Lecture 6: Exotic Markets

Modeling and Marketing Making in Foreign Exchange

Local Vol/Stoch Vol Mixture

- The Street has largely converged on local volatility/stochastic volatility mixture models
  - The mixture parameter controls risk reversal beta

- … but these models can be slow
  - Two-factor models (spot & volatility factor)
  - Numerical calibration
  - Numerical vanilla & exotic pricing
  - Even using backward induction, two factor models are slow
    - Quite uncommon to use Monte Carlo in FX

LV/SV Approximation

- Some shops trade off model accuracy for computational speed by using a simpler model that has most of the same features as the full model
- Model spot as a diffusive process still…
- … but allow the stochastic volatility factor to take only a few discrete values instead of modeling as a diffusive process
  - eg two or three “layers” for stochastic volatility
  - Stochastic volatility factor jumps between the different discrete values according to a Markov process

LV/SV Approximation

- One simple formulation

$$\frac{dS(t)}{S(t)} = \mu\, dt + \sigma(S,t)\sqrt{v(t)}\, dz_s(t)$$

$$v(t) = e^{Y(t)\varepsilon}$$

- S(t) = spot at time t, σ(S,t) = local volatility function, v(t) = stochastic vol factor
- Y(t) can be -1, 0, or +1
- ε is something like a volatility of volatility

LV/SV Approximation

- Need a way to define the transition probabilities for Y(t)
  - Markov transition matrix

- Define transition probabilities to mimic a mean-reverting square root process for v(t) as closely as possible
  - An approximation to the Heston-like formulation of LV/SV

$$dv(t) \sim \beta(1 - v(t))dt + \alpha\sqrt{v(t)}\, dz_v(t)$$

LV/SV Approximation

- Six transition probabilities are required
  - Describe them as “frequencies” – probability of transition in a time period dt equals frequency * dt
    - Like a Poisson process jump
  - λ_{0+} and λ_{0-} represent the frequencies of jumping from Y=0 to Y=+1 or Y=-1 respectively
  - λ_{+0} and λ_{+-} represent the frequencies of jumping from Y=+1 to Y=0 or Y=-1 respectively
  - λ_{-0} and λ_{-+} represent the frequencies of jumping from Y=-1 to Y=0 or Y=+1 respectively

- We’ll choose these frequencies to try to match the Heston-like diffusive process mean and variance in each state

LV/SV Approximation

- Consider the middle state, Y=0
  - Here we want zero drift (already at the mean level)
  - Variance = α^2 dt to approximate the Heston process variance

$$E[v(t+dt)-v(t)]_0 = \lambda_{0+}dt\left(e^{\varepsilon}-1\right)+\lambda_{0-}dt\left(e^{-\varepsilon}-1\right)=0$$

$$Var[v(t+dt)-v(t)]_0 = \lambda_{0+}dt\left(e^{\varepsilon}-1\right)^2+\lambda_{0-}dt\left(e^{-\varepsilon}-1\right)^2=\alpha^2 dt$$

LV/SV Approximation

- Can rearrange those to solve for the transition frequencies that match the Heston-like process mean and variance from that state

$$\lambda_{0+} = \frac{\alpha^2}{\left(e^{\varepsilon} - 1\right)\left(e^{\varepsilon} - e^{-\varepsilon}\right)}$$

$$\lambda_{0-} = \frac{\alpha^2}{\left(1 - e^{-\varepsilon}\right)\left(e^{\varepsilon} - e^{-\varepsilon}\right)}$$

LV/SV Approximation

- Now consider the high-volatility state Y=+1
  - Should have a negative expected value due to mean reversion

$$E[v(t+dt)-v(t)]_+ = -\lambda_{+0}dt\left(e^{\varepsilon}-1\right)-\lambda_{+-}dt\left(e^{\varepsilon}-e^{-\varepsilon}\right) = -\beta dt\left(e^{\varepsilon}-1\right)$$

$$Var[v(t+dt)-v(t)]_+ = \lambda_{+0}dt\left(e^{\varepsilon}-1\right)^2+\lambda_{+-}dt\left(e^{\varepsilon}-e^{-\varepsilon}\right)^2 = \alpha^2 e^{\varepsilon}dt$$

