Monte Carlo and Crank–Nicolson Option Pricing

Numerical Comparison and Bloomberg SPX Chain Application

Hassan Ahmed

February 24, 2026

Contents

1 Introduction

1.1 Preface

This project is an adapted version of a project I did at King's College London for my Financial Mathematics MSc.

The computational techniques and mathematical methodologies incorporated were described and adapted from the lecture notes, specifically the 'Finite Difference Methods' and 'Monte Carlo Pricing' lectures by John Armstrong [1].

The work has two connected parts:

- **Part A (toy model):** a CEV-type diffusion with time-dependent volatility. This is used to compare Monte Carlo (Euler–Maruyama) against Crank–Nicolson finite differences for pricing and Greeks, including convergence behaviour.
- **Part B (market application):** the same numerical toolkit is then used as a small "pricing and risk engine" on a Bloomberg SPX options snapshot/chain. We enrich the chain with model prices (Black–Scholes baseline, Monte Carlo under GBM, and Crank–Nicolson under the BS PDE), Greeks, and mispricing summaries, and we visualise implied volatility structure and risk profiles across strikes and expiries.

1.2 Problem Setup (Part A)

It is given that a stock price S_t evolves according to

$$dS_t = \mu S_t\,dt + \sigma(t) S_t^{\gamma}\,dW_t.$$

However, for pricing we work under the risk-neutral measure $\mathbb{Q}$, where the drift becomes the risk-free rate r (under standard no-arbitrage assumptions). Hence the dynamics we use is:

$$dS_t = rS_t\,dt + \sigma(t) S_t^{\gamma}\,dW_t,$$

where W_t denotes a $\mathbb{Q}$-Brownian motion.

The parameters are $\gamma = 0.95$, $\sigma(t) = 0.18 + 0.07 \cdot \frac{t}{T}$, $K = 140$, $r = 0.02$, $S_0 = 140$, and $T = 1.40$.

The goal in Part A is to compute the European put price $V(S,t)$ and delta Δ under the risk-neutral dynamics, then compare two numerical approaches:

- Monte Carlo simulation using Euler–Maruyama,

- Crank–Nicolson finite differences applied to the pricing PDE.

To benchmark the outputs, we also compute an *estimated* Black–Scholes value using an effective volatility σ_{eff}, since the true model here is not log-normal.

In Part B we move from the toy model to real market inputs: a Bloomberg SPX options chain provides strikes, expiries, quotes and implied volatilities. Under the standard log-normal pricing framework (GBM with dividend yield), we generate model prices and Greeks at scale and summarise mispricings against market mids.

2 Hedging Argument for the Pricing PDE

We keep this derivation brief, but still include the key steps. Consider the option value $V(S,t)$ and form a self-financing hedged portfolio by shorting one option and holding Δ shares:

$$\Pi(t) = -V(S_t,t) + \Delta_t S_t.$$

The portfolio change is

$$d\Pi = -dV + \Delta\, dS.$$

Using Itô's lemma on $V(S_t,t)$ under the risk-neutral dynamics:

$$dV = V_t\, dt + V_S\, dS + \frac{1}{2} V_{SS}\, (dS)^2.$$

Since $dS = rS\, dt + \sigma(t) S^{\gamma} dW$, we have $(dS)^2 = \sigma^2(t) S^{2\gamma} dt$ (and the mixed terms vanish). Substituting gives:

$$dV = \left(V_t + rSV_S + \frac{1}{2}\sigma^2(t) S^{2\gamma} V_{SS} \right) dt + \sigma(t) S^{\gamma} V_S\, dW.$$

Now plug this into the portfolio dynamics. If we choose the hedge ratio $\Delta = V_S$, the dW terms cancel (this is the point of delta hedging), and the portfolio becomes locally risk-free:

$$d\Pi = -\left(V_t + \frac{1}{2}\sigma^2(t) S^{2\gamma} V_{SS} \right) dt.$$

Under no-arbitrage in the risk-neutral world, a risk-free portfolio must earn the risk-free rate:

$$d\Pi = r\Pi\, dt = r(-V + SV_S)\, dt.$$

Equating these and rearranging yields the pricing PDE:

$$V_t + rSV_S + \frac{1}{2}\sigma^2(t) S^{2\gamma} V_{SS} - rV = 0,$$

with terminal condition for a European put:

$$V(S,T) = (K - S)^+.$$

This PDE is what we discretise in the Crank–Nicolson method, and it also links to Monte Carlo via Feynman–Kac.

For the market-data application (Part B), the same hedging logic produces the standard Black–Scholes PDE under GBM, with the minor modification that a continuous dividend yield q enters the drift and boundary behaviour. In that setting we use Black–Scholes as a baseline model, Monte Carlo under GBM as a probabilistic cross-check, and Crank–Nicolson as the PDE-based solver.

3 Monte Carlo Simulation Approach

3.1 Methodology

By Feynman–Kac, the put price can be written as a discounted expectation:

$$V(S,t) = \mathbb{E}^{\mathbb{Q}}\Big[e^{-r(T-t)}(K - S_T)^+ \mid S_t = S\Big].$$

Monte Carlo approximates this expectation by simulating many paths of S_t and averaging the discounted payoff.

