LOCAL IMPLIED VOLATILITY:
A MARKET MODEL APPROACH

BY

THOM DE JONG

A

THESIS

IN PARTIAL FULFILLMENT FOR THE DEGREE OF

MASTER OF SCIENCE

SUBMITTED TO THE DEPARTMENT OF MATHEMATICS OF

UTRECHT UNIVERSITY

FEBRUARY 2013

SUPERVISED BY

MARTIJN PISTORIUS
UNIVERSITY OF AMSTERDAM

TOBIAS MÜLLER
UTRECHT UNIVERSITY

Abstract

Option valuation models specifying the dynamics of the stock price directly always face the problem of calibration to the volatility surface, which shows to be a troublesome procedure. Market models take the option price surface as an input of the model and are hence perfectly calibrated to the market. Market models try to specify the dynamics of the option prices, rather than the stock price itself. Due to no-arbitrage constraints, specifying the dynamics of the option prices becomes unworkable quickly. Therefore, the idea is to find a one-to-one parameterization of the option price surface. We then specify dynamics for this parameterization to equivalently have the evolution of the option prices. The dynamics of the parameterization are subject to no-arbitrage conditions, which are either complicated or restricted to a single strike or maturity. Johannes Wissel recently came with a model that works for option prices on a fixed grid of strikes and maturities, without invoking complicated restraints. Based on this option valuation model, we analyze the way such a model works in practice. We find ways to calibrate the dynamics of the parameterization and apply the model to the valuation of exotic derivatives that potentially benefit from this approach, such as forward start options, forward start variance swaps and structured products.

Keywords: ◇ Arbitrage-free call price smoothing ◇ Exotic option valuation ◇ Stochastic differential equations ◇ Local implied volatility ◇ Market model ◇ Monte Carlo

Acknowledgements

This research would not have been possible without the valuable guidance of several people along my path.

First of all, the project's supervisor dr. Martijn Pistorius (University of Amsterdam) has been a great help for the moments I got stuck. The time he took to catch up with me and the discussions that reached topics beyond mathematics are highly appreciated. Also, I encourage his new course in stochastic volatility, which helped me abundantly to get a wider view of option modeling.

Second, I am grateful to dr. Peter Spreij (University of Amsterdam) for teaching measure theory, the building block of the Master program Stochastics and Financial Mathematics. He also gave me the first steps towards stochastic volatility by recommending the right books to give me a solid background knowledge of the area I was entering.

Third, I would like to thank dr. Tobias Müller for his position as my supervisor from Utrecht University and dr. Karma Dajani (Utrecht University) for her position as additional reader.

Special thanks go to my fellow student and ex-neighbor Stefan Radnev. We have gone through most courses of our Master together. His way of thinking highly complemented mine, leaving us with many insights we could not have obtained elsewhere. Not only have we helped each other out tremendously through courses and the final project, but also in our social life. In my opinion he boosted our overall performance.

Finally, my team consisting of Cédric van der Haert and Tom Groothaert at Credit Suisse are thanked for the opportunity they gave me to get valuable practical experience and knowledge on the trading floor. With this practical background a lot more insights have been obtained, steepening the learning curve I've gone through during this project.

Contents

1 Introduction

After Bachelier introduced the option in his thesis "Théorie de la Spéculation" in 1900, it took 73 years for the valuation of options to become standardized by Black & Scholes [Bla73]. In their framework the stock price moves according to the stochastic differential equation (SDE)

$$\frac{dS_t}{S_t} = \mu dt + \sigma dW_t, \quad S_0 = s, \tag{1.1}$$

where S_t denotes the stock price in consideration at time t, $\mu \in \mathbb{R}$ a constant drift term, $\sigma > 0$ the stock's volatility and W_t a $\mathbb{P}$-Brownian motion. Black & Scholes' main contribution was in fact not model (1.1) itself, which was due to Samuelson [Sam65], but the fundamental idea that any contingent claim $H(S_T)$ for a sufficiently well-behaving function H can be perfectly replicated by continuous trading in the stock and bonds. This means that there is no risk in selling such a contingent claim H. Let Φ denote the cumulative distribution function of a standard normal distributed random variable, r be the risk-free rate and d the anticipated dividend yield, then the Black & Scholes (BS) price of a European call option is given by the famous formula

$$C_t^{BS}(S_t, K, T, r, d, \sigma) := S_t e^{-d\tau}\Phi(d^+) - Ke^{-r\tau}\Phi(d^-), \tag{1.2}$$

where

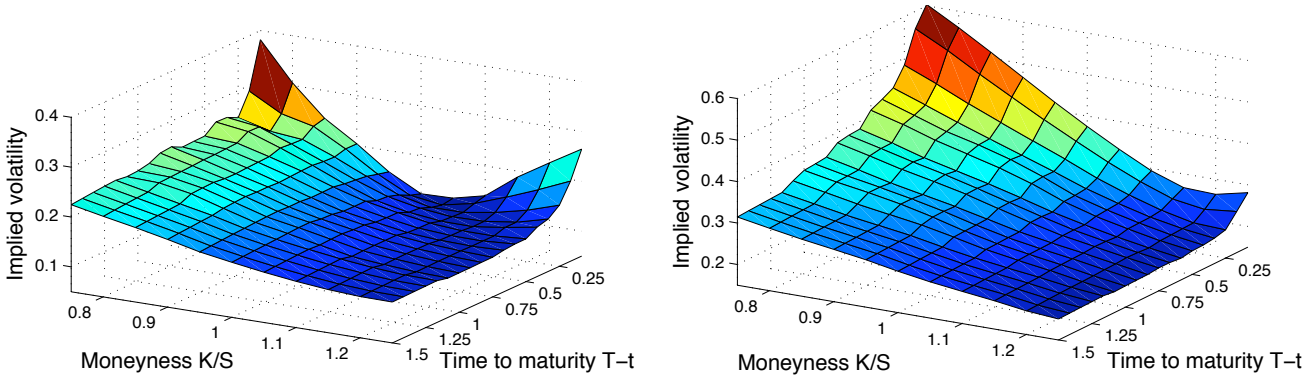
$$d^{\pm} := \frac{\ln(S_t/K) + (r - d \pm \frac{\sigma^2}{2})\tau}{\sigma\sqrt{\tau}},$$
$$\tau := T - t.$$

Note that the call price is independent of the drift term μ. The impact of their work has been tremendous. Markets trading in financial derivatives expanded rapidly, resulting in a massive growth of the amount of traded assets, liquidity and demand for more exotic securities. A large extent of derivative pricing is based on their idea of replicating the pay-off function H. With the increased liquidity nowadays, there is no need to theoretically price the basic European options anymore; the price of these so-called vanillas are now driven by the market. The search is now after the pricing of (possibly not replicable) contingent claims H .

In the BS framework there is a one-to-one correspondence between the theoretical European option price and the volatility parameter σ. Let $\widehat{C}_t$ denote the observed market price of a standard European call option with strike K and maturity T. The volatility matching $\widehat{C}_t$ is termed the implied volatility $\widehat{\sigma}$, i.e. $\widehat{\sigma}$ is such that

$$\widehat{C}_t(K, T) = C_t^{BS}\big(S_t, K, T, r, d, \widehat{\sigma}\big),$$

at $t \in [0, T]$. We can construct a surface of implied volatilities across strike and maturity. Specifying the implied volatility surface at any given date is equivalent to specifying the prices of all vanillas at that date, without losing information. Market data shows that $\widehat{\sigma} = \widehat{\sigma}_t(K, T)$, i.e. the implied volatility depends on time, strike and maturity, the latter usually being referred to as the term structure. Typical examples of implied volatility surfaces can be seen in Figure 1.1 below.



(a) 18-02-11

(b) 08-08-11

Figure 1.1: Typical volatility surfaces observed in equity markets. Taken from the S&P500 Index.

Looking at the surface as a function of strike, we see that the surface reveals a skewed smile pattern. In equity markets the surface commonly shows to have pure skew or a skewed smile, whereas it can be even increasing in commodities markets and purely smiling in FX markets. Even within the same asset class the surface can have different shapes: The Japanese Nikkei 225 Index generally shows little to no skew. A skewed volatility surface was barely observed in equity markets before the financial crisis in 1987 where equity indices lost more than 20% of its value. There are several explanations that cover for the skew's reason of being. A theoretical explanation is given by the fact that the BS framework assumes the light tailed lognormal distribution of returns of the underlying asset, where empirically it's found that the distribution shows to be heavy tailed. From a more practical trader's perspective the reason is primarily due to the general fear of massive losses. Since the option price is strictly increasing in σ, traders tend to increase $\widehat{\sigma}$, i.e. overprice, low strike options to cover for eventual losses in an unbalanced portfolio occurring when the asset gaps. This risk is clearly larger for almost maturing options, which explains the surface being more pronounced for small maturities.

We see that we violate one of the main assumptions made in the BS framework, being that $\widehat{\sigma}$ remains constant over time. While this is a crucial violation, there are other fairly unrealistic assumptions including a constant risk-free interest rate r, no transaction costs and that the underlying S does not pay dividends during the lifetime of the contract. Although there are some assets that are not subject to dividends, e.g. performance funds or indices like the German DAX Index, it has a substantial influence on the performance of the underlying and should hence be incorporated into stock modeling.

Due to these nonrealistic assumptions, the valuation and risk management of more exotic options in the BS framework is not accurate. To model derivatives more accurately, stochastic volatility (SV) models came to existence. These models, pioneered by Hull [Hul00], Stein & Stein [Ste91] and most popular models Heston [Hes93] and SABR [Hag02], assume that the nature of the instantaneous volatility is in fact stochastic and is thus modeled directly as a random variable. To understand the dynamics of such a model, we look at the Heston model more closely, in which the stock price is described according to the SDE

$$\frac{dS_t}{S_t} = \mu dt + \sqrt{\nu(t)} dW_t^1, \quad S_0 = s.$$

The volatility is said to be of the Cox-Ingersoll-Ross [Cox85] type

$$d\nu_t = \kappa(\theta - \nu(t))dt + \sigma\sqrt{\nu(t)}dW_t^2,$$

which is mean reverting, a commonly observed future of volatility in market data. The parameters of this model are given by κ, θ, ν_0, σ, and ρ, where κ controls the speed of mean reversion, θ the level of mean reversion, $\sqrt{\nu_0}$ the short volatility (in contrast to $\sqrt{\theta}$ being the long volatility), σ the volatility of volatility ("volvol") and ρ the correlation between the $\mathbb{P}$-Brownian motions W^1, W^2. A negative correlation gives rise to earlier discussed downward skew.
The popularity of this model lies within the fact that European options can be priced using a semi-closed form Fourier transformation, easing the calibration to market data. This in contrast to many other SV models; most of these models do not have a closed form or direct way to price European options and hence calibration can be cumbersome. Since SV models have one extra source of randomness generated by Brownian motion W^2, we encounter a risk exposure to volatility. Delta hedging becomes insufficient to eliminate risk and a market with the underlying asset and a risk-free money account becomes incomplete. An incomplete market leads to non-unique prices for most contingent claims. From this we see that every additional source of randomness in the model requires an additional tradable instrument to eliminate the additional risk. Trading in volatility derivatives such as variance swaps becomes more and more liquid these days, easing the hedging of volatility risk. However, we will see later that the valuation of these derivatives is far from standardized.

Moving away from the stochastic nature of volatility, the smile can also fitted by the less computationally complex local volatility (LV) models. Dupire [Dup94] (in continuous time) and Derman & Kani [Der94] (in discrete time) noted the existence of a unique diffusion process consistent with the risk-neutral density obtained from market prices of vanilla options. In contrast to the SV models, Dupire's main drive was to keep the market complete. The volatility is denoted as the state and time dependent diffusion coefficient $\sigma_{LV} = \sigma(S_t, t)$, given by

$$\sigma_{LV}^2(S_t, t) = 2\frac{\partial_T C_t + (r_T - q_T)K\partial_K C_t + q_T C_t}{K^2\partial_{KK} C_t}, \tag{1.3}$$

where r_T and q_T denote the risk-free rate and dividend yield corresponding to maturity T. This is said to be Dupire's formula, but was in fact derived by Derman & Kani using Dupire's method.
Since there is no additional randomness involved, delta-hedging is enough to eliminate risk and the market is complete. However, the LV approach relies upon the continuity of C_t and its derivatives with respect to strike K and maturity T. Since the option price will usually not be an analytical function and only observed on discrete data points, we will have to obtain a continuum of option prices across all strikes and maturities by interpolation or smoothing. These prices are then constrained to guarantee that the option price is decreasing and convex in K and increasing in T, to avoid complex local volatilities. The restriction $\partial_T C_t$ can be very unreasonable in the presence of dividends. We will have to estimate these derivatives, inducing errors. Consider the denumerator of (1.3) for far out-of-the-money and far in-the-money strikes, in particular close to maturity. The derivative in the denominator will be very small and small errors in the approximation will multiplied with a squared strike, blowing up the absolute error.
At each calibration of the LV surface, the LV surface is a deterministic function of S_t and t, but over time this surface shows to have stochastic dynamics. LV models thus predict wrong dynamics for the stock's volatility and is hence not suitable for pricing and hedging exotic derivatives, especially those derivatives that are exposed to volatility. It has been shown that BS hedging performs even better (e.g. an empirical study by Dumas et al. [Dum98] on the S&P500 Index).

By far the most literature is focused on the above approaches. There is vast literature available trying to use a Lévy process to describe the stock price, see for example [Con03]. These processes are of the form

$$S_t = S_0 e^{X_t}, \quad S_0 = s,$$

where X_t has to satisfy certain integrability conditions. X_t usually contains a drift, a Poisson component to account for jumps in the process and a Gaussian component to account for fluctuations. Note that the BS model is a special case of the Lévy approach with X_t being a drift plus Brownian motion. The main advantage of these models comes down to the calibration and fit of the model to short maturities. Most Lévy processes have an expression for the option price in terms of a characteristic function $\phi_{X_t}(u) := \mathbb{E}[e^{iuX_t}]$ which can be used to calibrate against market data. Often this is the only tool to use, as the distribution function of a Lévy process is generally not possible to find. Lévy processes also allow for jumps, which covers the tail problem in the distribution discussed above.

All previous models are free of arbitrage by the specification of a pricing measure $\mathbb{P}^* \sim \mathbb{P}$ and are then calibrated or hopefully consistent with the smile. Further knowledge of the joint dynamics of the stock and its options is not ensured. If we are to use the stock together with a range of liquid reference options, a more natural approach would be to model the evolution of these via a system of SDEs directly, arriving at the realm of market models. Such a framework has the advantage that the options are an integral part of the model and that the model yields hedging strategies directly in terms of the reference options.

In a market model one tries to parameterize the surface of reference options $\widehat{C}_t(K,T)$ or $\widehat{\sigma}_t(K,T)$ using a one-to-one function which translates the prices into the so-called code-book. Since the code-book is an one-to-one translation of the call prices, we can specify a system of SDEs describing its dynamics and equivalently have the call prices. We then get around the problem of calibration to the volatility surface, since price surface serves as the input to our model. The SDEs specifying the price dynamics are however subject to no-arbitrage and integrability conditions.
Market models have their roots in interest rate modeling, where normal short rate models have to be calibrated against the yield curve and market models take the market's yield curve as input. For equity modeling, market models aimed to to use implied volatility as a code-book. This means that the one-to-one translation of the call price is chosen to be the inverse of the BS pricing function C^{BS} (1.1) with respect to the volatility parameter σ. There are great advantages motivating the use of a model specified in terms of implied volatility. In difference to quantities such as the previously discusses instantaneous volatility (SV), LV or Lévy models, a model based on implied volatility is much easier to understand for the practitioner. They are familiar with the concept, encouraged by relatively recent inceptions of volatility indices. Furthermore, the movements in the implied volatility surface across K and T are highly correlated, hence their joint dynamics will be driven by only a few factors, resulting in computational benefits.

The main problem in a market model approach comes from ensuring the absence of arbitrage, which can be divided into static and dynamic arbitrage. Absence of static arbitrages constrains the call option surface $(C_t(K,T))_{K\in\mathcal{K},T\in\mathcal{T}}$ for every fixed t in such way that we can not generate a risk-free profit by an one-off trading strategy. The absence of static arbitrage means that when we define a code-book by SDEs, the first thing to check is whether the constraints on the price surface are not broken. The parameterization should be chosen such that the static arbitrage restrictions do not result in complicated state space for quantities involved in the model. Usually the bounds are implied by the absence of dynamic arbitrage. Dynamic arbitrage is absent under the existence of a local martin-

gale measure $\mathbb{P}^* \sim \mathbb{P}$, usually ensured by a drift restriction on the system of SDEs. The drift restrictions will then depend on the Gaussian component v of the SDEs, leaving those as the degree of freedom. When a particular volatility structure v is chosen, it remains to show the existence of a solution to the SDE. This turns out to be a cumbersome procedure due to the explicit dependence of the drift on volatility v.

The original interest rate market models choose forward rates as a parameterization and then need to satisfy certain drift restrictions established by Heath et al. [Hea92] to exclude arbitrage opportunities. This idea was extended to equity markets by Schönbucher [Sch99] using both implied volatility and forward implied volatility as a parameterization. He manages to specify drift conditions to exclude dynamic arbitrage and ensures static no-arbitrage conditions for $\mathcal{K} = \{K\}, \mathcal{T} = (0, \infty)$. Similar results were obtained by Jacod & Protter [Jac10] and Schweizer & Wissel [Sch08b], where Schweizer & Wissel also cover the existence. The opposite case, $\mathcal{K} = (0, \infty), \mathcal{T} = \{T\}$, is covered by Schweizer & Wissel [Sch08a] using local implied volatility. On the full surface necessary conditions have been constructed for the dynamics by Brace et al. [Bra01], Ledoit et al. [Led02], but no sufficient conditions, nor proof of existence have been given.

In [Sch08a] it is argued that classical implied vols are an ill-suited code-book. To overcome this problem they invoke the concept of local implied volatility and a price level. This concept is extended in Wissel's dissertation [Wis08] to a finite grid of strikes and maturities, coming with existence results inspired by their joint work. Although not holding for strike and maturity, the no-arbitrage constraints are directly computed and the degree of freedom is large. Similar work by Carmona & Nadtochiy [Car08] got published around the same time where they derive a similar model, but their drift restrictions are not available in closed form: a PDE for the call price needs to be solved at every simulation time t. Carmona & Nadtochiy do derive an existence result.
Wissel's model has been the main motivation for this thesis. We use his framework and incorporate dividends, which can be extended to stochastic. As a by-product we look at ways to smooth the call price surface in an arbitrage-free way. We wish to analyze the efficiency and usability of a market model as opposed to the straight forward single stock models. We will look at ways to calibrate the volatility structure and the possibilities for such a model to price exotic derivatives. Preferably the model should have a parameterization that is easy to interpret, such as implied volatility, the original motivator of equity market models. We would also like the parameters of the model to have a clear influence on the forthcoming dynamics of both the options and the underlying. In particular we are interested in the greater possibilities market models could bring compared to the single stock modeling approach.

The rest of this thesis is organized as follows. Section 2 goes through the background in stochastic calculus needed to comprehend the analysis throughout the thesis. We will introduce the concepts of forwards and show how to disconnect the influence of interest rates and dividends from the stock price. We'll have an in-depth look on how local volatility is derived and go through the concept of static arbitrages and hedges. Section 3 builds the theoretical background of the model we will use, mainly based on Wissel's concept of local implied volatlity in [Wis08]. We will then look at how such a model can be implemented together with the problems we find along the way in section 4. The volatility structure will need calibration, which becomes an involved procedure explained in section 5. We apply the model in section 6 to price exotic derivatives such as forward start options and (forward start) variance swaps. We will also see how the model can be put to use in the pricing of structured products. Section A discusses the code with its pitfalls after which we conclude our results in section 7.

2 Preliminary results

In this section we will discuss technical details which will serve as building blocks for the theory developed later on, such as local martingales and Itô processes. A reader familiar with these advanced concepts in stochastic calculus might want to go to section 2.2 immediately, where we start specifying the stock price dynamics. Following [Bue09], we will build a stock price process that is purely exposed to volatility, leaving interest rates and dividend yields on the side via an affine transformation. Section 2.3 derives the concept of Dupire's local volatility by relying on the Fokker-Planck equation. We will exploit the consequences of a statically arbitrage-free option model in section 2.4.

2.1 Stochastic calculus

We will model on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$, where $\mathbb{F} := (\mathcal{F}_t)_{t\in[0,T]}$ the $\mathbb{P}$-augmented filtration generated by a D-dimensional Brownian motion $W = (W_t^1, \dots, W_t^D)_{t\in[0,T]}$, $T < \infty$ and $D \in \mathbb{N}$. Write $\mathcal{F} = \mathcal{F}_T$.
Let $\mathcal{B}(E)$ denote the σ-algebra generated by the set $E \subset \mathbb{R}$ and introduce the notations $x^+ := \max(x, 0)$, $x \vee y := \max(x, y)$ and $x \wedge y := \min(x, y)$.

Definition 2.1.
A process $(X_t)_{t\in[0,T]}$ defined on the filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$ is a local martingale under $\mathbb{P}$ if X is adapted to its filtration $\mathbb{F}$ such that:

- *there exists a sequence of stopping times $(\tau_k)_{k=1,2,\dots}$ such that $\tau_k \le \tau_{k+1}$ $\mathbb{P}$-a.s. for each k,*
- *$\lim_{k\to\infty} \tau_k = \infty$ $\mathbb{P}$-a.s.,*
- *the stopped process*

$$X_t^k := X_{t\wedge\tau_k}$$

is a martingale under $\mathbb{P}$ for for each k and $t \in [0, T]$.

Recall the definition of a progressively measurable process.

Definition 2.2.
A process $(X_t)_{t\in[0,T]}$ defined on the filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$ is said to be progressively measurable or simply progressive if the map $(s, \omega) \to X_s(\omega)$ from $[0, T] \times \Omega \to \mathbb{R}$ is $\mathcal{B}[0, T] \otimes \mathcal{F}$ measurable

We can now define the space $\mathcal{L}^p_{loc}(\mathbb{R}^D)$ as the space of all $\mathbb{R}^D$-valued progressively measurable, locally p-integrable processes on $[0, T]$, with $D, p \in \mathbb{N}$, i.e. if $X \in \mathcal{L}^p_{loc}$, then X is a D-dimensional progressive process and there exists a sequence of stopping times $(\tau_k)_{k=1,2,\dots}$ such that $\tau_k \le \tau_{k+1}$ $\mathbb{P}$-a.s. for each k, $\lim_{k\to\infty} \tau_k = \infty$ $\mathbb{P}$-a.s. and for each k

$$\mathbb{E}\left[\left|X^k\right|^p \right] < \infty.$$

Recall that two measures $\mathbb{P}$ and $\mathbb{P}^*$ are defined to be equivalent if they share the same sets of probability zero, i.e.

$$\mathbb{P}(A) = 0 \iff \mathbb{P}^*(A) = 0, \quad A \in \mathcal{F},$$

which we denote by $\mathbb{P} \sim \mathbb{P}^*$. We can now define the notion of a risk-neutral measure.

Definition 2.3.
A risk-neutral measure, also called an equivalent local martingale measure or a pricing measure, is a probability measure $\mathbb{P}^*$ on $(\Omega, \mathcal{F})$ such that $\mathbb{P}^* \sim \mathbb{P}$ and the stock price S is a local martingale under $\mathbb{P}^*$.

The dynamics of a stochastic process can be described via Itô calculus. Consider the SDE

$$dX_t = \mu_t(X_t)dt + \sigma_t(X_t)dW_t, \quad X_0 = x_0. \tag{2.1}$$

The term μ has to be interpreted as a non-stochastic drift term and σ the Gaussian noise component. The larger σ becomes, the more volatile the process behaves relative to the drift. Setting $\mu \equiv 0$ yields X to be a martingale. SDE (2.1) is said to have a strong solution if X is an Itô process.

Definition 2.4.
A process X is called an N-dimensional Itô process on $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$ driven by a D-dimensional Brownian motion W if it is a $\mathbb{P}$-a.s. continuous, $\mathbb{F}$-adapted process $X = (X_t)_{t\in[0,T]}$ satisfying $\mathbb{P}$-a.s.

$$X_t = x_0 + \int_0^t \mu_s(X_u)ds + \sum_{d=1}^{D}\int_0^t \sigma_s^d(X_s)dW_s, \quad t \in [0,T], \tag{2.2}$$

with

- $(\mu_t)_{t\in[0,T]}$ *a N-dimensional, progressive process such that $\int_0^T |\mu_s|\, ds < \infty$ $\mathbb{P}$-a.s.,*
- $(\sigma_t^d)_{t\in[0,T]}$ *a N-dimensional, progressive process such that $\int_0^T (\sigma_s^d)^2\, ds < \infty$ $\mathbb{P}$-a.s. for each $d = 1, \dots, D$,*
- x_0 *is $\mathcal{F}_0$-measurable.*

Throughout this thesis we will only consider the case where $N = 1$. One of our main concerns is whether there exists a solution to SDE (2.1), which is the case when X defines an Itô process with its solution given as (2.2). To this end, we need to introduce the concept of (local) Lipschitz continuity.

Definition 2.5.
Let $f : \mathbb{R} \to \mathbb{R}$. Then f is called Lipschitz continuous if there exists an $L \in \mathbb{R}$ such that for all $x, y \in \mathbb{R}$ we have

$$|f(x) - f(y)| \le L|x - y|,$$

where L is a constant, independent of x and y. L is also called the Lipschitz constant.
Moreover, f is called locally Lipschitz if for every $N > 0$ there exists an $L_N > 0$ such that f is Lipschitz continuous with Lipschitz constant L_ϵ provided that $|x|, |y| \le N$.

The Lipschitz condition can be seen as a strong form of continuity. A Lipschitz function is limited in its growth, as any two points in its domain have a slope between them limited by a real-valued Lipschitz constant. Using the concept of local Lipschitz continuity, we obtain the following result.

Theorem 2.1.
Assume that for all $t \in [0,T]$ the functions $\mu_t(x)$ and $\sigma_t(x)$ are Lipschitz continuous functions of x with Lipschitz constant L_N independent of t, x. Furthermore, assume that for all $t \in [0,T]$ the functions $\mu_t(x)$ and $\sigma_t(x)$ satisfy the linear growth condition

$$|\mu_t(x)| + |\sigma_t(x)| \le L(1 + |x|),$$

with $L > 0$. Then, for any $\mathcal{F}_0$-measurable square integrable random variable X_0 the SDE (2.1) admits a strong solution and is unique on $[0,T]$. Moreover, if additionally $\mathbb{E}[X_0^2] < \infty$ holds, we have

$$\mathbb{E}\left[\sup_{t\in[0,T]} |X_t|^2\right] < L_{N,T} \cdot \left(1 + \mathbb{E}[X_0^2]\right)$$

for all $t \in [0,T]$, where the constant $L_{N,T}$ depends on L_N and L_T.

Proof. See Theorem 2.2 in chapter 5 of Friedman [Fri75]. □

The following definition holds in fact for all continuous semimartingales, a concept that lies beyond the scope of this thesis. An Itô process is one of its most common examples and suffices for our purposes.

Definition 2.6.
Let X be an Itô process, $P = \{0 = t_0 \leq t_1 \leq \ldots \leq t_I = t\}$ and set $|P| := \max\{t_i - t_{i-1} | i = 1, \ldots, I\}$. The Doléans-Dade exponential $\mathcal{E}(X)$ is defined by

$$\mathcal{E}_t(X) := \exp\left(X_t - X_0 - \frac{1}{2}\langle X\rangle_t\right),$$

where $\langle X\rangle_t$ denotes the quadratic variation of X, i.e.

$$\langle X\rangle_t := \lim_{|P|\to 0} \sum_{i=1}^{I} (X_{t_i} - X_{t_{i-1}})^2, \quad \mathbb{P} - a.s..$$

The Doléans-Dade exponential will become important later on when we want to switch between the measures $\mathbb{P}$ and $\mathbb{P}^*$. We will need the following theorem.

Theorem 2.2.
Let X be a continuous local martingale with $X_0 = 0$. Then $\mathcal{E}(X)$, its Doléans-Dade exponent, is a local martingale.

