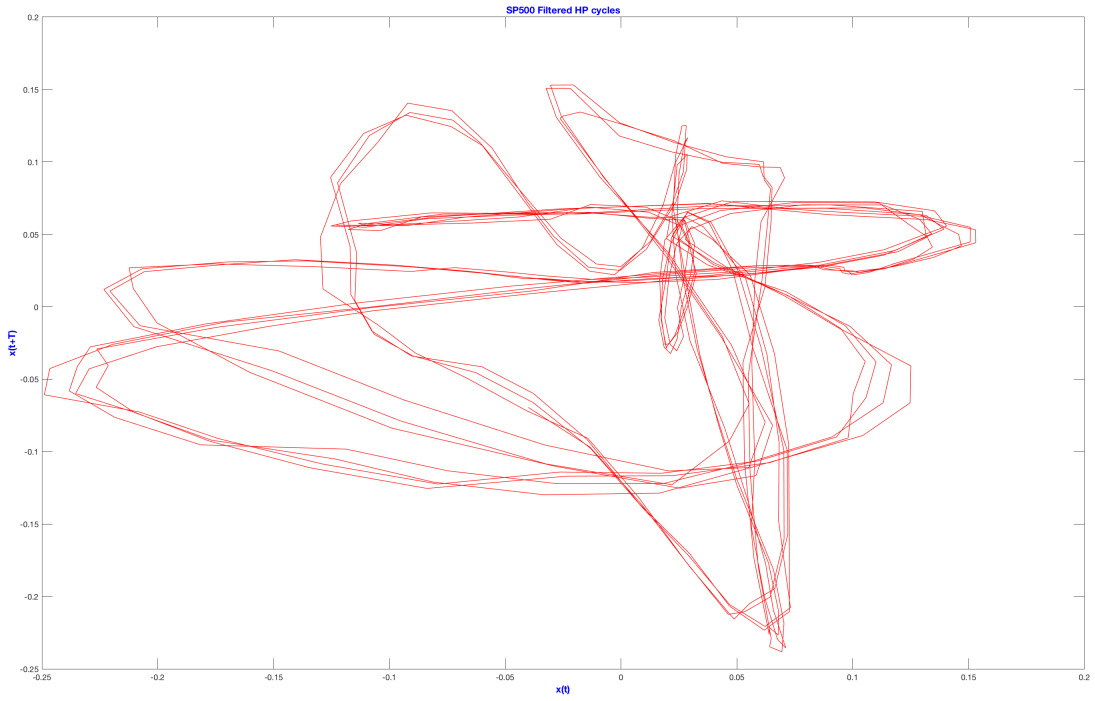

Figure 5.8: Phase Portrait for log(SP500) unfiltered series $T = 60$. A pattern of strange attractor can be observed in the graph. The threshold value that depends on H ($H = 0.5$) has less significant impact to the pattern emergence whereas the size of m, n in $C(m, n)$, the Gabor Coefficient has significant impact to the pattern of strange attractor. The $16x16$ Gabor Coefficient was used. The graph was created by using myAttractor.m (Matlab program) and it is attached in the Appendix B.

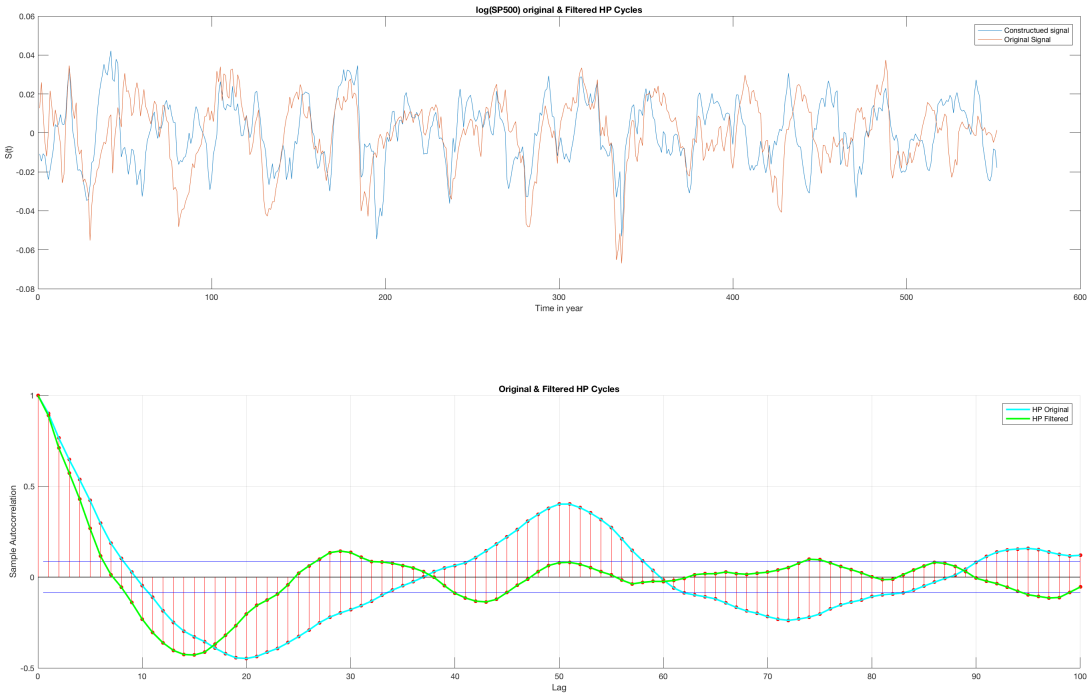

Figure 5.9: Fig a) The original and reconstructed time series of log(SP500) HP Cycles. Gabor Coefficients are created using the parameters - .$\Delta M = 8; \Delta N = 4; m = 16; n = 16; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Fig b) Autocorelations of the original and reconstructed time series of log(SP500) HP cycle. The program used to create the graph is myreconstfromgabor.m and it is attached in the appendix B.

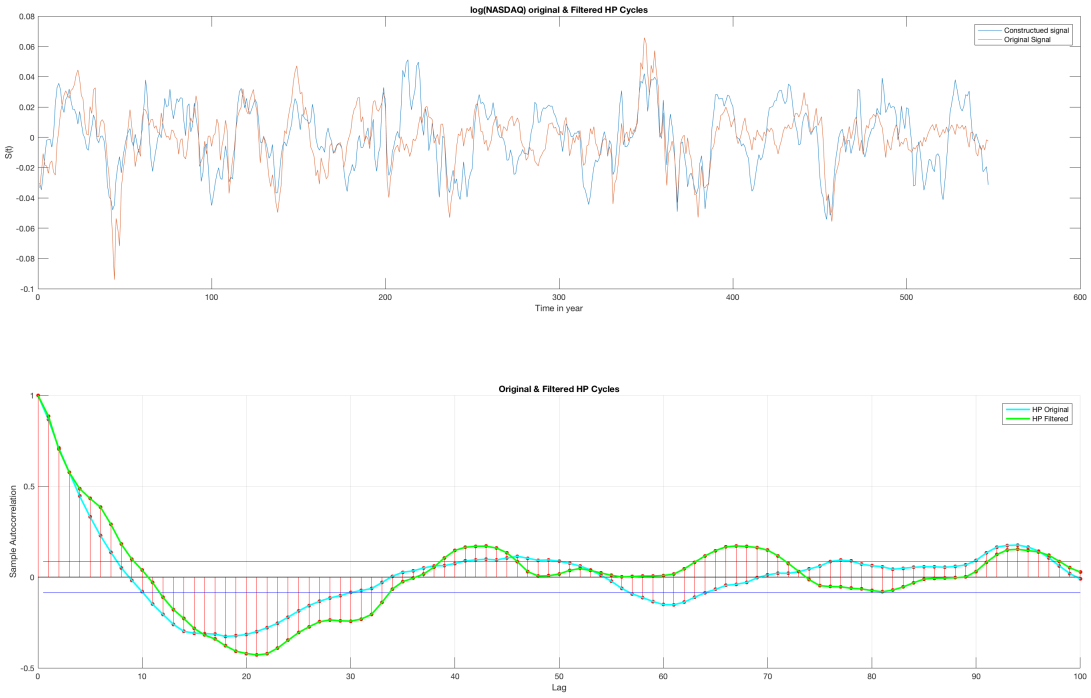

Figure 5.10: Fig a) The original and reconstructed time series of log(NASDAQ) HP Cycles. Gabor Coefficients are created using the parameters - .$\Delta M = 8; \Delta N = 4; m = 16; n = 16; \sigma = \sqrt{\frac{\Delta ML}{\Delta N 2\pi}}$. Fig b) Autocorelations of the original and reconstructed time series of log(NASDAQ) HP cycle. The program used to create the graph is myreconstfromgabor.m and it is attached in the appendix B.

