Extensions of the SABR Model for Equity Options

IRAKLI KHOMASURIDZE

13 July 2009

Contents

Chapter 1

Introduction

Modeling of stock price behavior (dynamic) is key concept in option theory, as based on chosen model one can further derive prices for options on underlying assets. It is more then obvious that the better model reflects real asset dynamics, the better option pricing will be.

This thesis discusses pricing of equity options using extension of "classical" SABR model. The key idea of this extension is that we assume that volatility is not only stochastic but also has non zero drift term. Drift term is chosen to be mean reverting, i.e. we assume that volatility is constantly pushed to some function with predefined mean reverting rate, while diffusion term is chosen to be similar to the one under "classical" SABR model.

1.1 Model of Asset Dynamic

Maybe the most intuitive way to define asset dynamic is to use random walk and Wiener process. First let us define random walk, suppose that we have N periods of length Δt and:

$$z\left(t_{k+1}\right) = z\left(t_k\right) + \epsilon\left(t_k\right)\sqrt{\Delta t}$$
$$t_{k+1} = t_k + \Delta t; \quad t_0 = 0; \quad z\left(0\right) = 0$$

for $k = 0, 1, \ldots N$. This process is called random walk. In this equation $\epsilon\left(t_k\right) \sim \mathbf{N}(0,1)$ standardized normal random variable. Additionally we assume that this variables are mutually uncorrelated $E\left[\epsilon\left(t_i\right)\epsilon\left(t_j\right)\right] = 0$ for $i \neq j$.

A standard Wiener process obtained by taking limit of the random walk process $\Delta t \to 0$. In the symbolic form we write limit of increment as:

$$dW_t = \epsilon\left(t\right)\sqrt{\Delta t}$$

This definition is not rigorous because we have no assurance that limiting process exists but it provides a good intuitive description. Generalized Wiener process is defined as:

$$dX_t = \nu dt + \sigma dW_t$$

here $\nu-$ is drift term and $\sigma-$ is diffusion term. The first one defines growth rate of X_t, and the last one defines level of variability (volatility) of process. Note that described process is stochastic by its nature. Thus, one can never deterministically name value of X_t at time t, although one can give distribution of the processes at t, in particular for generalized Wiener process we have that: $X_t \sim \mathbf{N}(\nu t, \sigma^2 t)$. Alternative way to define Wiener process is to list all its features. Interested reader could refer to [15].

Our next logical step is to model stock price process using Wiener process:

$$d\left(\ln\left(S_t\right)\right) = dX_t = \nu dt + \sigma dW_t$$

Reason behind this model is that:

- Stock price could not be negative like Wiener process for example, and that is why it assumed to be exponent of generalized Wiener process.
- Stock price will be exponentially growing in the mean. This fact also pretty well fits to reality.

One can immediately see that the distribution of stock price S_t will be lognormal as: $\ln(S_t) \sim \mathbf{N}\left(\ln(S_0) + \nu t, \sigma^2 t\right)$. Described process of stock price behavior is sometimes called GBM (Geometric Brownian Motion) and might be simplest between all plausible models for asset dynamics.

1.1.1 Ito's Formula

In order to express stock price dynamic explicitly by Wiener process and to derive later option pricing formulas we need to introduce Ito' formula. This formula allows us to systematically perform transformation of different stochastic processes.

Let us consider random process X_t defined by:

$$dX_t = a(X_t, t)dt + b(X_t, t)dW_t$$

Suppose that the process $Y_t = F(X_t, t)$ is defined. Then Y_t satisfies the Ito equation:

$$dY_t = dF(X_t, t) = \left(\frac{\partial F}{\partial t} + \frac{\partial F}{\partial x}a + \frac{1}{2}\frac{\partial^2 F}{\partial x^2}b^2\right)dt + \frac{\partial F}{\partial x}bdW_t$$

where dW_t is the same Wiener process as in the expression for dX_t.

If we apply Ito's formula to random process:

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{1.1}$$

and function $F(x, t) = \ln(x)$ then we will receive:

$$d(\ln(S_t)) = \left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma dW_t$$

Denoting $\nu = \mu - \frac{\sigma^2}{2}$ we will get above mentioned GBM dynamic, which agrees with our earlier results.

More general form of Ito's formula for function of time and few stochastic variables and more rigorous proof could be found in [17].

1.2 Concept of Equity Option

Holder of stock with price dynamic described above might worry about the possibility of the stock price to drop below some level K (strike price) in T (maturity) years from now. For this reason he/she might want to buy protection against undesirable price movement from second party. Protection is called *European Put option* if it promises that:

"Option holder has the right, but not obligation to sell stock for K in T years from now"

As it immediately follows from the contract, this option will always have positive value, as it is right but not obligation (holder will never lose). Formally option payment at maturity time could be expressed as:

$$P = \max(K - S_T, 0)$$

So if option price in T years will be above level K there is no reason to execute (sell) stock for lower price. On the contrary, if it is below K, option holder can be compensated for price drop.

If one holds short[1] position in equity stock he/she might worry about possibility of asset price to go above level K in T year from now and buy *European Call option*. This type of option allows its holder to buy stock for K in T years from now:

$$C = \max(S_T - K, 0)$$

Another modification of European option is *American option* and this type of option gives holder an additional right to exercise it any time before maturity T. Described options are widely traded on the market in contrast to *Exotic option* (which are usually traded over the counter). Exotic options have more unusual features implemented in the contract. For example, European put option, which terminates in case the stock price goes below some barrier level L before the time of maturity, will be called *Barrier option*.

One can see that price dynamic of underlying asset is very important, as options are derived based on them, thus pricing of option is not a trivial problem in general.

1.2.1 Black-Scholes Equation

In some cases it is possible to find prices of options analytically, but then one needs to make some simplifying assumption about stock price dynamics. Black-Scholes developed their theory assuming that stock price dynamics is described by GBM and gave analytical formulas for European put and call options. Black-Scholes formula derived as solution of Partial Differential Equation. Main assumptions to be made in order to derive Black Scholes PDE are:

1. Stock price dynamic is described by (1.1)
2. On the market there exists risk free asset carrying interest rate r, with following dynamic: $dB = rBdt$.
3. Option price process has form $f(S(t), t)$.

Further to derive PDE one should:

- Apply Ito's rule to: $f(S(t), t)$ in order to receive option price dynamic
- Construct self financing portfolio consisting of Δ_S amount of stock and Δ amount of options
- Choose Δ_S and Δ in the way to eliminate source of uncertainty in portfolio.
- Use the fact that constructed portfolio will be risk free and use no arbitrage arguments to derive final PDE

And finally Black Scholes PDE for asset dynamics:

$$dS_t = rS_t dt + \sigma S_t dW_t$$

has form:

$$\frac{\sigma^2 S^2}{2}\frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - rV = 0$$

Black-Scholes formula (solution of Black Scholes PDE equation for European put and call options) is given by:

$$\begin{aligned} V_{call} &= S_0 N(d_1) - Ke^{-rT}N(d_2) \\ V_{put} &= Ke^{-rT}N(-d_2) - S_0 N(-d_1) \end{aligned}$$

[1]Short position in equity stocks is liability to return to somebody indicated amount of stocks at given time. Long position in equity stock is equivalent to owning stock.

where $N(\cdot)$ is cumulative distribution function for standard normal and

$$\begin{aligned} d_1 &= \frac{\ln(S/K) + \left(r + \sigma^2/2\right) T}{\sigma\sqrt{T}} \\ d_2 &= d_1 - \sigma\sqrt{T} \end{aligned}$$

Detailed derivation together with initial and boundary conditions for more general stock price process is presented in 2.1.

1.3 Generalization

Despite the fact that Black-Scholes formula gives nice analytical solutions, it is rarely used with all initial assumptions made. And the problem is that real stock price dynamic does not have form (1.1). Although in practice traders are still using Black-Scholes formula assuming the volatility of stock price to be the function of strike and maturity. This function is found by mark to market of option prices (with different strikes and maturities) to Black-Scholes formula and is named *implied volatility.* It might look controversial and false that stock price volatility is assumed to be function of strike and time to maturity, but one can alternatively think of implied volatility function as method for approximating stock price dynamics by GBM.

Hence, in order to get more realistic option pricing methods one needs to develop a new model for dynamics of stock price. And there are in general three possibilities to develop new theory. First, one is to assume that stock price is non-Markovian. Second, one is to leave it Markovian, but base the model on stochastic process other than Wiener's. Third, one is to assume that volatility is not only variable (for example time dependent) but also stochastic. Last models are called models with stochastic volatility and will be the focus of the present research.

1.3.1 SABR model

In the derivation of SABR model (Stochastic $\alpha\beta\rho$ model) Hagan [5] chose the third option. Under SABR model it is assumed that the dynamic of forward stock value $f_t = S_0 \exp(rt)$ under risk neutral measure is described by:

$$\begin{cases} df_t = \alpha_t f_t^{\beta} dW_t^1 \\ d\alpha_t = \nu \alpha_t dW_t^2 \end{cases} \tag{1.2}$$

where α_t- is stochastic volatility, $\nu-$ volatility of volatility, $\beta-$ is a positive constant, additionally it is assumed that W_t^1 and W_t^2 are two correlated sources of uncertainty: $dW_t^1 dW_t^2 = \rho dt$. Also we assume that at $t = 0$ forward stock value and volatility are given: f_0, α_0.

Under this model volatility α_t is stochastic and lognormally distributed. To show this one should apply Ito's rule to $\ln(\alpha_t)$ to receive:

$$d\left(\ln(\alpha_t)\right) = -\frac{1}{2}\nu^2 dt + \nu dW_t^2$$

In his research Hagan received analytical approximation for implied volatility function:

$$\begin{aligned} \sigma(K,f) &= \frac{\alpha_0}{(fK)^{\frac{1-\beta}{2}}\left[1 + \frac{(1-\beta)^2}{24}\ln^2\left(\frac{f}{K}\right) + \frac{(1-\beta)^4}{1920}\ln^4\left(\frac{f}{K}\right) + \dots\right]} \cdot \left(\frac{z}{x(z)}\right) \cdot \\ &\cdot \left(1 + \left[\frac{(1-\beta)^2}{24}\frac{\alpha_0^2}{(fK)^{1-\beta}} + \frac{1}{4}\frac{\rho\beta\nu\alpha_0}{(fK)^{\frac{1-\beta}{2}}} + \frac{2-3\rho^2}{24}\nu^2\right] T + \dots\right) \end{aligned} \tag{1.3}$$

where:

$$\begin{aligned} z &= \frac{\nu}{\alpha_0}(fK)^{\frac{1-\beta}{2}} \ln\left(\frac{f}{K}\right) \\ x(z) &= \ln\left[\frac{\sqrt{1-2\rho z + z^2} + z - \rho}{1-\rho}\right] \end{aligned}$$

and $f = S_0 \exp(rT)-$ forward price of stock under risk neutral measure, $K-$ strike price, $T-$ time to maturity. Substituting 1.3 into Black Scholes formula one will receive price of European put or call option based on a stock with stochastic volatility given by (1.2).

Derived formula for implied volatility will be used as our main touchstone for valuing and deriving numerical methods.

1.3.2 Research questions

In present research we further generalize the system describing stock price dynamic and consider following model for stock price and volatility dynamic:

$$\begin{cases} dS_t = r(t) S_t dt + \alpha_t (S_t)^{\beta} dW_t^1 \\ d\alpha_t = \kappa (\theta(t) - \alpha_t) dt + \nu \alpha_t dW_t^2 \end{cases}$$

which generalizes features of SABR model by means of introducing volatility mean reverting term $\mu_\alpha = \kappa(\theta(t) - \alpha_t)$, with mean reverting limit function $\theta(t)$, mean reverting rate κ and time dependent correlation $dW_t^1 dW_t^2 = \rho(t) dt$. It should be mentioned that recently approximation for implied volatility for above model with constant coefficients ($r(t) = r$, $\theta(t) = \theta$, $\rho(t) = \rho$) where presented [11].

In this project we would like to:

- Design numerical methods to reproduce European option prices generated by the model based on: Finite Difference and Monte Carlo methods.
- Extend such methods for options with early exercise opportunity (American options) and on options that pay continuous or cash dividends.
- Determine whether it is possible to derive pricing formula similar to (1.3) for this model as well.
- Fit the model parameters to real market data.

Chapter 2

Extended SABR model and PDE

Under the "extended" SABR model we assume that stock price dynamics S_t with stochastic volatility α_t is described under risk neutral measure as:

$$\begin{cases} dS_t = (r(t) - q(t))\, S_t dt + \alpha_t (S_t)^{\beta} dW_t^1 \\ d\alpha_t = \kappa(\theta(t) - \alpha_t)\, dt + \nu \alpha_t dW_t^2 \end{cases} \tag{2.1}$$

in order to shorten notation and for further generalization of the above system, we will denote $\mu_S = \mu_S(S, \alpha, t) = (r(t) - q(t))\, S_t$, $\sigma_S = \sigma_S(S, \alpha, t) = \alpha_t (S_t)^{\beta}$, $\mu_\alpha = \mu_\alpha(\alpha, t) = \kappa(\theta(t) - \alpha_t)$, $\sigma_\alpha = \sigma_\alpha(\alpha, t) = \nu\alpha_t$. Where $r(t)$ - risk free rate, $q(t)$ - continuous dividend rate, $\theta(t)$ - mean reverting limit for volatility, κ - mean reverting rate of volatility, ν - volatility of volatility, $dW_t^1$and dW_t^2 assumed to be correlated $dW_t^1 dW_t^2 = \rho(t)\, dt$. Additionally $r(t)$, $q(t)$, $\theta(t)$ - are non negative and non stochastic; correlation is bounded $-1 \leq \rho(t) \leq 1$ and non-stochastic; κ, ν - positive real numbers. Also we assume that at $t = 0$ stock price and volatility are defined S_0, α_0.

Unfortunately it seems rather difficult to directly find distribution for S_t, but in order to get the feeling of the model we can try to find distribution of volatility (for $\theta(t) = \theta$). Using Ito's rule for $\ln(\alpha_t)$ we will get:

$$d(\ln(\alpha_t)) = \left[\kappa\left(\frac{\theta}{\alpha_t} - 1\right) - \frac{\nu^2}{2}\right] dt + \nu dW_t$$

and again we can not directly integrate this SDE as drift term includes α_t. Let us try to use different technique in order to find first moments for volatility $E[\alpha_t]$ and $E\left[\alpha_t^2\right]$. Taking expectation of:

$$d\alpha_t = \kappa(\theta - \alpha_t)\, dt + \nu\alpha_t dW_t$$

interchanging order of expectation and differentiation we will get:

$$dE[\alpha_t] = \kappa(\theta - E[\alpha_t])\, dt$$

Integrating above as usual ODE w.r.t. $E[\alpha_t]$, with initial conditions $E[\alpha_0] = \alpha_0$, we will finally obtain:

$$E[\alpha_t] = \theta + (\alpha_0 - \theta)\, e^{-\kappa t} \tag{2.2}$$

Now in order to find $VAR[\alpha_t]$ let us one more time use Ito's rule for α_t^2:

$$d\left(\alpha_t^2\right) = \left[2\kappa\theta\alpha_t + \left(\nu^2 - 2\kappa\right)\alpha_t^2\right] dt + 2\nu\alpha_t^2 dW_t$$

taking an expectation and again interchanging order of differentiating and expectation:

$$d\left(E\left[\alpha_t^2\right]\right) = \left[2\kappa\theta E[\alpha_t] + \left(\nu^2 - 2\kappa\right) E\left[\alpha_t^2\right]\right] dt$$

Substituting obtained expression for $E[\alpha_t]$ (2.2) and integrating as usual ODE w.r.t. $E\left[\alpha_t^2\right]$, with initial conditions $E\left[\alpha_0^2\right] = \alpha_0^2$:

$$E\left[\alpha_t^2\right] = \frac{2\kappa\theta^2}{2\kappa - \nu^2} + \frac{2\kappa\theta\left(\alpha_0 - \theta\right)}{\kappa - \nu^2} e^{-\kappa t} + \left(\alpha_0^2 - \frac{2\kappa\theta^2}{2\kappa - \nu^2} - \frac{2\kappa\theta\left(\alpha_0 - \theta\right)}{\kappa - \nu^2}\right) e^{-\left(2\kappa - \nu^2\right)t}$$

and finally

$$VAR\left[\alpha_t\right] = E\left[\alpha_t^2\right] - E^2\left[\alpha_t\right] \tag{2.3}$$

For "classical" SABR model $\kappa = 0$ ($\mu_\alpha = 0$) we can immediately see that:

$$E\left[\alpha_t\right] = \alpha_0; \quad VAR\left[\alpha_t\right] = \alpha_0^2\left(e^{t\nu^2} - 1\right) \tag{2.4}$$

Comparing above formulas one can immediately notice that in case of "extended" SABR model variance of volatility tends to final limit when $t \longrightarrow \infty$, but for "classical" SABR model it tends to infinity. Expectation of volatilities for "classical" model stays constant, while for "extended" model it tends to θ when $t \longrightarrow \infty$.

Using described technique one can find higher moments for volatility dynamics as well.

2.1 Deriving PDE

Let us denote by $V = V(S, \alpha, t)$ price of derivative on underlying asset S with volatility α. Dynamic of V could be found using bivariate Ito's rule:

$$\begin{aligned} dV &= \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS_t + \frac{\partial V}{\partial \alpha}d\alpha_t \\ &\quad + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}dS_t dS_t + \frac{\partial^2 V}{\partial S \partial \alpha}dS_t d\alpha_t + \frac{1}{2}\frac{\partial^2 V}{\partial \alpha^2}d\alpha_t d\alpha_t \end{aligned} \tag{2.5}$$

and Box Algebra

$$\begin{bmatrix} & dt & dW_t^1 & dW_t^2 \\ dt & 0 & 0 & 0 \\ dW_t^1 & 0 & dt & \rho(t)\,dt \\ dW_t^2 & 0 & \rho(t)\,dt & dt \end{bmatrix} \tag{2.6}$$

From (2.5) and (2.6) we obtain following dynamics for V:

$$\begin{aligned} dV &= \left[\frac{\sigma_S^2}{2}\frac{\partial^2 V}{\partial S^2} + \rho(t)\,\sigma_S\sigma_\alpha\frac{\partial^2 V}{\partial S \partial \alpha} + \frac{\sigma_\alpha^2}{2}\frac{\partial^2 V}{\partial \alpha^2} + \mu_S\frac{\partial V}{\partial S} + \mu_\alpha\frac{\partial V}{\partial \alpha} + \frac{\partial V}{\partial t}\right]dt + \\ &\quad \sigma_S\frac{\partial V}{\partial S}dW_t^1 + \sigma_\alpha\frac{\partial V}{\partial \alpha}dW_t^2 \end{aligned}$$

or denoting differential operator in square brackets by $\mathcal{L}(\cdot)$:

$$dV = \mathcal{L}(V)\,dt + \sigma_S\frac{\partial V}{\partial S}dW_t^1 + \sigma_\alpha\frac{\partial V}{\partial \alpha}dW_t^2$$

Dynamic of the portfolio $P = V - \Delta_1 S - \Delta_2 V_1$ consisting of two different derivatives V, V_1 (on the same underlying asset) and underlying asset S itself will be:

$$\begin{aligned} dP &= dV - \Delta_1 dS_t - \Delta_2 dV_1 \\ &= \left[\mathcal{L}(V)\,dt + \sigma_S\frac{\partial V}{\partial S}dW_t^1 + \sigma_\alpha\frac{\partial V}{\partial \alpha}dW_t^2\right] - \Delta_1\left[\mu_S dt + \sigma_S dW_t^1\right] - \\ &\quad \Delta_2\left[\mathcal{L}(V_1)\,dt + \sigma_S\frac{\partial V_1}{\partial S}dW_t^1 + \sigma_\alpha\frac{\partial V_1}{\partial \alpha}dW_t^2\right] \end{aligned}$$

rearranging terms we get:

$$\begin{aligned} dP &= \left[\mathcal{L}(V) - \Delta_1\mu_S - \Delta_2\mathcal{L}(V_1)\right]dt \\ &\quad + \left[\sigma_S\frac{\partial V}{\partial S} - \Delta_1\sigma_S - \Delta_2\sigma_S\frac{\partial V_1}{\partial S}\right]dW_t^1 + \left[\sigma_\alpha\frac{\partial V}{\partial \alpha} - \Delta_2\sigma_\alpha\frac{\partial V_1}{\partial \alpha}\right]dW_t^2 \end{aligned} \tag{2.7}$$

In order to eliminate uncertainty and receive risk free self financing portfolio we choose Δ_1 and Δ_2 to satisfy the following system:

$$\begin{aligned} \sigma_S\frac{\partial V}{\partial S} - \Delta_1\sigma_S - \Delta_2\sigma_S\frac{\partial V_1}{\partial S} &= 0 \\ \sigma_\alpha\frac{\partial V}{\partial \alpha} - \Delta_2\sigma_\alpha\frac{\partial V_1}{\partial \alpha} &= 0 \end{aligned}$$

solving this system we will get:

$$\begin{aligned} \Delta_1 &= \frac{\partial V}{\partial S} - \frac{\partial V_1}{\partial S}\frac{\partial V}{\partial \alpha}\Big/\frac{\partial V_1}{\partial \alpha} \\ \Delta_2 &= \frac{\partial V}{\partial \alpha}\Big/\frac{\partial V_1}{\partial \alpha} \end{aligned}$$

To exclude arbitrage possibility, return of constructed portfolio must be equal to return of risk free investment:

$$dP = r(t)\,Pdt = r(t)\left[V - \Delta_1 S_t - \Delta_2 V_1\right]dt \tag{2.9}$$

Equating (2.9) and (2.7) and substituting expressions for Δ_1, Δ_2 we obtain:

$$\left[\mathcal{L}(V) - r(t)\,V\right]\Big/\frac{\partial V}{\partial \alpha} = \left[\mathcal{L}(V_1) - r(t)\,V_1\right]\Big/\frac{\partial V_1}{\partial \alpha}$$

Above equation must hold independently for arbitrary V and V_1, thus left and right hand sides of equation must be equal to some function $\lambda(S,\alpha,t)$ which is volatility risk premium.