Proof. Define $f(X_t, \langle X\rangle_t) := \mathcal{E}_t(X)$. Itô's fomula gives

$$\begin{aligned} df &= \mathcal{E}_t(X)dX_t + \frac{1}{2}\mathcal{E}_t(X)d\langle X\rangle_t - \frac{1}{2}\mathcal{E}_t(X)d\langle X\rangle_t \\ &= \mathcal{E}_t(X)dX_t. \end{aligned}$$

Using $\mathcal{E}_0(X) = 1$, we can solve this to

$$\mathcal{E}_t(X) = 1 + \int_0^t \mathcal{E}_s(X)X_s ds, \quad t \geq 0,$$

which is a stochastic integral with respect to X. From results in stochastic integration such as in [Pro04] we know that the local martingale property is preserved under stochastic integration, which concludes the proof. □

2.2 Stock price dynamics

Stock prices are subject to interest rates and dividend payments. We will assume that the interest rates $r = (r_t)_{t\in[0,T]}$ are deterministic. We assume that dividend payments can be considered in two separate classes; On each dividend date $0 = \tau_0 < \tau_1 < \tau_2 \ldots$ first a proportional dividend $\beta_k = 1 - e^{-d_k}$ and after which an absolute cash dividend α_k is paid. This means that at a dividend date τ_k the stock drops to a relative amount e^{-d_k} after which α_k is subtracted, i.e.

$$S_{\tau_k} = S_{\tau_k -} e^{-d_k} - \alpha_k, \tag{2.3}$$

where $S_{\tau_k -}$ denotes the stock price just before the payments of the dividend. Here we make the assumption that the ex-dividend date, record date and payment date all coincide. In practice the share trades cum-dividend until the ex dividend date, after investors acquiring the share will not be entitled to receive the dividend. Exchanges then usually have a fixed amount of days until the record date, where the company looks in the shareholder register

to allocate the payments. The amount of days between the ex dividend date and record date is at least the time needed for the share trade to settle. The actual dividend payment can be days away from the record date. We can now define the future price of S maturing at time T, arguing by replication. Buying n shares of stock equals to an investment of nS_t zero-coupon bonds at time t. We will have the stock which will give us dividends. Reinvesting all proportional dividends leads to a long position of

$$n \exp\left(\sum_{k:t<\tau_k\leq T} d_k\right) \tag{2.4}$$

at time T. Reinvesting d leads to a cash dividend payment at time τ_k of

$$n\alpha_k \exp\left(\sum_{j:t<\tau_j\leq\tau_k} d_j\right),$$

which we use to pay back the debt obtained from buying the stocks. Then, at maturity T, we are short

$$nS_t \exp\left(\int_t^T r_u du\right) - n\sum_{k:t<\tau_k\leq T} \alpha_k \exp\left(\sum_{j:t<\tau_j\leq\tau_k} d_j\right) \exp\left(\int_{\tau_k}^T r_u du\right)$$

zero-coupon bonds. In (2.4) we see that in order to reproduce exactly one share, we should buy $n = e^{-\sum_{k:t<\tau_k\leq T} d_k}$ shares of stock at time t, whence the future price F should be

$$F_t(T) = S_t \exp\left(\int_t^T r_u du - \sum_{k:t<\tau_k\leq T} d_k\right) - \sum_{k:t<\tau_k\leq T} \alpha_k \exp\left(\int_{\tau_k}^T r_u du - \sum_{j:\tau_k<\tau_j\leq T} d_j\right).$$

Following Buehler [Bue09], define

$$R_t(T) := \exp\left(\int_t^T r_u du - \sum_{k:t<\tau_k\leq T} d_k\right).$$

and so the future price becomes

$$F_t(T) = R_t(T)\left(S_t - \sum_{k:t<\tau_k\leq T} \frac{\alpha_k}{R(t,\tau_k)}\right). \tag{2.5}$$

Note that the future price has to be non-negative, since the stock price can't become negative. This means that

$$S_t \geq \sum_{k:t<\tau_k\leq T} \frac{\alpha_k}{R_t(\tau_k)}.$$

Since this holds for all T, the stock is bounded from below by a "discounted growth-rate", meaning that

$$\begin{aligned} &S_t \geq \mathcal{A}_t, \\ &\mathcal{A}_t := \sum_{k:\tau_k>t} \frac{\alpha_k}{R_t(\tau_k)}. \end{aligned}$$

This lower bound indeed makes sense, as all dividends are tradable assets via e.g. dividend swaps and zero strike calls; the stock should at least exceed their discounted value. This conflicts with the BS framework, in which the SDE (1.1) models a share price that can come arbitrary close to zero with positive probability. Turning to an

alternative process X that does not suffer from this drawback, we know it needs to allow for the existence of a pricing measure $\mathbb{P}^* \sim \mathbb{P}$, such that X is a local martingale to exclude dynamic arbitrage. Buehler proves that the only consistent way to model a local martingale representing the "true-part" $X_t = F_t - \mathcal{A}_t$ of the stock price, while ensuring the future price F is priced correctly, is by

$$S_t = F_0(t)X_t + \mathcal{A}_t, \tag{2.6}$$

$$F_0(t) := \left(S_0 - \sum_{k=1}^{\infty} \frac{\alpha_k}{R_0(\tau_k)} \right) R_0(t), \tag{2.7}$$

where X_t is thus a non-negative local martingale "pure" stock price process with $X_0 = 1$. The main advantage of switching to such a pure process is that X is now purely exposed to volatility and not to external equity influencing factors such as interest rates and dividends, which will take away some of the problems we encounter using local volatility as a code-book for our call prices later on. Buehler in fact takes into account repo rates and most importantly credit risk. The former represents the rate earned on lending out money with stock as collateral and the latter represents the exposure to risk that the counterparty does not pay back its liabilities. Both can be taken out of the original stock price process. Buehler therefore focuses on the forward price, rather than the future price. The future is an exchange traded contract and so a clearing house takes care of the credit risk. The forward is an over-the-counter (OTC) contract, mutually agreed between two parties. Since this is done off-exchange, credit risk becomes an important factor in the price F.

Put-call parity revised

The second fundamental theorem of asset pricing guarantees the existence of a unique pricing measure $\mathbb{P}^*$ in a complete market. The price C of a contingent claim H is then given by the discounted expected value of the payoff taken under $\mathbb{P}^*$, i.e.

$$C_t(S_t, T, \cdot) := B_t(T) \cdot \mathbb{E}^* \big[H(S_T) \big| \mathcal{F}_t \big],$$
$$B_t(T) := \exp\left(- \int_t^T r_u du \right),$$

where we have used $\mathbb{E}^* := \mathbb{E}_{\mathbb{P}^*}$ to distinguish expectations taken under $\mathbb{P}$ as opposed to $\mathbb{P}^*$. B can be seen as the price of a zero-coupon bond paying one unit of cash at maturity T. Buhler's affine transformation of the stock price (2.6) makes it possible to express call prices on S expressed in prices on X. Let $\widetilde{C}_t, C_t$ denote call prices on S respectively X, then

$$\begin{aligned} \widetilde{C}_t(S_t, K, T, \cdot) &:= B_t(T) \cdot \mathbb{E}^* \left[(S_T - K)^+ | \mathcal{F}_t \right] \\ &\overset{(2.6)}{=} B_t(T) \cdot F_t(T) \cdot \mathbb{E}^* \left[\left(X_T - \frac{K - \mathcal{A}_T}{F_t(T)} \right)^+ \middle| \mathcal{F}_t \right] \\ &= B_t(T) \cdot F_t(T) \cdot C_t \left(X_t, \frac{K - \mathcal{A}_T}{F_t(T)}, T \right). \end{aligned} \tag{2.8}$$

For put options we can use the same arguments to get

$$\widetilde{P}_t(S_t, K, T, \cdot) = B_t(T) \cdot F_t(T) \cdot P_t \left(X_t, \frac{K - \mathcal{A}_T}{F_t(T)}, T \right).$$

Recall the put-call parity

$$\widetilde{C}_t(S_t, K, T, \cdot) - \widetilde{P}_t(S_t, K, T, \cdot) = B_t(T)(F_t(T) - K),$$
$$C_t(X_t, K, T) - P_t(X_t, K, T) = X_t - K.$$

2.3 Local volatility

Dupire [Dup94] approached the call price surface by means of local volatility, assuming the availability of a continuum of European option prices $C(T,K)$. His idea was to find a function $\sigma : \mathbb{R}^2_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ such that a solution to

$$\frac{dX_t}{X_t} = \sigma_t(X_t)dW_t$$

exists, is unique and has the martingale property. The price of a call option on X with strike K and maturity T can be expressed as

$$C_t(X_t,K,T) = \mathbb{E}^*\left[(X_T-K)^+|\mathcal{F}_t\right] = \int_{\mathbb{R}}(x-K)^+ f_T(x)dx = \int_K^\infty (x-K)f_T(x)dx,$$

where f_T denotes density of X_T under $\mathbb{P}^*$, i.e. f_T is the risk-neutral density of X_T. The risk-neutral density can be deduced from market data via

$$\partial_K C_t = -\int_K^\infty f_T(x)dx, \tag{2.9}$$

$$\partial_{KK} C_t = f_T(K), \tag{2.10}$$

where (2.10) holds under the assumption that $\lim_{x\to\infty} f_T(x) = 0$. Dupire noted that there is a unique diffusion process satisfying the equation

$$\partial_T f_T(x) = \frac{1}{2}\partial_{xx}[x^2\sigma^2 f_T(x)], \tag{2.11}$$

In this equation f_T is known and the volatility coefficient σ is unknown. Note that Dupire solved the inverse of the Fokker-Planck equation, in which σ is known and f_T to be found. We can now use Dupire's equation to get

$$\partial_T C_t = \int_K^\infty (x-K)\partial_T f_T(x)dx \overset{(2.11)}{=} \frac{1}{2}\int_K^\infty (x-K)\partial_{xx}[x^2\sigma^2 f_T(x)]dx.$$

Assuming $\lim_{x\to\infty}\partial_x[\sigma^2x^2 f_T(x)] = 0$, integration by parts gives

$$\begin{aligned}
\partial_T C_t &= \frac{1}{2}\left[\int_K^\infty (x-K)\partial_x[x^2\sigma^2 f_T(x)]dx\right]_{x=K}^{x=\infty} - \frac{1}{2}\int_K^\infty \partial_x[x^2\sigma^2 f_T(x)]dx \\
&= -\frac{1}{2}\int_K^\infty \partial_x[x^2\sigma^2 f_T(x)]dx \\
&= \frac{1}{2}\sigma^2K^2 f_T(K) \\
&\overset{(2.10)}{=} \frac{1}{2}\sigma^2K^2\partial_{KK}C_t,
\end{aligned}$$

with $\sigma = \sigma_t(X_t,K,T)$. By absence of static arbitrage, both sides are positive. By isolating σ, we can thus write

$$\sigma_t(X_t,K,T) = \sqrt{\frac{2}{K^2}\frac{\partial_T C_t(X_t,K,T)}{\partial_{KK}C_t(X_t,K,T)}}. \tag{2.12}$$

We can now see the main advantage of using pure process X. If we had sticked to modeling call options on S instead, dividends would have caused problems. A high dividend yield compared to low interest rates, especially on low strikes, and cash dividends can make the time derivative of a call option become negative. A call option maturing before dividend payments will usually have a higher expected payoff than call options maturing straight after. Dupire derived the local volatility (2.12) in [Dup96] by taking the square root of the risk-neutral expectation of the instantaneous variance v conditioned on the final stock price S_T equal to the strike, i.e.

$$\sigma_t^2(X_t,K,T) = \mathbb{E}^*[v_T|X_T = K].$$

2.4 Static arbitrage

As mentioned in the introduction, the call price surface is subject to constraints to exclude static arbitrage. These conditions are given by

(i) increasing in T,

(ii) nonincreasing and convex in K,

(iii) converge to X_t as $K \to 0$ and to 0 as $K \to \infty$.

To see this, we first look at (i). Assume that at time t we observe $C_t(K, T_1) \geq C_t(K, T_2)$ on some strike K and expiry dates $t < T_1 < T_2$. We will see that this leads to an arbitrage opportunity via portfolio V_t, constructed by shorting call $C_t(K, T_1)$ and buying call $C_t(K, T_2)$. Note that this portfolio gives us an initial non-negative cash amount already. Then, at $t = T_1$, our portfolio is worth

$$V_{T_1} = C_{T_1}(K, T_2) - (X_{T_1} - K)^+ = \begin{cases} C_{T_1}(K, T_2), & X_{T_1} \leq K \\ C_{T_1}(K, T_2) - (X_{T_1} - K), & X_{T_1} > K. \end{cases} \tag{2.13}$$

In both cases we end up with a non-negative profit. In the first case we can sell the call maturing at T_2 or hold it until T_2 for a possible profit. In the second case, using that X is independent of interest rates and dividends, the value our portfolio at T_1 equals to the price of a put maturing at T_2 by put-call parity. We can lock in this profit by selling the put or holding the put until T_2 for a possible profit.
Looking at (ii), assuming that the call price is decreasing or not convex in K, we can construct a portfolio V_t called a butterfly spread. A butterfly spread consists of buying two calls with respective strike $K-\epsilon$ and $K+\epsilon$ and shorting two call options with K, all with the same maturity T. At $t = T$ this portfolio has payoff

$$V_T = C_T(K - \epsilon, T) - 2C_T(K, T) + C_T(K + \epsilon, T) = |X_T - K| \mathbf{1}_{\{|X_T - K| < \epsilon\}} \geq 0$$

For (iii), $K \to \infty$ is trivial and for $K \to 0$ we note that a call on $K = 0$ is like buying a call on the future price, i.e. the only fair price for this contract is given by

$$C_t(K = 0, T) = \mathbb{E}^* \left[(X_T)^+ | \mathcal{F}_t \right] = 1.$$

Remark 2.1.
In presence of interest rates and dividends the stock S doesn't converge to S_t as $K \to 0$. Using (2.5), we see that

$$\begin{aligned} C_t(K = 0, T) = B_t(T) \cdot \mathbb{E}^* \left[(S_T)^+ | \mathcal{F}_t \right] &= B_t(T) \cdot F_t(T) \\ &\overset{(2.5)}{=} B_t(T) \cdot R_t(T) \left(S_t - \sum_{k : t < \tau_k \leq T} \frac{\alpha_k}{R_t(\tau_k)} \right) \\ &= \exp\left(- \sum_{k : t < \tau_k \leq T} d_k \right) \left(S_t - \sum_{k : t < \tau_k \leq T} \frac{\alpha_k}{R_t(\tau_k)} \right) \\ &\neq S_t. \end{aligned}$$

As K goes to zero, the call price converges to the discounted future price. This argument will also show us that the surface doesn't have to be increasing in T. Consider two calls maturing at T_1 and T_2 such that $T_1 < \tau_k < T_2$

for dividend date τ_k. For ease of notation, assume there are no cash dividends. Provided that

$$\int_{T_1}^{T_2} r_u du < \sum_{k:T_1<\tau_k\leq T_2} d_k,$$

we get

$$\begin{aligned} F_t(T_1) &= S_t \exp\left(\int_t^{T_1} r_u du - \sum_{k:t<\tau_k\leq T_1} d_k\right) \\ &> S_t \exp\left(\int_t^{T_2} r_u du - \sum_{k:t<\tau_k\leq T_2} d_k\right) = F_t(T_2). \end{aligned}$$

By the argument for $K \to 0$ above, we see that there is a strike K sufficiently small to guarantee that the call price is not increasing in T for strikes smaller or equal to that K.

From now on we let the options on X be defined on the grid $\mathcal{K} = \{K_0, \ldots, K_{N+1}\} \times \mathcal{T} = \{T_0, \ldots, T_M\}$, where $0 = K_0 < K_1 < \ldots < K_N < K_{N+1} < \infty$ and $0 = T_0 < T_1 < \ldots < T_M < \infty$. Write

$$C_n^m(t) := C_t(K_n, T_m), \quad n = 0, \ldots, N+1,\ m = 1, \ldots, M.$$

A statically arbitrage free surface can be characterized via Davis & Hobson [Dav07].

Definition 2.1.
An option model $(C_n^m)_{n=0,\ldots,N+1,m=1,\ldots,M}$ is called free of static arbitrage if we have $\mathbb{P}$-a.s.

$$-1 \leq \frac{C_n^m(t) - C_{n-1}^m(t)}{K_n - K_{n-1}} \leq \frac{C_{n+1}^m(t) - C_n^m(t)}{K_{n+1} - K_n} \leq 0, \quad n = 1, \ldots, N, \tag{2.14}$$

for all $t \in [0, T_m)$, $m = 1, \ldots, M$,

$$C_n^{m-1}(t) \leq C_n^m(t), \quad n = 1, \ldots, N, \tag{2.15}$$

for all $t \in [0, T_{m-1}]$, $m = 2, \ldots, M$ and

$$\begin{aligned} \lim_{K\to 0} C_n^m(t) &= X_t, \\ \lim_{K\to\infty} C_n^m(t) &= 0, \end{aligned} \tag{2.16}$$

for all $t \in [0, T_m]$, $m = 1, \ldots, M$.
Moreover, the model is called admissible if strict inequalities hold in (2.14) for all $t \in [0, T_m)$, $m = 1, \ldots, M$ and in (2.15) for all $t \in [0, T_{m-1}]$, $m = 2, \ldots, M$.

In order for an option model to be admissible for $m = 1, \ldots, M$, we want the price surface to be strictly convex on $K_0, \ldots, K_{n+1}$. To this end, we have to make the assume for every m that

$$\mathbb{P}^*\left[\int_{K_{n-1}}^{K_n} f_{T_m}(x)dx > 0 \middle| \mathcal{F}_t\right] > 0,$$

for all $t \in [0, T_m)$ and $n = 1, \ldots, N+1$. For every set of strikes such that $K_{n-1} < K_n$ we then get

$$0 < \int_{K_n}^{\infty} f_{T_m}(x)dx < \int_{K_{n-1}}^{\infty} f_{T_m}(x)dx < 1$$

By (2.9), this is equivalent to

$$-1 < \partial_K C_{n-1}^m(t) < \partial_K C_n^m(t) < 0,$$

giving us strict inequalities in (2.14). By similar reasoning we have to assume that $\mathbb{P}^*\left[f_T(\cdot) > 0 \middle| \mathcal{F}_t\right]$ as a function of T

$$\partial_T C_n^m(t) = \partial_T \int_{K_n}^{\infty} (X_T - K)^+ f_{T_m}(x) dx$$

2.5 Static hedging

The great advantage of modeling an underlying with its options is the extended hedging strategies. The term static hedging means the replication of future liability by taking a position in the underlying, futures and options at only one trading time t. This is opposed to semi-static hedging and continuous hedging strategies, where it is allowed to rebalance your trading books a finite number of times respectively continuously. Note that in practise continuous hedging becomes semi-static hedging, since we can't trade on a continuous base.
Most European option payoffs have a convex payoff function $H : \mathbb{R}_{>0} \mapsto \mathbb{R}_{>0}$. Let $x, k \in \mathbb{R}_{>0}$. Following Carr [Car02], we can write

$$\begin{aligned} H(x) &= H(k) + \mathbf{1}_{x>k} \int_k^x H'(z)dz - \mathbf{1}_{x<k} \int_x^k H'(z)dz \\ &= H(k) + \mathbf{1}_{x>k} \int_k^x \left(H'(k) + \int_k^z H''(u)du \right) dz - \mathbf{1}_{x<k} \int_x^k \left(H'(k) - \int_z^k H''(u)du \right) dz \\ &= H(k) + H'(k)(x-k) + \mathbf{1}_{x>k} \int_k^x \int_k^z H''(u)dudz + \mathbf{1}_{x<k} \int_x^k \int_z^k H''(u)dudz \end{aligned}$$

By Fubini's theorem we get

$$H(x) = H(k) + H'(k)(x-k) + \mathbf{1}_{x>k} \int_k^x \int_u^x H''(u)dzdu + \mathbf{1}_{x<k} \int_x^k \int_x^u H''(u)dzdu$$

Working out the inner integral yields

$$\begin{aligned} H(x) &= H(k) + H'(k)(x-k) + \mathbf{1}_{x>k} \int_k^x H''(u)(x-u)du + \mathbf{1}_{x<k} \int_x^k H''(u)(u-x)du \\ &= H(k) + H'(k)(x-k) + \int_k^{\infty} H''(u)(x-u)^+ du + \int_0^k H''(u)(u-x)^+ du. \end{aligned}$$

Taking discounted risk-neutral expectations on both sides together then yields the value of H with maturity T and strike K

$$B_t(T) \cdot \mathbb{E}^*[H(S_T)|\mathcal{F}_t] = B_t(T)H(K) + H'(K)(C_t(T,K) - P_t(T,K)) + \int_0^K H''(u)P_t(T,u)du + \int_K^{\infty} H''(u)C_t(T,u)du.$$

Since the choice of strike K is arbitrary, we can set it equal to the future. By put-call parity the second term then vanishes and so

$$\mathbb{E}^*[H(S_T)|\mathcal{F}_t] = H\big(F_t(T)\big) + \exp\left(\int_t^T r_u du \right) \cdot \left(\int_0^{F_t(T)} H''(u)P_t(T,u)du + \int_{F_t(T)}^{\infty} H''(u)C_t(T,u)du \right). \quad (2.17)$$

We now see that any convex payoff can be perfectly hedged by investing in $B_t(T)H\big(F_t(T)\big)$ bonds and a continuum of out of the money calls and puts expiring at maturity T. In practice we clearly get an approximation, as there is no possibility to trade a continuum of these option contracts. Applications of this static hedging strategy include the the log contract which we will use to replicate contracts on realized variance. A static hedging strategy is model-independent and replicates the payoff H, hence there is no risk in selling these contracts when sold for the price of the hedging strategy.

3 Model setup

Since the parameterization of implied volatilities does not serve the market model approach very well, we pursue the approach of Wissel [Wis08], who used a discretized version of the local implied volatilities (2.12) joined with a parameterization by him and Schweizer in [Sch08a] for the case $\mathcal{K} = (0, \infty), \mathcal{T} = \{T\}$. The latter parameterization will be applied to the option prices on the closest maturity T_l, where $l := \min\{m : t < T_m\}$. We extend his approach mathematically by making it more more realistic and implementation friendly. We will explicitly rely on the affine transformation (2.6), which allows the model to incorporate (possibly stochastic) interest rates and dividends. After introducing the parameterization we specify a term structure for these quantities subject to drift restrictions to exclude dynamic arbitrage.

3.1 Parameterization

Our aim is to model the call option price processes

$$(C_n^m(t))_{t\in[0,T_m]} := (C_t(K_n, T_m))_{t\in[0,T_m]},$$

with $n = 0, \dots, N+1$ and $m = 1, \dots, M$. To have all price processes defined on $[0, T_M]$, we let $C_n^m(t) = C_n^m(T_m)$ for all $t \geq T_m$. We include the stock price process by letting $C_0^m(t) := X_t$ for all $t \leq T_m$. Finally, we take K_{N+1} such that $C_{N+1}^m(t) = 0$ for all $t \in [0, T_M]$ and $m \in \{1, \dots, M\}$, i.e. we have

$$X_t < K_{N+1} \ \ \forall t \in [0, T_M], \ \ \mathbb{P}\text{-a.s.}.$$

Note that we have taken care of the limits $K \to 0$ and $K \to \infty$ in Definition 2.1 by including $K_0 = 0$ and choosing K_{N+1} sufficiently high.

We can now define the price level and local implied volatilities, serving as the building blocks of our model. We stretch Definition 5.3 in [Wis08] to a free-to-choose reference strike by using [Sch08a].

Definition 3.1.
Let the option model $(C_n^m)_{n=0,\dots,N+1,\ m=1,\dots,M}$ be admissible. For each $l = 1, \dots, M$ we define for $t \in [T_{l-1}, T_l)$ and "reference strike" $K_{n^*} \in \mathcal{K}$ the price level y by

$$y(t) := \sqrt{T_l - t}\Phi^{-1}\left(\frac{C_{n^*}^l(t) - C_{n^*+1}^l(t)}{K_{n^*+1} - K_{n^*}}\right) \tag{3.1}$$

and the local implied volatilities $(x_n^m)_{n=1,\dots,N}$ for $m \geq l$ by

$$x_n^l(t) := \frac{K_{n+1} - K_n}{K_n\sqrt{T_l - t}} \Bigg/ \left(\Phi^{-1}\left(\frac{C_{n-1}^l(t) - C_n^l(t)}{K_n - K_{n-1}}\right) - \Phi^{-1}\left(\frac{C_n^l(t) - C_{n+1}^l(t)}{K_{n+1} - K_n}\right)\right), \tag{3.2}$$

$$x_n^m(t) := \sqrt{\frac{2}{K_n^2}\frac{C_n^m(t) - C_n^{m-1}(t)}{T_m - T_{m-1}} \Bigg/ \frac{(K_n - K_{n-1})C_{n+1}^m(t) - (K_{n+1} - K_{n-1})C_n^m(t) + (K_{n+1} - K_n)C_{n-1}^m(t)}{(K_{n+1} - K_n)(K_n - K_{n-1})(K_{n+1} - K_{n-1})/2}}, \tag{3.3}$$

for $m > l$. So y is a process on $[0, T_M)$ and each x_n^m a process on $[0, T_m)$.

Definition 3.1 invokes the price level $y \in \mathbb{R}$ and local implied volatilities $x \in \mathbb{R}_{>0}$, which are well-defined as long as the option model is admissible. The price level y and local implied volatilities x_n^l are the discretized analogue of

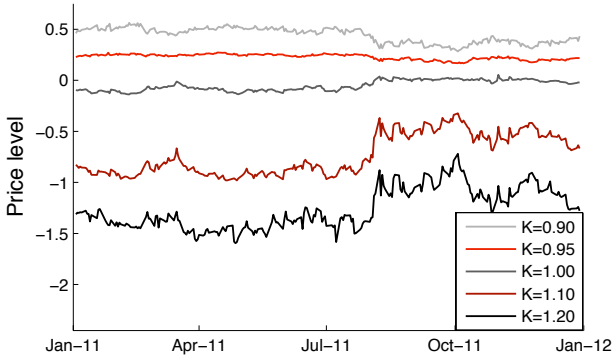


Figure 3.1: Price level calculated daily with $T_l = 3/12$ for different reference strikes K_{n^*}.

the price level and local implied volatility in [Sch08a], where the parameterizations are given as

$$x_n^l(t) = \frac{\phi\big(\Phi^{-1}\big(-\partial_K C_n^l(t)\big)\big)}{\sqrt{T_l - t}\, K_n \partial_{KK} C_n^l(t)}, \tag{3.4}$$

$$y(t) = \sqrt{T_l - t}\, \Phi^{-1}\big(-\partial_K C_{n^*}^l(t)\big), \tag{3.5}$$

with $\phi = \Phi'$. Using the BS pricing function (1.1) to find the derivatives

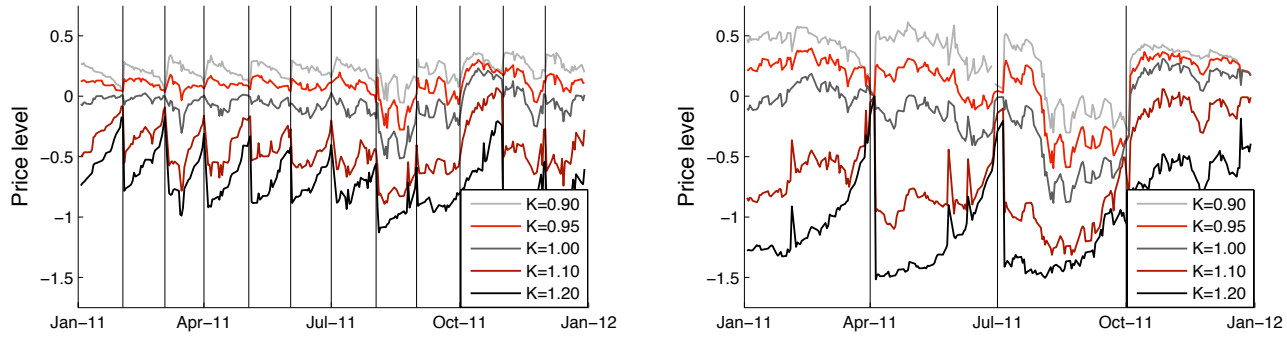
$$\partial_K C_t^{BS} = -\Phi(d^-),$$
$$\partial_{KK} C_t^{BS} = \frac{\phi(d^-)}{K\sigma\sqrt{T-t}},$$

we can rewrite (3.4) and (3.5) to

$$x_n^l(t) = \sigma,$$
$$y(t) = \sqrt{T-t}\, d^- = \sigma^{-1}\log(X_t/K_{n^*}) - \tau\sigma/2. \tag{3.6}$$

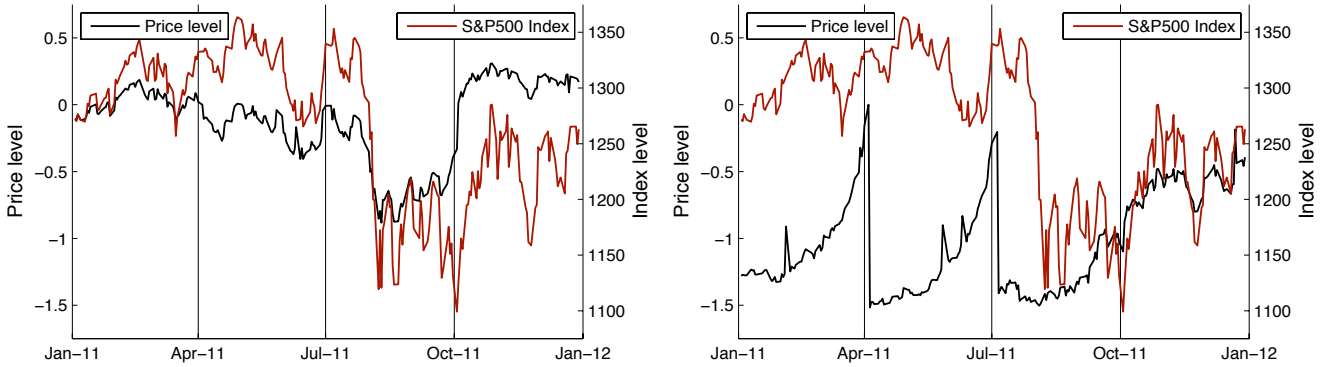
We have the freedom to choose any reference strike $K_{n^*} \in \mathcal{K}$. Note that y is a direct parameterization of the skew of the call price $C_{N^*}^l$ via the first derivative of the call price with respect to the strike. By modeling y we in fact model the call price skew around the reference strike K_{n^*}. It is interesting to see how y reveals information on the call price surface of T_l.