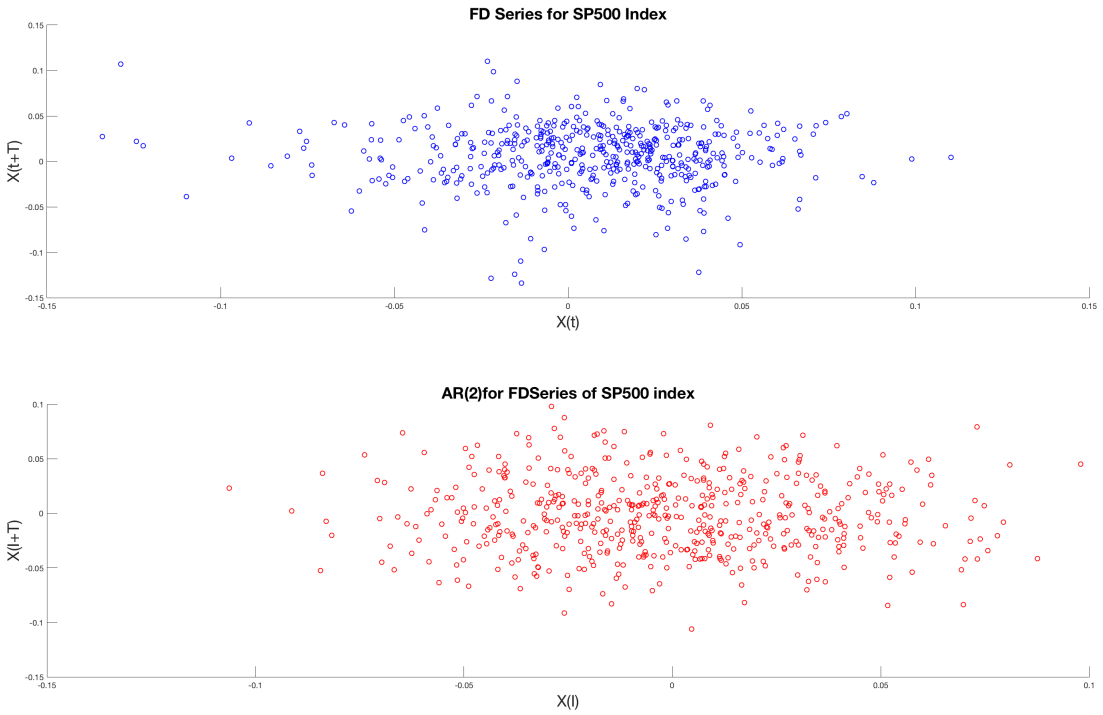

Figure 5.11: Fig a) SP500 FD Series. T = 40. The pattern demonstrates the existence of dominant of high frequency noise. Fig b) AR(2) for the FD Series of SP500. T=5. The program used to create the graph is myFDFilter.m and it is attached in the appendix B.

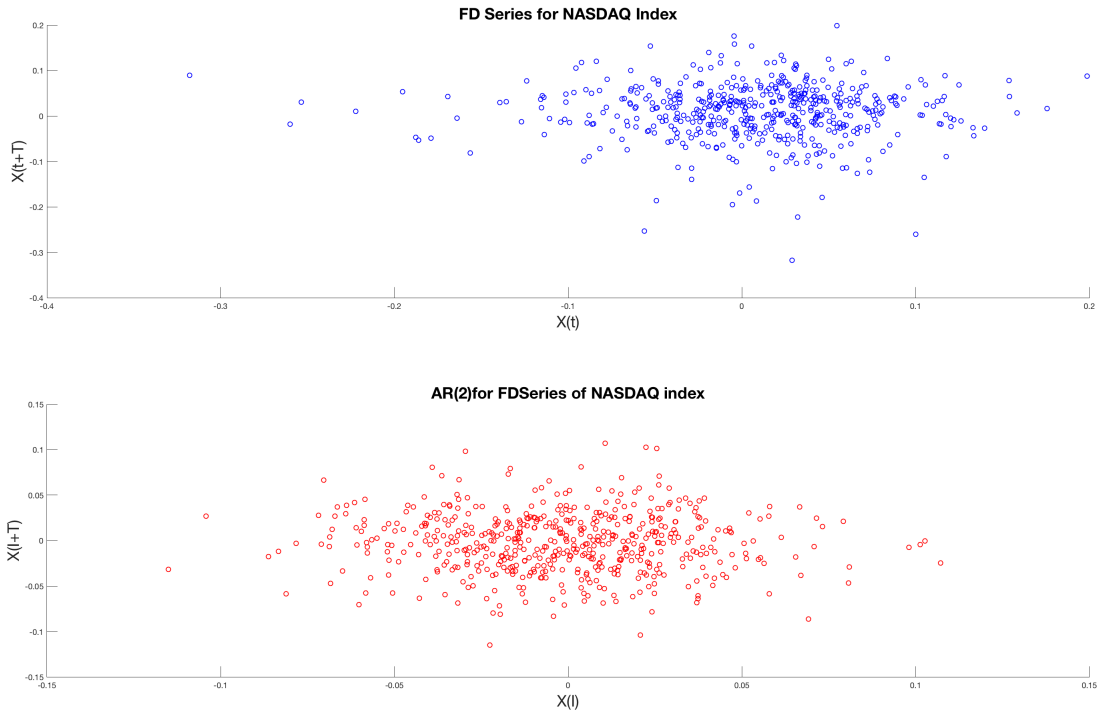

Figure 5.12: Fig a) NASDAQ FD Series. T = 40. The pattern demonstrates the existence of dominant of high frequency noise. Fig b) AR(2) for the FD Series of NASDAQ. T=5. The program used to create the graph is myFDFilter.m and it is attached in the appendix B.

