# Pricing European options using artificial neural networks

**M Thulo**

**orcid.org 0000-0001-9253-6651**

Dissertation accepted in partial fulfilment of the requirements for the degree *Master of Science in Risk Analytics* at the North-West University

Supervisor: Dr ME Sonono

Co-supervisor: Prof HP Mashele

Graduation October 2023

39627594

# Declaration

I hereby declare that the research work `Pricing European options using artificial neural networks` is of my own originality. It had not been submitted to any other university or institution for examination purposes. All the sources consulted in this work are fully acknowledged and are completely found in the reference section. Furthermore, this research work is submitted in partial fulfilment of the requirements for the degree Master of Science at the Centre for Business Mathematics and Informatics, North-West University (Potchefstroom campus).

# Dedication

*To the pursuit of knowledge.*

*"The will of God will never take you where His grace cannot sustain you."*

Romans 12 vs 1-2, John 13 vs 13-34, Thessalonians 5 vs 18

# Acknowledgements

I would like to thank God for leading me through the difficult periods I had to go through in order to finish this course. I move in the brilliance of your mercy.

I also like to appreciate the The North-West University (Potchefstroom campus). The Faculty of Natural science and Agriculture, the Centre for Business Mathematics and Informatics and the NWU postgraduate fellowship. I appreciate the guidance and information you provided.

To both of my supervisors, Dr. Energy Sonono and Prof. Phillip Mashele. Prof. Mashele I am incredibly appreciative that you gave me the chance to take this course. Dr Energy Sonono, I appreciate the contributions you made to this work. Thank you for your time and effort. Your encouragement and admiration for honesty and diligence influenced me.

I would like to express my gratitude to my family for their unwavering love and support. I value all the efforts you have made to help me adjust to being away from home. To my parents for always supporting me wholeheartedly and big brother, this world is a little bit better because of you.

My stay in Potch was enjoyable and unforgettable thanks to the Financial Freedom Fighters. I thank you all for your encouragement and academic support.

# Abstract

The study's focus is to examine the pricing of European options using artificial neural networks. It aims to use the predictive powers of artificial neural networks to forecast option prices for European puts and calls when presented with options data. For this investigation, two artificial neural networks the multi-layer perceptron neural network and the radial basis function neural network were employed. Since option price data are not readily available, the study uses Monte Carlo method to generate the data that is required for European put and call options training of the two artificial neural networks. These models are developed, tested and trained using simulated data and are then used to predict option prices. Training capability and comparisons are tested using several performance measures which include $R^2$, MD, MAD, MSD and SMAPD. Numerical tests are used to estimate the generalisation capabilities of artificial neural networks. The findings from the neural network models numerical tests are compared with the analytical Black-Scholes model. Finally, what-if-analysis on the option prices as some of the option input parameters are varied is performed.

The key outcomes of the study are that both the MLP and RBF neural network are very accurate at approximating the option prices. A closer look at the findings indicated that the MLP was a better fit to the data as compared to the RBF. This might be alluded to the use of ReLU activation function in the MLP which provided better learning for the MLP model as compared to the Gaussian radial basis function that was used for RBF. Numerical tests were carried out using the trained neural network models to predict the option prices and the findings were compared to the analytical Black-Scholes model. The findings reveal that the option prices obtained using the MLP are not considerably different from those of the Black-Scholes analytical model. However, the RBF model produced some option prices with considerably sizeable values different from the analytical values. Thus, the MLP was better at predicting option prices in comparison to the RBF. The study proposes a future related investigation into pricing European options using a different neural network structure and

using a different analytical or numerical approach.

**Keywords:** Black-Scholes model, Monte Carlo methods, Artificial neural network, Radial basis function, Multi layer perceptron, Stochastic processes, Supervised learning, Performance measures, Machine learning.

# List of Acronyms

| | | |
|---|---|---|
| **ANN** | - | Artificial Neural Network |
| **ARCH** | - | Auto Regressive Conditional Heteroskedasticity |
| **B-S** | - | Black-Scholes |
| **CDF** | - | Cumulative Distribution Function |
| **CEV** | - | Constant Elasticity of Variance |
| **FNN** | - | Feedforward Neural Network |
| **GARCH** | - | Generalized Auto Regressive Conditional Heteroskedasticity |
| **GBM** | - | Geometric Brownian Motion |
| **MCS** | - | Monte Carlo Simulation |
| **MLPNN** | - | Multi Layer Perceptron Neural Network |
| **RBFNN** | - | Radial Basis Function Neural Network |
| **RELU** | - | Rectified Linear Unit |
| **RNN** | - | Recurrent Neural Network |
| **SV** | - | Stochastic Volatility |

# Contents

# List of Figures

# List of Tables

# 1. Introduction

## 1.1 Background

Black & Scholes (1973) derived the first and most famous formula which was designed to price European style options. The Black-Scholes model is one of the best models with high precision that is used for option pricing, (Anwar & Andallah 2018). The variables that determine an option's price are all taken into consideration by this formula, they are referred to as model parameters. It is acquired with the premise that the underlying asset price moves in a path that follows a Geometric Brownian motion with returns that follow a log-normaldistribution, (Bhattacharya 1980). The Black-Scholes model can be implementedreasonably fast. It uses a mathematical formula to determine the option price. It makes it a desirable alternative when quick responses are necessary. The methodology is adaptable enough to be used to price options that are written on various financial assets. It is not restricted to pricing options written on equities.

The Black-Scholes model, however, has limitations. This approach ignores any other potentially important factors such as changes in volatility and only takes into consideration the changes in the price of the underlying asset. One of many assumptions upon which this pricing model is built is the log-normal distribution of stock price returns. Inconsistencies are caused by major flaws in the log-normal assumption of asset price returns as shown by implied volatilities (skewness), term structures and implied volatility smiles. Once more, the model makes numerous unrealistic assumptions such as the efficiency of the market and also absence of opportunities for arbitrage and many others, (Janková 2018). The Black-Scholes model calculates option values when the option contract is due to be exercised, (Shinde & Takale 2012). This limits the model to the determination of prices of options that are exercised at maturity (European options) and offers no closed form solution to other options that can be

exercised before and up to expiry.

Some measures have been taken to reduce option pricing limitation of the Black-Sholes model. Researchers have presented several approaches, some not straying too far from the B-S model. Models have been derived in order to solve the issue of the assumption of constant volatility throughout the contract, stochastic volatility (SV) models were developed. According to stochastic volatility models, the underlying asset's volatility is modelled as a dynamic component (Taylor 1994). These types of models are inspired by the Black-Scholes model and incorporate some changes. They are improved by substituting the constant variance with a process that follows a Geometric Brownian motion or other stochastic processes. The type of process depends on the model being discussed. Models that incorporate stochastic volatility include the GARCH model, Heston model, Constant elasticity of variance (CEV) model and others. When there is a correlation between assets, the price of a European call or put option can be obtained using the Heston model. This criteria proposes an explicit stochastic volatility model. The constant elasticity of variance (CEV) model is a model of dynamic volatility that aims to incorporate stochastic volatility and the leverage effect, it was first introduced by (Cox & Ross 1976). Beckers (1980) and Carr & Linetsky (2006) studied this version. A statistical archetype called Generalized Auto Regressive Conditional Heteroskedasticity (GARCH) is used to calculate volatility that varies in time (Duan 1995). The heteroscedasticity of stock returns is intended to be captured by GARCH models, which are symmetric ARCH models. (Donaldson & Kamstra 1997). They presume that an autoregressive moving average process governs the variance of the error term. SV models are better than the B-S model because their theoretical assumptions are more grounded in reality. In comparison to the B-S model, empirical solutions from SV models offer superior prices for options.

However SV models are difficult to implement. They do not entirely remove pricing biases. Further improvements led to jump diffusion models. They were introduced in 1976 by Robert Merton. A more intricate stock price behaviour model that includes stochastic volatility as well as sporadic jumps is called a jump diffusion model. The

poisson process, which may be utilised to simulate systematic jumps caused by abrupt changes in stock price brought on by the receipt of substantial new information, serves as the foundation for the jump diffusion process. Unlike the Black-Scholes model, in real life practical applications jump diffusion models are not preferred because they are incomplete and hard to implement.

The reasonable price for the value of an option contract is dependent on the current estimation of the underlying asset's potential future movements. Researchers proposed a data driven model to prescribe this fair price. Option premiums can be viewed as the output of a function that takes in contracted terms as inputs, as such, machine learning algorithms can be applied to option pricing. Hutchinson, Lo & Paggio (1994) compared their results to the Black-Scholes methodology and took into account an artificial neural network as a method to approximate the option pricing function. ANNs are tools for processing information that is frequently used in prediction and categorization (Bennell & Sutcliffe 2003). Artificial neural networksare excellent atproviding context from ambiguous or complex information by identifying patterns or relationships. The features of the organic nervous system that allow for experiential learning and the application of lessons to novel circumstances served as their inspiration. ANNs are created to imitate the manner in which the brain learns as a consequence, these qualities of ANNs make them desirable for problems like option pricing.

## 1.2 Problem Statement

The Black-Scholes framework served as the foundation for the first option pricing models. This model attempts to describe the underlying asset's movement as a stochastic process and makes assumptions about the market and other parameters that influence this movement. Although such models have been largely studied and seen positive results, it still remains that their success depends on the chance that the

assumptions from which they are based apply to the real world dynamic, (Ghaziri, Elfakhani & Assi 2000*b*). This study investigates a different route to pricing options. A method of option pricing that determines the value of an option by learning patterns and relations in past market performances and creating a generalisation rule that can be used to similar type data is studied. This method makes use of machine learning techniques.

Computers may be taught to learn from data automatically using a technique called machine learning, (Mahesh 2020). Machine learning is a discipline of artificial intelligence that is founded on the notion that machines can learn from data, spot patterns and make judgments with little to no human involvement, (Wang, Ma & Zhou 2009). This study focuses on the application of machine learning technology known as artificial neural networks. ANNs (deep learning) came into emergence as a result of functional constraints in machine learning. Machine learning was incapable of handling data with several different variables. It was also unsuitable for object and image recognition, as well as processing high-dimensional data in general. This led to the development of deep learning, in which neural networks play a central role. As previously proposed, artificial neural networks generalise data, which means they do not rely on theoretical assumptions that are fundamental to stochastic option pricing models.

In the 90's there was a significant rise in the use of computers in different industries. The finance sector also bought into this as computers could deal with vast amounts of data faster and more accurately than analogue techniques. Since it does not require theoretical assumptions to make inferences for option prices, the use of computers specifically neural networks, completely changed the pricing dynamic. This is an advantage where the assumptions are not reflected in the real world. Neural networks are used in finance because they can reduce human intervention to a bare minimum and analyse data without the restrictive parametric assumptions that conventional model methods usesuch as log-normal returns or continuous sample paths. They can detect underlying patterns in multidimensional data and are relatively simple to

implement. Artificial neural networks can represent non-linear correlations in data, which makes them a contender for the best fit model for pricing options of any style. The sole input is historical data. The network can find connections between the input variables that characterise the underlying phenomena being mimicked.

Studies have been conducted on using artificial neural networks as a method of pricing options. Malliaris & Salchenberger (1996) derived a list of variables including historical volatilities from S&P100 and used a neural network model that yielded impressive results in forecasting future volatility in 1992. Hutchinson et al. (1994) considered a data driven approach as an alternative method of application to the price and delta-hedge S&P500 futures options from 1987 to 1991 using artificial neural networks. This approach showed that the non-parametric approach can be used in substitution when the parametric approach cannot be applied. Using real-world data, Kelly (1994) used a neural network to estimate the market value function for American put options non-parametrically, and they found that the neural network method may be used to value stocks whose value is difficult to determine. Hermann & Narr (1997) revealed how to use artificial neural networks to get underlyingpricing formulas from market data, and these formulas outperformed the Black-Scholes models by a wide margin. Lajbcygier & Connor (1997) assessed disparities in actual transaction prices and the Black-Scholes pricing model to demonstrate prediction for daily Australian SPI options data that spanned over the period from 1992 to 1994. Tsaih (1999) used the Black-Scholes formula in the evaluation of sensitivity analysis as a tool for reading knowledge embedded in artificial neural networks and discovered that the result from sensitivity analysis and Black-Scholes are consistent. Carelli, Silani & Stella (2000) investigated a procedure to select a network structure for modelling implied volatility and realised that feed forward neural networks training is a non-linear least squares problem and the main tools for non-linear regression analysis should be used. Sequential training and predictions were used to demonstrate that multilayer perceptrons may be taught with data arriving one at a time. Full estimates of one-step forecasts' probability density functions were determined using financial data (de Freitas, Ni-

ranjan & Gee 2000). Ghaziri, Elfakhani & Assi (2000*a*) used multilayer feed forward neural networks and fuzzy neuro networks to price S&P500 index call options and provided a contrast of the outcomes with those produced by the Black-Scholes, the former results showed to be superior.

Yao, Li & Tan (2000) provided a comprehensive study that forecast option prices of the Nikkei 225 index to determine that for volatility markets a neural network option pricing model yields better results as compared to the Black-Scoles model, also came to the determination that neural networks models are more suited for risk seeking investors. Dugas, Bengio, Bélisle, Nadeau & Garcia (2000) improved the regression of call options prices using new types of function classes that incorporated priori knowledge. Gencay & Qi (2007) indicated that Bayesian regularization outperforms neural networks and the Black-Scholes model in some of the years when it comes to pricing and delta-hedging errors for the daily S&P500index call options from January 1988 to December 1993. Meissner & Kawano (2001) used volatility input that is generated by GARCH approach to formulate an option pricing model and used different neural networks and proceed to rank the performances of the generated networks. Zapart (2002) compared the Black-Scholes model and the MLP neural network revealing that the neural network models outperformed the benchmarks that had been set in both pricing and hedging performance. The neural network was applied to the daily Swedish stock index call options from 1997 to 1999. In 1993 the work of Malliaris & Salchenberger (1996) involved builing a neural network model that learns the relationship between the financial input data and the option price from previous data and does not require the assumption of a distribution. Kelly (1994) used real data as input fora neural network to estimate the market valuation function for an American put option and found that the neural network approach may be used to evaluate securities whose values are not tractable. Kim & Kim (2019) studies in pricing of Apple's European call options used an ANN model that combines financial theory. This research will look at using artificial neural networks to price European options based on simulated data. It will perform a comparison of the performance of artificial

neural networks and the Black-Scholes analytical model and appropriate conclusions will be derived.

## 1.3 Aim of the study

This study aims to price European options using artificial neural networks.

## 1.4 Objective of the study

The objective of this study is to:

- To investigate the use of Black-Scholes model in pricing of European type options.
- To train two artificial neural networks towards pricing of European options. This study will be concerned with the Multi-Layer Perceptron Neural Network (MLPNN) and the Radial Basis Function Neural Network (RBFNN).
- To compare the results obtained from artificial neural networks to those of the original Black-Scholes model.

## 1.5 Motivation

Options do not readily have value, as such, investors need to have means to determine when options do have value and how to calculate that value. An analytical solution for European call options exists, but this solutions is obtained from the Black-Scholes model which is a parametric model. Approximate solutions that are closer to real

world dynamics are more appetising in order to avoid risky investments. The study explores an alternative way of pricing European options using simulated options data that incorporates past market trends. It consider the use of two different types artificial neural networks that is the radial basis function neural network and multilayer perceptron neural network.

## 1.6 Methods of investigation

For this study, the following methods of investigation will be implemented:

- The mathematical and financial concepts used throughout this study will be presented and discussed. These include introduction of stochastic calculus and its concepts as it relates to pricing of options.
- Simulation of data that will be used in the study using Monte Carlo methods will be explored.
- Explore the theory behind artificial neural networks. How they are suited to pricing problems and their general structural build up.
- The implementation of artificial neural network models used as a non-parametric alternative to pricing European options. This includes training ANN to value European put and call options and then test the trained models on out of sample data.
- Compare the results of the prices of the ANN models with the results of the Black-Scholes model.

## 1.7 Chapter Overview

**Chapter 1 Introduction:** This chapter gives an introduction to the topic of research, it outlines the background on the topic, describes the problems statement and motivates the research. This chapter also includes the aims and objectives of this study and briefly states the methods of investigation that will be employed through this research.

**Chapter 2 Mathematical Preliminaries:** This chapter gives technical tools and definitions that will be used throughout this study. These include the technical ground work for derivation of the Black-Scholes formula and the mathematical background of artificial neural networks.

**Chapter 3 Artificial Neural Networks:** This chapter discusses in detail theory of artificial neural networks, their architecture and the types of neural networks that will be used in this study.

**Chapter 4 Data and numerical tests:** This chapter presents the numerical findings of the study with the objective of exploring how the artificial neural networks perform in terms of pricing European put and call options as compared to the traditional Black-Scholes model.

**Chapter 5 Conclusions and Recommendations:** This chapter is a summary of the study, it discusses the successes and failures of our findings and makes recommendations for a future study.

# 2. Mathematical Preliminaries

This chapter presents some technical aspects (mathematical and financial) that give foundation to the study.

## 2.1 Definitions

### 2.1.1 Definition. Sigma algebra

The system $\mathbb{F}$ of subsets of $\Omega$ is said to be the $\sigma$-algebra associated with $\Omega$, if the following properties are fulfilled (Shreve et al. 2004):

- The empty set $\emptyset$ belongs to $\mathbb{F}$.
- For any set $A_n \in \mathbb{F}(n = 1, 2, ...)$ the countable union of elements in $\mathbb{F}$ belongs to the $\sigma$-algebra $\mathbb{F}$, as well as the intersection of elements in $\mathbb{F}$ : $\bigcup_{i=1}^{\infty} A_n \in \mathbb{F}$ $\bigcap_{n=1}^{\infty} A_n \in \mathbb{F}$.
- For any set $A \in \mathbb{F}$, its complement belongs to the $\sigma$-algebra $\overline{A} := \{\omega \in \Omega | \omega \in A\} \in \mathbb{F}$.

The sigma algebra is a collection of subsets of the set $\Omega$ of all possible outcomes of an experiment including the empty set $\emptyset$.

### 2.1.2 Definition. Probability Space

The triplet $(\Omega, \mathbb{F}, P)$ is called a probability space. Where $P$ is a given probability measure and $P$ has the property that it has a total mass equal to unity that is,

$$P(\Omega) = 1.$$

The underlying space $\Omega$ is often referred to as the sample space and the elements of the sigma-algebra $\mathbb{F}$ are called events (Shreve et al. 2004).

