# Stochastic Local Volatility: Excursions in Finite Differences

ICBI Global Derivatives
Paris
April 2011

Jesper Andreasen
Danske Markets, Copenhagen
kwant.daddy@danskebank.dk

**Outline**

- Motivation: Part A & B.

- Part A: Volatility Interpolation -- from discrete quotes to a full arbitrage consistent continuum.

  - The local volatility model.

  - The implicit finite difference method used for volatility interpolation.

  - Relation to Variance Gamma and Laplace transforms.

  - Numerical example for SX5E.

- Part B: Finite difference based calibration and simulation.
    - The stochastic local volatility model.
    - Backward and forward partial differential equations.
    - Finite difference solution of PDEs and discrete forward-backward equations.
    - Calibration.
    - Simulation in an FD grid.
    - Numerical examples for exotic options.
    - Correlation: Simulation with negative probabilities.
- Conclusion.

## References


- Andreasen, J. and B. Huge (2010): “Finite Difference Based Calibration and Simulation.” www.ssrn.com.
- Andreasen, J. (2009): “Planck-Fokker Boundary Conditions.” Danske Markets working paper.
- Andreasen, J. and B. Huge (2011): “Volatility Interpolation.” *Risk* March, 86-89.
- Carr, P. (2008): “Local Variance Gamma.” Bloomberg working paper.
- Dupire, B. (1994): “Pricing with a Smile.” *Risk* July, 18-20.
- Nabben, R. (1999): “On Decay Rates of Tridiagonal and Band Matrices.” *SIAM J. Matrix Anal. Appl.* 20, 820-837.
- Østerby, O. (2002): “The Finite Difference Method for Parabolic Partial Differential Equations.” Aarhus University.

## Part A: Volatility Interpolation

- Generally, (stochastic) local volatility models require a full arbitrage consistent continuum of European option prices.

- But we only observe European option prices for discrete expiries and strikes.

- We provide an efficient and stable numerical algorithm for interpolation and extrapolation of a set of discrete option prices into a full continuum of arbitrage consistent option prices.

- The method is based on the fully implicit finite difference method and results by Nabben (1999).

- The method is fast: approximately 0.05s for stretching 10 expiries with 10 strikes into a full continuous surface of option prices that is consistent with absence of arbitrage.

**Part B: FD Calibration and Simulation**

- A typical practitioner model set-up is:

  1. SDE: Dynamics given by a an Stochastic Differential Equation.

  2. Calibration: to European options using closed-form results or approximations.

  3. FD: Backward pricing of exotics by numerical solution of finite difference approximation of the partial differential equation (PDE) associated with the SDE.

  4. MC: Forward pricing of exotics by Monte-Carlo simulation of Euler approximation of SDE.

- Example:

    1. SDE: Heston stochastic volatility model.

    2. Calibration: by numerical inversion of Fourier transforms.

    3. FD solution on 2D Craig-Sneyd.

    4. MC by Andersen (2008) approximations of the Heston model.

- Here we present an alternative approach where:

  1. SDE: Stochastic local volatility model.

  2. Calibration by solution of a 2D implicit FD scheme for the forward (Fokker-Planck) PDE.

  3. Backward pricing by solution of a 2D implicit FD scheme for the backward PDE.

  4. Forward pricing by Monte-Carlo simulation in a 2D implicit FD grid.

- Two pieces of innovation here:

  MC simulation in an implicit FD grid can be implemented to be as efficient as Euler discretisation but significantly more accurate.

  The discrete finite difference schemes in (2, 3, 4) can be set up to be mutually consistent.

- Again, the results are based on a combination of results in Nabben (1999) and excursions in the fully implicit finite difference method.

- We will present the results in the context of a stochastic local volatility model but the methodology is generally applicable to other types of models.

- For the SLV case the methodology fits very nicely with the volatility interpolation scheme presented in part A.

- If time permits, we will round off by presenting some ideas for extending the methodology to non-zero correlation between stock and volatility.

## Part A: Local Volatility

- Assume zero rates and that the underlying stock evolves according to the Markov diffusion

$$ds(t) = \sigma(t, s(t))dW \tag{1}$$

- Option prices satisfy the backward PDE

$$0 = V_t + \frac{1}{2}\sigma(t,s)^2 V_{ss} \tag{2}$$

- …subject to boundary conditions defining the particular claim in question.
- The initial European option prices are given by

$$c(t,k) = E[(s(t) - k)^+] \tag{3}$$

- Double integration of the Fokker-Planck equation or local time arguments can be used to derive the Dupire's forward equation

$$0 = -c_t + \frac{1}{2}\sigma(t,k)^2 c_{kk} \quad , c(0,k) = (s_0 - k)^+ \tag{4}$$

- This equation gives a direct link between (time 0) observed option prices and the *local volatility* function:

$$\sigma(t,k)^2 = 2\frac{c_t(t,k)}{c_{kk}(t,k)} \tag{5}$$

- This is the local volatility approach: Numerically differentiate the surface of observed option prices to obtain a local volatility function and use this local volatility function to price exotics by numerical solution of the backward equation or Monte-Carlo simulation.

- This only works if we have an arbitrage consistent surface in expiry and strike of initial option prices, i.e. if

$$c_t(t,k) > 0 \quad , c_{kk}(t,k) > 0 \quad , c_k(t,k) < 0 \tag{6}$$

for all $(t,k)$.

## Volatility Interpolation

- Suppose the volatility function is independent of time and consider the *fully implicit* finite difference discretisation of Dupire's equation:

$$(1-\frac{1}{2}\Delta t\sigma(k)^2\delta_{kk})c(t+\Delta t,k)=c(t,k) \quad ,c(0,k)=(s_0-k)^+ \tag{7}$$

where

$$\delta_{kk}f(k)=\frac{1}{\Delta k^2}(f(k+\Delta k)-2f(k)+f(k-\Delta k)) \tag{8}$$

- In matrix format (7) can be written as

$$Ac(t+\Delta t)=c(t) \tag{9}$$

where

$$A = \begin{bmatrix} 1 & 0 & & & & \\ -z_1 & 1+2z_1 & -z_1 & & & \\ & -z_2 & 1+2z_2 & -z_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -z_{n-1} & 1+2z_{n-1} & -z_{n-1} \\ & & & & 0 & 1 \end{bmatrix}, z_j = \frac{1}{2}\frac{\Delta t}{\Delta k^2}\sigma(k_j)^2 \qquad (10)$$

- This matrix is diagonally dominant with positive diagonal and non-positive off diagonal elements. Nabben (1999) shows that