APPENDICES

APPENDIX A

R code that are used to do analysis

```
fdplot <-function()
{
  #Program used to create the Difference stationary for NASDAQ & SP500 indexes
  # Praba Siva
  # praba@umich.edu
  # @prabasiva
  layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
  setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
  fspcom=read.table('fspcom.dat')
  dat = log(fspcom[,5])
  year=fspcom[,2]+1/12*fspcom[,3]
  t1=dat[1:length(dat)-1]
  t2=dat[2:length(dat)]
  plot(year[1:length(year)-1],t2-t1,type='l',
       main="Difference stationary of first Differencing of log(x(t))\nx(t) = S&P 500",
       xlab="Year",ylab="FD",col='blue',
       ylim=c(-.2,.2),cex.axis=1.1,cex.lab=1.5,lwd=2.2)
  setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
  dat <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
  year=dat[,1]+1/12*dat[,2]
  dat=log(dat[,3])
  t1=dat[1:length(dat)-1]
  t2=dat[2:length(dat)]
  plot(year[1:length(year)-1],t2-t1,type='l',
       main="Difference stationary of first Differencing of log(x(t))\nx(t) = NASDAQ",
       xlab="Year",ylab="FD",col='red',
       ylim=c(-.2,.2),cex.axis=1.1,cex.lab=1.5,lwd=2.2)

}
```

```
llt <-function()
{
  #Program used to Log linear trend and cycles for SP500 & NASDAQ index
  # Praba Siva
  # praba@umich.edu
  # @prabasiva
setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
fspcom=read.table('fspcom.dat')
year=fspcom[,2]
tsfspcom=ts(log(fspcom[,5]),start=year[1],
            end=c(year[length(year)],12),frequency=12)
loglinear=stl(log(tsfspcom),s.window=5)
plot(loglinear,main="A Seasonal-Trend Decomposition of S&P 500",
     col='red')
setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
dat <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
year=dat[,1]
dat=dat[,3]
tsnasdaq=ts(dat,start=year[1],
            end=c(year[length(year)]-1,12),frequency=12)
nloglinear=stl(log(tsnasdaq),s.window=5)
plot(nloglinear,main="A Seasonal-Trend Decomposition of NASDAQ",
     col="blue")
strend=loglinear$time.series[,2]
ntrend=nloglinear$time.series[,2]
plot(year[1:length(strend)],strend[1:length(strend)],
     col='blue',type='l',ylim=range(strend,ntrend))
lines(year[1:length(ntrend)],ntrend[1:length(ntrend)],
      col='red',type='l')
}
```

```

hpfilt <- function()
{
  #Program used to create the HP filter for lambda 80 & 800 for S&P 500 index
  #Load the file and invoke hpfilt()
  #Filename: hpfilter_slave.R
  # Praba Siva;praba@umich.edu; @prabasiva
library(mFilter)
library(latex2exp)
opar <- par(no.readonly=TRUE)
setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
fspcom=read.table('fspcom.dat')
dat = fspcom[,5]
syear=fspcom[,2]+1/12*fspcom[,3]
ldat=log(dat)
dat=ldat
dat.hp1 <- hpfilter(dat, freq=80,type="frequency",drift=FALSE)
dat.hp2 <- hpfilter(dat, freq=800,type="frequency",drift=FALSE)
par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
plot(syear,dat,  ylim=range(dat),
     main="S&P 500 Index ",
```

```
      col=2, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
plot(syear,dat.hp1$trend,  ylim=range(dat.hp1$trend),
      main="HP filter of S&P 500 Index: Trend,Lambda=80 ",
      col=4, xlab='praba siva', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
plot(syear,dat.hp1$cycle,  ylim=range(dat.hp1$cycle),
      main="HP filter of S&P 500 Index: Cycle,Lambda=80 ",
      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
plot(syear,dat,  ylim=range(dat),
      main="S&P 500 Index ",
      col=2, ylab="",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
plot(syear,dat.hp2$trend,  ylim=range(dat.hp1$trend),
      main="HP filter of S&P 500 Index: Trend,Lambda=800 ",
      col=4, xlab='praba siva', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.3,lwd=2.2)
plot(syear,dat.hp1$cycle,  ylim=range(dat.hp2$cycle),
      main="HP filter of S&P 500 Index: Cycle,Lambda=800 ",
      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
par(opar)
opar <- par(no.readonly=TRUE)
setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
nasda <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
nyear=nasda[,1]+1/12*nasda[,2]
nasda=nasda[,3]
lnasda=log(nasda)
nasda=lnasda
nasda.hp1 <- hpfilter(nasda, freq=80,type="frequency",drift=FALSE)
nasda.hp2 <- hpfilter(nasda, freq=800,type="frequency",drift=FALSE)
par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
plot(nyear,nasda,  xlab="Year",ylab="log s(t)",ylim=range(nasda),
      main="NASDAQ Index ",
      col=2, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
plot(nyear,nasda.hp1$trend,  xlab='Year', ylim=range(nasda.hp1$trend),
      main="HP filter of NASDAQ Index: Trend,Lambda=80 ",
      col=4, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
plot(nyear,nasda.hp1$cycle,  ylim=range(nasda.hp1$cycle), xlab="Year",
      main="HP filter of NASDAQ Index: Cycle,Lambda=80 ",
      col=3, type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
par(mfrow=c(3,1),mar=c(3,3,2,1),cex=.8)
plot(nyear,nasda,  ylim=range(nasda),
      main="NASDAQ Index ",
      col=2, ylab="log s(t)",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
plot(nyear,nasda.hp2$trend,  ylim=range(nasda.hp1$trend),
      main="HP filter of NASDAQ Index: Trend,Lambda=800 ",
      col=4, xlab='Year', ylab="log(s(t))",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
plot(nyear,nasda.hp1$cycle,  ylim=range(nasda.hp2$cycle),
      main="HP filter of NASDAQ Index: Cycle,Lambda=800 ",
      col=3, ylab="",type='l',cex.axis=1.1,cex.lab=1.5,lwd=2.2)
par(opar)
nasda.hp3 <- hpfilter(nasda, freq=1600,type="frequency",drift=FALSE)
nasda.hp4 <- hpfilter(nasda, freq=14400,type="frequency",drift=FALSE)
lambda=c(80,800,1600,14400);
c=1:4;
layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
plot(nyear,nasda.hp1$trend,ylab='Log NASDAQ',
      main=TeX('NASDAQ Trend HP filter with different $\\lambda$'),
      xlab='Year',col=c(1),type='l',cex.axis=1,cex.lab=1.3,lwd=2.2);
```

```
lines(nyear,nasda.hp2$trend,main=TeX('NASDAQ Trend HP filter with different $\\lambda$') ,
      col=c(2),type='l',cex.axis=1,cex.lab=1.1,lwd=2.2);
lines(nyear,nasda.hp3$trend,main=TeX('NASDAQ Trend HP filterwith different $\\lambda$'),
      col=c(3),type='l',cex.axis=1,cex.lab=1.1,lwd=2.2);
lines(nyear,nasda.hp4$trend,main=TeX('NASDAQ Trend HP filterwith different $\\lambda$'),
      col=c(4),type='l',cex.axis=1,cex.lab=1.1,lwd=2.2);
legend('topleft', legend=TeX(sprintf("$\\lambda = %d$", lambda)), lwd=1, col=c)
dat.hp3 <- hpfilter(dat, freq=1600,type="frequency",drift=FALSE)
dat.hp4 <- hpfilter(dat, freq=14400,type="frequency",drift=FALSE)
lambda=c(80,800,1600,14400);
c=1:4;
plot(syear,dat.hp1$trend,ylab='Log SP500',main=TeX('SP500 Trend HP filter with different ...
    $\\lambda$'),
     xlab='Year',col=c(1),type='l',cex.axis=1,cex.lab=1.3,lwd=2.2);
lines(syear,dat.hp2$trend,main=TeX('SP500 Trend HP filter with different $\\lambda$') ,
      col=c(2),type='l',cex.axis=1.1,cex.lab=1,lwd=2.2);
lines(syear,dat.hp3$trend,main=TeX('SP500 Trend HP filterwith different $\\lambda$'),
      col=c(3),type='l',cex.axis=1.1,cex.lab=1,lwd=2.2);
lines(syear,dat.hp4$trend,main=TeX('SP500 Trend HP filterwith different $\\lambda$'),
      col=c(4),type='l',cex.axis=1.1,cex.lab=1,lwd=2.2);
