DYNAMIC MODELING AND FORECASTING ALGORITHMS
FOR FINANCIAL DATA SYSTEMS

by

MANISH MAHAJAN

A dissertation submitted to the
Graduate School—New Brunswick
Rutgers, The State University of New Jersey
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
Graduate Program in Electrical and Computer Engineering

Written under the direction of
Dr. N. N. Puri

and approved by

______________________________

______________________________

______________________________

______________________________

______________________________

New Brunswick, New Jersey
January, 2011
ABSTRACT OF THE DISSERTATION

Dynamic Modeling and Forecasting Algorithms for Financial Data Systems

by Manish Mahajan

Dissertation Director: Dr. N. N. Puri

It is a valid question that why a Control Systems Engineer would be interested in dealing with financial instruments. Financial instruments involving option theory are very elegant, math oriented and practical. These mathematical tools have created a new industry known as 'Derivative Industry' or 'Hedge-Fund Industry' or so called 'Risk-Management Industry'.

This thesis is aimed at developing investment strategies involving the decision making needs via control system techniques. The problem, in general, is computationally challenging particularly when investment of many securities is involved resulting in a high dimensional computational framework. Furthermore, complications may arise due to realistic restrictions and non-linearities. The various areas of financial engineering are very fertile for the application of the system methodology and control theory techniques. Modeling, optimization, identification and computational methods used in the Systems Engineering can be successfully applied to the financial instruments. The ideas developed in this thesis are more about the scientific reasoning involving financial instruments rather than specific situations alone. Major contribution of this thesis is the time series optimal prediction filter and the development of the Dynamic Modeling and Forecasting Algorithm (DMFA). The proposed algorithm predicts the next data point of the financial time series while dynamically computing the parameters from existing data. The computation of the parameters is optimized by use of the recursive
matrix inversion algorithm. The system is solved via an innovative technique of inversion such that it avoids explicit inversion of more than a 2 X 2 matrix and computation of higher dimensional determinants and co-factors. This results in new contributions to computation finance and numerical methodology along with arbitrage decision and hedging strategies under market uncertainties as well as robust control applications. The minimum mean-square algorithm used assures system stability via poles within the unit circle. The DMFA method is a superior auto regression (AR) model as a general system of time-series realizations in-order to calculate the coefficients that fit the model for a better prediction. Theoretical modeling and market specific volatility models, updated volatility computation are derived from the observation data.
Dedication

During the trials and tribulations of this endeavor, I am deeply grateful to my thesis advisor, Dr. N. N. Puri, my graduate director, Dr. Zoran Gajic and my parents for their support and encouragement. Finally, I want to thank my wife Sonal, who has maintained her own and my equilibrium through this process.
# Table of Contents

Abstract .............................................................. ii

Dedication ......................................................... iv

List of Figures .................................................... vii

1. Introduction ................................................... 1
   1.1. Problem Statement ..................................... 2
   1.2. Practical Implications for Financial Instruments .... 3

2. Dynamic Modeling and Forecast Algorithm ................. 5
   2.1. Introduction ........................................... 5
   2.2. Literature ............................................ 5
   2.3. Preliminaries ......................................... 6
   2.4. Dynamic Model Parameter Estimation Algorithm .... 8
   2.5. Recursion Algorithm for solution of Eq. 2.3 .......... 10
      2.5.1. Recursive Inversion of the matrix $B_k(\tau)$ .... 10
      2.5.2. Incremental Inversion Algorithm for Solution of Eq. 2.3 11
   2.6. Prediction Algorithm .................................. 12
   2.7. Parameter Updating ................................... 13
   2.8. Portfolio Optimization ............................... 13
   2.9. Conclusion .......................................... 15
   2.10. Future Research Directions for Modeling and Forecasting 15
   2.11. DMFA Performance ..................................... 16

   3.1. State of the Art ...................................... 19
List of Figures

2.1. Data set $\{d_k(\tau, i)_{i=1}^{n+m}\}$ formation from data $\{d_k(t)_{t=1}^{\infty}\}$, $t = 1, 2, \cdots k = 1, \cdots, N$ 7

2.2. Prediction Algorithm ................................................................. 13

2.3. Portfolio forecast and optimization ................................................... 15

2.4. Portfolio estimation algorithm, $m = 3$ parameters, $n = 20$ initial data .......... 17

2.5. Portfolio estimation algorithm, $m = 3$ parameters, $n = 30$ initial data .......... 17

2.6. Portfolio estimation algorithm, $m = 4$ parameters, $n = 35$ initial data .......... 18

2.7. Portfolio estimation algorithm, $m = 5$ parameters, $n = 40$ initial data .......... 18

3.1. Black-Scholes Risk Free Algorithm .................................................. 41

3.2. Time-Price Grid ............................................................................ 55
Chapter 1
Introduction

It is valid to ask why a Control Systems Engineer would be interested in dealing with financial instruments. Financial instruments involving option theory are very elegant, math oriented and practical. They have all the control elements and lend themselves easily to the ideas from Control Theory involving control of industrial processes.

This thesis is aimed at developing investment strategies involving the decision making needs via control system techniques. The problem, in general, is computationally challenging particularly when investment of many securities is involved resulting in a high dimensional computational framework. Furthermore, complications may arise due to realistic restrictions and non-linearities. The various areas of financial engineering are very fertile for the application of the system methodology and control theory techniques. Modeling, optimization, identification and computational methods used in the Systems Engineering can be successfully applied to the financial instruments. The ideas developed in this thesis are more about the scientific reasoning involving financial instruments rather than specific situations alone.

Latest Research activities and Trends in Financial Instruments

(1) New contributions in Computational Finance and Numerical Methodology.

(2) Data driven theoretical modeling and market specific volatility models and updated volatility computation from the observation data.

(3) Control system tools involving optimization for the risk management. Formulation of Lagrangian and Hamiltonian approach to portfolio optimization and risk sensitive control decisions.

(4) Arbitrage decision and hedging strategies under market uncertainties as well as robust control applications.
Modeling with transaction costs. The most popular models at present involving Black-Scholes do not take this into account.

1.1 Problem Statement

This thesis deals with following two problems.

Problem #1 Time series optimal prediction filter. Development of Dynamic modeling and Forecasting Algorithm (DMFA).


Algorithms are developed for the design and implementation of an optimal prediction filter. Input to the filter is a financial time series data and the output is an estimated future value of the given input series. The filter works by creating an optimal map by first predicting in-sample values and uses this map to predict out-of-sample data points. This optimal mapping is recursively and computationally efficient. The method builds on the AR model for superior results. The algorithm presented in this thesis will be referred to as Dynamic Modeling and Forecasting Algorithm (DMFA) and assures the system stability while presenting a new approach based on datasets.

Problem #2 Computationally stable solutions of financial options via orthogonal polynomials involving Black-Scholes equations.
In this section we derive Black-Scholes equations from a probability point of view and present new numerical results which yield stability. Pricing and Hedging of Derivative Securities by L.T. Nielson [Nielson, L.T.], Oxford University Press discusses theory of pricing and hedging derivatives and discusses stochastic processes thoroughly.

Control Systems permeate every facet of our life. On the other hand, corner stone of financial hedging is the Black-Scholes financial model. So it is natural to apply the theoretical techniques developed in the Control Systems studies to the financial systems in a realistic environment. This thesis presents different ways of studying and solving Black-Scholes equation to evolve trading strategies, minimizing the risk liabilities under the volatile market conditions. Since the market parameters are uncertain and time varying, a fixed model will not do the job. In this document, the online parameter determination is incorporated into the modeling. Our control strategy model is driven by updated volatility and stock market price and the output of this model is a risk-free fair price option as well as its sensitivity to the underlying stock price fluctuations. New contribution to stable numerical algorithms has also been presented. The proposed method shows how the orthogonal polynomials, known as Kautz polynomials can be used for the solution of Black-Scholes type equations for European style options, significantly reducing the computational burden and simultaneously increasing the stability of the computational algorithm. These ideas can be extended to include the American style options.

1.2 Practical Implications for Financial Instruments

Research Ideas Developed in this Thesis

In an oft quoted paper published in 1973, Fisher Black and Myron Scholes [Fisher, B.] derived an equation for Risk-fair pricing of European Calls and Puts for non-dividend, non Transactional cost stocks. This has been a role model for financial transactions giving rise to a multi trillion dollars of derivatives.

Every financial transaction requires due diligence, decision making and monitoring strategies. It is important that analytic tools be available for credit-risk management involving credit portfolio and risk transfer. Active management can diversify credit risk across many securities
and entities. This requires structured finance activity such as Puts and Calls as well as credit
derivatives. The first part of this thesis derives optimal and adaptive models from streaming
data. In the second part of this thesis we develop mathematical tools needed for risk-transfer
via credit derivatives thereby shifting around the risks embedded in various financial transac-
tions. These mathematical tools have created a new industry known as “Derivative Industry” or
“Hedge-Fund Industry” or so called “Risk-Management Industry”.

If one follows the mathematics involved in the derivation of the Black-Scholes risk-free
equation, it becomes clear that to hedge volatility, the portfolio is \textit{continuously variable}.
\textbf{Caution about erroneous assumption, spurious effects and remedies.}

(1) Non realistic assumptions may yield erroneous results. Hence it is necessary to treat the
results obtained via traditional Black-Scholes equation as only a first cut basis for hedging
strategies. Inclusion of non-linearity in the volatility term as well as the transaction costs
may make the portfolio adjustments expensive. \textbf{The new computational method that we
propose will make this rebalancing of the portfolio very affordable.}

(2) Introduction of the adaptive techniques for the on-line updating and identification of
parameters such as interest rate, volatility and the so called “drift”.

(3) Effect of the non-linear dynamics (instead of being linear) on the portfolio.

(4) Inclusion of the adverse changes due to the market-risks and the credits-risks.

(5) Development of Stochastic models for equities, involving realistic assumptions. This in-
volves control strategies for the derivatives of the securities (puts and calls) to hedge the
financial risk undertaken by the investors. These strategies will allow a trader or an investor
to compute risk-free and profitable prices for buying or selling the derivatives.

(6) Comparison of the actual market observations with the market model and design of pa-
rameters observer so that the discrepancies between the two are mitigated via the observer
theory so popular in the Control Systems.
Chapter 2
Dynamic Modeling and Forecast Algorithm

2.1 Introduction

Mathematical methods developed in last 20 years have played an important role in the study of economics and financial markets. This is possible because of the computing resources being available at will. But even the powerful computational facilities have limitations when the amount of data is astronomical and the system dynamics are changing. This thesis presents a dynamic modeling and forecasting algorithm which uses innovative ideas of recursion techniques. Many of these ideas were developed in the hay days of Aerospace advances. In the existing methods in the literature, the model parameters are determined from the data and then passively used to predict the future portfolio values. The motivation behind the presented method has the advantage of dynamically updating the parameter values utilizing the changing data and thereby evolving an optimally updated model which takes into account the structural changes that may have developed in the actual process. Hence, this yields a superior capability. Thus the model is updated in a closed-loop fashion.

2.2 Literature

Box and Jenkins [Box G.] popularized a three-stage method aimed at selecting an appropriate ARIMA(p,q) model for the purpose of estimating and forecasting a time series. This was characterized by model identification, i.e. deciding on the order of p,q, estimation, i.e. fitting of the parameters in the ARIMA model and goodness of the fit. This method is widely used across many applied fields. It is used extensively in time series modeling and model selection capabilities for linear regression models with ARIMA by the Census Bureau [Census].
The algorithm proposed below has many applications across diverse fields such as Aerospace, Seismography, Stock-Bond portfolio valuations. In Aerospace Engineering, the model can predict and make correction for the parameters. In Seismography, where earth-quake predictions are immensely useful the tool can be a powerful predictor of impending disasters. For stock and bond portfolio, the projections of vectors (such as interest rates, home price index appreciation, forex-rates, swaps) from the algorithm can be used to run valuations on the portfolio. This may be used by a portfolio manager to buy/sell individual entities to keep a well-balanced and fund-centric portfolio. Furthermore, valuations may also be used for auditing purposes such as mark-to-market vs mark-to-model. The method predicts the next data point while dynamically computing the parameters from existing data. The computation of the parameters is optimized by use of the recursive matrix inversion algorithm. The updated parameters are used as the updated model.

2.3 Preliminaries

This dynamic modeling and forecast algorithm will be referred to as DMFA. Original financial data streams are \( \{d_k(t)\}_{t=1}^{\infty}, k = 1, \cdots, N \), being fed into the \( m \) parameter dynamic modeling filter. These data stream are used to form data sets;

\[
\{d_k(\tau, i)\}_{i=1}^{m+n}, \quad \tau = 0, 1, 2, \cdots, \quad k = 1, \cdots, N \quad \text{such that}
\]

\[N = \# \text{ of financial entities in a given portfolio}\]

\[m = \# \text{ of dynamic model parameters}\]

\[m + n = \# \text{ of points in each set required by modeling algorithm}\]

\[d_k(\tau, i) = d_k(\tau + i)\]

\[d_k(\tau + 1, i) = d_k(\tau, i + 1) = d_k(\tau + i + 1)\]

Fig. 2.1 shows how the data stream \( \{d_k(t)\}_{t=1}^{\infty} \) yields data sets \( \{d_k(\tau, i)\}_{i=1}^{m+n} \).

Thus the new data sets are created from previous data set by dropping its starting data point and
$\{d_k(t)\}_{t=1}^{\infty}$, $t = 1, 2, \ldots, k = 1, \ldots, N$

$k=1,2,\ldots,N$, where $N$ is the total number of entities in the portfolio

Figure 2.1: Data set $\{d_k(\tau,i)\}_{i=1}^{n+m}$ formation from data $\{d_k(t)\}_{t=1}^{\infty}$, $t = 1, 2, \ldots, k = 1, \ldots, N$
adding the next data point. Suffice to say that we are given a continous stream of data sets 
\( \{ d_k(\tau, i) \}_{i=1}^{n+m} \). Using the data sets \( \{ d_k(\tau, i) \}_{i=1}^{n+m} \), our task is to develop:

a) The dynamic model represented by \( m \) parameters (\( m \) is variable) such that using data \( \{ d_k(\tau, i) \}_{i=1}^{n+m} \) results in a least mean square error dynamic model.

b) Use the dynamic model to forecast future securities values resulting from such a model. The model parameters are updated when new data point arrives.

**Data Normalization:** Data is normalized as following:

Let

\[
m_k(\tau) = \left( \frac{1}{n + m} \right) \sum_{i=1}^{n+m} d_k(\tau, i) \quad \text{mean of the dataset at } \tau
\]

\[
x_k(\tau, i) = \frac{\left( d_k(\tau, i) - m_k(\tau) \right)}{|d_k(\tau, i)|_{\text{max}}} \quad \% \text{ deviation from the mean}
\]

So that, \( |d_k(\tau, i)|_{\text{max}} \) is the maximum absolute value in the dataset at \( \tau \). The normalized data can be converted to actual data as:

\[
d_k(\tau, i) = m_k(\tau) + |d_k(\tau, i)|_{\text{max}} x_k(\tau, i)
\]

\( x_k(\tau, i) \) will be used to represent normalized data throughout the remaining sections.

**Note:** All \( d_k(\tau, i) \) are assumed to be positive prices.

### 2.4 Dynamic Model Parameter Estimation Algorithm

System model is defined as:

\[
x_k(\tau, m + 1) = a_k(\tau, 1)x_k(\tau, 1) + a_k(\tau, 2)x_k(\tau, 2) + \cdots + a_k(\tau, m)x_k(\tau, m)
\]

\[
x_k(\tau, m + 2) = a_k(\tau, 1)x_k(\tau, 2) + a_k(\tau, 2)x_k(\tau, 3) + \cdots + a_k(\tau, m)x_k(\tau, m + 1)
\]

\[ \vdots \]

\[
x_k(\tau, m + n) = a_k(\tau, 1)x_k(\tau, n) + a_k(\tau, 2)x_k(\tau, n + 1) + \cdots + a_k(\tau, m)x_k(\tau, m + n - 1)
\]

The variables \( \{ x_k(\tau, i) \}_{i=m+1}^{m+n} \) represent the model outputs.