LV/SV Approximation

- Similarly can write out the mean and variance required in the lower state
  - Expected value should be positive due to mean reversion

$$E[v(t+dt)-v(t)]_{-} = \lambda_{-0}dt\left(1-e^{-\varepsilon}\right) + \lambda_{-+}dt\left(e^{\varepsilon}-e^{-\varepsilon}\right) = \beta dt\left(1-e^{-\varepsilon}\right)$$

$$Var[v(t+dt)-v(t)]_{-} = \lambda_{-0}dt\left(1-e^{-\varepsilon}\right)^{2} + \lambda_{-+}dt\left(e^{\varepsilon}-e^{-\varepsilon}\right)^{2} = \alpha^{2}e^{-\varepsilon}dt$$

LV/SV Approximation

- Can use those to solve for the other four transition frequencies
- However: still one free parameter
  - The distance between the states parameter, ε
- Choose that to make the probability zero of jumping from Y=+1 state to Y=-1 state, or vice versa
  - Means no really big vol jumps
  - And gives a reasonable state spacing as well

LV/SV Approximation

- Choose e to be equal to

$$\varepsilon = 2\sinh^{-1}(\gamma)$$

$$\gamma = \frac{\alpha}{2\sqrt{\beta}}$$

LV/SV Approximation

- With this choice for ε

$$\lambda_{+0} = \lambda_{-0} = \beta$$

$$\lambda_{+-} = \lambda_{-+} = 0$$

$$\lambda_{0+} = \beta\left(\frac{1}{2} - \frac{\gamma}{2\sqrt{1+\gamma^2}}\right)$$

$$\lambda_{0-} = \beta\left(\frac{1}{2} + \frac{\gamma}{2\sqrt{1+\gamma^2}}\right)$$

LV/SV Approximation

- What is the initial value of Y?

- Could start with Y=0; that's like starting with a known value of v(0) in a Heston-like model

- Could also assume we don't know the value of Y
  - Assume it takes its stationary distribution across the three states
  - Better: gives a large short-expiration smile than a fixed initial value of Y
    - Matches market behavior better

LV/SV Approximation

- Stationary probabilities (where probability in matches probability out)

$$P_0 = \frac{1}{2}$$

$$P_+ = \frac{1}{4} - \frac{\gamma}{4\sqrt{1+\gamma^2}}$$

$$P_- = \frac{1}{4} + \frac{\gamma}{4\sqrt{1+\gamma^2}}$$

- Price of a derivative is price weighted across the three layers

$$V = V_0 P_0 + V_+ P_+ + V_- P_-$$

LV/SV Approximation

- Local volatility uses a parametric form

- Define a transformed spot variable

$$x(t) = \frac{\ln\left(\frac{S(t)}{F(0,t)}\right)}{\sigma_0 \sqrt{t}}$$

- Then mark local volatility for five points in x
  - x=-1.28, -0.68, 0, +0.68, +1.28 (roughly ATM, 25d, and 10d points)
  - σ_0 is the local volatility for x=0
  - Cubic spline between them
  - Same extrapolation as we used for implied volatilities before
    - Cubic spline extrapolation parameter determines how quickly local volatilities flatten out outside x=-1.28 and x=+1.28

LV/SV Approximation

- Pricing under the model can be done with backward induction
- Two operations for each step
- First, backward induct one time step for all three Y layers independently, solving the Y-specific PDE

$$\frac{\sigma^2(S,t)S^2e^Y}{2}\frac{\partial^2 p_Y}{\partial S^2}+\mu S\frac{\partial p_Y}{\partial S}+\frac{\partial p_Y}{\partial t}=r_d p_Y$$

LV/SV Approximation

- Then, mix between layers according to the probabilities of transitioning between the layers