In the summer of 2011 equity markets crashed and consequently short term implied volatilities increased tremendously. High volatility means high call prices, lowering the convexity of the price with respect to the strike. Thinking of the option's extrinsic value, i.e. $C_t - (X_t - K_{n^*})^+$, should increase massively for K_{n^*} close to at-the-money (K_{n^*}=1.00). Consequently the derivatives $\partial_K C_n^l$ for K_n away from at-the-money (ATM) change: it decreases for OTM strikes and increases for ITM strikes. This effect can be seen in Figure 3.1. We see that the quantities stay relatively stable around ATM compared to the more extreme strikes. This effect is worse for $T_l = 1/12$.
The path taken by the price level reveals valuable information as well. Observe in Figure 3.2, where we can see the dynamics of the price level for different reference strikes with T_l (a) reset every month to one month and (b) reset every three months to three months. We see that the empirical behavior is identical to what we expect from the BS framework. Taking (b) in particular we see that the second term in (3.6) adds up to decreasing price levels in the third quarter: as the market goes down, negative implied volatilities go down as well. The higher the strike,



(a) $T_l = 1/12$ (b) $T_l = 3/12$

Figure 3.2: Price level for different reference strikes K_{n^*} with T_l resetting every (a) 1 month and (b) 3 months.



(a) $K_{n^*} = 1.00$ (b) $K_{n^*} = 1.20$

Figure 3.3: Price level with $T_l = 3/12$ compared to the S&P500 Index reset every quarter for different reference strikes K_{n^*}.

the less pronounced the effect due to the decreasing convexity discussed above. Interesting is the convergence to zero close T_l that takes place, as this is what (3.6) suggests.

From (3.6) the X term and negative implied volatility we expect to see a strong correlation between the price level and the S&P500 Index. Figure 3.3 shows that this is indeed the case, where the effect is stronger in (a) for $K_{n^*} = 1.00$. For an ATM reference strike the quantity is close to zero already and so the convergence to zero has less influence, increasing the correlation.

The local implied volatility x_n^m for $m > l$ in (3.3) are obtained via the discretization of the derivatives

$$\partial_T C_t = \frac{C_n^m(t) - C_n^{m-1}(t)}{T_m - T_{m-1}},$$
$$\partial_{KK} C_t = \left(\frac{C_{n+1}^m(t) - C_n^m(t)}{K_{n+1} - K_n} - \frac{C_n^m(t) - C_{n-1}^m(t)}{K_n - K_{n-1}} \right) \Bigg/ \left(\frac{K_{n+1} - K_n}{2} - \frac{K_n + K_{n-1}}{2} \right).$$

In Figure 3.4 we see that local implied volatilities show similar features compared to implied volatilities. This is expected for x^l, but also holds for $(x_1^m, \ldots, x_N^m)$ with $m = l+1, \ldots, M$. (a) shows the monthly paths of local implied volatility going from $T_l = 1/12$ to $T_l = 0$. We know that implied volatility blows up and is more volatile close to

maturity, exactly what we observe here as well. (b) shows the monthly path of the local implied volatility going from $T_m = 6/12$ to $T_m = 5/12$. These paths are not influenced by problems close to maturity, but more by the overall level of the local volatility surface. It is therefore not surprising that it is less volatile and shows strong negative correlation with the underlying. The reason why x has a different parameterization for T_l and T_m with $m > l$ is because the model will now allow for constant local implied volatility dynamics which we will consider in section 5.1.

We will specify dynamics for call prices parameterized in x and y. Therefore, we need the map

$$\begin{aligned}(x, y) : \{&\text{admissible } (C_n^m)_{n=0,\dots,N, m=l,\dots,M} \text{ on } [T_{l-1}, T_l)\} \\ &\mapsto \{\text{positive } (X_n^m)_{n=1,\dots,N, m=1,\dots,M} \text{ on } [T_{l-1}, T_l)\} \times \{\text{real-valued } y \text{ on } [T_{l-1}, T_l)\}\end{aligned}$$

to be a bijection for each $l \in \{1, \dots, M\}$. Wissel shows this by defining

$$\begin{aligned}\zeta_n^m &:= \frac{K_{n+1} - K_n}{K_{n+1} - K_{n-1}} \frac{K_n^2 (T_m - T_{m-1})}{(K_{n+1} - K_n)(K_n - K_{n-1})}, \\ \eta_n^m &:= \frac{K_n - K_{n-1}}{K_{n+1} - K_{n-1}} \frac{K_n^2 (T_m - T_{m-1})}{(K_{n+1} - K_n)(K_n - K_{n-1})}, \\ \chi_n^m &:= (x_n^m)^2,\end{aligned}$$

for every $m = 2, \dots, M$ and $n = 1, \dots, N$. Let I_N be the N-dimensional identity matrix. The tridiagonal $(N \times N)$ matrices

$$A_m := I_N + \begin{bmatrix} \chi_1^m(\zeta_1^m + \eta_1^m) & -\chi_1^m \eta_1^m & & & \\ -\chi_2^m \zeta_2^m & \chi_2^m(\zeta_2^m + \eta_2^m) & -\chi_2^m \eta_2^m & & \\ & \ddots & \ddots & \ddots & \\ & & -\chi_{N-1}^m \zeta_{N-1}^m & \chi_{N-1}^m(\zeta_{N-1}^m + \eta_{N-1}^m) & -\chi_{N-1}^m \eta_{N-1}^m \\ & & & -\chi_N^m \zeta_N^m & \chi_N^m(\zeta_N^m + \eta_N^m) \end{bmatrix}, \tag{3.7}$$

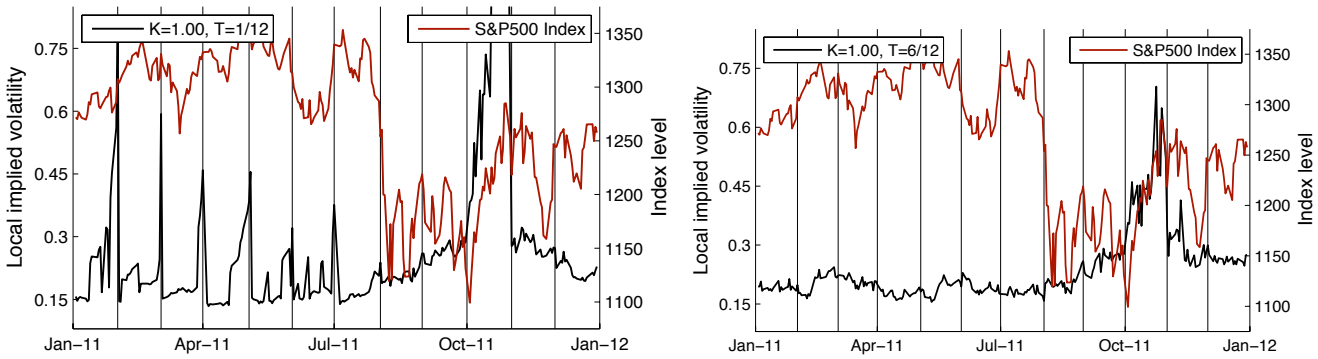
for $m = 2, \dots, M$. A_m are shown to be invertible with all entries in the A_m^{-1} strictly positive. Let C^m denote the column vector $(C_1^m, \dots, C_N^m)$ and e the unit column vector $(1, 0, \dots, 0) \in \mathbb{R}^N$. The following theorem captures the bijection and gives an explicit inverse to translate the code-book to option prices. The theorem consists of Theorem 5.4 in [Wis08] for an arbitrary reference strike by using [Sch08a].

Theorem 3.1.

(i) Let $(x_n^m)_{n=1,\dots,N, m=1,\dots,M}$ be local implied volatities and y the price level of an admissible option price model. Then for each $l = 1, \dots, M$ the option prices $C_n^m(t), m = l, \dots, M$, for $t \in [T_{l-1}, T_l)$ are given by

$$\begin{aligned} C_n^l(t) &= \sum_{j=n^*}^{n} \Phi\left(\frac{y(t) + \sum_{i=n^*}^{j+1} (K_{i+1} - K_i) / \big(K_i \, x_i^l(t)\big)}{\sqrt{T_l - t}} \right) \cdot (K_{j+1} - K_j), \quad n = 0, \dots, N, \\ C^m(t) &= A_m^{-1}(t)(C^{m-1}(t) + X_t \chi_1^m(t) \zeta_1^m e), \qquad\qquad m = l+1, \dots, M. \end{aligned} \tag{3.8}$$

(ii) Conversely, for positive processes x_n^m on $[0, T_m)$ with $n = 1, \dots, N$ and $m = 1, \dots, M$ and a real-valued process y on $[0, T_M)$, define for each $l = 1, \dots, M$ option prices $C_n^m(t)$, $m = l, \dots, M$ for $t \in [T_{l-1}, T_l)$ and all n recursively via (3.8). For $l = 1, \dots, M-1$, set $C_n^l(T_l) := (X_{T_m} - K_n)^+$ for all n. Then the model $(C_n^m)_{n=0,\dots,N, m=1,\dots,M}$ is an admissible option price model with local implied volatilities x_n^m and price level y.



(a) ATM 1M LIV

(b) ATM 6M LIV

Figure 3.4: Local implied volatility compared to the S&P500 Index reset every month for $K_n = 1.00$ and different T_m.

The summation in here and in the following theorems sum up negative terms when number to which we sum stands below the reference strike. This theorem shows the convenience of choosing local implied volatility and price level as a code-book. The option model can be admissible without constraining the code-book state space, which is $(0,\infty)^{N(M-l+1)} \times \mathbb{R}$ for $t \in [T_{l-1}, T_l)$. This stands in contrast to the implied volatility approaches, where the state space of implied volatility σ_n^m generally depends on strike, maturity, interest rates and dividends.

3.2 Absence of dynamic arbitrage

Having defined a bijection between the code-book and call prices, we are ready to specify dynamics for the code-book in terms of SDEs. We change notation of a Brownian motion from W_t to $W(t)$ to keep the notation for all stochastic processes consistent. We specify the dynamics according to

$$\frac{dx_n^m(t)}{x_n^m(t)} = u_n^m(t)dt + v_n^m(t)dW(t), \quad m = 1,\dots,M,\ n = 1,\dots,N, \tag{3.9}$$

$$dy(t) = \mu(t)dt + \xi(t)dW(t), \tag{3.10}$$

with $u_n^m, \mu \in \mathcal{L}^1_{loc}(\mathbb{R})$ and $v_n^m, \xi \in \mathcal{L}^2_{loc}(\mathbb{R}^D)$. The vector $x^m := (x_1^m,\dots,x_N^m)$ is expected to make jumps and possibly change of dynamics going from $t \in [0, T_{m-1})$ to $t \in [T_{m-1}, T_m)$ due to the different parameterization, whereas y jumps every maturity T_l.

Define for each $l = 1,\dots,M$ the processes $\lambda_n^m = (\lambda_n^m(t))_{t\in[T_{l-1},T_l)} \in \mathcal{L}^2_{loc}(\mathbb{R}^D)$, where $m = l,\dots,M$ and $n = 0,\dots,N$, according to

$$\lambda_n^l(t) := \sum_{j=n}^{N} \phi\left(\frac{y(t) + \sum_{i=n^*}^{j+1} \frac{K_{i+1}-K_i}{K_i x_i^l(t)}}{\sqrt{T_l - t}}\right) \frac{1}{\sqrt{T_l - t}} \left(\xi(t) - \sum_{i=n^*}^{j+1} \frac{K_{i+1}-K_i}{K_i x_i^l(t)}\, v_i^l(t)\right)(K_{j+1} - K_j). \tag{3.11}$$

Recursively for $m > l$, define $\lambda_0^m(t) := \lambda_0^{m-1}(t)$ and

$$\lambda_n^m(t) := A_{n,1}^{-1}(t)\zeta_1^m \chi_1^m(t)\lambda_0^{m-1}(t) + \sum_{j=1}^{N} A_{n,j}^{-1}(t)\left(\lambda_j^{m-1}(t) + 2v_j^m(t)\left(C_j^m(t) - C_j^{m-1}(t)\right)\right). \tag{3.12}$$

Wissel proves that the model is free of dynamic arbitrage, i.e. a pricing measure $\mathbb{P}^*$ on $\mathcal{F}$ exists, if and only if

- ◇ a market price of risk process $b \in \mathcal{L}^2_{loc}(\mathbb{R}^D)$ exists such that

$$\frac{d\mathbb{P}^*}{d\mathbb{P}} = \mathcal{E}\left(\int_0^{T_M} b(t)dW(t)\right),$$

  where $\mathcal{E}$ denotes the Doléans-Dade exponential defined in section 2.1,

- ◇ the drift coefficients μ and u satisfy $\mathbb{P}^*$-a.s.

$$\mu(t) = \frac{1}{2}\frac{y(t)}{T_l - t}\left(|\xi(t)|^2 - 1\right), \tag{3.13}$$

$$u_n^l(t) = |v_n^l(t)|^2 + \frac{1}{T_l - t}\left[\frac{1}{2} - \frac{1}{2}\left|\xi(t) - \sum_{i=n^*}^{n}\frac{K_{i+1} - K_i}{K_i x_i^l(t)}v_i^l(t)\right|^2 \left(y(t) + \sum_{i=n^*}^{n+1}\frac{K_{i+1} - K_i}{K_i x_i^l(t)}\right)\left(\xi(t) - \frac{1}{2}\frac{K_{n+1} - K_n}{K_n\, x_n^l(t)}v_n^l(t) - \sum_{i=n^*}^{n+1}\frac{K_{i+1} - K_i}{K_i\, x_i^l(t)}v_i^l(t)\right)v_n^l(t)\right], \tag{3.14}$$

$$u_n^m(t) = \left(-\frac{\lambda_n^m(t) - \lambda_n^{m-1}(t)}{C_n^m(t) - C_n^{m-1}(t)} + \frac{3}{2}v_n^m(t)\right)v_n^m(t), \qquad m > l, \tag{3.15}$$

  for every $t \in [T_{l-1}, T_l)$ and each $l = 1, \ldots, M$,

- ◇ the processes $C_n^m(\cdot)$ are $\mathbb{P}^*$-a.s. continuous at each T_l for each $n = 0, \ldots, N$ and $m = 1, \ldots, M$.

Wissel shows that consequently that the risk-neutral dynamics of the stock and option prices are given by

$$dC_n^m(t) = \lambda_n^m(t)dW^*(t),$$

for every $t \in [0, T_m)$ and each $n = 0, \ldots, N$ and $m = 1, \ldots, M$, where W^* denotes a D-dimensional Brownian motion under risk-neutral measure $\mathbb{P}^*$. Girsanov's theorem tells us that $W^*(t) = W(t) - \int_0^t b(t)dt$, and thus the real-world dynamics can be modeled by additionally specifying a market price of risk process $b(t)$. The theorem consists of Theorem 5.11 in [Wis08] for an arbitrary reference strike by using [Sch08a] and tells us the restrictions that the Gaussian parameter have to adhere to in order for the SDEs to have a unique solution.

Theorem 3.2.

Fix $l \in \{1, \ldots, M\}$. Suppose that ξ is locally Lipschitz in y, that for each $m = l, \ldots, M$ the functions v_n^m are locally Lipschitz in x^m and that

$$\left||\xi(t,y)|^2 - 1\right| \leq M_1(T_l - t), \tag{3.16}$$

$$\left|v_n^l(t,y,x^l)\right| \leq M_2(T_l - t)\, x_n^l\left(1 + \left|y + \sum_{i=n^*}^{n+1}\frac{K_{i+1} - K_i}{K_i x_i^l}\right|\right)^{-1} \tag{3.17}$$

$$\left|v_n^{l+1}(t,y,x^l,x^{l+1})\right| \leq M_3\left|\frac{C_n^{l+1}(t,y,x^l,x^{l+1}) - C_n^l(t,y,x^l)}{\lambda_n^{l+1}(t,y,x^l,x^{l+1}) - \lambda_n^l(t,y,x^l)}\right| \tag{3.18}$$

for $t \in [T_{l-1}, T_l]$. Then for any $\mathcal{F}_{T_{l-1}}$-measurable initial condition of positive random variables $x_n^m(T_{l-1})$ and real-valued $y(T_{l-1})$, the SDEs (3.9) and (3.10) with inverses (3.8) and drift coefficients (3.11)-(3.15) have a unique solution on the closed interval $[T_{l-1}, T_l]$ consisting of positive processes x_n^m, $n = 1, \ldots, N$, $m = l, \ldots, M$ and a real-valued process y.

3.3 Examples

With Theorem 3.2 we have an explicit existence result of the option model (C_n^m). With the assumptions in this theorem we are able to give explicit examples of the model in terms of the input parameters volatilities of the local implied volatility v_n^m and price level volatility ξ. We start with generalizing to functions that satisfy the conditions (3.16) -(3.18). To this extent we write the price level's volatility function as

$$\xi(t,y) = (1 + (T_l - t)g(t,y), 0, \dots, 0) \in \mathbb{R}^D, \tag{3.19}$$

where g is bounded and locally Lipschitz continuous. The vertical translation of size 1 comes in to avoid problems where $t \approx T_l$ in (3.16). If we let g include a factor T_l^{-1} we can control the influence of ξ over the interval $[T_{l-1}, T_l)$. Note that we can assume this form of ξ without loss of generality, since Brownian motions are invariant under orthogonal transformations. Generalize the volvols by invoking a function V depending on time to maturity and moneyness by defining for each $l = 1, \dots, M$

$$\begin{aligned}
v_n^l(t,y,x^l) &= V\left(T_l - t, \frac{K_n}{S(t,y,x^l)}\right)\frac{x_n^l}{1+x_n^l}\,\eta_n^l,\\
v_n^{l+1}(t,y,x^l,x^{l+1}) &= V\left(T_{l+1} - t, \frac{K_n}{S(t,y,x^l)}\right)\eta_n^{l+1},\\
v_n^m(t,y,x^l,\dots,x^m) &= V\left(T_m - t, \frac{K_n}{S(t,y,x^l)}\right)\eta_n^m \quad \text{for } m > l+1,
\end{aligned}$$

where

$$\eta_n^m := \begin{cases} \left(1 + \left| y + \sum_{i=n+1}^N \frac{K_{i+1}-K_i}{K_i\ x_i^l}\right|\right)^{-1} & \text{for } m = l,\\ \left|\frac{C_n^{l+1}(t,y,x^l,x^{l+1}) - C_n^l(t,y,x^l)}{\lambda_n^{l+1}(t,y,x^l,x^{l+1}) - \lambda_n^l(t,y,x^l)}\right| & \text{for } m = l+1,\\ 1 & \text{for } m > l+1. \end{cases} \tag{3.20}$$

The invoked fraction $\frac{x_n^l}{1+x_n^l}$ in v_n^l turns out to be convenient for calibration later on. Note that we need V to be linearly decreasing in time to maturity and bounded by $M_2 \wedge M_3$.

4 Implementation

We will look at the problems encountered during implementation of this model. We start with discussing the data after which we show how we can make the local implied volatilities numerically well-defined. Having them well-defined, we can then look at the way to simulate SDEs.

4.1 Data

The data we will take in consideration for calibration regards the call option prices on the S&P500 Index. This index consists out of 500 shares from a wide range of industries in the U.S. such as energy, financial services and information technology, capturing around 75% of the total U.S. traded equity. The index level is a capitalization-weighted average of the stocks, meaning that the stocks are weighted according to the total market value of the outstanding tradable shares. We will consider the call option prices given over the year 2011. On every date we have quotes on around 12 different maturities reaching from days to 2 years, each with a large number of different strikes. Note that the quotes are observed as implied volatilities $\widehat{\sigma}_n^m$ rather than actual call prices $\widehat{C}_n^m$. Additionally, we have the S&P500 estimated dividend yield d p.a. and annualized interest rate r for different maturities. We apply a cubic spline to get the full yield curve.
Note that in the case of an index all cash dividend $\alpha_k = 0$, since an index is assumed to have a dividend yield rather than cash dividends. This directly gives $\mathcal{A}_T = 0$ and we include the dividend yield d_k by assuming that proportional dividend is paid continuously, such that the future price (2.7) becomes

$$F_t(T) := S_t \cdot R_t(T) = S_t \exp\left(\int_t^T r_u - d_u \, du\right).$$

4.2 Arbitrage-free smoothing of the price surface

Definition 3.1 shows that the local implied volatility is depending on quotes on previous maturities and adjacent strikes. Let $\mathcal{K}^m$ be the set of strikes having quotes for maturity T_m. The local implied volatilities x will be well-defined as long as the $\mathcal{K}^1 \supseteq \ldots \supseteq \mathcal{K}^M$. Market data is usually sparse on the strike-maturity grid and does not show the inclusions of the $\mathcal{K}^m$. In fact, market data tends to have the opposite structure: for large maturities the quotes will be on a wide range of strikes, whereas close maturities have quotes on strikes close to at-the-money. Usually these strikes overlap with the strikes of the larger maturities. We will have to interpolate or smooth the discrete market price data such that surface remains free of static arbitrage. Estimating the implied volatility surface is not straight forward, e.g. simple interpolation or smoothing of the implied volatility quotes does not respect the constraints of static arbitrage. The estimation does not only depend on the implied volatility, but also interest rate and dividends. It is therefore more convenient to estimate in the call price space, in which absence of static arbitrage translates into a clear set of constraints as in Definition 2.1. Two significant papers by Kahalé [Kah04] and Fengler [Fen09] tackle the estimation in the price surface respecting continuity of the first and second derivative to K, easing the use of local volatility. Both methods generate a continuous second derivative of the call price with respect to the strike, on which local volatility highly depends. Both methods are implementation friendly and computationally light. We choose to pursue Fengler's approach for two reasons. First, a butterfly spread on an equidistant strike grid will have a non-negative payoff. Since we have to choose K_{N+1} sufficiently high to assure $C_{N+1}^m = 0$ $\mathbb{P}$-a.s., the butterfly spread

$$C_{N-1}^m - 2C_N^m + C_{N+1}^m$$

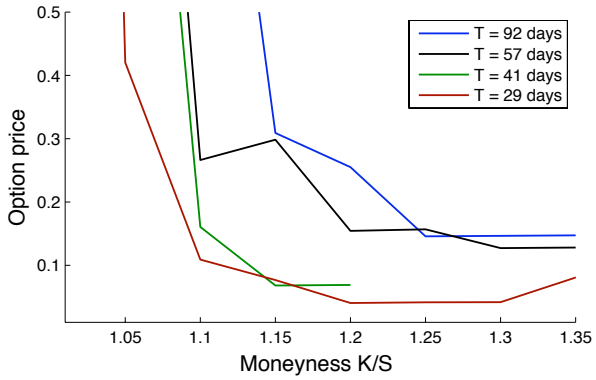


Figure 4.1: Linearly interpolated call option prices on $18-02-11$

is likely to entail a negative payoff at $t = T_m$ with a maximum of $\left|K_{N+1} - |K_N - K_{N-1}|\right|$ for some $m \in \{1, \dots, M\}$, resulting in $x_N^m \in \mathbb{C}$. In Fengler's method we can make sure this spread is priced positive.
Secondly, the input data doesn't need to be free of static arbitrage, since the employed method is based on smoothing rather than on interpolation. This is very convenient, as arbitrage is observed frequently (in particular) for close maturities. In Figure 4.1 we translated the observed quotes $\widehat{\sigma}_n^m$ to $\widehat{C}_n^m$ on February 18th of 2011, interpolating linearly between the quoted prices. The figure shows clear violations of the arbitrage constraints: The option with 29 days left to maturity becomes increasing for high moneyness and the option prices with 57 and 92 days left to maturity are not convex in the strike and are not increasing in maturity. Although these violations are outnumbered by transaction costs in practice, they will ruin the local implied volatilities on the arbitrage-infected maturity and all maturities thereafter.

To explain the algorithm, we start with defining a cubic spline.

Definition 4.1.
Let $C(k)$ denote the call price for strike k for some arbitrary maturity T. We call C a cubic spline if it is a cubic polynomial on every subinterval $(K_0, K_1), \dots, (K_N, K_{N+1})$ and $C \in \mathcal{C}^2([K_0, K_{N+1}])$, the class of twice differentiable functions. C thus admits the following representation

$$\begin{aligned} C(k) &:= \sum_{n=0}^{N} \mathbf{1}_{\{[K_n, K_{n+1})\}} s_n(k), \\ s_n(k) &:= d_n(k - k_n)^3 + c_n(u - u_n)^2 + b_n(u - u_n) + a_n, \end{aligned} \tag{4.1}$$

for given constants a_n, b_n, c_n, d_n.
Moreover, C is called a natural cubic spline if in addition the second order derivatives in the first and last subinterval are zero, i.e.

$$c_0 = d_0 = c_N = d_N = 0$$

For any fixed maturity T_m with $m = 1, \dots, M$, denote $C_n^m := C(K_n)$ and $\gamma_n := \partial_{kk} C_n^m$ for $n = 1, \dots, N$. Note that $\gamma_1^m = \gamma_N^m = 0$ for all m by definition of the natural cubic spline. Furthermore, let

$$\begin{aligned} \boldsymbol{C} &:= (C_1^m, \dots, C_N^m)^T, \\ \boldsymbol{\gamma} &:= (\gamma_2^m, \dots, \gamma_{N-1}^m)^T. \end{aligned}$$

Let $h_n := K_{n+1} - K_n$. Slightly correcting Fengler on notation, construct the $N \times (N-2)$-matrix Q containing only zeros except for

$$Q_{n-1,n} = h_{n-1}^{-1}, \quad Q_{n,n} = -h_{n-1}^{-1} - h_n^{-1} \quad Q_{n+1,n} = h_n^{-1},$$

where $n = 2, \ldots, N-1$. Furthermore, let R be a symmetric $(N-2) \times (N-2)$-matrix containing only zeros except for

$$\begin{aligned} R_{n,n} &= \frac{1}{3}(h_{n-1} + h_n) \quad \text{for } n = 2, \ldots, N-1, \\ R_{n,n+1} = R_{n+1,n} &= \frac{1}{6} h_n \qquad\qquad \text{for } n = 2, \ldots, N-2. \end{aligned}$$

Then define the matrices

$$A := \begin{bmatrix} Q \\ -R^T \end{bmatrix}, \quad B := \begin{bmatrix} I_N & 0 \\ 0 & \lambda R \end{bmatrix}.$$

Finally, define $\boldsymbol{y} := (C_1^m, \ldots, C_N^m, 0, \ldots, 0)$ as a $(2N-2)$-vector with the observed call prices C_n^m on strike K_n, maturity T_m and $\boldsymbol{x} := (\boldsymbol{C}^T, \boldsymbol{\gamma}^T)^T$ as a $(2N-2)$-vector. Theorem 2.1 in Green & Silverman [Gre93] shows that $\boldsymbol{x}$ defines a cubic spline if and only if $A^T \boldsymbol{x} = 0$. Using this result, Fengler shows that we obtain an arbitrage-free surface by the algorithm

(i) Pre-smooth the implied volatility surface by estimation on a regular moneyness grid $A = [K_1, \ldots, K_N] \times [T_1, \ldots, T_M]$.