The justification for this approach comes from the Law of Large Numbers. The discounted payoff $e^{-rT}(K - S_T)^+$ has finite expectation under the risk-neutral measure, and so the sample average of independent simulations converges almost surely to the true expectation as $N \to \infty$. Moreover, by the Central Limit Theorem, the estimator is asymptotically normal with variance proportional to $1/N$, which implies that the statistical error decays at rate $\mathcal{O}(N^{-1/2})$.

To simulate paths under the risk-neutral SDE, we discretise using Euler–Maruyama with $\Delta t = T/n$:

$$S_{i+1} = S_i + rS_i\,\Delta t + \sigma(t_i)\,S_i^{\gamma}\sqrt{\Delta t}\,\varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0,1).$$

In the computations we used $n = 100$ time steps and $N = 10{,}000$ simulated paths.

3.2 Computing Price & Confidence Interval

The Monte Carlo estimator is:

$$\hat{V} = e^{-rT}\frac{1}{N}\sum_{j=1}^{N}(K - S_T^{(j)})^+.$$

A 95% confidence interval is:

$$\hat{V} \pm z_{0.975}\frac{\hat{s}}{\sqrt{N}},$$

where $\hat{s}$ is the sample standard deviation of discounted payoffs.

Obtained Monte Carlo Put Price: 9.0577
95% confidence interval for the price: [8.7989, 9.3165]

Figure 1 is included as a *visual sanity check* on the simulated underlying dynamics: the fan-shaped widening of the band over time is consistent with accumulating diffusion uncertainty. This plot is illustrative (it does not validate pricing accuracy), but it helps interpret why Monte Carlo estimates carry sampling error even when the implementation is correct.

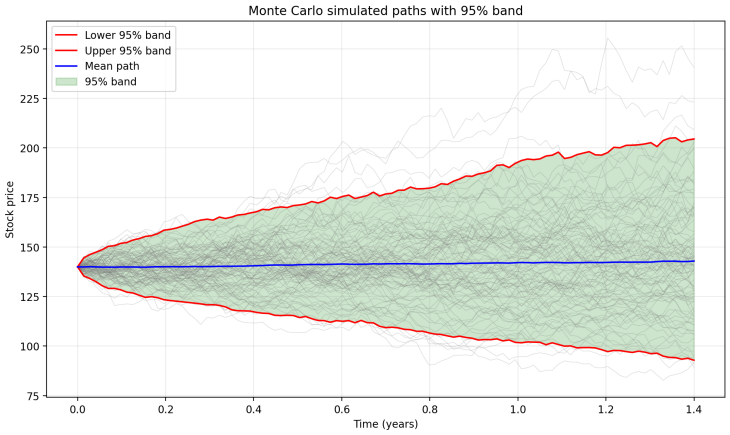

Figure 1: Monte Carlo simulated stock paths with 95% band and mean path.

3.3 Computing Delta & Confidence Interval

Delta is defined as $\Delta = \frac{\partial V}{\partial S}$. We approximate it using a central difference:

$$\Delta \approx \frac{V(S_0 + \epsilon) - V(S_0 - \epsilon)}{2\epsilon},$$

with $\epsilon = 0.01 S_0$. To reduce variance, we used common random numbers when estimating $V(S_0 + \epsilon)$ and $V(S_0 - \epsilon)$.

The delta obtained via Monte Carlo: -0.4009
Its 95% confidence interval: [-0.4091, -0.3927]

3.4 Computing Greeks (brief)

Using finite-difference bumps (as implemented in the code), we obtained:

$$\Gamma \approx 0.0141, \quad \text{Vega} \approx 49.2514, \quad \Theta \approx -3.8189, \quad \rho \approx -91.0930.$$

A small interpretation note: these are derivatives with respect to *decimal* parameters. For example, $\rho \approx -91.09$ is per $+1.00$ change in r, so a $+1\%$ (0.01) change corresponds to roughly -0.91 change in option value, which is reasonable given the option price is around 9.

3.5 Monte Carlo standard error plot

In the convergence plots later on, Monte Carlo empirical error can wobble because it is measured against a reference value which is itself estimated numerically (so the "ground truth" has its own noise). To separate this from the fundamental Monte Carlo uncertainty scale, Figure 2 plots $\hat{\sigma}/\sqrt{N}$, which is the standard error implied by the payoff variance.

This is an *auxiliary diagnostic* rather than a primary result: it confirms the expected $N^{-1/2}$ scaling of Monte Carlo uncertainty, while the main numerical comparison is provided by the MC-vs-CN convergence plots in Section 5–6.

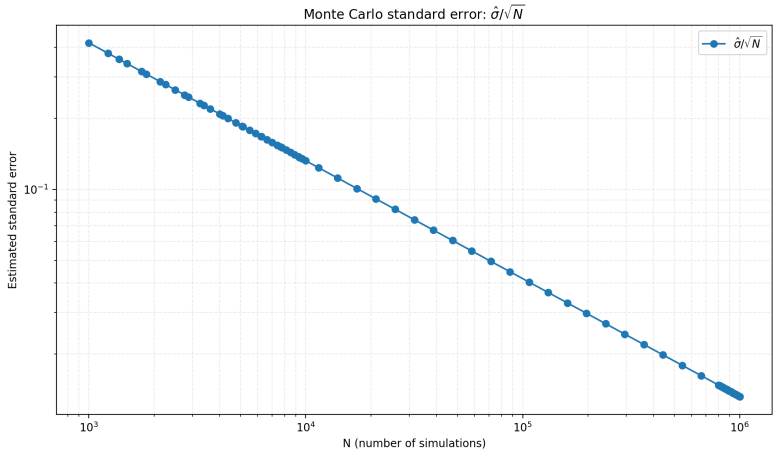

Figure 2: Monte Carlo estimated standard error line $\hat{\sigma}/\sqrt{N}$ (log–log), which follows an $N^{-1/2}$ slope by construction.

