Essays on modeling and analysis of dynamic sociotechnical systems

A Dissertation Presented

by

David Rushing Dewhurst

to

The Faculty of the Graduate College

of

The University of Vermont

In Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
Specializing in Complex Systems and Data Science

May, 2020

Defense Date: February 14th, 2020
Dissertation Examination Committee:

Peter Sheridan Dodds, Ph.D., Advisor
Christopher Danforth, Ph.D.
Safwan Wshah, Ph.D.
Nicholas Allgaier, Ph.D., Chairperson
Cynthia J. Forehand, Ph.D., Dean of Graduate College

Abstract

A sociotechnical system is a collection of humans and algorithms that interact under the partial supervision of a decentralized controller. These systems often display intricate dynamics and can be characterized by their unique emergent behavior. In this work, we describe, analyze, and model aspects of three distinct classes of sociotechnical systems: financial markets, social media platforms, and elections. Though our work is diverse in subject matter content, it is unified though the study of evolution- and adaptation-driven change in social systems and the development of methods used to infer this change.

We first analyze evolutionary financial market microstructure dynamics in the context of an agent-based model (ABM). The ABM's matching engine implements a frequent batch auction, a recently-developed type of price-discovery mechanism. We subject simple agents to evolutionary pressure using a variety of selection mechanisms, demonstrating that quantile-based selection mechanisms are associated with lower market-wide volatility. We then evolve deep neural networks in the ABM and demonstrate that elite individuals are profitable in backtesting on real foreign exchange data, even though their fitness had never been evaluated on any real financial data during evolution.

We then turn to the extraction of multi-timescale functional signals from large panels of timeseries generated by sociotechnical systems. We introduce the discrete shocklet transform (DST) and associated similarity search algorithm, the shocklet transform and ranking (STAR) algorithm, to accomplish this task. We empirically demonstrate the STAR algorithm's invariance to quantitative functional parameterization and provide use case examples. The STAR algorithm compares favorably with Twitter's anomaly detection algorithm on a feature extraction task. We close by using STAR to automatically construct a narrative timeline of societally-significant events using a panel of Twitter word usage timeseries.

Finally, we model strategic interactions between the foreign intelligence service (Red team) of a country that is attempting to interfere with an election occurring in another country, and the domestic intelligence service of the country in which the election is taking place (Blue team). We derive subgame-perfect Nash equilibrium strategies for both Red and Blue and demonstrate the emergence of arms race interference dynamics when either player has "all-or-nothing" attitudes about the result of the interference episode. We then confront our model with data from the 2016 U.S. presidential election contest, in which Russian military intelligence interfered. We demonstrate that our model captures the qualitative dynamics of this interference for most of the time under study.

For N.

...the sunsets and the mountains never changed, the stars were the same stars.
-Roberto Bolaño

Acknowledgements

There are many people to thank in a short length of paper. My dissertation committee, of course: my good friends, mentors, and colleagues Peter Dodds and Chris Danforth; Safwan Wshah, who very kindly agreed to be on my dissertation committee at the last minute; and Nick Allgaier, with whom I have had many pleasant conversations and who kindly accepted my request to serve as chair of my committee.

Other members of the academy—Ken Golden, Richard Foote, Bill Gibson, Marc Law, and Peter Von Doepp— have influenced the way in which I approach scientific inquiry and my beliefs more generally. I hope that they will recognize their work in mine for many years hence. My colleagues at MassMutual—Alex Bogdan, Yi Li, and Jasmine Geng—have provided me with both excellent research suggestions and good company in the relatively short time we have been acquainted.

I have drawn support from many incredible administrative folks over the years. Two in particular, Sean Milnamow in the financial services office and our office manager Melissa Rubinchuk, have essentially made it possible for me to function as a human while pursuing my doctorate.

My former students probably taught me as much as I taught them, though the topics of instruction were (hopefully) largely orthogonal. I hold particularly fond memories of Kyle Cyr, Britton Blanchard, Sierra Natzke, Richard Lowrey, Max Seckler, Jeremy Nadler, Chloe Moulin, Julia Williams, and Amra Jusupovic, and hope they are doing well, wherever they are.

Colleagues from my time in our nation's capital—Alex Bales (who was more convinced

than I that I could earn a doctorate), Nour Hayek, Gwan Sung Yi, Tomas Seo, Alexa Weaver, Bolei Liu, Ross Dietrich, James Griffin, Andrew Lundeen, Gavin Ekins, and Mark Calabria—will know how much their support meant to me.

I owe special thanks to Jeff Yass and Wade Shen for providing me with fleeting glimpses of worlds on which I have become singularly focused.

Other people I will thank and not give reasons. I guess they will know well enough why. My (under-)graduate colleagues and office-mates: John Ring, Colin Van Oort, Josh Minot, Michael Arnold, Thayer Alshaabi, Kelly Gothard, Anne Marie Stupinski, Vanessa Myhaver, Melissa Seib, Sophie Hodson, and Max Green. Tyler Gray in particular — the feeling is mutual, my friend.

Friends from near and far days: Alex, Amber, Dylan, Guiseppe, Bri, Kayli, Lindsay. Many others from a long time ago.

My parents, Stephen Dewhurst and Sarah Hewins. My blood family: Lena, Sonia, Max, Alison, Hilary, Raúl, David, and Cheryl. My sister- and brother-in-law, Nicole and Dylan Comeau. My wife, Casey Dewhurst (née Comeau). And finally, I thank my (at time of writing) as-yet-unborn son Jacob, of whom I have been thinking fondly for some time.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Sociotechnical systems

This dissertation is about the modeling and analysis of sociotechnical systems. We define a "sociotechnical system" to be a system that is composed of humans and algorithms that interact, both competitively and cooperatively, under the partial supervision of a (usually decentralized) controller. We will now clarify various parts of this definition and explain why we believe that each of these parts is necessary.

- *What is an algorithm?* — Any particular sequence of steps that is taken to achieve a goal is an algorithm by definition. What we mean is a formalized algorithm, designed with some measure of optimality in mind, that is written for execution by, and usually executed on, a computer. In this sense, algorithms attempt to solve well-defined optimization problems, meaning that the objective function to be optimized can be concisely described and evaluated relatively easily. In general, algorithms attempt to find the solution to only one optimization problem at a time and do not consider the consequences to other problem spaces of the actions that they take in solving the problem for which they were designed. Algorithms are usually designed by humans. This is not always the case; for example, the field of genetic programming applies evolutionary pressure to populations of computer programs, while gradient-based optimization methods apply automatic differentiation to the parameters of algorithms that are represented as directed, acylic graphs.

- *Why differentiate between humans and algorithms?* — In general, humans do not solve problems in the same ways in which they design algorithms to solve problems [7, 8, 9, 10]. That is to say, humans do not usually specify a formal

objective function and then carefully construct a method by which the function can be optimized.

Though this dissertation is not about satisficing or heuristic decision-making strategies, we raise this point because we hypothesize that this divergence between humans' and algorithms' decision-making procedures when applied in the same problem space is the partial generator of many interesting phenomena. We display examples of these phenomena in each chapter. In Ch. 2 we demonstrate, in the context of an agent-based model (ABM), that interactions between heuristic trading strategies and trading strategies that are gradually optimized through evolutionary pressure generate multiple observed statistical features of real-world asset markets. Using the discrete shocklet transform, a qualitative, shape-based similarity search method developed in Ch. 3, we show that socially-significant events—events that are meaningful to humans for some cultural reason—generate word usage time series that are statistically different from time series corresponding to function words or words that are used seasonally. And in Ch. 4, we show that a model of rational interference in a two-candidate first-past-the-post election exhibits drastically different behavior depending on the motives of the interfering agent and the agents that seeks to mitigate such interference. Each of these results provides both direct (data-driven) and indirect (model-driven) evidence to suggest that our hypothesis is not without merit.

- *Why both cooperative and competitive?* — Cooperation and competition are not mutually-exclusive, though cooperation can be analyzed entirely within the framework of non-cooperative game theory without loss of generality [11].

Within the same universe, coalitions of agents can agree to cooperate for a certain length of time and then compete or *vice versa*. Cooperation can also emerge from competitive behavior. This fact has explicit ramifications for observed phenomena generated by a collection of humans and algorithms. For example, cooperative behavior between trading agents emerges in the context of the ABM introduced in Ch. 2 even as these agents are subject to strong evolutionary pressure and hence are competing with one another.

- *What is a controller, and what is it controlling?* — Human society generally operates under laws or *de facto* rules of play promulgated by states or state-like entities [12]. Agent-to-agent interactions are also governed by private rules and regulations created, and actions (potential or actual) taken, by the creators of the marketplace in which the interactions take place. We term a "controller" the rules, regulations, and potential or actual actions taken by the designers of the framework in which the humans and algorithms are interacting. Examples of these controllers, both public and private, are: the terms of service of Twitter and Facebook coupled with their power to remove (or not) automated accounts; limit-up / limit-down bands set by stock exchanges' matching engines; and supervision of electoral processes, counting of votes, and restrictions on campaign finance. It is clear to us that well-designed rules and regulatory actions can play an integral role in ensuring the stability of collections of interacting humans and algorithms, just as poorly-designed or -implemented ones can generate deleterious outcomes for members of such a collection. Thus, even when not explicitly modeled or accounted for in data analysis, we believe that it is vital to acknowledge the existence of these controllers and consider the ways

in which their existence and operation affects the activity, both observable, and latent, of the collection of interacting agents.

It is obvious that such a controller need not be centralized. As an example, the current U.S. National Market System (NMS), the network of exchanges that trade public common stocks, consists of no fewer than 13 active stock exchanges and an uncertain number of less-regulated alternative trading systems (ATS). Each of these trading venues has its own rules of operation: in what manner agents may submit orders to the venue, what types of orders are allowed in the first place, how these orders can be modified or canceled, how much the price of the stock is allowed to move in a fixed amount of time, and so on. But trading agents are perfectly capable of interacting in more than one stock exchange or ATS, and frequently do so [13]. Hence, agents are subject to control from more than one source; the control is decentralized. An even simpler example is the voting system for federal nation-wide elections (i.e., presidential elections) in the United States; each state has its own voter registration laws and regulations.

So this dissertation is about sociotechnical systems—and we now have a concrete outline of what it means for a system to be sociotechnical. But this dissertation is also particularly about dynamic phenomena. Many static phenomena are interesting as well: calculations of risk in uncertain environments, one-shot games of imperfect information, inference of network structure and properties of observed networks, and others. But we are chiefly interested in dynamic phenomena because, as reasoning agents ourselves, we interact with the world around us in a dynamic way. Dynamic interactions are at the core of the language of science; Bayes's theorem is fundamentally a statement about how best to update posterior probabilities of events as we

obtain new information with the passage of time.

Our focus will be on systems that change in time, either through exogenous shocks, agent adaptation and strategic interaction, or evolutionary pressure. The models that we develop in Chs. 2 and 4 each are fundamentally dynamic, though the way in which agents interact with one another through time differs. In the ABM, cooperation and competition occurs intra-generation, but changes in model composition arising from application of selective pressure occur inter-generation in evolutionary time. In the model of electoral interference, agents are explicitly rational and construct optimal closed-loop policies through backward induction.

1.2 Outline

We now briefly outline the content of this work.

1.2.1 Chapter 2

In Ch. 2, we introduce an agent-based model of a financial market. This market is based on a new type of auction type, the frequent batch auction (FBA), that can be used to reduce the potential profitability of high-frequency trading algorithms [14, 15]. While we do not make normative comments on this objective, we do outline the differences between an FBA and a continuous double auction, the auction type more prominent in today's markets, review some of the existing literature on FBAs, and describe the logic used in the implementation of the auction.

Using this agent-based model, we conduct two experiments on the interplay between evolutionary dynamics and market microstructure. We first explore the extent to

which the design of the evolutionary mechanism affects volatility in the ABM. Agents have "volatility preferences" that control the degree to which they gain utility from increasing or lowering the variance about the mean of the order prices that they submit. We show that selection mechanisms incorporating a quantile-based component are associated with lower aggregate volatility preferences and hence with lower macro volatility across the market at large. The second experiment demonstrates what we believe is a new paradigm in training automated trading strategies. Instead of training strategies on past market data in an effort to predict future market prices or returns, we show that it is possible to instead design an ABM to be an accurate simulacrum of a real financial market, and then train or evolve trading strategies in the ABM. The trained or evolved strategies are subsequently used to trade in real markets. We demonstrate that trading algorithms evolved in this way are profitable in a backtesting environment.

1.2.2 Chapter 3

Putting aside the modeling for a chapter, we introduce a qualitative, shape-based, timescale-independent similarity search algorithm, termed the Shocklet Transform and Ranking (STAR) algorithm, that is used for annotating shocks, spikes, and cusps in observed sociotechnical time series. Inspired by the diverse behavior exhibited by time series of word usage on Twitter, one integral piece of this algorithm is the Discrete Shocklet Transform (DST), a time-space integral transform that uses archetypal kernels and actions of the 2-dimensional reflection group to extract shock-, spike-, or cusp-like dynamics at all timescales from N sociotechnical time series. These extracted dynamics are then combined into N indicator time series that display the

degree to which the original time series exhibit these dynamics at all timescales. From the indicator time series, a univariate ranking time series (first-most shocklike, second-most shocklike, etc.) and defines the observations of an $N+1$-dimensional Markov chain for which the transition probability $p_{n'n}$ represents the probability of switching from time series n being the most shocklike to time series n' being the most shocklike [1]. We then demonstrate the utility of the DST and STAR in annotating functionally-anomalous behavior in sociotechnical time series ranging from the data that initially inspired our construction of this algorithm (time series of word usages on Twitter) to asset price data and U.S. macroeconomic indicators of recesesion.

1.2.3 Chapter 4

In our final chapter, we return to modeling with a bare-bones model of rational electoral interference. After justifying our assumption that an election not subject to foreign power interference is reasonably well-described by a random walk in a latent space, we derive plausible goals of foreign intelligence agencies wishing to interfere with elections and the corresponding goals of domestic intelligence agencies who wish to stifle such interference. We derive a system of coupled nonlinear partial differential equations that describe the value functions of these actors, solve them numerically, and derive analytical results in some simplified cases. In particular, we are able to use path integral control when one player credibly commits to playing a particular deterministic strategy for the entire game, and we derive the analytical solutions of the value function when this deterministic strategy is constant over the entire time of the

[1] The $+1$ in the dimensionality of the Markov chain is due to the simple fact that, at time t, perhaps none of the timeseries are shocklike. In this case we just say the most shocklike timeseries is $\emptyset$.

game. We then turn to a recent example of electoral interference by foreign intelligence agencies: Russian interference in the U.S. 2016 presidential election contest. We fit a discrete-time version of our analytical model, represented as a Bayesian structural time series model, to observed election time series and time series of tweets by Russian intelligence bot accounts. Then, through a Gaussian process optimization procedure, we fit the parameters of our analytical model using the latent time series inferred by the structural time series model.

1.3 Citations

The material in this dissertation is drawn largely from published work and work that is, at time of writing, in review.

- Dewhurst, David Rushing, Michael Vincent Arnold, and Colin Michael Van Oort. "Selection mechanisms affect volatility in evolving markets." In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 90-98. ACM, 2019.

- Dewhurst, David Rushing, Yi Li, Alexander Bogdan, and Jasmine Geng. "Evolving ab initio trading strategies in heterogeneous environments."
  arXiv preprint arXiv:1912.09524 (2019). (Submitted to GECCO 2020.)

- Dewhurst, David Rushing, Thayer Alshaabi, Dilan Kiley, Michael V. Arnold, Joshua R. Minot, Christopher M. Danforth, and Peter Sheridan Dodds. "The shocklet transform: a decomposition method for the identification of local, mechanism-driven dynamics in sociotechnical time series." *EPJ Data Science*

9, no. 1 (2020): 3.

- Dewhurst, David Rushing, Christopher M. Danforth, and Peter Sheridan Dodds. "Noncooperative dynamics in election interference."
  arXiv preprint arXiv:1908.02793 (2019). (Accepted, *Physical Review E.*)

Chapter 2

Agent-based modeling of modern financial markets

2.1 Micro-to-macro volatility in frequent batch auctions

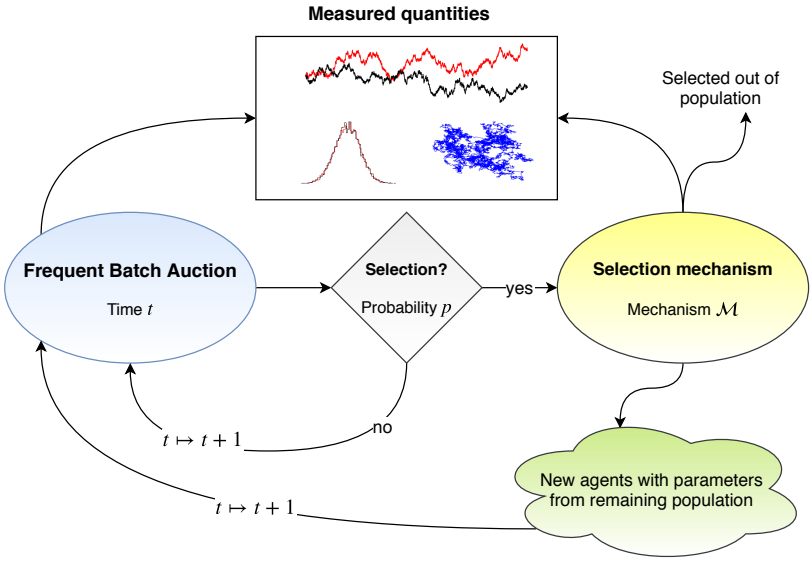

Figure 2.1: A cartoon of the financial system considered here is shown. Agents interact via the mechanism of a frequent batch auction, explained in Section 2.1, and are subject to a type of probabilistic selection mechanism that discards agents with low fitness, which is here defined by profit, and replaces discarded agents with new agents whose parameters are drawn from the distribution of parameters among remaining agents. Statistics from market activity and the selection process are gathered during iterations of the simulation and subsequently analyzed.

The concept of adaptive financial markets has been studied extensively in quantitative finance for nearly twenty years. The efficient markets hypothesis (EMH), which in its weakest form states that the price of an asset should, under conditions including costless information and agents with rational expectations about the future, reflect all publicly-available past information, has been an influential starting point for the study of financial theory since its initial publication in the late 1960s [16]. However,

there is empirical evidence that this hypothesis does not hold. A well-documented momentum effect exists for asset prices: assets that have done well (poorly) in past time periods will tend to do well (poorly) in future time periods, for periods ranging up to a year in the future [17]. In addition, there have been objections to the rational expectations assumption of EMH on a theoretical basis [18, 19, 20]. Critics of the EMH have proposed a so-called "adaptive-markets hypothesis" (AMH), in the framework of which the population of agents is in constant flux, adapting to changing market forces and subject to evolutionary pressure [21]. The rise of high-frequency trading (HFT) in response to a shift in the regulatory environment in U.S. asset markets in the mid-2000s is one factor that has lent credence to the AMH theory [22, 23, 24].

As a result of the apparent adaptive nature of modern financial markets, there has been substantial application of agent-based model (ABM) methods to model various market features of interest [25, 26, 27]. Such models often assume constant a particular selection mechanism by which agents of low fitness (usually, low profitability) are selected out of the market and agents of higher fitness remain [28, 29, 30]. However, the design of the selection mechanism may have a material effect on measurable quantities in the marketplace, such as price or return time series, preferences (parameters) of high-fitness agents, and volatility.

In this work, we analyze the role of various selection mechanisms in determining the preferences of a population of evolving zero-intelligence agents interacting through the means of an auction mechanism. Comparing two fundamentally distinct mechanisms—one a global mechanism based on population profit quantiles and the other a local mechanism based on sample profitability—we show that this choice not only affects

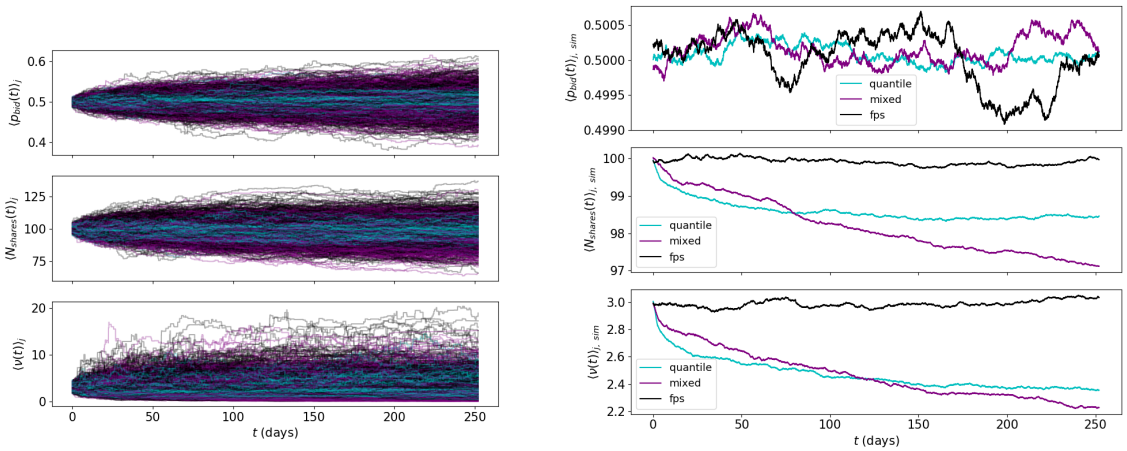

Figure 2.2: Means and standard deviation of parameter time series differ by selection mechanism. The left panel displays parameter time series averaged over agents; a single time series is plotted for each run of the simulation. The right panel displays parameter time series averaged over both agents and runs of the simulation. Overall, the quantile selection mechanism leads to lower spatial standard deviations across runs of the simulation, as can be observed in the left panel. While both the quantile and mixed selection mechanisms show decaying average N_{shares} and ν, fitness-proportionate selection shows no such behavior. The fitness-proportionate selection mechanism shows larger variation across runs of the simulation in these variables as well, with much larger extreme values of ν than either of the other mechanisms. When averaged over both agents and runs of the simulation, p_{bid} shows effectively no variation in time.

the dynamic behavior and distribution of agent parameters as shown in Figures 2.2 and 2.3, but also has a significant effect on micro-macro volatility correlations. We find that incorporating local fitness-proportionate selection greatly increases the correlation between a micro-level, risk aversion parameter and macro-level volatility as measured by standard financial econometric machinery, compared to purely quantile-based selection.

Theory and simulation

We focus our attention on the mechanism by which agents of low fitness—unprofitable agents— are selected out of the market. In real-world financial markets, agents whose

trading strategies produce low returns on capital can experience an outflow of funds to agents whose strategies produce better returns as investors seek the highest possible return subject to their risk preferences. In a world of perfect information, firms would thus be selected out of a market according to a type of fitness-proportionate selection. Real financial markets—and markets of all kinds—are rife with information asymmetries [31, 32]; here, we focus on the situation of perfect information to highlight the importance of the selection mechanism on macro-level observables.

Agent i's fitness function at time t is given by its profit at that time, defined as

$$\pi_i(t) = c_i(t) + s_i(t)X(t), \tag{2.1}$$

where c_i and s_i are the amount of cash held by agent i (units of currency), and number of shares of the asset held by agent i, respectively, and X is the price of the asset. Agents are permitted to "sell short": they are not restricted to have a non-negative amount of cash. Agents are zero-intelligence [33, 34] in the sense that their actions are purely random given a set of parameters; agents do not adapt in our model but are subject to evolutionary pressure across generations. The behavior of an agent is determined by three parameters: $p_{\text{bid},i}$, the probability of submitting a bid order in a time period given that the agent trades in that time period; $N_{\text{shares},i}$, the mean number of shares submitted by the agent in a time period; and ν_i, the so-called "volatility preference" of the agent, the role of which we will describe presently. Given the asset price at time t, $X(t)$, the agent submits a bid order with probability $p_{\text{bid},i}$ (equivalently, an ask order with probability $1 - p_{\text{bid},i}$) with number of shares distributed as $N_i(t) \sim \text{Poisson}(N_{\text{shares},i})$ and price distributed according to the random

variable

$$X_i^{(\text{order})}(t+1) = X(t) + \nu_i u_i(t), \tag{2.2}$$

where $u_i(t) \sim \mathcal{U}[-1, 1]$. The volatility preference parameter thus encodes a measure of regard for the current price level $X(t)$: low ν_i implies a preference for the current price level, while larger values lead to larger moves in both positive and negative directions. This parameter is interpreted as a measure of risk aversion (small ν) or risk neutrality / risk seeking (large ν).

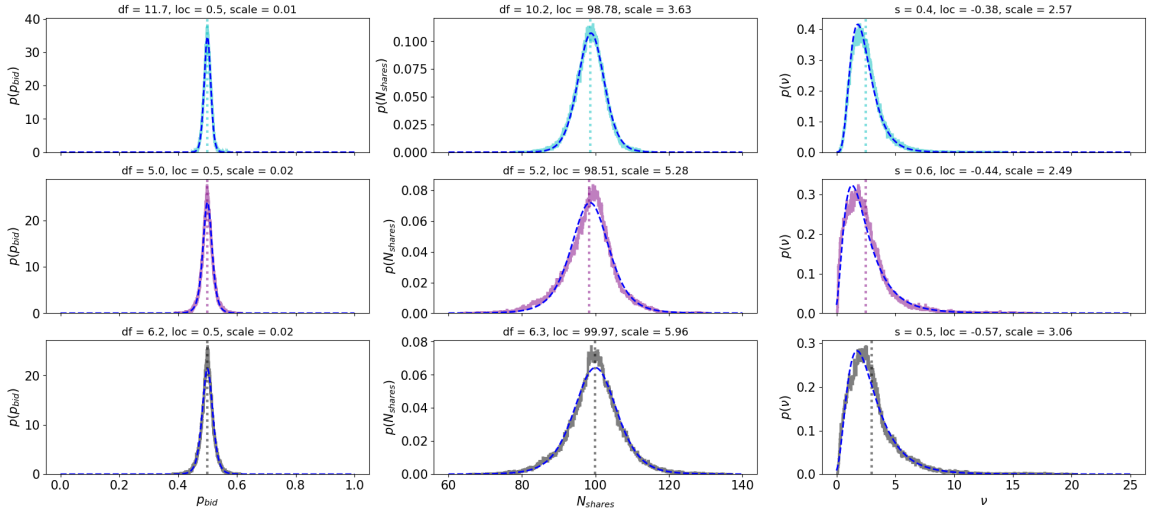

Figure 2.3: When uncoupled from time, distributions of parameters are similar across selection mechanisms. These distributions are calculated by computing the empirical pdfs over the union of time series of parameters over all points in time and runs of the simulation. The mixed selection mechanism displays the heaviest tails in the distributions of p_{bid} and N_{shares}, followed by the fitness-proportionate mechanism. From top to bottom: the quantile-based, mixed, and fitness-proportionate mechanisms. The blue dashed curves and titles indicate optimal fits to the empirical distributions as computed using maximum likelihood estimation. The distributions of p_{bid} and N_{shares} are well-fit by a **t***-distribution, while the distribution of ν is well-fit by a log-normal distribution.*

Market price is determined by a frequent batch auction (FBA), introduced by Budish *et al.* as a response to HFT strategies [15, 14], which we now describe briefly. Modern

financial markets primarily use a continuous double auction (CDA) mechanism to match buyers and sellers, though FBA has recently attracted much theoretical and intellectual property interest [35, 36], and batch auctions more generally have been in use since at least 2001 on the Paris Bourse [37]. CDA and FBA share several attributes. Both mechanisms are double-sided mechanisms in which any number of buyers and sellers may participate, and participants may enter or leave the market at any time under both mechanisms. Both mechanisms also maintain an order book, which accumulates orders that have not yet been executed. In practice, both mechanisms feature a similar price-time execution priority for resting orders, though the implementation may vary slightly. In other words, orders that have a better price, bids with higher prices or asks with lower prices, are executed first. Ties in price are broken by the age of the order, with older orders executing first.

CDAs allow agents to submit orders at any time, and these orders are immediately matched against resting orders if possible. Orders that are not immediately executed will be added to the order book, where they will wait for a counter-party to accept their conditions. This procedure results in trading that occurs continuously, aligning with the name of the mechanism. On the other hand, FBAs divide trading into discrete intervals. Within each interval agents may submit orders at any time, which are then placed in the order book. At the end of a trading interval, a single uniform execution price is selected by locating the intersection of the supply and demand curves (i.e. price and quantity of orders from both sides of the market are used to identify the execution price). Orders to buy with a limit price at least as high as the selected execution price and orders to sell with a limit price at least as low as the selected execution price are then eligible to execute. Eligible orders are then matched

together following price-time priority, i.e. bids with higher prices and asks with lower prices are matched first, with ties broken by order age, and further ties broken by uniform random selection. The orders that did not execute at time t remain in the book and are reconsidered for execution in future time periods until such time as the matching engine considers them to be "stale", or too old for consideration. The implementation of FBA considered here sets the maximum allowed time for an order to remain in the book to be 24 time periods, or one day.

Since the aim of this work is to understand the effects of selection pressure and different selection mechanisms on macro-statistics of market activity, we attempt to abstract away other details of real-world asset markets. Though the U.S. National Market System (NMS) is a fragmented market with no fewer than thirteen exchanges operating at time of writing [38], we consider only a single exchange and matching engine here. As noted above, agents are effectively zero-intelligence; though they are subject to selective pressure and thus the population of agents may become more profitable over time as weak agents are selected out, individual agents do not adapt to changing market circumstances.

2.1.1 Selection mechanisms

Selection occurs with constant probability of $p_{\text{selection}} = \frac{1}{24}$ each time period, so that there is a selection event in one out of every 24 time periods (hours) on average. We consider three selection mechanisms: a quantile-based mechanism (truncation selection), denoted by $\mathcal{M}_{\text{quantile}}$; a type of fitness-proportionate selection, $\mathcal{M}_{\text{fps}}$, and a mixture of the two mechanisms, $\mathcal{M}_{\text{mixed}}$, each of which is a well-known selection method [39]. The quantile-based mechanism removes agents i whose profit satisfies

$\pi_i(t) < F^{\leftarrow}_{\pi(t)}(q)$, where q is a quantile (number between 0 and 1) and $F^{\leftarrow}_{\pi(t)}$ is the quantile function of the profit distribution across all agents active at time t. We set $q = 0.1$ to remove the bottom 10% of agents each time the quantile-based mechanism is activated. The fitness-proportional selection mechanism is a standard implementation of such a procedure: a random sample $\mathcal{S}(t)$ of agents is selected from the population and each is kept in the population with probability given by $p_i(t) = \frac{\pi_i(t)}{\sum_{j \in \mathcal{S}(t)} \pi_j(t)}$. We set $|\mathcal{S}(t)| = 10$ in this implementation. The mixed selection mechanism interpolates between $\mathcal{M}_{\text{quantile}}$ and $\mathcal{M}_{\text{fps}}$. When a selection event occurs, with probability $\frac{1}{2}$ the mechanism $\mathcal{M}_{\text{quantile}}$ is used and with probability $\frac{1}{2}$, $\mathcal{M}_{\text{fps}}$ is used.

When agents are selected out of the population, new agents are added to replace the ones that have exited so that the number of agents in the population is conserved. We set the number of agents $N_{\text{agents}} = 100$ in each run of the simulation. When new agents enter the model after a selection event, with probability $p_{\text{innovation}}$ they draw their governing parameters (p_{bid}, N_{shares}, and ν) from stationary probability distributions that do not change with selective pressure, and with probability $1 - p_{\text{innovation}}$ they draw their governing parameters from the distributions of these parameters among the members of the population of agents that did not get selected out of the market. In this work, we set $p_{\text{innovation}} = 0.01$. We choose these selection mechanisms not because they are in some way optimal methods for selecting individuals in an evolving system—in fact, the disadvantages of fitness-proportionate selection are well-documented [40]—but for their interpretation in the context of a financial market. The quantile-based method models an environment in which an investing public (individuals, firms, etc.) actively avoid firms that are performing badly in the market, but do not actively seek out firms whose profits are the highest.

In contrast, a fitness-proportionate scheme models a scenario in which investors seek out the firms that have the highest total profits and allocate their funds to these firms in proportion to their past performance. We also included a control simulation model in which no selection was present and all agents initially in the simulation at time $t = 0$ remained in the simulation for the entire time.

We turn briefly to a theoretical model of the evolution of agents' parameters: p_{bid}, N_{shares}, and ν. For the sake of convenience we pass to a continuous time description, though the discrete time of the simulation is recovered by simply setting $dt = \frac{1}{24}$ days. We assume that prices evolve according to a zero-mean Lévy flight,

$$dX(t) = \sigma_X dL_X^{(\alpha)}(t),\ X(0) = X_0, \tag{2.3}$$

with tail exponent $\alpha \in (1.7, 2)$ as suggested by Mandelbrot [41]. This model has been shown to give superior fit to real data when compared with the geometric Brownian motion model of asset prices [42, 43]. Since any agent whose bid probability deviates too far from the natural equilibrium of $p^*_{\text{bid}} = \frac{1}{2}$ will soon become rapidly unprofitable and hence be selected out of the market, we assume p_{bid} evolves according to a type of Ornstein-Uhlenbeck process,

$$dp_{\text{bid}}(t) = \theta_{p_{\text{bid}}}(p^*_{\text{bid}} - p_{\text{bid}}(t))\ dt + \sigma_{p_{\text{bid}}} dL^{(\alpha)}_{p_{\text{bid}}}(t). \tag{2.4}$$

In contrast, there is no logical steady state for N_{shares}, so we assume that its evolution is governed by a standard random walk with heavy-tailed increments arising from the

auction mechanism,

$$dN_{\text{shares}}(t) = \mu_{N_{\text{shares}}}\, dt + \sigma_{N_{\text{shares}}}\, dL^{(\alpha)}_{N_{\text{shares}}}(t). \tag{2.5}$$

The parameter $\mu_{N_{\text{shares}}}$ is interpreted as evolutionary drift. The interpretation of volatility preference ν as a measure of risk aversion (small ν) or risk neutrality / seeking (large ν) gives insight into a possible model for its evolution. Simply put, volatility preference increments in proportion to the current level of volatility preference: if the population is risk averse, the variation in volatility preference should be low; if the population is risk neutral or risk-seeking, the variation in volatility preference will likely be high. Incorporating an evolutionary drift term, a reasonable model for this phenomenon is

$$d\nu(t) = \nu(t)[\mu_\nu dt + \sigma_\nu dL^{(\alpha)}_\nu(t)]. \tag{2.6}$$

For example, orders submitted according to Eq. 2.2 with ν much larger than the population average are unlikely to be executed if the resultant price is favorable to the submitting agent (i.e., very high ask price or very low bid price relative to the last equilibrium price) and will result in a large financial loss to the agent if the resultant price is likely to be executed (i.e., very high bid price or very low ask price).

We seek an understanding of the effects of the selection mechanism on micro- and macro-market statistics. Are there cross-mechanism differences between optimal parameter combinations, or, more fundamentally, is there a steady-state optimal parameter combination at all? How do the time series of parameters—which, in a real financial market, would be unobservable—affect macro-observable quantities such as

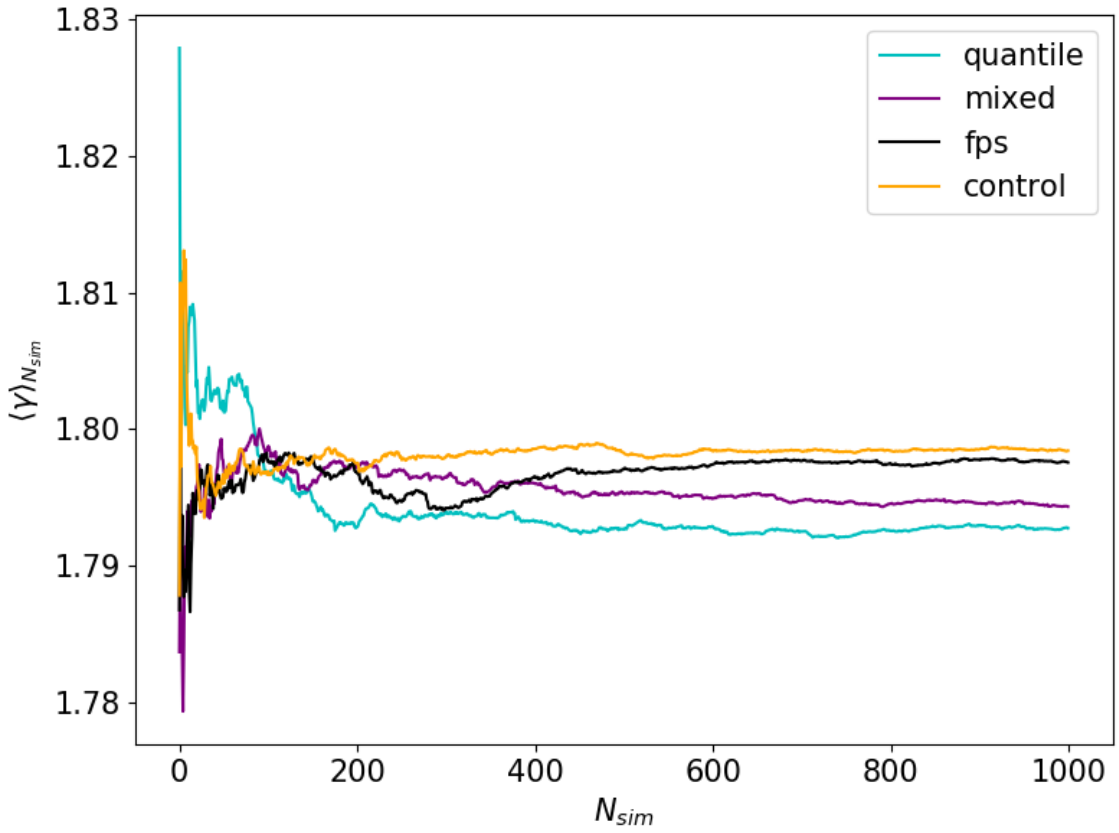

Figure 2.4: The mean power spectral density (PSD) exponent of population price time series, $\langle\gamma\rangle_{N_{sim}} = \frac{1}{N_{sim}}\sum_{n=1}^{N_{sim}}\gamma_n$, where γ_n is defined by $S_{xx}(\omega) \sim \omega^{-\gamma_n}$. All PSD exponents converge to a value near $\langle\gamma\rangle_{N_{sim}} \sim 1.8$, though the quantile mechanism has the largest exponent and hence the average price time series associated with the quantile mechanism is less autocorrelated than the others.

leptokurticity of returns or volatility? To answer these questions, we first characterize basic macro properties of the simulations under each selection mechanism. Aside from the price $X(t)$ and return $r(t) = \log_{10} X(t) - \log_{10} X(t-1)$ time series, we calculate the price power spectral density, defined by $S_{xx}(\omega) = \hat{X}(\omega)\hat{X}^{\dagger}(\omega)$, where we have defined the Fourier transform on the interval $[0, T]$ by

$$\hat{X}(\omega) = \frac{1}{\sqrt{T}} \sum_{t=1}^{T} X(t) e^{-i\omega t} \Delta t, \tag{2.7}$$

where $\Delta t = \frac{1}{24}$, so that the units of the Fourier transform are 1/days. For financial price time series we expect $S_{xx}(\omega) \sim \omega^{-\gamma}$, where $\gamma \in (1.7, 2)$. Brownian motion has

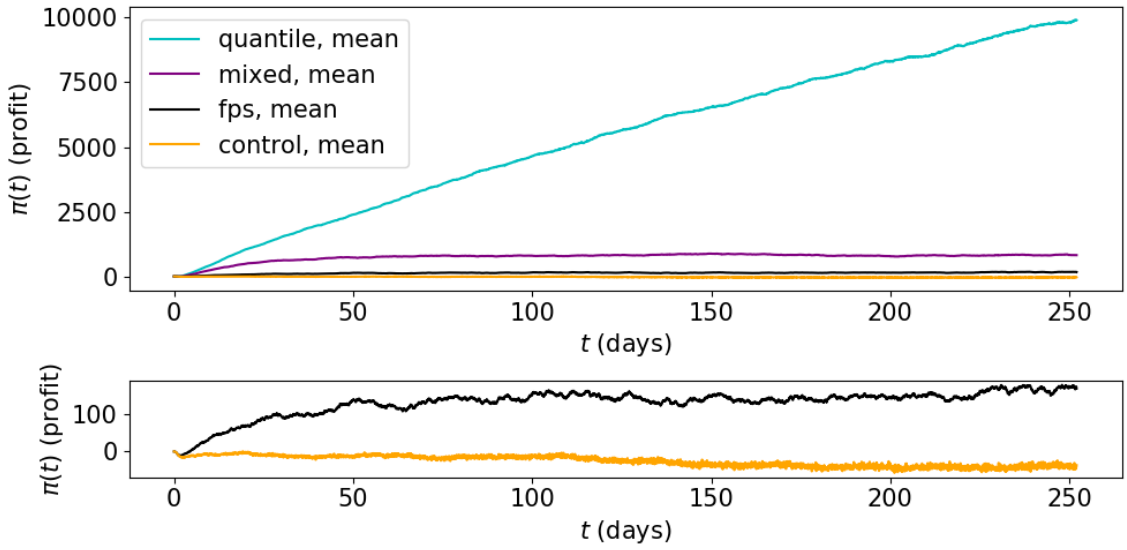

Figure 2.5: Mean profit levels differed by selection mechanism. The quantile (truncation) selection mechanism lead to average profits that were approximately an order of magnitude higher than that of the second-most profitable mechanism, the mixture of fitness-proportionate selection and quantile selection. While returning positive average profits, fitness-proportionate selection was the least profitable of the non-control selection mechanisms. In this context, average profit is defined by $\langle \pi(t) \rangle_{j,sim} = \frac{1}{N_{sim} N_{agents}} \sum_{n=1}^{N_{sim}} \sum_{j \text{ active at time } t} \pi_{j,n}(t)$

$\gamma = 2$, while real asset markets exhibit $\gamma \sim 1.8$ in price dynamics [41, 44]. Time series of the parameters $p_{\text{bid},j}$, $N_{\text{shares},j}$, and ν_j are described and their distributions are fit and compared with distributions predicted from the theoretical models described above. Finally, we analyze the link between the agent-level micro-volatility parameters ν_j and macro-volatility as measured from price or return time series and remark on its differentiation by selection mechanism.

Results

We ran 1000 runs of the artificial asset market simulation for each selection mechanism (control, $\mathcal{M}_{\text{quantile}}$, $\mathcal{M}_{\text{fps}}$, and $\mathcal{M}_{\text{mixed}}$) for a total of 4000 simulations. Each simulation was composed of 24 "hour" trading periods in each trading "day". A total of 252 trading days per year (in analogy with the calendar of the U.S. national market system) resulted in a total of 6048 trading periods per simulation. The number of agents in each simulation was held constant at 100. To determine that the number of runs of the simulation was adequate for the calculation of population averages, we generated reruns of the simulation until temporal averages of the population price time series power spectral density exponents appeared to converge. This convergence is displayed in Figure 2.4.