(ii) From the last to the first maturity, iterate backwards by:

- ◇ For $m = M$, solve

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \ \frac{1}{2} \boldsymbol{x}^T B \boldsymbol{x} - \boldsymbol{y}^T \boldsymbol{x},$$

subject to the constraints

$$\begin{aligned} A^T \boldsymbol{x} &= 0 & \gamma_n^m &\geq 0, \quad n = 1, \ldots, N \\ C_2^m - C_1^m &\geq K_2 - K_1, & C_{N-1}^m - C_N^m &\geq 0, \\ C_1^m &\geq X_0 - K_1, & C_N^m &\geq 0, \\ (*) \quad C_1^m &\leq X_0, & C_{N-1}^m - 2C_N^m + C_{N+1}^m &\geq 0. \end{aligned} \tag{4.2}$$

- ◇ For $m = M-1, \ldots, 1$, solve the same scheme with constraint $(*)$ replaced by

$$C_n^m < C_n^{m+1}, \quad n = 1, \ldots, N.$$

In step (i) we pre-smooth on the grid A to fill up the space with quotes. As discussed above, data is sparse in contrast to the needed full grid for the algorithm. We take a non-parametric approach applying a Nadaraya–Watson estimator to evaluate $\sigma(K_n, T_m)$ at the grid A. The Nadaraya-Watson estimator estimates volatilities according to

$$\sigma(K_n, T_m) = \frac{\sum_{i \in \mathcal{I}} \widehat{\sigma}(i) g(K_n - K_i, T_m - T_i)}{\sum_{i \in \mathcal{I}} g(K_n - K_i, T_m - T_i)},$$

where $\mathcal{I}$ denotes the set of all observed implied volatilities $\widehat{\sigma}(i)$ on strike K_i, maturity T_i and $g : \mathbb{R}^2 \mapsto \mathbb{R}$ the Gaussian kernel defined as

$$g(x, y) := \frac{1}{2\pi} \exp\left(-\frac{x^2}{2h_1}\right) \exp\left(-\frac{y^2}{2h_2}\right),$$

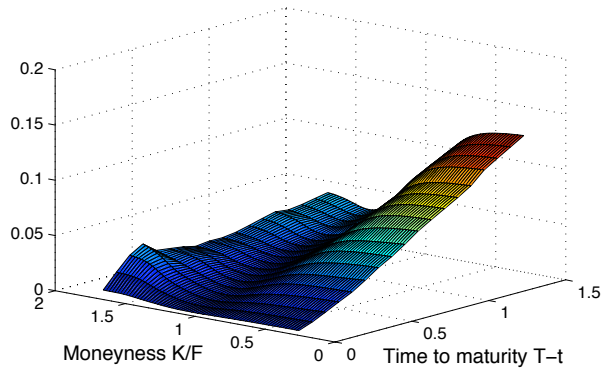


Figure 4.2: Total variance plot implied by a pre-smoothed surface.

with bandwidth parameters h_1, h_2. Small values of the bandwidth parameter lead to a bumpy surface in the corresponding dimension and large values smooth details away. The Nadaraya-Watson estimator works as a convolution of the observed data smoothed by g. The choice of the pre-smoothing method nor the choice of the bandwidth parameters h does not change the result with respect to our goal very much, as long as we reproduce the shape of an implied volatility surface. Note that we are very likely to introduce arbitrage here, since the pre-smoother is not constrained in any way. In fact, Kahalé [Kah04] argues that the BS option price is a strictly increasing function of the total implied variance $\sigma^2(K,T)T$. Absence of arbitrage becomes equivalent to the total implied variance being increasing in T. The total variance plot of the pre-smoothed surface on A of Figure 4.2 shows heavy violations in the problem region extreme strikes $\times$ small maturities. In algorithm (4.2) step (ii) we find the objective function to minimize, where the minimum defines the call price curve $K \mapsto C(K)$ for a fixed maturity. As mentioned above, the equality constraint comes from Green & Silverman in [Gre93], which state that this is a necessary and a sufficient conditions for the C and γ to define a natural cubic spline. The non-negativity constraints of the γ's comes from the call price being increasing and convex with respect to the strike. The rest of the conditions also follow from the static arbitrage constraints, where the very last condition is added to make sure the local implied volatility for $n = N$ is well defined.

From now on C_n^m will refer to smoothed, arbitrage-free call option prices.

Pricing on arbitrary strikes

Fengler wants to smooth on a regular forward moneyness grid $\left[\frac{K_1}{F_t(T)}, \ldots, \frac{K_N}{F_t(T)}\right] \times [T_1, \ldots, T_M]$. Since X is a martingale, the future price of X_t is simply X_t for any T and we immediately obtain prices on strikes $[K_1, \ldots, K_N]$. Hence, in our case we do not insist on finding the coefficients in (4.1) to find option prices on $k \in (K_n, K_{n+1})$. Suppose we run through the smoothing algorithm and want to find option prices with maturity T_m, $m = 1, \ldots, M$ and strike $k \in [0, K_{N+1}] \backslash \{K_0, \ldots, K_{N+1}\}$. This can be the case when for example it is required to switch from the forward moneyness grid $[\frac{K_1}{F_t(T_m)}, \ldots, \frac{K_N}{F_t(T_m)}]$ to moneyness grid $[K_1, \ldots, K_N]$ or when prices on a large number of strikes is needed and the algorithm only calculated the prices for a small N to save computational time. Since $C \in \mathcal{C}^2([K_0, K_{N+1}])$, we must have

$$s_{n-1}(K_n) = s_n(K_n), \quad s'_{n-1}(K_n) = s'_n(K_n), \quad s''_{n-1}(K_n) = s''_n(K_n).$$

For every $m = 1, \ldots, M$ we can solve this to get

$$\begin{aligned} a_n^m &= C_n^m, \\ b_n^m &= \frac{C_{n+1}^m - C_n^m}{h_n} - \frac{h_n}{6}(2\gamma_n^m + \gamma_{n+1}^m), \\ c_n^m &= \frac{1}{2}\gamma_n^m, \\ d_n^m &= \frac{1}{6h_n}(\gamma_{n+1}^m - \gamma_n^m), \end{aligned}$$

for $n = 1, \ldots, N-1$ and

$$\begin{aligned} a_0 = a_1 &= C_1^m, & a_N^m &= C_N^m, \\ b_0^m &= b_1^m, & b_N^m &= \frac{C_N^m - C_{N-1}^m}{h_{N-1}} + \frac{h_{N-1}\,\gamma_{N-1}^m}{6}, \\ c_0^m = d_0^m &= 0, & c_N^m = d_N^m &= 0, \end{aligned}$$

correcting Fengler on b_N^m. We can now use representation (4.1) to find the call price for any desired strike k. By put-call parity we also get arbitrage-free prices for standard European put options. Bearing this in mind, we see that it now becomes easy to price a large class of European options such straddles, strangles, butterfly spreads and call spreads. We can replicate their payoff by purchasing European calls and puts, i.e. a static hedge. The price of these options strategies is thus equivalent to the price of their replicate. The payoff and replication of these options are given below.

- ◇ Buying strangle makes the investor long volatility, since its payoff will increase as the underlying moves away from the strike (usually chosen close to at-the-money). The payoff function is given by

  $$H(S_T, K_1, K_2, T) = (K_1 - S_T)^+ + (S_T - K_2)^+,$$

  with $K_1 \leq K_2$. The strangle is replicated by being long both a call and a put maturing at T. The special case $K_1 = K_2$ is called a straddle.

- ◇ Buying butterfly spread makes the investor short volatility, since the maximum of the payoff takes place when the market doesn't move. The payoff function is given by

  $$H(S_T, K_1, K_2, K_3, T) = (S_T - (K_1))^+ \mathbf{1}_{S_T \in [K_1, K_2]} + ((K_3) - S_T)^+ \mathbf{1}_{S_T \in [K_2, K_3]},$$

  with $K_1 < K_2 < K_3$. The butterfly spread is replicated by being long one call on K_1, short two calls on K_2 and long one call on K_3 all maturing at T. Note that we can also replicate in terms of puts by being short one call on K_1, long two calls on K_2 and short one put on K_3 all maturing at T.

- ◇ Buying a call spread or capped call gives the investor upside benefits as with a normal call, but his gains are limited by cap C. This will make the option cheaper than a normal call. The payoff function is given by

  $$H(S_T, K, T, C) = \min((S_T - K)^+, C),$$

  with cap $C > 0$. A call spread is replicated by being long a call on strike K and short a call on strike $K + C$, both maturing at T.

4.3 Monte Carlo

Algorithm (4.2) gives us an arbitrage-free call price surface, which we can translate into local implied volatilities and price levels. We arrive at simulation, as we need to simulate the SDEs 3.9 and 3.10 to model their dynamics through time. The most common method for approximating SDEs is by using the Euler scheme. We start with discretizing time in steps of size $\Delta t = 1/252$, approximating one business day. For any $n = 1, \dots, N$ and $m = 1, \dots, M$ we discretize (3.9) according to

$$x_n^m((i+1)\Delta t) = x_n^m(i\Delta t) + x_n^m(i\Delta t)\big(u_n^m(i\Delta t)\Delta t + v_n^m(i\Delta t)\Delta t Z\big),$$

and (3.9) to

$$y((i+1)\Delta t) = y(i\Delta t) + \mu(i\Delta t)dt + \xi(i\Delta t)\Delta t Z,$$

where $Z \overset{d}{=} N(0,1)$. Sampling from the standard normal distribution is numerically well-understood. Note that $i\Delta t Z$ converges to a Brownian motion in the limit of $\Delta t \to 0$. The option model will work as follows:

(i) Obtain a statically arbitrage-free call price surface on $\mathcal{K} \times \mathcal{T}$ via algorithm (4.2) using the observed implied volatilities.

(ii) Determine ξ, v using calibration procedures, which we will see in section 5.

(iii) Calculate λ, μ, u via equations (3.11)-(3.12) and (3.13)-(3.15).

(iv) Extract a sample Z from the standard normal distribution.

(v) Determine $x_n^m((i+1)\Delta t)$ and $y((i+1)\Delta t)$.

(vi) Repeat (ii)-(v) until the cummulative time steps reach any desired t and retrieve the call price surface via bijection (3.8).

We call one trajectory for the index and its options a Monte Carlo (MC).

Remark 4.1.
Note that we do not explicitly need the inverse of A_m defined in (3.7). Consider equation (3.12). For every fixed $m = l+1, \dots, M$ we set up the N-dimensional vector $\Lambda^m := \big(\lambda_1^m(t), \dots, \lambda_N^m(t)\big)$. If we drop the subscript m and the dependence on t, $A \cdot \Lambda$ yields

$$A_{1,1}\lambda_1^m + A_{1,2}\lambda_2^m = \Big(A_{1,1}A_{1,1}^{-1} + A_{1,2}A_{2,1}^{-1}\Big)\Big(\zeta_1^m\chi_1^m\lambda_0^{m-1} + \lambda_1^{m-1} + 2v_1^m\big(C_1^m - C_1^{m-1}\big)\Big),$$

$$A_{n,-1}\lambda_{n-1}^m + A_{n,n}\lambda_n^m + A_{n,n+1}\lambda_n^m = \Big(A_{n-1,n}A_{n-1,n}^{-1} + A_{n,n}A_{n,n}^{-1} + A_{n,n+1}A_{n+1,n}^{-1}\Big)\Big(\lambda_n^{m-1} + 2v_n^m\big(C_n^m - C_n^{m-1}\big)\Big),$$

$$A_{N-1,N}\lambda_{N-1}^m + A_{N,N}\lambda_N^m = \Big(A_{N-1,N}A_{N-1,N}^{-1} + A_{N,N}A_{N,N}^{-1}\Big)\Big(\lambda_N^{m-1} + 2v_N^m\big(C_N^m - C_N^{m-1}\big)\Big).$$

Since the matrix coefficients add up to one, we simply get

$$A \cdot \Lambda = \Big(\zeta_1^m\chi_1^m\lambda_0^{m-1} + \lambda_1^{m-1} + 2v_1^m\big(C_1^m - C_1^{m-1}\big), \lambda_2^{m-1} + 2v_2^m\big(C_2^m - C_2^{m-1}\big), \dots, \lambda_N^{m-1} + 2v_N^m\big(C_N^m - C_N^{m-1}\big)\Big)^T.$$

This is just a system of N linear equations with N variables λ_n^m for every $m = l+1, \dots, M$, which can be solved significantly faster than finding each inverse of A_m. We avoid the calculation of A_m^{-1} in (3.8) of Theorem 3.1 by the same argument.

5 Calibration

The parameters governing the surface dynamics have to be calibrated to the observed option prices. Calibration for volatility models with parameter set Θ usually goes by finding Θ via the minimization of the sum of squared errors of the implied model volatility versus the observed implied volatility, i.e.

$$\Theta = \operatorname{argmin} \sum_{j \in \mathcal{J}} \Big(\sigma_t(K_j, T_j, \Theta) - \widehat{\sigma}_t(K_j, T_j) \Big)^2, \tag{5.1}$$

where $\sigma_t(K_j, T_j, \Theta)$ denotes the volatility implied by the option price on (K_j, T_j) found via the model. Lévy models try to calibrate using a similar approach. Here Θ is such that the characteristic function of the option price implied by the model is minimizing the sum of squared errors relative to the market data. Either way, market models can't pursue this approach, since we have $\widehat{\sigma} = \sigma$ and hence nothing to minimize. The calibration problem we face is the determination of the Gaussian components v_n^m and ξ in the SDEs defining the dynamics of local implied volatilities x_n^m and price level y.

5.1 Constant local implied volatility

The model can be simplified tremendously by considering the one-factor version with $v_n^m = 0$ for all n, m and $\xi \equiv 1$. The drift restrictions (3.13) - (3.15) become $\mu = 0$ and $u_n^m = 0$ for all n, m. The price level y now evolves according to

$$y(t) = y(T_{l-1}) + W^*(t - T_{l-1}),$$

under $\mathbb{P}^*$ and for all $t \in [T_{l-1}, T_l)$ we have

$$x_n^m(t) = x_n^m(T_{l-1}).$$

The randomness pinning down the dynamics of the underlying with its options only comes from the Brownian motion moving the price level. This gives us insight in the choice of reference strike K_{n^*} and the influence of the price level y.

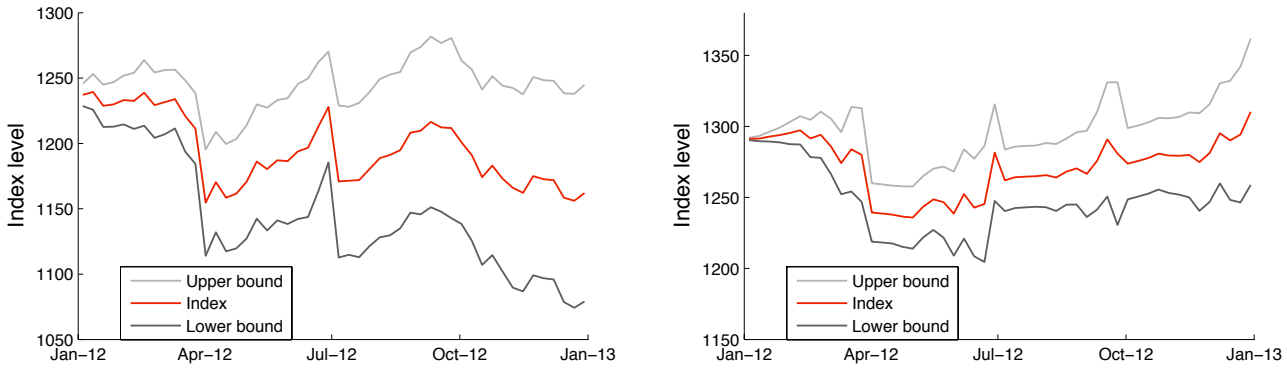
We have seen in Figure 3.1 that $y(t)$ does not show to have a drift. It is thus tempting to model y with $|\xi|^2 = 1$ to eliminate μ in (3.13), which would give

$$\operatorname{VAR}[y] = \operatorname{VAR}\left[\sum_{d=1}^{D} \xi_d \, W^*(t) \right] = \sum_{d=1}^{D} \xi_d^2(t) \operatorname{VAR}[W_d^*(t)] = |\xi(t)|^2 t = t.$$

Unfortunately, Table 5.1 below shows that the variance of one year time series of y with reference strike K_{n^*} and maturity $T_l = 3/12$ does not come close to one for any of the reference strikes.

K_{n^*}	0.90	0.95	1.00	1.10	1.20
VAR $\big[y(t)\big]$ in %	0.53	.071131	0.26	3.86	3.84
SD $\big[y(t)\big]$ in %	7.29	2.67	5.11	19.64	19.59

Table 5.1: Variance and volatility of price level $y(t)$ with $T_l = 3/12$ for different reference strikes K_{n^*}.



(a) $K_{n^*} = 1.00,\ \xi \equiv 100\%$

(b) $K_{n^*} = 1.00,\ \xi = 5.11\%$

Figure 5.1: Dynamics of the underlying implied by using constant local implied volatility with a 95% confidence interval. $\mathrm{Nm} = 100$.

Figure 5.1(a) shows the implied dynamics of the underlying using constant local implied volatility. We have calculated a 95% confidence interval around the implied index level at every time t calculated as

$$\left[\bar{S}_t - 1.96 \cdot \frac{SD\left[S_t\right]}{\sqrt{\mathrm{Nm}}}, \bar{S}_t + 1.96 \cdot \frac{SD\left[S_t\right]}{\sqrt{\mathrm{Nm}}}\right],$$

where Nm denotes the number of Monte Carlo simulations and $\mathrm{SD}\left[S_t\right]$ the standard deviation of the Nm values found for S_t. Furthermore we used a vector of three month maturities, which we see back in the change of dynamics every three months.

We face a discretization problem. The theory is built on continuous dynamics. Since we have discretized the SDEs, we are likely to introduce errors. We have $\xi = 1$ and $D = 1$, which makes the Brownian motion relatively volatile.. The samples catalyzing the jumps every three months can lead to errors when they are large. They make the price level jump significantly, which causes problems inverting matrix A?. We work around this problem by using constant local implied volatility with ξ set equal to its realised volatility, explained below, of 5.11% and let it increase linearly towards 1. The resulting process of the underlying is given in Figure 5.1(b), where the amount of MCs in which we find the discretization problem is reduced by a factor three. The jump size and overall volatility of the underlying are reduced. Another solution could have been to make the step size Δ very small around T_l, which should again reduce the amount of MCs. We do not proceed with this approach, since the influence of a change in Δ goes through the square root of Δ. We would have to decrease Δ tremendously to get the desired effect, highly increasing the computational time.

We emphasize here that we have set the reference strike to ATM, i.e. $K_{n^*} = 1.00$, since it intuitively makes the most sense. Unfortunately this results in a jump from the closing price of the S&P500 jumping from 1257.6 at the last business day of 2011 to 1292.1 after the first five business days. While this is only a 2.74% increase in one business week, we could mitigate this "starting-up" effect by setting the reference strike somewhat higher. The opposite holds for choosing the reference strike lower, which would make the initial jump smaller. The choice of a reference strike has no significant influence on the dynamics of the stock and its options, which is expected given that it is an arbitrary parameter in [Sch08a].

5.2 Realized volatility

The starting point of the calibration based on realized volatility is a time series of call prices $\big(\widehat{C}(t_i)\big)_{i\in\mathcal{I}}$ on underlying S observed on daily observation dates t_i, where we assume $\Delta t = t_i - t_{i-1}$ is constant for all i, and the underlying itself on the same observation dates t_i. We have seen before that $\widehat{C} = \big(\widehat{C}(K_j, T_j)\big)_{j\in\mathcal{J}}$ with $\mathcal{J}$ the set of all observed strike and maturity points (K_j, T_j) is likely to be sparse data. We translate these to a time series of call prices $\big(C(t_k)\big)_{k\in\mathcal{K}}$ on pure process X via (2.8), after which we smooth the pure price surface by applying Fengler's method to retrieve a full options price surface on the region $A = \big[K_1, \dots, K_N\big] \times \big[T_1, T_1 + \Delta t, T_2, T_2 + \Delta t \dots, T_M, T_M + \Delta t\big]$ consisting of fixed time to maturities and strike pairs (T_m, K_n). Recall the risk-neutral dynamics of the call price

$$dC_n^m(t) = \lambda_n^m(t)\, dW^*(t), \tag{5.2}$$

for $m = l, \dots, M$ and $n = 0, \dots, N$. In equations (3.11) and (3.12) we see that every $\lambda_n^m(t) = \lambda_n^m(t;\, \xi, v_n^m)$, i.e. λ_n^m depends directly on our choice of ξ and v_n^m. This means that we can choose these parameters such that the option price dynamics will follow the realized dynamics closely. Recall the notion of variation introduced in Definition 2.6. Note that the SDE (5.2) implies that λ can be found as the square root of the quadratic variation of the call price, i.e.

$$\lambda_n^m(t) = \sqrt{\langle C_n^m \rangle_t}.$$

We can estimate λ by the unbiased estimator

$$\widehat{\lambda}_n^m(t_0, I) := \left(\frac{252}{I}\sum_{i=1}^{I}\big(C_n^m(t_i) - C_n^m(t_{i-1})\big)^2\right)^{1/2}$$

We can thus choose our parameters $\Theta = \big(\xi, (v_n^m)_{m=l,\dots,M,\ n=0,\dots,N}\big)$ such that they minimize the squared difference between the one step realized volatility (RV) of our call price time series and λ, i.e

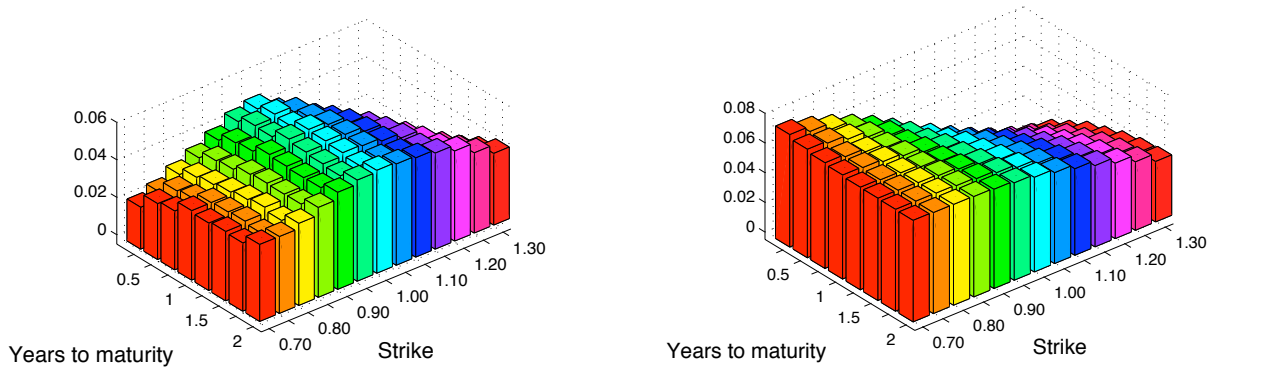
$$\Theta = \underset{\substack{n=0,\dots,N,\\ m=1,\dots,M}}{\operatorname{argmin}} \sum_{i=1}^{I}\Big(\lambda_n^m(t_{i-1};\Theta) - \widehat{\lambda}_n^m(t_{i-1}, 1)\Big)^2, \tag{5.3}$$

$$\Theta = \underset{\substack{n=0,\dots,N,\\ m=1,\dots,M}}{\operatorname{argmin}} \Big(\lambda_n^m(t_I;\Theta) - \widehat{\lambda}_n^m(t_0, I)\Big)^2. \tag{5.4}$$

Method (5.3) seems appealing, but it gets computationally very expensive as the data set grows. Method (5.4), however, can be performed relatively fast.

We would like Θ to approximate the general shape of $\widehat{\lambda}$, which can be observed in Figure 5.2 (a). We see that the RV is strictly increasing as a function of the extrinsic option value, simply because it is the variable part in the option's price.

Let $C_1, C_2 \in \mathbb{R}$ denote constants. We can uniquely minimize the function $f(x) = (x - C_1)^2$ with $x \in \mathbb{R}$, but there would be an infinite amount of minima for the function $f(x, y) = (x + y - C_1)^2$ with $x, y \in \mathbb{R}$. We could, however, uniquely minimize the functions $f_1(x) = (x - C_1)^2$ and $f_2(y) = (y - C_2)^2$ simultaneously. This example can be extended to our situation, where we have a multi-factor model with D dimensions inducing $D(MN + 1)$ variables to calibrate using $M(N + 1)$ values of $\widehat{\lambda}$. We conclude that calibration using RV will only work for D equal to one.



(a) Realized volatility $\widehat{\lambda}(t_0, I)$ over 2011

(b) Estimated realized volatility $\lambda(t_I; \Theta)$ with $|v| \leq \mathrm{LIVRealizedVol}$

Figure 5.2: (Estimated) realized volatility of the call prices.

Expectations of v

It is interesting to highlight that we are indirectly affected by a well-known issue in calibration. Close to maturity option prices tend to float away from the theory, due to possible jumps in the underlying asset. We could observe this before in the volatility surfaces of Figure 1.1, where the implied volatility in the extreme strike and close to maturity region increases massively. Prices of the corresponding options are relatively expensive, because traders want to price in a buffer to cover for unhedgeable jumps. Models incorporating jumps such as Lévy models and the Black-Scholes-Merton model this phenomenon and are hence more suitable for option pricing on short maturities. The way we are affected by this issue is by the definition of x^l, in particular for the OTM strikes. Looking at the volatility of the time series $x^l(t)$ in Figure 5.3 we see that x^l behaves more smoothly when choosing T_l larger.

Numerical results

To measure the quality of the calibration, we will have to take into account that we want $v^l \to 0$ and $\xi \to 1$ for $t \to T_l$. We take the residual norm as an error measure, i.e.

$$\mathrm{ResNorm} := \sum_{n=0}^{N} \sum_{m=1}^{M} \left(\lambda_n^m(\Theta) - \widehat{\lambda}_n^m \right)^2.$$

Since Matlab [MAT09] requires the values of λ to be given in matrix form, we have to copy λ_0 over all maturities. This means that the error induced by λ_0 is enlarged with a factor M.
Using three months maturities from now until two years ($M = 8$) and five percent strikes differences ranging from 0.70 to 1.30 ($N = 13$), we will analyze the implications of this calibration method. Without constraining the state space of ξ and v we can reduce the error to practically zero as the residual norm strikes at $4.3766 \cdot 10^{-8}\%$. Although this seems appealing, it comes at the expense of the condition that we prefer the volvols v to be between zero and their RV and ξ between its RV and 100% . We find many values of v with an absolute value exceeding 100% and a few that even exceed 1000% by far. We find ξ striking at 15.68%, compared to its RV of 13.47%

Since these values drift far away from their RVs, we invoke the constraint $|v_n^m| \leq 100\%$ for $n = 1, \ldots, N, m = 1, \ldots, M$. Simply replacing the values of v that exceed more than 100% by simply 100% gives λ_n^m different signs for different values of m, n, which will lead numerical errors in the call price dynamics. Calibration with the added

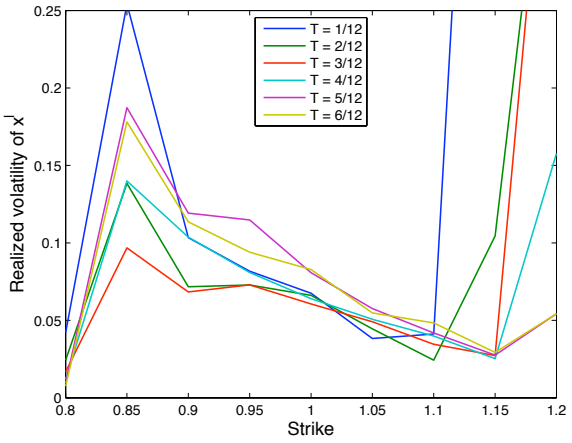


Figure 5.3: Realized volatility of x^l for different values of T_l.

constraint gives a higher residual norm of 13.55%, since the volatility of the index is now estimated at 13.53%. For ξ we find 34.45%. Alternatively trying to calibrate setting $\xi \equiv 100\%$ worsens the result with a residual norm of 30.58% and an index volatility estimate of 16.08%. With these new constraints we find the estimates for v almost all located at the boarder where $|v_n^m| = 1$. By similar arguments as for the constant local implied volatility case where $\xi \equiv 100\%$, we get numerical errors around T_l when v is found to be at the border and even worse when also $\xi \equiv 100\%$ as well. Instead, we invoke the constraint $|v_n^m| \leq \mathrm{LIVRealizedVol}\,\%$ for $n = 1, \ldots, N, m = 1, \ldots, M$. We end up with $v = -\mathrm{LIVRealizedVol}\,\%$ for most strikes and maturities and $\xi = 24.39\%$. With a residual norm of 22.65% this is not a very good result, but the error mainly comes from the fact that the index volatility is estimated at 7.94% compared to its RV 23.33% and thus adds $8 \cdot (23.33\% - 7.94\%)^2 = 18.95\%$ to the overall residual norm.

Summarized we find that the best result is obtained for the constraint $|v_n^m| \leq \mathrm{LIVRealizedVol}$, whereas ξ should be left unconstrained. Figure 5.2(b) shows the shape of λ induced by Θ under these constraints. We see that the overall fit is good, but the problem region of small strike and short maturity doesn't fit well. This can be explained by the desired fit of λ_0 defined in (3.11) to the RV of the index. The RV of the call prices is decreasing for decreasing strikes below one, while we found the RV of the index struck at 23.33% over the year 2011. This is a few factors higher than the ATM RV of the call price already and hence where this method fails. If we had persuaded an approach without matching the RV of the underlying, the overall shape and residual norm would have been a lot better. The problem with this approach is that volatility of the underlying would then be estimated close to zero, reducing the randomness its dynamics. For our purposes we are interested in the dynamics of the underlying for the sake of overall capacity of the model, whereas in practice one could choose to leave this behind when pricing forward start options or forward start variance swaps. We can then model the future price of the underlying at every modeled maturity by finding the strike for which the call price equals the put price (recall the put-call parity in section 2.2), which can be of more interest than the spot price itself.