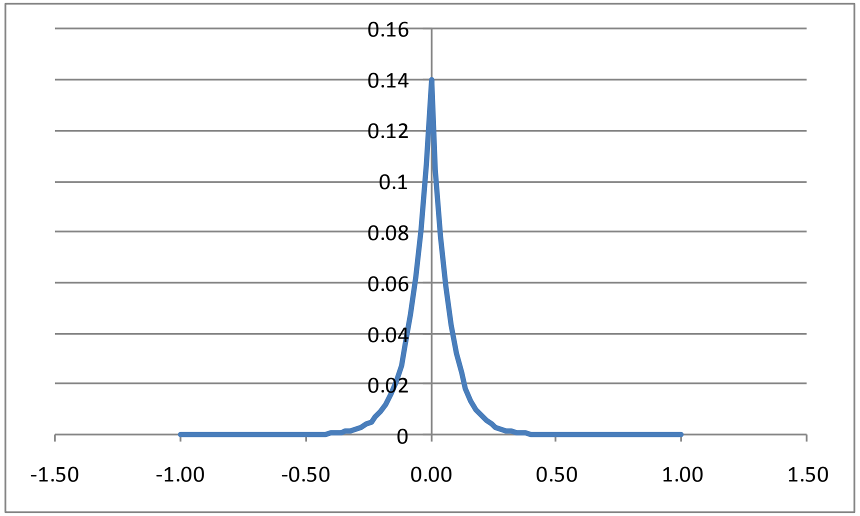
$$A^{-1} \geq 0 \qquad (11)$$

- Also, we have that

$$A^{-1}\iota = \iota \quad , \iota = (1,\ldots,1)' \qquad (12)$$

- …so the rows of $A^{-1}$ can be interpreted as transitions probabilities and the results (11) and (12) imply that these probabilities are *non-negative and sum to unity*.

- The rows of $A^{-1}$ look like exponential densities (for constant volatility):




- The fact that the rows in $A^{-1}$ are densities can be used to show that the mapping (7) is *convexity preserving* and *increasing in expiry*. Hence,

$$\delta_{kk}c(t+h)\geq 0 \quad ,\frac{\partial c(t+h)}{\partial h}=A^{-1}(\frac{1}{2}\sigma^2\delta_{kk}c(t+h))\geq 0 \quad ,\delta_k c(t+h)\leq 0 \tag{13}$$

for all $h\geq 0$ provided that $\delta_{kk}c(t)\geq 0$.

- Proof of convexity preservation:

$$
\begin{aligned}
&(1-h\sigma^2\delta_{kk})c(t+h)=c(t)\\
&\Rightarrow \delta_{kk}(1-h\sigma^2\delta_{kk})c(t+h)=\delta_{kk}c(t)\\
&\Rightarrow (1-h\delta_{kk}\sigma^2)\delta_{kk}c(t+h)=\delta_{kk}c(t)\\
&\Rightarrow \delta_{kk}c(t+h)=(A^{-1})'\delta_{kk}c(t)\geq 0
\end{aligned}
$$

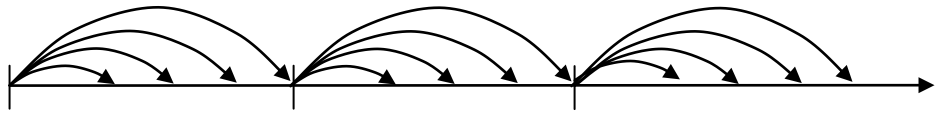
- Proof of increasing values in expiry:

$$
\begin{aligned}
&(1-h\sigma^2\delta_{kk})c(t+h)=c(t)\\
&\Rightarrow \partial_h[(1-h\sigma^2\delta_{kk})c(t+h)]=0\\
&\Rightarrow -\sigma^2\delta_{kk}c(t+h)+(1-h\sigma^2\delta_{kk})\partial_h c(t+h)=0\\
&\Rightarrow \partial_h c(t+h)=(A^{-1})[\sigma^2\delta_{kk}c(t+h)]\geq 0
\end{aligned}
$$

- So given arbitrage consistent prices at one expiry, $c(t_i, k)$, and a volatility function, $\sigma_i(k)$, we can construct a full surface of arbitrage consistent European option prices up to the next expiry by

$$c(t) = [1 - \frac{1}{2}(t - t_i)\sigma^2 \delta_{kk}]^{-1} c(t_i) \quad , t \in [t_i, t_{i+1}] \tag{14}$$

- It is important to note that for all $t \geq t_i$ we roll to $t$ from $t_i$ in *one* step. This (of course) includes the next expiry $t_{i+1}$.



**Volatility Interpolation Algorithm**

- For the strike grid $\{k_j = k_0 + j \cdot \Delta k\}_{j=1,\dots,N}$, the expiries $0 = t_0 < t_1 < \dots < t_m$, and the discrete option prices $\{\hat{c}(t_i, \hat{k}_{ij})\}_{i=1,\dots,m;\, j=1,\dots,n_i}$, the algorithm can be summarized as:

  0. Set $i = 0$ and $c(t_0, k_j) = (s_0 - k_j)^+$.

  1. Let the volatility function $\sigma_i(k)$ be piecewise flat with $n_i$ steps.

  2. Solve the problem non-linear problem

  $$\inf_{\sigma_i(\cdot)} \sum_j (c(t_{i+1}, \hat{k}_{i+1,j}) - \hat{c}(t_{i+1}, \hat{k}_{i+1,j}))^2 \qquad , \quad c(t_{i+1}, k) = [1 - \frac{1}{2}(t_{i+1} - t_i)\sigma_i(k)^2 \delta_{kk}]^{-1} c(t_i, k)$$

  3. If $i < m$ set $i := i + 1$ and go to 1.

4. For each $t_i < t < t_{i+1}$ find the option prices as

$$c(t,k) = [1 - \frac{1}{2}(t - t_i)\sigma_i(k)^2 \delta_{kk}]^{-1} c(t_i, k)$$

- The update in step 2 is extremely quick as it only involves solution of one tri-diagonal matrix system with a computational cost $O(N)$.

- 10 expiries, each with 10 discrete option quotes solved on a FD grid of 100-200 strike points goes in less than 0.05s of CPU time.

- Due to stability of the fully implicit finite difference method the algorithm is robust.

- In fact, we have implemented this as a default pre pricing step for all our stochastic local volatility model.

## Tridag Matrix Solution

- Matrix inversion is normally time consuming but for a tri-diagonal matrix it is actually very easy and $O(N)$. From Numerical Recipes:

```
void
kMatrixAlgebra::tridag(
	int rows, const kValArray<double>& a, const kValArray<double>& b, const kValArray<double>& c,
	const kValArray<double>& r, kValArray<double>& x, kValArray<double>& ws)
{
	int j;
	double bet;

	bet  = 1.0/b(0);
	x(0) = r(0)*bet;

	//	decomposition and forward substitution
	for(j=1;j<rows;++j)
	{
		ws(j) = c(j-1)*bet;
		bet   = 1.0/(b(j) - a(j)*ws(j));
		x(j)  = (r(j) - a(j)*x(j-1))*bet;
	}
	//	backsubstitution
	for(j=rows-2;j>=0;--j)
	{
		x(j)  -= ws(j+1)*x(j+1);
	}
}
```

## Relation between Model and Local Volatility

- In between calibration expiries we have

$$c(t,k) = [1 - \frac{1}{2}(t - t_i)\sigma_i(k)^2 \delta_{kk}]^{-1} c(t_i,k)$$

- Note that the resulting local volatilities $\vartheta(t,k)$ are linked to the model volatilities through the non-linear relationship:

$$0 = [\frac{\partial}{\partial t} + \frac{1}{2}\vartheta(t,k)^2 \frac{\partial^2}{\partial k^2}][1 - \frac{1}{2}(t - t_i)\sigma_i(k)^2 \delta_{kk}]^{-1} c(t_i,k)$$

- So the local volatilities will not be (perfectly) piecewise flat in time even though the generating model volatilities are.

