SONNETS : Scalability Oriented Novel Network of Event Triggered Systems

EPSRC Programme (EP/X036006/1)

David Thomas, Wayne Luk, Rishad Shafik, Andrew Brown

Mark Vousden, Graeme Bragg, Alex Yakovlev, Kabita Adhikari, Walter Distaso

Shidur Rahman, Alex Chan, Tousif Rahman, Ce Geo, Paul Labonne, Bob Pattison

May 2025

This is the first advisory board report for the SONNETS (Scalability Oriented Novel Network of Event Triggered Systems) programme grant. The objective of the SONNETS project is to explore a new paradigm for describing large-scale distributed systems called Event-Triggered Computing (ETC), with the objective of pioneering new AI-based financial risk-analysis capabilities for the UK.

The report contains eight main sections:

- Section 1 — A one page executive summary of the report.
- Section 2 — Advisory board meeting information, including schedule and locations.
- Section 3 — An overview of the SONNETS programme's overall objectives and current progress.
- Three sections describing research in the three research areas underlying SONNETS:
  - Section 4 — ETC : The Event-Triggered Computing approach and ETC infrastructure developed.
  - Section 5 — ETM (Event-Triggered Modelling) : Financial modelling using Causal Graphs.
  - Section 6 — ETAI (Event-Triggered AI) : Two new approaches to using AI to analyse financial risk.
- Section 7 — Programme management and collaboration information.
- Section 8 — Progress on engagement, dissemination, and publications.

We (the SONNETS team) recognise that this has ended up a long report despite our best efforts and apologise — we were *aiming* for something shorter. We intentionally recruited a diverse advisory board to represent all interested parties, so recognise that some members will be less interested in certain topics. If so, you may wish to read the SONNETS overview section, then skip to the research area (ETC, ETM, or ETAI) that most resonates with you, as we have tried to make them readable as standalone sections. The final two sections are about the research programme, and can be omitted by readers who are purely interested in the technical research aspects.

Document Version 1: May 20th 2025

1. Executive summary

SONNETS (Scalability Oriented Novel Network of Event Triggered Systems) is a five-year UKRI-funded research programme. It aims to transform real-time national financial risk analysis through a novel computing paradigm: **Event-Triggered Computing (ETC)**. SONNETS integrates computing, modelling, and AI to create scalable, explainable, and deployable systems for financial surveillance and stress testing.

Inspired by a vision of real-time financial risk monitoring, SONNETS targets two core demonstration systems:

- **System-A**: Continuous, real-time financial surveillance.
- **System-B**: Scenario generation and stress testing for regulatory insights.

These systems will provide a “digital shadow” of the UKs financial system, enabling timely, explainable, and actionable insights for policymakers and regulators.

SONNETS is structured around three interdependent research tiers:

- **ETC (Event-Triggered Computing)**: Cloud-deployable infrastructure for distributed, large-scale, real-time applications created using an "event-triggered" programming abstraction.
- **ETM (Event-Triggered Modelling)**: Real-time financial analysis to uncover financial interdependencies, estimate shock propagation, and generate risk scenarios.
- **ETAI (Event-Triggered Artificial Intelligence)**: Novel types of machine learning and feature extraction to provide transparent and interpretable forecasting.

Achievements during the first year (phase 1) of the project include:

- Integrated demonstrators combining real-time data ingestion, modelling, and AI-driven forecasting.
- Creation of the Lyrical framework for developing ETC applications in multiple languages, and supporting distributed containerised deployment.
- High performance causal graph extraction algorithms, which can be used to estimate shock propagation through large numbers of financial assets over time.
- Application of Tsetlin Machines (a novel type of Machine Learning algorithm) to financial forecasting, using a new "Graph Visibility" technique to extract features from the market.

Over the next two years (phase 2) we plan to:

- Develop and deploy a live, always-on risk analysis dashboard with public and private versions.
- Continue ETC infrastructure development and add verification and performance optimization.
- Combine causal graph analysis with other modelling methods such as scenario generation and evaluation, and explore performance improvements using GPUs and FPGAs.
- Further explore how explainable AI methods can be used support national risk analysis.
- Continue stakeholder engagement and outreach to regulators, academia, and the public.

Table of Contents

2. Advisory Board Meeting information

Thank you for agreeing to attend the first SONNETS Advisory Board Meeting, which will take place on May 30 from 10:00 till 16:00 at Imperial College London.

The advisory board will review our research progress and future plans, and provide guidance on both the research and the overall project.

We provide information about the grant in two forms:

- An Advisory Board Report (this document), describing research and project progress so far.
- Presentations and demonstrations of the research outputs during the meeting.

2.1. Meeting location

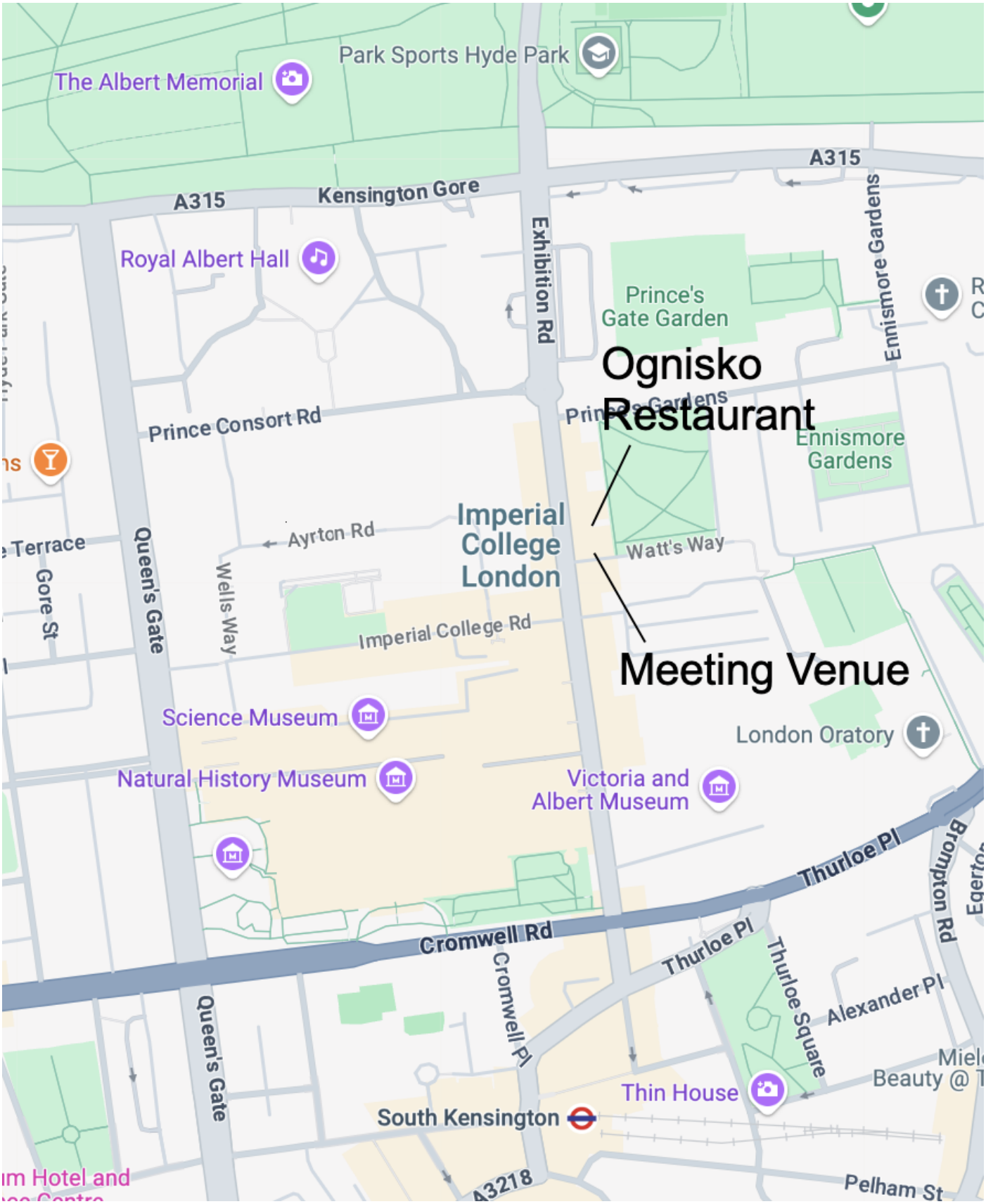
The meeting location is at:

- The Ballroom
- 58 Prince's Gate (just to the East of Imperial College)
- South Kensington
- London
- SW7 2PG

Travel information can be found at: https://www.imperial.ac.uk/students/new-students/undergraduates/arrivals-and-induction/directions-to-south-kensington-campus/

A campus map can be found at: https://www.imperial.ac.uk/media/imperial-college/visit/public/SouthKensingtonCampus.pdf, where 58 Prince's Gate can be found in Box C2 on this map.

For those arriving or available the night before, we are having dinner the night before at Ognisko (https://www.ogniskorestaurant.co.uk/), which is 20 metres North of the meeting venue at 55 Exhibition Rd, SW7 2PG.


Park Sports Hyde Park
The Albert Memorial
A315
Kensington Gore
A315
Royal Albert Hall
Exhibition Rd
Ennismore Gardens
Prince's Gate Garden
Ognisko Restaurant
Prince Consort Rd
Ennismore Gardens
Imperial College London
Ayrton Rd
Watt's Way
Queen's Gate
Wells Way
Gore St
Imperial College Rd
Meeting Venue
Science Museum
London Oratory
Natural History Museum
Victoria and Albert Museum
Thurloe Pl
Brompton Rd
Cromwell Rd
Thurloe Pl
Thurloe Square
Alexander Pl
Cromwell Pl
Queen's Gate
Thin House
South Kensington
A3218
Pelham St

2.2. Timetable

May 30th 2025

An indicative timetable is as follows, though the internal structure may change between preparing the report and the meeting day.

- 9:30 - 10:00 : Arrival and coffee
- 10:00 : Meeting starts
- 10:00 - 10:05 : Introductions to Team and Advisory Board
- 10:05 - 10:45 : SONNETS Overview, including management and goals
- 10:45 - 11:15 : ETC : Event-triggered Computing
- 11:15 - 11:30 : Coffee break
- 11:30 - 12:00 : ETM : Event-Triggered Modelling approaches
- 12:00 - 12:30 : ETAI : Event-Triggered AI methods
- 12:30 - 13:00 : Integrated Demonstrators
- 13:00 - 14:00 : Lunch, including more demonstrators
- 14:00 - 14:30 : Explainable AI using Tsetlin Machines
- 14:30 - 15:00 : Future plans
- 15:00 - 15:30 : (Advisory board private discussion)
- 15:30 - 16:00 : Feedback of advisory board
- 16:00 : Meeting closes

2.3. Travel and accommodation

If you need travel or hotels booked, then this can be organised and paid for via University of Southampton. Please get in touch with David (dbt1c21@soton.ac.uk) to identify specific needs and we can book it for you.

2.4. Non-disclosure agreement

As part of the legal agreement between the universities, there is an expectation that going forwards the formal members of the advisory board will enter into a mutual confidentiality agreement. This is to protect both confidential information presented by the research team, and confidential information contributed by advisory board members.

For this first advisory board there is as yet no confidential information from the SONNETS team (we intend to make this report public and are actively presenting and publishing the work elsewhere), and we do not (initially) expect to receive confidential information from advisory board members. For future boards, and for follow-on engagement after this meeting, we will ask members to sign a two-way confidentiality agreement, as we do expect that proprietary and sensitive data will need to be discussed.

3. SONNETS overview

SONNETS is a five year programme funded by UKRI which will provide new capabilities for observing and analysing national-scale complex real-world systems in real-time, creating a continually updated live estimate of key features and properties of those systems. The core research is domain-agnostic, but the work is heavily focussed on the specific demonstration domain of financial risk analysis. The research brings together investigators across three areas:

- Computing: new paradigms for creating and running loosely coupled applications which can exploit heterogenous computer hardware distributed across the cloud.
- Modelling: theory and tools for modelling complex systems, including automatic scenario generation and evaluation.
- Artificial Intelligence (AI): new methods for learning and inference that can work with multiple types of temporal data, while providing explainable output to users.

The main case study driving the research is to provide new tools to view and understand the current financial risk position of the UK, through a demonstrator which is deployable in contemporary cloud infrastructure and applies both the new and existing modelling and learning approaches to provide daily or hourly real-time updates.

3.1. Context and prior work

While SONNETS is exploring new ideas, some of these build on work started in the previous POETS (Partially Ordered Event-Triggered Systems) programme grant (EP/N031768/1) which ran from May 2016 to Nov 2022. The POETS project explored an approach called event-triggered computing which decomposed computational problems into millions of tiny "shared nothing" communicating state machines. The state machines then executed concurrently in a bespoke hardware stack, supporting 10000+ threads within a single machine sending tiny messages to each other. POETS was aimed at HPC-style tasks, and saw success in a number of disparate areas including particle simulation, spiking neural networks, and genetic imputation.

A strength and a weakness of POETS was the bespoke hardware stack, which was a barrier to more general deployment, particularly as cloud computing came to support the movement of both standard and hardware accelerated compute off-premises. A strength was the enforced "shared nothing" programming approach for large-scale concurrent systems, and during POETS a number of opportunities were identified for using this approach to engineer cloud-based systems spanning a much larger range of tasks. At the same time we observed arising opportunities and challenges in deploying and managing AI and modelling at scale, and identified that some of the approaches from POETS might help address those challenges. As a result a sub-set of the POETS team re-thought the approach, and with additional team members conceived the SONNETS project, with the idea of integrating novel Machine Learning (ML) and real-time large-scale modelling through event based paradigms.

3.2. Over-arching vision

When trying to identify a challenging application domain that would provide new capabilities for the UK, financial analysis was identified as a good candidate. However, rather than working on the profit-seeking commercial side, which is already served by large proprietary in-house systems in financial

institutions, we chose to look at *national financial risk analysis*. Clearly there is already a great deal of risk analysis and estimation going on within UK regulatory and governmental bodies, but many of the processes and systems operate relatively slowly on a month-by-month level, with a lot of manual data gathering and processing of data. Particularly inspiring is a quote from a 2014 lecture by Andy Haldane, previously Executive Director and Chief Economist of the Bank of England:

> "I have a dream. It is futuristic, but realistic. It involves a Star Trek chair and a bank of monitors. It would involve tracking the global flow of funds in close to real time… in much the same way as happens with global weather systems… Its centerpiece would be a global map of financial flows, charting spill-overs and correlations. Such a global financial surveillance system could serve a number of policy ends. It would allow policymakers to monitor the evolution of the financial system in real time, as it expanded, contracted and changed shape. It would also allow them to simulate and stress test this system to help detect impending financial cliff-edges.", A. Haldane, 2014

Haldane's dream involves two capabilities:

A. Continuous financial surveillance with real-time analysis.
B. Exploration of scenarios to detect financial instabilities and to support policy design.

This focus of SONNETS will be the UK national financial system rather than the global financial system Haldane imagined, although it must also cover global financial effects on the UK's financial system. Insights from (A) would enhance (B); insights from (B) would lead to improvements in (A). Two demonstration systems, System-A and System-B, will be developed to showcase these capabilities.

For (A), one key issue is what type of useful analysis could be performed using continuous financial surveillance. One initial example would be monitoring and forecasting national "health indicators" like the UK Monetary and Financial Conditions Index (MFCI), and trying to estimate the risk over the next few days. Our research on (A) will take into account the experience from these national health indicators, while considering how best to exploit the availability of more fine-grained data from high-frequency trading. System-A will demonstrate end-to-end capabilities to cover (A).

For (B), this capability is intended to support regulators in two ways:

1. Creating more accurate risk models to help prioritise regulatory efforts, and gain real-time and near-real-time insights into institutional behaviour and market outcomes so that regulators can intervene earlier.
2. Providing greater visibility of firms and markets operating around the perimeter of what is being regulated and what is not.

Two areas of particular interest are using ML and AI to identify possibly negative outcomes (such as credit default or financial distress) before they happen, and help policy design through causal estimation. Our research will enhance regulatory efficiency and effectiveness: it will lead to a new generation of stress tests covering a wider range of scenarios for the entire UK financial system; System-B will cover these.

(A) and (B) will be made possible through advances in ETM (Event-Triggered Modelling), ETAI (Event-Triggered AI) and ETC (Event-Triggered Compute). While there is much research on many large-scale domain-specific mathematical computing problems (weather modelling, material science, drug discovery), there is little work on national-level real-time financial risk analysis. This key area is being addressed by SONNETS: the two demonstration systems show how the 3 tiers work together.

SONNETS will adopt the idea of Event-Triggered Systems (ETS) to create distributed modelling and analysis capabilities. This will result in a form of "digital shadow" for the UK's financial system, which strips away all the detail while retaining the most important signals and relationships. A measurable outcome for SONNETS is a roadmap for regulators to adopt SONNETS technology.

3.3. Underlying technical ideas

We chose national financial risk as a target because it is too large and complex to handle using traditional techniques: thousands of financial institutions trade millions of instruments (stocks, bonds, and other assets) with each other, with millions of distinct trades per day, resulting in a constantly changing portfolio in each institution. Local risk can be evaluated within each institution, but it is difficult to evaluate national-level risk without considering the *network* of trades across *multiple* institutions. This is difficult because building a consistent snapshot of all institutions is a huge problem, from a technical, legal, and political point of view. Even if we could capture all these data, existing analytical tools are not able to process this type of massive rapidly-evolving un-labelled data-set in real-time — it is no good learning about a high-risk scenario months after the scenario already came true.

The vision of SONNETS is to tackle this problem at multiple levels simultaneously: it is not just a compute problem, or an analysis problem, or a modelling problem — it is all three. We take the ETC approach from POETS, and generalise it to support much larger systems, hence ET***S***. Through this abstraction we are co-developing new real-time modelling and analysis capabilities that can tackle the entire problem holistically, using heterogeneous cloud-resources to provide a robust stack that can handle multiple data sources. This means that we must develop three inter-dependent capabilities, all of which are supported by the idea of ETS:

- Event-Triggered Computing (ETC): a programming paradigm for ETS which makes it easier to develop efficient large-scale applications using heterogeneous cloud computing resources.
- Event-Triggered Modelling (ETM): a new modelling and analysis approach which exploits event-triggered compute and event-triggered AI to understand national-level problems, with UK-wide financial risk modelling as the main target.
- Event-Triggered Artificial Intelligence (ETAI): a suite of AI technologies which can handle large irregular data-sets efficiently in the cloud. These will be designed as "explainable AI" which are able both to make an inference from given input data *and* to produce a human-level explanation for how and why that inference was made.

3.4. Vision for final demonstration systems

National-level risk modelling is already performed through regulatory, governmental, and academic activities, using a number of methods. One approach is national-level stress testing, where regulators define a severe market shock scenario and ask each institution to individually estimate their exposure under that scenario (i.e. how much money each institution might lose), then combining those to produce

a national-level risk estimate. Another approach is the creation of combined risk metrics, using combinations of market data and other estimations of economic activity and sentiment. Both of these are effective, but are also relatively slow compared to the speed of the economy and may lag behind reality by weeks or months. To reduce this lag it is now common to use forecasting methods to provide "nowcasts", which use many high-frequency low-lag variables to estimate the current value of high-lag risk metrics at the current point in time.