Recall the examples that followed from the Lipschitz conditions in section 3.3. After performing the above calibration we set $g(t, y) = 1 - \xi$, ensuring condition (3.16) since $\xi(t) \to 1$ linearly as $t \to T_l$.

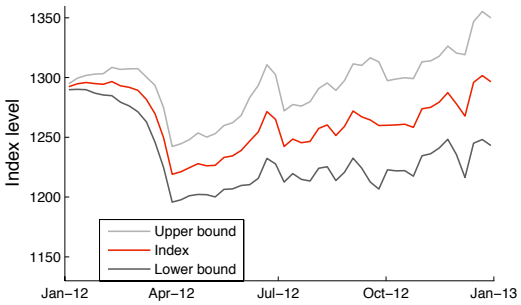


Figure 5.4: Dynamics of the underlying implied by using realized volatility with a 95% confidence interval. $\mathrm{Nm} = 100$.

Subsequently we put the volvol functions equal to

$$V\Big(T_m - t, K_n\Big) = \begin{cases} v_n^m(T_m - t) & m = l, l+1, \\ v_n^m & m > l+1. \end{cases}$$

Assuming that M_2 is large enough to ensure

$$(T_l - t)|v_n^l(t, y, x^l)| \leq M_2\, x_n^l \left(1 + \left| y + \sum_{i=n+1}^{N} \frac{K_{i+1} - K_i}{K_i x_i^l} \right| \right)$$

for all $t \in [T_{l-1}, T_l]$ makes condition (3.16) satisfied. Finally, we assume (3.18) holds.

We have seen that working in one dimension leads to the significant jumps as seen in Figure 5.2 at every three months, with numerical errors due to discretization as a result. In order to alleviate this we will try to calibrate the model in a multi-factor environment in the next section.

5.3 Principal component analysis

It is given that the market movements can not be captured within one source of randomness and so the one-factor model implied by the RV calibration is arguable. Principal component analysis (PCA) works around this and will lead to a multi-factor model.
Whereas we smoothed the call prices on the grid $A = \big[K_1, \ldots, K_N\big] \times \big[T_1, T_1 + \Delta t, T_2, T_2 + \Delta t \ldots, T_M, T_M + \Delta t\big]$ for the calibration on using RV, the grid $A = \big[K_1, \ldots, K_N\big] \times \big[T_1, T_2, \ldots, T_M\big]$ will suffice here. This arbitrage-free data can then be translated into the time series $\big(x(t_i), y(t_i)\big)_{i \in \mathcal{I}}$ on which we will perform calibration. Note that we do need arbitrage-free data, as the quantities defining the time series $\big(x(t_i), y(t_i)\big)_{i \in \mathcal{I}}$ are only defined if the data set is arbitrage-free. The time series $\big(x(t_i), y(t_i)\big)_{i \in \mathcal{I}}$ can be calculated quite fast due to Fengler's efficient algorithm discussed in section 4.2. As for time series of implied volatilities, the local implied volatilities are very stable over time. More precisely we say that the surfaces show a high level of autocorrelation, defined for time points $s > t$ as

$$\mathrm{AC}(t, s) := \mathbb{E}\big[\big(X_t - \mathbb{E}[X_t]\big)\big(X_s - \mathbb{E}[X_s]\big)\big|\mathcal{F}_t\big].$$

For such type of time series it makes sense to trust on the theory built on PCA. PCA tries to extract the uncorrelated sources of variation in a multivariate system. For our purposes we usually see a maximum of three components,

where the first component can be related to the overall level, the second as tilt and the third as curvature. The theory of PCA is most suited for time series analysis of implied volatility surfaces, but will work for time series analysis on local implied volatility surfaces as well. The dynamic properties of implied volatility time series are usually analyzed for a fixed maturity or (ATM) strike. We will calibrate across strike and moneyness using methods developed by Cont & da Fonseca in [Con02]. Cont & da Fonseca perfom a Karhunen-Loève (KL) decompostion on implied volatility surfaces across maturity and moneyness, which is a generalization of PCA to higher dimensional random fields.

Theoretical framework

Let us be given a (function of a) local implied volatility surface x. We will view this as a random surface, denoted by X. Assume $X \in \mathcal{L}^2(\mathbb{R})$. We decompose this non-zero mean surface X into

$$X(\omega, \mathbf{a}) = \mathbb{E}[X(\omega, \mathbf{a})] + Y(\omega, \mathbf{a}), \quad \mathbf{a} \in A, \tag{5.5}$$

where Y is a zero-mean random surface. Both the correlation function $R_Y(\mathbf{a}_1, \mathbf{a}_2)$ and the covariance function $C_Y(\mathbf{a}_1, \mathbf{a}_2)$ of the zero-mean random process Y are equal to the covariance function $C_X(\mathbf{a}_1, \mathbf{a}_2)$ of the non-zero mean process X, which we will all denote as simply C hereafter.
Let Ω be a Hilbert space of the random variables $Y(\omega) : \Omega \to \mathbb{R}$ on our probability space. Let $\big(E_k(\omega)\big)_{k\geq 1}$ denote the Hilbert basis of Ω. The KL theorem of a zero-mean random process such as Y tells us that Y admits a representation

$$Y(\omega, \mathbf{a}) = \sum_{k=1}^{\infty} c_k(\mathbf{a}) E_k(\omega), \tag{5.6}$$

where the E_k are orthonormal, i.e. normalized and uncorrelated, random variables. This makes the covariance function equal to

$$C(\mathbf{a}_1, \mathbf{a}_2) = \mathbb{E}\big[Y(\omega, \mathbf{a}_1) Y(\omega, \mathbf{a}_2)\big] = \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} c_j(\mathbf{a}_1) c_k(\mathbf{a}_2) \mathbb{E}\big[E_j(\omega) E_k(\omega)\big] = \sum_{k=1}^{\infty} c_k(\mathbf{a}_1) c_k(\mathbf{a}_2), \tag{5.7}$$

where we used the monotone convergence theorem in the second equality and orthonormality of the E_k in the last. Since the covariance function C is symmetric, it is Hermitian. The spectral theorem (e.g. [Dri03], Theorem 35.17) then tells us that

$$C(\mathbf{a}_1, \mathbf{a}_2) = \sum_{k=1}^{\infty} \lambda_k f_k(\mathbf{a}_1) f_k(\mathbf{a}_2), \tag{5.8}$$

where $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$ are the normalized eigenvalues and $\big(f_k(\mathbf{a})\big)_{k\geq 1}$ the orthonormal eigenfunctions corresponding to C. They are obtained as the solution to the eigenvalue problem found by the projection of (5.8) on $f_k(\mathbf{a}_1)$, i.e. by solving

$$\int_A C(\mathbf{a}_1, \mathbf{a}_2) f_k(\mathbf{a}_1) d\mathbf{a}_1 = \lambda_k f_k(\mathbf{a}_2). \tag{5.9}$$

Combining (5.7) and (5.8) we see that

$$c_k(\mathbf{a}) = \sqrt{\lambda_k} f_k(\mathbf{a}),$$

which can be used in (5.6) to get

$$Y(\omega, \mathbf{a}) = \sum_{k=1}^{\infty} \sqrt{\lambda_k} f_k(\mathbf{a}) E_k(\omega).$$

Note that Y consists of random variables E_k orthogonal in $\mathcal{L}^2(\Omega, \mathbb{P})$ and of surfaces f_k orthogonal in $\mathcal{L}^2(A)$. We call this form of Y its KL decomposition. Using (5.5), we hence get

$$X(\omega, \mathbf{a}) = \mathbb{E}[X(\omega, \mathbf{a})] + \sum_{k=1}^{\infty} \sqrt{\lambda_k} f_k(\mathbf{a}) E_k(\omega)$$

Numerical approach

In order to solve the problem numerically, we approximate the eigenfunctions f_k according to

$$f_k(\mathbf{a}) \approx \sum_{j=1}^{D} \mathbf{f}_{j,k} h_j(\mathbf{a}), \tag{5.10}$$

where $\{h_j(\mathbf{a}) : A \to \mathbb{R}\}$ is a set of shape functions such as the Nadaraya-Watson estimator which we have discussed in section 4.2. Plugging this into equation (5.9) and changing the order of integration and summation gives

$$\sum_{j=1}^{J} \mathbf{f}_{j,k} \left(\int_A C(\mathbf{a}_1, \mathbf{a}_2) h_j(\mathbf{a}_1) d\mathbf{a}_1 - \lambda_k h_j(\mathbf{a}_2) \right) = \epsilon_D,$$

where the error ϵ_D is induced by the truncated summation (5.10). The Galerkin method requires that ϵ_D stands orthogonal to the approximating function h_j, i.e. for all $j = 1, \ldots, D$ we need

$$\sum_{j=1}^{D} \mathbf{f}_{j,k} \left(\int_A \int_A C(\mathbf{a}_1, \mathbf{a}_2) h_j(\mathbf{a}_1) h_i(\mathbf{a}_2) d\mathbf{a}_1 d\mathbf{a}_2 - \lambda_k \int_A h_i(\mathbf{a}_2) h_j(\mathbf{a}_2) d\mathbf{a}_1 d\mathbf{a}_2 \right) = 0. \tag{5.11}$$

Defining the $(D \times D)$ matrices

$$\begin{aligned}
\mathbf{C}_{i,j} &= \int_A \int_A C(\mathbf{a}_1, \mathbf{a}_2) h_j(\mathbf{a}_1) h_i(\mathbf{a}_2) d\mathbf{a}_1 d\mathbf{a}_2, \\
\mathbf{H}_{i,j} &= \int_A h_i(\mathbf{a}_2) h_j(\mathbf{a}_2), \\
\mathbf{f}_{j,l} &= \mathbf{f}_{j,l}, \\
\mathbf{\Lambda}_{l,k} &= \lambda_k \mathbf{1}_{\{l=k\}},
\end{aligned}$$

equation (5.11) becomes

$$\sum_{j=1}^{D} \mathbf{C}_{i,j} \mathbf{f}_{j,k} = \sum_{j=1}^{D} \sum_{l=1}^{D} \mathbf{H}_{i,j} \mathbf{f}_{j,l} \mathbf{\Lambda}_{l,k}.$$

This is simply

$$\mathbf{Cf} = \mathbf{Hf\Lambda},$$

a D-dimensional generalized eigenvalue problem which can be solved easily numerically to find $\mathbf{f}$ and $\mathbf{\Lambda}$. We then obtain the eigenfunctions f_k for $k = 1, \ldots, D$ by substituting $\mathbf{f}$ in (5.10). Subsequently, we normalize and then rearrange the eigenvectors and eigenfunctions in a decreasing order.

Implementation

Recall the generalization of the volvols by the use of a generalizing V of section 3.3. Introduce the transformations Z of our local implied volatilities for each $l = 1, \dots, M$ by

$$Z_n^l(t) = \log\left(x_n^l(t)\right) - \left(x_n^l(t)\right)^{-1},$$
$$Z_n^m(t) = \log\left(x_n^m(t)\right) \quad \text{for } m > l.$$

By Itô's formula, the dynamics of Z are given by

$$dZ_n^m(t) = \theta_n^m dt + V\left(T_m - t, \frac{K_n}{S(t,y,x^l)}\right) dW(t), \tag{5.12}$$

with

$$\theta_n^m := \begin{cases} \frac{x_n^m+1}{x_n^m} u_n^m - \frac{1}{2}\frac{x_n^m+2}{x_n^m}(v_n^m)^2, \\ u_n^l - \frac{1}{2}(v_n^l)^2 \quad \text{for } m > l. \end{cases} \tag{5.13}$$

We can approximate $\frac{dZ}{\sqrt{dt}}$ by the defining the time series $\left(U(t_i)\right)_{i\in\mathcal{I}}$ by

$$U_n^m(t_i) := \frac{Z_n^m(t_i) - Z_n^m(t_{i-1})}{\sqrt{t_i - t_{i-1}}} \approx \frac{dZ_n^m(t_i)}{\sqrt{dt}}.$$

Using (5.12) together with the way we defined v, we get

$$\frac{dZ_n^m(t_i)}{\sqrt{dt}} = \theta_n^m\sqrt{dt} + V\left(T_m - t, \frac{K_n}{S(t,y,x^l)}\right)\frac{dW(t)}{\sqrt{dt}} = \theta_n^m\sqrt{dt} + V\left(T_m - t, \frac{K_n}{S(t,y,x^l)}\right)d\widetilde{W}(t),$$

where $\widetilde{W}(t) = \left(\widetilde{W}_1(t), \dots, \widetilde{W}_D(t)\right)$ is a time-scaled Brownian motion with $\mathbb{E}[\widetilde{W}_k] = 0$ and $\mathbb{E}[\widetilde{W}_k^2] = 1$ for all $k = 1, \dots, D$. Via the procedure described above, we estimate V_k by

$$\widehat{V}_k(\mathbf{a}) = \sqrt{\lambda_k} f_k(\mathbf{a}), \quad \mathbf{a} \in A.$$

This method will only estimate the V in the trading region A, and hence we set $V(\mathbf{a}) = 0$ for $\mathbf{a} \notin A$ to make sure the conditions on V are satisfied.

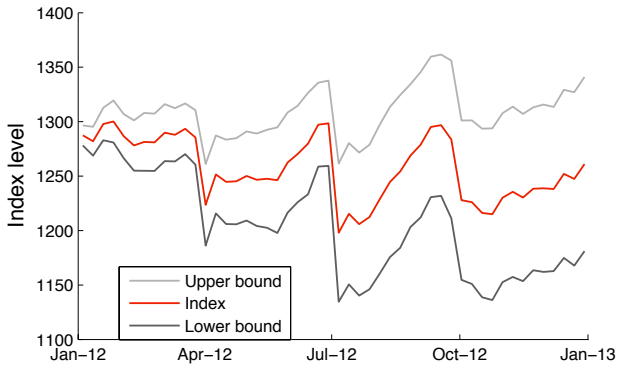
Principal component analysis can only be used when the number of principal components D is smaller than the dimensions of the observed variable. Whereas we had NM observed variables per observation date t_k before, we now only have one. This means that we are still left with determining ξ.

Numerical results

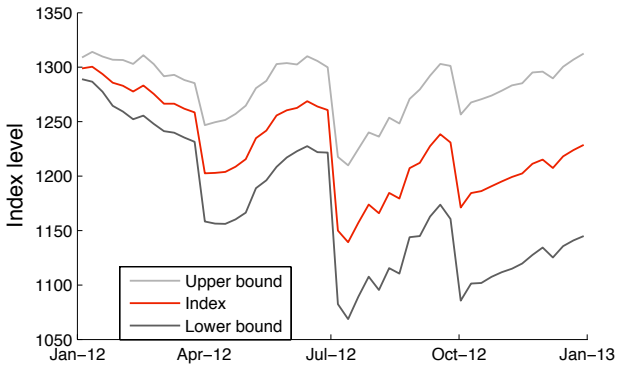
We find empirically that we can cover for 98% of the variance in the first three principal components. We find the covered variance by summing the square of the normalized eigenvalues, i.e. the number of components D is such that

$$D = \inf\left\{I : \sum_{i=1}^{I} \lambda_i^2 > 0.98\right\}.$$

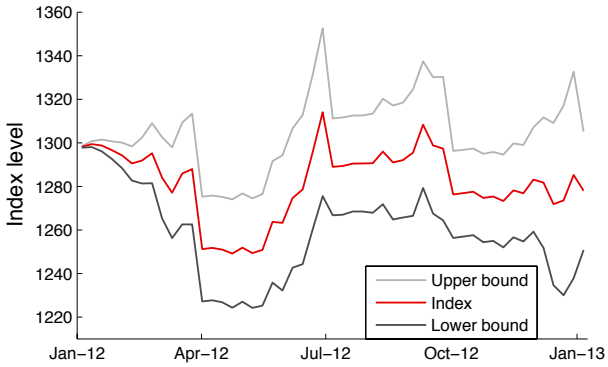
This method is computationally fast compared to calibration via realized volatility. The time needed for an analysis consisting of up to 20 principal components is a matter of seconds.



(a) $\xi = (0, 1, 0)^T$



(b) $\xi = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})^T$



(c) $\xi = (0, 0, 0)^T$

Figure 5.5: Dynamics of the underlying implied by using principal component analysis with a 95% confidence interval plotted for different ξ. $\mathrm{Nm} = 100$.

Figure 5.5 shows us the dynamics of the underlying after calibration using PCA. Since we have ξ free to choose, we let $|\xi|^2 = 1$ in (a) and (b) by setting them equal to respectively $(0, 1, 0)^T$ and $(\frac{1}{\sqrt{D}}, \frac{1}{\sqrt{D}}, \frac{1}{\sqrt{D}})^T$. The influence on the path taken is small, also when we put the weight of ξ on the first or third dimension of the Brownian motion. As expected, the number of MCs with an error is larger in (a), since all the weight is put on one Brownian motion. Setting $\xi = (0, 0, 0)^T$ in (c) such that it increases linearly to $(\frac{1}{\sqrt{D}}, \frac{1}{\sqrt{D}}, \frac{1}{\sqrt{D}})^T$as $t \to T_l$ gives very stable results, since at every T_l the Gaussian parameters v and ξ are very small. As with every path we have shown up till now, the change of dynamics has most impact at the first T_l.

6 Pricing

Modeling the call price surface as a whole adjoined with the underlying lends itself very well to pricing path dependent, as well as forward start derivatives. We will look into forward start derivatives. In particular, we will look at forward start calls and variance swaps, giving the investor exposure to forward volatility. After this we will shortly touch upon the barrier option.

6.1 Forward start options

Forward start options are options that convert a notional amount N into options with a strike set ATM of a percentage thereof on reset date T_1. The premium for this option is paid when the contract is sold at $t < T_1$, i.e. before the option actually comes into existence. Forward start options are therefore complex. First of all they are directly exposed to the term structure of volatility which makes the pricing challenging. Secondly they lack of specific greeks until the time they come into existence. There is no possibility to delta-hedge a forward start option, since the price of a forward start option does not change as the price of the underlying changes.
Let $t < T_1 < T_2$, where T_1 denotes the reset date and T_2 the option's maturity. The payoff function is given by

$$H^{FS}(S_{T_2}) = (S_{T_2} - KS_{T_1})^+.$$

Now set $t = T_0 < T_1 < \ldots < T_M$, where $T_1, \ldots, T_{M-1}$ are called the reset dates. The generalization of forward start options is given by the cliquet, which pays

$$H^{CL}(S_{T_M}, T_M) = \sum_{m=1}^{M} (S_{T_m} - KS_{T_{m-1}})^+$$

at maturity T_M. As the forward start options serve as the building blocks of a cliquets, we will focus on this first. The analytical expression in the BS framework for the price of a forward start option is found by first noting from the BS formula (1.2) that at T_1 the price can be written as

$$C_{T_1}^{BSFS}(S_t, K, T_1, T_2, r, q, \sigma) = \frac{S_{T_1}}{S_t} C_t^{BS}(S_t, K, T_2 - T_1, r, d, \sigma)$$

We then get time t value

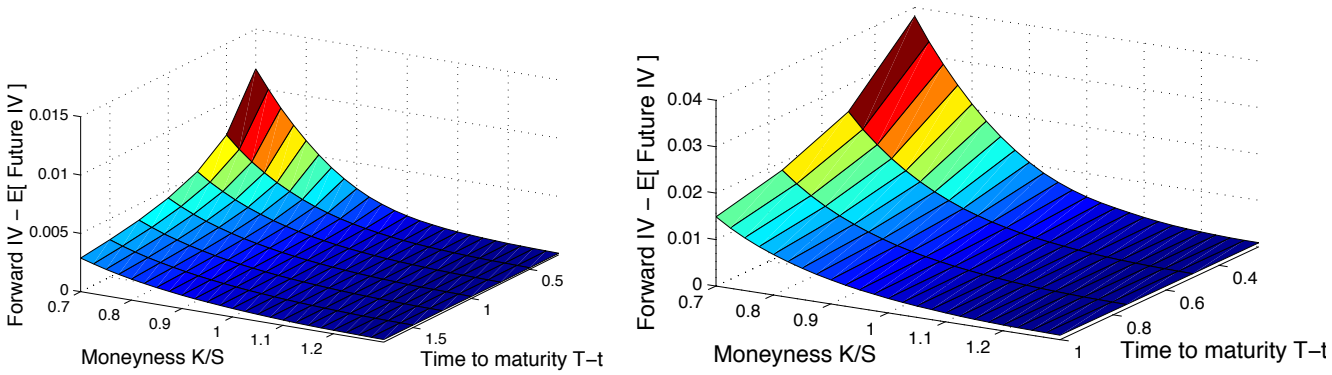
$$\begin{aligned} C_t^{BSFS}(S_t, K, T_1, T_2, r, q, \sigma) &= B_t(T_1) \cdot \mathbb{E}^* \left[\frac{S_{T_1}}{S_t} C_t^{BS}(S_t, K, T_2 - T_1, r, d, \sigma) \middle| \mathcal{F}_t \right] \\ &= B_t(T_1) \cdot \mathbb{E}^* \left[\frac{S_{T_1}}{S_t} \middle| \mathcal{F}_t \right] \cdot C_t^{BS}(S_t, K, T_2 - T_1, r, d, \sigma) \\ &= \exp\left(\int_t^{T_1} -d_u du \right) \cdot C_t^{BS}(S_t, K, T_2 - T_1, r, d, \sigma). \end{aligned} \tag{6.1}$$

The volatility σ to use here is the implied volatility corresponding to maturity $T_2 - T_1$. As [Ber07] mentions, the price of a forward start option must be very accurate, since at the reset date T_1 the option becomes a liquid vanilla. Our market model should thus be able to capture the forward implied volatilities, in particular those of $T_2 - T_1$.

Forward and expected future implied volatility

The common way to quote the price of forward start options is in terms of forward implied volatility. The forward implied volatility is the $\widehat{\sigma}$ such that the BS forward start price matches the observed forward start price $\widehat{C}^{FS}$, i.e.

$$\widehat{C}_t^{FS}(T_1, T_2, K) = C_t^{BSFS}\big(S_t, K, T_1, T_2, r, q, \widehat{\sigma}(T_2 - T_1, K)\big).$$



(a) FS = 3M

(b) FS = 12M

Figure 6.1: Forward implied volatility compared to the expected implied volatility obtained by the model.

The market model approach now shows its additional benefit, as it implies an expected future implied volatility surface. We can see this as the implied volatility that belongs to the expected call price at future time T, i.e. we average the call prices found at T over all taken Monte Carlos. Inverting this then gives $\mathbb{E}_t[\widehat{\sigma}_{T_1}(K, T_2)]$. For the model to be consistent we should obtain the difference between the forward implied volatility and expected future implied volatility converges to zero as $t \to T_1$. We would like to see the difference to be generally small; if our model's expected future implied volatility equals the forward implied volatility it is in some sense self-consistent. Note that this is not the smile consistency we normally see, since this terminology would mean that have a neat calibration of our model to the future implied volatility surface. Figure 6.1 shows us that we indeed have a small error over the whole surface for t close T_1, but unfortunately the error in the problem region small strike and close to maturity increases as T_1 is chosen larger. The error for high strikes stays very low and stable across the different maturities. The same results are found using the other calibration methods.

We conclude that we can price forward start options with this model in a correct way. The most liquid forward start options are the ones where the strike is set higher or equal to ATM, which is exactly where we see the consistency. The forward implied volatility we would quote from the model "agrees" with the dynamics of the call prices. In the next section we will see how products on the market rely on forward start options.

6.2 Structured products

Many products trading on the markets have combined an element of derivatives in them to achieve a specific risk/return profile for the investor. These products are referred to as structured products. They are OTC instruments offering a high level of flexibility. Common features of these products are:

- ◇ Callable, where the issuer can cancel the instrument by buying it back from the investor. This is typically the case when interest rates lower.
- ◇ Putable, where the investor can sell the instrument back to the issuer. This is typically the case when interest rates rise.
- ◇ Convertible, where the instrument can be converted into a certain number of equities. This is often seen as a low risk way of gaining exposure to an equity price increase.

- ◇ Index-linked, where the coupon on the instrument is variable and is determined according to a specified reference index.
- ◇ Yield-enhancing, where the embedded derivatives aim to leverage the product's coupon level.
- ◇ Capital protecting, where the design of the product is such that the investor's capital is guaranteed at maturity, while also having the chance of an excess return. The way to achieve this is by the investment in zero coupon bonds, ensuring a guaranteed capital return. As the bonds will be bought at a discount to their maturity value, this allows money left over to invest in derivatives to construct the required payoff profile.

A commonly sold structured product is the Barrier Reverse Convertible. An example term sheet can be found in appendix A.5.

We explore the use of exotic derivatives in structured products. Suppose an investor has a capital of size N available for a structured product. A capital protection note with a fixed coupon C and participation level x pays

$$H_T(S_T, C, T, x) = N \cdot \left(1 + \max\left(C, x\left(\frac{S_T - S_t}{S_t}\right)\right)\right).$$

Keeping C fixed (possibly at zero), the selling institution will invest in a zero-coupon bond such that it returns $N \cdot (1+C)$ at maturity T and use the residual money to invest in call options to settle participation in the performance of S, where participation level x is left to be determined. Alternatively, the selling institution could fix a participation level x and search for the coupon C such that the difference between the time t price of the zero-coupon bond and its redemption value at time T equals the price of the call structure. Let S denote the vector $(S_0, \ldots, S_M)$, a reverse cliquet becomes

$$H_T(S, C, T) = N \cdot \left(1 + \max\left(C, \sum_{i=1}^{M} \min\left(\frac{S_{T_i}}{S_{T_{i-1}}} - 1, 0\right)\right)\right).$$

The idea remains the same, the institution will invest in a zero-coupon bond and use the residual to invest in options that give the upside participation. The key element is then to find an attractive coupon C that guarantees the minimum return on the structured product. Clearly, a high interest rate environment will make the zero-coupon bond relatively cheap and thus increases the money left over for investing in derivatives.
We will concern ourselves with the pricing of the equity participation and thus indirectly with finding the coupon. The payoff of a reverse cliquet is given by

$$H(S, C, T_M) = N \cdot \max\left(C, \sum_{i=1}^{M} \min\left(\frac{S_{T_i}}{S_{T_{i-1}}} - 1, 0\right)\right).$$

We can see this as a strip of variable notional forward start options and so we can apply our model to find these products.

6.3 Variance swaps

Every derivative has its embedded risks, which are quantified by the derivative's greeks. The most common greek is delta, defined as the sensitivity of the portfolio's value relative to a change in the underlying. The most common eliminated risk is the risk coming from the delta. Another important greek is vega, which measures the sensitivity of the portfolio relative to change in volatility. Equity derivatives depend to a very large extend on volatility and

traders seek ways to eliminate the vega risk they are exposed to. Although vega risk can be eliminated via trading strategies using calls or puts, these turn out not to be self-financing or depend heavily on the strikes of these vanilla options. Apart from this we would also invoke exposure to the underlying , i.e. it will be difficult to hedge vega risk while staying delta neutral. Variance swaps then come in naturally, as their payoff is directly the realized variance of the underlying and does not depend on external factors. Moreover, a variance swap can be replicated via standard calls and puts, easing their replication and hence sale. The wider class of volatility derivatives including e.g. volatility swaps or forward variance calls can be used not only to vega hedge, but also to trade the spread between implied and realized volatility or to gain exposure to future variance.