4 Crank–Nicolson Finite Difference Approach

4.1 Crank–Nicolson methodology

The Crank–Nicolson method blends explicit and implicit schemes and takes their average. In practice, this gives a second-order accurate time discretisation (in the smooth regime) and is typically much more stable than a purely explicit method.

Like the Monte Carlo method, discretisation is required. However, here we discretise the pricing PDE directly on a grid. We choose a spatial grid

$$S_i = i\,\delta S, \qquad i = 0, 1, \ldots, M,$$

and a time grid

$$t_j = j\,\delta t, \qquad j = 0, 1, \ldots, P, \qquad \delta t = \frac{T}{P}.$$

We write $V_i(t) \approx V(S_i, t)$, and at each fixed t we apply central differences in S:

$$V_S(S_i, t) \approx \frac{V_{i+1}(t) - V_{i-1}(t)}{2\,\delta S}, \qquad V_{SS}(S_i, t) \approx \frac{V_{i+1}(t) - 2V_i(t) + V_{i-1}(t)}{(\delta S)^2}.$$

Substituting these into the PDE and rearranging to isolate the time derivative gives a semi-discrete ODE system for each interior node $i = 1, \ldots, M-1$:

$$\frac{dV_i}{dt} + \frac{1}{2}\sigma^2(t)\,(S_i)^{2\gamma}\frac{V_{i+1} - 2V_i + V_{i-1}}{(\delta S)^2} + rS_i\frac{V_{i+1} - V_{i-1}}{2\,\delta S} - rV_i = 0.$$

Equivalently,

$$\frac{dV_i}{dt} = a_i(t)V_{i-1} + b_i(t)V_i + c_i(t)V_{i+1},$$

where the coefficients are:

$a_i(t) = -\frac{1}{2}\frac{\sigma^2(t)(S_i)^{2\gamma}}{(\delta S)^2} + \frac{rS_i}{2\,\delta S}$

$b_i(t) = r + \frac{\sigma^2(t)(S_i)^{2\gamma}}{(\delta S)^2}$

$c_i(t) = -\frac{1}{2}\frac{\sigma^2(t)(S_i)^{2\gamma}}{(\delta S)^2} - \frac{rS_i}{2\,\delta S}$

Let $\boldsymbol{V}(t)$ denote the vector of interior option values

$$\boldsymbol{V}(t) = \big(V_1(t), V_2(t), \ldots, V_{M-1}(t)\big)^\top.$$

Stacking the semi-discrete equations across all interior nodes yields a linear ODE system of the form:

$$\frac{d\boldsymbol{V}}{dt} = \Lambda(t)\boldsymbol{V} + \boldsymbol{w}(t),$$

where:

$\boldsymbol{V}(t)$ is the vector of option values at the interior grid asset prices.

$\Lambda(t)$ is the tridiagonal matrix containing the drift and diffusion coefficients $a_i(t), b_i(t), c_i(t)$.

$\boldsymbol{w}(t)$ is the vector of boundary contributions (only the first and last interior equations depend on the boundary values).

Grid: $(P+1) \times (M+1)$ for the full grid, with $(M-1)$ interior spatial nodes and $(P+1)$ time levels; $\delta t = T/P$ and $\delta S = S_{\max}/M$.

$$\Lambda(t) = \begin{pmatrix} b_1(t) & c_1(t) & 0 & 0 & \cdots & 0 \\ a_2(t) & b_2(t) & c_2(t) & 0 & \cdots & 0 \\ 0 & a_3(t) & b_3(t) & c_3(t) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & b_{M-2}(t) & c_{M-2}(t) \\ 0 & 0 & \cdots & 0 & a_{M-1}(t) & b_{M-1}(t) \end{pmatrix}.$$

$$\boldsymbol{w}(t) = \begin{pmatrix} a_1(t)\, V_0(t) \\ 0 \\ \vdots \\ 0 \\ c_{M-1}(t)\, V_M(t) \end{pmatrix},$$

where $V_0(t) = V(S_0, t)$ and $V_M(t) = V(S_M, t)$ are the boundary values at $S = 0$ and $S = S_{\max}$, respectively. This form makes it clear that the spatial discretisation leads to a tridiagonal linear operator, which is why each Crank–Nicolson time step can be computed efficiently by solving a tridiagonal linear system.

4.2 Boundary and terminal conditions

To make the PDE numerically solvable we truncate the spatial domain to $S \in [0, S_{\max}]$, with $S_{\max}$ chosen sufficiently large so that the put is essentially worthless at the top boundary. In the implementation used here we set $S_{\max} = 250$, which is well above $K = 140$ and produced stable results.

The boundary and terminal conditions used were:

Top boundary: $V(S_{\max}, t) = 0$.

Bottom boundary: $V(0, t) = Ke^{-r(T-t)}$.

Terminal condition: $V(S, T) = (K - S)^+$.