$$p_0 \to (1-\beta dt)\,p_0 + \lambda_{0+}dt\,p_+ + \lambda_{0-}dt\,p_-$$

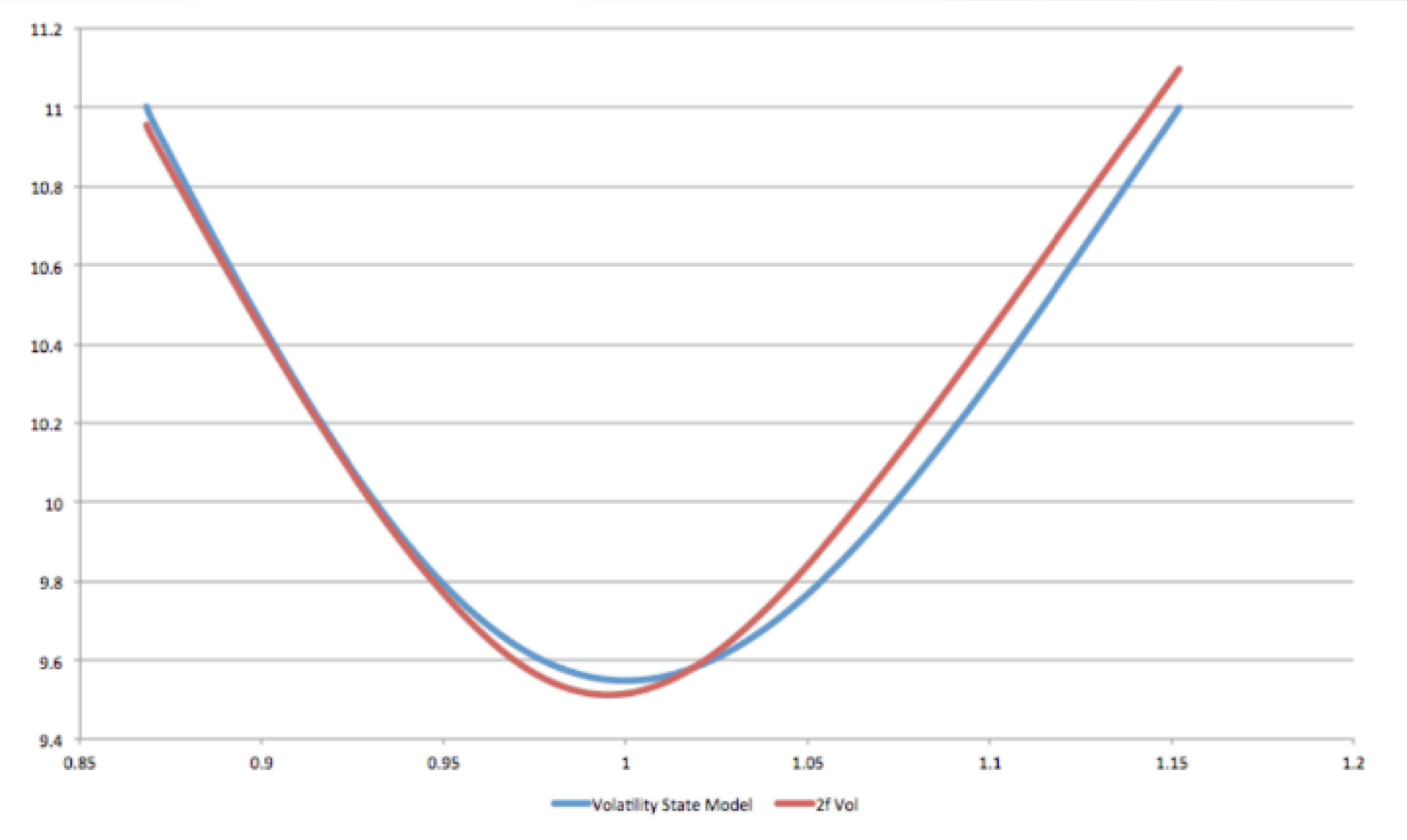
$$p_+ \to (1-\beta dt)\,p_+ + \beta dt\,p_0$$

$$p_- \to (1-\beta dt)\,p_- + \beta dt\,p_0$$

- Repeat this two-step process for each time step as you move backward from the derivative expiration to t=0 (today)