The mean profitability of agents under each selection mechanism is displayed in Figure 2.5. Here, we define an average over both runs of the simulation and active agents, *viz.*

$$\langle \pi(t) \rangle_{j,\ \text{sim}} = \frac{1}{N_{\text{agents}} N_{\text{sim}}} \sum_{n=1}^{N_{\text{sim}}} \sum_{j \text{ active at time } t} \pi_{j,n}(t). \tag{2.8}$$

The purely quantile-based mechanism displays average profitability that is over an order of magnitude greater than either $\mathcal{M}_{\text{mixed}}$ or $\mathcal{M}_{\text{fps}}$, while $\mathcal{M}_{\text{mixed}}$ was still much

more profitable on average than was $\mathcal{M}_{\text{fps}}$. This differentiation is likely to due to the fact that $\mathcal{M}_{\text{quantile}}$ selects out the ten worst-performing individuals each time it is active, while $\mathcal{M}_{\text{fps}}$ selects out on average $|\mathcal{S}(t)| - \sum_{j \in \mathcal{S}(t)} p_j = 9$ individuals that are randomly sampled from the population; while the individuals selected out are, on average, the worst performing individuals in that particular $\mathcal{S}(t)$, they are by no means the worst-performing individuals in the entire population. Though this implementation of $\mathcal{M}_{\text{fps}}$ results in significantly less selective pressure on the population than does $\mathcal{M}_{\text{quantile}}$, this choice is made to hold constant the number of individual agents involved in the selection step of the market simulation.

Agents' parameters—the probability of submitting a bid, $p_{\text{bid},j}$, the mean number of shares submitted in an order $N_{\text{shares},j}$, and the volatility preference ν_j—were influenced by the choice of selection mechanism. Overall, $\mathcal{M}_{\text{quantile}}$ was associated with lower standard deviations of parameter time series as calculated over runs of the simulation. Figure 2.2 displays parameter time series for all runs of the simulation in the left panel, and averages over runs of the simulation in the right panel. Both $\mathcal{M}_{\text{quantile}}$ and $\mathcal{M}_{\text{mixed}}$ showed time decay toward lower values in N_{shares} and ν when averaged over both active agents and runs of the simulation. On the contrary, $\mathcal{M}_{\text{fps}}$ showed no decay in either parameter when the same average was performed. When decoupled from time, distributions of the parameters showed remarkable similarity across mechanisms, showing evidence for a unified underlying evolutionary model as proposed in Eqs. 2.4 - 2.6, the parameters of which depend on the selection mechanism. These time-decoupled distributions are displayed in Figure 2.3.

2.1.2 Volatility correlation

Since it seems reasonable that a fitness-proportionate selection mechanism most closely approximates the selection mechanism operating in today's financial asset markets, we are particularly interested in correlations between micro-volatility—agents' volatility preferences ν_j—and macro measures of volatility. We are interested in the effects of mechanism on these macro measures of volatility, and particularly wish to test if micro-volatility is correlated with macro-volatility in the cases of $\mathcal{M}_{\text{fps}}$ and $\mathcal{M}_{\text{mixed}}$, as this could provide some insight into how volatility is generated in real financial markets. Macro-volatility—volatility as measured from market-wide statistics such as price and returns—is often modeled using a generalized autoregressive conditional heteroskedasticity (GARCH) model [45], which, in its most basic form, hypothesizes that log returns $r(t) = \log_{10} X(t) - \log_{10} X(t-1)$ can be decomposed as

$$r(t) = \mu + \varepsilon(t) \tag{2.9}$$

$$\varepsilon(t) = \sigma(t) z(t) \tag{2.10}$$

$$\sigma^2(t) = \xi + \alpha \varepsilon^2(t-1) + \beta \sigma^2(t-1), \tag{2.11}$$

where $z(t) \sim \mathcal{N}(0,1)$. For each simulation, we compute a GARCH model of the form given above and calculate the Spearman correlation coefficient $\rho(\langle \nu \rangle, \sigma)$ between the average agent volatility preference $\langle \nu(t) \rangle_j = \frac{1}{N_{\text{agents}}} \sum_{j \text{ active at time } t} \nu_j(t)$ and the fitted volatility $\sigma(t)$. Figure 2.6 displays $\langle \nu(t) \rangle_j$ and $\sigma(t)$ for an arbitrarily chosen run of the simulation. Figure 2.7 displays the empirical probability density function (pdf) of $\rho(\langle \nu \rangle_j, \sigma)$ across all non-control simulations. (The pdf of correlations for the control

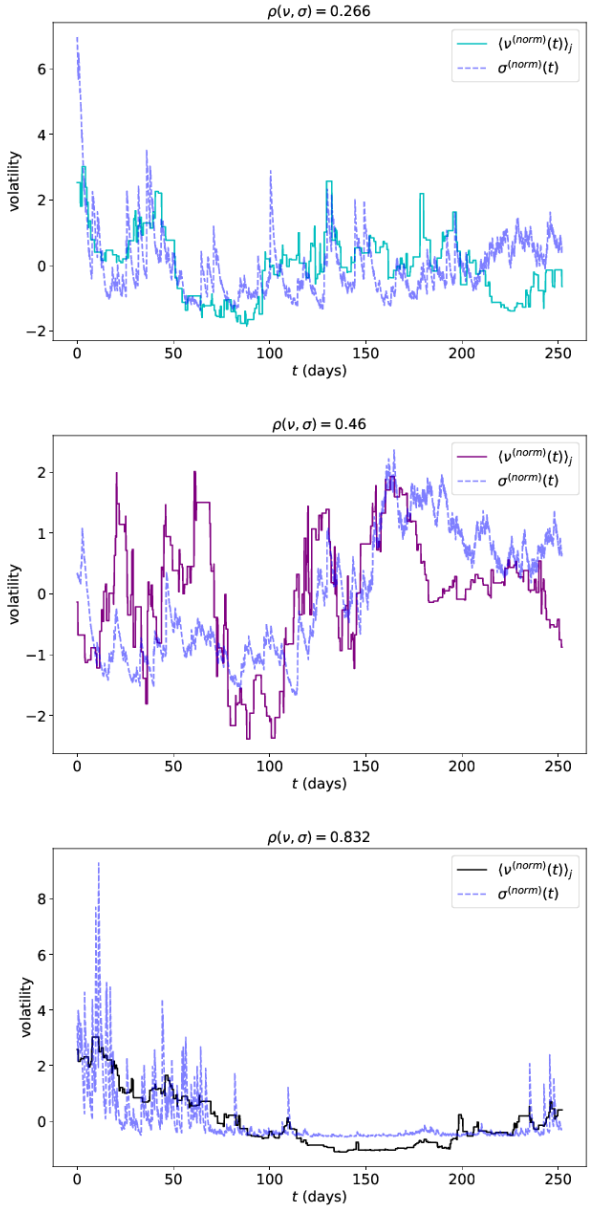

Figure 2.6: Micro-macro volatility correlation varies by selection mechanism. We chose an arbitrary rerun and show the average volatility preference, $\langle \nu(t) \rangle_j = \frac{1}{N_{agents}} \sum_{j \text{ active at time } t} \nu_j(t)$*, displayed as a solid curve, plotted against macro-volatility calculated as the solution of a* $GARCH(1,1)$ *process, displayed as a dashed curve. After calculation, these processes were normalized to have zero mean and unit variance for display on the same scale. From top to bottom:* $\mathcal{M}_{quantile}$*,* $\mathcal{M}_{mixed}$*, and* $\mathcal{M}_{fps}$*.*

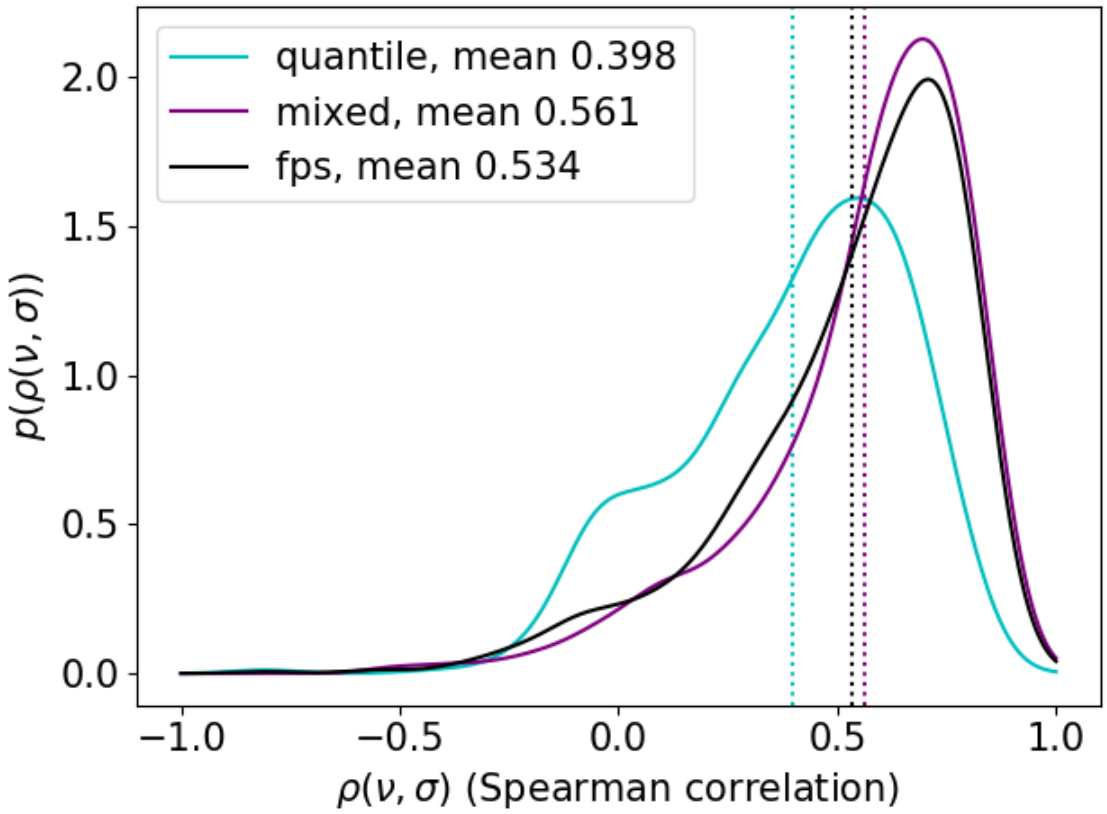

Figure 2.7: Micro and macro volatility measures are highly correlated when fitness-proportionate selection is included in the selection mechanism (i.e., the mechanism is either mixed or fitness-proportionate). There is correlation between micro and macro volatility under the pure quantile mechanism, but the effects of agents' volatility preferences are muted in comparison. Calculated values were used in a kernel density estimate, plotted above, computed using Gaussian kernels and the Silverman rule for bandwidth estimation.

is sharply peaked about zero and uninteresting as there is no evolution of ν_j in this case.) The pdf of correlation coefficients for $\mathcal{M}_{\text{quantile}}$ is bimodal, with one mode about zero and another near $\rho = 0.5$, while for $\mathcal{M}_{\text{mixed}}$ and $\mathcal{M}_{\text{fps}}$ the pdfs are are peaked near $\rho \simeq 0.75$ with a long left tail.

Since the theoretical models for the evolution of agents' parameters given by Eqs. 2.4 - 2.6 contain nine free parameters in total, to assess their suitability as a first-order theoretical model of the evolutionary phenomena occurring here we must fit these parameters from the data generated by the agent-based model. To do this we hypothesize a parametric form $p_{\text{theo}}(x|\beta)$ for each distribution: $p(p_{\text{bid}})$, $p(N_{\text{shares}})$, and $p(\nu)$. The optimal values of β are defined as the vector that minimizes

$$\int_{x\in\Omega} p_{\text{abm}}(x) \log\left(\frac{p_{\text{abm}}(x)}{p_{\text{theo}}(x|\beta)}\right)\, dx, \tag{2.12}$$

the Kullback-Leibler (KL) divergence of the theoretical distribution away from the distribution produced by the ABM. The domain of integration Ω is defined as all observed values of the quantity x for each time step and each run of the simulation. This integral is minimized using differential evolution [46], at each iteration of which a number of simulations of the theoretical model Eqs. 2.4 - 2.6 are calculated and the maximum likelihood estimation of the parameter vector β is found, which is then substituted into the functional form of p_{theo} used in the definition of KL divergence.

Figure 2.8 displays comparisons between the fitted theoretical distributions and distributions arising from the ABM for $\mathcal{M}_{\text{mixed}}$. To emphasize that the restriction of the fit distribution to a parameterized form does not result in a model that fits the data poorly, random variates drawn from each model are drawn and their histogram

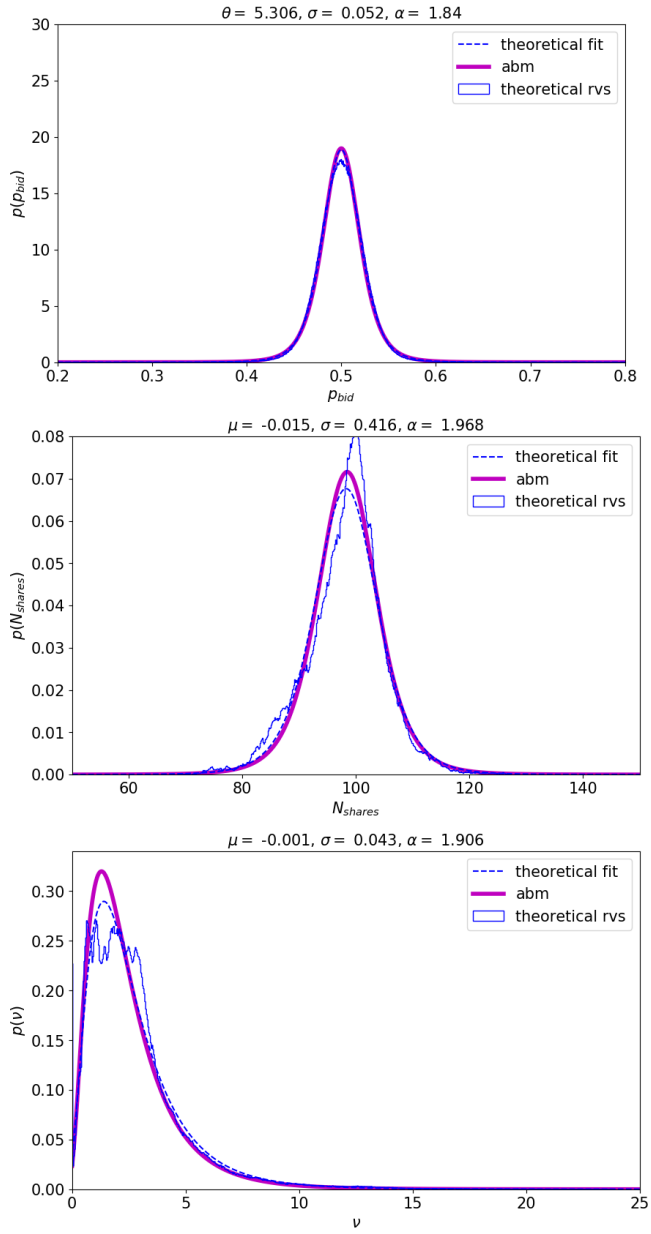

Figure 2.8: Parameters to theoretical models of p_{bid}, N_{shares}, and ν were fit using maximum likelihood estimation and differential evolution, as described in the text. Displayed here are the fit distributions of the theoretical models for the mixed mechanism in dashed blue curves, random variates drawn from the theoretical model in solid blue curves, and fit distributions of the ABM in magenta curves., Calculated optimal values of free parameters for each model are displayed in the title of each panel.

is plotted along with the fit distributions. The calculated optimal values of the free parameters for each model are displayed in the title of each panel. There is strong restorative force ($\theta_{p_{\text{bid}}} = 5.306$) to the equilibrium bid probability $p_{\text{bid}} = \frac{1}{2}$, while there is negative evolutionary drift in mean number of shares submitted per order ($\mu_{N_{\text{shares}}} = -0.015$) and volatility preference ($\mu_\nu = -0.001$).

Discussion

We find that choice of selection mechanism is associated with differential behavior of asset price spectra, agent parameter distributions and time series, and volatility. While the probability of submitting a bid order fluctuates regularly about its natural equilibrium value of $p^*_{\text{bid}} = \frac{1}{2}$ under all three mechanisms, the time series of the average number of shares traded and the volatility preference parameter varies functionally depending on the presence of a quantile-based component to the selection mechanism. When a quantile-based component is not present ($\mathcal{M}_{\text{fps}}$), these time series vary in the mean case very little from their initial values, with a slight upward trend. However, when a quantile-based component is present, in the mean case these series exhibit a steady trend toward lower values. In both N_{shares} and ν, $\mathcal{M}_{\text{mixed}}$ trends most strongly toward lower values and does not appear to converge in the time period covered by our simulation (252 days of trading once per hour), suggesting that longer simulation run times are necessary to discern the nature of the steady state of these parameters under mechanisms containing a quantile-based component, if such steady-states exist.

All three mechanisms show significant correlation between micro-volatility, as measured by the risk-aversion / volatility preference parameter ν, and market-wide volatil-

ity measured from the market price using standard econometric models (GARCH). All distributions of Pearson correlation coefficients of micro- and macro-volatility exhibited negative skew (more weight in the left-hand tail). The quantile-based mechanism displayed bimodality in this distribution, with a small peak near zero correlation and a large peak near $\rho = 0.5$. Contrasting with this, $\mathcal{M}_{\text{mixed}}$ and $\mathcal{M}_{\text{fps}}$ were unimodal, with peaks near $\rho \simeq 0.75$, displaying a strong median correlation between micro- and macro-volatility.

Taken together, these results paint a picture of nontrivial interaction between selection mechanism and market outcomes. Mechanisms that include a fitness-proportionate component show higher volatility than a purely quantile-based mechanism, and under those mechanisms micro-volatility is more highly correlated with observable macro-volatility, providing a possible mechanistic explanation for the generation of macro-volatility in real financial markets. However, mechanisms that contain a quantile-based component show significant evolutionary drift in the average number of shares submitted per order and in volatility preference. When taken along with the fact that these mechanisms produced far higher average profits than did the purely fitness-proportionate method, this suggests that lower values of these parameters are—in a population of zero-intelligence agents, at least—associated with higher average profit levels, possibly due to an increase in risk-aversion among the population of agents and a corresponding decrease in the frequency of agents that experience massive trading losses.

Our study has several areas on which future work could improve, the most important of which being our neglection of other selection mechanisms. There are far more—and more realistic!—mechanisms that provide a model for how agents may be removed

from, and added to, a financial market. Drawing definitive conclusions about the nature of market selection and competition from a study of only two fundamental mechanisms is ill-advised, and we decline to do this. Another shortcoming is our lack of variation of many parameters in this study. In order to understand these mechanisms in more depth, a detailed study of macro-observable market statistics as a function of, e.g., tournament size, quantile, and mixture probability between the two fundamental mechanisms is required. Future work should focus on inclusion of more and different selection mechanisms, as well as inclusion of more advanced agents.

2.2 Evolving trading agents

Ab initio artificial intelligence—algorithms that are capable of learning or evolving master-level performance from a zero-knowledge baseline in a task normally performed by humans—is a long-held goal of the field in general [47]. Recent years have seen substantial progress toward this goal [47, 48, 49]. One particular area of interest is the development of algorithms that are able to trade financial assets without human supervision. This problem is relatively difficult not only due to its fundamentally-stochastic nature (unlike the deterministic non-cooperative games of Go, chess, shogi, and Atari with which algorithms have had such success), but because of obvious economic incentives: if an algorithm has a non-transient ability to make a statistically significant positive profit, the owner of that algorithm stands to reap large financial gains.

Though there has been prior work on *ab initio* trading strategies, such work has fo-

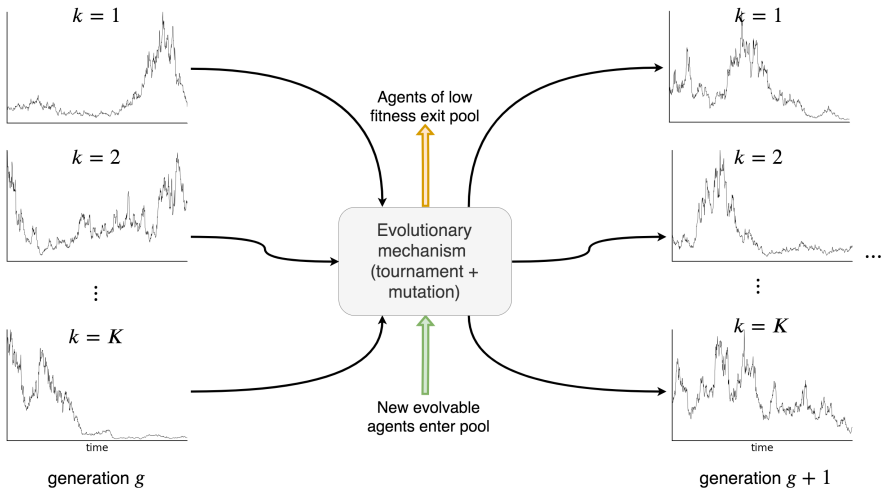

Figure 2.9: At each generation g of the evolutionary process, we initialize K independent markets in which agents (Sec. 2.2.3) interact via the matching engine described in Sec. 2.2.2. At the end of T timesteps, agents subject to evolution are pooled, selection is applied, and then new agents are introduced using the mechanism described in Sec. 2.2.4. The process then begins again in generation $g + 1$.

cused on small, homogeneous collections of agents that interact over shorter timescales than those considered in this study [50]. Trading strategies that use statistical and algorithmic methods more broadly are exceptionally common in the quantitative finance literature [51, 52, 53], and are used in practice with mixed results [54, 55, 56]. Evolutionary approaches to the development of trading strategies have focused on the development of technical trading rules using observed market data as the training dataset and then backtesting the evolved rules on out-of-sample test data. [57, 58]. Likewise, evolutionary computation and agent-based models have been used extensively to study the macro properties of artificial asset markets rather than specifically studying the micro properties of individual trading strategies [59, 60, 61, 62, 63, 64, 65]. However, to our knowledge there has been no academic study of the possibility of developing *in vivo* trading strategies using purely *ab initio* methods—trading strategies that train or evolve using artificial data only, and then in actuality trade on real asset

prices—which is the approach that we take here.

We pursue this objective for two reasons: first, achieving this goal would be a useful step in the development of evolutionary "self-play" techniques in the context of stochastic games with many players [66, 67, 68]; and second, this would demonstrate that the development of profitable trading strategies could be realized by attempting to simulate with increasing accuracy the underlying mechanisms of financial markets instead of by predicting future real market prices.

The paper proceeds as follows: in Sec. 4.2, we describe the theory and details behind our agent-based financial market model, including the design of the price-discovery (auction) mechanism, heterogeneous agent population, evolutionary algorithm (summarized graphically in Fig. 2.9), and method to convert evolved individuals into trading strategies; in Sec. 3.3 we summarize descriptive and quantitative results of the evolutionary dynamics and the performance of evolved trading algorithms back-tested on real data; and in Sec. 3.4 we discuss these results and provide suggestions for future work.

2.2.1 Theory and simulation

Our simulation methodology is based on an agent-based market model (ABM) composed of a matching engine and heterogeneous agents, which we describe in turn [1]. We then turn to an outline of the evolutionary mechanism, how it interfaces with the ABM, and the methodology by which we generate functional trading strategies from evolved agents.

[1] All source used in this project is open-source and available at https://gitlab.com/daviddewhurst/coco-neuro-trader-abm

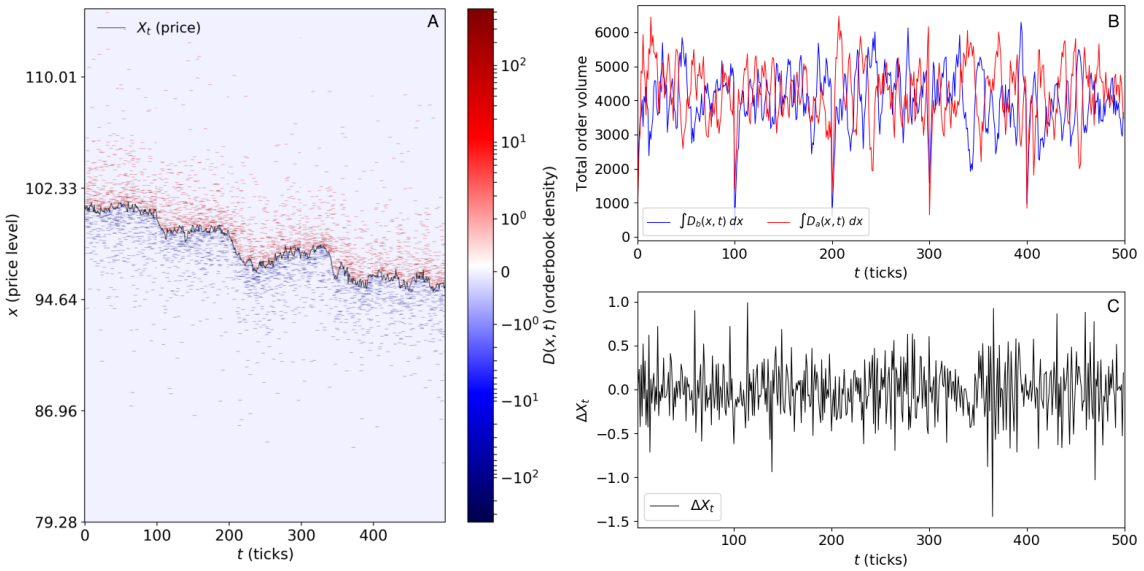

Figure 2.10: In panel A, we display an example orderbook corresponding with a very simple market simulation ($\alpha = (66, 33, 0, 0, 0, 0, 0)$ along with the resulting asset price time series. We denote bid interest by negative numbers (corresponding with positive $D_b(x,t)$ later) and ask interest by positive numbers (corresponding with positive $D_a(x,t)$ later). We display the time series of total bid and ask interest in panel B and the time series of ΔX in panel C. Differences in total bid and ask interest are strongly associated with changes in level of ΔX. The periodicity of large drops in $\Delta\hat{V}^{(b)}$ and $\Delta\hat{V}^{(a)}$ is due to end-of-day orderbook clearing by the matching engine.

2.2.2 Details of price discovery mechanism

At each timestep t, agents can submit orders to the matching engine, which attempts both find an equilibrium price for that timestep and to match orders with one another so that exchange of shares for cash can occur. Orders are described by a three-tuple, $o = (s, N^{(o)}, X^{(o)})$, consisting of the desired side $s \in \{\text{buy, sell}\}$, the number of shares that the agent would like to purchase or sell $N^{(o)}$, and the requested price at which the agent would like to transact $X^{(o)}$. The matching engine collects all orders submitted to it and matches bid orders with ask orders using a frequent batch auction (FBA) mechanism [14, 15]. This mechanism is an alternative to the continuous double auction (CDA) mechanism that is in use in most securities markets. Both CDAs and FBAs are double-sided auctions, meaning that they match multiple buyers with multiple sellers of an asset at the same time, unlike auctions that match a single seller with multiple buyers (e.g., English, Japanese, or Dutch auctions) [69, 70, 71]. However, CDAs and FBAs differ fundamentally as CDAs operate in continuous time and FBAs operate in discrete time. CDAs match orders as they are received; if the order cannot be immediately executed, it is placed into an order book where it waits to be matched with a future incoming order. In contrast, an FBA collects orders in discrete time trading intervals. At the end of each trading interval, orders are sorted according to price preference (bids are sorted from highest to lowest price, while asks are sorted from lowest to highest). Matching then occurs in price-time priority order, meaning that bids with a higher price and asks with a lower price are matched first. Ties in price are broken by age, where older orders—orders that are

already resting in the order book from previous batches—are matched first [2]. Orders submitted at timestep t that also do not execute at timestep t remain in the orderbook for consideration in future time periods. Orders that remain in the orderbook—and hence are not matched with new, arriving orders—past a certain amount of time are considered "stale" by the matching engine and are removed. In our implementation, we set this period of time to be one day (or 100 time increments). Agents are able to observe the price X_t at time t and the volume of resting bid (b) and ask (a) orders in the orderbook at price level x at time t, written $D_b(x,t)$ and $D_a(x,t)$. All statistics used by agents in calculating side of book, price level and number of shares to submit are functions of these observable random variables and of the agents' own internal state, which we describe in the next section. We display an example orderbook, along with the corresponding price trajectory, in panel A of Fig. 2.10.

2.2.3 Heterogeneous agents

Our agent population is heterogeneous, comprised of seven qualitatively distinct varieties (species) of agents that analyze statistical behavior of asset prices differently and concomitantly exhibit divergent trading behavior. We do not apply evolutionary pressure to six out of seven of these species; we thus separate the descriptions of the agents below into those not subject to evolutionary pressure (static agents) and those that do evolve.

[2] The price discovery algorithm used in our matching engine implementation is a modified version of the logic used in the Australian Securities Exchange's matching engine [72]. An open-source implementation of the matching engine is at https://gitlab.com/daviddewhurst/dragonbridge-abm.

Static agents

There are six types of agents in our model to which we do not apply evolutionary pressure. We give a brief overview of them below (the curious reader is encouraged to consult the supplementary information for more detail).

- *Zero-intelligence* (zi): Inspired by work on statistical aspects of asset markets [33, 34], these agents submit a random bid or ask order with order price uniformly distributed around the last market price and number of shares Poisson-distributed around a fixed value (here set to be 100 shares).

- *Zero-intelligence priceless* (zip): identical to zero-intelligence agents, except these agents submit "market" orders — orders that do not have a price but rather have first-priority execution and execute against the highest bid (for a market ask) or lowest ask (for a market bid).

- *Momentum* (mo): Momentum trading agents postulate that prices that have recently risen will continue to rise (and that prices that have recently fallen will continue to fall) [73, 74]. Our agents submit bid orders if the change in price is above some positive threshold and ask orders if the change in price is below a symmetric negative threshold.

- *Mean-reverting* (mr): postulate a return to some mean value of the asset price [75, 76]. When the asset prices moves above a rolling mean value, these agents submit ask orders; when the price moves below the rolling mean, they submit bid orders.

- *Market-making* (mm): market-making strategies attempt to profit off of small

price imbalances on either side of the orderbook; in doing so, they provide liquidity to the asset market [77, 78].

- *Fundamental-value* (fv): these agents have a certain fixed price set at which they "believe" the asset is fairly valued; if the asset price rises above that fundamental value, they submit ask orders, while if the asset price falls below it, they submit bid orders. Note that this approach differs from that of mr traders since fv traders' valuations of the asset do not change over time.

The typology of strategies illuminated here has a nonempty intersection with that introduced in the context of modern-day futures markets [79] but does exhibit some differences — in particular, we do not implement a pure "high-frequency trader" agent since this does not make sense in the context of an FBA [15, 80]. We implement such a wide variety of strategies so that, in order for evolving agents to achieve high fitness, they must perform well in many different environments. We believe this will increase the likelihood of high-fitness agents' performing well when confronted with real price data on which to trade, since real asset markets are also composed of heterogeneous agents [79].

Evolvable agents

We model technologically-advanced agents that are subject to evolutionary pressure as deep neural networks, denoted by f_θ (we will omit the vector of parameters θ when it is contextually unnecessary). These neural networks take as inputs changes in price, changes in total orderbook volume, and changes in internal state (cash, shares held, and profit) and output a three-tuple of side (bid or ask), number of shares in the

order, and price level of the order:

$$(s_t, N_t^{(o)}, \Delta X_t^{(o)}) = f(\Delta X, \Delta \hat{V}^{(b)}, \Delta \hat{V}^{(a)}, \Delta c, \Delta N, \Delta \pi) \tag{2.13}$$

We briefly mathematically describe the inputs of this function and their derivation from observable market statistics. The change in asset price from $t-1$ to t is given by $\Delta X_t = X_t - X_{t-1}$, while the change in cash (Δc), shares (ΔN), and profit ($\Delta \pi$) are defined analogously. The total bid and ask interest in the orderbook at time t are given by $\hat{V}_t^{(b)} = \int_0^{X_t} D_b(x,t)\, dx$ and $\hat{V}_t^{(a)} = \int_{X_t}^{\infty} D_a(x,t)\, dx$ respectively; we then have $\Delta \hat{V}_t^{(b)} = \hat{V}_t^{(b)} - \hat{V}_{t-1}^{(b)}$ and $\Delta \hat{V}_t^{(a)} = \hat{V}_t^{(a)} - \hat{V}_{t-1}^{(a)}$. The neural network is a four-layer feed-forward model. The two hidden layers have 20 and 10 neurons respectively, for a total of 383 evolvable parameters in each neural network [3].

It is an analytical convenience to consider a single generation of the evolutionary process described in Sec. 2.2.4 as a draw from a stochastic function $\mathcal{G}(\alpha, \mathcal{M})$, where α describes the specification of the agents in the model and $\mathcal{M}$ is the evolutionary mechanism (selection and mutation); we will describe these parameters in some detail in Sec. 2.2.4. This function yields orderbooks and price time series; each call to the function yields a tuple of bid and ask order density and a price time series,

$$(D_b(x,t), D_a(x,t), X_t) = \mathcal{G}(\alpha, \mathcal{M}). \tag{2.14}$$

This way of looking at the process makes it easy to express Monte Carlo estimates of theoretical quantities and provides the theoretical basis for conversion of evolved agents into trading algorithms as we outline in Sec. 2.2.5. We can re-express the above

[3] Number of parameters is equal to number of parameters in the weight matrices plus the number of parameters in the bias vectors $= 350 + 33$.

integrals as Monte Carlo estimates (which is how we compute them in practice) so that $\hat{V}_t^{(b)} \approx \sum_{\text{observed bids } x'} D_b(x',t)$ and $\hat{V}_t^{(a)} \approx \sum_{\text{observed asks } x'} D_a(x',t)$, where $D_b(x,t)$, $D_a(x,t)$, and X_t are given by Eq. 2.14. As a visual reference point, in Panel B of Fig. 2.10 we display an example realization of Monte Carlo-approximated $\Delta\hat{V}^{(b)}$ and $\Delta\hat{V}^{(a)}$, while in panel C of this figure we display the corresponding ΔX.

2.2.4 Evolutionary dynamics

We provide a summary of the evolutionary dynamics in Fig. 2.9. We first describe the selection and mutation mechanism $\mathcal{M}$, since this mechanism changes on the objective of the simulation, and then describe the simulation in general. We set the evolutionary mechanism $\mathcal{M}$ to be either tournament selection-based or the identity (no evolution): we use the tournament selection mechanism when we are attempting to evolve agents of high fitness, while we use the identity mechanism when we are generating empirical market statistics for use with real data, as described in Sec. 2.2.5.

The tournament selection mechanism is standard [39], designed as follows: given a population of evolvable agents, at the end of each generation a tournament of τ agents is selected from the population at random. We set $\tau = 17$. These agents are sorted by fitness—their total profit π in the market simulation of the past generation—so that $\pi_{(1)} \geq \pi_{(2)} \geq \cdots \geq \pi_{(\tau)}$. Agent (i) is selected to remain with probability $p(1-p)^{i-1}$, where we set $p = \frac{1}{2}$; the remaining agents are discarded. A total of $\tau - 1$ new agents are initialized with the parameters from the selected agent (i), denoted by $\theta_{(i)} = (\theta_{(i),1}, ..., \theta_{(i),L})$ where the agent has a total of L parameters. These new parameters are then subjected to centered Gaussian mutation; the parameters of new agent i' are given by $\theta_{i'} = \theta_{(i)} + z_{i'}$, where $z_{i'} \sim \mathcal{N}(0, \gamma^2\Sigma)$ and we set $\gamma = 0.1$. The

covariance matrix Σ of the Gaussian is diagonal, with $\Sigma_{\ell\ell} = \text{Var}(\theta_{(i),\ell})$. The $\tau - 1$ agents are added back into the entire population of evolvable agents for use in further generations, described in the next paragraph.

At each of $g = 1, ..., G$ generations, we initialize $k = 1, ..., K$ independent markets, which we set to $K = 24$. We set $G = 100$. In each market, we initialize N_A agents with agent parameter vector distributed as $(\alpha_{a,k})_{a \in A} = \alpha_k \sim p(\alpha)$, where A is the set of agent types outlined in Sec. 2.2.3. Given a drawn α_k, there are $\alpha_{\text{zi},k}$ zero intelligence agents, $\alpha_{\text{mo},k}$ momentum agents, and so on. The probability distribution $p(\alpha)$ is a joint distribution over agent types that factors as a uniform distribution over the number of neural network agents and a multinomial distribution over the number of other agents that is dependent on the number of drawn neural network agents. Given a number of neural network agents $\alpha_{\text{nn},k}$ drawn uniformly at random between nn_{min} and nn_{max}, the remaining $N_A - \alpha_{\text{nn},k}$ are drawn from a multinomial distribution with probabilities $\rho_a = \frac{1}{6}$. We set $\text{nn}_{\text{min}} = 2$ and $\text{nn}_{\text{max}} = 10$. Within each market, agents interact *vis-à-vis* the matching engine described in Sec. 2.2.2, trading for a total of T timesteps within each generation. We set $T = 500$ and set the number of trading timesteps per day equal to 100, as outlined in the previous section. After T timesteps, the population of neural networks is pooled—removed from each simulation and collated into one set—and the evolutionary mechanism $\mathcal{M}$ is applied to all $\sum_{k=1}^{K} \alpha_{\text{nn},k}$ neural networks, generating a (partially) new population, as outlined in the previous paragraphs of this section. The new population of neural networks is then shuffled and divided into K new independent markets according to the $\alpha_{\text{nn},k}$, along with new static agents again drawn from the multinomial distribution with equal probabilities, and the process begins again in generation $g + 1$.

2.2.5 From evolved agent to trading algorithm

We convert evolved neural networks into executable trading strategies that we subsequently backtest on real financial data. The major impediment to simply using the evolved networks as trading strategies is the lack of readily-available orderbook information for real financial markets: though such information is available for sale, it is prohibitively expensive to purchase [13]. Instead, we use orderbook data generated by $\mathcal{G}(\alpha, \mathcal{M})$ as a surrogate for real orderbook data, which we believe to be an important input into the algorithms as orderbook data has been shown to carry non-zero information about future prices and be useful in making profitable short-term trading decisions [81, 82, 83]. We reason that, if $\mathcal{G}(\alpha, \mathcal{M})$ is a good simulacrum of the true orderbook-generating process active in a real financial market, then the values of $\Delta\hat{V}^{(b)}$ and $\Delta\hat{V}^{(a)}$ generated by the agent-based model, given an observed value of ΔX from a real market, should be similar to the changes in resting bid and ask volume that existed in the real market, even though we do not have access to that data.

Because of our lack of orderbook data, we must simulate $\Delta\hat{V}^{(b)}$ and $\Delta\hat{V}^{(a)}$ that correspond with the observed change in price ΔX. To do this, we first simply draw many values from $\mathcal{G}(\alpha, \mathrm{id})$ (the generative model with no evolutionary mechanism). Then, given ΔX from the asset market, we draw multiple $(\Delta\hat{V}^{(b)}, \Delta\hat{V}^{(a)})$ pairs from their empirical joint pdf conditioned on this observation, which is generated by the draws from $\mathcal{G}(\alpha, \mathrm{id})$:

$$(\Delta\hat{V}^{(b)}, \Delta\hat{V}^{(a)}) \sim \hat{p}_{\mathcal{G}(\alpha,\mathrm{id})}(\Delta V^{(b)}, \Delta V^{(a)} | \Delta X), \tag{2.15}$$

and evaluate f using the conditional expectation of these values; this new "marginalized" algorithm is given by

$$\begin{aligned} &f_{\text{marg}}(\Delta X_t, \Delta c_t, \Delta N_t, \Delta \pi_t) \\ &\quad = f(\Delta X, E[\Delta \hat{V}^{(b)} | \Delta X], E[\Delta \hat{V}^{(a)} | \Delta X], \Delta c, \Delta N, \Delta \pi), \end{aligned} \tag{2.16}$$

where the expectations are taken under the pdf displayed in Eq. 2.15. As with the non-marginalized algorithm f, f_{marg} returns a side, change in shares, and change in price:
$(s_t, N_t^{(o)}, \Delta X_t^{(o)}) = f_{\text{marg}}(\Delta X_t, \Delta c_t, \Delta N_t, \Delta \pi_t)$. Since we are backtesting the trading algorithm and hence it is impossible to perform price discovery, we do not use $\Delta X_t^{(o)}$. We simply simulate the execution of a market buy ($s_t = 1$) or sell ($s_t = -1$) order for $N_t^{(o)}$ spot contracts of the asset and then update the algorithm's internal state. As in the evolutionary process within the agent-based model, we allow the algorithm to sell short so that N_t may be negative.

We implement three basic risk management routines that supervise the execution of the algorithm. These routines act as "circuitbreakers" to halt the algorithm's operation if certain risk limits are reached and consist of two versions of the traders' adage "cut your losses but let your winners run," ensuring that maximum loss is capped at some user-set limit, and one leverage limit routine that halts execution if the algorithm is long or short a certain large number of contracts (we set this number equal to 150) [84]. (The interested reader is referred to the supplementary information for more detail.) Though these routines pale in comparison with real risk-management software used in algorithmic trading [85, 86, 87], we believe that they are sufficient for the purposes of this work.

2.2.6 Results

Evolutionary dynamics

We ran 100 independent simulations of the entire process outlined in Sec. 2.2.4. This resulted in a total of 24,000 (= 24 independent markets per generation $\times$ 100 generations $\times$ 100 independent simulations) conditionally independent market simulations from which to sample evolved neural networks for validation and testing on real financial asset data. At each generation of each simulation, we saved the best individual for possible further use.

