Pricing of Contingent Interest Rate Claims, Foundations and Application of the Hull-White Extended Vasicek Term Structure model

Master Thesis

Master of Science in Applied Economics and Finance

Anders Bæk Holmgaard

Cpr xxxxxx-xxxx

(anli04ag@mail.cbs.dk)

January 29, 2013

Supervisor

Jesper Lund, Department of Finance

Copenhagen Business School, 2013

Characters: 178.041
Normal Pages: 78.26

Anders Bæk Holmgaard
M.Sc. stud, Applied Economics and Finance
Copenhagen Business School
anli04ag@mail.cbs.dk
anders.rikke@gmail.com

Danish title:
Prisning af betingede rentekrav, implementering og anvendelse af Hull-White Extended Vasicek rentestrukturmodellen.

Typesetting: by the author using LaTeX.[1]
Graphs and figures: by the author using OriginLab, OriginPro™.
Scripts: by the author using Microsoft Excel™, Visual Basic for Applications.
Market Data: collected by the author using Bloomberg™, >SWPM< and >VCUB<.
Production: Pitney Bowes.

[1] 11pitch, margins; left 3.0, right 2.5, line spacing one.

Abstract

This thesis is concerned with the modeling of stochastic interests using Gaussian mean-reverting short rate term structure models. Particularly, we consider the one-factor Vasicek model and its close descendants; the Hull-White and the Hull-White Extended Vasicek models. We provide a detailed derivation of the expressions behind the Vasicek model, and these form the basis for discussing the differences to the more elaborate versions of Vasicek's original model. More specifically, we consider the distributional assumptions of the short rate, forward rates and bond prices, as well as their implications for the volatility structure implied by the given model. Based on this review, the Hull-White Extended Vasicek model has been chosen as platform for the subsequent analysis, due to the significantly improved flexibility offered by this model. A piecewise linear volatility function has been derived for this purpose.

To enable pricing of complex and exotic interest rate derivatives a numerical procedure is needed, whenever a analytical solution is absent. We follow the strategy of a Monte Carlo setup to implement the stochastic differential equation, describing the short rate dynamics within the Hull-White Extended Vasicek model. The model has been implemented in Excel Visual Basic and calibrated to the Black76 volatility surface of European Cap (Floors). Moreover, its numerical performance has been tested by comparing to market prices. The implementation has been applied to two illustrative applications related to pricing of different complex derivatives: a Barrier Swap and a Range Accrual note. To justify the Monte Carlo setup, a pseudo path-dependency has been imposed on the latter. For a well full-grid calibrated model, we find that the Hull-White Extended Vasicek model reasonably captures the prices of options not too far away from ATM. Consequently, the model is less suited for instrument exposed to broader areas of the volatility surface, as it requires repeatedly re-calibration, thereby becoming expensive in time and generality.

Contents

Abbreviations VII

Preface IX

Introduction 1
- Problem Identification . . . 1
- Limitations . . . 2
- Method and Tools . . . 2
- Structure of the Thesis . . . 3

Part I Foundations and Theory 4

1 Basic Rates and Notation 4
- 1.1 Spot Rates, Forward Rates and more . . . 4
- 1.2 The Instantaneous Short Rate . . . 5
- 1.3 Libor Rates . . . 5

2 Pricing of Contingent Claims - Risk Neutral Pricing 6
- 2.1 The PDE Approach . . . 6
  - 2.1.1 The Feynman-Kac Formula - an Equivalent Representation . . . 8
- 2.2 The Martingale Approach . . . 8
  - 2.2.1 Martingales and the Change of Measure Technique . . . 9
  - 2.2.2 Numeraires and the Equivalent Martingale Measure . . . 10
  - 2.2.3 The Forward Rate - a Martingale under the Q^T-Measure . . . 11
  - 2.2.4 The Money Market Account as Numeraire - an Equivalent Q-Measure . . . 12

3 Applied Fixed Income Instruments 13
- 3.1 Fixed for Floating Swaps . . . 13
- 3.2 European Caps and Floors . . . 14
  - 3.2.1 The Put/Call Parity . . . 15
- 3.3 Digital Caps and Floors . . . 16
- 3.4 Exotic Libor Based Range Accruals . . . 16

4 Standard Vanilla Models 17
- 4.1 The Black76 model . . . 17
- 4.2 The Normal Model . . . 18
- 4.3 Interpretation of the Volatility Specification . . . 19
- 4.4 The Volatility Term Structure and Skew . . . 20

5 Term Structure Models **22**
5.1 The Vasicek Model . . . 22
5.1.1 Definition of the model . . . 22
5.1.2 Moments and Distribution of the Short Rate . . . 23
5.1.3 Future Bond Prices . . . 25
5.1.4 The Implied Term Structure of Interest Rates . . . 27
5.2 The Hull-White Vasicek Model . . . 28
5.2.1 Definition of the model . . . 29
5.2.2 Calibration to the Current Yield Curve . . . 30
5.2.3 Future Bond Prices . . . 32
5.3 The Hull-White Extended Vasicek Model . . . 33
5.3.1 Definition of the model . . . 34
5.3.2 Moments and Distribution of the Short Rate . . . 34
5.3.3 Future Bond Prices . . . 34
5.3.4 The Implied Term Structure of Interest Rates . . . 35
5.3.5 Derivation of a Piecewise Linear Volatility Function . . . 35
5.4 European Options Pricing under Vasicek and the Descendants Models . . . 36
5.4.1 Options on Zero Coupon Bonds . . . 36
5.4.2 Replication Strategy for European Caps and Floors . . . 37
5.4.3 Vasicek and the Volatility Skew . . . 38

Part II Implementation and Calibration **39**

6 Outline of Market Data **40**
6.1 Market Data and Source . . . 40

7 Analytic Pricing of European Caps and Floors **42**
7.1 The Notion of a Good Term Structure Model . . . 42
7.1.1 The Calibration Process . . . 43
7.1.2 The Calibration Problem . . . 44
7.1.3 Implementation and Source Code . . . 46
7.2 Results and Performance . . . 46
7.3 Conclusion . . . 53

8 Simulation of the Hull-White Extended Vasicek model **55**
8.1 Monte Carlo Simulation . . . 55
8.1.1 Euler Discretization and Numerical Precision . . . 56
8.1.2 Random Numbers . . . 58
8.1.3 Implementation and Source Code . . . 58
8.1.4 Test of Performance and Pricing Capabilities . . . 61

Part III Application and Case Sections **63**

9 Pricing of Two Different Complex Interest Rate Derivatives **64**
9.1 Barrier Swap . . . 64
9.1.1 Calibration . . . 65
9.1.2 Results . . . 66
9.2 Range Accrual Swap . . . 68
9.2.1 Calibration . . . 70
9.2.2 Results . . . 71

10 Conclusion and Closing Remarks **73**

Appendices **77**

A Dataset **77**
A.1 Market Data . . . 77
Yield and Discount Curves . . . 77
Forward Swap Matrix . . . 81
Black76 Cap Vol Surface - Relative Strike levels . . . 81
Black76 Cap Vol Surface - Absolute Strike levels . . . 81

B Graphical User Interface **81**

C VBA Source Script **82**
C.1 Basic Rates . . . 82
C.1.1 ParSwap . . . 82
C.1.2 ResetRates . . . 83
C.1.3 ZeroRates . . . 83
C.1.4 ForwardDisc . . . 83
C.2 Standard Vanilla Models . . . 85
C.2.1 BblackCall . . . 85
C.2.2 BblackPut . . . 85
C.2.3 BnormCall . . . 85
C.2.4 BnormPut . . . 85
C.2.5 CapBS . . . 86
C.2.6 FloorBS . . . 86
C.3 HWExtV model - Analytical setup . . . 88
C.3.1 BHW . . . 88
C.3.2 PhiHWExtV . . . 88
C.3.3 ZCB-HW . . . 89
C.3.4 OptZCBHW . . . 89
C.3.5 CapFloorReplicationHW . . . 90
C.4 HWExtV Monte Carlo Simulation Source Code . . . 92
C.4.1 Preample - Public Variables . . . 92
C.4.2 EventSchedule . . . 92
C.4.3 Simulation-Dates . . . 93
C.4.4 AccrualEventDates . . . 95
C.4.5 VolInitialize . . . 96
C.4.6 LIntPdf . . . 97
C.4.7 Hull-White-MC-Sub . . . 99
C.4.8 PayOffCapFloor . . . 102
C.4.9 PayOffDigi . . . 104
C.4.10 PayOffRangeAccrual . . . 106
C.4.11 ImplVol . . . 110
C.4.12 PriceInstr-Routine . . . 110
C.4.13 UpdateCF-Routine . . . 110
C.4.14 ClrEventSchedule . . . 111
C.5 Third Party Code . . . 112
C.5.1 QuickSort1 . . . 112
C.5.2 RandNorm . . . 112

D Re-Calibrated Parametrization (Barrier Swap) **114**

E Re-Calibrated Parametrization (Range Accrual Swap) **115**

List of Figures

4.1 The Volatility Term Structure and Skew 21

5.1 The Vasicek Model - an graphical outline of the dynamics 28
5.2 Vasicek and the Volatility Skew - ZCB Options 38
5.3 Vasicek and the Volatility Skew - Implications on Caps (Floors) 39

6.1 Graphical Outline of the Market Data 41

7.1 Diagram of the Calibration Process 43
7.2 HWV - Calibration Results 47
7.3 HWExtV - Calibration Results 48
7.4 Cross-Model Comparison of the Parameter Vector of σ_i 49
7.5 HWExtV - Calibration Results versus ATMF (Full Grid Parametrization) 50
7.6 HWExtV - Skew Characteristics 52
7.7 HWExtV - Black76 Implied Volatility Grid 53
7.8 HWExtV - Price Residuals (Grid) 53

8.1 Process Flow - Developed Source Code 59

9.1 Barrier Swap - Payoff Profile of a Barrier Leg 65
9.2 Barrier Swap - Context of Application 65
9.3 Barrier Swap - The Forward Curve vs Strike levels 66
9.4 Barrier Swap - Re-Calibration versus Barrier Strikes 66
9.5 Payoff from a Range Accrual Leg 68
9.6 Range Accrual Swap - Context of Application 69
9.7 Range Accrual Swap - The Forward Curve vs Accrual Strikes. 69
9.8 Range Accrual Swap - Re-Calibration versus Accrual Strikes 71

B.1 Graphical User Interface 81

D.1 Barrier Swap - Cross-Calibration Comparison of the Parameter Vector of σ_i 114

E.1 Range Accrual Swap - Cross-Calibration Comparison of the Parameter Vector of σ_i . 115

List of Tables

6.1 Market Data . . . 41

7.1 Calibrated Model Parametrizations . . . 47
7.2 Price Comparison Across Strike Levels of a 5Y 10m$ European Cap . . . 54

8.1 Price Errors for Various Combinations of Δt_j and N . . . 62
8.2 Price Errors for Various European Caps . . . 63

9.1 Barrier Swap - Price Composition and Model Sensitivities . . . 67
9.2 Structure Range Accrual Note - Term Sheet . . . 70
9.3 Range Accrual Swap - Price Composition and Model Sensitivities . . . 72

A.1 Market Data - Yield and Discount Curves . . . 77
A.2 Forward Swap Matrix . . . 78
A.3 Market Data - Black76 Cap Vol Surface - Relative Strike levels . . . 79
A.4 Market Data - Black76 Cap Vol Surface - Abolute Strike levels . . . 80

Abbreviations

APT	Abitrage Price Theory
ATMF	At The Money Forward
BD	Business days
Black-vol	Log normal volatility parameter in the Black76 model
BM	Brownian Motion
BP-vol	Basis Point Volatility - volatility measure / volatility parameter in the Normal model
DV01	Dollar Duration or DV01 is defined as the first derivative of the value with respect to the yield. ($ per 1 basis point change in the yield).
EMM	The Equivalent Martingale Measure Result
FEYK	The Feynman-Kac formula
FRA	Forward Rate Agreement
GBM	Geometrical Brownian Motion
HJM	Heath Jarrow Morton framework
HWExtV	The Hull-White Extended Vasicek model - meaning an extension of the original Vasicek77-model where κ is a constant, $\theta(t)$ is chosen so as to perfectly fit the current term structure of interest rates and $\sigma(t)$ is chosen so as to match the current volatility term structure as close as possible.
HWV	The Hull White Vasicek model - meaning an extension of the original Vasicek77-model where κ and σ are both constants and $\theta(t)$ is chosen so as to perfectly fit the current term structure of interest rates.
IRS	Interest Rate Swap
ITM	In The Money
LMA	Levenberg-Marquardt algorithm
NMA	Nelder-Mead downhill simplex algorithm
NWLS	Non-Weighted Least Squares
ODE	Ordinary Differential Equation
OTM	Out Of The Money
PDE	Partial Differential Equation

RNG	Random Number Generator
SDE	Stochastic differential Equation
TSM	Term Structure Model
UDF	Visual Basic - User defined functions
VBA	Microsoft Visual Basic for Applications™

Preface

"Theory without practice is idle, practice without theory is blind"

— Ancient Chinese Proverb

From a practitioner's point of view, I am well-traveled working with pricing of linear and (complex) contingent interest claims.[2] Through my studies in finance at the Copenhagen Business School (B.Sc. Finance, HD(F) and M.Sc. Applied Economics and Finance), I have achieved knowledge on the fundamental financial rates theory, mainly, but not limited to, static term structure modeling and bond price analysis. This aim of this thesis was to expand this theoretical knowledge to full term structure modeling under stochastic interest rates. In particular, emphasis has been on the arguments and theory, as well as implementational aspects, necessary to enable pricing of more complex interest rate derivatives in practice. Although, the Black76 vanilla model, for pricing of interest rate options, is an integrated part of most finance courses, the notion of interest rate volatility and calibration of the volatility surface (broadly speaking) is still unfamiliar to most practitioners.[3]

While the theoretical part of this thesis has been both insightful and challenging, mostly, due to the mathematical derivations, the applied part has been outermost inspiring and motivating, though it at the same time proved to be a major task(!) It has been a massive work to transform theory into a functional implementation, ensuring proper calibration procedures, writing simulation engine, payoff and pricing modules, and at the same time, ensuring consistency between the gained results and the financial theory. Consistently, it has been my aim to implement the Hull-White Extended Vasicek model in such a way that it (at least to some extent) mimics a live setting. Although, the result is far from the complexity and rigor of term structure modeling used in investment banks nowadays, it does provide insight on the basic concepts and gives a small glimpse of the amount of work, coordination and complexity provided by modern quant departments. Furthermore, the notion of "model risk" has become very familiar and readily recognizable.

It is my hope that the theoretical and practical insight on full term structure modeling acquired through this thesis will empower my skills in liaising, both internally, between the different sections within front office (management, trading, quants, risk managers and more), as well as externally, with customers, when translating back and forth between practical issues and theoretical argumentation. I believe that this thesis has improved my abilities to provide value toward customers and the represented financial institution. Last but not least, it has broadened my horizon and incited my curiosity on the subject even further.

Finally, I would like to express deep gratitude to my supervisor, Associate Professor Jesper Lund. His advise, guidance and understanding have been invaluable. Also, I am deeply indebted to Nanna Holmgaard List (Ph.D. stud Quantum Chemistry, University of Southern Denmark) for her advise while working on the mathematical parts, as well as on reviewing the final thesis. I owe a lot of thanks to Morten Skelmose (HD stud, University of Southern Denmark), who has contributed with highly valuable discussions on structuring of the VBA code, as well as on numerous

[2]The author has been full time employee with Nordea since 2001, working at Nordea Capital Market since 2007 - mainly as Senior Derivatives Marketer toward the large Cap Corporate segments.

[3]At least to ourselves.

technical questions on the use of OriginPro™, Latex and more. I am grateful to my wife Rikke Bæk Holmgaard (Ph.D. Memorial Sloan-Kettering Cancer Center) for her valuable comments during the process of reviewing the thesis. I thank Steen Hahn Andersen, Director Nordea Markets for his support on the logistics of the (at times impossible) task of combining studies and full time employment. Lastly, I thank Anders Meinert, Director Risk Management Nordea Bank AB, for his initial encouragement and inspiration that led to the progression of my general studies to the M.Sc. level.

To my wife for her persistent and loving support through times of great geographical distance.

Anders Bæk Holmgaard
Copenhagen, January 2013

Introduction

The use of full term structure models have become an important tool for the pricing of more complex derivatives not readily quoted in the market and where no analytical solution exists. Overall, the aim of this thesis is to gain both theoretical and practical insight on full term structure modeling to better understand the concepts of stochastic interest rates, different volatility measures and the pricing of more (complex) contingent interest rate claims in practice. For analysis purposes, we have chosen the one-factor (Hull-White Extended) Vasicek framework, since it, from a theoretical point of view, constitutes a natural point of departure for more elaborate models.

Truly enough, the one-factor Hull -White Extended Vasicek framework is now outdated as means of pricing measure. However, in the beginning of the 2000s many banks still used this framework (or similar) for pricing and handling of interest volatility books (e.g., pricing embedded bond options).[4] Therefore, the model framework is of relevance, and for our purposes, interesting to investigate further.

> *"... the Hull-White Extension of the Vasicek model is one of the historically most important interest rate models, being still nowadays used for risk-management purposes."*
> [1, p. 72].

Problem Identification

The main purposes of this thesis were to:

- Give a thorough exposition of the Vasicek77 model as well as of the Hull-White and Hull-White Extended versions of the model (derivations, properties and the theoretical abilities in fitting the current yield curve and volatility term structure).
- Present a numerical procedure (Monte Carlo simulation) for the valuation of complex contingent interest rate claims under the Hull-White Extended Vasicek model (properties of convergence).
- Outline a strategy for the derived method's combined evaluation on a computer (Excel Visual Basic), including the development and implementation of code for:
    1) Analytical calibration of the model to the current volatility surface.
    2) Monte Carlo simulation setup, including pricing modules for various payoffs (European Caps, Digitals and Range Accruals).
- Perform a analytical "full grid" model calibration to the US Cap volatility surface (discussions on the skew fit as well as ways to improve the calibration result).
- To apply the developed tools to price three different cases of complex interest rate derivatives (Barrier Swap and Range Accrual Swap), and for each of these test cases, test the performance and generality (robustness and sensitivity toward changes in the parametrization) of the models.

[4] Discussions with Supervisor.

- To conclude the thesis with a discussion on pros and cons for a live implementation of the Hull-White Extended Vasicek model.

To ensure the theoretical foundation, we further provide an introduction to 1) the applied financial instruments, 2) the theory related to risk neutral pricing under stochastic interest rates (the fundamental partial differential equation and Martingale approaches), and 3) the benchmark vanilla models (Black76 and the Normal model) for quotation interest rate volatility (distributional assumptions, interpretation of the volatility specifications and issues versus the volatility skew) that set the stage for the later full term structure modeling.

Delimitation

In order to be consistent with the scope of the thesis the following limitations have been implemented:

- Only risk neutral measures are considered, i.e., the estimation of risk premiums are disregarded.
- We assume perfect markets (arbitrage free with no frictions). The significance of this assumption will not be discussed.
- No accountancy is made for competing models.
- In the numerical sections, we apply a first order Euler Scheme means of discretization tool for the Stochastic differential equation. Other methods will not be discussed and we only mention their existence in passing.
- The model is calibrated to Cap (Floor) volatilities only.[5]
- We generally disregard the calculation of risk-figures.[6]
- No regards is made to business calendars or various other market conventions.

Method and Tools

The developed framework is implemented using Microsoft Visual Basic for Applications™ [VBA]. The high-level language of VBA is not the right tool for real live applications or advanced academic analysis, due to the lack of flexibility and generally poor performance[7]. We became aware of these drawbacks during the implementation process, where, e.g., matrix computations were rather cumbersome and the large amounts of data/calculations significantly reduced its performance, even with proper variable declarations and the use of third party code for random number generation. Further, and even more severe, the relatively low memory capacity of matrix dimensions, meant that the total number of simulations for longer dated instruments were significantly reduced. A feature, which obviously limits the usage of the program.

VBA was, however, initially chosen as it is readily available to most practitioners being an integrated part of the Microsoft Office package. Moreover, it offers easy access to user defined functions [UDF], automated processes, an advanced solver algorithm and other functionality with little introduction needed. However, evaluating our work on future implementations of this size, we would highly recommend others to use alternatives, such as MatLab.

[5] While the inclusion of Swaptions to the calibration-set would have been preferred, is has been delineated due to time space and constraints of the thesis. We seek to remedy this issue by only using the model for pricing of Libor based payoffs.

[6] However, the notion of dollar duration is considered known and it is used in the Case sections.

[7] Real live applications are often implemented in C++ or other low-level languages, whereas academic analysis and pre-implementation scripts often are performed in e.g. MatLab for ease of programming.

Structure of the Thesis

This thesis is structured in three main parts. The first part (chapters 1-5) establishes the theoretical background necessary for a practical implementation of the Hull-White extended Vasicek term-structure model.

- Chapter 1 introduces briefly the basic concepts of interest rates.
- Chapter 2 describes two approaches to achieving consistent arbitrage-free prices of contingent interest rate claims. First, the fundamental partial differential equation approach is reviewed together with the equivalent Feynman-Kac representation for general risk neutral valuation using numerical procedures. Secondly, the alternative martingale approach to establishment of arbitrage-free price is considered.
- Chapter 3 provides a short outline of the derivative instruments applied in this thesis (Interest Rate Swaps, European Caps (Floors) and Digital Caps and Range Accrual Swaps).
- Chapter 4 concerns the vanilla models used as benchmark for the quotation of interest rate volatility in the market (Black76 and the Normal model). Particular focus is on the theoretical assumptions underlying these models, the interpretation of the volatility parameters as well as their practical application. Moreover, the inherent problems when considering the observed volatility skew are discussed. This latter point provides layup and arguments behind our later implementation of a full term structure model.
- Chapter 5 provides a detailed exposition of the main theory behind the closely related models; the Classical Equilibrium Vasicek model and the two Arbitrage-Free descendants, the Hull-White Vasicek and the Hull-White Extended Vasicek models. The aim of this chapter is to establish the theoretical foundation, both in terms of mathematical derivations and financial interpretation, needed for the later application of these models to pricing of interest rate derivatives via analytical and numerical methods.

The second part (chapter 6-8) covers the implementational aspects of the Hull-White Vasicek and Extended Vasicek models.

- Chapter 6 discusses the market data used for the calibration of the implemented models.
- Chapter 7 discusses the calibration procedures behind the models to the standard European Cap (Floor) volatilities. Moreover, the performance of the models in fitting the volatility surface are reviewed, and based on this, the model of choice is identified. Finally, the global best-fit parametrization for this model is established.
- Chapter 8 considers the theory of simulating the Hull-White Extended Vasicek model by means of a standard Monte Carlo setup using a Euler discretization scheme. Moreover, chapter 8 describes the programming structure of the underlying Visual Basic scripts as well as the performance of the simulation setup in fitting prices of standard products based on the global-fit parametrization determined in provided in chapter 7.

The third part (chapter 9) of this thesis comprises two practical applications of the model of choice for pricing of more complex interest rate derivatives.

- Chapter 9 focuses on the applications of the developed setup to price two cases of complex interest rate derivatives; a Barrier Swap and a Range Accrual Swap. Particularly, each application is presented by means of a small business case with the aim of adding a touch of real life to the context of the financial and theoretical aspects covered by the previous chapters. The achieved modeled pricing precision and as such the generality of the global best-fit parametrization presented in chapter 7 are assessed.

Finally, chapter 10 concludes the thesis and comments further on the issues of model risk and the problems inherent in using a one-factor model for the pricing of more complex structures.

Part I
Foundations and Theory

Chapter 1

Basic Rates and Notation

Chapter 1 gives a brief outline of the basic concepts of interest rates. The concept of the Instantaneous Short Rate is important for the later development of short-rate models. Explanations are kept short as most of content is considered known by most readers. All sections in chapter 1 follows closely ref [2, chpt. 2].

1.1 Spot Rates, Forward Rates and more

Simply Compounded Rates The simple or simply compounded **spot rate** for the period $[t, T]$, where $\tau_t = (T\text{-}\, t)$, prevailing at time t is defined as

$$R(t,T) = \frac{1 - P(t,T)}{\tau_t\, P(t,T)}. \tag{1.1}$$

This shows that $P(t,T)\big(1 + R(t,T)\tau_t\big) = 1$, so that a bond price can be expressed in terms of the spot rate as

$$P(t,T) = \frac{1}{\big(1 + R(t,T)\tau_t\big)}. \tag{1.2}$$

The simply compounded **forward rate** for the period $[S, T]$, prevailing at time t is defined as

$$F(t;S,T) = \frac{P(t,S) - P(t,T)}{\tau_S\, P(t,T)}, \tag{1.3}$$

which is equivalent to

$$1 + \tau_S F(t;S,T) = \frac{P(t,S)}{P(t,T)}. \tag{1.4}$$

Continuously Compounded Rates The continuously compounded **spot rate** for the period $[t, T]$, prevailing at time t is defined as

$$y(t,T) = f(t;t,T) = -\frac{\ln P(t,T)}{\tau_t}, \tag{1.5}$$

where $f(t;s,T)$ is the continuously compounded **forward rate** for the period $[S, T]$, prevailing at time t

$$f(t;S,T) = -\frac{\ln P(t,T) - \ln P(t,S)}{\tau_S}. \tag{1.6}$$

1.2 The Instantaneous Short Rate

Conceptually, in the limit as $S \to T$ the continuously compounded, instantaneous and unobservable rates are achieved. The **instantaneous forward rate** is defined as

$$f(t,T) = \lim_{S \to T} f(t;S,T) = -\frac{\partial \ln P(t;T)}{\partial T}, \tag{1.7}$$

while the **instantaneous short rate** becomes

$$r(t) = f(t,t) \;=\; \lim_{\tau_t \to 0} y(t,T). \tag{1.8}$$

Note that (1.7) together with the a priori condition of $P(T,T) = 1$[1] gives

$$P(t,T) = e^{-\int_t^T f(t,s)ds}. \tag{1.9}$$

1.3 Libor Rates

Libor rates (London Interbank Offered Rate) indicate the price of interbank borrowing. Rates are quoted daily at maturities ranging from over-night (O/N) to 12 month. Libor rates serve as global primary benchmark indices for settlement of financial over the counter [OTC] interest rate derivatives. Generally, Libor rates are simply-compounded and the currently observed Libor spot rate at time t for the period $[t,T]$ reads

$$L(t;t,T) = F(t;t,T) = R(t,T) = \frac{1 - P(t,T)}{\tau_t \, P(t,T)}. \tag{1.10}$$

Thus, the forward Libor rate at time t for the period $[S,T]$ is given as

$$L(t;S,T) = F(t;S,T) = \frac{P(t,S) - P(t,T)}{\tau_S \, P(t,T)}. \tag{1.11}$$

Throughout this thesis the tenor structure used is 3M USD Libor and for the sake of notational convenience, we often use the shorthand notation

$$L_S(t) = L(t;S,T), \tag{1.12}$$

to designate the forward Libor rate at time t for the period $[S,T]$ for a pre-specified tenor structure. Hence, the future Libor spot rate, observed at time S, is $L_S(S)$, and thus, the current spot Libor rate is $L_t(t)$.

[1] That is, all bonds matures at par.

Chapter 2

Pricing of Contingent Claims - Risk Neutral Pricing

This chapter reviews the theoretical foundation needed to consistently price contingent interest rate claims in an arbitrage free manor. First, we develop the fundamental partial differential equation [PDE] approach using the Arbitrage Price Theory [APT] restriction and derive the PDE for bond prices. In this context, we also consider the equivalent Feynman-Kac representation [FEYN] for general risk neutral valuation using numerical procedures. Second, we consider the martingale approach to arbitrage-free pricing and cover the basic properties of martingales, the equivalent martingale measure result [EMM], change-of-numeraire technique and the rationale behind two of the numeraires applied in the later chapters, namely the money market account and the terminal measure. The underlying APT of martingales is complex and beyond the scope of this thesis.

2.1 The PDE Approach

In the following, we develop the framework on which the single factor term structure models are build. The method is used in the subsequent chapters to develop the Vasicek model and its descendants. When introducing uncertainty about future interest rates we need to establish an arbitrage-free relationship between the price of a certain interest rate derivative and the underlying interest rate.

Aside from our general assumption of perfect markets (arbitrage free and frictionless), we make the following assumptions:

- All bonds are assumed infinitely divisible.
- Investors have positive marginal utility of wealth at all wealth levels.
- All derivatives considered depend on a single state variable; the instantaneous short rate.
- The instantaneous short rate follows the general stochastic differential equation [SDE] outlined below.

$$dr_t = \mu_{r_t} dt + \sigma_{r_t} dW_t, \tag{2.1}$$

where $\mu(r_t)$ and $\sigma(r_t)$ denote the drift and volatility functions, while W_t follows a Brownian motion. Particularly, the underlying variable r_t could have been any variable x_t, e.g. a stock price, but of course for our use, we define the instantaneous short rate r_t.

Next, we follow ref [3, chpt. 3], and introduce a derivative in the form of a bond price $P(t, T)$ with a yet unknown dependency on r_t. By Itô's lemma,[1] we get the following stochastic differential

[1] [4, Theorem 1.1.5]

equation [SDE]

$$dP(t,T) = \mu_{P(t,T)}P(t,T)dt + \sigma_{P(t,T)}P(t,T)dW_t, \tag{2.2}$$

where $\mu_{P(t,T)}P(t,T)$ and $\sigma_{P(t,T)}P(t,T)$ are given as

$$\mu_{P(t,T)}P(t,T) = \frac{\partial P}{\partial r_t}\mu(r_t) + \frac{\partial P}{\partial t} + \frac{1}{2}\frac{\partial^2 P}{\partial r_t^2}\sigma(r_t)^2 \tag{2.3}$$

$$\sigma_{P(t,T)}P(t,T) = \frac{\partial P}{\partial r_t}\sigma(r_t). \tag{2.4}$$

As previously discussed, $\mu_{P(t,T)}$ and $\sigma_{P(t,T)}$ are unknowns for all possible $T > t$, and therefore, we cannot determine a general expression for $P(t,T)$. However, as all bond prices depend on the same single state variable ,r_t, they must all contain one common source of uncertainty, which equals to W_t. Accordingly, we are able to construct a portfolio of two differently maturing bonds, such that the risky term is eliminated, and further use the rule of no arbitrage to specify a single market price of risk common to all derivatives, which solely depend on r_t.

We define the value of a portfolio, consisting of w_i amounts of $P(t,T_i)$ at time t as

$$\Pi_t = \omega_1 P(t,T_1) + \omega_2 P(t,T_2), \qquad \text{where } T_1 \neq T_2, \tag{2.5}$$

satisfying the portfolio SDE

$$\begin{aligned} d\Pi_t =& \Big(\omega_1\mu_{P(t,T_1)}P(t,T_1) + \omega_2\mu_{P(t,T_2)}P(t,T_2)\Big)dt \\ &+ \Big(\omega_1\sigma_{P(t,T_1)}P(t,T_1) + \omega_2\sigma_{P(t,T_2)}P(t,T_2)\Big)dW_t. \end{aligned} \tag{2.6}$$

By a specific and continuously adjusted choice of ω_i, $i = 1, 2$, we are able to eliminate the risky terms, so that[2]

$$\Big(\omega_1\sigma_{P(t,T_1)}P(t,T_1) + \omega_2\sigma_{P(t,T_2)}P(t,T_2)\Big)dW_t = 0, \tag{2.7}$$

thereby reducing the portfolio SDE to

$$d\Pi_t = \big[\omega_1\mu_{P(t,T_1)}p(t,T_1) + \omega_2\mu_{P(t,T_2)}p(t,T_2)\big]dt, \tag{2.8}$$

which is locally deterministic; that is, riskless. By the absence of an arbitrage argument, the resulting portfolio SDE yields no more than the riskless rate, r_t. Thus,

$$\omega_1\Big(\mu_{P(t,T_1)} - r_t\Big)P(t,T_1) + \omega_2\Big(\mu_{P(t,T_2)} - r_t\Big)P(t,T_2) = 0. \tag{2.9}$$

Inserting the chosen weights, and rearranging terms, yields

$$\lambda(r_t) \equiv \frac{\mu_{P(t,T_1)} - r_t}{\sigma_{P(t,T_1)}} = \frac{\mu_{P(t,T_2)} - r_t}{\sigma_{P(t,T_2)}}, \tag{2.10}$$

which we define as the "market price of risk" (preference) parameter. We note that $\lambda(r_t)$ may depend on r_t through $\mu_{P(t,T_i)}$ and $\sigma_{P(t,T_i)}$, respectively, but is not in anyway dependent on the nature of $P(t,T_i)$ itself. That is, the particular choice of the two T_is plays no role in its determination, and generally, $\lambda(r_t)$ is identical across all derivatives exhibiting sole r_t-dependency. A slight re-arrangement of eq (2.10) yields

$$\mu_{P(t,T_i)} = r_t + \lambda(r_t)\sigma_{P(t,T_i)}, \tag{2.11}$$

[2] Specifically $\omega_1 = -\sigma_{P(t,T_2)}P(t,T_2)$, $\omega_2 = \sigma_{P(t,T_1)}P(t,T_1)$

known as the APT restriction. We realize that by eq (2.11), the problem of solving $\mu_{P(t,T)}$ at all times T, is reduced to specifying a single parameter; namely $\lambda(r_t)$. Further, the APT restriction may be written as[3]

$$\begin{aligned}\mu_{P(t,T_i)}P(t,T) &= r_t P(t,T) + \lambda(r_t)\sigma_{P(t,T_i)}P(t,T)\\ &= r_t P(t,T) + \frac{\partial P}{\partial r}\lambda(r_t)\sigma_{r_t}.\end{aligned} \tag{2.12}$$

Equating this with the right hand side of eq (2.3), the mean-drift derivation by Itô's lemma and rearranging terms, we arrive at the fundamental PDE for the bond price

$$\frac{\partial P}{\partial r_t}\big(\mu(r_t) - \lambda(r_t)\sigma(r_t)\big) + \frac{\partial P}{\partial t} + \frac{1}{2}\frac{\partial^2 P}{\partial r_t^2}\sigma_r^2 = r_t P, \tag{2.13}$$

with the obvious boundary condition $P(T,T) = 1$. In summary, by the APT, we have ascertained that all derivatives, solely dependent on r_t, satisfy the fundamental PDE in combination with some asset-specific boundary condition(s). We will later use the developed PDE approach, when considering the derivation of the Vasicek- and its Hull-White descendant models, in chapter 5.

2.1.1 The Feynman-Kac Formula - an Equivalent Representation

An equivalent representation of the solution to the PDE in eq (2.13) is the so-called FEYK formula.[4]

$$P(t,T) = E_t^Q\left[e^{-\int_t^T r_s ds}P(T,T)\right], \tag{2.14}$$

where r_t follows the drift-adjusted Itô process

$$dr_t = [\mu(r_t) - \lambda(r_t)\sigma(r_t)]\,dt + \sigma(r_t)dW_t^Q. \tag{2.15}$$

Here, W_t^Q is a Brownian motion[5] under the Q-measure also known as the "traditional risk-neutral measure".[6] The FEYK formula establishes a link between the PDE and stochastic processes, and offers a general method of solving PDEs by simulating random paths of the underlying stochastic process. At the same time the FEYK provides a lot of intuition on arbitrage-free pricing in term structure models. The current price of a derivative $f(r_t)$ may be obtained by simulating the short rate in a risk-neutral world, calculate its payoff at time T and discount back, by the average short rate. Since future short rates are stochastic, i.e. unknown at time t, the discounting lies in the expectation operator. The technique is known as "traditional risk-neutral valuation", and for more detail, we refer section 8.1, where a Monte Carlo simulation setup for numerical pricing, is implemented for each of the Hull-White Vasicek frameworks.

2.2 The Martingale Approach

A more recent strategy, the martingale approach, uses the theory of martingales to establish arbitrage-free prices. The underlying arguments and derivations will not be considered here.[7] Instead, we take a heuristic approach seeking a **basic** understanding of the involved concepts simply to allow us to apply the forward-risk adjusted terminal measure, when pricing selected options via the standard vanilla models described in sections 3 and 4 and when pricing options on zero-coupon bonds under the Hull-White Vasicek frameworks reviewed in section 5.4. To this end, by an excusable lack of comprehensive theoretical details we briefly sketch the parts needed for our purpose.

[3] Multiply by P(t,T) and inserting the expression of $\sigma_{P(t,T)}P(t,T)$ from eq (2.4).

[4] Refer to [5, p. 245] for the general exposition of the FEYK formula or directly to [6].

[5] [4, Sec. 1.1].

[6] We will revert to a closer definition on the notion of different measures in section 2.2

[7] We refer to the original text by [7].

2.2.1 Martingales and the Change of Measure Technique

As established in eq (2.14), the discounting is located within the expectation operator, under the traditional risk neutral Q-measure, as the value of an arbitrary r_t-dependent derivative V_t, is given by the product of two inter-dependent random variables. What we would like to develop now is the ability to price V_t in the following way

$$V_t = P(t,T)E_t^{Q^T}\left(\frac{V_T}{P(T,T)}\right), \tag{2.16}$$

in which the expectation is now taken under the forward-risk adjusted terminal measure. In our effort to understand the approach we sketch the properties of a martingale.[8] A martingale is a zero-drift stochastic process and a given derivative, f_t, follows a martingale if the process takes the form

$$df_t = \sigma_{f_t} dW_t, \tag{2.17}$$

where dW_t is a Brownian motion,[9] and hence, the process is normally distributed with a zero mean. Accordingly, a martingale holds the important feature that its expected future value at any time t equals its current value. That is

$$E(f_T) = f_t. \tag{2.18}$$

This is a decisive factor and the clue behind the martingale approach. If we are able to transform the price process of a given derivative into a martingale then by eq (2.18), it becomes trivial to determine its price.

To transform price processes into martingales we need a set of tools - namely a change of probability measure technique and the declaration of a numeraire security. Considering the former first, it should be noted, that we during the previous introduction of the FEYK formula, implicitly used a change of probability measure in calculating expected values under Q using a drift-adjusted process of the short rate. In general, we may use "Girsanovs Transformation theorem"[10] to define "equivalent probability measures"[11] so that

$$W_t^Q = W_t^P + \int_0^t \lambda(r_s)ds, \tag{2.19}$$

or in differential form

$$dW_t^Q = dW_t^P + \lambda(r_t)dt. \tag{2.20}$$

If we consider the price of a derivative, V_t, which follows an Itô process similar in form to eq (2.2), and further recall our derived definition of $\lambda(r_t)$ from eq (2.11) we may write

$$\begin{aligned} dV_t &= \mu_{V_t} V_t dt + \sigma_{V_t}(t) V_t dW_t^P && (2.21)\\ &= \mu_{V_t} V_t dt + \sigma_{V_t}(t) V_t \Big(dW_t^Q - \lambda(r_t)dt\Big) \\ &= \Big(\mu_{V_t} - \lambda(r_t)\sigma_{V_t}(t)\Big) V_t dt + \sigma_{V_t}(t) V_t dW_t^Q \\ &= r_t V_t dt + \sigma_{V_t}(t) V_t dW_t^Q, && (2.22) \end{aligned}$$

thereby recovering the traditional risk-free world from before. Accordingly, we realize that under different assumptions about $\lambda(r_t)$, we can change the probability measure, i.e. the drift of the

[8] [4, p. 5] provides a rigor definition of Martingales.
[9] [8, sec. 4.3]
[10] Girsanovs Transformation theorem [4, p. 12].
[11] Equivalent probability measures and the Radon-Nikodym Theorem [4, p. 8-11].

process, however, noting that its volatility remains the same. The change of probability measure from P to Q simply corresponds to a displacement of the original Brownian motion by $-\int_0^t \lambda(r_s)ds$. Under Q, the probability space is thus the same, and only the relative likelihood of each path has changed.[12] Hence, generating paths of the underlying process using a drift-adjusted risk neutral SDE in combination with discounting of payoffs by the stochastic risk-free rate will provide the same price as obtained in terms of real world probabilities under P because the two elements are offset. Consequently, we are left with a simpler method of calculating prices because the task of estimating $\lambda(r_t)$ drops out of the equation.[13]

2.2.2 Numeraires and the Equivalent Martingale Measure

Next, we turn to consider the declaration of a suitable numeraire security. A deflated price process is defined by[14]

$$F_t = \frac{V_t}{P(t,T)}, \tag{2.23}$$

where $P(t,T)$ is the numeraire (security) for $t \in [0,T]$ and F_t is the price of V_t in units of the T-maturing bond price. When arbitrage opportunities are absent, the "equivalent martingale measure result" [EMM][15] shows that replacing $\lambda(r_t)$ with the volatility of $P(t,T)$, $\sigma_{P(t,T)}$, F_t is a martingale for *all* security prices V_t.

To show this general result, we restate the two processes V_t and $P(t,T)$ under Q, inspired by ref [10, chpt. 3], and apply Itô's lemma to form the process followed by F_t

$$dV_t = r_t V_t dt + \sigma_{V_t}(t) V_t dW_t^Q \tag{2.24}$$

$$dP(t,T) = r_t P(t,T)dt + \sigma_{P(t,T)}(t)P(t,T)dW_t^Q. \tag{2.25}$$

To ease the calculation, we redefine the two new processes as $\ln(x)$, recognizing that $\frac{d\ln(x)}{dx} = \frac{1}{x}$, $\frac{d^2\ln(x)}{dx^2}$=-x^{-2}. This yields

$$d\ln(V_t) = \Big(r_t - \frac{\sigma_{V_t}(t)^2}{2}\Big)dt + \sigma_{V_t}(t)dW_t^Q \tag{2.26}$$

$$d\ln(P(t,T)) = \Big(r_t - \frac{\sigma_{P(t,T)}(t)^2}{2}\Big)dt + \sigma_{P(t,T)}(t)dW_t^Q. \tag{2.27}$$

So that $d\ln\big(\frac{V_t}{P(t,T)}\big)$ solves as

$$d\ln(V_t) - d\ln\big(P(t,T)\big) = \frac{\big(\sigma_{P(t,T)}(t)^2 - \sigma_{V_t}(t)^2\big)}{2}dt + (\sigma_{V_t} - \sigma_{P(t,T)})dW_t^Q. \tag{2.28}$$

To back-transform into level form, $d\big(\frac{V_t}{P(t,T)}\big)$, we define the process $F_t = e^{\ln(F_t)}$ and apply Itô's lemma with $\frac{de^x}{dx} = \frac{d^2e^x}{dx^2}$=$e^x$ such that

$$\begin{aligned} dF_t &= \left(\frac{\big(\sigma_{P(t,T)}(t)^2 - \sigma_{V_t}(t)^2\big)}{2} + \frac{\big(\sigma_{P(t,T)}(t) - \sigma_{V_t}(t)\big)^2}{2}\right)F_t dt + \big(\sigma_{V_t}(t) - \sigma_{P(t,T)}(t)\big)\,F_t dW_t^Q \\ &= \sigma_{P(t,T)}(t)\,\big(\sigma_{P(t,T)}(t) - \sigma_{V_t}(t)\big)\,F_t dt + \big(\sigma_{V_t}(t) - \sigma_{P(t,T)}(t)\big)\,F_t dW_t^Q. \end{aligned} \tag{2.29}$$

[12] An excellent example is provided by ref [5, p. 246-247].

[13] We remind, that even though prices may be evaluated under Q, the underlying process of the short rate is displaced. Consequently, if we try to evaluate in level form, the Q-generated process of the short rate itself, e.g. as means of estimating future Libor rates or calculating future Cash flow at risk numbers, this generally provides biased (wrong) results. This is apparent by examining ref [5, p. 246-247], and particularly figure A.1.

[14] [4, chpt. 1].

[15] [9, chpt. 25] and [5, chpt. 4.2].

Furthermore, we define a new probability measure, Q^t, as

$$W_t^{Q^T} = W_t^Q - \int_0^t \sigma_{P(t,T)}(t)ds, \qquad t \in [0,T] \tag{2.30}$$

where the right hand side is a Brownian motion under Q^T, also known as the forward-risk adjusted terminal measure.[16] By inserting the differential form of eq (2.30) into (2.29), we obtain[17]

$$\begin{aligned} dF_t = \; & \sigma_{P(t,T)}(t)\left(\sigma_{P(t,T)}(t) - \sigma_{V_t}(t)\right)F_t dt + \left(\sigma_{V_t}(t) - \sigma_{P(t,T)}(t)\right)F_t\left(dW_t^{Q^T} + \sigma_{P(t,T)}(t)dt\right) \\ = \; & \left(\sigma_{V_t}(t) - \sigma_{P(t,T)}(t)\right)F_t dW_t^{Q^T}. \end{aligned} \tag{2.31}$$

Therefore, according to eq (2.31), F_t is a martingale under Q^T. This is an extremely useful result as, by to the martingale property (2.18), we have

$$F_t = E_t^{Q^T}(F_T), \tag{2.32}$$

which allows us to write

$$\begin{aligned} V_t &= P(t,T)E_t^{Q^T}\left(\frac{F_T}{P(T,T)}\right) \\ &= P(t,T)E_t^{Q^T}\left(F_T\right). \end{aligned} \tag{2.33}$$

This concludes the EMM result, and as seen, by a right choice of numeraire security, we achieve a simpler way of calculating V_t as the discounting now in placed outside the risk-neutral expectation. The only thing left to determine/make assumptions about is the distribution of the expected future payoff(s) under Q^T.