4.3 Matrix form and Crank–Nicolson time stepping

Let $\boldsymbol{V}(t)$ denote the vector of interior option values

$$\boldsymbol{V}(t) = \big(V_1(t), V_2(t), \ldots, V_{M-1}(t)\big)^\top.$$

Then the semi-discrete system can be written as

$$\frac{d\boldsymbol{V}}{dt} = \Lambda(t)\boldsymbol{V} + \boldsymbol{w}(t),$$

where $\Lambda(t)$ is the tridiagonal matrix built from $a_i(t), b_i(t), c_i(t)$ and $\boldsymbol{w}(t)$ contains the boundary contributions (only the first and last interior equations depend on V_0 and V_M).

Crank–Nicolson applies the trapezium rule in time. Writing $\boldsymbol{V}^j \approx \boldsymbol{V}(t_j)$, a standard CN discretisation is:

$$\frac{\boldsymbol{V}^{j+1} - \boldsymbol{V}^j}{\delta t} = \frac{1}{2}\Big(\Lambda^{j+1}\boldsymbol{V}^{j+1} + \Lambda^j\boldsymbol{V}^j\Big) + \frac{1}{2}\Big(\boldsymbol{w}^{j+1} + \boldsymbol{w}^j\Big).$$

Since we solve backwards from the terminal condition at $t = T$, at each step we know $\boldsymbol{V}^{j+1}$ and solve a tridiagonal linear system to obtain $\boldsymbol{V}^j$.

4.4 Rannacher smoothing

The payoff $(K - S)^+$ is not smooth at $S = K$, and a direct Crank–Nicolson step can produce oscillations near maturity. To reduce this, we use Rannacher smoothing: the first steps closest to maturity are replaced by two half time steps of backward Euler (fully implicit). In the code we used two Rannacher steps.

4.5 Crank–Nicolson: Computing Price & Delta

For the reported surface and $t = 0$ quantities, we used $P = 100$ time steps and $M = 300$ spatial steps on $S \in [0, 250]$. The put price at $S_0 = 140$ is obtained by interpolating between the nearest grid values at $t = 0$.

Obtained Crank–Nicolson Put Price: 9.1295

For delta, we compute the central difference on the $t = 0$ grid,

$$\Delta(S_i, 0) \approx \frac{V^0_{i+1} - V^0_{i-1}}{2\,\delta S},$$

and then interpolate to $S_0 = 140$. This gave:

Obtained Crank–Nicolson Delta Value: -0.4071

Figure 3 provides a *qualitative* view of the computed surface $V(S, t)$. It is included mainly as a visual check that the solution is smooth away from the payoff

kink and that values decay toward maturity as expected; the key quantitative validation for Crank–Nicolson in this report comes from the delta profile and the convergence plots.

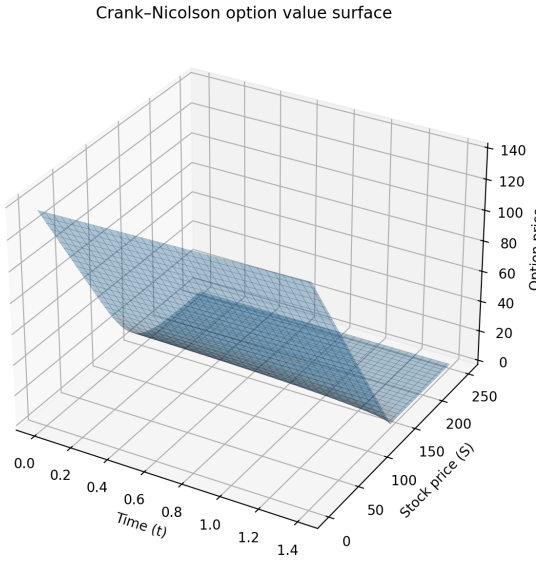

Figure 3: Crank–Nicolson value surface $V(S,t)$ over $S \in [0, 250]$ and $t \in [0, T]$.

4.6 Delta profile at $t = 0$

To make the hedging interpretation more concrete, we also plotted the full delta profile $\Delta(S, 0)$ from the Crank–Nicolson grid (Figure 4). For a put, deep in-the-money (small S) delta should be close to -1, and deep out-of-the-money (large S) delta should approach 0, with the transition happening around K.

This is exactly what Figure 4 shows: the curve is near -1 for small S, transitions sharply around $S \approx K = 140$, and tends to 0 as S increases. This is a strong end-to-end validation because it checks the PDE solution *and* the numerical differentiation used to produce hedge ratios across the full spatial grid (not just at S_0).

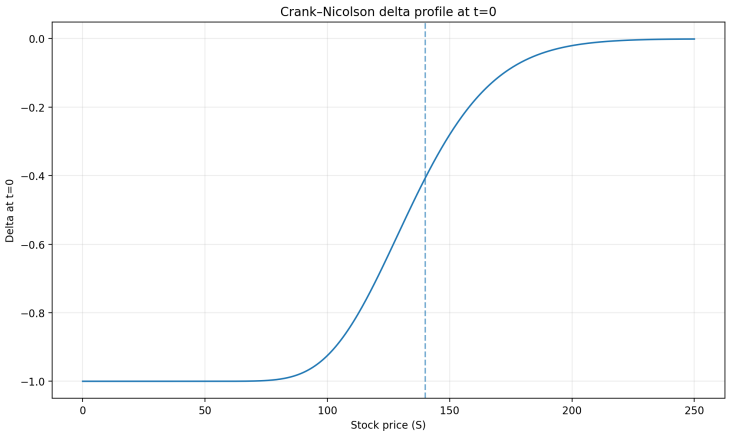

Figure 4: Crank–Nicolson delta profile $\Delta(S, 0)$ across the spatial grid (dashed line at $S = K$).

