Parametric Arbitrage-free Models for Implied Smile Dynamics



Junchen Zhao

Lady Margaret Hall

University of Oxford

A thesis submitted in partial fulfillment of the MSc in

Mathematical and Computational Finance

June 2010

To my loving dad, who has not only earned his bread but also my bread for over twenty years.

And to my loving mum, who could never fall asleep if she has not heard my voice on the phone.

I love you!

Acknowledgement

I would like to thank my supervisor, Dr. Sergey Nadtochiy, who strongly supported my work on the thesis. Every time I meet him, I learn a lot. I believe that without his supervision, this thesis could never be finished of this quality.

Abstract

Based on the theory of Tangent Lévy model [1] developed by R. Carmona and S. Nadtochiy, this thesis gives a paramatrized realization of dynamic implied smile. After specifying a Dirac style Lévy measure, we give argument about the consistency issue of our model with the Tangent Lévy Model. A corresponding no arbitrage drift condition is derived for the parameters. Numerical setup under our model for option pricing and parameter estimation for calibration is given. Implementation results are illustrated in detail and in the end we provide with simulation results of one day ahead implied smile.

Contents

Bibliography **1**

1 Introduction **1**

2 Tangent Levy Model and Explicit Implied Smile Dynamics **4**
2.1 Preliminary: Exponential Additive Processes 4
2.2 Simplication of Pricing PIDE with Dirac style Lévy measures 6
2.3 No Arbitrage Drift Condition for λ's Dynamics and Model Consistency 8

3 Calibration to Market Call Prices and Implied Smile Simulation **14**
3.1 Discretisation and Numerical Setup 14
3.2 Regularization of Optimization 16
3.3 Parameter Estimation of λ's Dynamics 20
3.4 One Day Ahead Implied Smile Simulation 23

4 Concluding Remarks **26**

A MATLAB Code **27**

Bibliography **34**

Chapter 1

Introduction

Black-Scholes model was the earliest used model for European option pricing (in this thesis all options refer to European options). However, it is an well known fact that at any time, in order to obtain the observed market prices with different maturities and strikes, the volatility numbers used to insert into the Black-Scholes formula are different. It is market practice to quote the option price in terms of the volatility number to fit into the Black-Schoels formula so that the corresponding market price is got. Such volatility is called implied volatility. As to FX options and equity index options, for strikes close to the current price (at the money) the implied volatility is small, and for strikes higher or lower (out of the money or in the money, for call options) the implied volatility is usually bigger. Such phenomenon is called *implied smile*, where the sloping is often symmetric. For equity options, it is more often called *implied skew* where deep out-of-the money side slopes downward. Here we talk about smile only.

If we plot the implied voltility against strikes and maturities for all the available market prices, what we obtain is called *volatility surface*. Local volatility models [6] and stochastic volatility models have been the most popular models used in practice. In the framework of modelling with jumps, the CGMY model [3] by Carr, Geman, Madan and Yor is able to fit the option prices of all maturities and all strikes by calibrating a Lévy density. Up to this point, the modelling for European option pricing seems to be perfect.

But it is not. We are not satisfied with only fitting our model parameters to market prices, because what it could explain is very limited. What is ideal is that we want to be able to say more about the future. We want to give dynamics to our volatility surface. Can we simply give dynamics to our parameters and simulate the future volatility surfaces? The answer is no, because for all the models mentioned above, they assume the parameters are constants. So as long as one calibrates to

the market data and get the parameters, one is not allowed to change them casually. Otherwise that could cause the inconsistency of modelling and even cause arbitrage. Let us look at the following example.

We assume the true model is

$$dS_t = S_t \sigma_t dB,$$

or equivalently

$$S_t = S_0 \exp\left(-\frac{1}{2}\int_0^t \sigma_u^2 du + \int_0^t \sigma_u dB_u\right),$$

under the pricing measure. Process σ_t is arbitrary, satisfying some (classical) regularity conditions. At time zero we notice that since implied volatility surface is flat ($\sigma_0^{IV} = constant$), so all European call and put options have the same prices as in the Black-Scholes model. Since any European payoff h can be replicated by call and put payoffs, we conclude that, at time zero (current time), the price of any European-type claim with payoff $h(S_T)$ is the same as the Black-Scholes price. Now apply this argument to $h(x) = \log((x/S_0))^2$ to conclude that

$$\mathbb{E}\zeta^2 = (\mathbb{E}\zeta)^2 = T^2\sigma_0^4$$

where $\zeta = \int_0^T \sigma_t^2 du$. This implies that the variance of ζ is zero, so it has to be constant, which is the current implied volatility. If some bank uses any model that calibrates the market data and restore the flat volatility surface and put the parameters in motion, of course this implies that they make the volatility stochastic. The arbitrage strategy is that we would sell them options with implied volatility as underlying, for example call options on VIX (VIX is a popular measure of the implied volatility of S&P 500 index options). According to the bank's analysis, they allow the volatility to be stochastic, so they of course believe that the option's value is higher than intrinsic value. Howerver, we have proved that the option should always be the same as intrinsic value. We sell them the options and arbitrage is exploited.

This example gives a counter example for the serious consequence of forgetting Fundemental Theorem for Asset Pricing ([9],FTAP) for pricing. Only if we can find a model in a big framework that not only allows the stochasticity of parameters but also allows us to do risk neutral pricing to fit the market prices can we be confident about the absense of arbitrage. All these thoughts lead to the attempts for constructing market models for vanilla prices, for example in [10],[12] and [7]. Recent works include [11] by Schönbucher and Wissel, and [2] and [1] by Carmona and Nadtochiy.

The difficulty for such market model is that we cannot give dynamics to the implied surface directly, because the space where the implied surface lives is hard to describe and difficult to look after if it is put in motion. There are many conditions to be satisfied for no-arbitrage purpose. For instance, call price needs to be nondecreasing in maturity and strike, and convex in strike. And they are just necessary conditions.

In [2] and [1] Carmona and Nadtochiy presented the idea of finding a code-book in order to linearize the dynamics of implied volatility. In [1] they successfully proved that their *Tangent Lévy Model* admits no arbitrage if and only if the price process generated by the corresponding Lévy density is a martingale. In such Tangent Lévy Model, one is able to both fit the option price and give no arbitrage dynamics to the Lévy density, and thus the implied volatility. The motivation of this thesis is to further develop the practice of this Tangent Lévy Model by specifying a parametrized Lévy measure and constructing a finite dimensional dynamic implied smile model, which could start from any observed smile and give dynamics to it, the details of the construction process is in Chapter 2. In Chapter 3 we give numerical side for the model as to how the prices can be calculated and how our parameters can be calibrated to the market data. For practical purpose, we also proposed a time-homogenous extension to describe k day ahead dynamics of the Lévy measure, so that we are able to get any k day ahead implied smile dynamics for simulation purpose.

Chapter 2

Tangent Levy Model and Explicit Implied Smile Dynamics

In this section we will first introduce the concept of exponential additive process and the pricing of European option under pure jump exponential additive model via Partial Integro-Differential Equation (PIDE). After that, we will assume of a specific parameterised Lévy measure, propose dynamics of that measure via the dynamics of its 'atoms', and derive the drift term of its dynamics under risk neutral measure, which is very similar to the HJM condition in interest rate modelling. Finally we will prove that since the dynamics of its 'atoms' are under risk neutral measure, the produced option prices will admit no arbitrage, and thus we could successfully give dynamics to the implied volatility smile.

2.1 Preliminary: Exponential Additive Processes

Exponential additive processes are important in the setup of our model's dynamics. First introduce the concept of additive processes. Most content of this section could be found in [4].

Definition 1 (Additive process) *A stochastic process $(X_t)_{t\leq 0}$ on $\mathbb{R}^d$ is called an additive process if it is cadlag, satisfies $X_0 = 0$ and has the following properties:*

1. *Independent increments: for every increasing sequence of times $t_0 \dots t_n$, the random variables $X_{t_0}, X_{t_1} - X_{t_0}, \dots, X_{t_n} - Xt_{n-1}$ are independent.*
2. *Stochastic continuity: $\forall \varepsilon > 0$, $\lim_{h\to 0} \mathbb{P}[|X_{t+h} - X_t| \leq \varepsilon] = 0$.*

In other words, additive processes are just Lévy processes without the assumption of stationary increments. Exponential additive processes are based on additive

processes. In fact, as its name indicates, they are the exponential of some additive processes. How is it possible? Let us look at the case of the classical geometric Brownian motion of asset price in the risk neutral measure:

$$S_t = \exp(rt + B_t^0), \text{ where } B_t^0 = -\frac{\sigma^2}{2}t + \sigma W_t. \tag{2.1}$$

Obviously B_t^0 is not a martingale itself, but $\exp(B_t^0)$ is a martingale. So instead of giving dynamics to S_t directly, we could similarly first suppose the form of S_t by means of another process X_t, such that:

$$S_t = S_0 \exp(rt + X_t).$$

If X_t is an additive process (or Lévy process), such model is called an *exponential additive model* (or *exponential Lévy model*).