This yields \( n \) equations with \( (m + n) \) observations.
Let
\[ \mathbf{x}_k(\tau, i) = \begin{bmatrix} x_k(\tau, i) \\ x_k(\tau, i + 1) \\ \vdots \\ x_k(\tau, i + m - 1) \end{bmatrix}, \quad i = 1, \ldots, n, \quad y_k(\tau) = \begin{bmatrix} x_k(\tau, m + 1) \\ \vdots \\ x_k(\tau, m + n) \end{bmatrix} \] (2.1)

yielding,
\[ y_k(\tau) = \mathbf{M}_k(\tau) \mathbf{a}_k(\tau) \] (2.2)

where
\[ \mathbf{a}_k(\tau) = \begin{bmatrix} a_k(\tau, 1) \\ a_k(\tau, 2) \\ \vdots \\ a_k(\tau, m) \end{bmatrix}, \quad \mathbf{M}_k(\tau) = \begin{bmatrix} x_k(\tau, 1) & x_k(\tau, 2) & \cdots & x_k(\tau, m) \\ x_k(\tau, 2) & x_k(\tau, 3) & \cdots & x_k(\tau, m + 1) \\ \vdots & \vdots & & \vdots \\ x_k(\tau, n) & x_k(\tau, n + 1) & \cdots & x_k(\tau, n + m - 1) \end{bmatrix} \]

The vector \( \mathbf{a}_k(\tau) \) represents the \( m \) parameters to be determined. The vector \( \mathbf{a}_k(\tau) \) is chosen as a minimum mean square solution (see Appendix B) yielding the estimated parameter vector,
\[ \hat{\mathbf{a}}_k(\tau) = \mathbf{B}_k^{-1}(\tau) \mathbf{z}_k(\tau) \] (2.3)

\[ \left[ \mathbf{M}_k^T(\tau) \mathbf{M}_k(\tau) \right] = \mathbf{B}_k(\tau), \quad \mathbf{z}_k(\tau) = \mathbf{M}_k^T(\tau) y_k(\tau) \]

\( \tau = 0, 1, 2, \ldots, \quad k = 1, 2, \ldots, N, \quad \) optimal parameter estimate

Eq. 2.3 is simple enough except for the fact that inverting a large matrix involves large number of floating point operations per second (FLOPS) along with the inaccuracies as a result of data representation and storage, particularly in this case where the inversion has to take place dynamically at every time \( \tau \). We propose solving Eq. 2.3 in such a way that we never have to invert more than a \( 2 \times 2 \) matrix and the calculations are performed recursively.
2.5 Recursion Algorithm for solution of Eq. 2.3

2.5.1 Recursive Inversion of the matrix \( B_k(\tau) \)

For convenience, we shall drop the subscript \( k \) assuming the same algorithm will be used for all portfolio entities. Hence \( B_k(\tau) \) will be treated as \( B(\tau) \).

In general, computation of \( B^{-1}(\tau) \) (for a large matrix) requires prohibitive computational time and storage. Here we propose a recursive inversion method which does not explicitly require inverting the matrix \( B(\tau) \). (Detailed proof is given in Appendix C).

Let us consider \( l \times l \) submatrices \( B_l(\tau) \) of the matrix \( B(\tau) \) involving \( l \) rows and \( l \) columns. When \( l = m \), we obtain \( B(\tau) \), namely \( B_m(\tau) = B(\tau) \), where

\[
B_l(\tau) = \begin{bmatrix}
b_{11}(\tau) & \cdots & b_{1l}(\tau) \\
\vdots & & \vdots \\
b_{1l}(\tau) & \cdots & b_{ll}(\tau)
\end{bmatrix}_{l \times l}, \quad B_{l+1}(\tau) = \begin{bmatrix}
b_{11}(\tau) & \cdots & b_{1,l+1}(\tau) \\
\vdots & & \vdots \\
b_{1,l+1}(\tau) & \cdots & b_{l,l+1}(\tau)
\end{bmatrix}_{(l+1) \times (l+1)}
\]

(Subscript \( l \), represents the \( l \)-dimensional sub-matrix of \( B_k(\tau) \) and not the portfolio entity index)

Assuming \( B_l^{-1}(\tau) \) is known, we are required to compute \( B_{l+1}^{-1}(\tau) \) without inversion. Of course we shall make use of \( B_l^{-1}(\tau) \), let

\[
B_{l+1}(\tau) = \begin{bmatrix}
B_l(\tau) & c_{l+1}(\tau) \\
c_{l+1}^T(\tau) & b_{l+1,l+1}(\tau)
\end{bmatrix}, \quad c_{l+1}(\tau) = \begin{bmatrix}
b_{1,l+1}(\tau) \\
\vdots \\
b_{l,l+1}(\tau)
\end{bmatrix}
\]

Following the details in Appendix C,

\[
B_{l+1}^{-1}(\tau) = \begin{bmatrix}
B_l^{-1}(\tau) + f_{l+1}(\tau) g_{l+1}(\tau) g_{l+1}^T(\tau) & -f_{l+1}(\tau) g_{l+1}(\tau) \\
-f_{l+1}(\tau) g_{l+1}^T(\tau) & f_{l+1}(\tau)
\end{bmatrix}
\]

\[
f_{l+1}(\tau) = \left( b_{l+1,l+1}(\tau) - c_{l+1}^T(\tau) g_{l+1}(\tau) \right)^{-1}
\]

\[
g_{l+1}(\tau) = B_l^{-1} c_{l+1}(\tau)
\]
[\[B_i^{-1}(\tau)\]]_{i=m} = B^{-1}(\tau)

### 2.5.2 Incremental Inversion Algorithm for Solution of Eq. 2.3

Let

\[
B_k(\tau) = \begin{bmatrix} x_k(\tau, 1) & x_k(\tau, 2) & \cdots & x_k(\tau, n) \end{bmatrix}
\]

\[
\begin{bmatrix} x_k^T(\tau, 1) \\ x_k^T(\tau, 2) \\ \vdots \\ x_k^T(\tau, n) \end{bmatrix}
\]

Furthermore,

\[
B_k(\tau + 1) = \begin{bmatrix} x_k(\tau, 2) & x_k(\tau, 3) & \cdots & x_k(\tau, n + 1) \end{bmatrix}
\]

\[
\begin{bmatrix} x_k^T(\tau, 2) \\ x_k^T(\tau, 3) \\ \vdots \\ x_k^T(\tau, n + 1) \end{bmatrix}
\]

This can be written as an incremental equation

\[
B_k(\tau + 1) = \left( B_k(\tau) + \Delta_k(\tau) \right)
\]
For $n = 3, m = 2$

$$B_k(\tau) = \begin{bmatrix} x_k(\tau, 1) & x_k(\tau, 2) & x_k(\tau, 3) \\ x_k(\tau, 2) & x_k(\tau, 3) & x_k(\tau, 4) \\ x_k(\tau, 3) & x_k(\tau, 4) & x_k(\tau, 5) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^{3} x_k^2(\tau, i) & \sum_{i=1}^{3} x_k(\tau, i)x_k(\tau, i + 1) \\ \sum_{i=1}^{3} x_k(\tau, i)x_k(\tau, i + 1) & \sum_{i=1}^{3} x_k^2(\tau, i + 1) \end{bmatrix}$$

$$\Delta_k(\tau) = \begin{bmatrix} x_k^2(\tau, 4) - x_k^2(\tau, 1) & x_k(\tau, 4)x_k(\tau, 5) - x_k(\tau, 1)x_k(\tau, 2) \\ x_k(\tau, 4)x_k(\tau, 5) - x_k(\tau, 1)x_k(\tau, 2) & x_k^2(\tau, 5) - x_k^2(\tau, 2) \end{bmatrix}$$

$\Delta_k(\tau)$ is perturbation of the matrix $B_k(\tau)$ due to new data and its determinant is much smaller than the determinant of $B_k(\tau)$.

Once $B_k^{-1}(\tau)$ has been computed, an incremental algorithm can be used to compute $B_k^{-1}(\tau + 1)$, thereby reducing the computational complexity. The incremental algorithm can be stated as:

$$B_k^{-1}(\tau + 1) = B_k^{-1}(\tau) - B_k^{-1}(\tau)\Delta_k(\tau)B_k^{-1}(\tau)$$  \hspace{1cm} (2.5)

$$\frac{|\Delta_k(\tau)|}{|B_k(\tau)|} << 1, \quad (| | \text{ representing determinants})$$

[For detailed derivation see Appendix D]

### 2.6 Prediction Algorithm

Utilizing the normalized dataset $\{x_k(\tau, i)\}_{i=1}^{m+n}$, the updated parameters $a_k(\tau, 1), a_k(\tau, 2), \cdots, a_k(\tau, m)$ are computed from the algorithm in section 4 involving computation of $a_k(\tau)$. The last available data before new prediction is designated as $x_k(\tau, m + n)$ as shown in Fig. 2.2

The task at hand is the prediction of the next value of $x_k(\tau + 1, n + m) = x_k(\tau, n + m + 1)$ and is designated as $\hat{x}_k(\tau + 1, n + m) = \hat{x}_k(\tau, n + m + 1)$, after having computed $\hat{a}_k(\tau)$. 
The Prediction equation can be written as:

\[
\hat{x}_k(\tau + 1, m + n) = \hat{x}_k(\tau, m + n + 1) \\
= \hat{a}_k(\tau, 1)x_k(\tau, n + 1) + \hat{a}_k(\tau, 2)x_k(\tau, n + 2) + \cdots + \hat{a}_k(\tau, m)x_k(\tau, m + n)
\]

or, from Eq. 2.1

\[
\hat{x}_k(\tau + 1, m + n) = \hat{x}_k(\tau, m + n + 1) = x_k^T(\tau, i)\hat{a}_k(\tau),
\]

where \( i = n + 1, \ldots, m + n \) (2.6)

2.7 Parameter Updating

At the instant \((\tau + 1)\), the normalized dataset \( \{x_k(\tau + 1, i)\}_{i=1}^{n+m} \) is available to update the parameter vector \( \hat{a}_k(\tau + 1) \).

\[
\hat{a}_k(\tau + 1) = B_k^{-1}(\tau + 1)M_k^T(\tau + 1)y_k(\tau + 1)
\]

(2.7)

The updated parameters are recursively used in the next computation for optimal prediction.

2.8 Portfolio Optimization

In this section we propose an optimization method for portfolio consisting of different entities whose structure is modeled and future values can be forecasted.
Let

\[ y_d(t) = y_d(\tau, i) = \text{Desired portfolio value at } t = \tau + i \]

\[ \hat{x}_k(\tau) = \hat{x}_k(\tau, i) = \text{Forecasted price of entities comprising the portfolio, } k = 1, \cdots, N \]

\[ w_k(\tau) = \text{Weighting of the entities } x_k(\tau, i), \tau = 1, 2, \cdots \]

\[ \hat{I}_o(\tau + 1) = \text{optimal estimate of portfolio at } \tau + 1 \]

The desired portfolio value can be formulated as:

\[ y_d(t) = y_d(\tau, i) = I_o e^{\alpha (\tau + i)}, \text{ where } \alpha = \text{profitability factor} > 1, i = 1, 2, \cdots, m + n \]

\[ I_o = \sum_{k=1}^{N} w_k(0) x_k(0, 0), \text{ where } w_k(0) \text{ are chosen apriori} \]

Given: \( y_d(\tau, i), \hat{x}_k(\tau, i), i = 1, \cdots, n + m, \quad k = 1, \cdots, N, \quad \tau = 0, 1, \cdots \)

Algorithm:

\[ y_d(\tau, i) = \hat{x}_1(\tau, i) w_1(\tau) + \hat{x}_2(\tau, i) w_2(\tau) + \cdots + \hat{x}_N(\tau, i) w_n(\tau) \]

Let

\[
\begin{bmatrix}
    y_d(\tau, 1) \\
    y_d(\tau, 2) \\
    \vdots \\
    y_d(\tau, n + m)
\end{bmatrix}, \quad
\begin{bmatrix}
    \hat{x}_1(\tau, 1) & \hat{x}_2(\tau, 1) & \cdots & \hat{x}_N(\tau, 1) \\
    \hat{x}_1(\tau, 2) & \hat{x}_2(\tau, 2) & \cdots & \hat{x}_N(\tau, 2) \\
    \vdots & \vdots & \ddots & \vdots \\
    \hat{x}_1(\tau, n + m) & \hat{x}_2(\tau, n + m) & \cdots & \hat{x}_N(\tau, n + m)
\end{bmatrix} = \begin{bmatrix}
    \hat{x}_1^T(\tau, 1) \\
    \hat{x}_1^T(\tau, 2) \\
    \vdots \\
    \hat{x}_1^T(\tau, n + m)
\end{bmatrix},
\]

\[
\begin{bmatrix}
    w_1(\tau) \\
    w_2(\tau) \\
    \vdots \\
    w_N(\tau)
\end{bmatrix}
\]

Thus

\[ y_d(\tau) = \hat{X}(\tau) w(\tau), \quad \tau = 1, 2, \cdots \]
The optimal weights $\hat{w}(\tau), \tau = 1, 2, \cdots$ are computed as:

$$\hat{w}(\tau) = \left[ \hat{X}^T(\tau)\hat{X}(\tau) \right]^{-1} \hat{X}^T(\tau)y_d(\tau)$$

The following diagram explains the detailed estimation and optimization of a given portfolio involving entities represented by $x_k(t) = x_k(\tau + i) = x_k(\tau, i), \tau = 0, 1, \cdots, i = 1, 2, \cdots, k = 1, \cdots, N$.

![Diagram](image)

**Figure 2.3: Portfolio forecast and optimization**

### 2.9 Conclusion

The DMFA method is a superior auto regression (AR) model as a general system of time-series realizations in-order to calculate the co-efficients that fit the model for a better prediction. The system is solved via an innovative technique of inversion such that it avoids explicit inversion of more than a $2 \times 2$ matrix and computation of higher dimensional determinants and co-factors. Furthermore, large number of parameters can be updated and the model re-fitted to reduce prediction errors. The model can be further extended to solve for a financial portfolio involving basket of securities. The minimum mean-square algorithm used assures system stability via poles within the unit circle.

### 2.10 Future Research Directions for Modeling and Forecasting

In this thesis, we have used the minimum mean square algorithm to predict, update and optimize the portfolio. We suggest that other optimization algorithms such as linear programming, non-linear programming and dynamic programming should be further explored to arrive at other
algorithms for optimal forecast.

2.11 DMFA Performance

In the over-determined system, increasing \( n \) increases estimation errors for the same number of parameters \( m \). Furthermore, increasing the number of parameters \( m \), confuses a system as information is inferred from a larger number of past states, thereby leading to inaccurate tracking. An optimal \( m, n \) for a given input series may vary for a different input data time series. Figures 2.4, 2.5, 2.6 and 2.7 present the comparison between actual and forecasted values for different \( m \) and \( n \). Parallel computing is a powerful tool when the number of securities is large.
Figure 2.4: Portfolio estimation algorithm, $m = 3$ parameters, $n = 20$ initial data

Figure 2.5: Portfolio estimation algorithm, $m = 3$ parameters, $n = 30$ initial data
Figure 2.6: Portfolio estimation algorithm, $m = 4$ parameters, $n = 35$ initial data

Figure 2.7: Portfolio estimation algorithm, $m = 5$ parameters, $n = 40$ initial data
Chapter 3

Stable Numerical Algorithms for Black-Scholes Financial Models

3.1 State of the Art

In this section of the thesis, we present the state of the art as well as some research ideas and new computational results for the financial instruments (derivatives). The starting point for us is the famous Black-Scholes equation for a fair price option modelling. This involves treating markets as continuous (or discrete) stochastic entities driven by a Wiener process (Brownian Motion). Modern economic forces make it imperative that commercial and household transactions select an appropriate, optimal level of risk. Due to their intrinsic structure, options as designed via Merton-Black-Scholes [Fisher, B.] equations allow for the efficient risk managed options (based upon existing assumptions). The resulting mathematical equation is relatively simple but its practical implications, when properly used are of immense importance. Properly selected numerical algorithms are essential for the satisfactory solutions of this Black-Scholes equation. The resulting Black-Scholes (B-S) formula has been used by thousands of investors all over the world.