5 Results: Numerical Comparison and Market Application

5.1 Black–Scholes baseline

Since the model is not Black–Scholes, the closed-form Black–Scholes price is not an exact ground truth. However, we can construct an estimated benchmark using an effective volatility:

$$\sigma_{\text{eff}} \approx \left(S_0^{2(\gamma-1)} \frac{1}{T} \int_0^T \sigma(t)^2 \, dt \right)^{1/2}.$$

This gave $\sigma_{\text{eff}} = 0.1687$ and the estimated Black–Scholes put price 9.1492. Both numerical prices are close to this baseline, and the Black–Scholes value lies inside the Monte Carlo confidence interval.

5.2 Initial Comparison

Table 1: Initial price and delta comparison (4 d.p.).

Black–Scholes (estimated) put price	9.1492
Effective volatility σ_{eff}	0.1687
Monte Carlo put price	9.0577
95% confidence interval (price)	[8.7989, 9.3165]
Monte Carlo delta	-0.4009
95% confidence interval (delta)	[-0.4091, -0.3927]
Crank–Nicolson put price	9.1295
Crank–Nicolson delta	-0.4071

Table 1 provides the first consistency check between the methods. The Monte Carlo price is slightly below the effective-volatility Black–Scholes baseline but remains well within the 95% confidence interval, so the difference is not statistically surprising at $N = 10{,}000$. The Crank–Nicolson price is very close to the baseline, consistent with the fact that the PDE solver does not suffer from sampling noise once the grid is reasonably fine.

For delta, the Monte Carlo and Crank–Nicolson estimates agree tightly, and the Crank–Nicolson delta lies inside the Monte Carlo 95% interval. This supports that (i) the CRN-based finite-difference delta estimator is behaving sensibly, and (ii) the PDE solution is producing coherent hedge ratios at $t = 0$.

5.3 Convergence plots

To compare numerical behaviour more directly, we plotted relative errors against high-resolution reference values:

- Monte Carlo errors are computed against a higher-order Monte Carlo estimate using $N_{\text{ref}} = 1{,}000{,}000$ paths (generated independently).
- Crank–Nicolson spatial errors fix $P = 500$ and compare against a high-resolution spatial reference with $M_{\text{ref}} = 100{,}000$.
- Crank–Nicolson temporal errors fix $M = 500$ and compare against a high-resolution temporal reference with $P_{\text{ref}} = 100{,}000$.

The horizontal axis is a simple computational-effort scale: N for Monte Carlo and $M \cdot P$ for Crank–Nicolson.

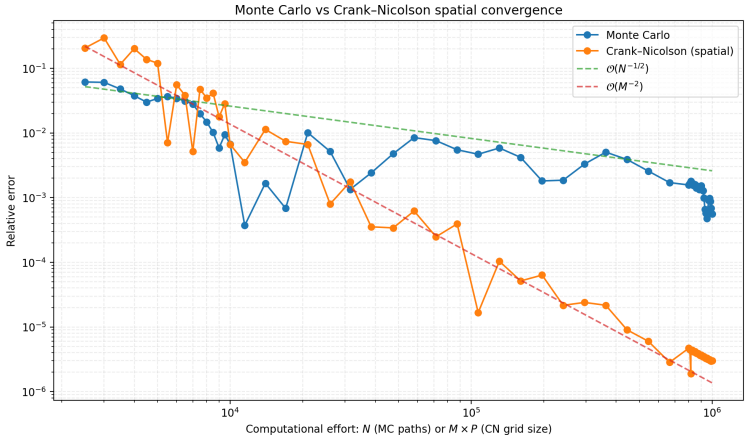

Figure 5: Monte Carlo relative error vs spatial Crank–Nicolson relative error (both measured against high-resolution references).

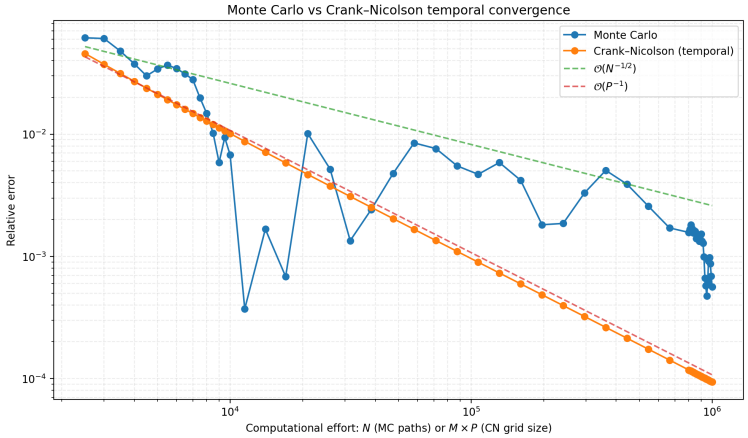

Figure 6: Monte Carlo relative error vs temporal Crank–Nicolson relative error (both measured against high-resolution references).