Finally, derivative $V = V(S,\alpha,t)$ must satisfy to:

$$\mathcal{L}(V) - r(t)\,V = \lambda(S,\alpha,t)\frac{\partial V}{\partial \alpha}$$

Then choosing $\lambda(S,\alpha,t) = 0$ and rewriting above equation in open form:

$$\frac{\sigma_S^2}{2}\frac{\partial^2 V}{\partial S^2} + \rho(t)\,\sigma_S\sigma_\alpha\frac{\partial^2 V}{\partial S\partial\alpha} + \frac{\sigma_\alpha^2}{2}\frac{\partial^2 V}{\partial \alpha^2} + \mu_S\frac{\partial V}{\partial S} + \mu_\alpha\frac{\partial V}{\partial \alpha} + \frac{\partial V}{\partial t} - r(t)\,V = 0 \tag{2.10}$$

Substituting expressions for extended SABR model:

$$\frac{\alpha^2 S^{2\beta}}{2}\frac{\partial^2 V}{\partial S^2} + \alpha^2 S^\beta\rho(t)\,\nu\frac{\partial^2 V}{\partial S\partial\alpha} + \frac{\alpha^2\nu^2}{2}\frac{\partial^2 V}{\partial \alpha^2} + r(t)\,S\frac{\partial V}{\partial S} + \kappa(\theta(t) - \alpha)\frac{\partial V}{\partial \alpha} + \frac{\partial V}{\partial t} - r(t)\,V = 0$$

or in shorter notation:

$$\frac{\partial V}{\partial t} + \mathcal{B}[V] = 0.$$

2.2 Transformation

Let us perform some transformation of variables in order to get more convenient form of (2.10). In particular let us introduce new time and space variables:

$$\tau = T - t, x = D(S), y = G(\alpha) \tag{2.11}$$

assume that there exists inverse functions:

$$t = T - \tau, S = D_0(x), \alpha = G_0(y) \tag{2.12}$$

and following derivatives:

$$\begin{aligned}
\frac{\partial \tau}{\partial t} &= -1; \\
\frac{\partial x}{\partial S} &= \left.\frac{\partial D(S)}{\partial S}\right|_{S=D_0(x)} = D_1(x); \quad \frac{\partial^2 x}{\partial S^2} = \left.\frac{\partial^2 D(S)}{\partial S^2}\right|_{S=D_0(x)} = D_2(x); \\
\frac{\partial y}{\partial \alpha} &= \left.\frac{\partial G(\alpha)}{\partial \alpha}\right|_{\alpha=G_0(y)} = G_1(y); \quad \frac{\partial^2 y}{\partial \alpha^2} = \left.\frac{\partial^2 G(\alpha)}{\partial \alpha^2}\right|_{\alpha=G_0(y)} = G_2(y);
\end{aligned}$$

then partial derivatives in (2.10) could be expressed as:

$$\begin{aligned}
&\frac{\partial V}{\partial t} = \frac{\partial V}{\partial \tau}\frac{\partial \tau}{\partial t} = -\frac{\partial V}{\partial \tau}; \\
&\frac{\partial V}{\partial S} = \frac{\partial V}{\partial x}\frac{\partial x}{\partial S} = \frac{\partial V}{\partial x} D_1(x); \\
&\frac{\partial V}{\partial \alpha} = \frac{\partial V}{\partial y}\frac{\partial y}{\partial \alpha} = \frac{\partial V}{\partial y} G_1(y); \\
&\frac{\partial^2 V}{\partial S \partial \alpha} = \frac{\partial V}{\partial x \partial y}\frac{\partial x}{\partial S}\frac{\partial y}{\partial \alpha} = \frac{\partial V}{\partial x \partial y} D_1(x) G_1(y); \\
&\frac{\partial^2 V}{\partial S^2} = \frac{\partial^2 V}{\partial x^2}\left(\frac{\partial x}{\partial S}\right)^2 + \frac{\partial V}{\partial x}\frac{\partial^2 x}{\partial S^2} = \frac{\partial^2 V}{\partial x^2} D_1^2(x) + \frac{\partial V}{\partial x} D_2(x); \\
&\frac{\partial^2 V}{\partial \alpha^2} = \frac{\partial^2 V}{\partial y^2}\left(\frac{\partial y}{\partial \alpha}\right)^2 + \frac{\partial V}{\partial y}\frac{\partial^2 y}{\partial \alpha^2} = \frac{\partial^2 V}{\partial y^2} G_1^2(y) + \frac{\partial V}{\partial y} G_2(y);
\end{aligned}$$

and finally our PDE will take form:

$$\begin{aligned}
\frac{\partial V}{\partial \tau} = -\mathcal{A}[V] = \quad & \frac{\sigma_x^2 D_1^2(x)}{2}\frac{\partial^2 V}{\partial x^2} + \rho(T-\tau)\sigma_x\sigma_y D_1(x) G_1(y)\frac{\partial^2 V}{\partial x \partial y} + \frac{\sigma_y^2 G_1^2(y)}{2}\frac{\partial^2 V}{\partial y^2} + \\
& \left[\mu_x D_1(x) + \frac{\sigma_x^2 D_2(x)}{2}\right]\frac{\partial V}{\partial x} + \left[\mu_y G_1(y) + \frac{\sigma_y^2 G_2(y)}{2}\right]\frac{\partial V}{\partial y} - r(T-\tau)V
\end{aligned} \tag{2.13}$$

note that (2.12) should be substituted into expressions for coefficients μ_S, σ_S, μ_α, σ_α and after this we denote them by $\mu_x = \mu_S(D_0(x), G_0(y), T-\tau)$, $\sigma_x = \sigma_S(D_0(x), G_0(y), T-\tau)$, $\mu_y = \mu_\alpha(G_0(y), T-\tau)$, $\sigma_y = \sigma_\alpha(G_0(y), T-\tau)$. Also it should be mentioned that this equation has the same partial derivatives as initial one and only coefficients in front of them differs.

Transformation of PDE could give valuable results not only from theoretical point of view but also from numerical perspective. In particular applying transformation and using finite difference method for solving transformed PDE one will receive finite-difference method with non-uniform grid.

In our research for numerical purposes the following transformations of space variables are used:

$$\begin{aligned}
&x = D(S) = (p_1 S)^{p_2} \qquad S = D_0(x) = x^{1/p_2}/p_1 \\
&D_1(x) = p_1 p_2 x^{\frac{p_2-1}{p_2}} \qquad D_2(x) = p_1^2 p_2 (p_2-1) x^{\frac{p_2-2}{p_2}}
\end{aligned} \tag{2.14}$$

$$\begin{aligned}
&x = D(S) = \ln(p_1 S) + p_2 \qquad S = D_0(x) = \exp(x-p_2)/p_1 \\
&D_1(x) = p_1 e^{p_2 - x} \qquad D_2(x) = -p_1^2 e^{2(p_2-x)}
\end{aligned} \tag{2.15}$$

$$\begin{aligned}
&x = D(S) = \left[\operatorname{asinh}\left(\tfrac{p_3(S-p_1)}{p_2}\right) + \operatorname{asinh}\left(\tfrac{p_3 p_1}{p_2}\right)\right]\Big/ p_3; \\
&S = D_0(x) = p_1 + p_2 \sinh\left(p_3 x - \operatorname{asinh}\left(\tfrac{p_3 p_1}{p_2}\right)\right)\Big/ p_3
\end{aligned} \tag{2.16}$$

The first two transformations are simple and widely used. Choosing $p_1 = p_2 = 1$ in the first one is the same as performing no transformation at all (thus we are free to perform power transformation or not). Second one is well known logarithmic transformation and in this case more care is needed when considering point $S = 0$, as $x \longrightarrow -\infty$ (thus we should replace zero with small positive number). Last transformation is suggested by [10] and allows to refine finite difference mesh near desired point by appropriate choice of parameters p_1, p_2, p_3. Here p_1 is grid concentration point, p_2 concentration level, while p_3 is chosen[1] in such a way to satisfy $x_{\max} = D(S_{\max})$ (note that $F(0) = 0$).

Note that everywhere below we will be applying and using equation (2.13). In numerical realization all mentioned transformation w.r.t. S and α are implemented and we can easily switch between them.

2.3 Problem Definition for European Option

In order to find price of the European put or call option one should find solution of (2.13) in the domain $\Omega = \{0 \le x \le x_{\max}; 0 \le y \le y_{\max}; 0 \le \tau \le \tau_{\max}\}$ with appropriate initial and boundary conditions and then perform corresponding inverse transformation defined by (2.12).

Let us first present initial and boundary conditions (for "classical" SABR model i.e. $\mu_\alpha = 0 \Longrightarrow \mu_y = 0$) in variables S, α and then translate them into corresponding conditions with transformed variables x, y. Initial condition[2]:

$$V(S, \alpha, 0) = F(S),$$

boundary conditions:

$$\begin{aligned}
&V(0, \alpha, t) = F(0) \exp\left(-\int_0^t r(s) ds\right) \\
&\frac{\partial^2 V(S_{\max}, \alpha, t)}{\partial S^2} = 0 \\
&V(S, 0, T) = F\left(S \exp\left(\int_0^t (r(s) - q(s))\, ds\right)\right) \exp\left(-\int_0^\tau r(s) ds\right) \\
&\frac{\partial V(S, \alpha_{\max}, t)}{\partial t} + \mathcal{B}\left[V(S, \alpha_{\max}, t)\right] = 0.
\end{aligned}$$

Since operator $\mathcal{B}\left[V(S, \alpha_{\max}, t)\right]$ includes partial derivatives we assume that on the bound $\alpha_{\max}$ there one-sided versions are used.

Now let us translate listed conditions in x, y and comment them. Initial condition reads:

$$V(x, y, 0) = F(D_0(x)) \tag{2.17}$$

boundary conditions are:

$$V(0, y, \tau) = F(D_0(0)) \exp\left(-\int_0^\tau r(T - s) ds\right) \tag{2.18}$$

$$\frac{\partial^2 V(x_{\max}, y, \tau)}{\partial x^2} D_1^2(x_{\max}) + \frac{\partial V(x_{\max}, y, \tau)}{\partial x} D_2(x_{\max}) = 0 \tag{2.19}$$

$$V(x, 0, \tau) = F\left(D_0(x) \exp\left(\int_0^\tau (r(T - s) - q(T - s))\, ds\right)\right) \exp\left(-\int_0^\tau r(T - s) ds\right) \tag{2.20}$$

$$\frac{\partial V(x, y_{\max}, \tau)}{\partial \tau} + \mathcal{A}\left[V(x, y_{\max}, \tau)\right] = 0 \tag{2.21}$$

[1]Value of this parameter is easily calculated using any one dimensional solver.
[2]Put: $F(S) = \max(K - S, 0)$ Call: $F(S) = \max(S - K, 0)$.

While setting boundary conditions on $x = x_{\max}$ and $y = y_{\max}$ one should always keep in mind that these conditions are "approximately" correct for large values of $x_{\max}$ and $y_{\max}$. So setting reasonable conditions on these boundaries will always be matter of correct modelling. For example condition (2.19) is "continuation" condition and suggests that for the large stock price Γ of put and call option should be close to zero. Alternatively this condition could be replaced by:

$$\text{put} \quad : \quad \frac{\partial V\left(x_{\max}, y, \tau\right)}{\partial x} D_1\left(x_{\max}\right) = 0 \tag{2.22a}$$

$$\text{call} \quad : \quad \frac{\partial V\left(x_{\max}, y, \tau\right)}{\partial x} D_1\left(x_{\max}\right) = 1 \tag{2.22b}$$

Condition for put option suggests that Δ should be close to zero, while for call option it should be close to 1.

Condition (2.21) is called "smoothing"condition[3] and is set on boundary $y = y_{\max}$. If bound $y = y_{\max} \gg 0$ is far from its origin then one might reasonably argue that the price of option will not be sensitive to volatility change (i.e. Vega $\mathcal{V} = 0$) and set:

$$\frac{\partial V\left(x, y_{\max}, \tau\right)}{\partial y} G_1\left(y_{\max}\right) = 0$$

During numerical experiments, it was justified that the best choice is still (2.21) as it is suitable for both large and comparably small values of $y_{\max}$.

Now let us present conditions for "extended" SABR model $\mu_y \neq 0$. All the above listed conditions, except (2.20), remain unchanged. On the boundary $y = 0$ we set again "smoothing" condition:

$$\mathcal{A}\left[V\left(x, 0, \tau\right)\right] + \frac{\partial V\left(x, 0, \tau\right)}{\partial \tau} = 0 \tag{2.23}$$

In order to explain this one might use following reasoning: In case of $\mu_y = 0$ and initially volatility $y = 0$, volatility stays equal to zero all time till maturity (this immediately follows from (2.2) when $\kappa = 0$ and $y_0 = 0$). Therefore we know that price of underlying asset will be deterministic, and as a consequence price of the option could be found explicitly, thus Dirichlet conditions (2.20) could be set. In case when volatility drift term $\mu_y \neq 0$ ($\kappa \neq 0$), we can not assert any more that volatility stays zero all time, even if it was initially zero (this also follows from (2.2) when $\kappa \neq 0$, $\theta \neq 0$ and $y_0 = 0$). Therefore we can not say that the price of option remains deterministic and we can not require (2.20) to hold. In this case we again set "smoothing" condition on boundary $y = 0$ (2.23).

In case when neither Dirichlet nor Neuman boundary conditions are posed, we require PDE itself to be satisfied on the bound. This is known as a "smoothing" condition and it seems to be natural choice. Thus, (2.23) and (2.21) are PDE operators set on corresponding bounds.

2.4 Problem Definition for American Option

For the American option not only boundary conditions should be changed but also the way we are solving PDE. The price of the American option can be obtained by solving time dependent complementarity problem in the domain $\Omega = \{0 \leq x \leq x_{\max}; 0 \leq y \leq y_{\max}; 0 \leq \tau \leq \tau_{\max}\}$:

$$\begin{cases} \frac{\partial V}{\partial \tau} + \mathcal{A}\left[V\right] \geq 0 \\ V \geq F\left(D_0(x)\right) \\ \left(V - F\left(D_0(x)\right)\right)\left(\frac{\partial V}{\partial \tau} + \mathcal{A}\left[V\right]\right) = 0 \end{cases}$$

where $F\left(D_0(x)\right)$ is obstacle function (same as initial condition). To solve linear complementarity problem using splitting technique (section 3.2.3) the above system should be rewritten into the following form:

$$\begin{cases} \frac{\partial V}{\partial \tau} + \mathcal{A}\left[V\right] = \lambda \\ \lambda \geq 0, \quad V \geq F\left(D_0(x)\right) \\ \left(V - F\left(D_0(x)\right)\right)\lambda = 0 \end{cases} \tag{2.24}$$

[3] For further discussion on "smoothing" condition see end of current section.

where λ is auxiliary function. In order to find stopping region (option exercising region) one should find part of the domain Ω where $\lambda \geq 0$. Later discrete formulation of the splitting problem will be given.

Initial and boundary conditions for the American option in case of "classical" SABR model ($\mu_y = 0$) are defined by:

$$V(x, y, 0) = F(D_0(x)) \tag{2.25}$$

$$V(0, y, \tau) = F(D_0(0)) \tag{2.26}$$

$$\frac{\partial^2 V(x_{\max}, y, \tau)}{\partial x^2} D_1^2(x_{\max}) + \frac{\partial V(x_{\max}, y, \tau)}{\partial x} D_2(x_{\max}) = 0 \tag{2.27}$$

$$V(x, 0, \tau) = F(D_0(x)) \tag{2.28}$$

$$\mathcal{A}[V(x, y_{\max}, \tau)] + \frac{\partial V(x, y_{\max}, \tau)}{\partial \tau} = 0 \tag{2.29}$$

Note that for the American option none of the conditions include discounting factor and this reflects the fact that the holder of the American option has right to exercise it at any given time. Additionally, condition on $x = x_{\max}$ like in the European case suggests that Γ for the American put or call option should be close to zero for large values of $x_{\max}$. This condition could be replaced by (2.22a) or (2.22b), as Δ of the American put (call) assumed to be close to zero (one). "Smoothing" condition (2.29) could be justified similarly to one for the European option.

In case of "extended" SABR model ($\mu_y \neq 0$) "smoothing" condition on $y = 0$ should be set:

$$\mathcal{A}[V(x, 0, \tau)] + \frac{\partial V(x, 0, \tau)}{\partial \tau} = 0 \tag{2.30}$$

Explanation of boundary condition replacement, goes similarly to one described for the European option (page 13).

2.5 Cash Dividends

In above discussion we assumed that dividends are paid out continuously with rate $q(T - \tau)$. Derived PDE together with corresponding boundary and initial conditions reflects this fact.

For discrete dividends we need to reformulate our PDE problem into PDE with initial-contact problem. Let us assume that the dividends are paid only once at time τ_1, where $0 < \tau_1 < \tau_{\max}$ and amount paid out is δ_1. Then our initial problem for the European option will be formulated for each intra-dividend interval separately. The first part will be[4]:

$$0 \leq \tau \leq \tau_1 : \begin{cases} \mathcal{A}[V_1] + \frac{\partial V_1}{\partial \tau} = 0 \\ V_1(x, y, 0) = F(D_0(x)) \\ V_1(0, y, \tau) = F(D_0(0)) \exp\left(-\int\limits_0^\tau r(T - s)\, ds\right) \\ V_1(x, 0, \tau) = F\left(D_0(x) \exp\left(\int\limits_0^\tau (r(T - s) - q(T - s))\, ds\right)\right) \exp\left(-\int\limits_0^\tau r(T - s) ds\right) \end{cases}$$

[4]Only PDE, intial and those of boundary conditions are listed that change.

the second part:

$$\tau_1 \leq \tau \leq \tau_{\max}: \quad \begin{cases} \mathcal{A}\left[V_2\right] + \frac{\partial V_2}{\partial \tau} = 0 \\ V_2\left(x, y, \tau_1\right) = G\left(x, y\right) \\ V_2\left(0, y, \tau\right) = G\left(0, y\right) \exp\left(-\int\limits_{\tau_1}^{\tau} r(T-s) ds \right) \\ V_2\left(x, 0, \tau\right) = G\left(D_0(x) \exp\left(\int\limits_{\tau_1}^{\tau} \left(r(T-s) - q(T-s)\right) ds \right), 0 \right) \exp\left(-\int\limits_{\tau_1}^{\tau} r(T-s) ds \right) \end{cases}$$

where:

$$G\left(x, y\right) = \begin{cases} V_1\left(x - \delta_1, y, \tau_1\right) & \delta_1 \leq x \\ V_1\left(0, y, \tau_1\right) & x < \delta_1 \end{cases}$$

For the second part, initial condition is shifted price of option (just before dividend) towards positive direction of x.

For the American option, the problem is formulated as:

$$0 \leq \tau \leq \tau_1: \quad \begin{cases} \frac{\partial V_1}{\partial \tau} + \mathcal{A}\left[V_1\right] \geq 0; \quad V_1 \geq F(D_0(x)); \quad \left(V_1 - F(D_0(x))\right)\left(\frac{\partial V_1}{\partial \tau} + \mathcal{A}\left[V_1\right]\right) = 0 \\ V_1\left(x, y, 0\right) = F(D_0(x)) \\ V_1\left(0, y, \tau\right) = F(D_0(0)) \\ V_1\left(x, 0, \tau\right) = F\left(D_0(x)\right) \end{cases}$$

and

$$\tau_1 \leq \tau \leq \tau_{\max}: \quad \begin{cases} \frac{\partial V_2}{\partial \tau} + \mathcal{A}\left[V_2\right] \geq 0; \quad V_2 \geq F(D_0(x)); \quad \left(V_2 - F(D_0(x))\right)\left(\frac{\partial V_2}{\partial \tau} + \mathcal{A}\left[V_2\right]\right) = 0 \\ V_2\left(x, y, 0\right) = G(x, y) \\ V_2\left(0, y, \tau\right) = G(0, y) \\ V_2\left(x, 0, \tau\right) = G(x, 0) \end{cases}$$

where

$$G(x, y) = \max\left(F(D_0(x)), \begin{cases} V_1\left(x - \delta_1, y, \tau_1\right) & \delta_1 \leq x \\ V_1\left(0, y, \tau_1\right) & x < \delta_1 \end{cases} \right)$$

If there is more than one dividend paid, then the problem will have more parts (intra-dividend intervals) and could be formulated in the similar way.

Above boundary conditions are given for case $\mu_y = 0$. If $\mu_y \neq 0$ then all boundary conditions set on $y = 0$ should be changed to "smoothing" condition (2.23).

Chapter 3

Finite Difference

Let us present convenient scheme for constructing finite difference approximations. Using this scheme one could find various approximations for derivatives having increased exactness, non uniform grid or one sided approximations and etc..

Assume that there are given discrete points (grid points): $z_0, z_1, \ldots, z_N$ and values of function: $V_0, V_1, \ldots, V_N$ defined in each point. Let us construct polynomial of power $N-1$:

$$f\left(z\right) = \sum_{i=0}^{N-1} p_i z^i$$

in order to find polynomial going through indicated points, one should solve the following linear system of equations with respect to p_i:

$$\begin{cases} f\left(z_0\right) = V_0 \\ \vdots \\ f\left(z_N\right) = V_N \end{cases}$$

The system should be solved explicitly and each coefficient will be function: $p_i = F_i(z_0, z_1, \ldots, z_N, V_0, V_1, \ldots, V_N)$. Now in order to receive finite difference approximation for the first and second derivatives in point $z = \tilde{z}$ (note that in general $\tilde{V}$ might not be given for $\tilde{z}$) one should differentiate polynomial with respect to z and substitute the obtained expressions for coefficients:

$$\delta_z\left[\tilde{V}\right] = f'(z)|_{z=\tilde{z},p_1=F_1(\ldots),\ldots,p_{N-1}=F_{N-1}(\ldots)} \tag{3.1a}$$

$$\delta_z^2\left[\tilde{V}\right] = f''(z)|_{z=\tilde{z},p_1=F_1(\ldots),\ldots,p_{N-1}=F_{N-1}(\ldots)} \tag{3.1b}$$

Note that in square brakes we write $\delta_z\left[\tilde{V}\right]$ and this is to be online with continuous derivative notation $f'(z)|_{z=\tilde{z}}$. In general case the above expressions are huge and difficult to work with. As soon as our grid points are uniformly distributed, their number are small and we are interested in finding derivatives in one of the grid points, expression will significantly simplifies.