In our model, we choose X_t to be an additive pure jump process. In fact, the specification of the process of X_t will be done via *Poisson random measure*:

Definition 2 (Poisson random measure) *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $E \subset \mathbb{R}^d$ and μ a given (positive) Radon measure on $(E, \mathcal{E})$. A Poisson random measure on E with intensity measure μ is an integer valued random measure:*

$$\begin{aligned} M : \Omega \times \mathcal{E} &\rightarrow \mathbb{N} \\ (\omega, A) &\mapsto M(\omega, A), \end{aligned}$$

such that

1. *For (almost all) $\omega \in \Omega$, $M(\omega, .)$ is an integer-valued Radon measure on E: for any bounded measurable $A \subset E$, $M(A) < \infty$ is an integer valued random variable.*

2. *For each measurable set $A \subset E$, $M(., A) = M(A)$ is a Poisson random variable with parameter $\mu(A)$:*

$$\forall k \in \mathbb{N}, \quad \mathbb{P}(M(A) = k) = e^{-\mu(A)} \frac{(\mu(A))^k}{k!}.$$

3. *For disjoint measurable sets $A_1, \ldots, A_n \in \mathcal{E}$, the variables $M(A_1), \ldots, M(A_n)$ are independent.*

The process of X_t can now be specified as:

$$\tilde{X_T} = \log \tilde{S_0} - \int_0^T \int_{\mathbb{R}} (e^x - x - 1)\tilde{\eta}(dx, du) + \int_0^T \int_{\mathbb{R}} x(\tilde{N}(dx, du) - \tilde{\eta}(dx, du)),$$

where $\tilde{N}$ is the Radon measure and $\tilde{\eta}$ is its compensator in the above definition. $\tilde{N}(dx, du)$ and $\tilde{\eta}(dx, du)$ are meant to put together because $\int_0^T \int_{\mathbb{R}} x(\tilde{N}(dx, du) - \tilde{\eta}(dx, du))$ is a martingale, which is analogous to the term σW_T of B_T^0 in (2.1). $\tilde{N}(dx, du) - \tilde{\eta}(dx, du)$ altogether is called the *compensated Poisson random measure*.

It could be easily checked that $\tilde{X_T}$ is an additive process and indeed it is time inhomogeneous. Such time inhomogeneity will be neccessary when we specify the form of the compensator $\tilde{\eta}(dx, du)$ later and give it dynamics (which of course means that the compensator has to be allowed to change with time).

Finally, we could write the *exponential additive model* of $\tilde{S_T} = \exp \tilde{X_T}$ (here we assume that the interest rate $r = 0$ for convenience). It could be check that it satisfies following stochastic integral equation:

$$\tilde{S_T} = \tilde{S_0} + \int_0^T \int_{\mathbb{R}} \tilde{S_{u^-}}(e^x - 1)(\tilde{N}(dx, du) - \tilde{\eta}(dx, du)). \tag{2.2}$$

Note that $\tilde{S_T}$ is a martingale if it satisfies the condition (Theorem 25.3 and 25.17 in [8]):

$$\int_0^T \int_{\mathbb{R}} (e^x - 1)\tilde{\eta}(dx, dt) < \infty.$$

The above integrability condition is satisfied because of the specific form of Lévy measure we choose later. Here we emphasize that (2.2) is not the actual model's dynamics that we will study, but rather an artificial model we use to fit to the option prices on each day (basically, we extract the information from the market prices and store it in the model parameters).

2.2 Simplication of Pricing PIDE with Dirac style Lévy measures

As mentioned in the last section, we assume the interest rates are zeros, pricing is linear and we follow the traditional risk neutral approach to do option pricing. The model dynamics is defined by (2.2) and we assume that the compensator of Poisson random measure satisfies:

$$\tilde{\eta}(dx, dT) = \sum_{j=-N}^{N} \lambda_j(T)\delta_{x^j}(dx)dT. \tag{2.3}$$

The market filtration is generated by $\tilde{S}$. We express the option price calculated from our model via conditional expection as:

$$C_t^{\tilde{\eta},S_t}(T,x) = \mathbb{E}\left[\left(\tilde{S_T} - e^x\right)^+ \middle| \tilde{S}_t = S_t\right]. \tag{2.4}$$

Because the compensator $\tilde{\eta}(dx, dT)$ is deterministic and call prices are uniquely determined by the conditional distribution of $(\tilde{S}_u)_{u\in[t,T]}$, given $\tilde{S}_t = S_t$, so the prices are therefore completely dependent on S_t and $\tilde{\eta}$. This justifies the notation $C^{\tilde{\eta},S_t}$.

Having the dynamics of $\tilde{S}_t$ and the pricing expression of (2.4), we could give something similar to the Black-Scholes PDE with some boundary condition. But because our parameters here are logarithm strike x and compensator $\tilde{\eta}$ instead of straightforward strike $K = e^x$ and diffusion volatility σ in the classical Black-Scholes model, what we will get is slightly different. In fact, according to the drivations from [3] or [5], we obtain Partial Integro Differential Equation (PIDE) instead of PDE for call prices:

$$\begin{cases} \partial_T C_t^{\tilde{\eta},S_t}(T,x) = \int_{\mathbb{R}} \psi(T; x-y) D_y C_t^{\tilde{\eta},S_t}(T,y)dy \\ \\ C_t^{\tilde{\eta},S_t}(t,x) = (S_t - e^x)^+, \end{cases} \tag{2.5}$$

where D_x denotes the second order partial differential operator $D_x = \partial^2_{x^2} - \partial_x$ and

$$\psi(T;x) := \begin{cases} \int_{-\infty}^{x} (e^x - e^z) \sum_{j=-N}^{N} \lambda_j(T) \delta_{x^j}(dz) & x < 0 \\ \\ \int_{x}^{\infty} (e^z - e^x) \sum_{j=-N}^{N} \lambda_j(T) \delta_{x^j}(dz) & x > 0, \end{cases} \tag{2.6}$$

is the double exponential tail function introduced in [3]. We will sometimes write $\psi(T;x)$ instead of $\psi(T;x)$ when the function f has two arguments.

The above PIDE can be simplified if we assume that the Lévy measure is in fact a finite linear combination of Dirac measures, in other words:

$$\tilde{\eta}(dx, dT) = \sum_{j=-N}^{N} \lambda_j(T) \delta_{x^j}(dx) dT, \tag{2.7}$$

for some atoms $\{x^j\}$ and a set of weights $\{\lambda_j(T)\}$ (where we set $\lambda_0 = 0$).

Let us compute ψ in this case

$$\psi(T;x) = \begin{cases} \sum_{x^j < x} \lambda^j(T) \left(e^x - e^{x^j}\right) & x < 0 \\ \\ -\sum_{x^j > x} \lambda^j(T) \left(e^x - e^{x^j}\right) & x > 0, \end{cases} \tag{2.8}$$

In order to make $\psi(T;x)$ a continuous function of x, we require

$$\psi(T;0+) = \psi(T;0-) \;\Leftrightarrow \sum_{j=-N}^{N} \lambda^j(T)\left(e^{x^j}-1\right) = 0 \tag{2.9}$$

Thus, throughout this thesis we assume that the right hand side of the above is satisfied. Then $\partial_x\psi$ is well defined and we have

$$\partial_x\psi(T;x) = \begin{cases} e^x \sum_{x^j<x} \lambda^j(T) & x<0 \\ -e^x \sum_{x^j>x} \lambda^j(T) & x>0, \end{cases} \tag{2.10}$$

Then, integrating by parts in (2.5), we obtain the following pricing equation

$$\begin{cases} \partial_T C_t^{\tilde{\eta},S_t}(T,x) = \sum_{j=-N}^{N} \lambda_j(T) e^{x^j} C_t^{\tilde{\eta},S_t}(T,x-x^j) - C_t^{\tilde{\eta},S_t}(T,x) \sum_{j=-N}^{N} \lambda^j(T) \\ \\ C_t^{\tilde{\eta},S_t}(t,x) = (S_t - e^x)^+. \end{cases} \tag{2.11}$$

This equation looks very nice as it is linear with respect to x. In this sense, this equation for pricing is even nicer than Black-Scholes PDE. And in the next chapter this is indeed the motivation of decretising the space of x for the calibration, because if x is discretised we just need to solve a linear ODE of a vector with call prices on different logarithmic strikes at each time point.

2.3 No Arbitrage Drift Condition for λ's Dynamics and Model Consistency

Following [1] we consider the following system

$$\begin{cases} \log S_t = \log S_0 - \int_0^t \int_{\mathbb{R}} \left(e^x - x - 1\right) \eta(dx,du) + \int_0^t \int_{\mathbb{R}} x \left[M(dx,du) - \eta(dx,du)\right], \\ \\ \lambda_t^j(T) = \lambda_0^j(T) + \int_0^t \alpha_u^j(T)du + \sum_{n=1}^{m} \int_0^t \beta_u^{j,n}(T) dB_u^n, \quad j=-N,\dots,N, \quad j\neq 0, \end{cases} \tag{2.12}$$

where $t, T \in [0,\bar{T}]$, for some fixed $\bar{T}>0$; M is an *integer valued random measure* with (stochastic) compensator η (see [1] for the technical conditions they have to satisfy); $B = (B^1,\dots,B^m)$ is a multidimensional Brownian motion; processes α^j and $\beta^{n,j}$ take values in the space of absolutely continuous functions on $[0,\bar{T}]$ with absolutely integrable and square integrable derivatives respectively, which implies that λ^j takes

values in the same space as α^j. In addition, we assume that λ_t satisfies (2.9) for any $t \in [0, \bar{T}]$.

At each moment of time t, we define $C_t^{\lambda_t, S_t}(T, x)$ as the price of a call option with strike e^x and maturity T in a "tangent" model generated by $\left\{\lambda_t^j(.)\right\}_{j\neq 0}$. In other words, $C_t^{\lambda_t}(T, x)$ is defined by (**??**) where "tangent" process $\tilde{S}$ is constructed through $\tilde{\eta}$ in (**??**), which, in turn, is given by $\lambda = \lambda_t$ via (2.7).