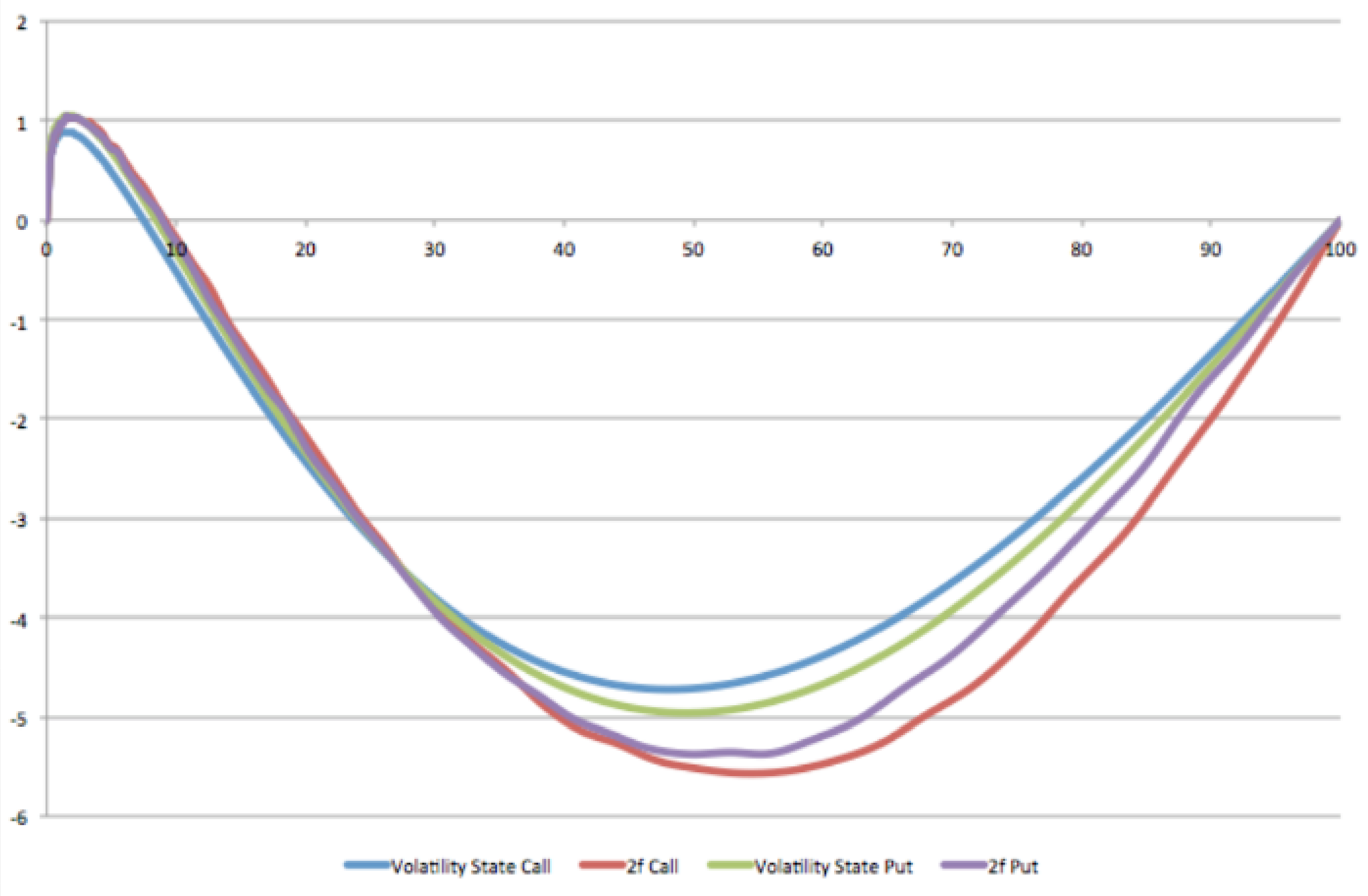
LV/SV Approximation

- Gives very similar (but not identical) implied volatilities as the full two-factor model where stochastic factor diffuses like a Heston process



LV/SV Approximation

- Gives very similar barrier option prices as well, as measured by one touch prices



LV/SV Approximation

- Calibration of the model means calibrating the local volatility parameters
  - Five local volatility parameters
  - Assumed to be piecewise-constant in time, between benchmark expiration dates
  - Bootstrap the parameters
    - Calculate the first set of five to match the five implied volatilities at the first expiration date
    - Then use those plus the five implied volatilities at the second expiration date to calibrate the second set of five local vols
    - … and so on

- Assumes the α and β parameters are marked a priori
  - These are "exotic parameters": not explicitly calibrated

LV/SV Approximation

- Numerical root-finding to solve for the local vol parameters

- Sped up by using “forward induction”
  - Lets you price vanilla options for many strikes in one pass
  - Analogous to pricing one vanilla option for many spots with backward induction

- This is a powerful technique to reduce computation time
  - … but we don’t have time to go through the details today
  - You’ll use it in the assignment