2.2.3 The Forward Rate - a Martingale under the Q^T-Measure

We can also show that the forward rate, $f(t,T)$, is a martingale under Q^T. Amongst others ref [11, p. 8] and ref [12, p. 415] show that the forward rate SDE under Q in a general Heath-Jarrow-Morton model [HJM][18], of which our Vasicek and Hull-White Vasicek frameworks are special cases,[19] is given by

$$df(t,T) = -\sigma(t,T)\sigma_{P(t,T)}dt + \sigma(t,T)dW_t^Q. \tag{2.34}$$

If we change the measure substituting eq (2.30) into (2.34) then

$$\begin{aligned} df(t,T) = \; & -\sigma(t,T)\sigma_{P(t,T)}dt + \sigma(t,T)\left(dW_t^{Q^T} + \sigma_{P(t,T)}dt\right) \\ = \; & \sigma(t,T)dW_t^{Q^T}. \end{aligned} \tag{2.35}$$

Accordingly, $f(t,T)$ is indeed a martingale under Q^T and by the EMM, we have

$$f(t,T) = E_t^{Q^T}(f(T)). \tag{2.36}$$

Literally, this means that when pricing under the forward-risk adjusted T-measure the current forward rate equals the expected future interest rate. This finding will be useful when pricing e.g. different kinds of Caps (Floors) in chapter 4. As a closing remark we note that this result further implies, that forward rates under the P-measure, are biased estimators for future rates. Hence, forecasting future spot rates by the current forward curve, has no or little predictive power, as also noted by [2, p. 9].

[16] We note that there exist a different measure for each payoff date, T.

[17] Note that $\sigma_{P(t,T)}(t)\left(\sigma_{P(t,T)}(t) - \sigma_{V_t}(t)\right)$ may be written as $-\sigma_{P(t,T)}(t)\left(\sigma_{V_t}(t) - \sigma_{P(t,T)}(t)\right)$ so that the two dt-terms cancels out.

[18] A further review of the general HJM framework lies outside the scope of this thesis.

[19] [4, Sec. 4.5.2].

2.2.4 The Money Market Account as Numeraire - an Equivalent Q-Measure

Finally, we consider another choice of numeraire, i.e., the risk-free money market account, which we will define β_t.[20] The money market account accrues at the short rate, r_t, with an initial value of \$1. That is

$$\beta_T = \beta_t \, e^{\int_t^T r_s ds}. \tag{2.37}$$

This corresponds to the following SDE

$$d\beta_t = r_t \beta_t dt, \tag{2.38}$$

where r_t takes the following general stochastic form under P

$$dr_t = \mu_{r_t} dt + \sigma_{r_t} dW_t^P. \tag{2.39}$$

Even though the drift term in eq (2.38) is stochastic, the volatility term is zero. Hence, in using β_t as numeraire the market price of risk, $\lambda(r_t)$, is vanished which corresponds to the change of measure shown in eq (2.22). According to the EMM result, recalling that $\beta_t = 1$ and $\beta_T = e^{\int_t^T r_s ds}$ (eq (2.37)), the price of the previously defined derivative V_t is given by

$$\frac{V_t}{\beta_t} = E_t^{Q^T}\left(\frac{V_T}{\beta_T}\right) \quad \Longleftrightarrow \quad V_t = E_t^Q\left(e^{-\int_t^T r_s ds} \, V_T\right). \tag{2.40}$$

We note that this result is equivalent to the FEYK solution of the PDE from section 2.1.1. Therefore, pricing in a "traditional risk-neutral world" is the equivalent of using the money market account as numeraire. Accordingly, under Q the PDE and Martingale approaches are identical.

[20] [5, p. 56].

Chapter 3

Applied Fixed Income Instruments

This section outlines the applied instruments considered in this thesis. First, liquid traded standard instruments such as Interest Rate Swaps, European Caps (Floors) and Digital Caps (Floors) are considered. Further, the notion of more exotic products such as Range Accrual Swaps are treated. The aim is to provide the needed range of instruments to price selected complex and exotic instruments in chapter 9. Using the definition in [4, p. 209], the treated instruments enables us to price Exotic products belonging to the class of Libor-based instruments such as Range Accruals and generally path-dependent products. Thus, this section provides a layup to pricing of: 1) Exotic Libor-based Swaps where the structured payoffs may be decomposed into standard "vanilla " products using static replication. 2) Extend to more complex pricing problems where generally full term structure models and numerical Monte Carlo methods are mandatory. For obvious reasons, emphasis is put on the latter, whereas the former provides some of the building blocks needed for more complicated structures.

3.1 Fixed for Floating Swaps

An interest rate swap [IRS] is an instrument, where a fixed (floating) rate payment stream is exchanged for a floating (fixed) rate payment stream (e.g. Libor). A firm with liabilities funded at Libor may linearly hedge its position by entering a payer IRS.

Inspired by ref [13, chpt. 2.3], consider the valuation of a **spot starting IRS** settled in arrears. Let $T_1 < ... < T_n$ denote each coupon date, T_0 the start date and t the initial/current time. Furthermore, $Sr(t, T_n)$ denotes the swap rate observed at time t, effective from T_0 to T_n. Accordingly, the fixed leg value is given by

$$\Pi_{fx} = Sr(t, T_n)A(t) \tag{3.1}$$

$$A(t) = \Sigma_{i=1}^{n} \alpha_i P(t, T_i), \tag{3.2}$$

where $\alpha_i = (T_i - T_{i\text{-}1})$ equals the fixed leg year fraction based on a suitable day count convention - often 30/360. Furthermore, the value of the floating leg is given by

$$\Pi_{fl} = \Sigma_{i=1}^{n} \delta_i L_{i\text{-}1}(t) P(t, T_i), \tag{3.3}$$

where $L_{i\text{-}1}(t)$ denotes the forward Libor rate and $\delta_i = (T_i - T_{i\text{-}1})$ the floating leg year fraction based on a suitable day count convention - typically act/360. For simplicity, we define $\tau_i \equiv \alpha_i = \delta_i$ from now on.[1,2] It should also be noted, that the value of the floating leg equivalently may be written

[1]However, in practice this is often not the case; $\alpha_i \neq \delta_i$ due to differences in day count convention and payment frequency.

[2]For more information on day count conventions we refer to e.g. ref [2, chpt. 2.5.1] or `http://en.wikipedia.org/wiki/Day_count_convention` (25.06.2012)

as[3]

$$\begin{aligned}\Pi_{floating} &= P(t,T_t) - P(t,T_n) \\ &= 1 - P(t,T_n).\end{aligned} \tag{3.4}$$

The present value of an IRS is the difference in value between the two legs. That is

$$\Pi_{swap} = \Pi_{fx} - \Pi_{fl}, \tag{3.5}$$

and for a given break-even (mid-market) swap rate, we have

$$\Pi_{fx} = \Pi_{fl}. \tag{3.6}$$

Substituting eq (3.2) and (3.4) into (3.6) and rearranging terms yields the spot starting mid-market swap rate. That is

$$Sr(t,T_n) = \frac{1 - P(t,T_n)}{A(t)}. \tag{3.7}$$

Similarly, consider the valuation of a **forward starting IRS** settled in arrears. Adapting the same notation, where $S \equiv T_0$, $Sr(t;S,T_n)$ now denotes the forward swap rate observed at time t, effective from S to T_n. Accordingly, the value of the floating leg changes to[4]

$$\Pi_{floating} = P(t,S) - P(t,T_n). \tag{3.8}$$

Hence, the forward starting break-even swap rate is

$$Sr(t;S,T_n) = \frac{P(t,S) - P(t,T_n)}{\Sigma_{i=1}^{n} \tau_i P(t,T_i)}, \tag{3.9}$$

which may be rewritten as

$$Sr(t;S,T_n) = \frac{1 - P(S,T_n)}{A(S^*)} \tag{3.10}$$

$$A(S^*) = \Sigma_{i=1}^{n} \tau_i P(S,T_i). \tag{3.11}$$

Thus, a forward starting swap rate is given by the same expression as the spot swap rate conditional on replacement of discount factors by the designated forward discount factors.

3.2 European Caps and Floors

A Cap (Floor)[5] is an instrument designed to protect against rising (falling) interest rates yet allowing the holder to benefit from the opposite scenario of low (high) rates. Formally, a Cap (Floor) is a strip of European Call (put) options on single forward rates also called Caplets (Floorlets) and thus have a close relation to Forward Rate Agreements [FRAs]. Each Caplet pays, at time T_i, the difference between a reference index (such as Libor) and a pre-agreed strike rate k. Using the positive-part operator, this can be expressed as

$$\tau_i \big(L_{i\text{-}1}(T_{i\text{-}1}) - k\big)^+, \tag{3.12}$$

per unit notional amount, reset at time $T_{i\text{-}1}$ and settled at time T_i. Similarly a Floorlet, pays

$$\tau_i \left(k - L_{i\text{-}1}(T_{i\text{-}1})\right)^+, \tag{3.13}$$

[3] $1 - P(t,T_n)$ equals the value of accrued interest. We refer to ref [14, p. 7] for the formal prove.

[4] Realize, that $P(t,S) - P(t,T_n)$ is the value of the forward-spaced accrued interest.

[5] [1, chpt. 1.6].

with the same notional amount, reset time and settlement time as above. As before, $\tau_i = (T_i - T_{i\text{-}1})$ denotes the compound period according to a pre-specified day count convention - typically act/360.

As formally proved in section 2.2, the forward rate may be modeled as a martingale under the terminal measure, and therefore, the value of the n-period cap (floor) at time t under Q^T is given by

$$\Pi_{Cap}(t) = \sum_{i=1}^{n} P(t,T_i)\,\tau_i\, E^{T_i}\Big(L_{i\text{-}1}(t) - k\Big)^+ \tag{3.14}$$

$$\Pi_{Floor}(t) = \sum_{i=1}^{n} P(t,T_i)\,\tau_i\, E^{T_i}\Big(k - L_{i\text{-}1}(t)\Big)^+. \tag{3.15}$$

In spot start caps (floors) the first caplet (floorlet) is skipped, as for obvious reasons, the initial Libor fixing is known in advance, merely corresponding to the underlying linear FRA.

To summarize, given the uncertainty of future interest rates each contingent T_i-claim may under the T-measure be discounted by the T_i-maturing zero-coupon bond provided that the expected value of the future spot rate is adjusted from $E^{T_i}\big(F(T_{i-1};T_{i-1},T_i)\big)$ to $E^{T_i}\big(L_{i\text{-}1}(t)\big)$.[6]

What remains, in order to solve the positive-part operator, is the distribution of $E^{T_i}\big(L_{i\text{-}1}(t)\big)$, which implies that caps (floors) can be priced using the standard vanilla models described in section 4.

While prices of individual Caplets (Floorlets) are not observed directly, various maturities in Caps (Floors) are liquid traded in the OTC market. Extracted volatility information implied by the vanilla models, can be used as input data, of which interest rate models for the pricing of exotic instruments can be calibrated. We revert to the subject of model calibration in section 7.1.1.

3.2.1 The Put/Call Parity

Caps (Floors) are strongly related to swaps and, and c.f. [15, chpt. 2.2] certain parity-relation exists

$$\Pi_{Cap}(t) - \Pi_{Floor}(t) = \Pi_{Payer\ Swap}(t), \tag{3.16}$$

where $\Pi_{Payer\ Swap}(t)$ is the value of a payer swap with swap rate k, unit nominal amount and equal tenor structure as the Cap (Floor). The Cap (Floor) is said to be at the money [ATM] if, for $S \geq t$ and $t = 0$

$$k = Sr(t;S,T_n) = \frac{P(t,S) - P(t,T_n)}{\Sigma_{i=1}^{n}\tau_i P(t,T_i)}. \tag{3.17}$$

Keeping the previous notation unchanged, we prove the Call-Put parity

$$0 = \Pi_{Cap}(t) - \Pi_{Floor}(t) \tag{3.18}$$

$$= \sum_{i=1}^{n} \tau_i\Big(\big(k - E^{T_i}\big(L_{i\text{-}1}(t)\big)^+ - E^{T_i}\big(L_{i\text{-}1}(t)\big) - k\big)^+\Big)P(t,i)$$

$$= \sum_{i=1}^{n} \tau_i\big(k - E^{T}\big(L_{i\text{-}1}(t)\big)P(t,i). \tag{3.19}$$

As discussed in section 2.2, the expected future spot rate equals the current Libor rate under Q^T. Eq (3.16)-(3.18) show that (3.19) is only true when

$$k \;=\; Sr(t;S,T_n), \tag{3.20}$$

which concludes the formal prove.

[6] $E^{T_i}\big(F(T_{i-1};T_{i-1},T_i)\big) = E^{T_i}\big(F(t;T_{i-1},T_i)\big) = E^{T_i}\big(L_{i\text{-}1}(t)\big)$

3.3 Digital Caps and Floors

Digital Caps (Floors) works like their European relatives, except that the nth Digital Caplet pays at time T_i, a unit amount, if Libor fixes above the strike level. Using the indicator function, $\times 1_{\{\#\}}$, in analogy with ref [4, chpt. 5.9], this can be written as

$$\tau_i \times 1_{\{L_{i\text{-}1}(T_{i\text{-}1})>k\}}, \tag{3.21}$$

reset at time T_{i-1} and settled at time T_i. Similarly, the Digital Floorlet pays

$$\tau_i \times 1_{\{L_{i\text{-}1}(T_{i\text{-}1})<k\}}, \tag{3.22}$$

with similar reset and settlement times. Often, the payoff is set to be a fixed rate of the notional amount. In analogy, the value of the n-period Digital Cap (Floor) at time t under Q^T is given by

$$\Pi_{Cap}(t) = \sum_{i=1}^{n} P(t,T_i)\,\tau_i\,E^{T_i}\big(1_{\{L_{i\text{-}1}(T_t)>k\}}\big) \tag{3.23}$$

$$\Pi_{Floor}(t) = \sum_{i=1}^{n} P(t,T_i)\,\tau_i\,E^{T_i}\big(1_{\{L_{i\text{-}1}(T_t)<k\}}\big). \tag{3.24}$$

Digital Caps (Floors) are not as liquid traded instruments as their European piers, wherefore prices are not readily available in the OTC market. Pricing may be analytically determined, using the standard vanilla models (as discussed in section 4),[7] customized for Digital options directly or approximated via static replication in Cap (call)-spreads. As in essence, Digital Caps (Floors) provides a leveraged bet on the future direction of the underlying interest rate (“money-or-nothing“), the latter approach is market practice, as it captures the inherent volatility “smile-risk”[8] caused by the increased sensitivity to changes in volatility around the strike level.

3.4 Exotic Libor Based Range Accruals

A Libor-based Range Accrual coupon may be defined as a rate, which only accrues when the Libor reference rate fixes inside (outside) a pre-set range. The accrual rates under consideration is fixed, floating or a combination hereof[9] and the basic accrual coupon structure is given by[10]

$$C_{i\text{-}1} \;=\; R_{i\text{-}1}(T_{i\text{-}1}) \times \frac{\sharp\{s \in [T_{i\text{-}1}, T_i] : X_n(s) \in [l,u]\}}{\sharp\{s \in [T_{i\text{-}1}, T_i]\}}, \tag{3.25}$$

where $R_{i\text{-}1}(T_{i\text{-}1})$ denotes the payment rate, $X(s)$ the reference rate on intermediary days s in the interval $[T_{i\text{-}1}, T_i]$ while l, and u are the lower and upper bounds, respectively. Further, $\sharp\{\cdot\}$ is the number of days for which the specified criterion is satisfied. Generally, we consider a composite payment rate consisting of

$$R_n(T_n) \;=\; [\gamma \times L_{i\text{-}1}(T_{i\text{-}1}) + Z\%], \tag{3.26}$$

where Z denotes a fixed rate, $L_{i\text{-}1}(T_{i\text{-}1})$ the usual 3M Libor rate, whereas γ is a participation rate determinant of the leverage in the underlying Libor index. Range Accruals may be decomposed into a combination of simpler Digital and European Caps (Floors), as for example, for a fixed payment rate, a Range Accrual may be recognized as merely a strip of daily Digital Caplets (Floorlets). Consequently, Range Accruals may be priced using standard models (directly or via Call-Spreads) as discussed in section 4. For the payment rate structure, the analytical procedure requires additional setting-up[11]. For that reason, we confine ourselves to pricing via the Hull-White Extended Vasicek Monte Carlo setup,r as outlined in section 9.2.

[7] In combination wih ref [16, chpt. 11.5].

[8] Refer to section 4.4.

[9] But could be also be a CMS rate or a CMS spread.

[10] [1, chpt. 13.13.1] or [4, chpt. 5.13.4].

[11] E.g. see ref [4, sec. 17.5].

Chapter 4

Standard Vanilla Models

In this section we provide a short description of the two standard vanilla models for quotation of interest rate volatility in the financial markets, namely the Black76 and the Normal model. First, we briefly discuss the underlying theoretical assumptions behind the two models. Next, we consider how to interpret the individual volatility measures and, using a small example, we infer market expectations on future rates, comparing the two models. The chapter is concluded by a short description of the inherent problems when considering the observed volatility skew, which provides a layup and argument behind our later implementation of a full term structure model.

Our exposition is kept short because at least the basic properties of Black's model is a part of the standard curriculum of most courses in finance. Our aim is merely to provide an outline of the practical application of the vanilla models as benchmark for quotation of interest rate volatility in the market. For that reason, the underlying mathematical proofs are will not be considered here but can be found in the literature.[1] Furthermore, the derivation of the Black-Scholes-Morton PDE follows closely the general approach as developed in section2.1. Alhough later parts of this thesis only use the notion of basis point volatility [BP-vol] from the Normal model to a limited extent, we remind that, in practice, the concept of BP-vol is maybe even more used than its Black76 [Black-vol] correspondent.

4.1 The Black76 model

The Black76 model is *the* benchmark for quotation of interest rate volatility in the market and the course of success lies primarily in its easy application and use. However, due to the assumption of constant volatility across strike levels underlying the model, it is no longer used for actual pricing but rather serves as a way of calculating market implied volatilities. The Black76 model assumes that the underlying forward rate, $F(t; s, T)$ (such as 3M Libor), follows a geometrical Brownian motion [GBM]. That is, forward rates are log normal distributed, and consequently, never take on negative values.[2] We recall from section 2.2.3 that the forward rate is a martingale under Q^T such that the forward rate process becomes

$$dF_T = \sigma_{BS} F_T dW^T, \tag{4.1}$$

where σ_{BS} denotes the log normal volatility,[3] known as the Black76 volatility or simply [Black-vol]. If we define a new process, $\ln(F_{s_i})$, and then apply Itô's lemma, we obtain

$$d\ln(F_{s_i}) = -\frac{\sigma_{BS}^2}{2}dt + \sigma_{BS} dW_t^T. \tag{4.2}$$

[1]For a derivation of the Black-Scholes-Morton formula, see e.g. ref [4, section 1.9]

[2]For an outline of the log normal probability density function we refer to: `http://en.wikipedia.org/wiki/Log-normal_distribution` (20.07.2012).

[3]I.e. the volatility of the percentage return of the forward rate.

As μ and σ_{BS} are both constants $\ln(F_{s_i})$ follows a generalized Brownian motion,[4] i.e., is normally distributed. That is

$$\ln(F_T) - \ln(F_{T_0}) \sim \phi\left[\left(-\frac{\sigma_{BS}^2}{2}\right)T\,;\,\sigma_{BS}\sqrt{T}\right] \tag{4.3}$$

$$\ln(F_T) \sim \phi\left[\ln(F_{T_0}) - \frac{\sigma_{BS}^2}{2}T\,;\,\sigma_{BS}\sqrt{T}\right]. \tag{4.4}$$

Hence, the percentage return of the logarithm of the forward rate, $\ln(F_{s_i})$, is normally distributed with a constant drift rate of -$\frac{\sigma^2}{2}T_i$ volatility parameter equal to $\sigma_{T_i}\sqrt{T_i}$ for each $i = \{1, ..., n\}$ across all strike levels. I.e. the process can be modeled as

$$\ln(f_T) = \ln(f_{T_0}) + \sigma dW_T - \frac{\sigma^2}{2}T. \tag{4.5}$$

The level process of f_{T_i} is obtained by reversing the logarithms on both sides

$$f_T = f_{T_0}e^{\sigma dW_T - \frac{\sigma^2}{2}T}. \tag{4.6}$$

Following Black,[5] the price of the ith Caplet under the log normal assumption may be written as

$$Cpl(t) = P(t,i)\,N\,\delta_i\left(f(t;T_{i-1},T_i)\Phi\big(d_1(t,i)\big) - k\Phi\big(d_2(t,i)\big)\right), \tag{4.7}$$

where $t \leq T_0$ and $\Phi(x)$ denotes the standard normal distribution, while $d_{(1,2)}$ equals

$$d_{1,2}(t,i) = \frac{\ln\left(\frac{f(t;T_{i-1},T_i)}{k}\right) \pm \frac{1}{2}\sigma(T_i)^2(T_{i-1}-t)}{\sigma(T_i)\sqrt{T_{i-1}-t}}. \tag{4.8}$$

Note that eq (4.7) has to be applied for each individual caplet $i = \{1, ..., n\}$ and that $\sigma(T_i)$ denotes the volatility of the ith caplet - in the market known as the Spot volatilities. However, market participants usually quote in terms of Flat volatilities. Here, $\sigma(T_i)$ is substituted by a fixed $\sigma(T_n)$, where $\sigma(T_n)$ denotes the "average" spot volatility for all Caplets up to and including the maturity for the period in question. Accordingly, Black76 for the T_n-maturing Cap is given by

$$Cp(t) = \sum_{i=1}^{n} P(t,i)\,N\,\delta_i\left(f(t;T_{i-1},T_i)\Phi\big(d_1(t,i)\big) - k\Phi\big(d_2(t,i)\big)\right), \tag{4.9}$$

where $t \leq T_0$. Apart from the replacement in eq (4.8) of $\sigma(T_i)$ with $\sigma(T_n)$, $d_{(1,2)}$ and $\Phi(x)$ are as defined above.

4.2 The Normal Model

Another widely used benchmark is the Normal model. The Normal model assumes that the underlying forward rate F_{T_i} follows a generalized Brownian motion, i.e., the forward rate is normally distributed. Accordingly, the rates in this environment may turn negative. Although in recent years, there have been several examples of negative rates in the short end of the curve in the market, this is seen as drawback of the model.[6] The process followed by $f(T_i)$ under Q^T is

$$df_{T_i} = \sigma dW^T, \tag{4.10}$$

[4] [9, p. 267].

[5] We refer to [9, p. 310] for the formal prove of (4.7) and (4.8).

[6] As per Jul-2012 e.g. Germany, Finland and Denmark have sold 2Y notes at negative yields. Ref Bloomberg™.

where σ is the normal volatility, i.e., the volatility measured in level form, also known as the BP-vol or the basis point volatility. The level process of f_{T_i} is straight forward

$$f_T = f_{T_0} + \sigma T. \tag{4.11}$$

Again, both μ and σ are constants, and therefore, the forward rates follow a generalized Brownian motion that was previously found to be normally distributed

$$f_T \sim \phi\left[f_{T_0},\, \sigma\sqrt{T}\right]. \tag{4.12}$$

According to eq (4.12), changes to the forward rate (measured in basis points) are normally distributed with a constant drift rate of zero and a constant volatility of $\sigma\sqrt{T}$. Similar to Black-vol, the BP-vol in the Black76 model is equal to $\sigma_{T_i}\sqrt{T_i}$ for each $i = \{1, ..., n\}$ across all strike levels. Thus, the process can be modeled as

$$f_T = f_{T_0} + \sigma dW_T. \tag{4.13}$$

Given the assumption of a normally distributed forward rate, the payoff, $\big(f(t; T_{i-1}, T_i) - k\big)^+$, and accordingly, the price of a Caplet can be priced as in eq (4.7), where eq (4.8) is substituted by

$$Cpl(t) = P(t,i)\, N\, \delta_i \left(f(t; T_{i-1}, T_i)\Phi\big(d_1(t,i)\big) - k\Phi\big(d_2(t,i)\big)\right) \tag{4.14}$$

where $t \leq T_0$, $\Phi(x)$ denotes the standard normal distribution and $d_{(1,2)}$ equals

$$d_{1,2}(t,i) = \frac{f(t; T_{i-1}, T_i) \pm K}{\sigma(T_i)\sqrt{T_{i-1} - t}}. \tag{4.15}$$

As with the Black76 model, the method has to be applied for each individual caplet $i = \{1, ..., n\}$ and that $\sigma(T_i)$ denotes the *volatility* of the $i'th$ caplet - in the market known as the *Spot volatilities.* However market participants usually quote in terms of *Flat volatilities* where $\sigma(T_i)$ is substituted by a fixed $\sigma(T_n)$ where $\sigma(T_n)$ denotes the "average" spot volatility for all Caplets up until maturity for the period in question.

Accordingly, Black76 for the T_n maturing Cap is

$$Cp(t) = \sum_{i=1}^{n} P(t,i)\, N\, \delta_i \left(f(t; T_{i-1}, T_i)\Phi\big(d_1(t,i)\big) - k\Phi\big(d_2(t,i)\big)\right) \tag{4.16}$$

$$d_{1,2}(t,i) = \frac{ln\Big(\frac{f(t;T_{i-1},T_i)}{k}\Big) \pm \frac{1}{2}\sigma(T_n)^2(T_{i-1} - t)}{\sigma(T_n)\sqrt{T_{i-1} - t}} \tag{4.17}$$

4.3 Interpretation of the Volatility Specification

In summary, the Black-vol is log normal and defined as the "standard deviation of the logarithm of percentage returns of the forward rate"[7]. On the other hand, the BP-vol is normal and is simply the "standard deviation of the forward rate" in nominal (bp) terms.[8] Both measures are quoted per annum, $\sigma_{BS/BP}\sqrt{\frac{T_n}{360}}$, and market participants usually quote in terms of flat volatilities regarded as an average volatility measure for the period in question.[9] To get an idea of the interpretation of the Black-vol, we consider the following instructive example.

[7] [15, chpt. 2.1].

[8] [15, chpt. 3.1].

[9] We note, however, that traders often in like to look at the spot volatilities of the individual Caplets (Floorlets) as it allows them to locate potential relative value trades.

Black76-vol in practice US Cap, 1x3, K=0.55% at the money forward [ATMF], σ_{BS}: 62.42
From eq (4.4), the distribution of the log of the forward rate is given as

$$\ln(f_T) \sim \phi\Big[\ln(0.55) - \frac{(62.42)^2}{2}\,; 62.42\Big] \tag{4.18}$$

$$\ln(f_T) \sim \phi\big[\text{-}5.3978\,; 62.42\big] \tag{4.19}$$

Accordingly, a 95-confidence interval around the ATMF rate can be established so that

$$\ln(f_T)|^{95} \in \big[\text{-}1.96\times 62.42 - 5.3978\,,\ 1.96\times 62.42 - 5.3978\big] \tag{4.20}$$

$$f_T|^{95} \in \Big[e^{\text{-}1.96\times 62.42 - 5.3978}\,,\ e^{1.96\times 62.42 - 5.3978}\Big] \tag{4.21}$$

$$f_T|^{95} \in \big[0.13\,,\ 1.54\big], \tag{4.22}$$

Because the forward rate is a martingale under Q^T, we have[10]

$$f_0 = f_T = e^{\text{-}5.3978 + \frac{1}{2}(62.42)^2} = 0.55, \tag{4.23}$$

Accordingly, (4.22) shows the expected 95% maximum deviation from the ATMF over the life of the Cap.[11] Note that the confidence band is right tailed due to the underlying log normal assumption, a finding which is particularly expressed when encountering very low interest rates. In conclusion, the Black-vol contains information about the uncertainty of future rates in terms of percentage changes of the logarithm of the forward rate rather than changes in level form. As a consequence, unchanged expectations on future interest rate volatility, under various nominal levels of interest rates, induce different levels of Black-vol(!).[12] Thus, one has to be cautious when interpreting (changes in) the Black-vol since increasing Black-vols might simply reflect a downwards correction in ATMF rather than an up-shift in the underlying risk perception of future interest rates. For this reason, the Black-vol can be a little tricky to communicate in a meaningful way and very often market participants communicate interest volatility in terms of BP-vol.[13] This is convenient as the BP-vol is independent of ATMF and directly reflects the uncertainty in level form basis points. In passing, we note that calculations similar to those above can be made also in the Normal model.

4.4 The Volatility Term Structure and Skew

According to eq (4.7), both the vanilla models have different (Flat) volatilities for each maturity, T_i for $i = \{1, ..., n\}$, also known as the term structure of volatilities. Typically, the market exhibits a significant Cap-hump, where the ultra-short end trades at very low levels, closely tied to the central bank leading rate. The front-mid segments (∼1-3Y) trades at significantly higher levels since most of the trading takes place in area, whereas on longer maturities the volatility expectations average out. However, there is no general agreement on the reason for the existence of the hump.[14] Figure 4.1(a) illustrates the current ATM volatility term structure for the market data, which will be presented in greater detail in section 6.1. Thus, the two vanilla models have no problems in fitting the ATM volatility term structure because they take as input the current volatility curve.

However, for a specific maturity, both vanilla models take a fixed Black-vol, i.e., it is independent of the strike. As displayed in figure 4.1(b), this feature is far from what is observed in the market.

[10] Using the mean of the log normal distribution: $e^{\mu + \frac{1}{2}\sigma^2}$.

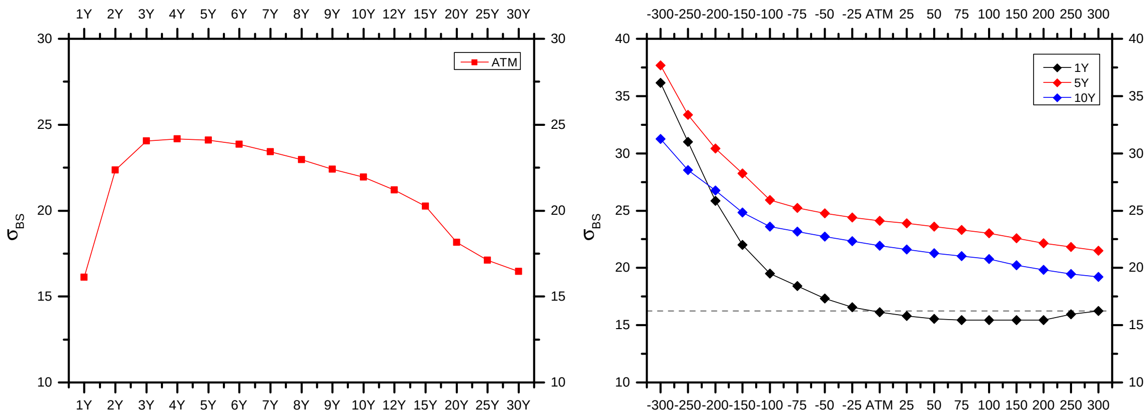
[11] Note that as we define by σ_{BS} the flat volatility, the confidence band represents an average maximum deviation. The interpretation is more clear for a single Caplet (Floorlet) or e.g. a Swaption as here the measure denotes the expected confidence interval for the underlying at maturity of the option.

[12] For example, a decrease in ATMF means an increase in Black-vol all other things equal, which can easily be verified by substituting into eq (4.8)-(4.9) solving for the implied Black-vol at different ATMF levels.

[13] Own experience from Nordea Markets.

[14] [9, p. 622]

Figure 4.1: The Volatility Term Structure and Skew. In figure (b) the gray dashed line represents the fixed 1Y ATM Black-vol feasible as input in Black's model.



(a) Tenor

(b) Relative Strike (K)

Figure 4.1(b) is thus produced by backing out the implied Black-vol via inserting the market price into eq (4.7). This is a major drawback of the vanilla models, and generally, a full term structure model is needed to be able to fit the volatility skew.[15]

As a closing remark we note that commonly, Caps (Floors) are said to be quoted in terms of their implied volatilities. However, such arguments are imprecise. In fact, the converse is true since market participants use the Black76 model (and often the Normal model) to infer implied volatilities from current market prices and not the other way around. Trading books are measured in terms of cash amounts, and thus, traders always quote Caps (Floors) in terms of prices and not volatilities.[16] A trader would say "this price translate into a Black-vol/BP-vol of X" and never use the volatility as a means of quoting the instrument. This claim is further supported by the fact that, apart form the actual Black-vol, parties need to agree on a yield curve before a price can be established in Black's model.

Both vanilla models have been implemented in VBA. The corresponding source code can be found in Appendix C.2 and should be self-explanatory. In this context, we highlight the simple binary search algorithm developed to back out implied Black volatilities when considering the Hull-White model later on. This algorithm can be found in Appendix C.4.11.

[15] We note that several later extensions to Black's model have much better skew-fitting capabilities, such as e.g. the Heston model.

[16] Discussions with volatility traders at Nordea Markets.

Chapter 5

Term Structure Models

In chapter 2.1, we developed a general framework within which all one-factor derivatives must fit. In this chapter, we turn to consider a specific class of one-factor frameworks known as Mean-Reverting Gaussian Short Rate Models using the derived results from chapter 2.1. First, we review the Classical Equilibrium Vasicek model and then turn to its arbitrage-free descendants; namely the Hull-White Vasicek [HWV] and the Hull-White Extended Vasicek models [HWExtV]. Different designations for various augmented versions of the original Vasicek model exists in the literature. Without a further discussion of their editorial origin, we define by the HWV model a setup with a deterministic time-varying mean-drift, while the HWExtV model introduces additionally a deterministic time-varying volatility parameter.

This chapter provides a detailed exposition of the principles and techniques of each of the closely related models, to gain a thorough understanding of the basic theoretical foundations of one-factor term-structure modeling. Further and foremost, our aim is to prepare for a practical implementation of the models to price interest rate derivatives via analytical and numerical methods. It is our intention that rigor and in-depth understanding of the financial and applicational context should not be sacrificed for the purpose of mathematical detail. To this end, full derivations for the Vasicek model is shown, whereas the reader is redirected to the vast amount of literature, for explicit derivations on a few selected expressions under the two Hull-White Vasicek frameworks.

5.1 The Vasicek Model

The Vasicek model (Vasicek77) was introduced by Oldrich Vasicek in 1977 [17]. The model suggests that the short rate follows a mean-reverting Gaussian process (originating from physics - known as a Ornstein-Uhlenbeck process) with dynamics given by the following SDE

$$dr_t = \kappa(\mu - r_t)dt + \sigma dW_t, \tag{5.1}$$

where κ, μ and σ are constants and with μ as the unconditional mean under P. The market price of risk is assumed to be constant; that is $\lambda(r_t) = \lambda$ for all t.

5.1.1 Definition of the model

Before a further evaluation of the model, we note that as this thesis regards development of prices only, modeling dynamics under P is inefficient. With reference to chapter 2.2, we deploy a convenient change of probability measure from P to the risk-neutral measure Q, whereby we defeat the challenge of estimating λ. Accordingly, following ref [3, chpt. 4], we restate the SDE in eq (5.1) under Q with the necessary adjustment of the drift term

$$dr_t = \kappa(\theta - r_t)dt + \sigma dW_t^Q, \tag{5.2}$$

where $\theta = \mu - \lambda\sigma/\kappa$ represents the risk neutral mean under Q, while κ measures the speed of mean reversion, and σ the instantaneous volatility of the short rate. The mean reverting dynamics is apparent from the first term; when $r_t > \theta$, the drift is negative, whereas for $r_t < \theta$, the drift becomes positive. Since the drift-term always reverts toward θ, it may be regarded as an "equilibrium" or long-run risk neutral mean of the short rate. The magnitude of κ adjusts the power of the "mean reverting pull" for which $\kappa \to 0^+$ corresponds to an elimination of the mean reverting dynamics, i.e. it leads to higher *implicit* volatility in r_t. The second term controls the *direct* instantaneous volatility of the short rate. For $\sigma \to 0^+$, the volatility in r_t vanishes, and eq (5.2) reduces to

$$dr_t = \kappa\left(\theta - r_t\right) dt, \tag{5.3}$$

where $r(0) = r_0$. Further this may be rewritten as

$$\frac{dr_t}{(\theta - r_t)} = \kappa dt.$$

Integration on both sides yields

$$\begin{aligned} \kappa \int_0^t dt &= \int_{r_0}^{r_t} \frac{dr}{(\theta - r_t)} \qquad \Longrightarrow \\ \kappa t &= -\ln(\theta - r_t)|_{r_0}^{r_t} \quad \Longrightarrow \end{aligned} \tag{5.4}$$

$$-\kappa t = \ln\left(\frac{\theta - r_t}{\theta - r_0}\right). \tag{5.5}$$

Taking the exponential of both sides and rearranging terms gives

$$r_t = \theta + (r_0 - \theta)\, e^{-\kappa t}. \tag{5.6}$$

It follows that $r_t \to \theta$, for $t \to \infty^+$. The mean drift takes place from above when $r_0 > \theta$ and conversely when $r_0 < \theta$.

5.1.2 Moments and Distribution of the Short Rate

To find the distribution of r_t, we introduce the following function, inspired by ref [18, eq. 7]

$$g(r_t) = r_t e^{\kappa t}. \tag{5.7}$$

By applying Itô's Lemma to $g(r_t)$, we get

$$dg = \left(\frac{\partial g}{\partial r_t}\mu + \frac{\partial g}{\partial t} + \frac{1}{2}\frac{\partial^2 g}{\partial r_t^2}\sigma^2\right) dt + \frac{\partial g}{\partial r_t}\sigma dW_t^Q. \tag{5.8}$$

As $\frac{\partial g}{\partial r_t} = e^{\kappa t}$ and $\frac{\partial g}{\partial t} = \kappa r_t e^{\kappa t}$, eq (5.8) can be rewritten so that

$$\begin{aligned} dg &= \left(e^{\kappa t}\mu + \kappa r_t e^{\kappa t}\right) dt + e^{\kappa t}\sigma dW_t^Q \\ &= \left(e^{\kappa t}\kappa\left(\theta - r_t\right) + \kappa r_t e^{\kappa t}\right) dt + e^{\kappa t}\sigma dW_t^Q \\ &= \theta\kappa e^{\kappa t} dt + e^{\kappa t}\sigma dW_t^Q. \end{aligned} \tag{5.9}$$

As the r.h.s. of eq (5.9) is independent of $g(r_t)$, then integration gives

$$\begin{aligned} \int_0^t dg &= \theta\kappa \int_0^t e^{\kappa s} ds + \sigma \int_0^t e^{\kappa s} dW_s^Q, \qquad \Longrightarrow \\ g(r_t) - g(r_0) &= \theta\kappa\left(e^{\kappa t}\frac{1}{\kappa} - \frac{1}{\kappa}\right) + \sigma \int_0^t e^{\kappa s} dW_s^Q, \\ &= \theta\left(e^{\kappa t} - 1\right) + \sigma \int_0^t e^{\kappa s} dW_s^Q. \end{aligned} \tag{5.10}$$

As $g(r_t)=r_t e^{\kappa t}$, $r_t=g(r_t)e^{-\kappa t}$ and thus $r_0=g(r_0)$, the solution for r_t is obtained by multiplying by $e^{-\kappa t}$ on both sides of eq (5.10)

$$r_t = r_0 e^{-\kappa t} + \theta(1-e^{-\kappa t}) + \sigma \int_0^t e^{-\kappa(t-s)} dW_s^Q. \tag{5.11}$$

Now, consider the last term in eq (5.11). From stochastic calculus, it can be shown that a stochastic integral of a deterministic function $f(s)$ with respect to a Brownian motion follows a Gaussian distribution with a zero mean and a variance of $\int_0^t f^2(s)ds$,[1] so that

$$\begin{aligned} \mathrm{Var}(r_t) &= \mathrm{Var}\left(e^{-\kappa t}\sigma \int_0^t e^{\kappa s} dW_s^Q\right) \\ &= e^{-2\kappa t}\sigma^2 \int_0^t e^{2\kappa s} ds \\ &= \frac{\sigma^2}{2\kappa}\left(1-e^{-2\kappa t}\right). \end{aligned} \tag{5.12}$$

This means that for $t \to \infty^+$, the variance of the short rate exponentially decays toward $\frac{\sigma_r^2}{2\kappa}$, i.e. longer rates are more volatile than shorter rates. Moreover, in the limit $\kappa \to 0^+$ by applying l'Hôpital's rule, we obtain

$$\begin{aligned} &\lim_{\kappa\to 0^+} \sigma^2 \frac{\left(1-e^{-2\kappa t}\right)}{2\kappa} \quad \left[\frac{0}{0}\right] \\ &= \lim_{\kappa \to 0^+} \sigma^2\left(te^{-2\kappa t}\right) = \sigma^2 t. \end{aligned} \tag{5.13}$$

Thus, elimination of the mean-reversion parameter would imply the variance of r_t to become infinite with the same properties as a standard Brownian motion. Accordingly, we have shown that the mean-reversion parameter *limits* the variance of the short rate in the Vasicek model.

To this end, as the mean of the stochastic integral in eq (5.11) is zero, r_t follows a Gaussian distribution with the following properties:

1. $r_t \sim \mathcal{N}$, normally distributed under Q
2. $r_0 e^{-\kappa t} + \theta(1-e^{-\kappa t})$
3. $\mathrm{Var}(r_t)=\frac{\sigma^2}{2\kappa}\left(1-e^{-2\kappa t}\right)$

From 2), we see that in the $\lim_{t\to\infty^+} E(r_t)=\theta$, where θ may be interpreted as the risk neutral long-run equilibrium level. Similarly from 3), $\lim_{t\to\infty^+} \mathrm{Var}(r_t)=\frac{\sigma^2}{2\kappa}$, as t reaches the limit, the variance of the short rate approaches a finite value due to the mean-reversion parameter of κ. Furthermore, as $r_t \sim \mathcal{N}$, a potential drawback of the model is that the short rate may assume negative values. As an example, figure 5.1(c) depicts 10 random paths of the short rate generated using a discretized version of the model SDE in eq (5.2) form which the risk of negative rates is apparent. For calibrated model parameters, the literature, however, suggests that the risk of negative rates may be negligible.[2] Moreover, recent developments in selected short term European Government bond-yields have shown, that negative short rates may not be as inconceivable as previously anticipated.[3] This has to some extent provided a small renaissance to the class of short-rate models.

[1] [8, sec. 4.3].
[2] [1, p. 74]
[3] As per Jul-2012 e.g. Germany, Finland and Denmark have sold 2Y notes at negative yields. Ref Bloomberg™.