Evolved neural networks (nn) quickly became the dominant species of trading agent, though they were not profitable at $g = 0$ (at which point they were simply random neural networks with normally-distributed weights and biases). Fig. 2.11 displays average (solid curves) and median (dashed curves) wealth trajectories for each agent type; these statistics are calculated over all simulations at that generation. The average and median wealth of nn agents increases quickly until about $g = 10$. It then increases more slowly until about $g = 40$, when it plateaus. Concomitant with the rise in average and median nn wealth is a decline in the wealth of nearly every other agent type. In particular, though fundamental value (fv) agents started out as the most profitable agent type (due to their strong beliefs about "true" asset values and market power, in early generations they had the ability to collectively set market price), they became the second-worst performing agent type as nn agents became more profitable. It is notable that momentum trading agents are the only static agent type that was still able to make positive profits during multiple sequential later generations (in particular $g > 30$), long after all other static agents had become, on

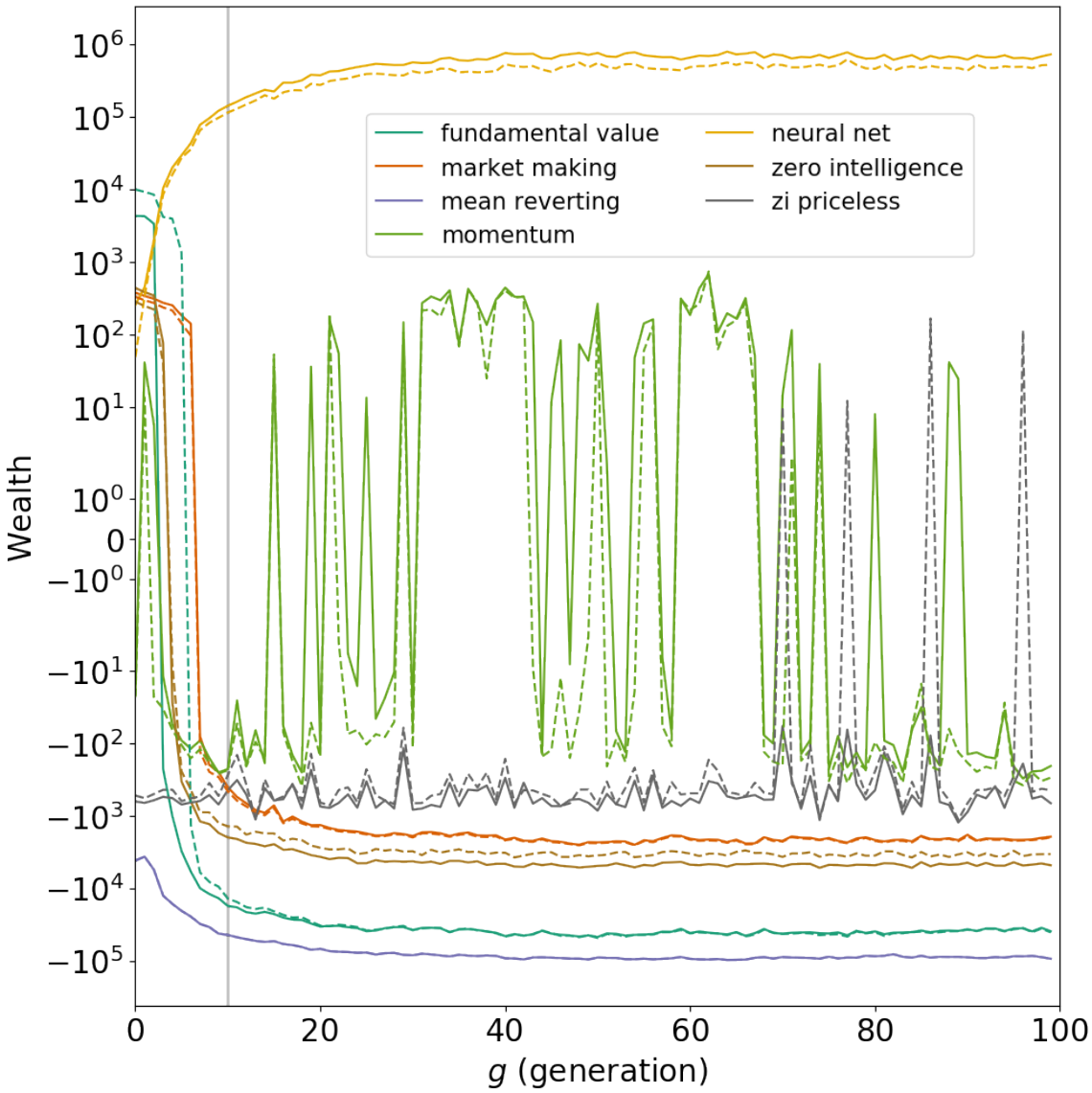

Figure 2.11: Evolving neural network agents quickly dominate all static agents; average wealth of neural network agents increases until about $g = 40$, where it plateaus while the average wealth of other static trading agents remains largely flat or decreases over time. In particular, mean-reverting and fundamental-value traders suffer large average wealth losses, even though fundamental-value traders start as the most profitable agent type. We also find that a static momentum trading strategy is, on average, the strategy that is least dominated by evolving neural networks and can actually be sustainably profitable for multiple generations; this result corresponds with the finding that a momentum-based strategy can be profitable in real financial markets [1].

average, very unprofitable. This is consistent with the finding that real asset prices may exhibit momentum effects and that trading strategies based on exploiting this momentum may result in positive expected profit [17].

As evolution progressed through generations, statistical properties of asset price time series X_t changed significantly. The mean square deviation (MSD) of X_t in generation g, defined by the exponent γ_g in the relationship $E_g[(X_t^{(g)} - \mu_t)^2] \propto t^{\gamma_g}$ where μ_t is the intertemporal mean of $X_t^{(g)}$, begins in the sub- or normally-diffusive region ($\gamma_g \leq 1$) but quickly rises to $\gamma_g \approx 1.8$ near $g = 10$ and remains there for the remainder of evolutionary time. We display the MSD of asset prices by generation in Fig. 2.12. MSD that grows superlinearly with time is termed anomalous superdiffusion [88, 89] and is commonly observed in real asset markets [90, 91, 92]. This may provide evidence that observed asset price superdiffusion in real asset markets is partially driven by purely endogenous evolutionary dynamics.

Validation and testing of evolved strategies

We chose a subsample of the neural networks that we extracted from the market simulations for consideration as algorithmic trading strategies to be used on real data. Though all evolved neural networks that we saved had high fitness in the context of the agent-based model, we hypothesized that it would not be the case that all of them would have high fitness when backtested on observed asset price data. We selected the high-fitness neural networks saved at generation $g = 10$, the set of which we will denote by $\mathcal{A}_{10}$, for two reasons. First, this was the approximate "elbow" of $\log_{10} \pi$, as displayed in Fig. 2.11; at generations later than approximately $g = 10$, nn agents exhibited decreasing marginal $\log_{10} \pi$. Second, this generation was the point at which

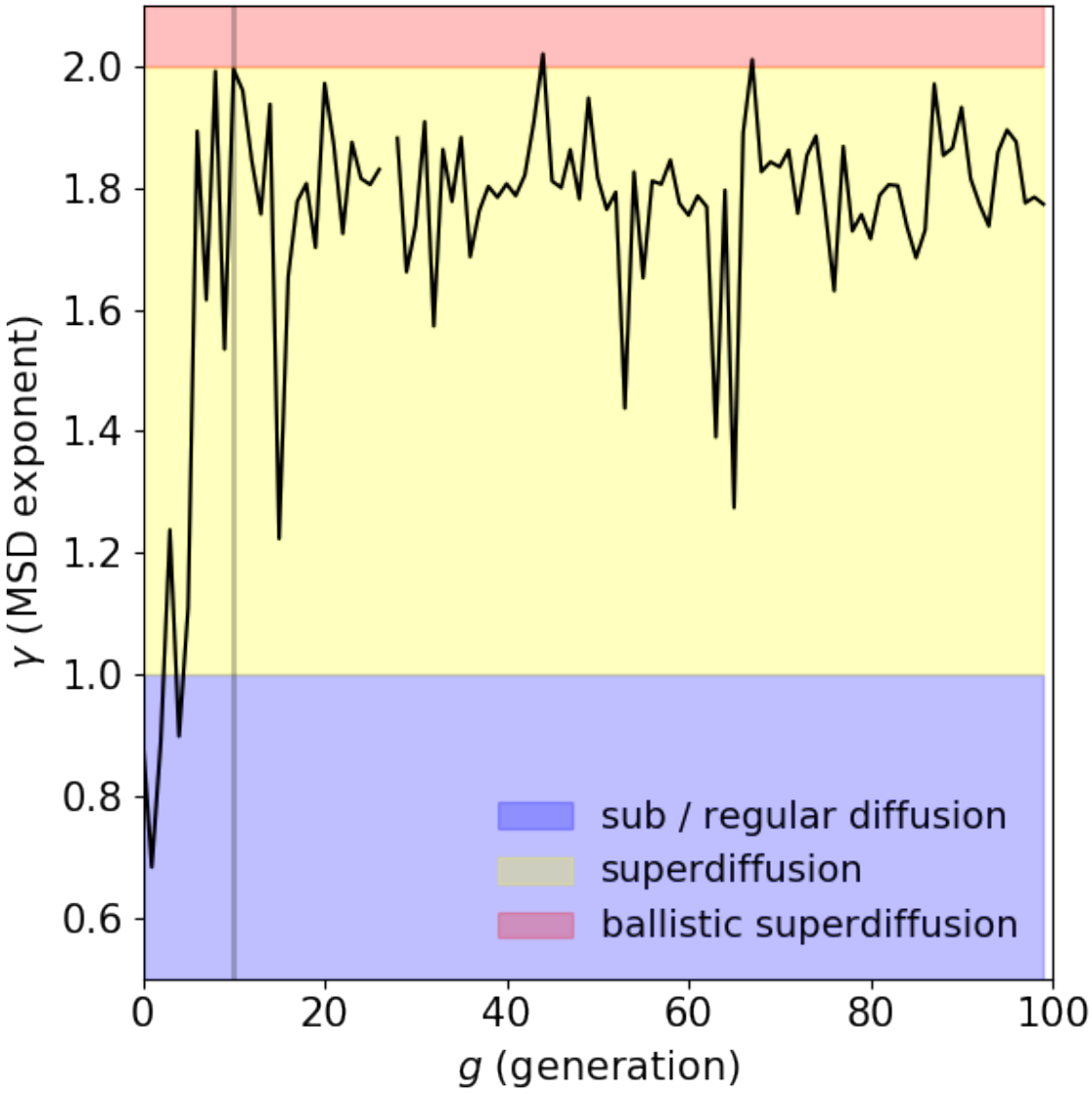

Figure 2.12: Asset price superdiffusion emerges as a byproduct of evolutionary pressure. Superdiffusion is defined by a superlinear relationship between mean squared deviation of a time series and time itself. At each generation g we fit a model of the form $E[(X_t^{(g)} - \mu)^2] \propto t^{\gamma_g}$ and plot the resulting γ_g as a function of generation g. This exponent of dispersion stabilizes at roughly $\gamma_g \simeq 1.8$ after approximately 10 generations of evolution (indicated by the vertical black line at $g = 10$), which influences our selection of $g = 10$ for validation and testing of evolved strategies on real data.

the exponent of the MSD of asset prices appeared to stabilize at $\gamma_g \approx 1.8$.

The asset prices produced by the interaction of agents in the ABM do not appear to exhibit geometric (multiplicative noise) dynamics, but rather arithmetic (additive noise) dynamics. Though many real financial assets do exhibit geometric- or geometric-like dynamics [93], other assets, such as foreign exchange (FX) spot contracts, typically display quasi-arithmetic dynamics instead [94]. We thus test our evolved neural networks on FX spot rate data, eight and a half (January 2015 through July 2019) years of millisecond-sampled EUR/USD and GBP/USD spot exchange rate data sourced from an over-the-counter trading venue [4]. We do not implement transaction costs in our backtesting as, if these are constant, their marginal incidence on profit decreases as the amount of leverage (net number of contracts traded) increases. We separate this dataset into a validation (2010 - 2015) and testing (2015 - 2019) split. Although this split is unnecessary in the context of overfitting to data (because the neural networks are just static at this point; their evolution occurred in the ABM and they do not update their weights based on the observed FX data), it is part of a method by which we lower the probability of false positive discovery of high-performance trading algorithms. The top k algorithms in $\mathcal{A}_{10}$, ranked by total profit accumulated by trading on validation data, were then used to trade on the test dataset; if these algorithms accumulated high profit on the validation dataset by chance, it is likely that they would not perform well on the test dataset. We set $k = 5$ and will denote these "elite" trading strategies by $\mathcal{A}_{\text{elite}}$ in what follows.

To create trading algorithms from the evolved neural networks, we followed the procedure described in Sec. 2.2.5. We resampled the spot FX rate time series at the

[4]The data is available from https://www.truefx.com/, which sources it from the Integral OCX ECN.

10s resolution, setting as X_t the mean price during that $10s$ interval multiplicatively rescaled by a constant $(100/X_0)$ so that the price on the first second of each month was equal to 100. Agents traded during the first 10^6 seconds (approximately 16.2 days[5]) of each month, a number of timesteps that we chose arbitrarily to avoid possible edge effects occurring at the end of the month. We will refer to one 10^6s time interval of trading on a single spot rate (EUR/USD or GBP/USD) as a single trading episode. In Fig. 2.13, we display an example time series of spot FX rate (EUR/USD during July of 2016) and corresponding profit made by an evolved neural network in this trading episode. In this example, though the spot rate fluctuates considerably and has $|X_T - X_0| < 0.01$ USD/contract, the profit time series increases fairly steadily throughout the entire time period, netting a total of $\pi_T \approx 0.125$ USD.

Evolved strategies differed significantly from random neural network strategies. We compared the distribution of profit on the validation dataset by $\mathcal{A}_{10}$, profit on the test dataset by $\mathcal{A}_{\text{elite}}$, and profit on the test dataset by 20 random neural network agents ($\mathcal{A}_{\text{random}}$) and display empirical cdfs of these distributions, along with bootstrapped pdfs of the means of these distributions, in Fig. 2.14. Elite individuals (the top $k = 5$ performers on the validation dataset) have the greatest maximum absolute difference (Kolmogorov-Smirnov statistic) between the empirical cdf of their profit and the other cdfs ($D(\mathcal{A}_{\text{elite}}, \mathcal{A}_{\text{random}}) = 0.4488$, $D(\mathcal{A}_{\text{elite}}, \mathcal{A}_{10}) = 0.3809$) while the maximum distance between the empirical cdf of random neural network profit and all evolved neural network profit was smaller, but still significantly greater than zero ($D(\mathcal{A}_{10}, \mathcal{A}_{\text{random}}) = 0.0956$), as demonstrated in panel A. Though the $\mathcal{A}_{10}$ and $\mathcal{A}_{\text{random}}$ profit distributions are far more similar to each other than they are to that

[5] 16.2 days $\approx \frac{10^6\text{s}}{60\text{s/m} \times 60\text{s/h} \times 24\text{h/trading day} \times \frac{5\text{trading days}}{7\text{days}}}$

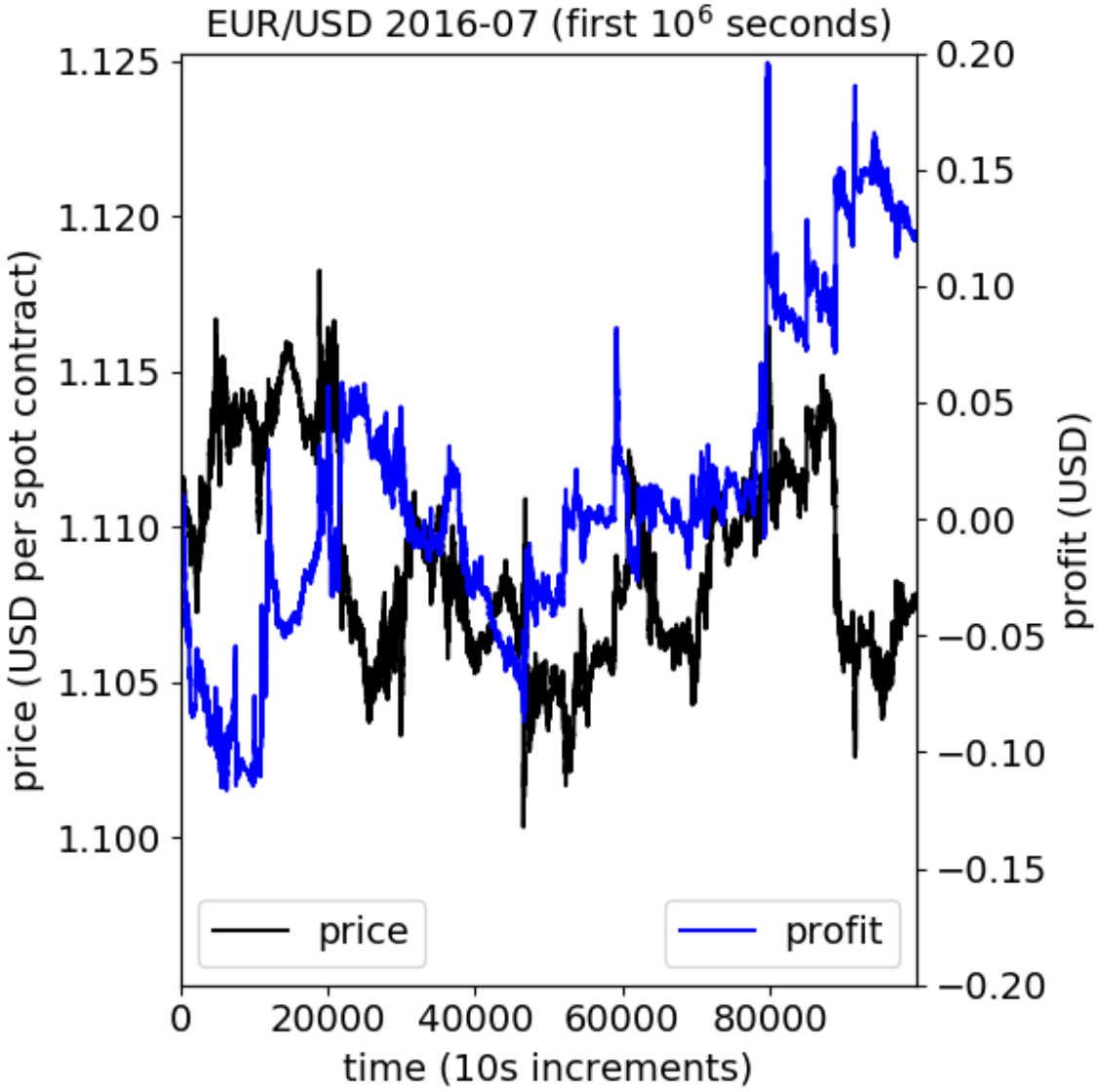

Figure 2.13: Elite evolved trading algorithms are able to obtain positive profit under a wide variety of backtested trading conditions. We show the spot price of EUR/USD for the first 10^6 seconds of July 2016 in the black curve; this time series displays both large increases and decreases during this time period, as well as regions of relatively low and high volatility. Despite these varied conditions, an elite evolved algorithm was able to capture positive profit (shown in the blue curve) over this time period, showing large gains in profit during both price drawdowns and ramp-ups. We note that this particular observation is significantly below the mean total profit generated by elite evolved algorithms, as demonstrated in Fig. 2.14.

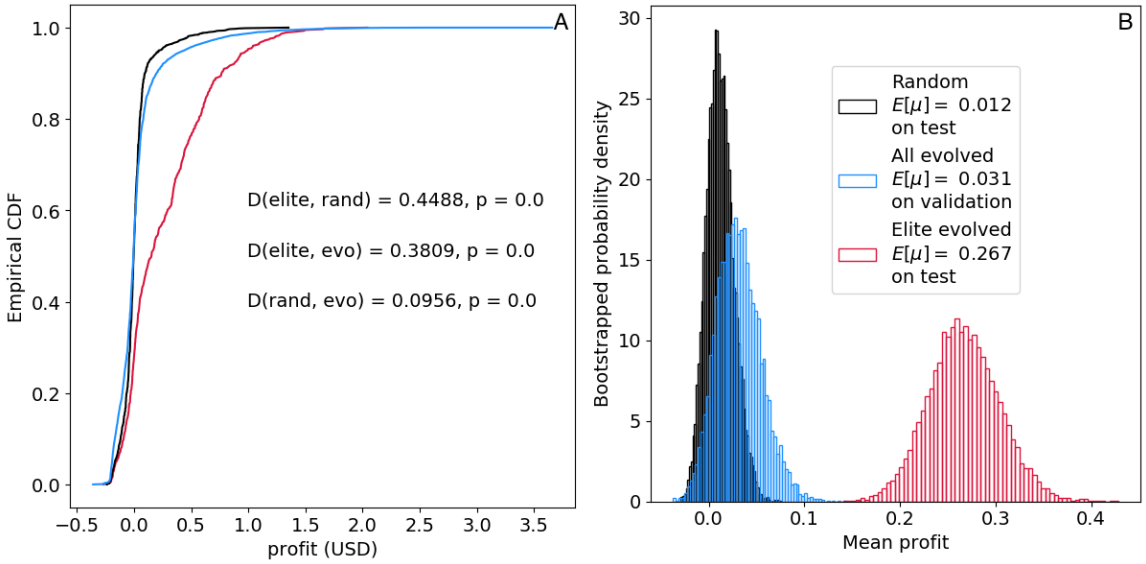

Figure 2.14: The profit distributions of all evolved neural networks, random neural networks, and elite individuals when evaluated on real FX spot rate data differ significantly, as we demonstrate in panel A. In panel B, we demonstrate that elite evolved neural network trading strategies have significantly higher mean profits on test data than do random neural network strategies or the set of all evolved strategies evaluated on validation data. (The separation between validation and test data is irrelevant for the set of all evolved neural networks, as these networks evolved in the agent-based model, not through evaluation on real data.) The data do not appear to have a diverging second moment; we do not concern ourselves with issues that arise with bootstrapping in distributions with tail exponent $\alpha < 2$ [2].

of $\mathcal{A}_{\text{elite}}$, they differ significantly in their tails: the $\mathcal{A}_{10}$ distribution has a higher likelihood of observing larger losses (though the magnitude of these losses are still severely damped by the risk management routines detailed in Sec. 2.2.5) and larger gains than does the $\mathcal{A}_{\text{random}}$ distribution. It is possible this occurred because many agents in $\mathcal{A}_{10}$ had high fitness in their particular realization of the ABM with agent concentration vector α, but α did not resemble the (effectively unobservable) makeup of agents that generated the real spot price time series. The mean profit of $\mathcal{A}_{10}$ on validation data was higher than that of $\mathcal{A}_{\text{random}}$ on test data, though estimates of this mean (displayed in panel B of Fig. 2.14) have greater dispersion that estimates of the mean of $\mathcal{A}_{10}$ profit. It is likely (probability ≈ 0.7617) that the true mean of the $\mathcal{A}_{10}$ profit distribution is greater than the mean of the $\mathcal{A}_{\text{random}}$ profit distribution. However, distributions of estimated mean for both sets of agents do contain zero (though the estimated probabilities that the mean is greater than zero are 0.7863 for $\mathcal{A}_{\text{random}}$ and 0.9241 for $\mathcal{A}_{10}$).

Crucially, mean $\mathcal{A}_{\text{elite}}$ profit on test data is very far from zero (approximately 0.267 USD per trading episode) and the estimated distribution of mean profit is bounded away from zero. We discuss these results further in the supplementary information.

2.2.7 Discussion and conclusion

We construct a method to develop *ab initio* trading algorithms using an agent-based model of a financial market. We subject expressive agents to evolutionary pressure, using profit generated in the agent-based model as the fitness function. We then backtest high-performing agents backtest on real financial asset price data (spot foreign exchange rates). We further tested "elite" evolved agents—agents that performed well

on validation data (EUR/USD and GBP/USD from 2010 to 2015)—on test data, the same currency pairs but during the time interval 2016 through 2019. We find that it is possible for evolved trading algorithms to make significantly positive backtesting profit for extended periods of time during varying market conditions, even though these evolved algorithms had never experienced real financial data during the process of evolution. This result provides evidence that a paradigm shift in the design of automated trading algorithms— from prediction of future market states to, instead, closely modeling the underlying mechanisms and agents of which the market is composed—may be both feasible and profitable.

Though this result is promising, it is important to note some shortcomings of our research and avenues for further exploration. First, and most importantly, we have not actually tested the performance of the "elite" evolved agents in a real market, but only backtested them on real market data. The difference between these actions is profound; the ultimate expression of confidence in a trading strategy is to use it with one's own capital and we have not yet done this [95]. Though the elite agents are consistently profitable when backtested, this does not guarantee that they will be profitable when used to trade "live" in a real financial market. If elite agents generated according to the methodology laid out in Secs. 2.2.4 and 2.2.5 are profitable when used to trade spot contracts in foreign exchange markets, we will be more confident in stating that this methodology is a robust method of generating *ab initio* trading algorithms.

Second, our implementations of multiple components of our methodology were intended as proofs-of-concept; four-layer feed-forward neural networks are decidedly not at the cutting-edge of neural network design [96]. Future work could focus on

improving the expressiveness and realism of agents used in the agent-based model, modifying the evolvable neural networks to use a recurrent architecture, using different order types in the matching engine (and hence increasing dimensionality of the neural networks' action space), and in general attempting to more closely match the composition of the agent-based model with the structure of modern-day securities markets (in particular, spot FX rate markets).

Finally, there is substantial room for more analysis of the free parameters in our agent-based model—for example, tuning the parameters of the tournament selection adaptively so that later generations do not evolve to simply exploit the structure of the agent-based model but rather continue to explore novel trading strategies. More generally, we should improve the design of the mechanism by which we select high-performing individuals from the model to be backtested on real data. We have used only a heuristic measure—the apparent stabilization of the MSD exponent and decreasing marginal log profit—as indicators as from which generation we should select, and what follows this is essentially just rejection sampling from the space of agents that are high-performers in the agent-based model through evaluation of these agents on validation data. We believe that there are probably better ways to implement this step.

Chapter 3

Shape-based time series similarity search

3.1 Introduction

The tasks of peak detection, similarity search, and anomaly detection in time series is often accomplished using the discrete wavelet transform (DWT) [97] or matrix-based methods [98, 99]. For example, wavelet-based methods have been used for outlier detection in financial time series [100], similarity search and compression of various correlated time series [101], signal detection in meteorological data [102], and homogeneity of variance testing in time series with long memory [103]. Wavelet transforms have far superior localization in the time domain than do pure frequency-space methods such as the short-time Fourier transform [104]. Similarly, the chirplet transform is used in the analysis of phenomena displaying periodicity-in-perspective (linearly- or quadratically-varying frequency), such as images and radar signals [105, 106, 107, 108]. Thus, when analyzing time series that are partially composed of exogenous shocks and endogenous shock-like local dynamics, we should use a small sample of such a function—a "shock", examples of which are depicted in Fig. 3.1, and functions generated by concatenation of these building blocks, such as that shown in Fig. 3.2. In this work, we introduce the Discrete Shocklet Transform (DST), generated by cross-correlation functions of a shocklet. As an immediate example (and before any definitions or technical discussion), we contrast the DWT with the DST of a sociotechnical time series—popularity of the word "trump" on the social media website Twitter—in Fig. 3.3, which is a visual display of what we claim is the DST's suitability for detection of local mechanism-driven dynamics in time series.

We will show that the DST can be used to extract shock and shock-like dynamics of particular interest from time series through construction of an indicator function

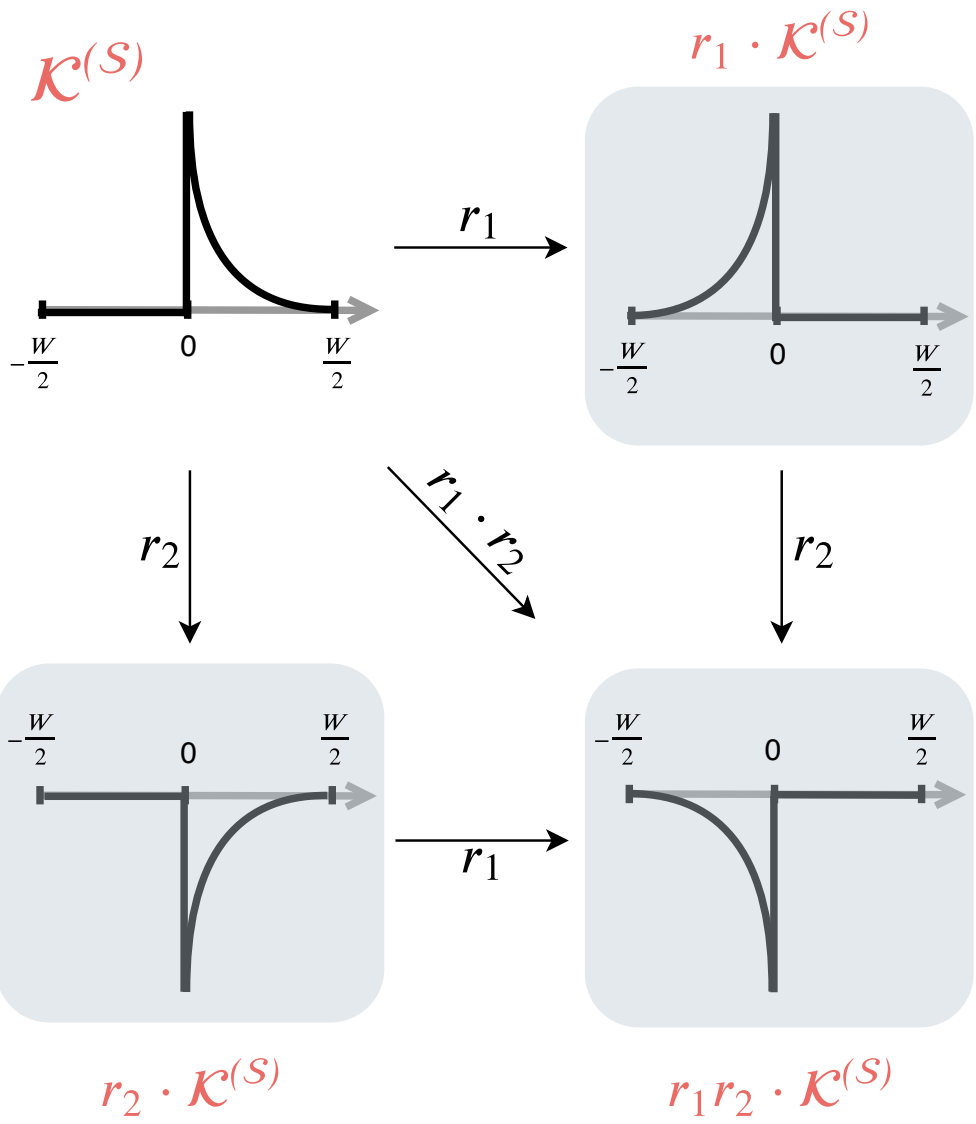

Figure 3.1: The discrete shocklet transform is generated through cross-correlation of pieces of shocks; this figure displays effects of the action of group elements $r_i \in R_4$ on a base "shock-like" kernel $\mathcal{K}$. The kernel $\mathcal{K}$ captures the dynamics of a constant lower level of intensity before an abrupt increase to a relatively high intensity which decays over a duration of $W/2$ units of time. By applying elements of R_4, we can effect a time reversal (r_1) and abrupt cessation of intensity followed by asymptotic convergence to the prior level of intensity (r_2), as well as the combination of these effects ($r_3 = r_1 \cdot r_2$). In Section 3.3.3 we illuminate a typology of shock dynamics derived from combinations of these basic shapes.

that compresses time-scale-dependent information into a single spatial dimension using prior information on timescale and parameter importance. Using this indicator, we are able to highlight windows in which underlying mechanistic dynamics are hypothesized to contribute a stronger component of the signal than purely stochastic dynamics, and demonstrate an algorithm—the Shocklet Transform And Ranking (STAR) algorithm—by which we are able to automate *post facto* detection of endogenous, mechanism-driven dynamics. As a complement to techniques of changepoint analysis, methods by which one can detect changes in the level of time series [109, 110], the DST and STAR algorithm detect changes in the underlying mechanistic local dynamics of the time series. Finally, we demonstrate a potential usage of the shocklet transform by applying it to the LabMT Twitter dataset [111] to extract word usage time-series matching the qualitative form of a shock-like kernel at multiple timescales.

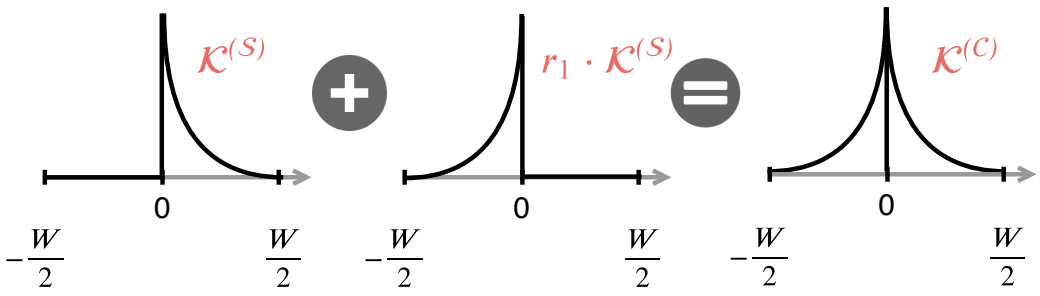

Figure 3.2: This figure provides a schematic for the construction of more complicated shock dynamics from a simple initial shape ($\mathcal{K}^{(S)}$). By acting on a kernel with elements r_i of the reflection group R_4 and function concatenation, we create shock-like dynamics, as exemplified by the symmetric shocklet kernel $\mathcal{K}^{(C)} = \mathcal{K}^{(S)} \oplus [r_1 \cdot \mathcal{K}^{(S)}]$ in this figure.

3.2 Data and Theory

3.2.1 Data

Twitter is a popular micro-blogging service that allows users to share thoughts and news with a global community via short messages (up to 140 or, from around November 2017 on, 280 characters, in length). We purchased access to Twitter's "decahose" streaming API and used it to collect a random 10% sample of all public tweets authored between September 9, 2008 and April 4, 2018 [112]. We then parsed these tweets to count appearances of words included in the LabMT dataset, a set of roughly 10,000 of the most commonly used words in English [111]. The dataset has been used to construct nonparametric sentiment analysis models [113] and forecast mental illness [114] among other applications [115, 116, 117]. From these counts, we analyze the time series of word popularity as measured by rank of word usage: on day t, the most-used word is assigned rank 1, the second-most assigned rank 2, and so on to create time series of word rank r_t for each word.

3.2.2 Theory

Algorithmic details: description of the method

There are multiple fundamentally-deterministic mechanistic models for local dynamics of sociotechnical time series. Nonstationary local dynamics are generally well-described by exponential, bi-exponential, or power-law decay functions; mechanistic models thus usually generate one of these few functional forms. For example, Wu

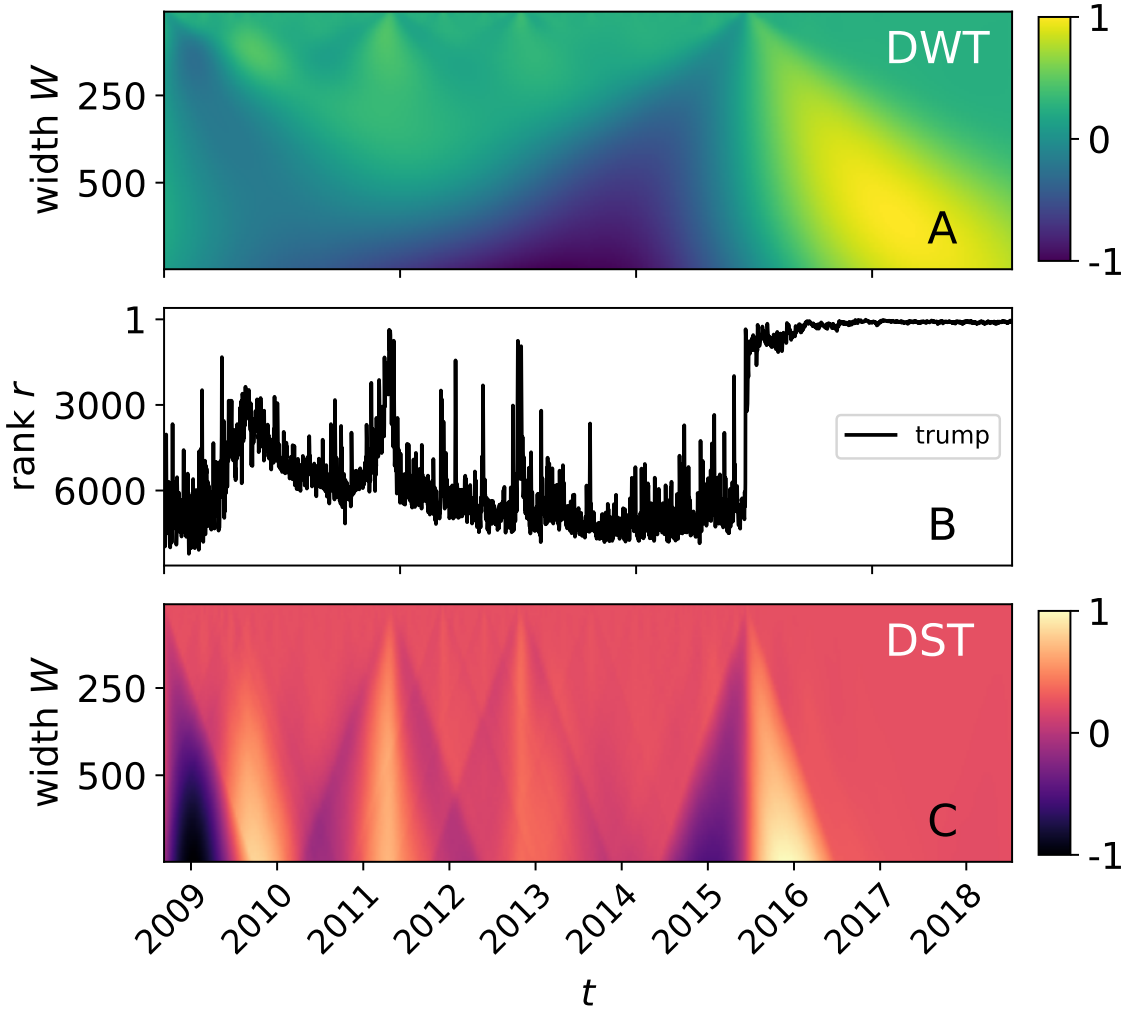

Figure 3.3: A comparison between the standard discrete wavelet transform (DWT) and our discrete shocklet transform (DST) of a sociotechnical time series. Panel B displays the daily time series of the rank r_t of the word "trump" on Twitter. As a comparison with the DST, we computed the DWT of r_t using the Ricker wavelet and display it in panel A. Panel C shows the DST of the time series using a symmetric power shock, $\mathcal{K}^{(S)}(\tau|W,\theta) \sim \mathrm{rect}(\tau)\tau^{\theta}$, with exponent $\theta = 3$. We chose to compare the DST with the DWT because the DWT is similar in mathematical construction (see Appendix D.2 for a more extensive discussion of this assertion), but differs in the choice of convolution kernel (a wavelet, in the case of the DWT, and a piece of a shock, in the case of the DST) and the method by which the transform accounts for signal at multiple timescales.

and Huberman described a stretched-exponential model for collective human attention [118], and Candia *et al.* derived a biexponential function for collective human memory on longer timescales [119]. Crane and Sornette assembled a Hawkes process for video views that produces power-law behavior by using power-law excitement kernels [120], and Lorenz-Spreen *et al.* demonstrated a speeding-up dynamic in collective social attention mechanisms [121], while De Domenico and Altmann put forward a stochastic model incorporating social heterogeneity and influence [122], and Ierly and Kostinsky introduced a rank-based, signal-extraction method with applications to meteorology data [123]. In Sec. 3.2.2 we conduct a literature review, contrasting our methods with existing anomaly detection and similarity search time series data mining algorithms and demonstrating that the DST and associated STAR algorithm differ substantially from these existing algorithms. We have open-sourced implementations of the DST and STAR algorithm; code for these implementations is available at a publicly-accessible repository [1].

We do not assume any specific model in our work. Instead, by default we define a kernel $\mathcal{K}^{(\cdot)}$ as one of a few basic functional forms: exponential growth,

$$\mathcal{K}^{(S)}(\tau|W,\theta) \sim \text{rect}(\tau - \tau_0)e^{\theta(\tau-\tau_0)}; \tag{3.1}$$

monomial growth,

$$\mathcal{K}^{(S)}(\tau|W,\theta) \sim \text{rect}(\tau - \tau_0)\tau^{\theta}; \tag{3.2}$$

power-law decay,

$$\mathcal{K}^{(S)}(\tau|W,\theta) \sim \text{rect}(\tau - \tau_0)|\tau - \tau_0 + \varepsilon|^{-\theta}, \tag{3.3}$$

[1] Python implementations of the DST and STAR algorithms are located at this git repository: https://gitlab.com/compstorylab/discrete-shocklet-transform

or sudden level change (corresponding with a changepoint detection problem),

$$\mathcal{K}^{(Sp)}(\tau|W,\theta) \sim \text{rect}(\tau - \tau_0)[\Theta(\tau) - \Theta(-\tau)], \tag{3.4}$$

where $\Theta(\cdot)$ is the Heaviside step function. The function rect is the rectangular function ($\text{rect}(x) = 1$ for $0 < x < W/2$ and $\text{rect}(x) = 0$ otherwise), while in the case of the power-law kernel we add a constant ε to ensure nonsingularity. The parameter W controls the support of $\mathcal{K}^{(\cdot)}(\tau|W,\theta)$; the kernel is identically zero outside of the interval $[\tau - W/2, \tau + W/2]$. We define the window parameter W as follows: moving from a window size of W to a window size of $W + \Delta W$ is equivalent to upsampling the kernel signal by the factor $W + \Delta W$, applying an ideal lowpass filter, and downsampling by the factor W. In other words, if the kernel function $\mathcal{K}^{(\cdot)}$ is defined for each of W linearly spaced points between $-N/2$ and $N/2$, moving to a window size of W to $W + \Delta W$ is equivalent to computing $\mathcal{K}^{(\cdot)}$ for each of $W + \Delta W$ linearly-spaced points between $-N/2$ and $N/2$. This holds the dynamic range of the kernel constant while accounting for the dynamics described by the kernel at all timescales of interest. We enforce the condition that $\sum_{t=-\infty}^{\infty} \mathcal{K}^{(\cdot)}(t|W,\theta) = 0$ for any window size W.

It is decidedly not our intent to delve into the question of how and why deterministic underlying dynamics in sociotechnical systems arise. However, we will provide a brief justification for the functional forms of the kernels presented in the last paragraph as scaling solutions to a variety of parsimonious models of local deterministic dynamics:

- If the time series $x(t)$ exhibits exponential growth with a state-dependent growth

damper $D(x)$, the dynamics can be described by

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \frac{\lambda}{D(x(t))}x(t),\ x(0) = x_0. \tag{3.5}$$

If $D(x) = x^{1/n}$, the solution to this IVP scales as $x(t) \sim t^n$, which is the functional form given in Eq. 3.2. When $D(x) \propto 1$ (i.e., there is no damper on growth) then the solution is an exponential function, the functional form of Eq. 3.1.

- If instead the underlying dynamics correspond to exponential decay with a time- and state-dependent half-life $\mathcal{T}$, we can model the dynamics by the system

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = -\frac{x(t)}{\mathcal{T}(t)},\ x(0) = x_0 \tag{3.6}$$

$$\frac{\mathrm{d}\mathcal{T}(t)}{\mathrm{d}t} = f(\mathcal{T}(t), x(t)),\ \mathcal{T}(0) = \mathcal{T}_0. \tag{3.7}$$

If f is particularly simple and given by $f(\mathcal{T}, x) = c$ with $c > 0$, then the solution to Eq. 3.6 scales as $x(t) \sim t^{-1/c}$, the functional form of Eq. 3.3. The limit $c \to 0^+$ is singular and results in dynamics of exponential decay, given by reversing time in Eq. 3.1 (about which we expound later in this section).

- As another example, the dynamics could be essentially static except when a latent variable φ changes state or moves past a threshold of some sort:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \delta\left(\varphi(t) - \varphi^*\right),\ x(0) = x_0 \tag{3.8}$$

$$\frac{\mathrm{d}\varphi(t)}{\mathrm{d}t} = g(\varphi(t), x(t)), \varphi(0) = \varphi_0. \tag{3.9}$$

In this case the dynamics are given by a step function from x_0 to x_0+1 the first time $\varphi(t)$ changes position relative to φ^*, and so on; these are the dynamics we present in Eq. 3.4.

This list is obviously not exhaustive and we do not intend it to be so.

