University of the Western Cape

Foreign Exchange Option Valuation under Stochastic Volatility

Mbongeni Africa Chamane

UNIVERSITY of the
WESTERN CAPE

A thesis submitted in partial fulfilment of the requirements for the Masters of Science Degree in Computational Finance.

Supervisor: Prof. D Kotze

Co Supervisor: AS Rafiou

March 2009

https://etd.uwc.ac.za/

UNIVERSITY of the
WESTERN CAPE

THES
UNIVERSITEIT VAN WES-KAAPLAND
BIBLIOTEEK
332.45 CHA
LIBRARY
UNIVERSITY OF THE WESTERN CAPE

https://etd.uwc.ac.za/

KEYWORDS

- Black–Scholes–Merton model
- Bachelier model
- Sprenkle model
- Boness model
- Samuelson model
- Multi–assets option
- Monte Carlo integration
- C++ simulation
- Multi–factor Gaussian distribution function
- Wiener process
- Brownian motion vector
- Box–Muller method
- Cholesky factor matrix
- Stochastic volatility models
- GARCH model
- ARCH model
- EWMA model
- SABR model
- Hull–White model

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

DECLARATION

I declare that the Foreign Exchange Option Valuation under Stochastic Volatility is my own work, that it has not been submitted before for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged as complete references.

UNIVERSITY of the
WESTERN CAPE

Mbongeni Africa Chamane

Date: Signature:

31.../ March / 2009.......... [illegible]...............................

https://etd.uwc.ac.za/

ABSTRACT

The case of pricing options under constant volatility has been common practise for decades. Yet market data proves that the volatility is a stochastic phenomenon, this is evident in longer duration instruments in which the volatility of underlying asset is dynamic and unpredictable. The methods of valuing options under stochastic volatility that have been extensively published focus mainly on stock markets and on options written on a single reference asset. This work probes the effect of valuing European call option written on a basket of currencies, under constant volatility and under stochastic volatility models.

We apply a family of the stochastic models to investigate the relative performance of option prices. For the valuation of option under constant volatility, we derive a closed form analytic solution which relaxes some of the assumptions in the Black–Scholes model. The problem of two–dimensional random diffusion of exchange rates and volatilities is treated with present value scheme, mean reversion and non–mean reversion stochastic volatility models. A multi–factor Gaussian distribution function is applied on lognormal asset dynamics sampled from a normal distribution which we generate by the Box–Muller method and make inter dependent by Cholesky factor matrix decomposition. Furthermore, a Monte Carlo simulation method is adopted to approximate a general form of numeric solution.

The historic data considered dates from 31 December 1997 to 30 June 2008. The basket contains ZAR as base currency, USD, GBP, EUR and JPY are foreign currencies.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

Acknowledgements

My primary thanks go to KPMG Financial Risk Management for assistance and providing data, the Quantitative and Economic team within the firm for a zealous support and insightful comments.

I would like to extend a special acknowledgement to Professor Danielle Kotze at the University of the Western Cape for supervising and reviewing the research. I am deeply grateful to my family and friends who supported and encouraged throughout the preparation.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

List of acronyms

ARCH	Auto Regressive Conditional Heteroskedasticity model
BSM	Black–Scholes–Merton model
BM	Box–Muller method
CF	Cholesky Factorization matrix
EWMA	Exponentially Weighted Moving Averages
EUR	European Dollar currency
FX	Foreign Exchange market
GARCH	Generalized Auto Regressive Conditional Heteroskedasticity
GBP	Great British Pound currency
HW88	Hull–White model of 1988
JPY	Japanese Yen currency
MC	Monte Carlo simulation technique
REPO	Repurchase Agreement Offer Rate
SABR	Stochastic Alpha Beta Rho model
ZAR	South African currency

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

List of tables

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

List of figures

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

CONTENTS

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

1 Introduction

Options open a world of opportunities to sophisticated investors. They can adjust to portfolio positions according to any scenario that may arise. A wide scale of using options varies from protecting a position from a decline to betting on the market movement. Their great risks come with great rewards (Aforbes, 2008). For this reason, Hedgers, Speculators, Arbitrageurs, Traders, Quantitative Analysts and Fund Managers are amongst those who enjoy their applications.

Various models have been developed to evaluate options and others are still under development (Yakovenko, 2002) and refer to section 2. This work presents the best practice methodologies and concepts behind pricing options. We restrict attention to the valuation of European call option written on a basket of currencies as underlying asset.

UNIVERSITY of the WESTERN CAPE

Options are financial contracts to buy or sell asset on exchange-traded market or over–the counter market for a pre-determined price (strike price) at a future date (maturity date) (Hull, 2006). Two basic types of options exist, a *call option* and *put option*. The former gives the holder the right to buy the underlying asset; the latter gives the holder the right to sell the underlying asset. Options that can be exercised at any time up to the maturity date are called *American options.* Those that can be exercised only at maturity date are termed *European options*. Since the holder is not obliged to trade the underlying asset, there is a cost to acquiring an option, its value (premium) is derived from the underlying asset, hence the term derivative.

For every option purchased, there is a (writer of option) seller of the option who is accountable if the option is exercised; to deliver the underlying asset, if the option is a *call*, buy the underlying asset, if option is a *put*. The holder and the writer of options have different rights and contrasting views about the future market movement. For

https://etd.uwc.ac.za/

example, the writer of a *call* option who believes that the underlying asset price will be lower than the strike price, is obliged to buy and deliver the underlying asset (if the asset is physical) to the holder if the option is exercised. Conversely the holder of a *call* option who believes that the underlying asset price will be higher than the strike price, has the right but not the obligation to buy the asset from the writer (Wilmott, 1998).

It is relatively risky to write an option[1]; the writer commits to buy shares in the future at a current strike price and to deliver the underlying asset to the holder who may not exercise the option's right. A premium is therefore necessary to compensate for bearing the risk.

The question of how the premium of a contract written on the underlying asset whose future price movement is yet unknown gets decided today, constitutes the main course of this paper. Since the option will be exercised if the payoff is a maximum, it follows that such a right will be purchased at a premium if it is equal to the maximum option payoff expected at a future date, more generally as,

UNIVERSITY of the WESTERN CAPE

$$V(t,S) = E[I(t).\max\{a(S_T - X),0\}] \qquad (1)$$

where *I(t)* is an appropriate discount factor. a is a positive unitary for a *call* and negative for a *put* option; S_T and X are asset price at maturity date and strike price, respectively.

The asset price S is a diffusion process whose evolution is governed by two processes;

$$dS = \mu(t)Sdt + \sigma(S)Sdz \qquad (2)$$

a deterministic process $\mu(t)dt$ and a stochastic process $\sigma(S)dz$. The parameter μ measures the drift in the price, and the square of σ measures volatility of the asset

[1] Investors still find this strategy appealing. Besides collecting premium reward, they also reap benefits from the option payoff – the benefits are limited but risk unlimited (Aforbes,2008).

https://etd.uwc.ac.za/

price. It is noteworthy that the volatility is implicitly defined in the premium function $V(t,S)$ and is not directly observed in the market but can be implied from equation (1) once the premium value is known.

Traditionally, μ and σ are treated as constant parameters so that $V(t,S;\mu,\sigma)$ is a function of only two variables; t and *S*, which means that the volatility is a constant function of time and asset price through out the life of a contract. However, the concept of constant volatility model finds itself at odds with market data. This is mostly evident when implied volatilities are analyzed across the evolution of strike price subjected under a fixed maturity date–as discussed in detail in chapter 5.

When the volatility function was plotted against the prices of the underlying asset; it was found that away from out–of–the–money the volatility function decreases; at–the–money the function is minimal; in–the–money it increases and forms a curve generally known as a volatility smile. It is therefore immediately conclusive that any constant volatility model will be insufficient to fully interpret market phenomena.

UNIVERSITY of the
WESTERN CAPE

Confronted with this reality, researchers admit that the volatility function should contain a deterministic part and a stochastic part (Haug, 2007). Hence, the volatility itself should be modeled as a stochastic process,

$$\sigma = f(t,S)$$

Various models have been brought forward to propose what the function $f(t,S)$ should be. Each one has its own upside and downside. Their solutions are either numerical approximation or analytical, but will end up with a set of parameters to be calibrated from the historic data set. For instance Yakovenko *et al* (Yakovenko, 2002) studied the Heston model by introducing the solution to the Fokker–Planck equation for stock returns on the Dow–Jones index and concluded that it remains to be seen if the solution holds for currency returns.

https://etd.uwc.ac.za/

2 Historic Background of Pricing Derivatives

2.1.1 Bachelier Model

The history of pricing derivatives is an interesting one. It can be traced as far back as March 29, 1900 when a French post graduate student, Louis Bachelier, successfully defended his thesis titled *Theory of Speculation*. As a work of exceptional merit, strongly supported by his supervisor Henri Poincare, it was published in Annales Scientifiques de l'Ecole Normale Superieure, one of the most influential French scientific journals. His analysis of stock and option markets added an enormous value to both finance and probability faculties, and was celebrated by many to an extent that, in 1991, Professor Edward *et al* referred to Bachelier as the father of modern option pricing theory in the Journal of Economic Education publication (Edward, 1991). Most recently in July 2000, a journal of Mathematical Finance considered March 29, 1900 as the birth date of Mathematical Finance (Kabanov *et al*, 2000). In his thesis, Bachelier assumed that the underlying asset price of an option followed a normal distribution as follows,

UNIVERSITY of the
WESTERN CAPE

$$dS = \sigma \times dz$$

where dS is a change in asset price and dz a Wiener process. This implied a positive probability for observing negative asset price–a feature that is not common for stocks and other assets with limited liability aspects. The current option price equals the expected price at maturity date which yields,

$$V(t) = a(S_T - X)N(a \times d_1) + \sigma\sqrt{T-t} \times n(d_1)$$

N and n are cumulative normal and standard normal distributions respectively. S_T, X and a, are as defined previously and

$$d_1 = \frac{S - X}{\sigma\sqrt{T-t}}$$

https://etd.uwc.ac.za/

In the same year, Alan Lewis (Lewis, 2000) argued that Bachelier's most celebrated model became suspect once the ideas of utility theory and risk–aversion became central in economic theory. Lewis questioned that if most investors were averse to risk, shouldn't that influence the price? Shouldn't an option price be worth less than its fair value, just like a stock price?

2.1.2 Sprenkle Model

In light of the flaws in Bachelier's model, Sprenkle extended Bachelier's work in 1960 to a lognormal distribution of asset prices as part of his doctoral thesis at Yale University, published in 1964 (Sprenkle, 1964), thus suggesting that asset prices followed a geometric Brownian motion. Sprenkle rejected normally distributed returns for several stocks based on calculating skewness and kurtosis (Haug, 2007). Furthermore, Sprenkle introduced a drift in the asset price (average growth rate), permitting positive interest rates and risk aversion. He assumed that today's call option value was equal to the discounted expected value at maturity as shown below,

UNIVERSITY of the WESTERN CAPE

$$V(t) = Se^{\rho(T-t)}N(d_1) - (1-k)X \times N(d_2)$$

$$d_1 = \frac{\ln(S/X) + (\rho + \sigma^2/2)(T-t)}{\sigma\sqrt{(T-t)}}$$

$$d_2 = d_1 - \sigma\sqrt{(T-t)}$$

where ρ is the average growth rate of an asset price and k a market risk adjustment factor. Other variables and constants are as defined previously.

2.1.3 Boness Model

In the same year 1964, Boness proposed a lognormal asset pricing model similar to Sprenkle's formula but he recognized the time value of money. He discounted a strike price at maturity date to a present date with the expected rate of return of the asset price and derived the present value of a call option,

https://etd.uwc.ac.za/

$$V(t) = SN(d_1) - e^{-\rho(T-t)}X \times N(d_2)$$

d_1 and d_2 are as defined above.

2.1.4 Samuelson Model

In 1965, Samuelson developed a pricing model similar to that of Boness, but adjusted the share price's average growth constant to include the average growth rate of option prices and expressed a present value of a call option as follows,

$$V(t) = e^{(w-\rho)(T-t)}S \times N(d_1) - e^{-\rho(T-t)}X \times N(d_2)$$

The model takes into account that the expected return w of an option is larger than that of the underlying asset price ρ (Samuelson, 1965), (Smith, 1976). It can be remarked that the model features resemble the Black–Scholes–Merton (BSM) model discussed in detail in the following section.

UNIVERSITY of the WESTERN CAPE

2.2 Black–Scholes–Merton Model

The BSM model is the most widely used formula in many disciplines. It has a wide domain of application in everyday use amongst Traders, Market Makers, Sales Specialists, Quantitative Analysts (Merton *et al*, 1994), (Finnerty, 2005) and (Jondeau *et al*, 2007), and is applied in several academic disciplines (Malliaris, 1993) and (Hang Chan *et al*, 2006). Researchers observe that derivative markets have been the fasted growing financial institutions of the century and are expected to see a dramatic growth over the next few years as documented in the *Beyond Credit Crisis* in KPMG publication (Seymour *et al*, 2008).

In 1997 Myron Scholes and Robert Merton were awarded the Nobel Prize (by the bank of Sweden in Economic Sciences in memory of Alfred Nobel). Sadly, Fischer Black died in 1995 before he also would have received the prize (Haug, 2007). It is believed that it was not the option pricing formula itself that resulted in the award of Nobel Prize

https://etd.uwc.ac.za/

as the formula was already invented–as notable in the above history. It was the way it was derived, with impressive consequences in continuous time dynamic delta hedging arguments, and compatibility with the Capital Asset Pricing model that led to the honour.

