Markov Switching Variance Gamma

Kateryna Chevplianska and Rodrigo Salas

The paper we chose to implement is "Option Pricing Using Variance Gamma Markov Chains" by Konikov and Madan

FQE
Financial Quants and Engineers Club

In the early 2000s, stochastic volatility was expensive and slow to calibrate.

Traders needed models that:
fit volatility smiles,
were fast to calibrate,
and could reproduce fat tails and jumps.

Variance Gamma (VG) became popular after Madan–Carr–Chang (1998).

Konikov–Madan (2002) introduced the idea of switching between two VG regimes, capturing volatility clustering.

What the Variance Gamma Model Is

The VG process:

$$X_t = \theta G_t + \sigma W_{G_t}$$

Where:

- $G_t \sim \Gamma(t; 1/\nu, \nu)$ is gamma time change,
- ν — variance of the subordinator
- θ — drift of the subordinator
- σ — Brownian volatility inside the gamma clock.

Characteristic function:

$$\phi_{VG}(u) = \left(1 - i\theta\nu u + \frac{1}{2}\sigma^2 \nu u^2\right)^{-t/\nu}$$

FQE
Financial Quants and Engineers Club

Problem: Homogeneous Lévy Fails Across Maturities

- For single state Lévy: variance is proportional to t, skewness ~ $1/\sqrt{t}$, kurtosis ~ $1/t$
- which means as maturity gets larger the variance grows but skewness and kurtosis shrink which makes the distribution more and more normal over time
- Empirical data shows that they stay high or even RISING vol, skewness, kurtosis with maturity
- Need regime-switching to capture volatility clustering
- Solution: Two-state Markov chain between VG processes

Two-State Markov Variance Gamma Model

FQE
Financial Quants and Engineers Club

- Process switches between two VG states via Markov chain
- State 0 (Calm): VG with params (σ_0, ν_0, θ_0)
- State 1 (Turbulent): VG with params (σ_1, ν_1, θ_1)
- The Markov chain has two diff transition rates: λ_{01} (Calm→Turbulent), λ_{10} (Turbulent→Calm), and 1 probability for the current state
- 9 parameters total: enough to capture vol clustering and the term structure of the skewness and kurtosis that we see in the data

What is MSVG?

- Two sets of VG parameters:

  $(\theta_1, \sigma_1, \nu_1)$

  $(\theta_2, \sigma_2, \nu_2)$

- Hidden 2-state Markov chain with transition matrix:

$$P = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}$$

Interpretation:

- State 1: calm market
- State 2: stressed market

Characteristic Function for MSVG

FQE
Financial Quants and Engineers Club

This recursion is the heart of the model. For each regime, multiply the VG characteristic function by the previous CF, and weigh it by transition probabilities. This gives us the full CF of the Markov-switching process.

If the regime at step n is i, the characteristic function evolves as:

$$\phi_{n+1}(u) = \left[\phi_1^{VG}(u)\phi_n(u)\right] p_{i1} + \left[\phi_2^{VG}(u)\phi_n(u)\right] p_{i2}$$

You can think of CF as a convient way to summarize the whole distribution that we'll use later for option pricing

MVG Characteristic Function (Proposition 2)

Here we just solve that recursion from the privious slide:

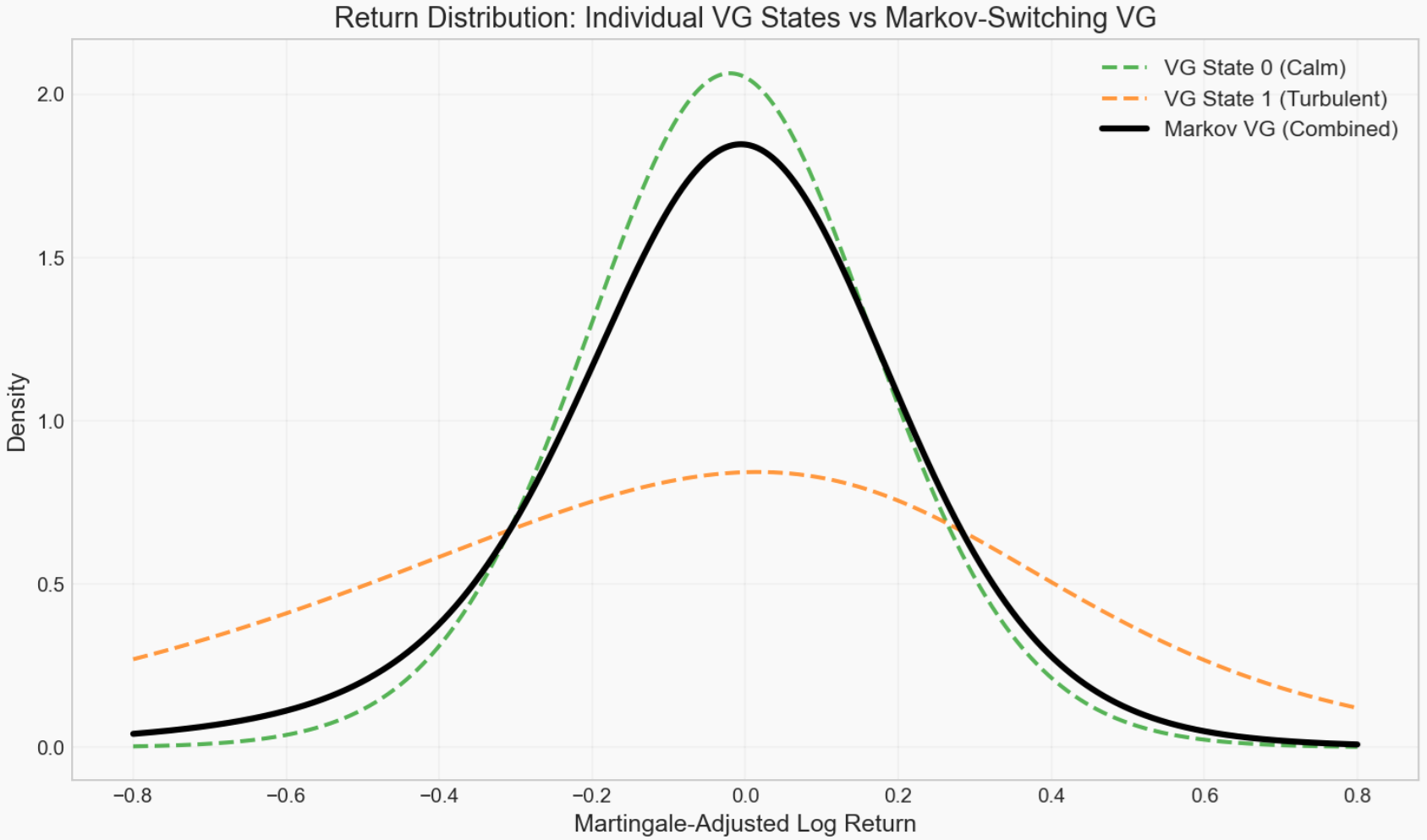
$$\phi\_X(t)(u) = \phi_0(u)^t \cdot g(\log(\phi_0(u)/\phi_1(u)))$$