legend('topleft', legend=TeX(sprintf("$\\lambda = %d$", lambda)), lwd=1, col=c)

#Statistics of the HP Detrending
print('SP500 ')
print('HP Filter with Lambda = 80')
sprintf("Mean = %5f",mean(dat.hp1$cycle))
sprintf("SD = %5f",sd(dat.hp1$cycle))
sprintf("Variance = %5f",var(dat.hp1$cycle))
print('HP Filter with Lambda = 800')
sprintf("Mean = %5f",mean(dat.hp2$cycle))
sprintf("SD = %5f",sd(dat.hp2$cycle))
sprintf("Variance = %5f",var(dat.hp2$cycle))
print('HP Filter with Lambda = 1600')
sprintf("Mean = %5f",mean(dat.hp3$cycle))
sprintf("SD = %5f",sd(dat.hp3$cycle))
sprintf("Variance = %5f",var(dat.hp3$cycle))
print('HP Filter with Lambda = 14400')
sprintf("Mean = %5f",mean(dat.hp4$cycle))
sprintf("SD = %5f",sd(dat.hp4$cycle))
sprintf("Variance = %5f",var(dat.hp4$cycle))

print('NASDAQ ')
print('HP Filter with Lambda = 80')
xm=sprintf("Mean = %5f",mean(nasda.hp1$cycle))
xs=sprintf("SD = %5f",sd(nasda.hp1$cycle))
xv=sprintf("Variance = %5f",var(nasda.hp1$cycle))

print('HP Filter with Lambda = 800')
sprintf("Mean = %5f",mean(nasda.hp2$cycle))
sprintf("SD = %5f",sd(nasda.hp2$cycle))
sprintf("Variance = %5f",var(nasda.hp2$cycle))

print('HP Filter with Lambda = 1600')
sprintf("Mean = %5f",mean(nasda.hp3$cycle))
sprintf("SD = %5f",sd(nasda.hp3$cycle))
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/sivasp1/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]
```

```
dftrend=intercept+slope*dur
dfcycle=mort-dftrend
dfacf=acf(dfcycle,plot=FALSE,100);
hpf=hpfilter(mort,freq=14400)
layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
color={'blue'}
ac1=acf(hpf$cycle, ci.type = "ma",plot=FALSE,100)
plot(year,mort,main='Log SP500 index',
     xlab='Year',ylab=TeX('log (SP500(t))'),
     type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='red');
bc1=acf(diffxy,ci.type="ma",plot=FALSE,100)
plot(ac1,main='Autocorrelation of log SP500 HP Cycles'
     ,xlab='Lag',ylab='AC(1)')
lines(ac1$lag,ac1$acf,main='Autocorrelation of log SP500 HP Cycles',
      xlab='Lag',ylab='AC(1)',type='l',
      col='blue',cex.axis=1.1,cex.lab=1.1,lwd=3)
plot(bc1,main='Autocorrelation of log SP500 FD ',
     xlab='Lag',ylab='AC(1)')
lines(bc1$lag,bc1$acf,main='Autocorrelation of log SP500 FD',
      xlab='Lag',ylab='AC(1)',type='l',col='blue',lwd=3)
plot(dfacf,main='Autocorrelation of log-linear SP500  ',
     xlab='Lag',ylab='AC(1)')
lines(dfacf$lag,dfacf$acf,main='Autocorrelation of log-linear SP500 ',
      xlab='Lag',ylab='AC(1)',type='l',
      col='blue',lwd=3)
layout(matrix(c(1,2), 2,1, byrow = TRUE))
plot(year,mort,main='Log SP500 index',
     xlab='Year',ylab=TeX('log (SP500(t))'),
     type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='red');
lines(year,dftrend,main='Trend of Log SP500 index using Log-linear',
      xlab='Year',ylab=TeX('log-linear(SP500(t))'),
      type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='blue');
legend("bottomright",c("Trend"),lty=c(1),lwd=c(2.5),col=c("blue"))
plot(year,dfcycle,main='Cycle of Log SP500 index using Log-linear',
     xlab='Year',ylab=TeX('log-linear(SP500(t))'),
     type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='green');
layout(matrix(c(1,2), 2,1, byrow = TRUE))
plot(year,mort,main='Log SP500 index',
     xlab='Year',ylab=TeX('log (SP500(t))'),
     type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='red');
plot(year[1:length(diffxy)],diffxy,
     main='Cycle of Log SP500 index using Log-linear trend',
     xlab='Year',ylab=TeX('log-linear(SP500(t))'),type='l',
     cex.axis=1.1,cex.lab=1.1,lwd=3,col='green');
sta.sp500=list(mean(dfacf$acf),sd(dfacf$acf),var(dfacf$acf),corrlength(dfacf),
               mean(ac1$acf),sd(ac1$acf),var(ac1$acf),corrlength(ac1),
               mean(bc1$acf),sd(bc1$acf),var(bc1$acf),corrlength(bc1));
layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
setwd("/Users/sivasp1/Documents/2016/Personal/Praba/MATH599/program")
dat <- read.csv(file="nasdaq_ready.csv",head=TRUE,sep=",")
year=dat[,1]+1/12*dat[,2]
dat=dat[,3]
mort=log(dat)
le=length(dat)
x=mort[2:le]
y=mort[1:le-1]
```

```
diffxy=x-y
dur=1:length(year)
lmr=lm(mort¬dur)
intercept=coef(lmr)[1]
slope=coef(lmr)[2]
dftrend=intercept+slope*dur
dfcycle=mort-dftrend
dfacf=acf(dfcycle,plot=FALSE,100);
hpf=hpfilter(mort,freq=14400)
ac1=acf(hpf$cycle, ci.type = "ma",plot=FALSE,100)
bc1=acf(diffxy,ci.type="ma",plot=FALSE,100)
layout(matrix(c(1,2), 2,1, byrow = TRUE))
plot(year,mort,main='Log NASDAQ index',
      xlab='Year',ylab=TeX('log (NASDAQ(t))'),
      type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='red');
lines(year,dftrend,main='Trend of Log NASDAQ index using Log-linear',
       xlab='Year',ylab=TeX('log-linear(NASDAQ(t))'),
       type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='blue');
legend("bottomright",c("Trend"),lty=c(1),lwd=c(2.5),col=c("blue"))
plot(year,dfcycle,main='Cycle of Log NASDAQ index using Log-linear',
      xlab='Year',ylab=TeX('log-linear(NASDAQ(t))'),
      type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='green');
layout(matrix(c(1,2), 2,1, byrow = TRUE))
plot(year,mort,main='Log NASDAQ index',
      xlab='Year',ylab=TeX('log (NASDAQ(t))'),
      type='l',cex.axis=1.1,cex.lab=1.1,lwd=3,col='red');
plot(year[1:length(diffxy)],diffxy,
      main='Cycle of Log NASDAQ index using Log-linear trend',
      xlab='Year',ylab=TeX('log-linear(NASDAQ(t))'),type='l',
      cex.axis=1.1,cex.lab=1.1,lwd=3,col='green');
layout(matrix(c(1,2,3,4), 4,1, byrow = TRUE))
plot(year,mort,main='Log NASDAQ index',
      xlab='Year',ylab=TeX('log (NASDAQ(t))'),type='l',
      col='red',cex.axis=1.1,cex.lab=1.1,lwd=3);
plot(ac1,main='Autocorrelation of log NASDAQ HP Cycles',
      xlab='Lag',ylab='AC(1)')
lines(ac1$lag,ac1$acf,main='Autocorrelation of log NASDAQ HP Cycles',
       xlab='Lag',ylab='AC(1)',type='l',
       col='blue',cex.axis=1.1,cex.lab=1.1,lwd=3)
plot(bc1,main='Autocorrelation of log NASDAQ FD ',
      xlab='Lag',ylab='AC(1)')
lines(bc1$lag,bc1$acf,main='Autocorrelation of log NASDAQ FD',
       xlab='Lag',ylab='AC(1)',type='l',
       col='blue',cex.axis=1.1,cex.lab=1.1,lwd=3)
plot(dfacf,main='Autocorrelation of log-linear NASDAQ  ',
      xlab='Lag',ylab='AC(1)')
lines(dfacf$lag,dfacf$acf,main='Autocorrelation of log-linear NASDAQ ',
       xlab='Lag',ylab='AC(1)',type='l',col='blue',lwd=3)
layout(matrix(c(1,2,3,4,5,6), 3, 2, byrow = TRUE))
#par(mfrow=c(2,1),mar=c(3,3,2,1),cex=.8)
x=seq(-15,15,.1);
y=sin(x)
ac1=acf(y,lag.max=100,plot=FALSE);
plot(x,y,main='Sin wave',xlab='T',ylab='Sin(t)',type='l',
      col='red',cex.axis=1.1,cex.lab=1.1,lwd=3)
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$acf,type='l',col='blue',lwd=.2)
158   corrlength(ac1)
159   sta.nasdaq=list(mean(dfacf$acf),sd(dfacf$acf),var(dfacf$acf),corrlength(dfacf),
160                   mean(ac1$acf),sd(ac1$acf),var(ac1$acf),corrlength(ac1),
161                   mean(bc1$acf),sd(bc1$acf),var(bc1$acf),corrlength(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   corrlength <- function(acfvector)
169
170   {
171
172     ind=min(which(acfvector$acf<0));
173
174     return ...
          ((abs(acfvector$acf[ind])+abs(acfvector$acf[ind-1])/10)*(abs(acfvector$acf[ind-1]))+ind-1)
175
176   }
```