It can be argued that the popularity amongst traders for using the option pricing model heavily relies on its hedging strategy (Derman and Taleb, 2005).

The BSM model can be derived by considering a portfolio which longs an option *V(S, t)* and shorts a quantity of n underlying assets S at time t. The change in value of the portfolio P as a result of this trade can be written as follows,

$$dP = dV - ndS$$

Substituting dS from equation (2) in the above equation and invoking Ito' Lemma calculus, the evolution of the portfolio under time and asset price variations can be written as,

UNIVERSITY of the
WESTERN CAPE

$$dP = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}dt - ndS \qquad (3)$$

2.2.1 Delta Hedging Argument

Observing that the changing portfolio process dP depends on two variables, namely; the deterministic process dt and the two stochastic processes dS, the random surprise dS which subjects the portfolio under risk can be eliminated by choosing the traded quantity n such that, $n = \frac{\partial V}{\partial S}$

This reduction of randomness is referred to as the *delta hedging strategy* (Wilmott, 1998). Thus, the portfolio changes are absolutely risk–less and can be quantified as,

$$dP = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt$$

https://etd.uwc.ac.za/

2.2.2 No Arbitrage Argument

The development of the no arbitrage opportunities argument can be considered as the consequence of delta hedging. Since the portfolio remains risk–less in continuous hedging, it follows that any fund invested in such a portfolio would also experience a risk–free growth at a rate *r*, and therefore the no arbitrage principle states that the portfolio grows at a risk–free rate such that there are no arbitrage opportunities as follows,

$$dP = rPdt\,,$$

The two arguments immediately prompt the BSM Partial Differential Equation (PDE),

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \qquad (4)$$

The BSM's PDE as any linear partial differential equation has a closed form solution, and interprets how option evolves under change in time and underlying asset. Further, it forms the foundations of derivative theory. The option's payoff value at maturity given by equation (1) can now be discounted in the risk–neutral world using equation (4) to predict today's value to yield,

$$V(t,S) = aS \times N(a \times d_1) - aX \times e^{-rT} N(a \times d_2)$$

where $$d_1 = \frac{\ln(S/X) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$$

and $$d_2 = d_1 - \sigma\sqrt{T}$$

$$V(t,S) = e^{-r(T-t)} E(payoff) \qquad (5)$$

The result means that the present value of an option can be found by discounting with risk–free rate its maximum payoff expected at the maturity date.

https://etd.uwc.ac.za/

3 Valuation of Multi–Assets Option

Having reviewed an option pricing formula for a single reference asset, we now wish to evaluate an option whose reference asset is a basket of four currencies. Paul Wilmott described such a multi–assets option as the *rainbow option* (Wilmott, 1998). To price the derivative, we reconstruct a BSM formula under the framework of a multi-dimensional assets dynamics approach. The lognormal random walk asset diffusion process is extended to the multi–dimension case,

$$dS_i = \mu_i(t)S_i dt + \sigma_i(S)S_i dz_i$$

where S_i is the price of the i^{th} asset with drift μ_i and volatility σ_i, i=1 ...to K reference assets. The component dz_i follows a Wiener process such that,

$$E(dz_i) = 0$$
$$E(dz_i^2) = dt \quad and \quad E(dz_i . dz_j) = \rho_{ij} dt$$

UNIVERSITY of the WESTERN CAPE

Moreover, the correlation matrix is positive semi-definite[2], where the volatility of individual assets is integrated into single portfolio volatility using the covariance matrix as,

$$(\alpha_j)(\rho_{ij})(\alpha_i)$$

Define:

$\underline{\underline{\rho}} \equiv (\rho_{ij})$: as the correlation matrix.

(α_i) : as a diagonal matrix of σ_i elements.

[2] Such that $\vec{y}^{T} (\rho_{ij})\vec{y} \geq 0$

https://etd.uwc.ac.za/

By invoking a multi–dimensional Ito's lemma calculus the value of the option under many assets can be represented as follows (Wilmott, 1998),

$$\frac{\partial V}{\partial t}+\frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k}\sigma_i\sigma_j S_i S_j\frac{\partial^2 V}{\partial S_i \partial S_j}+\sum_{i=1}^{k}(r^d - r_i^f)S_i\frac{\partial V}{\partial S_i}-r_d V=0$$

where r^d and r_i^f are domestic and foreign interest rates respectively. This is a version of the BSM equation in higher dimensions. Using Green's function (Silverman, 1999) and (Dorfleitner *et al*, 2008), the solution can be written as,

$$V(t,S_1,\ldots,S_k)=\frac{e^{-r^d(T-t)}}{(\sigma_1.\sigma_2,\ldots,\sigma_k)[2\pi(T-t)]^{k/2}\sqrt{(Det\,\underline{\underline{\rho}})}}\times$$

$$\int_0^\infty\ldots\int_0^\infty\frac{payoff(S_1,\ldots,S_k)}{S_1,\ldots,S_k}e^{(-\frac{1}{2}\bar{\alpha}^T\underline{\underline{\rho}}^{-1}\bar{\alpha})}dS_1,\ldots,dS_k$$

where $$\alpha_i=\frac{1}{\sigma_i(T-t)^{1/2}}\left(\log\left(\frac{S_i}{X_i}\right)+\left(r^d-r_i^f-\sigma_i^2/2\right)(T-t)\right) \qquad (6)$$

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

4 Principles of Stochastic Volatility Models

We review the principles behind the latest methodologies of modeling stochastic volatilities and present their strengths and flaws as informed by market data.

Based on the ability to explain market observations and the ease of implementation, we select a set of the best practice methodologies which is adopted throughout option valuation from which the relative performance results are compared and remarked upon.

4.1 EWMA Model

Exponentially Weighted Moving Averages (EWMA) method estimates the current level of volatility of market variables from historic data. The weighting scheme exponentially assigns more weight to the more recent observations (Hull, 2006).

Let there be volatility of a market variable on day *n* estimated at the end of day *n-1* defined as σ_n^2. Then EWMA can be written as,

UNIVERSITY of the
WESTERN CAPE

$$\sigma_n^2 = \lambda\sigma_{n-1}^2 + (1-\lambda)u_{n-1}^2$$

Define:

λ : a constant parameter between *0* and *1*

μ_{n-1} : a return of the market variable at the end of day *n-1*

The estimate of volatility on day *n* is based on the estimates of volatility and return that were made on day *n-1*. This EWMA approach can be developed by modifying a standard variance equation of *m* recent observations defined as,

$$\sigma_n^2 = \sum_{i=1}^{m} P_i(u_{n-i} - \overline{u})^2 \qquad (7)$$

https://etd.uwc.ac.za/

P_i is the amount of weight given to the observation i days ago, and as such, satisfies the requirement,

$$\sum_{i=1}^{m} P_i = 1$$

Assuming that $\bar{\mu}$ is zero and adding a weight γ of a long–run variance V, equation (7) becomes,

$$\sigma_n^2 = \gamma V_L + \sum_{i=1}^{m} P_i u_{n-i}^2 \qquad (8)$$

Since the total weight must be unity, the condition that $\gamma + \sum_{i=1}^{m} P_i = 1$ prevails.

Equation (8) is known as an ARCH (m) model. The EWMA model is born when the weight $P_{i+1} = \lambda P_i$ is inserted into equation (8).

The model tracks changes in volatility and is exponentially sensitive to λ, a low value of the parameter leads to more weight being given to the more recent observations and subsequently high volatility estimates. A high value of λ leads to estimates that cause slow response to new information (Hull, 2006).

The advantage of the model is that not much history of market data is required since the volatility is estimated with the previous day estimates. The chief disadvantage is that the model is a non–mean reversion process. In the long run the variance does not get pulled back to the mean variance as it would be expected from a robust model.

4.2 GARCH (1, 1) Model

ARCH was first suggested by R. Engle when he was visiting London Stock Exchange in 1979 and published in 1982. His student Tim Bollerslev developed the GARCH model, the generalization of ARCH during his doctoral thesis; it was published in 1986 (Bollerslev, 1986).

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

A GARCH (1, 1) equation can be expressed as,

$$\sigma_n^2 = \gamma V_L + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2 \qquad (10)$$

where $\gamma + \alpha + \beta = 1$

When $\gamma = 0$, $\sigma = 1 - \lambda$ and $\beta = \lambda$ the equation reduces to EWMA. The GARCH (1, 1) model is a member of the family of stochastic processes,

$$dV = a(V_L - V)dt + \xi V dz\,,$$

where $a = 1 - \alpha - \beta$, and $\xi = \alpha\sqrt{2}$

This model has a mean reversion property. A drift pulls the variance V back to the long–run variance mean V_L at a relaxation time $\frac{1}{a}$, ξ is a dragging coefficient of the volatility. These features make GARCH more appealing than the EWMA model.

UNIVERSITY of the WESTERN CAPE

4.2.1 Calibrating GARCH (1, 1) Parameters

The parameters can be calibrated from the data series using the maximum likelihood estimator method. This is an iterative procedure which optimally selects parameters that maximize the likelihood of the observed data series (Hull, 2006).

The likelihood of occurrence of m independently and identically normally distributed observations can be written as,

$$\prod_{i=1}^{m} p_i = \prod_{i=1}^{m}\left[\frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\left(-\frac{u_i^2}{2\sigma_i^2}\right)}\right] \qquad (11)$$

https://etd.uwc.ac.za/

The best parameters of equation (10) are the ones that maximize equation (11). This is equivalent to maximizing[3]

$$\sum_{i=1}^{m}\left[-\ln(\sigma_i^2)-\frac{u_i^2}{\sigma_i^2}\right]$$

Starting at day three $\sigma_3^2 = u_2^2$, we can find the parameters that maximize the above equation by searching iteratively.

4.2.2 GARCH (1, 1) Volatility Forecast

Given a current level of volatility estimated from the previous day. Based on the projection of expected future volatility and return and by a recursive valuation of the discounted expectations, we forecast future volatility by appropriately selecting the accumulated weight parameters such that the expected variance reverts to the long–run variance.

UNIVERSITY of the WESTERN CAPE

Let there be a current level of volatility σ_n^2 estimated from the previous day's volatility σ_{n-1}^2 and return u_{n-1}^2. Equation (10) can be expressed as,

$$\sigma_n^2 - V_L = \alpha(u_{n-1}^2 - V_L) + \beta(\sigma_{n-1}^2 - V_L)$$

Define the projections into time t of volatility and return as σ_{n+t}^2 and u_{n+t}^2 respectively. The expected levels become,

$$E[\sigma_{n+t}^2 - V_L] = \alpha(E[u_{n+t-1}^2] - V_L) + \beta(\sigma_{n+t-1}^2 - V_L)$$

Recognizing that $E[u_{n+t-1}^2] = \sigma_{n+t-1}^2$ the expression becomes,

[3] Since it is easier to work with summation rather than multiplication, log operation is applied on equation (11) to yield the summation when constants are ignored.

https://etd.uwc.ac.za/

$$E[\sigma^2_{n+t} - V_L] = (\alpha + \beta)(\sigma^2_{n+t-1} - V_L)$$

By recursively taking the expectation values, -*t* times, we arrive at future expectation of volatility which can be written as,

$$E[\sigma^2_{n+t}] = V_L + (\alpha + \beta)^t(\sigma^2_n - V_L) \qquad (12)$$

If we choose $\alpha + \beta < 1$, the mean reversion property is preserved such that

$$E[\sigma^2_{n+t}] \to V_L \qquad as \qquad t.... >> \text{Large}$$

4.3 Hull–White Models

Hull and White adopted two different approaches to address the issue of a stochastic volatility model. The first approach considered a case where the asset price dynamics and volatility dynamics have uncorrelated Brownian Motions. The second approach solved the case where the asset price and volatility dynamics are correlated.

UNIVERSITY of the
WESTERN CAPE

4.3.1 Uncorrelated Stochastic Volatility Model

In 1987, Hull and White valuated options under a special case where the volatility processes is uncorrelated with asset price dynamics.

$$dS = bS_t dt + \sigma S_t dz$$
$$dV = \alpha V dt + \xi V dw$$
$$\rho_{SV} = 0$$

The asset price *S* has a drift b and variance $V = \sigma^2$. The variance *V* has a drift α and variance ξ. Hull and White (Hull-White, 1987) expressed the discounted payoff of an option as the BSM formula with the mean variance $\overline{V}$ distributed over the life time of the option as follows,

https://etd.uwc.ac.za/

$$\int c(\overline{V}) \times h(\overline{V} / \sigma_t^2) d\overline{V} \tag{13}$$

where $\overline{V} = \frac{1}{T} \int_t^T \sigma_t^2 dt$

Here the asset price S is not affected by the drift of the variance α and the volatility of variance ξ.

Rather than finding the close form solution of equation (13). Hull–White calculated the moments of $\overline{V}$ and showed that under small values of ξ the Tailor series of the equation (13) converges to an approximate solution.

4.3.2 Correlated Stochastic Volatility Model

Hull and White realized that it is not possible to mix derivatives in a model where there is no correlation between asset price and volatility. In 1988 they proposed a model where the asset price process is correlated to the stochastic volatility dynamics. In this model asset price would be explicitly sensitive to the square root volatility.

$$\begin{aligned} dS &= bS_t dt + \sqrt{V} S_t dz \\ dV &= (a + \beta V)dt + \xi \sqrt{V} dw \\ \rho_{SV} &\neq 0 \end{aligned} \tag{14}$$

Where a and β are constants that determine the drift of the variance. This is a mean reversion stochastic process that pulls back the variance to the mean level $-\frac{a}{\beta}$ at a speed $-\beta$. They derived a closed–form approximation solution for a European call option using a Tailor Series expansion.