## SX5E Example

- Black volatilities (as of March 2010):

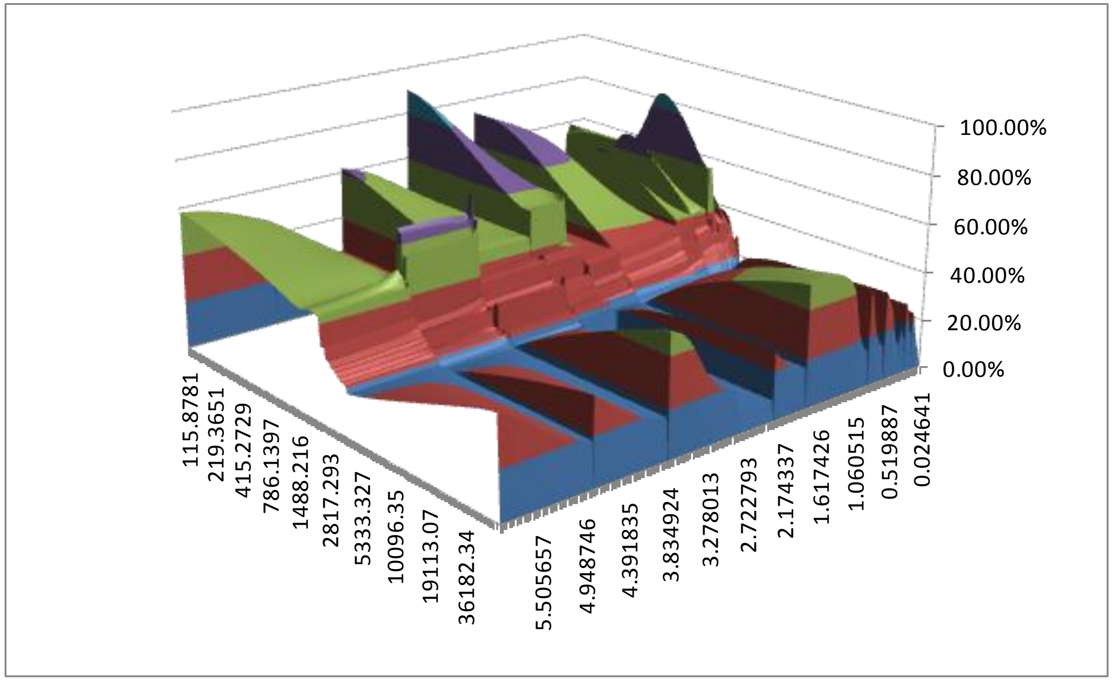
| t\k | 0.5131 | 0.5864 | 0.6597 | 0.733 | 0.7697 | 0.8063 | 0.843 | 0.8613 | 0.8796 | 0.8979 | 0.9163 | 0.9346 | 0.9529 | 0.9712 | 0.9896 | 1.0079 | 1.0262 | 1.0445 | 1.0629 | 1.0812 | 1.0995 | 1.1178 | 1.1362 | 1.1728 | 1.2095 | 1.2461 | 1.3194 | 1.3927 | 1.466 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.025 | | | | | | | | 0.3365 | 0.3216 | 0.3043 | 0.288 | 0.2724 | 0.2586 | 0.2466 | 0.2358 | 0.2247 | 0.2159 | 0.2091 | 0.2056 | 0.2045 | 0.2025 | 0.1933 | | | | | | | |
| 0.101 | | | | | | | | | 0.2906 | 0.2797 | 0.269 | 0.259 | 0.2488 | 0.239 | 0.23 | 0.2213 | 0.214 | 0.2076 | 0.2024 | 0.1982 | 0.1959 | 0.1929 | | | | | | | |
| 0.197 | | | | | | | | | 0.2764 | 0.2672 | 0.2578 | 0.2489 | 0.2405 | 0.2329 | 0.2253 | 0.2184 | 0.2123 | 0.2069 | 0.2025 | 0.1984 | 0.1944 | 0.192 | | | | | | | |
| 0.274 | | | | | 0.3262 | 0.3058 | 0.2887 | | 0.2717 | | 0.2557 | | 0.2407 | | 0.2269 | | 0.2142 | | 0.2039 | | 0.1962 | | 0.1902 | 0.1885 | 0.1867 | 0.1871 | | | |
| 0.523 | | | | | 0.3079 | 0.2936 | 0.2798 | | 0.2663 | | 0.2531 | | 0.2404 | | 0.2284 | | 0.2173 | | 0.2074 | | 0.1988 | | 0.1914 | 0.1854 | 0.1811 | 0.1785 | | | |
| 0.772 | | | | | 0.3001 | 0.2876 | 0.275 | | 0.2637 | | 0.2519 | | 0.2411 | | 0.2299 | | 0.2198 | | 0.2104 | | 0.2022 | | 0.195 | 0.1888 | 0.1839 | 0.1793 | | | |
| 1.769 | | | | | 0.2843 | 0.2753 | 0.2666 | | 0.2575 | | 0.2497 | | 0.2418 | | 0.2347 | | 0.2283 | | 0.2213 | | 0.2151 | | 0.2091 | 0.2039 | 0.199 | 0.1945 | | | |
| 2.267 | | | | | | 0.2713 | | | 0.2555 | | | | 0.241 | | | | 0.2275 | | | | 0.2161 | | | 0.2058 | | | | | |
| 2.784 | 0.3366 | 0.3178 | 0.3019 | 0.2863 | | 0.2711 | | | 0.258 | | | | 0.2448 | | | | 0.2322 | | | | 0.2219 | | | 0.2122 | | 0.2054 | 0.1988 | 0.193 | 0.1849 |
| 3.781 | 0.3291 | 0.3129 | 0.2976 | 0.2848 | | 0.2711 | | | 0.2585 | | | | 0.2469 | | | | 0.2384 | | | | 0.2269 | | | 0.2186 | | 0.2103 | 0.2054 | 0.2002 | 0.1964 |
| 4.778 | | 0.3008 | 0.2975 | 0.2848 | | 0.2722 | | | 0.2611 | | | | 0.2501 | | | | 0.2392 | | | | 0.2305 | | | 0.2223 | | 0.2164 | 0.2105 | 0.2054 | 0.2012 |
| 5.774 | | | | | | 0.2809 | | | 0.2693 | | | | 0.2584 | | | | 0.2486 | | | | 0.2399 | | | 0.2321 | | 0.2251 | 0.219 | 0.2135 | |