Figures 5 and 6 are the *primary* numerical comparison in Part A. They show a clear separation between statistical error (Monte Carlo) and discretisation error (Crank–Nicolson). On the right of both plots, at effort around 10^6, the Monte Carlo relative error sits around the 10^{-3} scale (a few $\times 10^{-4}$ to 10^{-3}), consistent with the slow $\mathcal{O}(N^{-1/2})$ rate.

In contrast, the Crank–Nicolson curves fall much faster. In the spatial refinement plot (Figure 5), Crank–Nicolson reaches around 10^{-5} error already by

effort on the order of a few $\times 10^5$, and pushes into the few $\times 10^{-6}$ range by 10^6, aligning with the $\mathcal{O}(M^{-2})$ guide once the asymptotic regime is reached. The early "spikiness" is expected before the clean second-order regime dominates (and is also influenced by reference-value accuracy and the payoff kink).

For temporal refinement (Figure 6), the Crank–Nicolson temporal error decreases roughly along the $\mathcal{O}(P^{-1})$ guide and reaches around 10^{-4} by effort 10^6. Compared with spatial refinement, temporal convergence is slower here, indicating that for sufficiently fine spatial grids the remaining error is time-discretisation driven (with the non-smooth payoff near maturity being a key contributor, mitigated but not eliminated by Rannacher smoothing). Overall, these plots explain why Crank–Nicolson can achieve very high accuracy at moderate effort, whereas Monte Carlo requires extremely large N to reach the same error levels.

5.4 Market-data application: Bloomberg SPX chain

In Part B the numerical methods are applied in a realistic setting: a Bloomberg snapshot provides an as-of timestamp, spot level and dividend yield, and the options chain provides strikes, expiries, quoted mids and implied volatilities. Under the standard log-normal model (GBM) with continuous dividend yield q, we compute:

- **BS baseline price** per option using the chain implied volatility,
- **MC price** under GBM with a 95% confidence interval,
- **CN price** by solving the Black–Scholes PDE with Crank–Nicolson,
- **Greeks** under Black–Scholes (Delta, Gamma, Vega, Theta, Rho),
- **Mispricing** defined as *model − market mid*.

The run used an SPX spot level of 682.85 and a dividend yield $q = 1.050\%$. The cleaned chain contained 947 contracts across 10 expiries and strikes from 510 to 750.

Table 2: Bloomberg SPX chain summary (Part B).

Options priced	947
Calls / Puts	280 / 667
Strike range	510 – 750
Implied vol range (mean)	10.7% – 53.5% (22.48%)
Dividend yield q	1.050%
Market mid mean (std)	3.2851 (3.1712)
BS mean (std)	3.0320 (3.0822)
MC mean (std)	2.9835 (3.0542)
CN mean (std)	3.0282 (3.0689)

Table 2 provides a key engine-level consistency check: BS and CN averages are extremely close, as they should be when fed the same inputs under the same Black–Scholes assumptions. This strongly validates the PDE implementation on real chain data. The Monte Carlo mean is slightly lower, which is consistent with finite-N sampling variability and (in practice) small input mismatches between chain fields and mid-implied calibration.

Table 3: Mispricing metrics vs market mid (Part B).

Model	**Mean (model-mid)**	**MAE**	**MaxAbs**
BS	-0.2531	0.3065	1.0717
MC	-0.3016	0.3378	1.1653
CN	-0.2569	0.3102	1.1130

The mispricing values in Table 3 should be interpreted cautiously. A non-zero mean mispricing does not automatically indicate an arbitrage: it can be driven by microstructure (bid–ask effects, discrete quoting), by differences between Bloomberg's chain "IV" field and the true *mid-implied* volatility, or by model simplifications (flat rates, log-normal dynamics, continuous dividend yield). The fact that BS and CN means are nearly identical is informative: any systematic bias vs mid is unlikely to be a PDE bug and is more plausibly an input/market-quoting effect.

5.5 Implied volatility and risk structure

For the SPX chain, the implied volatility skew and term structure provide immediate context for risk. Figure 7 shows the skew across expiries, while Figure 8 summarises how Delta/Gamma/Vega exposures vary by strike and maturity.

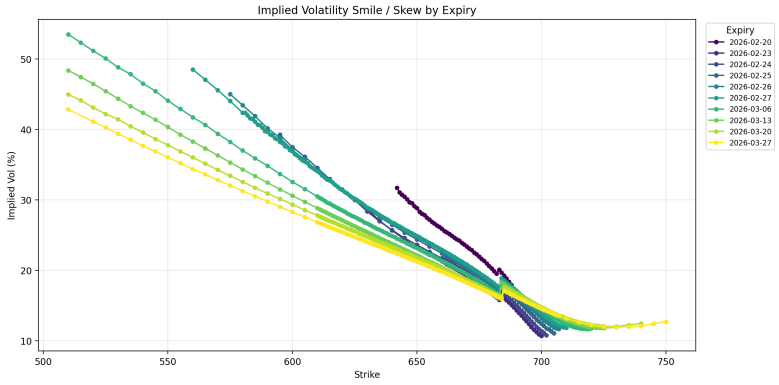

Figure 7: Bloomberg SPX implied volatility smile/skew by expiry (Part B).

Figure 7 shows a pronounced *downside skew*: implied volatilities are substan-

tially higher at lower strikes (deep OTM puts) and decline toward the ATM region. Across the higher-strike region the curves flatten and even show a slight right-tail uptick for some maturities, consistent with a skew/smile shape rather than a perfectly flat volatility surface. The separation between expiry curves reflects term structure: overall level and slope vary by maturity, which is precisely why a chain-level engine needs to handle expiry-by-expiry inputs rather than a single volatility number.