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

Hull–White models achieve to explain stochastic volatility very well. However, given the Tailor series expansion approach, there's a sharp limit to the degree at which the parameters are used. This has largely caused the models to be unpopular in practice.

4.4 SABR Model

The SABR stands for Stochastic Alpha, Beta and Rho, the parameters of the model derived by inverting the Black 76 option pricing model. Traders bypass the assumption of constant volatility in the BSM model by using a different volatility for every strike price when calculating an option price. Hagan *et al* published the Managing Smile Risk paper in which they derived a SABR model that accurately predicts a number of market smiles in option currencies (Hagan, 2002). They discovered that the market smile managed by using local volatility models was the opposite of the observed market behavior. The two factor SABR model states that the forward value of asset is related to the volatility as follows,

$$\begin{aligned} dF &= \alpha F^{\beta} dz \\ d\alpha &= \xi \alpha dw \end{aligned} \qquad (15)$$

UNIVERSITY of the
WESTERN CAPE

F is the future/forward price, β is a constant determining the distribution of asset price, α is the volatility of forward price, ξ is the volatility and dw and dz are two correlated Wiener processes. A singular perturbation technique was used to arrive at an analytical solution of volatility as a function of forward price. The analytical solution is then inserted into the BSM model to calculate the option price. The results can be elegantly expressed as follows,

`

$$\sigma(F;\beta,\alpha,\rho,\xi) = \phi_1(F)\times\phi_2(F)\times\phi_3(F) \qquad (16)$$

where β,α,ρ,ξ parameters are calibrated from the market data.

https://etd.uwc.ac.za/

Equation (16) is then inserted into the BSM equation (5) to calculate option premium. The SABR model hence becomes an extension of the BSM model adjusted to stochastic volatility.

4.4.1 Calibrating SABR Parameters

Normally in the FX markets where lognormal terms are assumed, β is chosen to be unity and zero for the normal terms. The parameters are calibrated in such away that the SABR volatility matches the at–the–money market volatility. By calculating at–the–money volatility from equation (16) when the strike price equals the forward price, (West, 2005) showed that α becomes a cubic equation whose singular root can be found by the Vie'te method.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

5 Existence of Stochastic Volatility and its Effects in Foreign Exchange Markets

This chapter motivates the necessity of adopting the stochastic volatility approach when evaluating options. With empirical data, we show that an underlying asset of an option should have non–constant volatility and then establish that any constant volatility model would be insufficient to fully interpret market observables. With these foundations, we apply the best practice stochastic volatility methodologies introduced in the preceding chapter for valuation.

When the volatility of currency prices was analyzed in real–time, a spectrum of high and low volatility points dominated the entire period– see figure 5.1. Stochastic trends (volatility jumps) were observed across the four major currencies; Sterling (GBP), ZAR, Euro and the Yen (JPY), with US Dollar as a base currency. This contrasted with the smooth curve or even a horizontal line expectation as the conventional methods of derivative pricing would have assumed.

UNIVERSITY of the
WESTERN CAPE

Figure 5.1 shows the volatility versus time graph constructed on a 60-day daily moving EWMA volatilities of log returns of currency over a period of 2 years ending 30 June 2008. The EWMA method applies more weight to the more recent observations as discussed in detail in the stochastic volatility section 4.1.

https://etd.uwc.ac.za/

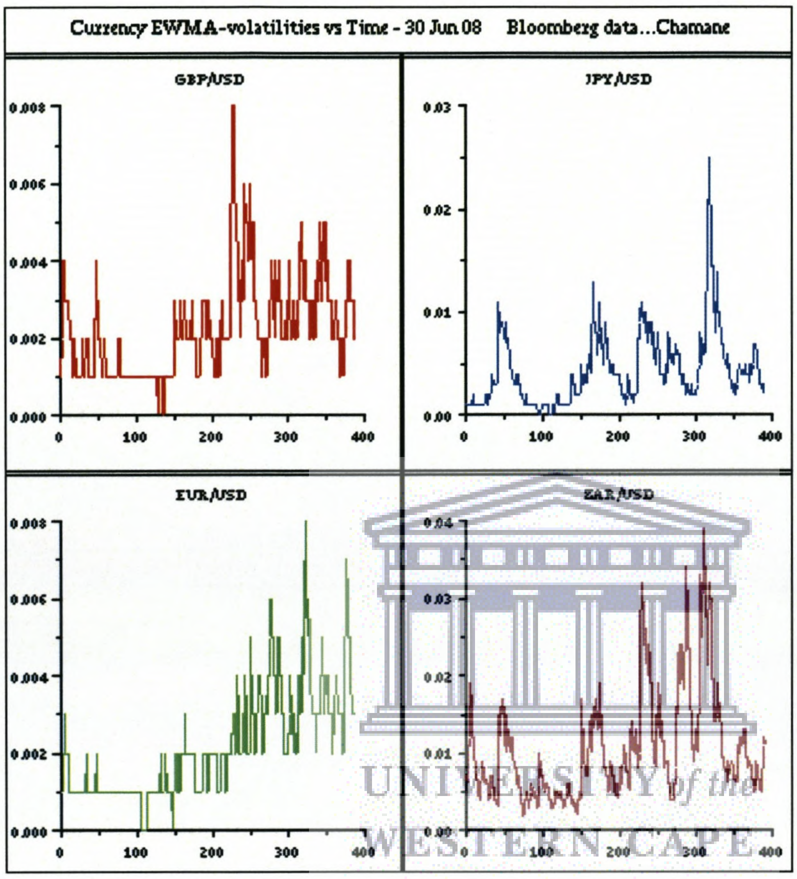

Figure 5.1 EWMA volatilities of currency prices

5.1 Volatility Smiles vs. Constant Volatility

While the volatility of exchange rate returns proved to be stochastic in real–time its behavior and effect on an option was fully manifested when the volatility calculations were implied from the option values at various strike price levels under a fixed maturity date.

An interesting picture was formed when the resultant implied volatility curves were compared with the constant or standard volatilities[4] that were used for pricing the option

[4] The standard volatilities result from an adjustment of constant volatility based on traders observation of market movement

https://etd.uwc.ac.za/

obtained from market data. The volatility corresponding to a call option rapidly decreased when the option was out–of–the–money and reached the minimum value at–the–money; the volatilities which corresponded to the put option in–the–money increased vigorously completing the formation of a curve known as the *implied volatility smile*.

Significant contrasts were drawn from the standard volatilities and the implied volatilities. It can be observed that standard volatilities are constant on average with respect to the changes in strike price, and form a horizontal line across the axis of the strike price. The implied volatilities formed a smile curve, almost a parabola with respect to changes in strike price–refer to figure 5.2.

The implied volatilities were calculated by applying equation (1) on the market data of strike prices and European options written on the EUR/USD currency for 3 months and 2 years maturity dates. On the figure the 3 months curve option appears on the major axis and the 2 years curve on the minor axis. The blue curves are the ranges of implied volatilities; the red lines are standard volatilities. Note that a declining curve corresponds to a call option, and to a put option on the incline.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

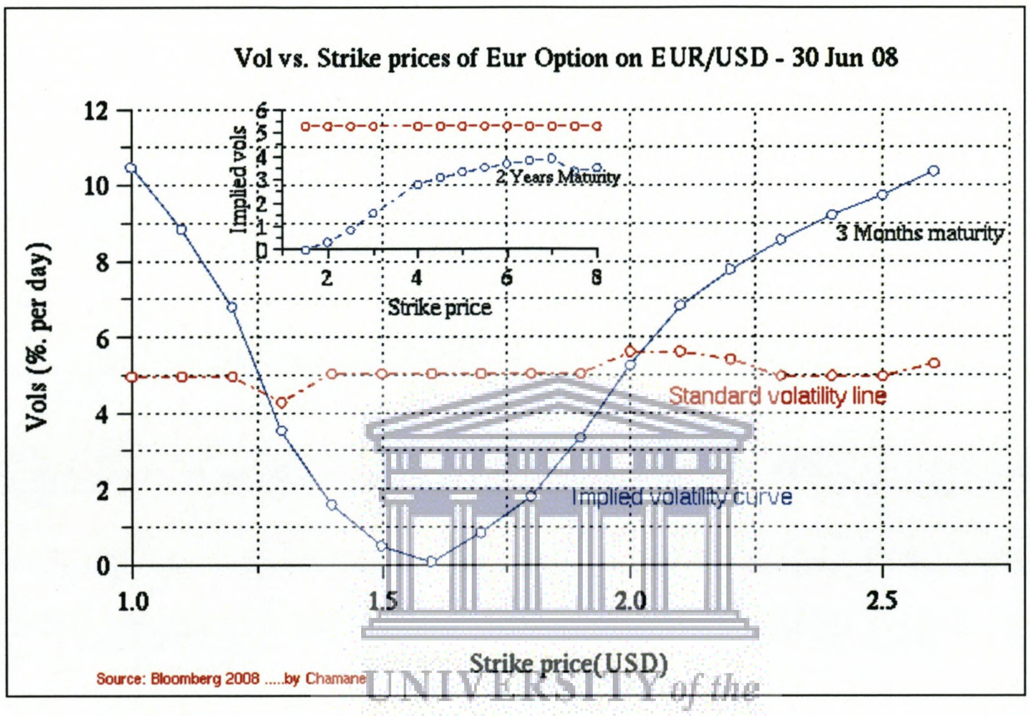

Figure 5.2 Volatility curves of option vs. strike

UNIVERSITY of the WESTERN CAPE

5.2 Conclusion

Based on the contrasting results of standard volatilities used for pricing and the implied volatilities calculated from the actual option prices, it is conclusive that the effect of low volatility increases the strike price of a call option. The effect of high values of volatility increases the strike price of a put option. These conclusions cannot be made by analyzing a constant volatility model alone.

https://etd.uwc.ac.za/

6 Valuation Methodology

Given a spot price, drift and volatility of an asset at time *t*, we wished to know what the asset price at maturity date *T* would be. Since the evolution of a single asset price is given by,

$$dS = \mu(t)Sdt + \sigma(S)Sdz$$

the solution can be expressed as,

$$S(t+dt) = S(t)\exp\left[\left(\mu - \sigma^2/2\right)dt + \sigma\eta\sqrt{dt}\right] \qquad (17)$$

Because the distribution of the asset price is based on the theory of random walk, more specifically, on Brownian motion which requires the normality on the asset price distribution (Wilmot, 1998), we created a random number generator which generates normal random variables η that were then used to sample equation (17). This was achieved by introducing a Box–Muller method (see section 6.1).

However since we are dealing with the case of multi-assets, the Brownian motion of each asset price from time *t* to time *T* is correlated with other asset prices in the portfolio, and the evolution of each asset price in the multi-assets dimension follows the process,

$$S_i(t+dt) = S_i(t)\exp\left[\left(\mu_i - \sigma_i^2/2\right)dt + \sigma_i\phi_i\sqrt{dt}\right] \qquad (18)$$

where $\mu_i = r^d - r_i^f$ and $E[\phi_i\phi_j] = \rho_{ij}$

For a single asset case, ϕ is an independently and normally distributed Brownian motion variable, but for a multi–assets case, ϕ_i is a normally distributed random variable correlated with the random variables of other assets.

https://etd.uwc.ac.za/

To apply equation (18) we needed to first generate a random vector $\bar{\phi}$ of correlated and normally distributed elements ϕ_i. This was achieved by generating a Cholesky factorization matrix (see section 6.2).

6.1 Box–Muller Method

The Box–Muller (BM) is a method in which descent normally distributed random variables can be generated from two uniformly distributed random variables between zero and one (Wilmott, 1998) as follows,

$$y_1 = \sqrt{-2\ln x_1} \times \cos(2\pi x_2) \text{ and } y_2 = \sqrt{-2\ln x_1} \times \sin(2\pi x_2) \tag{19}$$

where x_1, x_1 are uniformly distributed and y_1, y_2 satisfy normality. Of special interest is that y_1 *and* y_2 are uncorrelated since sine and cosine functions are orthogonal to each other. The variables y_1 *and* y_2 then create a vector $\bar{\eta}$ whose elements are uncorrelated and normally distributed as we desired.

UNIVERSITY of the
WESTERN CAPE

6.2 Cholesky Factor Matrix

We further seek to transform the uncorrelated Brownian motion vector $\bar{\eta}$ into a correlated vector $\bar{\phi}$ such that,

$$\bar{\phi} = \underline{\underline{C}}\bar{\eta} \tag{20}$$

To find the matrix $\underline{\underline{C}}$ we factorized the semi–positive definite correlation matrix $\underline{\underline{\rho}} \equiv (\rho_{ij})$ into two factors,

$$\underline{\underline{\rho}} \equiv \underline{\underline{C}} \times \underline{\underline{C}}^T.$$

To show that this transformation works, combine equations (18) and (19) to yield,

$$E(\phi_i \phi_j) = \underline{\underline{C}} E(\bar{\eta}\bar{\eta}^T)\underline{\underline{C}}^T \qquad \Rightarrow \underline{\underline{\rho}}$$

https://etd.uwc.ac.za/

It follows that $\underline{\underline{C}}$ can be found. This matrix is known as the Cholesky Factor Matrix (CF).

6.3 Monte Carlo Simulation Method

The Monte Carlo (MC) simulation method solves probabilistic numerical problems using generated suitable random numbers by picking randomly distributed points in a multi–dimensional volume to determine the integral of a function (Hammersley, 1960). It was named by the mathematician S. Ulam in 1946 after a relative who had the habit to gamble (Hoffman, 1998), (Glasserman, 2003).