- Calibration accuracy

| k\t | 0.5131 | 0.5864 | 0.6597 | 0.733 | 0.7697 | 0.8063 | 0.843 | 0.8613 | 0.8796 | 0.8979 | 0.9163 | 0.9346 | 0.9529 | 0.9712 | 0.9896 | 1.0079 | 1.0262 | 1.0445 | 1.0629 | 1.0812 | 1.0995 | 1.1178 | 1.1362 | 1.1728 | 1.2095 | 1.2461 | 1.3194 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.025 | | | | | | | | 0.01% | -0.07% | 0.02% | 0.01% | -0.02% | 0.00% | 0.02% | -0.01% | 0.01% | 0.01% | 0.01% | -0.06% | 0.00% | -0.10% | -0.02% | | | | | |
| 0.101 | | | | | | | | | -0.05% | 0.01% | 0.01% | -0.02% | 0.00% | 0.01% | -0.01% | 0.00% | -0.01% | 0.00% | -0.01% | 0.00% | -0.09% | 0.03% | | | | | |
| 0.197 | | | | | | | | | 0.01% | 0.00% | 0.00% | 0.00% | 0.01% | -0.01% | 0.00% | 0.00% | 0.00% | 0.02% | 0.00% | -0.02% | 0.00% | -0.04% | | | | | |
| 0.274 | | | | | -0.02% | -0.02% | 0.00% | | 0.02% | | 0.02% | | 0.00% | | 0.00% | | 0.00% | | 0.01% | | -0.02% | | 0.03% | -0.03% | 0.01% | 0.00% | |
| 0.523 | | | | | -0.01% | -0.01% | 0.00% | | 0.01% | | 0.01% | | 0.00% | | 0.00% | | 0.00% | | 0.00% | | 0.00% | | 0.00% | 0.00% | 0.00% | 0.02% | |
| 0.772 | | | | | 0.00% | 0.00% | 0.00% | | -0.01% | | 0.00% | | 0.00% | | 0.00% | | -0.01% | | 0.03% | | 0.02% | | -0.01% | 0.01% | -0.02% | 0.07% | |
| 1.769 | | | | | 0.00% | 0.01% | -0.02% | | 0.01% | | -0.01% | | 0.00% | | 0.00% | | -0.01% | | 0.01% | | -0.01% | | 0.00% | 0.00% | 0.00% | 0.02% | |
| 2.267 | | | | | | 0.00% | | | 0.00% | | | | 0.00% | | | | 0.00% | | | | 0.00% | | | 0.00% | | | |
| 2.784 | 0.00% | 0.00% | 0.00% | 0.00% | | 0.00% | | | 0.00% | | | | 0.00% | | | | 0.00% | | | | -0.01% | | | 0.02% | | -0.03% | 0.00% |
| 3.781 | 0.00% | -0.02% | 0.02% | -0.02% | | 0.01% | | | -0.01% | | | | 0.01% | | | | -0.03% | | | | 0.02% | | | -0.02% | | 0.02% | -0.05% |
| 4.778 | | 0.08% | -0.23% | 0.05% | | 0.06% | | | -0.02% | | | | -0.01% | | | | 0.00% | | | | -0.02% | | | 0.00% | | -0.02% | 0.01% |
| 5.774 | | | | | | 0.00% | | | 0.00% | | | | 0.00% | | | | 0.00% | | | | 0.00% | | | 0.00% | | 0.00% | 0.00% |

## Forward Volatility Example

- The calibration runs in less than 0.05s on a strike grid of 200 points. Std error of 0.03% BS.
- After calibration we compute all the option prices (<0.01s) and from these we can derive the forward volatility surface using the Dupire equation $\vartheta(t,k)^2 = 2c_t(t,k)/c_{kk}(t,k)$.



- The forward volatility surface is not perfectly smooth but the important point is that it contains no poles, i.e. no spikes or negative variance.

- …and consequently, there are no arbitrages in the surface of option prices.

- Within strikes close to spot the local volatilities are almost flat.

## Relation to Variance Gamma and Laplace Transform

- The results that we have presented here are related to the results found by Carr (2008) for the local variance gamma model. However, the ideas of the numerical routine and the filling of the gaps between the discrete expiries by one-stepping the implicit FD algorithm are, to our knowledge, new.

- Consider the option prices generated by the time-homogeneous Dupire equation

$$0 = -g_t + \frac{1}{2}\sigma(k)^2 g_{kk} \quad , g(0,k) = (s_0 - k)^+ \tag{15}$$

- Then the Laplace transform of the option prices

$$h(u,k) = \int_0^\infty \frac{1}{T(u)} e^{-t/T(u)} g(t,k)\,dt \tag{16}$$

- …satisfy the equation of the implicit finite difference equation:

$$[1-\frac{1}{2}T(u)\vartheta(k)^2\frac{\partial^2}{\partial k^2}]h(u,k)=g(0,k) \tag{17}$$

- So the prices of the fully implicit finite difference scheme (7) correspond to option prices from the model (1) where we draw the maturity from an exponentially distribution.

- It also shows that we can introduce a deterministic time change to control the interpolation in the expiry dimension.

- The option prices can also be interpreted as coming from an local volatility variance gamma model with a specific choice of parameters.

## Part B: Stochastic Local Volatility

- Assume zero rates and that the underlying stock and volatility evolve according to the SDEs

$$\begin{aligned} ds &= \sqrt{z}\sigma(t,s)dW \\ dz &= \theta(1-z)dt + \varepsilon z^{\gamma} dZ \quad ,z(0)=1 \\ dW \cdot dZ &= 0 \end{aligned} \tag{18}$$

- For now we assume zero-correlation between the stochastic volatility factor and the underlying stock.

- Derivatives prices $v = v(t, s = x, z = y)$ satisfy the backward PDE

$$\begin{aligned} 0 &= \frac{\partial v}{\partial t} + D_x v + D_y v \\ D_x &= \frac{1}{2}\sigma^2 y \frac{\partial^2}{\partial x^2} \\ D_y &= \theta(1-y)\frac{\partial}{\partial y} + \frac{1}{2}\varepsilon^2 y^{2\gamma} \frac{\partial^2}{\partial y^2} \end{aligned} \tag{19}$$

- …subject to boundary conditions defining the particular claim in question.

- We can find European derivatives prices by integrating the payoff over the marginal density

$$v(0,s,z)=\int\int p(t,x,y)v(t,x,y)dxdy \tag{20}$$

- …where the density (or the Green's function) satisfies the forward (Fokker-Planck) equation

$$\begin{aligned}
&0=-\frac{\partial p}{\partial t}+D_x^* p+D_y^* p \quad ,p(0,x,y)=\delta(x-s(0))\cdot\delta(y-z(0))\\
&D_x^* p=\frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2 yp]\\
&D_y^* p=-\frac{\partial}{\partial y}[\theta(1-y)p]+\frac{1}{2}\frac{\partial^2}{\partial y^2}[\varepsilon^2 y^{2\gamma} p]
\end{aligned} \tag{21}$$

- This equation is dual to the backward equation. At skool the proof is typically buried in adjoint operator theory and things like that.

- The main point here is that (21) runs forward from time 0 whereas (19) runs backward to time 0 from the terminal boundary conditions.

- Essentially, the difference is that the solution to (19) is the price of one derivative for all times, spot and volatility levels whereas the solution to (21) are the densities (as seen from time 0) to all spot and volatility levels in future time.

- …or equivalently the option prices for all strikes and expiries.