$$g(\lambda) = p \cdot g_0(\lambda) + (1-p) \cdot g_1(\lambda)$$

g_0, g_1 from Riccati equation (simple nonlinear differential equation) with roots η_1, η_2

Key: Laplace transform (continuous time generating function) of time spent in state 1

Return Distribution: VG States vs Markov-Switching



Implementation: msvg Package

vg_process.py

- VGParams class
- Characteristic exponent ψ(u)
- Numba-accelerated simulation

markov_chain.py

- MVG characteristic function
- Parallel Monte Carlo kernel
- Riccati solution for g_0, g_1

option_pricing.py

- FFT call pricing (Carr-Madan)
- PDF recovery from CF

Calibration.py

- Sum of squared errors
- Robust bounds handling

*We price options in two ways — **FFT for speed, Monte Carlo for verification**. The FFT method is extremely fast because both the VG and MSVG CF's are closed-form.*

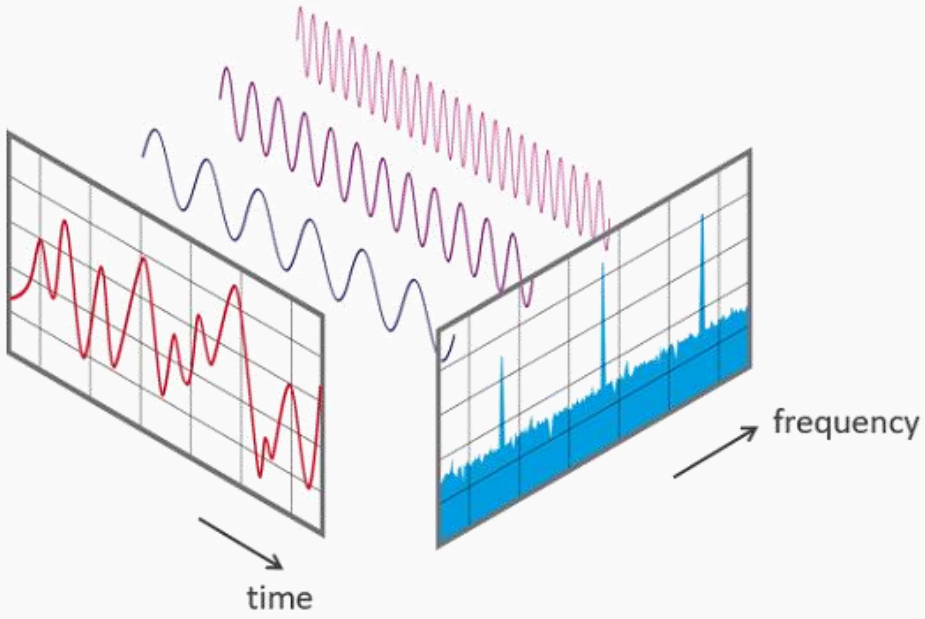
FFT (Fast Fourier Transform) Option Pricing (Carr-Madan Method)

FQE
Financial Quants and Engineers Club

Carr–Madan says: “If you know the CF, you can price a whole grid of call options very fast using an FFT" So instead of doing an integral separately for every strike, we:

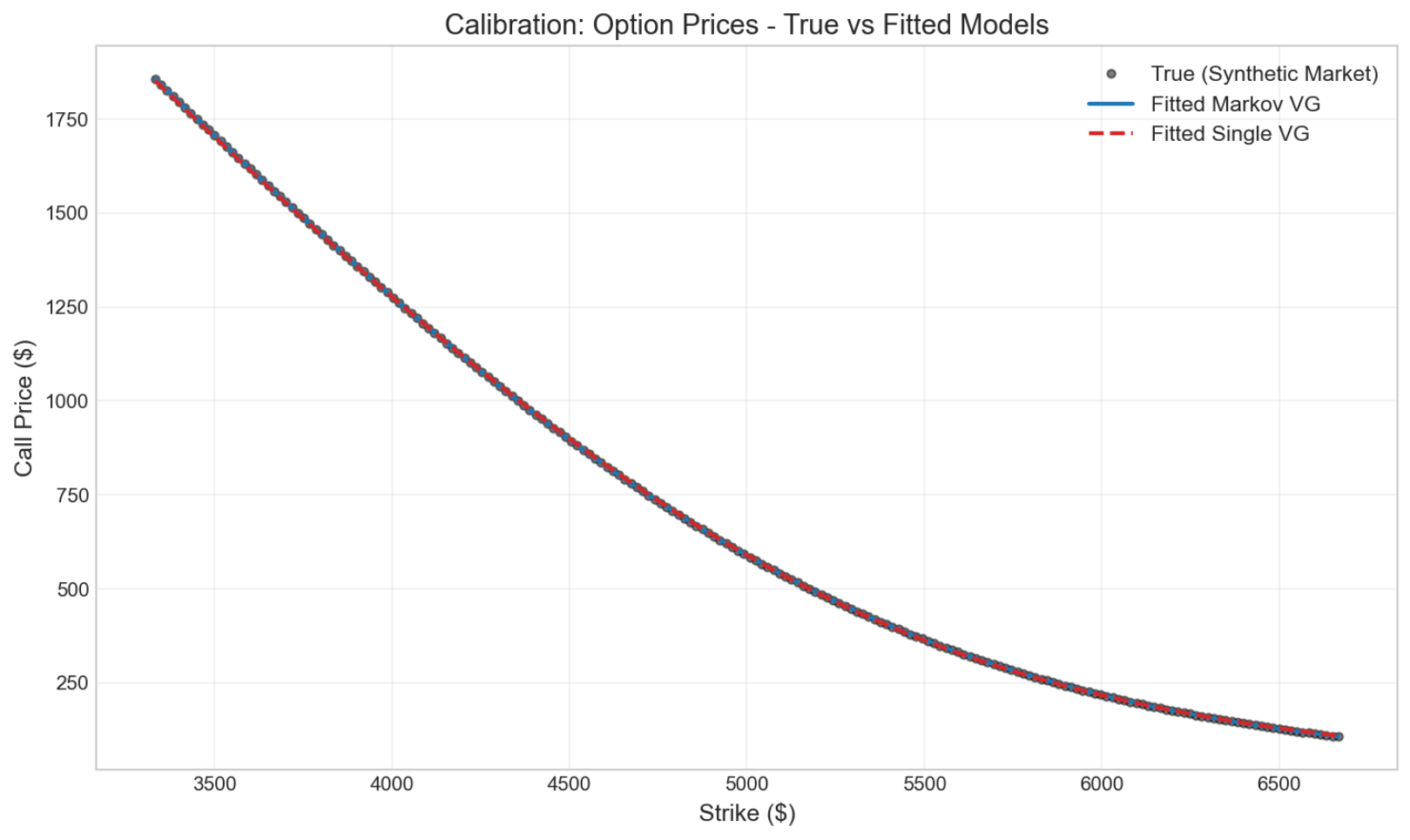
1. Transform the call price into the Fourier / frequency domain,
2. Do one FFT inversion,
3. Then interpolate to get prices at any strike we want.

$$
\begin{aligned}
\psi_T(v) &= \int_{-\infty}^{\infty} \mathrm{e}^{\mathrm{i}vk} \int_{k}^{\infty} \mathrm{e}^{\alpha k}\mathrm{e}^{-rT}(\mathrm{e}^{s}-\mathrm{e}^{k})q_T(s)\,\mathrm{d}s\,\mathrm{d}k \\
&= \int_{-\infty}^{\infty} \mathrm{e}^{-rT}q_T(s)\int_{-\infty}^{s}(\mathrm{e}^{s+\alpha k}-\mathrm{e}^{(1+\alpha)k})\mathrm{e}^{\mathrm{i}vk}\,\mathrm{d}k\,\mathrm{d}s \\
&= \int_{-\infty}^{\infty} \mathrm{e}^{-rT}q_T(s)\left(\frac{\mathrm{e}^{(\alpha+1+\mathrm{i}v)s}}{\alpha+\mathrm{i}v}-\frac{\mathrm{e}^{(\alpha+1+\mathrm{i}v)s}}{\alpha+1+\mathrm{i}v}\right)\mathrm{d}s \\
&= \frac{\mathrm{e}^{-rT}\phi_T\big(v-(\alpha+1)\mathrm{i}\big)}{\alpha^2+\alpha-v^2+\mathrm{i}(2\alpha+1)v}.
\end{aligned}
$$