We can use kernel functions $\mathcal{K}^{(\cdot)}$ as basic building blocks of richer local mechanistic dynamics through function concatenation and the operation of the two-dimensional reflection group R_4. Elements of this group correspond to $r_0 = \text{id}$, $r_1 =$ reflection across the vertical axis (time reversal), $r_2 =$ negation (e.g., from an increase in usage frequency to a decrease in usage frequency), and $r_3 = r_1 \cdot r_2 = r_2 \cdot r_1$. We can also model new dynamics by concatenating kernels, i.e., "glueing" kernels back-to-back. For example, we can generate "cusplets" with both anticipatory and relaxation dynamics by concatenating a shocklet $\mathcal{K}^{(S)}$ with a time-reversed copy of itself:

$$\mathcal{K}^{(C)}(\tau|W,\theta) \sim \mathcal{K}^{(S)}(\tau|W,\theta) \oplus [r_1 \cdot \mathcal{K}^{(S)}(\tau|W,\theta)]. \tag{3.10}$$

We display an example of this concatenation operation in Fig. 3.2. For much of the remainder of the work, we conduct analysis using this symmetric kernel.

The discrete shocklet transform (DST) of the time series $x(t)$ is defined by

$$\mathrm{C}_{\mathcal{K}^{(S)}}(t,W|\theta) = \sum_{\tau=-\infty}^{\infty} x(\tau+t)\mathcal{K}^{(S)}(\tau|W,\theta), \tag{3.11}$$

which is the cross-correlation of the sequence and the kernel. This defines a $T \times N_W$ matrix containing an entry for each point in time t and window width W considered.

To convey a visual sense of what the DST looks like when using a shock-like, asymmetric kernel, we compute the DST of a random walk $x_t - x_{t-1} = z_t$ (we define $z_t \sim \mathcal{N}(0,1)$) using a kernel function $\mathcal{K}^{(S)}(\tau|W,\theta) \sim \text{rect}(\tau)\tau^\theta$ with $\theta = 3$ and display the resulting matrix for window sizes $W \in [10, 250]$ in Fig. 3.4. The effects of time reversal by action of r_1 are visible when comparing the first and third panels with the second and fourth panels, and the result of negating the kernel by acting on it with r_2 is apparent in the negation of the matrix values when comparing the first and second panels and with the third and fourth. For this figure, we used a random walk as an example time series here as there is, by definition, no underlying generative mechanism causing any shock-like dynamics; these dynamics appear only as a result of integrated noise. We are equally likely to see large upward-pointing shocks as large downward-pointing shocks because of this, which allows us to see the activation of both upward-pointing and downward-pointing kernel functions.

As a comparison with this null example, we computed the DST of a sociotechnical time series, the rank of the word "bling" among the LabMT words on Twitter, and two draws from a null random walk model, and displayed the results in Fig. 3.5. Here, we calculated the DST using the symmetric kernel given in Eq. 3.10. (For more statistical details of the null model, see Appendix D.2.) We also computed the DWT of each of these time series and display the resulting wavelet transform matrices next to the shocklet transform matrices in Fig. 3.5. Direct comparison of the sociotechnical time series (r_t) with the draws from the null models reveals r_t's moderate autocovariance as well as the large, shock-like fluctuation that occurs in late July of 2015. (This underlying driver of this fluctuation was the release of a popular song entitled "Hotline Bling" on July 31st, 2015.) In comparison, the draws

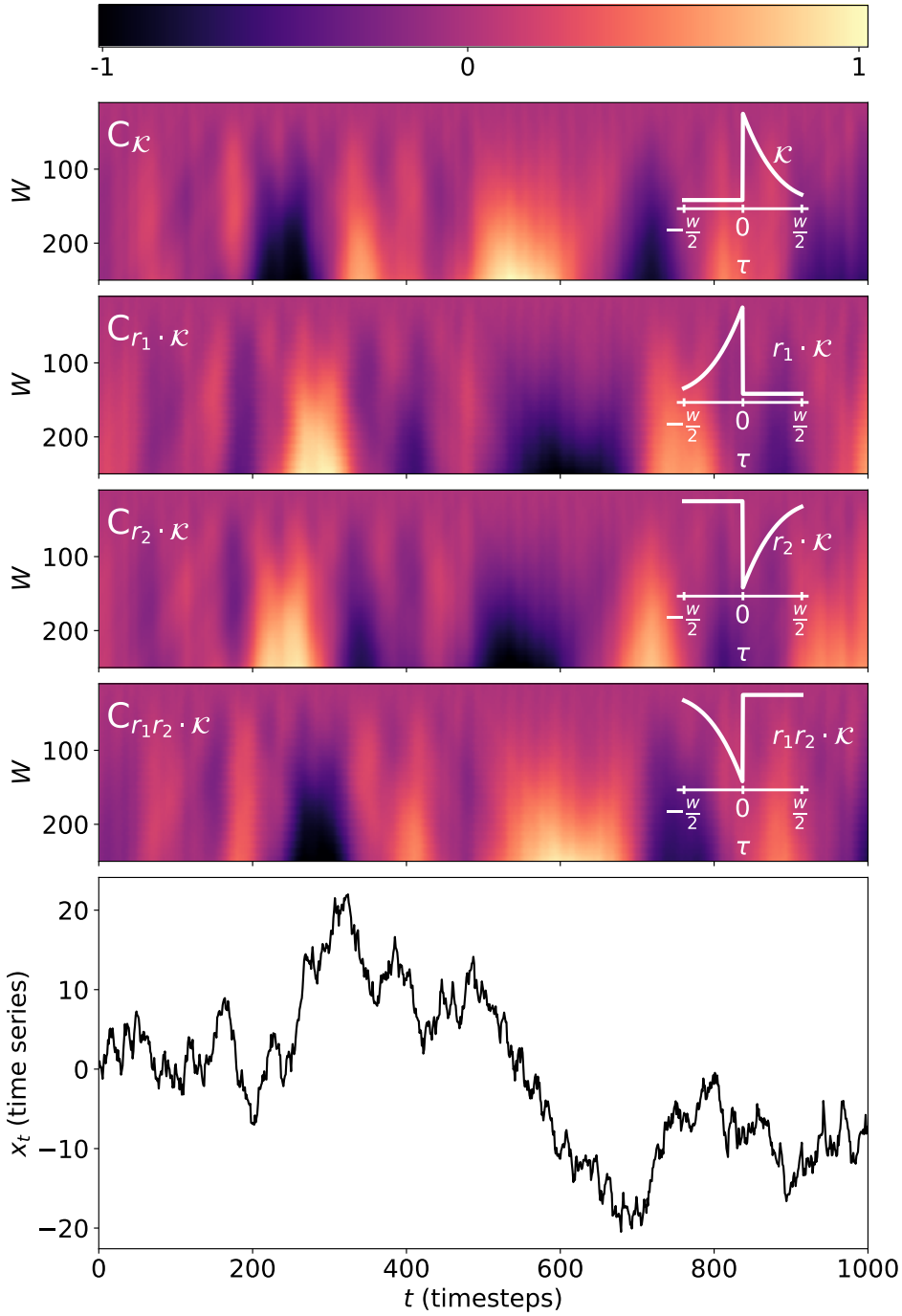

Figure 3.4: Effects of the reflection group R_4 on the shocklet transform. The top four panels display the results of the shocklet transform of a random walk $x_t = x_{t-1} + z_t$ with $z_t \sim \mathcal{N}(0, 1)$, displayed in the bottom panel, using the kernels $r_j \cdot \mathcal{K}^{(S)}$, where $r_j \in R_4$.

from the null model have a covariance with much more prominent time scaling and do not exhibit dramatic shock-like fluctuations as does r_t. Comparing the DWT of these time series with the respective DST provides more evidence that the DST exhibits superior space-time localization of shock-like dynamics than does the DWT.

To aggregate deterministic behavior across all timescales of interest, we define the shock indicator function as the function

$$\mathrm{C}_{\mathcal{K}^{(S)}}(t|\theta) = \sum_{W} \mathrm{C}_{\mathcal{K}^{(S)}}(t, W|\theta) p(W|\theta), \tag{3.12}$$

for all windows W considered. The function $p(W|\theta)$ is a probability mass function that encodes prior beliefs about the importance of particular values of W. For example, if we are interested primarily in time series that display shock- or shock-like behavior that usually lasts for approximately one month, we might specify $p(W|\theta)$ to be sharply peaked about $W = 28$ days. Throughout this work we take an agnostic view on all possible window widths and so set $p(W|\theta) \propto 1$, reducing our analysis to a strictly maximum-likelihood based approach. Summing over all values of the shocklet parameter θ defines the shock indicator function,

$$\mathrm{C}_{\mathcal{K}^{(S)}}(t) = \sum_{\theta} \mathrm{C}_{\mathcal{K}^{(S)}}(t|\theta) p(\theta) \tag{3.13}$$

$$= \sum_{\theta, W} \mathrm{C}_{\mathcal{K}^{(S)}}(t, W|\theta) p(W|\theta) p(\theta). \tag{3.14}$$

In analogy with $p(W\theta)$, the function $p(\theta)$ is a probability density function describing our prior beliefs about the importance of various values of θ. As we will show later in this section, and graphically in Fig. 3.6, the shock indicator function is relatively

insensitive to choices of θ possessing a nearly-identical ℓ_1 norm for wide ranges of θ and different functional forms of $\mathcal{K}^{(S)}$.

After calculation, we normalize $\mathrm{C}_{\mathcal{K}^{(S)}}(t)$ so that it again integrates to zero and has $\max_t \mathrm{C}_{\mathcal{K}^{(S)}}(t) - \min_t \mathrm{C}_{\mathcal{K}^{(S)}}(t) = 2$. The shock indicator function is used to find windows in which the time series displays anomalous shock- or shock-like behavior. These windows are defined as

$$\{t \in [0, T] : \text{intervals where } \mathrm{C}_{\mathcal{K}^{(S)}}(t) \geq s\}\,. \tag{3.15}$$

where the parameter $s > 0$ sets the sensitivity of the detection.

The DST is relatively insensitive to quantitative changes to its functional parameterization; it is a qualitative tool to highlight time periods of unusual events in a time series. In other words, it does not detect statistical anomalies but rather time periods during which the time series appears to take on certain qualitative characteristics without being too sensitive to a particular functional form. We analyzed two example sociotechnical time series—the rank of the word "bling" on Twitter (for reasons we will discuss presently)— and the price time series of Bitcoin (symbol BTC) [124], the most actively-used cryptocurrency [125], and of one null model, a pure random walk. For each time series, we computed the shock indicator function using two kernels, each of which had a different functional form (one kernel given by the function of Eq. 3.10 and one of the identical form but constructed by setting $\mathcal{K}^{(S)}(\tau|W, \theta)$ to the function given in Eq. 3.1), and evaluating each kernel over a wide range of its parameter θ. We also vary the maximum window size from $W = 100$ to $W = 1000$ to explore the sensitivity of the shock indicator function to this parameter. We display

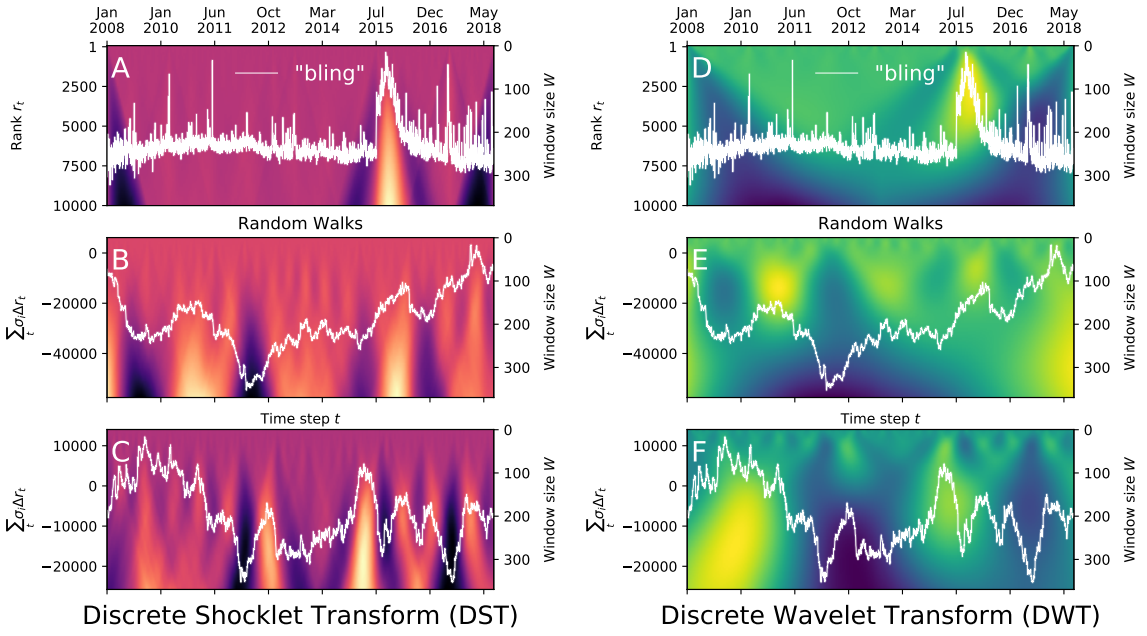

Figure 3.5: Intricate dynamics of sociotechnical time series. Panels A and D show the time series of the ranks down from top of the word "bling" on Twitter. Until mid-summer 2015, the time series presents as random fluctuation about a steady, relatively-constant level. However, the series then displays a large fluctuation, increases rapidly, and then decays slowly after a sharp peak. The underlying mechanism for these dynamics was the release of a popular song titled "Hotline Bling". To demonstrate the qualitative difference of the "bling" time series from draws from a null random walk model, the details of which are given in Appendix D.2. Panels A, B, and C show the discrete shocklet transform of the original series for "bling" and the random walks $\sum_{t' \leq t} \Delta r_{\sigma_i t}$, showing the responsiveness of the DST to nonstationary local dynamics and its insensitivity to dynamic range. Panels D, E, and F, on the other hand, display the discrete wavelet transform of the original series and of the random walks, demonstrating the DWT's comparatively less-sensitive nature to local shock-like dynamics.

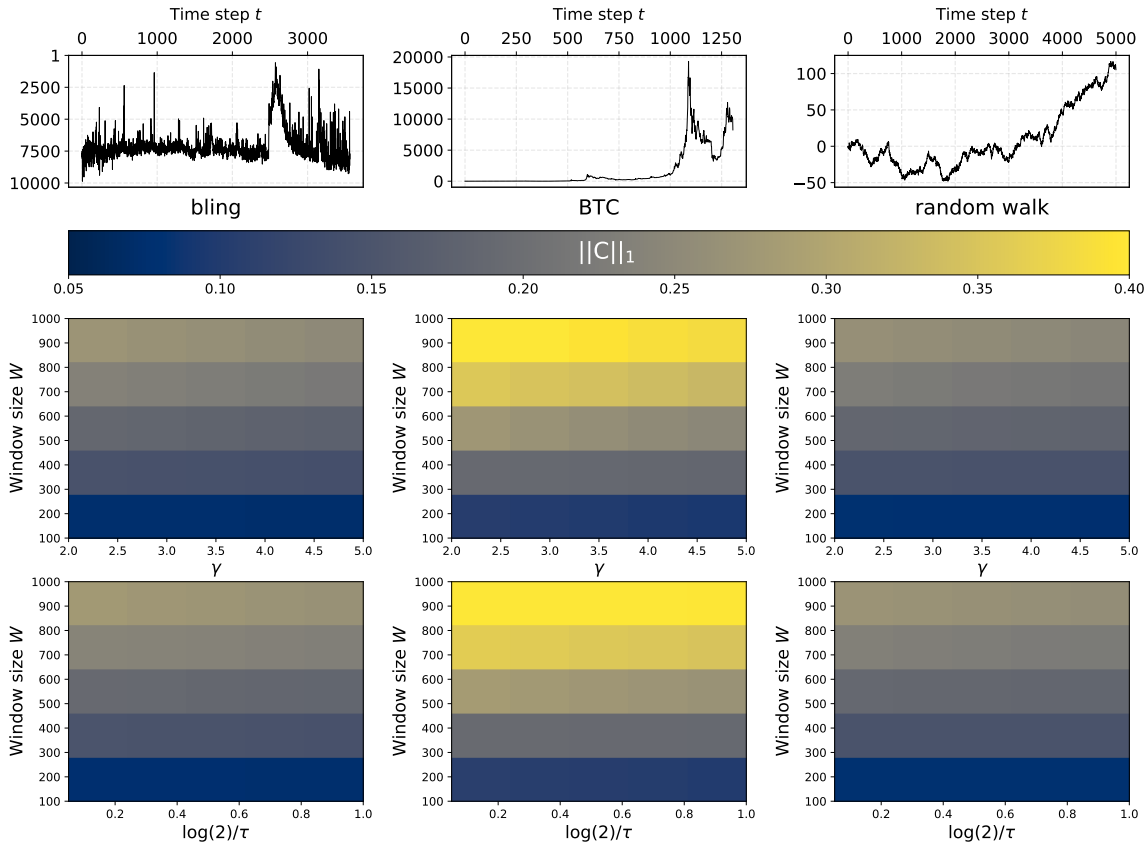

Figure 3.6: The shock indicator function is relatively insensitive to functional forms $\mathcal{K}^{(\cdot)}$ and values of the kernel's parameter vector θ so long as the kernel functions are qualitatively similar (e.g., for cusp-like dynamics—as considered in this figure and in Eq. 3.10—$\mathcal{K}^{(C)}$ displaying increasing rates of increase followed by decreasing rates of decrease). Here we have computed the shock indicator function $\mathrm{C}_{\mathcal{K}^{(S)}}(\tau|\theta)$ (Eq. 3.12) for three different time series: two sociotechnical and one null example. From left to right, the top row of figures displays the rank usage time series of the word "bling" on Twitter, the price of the cryptocurrency Bitcoin, and a simple Gaussian random walk. Below each time series we display parameter sweeps over combinations of $(\theta, W_{\max})$ for two kernel functions: one kernel given by the function of Eq. 3.10 and another of the identical form but constructed by setting $\mathcal{K}^{(S)}(\tau|W,\theta)$ to the function given in Eq. 3.1. The ℓ_1 norms of the shock indicator function are nearly invariant across the values of the parameters θ for which we evaluated the kernels. However, the shock indicator function does display dependence on the maximum window size $W_{\max}$, with large $W_{\max}$ associated with larger ℓ_1 norm. This is because a larger window size allows the DST to detect shock-like behavior over longer periods of time.

the results of this comparative analysis in Fig. 3.6. For each time series, we plot the ℓ_1 norm of the shock indicator function for each (θ, W) combination. We find that, as stated earlier in this section, the shock indicator function is relatively insensitive to both functional parameterization and value of the parameter θ; for any fixed W, the ℓ_1 norm of the shock indicator function barely changed regardless of the value of θ or choice of $\mathcal{K}^{(\cdot)}$. However, the maximum window size does have a notable effect on the magnitude of the shock indicator function; higher values of W are associated with larger magnitudes. This is a reasonable finding, since higher maximum W means that the DST is able to capture shock-like behavior that occurs over longer timespans and hence may have values of higher magnitude over longer periods than for comparatively lower maximum W.

That the shock indicator function is a relative quantity is both beneficial and problematic. The utility of this feature is that the dynamic behavior of time series derived from systems of widely-varying time and length scales can be directly compared; while the rank of a word on Twitter and—for example—the volume of trades in an equity security are entirely different phenomena measured in different units, their shock indicator functions are unitless and share similar properties. On the other hand, the Shock Indicator Function carries with it no notion of dynamic range. Two time series x_t and y_t could have identical shock indicator functions but have spans differing by many orders of magnitude, i.e., $\text{diam}\, x_t \equiv \max_t x_t - \min_t x_t \gg \text{diam}\, y_t$. (In other words, the diameter of a time series in interval I is just the dynamic range of the time series over that interval.) We can directly compare time series inclusive of their dynamic range by computing a weighted version of the shock indicator function, $\mathrm{C}_{\mathcal{K}}(t)\Delta x(t)$, which we term the weighted shock indicator function (WSIF). A simple

choice of weight is

$$\Delta x(t) = \operatorname*{diam}_{t\in[t_b,t_e]} x_t, \tag{3.16}$$

where t_b and t_e are the beginning and end times of a particular window. We use this definition for the remainder of our paper, but one could easily imagine using other weighting functions, e.g., maximum percent change (perhaps applicable for time series hypothesized to increment geometrically instead of arithmetically).

These final weighted shock indicator functions are the ultimate output of the shocklet transform and ranking (STAR) algorithm; the weighting corresponds to the actual magnitude of the dynamics and constitutes the "ranking" portion of the algorithm, while the weighting will only be substantially larger than zero if there existed intervals of time during which the time series exhibited shock-like behavior as indicated in Eq. 3.15. We present a conceptual, bird's-eye view of the STAR algorithm (of which the DST is a core component) in Fig. 3.7. Though this diagram is lacking in technical detail, we have included it in an effort to provide a bird's-eye view of the entire STAR algorithm and to help orient the reader on the conceptual process underpinning the algorithm.

Algorithmic details: Comparison with existing methods

On a coarse scale, there are five nonexclusive categories of time series data mining tasks [126]: similarity search (also termed indexing), clustering, classification, summarization, and anomaly detection. The STAR algorithm is a qualitative, shape-based, timescale-independent, similarity search algorithm. As we have shown in the previous section, the discrete shocklet transform (a core part of the overarching STAR algorithm) is *qualitative*, meaning that it does not depend too strongly on values of

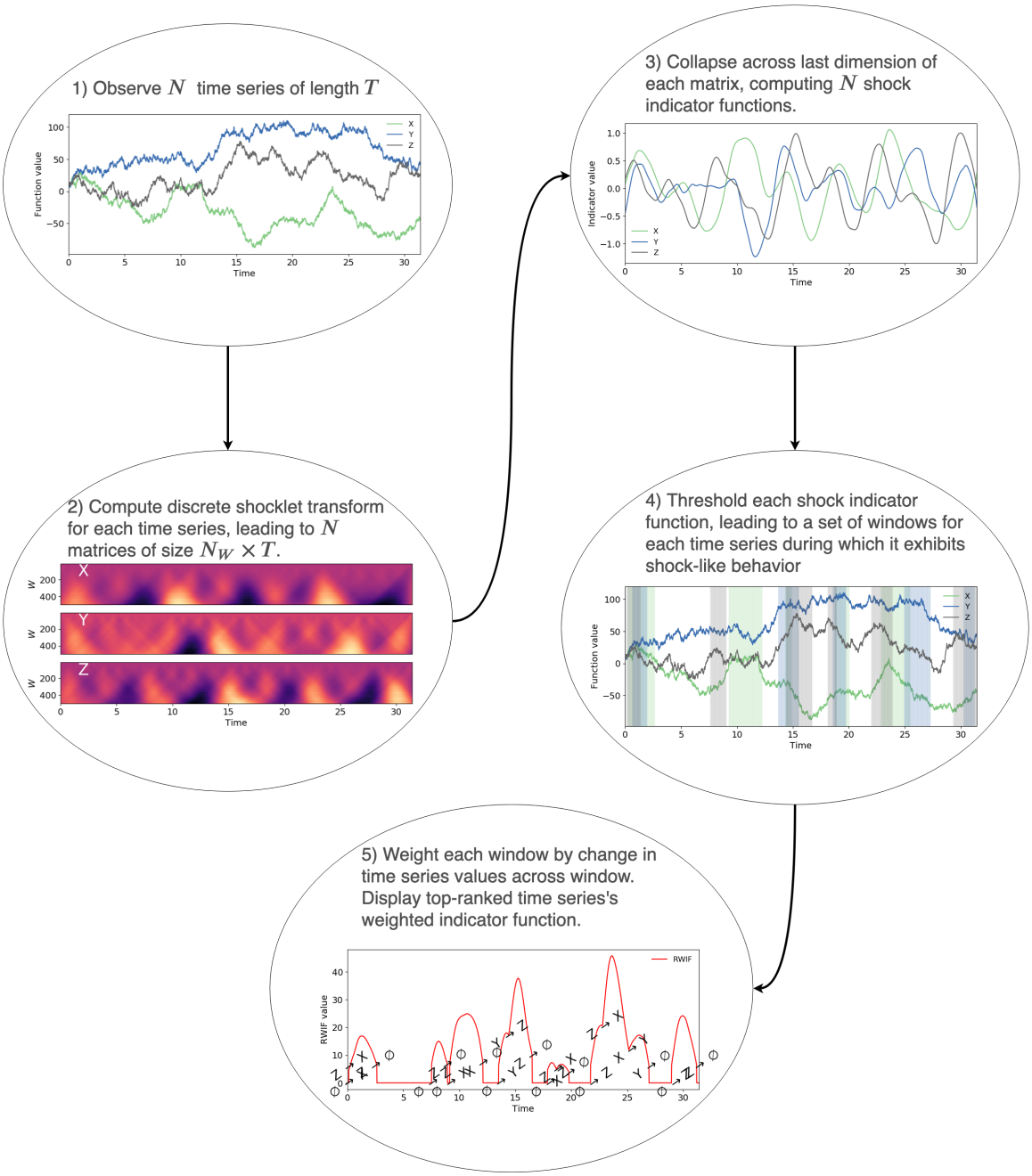

Figure 3.7: The Shocklet Transform And Ranking (STAR) algorithm combines the discrete shocklet transform (DST) with a series of transformations that yield intermediate results, such as the cusp indicator function (item (3) in the figure) and windows during which each univariate time series displays shock-like behavior (item (4) in the figure). Each of these intermediate results is useful in its own right, as we show in Sec. 3.3. We display the final output of the STAR algorithm, a univariate indicator that condenses information about which of the time series exhibits the strongest shock-like behavior at each point in time.

functional parameters or even the functions used in the cross-correlation operation themselves, as long as the functions share the same qualitative dynamics (e.g., increasing rates of increase followed by decreasing rates of decrease for cusp-like dynamics); hence, it is primarily *shape-based* rather than relying on the quantitative definition of a particular functional form. STAR is *timescale-independent* as it is able to detect shock-like dynamics over a wide range of timescales limited only by the maximum window size for which it is computed. Finally, we believe that it is best to categorize STAR as a *similarity search* algorithm as this seems to be the best-fitting label for STAR that is given in the five categories listed at the beginning of this section; STAR is designed for searching within sociotechnical time series for dynamics that are similar to the shock kernel in some way, albeit similar in a qualitative sense and over any arbitrary timescale, not functionally similar in numerical value and characteristic timescale. However, it could also be considered a type of qualitative, shape-based anomaly detection algorithm because we are searching for behavior that is, in some sense, anomalous compared to a usual baseline behavior of many time series (though see discussion at the beginning of the anomaly detection subsection near the end of this section: STAR is an algorithm that can detect defined anomalous behavior, *not* an algorithm to detect arbitrary statistical anomalies).

As such, we are unaware of any existing algorithm that satisfies these four criteria and believe that STAR represents an entirely new class of algorithms for sociotechnical time series analysis. Nonetheless, we now provide a detailed comparison of the DST with other algorithms that solve related problems, and in Sec. 3.3.1 provide an in-depth quantitative comparison with another nonparametric algorithm (Twitter's anomaly detection algorithm) that one could attempt to use to extract shock-like

dynamics from sociotechnical time series.

Similarity search - here the objective is to find time series that minimize some similarity criterion between candidate time series and a given reference time series. Algorithms to solve this problem include nearest-neighbor methods (e.g., k-nearest neighbors [127] or a locality-sensitive hashing-based method [128, 129]), the discrete Fourier and wavelet transforms [130, 131, 101, 132]; and bit-, string-, and matrix-based representations [133, 134, 126, 135]. With suitable modification, these algorithms can also be used to solve time series clustering problems. Generic dimensionality-reduction techniques, such as singular value decomposition / principal components analysis [136, 137, 138], can also be used for similarity search by searching through a dataset of lower dimension. Each of these classes of algorithms differs substantially in scope from the discrete shocklet transform. Chief among the differences is the focus on the entire time series. While the discrete shocklet transform implicitly searches the time series for similarity with the kernel function at all (user-defined) relevant timescales and returns qualitatively-matching behavior at the corresponding timescale, most of the algorithms considered above do no such thing; the user must break the time series into sliding windows of length τ and execute the algorithm on each sliding window; if the user desires timescale-independence, they must then vary τ over a desired range. An exception to this statement is Mueen's subsequence similarity search algorithm (MSS) [139], which computes sliding dot products (cross-correlations) between a long time series of length T and a shorter kernel of length M before defining a Euclidean distance objective for the similarity search task. When this sliding dot product is computed using the fast Fourier transform, the computational complexity of this task is $\mathcal{O}(T \log T)$. This computational step is also at the core of the discrete

shocklet transform, but is performed for multiple kernel function arrays (more precisely, for the kernel function resampled at multiple user-defined timescales). Unlike the discrete shocklet transform, MSS does not subsequently compute an indicator function and does not have the self-normalizing property, while the matrix profile algorithm [135] computes an indicator function of sorts (their "matrix profile") but is not timescale-independent and is quantitative in nature; it does not search for a qualitative shape match as does the discrete shocklet transform. We are unaware of a similarity-search algorithm aside from STAR that is both qualitative in nature and timescale-independent.

Clustering - given a set of time series, the objective is to group them into groups, or clusters, that are more homogeneous within each cluster than between clusters. Viewing a collection of N time series of length T as a set of vectors in $\mathbb{R}^T$, any clustering method that can be effectively used on high-dimensional data has potential applicability to clustering time series. Some of these general clustering methods include k-means and k-medians algorithms [140, 141, 142], hierarchical methods [143, 144, 145], and density-based methods [143, 146, 147, 148]. There are also methods designed for clustering time series data specifically, such as error-in-measurement models [149], hidden Markov models [150], simulated annealing-based methods [151], and methods designed for time series that are well-fit by particular classes of parametric models [152, 153, 154, 155]. Although the discrete shocklet transform component of the STAR algorithm could be coerced into performing a clustering task by using different kernel functions and elements of the reflection group, clustering is not the intended purpose of the discrete shocklet transform or STAR more generally. In addition, none of the clustering methods mentioned replicate the results of the STAR

algorithm. These clustering methods uncover groups of time series that exhibit similar behavior over their entire domain; application of clustering methods to time series subsequences carries leads to meaningless results [156]. Clustering algorithms are also shape-independent in the sense that they cluster data into groups that share similar features, but do not search for specific known features or shapes in the data. In contrast with this, when using the STAR algorithm we already have specified a specific shape—for example, the shock shape demonstrated above—and are searching the data across timescales for occurrences of that shape. The STAR algorithm also does not require multiple time series in order to function effectively, differing from any clustering algorithm in this respect; a clustering algorithm applied to $N = 1$ data points trivially returns a single cluster containing the single data point. The STAR algorithm operates identically on one or many time series as it treats each time series independently.

Classification - classification is the canonical supervised statistical learning problem in which data x_i is observed along with a discrete label y_i that is taken to be a function of the data, $y_i = f(x_i) + \varepsilon$; the goal is to recover an approximation to f that precisely and accurately reproduces the labels for new data [157]. This is the category of time series data mining algorithms that least corresponds with the STAR algorithm. The STAR algorithm is unsupervised—it does not require training examples ("correct labels") in order to find subsequences that qualitatively match the desired shape. As above, the STAR algorithm also does not require multiple time series to function well, while (non-Bayesian) classification algorithms rely on multiple data points in order to learn an approximation to f [2].

[2] Bayesian classification algorithms can perform classification based only on prior information, but this is also not similar to the STAR algorithm, since the STAR algorithm is a maximum-likelihood

Summarization - since time series can be arbitrarily large and composed of many intricately-related features, it may be desirable to have a summary of their behavior that encompasses the time series's "most interesting" features. These summaries can be numerical, graphical, or linguistic in nature. Underlying methodologies for time series summary tasks include wavelet-based approaches [158, 159], genetic algorithms [160, 161], fuzzy logic and systems [162, 163, 164], and statistical methods [165]. Though intermediate steps of the STAR algorithm can certainly be seen as a time series summarization mechanism (for example, the matrix computed by the DShT or the weighted shock indicator functions used in determinning rank relevance of individual time series at different points in time), the STAR algorithm was not designed for time series summarization and should not be used for this task as it will be outperformed by essentially any other algorithm that was actually designed for summarization. Any "summary" derived from the STAR algorithm will have utility only in summarizing segments of the time series the behavior of which match the kernel shape, or in distinguishing segments of the time series that do have a similar shape as the kernel from ones that do not.

Anomaly detection - if a "usual" model can be defined for the system under study, an anomaly detection algorithm is a method that finds deviations from this usual behavior. Before we briefly review time series anomaly detection algorithms and compare them with the STAR algorithm, we distinguish between two subtly different concepts: this data mining notion of anomaly detection, and the physical or social scientific notion of anomalous behavior. In the first sense, *any* deviation from the "ordinary" model is termed an anomaly and marked as such. The ordinary model may not be a

method that by definition requires at least one time series to operate.

parametric model to which the data is compared; for example, it may be implicitly defined as the behavior that the data exhibits most of the time [166]. In physical and social sciences, on the other hand, it may be observed that, given a particular set of laboratory or observational conditions, a material, state vector, or collection of agents exhibits phenomena that is anomalous when compared to a specific reference situation, even if this behavior is "ordinary" for the conditions under which the phenomena is observed. Examples of such anomalous behavior in physics and economics include: spectral behavior of polychromatic waves that is very unusual compared to the spectrum of monochromatic waves (even though it is typical for polychromatic waves near points where the wave's phase is singular) [167]; the entire concept of anomalous diffusion, in which diffusive processes with mean square displacement (autocovariance functions) scaling as $\langle r(t) \rangle \sim t^{\alpha}$ are said to diffuse anomalously if $\alpha \not\approx 1$ (since $\alpha = 1$ is the scaling of the Wiener process's autocovariance function) [168, 169], even though anomalous diffusion is the rule rather than the exception in intra-cellular and climate dynamics, as well as financial market fluctuations; and behavior that deviates substantially from the "rational expectations" of non-cooperative game theory, even though such deviations are regularly observed among human game players [170, 171]. This distinction between algorithms designed for the task of anomaly detection and algorithms or statistical procedures that test for the existence of anomalous behavior, as defined here, is thus seen to be a subtle but significant difference. The DST and STAR algorithm fall into the latter category: the purpose for which we designed the STAR algorithm is to extract windows of anomalous behavior as defined by comparison with a particular null qualitative time series model (absence of clear shock-like behavior), not to perform the task of anomaly detection writ large by indicating the

presence of arbitrary samples or dynamics in a time series that does not in some way comport with the statistics of the entire time series.

With these caveats stated, it is not the case that there is no overlap between anomaly detection algorithms and algorithms that search for some physically-defined anomalous behavior in time series; in fact, as we show in Sec. 3.3.1, there is some significant convergence between windows of shock-like behavior indicated by STAR and windows of anomalous behavior indicated by Twitter's anomaly detection algorithm when the underlying time series exhibits relatively low variance. Statistical anomaly detection algorithms typically propose a semi-parametric model or nonparametric test and confront data with the model or test; if certain datapoints are very unlikely under the model or exceed certain theoretical boundaries derived in constructing the test, then these datapoints are said to be anomalous. Examples of algorithms that operate in this way include: Twitter's anomaly detection algorithm (ADV), which relies on generalized seasonal ESD test [172, 173]; the EGADS algorithm, which relies on explicit time series models and outlier tests [174]; time-series model and graph methodologies [175, 176]; and probabilistic methods [177, 178]. Each of these methods is strictly focused on solving the first problem that we outlined at the beginning of this subsection: that of finding points in one or more time series during which it exhibits behavior that deviates substantially from the "usual" or assumed behavior for time series of a certain class. As we outlined, this goal differs substantially from the one for which we designed STAR: searching for segments of time series (that may vary widely in length) during which the time series exhibits behavior that is qualitatively similar to underlying deterministic dynamics (shock-like behavior) that we believe is anomalous when compared to non-sociotechnical time series.

3.3 Empirical results

3.3.1 Comparison with Twitter's anomaly detection algorithm

Through the literature review in Sec. 4.2 we have demonstrated that, to our knowledge, there exists no algorithm that solves the same problem for which STAR was designed—to provide a qualitative, shape-based, timescale-independent measure of similarity between multivariate time series and a hypothesized shape generated by mechanistic dynamics. However, there are existing algorithms designed for nonparametric anomaly detection that could be used to alert to the presence of shock-like behavior in sociotechnical time series, which is the application for which we originally designed STAR. One leading example of such an algorithm is Twitter's Anomaly Detection Vector (ADV) algorithm [3]. This algorithm uses an underlying statistical test, seasonal-hybrid ESD, to test for the presence of outliers in periodic and nonstationary time series [172, 173]. We perform a quantitative and qualitative comparison between the STAR and ADV to compare their effectiveness at the task for which we designed STAR—determining qualitative similarity between shock-like shapes over a wide range of timescales—and to contrast the signals picked up by each algorithm, which, as we show, differ substantially. Before presenting results of this analysis, we note that this comparison is not entirely fair; though ADV is a state-of-the-art anomaly detection algorithm, it was not designed for the task for which we designed STAR, and so it is not exactly reasonable to assume that ADV would perform as well

[3] https://github.com/twitter/AnomalyDetection.

as STAR on this task. In an attempt to ameliorate this problem, we have chosen a quantitative benchmark for which our *a priori* beliefs did not favor the efficacy of either algorithm.

As both STAR and ADV are unsupervised algorithms, we compare their quantitative performance by assessing their utility in generating features for use in a supervised learning problem. Since the macro-economy is a canonical example of a sociotechnical system, we consider the problem of predicting the probability of a U.S. economic recession using only a minimal set of indicators from financial market data. Models for predicting economic recessions variously use only real economic indicators [179, 180, 181], only financial market indicators [182, 183], or a combination of real and financial economic indicators [184, 185]. We take an approach that is both simple and relatively granular, focusing on the ability of statistics of individual equity securities to jointly model U.S. economic recession probability. For each of the equities that was in the Dow Jones Industrial Average between 1999-07-01 to 2017-12-31 (a total of $K = 32$ securities), we computed both the DST (outputting the shock indicator function), STAR algorithm (outputting windows of shock-like behavior), and the ADV routine on that equity's volume traded time series (number of shares transacted), which we sampled at a daily resolution for a total of $T = 6759$ observations for each security. We then fit linear models of the form

$$E\left[\log \frac{\mathbf{p}}{\mathbf{1}-\mathbf{p}}\right] = \mathbf{X}\boldsymbol{\beta}, \tag{3.17}$$

where p_t is the recession probability on day t as given by the U.S. Federal Reserve (hence $\mathbf{p}$ is the length-T vector of recession probabilities) [4]. When we the model rep-

[4] Data is available at https://fred.stlouisfed.org/series/RECPROUSM156N.

resented by Eq. 3.17 using ADV or STAR as the algorithms generating features, the design matrix $\mathbf{X}$ is a binary matrix of shape $T \times (K+1)$ with entry X_{tk} equal to one if the algorithm indicated an anomaly or shock-like behavior respectively in security k at time t and equal to zero if it did not (the $+1$ in the dimensionality of the matrix corresponds to the prepended column of ones that is necessary to fit an intercept in the regression). When we fit the model using the shock indicator function generated by the DST, the matrix $\mathbf{X}$ is instead given by the matrix with column k equal to the shock indicator function of security k. We evaluate the goodness of fit of these linear models using the proportion of variance explained (R^2) statistic; these results are summarized graphically in Fig. 3.8. The linear using ADV-indicated anomalies as features had $R^2_{ADV} = 0.341$, while the model using the shock indicator function as columns of the design matrix had $R^2_{DST} = 0.455$ and the model using STAR-indicated shocks as features had $R^2_{STAR} = 0.496$. This relative ranking of feature importance remained constant when we used model log-likelihood ℓ as the performance metric instead of R^2, with ADV, DST, and STAR respectively exhibiting $\ell_{ADV} = -16,278$, $\ell_{DST} = -15,633$, and $\ell_{STAR} = -15,372$. Each linear model exhibited a distribution of residuals ε_t that did not drastically violate the zero-mean and distributional-shape assumptions of least-squares regression; a maximum likelihood fit of a normal probability density to the empirical error probability distribution $p(\varepsilon_t)$ gave mean and variance as $\mu = 0$ to within numerical precision and $\sigma^2 \approx 6.248$, while a maximum likelihood fit of a skew-normal probability density [186] to the empirical error probability distribution gave mean, variance, and skew as $\mu \approx 0.043$, $\sigma^2 \approx 6.025$, and $a \approx 2.307$. Taken in the aggregate, these results constitute evidence to suggest that features generated by the DST and STAR algorithms are superior in the task of clas-

sifying time periods as belonging to recessions or not than are features derived from the ADV method.

As a further comparison of the STAR algorithm and ADV, we generated anomaly windows (in the case of ADV) and windows of shock-like behavior (in the case of STAR) for the usage rank time series of each of the 10,222 words in the LabMT dataset. We computed the Jaccard similarity index for each word w (also known as the intersection over union) between the set of STAR windows $\{I_i^{STAR}(w)\}_i$ and the set of ADV windows $\{I_i^{ADV}(w)\}_i$,

$$J_w(STAR, ADV) = \frac{\left(\bigcup_i I_i^{STAR}(w)\right) \cap \left(\bigcup_i I_i^{ADV}(w)\right)}{\bigcup_{j\in\{STAR,ADV\}} \bigcup_i I_i^j(w)}. \tag{3.18}$$

We display the word time series and ADV and STAR windows for a selection of words pertaining to the 2016 U.S. presidential election in Fig. 3.9. (These words display shock-like behavior in a time interval surrounding the election, as we demonstrate in the next section, hence our selection of them as examples here.) A figure for each word that depicts the usage rank time series along with ADV and STAR-indicated windows is available at the authors' website [5]. We display the distribution of all Jaccard similarity coefficients in Fig. 3.10. Most words have relatively little overlap between anomaly windows returned by ADV and windows of shock-like dynamics returned by STAR, but there are notable exceptions. In particular, a review of the figures contained in the online index suggests that ADV's and STAR's windows overlap most when the shock-like dynamics are particularly strong and surrounded by a time series with relatively low variance; they agree the most when hypothesized un-

[5] http://compstorylab.org/shocklets/all_word_plots/

derlying deterministic mechanics are strongest and the effects of noise are lowest. The pronounced spikes in the words "crooked" and "stein" in Fig. 3.9 are an example of this phenomenon. However, when the time series has high variance or exhibits strong nonstationarity, ADV often does not indicate that there are windows of anomalous behavior while STAR does indicate the presence of shock-like dynamics; the panels of the words "trump", "jill", and "hillary" in Fig. 3.9 demonstrate these behaviors.

Taken in the aggregate, these results suggest that a state-of-the-art anomaly detection algorithm, such as Twitter's ADV, and a qualitative, shape-based, timescale-independent similarity search algorithm, such as STAR, do have some overlapping properties but are largely mutually-complementary approaches to identifying and analyzing the behavior of sociotechnical time series. While ADV and STAR both identify strongly shock-like dynamics that occur when the surrounding time series has relatively low variance, their behavior diverges when the time series is strongly nonstationary or has high variance. In this case, ADV is an excellent tool for indicating the presence of strong outliers in the data, while STAR continues to indicate the presence of shock-like dynamics in a manner that is less sensitive to the time series's stationarity or variance.