Black-Scholes equation [Fisher, B.] is a linear parabolic partial differential equation with terminal boundary conditions or artificial boundary conditions. The numerical methods used for computing the solution of B-S equation are sensitive to the parameters and furthermore have their own inherent stability problems. Chawla et al [Chawla, M.] have proposed a time integration scheme based upon generalized trapezoidal splines. Their claim is that the resultant solution is unconditioned stable and superior to the Crank-Nicholson [Crank, J.] scheme discussed in the literature.

Furthermore, most of the existing efforts assume a linear fair price options model. This is an inherent flaw for the successful application of the above methods. In the proposed extension of the Black-Scholes equation we consider an extension to include non-linearity and time varying
parameters in the market models resulting in an adaptive risk management strategies. In a nut shell, for the market model to be realistic, it should consider parameter changes as well as nonlinearity. We extend the model to be adaptive along with a polynomial type nonlinearity and new computing schemes involving orthogonal functions.

3.2 Specific Tasks

- Development and analysis of orthogonal functions such as Kautz polynomials for linear and nonlinear option models and resulting change of time variables in order to assure numerical stability. Kautz polynomials are inherently more stable than Trapezoidal formulas or finite differences. The numerical scheme will easily handle all types of Boundary conditions

- Since the Black-Scholes equation is linear parabolic diffusion type partial differential equation, we shall develop analytical techniques based upon separation of variables to solve this type of equation. This is facilitated by change of variables.

- Introduction of time variation into the volatility parameter and the interest rate parameter so as to be able to formulate adaptive risk free strategies. Furthermore, we shall include the cost of transaction, tax implications, effect of the dividends as well as price jump phenomenon to make the results more realistic.

- We propose nonlinear optimization methods for parameter identification, and hedging.

3.3 Financial Markets: Risk Free Options (Puts and Calls)

Option is a right to buy or sell an asset at some future time. The price allocated to this option depends upon the future price behaviour of the assets. There are five fundamental relations required to provide a mathematical structure to this task resulting in the celebrated Black-Scholes equation.

1. Market process describing the asset price evolution

2. Stock price determination via Ito’s Lemma.
3. Option Price change law via Ito’s lemma.

4. Value of the portfolio as a function of the stock price and the option price

5. Risk Free Portfolio Algorithm (Black-Scholes Model)

Assumptions and Limitations of Black-Scholes model.

(1) Price dynamics follows Wiener process with fixed drift and volatility. We can rectify this limitation by adaptively determining both.

(2) There is transaction cost. We can take this into account in the price dynamics which may result in non-linearities.

(3) Stocks do not pay dividend. We can modify the dynamic model to rectify the deficiency.

(4) European option terms are used. We can modify this to include the American options.

(5) Risk-Free interest rates remain constant. This is a major limitation, in reality not so and hedging strategies such as interest rate swap can be developed to mitigate this limitation.

(6) Model assumes continuous price function. In reality there are abrupt changes in the price which may require modification of the hedging strategies. This may require resetting the modeling initiation.

(i) **Market Process Dynamics.**

Stock or “Security” price is given by an a-priori law governed by a stochastic differential equation:

\[ dy(t) = \mu y(t)dt + \sigma y(t)dW(t) \]  
\[(3.1)\]
where

\[ W(t) = \text{Wiener process noise with zero mean and } \sqrt{t} \text{ variance.} \]

\[ y(t) = \text{unit price of Stock or “security” at the time } t \]

(a unit comprises of 100 shares as is customary for trading the derivatives.)

\[ t = \text{time variable } 0 \leq t \leq T < \infty \]

\[ \mu = \text{“Drift” of the stock price per unit time (percentage)} \]

\[ \sigma = \text{“Volatility” of the stock due to random fluctuations driven by the incremental Wiener process } dW(t). \]

\[ E [dW(t)] = 0 \]

\[ E \left[ (dW(t))^2 \right] = dt \]

In the state-of-the-art models, both \( \mu \) and \( \sigma \) are treated as constants. \textbf{In our contributions involving this thesis, we shall treat them as time dependant parameters. This is one of our new contributions to this thesis resulting in an adaptive financial model.}

(ii) \textbf{Derivation of Stock Price Formula, driven by Brownian motion, } W(t)

Let \( g(y(t), t) \) be a smooth stock option function depending upon the stock price \( y(t) \) at the time \( t \). Since \( y(t) \) represents stochastic process, its incremental change is governed by the Ito’s process, yielding an extra \( \frac{\partial^2 g}{\partial y^2} \) term.

From Taylor series:

\[ dg = g(y(t) + dy, t + dt) - g(y(t), t) \]

\[ = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial y} dy + \frac{1}{2} \left[ \frac{\partial^2 g}{\partial t^2} (dt)^2 + \frac{\partial^2 g}{\partial y^2} (dy)^2 + 2 \frac{\partial^2 g}{\partial y \partial t} (dy)(dt) \right] + \text{Higher order terms.} \]

Since \( (dt) \) is a real variable

\[ (dt)^2 \to 0 \quad \text{(second order terms in real variables)} \]

\[ (dy \, dt) \to 0 \]

\[ (dy)^2 \to 0 \]
But

\[(dy)^2 \neq 0 \]  (Being Stochastic has a variance.)

Thus neglecting higher than the second order terms and applying the above simplification:

\[
dg = \frac{\partial g}{\partial t}dt + \frac{\partial g}{\partial y}dy + \frac{1}{2} \frac{\partial^2 g}{\partial y^2} (dy)^2 \]

\[(3.2)\]

From Eq. 3.1,

\[
dy = \mu y(t)dt + \sigma y(t)dW(t)\]

\[(dy)^2 = (\mu)^2 (y(t))^2 (dt)^2 + (\sigma)^2 (y(t))^2 (dW(t))^2 + 2\mu \sigma (y(t))^2 dW(t)dt\]

via Ito’s Calculus

\[(dt)^2 \rightarrow 0\]

\[(dW(t))dt \rightarrow 0\]

\[(dW(t))^2 \rightarrow (\sqrt{dt})^2 = dt\]

yielding

\[(dy)^2 = \sigma^2 y^2(t)dt\]

\[(3.3)\]

Substituting the above expressions for dy and (dy)^2 in Eq. 3.2,

\[
dg = \frac{\partial g}{\partial t}dt + \frac{\partial g}{\partial y}dy + \frac{1}{2} \frac{\partial^2 g}{\partial y^2} (dy)^2\]

or

\[
dg = \frac{\partial g}{\partial t}dt + \frac{\partial g}{\partial y} (\mu ydt + \sigma ydW(t)) + \frac{1}{2} \frac{\partial^2 g}{\partial y^2} (\sigma^2 y^2 dt)\]

yielding

\[dg = Agdt + Bg dW(t)\]

\[(3.4)\]
where

\[ A_g = \frac{\partial g}{\partial t} + \mu \frac{\partial g}{\partial y} + \frac{1}{2} \sigma^2 y^2 \frac{\partial^2 g}{\partial y^2} \]
\[ B_g = \sigma y \frac{\partial g}{\partial y}, \quad y = y(t) \]

We are now in a position to derive the expected value of \( y(t) \), defined as \( \mathbb{E}[y(t)] \).

(In the later section, we shall derive the expected value of the options or the derivatives.)

Consider the market dynamic equation Eq. 3.1

\[ dy(t) = \mu y(t) dt + \sigma y(t) dW(t) \]

or

\[ \frac{dy(t)}{y(t)} = \mu \, dt + \sigma \, dW(t) \]

integrating both sides,

\[ \ln y(t) \big|_{y(0)}^{y(t)} = \mu t + \sigma W(t) \]

yielding

\[ y(t) = y(0) e^{\mu t + \sigma W(t)} = y(0) e^{\mu t} e^{\sigma W(t)} \quad \text{(3.5)} \]

Taking estimates of \( y(t) \) yields,

\[ \mathbb{E}[y(t)] = y(0) e^{\mu t} \mathbb{E} \left[ e^{\sigma W(t)} \right] \quad \text{(3.6)} \]

We shall prove that

\[ \mathbb{E} \left[ e^{\sigma W(t)} \right] = e^{\frac{\sigma^2}{2}t} \]

yielding

\[ \mathbb{E}[y(t)] = y(0) e^{\mu t} e^{\frac{\sigma^2}{2}t} = y(0) e^{(\mu + \frac{\sigma^2}{2})t} \]

Proof of \( \mathbb{E} \left[ e^{\sigma W(t)} \right] = e^{\frac{\sigma^2}{2}t} \)

Let

\[ Z(t) = \sigma W(t), \quad g(Z(t)) = e^{\sigma W(t)} = e^{Z(t)} \]
$z(t)$ is the variable representation of random function $Z(t)$.

Taking derivatives

$$dZ(t) = \sigma dW(t)$$

From Ito’s Calculus

$$dg(z(t)) = \frac{\partial (g(z(t)))}{\partial t} dt + \frac{\partial g(z(t))}{\partial z(t)} dz + \frac{1}{2} \frac{\partial^2 g(z(t))}{\partial^2 z(t)} (dz)^2 \quad (3.7)$$

But

$$\frac{\partial g(z(t))}{\partial z(t)} = e^{z(t)} = g(z(t))$$
$$\frac{\partial^2 g(z(t))}{\partial^2 z(t)} = e^{z(t)} = g(z(t))$$
$$\frac{\partial g(z(t))}{\partial t} = 0 \quad (3.8)$$

Hence Eqs. 3.7 and 3.8 yield

$$dg(z(t)) = g(z(t))\sigma dW(t) + g(z(t))\left(\frac{1}{2} \sigma^2 (dW(t))^2\right)$$

or

$$\frac{dg(z(t))}{g(z(t))} = \sigma dW(t) + \frac{1}{2} \sigma^2 (dW(t))^2$$

Taking estimate of both sides,

$$E \left[ \frac{dg(z(t))}{g(z(t))} \right] = E \left[ \sigma dW(t) + \frac{1}{2} \sigma^2 (dW(t))^2 \right] = 0 + \frac{1}{2} \sigma^2 dt$$

Integrating

$$E \left[ \ln g(z(t)) \right] = E \left[ \ln e^{\sigma W(t)} \right] = \frac{1}{2} \sigma^2 t$$

or

$$E \left[ g(z(t)) \right] = E \left[ e^{\sigma W(t)} \right] = e^{(\sigma^2/2)t}$$

yielding

$$E \left[ y(t) \right] = y(0)e^{(\mu + \sigma^2/2)t} \quad (3.9)$$

“$E$” stands for the expected value.
3.4 Value of Risk-Free Security

In a risk-free environment, securities should increase according to the prevailing interest rate. If the interest rate is \( r \), then

\[
E[y(t)] = y(0)e^{rt} \quad \cdots \text{ Risk Free Environment.} \quad (3.10)
\]

From Eq. 3.9 and Eq. 3.10

\[
e^{rt} = e^{(\mu + \sigma^2/2)t}
\]

yielding

\[
r = \mu + \frac{\sigma^2}{2} \quad \text{Risk Free Interest Rate.} \quad (3.11)
\]

Conclusion:

Risk-Free interest rate is higher than the stock growth rate due to volatility

3.5 Risk-Free, Fair-Price Options Dynamics

(i) Option Dynamics of the Deterministic Market \((\sigma = 0)\)

Let

\[
\hat{F}(y(t), t) = \text{Risk Free options for a deterministic market.}
\]

The differential using Taylor series expansion is:

\[
\begin{align*}
\text{d}\hat{F} &= \hat{F}(y(t) + dy, t + dt) - \hat{F}(y(t), t) \\
&= \frac{\partial \hat{F}}{\partial t} dt + \frac{\partial \hat{F}}{\partial y} dy + \frac{1}{2} \left[ \frac{\partial^2 \hat{F}}{\partial t^2} (dt)^2 + \frac{\partial^2 \hat{F}}{\partial y^2} (dy)^2 + 2 \frac{\partial^2 \hat{F}}{\partial y \partial t} (dy)(dt) \right] \\
&\quad + \text{Higher order terms.}
\end{align*}
\]

Since all variables are deterministic, we neglect the second and higher order terms, yielding:

\[
\begin{align*}
\text{d}\hat{F} &= \frac{\partial \hat{F}}{\partial t} dt + \frac{\partial \hat{F}}{\partial y} dy \\
\text{Deterministic Model of a Put or a Call}
\end{align*}
\]

(3.12)
(ii) **Option Dynamics of a Stochastic Market**

Let, $F(y(t), t)$ be the option price for a non-deterministic case.

$y(t)$ represents stochastic process, its incremental changes is governed by the Ito process, yielding an extra $\frac{\partial^2 F}{\partial y^2}$ term compared with deterministic case.

From Taylor series

$$
\begin{align*}
dF &= F(y(t) + dy, t + dt) - F(y(t), t) \\
&= \frac{\partial F}{\partial t} dt + \frac{\partial F}{\partial y} dy + \frac{1}{2} \left[ \frac{\partial^2 F}{\partial t^2} (dt)^2 + \frac{\partial^2 F}{\partial y^2} (dy)^2 + 2 \frac{\partial^2 F}{\partial y \partial t} (dy)(dt) \right] \\
&\quad + \text{Higher order terms.}
\end{align*}
$$

Since ($dt$) is a real variable,

$$(dt)^2 \to 0, \quad (dydr) \to 0$$

But

$$(dy)^2 \neq 0 \quad \text{(stochastic)}$$

yielding

$$
\begin{align*}
dF(y(t), t) &= \frac{\partial F}{\partial t} dt + \frac{\partial F}{\partial y} dy + \frac{1}{2} \frac{\partial^2 F}{\partial y^2} (dy)^2 \\
&\quad \text{(3.13)}
\end{align*}
$$

**Stochastic Model of a Put or a Call**

The reader should note the difference between Eqs. 3.12 and 3.13.
3.5.1 Derivation of the Risk Free Fair Price Call Options

1. Deterministic Case: Fair Price Call Option

Let

\[ y(t) = \text{Price of the 1 unit made up of 100 shares at time } t \]

\[ \hat{F}(y(t), t) = \text{Price of the call option at } t, \text{ sold for a period } T \]

at a strike price \( K \) of the stock \( y(t) \). \((-\hat{F}(y(t), t) \text{ is put}).\]

\[ \hat{F}(y(T), T) = 0, \quad T \text{ being the duration of the option.} \]

\[ \mu = \% \text{ growth rate of the stock per unit time (drift)} \]

\[ r = \text{Interest rate charged by the banks} \]

\[ y(0) = \text{Price of the stock at } t = 0 \]

\[ \hat{F}(0) = \text{Fair price of the above option at } t = 0 \]

\[ I(t) = \text{Total investment at time } t. \]

\[ V(t) = y(t) = \text{Value of the portfolio} \]

Now,

\[ I(t) = V(t) - \hat{F}(t) \quad I(0) = V(0) - \hat{F}(0) = \text{Initial Investment} \quad (3.14) \]

(We assume that securities are bought into the portfolio and call options are sold)

\[ \dot{V}(t) = \mu V(t), \quad V(t) = e^{\mu t}V(0) \]

\[ \dot{I}(t) = r I(t), \quad I(t) = e^{rt}I(0) = e^{rt}(V(0) - \hat{F}(0)) \quad (3.15) \]

Let \( K \) be the strike price of the security for which the call option is being sold. It is important for seller of the option to know what should be the fair price of the option at time \( t \). Thus the seller of the option would like to price it at the fair or higher price.