It is almost impossible to evaluate multi-integrals in higher dimensions; granted that in this work we intensively evaluate derivatives written on four currencies given by equation (6). We thus use an (MC) integration technique which can be pursued as follows;

We evaluate an integral over a volume $dS_1....dS_k$. The MC integration allows the integral to be written as,

UNIVERSITY of the
WESTERN CAPE

$$\int....\int f(S_1....S_k)dS_1....dS_k = volume \times average.f$$

Then rescale the integration to unity and yield,

$$\int_0^1....\int_0^1 f(S_1....S_k)dS_1....dS_k = average.f$$

Then sample the average value of f using MC simulation on the distribution of S such that after N samplings, the average f is approximately,

$$f \approx \overline{f} = \frac{1}{N}\sum_{i=1}^{N} f(S_i)$$

The error in the sampling can be measured by the standard deviation

https://etd.uwc.ac.za/

$$s = \sqrt{\frac{1}{N}(\overline{f^2} - \overline{f}^2)} \qquad (21)$$

The equation (21) states an important property of MC simulation which means that if we need to double the accuracy of estimating the option value, we need to quadruple the number of simulations.

To evaluate the call option with MC simulation, we make the option to be path dependent; divide the path into discreet time steps Δt and computing the value of each currency price from time *S(t)* to maturity time *S(T)*. The number of simulations required to arrive at *S (T)* is then $M = T / \Delta t$.

We henceforth repeat the sampling of *S (T) N* times and proceed to calculate the call option average payoff as follows,

$$\text{Average payoff} = \frac{1}{N}\sum_{i=1}^{N} \max(S_i(t) - X, 0)$$

The value of the option can then be written as,

UNIVERSITY of the
WESTERN CAPE

$$V = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} \max\left[\left(S_{j-1}\exp(r^d - r_i^f - \sigma_{ij}^2/2)\Delta t + \sigma_{ij}\phi_{ij}\sqrt{\Delta t}\right) - X, 0\right] \qquad (22)$$

6.4 Summary of Methodology

Once the maturity date was chosen, the valuation was carried out as follows:

- Pick a three months holding period to form a sample path from 30 June 2008 to 30 Sep 2008.
- On each day for each currency, generate two independently and normally distributed random vectors using the BM method,

https://etd.uwc.ac.za/

$BoxMuller[0], BoxMuller[1]$.

- Apply the BM vectors to create a correlated Brownian motion vector

  $coBrown[t]$.

  - Create a Wiener process using the two Box –Muller vectors as follows,

$$Wiener[0] = BoxMuller[0]$$
$$Wiener[t] = BoxMuller[0] + BoxMuller[1] \times \sqrt{dt}$$

  - Calculate a correlation matrix for the latest date, and apply CF to obtain factor C.

  - A correlated Brownian motion vector was then obtained by taking the product of Cholesky matrix and Wiener process vector.

- Apply volatility, correlate Brownian motion vector elements found from above, and the risk-free rate as input to simulate the asset value at maturity as described in the MC simulation (see section 6.3).

- Calculate the payoff at maturity date, and discount in the risk-neutral to obtain the present value option price.

UNIVERSITY of the
WESTERN CAPE

6.5 Modeling Framework

C++ has been adopted as a suitable programming language to overcome some of the serious deficiencies of Monte Carlo (MC) simulation as implemented throughout the valuation.

6.5.1 Software Speed and Reusability Properties

The deficiencies in the MC method stem from the fact that the accuracy of valuation is inversely proportional to the square root of the number of simulations, as discussed in detail in section 6.3, refer to the equation (21). This means that to double accuracy the

https://etd.uwc.ac.za/

number of simulations must be quadrupled. For instance, in the case of valuing a single asset option, 10000 simulations were required to yield a sensible value on average. This implied that 40000 simulations would be required to double the accuracy in order to avoid inheriting errors. Given that we ran simulations in high dimensions of multi–assets, the actual number was much higher than 40000 such that high speed computation became a necessity.

$$error \propto \frac{1}{\sqrt{N}} \quad \text{......from equation (21)}$$

As a compiled language, C++ meets these requirements. Interpretive languages such as VBA that translate source code instructions directly into the machine takes up to a day of running. The computational speed in C++ is derived from the fact that instructions are compiled directly into intermediary forms of independent object files that are then linked to the machine as executable codes (Stroustrup, 1997). This saw the overall number of 60000 simulations taking about 15 seconds to complete runs. Because the source codes are compiled into executable objects, software reusability is well permitted; when we wish to reuse the codes in other programs we simply link them in the make files.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

6.5.2 Software Structure

The main structure of the software flow and the implemented methods are presented below.

Table 6.5.2a MARKET DATA

Function Name	Arguments	Descriptive Performance
dataParameters ()	Spot price Strike price Interest rate	Uploads market data file from local directory. Activates arguments for global use
dataSeries ()	PriceVector ReturnVector Timeseries length	Activates arguments for global use.
discountCurve ()	ZarDF	Upload yield curve from local directory. Computes discount curve for global use.

Table 6.5.2b UTILITY FUNCTIONS

Function Name	Arguments	Descriptive Performance
normalRV ()	BoxMullerVector	Generates Box –Muller normal random vectors.
uncorBrownM ()	Wiener	Create Wiener process vectors of all assets.
Risks ()	corrMatrix varVector	Computes correlation matrix and variance using vol link method.
Cholesky ()	corrMatrix choleskyMatrix	Accepts correlation matrix. Output Cholesky matrix for global use.
corBrownM	corBrownVector	Generate correlated Brownian motion vectors.
nAssets ()	All generated arguments	Performs valuation on each asset and on the basket. Generates associated errors.

Table 6.5.2.c STOCHASTIC VOLATILITY MODELS

Function Name	Arguments	Descriptive Performance
EWMvol()	PriceVector	Uses price series to create volatility vectors.
GARCHvol()	PriceVector	
Hull-Whitevol()	PriceVector	
SABRvol()	PriceVector	

Table 6.5.2.d VALUATION MODEL

Function Name	Arguments	Descriptive Performance
All created functions	All activated arguments	Performs option valuations.

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

6.6 Key data input

Presented below are the key data input tables crucial to the model valuation which were obtained from the FX markets through Bloomberg data services.

Table 6.6.a show exchange rates with USD as the base currency, however the repo rate is in ZAR currency.

Table 6.6.a. Key data input parameters

Call currency	Put currency	Spot price	Strike price	Notional	Foreign interest	Domestic interest(S.A)
ZAR	USD	7.8199	8.0147	1 000 000.00	2.78%	11.80%
GBP	USD	1.9923	1.9785	1 000 000.00	5.61%	11.80%
JPY	USD	106.22	105.7	1 000 000.00	0.84%	11.80%
EUR	USD	1.5755	1.5682	1 000 000.00	4.62%	11.80%

Table 6.6.b displays valuation and maturity dates of European options.

Table 6.6.b Maturity dates

European option	Dates
Valuation date	30/06/2008
Premium date	30/06/2008
Maturity date	3 months
Expiry date	09/30/08

UNIVERSITY of the WESTERN CAPE

Table 6.6.c shows a time series forecast of the domestic interest rates beyond the maturity date which is used as the risk-free discount rate in the valuation Sourced-Bond Exchange of South Africa (BESA).

Table 6.6.c. ZAR swap rates:

ZAR Swap Curve	
30-Jun-08	11.80
01-Jul-08	11.81
03-Jul-08	11.81
07-Jul-08	11.83
30-Jul-08	11.91
29-Aug-08	12.00
30-Sep-08	12.10

The history of 8 years' daily prices series for the exchange rates is used in the calibration–see appendix A.

https://etd.uwc.ac.za/

7 Portfolio Valuation Results, Analysis and Remarks

A bottom up approach was adopted in calculating the option basket in which the payoff of asset at maturity date was obtained by Monte Carlo simulating asset path using equation (20), and summing all the resultant payoffs weighted proportionally to yield an integral payoff of the basket. To obtain the present value of the basket, the integrated payoff was discounted in the risk–neutral world using the ZAR swap zero–coupon bond curve–this is shown in table 6.6.c. The errors estimated were then calculated from equation (21) for each option's payoff simulated to get a sense of the true values.

7.1 Valuation under EWMA Model

The volatility of each underlying currency on a particular date was calculated by appropriately implementing the EWMA model as described in the volatility section. The weights applied on the observations along the time series were compounded by a lambda factor of 0.84. The seed period of moving averages was taken to be 40% of the times series length, this procedure was done on all of the four currencies. By moving an average to the latest date of valuation, a vector set of four daily volatilities of currency prices was then obtained and used as input in the valuation.

UNIVERSITY of the WESTERN CAPE

7.1.1 Tables of Result

Table 7.1 Valuation via EWMA model

Currency	EWMA Vol	Option Premium	Error Estimate
ZAR	0.010363%	306.034	0.008747%
GBP	0.001018%	33.4528	0.002892%
JPY	0.017982%	4019.06	0.031698%
EUR	0.002762%	33.3328	0.002887%
Basket			
Payoff	Premium		
1965.74	1097.97		
		Backtesting	
Option	**Valuated**	**Market valuation**	**Comparability**
ZAR	306.034	284.437	0.92943
GBP	33.453	33.520	1.00200
JPY	4019.060	23.545	0.00586
EUR	33.333	31.810	0.95430

https://etd.uwc.ac.za/

7.1.2 Analysis

Table 7.1 shows the results of valuating the European call option, on each of the underlying currencies and on the basket under the EWMA volatility model. The values are in USD, hence a long position in one million GBP sterling that matures in 3 months worth 33.45 USD on the 31 June 2008, at this date, the exchange rate is 7.82 ZAR/USD in ZAR terms. The option is therefore $33.45 \times 7.82 \Rightarrow R261.59$ million the basket premium of purchasing the four currencies is $1097.97USD \times 7.8199 \Rightarrow R8.59$ billion.

7.1.3 Conclusion

Since there was no volatility forecast conducted, the model assumed that the volatility would be 0.010363% for the next 3 months. This is in line with Bloomberg's calculations which imposed a similar assumption. It is well known, however, that the foreign exchange markets have been very volatile during the three months period with ZAR/USD volatility trending at 0.034% on 31 September 2008. It follows that the calculated values including the basket are not reflecting the reality. This is despite the fact that the results are strongly favoured by backtesting against the market, and as such the market and this model similarly made incorrect assumptions about the volatility.

7.2 Valuation under GARCH (1, 1) Model

Currency price volatilities are calculated by applying the mean reversion stochastic model given by equation (10) on the returns. By optimizing the probability function of observing prices subjected under the model and constrained under the unity condition, the model parameters that best explain the observations are obtained. With these parameters, namely; the weights on the previous days' volatility, the long run variance and current level of volatility, future levels are forecasted by appropriately applying equation (12) and then used as input into the valuation model.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

7.2.1 Results

The illustration shows the volatility versus time plot of each currency over a period of 400 days. The straight lines on the graphs correspond to long run mean variances.

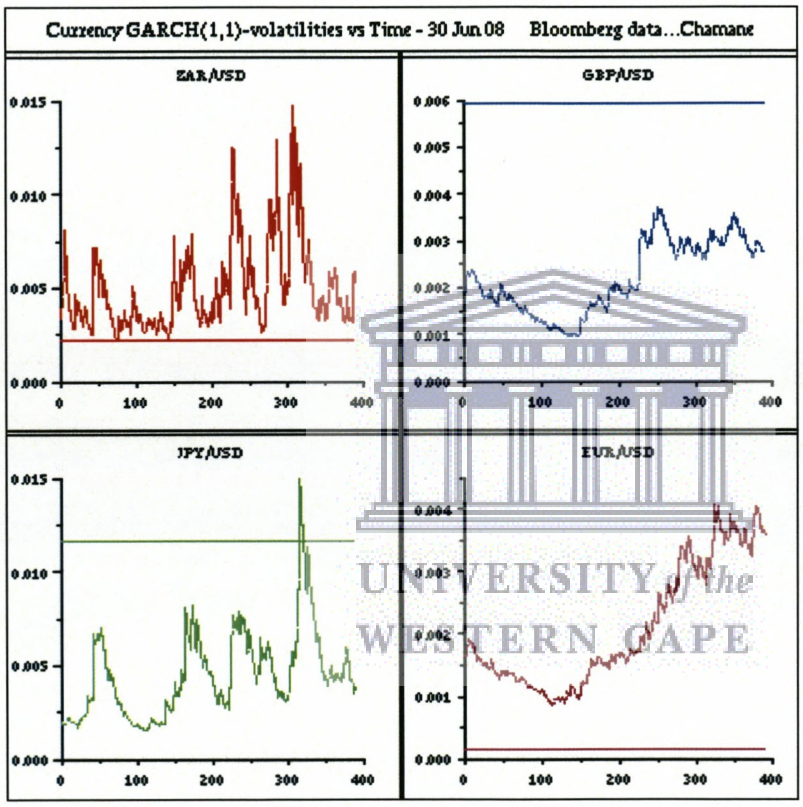

Figure 7.2.1 GARCH (1, 1) volatilities

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

Presented below are the results of volatility valuation and the portfolio. The option premiums are valuated based on a single currency and on the basket.

