A Appendix

A.1 Problem setting.

Assume that our market model has N different assets. The price of asset number i at time t will be described by stochastic process S_t^i. Our model will use the concept of stochastic volatility and stochastic correlation, so we will have N volatility processes, denoted by ν_t^i and correlation process denoted by ρ_t. General setting of our model will be described by the system of SDEs.

$$dS_t^i = \mu^i(t, S_t^i)dt + \sigma^i(t, S_t^i, \nu_t^i)dW_t^i,\ i = 1, ..., N$$

$$d\nu_t^i = m^i(t, \nu_t^i)dt + D^i(t, S_t^i, \nu_t^i)dB_t^i,\ i = 1, ..., N$$

$$d\rho_t^i = \alpha\Big(\Psi(\nu_t) - \rho_t\Big)dt + \Omega(\rho_t)d\hat{W}_t$$

Where dW_t^i, dB_t^i and $d\hat{W}_t$ are Wiener processes with covariation matrix:

$$\mathcal{C} = \begin{pmatrix} 1 & \rho_t & ... & \rho_t & \rho_{11} & ... & \rho_{1N} & \tilde{\rho} \\ \rho_t & \vdots & ... & \vdots & \vdots & ... & ... & \vdots \\ \vdots & \vdots & ... & \vdots & \vdots & ... & ... & \vdots \\ \rho_t & ... & ... & 1 & \rho_{N1} & ... & \rho_{NN} & \tilde{\rho} \\ \rho_{11} & ... & \rho_{1N} & 1 & \hat{\rho}_{12} & ... & \hat{\rho}_{1N} & \tilde{\rho} \\ \vdots & \vdots & ... & \vdots & \vdots & ... & ... & \vdots \\ \rho_{N1} & ... & \rho_{NN} & \hat{\rho}_{N1} & ... & ... & 1 & \tilde{\rho} \\ \tilde{\rho} & ... & ... & \tilde{\rho} & \tilde{\rho} & ... & \tilde{\rho} & 1 \end{pmatrix}$$

A.2 Simple Euler scheme.

Here is described the simplest simulation scheme for our model. Assume that we have initial values: $t_0, S_0^1, ..., S_0^N, \nu_0^1, ..., \nu_0^N, \rho_0$. Now, inductively assume that we have done i iterations.

Step 0.
Generate independently identically distributed standard normal variables: $\xi_1(i), ..., \xi_N(i), \eta_1(i), ..., \eta_N(i),$
Step 1.
Calculate the vector of Wiener increments:

$$\begin{pmatrix} \Delta W_t^1(i) \\ \vdots \\ \Delta B_t^1(i) \\ \Delta\hat{W}_t(i) \end{pmatrix} = \mathcal{C}^{\frac{1}{2}}\sqrt{dt}\begin{pmatrix} \xi^1(i) \\ \vdots \\ \eta^1(i) \\ \zeta(i) \end{pmatrix}$$

Where $\mathcal{C}^{\frac{1}{2}}$ is Cholesky decomposition of $\mathcal{C}$.
Step 2.

Generate volatility-process increments (do not confuse index i with imaginary unit):

$$\nu^j(i+1) = \nu^j(i) + m^j(i*h, \nu^j(i))*h + D^j(i*h, \nu^j(i))*\Delta B^j(i)$$

Generate correlation-process increment:

$$\rho(i+1) = \rho(i) + \alpha\Big(\Psi(\nu(i) - \rho(i)\Big) + \Omega(\rho(i))*\Delta\hat{W}(i)$$

Generate asset-price-process increment:

$$S^j(i+1) = S^j(i) + \mu^j(i*h, S^j(i))*h + \sigma^j(i*h, S^j(i), \nu^j(i))\Delta W^j(i)$$

A.3 First and second refinements.

First of all lets illustrate some theory that stands behind our calculations. Take a process X^t that can be obtained as a solution of the SDE:

$$dX_t = \mu_t(.)dt + \sigma_t(.)dW_t$$

Simple Euler scheme approximated process increment using the following expansion:

$$X_{t+h} = X_t + \int\limits_t^{t+h} \mu_u(.)du + \int\limits_t^{t+h} \sigma_u(.)dW_u$$

In the simplest scheme we were taking $\int\limits_t^{t+h} \mu_u(.)du \approx \mu_t(.)*h$, $\int\limits_t^{t+h} \sigma_u(.)dW_u \approx \sigma_t(.)*\Delta W$. To make our simulation more accurate we may want to apply Ito formula to coefficient functions μ, σ and then make our approximation.

$$\mu_u(.) = \mu_t(.) + \int\limits_t^u \mathcal{L}_0\mu_s ds + \int\limits_t^u \mathcal{L}_1\mu_s dW_s$$

Same logic holds for any coefficient function.

So, our plan is to find operators $\mathcal{L}_0, \mathcal{L}_1$ for all types of coefficient functions that appear in our model. Calculations logic and general formula is given in Gatheral. Here we are going to adjust it to our model.