LV/SV Approximation

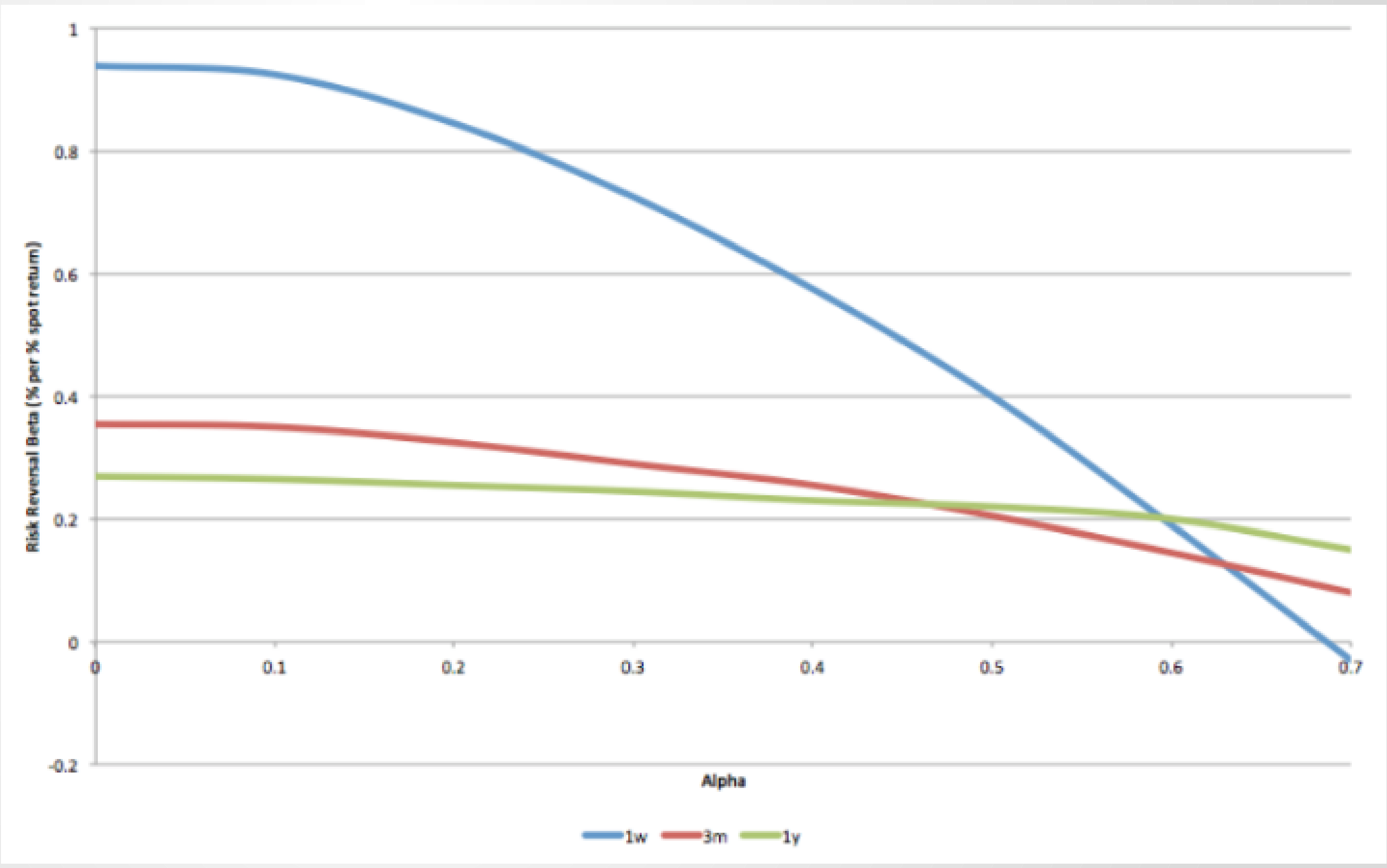
- With a calibrated model we can look at how barrier option prices are affected by the mixture parameter

- First, we should look at what the model predicts for risk reversal beta
  - Risk reversal beta is the most important dynamic for pricing barrier options

- Risk reversal beta in the model is determined by how risk reversal moves as spot moves and accesses different local vols
  - Still averaging across the three Y layers