For example let us assume that there are given three uniformly distributed grid points: $z_0 = 0$, $z_1 = \Delta z$, $z_2 = 2\Delta z$ and in each grid point function values are given: V_0, V_1, V_2. Choosing $f\left(z\right)$ to be parabola we can find its coefficients by solving:

$$\begin{cases} p_0 + p_1 z_0 + p_2 z_0^2 = V_0 \\ p_0 + p_1 z_1 + p_2 z_1^2 = V_1 \\ p_0 + p_1 z_2 + p_2 z_2^2 = V_2 \end{cases}$$

Coefficients could be found explicitly:

$$\begin{aligned} p_0 &= V_0 \\ p_1 &= \frac{-3V_0 + 4V_1 - V_2}{2\Delta z} \\ p_2 &= \frac{V_0 - 2V_1 + V_2}{2\Delta z^2} \end{aligned}$$

In order to find approximation for first and second derivative in central point of grid $\tilde{z} = z_1$ one should substitute above expressions for p_0, p_1, p_2 into (3.1a) and (3.1b). Simplifying them we will get well known central difference approximations:

$$\begin{aligned} \delta_z [V_1] &= \frac{V_2 - V_0}{2\Delta z} \\ \delta_z^2 [V_1] &= \frac{V_2 - 2V_1 + V_0}{\Delta z^2} \end{aligned}$$

Approximation for first and second derivatives in left hand side point of grid $\tilde{z} = z_0$ could be found by substitute expressions for p_0, p_1, p_2 into (3.1a) and (3.1b). In this case we will get so called one sided (left sided) approximation:

$$\begin{aligned} \delta_{-z} [V_0] &= -\frac{3V_0 - 4V_1 + V_2}{2\Delta z} \\ \delta_{-z}^2 [V_0] &= \frac{V_0 - 2V_1 + V_2}{\Delta z^2} \end{aligned}$$

Note that in order to distinguish from central difference approximation we denote this approximation with "-".

In the same manner we can find one sided (right sided) derivatives in right hand side of grid $\tilde{z} = z_2$

$$\begin{aligned} \delta_{+z} [V_2] &= \frac{3V_2 - 4V_1 + V_0}{2\Delta z} \\ \delta_{+z}^2 [V_2] &= \frac{V_2 - 2V_1 + V_0}{\Delta z^2} \end{aligned}$$

and denote it with "+".

Constructing central difference approximation with higher precision goes in the same way. The only difference is that we consider uniform grid with five points $z_0 = 0$, $z_1 = \Delta z$, $z_2 = 2\Delta z$, $z_3 = 3\Delta z$, $z_4 = 4\Delta z$ with corresponding function values V_0, V_1, V_2, V_3, V_4. Solving system of equations 5×5, substituting expressions for coefficients into (3.1a) and (3.1b) for the central point of grid $\tilde{z} = z_2$, one will get:

$$\begin{aligned} \dot{\delta}_z [V_2] &= \frac{V_0 - 8V_1 + 8V_3 - V_4}{12\Delta z} \\ \dot{\delta}_z^2 [V_2] &= \frac{-V_0 + 16V_1 - 30V_2 + 16V_3 - V_4}{12\Delta z^2} \end{aligned}$$

Dot above notation indicates that this approximation has higher exactness. In the appendix C all used derivatives are listed.

For cross derivative one should find coefficients of polynomial $f(z, w) = p_0 + p_1 z + p_2 w + p_3 zw$.

More detailed discussion of finite difference and precision of different approximations could be found in [14].

3.1 Domain

We change our continuous domain $\Omega = \{0 \leq x \leq x_{\max}; 0 \leq y \leq y_{\max}; 0 \leq \tau \leq \tau_{\max}\}$ by discrete domain $\dot{\Omega} = \{x_1, \ldots x_i, \ldots x_{N_x+1}; y_1, \ldots y_j, \ldots y_{N_y+1}; \tau_1, \ldots \tau_k, \ldots \tau_{N_\tau+1}\}$, where

$$\begin{aligned}
i &= 1, ...N_x + 1,\ x_1 = 0,\ x_{N_x+1} = x_{\max},\ x_i - x_{i-1} = \Delta x = \frac{x_{\max}}{N_x} \\
j &= 1, ...N_y + 1,\ y_1 = 0,\ y_{N_y+1} = y_{\max},\ y_i - y_{i-1} = \Delta y = \frac{y_{\max}}{N_y} \\
k &= 1, ...N_\tau + 1,\ \tau_1 = 0,\ \tau_{N_\tau+1} = \tau_{\max},\ \tau_k - \tau_{k-1} = \Delta \tau = \frac{\tau_{\max}}{N_\tau}
\end{aligned}$$

and define discrete function $V_{i,j,k}$ on domain $\dot{\Omega}$.

In order to distinguish between continuous and corresponding discrete function we will always use brackets $V(\tau_k, x_i, y_j)$ for continuous function and subscripts $V_{i,j,k}$ for discrete function defined in point (τ_k, x_i, y_j) (i.e. in grid point i, j, k of domain $\dot{\Omega}$) .

When referring to the discrete function defined on $\dot{\Omega}$ we sometimes omit some of superscripts in order to shorten formulas. For example writing $V_{i,j}$ we assume that k is arbitrarily chosen; writing V_k we assume that i,j are arbitrary and etc. Same notational agreement will be valid for continuous function.

3.2 PDE discretization

Our main goal below will be to replace continuous function $V(\tau, x, y)$ defined on with Ω by discrete function $V_{i,j,k}$ defined on $\dot{\Omega}$, transform partial differential equation: $\frac{\partial V}{\partial \tau} + \mathcal{A}[V] = 0$ into its finite difference replica and finally define initial and boundary conditions for discrete function. As a consequence we will receive linear system of equation w.r.t. unknowns $V_{i,j,k}$. Solution of this system will be considered as finite difference approximation of continuous solution $V(\tau, x, y)$.

Discretization is performed in two steps. First step discretize operator $\mathcal{A}[V]$ and transforms it into linear system of equations (space discretization). Second step discretize τ and finally defines problem as iterative process (time discretization); each iteration gives us solution for corresponding time step.

While constructing finite difference scheme it is important to keep in mind what kind of matrix we are receiving for our linear system of equations. Usually our matrix will be sparse (with lot of zero elements) and diagonal.

3.2.1 Space discretization

Problem with finite difference for cross derivatives is that non of them includes $V_{i,j}$, and thus they do not donate into diagonal superiority of the final finite difference matrix. In order to obtain "good" approximation for cross derivative we are using technique described in [9]. Taylor series expansion reads:

$$\begin{aligned}
V(x_{i+1}, y_{j+1}) &\approx V(x_i, y_j) + \Delta x \frac{\partial V}{\partial x} + \Delta y \frac{\partial V}{\partial y} + \frac{1}{2}\left[\Delta x^2 \frac{\partial^2 V}{\partial x^2} + 2\Delta x \Delta y \frac{\partial^2 V}{\partial x \partial y} + \Delta y^2 \frac{\partial^2 V}{\partial y^2}\right] && (3.2a) \\
V(x_{i-1}, y_{j-1}) &\approx V(x_i, y_j) - \Delta x \frac{\partial V}{\partial x} - \Delta y \frac{\partial V}{\partial y} + \frac{1}{2}\left[\Delta x^2 \frac{\partial^2 V}{\partial x^2} + 2\Delta x \Delta y \frac{\partial^2 V}{\partial x \partial y} + \Delta y^2 \frac{\partial^2 V}{\partial y^2}\right] && (3.2b) \\
V(x_{i+1}, y_{j-1}) &\approx V(x_i, y_j) + \Delta x \frac{\partial V}{\partial x} - \Delta y \frac{\partial V}{\partial y} + \frac{1}{2}\left[\Delta x^2 \frac{\partial^2 V}{\partial x^2} - 2\Delta x \Delta y \frac{\partial^2 V}{\partial x \partial y} + \Delta y^2 \frac{\partial^2 V}{\partial y^2}\right] && (3.2c) \\
V(x_{i-1}, y_{j+1}) &\approx V(x_i, y_j) - \Delta x \frac{\partial V}{\partial x} + \Delta y \frac{\partial V}{\partial y} + \frac{1}{2}\left[\Delta x^2 \frac{\partial^2 V}{\partial x^2} - 2\Delta x \Delta y \frac{\partial^2 V}{\partial x \partial y} + \Delta y^2 \frac{\partial^2 V}{\partial y^2}\right] && (3.2d)
\end{aligned}$$

summing up equations (3.2a), (3.2b) and solving for cross derivative we will get:

$$\frac{\partial^2 V}{\partial x \partial y} \approx \frac{1}{2\Delta x \Delta y}\left[V(x_{i+1}, y_{j+1}) - 2V(x_i, y_j) + V(x_{i-1}, y_{j-1})\right] - \frac{\Delta x}{2\Delta y}\frac{\partial^2 V}{\partial x^2} - \frac{\Delta y}{2\Delta x}\frac{\partial^2 V}{\partial y^2}$$

while summing up the equations (3.2c), (3.2d) and solving for cross derivative:

$$\frac{\partial^2 V}{\partial x \partial y} \approx \frac{-1}{2\Delta x \Delta y}\left[V(x_{i+1}, y_{j-1}) - 2V(x_i, y_j) + V(x_{i-1}, y_{j+1})\right] + \frac{\Delta x}{2\Delta y}\frac{\partial^2 V}{\partial x^2} + \frac{\Delta y}{2\Delta x}\frac{\partial^2 V}{\partial y^2}$$

Finally replacing continuous function with discrete one and substituting finite difference approximations of corresponding continuous derivatives:

$$\begin{aligned}
\hat{\delta}^2_{x,y}\left[V_{i,j}\right] &= \frac{1}{2\Delta x \Delta y}\left[V_{i+1,j+1} - 2V_{i,j} + V_{i-1,j-1}\right] - \frac{\Delta x}{2\Delta y}\delta^2_x\left[V_{i,j}\right] - \frac{\Delta y}{2\Delta x}\delta^2_y\left[V_{i,j}\right] \\
\check{\delta}^2_{x,y}\left[V_{i,j}\right] &= \frac{-1}{2\Delta x \Delta y}\left[V_{i+1,j-1} - 2V_{i,j} + V_{i-1,j+1}\right] + \frac{\Delta x}{2\Delta y}\delta^2_x\left[V_{i,j}\right] + \frac{\Delta y}{2\Delta x}\delta^2_y\left[V_{i,j}\right]
\end{aligned}$$

Choice between $\hat{\delta}^2_{x,y}$ and $\check{\delta}^2_{x,y}$ is determined by the sign of multiplier of cross derivative in (2.13). Idea behind this is that we should try to choose such approximation for cross derivative, that keeps diagonal element of final matrix "heavy" thus providing us with diagonal superiority.

Now we can construct finite difference replica of operator $\mathcal{A}$ in (2.13), but first let us rewrite it in the following form:

$$\mathcal{A}\left[V\right] = a\frac{\partial^2 V}{\partial x^2} + b\frac{\partial^2 V}{\partial x \partial y} + c\frac{\partial^2 V}{\partial y^2} + d\frac{\partial V}{\partial x} + e\frac{\partial V}{\partial y} + fV$$

and note that $a < 0$, $b \gtreqless 0$ if $\rho(\tau) \lesseqgtr 0$, $c < 0$, $d < 0$, $e \gtreqless 0$ if $\mu_y \lesseqgtr 0$, $f > 0$. Now replacing derivatives in continuous operator $\mathcal{A}$ with corresponding finite differences ($\mathbb{I}_{b>0}$ -indicator function):

$$\mathbb{A}_k\left[V_{i,j}\right] = a\delta^2_x\left[V_{i,j}\right] + \mathbb{I}_{b>0} b\hat{\delta}^2_{x,y}\left[V_{i,j}\right] + \left(1 - \mathbb{I}_{b>0}\right) b\check{\delta}^2_{x,y}\left[V_{i,j}\right] + c\delta^2_y\left[V_{i,j}\right] + d\delta_x\left[V_{i,j}\right] + e\delta_y\left[V_{i,j}\right] + fV_{i,j} \quad (3.3)$$

substituting expressions for finite difference, collecting and rearranging terms we will finally get:

$$\begin{aligned}
\mathbb{A}_k\left[V_{i,j}\right] = \; & V_{i,j}\left[f - \frac{2a}{\Delta x^2} - \frac{2c}{\Delta y^2} + \frac{b\left(2\mathbb{I}_{b>0} - 1\right)}{\Delta x \Delta y}\right] + \\
& V_{i-1,j-1}\left[\frac{b\mathbb{I}_{b>0}}{2\Delta x \Delta y}\right] + V_{i,j-1}\left[\frac{c}{\Delta y^2} - \frac{e}{2\Delta y} - \frac{b\left(2\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right] + \\
& V_{i+1,j-1}\left[\frac{b\left(\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right] + V_{i+1,j}\left[\frac{a}{\Delta x^2} + \frac{d}{2\Delta x} - \frac{b\left(2\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right] + \\
& V_{i+1,j+1}\left[\frac{b\mathbb{I}_{b>0}}{2\Delta x \Delta y}\right] + V_{i,j+1}\left[\frac{c}{\Delta y^2} + \frac{e}{2\Delta y} - \frac{b\left(2\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right] + \\
& V_{i-1,j+1}\left[\frac{b\left(\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right] + V_{i-1,j}\left[\frac{a}{\Delta x^2} - \frac{d}{2\Delta x} - \frac{b\left(2\mathbb{I}_{b>0} - 1\right)}{2\Delta x \Delta y}\right]
\end{aligned}$$

Subscript k in the operator indicates that in general case coefficients of finite difference operator is time dependent. Applying operator $\mathbb{A}_k\left[V_{i,j}\right]$ (for fixed time step k) for each inner point of domain $\dot{\Omega}$ we will get linear system of equations and denote matrix associated with this system by $\mathbb{A}_k$ (without square brackets).

It should be mentioned separately that constructing finite difference replica of operator $\mathcal{A}$ could be performed using more precise approximating scheme:

$$\dot{\mathbb{A}}_k\left[V_{i,j}\right] = a\dot{\delta}^2_x\left[V_{i,j}\right] + \mathbb{I}_{b>0} b\hat{\delta}^2_{x,y}\left[V_{i,j}\right] + \left(1 - \mathbb{I}_{b>0}\right) b\check{\delta}^2_{x,y}\left[V_{i,j}\right] + c\dot{\delta}^2_y\left[V_{i,j}\right] + d\dot{\delta}_x\left[V_{i,j}\right] + e\dot{\delta}_y\left[V_{i,j}\right] + fV_{i,j} \quad (3.4)$$

Problem with this approximation is that it involves more "off diagonal" elements in stencil, thus significantly decreasing property of matrix diagonal superiority.

3.2.2 Time Discretization

Replacing continuous derivative $\frac{\partial V}{\partial \tau}$ with one sided finite differences $\delta_{+\tau}[V_k]$, $\dot{\delta}_{+\tau}[V_k]$, $\ddot{\delta}_{+\tau}[V_k]$ we will get following iterative scheme for finding $V_{i,j,k}$:

$$\begin{aligned}
&(I+\Delta\tau\mathbb{A}_2)\,V_2 = V_1\\
&\left(I+\frac{2}{3}\Delta\tau\mathbb{A}_3\right)V_3 = \frac{4}{3}V_2-\frac{1}{3}V_1\\
&\left(I+\frac{6}{11}\Delta\tau\mathbb{A}_k\right)V_k = \frac{18}{11}V_{k-1}-\frac{9}{11}V_{k-2}+\frac{2}{11}V_{k-3} \text{ for } k=4,\dots N_T+1
\end{aligned}$$

Note that above equations are written in matrix form, V_1 is given by initial conditions and I is identity matrix.

Finally we will denote iterative finite difference scheme by:

$$(I+\Delta_k\mathbb{A}_k)\,V_k = l_k^1V_{k-1}+l_k^2V_{k-2}+l_k^3V_{k-3} \tag{3.5}$$

which we assume to hold in all inner points of domain $\dot{\Omega}$. This type of scheme usually called BDF3 (backward difference formula with 3 time steps). Another types of scheme are described in [9]

BDF3 allows to manage diagonal superiority of matrix by means of decreasing $\Delta\tau$, but "price" to be paid for this will be increased number of steps (iteration) and as a consequence increased time required for solution.

3.2.3 Discrete Splitting

Now in order to adjust the operator splitting method described in section 2.4 to numerical calculation of price of the American option, we divide the method into two steps.

In the first step system of linear equation is solved, in second step an intermediate solution and auxiliary variable are updated in such a way that they satisfy constrains. Intermediate solution V_k should be greater then or equal to obstacle function $F(D_0(x_i))$, while λ_k is required to be positive. System of linear equations to be solved is:

$$(I+\Delta_k\mathbb{A}_k)\,\tilde{V}_k = l_k^1V_{k-1}+l_k^2V_{k-2}+l_k^3V_{k-3}+\Delta_k\lambda_{k-1} \tag{3.6}$$

where $\tilde{V}_k$ is intermediate solution.

Constrains are written as:

$$\begin{cases}
V_k-\tilde{V}_k-\Delta_k(\lambda_k-\lambda_{k-1})=0\\
(\lambda_k)^T\left(V_k-F(D_0(x_i))\right)=0\\
V_k\geq F(D_0(x_i)), \quad \lambda_k\geq 0
\end{cases}$$

or:

$$\begin{aligned}
\lambda_k &= \max\left[\frac{F(D_0(x_i))-\tilde{V}_k}{\Delta_k}+\lambda_{k-1},0\right]\\
V_k &= \tilde{V}_k+\Delta_k(\lambda_k-\lambda_{k-1})
\end{aligned}$$

where V_k is solution and λ_k is auxiliary variable.

For initial guess we take $\lambda_1=0$, as the American option at maturity is exercised immediately.

Accuracy consideration for operator splitting method is described in [9].

3.3 Discretization of Initial and Boundary conditions

As it was mentioned above finding exact conditions reflecting real behavior of option price on boundaries $x=x_{\max}$ and $y=y_{\max}$ might be difficult, if ever possible. While setting conditions on $x=x_{\max}$ and

$y = y_{\max}$ one should always keep in mind that they are only "approximately correct". In this section we will define a few new conditions and in section 5 we will use and compare them.

Below we translate all continuous boundary and initial conditions listed above into their finite difference replicas. Also we introduce some new conditions and also translate them. New conditions will be first defined for continuous case and then translated into finite difference.

3.3.1 Initial condition $\tau = 0$ $(\mathbf{k} = \mathbf{1})$

Initial condition for European or American option is similar to each other (2.17), (2.25). It could be translated in a finite difference:

$$V_{i,j,1} = F(D_0(x_i)) \tag{3.7}$$

Thus the first solution (zero solution) could be found directly from initial conditions.

3.3.2 Bound: $x = 0$ $(\mathbf{i} = \mathbf{1})$

For the European option we translate Dirichlet condition (2.18) set on the bound and adjacent corners $x = 0, y = 0$ and $x = 0, y = y_{\max}$ into discrete version for function $V_{i,j,k}$:

$$V_{1,j,k} = F(D_0(x_1)) \exp\left(-\int\limits_0^{\tau_k} r(T-s)ds\right) \tag{3.8}$$

For the American option (2.26):

$$V_{1,j,k} = F(D_0(x_1)) \tag{3.9}$$

3.3.3 Bound: $y = 0$ $(\mathbf{j} = \mathbf{1})$

Conditions on this bound are chosen differently for zero volatility drift term and non zero volatility drift term.

When $\mu_y = 0$ we translate Dirichlet condition for the European option (2.20) on the bound and adjacent corner $x = x_{\max}, y = 0$ into:

$$V_{i,1,k} = F\left(D_0(x_i) \exp\left(\int\limits_0^{\tau_k} (r(T-s) - q(T-s))\, ds\right)\right) \exp\left(-\int\limits_0^{\tau_k} r(T-s)ds\right) \tag{3.10}$$

For the American option condition (2.28) is translated into:

$$V_{i,1,k} = F\left(D_0(x_i)\right) \tag{3.11}$$

If $\mu_y \neq 0$ condition (2.23) for the European, and condition (2.30) for the American are translated (excluding adjacent corners), into:

$$\left(I + \Delta_k \mathbb{A}_k^{y=0}\right) V_k = l_k^1 V_{k-1} + l_k^2 V_{k-2} + l_k^3 V_{k-3} \tag{3.12}$$

where matrix $\mathbb{A}_k^{y=0}$ is constructed using one sided finite differences w.r.t. y in operator $\mathcal{A}$:

$$\begin{aligned} \mathbb{A}_k^{y=0}\left[V_{i,1}\right] &= a\delta_x^2\left[V_{i,1}\right] + b\delta_{x,+y}^2\left[V_{i,1}\right] + c\delta_{+y}^2\left[V_{i,1}\right] + \\ &\quad d\delta_x\left[V_{i,1}\right] + e\delta_{+y}\left[V_{i,1}\right] + fV_{i,1} \end{aligned}$$

This approximation is changing stencil and allows us to use it on the bound, otherwise we would face a problem with external grid points.

3.3.4 Bound: $x{=}x_{\max}$ $(\mathbf{i} = \mathbf{N}_x + 1)$

Few possible conditions could be set on this bound (excluding adjacent corners). Let us first of all translate those ones described in previous sections. Neuman conditions (2.22a) for both the American and European put option cases are same and could be translated into:

$$\delta_{+x}\left[V_{N_x+1,j}\right] D_1\left(x_{N_x+1}\right) = 0, \tag{3.13}$$

for the call option we set above to be equal to one. Another possibility is to translate equivalent conditions (2.19) and (2.27) into:

$$\delta^2_{+x}\left[V_{N_x+1,j}\right] D^2_1\left(x_{N_x+1}\right) + \delta_{+x}\left[V_{N_x+1,j}\right] D_2\left(x_{N_x+1}\right) = 0 \tag{3.14}$$

Note that this condition is held for both put and call option.

Now let us describe alternative conditions (not stated in continuous form). First one is to set smoothing conditions, i.e. to translate continuous operator

$$\mathcal{A}\left[V\left(x_{\max}, y, \tau\right)\right] + \frac{\partial V\left(x_{\max}, y, \tau\right)}{\partial \tau} = 0$$

into its finite difference replica:

$$\left(I + \Delta_k \mathbb{A}^{x=x_{\max}}_k\right) V_k = l^1_k V_{k-1} + l^2_k V_{k-2} + l^3_k V_{k-3} \tag{3.15}$$

where matrix $\mathbb{A}^{x=x_{\max}}_k$is constructed using one sided finite differences w.r.t. x in operator $\mathcal{A}$:

$$\begin{aligned}\mathbb{A}^{x=x_{\max}}_k\left[V_{N_x+1,j}\right] \quad = \quad & a\delta^2_{+x}\left[V_{N_x+1,j}\right] + b\delta^2_{+x,y}\left[V_{N_x+1,j}\right] + c\delta^2_{y}\left[V_{N_x+1,j}\right] + \\ & d\delta_{+x}\left[V_{N_x+1,j}\right] + e\delta_{y}\left[V_{N_x+1,j}\right] + fV_{N_x+1,j}\end{aligned}$$

Second possibility is to set (for put option only) smoothing condition and Neuman condition simultaneously:

$$\begin{aligned}\mathcal{A}\left[V\left(x_{\max}, y, \tau\right)\right] + \frac{\partial V\left(x_{\max}, y, \tau\right)}{\partial \tau} &= 0 \\ \frac{\partial V\left(x_{\max}, y, \tau\right)}{\partial x} &= 0\end{aligned}$$

translating this we will get two conditions:

$$\begin{aligned}&\left(I + \Delta_k \mathbb{A}_k\right) V_k = l^1_k V_{k-1} + l^2_k V_{k-2} + l^3_k V_{k-3} \\ &\delta_{+x}\left[V_{N_x+1,j}\right] D_1\left(x_{N_x+1}\right) = 0\end{aligned} \tag{3.16}$$

Note that simultaneously setting these two finite difference conditions on the boundary is equal to assuming that all external grid points are equal to corresponding inner points: $V_{N_x+2,j} = V_{N_x,j}$ for all j. This condition is described in [9].