- For the initial European option prices

$$c(t,k)=E[(s(t)-k)^+] \qquad (22)$$

- …double integration of the Fokker-Planck equation or local time arguments can be used to derive Dupire's forward equation

$$\sigma(t,k)^2 = \frac{c_t(t,k)}{\frac{1}{2}E[z(t)\,|\,s(t)=k]c_{kk}(t,k)} \qquad (23)$$

- Here we can compute the conditional expectation as

$$E[z(t) \mid s(t)=k] = \int z \Pr(z \mid s(t)=k) dz = \int y \cdot \left( \frac{p(t,k,y)}{\int p(t,k,y') dy'} \right) dy \qquad (24)$$

## Calibration

- The above suggest that the local volatility surface can be calibrated to the initially observed European option prices can by:

  0. Take all European option prices $\{c(t,k)\}_{t\geq 0}$ as given (or from the vol interpol algo)

  1. Discretise (19) for numerical solution on the time grid $0=t_0<t_1<\ldots$. Set $i=1$.

  2. Set $\sigma(t_i,k)^2=2\dfrac{c_t(t_i,k)}{E[z(t_{i-1})|s(t_{i-1})=k]c_{kk}(t_i,k)}$ and solve for (21) for $E[z(t_i)|s(t_i)=k]$.

  3. Set $\sigma(t_i,k)^2=2\dfrac{c_t(t_i,k)}{E[z(t_i)|s(t_i)=k]c_{kk}(t_i,k)}$, $i:=i+1$, and go to 2.

- After calibration of the local volatility surface we can solve for exotic option prices by discrete solution of the backward equation (19) and/or Monte-Carlo simulation of a discretisation of the SDE (18).

- This is the route taken in Lipton (2002) and in the first generation of our own SLV.

- Problem: The discrete forward solution will (typically) not be consistent with the discrete backward solution or the simulation scheme.

- The backward FD issue is typically less of an issue than the consistency with the simulation scheme.

- Very fine time stepping may be necessary in the MC to achieve a sufficient degree of convergence and accuracy.

- Direct discretisation of the forward equation (21) necessitates a set of non-trivial boundary conditions at the edges of the grid. See Lucic (2008).

## Finite Difference Solution

- A slightly primitive discretisation of the backward equation (19) is the LOD scheme

$$\begin{aligned}(1-\Delta t\bar{D}_x)v(t_{i+1/2}) &= v(t_{i+1})\\ (1-\Delta t\bar{D}_y)v(t_i) &= v(t_{i+1/2})\end{aligned} \qquad (25)$$

- …where $\bar{D}$ is a finite difference tridiagonal matrix discretisation of the operator

$$1-\Delta t D_x = \begin{bmatrix} 1 & 0 & & & & \\ -a_1 & 1+2a_1 & -a_1 & & & \\ & -a_2 & 1+2a_2 & -a_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -a_{n-1} & 1+2a_{n-1} & -a_{n-1} \\ & & & & 0 & 1 \end{bmatrix}, a_j = \frac{1}{2}\frac{\Delta t}{\Delta x^2}\sigma(t_i, x_j)^2 \qquad (26)$$

- In (25) the first equation has to be solved for each $y$ and then the second equation has to be solved for each $x$.

- The scheme (25) is as stable as the implicit scheme, in the Nabben sense that we have

$$(1-\Delta t\bar{D}_y)^{-1}(1-\Delta t\bar{D}_x)^{-1} \geq 0$$

and $O(\Delta t+\Delta x^2+\Delta y^2)$ accurate and convergent.

- We have

$$v(t_i)=(1-\Delta t\bar{D}_y)^{-1}(1-\Delta t\bar{D}_x)^{-1}v(t_{i+1}) \quad (27)$$

- Multiplying (27) with a function $p(t,x,y)$ yields

$$p(t_i)'v(t_i)=p(t_i)'(1-\Delta t\bar{D}_y)^{-1}(1-\Delta t\bar{D}_x)^{-1}v(t_{i+1}) \quad (28)$$

- So if we define $p(t,x,y)$ to be the solution to

$$
\begin{aligned}
&p(t_0) = 1_{x=x(0)} 1_{y=y(0)} \\
&(1-\Delta t \bar{D}_y)' p(t_{i+1/2}) = p(t_i) \\
&(1-\Delta t \bar{D}_x)' p(t_{i+1}) = p(t_{i+1/2})
\end{aligned} \tag{29}
$$

- …then $p(t,x,y)$ is the Green's function for the discrete system (25), in the sense that European option prices that satisfy (25) also satisfy

$$
v(t_0) = p(t_n)' v(t_n) = \sum_x \sum_y p(t_n, x, y) v(t_n, x, y) \tag{30}
$$

- Equation (29) is thus exactly consistent with the backward scheme (25).

- The adjoint operators are in this case replaced by the transpose of the matrix. There is no need to take special care of the boundary conditions at the edges because these are implicitly defined by the backward scheme.

- (29) can in turn be manipulated to yield the following equation for the European call values

$$0=-\frac{1}{\Delta t}[c(t_{h+1},x)-c(t_{h+1/2},x)]+\frac{1}{2}\sigma_h(x)^2 E[y(t_{h+1})|x]\delta_{xx}c(t_{h+1},x)$$

$$c(t_{h+1/2},k)=\sum_x\sum_y(x-k)^+\,p(t_{h+1/2},x,y) \qquad (31)$$

$$E[y(t_{h+1})|x]=\frac{\sum_y yp(t_{h+1},x,y)}{\sum_y p(t_{h+1},x,y)}$$

- …which is the discrete version of the stochastic volatility Dupire equation

$$0=-\frac{\partial c}{\partial t}+\frac{1}{2}E[\sigma^2|s=k]\frac{\partial^2 c}{\partial k^2} \qquad (32)$$

- Equation (31) is used as the basis for calibrating the model to observed European call prices, rather than the continuous time equation (32).

- This achieves full consistency between the backward FD solution and the calibration.

- Calibration algorithm:

  1. Take all European option prices $\{c(t,k)\}_{t\geq 0}$ as given (or from the vol interpol algo)

  2. Discretise the model on the time grid $0=t_0<t_1<\ldots$. Set $h=0$.

  3. Set $\sigma(t_h,x)^2 = 2\dfrac{c(t_{h+1},x)-c(t_h,x)}{E[y(t_h)\,|\,x]\delta_{xx}c(t_{h+1},x)}$ .

  4. Solve (29) for $p(t_{h+1},x,y)$.

  5. Set $\sigma(t_h,x)^2 = 2\dfrac{c(t_{h+1},x)-c(t_{h+1/2},x)}{E[y(t_{h+1})\,|\,x]\delta_{xx}c(t_{h+1},x)}$.

  6. $h:=h+1$, and go to 3.

- The steps 4-5 can in principle be repeated but this does not seem necessary in practice.

**Monte-Carlo**

- We have the transition

$$v(t_i) = \underbrace{(1-\Delta t\bar{D}_y)^{-1}}_{random\, y}\underbrace{(1-\Delta t\bar{D}_x)^{-1}}_{random\, x} v(t_{i+1}) \tag{33}$$

- The point here is that this can be simulated as first step $y$ and then step $x$.