Table 7.2.1 **Valuation via GARCH (1, 1) model**

Cur	β	α	γ	$\beta+\alpha+\gamma$
ZAR	9.124273E-01	8.829156E-02	1.518556E-02	1.02
GBP	9.150598E-01	4.168596E-02	1.828917E-02	0.98
JPY	9.126513E-01	5.854001E-02	1.051310E-02	0.98
EUR	9.534065E-01	5.101019E-02	8.955590E-03	1.01
	Long Run Variance	**GARCH Forecast**	**Option Premium**	**Error Estimate**
ZAR	0.002264%	0.005343%	398.12	0.009976%
GBP	0.005963%	0.005922%	40.05	0.003164%
JPY	0.011595%	0.010666%	4708.14	0.034308%
EUR	0.000153%	0.006904%	36.71	0.003029%
		Basket		
	Payoff	**Premium**		
	2319.84	1295.00		
		Backtesting		
	Valuated	**Market valuation**	**Comparability**	
ZAR	398.12	284.44	0.71	
GBP	40.05	33.52	0.84	
JPY	4708.14	23.55	0.01	
EUR	36.71	31.81	0.87	

UNIVERSITY of the
WESTERN CAPE

7.2.2 Analysis

a) Volatility graphs

Figure 7.2.1 shows GARCH volatility time graphs of each currency plotted on the same set of axis as its long run mean variance component.

For the ZAR/USD volatility trend, the upper bound volatility is at 0.015%. The lower bound being its mean variance level is at 0.0025% which also serves as a support line resistance for the trend. Therefore it is sensible that a three months volatility forecast is 0.0053%(from table 7.2.1) in between the bound limits as after sufficient time the volatility will revert back to the long run variance level. For GBP/USD volatility, the trend starts at 0.0025% below its long run mean variance at 0.005963%. The forecasted three months volatility is 0.005922%

https://etd.uwc.ac.za/

which is closer to the mean. This means that it would take only three months for the volatility to move from its lowest point to the highest stable value and this indicates the magnitude of risk momentum.

The volatility of JPY/USD also starts at 0.0024% below the line of long run mean variance which is at 0.01159%. However, the forecast volatility is 0.01066% which is in agreement with the mean figure.

The EUR/USD volatility behaves similarly as the ZAR/USD volatility trending above the long run mean-variance line at 0.00153%. The forecast is 0.0069% which is also a huge momentum for a three month period.

b) Table of results

Under optimization the condition that prevailed is that weights had to sum to unity on each currency. We analyze the closeness of these sums to unity in order to discern the integrity of the results.

UNIVERSITY of the WESTERN CAPE

The sums are 1.02, 0.98, 0.98 and 1.01 for ZAR, GBP, JPY and EUR respectively. Since these ratios are satisfactorily close to 1.0 it can be concluded that we did not compromise the constrain condition and therefore the results are valid.

The option of purchasing one million units of ZAR/USD currency at a strike price of R8.01 each in three months time from 31 June 2008, costs R398.12 million. The valuation is done with 99.01% precision. Without volatility forecast the market valued this option at R284.44 million which traced 71% of the valuation by backtesting.

An option to buy one million units of GBP/USD currency at a strike price of $£1.97 \div (£1.99 \times R7.82) \Rightarrow R7.74$ each, costs $£33.52 \div £1.99 \times R7.82 \Rightarrow R131.72$

https://etd.uwc.ac.za/

millions. The valuation is done with 99.69% precision of simulation and is 84% comparable with the market valuation by backtesting.

To purchase one million units of EUR/USD currency at a strike price of €1.568 $\div$€1.576$\times$R7.82$\Rightarrow R7.86$ each, costs €36.71 $\div$€1.576$\times$R7.82$\Rightarrow R183.08$ under 99.67% precision. The option is 87% comparable with the market valuation.

An interesting valuation is that of JPY/USD currency. The option costs ¥4708.14 $\div$¥106.22$\times$R7.82$\Rightarrow R346.61$ million at a strike price of ¥105.70 $\div$¥106.22$\times$R7.82$\Rightarrow R7.78$ for each currency. However in the market the option cost R1.74 million, leading to the comparability of 1% in backtesting. This implication will be investigated further as it cannot be understood at this time.

The premium of the option basket written under the four currencies equally weighted for a notional amount of one million each costs USD1295.00$\times$7.82$\Rightarrow R10.13$ billion.

UNIVERSITY of the
WESTERN CAPE

7.2.3 Conclusion

Using GARCH volatility modeling a European call option written on a basket of four currencies is successfully modeled. Since such a complex derivative is not readily available in the market, the valuation was first conducted on option for a single reference asset for which a comprehensive backtesting was conducted. The valuation is then validated against market valuation. This saw the results strongly comparable with the market between 71% and 84% and suggests that if the markets were evaluating the options with some advanced stochastic volatility model, it would likely compare 100% with our valuation. We conclude that the valuation is a success and are confident that the results including the basket premium are sensible.

https://etd.uwc.ac.za/

7.3 Valuation under Hull-White Model

A Hull–White 1988 (HW88) model which takes into account the correlation structure of the underlying currency and its volatility movements was calibrated for four parameters; the correlation coefficient, volatility of volatility, the long run mean variance and its weight. The first two parameters have already been calibrated in the SABR model. The last two are calibrated in GARCH (1, 1) model. Listed below is the result of option valuation under HW88 model.

Table 7.3 Valuation via HW88 model

Currency	Hull-White88	Option Premium	Error Estimate
ZAR	0.037625%	232.15	0.007618%
GBP	0.038905%	15.52	0.001970%
JPY	1.468080%	614.66	0.012396%
EUR	0.027615%	27.87	0.002639%
Basket			
Payoff	Value		
398.44	222.55		
Backtesting			
	Valuated	**Market valuation**	**Comparability**
ZAR	232.15	284.44	0.82
GBP	15.52	33.52	0.46
JPY	614.66	23.55	26.11
EUR	27.87	31.81	0.88

UNIVERSITY of the WESTERN CAPE

7.3.1 Analysis

The ZAR, GBP and EUR currency options valuation yield meaningful results when backtested against market premiums and compared with the previous valuations. However, for some factors yet to be investigated further the JPY option premium contrasts significantly and is 26 times higher than the average market premium.

7.3.2 Conclusion

The valuation has been successfully carried out, with the basket valued at USD 222 million. However, a single valuation on JPY option premium yields different result which comes with high volatility and a low degree of accuracy. The cause of this outlier

https://etd.uwc.ac.za/

is not yet immediately known. The fact that a Tailor series expansion in which volatility parameters are estimated in the order terms is used in HW88 is a prime suspect to causing the option premium to be highly sensitive to the valuation. Given that JPY currency has the highest volatility level we can infer that this was the case in the valuation.

7.4 Valuation under SABR Model

The volatility model parameters namely; volatility–of–volatility, correlation, at–the–money volatility and the volatility of forward prices (alpha), are calibrated from the historical data series of currency prices. Various levels of the parameters are plotted on a three dimensional surface to analyze the domain under which the model is stable and therefore the acceptable range of alpha. The calibrated set is then checked whether it is within the acceptable range. The SABR volatilities are calculated for the date of valuation and applied in the simulation of the option prices.

UNIVERSITY of the
WESTERN CAPE

7.4.1 Results

Presented below are the results of volatility calculations using SABR model and option valuations implementing the model.

https://etd.uwc.ac.za/

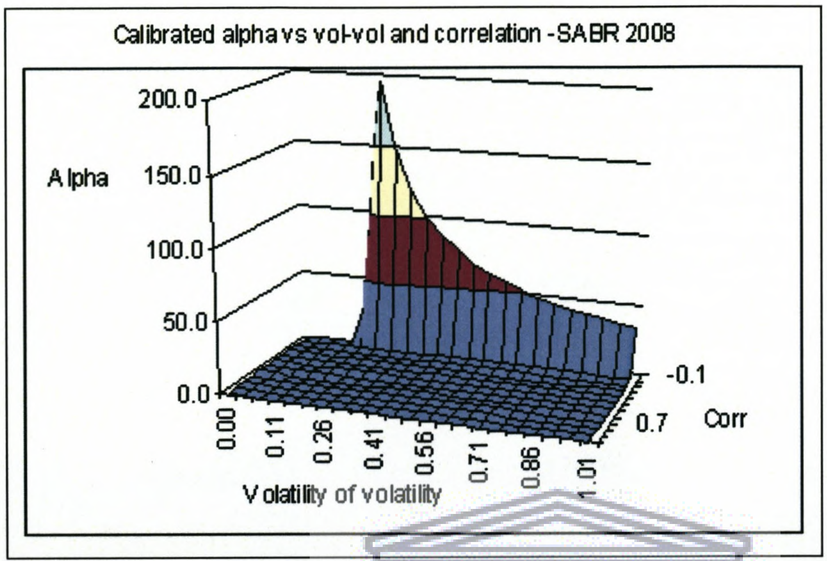

Figure 7.4.1 SABR surface parameters

Table 7.4.1 shows the result of valuation via SABR, the first section displays the input parameters from calibration, the second section shows the resultant volatilities, and the last section displays the portfolio valuation results.

UNIVERSITY of the
WESTERN CAPE

Table 7.4.1 Valuation via SABR model

	ZAR	GBP	JPY	EUR
		Calibration		
Vol-vol	0.7761%	0.4726%	0.3272%	0.4119%
Atm-vol	0.0048%	0.0013%	0.0020%	0.0018%
Correlation	0.001044	0.000042	0.000074	0.000025
Beta	0.999	0.999	0.999	0.999
Alpha	1.00206	1.00069	1.00468	1.00045
		Valuation		
Volatility	3.8372%	3.8373%	3.8373%	3.8373%
Error estimate	0.0139%	0.0050%	0.0368%	0.0034%
Premium	769033.10	100441.00	5425320.00	45092.60
Basket		Payoff	Valuation	
		2837.63	1584.97	
		Backtesting		
Valuated	769033.10	100441.00	5425320.00	45092.60
Market	284437.22	33519.65	23545.30	31809.50
Comparability	36.99%	33.37%	0.43%	70.54%

https://etd.uwc.ac.za/

7.4.2 Analysis

a) Graph

Figure 7.4.1 shows a plot of the alpha against the surface of correlations and volatility of volatility. Above low correlation and low volatility of volatility coordinates alpha is unstable and rises to infinite. Above low correlation and increasing volatility of volatility alpha decreases but is still unstable. When the correlation increases away from -1.0 and closer to 1.0 the alpha assumes definite values and the model approaches stability.

b) Table of results

The table 7.4.1 is divided into three sections the calibration, valuation and the backtesting section. The row of at-the-money volatilities is calculated internally as a result of readjustment in the calibration of other parameters. The volatility of volatility and the correlation coordinates show high values that are well above negative. This is to ensure that the right parameters to keep the model stable are selected. The model stability can be noticed from the result of calibrated alphas which range from 1.0026 to 1.0045 which are stable and definite. The premiums of individual currency options are;

$$\text{ZAR option}\, 769033.17 \times 7.82 \Rightarrow R601.3\,\text{million}$$

$$\text{GBP option}\, 1004.41 \div 1.992 \times 7.82 \Rightarrow R39.47\,\text{million}$$

$$\text{JPY option}\, 542532.0 \div 106.22 \times 7.82 \Rightarrow R39.94\,\text{million}$$

$$\text{EUR option}\, 76903.317 \div 1.576 \times 7.82 \Rightarrow R22.37\,\text{million}$$

$$\text{Basket option}\, 12394.461 \times 7.82 \Rightarrow R12.39\,\text{billion}$$

https://etd.uwc.ac.za/

7.4.3 Conclusion

The basket of currency option is successfully valuated under SABR option pricing volatility model. The success behind the evaluation lies in the fact that the parameters of volatility model were carefully selected for calibration as the model easily becomes unstable under a certain range of low correlation and volatility of volatility domain. The results are checked against the market valuation of each currency option and compared with the results of valuation under EWMA, GARCH and Hull-White models, we find that there is no significant difference across the methods and conclude that SABR model was stable through the valuation under a careful consideration of the range of acceptable parameters.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

8 Relative Performance Analysis

This chapter presents the performance analysis of the option on the currency basket relative to all the models of stochastic volatilities discussed previously. It discusses the integration of all currency risks under a particular volatility model into a single risk of the basket and examines the effect of such a risk on the overall option's result.

The value of integral volatility of the basket can be found if the correlation matrix of the individual currencies, their respective volatilities and their proportional weights in the basket are known. Refer to the treatment of multi–assets option discussed in section 3 where the portfolio volatility is integrated from individual volatilities of underlying assets (Wilmott, 1998).

Let an m–dimensional correlation matrix of currencies be ρ_{mm} and their volatility vector $\bar{v}_X$ computed under a particular volatility model X. The total volatility V_B of the basket can be expressed as follows,

UNIVERSITY of the WESTERN CAPE

$$V_B = \bar{v}_X' (\rho_{mm}) \bar{v}_X \times \frac{1}{m^2} \tag{23}$$

The correlation matrix of the currencies is calculated using a variance–covariance updating scheme (EWMA) model on the historical prices which date from 31-December-1997 to 31-June-2008. The results are shown in Table 8.a

Table 8.a Correlation matrix

Curr	ZAR	GBP	JPY	EUR
ZAR	1.00	-0.26	0.06	-0.29
GBP	-0.26	1.00	-0.31	0.67
JPY	0.06	-0.31	1.00	-0.36
EUR	-0.29	0.67	-0.36	1.00

https://etd.uwc.ac.za/

The diversification level of the basket can be understood by observing the matrix elements. The GBP and EUR are negatively correlated with respect to the JPY and with respect to the ZAR but strongly correlated with respect to each other. The JPY is least correlated with the ZAR.

This means that an adverse movement in any of the western currency markets will not have a severe impact on the overall basket and that the basket is therefore well diversified.

A summary of results of currency volatilities for the 31-June-2008 which are calculated by implementing all the stochastic volatility models discussed previously is presented in Table8.b.