### 2.1.3 Definition. Filtered Probability Space

A filtered probability space is a set consisting of $(X, \mathbb{F}, P, \mathbb{F}_t)$, where $\mathbb{F}_t$ is a filtration which describes the collection of information for all times up to and including $t$, (Shreve et al. 2004). $\mathbb{F}_t$ mathematically describes an increasing sequence of $\sigma$-algebras which is defined on a given measurable space $(X, \mathbb{F})$. Thus, we have

$$\mathbb{F}_s \subset \mathbb{F}_t \subset \mathbb{F}_t \subset \mathbb{F},$$

for all $0 \leq s \leq t \leq T$.

### 2.1.4 Definition. Stochastic process

A stochastic process is a mathematical tool that describes a random phenomenon changing in time. Let $T$ be a subset of $[0, \infty)$ A family of random variables $\{X_t\}_{t \in T}$, indexed by T, is called a stochastic or random process. When $T = \mathbb{N}$ or $T = \mathbb{N}_0$, $\{X_t\}_{t \in T}$ is said to be a discrete-time process and when $T = [0, \infty)$, it is called a continuous-time process (Björk 2009).

### 2.1.5 Definition. Adapted Stochastic process

A stochastic processs is a collection of random variables $X(t), t \in T$ defined on a given filtered probability space. A continuous time process is considered if the random variable $X(t)$ is $\mathbb{F}_t$-measurable. Hence, the value of the random variable $X(t)$ can be completely explained by the information at time $t$ (Björk 2009).

### 2.1.6 Definition. Brownian motion

A real-valued stochastic process $\{B(t) : t \geq 0\}$ is called a (linear) Brownian motion with start in $x \in \mathbb{R}$ if the following holds (Björk 2009).

- $B(0) = x$.
- The process has independent increments i.e. for all times $0 \leq t_1 \leq t_2 \leq ... \leq t_n$ the increments $B(t_n) - B(t_{n-1}), B(t_{n-1}) - B(t_{n-2}), ..., B(t_2) - B(t_1)$ are independent random variables.
- For all $t \geq 0$ and $h > 0$, the increments $B(t+h) - B(t)$ are normally distributed with expectation zero and varience $h$ almost surely, the function $t \to B(t)$ is continuous.

$\{B(t) : t \geq 0\}$ is otherwise referred to as a standard Brownian motion if $x = 0$.

### 2.1.7 Definition. Wiener process

A stochastic process $W$ is called a Wiener process if the following conditions hold (Björk 2009).

- $W(0) = 0$.
- The process W has independent increments, i.e. if $r < s \leq t < u$ then $W(u) - W(t)$ and $W(s) - W(r)$ are independent stochastic variables.
- For $s < t$ the stochastic variable $W(t) - W(s)$ has the Gaussian distribution $N(0, \sqrt{t-s})$.
- $W$ has continuous trajectories.

### 2.1.8 Definition. Stochastic differential equation

For a stochastic process $X(t)$ a stochastic differencial equation is an equation which consists of a combination of a deterministic term and a stochastic (white noise) term (Björk 2009). This is a differential equation of the form,

$$dX_t = f(X_t, t)dt + g(X_t, t)dW_t, \tag{2.1.1}$$

where $t \in [t_0, T]$, $T > 0$. The adapted processes, $f$ and $g$ are termed drift and diffusion coefficient functions respectively.

### 2.1.9 Definition. Ito process

An $n$-dimensional Ito process is a process that satisfies.

$$dX_t = a(t, X_t)dt + b(t, X_t)dWt, \tag{2.1.2}$$

where $W$ is an $m$-dimensional standard Brownian motion for some number $m$, $a$ and $b$ are $n$-dimensional and $n \times m$-dimensional adapted process, respectively, $a(t, X_t)$ is the drift, $b(x, X_t)$ is the standard deviation (Björk 2009).

$X_t$ is the solution to such a differential equation if it satisfies,

$$X_t = X_0 + \int_0^t a(s, X_s)ds + \int_0^t b(s, X_s)dWs, \tag{2.1.3}$$

where $X_0$ is a constant.

### 2.1.10 Theorem. *Ito's formula*

Assume that the process $X$ has a stochastic differential given by (Shreve et al. 2004).

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dWt, \tag{2.1.4}$$

where $\mu$ and $\sigma$ adapted processes,and let $f$ be a $C^{1,2}$-function. Define the process $Z$ by $Z_t = f(t, X_t)$. Then $Z$ has a stochastic differential given by

$$df(X_t) = \left(\frac{\partial f}{\partial t}(X_t) + \mu\frac{\partial f}{\partial x}(X_t) + \frac{1}{2}\sigma^2\frac{\partial^2 f}{\partial x^2}(X_t)\right)dt + \sigma\frac{\partial f}{\partial x}(X_t)dW_t. \tag{2.1.5}$$

### 2.1.11 Lemma. Ito's lemma

Suppose $f \in l^2$. Then with probability 1, for all $t \geq 0$ then $f$ is also an Ito process with its differential given by;

$$df(X_t) = \left(\mu_t \frac{\partial f}{\partial x}(X_t) + \frac{1}{2}\sigma_t^2 \frac{\partial^2 f}{\partial x^2}(X_t)\right) dt + \sigma_t \frac{\partial f}{\partial x}(X_t) dW_t. \tag{2.1.6}$$

**2.1.12 Proposition. Ito's formula**

With assumptions as in 2.1.6, $df$ is given by

$$df = \frac{\partial f}{\partial t} dt + \frac{\partial f}{\partial x} dX + \frac{1}{2}\sigma^2 \frac{\partial^2 f}{\partial x^2} dX + \sigma \frac{\partial f}{\partial x} (dX)^2. \tag{2.1.7}$$

where we use the following formal multiplication table

$$\begin{cases} (\mathrm{d}t)^2 & = 0\,, \\ \mathrm{d}t.\mathrm{d}W & = 0\,, \\ (\mathrm{d}W)^2 & = \mathrm{d}t\,. \end{cases}$$

## 2.2 Theory of option pricing

This section presents the theory of option pricing. For further information, one is referred to (Björk 2009).

**2.2.1 Definition. Derivative**

This is a contract or product whose value is determined by the value of an underlying asset, such as a commodity, cash, or security (Björk 2009).

**2.2.2 Definition. Derivative contracts**

Derivative contracts or contingent claims are contracts based on the underlying asset $X_t$.

When a contract is bought, it is said to be in a long position, and when it is sold, it is in a short position (Björk 2009).

### 2.2.3 Definition. Arbitrage

Arbitrage is the act of buying securities in one market and selling them in another to make a profit from unjustifiable price differences in the markets. This is otherwise described as a free lunch, where profit is made with no risk (Björk 2009).

### 2.2.4 Definition. Principle of no arbitrage

The principle of no arbitrage asserts that there are no arbitrage opportunities (Björk 2009).

### 2.2.5 Definition. Portfolio

A portfolio is a combination of different financial assets or securities (Björk 2009).

### 2.2.6 Definition. Self-financing Portfolio

A self-financing portfolio in which there is no withdrawal out of or deposit into the portfolio, that is the purchase of a new asset is financed by the sale of another.

Let the $N$-dimensional price process $f(S_t); t \geq 0$ be given.

1. A portfolio strategy is any ${\mathbb{F}_s}^t$-adapted $N$-dimensional process $f(h_t); t \geq 0$.
2. The portfolio $h$ is said to be Markovian if it is of the form

$$h(t) = h(t; S(t)),$$

for some function $h : R_+ \times R^N \to R^N$ The value process $V^h$ corresponding to the portfolio $h$ is given by

$$V^h(t) = \sum_{i=0}^{N} h_i(t) S_i(t).$$

A portfolio is called self-financing if the value process $V^h$ satisfies the condition

$$dV^h(t) = \sum_{i=0}^{N} h_i(t) dS_i(t).$$

An **arbitrage possibility** on a financial market is a self-financed portfolio $h$ such that

- $V^h(0) = 0$,
- $P(V^h(T) \geq 0) = 1$,
- $P(V^h(T) > 0) > 0$.

The market is arbitrage free if there are no arbitrage possibilities (Björk 2009).

### 2.2.7 Definition. European call option

A European call option with strike price $K$ and time of maturity $T$ on the underlying asset $S$ is a contract defined by the following.

- The holder of the option has at time $T$, the right to buy the underlying stock at the price $K$ from the underwriter of the option.
- The holder of the option is in no way obliged to buy the underlying stock.

- The right to buy the underlying stock at the price $K$ can only be exercised at the precise time $T$.

### 2.2.8 Definition. European put option

A European put option with strike price $K$ and time of maturity $T$ on the underlying asset $S$ is a contract defined by the following.

- The holder of the option has, at time $T$, the right to sell the underlying stock at the price $K$ to the underwriter of the option.
- The holder of the option is in no way obliged to buy the underlying stock.
- The right to sell the underlying stock at the price $K$ can only be exercised at the precise time $T$.

### 2.2.9 Definition. Payoff of an option

Payoff represents the value of the option at the date of maturity $T$. The payoff graphs for the call and put options are shown in Figure 2.1:

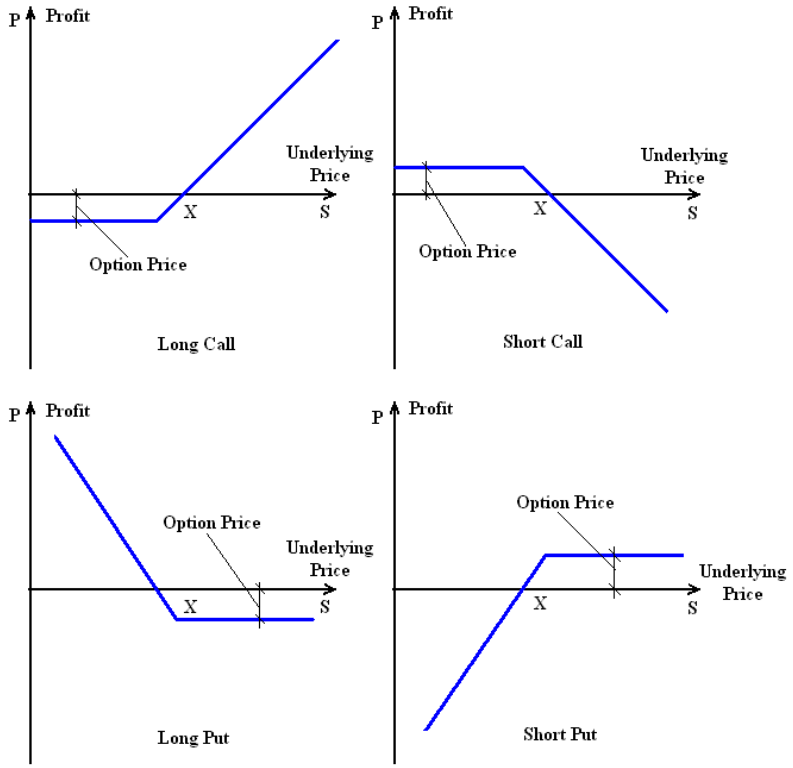


Figure 2.1: Payoff diagrams for call and put options

### 2.2.10 Definition. Contingent claims

A contingent claim with expiry $T$ is any stochastic variable $X \in \mathbb{F}_T^S$. $X$ is a simple claim if it is of the form

$$X = \Phi(S(T)).$$

The function $\Phi$ is the contract function.

The European call option is a simple contingent claim such that

$$\Phi(X) = max[S(T) - K, 0].$$

The European put option is a simple contingent claim such that;

$$\Phi(X) = max[K - S(T), 0].$$

The function $\Phi(S(T))$ describes the pay-off for the European option at maturity similarly the pay off for the European put option is $max[K - S(T), 0]$.

### 2.2.11 Factors affecting option prices

The following are the main factors that affect option prices.

1. **The underlying asset price**: The price of the put option and the call option are impacted by the underlying share price in distinct ways. In the case of a call option, the amount needed to purchase the call option written on the asset increases in proportion to the price of the asset, and vice versa. For the put option issued on the asset, the inverse is true.

2. **The strike price** : A high strike price would result in a low call option premium and a high put option premium.

3. **Time to expiry**: Both the put and call premiums decline as expiration comes closer. As a result, both put and call option premiums would be high for options with a longer term until expiration.

4. **Volatility**: Both the call and put option premiums are high when the underlying asset is volatile because they are bound below. This means that the downside risk of fluctuating stock prices is mitigated through options.

5. **Risk-free rate of interest**: Money invested in a risk-free bond is affected by the market's risk-free rate of interest. The value of the underlying stock need not be paid in full when purchasing an option. Instead, this sum can be invested

at a risk-free rate of return in a risk-free bond. The price of the call rises as the risk-free rate of interest does in order to maintain an arbitrage-free portfolio. On the other hand, if the risk-free rate of interest rises, the price of the put option values decline.

6. **Dividends**: Since every dividend payment lowers the value of the underlying asset which we have previously mentioned as having a positive impact on the call price, the call option premium drops if we assume that the asset pays dividends throughout the term of the option. The cost of the put option in this situation rises in a similar way.

## 2.3 Martingale Theory

### 2.3.1 Definition. Martingale

An integrable process $X_t$ where $0 \leq s \leq t$, is a martingale with respect to the filtration $\mathbb{F}_t$,

$$E[X_t|\mathbb{F}_s] = X_s, \tag{2.3.1}$$

in the sequel,

$$L^2(\Omega) := \{\mathbb{F} : \Omega \to R : E[|\mathbb{F}_s|^2] < \infty\}, \tag{2.3.2}$$

denotes the space of square- integrable random variables (Björk 2009).

### 2.3.2 Theorem. *Randon-Nikodym*

Let $P$ and $Q$ be probability measures on $(\Omega, \mathbb{F})$. Then the following are equivalent:

1. Q << P.

2. There exists an $\mathbb{F}$-measurable function $\phi \leq 0$ with $E[\phi] = 1$ such that

$$Q[A] = \int_A \phi dP = E[\phi 1_A], \text{for all A} \in \mathbb{F} .$$

The function $\phi$ is called density or Randon-Nikodym derivative and is often denoted by $\phi(\omega) = \frac{dQ}{dP}(\omega)$ .

**2.3.3 Definition. Equivalent measures**

Let P and Q be two probalility measures on $(\Omega, \mathbb{F})$. Q is equivalent to P if $Q << P$ and $P << Q$, that is

$$P[A] = 0 \Leftrightarrow Q[A] = 0, \text{for all A} \in \mathbb{F} .$$

## 2.4 Geometric Brownian Motion

**2.4.1 Definition. Geometric Brownian Motion**

The Black-Scholes model for pricing options assumes that the stock price follows a Geometric Brownian Motion (GBM). That is for the stock price $S(t)$ the dynamics are as follows.

$$dX_t = \alpha X_t dt + \sigma X_t dWt,$$

$$X_0 = x_0.$$

The solution to the corresponding linear deterministic function is an exponential function of time

$$X_t = x_0 e^{\alpha t},$$

and for small $\sigma$ the graph of the GBM is close to that of the exponential function above.

We now investigate a process $Z$ defined by $Z_t = lnX_t$. The Ito formula gives

$$dZ = \frac{1}{X}dX + \frac{1}{2}\left\{\frac{-1}{X^2}\right\}[dX]^2,$$

thus substituting $dX_t$ we get

$$dZ_t = \left(\alpha + \frac{1}{2}\sigma^2\right)dt + \sigma dWt,$$

$$Z_0 = lnx_0,$$

and can be integrated directly to give:

$$Z_t = lnx_0 + (\alpha + \frac{1}{2}\sigma^2)dt + \sigma dWt,$$

$$X_t = x_0 exp(\alpha + \frac{1}{2}\sigma^2)dt + \sigma dWt,$$

the expectation of $X_t$ is given by:

$$E[X_t] = x_0 e^{\alpha t}.$$

Suppose $c$ is a European call option with strike price $K$ and expiry $T$ on an underlying asset $S$. We assume that $S$ follows a Geometric Brownian Motion under the real world probability measure $P$. That is for constants $\mu$ and $\sigma$ and $W_t \sim N(0,t)$,

$$dS_t = S_t(\mu dt + \sigma dW_t). \tag{2.4.1}$$

The payoff of the call is given by $c_T = max(S_t - K, 0)$. The present value of $c$ is obtained by calculating the discounted expectation of its payoff under the equivalent martingale measure $Q$:

$$c_0 = e^{-rT}E_Q[max(S_t - K, 0)]. \tag{2.4.2}$$

## 2.5 Black-Scholes model

Black and Scholes developed a technique for estimating the precise value of European options under particular assumptions in a Nobel Prize-winning article. It is critical to first investigate the basic framework that has led to a knowledge of the pricing process (Björk 2009) and (Shreve et al. 2004).

**2.5.1 Definition. Black-Scholes PDE**

Let's assume that the option's value is $f(t, St)$, the stock return's standard deviation is $\sigma$, and the risk-free interest rate is $r$. Consequently, the following partial derivative equation can be used to explain the price of an option over time (Ye 2013).

$$rS_t\frac{\partial f}{\partial S} + \frac{\partial f}{\partial t} + \frac{1}{2}\sigma^2\frac{\partial^2 f}{\partial S^2} = rf. \tag{2.5.1}$$

Next, the derivation of the Black-Scholes PDE is shown.

### 2.5.2 Deriving the Black-Scholes formula using the no-arbitrage method

Assume that the stock price $S$ follows a geometric Brownian motion so that,

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{2.5.2}$$

where $W_t$ is a standard Brownian motion, $\mu$ and $\sigma$ are known constants. to derive the option pricing formula, Shreve et al. (2004) made some assumptions. The creation of a riskless portfolio and the non-arbitrage argument are key components of the B-S technique. The process is as follows.