We call dynamics (2.12) a **tangent Lévy model** if $\lambda_t^j \geq 0$ at all $t \in [0, \bar{T}]$ almost surely, S is a true martingale and, most importantly, **the call prices produced by λ_t coincide almost surely with the true call prices** given by

$$C_t(T, x) = \mathbb{E}\left(S_T - e^x \middle| \mathcal{F}_t\right), \tag{2.13}$$

for each $t \in [0, T]$.

It was shown in [1] (see Theorem 1 in Section 4) that, in a similar setup, system (2.12) defines a tangent Lévy model if and only if the so-called "consistency conditions" are satisfied. These conditions consist of "drift restriction" and a "compensator specification". The setup of [1] is slightly different (the compensator η and tangent Lévy measure $\tilde{\eta}$ are assumed to be absolutely continuous), however, it is straight forward to find the analogues of the consistency conditions in the present case: "drift restriction" implies that α should be expressed through β in a specific way, and the "compensator specification" states that $\eta(dx, dt)$ has to coincide with

$$\eta(dx, dt) = \sum_{j\neq 0} \lambda_t^j(t)\delta_{x^j}(dx)dt \tag{2.14}$$

Of course, one has to derive the exact formulation of the "drift condition" in the present case, since it will not coincide exactly, at least formally, with the one given in [1]. However, it is not hard to shown that the compensator specification in the present setup coincides with the analogue of compensator specification in [1], formulated above. And since our goal is not to provide the most geenral characterization of the consistency of dynamics (2.12) but rather to construct and implement a large class of tangent Lévy models, we will not prove that (2.14) is necessary but we will assume it holds. In this case, the dynamics (2.12) become

$$\begin{cases} \log S_t = \log S_0 + \sum_{j\neq 0} x^j N^j_{\int_0^t \lambda_u^j(u)du}, \\ \lambda_t^j(T) = \lambda_0^j(T) + \int_0^t \alpha_u^j(T)du + \sum_{n=1}^m \int_0^t \beta_u^{j,n}(T)dB_u^n, \quad j = -N, \ldots, -1, 1, \ldots, N, \end{cases} \tag{2.15}$$

where all processes are as defined in (2.12), except for N^j's which are Possion processes with jumps of size 1 and intensity 1, independent of each other and of α, β and B.

As discussed in [1] (see Section 5), the above specification will also guarantee that S is a true martingale. Therefore, we only need to characterize the "consistency" of (2.15) (namely, when C^{λ_t} coincides with the true call prices) and make sure that λ stays nonnegative and (3.2) is satisfied.

First, we address the issue of consistency. Introduce

$$\Delta_t(T,x) = -\partial_x C_t^{\lambda_t}(T,x) \tag{2.16}$$

Then in the notation $\lambda_0(T) = -\sum_{j\neq 0, -N\leq j\leq N} \lambda_t^j(T)$ and $x^0 = 0$, differentiating both sides of (2.11) we obtain

$$\begin{cases} \partial_T \Delta_t(T,x) = \sum_{j=-N}^{N} \lambda_t^j(T) e^{x^j} \Delta_t(T, x-x^j) \\ \\ \Delta_t(t,x) = e^x \mathbf{1}_{(-\infty, \log S_t]}(x) \end{cases} \tag{2.17}$$

Now introduce

$$\hat{\Delta}_t(T,\xi) = \int_{\mathbb{R}} e^{-2\pi i x\xi} \Delta_t(T,x) dx, \tag{2.18}$$

and equation (2.17) yields

$$\begin{cases} \partial_T \hat{\Delta}_t(T,\xi) = \sum_{j=-N}^{N} \lambda_t^j(T) e^{x^j - 2\pi i x^j \xi} \hat{\Delta}_t(T,\xi) \\ \\ \hat{\Delta}_t(t,\xi) = \frac{e^{\log S_t(1-2\pi i\xi)}}{1-2\pi i\xi}, \end{cases} \tag{2.19}$$

and hence

$$\hat{\Delta}_t(T,\xi) = \frac{e^{\log S_t(1-2\pi i\xi)}}{1-2\pi i\xi} \exp\left(\sum_{j=-N}^{N} \int_t^T \lambda_t^j(u) du e^{x^j - 2\pi i x^j \xi} \right) \tag{2.20}$$

Let us now recall some of the results of [1]. First of all, Proposition 1 in Section 4 states that consistency of the model is equivalent to $C_t^{\lambda_t}(T,x)$ being martingales in $t \in [0,T)$. In fact even local martingale property is enough, since $C_t^{\lambda_t}$ is bounded by S_t which is a uniformly integrable martingale.

Next, one needs to repeat the first part of the proof of Theorem 1 in [1] to see that $C_t^{\lambda_t}$ is a local martingale if and only if $\hat{\Delta}_t$ is (this is not hard to understand intuitively, since $\hat{\Delta}_t$ is obtained from $C_t^{\lambda_t}$ via a linear operator).

Thus, we only need to find when $\hat{\Delta}_t(T,\xi)$ is a local martingale in $t \in [0,T)$ for all $T \in (0,\bar{T}]$ and $\xi \in \mathbb{R}$. We can use Itô's formula to obtain a semimartingale representation of $\hat{\Delta}_t$

$$
\begin{aligned}
d\hat{\Delta}_t(T,\xi) &= \hat{\Delta}_t(T,\xi)\left[-\sum_{j=-N}^{N}\lambda_t^j(t)e^{x^j-2\pi i x^j\xi}\right.\\
&+ \sum_{j\neq 0}\lambda_t^j(t)\left(e^{x^j(1-2\pi i\xi)}-e^{x^j}(1-2\pi i\xi)-2\pi i\xi\right)\\
&+ \sum_{j=-N}^{N}\int_t^T \alpha_t^j(u)du e^{(1-2\pi i\xi)x^j}\\
&+ \left.\frac{1}{2}\sum_{n=1}^{m}\left(\sum_{j=-N}^{N}\int_t^T e^{(1-2\pi i\xi)x^j}\beta_t^{j,n}(u)du\right)^2\right]dt+(\ldots)dM_t\\
&= \hat{\Delta}_t(T,\xi)\left[\sum_{j=-N}^{N}\int_t^T \alpha_t^j(u)du e^{(1-2\pi i\xi)x^j}\right.\\
&+ \left.\frac{1}{2}\sum_{n=1}^{m}\left(\sum_{j=-N}^{N}\int_t^T e^{(1-2\pi i\xi)x^j}\beta_t^{j,n}(u)du\right)^2\right]dt+(\ldots)dM_t,
\end{aligned}
$$

where M is a local martingale. In the above we applied Itô's formula in a way analogous to the derivation of Corollary 2 in Section 4 of [1], and we used (3.2) to obtain the last equality.

Setting the drift to zero we obtain:

$$
\sum_{j=-N}^{N}\alpha_t^j(T)e^{(1-2\pi i\xi)x^j} = -\sum_{n=1}^{m}\left(\sum_{j=-N}^{N}e^{(1-2\pi i\xi)x^j}\beta_t^{j,n}(T)\right)\left(\sum_{j=-N}^{N}e^{(1-2\pi i\xi)x^j}\int_t^T\beta_t^{j,n}(u)du\right)
$$

Now we need to expand the product of series on the right hand side and equate the terms with the same exponential. However, before we do that we need to be careful because the index of j will go beyond the range of $-N$ to N. In order to get perfect equality, we assume that $\beta_t^{j,n}=0,\ \ \forall j>N/2, j<-N/2$. This assumption makes sense because when $|j|$ is big, the dynamics of $\lambda_t^j(T)$ should not be significant, so it is reasonable to ignore their stochasticity and set $\beta_t^{j,n}=0$. Now we can derive the drift condition for α_t^j :

$$
\begin{aligned}
\sum_{j=-N}^{N}\alpha_t^j(T)e^{(1-2\pi i\xi)x^j} &= -\sum_{n=1}^{m}\left(\sum_{j=-N/2}^{N/2}e^{(1-2\pi i\xi)x^j}\beta_t^{j,n}(T)\right)\left(\sum_{j=-N/2}^{N/2}e^{(1-2\pi i\xi)x^j}\int_t^T\beta_t^{j,n}(u)du\right)\\
&= -\sum_{n=1}^{m}\sum_{j_1+j_2=j}\beta_t^{j_1,n}(T)\int_t^T\beta_t^{j_2,n}(u)du\left(e^{(1-2\pi i\xi)x^j}\right)
\end{aligned}
$$

Since $(e^{(1-2\pi i\xi)x^j})_{j=-N,\dots,N}$ are linearly independent, therefore the above equation is true if and only if:

$$\alpha_t^j(T) = -\sum_{n=1}^{m} \sum_{j_1+j_2=j} \beta_t^{j_1,n}(T) \int_t^T \beta_t^{j,n}(u)du, \tag{2.21}$$

which looks very similar to HJM drift condition in interest rate modelling. For notational convenience, we set $\bar{\beta}_t^{j,n}(T) := \int_t^T \beta_t^{j,n}(u)du$.