Let $\mathcal{I} = \{t_0, \dots, t_I = T\}$ be set of daily observation dates, then the realized variance is denoted by

$$\overline{\sigma}^2(S,T) := \frac{1}{T}\sum_{i=1}^{I}\left(\log\left(\frac{S_i}{S_{i-1}}\right)\right)^2 = \frac{252}{I}\sum_{i=1}^{I}\left(\log\left(\frac{S_i}{S_{i-1}}\right)\right)^2. \tag{6.2}$$

Following [Dem99], a variance swap based on notional N pays

$$N \cdot \left(\overline{\sigma}^2(S,T) - K_{vol}^2\right)$$

with agreed observation dates $\mathcal{I}$ and strike σ_K^2. We will assume that the notional amount equals one for ease of computations. The strike K_{vol} is usually chosen such that the risk-neutral expectation of this contract is zero, as with any swap. Assume that the stock price moves according to

$$\frac{dS_t}{S_t} = \mu_t dt + \sigma_t dW_t,$$

where W is a $\mathbb{P}$-Brownian motion. As before, we will use quadratic variation of the returns of S as the unbiased estimator

$$K_{vol}^2 = \frac{1}{T-t}\mathbb{E}^*\left[\langle \log(S)\rangle_T \middle| \mathcal{F}_t\right] = \frac{1}{T-t}\mathbb{E}^*\left[\int_t^T \sigma_u^2 du\right].$$

We are left with finding an expression for this expectation. By applying Itô's lemma we can write

$$d\log(S_t) = \left(\mu_t - \frac{\sigma_t^2}{2}\right)dt + \sigma_t dW_t.$$

Consequently, we get

$$\frac{dS_t}{S_t} - d\log(S_t) = \frac{\sigma_t^2}{2}dt,$$

which brings us to

$$\int_t^T \sigma_u^2 du = 2\left(\int_t^T \frac{dS_u}{S_u} - \log\left(\frac{S_T}{S_t}\right)\right). \tag{6.3}$$

We basically have an expression for K^2, but unfortunately log contracts are not liquidly traded in any market. We can work around this problem by splitting up the logarithm to $\log\left(\frac{S_T}{S_t}\right) = \log\left(\frac{S_T}{F_t(T)}\right) + \log\left(\frac{F_t(T)}{S_t}\right)$. By the static hedge strategy in (2.17), we can replicate the former according to

$$-\mathbb{E}^*\left[\log\left(\frac{S_T}{F_t(T)}\right)\middle|\mathcal{F}_t\right] = \exp\left(\int_t^T r_u du\right)\left(\int_0^{F_t(T)} \frac{1}{K^2}P_t(T,K)dK + \int_{F_t(T)}^{\infty}\frac{1}{K^2}C_t(T,K)dK\right).$$

The latter equals

$$\mathbb{E}^*\left[\log\left(\frac{F_t(T)}{S_t}\right)\middle|\mathcal{F}_t\right] = \int_t^T r_u - d_u\, du = \mathbb{E}^*\left[\int_t^T \frac{dS_u}{S_u}\right],$$

where the last equality uses the fact that integrals over Brownian motions equal to zero. Summarily, we find the fair price of the variance swap to be

$$\begin{aligned}
K_{vol}^2 &= \frac{1}{T-t}\mathbb{E}^*\left[\int_t^T \sigma_u^2 du\right] \\
&= \frac{2}{T-t}\mathbb{E}^*\left[\int_t^T \frac{dS_u}{S_u} - \log\left(\frac{S_T}{S_t}\right)\middle|\mathcal{F}_t\right] \\
&= -\frac{2}{T-t}\mathbb{E}^*\left[\log\left(\frac{S_T}{F_t(T)}\right)\middle|\mathcal{F}_t\right] \\
&= \frac{2}{T-t}\exp\left(\int_t^T r_u du\right)\left(\int_0^{F_t(T)} \frac{1}{K^2}P_t(T,K)dK + \int_{F_t(T)}^{\infty}\frac{1}{K^2}C_t(T,K)dK\right),
\end{aligned}$$

meaning that we can replicate the realized variance on S by a static position in OTM puts and calls, where the quantity of the contracts are inversely proportionally weighted with their squared strike.
Note that we do not have a continuum of option prices, whence we have to approximate K_{vol}^2. We can construct a variance swap curve by using the full option price grid. Let n_τ denote the number such that

$$n_\tau = \underset{n\in\{0,\dots,N\}}{\operatorname{argmin}} \left|K_n - F_t(T)\right|$$

The fair strike is approximated by

$$K_{vol}^2 = \frac{2}{T-t}\exp\left(\int_t^T r_u du\right)\left(\sum_{n=0}^{n_\tau}\frac{1}{k^2}P_t(T,k)dk + \sum_{n=n_\tau}^{K_N}\frac{1}{k^2}C_t(T,k)dk\right). \tag{6.4}$$

We see that strike K_{vol} is highly dependent on the volatility surface. On the other hand, the swap's payoff is completely determined by the actual volatility. Following [Car07], it's observed that the variance swap strikes generally stand above the realized variance. There is hence a "cost of carry" when having a variance swap. This brings us to a practical volatility mismatch, where the trader profits when he is short variance. The Credit Suisse Global Carry Selector I and II funds are two strongly performing funds that obtain their performance from the realized and implied volatility mismatch in variance swaps.

Forward start variance swaps

A forward start variance swap is a contract in which the buyer agrees on time t to pay a fixed K_{vol}^2 on maturity T_2 and in return receive the realized variance accrued over the period from T_1 to T_2, with $t < T_1 < T_2$. We simulate the call prices on a daily basis and consider the call prices on every quarter. On these points in time we take the average of every Monte Carlo, giving us the model's call prices on forward start periods that are multiples of three months. We then calculate the fair price of the variance swap as the sum of OTM option prices weighted by the inverse of their squared strike according to equation (6.4), giving Figure 6.2. We see that the variance swap rates lie higher than the spot curve. The rates do not necessarily lie higher per forward start period, e.g. the curve with a forward start period of nine months lies below the curves with a forward start period of three and six months. A variance swap curve is also generally observed to have an upward slope, rather than the downward slope we have here. Another observation here is that the variance swap rates we see in Figure 6.2 are unfortunately highly unrealistic. Volatility empirically barely spikes above levels as high as 40% and hence variance swap rates obtained from our model at this height makes the model not particularly useful.

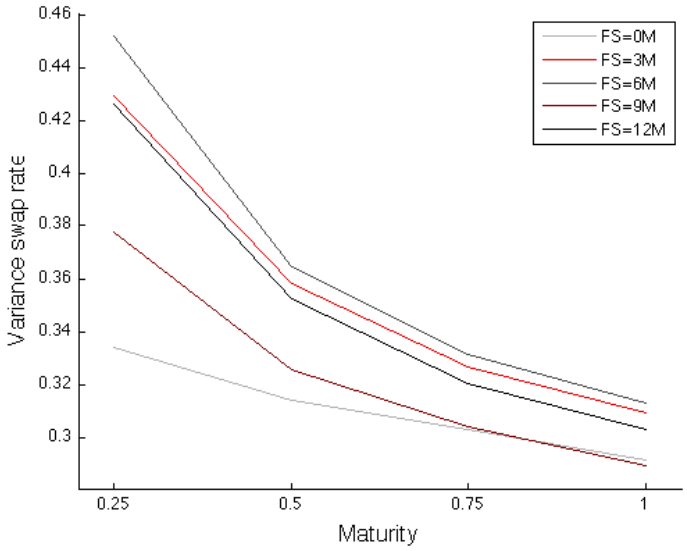


Figure 6.2: The variance swap rates K_{vol} obtained from the model for different forward starting periods.

From the way realized variance is defined in (6.2) we can conclude that variance is additive. From (6.2) we get

$$\overline{\sigma}^2(S,T_2)\cdot T_2 = \overline{\sigma}^2(S,T_1)\cdot T_1 + \overline{\sigma}^2(S,T_2-T_1)\cdot(T_2-T_1), \tag{6.5}$$

so the fair strike for a forward start variance swap starting at T_1 and maturing at T_2 would be

$$\overline{\sigma}^2(S,T_2-T_1) = \frac{\overline{\sigma}^2(S,T_2)\cdot T_2 - \overline{\sigma}^2(S,T_1)\cdot T_1}{T_2-T_1}. \tag{6.6}$$

Equation (6.6) in fact shows us how a variance swap is hedged in the market. The variance swap can be replicated by buying $T_2/(T_2-T_1)$ variance swaps with maturity T_2 and selling $T_1/(T_2-T_1)$ variance swaps with maturity T_1.

We can now compare the forward variance rates that would be quoted according to equation (6.6) to the forward variance swap rates implied by the model. Unfortunately, as discussed above, the forward start variance swap rates obtained from the model are fairly unrealistic. The figure shows that the quoted variance swap rates indeed lie below the high rates implied by the model, except for the maturities from nine months up to one year, where the fit is relatively close for all four observation periods T_2-T_1.

6.4 Barrier options

Among the most liquid exotics are the barrier options. Barrier options come in to or out of existence when the underlying crosses a contract dependent barrier L on pre-specified observation dates t_i. Let $\overline{S}$ denote the running maximum and $\underline{S}$ the running minimum over these observation dates, i.e.

$$\overline{S}_t := \max_{i:\ t_i\le t} S_{t_i}, \quad \underline{S}_t := \min_{i:\ t_i\le t} S_{t_i}.$$

Most commonly the barrier call option is available in four different variants. These are named down-and-in (DI), down-and-out (DO), up-and-in (UI) and up-and-out (UO) with respective payoff functions

$$H^{DI}(S_T,L,K,T) := (S_T-K)^+\mathbf{1}_{\underline{S}_T\le L}, \qquad H^{UI}(S_T,L,K,T) := (S_T-K)^+\mathbf{1}_{\overline{S}_T\ge L},$$
$$H^{DO}(S_T,L,K,T) := (S_T-K)^+\mathbf{1}_{\underline{S}_T> L}, \qquad H^{UO}(S_T,L,K,T) := (S_T-K)^+\mathbf{1}_{\overline{S}_T< L},$$

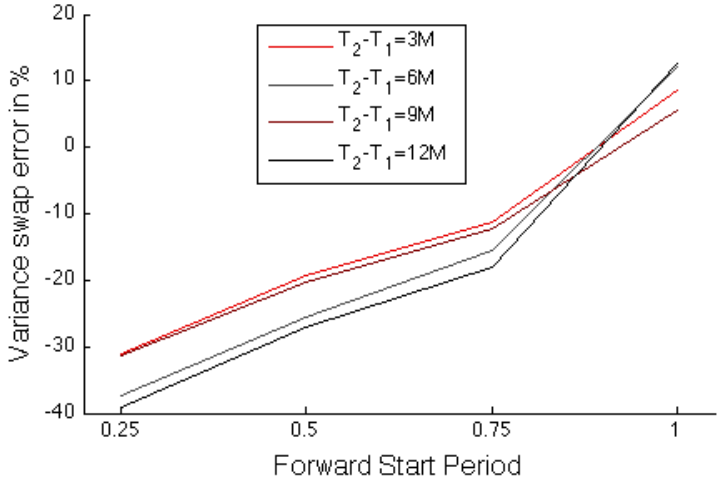


Figure 6.3: Relative error of forward start quotes versus the model's implied variance swap rates.

and corresponding prices $C^{DI}, C^{DO}, C^{UI}, C^{UO}$. It is clear that once a barrier is breached, the value of a barrier option becomes equal to zero for the "out" variants or to that of a vanilla for the "in" variants . For fixed L, K and T we have the equalities

$$C_t = C_t^{DI} + C_t^{DO}, \quad C_t = C_t^{UI} + C_t^{UO}. \tag{6.7}$$

We see that having the market price of a vanilla call together with one modeled barrier price allows us to find all prices. Our model allows for the calculation of the "in" barrier option by the following steps.

(i) Simulation of the stock with options over the interval $[0, T]$.

(ii) Determine $\tau := \inf\{t \in [0, T] : S \geq / \leq L\}$.

(iii) Discount the call value from $t = \tau$ to $t = 0$.

(iv) Average over all sample paths to get the barrier call price.

In practice the observation dates can be of the American, Bermudan or European type. We can easily implement these irregularities in the observation dates by adjusting the dates on which we store the simulations.

Unfortunately, the barrier option valuation is highly dependent on the dynamics of the stock price, which we have seen not to be very accurate for any type of calibration. Moreover, Monte Carlo methods are slow compared to other pricing methods, especially in the case of a full strike and maturity market model where we simulate for many strikes and maturities at the same time. On the other hand we can clearly outnumber the computational time by using the analytical expression for the price of a barrier option in the BS framework. However, the assumed lognormal dynamics for the underlying are wrong and so the BS framework should be rejected for path-dependent options, especially when the barriers are low. Among others, barriers can be priced effectively and realistically using finite difference schemes that outnumber the computational time for low-factor models. It is hence beyond the scope of this thesis to analyze the pricing of barrier options via market models.

7 Conclusion

We have been analyzing the mathematics of a market model. In order for our market model to work, we had to use a grid with arbitrage-free call prices. Unfortunately option prices are quite often with light arbitrages, making the model non-usable. We worked our way around this by applying a smoothing algorithm on the option prices. The algorithm returns an arbitrage-free grid even when the data is not free of arbitrage. We have seen that this enables us to price derivatives such as straddles, strangles and butterfly spreads in an arbitrage-free way.

We parameterized the grid with its call prices on fixed strikes and maturities into a price level and local implied volatilities. The price level turned out to be fairly correlated with the underlying as long as we set the reference strike to at-the-money. The local implied volatility showed similar properties as the well-understood implied volatilities and the price level turned out to be a direct translation of the price skew. We could specify their dynamics via a system of SDEs ensuring the absence of dynamic arbitrage. Since the parameterization is a bijection, we equivalently obtained dynamics for the call prices. The inversion of local implied volatilities and price level to call prices then also ensures the absence of static arbitrage. We have made the model usable in a interest rate and dividend environment by using an affine transformation of the underlying's price. This transformation can be the solution to many more models that don't work in a dividend environment. Having taking interest rates and dividends apart from the price modeling, we could add stochastic dynamics to the interest rates and dividends as well.

After setting up the model we looked into the calibration of the parameters that are left free to choose, subject to some constraints. For every SDE we had to set the Gaussian parameter, being ξ corresponding to price level y and v_n^m corresponding to local implied volatility x_n^m. We tried to calibrate these to the realized volatility of time series of the local implied volatilities, obtaining an one-factor model. Another approach has been to perform a principal component analysis, yielding a three-factor model. Although both methods are appealing, we continued using the principal component three-factor model after considering the numerical stability, speed of calibration and mathematical sense. Unfortunately the model does not enable the user to have a view on the call or stock price process, mainly due to the fact that the Gaussian parameters had to be Lipschitz continuous. Moreover, the stock price dynamics coming from the model are not very realistic, the jump on the first expiry date of the call options is too large. Since the model is more focussed on call price dynamics rather than stock price dynamics, one could argue the relevancy of matching the dynamics of the stock price.
From calibration we went to the pricing of derivatives. We started with looking at forward start options, a class of options with difficult pricing and risk management. The forward start option does not explicitly depend on the stock price dynamics. We have seen that the expected call prices are conform to the forward start prices for strikes higher or equal to at-the-money. Since these are the most liquid forward start options, we found the model to be very suitable for these derivatives. However, we would not need a full grid of option prices, a market model specifying dynamics on a fixed strike relative to the underlying's price would suffice. To show the usability of these derivatives we have seen an example of their role in certain structured products.
Finally we looked at the replication of variance swaps and used the additive property of variance to check if a forward start variance swap priced using the model comes close to the price traders would give based on the observed option price surface. Forward start variance swap prices fully depend on dynamics of the whole option price surface and do not explicitly depend on the stock price dynamics, which would put this market model in a qualified position. The match of the forward start variance swap curve compared to forward start variance swap prices given in practice using the additive property of variance did not turn out to be a close match, except for

forward start periods from nine months up to one year. We are left with the forward start options as the derivative being served most by this market model. We finished with looking into barrier options. Although a market model lends itself very well to price these derivatives, we didn't numerically pursue their pricing. The stock price dynamics will have to be accurate, which is not the case. Apart from that the computational time is high and barrier options belong to the simplest exotic derivatives, leading to many models that can price them faster and more accurate.

The coding needed for this model is included with a discussion of the pitfalls encountered. The smoothing algorithm is widely applicable, whereas the pricing model itself will serve fewer readers.

Future research is left for market models that work for all strikes and maturities, rather than on a fixed strike and maturity grid. We have worked our way around this by applying the smoothing algorithm at every modeled time, but this is not a neat procedure.
From a mathematical point of view the hunt is after a model that could take data (possibly containing arbitrage) on a variable strike and maturity grid which returns call prices on arbitrary strikes and maturities. From a practical point of view it is not necessary for the model to work for all strike and maturities at the same time, since for pricing purposes it is generally sufficient to have all strikes versus one maturity or one strike versus all maturities, where the latter reminds us of modeling the yield curve in the interest rate world. These conditions have mathematically been worked out for fixed income and equity markets and save a lot of computational time compared to a market model on a full grid.

References

[Ber07] Bermudez, A., Buehler, A., Ferraris, A., Jordinson, C., Lamnouar, A. *Equity Hybrid Derivatives*. John Wiley & Sons, 2007.

[Bla73] Black, F., Scholes, M. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–659, 1973.

[Bra01] Brace, A., Goldys, B., Klebaner, F., Wombersley, R. Market model of stochastic implied volatility with application to the bgm model. *Working paper, University of New South Wales*, 2001.

[Bue09] Buehler, A. Volatility and dividends. *Draft 1.201, TU Berlin*, 2009.

[Car02] Carr, P.,Verma, A. Static and semi-static hedging with options. *Presentation for Cornell Theory Center*, 2002.

[Car07] Carr, P., Wu, L. Variance risk premia. *AFA 2005 Philadelphia Meetings*, 2007.

[Car08] Carmona, R., Nadtochiy, S. Local volatility dynamic models. *Finance and Stochastics*, 13(1):1–48, 2008.

[Con02] Cont R., da Fonseca J. Dynamics of implied volatility surfaces. *Quantitative Finance*, 2(1):45–60, 2002.

[Con03] Cont, R., Tankov, P. *Financial Modelling with Jump Processes*. Chapman & Hall/CRC, 2003.

[Cox85] Cox, J.C., Ingersoll Jr., J.E., Ross, S.A. A theory of the term structure of interest rates. *Econometrica*, 53(2):385–407, 1985.

[Dav07] Davis, M., Hobson, D. The range of traded option prices. *Mathematical Finance*, 17:1–14, 2007.

[Dem99] Demeterfi, K., Derman, E., Kamal, M., Zou, J. More than you ever wanted to know about volatility swaps. *Goldman Sachs Quantitative Research Notes*, 1999.

[Der94] Derman, E., Kani, I. The volatility and its implied tree. *Goldman Sachs Quantitative Research Notes*, 1994.

[Dri03] Driver, B.K. *Analysis Tools with Applications*. Springer, 2003.

[Dum98] Dumas, B., Fleming, J., Whaley, R.E. Implied volatility functions: empirical tests. *The Journal of Finance*, 53(6):32–39, 1998.

[Dup94] Dupire, B. Pricing with a smile. *Risk*, 7:32–39, 1994.

[Dup96] Dupire, B. A unified theory of volatility. *Discussion paper Paribas Capital Markets*, 1996.

[Fen09] Fengler, M.R. Arbitrage-free smoothing of the implied volatility surface. *Quantitative Finance, Taylor and Francis Journals*, 9(4):417–428, 2009.

[Fri75] Friedman, A. *Stochastic Differential Equations and Applications*. Dover Publications, 1975.

[Gre93] Green, P. J., Silverman, B. W. *Nonparametric Regression and Generalized Linear Models: a roughness penalty approach*. Chapman and Hall, 1993.

[Hag02] Hagan, P.S., Kumar, D., Lesniewski, A.S., Woodward, D.E. Managing smile risk. *Wilmott Magazine*, September:84 – 108, 2002.

[Hea92] Heath, D., Jarrow, R., Morton, A. Bond pricing and the term structure of interest rates: a new methodology for contingent claims valuation. *Econometrica*, 60:77–105, 1992.

[Hes93] Heston, S. Closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2):327–343, 1993.

[Hul00] Hull, J. *Options, Futures, and Other Derivatives*. Prentice Hall, Englewood Cliffs, New Jersey, 4th edition, 2000.

[Jac10] Jacod, J., Protter, P. Risk neutral compatibility with option prices. *Finance and Stochastics*, 14(2):285–315, 2010.

[Kah04] Kahalé, N. An arbitrage-free interpolation of volatilities. *Risk*, 17(5):102–106, 2004.

[Led02] Ledoit, O., Santa-Clara, P., Yan, S. Relative pricing of options with stochastic volatility. *Working paper, University of California*, 2002.

[MAT09] MATLAB. *version 7.10.0 (R2009a)*. The MathWorks Inc., Natick, Massachusetts, 2009.

[Pro04] Protter, P.E. *Stochastic Integration and Differential Equations*. Springer, 2 edition, 2004.

[Sam65] Samuelson, P. Rational rheory of warrant pricing. *Industrial Management Review*, 6:13–31, 1965.

[Sch99] Schönbucher, P.J. A market model for stochastic implied volatility. *Philosophical Transactions of the Royal Society A*, 357(1758):2071–2092, 1999.

[Sch08a] Schweizer, M., Wissel, J.S. Arbitrage-free market model for option prices: the multi-strike case. *Finance and Stochastics*, 12(4):469–505, 2008.

[Sch08b] Schweizer, M., Wissel, J.S. Term structures of implied volatilities: absence of arbitrage and existence results. *Mathematical Finance*, 18(1):77–114, 2008.

[Ste91] Stein, E., Stein, J. Stock price distributions with stochastic volatility. *Review of Financial Studies*, 4(4):727–752, 1991.

[Wis08] Wissel, J.S. Arbitrage-free market model for option prices. *Dissertation, Eidgenössische Technische Hochschule ETH Zürich*, 2008.

A Appendix

In this appendix we will go through the Matlab codes corresponding to the theory we have gone through. We will discuss the pitfalls in the algorithms as well as certain choices made throughout the coding.
We assume that we are given the index level, a $M \times N$ matrix of quoted call prices, an interest rate r and a dividend yield d corresponding to observation dates t_k. We conclude the appendix with a term sheet of a structured product.

A.1 Inverting Black & Scholes

The inversion of the Black & Scholes call price formula is a fundamental procedure for this thesis. We hence give the Matlab function that inverts (1.2) below. If additionally a forward start date T_1 is specified, the inverse is taken using the Black and Scholes formula for forward start call options (6.1) instead.

```
function volatility = blsimplvol(S,T,K,optionvalue,r,q,T_1)

% Set options
options = optimset('fzero');
options = optimset(options, 'TolX', 1e-6, 'Display', 'off');
LB = eps(1);
UB = 10;

if nargin ==6
    if blsprice(S,T,K,LB,r,q) < optionvalue

        if blsprice(S,T,K,UB,r,q) <= optionvalue
            volatility = NaN;
        else
            [volatility, fval, exitFlag] = fzero(@objfcn,[LB UB],options,...
                S,K,r,T,optionvalue,q);
            if exitFlag < 0
                volatility = NaN;
            end
        end
    else

        Spot = S
        Strike = K
        Maturity = T
        PriceSigma0 = blsprice(S,T,K,LB,r,q)
        OptionValue = optionvalue
        PriceSigma2 = blsprice(S,T,K,UB,r,q)
        volatility  = NaN;
    end

else %nargin==7

    if blsprice(S,T,K,LB,r,q,T_1)< exp(-r*T_1)*optionvalue

        if blsprice(S,T,K,UB,r,q,T_1) <=  exp(-r*T_1)*optionvalue
            volatility = NaN;
```

```
        else
            [volatility, fval, exitFlag] = fzero(@objfcn2,[LB UB],options,...
                S,K,r,T,optionvalue,q,T_1);
            if exitFlag < 0
                volatility = NaN;
            end
        end
    else

        Spot = S
        Strike = K
        Maturity = T
        PriceSigma0T1 = blsprice(S,T,K,LB,r,q,T_1)
        OptionValue   = optionvalue
        PriceSigma2T1 = blsprice(S,T,K,UB,r,q,T_1)
        volatility    = NaN;

    end
end
end

function delta = objfcn(volatility, S, K, r, T, optionvalue,q)
    callprice = blsprice(S,T,K,volatility,r,q);
    delta = callprice - optionvalue;
end


function delta = objfcn2(volatility, S, K, r, T, optionvalue,q,T_1)
    callprice = blsprice(S,T,K,volatility,r,q,T_1);
    delta = callprice - exp(-r*T_1)*optionvalue;
end
```

A.2 Arbitrage-free smoothing of the call price surface

For Fengler's algorithm to smooth the call price surface, we used the code below. This code assumes that we have pre-smoothed the price surface such that the whole $M \times N$ matrix of call prices is filled up with quotes. First thing to notice is that we invoke a parameter Ns, which is there to make the algorithm work for pure call prices as well. It is there to blow up the index level and call prices, since Matlab's quadratic programming $\mathrm{quadprog}$ does not find a solution when the difference between the call prices is very small. $\mathrm{quadprog}$ does not obey the boundaries of the constraints in which it has to find a solution perfectly and therefore we change boundaries that should just be zero to a value $\epsilon = 10^{-6}$.

```
function [CALL_AF,PUT_AF,IMPLV_AF] = Fengler(stock,fmoneynessCALL_AF,...
    maturity,fmoneynessCALL,CALL,L,r,q,Nd,Ns,CALLonFixedGrid,...
    CALL_AFonFixedGrid,t)
%FENGLER is the numerical implementation of Fengler's algorithm step II,
% in which he smooths the implied volatilty surface using cubic splines.
% Input CALL is a full call surface on grid fmoneynessCALL X maturity.
% This is likely to have a lot of arbitrage.
% Input fmoneynessCALL_AF is in order of magnitude of CALL
```

```
% Input fmoneynessCALL the vector of size(CALL,2) with containing
% Input maturity is in years

% If Ns>1 while working with real stock and call prices, then the error
% will be that there is no feasible solution

% Set parameters and sizes
M  = length(maturity);
Nk = size(fmoneynessCALL_AF,2)-1; %Number of active strikes
CALL_AF  = zeros(M,Nk);
l        = find(any(CALL,2),1,'first');

if size(fmoneynessCALL_AF,1)==1 && CALL_AFonFixedGrid == 1
    fmoneynessCALL_AF = repmat(fmoneynessCALL_AF,[M 1]);
end
% Set scaling
Kmax = fmoneynessCALL_AF(:,end);
fmoneynessCALL_AF(:,end) = [];
fmoneynessCALL_AF = fmoneynessCALL_AF*Ns;
fmoneynessCALL = fmoneynessCALL*Ns;
CALL  = CALL*Ns;
stock = stock*Ns;
Kmax  = Kmax*Ns;

for m=M:-1:1

    r_loc(m) = max(1,round(maturity(m)*Nd));
    forward  = stock*exp( (r(r_loc(m)) - q)*maturity(m) );

    if CALL_AFonFixedGrid == 1
        K_AF(m,:) = fmoneynessCALL_AF(m,:);
        K_max     = Kmax(m,1);
    else
        K_AF(m,:)= fmoneynessCALL_AF.*forward;
        K_max    = Kmax*forward;
    end

    if CALLonFixedGrid == 1
        % Plug in the fixed grid
        K = unique(horzcat(fmoneynessCALL(m,:),K_max,1.05*K_max));
    else %CALLonFixedGrid = 0
        K = fmoneynessCALL.*forward;
    end

    N = length(K);
    Q = zeros(N,N-2);
    R = zeros(N-2,N-2);
    h = diff(K);

    for j=1:N-1
        if j ~=1
            Q(j,j)   = - h(j)^(-1) - h(j-1)^(-1);
            Q(j+1,j) = h(j)^(-1);
```

```
            Q(j-1,j) = h(j-1)^(-1);
            R(j,j)   = (h(j-1)+h(j))/3;
            if j~=N-1
                R(j+1,j) = h(j)/6;
                R(j,j+1) = R(j+1,j);
            end
        end
    end

    Q(:,1)   = [];
    R(:,1)   = [];
    R(1,:)   = [];

    A = vertcat(Q,-R');
    B = horzcat(vertcat(diag(ones(N,1)),zeros(N-2,N)),vertcat(zeros(N,N-2),L*R));
    y = zeros(2*N-2,1);

    if CALLonFixedGrid ==1
        y(1:N,1) = horzcat(CALL(m,:),blsprice(stock,maturity(m),...
            K_max,0.3,r(r_loc(m)),q),blsprice(stock,maturity(m),...
            1.1*K_max,0.3,r(r_loc(m)),q));
    else
        y(1:N,1) = horzcat(CALL(m,:));
    end

    epsilon   = 10^(-6); %epsilon for avoiding machine precision
    A1        = zeros(2,2*N-2);
    B1(1:2,1) = 0;
    A1(1,1)   = 1;
    A1(1,2)   = -1;
    A1(2,N-1) = -1;
    A1(2,N)   = 1;
    B1(1,1)   = exp(-r(r_loc(m))*maturity(m))*(K(2)-K(1));
    B1(2,1)   = -epsilon;

    % Ensure a non-negative call spread at strike N
    if CALLonFixedGrid == 0
        if K_AF(Nk) < max(K)

            idx       = find(round(100*K_AF(Nk))<=round(100*K),1,'last');
            A1          = vertcat(A1,0*A1(1,:));
            A1(3,idx-1) = -(K_max - fmoneynessCALL_AF(Nk));
            A1(3,idx)   = K_max - fmoneynessCALL_AF(Nk-1);
            B1(3,1)     = -10^(-5); %Make larger if LIV at strike N is weird

        else
            warning('Convexity in maximum AF strike not guaranteed')
        end
    end

    % Due to working so close to machine precision:
    lb        = ones(2*N-2,1)*epsilon;
    lb(1:N)   = lb(1:N);            % Increase lower bound of call price
```

```
115     lb(1)     = exp(-q*maturity(m))*stock - exp(-r(r_loc(m))*maturity(m))*K(1);
116     options   = optimset('LargeScale','off');
117
118     if m==M
119         x0        = zeros(2*N-2,1);
120         x0(1:N)   = exp(-q*maturity(m))*stock;
121         ub        = ones(2*N-2,1);
122         ub(2:end) = Inf;
123         ub(1)     = exp(-q*maturity(m))*stock;
124
125         x(m,:)    = quadprog(B,-y,A1,B1,A',zeros(N-2,1),lb,ub,x0,options);
126     else
127         x0          = x(m+1,:);
128         ub          = ones(2*N-2,1);
129         ub(N+1:end) = Inf;
130         ub(1:N)     = exp(q*(maturity(m+1)-maturity(m))).*x(m+1,1:N);
131
132         % Close maturities give ub(1) < lb(1) sometimes. We make this
133         % call a bit more cheaper to let quadprog still continue
134
135         if ub(1)<lb(1)
136             warning('ub(1)<lb(1). Fixed')
137             idx     = (ub(1:N)<lb(1:N));
138             lb(idx) = ub(idx)-epsilon;
139         end
140
141         if ub(N)<lb(N)
142             warning('ub(N)<lb(N). Fixed')
143             idx = (ub(1:N)<lb(1:N));
144             lb(idx) = ub(idx)-epsilon;
145         end
146
147         if ub(N)<lb(N)
148             warning('Change ub(idx)-epsilon')
149         end
150
151         x(m,:) = quadprog(B,-y,A1,B1,A',zeros(N-2,1),lb,ub,x0,options);
152     end
153
154     gamma = zeros(N,1);
155     gamma(2:N-1,1) = x(m,N+1:end);
156
157     if min(min(x(:,1:N)))<0
158         x(x<0) = eps; % Due to machine precision
159     end
160
161     if min(min(x(:,N+1:end)))<0
162         warning('Feng: There is a negative gamma. Increase lb of gamma');
163     end
164
165     % Pricing on AF grid
166     for n=1:Nk
167         if K_AF(m,n)>=K_max
```

```
168             CALL_AF(m,n)=0;
169         else
170
171             i = find(K <= K_AF(m,n), 1, 'last' );
172             K(i)    = round(K(i));
173             K_AF(m,n) = round(K_AF(m,n));
174
175             if i==N-1 %Moet eigenlijk N-1 zijn en als dan i=N dan probleem
176
177                 b = (x(m,N-1) - x(m,N-2))/h(N-2);
178                 a = x(m,N-1);
179
180                 CALL_AF(m,n) = ...
181                     min(CALL_AF(m,n-1)/2,max(+b*(K_AF(m,n)-K(i)) + a, ....
182                     blsprice(stock,maturity(m),K_AF(m,n),0.35,r(r_loc(m)),q)));
183
184             elseif i==N
185
186                 CALL_AF(m,n) = min(CALL_AF(m,n-1)/2,...
187                 blsprice(stock,maturity(m),K_AF(m,n),0.35,r(r_loc(m)),q));
188
189             elseif i>0
190
191                 d = (gamma(i+1,1)-gamma(i,1))/(6*h(i));
192                 c = gamma(i,1)/2;
193                 b = (x(m,i+1)-x(m,i))/h(i) - h(i)*(2*gamma(i)+gamma(i+1))/6;
194                 a = x(m,i);
195
196                 CALL_AF(m,n) = d*(K_AF(m,n)-K(i))^3 + c*(K_AF(m,n)-K(i))^2 + ...
197                     b*(K_AF(m,n)-K(i)) + a;
198
199                 if CALL_AF(m,n)<0
200                     warning('Fengler:Negative call price. Change epsilon')
201                     if m<M && n>1
202                         CALL_AF(m,n)= min(CALL_AF(m,n-1),CALL_AF(m+1,n))/2;
203                     end
204                 end
205
206             else % Case i=0
207
208                 b = (x(m,2) - x(m,1))/h(1) - h(1)*gamma(2)/6;
209                 a = x(m,1);
210
211                 CALL_AF(m,n) = b*(K_AF(m,n)-K(1)) + a;
212
213             end
214         end
215     end
216 end
217
218 % If an arbitrage free PUT surface is requested
219 if nargout > 1
220     PUT_AF = 0*CALL_AF;
```

```
    for m=1:M
        D_r = exp(-r(r_loc(m))*maturity(m));
        D_q = exp(-1.5*q*maturity(m));
        for n=1:Nk
            PUT_AF(m,n) = CALL_AF(m,n) + D_r*K_AF(m,n) - D_q*stock;
        end
    end
end

% If an arbitrage-free IVS is requested
if nargout > 2
    IMPLV_AF = 0*CALL_AF;
    for m=1:M
        for n=1:Nk
            IMPLV_AF(m,n) = blsimplvol(stock,maturity(m)-t,K_AF(m,n),CALL_AF(m,n),r(r_loc(m)),q);
        end
    end
end

CALL_AF = CALL_AF/Ns;

end
```