Since

\[ \hat{F}(t) = V(t) - I(t) \]

For a fair value situation

\[ I(0) = Ke^{-rT} \]
or
\[ \hat{F}(0) = V(0) - Ke^{-rT} \]

For a profitable arbitrage
\[ \hat{F}(0) \geq \max \left[ V(0) - Ke^{-rT}, 0 \right] \]

Changing the time origin,
\[ \hat{F}(t) \geq \max \left[ V(t) - Ke^{-r(T-t)}, 0 \right] \]

From Eqs. 3.14 and 3.15

\[ \hat{F}(t) = V(t) - I(t) = e^{\mu t}V(0) - e^{rt}(V(0) - \hat{F}(0)) \]
\[ \hat{F}(T) = 0 = V(0)e^{\mu T} - V(0)e^{rT} + \hat{F}(0)e^{rT} \hspace{1cm} \text{(European options)} \]

or
\[ \hat{F}(0) = V(0) \left( 1 - e^{(\mu - r)T} \right) = \text{Initial fair price of the option.} \]

Furthermore,
\[ \hat{F}(t) = V(t) \left[ 1 - e^{(\mu - r)(T-t)} \right] \]

If \( K \) is the strike price of the stock at \( t = T \), then for the arbitrage to be profitable
\[ I(T) \leq K \]

or
\[ I(0) \leq Ke^{-rT} - \hat{F}(0) \]

or
\[ I(t) \leq Ke^{-r(T-t)} - \hat{F}(t) \]

or
\[
\hat{F}(0) \geq Ke^{-rt} - V(0) \quad \text{European option} \tag{3.16}
\]
\[
\hat{F}(t) \geq Ke^{-(T-t)} - V(t)
\]

For the American option

\[
\hat{F}(T) = e^{rt}V(0) - e^{rT}V(0) + e^{rt}\hat{F}(0) > 0 \quad 0 < t < T
\]

or

\[
\hat{F}(0) \leq V(0)\left(1 - e^{(\mu - r)t}\right)
\]

2. **Stochastic Case: Fair Price Call Option**

This situation yields a partial differential equation discussed below.

**Derivation of Black-Scholes Equations**

(a) **Process Dynamics:**

\[
dy = \mu y(t)dt + \sigma y(t)dW(t) \tag{3.17}
\]

(b) **Stochastic Option Dynamics:**

\[
dF(y(t), t) = \frac{\partial F}{\partial t} dt + \frac{\partial F}{\partial y} dy + \frac{1}{2} \frac{\partial^2 F}{\partial y^2} (dy)^2
\]

\[
(dy)^2 = \mu^2(y(t))^2(dt)^2 + \sigma^2(y(t))^2(dW(t))^2 + 2\mu\sigma(y(t))^2dW(t)dt
\]

Via Ito’s calculus

\[
(dt)^2 \rightarrow 0
\]
\[
(dW(t))dt \rightarrow 0
\]
\[
(dW(t))^2 \rightarrow (\sqrt{dt})^2
\]

yielding

\[
(dy)^2 = \sigma^2 y^2(t) dt
\]

\[
dF(y(t), t) = \frac{\partial F}{\partial t} dt + \frac{\partial F}{\partial y} (\mu y(t)dt + \sigma y(t)dW(t)) + \frac{1}{2} \frac{\partial^2 F}{\partial y^2} \left(\sigma^2 y^2(t)dt\right)
\]
yielding

\[ dF = A_f dt + B_f dW(t) \]  \hspace{1cm} (3.20)

where

\[ A_f = \frac{\partial F}{\partial t} + \mu y(t) \frac{\partial F}{\partial y} + \frac{1}{2} \sigma^2 y^2(t) \frac{\partial^2 F}{\partial y^2} \]

\[ B_f = \sigma y(t) \frac{\partial F}{\partial y} \]

**Portfolio Equation with variable number of shares as “Control”**

If

\[ V(y(t), t) = \text{Value of the portfolio} = n \times y(t) \]

\[ y(t) = \text{Price of the stock (security)} \]

\[ n = \text{Number of shares with price } y(t) = \frac{dV}{dy} \]

\[ F(y(t), t) = \text{Option price} \]

Then, it is obvious that

\[ I(y(t), t) = ny(t) - F(y(t), t) \]  \hspace{1cm} (Investment at time \( t \))  \hspace{1cm} (3.21)

Differentiating Eq. 3.21 yields

\[ dI(y(t), t) = ndy(t) - dF(y(t), t) \]  \hspace{1cm} (3.22)

A risk free portfolio, earning \( r\% \) interest rate yields:

\[ dI(y(t), t) = rI(y(t), t) dt \]  \hspace{1cm} Risk-Free Portfolio Equation.  \hspace{1cm} (3.23)

Comparing Eqs. 3.22 and 3.23, we arrive at the **Fundamental Risk-Free Relation**:

\[ d[ny(t) - F(y(t), t)] - r [ny(t) - F(y(t), t)] dt = 0 \]  \hspace{1cm} (3.24)
Performing the above differentiation yields:

\[ ndy(t) - dF(y(t), t) - r(ny(t) - F(y(t), t)) \, dt = 0 \]

Substituting the expression for \( dy(t) \) and \( dF(y(t), t) \) from Eqs. 3.1 and 3.4, the above equation yields:

\[ n \left[ \mu y(t) dt + \sigma y(t) dW(t) \right] - \left[ A_f dt + B_f dW(t) \right] - r \left[ ny(t) - F(y(t), t) \right] \, dt = 0 \]

Collecting the terms in \( dt \) and \( dW(t) \)

\[ Adt + BdW(t) = 0 \quad (3.25) \]

where

\[ A = \mu ny(t) - A_f - rny(t) + rF(y(t), t) \]
\[ B = \left( \sigma ny(t) - B_f \right) = \sigma y(t) \left( n - \frac{\partial F}{\partial y(t)} \right) \]

Eq. 3.25 results in the famous Black-Scholes equation which for a general \( dt \) and \( dW(t) \) yields:

\[
\begin{align*}
B &= 0 \quad \text{implying} \quad n = \frac{\partial F}{\partial y(t)} \\
\text{and} \quad A &= 0 \quad (3.26)
\end{align*}
\]

when Eqs. 3.26 and 3.27 are combined, we obtain

\[
\mu y(t) \frac{\partial F}{\partial y(t)} - \frac{\partial F}{\partial t} - \mu y(t) \frac{\partial F}{\partial y(t)} - \frac{1}{2} \sigma^2 y^2(t) \frac{\partial^2 F}{\partial y(t)^2} - r y(t) \frac{\partial F}{\partial y(t)} + rF(y(t), t) = 0
\]

Simplifying this final form yields the partial differential equation, which does not include the parameter \( \mu \), is the famous Black-Scholes equation:
Black-Scholes Differential Equation - Risk Free Fair Price Call.

\[ \frac{\partial F}{\partial t} + r y(t) \frac{\partial F}{\partial y(t)} + \frac{1}{2} \sigma^2 y^2(t) \frac{\partial^2 F}{\partial y^2(t)} - r F(y(t), t) = 0 \quad y(t) \geq 0, \quad 0 < t \leq T \ (3.28) \]

with the boundary condition: \( \max(y(T) - K, 0) \)

\( K \) being the strike price.

\( T \) being the option duration.

The equation is valid for a European Call or Put.

**Note:** Risk-Free, fair price European call option, gives the right to buy a share of the security after time \( T \) at a strike price \( K \).

A European call option allows the holder to exercise the option (i.e., to buy) only on the option expiration date. In contrast, an American call option can be exercised at any time during the life of the option.

**Black-Scholes Equation with Dividends**

If the constant dividend payment is defined as \( D_y(t) \) per share, then the terms \( A \) and \( B \) in Eq. 3.25, take the form

\[ A = \mu n y(t) - A_f - r n y(t) + r F(y(t), t) - D n y(t) = 0 \]

\[ B = \sigma y(t) \left( n - \frac{\partial F}{\partial y(t)} \right) = 0 \]

yielding modified Black-Scholes Equation with Dividend

\[ \frac{\partial F}{\partial t} + \frac{1}{2} \sigma^2 y^2(t) \frac{\partial^2 F}{\partial y^2(t)} + (r - D) y(t) \frac{\partial F}{\partial y(t)} - r F = 0 \ (3.29) \]

**Transformation From Terminal Condition to Initial Condition Problem**

Since

\[ F(y(T), T) = 0, \]
solution to the Eq. 3.29 represents a terminal condition problem and thereby requires the backward solution. To make it an initial condition problem, we define

$$\tau = T - t = \text{“Time to go” (as in aerospace problems)}$$

also

$$F(y(t), t) = F(y(T - \tau), T - \tau) = f(y(\tau), \tau), \quad y(T - \tau) = s(\tau)$$

Black-Scholes equation takes the form:

$$-\frac{\partial f}{\partial \tau} + \frac{1}{2}\sigma^2 s^2(\tau) \frac{\partial^2 f}{\partial s^2(\tau)} + rs(\tau) \frac{\partial f}{\partial s(\tau)} - rf = 0 \quad (3.30)$$

or

$$\frac{\partial f}{\partial \tau} = L[F] \quad \text{where} \quad L = \frac{1}{2}\sigma^2 s^2(\tau) \frac{\partial^2 f}{\partial s^2(\tau)} + rs(\tau) \frac{\partial f}{\partial s(\tau)} - rf$$

Summarizing

$$\begin{align*}
    f(y(t), t) &= f(s(\tau), \tau) \\
    \tau &= T - t \\
    f(s(\tau), \tau) &\rightarrow s(\tau) - Ke^{-rt} \quad \text{as} \quad s(\tau) \rightarrow \infty \\
    f(s(\tau), \tau) &\rightarrow 0 \quad \text{as} \quad s(\tau) \rightarrow 0 \\
    f(s(0), 0) &= \max (s(0) - K, 0)
\end{align*} \quad (3.31)$$

Presence of $s(\tau)$ in the coefficients of this equation makes it a “time varying” PDE.

We can simplify this equation realizing that using “logarithmic prices”, this time variation disappears. In what follows, sometimes, whenever convenient, we shall be using the variables $y(t)$ or $s(\tau)$ as the situation may require.

**Logarithmic Version of Black-Scholes Equation**

Let

$$y(t) = e^{x(t)}, \quad x(t) = \ln y(t)$$

$$\frac{\partial F}{\partial y(t)} = \frac{\partial F}{\partial x(t)} \frac{\partial x(t)}{\partial y(t)} = 1 \frac{\partial F}{\partial y(t)}$$

Similarly
\[ \frac{\partial^2 F}{\partial y^2(t)} = \frac{1}{y^2(t)} \frac{\partial^2 F}{\partial x^2(t)} - \frac{1}{y^2(t)} \frac{\partial F}{\partial x(t)} \]

The simplified Black-Scholes equation takes the form:
\[ \frac{\partial F}{\partial t} + \left( \frac{\sigma^2}{2} \right) \frac{\partial^2 F}{\partial x^2(t)} + (r - \frac{\sigma^2}{2}) \frac{\partial F}{\partial x(t)} - rF(x(t), t) = 0 \quad (3.32) \]
\[ \hat{r} = (r - \sigma^2/2), \quad x(t) = \ln y(t) \]

and number of shares bought \( n \) is:
\[ n = e^{-x(t)} \frac{\partial F}{\partial x} \]

Black-Scholes equation described above is a parabolic PDE which can be transformed to a simplified heat equation so familiar to the physical scientists. The fundamental assumption for the equation is that the investment grows in time in a risk-free manner due to the interest rate \( r \) as \( e^{rt} \).

**This suggests a solution based upon separation of variables.**

Following three steps will be used to transform the Black-Scholes equation to the heat equation.

**Step 1: Separation of variables**

Let
\[ F(x(t), t) = F_1(t)F_2(x(t), t) \quad (3.33) \]

Then
\[ \frac{\partial F}{\partial t} = \frac{\partial F_1(t)}{\partial t} F_2(x(t), t) + F_1(t) \frac{\partial F_2(x(t), t)}{\partial t} \]
\[ \frac{\partial F}{\partial x(t)} = F_1(t) \frac{\partial F_2(x(t), t)}{\partial x(t)} \]
\[ \frac{\partial^2 F}{\partial^2 x(t)} = F_1(t) \frac{\partial^2 F_2(x(t), t)}{\partial^2 x(t)} \]

Substituting the above in Eq. 3.32 yields:
\[ F_1(t) \frac{\partial F_2(x(t), t)}{\partial t} + \frac{\partial F_1(t)}{\partial t} F_2(x(t)) + \frac{1}{2} \sigma^2 F_1(t) \frac{\partial^2 F_2(x(t))}{\partial^2 x(t)} + \hat{r} F_1(t) \frac{\partial F_2(x(t))}{\partial x(t)} - rF_1(t)F_2(x(t)) = 0 \]
or

\[ \left[ \frac{dF_1(t)}{dt} - rF_1(t) \right] F_2(x(t)) + \left[ \frac{\partial F_2(x(t), t)}{\partial t} + \frac{1}{2} \sigma^2 \frac{\partial^2 F_2(x(t))}{\partial x(t)^2} + \hat{r} \frac{\partial F_2(x(t))}{\partial x(t)} \right] F_1(t) = 0 \]

yielding two differential equations

\[ \frac{dF_1(t)}{dt} - rF_1(t) = 0 \]
\[ \frac{\partial F_2(x(t), t)}{\partial t} + \frac{\sigma^2}{2} \frac{\partial^2 F_2(x(t))}{\partial x(t)^2} + \hat{r} \frac{\partial F_2(x(t))}{\partial x(t)} = 0 \]

Solution to the first function is:

\[ F_1(t) = ce^{rt} \]

we chose

\[ F_1(T) = 1 \]

Hence

\[ c = e^{-rT} \]

or

\[ F_1(t) = e^{-r(T-t)} \] (3.34)

Second term takes the form:

\[ \frac{\partial F_2(x(t), t)}{\partial t} = -\hat{r} \frac{\partial F_2(x(t), t)}{\partial x(t)} - \frac{1}{2} \sigma^2 \frac{\partial^2 F_2(x(t), t)}{\partial x(t)^2} \] (3.35)

\[ \hat{r} = r - \frac{\sigma^2}{2} \]

**Step 2: Scaling**

To simplify the above equation in the standard diffusion equation form.

Let

\[ \hat{x}(t) = k_1 x(t) \]

and

\[ \hat{t} = k_2 (T - t) \]
\[ F_2(x(t), t) = F_2(k_1^{-1} \tilde{x}, -k_2^{-1} \tilde{t} + T) = \hat{F}_2(\tilde{x}, \tilde{t}) \]

Then

\[
\begin{align*}
\frac{\partial F_2}{\partial t} &= -k_2 \frac{\partial \hat{F}_2}{\partial \tilde{t}} \\
\frac{\partial F_2}{\partial x} &= k_1 \frac{\partial \hat{F}_2}{\partial \tilde{x}} \\
\frac{\partial^2 F_2}{\partial x^2} &= k_1^2 \frac{\partial^2 \hat{F}_2}{\partial \tilde{x}^2}
\end{align*}
\]

(3.36)

From Eq. 3.35 and Eq. 3.36

\[
-\left( \frac{2k_2}{\sigma^2} \right) \frac{\partial \hat{F}_2}{\partial \tilde{t}} = -\left( \frac{2\hat{\rho}}{\sigma^2 k_1} \right) \frac{\partial \hat{F}_2}{\partial \tilde{x}} - k_1^2 \frac{\partial^2 \hat{F}_2}{\partial \tilde{x}^2}
\]

Let

\[
\frac{2\hat{\rho}}{\sigma^2} = k_1 \\
\frac{2k_2}{\sigma^2} = k_1^2
\]

yielding:

\[ k_2 = \frac{2\hat{\rho}^2}{\sigma^2} \]

The simplified transformed equation is:

\[
\frac{\partial \hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t})}{\partial \tilde{t}} = \frac{\partial \hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t})}{\partial \tilde{x}} + \frac{\partial^2 \hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t})}{\partial \tilde{x}^2}
\]

(3.37)

Above equation can be further simplified by the change of the logarithmic price co-ordinates as following:

**Step 3:**

Let

\[
\tilde{x}(t) = u(\tilde{t}) + \tilde{t}
\]

\[
\hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t}) = \hat{F}_2(u(\tilde{t}) + \tilde{t}, \tilde{t}) = \hat{F}_2(u(\tilde{t}), \tilde{t})
\]

\[
\frac{\partial \hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t})}{\partial \tilde{x}(\tilde{t})} = \frac{\partial \hat{F}_2(u(\tilde{t}), \tilde{t})}{\partial u(\tilde{t})}
\]

\[
\frac{\partial^2 \hat{F}_2(\tilde{x}(\tilde{t}), \tilde{t})}{\partial \tilde{x}^2(\tilde{t})} = \frac{\partial^2 \hat{F}(u(\tilde{t}), \tilde{t})}{\partial u^2(\tilde{t})}
\]

(3.38)
Furthermore,
\[
\frac{\partial \hat{F}_2(u(\hat{t}), \hat{t})}{\partial \hat{t}} = \frac{\partial \hat{F}_2(\hat{x}(\hat{t}), \hat{t})}{\partial \hat{t}} - \frac{\partial \hat{F}_2(\hat{x}(\hat{t}), \hat{t})}{\partial \hat{x}}
\]

From Eqs. 3.37 and 3.38
\[
\frac{\partial \hat{F}_2(u(\hat{t}), \hat{t})}{\partial \hat{t}} = \frac{\partial^2 \hat{F}_2(u(\hat{t}), \hat{t})}{\partial u^2(\hat{t})}
\]
(3.39)

All this manipulation of variables is done so that:
\[
\frac{\partial \hat{F}_2}{\partial \hat{t}} = \frac{\partial \hat{F}_2}{\partial \hat{x}} + \frac{\partial^2 \hat{F}_2}{\partial \hat{x}^2}
\]

gets transformed to
\[
\frac{\partial \hat{F}_2}{\partial \hat{t}} = \frac{\partial^2 \hat{F}_2}{\partial u^2(\hat{t})} \quad \hat{u}(\hat{t}) = \text{function of } \hat{x} \text{ and } \hat{t}
\]
\[
\hat{F}_2 = \hat{F}_2(\hat{x}(\hat{t}), \hat{t}) = \hat{F}_2(\hat{u}(\hat{t}), \hat{t})
\]

This is the typical diffusion (heat) equation along with proper domain on \(\hat{t}\) and \(u(\hat{t})\).