Table 8.b Stochastic volatility models

Curr	EWMA volatility	GARCH volatility	HW88 volatility	SABR volatility
ZAR	1.017988%	0.730958%	1.939716%	1.958884%
GBP	0.319061%	0.769545%	1.972435%	1.958911%
JPY	1.340970%	1.032763%	12.116435%	1.958909%
EUR	0.525547%	0.830903%	1.661776%	1.958909%

The volatility of the portfolio is calculated by integrating the volatilities in each column of Table 10b using equation (23). The premium of portfolio is then valuated under single portfolio volatility. Table 8.c shows the relative performance of the portfolio under various volatility models.

Table 8.c Portfolio's relative performance

Model	Volatility	Premium
EWMA	0.006%	R 8.59 billion
GARCH	0.005%	R 10.13 billion
HW88	0.328%	R 1.74 billion
SABR	0.029%	R 12.39 billion

The portfolio is composed of ZAR, GBP, EUR and JPY currencies, each contributes 1 million shares. The total number of holdings in the portfolio is 4 million shares.

https://etd.uwc.ac.za/

The GARCH model yields the lowest portfolio volatility 0.005% with the premium valued at R10.13 billion. The second least volatility result is 0.006% calculated under EWMA model; the corresponding premium is valued at R8.59 billion. The highest premium R12.39 billion is obtained under SABR model which has the second highest volatility 0.029%.

The highest volatility is obtained under HW88 model 0.328%. However the premium valuated under this model is far low at R1.74 billion which highly deviates from the average. We treat this result as an outlier. The degree of sensitivity of premium under the model parameters used in the derivation of HW88 volatility discussed previously can be seen from the result.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

9 Further Conclusion

An outlier in the calculation of option premium under HW88 volatility model was expected because when the premiums of individual currencies were calculated under this model the results were largely distorted by the values of volatility. The volatility of JPY in particular severely affected the premium when verified against the market premium obtained from Bloomberg data services. The model failed the backtesting against the market valuation under single currency. When the principle behind the construction of the model was revisited we noted that the choice of estimating the model parameters from Tailor Series expansion determined the premium result. It is for this reason that HW88 model is seen as useful for experimentation and is elementary for learning the concepts behind stochastic volatility models, however it not recommend for commercial practice.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

10 Bibliography

Aforbes Digital Company, *Option Writer*, Investopedia, http://www.investopedia.com/terms/w/writer.asp; Accessed 2pm, 22-Aug-2008.

Bollerslev, T., 1986, *Generalized Autoregressive Conditional Heteroskedasticity*, Journal of Econometrics, 31, 307–327.

Derman, E., Taleb, N., 2005, *The Illusion of Dynamic Delta Replication*, Quantitative Finance, 5(4), 323–326.

Dragulescu, A.A., Yakovenko, V.M., 2002, *Probability Distribution of Returns in the Heston Model with Stochastic Volatility*, Journal of Finance, 2, 443–453.

Dorfleitner, G., Schneider, P., Hawlitschek, K., Buch, A., 2008, *Pricing Options with Green's Function when Volatility, Interest Rate and Barrier Depend on Time*, Quantitative Finance, 8, 119–133.

Edward, J.S., Timothy, M.W., 1991, *Louis, The Father of Modern Option Pricing Theory,* Journal of Economic Education, 22(2), 165–171.

Engle, R.F., Mezrich, J., 1995, *Grappling with GARCH*, September Risk Magazine.

Finnerty, J.D.,2005,*Extending Black-Scholes-Merton Model To Value Employee Stock Option*, January Finnerty Economic Consulting.

Glasserman, P., 2003, *Monte Carlo Methods in Financial Engineering*, Springer–Verlag, New York.

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

Haug, E.G., 2007, *Option Pricing Formulas, Second Edition*, McGraw–Hill, New York.

Hull, J.C., 2006, *Options Futures and Other Derivatives*, Sixth Edition, Prentice Hall, New Jersey.

Hagan, P.S., Kumar, D., Lesniewski, A.S., Woodward, D.E., 2002, *Managing Smile Risk*, Wilmott Magazine, New York.

Hoffman, P., 1998, *The Man Who Loved Only Numbers: The Story of Paul Erdos and Search for Mathematical Truth*, Hyperion, New York.

Hull, J., White, A., 1987, *The Pricing of Options on Assets with Stochastic Volatilities*, Journal of Finance, 2, 281–300.

Hammersley, J.M., 1960, *Monte Carlo Methods for Solving Multivariable Problems*, Ann. Acad Sci, 86, 844–874, New York.

Hang Chan,N., Wong, H.W., 2006, Simulation Techniques in Financial Risk Management, Wiley Intelligence,31–45, New Jersy.

Jondeau, E., Ser-Huang, P., Rockinger, M., 2007, *Business and Economics*, 541

Kabanov, Y., Bru, B., Courtault, J.M., Crepel, P., Lebon, I., Marchand A.L., 2000, *On The Centary of Theorie De La Speculation*, Mathematical Finance,10(3), 341–353.

Lewis, A.L., 2000, *Option Valuation under Stochastic Volatility*, Finance Press, California.

Merton, R.C., Simon, R.V., Wilkie, A.D., 1994, *Infuence of Mathematical Models in Finance on Practise* Mathematical Models in Finance, 347(1684), 451–463

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

Malliaris, M., 1993, *A nearal Network Model for Estimating Option Prices*, Journal of Applied Intelligence, 3,193–206

Seymour, D.W., Brown, T., Suglia, J., 2008, *Beyond the Credit Crisis,* KPMG FRM, Global Survey.

Stroustrup, B., 1997*, The C++ Programming Language*, Third Edition, A.W Longman, New York.

Smith, Jr., C.W, 1976, *Option Pricing: A Review, Journal of Financial Economics*, 3, 3–51.

Samuelson, P., 1965, *Rational Theory of Warrant Pricing, The Industrial Management* Review, 6, 13–31.

Sprenkle, C., 1964, *Warrant Prices as Indicators of Expectations and Preferences*, Cootner, P, MIT Press, Cambridge.

Silverman, D., 1999, *Solution of the Black Scholes Equation Using the Green's Function of the Diffusion Equation*, Department of Physics and Astronomy,Irvin, CA 92697-4575, University of California

West, G., 2005, *Calibration of the SABR Model in Illiquid Markets*, Applied Mathematical Finance, 12(4), 375–385.

Wilmott, P., 1998, Derivatives*: The Theory & Practice of Financial Engineering*, John Wiley & Sons.

Weisstein, E.W., *Monte Carlo Method*, MathWorld--A Wolfram, http://mathworld.wolfram.com/MonteCarloMethod.html. 3pm, 22-Aug-2008.

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

Yakovenko, V.M., Dragulescu, A.A., 2002, *Probability Distribution of Returns in the Heston Model with Stochastic Volatility*, Quantitative Finance, 2(2002), 443-453.

UNIVERSITY *of the*
WESTERN CAPE

https://etd.uwc.ac.za/

11 Appendix A

11.1 Historic data input

Table11.1 shows a sample data of daily price movement of the four main currencies with respect to the USD dollar currency sourced from Bloomberg data services.

Table 11.1 Currency price movement against USD

Date	ZAR	GBP	JPY	EUR		ZAR	GBP	JPY	EUR
30/06/2008	7.819	1.992	106.210	1.576	06/05/2008	7.565	1.972	104.850	1.550
27/06/2008	7.915	1.995	106.130	1.579	05/05/2008	7.583	1.972	105.400	1.542
26/06/2008	7.958	1.989	106.810	1.576	02/05/2008	7.609	1.975	104.430	1.547
25/06/2008	7.855	1.975	107.800	1.567	01/05/2008	7.562	1.987	103.910	1.562
24/06/2008	8.024	1.971	107.820	1.557	30/04/2008	7.610	1.970	104.020	1.557
23/06/2008	8.055	1.965	107.850	1.552	29/04/2008	7.528	1.991	104.190	1.566
20/06/2008	7.991	1.976	107.330	1.561	28/04/2008	7.617	1.986	104.420	1.563
19/06/2008	7.946	1.972	108.000	1.550	25/04/2008	7.705	1.974	104.260	1.568
18/06/2008	8.012	1.960	107.880	1.554	24/04/2008	7.676	1.980	103.380	1.589
17/06/2008	8.030	1.957	107.930	1.551	23/04/2008	7.650	1.995	103.020	1.599
16/06/2008	8.085	1.963	108.220	1.548	22/04/2008	7.771	1.980	103.270	1.591
13/06/2008	8.106	1.948	108.190	1.538	21/04/2008	7.785	1.998	103.670	1.582
12/06/2008	8.148	1.946	107.960	1.544	18/04/2008	7.800	1.991	102.480	1.591
11/06/2008	8.015	1.963	106.960	1.555	17/04/2008	7.862	1.972	101.830	1.595
10/06/2008	7.985	1.954	107.440	1.547	16/04/2008	7.939	1.963	101.830	1.579
09/06/2008	7.901	1.975	106.310	1.565	15/04/2008	7.879	1.978	101.100	1.583
06/06/2008	7.876	1.971	104.930	1.578	14/04/2008	7.821	1.969	100.950	1.581
05/06/2008	7.834	1.958	105.940	1.559	11/04/2008	7.855	1.971	101.950	1.574
04/06/2008	7.791	1.956	105.230	1.544	10/04/2008	7.845	1.976	101.800	1.583

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

03/06/2008	7.754	1.963	105.080	1.545	09/04/2008	7.778	1.970	102.660	1.571
02/06/2008	7.708	1.967	104.430	1.554	08/04/2008	7.800	1.988	102.400	1.571
30/05/2008	7.610	1.982	105.520	1.555	07/04/2008	7.875	1.993	101.470	1.574
29/05/2008	7.584	1.977	105.500	1.552	04/04/2008	7.776	1.997	102.250	1.568
28/05/2008	7.674	1.981	104.690	1.564	03/04/2008	7.823	1.987	102.360	1.569
27/05/2008	7.732	1.976	104.240	1.569	02/04/2008	7.984	1.976	101.850	1.561
26/05/2008	7.696	1.982	103.430	1.577	01/04/2008	8.091	1.984	99.690	1.579
23/05/2008	7.681	1.980	103.380	1.576	31/03/2008	8.143	1.994	99.230	1.580
22/05/2008	7.636	1.980	104.070	1.573	28/03/2008	8.008	2.007	99.650	1.578
21/05/2008	7.735	1.973	103.050	1.580	27/03/2008	8.057	2.009	99.200	1.585
20/05/2008	7.663	1.968	103.680	1.565	26/03/2008	8.050	2.006	99.980	1.565
19/05/2008	7.534	1.949	104.330	1.551	25/03/2008	8.135	1.986	100.740	1.542
16/05/2008	7.474	1.957	104.040	1.558	24/03/2008	8.173	1.982	99.580	1.543
15/05/2008	7.547	1.947	104.740	1.545	21/03/2008	8.157	1.985	99.510	1.543
14/05/2008	7.651	1.946	105.040	1.547	20/03/2008	8.086	1.984	99.030	1.563
13/05/2008	7.579	1.946	104.750	1.547	19/03/2008	7.933	2.006	99.850	1.563
12/05/2008	7.614	1.958	103.750	1.555	18/03/2008	8.118	1.999	97.330	1.573
09/05/2008	7.735	1.954	102.870	1.548	17/03/2008	7.934	2.020	99.090	1.567
08/05/2008	7.598	1.954	103.740	1.539	14/03/2008	7.903	2.034	100.650	1.564
07/05/2008	7.574	1.954	104.730	1.539	13/03/2008	7.873	2.027	101.790	1.555

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

12 Appendix B

12.1 Model source code

Functions

Header file 12.1a

```
// FILE NAME: FUNCTIONS.h
/* WRITTEN BY: Africa M Chamane
   FOR: Computational Finance course - Msc Thesis
   DATE: Nov 2008
   PRUPOSE: .Create utility functions to register marketdata file for
the model
              .The functions calculates dynamic risk matrix based on
the input marketdata file
*/
#ifndef FUNCTIONS_H
#define FUNCTIONS_H

// link inbuilt libraries
#include <vector>
using std::vector;

/* Defines market data series upload functions....targeting ewma
correlation matrix...
   all arguments are output.
*/
void ewmaRisks(vector<vector<double> >& returnVec //Return vector
                 ,int& tLine
                 ,int& nObjects
                 ,vector<vector<double> >& ewmaVarVec
                 ,vector<vector<vector<double> > >& ewmaCovarMatrix
                 ,vector<vector<vector<double> > >& ewmaCorrMatrix

               );
/* Factorizes correltion matrix into Cholesky factor matrix
   invokes correlation matrix arguments from function...output
    Cholesky factor matrix as argument from choleskyMatrix.
*/
void cholesky(int& tLine,int& nObjects,vector<vector<vector<double> >
>& ewmaCorrMatrix
                       ,vector<vector<vector<double> > >&
choleskyMatrix
                       );
void normalRV(int& tLine,vector<vector<double> >&
maturity,vector<double>& mullerBox,vector<vector<double> >&
mullerBoxVec);
void uncorBrownM(int& tLine,int& nObjects,vector<vector<double> >&
maturity,vector<double>& mullerBox,vector<vector<double> >& wiener);
void corrBrownM(int& tLine,int& nObjects,vector<vector<double> >&
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
maturity,vector<vector<vector<double> > >& choleskyMatrix
                   ,vector<vector<double> >&
wiener,vector<vector<double> >& corBrownVec);
void readData(const int tLine,vector<double>& mullerBoxVec);

void nAssets(int& nObjects,int& tSeries
              ,vector<vector<double> >& spotPriceVec      // Spot
price vector
              ,vector<vector<double> >& maturity          // Maturity
dates vector
              ,vector<vector<double> >& interestRate      // Interest
rate -Repo rates vector
              ,vector<vector<double> >& strikePriceVec // Strike price
              ,vector<double>& zarDF                    // ZAR discount
factor
              ,vector<vector<double> >& corBrownVec  // correlated
Brownian vector
              ,vector<vector<double> >& wh88        //ewmaVarVec
//garchForecast  // sabrVol/hw88
              ,vector<vector<double> >& assetPrice // Asset(t)
              );


#endif // FUNCTIONS_H
```