Recall that we also need $\sum_{j=-N}^{N} \lambda_t^j(T) = 0$. If the initial value λ_0 satisfies this condition, then it will be preserved by the dynamics if $\sum_{j=-N}^{N} \beta_t^{j,n}(T) = 0$ and $\sum_{j=-N}^{N} \alpha_t^j(T) = 0$. Thus, we assume that

$$\sum_{j=-N/2}^{N/2} \beta_t^{j,n}(T) = 0.$$

The above implies that $\sum_{j=-N/2}^{N/2} \bar{\beta}_t^{j,n}(T) = 0$ due to linearity. As for $\alpha_t^j(T)$:

$$\begin{aligned}
\sum_{j=-N}^{N} \alpha_t^j(T) &= -\sum_{j=-N}^{N} \sum_{n=1}^{m} \sum_{j_1+j_2=j} \beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T) \\
&= -\sum_{n=1}^{m} \sum_{j_1=-N/2}^{N/2} \sum_{j_2=-N/2}^{N/2} \beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T) \\
&= -\sum_{n=1}^{m} \left(\sum_{j_1=-N/2}^{N/2} \beta_t^{j_1,n}(T) \right) \left(\sum_{j_2=-N/2}^{N/2} \bar{\beta}_t^{j_2,n}(T) \right) \\
&= 0.
\end{aligned}$$

In addition, we need the symmetry condition $\sum_j (e^{x^j} - 1)\lambda_t^j(T) = 0$ to be satisfied. Therefore, we assume

$$\sum_j (e^{x^j} - 1)\beta_t^{j,n}(T) = 0$$

Similarly, $\sum_j (e^{x^j} - 1)\bar{\beta}_t^{j,n}(T) = 0$ also naturally follows from the above, and we just need to prove this equality for $\alpha_t^j(T)$:

$$\begin{aligned}
\sum_j (e^{x^j} - 1)\alpha_t^j(T) &= -\sum_j (e^{x^j} - 1) \sum_{n=1}^{m} \sum_{j_1+j_2=j} \beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T) \\
&= -\sum_{n=1}^{m} \sum_j \sum_{j_1+j_2=j} (e^{x^{j_1+j_2}} - 1)\beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T)
\end{aligned}$$

$$
\begin{aligned}
&= -\sum_{n=1}^{m}\sum_{j}\sum_{j_1+j_2=j}\Big[(e^{x^{j_1}}-1)(e^{x^{j_2}}-1)\\
&\quad + (e^{x^{j_1}}-1)+(e^{x^{j_2}}-1)\Big]\beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T)\\
&= -\sum_{n=1}^{m}\sum_{j_1=-N/2}^{N/2}\sum_{j_2=-N/2}^{N/2}\Big[(e^{x^{j_1}}-1)(e^{x^{j_2}}-1)\\
&\quad + (e^{x^{j_1}}-1)+(e^{x^{j_2}}-1)\Big]\beta_t^{j_1,n}(T)\bar{\beta}_t^{j_2,n}(T)\\
&= -\sum_{n=1}^{m}\sum_{j_1}(e^{x^{j_1}}-1)\beta_t^{j_1,n}\sum_{j_2}(e^{x^{j_2}}-1)\bar{\beta}_t^{j_2,n}\\
&\quad -\sum_{n=1}^{m}\sum_{j_1}(e^{x^{j_1}}-1)\beta_t^{j_1,n}\sum_{j_2}\bar{\beta}_t^{j_2,n}\\
&\quad -\sum_{n=1}^{m}\sum_{j_1}\beta_t^{j_1,n}\sum_{j_2}(e^{x^{j_2}}-1)\bar{\beta}_t^{j_2,n}\\
&= 0.
\end{aligned}
$$

To this point we can conclude that the $\lambda_t^j(T)$ coming from our specified dynamics will still satisfy the condition

$$\sum_{j=-N}^{N}\lambda_t^j(T)=0, \quad \text{and (2.9)}.$$

Chapter 3

Calibration to Market Call Prices and Implied Smile Simulation

In this chapter, first we will present numerical experiments that calibrate to USD/YEN currency options. Although this is static fitting, which is not our main focus, it may still be worth mentioning that our discretisation of pricing equation (2.4) enables us to calculate the call prices from any λ's more easily and approach the calibration more directly. After that we will estimate the parameters of dynamics of λ's, and thus obtain the dynamics of implied smile. Finally, the one day ahead implied smile will be simulated. And most importantly, remember that those implied smiles are within our big framework of tangent levy model, so they are arbitrage free.

3.1 Discretisation and Numerical Setup

If we now discretize the x-space, instead of a function $C_t(T,.)$ we can deal with a vector

$$\left(V^j(T) = C_t(T, \log S_t + x^j) - S_t(1 - e^{x^j})^+\right)_{j=-N}^{N}.$$

Here we focus on the time value, which is the option price over intrinsic value (recall the well known fact that European call prices with no dividend are always higher than intrinsic value), because we are calibrating to short-maturity call prices and their intrinsic values are small. Time value and full option price may be in different magnitudes, so the error may be large if we do not extract the time value from the option price and do the calibration.

As to x_j we choose them to be equidistantly distributed with zeros in the middle:

$$x^j = \frac{j}{N}M, \quad j = -N, \ldots, N,$$

where $[-M, M]$ is the range of x^j and $1/N$ is the distance in betwee. Under this setup then the system (2.11) can be approximated by

$$\begin{cases} \partial_T V^k(T) = \sum_{j=-N}^{N} \lambda_j(T) e^{x^j} V^{k-j}(T) - V^k(T) \sum_{j=-N}^{N} \lambda_j(T) \\ -S_t(1-e^{x^k})^+ \sum_{j=-N}^{N} \lambda_j(T) + S_t \sum_{j=-N}^{N} \lambda_j(T) e^{x^j}(1-e^{x^{k-j}})^+, \quad k=-N,\dots,N \\ V^k(T) = 0, \quad k < -N \text{ and } k > N, \\ V^k(t) = 0, \quad k = -N,\dots,N \end{cases} \tag{3.1}$$

As in the last chapter, for notational convenience, we now set $\lambda_0(T) = -\sum_{j\neq 0, -N\leq j\leq N} \lambda_j(T)$. Recall that we still assume that

$$\sum_{j\neq 0} \lambda^j(\exp(x^j) - 1) = 0. \tag{3.2}$$

This equation is in fact an important constraint to our calibration and cannot be ignored.

System (3.1) is an evolution equation for vector V and can be written in the matrix form

$$\begin{cases} \partial_T V(T) = A(T)V(T) + b(T) \\ V(t) = 0, \end{cases} \tag{3.3}$$

where $A = (a_{ij})_{i,j=-N}^{N}$, with $a_{ij} = \mathbf{1}_{i-j\geq -N}\mathbf{1}_{i-j\leq N}\lambda_{i-j}(T)e^{x^{i-j}}$,and $b_i = S_t \sum_{j=-N}^{N} \lambda_j(T)$ $e^{x^j}(1-e^{x^{i-j}})^+$.

Since we would like to restrict ourselves to the case of single short maturity, $T - t \ll 1$, for the purpose of numerical implementation it is reasonable to consider $\lambda^j(u)$'s which are constant as functions of $u \in [t, T]$. In this case, $A(u)$ and $b(u)$ do not actually depend upon u, and solution to (3.3) is given by the *matrix exponential*:

$$V(T) = \left(\exp\left((T-t)A\right) - I\right) A^{-1} b. \tag{3.4}$$

Note that (3.4) could actually be further simplified with Taylor expansion:

$$\begin{aligned} V(T) &= \left(\exp\left((T-t)A\right) - I\right) A^{-1} b \\ &= \left((T-t)A + \frac{(T-t)^2}{2}A^2 + \cdots\right) A^{-1} b \\ &= (T-t)\left(I + \frac{T-t}{2}A + \frac{(T-t)^2}{6}A^2 + \cdots\right) b. \end{aligned}$$

This expansion is not hard to implement. It will help improve the efficiency and will avoid the possible singularity problem of solving $A^{-1}b$ if we directly evaluate (3.4).

Recall that $A = A^\lambda$ and $b = b^\lambda$ depend only on our choice of $\lambda = \{\lambda_j\}_{j\neq 0}$. Denote the market data (which, in the present case, is the set of observed call prices) by

$$\bar{V} := \left\{ C_t^{mkt}(T, \log S_t + x^{k_j}) - S_t(1 - e^{x^{k_j}})^+ \right\}_{j\in\mathcal{J}}, \tag{3.5}$$

where $\left\{S_t \exp\left(x^{k_j}\right)\right\}_{j\in\mathcal{J}}$ are the strikes of options available on the market. And the calibration problem becomes

$$\min_{\lambda\geq 0,\, \langle\lambda, e\rangle = 0} \left\| \left(\exp\left((T-t)A^\lambda\right) - I\right) \left(A^\lambda\right)^{-1} b^\lambda - \bar{V} \right\|_{\mathcal{J}}, \tag{3.6}$$

where the norm $\|.\|_{\mathcal{J}}$ can be chosen in many different ways: for example, it can be a sum of squares of the entries that correspond to the subindex $\mathcal{J}$. One thing we could improve here is to change the (3.6) because when we take into consideration the bid-ask spread, it might be a better idea to give different weights to prices with different strikes by minimising the following instead:

$$\min_{\lambda\geq 0,\, \langle\lambda, e\rangle = 0} \left(\sum_{j\in\mathcal{J}} \omega_j |V^k - \bar{V}^k|^2 \right), \quad \text{where } \omega_j = \frac{1}{|\bar{V}_j^{bid} - \bar{V}_j^{ask}|^2}. \tag{3.7}$$

Options with different strikes have different liquidity, and thus different importance. The above optimization is able to give more weights to more liquid options, which have smaller bid-ask spread.