- Again, the matrix

$$A = 1-\Delta t D_x = \begin{bmatrix} 1 & 0 & & & & \\ -a_1 & 1+2a_1 & -a_1 & & & \\ & -a_2 & 1+2a_2 & -a_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -a_{n-1} & 1+2a_{n-1} & -a_{n-1} \\ & & & & 0 & 1 \end{bmatrix}, a_j = \frac{1}{2}\frac{\Delta t}{\Delta x^2}\sigma(t_i, x_j)^2 \tag{34}$$

- …is diagonally dominant with positive diagonal and non-positive off diagonal elements. Nabben (1999) shows that

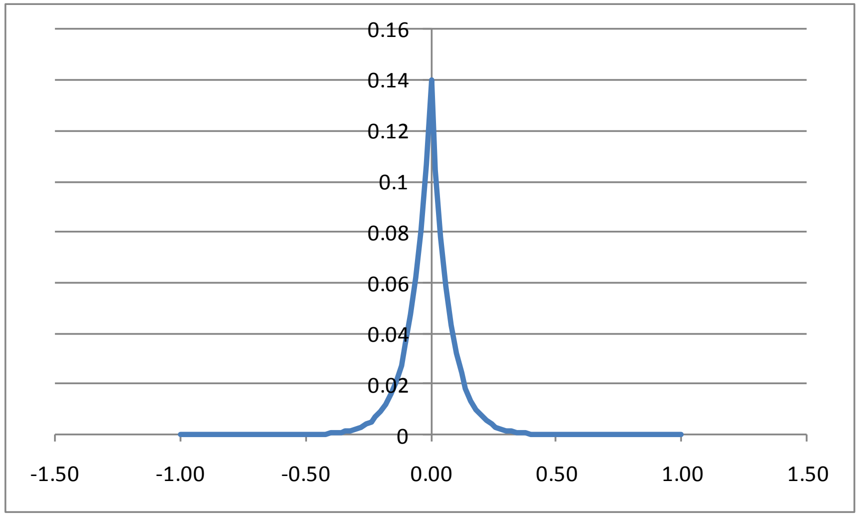
$$A^{-1} \geq 0 \tag{35}$$

- Also, as

$$A^{-1}\iota = \iota \quad , \iota = (1, \ldots, 1)' \tag{36}$$

- …the rows of $A^{-1}$ can be interpreted as *transitions probabilities* and the results (35) and (36) imply that these probabilities are *non-negative* and *sum to unity*.

- As we have seen the rows of $A^{-1}$ look like exponential densities:




- The transition matrices

$$(1-\Delta t\bar{D}_y)^{-1}(1-\Delta t\bar{D}_x)^{-1}$$

can thus be seen as discrete Markov chains.

## Decomposition of Tridiagonal Matrices

- Computing inverses of matrices is generally very time consuming, $O(n^3)$.

- Though we also know that tridiagonal matrix equations $Ax = b$ can be solved in linear time $O(n)$.

- Nabben (1999), however, show that the inverse of a tridiagonal matrix $A^{-1}$ can be found in *linear time* $O(n)$ in the sense that there exist vectors $x, y, p, q$ which can be found in linear time so that

$$A_{ij}^{-1} = \begin{cases} x_i y_j q_j / q_i & ,i \le j \\ x_j y_i p_i / p_j & ,i > j \end{cases} \tag{37}$$

- For simulation we need the distribution functions

$$Q_{kl} = Q_k(x_l) = \Pr(x(t_{i+1}) \le x_l \,|\, x(t_i) = x_k) = \sum_{m \le l} A_{km}^{-1} \tag{38}$$

- The distribution functions can be used for propagating $x$ using uniforms by the scheme

$$x(t_{i+1}) = Q_k^{-1}(u) \quad ,u \sim U(0,1) \tag{39}$$

if $x(t_i) = x_k$.

- The distribution functions can also be found in linear time, in the sense that there exists linear time algorithms that produce decomposition vectors $U,V$ so that $Q_{kl} = U_k V_l$.

- This can be combined with bisection to yield efficient routines for simulation.

- The trouble, however, with all these routines is that they are prone to numerical overflow, in the sense that the elements of the vectors $U,V$ tend to be either very large (say 1e23) or very small (say 1e-23).

- So in our implementation we actually use an algorithm that finds the diagonal elements of $\{Q_{kk}\}$ (in linear time) and then uses a recursive formula to step away from the diagonal.

- The point here is that in most cases we will only need a few steps (0,1 or 2) to find the right column entry.

- Huge (2010) presents an $O(n)$ algorithm for identifying $\tilde{x},\tilde{y},\tilde{d} \in \mathbb{R}^n$ satisfying

$$\sum_{j \le i} A_{ij}^{-1} = \tilde{d}_i$$

$$A_{ii}^{-1} = \tilde{x}_i \tilde{y}_i / A_{ii}$$

$$A_{ij}^{-1} = A_{i,j+1}^{-1} \cdot \frac{\tilde{x}_j}{\tilde{x}_{j+1}} \cdot \frac{-A_{j+1,j}}{A_{jj}} \quad , j < i$$

$$A_{ij}^{-1} = A_{i,j-1}^{-1} \cdot \frac{\tilde{y}_j}{\tilde{y}_{j-1}} \cdot \frac{-A_{j-1,j}}{A_{jj}} \quad , j > i$$

- This combined with $Q_{ij} = Q_{ii} - \sum_{k=j+1}^{i} A_{ik}^{-1} + \sum_{k=i+1}^{j} A_{ik}^{-1}$

… can be used for going from one column to the next in the transition matrix.

# Algorithm for Decomposition of Tridiagonal Matrix

```
//  the vectors a, b, c are the three diagonals of the matrix A
void
hugeDecomp(
    int n, const double* a, const double* b, const double* c,   //  input
    double* x, double* y, double* d)                            //  output
{
    //  dimension check
    if(n<=0) return;
    if(n==1)
    {
        x[0] = 1.0;
        y[0] = 1.0;
        d[0] = 1.0/b[0];
        return;
    }

    //  help
    int i;

    //  fwd
    x[0] = 1.0;
    x[1] = 1.0;
    for(i=2;i<n;++i)
    {
        x[i] = x[i-1] - (a[i-1]/b[i-1])*(c[i-2]/b[i-2])*x[i-2];
    }

    //  bwd
    y[n-1] = 1.0/(x[n-1] - (a[n-1]/b[n-1])*(c[n-2]/b[n-2])*x[n-2]);
    y[n-2] = y[n-1];
    for(i=n-3;i>=0;--i)
    {
        y[i] = y[i+1] - (a[i+2]/b[i+2])*(c[i+1]/b[i+1])*y[i+2];
    }

    //  set d
    d[0] = x[0]*y[0]/b[0];
    for(i=1;i<n;++i)
    {
        d[i] = -(a[i]/b[i])*(y[i]/y[i-1])*d[i-1] + x[i]*y[i]/b[i];
    }
}
```

## Simulation Algorithm

- Simulation algorithm

  0. Suppose $x(t_h) = x_i$. Set $j = i$.

  1. Draw a uniform $\tilde{u} \sim U(0,1)$.

  2. If $\tilde{u} \le Q_{ii}$: while $\tilde{u} < Q_{i,j-1}$ set $j := j-1$. Go to 4.

  3. If $\tilde{u} > Q_{ii}$: while $\tilde{u} > Q_{i,j+1}$ set $j := j+1$. Go to 4.

  4. Set $x(t_{h+1}) = x_j$.

- Note that all calculations here are based on simple operations – no exp() or ln() or similar.