Source code 12.1b

```
// FILE: MARKETDATA.cpp
// {L}
// Source file

#include "MARKETDATA.h"
// link inbuilt libraries
#include<iostream>
#include <cmath>                   // for trigonometry functions
#include <string>
#include <fstream>
#include <vector>
#include <iomanip>
#include <stdlib.h>
#include <math.h>
using std::vector;
#include <cstdlib>  // For srand() and rand()
#include <ctime>    // For time()
using namespace std;

// Method to create normal random variables
void normalRV(int& tLine,vector<vector<double> >&
maturity,vector<double>& mullerBox,vector<vector<double> >&
mullerBoxVec){

                        //  initialized at each time method is called.
     // Calls random variates 1 and 10.
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
     double r1, r2;
     // Calculates random variates between 0 and 1.
     double m1,m2;
     double f=0.0;
     //double m2=r2/10;
     const double pi=22.0/7.0;
     // Initialize Muller Box RV's
     int mpaths=static_cast<int>(maturity[1][2]);
     mullerBoxVec=vector<vector<double> >(2,vector<double>(mpaths));
     mullerBox=vector<double>(2);
     mullerBox[0]=0;
     mullerBox[1]=0;
     srand(time(0)); // point of random numbers for each sequence is
     for(int t=0;t < mpaths; ++t){
           // Initialize random number generator-A starting
          r1 = (rand() % 10)+1;
          r2 = (rand() % 10)+1;
          m1=r1/10;
          m2=r2/10;
          mullerBox[0]=sqrt(-2.0*log(m1))*cos(2.0*pi*m2); //Horizontal
component random vector
          mullerBox[1]=sqrt(-2.0*log(m1))*sin(2.0*pi*m2); //Vertical
component random

          mullerBoxVec[0][t]=mullerBox[0];
          mullerBoxVec[1][t]=mullerBox[0];
     }// end testing
}//End of Muller Box  normal random generator

//Method to create uncorrelated Borwnian Motion
void uncorBrownM(int& tLine,int& nObjects,vector<vector<double> >&
maturity,vector<double>& mullerBox,vector<vector<double> >& wiener){

     // Declaring arguments for normalRV function
     double dt=1.0; // time step
     vector<vector<double> > mullerBoxVec;
     vector<double> w(2);

     int mpaths=static_cast<int>(maturity[1][2]);
     wiener=vector<vector<double> >(nObjects,vector<double>(mpaths));

     for(int n=0;n < nObjects;++n){
          normalRV(tLine,maturity,mullerBox // only these are useful
here
                         ,mullerBoxVec // this is not used here
                         );
          for(int t=1;t < mpaths;++t){
               w[0]=mullerBoxVec[0][t];
               w[1]=w[0] + mullerBoxVec[1][t]*sqrt(dt);
               wiener[n][t]=w[1];
          }
     }
}//End of Uncorrelated Brownian Motion vector method
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
//Method to generate Cholesky Factorization matrix
void cholesky(int& tLine,int& nObjects,vector<vector<vector<double> >
>& ewmaCorrMatrix
                        ,vector<vector<vector<double> > >&
choleskyMatrix
                        ){
     double seed=0.4; //The series moving average to be used in the
calculation
     int seedLine=int(tLine*seed);
     double sum; // For summing square mstrix elements
     choleskyMatrix= vector<vector<vector<double> >
>((tLine),vector<vector<double> >(nObjects,vector<double>(nObjects)));
     for(int t=0; t < (tLine-seedLine); t++){
          // Factorizing correlation matrix intor Cholesky factors

          for(int x=0;x <nObjects; ++x){
               sum=0.0;
               for(int y=0;y <=x-1; ++y){

                    sum=sum +
ewmaCorrMatrix[t][x][y]*ewmaCorrMatrix[t][x][y];

               }// looping through inner dimension
               choleskyMatrix[t][x][x]=ewmaCorrMatrix[t][x][x]-sum;
               //cout << "\n ....choleskyMatrix[t][x][x]:"
<<choleskyMatrix[t][x][x];...ok
               if(choleskyMatrix[t][x][x]<=0.0){
                    break;
               }//End if
               else

     {choleskyMatrix[t][x][x]=sqrt(choleskyMatrix[t][x][x]);}
               for(int k=x;k < nObjects;++k){
                    sum=0.0;
                    for(int y=0;y <=x-1; ++y){

                         sum=sum +
ewmaCorrMatrix[t][k][y]*ewmaCorrMatrix[t][x][y];

                    }// looping through inner dimension (y)
                    choleskyMatrix[t][x][k]=(ewmaCorrMatrix[t][x][k]-
sum)/choleskyMatrix[t][x][x];
               }//looping through outer dimension (k)
          }// looping through outer dimension (x)
     }// looping through time dimension
} // End of choleskyMatrix method

// Method to compute EWMA model
void ewmaRisks(vector<vector<double> >& returnVec //Return vector
                  ,int& tLine
                  ,int& nObjects
                  ,vector<vector<double> >& ewmaVarVec
                  ,vector<vector<vector<double> > >& ewmaCovarMatrix
                  ,vector<vector<vector<double> > >& ewmaCorrMatrix
```

UNIVERSITY of the
WESTERN CAPE
https://etd.uwc.ac.za/

```
                  ){
     using std::cout;
     using std::cin;

     double seed=0.4;   // Period ratio of desired moving averages
     const double lamda=0.84; // Ewma weight coefficient
      //Call data series file to activate arguments.
     int seedLine=int(tLine*seed);         // Period of moving
averages
     //Defining seed moving average volatility
     vector<double> rMeanVec(nObjects);// returns mean
     vector<double> varVec(nObjects); // volatiltity of returns
     vector<vector<double> >
covarVec(nObjects,vector<double>(nObjects));
     //vector<vector<vector<double> > >
wmaCovarMatrix(vector<double>(nObjects),vector<double>(nObjects),vecto
r<double>(nObjects));
     //Declare ewma variance vector, ewmacovar matrix and ewma
correlation matrix

     ewmaVarVec=vector<vector<double> >(nObjects,vector<double>(tLine-
seedLine));
     ewmaCovarMatrix= vector<vector<vector<double> >
>((tLine),vector<vector<double> >(nObjects,vector<double>(nObjects)));
     ewmaCorrMatrix= vector<vector<vector<double> >
>((tLine),vector<vector<double> >(nObjects,vector<double>(nObjects)));


     //...calculates mean return of seed
     for(int n=0;n < nObjects; ++n){
          //Initialization
          rMeanVec[n]=0.0;
          for(int t=1; t <= seedLine; t++){
               rMeanVec[n]=rMeanVec[n]+returnVec[n][t]/(seedLine-1);

          }
     }
     //...Computes average var vector on the seed.....imposes equal
weighting on returns deviation from the mean return.
     for(int n=0;n < nObjects; ++n){
          varVec[n]=0.0;//Initialization.....
          for(int t=1; t <= seedLine; t++){
               varVec[n]=varVec[n]+pow((returnVec[n][t]-
rMeanVec[n]),2.0)/(seedLine-1);
          }
     }
     //...Computes EWMA variance vector...Exponentially assigns more
weight to the more recent returns.
     for(int n=0;n < nObjects; ++n){
          ewmaVarVec[n][0]=varVec[n]; //Initialization.....

          for(int t=seedLine+1; t < (tLine); t++){
               ewmaVarVec[n][t-seedLine]=lamda*ewmaVarVec[n][t-
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
seedLine-1]+(1-lamda)*pow(returnVec[n][t-1],2.0);
          }
     }
     //...Computes average covariance matrix on the seed ...imposes
equal weighting on returns deviation from the mean return.
     for(int x=0;x < nObjects; ++x){
          for(int y=0;y < nObjects; ++y){
               //Initialization.....
               covarVec[x][y]=0.0;
               for(int t=1; t <= seedLine; t++){
                    covarVec[x][y]=covarVec[x][y]+(returnVec[x][t]-
rMeanVec[x])*(returnVec[y][t]-rMeanVec[y])/(seedLine-1);
               }// looping through returns
          }// looping through inner dimension
     }// looping through outer dimension

     //Computes EWMA covariance matrix...Exponentially assigns more
weight to the more recent returns.
     for(int x=0;x < nObjects; ++x){
          for(int y=0;y < nObjects; ++y){
               //Initialization.....
               ewmaCovarMatrix[0][x][y]=covarVec[x][y];
               for(int t=seedLine+1; t < (tLine); t++){
                    ewmaCovarMatrix[t-
seedLine][x][y]=lamda*ewmaCovarMatrix[t-seedLine-1][x][y]+(1-
lamda)*(returnVec[x][t-1]*returnVec[y][t-1]);
                    //cout<<"\n...ewmaCovarMatrix
:"<<ewmaCovarMatrix[t][x][y];
               }// looping through returns
          }// looping through inner dimension
     }// looping through outer dimension
     //Computes EWMA correlation matrix...Exponentially assigns more
weight to the more recent returns.
     for(int t=0; t < (tLine-seedLine); t++){
          for(int x=0;x < nObjects; ++x){
               for(int y=0;y < nObjects; ++y){


     ewmaCorrMatrix[t][x][y]=ewmaCovarMatrix[t][x][y]/sqrt(ewmaVarVec[x
][t]*ewmaVarVec[y][t]);
               }// looping through inner dimension
          }// looping through outer dimension
     }// looping through time dimension
}// End main method

// Method to generate correlated Brownian Motion
void corrBrownM(int& tLine,int& nObjects,vector<vector<double> >&
maturity,vector<vector<vector<double> > >& choleskyMatrix
                    ,vector<vector<double> >&
wiener,vector<vector<double> >& corBrownVec){

     int mpaths=static_cast<int>(maturity[1][2]);
     //Declare correlated Brownian vector
     corBrownVec=vector<vector<double>
```

UNIVERSITY of the
WESTERN CAPE
https://etd.uwc.ac.za/

```
>(nObjects,vector<double>(mpaths));
     for(int t=1; t < mpaths; t++){
          for(int m=0;m < nObjects; ++m){
               corBrownVec[m][t]=0.0;
               for(int n=0;n < nObjects; ++n){

                    corBrownVec[m][t]+=
choleskyMatrix[0.6*tLine][m][n]*wiener[n][t];
               }// looping through inner dimension

          }// looping through outer dimension
     }// looping through time dimension

}//End of correlated Brownian Motion vector

//Method of updating scheme of n-dimensional assets using Monte Carlo
Integration
void nAssets(int& nObjects,int& tSeries
               ,vector<vector<double> >& spotPriceVec      // Spot
price vector
               ,vector<vector<double> >& maturity          // Maturity
dates vector
               ,vector<vector<double> >& interestRate      // Interest
rate -Repo rates vector
               ,vector<vector<double> >& strikePriceVec // Strike price
               ,vector<double>& zarDF                    // ZAR discount
factor
               ,vector<vector<double> >& corBrownVec  // correlated
Brownian vector
               ,vector<vector<double> >& wh88        //ewmaVarVec
//garchForecast  // sabrVol/hw88
               ,vector<vector<double> >& assetPrice // Asset(t)
               ){

     int nsimulations;
     int count=0;
     int mpaths=static_cast<int>(maturity[1][2]); //Define Asset path
     nsimulations=40000;                                 //Arbitrary
chosen number of simulatiolns
     double dt=maturity[1][2]/365.0;              // Time step
     double basketPayoff=0.0;                    //Initilise basket
payoff
     double optionValue=0.0;
     vector<double> drift(nObjects);
     vector<double> vol(nObjects);
     vector<double> payoffSum(nObjects);
     vector<double> eachSharePayoff(nObjects);
     assetPrice=vector<vector<double>
>(nObjects,vector<double>(mpaths));

     // Error calcs arguments..
     vector<vector<double> >
payoff(nObjects,vector<double>(nsimulations));
     vector<double> payoffMean(nObjects);
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
     vector<double> erro(nObjects);

     for(int n=0; n < nObjects; ++n){
          //vol[n]=ewmaVarVec[n][int(0.6*tSeries)]; //Valuating via
EWMA vol model
          //vol[n]=garchForecast[n][0];         //Valuating via
GARCH(1,1) vol model
          //vol[n]=sabrVol[n][0];      //Valuating via SABR vol model
          vol[n]=wh88[n][0];      //Valuating via SABR vol model
          assetPrice[n][0]=spotPriceVec[n][0];      // Spot prices

          drift[0]=interestRate[0][3]; //Domestic..ZAR interest rate
          count+=1;

          if(count<=nObjects){drift[count]=interestRate[0][3]-
interestRate[count][3];} //domestic - foreign interest
          for (int s=0; s < nsimulations; ++s ){ //sampling
               payoffSum[n]=0.0;
               //Asset path
               for (int t=1; t < mpaths; ++t){
                    assetPrice[n][t]=assetPrice[n][t-1]*exp((drift[n]-
0.5*abs(vol[n]))*(dt)+sqrt(abs(vol[n]))*corBrownVec[n][t]*sqrt(dt));
               }//Path ends here
               // payoff of each asset
               payoffSum[n]+=max(assetPrice[n][mpaths-1]-
strikePriceVec[n][1],0.0);
               payoff[n][s]=max(assetPrice[n][mpaths-1]-
strikePriceVec[n][1],0.0); //records samples
          }// sampling function ends here
          payoffSum[n]=payoffSum[n]/ nsimulations;
     }// nObjects

     for(int n=0; n < nObjects; ++n){
          basketPayoff+=payoffSum[n]/nObjects;
          eachSharePayoff[n]=payoffSum[n];
          for (int t=0; t < mpaths;
++t){eachSharePayoff[n]=eachSharePayoff[n]*zarDF[t];
          }// Each share payoff discount
     }
     optionValue=basketPayoff;
     for (int t=0; t < mpaths; ++t){optionValue=optionValue*zarDF[t];}
//Basket payoff discount
     // Valuating Error
     for(int n=0; n < nObjects; ++n){ //For each object
          payoffMean[n]=payoffSum[n]; // payoffSum already an average
          erro[n]=0.0;
          for (int s=0; s < nsimulations; ++s ){
               erro[n]+=pow((payoff[n][s]-
payoffMean[n]),2.0)/nsimulations;
          }
          erro[n]=sqrt(erro[n]);
          for (int t=0; t < mpaths; ++t){erro[n]=erro[n]*zarDF[t];}

     }//End of all objects
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
    //.....Printing...
    cout <<"\n    ";
    cout <<" Option Value of each[n] :";
    cout <<" error  :";
    cout <<" Vol[n] :";
    for(int n=0;n < nObjects; ++n){
         cout <<"\n ";
         cout <<eachSharePayoff[n];
         cout <<"    :    ";
         cout <<sqrt(erro[n])/nsimulations;
         cout <<"    :    ";
         cout <<vol[n];
    }
    cout << endl << endl;
    cout << "\n  optionValue:" <<optionValue;
    cout << "\n  basketPayoff:" <<basketPayoff;
    cout << "\n  Path size:" <<mpaths;

}//End of Monte Carlo Method
```