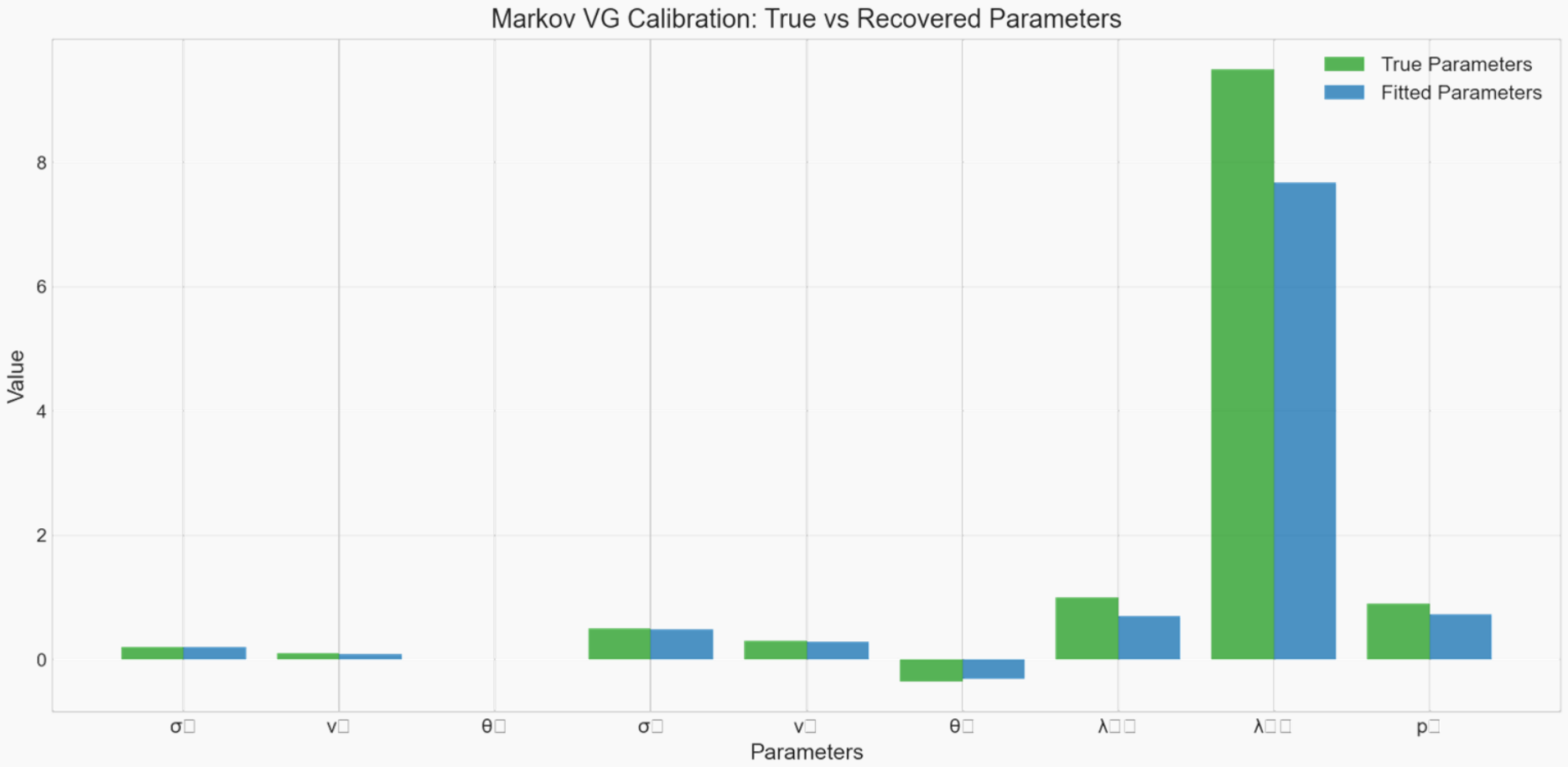


Calibration: Synthetic Market Prices

FQE
Financial Quants and Engineers Club

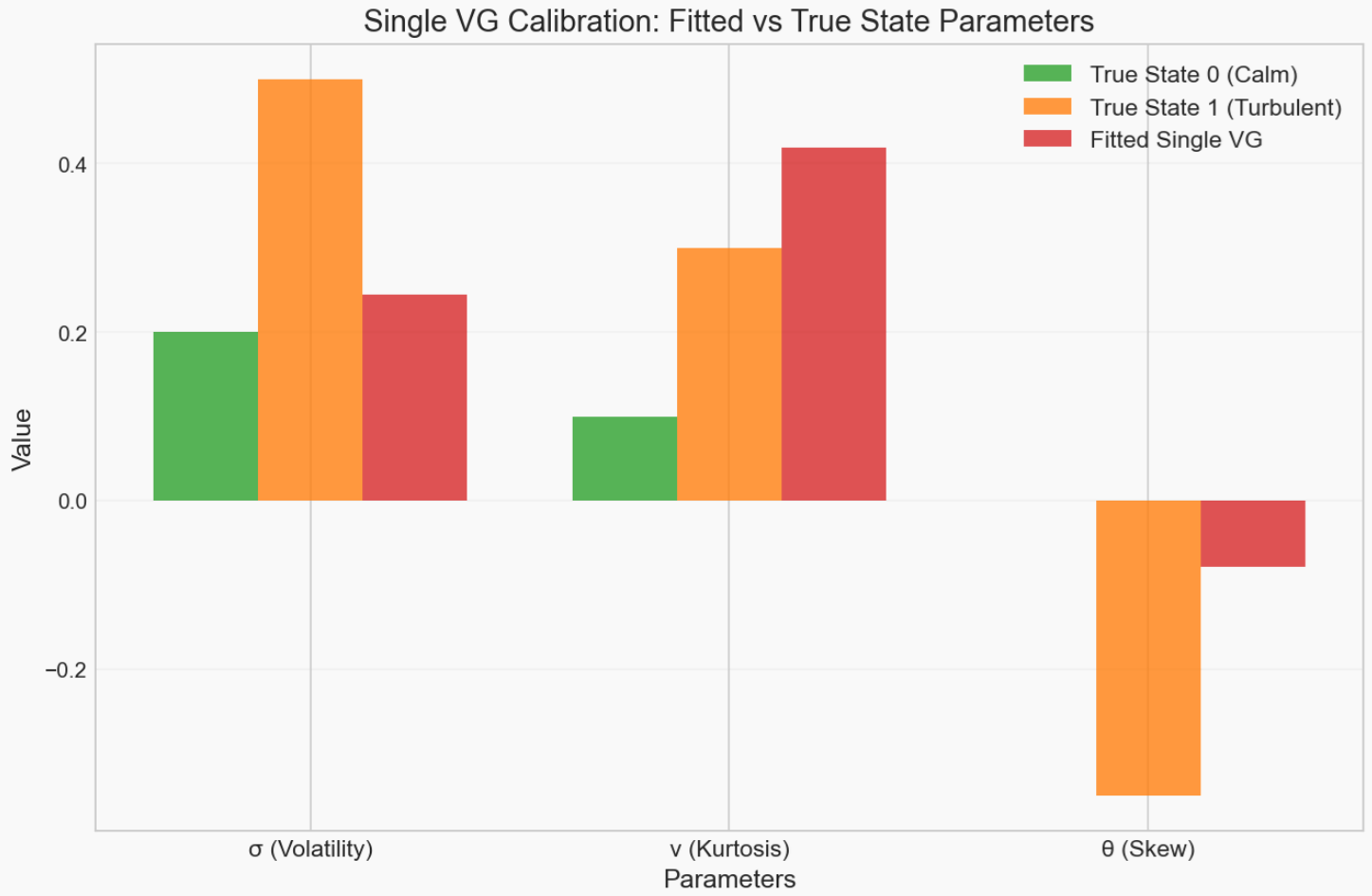


Parameter Recovery: Markov VG