With SONNETS we want to create and demonstrate alternative real-time risk capabilities which are:

- **Real-time** : new data is integrated into risk models and analysed in real-time as it arrives, producing updates at the daily or hourly level.
- **Deployable** : AI and modelling is designed from the outset to take into account the realities of running in cloud infrastructure, such as ensuring they support incremental model updates and re-fitting and can achieve latencies needed for real-time operation.
- **Explainable** : the system's outputs and conclusions, particularly those derived from ML and AI, should be meaningful to end users, including the derivation of the underlying models.
- **Extensible** : it should be feasible to integrate classic statistical techniques, alongside new approaches using ML and other forms of modelling.
- **Robust** : running systems should be secure and reliable by default.
- **Heterogeneous** : we take advantage of the full gamut of cloud hardware to improve performance and efficiency.

Defining exactly what the final risk modelling capability should be is part of the research, as it requires extensive collaboration between ourselves and regulatory and industrial users to explore the boundaries between useful and feasible - hence the diversity of investigator skill sets and research partners. Advisory boards are one of the mechanisms we use to shape both the overall headline goal, as well as the technical research which underlies it.

At the highest level of abstraction, our vision for the ultimate demonstration system is a console or dashboard that allows an end-user to get a big picture view of the current "riskiness" of the UK. A minimal level of functionality is to include live views of classic risk indicators and metrics (System-A), but with a "nowcasting" focus on presenting live estimates about today and the near future, rather than lagged data describing a week, month, or quarter ago. Alongside standard nowcasting methods, for example the widely used "MIDAS" approach and dynamic factor models, the system will present more exploratory methods based on ML/AI to attempt to estimate current risk.

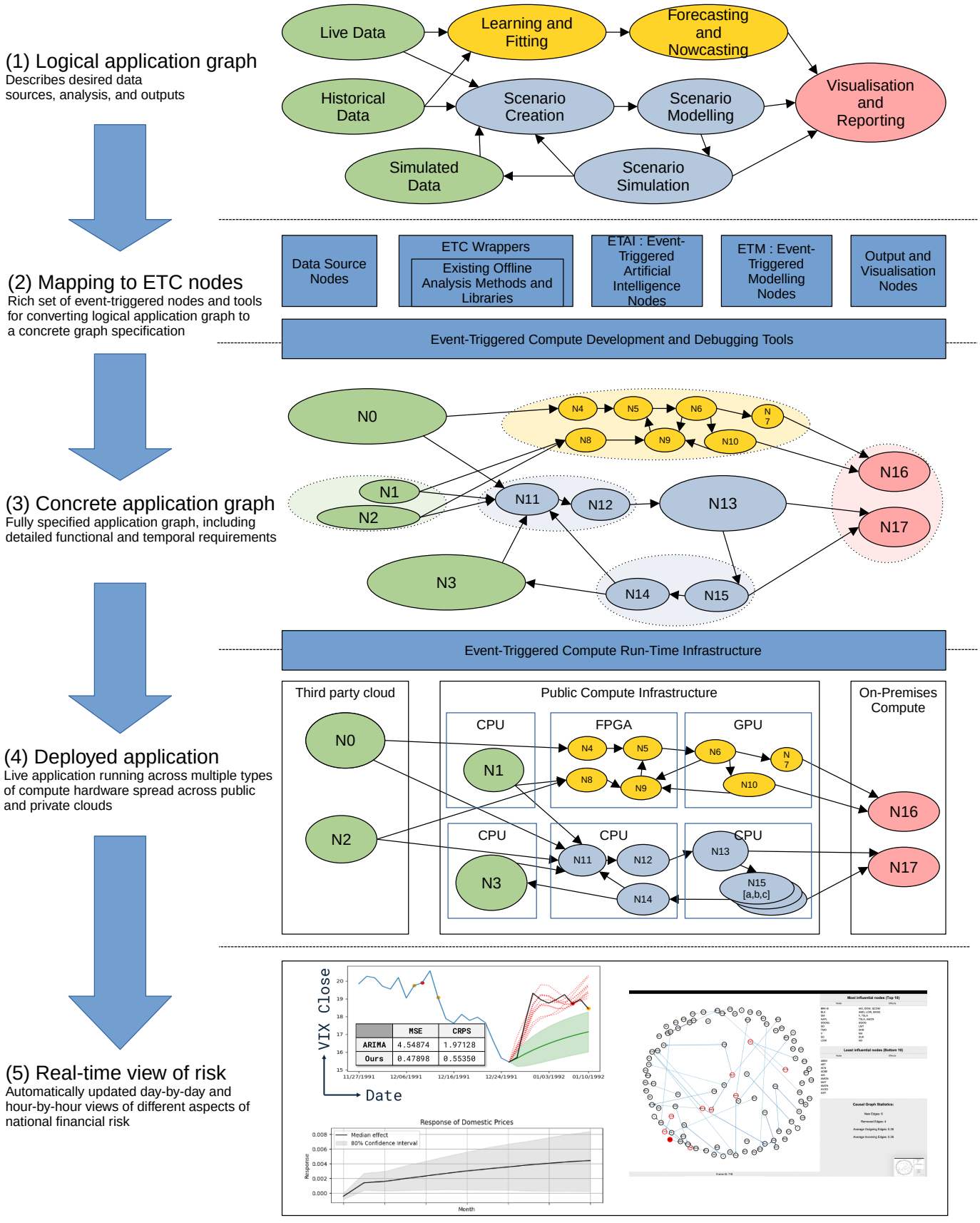


Figure 1. Overview of the SONNETS project, from top to bottom: (1) Logical application graph describing broad desired functionality; (2) suite of ETC nodes providing different modelling and analysis capabilities; (3) concrete application graph containing detailed specification for final graph, including constraints and real-time requirements; (4) actual graph running in the cloud, using a combination of different compute hardware; (5) live updated visualisations of current risk estimates.

Figure 1 gives an overview of the SONNETS flow from abstract functionality at the top, down to live risk visualisations at the bottom. The main stages illustrated are:

- (1) **Logical application graph** : The input is a description of the overall desired analysis system, including data sources, the types of analysis and learning to do, and what sort of results to collect. In the ETC approach an application is expressed as a graph of logical computational nodes, with edges representing the logical channels for messages to move between the nodes. Here the messages could range from a few bytes representing a single change in a stock price, up to multi-megabyte messages representing a trained machine learning model. The "events" of ETC occur when nodes send and receive messages over these channels, which is used both for messaging, and also for identifying and controlling when nodes may change state (see Section 4.1). In the near term our goal is to support static application graphs, but in the final system we would expect to be able to add or remove nodes while the system is running.
- (2) **Mapping to ETC nodes** : Within the SONNETS framework we are developing a large set of ETC nodes, where each node is a composable unit of functionality. This set includes the flagship ETM and ETAI nodes which are designed in an event-first manner, but also includes nodes that wrap existing offline or batch analysis libraries, along with live and historical data sources. A suite of ETC-aware tools support the mapping of the application graph to nodes, including debugging and verification. The mapping process can also add extra run-time requirements such as constraints on maximum processing latency, or the freedom to defer or even eliminate some compute tasks when the system is over-loaded.
- (3) **Concrete application graph** : The concrete graph has all processing nodes bound to concrete execution semantics. Some processing nodes may be refined into sub-nodes, both to expose more parallelism, and to provide more freedom to the run-time in terms of placing processing nodes on computers. The ETC approach has strict but composable semantics for both nodes and overall graphs, which makes it possible to prove equivalence between transformed graphs, and to prove that there actually exist feasible solutions for a given concrete graph. For example, we might discover at this stage that desired real-time updates of once per hour are infeasible given a critical path through the graph of nodes. The concrete graph also captures run-time optimisation opportunities, such as the ability to scale a node up using acceleration hardware (e.g. FPGAs or GPUs), or scaling out by instantiating replica nodes.
- (4) **Deployed Application** : Given a fully specified concrete application graph the ETC run-time will deploy it across a combination of hardware compute resources. This includes public infrastructure such as public commercial clouds, along with on-premises compute for display nodes and proprietary data, as well as third-party clouds which might contain data sources or external black-box analysis. The run-time infrastructure is responsible for allocating nodes to compute infrastructure, and also logical channels to networking and messaging routes. It can also exploit freedom and optimisation opportunities within the application graph, such as deploying to hardware accelerated cloud nodes, or replicating nodes to increase throughput.
- (5) **Real-time view of risk** : For the demonstration systems the goal is to provide real-time intelligence on risk, which ultimately appears as live updating reports, graphs, and analyses to end-users such as regulatory bodies and policy makers. At the bottom of Figure 1 we have included example outputs from analysis approaches - clearly these are too small to read here, but they will be described in more detail in Section 5 and Section 6. The exact nature of the dashboard is going to be highly dependent on the analysis and models chosen for development over the lifetime of the project, but we expect it to include novel ML, modelling and AI methods, alongside more established statistical risk analysis methods.

A natural concern about using more complex ML and AI methods is "*why did it say that, and how confident is it?*", especially when models are innovating rather than interpolating. Our plan is that the ultimate demonstration system will be able to justify the estimates it makes in a number of ways, providing information about where predictions came from. This ranges from standard methods such as presenting historical responses to similar scenarios, through to providing automatic English language explanations of the inference process.

Traditional risk estimation and nowcasting techniques rely on estimating models and re-estimating them as new data comes in, which may be computationally expensive, particularly when scaling to more inputs. This problem is compounded by contemporary ML models, where training and re-training costs can be much higher, with the danger that model re-training time is slower than the desired inference interval. Our plan is that re-estimation and re-learning is automatic and adaptive, avoiding continual re-fitting where it is a waste of resources, but also initiating re-fitting when conditions suggest the model is out of date, including allocating additional temporary compute resources when needed.

Stress testing is another big component of risk analysis, but usually this requires human input both in terms of scenario generation and the actual estimation of risk for a scenario. Our vision is that both scenario generation and risk estimation are an embedded part of the end-to-end risk analysis approach (System-B). While end-users will still find it useful to ask "what-if?" style questions that consider huge economic movements, we also want the system to continually be generating and evaluating more subtle movements, including those that humans might not consider.

Overall our goal is not to wholesale replace existing approaches and systems, as there will still be a need for human oversight and intuition, and things like large-scale stress tests will still be needed. The daily "digital shadow" that SONNETS provides will be a new complementary tool that provides additional information and explanations, and can also uncover previously unconsidered sources of risk. Ultimately SONNETS is a *decision-assisting* tool.

3.5. Expected benefits

We expect the research outputs of SONNETS to benefit a number of groups in different ways. Here we briefly consider some of these groups, plus some of the specific expected benefits to each.

Regulatory end-users: Regulatory bodies and government agencies responsible for overseeing financial markets and institutions, who require advanced tools to monitor, analyze, and manage financial risks in real time. Benefits include:

- Live analysis: Real-time system with multi-modal data inputs deployable on standard COTS (Commercial-off-the-Shelf) infrastructure.
- Autonomous operation: hands-off continual operation with no manual steps required for re-estimation or re-training.
- Access to novel methods: Access to new algorithms from academia, alongside existing classical methods.
- Explainable results: White-box explanations of inferences and predictions from learnt models.
- Automatic refitting: Continuous scenario generation and co-simulation of models, both of existing fitted models, and to support fitting of new models.

System configuration and management: IT professionals and system administrators responsible for setting up, maintaining, and optimizing computational infrastructure who need reliable and efficient systems with low setup and maintenance costs. Benefits include:

- Verification support: Formal verification of compositional properties such as liveness, deadlock, and response times.
- Debugging tools: Model checking and fuzz testing of interaction patterns to discover and debug failure modes.
- Performance management: Tactical demand-driven scale-up and scale-out of compute to increase performance by switching node implementations and replicating node instances.
- Efficient use of resources: Strategic static analysis to increase efficiency and reduce cost by time multiplexing nodes and downgrading compute resources.

Creators of new analysis algorithms and methods: Researchers and developers (both academic and industrial) focused on creating innovative algorithms and analytical methods, who need environments that support experimentation, testing, and deployment of their work. Benefits include:

- Simplified development: Local lightweight development evironment for testing and evaluation algorithms against multiple data-sources.
- Fast Evaluation: Immediate access to a deployable real-time environment at a number of scales.
- Combined learning and inference: Integration of both training/estimation and inference/prediction into a single framework .
- Increased Impact: Larger impact of their work due to reduced barriers between offline academic code and real-world use.

Developers of optimised and hardware accelerated algorithms: Engineers and developers specializing in creating and selling optimised algorithms for hardware acceleration, who need tools and environments that facilitate development and deployment of their high-performance solutions. Benefits include:

- System integration: Ability to focus on core compute acceleration tasks, with IO handled externally.
- Rapid evaluation: Automatic unit testing and benchmarking of accelerated implementations against existing reference versions.
- Increased Reach: Greater access to potential users due to larger user-base and reduced integration costs.

This is a non-exhaustive list of beneficiaries, and we expect to identify other opportunities as the research progresses.

3.6. Current Progress

This is a long-term project which will be steered by internal and external input over the years, but in order to get started we defined an initial work-plan for the first year. We will first present a brief overview of the overall activities at a project level here, then in the next three chapters will dive into specific activities in each of the ETM, ETAI, and ETC themes.

3.6.1. Over-arching project planning

Our original work plan splits the research into three broad phases:

- Fundamentals (year 1) : development of shared documentation and APIs (Application Programming Interfaces), initial prototyping and research in the three tiers (ETM, ETAI, ETC), and creation of prototype demonstrators (**we are here**).
- Exploration (year 2-3) : concrete goals for this phase are presented and refined in this advisory board meeting. In the first half of the phase the shared APIs are further refined and extended while core research in the tiers ramps up. Generalisation of the objectives based on research progress and new ideas will be reported in the second advisory board meeting, then the phase finishes with refined prototypes targeting the two eventual demonstration systems.
- Execution (year 4-5). the third advisory board meeting identifies overall final objectives for the project, then work proceeds in the same two-year structure: shared API development and refinement, core research in the tiers throughout, and final push for system integration and evaluation of the demonstration systems.

3.6.2. Phase 1 : an overview of what we've been doing

The main goal of our first "fundamentals" phase was to explore how to match the capabilities of traditional solutions using event-triggered solutions, and how to use event-triggered APIs and abstractions to communicate between the research performed within the tiers. The objectives for this phase were partially determined before the research project started, with a strong emphasis on creating a full cross-tier event-triggered platform as the foundation for the two demonstration systems as soon as possible, then leaving space for research within the tiers.

A challenge we knew we would face is the lack of infrastructure for doing real-time financial analysis, such as data sources, batching of data, messaging between components, and so on. All of these things are handled well by (proprietary) in-house and (expensive) commercially available financial analysis systems, but we are trying to create an eco-system that is both open and deployable. So our initial goal was to have something non-commercial that could allow an individual academic to develop an analysis component, test it against historical data, and then deploy it into a real-time system. Within this we then integrated classic off-the-shelf analysis and learning, as well as exploring less conventional algorithms and techniques.

Later sections will give a more detailed technical overview of what we have achieved, and we will present the integrated demonstrators live at the meeting, but we will summarise here some of the main contributions in terms of the integrated goals and the three main themes:

1. Integrated demonstrators:
    a. Multiple classical and new ML based estimation tasks, running over both live real-time data and historical financial data.
    b. Distribution of computational tasks across multiple networked computers in a local cloud.
    c. Online learning and re-learning of a variety of models at different temporal scales.
    d. End-to-end data source through to visualisation processing graphs incorporating all the above features.

2. ETC Infrastructure:
    a. Cross-language and cross-platform APIs and semantics for capturing computational tasks as event-triggered nodes, and assembling real-time applications from multiple nodes.
    b. Local test environment for running individual nodes and entire applications.
    c. Distributed execution environment that can use a number of messaging transports such as TCP and RabbitMQ to execute applications in cloud environments.
    d. Prototype distributed execution environment for automatically deploying event-triggered applications in the cloud using Kubernetes.
    e. Debug and monitoring facilities for understanding the asynchronous execution of event-triggered applications, both locally and in the cloud.
3. ETM Algorithms:
    a. New "causal graph" extraction algorithms, which are able to rapidly estimate causal relationships from distinct time series.
    b. Optimisation of the algorithms to allow them to scale to larger numbers of time series.
    c. Exploration of ways to apply these algorithms to data in real time, including window based and online approaches.
4. ETAI Algorithms:
    a. Application of the existing STRIPE (Shape and Time diverRsIty in Probabilistic forEcasting) ML algorithms for estimating distributions of non-stationary time-series.
    b. New algorithms for history-based time-series extrapolation using a graph-visibility based algorithm.
    c. Tsetlin Machine based learners for categorical extrapolation that can continually learn based on multi-modal input data.
    d. Human-level explanations of model outputs by converting learnt models into logical clauses and sentences.

These achieve the objectives we originally set out for this first "fundamentals" stage. While there are definitely research achievements within these — with papers published (Section 8.4) and more planned — a large component in all three tiers has been building shared understanding and infrastructure for the cross-cutting demonstrators.

3.6.3. Plans for phase 2 : "exploration"

In phase 1 we have built an initial infrastructure for designing, debugging and running ETC applications, alongside a number of novel modelling and learning techniques that can perform real-time analysis within the system. Over the next two years of phase 2 we plan to perform a combination of exploratory research and development across the three tiers, working towards an intermediate combined demonstrator.

Our proposed cross-cutting demonstration system for the end of phase two is an always-on, continually updated web dashboard integrating a variety of novel and classical risk analyses methods refreshed and recalculated at a daily level. This system would be deployed across a combination of commercial and private cloud infrastructure, and automatically scale up or out according to analysis demands (e.g. if

market conditions change substantially), but also scale down most of the time for cost and power efficiency. In order to ensure that we understand the actual problems associated both with deployment and maintenance, we would aim for substantial up-times e.g. measured in weeks at the end of phase 2.

We anticipate two versions would be produced, both due to restricted data sources and Responsible Research and Innovation concerns :

- A public facing version available to anyone and to be used for outreach and promotion, containing a "safe" subset of the analysis of public data and analysis results.
- A private version for team members and stakeholders, which would rely on more speculative or risky-to-release analyses, and might rely on non-public underlying data or other proprietary inputs.

The exact nature of the analyses to be performed will be chosen as part of the research and through guidance from stakeholders and partners, but in line with original plans we would expect to cover the two broad types of risk estimation we originally identified in Section 3.2:

A. Continuous financial surveillance with real-time analysis, with the goal of estimating key financial conditions indices.
B. Automatic generation and exploration of scenarios to identify market movements that could substantially affect key financial conditions indices.

This financial condition indices we monitor are a point to be refined, but an example would be to use the MFCI (Monetary and Financial Conditions Index). From a surveillance point of view this could be a one week ahead daily estimate and confidence intervals of MFCI. From a scenario generation point of view an initial goal would be to identify market changes now which would lead to the largest change in MFCI in a week's time. We also expect to include less traditional forms of surveillance, for example by looking at changes in estimated causal dependencies between multiple time series (Section 5.1).

The exact methods underlying these two types of risk estimation are part of the research, but we would expect to maintain a balance between well-understood and trusted off-the-shelf methods, as well as the cutting edge methods we are developing in ETM and ETAI. For example, methods such as dynamic factor analysis, and algorithms like MIDAS (MIxed DAta Sampling) would sit alongside the new methods we are developing including Tsetlin Machines, auto-encoder methods, and causal graph extraction. Alongside traditional quantitative displays of time series and confidence intervals, a key goal is to be able to display explanations of the analysis, as well as to back-test previous estimates as ground-truth data arrive.