3.3.5 Corner: $x{=}x_{\max}$, $y{=}\,\mathbf{0}$ $(\mathbf{i} = \mathbf{N}_x + 1,\ \mathbf{j} = \mathbf{1})$

Lower corner point is treated in two different ways. If $\mu_y = 0$ then Dirichlet condition (3.10) or (3.11) should be set.

If $\mu_y \neq 0$, the following conditions could be set and translated into finite differences. The first one is usual Neuman condition

$$\frac{\partial V\left(x_{\max}, 0, \tau\right)}{\partial x} = 0$$

which will translate into:

$$\delta_{+x}\left[V_{N_x+1,1}\right] D_1\left(x_{N_x+1}\right) = 0 \tag{3.17}$$

For the call option instead of zero 1 is substituted.

Second one is smoothing condition:

$$\mathcal{A}\left[V\left(x_{\max},0,\tau\right)\right]+\frac{\partial V\left(x_{\max},0,\tau\right)}{\partial\tau}=0$$

translated into:

$$\left(I+\Delta_k\mathbb{A}_k^{x=x_{\max},y=0}\right)V_k=l_k^1V_{k-1}+l_k^2V_{k-2}+l_k^3V_{k-3}\tag{3.18}$$

Where

$$\begin{aligned}\mathbb{A}_k^{x=x_{\max},y=0}\left[V_{N_x+1,1}\right] &= a\delta_{+x}^2\left[V_{N_x+1,1}\right]+b\delta_{+x,+y}^2\left[V_{N_x+1,1}\right]+c\delta_{+y}^2\left[V_{N_x+1,1}\right]+\\ &\quad d\delta_{+x}\left[V_{N_x+1,1}\right]+e\delta_{+y}\left[V_{N_x+1,1}\right]+fV_{N_x+1,1}\end{aligned}$$

and again we are using one sided finite difference in order to exclude the external grid points from the finite difference operator.

3.3.6 Corner: x=$x_{\max}$, y=$y_{\max}$ $\left(\mathbf{i}=\mathbf{N}_x+1,\ \mathbf{j}=\mathbf{N}_y+\mathbf{1}\right)$

For the upper corner point few possible conditions could be set. The first one is usual Neuman condition w.r.t. x:

$$\frac{\partial V\left(x_{\max},0,\tau\right)}{\partial x}=0$$

translated into:

$$\delta_{+x}\left[V_{N_x+1,N_y+1}\right]D_1\left(x_{N_x+1}\right)=0\tag{3.19}$$

For the call option instead of zero, 1 should be substituted.

The second one is a smoothing condition:

$$\mathcal{A}\left[V\left(x_{\max},y_{\max},\tau\right)\right]+\frac{\partial V\left(x_{\max},y_{\max},\tau\right)}{\partial\tau}=0$$

translated into finite difference replica using one sided derivatives:

$$\left(I+\Delta_k\mathbb{A}_k^{x=x_{\max},y=y_{\max}}\right)V_k=l_k^1V_{k-1}+l_k^2V_{k-2}+l_k^3V_{k-3}\tag{3.20}$$

where

$$\begin{aligned}\mathbb{A}_k^{x=x_{\max},y=y_{\max}}\left[V_{N_x+1,N_y+1}\right] &= a\delta_{+x}^2\left[V_{N_x+1,N_y+1}\right]+b\delta_{+x,+y}^2\left[V_{N_x+1,N_y+1}\right]+c\delta_{+y}^2\left[V_{N_x+1,N_y+1}\right]+\\ &\quad d\delta_{+x}\left[V_{N_x+1,N_y+1}\right]+e\delta_{+y}\left[V_{N_x+1,N_y+1}\right]+fV_{N_x+1,N_y+1}\end{aligned}$$

The third possibility is to set smoothing and two Neuman conditions simultaneously:

$$\begin{aligned}\mathcal{A}\left[V\left(x_{\max},y_{\max},\tau\right)\right]+\frac{\partial V\left(x_{\max},y_{\max},\tau\right)}{\partial\tau} &= 0\\ \frac{\partial V\left(x_{\max},y_{\max},\tau\right)}{\partial x} &= 0\\ \frac{\partial V\left(x_{\max},y_{\max},\tau\right)}{\partial y} &= 0\end{aligned}$$

replicating these into finite difference we will get:

$$\begin{aligned}&\left(I+\Delta_k\mathbb{A}_k\right)V_k=l_k^1V_{k-1}+l_k^2V_{k-2}+l_k^3V_{k-3}\\ &\delta_{+x}\left[V_{N_x+1,N_y+1}\right]D_1\left(x_{N_x+1}\right)=0\\ &\delta_{+y}\left[V_{N_x+1,N_y+1}\right]G_1\left(y_{N_x+1}\right)=0\end{aligned}\tag{3.21}$$

note that setting additional Neuman conditions in the corner is equal to assuming that external grid points $V_{N_x+2,N_y+2}=V_{N_x,N_y}$, $V_{N_x+2,N_y+1}=V_{N_x,N_y+1}$, $V_{N_x+1,N_y+2}=V_{N_x+1,N_y}$ [9].

3.3.7 Bound: $y{=}y_{\max}$ $\quad \left(\mathbf{j} = \mathbf{N}_y{+}\mathbf{1}\right)$

Smoothing conditions (2.21) and (2.29) are equal and could be translated into:

$$\left(I + \Delta_k \mathbb{A}_k^{y=y_{\max}}\right) V_k = l_k^1 V_{k-1} + l_k^2 V_{k-2} + l_k^3 V_{k-3} \tag{3.22}$$

where using one sided derivatives we can get:

$$\begin{aligned} \mathbb{A}_k^{y=y_{\max}} \left[V_{i,N_y+1}\right] &= a\delta_x^2 \left[V_{i,N_y+1}\right] + b\delta_{x,-y}^2 \left[V_{i,N_y+1}\right] + c\delta_{-y}^2 \left[V_{i,N_y+1}\right] + \\ &\quad d\delta_x \left[V_{i,N_y+1}\right] + e\delta_{-y} \left[V_{i,N_y+1}\right] + fV_{i,N_y+1} \end{aligned}$$

finite difference operator with excluded external grid points.

The second possibility is to require additionally Neuman condition to be satisfied:

$$\begin{aligned} \mathcal{A}\left[V\left(x, y_{\max}, \tau\right)\right] + \frac{\partial V\left(x, y_{\max}, \tau\right)}{\partial \tau} &= 0 \\ \frac{\partial V\left(x, y_{\max}, \tau\right)}{\partial y} &= 0 \end{aligned}$$

and this is translated into:

$$\begin{aligned} &\left(I + \Delta_k \mathbb{A}_k\right) V_k = l_k^1 V_{k-1} + l_k^2 V_{k-2} + l_k^3 V_{k-3} \\ &\delta_{+y} \left[V_{i,N_y+1}\right] G_1\left(y_{N_x+1}\right) = 0 \end{aligned} \tag{3.23}$$

note that setting two conditions on the boundary is equal to assuming that all external grid points are equal to corresponding internal grid points: $V_{i,N_y+2} = V_{i,N_y}$ for all i [9].

Chapter 4

Monte Carlo

In this chapter we are describing well known and widely used Monte Carlo method for option pricing. In case of American option we are using modification of Monte Carlo method (least square approach) described in [12].

4.1 European option

Key idea of Monte Carlo method is to simulate M paths of the underlying asset and based on simulated paths derive the price of option under the "extended" SABR model. This method is known to be very time consuming but easy to apply, even for the options with path-dependent price. First we will discuss implementation of the European option case for stock paying cash dividends.

Let us divide interval $[0, T]$ into N equidistant parts $\Delta t = \frac{T}{N}$:

$$t_1 = 0, t_2 = \Delta t, \ldots, t_{N+1} = N\Delta t = T$$

and calculate the paths for volatility and stock price using:

$$\begin{aligned}
&\alpha_{k+1} = \alpha_k + \mu_\alpha(\alpha_k, t_k)\Delta t + \sigma_\alpha(\alpha_k, t_k)\left[\rho(t_k)\epsilon_1(t_k) + \sqrt{1-\rho^2(t_k)}\epsilon_2(t_k)\right]\sqrt{\Delta t};\\
&S_{k+1} = S_k + \mu_S(S_k, \alpha_k, t_k)\Delta t + \sigma_S(S_k, \alpha_k, t_k)\epsilon_1(t_k)\sqrt{\Delta t};\\
&\text{here } k = 1, 2, \ldots, N; \text{ and } \alpha_1 = \alpha, \quad S_1 = S
\end{aligned}$$

where $\mu_S(S, \alpha, t) = (r(t) - q(t))S_t$, $\sigma_S(S, \alpha, t) = \alpha_t (S_t)^\beta$, $\mu_\alpha(\alpha, t) = \kappa(\theta(t) - \alpha_t)$, $\sigma_\alpha(\alpha, t) = \nu\alpha_t$. Expression $\left[\rho(t_k)\epsilon_1(t_k) + \sqrt{1-\rho^2(t_k)}\epsilon_2(t_k)\right]$ appears because Wiener processes are correlated[1], while $\epsilon_1(t_k)$ and $\epsilon_2(t_k)$ are two independent random samples from standard normal distribution. Note that if dividend is paid out at t_k then the price of stock "just before" dividend is calculated according to given formula for S_{k+1} while the price of stock "just after" dividend is $\tilde{S}_{k+1} = \max(S_{k+1} - \delta_{t_k}, 0)$ and after this value of $\tilde{S}_{k+1}$ is used to calculate next S_{k+2}.

After generating path for stock price one can immediately calculate put (or any other claim) payoff for each generated path using:

$$V = \exp\left(-\int_0^T r(s)ds\right)(K - S_{N+1})^+$$

[1]The Cholesky decomposition is commonly used in the Monte Carlo method for simulating systems with multiple correlated variables. The matrix of inter-variable correlations is decomposed, to give the lower-triangular matrix. Applying this to a vector of uncorrelated samples, produces a new sample vector with the covariance properties of the system being modeled.

Since M paths were generated, one should find mean μ_V and standard deviation σ_V of V in order to estimate confidence interval for price of option:

$$\left[\mu_V - N^{-1}\left(1-0.05/2\right)\frac{\sigma_V}{\sqrt{M}};\quad \mu_V + N^{-1}\left(1-0.05/2\right)\frac{\sigma_V}{\sqrt{M}}\right]$$

4.2 American option

In case of the American option, Monte Carlo simulation can not be used directly. While we are calculating value of American option, price of immediate exercise should always be compared with expected cash flow from continuing. However, expected cash flow from continuing could not be found directly from Monte Carlo simulation.

The method to derive prices of American options using Monte Carlo simulations starts in the same way as the one described above for European options, namely simulating M paths of the underlying asset S. Again the life of the option can be divided into N short time intervals Δt, and paths for volatility and stock price can be approximated as before. The difference is now that the holder of the option can also choose to exercise the option at each moment in time between time zero and time T (option is exercised only once). This means for the approximation that at each time step it has to be evaluated if exercising at that moment gives a higher payoff than the expected discounted payoff of holding the option at least one more time step. The payoff of exercising at time t_k is easy to determine, since this decision can only be made at time t_k itself. So the value of the stock at time t_k is known, and the payoff of exercising the option can be easily computed.

The expected discounted payoff of continuing however is far more difficult to calculate. Longstaff and Schwartz in [12] provide a way to approximate this value when Monte Carlo simulation is used, namely using modification called the Least-Squares Monte Carlo (LSM) method. There are other methods based on Monte Carlo simulation, like the one proposed by Andersen [1], but the LSM method is easier to apply to models with multiple stochastic factors, and has a good trade-off between computational time and precision. That is why the Least-Squares Monte Carlo method is used here to derive prices of American options under the "extended" and "classical" SABR models.

After sampling M paths for S, there are M possible values for each S_k. First the option payoff for each path when exercising at $t_{N+1} = T$ is derived. After that, all paths for which the option is in-the-money at time t_N are considered, which forms a set Z_N.

Now the key idea of Longstaff and Schwartz [12] comes into play. They assume an approximate relationship between the conditional expected value of continuing and the value of the stock:

$$V_N \approx \sum_{i=1}^{j} l_N^i f^i\left(S_N\right) \tag{4.1}$$

where V_N is the approximated value of continuing discounted back to the point t_N, S_N is the value of the stock at time t_N, l_N^i are constants, $f^i\left(\cdot\right)$ is the i'th function of a chosen set of basis functions (like Laguerre, Legendre polynomials or any other set of orthogonal or usual functions) and j is the number of basis functions.

To find the constants l_N^i, the following functional is minimized:

$$\sum_{z\in Z_N}\left[V_N^z - \sum_{i=1}^{j} l_N^i f^i\left(S_N^z\right)\right]^2$$

where V_N^z is the value of continuing with path z at time t_N discounted back to time t_N, and S_N^z is the stock price of path z at time t_N.

Now equation (4.1) gives the expected payoff when continuing at t_N, and these values are compared with the pay-off of exercising at time t_N. With this data the decision to exercise or not at t_N can be taken for each path. Of course the option will not be exercised at t_N, when it is out of the money at t_N.

After this step, all paths for which the option is in the money at time t_{N-1} are considered, and so on until time $t_1 = 0$. At every time step functional

$$\sum_{z \in Z_k} \left[V_k^z - \sum_{i=1}^{j} l_k^i f^i \left(S_k^z \right) \right]^2$$

is repeatedly minimized, to derive all values of l_k^i, where Z_k is the set of paths for which the option is in the money at time t_k. The value of continuing is again compared with the value of exercising at time t_k.

At the end of this procedure, each generated path has one exercise time. These M payoffs should all be discounted to time t_1 and averaged. This will give the value of the option with the LSM method suggested by Longstaff and Schwartz.

A detailed numerical example of this LSM method can be found in the [12].

Chapter 5

Numerical Experiment

To perform reliable numerical experiment we need to compare not only different boundary conditions and choose those ones which will give us higher exactness, but also different finite difference schemes. As a measure of exactness we will choose the residue between existent analytical solution and solution obtained by finite difference method. Of course, analytical solution does not exist for general choice of parameters (otherwise there would be no reason in constructing finite differences), but there exists analytical solution for some special cases. As soon as we are certain about the exactness of numerical solution for this special cases, we can assume that it will work also for the general choice of parameters. However, in last case we additionally compare two different numerical methods (Finite Difference and Monte Carlo).

5.1 Boundary Conditions

As it was mentioned above, boundary conditions could be conventionally divided into two groups.

The first group includes all conditions that are given in analytical form, for example (2.20), (2.18) for the European option and (2.28), (3.9) for the American option. These conditions could not be somehow improved or perfected, as they are only possible and unique conditions.

The second group includes conditions that are only "approximately" correct. We are facing problem with this type of conditions due to the fact that initially the problem is set on semi-infinite domain. Since we are using finite difference method we have to consider finite domain by means of "cutting off" infinite parts, thus condition on infinity should be replaced by some "approximate" conditions on finite boundaries. Additionally, there is always a balance between "numerical" and "mathematical" levels of correctness for the condition. For example, requiring on $x = x_{\max}$, $\Gamma = 0$ seems to be equally correct, as to require (for put option) $\Delta = 0$. Though, one should keep in mind that Γ converges to zero faster then Δ, but finite difference approximation (with same amount of grid points) is more precise for Δ. In our particular case we are using the condition $\Delta = 0$, since we are performing variable transformation and for very large $x_{\max}$ there is no "numerical" difference between choosing $\Gamma = 0$ or $\Delta = 0$ and thus we are choosing last one as a simpler. On the boundary $x_{\max}$ we can also set "smoothing" condition separately (3.15) or together with $\Delta = 0$ (3.16). During the numerical experiment it was found that setting only "smoothing" condition on this bound gave higher error then setting coupled conditions.

Choosing the appropriate condition for $y = y_{\max}$ is a bit tricky, since Vega - $\mathcal{V}$ is not uniformly converging to zero when volatility increases, in contrast with Δ for example. Thus setting $\mathcal{V} = 0$ separately or coupled with "smoothing" conditions (3.23) could be justified only for very small or very big volatilities. It was found that "smoothing" condition alone (3.22) gave best result and proved to be multipurpose (for small, mid and big volatilities $y_{\max}$).

5.2 Finite Difference Schemes

Setting perfect boundary and initial conditions does not guarantee nice numerical approximation. Constructing replica of differential operator $\mathcal{A}$ is the first and might be the most important task while working with finite difference method. In present research, two different replicas of continuous operator are considered: (3.3) and (3.4). The first one involves "usual" central differences, the last one uses central differences with more grid points and, as a result, higher precision. Price to pay for using larger stencil is additional computational time. And this is caused by the fact that we are obtaining finite difference matrix with "lighter" diagonal and larger number of nonzero elements. "Diagonal superiority" could be managed by decreasing time step size (3.5), but there is noting that could decrease the memory usage for non zero elements additionally occupied by bigger stencil.

Experimenting with different step sizes and two mentioned stencils, it was found that it is better to decrease step size and use smaller stencil rather than use larger one.

5.3 Variable Transformation

Variable transformation allows us to resolve a few problems. The first is that boundary conditions set on $x = x_{\max}$ and $y = y_{\max}$ should imitate behavior of option price on infinity. Thus, the bigger values for $x_{\max}$ and $y_{\max}$ are chosen, the better imitation will be. While choosing larger domain, we need to dramatically increase the number of grid points and that is the main drawback. In order to avoid this problem we should apply variable transformation, which allows us to concentrate grid points in the domain of interest and "send" boundary to infinity.

The second is that usually we are interested in the increasing exactness of the solution for some particular parts of the whole domain (for example for "At The Money Option"). For this reason we should refine grid points in this parts and this is possible only by applying transformation.

It was found that the most convenient transformation is (2.16) since it allows of successfully resolving both of the mentioned problems. For transformation with respect to x parameters p_1, p_2, p_3 of (2.16) should be chosen in a way to concentrate (refine) grid points near strike and coarse grid points on the boundary $x = x_{\max}$.

5.4 Comparison

In this section we will find price for the European put option with following data: $x_{\max} = 600$, $y_{\max} = 2$, $\tau_{\max} = 1$, $K = 25$, $N_x = 200$, $N_y = 200$, $N_T = 100$, $r = 0.03$, $q = 0$ and numerical solution will be compared with existing analytical solution (for special cases) or with numerical solution obtained by Monte Carlo simulation (for general case).

5.4.1 SABR model $\left(\mu_y = 0 \iff \kappa = \theta = 0\right)$

Coefficients of the system (2.1) are chosen to be constant and $\rho = 0.2$, $\beta = 0.9$, $\kappa = 0.0$, $\theta = 0.0$, $\nu = 0.4$. For this model we are choosing as an "analytical solution" the price of put found by equation for volatility surface (1.3) suggested by Hagan in [5] which is proved to be very precise analytical approximation. Our finite difference scheme is defined by (3.5), (3.7), (3.8), (3.10), (3.13), (3.19), (3.22). The residual (error) between numerical and analytical solutions for $\tau = 1$ is presented in Figure 5.1. Maximal absolute error of solution is concentrated in domain (filled with gray color) with high volatility and big time to maturity (for $\tau < 1$ maximal error is few times smaller). Also there is concentration of error near point with $S = K = 25$ and $\alpha = 0$, though absolute value of error is smaller then 0.003. Discussion of errors and recipes for them are given below.

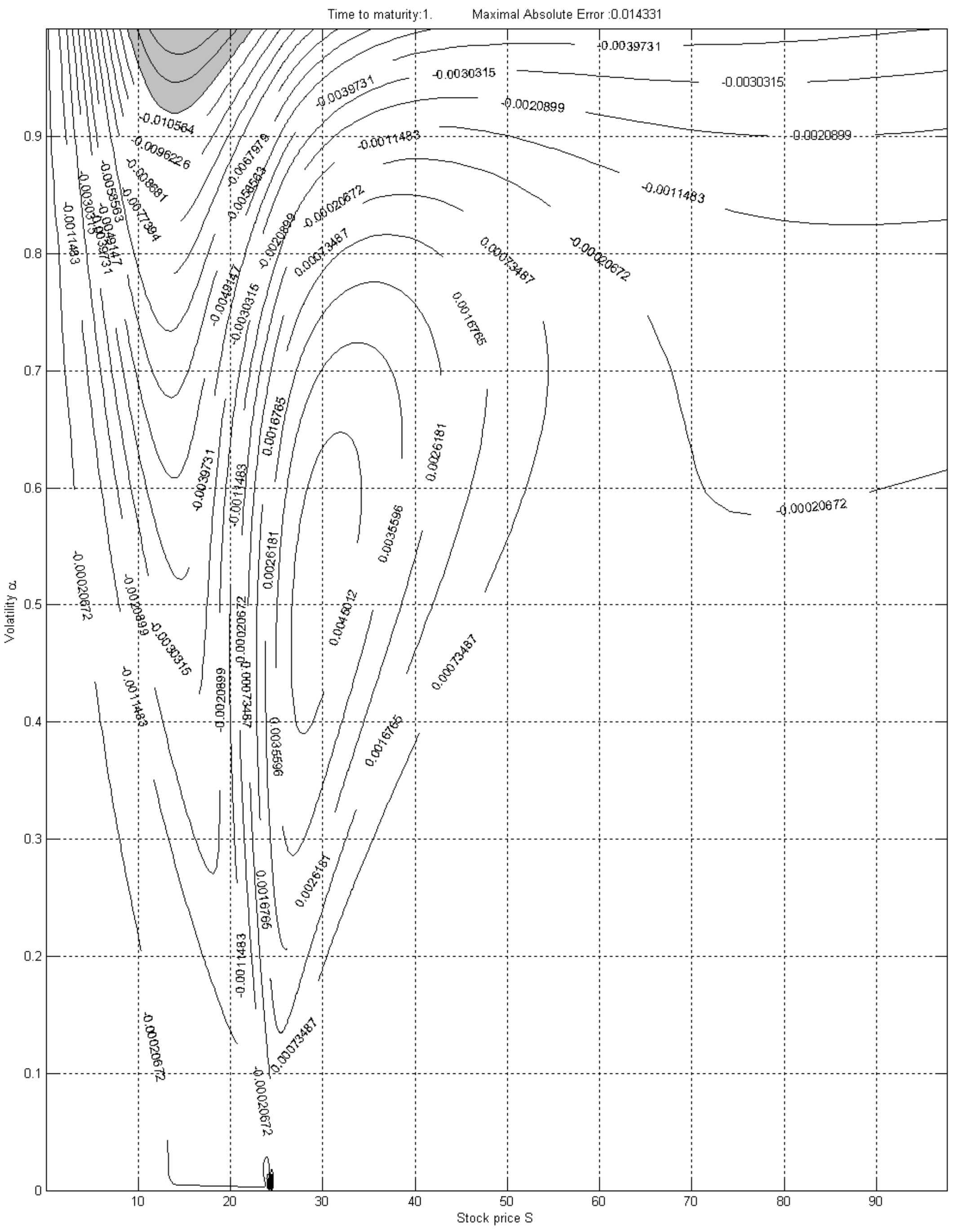


Figure 5.1: The residual between numerical and analytical solutions for the "classical" SABR model.