APPENDIX B

Matlab code that are used to do analysis

```
function DrawSinFourierGraph()
Fs = 1000;                    % Sampling frequency
T = 1/Fs;                     % Sampling period
L = 200;                      % Length of signal
t = (0:L-1)*T;                % Time vect
f = [50,150,300];
x1 = sin(2*pi*f(1)*t);        % First row wave
x2 = sin(2*pi*f(2)*t);        % Second row wave
x3 = sin(2*pi*f(3)*t);        % Third row wave

X = [x1; x2; x3;];

figure;

for i = 1:3
    g=subplot(3,2,i+i-1)
    plot(t(1:L),X(i,1:L),'r','LineWidth',1)
    ylabel('sin(2\pift)','FontSize',16,'FontWeight','bold')
    xlabel('\fontname{Helvetica} Time','FontSize',16,'FontWeight','bold')
    title(['A sine wave of frequency f  = ',num2str(f(i)),' in the Time ...
        Domain'],'FontSize',18,'FontWeight','bold')
    p=get(g,'position');
    p(1)=.7*p(1);
    p(4)=1.1*p(4);
    set(g,'position',p);
end


n = 2^nextpow2(L);
dim = 2;
Y = fft(X,n,dim);
P2 = abs(Y/n);
```

```
P1 = P2(:,1:n/2+1);
P1(:,2:end-1) = 2*P1(:,2:end-1);


for i=1:3
    g=subplot(3,2,i*2);
    plot(0:(Fs/n):(Fs-Fs/n),P2(i,1:n),'b','LineWidth',1)
    title(['Sin wave of frequency  ',num2str(f(i)), ' in the Frequency ...
        Domain'],'FontSize',18,'FontWeight','bold')
    ylabel(' |F(s)|','FontSize',16,'FontWeight','bold')
    xlabel('Freqency','FontSize',16,'FontWeight','bold')
    p=get(g,'position');
    p(1)=.9*p(1);
    p(4)=1.1*p(4);
    set(g,'position',p);
end
```

```
function SPNASDAQFourier()
%Program used to Discrete Fourier transform
% Praba Siva
% praba@umich.edu
% @prabasiva
% Filename: SPNASDAQFourier.m
Fs = 1000;              % Sampling frequency
T = 1/Fs;               % Sampling period
L = 1000;               % Length of signal
t = (0:L-1)*T;          % Time vector
S = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
X = S + 2*randn(size(t));
wsize=1000;
g=subplot(5,2,1);
plot(1000*t(1:wsize),X(1:wsize),'r','LineWidth',1);
title('Signal Corrupted with Zero-Mean Random Noise','FontSize',18,'FontWeight','bold')
xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
ylabel('f(t)','FontSize',16,'FontWeight','bold')
%N=2^nextpow2(L);
Y = fft(X);
P2 = abs(Y/L);P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = Fs*(0:(L/2))/L;
g=subplot(5,2,2);
plot(f,P1,'b','LineWidth',1)
title('Single-Sided Amplitude Spectrum of f(t)','FontSize',16,'FontWeight','bold')
xlabel('w','FontSize',16,'FontWeight','bold')
ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
g=subplot(5,2,3);
plot(1000*t(1:wsize),S(1:wsize),'r','LineWidth',1);
title('Signal with No Random Noise','FontSize',18,'FontWeight','bold')
xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
ylabel('f(t)','FontSize',16,'FontWeight','bold')
Y = fft(S);
P2 = abs(Y/L);
P1 = P2(1:L/2+1);
```

```
P1(2:end-1) = 2*P1(2:end-1);
f = Fs*(0:(L/2))/L;
g=subplot(5,2,4);
plot(f,P1,'b','LineWidth',1)
title('Single-Sided Amplitude Spectrum of f(t)','FontSize',18,'FontWeight','bold')
xlabel('w','FontSize',16,'FontWeight','bold')
ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
Fs = 100;            % Sampling frequency
t = -0.5:1/Fs:0.5;   % Time vector
L = length(t);       % Signal length
X = 1/(4*sqrt(2*pi*0.01))*(exp(-t.^2/(2*0.01)));
g=subplot(5,2,5)
plot(t,X,'r','LineWidth',1)
title('Gaussian Pulse in Time Domain','FontSize',18,'FontWeight','bold')
xlabel('Time (t)','FontSize',16,'FontWeight','bold')
ylabel('f(t)','FontSize',16,'FontWeight','bold')
n = 2^nextpow2(L);
Y = fft(X,n);
f = Fs*(0:(n/2))/n;
P = abs(Y/n);
g=subplot(5,2,6)
plot(fftshift(P),'b','LineWidth',1)
%plot(f,fftshift(P(1:n/2+1)),'b','LineWidth',1)
%plot(1:wsize,P(1:wsize),'b','LineWidth',1)
title('Gaussian Pulse in Frequency Domain','FontSize',18,'FontWeight','bold')
xlabel('Frequency (w)','FontSize',16,'FontWeight','bold')
ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
[naq,year]=getData(2);
[dum,naq]=hpfilter(naq,14400);
g=subplot(5,2,7)
plot(year,naq,'r','LineWidth',1);
title('HPFilter of Log Nasdaq','FontSize',18,'FontWeight','bold');
xlabel('Year.month','FontSize',16,'FontWeight','bold');
ylabel('Cycles','FontSize',16,'FontWeight','bold');
Fs = 1000;
[L,tp]=size(naq)
n = 2^nextpow2(L);
Y = fft(naq,n);
f = Fs*(0:(n/2))/n;
P = abs(Y/n);
g=subplot(5,2,8)
%plot(f,P(1:n/2+1),'b','LineWidth',1)
plot(1:n/2+1,P(1:n/2+1),'b','LineWidth',1)
title('HPFilter of NASDAQ in Frequency Domain','FontSize',18,'FontWeight','bold')
xlabel('Frequency (w)','FontSize',16,'FontWeight','bold')
ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
[sp500,year]=getData(1);
[dum,sp500]=hpfilter(sp500,14400);
g=subplot(5,2,9)
plot(year,sp500,'r','LineWidth',1);
title('HPFilter of Log S&P 500','FontSize',18,'FontWeight','bold');
xlabel('Year.month','FontSize',16,'FontWeight','bold');
ylabel('Cycles','FontSize',16,'FontWeight','bold');
[L,tp]=size(sp500)
n = 2^nextpow2(L);
Y = fft(sp500,n);
```

```
f = Fs*(0:(n/2))/n;
P = abs(Y/n);
g=subplot(5,2,10)
%plot(f,P(1:n/2+1),'b','LineWidth',1);
plot(1:n/2+1,P(1:n/2+1),'b','LineWidth',1);
title('HPFilter of S&P in Frequency Domain','FontSize',18,'FontWeight','bold')
xlabel('Frequency (w)','FontSize',16,'FontWeight','bold')
ylabel('|F(w)|','FontSize',16,'FontWeight','bold')
```

```
function ghamwin()
s={'Gaussian Window' 'Hamming Window'};
wlen=25;
figure;
for fla=0:1
    if fla<1
    % form a periodic hamming window
        win = hamming(wlen, 'periodic');
    else
        win=gausswin(wlen)
    end
    g=subplot(1,2,1+fla);
    plot(abs(win),'r','LineWidth',1);
    xlabel('n (N=25)','FontSize',16,'FontWeight','bold')
    if fla ==0
        title('\sigma = 2.5 Gaussian Window ...
            function','interpreter','Tex','Fontsize',16,'FontWeight','bold')
       ylabel('e^{-n^2/2\sigma^2}','interpreter','Tex','FontSize',20)
    else
        title([s(fla+1),'function'],'FontSize',16,'FontWeight','bold')
        ylabel('0.54-0.46cos(2\pi*n/N)','interpreter','Tex','FontSize',20)
    end
end
```

```
function drawSpecEg()
close all;
clear all;
 Fs = 1000;            % Sampling frequency
    T = 1/Fs;             % Sampling period
    L = 1000;             % Length of signal
    t = (0:L-1)*T;        % Time vector
    S = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
    Y = S + 2*randn(size(t));
s={'using Gaussian Window' 'using Hamming Window'};
wlen=2^6;
    figure;

for fla=0:1
    g=subplot(2,3,1+(3*fla));
    if fla==1
      X=S;
```

```
18      else
19          X=Y;
20      end;
21      plot(L*t,X,'r','LineWidth',1);
22
23      if fla==1
24
25          title('Signal with No Random Noise','FontSize',18,'FontWeight','bold')
26      else
27          title('Signal Corrupted with Zero-Mean Random ...
                Noise','FontSize',18,'FontWeight','bold')
28      end
29      xlabel('t (milliseconds)','FontSize',16,'FontWeight','bold')
30      ylabel('f(t)','FontSize',16,'FontWeight','bold')
31     g=subplot(2,3,2+(3*fla));
32    spectrogram(X,gausswin(wlen));
33     ylabel(g,'Time (ms)');
34       title(['|STFT(t,\omega)|', s(1)],'FontSize',16,'FontWeight','bold')
35     g=subplot(2,3,3+(3*fla));
36     spectrogram(X,hamming(wlen));
37       title(['|STFT(t,\omega)|', s(2)],'FontSize',16,'FontWeight','bold')
38       ylabel(g,'Time (ms)');
39      end
40
41  end
```

```
1   function drawSpecEco()
2   close all;
3   clear all;
4   [sp500,syear]=getData(1);
5   [naq,nyear]=getData(2);