3.3.2 Social narrative extraction

We seek both an understanding of the intertemporal semantic meaning imparted by windows of shock-like behavior indicated by the STAR algorithm and a characterization of the dynamics of the shocks themselves. We first compute the shock indicator and weighted shock indicator functions (WSIFs) for each of the 10,222 labMT words filtered from the gardenhose dataset, described in section 3.2.1, using a power kernel

with $\theta = 3$. At each point in time, words are sorted by the value of their WSIF. The j-th highest valued WSIF at each temporal slice, when concatenated across time, defines a new time series. We perform this computation for the top ranked $k = 20$ words for the entire time under study. We also perform this process using the "spike" kernel of Eq. 3.4 and display each resulting time series in Fig. 3.11 (shock kernel) and Fig. 3.12 (spike kernel). (We term the spike kernel as such because we have $\frac{\mathrm{d}\mathcal{K}^{(Sp)}(\tau)}{\mathrm{d}\tau} = \delta(\tau)$ on the domain $[-W/2, W/2]$, the Dirac delta function; its underlying mechanistic dynamics are completely static except for one point in time during which the system is driven by an ideal impulse function.) The $j = 1$ word time series is annotated with the corresponding word at relative maxima of order 40. (A relative maximum x_s of order k in a time series is a point that satisfies $x_s > x_t$ for all t such that $|t - s| \leq k$.) This annotation reveals a dynamic social narrative concerning popular events, social movements, and geopolitical fluctuation over the past near-decade. Interactive versions of these visualizations are available on the authors' website [6]. To further illuminate the often-turbulent dynamics of the top j ranked weighted shock indicator functions, we focus on two particular 60-day windows of interest, denoted by shading in the main panels of Figs. 3.11 and 3.12. In Fig. 3.11, we outline a period in late 2011 during which multiple events competed for collective attention:

- the 2012 U.S. presidential election (the word "herman", referring to Herman Cain, a presidential election contender);
- Occupy Wall Street protests ("occupy" and "protestors");
- and the U.S. holiday of Thanksgiving ("thanksgiving")

[6] http://compstorylab.org/shocklets/

Each of these competing narratives is reflected in the top-left inset. In the top right inset, we focus on a time period during which the most distinct anomalous dynamics corresponded to the 2014 Gaza conflict with Israel ("gaza", "israeli", "palestinian", "palestinians", "gathered"). In Fig. 3.12, we also outline two periods of time: one, in the top left panel, demonstrates the competition for social attention between geopolitical concerns:

- street protests in Egypt ("protests", "protesters" "egypt", "response");
- and popular artists and popular culture ("rebecca", referring to Rebecca Black, a musician, and "@ddlovato", referring to another musician, Demi Lovato).

In the top right panel we demonstrate that the most prominent dynamics during late 2015 are those of the language surrounding the 2016 U.S. presidential election immediately after Donald Trump announced his candidacy ("trump", "sanders", "donald", "hillary", "clinton", "maine").

We note that these social narratives uncovered by the STAR algorithm might not emerge if we used a different algorithm in an attempt to extract shock-like dynamics in sociotechnical time series. We have already shown (in the previous section) that at least one state-of-the-art anomaly detection algorithm is unlikely to detect abrupt, shock-like dynamics that occur in time series that are nonstationary or have high variance. We display side-by-side comparisons of the indicator windows generated by each algorithm for every word in the LabMT dataset in the online appendix (http://compstorylab.org/shocklets/all_word_plots/). A review of figures in the online appendix corresponding with words annotated in Figs. 3.11 and 3.12 provides evidence that an anomaly detection algorithm, such as ADV, may not necessarily

capture the sane dynamics as does STAR. We include selected panels of these figures in Appendix D.3, displaying words corresponding with some peaks of the weighted shock and spike indicator functions. (We hasten to note that this of course does not preclude the possibility that anomaly detection algorithms might indicate dynamics that are not captured by STAR.)

3.3.3 Typology of local mechanistic dynamics

To further understand divergent dynamic behavior in word rank time series, we analyze regions of these time series for which Eq. 3.15 is satisfied—that is, where the value of the shock indicator function is greater than the sensitivity parameter. We focus on shock-like dynamics since these dynamics qualitatively describe aggregate social focusing and subsequent de-focusing of attention mediated by the algorithmic substrate of the Twitter platform. We extract shock segments from the time series of all words that made it into the top $j = 20$ ranked shock indicator functions at least once. Since shocks exist on a wide variety of dynamic ranges and timescales, we normalize all extracted shock segments to lie on the time range $t_{\text{shock}} \in [0, 1]$ and have (spatial) mean zero and variance unity. Shocks have a focal point about their maxima by definition, but in the context of stochastic time series (as considered here), the observed maximum of the time series may not be the "true" maximum of the hypothesized underlying deterministic dynamics. Shock points—hypothesized deterministic maxima—of the extracted shock segments were thus determined by two methods: The maxima of the within-window time series,

$$t_1^* = \underset{t_{\text{shock}} \in [0,1]}{\arg\max}\, x_{t_{\text{shock}}}; \tag{3.19}$$

and the maxima of the time series's shock indicator function,

$$t_2^* = \underset{t_{\text{shock}} \in [0,1]}{\arg\max}\, \mathrm{C}_{\mathcal{K}^{(S)}}(t_{\text{shock}}). \tag{3.20}$$

We then computed empirical probability density functions of t_1^* and t_2^* across all words in the LabMT dataset. While the empirical distribution of t_1^* is uni-modal, the corresponding empirical distribution of t_2^* demonstrated clear bi-modality with peaks in the first and last quartiles of normalized time. To better characterize these maximum *a posteriori* (MAP) estimates, we sample those shock segments x_t the maxima of which are temporally-close to the MAPs and calculate spatial means of these samples,

$$\langle x_{t_{\text{shock}}} \rangle_n = \frac{1}{|\mathcal{M}|} \sum_{n \in \mathcal{M}} x^{(n)}_{t_{\text{shock}}}, \tag{3.21}$$

where

$$.\mathcal{M} = \left\{ n : \left| \underset{t_{\text{shock}} \in [0,1]}{\arg\max}\, x^{(n)}_{t_{\text{shock}}} - t^* \right| < \varepsilon \right\}. \tag{3.22}$$

The number ε is a small value which we set here to $\varepsilon = 10/503$ [7]. We plot these curves in Fig. 3.13. Shock segments that are close in spatial norm to the $\langle x_{t_{\text{shock}}} \rangle_n$—that is, shock segments $x_{t_{\text{shock}}}$ that satisfy

$$\| x_{t_{\text{shock}}} - \langle x_{t_{\text{shock}}} \rangle_n \|_1 \leq F^{\leftarrow}_{\| x_s - \langle x_{t_{\text{shock}}} \rangle_n \|_1}(0.01), \tag{3.23}$$

[7] This value comes from an arbitrary but small number of indices (five) we allow a shock segment to vary ($\pm$) about the index of the MAP estimate of the distributions of shock points, each of which can be considered as multinomial distributions supported on a 503-dimensional vector space. The number 503 is the dimension of each shock segment after time normalization since the longest original shock segment in the labMT dataset was 503 days.

where $F_Z^{\leftarrow}(q)$ is the quantile function of the random variable Z—are plotted in thinner curves. From this process, three distinct classes of shock segments emerge, corresponding with the three relative maxima of the shock point distributions outlined above:

- **Type I**: exhibiting a slow buildup (anticipation) followed by a fast relaxation;
- **Type II**: with a correspondingly short buildup (shock) followed by a slow relaxation;
- **Type III**: exhibiting a relatively symmetric shape.

Words corresponding to these classes of shock segments differ in semantic context. Type I dynamics are related to known and anticipated societal and political events and subjects, such as:

- "hampshire" and "republican", concerning U.S. presidential primaries and general elections,
- "labor", "labour", and "conservatives", likely concerning U.K. general elections,
- "voter", "elected", and "ballot", concerning voting in general, and
- "grammy", the music awards show.

To contrast, Type II (shock-like) dynamics describe events that are partially- or entirely-unexpected, often in the context of national or international crises, such as:

- "tsunami" and "radiation", relating to the Fukushima Daichii tsunami and nuclear meltdown,

Classification	Shock shape	Words
Type I	Slow buildup, fast relaxation	rumble, veterans, dusty, labour, scattered, hampshire, #tinychat, elected, ballot, selection, labor, entering, beam, phenomenon, voters, mamma, anonymity, republican, #nowplaying, indictment, wages, conservatives, pulse, knee, grammy, essays, #tcot, kentucky, fml, netherlands, jingle, valid, whitman, syracuse, dems, deposit, bail, tomb, walker, reader
Type II	Fast buildup, slow relaxation	xbox, chained, yale, bombing, holocaust, connecticut, #tinychat, civilian, jill, turkish, tsunami, ferry, #letsbehonest, beam, agreement, riley, ethics, phenomenon, harriet, privacy, israeli, #nowplaying, gun, dub, pulse, killings, herman, enormous, fbi, dmc, searched, norman, joan, affected, arthur, sandra, radiation, army, walker, reader,
Type III	Roughly symmetric	rumble, memorial, sleigh, veterans, costumes, greeks, britney, separated, father's, shark, grammys, labor, costume, x-mas, bunny, commonwealth, clause, olympics, olympic, daylight, cyber, wrapping, rudolph, drowned, re-election

Table 3.1: Words for which at least one shock segment was close in norm to a spatial mean shock segment as detailed in Section 3.3. We display the distributions of "shock points"—hypothesized deterministic maxima of the noisy mechanistically-generated time series—in Fig. 3.13. Every word may also have several "shock points" where each point could corresponds to a different shock dynamics due to the way each word is used throughout its life span on the platform, hence a few of these examples (e.g. rumble, anonymity, #nowplaying) appear in multiple categories.

- “bombing”, “gun”, “pulse”, “killings”, and “connecticut”, concerning acts of violence and mass shootings, in particular the Sandy Hook elementary school shooting in the United States;
- “jill” (Jill Stein, a 2016 U.S. presidential election competitor), “ethics”, and “fbi”, pertaining to surprising events surrounding the 2016 U.S. presidential election, and
- “turkish”, “army”, “israeli”, “civilian”, and “holocaust”, concerning international protests, conflicts, and coups.

Type III dynamics are associated with anticipated events that typically re-occur and are discussed substantially after their passing, such as

- “sleigh”, “x-mas”, “wrapping”, “rudolph”, “memorial”, “costumes”, “costume”, “veterans”, and “bunny”, having to do with major holidays, and
- “olympic” and “olympics”, relating to the Olympic games.

We give a full list of words satisfying the criteria given in Eqs. 3.22 and 3.23 in Table **??**. We note that, though the above discussion defines and distinguishes three fundamental signatures of word rank shock segments, these classes are only the MAP estimates of the true distributions of shock segments, our empirical observations of which are displayed as histograms in Fig. 3.13; there is an effective continuum of dynamics that is richer, but more complicated, than our parsimonious description here.

3.4 Discussion

We have introduced a nonparametric pattern detection method, termed the discrete shocklet transform (DST) for its particular application in extracting shock- and shock-like dynamics from noisy time series, and demonstrated its particular suitability for analysis of sociotechnical data. Though extracted social dynamics display a continuum of behaviors, we have shown that maximizing *a posteriori* estimates of shock likelihood results in three distinct classes of dynamics: anticipatory dynamics with long buildups and quick relaxations, such as political contests (Type I); "surprising" events with fast (shock-like) buildups and long relaxation times, examples of which are acts of violence, natural disasters, and mass shootings (Type II); and quasi-symmetric dynamics, corresponding with anticipated and talked-about events such as holidays and major sporting events (Type III). We analyzed the most "important" shock-like dynamics—those words that were one of the top-20 most significant at least once during the decade of study—and found that Type III dynamics were the most common among these words (40.9%) followed by Type II (36.4%) and Type I (22.7%). We then showcased the discrete shocklet transform's effectiveness in extracting coherent intertemporal narratives from word usage data on the social microblog Twitter, developing a graphical methodology for examining meaningful fluctuations in word—and hence topic—popularity. We used this methodology to create document-free nonparametric topic models, represented by pruned networks based on shock indicator similarity between two words and defining topics using the networks' community structures. This construction, while retaining artifacts from its construction using intrinsically-temporal data, presents topics possessing qualitatively sensible semantic

structure.

There are several areas in which future work could improve on and extend that presented here. Though we have shown that the discrete shocklet transform is a useful tool in understanding non-stationary local behavior when applied to a variety of sociotechnical time series, there is reason to suspect that one can generalize this method to essentially any kind of noisy time series in which it can be hypothesized that mechanistic local dynamics contribute a substantial component to the overall signal. In addition, the DST suffers from noncausality, as do all convolution or frequency-space transforms. In order to compute an accurate transformed signal at time t, information about time $t + \tau$ must be known to avoid edge effects or spectral effects such as ringing. In practice this may not be an impediment to the DST's usage, since: empirically the transform still finds "important" local dynamics, as shown in Fig. 3.11 near the very beginning (the words "occupy" and "slumdog" are annotated) and the end (the words "stormy" and "cohen" are annotated) of time studied. Furthermore, when used with more frequently-sampled data the lag needed to avoid edge effects may have decreasing length relative to the longer timescale over which users interact with the data. However, to avoid the problem of edge effects entirely, it may be possible to train a supervised learning algorithm to learn the output of the DST at time t using only past (and possibly present) data. The DST could also serve as a useful counterpart to phrase- and sentence-tracking algorithms such as MemeTracker [187, 188]. Instead of applying the DST to time series of simple words, one could apply it to arbitrary n-grams (including whole sentences) or sentence structure pattern matches to uncover frequency of usage of verb tenses, passive/active voice construction, and other higher-order natural language constructs. Other work

could apply the DST to more and different natural language data sources or other sociotechnical time series, such as asset prices, economic indicators, and election polls.

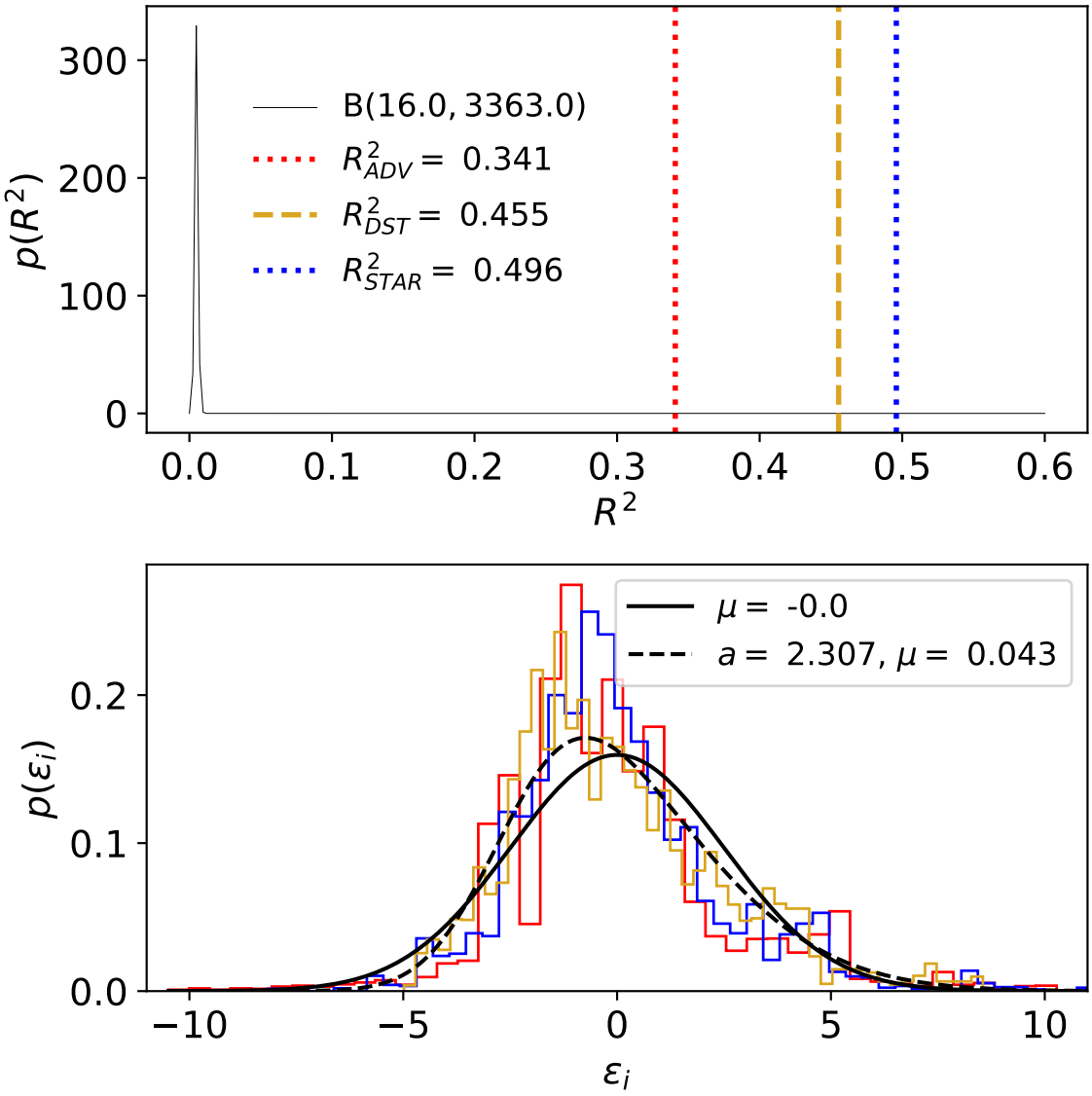

Figure 3.8: We modeled the log odds ratio of a U.S. economic recession using three ordinary least squares regression models. Each model used one of the ADV method's anomaly indicator, the shock indicator function resulting from the discrete shocklet transform, and the windows of shock-like behavior output by the STAR algorithm as elements of the design matrix. The models that used features constructed by the DST or STAR outperformed the model that used features constructed by ADV as measured by both R^2 (displayed in the top panel) and model log-likelihood. The black curve in the top panel displays the null distribution of R^2 under the assumption that no regressor (column of the design matrix) actually belongs to the true linear model of the data [3, 4]. The lower panel displays the empirical probability distributions of the model residuals ε_i.

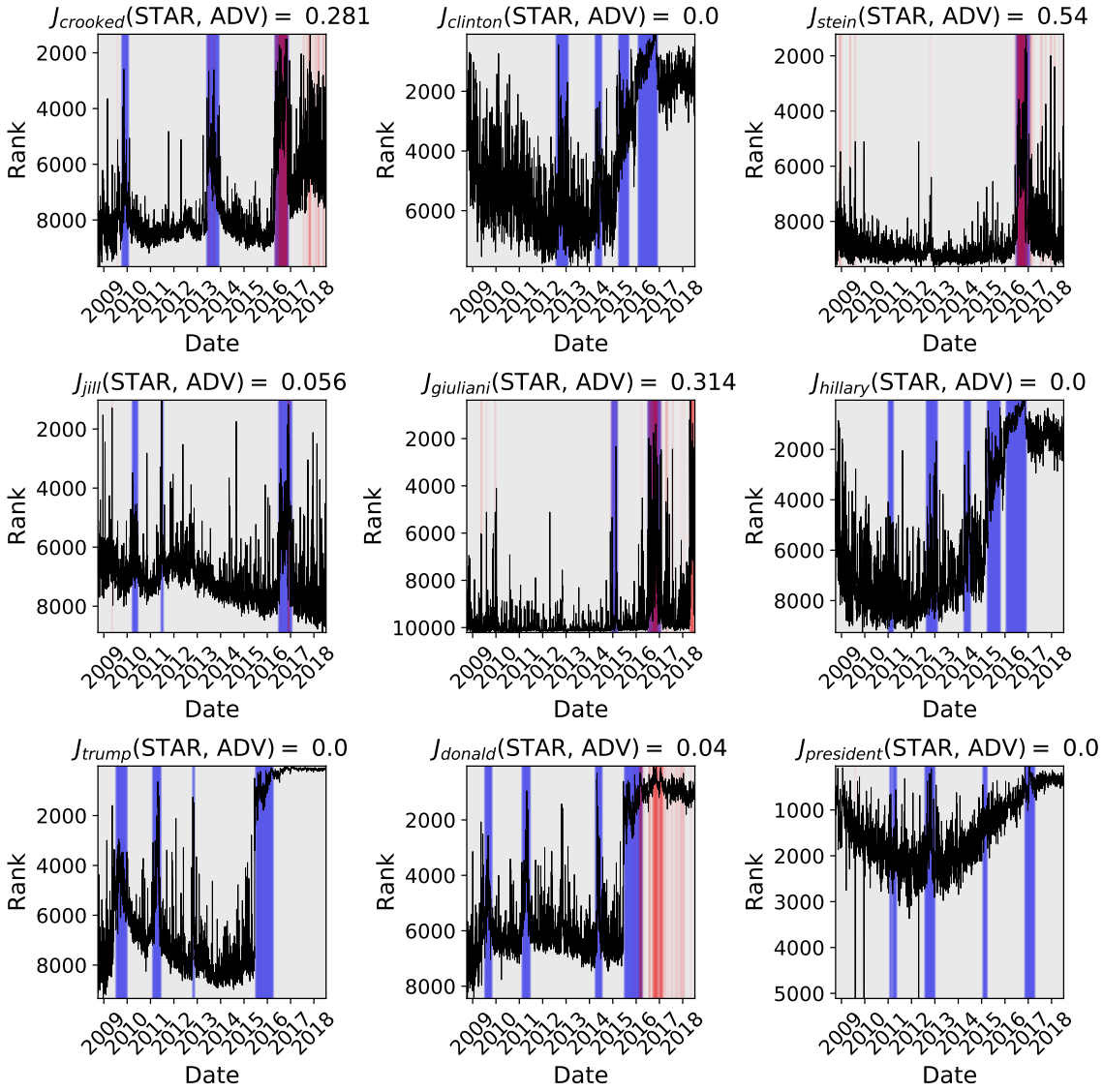

Figure 3.9: Comparison of STAR and Twitter's Anomaly Detection Vector (ADV) algorithm used for detecting phenomena in Twitter 1gram time series. The Jaccard similarity coefficient is presented for each 1-gram and the region where events on detected are shaded for the respective algorithm. Blue-shaded windows correspond with STAR windows of shock-like behavior, while red-shaded windows correspond with ADV windows of anomalous behavior (and hence purple windows correspond to overlap between the two). In general, ADV is most effective at detecting brief spikes or strong shock-like signals, whereas STAR is more sensitive to longer-term shocks and shocks that occur in the presence of surrounding noisy or nonstationary dynamic. ADV does not treat strong periodic fluctuations as anomalous by design; though this may or may not be a desirable feature of a similarity search or anomaly detection algorithm, it is certainly not a flaw in ADV but simply another differentiator between ADV and STAR.

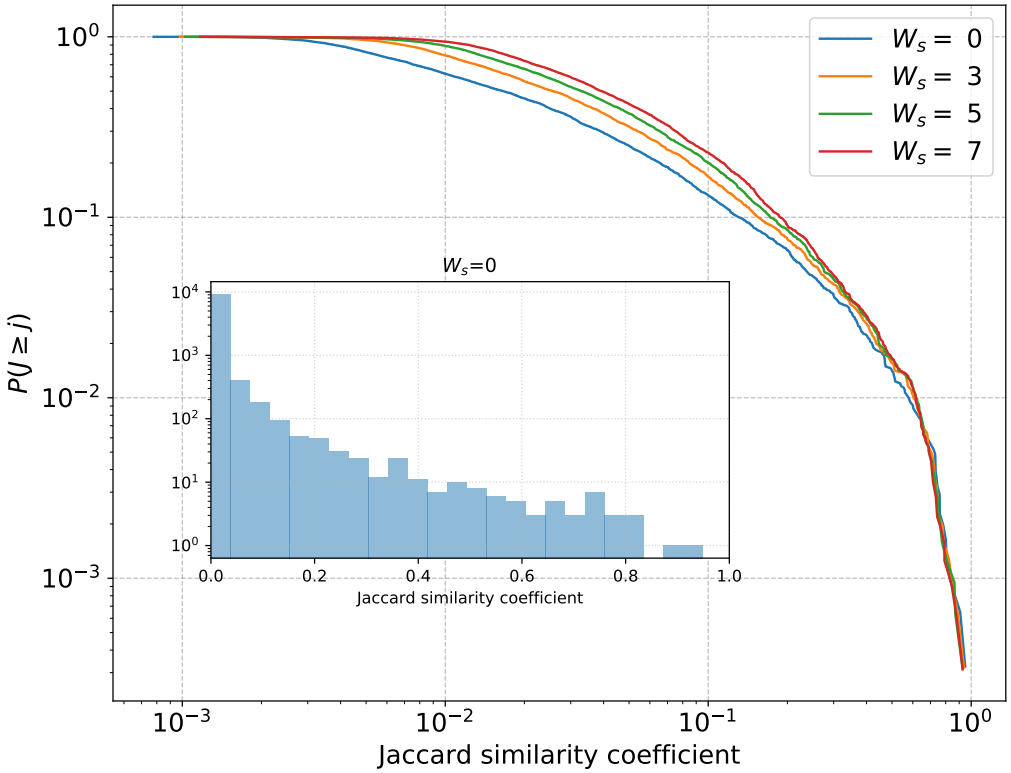

Figure 3.10: Complimentary cumulative distribution function (CCDF) of Jaccard similarity coefficients for regions that Twitter's ADV and our STAR algorithm detect patterns or anomalies (see Fig. 3.9). Window sizes are varied to include $W_s \in \{0, 3, 5, 7\}$ (i.e. detections within $t_i \pm W_s$ are as part of the intersection). Time series with $J_{\text{word}_i} = 0$ are omitted from the CCDF. The inset histogram shows the distribution of Jaccard similarity coefficients for $W_s = 0$ (i.e. exact matches), $J = 0$ time series are included.

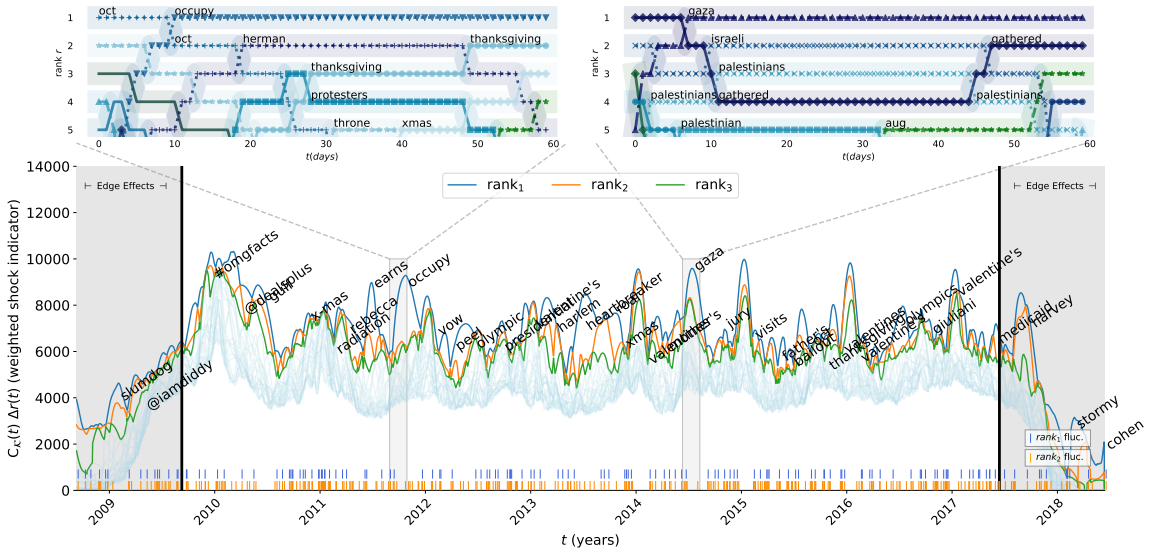

Figure 3.11: Time series of the ranked and weighted shock indicator function. At each time step t, the weighted spike indicator functions (WSIF) are sorted so that the word with the highest WSIF corresponds to the top time series, the words with the second-highest WSIF corresponds to the second time series, and so on. Vertical ticks along the bottom mark fluctuations in the word occupying ranks 1 and 2 of WSIF values. Top panels present the ranks of WSIF values for words in the top 5 WSIF values in a given time step for the sub-sampled period of 60 days. An interactive version of this graphic is available at the authors' webpage: http://compstorylab.org/shocklets/ranked_shock_weighted_interactive.html.

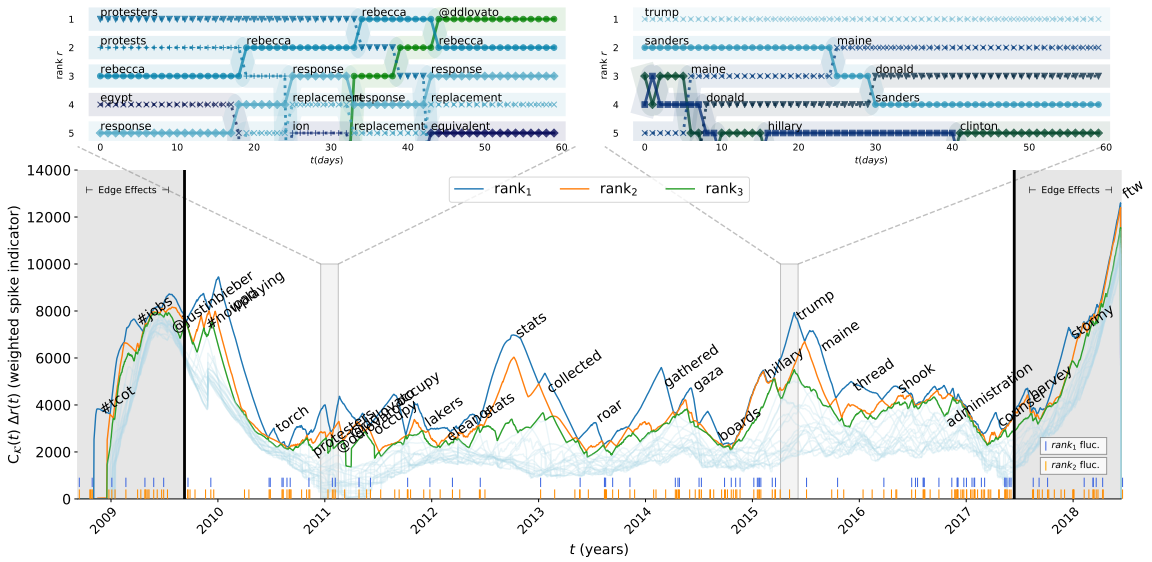

Figure 3.12: Time series of the ranked and weighted spike indicator function. At each time step t, the weighted spike indicator functions (WSpIF) are sorted so that the word with the highest WSpIF corresponds to the top time series, the words with the second-highest WSpIF corresponds to the second time series, and so on. Vertical ticks along the bottom mark fluctuations in the word occupying ranks 1 and 2 of WSpIF values. Top panels present the ranks of WSpIF values for words in the top 5 WSpIF values in a given time step for the sub-sampled period of 60 days. The top left panel, demonstrates the competition for social attention between geopolitical concerns,Äîstreet protests in Egypt–and popular artists and popular culture influence–Rebecca Black and Demi Lovato. The top right panel displays the language surrounding the 2016 U.S. presidential election immediately after Donald Trump announced his candidacy. An interactive version of this graphic is available at the authors' webpage: http://compstorylab.org/shocklets/ranked_spike_weighted_interactive.html.

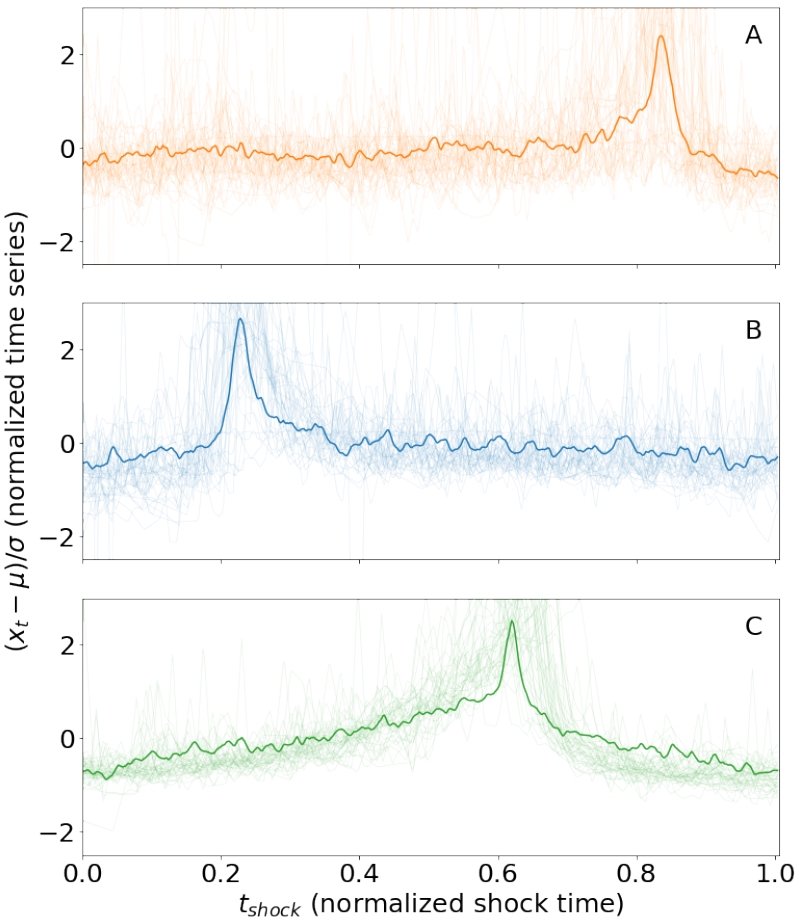

Figure 3.13: Extracted shock segments show diverse behavior corresponding to divergent social dynamics. We extract "important" shock segments (those that breach the top $k = 20$ ranked weighted shock indicator at least once during the decade under study) and normalize them as described in Section 3.3. We then find the densities of shock points t_1^*, measured using the maxima of the within-window time series, and alternatively measured using the maxima of the (relative) shock indicator function. We calculate relative maxima of these distributions and spatially-average shock segments whose maxima were closest to these relative maxima; we display these mean shock segments along with sample shock segments that are close to these mean shock segments in norm. We introduce a classification scheme for shock dynamics: Type I (panel A) dynamics are those that display slow buildup and fast relaxation; Type II (panel B) dynamics, conversely, display fast (shock-like) buildup and slow relaxation; and Type III (panel C) dynamics are relatively symmetric. Overall, we find that Type III dynamics are most common (40.9%) among words that breach the top $k = 20$ ranked weighted shock indicator function, while Type II are second-most common (36.4%), followed by Type I (22.7%).

Chapter 4

Non-cooperative dynamics in election interference

4.1 Introduction

In democratic and nominally-democratic countries, elections are societally and politically crucial events in which power is allocated [189]. In fully-democratic countries elections are the method of legitimate governmental change [190]. One country, whom we will label "Red", may wish to influence or appear to influence the outcome of an election in another country, whom we will label "Blue", because of the importance or perceived importance of elections in Blue with respect to Red's interests. Such attacks on democracies are not new; it is estimated that the United States (U.S.) and Russia or its predecessor, the Soviet Union, often interfere in the elections of other nations and have consistently done this since 1946 [191]. Though academic study of this area has increased [192], we are unaware of much formal modeling of noncooperative dynamics in an election interference game. Recent approaches to the study of this phenomena have focused mainly on the compilation of coarse-grained (e.g., yearly frequency) panels of election interference events and qualitative analysis of this data [193, 194], and data-driven studies of the aftereffects and second-order effects of interference operations [195, 196]. Attempts to create theoretical models of interference operations are more sparse and include qualitative causal models of cyberoperation influence on voter preferences [197] and models of the underlying reasons that a state may wish to interfere in the elections of another [198].

Here, we consider a Red - Blue two-player game in which Red wishes to influence a two-candidate, zero-sum election taking place in Blue's country, as outlined above. In this context, we think of Red and Blue as the respective foreign (Red) and domestic (Blue) intelligence services of the two countries. Red wants a particular candidate

(candidate A) to win the election, while Blue wants the effect of Red's interference to be minimized. We characterize this problem theoretically, deriving a noncooperative, non-zero-sum differential game, and then explore the game numerically. We find that all-or-nothing attitudes by either Red or Blue can lead to arms-race conditions in interference operations. In the event that one party credibly commits to playing a particular strategy, we derive further analytical results.

Turning to a recent instance of election interference, we confront our model with the 2016 U.S. presidential election in which Russia conducted interference operations [199]. After fitting a Bayesian structural time series model to election polls and social media posts associated with Russian Internet Research Agency Twitter troll accounts, we show that our model, though simple, is able to adequately capture many of the observed and inferred parameters' dynamics. We close by proposing some theoretical and empirical extensions to our work.

4.2 Theory

4.2.1 Election interference model

We consider the case of a simple election between two candidates in a homogeneous environment (e.g., no institutions such as an Electoral College) so that the election process at any time $t \in [0, T]$, a noisy representation of which is a public poll, can be represented by a scalar $Z_t \in [0, 1]$. The model is set in continuous time here; when we estimate parameters statistically in Sec. 4.3 we move to a discrete-time analogue. We hypothesize that the election dynamics take place in a latent space, where dynamics

are represented by $X_t \in \mathbb{R}$. Without loss of generality, we will set $x < 0$ to be values of the latent poll that favor candidate A and $x > 0$ that favor candidate B. The latent and observable space are related by $Z_t = \phi(X_t)$, where ϕ is a sigmoidal function which we choose to be $\phi(x) = \frac{1}{1+e^{-x}}$. (This choice is arbitrary; any sigmoidal function that is bounded between zero and one will suffice, leading only to different parameter estimates in the context of statistical estimation.) The actual result of the election is given by $\phi(X_T)$, by which we mean the number of votes that are earned by candidate B is $\phi(X_T)$. The election takes place in a population of N voting agents, each of whom updates their preferences over the candidates in the latent space at each time step t_n by a small random variable ξ_{n,t_n}, each of which satisfies $E_n\left[\xi_{n,t_k}\right] = 0$ for all t. The election process's increments are the sample mean of the realizations of the voting agents' preferences at time t. In the absence of interference, the stochastic election model is thus very simple—an unbiased random walk, which we write as

$$X_{t_{k+1}} = X_{t_k} + \frac{1}{N} \sum_{1 \leq n \leq N} \xi_{n,t_k} \Delta t, \tag{4.1}$$

where $\Delta t = t_{k+1} - t_k$. We display sample realizations of this process for different distributions of ξ_{n,t_k} in Fig. 4.1. Though one distribution of ξ_{n,t_k} describes the process of hardening of political preferences and another characterizes a system in which voting agents usually have fluctuating political preferences, the sample paths of X_{t_k} are statistically similar since $\frac{1}{N}\sum_n \xi_{n,t_k}$ does not vary much between the distributions. When N is large we can reasonably approximate this process by the Wiener process, $\mathrm{d}X_t = \sigma \mathrm{d}\mathcal{W}_t$, where $\sigma^2 \approx \mathrm{Var}\left(\frac{1}{N}\sum_{1 \leq n \leq N} \xi_{n,t}\right)$, which is valid in the limit of large N. If the preference change random variables $\xi_{n,k}$ did not satisfy $E_n[\xi_{n,k}] = 0$, this would not necessarily be true. For example, if $\{\xi_{n,k}\}_{k \geq 0}$ were a random walk or were

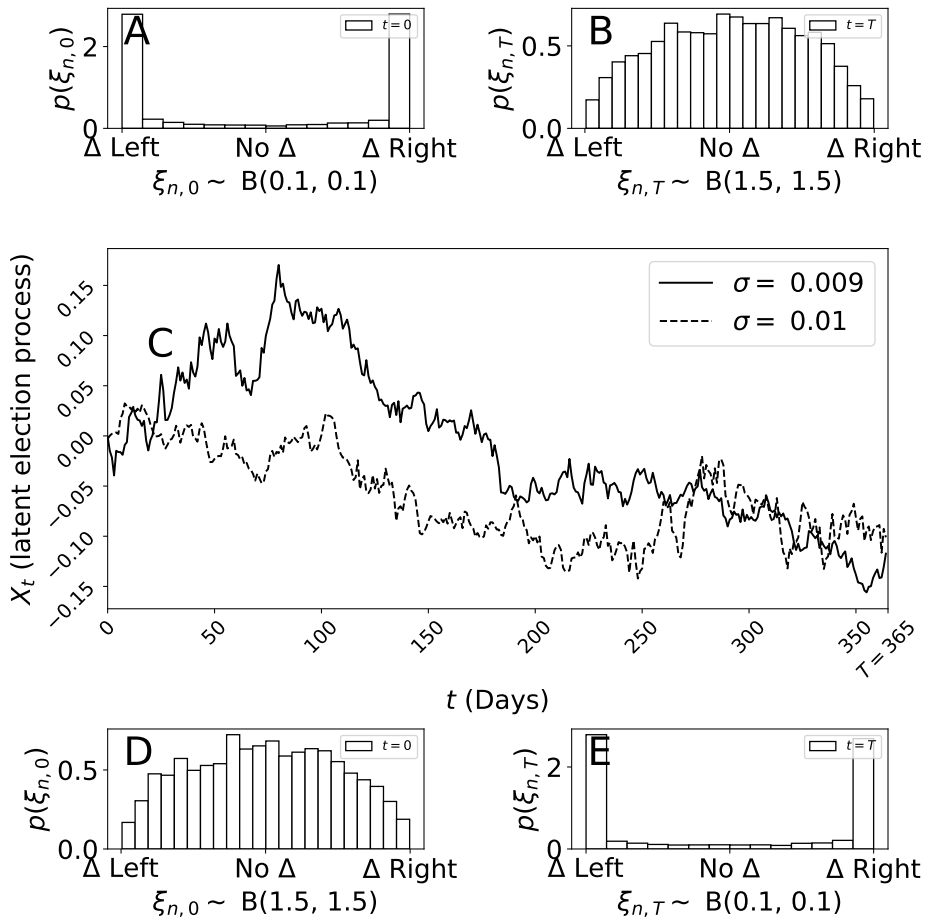

Figure 4.1: Though simple, the random walk latent space election model is an approximation to varied population candidate preference updates. The latent election process evolves according to $X_{k+1} = X_k + \frac{1}{N}\sum_{1 \leq n \leq N} \xi_{n,k}$*, where* $\xi_{n,k}$ *is voting agent* n*'s shift toward the left (*< 0*) or right (*> 0*) of the political spectrum at time* k*. In the center panel, the solid curve is a draw from the latent election process resulting from the preference updates* $\xi_{n,t} \sim B\left(0.1\frac{T-t}{T} + 1.5\frac{t}{T}, 0.1\frac{T-t}{T} + 1.5\frac{t}{T}\right)$*, where* $B(\alpha, \beta)$ *is the Beta distribution and we have set* $T = 365$*. This change in political preference shift distribution describes an electorate with increasing resistance to change in their political viewpoints. We display the preference shift distributions at* $t = 0$ *(*$t = T$*) in Panel A (Panel B). For contrast, the dashed curve is a draw from the latent election process resulting from* $\xi_{n,t} \sim B\left(1.5\frac{T-k}{T} + 0.1\frac{k}{T}, 1.5\frac{T-k}{T} + 0.1\frac{k}{T}\right)$*, which describes an electorate in which the component agents often have changing political preferences. We show the corresponding preference shift distributions at* $t = 0$ *(*$t = T$*) in Panel D (Panel E). Despite these preference updates that are, in some sense, opposites of each other, the latent processes* X_t *are statistically very similar and are both well-modeled by the continuum approximation* $\mathrm{d}X_t = \sigma \mathrm{d}\mathcal{W}_t$*.*

trend-stationary for each n, then $\{E_n[\xi_{n,k}]\}_{k\geq 0}$ would also respectively be a random walk or trend-stationary and hence its sum would not be well-described by a random walk model. A unit root or trend-stationary $\xi_{n,k}$ would model a population in which political preferences were undergoing a shift in population-wide distribution rather than just in individual preferences. However, even if $E_n[\xi_{n,k}] \neq 0$ for each n, it is also not necessarily the case that a version of the random walk model is *not* a valid approximation for the electoral process. If the stochastic evolution equation for $E_n[\xi_{n,k}]$ has as its solution a stationary colored noise with exponentially-decaying covariance function, then the sum (integral) of this noise will satisfy a Stratonovich-type equation [200, 201, 202] that may be a suitably generalized (in terms of the covariance structure) version of the basic random walk model considered here.