3.7. Conclusion

In this chapter we have been considering the overall SONNETS project and the cross-cutting goals and capabilities. At the advisory board meeting we will show these capabilities in a number of live and/or recorded demonstrations (depending how confident we are feeling!). These combined demonstrations all sit on top of work within the three tiers (ETC, ETM, and ETAI), so in the next three chapters we will dive into research work and achievements within each of the tiers.

4. ETC : Event-Triggered Computing infrastructure

The ETC tier combines fundamental research into specifying, analysing, and creating event-triggered compute applications, as well as creating a concrete deployable infrastructure that can run such applications in real-world compute infrastructure.

The initial ETC goals for the first phase of SONNETS were to:

1. Explore possible semantics for ETC in the new context of cloud-based AI, modelling, and risk analysis.
2. Define APIs and semantics for capturing and exposing nodes written in a number of languages.
3. Create local tools and run-time environments for initial creation, execution, and debugging of ETC applications.
4. Develop a distributed run-time environment for executing applications using modern cloud infrastructure.

We will now introduce the idea of ETC in more detail, then we will discuss how ETC is being developed within SONNETS.

4.1. What is ETC?

Event-triggered computing (ETC) is a new computing abstraction for large-scale event-triggered systems (ETS). In ETC we describe and execute applications as a graph, where vertices are communicating finite-state machines called “devices”, and edges are the logical channels over which devices communicate. We will briefly describe the legacy POETS approach to ETC here, then in Section 4.3 will explain the newer model developed for SONNETS. The left of Figure 1 shows a simple abstract graph, including: device instances D0..D3; “pins” on each device, which are where messages can enter and leave devices; and “channels” describing how messages leaving one pin should be transported to pins on other devices.

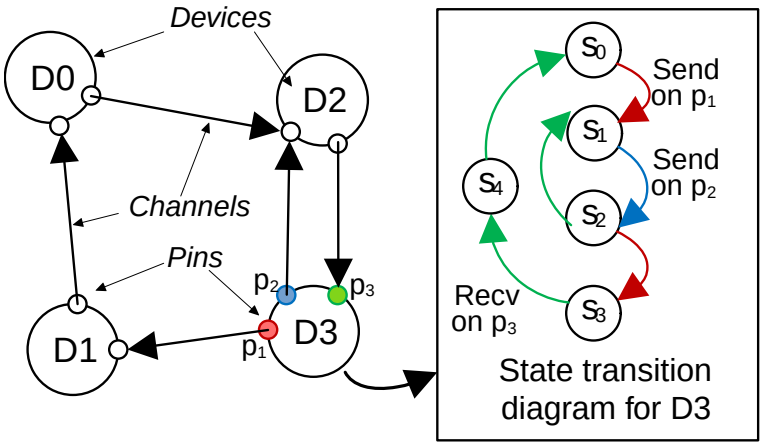


Figure 2. Overview of POETS-style ETC, with a graph of devices on the left, and the logical structure of a device state machine on the right.

Each device is an FSM (Finite State Machine) moving between states in response to “events”, with the two main event types being OnReceive to receive a message, and OnCanSend to give the opportunity to

send a message. The right of Figure 1 shows an FSM for D3, with transitions between the five states occurring when messages are sent/received on pins p1..p3. Messages are asynchronously moved between devices over logical channels, with no guarantees on timing or global ordering of events. A feature of ETC is that sending and receiving of messages is treated on an equal footing: for example state s2 can be left by sending on p1 or receiving on p3. It is up to the local run-time environment to decide when there is enough network and/or compute capacity to prepare and send a message, or if instead it should deliver incoming messages first.

The benefits of ETC are that it forces applications to expose huge amounts of concurrency, while giving substantial freedom in exactly how that is implemented. It also means that applications can be designed and verified against an abstract hardware platform with strict semantics, rather than worrying about implementation details. Mapping a logical application graph onto physical compute nodes then becomes an optimisation problem, which can be re-visited at compile-time, deployment-time, and run-time, with the running application remaining operational even while the underlying hardware and network implementation is being optimised.

4.2. Prototyping ETC systems

A number of initial prototypes were developed, in order to experiment with different ETC APIs, explore user-interfaces, and evaluate existing cloud and enterprise messaging backends.

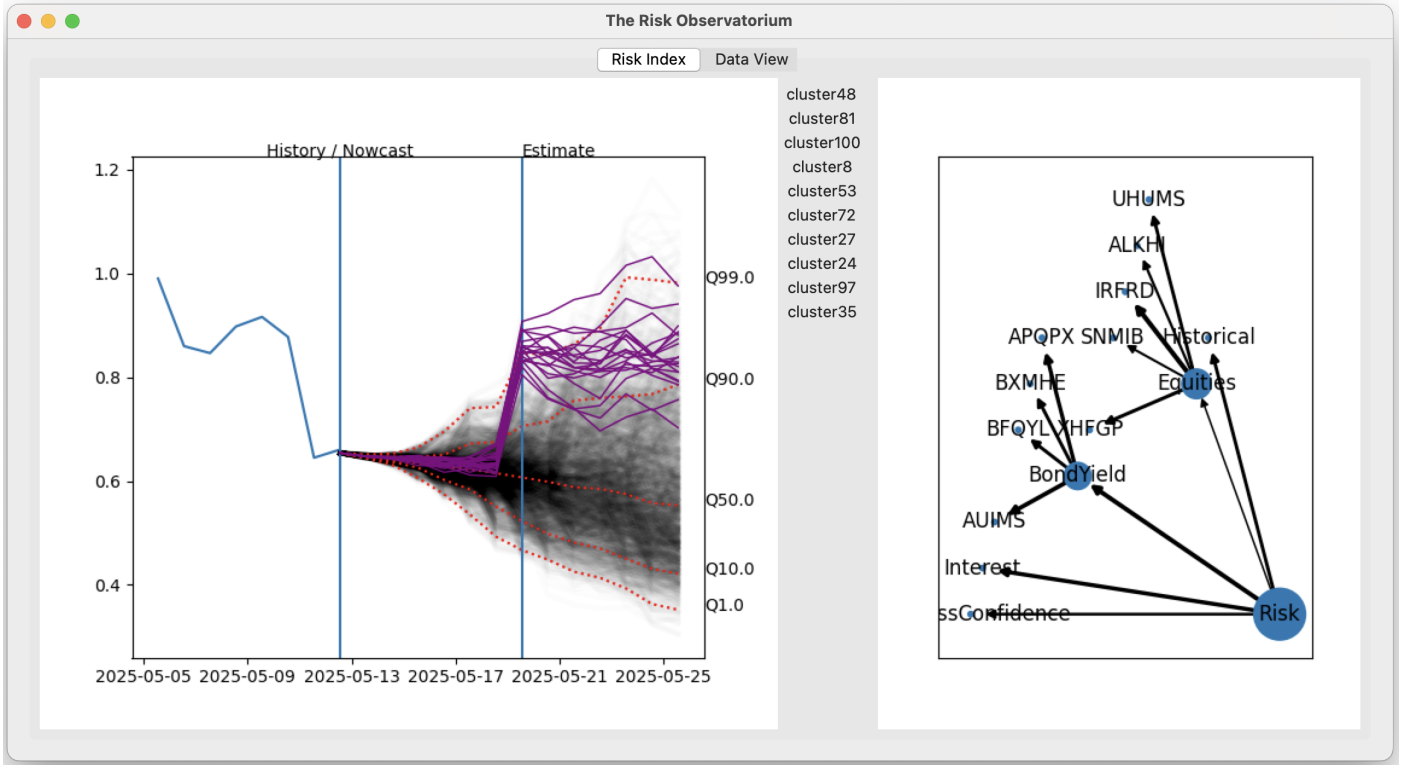


Figure 3. Observatorium UI prototype for interactive exploration of risk and input factors.

One example of prototyping user-interfaces is shown in Figure 3, which shows an interactive mock-up of a risk exploration tool. This was used as a discussion point in the project kick-off meeting, providing an explorable (if fake!) risk modelling capability. The left panel shows the known history of a risk metric over time up until the first vertical blue line labelled "History / Nowcast". To the right of the line we start seeing a "nowcast" for what the risk metric is estimated to currently be, but with uncertainty starting to blur the distribution. To the right of the second blue line the uncertainty has become large enough to be a forecast rather than a nowcast. The panel in the middle lists clusters of simulated paths from different

scenarios — clicking on one selects a subset of forecasts for that scenario, as seen in the cluster of paths which rise abruptly just before the "Estimate" vertical line. The right panel of the graph shows an estimated causal graph (see Section 5.1), representing the dependency structure assumed for that particular scenario. While the underlying data is pure fiction, this type of prototyping was useful for having a shared dialogue within the team about what type of demonstrator we were building towards.

Another prototype called "Pentameter" was created to investigate aspects of ETC infrastructure design, with the purpose of:

- Exploring existing messaging infrastructure for moving messages between nodes.
- Trying out semantics and API styles for integrating nodes into an ETC framework.
- Implementing some realistic nodes and data-flows that represented plausible workloads.

This resulted in a fully working ETC prototype system, including simple market models, financial analysis nodes, and visualisation, all of which could work in real-time. It was designed to be a throw-away prototype, and we did indeed throw it away, but lessons from Pentameter were important in the design of the follow-up Lyrical framework (which we will describe next). Some of the ways in which prototyping helped in the design include:

- Refining the existing ETC ideas to have a specific focus on real-time financial analysis, resulting in changes to ETC semantics compared to POETS around message sending and notions of "time" in ETC.
- Experimenting with API styles for managing the boundary between ETC node implementations and the ETC infrastructure.
- Identifying (and rejecting) candidate underlying messaging systems and tools.
- Finding common problems that arise as we move existing analysis approaches into real-time systems, particularly around assumptions built into many analysis libraries around batch and offline processing of input and output data.
- Working out how we might try to display results, both in terms of updating displays in real-time, and also in how to select and format the data using multiple display modes.

4.3. The Lyrical framework

A year-1 priority in the project work-plan was creating working ETC infrastructure with stable APIs, so that components from the ETAI and ETM layers could start to be integrated into real-time systems. This has been achieved through a system that was initially code-named Lyrical, but now "Lyrical" has come to mean the general set of APIs and tools used to run and debug ETC applications.

Particular requirements for Lyrical were:

- Public ETC APIs which would remain stable in the medium term, with a focus on core mission-critical functionality rather than adding lots of features.
- Ability to run ETC applications locally within in a single node (e.g. a laptop), while still supporting the ability to run the same applications in distributed environments.
- Support for ETC nodes as either processes or a container, particularly to support multi-language applications (separate processes) and difficult to package or incompatible machine-learning dependencies (separate containers).

- Multi-language node development with a consistent cross-language ETC API — our initial targets were Python, C++, C, and Julia, as this provides good coverage for the languages we found for existing analysis modules, and the languages we expect to be using for ETM and ETAI development.
- Support for debugging and tracing applications, including intra-node and inter-node logging.

The specific semantics chosen for the ETC layer are an evolution of those from POETS, and are designed for applications that run in cloud servers, as opposed to the highly constrained bare-metal threads used in POETS. This affected some of the terminology, in that rather than talking about FSMs as "devices", we moved to the more abstract term "nodes". This reflects that a node within a SONNETS graph is much more abstract and mutable — whereas in POETS a device was always a thread on an embedded CPU, a SONNETS node could represent CPU resources, a GPU, an FPGA accelerator or other types of component.

The broad semantics are:

1. Each *node* is a state machine, and *ideally* an FSM, though we do accept that some nodes may no longer be *finite* state machines.
2. Each node corresponds to a single process at run-time, ensuring isolation.
3. State transitions occur when certain events happen, resulting in a call to a corresponding node handler which updates node state. The three main events/handlers are:
    a. OnReceive: a message has arrived addressed to the node
    b. OnTimePoint: a pre-determined point in wall-clock time has been passed
    c. OnCanSend: there is now network capacity to send more messages
4. During state transition handlers nodes can *try* to send messages, but if there is a lack of network capacity they may have to wait, either until there is network capacity (handled via OnCanSend), or a time point is reached (handled via OnTimePoint).
5. Each handler returns a "wake-up" set, which indicates if/when it would like to wake-up. Possible values include:
    a. Empty : Wake up when a message arrives (via OnReceive)
    b. At-Time-Point=T : also wake up when a time-point has passed (via OnTimePoint)
    c. Can-Send-Messages : also wake up when it is possible to send messages (via OnCanSend)

The main differences compared to the POETS formalism are:

1. It is no longer a hard requirement that nodes are strictly finite state.
2. It is possible to (try to) send from within a handler, rather than needing to wait for OnCanSend.
3. A standard definition of time is explicitly embedded into execution semantics via OnTimePoint, due to the importance of absolute and relative time in the target applications. That said, synchronisation between nodes is only maintained via sending/receiving messages, as the graph of nodes remains an asynchronous distributed system.
4. Nodes are now full processes, rather than the POETS devices which were bare-metal threads sharing a processor.

This API is defined using an inheritance based approach, adapted for each supported language, with node implementations inheriting from the abstract API. Node implementations also declare information

about the type of node that they provide, as well as how to instantiate them at run-time — for example, which program should be executed, what input/output pins does the node have, and what configuration parameters does the node support?

As previously shown in Figure 1 an application to be executed is described as a graph of nodes, with vertices describing the nodes to be executed, and edges representing the flow of messages between them. Based on earlier prototyping and internal discussions we chose quite strict initial semantics, requiring that all messages produced will eventually be delivered, and messages between any pair of nodes will be delivered in order. However, there is no guarantee on timing of delivering, nor of the ordering/interleaving between messages arriving from different nodes. Channels are also assumed to have some unspecified but fixed capacity in order to avoid fast producers flooding channels — this was a particular lesson from the prototyping stage in order to maintain real-time behaviour.

The Lyrical prototype was built around these semantics and APIs, with the goal of having as few dependencies as possible to make it cross-platform and easy to install. For example, the underlying network uses either pipes or TCP, messages are formatted using JSON-lines, and node instantiation is handled by directly launching processes. To avoid coupling node implementations to the underlying ETC run-time, an intermediate "ETCLib" is dynamically injected into the process when it is launched. This allows node implementations to be agnostic to the underlying messaging approach and logic of the run-time, allowing messaging to be modified and for new infrastructure features to be injected.

Features of this system include:

- JSON-Schema definitions of node meta-data, supporting syntax checking and auto-complete when creating meta-data.
- Visualisation tools for viewing application graph structure.
- Extensive automated system integration tests testing individual nodes and graphs of nodes.
- Manual multi-computer deployment using SSH to launch remote nodes.
- Logging features to optionally extract exact execution graphs, including handler executions and causal dependencies.
- Monitoring tools to view the real-time CPU utilisation, message rates, and handler execution times of nodes in running graphs.
- Debugging tools for visualising handler execution graphs and the dependencies between handler executions across nodes.
- General purpose source, sink, and utility nodes, including the ability to replay historical data, extract live data from online services, and to interactively plot different types of output date.

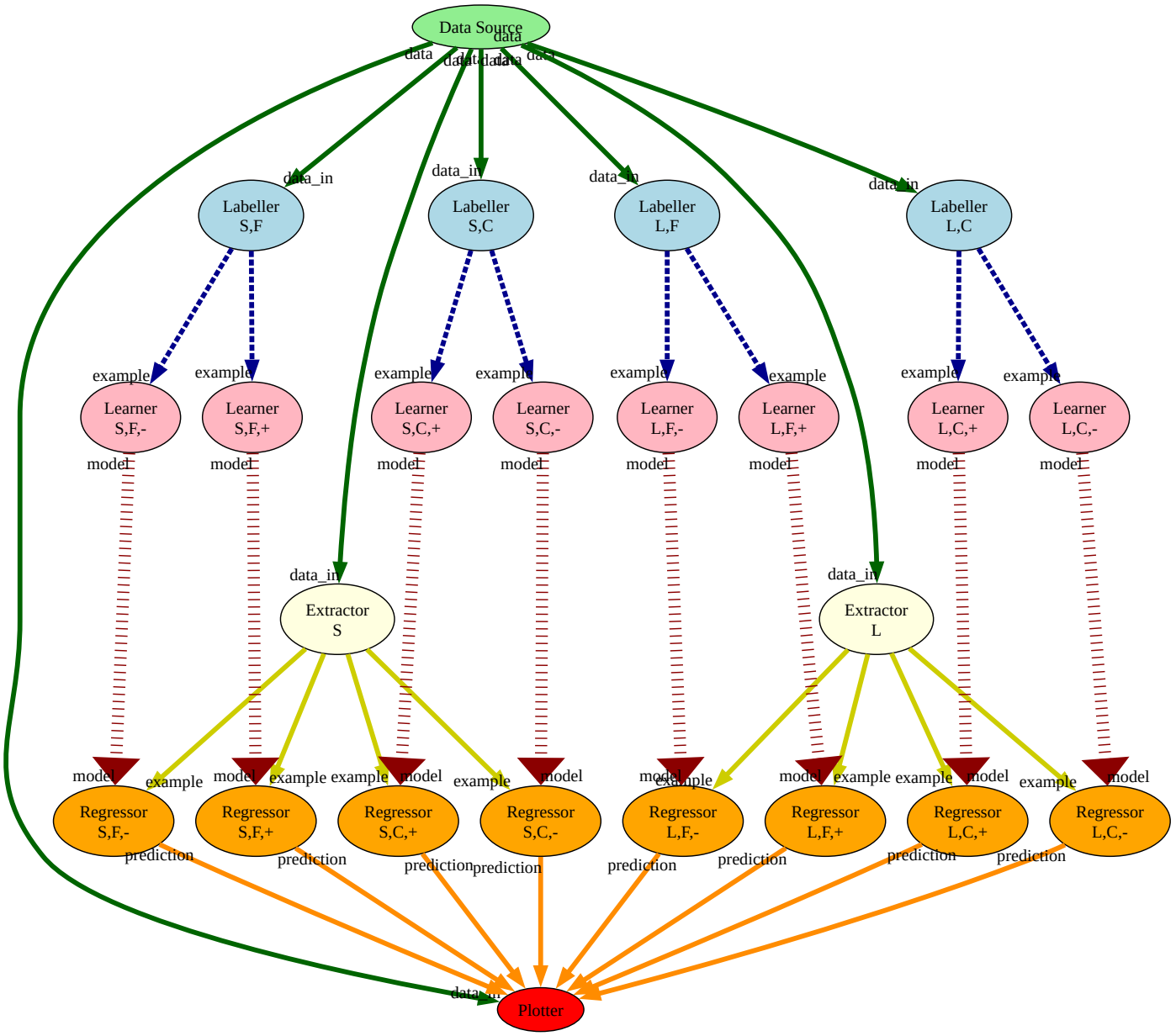


Figure 4. Example real-time learning and inference application graph

An example of an application used to stress test the system is the application graph shown in Figure 4. This graph contains a number of components including (from top to bottom):

1. A market data source node (green): can use historical market data, live market data, or synthetic data.
2. Labelling nodes (light blue): take market time-series data and builds labelled windows of data.
3. Learning nodes (light red): when sufficient labelled data has been gathered these will train (or re-train) a Multi-Layer-Perceptron neural network, eventually producing a new learnt model.
4. Feature extraction nodes (light yellow): extracts unlabelled windows of live data.
5. Regression nodes (orange): take incoming extracted windows and apply the most recent model to produce a prediction.
6. Plotting node (red): takes the live incoming data and the incoming sets of predictions from different models, and displays them all together.