It is an illusion that we are done now, because intuitively as long as we have introduced enough number of λ's we should always be able to perfectly fit all the market prices, but that is unfortunately not true. Because the optimization (3.6) is ill-posed and thus might not allow us to find its global minimum easily. In fact, we may merely get local minimum if we try it straight forward. In order to make our optimization more stable, we need to do some regularization.

3.2 Regularization of Optimization

As we mentioned in the end of last section, because of the high non-linearity of the nature of λ's influence on the target function, straight forward treatment of optimization (3.6) will very probably cause local minimum and might not be satisfactory. One popular way to tackle this problem is to add a *penalisation* term, which is usually a

convex function. Therefore, the problem will turn into minimising the sum of (3.7) and another convex penalisation term:

$$\min_{\lambda\geq 0,\, \langle \lambda, e\rangle = 0} \left(\sum_{j\in\mathcal{J}} \omega_j |V^k - \bar{V}^k|^2 \right) + \alpha F(\lambda) \tag{3.8}$$

It is known that a convex optimization problem with linear constraints always have unique global minimum solution, so as long as our constant α is large enough, the optimization will be forced to have unique and stable solution, which cures the ill-posedness problem. However on the other hand, α cannot be too large as it could harm our original aim to fit the call prices. What we know is that usually the first term of (3.8) decreases with α, so we could set the tradeoff between optimization stability and fitting accuracy by setting α to be:

$$\alpha^* = \sup\{\alpha : \sum_{j\in\mathcal{J}} \omega_j |V^k - \bar{V}^k|^2 \leq \varepsilon_0\},$$

where ε is a predefined endurance of accuracy. In practice, we could find reasonable α^* by giving an initial guess and then run a few steps by Newton's method or bisection method (refer to the MATLAB in the Appendix), so that we will start with some α^* that will generate comparably stable solution, and the fitting accuracy is acceptable.

Then we arrive at the question of how to choose the penalization function $F(\lambda)$. There are several things we may want to consider:

1. Since we want to obtain regular enough solutions λ vanishing for large $|x^j|$, it may be a good idea to add a penalisation term

$$\varepsilon_1 \sum_{j=-N}^{-N_1} |\lambda_j| + \varepsilon \left(\sum_{j=-N_1+1}^{N_2} |\lambda_j - \lambda_{j-1}|^n \right)^{\frac{1}{n}} + \varepsilon_2 \sum_{j=N_2}^{N} |\lambda_j| \,,$$

   where $\varepsilon_1, N_1, \varepsilon, n, \varepsilon_2$ can all be specified flexibly.

2. To prepare for giving dynamics to λ's, we may want the shape to look 'similar', so that when we do simulations it could be reasonably expected that the future λ's could be described by some curve around the current one. Therefore, we might add the following penalisation term:

$$\varepsilon_3 \sum_{j=-N}^{-N} \left| 1 - \frac{\lambda_j}{\tilde{\lambda}_j} \right|, \quad \text{where } \tilde{\lambda} \text{ is the prior.}$$

   Note that it is also a convext function with respect to λ. One special effect of this term is that it allows λ^j's to move in different scales: for example, if the

corresponding prior $\tilde{\lambda}^j$ is close to zero, λ^j will also be forced to be close to zero. This is good for our desired shape consistency.

An alternative of the second one above for regularization is by adding the so-called *relative entropy*, in order to find the *minimal entropy measure*, but its idea is similar to our second consideration. In our specific case, our penalisation is the combination of the above two penalisation terms together.

Besides the regularisation, there are still a few details wee need to figure out or assume. For example, to start the optimization, a priori guess is needed. One may want to set certain values of λ_j's to be close to zero: it is a good idea to allow only for jumps of small magnitude, which means that only λ_j's with small enough absolute value of j should be considered as the optimization variables, and the rest is set to some small numbers, not neccessarily zero because later when we want do simulate the future λ's, zero might give us trouble because the drift term α^j could be negative.

The choice of parameters M and N is made according to the following considerations. Number M corresponds to truncation of the space domain, so, in principle, it should be large, however, a reasonable trade-off with the computational complexity is needed. For example, one can choose M so that the corresponding increments of the underlying $|S_{t+T} - S_t|$ never exceed M, for t changing in a long enough time interval. However, since we are not dealing with physical measure and intensity values do not hold under different equivalent martingale measures, we might want to further extend the range a bit for M so that we can describe the tail behaviour better. The "size of the grid" N should be chosen, again, as large as possible, but so as to keep a reasonable computational speed.

it also makes sense to assume that $\lambda_j \approx 0$ whenever $|j|$ is large enough (jumps cannot be too large). The exact level of truncation depends on the choice of time-scale: after all we have to agree what we understand by the "typical" jumps, at least over which period the "elementary" jumps are considered. Once the rough bounds on the magnitude of the jumps has been set, the initial value of λ (to feed into the minimizer) should be chosen so as to be consistent with the historic square variation of the underlying.

The data we deal with is PHLX Currency Option Database. The option is 4-day-to-mature European JPY/USD call option from February 1995 to December 1997 with always three strikes: one in the money, one at the money, and one out of the money. There is a maturity day in every month, so overall there are 33 series and 99 prices to fit. The paremeters we select are: $M = \log 1.1, N = 8, N_1 = N_2 =$

$N-1, \varepsilon_1 = \varepsilon_2 = 0.1, n = 2, \varepsilon = 0.02, \varepsilon_3$ is adapted by the use of bisection method as mentioned before. We use the MATLAB function `@fmincon` to do the optimization. The linear constraint we set include of course include the positivity of λ^j's (in fact, we set $\lambda^j > 0.1$) and the symmetry condition (2.9). Correspondingly we get 33 series of λ's and they are plotted together in Figure 3.1. They all behave bell-shaped quite consistently and the decay can be seen in the tails, which is what we desire. Note that although they are plotted as curves, λ^j are afterall parameters of the Dirac style measure we specified, not the density function for the Lévy measure.

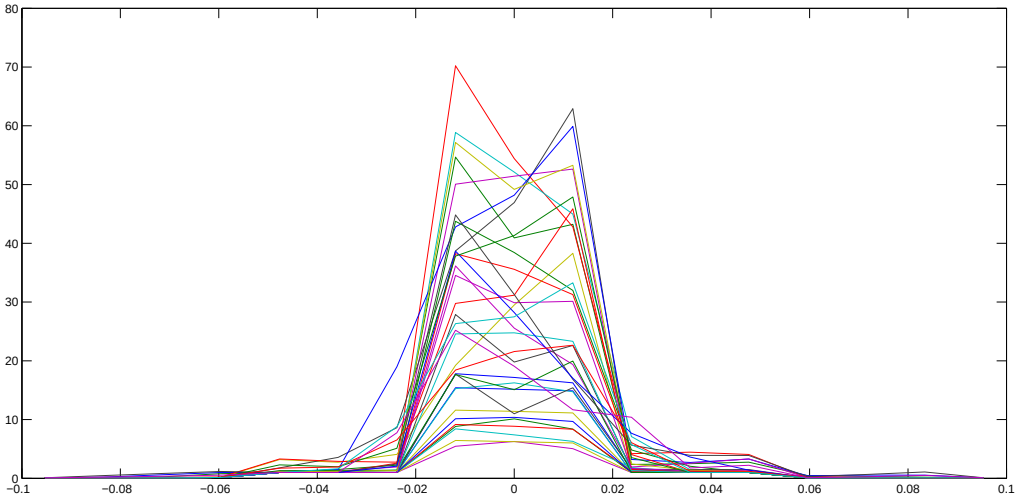


Figure 3.1: 33 series of λ's calibrated to JPY/USD 4-day-to-mature European call option prices

To visualize how they fit the historical prices of different strikes, see Figure 3.2, Figure 3.3 and Figure 3.4. It can be seen that most of the time, our fitting is between the bid-ask spread. We emphasize here again that our aim is to find the no arbitrage dynamics of implied smile, not prefectly fitting all the call prices, so the precision of fit can be sacrified to a certain extent in order to obtain more stable dynamics. But the λ^j (the 31th one) we use later for starting our simulation perfectly fit the medium of bid and ask prices.

It is also worth mentioning that our symmetry condition (2.9) is meanwhile also satisfied. Besides, in our case we actually add another symmetry condition:

$$\sum_{j=-N/2}^{N/2} \lambda^j(T) \left(e^{x^j} - 1 \right) = 0$$

and force it to be satisfied all the time. The reason will be mentioned in the next section.

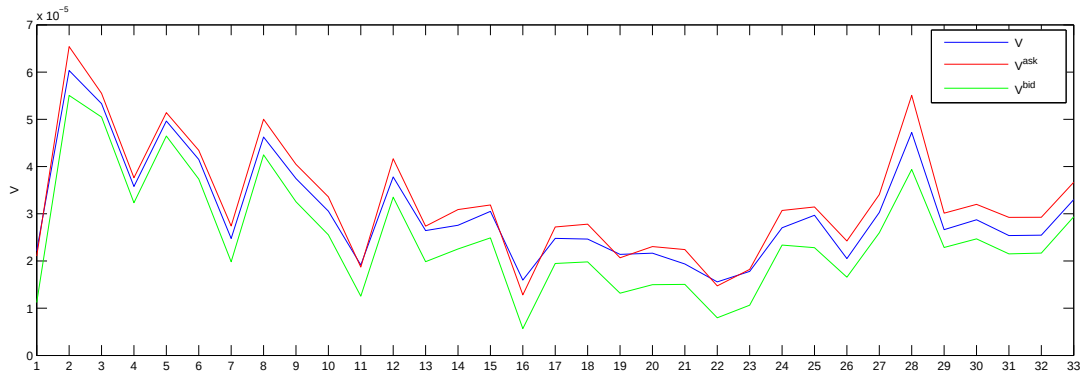


Figure 3.2: Fitting to time value of JPY/USD out of the money European call prices. The area between green and red lines is the bid-ask spread. Same in the following figures.