5.1.3 Future Bond Prices

Next, we turn to the derivation of bond prices, i.e. an analytical expression for $P(t,T)$. Combined with the general expression of bond prices from (1.9) the Vasicek short rate SDE in (5.2), leads to the following PDE for bond prices

$$\frac{1}{2}\frac{\partial^2 P}{\partial r_t^2}\sigma^2 + \frac{\partial P}{\partial r}[\kappa(\theta - r_t)] + \frac{\partial P}{\partial t} - r_t P = 0, \tag{5.14}$$

with the boundary condition $P(T,T) = 1$, i.e. all bond prices at maturity equal unity. By an "educated guess", inspired by [3, chpt. 4], we propose a solution to (5.14) to take a so-called exponential-affine form

$$P(t,T) = \exp[A(\tau) - B(\tau)r_t]. \qquad \tau = T\text{-}t \tag{5.15}$$

To confirm that (5.15) is a solution to (5.14), we derive its partial derivatives with respect to r_t and t

$$\frac{\partial P}{\partial r_t} = -B(\tau)\exp[A(\tau) - B(\tau)r_t] = -B(\tau)P(t,T) \tag{5.16a}$$

$$\frac{\partial^2 P}{\partial r_t^2} = \exp[A(\tau) - B(\tau)r_t]B(\tau)^2 = B(\tau)^2 P(t,T) \tag{5.16b}$$

$$\frac{\partial P}{\partial t} = -\frac{\partial P}{\partial \tau} = -[A'(\tau) - B'(\tau)r_t]P(t,T), \tag{5.16c}$$

where $A'(\tau) = \frac{\partial A(\tau)}{\partial \tau}$ and $B'(\tau) = \frac{\partial B(\tau)}{\partial \tau}$. Substituting (5.16a)-(5.16c) into eq (5.14) moving $P(t,T)$ outside the brackets yields

$$\left(\frac{1}{2}B^2(\tau)\sigma^2 - B(\tau)[\kappa(\theta - r)] - A'(\tau) + B'(\tau)r - r\right)P = 0. \tag{5.17}$$

For this to hold for an arbitrary P, we have

$$\left(\frac{1}{2}B^2(\tau)\sigma^2 - B(\tau)\kappa\theta - A'(\tau)\right) - \left(B(\tau)\kappa + B'(\tau) - 1\right)r_t = 0, \tag{5.18}$$

so that each coefficient of the 1. order polynomial in P must vanish for the PDE in eq (5.18) to be satisfied. This leads to the following two first-order ODEs

$$A'(\tau) = \frac{1}{2}\sigma^2 B^2(\tau) - B(\tau)\kappa\theta \tag{5.19}$$

$$B'(\tau) = 1 - \kappa B(\tau). \tag{5.20}$$

If we are able to find a solution to the two ODEs this will confirm that eq (5.15) is indeed a solution to eq (5.14). As $\exp(0) = 1$, the PDE boundary condition of $P(T,T) = 1$ translates according to eq (5.15) into the following boundary conditions for the ODEs

$$A(0) = 0 \quad \text{and} \quad B(0) = 0. \tag{5.21}$$

As eq (5.19) only involves $B(\tau)$, it is possible to solve the systems of ODEs recursively. First, we derive (5.20) and then by substitution into (5.19), $A(\tau)$ can be found. First, we rewrite $B'(\tau)$ as

$$B'(\tau) + \kappa B(\tau) = 1, \tag{5.22}$$

and augment by the factor $e^{\kappa\tau}$

$$B'(\tau)e^{\kappa\tau} + \kappa B(\tau)e^{\kappa\tau} = e^{\kappa\tau}. \tag{5.23}$$

By recognizing the structure of the l.h.s. using the product rule for differentiation, eq (5.23) can be rewritten as

$$\frac{d}{d\tau}\left\{e^{\kappa\tau}B(\tau)\right\} = e^{\kappa\tau}. \tag{5.24}$$

Since $B(\tau)$ does not appear on the r.h.s of eq (5.24), we may obtain a general solution through integration

$$\begin{aligned}
\int \frac{d}{d\tau}\left\{e^{\kappa\tau}B(\tau)\right\} &= \int e^{\kappa\tau} d\tau \quad \Longrightarrow \\
e^{\kappa\tau}B(\tau) &= \frac{1}{\kappa}e^{\kappa\tau} + c \quad \Longrightarrow \\
B(\tau) &= \frac{1}{\kappa} + ce^{-\kappa\tau}.
\end{aligned} \tag{5.25}$$

The particular solution of (5.20) satisfing $B(0) = 0$ thus becomes

$$\begin{aligned}
B(\tau) &= \frac{1}{\kappa} - \frac{1}{\kappa}e^{-\kappa\tau} \\
&= \frac{1 - e^{-\kappa\tau}}{\kappa}.
\end{aligned} \tag{5.26}$$

Next, we consider the solution for $A(\tau)$ which is obtain by integrating eq (5.19) using the boundary condition $A(0)=0$

$$A(\tau) = A(0) + \int_0^\tau A'(s)ds = \frac{1}{2}\sigma^2 \int_0^\tau B^2(s)ds - \kappa\theta \int_0^\tau B(s)ds. \tag{5.27}$$

Accordingly, we need to find a solution for the integrals of $B(s)$ and $B^2(s)$, respectively. We have

$$\begin{aligned}
-\kappa\theta \int_0^\tau B(s)d(s) &= -\frac{\theta}{\kappa}\left[\kappa\tau + e^{-\kappa\tau}\right]_0^\tau \\
&= -\frac{\theta}{\kappa}\left(\kappa\tau + e^{-\kappa\tau} - 1\right) \\
&= -\theta\left(\frac{e^{-\kappa\tau} - 1 + \tau}{\kappa}\right),
\end{aligned} \tag{5.28}$$

and

$$\begin{aligned}
\frac{1}{2}\sigma^2 \int_0^\tau B^2(s)ds &= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left(e^{-\kappa\tau}\right)^2 \\
&= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left(1 + e^{-2\kappa\tau} - 2e^{-\kappa\tau}\right) \\
&= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left[\tau - \frac{e^{-2\kappa\tau}}{2\kappa} + \frac{2e^{-\kappa\tau}}{\kappa}\right]_0^\tau \\
&= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left[\tau - \frac{e^{-2\kappa\tau}}{2\kappa} + \frac{2e^{-\kappa\tau}}{\kappa} - \left(0 - \frac{e^{-2\kappa 0}}{2\kappa} + \frac{2e^{-\kappa 0}}{\kappa}\right)\right] \\
&= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left[\frac{1 - e^{2\kappa\tau}}{2\kappa} + 2\frac{(e^{-\kappa\tau} - 1)}{\kappa} + \tau\right] \\
&= \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2 \left[\frac{1 - e^{-2\kappa\tau} - 4\left(1 - e^{-\kappa\tau}\right)}{2\kappa} + \tau\right].
\end{aligned} \tag{5.29}$$

The particular solution, satisfying the boundary conditions, in (5.21) thereby reads (after rearranging the expressions)

$$B(\tau) = \frac{1 - e^{-\kappa\tau}}{\kappa} \tag{5.30}$$

$$A(\tau) = \left(\theta - \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2\right)\left(B(\tau) - \tau\right) - \frac{\sigma^2}{4\kappa}B^2(\tau), \tag{5.31}$$

and accordingly, bond prices in the Vasicek77 model are given as

$$P(t,T) = \exp\left[A(\tau) - B(\tau)r_t\right] \qquad \text{where } \tau = T\text{-}\,t, \tag{5.32}$$

which corresponds to eq (44)-(46) in ref [3, p. 9], though with a slightly different notation. As clearly seen from eq (5.30), $B(\tau) > 0$ and so an increase in the short rate r_t, lowers bond prices as we would widely expect.

5.1.4 The Implied Term Structure of Interest Rates

Next, we turn to consider the possible term structure of interest rates implied by the model. By the simple relation between spot rates and bond prices from eq (1.5) we have

$$\begin{aligned} y(\tau) &= -\frac{\ln P(\tau)}{\tau} \\ &= -\frac{1}{\tau}\left[A(\tau) - B(\tau)r_t\right], \end{aligned} \tag{5.33}$$

where $y(\tau)$ denotes the continuously compounded spot rate for maturity $\tau_t = T\text{-}\,t$. As $y(\tau)$ is linear in r_t, and since r_t is the product of a one-factor SDE, we know that all spot rates $y(\tau)$ are perfectly correlated in the model.[4] Such features have serious short-comings when pricing relative curve-dependent instruments, such as spread options, as the model produces too small amounts of spread volatility.[5] As spread options lies outside the scope of this thesis, we restrict ourselves to noting the issue. In eq (5.33), we note that a finite limit for the spot rate exists

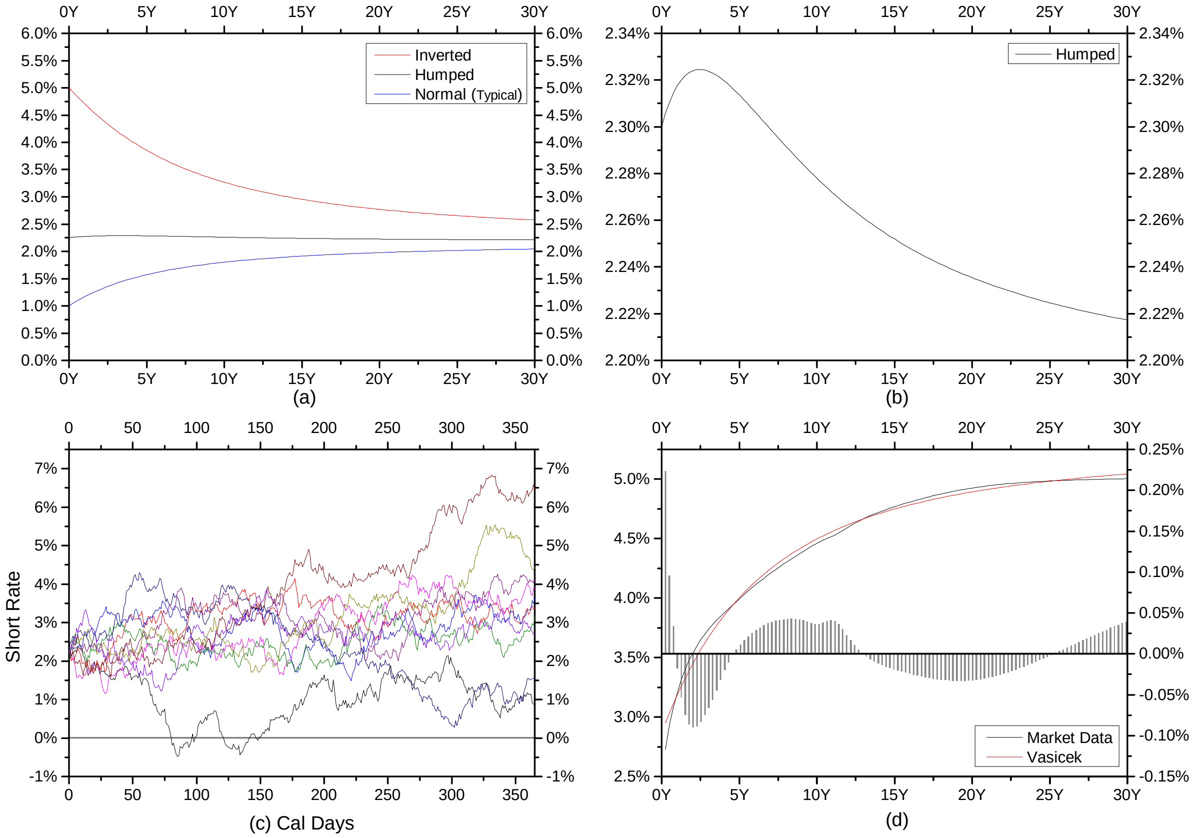
$$y_\infty = \lim_{\tau\to\infty} y(\tau) = \theta - \frac{1}{2}\left(\frac{\sigma}{\kappa}\right)^2. \tag{5.34}$$

1. If $r_t > \theta$, then $y_\infty(\tau)$ decreases in τ
2. If $r_t < y_\infty - \frac{\sigma_r^2}{4\kappa^2}$, then $y(\tau)$ increases in τ
3. Otherwise, if $y_\infty - \frac{\sigma_r^2}{4\kappa^2} < r_t < \theta$ then $y(\tau)$ first increases then decreases in τ i.e. is (slightly) humped

Further, it can be shown that the future term structure produced by the Vasicek model may assume three different curve shapes[6] We remark that the possible future yield curve dynamics under the Vasicek77 model are quite restrictive, as clearly verified in figure 5.1(a)-(b). Further, we consider the special case of $y(0,T)$, i.e. the current spot curve at $t=0$. In belonging to the group of classical equilibrium models, the Vasicek77 model *is* able to reproduce "the" current yield curve. However, even a well calibrated model fails in fitting the current $(t=0)$ yield curve accurately enough for pricing applications which in turn is a major drawback for the model framework. Figure (5.1)(d) illustrates the problem for a case where we have calibrated the model to the initial yield curve only. That is, no volatility information have been used in the calibration and thus figure (5.1)(d)

[4] [18, p. 8].
[5] Discussions with Supervisor.
[6] [5, p. 65].

Figure 5.1: The Vasicek Model - an graphical outline of the dynamics. Top left (a), shows the three feasible yield curve dynamics in the model; 1) asymptotically upwards sloping (typical), 2) Inverted and 3) slightly humped $r_0 = \{1.00, 2.25, 5.00\}$. Top right (b), verify the only "slighty" humped shape $r_0 = 2.30$. Bottom left (c), depicts 10 random paths of the short rate generated using a discretized version of the model SDE in eq (5.2) $r_0 = 2.30$ and $\delta t = 1W$. Bottom right (d), illustrates the Vasicek model calibrated to fit the initial ($t=0$) yield curve only † ††. Parametrization: [$\kappa = 0.25$, $\theta = 0.03$, $\sigma = 0.02$].



† Market data from section 6.1. †† Parametrization for the yield curve (only) calibrated Vasicek model in (d) [$\kappa = 0.25$, $\theta = 0.06$, $\sigma = 0.02$, $r_0 = 2.87$]

emphasizes the issue even further. Examining eq (5.30)-(5.32), this finding is not surprising as no information from the current yield curve is used as input to the model. Another, yet more severe example of its short-comings were apparent during Q1-2 2012, where the US swap curve expressed a concave upwards sloping term structure in the 2-10Y segment.[7] As evident from figure 5.1(a)-(b), such dynamics cannot, even somewhat, be reproduced in the model.

The ability to fit current market prices is, for obvious reasons, an instrumental feature to expect from any term structure model. To this end, we will in the following consider certain well-known measures to circumvent the issue.

5.2 The Hull-White Vasicek Model

The poor fitting of the current term structure incurred by the Vasicek77 model has been addressed by Hull and White in a series of papers (the first published in 1990). In their most basic version, the original model is augmented with a deterministic time-varying mean-drift.

[7] Bloomberg™.

As we will show below, this modification allows the initial term structure of interest rates to be perfectly matched. In fact, we will see that the way the HWV model takes the initial yield curve as input corresponds to an internal a-priori calibration of the curve regardless of its functional form. Provided the match of the current yield curve, we will however see, that problems regarding matching the initial term structure of volatilities still remains.

5.2.1 Definition of the model

Under the risk neutral measure, the instantaneous short rate under HWV evolves according to the SDE

$$dr_t = \kappa\big(\theta_t - r_t\big)dt + \sigma dW_t^Q, \tag{5.35}$$

where θ_t now is a deterministic time-varying riskless mean-drift parameter. To solve the SDE we follow the same approach as previous and introduce the function $g(r_t) = r_t e^{\kappa t}$. Applying Itô's Lemma gives

$$dg = \theta_t \kappa e^{\kappa t} dt + e^{\kappa t}\sigma dW_t^Q, \tag{5.36}$$

and integration of both sides yields

$$\begin{aligned}\int_0^t dg &= \kappa \int_0^t \theta_s e^{\kappa s} ds + \sigma \int_0^t e^{\kappa s} dW_s^Q \\ g(r_t) - g(r_0) &= \kappa \int_0^t \theta_s e^{\kappa s} ds + \sigma \int_0^t e^{\kappa s} dW_s^Q.\end{aligned} \tag{5.37}$$

Now, realizing that $g(r_t)=r_t e^{\kappa t}$, i.e. $r_t=g(r_t)e^{-\kappa t}$ and thus $r_0=g(r_0)$, the solution for r_t is obtained by multiplying by $e^{-\kappa t}$ on both sides of eq (5.37) and rearranging terms

$$\begin{aligned}r_t e^{\kappa t} - r_0 &= \kappa \int_0^t \theta_s e^{\kappa s} ds + \sigma \int_0^t e^{\kappa s} dW_s^Q \\ r_t &= r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)} \kappa \theta_s ds + \sigma \int_0^t e^{-\kappa(t-s)} dW_s^Q,\end{aligned} \tag{5.38}$$

where (5.38) yields the short rate process in level form as the solution to the SDE in (5.35). Before discussing the distributional moments, we note that the overall aim is to obtain a model, capable of matching the current yield curve; that is, we need to calibrate the introduced time-varying mean-drift parameter θ_t. To ease the calibration process, we follow ref [11, chpt. 2.1] and introduce an equivalent representation of eq (5.38)[8]

$$r_t = m_t + x_t, \tag{5.39}$$

where m_t and x_t are defined as

$$m_t = r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)} \kappa \theta_s ds \tag{5.40}$$

$$dx_t = -\kappa x_t dt + \sigma dW_t^Q \qquad \text{with } x_0 = 0. \tag{5.41}$$

Before further exploration of the calibration procedure, we note that by the above and in combination with the Feynman-Kac formula from eq (2.14), the arbitrage-free bond price is given by the risk-neutral expectation

$$P(t,T) = E_t^Q\left[e^{-\int_t^T r_s ds}\right],$$

[8]For a formal prove of equivalence we refer to ref [19].

which, analogous to ref [11, p. 2], can be rewritten according to

$$
\begin{aligned}
P(t,T) &= e^{-\int_t^T m(s)ds}\, E_t^Q\left[e^{-\int_t^T x_s ds}\right] \\
&= \exp\Big(-\int_t^T m(s)ds\Big)\exp\Big(A(\tau)-B(\tau)x_t\Big),
\end{aligned} \tag{5.42}
$$

where

$$
B(\tau) = \frac{1-e^{-\kappa\tau}}{\kappa} \tag{5.43}
$$

$$
A(\tau) = \frac{1}{2}\sigma_r^2\int_t^T B^2(s)ds = \frac{1}{2}\left(\frac{\sigma_r}{\kappa}\right)^2\left[\frac{1-e^{-2\kappa\tau}-4(1-e^{-\kappa\tau})}{2\kappa}+\tau\right]. \tag{5.44}
$$

We note that $P(t,T)$, in (5.42), is defined as the product of a deterministic factor and the bond price from the original Vasicek77 model under Q. Furthermore, we remark that the necessary calibration is still apparent from (5.42) via m_t.

5.2.2 Calibration to the Current Yield Curve

Our aim is essentially to be able to accurately fit the current yield curve (at time $t=0$), which we will represent by the discount function $d(T)$ from section 1. Using (5.42) where from (5.41), $x_0=0$, gives

$$
P(0,T) = \exp\Big(-\int_0^T m(s)ds + A(T)\Big) = d(T). \tag{5.45}
$$

Taking logarithms and rearranging terms yields

$$
\int_0^T m(s)ds \;=\; -\ln d(T) + A(T). \tag{5.46}
$$

Further, we differentiate eq (5.46) with respect to T on both sides using the first fundamental theorem of calculus, realizing that the first term is already given as the instantaneous forward rate from eq (1.7) so that

$$
m(T) = -\frac{d\ln d(T)}{dT} + \frac{dA(T)}{dT} = f(0,T) + \frac{1}{2}\sigma_r^2 B^2(T). \tag{5.47}
$$

Eq (5.47) shows that $m(T)$ is obtained from the current forward curve and when calibrated, it mimics the dynamics of the forward curve rather closely. The time-invariant parameters κ and σ have to be backed out from market prices of standard Caps (Floors) and Swaptions prior to the calibration of $m(T)$. Such calculations and the calibration of $m(T)$ are revisited in practice in section 7 when calibrating the model to market prices of Caps (Floors).

For now, we will proceed by the determination of $\theta(t)$. Applying the Leibniz integral rule[9] to the case, in which the limits of integration and the integrand are all functions of the same parameter t, the first derivative of $m(t)$ in (5.40) can be written as

$$
m'(t) = -\kappa e^{-\kappa t} r_0 + \kappa\theta(t) - \kappa^2\int_0^t e^{-\kappa(t-s)}\theta(s)ds, \tag{5.48}
$$

where the additional term $\kappa\theta(t)$ arises from the upper boundary term in the Leibniz rule. Further, eq (5.48) can be rewritten as

$$
m'(t) = \kappa\theta(t) - \kappa m(t). \tag{5.49}
$$

[9] The Leibniz Rule: $\frac{d}{d\alpha}\int_{a(\alpha)}^{b(\alpha)} f(x,\alpha)dx = \frac{db(\alpha)}{d\alpha}f\big(b(\alpha),\alpha\big) - \frac{da(\alpha)}{d\alpha}f\big(a(\alpha),\alpha\big) + \int_{a(\alpha)}^{b(\alpha)}\frac{\partial}{\partial\alpha}f(x,\alpha)dx$.

Utilizing this result and the definition of $m(t)$ from (5.47), we have

$$\begin{aligned}\kappa\theta(t) &= \kappa m(t) + m'(t)\\ &= \kappa f(0,t) + \frac{1}{2}\kappa\sigma_r^2 B^2(t) + \frac{\partial f(0,t)}{\partial t} + \sigma^2 B(t)B'(t)\\ &= \kappa f(0,t) + \frac{\partial f(0,t)}{\partial t} + \phi(t), \end{aligned} \tag{5.50}$$

where $\phi(t)$ is defined as

$$\phi(t) = \frac{1}{2}\kappa\sigma^2 B^2(t) + \sigma^2 B(t)B'(t). \tag{5.51}$$

We proceed by inserting $B(t)$ and its time-derivative into (5.51)

$$\begin{aligned}\phi(t) &= \frac{1}{2}\kappa\sigma^2\left(\frac{1-e^{-\kappa t}}{\kappa}\right)^2 + \sigma^2\left(\frac{1-e^{-\kappa t}}{\kappa}\right)e^{-\kappa t}\\ &= \frac{1}{2}\frac{\kappa\sigma^2}{\kappa^2}\left(1+e^{-2\kappa t}-2e^{-\kappa t}\right) + \frac{\sigma^2}{\kappa}\left(e^{-\kappa t}-e^{-2\kappa t}\right)\\ &= \frac{\sigma^2}{\kappa}\left(\frac{1}{2}+\frac{1}{2}e^{-2\kappa t}-e^{-\kappa t}+e^{-\kappa t}-e^{-2\kappa t}\right)\\ &= \frac{\sigma^2}{\kappa}\left(\frac{1}{2}-\frac{1}{2}e^{-2\kappa t}\right)\\ &= \frac{\sigma^2}{2\kappa}\left(1-e^{-2\kappa t}\right) = \mathrm{Var}_0[r_t]. \end{aligned} \tag{5.52}$$

As with the Vasicek model eq (5.52), is recognized as the variance of $[r_t|_{r_0}]$.[10] Finally, we insert $\kappa\phi(t)$ into the SDE in (5.35), thereby yielding

$$dr_t \;=\; \left[\kappa\big(f(0,t)-r_t\big) + \frac{\partial f(0,t)}{\partial t} + \phi(t)\right]dt + \sigma_r dW_t^Q. \tag{5.53}$$

This illustrates how the time-dependent parameters of the SDE are obtained from the current forward curve. Further, it should be noted that $\phi(t)$ usually is fairly small[11] and so if the contribution from $\phi(s)$ to the drift-term is ignored, r_t on average follows the slope of the current forward curve and when deviations occur it reverts at a rate of κ.[12]

For implementation of numerical procedures engaging in direct simulation of r_t via eq (5.53), it is, however, not the most convenient approach as it involves the determination of the time-derivative of the forward curve. Hence, this requires a fully differentiable forward curve which is rarely available in practice and thus, it has to be approximated using e.g. splines static curve-smoothing techniques.[13] Therefore, we follow the alternative SDE-representation ((5.39)-(5.41)) for the later numerical implementations.

Before proceeding, we consider the volatility of the instantaneous forward rate $f(t,T)$, which we shall denote $\sigma_f(t,T)$. From eq (1.7), we have the basic relationship between bond prices and instantaneous forward rate

$$f(t,T) = -\frac{\partial \ln P(t,T)}{\partial T}. \tag{5.54}$$

[10] [1, chpt. 3.3].
[11] [9, p. 656].
[12] Ref [9, Fig. 28.4] illustrates the dynamics.
[13] Which are without further contribution to the subject of this thesis.

From section 2.2, the bond prices evolve according to the SDE

$$dP(t,T) = r_t P(t,T)dt + \sigma_{P(t,T)} P(t,T) dW_t^Q,$$

while, the conceptual instantaneous forward rate SDE evolve as

$$df(t,T) = \mu_f(t,T)dt + \sigma_f(t,T)dW_t^Q,$$

where $\mu_f(t,T)$ and $\sigma_f(t,T)$ are of course yet unknown expressions. As $\frac{\partial f}{\partial P(t,T)} = -\frac{\partial}{\partial T}\frac{1}{P(t,T)}$ and $\frac{\partial^2 f}{\partial P(t,T)^2} = \frac{\partial}{\partial T}\frac{1}{P(t,T)^2}$, then by the last term of Itô's lemma, the forward rate volatility is given as

$$\begin{aligned}\sigma_f(t,T) &= \sigma \frac{\partial}{\partial T} B(t,T) \\ &= \sigma \frac{\partial}{\partial T}\left(\frac{1-e^{-\kappa(T-t)}}{\kappa}\right) \\ &= \sigma\, e^{-\kappa(T-t)}. \end{aligned} \tag{5.55}$$

Thus, the mean-reversion induces exponential decay in the volatility term structure of forward rates produced by the HWV model. Particularly, ref [12, p. 418] finds (5.55) appealing as it offers time-stationarity of the forward rate volatility term structure. That is, from (5.55) as time goes the volatility term structure $\sigma_f(t+i, T+i)$ will look the same as today. In absence of other information, this assumption is often very reasonable and consistent with empirical observations.[14]

As found in figure 4.1, in practice however, it is quite common to observe that the volatility term-structure[15] exhibits a marked "hump" on shorter dated options. Accordingly, such dynamics are not replicable in the HWV model because $\theta(t)$ is the only time-dependent parameter. We will further investigate the implications of the above when calibrating the HWV model to market volatilities in section 7.2, and the issue will be expanded upon in the following section on the Hull-White Extended Vasicek model.

5.2.3 Future Bond Prices

Next, we will consider the analytical determination of future bond prices $P(t,T)|_{t=0}$. From eq (5.42) in combination with (5.39), we have

$$P(t,T) = \exp\Big(-\int_t^T m(s)ds\Big)\exp\Big[A(\tau) - B(\tau)\big(r_t - m(t)\big)\Big], \tag{5.56}$$

where $m(s)$ is calibrated to the initial term structure. To obtain an integral-free expression, eq (5.56) is rewritten. First we note, that the forward bond price$|_{t=0}$ for the T-maturing bond is given by (using that $x_0 = 0$, cf. (5.41))

$$\begin{aligned}\frac{P(t,T)}{P(0,t)} &= \frac{\exp\left(-\int_0^T m(s)ds + A(T)\right)}{\exp\left(-\int_0^t m(s)ds + A(t)\right)} \\ &= \exp\left(-\int_t^T m(s)ds + A(T) - A(t)\right). \end{aligned} \tag{5.57}$$

[14] [12, p. 418]

[15] Here represented by the flat volatilities term-structure of Caps (Floors).

Second, we utilize (5.57) to rewrite (5.56) so that

$$P(t,T) = \exp\Big(\Big(-\int_t^T m(s)ds + A(T) - A(t) - A(T) + A(t)\Big)\Big) \exp\Big[A(\tau) - B(\tau)\big(r_t - m(t)\big)\Big]$$

$$= \frac{P(0,T)}{P(0,t)} \exp\Big[A^*(t,T) - B(\tau)\big(r_t - m(t)\big)\Big], \tag{5.58}$$

where $A^*(t,T) = \Big(-A(T) + A(t) + A(\tau)\Big)$. If we insert the expression for $A(\tau)$ then $A^*(t,T)$ can be rewritten as[16]

$$A^*(t,T) = -\frac{1}{2}B^2(\tau)\phi(t) + \frac{1}{2}\sigma^2 B(\tau)B^2(t), \tag{5.59}$$

where $\phi(t)$ is given in (5.51). Further, using the expression for $m(t)$ in eq (5.47), we have

$$P(t,T) = \frac{P(0,T)}{P(0,t)} \exp\Big[-\frac{1}{2}B^2(\tau)\phi(t) - B(\tau)\big(r_t - f(0,t)\big)\Big]. \tag{5.60}$$

From (5.60) we realize that at $t=0$ both terms in the brackets vanish, and thus, $P(t,T) = \frac{P(0,T)}{P(0,t)}$. By definition, this corresponds to the model being perfectly calibrated to the current forward curve represented by $P(0,T)$ - regardless of its shape. Moreover, for $t > 0$, $P(t,T) \neq \frac{P(0,T)}{P(0,t)}$ and depends on the current forward curve and the distribution of r_t (in section 5.4 we shall consider the exact distribution of $P(0,t)$). In the applied section 9, we shall see how this "by construction" calibration to the current forward curve via $m(t)$ takes form when implementing numerical procedures. As a final remark, we note that as $B(\tau) > 0$, an increase in the short rate still lowers bond prices and conversely and so still in the HWV model, the fundamental relationship between rates and bond prices exists.

In this section we have shown, that by application of the arbitrage-free HWV model the problems related to matching the current yield curve faced in section 5.1 can be solved. Further, we found that the fundamental dynamics of the HWV model are essentially unchanged, and thus, the forward rate volatilities have the same time-stationary appealing expression as in the Vasicek77 model.[17] However, we also found that the strictly exponential decay of forward rate volatilities implied by the HWV model revealed one of its largest drawbacks as it often fails in capturing the current volatility term structure in practice. Thus, the HWV model offers too few degrees of freedom for many derivative pricing applications as it rarely calibrates well to observed prices of vanilla options.[18] This point will be verified in the applicational section 7.2 when calibrating the HWV model to market volatilities of traded Caps (Floors).

To circumvent this particular issue, the so-called Hull-White Extended Vasicek will be considered in the next section.

5.3 The Hull-White Extended Vasicek Model

To approximate the frequently observed "volatility hump", Hull and White introduced in their most general form a setup where all model parameters; $\kappa(t)$, $\theta(t)$ and $\sigma(t)$ are allowed deterministic functions of time. Applied to specific market situations the general setup may, however, cause issues with too strong non-stationarity and unrealistic evolution in forward rate volatilities as noted by Hull and White themselves [20]. We refer to ref [12, section 10.2.2.3] for a good discussion of the issue.

[16] We refer to ref [11, p. 4], upon which our approach is inspired.

[17] We did not actually derive the forward rate volatilities in the Vasicek77 section, however looking at the derivation of eq (5.55) verifies that the calculations are identical in the two models.

[18] [12, sec. 10.1.2.3].

To this end, our setting follows the approach suggested by ref [12, p. 419], where κ is kept constant, while both $\theta(t)$ and $\sigma(t)$ are allowed functions of time. This setup will be referred to as the Hull-White Extended Vasicek model [HWExtV]. Since dynamics and derivations of the HWExtV model are very similar to the two preceding model setups, we are generally content with the end results needed for later implementation.

5.3.1 Definition of the model

Under the risk neutral measure, the instantaneous short rate under the HWExtV-model evolves according to the SDE[19]

$$dr_t = \kappa\big(\theta(t) - r_t\big)dt + \sigma(t)dW_t^Q, \tag{5.61}$$

where both $\theta(t)$ and $\sigma(t)$ are deterministic functions of time. By a similar approach as before, the level form process of the short rate process is given by[20]

$$r_t = r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)}\kappa\theta(s)ds + \int_0^t \sigma(s)e^{-\kappa(t-s)}dW_s^Q, \tag{5.62}$$

where $\sigma(t)$ now appears in the integrand of the last term. The model needs calibration of both $\theta(t)$ and $\sigma(t)$, and using the previously introduced alternative representation from eq (5.39), we have

$$r_t = m(t) + x_t, \tag{5.63}$$

where $m(t)$ and x_t are defined as

$$m(t) = r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)}\kappa\theta(s)ds \tag{5.64}$$

$$dx_t = -\kappa x_t dt + \sigma(t)dW_t^Q \qquad \text{with } x_0 = 0. \tag{5.65}$$

5.3.2 Moments and Distribution of the Short Rate

From eq (5.62), moments and distribution of $[r_t|_{t=0}]$ is given as

1. $r_t \sim \mathcal{N}$ normally distributed
2. $E(r_t|_{r_0}) = r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)}\kappa\theta(s)ds$
3. $Var(r_t|_{r_0}) = \int_0^t \sigma^2(s)e^{-2\kappa(t-s)}ds$

which, taking into account the time dependency of $\sigma(t)$, essentially is the same dynamics as in the HWV model.

5.3.3 Future Bond Prices

Bond prices are, as before, given as

$$P(0,T) = E^Q\left[e^{-\int_t^T r_s ds}\right].$$

In analytical form, bond prices can be shown to develop according to[21]

$$P(t,T) = \frac{P(0,T)}{P(0,t)}\exp\left(-\frac{1}{2}B^2(\tau)\phi(t) - B(\tau)\big(r_t - f(0,t)\big)\right), \tag{5.66}$$

[19] [21, p. 34].
[20] [1, chpt. 3.3].

where $B(\tau)$ is given as in (5.44) and

$$\phi(t) = \int_0^t \sigma^2(s) e^{-2\kappa(t-s)} ds. \tag{5.67}$$

From eq (5.66) we see that similar to the HWV model, the perfect calibration to the current yield curve is built into the HWExtV model, as the bracketed terms in eq (5.66) becomes zero, for $t=0$. Moreover, the term structure of forward rate volatilities takes the form[22]

$$\sigma_f(t,T) = \sigma(t) e^{-\kappa(\tau)}, \tag{5.68}$$

which, as noted in ref [12, p. 419], is *not* time-stationary due to the time-varying structure of $\sigma(t)$, although it retains a persistent exponential development of the forward rate volatility structure through time. As this thesis is only concerned with pricing of contingent claims at time $t=0$, the inherent problem of non-stationarity is ignored. Moreover, we remark that the HWExtV model is unstable; that is, all model parameters have to be re-calibrated over time so as to fit the market.[23]

5.3.4 The Implied Term Structure of Interest Rates

By the simple relation between spot rates and bond prices from (1.5), the yield curve is given as

$$y(t,T) = -\frac{1}{\tau} \ln\left(\frac{P(t,T)}{P(0,t)}\right) \left[-\frac{1}{2} B^2(\tau)\phi(t) - B(\tau)\big(r_t - f(0,t)\big)\right], \tag{5.69}$$

where $\phi(t)$ is still given as in eq (5.67).

5.3.5 Derivation of a Piecewise Linear Volatility Function

As $\phi(t)$, under the HWExtV model, involves the integral of some unknown function, $\sigma(t)$, we need to specify its functional in order to deploy the HWExtV setup. From eq (5.67), we have

$$\phi(t) = \int_0^t \sigma^2(s) e^{-2\kappa(t-s)} ds. \tag{5.70}$$

In specifying the functional form of $\sigma(t)$, we chose to follow the parametrization proposed by ref [21, p. 39], where the volatility is defined as a piecewise constant (step) function of time. That is; $\sigma(t)=\sigma_j$ for any $t\in[t_{j-1}, t_j]$, $j\in 1,2,..,n$, where $t_n=t$, so that

$$\begin{aligned} \phi(t) &= \sum_{j=1}^{n} \sigma_j^2 \int_{t_{j-1}}^{t_j} e^{-2\kappa(t-s)} ds \\ &= \frac{1}{2\kappa} \sum_{j=1}^{n} \sigma_j^2 \Big[e^{-2\kappa(t-s)}\Big]_{t_{j\text{-}1}}^{t_j}. \end{aligned} \tag{5.71}$$

We deploy eq (5.71) in section 7.1.3 and the applied source code can be found in Appendix C, module `HullWhite/PHIHW`.

[21] [21, p. 35].
[22] [12, p. 419].
[23] All three models; Vasicek77, HWV and the HWExtV model hold this basic issue which is a widely common problem of most term structure models.

5.4 European Options Pricing under Vasicek and the Descendants Models

In this section our aim is to develop analytical formulae for pricing of European Caps (Floors) in the reviewed models. First, we describe how options on zero-coupon bonds can be evaluated analytically. Second, we derive a replication strategy in zero-coupon bonds, which allows us to price standard European Caps (Floors) as a portfolio of put (call) options on zero-coupon bonds.

These combined tools are important for the later application of the models, which allows us to efficiently calibrate the model parameters via analytical measures. Thus, the methods, described in the following section, will be repeatedly used for calibration purposes, in sections 7.2, 9.1.1 and 9.2.1

5.4.1 Options on Zero Coupon Bonds

The payoff from a call option on a T-maturing zero coupon bond [ZCB] with option expiry at time s, $s<T$, and a payoff occurring at time s (the time of calculation/reset), using the positive part operator is

$$C_s = \big(P(s,T) - k\big)^+, \tag{5.72}$$

where k is the strike price of the option. Pricing under the forward-risk adjusted T-measure is interesting because we need the distribution of the payoff C_s at time s, which only depends on $P(s,T)$ (unknown at time $t=t_0$), and thus, under the terminal measure can be calculated directly as the relative price

$$F(t;s,T) = \frac{P(s,T)}{P(s,s)}, \tag{5.73}$$

due to the well known property, $P(s,s)=1$. We recognize $F(t;s,T)$ as the current forward price of the T-maturing bond, delivered at time s. Provided the results derived in section 2.2.2, and in particular eq (2.31), the forward price, $F(t;s,T)$, will be a martingale under Q^T. That is

$$\begin{aligned} dF(t;s,T) &= \big(\sigma_{P(t,T)} - \sigma_{P(t,s)}\big)F(t;s,T)dW_t^{Q^T} \\ &= \sigma_{F(t;s,T)}F(t;s,T)dW_t^{Q^T}. \end{aligned} \tag{5.74}$$

In the three Vasicek descendant models, discussed in the previous two sections, the short rate was shown to be normally distributed under Q, and hence, bond prices followed a Geometrical Brownian Motion [GBM] that is; were lognormal.[24] Under Q^T we can show that these properties are retained. According to ref [10, p. 3], the volatility of the bond price is given as $\sigma_{P(t,T)} = -\sigma B(t,T)$. Hence, applying the change-of-measure technique described in section 2.2.1, (cf. eq (2.30)), we have

$$dW_t^{Q^T} = \sigma B(t,T)dt + dW_t^Q, \tag{5.75}$$

so that eq (5.2) can be rewritten as

$$\begin{aligned} dr_t &= \kappa(\theta - r_t)dt + \sigma dW_t^Q \\ &= \kappa(\theta - r_t)dt + \sigma(dW_t^{Q^T} - \sigma B(t,T)) \\ &= \big(\kappa(\theta - r_t) - \sigma B(t,T)\big)dt + \sigma dW_t^{Q^T}, \end{aligned} \tag{5.76}$$

[24] [9, p. 269].

where W^{Q^T} is a Brownian motion under Q^T. This shows that under the Q^T measure the short rate is normal and bond prices are still lognormal. Taking the logarithm of the forward bond price in eq (5.74) using Itô's lemma, where $\frac{\partial \log F}{\partial F} = \frac{1}{F}$ and $\frac{\partial^2 \log F}{\partial F^2} = -\frac{1}{F^2}$, gives

$$d\log F(t;s,T) = -\frac{1}{2}\sigma^2_{F(t;s,T)}dt + \sigma_{F(t;s,T)}dW_t^{Q^T}. \tag{5.77}$$

After integration, we have[25]

$$\log F(t;s,T) = \log P(s,T) = \log F(0;s,T) - \frac{1}{2}\int_0^s \sigma^2_{F(t;s,T)}dt + \int_0^s \sigma_{F(t;s,T)}dW_t^{Q^T}. \tag{5.78}$$

As $\sigma_{F(t;s,T)}$ is deterministic for all the reviewed (Gaussian) one-factor Vasicek descendant models,[26] then for our use, $\log F(t;s,T)$ is normally distributed with first and second moments given as

$$\mu_{F(s,T)} = \log F(0;s,T) - \frac{1}{2}\int_0^s \omega^2_{F(s,T)}dt \tag{5.79}$$

$$\omega^2_{F(s,T)} = \int_0^s \sigma^2_{F(t;s,T)}dt. \tag{5.80}$$

C.f. ref [10, p. 5], then $\omega^2_{F(s,T)}$ for our three models is given as

$$\begin{aligned}\omega^2_{F(s,T)} &= B^2(T-s)\mathrm{Var}^{Q^T}_{t=0}(r_s)\\ &= B^2(T-s)\phi(s),\end{aligned} \tag{5.81}$$

where we recognize the model-specific $\phi(s)$ from eq (5.12), (5.52) and (5.67).[27]

Finally, via calculations analogous to those of the Black-Scholes model on stock options, the price of the call option, $C_{s,k}$, is given by[28]

$$\begin{aligned}C_{s,k} &= P(0,s)E^{Q^T}_{t=0}[C_s]\\ &= P(0,T)\Phi(d_1) - P(0,s)\,k\,\Phi(d_2),\end{aligned} \tag{5.82}$$

where $\Phi(x)$, as in chapter 4, is the cumulative standard normal distribution function and

$$d_1 = \Big(\log\frac{P(0,T)}{P(0,s)} - \log k + \frac{1}{2}\omega^2_{F(s,T)}\Big)/\omega_{F(s,T)} \tag{5.83}$$

$$d_2 = d_1 - \omega_{F(s,T)}. \tag{5.84}$$

Accordingly, we have shown that options on zero-coupon bonds may be priced analytically under the reviewed Vasicek descendant frameworks. This finding shall now be utilized to price European Caps (Floors).

5.4.2 Replication Strategy for European Caps and Floors

Our goal is to price European Caps (Floors) via analytical measures and it turns out that it is possible to characterize the cash flow occurring in a Cap (Floor) by a replicating strip of put (call) options on zero-coupon bonds, with payoff on the puts (calls) at the time at which they are calculated.

[25] Recall that $P(s,s)=1$ which gives $\log F(t;s,T)=\log P(s,T)$.
[26] [10, p. 4].
[27] We note that eq (5.12) and (5.52) obviously are identical.
[28] We refer to ref [5, Appendix B.1] for a formal proof.

To show this, we start by considering the payoff from the ith Caplet, occurring at time T_i and from section 3.2, we have

$$\Pi_{Caplet}(T_i) = N\tau_i\big(L_{i\text{-}1}(T_{i\text{-}1}) - k\big)^+, \tag{5.85}$$

per N-units notional amount, where k as usual, is the strike rate. As found in section 2.2.3, the forward rate is a martingale under Q^T and so cf. section 3.2, eq (5.85) may, at time $T_{i\text{-}1}$, be valued as

$$\Pi_{Caplet}(T_{i\text{-}1}) = P(T_{i\text{-}1}, T_i)N\tau_i\big(L_{i\text{-}1}(T_{i\text{-}1}) - k\big)^+. \tag{5.86}$$

By inserting the definition of the Libor rate from section 1.3, we get

$$\begin{aligned}\Pi_{Caplet}(T_{i\text{-}1}) =& P(T_{i\text{-}1}, T_i)N\left(\frac{1}{P(T_{i\text{-}1}, T_i)} - 1 - k\tau_i\right)^+ \\ =& N\Big(1 - (1 + k\tau_i)P(T_{i\text{-}1}, T_i)\Big)^+.\end{aligned} \tag{5.87}$$

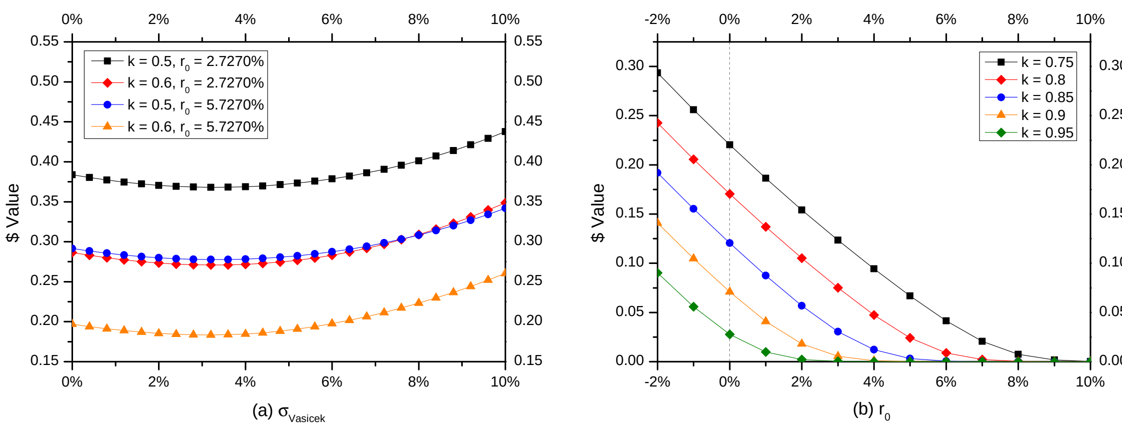
Thus, the value of a Caplet (Floorlet) at time $T_{i\text{-}1}$ can be expressed as a scaled payoff of a put (call) option on the zero-coupon bond, $P(T_{i\text{-}1}, T_i)$, when the face value of the bond is $N(1 + k\tau_i)$ and the strike price of the put (call) is N.

Accordingly, a European Cap (Floor) can be replicated by a strip of put (call) options on zero-coupon bonds and therefore, may be valued using the analytical formula given in eq (5.82). These results will be used repeatedly, in the applicational sections.

5.4.3 Vasicek and the Volatility Skew

Before proceeding to the implementational sections, we review, in short, the implications of the described theoretical (Hull-White) Vasicek framework(s). Particularly, we consider 1) prices of options on ZCBs and 2) modeled Black76 implied volatilities, when pricing European Caps (Floors) using our models. We use the basic Vasicek77 model to show the way, as the dynamics under the

Figure 5.2: Option on a ZCB in the Vasicek model. (a) Shows the \$-value of a call option on a ZCB at various strike levels and initial short rates, plotted as function of the volatility parameter. (b) Depicts the \$-value of a call option on a ZCB for various strike levels, plotted as function of the initial short rate. Parametrization: $\kappa = 0.0577$, $\theta = 3.0972$, $\sigma = 0.0115$, $r_0 = 2.727$, $\lambda = 0.1$.