A.3.1 Asset-price drift μ^j

$$\mathcal{L}^{\text{drift}} = \frac{\partial}{\partial t} + \mu^j\frac{\partial}{\partial S^j} + \frac{(\sigma^j)^2}{2}\frac{\partial^2}{(\partial S^j)^2}$$

$$\mathcal{L}^{\text{diffusion}} = \sigma^j\frac{\partial}{\partial S^j}$$

A.3.2 Asset-price diffusion σ^j

$$\mathcal{L}^{\text{drift}} = \frac{\partial}{\partial t} + \mu^j \frac{\partial}{\partial S^j} + m^j \frac{\partial}{\partial \nu^j} + \frac{(\sigma^j)^2}{2} \frac{\partial^2}{(\partial S^j)^2} + \rho_{jj} \sigma^j D^j \frac{\partial^2}{\partial S^j \partial \nu^j} + \frac{(D^j)^2}{2} \frac{\partial^2}{(\partial \nu^j)^2}$$

$$\mathcal{L}_\sigma^{\text{diffusion}} = \sigma^j \frac{\partial}{\partial S^j}$$

$$\mathcal{L}_\sigma^{\text{diffusion}} = D^j \frac{\partial}{\partial \nu^j}$$

A.3.3 Volatility drift m^j and diffusion D^j

$$\mathcal{L}^{\text{drift}} = \frac{\partial}{\partial t} + m^j \frac{\partial}{\partial \nu^j} + \frac{(D^j)^2}{2} \frac{\partial^2}{(\partial \nu^j)^2}$$

$$\mathcal{L}^{\text{diffusion}} = D^j \frac{\partial}{\partial \nu^j}$$

A.3.4 Correlation drift

$$\mathcal{L}^{\text{drift}} = \sum_{k=1}^{N} m^k \frac{\partial}{\partial \nu^k} + \alpha\Big(\Psi(\nu) - \rho\Big) \frac{\partial}{\partial \rho} + \frac{1}{2} \sum_{k=1}^{N} \sum_{l=1}^{N} D^k D^l \frac{\partial^2}{\partial \nu^k \partial \nu^l} + \sum_{k=1}^{N} \sigma^k \Omega \frac{\partial^2}{\partial \nu^k \partial \rho}$$

$$\mathcal{L}_D^{\text{diffusion}} = D^j \frac{\partial}{\partial \nu^j}$$

$$\mathcal{L}_\Omega^{\text{diffusion}} = \Omega \frac{\partial}{\partial \rho}$$

A.3.5 Correlation diffusion Ω

$$\mathcal{L}^{\text{drift}} = \alpha\Big(\Psi(\nu) - \rho\Big) \frac{\partial}{\partial \rho} + \frac{\Omega^2}{2} \frac{\partial^2}{(\partial \rho)^2}$$

$$\mathcal{L}^{\text{diffusion}} = \Omega \frac{\partial}{\partial \rho}$$

After substituting all of the Ito operators in our simulation scheme and after approximation double integrals will appear. Gatheral provides us with ready solutions. Assume h is the size of time-step of our simulation.

$$I(0,0) := \int_t^{t+h} \int_t^{u} ds du = \frac{h^2}{2}$$

$$I(k,0) = I(0,k) := \int_t^{t+h} \int_t^{u} ds dW_u^k = \frac{h}{2} \Delta W^k$$

$$I(j,k) := \int\limits_{t}^{t+h} \int\limits_{t}^{u} dW_s^j dW_u^k = \frac{1}{2}\Big(\Delta W^j \Delta W^k - V^{jk}\Big)$$

,
where $V^{jk}, j < k - -$ iid. $V^{jk} = h$ with probability $p = 0.5$ and $-h$ o/w. $V^{kj} = -V^{jk}$.

Closest plans. Research proposal.

The simulation scheme represented in previous section assumes that we know the closed form of all coefficient functions, with may provide us with a huge amount of parameters. After implementing Euler scheme, refinements and all needed classes we are facing the problem of calibration. To calibrate such a huge model we need a lot of computational resources with may play a bad joke with us.

First way to handle calibration problem is to build a neural network (nn) with will interpolate the output of our model. This approach is appropriate when we know the exact derivative we are pricing. The learning of our nn is fully offline, so the calibration requires only predict-operations, which are pretty fast. So, deep learning in the role of interpolation is an interesting branch for our research.

Second way to calibrate our model is called the Markovian projection. The method is studied in academy and is used in the industry. This one will help us to reduce the amount of SDEs in our system from $2N + 1$ to at most $N + 1$, which will reduce the amount of parameters.

After dealing with calibration we are ready to think about applications. Main applied problem, our model can help us with is the pricing of a derivative in case its price is sensitive to correlation between underlying assets. I'm also planning to check whether my model is more robust than usual benchmarks in periods of market failure.