A.3 Local implied volatility and price level

```
function x = LocImplVol(t,stock,callmatrix,maturity,strike,q)
%LV_SURFACE returns the local implied volatility surface of callmatrix with size(maturity,strike-1)
% equals size(callmatrix)
% Input maturity = [T_1,...,T_M]
% Input strike = [K_1,...,K_{N+1}]

% l selects shortest maturity for which options are alive.
l = find(t<maturity,1,'first');

% Set sizes
M = size(callmatrix,1);
N = size(callmatrix,2);
x = 0*callmatrix;

for m=1:M
for n=1:N

    if m==l

        call0 = stock*exp(-q*(maturity(m)-t));
        numerator    = ( strike(n+1)-strike(n) )/( strike(n)*sqrt(maturity(m)-t) );

        if n==1
            denumerator1 = norminv( (call0-callmatrix(m,n)             )/( strike(n)-0           ));
            denumerator2 = norminv( (callmatrix(m,n)-callmatrix(m,n+1) )/( strike(n+1)-strike(n) ));
```

```
        elseif n==N
            denumerator1 = norminv( (callmatrix(m,n-1)-callmatrix(m,n) )/( strike(n)-strike(n-1) ));
            denumerator2 = norminv( (callmatrix(m,n)-0                 )/( strike(n+1)-strike(n) ));
        else
            denumerator1 = norminv( (callmatrix(m,n-1)-callmatrix(m,n) )/( strike(n)-strike(n-1) ));
            denumerator2 = norminv( (callmatrix(m,n)-callmatrix(m,n+1) )/( strike(n+1)-strike(n) ));
        end

        x(m,n) = numerator / (denumerator1 - denumerator2);

    else

        call0 = stock*exp(-q*(maturity(l)-t));
        numerator = (callmatrix(m,n) - callmatrix(m-1,n) )/( maturity(m) - maturity(m-1) );

        if n==1
            denumeratornumerator    = ( (strike(n)-0)*callmatrix(m,n+1) ...
                - (strike(n+1)-0)*callmatrix(m,n) + (strike(n+1)-strike(n))*call0 )*(strike(n))^2;
            denumeratordenumerator  = ( strike(n+1)-strike(n) )*( strike(n)-0 )*( strike(n+1)-0 );
        elseif n==N
            denumeratornumerator    = ( (strike(n)-strike(n-1))*0 ...
                - (strike(n+1)-strike(n-1))*callmatrix(m,n) ...
                + (strike(n+1)-strike(n))*callmatrix(m,n-1) )*(strike(n))^2;
            denumeratordenumerator  = ( strike(n+1)-strike(n) )*( strike(n)-strike(n-1) )...
                * ( strike(n+1)-strike(n-1) );
        else
            denumeratornumerator    = ( (strike(n)-strike(n-1))*callmatrix(m,n+1) ...
                - (strike(n+1)-strike(n-1))*callmatrix(m,n) ...
                + (strike(n+1)-strike(n))*callmatrix(m,n-1) )*(strike(n))^2;
            denumeratordenumerator  = ( strike(n+1)-strike(n) )*( strike(n)-strike(n-1) )...
                * ( strike(n+1)-strike(n-1) );
        end

        x(m,n) = sqrt(numerator/(denumeratornumerator/denumeratordenumerator));

    end

    if x(m,n)==0
        warning('LocImplVol: There is a zero LIV')
    end

    if x(m,n) > 2
        warning('LocImplVol: There is an extreme LIV too large.Change strike maturity grid')
        x(m,n)=2;
    end

    if abs(imag(x(m,n))) > 0 || isnan(x(m,n))==1
        warning('LocImplVol: There is a negative LIV.')
        x(m,n)=-0.01;
    end

    if numerator<0
        warning('LocImplVol','Decreasing call price in m>1')
```

```
79      end
80
81  end
82  end
83
84  end
```

```
1   function y= PriceLevel(t,CallMatrix,maturity,strike,RefStrike)
2   %PRICE_LEVEL returns the price level of callmatrix with:
3   % Input maturity with numel(maturity) = size(CallMatrix,1).
4   % Input strike needs to contain a highest strike for which all prices are zero.
5
6   % Include the case RefStrike = N by adding an extra column of zeros
7   CallMatrix = horzcat(CallMatrix,zeros(size(CallMatrix,1),1));
8
9   % l selects shortest maturity for which options are alive.
10  l = find(t<maturity ,1,'first');
11
12  y = sqrt(maturity(l) - t)* norminv( (CallMatrix(l,RefStrike) - CallMatrix(l,RefStrike+1)) / ...
13      (strike(RefStrike+1) - strike(RefStrike)) );
14  end
```

A.4 Calibration

Coding the calibration turns out to be a tedious procedure. We first consider the method using realized volatility and after that we turn to the method using principal component analysis.

Realized volatility

Calibration using realized volatility relies heavily upon Matlab's implemented $\mathrm{lsqcurvefit}$, which tries to find the best data fit for nonlinear functions.

```
1   function [v,xi,RVs,Xvol,Yvol,LambdaCheck,resnorm] = ...
2       Calibrate_RV(stock,PURECALLS_AF,PureX,PureY,T_liquid,MONEY_total,RefStrike)
3   %CALIBRATE_RV calibrates v and xi using realized volatility.
4   % Input stock and PureY are series of the stock and price level on the observations dates
5   % Input PURECALLS_AF and PureX are [T_1,T_1+dt...,T_M,T_M+dt] x [K_1,...K_N]
6   % T_liquid = [T_1,...,T_M]
7   % MONEY_total = [K_1,...,K_{N+1}]]
8   % RefStrike = K_{n^*}
9
10  % Set parameters and sizes
11  D=1;
12  Nb = 252;
13  Nobs = length(stock);
14  M = length(T_liquid);
15  Nk = length(MONEY_total)-1;
16
```

```
% Calculate realized volatility of LIV time series
for m=1:M
    % Delete the +dt maturities
    PureCalls_AF(m,:,:) = PURECALLS_AF(2*m-1,:,:);
    for n=1:Nk
        % Variance of X
        Xvol(m,n) = sqrt(var(squeeze(PureX(m,n,:))));
        % Quadratic Variance of X
        %Xvol2(m,n) = sqrt(Nb*sum(diff(PUREx(2*m-1,n,:),1,3).^2,3)/(Nobs-1));
    end
end
Yvol = sqrt(var(PureY));
Xvol(Xvol<eps)=eps;

% CALCULATE VARIATIONS
RVstock = sqrt( Nb * sum(diff(log(stock)).^2) / (Nobs-1) );
RVcalls = zeros(M,Nk);
for m=1:M
    SUM = zeros(1,Nk);
    for i=1:Nobs-1
    SUM(1,:) = SUM(1,:) + (squeeze(PURECALLS_AF(2*m,:,i+1)-PURECALLS_AF(2*m-1,:,i))).^2;
    end
    RVcalls(m,:) = sqrt(Nb * SUM / (Nobs-1));
end
RVstock = repmat(RVstock,[M 1]);

% Calibration on full surface
v_xi  = 0.1*horzcat(ones(M,size(PureX,2)),ones(M,1)); % [T_1,...,T_M] x [K_1,...,K_N xi]
x_y   = zeros(M,Nk+1);
for m=1:M
    x_y(m,:) = horzcat(PureX(m,:,end-1),PureY(end-1));
end

RVs = horzcat(RVstock,RVcalls);
options = optimset('TolFun',1e-8,'TolX',1e-8,'PrecondBandWidth',0,'MaxFunEvals',20000);
InitialCondition = zeros(size(v_xi));

% Set lower and upper bounds for v^m_n and xi
ub = Inf*ones(size(InitialCondition));
lb = -Inf*ones(size(ub));
lb(:,end) = 0;

% FOR XI
%ub(1,end)     = 1;
%lb(1,end)     = 1-eps;

%ub(1,end)     = Yvol;
%lb(1,end)     = Yvol-eps;

% Keeping v equal to Xvol
%ub(:,1:end-1) = eps;
%lb(:,1:end-1) = -eps;

```

```
% All parameters zero and xi == 1
ub(:,1:end-1) = Xvol(:,1:end);
lb(:,1:end-1) = -Xvol(:,1:end);
%           lb(:,end,1) = 0;

[v_xi,resnorm] = ...
    lsqcurvefit(@(v_xi,x_y) lambda(v_xi,x_y,T_liquid,MONEY_total,D,...
    PureCalls_AF,RefStrike),InitialCondition,x_y,RVs,lb,ub,options);

%v_xi(v_xi(:,1:end-1,:)>1)=1;
%v_xi(v_xi(:,1:end-1,:)<-1)=-1;
% Input x_y = [T_1 (,...,T_m)] x [K_1,...,K_N y]
LambdaCheck = lambda(v_xi,x_y,T_liquid,MONEY_total,D,PureCalls_AF,RefStrike);
v = v_xi(:,1:end-1,:);
xi = squeeze(v_xi(1,end,:));

end

function lambda = lambda(v_xi,x_y,maturity,strike,D,callmatrix,RefStrike)
%LAMBDA2 returns lambdastock = \lambda^m_0 and lambda = \lambda^1_n for all m
% corresponding to every x^m_n, to be used for calibration of v and xi
% Input strike = K_1,...,K_{N+1}
% Input maturity = T_1,..,T_M
% Input X = [T_1,..,T_M] X [K_1,..,K_N]

% Input v_xi = [T_1,..,T_M] X [K_1,..,K_N XI] x D
% Input x_y = [T_1 (,...,T_m)] x [K_1,...,K_N y] x D if UsedForDaily = 0
% Input x_y = [T_1 (,...,T_m)] x [K_1,...,K_N y] x (Nobs-1) if UsedForDaily = 1

% Input callmatrix = [T_1 (,...,T_m)] x [K_1,...,K_N y] x D

% Output lambda = [T_1 (,..,T_M] X [K_1,..,K_N] x D

v = v_xi(:,1:end-1,:);
% v = [T_1 (,...,T_M)] x [K_1,...,K_N] x D
xi = squeeze(v_xi(1,end,:))';
% |xi|^2 = 1 if D=2 by
% xi(1,1) = sqrt(1-xi(1,2)^2);
xi = repmat(xi,[size(v,1) 1]);
% xi = [T_1 (,...,T_M)] x D

x  = x_y(:,1:end-1,:);
% x = [T_1 (,...,T_M)] x [K_1,...,K_N] x Nobs
y  = x_y(1,end,:);
y  = squeeze(y);
% y = Nobs x 1

t = 0;
l = find(t < maturity,1,'first');
M = size(x,1);
N = size(x,2);

% Set sizes
```

```
NewSum      = zeros(D,1);
PreviousSum = 0*NewSum;
lambdastock = zeros(M,1,D);
lambdas     = zeros(M,N,D);

for m=1:M
    if m==1

        Sum  = zeros(N+1,1);
        SumV = zeros(N+1,D);

        if RefStrike<N
            for n=RefStrike-2:N-1
            Sum(n+2,1)  = Sum(n+1,1)  - (strike(n+1)-strike(n))/(strike(n)*x(1,n));
            SumV(n+2,:) = SumV(n+1,:) - squeeze(v(1,n,:))'*(strike(n+1)-strike(n))/ ...
                (strike(n)*x(1,n));
            end
        end

        for n=RefStrike-1:-1:0

            if x(m,n+1)~=0
                Sum(n+1,1) = Sum(n+2,1)  + ...
                (strike(n+2)-strike(n+1))/(strike(n+1)*x(1,n+1));

                SumV(n+1,:) = SumV(n+2,:) + ...
                squeeze(v(1,n+1,:))'*(strike(n+2)-strike(n+1))/(strike(n+1)*x(1,n+1));
            end

        end

        for n=N:-1:0

            if n~=N
                PreviousSum = squeeze(lambdas(m,n+1,:));
            end

            if n>0
                NewSum = normpdf( (y+Sum(n+1,1))/sqrt(maturity(l)-t))*...
                    (xi(m,:)' - squeeze(SumV(n+1,:))')...
                    *(strike(n+1)-strike(n))/sqrt(maturity(l)-t);
            else
                NewSum = normpdf( (y+squeeze(Sum(n+1,1)))/sqrt(maturity(l)-t))*...
                    (xi(m,:)' - squeeze(SumV(n+1,:))')...
                    *(strike(n+1)-0)/sqrt(maturity(l)-t);
            end

            if n~=0
                lambdas(l,n,:) = PreviousSum + NewSum;
            else
                lambdastock(l,1,:) = PreviousSum + NewSum;
            end
        end
```

```
176
177     else
178
179         lambdastock(m,1,:) = lambdastock(l,1,:);
180         Amatrix = A(x,maturity,strike); % N x N x M matrix
181         Bvec    = zeros(N,D);
182
183         for n=1:N
184             Bvec(n,:) = squeeze(lambdastock(m,1,:))*beta(m,1,maturity,strike)*x(m,1)^2 ...
185                    + (squeeze(lambdas(m-1,n,:))+2*squeeze(v(m,n,:))*...
186                    (callmatrix(m,n)-callmatrix(m-1,n)));
187         end
188
189         lambdas(m,:,:) = linsolve(Amatrix(:,:,m),Bvec);
190
191     end
192
193     lambda = horzcat(lambdastock,lambdas);
194
195 end
196
197 lambda = squeeze(lambda);
198
199 end
200
201 function A = A(x,maturity,strike)
202 % Input LIV x corresponding to [T_1,...,T_M] X [K_1,...,K_N]
203 % Input maturity = [T_1,...,T_M]
204 % Input strike = [K_1,...,K_{N+1}]
205 % Output A = N x N x M
206
207 M = size(x,1);
208 N = size(x,2);
209 A = zeros(N,N,M);
210
211 % Note that A for m=1 will not be used
212 for m=2:M
213        for i=1:N
214           A(i,i,m) = 1+( beta(m,i,maturity,strike)+gamma(m,i,maturity,strike) )*x(m,i)^2;
215        end
216        for i=1:N-1
217           A(i,i+1,m)= -gamma(m,i,maturity,strike)*x(m,i)^2;
218           A(i+1,i,m)= -beta(m,i+1,maturity,strike)*x(m,i+1)^2;
219        end
220 end
221 end
222
223 function beta = beta(m,n,maturity,strike)
224 % Input m>1=1
225 % Input maturity = [T_1,...,T_M]
226 % Input strike is [K_1,...,K_{N+1}]
227
228     numerator   = ((strike(n+1)-strike(n))*(maturity(m)-maturity(m-1)).*(strike(n)).^2 );
```

```
229 if n==1
230     denumerator = ((strike(n+1)-0)*(strike(n+1)-strike(n))*(strike(n)-0));
231 else
232     denumerator = ((strike(n+1)-strike(n-1))*(strike(n+1)-strike(n))*(strike(n)-strike(n-1)));
233 end
234     beta = numerator / denumerator;
235 end
236
237 function gamma = gamma(m,n,maturity,strike)
238 % Input m>l=1
239 % Input maturity = [T_1,...,T_M]
240 % Input strike is [K_1,...,K_{N+1}]
241
242 if n==1
243     numerator   = ((strike(n)-0)*(maturity(m)-maturity(m-1))*(strike(n))^2 );
244     denumerator =   ((strike(n+1)-0)*(strike(n+1)-strike(n))*(strike(n)-0));
245 else
246     numerator   = ((strike(n)-strike(n-1))*(maturity(m)-maturity(m-1))*(strike(n))^2 );
247     denumerator =   ((strike(n+1)-strike(n-1))*(strike(n+1)-strike(n))*(strike(n)-strike(n-1)));
248 end
249     gamma = numerator / denumerator;
250 end
```

Principal component analysis

Calibration using PCA starts with the log-transformation of the local implied volatilities. Calculating the covariance of a $M \times N$ matrix should be interpreted as taking the covariance between two vectors of length $M \cdot N$. This requires some ordering work done by the function $\mathrm{covariance3D}$.

```
1  function [v,DdNormalizedSquared,CutOff] = Calibrate_PCA(PureX,PureY,D,CutOffLevel,Nb)
2  %CALIBRATE_PCA returns a calibrated v using principal component analysis
3
4  % Filter negative X - should not occur
5
6  if sum(sum(any(PureX<0)))>0
7      warning('Calibrate_PCA: There is a negative LIV')
8      PureX(PureX<=0)=0.01;
9  end
10
11 % Get time series U by transforming to Z and dividing by sqrt(1/Nb)
12 U = sqrt(Nb)*diff(Ztransform(PureX),1,3);
13
14 % Covariance matrix (NxM) x (NxM)
15 K_cov = covariance3D(U);
16
17 % size(h,3) is gelijk aan = D
18 h = PureX(:,:,end);
19 for d=1:D-1
20     h = cat(3,h,PureX(:,:,end-d));
21 end
22
```

```
% Solve Cc*Aa = Bb*Aa*Dd or C*f = H*f*lambda
Bb = Integrate_Shape(h);
Cc = Integrate_ShapeCovariance(h,K_cov);
[Aa,Dd] = eig(Cc,Bb);

% Normalize & order descending
[Dd,idx] = sort(diag(Dd)/norm(diag(Dd)),'descend');
Aa = Aa(:,idx);
Aa_norm = vnorm(Aa);
for j=1:size(Aa,1)
    Aa(j,:) = Aa(j,:)./Aa_norm;
end

% Show the contribution per component
DdNormalizedSquared = Dd.^2;
CutOff = find(cumsum(DdNormalizedSquared)>CutOffLevel,1,'first')

f_k = eigenfunctions(Aa,h);

v = zeros(size(h));
for d=1:size(v,3)
    v(:,:,d) = Dd(d)*f_k(:,:,d);
end

v(:,:,CutOff+1:end)=[];

end

function Z = Ztransform(x)

Z = zeros(size(x));

Z(1,:,:)     = log(x(1,:,:)) - 1./x(1,:,:);
Z(2:end,:,:) = log(x(2:end,:,:));

end

function COV = covariance3D(U)
%COV(U) calculates the covariance of a 3 dimensional matrix with with 2
% dimensional observations and size(U,3) observations

% Set sizes
[M N Nobs] = size(U);
COV = zeros(M,N,M,N);

% Make time series U zero-mean
U = U - repmat(mean(U,3),[1 1 size(U,3)]);

for m=1:M
for n=1:N
for mm=1:M
for nn=1:N
    COV(m,n,mm,nn) = sum( (U(m,n,:)).*(U(mm,nn,:))  ) / ( Nobs-1 );
```

```
76  end
77  end
78  end
79  end
80
81
82  end
83
84  function B = Integrate_Shape(h)
85
86  % Set sizes
87  [M N D] = size(h);
88  B = zeros(D);
89
90  for i=1:D
91  for j=1:D
92      B(i,j) = sum(sum(h(:,:,i).*h(:,:,j)));
93  end
94  end
95
96  end
97
98  function C = Integrate_ShapeCovariance(h,K)
99
100 % Set sizes
101 [M N D] = size(h);
102 C = zeros(D);
103
104 for i=1:D
105 for j=1:D
106     SUM = 0;
107     for m=1:M
108     for n=1:N
109         SUM = SUM + sum(sum(h(m,n,i).*squeeze(K(m,n,:,:)).*squeeze(h(:,:,j))));
110     end
111     end
112     C(i,j) = SUM;
113 end
114 end
115
116 end
117
118 function f_k = eigenfunctions(f,h)
119 %EIGENFUNCTIONS calculates f_k = sum_i f_{i,k} h_i
120 % h = M x N x D
121
122 % Set sizes
123 [M N D] = size(h);
124 f_k = zeros(M,N,D);
125
126 for i=1:D
127 for m=1:M
128 for n=1:N
```

```
129     f_k(m,n,i) = f(:,i)'*squeeze(h(m,n,:));
130 end
131 end
132 end
133
134 end
```