We denote the control policies of Red and Blue— the functions by which Red and Blue attempt to influence (or prevent influence on) the election—by $u_{\mathrm{R}}(t)$ and $u_{\mathrm{B}}(t)$. These functions are one-dimensional continuous-time stochastic processes (time series); the term "policy" originates from the fields of economics and reinforcment learning. These control policies are abstract variables in the context of our model but can be interpreted as monetary expenditures on interference operations. We will assume that Red and Blue can affect the mean trajectory of the election but not its volatility (variance of its increments). We make this assumption because X_t is an approximation to the process described by Eq. 4.1 and, as displayed in Fig. 4.1 and described above, this process's statistical characteristics do not change much even when the voting population's underlying preference change distributions are significantly different. Thus, under the influence of Red's and Blue's control policies, the

election dynamics become

$$\mathrm{d}X_t = f(u_{\mathrm{R}}(t), u_{\mathrm{B}}(t))\mathrm{d}t + \sigma \mathrm{d}\mathcal{W}_t,\ X_0 = y. \tag{4.2}$$

To first order expansion we have $f(u_{\mathrm{R}}(t), u_{\mathrm{B}}(t)) = a_0 + a_{\mathrm{R}}u_{\mathrm{R}}(t) + a_{\mathrm{B}}u_{\mathrm{B}}(t) + \mathcal{O}(u^2)$ which is most accurate near $u = 0$, so we approximate the state equation by

$$\mathrm{d}X_t = [u_{\mathrm{R}}(t) + u_{\mathrm{B}}(t)]\mathrm{d}t + \sigma \mathrm{d}\mathcal{W}_t,\ X_0 = y, \tag{4.3}$$

since we have assumed *a priori* zero drift and can absorb constants into the definition of the control policies. We will use Eq. 4.3 as the state equation for the remainder of the paper.

4.2.2 Subgame-perfect Nash equilibria

Red and Blue each seek to minimize separate scalar cost functionals of their own control policy and the other agent's control policy; for now, we will assume that the agents do not incur a running cost from the value of the state variable. The cost functionals can thus be written

$$E_{u_{\mathrm{R}},u_{\mathrm{B}},X}\left\{\Phi_{\mathrm{R}}(X_T) + \int_0^T C_{\mathrm{R}}(u_{\mathrm{R}}(t), u_{\mathrm{B}}(t))\ \mathrm{d}t\right\}, \tag{4.4}$$

and

$$E_{u_{\mathrm{R}},u_{\mathrm{B}},X}\left\{\Phi_{\mathrm{B}}(X_T) + \int_0^T C_{\mathrm{B}}(u_{\mathrm{R}}(t), u_{\mathrm{B}}(t))\ \mathrm{d}t\right\}. \tag{4.5}$$

The functions C_{R} and C_{B} represent the running cost or benefit of conducting election interference operations. We assume the cost functions have the form

$$C_i(u_{\mathrm{R}}, u_{\mathrm{B}}) = u_i^2 - \lambda_i u_{\neg i}^2 \tag{4.6}$$

for $i \in \{\mathrm{R}, \mathrm{B}\}$. (The notation $\neg i$ indicates the other player—for example, if $i = \mathrm{R}$, $\neg i = \mathrm{B}$—and originates in the study of noncooperative economic games.) The non-negative scalar λ_i parameterizes the utility gained by player i from observing player $\neg i$'s effort; if λ_i is high, player i gains utility from player $\neg i$'s expending resources, while if $\lambda_i = 0$, player i has no regard for $\neg i$'s level of effort but only for their own running cost and the final cost. The assumption of quadratic control is common in optimal control theory as it can be justified as a Taylor approximation to an arbitrary even cost function. If we write an arbitrary analytic cost function for player i as $\mathcal{C}_i(u_{\mathrm{R}}, u_{\mathrm{B}}) = \mathcal{C}^{(i)}(u_i) - \lambda_i \mathcal{C}^{(\neg i)}(u_{\neg i})$ and make the assumptions that it is equally costly to conduct operations that favor candidate A or candidate B (hence imposing that $\mathcal{C}^{(i)}$ and $\mathcal{C}^{(\neg i)}$ are even) and that player i conducting no interference operations results in player i's incurring no direct cost from this choice, then the first non-zero term in the Taylor expansion of $\mathcal{C}_i$ is given by Eq. 4.6.

Though the running cost functions are equivalent across players in form, the final conditions differ between Red and Blue because of their qualitatively distinct objectives. Since Red wants to influence the outcome of the election in Blue's country in favor of candidate A, their final cost function Φ_{R} must satisfy $\Phi_{\mathrm{R}}(x) < \Phi_{\mathrm{R}}(y)$ for all $x < 0$ and $y > 0$; in our example final conditions presented here we also assume that Φ_{R} is monotonically non-decreasing everywhere, but we relax this assumption in Sec.

4.3 in which we confront this model with election interference-related data; to the extent that this model describes reality, it is probably not true that these restrictive assumptions on the final condition are always satisfied by all Red teams conducting election interference operations. One possible final condition that satisfies these requirements is $\Phi_{\mathrm{R}}(x) = c_0 + c_1 x$, but this allows the somewhat unrealistic limiting condition of infinite benefit (cost) if candidate A gets 100% (0%) of the vote in the election. We will thus also consider two Red final conditions with bounded extremal cost: one smooth, $\Phi_{\mathrm{R}}(x) = \tanh(x)$; and one discontinuous, $\Phi_{\mathrm{R}}(x) = \Theta(x) - \Theta(-x)$. By $\Theta(\cdot)$ we mean the Heaviside step function.

Blue is attempting to ameliorate the effects of Red's control policy—reduce the overall impact of Red's interference operations on the electoral process—hence the form of the state dynamics presented in Eq. 4.3. Since Blue is *a priori* indifferent between the outcomes of the election, at first glance it appears that the final condition $\Phi_{\mathrm{B}}(x) = 0$ is a reasonable modeling choice. However, for the case $\lambda_{\mathrm{B}} = 0$, this results in Blue taking no action at all in the game due to the functional form of Eq. 4.6. In other words, if Blue does not gain utility from Red expending resources, then Blue will not try to stop red from interfering in an election in Blue's country! Hence it appears likely that Blue must actually have nontrivial preferences over the election outcome.

We present three possible alternatives for a cost function representing Blue's preferences; as in the case of Red's final condition, this list is entirely non-exhaustive. Blue may simply be suspicious that a result was due to Red's interference if X_T is too far from $E_0[X_T] = 0$. An example of a smooth function that represents these preferences over the election outcome is $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. However, this neglects the reality that Red's objective is not to have either candidate A or candidate B win by a large

margin, but rather to have candidate A win (i.e., have $X_T < 0$). Thus Blue might be unconcerned about larger positive values of the state variable and, modifying the previous function suitably, have $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Alternatively, Blue may accept the result of the election as long as it does not stray "too far" from the initial expected value. An example of a discontinuous final condition that can represent these preferences is $\Phi_{\mathrm{B}}(x) = \Theta(|x| > \Delta) - \Theta(|x| \leq \Delta)$, where $\Delta > 0$ is Blue's accepted margin of error.

A nondenumerable panolopy of other final conditions can be hypothesized, but the example functions that we have presented here give some qualitative sense of the range of possible payoff structures. We include:

- "First-order" functions that could result from the Taylor expansion about zero of an arbitrary analytic final condition—linear, in the case of Red's antisymmetric final condition, and quadratic in the case of Blue's smooth symmetric final condition (which is the first non-constant term in the Taylor expansion of an even analytic function);

- Smooth functions that represent preferences over the result of the electoral process that we deem marginally more realistic, such as bounded benefit / cost and the recognition that Red (by assumption) favors one candidate in particular; and

- Discontinuous final conditions that model "all-or-nothing" preferences over the outcome (either candidate A wins or they do not; either Red interferes less than a certain amount or they interfere more).

These functions do not capture some interesting behavior that might exist in real

election interference operations. For example, Red's preferences concerning the result of the election outccome might be as follows: "we would prefer that candidate A wins the election, but if they cannot, then we would like candidate B to win by a landslide so that we can claim the electoral system in Blue's country was rigged against candidate A". These preferences correspond to a final condition with a global minimum at some $x < 0$ but a secondary local minimum at $x \gg 0$; this situation is clearly not modeled by any of the final conditions given above. In Sec. 4.3 we will drop the assumption that the final conditions are parameterized according to any of the functional forms considered in thise section and instead infer them from observed election and election interference proxy data.

The application of the dynamic programming principle [203, 204] to Eqs. 4.3, 4.4, and 4.5 leads to a system of coupled Hamilton-Jacobi-Bellman equations for the value functions of Red and Blue,

$$-\frac{\partial V_{\mathrm{R}}}{\partial t} = \min_{u_{\mathrm{R}}} \left\{ \frac{\partial V_{\mathrm{R}}}{\partial x}[u_{\mathrm{R}} + u_{\mathrm{B}}] + u_{\mathrm{R}}^2 - \lambda_{\mathrm{R}} u_{\mathrm{B}}^2 + \frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{R}}}{\partial x^2} \right\}, \tag{4.7}$$

and

$$-\frac{\partial V_{\mathrm{B}}}{\partial t} = \min_{u_{\mathrm{B}}} \left\{ \frac{\partial V_{\mathrm{B}}}{\partial x}[u_{\mathrm{R}} + u_{\mathrm{B}}] + u_{\mathrm{B}}^2 - \lambda_{\mathrm{B}} u_{\mathrm{R}}^2 + \frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{B}}}{\partial x^2} \right\}. \tag{4.8}$$

The dynamic programming principle does not result in an Isaacs equation because the game is not zero-sum and the cost functionals for Red and Blue can have different functional forms. (The Isaacs equation is a nonlinear elliptic or parabolic equation that arises in the study of two-player, zero-sum games in which one player attempts to maximize a functional and the other player attempts to minimize it [205, 206].) Performing the minimization with respect to the control variables gives the Nash

equilibrium control policies,

$$u_{\mathrm{R}}(t) = -\frac{1}{2}\frac{\partial V_{\mathrm{R}}}{\partial x}\bigg|_{(t,X_t)} \tag{4.9}$$

$$u_{\mathrm{B}}(t) = -\frac{1}{2}\frac{\partial V_{\mathrm{B}}}{\partial x}\bigg|_{(t,X_t)}, \tag{4.10}$$

and the exact functional form of Eqs. 4.7 and 4.8,

$$-\frac{\partial V_{\mathrm{R}}}{\partial t} = -\frac{1}{4}\left(\frac{\partial V_{\mathrm{R}}}{\partial x}\right)^2 - \frac{1}{2}\frac{\partial V_{\mathrm{R}}}{\partial x}\frac{\partial V_{\mathrm{B}}}{\partial x} - \frac{\lambda_{\mathrm{R}}}{4}\left(\frac{\partial V_{\mathrm{B}}}{\partial x}\right)^2 + \frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{R}}}{\partial x^2}, \qquad V_{\mathrm{R}}(x,T) = \Phi_{\mathrm{R}}(x); \tag{4.11}$$

$$-\frac{\partial V_{\mathrm{B}}}{\partial t} = -\frac{1}{4}\left(\frac{\partial V_{\mathrm{B}}}{\partial x}\right)^2 - \frac{1}{2}\frac{\partial V_{\mathrm{B}}}{\partial x}\frac{\partial V_{\mathrm{R}}}{\partial x} - \frac{\lambda_{\mathrm{B}}}{4}\left(\frac{\partial V_{\mathrm{R}}}{\partial x}\right)^2 + \frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{B}}}{\partial x^2}, \qquad V_{\mathrm{B}}(x,T) = \Phi_{\mathrm{B}}(x). \tag{4.12}$$

When solved over the entirety of state space, solutions to Eqs. 4.11 and 4.12 constitute the strategies of a subgame-perfect Nash equilibrium because, no matter the action taken by player $\neg i$ at time t, player i is able to respond with the optimal action at time $t + \mathrm{d}t$. Given the solution pair $V_{\mathrm{R}}(x,t)$ and $V_{\mathrm{B}}(x,t)$, the distribution of $Z = (x, u_{\mathrm{R}}, u_{\mathrm{B}})^T$ can be written analytically. Substitution of Eqs. 4.9 and 4.10 into Eq. 4.3 gives $\mathrm{d}x = -\frac{1}{2}\left\{\frac{\partial V_{\mathrm{R}}}{\partial x}|_{(t,x)} + \frac{\partial V_{\mathrm{B}}}{\partial x}|_{(t,x)}\right\}\mathrm{d}t + \sigma\mathrm{d}\mathcal{W}$. We discretize the state equation to obtain

$$\begin{aligned} x_{n+1} - x_n + \frac{\Delta t}{2}\left[V'_{\mathrm{R}n} + V'_{\mathrm{B}n}\right] \\ - (\Delta t)^{1/2}\sigma w_n - y\delta_{n,0} = 0, \end{aligned} \tag{4.13}$$

with $w_n \sim \mathcal{N}(0,1)$ and where we have put $V'_{in} \equiv V'_i(x_n, t_n)$. Thus the distribution of

an increment of the latent electoral process is

$$p(x_{n+1}|x_n) = \frac{1}{\sqrt{2\pi\sigma^2\Delta t}} e^{-\frac{\Delta t}{2\sigma^2}(\frac{x_{n+1}-x_n}{\Delta t} + \frac{1}{2}[V'_{\mathrm{R}n}+V'_{\mathrm{B}n}] - y\frac{\delta_{n0}}{\Delta t})^2}. \tag{4.14}$$

Now, using the Markov property of X_t, we have

$$p(x_1, ..., x_N|x_0) = \prod_{n=0}^{N-1} p(x_{n+1}|x_n) \tag{4.15}$$

$$= \frac{1}{(2\pi\sigma^2\Delta t)^{N/2}} \exp\left\{-S(x_1, ..., x_N)\right\}, \tag{4.16}$$

where

$$\begin{aligned} S(x_1, ..., x_N) = \frac{1}{2\sigma^2} \sum_{n=0}^{N-1} \Delta t \Big[& \frac{x_{n+1}-x_n}{\Delta t} \\ & + \frac{1}{2}[V'_{\mathrm{R}n} + V'_{\mathrm{B}n}] - \frac{y\delta_{n,0}}{\Delta t} \Big]^2. \end{aligned} \tag{4.17}$$

Taking $N \to \infty$ as $N\Delta t = T$ remains constant gives a standard Gaussian path integral with an action $S(x(t))$ that incorporates the derivatives of the value functions. Since u_{R} and u_{B} are just time-dependent functions of $x(t)$, their distributions can also be found explicitly using the probability distribution Eq. 4.16 and the appropriate (time-dependent) Jacobian transformation. Unfortunately, these analytical results are of limited utility because we are unaware of analytical solutions to the system given in Eqs. 4.11 and 4.12, and hence $V'_{\mathrm{R}}(x,t)$ and $V'_{\mathrm{B}}(x,t)$ must be approximated. However, we will have something to say about analytical solutions presently in the case that player i announces a credible commitment to a particular control path.

In the general case presented above, we find the value functions $V_{\mathrm{R}}(x,t)$ and $V_{\mathrm{B}}(x,t)$ numerically through backward iteration, enforcing a Neumann boundary condition at $x = \pm 3$, which corresponds to bounding polling popularity of candidate B from below

by 4.7% and from above by 95.3% [1]. Fig. 4.2 displays example realizations of the value functions for different λ_i and final conditions. The value functions display diffusive behavior in common due to the game's stochasticity, but also differ qualitatively depending on the effect of the final condition propagating backward in time. When the final conditions are discontinuous, as is the case in the top panels of Fig. 4.2, the derivatives of the value function assume greater magnitudes and vary more rapidly throughout the game than do the derivatives of the value function when the final conditions are continuous; this is a typical feature of solutions to equations of HJB-type [207] and has consequences for the game-theoretic interpretation of these results, as we discuss below. Fig. 4.2 also demonstrates that the extrema of the value functions are not as large in magnitude when $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 0$ as when $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 2$; this is because higher values of λ_i mean that player i derives utility not only from the final outcome of the game but also from causing player $\neg i$ to expend resources in the game.

Eqs. 4.7 and 4.8 give the closed-loop control policies u_{R} and u_{B} respectively given the current state X_t and time t. We display samples of u_{R}, u_{B}, and the electoral process Z_t in Fig. 4.3 to illuminate some of the qualitative properties of this game before considering a more comprehensive sweep over parameters. We simulate the game with parameters $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 2$, $\Phi_{\mathrm{R}}(x) = x$, and $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. We plot the control policies in the top panel including the mean control policies $E[u_{\mathrm{R}}]$ and $E[u_{\mathrm{B}}]$, displayed in thicker curves. For this parameter set, it is optimal for Red to begin play with a relatively large amount of interference and, in the mean, decrease the level of interference over time. Conversely, throughout the game Blue increases their resistance to Red's interference. Despite this resistance, the bottom panel reveals

[1] Code to recreate simulations and plots in this paper—or to create new simulations and "what-if" scenarios—is located at https://gitlab.com/daviddewhurst/red-blue-game.

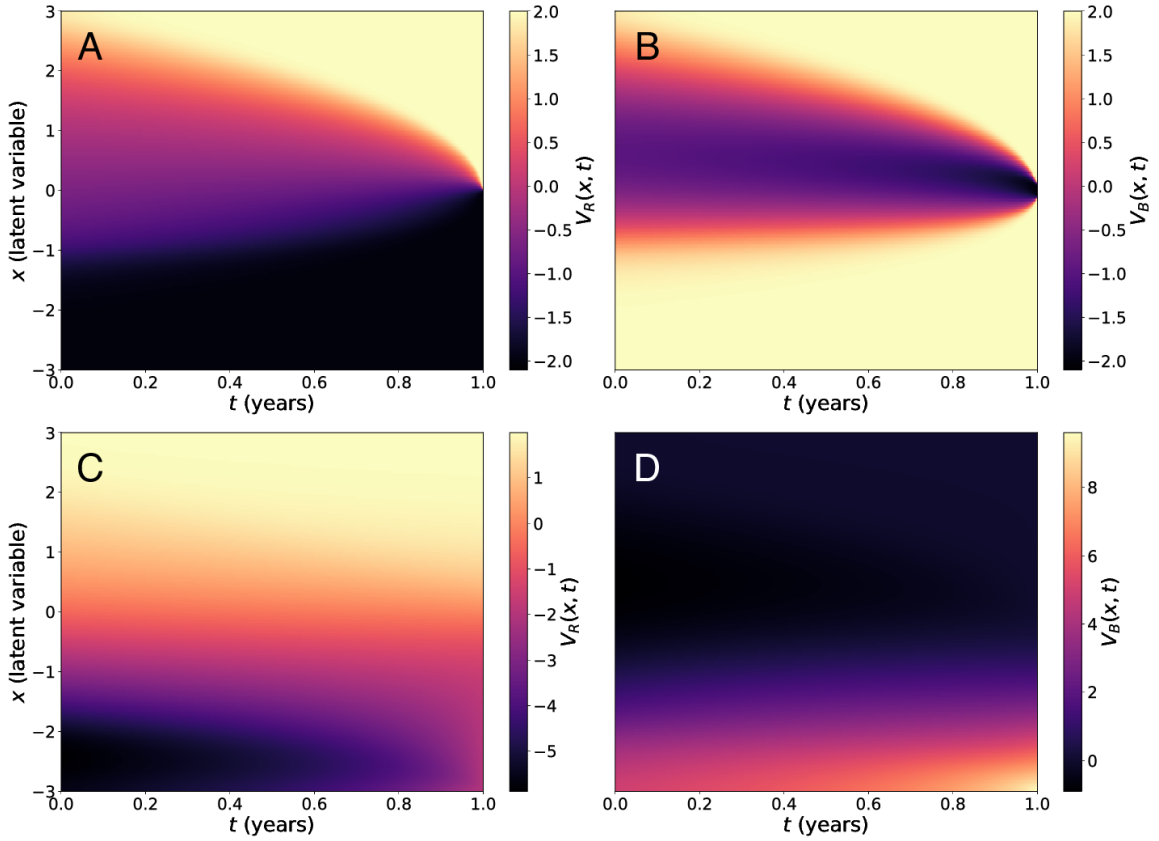

Figure 4.2: Example value functions corresponding to the system Eqs. 4.11 and 4.12. Panels A and B display $V_{\mathrm{R}}(x,t)$ *and* $V_{\mathrm{B}}(x,t)$ *respectively for* $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 0$, $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$, *and* $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| > 0.1) - \Theta(|x| \leq 0.1)]$ *with* $\Delta = 0.1$, *while panels C and D display* $V_{\mathrm{R}}(x,t)$ *and* $V_{\mathrm{B}}(x,t)$ *respectively for* $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 2$, $\Phi_{\mathrm{R}}(x) = 2\tanh(x)$, *and* $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. *For each solution we enforce Neumann no-flux boundary conditions and set* $\sigma = 0.6$. *The solution is computed on a grid with* $x \in [-3, 3]$, *setting* $\mathrm{d}x = 0.025$, *and integrating for* $N_t = 8000$ *timesteps.*

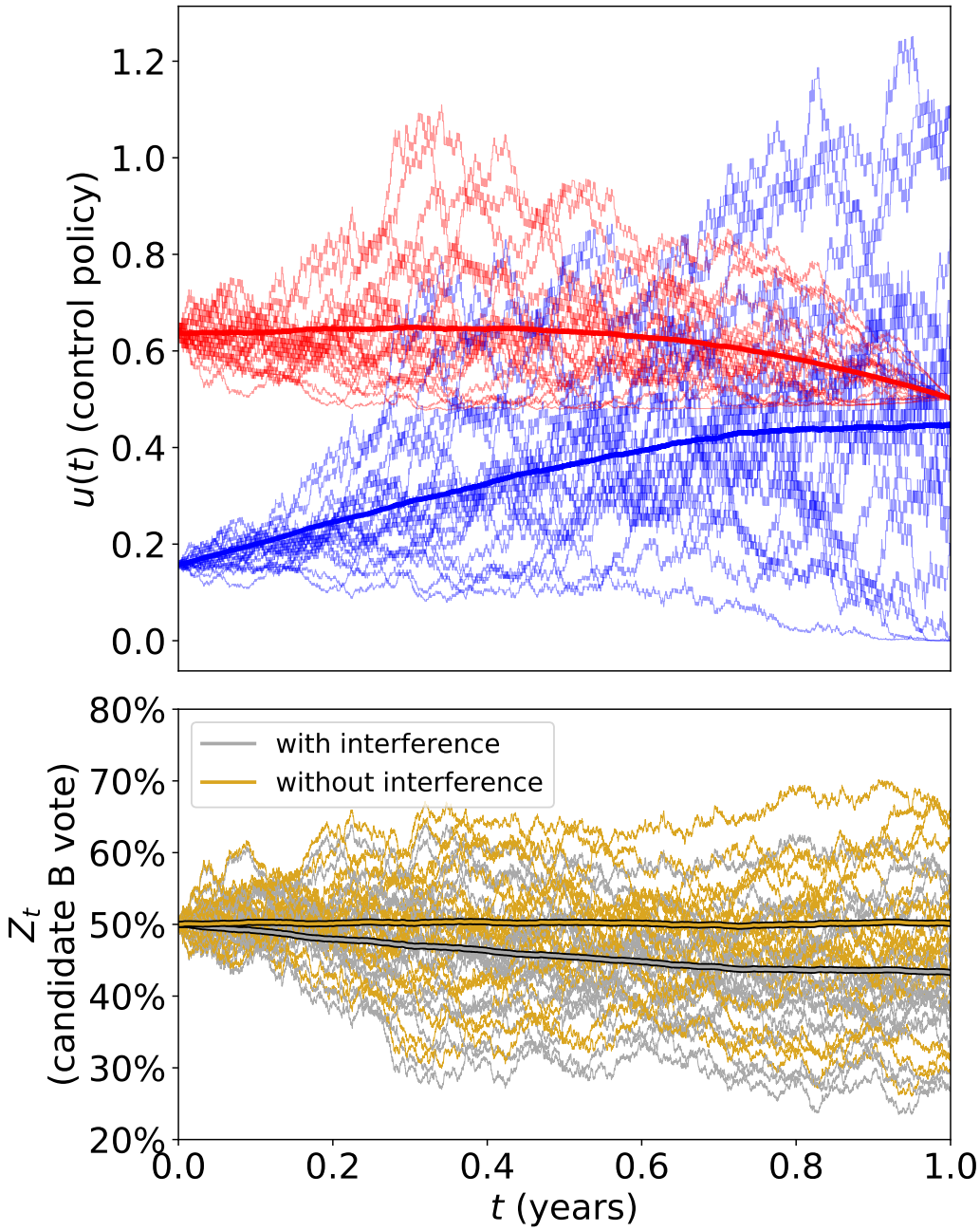

Figure 4.3: We display realizations of u_{R} and u_{B} in the top panel and paths of the electoral process in the bottom panel. We draw these realizations from the game simulated with parameters $\lambda_{\mathrm{R}} = \lambda_{\mathrm{B}} = 2$, $\Phi_{\mathrm{R}}(x) = x$, and $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. For this parameter set, Blue is fighting a losing battle—the bottom panel clearly shows that, even with Blue attempting to stop Red from interfering in the game, optimal play by both players results in a significantly lower $E[Z_t]$ than for the electoral process without any interference.

that, for this parameter set, Red is able to accomplish their objective of causing candidate A to win: in the mean case, candidate A enjoys a comfortable lead in the election poll by the final time.

To gain a better idea of the qualitative nature of this game for a more varied set of parameters, we conducted a coarse parameter sweep over λ_{R}, λ_{B}, Φ_{R}, and Φ_{B}. Figs. 4.4 and 4.5 displays the results of this parameter sweep for two combinations of final conditions; holding Blue's final condition of $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$ constant, we compare the means and standard deviations of the Nash equilibrium strategies $u_{\mathrm{R}}(t)$ and $u_{\mathrm{B}}(t)$ across values of the coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}} \in [0, 3]$ as Red's final condition changes from $\Phi_{\mathrm{R}}(x) = \tanh(x)$ to $\Phi_{\mathrm{R}}(x) = \Theta(x) - \Theta(-x)$. For these combinations of final conditions, higher values of the coupling parameters λ_i cause greater fluctuation in control policies. This increase in fluctuation is more pronounced when Red's final condition is discontinuous, which is sensible since in this case $\lim_{t\to T^-} u_{\mathrm{R}}(x,t) = -\frac{1}{2}\delta(x)$. Appendix **??** contains similar figures for each $3^2 = 9$ combinations of Red example final conditions, $\Phi_{\mathrm{R}}(x) \in \{\tanh(x), \Theta(x) - \Theta(-x), x\}$ and Blue example final conditions, $\Phi_{\mathrm{B}}(x) \in \left\{\frac{1}{x}x^2, \frac{1}{2}x^2\Theta(-x), \Theta(|x| > \Delta) - \Theta(|x| < \Delta)\right\}$. We also find that certain combinations of parameters lead to an "arms-race" effect in both players' control policies; for these parameter combinations, Nash equilibrium strategies entail superexponential growth in the magnitude of each player's control policy near the end of the game. Figure 4.6 displays $E[u_{\mathrm{R}}]$ and $E[u_{\mathrm{B}}]$ for these parameter combinations, along with the middle 80 percentiles (10th to 90th percentile) of $u_{\mathrm{R}}(t)$ and $u_{\mathrm{B}}(t)$ for each t. This precipitous growth in the magnitude of the control policies occurs when either player has a discontinuous final condition. Although a discontinuous final condition by player i leads to a greater increase in the mean

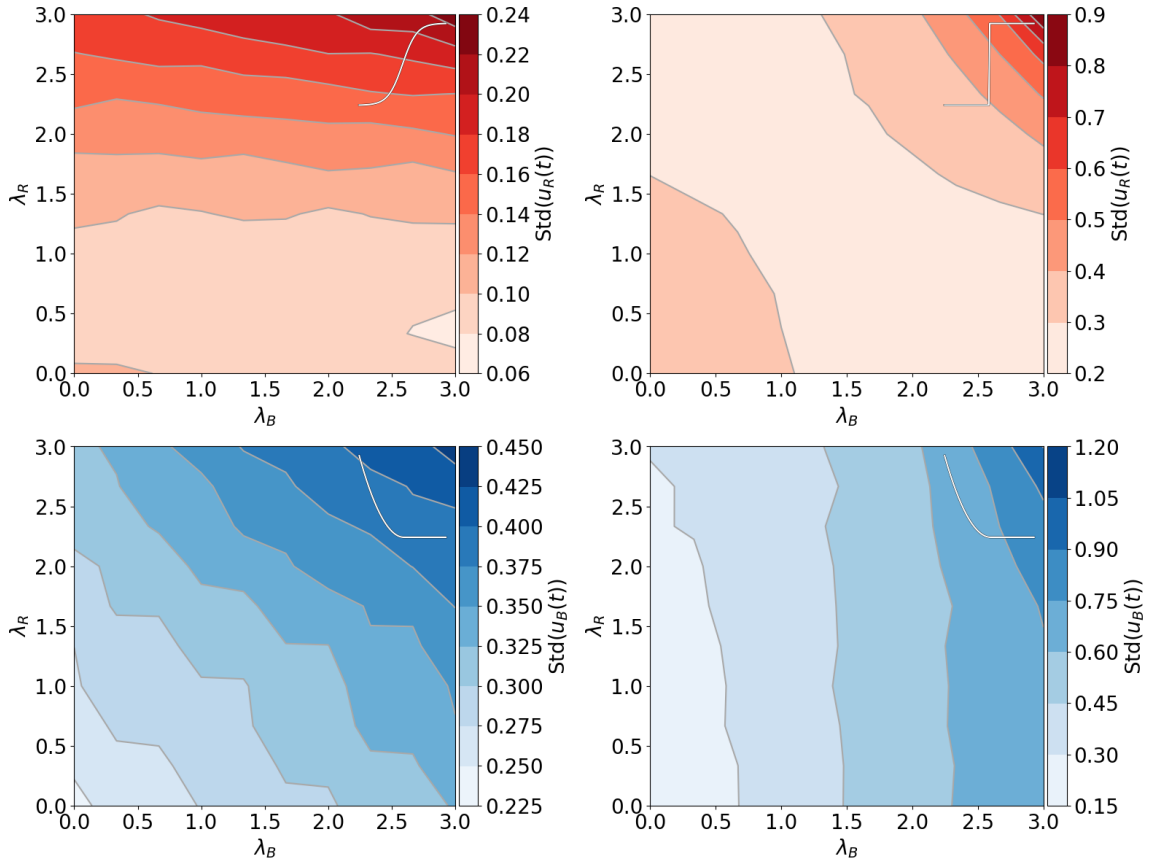

Figure 4.4: Example sweeps over the coupling parameters λ_{R} and λ_{B} when Blue's final condition is set to $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. We vary the coupling parameters over $[0, 3]$ and display the resulting standard deviation of the control policies $u_{\mathrm{R}}(x)$ and $u_{\mathrm{B}}(x)$. Panels A and B represent one coupled system of equations, while panels C and D represent a coupled system of equations with a different set of final conditions. In panel A, Red's value function is set to $\Phi_{\mathrm{R}}(x) = \tanh(x)$, while in panel B it is given by $\Phi_{\mathrm{R}}(x) = \Theta(x) - \Theta(-x)$, where $\Theta(\cdot)$ is the Heaviside function. We display a glyph of the corresponding final condition in the upper right corner of each panel. Changing Red's continuous final condition $\tanh(x)$ to the discontinuous $\Theta(x) - \Theta(-x)$ results in substantially increased variation in the control policies of both players.

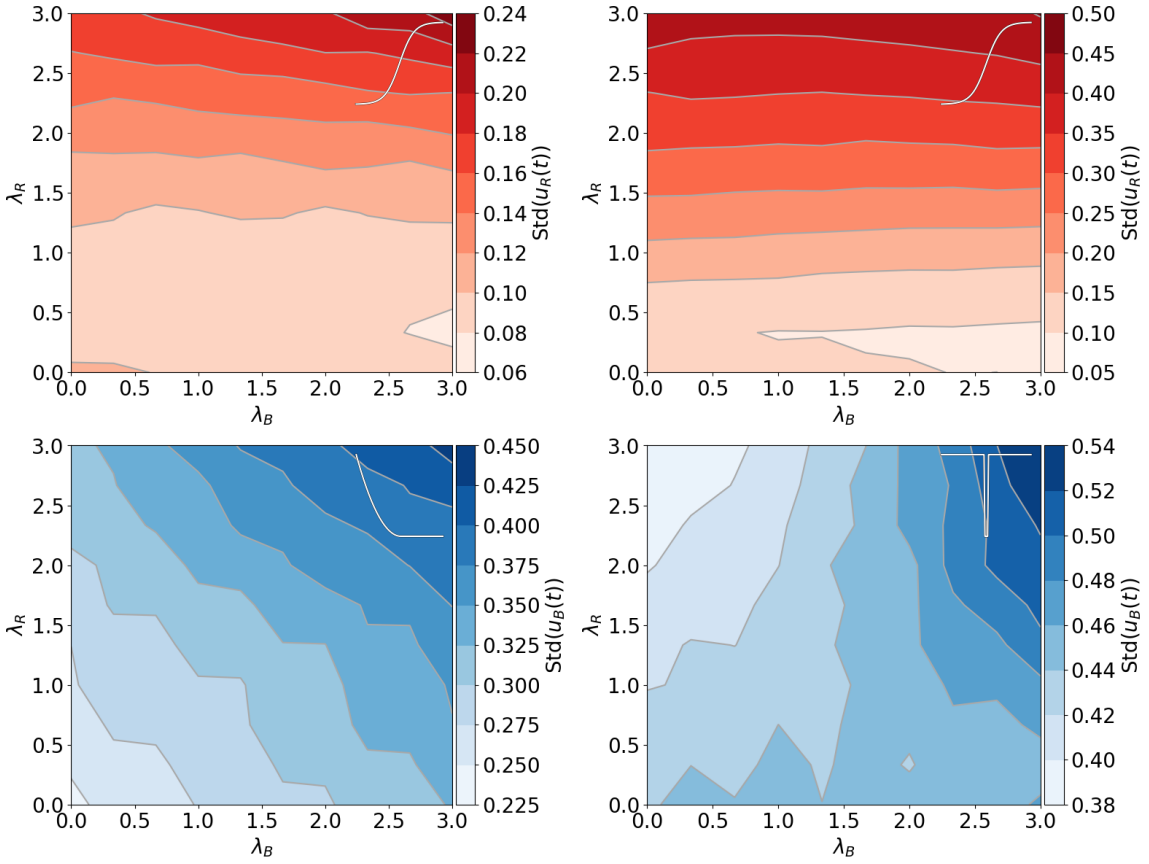

Figure 4.5: Example sweeps over the coupling parameters λ_{R} and λ_{B} when Blue's final condition is set to $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. We vary the coupling parameters over $[0, 3]$ and display the resulting means of the control policies $u_{\mathrm{R}}(x)$ and $u_{\mathrm{B}}(x)$. Panels A and B represent one coupled system of equations, while panels C and D represent a coupled system of equations with a different set of final conditions. In contrast with Fig. 4.4 we alter Blue's final condition from $\Phi_{\mathrm{B}}(x) = -\frac{1}{2}x^2\Theta(-x)$ in panel C to $\Phi_{\mathrm{B}}(x) = \Theta(|x| > 0.1) - \Theta(|x| \leq 0.1)$ in panel D, while Red's final condition is held constant at $\Phi_{\mathrm{R}}(x) = \tanh(x)$. Altering Blue's final condition from continuous to discontinuous causes a greater than 100% increase in the maximum value of the mean of Red's control policy.

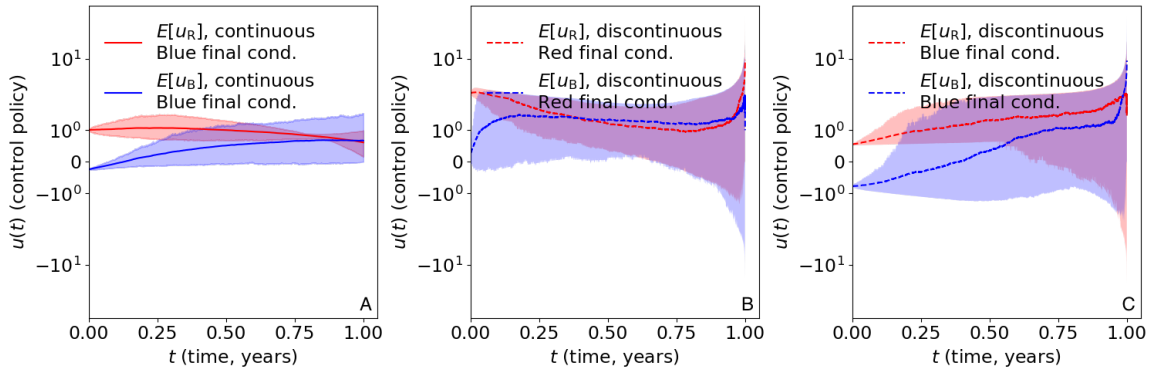

Figure 4.6: In the case of strong coupling (λ_R and $\lambda_B \gg 0$), discontinuous final solutions by either player cause superexponential growth in the magnitude of each player's control policy. Here we set $\lambda_R = \lambda_B = 3$ and integrate three systems, varying only one final condition in each. Panel A displays a system with two continuous final conditions: $\Phi_R(x) = \tanh(x)$ and $\Phi_B(x) = \frac{1}{2}x^2\Theta(-x)$. Panel B displays the mean Red and Blue control policies when the Red final condition is changed to $\Phi_R(x) = \Theta(x) - \Theta(-x)$ as the Blue final condition remains equal to $\frac{1}{2}x^2\Theta(-x)$, while panel C shows the control policies when $\Phi_B(x) = \Theta(|x| > 1.) - \Theta(|x| < 1)$ and $\Phi_R(x) = \tanh(x)$. The shaded regions correspond to the middle 80 percentiles (10th to 90th percentiles) of $u_R(t)$ and $u_B(t)$ for each t. When either player has a discontinuous final condition, the inter-percentile range is substantially wider for both players than when both players have continuous final conditions.

magnitude in player i's control policy than in player $\neg i$'s, the distribution of each player's policy exhibits a similar superexponential growth in dispersion (and hence in magnitude). To the extent that this model reflects reality, this points to a general statement about election interference operations: An all-or-nothing mindset by either Red or Blue regarding the final outcome of the election leads to an arms race that negatively affects both players. This is a general feature of any situation to which the model described by Eqs. 4.3 – 4.5 applies.