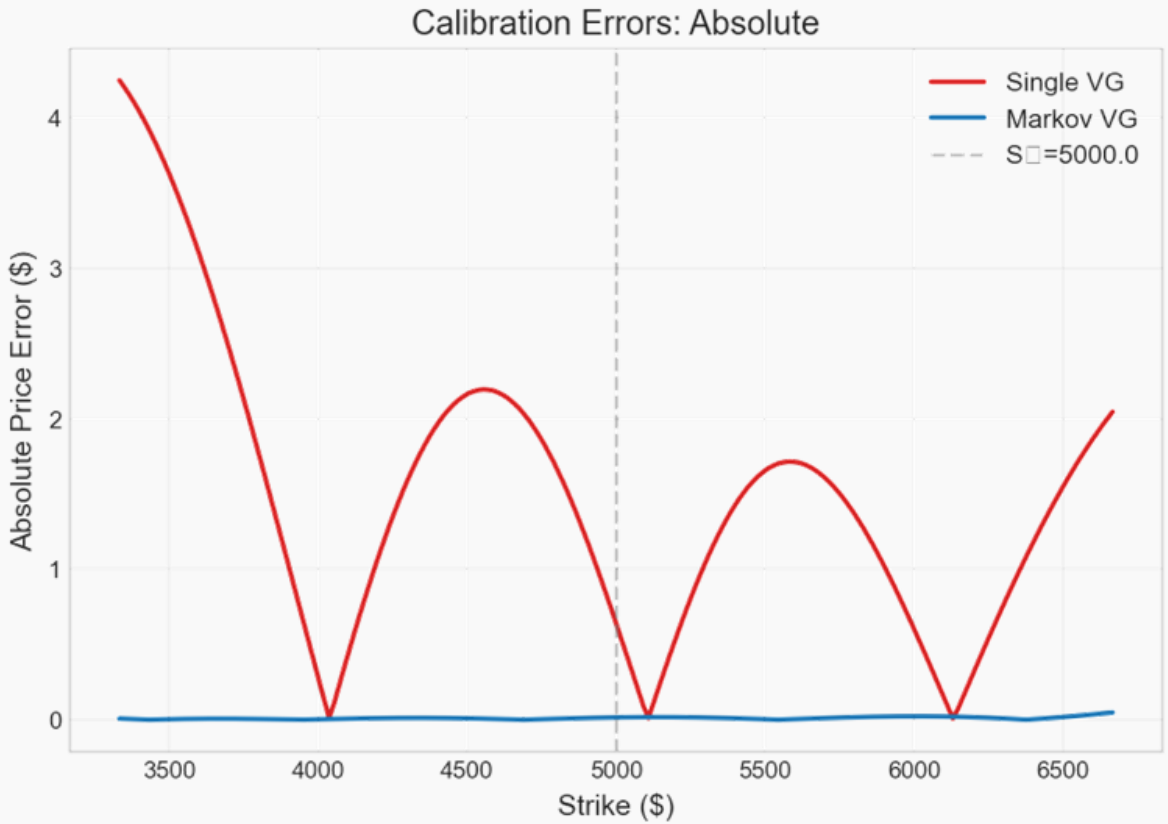
Single VG: Averages Between Regimes

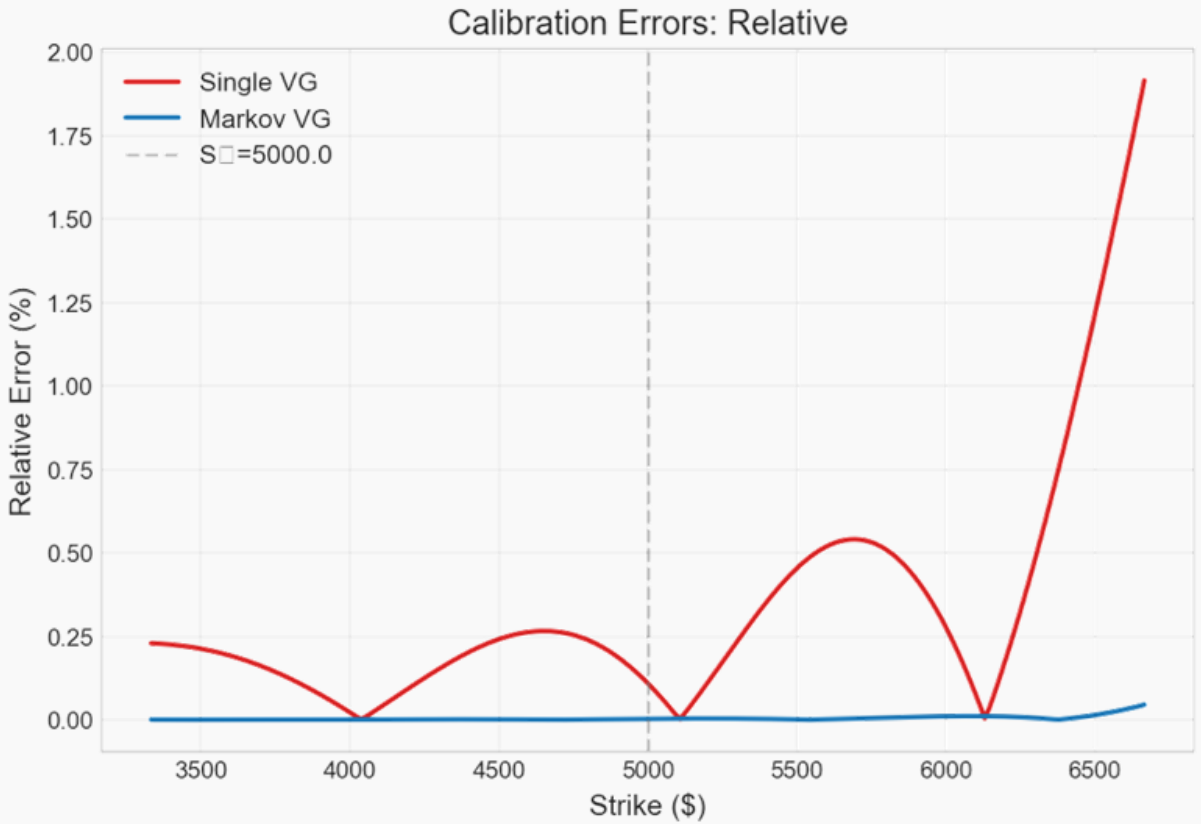
FQE
Financial Quants and Engineers Club



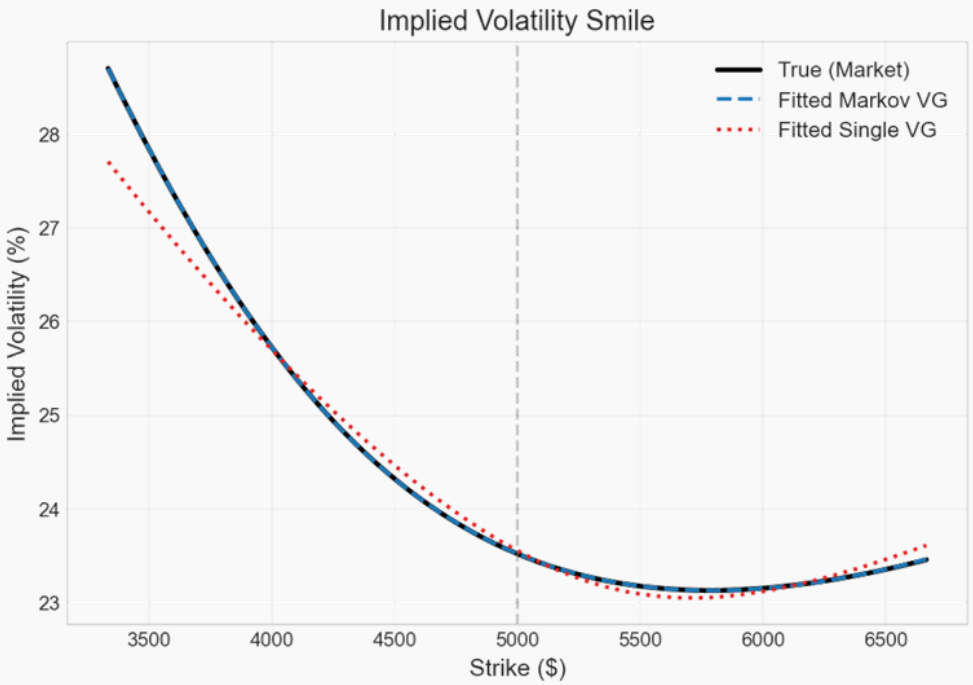
