

Emilian Joseph Bowry

07831799619 emil.bowry@icloud.com <https://github.com/emilbowry>

Education

Trinity College, University of Cambridge

BA, Engineering Tripos, 2020-2025

Note: Degree awarded unclassified due to medical intermission preventing completion of Part IIB

Judge Business School, University of Cambridge

Accelerate Cambridge, August 2022-July 2023

Experience

Software Developer

Remote

AI Compatible
August 2025 - Current

- **Developing an automated privacy policy analysis model, this included:**

- Utilises Large Language Models for initial data extraction: Defining the output form using an OpenAPI schema object for consistently returned format for processing.
- Improving and refining the Hader et al's methodology [DOI: 10.1007/s00607-024-01331-9] for a generative, and automatable process to gain more nuanced insight into privacy policy and reduce the amount of API calls.
- Unsupervised semantic equivalence detection model utilises Bayesian Inference, Topology, Linear Algebra, NLP techniques and Non-linear Systems analysis. Part of my system derived an analog for Random k Conditional Nearest Neighbours, a recently published technique for the classification of high-dimensional data.

- **Redeveloping full-stack website:**

- Defining a PDF creation system that composes PDF document directly a set of data/json using the PDF 'page description language'.
- Generalising CSSTypes (Typescript) to be able to create more complex, type-safe style objects.
- Creating a geometry engine to compose mathematically definable UI styles, including the capability to have seamless non-uniform backgrounds over disconnected, and unconventionally shaped UI components.
- Simulating a toroidal topology for a scroll-bar to define well ordered cyclical UI wheel elements.
- Dynamic Background Image generator: Efficiently creates tileable background images on the clientside without the need for image processing tools.
- Managing databases with Prisma, PostgreSQL, mongoDB.
- Handling authentication, payment processing and third-party APIs.
- Hit metadata analysis engine, creates interaction based profiles utilising site-entry points, device and location data to profile and analyse site interactions.

- Managing cloud infrastructure, including virtual machines and networking.

Co-founder

Cambridge

Luucid.tech
August 2022 - October 2023

- Created novel electrochemical and material mechanisms for detecting spiking agents in beverages.
- Grant writing to organising a UK wide drink spiking prevalence assessment.

- Full stack web development using a variety of languages and technologies.
- Anomaly detection system that analysis of fuel levels in a vehicle fleet to infer when there may have been an incident of fuel theft.
- Analysed new business opportunities and ventures, writing insight articles.
- Led skill days, which taught developers the low-end networking implementations of the tools they use: <https://github.com/emilbowry/NetworkProgrammingLesson>
- Organised the weekly cyber-security brief about emerging threats and vulnerabilities.

**Published and
Open Source
Software**

Plotting Tools:

<https://github.com/emilbowry/Plots>
<https://pypi.org/project/plottingtools-emilbowry>

Extension of the python Plotly library to conveniently visualise 4-dimensional datasets. Using metaprogramming techniques to create a robust and adaptable framework.

Code Editor:

<https://github.com/emilbowry/editor>

A fork of Microsoft VSCode that:

- Removes all telemetry pipelines at the source, rather than no-opping the end-points like the ‘Codium’ alternative.
- Removes LLM integrations and bloat.
- Added new features, like a persistent homepage, and cross-codebase note taking system

Monochrome:

<https://github.com/emilbowry/monochrome>

Fork of the strict Black code-formatter. Using tabs and other slight formatting changes

AST Debug Logger:

<https://github.com/emilbowry/AST-Debugger>

A debugging tool that intercepts python code before execution to toggle any ‘debug’ flags, even from orphans and disconnected nested code. It also intercepts and saves a logs.