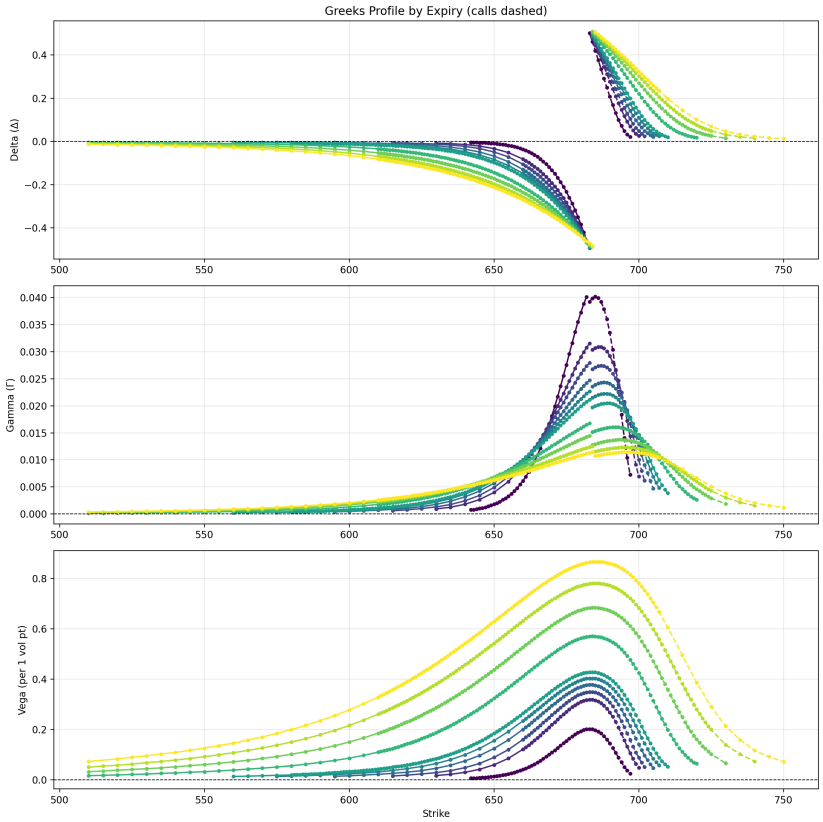

Figure 8: Bloomberg SPX Greeks profiles by expiry (Black–Scholes Greeks; calls dashed) (Part B).

Figure 8 matches standard option-risk intuition and also acts as a sign/convention check on the engine outputs. Gamma peaks around the ATM region (near spot), with the highest peaks occurring in the shorter-dated expiries, while longer-dated expiries exhibit lower gamma but higher vega. The vega curves rise with maturity and peak around ATM, reflecting greater sensitivity of longer-dated options to volatility changes.

5.6 Model vs market diagnostics

A useful diagnostic is whether model prices track market mids within plausible tolerances. Figure 9 shows mispricing distributions, and Figure 10 overlays Monte Carlo confidence intervals against mids for an ATM-ish subset.

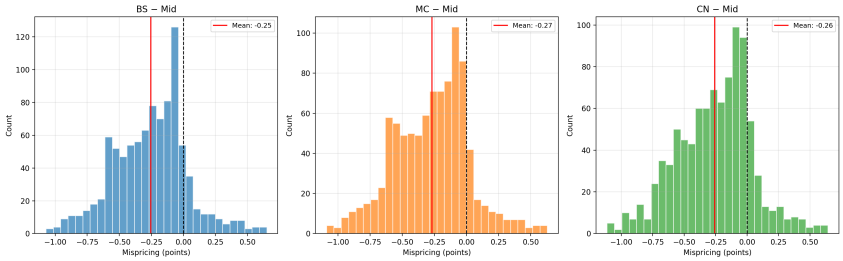

Figure 9: Mispricing distributions (model minus market mid) for BS, MC and CN (Part B).

Figure 9 shows that all three models have a negative mean (around −0.25 to −0.30 points in this run), consistent with the summary statistics in Table 3. Importantly, BS and CN distributions are very similar, reinforcing that the PDE solver is reproducing the closed-form baseline at scale. The MC distribution is slightly wider and more shifted, which is consistent with sampling noise and finite-N effects on top of any systematic input mismatch.

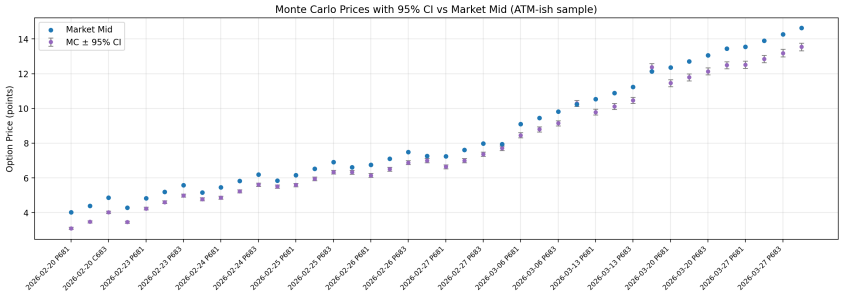

Figure 10: Monte Carlo price estimates with 95% CI vs market mid for an ATM-ish subset (Part B).