6
7   s={'using Gaussian Window' 'using Hamming Window'};
8   wlen=2^6;
9       figure;
10
11  for fla=0:1
12      g=subplot(2,3,1+(3*fla));
13      if fla==1
14         [trend, X] = hpfilter(sp500,14400);
15         xaxis=syear;
16      else
17          [trend, X] = hpfilter(naq,14400);
18          xaxis=nyear;
19      end;
20      plot(xaxis,X,'r','LineWidth',1);
21
22      if fla==1
23          title('HP Filter cycles (\lambda =14400) for SP500','FontSize',18,'FontWeight','bold')
24      else
25          title('HP Filter cycles (\lambda =14400) for ...
                NASDAQ','FontSize',18,'FontWeight','bold')
26      end
```

```
27      xlabel('Years','FontSize',16,'FontWeight','bold')
28      ylabel('Cycles','FontSize',16,'FontWeight','bold')
29     g=subplot(2,3,2+(3*fla));
30    spectrogram(X,gausswin(wlen));
31       title(['|STFT(t,\omega)|', s(1)],'FontSize',16,'FontWeight','bold')
32       ylabel(g,'Time');
33      g=subplot(2,3,3+(3*fla));
34     spectrogram(X,hamming(wlen));
35     ylabel(g,'Time');
36       title(['|STFT(t,\omega)|', s(2)],'FontSize',16,'FontWeight','bold')
37      end
38
39  end
```

```
1   function drawSTFTECo3D()
2   close all;
3   clear all;
4   [sp500,syear]=getData(1);
5   [naq,nyear]=getData(2);
6
7   s={'using Gaussian Window' 'using Hamming Window'};
8   wlen=2^6;
9       figure;
10
11  for fla=0:1
12      g=subplot(2,3,1+(3*fla));
13      if fla==1
14         [trend, X] = hpfilter(sp500,14400);
15         xaxis=syear;
16      else
17          [trend, X] = hpfilter(naq,14400);
18          xaxis=nyear;
19      end;
20      plot(xaxis,X,'r','LineWidth',1);
21
22      if fla==1
23          title('HP Filter cycles (\lambda =14400) for SP500','FontSize',18,'FontWeight','bold')
24      else
25          title('HP Filter cycles (\lambda =14400) for ...
                NASDAQ','FontSize',18,'FontWeight','bold')
26      end
27      xlabel('Years','FontSize',16,'FontWeight','bold')
28      ylabel('Cycles','FontSize',16,'FontWeight','bold')
29     g=subplot(2,3,2+(3*fla));
30     out=spectrogram(X,gausswin(wlen));
31     surf(abs(out));
32      title(['|STFT(t,\omega)|', s(1)],'FontSize',16,'FontWeight','bold')
33      g=subplot(2,3,3+(3*fla));
34      out=spectrogram(X,hamming(wlen));
35      surf(abs(out));
36      title(['|STFT(t,\omega)|', s(2)],'FontSize',16,'FontWeight','bold')
37      end
38
```

```
39 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/sivasp1/Documents/2016/Personal/Praba/MATH599/program/fspcom.dat');
4       [maxx,maxy]=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/sivasp1/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]=hpfilter(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  sig = [.1,.8,1,1.2,10];
12  j=1;
13  [dummy,winx]=size(sig);
14  winy=2;
15  for i = 1:winx
16      t=-128:127;
17      sigma=sig(i);
```

```
18      coef=nthroot(pi*sigma*sigma,-4);
19      expo=(t.*t)/2*sigma*sigma;
20      g=coef*exp(-expo);
21      subplot(winx,winy,j);
22      plot(t,g);
23      title(['Sigma = ',num2str(sigma)]);
24      set(gca,'fontsize',12)
25
26      j=j+1;
27      [wd,freq]=wig2(g);
28      subplot(winx,winy,j)
29      j=j+1;
30      contour(t,freq,wd',8)
31      set(gca,'fontsize',12)
32  end
```

```
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      Δm = 8;
12      %M=16;
13      M=50
14      Δn=4;
15      %nn=32;
16      nn=100;
17
18
19      [s2,s1]=hpfilter(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
36      end
37
```

```
38      [s2,s1]=hpfilter(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 ...
                log(sp500)','FontSize',16,'FontWeight','bold','Color','r')
60      else
61          text(-15, 0.4, 'Fig a: Gabor Coefficient for ...
                log(sp500)','FontSize',16,'FontWeight','bold','Color','r')
62      end;
63      xlabel('Time','FontSize',12,'FontWeight','bold','Color','b')
64      h=get(gca,'xlabel');
65      set(h,'rotation',30)
66      ylabel('Frequency','FontSize',12,'FontWeight','bold','Color','b')
67      h=get(gca,'ylabel');
68      set(h,'Position',get(h,'Position') +[2 4 0])
69      set(h,'rotation',140)
70      zlabel('|C(m,n|^2','FontSize',12,'FontWeight','bold','Color','b')
71
72      subplot(1,2,2);
73      surf(abs(c2));
74      colormap hsv;
75      if M==16
76          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,syear]=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]=hpfilter(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.*h(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
31          end
32
33      end
34      [s2,s1]=hpfilter(naq,1600);
35      s1=s1';
36      L=length(s1);
37      t=1:L;
38      N=L/2;
39      nn2=nn/2;
40      sigma=sqrt((Δm*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      p1=plot(li,'LineWidth',6,'Color','b');
96      legend(p1,'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     p1=plot(li,'LineWidth',6,'Color','b');
108     legend(p1,'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  p1=plot(li,'LineWidth',6,'Color','b');
118  legend(p1,'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 ...
                 log(sp500)','FontSize',16,'FontWeight','bold','Color','r')
130      else
131          text(-15, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
                 log(sp500)','FontSize',16,'FontWeight','bold','Color','r')
132      end;
133      xlabel('Time','FontSize',12,'FontWeight','bold','Color','b')
134      h=get(gca,'xlabel');
135      set(h,'rotation',30)
136      ylabel('Frequency','FontSize',12,'FontWeight','bold','Color','b')
137      h=get(gca,'ylabel');
138      set(h,'Position',get(h,'Position') +[2 4 0])
139      set(h,'rotation',140)
140      zlabel('|C(m,n|^2','FontSize',12,'FontWeight','bold','Color','b')
141
142      subplot(1,2,2);
143      surf(abs(c2));
144      colormap hsv;
145      if M==16
146          text(-5, 0.4, 'Fig a: Filtered Gabor Coefficient for ...
                 log(nasdaq)','FontSize',16,'FontWeight','bold','Color','r')
147      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
```

```
function myreconsfromgabor()
%Program used to Gabor Coefficients for SP500 & NASDAQ
% Praba Siva
% praba@umich.edu
% @prabasiva
% Filename: myreconsfromgabor.m

        close all;
        clear all;
            [sp500,syear]=getData(1);
          %  sp500=log(sp500);
            [s2,s1]=hpfilter(sp500,1600);


        Δm = 8;
        M=17;
       % M=50;
        Δn=4;
        nn=84;
        %nn=100;
        thrcont=3;

        s1=s1';
        L=length(s1);
        t=1:L;
        N=L/2;
        nn2=nn/2;
        sigma=sqrt((Δm*L)/(Δn * 2 * pi));
        c=nthroot(pi*sigma*sigma,-4);
        h0 = @(b) c*exp(-((b.*b)/(2*sigma*sigma)));
        h = @(ii) h0( mod(ii + N, L)-N);
        for m = 1:M
            for n = 1:nn2
                c1(m, n)= sum(s1.*h(mod(t - m*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
            end

        end


        [m,n]=size(c1);

        H=0.5;
        %thr=max(abs(c1));
        %spmask = (max(thr)-min(thr))/thrcont;
        spmask=mean(c1)+H*std(c1);
        for k = 1:m
            for j = 1:n
                if abs(c1(k,j)) < abs(spmask(k))
                    sp500maskmatrix(k,j)=0;
                else
                    sp500maskmatrix(k,j)=1;
                end;
            end;
        end;