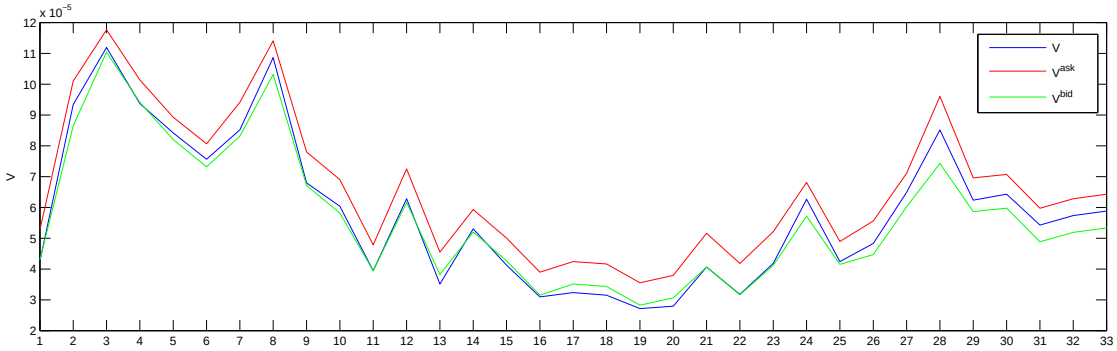


Figure 3.3: Fitting to time value of JPY/USD at the money European call prices.

3.3 Parameter Estimation of λ's Dynamics

Since we have got the time series of λ's, now we can start estimating the parameters α^j and $\beta^{j,n}$ in the dynamics of λ's.

The first thing we need to do is to find m in (2.15). In other words, we need to determine the number of factors to describe the dynamics of $\lambda^j, j = -N, \ldots, N$. If we discretise (2.15) we will get:

$$\lambda^j_{t+\Delta t} - \lambda^j_t = \alpha^j \Delta t + \sum_{n=1}^{m} \beta^{j,n} B_{\Delta t},$$

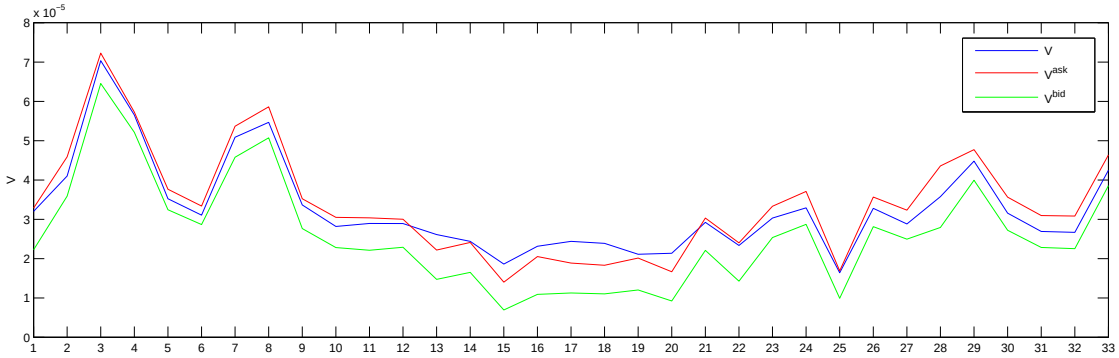


Figure 3.4: Fitting to time value of JPY/USD in the money European call prices.

where we ignore the dependence on T for all the variables because $T-t$ is assumed small. Besides, because in short period of time the brownian motion diffusion term is responsible for higher magnitude of movement than the drift term, so our question turns into "how many indenpendent factors could account for the variance of $\lambda_{t+\Delta t} - \lambda_t$? It seems that Principle Component Analysis (PCA) is able to help us answer this question.

Suppose there is a random vector V (this is the time series of $\Delta\lambda$ in our case) with covariance matrix Σ, then what PCA does is that it will find an orthogonal matrix A, such that A^TV has independent entries, and $A\Sigma A^T$ is its covariance matrix, which is diagonal. So we conclude that $V = A\xi$, where ξ is a vector of independent normals, which is the vector of increments of Brownian motions normalized according to the eigenvalues. In other words, after the transform we are able to separate the different souces of independent factors and identify their significance. What is usually done is that we choose the first m entries in the diagonal of V and they sum up to about 90 percent of the sum of all entries of the diagonal.

Now if we take $V_j = \lambda^j_{t+\Delta t} - \lambda^j_t$ and use the MATLAB function `@princomp` then PCA will return us the eigenvectors $A = [A_1, A_2, A_3, \cdots, A_{2N+1}]$ and eigenvalues in V_{ii}. And PCA shows that our first two eigenvectors already account for about 97 percent of all the eigenvalues, so it is sufficient to set $m = 2$ and use only two independent brownian motions. If we compare what we have now and the form of our dynamics (2.15) we could propose

$$d\begin{pmatrix} \lambda^1_t \\ \vdots \\ \lambda^{2N+1}_t \end{pmatrix} = \begin{pmatrix} \alpha^1_t \\ \vdots \\ \alpha^{2N+1}_t \end{pmatrix} dt + \sigma \sum_{n=1}^{2} A_n (\Lambda_{nn})^{\frac{1}{2}} dB^n_t. \tag{3.9}$$

In order to estimate σ, we use the following relationship:

$$\begin{aligned}\sum_{i=1}^{2N+1} \mathrm{var}(V_j) &= \sigma^2 \Delta t \sum_{i=1}^{2N+1} \left(\Lambda_{11}(A_1)_i^2 + \Lambda_{22}(A_2)_i^2\right) \\ &= \sigma^2 \Delta t \left(\Lambda_{11} \sum_{i=1}^{2N+1} (A_1)_i^2 + \Lambda_{22} \sum_{i=1}^{2N+1} (A_2)_i^2\right) = \sigma^2 \Delta t (\Lambda_{11} + \Lambda_{22})\end{aligned}$$

Therefore,

$$\sigma = \left(\frac{\sum_{i=1}^{2N+1} \mathrm{var}(V_i)}{\Delta t(\Lambda_{11} + \Lambda_{22})}\right)^{\frac{1}{2}}.$$

And finally we get $\beta^{j,n} = \sigma(\Lambda_{nn})^{\frac{1}{2}}(A_n)^j + \sigma(\Lambda_{nn})^{\frac{1}{2}}(A_n)^j, n = 1, 2.$ $(\beta^{j,1})_j$ and $(\beta^{j,2})_j$ are called *factor loadings* as they explain the weights that different variables contribute to the variability in the whole picture. They are shown in Figure 3.5. It is interesting to observe that the first loading factor indicates that the biggest source of movement come from the λ^j where j is closely around zero, and usually those λ^j move in the same direction. The second loading factor implies that the second significant movement is also λ^j with small $|j|$, but this loading factor shows a different behaviour: $(\lambda^j)_{j>0}$ and $(\lambda^j)_{j<0}$ move in different directions, and thus has a twisting effect on the shape of the peak. Note that because PCA is linear operator, the eigenvectors and thus the loading factors satisfy the (2.9) condition too. Besides, recall the extra symmetry (3.2) we added, the truncated loading factors also satisfy it. In fact, it could been seen that the decay condition we put on $\beta^{j,n}$ where $j > N/2$ makes sense in that the two quarters on both sides decay apparently. Those facts are important to simulation of λ's in the next section.

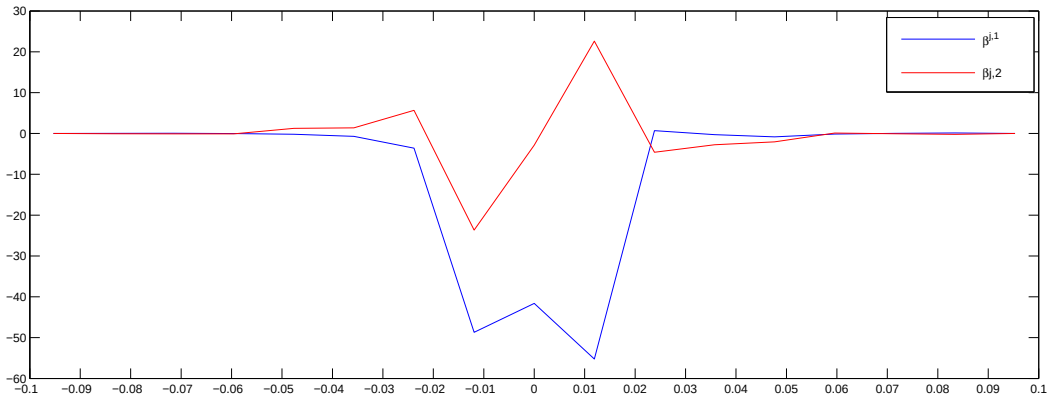


Figure 3.5: The two most significant factor loadings of λ's

3.4 One Day Ahead Implied Smile Simulation

Recall that if we ignore the dependence on T ($T - t \ll 1$) and assume $\alpha_t^j(T)$ and $\beta_t^{j,n}(T)$ are different constants for different j and n, the dynamics we pose on λ_t^j becomes:

$$d\lambda_t^j = \alpha^j dt + \sum_{n=1}^{m} \beta^{j,n} dB_t^n, \quad j = -N, \ldots, N.$$

However before we apply the drift condition to obtain the values of α^j, remember the assumption we made on $\beta^{j,n}$: $\beta^{j,n} = 0, \forall j > N/2, j < -N/2$. We need to be careful if we want to truncate $\beta^{j,n}$ in this way, because this may violate the symmetry condition (2.9) generated from the above SDE. But we have taken care of this already and that is why we add the extra symmetry condition (3.2) on the truncated λ^j. Then we can safely truncate the $\beta^{j,n}$ and apply the drift condition (2.21) the get the α^j.