This particular graph applies multiple learning models over different time-scales and features. The time scales used in this example are:

- "Short (S)" vs "Long (L)" : How long a window of time is being used for regression.
- "Close (C)" vs "Far (F)" : The time horizon to make predictions over. This means that we have all combinations of Short/Long vs Close/Far. For example, the left-most blue labeller node is extracting "S,F", which means each labelled piece of data uses a short window of observed data, but then is labelled with the known value at a point far in the future.
- "High Effort (+)" vs "Low Effort (-)" : We can apply different amounts of effort in the learning nodes, which provides different balances between the recency of the model vs the quality of the model fit.

These different configuration parameters result in very different loads in terms of event rates (i.e. frequency of messages) vs compute time (how long handlers take to execute).

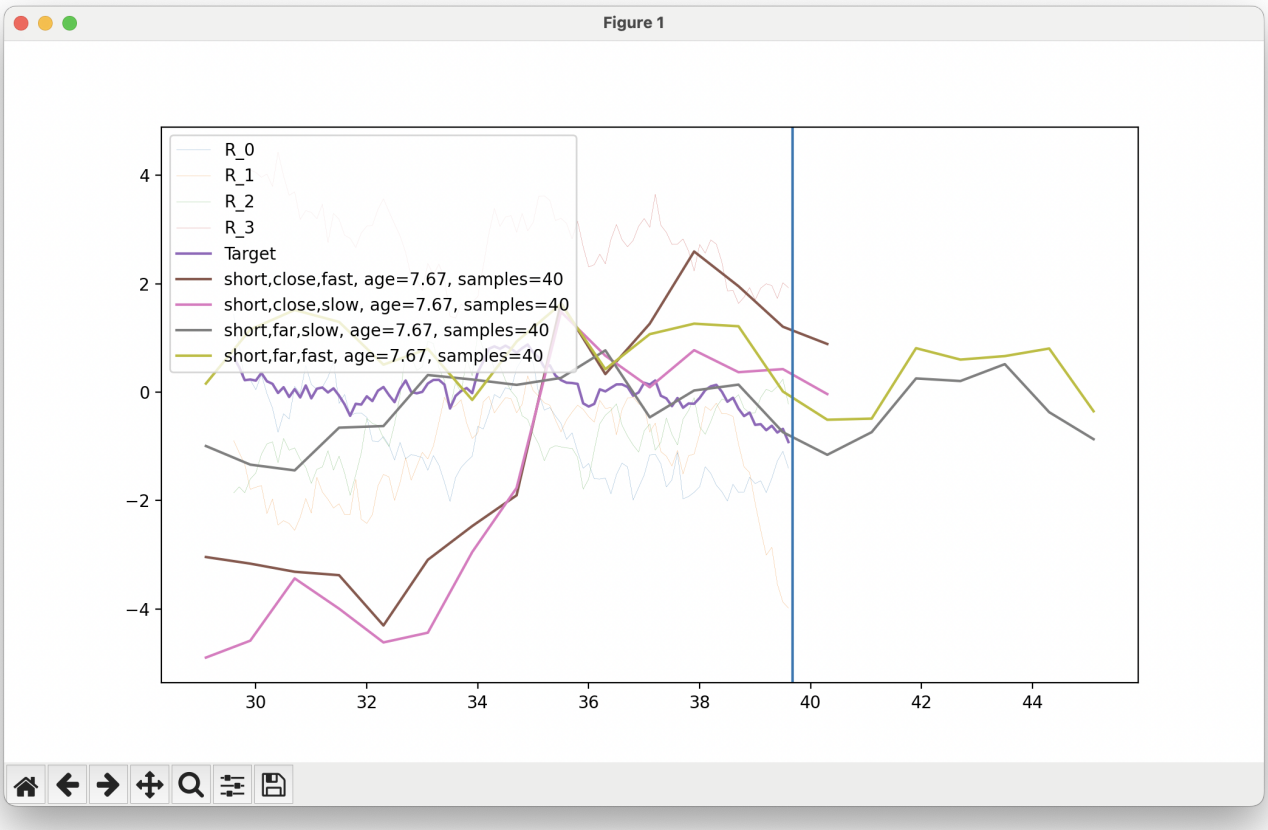


Figure 5. Screenshot of a real-time learning and inference demo application, with historical data to the left of the blue line, and predictions from different models to the right.

The visualised output of the system is shown in Figure 5, which is continually updated as the system runs. The blue vertical line shows the current time, with historical data for four synthetic underlyings (R_0..R3) and a target index (Target) shown to the left of the line, all providing one update per second. Predictions from learners operating with different window lengths and prediction horizons appear to the right of the blue line, then as time passes they slide left until they are to the left of the blue line and can be compared with the actual target data. The generic neural network learners in this example are not very well tuned for time series analysis, so in this case we can see significant deviation between reality and predictions.

Each learner operates independently, and is triggered when it receives enough new data to update its model. The associated regression nodes continually produce a prediction based on the most recent model. Different nodes within the application operate on very different time-scales, with the learners sometimes taking 100s of seconds to re-learn, but the overall graph still operates in real-time, with regression nodes providing updates on a second-by-second basis using the latest data.

The initial Lyrical prototype was developed and documented, and then introduced to the wider SONNETS team at the six month project meeting. Since then it has been used as the main method for developing and executing event-triggered AI and modelling, while in parallel more scalable and robust ETC infrastructure is being developed.

4.4. Migration to scalable network architectures

The initial Lyrical prototype relies only on files and sockets for communication, as it is intended to support lightweight local development of nodes with minimal dependencies. An inevitable consequence is that it suffers from performance bottlenecks, scalability and reliability issues, so a more robust messaging layer is needed for full scale deployment. During the initial prototyping stage we explored a number of off-the-shelf messaging frameworks and protocols, and from these decided to use RabbitMQ. This is a well-known distributed message broker, enabling structured message routing through exchanges, queues, and bindings. The migration involved setting up RabbitMQ for a given application graph, wrapping ETC nodes as consumers and producers, and ensuring message acknowledgment and persistence. A key concern, and part of the value of the ETC approach, is that application code should be completely unchanged as we modify the back-end — a goal that was achieved.

Abstract messaging graph : As shown on the left of Figure 6, the logical ETC graph uses a variety of connection patterns between nodes, including broadcasting, fan-in, and point-to-point channels. In the local Lyrical environment these are implemented using a combination of point-to-point channels such as TCP, along with internal hidden infrastructure nodes to support multiplexing and broadcasting of messages. This works very well for local development, but becomes impossible to manage and scale up in full cloud environments.

RabbitMQ-Based Messaging : The transition to RabbitMQ, shown on the right of Figure 6, decouples message producers and consumers through an intermediary broker. Messages from the source node are published to an exchange, which routes them to bound queues before being delivered to the destination nodes. This approach improves scalability by allowing multiple consumers to process messages concurrently, ensuring dynamic workload distribution. Reliability is improved through message persistence, acknowledgments, and automatic retries, preventing data loss. In addition, RabbitMQ offers built-in monitoring tools, facilitating real-time tracking of message flow and system health. Its flexible routing mechanisms, including direct, fanout, topic, and header-based exchanges, further optimize event-driven processing. Built-in security features such as authentication, authorization, and TLS encryption provide a secure messaging environment. Migrating to RabbitMQ has improved the efficiency, scalability, and reliability of the messaging system, and supports our ongoing work in moving towards larger scale systems with many more nodes.

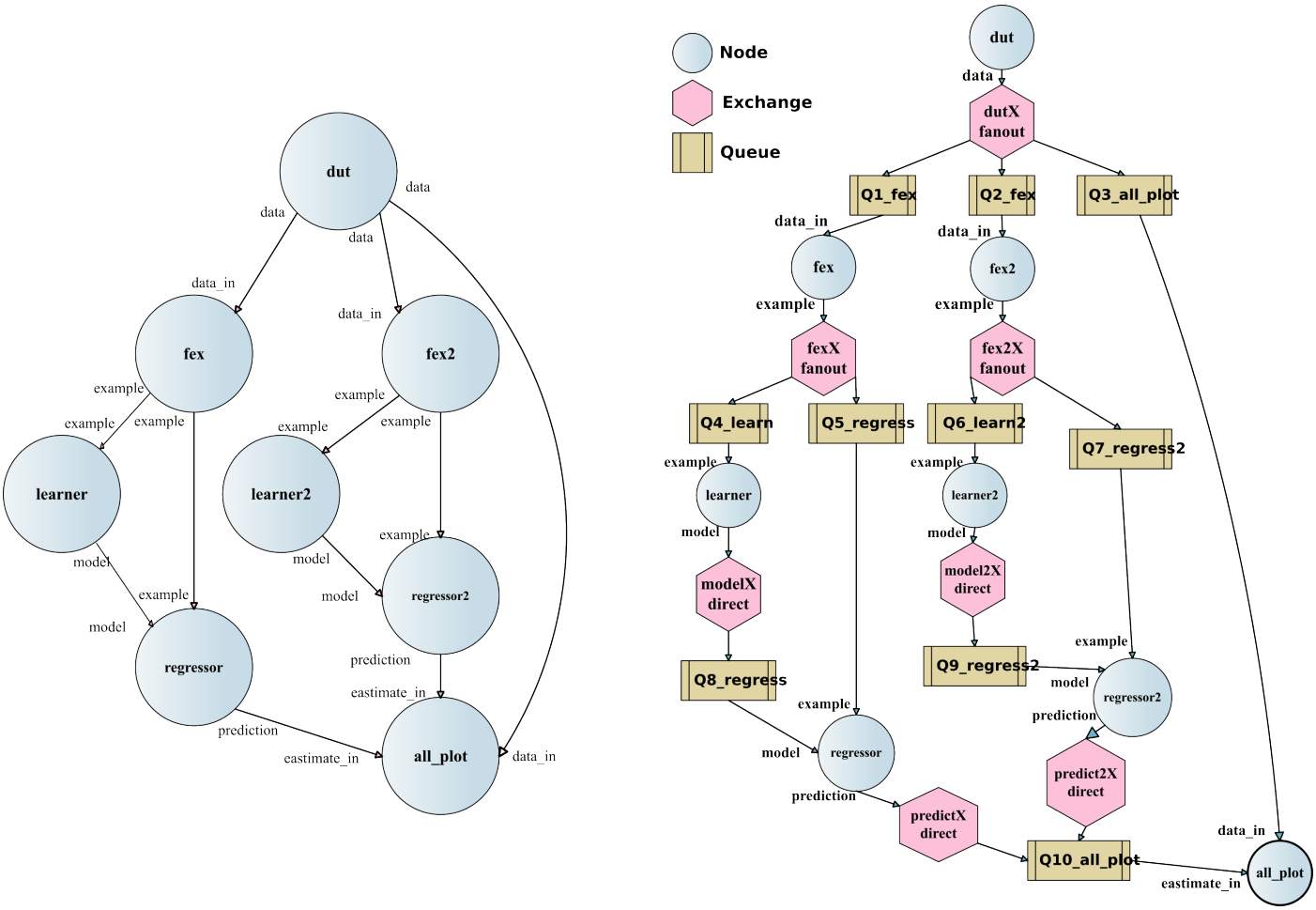


Figure 6. A comparison of an abstract ETC learning graph (left) versus concrete RabbitMQ graph (right). In the RabbitMQ graph the implicit multi-cast and joins have been replaced with a combination of message exchanges and queues, matching ETC semantics while adding performance and robustness.

4.5. Ongoing work : infrastructure scaling and hardening

The existing prototypes and year-one frameworks can only operate on a small scale, up to hundreds of nodes and tens of computers. Such operation is essential for investigating the viability of technologies, and for developing the interfaces through which partners' software will communicate. However, in order to demonstrate ETC at scale in the context of financial risk analysis, infrastructure must be engineered to:

1. Achieve scaling performance sufficient to maintain and support a system that can react to data as it arrives.
2. Support the deployment of huge ETC applications not only in a cloud-computing context, but also on "local" hardware for testing and development purposes.
3. Be robust in the face of adverse incoming traffic patterns.
4. Support test-driven development of ETC applications, and general software engineering practices.
5. Be hardened against security risks, remaining compatible with security best practices from both the network and node (compute) perspectives.

This infrastructure is still under development, but will exist in the form of services and software that allow **developers to engineer production ETC applications** and allow **users to deploy ETC applications using cloud compute resources**, both **at scale**.

Amongst other research and development activities in SONNETS, the engineering of the infrastructure is a sizeable task given the resources available. Fortunately, this infrastructure shares some challenges faced by existing cloud computing software that operates at scale, so many of the tools available for use fit well here. Most notable here is the Linux Container, which is the (de facto) standard mechanism for software deployment to cloud computing hardware. Containers are a mechanism by which a process (analogous to a node, see Section 4.1) can be encapsulated to run under resource constraints and then deployed in multiples to the cloud. Containers are perhaps best known through the Docker container provider (https://www.docker.com/) for running standalone containers, but there is a much larger ecosystem for running and managing containers in cloud and enterprise systems.

The container orchestration system can then assemble one or more computers or virtual machines together into a cluster, which host containers using a task-based parallelism paradigm. Again, there are good existing tools for running and managing containers across distributed systems, with the best known being Kubernetes (https://kubernetes.io). Given a desired set of container instances to run, Kubernetes will allocate compute resources from within a pool of available computers, and start the containers within the allocated resources. In our case we are using containers to host individual ETC nodes, and translate the abstract set of ETC nodes from the application graph into a concrete set of containers that Kubernetes can then run.

SONNETS infrastructure consists of the following connected components:

- **Continuous Integration and Deployment (CI/CD)**: In response to a developer's modification to the code of a SONNETS ETC application, the CI/CD software will run software tests and produce ETC executables for each type of hardware device, if the tests pass. These executables and other metadata are passed to the Provisioner to build container images. A reduced version of this workflow is shown in Figure 7. CI/CD is also responsible for running integration-level tests using outputs from the Provisioner.
- **Provisioner**: Takes a set of ETC executables and creates a new container image for each type of hardware device in an ETC application. The Provisioner uses Ansible (to run commands) and Vagrant (to build the container). Once the container images are created, they are added to a private online collection of container images that can be connected together using Aria (see next point), or a container orchestration system directly. A reduced version of this workflow is shown in Figure 8.
- **Aria**: Given a set of container images (which may be created by the Provisioner), a new SONNETS run-time component called Aria deploys, starts, and stops containers by interfacing with container orchestration software. Aria also supports introspection of running containers using Inspectors: these are special containers that act as an adapter between two "applications", analogous to a packet sniffer, with their own inspector-communication mechanism. Inspectors may:
  - Batch and compress traffic, and forward it to a persistent filestore for diagnosis,
  - Present a user interface on a particular output port,
  - Provide telemetry on how the outgoing traffic pattern of certain containers, which can be queried through Aria,
  - Detect and report unusual traffic loads.

These three tools work together to transparently build and execute ETC applications for developers and users at scale.

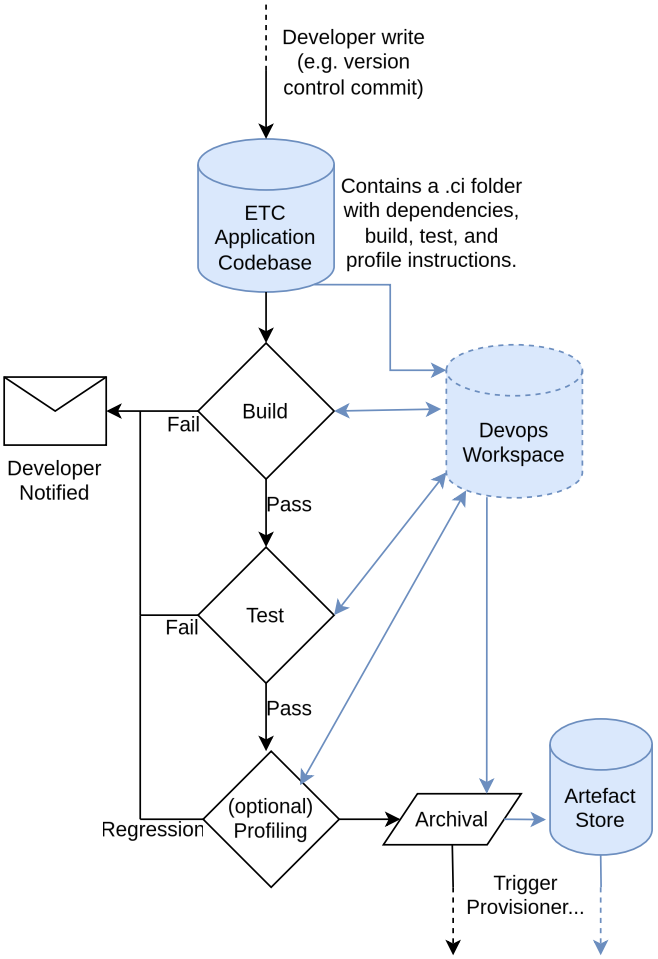


Figure 7. Workflow of the testing CI/CD process for one component of an ETC application. Black and blue arrows represent control and data flow respectively. The output is a test report and, if successful, a set of artefacts representing the built component.

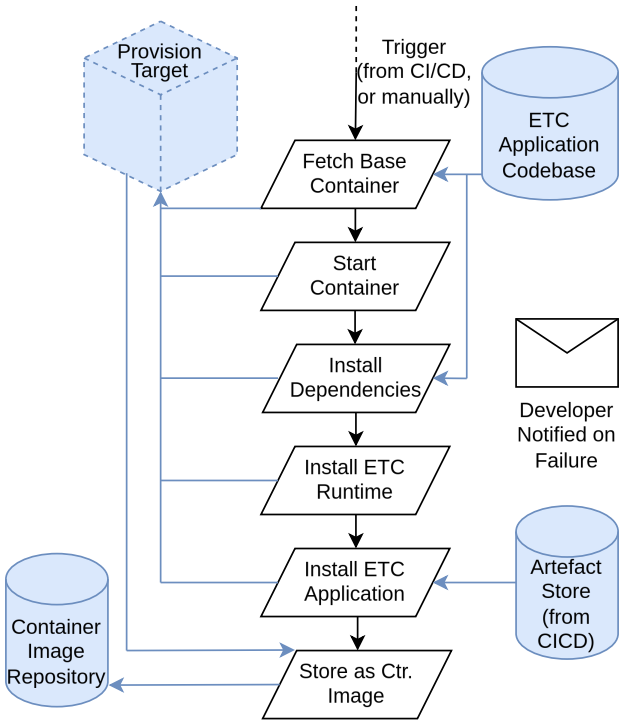


Figure 8. Workflow of the provisioner, which produces one deployable container image of an ETC device, given artifacts from the CI/CD workflow. Black and blue arrows represent control and data flow respectively. The output, if successful, is a container image that can be deployed by Aria.

Currently, the provisioner uses Vagrant and Ansible to manage and provision containers hosted by "containerd", and to produce container images, but the integration with ETC applications and the ETC runtime is not yet implemented. Example "hand-built" containers are used for exploratory development with container management systems, of which Kubernetes has been the primary development target. Short term development targets include a deeper understanding of the networking stack underpinning Kubernetes, and an implementation of Aria.