Note:

There is a major drawback in the derivation of the Black-Scholes equation, even though
the original derivation described above still yields the correct B-S equation but the hedg-
ing strategy may be modified. The drawback lies in the fact that it calls for a fixed number
of shares to be purchased by the investor even though the derivation requires them to be
variable. We shall present below a modified derivation which allows the shares to be var-
ied and still yield the same B-S equation.
3.6 Modified Derivation of the Black-Scholes Equation

Let

\[ I(t) = \text{investment, such that} \]
\[ dI(t) = rI(t)dt, \quad r \text{ being the interest rate} \quad (3.40) \]

\[ V(y(t)) = \text{value of the shares bought by the investor at price } y(t) \text{ per share} \]
\[ V(0) = 0 \quad (3.41) \]

\[ V(y(t)) \text{ is assumed as a linear function of } y(t), \text{ the price of the share and can be expressed as:} \]
\[ V(y(t)) = y(t) \frac{\partial V(y(t))}{\partial y(t)}, \quad V(0) = 0 \quad \text{(Taylor series)} \quad (3.42) \]

Thus

\[ dV(y(t)) = \frac{\partial V(y(t))}{\partial y(t)} dy(t) \]

\[ \frac{\partial V(y(t))}{\partial y(t)} = \text{constant in variable } y(t) \text{ (but still a function of time } t) \]

From investment dynamics:

\[ I(t) = V(y(t)) - F(y(t), t) \quad (3.43) \]

where \( F(y(t), t) \) is the value of the call options for the share price \( y(t) \).

The market dynamics are given as:

\[ dy(t) = \mu y(t) dt + \sigma y(t) dW(t) \quad \text{Linear dynamics} \quad (3.44) \]

\( W(t) \) is the Brownian motion with zero mean and \( \sqrt{\tau} \) variance.

From Ito’s Lemma (described earlier)

\[ dF(y(t), t) = \frac{\partial F(y(t), t)}{\partial t} dt + \frac{\partial F(y(t), t)}{\partial y(t)} dy(t) + \frac{1}{2} \frac{\partial^2 F(y(t), t)}{\partial y^2(t)} dy^2(t) \]

\[ (dy(t))^2 = \sigma^2 y^2(t) dt \]
Hence
\[
dF(y(t), t) = \frac{\partial F(y(t), t)}{\partial t} dt + \frac{\partial F(y(t), t)}{\partial y(t)} dy(t) + \frac{\sigma^2}{2} y^2(t) \frac{\partial^2 F(y(t), t)}{\partial y^2(t)} dt
\] (3.45)

The \(dy(t)\) has a Brownian motion component \(dW(t)\) inside it.

From Eqs 3.40, 3.42 and 3.43

\[
dI(t) = dV(y(t)) - dF(y(t), t)
\]
or
\[
dF(y(t), t) = dV(y(t)) - dI(t) = \frac{\partial V(y(t))}{\partial y(t)} dy(t) - r(V(y(t)) - F(y(t), t)) dt
\]
or
\[
dF(y(t), t) = \frac{\partial V(y(t))}{\partial y(t)} dy(t) + rF(y(t), t) dt - ry(t) \frac{\partial V(y(t))}{\partial y(t)} dt
\] (3.46)

Equating Eqs. 3.45 and 3.46 yields:

\[
\frac{\partial F(y(t), t)}{\partial t} dt + \frac{\partial F(y(t), t)}{\partial y(t)} dy(t) + \frac{\sigma^2}{2} y^2(t) \frac{\partial^2 F(y(t), t)}{\partial y^2(t)} dt = \frac{\partial V(y(t))}{\partial y(t)} dy(t) + rF(y(t), t) dt - ry(t) \frac{\partial V(y(t))}{\partial y(t)} dt
\] (3.47)

To make the Eq. 3.47 risk-free and remove volatility which involves the \(dy(t)\) terms, we impose the condition:

\[
\frac{\partial F(y(t), t)}{\partial y(t)} = \frac{\partial V(y(t))}{\partial y(t)}
\]
yielding

\[
\frac{\partial F(y(t), t)}{\partial t} + ry(t) \frac{\partial F(y(t), t)}{\partial y(t)} + \frac{\sigma^2}{2} y^2(t) \frac{\partial^2 F(y(t), t)}{\partial y^2(t)} - rF(y(t), t) = 0
\] (3.48)

\[
V(y(t)) = y(t) \frac{\partial F(y(t), t)}{\partial y(t)}
\] (3.49)

Instead of varying the number of shares, an alternate way would be to vary the number of options and keep the number of shares constant. Following Block diagram brings a feedback perspective and a control systems formulation to the Black-Scholes equation.
In our research work, we shall try to invoke the optimal control theory algorithms for the hedging strategies involving the above system. In the next section, we derive the integral form of the Black-Scholes equation which is often referred to as “Black-Scholes Formula”.

### 3.7 Derivation of Black-Scholes Equation - Integral Form

**Risk-Free Environment**

Consider the market dynamics

\[
\frac{dy(t)}{y(t)} = \mu dt + \sigma dW(t)
\]

Let

\[W(t) = (\sqrt{t}) \hat{Z}\]

Thus

\[
y(t) = \left(y(0)e^{\mu t}\right)e^{\sigma W(t)} = \left(y(0)e^{\mu t}\right)e^{\sigma (\sqrt{t}) \xi}
\]
$W(t)$ is Wiener process with zero mean and variance $\sqrt{t}$. $\hat{Z}$ is a Gaussian random variable with zero mean and unity variance and is represented by a variable $\hat{z}$, $-\infty < \hat{z} < \infty$.

Let

\[
F(y(t), t, K) = \text{Estimated value of the stochastic European risk-free option at time } t
\]

\[
F(y(t), T, K) = \text{Estimated value of the stochastic European risk-free option at } t = T
\]

for a strike price $K$

\[
E[\cdot] = \text{Estimated value}
\]

Then

\[
F(y(T), T, K) = [F(y(t), t, K)]_{t=T} = [e^{-rt} E [\max((y(t) - K), 0)]]_{t=T} \quad (3.50)
\]

Thus

\[
F(y(t), t, K) = e^{-rt} \int_{-\infty}^{\infty} \max\left((y(0)e^{\mu t}e^{\sigma(\sqrt{t})\hat{z}} - K), 0\right) \left[\frac{1}{\sqrt{2\pi}} e^{-\hat{z}^2/2}\right] d\hat{z} \quad (3.51)
\]

The above integral has zero value when

\[
y(0)e^{\mu t}e^{\sigma(\sqrt{t})\hat{z}} - K \leq 0
\]

Let us designate the value of $\hat{z}$ as $\hat{z}^*$ when the equality occurs, namely

\[
y(0)e^{\mu t}e^{\sigma(\sqrt{t})\hat{z}^*} - K = 0
\]

or

\[
\hat{z}^* = \frac{\ln(y(0)/K) - \mu t}{\sigma(\sqrt{t})}
\]

obviously

\[
\max\left((y(0)e^{\mu t}e^{\sigma(\sqrt{t})\hat{z}} - K), 0\right) = \begin{cases} 
0 & \text{for } \hat{z} \leq \hat{z}^* \\
(y(0)e^{\mu t}e^{\sigma(\sqrt{t})\hat{z}} - K) > 0 & \text{for } \hat{z} > \hat{z}^*
\end{cases}
\]
Therefore
\[ F(y(t), t, K) = \frac{e^{-\gamma t}}{\sqrt{2\pi}} \int_{\hat{z}} \infty (y(0)e^{\mu t} e^{\sigma(\sqrt{t})\hat{z}} - K) e^{-\hat{z}^2/2} d\hat{z} \]

or
\[ F(y(t), t, K) = \frac{e^{-(r-\mu)t}}{\sqrt{2\pi}} y(0) \int_{\hat{z}} \infty e^{-(\hat{z}^2-2\sigma(\sqrt{t})\hat{z})/2} d\hat{z} - \frac{e^{-\gamma t}K}{\sqrt{2\pi}} \int_{\hat{z}} \infty e^{-\hat{z}^2/2} d\hat{z} \]

or
\[ F(y(t), t, K) = \left[ e^{-(r-\mu-\sigma^2/2)t} \right] y(0) \int_{\hat{z}} \infty \frac{e^{-\hat{z}^2/2}}{\sqrt{2\pi}} d\hat{z} - (e^{-\gamma t}K) \int_{\hat{z}} \infty \frac{e^{-\hat{z}^2/2}}{\sqrt{2\pi}} d\hat{z} \]

To Summarize:

Risk-Free, fair price European call option, which gives the right to buy a share of the security after time \( T \) at a strike price \( K \) is given by:

\[ F(K, T) = y(0)e^{-(r-\mu-\sigma^2/2)t}I_1(T) - Ke^{-\gamma T}I_2(T) \]

where

\[ I_1(T) = \frac{1}{\sqrt{2\pi}} \int_{\hat{z}} \infty e^{-(\hat{z}^2+\sigma^2\sqrt{T})/2} d\hat{z}, \quad I_2(T) = \frac{1}{2\pi} \int_{\hat{z}} \infty e^{-\hat{z}^2/2} d\hat{z} \]

\[ \hat{z}^* = \left( \frac{1}{\sigma \sqrt{T}} \right) \left( \ln(y(0)/K) - \mu T \right) \]

\( r = \) Risk-Free interest rate

\( \mu = \) Growth rate of the security

\( \sigma = \) Volatility

The first term represents the benefit of the option if one was to buy the stock outright with a value \( y(0) \) at time \( t = 0 \). The second term represents cost due to the exercise price \( K \). Difference between the benefit represented by the first term and the cost represented by the second term is the risk-free fair price of the option.

The expression inside the integral represents the familiar normal distribution density functions. Numerical computing algorithms are required to compute these integrals. Eqs 3.28 and 3.51 yield the same solution, one being a differential equation while the other is an integral equation.
In the next section we derive the equivalence of the differential and the integral approach. Differential approach yields the “diffusion” or the “heat” equation, while the integral approach arrives at the Black-Scholes Integral formula. We shall prove that via change of variables the heat equation can be converted to the integral form.

### 3.8 Equivalence Between Differential and Integral forms of Black-Scholes equation

For notational simplification we shall drop ‘’ from the symbols. Eq. 3.39 represents the modified form of Black-Scholes equation in the “Heat Equation” form as:

\[
\frac{\partial F}{\partial t} = \frac{\partial^2 F}{\partial u^2}, \quad F = F(t, u)
\]  
(3.52)

Since second partial derivative in “u” and first partial derivative in “t” are equal, it suggests a **Similarity Transformation** involving a new variable \(u^2/t\):

\[
z^2 = \frac{u^2}{2t} \quad \text{(factor of 2 is used for normalization)}
\]

or

\[
z = \frac{1}{\sqrt{2}} t^{-1/2} u
\]

Therefore

\[
\frac{\partial z}{\partial t} = \left( -\frac{1}{2\sqrt{2}} \right) t^{-3/2} u = \left( -\frac{1}{2} \right) t^{-1} z
\]

\[
\frac{\partial z}{\partial u} = \frac{1}{\sqrt{2}} t^{-1/2}
\]

Thus

\[
\frac{\partial F}{\partial t} = \left( \frac{\partial F}{\partial z} \right) \frac{\partial z}{\partial t} = \left( \frac{\partial F}{\partial z} \right) \left( -\frac{1}{2} t^{-1} z \right)
\]  
(3.53)

\[
\frac{\partial F}{\partial u} = \left( \frac{\partial F}{\partial z} \right) \frac{\partial z}{\partial u} = \left( \frac{\partial F}{\partial z} \right) \left( \frac{1}{\sqrt{2}} t^{-1/2} \right)
\]

\[
\frac{\partial^2 F}{\partial u^2} = \left( \frac{\partial^2 F}{\partial z^2} \right) \frac{\partial^2 z}{\partial u^2} = \frac{\partial^2 F}{\partial z^2} \left( \frac{1}{\sqrt{2} t^{-1/2}} \right)^2
\]  
(3.54)
From Eqs. 3.52, 3.53 and 3.54

\[
\left( \frac{\partial F}{\partial z} \right) \left( -\frac{1}{2} f^{-1} z \right) = \left( \frac{\partial^2 F}{\partial z^2} \right) \left( \frac{1}{\sqrt{2}} f^{-1/2} \right)^2
\]

Let

\[
\frac{\partial F}{\partial z} = \frac{dF}{dz} = h(z)
\]

Then

\[
-h(z)z = \frac{dh(z)}{dz}
\]

or

\[
\frac{1}{h(z)} \frac{dh(z)}{dz} = -z
\]

Integrating:

\[
\ln h(z) = -\frac{z^2}{2}
\]

or

\[
h(z) = e^{-\frac{z^2}{2}}
\]

or

\[
\frac{dF(z)}{dz} = e^{-\frac{z^2}{2}}
\]

or

\[
F(z) = c \int_{-\infty}^{z} e^{-\frac{z'^2}{2}} dz'
\] (3.55)

When all the boundary condition and all the variables along with proper parameters are substituted into the above integral, the Eq. 3.54 yields the same equation as the Eq. 3.51
### 3.9 Determination of the Updated Parameters of the Black-Scholes Equation

Given financial dynamics:

\[ dy(t) = \mu y(t) dt + \sigma y(t) dW(t) \]

\( y(t) \) = price of the security

\( \sigma \) = volatility index

\( \mu \) = drift in price of security per unit time

\( t \) = time variable \( 0 \leq t < T \)

\( W(t) \) = Brownian motion (noise)

\( x(t) = \ln y(t) \) = Logarithmic price of security = Lprice

Given: Measured Lprice of security:

\[ x_i = x(t_i), \quad i = 1, 2, \ldots, n \]

\[ t_{i+1} - t_i = dt \]

Required:

Determine the parameters \( \mu \) and \( \sigma \).

**Algorithm**

- **Determination of updated \( \mu \)**

\[ m_i = \Delta x_i = x_{i+1} - x_i = \mu dt + \epsilon_i \quad (3.56) \]

\[ \epsilon_i = \sigma dW(t_i) \quad i = 1, 2, \ldots, n \]

\[ H(\mu) = \sum_{i=1}^{n} (m_i - \mu dt)^2 = \sum_{i=1}^{n} \epsilon_i^2 \quad (3.57) \]

Minimizing \( H(\mu) \)

\[ \frac{\partial H(\mu)}{\partial \mu} = dt \sum_{i=1}^{n} (m_i - \mu dt) = 0 \]
or

\[ n\mu dt = \sum_{i=1}^{n} m_i \]

yielding

\[ \text{updated estimate } \mu = \frac{1}{ndt} \sum_{i=1}^{n} m_i \quad (3.58) \]

- **Determination of \( \sigma \)**

\[ e_i^2 = (m_i - \mu dt)^2 \]

or

\[ \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (m_i - \mu dt)^2 \]

But

\[ \sum_{i=1}^{n} e_i^2 = \sigma^2 \sum_{i=1}^{n} (dW(t_i))^2 = n\sigma^2 dt \]

Hence

\[ \text{updated estimate } \sigma = \left[ \left( \frac{1}{ndt} \right) \sum_{i=1}^{n} (m_i - \mu dt)^2 \right]^{1/2} \quad (3.59) \]

Thus Black-Scholes equation updated parameters are computed instant by instant.