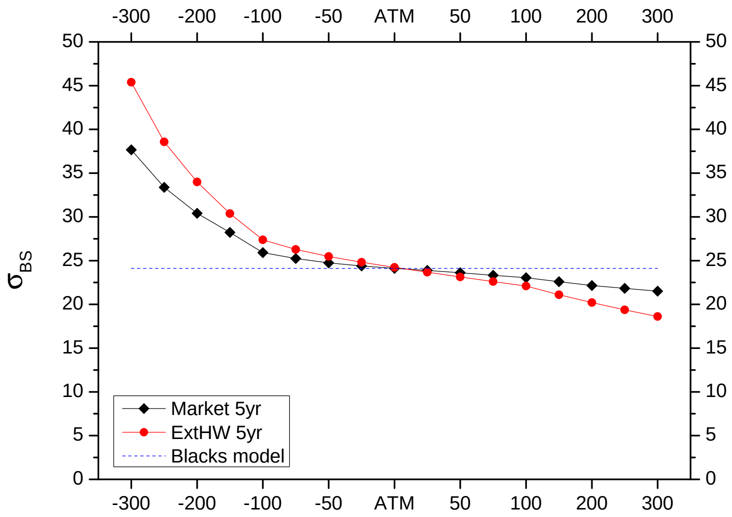
The initial forward bond prices, ZCB(t_0; $S = 1$, $T = 5$), are 0.8952 | $r_0 = 2.727$ and 0.8042 | $r_0 = 2.727$. The used parametrization is discretionary chosen so as to: 1) match the results of the later parametrization in table 7.1(a) and 2) cohere for r_0 and μ with respect to the market data provided in Appendix A.1 (i.a. $\theta = \mu - \lambda\frac{\sigma}{\kappa}$). λ is an arbitrary fixed value.

HWV and HWExtV models are essentially the same (although, they do provide more flexibility). From previous sections, we know that the determinants of the modeled interest rate volatility in the Vasicek framework(s) are the parameters of κ and σ, respectively.

In figure 5.2(a), we consider an instructive example of the pricing of a single call option on a ZCB, as modeled in the Vasicek77 model. We notice that the price of the bond, for high levels of volatility, decreases in σ, whereas for low levels volatility the converse is in fact the case. This finding is clearly in conflict to Black's model, where it is commonly known that prices are strictly increasing in the level of volatility. As each of the four options in figure 5.2(a) are significantly in the money (as seen from the figure legend below figure 5.2), the Vasicek model essentially compensates, on very low levels of volatility, for the reduced probability of options expiring out of the money.

In figure 5.2(b)shows another opponent to Black's model, as the price of the ZCB option is well defined also for negative initial short rates. This finding is not surprising, as we upon each of the reviewed models found that the short rate is Gaussian and bonds prices are log normal, meaning that bond prices never become negative, but are well defined even for negative interest rates. Next, we consider the above described implications, when pricing European Caps (Floors).

Figure 5.3: The Volatility Skew - Implications on Caps (Floors). Parametrization: 7.1(d).



For this purpose, and without further introduction, we use the achieved parametrization results for the HWExtV model, as found in table 7.1(d).[29] Figure 5.3(a) depicts the implied Black76 volatilities from: the Market, the HWExtV model as well as from Black's model, with a constant volatility across strike levels (straight line). It is clearly seen that the HWExtV model does have capabilities of creating a certain degree of volatility skew. Furthermore, when transformed into prices in figure 5.3(b), we find a seemingly good performance by the model, while at the same time, the short comings of Black's model in its basic form are clearly noticed.

In conclusion, based on the theoretical knowledge developed through chapter 5, we know that the Vasicek framework(s) *are* capable of producing a certain degree of volatility skew. At the same time we know that the skew is created somewhat endogenously, with only few indirect parameters controlling the volatility (hence skew) generation in the model. As such, we have to believe that the few "levers“ available will be enough to calibrate the model properly to market prices across the volatility surface.

[29]The usage of the HWExtV model is without further contribution to the current context. In this respect, the dynamics of three models are similar and the choice were made, merely, for the sake of reducing the amount of somewhat repeated calculations.

Part II
Implementation and Calibration

Chapter 6

Outline of Market Data

The following sections contain the empirical and applied work, which is the main part of this thesis. The aim has been to apply the reviewed Hull-White Vasicek frameworks in such a way that it mimics a live setting. Although the result is far from the complexity and rigor faced by investment banks nowadays, it still provides insight on the basic concepts, and gives small glimpse of the amount of work, coordination and complexity provided by modern quant departments.

The empirical work is divided into three main sections: 1) Analytic pricing of standard products, 2) Simulation of the Extended Hull-White model, and 3) Two cases of simulation based pricing of selected complex derivatives.

6.1 Market Data and Source

All empirical work is based on data from the US market. The needed input data is (1) the zero-coupon yield curve represented by the discount function, $d(T)$ and (2) the Implied Flat Black76 Volatility grid, σ_{BS}. Several considerations were made before electing the data set. The quite restrictive dynamics of the Vasicek framework meant that we initially applied the Vasicek77 model to an yield curve environment feasible with its dynamics so to calibrate at least somewhat well to the observed yield curve. As discussed in section 5.1.4, the apparent US Swap curve in Q1-2 2012 in having a concave upwards sloping term structure in the 2-10Y segment was a severe example of the classical equilibrium Vasicek77 models short-comings, where the yield curve as "best-fit" was reproduced as a straight line(!) Obviously, the shape of the initial yield curve was no problem for the two arbitrage-free Hull-White Vasicek frameworks. However, the emerge of a significant tenor basis[1] and shift in market standards from uncollateralized Libor-based prices to collateralized over-night index swap [OIS]-discounted prices[2] meant that we would like to refrain from a collateral based multi-curve environment and restrict ourselves to the “older” Libor-based one-curve setting.

To this end, we have chosen a 30Y pre 2008-09 data set, with a neat upward sloping/positive yield curve, and [3M, 2Y, 10Y, 30Y] rates levels at around [2.50, 3.55, 4.50, 5.10%]. The volatility term structure has the usual “humped” shape, as discussed earlier, and the grid is represented by strike levels relative to ATMF to ease the graphical representation of the results. The data set is outlined in tabel (6.1) and the yield curve/discount factors and volatility grid are shown in figure 6.1(a)-(b). The actual numbers are provided in Appendix A. Usually, the market quotes

[1] The notion of tenor basis lies outside this thesis, but essentially covers the difference in credit risk inherent from lending at different maturities. Recall that the referenced index rate Libor, reflects the cost of uncollateralized borrowing between prime banks at **different maturities**. This means that a lender is exposed to different credit risks depending on the maturity. Market participants have adjusted their models from single-curve to multi-curve setups to account for the tenor basis emerging on the back of the financial crisis in 2008-09.

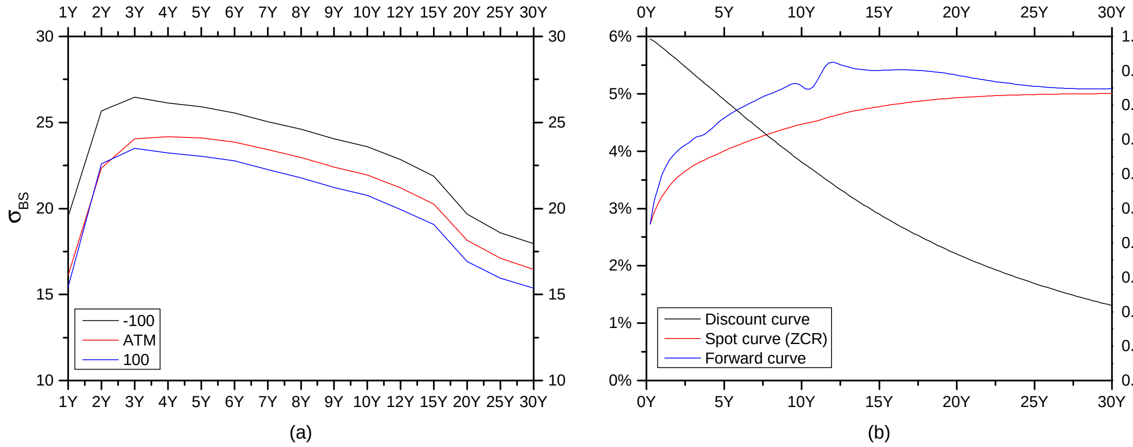
[2] The notion of OIS-discounting and Libor adjusted forwards lies outside this thesis, but essentially covers the fact that interbank OTC-trading today include daily exchange of cash collateral. Hence, the discounting (funding/lending) curve becomes the risk-free O/N rate, e.g., in EUR a Euro over-night index average [EONIA] curve.

Table 6.1: Market data - Discount factors extracted via >SWPM<, corresponding Black76 Implied Cap Volatilities extracted from >VCUB<. Source: Bloomberg™

Discount factors, $d(T)$	
Date Range	31-01-2005 to 31-01-2035
Maturity (3M intervals)	3M, 6M, 9M, 1Y,...,30Y
Tenor and Ccy	3M USD Libor
Black76 Cap Volatilities, σ_{BS} (Flat)	
Date	31-01-2005
Maturity	1Y, 2Y,..., 10Y, 12Y, 15Y, 20Y, 25Y, 30Y
Strike (50bp intervals)	ATM ±25, 50, 75, 100, 150,..., 300bp
Tenor and Ccy	3M USD Libor

in terms of yields rather than discount factors. Hence, typically we would need to construct the discount curve from yields of different traded instruments applying smoothing techniques between observations, such as Splines, while accounting for differences in compound conventions.[3] However, as the process of yield curve construction essentially is without contribution to the overall aim of this thesis, we have generated the discount factors with 3M intervals directly via Bloomberg™ using >SWPM< and applied simple linear interpolation between observations. While linear interpolation and shortage of data points might seem too serious an approximation in the short end, for longer (>1Y) maturities we generally have more observations. Overall, we have applied an approximation method to shortcut the target.

Figure 6.1: Graphical outline of the market data. Left (a) shows Black76-vol term structure at different absolute strike levels. Right (b) depicts the spot zero coupon rate [ZCR] curve, Forward (3M) curve and the Discount curve (d(T)) (r.a.).



It may well be argued that the selective approach in terms of application of historical market data is far from reasonable. However, the aim of this thesis is to achieve insight on means and concepts behind interest modeling, more than the actual price levels of the end results. In such a context, the use of historical market data should be of less concern.

[3]Usually, the curve is constructed from FRAs and Libor futures in the short end and Swap rates (1Y intervals) on longer maturities.

Chapter 7

Analytic Pricing of European Caps and Floors

Section 5.4 provided a setup for analytical pricing of Standard European Caps and Floors in the Vasicek and Hull-White descendant models. In this section, we apply the derived theory and scrutinize the abilities in fitting market prices across the volatility grid for the two Hull-White Vasicek frameworks. The achievement of a satisfactory fit to liquid market prices of standard products is essential to ensure sufficient amounts of confidence in the later application, when pricing products are not directly quoted in the market via numerical methods.

Provided the theoretical knowledge from section 5.4.3 we know that the models *is* capable of producing a certain degree of volatility skew. However, we also know that for our model-specifications the skew is created endogenously. Hence, we are left without exogenous parameters directly controlling the skew generation and have to put faith in that the models are able to adjust internally to market prices across the grid. Further, sections 5.2 and 5.3 lead us to expect that only the Hull-White Extended Vasicek specification will be able to fit the volatility "hump" apparent from the data set in figure 6.1(a).

7.1 The Notion of a Good Term Structure Model

Before turning to the actual calibration process, it is noteworthy to consider some of the features characterizing a "good" interest rate model.

First and foremost, such model clearly has to provide a sufficient fit of the market prices. Secondly, the model should have a reasonable stable time-development of its parameters so as to reduce the need for re-calibration. Even though this feature is highly desirable most interest rate models are unfortunately unstable, and thus, need frequent (daily) re-calibrations.[1,2] Thirdly, (and somewhat related to the second point) the model should produce accurate and robust results with low parameter sensitivities to changes in the underlying volatility grid, thereby ensuring the use of the models for hedging purposes.

The notion of parameter time-stability may be assessed by re-calibrating the model to a sequence of historical data. To comprehend the last issue on model robustness, ref [22, Chpt. 8] deals with the following two questions:

1. How sensitive are the model's parameters to changes in implied volatilities?
2. How sensitive is the value of the instrument being priced to small perturbations in the model parameters?

While time-stability analysis and computation of risk numbers ("Greeks") lie outside the scope

[1] [3].

[2] Even though frequent re-calibration is needed, small changes in parameters are still highly desirable.

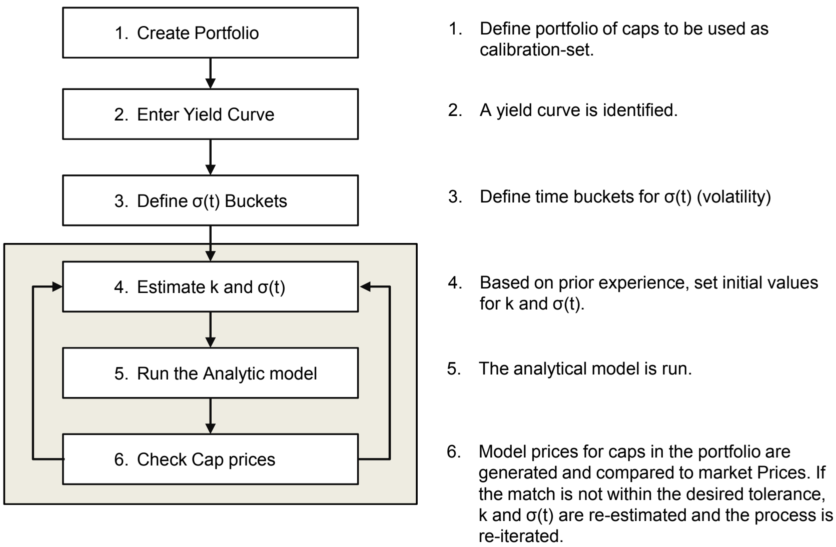
of this thesis, we touch upon the second question on model robustness while considering complex pricing in chapter 9.

7.1.1 The Calibration Process

The achievement of a good model is widely based on skills during the actual calibration process. To a large extent the work is empirical and such knowledge is accumulated via numerous iterations by professionals in the banking industry. As a consequence, a lot of the empirical know-how is literally concealed and only superficially represented in the literature as no-one have commercial interests in revealing their findings ("best bets"). The general calibration procedure, however, takes form as shown in figure 7.1.

First, a relevant portfolio of liquid instruments, sufficiently representing the current market conditions, is to be determined. Such portfolio is designated by the "calibration-set". In practice, and as earlier discussed, the short end is generally more liquid than for longer maturities and requires more instruments (data points) in order to capture the dynamics properly. The second step involves extracting the discount function from the current yield curve. Both steps one and two were addressed in section 6.1, where we, in summary, had 15 instruments across 17 strike levels now outlined as our calibration-set[3] and further a discount function represented by 30x4 extracted data points, quarterly spaced from 3M to 30Y.

Figure 7.1: Diagram of the Calibration Process



Inspired by findings in ref [21, p. 25-28], [23, p. 706] and [22, Chpt. 8].

Next, the decision on the total number of time buckets in the parameter vector, σ_t, is to be determined. As highlighted, e.g., by ref [1, chpt. 4.2.7] and [9, chpt. 28.8], it is important that the total number of volatility parameters, that is $(\sigma_i + \kappa)$ for $i = 1, 2, .., n$, does not exceed the total number of instruments in the calibration-set, in order to avoid over-fitting caused by too many degrees of freedom. However, at the same time we want enough flexibility in the model to ensure a meaningful fit of market prices. As such, no exact number exists and the "correct" answer must once again be based on heuristics. In the forth step, we have to set "reasonable" initial values for the parameter vector. At first hand this seems difficult; however, browsing other academic work[4] combined with own "trial-and-error" yields a good starting point. Fifth, the model is run using the

[3]We will later discuss whether individual data points in the calibration-set are properly chosen.
[4]E.g. ref [24] currently a Post-doc Research Fellow at the Faculty of Mathematics - University of Vienna.

inputs from steps two and three. The last step concludes the process by comparing the analytical derived model-prices to the observed market prices in calibration-set. This imply setting a norm or objective function which we shall consider in the following section. Via re-iteration over steps four through six, while minimizing the residuals, we continue the procedure until a sufficient fit is obtained.

Before discussing the results in section 7.2, we note that while the above procedure might seem easy at first hand, the process consumes significant amounts of time before having the setup to work in a meaningful way in practice.

7.1.2 The Calibration Problem

The sixth step of the calibration procedure outlined in the previous section involves setting a norm or objective function to determine the "Goodness-of-fit". Ref [5, p. 230] suggests a common approach recognized as a weighted least squares minimization problem

$$\min_{\Omega} S(\Omega) \;=\; \min_{\Omega} \sum_{i=1}^{N} \sum_{j=1}^{M} w_{ij} \left[C_{ij}^{\Omega}(K_i, \tau_j) - C_{ij}^{M}(K_i, \tau_j) \right]^2, \tag{7.1}$$

where Ω is the calibrated vector of parameters (thus $\Omega = (\kappa, \sigma_t)$), $C_{ij}^{\Omega}(K_i, \tau_j)$ is the model price and $C_{ij}^{M}(K_i, \tau_j)$ is the market price for the ij'th Cap with strike K_i and time to maturity τ_{ij}, respectively. M denotes the number of maturities and N the number of strike levels for a total number of calibrating instruments of $N \times M$. In certain situations it may be desirable to assign different weights to the minimization problem across the grid - a feature, which may be controlled by the factor, w_{ij} (though often $w_{ij} \equiv 1$).[5]

C.f. ref [23, p. 708] and [9, chpt. 28.8] suggests a slightly different approach adding penalties to the initial objective function to induce a "well-behaved" σ_t function with a sufficient level of smoothness[6]

$$\begin{aligned} \min_{\Omega} S(\Omega) \;&=\; \min_{\Omega} \sum_{i=1}^{N} \sum_{j=1}^{M} w_{ij} \left[C_{ij}^{\Omega}(K_i, \tau_j) - C_{ij}^{M}(K_i, \tau_j) \right]^2 \\ &+\; \min_{\Omega} \sum_{i=1}^{K} \alpha_i [\sigma_i - \sigma_{i-1}]^2 \\ &+\; \min_{\Omega} \sum_{i=1}^{K-1} \beta_i [\sigma_{i-1} + \sigma_{i+1} - 2\sigma_i]^2. \end{aligned} \tag{7.2}$$

The second term in (7.2) provides a penalty for large changes in σ_t, and the third term adds a penalty for high curvature in σ_t. Appropriate values for α_i and β_i are, once again, to be based on experience. For our scope, we use (7.2) and utilize different specifications for w_{ij}, α_i and β_i.

The minimization problem in (7.2) is complex in nature, as the function $S(\Omega)$ does not contain any particular shape or structure. To this end, most (all) optimization algorithms are only able to determine a local minima to the problem, and therefore the procedure often becomes highly sensitive to the initial parametrization.[7] Without prior knowledge on the direction of the solution ("qualified guess"), this may be approached by parameter space partitioning analysis. Each parameter is divided into g sub-values across the parameter space and the squared residual, $S(\Omega)$, are calculated upon each combination $S_s\left(\Omega_s(\kappa_j, \sigma_{i_j})\right)$, for $j = 1, 2, .., g$, $i = 1, 2, .., n$ and $s = 1, 2, .., (n+1)^g$.

[5] [5, p. 230].

[6] The notation used is slightly different from those used in the two references and further ref [9, chpt. 28.8] implicitly assumes that $w_{ij} \equiv 1$.

[7] We refer to ref [5, p. 231], which provides a small sketch on the issue in context. Moreover, ref [25] has an excellent exposition of the general case.

Subsequently, the, e.g., 5 best parametrizations are elected and evaluated in the following calibration process.

While the method is effective, it readily becomes a tedious task to implement as the number of different parametrizations grows exponentially. For 14 different model parameters (as we have in the later analysis) in the initial vector $\Omega(\kappa+\sigma_i)$ for $n=13$, and e.g. 5 individual sub-partitions per parameter, gives a total of $14^5 = 537.824$ different calculations of $S_s(\Omega_s)$(!) Having in mind that $S(\Omega)$, in itself, often consists of residuals across several tenors and strike levels, the computational time readily becomes vast. As a consequence, for the initial parametrization, we take a more heuristic approach based on available references and introduce reasonable boundary conditions for each parameter space and number of partitions.

Similarly, the minimization problem in (7.2) may be reduced by application of a strong optimization algorithm. The Solver functionality in Excel uses a generalized reduced gradient decent (multi-derivative) algorithm for optimization of nonlinear problems,[8] which, in general, is inferior to the "non-derivative" Nelder-Mead downhill simplex algorithm [NMA] - often thought to be the best general purpose algorithm according to ref [26, p. 11]. When minimizing sum of squares of "nonlinear multi-variable multi-valued functions"[9] the Levenberg-Marquardt algorithm [LMA], as suggested by ref [9, p. 672], has become a standard. The method uses interpolation between a Gauss-Newton algorithm and the method of gradient descent, which in practice works extremely well with a good global convergence property.[10] Both the ND and LMA algorithms are, i.a., built-in to MatLab. A further review of the underlying methods behind the algorithms are beyond the scope of this thesis but we note that even though LMA is a very powerful least squares optimizer it is still not flawless and the risk of converging to a local minima is still present.

The author did not have any prior experience with either NMA nor LMA but the "Foxes Team Community" [26] provided a very recommendable and (somewhat) easy to comprehend exposition of the methods. similarly, Foxes Team also provided an Excel add-in implementation, ref [25], which contained a range of different optimization algorithms not readily available elsewhere. The ND optimization tool were easy to setup (essentially the same as the Excel Solver), whereas the LMA non-linear regression setup required extra work before application.

Unfortunately, the add-in showed extremely poor performance in Excel 2007[11] (5 single iterations of the model versus 9 ATM caps took +45mins) and only handled up to 10 input variables at a time, which for our (later) specification of the model meant that we were only able to calibrate the model versus caps with up to 9Ys maturity. An alternative LMA implementation for VBA was found with the russian company *ALGLIB Project* [27]. ALGLIB is a cross-platform numerical analysis and data processing tool-kit with support for VBA. The tool comes in library format (102 separate .bas files) with no user interface nor installation routine[12] and working with the source code it quickly became clear that the time needed for proper implementation of the library were far beyond the scope of this thesis. As a consequence, we (reluctantly) settled with the Generalized Reduced Gradient [GRG] algorithm and the Excel Solver functionality for the remainder of the thesis.

[8]We refer to `http://support.microsoft.com/kb/214115` (23.09.2012).

[9]That is, several equations based on the same set of variables with the aim to minimize all of the equations simultaneously. I.e. not necessarily the minimum of each individual function, but the set of variables that provides a minimum of the sum of the functions.

[10][26, p. 12].

[11]This may i.a. reside from two things 1) the Optimiz-tool is only optimized from Windows 2000/XP, Excel 2003/XP and VB6 - a warning is given for Vista/Excel 2007+ users and 2) our rather large setup which already includes extensive use of UDF via VBA.

[12]For reference, a small third party installation script can be found at: `http://newtonExcelbach.wordpress.com/2010/05/20/installing-alglib-with-Excel-vba/` (23.09.2012).

7.1.3 Implementation and Source Code

The source code for the calibration routine are provided in Appendix C.3 and we briefly describe each of the developed components in the process flow, as outlined in figure 7.1. Individual VBA components are designated in text format `courier` by their module and subroutine in capital letters, e.g. `module/SUBROUTINE`.

To estimate κ and σ_t in step 4 in figure 7.1, we compare market prices of the calibration-set to the modeled prices of the (HWV) HWExtV setups. To establish the modeled prices we use subroutine of `hull_white/CAPFLOORREPHW`. The routine implements the replication strategy for European Caps (Floors), as described in section 5.4.2. In its implementation, we draw further upon the results in section 5.4.1 to establish the underlying option prices of the ZCBs, which is processed in subroutine `hull_white/OPTZCBHW`. As a natural sub-component of this calculation is the ZCB prices themselves, as outlined in the two (identical) eqs (5.60) and (5.66), which requires the calculations of $\beta(\tau)$ in eq (5.44) as well as $\phi(t)$ in eqs (5.52) and (5.67). The calculation of $\beta(\tau)$ is contained in `hull_white/BHW` and the calculations of $\phi(t)$ is constituted in `hull_white/PHIHWV` and `hull_white/PHIHWExtV`.[13] To back out the implied Black76 volatilities from each of the Hull-White frameworks, we use the binary search algorithm `mods/IMPLVOL`, as provided in Appendix C.4.11. During the iterative process in figure 7.1, steps 4-6, we recalculate and compare the modeled and market prices of the instruments in the calibration-set, using the objective function in eq (7.2) as well as the Excel solver functionality, to find the optimal parametrization.

7.2 Results and Performance

This section gives an exposition of the achieved results during the calibration process. It is noteworthy that the actual work contained many more iterations and different specification as is shown here, and consumed significant amounts of work before reaching meaningful results.

All calibrations have been performed versus the calibration-set of European Caps (Floors) described in Section 6.1 using the augmented objective function from eq (7.2). Results for both the HWV model and the HWExtV model are provided, however, we quickly focus on the HWExtV version to enable fit to the volatility hump, as depicted in figure 6.1(a). Even though we persistently optimize over different parametrizations of eq (7.2) we rigorously compare *across* models and model-specifications by their non-weighted least square [NWLS] Goodness-of-fit measure. NWLS values are provided for each ATMF, in the money [ITM]/out of the money [OTM][14] and the Full Grid in table 7.1. Further, a graphical inspection of selected results in terms of prices and implied Black76 volatilities are provided at key point through and across the volatility surface. Further, a graphical inspection of the resulting parametrization vector of, σ_i is provided.

Concluding the section we show a graphical comparison of our "best bet" full-grid calibration parametrization for the HWExtV model versus the market in terms of implied Black76 volatilities as well as a table format comparison of nominal price deviations for a 5Y 10m$ notional European Cap.

Table 7.1 collects our calibration results. 7.1(a) show the result for the HWV model calibrated versus ATMF. We note that the objective function is equally weighted and in this case has no further adjustments (that is $\omega_{i,j}=1$ and α_i, $\beta_i=0$). Both calibrated model parameters, κ and σ, lie inside reasonable levels, with a projected risk-neutral mean-reversion speed of approximately 5.75% and a fixed σ value of 1.15%. NWLS values suggests a somewhat reasonable fit to ATMF, which clearly is reduced when measured versus the full grid. Figure 7.2(a)-(b) illustrates the achieved results in terms of Black76 implied volatilities and Cap (Floor) prices - market versus model.

[13]In Appendix C.3 we only provide the scripts for the HWExtV model as they are very close to the results of the HWV model. For example, it requires very little additional setting up to use the routine of `hull_white/PHIHWExtV` as substitute for the `hull_white/PHIHWV`.

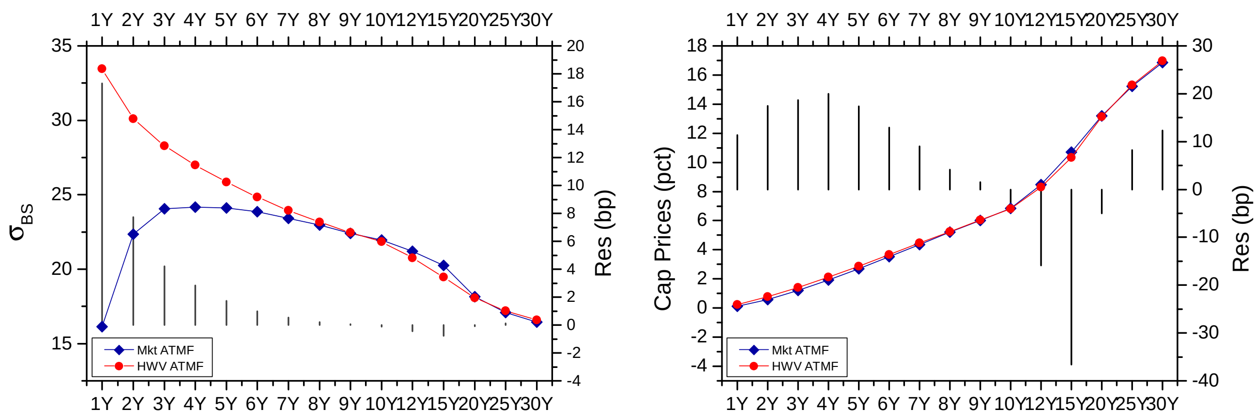
[14]Note that both ITM and OTM values includes ATMF.

Table 7.1: Calibrated model parametrization †, objective function weights and Goodness-of-fit *non-weighted* least square values for ATMF, ITMF, OTMF and the *full grid* (percentage prices). Highlighted cells denote the calibration objective criterion ††. Initial parametrization; $\kappa \equiv 0.05$, $\sigma_i \equiv 0.01$ across all entries †††

	Parameters		Weights			Goodness-of-fit			
Results	κ	$\frac{1}{K}\sum_{i=1}^{K}\sigma_i$	w_{ij}	α_i	β_i	*amtf*	*fg*	*ITM*	*OTM*
a (HWV)	0.0577	0.0115	1	-	-	3.603E-5	4.488E-3	1.101E-3	3.423E-3
b (HWExtV)	0.0570	0.0109	1	-	-	7.347E-7	3.378E-3	7.431E-4	2.635E-3
c (HWExtV)	0.2574	0.0186	1	0.0050	0.0050	5.509E-5	2.697E-3	1.159E-3	1.593E-3
d (HWExtV)	0.0894	0.0131	$mat_j^{-4}\cdot 13.86$	0.0050	0.0050	2.344E-5	3.249E-3	1.313E-3	1.959E-3

†Full outline of $\sigma(i)$ provided in figure 7.4. ††Even though columns 7-10 show NWLS values, we remind that the applied objective function is (7.2) with the specified weights in columns 4-6. †††Initial parametrization found by partitioning space analysis in the HWV model and extrapolated to the HWExtV model. $\kappa_j = \{0.05, 0.10, 0.20, 0, 40\}$ and $\sigma_j = \{0.01, 0.02, 0.03, 0.04\}$. σ-partitioning wrapped around results proposed by [24, p. 53] over a similar period.

Figure 7.2: HWV model - Calibration results market vs model (ATMF). R.h.a. shows residuals in level form. Parametrization: Table 7.1(a).



(a) Black76 Implied Volatilities (b) Cap (Floor) prices

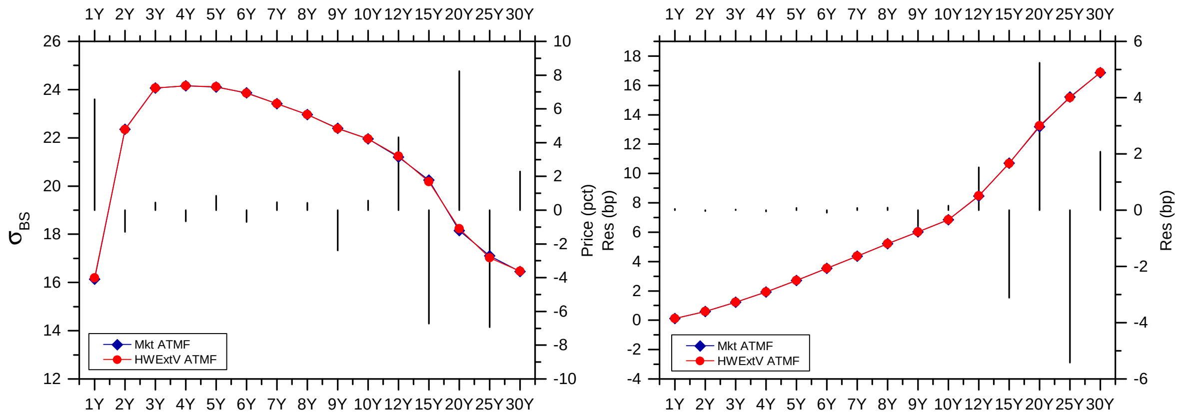
In figure 7.2(a) we clearly verify that the HWV model has distinct problems in fitting the volatility hump in the short end of the curve. This is caused by the induced exponential decay of the forward rate volatilities, as discussed in section 5.2.2. For longer maturities, the model seems to be more in-line with the market, although, dynamics around the 15Y point are not fully captured. This might be due to 1) lack of flexibility in the model or 2) a mispricing in the market due to potential illiquidity around the 15Y point (We will revert to this particular issue in the later calibrations). Translated into prices (figure 7.2(b)) the lack of fit in term of volatilities in the short end is of less importance.[15] This is mainly due to the short option periods, i.e. the lower relative sensitivity toward changes in the volatility grid. However, from a pricing precision point of view, bearing in mind the actual price levels, the percentage errors committed in the short end are far more severe. As a consequence of the lack of flexibility in the volatility specification, we skip the HWV model and conduct all subsequent analysis in the HWExtV model.

Table 7.1(b) shows the results from the HWExtV model calibrated versus ATMF. Once again, the objective function is equally weighted and has no further adjustments. The HWExtV model clearly benefits from the introduced flexibility in the parameter vector of, σ_i, as verified by the significant reduction in NWLS values, across all sections of the grid. This is confirmed in figure 7.3(a), where the cap hump now is neatly fitted and only minor deviations is found. In 7.3(b)

[15]E.g. comparing deviations for the 1Y to 4Y point vs the 15Y point of -40 bp.

we see that the committed price errors is of $<\pm 6$ bp and that such deviations is in fact only the case in the very long end. The calibrated parameter, κ, is once again within expected levels, with a projected mean-reversion around 5.75%. While the average parameter vector, $\bar{\sigma}_i$, is at an

Figure 7.3: HWExtV model - Calibration results market vs model (ATMF). R.h.a. shows residuals in level form. Parametrization: Table 7.1(b).



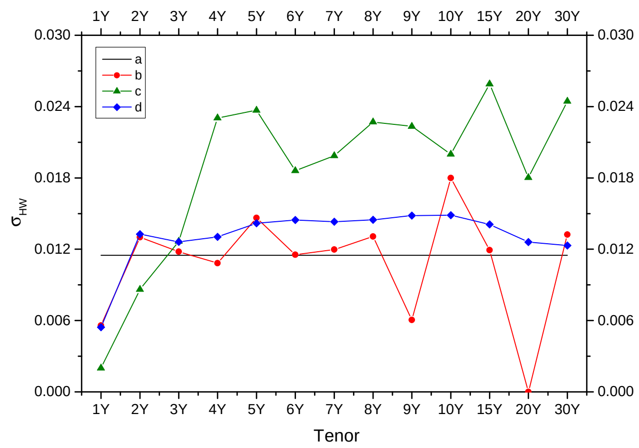
(a) Black76 Implied Volatilities

(b) Cap (Floor) prices

immediate comparable level to the (fixed/constant) σ-level in the HWV model, the number does cover over a significant variation across time-buckets, as depicted in figure 7.4. Especially the 20Y bucket very close to zero calls upon further attention. Comparing figures 7.2(a) and 7.3(a), indicates that the low σ_i value in the 20Y bucket corresponds to the HWExtV models increased ability/flexibility to compensate for the sudden drop in market volatilities on that particular part of the volatility term structure curve shifting between the 15-20Y points. As long as each entity in the σ_i-vector is reasonably stable over time (cross daily re-calibrations) the variation is acceptable and simply represents the best compromise for the model to fit the market. However, provisional risk of "mispricing" on various points on the curve, potentially caused by illiquidity, may encourage the introduction of a light inertia in large jumps and high curvature in the σ_i vector. This may be done to stabilize the model, the generated prices and the associated risk numbers, as we discussed in section 7.1.2. Another way to deal with the issue is to reduce the number of entities/buckets in the σ_i vector thereby essentially letting the model "average out" the variation over an extended time-bucket. Such approach would typically be followed if the "troubled" point on the curve is thought to be illiquid in general, which would cause an over-specified model as discussed in section 7.1.1. As time stability analysis is outside the scope of this thesis, we have chosen to follow the former explanation of a temporary slightly illiquid 15Y point, which we in general would want to include in our calibration as a significant point on the curve. Accordingly, in the following analysis, we introduce jump and curvature penalty weights, that is α_i, $\beta_i > 0$. As the initial results for the HWExtV model performed well versus ATMF, we now turn ourselves toward the calibration of the full grid. The results from an augmented calibration, including the full grid to our objective function, is represented in table 7.1(c). First we note, that the full grid NWLS value has decreased at the expense of the ATMF fit. The full grid improvement is driven by a significant improvement for OTM options whereas the fit on ITM options and ATMF in fact both decreases.[16] As noticed in figure (7.4) the introduction of jump- and curvature penalties (weights found by repeated iterations) seems to have induced less variation across the parameter vector σ_i at least in the long end, although the effect is not too convincing. Moreover, we note that the entries in the short end of the vector markedly have decreased whereas conversely, the long end exhibits significant increases, causing

[16] The general coherence of the NWLS numbers in table 7.1 may be verified for each calibration, by adding the values for ITM and OTM options while subtracting the value of the ATMF. This result must correspond to the NWLS value for the full grid, for the numbers to be internally consistent.

Figure 7.4: Outline and cross-model comparison of the parameter vector of σ_i.



an overall increase to the average $\bar{\sigma}_i$ level. Dubiously, κ is found to be >25%, which from an empirical point of view seems to be excessive. The elevated mean reversion may partly be offset by the increased average σ_i level (Recall that $\kappa\uparrow$ reduces the volatility in the model by inducing a higher degree of mean reversion in the short rate, whereas an average $\bar{\sigma}_i\uparrow$ obviously increases the modelled volatility), nonetheless a 25% mean reversion of all "chocks" to the short rate is likely to suggests mis-specification of the model more than being in coherence with real market behavior.[17] A review of the figures 7.5(a)-(b) additional adds to the suspicion of mis-specification as we reveal clear issues with the calibration. The volatility term structure is significantly mis-represented on shorter maturities translating into severe negative price deviations in the short end of the curve (by more than 40 bp!). Consequently, not satisfied with the achieved parametrization, we discard completely the results found in table 7.1(c) and a solution is sought.

To alleviate the problems occurring in the short end of the curve in the previous full grid calibration, a range of different measures were examined. In essence we wanted to compensate in our objective function, eq (7.2), for the increased amount of weight implicitly being put on longer dated instruments due to the introduction of the full grid. The bias of the non-weighted calibration essentially occurs due to the fact that modelled nominal price deviations for the more expensive longer dated instruments are, for obvious reasons, often much higher compared to the corresponding short end of the curve. This fact, causes the algorithm to favor an optimization of residuals originating from the longer end of the curve, as here the total squared residuals are reduced the most. Our work ultimately led to the following two amendments; 1) jump/curvature penalties were relaxed in the "ultra short" end ($\leq$1Y) to allow for enough flexibility in the parameter vector, σ_i, to match the steep drop in volatilities in the front.[18] Next, 2) a functional expression for w_j across maturities (and constant across strike levels) was developed to generally increase the impact of price deviations on shorter maturities during the calibration process. That is

$$w_j = \tau_j^{-x} \cdot sf \qquad\qquad sf = M\left(\sum_{j=1}^{M} \tau^{-x}\right)^{-1}, \tag{7.3}$$

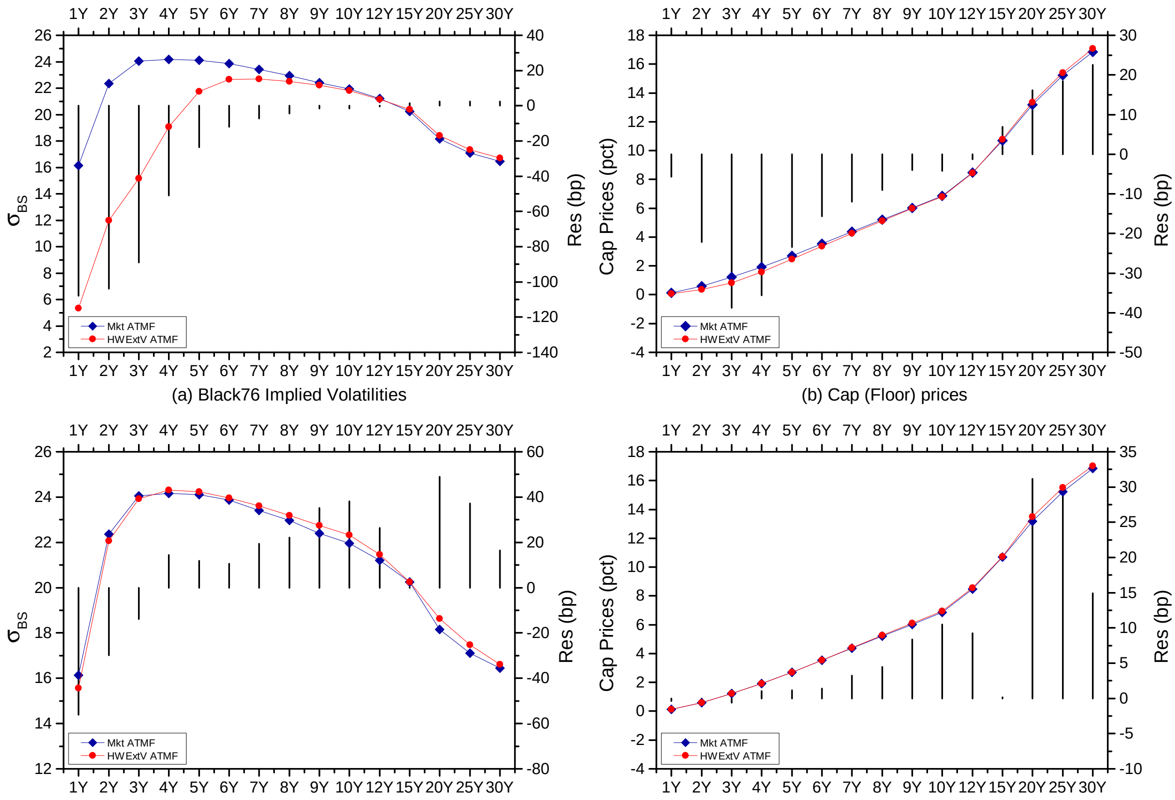
where τ_j denotes the maturity in years of the j maturing cap, M denotes the total number of maturities in the calibrating portfolio, sf denotes a scale factor ensuring that the sum of weights $\sum_{j=1}^{M} w_j$ are kept constant (=M) to reduce any interference with the jump/curvature corrections, while $\bar{x}$ is a fixed exogenous input variable which determines the "*force of front-correction*". We

[17] We note that as the calibrating portfolio does only contain caps, we lack information from the swaption grid in order to properly control κ. However, since we in this thesis focus on Caps (Floors), we will not discuss this issue further, but only mention its existence in passing.

[18] That is, the two last terms in eq (7.2) were redefined as $\sum_{i=2}^{K} \alpha_i[\sigma_i - \sigma_{i-1}]^2 + \sum_{i=2}^{K-1} \beta_i[\sigma_{i-1} + \sigma_{i+1} - 2\sigma_i]^2$.

note that eq (7.3) is an fixed-scaled exponentially decreasing function, where $\bar{x} \uparrow$ increases the weight allocated to the front end of the curve and conversely.

Figure 7.5: Calibration results market vs the HWExtV model(ATMF). R.h.a. shows residuals in level form. Parametrizations: First row stems from results in table 7.1(c). Second row stems from results in table and 7.1(d).



(a) Black76 Implied Volatilities
(b) Cap (Floor) prices
(c) Black76 Implied Volatilities
(d) Cap (Floor) prices

Table 7.1(d) shows the results for the *front corrected* full grid calibration. The mean reversion speed, κ, is found slightly higher than previously at 8.94%, which however still, is much more in line with real market behavior than the discarded results in table 7.1(c). Further, we find that the slightly elevated mean reversion level is similarly reflected/partly offset by a small but noticeable increase in the average parameter vector, $\bar{\sigma_i}$. A review of figure 7.4 shows a high level of smoothness across entries/buckets in the parameter vector, σ_i, which almost resemble the development of the results found in Table 7.1(b), though with a much lower variation across the far end buckets. The relaxed jump/curvature correction in the front has enabled the parameter vector of σ_i to adjust accordingly, lifting the “ultra short” end volatilities to a more reasonable level. The full grid NWLS value has further improved compared to the result found in table 7.1(b). Once again, the improvement has taken place at the expense of the ATMF and we do find a noticeable increase in price residuals, when comparing across figures 7.3(b)-7.5(d) against the HWExtV model calibrated versus ATMF only. However, the achieved NWLS value for ATMF lies still at a comfortable level well below the result gained in basic HWV model from table 7.1(a). By a further look at the sub components, the full grid improvement is again driven by OTM options while both ITM- and (as described) ATMF NWLS values are slightly reduced. Comparing figure 7.5(c)-(d) with 7.2(a)-(b) verifies that the parametrization in table 7.1(d) does indeed provide a significant improvement of the calibration results over those of the basic HWV model in table 7.1(b). The calibrated ATMF volatility term structure in figure 7.5(c) is now much more in line with market volatilities, where only a slight undershoot in the front of the curve combined with converse slight overshoot in the long end is tracable. Translated into prices in figure 7.5(d) the fit is significantly improved on maturities $\leq$ 10Y, whereas the long end shows slightly higher deviations as should be expected due the introduced “front end correction function”.