4.6. Future work

With the basic ETC infrastructure in place there are a number of capabilities which we are considering adding over the next two years, in addition to those already described. These include:

- A richer set of messaging backends, allowing different channels to be mapped to the best backend.
- Annotations on nodes and channels to capture execution guarantees and required constraints.
- Formal checking of application graphs and annotations to prove properties such as liveness and ability to meet deadlines.
- Model-checking of potential node execution orderings in order to discover functional and non-functional bugs.
- Traffic-monitoring inspectors will be engineered and deployed using Aria.
- CI/CD workflow for the provisioner, including integration-level tests using Aria for "exemplar" ETC deployments.
- Automatic duplication of congested nodes to transparently remove execution bottlenecks.
- Support for hardware accelerated nodes, to support GPU and FPGA implementations of optimised ETAI and ETM nodes.

Another goal is start augmenting functional testing and evaluation with non-functional aspects, as so far the emphasis has been on making sure ETC systems run correctly. In the near future we plan to add extensive performance tests as part of our standard integration testing suites, and starting to track performance metrics such as achieved bandwidth, latency, and other metrics. This is expected to result in a near-term paper which considers the achieved performance metrics for a large set of ETC applications when running on different underlying backends and with different amounts of resources.

5. ETM : Event-triggered Modelling for financial analysis

Time-series causal discovery is a set of methods for identifying directional, cause-and-effect relationships among variables that evolve over time. Rather than measuring simple correlations, these techniques examine how past values of one series help predict future values of another, while accounting for the joint dynamics of all series under study.

Causal graph analysis complements these discovery techniques by representing the inferred relationships as a directed graph. In such a graph, each node corresponds to a random variable, such as an interest rate, equity index or credit spread. Each directed edge indicates a potential causal influence. This visual and mathematical structure enables clear interpretation of how variables affect each other over time.

5.1. Overview of Causal Graph analysis

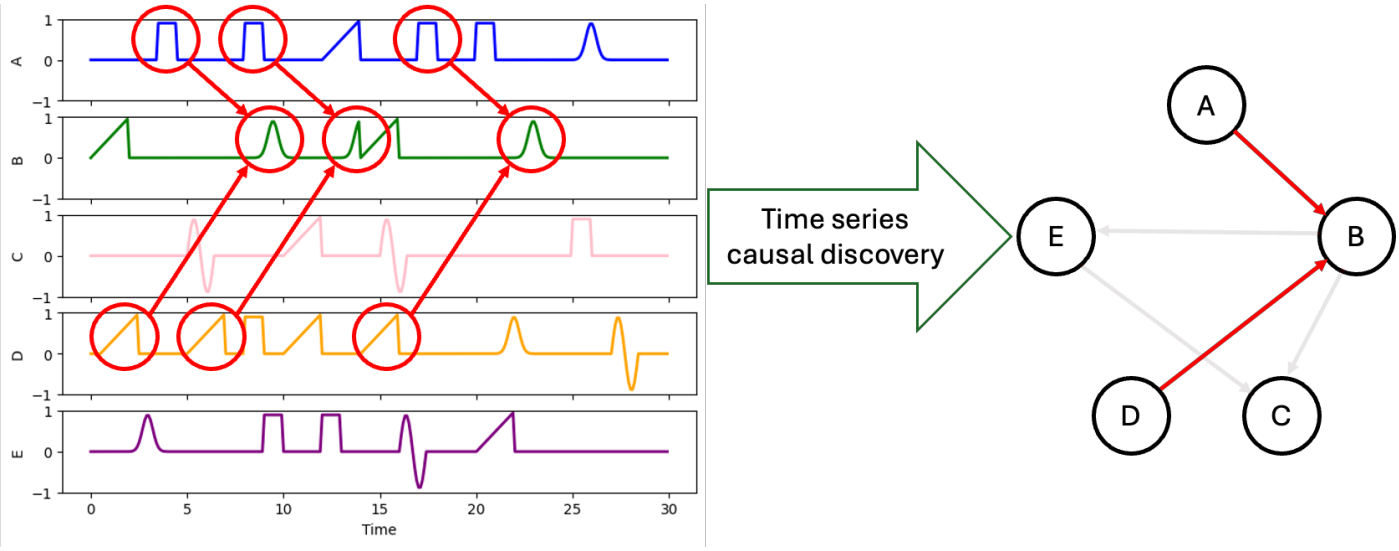


Figure 9. Example of causal graph extraction, with input time series on the left, and extracted causal graph on the right.

A causal graph is extracted from multiple time series, and captures the estimated causality in terms of nodes (not to be confused with *ETC* nodes) and edges:

- Nodes : Each causal graph node represents a variable of interest that varies over time. These could include equity indices (e.g. the FTSE100 stock market index), bond yields (e.g. interest rate on UK Treasury Bonds), a foreign-exchange rate, a volatility measure (e.g the VIX index that estimates uncertainty in future stock prices), or another risk factor.
- Directed Edges : An edge from Node A to Node B indicates that past movements in A helps to predict future movements in B, and the weight on the edge estimates the size of the relationship. If no arrow exists between two nodes, or the weight is very small, then one series offers no additional predictive power for the other.

Figure 9 gives a simplified example of the causal graph extraction process. On the left side are multiple input time series, which could be regularly sampled time-series, discrete event series, or combinations of both. Causal graph extraction looks for events within those time series, and then tries to estimate

which earlier events usually "cause" later events. On the right of the figure is a causal graph, which estimates that events in B *appear* to be strongly caused by earlier events in A and D.

The causal graph allows us to identify which instruments or indicators are likely to drive changes elsewhere. In particular, the causal links can show where risk can concentrate and suggest which variables need closer monitoring or deeper investigation. This mapping from multivariate relationships to a graph simplifies the tracing of potential contagion channels, the design of targeted stress-test scenarios, and the execution of scenario analyses.

Traditional risk analyses often focus on correlations between variables, but modelling correlations does not distinguish between merely related events and real drivers. Causal discovery and causal graph analysis goes a step further by revealing directional links between variables over time. There are three potential benefits for financial risk management:

1. Modelling Shock Propagation through a Financial Network : A causal graph treats asset prices, indices and economic indicators as nodes linked by arrows that show influence. Estimating this graph from historical time series, such as equity prices, interest rates and commodity indices, allows analysts to trace how a shock in one variable, for example a sudden rise in government bond yields, can spread to others like banking stocks or currency pairs. It also supports stress testing by simulating extreme events to reveal potential loss pathways. Finally, it highlights the most critical variables, enabling regulators and risk teams to focus monitoring efforts where they matter most.
2. Enabling Counterfactual Scenario Analysis : We can perform counterfactual analysis on a chosen node, for example by increasing oil prices by 10 per cent, and computing the resulting changes in downstream variables. This approach allows quantitative scenario planning, providing precise estimates of impact magnitudes across multiple series. It strengthens decision making by forecasting the effects of policy actions, geopolitical events or strategic moves before they occur. As new data arrive, the model can be updated and scenarios re-evaluated quickly.
3. Identifying Likely Causes of Observed Market Behaviour : When unusual patterns arise, such as a sudden spike in currency volatility, causal graphs can suggest the variables most likely to have caused the change. This facilitates investigation by directing analysts' attention to a small set of candidate drivers instead of testing dozens of hypotheses. It improves attribution by providing clearer explanations to stakeholders and regulators about why risks are formed. Over time, as more data become available, the model refines its causal links, helping analysts to detect new market drivers' behaviour at an early stage.

Some of the opportunities for risk analysis that we see in SONNETS are:

1. Real-time causal discovery and structure change detection with an accelerated computational system : By designing optimized algorithms and parallel processing systems, we identify evolving causal links among financial time series as new data arrive. Sudden changes in network topology such as the emergence or disappearance of key edges are flagged immediately. This allows risk teams to adapt strategies without delay.
2. Generating causal features for machine-learning models : From causal graphs we can calculate derived quantitative metrics which can serve as input features for predictive models. These could range from generic graph metrics like node centrality which estimate which nodes are closest to some notional "centre" of the graph, through to causal-graph specific metrics like causal strength. These features enrich standard datasets with information on dynamic interdependencies and can improve predictive accuracy.

3. Building data-driven simulation systems for scenario analysis : By encoding causal relationships into a simulation engine, the system can simulate the impact of hypothetical interventions such as interest rate changes or commodity price shocks on a network of variables. For example Figure 10 shows a snapshot of a shock transmission simulator based on a causal graph.

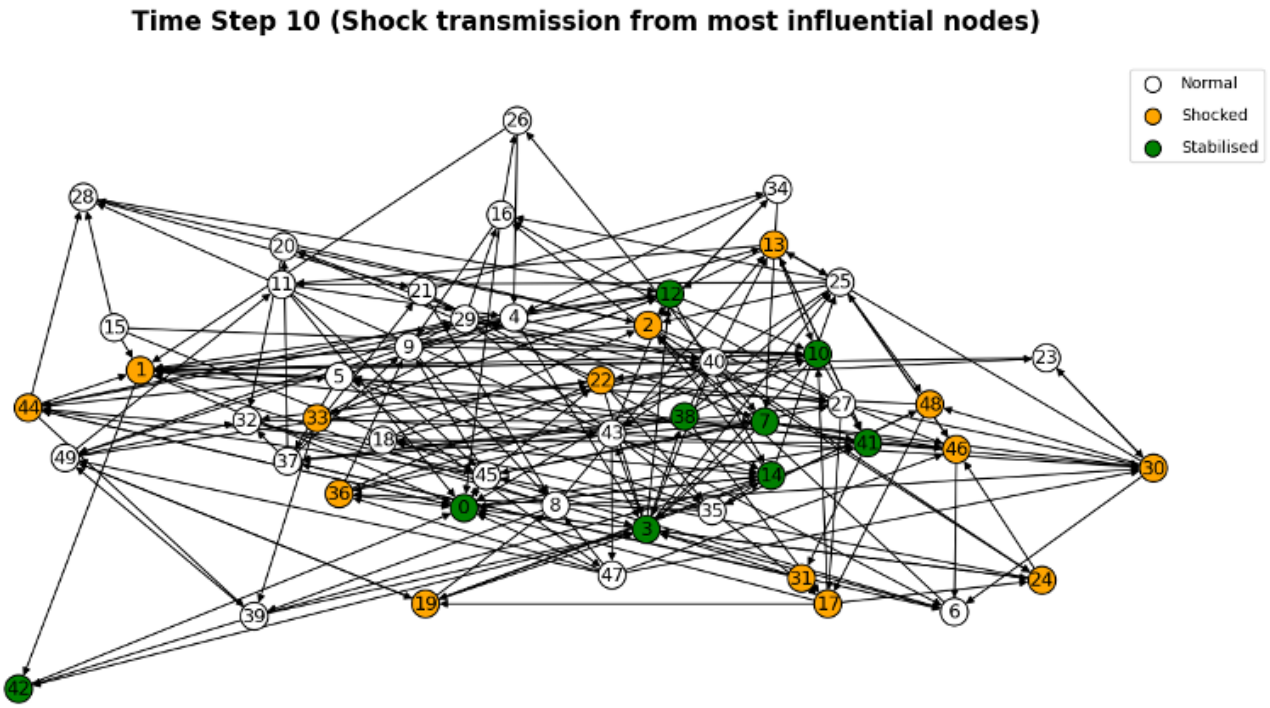


Figure 10. Snapshot of shock transmission through a network of variables over time, using a causal graph to track the propagation. An initial shock earlier started propagating through the network, and at this time step the green nodes have already received the stock and stabilised, yellow nodes are currently processing the shock, and white nodes have not yet experienced the shock.

The challenges for causal discovery in the SONNETS project include the following:

1. Data reliability challenge : Financial and economic time series are often sparse, noisy or irregularly sampled. This can cause causal discovery methods to infer links that vary across data subsets. Without verification across multiple samples, networks may include inconsistent or unreliable connections. The lack of ground truth in real datasets further complicates the evaluation of algorithm performance.
2. Scalability challenge : Most discovery methods scale poorly as the number of variables increases, making exhaustive searches or conditional-independence tests infeasible for thousands of series. Leveraging dimension reduction, sparsity priors or parallel architectures is essential to approach real-time performance without sacrificing accuracy.
3. Expert validation and interpretability challenge : Automated graphs can reflect spurious links or hidden confounders. Embedding expert review, economic theory alignment and transparent visualisations ensures that inferred relations are both statistically robust and economically credible.

5.2. Technical improvements to Causal Graph analysis

Our technical improvements of causal discovery and analysis include the following, addressing the three challenges in the previous section:

Robust consensus-driven (RCD) framework for better use of data. Our new Robust Consensus-Driven framework improves the reliability of causal links by applying the chosen discovery algorithm to multiple overlapping subsets of the original data. Each subset yields its own causal graph. Links that appear in many of the graphs with similar sign and magnitude are retained, while those that vary substantially are removed or down-weighted. This approach is equivalent to conducting a stability test on every inferred relationship, ensuring that only those supported under different sample splits form the final network. Tests on both synthetic and empirical datasets demonstrate that this procedure increases detection accuracy, particularly when data are noisy or display changing volatility regimes. The following table shows the accuracy improvement of the proposed method compared to two base algorithms, PCMCI and VarLiNGAM:

Dataset	PCMCI	RCD-PCMCI	VarLiNGAM	RCD-VARLiNGAM
fMRI	0.63	0.64	0.61	0.57
Antivirus	0.30	0.33	0.33	0.43
MoM	0.48	0.52	0.45	0.54
Web	0.45	0.46	0.44	0.51

This improvement addresses the data reliability challenge by reinforcing only those causal links that persist across multiple overlapping samples, thereby producing a more stable and trustworthy network even when observations are limited or noisy.

Optimised VarLiNGAM via precomputation. VarLiNGAM is an existing algorithm for extracting high-quality causal graphs from panels of variables (i.e. collections of time series), but it incurs very high compute cost when applied to large panels of variables, because it repeatedly recomputes entropy-based measures during the search for causal order. Our optimisations reorganise this work by calculating all variable- and residual-level entropies once at the outset. These precomputed terms are stored and then retrieved in constant time during the causal-order search, reducing the overall computational complexity. The following table summarises the execution times and speed-ups compared with the original CPU-based VarLiNGAM and a state-of-the-art GPU accelerated version :

# Vars	Orig CPU (s)	GPU (s)	Opt CPU (s)	Speed-up (Orig→Opt)	Speed-up (GPU→Opt)
25	4.03	8.83	1.88	2.14	4.69
50	27.31	32.89	8.70	3.14	3.78
100	230.06	168.04	44.54	5.17	3.77
200	1660.57	1030.17	226.80	7.32	4.54
400	21404.76	7291.09	1601.80	13.36	4.55

This method addresses the computational scalability challenge by reducing computational complexity and enabling the same hardware to analyse larger datasets much faster, making near-real-time causal updates feasible.

Unified framework with structural vector autoregressions (SVARs). SVARs are perhaps the most widely used tool by economists for conditional forecasting, a critical feature of scenario analysis. They provide a coherent framework for analysing and interpreting the impact of economic shocks, notably through impulse response functions (IRFs) and historical decompositions. However, economists typically rely on theory for identifying SVARs, that is estimating contemporaneous causal relationships, by fixing either the causal order or the directions of causality among the variables used. Causal discovery methods like LiNGAM provide a purely data-driven approach for identifying SVARs as well as a novel way to visualise the resulting economic dependencies. Constructing a coherent framework for exploiting causal discovery methods in SVARs offers a very promising avenue for scenario and risk analysis.

5.2.1. Causal graph analysis for risk

Causal graph analysis has barely been used in financial risk management. One early study in 2020 showed how you can take stock prices, test whether past movements of one share help predict another, and then build a network of these “cause-and-effect” links. By comparing the network at different stages of the 2007-08 crisis, it became clear which companies and sectors were driving stress and which were absorbing it. Beyond this proof of concept, no practical system yet exists to plug causal graphs into live risk monitoring, stress tests or automated feature building. That gap represents a real chance to turn these ideas into tools that help forecast and manage future risks.

The following simplified example helps illustrate the benefits of combining SVARs with causal discovery methods. The Bank of England’s main objective is to maintain price and financial stability. Subject to this, it can support the economic policy of the government, such as economic growth. Economists at the Bank are therefore interested in questions such as "*What happens to inflation and economic growth if the Pound Sterling depreciates by 5%?*". To tackle this question we model the nominal broad effective sterling exchange rate, real economic growth, domestic prices (CPI index) and import prices (Import PPI). Figure 11 displays impulse response functions from a purely data-driven SVAR estimated with LiNGAM where the object of interest is a 5% negative shock in the pound sterling variable. Import prices increase on impact and then reduce over the year, while domestic prices are not affected immediately but then slowly increase through the year. Economic activity sees a 2% decline after 4 months, recovering somewhat to a 1% reduction in economic growth after one year.

Our causal-graph approach differs from traditional methods. First, rather than relying on expert-led techniques with extensive domain knowledge and manual scenario construction, it is entirely data-driven and requires only minimal domain assumptions. This reduces bias and uncovers unanticipated risk drivers. Second, once the causal graph has been computed, what-if analyses proceed by simple propagation of interventions through the network rather than by running computationally intensive Monte Carlo simulations. Once the causal discovery stage is complete, large numbers of counterfactual tests can be executed in milliseconds, providing extensive scenario coverage without the heavy computing demands of conventional simulation frameworks.

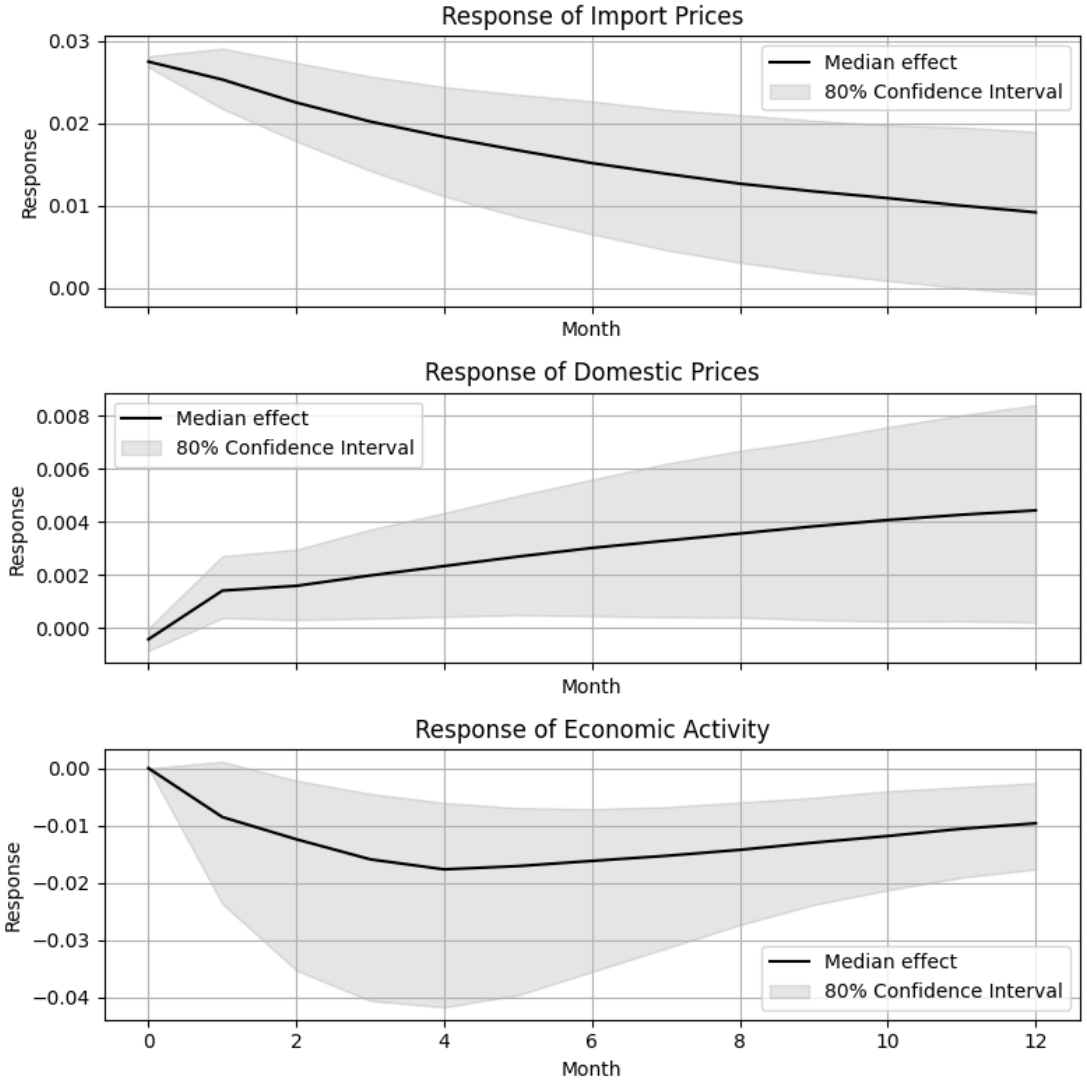


Figure 11. Impulse response of economic activity to 5% negative shock in Pounds Sterling, estimated using SVAR and LiNGAM.