### 3.10 Computation of the Initial Fair Price of the Option

In order to solve the Black-Scholes equation, we need to compute the initial Fair price of the option which depends upon:

1. Duration of the option \( T \)
2. Initial price of the security, \( y(0) \)
3. Interest rate per unit time, \( r \)
4. Strike price of the option, \( K \)
Let

\[ y(t) = \text{price of the security at time } t \]
\[ \hat{F}(t) = \text{Fair price of the option at time } t \]
\[ r = \text{interest rate} \]

Let us look at strike price \( k(t) \) as an investment which increases at the interest rate \( r \) and has a final value \( k(t)|_{t=T} = K \).

The governing equation is:

\[ \dot{k}(t) = rk(t), \quad k(T) = K \]

The resulting solution is:

\[ k(t) = Ke^{r(T-t)} \]

Hence the initial value is:

\[ k(0) = Ke^{-rT} \]

This initial value \( k(0) \) will result in a value

\[ k(T) = K, \quad \text{the strike price} \]

Let

\[ \hat{F}(0) = \text{Fair value of the option at } t = 0 \]

Thus the fair value \( \hat{F}(0) \) at the start of the option can be taken as:

<table>
<thead>
<tr>
<th>Initial value of the fair value option without volatility</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \hat{F}(0) = \left( Ke^{-rT} - y(0) \right) + \text{bias} ]</td>
</tr>
</tbody>
</table>

Bias can be computed from volatility.

\( \hat{F}(0) \) can be used to compute \( \hat{F}(t) \) via **Black-Scholes equations**.
3.11 Hedging Strategies via “The Greeks”

Greek Symbols

To hedge against implicit exposure to volatility, such as expenses due to frequent trades, a number of important relationships between the value of the option and its associated parameters have been developed. Since most of these are represented by Greek letters, they are referred to as the “Greeks”. The most popular Greeks are:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δ</td>
<td>Delta</td>
<td>Most important parameter</td>
</tr>
<tr>
<td>Γ</td>
<td>Gamma</td>
<td>Rate of change of Δ, indicator of sensitivity</td>
</tr>
<tr>
<td>θ</td>
<td>Theta</td>
<td>Rate of change of option with respect to time</td>
</tr>
<tr>
<td>ν or κ</td>
<td>Vega (or Kappa)</td>
<td>Sensitivity to volatility</td>
</tr>
<tr>
<td>ρ</td>
<td>Rho</td>
<td>Sensitivity to interest rate</td>
</tr>
</tbody>
</table>

These parameters are associated with the option in the following way:

\[
\Delta = \frac{\partial F}{\partial y} = \text{# of options which are required to remove volatility} = n
\]

\[
\Gamma = \frac{\partial^2 F}{\partial y^2} = \text{Low } \Gamma \text{ implies infrequent rebalancing}
\]

\[
\theta = \frac{\partial F}{\partial t}
\]

\[
\rho = \frac{\partial F}{\partial r}
\]

\[
\nu = \frac{\partial F}{\partial \sigma}
\]

In general, Δ and Γ are important and the hedging is referred to as “delta hedging”.
3.12 Analytical Expressions for the Greeks
for the European Options

\[ y(t) = \text{Stock price} \]
\[ F(y(t), t) = \text{European option} \]
\[ T = \text{Expiration time} \]
\[ K = \text{Strike price} \]
\[ \sigma = \text{Volatility} \]
\[ r = \text{Risk-free interest rate} \]
\[ D = \text{Continuous constant dividend} \]
\[ N(z^*) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z^*} e^{-z^2/2} dz \]
\[ d_1 = \frac{\ln(y(t)/K) - (r - D + \sigma^2/2)(T - t)}{\sigma \sqrt{T - t}} \]
\[ d_2 = d_1 - \sigma \sqrt{T - t} \]
\[ F(y(t), t) = \left( y(t)e^{-D(T-t)} \right) N(d_1) - \left( Ke^{-r(T-t)} \right) N(d_2) \]
\[ \Delta = e^{-D(T-t)} N(d_1) \]
\[ \Gamma = e^{-r(T-t)} N'(d_1) \left( \frac{1}{\sigma y(t) \sqrt{T - t}} \right) \]
\[ \theta = D y(t) N(d_1) e^{-D(T-t)} - r y(t) e^{-r(T-t)} N(d_2) - \Gamma \left( \frac{\sigma^2 y^2(t)}{2 \sqrt{T - t}} \right) \]
\[ \nu = e^{-D(T-t)} y(t) \sqrt{T - t} N'(d_1) \]
\[ \rho = e^{-r(T-t)} y(t)(T - t) N(d_2) \]

It is a real challenge to compute the Greeks with some reasonable accuracy unless the markets are smooth.

3.13 Numerical Methods for solving the Black-Scholes Type Equations

This section presents some existing methods for the solution of Black-Scholes type equation. These methods are essentially different variations of the finite difference grid and seek faster
solution and fewer grid points. These methods have an essential drawback due to “oscillations” and stability problems.

Let us consider the call options Black-Scholes equation (and its dividend variation):

\[
\frac{\partial F(x(t), t)}{\partial t} + a_2 \frac{\partial^2 F(x(t), t)}{\partial x^2(t)} + a_1 \frac{\partial F(x(t), t)}{\partial x(t)} + a_0 F(x(t), t) = 0
\] (3.60)

where

\[
a_2 = \frac{1}{2} \sigma^2
\]
\[
a_1 = \left( r - \frac{1}{2} \sigma^2 \right) = \hat{r}
\]
\[
a_0 = -r
\]
\[x(t) = \ln y(t), \quad \text{logarithmic stock price, } y(t) \text{ being the stock price}
\]

This is typical heat equation with backward time and the terminal conditions are:

\[
F(x(T), T) = \begin{cases} 
\max(\ln(K/y(T)), 0) & \text{for the put} \\
\max(\ln(y(T)/K), 0) & \text{for the call}
\end{cases}
\]

\[K = \text{exercise price of the European option expiring at time } T\]

Most common method for solving this equation is the finite difference method discussed by Geske and Shastri [Geske, R.], Courtadon [Courtadon, G.], Wilmott et al [Wilmott, P.], White and Hull [Hull, J.]. Bunnin et al [Bunnin, F.] use pseudospectral methods with increased collocation points involving Chebychev polynomial. Most schemes have accuracy and convergence problems. We propose a new method involving Kautz polynomial which are exponentially decaying functions providing unconditional convergence and stability. Kautz polynomials used here involve the space of orthogonal polynomials involving logarithmic stock price from \(x(0)\) to \(x(T)\) and transforming it into infinite domain variable \(z\), where \(0 < z < \infty\) is the domain of the space.

1. Method #1 \textbf{Finite Difference Method - Price Discretization}
Consider
\[ \frac{\partial F}{\partial t} = -a_2 \frac{\partial^2 F}{\partial x^2} - a_1 \frac{\partial F}{\partial x} - a_0 F, \quad F = F(x, t), \quad 0 < t < T \quad (3.61) \]
\[ F(x(0), t) = a(t) \]
\[ F(x(t), t) = b(t) \]

Let
\[ \frac{x(T) - x(0)}{N} = \Delta x = h \]
\[ x(0) + ih = x_i \]
\[ F(x(0) + ih) = F(x_i) = F_i = F_i(t) \]

Using second order approximation,
\[ \frac{\partial F_i}{\partial t} = \left( -\frac{a_2}{h^2} \right) (F_{i+1} - 2F_i + F_{i-1}) \]
\[ + \left( -\frac{a_1}{2h} \right) (F_{i+1} - F_{i-1}) + (-a_0) F_i + b_i(t) \quad 1 < i < N - 1 \quad (3.62) \]

The boundary conditions are:
\[ \frac{\partial F_1}{\partial t} = \left( -\frac{a_2}{h^2} \right) (F_2 - 2F_1 + F_0) + \left( -\frac{a_1}{2h} \right) (F_2 - F_0) + (-a_0) F_1 + b_1 \quad (3.63) \]
\[ \frac{\partial F_{N-1}}{\partial t} = \left( -\frac{a_2}{h^2} \right) (F_N - 2F_{N-1} + F_{N-2}) \]
\[ + \left( -\frac{a_1}{2h} \right) (F_N - F_{N-2}) + (-a_0) F_{N-1} + b_{N-1} \quad (3.64) \]
\[ F_N = x(T), \quad F_0 = x(0) \quad (3.65) \]

The variable \( b_i \) appears only in the first and last equation as
\[ b_i = \begin{cases} 
0 & 2 \leq i \leq N - 2 \\
-(x(0) + h) \left( a_2 \frac{a_1}{h^2} - a_1 \frac{1}{2h} \right) & i = 1 \\
-(x(t) + h) \left( a_2 \frac{a_1}{h^2} + a_1 \frac{1}{h} \right) & i = N - 1
\end{cases} \quad (3.66) \]
Eq. 3.62 can be written in the matrix form:

\[
\frac{dF(t)}{dt} = AF(t) + b(t) \tag{3.67}
\]

yielding a solution

\[
F(t) = e^{At}F(0) + \int_0^T e^{A(t-\sigma)}F(\sigma)d\sigma \tag{3.68}
\]

Note: These equations have been derived via second order approximation and can be modified and made more accurate via higher order Runge-Kutta approximation. A further assumption is that all the partial derivatives exist and can be computed.

Example: Parameters

\[
\begin{align*}
  r &= .1, \quad \sigma = .4, \quad (\mu = .2 \text{ not needed}) \\
  T &= .5, \quad K = 60 \text{ or } 100 \\
  \Delta t &= h = .005, \quad x(0) = 20, \quad x(T) = 100 \\
  F(x, T) &= \max(x - K, 0) \\
  F(x, t) &= 0 \\
  F(100, t) &= x - Ke^{-r(T-t)}
\end{align*}
\]
2. Method #2 Crank Nicholson Algorithm - Time-Price Discretization

Consider the Black-Scholes Heat Equation

\[
\frac{\partial F}{\partial t} = \frac{\partial^2 F}{\partial u^2} \quad 0 < t < T
\]

with

Initial condition:

\[F(u, 0)\]

and

Boundary conditions:

\[
\lim_{u \to \infty} F(u, t) = a \quad (3.69)
\]
\[
\lim_{u \to -\infty} F(u, t) = 0
\]

The computational space is divided into a price-time grid:

\[
t_m = m\Delta t, \quad t_M = T, \quad m = 0, 1, \ldots M
\]
\[
u_n = n\Delta u, \quad n = 0, 1, \ldots, N + 1
\]

\[F(u = u_n, t = t_m) = F(n, m) \quad (3.70)\]

Initial condition:

\[F(n, 0) = g_n \quad n = 0, 1, \ldots, N + 1 \quad (3.71)\]

Boundary conditions:

\[F(0, m) = \alpha_m \quad (3.72)
\]
\[F(N + 1, m) = \beta_m\]

The computation algorithm stability is very sensitive to time step-price step ratio. Recalling the similarity transformation while deriving the heat equation we came to the
conclusion that $u^2$ and $2t$ are similar to each other yielding a rough bound

$$u \approx \sqrt{2} t$$

or

$$\Delta u \approx \sqrt{2} \Delta t$$

In practice, we find the solution is stable if

$$\Delta u \geq \sqrt{2} \Delta t$$

Let

$$\frac{\Delta t}{2 (\Delta u)^2} = p \leq 1 \quad (3.73)$$

This can be verified by the eigenvalues of the ensuing matrix $A$ which should like within the unit circle.

We can discretize the heat equation as:

$$[F(n, m + 1) - F(n, m)] = p [F(n - 1, m + 1) - 2F(n, m + 1) + F(n + 1, m + 1)$$

$$+F(n - 1, m) - 2F(n, m) + F(n + 1, m)]$$

Figure 3.2: Time-Price Grid
or

\[-pF(n-1,m+1) + (1+2p)F(n,m+1) - pF(n+1,m+1) = pF(n-1,m) + (1-2p)F(n,m) + pF(n+1,m)\]  \hspace{1cm} (3.74)

In matrix form

\[(I + pA)u(m+1) = (I - pA)u(m) + b(m)\]

where

\[
A = \begin{bmatrix}
2 & -1 & 0 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 \\
0 & -1 & 2 & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 2 & -1 \\
0 & 0 & 0 & -1 & 2 \\
0 & 0 & 0 & 0 & -1 & 2
\end{bmatrix}, \hspace{0.5cm} b(m) = \begin{bmatrix}
p\alpha(m) \\
p\alpha(m+1) \\
0 \\
\vdots \\
p\beta(m) \\
p\beta(m+1)
\end{bmatrix}, \hspace{0.5cm} u(m) = \begin{bmatrix}
u(0,m) \\
u(1,m) \\
u(2,m) \\
\vdots \\
u(N-1,m) \\
u(N,m) \\
u(N+1,m)
\end{bmatrix}
\]

3. Method #3 Solution of Black-Scholes Equation via Kautz Polynomials

**Kautz Polynomial and Orthogonal Normal Functions**

These functions are commonly used in Electrical and Communication Systems.

**Preliminaries**

(a) Scaling:

The infinite space of the stock function is truncated to \([a, b]\) such that

\[x(0) = a\]
\[x(T) = b\]

Define a new variable \(z\), such that:

\[z = \frac{x(t) - a}{x(t) - b} = z(t), \quad \text{a real function}\]
\[ x(t) = \left( \frac{bz(t) - a}{z(t) - 1} \right) \]

We can perform the same scaling procedure for \( y(t) \) instead of logarithm value \( x(t) \), if required.

Thus

\[ z(0) = 0 \quad \text{and} \quad z(T) = \infty \]

(b) \textbf{Algorithm for generation of Kautz Polynomials}.

Consider a set of linearly independent exponential functions

\[ e^{-z}, e^{-2z}, \ldots, e^{-Nz} \quad 0 \leq z \leq \infty \]

These are our basis functions for generation of Kautz orthonormal polynomials.

(i) Let: \( \varphi_1(z), \varphi_2(z), \ldots, \varphi_N(z) \) be an orthonormal set such that:

Inner Product of \( \varphi_i(z) \) and \( \varphi_j(z) = (\varphi_i(z), \varphi_j(z)) = \int_0^\infty \varphi_i(z)\varphi_j(z)dz = \delta_{ij} \)

\[
\begin{cases} 
0 & i \neq j \\
1 & i = j
\end{cases}
\]

(ii) Select:

\[
\varphi_1(z) = \alpha_{11} e^{-z} \\
\int_0^\infty \varphi_1^2(z)dz = \alpha_{11}^2 \int_0^\infty e^{-2z}dz = \frac{\alpha_{11}^2}{2} = 1
\]

Thus

\[ \varphi_1(z) = \sqrt{2} e^{-z} \]

(iii) Let

\[ \varphi_2(z) = \alpha_{21} e^{-z} + \alpha_{22} e^{-2z} \]
The two parameters $\alpha_{21}, \alpha_{22}$ are computed as following:

$$(\varphi_1(z), \varphi_2(z)) = \alpha_{11} \left[ \alpha_{21} \int_0^\infty e^{-2z} dz + \alpha_{22} \int_0^\infty e^{-3z} dz \right] = \sqrt{2} \left( \frac{\alpha_{21}}{2} + \frac{\alpha_{22}}{3} \right) = 0$$

$$(\varphi_2(z), \varphi_2(z)) = \alpha_{21}^2 \int_0^\infty e^{-2z} dz + \alpha_{22}^2 \int_0^\infty e^{-4z} dz + 2\alpha_{21}\alpha_{22} \int_0^\infty e^{-3z} dz = 1$$

or

$$\frac{\alpha_{21}}{2} + \frac{\alpha_{22}}{3} = 0$$

$$\frac{\alpha_{21}^2}{2} + \frac{\alpha_{22}^2}{4} + 2\frac{\alpha_{21}\alpha_{22}}{3} = 1$$

Yielding

$$\alpha_{21} = -1/3, \quad \alpha_{22} = 1/2$$

Hence

$$\varphi_2(z) = -\frac{1}{3} e^{-z} + \frac{1}{2} e^{-2z}$$

(iv) Similarly choose:

$$\varphi_3(z) = \alpha_{31} e^{-z} + \alpha_{32} e^{-2z} + \alpha_{33} e^{-3z}$$

The parameters $\alpha_{31}, \alpha_{32},$ and $\alpha_{33}$ are computed as in (ii).