5.4.2 Non stochastic volatility ($\beta = 1, \quad \sigma_y = 0 \iff \nu = 0$)

Coefficients of system (2.1) are chosen to be: $\rho = 0$, $\beta = 1$, $\kappa = 0.25$, $\theta = 0.8$, $\nu = 0$. In this case volatility is time dependent but non stochastic ($\nu = 0$). There exist analytical solution for this problem, in particular averaging variance w.r.t. time i.e.:

$$\bar{\sigma}^2(t) = \frac{1}{t}\int\limits_0^t \sigma^2(s)\,ds \tag{5.1}$$

and substituting $\bar{\sigma}$ into Black Scholes formula for put one can immediately find analytical price. Our finite difference scheme is: (3.5), (3.7), (3.8), (3.12), (3.17), (3.13), (3.19), (3.22). Residual between numerical and analytical solutions for $\tau = 1$ is presented in Figure 5.2. First of all one can immediately see that maximal absolute error for this problem approximately 10 times smaller compared to Figure 5.1. Domain of maximal error is again filled with gray color. It is interesting to note that there is no error concentration for at the money option with zero volatility ($S = K = 25$ and $\alpha = 0$) like it is in the previous case.

5.4.3 Monte Carlo

Coefficients of system (2.1) are chosen to be: $\rho = 0.2$, $\beta = 1$, $\kappa = 0.25$, $\theta = 0.8$, $\nu = 0.4$. We are comparing finite difference solution with the one obtained by Monte-Carlo simulation. Due to the fact that Monte-Carlo is computationally very expensive method, we are comparing numerical solutions in: $y = G(0.5)$, $x = D(25)$. Comparison is made for both European and American puts Figure 5.3. It should be mentioned that since both pricing methods are numerical, they bear some error, thus qualitative behavior of presented plots are more informative rather then purely quantitative measuring. Qualitative behavior shows that both methods give nice pricing and real values should be in range for American and European options.

Operator splitting method for the American option guarantees that price obtained will always be greater than obstacle function. From Figure 5.3 one can also see that price of American option is greater than price of European option. In Monte Carlo method adapted for American option (least square approach), this feature is not "built-in" in the algorithm, thus price obtained for the American option might be lower then price of the European option if insufficient number of simulations performed. From Figure 5.3 one can also note that $100,000$ simulations were not enough to provide this feature.

5.5 Error types and recipes

We can classify errors associated with finite difference algorithm and bring some recipes for each of them:

1. **Initial Error:** This type of error is usually caused by discontinuity for put and/or call options of $\Delta\big|_{x=D(K)}$ for $\tau = 0$ (initial conditions) and concentrated near small values of τ and $x \approx D(K)$. In order to decrease this error regularization (smoothing) of initial conditions could be applied. Replacing payoff function $F(\cdot)$ in (3.7) with the price of option (put or call) given by classical B-S equation for small values of τ i.e.: $V_{i,j,1} = P(D_0(x_i), K, G_0(y_j), T - \tilde{\tau})$ where $0 < \tilde{\tau} \ll 1$, this error could be eliminated. Another method for decreasing this type of error is to refine mesh near $x = D(K)$ using, for example, transformation (2.16). It was found that mesh refining could be applied directly, but requires some additional computational resources. Regularization of initial conditions is effective but requires carefully choose of $\tilde{\tau}$.

2. **Strike Error:** This error is caused by discontinuity of Δ at the boundary $y = 0$. Error arises when we set Dirichlet condition (only for the case when mean reverting term $\mu_y = 0$) on the boundary $y = 0$ and concentrating near $y = 0$, $x = K$. Like in previous error type, regularization (smoothing) technique could be applied for this type of error. Another recipe is to refine mesh near $x = D(K)$ as in the previous case.

3. **Corner Error:** Is concentrated near the corner point $y = y_{\max}$, $x = x_{\max}$, $\tau \gg 0$ and caused by the fact that Neuman or Dirichlet conditions are "approximately correct". If we set in the corner point

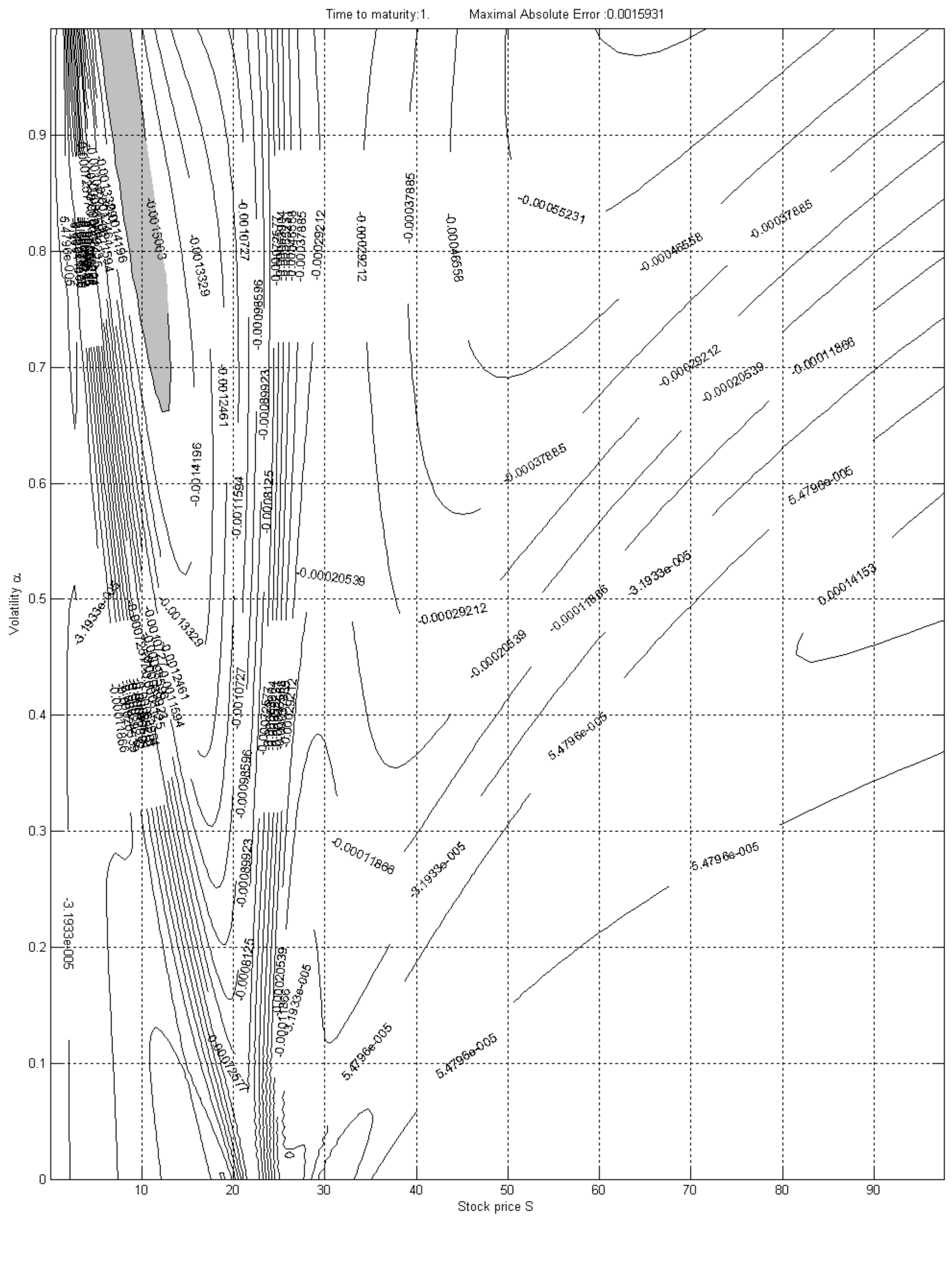


Figure 5.2: The residual between numerical and analytical solutions for model with Non Stochastic Volatility.

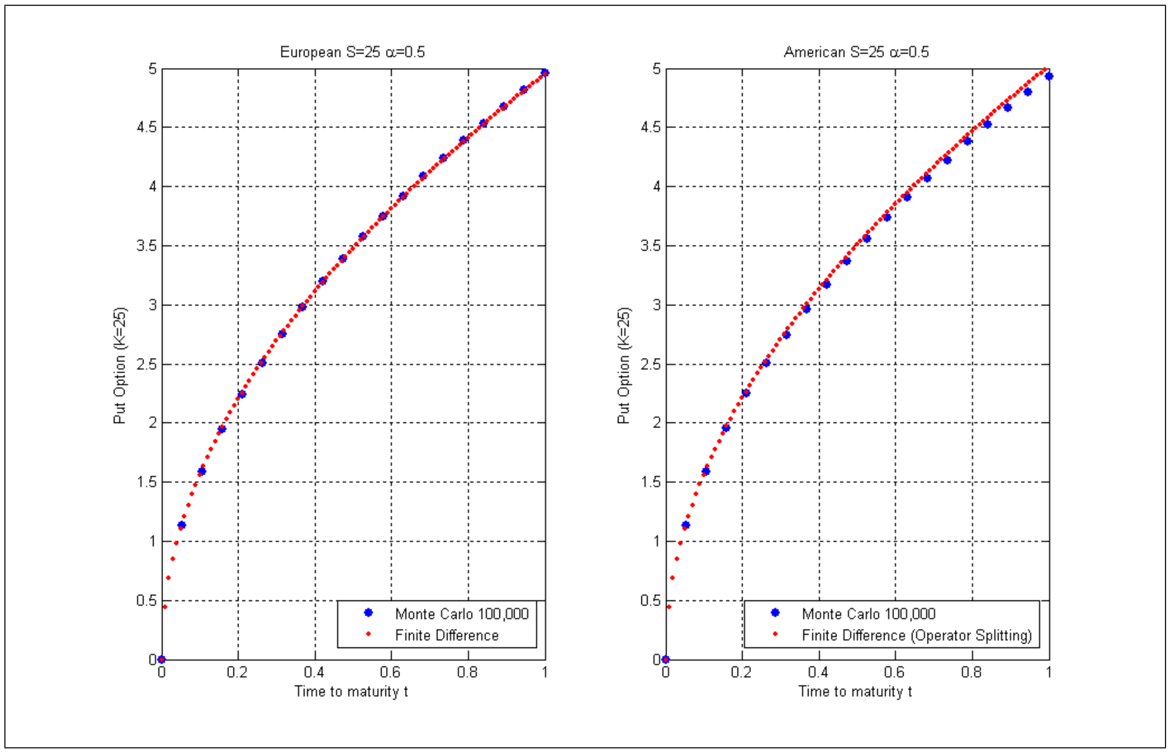


Figure 5.3: Prices of European and American options obtained by Finite Difference and Monte Carlo methods.

"smoothing" condition (3.20), error will still occur because of poor approximating possibility near the boundaries and even more in the corner. The most natural way to reduce this type of error is to take large value for $x_{\max} \approx 30K$ and in this case the error level will be acceptable.

4. **Vega Error:** This error is concentrating near $y = y_{\max}$, $x = D(K)$, $\tau \gg 0$ caused by the fact that Vega $\mathcal{V}$ is growing quickly for at the money options. If volatility mean reverting term μ_y is comparable to volatility diffusion term σ_y this type of error significantly decreases as $\mathcal{V}$ is not growing so fast any more (this could be seen from Figure 5.1 and Figure 5.2). Refining mesh for at the money option $x = D(K)$ might reduce this error, though improvement is not dramatic. General PDE transformation allows to remove the cross derivative term and this might significantly decrease this type of error. In the next section we are discussing possible alternative technique "Point Pinning" for reducing this error.

In order to give graphical interpretation of "Vega Error" in Figure 5.4 two analytical solutions for $\tau = 1$ are compared, first one is generated using "classical" SABR model ($\mu_y = 0$) and second one is generated assuming that volatility is time dependent but non stochastic NSV ($\sigma_y = 0$). One can easily notice that the first surface is much stepper for at the money option with larger initial volatility and that causes inexactness in finite difference approximation.

5.6 Point pinning

Basic idea of this technique is to choose a few points on the grid (inside domain or on the boundary) and set in these points Dirichlet conditions:

$$V_{s_i,s_j,s_k} = D(x_{s_i}, y_{s_j}, \tau_{s_k})$$

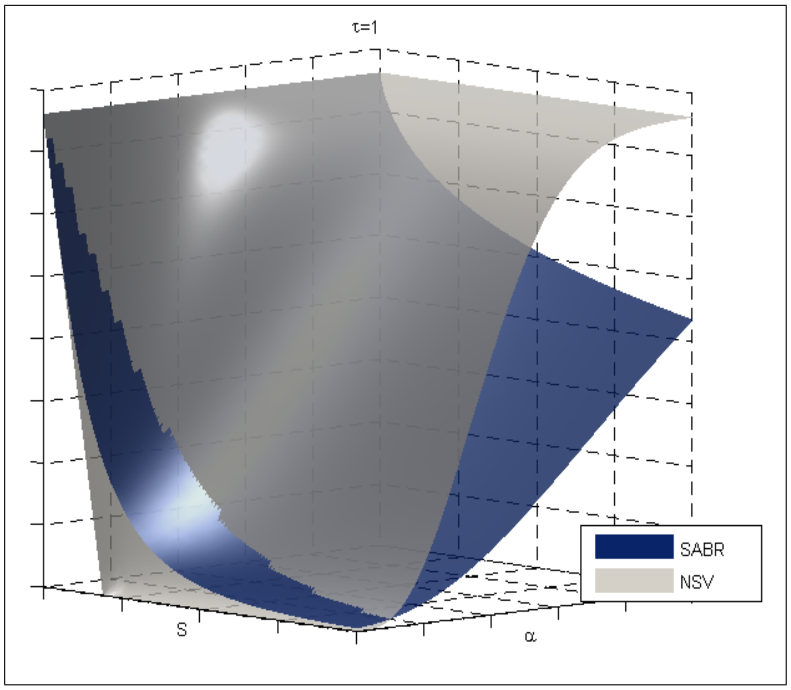


Figure 5.4: Analytical solutions of "classical" SABR and NSV models.

where s_i, s_j, s_k are chosen indices for the points in discrete domain, $D(x_{s_i}, y_{s_j}, \tau_{s_k})$ is values set on this point. In order to find appropriate values for $D(x_{s_i}, y_{s_j}, \tau_{s_k})$ one should use supporting method,(Monte-Carlo, Binomial Tree or use some analytical approximation in the point $x_{s_i}, y_{s_j}, \tau_{s_k}$). Using the supporting numerical method in order to improve finite difference method might seem controversial, but one should not forget that finite difference method returns "set" of solutions (as a matter of fact each grid point is price of option) while Monte-Carlo or Tree method return solution for one isolated point in grid. Of course, time spent on supporting numerical methods should be taken into account.

Interpretation of the point pinning approach is if one assumes that finite difference operator (3.5) acts like most natural interpolator in between of pinned points. Note that implementing this method does not really affects the time spent on creating the matrix as only few rows (equations) should be replaced.

If we apply this method to reduce "Vega Error", then error is decreased not only in pinned point but also in nearby domain. In Figure 5.5 residual function (error) presented for the "classical" SABR model with one pinned point: $S = 15$ and $\alpha = 0.95$. Comparing this with Figure 5.1 one can notice that domain (filled with gray color) with maximal absolute error has shrunk.

In case of the American option this technique could be further generalized. In particular, while using the supporting numerical method, one is receiving not only the value of option but also the approximation for the shape of the open boundary and this fact could be exploited while constructing the finite difference method.

5.7 Accuracy versus Speed

To analyze the accuracy of algorithm and the time spent on calculation, the following experiment is performed. We are choosing three different initial values for maturities, initial stock prices and initial volatilities for put option with strike $K = 25$. After this we are plotting Figure 5.6 difference (error) between the obtained prices, in chosen points, with corresponding theoretical values for four different grid resolutions ($50 \times 50 \times 50$, $100 \times 100 \times 100$, $150 \times 150 \times 150$, $200 \times 200 \times 200$) versus computational time spent for each resolution.

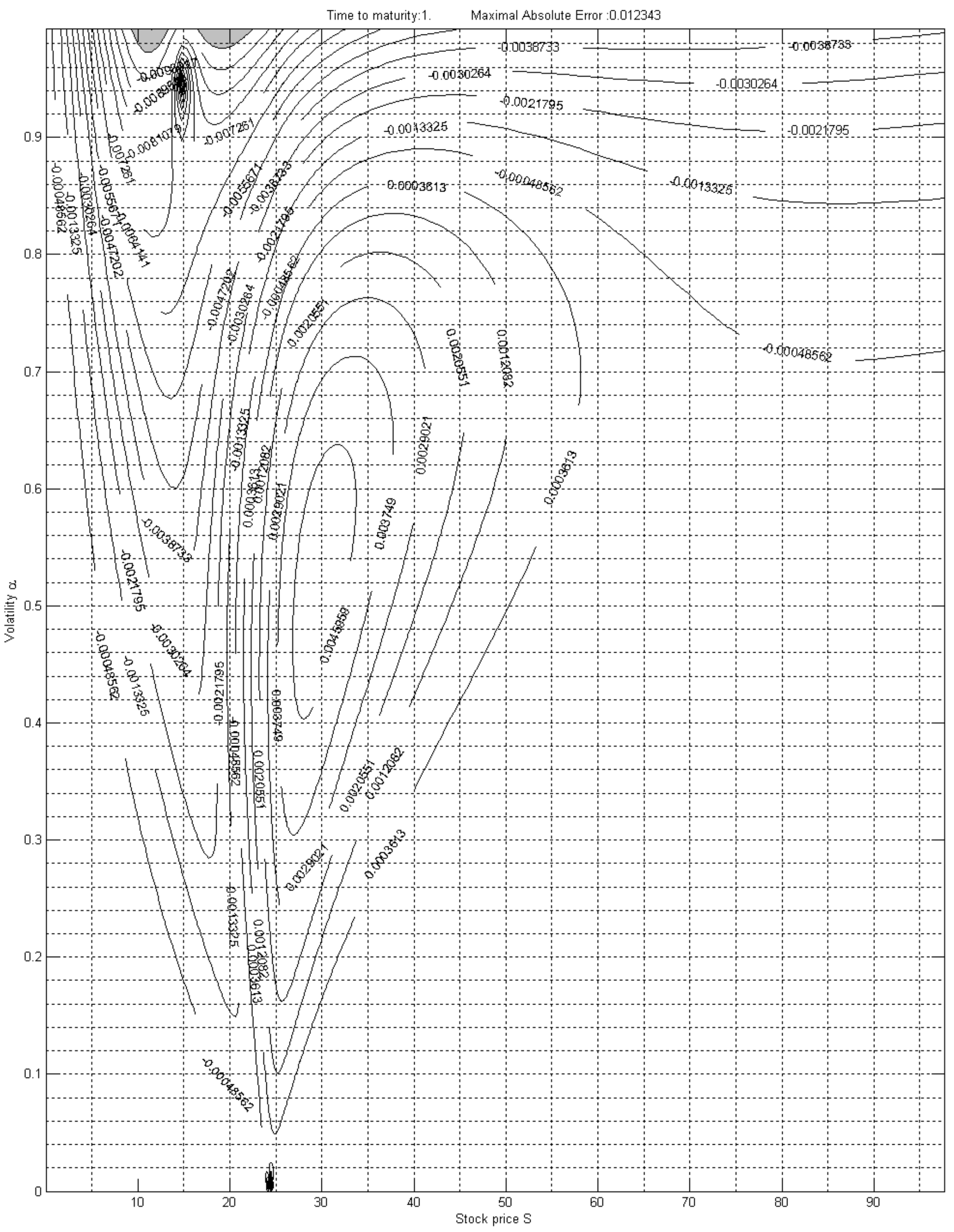


Figure 5.5: Point with $S = 15$ and $\alpha = 0.95$ is pinned.