Suppose that $f$ is the price of a call option or other derivative contingent on $S$. By Ito's lemma,

$$df = \left(\frac{\partial f}{\partial S}\mu S + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\right) dt + \frac{\partial f}{\partial S} dWt. \tag{2.5.3}$$

Lets then establish a portfolio that consists of a long position in $\Delta$ stock units and a short position in a call option. The portfolio's value is defined as $\Pi$ and it is given by

$$\Pi = -f + \Delta S. \tag{2.5.4}$$

The change in the value of this portfolio in a small time interval is given by

$$d\Pi = -df + \Delta dS. \tag{2.5.5}$$

Substituting 2.5.3 into 2.5.5 yields

$$\begin{aligned} d\Pi &= -\left(\frac{\partial f}{\partial S}\mu S + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\right) dt - \frac{\partial f}{\partial S}\sigma dWt + \Delta\mu S dt + \Delta\sigma S dW_t \\ &= \left(-\frac{\partial f}{\partial S}\mu S - \frac{\partial f}{\partial t} - \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2 + \Delta\mu S\right) dt + \left(-\frac{\partial f}{\partial S}\sigma S + \Delta\sigma S\right) dW_t. \end{aligned} \tag{2.5.6}$$

To make the portfolio riskless, choose $\Delta = \frac{\partial f}{\partial S}$. Then,

$$d\Pi = \left(-\frac{\partial f}{\partial t} - \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\right) dt. \tag{2.5.7}$$

On the contrary, this risk-free portfolio must generate a risk-free rate, $r$, in the absence of arbitrage opportunities.

$$d\Pi = r\Pi dt. \tag{2.5.8}$$

Substituting from 2.5.7 and 2.5.5, this becomes

$$\left(-\frac{\partial f}{\partial t} - \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\right) dt = r\left(-f + \frac{\partial f}{\partial S}S\right) dt, \tag{2.5.9}$$

or

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 f}{\partial S^2} = rf \tag{2.5.10}$$

Equation 2.5.10 is the well-known Black-Scholes partial differential equation. The answer is determined by the boundary conditions. In the case of a European call option, the final condition is that the option price is just its payoff at maturity.

$$f = max(S - K, 0), t = T. \tag{2.5.11}$$

A closed form solution for the European call option can be obtained through solving the PDE 2.5.10. The European call option pricing formula is given as follows considering $\Phi$ is the cumulative probability distribution of the standard deviation.

$$c = S_0\Phi(d_1) - e^{-r(T)}K\Phi(d_2), \tag{2.5.12}$$

where,

$$d_1 = \frac{ln(S_0/K) + (r + \sigma^2/2)(T)}{\sigma\sqrt{T}}, \tag{2.5.13}$$

and

$$\begin{aligned} d_2 &= \frac{ln(S_0/K) + (r - \sigma^2/2)T}{\sigma\sqrt{T}} \\ &= d_1 - \sigma\sqrt{T}. \end{aligned} \tag{2.5.14}$$

Now that an equation for the price of the call option has been derived, we make use of the relation between the call and the put, known as put call parity to evaluate the put option price under the Black-Scholes model. Consider $V_A^t$ and $V_A^t$ to be the value for portfolio A and portfolio B at time $t$ and at time $T$ (expiry).

Portfolio A consist of the pay-off of a call option and the amount of cash $K$, that is,

$$V_A^T = max(S_T - K, 0) + K, \tag{2.5.15}$$

and Portfolio B consists of the value for the payoff of the put option and the stock at expiry that is,

$$V_B^T = max(K - S_T, 0) + S_T, \tag{2.5.16}$$

comparing the two portfolios it can be observe that:

$$\begin{aligned} max(S_T - K, 0) + K &= max(S_T, K), \\ max(K - S_T, 0 + S_T &= max(K, S_T), \\ V_A^T &= V_B^T. \end{aligned} \tag{2.5.17}$$

Then by the principle of no arbitrage if $V_A^T = V_B^T$, then $V_A^0 = V_B^0$ at time $t = 0$. The value of portfolio A consists of one call option and a discounted amount of cash,

$$V_A^0 = c_0 + Ke^{-rT}. \tag{2.5.18}$$

The value of portfolio B consists of one put option and one unit of stock,

$$V_B^0 = p_0 + S_0. \tag{2.5.19}$$

Now by principle of no arbitrage,

$$\begin{aligned} V_A^0 &= V_B^0, \\ c_0 + Ke^{-rt} &= p_0 + S_0. \end{aligned} \tag{2.5.20}$$

Adapting this relation to the Black-Scholes model, we have that

$$c_t + e^{-r(T-t)}K\phi(d_2) = p_t + S_t\phi(d_1). \tag{2.5.21}$$

Now the put option price is given by

$$p_t = c_t + e^{-r(T-t)}K\phi(d_2) - S_t\phi(d_1). \tag{2.5.22}$$

**2.5.3 Definition. Put-Call Parity**

For a European call option, $c$, and a European Put option, $p$, with the same strike price $K$, and maturity $T$, put-call parity states:

$$Ke^{-r(T-t)} + c = S + p. \tag{2.5.23}$$

## 2.6 The Greeks

The derivatives of the pricing equation, with respect to the Black-Scholes model parameters are called the Greeks. The Greeks are sensitivity measures for the price

of the option. They are bounded between zero and one for call options and zero and negative one for put options. Hedging depends on the efficient management of the Greeks Horasanl (2008).

### 2.6.1 Proposition. Greeks

For a European call option, $c$, with strike price $K$ and maturity $T$. The following relations for the Greeks are established (Que 2019) . The letter $\Phi$ denotes the density function of the standard normal distribution, $\Phi \sim \mathcal{N}(0,1)$. The following relations are established by taking derivatives of the Black-Scholes call option formula with respect to the corresponding parameters (Chen 2020).

$$\begin{aligned}
\Delta &= \frac{\partial c}{\partial S_t} = \Phi(d_1) \\
\Gamma &= \frac{\partial^2 c}{\partial S_t^2} = \frac{\phi(d_1)}{s\sigma\sqrt{T-t}} \\
\rho &= \frac{\partial c}{\partial r} = K(T-t)e^{-r(T-t)}\Phi(d_2) \\
\Theta &= \frac{\partial c}{\partial t} = -\frac{s\phi(d_1)\sigma}{2\sqrt{T-t}} - rKe^{-r(T-t)}\Phi(d_2) \\
\upsilon &= \frac{\partial c}{\partial \sigma} = s\phi(d_1)\sqrt{T-t}
\end{aligned} \tag{2.6.1}$$

#### Delta

From 2.6.1 The Greek Delta measures sensitivity of the option price with respect to the change in the underlying stock price. It is the slope of the curve that connects the option price to the underlying asset.

$$\begin{aligned}\Delta &= \frac{\partial c}{\partial S_t} \\ &= \frac{\partial S_t \Phi(d_1) - e^{-r(T-t)} K \Phi(d_2)}{\partial S_t} \\ &= \Phi(d_1) + S_t \frac{\partial \Phi(d_1)}{\partial d_1}\frac{\partial d_1}{\partial S_t} - K e^{-r(T-t)} \frac{\partial \Phi(d_2)}{\partial d_2}\frac{\partial d_2}{\partial S_t} \\ &= \Phi(d_1) + S_t \Phi^{'}(d_1)\frac{1}{S_t \sigma \sqrt{T-t}} - K e^{-r(T-t)} \Phi^{'}(d_2)\frac{1}{S_t \sigma \sqrt{T-t}}\end{aligned} \tag{2.6.2}$$

$$\Phi^{'}(x) = \frac{1}{\sqrt{2\Pi}} e^{-\frac{x^2}{2}} \tag{2.6.3}$$

$$\begin{aligned} ln(S_t/K) + (r + \frac{1}{2}\sigma^2)(T-t) &= d_1 \sigma \sqrt{T-t}, d_2 \\ &= d_1 - \sigma\sqrt{T-t} \\ \Rightarrow ln(S_t/K) r(T-t) &= d_1 \sigma \sqrt{T-t} - \frac{1}{2}\sigma^2 (T-t) \\ &= -\frac{1}{2}(\sigma\sqrt{T-t} - d_1)^2 + \frac{1}{2} d_1^2 \\ \Rightarrow ln(S_t/K) - \frac{1}{2} d_1^2 &= -\frac{1}{2}(-d_2)^2 - r(T-t) \\ &= -\frac{1}{2}(d_2)^2 - r(T-t) \\ \Rightarrow \frac{S_t}{K} e^{-\frac{1}{2}d_1^2} e^{-r(T-t)} & \\ \Rightarrow S_t \Phi^{'}(d_1) &= K e^{-r(T-t)} \Phi^{'}(d_2) \end{aligned} \tag{2.6.4}$$

Therefore, the equation 2.6.3 can be simplified as

$$\begin{aligned}\Delta &= \Phi(d_1) + \frac{1}{S_t\sigma\sqrt{T-t}}(S_t\Phi^{'}(d_1) - Ke^{-r(T-t)}\Phi^{'}(d_2)) \\ &= \Phi(d_1)\end{aligned} \tag{2.6.5}$$

**Gamma**

Gamma $\Gamma$ is the second derivative of the option's price relative to the underlying asset (the derivative of Delta with respect to the stock price). It is used to measure the size of hedging errors or changes. This means that small gamma indicates a small change and large gamma indicates a rapid change.

$$\begin{aligned}\Gamma &= \frac{\partial\Delta}{\partial S_t} \\ &= \Phi^{'}(d_1)\frac{\partial d_1}{\partial S_t} \\ &= \frac{1}{\sqrt{2\Pi}}e^{\frac{-d_1^2}{2}}\frac{1}{S_t\sigma\sqrt{T-t}}\end{aligned} \tag{2.6.6}$$

**Rho**

$\rho$ is the rate of change of the option price with respect to the interest rate it is used to measure the sensitivity of the option prices relative to to the change of the risk

free interest.

$$\begin{aligned}\rho &= \frac{\partial c}{\partial r} \\ &= S_t\Phi^{'}(d_1)\frac{\partial d_1}{\partial r} - (Ke^{-r(T-t)}(-(T-t))\Phi(d_2) + Ke^{-(T-t)}\Phi^{'}(d_2)\frac{\partial d_2}{\partial r} \\ &= S_t\Phi^{'}(d_1)\frac{\partial d_1}{\partial r} + (T-t)Ke^{-r(T-t)}\Phi(d_2) - Ke^{-r(T-t)}\Phi^{'}(d_2)\frac{\partial(d_1 - \sigma\sqrt{T-t}}{\partial r} \\ &= \frac{\partial d_1}{\partial r}(S_t\Phi^{'}(d_1) - Ke^{-r(T-t)}\Phi^{'}(d_2)) + (T-t)Ke^{-r(T-t)}\Phi(d_2) \\ &= (T-t)Ke^{-r(T-t)}\Phi(d_2)\end{aligned} \tag{2.6.7}$$

**Theta**

Theta is the measure of sensitivity of the price of the option with respect to the evolution of time. It describes the rate of change of the value of the option as the length of the contract changes. It is otherwise known as the time decay of the option value. Theta assumes a negative value for an option because the option contract becomes less valuable as it approaches expiry. Let $\tau = T - t$ then,

$$
\begin{aligned}
\Theta &= \frac{\partial c}{\partial t} \\
&= -\frac{\partial c}{\partial \tau} \\
&= -(S_t\Phi^{'}(d_1)\frac{\partial d_1}{\partial r} - (Ke^{-r\tau}(-r)\Phi(d_2) + Ke^{-r\tau}\Phi^{'}(d_2)\frac{\partial d_2}{\partial r})) \\
&= -(rKe^{-\tau}\Phi(d_2) + S_t\Phi^{'}(d_1)\frac{\partial d_1}{\partial \tau} - Ke^{-r\tau}\Phi^{'}(d_2)\frac{\partial (d_1 - \sigma\sqrt{\tau})}{\partial \tau} \\
&= -(rKe^{-r\tau}\Phi(d_2) + \frac{d_1}{\partial \tau}(s_t\Phi^{'}(d_1) - Ke^{-r\tau}\Phi^{'}(d_2)) + Ke^{-r\tau}\Phi^{'}(d_2)\frac{\sigma}{2\sqrt{\tau}}) \\
&= (rKe^{-r\tau}\Phi(d_2) + S_t\Phi^{'}(d_1)\frac{\sigma}{2\sqrt{\tau}}) \\
&= -\frac{S_t\sigma}{2\sqrt{(T-t}}\frac{1}{\sqrt{2\Pi}}e^{-\frac{d_1^2}{2}} - rKe^{-(T-t)}\Phi(d_2)
\end{aligned}
\tag{2.6.8}
$$

**Vega**

Vega is used to measure the sensitivity of option prices relative to the change of volatility of the underlying asset. The assumption so far is that volatility of asset movements is constant but in reality volatility changes with change in time.

$$
\begin{aligned}
\upsilon &= \frac{\partial c}{\partial \sigma} \\
&= S_t\Phi^{'}(d_1)\frac{\partial d_1}{\partial \sigma} - Ke^{-r(T-t)}\Phi^{'}(d_2)\frac{\partial d_2}{\partial \sigma} \\
&= S_t\Phi^{'}(d_1)(\sqrt{T-t} - \frac{d_1}{\sigma}) - Ke^{-r(T-t)}\Phi^{'}(d_2)(\sqrt{T-t} - \frac{d_1}{\sigma} - \sqrt{T-t}) \\
&= -\frac{d_1}{\sigma}(S_t\Phi^{'}(d_1) - Ke^{-r(T-t)}\Phi^{'}(d_2)) + S_t\Phi^{'}(d_1)\sqrt{T-t} \\
&= S_t\Phi^{'}(d_1)\sqrt{T-t} \\
&= S_t\sqrt{T-t}\frac{1}{\sqrt{2\Pi}}e^{\frac{-d_1^2}{2}}
\end{aligned}
\tag{2.6.9}
$$

Generally the Greeks are used as sensitivity measures for the option price and as such they can be used to avoid risk.

## 2.7 The Monte Carlo framework

### 2.7.1 Definition. Random Variable

A random variable is a real valued function defined on a sample space that assigns a real value, $X$, for each elementary outcome.

### 2.7.2 Definition. Expectation

Mathematical expectation also known as the expected value is the generalisation of the weighted average or otherwise defined as a product of the probability of an event occurring and the outcome of the event.

### 2.7.3 Definition. Law of large numbers

According to the law of large numbers, the average findings of an experiment can be used as an unbiased estimator of the expected value if the experiment is repeated independently a large number of times.

### 2.7.4 Definition. The central limit theorem

Let $X_1, X_2, ..., X_n$ be independent identically distributed random variables with expected value $EX_i = \mu < \infty$ and variance $0 < Var(V_i) = \sigma^2 < \infty$. Then, the random variable

$$Z_n = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} = \frac{X_1 + X_2 + ... + X_n - n\mu}{\sqrt{n}\sigma} \tag{2.7.1}$$

When $n$ boundlessly increases, the distribution eventually converges to that of the typical normal random variable that is,

$$\lim_{n\to\infty} P(Z_n \geq x) = \Phi(x), \text{ for all } x \in \mathbb{R} \tag{2.7.2}$$

where $\Phi(x)$ is the standard normal CDF.

### 2.7.5 Definition. Monte Carlo Methods

Monte Carlo simulations or Monte Carlo methods are one class of three major classes of simulations models that solves problems by the use of randomly generated numbers. Monte Carlo methods generate independent, identically distributed random samples $X_1, X_2, ..., X_n$ from the distribution of $X$. and estimate $E[h(X)]$ by the corresponding sample average.

Suppose we want to integrate a one dimensional function $f(x)$ from $a$ to $b$.

$$F = \int_a^b f(x)dx \tag{2.7.3}$$

By averaging a sample of the function $f$ at consistent random locations over the interval, we may approximate this integral. If there are $N$ random variables and $X_i \in [a, b)$ with a corresponding PDF of $\frac{1}{(b-a)}$, then using the Monte Carlo estimator to calculate $F$ gives,

$$\hat{F} = (b-a)\frac{1}{N-1}\sum_{i=0}^{N} f(X_i) \tag{2.7.4}$$

### 2.7.6 Simulating asset paths

- The first step in pricing European options with Monte Carlo methods is to generate a large number of potential values of the underlying asset at the time of expiry. To calculate these values we assume that the underlying asset price, $S$, follows a Geometric Brownian motion that is,

$$dSt = \mu St + \sigma StdZt \tag{2.7.5}$$

with the solution

$$S_T = S_0 \exp\left[(\mu - \frac{1}{2}\sigma^2)T + \sigma\sqrt{T}Z_T\right] \tag{2.7.6}$$

for the asset price. Where $Z_T$ is a random number sampled from the standard normal distribution. Sample paths are created with $n$ such random numbers.

Parameters are described as those of the Black-Scholes model ($\mu$ is the risk free rate of return).

1. The stock price, $S$.
2. The exercise price, $K$.
3. The riskfree interest rate,$\mu$.
4. The volatility, $\sigma$.
5. The time, $T$.

- Now calculate the payoffs for each sample path at the expiry using the formulas.

$$P_c = max(S_T - K, 0) \tag{2.7.7}$$

payoff $P_p$ for options

$$P_p = max(k - S_T, 0) \tag{2.7.8}$$

- We create paths for $S_T$ by numerically solving the equation above with random parameter inputs.
  - Option values are equal to the product of the discounting factor $exp(-rT)$ and the average of the payoffs generated. Payoff for call options is given by

$$c_0 = exp(-rT)average(max((S_T)_i - K, 0) \tag{2.7.9}$$

    while payoff for put options is given by

$$p_0 = exp(-rT)average(max(K - (S_T)_i), 0) \tag{2.7.10}$$

    where $i = 0, 1, 2, \ldots, n$ and $n$ is the number of random paths $(S_T)_i$ is the $i$th price for the asset at time $T$ generated by the $i$th path.
  - The number of payoffs is determined by the number of random paths generated.

# 3. Artificial Neural Networks

## 3.1 Intoduction to artificial neural networks

Artificial neural networks are data structures made up of highly interconnected processing elements called nodes. These models are also known as deep neural networks and were popularised after the 1980's to solve problems such as image recognition, speech recognition and other approximation problems in general (Jang & Lee 2018). Artificial Neural Networks before today had to go through stages of development. In the forties, ANNs had their first peak when McCulloch and Pitts developed a computational unit that mimicked a biological neuron (Andina, Pham, Andina, Vega-Corona, Seijas & Torres-Garcìa 2007). This was the first step towards the Neural Networks that we know today.

During the sixties there was a second peak in ANNs, that was the development of the Rosenblatt's perceptron (Rosenblatt 1960). This is a binary single neuron that can solve linear classification problems that uses simple learning algorithms. As it can be deduced this model cannot solve non-linear problems. This limitation of the Rosenblatt's perceptron was part of the work of (Marvin & Seymour 1969). This slowed down the trajectory of interest in ANNs. However the eighties brought about renewed interest with the Hopfield network, Hopfield (1982) that consisted of one or more fully connected recurrent neurons. Also this was the time when the back-propagation algorithm for multilayer networks was proposed and hence became popular. ANNs are considered to be universal functional appropriators of non-linear functions (Yadav 2018). They are distinctive for their ability to learn and adapt to patterns in data. One of the most appealing characteristics of neural networks is their capacity for learning. This distinguishing characteristic of the neural network divides networks into various groups, each of which learns in a different manner.

The fundamental component of any neural network is the neuron, which was modelled after brain neurons. In the brain, neurons transmit electrical impulses from one end to another. The human brain has millions of these neurons which makes the complexity very hard to replicate (Han, Kim, Kim & Youn 2018). However, artificial neural networks are designed to capture the basic operations, which has been achieved by the development of a perceptron. In neural networks, neurons receive input data, perform some process and give an output. The process that leads to an output is performed by a function called an activation function. Activation functions are an integral part of an artificial neural network. They are non-linear functions that introduce non-linearity and compute complex relations in the data. An artificial neural network can be considered as a linear regression model in the absence of activation functions. A weighted summation of the inputs and a bias term make up the argument of the activation function. The weighted sum of the inputs and the output are both adjusted by a bias term, which is a constant parameter.