Finally we arrive at:

$$\varphi_N(z) = \alpha_{N1} e^{-z} + \alpha_{N2} e^{-2z} + \cdots + \alpha_{NN} e^{-Nz}$$

Hence we have presented an algorithm to generate a set of orthogonal polynomials

$$\varphi_1(z), \varphi_2(z), \cdots, \varphi_N(z)$$
3.13.1 Solution of Black-Scholes Equation via Kautz Polynomials

With the change of variables from \( x \) to \( z \), Eq. 3.61 takes the form:

\[
\frac{\partial F(z,t)}{\partial t} + a_2 \frac{(z-1)^4}{(b-a)^2} \frac{\partial^2 F(z)}{\partial z^2} + a_1 \frac{(z-1)^2}{(b-a)} \frac{\partial F(z)}{\partial z} + a_0 F(z,t) = 0 \quad (3.75)
\]

Let

\[
F(z(t), t) \approx \sum_{k=1}^{N} c_k(t) \phi_k(z) \quad (3.76)
\]

From Eqs. 3.75 and 3.76

\[
\sum_{k=1}^{N} \dot{c}_k(t) \phi_k(z) + \sum_{k=1}^{N} \left( a_0 \phi_k(z) + a_1 \phi_k'(z) + a_2 \phi_k''(z) \right) c_k(t) = 0 \quad (3.77)
\]

where \( \dot{\cdot} \) stands for derivative with respect to \( z \).

Taking inner product of the above expression with respect to \( \phi_j(z) \) \( (j = 1, 2, \ldots, N) \) yields:

\[
\sum_{k=1}^{N} \dot{c}_k(t) \left( \phi_k(z), \phi_j(z) \right) + \sum_{k=1}^{N} \left( \phi_j(z), \left( a_0 \phi_k(z) + a_1 \phi_k'(z) + a_2 \phi_k''(z) \right) c_k(t) \right) = 0 \quad (3.78)
\]

Let

\[
\begin{align*}
\left( \phi_j(z), \phi_j'(z) \right) &= \alpha_{jk} \\
\left( \phi_j(z), \phi_j''(z) \right) &= \beta_{jk} \\
\left( \phi_j(z), \phi_k(z) \right) &= \delta_{jk}, \quad \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases} \\
a_0 \delta_{jk} + a_1 \alpha_{jk} + a_2 \beta_{jk} &= a_{jk}
\end{align*}
\]

\[
A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}, \quad \mathbf{c}(t) = \begin{bmatrix} c_1(t) \\ c_2(t) \\ \vdots \\ c_N(t) \end{bmatrix}
\]
The solution of Eq. 3.76 is obtained as a matrix equation:

\[ \dot{c}(t) = Ac(t) \]

or

\[ c(t) = e^{At}c(0) \]

The initial condition for \( \{c_k(t)\}_{k=0}^N \) are determined by

\[ F(x(0), 0) = \ln(y(0) - Ke^{-rT}), \quad (x(0) = \ln y(0)) \]

or

\[ \sum_{k=1}^{N} c_k(0)\varphi_k(x(0)) = F(x(0), 0) = \ln(y(0) - Ke^{-rT}), \quad (x(0) = \ln y(0)) \]

This equation is computed at \( N \) points.

If \( x(0) \) is an \( N \) collocation points \( x_j(0), \quad j = 1, \ldots, N \), we obtain:

\[
\begin{bmatrix}
\varphi_1(x_1(0)) & \varphi_2(x_1(0)) & \cdots & \varphi_N(x_1(0)) \\
\varphi_1(x_2(0)) & \varphi_2(x_2(0)) & \cdots & \varphi_N(x_2(0)) \\
\vdots \\
\varphi_1(x_N(0)) & \varphi_2(x_N(0)) & \cdots & \varphi_N(x_N(0))
\end{bmatrix}
\begin{bmatrix}
c_1(0) \\
c_2(0) \\
\vdots \\
c_N(0)
\end{bmatrix}
= 
\begin{bmatrix}
\ln(y_1(0) - Ke^{-rT}) \\
\ln(y_2(0) - Ke^{-rT}) \\
\vdots \\
\ln(y_N(0) - Ke^{-rT})
\end{bmatrix}
\]
In this thesis, we have developed investment strategies involving the decision making needs via control systems technique. Modeling, optimization, identification and computation methods used in systems engineering have been successfully applied. The major contribution of this thesis is the dynamic modeling and forecasting algorithm as well as development of new ideas involving hedging strategies presented by the Black-Scholes equation and their stable solutions. Introduction of adaptive techniques for online updating and identification of parameters such interest rate, volatility and drift are presented. The power of dynamic modeling and forecasting algorithm is demonstrated by comparing the computer results with the actual S&P 500 index data involving 300 days. In addition, financial tools have been developed for hedging risk. Furthermore, an additional contribution of this thesis is an optimization technique for portfolios involving multiple entities. The mean square algorithm presented, assures stability. Furthermore, the thesis presents a new way of deriving the Black-Scholes equations and how to modify these equations so that the desired profitability is assured. The second part of the thesis considers transform techniques involving orthonormal polynomials known as 'Kautz polynomials', which compared with the methods in the literature, provides stable solutions. This opens a line of research for stable numerical computation of Black-Scholes equations involving optimal option pricing.
Appendix A

Normal Distribution Expectation

Let $z$ be a normally distributed random variable with mean $m$ and variance $\sigma$, defined as $N(m, \sigma^2)$. The probability distribution function of this random variable is defined as:

$$F(Z \leq z) = N(z^*) = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{z^*} e^{-(z-m)^2/2\sigma^2} \, dz$$  \hspace{1cm} (A.1)$$

Associated with it is the probability density function:

$$f(z) = \frac{1}{\sigma \sqrt{2\pi}} e^{-(z-m)^2/2\sigma^2}$$

It is easy to see that

$$N(z^*) + N(-z^*) = 1$$  \hspace{1cm} (A.2)$$

Furthermore,

$$E[z] = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{+\infty} z e^{-(z-m)^2/2\sigma^2} \, dz$$  \hspace{1cm} (A.3)$$

$$E[g(z)] = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{+\infty} g(z) e^{-(z-m)^2/2\sigma^2} \, dz$$  \hspace{1cm} (A.4)$$

$$1 = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-z^2/2\sigma^2} \, dz$$  \hspace{1cm} (A.5)$$

This appendix is the mainstay for the derivation of Black-Scholes formula.
Appendix B

Minimum Mean Square Solution

Given \( y(\tau) \) and \( M(\tau) \), the vector \( a(\tau) \) is so chosen as to minimize

\[
I = (M(\tau)a(\tau) - y(\tau))^T(M(\tau)a(\tau) - y(\tau))
\]

This minimization results in:

\[
\hat{a}(\tau) = \left( M^T(\tau)M(\tau) \right)^{-1}M^T(\tau)y(\tau) \quad (B.1)
\]

Let

\[
\begin{bmatrix} M^T(\tau)M(\tau) \end{bmatrix} = B(\tau) \quad \text{(a symmetric matrix)},
\]

\[ M^T(\tau)y(\tau) = z(\tau) \]

yielding

\[ \hat{a}(\tau) = B^{-1}(\tau)z(\tau) \]
Appendix C

Derivation of Recursive Algorithm for Matrix Inversion

Let
\[
B_l(\tau) = \begin{bmatrix} b_{11}(\tau) & \cdots & b_{1l}(\tau) \\ \vdots & \ddots & \vdots \\ b_{1l}(\tau) & \cdots & b_{ll}(\tau) \end{bmatrix}_{l \times l}, \quad B_{l+1}(\tau) = \begin{bmatrix} b_{11}(\tau) & \cdots & b_{1l+1}(\tau) \\ \vdots & \ddots & \vdots \\ b_{1l+1}(\tau) & \cdots & b_{ll+1}(\tau) \end{bmatrix}_{(l+1) \times (l+1)}
\]

Define,
\[
B_{l+1}(\tau) = \begin{bmatrix} B_l(\tau) & c_{l+1}(\tau) \\ c_{l+1}^T(\tau) & b_{l+1,l+1}(\tau) \end{bmatrix}, \quad c_{l+1}(\tau) = \begin{bmatrix} b_{1,l+1}(\tau) \\ \vdots \\ b_{l,l+1}(\tau) \end{bmatrix}
\]

\[
B_{l+1}(\tau)B_{l+1}^{-1}(\tau) = I_{l+1} \quad \text{Identity Matrix}
\]

\[
B_l(\tau)A_{l+1}(\tau) + c_{l+1}(\tau)e_{l+1}(\tau) = I \quad \text{(Identity matrix)}
\]

\[
B_l(\tau)e_{l+1}(\tau) + c_{l+1}(\tau)f_{l+1}(\tau) = 0
\]

\[
c_{l+1}^T(\tau)e_{l+1}(\tau) + b_{l+1,l+1}(\tau)f_{l+1}(\tau) = 1
\]

Let
\[
B_l^{-1}c_{l+1}(\tau) = g_{l+1}(\tau)
\]

Solution of Eq. C.1 is:

\[
e_{l+1}(\tau) = -\left(f_{l+1}(\tau)\right)g_{l+1}(\tau)
\]

\[
f_{l+1}(\tau) = \left(b_{l+1,l+1} - c_{l+1}^T(\tau)g_{l+1}(\tau)\right)^{-1}
\]

\[
A_{l+1}(\tau) = B_l^{-1}(\tau) + f_{l+1}(\tau)g_{l+1}(\tau)g_{l+1}^T(\tau)
\]

Thus \( B_{l+1}^{-1}(\tau) \) is computed from \( B_l^{-1}(\tau) \) and \( B_{l+1}(\tau) \). When \( l = n \), the process is terminated.
Appendix D

Derivation of Incremental Matrix Inversion

Derivation of incremental Matrix Inversion.

For convenience purpose, we shall drop the subscript \( k \) assuming the same algorithm will be used for all portfolio entities. Hence \( B_k(\tau) \) will be treated as \( B(\tau) \).

Given:

\[
B(\tau + 1) = \left(B(\tau) + \Delta(\tau)\right)
\]

Let

\[
(B(\tau + 1))^{-1} = \left(B(\tau) + \Delta(\tau)\right)^{-1} = \left(B(\tau) + \epsilon\Delta(\tau)\right)^{-1}, \quad \epsilon = 1
\]

From Taylor series,

\[
\left(B(\tau) + \epsilon\Delta(\tau)\right)^{-1} = T_0(\tau) + \epsilon T_1(\tau) + \epsilon^2 T_2(\tau) + \cdots = \sum_{i=0}^{\infty} \epsilon^i T_i(\tau)
\]

or

\[
\left(\sum_{i=0}^{\infty} \epsilon^i T_i(\tau)\right)\left(B(\tau) + \epsilon\Delta(\tau)\right) = I
\]

Equating powers of \( \epsilon \),

\[
T_i(\tau) = (-1)^i T_{i-1}(\tau)\Delta(\tau)B^{-1}(\tau), \quad i = 1, 2, \cdots
\]

\[
T_0(\tau) = B^{-1}(\tau)
\]

For

\[
\frac{|\Delta(\tau)|}{|B(\tau)|} << 1
\]

\[
B^{-1}(\tau + 1) \approx B^{-1}(\tau) - B^{-1}(\tau)\Delta(\tau)B^{-1}(\tau)
\]
Appendix E

DMFA Program code

The major classes for the DMFA algorithm are as shown below:

(a) MatrixUtil.vb (matrix operations and functions) Public Function initialStateMatM(ByVal n As Integer, ByVal m As Integer, ByRef datapts As Decimal()) As Decimal()

Dim matM As Decimal()
Dim i, j As Integer
ReDim matM(n - 1, m - 1)
'loop to populate matM
For i = 0 To n - 1
For j = 0 To m - 1
matM(i, j) = datapts(i + j)
Next
Next
Return matM
End Function

'Function to transpose a matrix
Public Function MatTranspose(ByRef matM As Decimal()) As Decimal()
Dim matMT As Decimal()
Dim dimN, dimM As Integer
Dim i, j As Integer
dimN = matM.GetLength(0)
dimM = matM.GetLength(1)
ReDim matMT(dimM - 1, dimN - 1)
`Transpose matM into matMT

For i = 0 To dimN - 1
For j = 0 To dimM - 1
matMT(j, i) = matM(i, j)
Next
Next
Return matMT
End Function

Public Function MatInverse(ByRef matB As Decimal(,)) As Decimal(,)
Dim matBInv As Decimal(,)
Dim matBRedDim As Decimal(,)
Dim matBRedInv As Decimal(,)
Dim rowDim, colDim As Integer
Dim i, j As Integer
Dim el As ErrorLogger = ErrorLogger.GetInstance
Try
rowDim = matB.GetLength(0) - 1
colDim = matB.GetLength(1) - 1
If rowDim = 0 AndAlso colDim = 0 Then
Dim matBInv0 As Decimal(,)
Dim epsilon As Decimal = 0.00001
ReDim matBInv0(0, 0)
´ensure matB(0,0) neq 0
If matB(0, 0) < epsilon AndAlso matB(0, 0) > -1.0 * epsilon Then
matBInv0(0, 0) = CDec(1.0)
Else
matBInv0(0, 0) = CDec(1.0 / matB(0, 0))
End If
Return matBInv0
Else
matBRedDim = MatReducedDim(matB)
matBRedInv = MatInverse(matBRedDim)
'remove last element
Dim c_nvec As Decimal()
Dim c_n As Decimal(,)
c_nvec = MatRowCol2Vec(matB, "columnvec", colDim)
ReDim Preserve c_nvec(c_nvec.Length - 2)
c_n = Vec2MatRowCol(c_nvec, "rowmat")
Dim c_nT As Decimal(,)
c_nT = MatTranspose(c_n)
Dim g_n As Decimal(,)
g_n = MatMultiply(matBRedInv, c_n)
Dim g_nT As Decimal(,)
g_nT = MatTranspose(g_n)
Dim f_n As Decimal
f_n = CDec(1.0 / (matB(rowDim, colDim) - MatMultiply(c_nT, g_n)(0, 0)))
Dim e_n As Decimal(,)
e_n = MatMultiplyScalar(CDec(-1.0 * f_n), g_n)
Dim e_nT As Decimal(,)
e_nT = MatTranspose(e_n)
Dim A_n As Decimal(,)
A_n = MatAdd(matBRedInv, MatMultiplyScalar(f_n, MatMultiply(g_n, g_nT)))
ReDim matBInv(rowDim, colDim)
  ’Fill each element of this matrix as per algorithm order
  ’---------------------------------------------------------------
  ’---------------------------------------------------------------
  ’---------------------------------------------------------------
  ’step 1: fill the first n X n from A_n
  ’---------------------------------------------------------------
For i = 0 To rowDim - 1
For j = 0 To colDim - 1
matBInv(i, j) = A_n(i, j)
Next
Next

' step 2: fill last column minus last element from e_n
For i = 0 To rowDim - 1
matBInv(i, colDim) = e_n(i, 0)
Next

' step 3: fill last row minus last element from e_nT
For i = 0 To colDim - 1
matBInv(rowDim, i) = e_nT(0, i)
Next

' step 4: fill last element from f_n
matBInv(rowDim, colDim) = f_n

Return matBInv
End If

Catch ex As Exception
el.WriteLine("Thread(" & Thread.CurrentThread.Name & "), Error in MatrixUtil.MatInverse:
" & ex.Message)
End Try
End Function