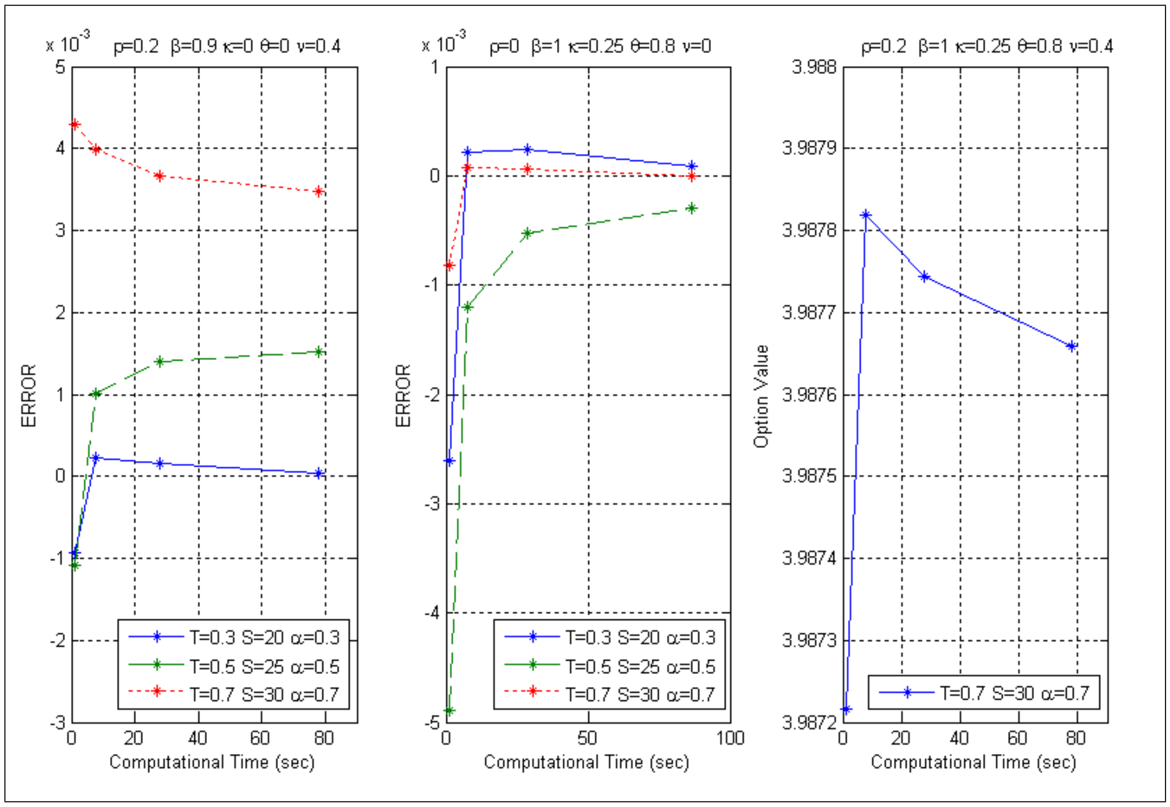


Figure 5.6: Accuracy, Speed, Convergence

Comparison with the theoretical value is performed for SABR model (first subplot) and the model with non stochastic volatilities (second subplot). We see that the convergence toward zero is obvious, though the speed of convergence differs for different point. Also one can notice that there is no significant improvement in the convergence after resolution $150 \times 150 \times 150$ (third point from left) and convergence to zero is much faster for the point with lower initial volatility.

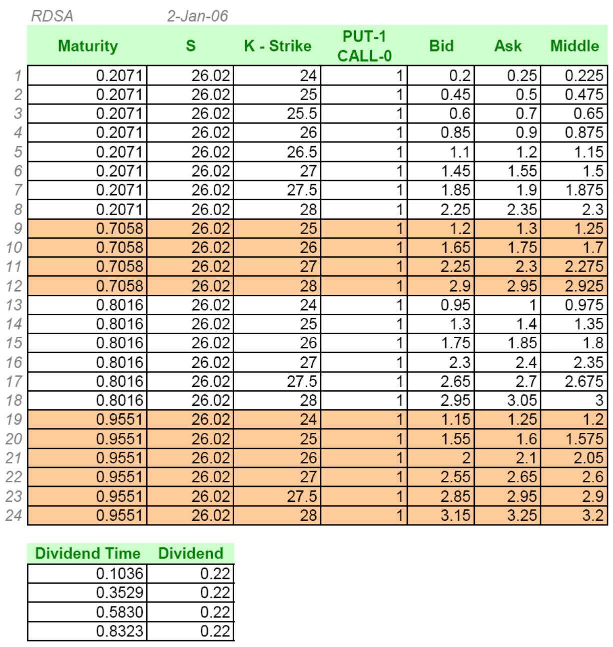
On the last subplot one can see the convergence of the absolute value of option price versus computational time (grid resolution) for general choice of parameters. It should be mentioned that convergence speed is mostly affected by refining resolution w.r.t. stock price and time rather then volatility.

Computation is performed on the following machine: *Intell CPU 1.8 GHz, RAM 1.00GB.*

5.8 Calibration

For calibrating parameters of the SABR model we are using data for the American options written on Royal Dutch Shell (*RDSA*). Data is collected for fixed time moment during trading day *2 January 2006* and presented in the Table 5.7. All maturities are presented as a fraction of year and prices are quoted in Euros. Note that underlying stock RDSA is paying dividends. For calibrating we are using middle (average) price from bid-ask spread. During fitting process we calibrate not only parameters of the SABR model but also volatility of stock, as it is not directly observable on the market. Since all option prices are taken for one fixed moment (snapshot) of a day we will have only one additional volatility parameter α to calibrate. If data would be collected for two different time moments, then two additional volatility parameters α_1 and α_2 should be taken into account and calibrated.

First set of calibrations are performed for following parameters: α - volatility of stock and "extended"



RDSA 2-Jan-06

	Maturity	S	K - Strike	PUT-1 CALL-0	Bid	Ask	Middle
1	0.2071	26.02	24	1	0.2	0.25	0.225
2	0.2071	26.02	25	1	0.45	0.5	0.475
3	0.2071	26.02	25.5	1	0.6	0.7	0.65
4	0.2071	26.02	26	1	0.85	0.9	0.875
5	0.2071	26.02	26.5	1	1.1	1.2	1.15
6	0.2071	26.02	27	1	1.45	1.55	1.5
7	0.2071	26.02	27.5	1	1.85	1.9	1.875
8	0.2071	26.02	28	1	2.25	2.35	2.3
9	0.7058	26.02	25	1	1.2	1.3	1.25
10	0.7058	26.02	26	1	1.65	1.75	1.7
11	0.7058	26.02	27	1	2.25	2.3	2.275
12	0.7058	26.02	28	1	2.9	2.95	2.925
13	0.8016	26.02	24	1	0.95	1	0.975
14	0.8016	26.02	25	1	1.3	1.4	1.35
15	0.8016	26.02	26	1	1.75	1.85	1.8
16	0.8016	26.02	27	1	2.3	2.4	2.35
17	0.8016	26.02	27.5	1	2.65	2.7	2.675
18	0.8016	26.02	28	1	2.95	3.05	3
19	0.9551	26.02	24	1	1.15	1.25	1.2
20	0.9551	26.02	25	1	1.55	1.6	1.575
21	0.9551	26.02	26	1	2	2.1	2.05
22	0.9551	26.02	27	1	2.55	2.65	2.6
23	0.9551	26.02	27.5	1	2.85	2.95	2.9
24	0.9551	26.02	28	1	3.15	3.25	3.2

Dividend Time	Dividend
0.1036	0.22
0.3529	0.22
0.5830	0.22
0.8323	0.22

Figure 5.7: Options prices and dividends.

SABR coefficients: ρ, κ, θ, ν (we assume that all parameters are constant). Risk free rate r and β are not calibrated, but we are taking different values of β for each calibration. Results for the four calibrations with different β are given in Table 5.8. In addition to the first guess of parameters and their final (calibrated) values[1] following data are presented in table: total time spent for calibration (in hours), number of required iterations to reach target tolerance, count of function calls for calculating residual functional, residual for each data point (option price), confirmation if prices calculated with calibrated parameters are in bid-ask spread and norm of residual functional $\|F(\cdot)\|$. At the end of the table parameters for the MATLAB function `lsqnonlin(·)`are given.

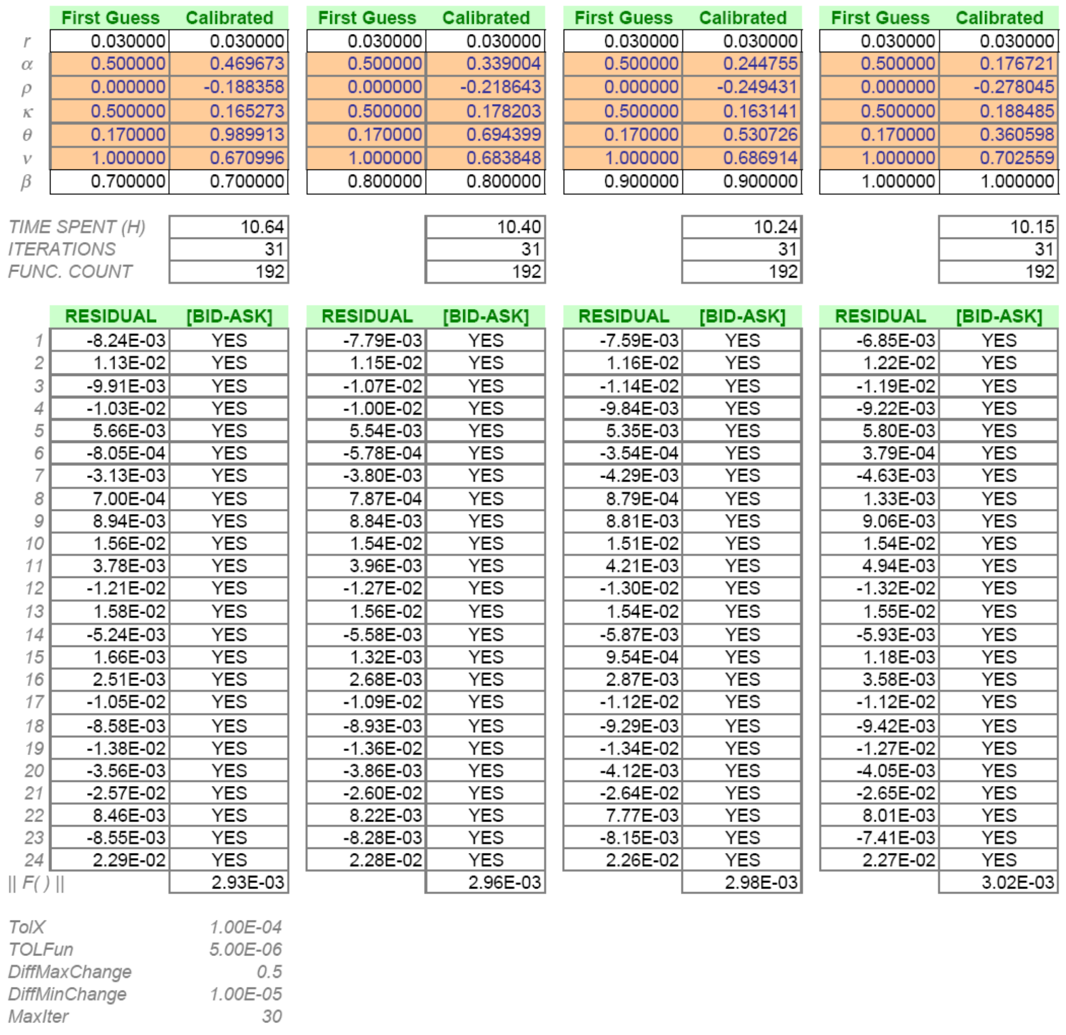
We can see that different choices of β significantly changes mean reverting limit θ and initial stock volatility α, while ρ, κ and ν are changed insignificantly. Change of α is very much online with the results obtained by [20] for "classical" SABR model. In "extended" case θ comes into play and adjusts to different values of β. It is interesting to note that calibrated values of α (for different choices of β) are almost equal to the corresponding values in the "classical" model (see Table 5.10 for calibrated values of α for "classical" SABR). Thus it is obvious that the mean reverting parameters θ and κ are playing main role in decreasing value of residual functional, and produce option prices that are within bid-ask spreads.

Second set of calibrations are performed for following parameters: α - volatility of stock and extended SABR coefficients: ρ, κ, θ, ν and β. In contrast to the previous set of calibrations we are additionally calibrating β and keeping fixed only risk free rate r. From Table 5.9 one can see that it is always better to keep β fixed, since functional $\|F(\cdot)\|$seems to have equal minimal values for different β (so called "ditch" of minimums with respect to β).

Third set of calibrations is performed for "classical" SABR model, i.e. following coefficients are calibrated: α, ρ, ν while κ, θ, β and risk free rate r are kept fixed[2] (for each calibration different fixed values of β are taken). From Table 5.10 one can notice that some of the produced prices are not in bid-ask spread and value of residual functional is greater then corresponding one in Table 5.8. This might not be surprising since the

[1]Values of r and β are kept fixed and written in white cells, while first guess and final calibrated values of parameters α, ρ, κ, θ and ν are written in colored cells.

[2]For "classical" SABR model $\kappa = \theta = 0$.



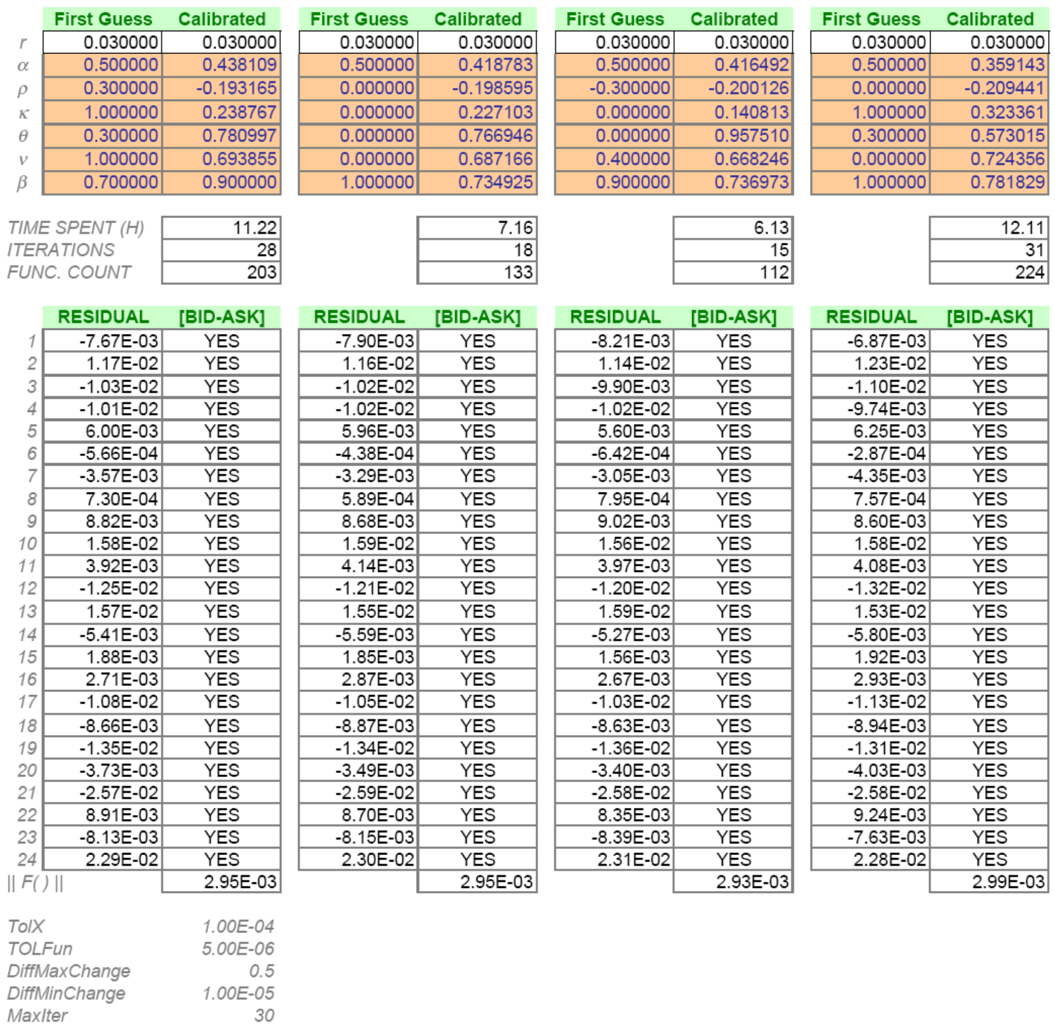
	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated
r	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000
α	0.500000	0.469673	0.500000	0.339004	0.500000	0.244755	0.500000	0.176721
ρ	0.000000	-0.188358	0.000000	-0.218643	0.000000	-0.249431	0.000000	-0.278045
κ	0.500000	0.165273	0.500000	0.178203	0.500000	0.163141	0.500000	0.188485
θ	0.170000	0.989913	0.170000	0.694399	0.170000	0.530726	0.170000	0.360598
ν	1.000000	0.670996	1.000000	0.683848	1.000000	0.686914	1.000000	0.702559
β	0.700000	0.700000	0.800000	0.800000	0.900000	0.900000	1.000000	1.000000

TIME SPENT (H)	10.64	10.40	10.24	10.15
ITERATIONS	31	31	31	31
FUNC. COUNT	192	192	192	192

	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]
1	-8.24E-03	YES	-7.79E-03	YES	-7.59E-03	YES	-6.85E-03	YES
2	1.13E-02	YES	1.15E-02	YES	1.16E-02	YES	1.22E-02	YES
3	-9.91E-03	YES	-1.07E-02	YES	-1.14E-02	YES	-1.19E-02	YES
4	-1.03E-02	YES	-1.00E-02	YES	-9.84E-03	YES	-9.22E-03	YES
5	5.66E-03	YES	5.54E-03	YES	5.35E-03	YES	5.80E-03	YES
6	-8.05E-04	YES	-5.78E-04	YES	-3.54E-04	YES	3.79E-04	YES
7	-3.13E-03	YES	-3.80E-03	YES	-4.29E-03	YES	-4.63E-03	YES
8	7.00E-04	YES	7.87E-04	YES	8.79E-04	YES	1.33E-03	YES
9	8.94E-03	YES	8.84E-03	YES	8.81E-03	YES	9.06E-03	YES
10	1.56E-02	YES	1.54E-02	YES	1.51E-02	YES	1.54E-02	YES
11	3.78E-03	YES	3.96E-03	YES	4.21E-03	YES	4.94E-03	YES
12	-1.21E-02	YES	-1.27E-02	YES	-1.30E-02	YES	-1.32E-02	YES
13	1.58E-02	YES	1.56E-02	YES	1.54E-02	YES	1.55E-02	YES
14	-5.24E-03	YES	-5.58E-03	YES	-5.87E-03	YES	-5.93E-03	YES
15	1.66E-03	YES	1.32E-03	YES	9.54E-04	YES	1.18E-03	YES
16	2.51E-03	YES	2.68E-03	YES	2.87E-03	YES	3.58E-03	YES
17	-1.05E-02	YES	-1.09E-02	YES	-1.12E-02	YES	-1.12E-02	YES
18	-8.58E-03	YES	-8.93E-03	YES	-9.29E-03	YES	-9.42E-03	YES
19	-1.38E-02	YES	-1.36E-02	YES	-1.34E-02	YES	-1.27E-02	YES
20	-3.56E-03	YES	-3.86E-03	YES	-4.12E-03	YES	-4.05E-03	YES
21	-2.57E-02	YES	-2.60E-02	YES	-2.64E-02	YES	-2.65E-02	YES
22	8.46E-03	YES	8.22E-03	YES	7.77E-03	YES	8.01E-03	YES
23	-8.55E-03	YES	-8.28E-03	YES	-8.15E-03	YES	-7.41E-03	YES
24	2.29E-02	YES	2.28E-02	YES	2.26E-02	YES	2.27E-02	YES
|| F( ) ||		2.93E-03		2.96E-03		2.98E-03		3.02E-03

TolX	1.00E-04
TOLFun	5.00E-06
DiffMaxChange	0.5
DiffMinChange	1.00E-05
MaxIter	30

Figure 5.8: Calibration results for "extended" SABR model (β fixed).



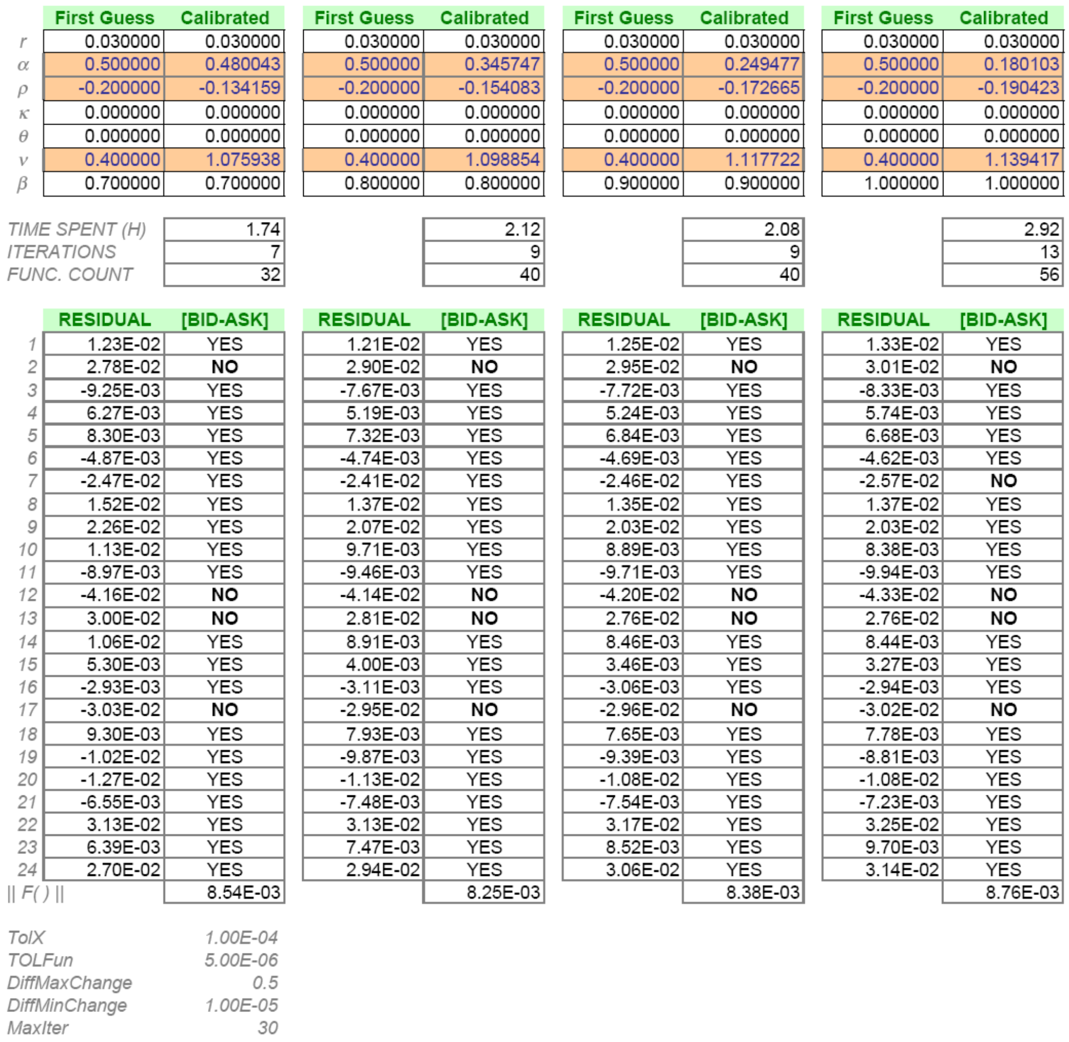
	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated
r	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000
α	0.500000	0.438109	0.500000	0.418783	0.500000	0.416492	0.500000	0.359143
ρ	0.300000	-0.193165	0.000000	-0.198595	-0.300000	-0.200126	0.000000	-0.209441
κ	1.000000	0.238767	0.000000	0.227103	0.000000	0.140813	1.000000	0.323361
θ	0.300000	0.780997	0.000000	0.766946	0.000000	0.957510	0.300000	0.573015
ν	1.000000	0.693855	0.000000	0.687166	0.400000	0.668246	0.000000	0.724356
β	0.700000	0.900000	1.000000	0.734925	0.900000	0.736973	1.000000	0.781829