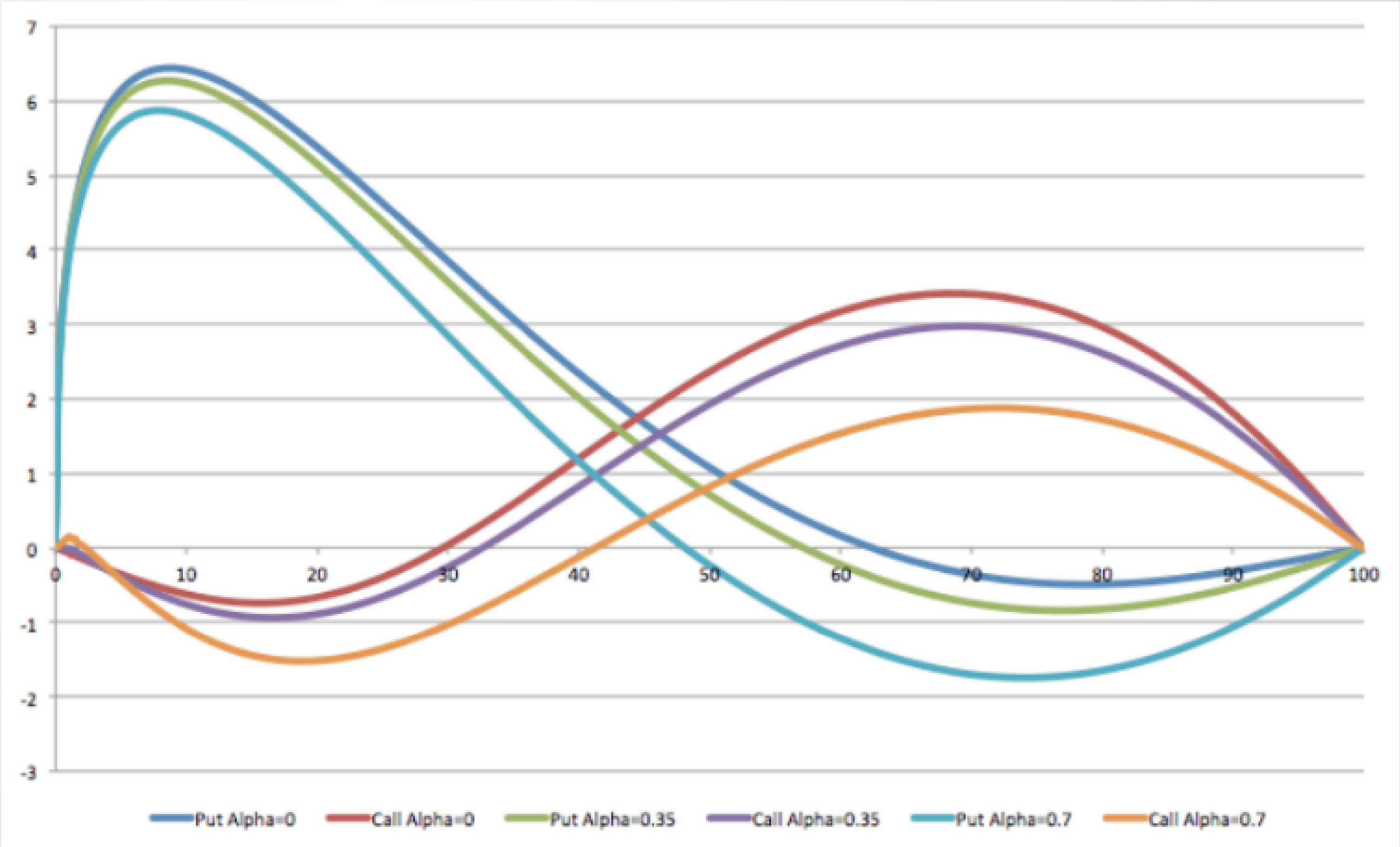
LV/SV Approximation

- Risk reversal beta for different expiration tenors, as a function of the volatility of volatility parameter α



LV/SV Approximation

- Can then look at impact of changing α (and recalibrating local vols) on one touch prices



Numerical Frameworks

- Pricing exotics under a model like this is computationally expensive
  - Generally want to delegate pricing to C++ for faster performance
  - Often an order of magnitude or two faster than in Python

- But you want to be able to play around with pricing for new exotic structures in Python
  - Want to be able to define contract terms in Python but delegate pricing to C++