```

```
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             Δm = 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((Δm*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*Δm,L)).*exp(-2*pi*i*Δn*n*t/L));
135              end
136
137          end
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');
208 end
```

```
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]=hpfilter(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
35          end
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
```

```
% Filename: myFDfilter.m
close all;
clear all;
[sp500,syear]=getData(1);
process(sp500,1);
[naq,nyear]=getData(2);
process(naq,2);


function process(sp500,flag)

 % First differening
 % Graph looks identifical as shown in the P.Chen's paper
 % P.Chen's paper says it is log of sp500 data but he used sp500
 % The graph in his paper shows it all.

    len=size(sp500);
    t1=sp500(1:len-1);
    t2=sp500(2:len);
    fd=t2-t1;
    lenfd=size(fd);
    Δ=40;
    x=fd(1:lenfd-Δ);
    y=fd(Δ+1:lenfd);
    figure;
    subplot(2,1,1);
    scatter(x,y,'b');
    xlabel('X(t)','Fontsize',20);
    ylabel('X(t+T)', 'Fontsize', 20);
    if flag ==1
        title('FD Series for SP500 Index','Fontsize',20);
    else
        title('FD Series for NASDAQ Index','Fontsize',20);
    end;
    y=fd;
    n=size(y);
    % C and Phi values are given in the P.Chen's paper.

    c = 0.006*0.002;
    phi = [0.000265*0.043, -0.81*0.043];

    % White noise created based on the standard deviation value
    noise = 0.033*randn(n(1),1);

    for t=3:n
       x(t-2) = c + phi(1)*y(t-1) + phi(2)*y(t-2) +noise(t-2);
    end
    lenfd=size(x);
    Δ=5;
    x1=x(1:lenfd-Δ);
    y1=x(Δ+1:lenfd);
    subplot(2,1,2);
    scatter(x1,y1,'r');
    xlabel('X(I)','Fontsize',20);
    ylabel('X(I+T)', 'Fontsize', 20);
    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
```

```
1   function mydft()
2
3    abc=dftmtx(50);
4   subplot(2,2,1);surf(real(abc));
5   title('Discrete Fourier Transform - Real Part','FontSize',18,'FontWeight','bold')
6
7    subplot(2,2,2);surf(imag(abc));
8    title('Discrete Fourier Transform - Imaginary Part','FontSize',18,'FontWeight','bold')
9
10
11  abc=inv(abc);
12  subplot(2,2,3);surf(real(abc));
13  title('Inverse Discrete Fourier Transform - Real Part','FontSize',18,'FontWeight','bold')
14
15   subplot(2,2,4);surf(imag(abc));
16   title('Inverse Discrete Fourier Transform - Imaginary ...
         Part','FontSize',18,'FontWeight','bold')
```

```
1   function drawCorrDim()
2   %Program used to Correlation Dimension for both SP500 & NASDAQ
3   % Praba Siva
4   % praba@umich.edu
5   % @prabasiva
6   % Filename: drawCorrDim.m
7
8   close all;
9   clear all;
10  titlestr={' for SP500',' for NASDAQ'};
11  for i = 1:2
12      [index,year]=getData(i);
13      [trend,cycles]=hpfilter(index,14400);
14      N=size(cycles);
15      N=N(1);
16      figure;
17      for delay = 2:2:8
18          subplot(2,2,delay/2);
19          for ed = 2:2:30
20              [r,c] = c2(cycles,ed,delay,400,500);
21              plot(r,c,'LineWidth',2);
22              set(gca,'fontsize',14,'fontweight','bold');
23
24              %xlabel('$\log_{10}(\epsilon)$','Interpreter','Latex','FontSize',14,'FontWeight','bold')
25              %ylabel('$\log_{10} ...
```

```
                C(\epsilon)$','Interpreter','Latex','FontSize',14,'FontWeight','bold')
26              xlabel('log10(\epsilon)','FontSize',18,'FontWeight','bold')
27              ylabel('log10 C(\epsilon)','FontSize',18,'FontWeight','bold')
28
29              if i==1
30                   title({['Correlation sum for SP500 indexes with embedding dimensions = ...
                         2,4,..30']
31                        ;['and delay = ',num2str(delay)]},'fontsize',16);
32              else
33                   title({['Correlation sum for NASDAQ indexes with embedding dimensions = ...
                        2,4,..30']
34                       ;['and delay = ',num2str(delay)]},'fontsize',16);
35
36              end
37              textlegmain(ed/2)={['m = ',num2str(ed)]};
38              edmat(delay/2,ed/2)=abs(c/r);
39              hold on;
40          end;
41          hold on;
42          legend(textlegmain,'Location','northwest','FontSize',8,'FontWeight','bold');
43          legend('show');
44          hold on;
45      end;
46      figure;
47  for delay = 2:2:8
48      plot(2:2:30,edmat(delay/2,:),'LineWidth',3);
49      set(gca,'fontsize',14,'fontweight','bold');
50      xlabel('Embedding dimension','FontSize',18,'FontWeight','bold');
51      ylabel('Correlation Dimension','FontSize',18,'FontWeight','bold');
52      if i==1
53          title('SP 500 Index','FontSize',24,'FontWeight','bold');
54      else
55          title('NASDAQ index','FontSize',24,'FontWeight','bold');
56      end
57      textleg(delay/2)={['Delay = ',num2str(delay)]};
58      mean(edmat(delay/2,:));
59      hold on;
60      end;
61      cdim= mean(mean(edmat));
62      ldata(1:15)=cdim;
63      plot(2:2:30,ldata,'--','LineWidth',3);
64      hold on;
65      legend(textleg,'Location','northwest','FontSize',10,'FontWeight','bold');
66      legend('show');
67      hold on;
68  end;
```

```
1  %Program used to Time Frequency for SP500 & NASDAQ
2  % Praba Siva
3  % praba@umich.edu
4  % @prabasiva
5  % Filename: spectrumAnalysis3.m
6
```

```
 7
 8  function spectrumAnalysis3()
 9  close all;
10  [sp500,syear]=getData(1);
11  [trend, sp] = hpfilter(sp500,14400);
12  [naq,nyear]=getData(2);
13  [trend, na] = hpfilter(naq,14400);
14  nSamp = 2000;
15  t = (0:nSamp-1)'/nSamp;
16  t1 = t(1:int16(nSamp/3));
17  t2 = t(int16(nSamp/3)+1:2*int16(nSamp/3));
18  t3 = t(2*int16(nSamp/3)+1:nSamp);
19  x11 = sin(2*pi*300*t1);
20  x12 = sin(2*pi*200*t2);
21  x13 = sin(2*pi*100*t3);
22  x=[x11;x12;x13];
23  N=length(x);
24  win=100;
25      subplot(5,3,1);
26      plot(t,x);
27       title('Signal with three different frequencies varying in ...
             time','FontSize',18,'FontWeight','bold')
28          xlabel('t (milliseconds)','FontSize',14,'FontWeight','bold')
29          ylabel('f(t)','FontSize',14,'FontWeight','bold')
30      subplot(5,3,4);
31      plot(abs(fft(x)));
32       title('Frequency Domain','FontSize',18,'FontWeight','bold')
33          xlabel('Frequency','FontSize',14,'FontWeight','bold')
34          ylabel('|F(s)|','FontSize',14,'FontWeight','bold')
