GARCH Risk Modelling (Aglaia) — *Python (arch, pandas), Power BI, LaTeX*

While working with the risk team at Aglaia, a multi-family office in Singapore, I supported enhancements to short-horizon risk metrics across 30 portfolios with a value totalling over $4B USD. Noticing persistent volatility clustering in equity markets, I explored more adaptive modelling to complement existing tools. I implemented a GARCH(1,1) process to model conditional volatility:

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2, \quad \text{where} \quad \epsilon_t = r_t - \mu.$$

Here, the innovation term ϵ_t was derived from de-meaned log returns, and parameter estimates were obtained via maximum likelihood. To account for structural heterogeneity across portfolios, I conducted a volatility clustering analysis (based on pairwise distance between rolling standard deviation profiles), enabling segmented GARCH calibrations. The output fed into VaR estimates at a 99% confidence level. I then applied Kupiec's Proportion of Failures (POF) test:

$$LR_{POF} = -2\ln\left((1-p)^{T-x}p^x\right) + 2\ln\left((1-\hat{p})^{T-x}\hat{p}^x\right),$$

with x breaches in T observations, comparing the empirical failure rate $\hat{p}$ to the theoretical threshold p. Results showed breach rates consistent with model assumptions. This analysis improved short-term risk awareness and provided quantifiable evidence of regime-sensitive VaR performance, enriching the team's existing monitoring tools.

Deep Hedging Under Rough Volatility (In Progress) — *Python (TensorFlow), LaTeX, FFT*

As part of my MSc Research Thesis I have developed a data-driven hedging framework for options under the rough Bergomi model. The instantaneous forward variance ξ_t follows:

$$\xi_t = \xi_0 \exp\left(\eta \int_0^t (t-s)^{H-0.5}\, dW_s\right), \quad H < 0.5,$$

capturing empirically observed roughness in volatility and long-memory effects that traditional models (e.g., Heston, Black–Scholes) fail to replicate. Simulated 100,000 asset price paths under rBergomi dynamics using:

- Cholesky decomposition for exact fBM covariance sampling.
- Circulant embedding for efficient FFT-based sampling on large time grids.

Each path included a correlated Brownian motion for the log-asset and a fractional kernel-driven volatility process. I trained a recurrent neural network (GRU/LSTM) to output hedge ratios Δ_t at each time step based on observable state variables (e.g., moneyness, time-to-maturity, volatility). The agent minimises a convex risk objective:

$$\mathcal{L} = \lambda \cdot \mathrm{CVaR}_\alpha(P\&L) \; + \; (1-\lambda)\,\mathbb{E}\big[(P\&L)^2\big].$$

The framework is designed to learn dynamic hedging strategies that target tail risk directly. As part of ongoing work, I plan to benchmark the learned strategy against Black–Scholes delta hedging on identical rough volatility paths, with a focus on tail performance and hedge path behaviour under market stress.

github.com/zaccheus-lines/rBergomi

Shortest Vector Problem Solver (C++) — *C++, OpenMP, Python (testing)*

Built a high-performance C++ solver for the Shortest Vector Problem (SVP), used in lattice cryptography and integer optimisation. Given a basis $B \in \mathbb{R}^{n\times n}$, the solver computes:

$$\min_{x \in \mathbb{Z}^n \setminus \{0\}} \|Bx\|_2$$

Implemented LLL-based basis reduction to tighten enumeration bounds, followed by a Schnorr–Euchner branch-and-bound search with a parallel tree traversal (where applicable) and radius pruning. The code is designed for low-latency performance and efficient memory usage. A comprehensive Python test suite (unit, property, and regression tests) ensures numerical stability and correctness.

github.com/zaccheus-lines/ShortestVectorProblem

VBA Automation for Reporting Pipelines (Aglaia) — *VBA, Power BI, Excel, ExpressSoft*

Designed and deployed a fully automated end-to-end reporting pipeline to streamline daily risk and performance monitoring across 30 portfolios. Starting from raw exports from ExpressSoft, I developed VBA macros to automate:

- **Data Extraction:** Automated parsing of multi-sheet Excel exports from ExpressSoft, consolidating fragmented position and trade-level data.
- **Data Transformation:** Cleaned, reshaped, and reconciled datasets (e.g., mapping ISINs to strategy-level tags, handling missing risk fields) via structured VBA routines.
- **Data Load:** Generated structured output tables directly ingested by linked Power BI dashboards through refreshable connections.

The automation significantly reduced manual processing time, enabling near-real-time reporting. This not only accelerated the daily risk briefing process for portfolio managers but also increased accuracy, reproducibility, and audit-ability of the data pipeline - particularly valuable during periods of elevated market volatility.

Spam Filtering with Credal Classification — *Python, NumPy, scikit-learn*

For my undergraduate dissertation I built a robust spam classification system based on credal classification, extending Naive Bayes to work with sets of priors $\mathcal{P}$ to model epistemic uncertainty. Computed lower and upper bounds on posterior probabilities:

$$\underline{P}(c|x) = \inf_{P \in \mathcal{P}} P(c|x), \quad \overline{P}(c|x) = \sup_{P \in \mathcal{P}} P(c|x)$$

Implemented a novel bound for interval dominance by solving a constrained multivariate minimisation problem to determine whether:

$$\underline{P}(c_i|x) > \overline{P}(c_j|x)$$

held under all admissible priors in $\mathcal{P}$. This enhanced the decision rule for multi-class classification under uncertainty. The method was integrated with `scikit-learn` pipelines and benchmarked on the Apache SpamBase dataset, achieving a 1.5% F1 improvement over standard Naive Bayes and increased robustness under label noise and adversarial perturbations.

github.com/zaccheus-lines/NaiveCredalClassifier

Fixed Income Yield Curve Construction and Hedging — *Python, LaTeX, Excel*

Developed a full fixed income analytics pipeline starting from raw bond price data. Bootstrapped zero-coupon yield curves from coupon-bearing government bonds by solving sequential discount factors:

$$P = \sum_{i=1}^{n} \frac{C_i}{(1+r_i)^{t_i}} + \frac{F}{(1+r_n)^{t_n}}$$

and compared output to market and dealer-implied curves. Estimated the discount function using both cubic spline interpolation and the Nelson–Siegel–Svensson (NSS) model. Analysed model fit, sensitivity to knot placement, and factor loadings to interpret term structure dynamics:

$$D(t) = \beta_0 + \beta_1 \frac{1 - e^{-t/\tau}}{t/\tau} + \beta_2 \left(\frac{1 - e^{-t/\tau}}{t/\tau} - e^{-t/\tau} \right)$$

Computed portfolio key rate durations (KRDs) to quantify exposure across the yield curve and constructed zero-cost long-short strategies to express curve steepening views. The project culminated in a hedging and relative value framework grounded in clean curve construction and sensitivity analysis, directly applicable to credit and rates trading.

zaccheus.lines@hotmail.com | +447487821133 | **LinkedIn** | **GitHub**