Figure 10 provides a sharper diagnostic: the Monte Carlo confidence intervals are relatively tight for the sampled subset, yet the market mids (blue) sit consistently above the MC estimates (purple) across much of the panel, and are often outside the MC error bars. This pattern is unlikely to be explained by Monte Carlo noise alone; it more strongly suggests a systematic mismatch between the model inputs used for pricing and the mid-consistent inputs implicit in the

quotes (for example, the chain IV field not matching the mid-implied volatility, or small differences in rates/dividend conventions or settlement details). In other words, this figure is useful precisely because it distinguishes "sampling uncertainty" from "systematic model/input differences".

5.7 Scenario PnL heatmap

Beyond point estimates and Greeks, a small risk-engine feature is scenario analysis: we revalue a representative "ATM-ish" subset of the chain under joint shocks to the underlying spot and to the implied volatility level. Concretely, we apply spot shocks of $\{-10\%, -5\%, 0\%, +5\%, +10\%\}$ and volatility shocks of $\{-5\text{pp}, -2\text{pp}, 0, +2\text{pp}, +5\text{pp}\}$ (where "pp" denotes absolute percentage points of implied volatility), and compute the change in value relative to the base case.

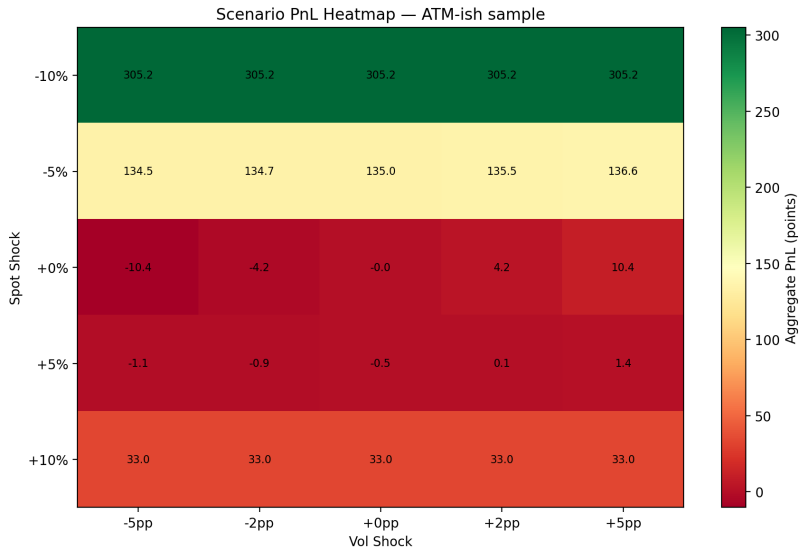

Figure 11: Scenario PnL heatmap for an ATM-ish subset under joint spot and volatility shocks (aggregate PnL in index points).

Figure 11 behaves as expected for option portfolios: for moderate spot moves (e.g. around $\pm 5\%$) the PnL responds to both spot and volatility, with higher implied volatility generally improving outcomes when the subset has net positive vega. Near the base spot level (0% shock), the monotone improvement in PnL as volatility increases is consistent with the sign of vega exposure. Under larger spot shocks (e.g. $\pm 10\%$), the heatmap indicates that spot-driven effects dominate and the dependence on volatility becomes comparatively weaker, reflecting that options can move deeper in/out of the money where vega sensitivity is reduced relative to intrinsic/delta-driven changes.

This scenario view complements the Greeks profiles by providing a direct "what-if" diagnostic: it translates local sensitivities into finite shocks and highlights where the portfolio risk is primarily spot-driven versus volatility-driven.

5.8 Conclusion: from method comparison to a usable pricing engine

Part A shows that Monte Carlo and Crank–Nicolson give consistent prices and hedge ratios under the same (non-lognormal) model assumptions, while also highlighting their distinct numerical behaviour: Monte Carlo converges slowly with statistical error $\mathcal{O}(N^{-1/2})$, whereas Crank–Nicolson can achieve substantially higher accuracy at comparable computational effort once the grid is sufficiently fine (especially under spatial refinement).

Part B demonstrates how these same numerical ideas translate into a practical workflow on real market data. Given a Bloomberg SPX options chain, the script produces (i) volatility structure plots, (ii) Greeks profiles, (iii) model-vs-market diagnostics (mispricing distributions and Monte Carlo confidence intervals), and (iv) scenario-based PnL summaries under joint spot/volatility shocks. The close agreement between Black–Scholes and Crank–Nicolson across the chain provides a strong internal consistency check on the PDE implementation. Monte Carlo serves as an independent probabilistic cross-check and supplies confidence intervals that help distinguish sampling uncertainty from more systematic discrepancies driven by inputs or market quoting conventions (e.g. chain IV fields not matching mid-implied calibration).

Overall, the project is therefore not only a numerical comparison of Monte Carlo vs Crank–Nicolson, but also an application of those methods to realistic SPX option data: taking market inputs, generating prices and sensitivities at scale, and producing diagnostics that connect directly to hedging, risk management, model validation, and scenario-based "what-if" analysis.

References

[1] John Armstrong. Lecture notes for 7CCMFM06, Computational and Numerical Methods in Mathematical Finance. https://keats.kcl.ac.uk/course/view.php?id=119832, 2025.