Satisfied with the results against ATMF, we progress with an examination of the modelled skew characteristics. Figure 7.6 depicts market versus modelled skew characteristics in terms of implied volatilities and prices at selected key-maturities (1Y, 5Y, 10Y and 30Y). We notice several characteristics (left column); First, it is clear that the calibrated model does not offer enough flexibility to provide a fully sufficient fit of the market volatilities and we find that differences of varying magnitude do occur across the grid. However, a further look at the associated residuals reveal that the HWExtV model does in fact seem to resemble the market skew dynamics quite fairly - as long we are as not moving too far away from ATMF. Next, we note that the deviations seems to be increasing on both sides of the skew, essentially balancing the calibration in such way that levels closely tied to the ATMF delivers the best results. A general remark is, that the HWExtV modelled skew generally decreases at a *faster* pace than compared to the market, regardless of the initially projected volatility level, a finding which is clearly verified looking at the l.h.s. of the skew in all four left-column figures in 7.6. Additionally, we observe that the modelled skew is *strictly decreasing* in strike level causing a persistent downwards sloping expression of the volatility term structure. This finding suggests that the HWExtV model might calibrate well in periods when markets exhibit a volatility “smirk” but that it is not capable of fitting the market when “smile” characteristics are present.[19] This finding is supported by figure 7.6(a) where the volatility skew in the front end actually expresses a slight “smile” shape. The HWExtV model completely fails in resembling the increase in volatilities from moving far OTM and does essentially just projects a straight downwards-sloping line. Figure 7.6 (right column) shows the associated results in terms of prices. Generally, the modelled prices fit our market date rather well and as expected ATMF levels in general provide the best fit. We note that the price discrepancies, when moving away from ATMF, are more pronounced in the long end, which is naturally explained by the higher sensitivity toward changes in the volatility (Vega) on longer dated instruments. As an example, the lack of precision in the short end (1Y) in figure 7.6(a) is hardly noticed when translating things into prices in figure 7.6(b). Further, we note that the HWExtV model tends to overestimate prices of ITM options while underestimate OTM options, which is equivalent of the previous argument, namely that the HWExtV modelled volatility skew tends to decrease at a faster pace than the market.

Figure 7.7 depicts the Black76 implied volatility surface produced by the HWExtV model plotted against the current market surface. The intersection of the two surfaces forms a line very closely tied to ATMF as expected, and we are still able to recognize some disturbances around the 15-20Y points at the very back end of the of the surfaces crossing. Most importantly, the distinct over-estimation on ITM options, combined with smaller but consistent under-estimation on OTM options is seen clearly in this setting, as well as its strictly decreasing shape. At the end of the day, we must though admit that the HWExtV model does capture the basic properties of the current market volatility surface. Both in terms of a smirk but also the humped shape in the front as discussed several in section 4. To further highlight and concretize the modelled price performance, table 7.2 shows the model against market prices of a 10m$ notional 5Y Cap at various strike levels. We note that the best fit is provided ATMF as expected with a price difference of 1.178,74$ / 0,44%. Compared to an absolute level price of 270.034,70$ this is quite acceptable in practice. The largest price differences both in nominal and percentage terms are on far OTM options. In nominal terms @250 shows the largest price difference of -14.149,36$ / -30,18%. The high percentage deviation is caused by the low prices on far OTM options due to their low exercise probability. For deep ITM options, the largest deviation is @-200 bt 10.303,19$ / 1,16%. In conclusion we find that as long as we are pricing options not too far away from ATMF, the HWExtV model does provide a fairly good performance.

[19]We did try to re-calibrate the model several times against a constructed and pronounced volatility smile, to see whether we were able to create/“force” the model to produce a “smile”. However, we were not able to succeed our efforts.

Figure 7.6: Skew characteristics - Calibration results market vs HWExtV model. The graph shows the achieved skew characteristics at selected maturities: 1Y, 5Y, 10Y and 30Y). Left column, sub-figures (a/c/e/g) shows the Black76 implied volatility skew. Right column, sub-figures (b/d/f/h) shows the Cap price across strike levels. Generally, r.h.a. shows residuals in level form. Parametrization: Table 7.1(d).

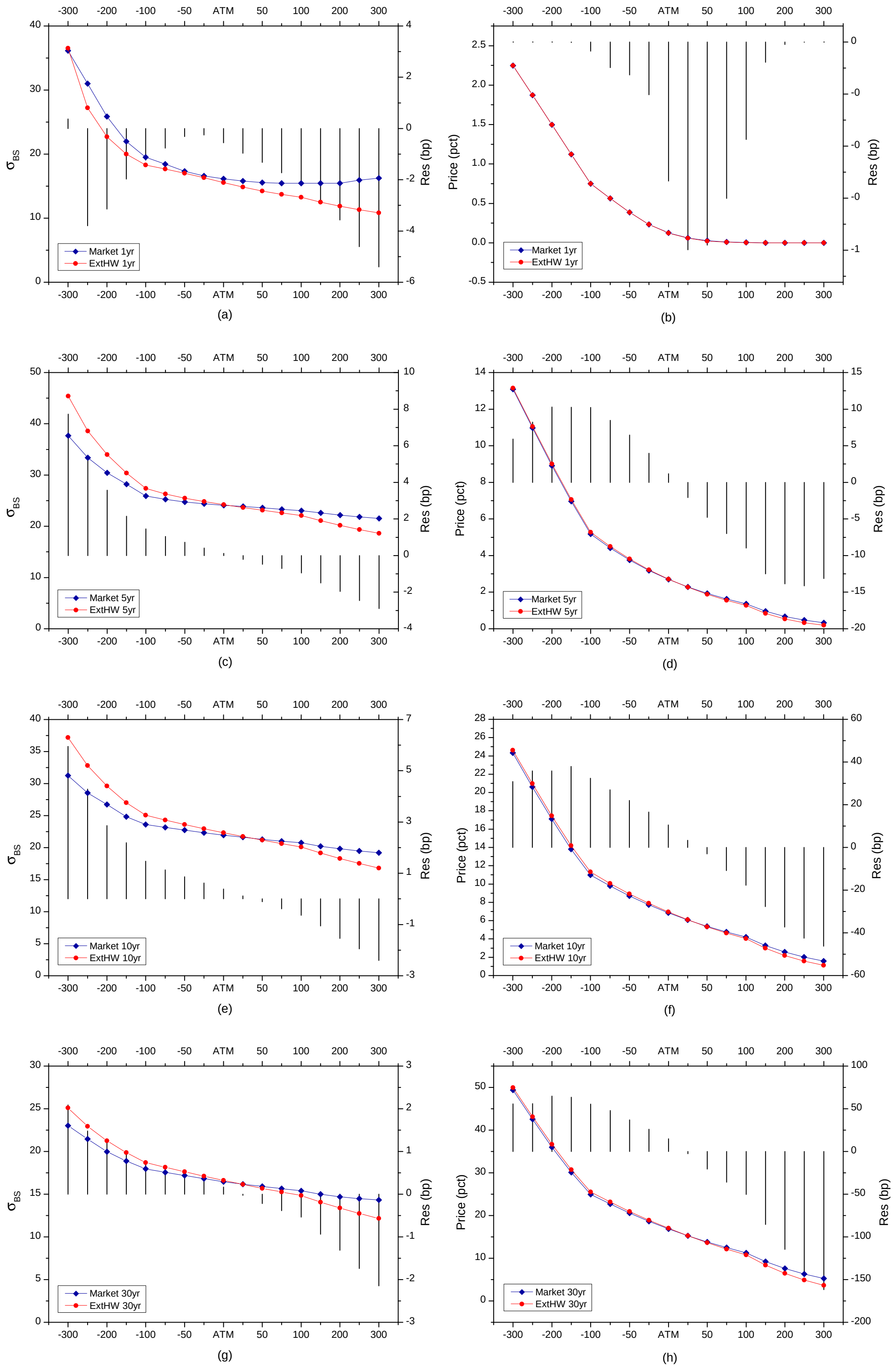


Figure 7.7: Black76 Implied Volatility Grid - Market versus the HWExtV model. The dark brown area shows the currect market Black76 implied volatility surface. The yellow/red/brown area shows the corresponding HWExtV modelled volatility surface. Parametrization: Table 7.1(d).

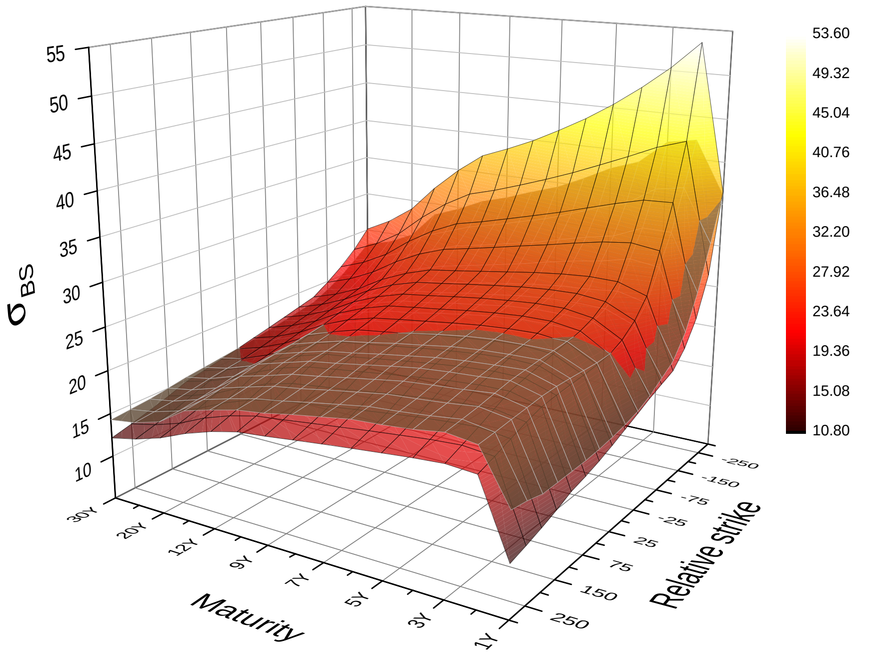
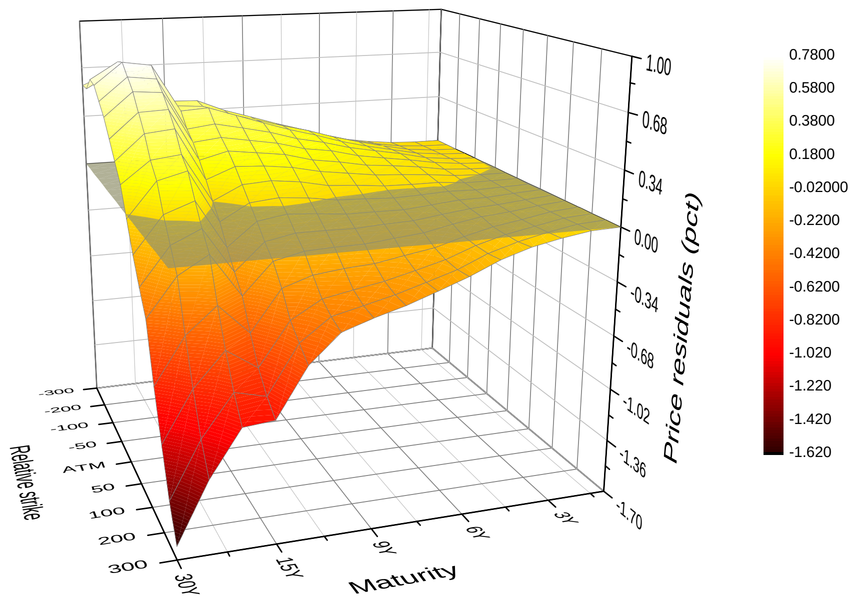


Figure 7.8: Price Residuals (Grid) - Market versus the HWExtV model. The glowing area shows the price residuals HWExtV minus Market. The shaded area indicates the zero line. Parametrization: Table 7.1(d).



7.3 Conclusion

Albeit the HWExtV model has shown excellent performance compared to the standard market models in section 4 it still has several short-comings in fitting the market, which is significant the farer we move away from ATMF. This is caused by its simple one-factor specification and lack of flexibility in the model-specification as we need extra “handles” to properly control the shape of the volatility surface. The biggest issues in percentage terms are found on far OTM options,

Table 7.2: Price comparison across strike levels of a 5Y 10m$ European Cap - Calibration results: market vs the HWExtV. Parametrization: Table 7.1(d).

Strike	Market price	HWExtV price	Difference $	Difference %
-300	1.308.798,65	1.314.712,86	5.914,22	0,45
-250	1.096.820,35	1.105.055,53	8.235,18	0,75
-200	890.774,45	901.077,65	10.303,19	1,16
-150	695.889,72	706.150,48	10.260,76	1,47
-100	517.469,11	527.700,80	10.231,70	1,98
-75	441.438,37	449.915,48	8.477,11	1,92
-50	375.269,01	381.745,83	6.476,82	1,73
-25	318.584,46	322.549,30	3.964,85	1,24
ATM	270.034,70	271.213,44	1.178,74	0,44
25	228.806,69	226.736,62	-2.070,07	-0,90
50	193.070,77	188.305,50	-4.765,27	-2,47
75	162.282,19	155.278,77	-7.003,42	-4,32
100	136.074,40	127.110,41	-8.963,99	-6,59
150	95.798,61	83.312,95	-12.485,65	-13,03
200	66.860,62	53.010,48	-13.850,14	-20,71
250	46.881,50	32.732,15	-14.149,36	-30,18
300	32.725,48	19.605,80	-13.119,69	-40,09

due to the strictly decreasing volatility structure of the HWExtV model. In a nutshell this is a significant drawback as more than 80-85% of all caps (Floors) are traded at OTM levels.[20] One way to circumvent the issue and improve OTM performance is to include only OTM options to the calibration-set or, even more specific, only options at the particular strike level in question. While this certainly improves the performance and in practice very well may be used it similarly reduces the generality of the model.

In the case sections in chapter 9, we revisit these issues as we will be conducting prices on both sides of the grid.

[20] Own experiences from Nordea Markets.

Chapter 8

Simulation of the Hull-White Extended Vasicek model

Closed-form solutions as the ones developed in section 5.4 are superior in terms of pricing precision and speed. However, for more complex structures closed-form solutions have not been derived why we are left with numerical procedures for their price determination.

This chapter introduces the concept of Monte Carlo simulation as means of pricing tool for the price determination of more advanced structures in section 9. First, the Monte Carlo setup is reviewed. Secondly, the the short rate SDE under the Hull-White Extended Vasicek framework is re-introduced and discretized, using a standard Euler discretization scheme. Thirdly, we review our practical implementation of the numerical procedure as source code in VBA - a piece of work which showed to be significantly time consuming and to contain several important considerations with regards to the transformation of theory into source code and a flexible user interface. We end this section by testing the performance of the developed tool when pricing standard European Caps/Floor so as to ensure sufficient confidence in the modelled results before progressing with the application of the model, to more advanced structures in chapter 9.

8.1 Monte Carlo Simulation

Monte Carlo simulation and risk-neutral valuation are powerful techniques to price interest rate derivatives. We follow eq (2.14) and the general approach for pricing of contingent claims described in sub-section 2.1.1. Applied to the case of an interest rate Cap with a strike of k and a present value of $Cp(r_s)$ we have

$$\widehat{Cp}(r) = E_t^Q\left[\int_t^T c(r)e^{-\int_t^s r_u du} ds\right] \qquad c(r) = (r_s - k)^+, \tag{8.1}$$

where r_s is the integral of the short rate, r. In essence, the price of the Cap is determined by taking the risk neutral expectation of its future payoffs; that is, using the risk neutral SDE of the Extended Hull White model we simulate paths of the future stochastic short rate, r_s, integrate over each individual tenor period to get the (stochastic) Libor rates, determine all intermediate payoffs $c(r)$ from each individual Caplet (Floorlet) in the Cap (Floor) and then finally discount all cash flows back to present by the average path specific short rate.[1]

While on standard European options clearly it would not make sense to implement numerical procedures, the technique however becomes advantageous, for a range of complex products where the payoff e.g. depends partly or exclusively on the price path followed by the underlying in reaching exercise or expiration. To be specific Asian (average price or rate) options, Look-back options, and certain types of Barrier options (e.g. down-and-out/up-and-in puts and calls) are all examples of

[1] We refer to e.g. ref [5, chpt 10.5] for a thorough stepwise exposition of the procedure.

products which contains path dependencies. Another example is a Sticky Ratchet Cap (Floor), where the strike of each individual Caplet (Floorlet) depends on the maximum (minimum) Libor for all preceding fixing periods.

8.1.1 Euler Discretization and Numerical Precision

Reconsider the SDE of the HWExtV model under Q. From eq (5.35) we have

$$dr_t = \kappa\big(\theta(t) - r_t\big)dt + \sigma dW_t^Q. \tag{8.2}$$

Using the described alternative representation of the level form process of the short rate presented in eq (5.63)-(5.65), we have

$$\begin{aligned} r_t &= m(t) + x_t \qquad x_0 = 0 \\ m(t) &= r_0 e^{-\kappa t} + \int_0^t e^{-\kappa(t-s)}\kappa\theta(s)ds \\ dx_t &= -\kappa x_t dt + \sigma(t) dW_t^Q. \end{aligned}$$

A first-order Euler Scheme[2] consists of approximating the above integrals via discretization.[3] Integration between t and $t + \Delta t$ yields

$$\widehat{r}_{t+\Delta t} = m_{t+\Delta t} + x_{t+\Delta t} \tag{8.3}$$

$$m_{t+\Delta t} = m_t + (r_0 e^{-\kappa(t+\Delta t)} - r_0 e^{-\kappa t}) + e^{-\kappa(\Delta t)}\kappa\theta(t)\Delta t \tag{8.4}$$

$$x_{t+\Delta t} = x_t - \kappa x_t \Delta t + \sigma(t)\,(W_{t+\Delta t}^Q - W_t^Q). \tag{8.5}$$

Applied iteratively for a given set of ts says

$$t = \{t_0, t_1, t_2, ..., t_m\} \quad \text{where } t_0 = 0 \text{ and } t_m = T,$$

provides a discretized approximation $\widehat{r}_t$ of the solution to r_t of the above SDE. By replacing the increments $(W_{t+\Delta t}^Q - W_t^Q)$ in eq (8.5), by $Z\sqrt{\Delta t}$[4] the implementation of the Euler scheme is straight forward. Consider the general discretized form of eq (8.1)

$$\widehat{Cp}(r) = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{k=0}^{\tau_k = T} c^i(r_{\tau_k})\Delta\tau_k \, \exp\left(-\sum_{j=0}^{t_j=\tau_k} r_{t_j}^i \Delta t_j\right)\right). \tag{8.6}$$

In this general form, index i controls the number of simulations, index k controls the time span between each payoff (i.e. Caplet) and index j is the step size of the underlying short rate process, where for obvious reasons $j = k$ for each k. As a remark, note that the expression further allows for varying increments in both τ and t.

Before implementation of the above procedure, we note that the described setup induces two types of uncertainties, formally known as simulation errors and discretization errors, respectively.[5] Looking at the former, we note that as the number of sample paths, N, is finite, $\bar{C}p_t(r)$ will inevitably be subject to some degree of randomness. Next, the discretization of the time period $[0; T]$ introduces several sources of errors:

[2] [4, p. 108-112].
[3] [1, chpt C.3].
[4] Where $Z \sim N(0; 1)$ provides a one dimensional vector of independent Standard Gaussian samples.
[5] [5, chpt. 10.5].

1. The approximation of integrals (both the overall discounting and the 3M Libor estimates) by a left sum will cause some degree of disturbance. E.g., for a monotonically increasing function, a left sum approximation will undershoot the value of the integral.[6]
2. The recursive calculation of $\widehat{r}_{t_j}^{\,i}$ in (8.3) means that committed local errors (errors per increment) are accumulated, causing the simulated distribution to differ from the true distribution of r_t.[7]

A higher degree of precision may be obtained by minimizing the two types of errors. Simulation errors may, e.g., be reduced by increasing the number of simulations, N, albeit the recursive discretization errors means that a significant increase in N has to be followed by a similar reduction in step size, Δt. Without such measures, applied simulations will in general converge to a wrong value. However, both larger a number of simulation and a lower discretization step come at the cost of reduced computational speed. The overall simulation error may be accessed by calculating the sample variance/standard error:

$$S^2 = \frac{1}{N-1}\sum_{j=1}^{N}(Cp^i(r) - \bar{C}p(r))^2, \qquad SE(\bar{C}p_t(r)) = \frac{S}{\sqrt{N}}. \tag{8.7}$$

The implicit convergence rate, $\frac{1}{\sqrt{N}}$, means that for a particular level of accuracy, N has to be very large. This further adds to the issue of prolonged computation time, where an, e.g., 10x precision improvement requires 100x increase of N. To circumvent this issue, various variance-reduction techniques have been developed to reduce S, instead of increasing N.

One of the most widely used techniques is the method of antithetic variates. This method calculates for every sample path $\{Z_{t_j}^i\}$ the antithetic path $\{\tilde{Z}_{t_j}^i = -Z_{t_j}^i\}$ and evaluate the associated payoffs from each. The advantage is twofold; 1) it reduces the amount of random numbers needed to produce N simulations, and 2) it reduces the variance of the simulation leading to improved accuracy. Accordingly, eq (8.6) may be augmented as follows[8]

$$\widehat{Cp} = N^{-1}\sum_{i=1}^{N}\left(\frac{Cp_i + \tilde{Cp}_i}{2}\right). \tag{8.8}$$

Further, the sample variance can be written as[9]

$$\begin{aligned} S^2(\widehat{Cp}) &= N^{-1}\,Var\left(\frac{Cp_i + \tilde{Cp}_i}{2}\right) \\ &= 2N^{-1}\,Var(Cp_i)\left[1 + \rho(Cp_i;\tilde{Cp}_i)\right]. \end{aligned} \tag{8.9}$$

As $\{\tilde{Z}_{t_j}^i\} = -Z_{t_j}^i$, the correlation, $\rho(Cp_i;\tilde{Cp}_i)$, is negative for most option payoffs; thus, $S^2(\widehat{Cp})$ is typically lower compared to applications of the Monte Carlo setup without the use of antithetic variates.[10] From eq (8.9) we clearly see that the highest reduction in variance is gained when the correlation is close to -1. Further, note that both its empirical mean and variance are based on averages over the pairs of antithetic variates, $\left(\frac{Cp_i+\tilde{Cp}_i}{2}\right)$. This is due to the fact that, when evaluated individually, the $(Z_i;\tilde{Z}_i)$, as opposed to their mean, are mutually dependent.[11]

In addition to the above measures, the literature suggests various ways to improve results from a numerical integration, preferably, but not limited to higher order discretization schemes (e.g. ref [4, cpht. 3.2]). Further discussion on these issues is outside the scope of this thesis.

[6] We refer to ref `http://en.wikipedia.org/wiki/Riemann_sum` (11.09.2012).
[7] [5, p. 185].
[8] Where Cp_i and $\tilde{Cp}_i = Cp_i(Z_i)$, $\tilde{Cp}_i(\tilde{Z}_i)$ respectively.
[9] [5, chpt. 10.5.5].
[10] [5, p. 195].
[11] [28, p. 67].

8.1.2 Random Numbers

To deploy the Euler scheme, we need to generate $\{Z^i_{t_j}\}$ i.i.d. draws from a standard Gaussian distribution. Most programming tools provide deterministic random number algorithms, which will only produce pseudo random numbers containing some degree of unevenness. In sketch, a typical random number generator [RNG] works by providing a seed/or start number[12] from which the algorithm calculates a long sequence of pseudo random numbers.[13] In a nutshell, the challenge is to create an algorithm capable of producing strong low discrepancy sequences, fast enough, to keep down the computational time. The literature suggests various ways to produce low discrepancy sequences;[14] however, this lies outside the scope of this thesis.

In general, we will rely on the pseudo random numbers created by Excel/VBA and solely aim at optimizing the computational time by imposing a Box-Muller Polar method[15] (For implementational aspects, we refer to e.g. [29, chpt. 29]) to avoid calling the inverse of the standard normal cumulative distribution via the slower spreadsheet function NORMSINV. The "polar method" was chosen over the "basic" form as to avoid calculation of trigonometric functions directly.

8.1.3 Implementation and Source Code

In Chapter 5, we considered the theoretical framework underlying the Hull-White Extended Vasicek model, and in the two previous sections, emphasis was put on a general Monte Carlo setup as well as the Euler discretization scheme. In this section, we outline the strategy for their combined evaluation using a computer.

We describe the process flow in terms of modulated source code developed during our application. First, each component in the process flow is briefly described. Second, we discuss in more detail, the main parts of the implementation. As before, individual VBA components are designated in `courier` by their module, followed by subroutine in capital letters, e.g. `module/SUBROUTINE`. All the subsequently derived source code can be found in Appendix C.

Figure 8.1 outlines the process flow. The code consists of three main parts, namely: I) Initialization routines for setting up the relevant data in vector form. II) Simulation of the underlying short rate process, and III) Payoff determination / pricing modules.

I) Initialization Routine In combination, the initialization routines retrieve the market data, user settings and calibrated model parameters to create the necessary vectors of input data, for use in subsequent procedures. First, a routine creates a vector of "event dates". The notion of event days regards all future dates, where some event determinant of the price of the derivative is taking place (e.g. trade date, effective date, payment dates, fixing dates, maturity dates etc.). Accordingly, upon all entries in the vector a simulation of the underlying is needed. Ideally, the tool would facilitate comprehensive structuring templates taking into account different date roll conventions,[16] business calendars, payment- fixing- and reset-lag,[17] compound methods and day count conventions,[18] amongst others. For our use, we, however, follow a shortcut method in which pre-adjusted payment dates are sourced using Bloomberg™ >SWPM<. Furthermore, we, for simplicity, assume that payment fixings and reset dates coincide (valid for European and Digital Caps (Floors)). While the application of the above conventions all are non-complex, the work is trivial and without further contribution to the scope of this thesis. The routine is contained in `mods/EVENTSCHEDULE` and is saved to memory.

[12]Excel uses the system clock to determine the starting point.

[13]Refer to `http://support.microsoft.com/kb/86523` (11.09.2012), for Excels random number algorithm.

[14]E.g. ref [5, chpt. 10.5.3].

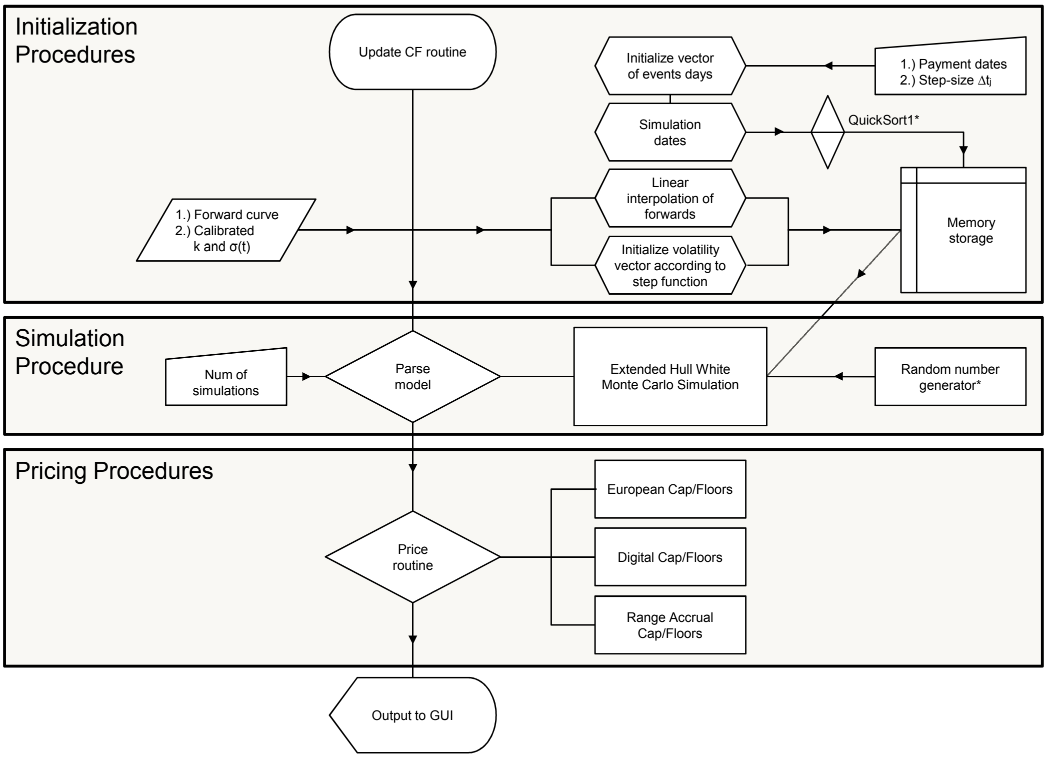
[15]We refer to `http://en.wikipedia.org/wiki/BoxMuller_transform` (11.09.2012).

[16]`http://en.wikipedia.org/wiki/Date_rolling` (06.10.2012).

[17]`http://en.wikipedia.org/wiki/Reset_(finance)` (06.10.2012).

[18]`http://en.wikipedia.org/wiki/Day_count_convention` (06.10.2012).

Figure 8.1: Process flow - Developed source code



Secondly, a routine creates two vectors of simulation dates and day count fractions respectively. The notion of simulation dates regards all dates on and in between dates contained in the event vector, added to enhance precision of the numerical procedure. Simulation dates are typically generated with a certain fixed step size and collapsed with the event vector to form a vector of simulation date with a fixed step size of, Δt_j. In our case, we generate a list of simulation dates according to a fixed user-specified step size input, Δt. As discussed in section 6.1 we assume a fixed 3M tenor of Libor rates, which means that we need to make sure that upon each entry to the list of simulation dates the vector also has its paired 3M value. Without too much detail on day count issues this is solved by creating an auxiliary vector of 3M paired values using the VBA function `DATEADD`. Further, our algorithm exploits the pre-adjustment of inputted payment days, when setting the intermediate simulation dates to avoid immense amounts of simulations around payment dates. The lists are merged with the event vector, any doublets are removed, and subsequently, the new vector is sorted using a well known third party sorting algorithm, Quicksort,[19] to form a final input vector of simulation dates with an adjusted step size of Δt_j. The vector of day count fractions is created using a fixed hard coded act/360 day count convention, as stated in section 6.1. All the coded routines are contained in `mods/Simulation_Dates` and are saved to the memory. The third-party sorting algorithm is contained in `third_party_code/QUICKSORT1`.

Thirdly, discount curve construction is initialized. As discussed in section 6.1 the current discount curve is normally constructed of yields from liquid traded instruments applying various interpolation techniques between data points. For our use, we generate the discount curve directly using Bloomberg™ >SWPM<. We note that it would be fairly simple to add a sub routine to our code, to perform curve construction from market yields if preferred, but this lies outside our scope. Next, linear interpolation between observations is applied (refer to section 6.1 for further discussion) to create a vector of discount factors matching each date in the simulation date vector. While discrete curve points are sourced with 3M intervals in our data set, the routine handles input down to daily observations. The routine is contained in `mods/LINTPDF` and is saved to memory.

[19] `http://en.wikipedia.org/wiki/Quicksort` (13.10.2012).

Fourth, calibrated model parameters, κ and σ_t, are initialized as determined during the calibration process outlined in section 7.1.1 (see figure 7.1). The volatility parameter buckets are matched and delineated in accordance with the simulation dates, to create a vector of volatilities according to the volatility step function described in eq (5.71). The volatility routine is contained in `mods/VOLINITIALIZE` and is saved to the memory.

This concludes the collection of initialization routines. In summary, we have created the following input vectors: event days, simulation dates, day count fractions, the current discount curve and a vector of volatilities.

II) Simulation of the underlying Simulating the paths of the underlying is the center of the numerical procedure. We simulate the short rate SDE using the Euler Scheme discussed in section 8.1.1

$$\widehat{r}_{t+\Delta t} = m_{t+\Delta t} + x_{t+\Delta t} \tag{8.10}$$

$$m_{t+\Delta t} = m_t + (r_0 e^{-\kappa(t+\Delta t)} - r_0 e^{-\kappa t}) + e^{-\kappa(\Delta t)} \kappa\theta(t)\Delta t \tag{8.11}$$

$$x_{t+\Delta t} = x_t - \kappa x_t \Delta t + \sigma(t)\, Z\sqrt{\Delta t} \qquad \text{with } x_0 = 0 \tag{8.12}$$

The structure of the calculation flow is setup to ease the internal calibration of m_t and is initiated by a matrix, which contains a sequential number of steps as columns and a specified number of generated paths of the underlying process as rows. As illustrated in eq (8.13), we execute a double loop procedure in columns, then rows calculating the first step in all paths before continuing to the next.

$$\#\text{Sim}\left\{\overbrace{\begin{pmatrix} x^1_{t_0} & x^1_{t_1} & \dots & x^1_{\tau} \\ x^2_{t_0} & x^2_{t_1} & \dots & x^2_{\tau} \\ \vdots & \Longrightarrow & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x^N_{t_0} & \dots & \dots & x^N_{\tau} \end{pmatrix}}^{\#\text{Steps}}\right. \tag{8.13}$$

First, we simulate $\{x^i_{t_j}\}$ using eq (8.12), where once again j designates each time step and i each generated path. As described in section 8.1.2, $\{Z^i_{t_j}\}$ is produced imposing a Box-Muller Polar method using one of the two generated random variables. The random number generation is contained in the subroutine `third_party_code/RANDNORM` and the derived $\{x^i_{t_j}\}$ is saved to the memory.
For the internal calibration of m_{t_j} we rewrite the known expression of the discount curve as

$$\begin{aligned} P(0,\tau) &= E^Q\left[e^{-\int_0^\tau r_s d_s}\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N} \exp\left(-\sum_{j=0}^{t_j=\tau} r^i_{t_j}\Delta_{t_j}\right) \\ &= \frac{1}{N}\sum_{i=1}^{N} \exp\left(-\sum_{j=0}^{t_j=\tau} m_{t_j}\Delta t_j - \sum_{j=0}^{t_j=\tau} x_{t_j}\Delta t_j\right) \\ P(0,\tau) &= \underbrace{\exp\left(-\sum_{j=0}^{t_j=\tau} m_{t_j}\Delta t_j\right)}_{\substack{\text{Requires calibration of } m_{t_j} \\ \text{to match } P(0,\tau)}} \underbrace{\frac{1}{N}\sum_{i=1}^{N} \exp\left(-\sum_{j=0}^{t_j=\tau} x_{t_j}\Delta t_j\right)}_{MC_{t_j}}. \end{aligned} \tag{8.14}$$

We proceed by isolating m_{t_j}

$$\begin{aligned}
\ln\left(\frac{P(0,\tau)}{MC_{t_j}}\right) &= -\sum_{j=0}^{t_j=\tau}\left(m_{t_j}\Delta t_j\right) \\
&= -\sum_{\substack{j=0\\ j<k}}^{t_j=\tau}\left(m_{t_j}\Delta t_j - m_k\Delta_k\right) \\
m_k\Delta k &= -\left(\ln\left(\frac{P(0,\tau)}{MC_{t_j}}\right)+\sum_{j<k}^{t_j=\tau} m_{t_j}\Delta t_j\right) \\
m_k &= \frac{1}{\Delta k}\left(-\ln\left(\frac{P(0,\tau)}{MC_{t_j}}\right)-\sum_{j<k}^{t_j=\tau} m_{t_j}\Delta t_j\right).
\end{aligned} \tag{8.15}$$

From eq (8.15) the chosen structure of the recursive procedure becomes clear. Calculating first step in each path before proceeding to the next means that MC_{t_j} is solved by the end of each loop. This readily facilitates successive calibration of m_k. The derived $\{m_tj\}$ vector is saved to memory. Having derived $\{m_{t_j}\}$, eq (8.10) can be applied to get the stochastic instantaneous short rate $\{r^i_{t_j}\}$. To ease the subsequent calculations of Libor rates, we instead employ the path specific version of eq (8.14)[20] to get a final $i\times j$ vector of stochastic discount factors $\{P(0,t_j)^i\}$. The simulation routine is contained in `mods/HULL_WHITE_MC_SUB` and the final $\{P(0,t_j)^i\}$ vector is saved to the memory.

III) Payoff determination / pricing modules The code is set up with three different pricing modules; European Caps (Floors), Digital Caps (Floors) and Exotic Libor-based range accrual legs. The structure of all three modules is essentially the same and the underlying equations are as described in sections 3.2-3.4.

First, a look-up algorithm, using the vector of event days, determines the sequential 3M pairs of discount factors starting from the effective date $[P(low), P(high)]$. Secondly, the corresponding stochastic Libor rates is calculated using eq (1.3). Thirdly, each module specific payoff is determined, discounted by the corresponding path specific discount factor and processed as outlined in eq (8.6). For the Range Accrual module, the process is augmented to include the calculation of Libor rates on all days in between payment dates to determine the relative number of days, where Libor is above/below the strike rate.

The resulting price estimates are submitted to the Graphical User Interface [GUI] alongside various information on Standard Error of Mean, 95%-confidence band and the execution time. We note that all our calculations implicitly assumes a $ unit notional amount and a bullet cash flow structure. [21] All price estimates are calculated as upfront basis points of the notional amount. The pricing routines are contained in `mods/PAYOFFCAPFLOOR`, `mods/PAYOFFDIGI` and `mods/PAYOFFEXOTIC`. Appendix B shows a picture of the developed GUI. We remark, that our Range accrual template takes customizable payoff functions as input, manually by a text string. E.g. for the depicted case, the payoff is determined as a fixed spread of $\left[\text{75bp+50\%}\times\text{Libor}\right]$, accrued for the the relative amount of days where Libor fixes above or below the strike rate. This payoff can be modified by augmenting the text string.

8.1.4 Test of Performance and Pricing Capabilities

To ensure sufficient amounts of confidence in the derived results, we need to test the modeled performance against liquid standard instruments, before application on more complex structures.

[20] That is; eq (8.14) is replaced by $P(0,t_j)^i = \exp\left(-\sum_{j=0}^{t_j=\tau} m_{t_j}\Delta t_j\right)\exp\left(-\sum_{j=0}^{t_j=\tau} x_{t_j}\Delta t_j\right)$.

[21] Refer to `http://en.wikipedia.org/wiki/Bullet_loan` (13.10.2012) for a brief description of a bullet loan.

We approach the issue by calculating the first and second moments as well as the calculation speed, upon various combinations of step size, Δt_j, and number of simulations,N. Given our former analysis, see e.g. section 8.1.1, we would expect that the precision of the modeled price estimate will increase when; 1) Δt_j is decreased and/or 2) when N is increased. Both at the expense of higher computational time.

The results from the described analysis performed against a highly liquid 5Y Standard ATMF European Cap are reported in table 8.1. A clear tendency toward decreasing price errors is found when moving from the left hand side to the right hand side of the table. This finding is likely caused by the right-sum approximation of the integral inherent in the simple first-order Euler discretization scheme (both in terms of the overall discounting and with respect to estimating the 3M Libor) as well as the amplification of local errors from the recursive calculation of $\hat{r}^i_{t_j}$. Next, a top-down examination of the residuals indicates an improved precision caused by the increasing number of simulations, although the effect is rather blurred in terms of the first moment. However, considering the second moment, the standard deviation, we find a consistent improvement in the uncertainty surrounding the modeled price estimates, due to the increased value of N. In general, when moving diagonally from the top left corner, we find that the precision is significantly increased at a clear expense of computational time. In the shown limits, the execution time spans from 2 seconds to >2 minutes.

Table 8.1: Price errors for various combinations of Δt_j and N. Benchmark: 5Y ATMF Cap with a market price 270.03 bp from section 6.1. Model parametrization: Table 7.1(d).

$10^3 \times$N \ Δt_j	30	14	7	4	1
5	6.88 (4.32 | 0:02)	5.22 (4.31 | 0:05)	4.23 (4.20 | 0:08)	3.31 (4.32 | 0:09)	1.42 (4.28 | 0:23)
10	6.84 (3.12 | 0:05)	4.59 (3.05 | 0:08)	5.29 (2.99 | 0:14)	3.47 (3.05 | 0:18)	1.80 (3.05 | 0:49)
15	6.97 (2.49 | 0:08)	5.01 (2.52 | 0:14)	5.18 (2.51 | 0:21)	2.83 (2.48 | 0:28)	1.07 (2.49 | 1:12)
20	8.23 (2.21 | 0:12)	5.57 (2.17 | 0:19)	4.73 (2.16 | 0:29)	2.82 (2.17 | 0:38)	1.13 (2.15 | 1:37)
25	8.44 (1.98 | 0:15)	5.50 (1.95 | 0:23)	5.21 (1.95 | 0:35)	3.10 (1.92 | 0:47)	1.82 (1.93 | 2:01)
⋮	⋮	⋮	⋮	⋮	⋮
65	8.52 (1.21 | 0:39)	4.54 (1.19 | 1:01)	5.35 (1.20 | 1:32)	3.52 (1.18 | 2:62)	n/a

The first number in columns 2-6 indicates the estimated price residual (HWExtV - Market), while the standard deviation and execution time are given in parentheses according to: $\hat{\text{res}}$(bp) ($\hat{\text{StDev}}$ | 0:00 min). The missing entry in the bottom right corner, is caused by the limited storage capabilities of VBA due the significant increase in total number of calculations for $\Delta t_j = 1$ and $N = 10^3 \times 65$.[22]

Considering an optimal trade-off between precision and speed at one hand, and on the other, the limiting storage capabilities of VBA, we find an optimal specification around $\Delta t_j = 4$ and $N = 10^3 \times 20$. Consequently, these settings will be applied in the remainder of the thesis.

Table 8.2 summarizes the price residuals from a range of liquid ATMF Caps to consider the performance at various maturities. First we note, that the achieved results resembles quite closely the results found during the analytical calibration process (refer to figure 7.5(c)-(d)). This finding is comfortable, as is indicates a clear coherence between our two developed methods of pricing. Examining the actual results, we find that clear issues are present around the 20Y segment, as similarly discussed in section 7.2. Furthermore, residuals (bp) seem to increase in tenor, which is no surprise due to the previously described discretization errors. Apart from the front and the 20Y segments, the pct deviations, however, are fairly stable in time, which is comforting from a modeling perspective, as it means that the relative errors are kept reasonably at bay. Commenting on the front of the curve, the larger pct deviation found here is, to some extent, caused by the low nominal

[22] Several remedies have been sought to increase the total calculation capacity of our VBA setup. This included, but was not limited, to thorough variable declarations, simplification of some of the calculation routines and reset of "expired" matrices in the calculation routines. As noted in section we recommend others to use alternative programming tools such as MatLab.

Table 8.2: Price errors for standard European Caps (various maturities). Model parametrization: Table 7.1(d). $\Delta t_j = 4$ and $N = 20.000$.

Maturity	1	3	5	7	10	20	30
Bp	0.49	1.41	2.82	3.47	7.93	28.63	14.26
Pct	3.87	1.15	1.04	0.79	1.16	2.17	0.85

The first row in columns 2-8 shows the estimated price residuals (bp) (HWExtV - Market), while the second row indicates the standard deviation.

price level. However, we cannot rule out that the coarse approximation of the yield curve[23] adversely affects the short end, due to the higher sensitivity toward changes in the underlying. Although, some deviations are detected for longer maturities, committed price errors are still within one digit basis point deviations and in percentage terms, they are all significantly below a 5% level. One could re-iterate the calibration procedure and, e.g., look even deeper into the issues around the 15-20Y segments. However, for our use, we will only need to price structures up to 10Y maturities. Consequently, we are satisfied with the results achieved using our HWExtV Monte Carlo setup and confident that the model will serve well during the case sections in chapter 9.

As a closing remark, we remind that the intended use of the numerical procedure obviously is within the pricing of more complex and exotic instruments, where bid/offer spreads are significantly wider than their corresponding standard piers.[24] This is a reflection of the inherent model risk and uncertainty associated with the price determination of more complex structures, and in such perspective, the achieved results are even stronger (particularly for 1-10Y maturities).

[23] As discussed in section 6.1

[24] Based on own observations from working at Nordea Markets.

Part III
Application and Case Sections

Chapter 9

Pricing of Two Different Complex Interest Rate Derivatives

In this section, we apply the developed Hull-White Extended Vasicek Monte Carlo setup to price two different cases of complex derivatives. Each case begins by an outline of the product, its composition and its payoff profile. Moreover, we establish the individual contexts to which, each product is applied. First, we consider a case of liability optimization (hedge) of the cash flow, occurring from a floating rate loan via implied sale of, i.a., digital optionality. Secondly, we turn to a pricing an asset/liability wrap, extending our discussions to how structured (in our case range accrual) bonds typically are created. The specified context approach, used throughout this chapter, is chosen not only to provide a theoretical understanding of the payoff profiles, but on the same time, to sketch a real life setting, for which the derived theory is commonly applied in practice. For each case, we discuss the price precision of our models as well as the generality of the global best fit - full grid calibration parametrization, as found in table 7.1(d). Accordingly, in the following sections, we draw upon all the reviewed theory and implemented algorithms treated in this thesis. The described setting represents similar cases of own experience.

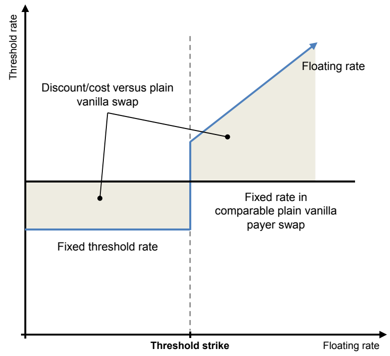
9.1 Barrier Swap

A barrier swap is an agreement often used as a liability instrument in which, a floating rate borrower may hedge against modest rises in interest rates, thereby achieving a lower fixed rate than a comparable plain vanilla interest rate swap. The none-capped feature of payments implies the product to be classified as an optimization tool more than an actual hedge. Party |A| pays a fixed rate against receiving a floating rate from party |B|, as long as the floating rate remains below a pre-agreed barrier on the fixing dates, specified in the contract. If, at a fixing date, the floating rate reaches the barrier, the fixed rate payer will instead pay the floating rate for the fixing period in question. Thus, upon each fixing, the product determines whether party |A| is to pay a floating rate, or a fixed rate. Savings on the fixed rate are increased the closer the pre-agreed barrier is to the fixed rate. The exposure on the barrier leg, as seen from party |A|, is shown in figure 9.1.