Calibration Errors: MVG vs Single VG

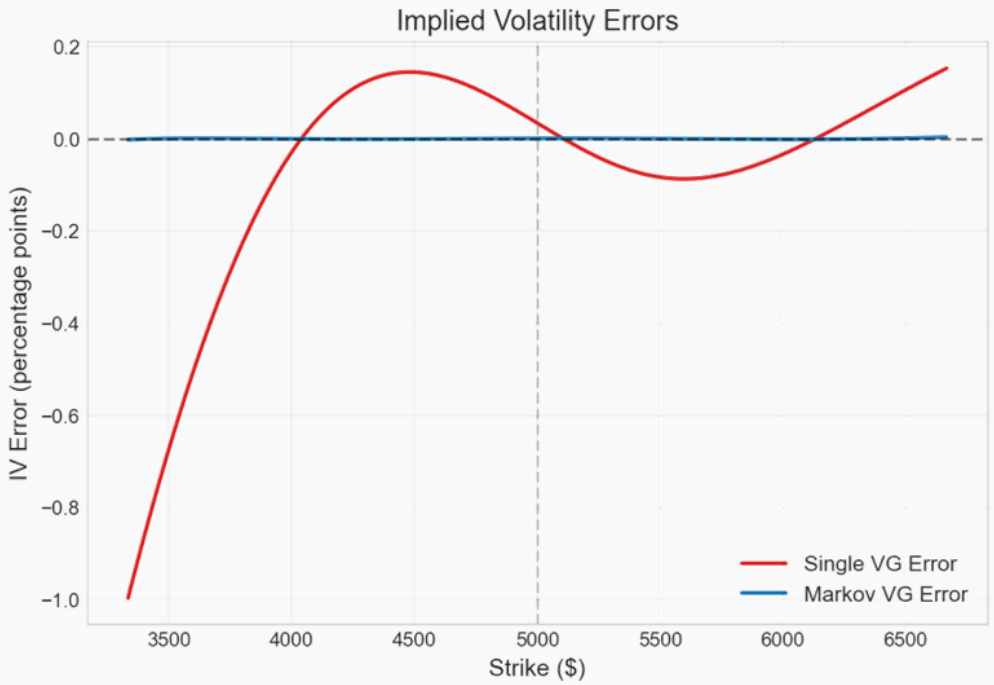
FQE
Financial Quants and Engineers Club





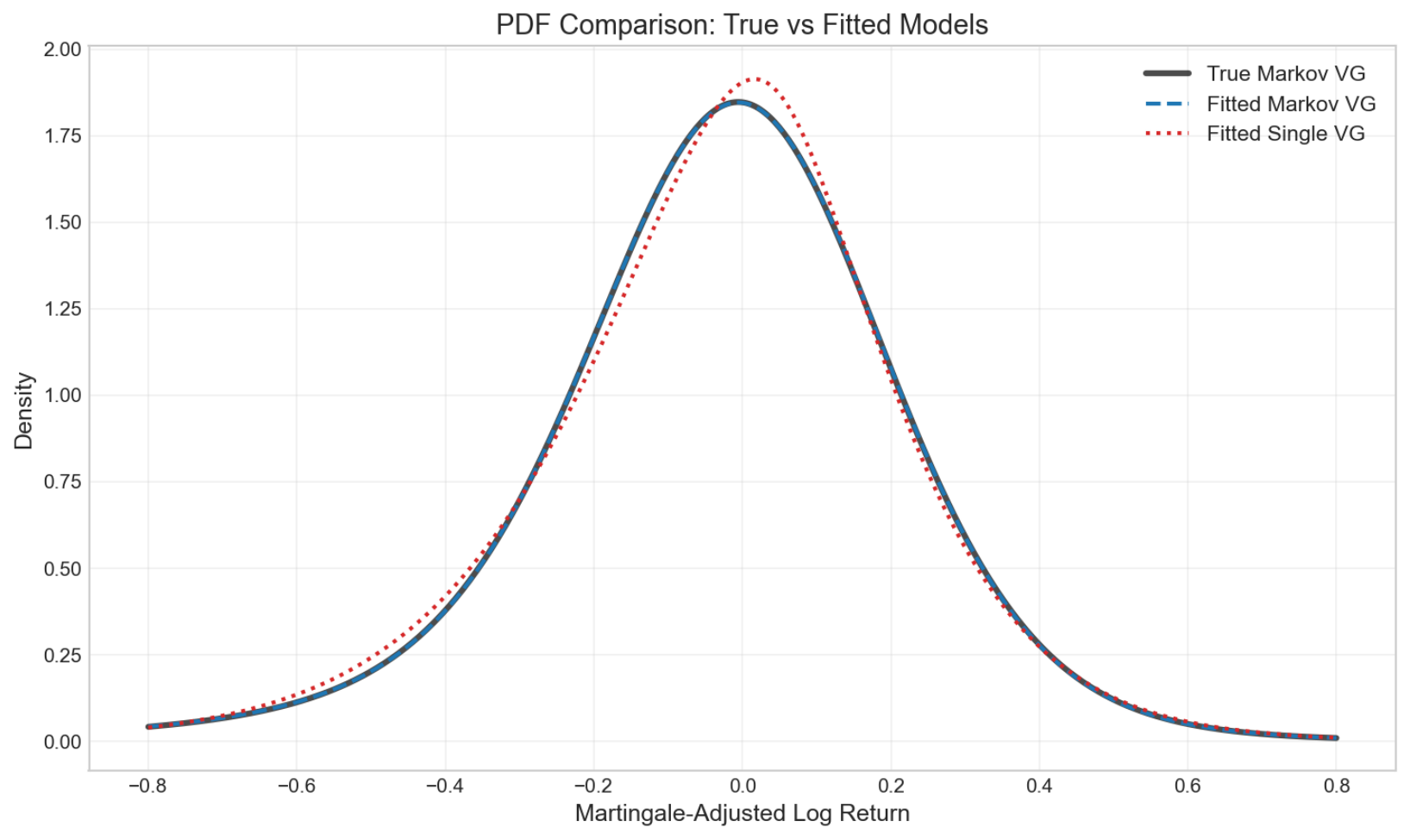
Implied Volatility: Theory Validation