According to, Abraham (2005), a neural network's architecture consists of several interconnected layers. Artificial neural networks use layers to store neurons before transferring them to the subsequent layer. Each neuron is an algorithm that multiplies the input by the weight assigned to it, passes the sums through the activation function, and then sends the results to other neurons, (Dongare, Kharde, Kachare et al. 2012). At the end of each neuron there exist a node. The input layer, output layer, and hidden layers are the structures in which these nodes are organised. All specified inputs are received by the input layer. The computations that result in the output are done in the hidden layers which is the middle framework. The output from the entire network operation is subsequently delivered via the output layer. The network is structured in such a way that each node links the previous neurons to the next. At each link point the nodes carry an associated activation function, normalises the computed input to produce an output, the weight determines the strength of one node's influence on another and the bias, a constant that is attached to the weighted summation of inputs to offset the results. In simple and plain terms, the layers of

the network filter the input to produce the desired output. As a result, a model that accurately predicts the data and yields results close to the ideal is produced. Neural networks undergo two phases in order to characterise the data that is being modelled. The phases of learning and training. Three categories can be used to categorise the learning phase which are reinforcement learning, unsupervised learning, and supervised learning, (Krenker, Bešter & Kos 2011). Neural networks are trained by feeding in the input and telling the network what the output is supposed to be. The weights are optimised in the training phase.

### 3.1.1 Model of Artificial Neural Networks

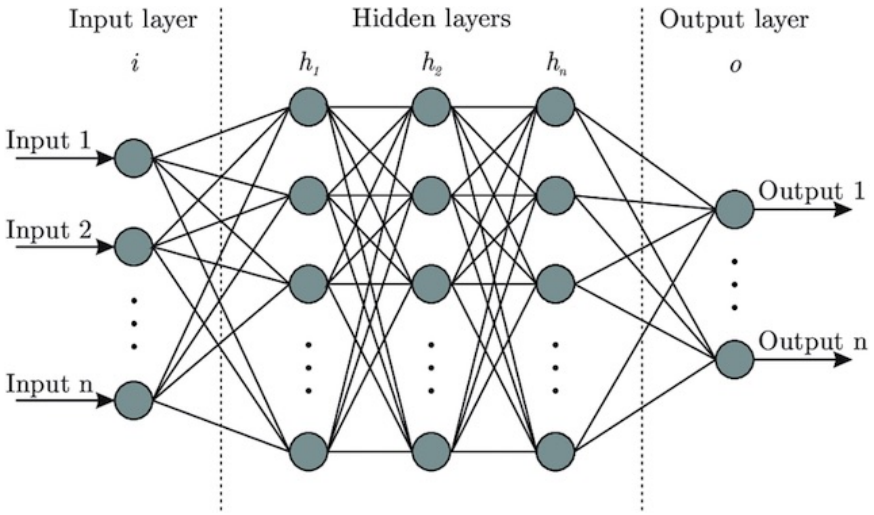


Figure 3.1: Architecture of an Artificial Neural Network

Figure 3.1 above shows the simple architecture of a neural network with supervised learning (Sharma, Sharma & Athaiya 2017). It consists of $n$ inputs, $n$ outputs and $n$ hidden layers. The layers are interconnected and meet at nodes/neurons where calculations take place. The input is disseminated through all the hidden layers in a forward direction to reach the output layer. This is called forward propagation.

## 3.2 Mathematical model of ANNs

A neuron is a neural network's fundamental unit, as such we shall take a closer look at a single neuron from a mathematical perspective in order to create an understanding of how a neural network functions.

### 3.2.1 A Neuron

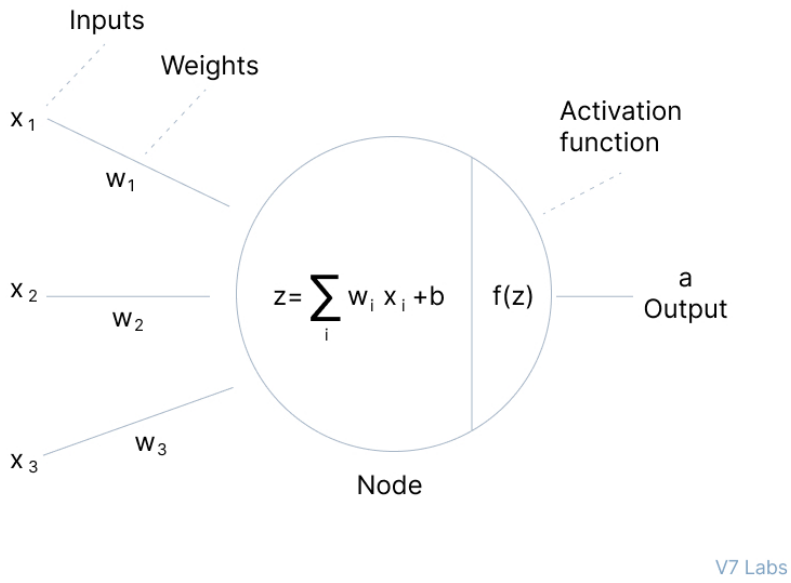


Figure 3.2: A neuron of an Artificial Neural Network

Figure 3.7 shows a neural network with a single neuron one middle structure, the hidden layer. The neuron receives the sum of weighted inputs $\sum_{i=1}^{n} x_i w_i$ and a bias term $b$ and applies a function, $f$ to compute the output, $a$. The function $f$ is the

activation function. From this information we can deduce that,

$$a = f\left(\sum_{i=1}^{n} x_i w_i + b\right). \tag{3.2.1}$$

A single neuron can perform only simple tasks. For complex tasks the structure is expended by adding more neurons and more hidden layers.

## 3.3 Activation functions

Neural networks can describe probable non-linearities in how input and output are related with the use of activation functions. With neural networks, a wide range of activation functions are used. The sigmoid function is regarded as a standard function. The sigmoid function is suitable in this context because it readily approaches linearity close to the origin and saturates quite rapidly as one moves away from the origin. The logistic function and the hyperbolic tangent are two functions that are typically regarded as sigmoid functions. Sharma et al. (2017) looked at the following activation functions.

- **Logistic/ Sigmoid function**

  The activation function used most frequently is the logistic function. It transforms values in the ranges of 0 to 1. It is defined as:

  $$a(z) = \frac{1}{1 + e^{-z}}, \tag{3.3.1}$$

  where $z$ is the slope parameter.

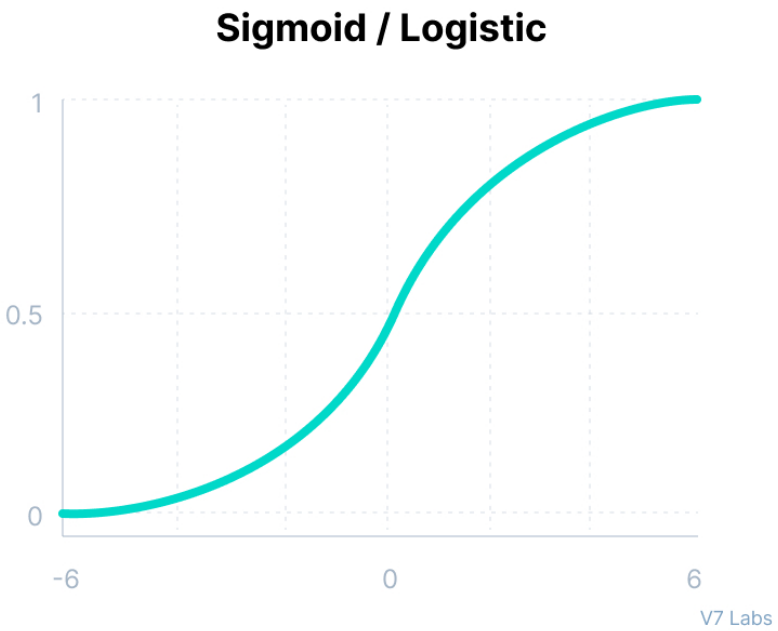


Figure 3.3: Sigmoid activation function

The logistic function is differentiable and its derivative is,

$$\begin{aligned} a'(z) &= \frac{e^{-z}}{(1+e^{-z})^2} \\ &= \frac{1}{1+e^{-z}}\left(1-\frac{1}{1+e^{-z}}\right) \\ &= a(z)(1-a(z)). \end{aligned} \tag{3.3.2}$$

- **Hyperbolic tangent function**

  The sigmoid function and the hyperbolic tangent function (tanh) are compara-

ble, but the latter is symmetric about 0. It is defined as,

$$\begin{aligned} a(z) &= tanh(z) \\ &= \frac{sinh(z)}{cosh(z)} \\ &= \frac{e^z - e^{-z}}{e^z + e^{-z}}. \end{aligned} \tag{3.3.3}$$

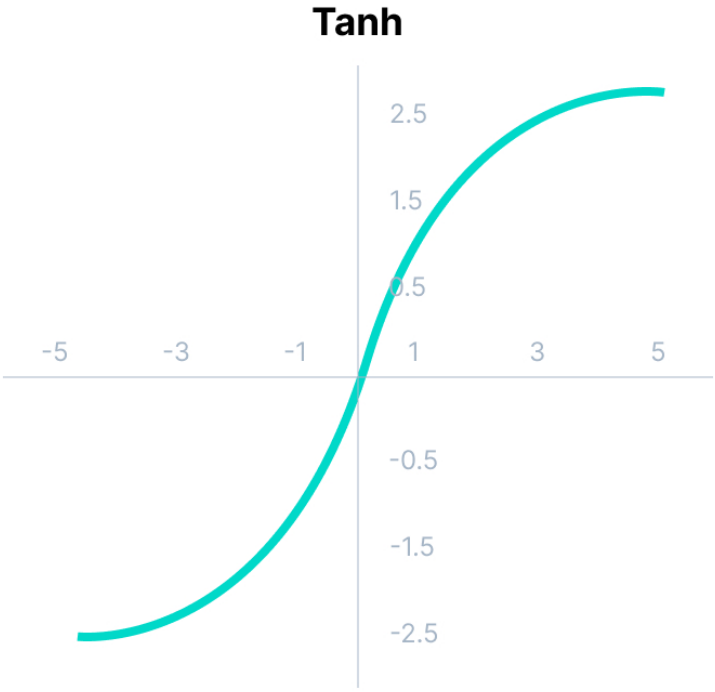


Figure 3.4: Tanh activation function

The hyperbolic tangent is differentiable, and its derivative is given by:

$$a'(z) = 1 - tanh^2(z). \tag{3.3.4}$$

- **RELU function**

  The RELU, rectified linear unit is a non-linear activation function. When applied, neurons are not activated at the same time, only a certain number is activated at a time. It is defined by:

$$a(z) = \begin{cases} z & \text{if } z \geq 0 \\ 0 & \text{if } z < 0. \end{cases} \tag{3.3.5}$$

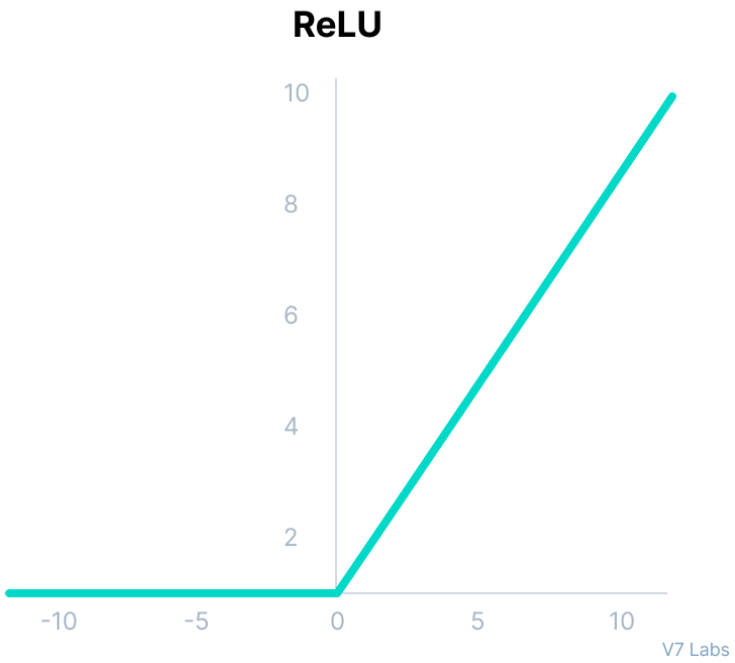


Figure 3.5: ReLU activation function

The derivative of the RELU is given by:

$$a'(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0. \end{cases} \tag{3.3.6}$$

## 3.4 Network architectures

Artificial neural networks are considered to be directed sums of connections weighted on between each layer of nodes or neurons. A key early design choice made by the neural network creators is how nodes are connected, which affects how computations occur. In the literature, different neural network topologies may be identified. Here, we define many networks that are frequently employed in contemporary literature.

### 3.4.1 Feedforward neural networks

This is a particular kind of supervised learning neural network where the network weights connected to the input are changed to make the calculated and intended outputs as similar as possible (Sovzil, Kvasnicka & Pospichal 1997). Here there is no response from the output of the neurons towards the input of the network. The direction of travel of the input data is only one. They are very simple and easy to design and maintain. They are also fast and very responsive. In this chapter, the research covers two of the most popular varieties of FNNs: the MLP and the RBF neural networks.

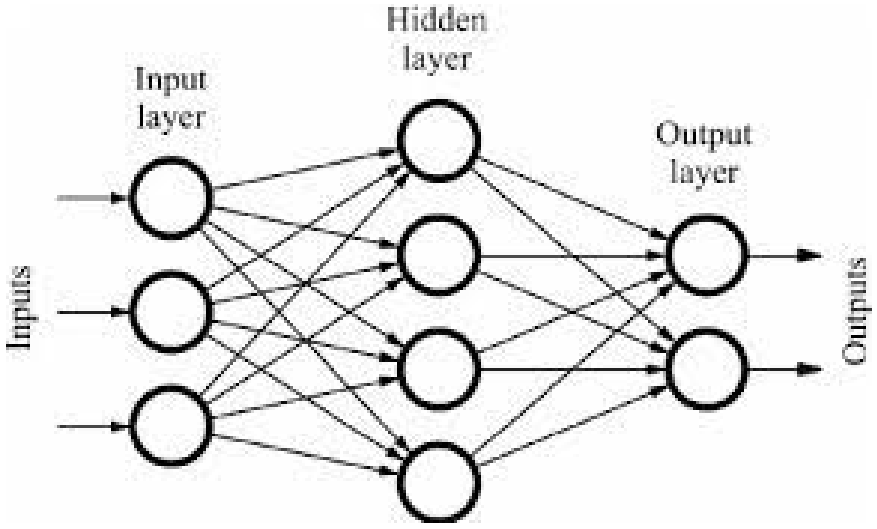


Figure 3.6: A feedforward neural network

### 3.4.2 Recurrent neural networks

For this type of network, there exists a feedback from either either from the overall network output or from the neuron output to the input. Their connections are circular, that is in the network architecture the arrows can lead to the start (input layer). Recurrent neural network uses sequential information with the assumption that the output is dependent on previous computations. RNNs have memory that stores information about what has previously been calculated. RNNs are different from feedforward neural networks. Feedforward neural networks use different parameters at each layer while recurrent neural networks share the same set of parameters for all steps.

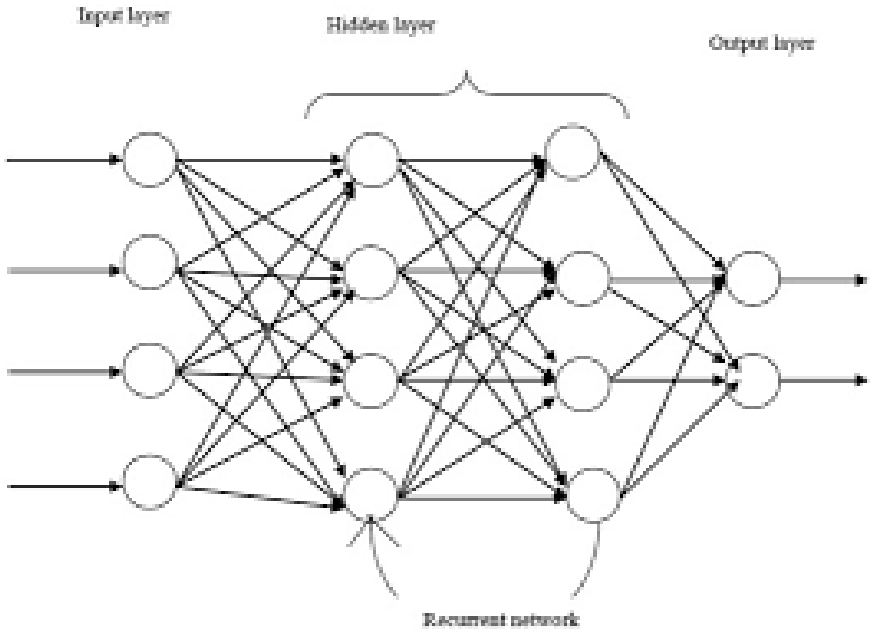


Figure 3.7: A recurrent neural network

## 3.5 Learning in Neural Networks

A neural network has to be designed in such a way that the desired set of outputs is produced by a set of inputs. The influence of each input component to the output is assigned a weight. The weights determine the degree in which one component bares to achieve the output. These weights are adjusted accordingly to establish a relation from the input that produces the output as accurately as possible. The process of adjusting the weights to find an optimal relation is how neural networks learn. Learning scenarios in neural networks can be split into three groups: supervised learning, unsupervised learning, and reinforcement learning. A brief definition of these learning networks is derived from (Krenker et al. 2011) and (Anderson & McNeill 1992).

### 3.5.1 Supervised learning

Learning here happens by comparison of the targeted outcome to the outcome of the network. For this type of learning not only input data is presented into the network but also the targeted outcome. Learning occurs when output data is compared against the provided output. Errors are then propagated back into the network to adjust the weights. Supervised learning is categorised into classification and regression algorithms.

### 3.5.2 Unsupervised learning

Input data is presented into the network without the target output. The learning goal is not explicitly defined but the expectation is that the network will create categories from correlations in the input data in order to produce an output. Unsupervised learning is categorised into clustering and association algorithms.

### 3.5.3 Reinforcement learning

For this type of learning input data is also presented without the target output. The output is generated by interactions in the environment. Decisions from the input are made sequentially. The output is dependent on the state of the current input and the resulting output then serves as the next input. When the output at each point has been defined, reinforcement learning uses algorithms to find an instruction that maximises the return. Positive reinforcement and negative reinforcement are the two types of reinforcement learning. Positive reinforcement is when an event occurs as a result of a particular state and increases the frequency of that particular state. Negative reinforcement is the behaviour that is triggered to stop a negative state from occurring.