TIME SPENT (H)	11.22	7.16	6.13	12.11
ITERATIONS	28	18	15	31
FUNC. COUNT	203	133	112	224

	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]
1	-7.67E-03	YES	-7.90E-03	YES	-8.21E-03	YES	-6.87E-03	YES
2	1.17E-02	YES	1.16E-02	YES	1.14E-02	YES	1.23E-02	YES
3	-1.03E-02	YES	-1.02E-02	YES	-9.90E-03	YES	-1.10E-02	YES
4	-1.01E-02	YES	-1.02E-02	YES	-1.02E-02	YES	-9.74E-03	YES
5	6.00E-03	YES	5.96E-03	YES	5.60E-03	YES	6.25E-03	YES
6	-5.66E-04	YES	-4.38E-04	YES	-6.42E-04	YES	-2.87E-04	YES
7	-3.57E-03	YES	-3.29E-03	YES	-3.05E-03	YES	-4.35E-03	YES
8	7.30E-04	YES	5.89E-04	YES	7.95E-04	YES	7.57E-04	YES
9	8.82E-03	YES	8.68E-03	YES	9.02E-03	YES	8.60E-03	YES
10	1.58E-02	YES	1.59E-02	YES	1.56E-02	YES	1.58E-02	YES
11	3.92E-03	YES	4.14E-03	YES	3.97E-03	YES	4.08E-03	YES
12	-1.25E-02	YES	-1.21E-02	YES	-1.20E-02	YES	-1.32E-02	YES
13	1.57E-02	YES	1.55E-02	YES	1.59E-02	YES	1.53E-02	YES
14	-5.41E-03	YES	-5.59E-03	YES	-5.27E-03	YES	-5.80E-03	YES
15	1.88E-03	YES	1.85E-03	YES	1.56E-03	YES	1.92E-03	YES
16	2.71E-03	YES	2.87E-03	YES	2.67E-03	YES	2.93E-03	YES
17	-1.08E-02	YES	-1.05E-02	YES	-1.03E-02	YES	-1.13E-02	YES
18	-8.66E-03	YES	-8.87E-03	YES	-8.63E-03	YES	-8.94E-03	YES
19	-1.35E-02	YES	-1.34E-02	YES	-1.36E-02	YES	-1.31E-02	YES
20	-3.73E-03	YES	-3.49E-03	YES	-3.40E-03	YES	-4.03E-03	YES
21	-2.57E-02	YES	-2.59E-02	YES	-2.58E-02	YES	-2.58E-02	YES
22	8.91E-03	YES	8.70E-03	YES	8.35E-03	YES	9.24E-03	YES
23	-8.13E-03	YES	-8.15E-03	YES	-8.39E-03	YES	-7.63E-03	YES
24	2.29E-02	YES	2.30E-02	YES	2.31E-02	YES	2.28E-02	YES
|| F( ) ||		2.95E-03		2.95E-03		2.93E-03		2.99E-03

TolX	1.00E-04
TOLFun	5.00E-06
DiffMaxChange	0.5
DiffMinChange	1.00E-05
MaxIter	30

Figure 5.9: Calibration results for "extended" SABR model.



	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated
r	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000	0.030000
α	0.500000	0.480043	0.500000	0.345747	0.500000	0.249477	0.500000	0.180103
ρ	-0.200000	-0.134159	-0.200000	-0.154083	-0.200000	-0.172665	-0.200000	-0.190423
κ	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
θ	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ν	0.400000	1.075938	0.400000	1.098854	0.400000	1.117722	0.400000	1.139417
β	0.700000	0.700000	0.800000	0.800000	0.900000	0.900000	1.000000	1.000000

TIME SPENT (H)	1.74	2.12	2.08	2.92
ITERATIONS	7	9	9	13
FUNC. COUNT	32	40	40	56

	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]
1	1.23E-02	YES	1.21E-02	YES	1.25E-02	YES	1.33E-02	YES
2	2.78E-02	**NO**	2.90E-02	**NO**	2.95E-02	**NO**	3.01E-02	**NO**
3	-9.25E-03	YES	-7.67E-03	YES	-7.72E-03	YES	-8.33E-03	YES
4	6.27E-03	YES	5.19E-03	YES	5.24E-03	YES	5.74E-03	YES
5	8.30E-03	YES	7.32E-03	YES	6.84E-03	YES	6.68E-03	YES
6	-4.87E-03	YES	-4.74E-03	YES	-4.69E-03	YES	-4.62E-03	YES
7	-2.47E-02	YES	-2.41E-02	YES	-2.46E-02	YES	-2.57E-02	**NO**
8	1.52E-02	YES	1.37E-02	YES	1.35E-02	YES	1.37E-02	YES
9	2.26E-02	YES	2.07E-02	YES	2.03E-02	YES	2.03E-02	YES
10	1.13E-02	YES	9.71E-03	YES	8.89E-03	YES	8.38E-03	YES
11	-8.97E-03	YES	-9.46E-03	YES	-9.71E-03	YES	-9.94E-03	YES
12	-4.16E-02	**NO**	-4.14E-02	**NO**	-4.20E-02	**NO**	-4.33E-02	**NO**
13	3.00E-02	**NO**	2.81E-02	**NO**	2.76E-02	**NO**	2.76E-02	**NO**
14	1.06E-02	YES	8.91E-03	YES	8.46E-03	YES	8.44E-03	YES
15	5.30E-03	YES	4.00E-03	YES	3.46E-03	YES	3.27E-03	YES
16	-2.93E-03	YES	-3.11E-03	YES	-3.06E-03	YES	-2.94E-03	YES
17	-3.03E-02	**NO**	-2.95E-02	**NO**	-2.96E-02	**NO**	-3.02E-02	**NO**
18	9.30E-03	YES	7.93E-03	YES	7.65E-03	YES	7.78E-03	YES
19	-1.02E-02	YES	-9.87E-03	YES	-9.39E-03	YES	-8.81E-03	YES
20	-1.27E-02	YES	-1.13E-02	YES	-1.08E-02	YES	-1.08E-02	YES
21	-6.55E-03	YES	-7.48E-03	YES	-7.54E-03	YES	-7.23E-03	YES
22	3.13E-02	YES	3.13E-02	YES	3.17E-02	YES	3.25E-02	YES
23	6.39E-03	YES	7.47E-03	YES	8.52E-03	YES	9.70E-03	YES
24	2.70E-02	YES	2.94E-02	YES	3.06E-02	YES	3.14E-02	YES
|| F( ) ||		8.54E-03		8.25E-03		8.38E-03		8.76E-03

TolX	1.00E-04
TOLFun	5.00E-06
DiffMaxChange	0.5
DiffMinChange	1.00E-05
MaxIter	30

Figure 5.10: Calibration results for "classical" SABR model (β fixed).

more freedom functional has the better its approximating features should be, but this is also an argument one more time confirming that "extended" SABR model more precisely describes stock price behavior.

Fourth set of calibrations are performed for "extended" SABR model with time dependent coefficients (for each calibration different fixed values of β are taken). During this part we assume that: $\theta = \theta(\tau) = \theta_1\tau^3 + \theta_2\tau^2 + \theta_3\tau + \theta_4$is function of time and it is given as cubic polynomial. Risk free rate is also time dependent and given in polynomial form: $r = r(\tau) = r_1\tau^3 + r_2\tau^2 + r_3\tau + r_4$. Coefficients:

$$\begin{aligned} r_1 &= -2.05619348 \cdot 10^{-4}, \\ r_2 &= -1.33536972 \cdot 10^{-3}, \\ r_3 &= -3.21917123 \cdot 10^{-3}, \\ r_4 &= 3.10604120 \cdot 10^{-2}, \end{aligned}$$

are found from interest rate term structure.

Calibrations are performed for the following parameters: α, ρ, ν, κ and θ_1, θ_2, θ_3, θ_4 (see Table 5.12). Initial guesses for values of parameters α, ρ, ν, κ are taken equal to corresponding calibrated values received

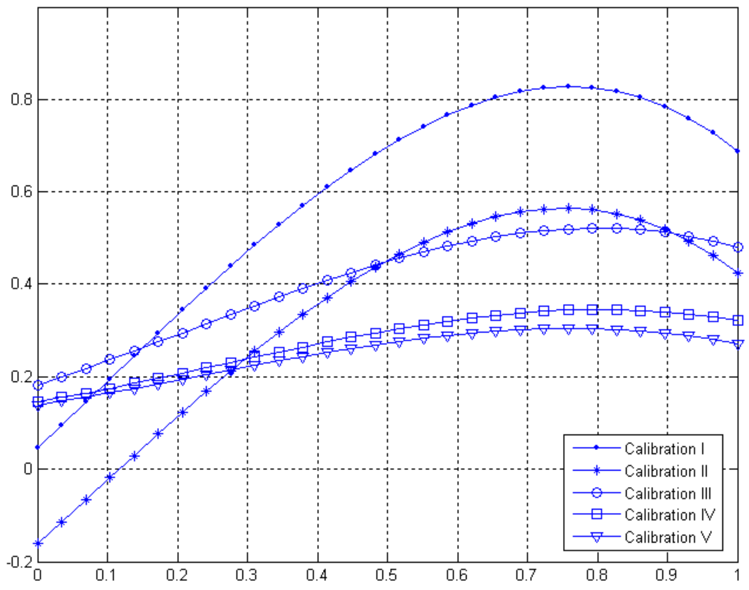
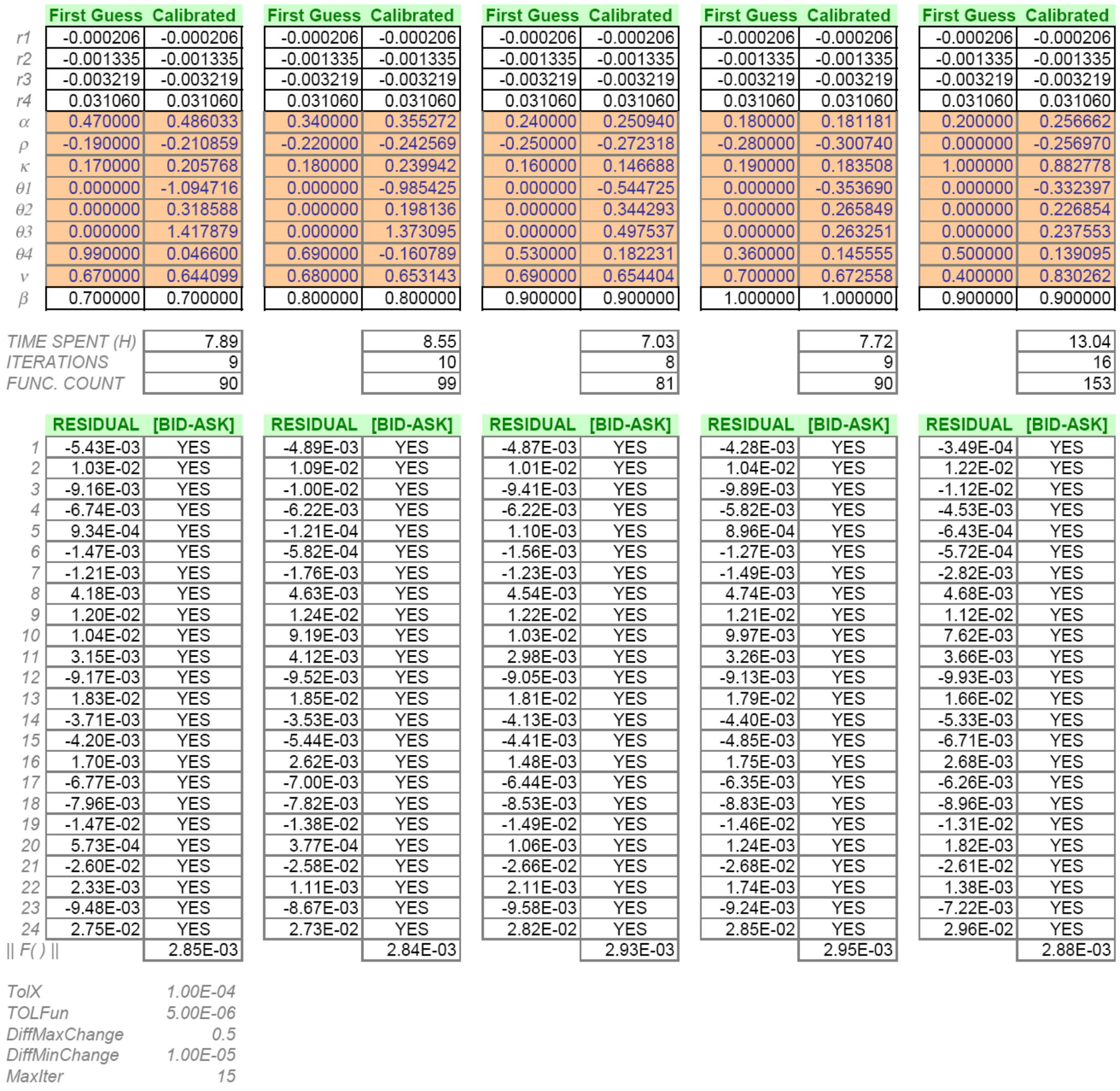


Figure 5.11: Time dependency of mean reverting limit $\theta(\tau)$.

during the first part of calibrations (see Table 5.8), this is done to avoid redundant computational time. Time dependency of mean reverting limit $\theta(\tau)$ could be seen in Figure 5.11.

Value of residual functional has not decreased noticeably (compared to "extended" model with constant coefficients Table 5.8), additionally one of the calibrated mean reverting limiting function $\theta(\tau)$ takes negative values. Negative values of mean reverting limit might partially be explained by poor approximating possibilities of cubic polynomial. But, unnoticeable decrease of residual functional indicates that time dependency of mean reverting limit could not be found out only from 24 observations (Table 5.7). It is known that in while fitting parameters of a system number of observation should be at least 5 times greater then the number of calibrated parameters (in our case 8).



	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated	First Guess	Calibrated
r1	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206	-0.000206
r2	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335	-0.001335
r3	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219	-0.003219
r4	0.031060	0.031060	0.031060	0.031060	0.031060	0.031060	0.031060	0.031060	0.031060	0.031060
α	0.470000	0.486033	0.340000	0.355272	0.240000	0.250940	0.180000	0.181181	0.200000	0.256662
ρ	-0.190000	-0.210859	-0.220000	-0.242569	-0.250000	-0.272318	-0.280000	-0.300740	0.000000	-0.256970
κ	0.170000	0.205768	0.180000	0.239942	0.160000	0.146688	0.190000	0.183508	1.000000	0.882778
$\theta 1$	0.000000	-1.094716	0.000000	-0.985425	0.000000	-0.544725	0.000000	-0.353690	0.000000	-0.332397
$\theta 2$	0.000000	0.318588	0.000000	0.198136	0.000000	0.344293	0.000000	0.265849	0.000000	0.226854
$\theta 3$	0.000000	1.417879	0.000000	1.373095	0.000000	0.497537	0.000000	0.263251	0.000000	0.237553
$\theta 4$	0.990000	0.046600	0.690000	-0.160789	0.530000	0.182231	0.360000	0.145555	0.500000	0.139095
ν	0.670000	0.644099	0.680000	0.653143	0.690000	0.654404	0.700000	0.672558	0.400000	0.830262
β	0.700000	0.700000	0.800000	0.800000	0.900000	0.900000	1.000000	1.000000	0.900000	0.900000

TIME SPENT (H)	7.89	8.55	7.03	7.72	13.04
ITERATIONS	9	10	8	9	16
FUNC. COUNT	90	99	81	90	153

	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]	RESIDUAL	[BID-ASK]
1	-5.43E-03	YES	-4.89E-03	YES	-4.87E-03	YES	-4.28E-03	YES	-3.49E-04	YES
2	1.03E-02	YES	1.09E-02	YES	1.01E-02	YES	1.04E-02	YES	1.22E-02	YES
3	-9.16E-03	YES	-1.00E-02	YES	-9.41E-03	YES	-9.89E-03	YES	-1.12E-02	YES
4	-6.74E-03	YES	-6.22E-03	YES	-6.22E-03	YES	-5.82E-03	YES	-4.53E-03	YES
5	9.34E-04	YES	-1.21E-04	YES	1.10E-03	YES	8.96E-04	YES	-6.43E-04	YES
6	-1.47E-03	YES	-5.82E-04	YES	-1.56E-03	YES	-1.27E-03	YES	-5.72E-04	YES
7	-1.21E-03	YES	-1.76E-03	YES	-1.23E-03	YES	-1.49E-03	YES	-2.82E-03	YES
8	4.18E-03	YES	4.63E-03	YES	4.54E-03	YES	4.74E-03	YES	4.68E-03	YES
9	1.20E-02	YES	1.24E-02	YES	1.22E-02	YES	1.21E-02	YES	1.12E-02	YES
10	1.04E-02	YES	9.19E-03	YES	1.03E-02	YES	9.97E-03	YES	7.62E-03	YES
11	3.15E-03	YES	4.12E-03	YES	2.98E-03	YES	3.26E-03	YES	3.66E-03	YES
12	-9.17E-03	YES	-9.52E-03	YES	-9.05E-03	YES	-9.13E-03	YES	-9.93E-03	YES
13	1.83E-02	YES	1.85E-02	YES	1.81E-02	YES	1.79E-02	YES	1.66E-02	YES
14	-3.71E-03	YES	-3.53E-03	YES	-4.13E-03	YES	-4.40E-03	YES	-5.33E-03	YES
15	-4.20E-03	YES	-5.44E-03	YES	-4.41E-03	YES	-4.85E-03	YES	-6.71E-03	YES
16	1.70E-03	YES	2.62E-03	YES	1.48E-03	YES	1.75E-03	YES	2.68E-03	YES
17	-6.77E-03	YES	-7.00E-03	YES	-6.44E-03	YES	-6.35E-03	YES	-6.26E-03	YES
18	-7.96E-03	YES	-7.82E-03	YES	-8.53E-03	YES	-8.83E-03	YES	-8.96E-03	YES
19	-1.47E-02	YES	-1.38E-02	YES	-1.49E-02	YES	-1.46E-02	YES	-1.31E-02	YES
20	5.73E-04	YES	3.77E-04	YES	1.06E-03	YES	1.24E-03	YES	1.82E-03	YES
21	-2.60E-02	YES	-2.58E-02	YES	-2.66E-02	YES	-2.68E-02	YES	-2.61E-02	YES
22	2.33E-03	YES	1.11E-03	YES	2.11E-03	YES	1.74E-03	YES	1.38E-03	YES
23	-9.48E-03	YES	-8.67E-03	YES	-9.58E-03	YES	-9.24E-03	YES	-7.22E-03	YES
24	2.75E-02	YES	2.73E-02	YES	2.82E-02	YES	2.85E-02	YES	2.96E-02	YES
|| F( ) ||		2.85E-03		2.84E-03		2.93E-03		2.95E-03		2.88E-03

TolX	*1.00E-04*
TOLFun	*5.00E-06*
DiffMaxChange	*0.5*
DiffMinChange	*1.00E-05*
MaxIter	*15*

Figure 5.12: Calibration results for time dependent "extended" SABR model (β fixed).

Chapter 6

Conclusion and Discussion

The main goal of our research was to:

- Compare the "extended" SABR model with the "classical" one and test how well the "extended" model can price European and/or American options.
- Design and compare numerical methods for reproducing option prices.
- Fit parameters of both models to real market data and compare them.

The comparison of models showed that the "extended" SABR model provides higher degrees of freedom, the result being predictable since any additional parameter inserted into the model will generalize it. But the main question was how much new parameters (volatility mean reverting terms in our case) increases degrees of freedom of the model and how well this extension in its turn models reality. Comparing solutions and problem statement for "extended" and "classical" models we can make following observations. First, is the in case of the "extended" SABR model, the boundary condition (in PDE formulation) for zero volatility $\alpha = 0$ changes from the Dirichlet type condition to the so called "smoothing" condition. Secondly, the mean reverting term adds partial derivative w.r.t. α to partial differential equation. Thirdly, under "classical" model variance of volatility tends to infinity, while in the "extended" model it pushed to a certain finite limit. Summing up all these facts one can see that extension of "classical" model provides non-trivial generalization with a sufficiently high degrees of freedom and seems to model stock price behavior better.

While fitting parameters to real market data it was found that:

1. β does not affect accuracy of fit neither in the "classical" nor in "extended" models. This fact is very much online with the results obtained by [20]. In the "classical" model initial stock volatility α adjusts to different choices of β and as a result is mostly influenced by the particular choice of β. In the "extended" model different choices of β affects mostly α and θ, while other parameters of the "extended" model are not noticeably changed. When β was calibrated together with the other parameters of the "extended" model we obtained the so called "ditch" of minimums w.r.t. β, and this fact is also online with the results obtained by [20].
2. Prices of options obtained during calibration of the "extended" model were always in bid-ask spread in contrast to prices obtained during calibration of the "classical" model. Additionally, the residual functional was 4 times smaller for the "extended" model. These facts once more underlines that the "extended" model can better approximate options prices.
3. Assuming that volatility mean reverting limit $\theta(\tau)$ is time dependent and calibrating additionally w.r.t. this function it was found that time dependent mean reverting limit is not constant and the shape of the curve is affected by particular choice of β. While increasing the number of calibrating parameters of the model (mean reverting limit $\theta(\tau)$ assumed to be polynomial with unknown coefficients) we are

facing the fact that the number of the local minimums of functional might dramatically increase, thus we might doubt if our minimum is local or global.

4. Observing decrease of the value of residual functional with number of iterations, we can conclude that the first $4, 5$ iterations will be enough for the options prices to drop into bid-ask spread in the case of the "extended" SABR model, while for the classical model even 10 iterations was not enough . And that is another fact supporting advantage of the "extended" SABR model.