To integrate causal graph analysis into a real-time risk system, we employ two strategies.

- The first strategy uses a sliding window approach. In this setup, the system continuously gathers new data and maintains a fixed-length window of the most recent observations. Optimised causal discovery algorithms then process each updated window as it arrives. By taking advantage of parallel processing and precomputed summaries, the system can re-estimate the causal graph for every new window of data without re-analysing the entire history.
- The second strategy relies on incremental, or online, causal discovery algorithms. These methods update the existing causal graph each time one or more new data points arrive and apply a forgetting mechanism to gradually downweight or remove the influence of older observations. Rather than rebuilding the entire network, the algorithm adjusts only the affected parts of the graph, adding or removing edges as needed. This approach minimises computational overhead and supports continuous learning, so that the system can rapidly incorporate emerging patterns and discard outdated relationships.

These strategies address the Computational Scalability challenge in Section 5.1.

Some problems that have to be solved as we move from a batch to real-time system include: (a) Finding appropriate parameter values for the above strategies, such as window length for the first strategy and number of new data points for incremental update for the second strategy, and what to do if such parameter values may need to be adapted over time. (b) Comparing the accuracy and other metrics of the proposed real-time approach against current batch approaches, exploring possible trade-offs between accuracy and response time.

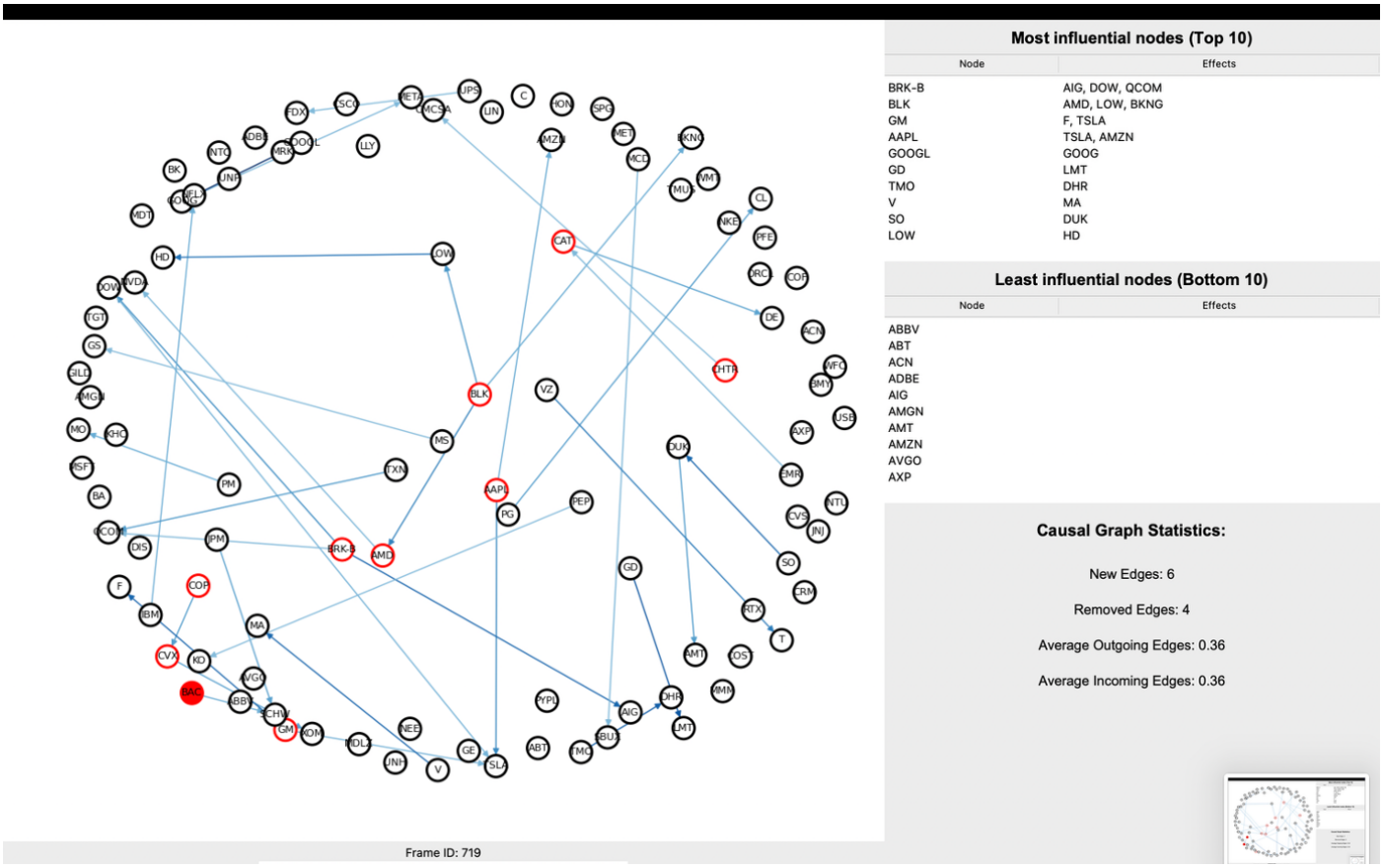


Figure 12. Example of causal graph analysis within an ETC application, applied to a portfolio of stocks. The current causality graph is shown to the left, along with statistics to the top right. Changes to the causality graph over time are summarised in the bottom right.

The causality extraction approach has been embedded into the ETC approach in order to allow for real-time processing of incoming data. Figure 12 gives a screenshot of a live causal graph analysis tool, which is able to analyse and display the updated graphs as they arrive. In this case it is being fed stock data, and the large graph in the centre shows a snapshot of the current causality graph. In the top right the current most "influential" nodes are shown: due to the simplicity of the input data, the results are not too unexpected — in this case BRK-B (Berkshire Hathaway) is seen as having a lot of effect, which a human could easily inspect and explain and/or discard. For larger scales the extracted correlations are less obvious, but also more difficult to visualise in our initial prototypes.

5.3. Future directions and opportunities

Near-term challenges we would like to address are:

1. Computational acceleration to address the scalability challenge : time-series causal discovery algorithms must scale to datasets with thousands of variables and observations. Computational cost grows rapidly as variable count increases, so optimisations in algorithm design, parallel processing and data reduction become essential. Achieving end-to-end computation for 1,000 variables and 10,000 observations within operational timeframes ensures that causal graphs remain timely and relevant for live risk monitoring.
2. Ground-truth-free evaluation to address the data reliability challenge : testing on real data is essential to assess the discovery power of causal discovery techniques. However, real datasets lack ground-truth causal structures, so direct validation is impossible. Developing evaluation frameworks that do not require ground truth, such as stability metrics over resampled data and causation-based prediction tasks, allows us to assess algorithm performance on actual data. The Data Reliability challenge described in Section 5.1 would also need to be addressed.
3. Real-world adoption : provide facilities to enable real-world users such as regulators to adopt the proposed approach and tools quickly and effectively.

We intend to address the challenges in the near future in the following ways:

1. Scalability : make use of multi-core CPUs, GPUs, and FPGAs to parallelise causal discovery calculations, in order to efficiently meet such requirements by reducing processing times for large variable panels, and enabling frequent graph updates on streaming data.
2. Evaluation : provide evaluation flows to support assessment of proposed casual graph techniques and tools based on: (a) real data from past big and mini crashes; (b) realistic synthetic data produced by machine learning methods such as Generative Adversarial Networks (GANs) or Variational Auto-Encoders (VAEs) to learn the underlying distribution of real data to generate similarly distributed synthetic data.
3. Real-world adoption : build an interactive dashboard that visualises causal graphs, runs scenario exploration and exports causal features to existing stress-test and ML workflows in real time.
4. Supporting a wide range of causal discovery methods : Build a comprehensive suite of causal discovery algorithms for estimating SVARs, each reflecting different assumptions and potential limitations of the data (e.g. acyclic or cyclic causality, latent confounders, high dimensions, sparsity, economic restrictions).

We will identify the high-priority tasks based on discussions with the advisory board, particularly those from the regulatory and financial sectors, and will adapt our work plan, milestones and deliverables where necessary to integrate the most exciting and the most urgent features into the next demonstrator.

6. ETAI : Event-triggered Artificial Intelligence

The goal of ETAI research has so far been to explore possibilities in interpretable and explainable forecasting with Machine Learning (ML) models. This means that the *procedure* by which a forecast is generated becomes explainable to and can be interpreted by a human. So far, this approach to interpretability is based on developing algorithms to address the following two questions:

- **Extracting from the Past:** How to explicitly show the relevant part(s) of a time series signal's *history* that were used to generate the forecast? Are there regions in the history where a high-risk event (e.g. a market crash) occurred that are relevant in the generated forecast?
- **Extracting from the Present:** Which *current trend(s)* in the time series signal impacted the generated forecast decision? How can trends in one time step be abstracted in a way that allows for easier comparison with the trends in previous time steps?

Towards Transparent Forecasting: These questions are answered through two algorithms that are underpinned by their transparency and interpretability. The first is the *Visibility Graph* (VG), allowing the datapoints in a time series to be connected to each other based on whether they are part of a specific trend. The second is the *Tsetlin Machine* (TM), an ML model that learns patterns in data through constructing logic expressions, for example: "TrendX *and* TrendZ *and* TrendQ *contributed to the generated prediction*". These two components form the proposed forecasting pipeline, allowing the above two questions to be answered through the following:

- **Representing the Data and Retrieving from History:** The VG approach to representing series data allows *motifs* to be captured to characterize each time step. Measuring similarity of motifs allows for the current time step (current data point) to be compared against the history of the signal.
- **Making Interpretable Forecasts:** The TM can use this information to build logic expressions that explain which combination of motifs and which data points from the history led to a generated forecast.

This chapter covers a number of topics, so to aid readability each section will have a starting sentence in each key paragraph that summarizes the main idea. Figures are intended to be sufficiently detailed to be read by themselves, but skip to the bottom **Key Takeaway** sentence of each figure's caption to get the gist of the message.

6.1. Motivation: transparency in ML-based forecasting

Primer on the Current State-of-the-Art: Our preliminary work explored how state-of-the-art ML models may be used to better predict sharp rises and falls in time series signals. Traditional statistical models such as ARIMA and GARCH are limited in this regard, as they assume linearity or rely on historical volatility patterns that may not capture the complex, nonlinear dynamics preceding sudden shifts. In contrast, modern ML architectures — such as deep learning and attention-based models — can learn intricate temporal patterns and interactions, enabling them to anticipate abrupt changes more effectively. These ideas are quantitatively summarized through Figure 13.

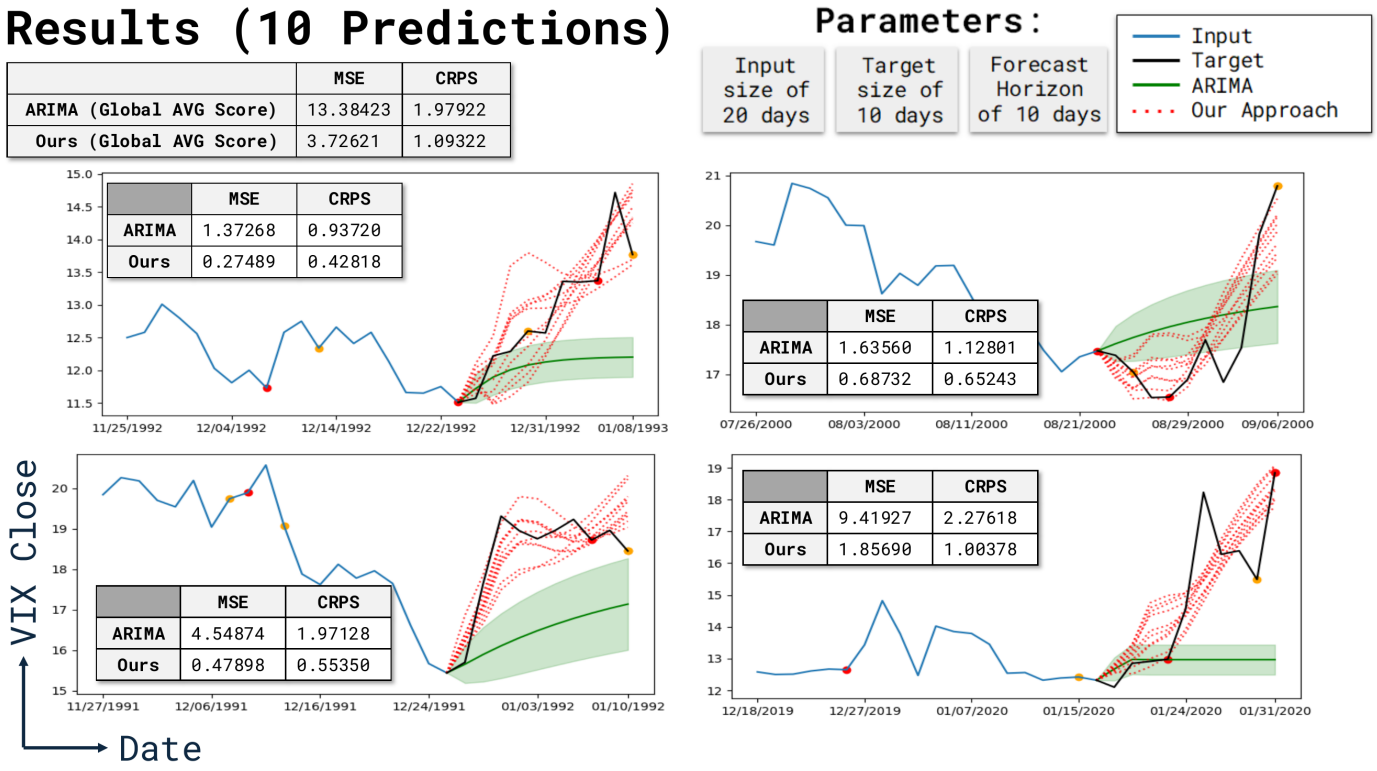


Figure 13. **Initial Study into Predicting Sudden Rises in Time Series Data:** *A rolling window of 20 days was used to predict the next 10 days for the Chicago Board Options Exchange's Volatility Index (VIX). This figure highlights the performance of an ARIMA model (green line for the ARIMA forecast and the green region where the ARIMA prediction error is used to estimate forecast variance) against our ML approach using a Conditional Variational Autoencoder (CVAE) adapted from STRIPE (red lines showing 10 diverse probabilistic forecasts). Notice how the probabilistic forecasts of STRIPE are far closer to the Target when forecasting with a horizon of 10 days.* **Key Takeaway:** *The MSE and CPRS metrics (lower is better) indicates that our ML method in the red is very good at prediction sudden changes. However, the complexity of this model makes it near impossible to distil how the these predictions were made at a human interpretable level.*

Black Box Forecasting — Performance over Interpretability: While these forecasting models demonstrate better performance in these sharp rise/fall events, it is very difficult to understand how the prediction was made. Distilling insights from these models would be useful for exploring underlying behaviours in the signal prior to the sudden change. However, modern ML approaches have, for the most part, abandoned white-box interpretability in favour of extracting as much performance as possible out of the forecasting models.

The next two sections briefly outline the fundamental ideas behind two algorithms that have been incorporated into our initial pipeline for addressing interpretability.

6.2. Visibility Graphs: representing data and relating to history

The focus of this section is to understand how data can be represented in a way that allows the trends and connectivity of the datapoints to be intuitively interpretable and how the current data point can be used to identify a point in history that is most similar.

From Time Series to Graphs: The **Visibility Graph (VG)** provides a way to convert time series data into a graph, where each time step becomes a *node*, and *edges* are formed between nodes that satisfy a visibility condition (described below). This transformation lets us analyze temporal patterns using graph-based methods, such as identifying similar past points in the signal.