## 3.6 Multilayer Perceptron Neural Network

The perceptron was introduced in 1957 by a psychologist Frank Rosenblatt, (Rosenblatt 1960). The perceptron is build from an element that accepts inputs and computes a weighted sum of these inputs where for each input the weight is fixed. The network design of a multilayer perceptron consists of an input layer with $m$ source nodes, $i$ hidden layers with $h_i$ neurons, and an output layer with $n$ neurons. The activation function is a differentiable and non-linear for each neuron. Also, every node of a layer of neurons has a connection to the prior layer. The signal flows in the forward direction through the network based on a layer-by-layer mechanism. Then finding the right weights for the inputs allows the network to learn.

Consider a simple network with a one input $x$, one neuron with the weight associated to the input $w$ and bias $b$ and one output $a$. To find the output $a$ we consider the variable

$$z = w \cdot x + b$$

then

$$a = \sigma(z)$$

where $\sigma(\cdot)$ is an activation function.

Consider a general case where the hidden layer has numerous inputs and neurons. A feedforward neural network is created by taking the dot product of the inputs, weights, and bias and passing it through an activation function. This is applied to all the layers within the network. The output at each neuron is the computed function of the dot product. In the general case with an $n$-dimensional input and $L$ layers. For layer 1, neuron 1.

$$Z_1^1 = \overline{w}_1^T \cdot \overline{x} + b_1^1 \tag{3.6.1}$$

$$Z_1^1 = \begin{bmatrix} w_{11}^1 & w_{12}^1 & \dots & w_{1n}^1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + b_1^1. \tag{3.6.2}$$

This is the dot product of the vector of inputs which is associated to the first input in the neurons of the first layer. Extending this to all the neurons in the first layer, it can be observed that the weight now assumes a matrix form and then $Z^1$ is given by

$$Z^1 = W^1 \cdot \overline{x} + \overline{b} \tag{3.6.3}$$

$$Z^1 = \begin{bmatrix} w_{11}^1 & w_{12}^1 & \dots & w_{1n}^1 \\ w_{21}^1 & w_{22}^1 & \dots & w_{2n}^1 \\ \vdots & \vdots & \dots & \vdots \\ w_{n1}^1 & w_{n2}^1 & \dots & w_{nn}^1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1^1 \\ b_2^1 \\ \vdots \\ b_n^1 \end{bmatrix} \tag{3.6.4}$$

It can be observed that $Z^1$ is an $(n \times 1)$ matrix and the output from this layer given by $A^1$ is also an $(n \times 1)$ matrix given by

$$A^1 = \sigma(Z^1) \tag{3.6.5}$$

where $\sigma$ is the activation function. $A^1$ is the input in the next layer.

Consider again an $n$-dimensional input and a $L$ layers and its output $y$. We have the

equation for a feedforward network given by

$$Z^L = W^L \cdot A^{L-1} + b^L \tag{3.6.6}$$

$$A^L = \sigma(Z^L). \tag{3.6.7}$$

The shape of the weight matrix is (neurons,inputs), $(L, n)$ and this is for one sample. For $m$ samples the input is a matrix of size $(n, m)$ the output is also a matrix of size $(m, 1)$ where each sample input corresponds to one output value.

According to Buscema (1998), a large family of artificial neural networks known as back-propagation have an architecture made up of various interconnected layers. Given a sufficient number of hidden layers, its learning rule is to adopt the deepest descent method, which uses back propagation to control weight value and threshold value of the network to achieve the smallest error of non-linear functions of high complexity.

From the previous section, we consider the summarised feedforward equation

$$Z^L = W^L \cdot A^{L-1} + b^L, \tag{3.6.8}$$

$$A^L = \sigma(Z^L) \tag{3.6.9}$$

and let the output layer be layer $L$ then

$$A^L = \overline{y} \tag{3.6.10}$$

$\overline{y}$ is the output from the network which we will compare to the actual output using a function that computes the error in a single predicted sample $\overline{y_i}$ and the corresponding actual value $y_i$ called the loss function. We will denote the loss function by $f$. The average of the loss function over all the m samples is called the cost. With back propagation our aim is to minimize this cost function. This will be achieved by the

means of a method of gradient descent. The cost function denoted by $C$ assumes the form

$$C = \frac{1}{m}\sum_{i}^{m} f(y_i - \overline{y}_i) \tag{3.6.11}$$

this is a function of $W$, $A$ and $b$. But it can be considered a function of of $W$ and $b$ since $A$ is the input

$$C = f(W, b). \tag{3.6.12}$$

It remains then to find the values for W and b that minimize the function $C$. First we observe the learning equation that is used to update the weights and bias

$$W = W - \alpha\frac{\partial C}{\partial W} \tag{3.6.13}$$

$$b = b - \alpha\frac{\partial C}{\partial b} \tag{3.6.14}$$

$\alpha$ is the learning rate (Rumelhart, Hinton & Williams 1986).

## 3.7 Radial Basis Function Neural Network

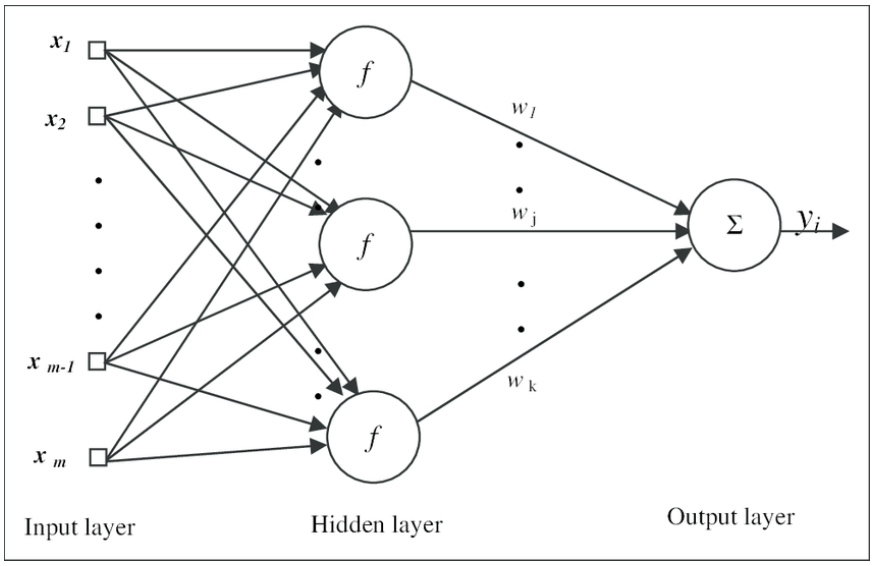


Figure 3.8: The network architecture of a Radial Basis Function Neural Network

Radial basis function neural networks are utilised in a variety of applications because they have the ability approximate all regular functions and train quicker than multilayer perceptrons (Turhan & Toprak 2013). The RBFNN can learn quickly since there are only two levels of weights and at each layer may be evaluated sequentially. The RBFNN's architecture, on the other hand, is not simple. It is composed of three layers which are the input, middle and output layers (Turhan & Toprak 2013).

### 3.7.1 Radial Basis Function

The RBF is a function with real values that calculates the length between an input $\mathbf{x}$ and some prototype vector $\mathbf{v}$. Radial basis functions are functions that make up the core of the radial basis function neural networks. They are defined by their argument.

The argument for the radial basis function is given by the euclidean distance, as such it is always positive. The value of this function is defined by:

$$\phi(x) = \phi(||x||). \tag{3.7.1}$$

The RBF, also called a kernel function or Gaussian function is often applied to this argument in an RBFNN. The one-dimensional input formula for a Gaussian is given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\Pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \tag{3.7.2}$$

### 3.7.2 Radial Basis Function Network (RBFN)

The activation unit in these neural networks is defined by how far apart the input and prototype vectors are from one another. The activation function that computes the weight for each neuron is the radial basis function. This network is usually used for classification problems as it can learn very fast, again this network is used for interpolation. RBF networks model data in terms of spheres or radial shapes dividing the space. For each sphere the further away from the centre the less accurate the results of the function. The function is chosen such that there is an abrupt drop off in the graph of the function. This drop off means that the precision of the model drops as a result of growth in distance from the centre of the radial shape containing a data point.

The network for the RBFNN consists of the input layer, output layer and the hidden layer. The following parametric model can serve as a representation for the model.

$$f(X) = \sum_{i=1}^{n} w_i \phi_i(||X - \mu_i||, \theta_1) \tag{3.7.3}$$

where $X \in R^n$ is the input

$\theta_i$ is the basis function of the network from $R^n$ to R

$w_i$'s are weights of the network

$\mu_i = (\mu_{i1}, ..., \mu_{in}$ is called the center vector of $ith$ node

$\theta_i = (\theta_{i1}, ..., \theta_{in})^T$ is called the band width vector of the $ith$ node

$||\cdot||$ denotes the Euclidean norm.

If the basis function of the network is the Gaussian, then

$$\phi_i(||X - \mu_i||, \theta_t) = exp\left\{\left[\frac{-||X - \mu_i|}{(\theta_{i1}, ..., \theta_{in})}\right]^2\right\}. \tag{3.7.4}$$

**Input layer** This layer comprises of artificial nodes that receive initial data to be introduced into the system to be processed in the subsequent layers of the network. Each input unit receives a certain attribute that forms part of the data. Here we use a linear function.

**Hidden layer** Center are determined at random. First we select the number of hidden units, then randomly pick input patterns from the data set. The wights are determined between the middle layer and input layer and are fixed when the centrers are determined. The wights are also determined between the output and the middle layer. Radial basis function neural networks training is usually a mapping from a vector **x** to a vector **y** from a training set.

### 3.7.3 RBF Activation functions

It has been mentioned before that the Gaussian function can be used as a radial basis function. It is worthwhile to look at other functions that can be used as radial basis

functions. Consider these functions, $r$ defines the euclidean distance and $\sigma$ is a tuning parameter.

### 3.7.4 Gaussian

$$\phi(r) = e^{-(\sigma r)^2} \tag{3.7.5}$$

### 3.7.5 Multiquadric

$$\phi(r) = \sqrt{1 + (\sigma r)^2} \tag{3.7.6}$$

### 3.7.6 Inverse quadric

$$\phi(r) = \frac{1}{1 + (\sigma r)^2} \tag{3.7.7}$$

### 3.7.7 Inverse multiquadric

$$\phi(r) = \frac{1}{\sqrt{1 + (\sigma r)^2}} \tag{3.7.8}$$

# 4. Data and Numerical Tests

This chapter presents the numerical outcomes of the study with the objective of exploring how the artificial neural networks perform in terms of pricing European put and call options as compared to the traditional B-S model. Furthermore, the chapter describes the process followed to generate data for the ANN. To be more specific, the MCS model is used to simulate the option prices. Then the RBF and MLP networks are trained using the generated option prices data as inputs. In addition, the performance estimates are computed in order to evaluate the prediction capability of the ANNs in option pricing.

## 4.1 Data Generation

It was essential to use simulated synthetic training data because there are not many resources for high-quality option price data. For the input variables that were converted into European call and put prices makimg use of the analytic solution of the B-S method, the artificial training data were randomly picked from a wide range and were simulated by the MCS method for pricing options (Boyle, Broadie & Glasserman 1997). The Monte Carlo pricing method for European options uses B-S model parameters as its inputs. The parameters are each varied randomly within certain bounds that were arbitrarily selected. We will look at the steps taken to generate data for this study. The Monte Carlo simulations method values options by simulation of a random walk and cashflows for the underlying asset, taking an average of the pay-off and discounting it to the the present day. We have applied this method to several assets that observe different values in their parameters. This model becomes more accurate when the number of sample paths is increased, Cuomo, Sica & Toraldo (2020) deduced that a factor of 100 in a set of simulations produces an increased accuracy of 10 percent.

It was decided to use 10,000 uniformly distributed random samples of each model parameter over the ranges indicated in table 4.1

Table 4.1: Input parameter ranges

| **Parameters** | **Range** |
|---|---|
| Stock price ($S$) | [50,100] |
| Exercise price ($K$) | [50,100] |
| Risk free interest rate ($r$) | [0.001,1] |
| Volatility ($\sigma$) | [0.001,0.5] |
| Time to expiry ($T$) | [0,1] |
| Option Price ($y$) | $\mathbb{R}^+$ |

By implementing Formula 2.7.6, the option prices are determined. It should be noted that a standard scaler is used on the input parameters. It is again used on the output option prices to scale them. Since the data we are working with spans a wide range, it is vital to scale and normalize the data to guarantee that each parameter has an equal impact on the outcomes and, consequently, the accuracy of the neural networks. In addition, option prices which violate the lower bound condition for either call or put options are removed. The samples of the simulated data for the European call option and European put option are displayed in Table 4.2. This procedure is repeated multiple times, with the number of selected standard normal random variables $n = 10000$, to generate enough data for the purpose of training and testing the neural networks. In Table 4.2 each row represents the input parameters for either the European call or put option.

Table 4.2: Sample of simulated European call and put options data and parameters

| | S | K | r | sigma | T | call price | put price |
|---|---|---|---|---|---|---|---|
| 0 | 66.9225 | 84.1248 | 0.9917 | 0.4613 | 0.2223 | 5.5654 | 6.0722 |
| 1 | 84.0152 | 70.1519 | 0.6035 | 0.2315 | 0.5098 | 32.4963 | 0.0026 |
| 2 | 87.0010 | 98.8384 | 0.3389 | 0.4986 | 0.3864 | 10.8864 | 10.5029 |
| 3 | 80.2140 | 53.0093 | 0.8281 | 0.2932 | 0.9404 | 55.9608 | 0.0000 |
| 4 | 96.6355 | 87.9691 | 0.1922 | 0.4575 | 0.7635 | 26.4280 | 5.6344 |
| 5 | 81.7406 | 82.4568 | 0.9081 | 0.2344 | 0.7753 | 41.0200 | 0.0003 |
| 6 | 57.7904 | 73.6136 | 0.5184 | 0.4343 | 0.6516 | 10.6241 | 5.2803 |
| 7 | 82.4563 | 56.9632 | 0.5792 | 0.4198 | 0.9699 | 50.1583 | 0.0769 |
| 8 | 65.8502 | 72.4852 | 0.0589 | 0.1351 | 0.9291 | 2.2891 | 5.0312 |
| 9 | 85.3324 | 90.9825 | 0.8937 | 0.2963 | 0.8715 | 43.6784 | 0.0202 |

## 4.2 Performance measurements

Four performance measurements were used in this study. These parameters include mean absolute deviation (MAD), mean deviation (MD), symmetric mean absolute proportionate deviation (SMAPD) and mean square deviation (MSD) (Bennell & Sutcliffe 2003). Let $y_i$ be the observed option price, $\hat{y}_i$ be the predicted option price and $i = 1, 2, \ldots, n$ are the option price observations. Then the following equations

show the performance measures.

$$\begin{aligned} MD &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i) \\ MAD &= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \\ SMAPD &= \frac{2}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(y_i + \hat{y}_i)} \\ MSD &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \end{aligned} \tag{4.2.1}$$

## 4.3 Training artificial neural networks

Using the data generated from the previous section, next the two corresponding neural networks are trained. Two artificial neural networks were considered for this study, namely the radial basis function neural network and the multilayer perceptron neural network. Both these models use the feed forward architecture for approximation problems. They have been trained using simulated data over one trading period of 252 days.

Before we use the neural networks to estimate option prices calculated from the Monte Carlo process pricing model, the input and output parameters are defined as follows:

- input parameter: Moneyness $S/K$, Time to expiration $T - t$
- output parameter: $y/K$

where $S$ is the stock price, $K$ is the strike price, $y$ is the call option price and $T - t$ is the time to expiration expressed in years.

The reason why using $y/K$ as the target variable rather than y is to make all features have similar ranges. For the purpose of explaining the fitting potency of neural networks, the examination uses 70% of data as in sample data for training and the 30% that is remaining of the data as out of sample data for testing. The input data is sectioned into the the input, $X$, and the output, $y$. where $X$ is a matrix with rows consisting of the number of samples and columns of the five Black-Scholes parameters. The output, $y$, is the resulting call or put value that was achieved from Monte-Carlo simulations.

The train data shall be referred to as Xtrain for the input and ytrain for the output, and the test data as Xtest for the input and ytest for the output. The neural network is trained utilising the train data where the model takes in Xtrain and trains the model towards the ytrain so that the model recognises the patterns in the data in Xtrain that result in the corresponding ytrain. This process is called model fitting. After fitting the model, we verify that the model has indeed learned by using the input Xtest and observing the output that results. We compare this output to the ytest data and calculate the prediction strength of our model. The correlation of the Xtest output and ytest is expected to be close to 1 if the model is a good fit. Both our neural networks have been trained using the same data.

To avoid model over-fitting, validation losses were tracked after each training session, allowing for a process that is driven by change of training the ANN. As a result, if the ANN's strength begins to deteriorate after a few sessions, training will come to an end immediately. Making use of model boundaries, which save the ANN configuration after each session can further improve this dynamic training process. When training is complete, the ideal ANN setup is retained and can be used for any approximation problems. In this case, the pricing of European options.

### 4.3.1 MLPNN model training

A neural network is defined by a function that takes input parameters as the input, has a middle and an output layer. Separating the input and output layers of the MLP, each of the three hidden layersin the middle is made up of nodes equal to the sets of inputs. The RELU activation function computes the output in this model. In this scenario, the loss function calculated by the mean squared deviation, which is minimised and propagated back through the model to provide the best possible output with the least amount of loss. There are $n$ a total of training samples. The MLP network design is described in Table 4.3.

Table 4.3: MLP neural architecture

| **Parameter** | **Configuration** |
|---|---|
| Number of hidden layers | 3 |
| Neurons in output layer | 1 |
| Hidden layer activation function | ReLU |
| Output layer activation function | ReLU |
| Learning rate | 0.01 |
| Epochs | 10 |

Training error is tested by how well the model can predict the output when the test data is presented as input for the model. The output is compared to the test data output. In this case 3000 sets of inputs were used to test the model and the performance measures for the call and put using MLP are shown in Table 4.4.

Table 4.4: Error analysis for MLP model training for call and put options

| | **MLP call** | **MLP put** |
|---|---|---|
| R-squared | 0.986953694 | 0.999998812 |
| MD | -0.084684445 | 1.031771192 |
| MAD | 0.10203192 | 1.031771192 |
| MSD | 0.013911015 | 1.085582577 |
| SMAPD | 0.049943904 | 0.064925384 |

From Table 4.4 for the European call the $R^2$ is almost close to 1. The MD, MAD, MSD, are very small in magnitude for the European call. Likewise for the European put it is observed that the $r^2$ is very close to 1. The rest of the performance measures are also relatively small in magnitude for the European put.