- Right way: build a backward induction engine in C++ and expose it in Python

Numerical Frameworks

- Example: pricing under the LV/SV approximation model in the WST environment
  - Since I have it available!
- First step: create a backward induction engine

```
import wst.core.analytics.fd as fd

bi = fd.LVSBICN(spot,
                volsdd,volsd,vols0,volsu,volsuu,
                rds,rfs,break_times,
                alpha,beta,extrap_fact,
                nu,nt,nsd)
```

Numerical Frameworks

- Next step: construct a payoff and add it to the engine as a boundary condition
  - Example: a vanilla option payoff
  - The payoff defines a payoff in the spot direction only…
  - … and is applied onto the engine at some specific time
  - Can add in payoffs as backward induction proceeds

```
import wst.core.analytics.payoff as payoff

p = payoff.OptionPayoff(is_call,strike)
bi.initialize_payoff(p,texp)
```

Numerical Frameworks

- Variation: include a knockout barrier in the contract terms
  - For this we add a knockout first, which constrains the limits of the grid
  - Then we initialize the grid with the option payoff

```
import wst.core.analytics.payoff as payoff

p = payoff.OptionPayoff(is_call,strike)
bi.add_knockout(barrier,is_up)
bi.initialize_payoff(p,texp)
```

Numerical Frameworks

- Now the backward induction!
  - To get the price today, backward induct to t=0
  - But could backward induct to an intermediate time, add on a new payoff, and then go back further, etc
- After backward inducting, you get a price by asking the engine to interpolate a price at the spot level you want

```
bi.go_back(0)
price = bi.interp(spot)
```

Numerical Framework

- All the code together to price a knockout option
  - Not very much of it! It’s easy to price new structures

```
import wst.core.analytics.fd as fd

bi = fd.LVSBICN(spot,
                volsdd,volsd,vols0,volsu,volsuu,
                rds,rfs,break_times,
                alpha,beta,extrap_fact,
                nu,nt,nsd)


import wst.core.analytics.payoff as payoff

p = payoff.OptionPayoff(is_call,strike)
bi.add_knockout(barrier,is_up)
bi.initialize_payoff(p,texp)

bi.go_back(0)
price = bi.interp(spot)
```

Numerical Framework

- Can use forward induction to get call option prices for a range of strikes with one call to the forward inductor

```
import wst.core.analytics.fd as fd

fi = fd.LVSFICN(spot,
                volsdd,volsd,vols0,volsu,volsuu,
                rds,rfs,break_times,
                alpha,beta,extrap_fact,
                nu,nt,nsd,max_time)

fi.go_forward(texp)
prices = [fi.interp(strike) for strike in strikes]
```

Pricing with Copulas

- Consider a European “dual digital” option
  - Pays 1 unit of currency if $spot_1 > strike_1$ **and** $spot_2 > strike_2$ on some expiration date

- A regular European digital is a measure of the risk neutral probability that spot>strike

- A dual digital is a measure of the joint probability that both spots are above the two strikes
  - Depends on correlation
  - What does correlation mean outside Black-Scholes?

Pricing with Copulas

- Copulas let you piece together known marginal distributions
  - If I know vanilla prices in the two markets, I know the two marginal distributions
- The choice of copula defines the specific correlation structure that gets applied
  - Defines the joint distribution
- Many different kinds of copulas
  - … and many different kinds of correlation structure
  - We'll focus on just one: Gaussian copula

Pricing with Copulas

- Basic idea for a Gaussian copula
  - Translate each spot to a standard normal variable
  - State that the joint distribution of those two standard normal variables is bivariate standard normal with a fixed correlation

- Then can price any European payoff on the two spots
  - Payoff at expiration in terms of the two spots
  - Translate to a payoff in terms of the two standard normals
  - Integrate the payoff over the joint distribution

Pricing with Copulas

- First step: translate spot to a standard normal variable
  - Match them up via the cumulative distribution functions

$$\int_{x'=-\infty}^{x} f_n(x')\,dx' = \int_{S'=0}^{S} f_S(S')\,dS'$$

$$N(x) = F(S)$$

$$x = N^{-1}(F(S)), \quad S = F^{-1}(N(x))$$

- $f_n(x)$ = standard normal PDF, N(X) = standard normal CDF
- $f_S(S)$ = risk neutral PDF of spot, F(S) = CDF of spot
  - F(S) is the (undiscounted) digital put price