4.2.3 Credible commitment

If player $\neg i$ credibly commits to playing a particular strategy $v(t)$ on all of $[0, T]$, then the difficult problem of player i's finding a subgame-perfect Nash equilibrium strategy profile becomes a slightly easier problem of optimal control. Player i now seeks to find the policy $u(t)$ that minimizes the functional

$$E_{u,X}\left\{\Phi(X_T) + \int_0^T (u(t)^2 + \lambda v(t)^2)\ \mathrm{d}t\right\}, \tag{4.18}$$

subject to the modified state equation

$$\mathrm{d}x = [u(t) + v(t)]\mathrm{d}t + \sigma \mathrm{d}\mathcal{W}. \tag{4.19}$$

Following the logic of Eqs. 4.7 and 4.9, player i's value function is now given by the solution to the considerably-simpler HJB equation

$$\begin{aligned} -\frac{\partial V}{\partial t} &= -\frac{1}{4}\left(\frac{\partial V}{\partial x}\right)^2 + v(t)\frac{\partial V}{\partial x} + \lambda v(t)^2 + \frac{\sigma^2}{2}\frac{\partial^2 V}{\partial x^2}, \\ V(x,T) &= \Phi(x). \end{aligned} \tag{4.20}$$

Though nonlinear, this HJB equation can be transformed into a backward Kolmogorov equation through a change of variables and subsequently be solved using path integral methods [208]. Setting $V(x,t) = -\eta \log \varphi(x,t)$, substituting in Eq. 4.20, and performing the differentiation, we are able to remove the nonlinearity if and only if $\frac{\eta^2}{4}\frac{1}{\varphi^2}\left(\frac{\partial \varphi}{\partial x}\right)^2 = \frac{\sigma^2\eta}{2}\frac{1}{\varphi^2}\left(\frac{\partial \varphi}{\partial x}\right)^2$, so we set $\eta = 2\sigma^2$. Performing the change of variables, Eq. 4.20 is now linear and has a time-dependent drift and sink term,

$$\begin{aligned} \frac{\partial \varphi}{\partial t} &= \frac{\lambda}{2\sigma^2}v(t)^2\varphi(x,t) - v(t)\frac{\partial \varphi}{\partial x} - \frac{\sigma^2}{2}\frac{\partial^2 \varphi}{\partial x}, \\ \varphi(x,T) &= \exp\left\{-\frac{1}{2\sigma^2}\Phi(x)\right\} \end{aligned} \tag{4.21}$$

Application of the Feynman-Kac formula gives the solution to Eq. 4.21 as [209]

$$\begin{aligned} \varphi(x,t) = \exp\left\{-\frac{\lambda}{2\sigma^2}\int_t^T v(t')^2 \, \mathrm{d}t'\right\} \times \\ E_{Y_t}\left\{\exp\left[-\frac{1}{2\sigma^2}\Phi(Y_T)\right]\middle| Y_t = x\right\}, \end{aligned} \tag{4.22}$$

where Y_t is defined by

$$\mathrm{d}Y_t = v(t)\,\mathrm{d}t + \sigma \mathrm{d}\mathcal{W}_t,\ Y_0 = x. \tag{4.23}$$

Using this formalism, path integral control can be applied to estimate the value function for arbitrary $v(t)$. Fig. 4.7 displays path integral solutions to Eq. 4.20 when

player $\neg i$ credibly commits to playing $v(t) = t^2$ for the duration of the game and player i's final cost function takes the form $\Phi(x) = \Theta(|x| > 1) - \Theta(|x| \leq 1)$. In the further restricted case where there is a credible commitment by one party to play $v(t) = v$, a constant control policy, we can say more about the nature of solutions. We will also show presently why this constraint is actually not all that restrictive. Under this assumption, the probability law corresponding with Eq. 4.23 is given by

$$u(y,t) = \frac{1}{\sqrt{2\pi\sigma^2 t}} \exp\left\{\frac{1}{2\sigma^2 t}[(y-x) - vt]^2\right\}, \tag{4.24}$$

so that the (exponentially-transformed) value function reads

$$\begin{aligned}\varphi(x,t) = &\frac{\exp\left\{-\frac{\lambda v^2}{2\sigma^2}(T-t)\right\}}{\sqrt{2\pi\sigma^2(T-t)}} \times \\ &\int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2\sigma^2}\left[\Phi(y) + \frac{((y-x) - v(T-t))^2}{T-t}\right]\right\} \mathrm{d}y.\end{aligned} \tag{4.25}$$

This integral can be evaluated exactly for many $\Phi(y)$ and, for many Φ, can be approximated using the method of Laplace. When $t \to T$ so that the denominator of the argument of the exponential in Eq. 4.25 approaches zero, Laplace's approximation to the integral reads

$$\begin{aligned}&\int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2\sigma^2}\left[\Phi(y) + \frac{((y-x) - v(T-t))^2}{T-t}\right]\right\} \mathrm{d}y \\ &\quad \simeq \sqrt{2\pi\sigma^2(T-t)} \exp\left\{-\frac{1}{2\sigma^2}\Phi(x + (T-t)v)\right\},\end{aligned} \tag{4.26}$$

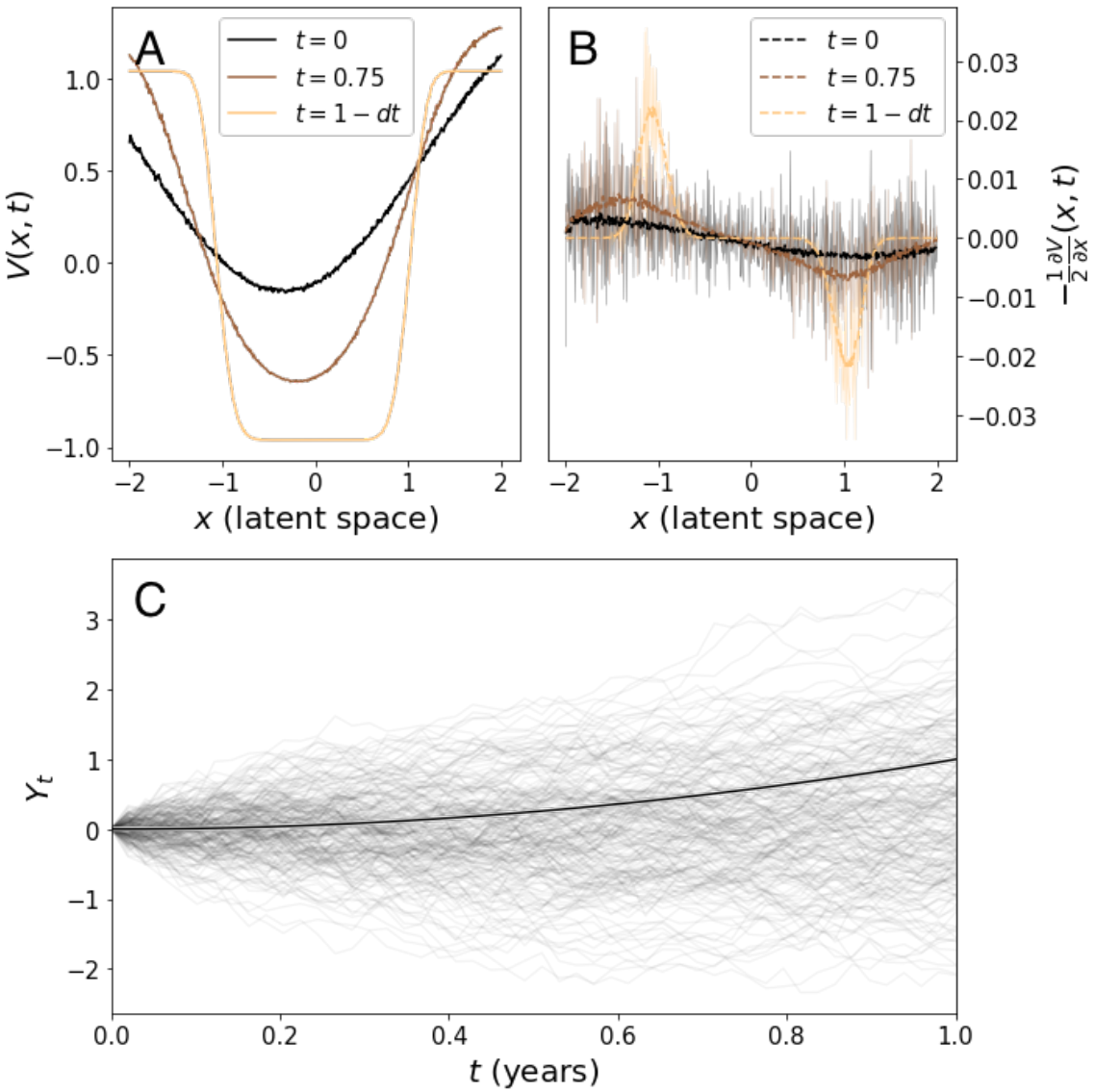

Figure 4.7: Result of the path integral Monte Carlo solution method applied to Eq. 4.20 with the final condition $\Phi(x) = \Theta(|x| > 1) - \Theta(|x| \leq 1)$ *and* $v(t) = t^2$*. Approximate value functions are computed using* $N = 10000$ *trajectories sampled from Eq. 4.23 for each point* (x, t)*. Approximate value functions are displayed in Panel A for* $t \in \{0, 0.75, 1 - \mathrm{d}t\}$ *and the corresponding approximate control policies in Panel B, along with their smoothed counterparts (15-step moving averages, plotted in dashed curves). Panel C displays realizations of* Y_t*, the process generating the measure under which the solution is calculated. This method can be advantageous over numerical solution of the nonlinear PDE when the final condition is discontinous, as here, since in this case Eq. 4.20 has a solution for all* $t \in [0, T]$ *only in the sense of distributions. The analytical control policy at* $t = T$ *is given by* $u(t) = -\frac{1}{2}[\delta(x - 1) - \delta(x + 1)]$*.*

so that, inverting the transformation above, the value function can be approximated by

$$V(x,t) = \lambda v^2(T-t) + \Phi(x + (T-t)v), \tag{4.27}$$

and the control policy by

$$u(t) = -\frac{1}{2}\Phi'(x + (T-t)v). \tag{4.28}$$

Fig. 4.8 displays the results of approximating the value function with Eq. 4.27 at $t = 0$, along with the true (numerically-determined) value function at both $t = 0$ and, for reference, $t = T$.

It is interesting to analyze the dependence of the Laplace-approximated value function on a free parameter. If the simplicity of the Laplace approximation is to have practical utility, the approximated control policy should ideally have similar scaling and asymptotic properties as the true control policy; solving an optimization problem for optimal values of the free parameter, which we will denote by a, may be one approach to satisfying this desideratum. To this end, as a case study we consider the behavior of the approximate value function $V^{(a)}(x,t)$ and its corresponding control policy $u^{(a)}(x,t)$ when the final condition is $\Phi^{(a)}(x) = \tanh(ax)$ as $a \to \infty$. We consider this specific example because $\Phi^{(a)}(x) \to \Theta(x) - \Theta(-x)$, where $\Theta(\cdot)$ is the Heaviside function; this limit can be the source of complicated behavior in a variety of fields such as piecewise-smooth dynamical systems (both deterministic and stochastic) [210, 211], Coulombic friction [212], and evolutionary biology [213]. Fig. 4.9 displays the exponentially-transformed value function Eq. 4.25 with final condition $\Phi_i(x) = \tanh(ax)$ for player i when player $\neg i$ commits to playing a constant strategy

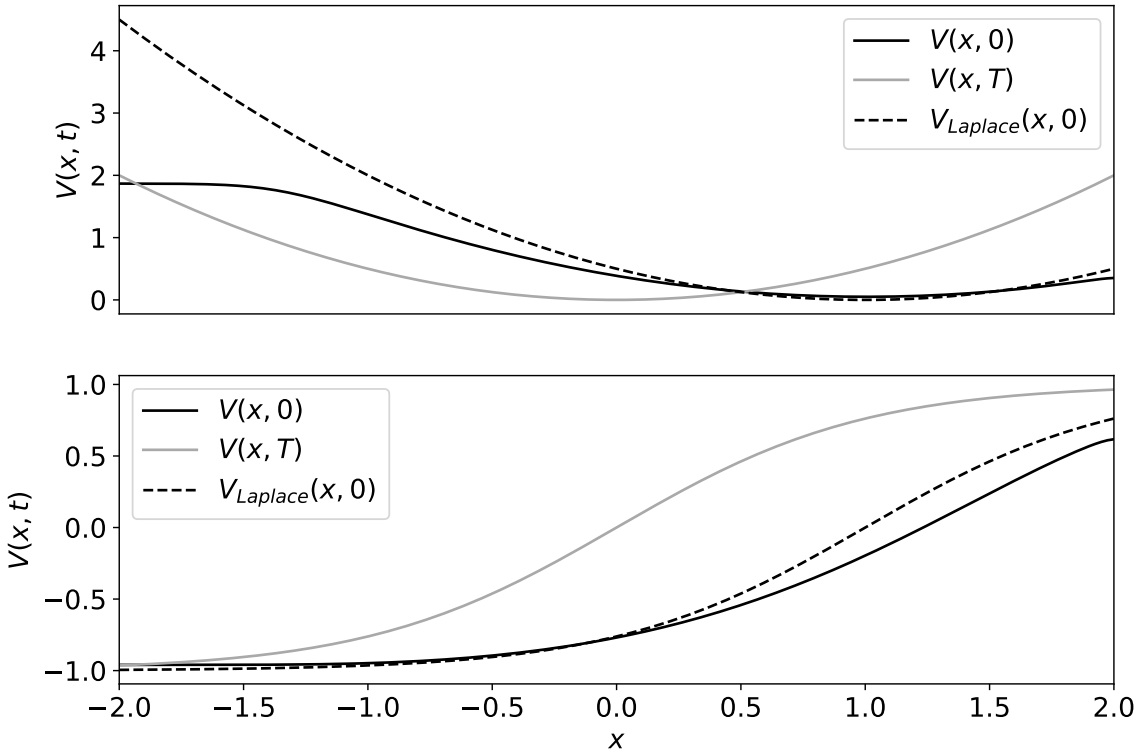

Figure 4.8: When player $\neg i$ commits to playing a constant strategy profile $v(t) = v$ for a fixed interval of time, an analytic approximate form for player i's value function $V(x,t)$ is given by $V(x,t) \simeq \lambda v^2(T-t) + \Phi(x+(T-t)v)$. The numerically-determined value functions at time $t = 0$ are shown above in black curves, while the Laplace approximations at $t = 0$ are displayed in dashed curves. The curves of lighter hue are the value functions at the final time T. The top panel demonstrates results for the final condition $\Phi(x) = \frac{1}{2}x^2$, while the bottom panel has $\Phi(x) = \tanh(x)$.

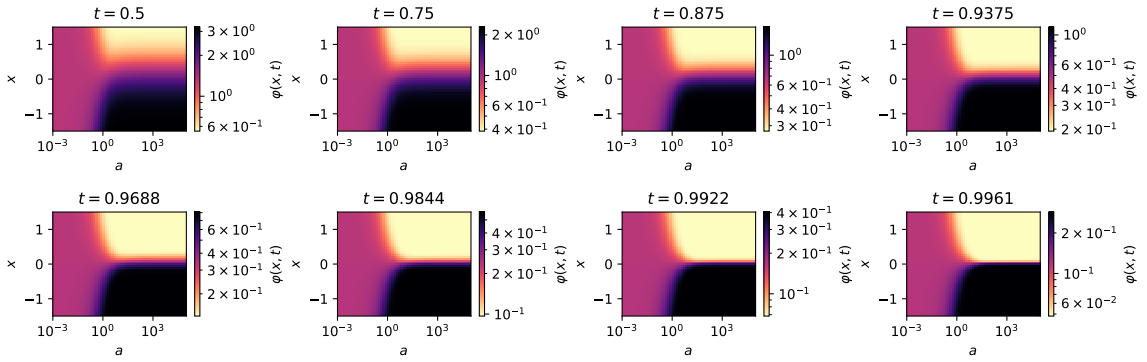

Figure 4.9: If player $\neg i$ credibly commits to a strategy of playing a constant strategy with value equal to v for the entire duration of the game, player i's (exponentially-transformed) value function $\varphi(x,t)$ has an integral representation given by Eq. 4.25. We display dynamics of $\varphi(x,t)$ in the case where $\Phi(x) = \tanh(ax)$ for $x \in \left(-\frac{3}{2}, \frac{3}{2}\right)$ and logarithmically equally-spaced values of $a \in [10^{-3}, 10^{5}]$. For $a < 10^{-1}$, the value function is nearly constant as a values this small render the final condition nearly constant over this range of the state space. When $a > 10^{1}$, $\frac{\partial}{\partial x}\varphi(x,t)$ rapidly increases in magnitude near $x = 0$ as $t \to T$.

of v for the entire time period. (The value function is computed numerically; we do not use the Laplace approximation here.) As $t \to T$, larger values of the steepness parameter a lead to an increasingly hard boundary betweeen areas of the state space that are costly for player i (positive values of x) and those that are less costly (negative values of x). Though this behavior is qualitatively similar to behavior arising from the final condition $\Phi_i(x) = \Theta(x) - \Theta(-x)$, we will show presently, using the Laplace approximation, that there are significant scaling differences in the control policies resulting from these strategies. From the Laplace approximation result (Eqs. 4.26 and 4.27), the value function is approximately

$$V^{(a)}(x,t) \approx \lambda v^2 (T-t) + \tanh\left\{a[x + v(T-t)]\right\}, \tag{4.29}$$

and hence the control policy is approximately given by

$$u^{(a)}(x,t) \approx -\frac{a}{2}\mathrm{sech}^2\left\{a[x+v(T-t)]\right\}, \tag{4.30}$$

with both expansions valid when $\frac{1}{\sigma^2(T-t)}$ is large. When $\Phi(x) = \Theta(x) - \Theta(-x)$, the value function can be computed analytically; we have

$$\begin{aligned}\frac{1}{\sqrt{2\sigma^2(T-t)}}\int_{-\infty}^{\infty}\exp\left\{-\frac{1}{2\sigma^2}\left[\Theta(y)-\Theta(-y)+\frac{((y-x)-v(T-t))^2}{T-t}\right]\right\}\mathrm{d}y \\ = \cosh\left(\frac{1}{2\sigma^2}\right)+\sinh\left(\frac{1}{2\sigma^2}\right)\mathrm{erf}\left(-\frac{x+v(T-t)}{\sqrt{2\sigma^2(T-t)}}\right),\end{aligned} \tag{4.31}$$

whereupon we find that

$$V(x,t) = \lambda v^2(T-t) - 2\sigma^2\log\left[\cosh\left(\frac{1}{2\sigma^2}\right)+\sinh\left(\frac{1}{2\sigma^2}\right)\mathrm{erf}\left(-\frac{x+v(T-t)}{\sqrt{2\sigma^2(T-t)}}\right)\right] \tag{4.32}$$

and

$$u(x,t) = -\sqrt{\frac{2\sigma^2}{\pi(T-t)}}\frac{\exp\left(\frac{-(x+v(T-t))^2}{2\sigma^2(T-t)}\right)}{\coth\left(\frac{1}{2\sigma^2}\right)+\mathrm{erf}\left(-\frac{x+v(T-t)}{\sqrt{2\sigma^2(T-t)}}\right)}. \tag{4.33}$$

The approximate control policy $u^{(a)}(x,t)$ and the limiting control policy have similar purely negative bell-like shapes but also differ substantially as $t \to T$: The true control policy decays as a Gaussian (though a Gaussian modulated by the asymmetric function $\mathrm{erf}(\cdot)$), while the approximate policy displays logistic decay and hence is larger in magnitude farther from its global minimum than is the true policy; while the approximate policy is symmetric, $u(x,t)$ is asymmetric due to the error function term. The differences between $u^{(a)}(x,t)$ and $u(x,t)$ can be minimized by considering

the free parameter a as a function of t and solving the functional minimization problem

$$\min_{a(t)} \int_t^T \int_{-\infty}^{\infty} [u^{(a(t))}(x,t) - u(x,t)]^2 \, \mathrm{d}x \, \mathrm{d}t. \tag{4.34}$$

The variational principle implies a stationary point of this problem is given by the solution to

$$\int_{-\infty}^{\infty} [u^{(a(t))}(x,t) - u(x,t)] \frac{\partial u^{(a(t))}(x,t)}{\partial a(t)} \, \mathrm{d}x = 0. \tag{4.35}$$

We are unable to compute this integral analytically upon substituting Eqs. 4.30 and 4.33; we instead find the solution to the problem of Eq. 4.34 by numerically solving Eq. 4.35 using the secant method for each of 100 linearly-spaced $t \in \left[\frac{1}{2}, \frac{9975}{10000}\right]$. We display the optimal $a(t)$, along with the corresponding $u^{(a(t))}(x,t)$ and true $u(x,t)$ in Fig. 4.10. We find that the optimal $a(t)$ grows superexponentially as $t \to T$ and that the accuracy of the approximation increases in this limit, which is expected given that $u^{(a)}(x,t)$ is derived using the Laplace approximation and it is in this limit that the Laplace approximation is valid.

Even with the seemingly-restrictive assumption of credible committment to a constant control policy v, this theory can be used to provide a method for value function approximation in a noncooperative scenario. For arbitrary $v(t)$, expansion about $t + \Delta t$ gives $v(t + \Delta t) = v(t) + v'(t)\Delta t$, leading to an approximate value function iteration over a small time increment Δt,

$$V(x, t + \Delta t) \approx \lambda v(t)^2 (T - t) + \Phi(x + (T - t)[v(t) + v'(t)\Delta t]). \tag{4.36}$$

In application, both of $v(t)$ and $v'(t)$ can be estimated from possibly-noisy data on

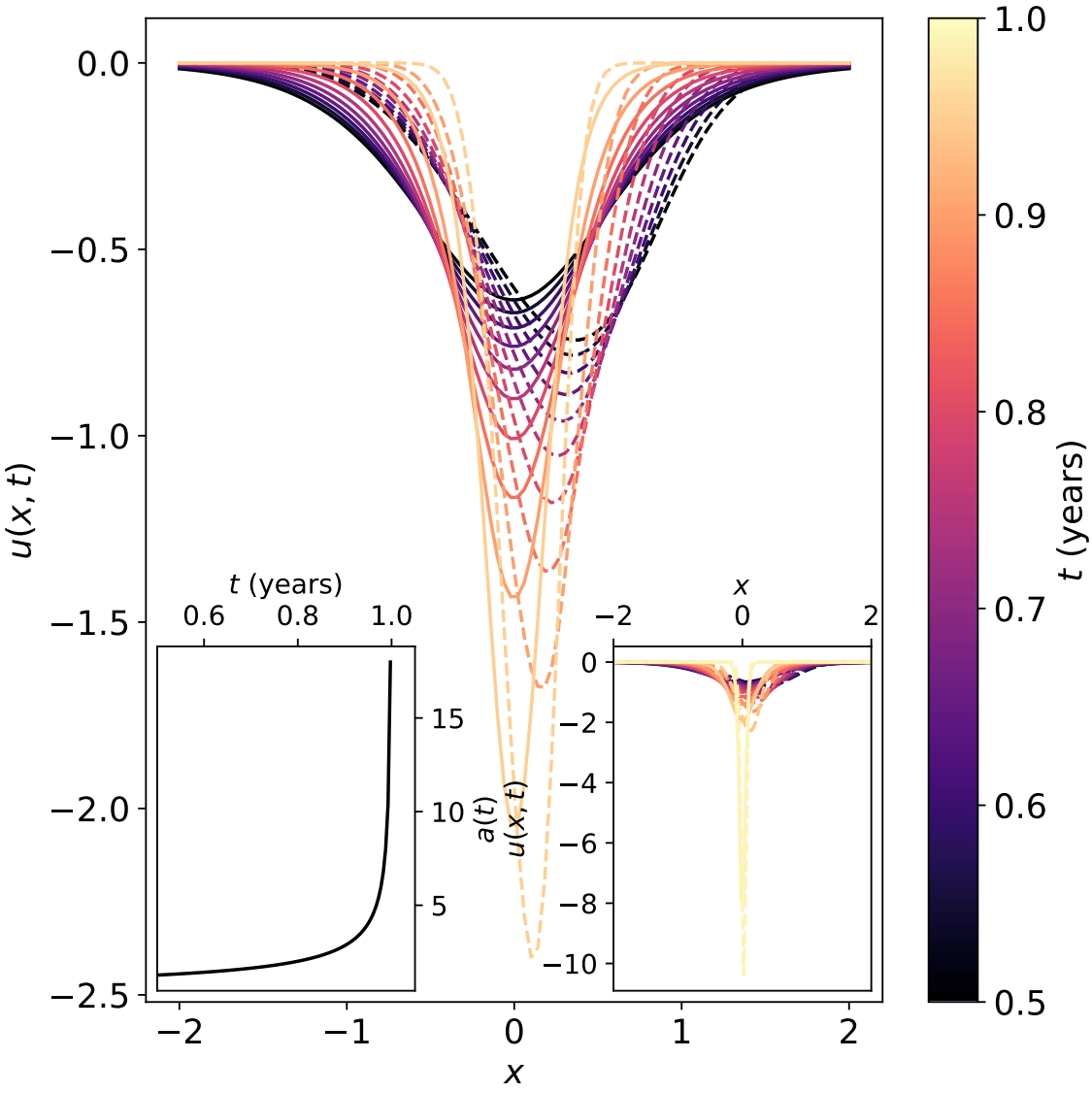

Figure 4.10: The solution to the problem formulated in Eq. 4.34 is a superexponentially-increasing $a(t)$ parameter in the Laplace method-derived value function $V^{(a)}(x,t) = \tanh(ax)$. We use this value function as an approximation to the exact value function given in Eq. 4.32. Dashed curves indicate $u(x,t)$, while solid curves indicate $u^{(a)}(x,t)$. The lower-right inset axis displays the same data as the main axis and also includes $u(x,t)$ and $u^{(a)}(x,t)$ at the last simulation timestep, $t = 0.9975$, to demonstrate the increasing accuracy of the approximation as $t \to T$. The lower-left inset displays the optimal $a(t)$.

$t' \in [0, t]$ and used in this approximation.

4.3 Application

A recent notable example of election interference operations is that of the Russian military foreign intelligence service (GRU)'s and Internet Research Agency (IRA)'s operations in the 2016 U.S. presidential election contest to attempt to harm one candidate's chances of winning (Hilary Clinton) and aid another candidate (Donald Trump) [199]. Though Russian foreign intelligence had conducted election interference operations in the past at least once before, in the Ukrainian elections of 2014 [214], the 2015–16 operations were notable in that IRA operatives used the microblogging website Twitter in an attempt to influence the election outcome. When this attack vector was discovered, Twitter accounts associated with IRA activity were shut down and all data associated with those accounts was collected and analyzed [215, 216, 217]. There has been extensive analysis of the qualitative and statistical effects of these and other election attack vectors (e.g., Facebook advertisement purchases) on election polling and the outcome of the election [218], while there has also been some work on the detection of election influence campaigns more generally [219, 220]. However, to the best of the authors' knowledge, there exists no publicly-available effort to reverse-engineer the exact qualitative nature of the control policies used by the the IRA—Red team—and by U.S. domestic and foreign intelligence agencies—Blue team.

In an effort to perform data-driven simulation of Red-Blue dynamics, we fit a form of the model described in Sec. 4.2.1 and compare it to qualitative theoretical pre-

dictions, finding the free parameters in the model that best describe the observed data and inferred latent controls. It is relatively nontrivial to fit the parameters of the theoretical model because we are faced with two distinct sources of uncertainty: first, we cannot observe either Red's or Blue's control policy directly because foreign and domestic intelligence agencies shroud their activities in secrecy; and second, the payoff structure to each player at the final time, which is necessary for a unique solution to Eqs. 4.11 and 4.12, is also secret and unknown to us. To partially circumvent these issues, we construct a two-stage model. The first stage is a Bayesian structural time series model, depicted graphically in Fig. 4.11, through which we are able to infer distributions of discretized analogues of $u_{\mathrm{R}}(t)$, $u_{\mathrm{B}}(t)$, and $x(t)$. Once these distributions are in hand, we minimize a loss function that compares the means of these distributions to the means of distributions produced by the model described in Sec. 4.2.1.

Since the U.S. presidential election system is of nontrivial complexity, owing both to the number of minor party candidates that also compete and also to the unique Electoral College system, we make the simplifying assumptions stated in Sec. 4.1—namely, that only two candidates contest the election and that the election process is modeled by a simple "candidate A versus candidate B" poll. Though there are undoubtedly better methods for forecasting elections, such as compartmental infection models [221], prediction markets [222], and more sophisticated Bayesian models [223, 224], we purposefully construct our statistical model to closely mimic the underlying election model of Sec. 4.2.1 to test the ability of this underlying model, coupled with our model of noncooperative interaction, to reproduce inferred control and observed election dynamics through posterior and posterior predictive distributions. We can

observe neither the Red $u_{\mathrm{R}}(t)$ nor Blue $u_{\mathrm{B}}(t)$ control policies, but are able to observe a proxy for u_{R}, namely, the number of tweets sent by IRA-associated accounts in the year leading up to the 2016 election [2]. This dataset contains a total of 2,973,371 tweets from 2,848 unique Twitter handles. Of these tweets, a total of 1,107,361 occurred in the year immediately preceding the election (08/11/2015 - 08/11/2016). We grouped these tweets by day and used the time series of total number of tweets on each day as an observable from which u_{R} could be inferred. We restricted the time range of the model to begin at the later of the end dates of the Republican National Convention (21 July 2016) and Democratic National Convention (28 July 2016) since the later of these dates, 28 July 2016, is the day on which the race is officially between two major party candidates. Of the above tweets, 363,131 occurred during the 102 days beginning on 28 July 2016 and ending the day before Election Day. We note that the presence of minor party candidates doubtless played a role in the result of the election, but even the most prominent minor parties (Libertarian and Green) received only single-digit support [225, 226]. We do not model these parties and instead consider only the zero-sum electoral contest between the two major parties. We used the RealClearPolitics poll aggregation as a proxy for the electoral process itself [3], averaging polls that occurred at the same timestamp and using the earliest date in the date range of the poll if it was conducted over multiple days as the timestamp of that observation.

Using these two observed random variables, we fit a Bayesian structural time series model [227] of the form presented in Fig. 4.11. We briefly describe the structure of the model and explain our choices of priors and likelihood functions. In the analytical

[2]Data can be downloaded at https://github.com/fivethirtyeight/russian-troll-tweets/

[3]Data can be downloaded at https://www.realclearpolitics.com/epolls/2016/president/us/general_election_trump_vs_clinton_vs_johnson_vs_stein-5952.html

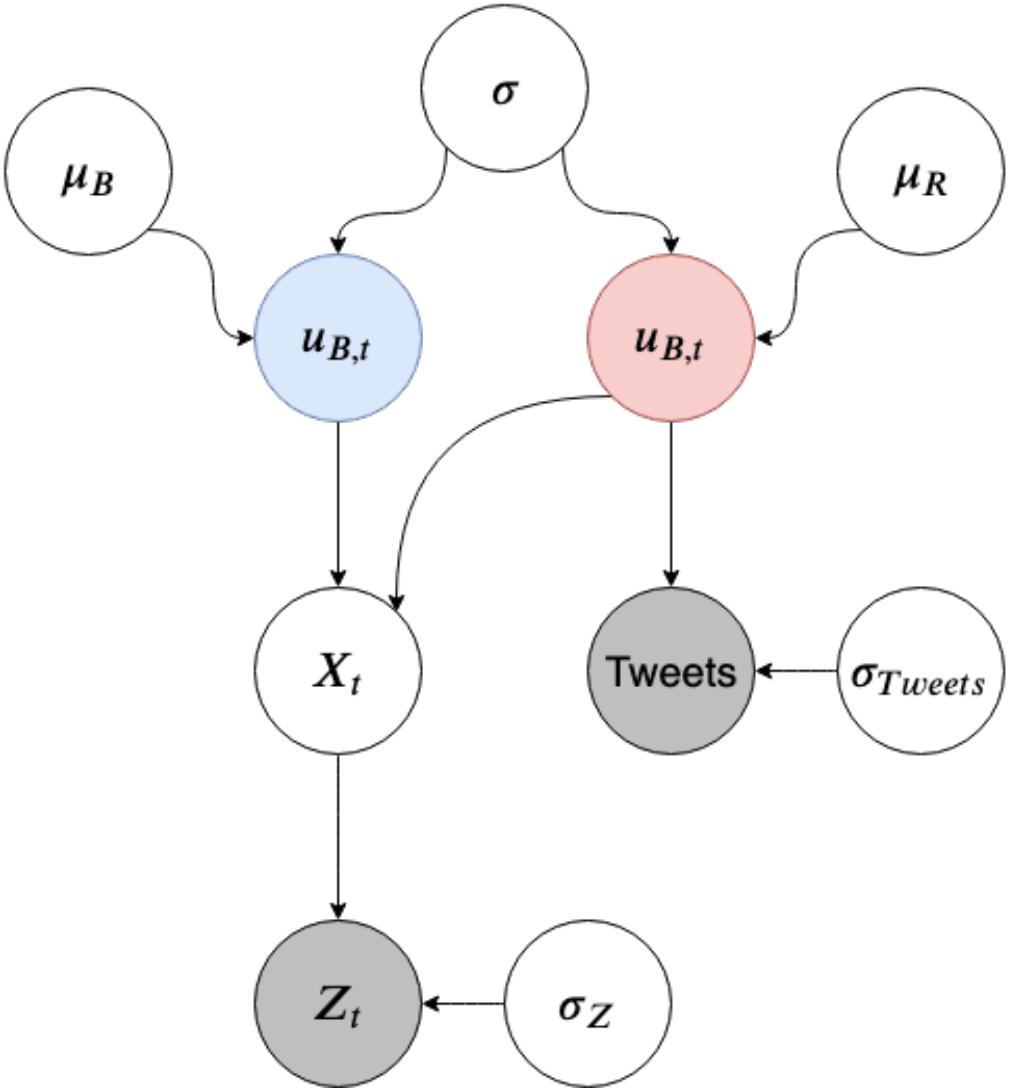

Figure 4.11: We approximate the time series components of the analytical model defined in Sec. 4.2.1 by a Bayesian structural time series model that we subsequently confront with data pertaining to the 2016 U.S. presidential election. Observed random variables are denoted by gray-shaded nodes, while latent random variables are represented by unshaded nodes or red ($u_{\mathrm{R},t}$) and blue ($u_{\mathrm{B},t}$) nodes. We observe a noisy election poll, denoted by Z_t, and a time series of tweets associated with the Russian Internet Research Agency, denoted by Tweets. Our objective in this modeling stage is to infer the latent electoral process, denoted by X_t, and the latent control policies.

model, we model the latent control policies $u_{\mathrm{R}}(t)$ and $u_{\mathrm{B}}(t)$ implicitly as time- and state-dependent Wiener processes. This is seen by recalling that the state equation evolves according to a Wiener process and applying Ito's lemma to the deterministic functions of a random variable $-\frac{1}{2}\frac{\partial V_{\mathrm{R}}}{\partial x'}\Big|_{x'=x_t}$ and $-\frac{1}{2}\frac{\partial V_{\mathrm{B}}}{\partial x'}\Big|_{x'=x_t}$, which define the control policies. A discretized version of the Wiener process is a simple Gaussian random walk; we thus model the latent Red and Blue control policies by Gaussian random walks:

$$p(u_{R,t}|u_{R,t-1},\mu_R,\sigma) = \mathcal{N}(u_{R,t-1}+\mu_R,\sigma^2) \tag{4.37}$$

$$p(u_{B,t}|u_{B,t-1},\mu_B,\sigma) = \mathcal{N}(u_{B,t-1}+\mu_B,\sigma^2) \tag{4.38}$$

Similarly, the latent election process is modeled by a discretized version of the state evolution equation, Eq. 4.3:

$$\begin{aligned} &p(X_t|X_{t-1},u_R,u_B) \\ &\quad = \mathcal{N}(X_{t-1}+u_{B,t-1}-u_{R,t-1},1) \end{aligned} \tag{4.39}$$

We assume that the latent election model is subject to normal observation error in latent space. Since we chose a logistic function as the link between the latent and real (on $(0,1)$) election spaces, the likelihood for the observed election process is thus given by a Logit-Normal distribution. The pdf of this distribution is

$$p(Z_t|X_t,\sigma_Z) = \sqrt{\frac{1}{2\pi\sigma_Z^2}}\frac{\exp\left\{-\frac{(\mathrm{logit}(Z_t)-X_t)^2}{2\sigma_Z^2}\right\}}{Z_t(1-Z_t)}. \tag{4.40}$$

Though the number of IRA tweets that occur on any given day is obviously a non-

negative integer, we chose not to model it this way. The usual model for a "count" random variable, such as the tweet time series, is a Poisson distribution with time-dependent rate parameter. However, this model imposes a strong assumption on the variance of the count distribution (namely, that the variance and mean are equal) which does not seem realistic in the context of the tweet data. Instead of searching for a discrete count distribution that meets some optimality criterion, we instead chose to first normalize the tweet time series to have zero mean and unit variance (thus making it a continuous random variable rather than a discrete one) and then to shift the time series so that the the new time series was equal to zero on the day during our study with the fewest tweets. We then modeled this time series Tweets_t by a normal observation likelihood,

$$p(\text{Tweets}_t | u_{R,t}, \sigma_{\text{Tweets}}) = \mathcal{N}(u_{R,t}, \sigma^2_{\text{Tweets}}). \tag{4.41}$$

We placed a weakly-informative prior, a Log-Normal distribution, on each standard deviation random variable (σ, σ_Z, σ_{Tweets}), and zero-centered Normal priors on each mean random variable (μ_R, μ_B). This model is high-dimensional, since the observed time series Tweets and Z and latent time series X, u_{R}, and u_{B} are inferred as T-dimensional vectors possessing the covariance structure imposed by the (biased) random walk assumption; in total this model has $5T + 5 = 515$ degrees of freedom.

We fit this statistical model, a graphical representation of which is displayed in Fig. 4.11, using the No-U-Turn Sampler algorithm [228], sampling 2000 draws from the model's distribution from each of two independent Markov chains, not including 1000

draws per chain of burn-in. The sampler appeared to converge well based on graphical consideration (i.e., the "eye test") of draws from the posterior predictive distribution of Z_t and Tweets_t, and—more importantly—because maximum values of Gelman-Rubin statistics [229] for all variables satisfied $R_{\max} < 1.01$ except for that of σ_Z ($R_{\max} = 1.07646$). Each of these values is well below the level $R = 1.1$ advocated by Brooks and Gelman [230]. Fig. 4.12 displays draws from the posterior and posterior predictive distribution of this model. Panel A displays draws of X_t from the posterior distribution, along with $E[X_t]$ and $\text{logit}(Z_t)$, while in panel B we show posterior draws of u_{R} and u_{B}, along with $E[u_{\text{R}}]$ and $E[u_{\text{B}}]$ in thick red and blue curves respectively. In panel C, we display Tweets_t and draws from its posterior predictive distribution. On 10/06/2016, Tweets_t exhibits a large spike that is very unlikely under the posterior predictive distribution. This spike likely corresponds with a statement made by the U.S. federal government on this date that officially recognized the Russian government as culpable for hacking the Democratic National Committee computers.

After inferring the latent control policies and electoral process, we searched for the parameter values $\theta = (\lambda_{\text{R}}, \lambda_{\text{B}}, \sigma, \Phi_{\text{R}}, \Phi_{\text{B}})$ of the theoretical model described in Sec. 4.2.2 that best explain the observed data and latent variables inferred by the time series model described in this section. For clarity in reference, we will hereafter refer to this theoretical model as $\hat{\mathcal{M}}$ and the Bayesian structural time series model derived earlier in this section as $\mathcal{M}$. We use Legendre polynomials to approximate the final conditions of Eqs. 4.11 and 4.12, Φ_{R} and Φ_{B}, setting $\Phi_i(x) \simeq \sum_{k=0}^{K} a_{ik} P_k(x)$, so that the actual parameter vector considered is $\theta = (\lambda_{\text{R}}, \lambda_{\text{B}}, \sigma, a_{0,r}, ..., a_{K,r}, a_{0,b}, ..., a_{K,b})$. In contrast with $\mathcal{M}$, $\hat{\mathcal{M}}$ has relatively few degrees of freedom since the assumption of state and policy co-evolution via solution of coupled partial differential equations

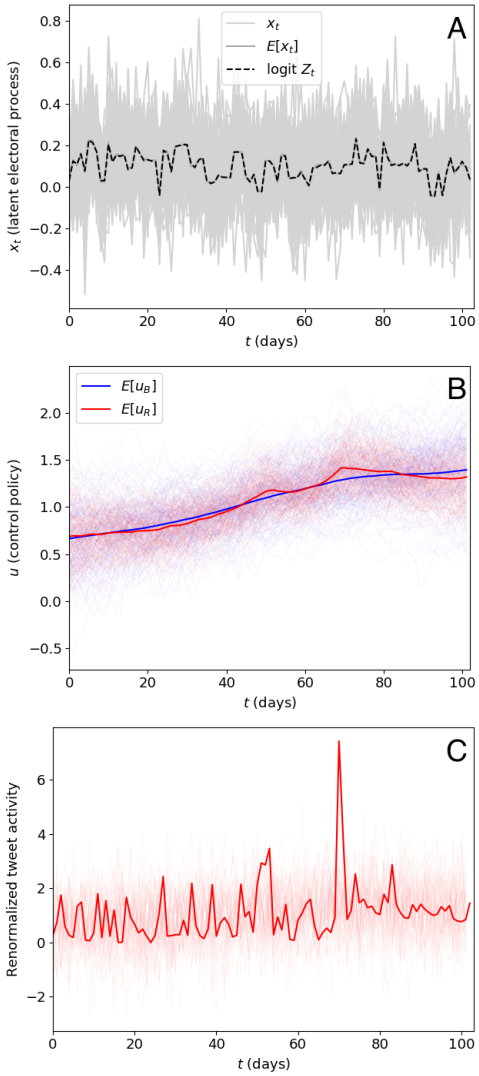

Figure 4.12: Panel A displays the logit of the observed election time series (black curve) $\text{logit}(Z_t)$*, along with the posterior distribution of the latent electoral process* X_t*. Panel B displays the mean latent control policies in thick red and blue curves, along with their posterior distributions. Panel C shows the true tweet time series (subject to the normalization described in the main body) along with draws from its posterior predictive distribution. The large spike in the tweet time series that is not predicted by the posterior predictive distribution corresponds to the day (10/06/2016) on which the U.S. federal government officially accused Russia of hacking the Democratic National Committee computers.*

(Eqs. 4.11 and 4.12) substantially restricts the system's dynamics. In total, $\hat{\mathcal{M}}$ has $2K + 3$ free parameters; we set $K = 10$ for a total of 23 degrees of freedom. The theoretical model $\hat{\mathcal{M}}$ can be viewed as a generative probabilitistic function so that, to find values of parameters that ensure $\hat{\mathcal{M}}$ best describes observed and inferred reality, we draw $(\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{B}}, \hat{X}) \sim \hat{p}(\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{B}}, \hat{X}|\theta, \hat{\mathcal{M}})$ and minimize a loss function of these generated values and the values inferred by $\mathcal{M}$. We defined a loss function of the form

$$L(\theta|\hat{\mathcal{M}}) = \sum_{(y,\hat{y})} \left[\|\mu_y - \mu_{\hat{y}}\|_2^2 + \eta\sigma_{\hat{y}} \right], \tag{4.42}$$

where $y \in \{u_{\mathrm{R}}, u_{\mathrm{B}}, X\}$ and $\hat{y} \in \{\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{B}}, \hat{X}\}$. We have defined the mean and standard deviation under the corresponding distribution by μ and σ respectively. The ℓ_2 loss terms penalize deviation by $\hat{\mathcal{M}}$ from the mean of $\mathcal{M}$'s inferred posterior distribution, while the standard deviation term imposes a penalty on dispersion. We minimized the loss function of Eq. 4.42 using a Gaussian-process Bayesian optimization algorithm, the details of which are beyond the scope of this work but are readily found in any review paper on the subject [231, 232, 233]. Fig. 4.13 displays the result of this optimization procedure for $K = 10$ and $\eta = 0.002$. For this set of hyperparameters, we found coupling parameter values of $\lambda_{\mathrm{R}} = 0.1432$ and $\lambda_{\mathrm{B}} = 1.7847$ and a latent space volatility of $\sigma = 0.7510$. Panel A of Fig. 4.13 displays $\mathrm{logit}(Z_t)$ in a thick black curve and draws of $\hat{X}$ from $\hat{\mathcal{M}}$ in grey curves; $\mathrm{logit}(Z_t)$ is centered in the distribution of $\hat{X}$ and hence has a high probability under $\hat{\mathcal{M}}$. In panel B, we show $E[u_{\mathrm{R}}]$ and $E[u_{\mathrm{B}}]$ in thick red and blue curves respectively along with empirical distributions of $\hat{u}_{\mathrm{R}}$ and $\hat{u}_{\mathrm{R}}$. These empirical distributions exhibit heteroskedasticity; their variance increases as $t \to 102$ days—the last day before the election. In Fig. 4.14 we expand on panel B, displaying a forest plot with the time-to-go $T - t$ on the vertical axis

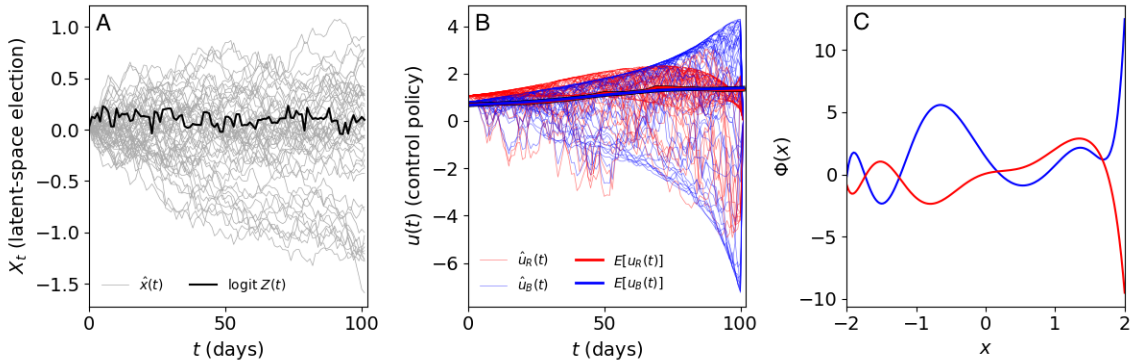

Figure 4.13: Parameter values found though application of a Bayesian optimization algorithm to the problem of finding optimal parameters of $\hat{\mathcal{M}}$ using the objective function given by Eq. 4.42 generate the above distributions of latent election process X and Red and Blue control policies, u_{R} and u_{B}. We ran the optimization algorithm with the number of terms of the Legendre expansion of Φ_{R} and Φ_{B} set to $K = 10$ and set the variance regularization to $\eta = 0.002$, which resulted in fit parameters of $\lambda_{\mathrm{R}} = 0.849$, $\lambda_{\mathrm{B}} = 0.727$, and $\sigma = 1.509$. Panel A displays draws from the latent electoral process under $\hat{\mathcal{M}}$, along with $\mathrm{logit}(Z_t)$, the logit-transformed real polling popularity process. Panel B displays draws from the distributions of $\hat{u}_{\mathrm{R}}$ and $\hat{u}_{\mathrm{B}}$ under $\hat{\mathcal{M}}$, while panel C displays the inferred final conditions $\Phi_{\mathrm{R}}(x)$ and $\Phi_{\mathrm{B}}(x)$.

and the middle 80 (10th to 90th) percentiles of the empirical distributions of $\hat{u}_{\mathrm{R}}$ and $\hat{u}_{\mathrm{B}}$ under $\hat{\mathcal{M}}$ on the horizontal axis. lie in the credible intervals for the first approximate fortnight after the end of the Democratic National Convention. One possible explanation for this phenomena is that, although the election does officially become a two-candidate contest at that time (notwithstanding our previous comments about third-party candidates), the effects of the Republican and Democratic primaries may take time to dissipate; unmodeled dynamics of noncooperative games in the presence of many candidates may still be dominant during this time. Finally, we display the inferred final conditions $\Phi_{\mathrm{R}}(x)$ and $\Phi_{\mathrm{B}}(x)$ in panel C of Fig. 4.13.