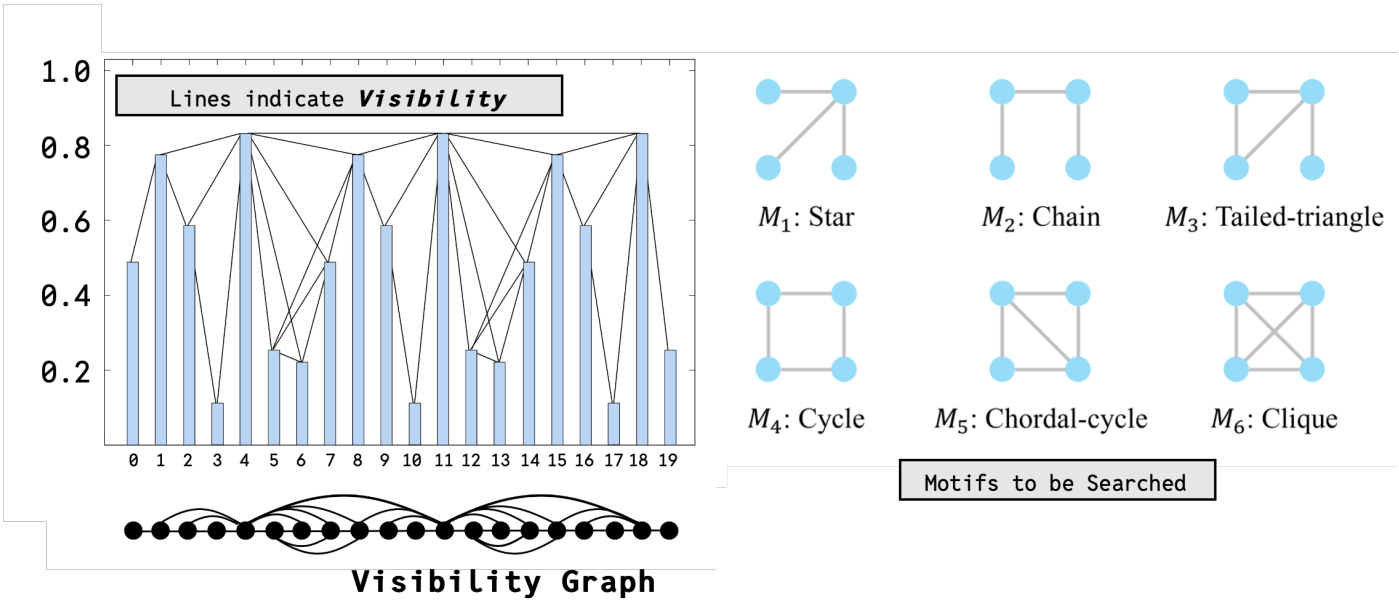


Figure 14. **Key Takeaway:** *VGs convert time series into graphs based on visibility between points (in accordance with Equation (1)). This enables topological analysis of patterns, such as identifying similar regions in history using local connectivity structures called motifs. The trends associated with each time series in the signal is now described through motifs.*

6.2.1. Visibility Graph fundamentals: nodes and motifs

Each data point becomes a *node* in the VG, with an edge drawn to another if no point between them blocks their line of sight. Given a sequence of observations $y_1, y_2, y_3, \ldots$ at times $t_1, t_2, t_3, \ldots$ this can be expressed as:

$$\frac{y_j - y_i}{t_j - t_i} < \frac{y_j - y_k}{t_j - t_k}, \quad \forall\, t_k \in (t_i, t_j) \qquad (1)$$

This method preserves the shape of the original time series, and allows past trends to be related to the current one by comparing node structures. To compare nodes, we analyze their local *motifs* — small subgraphs like stars, chains, and cycles (see the right side of Figure 14) — which capture common connectivity patterns. Each node is described by a probability vector:

$$P_i = \left[p_i^{M_1},\ p_i^{M_2},\ \ldots,\ p_i^{M_{|\mathbb{M}|}} \right] \qquad (2)$$

where $p_i^{M_m}$ is the proportion of times node i in a VG appears in a particular motif M_m (like the ones seen in Figure 14):

$$p_i^{M_m} = \frac{n_i^{M_m}}{\sum_{m=1}^{|\mathbb{M}|} n_i^{M_m}} \qquad (3)$$

Similarity between the current node (current time step) and previous nodes (all time steps in the signal history) is calculated using the Jensen-Shannon divergence metric:

$$\phi_{ij} = JSD(P_i \| P_j) \qquad (4)$$

The divergence is the smallest i where the motifs (or trends) of a particular time step in the signal history are most similar to the current time step. This provides a deterministic way to find the past moment most similar to the present, based on the local structure of the graph.

6.2.2. Planned explorations with Visibility Graphs

In financial data, peaks and valleys often become *high-degree nodes*, making key events easy to detect. Groups of connected nodes (*communities*) reflect consistent behaviours — e.g., trends, reversals, or volatility clusters. By mapping the current point to a similar region in the VG, we can say: "*This forecast aligns with a known pattern (combination of motifs) from [historical period], which had a similar structure.*"

Thoughts about Going Further: We can also annotate VG regions with semantic labels like *"crash onset"* or *"early recovery"*, helping to contextualize current decisions based on past events - returning to Figure 13, this should allow greater explainability to those sharp rises and falls. These annotations can be generated algorithmically or they can be from domain knowledge (e.g., 2008 crash, COVID dip, political events, important X (formerly Twitter) tweets, etc.). Because VGs are deterministic and interpretable, they offer a layer of **accountability**: if a forecast is poor, the graph structure reveals which past patterns it was influenced by, helping to refine or challenge the model.

6.3. Tsetlin Machines: interpretable time series forecasting

The Big Picture — Learn Logic Expressions from Time Series Data: VGs present a useful method of representing data and using graph theory to extract useful features. The challenge of forecasting the next datapoint still remains, which is done through the TM. If the data at a given time step can be expressed as Boolean data (e.g. (Motif is a Triangle and appears more than 20 times), (Motif is a Square and appears 50 times), ...) then the TM can be used to learn logical expressions that describe the data at a given time step as shown below in a rudimentary form:

```
(Trend X) AND (NOT Trend Y) AND (Trend Z) = Signal Decrease
```

6.3.1. Tsetlin Machine fundamentals

An Algorithm with Fully Interpretable Prediction Procedure: The TM is a rule-based ML algorithm that offers a transparent alternative to state-of-the-art Neural Network-based approaches (like the CVAE shown in Figure 13). The core reason for choosing the TM as the main predictor is its full prediction procedure is interpretable. The fundamental components of the algorithm are highlighted in Figure 15.

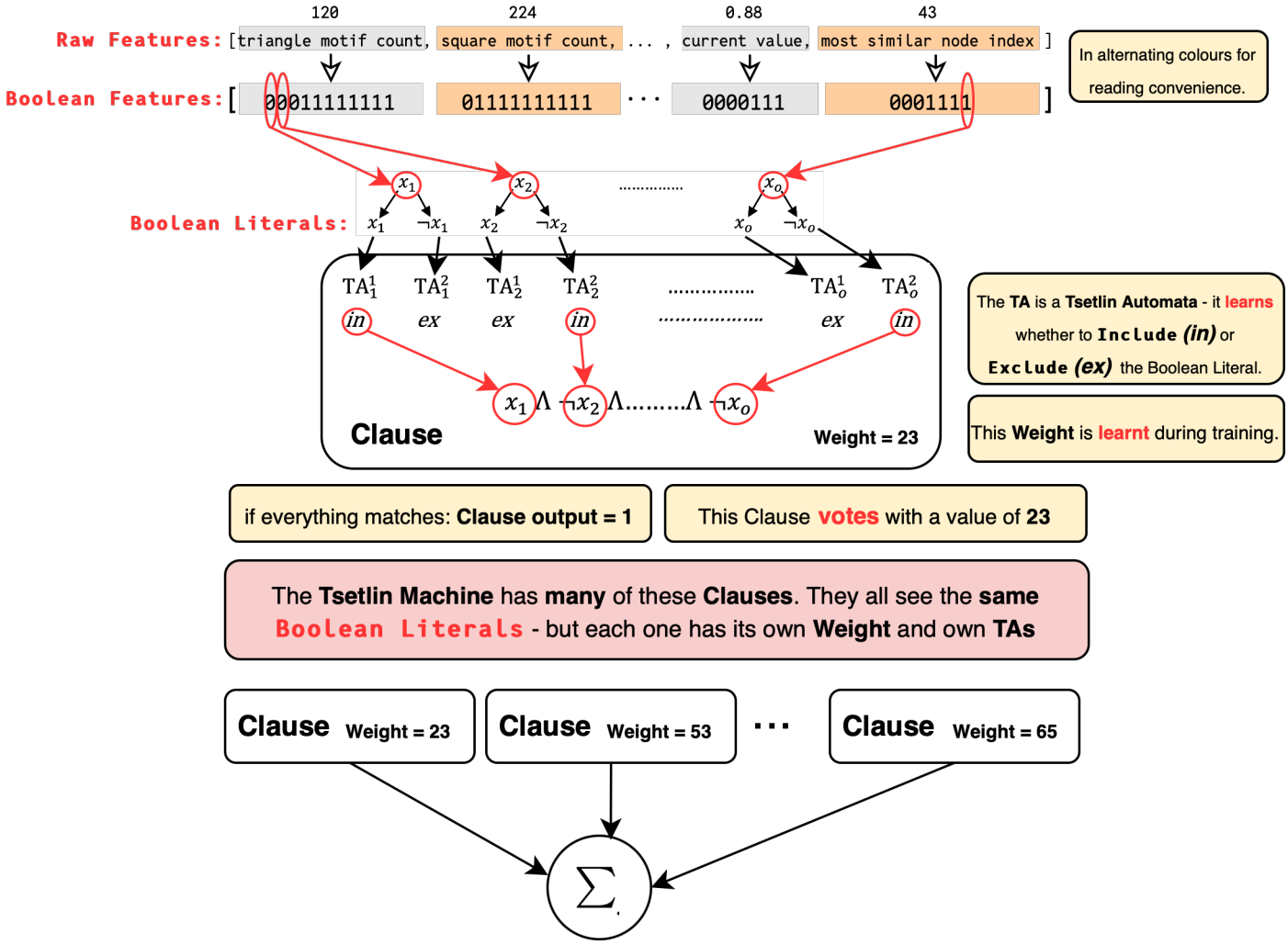


Figure 15. The Tsetlin Machine's main components are highlighted in this figure. Data must be converted to Boolean Literals. In this case the data could be a combination of integer and floating point values — they are converted into Boolean Features (a thermometer encoding is used to represent the data in this case). The Boolean Features and their complements form the Boolean Literals. These literals can be used by the learning elements of the TM called Tsetlin Automata to build logical clauses. **Key Takeaway:** *The TM can learn logical expressions to produce classification and regression outputs. This type of representation allows model forecasts to be interpretable as the clause can be traced back to the original data.*

Automata based Learning to Create Logic Expressions: The TM uses propositional logic expressions to classify and predict patterns in binary or discretized input data, making it well-suited for interpretable time series forecasting. The TM operates through a collection of clauses —- each composed of combinations of input Boolean features and their negations (the set of features and negations are referred to as Boolean literals). Learning happens using Tsetlin Automata (TA) assigned to each Boolean literal; each TA learns whether to *include* or *exclude* a literal to build a clause (see Figure 15).

Algorithm in Infancy: TM-based algorithms are still in their infancy; they have shown very good performance in classification, reaching state-of-the-art in MNIST and similar complexity datasets. Driven by our initial motivation, we would particularly like to explore the *regression* capabilities of the TM. In this setting, where the TM is called the "Regression TM" (RTM), its operations are particularly applicable to time series forecasting, where each clause contributes a weighted vote towards a real-valued prediction, as shown in Figure 15. These weights are learned iteratively, rewarding clauses that improve accuracy and penalizing those that introduce noise. This learning mechanism is inherently interpretable, as each weight update is explainable due to the discrete decision about pattern inclusion or exclusion.

6.3.2. Regression Tsetlin Machines: performance and challenges

Table 1 reports initial investigations into RTM performance. The mean absolute error (MAE) with 95% confidence intervals is used to evaluate forecasting models across five diverse datasets : dengue incidences (disease occurrence), energy performance, stock selection (split into annual return and real win rate), real estate valuation, and aerofoil noise. These datasets span epidemiological forecasting, energy modelling, financial prediction, property valuation, and acoustic modelling, respectively. The Regression Tsetlin Machine (RTM) consistently outperforms baseline models -— including Regression Trees (RT), Random Forests (RF), and Support Vector Regression (SVR) -— on most tasks. In particular, RTM achieves the lowest MAE on the dengue and stock selection datasets and performs competitively on real estate and aerofoil benchmarks. Notably, while SVR slightly outperforms RTM on stock annual return, RTM remains the most consistently accurate and interpretable model overall, especially in structured or pattern-rich datasets such as dengue incidence and energy consumption. These chosen algorithms also offer a level of interpretability — ongoing research is focused on comparing the ability to visualise the model's prediction procedures.

Table 1. Mean MAE with 95% confidence intervals for selected models on five datasets.

Model	Dengue incidences	Energy performance	Stocks: Return	Stocks: Win rate	Real estate	Aerofoil
RTM	5.184	0.503 ± 0.00	0.077 ± 0.00	0.089 ± 0.00	5.218 ± 0.00	2.244 ± 0.00
RT	7.769	0.598 ± 0.00	0.091 ± 0.00	0.096 ± 0.00	6.033 ± 0.11	2.280 ± 0.07
RF	9.122	0.563 ± 0.00	0.088 ± 0.00	0.097 ± 0.00	5.360 ± 0.00	1.921 ± 0.00
SVR	5.305	1.155 ± 0.00	0.073 ± 0.00	0.092 ± 0.00	5.697 ± 0.00	2.156 ± 0.00

Challenges: One of the core challenges of all TM-based implementations is the impact of Booleanization. Some of the effective approaches seen so far involve thresholding data into pre-defined ranges, but this is still ongoing research. Booleanization must allow clauses to remain interpretable. Another exploration is how the hyperparameters of the RTM affect the learning performance; as yet, this is unexplored in the TM community.

6.4. Proposed pipeline

VG + TM: By merging the capabilities of both the VG and TM, we can achieve a promising strategy that applies the TM's learnable rule-based, yet interpretable, regression in combination with the rich, network-derived features from the VG and Motif analysis. A run-time view of our prototype User Interface is shown in Figure 16.

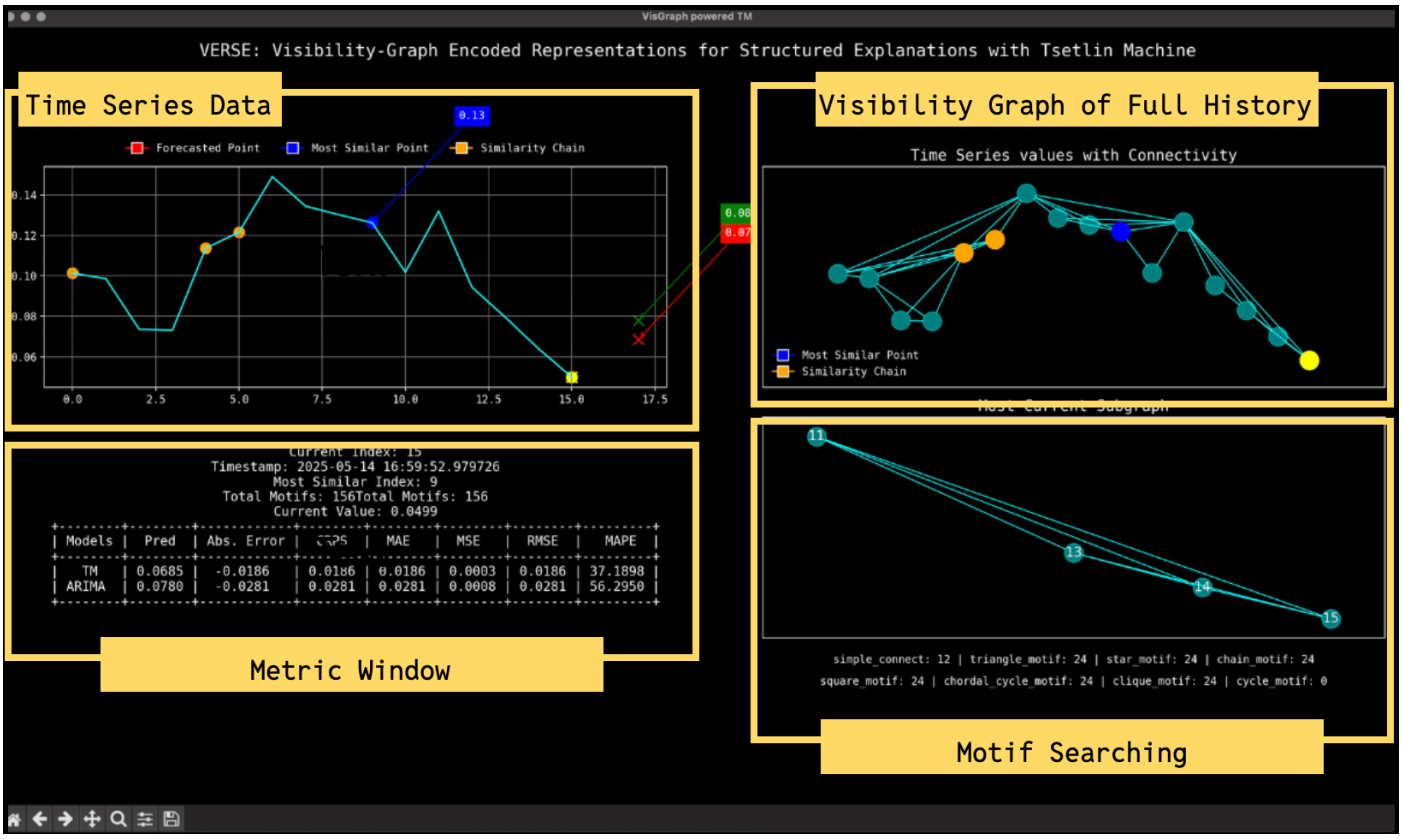


Figure 16. **Prototype User Interface:** *This figure show a runtime view of the proposed approach of interpretable forecasting using VGs and TMs. The time series data is represented in its standard from through the Time Series Data window where the forecasts will be made (our RTM forecasted point is shown with a red cross, ARIMA is shown with a green cross). Notice there are different coloured points. The yellow point refers to the current time step. The blue point represents the most similar point in the history and the orange points show the recursively most similar points using the blue most similar point as a starting point. The bottom right window shows the connections of the current yellow point (the current subgraph). This graph is searched to get all the motif counts which are reported at the bottom. The full VG of this time series is seen in the top right window.* **Keys Takeaway:** *VG + TM based prediction and analysis can be applied to real time data streams.*

This hybrid approach can be envisioned as shown in Figure 17 with the following steps:

1. When a raw time series is given, we map out the time series to the VG, where Motif analysis is used to extract local structural information, before computing a similarity (or trend) measure from the historical nodes. This data is then prepared for the TM to learn the patterns found from identifying the most similar nodes with respect to the current node's Motifs.
2. Next the set of data (except the values tied to the current node's value, barring the Motifs) is Booleanized to produce a binary feature representation of the VG preprocessed time series data.
3. The Booleanized representation of the data is then passed into the Regression TM pipeline, which consists of clause formation, aggregation and continuous prediction.
4. The outputs of both the TM and the VG are then combined to enable interpretable outputs/decisions made by the TM with respect to the VG data. While creating this outputs we are able to traverse back through multiple nodes in the history to identify similarity of changes that happened in the past so that we can identify potential significant events in the upcoming present/future. As such, both the final forecast will provide the benefits from the Regression TM's interpretable rule-based expressions and the direct similarity and trend that is inferred from VG analysis.