To create the outlined payoff from the complex product, we consider its underlying components. As seen from party |A|, we have: 1) a payer plain vanilla IRS, as described in section 3.1, 2) a sold Digital Cap, as described in section 3.3, and finally 3) a sold European Cap, as described in section 3.2. The payer IRS creates the fundamental fixed for floating structure in the product. The sold Digital and European Caps, both struck at the agreed barrier level, create the jumped linear payoff, above the strike rate. The European Cap ensures that party |A| above the strike rate pays the difference between the barrier and the current floating rate. The Digital Cap adjusts the base, meaning that party |A| above the barrier pays

$$\{\text{Fixed Barrier Rate} + \text{Sold Digital payoff} + \text{Sold European payoff}\} = \text{Floating Rate}. \tag{9.1}$$

Figure 9.1: Barrier Swap - Payoff profile of the barrier leg.

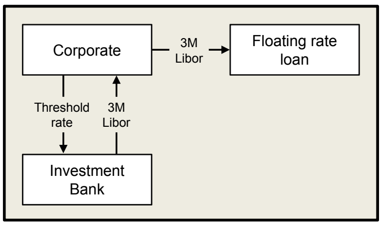


The sold optionality means that the fixed Barrier Swap Rate < a plain vanilla market level IRS. For a given choice of barrier, this implies simultaneous solving of a fixed payoff spread in the Digital Cap, and further, a corresponding fixed rate of the IRS such that a consistent payoff, as shown in figure 9.1, is obtained alongside a package-NPV of[1]

$$\{\text{NPV Barrier IRS} + \text{Received premiums on Digital and European Caps}\} = 0. \tag{9.2}$$

Business Case: Consider a Corporate institution |A|, having achieved a 10Y floating rate loan for longer real investment purposes. The Treasury Department prefers fixed rather than floating funding; however, they expect only modest upside potential of short rates during the path of the tenure. Specifically, the company does not believe that the floating rate will exceed 5% (6%), and

Figure 9.2: Barrier Swap - Context of application.



therefore, wants to sell the potential that lies over this particular barrier(s) so as to lower the fixed rate of the hedge. The transaction follows the outlined structure in figure 9.2.

9.1.1 Calibration

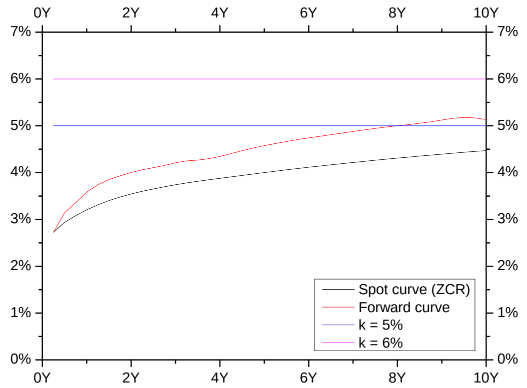
In table 7.1(d), we found a reasonable full grid calibration result. However, for this particular product, one issue draws upon attention. The barrier lies significantly OTM for most of the forward curve, as recognized in figure 9.3 by an area of the grid, where our parametrization indicated some deviations (refer to figure 7.6(e)-(f)). This suggests that the current calibration may not be sufficiently accurate in this particular area. To test the generality of our parametrization (table 7.1(d)), we re-calibrate the model against: 1) the absolute strike rate (barrier), as chosen by the client, and 2) the relevant part of the yield curve (≤10Y).[2] By narrowing the calibration problem, we are likely to gain a very close fit of market prices, which enable us to verify the price sensitivity toward changes in the underlying parametrization of the model, i.e., its robustness, as discussed in section 7.1. We remind that while the need for a re-run of the calibration procedure would be

[1]Without regards to credit or trading spreads.

[2]An outline of the corresponding Black76 volatility surface in absolute strike levels, may be found in Appendix A.1.

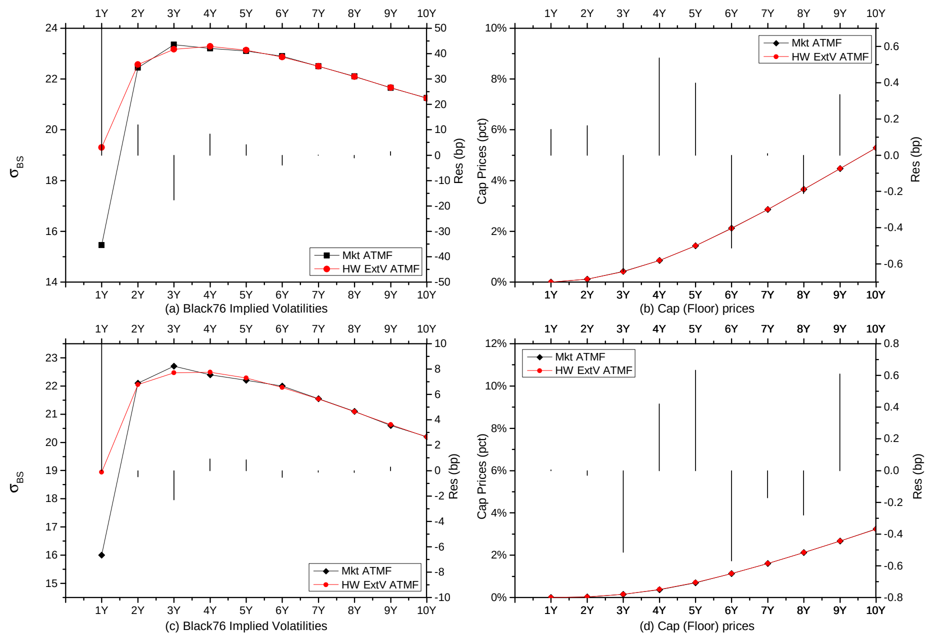
a significant drawback, it may be necessary in certain situations, due to the lack of flexibility in the HWExtV model, as discussed in the previous sections. Figure 9.4(a)-(b) depicts the obtained

Figure 9.3: Barrier Swap - The Forward Curve vs Strike levels.



results from an absolute 5%-strike level re-calibration, verifying the anticipated close fit of market prices. A re-calibration against a 6%-strike level was performed with a similar result as seen in 9.4(c)-(d). Both parametrizations can be found in Appendix D. The new parametrizations work as benchmarks during the later pricing process.

Figure 9.4: Re-calibration versus absolute Barrier Strikes. (a)-(b) show the calibration results for the 5% strike level. (c)-(d) depicted the 6% level results.



9.1.2 Results

The results of pricing a Barrier Swap (10Y), using the initially obtained full grid parametrization from table 7.1(d), are shown in table 9.1. The Swap is priced at two different strike levels (barriers) of 5 and 6%, respectively, to check the generality and performance of the model at a varying distance from ATMF. We remark that the model is only applied to the Digital Cap as both the Swap and European Cap components are already given in the market using the analytical formulas, eq (3.7) for the IRS and eq (4.9) together with the market Black76 implied volatility from section 6.1, for the European Cap.

Table 9.1: Barrier Swap (10Y) - Price composition and model sensitivity toward changes in the underlying parameter set. Unless otherwise stated, numbers are derived using the full grid parametrization from table 7.1(d). The benchmark shows the "true" value of the Digital Cap, according to the new re-parametrization provided in Appendix D. Rates and payoffs are all shown in (pct), whereas present values are in upfront (bp) of the notional amount (100m$).

Barrier	5.00	6.00
Present value, upfront bp		
Cap	529.49	323.50
Digital (payoff 1.88 | 2.53)	543.33	458.06
NPV, sold optionality	1,072.82	781.56
Fixed Barrier Swap Rate*	**3.1210**	**3.4743**
Fixed rate discount versus plain vanilla	1.3011	0.9478
Benchmark		
Digital	548.31	478.98
△Pct	-0.91	-4.37
△Bp	-4.97	-20.92
△Nom	-49,724	-209,185

Reference: Par US IRS (10Y) 4.4221 | 3M Libor 2.7362 | Black76 Vol: K= 5%@21.25 / K= 6%@20.20.

* The barrier swap rate (fixed) is derived by insertion of the positive NPV from the two sold caps into eq (3.7). The "true" trade specification according to our benchmark(s): Barrier @ 5.00 [NPV 1,080.71 | Payoff digi 1.89 | Fixed rate 3.1115] and Barrier @ 6.00 [NPV 808.16 | Payoff digi 2.56 | Fixed rate 3.442], as derived for value par.

First, the composition of the all-in barrier IRS rate (fixed) should be noted. The sold optionality provides a total of 1072.82 / 781.56 bp (upfront) worth of income to the structure. Using the positive NPV as upfront payment in a payer IRS reduces its fixed rate by 1.3011 / 0.9478 pct points, corresponding to an approximate DV01[3] of 8.24 bp (upfront value) of the underlying IRS. Moreover, the initial negative carry[4] of the structure is reduced from 168.59 bp to 38.48 / 73.81 bp, which is often one of the main arguments used by borrowers to enter into Barrier Swaps.

The bottom of table 9.1 shows the price of an identical Digital Cap using our benchmark calibration. In general, we verify that the full-grid parametrization from table 7.1(d), consistently, underestimates the fair value of the Digital Cap according to our benchmark. Further, we see that the level of the under-projection increases, as widely expected (refer to figure 7.6(e)-(f)), by the distance above ATMF. At a 5% strike level (58 bp above ATMF), we find an improved performance. The general calibration provides a price deviation of only -4.97 bp (upfront) below target, corresponding to a -0.91 pct deviation in the total premium of the Digital Cap. Bearing in mind that the actual bid-ofr spreads for European (10Y) Caps (Floors) are around 10 bp,[5] the modelled price deviation of this complex instrument is considered quite acceptable.

Next, we turn to consider the performance further out, at a 6% strike level (158 bp above ATMF). In this area; however, the picture deteriorates rather significantly, since our full grid parametrization comes in -20.92 bp below target, corresponding to a -4.37 pct deviation in the total premium of the Digital cap. In practice, this result is no longer within acceptable tolerances, as the price deviations are more than even the bid-ofr spreada and credit spread would be able to cover. In the notes of table 9.1, we provide the "true" deal specifications (value par) for the two considered instruments. Even though, we find only modest changes to the barrier fixed rates, this is mainly caused by the different magnitude in DV01 of the linear and the non-linear components. Overall, we find that if party |A| trades at the HWExtV modeled prices, as described in table 9.1,

[3] For a description on Dollar Duration we refer to most standard text books on bonds price analysis or ref `http://en.wikipedia.org/wiki/Bond_duration` (28.11.2012).

[4] Defined as the difference between the paid barrier rate (fixed) and the received 3M Libor.

[5] Bloomberg™ - ICap.

he / she will incur a -4,97 / -20.92 bp (upfront) initial fair value loss, which in nominal terms, per 100m$, corresponds to -49,724 / -209,185$.

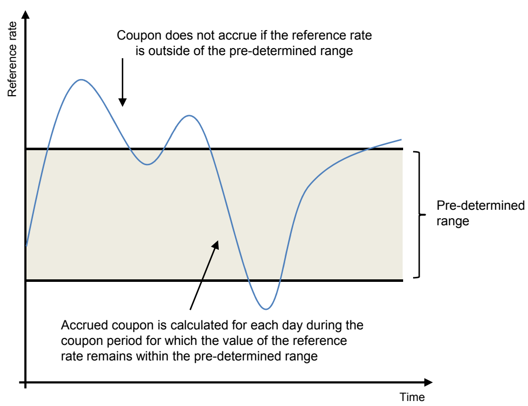
In conclusion, we find that for this particular Barrier Swap(s), a well calibrated (full-grid) HWExtV model *is* able to provide quite good results, for strike levels fairly close to ATMF (<100 bp above). Admittedly, for barriers placed further OTM, the result deteriorates, and at the investigated level (ATMF + 158 bp), the calculation is no longer valid for pricing purposes. In summary, we find that while the HWExtV model is not complete in its description of the volatility surface, it does provide enough flexibility to match certain OTM areas, accurately enough, for pricing purposes. As a consequence, for our considered case, we find only partial arguments in support of a re-calibration of the model parametrization, conditional on the need for pricing options further OTM. A pleasant and valuable finding, both in terms of time consumption and generality of the model.

Before getting too exited about the achieved result, we note that Digital Caps may be analytically priced, using either static replication in Call-spreads or the standard vanilla models, as described in section 3.3. Thus, while in this case, the HWExtV model served our purpose well (at least at the 5% strike level), in practice, numerical procedures would generally not be used.

9.2 Range Accrual Swap

A range accrual swap follows the theory outlined in section 3.4. In summary, a range accrual swap is an agreement between two parties, where a party |A| pays (receives) a floating or fixed rate in a given period, against receiving (paying) an accrual rate from a party |B|. The accrual rate is a fixed or floating rate plus a spread and accrues only for the amount of days, where a specified reference rate is within a pre-determined range.[6] Accordingly, the accrual of interest on the accrual leg is linearly dependent on the fraction of days during the given period in which the reference rate has remained within the pre-determined range. Numerous combinations of accrual rates and reference rates exists; however, we will restrict ourselves to consider the specific case, where party |A| pays a floating rate plus a spread against receiving an accrual rate from party |B|, determined as a fraction of the floating rate plus a fixed spread accrued only for the amount of days, where 3M Libor is within a specified range.

Figure 9.5: Payoff from a Range Accrual leg



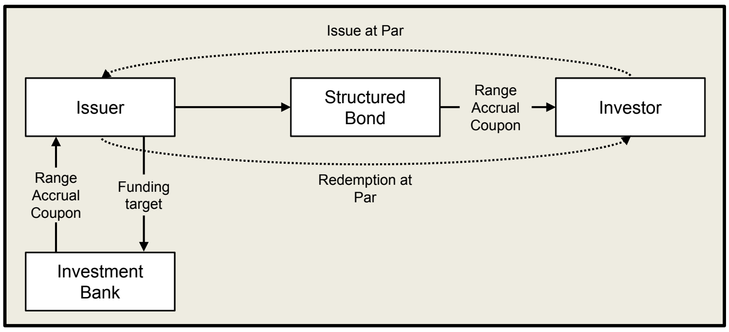
Business Case: Consider an Investment Bank with access to a range of investors, whom wants to place surplus liquidity on a 5Y horizon. In return, the investors expect to receive a yield constructed to fit their expectations on a specified future range for the 3M USD Libor. At the other hand, the Bank has an investment grade[7] corporate entity that seeks to obtain 5Y funding at a fixed

[6] We refer to the glossary of SuperDerivatives for a verbal outline of the product `http://www.sdgm.com/Support/Glossary.aspx?term=Range\%20accrual\%20swap` (13.01.2013).

[7] For a brief outline of Standard & Poors definition of "investment grade" entities we refer to `http://en.wikipedia.org/wiki/Standard_\%26_Poor's` (14.12.2012).

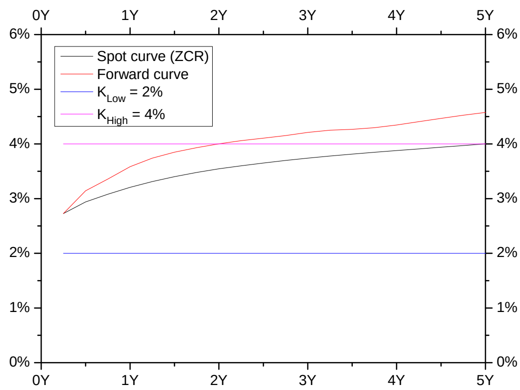
funding target of 3M USD Libor plus 25bp.[8] The Investment Bank wishes to create a structured bond matching the demand of the investors, while simultaneously sustaining the supplying Issuer's funding target and preference for standard floating rate funding. Such structure may be obtained, as outlined in figure 9.6.

Figure 9.6: Range Accrual Swap - Context of application.



A pricing process could be as follows; First, the Investment bank indicatively prices up a range accrual swap versus the Issuer, where the bank pays the structured leg against receiving the Issuer's funding target. Secondly, different terms on the accrual leg are discussed with the investors, to best fit their preferences. This iterative procedure, also referred to as soft-sounding, typically also includes discussions with the Issuer with regards to improvements of their funding target such as to induce sufficient investor appetite.[9] Finally, when equilibrium terms are derived, the bond is issued at par. The Issuer receives the accrual leg (running), against paying the funding target in a hedge transaction, simultaneously entered with the Bank. Further, the Issuer passes on the received accrual leg as coupon payments to the investors (refer to figure 9.6). We note that as the

Figure 9.7: Range Accrual Swap - The Forward Curve vs Accrual Strikes.



derived accrual leg in the swap contains a running 25 bp of surplus value above the forward curve, one could argue that the bond, only consisting of the cash flow from that "one leg", should issue above par. However, we remind that, as opposed to the swap, the bond implies actual funding (that is, transfer of the principal amount between the two parties), so the inherent credit risk of the Issuer means that the surplus value of 25 bp represents exactly the market value of that risk,

[8] Assumed that the Issuer have similar preference for paying a standard floating rate plus spread. In practice this may not need to be the case and the Issuer could, for instance, wish to pay an equivalent fixed rate where however this is without further contribution to the subject.

[9] We remark that a higher numerical funding spread paid by the Issuer means that a higher amount of value is transferred to investors in terms of a more attractive structured coupon.

and so causes the bond to price at par. Obviously, off-market terms of the bond range accrual coupon could be agreed, which would simply be offset by an issue price above/below par. To this end, off-market terms are without further contribution to the subject. In summary, our business

Table 9.2: Term sheet - Structure Range Accrual Note. Investment grade Issuer.

Currency	USD
Issue Size	500m
Maturity	5Y
Issue Price	Par
Accrual Coupon	$^{+}\big[0.01\%\,,\ \frac{n}{N}\times\big(\ \gamma\ \times \text{reference rate} + 0.5\%\big)\times \text{Act/360}\big]$
Accrual Range	$2\% < \text{reference rate} < 4\%$
Reference rate	3M Libor
Issuer funding target	3M Libor + 25 bp
Issuer Rating	Investment grade

Where n denote the total number of calendar days in the period where the reference rate fixes within the specified range (in case of holidays the fixing from the previous business day is used). N denotes the total number of calendar days in the period.

case provides the following specified terms, see table 9.2. As seen in the table, the accrual coupon is contained in the positive part operator $(0.01\%\,,\text{ accrual coupon})^{+}$. This means that the actual coupon rate itself, is floored at 0.01% regardless. This is a common feature on structured bonds, as accounting systems in general, are not very fond of bond coupons at zero. Another argument, which actually leads us to include this feature is the fact that, it introduces as pseudo-form of path dependency, which only can be solved by numerical measures.[10] Even though in practical terms, the effect of this feature is negligible, our argument behind its inclusion is obvious; if we price a case of path dependency, the development of the HWExtV Monto Carlo setup is justified.

A further study of the terms in table 9.2 shows that the structure only contains one unknown (competitive) parameter, the participation rate γ.[11] To determine γ, we need to calculate the nominal value of a floating leg plus 25 bp using parts of eq (3.7). Further, we need to compare with the nominal value of the accrual leg (iteratively) for different values of γ using eq (3.25), so that

$$\{\text{NPV floating leg - NPV Accrual leg}\} = 0. \tag{9.3}$$

As with the Barrier Swap, we proceed by a re-calibration of the model parameters to create a benchmark for our initial full grid parametrization results in table 7.1(d).

9.2.1 Calibration

As previously discussed in relation to table 7.1, calibration (d) provided a reasonably good full grid parametrization result. However, once again the particular pricing problem in question draws upon attention, as the Range Accrual Swap contains two uneven strike levels; one significantly ITM and the other almost exactly ATM, as shown in figure 9.7. As found in section 7.2, some deviations on far OTM options (though less pronounced at 5Y) were apparent (refer to figure 7.6(c)-(d)), thereby challenging the accuracy of the general parametrization for pricing in these particular areas. To test the generality of our initial full grid calibration onto this complex Range Accrual Swap, we re-calibrated the model separately, against each of the two accrual strike levels, and only for the relevant parts of the curve ($\leq 5Y$). Once again narrowing the calibration problem, this time by an

[10] Recall from section 3.4 that a range accrual leg, merely is a sum of daily digital Caps and floors and thus, can be decomposed and priced via analytical vanilla models, even though the procedure requires quite some additional setting up.

[11] Participation rate refers to the return of the structured leg. If the structured leg increases or decreases at the same rate as the change in value of the underlying, the payout is said to have a one-to-one return or 100% participation rate. If the structured leg return increases or decreases at a faster rate than the underlying the return is subject to a multiplier.

even more time-consuming process of dual calibration, we created a benchmark for the evaluation of the generality and robustness of the models.

Figure 9.8: Range Accrual Swap - Re-calibration versus absolute Accrual Strikes. (a)-(b) display results for the calibration vs 2.00%. (c)-(d) depicts the 4.00% level.

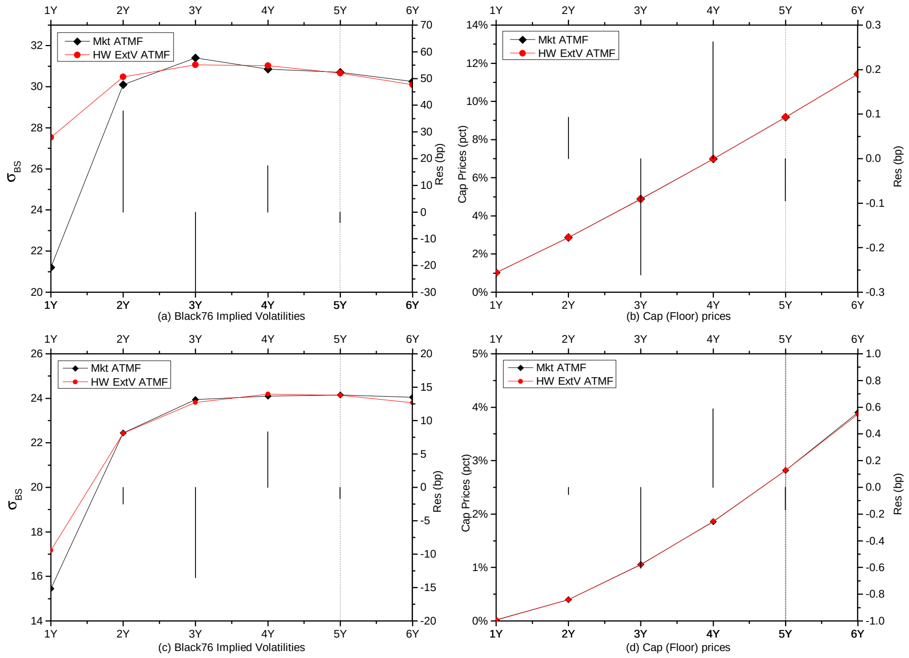


Figure 9.8 shows results from each of the benchmark re-calibrations, both producing close fits of market prices. Each parameter set can be found in Appendix E. The two re-calibrations work as benchmarks in the subsequent pricing process.

9.2.2 Results

The results from pricing the outlined Range accrual bond (5Y), using the initially obtained full grid parametrization from table 7.1(d), are shown in table 9.3. First, we note that the value of the sold floating leg + spread is determined using the relevant parts of eq (3.7), from now on designated as the funding leg. Further, the participation rate, in the middle of the table, has been derived iteratively, so that the combined sell / buy of the accrual floors equals exactly the NPV of the sold funding leg. We find, that at a participation rate of $\gamma = 3.0543$, the Range Accrual bond is priced at par (zero package NPV).

The lower part of table 9.3 contains the prices of the identical Accrual Floors, as obtained from our benchmark calibrations. In general, we find that the full-grid parametrization from table 7.1(d), consistently, overestimates the fair value of both Accrual Floors according to benchmark. Further, we see that the level of the over-projection increases, when shifting from the 4% to the 2% strike level, as widely expected (refer to figure 7.6(e)-(f)), when moving deeper ITM.

At the 2.00% strike level (207 bp below ATMF), we find that the model performs quite poorly. The general calibration provides a price deviation of 24.91 bp (upfront) above the target, corresponding to a 24.91 pct deviation in the premium of the sold Accrual Floor, clearly an insufficient result. At the 4.00% strike level (6.7 bp below ATMF), we find more a accurate result. The general calibration provides a price deviation of 23.33 bp (upfront) above the target, which only corresponds to a 1.11 pct deviation in the premium of the bought Accrual Floor. Clearly this result is far better, although, in terms of nominal values, the deviation is still large.

Concluding on the individual results, the HWExtV model does not serve us well, due to the fact that the utilized area on the volatility surface is too big a task for our model, and we are generally

Table 9.3: Barrier Swap (10Y) - Price composition and model sensitivity toward changes in the underlying parameter set. Unless otherwise stated, numbers are derived using the full grid parametrization from table 7.1(d). The benchmark shows the "true" value of the Accrual Floors, according to the new re-parametrization provided in Appendix D. Rates and payoffs are all shown in (pct) and present values are in upfront (pct) of the notional amount (100m$).

Range accrual (Strike lvl, K)	2.00	4.00	Net
	Sell	Buy	
NPV (Upfront Bp)			
Accrual Floors	1.59	-21.11	-19.52
Floating leg + Spread (25 Bp)	19.52		19.52
NPV (total package)			0.00
Participation Rate			3.0543
Benchmark			
Accrual Floor	1.34	-20.88	-19.54
NPV (Total packages)			-0.02
ΔPct	18.59	1.11	0.09
ΔBp	24.91	23.23	2.00
ΔNom	249,119	-232,258	16,860

Reference: Par US IRS (5Y) 3.9959 | 3M Libor 2.7362 | Black76 Cap-Vol: K= 2%@30.70 / K= 4%@24.15

* The participation rate (fixed) is derived by equating the NPV from the two sold/bought Accrual Floors with the NPV of the floating leg + spread. The "true" trade specification according to our benchmark: Participation rate, $\lambda = 3.05154$ $\left[K_{Low} = 2.00 \,|\, \text{NPV } 1.3390 \,|\, K_{High} = 4.00 \,|\, \text{NPV } 20.8573\right]$.

short in flexibility in such a case. Also, table 9.3 shows that the benchmark package NPV is almost identical to the result provided by our full grid calibration, 19.52 / 19.54(!) In essence, our model provides the correct price, and as such, trading at the result would only cause an off market value of 2 bp upfront(!) In effect, as both the components are placed ITM, and since we simultaneously are selling deep ITM options for the purchase of options closer to ATMF, deviations on both sides of the skew offsets. Rather than satisfied, we are discontent by this unpleasant finding. For certain cases, this means that the model will produce the desired results, leading one to think that we are performing well, although, the results are in fact based on severe mispricing of each component.

Without further comments, this strongly underlines the notion of model risk, and we lead to conclude that the HWExtV model, is not well suited for pricing cases as the one in question. Consequently, repeated re-calibrations are needed in order for the model to produce accurate results. An unsatisfying result highly expensive in terms of time and generality.

Chapter 10

Conclusion and Closing Remarks

The aim of this thesis was to expand our theoretical knowledge to include full term structure modeling under stochastic interest rates. In particular, we sought to emphasize the theoretical argumentation as well as the implementational aspects that are required to enable pricing of more complex interest rate derivatives in practice. Consequently, the main focus of this thesis has been to provide a thorough exposition of the theory related to the equilibrium Vasicek77 model and its descendants, the arbitrage-free HWV and HWExtV models. Furthermore, we presented the Monte Carlo method for numerical pricing in these models, and finally, outlined a strategy for a combined evaluation of the derived methods on a computer.

When examining the market vanilla models, Black76 and the Normal model, we found that Black's model assumes that forward rates are log normal distributed, and as a consequence, never take on negative values. The Normal model, in contrast, assumes normality of forward rates, meaning that negative rates are possible in this model. Further, we verified the problems inherent to both models in matching the volatility skew.

In chapter 5, we gave a detailed exposition of each of the closely related Vasicek descendant models. First, we found that the one-dimensional dynamics of the instantaneous short rate was very convenient, since all rates and bond prices where readily defined, by no arbitrage arguments, as the expectation of the functional expression of the short rate process, in each of the three models. Furthermore, we saw that both spot rate, and forward rates are Gaussian, whereas bond prices are log normal. Accordingly, a bond price never attains a negative value and is thus well defined, even at negative rate levels. All three models exhibit mean reverting functional expressions, in the sense that the expected value of the short rate settles at a constant value. For the Vasicek model, this long-run mean level is represented by μ under P, whereas in the two Arbitrage-Free HW models, the risk neutral mean-reversion levels were approximately represented by θ_t. This latter specification in the HW models solved the inherent problems of the Vasicek model, related to fitting the the initial yield curve. We also found that a analytical solution for bond options is readily available within all three models. Thus, via a replication strategy, we were able to price European Caps analytically, which was an important feature for the later calibration of the models. Moreover, we considered the resulting volatility structures. We saw that both the Vasicek and the HWV model exhibited exponential decay in their forward rate volatilities, somewhat limiting their practical generality, as fitting the often apparent volatility hump was infeasible. The introduction of time dependency into σ_t (the HWExtV model), furnished a solution to the problem, and we derived a piecewise linear volatility function that could then be used in the later implementational chapters. Finally, we found that the Vasicek frameworks are able to express a certain degree of volatility skew, considering both options on ZCBs and Caps (Floors), although, the "levers" available for adjustments of the volatility skew are limited to a few very indirect parameters. The (seasoned) Vasicek frameworks do not account for the emerge of a significant tenor basis and OIS discounted prices. To avoid possible issues related to this, we chose to use historical market data, pre 2007 and the financial crises.

Next, the calibration procedure and objective functions were discussed. This was followed by calibration of the HWV and HWExtV models to the European Cap (Floor) volatility surface. Here, we developed a non-scientific expression for weights in the objective function to induce the solver algorithm to search for a solution in the right direction. Albeit, the HWExtV model showed excellent performance comparing with the HWV model, it still provided an insufficient fitting of the market skew, when moving further away from ATMF. The biggest issues were found on far OTM options, due to the strictly decreasing volatility structure of the HWExtV model. In a nutshell, this is a significant drawback, as more than 80-85% of all caps (Floors) are traded OTM.

Concerning the theory of simulating the HWExtV model by means of a standard Monte Carlo setup, using a Euler discretization scheme, we discussed both the resulting simulation and discretization errors, and how to improve the performance of the simulation procedure. Next, we described in detail the developed process flow and source code of the present VBA implementation. Despite the limiting memory capacities of VBA, we found, in coherence with the analytical calibration results, a reasonable good performance of our simulation setup. The optimal discretization and number of simulation settings were then determined for later applicational purposes.

Finally, we could introduce our applicational case sections, pricing two examples of complex interest rate derivatives (a Barrier Swap and a Range Accrual Swap). Each product was applied, using a specified context approach, in the framework of a small business case, with the aim of adding a touch of real life to the context of the financial and theoretical aspects. The model performed well, when pricing the Barrier Swap at strike levels not too far away from ATMF ($\leq$100bp). However, at a 6% strike level the deviations were too high for pricing purposes, and the HWExtV model required a recalibration to the strike level in question, limiting the generality of the setup. Next, we turned to price a Range Accrual Swap. At first hand, the modeled results seemed fairly acceptable; however, a thorough examination showed that both accrual components were significantly mispriced. In fact, as both components were placed at one side of ATM, and since we were selling deep OTM options for the purchase of options almost ATM, the individual deviations offsetted when combined(!) This, rather inconvenient finding, highlighted even further, the notion of model risk, also widely apparent through the entire tenure of our implementational and applied work.

In conclusion, we found that while the HWExtV model does have many desirable features and provides an instructive starting point, it is not suficiently flexibility to keep up with the precision, required for pricing purposes nowadays. Although, the one-factor HWExtV framework is outdated in many ways, still, as mentioned in the introduction;

> *"... the Hull-White Extension of the Vasicek model is one of the historically most important interest rate models, being still nowadays used for risk-management purposes."* [1, p. 72].

Bibliography

[1] D. Brigo and F. Mercurio. *Interest rate models - Theory and practice with smile, inflation, and credit*, volume 2. Springer Verlag, 2006.

[2] D. Filipovic. *Term-Structure Models - A graduate Course*, volume 1. Springer Finance, 2009.

[3] Jesper Lund. Lecture notes - review of continuous-time term-structure models - part i and associated slides. 1998.

[4] L.B.G. Andersen and V.V. Piterbarg. *Interest Rate Modeling: Foundations and Vanilla Models*, volume 1. Atlantic Financial Press, 2010.

[5] A.J.G. Cairns. *Interest Rate Models: an Introduction*, volume 10. Princeton University, 2004.

[6] M. Kac. On distributions of certain wiener functionals. *Transactions of the American Mathematical Society*, 65(1):1–13, 1949.

[7] J.M. Harrison and D.M. Kreps. Martingales and arbitrage in multiperiod securities markets. *Journal of Economic Theory*, 20(3):381–408, 1979.

[8] T Bjrk. *Arbitrage Theory in Continuous Time*, volume 3. Oxford Finance, 2009.

[9] John C. Hull. *Futures and other Derivatives.* Prentice Hall International, New Jersey, 2006.

[10] Jesper Lund. Lecture notes - pricing fixed-income derivatives with the forward-risk adjusted measure. 1998.

[11] Jesper Lund. Lecture notes - review of continuous-time term-structure models - part ii - and associated slides. 1998.

[12] L.B.G. Andersen and V.V. Piterbarg. *Interest Rate Modeling: Term Structure Models*, volume 1. Atlantic Financial Press, 2010.

[13] Andrew Lesniewski. Interest rate and fx models - the forward curve. Feb 17 - 2011.

[14] Andrew Lesniewski. Interest rate and fx models - girsanov, numeraires, and all that. Feb 17 - 2011.

[15] Andrew Lesniewski. Interest rate and fx models - the volatility cube. Feb 17 - 2011.

[16] P.j. Hunt and J.E. Kennedy. *Financial Derivatives in Theory and Practice*, volume Revised Edition. John Wiley and Sons Inc, 2004.

[17] O. Vasicek. An equilibrium characterization of the term structure. *Journal of Financial Economics*, 5(2):177–188, 1977.

[18] Fabio Bellini. Lecture notes - the vasicek model. 2009.

[19] K. Bernt. Stochastic differential equations. *An Introduction with Applications. Berlin: Springer*, 1995.

[20] J.C. Hull and A.D. White. A note on the models of hull and white for pricing options on the term structure. *The Journal of Fixed Income*, 5(2):97–102, 1995.

[21] Jesper Lund. Lecture notes - obligationer og optioner: Teori, empiri og praksis. 2001.

[22] I. Nelken. *Volatility in the Capital Markets*. Glenlake Business Monographs. Taylor & Francis, 1997.

[23] L.B.G. Andersen and V.V. Piterbarg. *Interest Rate Modeling: Products and Risk Management*, volume 1. Atlantic Financial Press, 2010.

[24] Christa Cuchiero. Affine interest rate models - theory and practice. Master's thesis, University of Vienna - Faculty of Mathematics, Vienna, 2006.

[25] Leonardo Volpi. Didactic optimization tool for excel®.

[26] Leonardo Volpi. *Optimization, Nonlinear Fitting and Equations Solving*. Foxes Team - Software for Numerical Analysis, Italy, 2. edition, 2006.

[27] Sergey Anatolyevich Bochkanov. Alglib® - cross-platform numerical analysis and data processing library, 2012.

[28] Niels Elken Sonderby. Simulation based pricing of american options in a model with stochastic volatility and jumps. Master's thesis, Copenhagen Business School - Faculty of Finance, Copenhagen, 2003.

[29] Simon Benninga. *Financial Modelling*, volume 3. The MIT Press, 2008.

Appendices

Appendix A

Dataset

A.1 Market Data

Table A.1: Market Data - Yield and Discount Curves. Conventions: Q, act.360.

Date	d(T)	Spot	FW	Date	d(T)	Spot	FW	Date	d(T)	Spot	FW
30.04.2005	0.993281	2.7270	2.7270	30.04.2015	0.627425	4.4856	5.0874	30.04.2025	0.362891	4.9353	5.3130
31.07.2005	0.985334	2.9386	3.1433	31.07.2015	0.619336	4.4998	5.0776	31.07.2025	0.358009	4.9398	5.3000
31.10.2005	0.976914	3.0800	3.3582	31.10.2015	0.611282	4.5144	5.1220	31.10.2025	0.353203	4.9440	5.2885
31.01.2006	0.968008	3.2070	3.5837	31.01.2016	0.603180	4.5306	5.2211	31.01.2026	0.348473	4.9480	5.2756
30.04.2006	0.959099	3.3114	3.7400	30.04.2016	0.595156	4.5487	5.3568	30.04.2026	0.343966	4.9516	5.2657
31.07.2006	0.949710	3.4021	3.8495	31.07.2016	0.586899	4.5688	5.4668	31.07.2026	0.339379	4.9552	5.2534
31.10.2006	0.940217	3.4784	3.9310	31.10.2016	0.578660	4.5895	5.5321	31.10.2026	0.334863	4.9585	5.2419
31.01.2007	0.930649	3.5444	4.0025	31.01.2017	0.570508	4.6097	5.5518	31.01.2027	0.330415	4.9616	5.2325
30.04.2007	0.921355	3.6004	4.0598	30.04.2017	0.562757	4.6281	5.5332	30.04.2027	0.326177	4.9645	5.2217
31.07.2007	0.911741	3.6513	4.1046	31.07.2017	0.554889	4.6458	5.5095	31.07.2027	0.321861	4.9672	5.2123
31.10.2007	0.902115	3.6974	4.1533	31.10.2017	0.547162	4.6625	5.4873	31.10.2027	0.317610	4.9699	5.2026
31.01.2008	0.892461	3.7405	4.2101	31.01.2018	0.539569	4.6781	5.4682	31.01.2028	0.313423	4.9723	5.1928
30.04.2008	0.883025	3.7793	4.2517	30.04.2018	0.532345	4.6923	5.4521	30.04.2028	0.309387	4.9745	5.1843
31.07.2008	0.873449	3.8144	4.2667	31.07.2018	0.524999	4.7062	5.4373	31.07.2028	0.305321	4.9767	5.1767
31.10.2008	0.863910	3.8468	4.2970	31.10.2018	0.517769	4.7194	5.4263	31.10.2028	0.301315	4.9787	5.1681
31.01.2009	0.854369	3.8782	4.3456	31.01.2019	0.510651	4.7320	5.4168	31.01.2029	0.297368	4.9806	5.1597
30.04.2009	0.845111	3.9086	4.4070	30.04.2019	0.503866	4.7436	5.4105	30.04.2029	0.293604	4.9824	5.1527
31.07.2009	0.835517	3.9399	4.4676	31.07.2019	0.496951	4.7551	5.4074	31.07.2029	0.289769	4.9840	5.1448
31.10.2009	0.825912	3.9709	4.5244	31.10.2019	0.490134	4.7662	5.4049	31.10.2029	0.285988	4.9856	5.1395
31.01.2010	0.816310	4.0014	4.5759	31.01.2020	0.483408	4.7770	5.4070	31.01.2030	0.282262	4.9871	5.1316
30.04.2010	0.807035	4.0303	4.6222	30.04.2020	0.476914	4.7872	5.4099	30.04.2030	0.278707	4.9884	5.1268
31.07.2010	0.797471	4.0594	4.6650	31.07.2020	0.470362	4.7974	5.4131	31.07.2030	0.275084	4.9897	5.1200
31.10.2010	0.787940	4.0877	4.7049	31.10.2020	0.463897	4.8073	5.4157	31.10.2030	0.271512	4.9910	5.1144
31.01.2011	0.778449	4.1151	4.7420	31.01.2021	0.457520	4.8168	5.4164	31.01.2031	0.267989	4.9921	5.1106
30.04.2011	0.769312	4.1409	4.7758	30.04.2021	0.451433	4.8259	5.4176	30.04.2031	0.264628	4.9932	5.1051
31.07.2011	0.759915	4.1668	4.8091	31.07.2021	0.445227	4.8349	5.4167	31.07.2031	0.261200	4.9942	5.1021
31.10.2011	0.750566	4.1921	4.8440	31.10.2021	0.439107	4.8436	5.4161	31.10.2031	0.257819	4.9952	5.0982
31.01.2012	0.741267	4.2168	4.8783	31.01.2022	0.433073	4.8521	5.4144	31.01.2032	0.254484	4.9961	5.0947
30.04.2012	0.732219	4.2405	4.9125	30.04.2022	0.427318	4.8600	5.4113	30.04.2032	0.251264	4.9970	5.0935
31.07.2012	0.723024	4.2642	4.9450	31.07.2022	0.421453	4.8679	5.4079	31.07.2032	0.248017	4.9978	5.0897
31.10.2012	0.713891	4.2872	4.9743	31.10.2022	0.415674	4.8755	5.4027	31.10.2032	0.244813	4.9987	5.0880
31.01.2013	0.704825	4.3097	5.0012	31.01.2023	0.409979	4.8828	5.3982	31.01.2033	0.241651	4.9995	5.0870
30.04.2013	0.696123	4.3309	5.0251	30.04.2023	0.404550	4.8896	5.3922	30.04.2033	0.238632	5.0002	5.0853
31.07.2013	0.687191	4.3523	5.0533	31.07.2023	0.399021	4.8963	5.3849	31.07.2033	0.235551	5.0009	5.0851
31.10.2013	0.678318	4.3734	5.0854	31.10.2023	0.393575	4.9028	5.3775	31.10.2033	0.232510	5.0017	5.0847
31.01.2014	0.669495	4.3944	5.1232	31.01.2024	0.388213	4.9090	5.3677	31.01.2034	0.229508	5.0024	5.0851
30.04.2014	0.661008	4.4146	5.1604	30.04.2024	0.383046	4.9147	5.3596	30.04.2034	0.226641	5.0031	5.0847
31.07.2014	0.652319	4.4348	5.1778	31.07.2024	0.377846	4.9203	5.3485	31.07.2034	0.223713	5.0038	5.0882
31.10.2014	0.643758	4.4538	5.1695	31.10.2024	0.372727	4.9257	5.3376	31.10.2034	0.220823	5.0045	5.0879
31.01.2015	0.635366	4.4710	5.1346	31.01.2025	0.367689	4.9307	5.3252	31.01.2035	0.217969	5.0052	5.0903

The discount curve information is subtracted from the US Market using Bloomberg™ >SWPM<. Zero coupon and Forward curves calculated using the results from section 1. Value date 31.01.2005.

Table A.2: Forward Swap Matrix - generated using the algorithm `basic_rates/PARSWAP` presented in appendix C.1.1. Conventions: Q, act.360.

Tenor	Spot	1Y	2Y	3Y	4Y	5Y	6Y	7Y	8Y	9Y	10Y	12Y	15Y	20Y	25Y
1Y	3.2172	3.9000	4.1536	4.3135	4.5194	4.7113	4.8563	4.9894	5.1044	5.1946	5.1601	5.5382	5.4512	5.3300	5.1513
2Y	3.5522	4.0242	4.2319	4.4140	4.6131	4.7821	4.9213	5.0453	5.1483	5.1778	5.3328	5.5054	5.4523	5.3074	5.1425
3Y	3.7448	4.1168	4.3235	4.5085	4.6903	4.8479	4.9792	5.0925	5.1520	5.2844	5.3975	5.4863	5.4492	5.2867	5.1362
4Y	3.8785	4.2109	4.4139	4.5893	4.7599	4.9073	5.0289	5.1081	5.2358	5.3427	5.4143	5.4782	5.4419	5.2682	5.1320
5Y	3.9959	4.3021	4.4943	4.6619	4.8221	4.9590	5.0525	5.1812	5.2899	5.3655	5.4197	5.4738	5.4307	5.2516	5.1300
6Y	4.1024	4.3842	4.5674	4.7268	4.8766	4.9884	5.1199	5.2329	5.3160	5.3770	5.4243	5.4692	5.4161	5.2370	n/a
7Y	4.1960	4.4591	4.6333	4.7840	4.9111	5.0525	5.1705	5.2616	5.3315	5.3859	5.4278	5.4630	5.4001	5.2244	n/a
8Y	4.2800	4.5268	4.6919	4.8231	4.9738	5.1025	5.2013	5.2804	5.3438	5.3928	5.4293	5.4544	5.3839	5.2138	n/a
9Y	4.3552	4.5873	4.7342	4.8853	5.0241	5.1352	5.2229	5.2955	5.3535	5.3972	5.4282	5.4434	5.3681	5.2053	n/a
10Y	4.4221	4.6327	4.7957	4.9363	5.0589	5.1592	5.2406	5.3077	5.3604	5.3988	5.4244	5.4310	5.3531	5.1987	n/a
12Y	4.5392	4.7466	4.8855	5.0020	5.1077	5.1959	5.2665	5.3230	5.3652	5.3934	5.4099	5.4052	5.3267	n/a	n/a
15Y	4.6752	4.8479	4.9656	5.0652	5.1543	5.2266	5.2820	5.3243	5.3541	5.3725	5.3816	5.3697	5.2973	n/a	n/a
20Y	4.7974	4.9379	5.0311	5.1084	5.1764	5.2306	5.2714	5.3020	5.3231	5.3356	5.3412	n/a	n/a	n/a	n/a
25Y	4.8473	4.9683	5.0479	5.1140	5.1726	5.2196	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
30Y	4.8693	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

Underlying data subtracted from the US Market using Bloomberg™ >SWPM<. Value date 31.01.2005.