35      subplot(5,3,7);
36           spectrogram(x,[],[],[],N,'yaxis')
37       title('Short Time Frequency Analysis (STFA)','FontSize',18,'FontWeight','bold')
38
39          xlabel('Time','FontSize',14,'FontWeight','bold')
40          ylabel('Frequency','FontSize',14,'FontWeight','bold')
41  subplot(5,3,10);
42           spectrogram(x,gausswin(win),[],[],N,'yaxis')
43      title({'Gaussian Window of window length = 100'},...
44          'FontSize',16,'FontWeight','bold')
45
46          xlabel('Time','FontSize',14,'FontWeight','bold')
47          ylabel('Frequency','FontSize',14,'FontWeight','bold')
48  subplot(5,3,13);
49           spectrogram(x,hamming(win),[],[],N,'yaxis')
50
51      title({'Hamming Window of window length = 100'},...
52          'FontSize',16,'FontWeight','bold')
53
54          xlabel('Time','FontSize',14,'FontWeight','bold')
55          ylabel('Frequency','FontSize',14,'FontWeight','bold')
56
57
58      subplot(5,3,2);
59      plot(syear,sp);
60      title('S&P Index - HP Filter Cycles \lambda = 14400','FontSize',18,'FontWeight','bold');
61          xlabel('Years','FontSize',14,'FontWeight','bold')
```

```
62          ylabel('Cycle','FontSize',14,'FontWeight','bold')
63
64      subplot(5,3,5);
65      plot(abs(fft(sp)));
66      title('S&P - HP Filter Cycles in Frequency Domain','FontSize',18, 'FontWeight','bold')
67          xlabel('Frequency','FontSize',14,'FontWeight','bold')
68          ylabel('|F(s)|','FontSize',14,'FontWeight','bold')
69
70      subplot(5,3,8);
71          spectrogram(sp,[],[],[],4/(1e7),'yaxis');
72
73      title('STFA of S&P - HP Filter Cycles \lambda = 14400','FontSize',16,'FontWeight','bold')
74          xlabel('Time in Years','FontSize',14,'FontWeight','bold')
75          ylabel('Frequency','FontSize',14,'FontWeight','bold')
76
77
78      subplot(5,3,11);
79          spectrogram(sp,gausswin(win),[],[],4/(1e7),'yaxis');
80
81      title({'Gaussian Window of window length = 100','S&P - HP Filter Cycles \lambda = ...
            14400'},...
82          'FontSize',16,'FontWeight','bold')
83          xlabel('Time in Years','FontSize',14,'FontWeight','bold')
84          ylabel('Frequency','FontSize',14,'FontWeight','bold')
85
86      subplot(5,3,14);
87          spectrogram(sp,hamming(win),[],[],4/(1e7),'yaxis');
88
89      title({'Hamming Window of window length = 100','S&P - HP Filter Cycles \lambda = ...
            14400'},...
90          'FontSize',16,'FontWeight','bold')
91          xlabel('Time in Years','FontSize',14,'FontWeight','bold')
92          ylabel('Frequency','FontSize',14,'FontWeight','bold')
93
94
95      subplot(5,3,3);
96      plot(nyear,na);
97      title('NASDAQ Index - HP Filter Cycles \lambda = 14400','FontSize',18,'FontWeight','bold')
98          xlabel('Years','FontSize',14,'FontWeight','bold')
99          ylabel('Cycle','FontSize',14,'FontWeight','bold')
100     subplot(5,3,6);
101     plot(abs(fft(na)));
102     title('NASDAQ - HP Filter Cycles in Frequency Domain','FontSize',18,'FontWeight','bold')
103         xlabel('Frequency','FontSize',14,'FontWeight','bold')
104         ylabel('|F(s)|','FontSize',14,'FontWeight','bold')
105
106     subplot(5,3,9);
107         spectrogram(na,[],[],[],4/(1e7),'yaxis')
108
109     title('STFA of NASDAQ - HP Filter Cycles \lambda = ...
            14400','FontSize',16,'FontWeight','bold')
110         xlabel('Time in Years','FontSize',14,'FontWeight','bold')
111         ylabel('Frequency','FontSize',14,'FontWeight','bold')
112
113     subplot(5,3,12);
114         spectrogram(na,gausswin(win),[],[],4/(1e7),'yaxis')
```

```
115
116       title({'Gaussian Window of window length = 100','NASDAQ - HP Filter Cycles \lambda = ...
              14400'},...
117           'FontSize',16,'FontWeight','bold')
118           xlabel('Time in Years','FontSize',14,'FontWeight','bold')
119           ylabel('Frequency','FontSize',14,'FontWeight','bold')
120
121       subplot(5,3,15);
122           spectrogram(na,hamming(win),[],[],4/(1e7),'yaxis')
123
124       title({'Hamming Window of window length = 100','NASDAQ - HP Filter Cycles \lambda = ...
              14400'},...
125           'FontSize',16,'FontWeight','bold')
126           xlabel('Time in Years','FontSize',14,'FontWeight','bold')
127           ylabel('Frequency','FontSize',14,'FontWeight','bold')
128   end
```

APPENDIX C

Mathematical Proof

C.1 Gabor Elementary function

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

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.

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

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

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

when t_0 is 0, then

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

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 $a = \alpha^2$

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

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

α 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 Δt 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 Δt

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

Straight forward steps to simply the value of Δt

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

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

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

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

Let me do the similar steps to calculate the effective frequency Δf. 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 Δf 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 $a = 2(\frac{\pi}{\alpha})^2$

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

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

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

Let me take the numerator now.

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

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

Substitute β 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 β

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

Apply the value of numerator and denominator of Δf

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

Step wise simplification steps to get the value of Δf

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

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

Apply both the value of Δf and Δt 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.