A graphical representation of some of the performance measures for European call options test sets are presented in Figures 4.1 - 4.2.

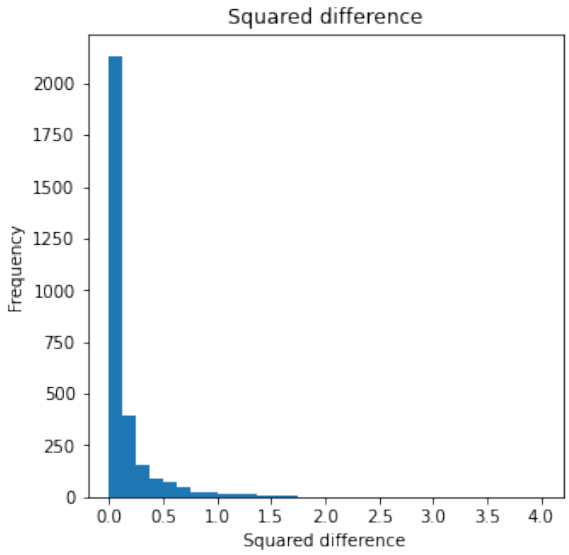


Figure 4.1: MLP squared difference for call options

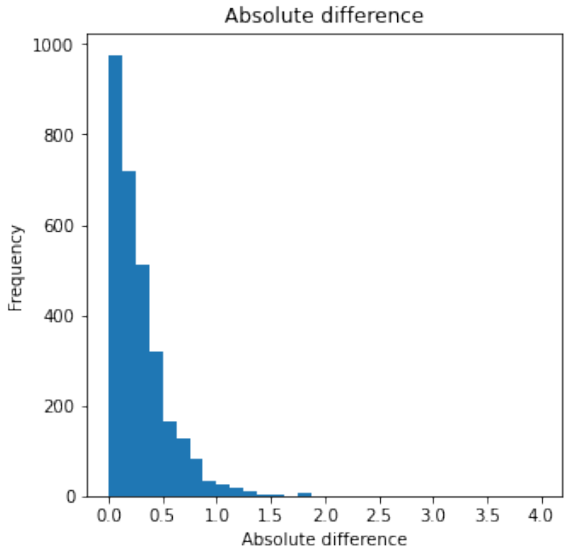


Figure 4.2: MLP absolute difference for call options

The plots in Figures 4.1-4.2 show that the magnitude of error decreased as the graphs are skewed towards the left and closer to zero. This hints that the MLP neural network was properly trained.

A graphical representation of some of the performance measures for European put options test sets are presented in Figures 4.3 - 4.4.

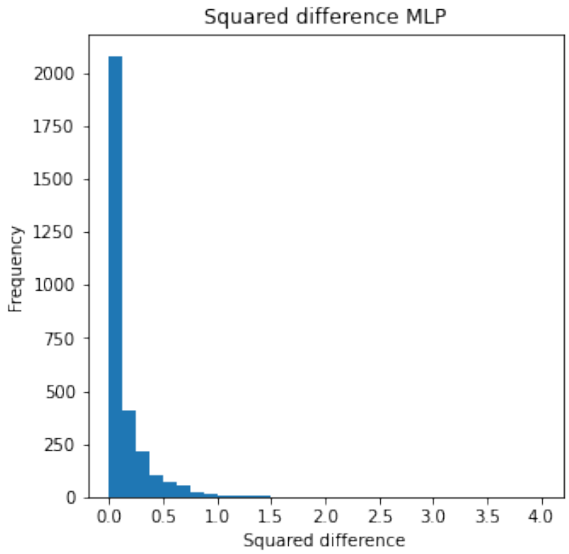


Figure 4.3: MLP squared difference for put options

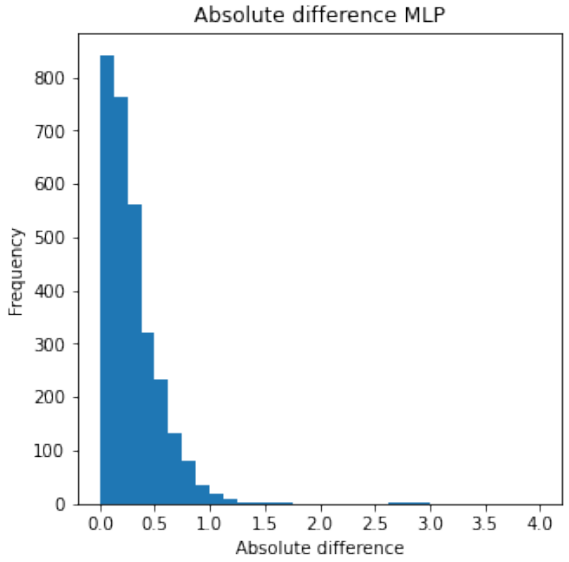


Figure 4.4: MLP absolute difference for put options

The plots in Figures 4.3-4.4 show that the magnitude of error of European put options mostly takes values between 0 and 0.5. This could be attributed to proper training of the model.

### 4.3.2 RBFNN model training

Three layers make up the RBFNN. The RBF employed one of each layer defined in the architecture. The hidden layer consists of nodes equal to the determined number of centers. The output in the RBF is computed by the Gaussian radial basis activation function. The loss function that is used in this case is the mean squared error and it is minimised and propagated back through the model to reach an optimum output with the smallest possible loss. The training sample is given by $n$ sets. The RBF network architecture is summarised in Table 4.5

Table 4.5: RBF neural architecture

| **Parameter** | **Configuration** |
|---|---|
| Number of hidden layers | 1 |
| Centres in first hidden layer | 70 |
| Neurons in first hidden layer | 70 |
| Hidden layer activation function | Gaussian RBF |

How effectively the model predicts the outcome when the test data is used as the model's input is how training error is measured. The output is contrasted with the output of the test data. In this instance, 3000 sets of inputs were used to test the model, and the call and put utilizing RBF performance metrics are displayed in Table 4.6.

Table 4.6: Error analysis for RBF model training for call and put options

| | **RBF call** | **RBF put** |
|---|---|---|
| R-squared | 0.999715334 | 0.999890142 |
| MD | 1.187827717 | 1.092062158 |
| MAD | 1.187827717 | 1.092062158 |
| MSD | 1.447484026 | 1.216660161 |
| SMAPD | 1.169739048 | 0.068818864 |

The Table 4.6 displays performance measure for the RBF model. For call options the $R^2$ is very close to 1 and the rest of the rest of the performace measures are very small. Similarly For European put options $R^2$ is close to 1 with MD,MAD,MSD and SMAPD being very small.

A graphical representation of some of the strength measures on the test set for the European call options are presented in Figures 4.5 - 4.6.

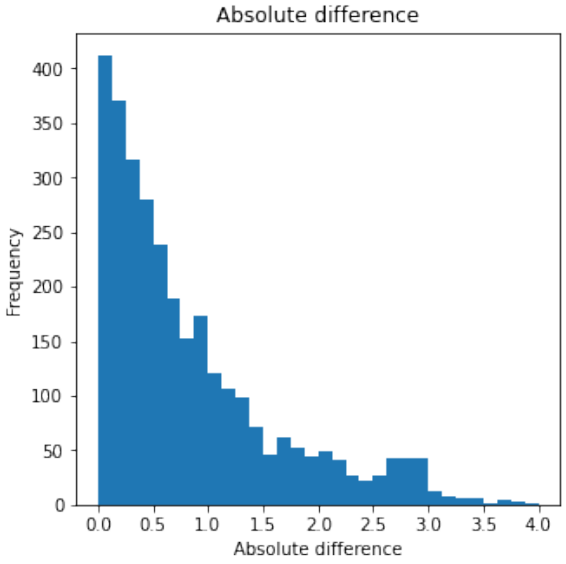


Figure 4.5: RBF absolute difference for call options

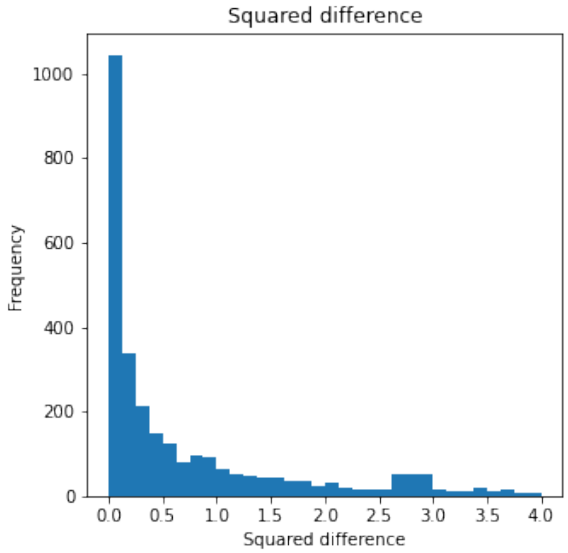


Figure 4.6: RBF squared difference for call options

The Figures 4.5-4.6 show plots that describe the spread of error for European call options. Most of the error values are concentrated towards the left of the plots nearing zero. This indication suggests a well trained model.

A graphical representation of some of the strength measures on the test set for the European call options are presented in Figures 4.7 - 4.8.

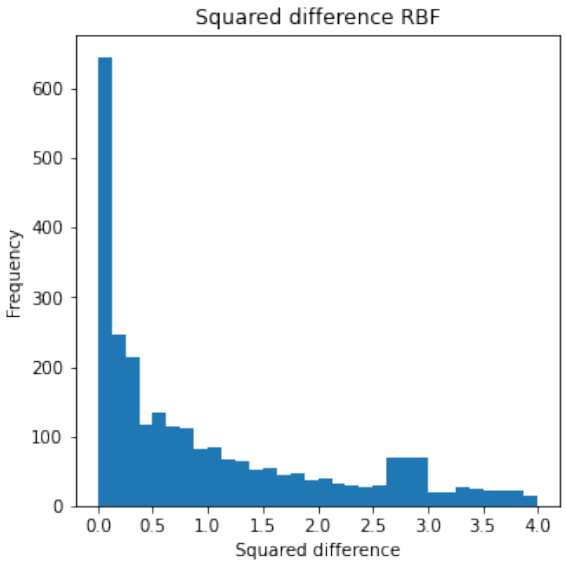


Figure 4.7: RBF absolute difference for put options

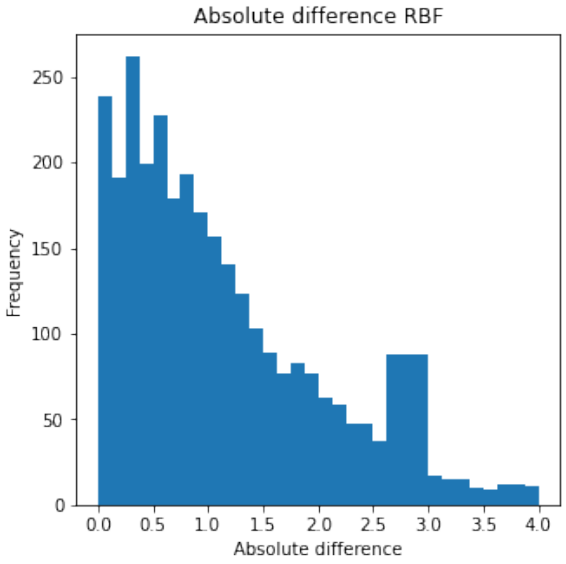


Figure 4.8: RBF squared difference for put options

The plots in Figures 4.7-4.8 show the error amounts for European put options are mainly between the values 0 and 1.5 with most of the error close to zero. The model could be properly trained.

## 4.4 Numerical tests

This section presents all numerical tests results using the trained artificial neural networks as introduced in the prior section. Then a comparison of the results to the Black-Scholes theoretical model is made. For analysis purposes, comparison will be done on the European call options and corresponding European put options.

### 4.4.1 Numerical analysis for European call option

This subsection presents numerical results from pricing the European call option. Figure 4.9 shows the predicted prices from the RBF with a comparison to the analytical option prices obtained using the B-S model. The line in blue indicates all points at which the RBF predicted prices exactly match the analytical prices. It is clear that the RBF is able to accurately predict the Black-Scholes analytical price values for the call option.

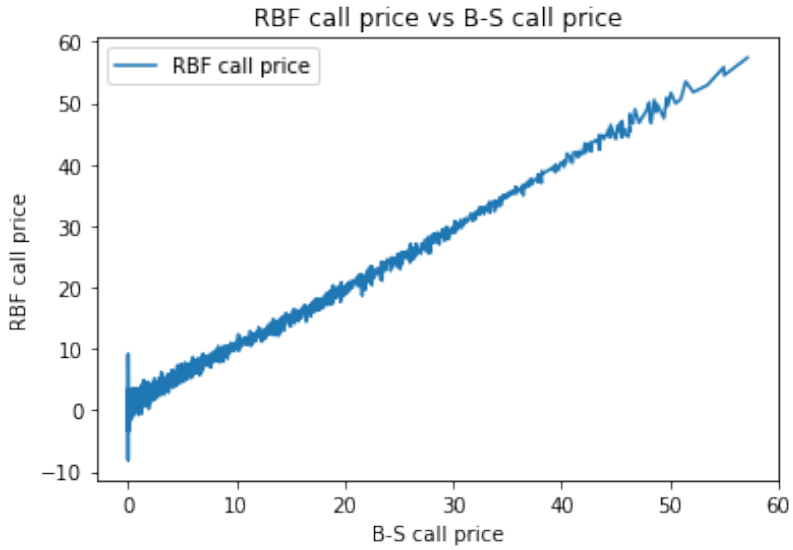


Figure 4.9: RBF predicted and analytical prices for the European call option

Figure 4.10 shows the predicted prices from the MLP with a comparison to the analytical option prices obtained using the analytical B-S model. The points along the blue line where the MLP anticipated prices exactly match the analytical values are all indicated. It is also evident that the MLP can correctly forecast the call option's Black-Scholes analytical prices.

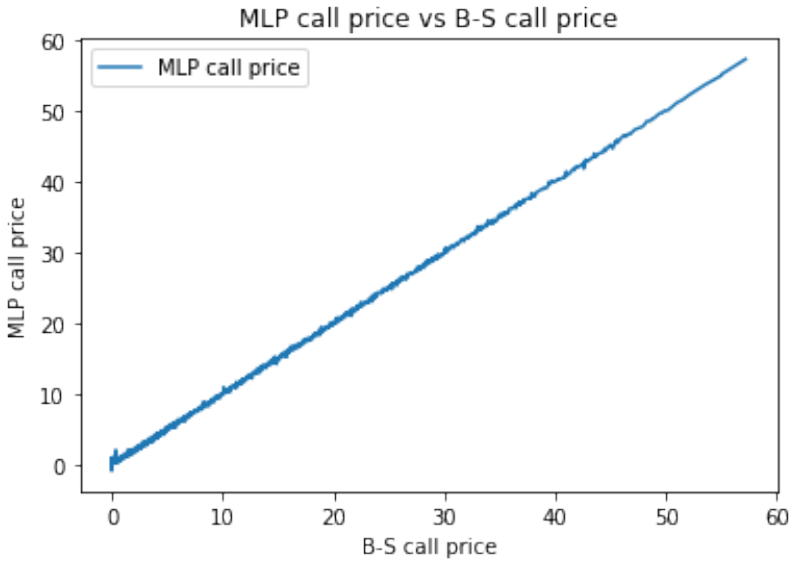


Figure 4.10: MLP predicted and analytical prices for the European call option

Next, what-if-analysis for the European call is presented where the parameters are varied. A comparison of the options is made using the Black-Scholes model and the RBF and MLP neural networks.

**a. Varying time to expiry for the European call**

Figure 4.11 shows the option prices as the time to expiry is varied. The set of parameter values used are: $S = 75$, $K = 75$, $r = 0.3$, $\sigma = 0.4$ and $T \in (\frac{1}{252}, 1)$.

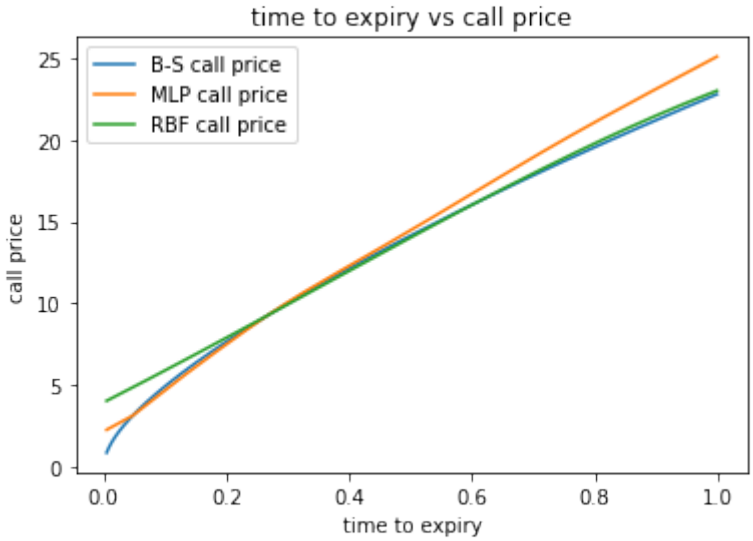


Figure 4.11: Call option prices as time to maturity is varied

From Figure 4.11 it can be observed that the value of the call option rises as time to expiry increases.

It can also be observed that for all the three models, when there is considerably much time to maturity, call option values have similar behaviour and have amounts that are very close in value. As time to maturity is reduced the results become less accurate, observable by divergence from the reference values. This is apparent to artificial neural network models. We can assume the explanation to this behaviour is due to neural networks being less sensitive to changes resulting from small $T$ values. In comparison The MLP results are more similar to those of the B-S model than those priced by using the RBF when all parameters are constant except time to expiry. The overall observation is that the call price is directly proportional to time to maturity. This implies that when the contract draws closer to expiry the option value decreases.

**b. Varying risk-free rate for the European call**

Figure 4.12 shows the option prices as risk-free rate of interest is varied. The set of parameter values used are: $S = 75$, $K = 75$, $r \in (0.001, 0.5)$, $\sigma = 0.3$ and $T = 0.5$.