Table A.3: Market Data - 3MvXY Cap (Floor) | Black76-vol Surface | Relative strike

Tenor	-300	-250	-200	-150	-100	-75	-50	-25	ATM	25	50	75	100	150	200	250	300
1Y	36.16	31.01	25.86	21.99	19.52	18.43	17.34	16.58	16.13	15.81	15.56	15.45	15.45	15.45	15.45	15.94	16.25
2Y	42.23	37.14	32.53	29.08	25.66	23.87	22.79	22.39	22.36	22.43	22.56	22.63	22.61	22.43	22.26	22.05	21.90
3Y	41.63	36.54	32.53	29.56	26.48	25.20	24.55	24.22	24.06	23.94	23.85	23.70	23.50	23.16	22.82	22.55	22.33
4Y	38.83	34.34	31.09	28.75	26.14	25.30	24.77	24.43	24.16	23.97	23.77	23.52	23.24	22.85	22.43	22.13	21.82
5Y	37.67	33.37	30.41	28.23	25.92	25.24	24.75	24.40	24.11	23.88	23.61	23.32	23.04	22.60	22.16	21.83	21.51
6Y	35.96	32.17	29.60	27.53	25.56	24.97	24.54	24.17	23.86	23.59	23.31	23.00	22.76	22.31	21.90	21.60	21.32
7Y	34.46	31.04	28.76	26.76	25.05	24.48	24.11	23.74	23.41	23.12	22.81	22.49	22.26	21.78	21.38	21.06	20.79
8Y	33.38	30.19	28.08	26.11	24.60	24.10	23.68	23.28	22.96	22.64	22.32	22.01	21.77	21.26	20.86	20.51	20.24
9Y	32.22	29.28	27.37	25.45	24.05	23.58	23.15	22.75	22.40	22.09	21.77	21.49	21.23	20.69	20.31	19.96	19.67
10Y	31.24	28.53	26.75	24.84	23.60	23.17	22.74	22.33	21.95	21.62	21.28	21.02	20.76	20.22	19.84	19.47	19.21
12Y	29.77	27.38	25.71	23.94	22.84	22.40	21.96	21.56	21.20	20.84	20.50	20.21	19.93	19.37	19.04	18.70	18.50
15Y	28.55	26.29	24.53	22.96	21.88	21.40	20.99	20.61	20.25	19.89	19.61	19.33	19.06	18.58	18.20	17.88	17.63
20Y	25.32	23.53	21.96	20.66	19.68	19.25	18.85	18.47	18.15	17.74	17.47	17.19	16.91	16.48	16.11	15.83	15.61
25Y	23.84	22.21	20.68	19.51	18.58	18.17	17.80	17.43	17.10	16.73	16.47	16.20	15.94	15.54	15.20	14.96	14.76
30Y	23.02	21.45	19.97	18.88	17.96	17.55	17.18	16.81	16.45	16.17	15.91	15.64	15.38	15.01	14.69	14.48	14.31

Data subtracted from the US Market using Bloomberg™ >SWPM<. Value date 31.01.2005.

Table A.4: Market Data - 3MvXY Cap (Floor) | Black76-vol Surface | Absolute strike

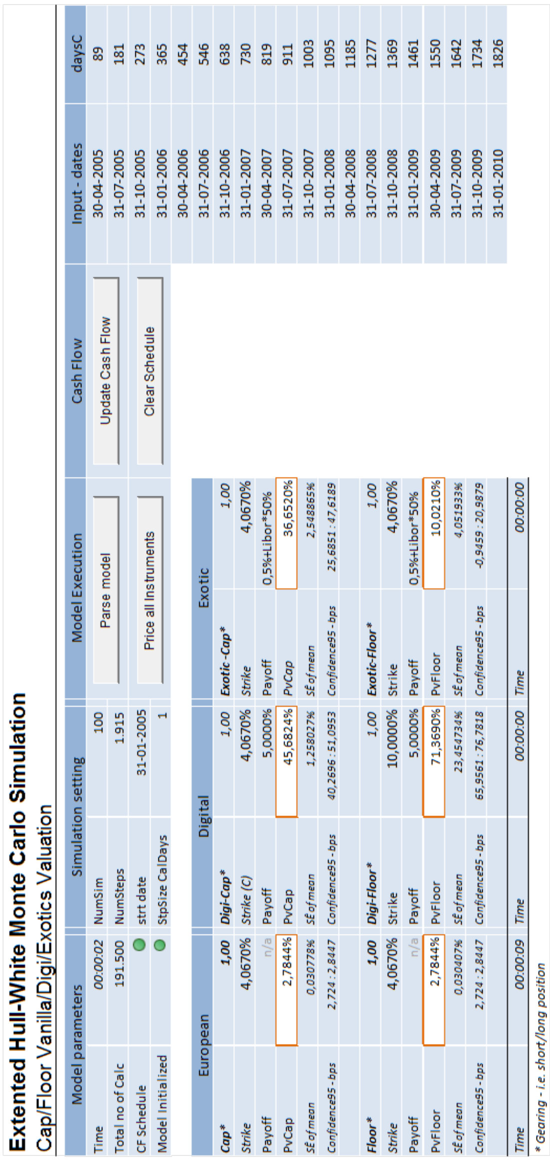
Tenor	-300	-250	-200	-150	-100	-75	-50	-25	ATM	25	50	75	100	150	200	250	300
1Y	36.16	31.01	25.86	21.99	19.52	18.43	17.34	16.58	16.13	15.81	15.56	15.45	15.45	15.45	15.45	15.94	16.25
2Y	42.23	37.14	32.53	29.08	25.66	23.87	22.79	22.39	22.36	22.43	22.56	22.63	22.61	22.43	22.26	22.05	21.90
3Y	41.63	36.54	32.53	29.56	26.48	25.20	24.55	24.22	24.06	23.94	23.85	23.70	23.50	23.16	22.82	22.55	22.33
4Y	38.83	34.34	31.09	28.75	26.14	25.30	24.77	24.43	24.16	23.97	23.77	23.52	23.24	22.85	22.43	22.13	21.82
5Y	37.67	33.37	30.41	28.23	25.92	25.24	24.75	24.40	24.11	23.88	23.61	23.32	23.04	22.60	22.16	21.83	21.51
6Y	35.96	32.17	29.60	27.53	25.56	24.97	24.54	24.17	23.86	23.59	23.31	23.00	22.76	22.31	21.90	21.60	21.32
7Y	34.46	31.04	28.76	26.76	25.05	24.48	24.11	23.74	23.41	23.12	22.81	22.49	22.26	21.78	21.38	21.06	20.79
8Y	33.38	30.19	28.08	26.11	24.60	24.10	23.68	23.28	22.96	22.64	22.32	22.01	21.77	21.26	20.86	20.51	20.24
9Y	32.22	29.28	27.37	25.45	24.05	23.58	23.15	22.75	22.40	22.09	21.77	21.49	21.23	20.69	20.31	19.96	19.67
10Y	31.24	28.53	26.75	24.84	23.60	23.17	22.74	22.33	21.95	21.62	21.28	21.02	20.76	20.22	19.84	19.47	19.21
12Y	29.77	27.38	25.71	23.94	22.84	22.40	21.96	21.56	21.20	20.84	20.50	20.21	19.93	19.37	19.04	18.70	18.50
15Y	28.55	26.29	24.53	22.96	21.88	21.40	20.99	20.61	20.25	19.89	19.61	19.33	19.06	18.58	18.20	17.88	17.63
20Y	25.32	23.53	21.96	20.66	19.68	19.25	18.85	18.47	18.15	17.74	17.47	17.19	16.91	16.48	16.11	15.83	15.61
25Y	23.84	22.21	20.68	19.51	18.58	18.17	17.80	17.43	17.10	16.73	16.47	16.20	15.94	15.54	15.20	14.96	14.76
30Y	23.02	21.45	19.97	18.88	17.96	17.55	17.18	16.81	16.45	16.17	15.91	15.64	15.38	15.01	14.69	14.48	14.31

Data subtracted from the US Market using Bloomberg™ >SWPM<. Value date 31.01.2005.

Appendix B

Graphical User Interface

Figure B.1: Graphical user interface [GUI]



Appendix C

VBA Source Script

As a general setting, the `Option Explicit` and `Option Base 1` are applied in the preamble of each module.

C.1 Basic Rates

C.1.1 ParSwap

```
Function ParSwap(range As range, Optional StarT As Long, Optional maturity As Long)
Application.ScreenUpdating = False
'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Dim i As Integer, rows As Integer, strt As Long, mat As Long, dayC() As Double
Dim annVec() As Double, a As Double, pvflVec() As Double, PVfl As Double
rows = range.rows.Count
ReDim dayC(rows), annVec(rows), pvflVec(rows)

' Determines start and maturity points in the grid
For i = 1 To rows
    'Starting point if user defined
    strt = strt + Abs(range(i, 1) = StarT) * i
    'Maturity point if user defined
    mat = mat + Abs(range(i, 2) = maturity) * i
Next i

If IsMissing(StarT) Or (StarT = 0) Then
'Starting point = 1 row if no user input
strt = 1
End If
If IsMissing(maturity) Or maturity = 0 Then
'Maturity = rows if no user input
mat = rows
End If


For i = strt To mat Step 1
    'Calcs the daycount faction with a fixed .360 base
    dayC(i) = (range(i, 2) - range(i, 1)) / 360
    'Calcs the annuity vector and multiplies the notional amount to a account for
        amortizations
    annVec(i) = dayC(i) * range(i, 6) * range(i, 4)
    If (i > 1) Then
    'Calcs PV Floating leg vector and multiplies the notional amount to a account for
        amortizations
    pvflVec(i) = (range(i - 1, 6) - range(i, 6)) * range(i, 4)
    Else
    pvflVec(i) = (1 - range(i, 6)) * range(i, 4)
    End If
Next i
a = WorksheetFunction.Sum(annVec)
PVfl = WorksheetFunction.Sum(pvflVec)
```

```
Application.ScreenUpdating = True
ParSwap = PVfl / a
End Function
```

C.1.2 ResetRates

```
Function ResetRates(range As range, numDiscColumn As Integer)

Dim j As Integer, i As Integer, rows As Integer, dayC() As Double, fwdiscVec() As Double
Dim ResetRatesVec() As Double
rows = range.rows.Count
j = numDiscColumn
ReDim dayC(rows), fwdiscVec(rows), ResetRatesVec(rows, 1)

'Calcs the daycount faction with a fixed .360 base
  For i = 1 To rows Step 1
    dayC(i) = (range(i, 2) - range(i, 1)) / 360
  Next i

'Calcs the forward discount factors
  For i = 1 To rows Step 1
    If i = 1 Then
    fwdiscVec(i) = range(i, j)
    Else
    fwdiscVec(i) = range(i, j) / range(i - 1, j)
    End If
  Next i

' Calcs each Simply compounded forward/reset rate using the relevant daycount and forward
' discount factors (Filipovic - Term-Structure Models 2.2 p.6)
  For i = 1 To rows Step 1
    ResetRatesVec(i, 1) = 1 / dayC(i) * (1 / fwdiscVec(i) - 1)
  Next i

ResetRates = ResetRatesVec
End Function
```

C.1.3 ZeroRates

```
Function ZeroRates(range As range)

Dim i As Integer, rows As Integer, dayCsum As Double, dayC As Double
Dim ZeroRatesVec() As Double
rows = range.rows.Count
ReDim dayCcum(rows), ZeroRatesVec(rows, 1)

dayCsum = 0

'Calcs each continiously componded ZeroRate
  For i = 1 To rows Step 1
    dayC = (range(i, 2) - range(i, 1)) / 360
    dayCsum = dayCsum + dayC
    ZeroRatesVec(i, 1) = -WorksheetFunction.Ln(range(i, 6)) / dayCsum
  Next i

ZeroRates = ZeroRatesVec
End Function
```

C.1.4 ForwardDisc

```
Function ForwardDisc(range As range, numDiscColumn As Integer)

Dim j As Integer, i As Integer, rows As Integer, dayC() As Double, fwdiscVec() As Double
rows = range.rows.Count
j = numDiscColumn
ReDim dayC(rows), fwdiscVec(rows, 1)

'Calcs the daycount faction with a fixed .360 base
  For i = 1 To rows Step 1
    dayC(i) = (range(i, 2) - range(i, 1)) / 360
  Next i
```

```
'Calcs the forward discount factors
  For i = 1 To rows Step 1
    If i = 1 Then
    fwdiscVec(i, 1) = range(i, j)
    Else
    fwdiscVec(i, 1) = range(i, j) / range(i - 1, j)
    End If
  Next i

ForwardDisc = fwdiscVec
End Function
```

C.2 Standard Vanilla Models

C.2.1 BblackCall

```
Function BblackCall(F0 As Double, K As Double, BlackVol As Double, StarT As Double, OptMat
    As Double)

Dim t As Double, D1 As Double, D2 As Double, Nd1 As Double, Nd2 As Double

    t = (OptMat - StarT) / 360

    D1 = (Log(F0 / K) + 0.5 * BlackVol ^ 2 * t) / (BlackVol * t ^ 0.5)
    D2 = (Log(F0 / K) - 0.5 * BlackVol ^ 2 * t) / (BlackVol * t ^ 0.5)

    Nd1 = WorksheetFunction.NormSDist(D1)
    Nd2 = WorksheetFunction.NormSDist(D2)

BblackCall = (F0 * Nd1 - K * Nd2)

End Function
```

C.2.2 BblackPut

```
Function BblackPut(F0 As Double, K As Double, BlackVol As Double, StarT As Double, OptMat As
     Double)

Dim t As Double, D1 As Double, D2 As Double, Nd1 As Double, Nd2 As Double

    t = (OptMat - StarT) / 360

    D1 = (Log(F0 / K) + 0.5 * BlackVol ^ 2 * t) / (BlackVol * t ^ 0.5)

    D2 = (Log(F0 / K) - 0.5 * BlackVol ^ 2 * t) / (BlackVol * t ^ 0.5)

    Nd1 = WorksheetFunction.NormSDist(-D1)
    Nd2 = WorksheetFunction.NormSDist(-D2)

BblackPut = (-F0 * Nd1 + K * Nd2)

End Function
```

C.2.3 BnormCall

```
Function BnormCall(F0 As Double, K As Double, BpVol As Double, StarT As Double, OptMat As
    Double)

Dim t As Double, D1 As Double, D2 As Double, Nd1 As Double, Nd2 As Double

    t = (OptMat - StarT) / 360

    D1 = (F0 - K) / (BpVol * Sqr(t))
    D2 = -(F0 - K) / (BpVol * Sqr(t))

    Nd1 = WorksheetFunction.NormSDist(D1)
    Nd2 = 1 / Sqr(2 * WorksheetFunction.Pi) * Exp(-(D1 ^ 2) / 2)

BnormCall = BpVol * Sqr(t) * (D1 * Nd1 + Nd2)

End Function
```

C.2.4 BnormPut

```
Function BnormPut(F0 As Double, K As Double, BpVol As Double, StarT As Double, OptMat As
    Double)

Dim t As Double, D1 As Double, D2 As Double, Nd1 As Double, Nd2 As Double

    t = (OptMat - StarT) / 360

    D1 = (F0 - K) / (BpVol * Sqr(t))
    D2 = -(F0 - K) / (BpVol * Sqr(t))
```

```
    Nd1 = WorksheetFunction.NormSDist(D2)
    Nd2 = 1 / Sqr(2 * WorksheetFunction.Pi) * Exp(-(D2 ^ 2) / 2)

BnormPut = BpVol * Sqr(t) * (D2 * Nd1 + Nd2)

End Function
```

C.2.5 CapBS

```
Function CapBS(range As range, Strike As Double, Optional BlackVol As Double)

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Analytical Cap pricing using the Black76 model - Filipovic '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Utilize the function BblackCall(F0, K, BlackVol, Start, OptMat)

'Input Range defined as Rows(i) x Columns(8)
'range(i,1) = Date Option Expiry
'range(i,2) = Pay date caplet - end period
'range(i,3) = Days or blank (not used)
'range(i,4) = Notional amount
'range(i,5) = Strike (%) - not used
'range(i,6) = Vol (black/bp flat or spot vols)
'range(i,7) = Reset rate
'range(i,8) = Discount factor

Dim i As Integer, dayC() As Double, rows As Integer, Columns As Integer
Dim sumCaplet() As Double
rows = range.rows.Count
ReDim sumCaplet(rows), dayC(rows)

If BlackVol = 0 Then     ' Spot vols / from Scheme
  For i = 2 To rows Step 1
    'Cals daycount frac based on .360 conventions
    dayC(i) = (range(i, 2) - range(i, 1)) / 360

   'Calcs: disc * notional frac * dayC * BblackCall(F0, K, BlackVol, Start, OptMat)
   sumCaplet(i) = range(i, 8) * range(i, 4) / range(2, 4) * dayC(i) * BblackCall(range(i, 7)
       , Strike, range(i, 6), range(1, 1), range(i, 1))
Next i

' Flat Vol / manual input
Else
  For i = 2 To rows Step 1
    'Cals daycount frac based on .360 conventions
    dayC(i) = (range(i, 2) - range(i, 1)) / 360

   'Calcs: disc * notional frac * dayC * BblackCall(F0, K, BlackVol, Start, OptMat)
   sumCaplet(i) = range(i, 8) * range(i, 4) / range(2, 4) * dayC(i) * BblackCall(range(i, 7)
       , Strike, BlackVol, range(1, 1), range(i, 1))
 Next i

End If

CapBS = WorksheetFunction.Sum(sumCaplet)

End Function
```

C.2.6 FloorBS

```
Function FloorBS(range As range, Strike As Double, Optional BlackVol As Double)

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Analytical Floor pricing using the Black76 model - Filipovic '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Utilize the function BblackPut(F0, K, BlackVol, Start, OptMat)

'Input Range defined as Rows(i) x Columns(8)
'range(i,1) = Date Option Expiry
'range(i,2) = Pay date Floorlet - end period
```

```
'range(i,3) = Days or blank (not used)
'range(i,4) = Notional amount
'range(i,5) = Strike (%) - not used
'range(i,6) = Vol (black/bp flat or spot vols)
'range(i,7) = Reset rate
'range(i,8) = Discount factor

Dim i As Integer, dayC() As Double, rows As Integer, Columns As Integer
Dim sumFloorlet() As Double
rows = range.rows.Count
ReDim sumFloorlet(rows), dayC(rows)
'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

If BlackVol = 0 Then    ' Spot vols / from Scheme
  For i = 2 To rows Step 1
    ' Cals daycount frac based on .360 conventions
    dayC(i) = (range(i, 2) - range(i, 1)) / 360

   'Calcs: disc * notional frac * dayC * BblackPut(F0, K, BlackVol, Start, OptMat)
   sumFloorlet(i) = range(i, 8) * range(i, 4) / range(2, 4) * dayC(i) * BblackPut(range(i,
       7), Strike, range(i, 6), range(1, 1), range(i, 1))
Next i

' Flat Vol / manual input
Else
  For i = 2 To rows Step 1
    'Cals daycount frac based on .360 conventions
    dayC(i) = (range(i, 2) - range(i, 1)) / 360

   'Calcs: disc * notional frac * dayC * BblackPut(F0, K, BlackVol, Start, OptMat)
   sumFloorlet(i) = range(i, 8) * range(i, 4) / range(2, 4) * dayC(i) * BblackPut(range(i,
       7), Strike, BlackVol, range(1, 1), range(i, 1))
 Next i

End If

FloorBS = WorksheetFunction.Sum(sumFloorlet)

End Function
```

C.3 HWExtV model - Analytical setup

C.3.1 BHW

```
Function BHW(strt As Double, mat As Double)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs the B(t,T) parameter in the analytical solution of the Hull White model.  '
' J.Lund Obligationer og Optioner, Teori, empiri og praksis p.39.                 '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Dim Kappa As Double
Kappa = Sheets("Hull White Cal").range("Y6").Value2    ' Source calibrated HW Kappa
'Application.Volatile (Kappa)

Dim tau
tau = (mat - strt)
BHW = (1 - Exp(-Kappa * tau)) / Kappa

End Function
```

C.3.2 PhiHWExtV

```
Function PhiHWExtV(t As Double) As Double

''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs Phi() in the Hull White Extended Vasicek model '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False

' -- Exogenous variables and values --
Dim Kappa As Double, SourceVol As Variant, VolDates As Variant, IndexBaseNo As Long

' Source calibrated HW Kappa from the sheet "Hull White Cal"
Kappa = Sheets("Hull White Cal").range("Y6").Value2
' Source calibrated HW Vols from the sheet "Hull White Cal"
SourceVol = Sheets("Hull White Cal").range("X6:X18").Value2
' Source the HW vols calibration dates (=Option maturity) from the sheet "Hull White Cal"
VolDates = Sheets("Hull White Cal").range("DC7:DC19").Value2
' Sets the IndexBaseNo equal to the trade date of the Market Data - sheet "Outline MData"
IndexBaseNo = Sheets("Outline MData").range("B5:B5").Value2

'Application.Volatile (Kappa)

' -- Endogenous variables --
Dim i As Integer, j As Integer, rowsVol As Integer, Sigma() As Variant
Dim EndVolDates As Integer, Schedule() As Double, EndSchedule As Integer, Phi As Double

' i, j          = Control counters
' rowsVol       = Count of rows in input Vol vector
' EndVolDates   = "Adjusted" end date from the input VolDates
' Sigma         = Adjusted vector of calibrated Sigma's
' Schedule      = Calibrated sigma dates including the case specific end date t
' EndSchedule   = Count of rows in the schedule vector
' Phi           = Function Output

rowsVol = Application.Count(SourceVol)
' Redimensions Schedule() to fit possible max.count of dates (full dimension not always used
    )
ReDim Schedule(0 To rowsVol + 1)
' Redimensions Vol() to fit possible max.count of dates (full dimension not always used)
ReDim Sigma(1 To rowsVol + 1)

' -- Start of code --
    ' Sets Sigma = SourceVol
    For j = 1 To rowsVol
        Sigma(j) = SourceVol(j, 1)
        VolDates(j, 1) = (VolDates(j, 1) - IndexBaseNo) / 360
    Next
    ' Extrapolates all volatilities > rowsVol at a constant equal to the last calibrated
    ' Sigma element
```

```
    Sigma(rowsVol + 1) = SourceVol(rowsVol, 1)

    ' Checks whether t > max Vol date and determines the last date in the vol input vector
    If t <= VolDates(rowsVol, 1) Then

    i = 1
    ' If t < max Vol date
    Do Until VolDates(i, 1) >= t
        i = i + 1
        Loop
        ' Adjusts the end to the step before t, so that the distance in the last step always
        ' is determined subsequently
        EndVolDates = i - 1
    Else
        ' If t > max Vol date, then EndVolDates = rowsVol (as the last step lies AFTER the
        ' max Vol date)
        EndVolDates = rowsVol
    End If

    ' Updates the Schedule vector with values from VolDates until the step before t
    ' (=EndVolDates)
    For i = 1 To EndVolDates
        Schedule(i) = VolDates(i, 1)
    Next
        ' Sets the front equal to the trade date of the Market Data
        'Schedule(0) = IndexBaseNo
        ' Sets the back end equal to t
        Schedule(i) = t
        EndSchedule = i

    ' Calcs Phi(t)
    For j = 1 To EndSchedule

    Phi = Phi + (1 / (2 * Kappa)) * Sigma(j) ^ 2 * (Exp(-2 * Kappa * ((t - Schedule(j)))) -
        Exp(-2 * Kappa * ((t - Schedule(j - 1)))))

    Next

PhiHWExtV = Phi

End Function
```

C.3.3 ZCB-HW

```
Function ZCB_HW(P_T As Double, AB, PT As Double, rt As Double, ft As Double, t As Double)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs the ZeroCouponBond P(t,T) at a future time t= 0 to T.          '
' J.Lund Obligationer og Optioner, Teori, empiri og praksis (62) p.36. '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' PT         = P(0,T) ZCB for the final maturity T
' Pt         = P(0,t) ZCB for the subject horizon t (if t=0 then P(0,t)=1 obviously)
' rt         = The time t instantaneous short rate ????
' ft         = f(0,t) Forward rate for the subject horizon (if t=0 then f(0,t)=r(t) the
'              instantaneous short rate)
' PhiHWExtV = Hull White Phi(t) function
' t          = Evaluation date t
' B          = B(t,T) Note that as Kappa here is a constant, B(t,T) reduces according to
'              page 39 and outlined is the function BHW
' Strt       = Period start date
' mat        = Period maturity date

Dim b As Double

b = BHW(strt, mat) ' Kappa

ZCB_HW = (P_T / PT) * Exp(-0.5 * b ^ 2 * PhiHWExtV(t) + b * (r0 - ft))

End Function
```

C.3.4 OptZCBHW

```
Function OptZCBHW(PutCall As String, Ps As Double, PT As Double, K As Double, s As Double, t
     As Double, FaceValue As Double)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs the price of option on Zero Coupon Bonds in the Hull White model    '
' J.Lund Obligationer og Optioner, Teori, empiri og praksis (71)-(74) p.37. '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'PutCall    = String determine whether Put or Call option
'Ps         = P(0,t) OptMat
'PT         = P(0,T) matZCB
'K          = Strike price
's          = Option Maturity as dayCountFrac act.360 [s=t]
'T          = Maturity ZCB as dayCountFrac Act.360
'FaceValue  = Principal of the bond

Dim ZCB_s As Double, ZCB_T As Double, Wf As Double, D1 As Double

ZCB_T = PT
ZCB_s = Ps

Wf = Sqr((BHW(s, t) ^ 2) * PhiHWExtV(s))

D1 = (1 / Wf) * Log((FaceValue * ZCB_T) / (ZCB_s * K)) + Wf / 2

'Calcs the PUT / CALL value respectively
If StrComp(PutCall, "CALL", vbTextCompare) = 0 Then

OptZCBHW = FaceValue * ZCB_T * WorksheetFunction.NormSDist(D1) - K * ZCB_s *
    WorksheetFunction.NormSDist(D1 - Wf)

ElseIf StrComp(PutCall, "PUT", vbTextCompare) = 0 Then

OptZCBHW = K * ZCB_s * WorksheetFunction.NormSDist(-D1 + Wf) - FaceValue * ZCB_T *
    WorksheetFunction.NormSDist(-D1)

Else
MsgBox "Please choose either CALL or PUT"
End If

End Function
```

C.3.5 CapFloorReplicationHW

```
Function CapFloorReplicationHW(CapFloor As String, range As range, StrikeCapFloor As Double,
     Kappa As Double, Sigma As Variant) As Double

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Uses the function OptZCBHW() to establish prices on a portefolio of ZCB's              '
' Calcs the present value option premium in Upfront bp of the notional amount/FaceValue '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Input Range defined as Rows(i) x Columns(8)
'range(i,1) = Date Option Expiry
'range(i,2) = Date Maturity ZCB
'range(i,3) = Not used
'range(i,4) = Notional amount Cap-/Floorlets
'range(i,5) = Strike price Cap-/Floorlets
'range(i,6) = Not used
'range(i,7) = Not used
'range(i,8) = Discount factor

Dim i As Integer, dayC() As Double, s As Double, t As Double, rows As Integer
Dim sumOptionsZCB() As Double, K As Double, PutCall As String, FaceValue As Double
rows = range.rows.Count
ReDim sumOptionsZCB(rows), dayC(rows)

' Sets Cap/Floor replication
   If StrComp(CapFloor, "Cap", vbTextCompare) = 0 Then
   PutCall = "Put"
   ElseIf StrComp(CapFloor, "Floor", vbTextCompare) = 0 Then
   PutCall = "Call"
   Else
```

```
   MsgBox "Please choose 'Cap' or 'Floor' replication"
   End If

For i = 2 To rows Step 1

'Cals daycount frac based on .360 conventions
  dayC(i) = (range(i, 2) - range(i, 1)) / 360     ' DayCountFrac between t-T (where t=s here
      )
  s = (range(i, 1) - range(1, 1)) / 360           ' Option maturity s=t
  t = (range(i, 2) - range(1, 1)) / 360           ' matZCB

' Sets the underlying ZCB strike = notional frac - Hull 26.12 p. 620
   K = range(i, 4) / range(2, 4)

' Adjust the FaceValue/Notional amount according to - Hull 26.12 p. 620
    FaceValue = range(i, 4) / range(2, 4) * (1 + dayC(i) * StrikeCapFloor)

' Calcs Vector: OptZCBHW(PutCall, Ps, PT, K, s, T, FaceValue)
   sumOptionsZCB(i) = OptZCBHW(PutCall, range(i - 1, 8), range(i, 8), K, s, t, FaceValue)

 Next i

CapFloorReplicationHW = WorksheetFunction.Sum(sumOptionsZCB)

End Function
```

C.4 HWExtV Monte Carlo Simulation Source Code

C.4.1 Preample - Public Variables

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Public Pub_SimDates() As Integer, Pub_dt() As Double, Pub_df() As Double                         '
Public Pub_Sigma() As Double, Pub_r() As Double, Pub_EvntD() As Long, Pub_dfSim() As Double'
Public Pub_CfRange As String, Pub_AccrualD() As Integer, Pub_dAccrualD() As Integer             '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Pub_SimDates() = Vector of sorted simulation dates                      : Simulation_Dates  '
'                  (amount of days from tradedate, tradedate = 0)                             '
' Pub_dt()       = delta t(j) - (i.e. days/360)                           : Simulation_Dates  '
' Pub_df()       = Linear interpolated market discountfactors             : LIntPdf           '
' Pub_Sigma()    = Vol-Vector Stepvise constant function                  : VolInitialize     '
' Pub_r()        = Matrice of Instataneous short rates (Not used)         : Hull_White_MC_Sub '
' Pub_EvntD()    = Vector of eventdates sourced via Pub_CfRange           : Hull_White_MC_Sub '
'                  (amount of days from tradedate, tradedate = 0)                             '
' Pub_dfSim()    = Matrice of Stochastic discount factor                  : Hull_White_MC_Sub '
' Pub_CfRange    = Array-reference to vector of daysC (I.e. "K5:Kx")      : EventSchedule     '
' Pub_AccrualD() = Vector of accrual dates for Accrual derivatives        : AccrualEventDates '
'                  (amount of days from tradedate, tradedate = 0)                             '
' Pub_dAccrualD()= Vector of delta accrual dates, Accrual derivatives     : AccrualEventDates '
'                  (amount of days between two accrual dates = d2-d1)                         '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

C.4.2 EventSchedule

```
Sub EventSchedule()

''''''''''''''''''''''''''''''''
' Calcs the vector of eventdays '
''''''''''''''''''''''''''''''''

Sheets("MC_ExtVasi").range("B5").Value = 0     ' Reset MC Timer

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False

Dim Date1 As String, Date2 As String, Date3 As String, i As Long, daysC As String,
    FormatArray As String

  If range("J5") <> "" Then

  Date1 = "J5"
  i = 5

  Do
  Date2 = "J" & i
  daysC = "K" & i
  range(daysC) = range(Date2).Value - range("E7").Value

  Date3 = "J" & i + 1
  i = i + 1
  Loop Until range(Date3).Text = ""

  'range("B6") = Date1 & ":" & daysC     ' Output - not used, information only
  Pub_CfRange = "K5" & ":" & daysC       ' Output used as input in Sub Hull_White_MC_Sub()

  FormatArray = Date1 & ":" & daysC

  range("AC5:AD5").Select
  Selection.Copy
  range(FormatArray).Select
  Selection.PasteSpecial Paste:=xlPasteFormats

  range("J4").Select
  Selection.Copy
  range("K4").Select
  Selection.PasteSpecial Paste:=xlPasteFormats
  range("K4") = "daysC"

  End If
```

```
  Application.CutCopyMode = False
  Application.ScreenUpdating = True

  Sheets("MC_ExtVasi").range("B7") = 1
End Sub
```

C.4.3 Simulation-Dates

```
Sub Simulation_Dates()

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs the vector of Monte Carlo simulation dates                             '
' I.e. payment dayment dates, Accrual dates/fracstep and their 3M tenor pairs   '
' (using dateadd) to determine the correct/ 3M point                           '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False

' -- Input variables and values --
Dim EvntD As range, I7 As range, D8 As Integer, StpSize As Integer, StartT As Long

StpSize = range("E8").Value     ' StpSize = User defined stepsize in number of days
StartT = range("E7").Value      ' Trade date (today)
Set EvntD = range(Pub_CfRange)  ' Event days, or more precise payment dates

' -- Endogenous variables --
Dim NumEvntD As Integer, dayCVec() As Long, dayCfrac As Double, Tenor As Byte, i As Integer,
j As Integer, s As Long, maxfracstep As Integer, fracStp() As Integer, Pairs() As Integer
Static SimDates() As Integer, dt() As Double

'NumEvntD       = Total number of event days
'dayCVec        = Support vector containing collapsed arrays
'dayCfrac       = the day count fraction used throughout the Sub is prespecified, however
'                 readily changable by chg the value of this valiable once
'Tenor          = Tenor in month's - prespecified throughout the Sub to Libor 3M, however
'                 readily changable by chg the value of this valiable once
'i              = Counter event days / misc - used several times
'j              = Counter simulation dates
's              = Support counter
'maxfracstep    = Max steps in multiple of the StpSize
'fracStp        = Vector of steps multiple of StpSize
'Pairs          = Vector of steps pairing each element in the fracstp vector + x month
'                 (where x = Tenor). This ensures that their always will be exactly two df's
'                 (one in each end) delimiting each requested Libor fixing
'SimDates       = Vector of sorted simulation dates
'QuickSort1     = Sub QuickSort1 - algorithem sorting an array
'dt             = delta t(j)

' -- Initial control variable values set --
NumEvntD = EvntD.rows.Count

ReDim dayCVec(0 To NumEvntD)

dayCfrac = 1 / 360        ' Pre-set to act/360
Tenor = 3                 ' Pre-set to 3M Libor

' -- Start of source Code --
  For i = 1 To NumEvntD
      j = i
      dayCVec(j) = EvntD(i)
  Next

  ' Submits "dayCVec" = EvntD to the public variable Pub_EvntD
  Pub_EvntD = dayCVec
  ' Redims the dayCVec to 3x (max payment date daycount) to ensure ample elements for the
  ' following collaps arrays
  ReDim Preserve dayCVec(0 To (Application.WorksheetFunction.Max(EvntD) * 3))

  maxfracstep = Int((EvntD(NumEvntD)) / StpSize)  ' Sets max steps in multiple of the
      StpSize

  ' IF StpSize <= max count eventdays then calculates intermediate fracsteps and pairs
  If maxfracstep <> 0 Then
```

```
' Redims fracstp & Pairs to maxfracstep
ReDim fracStp(1 To maxfracstep), Pairs(1 To maxfracstep)

' Loops through elements in each vector fracstp & Pairs
For i = 1 To maxfracstep
    fracStp(i) = StpSize * i
    s = fracStp(i) + StartT
    ' Controls the paired dates through dateadd(Tenor)
    Pairs(i) = DateAdd("m", Tenor, s) - StartT
Next

' -- Merge arrays --
j = NumEvntD + 1
For i = 1 To maxfracstep     ' Loop merging dayCVec with fracstp
    dayCVec(j) = fracStp(i)
    j = j + 1
Next

For i = 1 To maxfracstep     ' Loop merging dayCVec with Pairs
    dayCVec(j) = Pairs(i)
    j = j + 1
Next

' Downscale elements in dayCVec to fit proper size + initialize output variable Simdates
ReDim Preserve dayCVec(0 To j - 1), SimDates(0 To j - 1)

' ELSE IF StpSize > max count eventdays then no intermediate fracsteps and the code sets
    the
' EventD = DayCVec = SimDates
Else

' Downscale elements in dayCVec to fit proper size + initialize the output variable
    Simdates
ReDim Preserve dayCVec(0 To j), SimDates(0 To j)

End If

' Sort all elements in dayCVec
Call QuickSort1(dayCVec)

' -- Remove duplicates --
j = 1
' Remove duplicates in dayCVec + write tofinal SimDates
For i = 1 To (UBound(dayCVec) - 1)
    If dayCVec(i) <> dayCVec(i + 1) Then
    SimDates(j) = dayCVec(i)
    j = j + 1
    End If
Next i

' Writes last/final element to SimDates
SimDates(j) = dayCVec(UBound(SimDates))

' Redims (downscale) elements in SimDates to fit proper size
ReDim Preserve SimDates(0 To j), dt(0 To j)

' Loop writes elements to dt multiplying by dayCfrac
For j = 1 To (UBound(SimDates))
    dt(j) = (SimDates(j) - SimDates(j - 1)) * dayCfrac
Next

' -- Output --
Pub_SimDates = SimDates     'Submits "SimDates" to a (module) public variable
Pub_dt = dt                 'Submits "dt" to a public variable

' -- Write to GUI --
' Feeds to spreadsheet: the total amount of simulation dates
range("E6").Value = UBound(SimDates)
' Feeds to the spreadsheet the total number of calculations
range("B6").Value = UBound(SimDates) * range("E5").Value

Application.ScreenUpdating = True
```

```
End Sub
```

C.4.4 AccrualEventDates

```
Sub AccrualEventDates()
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Calcs the vector of accrual dates used for Accrual derivatives '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False

' -- Input variables and values --
Dim EvntD As range, I7 As range, D8 As Integer, StpSize As Integer, StartT As Long

StpSize = range("E8").Value     ' StpSize = User defined stepsize in number of days
StartT = range("E7").Value      ' Trade date (today)
Set EvntD = range(Pub_CfRange)  ' Event days, or more precise payment dates

' -- Endogenous variables --
Dim NumEvntD As Integer, dayCVec() As Long, i As Integer, j As Integer, s As Long
Dim maxfracstep As Integer, fracStp() As Integer
Static AccrualD() As Integer, dAccrualD() As Integer

'NumEvntD       = Total number of event days
'dayCVec        = Support vector containing collapsed arrays
'i              = Counter event days / misc - used several times
'j              = Counter simulation dates
's              = Support counter
'maxfracstep    = Max steps in multiple of the StpSize
'fracStp        = Vector of steps multiple of StpSize
'AccrualD       = Vector of sorted simulation dates
'QuickSort1     = Sub QuickSort1 - algorithem sorting an array
'dAccrualD      = delta t(j) - i.e. d2-d1

' -- Initial control variable values set --
NumEvntD = EvntD.rows.Count
ReDim dayCVec(0 To NumEvntD)

' -- Start code --
For i = 1 To NumEvntD
    j = i
    dayCVec(j) = EvntD(i)
Next

' Submits "dayCVec" = EvntD to the public variable Pub_EvntD
Pub_EvntD = dayCVec
' Redims the dayCVec to 3x (max payment date daycount) to ensure ample elements for the
' following collaps arrays
ReDim Preserve dayCVec(0 To (Application.WorksheetFunction.Max(EvntD) * 3))

' Sets max steps in multiple of the StpSize
maxfracstep = Int((EvntD(NumEvntD)) / StpSize)

' IF StpSize <= max count eventdays then calculates intermediate fracsteps and pairs
If maxfracstep <> 0 Then

' Redims fracstp & Pairs to maxfracstep
ReDim fracStp(1 To maxfracstep), Pairs(1 To maxfracstep)

' Loops through elements in the vector fracstp
For i = 1 To maxfracstep
    fracStp(i) = StpSize * i * Abs(StpSize * i > EvntD(1))

Next

' -- Merge arrays --
j = NumEvntD + 1
' Loop merging dayCVec with fracstp
For i = 1 To maxfracstep
    dayCVec(j) = fracStp(i)
    j = j + 1
Next
```

```
' Downscale elements in dayCVec to fit proper size + initialize the output variable AccrualD
ReDim Preserve dayCVec(0 To j - 1), AccrualD(0 To j - 1)

' ELSE IF StpSize > max count eventdays then no intermediate fracsteps and the code sets the
' EventD = DayCVec = AccrualD
Else

' Downscale elements in dayCVec to fit proper size + initialize the output variable AccrualD
ReDim Preserve dayCVec(0 To j), AccrualD(0 To j)

End If

' Sort all elements in dayCVec via the third party code, QuickSort1()
Call QuickSort1(dayCVec)

' -- Remove duplicates --
j = 1
' Remove duplicates in dayCVec and write to final AccrualD
For i = 1 To (UBound(dayCVec) - 1)
    If dayCVec(i) <> dayCVec(i + 1) And dayCVec(i) > 0 Then
    AccrualD(j) = dayCVec(i)
    j = j + 1
    End If
Next i

' Writes last/final element to AccrualD
AccrualD(j) = dayCVec(UBound(AccrualD))

' Redims (downscale) elements in AccrualD to fit proper size
ReDim Preserve AccrualD(0 To j), dAccrualD(0 To j)

' Loop writes elements to dAccrualD multiplying by dayCfrac
For j = 1 To (UBound(AccrualD) - 1)
    dAccrualD(j) = (AccrualD(j + 1) - AccrualD(j))
Next

' -- Output --
Pub_AccrualD = AccrualD      'Submits "AccrualD" to a public variable
Pub_dAccrualD = dAccrualD    'Submits "dAccrualD" to a public variable

Application.ScreenUpdating = True

End Sub
```

C.4.5 VolInitialize

```
Sub VolInitialize()

'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' Vol-initialize sets Vol-vector of sigma(i) according to a Stepvise constant function
    '
' Ref: J. Lund 2001 - Obligationer og optioner, Teori, empiri og praksis formula (77)
    '
' The resulting Sigma(j) vector is subsequently sourced to the public variable Pub_Sigma(j)
    '
'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''


Application.ScreenUpdating = False

' -- Exogenous variables and values --
Dim Vol As Variant, VolDates As Variant, IndexBaseNo As Long, SimDates() As Integer

' Source Calibrated Hull White Vols from sheet "Hull White Cal"
Vol = Sheets("Hull White Cal").range("X6:X18").Value2
' Sets date delimitor for the Vol-step vector. NB: Limit-dates equal to option maturities
VolDates = Sheets("Hull White Cal").range("DC7:DC19").Value2
' Sets the IndexBaseNo equal to the trade date of the Market Data
IndexBaseNo = Sheets("Outline MData").range("B5:B5").Value2
```

```
' Source list of simulation dates from public variable "Pub_SimDates"
SimDates = Pub_SimDates

' -- Endogenous variables --
Dim i As Integer, j As Integer, NumOfSimDates As Integer, rows As Integer, x As Long
Static Sigma() As Double

' i              = Control counter Vol() input vector
' j              = Counter simulation dates in Sigma() vector
' NumOfSimDates = Implementation specific variable setting the dimension of the Sigma()
'                 variable, hence controlling the amount of times the basic function loop's
' rows           = Count of rows in input Vol vector
' x              = Sets the evaluation date = SimDates(j) + IndexBaseNo
' Sigma          = Resulting variable

rows = Application.Count(Vol)
' Counts the number of Pub_SimDates initialized
NumOfSimDates = Application.Count(Pub_SimDates) - 1

' Redimentions Sigma() to fit the count of simulation dates
ReDim Sigma(NumOfSimDates)

' -- Start of code --
i = 1

' Major loop though all simulation dates
For j = 1 To NumOfSimDates

    ' Sets the evaluation date
    x = SimDates(j) + IndexBaseNo

    ' If x <= than the VolDates(i, 1)
    If x <= VolDates(i, 1) Then

        Sigma(j) = Vol(i, 1)

    ' If x > than the last VolDates(rows, 1)
    ElseIf x > VolDates(rows, 1) Then

        Sigma(j) = Vol(rows, 1)

    ' If x lies between VolDates(i, 1) < x < VolDates(rows, 1)
    ElseIf VolDates(i, 1) < x And x < VolDates(rows, 1) Then

    ' Ratchet property i.e. loop is minimized - input dates needs to be sorted ascending
    Do Until VolDates(i, 1) > x
        i = i + 1
        Loop

        Sigma(j) = Vol(i, 1)

    End If

' loops through all simulation dates
Next j

Pub_Sigma = Sigma

End Sub
```