We choose November 1997 to start our simulation from and we will simulate the λ^j on the next day. Figure 3.6 shows the resulting λ^j with 1000 simulations. Having the simulated λ^j we could calculate the time value and of course the intrinsic value to get the option prices. Then we could simply invert the Black-Scholes formula and get the implied volatility smile. Here we use the MATLAB function `@blsimpv` to realize that. The simulated smiles are shown in Figure 3.7.

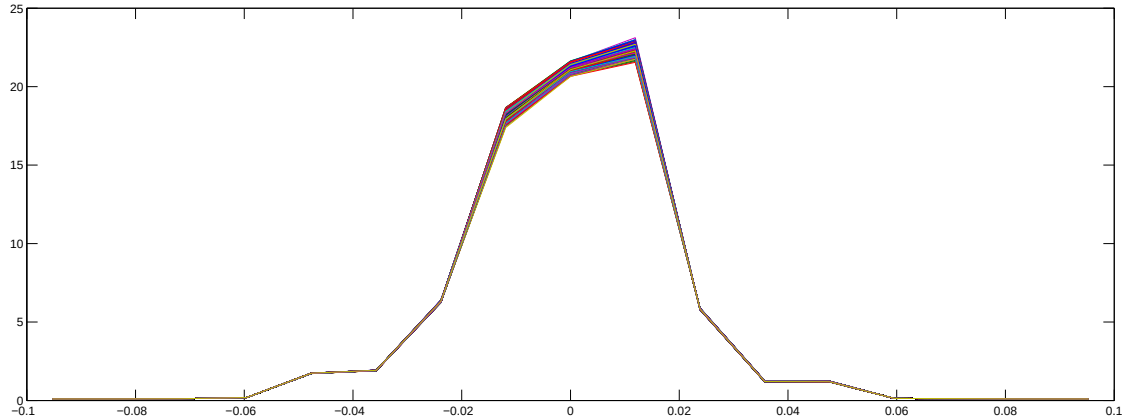


Figure 3.6: Simulated λ's against x. Starting from the λ's obtained from the 4-day-to-mature JPY/USD call option in October 1997. 1000 paths have been simulated.

It seems that we are restricted because we may have assumed too much in the beginning of the section. There is one improvement we could do. Instead of setting all $\alpha_t^j(T)$ and $\beta_t^{j,n}(T)$ to be constant, we could let them depend on $\tau = T - t$. And

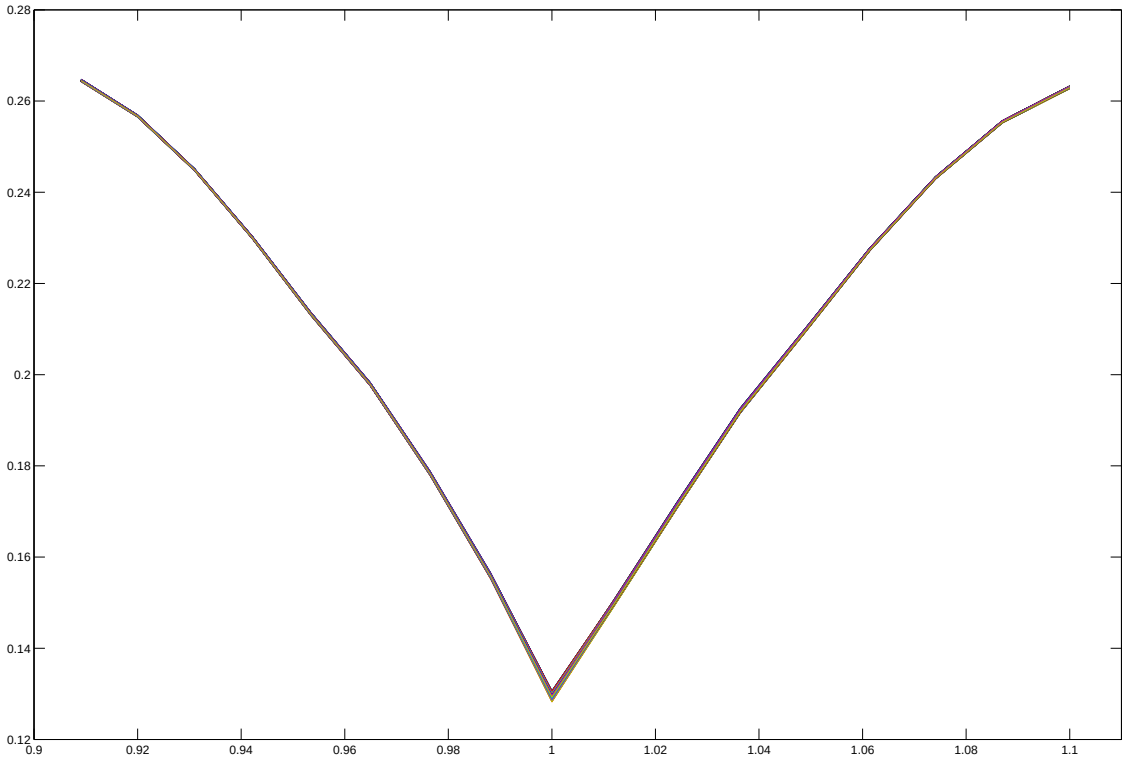


Figure 3.7: One day ahead simulated smiles against $\frac{K}{S_t}$. They look natural as European FX options usually have symmetric implied smiles.

we could give a time-homogeneous dynamics to $\lambda_t^{j,\tau} := \lambda_t^j(t+\tau) = \lambda_t^j(T)$ as follows:

$$d\lambda_t^{j,\tau} = \alpha^{j,\tau}dt + \sum_{n=1}^{m} \beta^{j,n,\tau} dB_t^n, \quad j = -N, \ldots, N.$$

In this framwork, our previous example is just for the case where $\tau = 4$ days. In general we could do similar things as the above precedure to options with maturity in $\tau = k, k-1, \ldots, k-n$ days. Correspondingly we will get $\alpha^{j,\tau}, \beta^{j,n,\tau}$ for $\tau = k, k-1, \ldots$. In order to get k days ahead simulation, notice that we have the following recursion if we do Euler approximation:

$$\begin{aligned} \lambda_{t+k}^j(T) &= \lambda_{t+k}^{j,\tau-k} = \lambda_{t+k-1}^{j,\tau-k+1} + \alpha^{j,\tau-k}\Delta t + \sum_{n=1}^{m} \beta^{j,n,\tau-k} \Delta B_{t_l}^n \\ &= \cdots = \lambda_t^{j,\tau} + \sum_{l=0}^{k} \alpha^{j,\tau-l}\Delta t + \sum_{l=0}^{k}\sum_{n=1}^{m} \beta^{j,n,\tau-l} \Delta B_{t_l}^n \end{aligned}$$

where $\lambda_t^{j,\tau} = \lambda_t^j(T)$ is known, $dB_t^{l,n} = B_{dt}^{l,n}$ are all independent normal random variables and every step dt stands for one day. Under this setup, our model is no longer restricted to short maturity options and we are able to simulate any k day-ahead

implied smiles. In fact, $\alpha^{j,\tau}, \beta^{j,n,\tau}$ with different τ might have different behaviour, for example the loading factors $\beta^{j,n,\tau}$ we obtained might have different behaviour from the case when $\tau = 4$ days. They may make very diversified contribution to the shape of $\lambda_t^j(T)$, which makes sense: we cannot really expect that the shape of $\lambda_t^j(T)$ k days ahead are only allowed to change by adding or substracting two vectors multiplied by some constant. We give only numerical results for one day ahead simulated smiles just for illustration here and do not do the generalized k day-ahead simulation.

Chapter 4

Concluding Remarks

In this thesis, we parametrized a special family of Lévy measure and afterwards derived the no arbitrage drift condition to the parameters. The symmetry condition (2.9) is interesting and it was not mentioned in the original paper of Tangent Lévy Model [1]. In fact, we can see (especially from the figure of factor loadings) that this condition somehow forces the possible shapes and movements that our parametrized Lévy measure could have. It is not really symmetric. Instead it keeps the "balance" of the distribution of λ^j: for example, the factor loadings could not have the shape of being only positive on the right side and being zero on the left side, which means adding intensity only to positive jumps. Such condition gives us more detailed description of the change of intensity distribution.

Another interesting thing is that because of the existence of the symmetry condition, we put more constraint on the calibration optimization problem. Unexpectedly it is much harder to get good fit. Although the attained λ^j in Figure 3.1 look natural and not complicated, it in fact takes a lot of effort to obtain them finally, by playing with the regularization criteria, initial guess and parameter inputs.

Overall, our parametric model seems to work properly. The strength of our model is that it is explicit and finite dimensional, but enough to fit any initial smile. Although the resulting implied smile dynamics is also finite dimensional, due to the possible use of different factor loadings its behaviour could be rich enough.