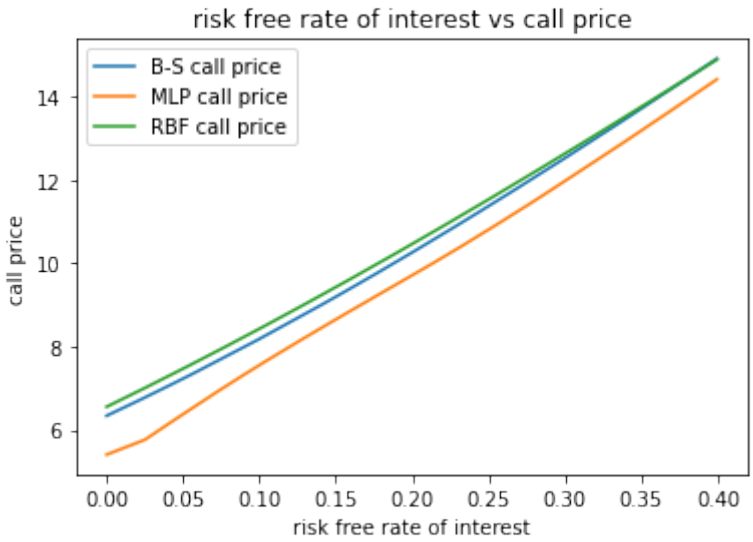


Figure 4.12: Call option prices as risk-free rate of interest is varied

Figure 4.12 The value of the call option increases with increase in risk free rate of interest.

It can be observed that when the rate of risk-free interest assumes very small values call option prices are very small for all the three models. As the rate of interest increases the values of the call options increase. The MLP call options appear to slightly differ from the B-S call option values as the interest rate increases. However the general proportion is still maintained.

**c. Varying volatility for the European call**

Figure 4.13 shows the option prices as volatility is varied. The set of parameter values used are: $S = 75$, $K = 75$, $r = 0.3$, $\sigma \in (0.01, 0.5)$ and $T = 0.5$.

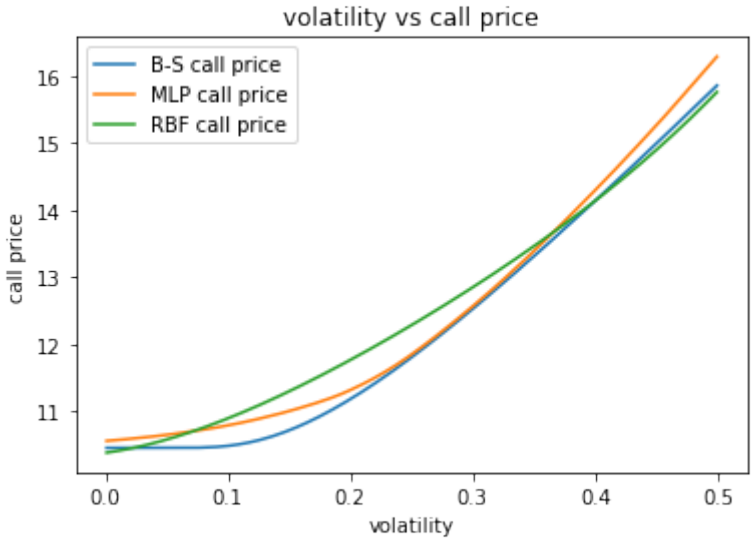


Figure 4.13: Call option prices as volatility is varied

From Figure 4.13 it can be observed that value of the option increases with increase in volatility.

Figure 4.13 gives a graph for call option values when volatility was allowed to vary while all the other parameters are held constant. For call options the price of the options rises with increase in volatility. The graph shows that when volatility is low the price for calls is almost constant and small then begins to constantly increase. When volatility is high, it means the price of the underlying asset at the time of exercise can occupy very high values and very low values. When the values are very high, the call option value will also be high and when low the value of call options can only go as low as 0. MLP option prices are more similar to B-S option prices than RBF option prices.

### d. Varying exercise price for the European call

Figure 4.14 shows the option prices as exercise price is varied. The set of parameter

values used are: $S = 75$, $K \in (50, 100)$, $r = 0.3$, $\sigma = 0.3$ and $T = 0.5$.

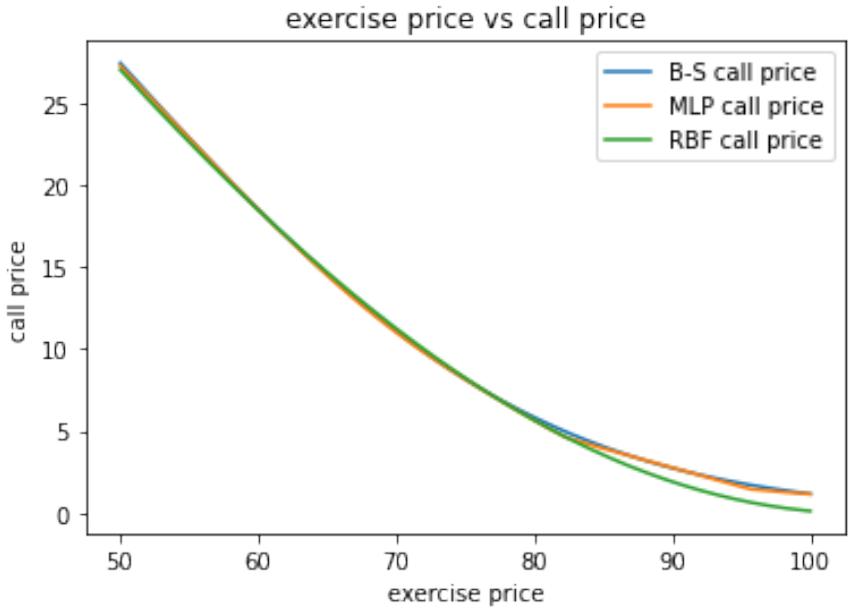


Figure 4.14: Call option prices as the exercise price is varied

Figure 4.14 shows that the price of the call option decreases with increased value of the exercise price.

The Figure 4.14 gives the graph for call options prices when the exercise price was allowed to vary while the values of all other parameters are held constant. The prices for call options declines as the price of the exercise price is increased for all the three models. For smaller strike prices the call price is very high and all the option prices are similar for all the three models. For low exercise prices the RBF prices start to diverge from the other models. It can be concluded from observation that for both the artificial neural network models, the MLP models provides a more accurate prediction when compared with the B-S model.

**e. Varying asset price for the European call**

Figure 4.15 shows the option prices as asset price is varied. The set of parameter values used are: $S \in (50, 100)$, $K = 75$, $r = 0.3$, $\sigma = 0.3$ and $T = 0.5$.

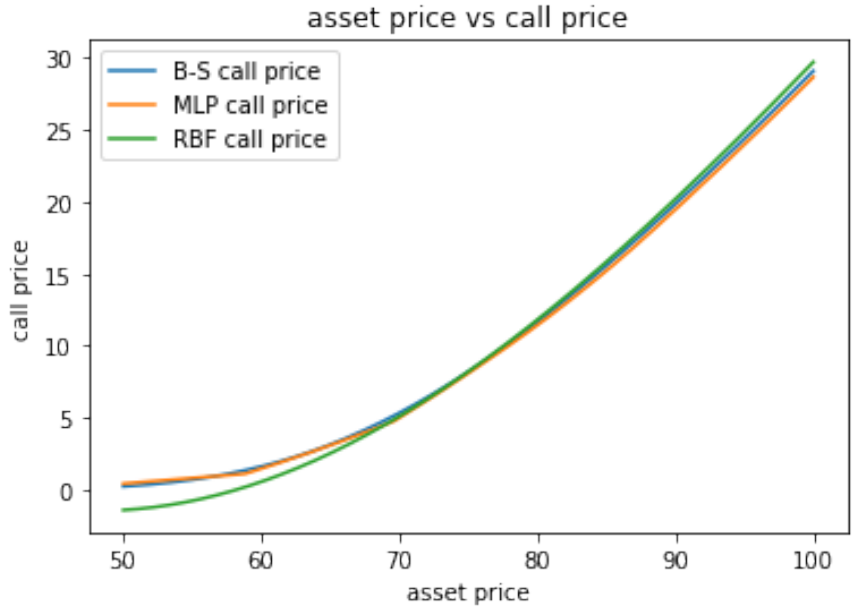


Figure 4.15: Call option prices as the asset price is varied

Figure 4.15 shows that the value of the option increases with increase in stock price.

Figure 4.21 gives a plot for the value call option prices when the asset price was varied while the values of all the other parameters were held constant. The value of call options rises when the value of the stock is increased. We can observe that from the figure, that value of call options appear to be very similar for all the models when the stock prices are high. For smaller stock prices the RBF model values seem to be different from MLP and B-S values. We can observe for both artificial neural network models values when compared to B-S models values, the MLP provides a more accurate prediction than the RBF.

Table 4.7 presents metrics for the Black-Scholes, MLP and RBF model.

Table 4.7: Metrics for European call options

| | BS | MLP | RBF |
|---|---|---|---|
| min price | 0 | -0,818993302 | -5,645003158 |
| max price | 57,1736 | 55,33295935 | 58,06854652 |
| min AD | - | 0,818993302 | 5,645003158 |
| max AD | - | 2,457881383 | 8,439974621 |
| min SD | - | 0,818993302 | 5,645436023 |
| max SD | - | 6,041180894 | 71,2331716 |

Table 4.7 is a table of matrices for European put options. It displays the maximum and minimum for call option prices, absolute deviation and squared deviation. Deviation for this table is calculated between artificial neural networks and the analytical Black-Scholes. It can be observed that the maximum deviations of the MLP model from the B-S model assumes values that are lower in magnitude than the deviation of the RBF model from the B-S. The minimum deviations in this table are relatively lower in magnitude for the MLP model than the RBF model. The minimum value for MLP call options is relatively closer to the minimum value for call options calculated using B-S model compared to the minimum value for the call options calculated using RBF model. This observation differs for the maximum price for call options. The maximum value for call options obtained from the RBF model is closer to the maximum value calculated from the B-S pricing method than one predicted by the MLP model.

### 4.4.2 Numerical analysis for European put option

This subsection presents numerical results from pricing European put options and how they perform in relation to the Black-Scholes analytical pricing model.

Figure 4.16 shows the predicted prices from the RBF with a comparison to the ana-

lytical option prices obtained using the Black-Scholes model. All points at which the RBF predicted prices exactly match the analytical prices are indicated by the blue line. The RBF is able to accurately predict the Black-Scholes analytical prices for the put options.

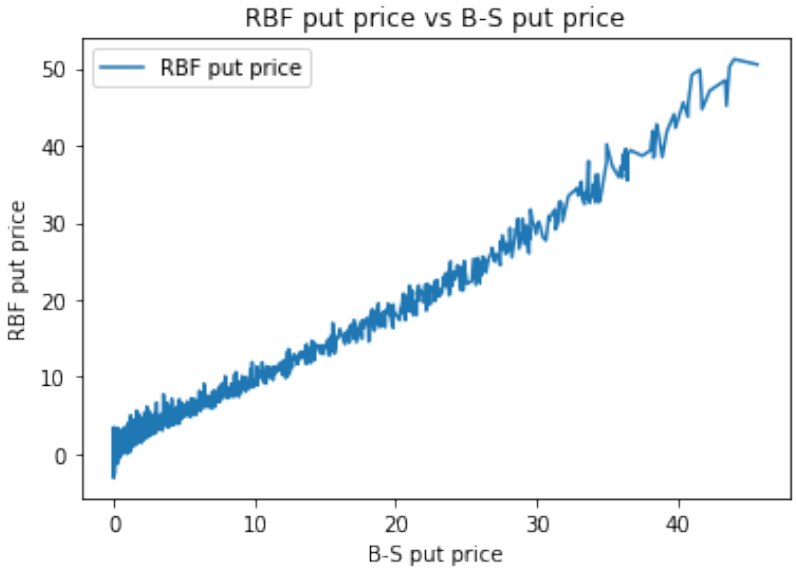


Figure 4.16: RBF predicted and analytical prices for the European put option

Figure 4.17 shows the predicted prices from the MLP with a comparison to the analytical option prices obtained using the Black-Scholes model. The blue line indicates all points at which the MLP predicted prices exactly match the analytical prices. It is clear that the MLP is able to accurately predict the Black-Scholes analytical prices for the put option

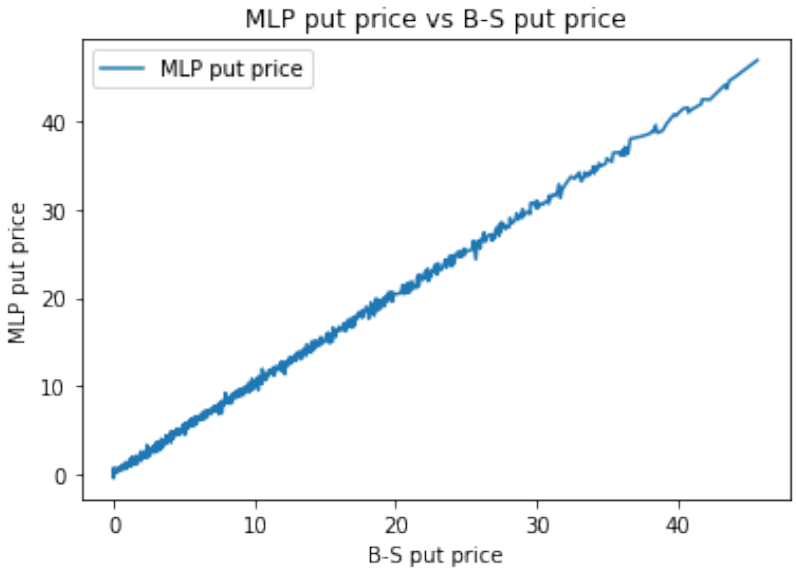


Figure 4.17: MLP predicted and analytical prices for the European put option

Next what-if-analysis for the European put is presented with the parameters are varied. A comparison for the options is made using the Black-Scholes model and the RBF and MLP neural network models.

**a. Varying time to expiry for the European put**

Figure 4.18 shows the option prices as the time to expiry is varied. The set of parameter values used are: $S = 75$, $K = 75$, $r = 0.3$, $\sigma = 0.4$ and $T \in (\frac{1}{252}, 1)$.

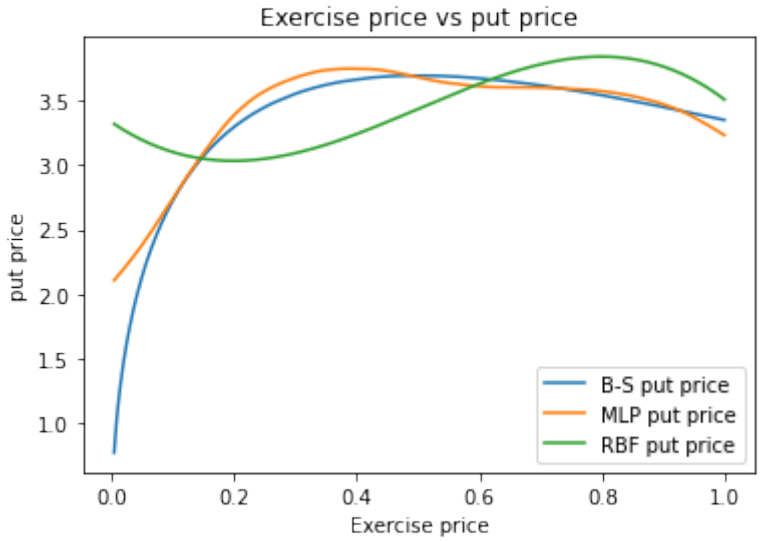


Figure 4.18: Put option prices as time to expiry is varied

Figure 4.18 shows that the value of the European put option decreases as time to expiry increases. This implies that when the contract draws closer to expiry the value of the option increases.

Figure 4.18 shows a graph of the results are out of sample obtained from trained RBF and MLP. The corresponding Black-Scholes model used the parameters described as inputs. We observe that for all the three models, when we have considerably much time to maturity, put option have similar values. As time to maturity is reduced the results become less accurate for the RBF model, observable by divergence from the B-S values. The graphs for MLP and RBF show that when time to maturity is varied and the models are compared to the B-S model we do not observe a good fit for put options.

**b. Varying risk-free rate for the European put**

Figure 4.19 shows the option prices as the time to expiry is varied. The set of

parameter values used are: $S = 75$, $K = 75$, $r \in (0.001, 0.5)$, $\sigma = 0.3$ and $T = 0.5$.

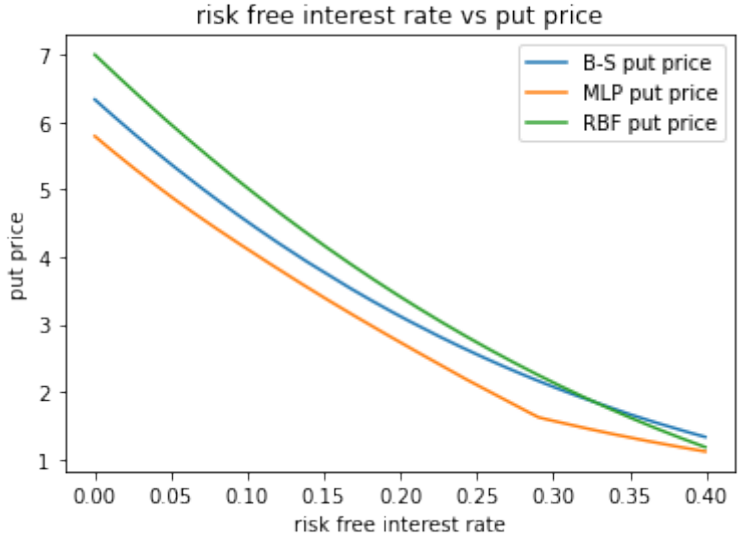


Figure 4.19: Put option prices as risk-free rate of interest is varied

Figure 4.19 shows that the value of the European put option decreases with rise in risk free rate of interest.

It can be observed that for small rates of interest the value of put options is large. The prices for put options resulting from artificial neural network models are similar to those the B-S models. The results from the RBF model are slightly less similar.

### c. Varying volatility for the European call

Figure 4.20 shows the option prices as volatility is varied. The set of parameter values used are: $S = 75$, $K = 75$, $r = 0.3$, $\sigma \in (0.01, 0.5)$ and $T = 0.5$.