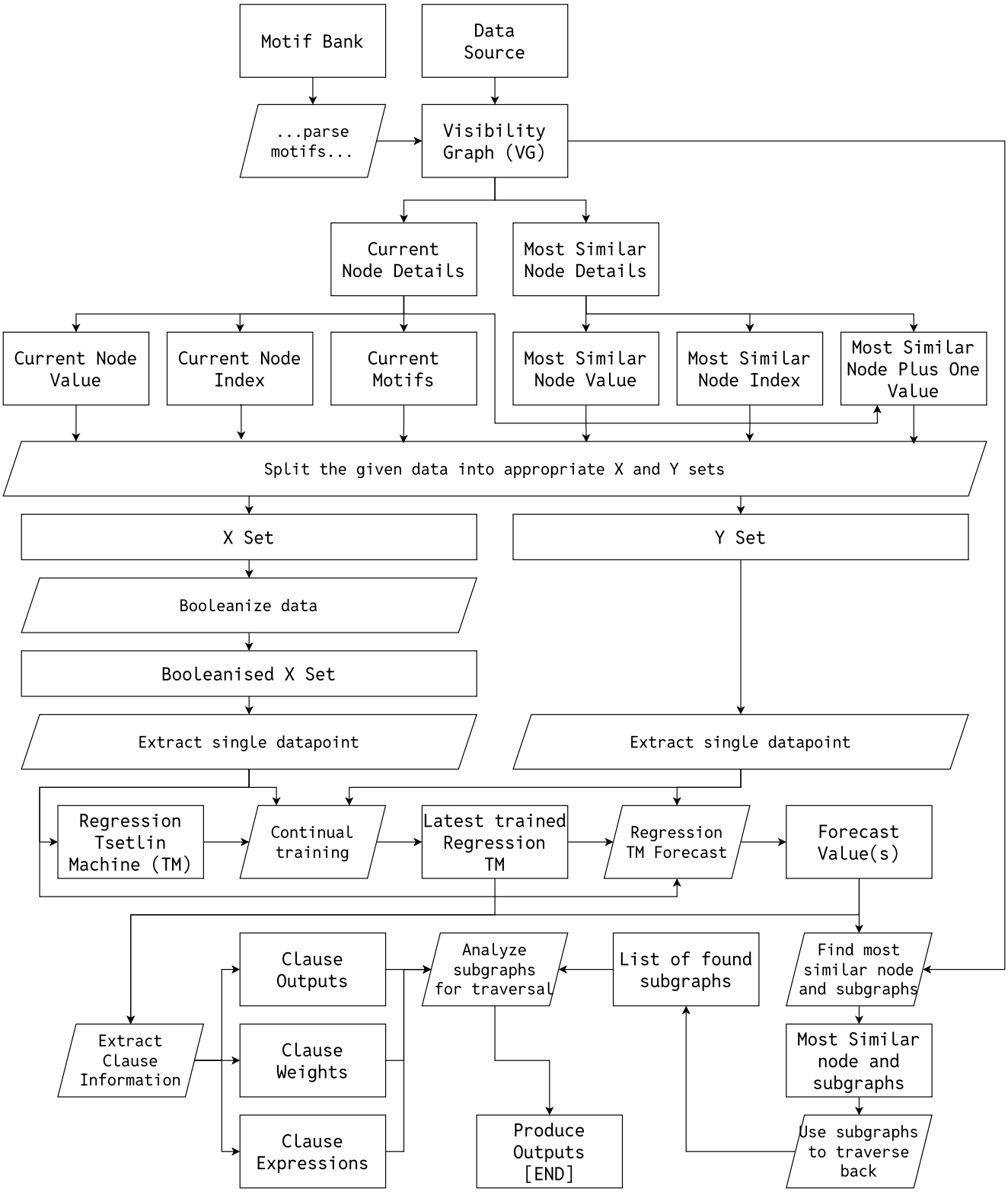


Figure 17. Combined Visibility Graph and Tsetlin Machine Workflow.

Thoughts on Risk: The VG provides a capability for identifying similar nodes in the past, while the TM's interpretable clause expressions can be annotated with structural sentences related to how the prediction is constructed. From the logical expressions that the TM has learned from the VG data, we are able to provide convenient means for traversing back in history, where we can pinpoint patterns of near-term/exact historical events that are inherently risky, e.g. the 2008 global market crash and the Covid-19 crisis.

7. Management

7.1. Working together

Frequent communications have been built into the project from the outset, with weekly all-hands Teams meetings from the start of the project. We have stuck to a consistent weekly schedule of Tuesday 16:00-17:00 throughout, with some gaps around the holidays. These meetings are attended by all SONNETS project members, with occasional per-person absences due to teaching clashes, conferences, holidays, etc. Meetings consist of a 10-20 minute update session where all members update on progress, then the remainder is taken up with: discussion of cross-cutting concerns around research and organisation; demonstrations and presentations from individual team members; or we just finish early and get some time back.

Alongside regular Teams meetings there have also been more focussed Teams meetings at various points, including:

- Training sessions : for example training on how to use the ETC layer
- Presentation rehearsal : in particular for developing the stakeholder presentation (see Section 8.1)
- Technical/coding sessions : dedicated to looking at code and solving integration problems

Collaboration is supported through two main tools:

- A shared Github organisation for code repositories
- A shared SharePoint site for documents and data

Both the Github and SharePoint site are access controlled, and restricted to the project team members, except for any GitHub repositories which have been deliberately made public. A complete data management policy was created at the start of the project, and covers how we will manage proprietary or confidential data as and when it arises.

Alongside virtual meetings we have used a number of in-person meetings to support the work. Given the long distances involved we have considered both demands on personal time and sustainability when organising meetings, and prefer fewer but longer in-person meetings. Particular meetings we have had include:

- Kick-off meeting (Southampton) : 1-day all-hands meeting to establish project direction and introduce everyone to each other.
- Mid-year all-hands Meeting (Imperial) : 1-day all-hands meeting to consider current project directions and brain-storm potential demonstrators.
- Hack-athons and research discussions (Newcastle) : subset of investigators have gather together on three separate occasions to have detailed in-person discussions on technical aspects and demonstrator planning.

7.2. Staffing

7.2.1. University of Southampton (UoS)

- *Principle Investigator*: Prof. David Thomas (DT)
- *Lead Investigator for UoS*: Prof. Andrew Brown (AB)
- *Investigator*: Dr. Mark Vousden (MV)
- *Investigator*: Dr. Graeme Bragg (GB)
- *Investigator*: Dr. Dana Dghaym (DD) : (currently on long-term leave)
- *Research Fellow*: Dr. Shidur Rahman (RS) : Aug 2024 - now

Dana Dghaym went on long-term leave for personal reasons at the start of the project, though she is still keen to take part in the project. Over summer 2025 we will discuss this further, and see if this remains tenable - hopefully this is achievable. If not, we will look to bring other early-career academics with similar skillsets into the project.

At the start of the project the investigators at Southampton were either new to the university (DT), new in post (MV, GB, DD), or have been reducing PhD intake (AB). As a result there are few known internal applicants for research fellows, so we rely on external recruitment and word of mouth through internal and external networks. Shidur Rahman was recruited through external advertisements. There is budget for a second post-doctoral research fellow position at Southampton, which has yet to be filled across three advertised rounds. We have now changed the adverts (with feedback from other SONNETS partners), and are advertising again. If still not successful we will look at increasing the advertised job level in order to compete more effectively with industry, as we are trying to recruit researchers with sought-after enterprise skills.

Fortunately the project has multiple academic investigators, all of whom have contributed a lot of hands-on skills and research knowledge to the project, particularly around the ETC-tier. This has been very useful in the initial API exploration and development phases, where the intent was always to have a lot of cross-cutting high-level discussion and protoyping. As a consequence the ETC-tier development has not been held back but this, but going forwards both the amount of engineering and theoretical development needed goes up, so recruiting another research fellow is a priority.

The project is also allocated 0.5 FTE for an administrator, and 0.5 FTE for an EDI+outreach support role, originally conceived as two distinct part-time roles. At the outset discussions were ongoing with a known internal applicant for the admin role with knowledge of this type of grant, but it eventually became clear this would not work. We had also thought to buy out time from existing in-school EDI and outreach expertise, but this was not feasible due to demands from other school activities. Opportunities for combined administration roles with other large grants starting in the faculty were then considered, but ultimately this ended up just wasting more time. The current approach now is to recruit a single joint role that covers both administration and EDI/outreach, though this has presented some difficulty due to the need to recruit non-academic staff and define appropriate job roles.

7.2.2. Newcastle University (NU)

- *Lead Investigator for NU*: Prof. Rishad Shafik (RS)
- *Investigator*: Prof. Alex Yakovlev (AY)
- *Investigator*: Dr. Kabita Adhikari (KA)
- *Research Fellow*: Dr. Alex Chan (AC) : Jan 2024 - now
- *Research Fellow*: Tousif Rahman (TR) : Jan 2024 - now
- *PhD Student*: Bob Pattison (BP)

Alex Chan has been with the project as a post-doctoral researcher from the start.

Tousif Rahman joined the project as a pre-doctoral researcher, and has recently been appointed a lecturer at Newcastle University. They are expected to remain a contributor to the project in their new role, though no longer as a full time research fellow.

7.2.3. Imperial College London (ICL)

- *Lead Investigator for ICL*: Prof. Wayne Luk (WL)
- *Academic Contributor*: Prof. Walter Distaso (WD)
- *Research Fellow*: Dr. Ce Geo (CG) : Jan 2024 - now
- *Research Fellow*: Dr. Filippo Pellegrino (FP) : Jan - Mar 2024
- *Research Fellow*: Dr. Paul Labonne (PL) : Apr 2025 - now

Filippo Pellegrino joined the project at the start, but left to take up another post doctoral position a few months into the project.

Walter Distaso was intended to be an Investigator at the project planning stage, but this has not been the case in practice due to a change of circumstances. Alongside FP leaving, this caused some interruption to the project, despite Walter remaining fully engaged in meetings and the overall research. Particular problems involved use of and transfers of research funds between departments within Imperial, and ability to advertise and recruit research fellows.

This has recently been largely managed on the funding side of things, resulting in the recent introduction of Paul Labonne to the project, who brings back in a large amount of finance domain-specific knowledge.

7.3. Responsible Research and Innovation

Our approach to responsible research and innovation is to regularly engage in an AREA (Anticipate, Reflect, Engage, React) style review of the project and the potential output. At the 6 month in-person meeting we brain-stormed in this area, particularly on the "Anticipate" side of things.

Some arising matters that we identified are:

- Graeme Bragg: *Could SONNETS create unexpected feedback loops between policy makers and the economy? Even if SONNETS is purely providing information to policy makers, it could still effectively create a closed loop.*
- Kabita Adhikari : *If we release everything as open-source so that the methods and model-creation algorithms are known, could financial institutions game the system to make themselves appear less risky according to the models, essentially hiding risk?*
- Mark Vousden: *Rather than institutions hiding their own risk exposure, could they use the open aspect of models for competitive advantage and profit-taking by betting against the risk models, essentially degrading the accuracy of the entire national risk model?*

Reflecting on these, they are real concerns, but for the moment not directly applicable to the research given the capabilities of the prototypes and algorithms. So we have logged them to re-consider as we develop new capabilities, as they will become important as capabilities improve.

8. Engagement and dissemination

8.1. Stakeholder engagement

A big part of the ambitious vision of SONNETS is trying to bridge the gap between engineering and AI researchers with new capabilities on one side, and potential end-users who could make use of those capabilities. As part of the project management plan we wanted to actively bridge that gap — partially through participation on the advisory board and collaboration, but also through engaging with current and potential stakeholders. In the context of this project's goals we have a particular focus on engaging with potential end-users from the regulatory and national risk management side, as this is needed to close the loop between technology capability and exploitation.

At the six month all-hands meeting the SONNETS team worked together to come up with three case study demonstrators, which could be used both to guide the integrated research and to act as discussion points with stakeholders. Over the next few months these three demonstrators were then refined down to two broad areas for presentation, discussion and engagement with potential end-users:

- Novel methods for probabilistic forecasting using new types of AI
- Causality extraction from time-series to identify which underlying time series are "driving" the overall market.

This was then used to guide underlying research, and also to start preparing a "stakeholder presentation", which is a 45 minute pitch intended to show-case our research ideas particularly to potential financial end-users. This pitch was developed over a number of months as the research evolved, first using Walter Distaso as an internal sounding board, then following on with two "proper" pitches to Prof. David Miles, who is head of economic analysis at the Office for Budget Responsibility, and previously a member of the Monetary Policy Committee.

Even with multiple rounds of refinement and re-pitching it was still difficult to cross the language barrier — both in terms of explaining what the techniques SONNETS is developed are, and in understanding some of the questions coming back from stakeholders.

Some lessons we have learnt in this initial stakeholder engagement are:

- Trust and explainability of **any** ML method is key for user acceptance. Things that we might casually accept as "modern ML practises" might be acceptable in commercial finance, but come across as opaque and arbitrary from the regulatory / macro-economic perspective.
- Cutting edge AI must be complemented by traditional methods. We are not seeking to replace existing analyses, and new methods should be integrated alongside a spectrum of existing methods, from classical statistical analysis through to human-designed heuristic indices.
- Going forwards we need to focus on individual outreach to multiple people, rather than trying to schedule "group" stakeholder meetings.

Acknowledgement: The SONNETS team would particularly like to thank Prof. David Miles for the time taken to provide multiple rounds of feedback during this process.

8.2. Current outreach activities

Our outreach plans consists of two main strands:

- Informing school leavers about further engineering study though talks and hands-on activities.
- Promoting engineering careers and PhD study both as successful, lucrative, and rewarding.

As well as generically advocating for these paths, we are particularly interested in supporting and encouraging traditionally under-represented groups in undergraduate and post-graduate engineering education. A known problem, of which multiple investigators on the project are aware, is the "pipeline" problem, where the set of potential recruits at the PhD or academic level is restricted to the set of people entering under-graduate and PhD programmes. In particular, we want to support underrepresented groups entering these pipelines, notably to address gender imbalances at the academic level.

Outreach activities so far include:

- Undergraduate Teaching : On the ELEC3219 Advanced Computer Architecture module at Southampton, both SONNETS and the previous POETS grant were used as an example of funded UK university research. This included discussion of funded research in the first lecture, and a three lecture case study at the end focusing on the outputs of funded research.
- Schools taster course : ECS Taster Course: Each year ECS at Southampton runs a week-long residential for Year12 students designed to give students a taste of what studying engineering and computer science is like, and an idea of the career options open to them. During the summer 2024 iteration DT gave a 30 minute talk covering aspects of computer engineering, but also explaining a bit about how research happens, using the SONNETS grant as an example. During questions afterwards a number of students said that they weren't aware of this type of research pathway, and expressed interest in doing a PhD after their degree.
- Undergraduate guest lecture : Invited Lecture by RS at Southampton University to MEng Year 4 Electronic and Computer Engineering students in 2024 called "Introducing Tsetlin Machine".
- Undergraduate recruitment : a number of Southampton academics have been involved in the design and delivery of two new programmes, one in Computer Engineering and another in Artificial intelligence. DT is an admissions tutor for both programmes, and during multiple in-person recruitment activities over the summer of 2024 SONNETS was used as a method of explaining impact and career trajectories in computer science in general, and artificial intelligence in particular. Using SONNETS as an example worked particularly well (in DT's subjective opinion) when engaging with both parents and prospective students at the same time, as many of them want to know what types of positive larger scale impact they could have if they pursue engineering.
- Schools outreach : Two outreach talks by RS in 2025 at Westerhope Primary School, Newcastle 2025: "STEM and Machine Learning".
- Schools outreach : as an initial external outreach activity we have been setting up a SONNETS-specific research and careers talk to 6th form students at Watford Grammar School for Boys. This is intended to be a pilot for a repeatable outreach engagement activity. This talk was initially scheduled for Apr 2025 to be delivered by DT, but has now been re-scheduled to June.
- Postgraduate guest lecture : Invited Lecture by RS at Newcastle University to MSc Embedded Systems and IoT students in 2025 called "Thinking Differently for Machine Learning: Introducing Tsetlin Machine".

8.3. External talks

- Workshop Talk: "Accelerating graph algorithms (and more) using 50,000 threads across 50 FPGAs", Delivery: D. Thomas, Jan 2024 : Invited talk at HiPEAC, which covered aspects of the previous POETS grant, and the change in goals and direction for SONNETS.
- Academic Talk: "SONNETS: National financial risk modelling for the UK", Delivery: D. Thomas, Feb 2024 : Selected talk at NVidia event in Southampton to around 200 academic GPU users, talking about the overall goals of SONNETS.
- Industrial Talk: "Using Event-Triggered Computing to Program the Heterogeneous Cloud", Delivery: D. Thomas, 2024 : Invited talk given at ARM campus in Cambridge, mainly targetting about 40 computer architecture practitioners. The talk covered hardware design from POETS, and then laid out the goals and motivation for SONNETS.
- Forum Talk: "From POETS to SONNETS", Delivery: G. Bragg, Mar 2024 : Talk given to UK Design Forum, which is a national-level forum bringing together industrial and academic attendees from the general area of digital and hardware design in the UK. This talk was promoting the new SONNETS grant, and establishing areas for future potential collaboration.
- Academic Talk: "Optimising Diverse Accelerator Designs", Delivery: W. Luk, 2025 : Invited presentation at WRC 2025 : Covered a set of existing work on optimising compute accelerators, with a call to arms for future deployment and optimisation using SONNETS.
- Workshop Talk: "Multi-FPGA multi-core overlay architectures for executing graph-based applications", Delivery: D. Thomas, Jan 2025 : Invited presentation at Workshop on Reconfigurable Computing as part of HiPEAC. This covered work from the POETS grant, and then the ongoing work in the SONNETS grant.
- Academic Talk: "Tsetlin Machines: stepping towards energy-efficient, explainable and dependable AI", Delivery: A. Yakovlev, Feb 2025 : Talk given at Durham University. Covered Tsetlin Machines as an example of advanced machine learning, and how they fit within the SONNETS project.
- Invited Presentation: “Energy-efficient and Explainable Machine Learning using Tsetlin Machine”, Delivery: R. Shafik, 2025 : Talk given at Durham University, 2024.
- Keynote Talk: “Empowering Green Machine Learning: One Logic Conjunction at a time”, Delivery: R. Shafik, 2024, Keynote at NILES Conferences in Egypt 2024,
- Invited Presentation: “Energy-efficient and High Throughput Machine Learning using Tsetlin Machine”, Delivery: R. Shafik, 2025, Invited presentation at EDGE AI Symposum in Austin Texas.

8.4. SONNETS publications

- A. Chan, A. Wheeldon, R. Shafik and A. Yakovlev, "Design of Event-Driven Tsetlin Machines Using Safe Petri Nets."," in "International Conference on Applications and Theory of Petri Nets and Concurrency", pp. 357-378, 2024.
- Z. Que, M. Zhang, H. Fan, H. Li, C. Guo and W. Luk, "Low Latency Variational Autoencoder on FPGAs," in IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 14, no. 2, pp. 323-333, 2024.
- S. Abbas, R. Shafik, N. Soomro, R. Heer and K. Adhikari, "AI predicting recurrence in non-muscle-invasive bladder cancer: systematic review with study strengths and weaknesses", Frontiers in Oncology, Vol 14, 2025.
- Q. Wang, Z. Que and W. Luk, "Trustworthy Codesign by Verifiable Transformations," 2024 IEEE International Test Conference in Asia (ITC-Asia), pp. 1-6, 2024.
- C. Guo, H. Wu, and W. Luk, "Resource-Constraint Bayesian Optimization for Soft Processors on FPGAs", Proceedings of the 14th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies, pp. 27-36, 2024.
- Z. Zhang, H. Fan,H. Chen, L. Dudziak,and W. Luk, "Hardware-Aware Neural Dropout Search for Reliable Uncertainty Prediction on FPGA", Proceedings of the 61st ACM/IEEE Design Automation Conference, article no. 301, 2024.
- J. Vandebon, J. G. Coutinho and W. Luk, "Auto-Generating Diverse Heterogeneous Designs," IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 116-123, 2024.
- S. Duan, R. Shafik and A. Yakovlev, "ETHEREAL: Energy-efficient and High-throughput Inference using Compressed Tsetlin Machine.", arXiv preprint arXiv:2502.05640, 2025.
- G. Mao, T. Rahman, S. Maheshwari, B. Pattison, Z. Shao, and R. Shafik, "Dynamic Tsetlin Machine Accelerators for On-Chip Training Using FPGAs", IEEE Transactions on Circuits and Systems, pp 1-14, 2025.