Appendix A

MATLAB Code

```
% MATLAB Code for calibrating to market prices
function [lambda,error,abserror,within,a,c1,c2]=density4()
clear;
load('alldata4.mat');
t=4/250; r=0; q=0;
N=8;
B=-eye(2*N+1); e=zeros(2*N+1,1); e(1:2*N+1,1)=-1e-7;
F=zeros(2*N+1,2*N+1); f=zeros(2*N+1,1);
c1=zeros(size(alldata4,1),3);
c2=zeros(size(alldata4,1),3);
k=zeros(size(alldata4,1),3);
s=zeros(size(alldata4,1),1);
for j=1:size(alldata4,1)
    s(j)=alldata4(j,1);
    for i=1:3
        k(j,i)=alldata4(j,3*i-1)/alldata4(j,5)*alldata4(j,1);
        sigma1=alldata4(j,i*3);sigma2=alldata4(j,i*3+1);
        d1=(log(s(j)/k(j,i))+(r-q+sigma1^2/2)*t)/sigma1/sqrt(t);
        d2=d1-sigma1*sqrt(t);
        c1(j,i)=s(j)*exp(-q*t)*normcdf(d1)-k(j,i)*exp(-r*t)*normcdf(d2);
        c1(j,i)=c1(j,i)-max(0,s(j)-k(j,i)); %c1 & c2 are time value of money
        d1=(log(s(j)/k(j,i))+(r-q+sigma2^2/2)*t)/sigma2/sqrt(t);
        d2=d1-sigma2*sqrt(t);
        c2(j,i)=s(j)*exp(-q*t)*normcdf(d1)-k(j,i)*exp(-r*t)*normcdf(d2);
        c2(j,i)=c2(j,i)-max(0,s(j)-k(j,i));
    end
```

```
    c=(c1+c2)/2;

    d=log(1.1); x=zeros(2*N+1,1);
    for i=1:2*N+1
        x(i)=-d+(i-1)*d/N;
    end
    logx=x;
    x=exp(x);
end

y=x-1;
F(1,:)=y';
F(2,N/2+1:1.5*N+1)=y(N/2+1:1.5*N+1)';

index=1;

save('dataset.mat');

error=zeros(size(alldata4,1),1);
lambda=zeros(size(alldata4,1),2*N+1);
lambda(1,1)=0.1; lambda(1,2*N+1)=(1-x(1))*lambda(1,1)/(x(2*N+1)-1);
for i=2:N/2
    lambda(1,i)=0.1+lambda(1,i-1);
    lambda(1,2*N+2-i)=(1-x(i))*lambda(1,i)/(x(2*N-i+2)-1);
end
for i=N/2:N
    lambda(1,i+1)=lambda(1,i)+0.5;
end
for i=N+2:1.5*N+1
    lambda(1,i)=lambda(1,i-1)-0.5;
end

lb=zeros(2*N+1,1);
lb(1:N/2)=0.1; lb(1.5*N+2:2*N+1)=0.1;
for i=N/2+1:1.5*N+1
```

```
    lb(i)=1;
end
eps3=0.1; save('dataset.mat','-append','eps3');
option=optimset('Algorithm','sqp','MaxFunEvals',100000,'MaxIter',1000);
[lambda(1,:),error(1)]=fmincon(@optimal,lambda(1,:),B,e,F,f,lb,[],[],option);
while (error(1)>3)&&(eps3>1e-3)
eps3=eps3/2; save('dataset.mat','-append','eps3');
[lambda(1,:),error(1)]=fmincon(@optimal,lambda(1,:),B,e,F,f,lb,[],[],option);
end
previous=lambda(1,:);

for i=2:size(alldata4,1)
lambda(i,:)=lambda(i-1,:);
index=index+1;
eps3=0.1;
save('dataset.mat','-append','index','previous','eps3');
[lambda(i,:),error(i)]=fmincon(@optimal,lambda(i,:),B,e,F,f,lb,[],[],option);
while (error(i)>3)&&(eps3>5e-3)
eps3=eps3/2; save('dataset.mat','-append','eps3');
[lambda(i,:),error(i)]=fmincon(@optimal,lambda(1,:),B,e,F,f,lb,[],[],option);
end
previous=lambda(i,:);
end

b=zeros(2*N+1,1);
A=zeros(2*N+1);

index=1;
a=zeros(size(alldata4,1),3);
abserror=zeros(size(alldata4,1),1);
V=zeros(size(alldata4,1),2*N+1);
within=zeros(size(alldata4,1),1);
for w=1:size(alldata4,1)
    l=sum(lambda(w,:));

    for i=1:2*N+1
```

```
  temp=0;
  for j=-N:N
    if (i-j>0) && (i-j<2*N+2)
        A(i,j+N+1)=lambda(w,i-j)*x(i-j);
    end
    temp=temp+
    lambda(w,j+N+1)*x(j+N+1)*max(0,s(index)*
    (1-exp(logx(1)+d/(N)*(i-j-1))));
  end
  A(i,i)=A(i,i)-l;
  b(i)=temp-l*max(0,s(index)*(1-x(i)));
end

v=zeros(2*N+1,1);

for i=1:2*N+1
    v(i)=max(0,s(index)-s(index)*x(i));
end

% Taylor series for (exp(tA)-I)*(A\b)
E = zeros(size(A));
F = eye(size(A));
p = 2;
while norm(E+F-E,1) > 0
    E = E + F;
    F = (t*A)*F/p;
    p = p+1;
end

E2 = E*t;

V(w,:)=(E2*b)';

for i=1:3
    for j=1:2*N
        if k(index,i)>=s(index)*x(j) && k(index,i)<s(index)*x(j+1)
```

```
                a(w,i)=V(w,j)+(k(index,i)-s(index)*x(j))/
                (s(index)*x(j+1)-s(index)*x(j))*(V(w,j+1)-V(w,j));
            end
        end
        if a(w,i)>c1(w,i) || a(w,i)<c2(w,i)
            within(w)=1;
        end
    end

    abserror(w)=2*norm((a(w,:)-c(index,:)).*
    ((c1(index,:)-c2(index,:)).^(-1)),2);
    index=index+1;
end

figure;
plot(error);
hold on
plot(abserror,'r');

figure;
plot(lambda');

for i=1:3
    figure;
    plot(a(:,i));
    hold on
    plot(c1(:,i),'r');
    plot(c2(:,i),'g');
end
save('temp.mat');
end

function [error]=optimal(Lamb)
load('dataset.mat');
lambda=Lamb;
```

```
b=zeros(2*N+1,1);
A=zeros(2*N+1);

l=sum(lambda);

for i=1:2*N+1
    temp=0;
    for j=-N:N
        if (i-j>0) && (i-j<2*N+2)
            A(i,j+N+1)=lambda(i-j)*x(i-j);
        end
        temp=temp+lambda(j+N+1)*x(j+N+1)*max(0,s(index)*
        (1-exp(logx(1)+d/N*(i-j-1))));
    end
    A(i,i)=A(i,i)-l;
    b(i)=temp-l*max(0,s(index)*(1-x(i)));
end

v=zeros(2*N+1,1);

for i=1:2*N+1
    v(i)=max(0,s(index)-s(index)*x(i));
end

% Taylor series for (exp(tA)-I)*(A\b)
E = zeros(size(A));
F = eye(size(A));
p = 2;
while norm(E+F-E,1) > 0
    E = E + F;
    F = (t*A)*F/p;
    p = p+1;
end

E2 = E*t;
```

```
V=E2*b;

a=zeros(1,3);
for i=1:3
    for j=1:2*N
        if k(index,i)>=s(index)*x(j) && k(index,i)<s(index)*x(j+1)
            a(i)=V(j)+(k(index,i)-s(index)*x(j))/
            (s(index)*x(j+1)-s(index)*x(j))*(V(j+1)-V(j));
        end
    end
end

error=2*norm((a-c(index,:)).*((c1(index,:)-c2(index,:)).^(-1)),2);
eps1=0.1; eps2=0.01;
if index==1
    error=error+abs(lambda(1))*eps1+abs(lambda(2*N+1))*eps1;
    error=error+norm(diff(lambda),2)*eps2;
else
    error=error+norm(min(abs(lambda-previous),abs(ones(1,2*N+1)-lambda.
    /previous)),2)*eps3+norm(diff(lambda),2)*eps2+abs(lambda(2*N+1))*
    eps1+abs(lambda(1))*eps1;
end
end
```

Bibliography

[1] R. Carmona and S. Nadtochiy. Tangent lévy market models. *submitted to Finance and Stochastics*, available at: http://users.ox.ac.uk/∼oxma0016.

[2] R. Carmona and S. Nadtochiy. Local volatility dynamic models. *Finance and Stochastics*, 13(1):1 – 48, 2009.

[3] P. Carr, H. Geman, D. Madan, and M. Yor. From local volatility to local lévy models. *Quantitative Finance*, 4:581 – 588, 2005.

[4] R. Cont and P. Tankov. *Financial modelling with jump processes*. Chapman and Hall/CRC, 2004.

[5] R. Cont and E. Voltchkova. Integro-differential equations for option prices in exponential lévy models. *Finance and Stochastics*, 9:299 – 325, 2005.

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

[7] J. Jacod and P. Protter. Risk neutral compatibility with option prices. Technical report, Université de Paris VI and Cornell University, April 2006.

[8] K. Sato. *Lévy processes and infinitely divisible distributions*. Cambridge University Press, 1999.

[9] F. Delbaen; W. Schachermayer. A general version of the fundamental theorem of asset pricing. *Mathematische Annalen*, 300(1):463520, 1994.

[10] P. Schönbucher. A market model for stochastic implied volatility. *Phil. Trans. of the Royal Society, Series A*, 357:2071 – 2092, 1999.

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

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