- In typical implementation 80% of the time will be spent in step 1.

## Monte-Carlo Implementation

- Before we do the simulations we need to solve one (calibration) forward grid and the equivalent of one backward grid (decomposition algorithm). [Could actually be combined.]

- In applications this runs in around 0.4s per underlying asset.

- The simulations themselves are very quick and only marginally more expensive than Euler discretisation and only involve elementary functions.

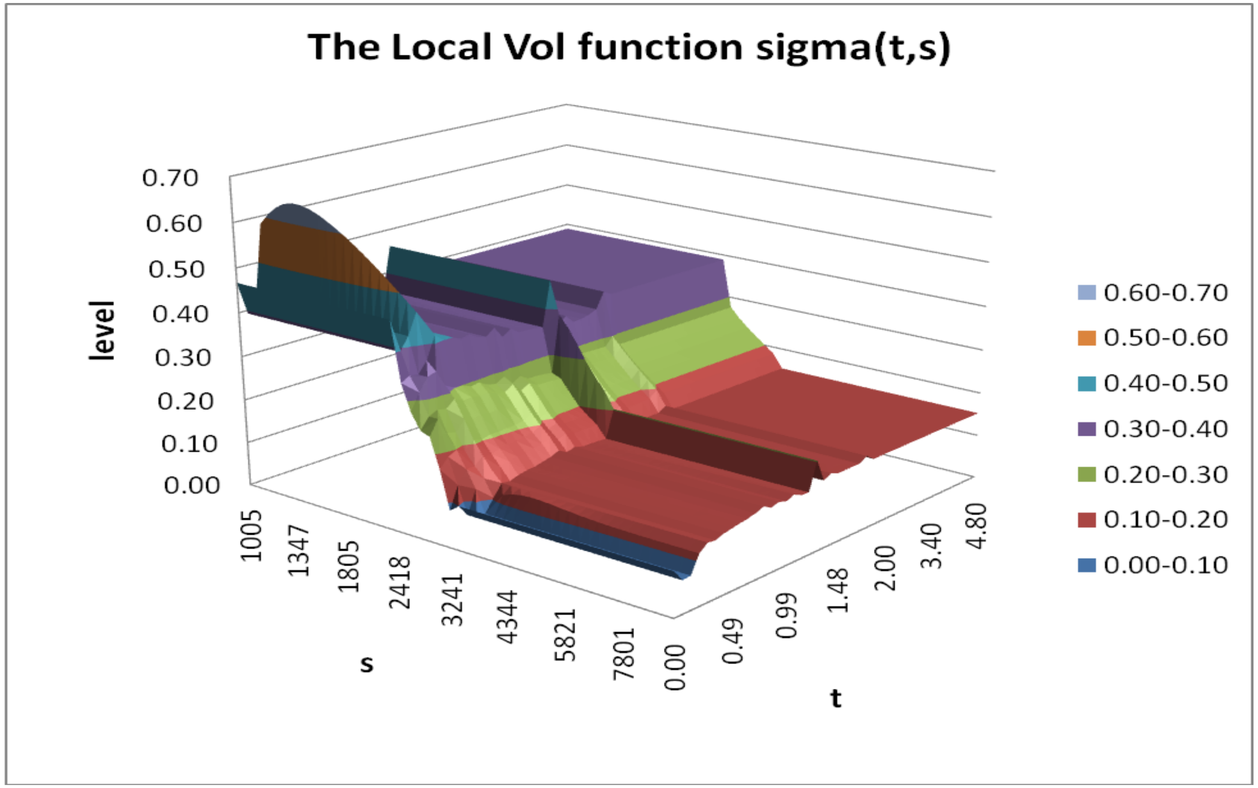
- ….but the important point is that the we do not need to do the fine time-stepping that is normally necessary with the Euler stepping.

- In fact, in the case of very frequent sampling is needed for a particular product we may use a Brownian bridge for sampling in between FD grid time points.

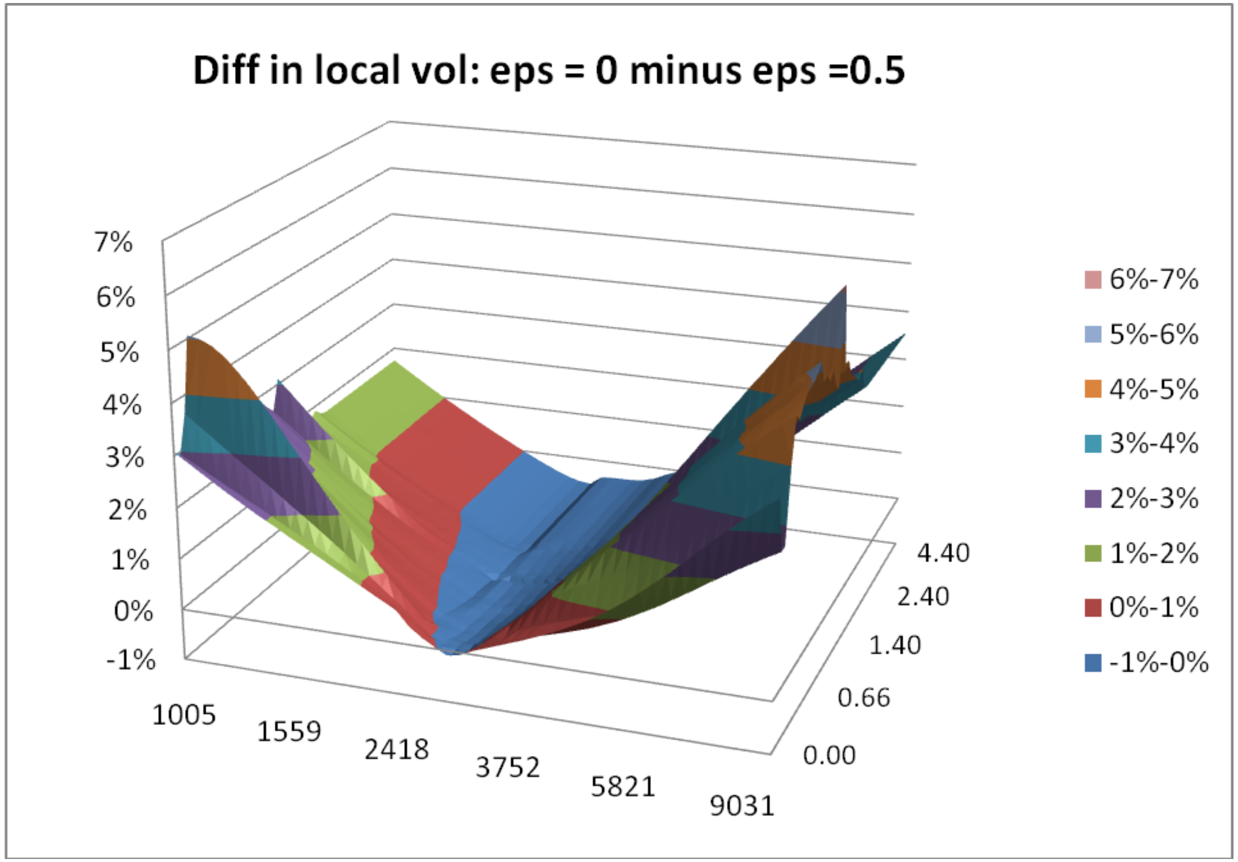
- Multi asset can be implemented by using a Gaussian copula between the different underlying stocks.