Markov VG captures the full IV smile; Single VG misses the wings

PDF Comparison: True vs Fitted Models



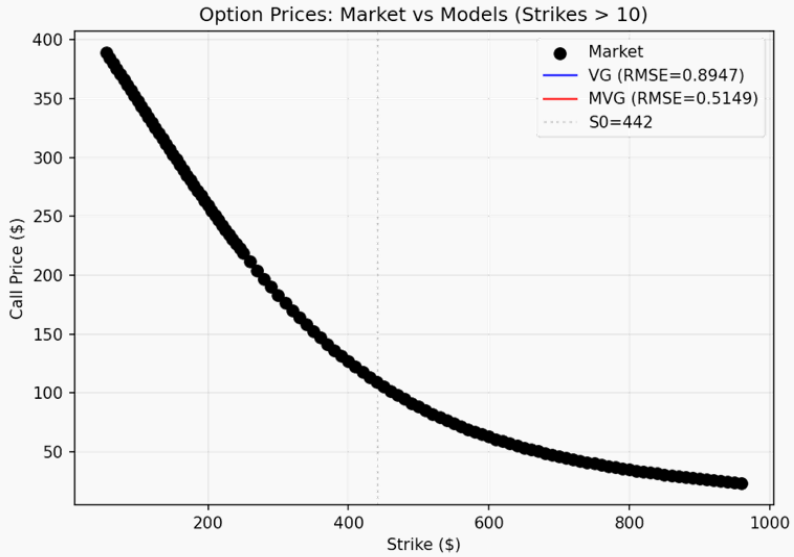
Markov VG matches the true distribution; Single VG underestimates tails