Algorithmic Trading resource: <https://github.com/emilbowry/algorithmicTrading>

A teaching resource that develops and implements knowledge gained in my signals processing, statistics, systems and other engineering courses, as well as other information I learnt from Dexter’s Notes of the Mathematics Tripos. This includes applications of:

- Analysis tools like: K-means clustering, linear regression
- Stastical tests like Augmented Dicky-Fuller, Variance Ratio, Generalised Autoregressive Conditional Heteroskedasticity, and Stationarity Analysis
- Signal Processing techniques including Fourier, Wavelet and Hilbert transforms, continuous and discrete filters.
- Optimisation techniques, it covers: linear programming, non-linear programming and convex optimisation
- Other general tools like PCA, semantic analysis, HMMs and inference tools, Metropolis-Hastings Algorithm

Projects and Additional Experience

Phasor Average Estimator: An estimator that models each event as a phasor. Each observation adds a new phasor to the model, allowing us to estimate where we expect to see events in the future. Modelling an event as a phasor allows us to:

- Be robust to noise and jitter: Interference of the phasors allows this to cancel out in probability
- Handle missed signals: The phasors still process regardless of observing a signal or not.
- React instantly to periodicity changes: Most estimators would require some evidence accumulation period before it changes period regime, this just spins a new phasor.
- Handle false positives: A new phasor adds to the bank, adding a different frequency component, however until repeated signals are observed, this does not counteract the dominant frequencies.
- Make reasonable predictions arbitrarily far into the future, given the events we have seen.
- Does not require any exogenous and somewhat arbitrary values like damping ratio since the energy can be bound.
- Scale-Invariant: Can handle signals in arbitrary time-frames and amplitudes

This was built to solve issues with hardware jitter and Inter-Symbol Interference (ISI), and sensor saturation in a Molecular Communication system.

Neural Data Analysis: Built a simulation framework for simulating Lateral Intraparietal Cortex (LIP) neurons. The core objective was to evaluate and test different statistical models for neuron impulses. The framework included:

- Common interface for simulation parameters and to run simulations of different models: Ramp, Step, Hidden Markov Model Approximations.
- Normalised methods to run statistical analysis on these simulations, i.e Peri-Stimulus Time Histogram, Fano-Factor.
- Cache and save simulation results in a database for later analysis.
- Centralise transformation between different formats, like ones appropriate for histograms, spike rasters and the ability to apply smoothing and other functions to the data.
- Graphing capability to easily evaluate each model and the variations it has due to different parameters, including 3D graphs and mutable plots with sliders, showing how the surface changes as we vary parameters

This framework was then used to do some larger analysis like evaluating model brittleness and mismatch.

iCloud Find-my messaging service: A system to piggyback on Apple's "Find my iPhone" API to remotely communicate between devices without knowledge of any identifiers like IP addresses (Side-Channel Analysis). This involved reverse engineering the protocol to analyse packet payloads/and database structures without documentation.

Automated Notes Reasoning: A system that makes inferences about my course notes, given their titles.

- Utilised Universal Properties (specifically Categorical Products) to programmatically define the existence of intersectional notes. This allowed the system to strictly enforce schema completeness by identifying missing 'product' nodes between disjoint topics.

- Modelled the relationship between Note Syntax (Grammar) and Semantic Concepts as an Adjunction. This allowed for bidirectional consistency checking: ensuring every valid grammatical title mapped to a logical object, and every logical gap could be described by a generated title.
- Utilises the note structure entirely by how they relate to these ‘Type Functors’ (Definitions, Equations, Methods), leading to understanding of the small note category.

The implementation necessitated reverse engineering parts of the ‘Obsidian Markdown Editor’, in order to have a better integration than was offered in the public API and documentation; parser design: Writing a Lexer/Parser for the custom note title grammar and knowledge graph construction: Automating the generation and verification between objects based on grammatical rules.

Module Type Objects: Built a parallel object system using modules as the core components in python to allow for more flexible and better controlled attributes.

Awards and Achievements

Goldman Sachs: Awarded a scholarship and Engineering Spring week.

Imperial College London: Awarded the President’s Scholarship to Imperial College London, given to the top 112 candidates that demonstrated the “highest academic excellence at interview”.

Referees available upon request