Pricing with Copulas

- Now I can price any payoff on two spots by asserting a bivariate normal distribution with a fixed correlation

$$v = \int_{x_1=-\infty}^{\infty} \int_{x_2=-\infty}^{\infty} P\left(S_1(x_1), S_2(x_2) \right) f_{12}(x_1, x_2) \, dx_1 \, dx_2$$

- Here, $P(S_1,S_2)$ is the payoff in terms of the two spots, which we write as functions of the two standard normal variables x_1 and x_2
- $f_{12}(x_1,x_2)$ is the bivariate standard normal distribution function

Pricing with Copulas

- The joint distribution is defined by a correlation parameter ρ

$$f_{12}(x_1, x_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{(x_1^2 + x_2^2 - 2\rho x_1 x_2)}{2(1-\rho^2)}}$$

Pricing with Copulas

- European joint digital price is pretty simple
  - Find $x_{1K}(S_1=K_1)$ and $x_{2K}(S_2=K_2)$ from the maps

$$v = \int_{x_1=x_{1K}}^{\infty} \int_{x_2=x_{2K}}^{\infty} f_{12}(x_1, x_2)\, dx_1\, dx_2$$

- Just the bivariate standard cumulative normal distribution function
  - No closed form, but efficient closed-form approximations

Risk with Copulas

- There is only one parameter that defines correlation structure in a Gaussian copula
  - The correlation parameter r

- We can calibrate to ATM volatility of the cross pair
  - eg asset 1 = EURUSD, asset 2 = GBPUSD
  - Cross spot is then EURGBP and we see those volatilities
  - Use the copula model to price the ATM EURGBP option and set ρ so that the model reproduces its price

Risk with Copulas

- The model then is calibrated only to EURGBP ATM volatility, not to its RR/BF levels

- In general a Gaussian copula will not hit the cross RR/BF
  - Tends to underestimate cross BF in particular

- One way to think about this: correlation is not really constant
  - It is stochastic, and if a derivative has non-linear exposure to a stochastic asset, its price should incorporate value from that gamma
  - If it is correlated with the spots, there is cross gamma to price as well

Variance Swaps

- A variance swap contract pays out against realized volatility squared

$$\sigma^2 = \frac{N_d}{N}\sum_{i=1}^{N}\ln^2\left(\frac{S_i}{S_{i-1}}\right)$$

- σ^2 is the realized vol squared swapped against a fixed strike
- N_d is the number of trading days/year: specified in contract
- N is the number of daily spot returns in the contract period
- S_i is the spot fixing for fixing date i

Variance Swaps

- Variance swaps with continuous fixings, on an asset with no jumps, can be replicated by a vanilla portfolio

$$\sigma^2 = \frac{2}{T}\int_0^\infty \frac{v(K)}{K^2}dK$$

- σ^2 here represents the fair strike for a variance swap settling time T in the future
- v(K) is a call option price for K>forward, put otherwise

Volatility Swaps

- Volatility swaps pay off against realized volatility
  - Not volatility squared, like a variance swap

$$\sigma = \sqrt{\frac{N_d}{N}\sum_{i=1}^{N}\ln^2\left(\frac{S_i}{S_{i-1}}\right)}$$

- You can think about pricing these as a square-root payoff on an asset that is the variance swap
  - The average of that asset is the fair strike for the variance swap
  - Need to model volatility of variance swap fair strikes

Volatility Swaps

- Hedging strategy for a volatility swap:
- Buy the volatility swap, a contract with a square root payoff against the "asset", the variance swap
- Sell an appropriate amount of the variance swap against it
  - Notional 1/2/sqrt(var swap fair strike), from derivative of square root
- When the market moves and the variance fair strike moves, the variance swap notional needs to be adjusted
  - Negative gamma to moves in the variance swap fair strike

Volatility Fair Strike

- Because of the negative convexity, the volatility swap fair strike is **less than** the square root of the variance swap fair strike
  - Buy vol swap, dynamically hedge with variance swap, lose money due to negative gamma

- The spread there is a function of the realized volatility of the variance swap fair strike
  - Pretty close to the realized volatility of ATM volatility

- This is the dynamic that you need to model for vol swaps
  - Very different to what barrier derivatives care about!