C.4.6 LIntPdf

```
Sub LIntPdf()

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Linear interpolation of the discount curve                      '
' Calcs running linear interpolation from a range of x's and y's '
' For values x.min > X > x.max the output = y.min or y.max        '
' Own UDF re-implemented as sub                                   '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False
```

```
' -- Exogenous variables and values --
Dim Value As Variant, Index As Variant, x() As Integer, IndexBaseNo As Long

' Source market discountfactors from sheet "Outline MData"
Value = Sheets("Outline MData").range("B10:B129").Value
' Sets index values equal to ZCB curve dates
Index = Sheets("Outline MData").range("A10:A129").Value
' Sets the IndexBaseNo equal to the trade date of the Market Data
IndexBaseNo = Sheets("Outline MData").range("B5:B5").Value
' Source list of simulation dates from public variable "Pub_SimDates"
x = Pub_SimDates

' -- Endogenous variables --
Dim i As Integer, j As Integer, rows As Integer, x1 As Integer, x2 As Integer, y1 As Double,
y2 As Double, slope As Double, intercept As Double, xAxis() As Integer, yAxis() As Double,
NumOfSimDates As Integer
Static output() As Double

' rows = Total count of rows in the index data set
' x1, x2, y1, y2 = handles senario input variables
' Slope = Slope
' Intercept = Intercept
' xAxis = Vector of x-values
' yAxis = Vector of y-values
' NumOfSimDates = Control-variable for the amount of times the basic function is looped
' Output = Output vector

rows = Application.Count(Value)
' (-1) accounts for the first row of zero in Pub_SimDates
NumOfSimDates = Application.Count(Pub_SimDates) - 1

ReDim xAxis(rows), yAxis(rows), output(NumOfSimDates)

' -- Start of code --
' Initialization of input data
For i = 1 To rows
    xAxis(i) = Index(i, 1) - IndexBaseNo
    yAxis(i) = Value(i, 1)
Next i

' Major loop though all X(j)
For j = 1 To NumOfSimDates

    ' If X(j) < than the first x-observation
    If x(j) < xAxis(1) Then

        x1 = 0
        x2 = xAxis(1)
        ' Implementation specific - per definition P(0,0) = 1
        y1 = 1
        y2 = yAxis(1)

        slope = (y2 - y1) / (x2 - x1)
        intercept = y1 - slope * x1

        output(j) = slope * x(j) + intercept

    ' If X(j) > than the last x-observation
    ElseIf x(j) > xAxis(rows) Then

        output(j) = yAxis(rows)

    ' If X(j) is in between two x-observations - xAxis(1) < X(j) And X(j) < xAxis(rows)
    ElseIf xAxis(1) < x(j) And x(j) < xAxis(rows) Then

        i = 1

    Do Until xAxis(i) > x(j)
        i = i + 1
        Loop

        x1 = xAxis(i - 1)
        x2 = xAxis(i)
```

```
        y1 = yAxis(i - 1)
        y2 = yAxis(i)

        slope = (y2 - y1) / (x2 - x1)
        intercept = y1 - slope * x1

        output(j) = slope * x(j) + intercept

    Else
        ' If X(j) is = to an x-observation
        i = 1

        Do Until x(j) = xAxis(i)
        i = i + 1
        Loop
        output(j) = yAxis(i)

        'Next i

    End If

' major loop through all X(j)
Next j

Pub_df = output

Application.ScreenUpdating = True
End Sub
```

C.4.7 Hull-White-MC-Sub

```
Sub Hull_White_MC_Sub()
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Monte Carlo implementation of the Hull White Extended Vasicek model SDE  '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' -- Reset GUI values --
Sheets("MC_ExtVasi").range("B8") = "------- Analyzing ------- "

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False

' -- Exogenous variables and values --
Dim Kappa As Double, Sigma() As Double, StartT As Long, NumOfSteps As Integer,
NumOfSim As Long, Dim EvntD As range, CapFloor As String, K As Single, I7 As range,
dt() As Double, df() As Double

' Source calibrated HW Kappa from the sheet "Hull White Cal"
Kappa = Sheets("Hull White Cal").range("Y6").Value
' Sources the vector of Sigma(i)'s from the public variable Pub_Sigma
Sigma = Pub_Sigma
' Trade date (today)
StartT = range("E7").Value
' Number of Steps in each Simulation
NumOfSteps = range("E6").Value
' Number of simulations
NumOfSim = range("E5").Value
' Source the Vector of eventdates from Pub_CfRange
Set EvntD = range(Pub_CfRange)
' Sources the vector of dt(i)'s from the public variable Pub_dt
dt = Pub_dt
' Sources the vector of df(i)'s from the public variable Pub_df
df = Pub_df

' -- Endogenous variables --
Dim i As Long, j As Integer, NumEvntD As Integer, Timestp_begin As Variant,
Timestp_end As Variant, x() As Double, EXP_xdtSum() As Double
Dim MC As Double, m() As Double, mdtSum As Double, xdtSum() As Double, Timer As Variant
Static r() As Double, dfSim() As Double

Dim Test As Double, test2
```

```
' i                  = Counter number of simulations
' j                  = Counter number of steps
' NumEvntD           = Number of Eventdays - in this implementation defined as payment days
' Timestp_begin      = Timestamp CPU start of code execution
' Timestp_end        = Timestamp CPU end of code execution
' TimeDiff           = Time difference - measures total execution time
' x(i,j)             = Matrice of "shocks" to the instantaneous short rate
' xdtSum(i)          = Vector of Sum[x(i,j) * dt(j)] - i.e. for each sim(i) the variable
'                      cummulates [x(i,j) * dt(j)] for each step(j), j=1 to NumOfSteps
' EXP_xdtSum(i)      = Vector of EXP[-xdtSum(i)] - i.e. for each sim(i) the variable calcs
   the
'                      Exponential value of -1*Sum[x(i,j) * dt(j)]
' MC                 = Calcs the Monte Carlo [MC(j)] part. I.e. Sums EXP_xdtSum(i) over all
'                      sim(i) and averages *[1/NumOfSim]
' mdtSum             = Calcs Sum[m(1 to j-1)*dt(1 to j-1)]
' m()                = m(t) vector - J.Lund Part II (4) p. 2
' dfSim()            = Matrice of Stochastic discount factor
' r()                = Matrice of Instataneous short rate (Not used)
' Timer              = Calcs the time it takes to execute the sub

ReDim x(NumOfSim, 0 To NumOfSteps), xdtSum(NumOfSim), EXP_xdtSum(NumOfSim),
m(0 To NumOfSteps), dfSim(NumOfSim, 0 To NumOfSteps)

Timestp_begin = Now()
NumEvntD = EvntD.rows.Count

' -- Start Code --
' The following double for loop establishes the Xij matrice by 1.) calculating the first
' step(j) for each path/simulation(i) then followed by the next step. Thus the matrice is
' established in columns NOT in rows (which would be the intuitive apporach)

' Controls each step in a specific path
For j = 1 To NumOfSteps

    ' Controls each simulated path
    For i = 1 To NumOfSim
    Randomize
        ' Calcs the Xij matrice, starting with first step of EACH path, then next step for
        ' EACH path etc. [NB: Sigma need to be set = sigma(j)]
        x(i, j) = x(i, j - 1) + (-Kappa * x(i, j - 1) * dt(j) + Sigma(j) * RandNorm() * Sqr(
            dt(j)))

            ' Calcs Sum(i)[x(j)*dt(j)], i.e. cumulative per sim(i)
            xdtSum(i) = xdtSum(i) + x(i, j) * dt(j)

                ' Calcs EXP(Sum(i)[x(j)*dt(j)]), NON-cumulative, overwritten for each step(j
                    )
                EXP_xdtSum(i) = Exp(-xdtSum(i))
    Next i

                    ' Calcs the stochastic MC(j) part of each P(0,t(j)). The value MC(j)= MC
                    ' is overwritten for each step(j) as all subsequent calcs using MC(j)
                        are
                    ' performed before "next j"
                    MC = WorksheetFunction.Sum(EXP_xdtSum) * (1 / NumOfSim)

                        ' Calcs Sum[m(1 to j-1)*dt(1 to j-1)]
                        mdtSum = mdtSum + m(j - 1) * dt(j - 1)

                            ' Calcs formula JeLu "Simple Vasicek MC" draft / rewritten
                            m(j) = (-1 / dt(j)) * ((WorksheetFunction.Ln(df(j) / MC)) +
                                mdtSum)
Next j

'----- Free memory -----
ReDim EXP_xdtSum(1), xdtSum(NumOfSim)
mdtSum = 0

'----- Stochastic discountfactors / Instataneous short rate initialization ------
' Controls each step in a specific path
For j = 1 To NumOfSteps

    mdtSum = mdtSum + m(j) * dt(j)
```

```
    ' Controls each simulated path
    For i = 1 To NumOfSim

        xdtSum(i) = xdtSum(i) + x(i, j) * dt(j)

            ' Calcs the stochastic discount factors
            dfSim(i, j) = Exp(-mdtSum) * Exp(-xdtSum(i))

                ' Sets the df(i,0) = 1
                dfSim(i, 0) = 1

                    ' Calcs the r(i,j) matrice - J.Lund Part II (6) - Currently not used
                    'r(i, j) = (m(j) + x(i, j))

                         ' Check of r(i,j) - Calcs r(i,j) through discount factors
                         ' -> For check of code results only
                         'r_test(i, j) = -(Log(dfSim(i, j)) - Log(dfSim(i, j - 1))) / dt(j)
    Next i
Next j


'----- Free memory -----
ReDim xdtSum(1), x(1)

' Write dfSim to Public function
Pub_dfSim = dfSim
' Write r(i, j) to Public function
'Pub_r = r

' -- Output --
' Sets timestamp end
Timestp_end = Now()
' Set Timer - Convert to Excel time - fraction of a day (60 * 60 * 24)
Timer = (DateDiff("s", Timestp_begin, Timestp_end)) / 86400

' -- Write to GUI --
Application.ScreenUpdating = True
Sheets("MC_ExtVasi").range("B5") = Timer
' Model Initalized
Sheets("MC_ExtVasi").range("B8") = 1

' --------------------- TEST OUTPUT -- ON/OFF ------------------------
' Test output - Writes m(t) vector to sheet "Outline MDATA"
'For i = 1 To 120
'Sheets("Outline MData").range("S" & i + 9).Value = m(i)
'Next

' Test output - Writes Pub_Sigma(t) vector to sheet "Outline MDATA"
'For i = 1 To 120
'Sheets("Outline MData").range("V" & i + 9).Value = Pub_Sigma(i)
'Next

'Test output - Writes dt(t) vector to sheet "Test"
'For i = 1 To NumOfSteps
'Sheets("Test").range("A" & i + 1).Value = Pub_dt(i)
'Next

' Test output - Writes df_Sim(1,j) vector to sheet "Test"
'For i = 1 To NumOfSteps
'Sheets("Test").range("B" & i + 1).Value = dfSim(1, i)
'Next

' Test output - Writes r(1,j) vector to sheet "Test"
'For i = 1 To NumOfSteps
'Sheets("Test").range("D" & i + 1).Value = r(1, i)
'Next
' ---------------------------------------------------------------------

'Application.Calculation = xlCalculationAutomatic
End Sub
```

C.4.8 PayOffCapFloor

```
Sub PayOffCapFloor()

''''''''''''''''''''''''''''''''''''''''''''''
' Pricing module for European Caps (Floors) '
''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False
Application.Calculation = xlCalculationManual

' -- Exogenous variables and values --
Dim EvntD() As Long, StartT As Long, SimDates() As Integer, dfSim() As Double
Dim NumOfSteps As Integer, NumOfSim As Long, NumSimDates As Long
Dim toggleCap As Single, N As Double, Kc As Double, toggleFloor As Single, Kf As Double

EvntD = Pub_EvntD                       ' Source the Eventdays vector from Pub_EventD
StartT = range("E7").Value              ' From GUI - Trade date (today)
SimDates() = Pub_SimDates               ' Source Simulation dates Vector from Pub_SimDates
NumOfSteps = range("E6").Value          ' Number of Steps in each Simulation
NumOfSim = range("E5").Value            ' Number of simulations
toggleCap = range("E11").Value          ' From GUI - Determines Off/short/long/gearing.
toggleFloor = range("E18").Value        ' --"-- : E.g. 1=long factor one of principal amount
    '
                                        ' -1.5=short one and a holf times the principal
                                          amount
N = 1                                   ' Notional frac=Unity
                                        ' Possible implementation in seperate module
Kc = range("B12").Value                 ' From GUI - Strike Cap
Kf = range("B19").Value                 ' From GUI - Strike Floor

' -- Endogenous variables --
Dim Timestp_begin As Variant, Timestp_end As Variant, NumEvntD As Long, s As Integer,
Dim j As Integer, i As Long, Low As Integer, High As Integer, dayCfrac As Double,
Dim Fwdf As Double, Libor() As Double, PayOffCap As Double, PayOffFloor As Double,
Dim PvCapletVec() As Double, PvFloorletVec() As Double, PvCap As Double, SECap As Double,
Dim LB As Double, RB As Double, ConfB95Cap As String, PvFloor As Double, SEFloor As Double,
Dim ConfB95Floor As String, Timer As Variant

'Timestp_begin   = Timestamp CPU start of code execution
'Timestp_end     = Timestamp CPU end of code execution
'NumEvntD        = Total number of event days
's               = Counter event days / misc - used several times
'j               = Counter of steps in each simulation
'i               = Counter simulation number
'dayCFrac        = The Libor specfic dayCount fraction Act/360 base
'Fwdf()          = The forward discountfactor
'Libor()         = Matrice of Libor estimates. One per EventD in each simulation
'PayOffCap       = PayOff function
'PvCapletVec()   = Vector of PvCaplets. Each element contains the MC Pv for Caplet(s)
'PayOffFloor     = PayOff function
'PvFloorletVec() = Vector of PvFloorlets. Each element contains the MC Pv for Floorlet(s)
'PvCapSim()      = Vector - Each element contain the MC Pv Cap(i) for each simulation
'PvFloorSim()    = Vector - Each element contain the MC Pv Floor(i) for each simulation
'PvCap           = Sum over all elements in PvCapSim
'SECap           = Standard Error of the mean
'LB              = Left bound 95% confidence level of mean
'RB              = Right bound 95% confidence level of mean
'ConfB95Cap      = 95% confidence band as string
'PvFloor         = Sum over all elements in PvFloorSim
'SEFloor         = Standard Error of the mean
'ConfB95Floor    = 95% confidence band as string
'Timer           = Calcs the time it takes to execute the sub

' -- Start Code --
Timestp_begin = Now()
NumEvntD = UBound(EvntD)

ReDim Libor(NumOfSim, 0 To NumEvntD - 1), PvCapletVec(1 To NumEvntD - 1)
ReDim PvFloorletVec(1 To NumEvntD - 1), PvCapSim(NumOfSim), PvFloorSim(NumOfSim)

' Double for loop determines corresponding subscripts to time t,
' T matching each Libor rate for each simulation path
```

```
For i = 1 To NumOfSim
    j = 1
    For s = 1 To (NumEvntD - 1)
        ' Loop search for subscript equivalent to t = low
        Do Until EvntD(s) = SimDates(j)
        j = j + 1
        Loop
        Low = j

        ' Loop search for subscript equivalent to t = high. Ratchet feature to minimize
            calcs
        ' as loop starts from previous s+1
        Do Until EvntD(s + 1) = SimDates(j)
        j = j + 1
        Loop
        High = j

' -- Calcs matrice of Libor rates --

        ' Daycount fraction based on act/360
        dayCfrac = (SimDates(High) - SimDates(Low)) / 360

        ' Calcs the forward discount fractor
        Fwdf = Pub_dfSim(i, High) / Pub_dfSim(i, Low)

        ' Calcs the simple compounded Libor Rate - Andrew Lesniewski/The Forward Curve p. 5
        Libor(i, s) = (1 / Fwdf - 1) * (1 / dayCfrac)

        ' Check calc
        'Libor(i, s) = (Ln(df(High)) - Ln(df(Low))) / (SimDates(High) - SimDates(Low)) / 360

' -- Calcs Digital PvPayoff(s) --

       'Caplet
        PayOffCap = WorksheetFunction.Max(Libor(i, s) - Kc, 0)              ' ITM/OTM
        PvCapletVec(s) = N * Pub_dfSim(i, High) * dayCfrac * PayOffCap      ' PV of payoff

       'Floorlet
        PayOffFloor = WorksheetFunction.Max(Kf - Libor(i, s), 0)            ' ITM/OTM
        PvFloorletVec(s) = N * Pub_dfSim(i, High) * dayCfrac * PayOffFloor  ' PV of payoff

    Next s
    PvCapSim(i) = Application.WorksheetFunction.Sum(PvCapletVec)
    PvFloorSim(i) = Application.WorksheetFunction.Sum(PvFloorletVec)
Next i

' -- Output --
' Sum over all elements in PvCapSim
PvCap = Application.WorksheetFunction.Sum(PvCapSim) * (1 / NumOfSim) * toggleCap
' Standard Error of the mean
SECap = Application.WorksheetFunction.StDev(PvCapSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvCap - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
 RB = Round(PvCap + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Cap = LB & " : " & RB

' Sum over all elements in PvFloorSim
PvFloor = Application.WorksheetFunction.Sum(PvFloorSim) * (1 / NumOfSim) * toggleFloor
' Standard Error of the mean
SEFloor = Application.WorksheetFunction.StDev(PvFloorSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvFloor - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
 RB = Round(PvFloor + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Floor = LB & " : " & RB

' Sets timestamp end
Timestp_end = Now()
' Converts to Excel time - fraction of a day (60 * 60 * 24)
Timer = (DateDiff("s", Timestp_begin, Timestp_end)) / 86400

' -- Write to GUI --
```

```
Application.ScreenUpdating = True
range("B14").Value = PvCap
range("B15").Value = SECap
range("B16").Value = ConfB95Cap
range("B21").Value = PvFloor
range("B22").Value = SEFloor
range("B23").Value = ConfB95Floor
range("B25").Value = Timer
'Application.Calculation = xlCalculationAutomatic
End Sub
```

C.4.9 PayOffDigi

```
Sub PayOffDigi()

''''''''''''''''''''''''''''''''''''''''''''''''
' Pricing module for Digial Caps and Floors '
''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False
Application.Calculation = xlCalculationManual

' -- Exogenous variables and values --
Dim EvntD() As Long, StartT As Long, SimDates() As Integer, dfSim() As Double
Dim NumOfSteps As Integer, NumOfSim As Long, NumSimDates As Long, toggleCap As Single
Dim N As Double, Kc As Double, toggleFloor As Single, Kf As Double, PayOffRateCap As Double
Dim PayOffRateFloor As Double

EvntD = Pub_EvntD                        ' Source the Eventdays vector from Pub_EventD
StartT = range("E7").Value               ' From GUI - Trade date (today)
SimDates() = Pub_SimDates                ' Source Simulation dates Vector from Pub_SimDates
NumOfSteps = range("E6").Value           ' Number of Steps in each Simulation
NumOfSim = range("E5").Value             ' Number of simulations
toggleCap = range("E11").Value           ' From GUI - Determines Off/short/long/gearing.
toggleFloor = range("E18").Value         ' --"--: E.g. 1=long factor one of principal amount,
               ' -1.5=short one and a half x the principal amount
N = 1                                    ' Notional frac = Unity
Kc = range("E12").Value                  ' From GUI - Strike Cap
Kf = range("E19").Value                  ' From GUI - Strike Floor
PayOffRateCap = range("E13").Value       ' From GUI PayOff Rate
PayOffRateFloor = range("E20").Value     ' From GUI PayOff Rate

' -- Endogenous variables --
Dim Timestp_begin As Variant, Timestp_end As Variant, NumEvntD As Long, s As Integer
Dim j As Integer, i As Long, Low As Integer, High As Integer, dayCfrac As Double
Dim Fwdf As Double, Libor() As Double, Bin As Boolean, PayOffCap As Double
Dim PayOffFloor As Double, PvCapletVec() As Double, PvFloorletVec() As Double
Dim PvCap As Double, SECap As Double, LB As Double, RB As Double, ConfB95Cap As String
Dim PvFloor As Double, SEFloor As Double, ConfB95Floor As String, Timer As Variant

'Timestp_begin   = Timestamp CPU start of code execution
'Timestp_end     = Timestamp CPU end of code execution
'NumEvntD        = Total number of event days
's               = Counter event days / misc - used several times
'j               = Counter of steps in each simulation
'i               = Counter simulation number
'dayCFrac        = The Libor specfic dayCount fraction Act/360 base
'Fwdf()          = The forward discountfactor
'Libor()         = Matrice of Libor estimates. One per EventD in each simulation
'Bin             = Support ITM/OTM boolean variable
'PayOffCap       = PayOff function
'PvCapletVec()   = Vector of PvCaplets. Each element contains the MC Pv for Caplet(s)
'PayOffFloor     = PayOff function
'PvFloorletVec() = Vector of PvFloorlets. Each element contains the MC Pv for Floorlet(s)
'PvCapSim()      = Vector - Each element contain the MC Pv Cap(i) for each simulation
'PvFloorSim()    = Vector - Each element contain the MC Pv Floor(i) for each simulation
'PvCap           = Sum over all elements in PvCapSim
'SECap           = Standard Error of the mean
'LB              = Left bound 95% confidence level of mean
'RB              = Right bound 95% confidence level of mean
'ConfB95Cap      = 95% confidence band as string
'PvFloor         = Sum over all elements in PvFloorSim
'SEFloor         = Standard Error of the mean
```

```
'ConfB95Floor     = 95% confidence band as string
'Timer            = Calcs the time it takes to execute the sub

' -- Start Code --
Timestp_begin = Now()
NumEvntD = UBound(EvntD)
ReDim Libor(NumOfSim, 0 To NumEvntD - 1), PvCapletVec(1 To NumEvntD - 1)
ReDim PvFloorletVec(1 To NumEvntD - 1), PvCapSim(NumOfSim), PvFloorSim(NumOfSim)

' Double for loop determines corresponding subscripts to time t,
' T matching each Libor rate for each simulation path
For i = 1 To NumOfSim
    j = 1
    For s = 1 To (NumEvntD - 1)
        ' Loop search for subscript equivalent to t = low
        Do Until EvntD(s) = SimDates(j)
        j = j + 1
        Loop
        Low = j

        ' Loop search for subscript equivalent to t = high. Ratchet feature to minimize
            calcs
        ' as loop starts from previous s+1
        Do Until EvntD(s + 1) = SimDates(j)
        j = j + 1
        Loop
        High = j

' -- Calcs matrice of Libor rates --

        ' Daycount fraction based on act/360
        dayCfrac = (SimDates(High) - SimDates(Low)) / 360

        ' Calcs the forward discount fractor
        Fwdf = Pub_dfSim(i, High) / Pub_dfSim(i, Low)

        ' Calcs the simple compounded Libor Rate - Andrew Lesniewski/The Forward Curve p. 5
        Libor(i, s) = (1 / Fwdf - 1) * (1 / dayCfrac)

        ' Check of calcs
        'Libor(i, s) = (Ln(df(High)) - Ln(df(Low))) / (SimDates(High) - SimDates(Low)) / 360

' -- Calcs Digital PvPayoff(s) --

       'Caplet
        Bin = WorksheetFunction.Max(Libor(i, s) - Kc, 0)                    ' ITM/OTM
            boolean
        PayOffCap = Abs(Bin) * PayOffRateCap                                ' Conversion to
            digital
        PvCapletVec(s) = N * Pub_dfSim(i, High) * dayCfrac * PayOffCap      ' PV of payoff

       'Floorlet
        Bin = WorksheetFunction.Max(Kf - Libor(i, s), 0)                    ' ITM/OTM
            boolean
        PayOffFloor = Abs(Bin) * PayOffRateFloor                            ' Conversion to
            digital Payoff
        PvFloorletVec(s) = N * Pub_dfSim(i, High) * dayCfrac * PayOffFloor  ' PV of payoff

    Next s
    PvCapSim(i) = Application.WorksheetFunction.Sum(PvCapletVec)
    PvFloorSim(i) = Application.WorksheetFunction.Sum(PvFloorletVec)
Next i
'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' -- Output --
' Sum over all elements in PvCapSim
PvCap = Application.WorksheetFunction.Sum(PvCapSim) * (1 / NumOfSim) * toggleCap
' Standard Error of the mean
SECap = Application.WorksheetFunction.StDev(PvCapSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvCap - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
```

```
 RB = Round(PvCap + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Cap = LB & " : " & RB

' Sum over all elements in PvFloorSim
PvFloor = Application.WorksheetFunction.Sum(PvFloorSim) * (1 / NumOfSim) * toggleFloor
' Standard Error of the mean
SEFloor = Application.WorksheetFunction.StDev(PvFloorSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvFloor - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
 RB = Round(PvFloor + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Floor = LB & " : " & RB

' Sets timestamp end
Timestp_end = Now()
' Converts to Excel time - fraction of a day (60 * 60 * 24)
Timer = (DateDiff("s", Timestp_begin, Timestp_end)) / 86400

' -- Write to GUI --
Application.ScreenUpdating = True
range("E14").Value = PvCap
range("E15").Value = SECap
range("E16").Value = ConfB95Cap

range("E21").Value = PvFloor
range("E22").Value = SEFloor
range("E23").Value = ConfB95Floor

range("E25").Value = Timer

'Application.Calculation = xlCalculationAutomatic
End Sub
```

C.4.10 PayOffRangeAccrual

```
Sub PayOffRangeAccrual()

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Pricing module for Exotic Libor based Range Accrual Caps (Floors) '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Application.ScreenUpdating = False
Application.Calculation = xlCalculationManual

' -- Exogenous variables and values --
Dim EvntD() As Long, AccrualD() As Integer, dAccrualD() As Integer, StartT As Long
Dim SimDates() As Integer, dfSim() As Double, NumOfSteps As Integer, NumOfSim As Long
Dim NumSimDates As Long, toggleCap As Single, N As Double, Kc As Double
Dim toggleFloor As Single, Kf As Double, PayOffStringCap As String
Dim PayOffStringFloor As String

EvntD = Pub_EvntD                       ' Source the Eventdays vector from Pub_EventD
AccrualD = Pub_AccrualD                 ' Accrual dates
dAccrualD = Pub_dAccrualD               ' delta t(j) - i.e. d2-d1
StartT = range("E7").Value              ' From GUI - Trade date (today)
SimDates() = Pub_SimDates               ' Source Simulation dates Vector from Pub_SimDates
NumOfSteps = range("E6").Value          ' Number of Steps in each Simulation
NumOfSim = range("E5").Value            ' Number of simulations
toggleCap = range("H11").Value          ' From GUI - Determines Off/short/long/gearing.
toggleFloor = range("H18").Value        ' --"--: E.g. 1=long factor one of principal amount,
                                        ' -1.5=short 1 and a half times the principal amount
N = 1                                   ' Notional frac = Unity
Kc = range("H12").Value                 ' From GUI - Strike Cap
Kf = range("H19").Value                 ' From GUI - Strike Floor
PayOffStringCap = range("H13").Value    ' From GUI - Source PayOffString Cap
PayOffStringFloor = range("H20").Value  ' From GUI - Source PayOffString Floor

' -- Endogenous variables --
Dim Timestp_begin As Variant, Timestp_end As Variant, NumEvntD As Long, Tenor As Byte
Dim i As Long, h As Long, s As Integer, j As Integer, Low As Integer, y As Long,
Dim AccrualPair As Integer, High As Integer, EndPeriodPmtD As Integer, BgnPeriod As Integer
Dim dayCfrac As Double, Fwdf As Double, Libor() As Double, BinC As Boolean, BinF As Boolean
Dim ITMdaysC As Integer, ITMdaysF As Integer, DaysP As Integer, PayOffCap As String
```

```
Dim PayOffFloor As String, PvCapletVec() As Double, PvFloorletVec() As Double
Dim PvCap As Double, SECap As Double, LB As Double, RB As Double, ConfB95Cap As String
Dim PvFloor As Double, SEFloor As Double, ConfB95Floor As String, Timer As Variant

'Timestp_begin    = Timestamp CPU start of code execution
'Timestp_end      = Timestamp CPU end of code execution
'NumEvntD         = Total number of event days
'Tenor            = Tenor in month's - prespecified throughout the Sub to Libor 3M, however
'                   readily changable by chg the value of this variable once
'i                = Counter simulation number
'h                = Counter of EventDates (payments dates)
's                = Counter event days / misc - used several times
'j                = Counter of steps (in each simulation)
'Low              = Temp variable timestamp of AccrualD(s) = SimDates(Low)
'y                = Temp support variable in relation to AccrualPair
'AccrualPair      = Sets High equal to low + 3 Month (via dateadd)
'High             = Temp variable timestamp of AccrualD(s) = SimDates(Low)
'BgnPeriod        = Temp variable - Begin period time stamp
'EndPeriodPmtD    = Temp variable - End period/payment date time stamp
'dayCFrac         = The Libor specfic dayCount fraction Act/360 base
'Fwdf()           = The forward discountfactor
'Libor()          = Matrice of Libor estimates. One per EventD in each simulation
'BinC             = Support ITM/OTM boolean variable, Cap
'BinF             = Support ITM/OTM boolean variable, Floor
'ITMdaysC         = Cumulative amount of ITM days, Cap
'ITMdaysF         = Cumulative amount of ITM days, Floor
'DaysP            = Days in period. I.e. EndPeriodPmtD - BgnPeriod
'PayOffCap        = PayOff function
'PvCapletVec()    = Vector of PvCaplets. Each element contains the MC Pv for Caplet(s)
'PayOffFloor      = PayOff function
'PvFloorletVec() = Vector of PvFloorlets. Each element contains the MC Pv for Floorlet(s)
'PvCapSim()       = Vector - Each element contain the MC Pv Cap(i) for each simulation
'PvFloorSim()     = Vector - Each element contain the MC Pv Floor(i) for each simulation
'PvCap            = Sum over all elements in PvCapSim
'SECap            = Standard Error of the mean
'LB               = Left bound 95% confidence level of mean
'RB               = Right bound 95% confidence level of mean
'ConfB95Cap       = 95% confidence band as string
'PvFloor          = Sum over all elements in PvFloorSim
'SEFloor          = Standard Error of the mean
'ConfB95Floor     = 95% confidence band as string
'Timer            = Calcs the time it takes to execute the sub

Dim LMarker_s As Integer

' -- Start Code --
Timestp_begin = Now()
NumEvntD = UBound(EvntD)
ReDim Libor(NumOfSim, 0 To UBound(AccrualD)), PvCapletVec(1 To UBound(AccrualD))
ReDim PvFloorletVec(1 To UBound(AccrualD)), PvCapSim(NumOfSim), PvFloorSim(NumOfSim)

' Tenor: Pre-set to 3M Libor
Tenor = 3

For i = 1 To NumOfSim

    ' Reset Event date counter
    h = 1
    ' Reset Low/High counter
    Low = 1
    ' Reset Low/High counter
    High = 1
    ' Reset ITMdaysC
    ITMdaysC = 0
    ' Reset ITMdaysF
    ITMdaysF = 0

    ' Frontcheck (Low)
    For s = 1 To (UBound(AccrualD))
        ' Loop search for subscript equivalent to t = low
        Do Until AccrualD(s) = SimDates(Low)
        ' Roll Simulation Date
        Low = Low + 1
```

```
        Loop

      ' Backcheck (High) - if on AccrualD/non payment date
      If AccrualD(s) <> EvntD(h) Then
        y = AccrualD(s) + StartT
        ' Calcs paired 3M (tenor) point
        AccrualPair = DateAdd("m", Tenor, y) - StartT

        ' Loop search for subscript equivalent to t = high. Ratchet feature to minimize
            calcs
        ' as loop starts from previous j+1
        Do Until AccrualPair = SimDates(High)
        ' Roll Simulation Date
        High = High + 1
        Loop

      ' Backcheck (High) - if on an AccrualD AND payment date (EvntD)
      Else
        ' The patch Abs(s<>UBound(AccrualD)*High) is implementet to solve bug when
        ' s=UBound(AccrualD). It Resets High to 0 when s=max
        High = Abs(s <> UBound(AccrualD)) * High
        ' Loop searches for subscript in SimDates so that SimDates(High) = EvntD(h+1). The
        ' patch (s=UBound(AccrualD)) is implementet to solve bug when s=UBound(AccrualD)
        ' (adds -1 if true) - corresponds to patch under dayCFrac - L. 453
        Do Until EvntD(h + 1 + (s = UBound(AccrualD))) = SimDates(High)
        ' Roll Simulation Date
        High = High + 1
        Loop
        ' Roll Eventdate (payment date)
        h = h + 1

        ' Sets the relevant interest rate period
        If s = 1 Then
        LMarker_s = s
        ' Here only set for first step in each simulation (s=1), as else the temp markers
        ' would be overwritten by the following period befor use
        BgnPeriod = Low
        ' Interest rate periods when s > 1 are set after the PvPayoff calculation
        EndPeriodPmtD = High
        End If

      End If

' -- Calcs matrice of Libor rates --
        ' Daycount fraction based on act/360. The patch (s=UBound(AccrualD)) is implementet
        ' to solve bug when s=UBound(AccrualD)
        dayCfrac = ((SimDates(High) - SimDates(Low)) + (s = UBound(AccrualD))) / 360

        ' Calcs the forward discount fractor
        Fwdf = Pub_dfSim(i, High) / Pub_dfSim(i, Low)

        ' Calcs the simple compounded Libor Rate - Andrew Lesniewski/The Forward Curve p. 5
        Libor(i, s) = (1 / Fwdf - 1) * (1 / dayCfrac)

        ' Check of Libor calcs (only for debug)
        'Libor(i, s) = (Ln(df(High)) - Ln(df(Low))) / (SimDates(High) - SimDates(Low)) / 360

' -- Calcs ITM/OTM digital --
        BinC = WorksheetFunction.Max(Kc - Libor(i, s), 0)  ' ITM/OTM boolean Cap
        BinF = WorksheetFunction.Max(Kf - Libor(i, s), 0)  ' ITM/OTM boolean Floor

     ' If NOT on a Payment date
     If EndPeriodPmtD <> Low Then

' -- ITM/OTM day Counter --
        ' Cap: Cumulative ITM days
        ITMdaysC = ITMdaysC + Abs(BinC) * dAccrualD(s)
        ' Floor: Cumulative ITM days
        ITMdaysF = ITMdaysF + Abs(BinF) * dAccrualD(s)

      ' IF on a Payment date
      Else
```

```
' -- Calcs Fixing + PvPayoff(s) --
        DaysP = (SimDates(EndPeriodPmtD) - SimDates(BgnPeriod))
        ' Daycount fraction based on act/360
        dayCfrac = DaysP / 360

       'PvCaplets
        ' Source PayOffString Cap
        PayOffCap = PayOffStringCap
        ' Replace "Libor" for UDF Libor
        PayOffCap = Replace(PayOffCap, "Libor", Libor(i, LMarker_s))  'BgnPeriod
        ' Replace "," for "." to have Evalute() to function
        PayOffCap = Replace(PayOffCap, ",", ".")
        ' Replace ";" for "." to delimit when evaluating min/max functions
        PayOffCap = Replace(PayOffCap, ";", ",")
        ' Evaluate PayOffString
        PayOffCap = Application.Evaluate(PayOffCap)

' PV of Caplet payoff
PvCapletVec(s) = N * Pub_dfSim(i, EndPeriodPmtD) * dayCfrac * (ITMdaysC / DaysP) * PayOffCap

       'PvFloorlets
        ' Source PayOffString Floor
        PayOffFloor = PayOffStringFloor
        ' Replace "Libor" for UDF Libor
        PayOffFloor = Replace(PayOffFloor, "Libor", Libor(i, LMarker_s)) 'BgnPeriod
        ' Replace "," for "." to have Evalute() to functioning
        PayOffFloor = Replace(PayOffFloor, ",", ".")
        ' Replace ";" for "." to delimit when evaluating min/max functions
        PayOffFloor = Replace(PayOffFloor, ";", ",")
        ' Evaluate PayOffString
        PayOffFloor = Application.Evaluate(PayOffFloor)

' PV of Floorlet payoff
PvFloorletVec(s) = N * Pub_dfSim(i, EndPeriodPmtD) * dayCfrac * (ITMdaysF / DaysP) *
    PayOffFloor

        ITMdaysC = Abs(BinC) * dAccrualD(s)      ' [Reset] Cap: Cumulative ITM days
        ITMdaysF = Abs(BinF) * dAccrualD(s)      ' [Reset] Floor: Cumulative ITM days

        LMarker_s = s
        BgnPeriod = Low                          ' Sets the relevant interest rate period
            markers for s>1
        EndPeriodPmtD = High                     ' --"--
      End If

    Next s
   'PvCap/Floor
    PvCapSim(i) = Application.WorksheetFunction.Sum(PvCapletVec)
    PvFloorSim(i) = Application.WorksheetFunction.Sum(PvFloorletVec)
Next i

' -- Output --
' Sum over all elements in PvCapSim
PvCap = Application.WorksheetFunction.Sum(PvCapSim) * (1 / NumOfSim) * toggleCap
' Standard Error of the mean
SECap = Application.WorksheetFunction.StDev(PvCapSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvCap - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
 RB = Round(PvCap + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Cap = LB & " : " & RB

' Sum over all elements in PvFloorSim
PvFloor = Application.WorksheetFunction.Sum(PvFloorSim) * (1 / NumOfSim) * toggleFloor
' Standard Error of the mean
SEFloor = Application.WorksheetFunction.StDev(PvFloorSim) / Sqr(NumOfSim)
 ' Left bound 95% confidence level of mean
 LB = Round(PvFloor - Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
 ' Right bound 95% confidence level of mean
 RB = Round(PvFloor + Application.WorksheetFunction.TInv(0.05, NumOfSim) * SECap, 6) * 100
ConfB95Floor = LB & " : " & RB

' Sets timestamp end
```

```
Timestp_end = Now()
' Converts to Excel time - fraction of a day (60 * 60 * 24)
Timer = (DateDiff("s", Timestp_begin, Timestp_end)) / 86400

' -- Write to GUI --
Application.ScreenUpdating = True
range("H14").Value = PvCap
range("H15").Value = SECap
range("H16").Value = ConfB95Cap

range("H21").Value = PvFloor
range("H22").Value = SEFloor
range("H23").Value = ConfB95Floor

range("H25").Value = Timer

'Application.Calculation = xlCalculationAutomatic
End Sub
```

C.4.11 ImplVol

```
Function ImplVol(Target As Double, precision As Double, range As range,
Optional Strike As Double)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Very simple binary search functionality, used for solving problems of monotonicity '
' Here used to calcs implied volatility in the BS/Norm models                        '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

    Application.ScreenUpdating = False

    Dim High As Double, Low As Double

    High = 1
    Low = 0

    Do While High - Low > precision

    If CapBS(range, Strike, (High + Low) / 2) > Target Then

    High = (High + Low) / 2

    Else

    Low = (High + Low) / 2

    End If

    Loop

    ImplVol = (High + Low) / 2

    Application.ScreenUpdating = True

End Function
```

C.4.12 PriceInstr-Routine

```
Sub PriceInstr_Routine()

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Executes sequence of modules related to Pricing of all product '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Call PayOffCapFloor
Call PayOffDigi
Call PayOffRangeAccrual

End Sub
```

C.4.13 UpdateCF-Routine

```
Sub UpdateCF_Routine()

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Executes sequence of modules related to update of the Cash Flow Schdule '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Call EventSchedule
Call Simulation_Dates
Call LIntPdf
Call VolInitialize
Call AccrualEventDates

End Sub
```

C.4.14 ClrEventSchedule

```
Sub ClrEventSchedule()

''''''''''''''''''''''''''''
' Clear Cash Flow Schedule '
''''''''''''''''''''''''''''

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False

Dim Date1 As String, Date2 As String, Date3 As String, i As Long, daysC As String
Dim FormatArray As String

If range("J5") <> range("I7") <> range("H11") <> "" Then

Date1 = "J5"
i = 5

Do
Date2 = "J" & i
daysC = "K" & i
range(Date2) = ""
range(daysC) = ""

Date3 = "J" & i + 1
i = i + 1
Loop Until range(Date3).Text = ""

FormatArray = Date1 & ":" & daysC

 range("J1").Select
 Selection.Copy
 range(FormatArray).Select
 Selection.PasteSpecial Paste:=xlPasteFormats

 range("J1").Select
 Selection.Copy
 range("K4").Select
 Selection.PasteSpecial Paste:=xlPasteFormats

range("I7,K4,E6") = ""

End If
Application.CutCopyMode = False

' -- Write to GUI --
Application.ScreenUpdating = True

range("B5").Value = 0                ' Resets timer
range("B6").Value = "-- n/a --"      ' Resets total number of calculations
range("B7").Value = 0                ' Resets CF schedule
range("B8").Value = 0                ' Resets model initialized

'Application.Calculation = xlCalculationAutomatic

End Sub
```

C.5 Third Party Code

C.5.1 QuickSort1

```
Public Sub QuickSort1(ByRef pvarArray As Variant, Optional ByVal plngLeft As Long,
           Optional ByVal plngRight As Long)

' Algorithem from http://www.vbforums.com/showpost.php?p=2909259&postcount=13
' Omit plngLeft & plngRight; they are used internally during recursion

    Dim lngFirst As Long
    Dim lngLast As Long
    Dim varMid As Variant
    Dim varSwap As Variant

    If plngRight = 0 Then
   plngLeft = LBound(pvarArray)
   plngRight = UBound(pvarArray)
    End If
    lngFirst = plngLeft
    lngLast = plngRight
    varMid = pvarArray((plngLeft + plngRight) \ 2)
    Do
   Do While pvarArray(lngFirst) < varMid And lngFirst < plngRight
       lngFirst = lngFirst + 1
   Loop
   Do While varMid < pvarArray(lngLast) And lngLast > plngLeft
       lngLast = lngLast - 1
   Loop
   If lngFirst <= lngLast Then
       varSwap = pvarArray(lngFirst)
       pvarArray(lngFirst) = pvarArray(lngLast)
       pvarArray(lngLast) = varSwap
       lngFirst = lngFirst + 1
       lngLast = lngLast - 1
   End If
    Loop Until lngFirst > lngLast
    If plngLeft < lngLast Then QuickSort1 pvarArray, plngLeft, lngLast
    If lngFirst < plngRight Then QuickSort1 pvarArray, lngFirst, plngRight
End Sub
```

C.5.2 RandNorm

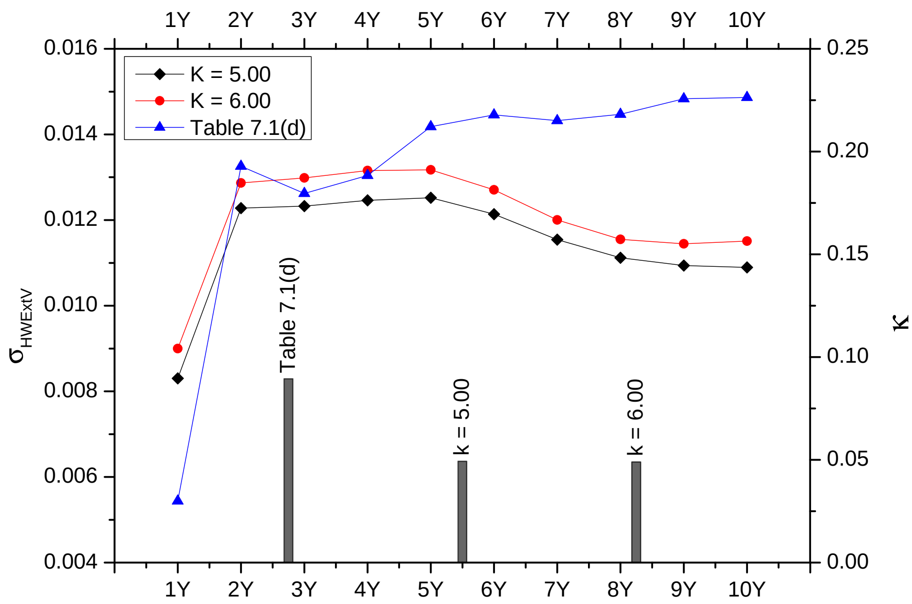
```
Function RandNorm(Optional Mean As Single = 0!, Optional Dev As Single = 1!, Optional
         fCorrel As Single = 0!, Optional bVolatile As Boolean = False) As Single

'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' Box-Muller Polar Method
                                                                                  '
' Donald Knuth, The Art of Computer Programming,
                                                           '
' Vol 2, Seminumerical Algorithms, p. 117
                                                                  '
'
    '
' Returns a pair of random deviates (Singles) with the specified (Function adjusted to only
    '
' output one value, Single instead of single()/ABH) mean, deviation, and correlation.
          '
' Orders of magnitude faster than =NORMINV(RAND(), Mean, Dev)
                                              '
'
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''


    Dim af(1 To 2) As Single
    Dim x        As Single
    Dim y        As Single
    Dim w        As Single
```

```
    If bVolatile Then Application.Volatile

    Do
        x = 2! * Rnd - 1!
        y = 2! * Rnd - 1!
        w = x ^ 2 + y ^ 2
    Loop Until w < 1!

    w = Sqr((-2! * CSng(Log(w))) / w)
    af(1) = Dev * x * w + Mean
    af(2) = Dev * y * w + Mean

    If fCorrel <> 0! Then af(2) = fCorrel * af(1) + Sqr(1! - fCorrel * fCorrel) * af(2)
    RandNorm = af(1) 'Function amended from af to af(1) /ABH
End Function
```

Appendix D

Re-Calibrated Parametrization (Barrier Swap)

Figure D.1: Barrier Swap - Cross-Calibration Comparison of the Parameter Vector of σ_i.



Appendix E

Re-Calibrated Parametrization (Range Accrual Swap)

Figure E.1: Range Accrual Swap - Cross-Calibration Comparison of the Parameter Vector of σ_i.