Evolution of the call price surface

For every Monte Carlo we have to repeat steps (ii)-(v) of section 4.3 to simulate the call price surface over time. After the determination of ξ and v via the calibration procedure above, we let the call prices evolve over time in terms of local implied volatility x and price level y via SDEs (3.9) and (3.10). Every time step involves the conversion of these values into pure call prices. We translate these pure prices into real call prices on pre-specified time points $\mathrm{OutputSteps}$, which will give the output of the function split up in primarily stock price, call price and put price. Additional output such as the process of the local implied volatilities, price level, pure stock price and pure call prices can be requested, as well as the expected future and forward implied volatility surfaces.

```
1  function [Stock,Calls,Puts,X_i,Y_i,PureStocks,PureCalls,Exp_Fut_IVs,Forward_IVs]= ...
2      TimeSeries2(PureStock_tmin1,CallSurface_tmin1,X_tmin1,Y_tmin1,tmin1,t,maturity,PricingStrikes,...
3      strike,dt,v,xi,Stock_tmin1,RefStrike,r,q)
4  %TIME_SERIES calculates the time evolution from the CallSurface_tmin1, X_tmin1 and Y_tmin1 to t.
5  % Input t gives the date(s) on which we want to see the output
6  % Input maturity = [T_1,...,T_M]
7  % Input strike = [K_1,...,K_{N+1}]
8  % Input v = [T_1,...,T_M] x [K_1,...,K_N] x D
9  % Input xi = D x 1
10
11 if any(X_tmin1<0)==1
12     warning('Negative LIV, change input TimeSeries2 to different LIV matrix')
13 end
14
15 % Set sizes and parameters
16 M  = size(X_tmin1,1);
17 N  = size(X_tmin1,2);
18 Nd = 360;
19 Ntimesteps  = floor((max(t)-tmin1)/dt);
20 OutputSteps = floor((t-tmin1)./dt);
21 counter     = 1;
22 D           = length(xi);
23 X_i         = zeros(M,N,Ntimesteps+1);
24 X_i(:,:,1)  = X_tmin1;
25 Y_i         = zeros(Ntimesteps+1,1);
26 Y_i(1)      = Y_tmin1;
27 PureStocks  = zeros(Ntimesteps+1,1);
28 PureStocks(1,1)  = PureStock_tmin1;
29 PureCalls       = zeros(M,N,Ntimesteps+1);
30 PureCalls(:,:,1) = CallSurface_tmin1;
31 RealStrikes = zeros(M,N);
32
33 for m=1:M
```

```
    r_loc    = max(1,round(maturity(m)*Nd));
    forward  = Stock_tmin1*exp((r(r_loc) - q)*maturity(m));
    RealStrikes(m,:) = strike(1:end-1)*forward;
end

Stock = zeros(length(OutputSteps),1);
Calls = zeros(M,length(PricingStrikes)-1,length(OutputSteps));
Puts = Calls;
Exp_Fut_IVs = Calls;
Forward_IVs = Calls;

% Perform Ntimesteps steps of size dt
for i=2:Ntimesteps+1

    % Find first active maturity
    l = find(tmin1+(i-2)*dt < maturity,1,'first');

    % Lipschitz conditions
    clear XI
    V = zeros(size(v));
    V(l:end,:,:) = v(1:end-l+1,:,:);

    % Let V converge to zero and xi to norm 1
    if any(xi==1)==1
        ChangeXI=0;
    else
        ChangeXI=1;
    end

    %ChangeXI=0
    if l==1

        V(l,:,:) = v(l,:,:) - v(1,:,:)*(i-2)*dt/(maturity(l) - tmin1);

        if ChangeXI==0
            XI = xi;
        else
            XI = xi + (1/sqrt(D)-xi)*(i-2)*dt/(maturity(l) - tmin1);
        end

    else

        V(l,:,:) = v(1,:,:) - v(1,:,:)   *((i-2)*dt-maturity(l-1))/(maturity(l) - maturity(l-1));

        if ChangeXI==0
            XI = xi;
        else
            XI = xi + (1/sqrt(D)-xi)*((i-2)*dt-maturity(l-1))/(maturity(l) - maturity(l-1));
        end
    end

    % Avoid dividing by zero
    if (i-1)*dt == maturity(l)
```

```
87          dtt = dt-eps(1);
88      else
89        dtt= dt;
90      end
91
92      % Calculate lambdas
93      [lambdastock,lambda] = ...
94          lambda2(tmin1+(i-2)*dtt,X_i(:,:,i-1),Y_i(i-1),V,XI,...
95          PureCalls(:,:,i-1),maturity,strike,RefStrike);
96
97      % Create D-dimensional Brownian motion movements
98      Normaldtt = sqrt(dtt).*randn(D,1);
99
100     % MOVEMENTS UNDER P^*
101     VdW = zeros(M,N);
102
103     for d=1:D
104         VdW(:,:)       = VdW + V(:,:,d)*Normaldtt(d);
105     end
106
107     MU = ...
108         mu(tmin1+(i-2)*dtt,Y_i(i-1),XI,maturity);
109     Y_i(i) = ...
110         Y_i(i-1) + MU*dtt + XI'*Normaldtt;
111     U = ...
112         u(tmin1+(i-2)*dtt,D,X_i(:,:,i-1),Y_i(i-1),V,XI,...
113         lambda,PureCalls(:,:,i-1),maturity,strike,RefStrike);
114     X_i(:,:,i) = ...
115         X_i(:,:,i-1) + X_i(:,:,i-1).*( U.*dtt + VdW);
116
117     [PureStocks(i,1), PureCalls(:,:,i)] = ...
118         CallPrice(tmin1+(i-1)*dtt,X_i(:,:,i-1),Y_i(i-1),maturity,strike,RefStrike);
119
120     % Obtain call, put and ivs surface at time t
121     if i == OutputSteps(counter)+1
122
123         Stock(counter) = Stock_tmin1*PureStocks(i,1);
124         RealCalls=zeros*PureCalls(:,:,i);
125
126         for m=1:M
127             D_q      = exp(-q*(maturity(m) - (i-1)*dt));
128             Dforward = Stock_tmin1*D_q;
129
130             RealCalls(m,:) = Dforward*PureCalls(m,:,i);
131         end
132
133         % Change to F/S grid by Fengler
134         L_smooth = 100;
135
136         % Calculates the call surface at the counter time on the stock
137         % moneynes grid.
138         [Calls(1:end,:,counter),Puts(1:end,:,counter)] =  ....
139             Fengler(Stock(counter),PricingStrikes*Stock(counter),maturity(1:end)...
```

```
            -(tmin1+(i-1)*dtt),RealStrikes,RealCalls(l:end,:),L_smooth,r,q,Nd,1,1,1);

        counter = counter+1;
    end
end
end

function MU = mu(t,y,xi,maturity)

l = find(t < maturity,1,'first');
MU = .5*y*( norm(xi)^2 - 1 )/( maturity(l)-t );

end

function U = u(t,D,x,y,v_t,xi_t,lambda,callmatrix,maturity,strike,RefStrike)
%U calculates the drift coefficient in the SDE of LIV X
% Input x = [T_1,...,T_M] x [K_1,...,K_N]
% Output u = [T_1,...,T_M] x [K_1,...,K_N]

% Set paramters and sizes
l = find(t < maturity,1,'first');
M = size(x,1);
N = size(x,2);
U = zeros(size(M,N));

for m=l:M
    if m==l

        Sum  = 0;
        SumV = zeros(D,1);

        if RefStrike<N
            for n=RefStrike+1:N
                if x(m,n)~=0
                    v_n   = squeeze(v_t(m,n,:)); % Column vector
                    SumV1 = SumV;
                    SumV  = SumV + v_n.*(strike(n+1)-strike(n))/(x(m,n)*strike(n));
                    Sum   = Sum + (strike(n+1)-strike(n))/(x(m,n)*strike(n));
                end
                U(m,n) = norm(v_n)^2 + (.5 - .5*norm(xi_t + SumV)^2 ...
                    + (y+Sum')*(xi_t' + SumV1')*v_n  )/( maturity(m)-t );
            end
        end

        for n=RefStrike
            v_n        = squeeze(v_t(m,n,:));
            SumV  = 0*SumV;
            SumV1 = SumV;
            U(m,n) = norm(v_n)^2 + (.5 - .5*norm(xi_t - SumV)^2 ...
                + (y+Sum')*(xi_t' - SumV1')*v_n  )/( maturity(m)-t );
        end

        if RefStrike>1
```

```
            for n=RefStrike-1:-1:1
                if x(m,n+1)~=0
                    v_n   = squeeze(v_t(m,n+1,:));
                    SumV1 = SumV;
                    SumV  = SumV + v_n.*(strike(n+2)-strike(n+1))/(x(m,n+1)*strike(n+1));
                    Sum   = Sum + (strike(n+2)-strike(n+1))/(x(m,n+1)*strike(n+1));
                end
                U(m,n) = norm(v_n)^2 + (.5 - .5*norm(xi_t - SumV)^2 ...
                    + (y+Sum')*(xi_t' - SumV1')*v_n  )/( maturity(m)-t );
            end
        end

    else % m>1
        for n=1:N
            v_n = squeeze(v_t(m,n,:));
            U(m,n) = (  (squeeze(lambda(m-1,n,:)-lambda(m,n,:)))'./ ...
                (callmatrix(m,n)-callmatrix(m-1,n)) + 1.5*v_n')*v_n;
        end
    end
end
end

function [lambdastock,lambdas] = lambda2(t,x,y,v,xi,callmatrix,maturity,strike,RefStrike)
%LAMBDA2 returns lambdastock = \lambda^m_0 and lambda = \lambda^1_n for all m
% corresponding to every x^m_n, to be used for TimeSeries
% Input strike = [K_1,...,K_{N+1}]
% Input maturity = [T_1,..,T_M]
% Input X = [T_1,..,T_M] X [K_1,..,K_N]
% Input v  = [T_1,..,T_M] X [K_1,..,K_N] x [1,..,D]
% Input xi = D x 1
% Input x  = [T_1,...,T_m] x [K_1,...,K_N]
% Input y  = 1 x 1
% Input callmatrix = [T_1,...,T_m] x [K_1,...,K_N]
% Output lambdastock = D x 1
% Output lambda = [T_1,..,T_M] X [K_0,...,K_N] x D

l = find(t < maturity,1,'first');
M = size(x,1);
N = size(x,2);
D = length(xi);
%xi = repmat(xi,[1 size(v,1)])';

% Set sizes
NewSum      = zeros(D,1);
PreviousSum = NewSum;
lambdastock = zeros(D,1);
lambdas     = zeros(M,N,D);
Amatrix     = A(x,maturity,strike); % N x N x M matrix

for m=l:M
    if m==l

        Sum  = zeros(N+1,1);
```

```
246         SumV = zeros(N+1,D);
247
248         if RefStrike<N
249             for n=RefStrike-2:N-1
250                     Sum(n+2,1)   = Sum(n+1,1)  - (strike(n+1)-strike(n))/(strike(n)*x(l,n));
251                     SumV(n+2,:) = SumV(n+1,:) - squeeze(v(1,n,:))'*(strike(n+1)-strike(n)) ...
252                         /(strike(n)*x(l,n));
253             end
254         end
255
256         for n=RefStrike-1:-1:0
257             if x(m,n+1)~=0
258                 Sum(n+1,1)   = Sum(n+2,1)  + (strike(n+2)-strike(n+1))/(strike(n+1)*x(l,n+1));
259                 SumV(n+1,:) = SumV(n+2,:) + squeeze(v(1,n+1,:))'*(strike(n+2)-strike(n+1))...
260                     /(strike(n+1)*x(l,n+1));
261             end
262         end
263
264         for n=N:-1:0
265
266             if n~=N
267             PreviousSum = squeeze(lambdas(m,n+1,:));
268             end
269
270             if n>0
271                 NewSum      = normpdf( (y+Sum(n+1,1))/sqrt(maturity(l)-t))*...
272                     (xi - squeeze(SumV(n+1,:))')*(strike(n+1)-strike(n))/sqrt(maturity(l)-t);
273             else
274                 NewSum      = normpdf( (y+Sum(n+1,1))/sqrt(maturity(l)-t))*...
275                     (xi - squeeze(SumV(n+1,:))')*(strike(n+1)-0)/sqrt(maturity(l)-t);
276             end
277
278             if n~=0
279                 lambdas(l,n,:) = PreviousSum + NewSum;
280             else
281                 lambdastock(:) = PreviousSum + NewSum;
282             end
283         end
284
285     else % m>1
286
287         Bvec = zeros(N,D);
288
289         for n=1:N
290             Bvec(n,:) = lambdastock(:)*beta(m,1,maturity,strike)*x(m,1)^2 ...
291                    + (squeeze(lambdas(m-1,n,:))+2*squeeze(v(m,n,:))*...
292                    (callmatrix(m,n)-callmatrix(m-1,n)));
293         end
294
295         lambdas(m,:,:) = linsolve(Amatrix(:,:,m),Bvec);
296
297     end
298 end
```

```
299 end
300
301 function [A] = A(x,maturity,strike)
302 % Input LIV x corresponding to [T_1,...,T_M] X [K_1,...,K_N]
303 % Input maturity = [T_1,...,T_M]
304 % Input strike = [K_1,...,K_{N+1}]
305 % Output A = N x N x M
306
307 % Set sizes
308 M = size(x,1);
309 N = size(x,2);
310 A = zeros(N,N,M);
311
312 % Note that A for m=1 will not be used
313 for m=2:M
314     for i=1:N
315         A(i,i,m) = 1+( beta(m,i,maturity,strike)+gamma(m,i,maturity,strike) )*x(m,i)^2;
316     end
317     for i=1:N-1
318         A(i,i+1,m)= -gamma(m,i,maturity,strike)*x(m,i)^2;
319         A(i+1,i,m)= -beta(m,i+1,maturity,strike)*x(m,i+1)^2;
320     end
321 end
322 end
323
324 function [beta] = beta(m,n,maturity,strike)
325 % Input m>1=1
326 % Input maturity = [T_1,...,T_M]
327 % Input strike is [K_1,...,K_{N+1}]
328
329     numerator   = ((strike(n+1)-strike(n))*(maturity(m)-maturity(m-1)).*(strike(n)).^2 );
330 if n==1
331     denumerator = ((strike(n+1)-0)*(strike(n+1)-strike(n))*(strike(n)-0));
332 else
333     denumerator = ((strike(n+1)-strike(n-1))*(strike(n+1)-strike(n))*(strike(n)-strike(n-1)));
334 end
335     beta = numerator / denumerator;
336 end
337
338 function [gamma] = gamma(m,n,maturity,strike)
339 % Input m>1=1
340 % Input maturity = [T_1,...,T_M]
341 % Input strike is [K_1,...,K_{N+1}]
342
343 if n==1
344     numerator   = ((strike(n)-0)*(maturity(m)-maturity(m-1))*(strike(n))^2 );
345     denumerator =   ((strike(n+1)-0)*(strike(n+1)-strike(n))*(strike(n)-0));
346 else
347     numerator   = ((strike(n)-strike(n-1))*(maturity(m)-maturity(m-1))*(strike(n))^2 );
348     denumerator =   ((strike(n+1)-strike(n-1))*(strike(n+1)-strike(n))*(strike(n)-strike(n-1)));
349 end
350     gamma = numerator / denumerator;
351 end
```

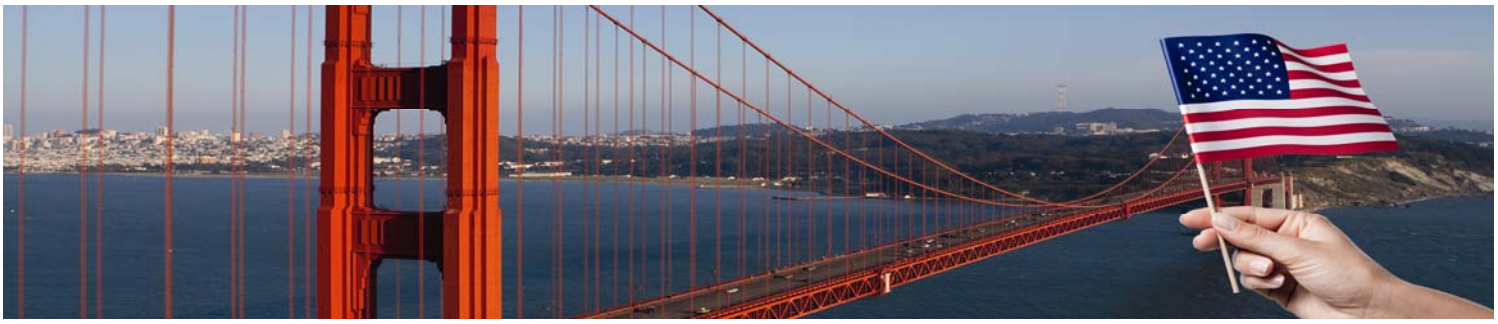
A.5 Term sheet

Below we see the term sheet of a structured product. A term sheet is a non-binding agreement stipulating the major terms and conditions of an investment. The structured product covered in the below term sheet regards a 2 year investment that will pay an annual coupon of 7.75% per annum and guarantees the repayment of the notional as long as the 3 underlying shares do not trade below 49% of the strike price their, which is set ATM based on the closing price of the shares as of 8 June 2012. The repayment is guaranteed as well when the barrier of 49% is breached but the underlying shares have a closing price at maturity that is greater than the set strike price. When they do not satisfy this condition, the investor will receive an amount equal to

$$\min\left(\frac{S_T^i}{S_t^i} \;\middle|\; i = 1, 2, 3\right),$$

where S^i denotes the closing price of share i at maturity T or strike fixing date t.

CREDIT SUISSE



Barrier Reverse Convertible

Indicative Terms, 31 May 2012

Subscription Period until 8 June 2012, 3:00 p.m. CET

7.75% p.a. on Coca-Cola / McDonald's / Starbucks

18 June 2012 until 18 June 2014

Barrier Reverse Convertibles offer an attractive yield in the form of a coupon. Depending on the performance of the Underlyings, the Barrier Reverse Convertibles will either be redeemed at 100% or the Underlying with the worst performance will be delivered (see Redemption Mode). The Coupon will be paid out in any case.

Your market expectations: sideways to slightly positive

This structured product does not constitute a participation in a collective investment scheme within the meaning of the Swiss Federal Act on Collective Investment Schemes (CISA) and is therefore not subject to authorization and supervision by the Swiss Financial Market Supervisory Authority (FINMA).

1. Product Description

Swiss Sec. Number / ISIN	18 599 715 / CH0185997159 (WKN: CLA3NV)
Ticker	CLANP
Product Type	Yield-Enhancement Products (category 1230: Barrier Reverse Convertibles), according to the Swiss Derivative Map of the Swiss Structured Products Association (www.svsp-verband.ch). Detailed information on profit and loss prospects, as well as risks can be found in section 2 and 3 on the following pages.
Issuer	Credit Suisse AG, Zurich, acting through its Nassau Branch, Bahamas
Lead Manager	Credit Suisse AG, Zurich
Rating	Aa1 (Moody's) / A+ (Standard & Poor's) / A (Fitch)
Calculation / Paying Agent	Credit Suisse AG, Zurich

Underlyings

Underlying	Bloomberg	ind. Strike Price	ind. Barrier	ind. No. of shares (Ratio)	Exchange
The Coca-Cola Company share	KO UN	USD 75.06	USD 36.7794	13.3227	New York Stock Exchange
McDonald's Corporation share	MCD UN	USD 90.11	USD 44.1539	11.0975	New York Stock Exchange
Starbucks Corporation share	SBUX UW	USD 54.73	USD 26.8177	18.2715	Nasdaq

Currency	USD
Issue Price	100% (USD 1,000)
Issue Size	USD 10,000,000 (10,000 Barrier Reverse Convertibles)
Denomination	USD 1,000 (Notional Amount) = 1 Barrier Reverse Convertible
Initial Fixing Date	8 June 2012
Payment Date	18 June 2012 (payment of the Issue Price)
Strike Price	100% of the official closing price of the respective Underlying on the relevant Exchange on the Initial Fixing Date
Coupon	**7.75% p.a. (indicative), paid annually** (Interest Payment: 0.62% p.a., Premium Payment: 7.13% p.a.) (indicative)
Coupon Payment Date(s)	18 June 2013, 18 June 2014 (Following Business Day Convention)
Barrier	**49%** (indicative) of the respective Strike Price
Barrier Period	**11 June 2012 until 12 June 2014 (continuous monitoring)**

Redemption Mode

a) If the Underlyings have never been traded at or below their Barriers during the Barrier Period, each Barrier Reverse Convertible will be redeemed at 100% of the Notional Amount (USD 1,000).

b) If at least one Underlying has been traded at or below its Barrier during the Barrier Period and if the Final Fixing Prices are equal to or higher than the respective Strike Prices, each Barrier Reverse Convertible will be redeemed at 100% of the Notional Amount (USD 1,000).

c) If at least one Underlying has been traded at or below its Barrier during the Barrier Period and if at least one of the Final Fixing Prices is lower than the respective Strike Price, the investor will receive the Underlying with the worst performance between Initial and Final Fixing Date in the respective above specified Ratio per Barrier Reverse Convertible. The respective fraction, calculated based on the Final Fixing Price of the delivered Underlying, will not be cumulated and will be paid out in cash.

Information

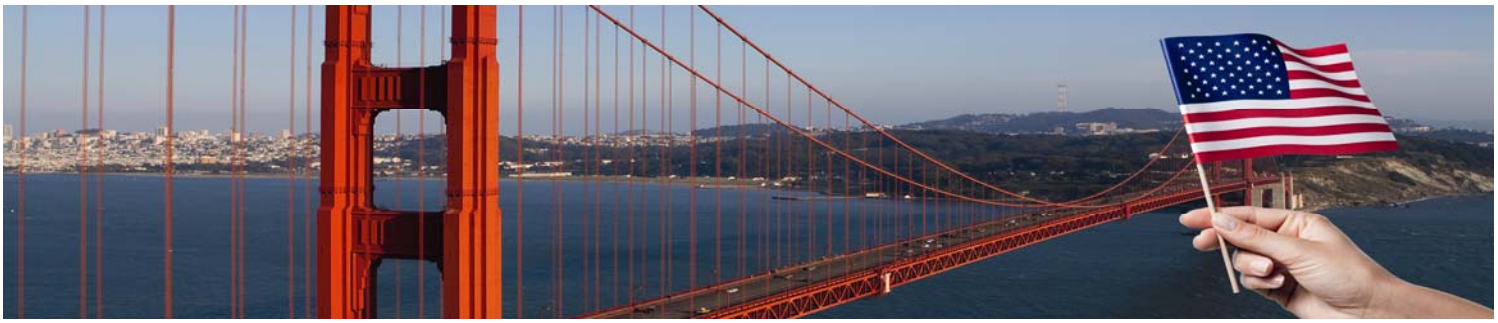
Credit Suisse AG
Transaction Advisory Group, XLAP 1
P.O. Box
CH-8070 Zurich

Telephone: +41 (0) 844 844 002*
Internet: www.credit-suisse.com/derivatives

* Please note that conversations on this line are recorded. We will assume your consent.
The original version of this document is in German; versions in other languages are non-binding translations only. This document can be obtained free of charge at the above address.

Product Identifiers

Swiss Sec. Number	18 599 715
ISIN	CH0185997159
Ticker	CLANP



Last Trading Date	12 June 2014 (until 12:00 p.m. CET)
Final Fixing Date	12 June 2014
Final Fixing Price	100% of the official closing price of the respective Underlying on the relevant Exchange on the Final Fixing Date
Redemption Date	18 June 2014 (redemption of the Barrier Reverse Convertibles)
Minimum Trading Lot	USD 1,000 (1 Barrier Reverse Convertible)
Listing	SIX Swiss Exchange listing will be applied for.
Clearing	SIX SIS Ltd, Clearstream Banking, Euroclear
Price Setting	Secondary market prices are quoted "clean", i.e. the accrued Coupon is not included in the price.
Secondary Market	Under normal market conditions, secondary trading will be maintained throughout the term of the product, during which the bid and offer prices may differ (spread).
Coupon Basis	30/360 (unadjusted)
Settlement	Cash settlement in USD / physical delivery
Publications	www.credit-suisse.com/derivatives, SIX Telekurs Ltd (publication of incidental changes and adjustments to the terms and conditions of the Barrier Reverse Convertibles, e.g. in consequence of an extraordinary event); Bloomberg <CSZH>, Reuters <CSZEQ00> (publication of prices)
Form of Securities	Barrier Reverse Convertibles are issued in the form of *unverurkundete Wertrechte* (dematerialised securities) by registration with SIX SIS Ltd. The Barrier Reverse Convertibles will not be issued in physical or certificated form, but will be maintained as intermediated securities while they are outstanding.
Sales Restrictions	USA, U.S. Persons, UK, European Economic Area, Bahamas
Governing Law / Jurisdiction	Swiss law; exclusive place of jurisdiction is Zurich
Fees	The distributor or the investor's bank may impose an agio of up to 2% on the Issue Price. Furthermore, the distributor or the investor's bank may impose a commission/brokerage fee. In connection with the product, the Issuer and/or its affiliates may pay to third parties, or receive from third parties as part of their compensation or otherwise, one-time or recurring remunerations (e.g. placement or holding fees). Please contact Credit Suisse AG for further information.
Taxes	**Stamp duties** No Swiss stamp tax will be imposed at issuance (primary market). However, Swiss security transfer stamp tax of 0.15% will be charged to Swiss resident investors on secondary market transactions (TK-Code 22). The investor will have to bear Swiss security transfer stamp tax and usual fees in case of delivery of an Underlying, based on the Strike Price. **Withholding tax** This product is not subject to Swiss withholding tax. **Income tax** The Interest Payment is subject to Swiss income tax for Swiss resident private investors. The Premium Payment qualifies as tax free capital gain for Swiss resident private investors and private assets. **EU savings tax** Certain payments made by Swiss paying agents to EU resident individuals will be subject to EU withholding tax. The Swiss paying agents may therefore withhold such amounts as are necessary to pay the EU withholding tax (TK-Code 6; "in scope"). The aforementioned taxes are valid at the time of launch of the issue and are not exclusive. Any taxation will depend on the investor's personal circumstances. The relevant tax laws or the regulations of the tax authorities are subject to change. Credit Suisse AG expressly excludes all liability in respect of any tax implications.

2. Profit and Loss Prospects

Each Barrier Reverse Convertible entitles its holder to receive the Coupon according to the coupon payment schedule. The Coupon will be paid out regardless of the performance of the Underlyings. The Barrier Reverse Convertibles are conditionally capital protected, i.e. they are capital protected as long as no Barrier has been breached. The Barrier Reverse Convertibles will be redeemed according to the Redemption Mode. If none of the Underlyings breaches its Barrier or if all Final Fixing Prices are equal to or higher than the respective Strike Prices, the investor will receive 100% of the Notional Amount. If at least one Underlying breaches its Barrier and if at least one of the Final Fixing Prices is lower than the respective Strike Price, the investor will receive the Underlying with the worst performance between Initial and Final Fixing Date in the respective above specified Ratio per Barrier Reverse Convertible. The respective fraction, calculated based on the Final Fixing Price of the delivered Underlying, will not be cumulated and will be paid out in cash.

The potential profit is limited to the Coupon. The maximum loss an investor may sustain consists in a total loss of the invested amount.

Information

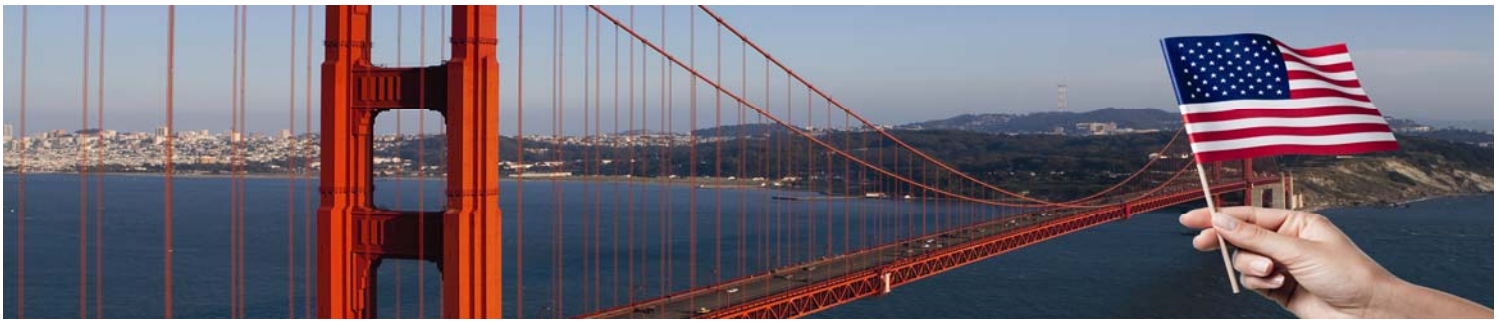
Credit Suisse AG
Transaction Advisory Group, XLAP 1
P.O. Box
CH-8070 Zurich

Telephone: +41 (0) 844 844 002*
Internet: www.credit-suisse.com/derivatives

* Please note that conversations on this line are recorded. We will assume your consent.
The original version of this document is in German; versions in other languages are non-binding translations only. This document can be obtained free of charge at the above address.

Product Identifiers

Swiss Sec. Number	18 599 715
ISIN	CH0185997159
Ticker	CLANP



3. Risks

General Risks of Structured Products

This investment product is a complex structured financial instrument and involves a high degree of risk. It is intended only for investors who understand and are capable of assuming all risks involved. Before entering into any transaction, investors should determine if this product suits their particular circumstances and should independently assess (with the assistance of any professional advisers) the specific risks (maximum loss, currency risks, etc.) and the legal, regulatory, credit, tax and accounting implications. The Issuer and/or its affiliates make no representation as to the suitability or appropriateness of this investment product for any particular investor or as to the future performance of this investment product. This document does not replace a personal conversation with your relationship manager, which is recommended by Credit Suisse AG before any investment decision.

This investment product is a derivative financial instrument and does not constitute collective capital investments within the meaning of the Federal Act on Collective Investment Schemes (CISA). Accordingly, it is not subject to the regulations of the CISA or the supervision of the Swiss Financial Market Supervisory Authority (FINMA). Consequently, the investor does not have the benefit of the specific investor protection provided by the CISA.

The Issuer has no obligation to invest in the Underlyings and investors have no recourse to the Underlyings or distributions, if any, made based on the Underlyings. From time to time the Issuer may engage in transactions with regard to the Underlyings or derivatives thereon. In addition, the Issuer may enter into certain transactions for purposes of hedging against risks emanating from issuing structured products. Such transactions may have a (potentially adverse) effect on the performance of the Underlyings or the performance of the product itself.

Product-Type-specific Risks

An investment in these Barrier Reverse Convertibles is not the same as an investment in the Underlyings. Changes in the market value of the Underlyings may not result in a comparable change in the value of the Barrier Reverse Convertibles. The potential loss associated with an investment in these Barrier Reverse Convertibles is limited to the difference in percent between the Strike Price and the value of the delivered Underlying on the Redemption Date, which may lead to a complete loss of the investment made. Nevertheless, the Barrier Reverse Convertibles may trade considerably below the Issue Price during their term, regardless of any Barrier being reached or breached. The Coupon will be paid in any case. For further details please consult the risk disclosure brochure "Special Risks in Securities Trading", which can be obtained free of charge from Credit Suisse AG.

Currency Risk

The investor may be exposed to a currency risk if the product is denominated in a currency other than that of the country in which the investor is resident. Currency fluctuations may therefore have an impact on the value of the investment.

Issuer Risk

The investor is subject to the risk of an impairment of the Issuer's financial strength; therefore, the value of this investment does not only depend on the performance and quality of the Underlyings but also on the Issuer's creditworthiness. The product is a direct, unsubordinated, unconditional and unsecured obligation of Credit Suisse AG and ranks equally with all of its other obligations of the equivalent type. The rating of Credit Suisse AG is Aa1 (Moody's) / A+ (Standard & Poor's) / A (Fitch). Credit Suisse AG is subject to the supervision of FINMA.

This document constitutes marketing material and is not the result of a financial analysis or research and therefore not subject to the Swiss Bankers Association's "Directives on the Independence of Financial Research". This document has been produced by Credit Suisse AG, Zurich, solely for information purposes and does not constitute an offer or a solicitation of an offer to purchase or sell any securities. Detailed information on Credit Suisse AG can be found in the Base Prospectus for the issuance of Warrants of Credit Suisse AG, dated 27 June 2011, under the heading 'Description of the Issuer' (pages 82 to 112). The legally binding terms as well as the Base Prospectus with regard to the description of the Issuer may be obtained directly from Credit Suisse AG.

For products not listed on the SIX Swiss Exchange this document constitutes the simplified prospectus as defined by art. 5 CISA. It does not constitute a listing prospectus under the rules of the SIX Swiss Exchange or a prospectus in the sense of art. 652a resp. 1156 of the Swiss Code of Obligations.

Information

Credit Suisse AG
Transaction Advisory Group, XLAP 1
P.O. Box
CH-8070 Zurich

Telephone: +41 (0) 844 844 002*
Internet: www.credit-suisse.com/derivatives

* Please note that conversations on this line are recorded. We will assume your consent.
The original version of this document is in German; versions in other languages are non-binding translations only. This document can be obtained free of charge at the above address.

Product Identifiers

Swiss Sec. Number	18 599 715
ISIN	CH0185997159
Ticker	CLANP