While comparing the Monte Carlo method with the Finite Difference method we take into account accuracy and speed. It was found that Finite Difference method provides better approximation for the "extended" model compared to the "classical" one. For example, the so called "Vega Error" (at the money options with high initial volatility and big time to maturity) is greater when volatility mean reverting terms θ and κ are much smaller then volatility of volatility ν. Monte Carlo method seems to work with equal effectiveness for both models, but the main drawback of the Monte Carlo method is the time spent on simulation. Especially power operations S^{β} significantly increase the calculation time when $\beta \neq 1$. Thus, the Finite Difference method seems to be an appropriate and good choice for solving this kind of problems. Additionally it should be mentioned that one "launch" of the Finite Difference method gives option prices (solutions) for each point of the finite difference grid (i.e. for different initial stock prices, volatilities and time to maturities), while one "launch" of the Monte Carlo method gives the solution only for one initial value of stock and volatility. It was shown that time required for computation is incomparable. The Finite Difference methods requires 80 seconds to achieve accuracy of 1 cent for option prices with different stock prices, volatilities and time to maturities defined by grid points. While the Monte Carlo methods takes $10 - 15$ minutes for the same accuracy for one point only (if $\beta \neq 1$ this difference is even greater). In the case of American option the Monte Carlo method additionally stores all sample paths in order to apply list square method and this requires extra waste of computational recourses. Thus, the Monte Carlo method could only be used to check and compare different results. An additional advantage of the Finite Difference method is that since option prices are given in each grid point Greeks: Δ, Γ and $\mathcal{V}$ of option could be immediately computed; also using multiple right hand sides of the system of equations (see B) volatility surface can be constructed.

Although the results are acceptable, some critical remarks should be made. This might lead to the improvement of the results and provide recommendations for further research:

1. More general variable transformation should be applied to PDE (2.10) in order to eliminate or decrease cross derivative and first derivatives w.r.t. volatility α and stock price S. Transformation could be adapted to different choices of μ_α, σ_α, σ_S and this will definitely improve the accuracy of a Finite Difference solution.

2. Recently formula for the implied volatility similar to (1.3) was developed by [11] for the "extended" SABR model with constant coefficients. It will be interesting to check the accuracy of the suggested formula by means of comparing it with some numerical results.

3. If formula suggested by [11] proves to be accurate and reliable, it will be interesting to apply the technique described in A.2 and A.3 to this formula in order to adjust it for the case when some parameters (θ for example) of the "extended" SABR model are time dependent.

Appendix A

Analytical Approximation of Solution

In this chapter we will refer to initial variables S, α, t and assume that later the transformation is performed. Let us try to find solution of PDE (2.13) in the form of:

$$V(S, \alpha, t) = \hat{V}(S, \alpha, t) + G(S, \alpha, t)$$

where $\hat{V}(S, \alpha, t)$ is price of option in analytical form, while $G(S, \alpha, t)$ is correction (or residual) term. Analytical form $\hat{V}(S, \alpha, t)$ is chosen (guessed) in a way to be as "close" as possible to the real value $V(S, \alpha, t)$ of derivative. Possible initial guesses for $\hat{V}(S, \alpha, t)$ will be described below.

The better initial guess $\hat{V}(S, \alpha, t)$ will be, the lessen error term $G(S, \alpha, t)$ should be and easier to find it in analytical form.

A.1 First guess - Non stochastic volatility

For the first guess of function $\hat{V}(S, \alpha, t)$ we are assuming that the variance of stock price is not stochastic but time dependent and has generalized form of (5.1):

$$\bar{\sigma}^2(t) = P(\rho(t), E[\alpha_t], E[\alpha_t^2])$$

where $E[\alpha_t]$ and $E[\alpha_t^2]$ are the first two moments for the extended model[1], while $P(\cdot)$ is polynomial of the three variables with initially guessed coefficients. After this substituting $\bar{\sigma}$ into Black Scholes formula we will get $\hat{V}(S, \alpha, t)$. In order to correctly guess coefficients for $P(\cdot)$ one might need to run optimization w.r.t. mentioned coefficients to reduce $\left\| V(S, \alpha, t) - \hat{V}(S, \alpha, t) \right\|$.

Further improvement of described approach is possible if one includes higher moments for stochastic volatility and carefully choose function $P(\cdot)$.

A.2 Second guess - Time adjusted modification

For the second guess of function $\hat{V}(S, \alpha, t)$ we are using price of derivative generated using some modification of Hagan's formula. To modify it we are assuming that parameters $\bar{\alpha}_0$ and $\bar{\nu}$ in (1.3)[2] are time dependent and for different t should be found from:

$$\begin{aligned} &\bar{\alpha}_0(t) = E[\alpha_t]; \\ &\bar{\alpha}_0^2(t)\left(\exp(t\bar{\nu}^2(t)) - 1\right) = VAR[\alpha_t] \end{aligned}$$

[1] In order to find first moments of volatility with time dependent coefficients one might use technique similar to one used for finding (2.2), (2.3) when coefficients are assumed to be constant.

[2] In order to avoid confusing notation we denote by *bar* initial volatility and the volatility of volatility used in Hagan's formula.

where $E[\alpha_t]$ and $VAR[\alpha_t]$ are defined by (2.2), (2.3) for "extended" SABR model. Identically:

$$\begin{aligned}\bar{\alpha}_0(t) &= \theta + (\alpha_0 - \theta) e^{-\kappa t} \\ \bar{\nu}(t) &= \sqrt{\frac{1}{t} \ln \left(\frac{1}{\bar{\alpha}_0^2(t)} VAR[\alpha_t] + 1 \right)}\end{aligned}$$

Such choice of parameters matches expectation and variance of volatility of "classical" SABR model for each t with corresponding expectation and variance of "extended" SABR models. Thus we are trying to approximate implied volatility surface for "extended" SABR model by using "classical" SABR model for each t.

On Figure A.1 evolution of residual function $G(S, \alpha, t)$ in time is presented.

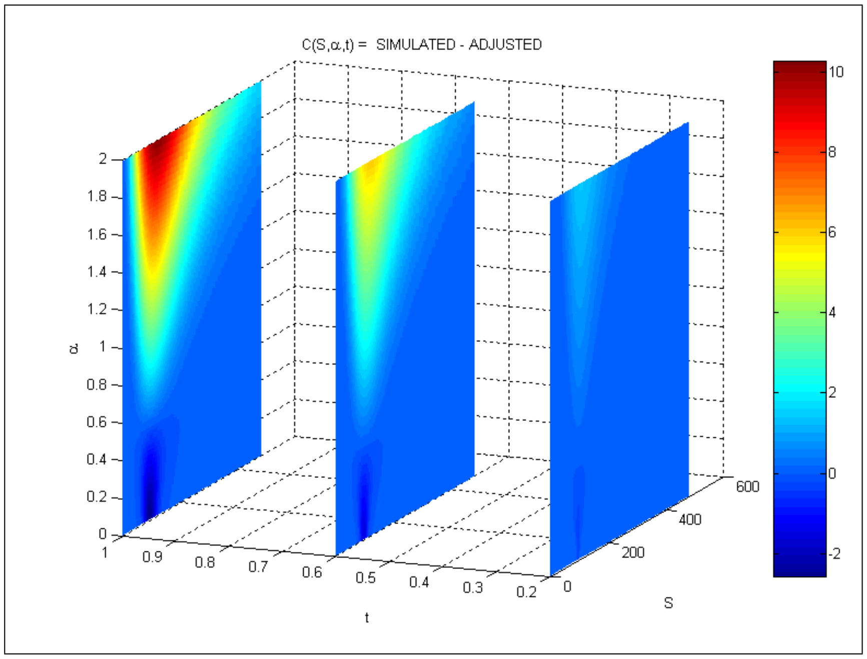


Figure A.1: Residual function $G(S, \alpha, t)$ for time adjusted modification.

A.3 Third guess - Time averaged modification

This case is identical to the previous one, we are assuming that parameters α_0 and ν are again time dependent but in this case we require from parameters to satisfy to the following equations:

$$\int_0^t \bar{\alpha}_0 ds = \int_0^t E\left[\alpha_s\right] ds$$

$$\int_0^t \bar{\alpha}_0^2 \left(e^{t\bar{\nu}^2} - 1\right) ds = \int_0^t VAR\left[\alpha_s\right] ds$$

Such choice of parameters averages w.r.t. time, expectation and variance of volatility of "classical" SABR model with corresponding expectation and variance of "extended" SABR model.

On Figure A.2 the evolution of residual function $G\left(S, \alpha, t\right)$ in time is presented. Note that time averaged modification gave better results compared to time adjusted modification. Additionally for time to maturity less then 0.2 residual is smaller then 1 cent.

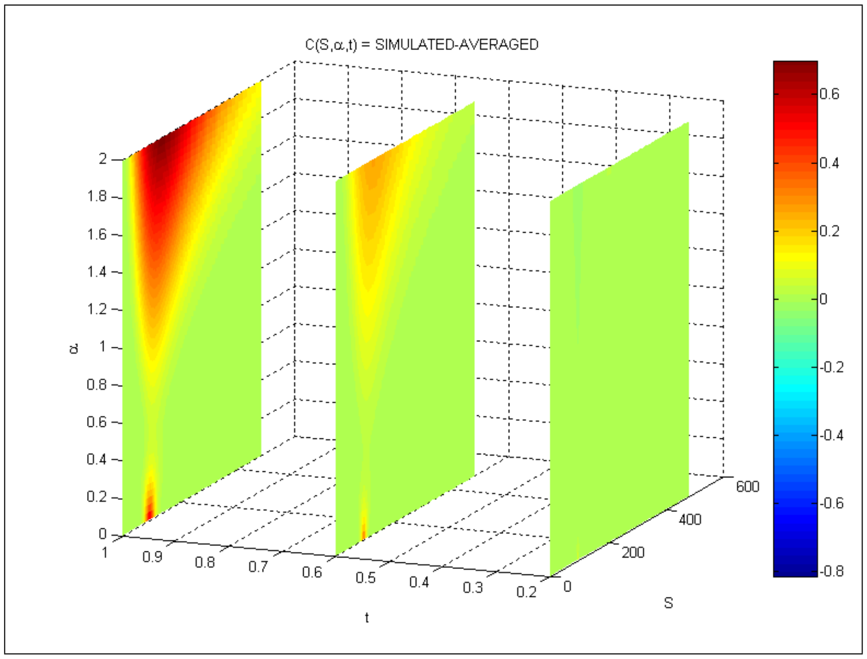


Figure A.2: Residual function $G\left(S, \alpha, t\right)$ for time averaged modification.

Appendix B

Implementation

The program code for performing numerical simulation and testing different algorithms associated with described theory is generated in MATLAB. Below we describe three techniques (tricks) used during program code development for speeding up the simulation:

- **Sparse triplets.** Sparse matrix allows user to work with huge matrix with overwhelming majority of zero elements. In order to store this matrix in memory the MATLAB automatically creates so called vector of sparse triplets:
$$\begin{gathered} i_1, j_1, a_1 \\ \vdots \\ i_n, j_n, a_n \end{gathered}$$
and stores all non zero elements of matrix as a sorted vector with triplet consisting of: row number i, column number j and element value a. Vector is sorted with respect to column indices j (that is to speed up solution of matrix equation using Gaussian elimination method). While assigning non zero value to some element of sparse matrix, MATLAB's algorithm each time automatically forces to run column sorting in order to preserve sorted structure. This might not be time consuming when the user is working with few non zero elements, but in case when each non zero element of sparse matrix should be calculated separately (finite difference stencil for example) that requires a lot of computational time. In order to speed up matrix generation we create vector of sparse triplets and add up new triplet for each non zero element of matrix (of course without intermediate sorting). After all non zero elements of matrix are stored in the vector of sparse triplets we use MATLAB's function `sparse(·)` to transform the vector of sparse triplet into sparse matrix. Described technique might seem naive and simple but it reduces time spend to sparse matrix generation immensely. For example, if finite difference matrix 40000×40000 requires approximately 40 seconds for generation, then using the described computational trick the time reduces to 0.6 sec.!!! This effect is especially noticeable when finite difference matrix has many non zero elements and it should be generated for each time step (for problem with time dependent coefficients).

- **Multiple strikes $\Longleftrightarrow$ Volatility surface.** In the program code we implemented additional feature which allows us to simultaneously solve a problem for different final payoffs (for example puts and/or calls with different strikes). Generating multiple right hand sides[1] B for matrix equation $AX = B$ and solving it simultaneously we can obtain solution for different final payoffs. This trick allows us not only to receive solution for different puts and/or calls, but also to reconstruct volatility surface. For example, after finding solutions for the puts (with different strikes) we can immediately find surface of implied volatility (with respect to strikes and times to maturity). In case of the American options this technique works in the same way, but in this case not only multiple right hand sides should be

[1] Boundary conditions define right hand side of matrix equation.

generated, but also multiple auxiliary functions and multiple obstacles (see sections 2.4 and 3.2.3). The time spent on solving matrix equation with multiple right hand sides is negligible comparing to the time spent on solving the problems separately.

- **Parallel computing.** This technique is not implemented in the program code, but should be mentioned separately. While applying to the most time consuming algorithm of calibration of parameters (up to 10 hours), this method can immensely speed it up. Calibration of parameters assumes that residual functional should be minimized in the space defined by parameters. Minimization problem assumes calculation of gradient i.e. of Hessian matrix (matrix consisting of partial differences of residual functional). In order to find partial derivatives one should calculate value of functional in a few points[2]. And calculation could be performed in parallel (i.e. for each point separately). This could decrease time spent on calibration (i.e. 10 hours spent on calibration of 4 parameters reduces to 2 hours). Additionally, parallel computing could be used for generating finite difference matrix for time dependent parameters.

[2] In initial point and points with increments w.r.t each coordiante. Thus if we minimising functional in three dimensional space (three parameters) we should calculate value of functional in initial point (x, y, z) and in $(x+\Delta x, y, z)$, $(x, y+\Delta y, z)$,$(x, y, z+\Delta z)$.

Appendix C

Formulas for finite difference

Finite difference w.r.t. x

$$\begin{aligned}
\delta_x\left[V_{i,j}\right] &= \frac{V_{i+1,j}-V_{i-1,j}}{2\Delta x}\\
\dot{\delta}_x\left[V_{i,j}\right] &= \frac{V_{i-2,j}-8V_{i-1,j}+8V_{i+1,j}-V_{i+2,j}}{12\Delta x}\\
\delta_{\pm x}\left[V_{i,j}\right] &= \pm\frac{3V_{i,j}-4V_{i\mp1,j}+V_{i\mp2,j}}{2\Delta x}\\
\delta_x^2\left[V_{i,j}\right] &= \frac{V_{i+1,j}-2V_{i,j}+V_{i-1,j}}{\Delta x^2}\\
\dot{\delta}_x^2\left[V_{i,j}\right] &= \frac{-V_{i-2,j}+16V_{i-1,j}-30V_{i,j}+16V_{i+1,j}-V_{i+2,j}}{12\Delta x^2}\\
\delta_{\pm x}^2\left[V_{i,j}\right] &= \frac{V_{i,j}-2V_{i\mp1,j}+V_{i\mp2,j}}{\Delta x^2}
\end{aligned}$$

Finite difference w.r.t. y

$$\begin{aligned}
\delta_y\left[V_{i,j}\right] &= \frac{V_{i,j+1}-V_{i,j-1}}{2\Delta y}\\
\dot{\delta}_y\left[V_{i,j}\right] &= \frac{V_{i,j-2}-8V_{i,j-1}+8V_{i,j+1}-V_{i,j+2}}{12\Delta y}\\
\delta_{\pm y}\left[V_{i,j}\right] &= \pm\frac{3V_{i,j}-4V_{i,j\mp1}+V_{i,j\mp2}}{2\Delta y}\\
\delta_y^2\left[V_{i,j}\right] &= \frac{V_{i,j+1}-2V_{i,j}+V_{i,j-1}}{\Delta y^2}\\
\dot{\delta}_y^2\left[V_{i,j}\right] &= \frac{-V_{i,j-2}+16V_{i,j-1}-30V_{i,j}+16V_{i,j+1}-V_{i,j+2}}{12\Delta y^2}\\
\delta_{\pm y}^2\left[V_{i,j}\right] &= \frac{V_{i,j}-2V_{i,j\mp1}+V_{i,j\mp2}}{\Delta y^2}
\end{aligned}$$

Finite difference w.r.t. τ

$$\begin{aligned}
\delta_{+\tau}\left[V_k\right] &= \frac{V_k-V_{k-1}}{\Delta\tau}\\
\dot{\delta}_{+\tau}\left[V_k\right] &= \frac{3V_k-4V_{k-1}+V_{k-2}}{2\Delta\tau}\\
\ddot{\delta}_{+\tau}\left[V_k\right] &= \frac{11V_k-18V_{k-1}+9V_{k-2}-2V_{k-3}}{6\Delta\tau}
\end{aligned}$$

Cross derivatives:

$$\begin{aligned}
\delta^2_{x,y}\left[V_{i,j}\right] &= \frac{V_{i+1,j+1} - V_{i-1,j-1} - V_{i-1,j+1} + V_{i-1,j-1}}{2\Delta x \Delta y} \\
\delta^2_{+x,y}\left[V_{i,j}\right] &= \frac{V_{i,j+1} - V_{i,j-1} - V_{i-1,j+1} + V_{i-1,j-1}}{2\Delta x \Delta y} \\
\delta^2_{x,+y}\left[V_{i,j}\right] &= \frac{V_{i+1,j} - V_{i-1,j} - V_{i+1,j-1} + V_{i-1,j-1}}{2\Delta x \Delta y} \\
\delta^2_{x,-y}\left[V_{i,j}\right] &= -\frac{V_{i+1,j} - V_{i-1,j} - V_{i+1,j+1} + V_{i-1,j+1}}{2\Delta x \Delta y} \\
\delta^2_{+x,+y}\left[V_{i,j}\right] &= \frac{V_{i,j} - V_{i-1,j} - V_{i,j-1} + V_{i-1,j-1}}{\Delta x \Delta y} \\
\delta^2_{+x,-y}\left[V_{i,j}\right] &= -\frac{V_{i,j} - V_{i-1,j} - V_{i,j+1} + V_{i-1,j+1}}{\Delta x \Delta y} \\
\hat{\delta}^2_{x,y}\left[V_{i,j}\right] &= \frac{1}{2\Delta x \Delta y}\left[V_{i+1,j+1} - 2V_{i,j} + V_{i-1,j-1}\right] - \frac{\Delta x}{2\Delta y}\delta^2_x\left[V_{i,j}\right] - \frac{\Delta y}{2\Delta x}\delta^2_y\left[V_{i,j}\right] \\
\check{\delta}^2_{x,y}\left[V_{i,j}\right] &= \frac{-1}{2\Delta x \Delta y}\left[V_{i+1,j-1} - 2V_{i,j} + V_{i-1,j+1}\right] + \frac{\Delta x}{2\Delta y}\delta^2_x\left[V_{i,j}\right] + \frac{\Delta y}{2\Delta x}\delta^2_y\left[V_{i,j}\right]
\end{aligned}$$

Bibliography

[1] **ANDERSEN L**. A simple approach to the pricing of Bermudan swaptions in the multifactor LIBOR market model. Journal of Computational Finance 3 (2000), 1–32.

[2] **BAXTER M. & RENNIE A**. Financial Calculus: An introduction to derivative pricing. Cambridge University Press. 2002.

[3] **BJORK T**. Arbitrage Theory in Continuous Time Finance, Oxford University Press, 2004

[4] **COX J. & ROSS S. & RUBENSTEIN M.** Option pricing: A simplified approach. Journal of Financial Economics 7 (1979), 229–263.

[5] **HAGAN P.S. & KUMAR D. & LESNIEWSKI A.S. & WOODWARD D.E**. Managing Smile Risk, 84-108, Wilmott Magazine, 2002, Downloadable via: www.math.columbia.edu/~lrb/sabrALL.pdf retrieval date: 2009 January

[6] **HAGAN P.S. & LESNIEWSKI A.S. & WOODWARD D.E**. Probability Distribution in the SABR Model of Stochastic Volatility, Working Paper, 2004. Downloadable via www.wilmott.com/attachments/SABR_ProbDistr.zip

[7] **VAN DER HORST J.**. American Option Pricing in the Heston Model: Dealing with Cash Dividends, M.Sc. Thesis, University of Delft, November 2007

[8] **HULL J.** Options, futures and other derivatives, 6 ed. Pearson Prentice Hall, 2006.

[9] **IKONEN S. & TOIVANEN J.** Operator Splitting Method for Pricing American Options with Stochastic Volatility, University of Jyvaskyla, 2004, Reports of the Department of Mathematical Information Technology, Series B. Scientific Computing, No. B 11/2004

[10] **KLUGE T**. Pricing derivatives in stochastic volatility environment using the finite difference method. Technical report, Technische Universitat Chemintz, 2002.

[11] **LABORDERE P.H.**. A Geometric Approach to the Asymptotic of Implied Volatility, John Willey & Sons, 2009, Frontiers In Quantitative Finance, pp 89-125, Downloadable via: www.cmap.polytechnique.fr/~rama/papers/FrontiersInQuantitativeFinance.pdf, retrieval date: 2009 May

[12] **LONGSTAFF F.A. & SCHWARTZ E.S**. Valuing American Options by Simulation: A Simple Least-Squares Approach, University of California, 2001, The Review of Financial Studies, Vol. 14, No. 1, pp 113-147

[13] **LUENBERGER D.G**. Investment Science, Oxford University Press, 1998

[14] **ROELOF S.**. Pricing Equity Derivatives under Stochastic Volatility: A Partial Differential Equation Approach, M.Sc. Thesis, University of Witwatersrand, August 2007.

[15] **ROSS S.M**. Stochastic Processes, John Willey & Sons, 1996

[16] **SEYDEL R.**. Tools for Computational Finance, Springer, 2006

[17] **STEELE J.M.**. Stochastic Calculus for Financial Application, Springer, 2001

[18] **STENSTOFT L.** Assessing the least squares Monte-Carlo approach to American option valuation. Review of derivatives research 7 (2004), 129–168.

[19] **VELLEKOOP M.H. & NIEUWENHUIS J.W.**. A tree-based Method to price American Options in the Heston Model. Preprint, University of Twente, 2006.

[20] **VLAMING G.**. Pricing Options with the SABR Model, M.Sc. Thesis, University of Utrecht, June 2008

[21] **WEST G.** Calibration of the SABR Model in illiquid Markets, Applied Mathematical Finance, 12 (4), 371-385, 2005.