12.2 Valuation model

```
#include"MARKETDATA.h"
#include"FUNCTIONS.h"
#include <math.h>
#include <cmath>
#include<iostream>
using namespace std;

int main(){

    //GARCH(1,1) Volatilities ---the function is not part of this
object, calculation done on  a C++ file
    vector<vector<double> > garchForecast(4,vector<double>(1));
    garchForecast[0][0]= 5.34292*pow(10.0,-5.0);
    garchForecast[1][0]= 5.92188*pow(10.0,-5.0);
    garchForecast[2][0]= 0.00010661;
    garchForecast[3][0]= 6.90425*pow(10.0,-5.0);

    //SABR model Volatilities ---the function is part of the source
code, calculation done on  a seperate  template and objets may be
linked
    vector<vector<double> > sabrVol(4,vector<double>(1));
    sabrVol[0][0]= 0.0383722552;
    sabrVol[1][0]= 0.0383777262;
    sabrVol[2][0]= 0.0383732486;
    sabrVol[3][0]= 0.0383732361;

    //HullWhite model Volatilities ---the function is part of the
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
source code, calculation done on  a seperate  template and objets may
be linked
     vector<vector<double> > wh88(4,vector<double>(1));
     wh88[0][0]= 0.000376250;
     wh88[1][0]= 0.000389053;
     wh88[2][0]= 0.014680800;
     wh88[3][0]= 0.000276154;
     //defines data parameter function input arguments

     //defines data series input arguments
     int tLine,nObjects;
     vector<vector<double> > PriceVec;
     vector<vector<double> > returnVec;
     // Activates ewmaRisks function input arguments from data series
file
     dataSeries(PriceVec,returnVec,tLine,nObjects);

     vector<vector<double> > spotPriceVec;     // Spot price vector
     vector<vector<double> > strikePriceVec;     // Strike price vector

     vector<vector<double> > maturity;       // Maturity dates vector

     vector<vector<double> > interestRate;  //  Interest rate -Repo
rates vecto
     dataParameters(spotPriceVec,strikePriceVec,maturity,interestRate);

     //cout << "\n ....tLine]:"<<tLine;
     vector<double> mullerBox;
     vector<vector<double> > mullerBoxVec;
     normalRV(tLine,maturity,mullerBox,mullerBoxVec);
     // Activates uncorrelated Brownian Motion
     vector<vector<double> > wiener;
     uncorBrownM(tLine,nObjects,maturity,mullerBox,wiener);

     //defines output arguments from ewmaRisks function
     vector<vector<double> > ewmaVarVec;       // EWMA volatilities

     vector<vector<vector<double> > > ewmaCovarMatrix;
     vector<vector<vector<double> > > ewmaCorrMatrix;
     // Calculates ewma vectors matrices from ewmaRisks function
arguments
     ewmaRisks(returnVec,tLine,nObjects // Input arguments
                 ,ewmaVarVec,ewmaCovarMatrix,ewmaCorrMatrix //Output
arguments
                 );

     //defines output arguments from choleskyMatrix function
     vector<vector<vector<double> > > choleskyMatrix;
     //Calculates choleskyMatrix from ewmaRisks function and data
series arguments
     cholesky(tLine,nObjects,ewmaCorrMatrix// Input arguments
                       ,choleskyMatrix //Output arguments
                       );
```

UNIVERSITY of the
WESTERN CAPE
https://etd.uwc.ac.za/

```
     vector<vector<double> > corBrownVec;
     corrBrownM(tLine,nObjects,maturity,
choleskyMatrix,wiener,corBrownVec);
     vector<double> zarDF;
     discountCurve(zarDF); // Discount factor vector
     vector<vector<double> > assetPrice;
     nAssets(nObjects,tLine,spotPriceVec,maturity,interestRate,strikePr
iceVec,zarDF,corBrownVec,wh88,assetPrice);//ewmaVarVec//garchForecast
//sabrVol/hw88--Variable 2-d vols function

return 0;
}
```

12.3 Market data

Header file 12.3a

```
// FILE NAME: MARKETDATA.h
/* WRITTEN BY: Africa M Chamane
   FOR: Computational Finance course - Msc Thesis
   DATE: Nov 2008
   PRUPOSE: Upload market data file from local directory
            and reformat for functions utilities
   Header file
 */
#ifndef MARKETDATA_H
#define MARKETDATA_H

// link inbuilt libraries
#include <vector>
using std::vector;

// Defines market data series uploading function
void dataSeries(vector<vector<double> >& PriceVec // Price series
vector
                 ,vector<vector<double> >& returnVec //Return vector
                 ,int& tSeries
                 ,int& nObjects
                   );
// Defines market data parameters uploading function
void dataParameters(vector<vector<double> >& spotPriceVec
                        ,vector<vector<double> >& strikePriceVec
                        ,vector<vector<double> >& maturity
                       ,vector<vector<double> >& interestRate

                       );
// Defines yield curve data uploading function
void discountCurve(vector<double>& zarDF // Discount factor vector
      );
#endif // MARKETDATA_H
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

Source code 12.3b

```
// FILE: MARKETDATA.cpp
// {L}
// Source file

#include "MARKETDATA.h"
// link inbuilt libraries
#include<iostream>
#include <cmath>                    // for trigonometry functions
#include <string>
#include <fstream>
#include <vector>
#include <iomanip>
using std::vector;
using namespace std;

// Provide a set of methods for uploading datafile from a directory
void dataParameters(vector<vector<double> >& spotPriceVec
                        ,vector<vector<double> >& strikePriceVec
                        ,vector<vector<double> >& maturity
                       ,vector<vector<double> >& interestRate

                       ){
     using std::cout;
     using std::cin;

     string iFile;       // string to hold data file name
     cout << "Enter data parameters file: ";
     cin >> iFile;       // read in data file name
     // create an input file stream called infile that allows data to
be read from
     // from file via the stream
     ifstream infile( iFile.c_str() );
     // check that the stream and hence the file exists. If not error
message
      if(!infile)
        cerr << "error: unable to open input file: " << iFile << "\n";
        double buffer;                        // to hold individual
datum from data file
     vector<double> paradata;        // to hold contents of data file
in a vector

     // loop over the infile until we get to end of file and nothing
gets
     // sent to buffer
      while (infile >> buffer) {
           paradata.push_back(buffer);    // fill up the vector data
with the buffer
      }// while loop ends
     const int nParameters=4;    // number of currencies
    const int nObjects =
static_cast<int>(paradata.size()/nParameters); // returns area of
```

UNIVERSITY of the WESTERN CAPE

https://etd.uwc.ac.za/

```
datafile, here is tLine*1
    int factor,j;

    //Declares global vector output
     spotPriceVec=vector<vector<double>
>(nObjects,vector<double>(nParameters));
     strikePriceVec=vector<vector<double>
>(nObjects,vector<double>(nParameters));
     maturity=vector<vector<double>
>(nObjects,vector<double>(nParameters)); // volatility of volatility
vector
     interestRate=vector<vector<double>
>(nObjects,vector<double>(nParameters));

     /*Extracting and converting/partitioning input data file
       into market data price series data
       Reference to objects positions
          0..ZAR
          1..GBP
          2..JPY
          3..EUR
     */
     for(int p=0;p < nParameters; ++p){
          factor=0;
          for(int n=0; n < nObjects; n++){
               j=p+n+factor; //Reposition allocator point
               if (p==0){
                    spotPriceVec[n][p]=paradata[j]; //Allocate spot
price
               }
               if (p==1){
                    strikePriceVec[n][p]=paradata[j]; //Allocate strike
price
               }
               if (p==2){
                    maturity[n][p]=paradata[j]; //Allocate maturities
               }
               if (p==3){
                    interestRate[n][p]=paradata[j]; //Allocate interest
rates
               }
               factor=factor+(nParameters-1);   // Move to the next
column in datafile
          }//End inner for loop
     }//End outer for loop
}// End main method

// Opens price seriese data file frpl local directory
void dataSeries(vector<vector<double> >& PriceVec // Price series
vector
                  ,vector<vector<double> >& returnVec //Return vector
                  ,int& tSeries
                  ,int& nObjects
                    ){
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
using std::cout;
     using std::cin;

     string iFile;        // string to hold data file name
     cout << "Enter data series file: ";
     cin >> iFile;        // read in data file name
     // create an input file stream called infile that allows data to
be read from
     // from file via the stream
     ifstream infile( iFile.c_str() );
     // check that the stream and hence the file exists. If not error
message
      if(!infile)
        cerr << "error: unable to open input file: " << iFile << "\n";
        double buffer;                         // to hold individual
datum from data file
     vector<double> seriesdata;        // to hold contents of data file
in a vector

     // loop over the infile until we get to end of file and nothing
gets
     // sent to buffer
      while (infile >> buffer) {
           seriesdata.push_back(buffer);   // fill up the vector data
with the buffer
      }// while loop ends
     nObjects=4;   // number of currencies
    tSeries = static_cast<int>(seriesdata.size()/nObjects); // returns
area of datafile, here is tLine*1
    int factor,j;

    //Declares global vector output
     PriceVec=vector<vector<double>
>(nObjects,vector<double>(tSeries));
     returnVec=vector<vector<double>
>(nObjects,vector<double>(tSeries));

     //Extracting and converting/partitioning input data file
     //into market data price series data
     for(int n=0;n < nObjects; ++n){
          factor=0;
          for(int t=0; t < tSeries; t++){
               j=n+t+factor; //Reposition allocator point
               PriceVec[n][t]=seriesdata[j]; //Allocate spot price

               factor=factor+(nObjects-1); // Move to the next column
in datafile
          }//End inner for loop
     }//End outer for loop for price series extraction
     // Calculating the returns from price input data file
     for(int n=0;n < nObjects; ++n){

          for(int t=1; t < tSeries; t++){
               returnVec[n][t]=log(PriceVec[n][t]/PriceVec[n][t-1]); //
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

```
input log-returns into returnVvec file
          }
          returnVec[n][0]=0.0; //returns at the begining of day    1
     }//End outer for loop for reuturns calc


}// End main method

void discountCurve(vector<double>& zarDF // Discount factor vector
     ){
     using std::cout;
     using std::cin;

     string iFile;        // string to hold data file name
     cout << "\n Enter yield curve file name: ";
     cin >> iFile;        // read in data file name
     // create an input file stream called infile that allows data to
be read from
     // from file via the stream
     ifstream infile( iFile.c_str() );
     // check that the stream and hence the file exists. If not error
message
      if(!infile)
        cerr << "error: unable to open input file: " << iFile << "\n";
        double buffer;                         // to hold individual
datum from data file
     vector<double> yield;        // to hold contents of data file in a
vector

     // loop over the infile until we get to end of file and nothing
gets
     // sent to buffer
      while (infile >> buffer) {
           yield.push_back(buffer);   // fill up the vector data with
the buffer
      }// while loop ends
     int nObjects=1;   // number of currencies
    int tSeries = static_cast<int>(yield.size()/nObjects); // returns
area of datafile, here is tLine*1
    cout <<"\n tSeries";
    cout <<tSeries;
    int factor,j;

    //Declares global vector output
     zarDF=vector<double>(tSeries);

     //Extracting and converting/partitioning input data file
     //into market data price series data
     for(int t=0; t < tSeries; ++t){
          zarDF[t]=exp(-0.01*yield[t]*t/365); //Allocate spot price
     }//End inner for loop

}// End main method
```

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/

UNIVERSITY of the
WESTERN CAPE

https://etd.uwc.ac.za/