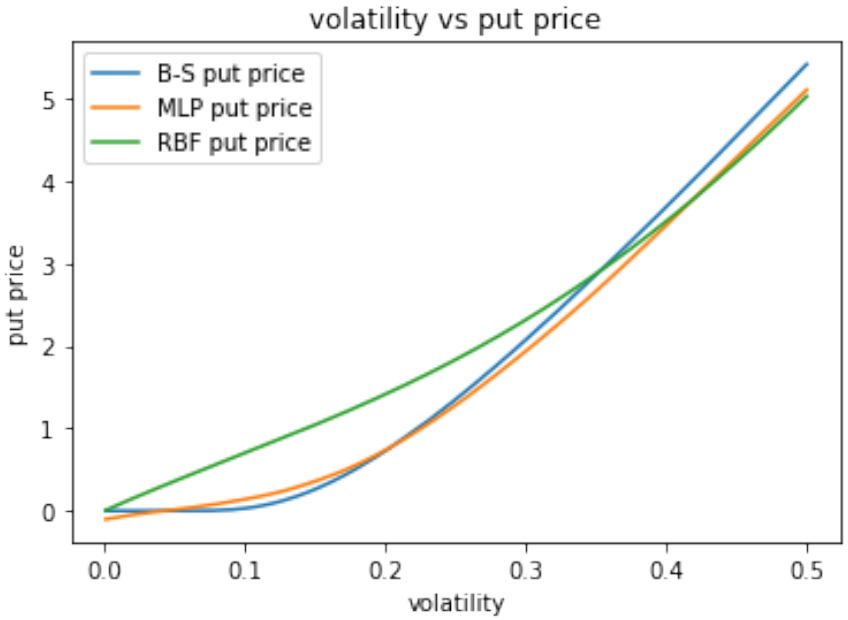


Figure 4.20: Put option prices as volatility is varied

Figure 4.20 shows how the price of the put option rises with increase in volatility.

We can observe that when volatility is low between the prices of put options are constant then begins to constantly increase. This behaviour is observed for all the models with the exception for the RBF model. The MLP and B-S models appear to be more aligned with each other than the RBF model. When volatility is high, it means the price of the underlying asset at the time of exercise can occupy very high values and very low values. When the values are very low, the put option value will be high and when the underlying stock price is high the value of put options is low and can only go as low as 0.

### d. Varying asset price the European call

Figure 4.21 shows the option prices as the asset price is varied. The set of parameter values used are: $S \in (50, 100)$, $K = 75$, $r = 0.3$, $\sigma = 0.3$ and $T = 0.5$.

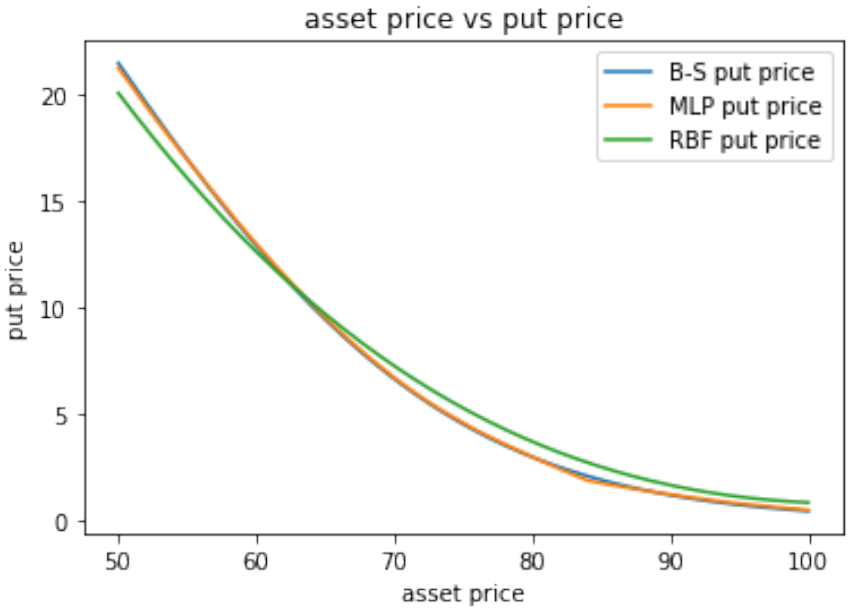


Figure 4.21: Put option prices as the asset price is varied

Figure 4.21 shows how the value of the European put option decreases with increase in stock price.

It can be observe that from the Figure 4.21 that the value of put options are similar for all the models when the stock prices are high. For smaller stock prices the RBF model values are less similar to those of the B-S model. It can be deduced from the graph the observation for both the artificial neural network models compared to the B-S models the MLP model provides a more accurate prediction.

**e. Varying exercise price for the European call**

Figure 4.21 shows the option prices as exercise price is varied. The set of parameter values used are: $S = 75$, $K \in (50, 100)$, $r = 0.3$, $\sigma = 0.3$ and $T = 0.5$.

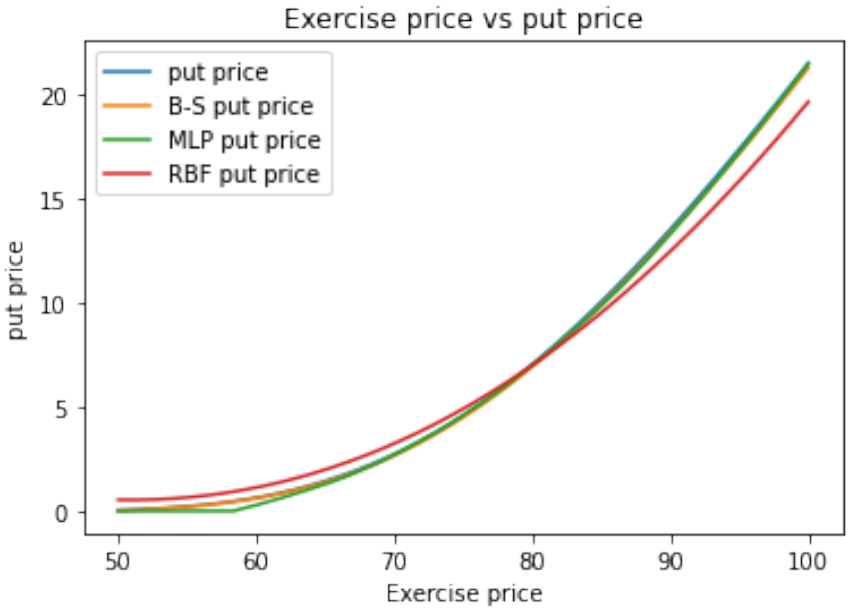


Figure 4.22: Put option prices as the exercise price is varied

Figure 4.22 show how the values of the European put option increases with increase in exercise price.

It can be observed from the figure, that value of put options appear to be very similar for all the models when the stock prices are low. For larger stock prices the RBF model values seem to be different from MLP and B-S values. We can observe for both artificial neural network models values when compared to B-S models values, the MLP provides a more accurate prediction than the RBF.

Table 4.8: Metrics for European put options

| | **BS** | **MLP** | **RBF** |
|---|---|---|---|
| min price | 0 | -0,523237827 | -3,844035275 |
| max price | 45,6418 | 45,41567291 | 50,18777022 |
| min AD | - | 1,9387E-05 | 0,000213493 |
| max AD | - | 3,250326343 | 8,428166112 |
| min SD | - | 3,75855E-10 | 4,55793E-08 |
| max SD | - | 10,56462133 | 71,033984 |

Table 4.8 is a table of metrics for European put options. It displays the maximum and minimum values for put option prices, absolute deviation and squared deviation. Deviation for this table is calculated between artificial neural networks and the Black-Scholes. It is observable that the maximum deviations resulting from the MLP model from the B-S model is lower in magnitude than the deviation resulting the RBF model from the B-S. The minimum deviations in this table are relatively low in magnitude for both the models. The minimum value for MLP put options is relatively closer to the minimum value for B-S model put options compared to the minimum value for the RBF model put options. This observation is similar for the maximum price for put options.

# 5. Conclusion and Recommendations

## 5.1 Conclusion

The study aimed to investigate the pricing of European options using artificial neural networks. It explored a non-parametric method of pricing European put and call options as opposed to the usual analytical methods. The investigation employed two artificial neural networks which are namely the multi-layer perceptron neural network and radial basis function neural network to price the European options and the findings were compared to the analytical Black-Scholes model.

Since option price data is not readily available, the study resorted to simulating the data. The data for both European call and put options were generated by Monte Carlo simulations. Using the data generated from the simulations, the two artificial neural networks were trained over a range of provided option input parameters. The training of the models was assessed using several performance measures including $R^2$, MD, MAD, MSD and SMAPD. After training the neural network models numerical tests to asses the prediction capability of the artificial neural networks were carried out. Neural network model findings from the tests were compared with the performance of the analytical Black-Scholes model. Finally, what-if-analysis on the option prices as some of the option input parameters were varied was performed.

The following is a summary of the main findings from the study:

- The error analysis from the training of the MLP neural network revealed that all performance measures had relatively small magnitudes and that the $R^2$ was close to one for both European call and put options. The error analysis from

the training of the RBF neural network showed that the performance measures were also relatively small in magnitude and the $R^2$ was close to one. These results implied that both models are a good fit. A closer look at the results indicated that the MLP was a better fit to the data as compared to the RBF. Also the use of ReLU activation function in the MLP provided better learning for the MLP model compared the Gaussian radial basis function that was used for RBF model.

- The trained neural network models were used to predict the option prices and the findings were compared to the analytical Black-Scholes model. The obtained maximum and minimum values for both call and put options from the prediction revealed that both maximum and minimum call and put values for MLP are not considerably different from those of the B-S options. Looking further at the minimum and maximum absolute and squared difference for options when using the MLP model it was discovered that these values are relatively low. The RBF model however produced some option prices with considerably sizeable values of maximum absolute and squared error from the analytical values.

- When the parameters of the options were varied in the what-if-analysis, the findings revealed that option parameters affect the option prices the same way for both neural network models and the Black-Scholes model. An increase in the asset price resulted in an increase in the call option price and a decrease in the put option price. When the exercise price was varied, put options prices rose while call options declined with the rise in exercise price. The price of calls and puts grew as the time to expiry and volatility increased. The varying of the risk-free rate of interest increased the value of calls while decreasing the value of puts. As a result, a well-trained artificial neural network model can reproduce Black-Scholes option prices.

## 5.2 Recommendations

Based on the results/findings of this study, the following recommendations can be made:

- Future research could look into different neural networks, such as LSTM, because recurrent neural networks provide better prediction accuracy.
- Increase the number of simulations and observe if there is further improvement to the pricing of options
- Also utilising another analytic model, such as the Heston option pricing model can be considered in this regard. It reflects the dynamics of the volatility parameter, stochastic volatility models provide a better approximate solution for European call and put options.

# Bibliography


Ye, Z. (2013), 'The black-scholes and heston models for option pricing'.

de Freitas, J., Niranjan, M. & Gee, A. (2000), 'Heirachical beyesian models for regularization in sequential learning', *Neural Computation* **12**(4), 955–993.

Que, D. (2019), Option pricing using neural networks, Master's thesis, Science.

Kim, G.-H. & Kim, S.-H. (2019), 'Variable selection for artificial neural networks with applications for stock price prediction', *Applied Artificial Intelligence* **33**(1), 54–67.

Han, S.-H., Kim, K., Kim, S. & Youn, Y. C. (2018), 'Artificial neural network: Understanding the basic concepts without mathematics', *Dementia and Neurocognitive Disorders* **17**, 83.

Cox, J. C. & Ross, S. A. (1976), 'The valuation of options for alternative stochastic processes', *Journal of financial economics* **3**(1-2), 145–166.

Yao, J., Li, Y. & Tan, C. (2000), 'Options price forecasting using neural networks', *Omega* **28**(4), 455–466.

Chen, L. (2020), 'Comparison between different numerical methods in the applications of option pricing', *Neurocomputing* .

Duan, J.-C. (1995), 'The garch option pricing model', *Mathematical finance* **5**(1), 13–32.

Wang, H., Ma, C. & Zhou, L. (2009), A brief review of machine learning and its application, *in* '2009 international conference on information engineering and computer science', IEEE, pp. 1–4.

Jang, H. & Lee, J. (2018), 'Generative bayesian neural network model for risk-neutral pricing of american index options', *Quantitative Finance* **19**, 1–17.

Carr, P. & Linetsky, V. (2006), 'A jump to default extended cev model: an application of bessel processes', *Finance and Stochastics* **10**(3), 303–330.

Björk, T. (2009), *Arbitrage theory in continuous time*, Oxford university press.

Yadav, K. (2018), 'Formulation of a rational option pricing model using artificial neural networks'.

Tsaih, R. (1999), Sensitivity analysis, neural networks, and the finance, *in* 'IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)', Vol. 6, IEEE, pp. 3830–3835.

Kelly, D. L. (1994), 'Valuing and hedging american put options using neural networks'.

Anwar, M. N. & Andallah, L. S. (2018), 'A study on numerical solution of black-scholes model', *Journal of Mathematical Finance* **8**(2), 372–381.

Cuomo, S., Sica, F. & Toraldo, G. (2020), 'Greeks computation in the option pricing problem by means of rbf-pu methods', *Journal of Computational and Applied Mathematics* **376**, 112882.
**URL:** *https://www.sciencedirect.com/science/article/pii/S0377042720301734*

Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C. & Garcia, R. (2000), Incorporating second-order functional knowledge for better option pricing, *in* T. Leen, T. Dietterich & V. Tresp, eds, 'Advances in Neural Information Processing Systems', Vol. 13, MIT Press.
**URL:** *https://proceedings.neurips.cc/paper/2000/file/44968aece94f667e4095002d140b5896-Paper.pdf*

Black, F. & Scholes, M. (1973), 'Pricing options and corporate liabilities', *Journal of Political Economy* **81**(3), 637–654.

Boyle, P., Broadie, M. & Glasserman, P. (1997), 'Monte carlo methods for security pricing', *Journal of Economic Dynamics and Control* **21**(8), 1267–1321. Computational financial modelling.
**URL:** *https://www.sciencedirect.com/science/article/pii/S0165188997000286*

Mahesh, B. (2020), 'Machine learning algorithms-a review', *International Journal of Science and Research (IJSR).[Internet]* **9**(1), 381–386.

Zapart, C. (2002), 'Stochastic volatility options price with wavelets and artificial neural networks', *Quantitative Finance* **2**(6), 487–495.

Taylor, S. J. (1994), 'Modeling stochastic volatility: A review and comparative study', *Mathematical finance* **4**(2), 183–204.

Shreve, S. E. et al. (2004), *Stochastic calculus for finance II: Continuous-time models*, Vol. 11, Springer.

Andina, D., Pham, D., Andina, D., Vega-Corona, A., Seijas, J. & Torres-Garcìa, J. (2007), *Neural Networks Historical Review*.

Shinde, A. & Takale, K. (2012), 'Study of black-scholes model and its applications', *Procedia Engineering* **38**, 270–279. INTERNATIONAL CONFERENCE ON MODELLING OPTIMIZATION AND COMPUTING.
**URL:** *https://www.sciencedirect.com/science/article/pii/S1877705812019480*

Sharma, S., Sharma, S. & Athaiya, A. (2017), 'Activation functions in neural networks', *towards data science* **6**(12), 310–316.

Marvin, M. & Seymour, A. P. (1969), 'Perceptrons', *Cambridge, MA: MIT Press* **6**, 318–362.

Sovzil, D., Kvasnicka, V. & Pospichal, J. (1997), 'Introduction to multi-layer feed-forward neural networks', *Chemom Intell Lab Syst* **39**(1), 43–62.

Gencay & Qi (2007), 'Model risk for european-style stock index options', *IEEE Transactions on Neural Networks* **18**(1), 193–202.

Turhan & Toprak, O. (2013), 'Comparison of high-volume instrument and advanced fiber information systems based on prediction performance of yarn properties using a radial basis function neural network', *Textile Research Journal* **83**, 130–147.

Buscema, M. (1998), 'Back propagation neural networks', *Substance use & misuse* **33**(2), 233–270.

Beckers, S. (1980), 'The constant elasticity of variance model and its implications for option pricing', *the Journal of Finance* **35**(3), 661–673.

Janková, Z. (2018), 'Drawbacks and limitations of black-scholes model for options pricing', *Journal of Financial Studies and Research* **2018**, 1–7.

Abraham, A. (2005), 'Artificial neural networks', *Handbook of measuring system design* .

Hermann, R. & Narr, A. (1997), 'Neural networks and valuation of derivatives-some insights into the implied pricing mechanisms of german stock index options'.

Krenker, A., Bešter, J. & Kos, A. (2011), 'Introduction to the artificial neural networks', *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech* pp. 1–18.

Carelli, A., Silani, S. & Stella, F. (2000), 'Profiling neural networks for options pricing', *International journal of theoratical and applied finance* **3**(2), 183–204.

Dongare, A., Kharde, R., Kachare, A. D. et al. (2012), 'Introduction to artificial neural network', *International Journal of Engineering and Innovative Technology (IJEIT)* **2**(1), 189–194.

Bennell, J. & Sutcliffe, C. (2003), 'Black-scholes versus artificial neural networks in pricing ftse 100 options', p. 1.

Ghaziri, H., Elfakhani, S. & Assi, J. (2000*a*), 'Neural networks approach to pricing options', *Neural Network World* **10**, 271–277.

Ghaziri, H., Elfakhani, S. & Assi, J. (2000*b*), 'Neural, netvvorks approach to pricinc, options', *Neural Network World* **1**(2/00), 271–277.

Hopfield, J. (1982), 'Neural networks and physical systems with emergent collective computational abilities', *Proceedings of the National Academy of Sciences of the United States of America* **79**, 2554–8.

Horasanl, M. (2008), 'Hedging strategy for a portfolio of options and stocks with linear programming', *Applied Mathematics and Computation* **199**(2), 804–810. **URL:** *https://www.sciencedirect.com/science/article/pii/S0096300307010806*

Meissner, G. & Kawano, N. (2001), 'Capturing the volatility smile of options on high-tech stocks- a combined garch-neural network approach', *Journal of Economics and Finance* **25**(3), 276–292.

Anderson, D. & McNeill, G. (1992), 'Artificial neural networks technology', *Kaman Sciences Corporation* **258**(6), 1–83.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), 'Learning representations by back-propagating errors', *nature* **323**(6088), 533–536.

Donaldson, R. & Kamstra, M. (1997), 'An artificial neural network-garch model for international stock return volatility', *Journal of Empirical Finance* **4**(1), 17–46. **URL:** *https://www.sciencedirect.com/science/article/pii/S0927539896000114*

Malliaris, M. & Salchenberger, L. (1996), 'Using neural networks to forecast the sp 100 implied volatility', *Neurocomputing* **10**, 183–195.

Rosenblatt, F. (1960), 'Perceptron simulation experiments', *Proceedings of the IRE* **48**(3), 301–309.

Hutchinson, J., Lo, A. & Paggio, T. (1994), 'A nonparametric approach to pricing and hedging derivative securities via learning networks', *NBER Working Paper* (4718).

Lajbcygier, P. & Connor, J. (1997), 'Improved options pricing using artificial neural networks and bootstrap methods', *International Journal of Neural Systems* **8**(4), 457–471.

Bhattacharya, M. (1980), 'Empirical properties of the black-scholes formula under ideal conditions', *Journal of financial and quantitative analysis* **15**(5), 1081–1105.