## Numerical Examples: Calibration

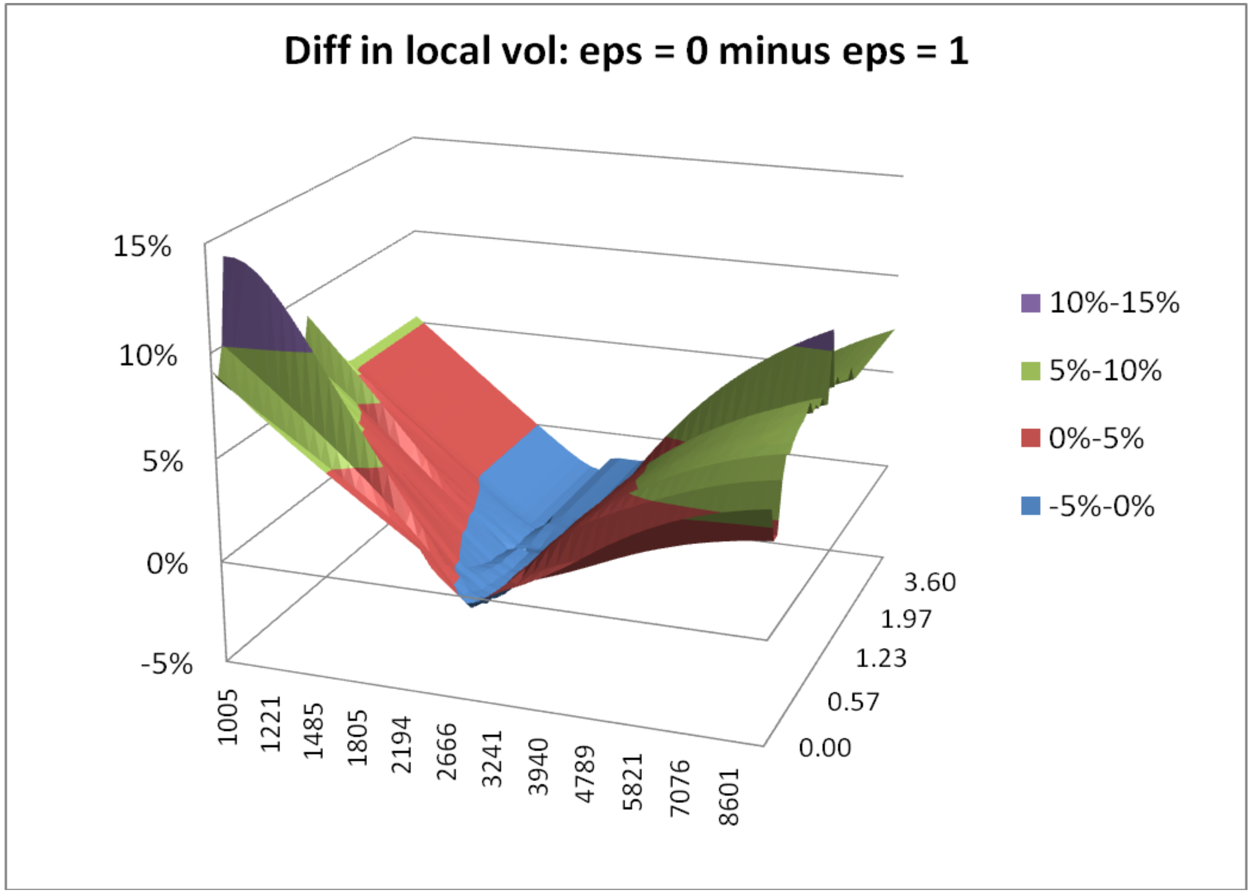
- SX5E on Feb 21, 2011. LVI model first used for interpolation and extrapolation of European option price quotes. Local volatility function for vol-of-var $\varepsilon = 0$:



- The difference between local volatilities for $\sigma(t,k)|_{\varepsilon=0} - \sigma(t,k)|_{\varepsilon=0.5}$:



- The difference between local volatilities for $\sigma(t,k)|_{\varepsilon=0} - \sigma(t,k)|_{\varepsilon=1}$:



**Numerical Examples: Exotic Options**

- MC simulate monthly returns over 36 months:

$$R_i = \frac{S(t_i)}{S(t_{i-1})} - 1$$

- Different pay-offs

- Realised variance: $V = \sum_i R_i^2$
- Capped realised variance: $\min(V, \bar{K})$
- Floored realised variance: $\max(\underline{K}, V)$
- Straddle sum: $\sum_i |R_i|$

- Sensitivity to vol-of-vol:

| Vol of Var | | | | | |
|---|---|---|---|---|---|
| dp slv mc.epsilon | 0.0000% | 100.0000% | 200.0000% | 300.0000% | |
| PRICES | | | | | Std error |
| *CAPPEDREALVAR* | 0.1357 | 0.1285 | 0.1164 | 0.1078 | 0.0007 |
| *FLOOREDREALVAR* | 0.2423 | 0.2466 | 0.2567 | 0.2665 | 0.0027 |
| *REALVAR* | 0.1891 | 0.1862 | 0.1843 | 0.1854 | 0.0030 |
| *STRADDLES* | 1.8123 | 1.7155 | 1.5230 | 1.3460 | 0.0089 |

- Intuition:

  Realised variance ~ variance swap ~ log contract ~ European pay-off.
  Straddles ~ sqrt(variance) => concave in variance => decrease in vol-of-vol.
  Capped variance ~ realised variance – cap on variance => decrease in vol-of-vol.
  Floored variance ~ realised variance + floor on variance => increase in vol-of-vol.

## Conclusion

- Excursions in the implicit finite difference method to come up with:
  - An arbitrage free volatility interpolation/extrapolation scheme.
  - Forward-backward consistent PDE pricing and calibration.
  - Consistent MC pricing and calibration.
- Methods are quick and simple in the sense that they only require elementary operations.
- Basic idea is to use the fully implicit finite difference method as a discrete model in itself rather than purely a tool for approximation of the continuous PDE.
- Doing so we are able to develop an effective volatility interpolation scheme and achieve full consistency between calibration, backward pricing and simulation in SLV models.

- Extensions:

  - SLV Model extends to the multi asset case via Gaussian copulas.

  - One-step FD grids can be used to rectify option price approximations (such as SABR) that produce negative implied densities.

- Non-zero correlation and applications to hybrids and interest rates under investigation.