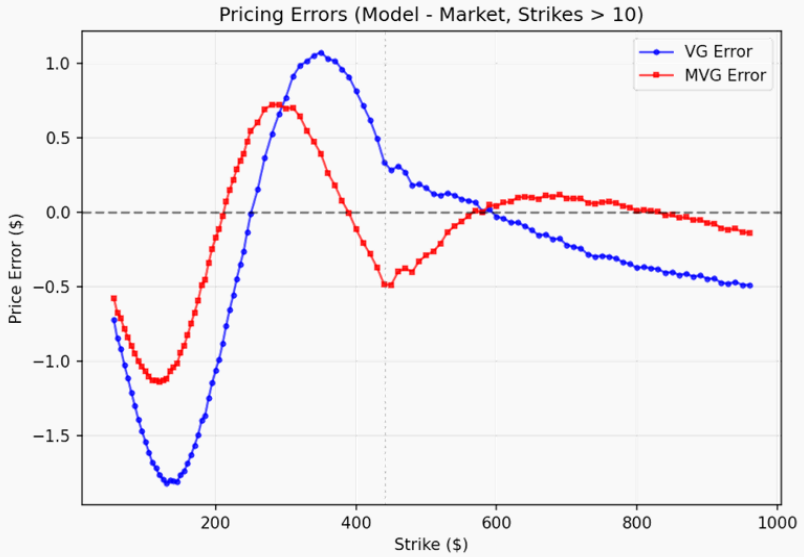
FQE
Financial Quants and Engineers Club

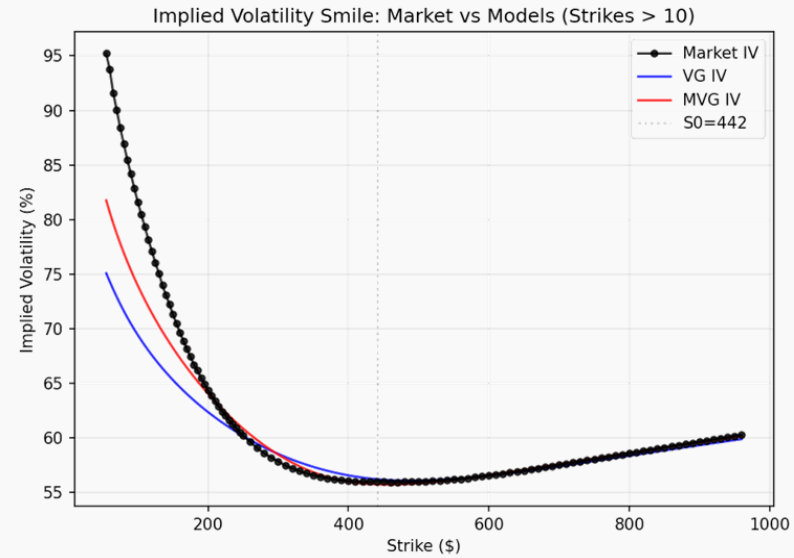
Real-World Application

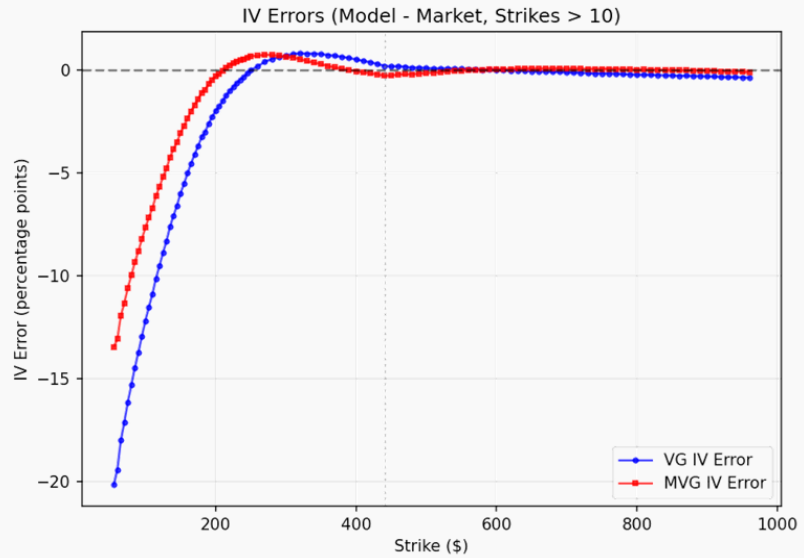
Calibration to TSLA Options Data

TSLA Results: VG vs MVG Comparison

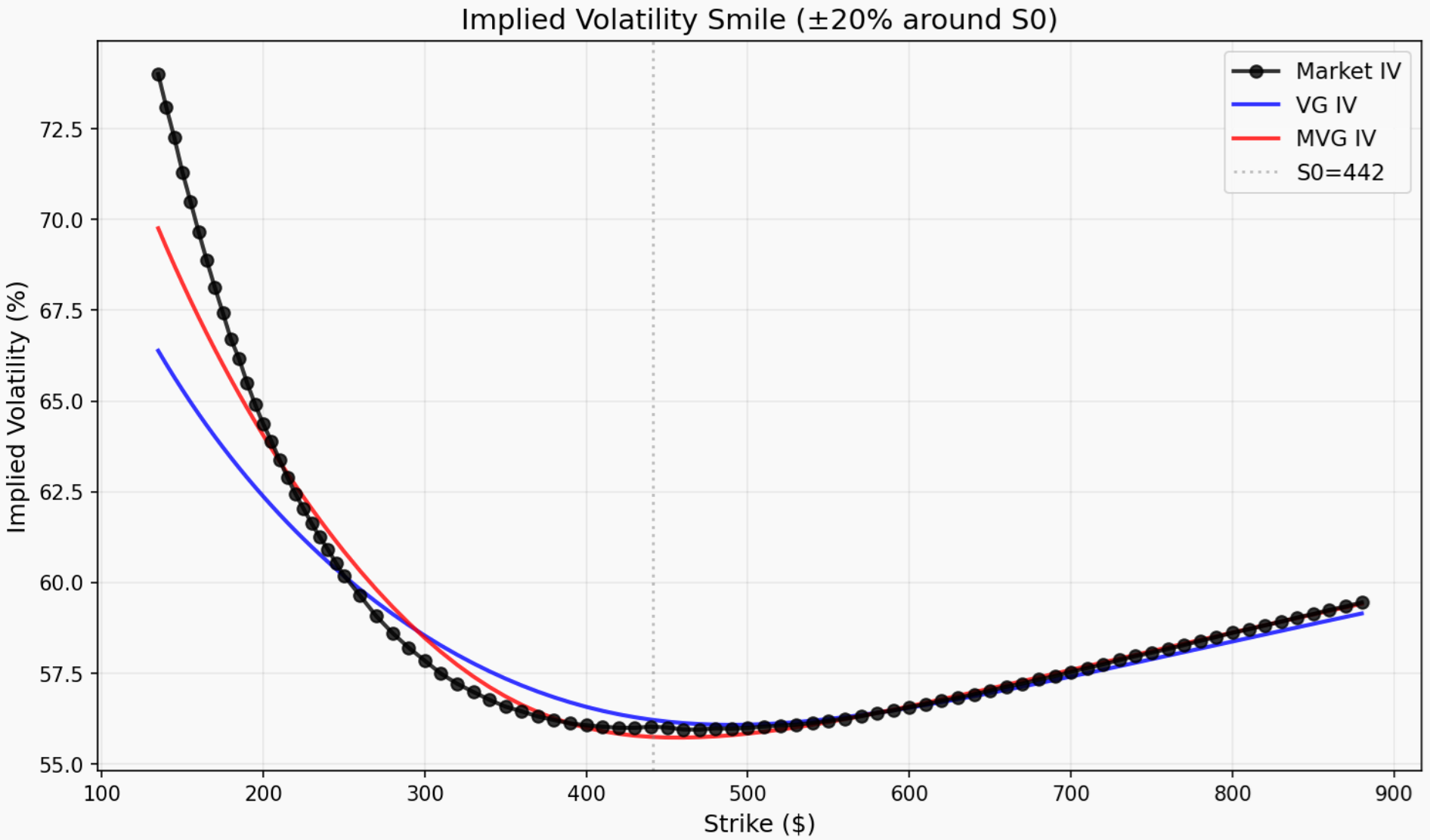








TSLA: Implied Volatility Smile (±20% ATM)



FQE
Financial Quants and Engineers Club

Thank you

Do you have any questions?