(b) DataDriver.vb (main driver program) Public Sub EstimateData()
    Dim numpts As Integer
Dim datapts As Decimal()
Dim dataframe As Decimal()
Dim dataframeNorm As Decimal()
Dim Tau As Integer
Dim i As Integer
Dim matM As Decimal(,)
Dim matMT As Decimal(,)
Dim matB As Decimal(,)
Dim matBInv As Decimal(,)
Dim matInvCheck As Decimal(,)
Dim matA_ As Decimal(,)
Dim vecA As Decimal()
Dim matY_ As Decimal(,)
Dim matZ_ As Decimal(,)
Dim matXT_ As Decimal(,)
Dim estpt_ As Decimal(,)
Dim estpt As Decimal()
Dim estdataframe As Decimal()
Dim errdataframe As Decimal()
Dim errpctdataframe As Decimal()
Dim el As ErrorLogger = ErrorLogger.GetInstance
Try
   'create n X m initial input matrix
   '-----------------------------------------------
   'FuncName = "exponentialSinCos" "linear" "parabola" "exponential" "exponentialSin-Cos" "exponentialSinCosLinear"
   'numpts = Util.NUMDATAPTS
   'M = Util.PARAM_M
   'N = Util.PARAM_N
'EstDatapts = Util.PARAM_ESTIMATED_DATAPOINTS
FuncName = IPDataRow.FuncName
numpts = IPDataRow.Numpts
M = IPDataRow.M
N = IPDataRow.N
EstDatapts = IPDataRow.EstDatapts

'Generate and write initial data vector to a file

datapts = Util.generateData(FuncName, numpts)
DatLog.WriteToFileData(datapts, DataFile, Util.Debug)

'get exact dataframe and fill from datapts depending on tau

ReDim dataframe(N + M - 1)
ReDim estdataframe(N + M + EstDatapts - 1)
ReDim matY_ (N - 1, 0)
ReDim matXT_ (0, M - 1)
TStart = Now

'Calculate for moving tau

For Tau = 0 To EstDatapts - 1
'Done for tau = 0 for now. For increasing tau drop first point in datapts and add another.
'copy into dataframe relevant datapoints from datapts
Dim len As Integer
len = dataframe.Length
Array.Clear(dataframe, 0, dataframe.Length)
'fill dataframe with required points from datapts
For i = 0 To N + M - 1
dataframe(i) = datapts(i + Tau)
estdataframe(i) = datapts(i)
Next
'fill vecY or matY_
For i = M To M + N - 1
matY_(i - M, 0) = dataframe(i)
Next
'fill vecX or matX_
For i = M + N - 1 To N Step -1
matXT_(0, M + N - 1 - i) = dataframe(i)
Next
'normalize the dataframe
'check for first datapoint neq 0 else inverse will fail
dataframeNorm = Util.NormalizeData(dataframe)
DatLog.WriteToFileData(dataframeNorm, NormdataFile, Util.Debug)
'create matrix M
matM = MatrixUtil.initialStateMatM(N, M, dataframe)
'create transpose of matrix M
matMT = MatrixUtil.MatTranspose(matM)
'multiple matMT with matM to create square matB of size m X m
matB = MatrixUtil.MatMultiply(matMT, matM)
DatLog.WriteToFileMat(matB, MatrixFile, Util.Debug)
'find inverse of matB to create matBInv
ReDim matBInv(matB.GetLength(0) - 1, matB.GetLength(1) - 1)
matBInv = MatrixUtil.MatInverse(matB)
DatLog.WriteToFileMat(matBInv, MatrixInvFile)
'calculate matBInv * matB to check for inverse calculation
matInvCheck = MatrixUtil.MatMultiply(matBInv, matB)
DatLog.WriteToFileMat(matInvCheck, MatrixInvCheckFile)
'if matinvCheck is not unity then log error and abort
If Not MatrixUtil.MatInverseChkOkay(matInvCheck) Then
    Exit Sub
End If

'calculate parameter vector vecA = matBInv * matMT * vecY
matZ_ = MatrixUtil.MatMultiply(matMT, matY_)
matA_ = MatrixUtil.MatMultiply(matBInv, matZ_)
vecA = MatrixUtil.MatRowCol2Vec(matA_, "columnvec", 0)
DatLog.WriteToFileData(vecA, ParameterFile, Util.Debug)

'predict next point
estpt_ = MatrixUtil.MatMultiply(matXT_, matA_)
estpt = MatrixUtil.MatRowCol2Vec(estpt_, "columnvec", 0)

'store predicted points in an array
estdataframe(N + M + Tau) = estpt(0)
Next
TStop = Now

'save predicted points in a file
DatLog.WriteToFileData(estdataframe, EstdataFile)

'Save Error Statistics
Dim firstprojpt As Integer
firstprojpt = M + N
errdataframe = Util.DataError(datapts, estdataframe, firstprojpt)
errpctdataframe = Util.DataErrorPct(datapts, estdataframe, firstprojpt)
DatLog.WriteToFileData(errdataframe, ErrdataFile)
If Not ReportStatistics(errdataframe, errpctdataframe) Then
Exit Sub
End If

Catch ex As Exception
el.WriteLine("Thread(" & Thread.CurrentThread.Name & "), Error in DataDriver.EstimateData:
" & ex.Message)
End Try
End Sub
Appendix F

Generation of Kautz Polynomials

Method #1

Given: A set of linearly independent vectors \( \{y_i = e^{-iz}\}_{i=1}^N, \ 0 \leq z \leq 1:\)

\[
\left(y_i(z), y_j(z)\right) = \int_0^\infty e^{-(i+j)z}dz = \frac{1}{(i+j)}, \quad i = 1, 2, \ldots, j = 1, 2, \ldots
\]

Required:

Generate a set \( \{\varphi_i(z)\}_{i=1}^N \) such that

\[
\left(\varphi_i(z), \varphi_j(z)\right) = \delta_{ij} = \int_0^\infty \varphi_i(z)\varphi_j(z)dz, \quad i = 1, 2, \ldots, j = 1, 2, \ldots
\]

Algorithm

Let

\[
y_1(z) = e^{-z}, \ ||y_1(z)|| = (e^{-z}, e^{-z}) = \int_0^\infty e^{-2z}dz = 1/2
\]

\[
\varphi_1(z) = \alpha_{11}e^{-z}, \quad (\varphi_1(z), \varphi_1(z)) = 1 = \alpha_{11}^2 \int_0^\infty e^{-2z}dz = \alpha_{11}^2/2
\]

1) Thus \( \varphi_1(z) = \sqrt(2)e^{-z} \)

\[
\varphi_k(z) = \sum_{l=1}^k \beta_{kl}e^{-lz}, \quad \beta_{11} = \alpha_{11}, \quad k = 1, 2, \ldots, i
\]

Thus \( \varphi_1(z), \varphi_2(z), \ldots, \varphi_i(z) \) are known. We are required to generate \( \varphi_{i+1}(z) \), when

\[
\beta_{kl}, l = 1, 2, \ldots, k, \quad k = 1, 2, \ldots, i \text{ are known}
\]
2) Let
\[
y_{i+1}(z) = \sum_{l=1}^{i} \alpha_{i+1,l} e^{-l z} + e^{-(i+1) z} \tag{F.1}
\]
\[
\varphi_{i+1}(z) = \sum_{l=1}^{i} k_{i+1} \alpha_{i+1,l} e^{-l z} + k_{i+1} e^{-(i+1) z}
\]
\[
(y_{i+1}(z), \varphi_{i+1}(z)) = 0, \quad k = 1, 2, \cdots, i \tag{F.2}
\]
\[
\varphi_{i+1}(z) = \sum_{l=1}^{i+1} \beta_{i+1,l} e^{-l z} \tag{F.3}
\]

So the problem boils down to determination of the coefficients

\[
(\beta_{i+1,1}, \beta_{i+1,2}, \cdots, \beta_{i+1,i+1}) \quad \text{and} \quad (\alpha_{i+1,1}, \alpha_{i+1,2}, \cdots, \alpha_{i+1,i})
\]

\[
(y_{i+1}(z), \varphi_{k}(z)) = 0, \quad k = 1, 2, \cdots, i \quad \text{yielding} \quad (i - 1) \text{ equations}
\]

\[
(y_{i+1}(z), \varphi_{k}(z)) = 0 = \sum_{j=1}^{i} \alpha_{i+1,j} \left( \varphi_{k}(z), e^{-j z} \right) + \left( \varphi_{k}(z), e^{-(i+1) z} \right), \quad k = 1, 2, \cdots, i
\]

Let

\[
\left( \varphi_{k}(z), e^{-j z} \right) = I_{jk}, \quad \left( \varphi_{k}(z), e^{-(i+1) z} \right) = I_{i+1,k}
\]

Thus

\[
-I_{i+1,k} = \sum_{j=1}^{i} \alpha_{i+1,j} I_{jk}, \quad k = 1, 2, \cdots, i
\]

\[
\begin{pmatrix}
I_{i+1,1} \\
I_{i+1,2} \\
\vdots \\
I_{i+1,i}
\end{pmatrix}
= 
\begin{pmatrix}
I_{11} & I_{21} & \cdots & I_{i1} \\
I_{12} & I_{22} & \cdots & I_{i2} \\
\vdots & \vdots & \ddots & \vdots \\
I_{1i} & I_{2i} & \cdots & I_{ii}
\end{pmatrix}
\begin{pmatrix}
\alpha_{i+1,1} \\
\alpha_{i+1,2} \\
\vdots \\
\alpha_{i+1,i}
\end{pmatrix}
\]

\[
(F.4)
\]

Eq. F.4 is used to solve for \( \alpha_{i+1,k}, k = 1, 2, \cdots, i \)

**Computation of \( I_{jk}, j = 1, 2, \cdots, i, \quad k = 1, 2, \cdots, i \)**

\[
I_{jk} = \left( \varphi_{k}(z), e^{-j z} \right) = \sum_{l=1}^{k} \beta_{kl} \int_{0}^{\infty} e^{-z} e^{-j z} dz = \sum_{l=1}^{k} \left( \frac{\beta_{kl}}{l + j} \right)
\]

\[
(F.5)
\]
$\beta_{kl}$ are known thus $I_{jk}$ are computed.

Let

$$\varphi_{i+1}(z) = k_{i+1}y_i(z)$$

{$y_i(t)$}$_{i=1}^N$ is an orthogonal set while {$\varphi_i(z)$}$_{i=1}^N$ is orthonormal set.

$$(\varphi_{i+1}(z), \varphi_{i+1}(z)) = 1 = k_{i+1}^2 \left( \sum_{l=1}^{i} \alpha_{i+1,l} e^{-lz} + e^{-(i+1)z} \right) \left( \sum_{m=1}^{i} \alpha_{i+1,m} e^{-jm} + e^{-(i+1)z} \right)$$

or

$$1 = k_{i+1}^2 \left[ \sum_{l=1}^{i} \sum_{m=1}^{i} (\alpha_{i+1,l}\alpha_{i+1,m}) \left( \frac{1}{l+m} \right) + \sum_{l=1}^{i} \alpha_{i+1,l} \left( \frac{1}{i+l+1} \right) + \sum_{m=1}^{i} \alpha_{i+1,m} \left( \frac{1}{i+m+1} \right) + \frac{1}{2(i+1)} \right]$$

Thus

$$\varphi_{i+1}(z) = \sum_{l=1}^{i} (K_{i+1}\alpha_{i+1,l}) e^{-lz} + K_{i+1} e^{-(i+1)z}$$

also

$$\varphi_{i+1}(z) = \sum_{j=1}^{i+1} \beta_{i+1,j} e^{-lz}$$

Thus

$$\beta_{i+1,l} = K_{i+1} \alpha_{i+1,l}, \quad l = 1, 2, \ldots, i$$

$$\beta_{i+1,i+1} = K_{i+1}$$

Summary

(1) $k_1 = \sqrt{2} = \alpha_{11}$

$$\varphi_k(z) = \sum_{l=1}^{k-1} K_l \alpha_{kl} e^{-lz} + K_k e^{-kz}, \quad k = 1, 2, \ldots, i$$

$K_k, \alpha_{kl}$ are known for $k = 1, 3, \ldots, i; l = 1, 2, \ldots, k - 1$ (scalar co-efficients)

(2) Compute

$$I_{jk} = \sum_{l=1}^{k-1} K_l \alpha_{kl} (e^{-lz}, e^{-ljz}) + K_k (e^{-lz}, e^{-jz})$$
or

\[ I_{jk} = \sum_{l=1}^{k-1} K_k \alpha_{kl} \left( \frac{1}{l+j} \right) + K_k \left( \frac{1}{k+j} \right) \quad k = 1, 2, \ldots, i \]

\[ j = 1, 2, \ldots, i + 1 \]

(3) This equation is used to solve for \( \alpha_{i+1,l}, l = 1, 2, \ldots, i \)

\[
\begin{bmatrix}
I_{i+1,1} \\
I_{i+1,2} \\
\vdots \\
I_{i+1,i}
\end{bmatrix} =
\begin{bmatrix}
I_{11} & I_{21} & \cdots & I_{i1} \\
I_{12} & I_{22} & \cdots & I_{i2} \\
\vdots & \vdots & \ddots & \vdots \\
I_{1i} & I_{2i} & \cdots & I_{ii}
\end{bmatrix}
\begin{bmatrix}
\alpha_{i+1,1} \\
\alpha_{i+1,2} \\
\vdots \\
\alpha_{i+1,i}
\end{bmatrix}
\]

(4) Compute

\[ h_{i+1} = \left[ \sum_{l=1}^{i} \sum_{m=1}^{j} (\alpha_{i+1,l} \alpha_{i+1,m}) \left( \frac{1}{l+m} \right) + 2 \sum_{l=1}^{i} \frac{\alpha_{i+1,l}}{(l+i+1)} + \frac{1}{2(i+1)} \right] \]

\[ K_{i+1} = \frac{1}{\sqrt{h_{i+1}}} \]

\[ \varphi_{i+1}(z) = \sum_{l=1}^{i} K_{i+1} \alpha_{i+1,l} e^{-lz} + K_{i+1} e^{-(i+1)z} \]
Method #2

\[ y(s) = \int_0^\infty y(z)e^{-zs} \, dz \]

\[ y_k(z) \perp y_j(z), \quad k \neq j \]

\[ y_k(s) = \left[ \prod_{i=1}^{k-1} (i - s) \right] \left[ \prod_{i=1}^k (i + s) \right] \]

\[ y_k(s) = \sum_{j=1}^k \alpha_{kj} \frac{1}{(j + s)} \quad \rightarrow \quad y_k(z) = \sum_{j=1}^k \alpha_{kj} e^{-jz} \]

where

\[ \alpha_{kj} = \left[ \prod_{i=1}^{k-1} (i - s) \right] \left[ \prod_{i+j}^k (i + s) \right] \]

\[ = \begin{cases} \frac{(1 + j)(2 + j) \cdots (k - 1 + j)}{(1 - j)(2 - j) \cdots (j - 1 - j)(j + 1 - j)(j + 2 - j) \cdots (k - j)} & j \leq k \\ 0 & j > k \end{cases} \]

\[ K_k^2 \left( \sum_{j=1}^k \sum_{l=1}^k \alpha_{kj} \alpha_{kl} \left( \frac{1}{j + l} \right) \right) = 1 \]

\[ K_k = \left( \frac{1}{\sum_{j=1}^k \sum_{l=1}^k \alpha_{kj} \alpha_{kl} \left( \frac{1}{j + l} \right)} \right)^{1/2} \]

\[ \varphi_k(z) = K_k \sum_{j=1}^k \alpha_{kj} e^{-jz}, \quad (\varphi_j(z), \varphi_k(z)) = \delta_{jk} \]
References


[Census] [http://www.census.gov/srd/www/x12a/](http://www.census.gov/srd/www/x12a/)


[Lando, D.] Lando, D. Credit Risk Modelling, Theory and Application, Princeton University Press.


Vita

Manish Mahajan

2011  Ph. D. in Electrical and Computer Engineering, Rutgers University

2002  M. S. in Electrical and Computer Engineering, Rutgers University

1997  B. E. in Electronics, VJTI, Mumbai University, Mumbai, India

2006-2010  Research and Modeling, Zais Group LLC, Red Bank, NJ.

2002-2006  Teaching assistant, Department of Chemistry and Chemical Biology, Rutgers University

1999-2002  Teaching assistant, Department of Electrical and Computer Engineering, Rutgers University

1997-1999  Systems Engineer, Router design, Cisco Systems, Chennai, India

1  Manish Mahajan, Manish Parashar, Managing QoS for Multimedia Applications in a Differentiated Services Environment, Journal of Network and Systems Management, July 2003