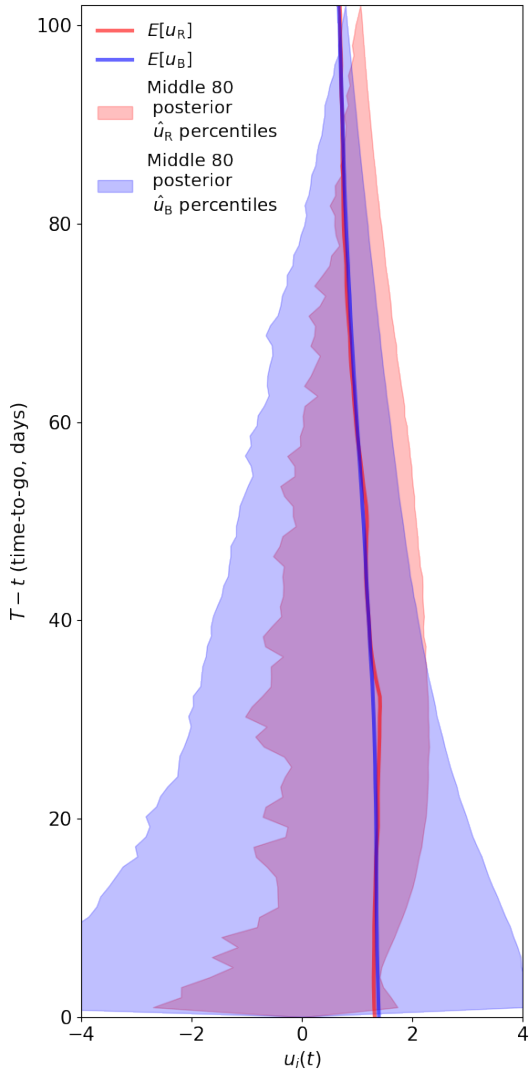

Figure 4.14: The mean latent Red and Blue control policies inferred in the context of $\mathcal{M}$ fall within the middle 80 percentiles of $\hat{\mathcal{M}}$ for almost the entire time period of study. The thick red and blue curves represent $E[u_R]$ and $E[u_B]$ respectively, while the upper and lower boundaries of the filled regions are the 10th and 90th percentiles of the respective probability distributions under $\hat{\mathcal{M}}$. During the first roughly 10 days after the Democratic National Convention (which occurred on 7/28/2016, or $T - t = 103$ on this plot), $E[u_R]$ or $E[u_B]$ fall outside of this credible interval.

Discussion and conclusion

We introduce, analyze, and numerically solve (analytically solve in simplified cases) a simple, first-principles model of noncooperative strategic interference by a foreign intelligence service from one country (Red) in an election occurring in another country (Blue) and attempts by Blue's domestic intelligence service to counter this interference. Though simple, our model is able to provide qualitative insight into the dynamics of such strategic interactions and performs well when fitted to polling and social media data surrounding the 2016 U.S. presidential election contest. We find that all-or-nothing attitudes regarding the outcome of the election interference (whether or not it was successful) with no gradation of utility, even if these attitudes are held by only one player, result in an arms race of spending on interference and counter-interference operations by both players. We then find analytical solutions to player i's optimal control problem in the case where player $\neg i$ credibly commits to a strategy $v(t)$ and detail an analytical value function approximation that can be used by player i even when player $\neg i$ does not commit to a particular strategy as long as player $\neg i$'s current strategy and its derivative can be estimated. We demonstrate the applicability of our model to real election interference scenarios by analyzing the Russian effort to interfere in the 2016 U.S. presidential election through observation of Russian Internet Research Agency (IRA) troll account posts on the website Twitter. Using this data, along with aggregate presidential election polling data, we infer the time series of Russian and U.S. control policies and find parameters of our model that best explain these inferred (latent) control policies. We show that, for most of the time under consideration, our model provides a good explanation for the inferred

variables.

There are several areas in which our work could be improved. From a theoretical point of view, our model is one of the simplest that can be proposed to model this situation. While from an *a priori* point of view it is derived using a minimum of assumption about the election mechanics, electorate, and cost (equivalently, utility) functions of the respective intelligence agencies and hence is justafiable on the grounds of parsimony and acceptable empirical performance (on at least one election contest), the kind of assumptions that we make are rather unrealistic. Though a pure random walk model for am election is not without serious precedent [234], a prudent extension of this work could incorporate non-interference-related state dynamics as a generalization of Eq. 4.3, e.g., as

$$\mathrm{d}x = [\mu_0 + \mu_1 x + u_{\mathrm{R}}(t) + u_{\mathrm{B}}(t)]\mathrm{d}t + \sigma \mathrm{d}\mathcal{W}. \tag{4.43}$$

This state equation can account for simple drift in the election results as a candidate endogenously becomes more or less popular or capture possible mean-reverting behavior in a hotly-contested race. Another interesting extension would introduce state-dependent running costs, particularly in the running cost of the Red player. Though the action of election interference is nominally intended to cause a particular candidate to win or lose, there are often other goals as well, such as undermining the Blue citizens' trust in their electoral process. Thus, Red might gain utility even just from having a particular candidate pull ahead in polls multiple times when that candidate would not have otherwise done so, even if the candidate does not actually win the election. In the context of our model, this can be represented by setting Red's

cost functional to be

$$E_{u_{\mathrm{R}},u_{\mathrm{B}},X}\Big\{\Phi_{\mathrm{R}}(X_T)+\int_0^T[-\Theta(-X_t) + u_{\mathrm{R}}^2(t)-\lambda_{\mathrm{R}}u_{\mathrm{B}}^2(t)]\,\mathrm{d}t\Big\}. \tag{4.44}$$

Both of these modifications are relatively easy to incorporate into the model and will not change the qualitative nature of Red and Blue's HJB equations since their effects will simply be to introduce an additional drift term (Eq. 4.43) or a continuous, non-differentiable source term (Eq. 4.44) into the HJB equations (Eqs. 4.11 and 4.12); the fundamental nature of these equations as nonlinear parabolic equations coupled through quadratic terms of self and other-player first spatial derivatives remains unchanged as these modifications to the theory do not introduce any new coupling terms. With the modification of Eq. 4.43, the HJB equations become

$$\begin{aligned} -\frac{\partial V_{\mathrm{R}}}{\partial t} &= (\mu_0+\mu_1 x)\frac{\partial V_{\mathrm{R}}}{\partial x}-\frac{1}{4}\left(\frac{\partial V_{\mathrm{R}}}{\partial x}\right)^2 \\ &\quad -\frac{1}{2}\frac{\partial V_{\mathrm{R}}}{\partial x}\frac{\partial V_{\mathrm{B}}}{\partial x}-\frac{\lambda_{\mathrm{R}}}{4}\left(\frac{\partial V_{\mathrm{B}}}{\partial x}\right)^2+\frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{R}}}{\partial x^2}, \\ V_{\mathrm{R}}(x,T) &= \Phi_{\mathrm{R}}(x) \end{aligned} \tag{4.45}$$

and

$$\begin{aligned} -\frac{\partial V_{\mathrm{B}}}{\partial t} &= (\mu_0+\mu_1 x)\frac{\partial V_{\mathrm{B}}}{\partial x}-\frac{1}{4}\left(\frac{\partial V_{\mathrm{B}}}{\partial x}\right)^2 \\ &\quad -\frac{1}{2}\frac{\partial V_{\mathrm{B}}}{\partial x}\frac{\partial V_{\mathrm{R}}}{\partial x}-\frac{\lambda_{\mathrm{B}}}{4}\left(\frac{\partial V_{\mathrm{R}}}{\partial x}\right)^2+\frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{B}}}{\partial x^2}, \\ V_{\mathrm{B}}(x,T) &= \Phi_{\mathrm{B}}(x), \end{aligned} \tag{4.46}$$

while with the modification of Eq. 4.44 Red's HJB equation reads

$$
\begin{aligned}
-\frac{\partial V_{\mathrm{R}}}{\partial t} &= -\frac{1}{4}\left(\frac{\partial V_{\mathrm{R}}}{\partial x}\right)^2 - \frac{1}{2}\frac{\partial V_{\mathrm{R}}}{\partial x}\frac{\partial V_{\mathrm{B}}}{\partial x} \\
&- \frac{\lambda_{\mathrm{R}}}{4}\left(\frac{\partial V_{\mathrm{B}}}{\partial x}\right)^2 - \Theta(-x) + \frac{\sigma^2}{2}\frac{\partial^2 V_{\mathrm{R}}}{\partial x^2}, \\
V_{\mathrm{R}}(x,T) &= \Phi_{\mathrm{R}}(x).
\end{aligned}
\tag{4.47}
$$

A more fundamental qualitative change would be to expand the scope of Red's interference to alter the latent volatility of the election process. For example, the objective of Red's interference operations might be not only to change the drift of the state equation to make it more likely for candidate A to win, but also to increase the uncertainty associated with the election's polling.

In addition to theoretical modifications, other work could simply extend the present results to other elections using similarly fine-grained data or, ideally, even more granular data. The principal difficulty with this approach lies in the inherent difficulty of finding any data at all with which to work. Though there do exist public datasets of election interference episodes [194], the characteristic timescale of this data is much longer than that used in our analysis. As we note in Sec. 4.3, we are able to confront our model to data only because the Russian interference in the 2016 U.S. presidential election was so well-publicized and because the interference took place at least partially through the mechanism of Twitter, a public data source. Even so, we found it necessary to infer the variables in which we were actually interested. Other than this event, we were unable to find any publicly-available data of sufficient temporal resolution for any other publicly-acknowledged election interference episode.

Bibliography

[1] Narasimhan Jegadeesh and Sheridan Titman. Profitability of momentum strategies: An evaluation of alternative explanations. *The Journal of finance*, 56(2):699–720, 2001.

[2] KB Athreya et al. Bootstrap of the mean in the infinite variance case. *The Annals of Statistics*, 15(2):724–731, 1987.

[3] Jan Solomon Cramer. Mean and variance of r2 in small and moderate samples. *Journal of econometrics*, 35(2-3):253–266, 1987.

[4] Mark L Carrodus and David EA Giles. The exact distribution of r2 when the regression disturbances are autocorrelated. *Economics Letters*, 38(4):375–380, 1992.

[5] M Ángeles Serrano, Marián Boguná, and Alessandro Vespignani. Extracting the multiscale backbone of complex weighted networks. *Proceedings of the national academy of sciences*, 106(16):6483–6488, 2009.

[6] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[7] Herbert A Simon. Rational choice and the structure of the environment. *Psychological review*, 63(2):129, 1956.

[8] Allen Newell, John Calman Shaw, and Herbert A Simon. Elements of a theory of human problem solving. *Psychological review*, 65(3):151, 1958.

[9] Amos Tversky and Daniel Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, 1981.

[10] Gerd Gigerenzer and Daniel G Goldstein. Reasoning the fast and frugal way: models of bounded rationality. *Psychological review*, 103(4):650, 1996.

[11] Andreu Mas-Colell, Michael Dennis Whinston, and Jerry R Green. *Microeconomic theory*, volume 1. Oxford University Press, 1995.

[12] Thomas Hobbes. *Leviathan*. 1651.

[13] Brian F Tivnan, David Rushing Dewhurst, Colin M Van Oort, IV Ring, H John, Tyler J Gray, Brendan F Tivnan, Matthew TK Koehler, Matthew T McMahon, David Slater, et al. Fragmentation and inefficiencies in us equity markets: Evidence from the dow 30. *arXiv preprint arXiv:1902.04690*, 2019.

[14] Eric Budish, Peter Cramton, and John Shim. Implementation details for frequent batch auctions: Slowing down markets to the blink of an eye. *American Economic Review*, 104(5):418–24, 2014.

[15] Eric Budish, Peter Cramton, and John Shim. The high-frequency trading arms race: Frequent batch auctions as a market design response. *The Quarterly Journal of Economics*, 130(4):1547–1621, 2015.

[16] Burton G Malkiel and Eugene F Fama. Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, 25(2):383–417, 1970.

[17] Narasimhan Jegadeesh and Sheridan Titman. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of finance*, 48(1):65–91, 1993.

[18] Andrew W Lo. The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. *SSRN*, 2004.

[19] Andrew W Lo. Reconciling efficient markets with behavioral finance: the adaptive markets hypothesis. 2005.

[20] Jae H Kim, Abul Shamsuddin, and Kian-Ping Lim. Stock return predictability and the adaptive markets hypothesis: Evidence from century-long us data. *Journal of Empirical Finance*, 18(5):868–879, 2011.

[21] J Doyne Farmer and Andrew W Lo. Frontiers of finance: Evolution and efficient markets. *Proceedings of the National Academy of Sciences*, 96(18):9991–9992,

1999.

[22] Neil Johnson, Guannan Zhao, Eric Hunsader, Jing Meng, Amith Ravindar, Spencer Carran, and Brian Tivnan. Financial black swans driven by ultrafast machine ecology. *arXiv preprint arXiv:1202.1448*, 2012.

[23] Reginald Smith. Is high-frequency trading inducing changes in market microstructure and dynamics? 2010.

[24] Albert J Menkveld. High frequency trading and the new market makers. *Journal of Financial Markets*, 16(4):712–740, 2013.

[25] Blake LeBaron. Agent-based computational finance: Suggested readings and early research. *Journal of Economic Dynamics and Control*, 24(5-7):679–702, 2000.

[26] Robert Savit, Radu Manuca, and Rick Riolo. Adaptive competition, market efficiency, and phase transitions. *Physical Review Letters*, 82(10):2203, 1999.

[27] Cars H Hommes. Modeling the stylized facts in finance through simple nonlinear adaptive systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7221–7228, 2002.

[28] Yi-Cheng Zhang. Modeling market mechanism with evolutionary games. *arXiv preprint cond-mat/9803308*, 1998.

[29] Kanta Kinoshita, Kyoko Suzuki, and Tetsuya Shimokawa. Evolutionary foundation of bounded rationality in a financial market. *IEEE Transactions on Evolutionary Computation*, 17(4):528–544, 2013.

[30] Cars Hommes and Florian Wagener. Complex evolutionary systems in behavioral finance. In *Handbook of financial markets: Dynamics and evolution*, pages 217–276. Elsevier, 2009.

[31] George A Akerlof. The market for ‚Äúlemons‚Äù: Quality uncertainty and the market mechanism. In *Uncertainty in Economics*, pages 235–251. Elsevier, 1978.

[32] Oliver Kim and Robert E Verrecchia. Market reaction to anticipated announcements. *Journal of Financial Economics*, 30(2):273–309, 1991.

[33] Dhananjay K Gode and Shyam Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of political economy*, 101(1):119–137, 1993.

[34] J Doyne Farmer, Paolo Patelli, and Ilija I Zovko. The predictive power of zero intelligence in financial markets. *Proceedings of the National Academy of Sciences*, 102(6):2254–2259, 2005.

[35] David Cushing. Automated batch auctions in conjunction with continuous financial markets, September 10 2013. US Patent 8,533,100.

[36] Elaine Wah, Dylan Hurd, and Michael P Wellman. Strategic market choice: Frequent call markets vs. continuous double auctions for fast and slow traders. *EAI Endorsed Trans. Serious Games*, 3(10):e1, 2016.

[37] Chris J Muscarella and Michael S Piwowar. Market microstructure and securities values:: Evidence from the paris bourse. *Journal of Financial Markets*, 4(3):209–229, 2001.

[38] Maureen O'Hara and Mao Ye. Is market fragmentation harming market quality? *Journal of Financial Economics*, 100(3):459–474, 2011.

[39] Tobias Blickle and Lothar Thiele. A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation*, 4(4):361–394, 1996.

[40] L Darrell Whitley et al. The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *ICGA*, volume 89, pages 116–123. Fairfax, VA, 1989.

[41] Benoit B Mandelbrot. The variation of certain speculative prices. In *Fractals and scaling in finance*, pages 371–418. Springer, 1997.

[42] Rosario N Mantegna and H Eugene Stanley. Econophysics: Scaling and its breakdown in finance. *Journal of statistical Physics*, 89(1-2):469–479, 1997.

[43] Rama Cont, Marc Potters, and Jean-Philippe Bouchaud. Scaling in stock market data: stable laws and beyond. In *Scale invariance and beyond*, pages 75–85. Springer, 1997.

[44] Anna Carbone, Giuliano Castelli, and H Eugene Stanley. Time-dependent hurst

exponent in financial time series. *Physica A: Statistical Mechanics and its Applications*, 344(1-2):267–271, 2004.

[45] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[46] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[47] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[48] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[49] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[50] Shu-Heng Chen, Ren-Jie Zeng, and Tina Yu. Co-evolving trading strategies to analyze bounded rationality in double auction markets. In *Genetic programming theory and practice VI*, pages 1–19. Springer, 2009.

[51] David Garcia and Frank Schweitzer. Social signals and algorithmic trading of bitcoin. *Royal Society open science*, 2(9):150288, 2015.

[52] Anton Golub, James B Glattfelder, and Richard B Olsen. The alpha engine: designing an automated trading algorithm. In *High-Performance Computing in Finance*, pages 49–76. Chapman and Hall/CRC, 2018.

[53] Amy Greenwald. The 2002 trading agent competition: An overview of agent strategies. *AI Magazine*, 24(1):83–83, 2003.

[54] Clive Davidson. A dark knight for algos. *Risk*, 25(9):32, 2012.

[55] Andrei A Kirilenko and Andrew W Lo. Moore's law versus murphy's law: Algorithmic trading and its discontents. *Journal of Economic Perspectives*, 27(2):51–72, 2013.

[56] Robert Kissell and Roberto Malamut. Understanding the profit and loss distribution of trading algorithms. *Trading*, 2005(1):41–49, 2005.

[57] Michael AH Dempster and Chris M Jones. A real-time adaptive trading system using genetic programming. *Quantitative Finance*, 1(4):397–413, 2001.

[58] Mark P Austin, Graham Bates, Michael AH Dempster 3, Vasco Leemans, and Stacy N Williams. Adaptive systems for foreign exchange trading. *Quantitative Finance*, 4(4):37–45, 2004.

[59] Blake LeBaron, W Brian Arthur, and Richard Palmer. Time series properties of an artificial stock market. *Journal of Economic Dynamics and control*, 23(9-10):1487–1516, 1999.

[60] RG Palmer, W Brian Arthur, John H Holland, and Blake LeBaron. An artificial stock market. *Artificial Life and Robotics*, 3(1):27–31, 1999.

[61] Blake LeBaron. Building the santa fe artificial stock market. *Physica A*, 2002.

[62] Shu-Heng Chen and Chia-Hsuan Yeh. Evolving traders and the business school with genetic programming: A new architecture of the agent-based artificial stock market. *Journal of Economic Dynamics and Control*, 25(3-4):363–393, 2001.

[63] Silvano Cincotti, Sergio M Focardi, Michele Marchesi, and Marco Raberto. Who wins? study of long-run trader survival in an artificial stock market. *Physica A: Statistical Mechanics and its Applications*, 324(1-2):227–233, 2003.

[64] Rama Cont. Volatility clustering in financial markets: empirical facts and agent-based models. In *Long memory in economics*, pages 289–309. Springer, 2007.

[65] Serafin Martinez-Jaramillo and Edward PK Tsang. An heterogeneous, endogenous and coevolutionary gp-based financial market. *IEEE Transactions on Evolutionary Computation*, 13(1):33–55, 2009.

[66] Thomas Philip Runarsson and Simon M Lucas. Coevolution versus self-play temporal difference learning for acquiring position evaluation in small-board go. *IEEE Transactions on Evolutionary Computation*, 9(6):628–640, 2005.

[67] Natalia Akchurina. Multiagent reinforcement learning: algorithm converging to nash equilibrium in general-sum discounted stochastic games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 725–732. Citeseer, 2009.

[68] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning*, pages 805–813, 2015.

[69] Paul R Milgrom and Robert J Weber. A theory of auctions and competitive bidding. *Econometrica: Journal of the Econometric Society*, pages 1089–1122, 1982.

[70] R Preston McAfee and John McMillan. Auctions and bidding. *Journal of economic literature*, 25(2):699–738, 1987.

[71] Lawrence M Ausubel. An efficient ascending-bid auction for multiple objects. *American Economic Review*, 94(5):1452–1475, 2004.

[72] Carole Comerton-Forde and James Rydge. The influence of call auction algorithm rules on market efficiency. *Journal of Financial Markets*, 9(2):199–222, 2006.

[73] Louis KC Chan, Narasimhan Jegadeesh, and Josef Lakonishok. Momentum strategies. *The Journal of Finance*, 51(5):1681–1713, 1996.

[74] Swaminathan G Badrinath and Sunil Wahal. Momentum trading by institutions. *The Journal of Finance*, 57(6):2449–2478, 2002.

[75] Werner FM De Bondt and Richard H Thaler. Anomalies: A mean-reverting walk down wall street. *Journal of Economic Perspectives*, 3(1):189–202, 1989.

[76] Hanqin Zhang and Qing Zhang. Trading a mean-reverting asset: Buy low and sell high. *Automatica*, 44(6):1511–1518, 2008.

[77] Maureen O'Hara and George S Oldfield. The microeconomics of market making.

Journal of Financial and Quantitative analysis, 21(4):361–376, 1986.

[78] Sanmay Das. The effects of market-making on price dynamics. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, pages 887–894. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[79] Andrei Kirilenko, Albert S Kyle, Mehrdad Samadi, and Tugkan Tuzun. The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, 72(3):967–998, 2017.

[80] Eric M Aldrich and K López Vargas. Experiments in high-frequency trading: Testing the frequent batch auction. *Experimental Economics*, 2018.

[81] Lawrence E Harris and Venkatesh Panchapagesan. The information content of the limit order book: evidence from nyse specialist trading decisions. *Journal of Financial Markets*, 8(1):25–67, 2005.

[82] Charles Cao, Oliver Hansch, and Xiaoxin Wang. The information content of an open limit-order book. *Journal of Futures Markets: Futures, Options, and Other Derivative Products*, 29(1):16–41, 2009.

[83] Gheorghe Cosmin Silaghi and Valentin Robu. An agent strategy for automated stock market trading combining price and order book information. In *2005 ICSC Congress on Computational Intelligence Methods and Applications*, pages 4–pp. IEEE, 2005.

[84] Jaakko Aspara and Arvid OI Hoffmann. Cut your losses and let your profits run: How shifting feelings of personal responsibility reverses the disposition effect. *Journal of Behavioral and Experimental Finance*, 8:18–24, 2015.

[85] Carol Clark et al. Controlling risk in a lightning-speed trading environment. *Chicago Fed Letter*, 272, 2010.

[86] Didier Sornette and Susanne von der Becke. Crashes and high frequency trading: An evaluation of risks posed by high-speed algorithmic trading. *The Future of Computer Trading in Financial Markets*, 2011.

[87] Bernard S Donefer. Algos gone wild: Risk in the world of automated trading strategies. *The Journal of Trading*, 5(2):31–34, 2010.

[88] Adam Ponzi and Yoji Aizawa. Evolutionary financial market models. *Physica A: Statistical Mechanics and its Applications*, 287(3-4):507–523, 2000.

[89] Marco Bartolozzi and Anthony William Thomas. Stochastic cellular automata model for stock market dynamics. *Physical review E*, 69(4):046112, 2004.

[90] Fredrick Michael and MD Johnson. Financial market dynamics. *Physica A: Statistical Mechanics and its Applications*, 320:525–534, 2003.

[91] AAG Cortines and R Riera. Non-extensive behavior of a stock market index at microscopic time scales. *Physica A: Statistical Mechanics and its Applications*, 377(1):181–192, 2007.

[92] Sandhya Devi. Financial market dynamics: superdiffusive or not? *Journal of Statistical Mechanics: Theory and Experiment*, 2017(8):083207, 2017.

[93] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654, 1973.

[94] Jimmy E Hilliard, Jeff Madura, and Alan L Tucker. Currency option pricing with stochastic domestic and foreign interest rates. *Journal of Financial and Quantitative Analysis*, 26(2):139–151, 1991.

[95] Constantine Sandis and Nassim Taleb. The skin in the game heuristic for protection against tail events. *Review of Behavioral Economics*, 2014.

[96] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

[97] Pimwadee Chaovalit, Aryya Gangopadhyay, George Karabatis, and Zhiyuan Chen. Discrete wavelet transform-based time series analysis and mining. *ACM Computing Surveys (CSUR)*, 43(2):6, 2011.

[98] Chin-Chia Michael Yeh, Nickolas Kavantzas, and Eamonn Keogh. Matrix profile vi: Meaningful multidimensional motif discovery. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 565–574. IEEE, 2017.

[99] Yan Zhu, Makoto Imamura, Daniel Nikovski, and Eamonn Keogh. Introducing time series chains: a new primitive for time series data mining. *Knowledge and*

Information Systems, pages 1–27, 2018.

[100] Zbigniew R Struzik and Arno PJM Siebes. Wavelet transform based multifractal formalism in outlier detection and localisation for financial time series. *Physica A: Statistical Mechanics and its Applications*, 309(3-4):388–402, 2002.

[101] Ivan Popivanov and Renee J Miller. Similarity search over time-series data using wavelets. In *Proceedings 18th international conference on data engineering*, pages 212–221. IEEE, 2002.

[102] K-M Lau and Hengyi Weng. Climate signal detection using wavelet transform: How to make a time series sing. *Bulletin of the American meteorological society*, 76(12):2391–2402, 1995.

[103] Brandon Whitcher, Simon D Byers, Peter Guttorp, and Donald B Percival. Testing for homogeneity of variance in time series: Long memory, wavelets, and the nile river. *Water Resources Research*, 38(5), 2002.

[104] R Benítez, VJ Bolós, and ME Ramírez. A wavelet-based tool for studying non-periodicity. *Computers & Mathematics with Applications*, 60(3):634–641, 2010.

[105] Steve Mann and Simon Haykin. The chirplet transform: a generalization of gabor's logon transform. In *Vision Interface*, volume 91, pages 205–212, 1991.

[106] Genyuan Wang, Xiang-Gen Xia, Benjamin T Root, and Victor C Chen. Moving target detection in over-the-horizon radar using adaptive chirplet transform. In *Proceedings of the 2002 IEEE Radar Conference (IEEE Cat. No. 02CH37322)*, pages 77–84. IEEE, 2002.

[107] PD Spanos, A Giaralis, and NP Politis. Time–frequency representation of earthquake accelerograms and inelastic structural response records using the adaptive chirplet decomposition and empirical mode decomposition. *Soil Dynamics and Earthquake Engineering*, 27(7):675–689, 2007.

[108] A Taebi and HA Mansy. Effect of noise on time-frequency analysis of vibrocardiographic signals. *Journal of bioengineering & biomedical science*, 6(4), 2016.

[109] ES Page. A test for a change in a parameter occurring at an unknown point. *Biometrika*, 42(3/4):523–527, 1955.

[110] Stephane Mallat and Wen Liang Hwang. Singularity detection and processing with wavelets. *IEEE transactions on information theory*, 38(2):617–643, 1992.

[111] Peter Sheridan Dodds, Kameron Decker Harris, Isabel M Kloumann, Catherine A Bliss, and Christopher M Danforth. Temporal patterns of happiness and information in a global social network: Hedonometrics and twitter. *PLOS one*, 6(12):e26752, 2011.

[112] Quanzhi Li, Sameena Shah, Merine Thomas, Kajsa Anderson, Xiaomo Liu, Armineh Nourbakhsh, and Rui Fang. How much data do you need? twitter decahose data analysis. 2017.

[113] Andrew J Reagan, Christopher M Danforth, Brian Tivnan, Jake Ryland Williams, and Peter Sheridan Dodds. Sentiment analysis methods for understanding large-scale texts: a case for using continuum-scored words and word shift graphs. *EPJ Data Science*, 6(1):28, 2017.

[114] Andrew G Reece, Andrew J Reagan, Katharina LM Lix, Peter Sheridan Dodds, Christopher M Danforth, and Ellen J Langer. Forecasting the onset and course of mental illness with twitter data. *Scientific reports*, 7(1):13006, 2017.

[115] Morgan R Frank, Lewis Mitchell, Peter Sheridan Dodds, and Christopher M Danforth. Happiness and the patterns of life: A study of geolocated tweets. *Scientific reports*, 3:2625, 2013.

[116] Lewis Mitchell, Morgan R Frank, Kameron Decker Harris, Peter Sheridan Dodds, and Christopher M Danforth. The geography of happiness: Connecting twitter sentiment and expression, demographics, and objective characteristics of place. *PloS one*, 8(5):e64417, 2013.

[117] Rupert Lemahieu, Steven Van Canneyt, Cedric De Boom, and Bart Dhoedt. Optimizing the popularity of twitter messages through user categories. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1396–1401. IEEE, 2015.

[118] Fang Wu and Bernardo A Huberman. Novelty and collective attention. *Proceedings of the National Academy of Sciences*, 104(45):17599–17601, 2007.

[119] Cristian Candia, C Jara-Figueroa, Carlos Rodriguez-Sickert, Albert-László Barabási, and César A Hidalgo. The universal decay of collective memory

and attention. *Nature Human Behaviour*, 3(1):82, 2019.

[120] Riley Crane and Didier Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences*, 105(41):15649–15653, 2008.

[121] Philipp Lorenz-Spreen, Bjarke Mørch Mønsted, Philipp Hövel, and Sune Lehmann. Accelerating dynamics of collective attention. *Nature communications*, 10(1):1759, 2019.

[122] Manlio De Domenico and Eduardo G Altmann. Unraveling the origin of social bursts in collective attention. *arXiv preprint arXiv:1903.06588*, 2019.

[123] Glenn Ierley and Alex Kostinski. A universal rank-order transform to extract signals from noisy data. *arXiv preprint arXiv:1906.08729*, 2019.

[124] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[125] Aamna Al Shehhi, Mayada Oudah, and Zeyar Aung. Investigating factors behind choosing a cryptocurrency. In *2014 IEEE International Conference on Industrial Engineering and Engineering Management*, pages 1443–1447. IEEE, 2014.

[126] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[127] Kiyoung Yang and Cyrus Shahabi. An efficient k nearest neighbor search for multivariate time series. *Information and Computation*, 205(1):65–98, 2007.

[128] David C Kale, Dian Gong, Zhengping Che, Yan Liu, Gerard Medioni, Randall Wetzel, and Patrick Ross. An examination of multivariate time series hashing with applications to health care. In *2014 IEEE International Conference on Data Mining*, pages 260–269. IEEE, 2014.

[129] Anne Driemel and Francesco Silvestri. Locality-sensitive hashing of curves. *arXiv preprint arXiv:1703.04040*, 2017.

[130] Eamonn J Keogh and Michael J Pazzani. A simple dimensionality reduction technique for fast similarity search in large time series databases. In *Pacific-Asia*

conference on knowledge discovery and data mining, pages 122–133. Springer, 2000.

[131] Yi-Leh Wu, Divyakant Agrawal, and Amr El Abbadi. A comparison of dft and dwt based similarity search in time-series databases. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 488–495. ACM, 2000.

[132] FK-P Chan, AW-C Fu, and Clement Yu. Haar wavelets for efficient similarity search of time-series: with and without time warping. *IEEE Transactions on knowledge and data engineering*, 15(3):686–705, 2003.

[133] Chotirat Ratanamahatana, Eamonn Keogh, Anthony J Bagnall, and Stefano Lonardi. A novel bit level time series representation with implication of similarity search and clustering. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 771–777. Springer, 2005.

[134] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. Ieee, 2005.

[135] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 1317–1322. IEEE, 2016.

[136] J Ronald Eastman and Michele Fulk. Long sequence time series evaluation using standardized principal components. *Photogrammetric Engineering and remote sensing*, 59(6), 1993.

[137] David Harris. Principal components analysis of cointegrated time series. *Econometric Theory*, 13(4):529–557, 1997.

[138] Joseph Ryan G Lansangan and Erniel B Barrios. Principal components analysis of nonstationary time series data. *Statistics and Computing*, 19(2):173, 2009.

[139] Abdullah Mueen, Krishnamurthy Viswanathan, Chetan Gupta, and Eamonn Keogh. The fastest similarity search algorithm for time series subsequences under euclidean distance, august 2017.

[140] Onur Seref, Ya-Ju Fan, and Wanpracha Art Chaovalitwongse. Mathematical programming formulations and algorithms for discrete k-median clustering of time-series data. *INFORMS Journal on Computing*, 26(1):160–172, 2013.

[141] Michail Vlachos, Jessica Lin, Eamonn Keogh, and Dimitrios Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In *In proc. workshop on clustering high dimensionality data and its applications*. Citeseer, 2003.

[142] Cyril Goutte, Peter Toft, Egill Rostrup, Finn Å Nielsen, and Lars Kai Hansen. On clustering fmri time series. *NeuroImage*, 9(3):298–310, 1999.

[143] Daxin Jiang, Jian Pei, and Aidong Zhang. Dhc: a density-based hierarchical clustering method for time series gene expression data. In *Third IEEE Symposium on Bioinformatics and Bioengineering, 2003. Proceedings.*, pages 393–400. IEEE, 2003.

[144] Pedro Pereira Rodrigues, Joao Gama, and Joao Pedro Pedroso. Odac: Hierarchical clustering of time series data streams. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 499–503. SIAM, 2006.

[145] Pedro Pereira Rodrigues, João Gama, and Joao Pedroso. Hierarchical clustering of time-series data streams. *IEEE transactions on knowledge and data engineering*, 20(5):615–627, 2008.

[146] Anne Denton. Kernel-density-based clustering of time series subsequences using a continuous random-walk noise model. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[147] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[148] Mete Çelik, Filiz Dadaşer-Çelik, and Ahmet Şakir Dokuz. Anomaly detection in temperature data using dbscan algorithm. In *2011 International Symposium on Innovations in Intelligent Systems and Applications*, pages 91–95. IEEE, 2011.

[149] Mahesh Kumar, Nitin R Patel, and Jonathan Woo. Clustering seasonality patterns in the presence of errors. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 557–563. ACM, 2002.

[150] Tim Oates, Laura Firoiu, and Paul R Cohen. Clustering time series with hidden markov models and dynamic time warping. In *Proceedings of the IJCAI-99 workshop on neural, symbolic and reinforcement learning methods for sequence learning*, pages 17–21. Citeseer, 1999.

[151] Thomas Schreiber and Andreas Schmitz. Classification of time series data with nonlinear similarity measures. *Physical review letters*, 79(8):1475, 1997.

[152] Konstantinos Kalpakis, Dhiral Gada, and Vasundhara Puttagunta. Distance measures for effective clustering of arima time-series. In *Proceedings 2001 IEEE international conference on data mining*, pages 273–280. IEEE, 2001.

[153] Anthony J Bagnall and Gareth J Janacek. Clustering time series from arma models with clipped data. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 49–58. ACM, 2004.

[154] Yimin Xiong and Dit-Yan Yeung. Time series clustering with arma mixtures. *Pattern Recognition*, 37(8):1675–1689, 2004.

[155] Sylvia Fröhwirth-Schnatter and Sylvia Kaufmann. Model-based clustering of multiple time series. *Journal of Business & Economic Statistics*, 26(1):78–89, 2008.

[156] Eamonn Keogh and Jessica Lin. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems*, 8(2):154–177, 2005.

[157] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[158] Anna C Gilbert, Yannis Kotidis, Shanmugavelayutham Muthukrishnan, and Martin Strauss. Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In *Vldb*, volume 1, pages 79–88, 2001.

[159] Saif Ahmad, Tugba Taskaya-Temizel, and Khurshid Ahmad. Summarizing time series: Learning patterns in ‚Äòvolatile‚Äôseries. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 523–532. Springer,

2004.

[160] Rita Castillo-Ortega, Nicolás Marín, Daniel Sánchez, and Andrea GB Tettamanzi. A multi-objective memetic algorithm for the linguistic summarization of time series. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, pages 171–172. ACM, 2011.

[161] Rita Castillo Ortega, Nicolás Marín, Daniel Sánchez, and Andrea GB Tettamanzi. Linguistic summarization of time series data using genetic algorithms. In *EUSFLAT*, volume 1, pages 416–423. Atlantis Press, 2011.

[162] Janusz Kacprzyk, Anna Wilbik, and Sławomir Zadrożny. Linguistic summarization of time series under different granulation of describing features. In *International Conference on Rough Sets and Intelligent Systems Paradigms*, pages 230–240. Springer, 2007.

[163] Janusz Kacprzyk, Anna Wilbik, and S Zadrożny. Linguistic summarization of time series using a fuzzy quantifier driven aggregation. *Fuzzy Sets and Systems*, 159(12):1485–1499, 2008.

[164] Janusz Kacprzyk, Anna Wilbik, and Sławomir Zadrożny. An approach to the linguistic summarization of time series using a fuzzy quantifier driven aggregation. *International Journal of Intelligent Systems*, 25(5):411–439, 2010.

[165] Lei Li, James McCann, Nancy S Pollard, and Christos Faloutsos. Dynammo: Mining and summarization of coevolving sequences with missing values. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 507–516. ACM, 2009.

[166] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[167] G Gbur, TD Visser, and E Wolf. Anomalous behavior of spectra near phase singularities of focused waves. *Physical review letters*, 88(1):013901, 2001.

[168] Vasiliki Plerou, Parameswaran Gopikrishnan, Luís A Nunes Amaral, Xavier Gabaix, and H Eugene Stanley. Economic fluctuations and anomalous diffusion. *Physical Review E*, 62(3):R3023, 2000.

[169] Jae-Hyung Jeon, Vincent Tejedor, Stas Burov, Eli Barkai, Christine Selhuber-

Unkel, Kirstine Berg-Sørensen, Lene Oddershede, and Ralf Metzler. In vivo anomalous diffusion and weak ergodicity breaking of lipid granules. *Physical review letters*, 106(4):048103, 2011.

[170] Thomas R Palfrey and Jeffrey E Prisbrey. Anomalous behavior in public goods experiments: how much and why? *The American Economic Review*, pages 829–846, 1997.

[171] C Monica Capra, Jacob K Goeree, Rosario Gomez, and Charles A Holt. Anomalous behavior in a traveler's dilemma? *American Economic Review*, 89(3):678–690, 1999.

[172] Bernard Rosner. Percentage points for a generalized esd many-outlier procedure. *Technometrics*, 25(2):165–172, 1983.

[173] Owen Vallis, Jordan Hochenbaum, and Arun Kejariwal. A novel technique for long-term anomaly detection in the cloud. In *6th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 14)*, 2014.

[174] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1939–1947. ACM, 2015.

[175] Philip K Chan and Matthew V Mahoney. Modeling multiple time series for anomaly detection. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[176] Haibin Cheng, Pang-Ning Tan, Christopher Potter, and Steven Klooster. Detection and characterization of anomalies in multivariate time series. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 413–424. SIAM, 2009.

[177] Huida Qiu, Yan Liu, Niranjan A Subrahmanya, and Weichang Li. Granger causality for time-series anomaly detection. In *2012 IEEE 12th international conference on data mining*, pages 1074–1079. IEEE, 2012.

[178] Hermine N Akouemo and Richard J Povinelli. Probabilistic anomaly detection in natural gas time series data. *International Journal of Forecasting*, 32(3):948–956, 2016.

[179] Marcelle Chauvet. An econometric characterization of business cycle dynamics with factor structure and regime switching. *International economic review*, pages 969–996, 1998.

[180] Michael Dueker. Dynamic forecasts of qualitative variables: a qual var model of us recessions. *Journal of Business & Economic Statistics*, 23(1):96–104, 2005.

[181] Pär Österholm. The limited usefulness of macroeconomic bayesian vars when estimating the probability of a us recession. *Journal of Macroeconomics*, 34(1):76–86, 2012.

[182] James D Hamilton and Gang Lin. Stock market volatility and the business cycle. *Journal of applied econometrics*, 11(5):573–593, 1996.

[183] Arturo Estrella and Frederic S Mishkin. Predicting us recessions: Financial variables as leading indicators. *Review of Economics and Statistics*, 80(1):45–61, 1998.

[184] Min Qi. Predicting us recessions with leading indicators via neural network models. *International Journal of Forecasting*, 17(3):383–401, 2001.

[185] Travis J Berge. Predicting recessions with leading indicators: Model averaging and selection over the business cycle. *Journal of Forecasting*, 34(6):455–471, 2015.

[186] A O'hagan and Tom Leonard. Bayes estimation subject to uncertainty about parameter constraints. *Biometrika*, 63(1):201–203, 1976.

[187] Jon Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[188] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[189] Paul SA Renaud and Frans AAM van Winden. On the importance of elections and ideology for government policy in a multi-party system. In *The logic of multiparty systems*, pages 191–207. Springer, 1987.

[190] Jorgen Elklit and Palle Svensson. What makes elections free and fair? *Journal of democracy*, 8(3):32–46, 1997.

[191] Dov H Levin. When the great power gets a vote: The effects of great power electoral interventions on election results. *International Studies Quarterly*, 60(2):189–202, 2016.

[192] Stephen Shulman and Stephen Bloom. The legitimacy of foreign intervention in elections: the ukrainian response. *Review of International Studies*, 38(2):445–471, 2012.

[193] Daniel Corstange and Nikolay Marinov. Taking sides in other people,Äôs elections: The polarizing effect of foreign intervention. *American Journal of Political Science*, 56(3):655–670, 2012.

[194] Dov H Levin. Partisan electoral interventions by the great powers: Introducing the PEIG Dataset. *Conflict Management and Peace Science*, 36(1):88–106, 2019.

[195] Erica D Borghard and Shawn W Lonergan. Confidence Building Measures for the Cyber Domain. *Strategic Studies Quarterly*, 12(3):10–49, 2018.

[196] Dov H Levin. Voting for Trouble? Partisan Electoral Interventions and Domestic Terrorism. *Terrorism and Political Violence*, pages 1–17, 2018.

[197] Isabella Hansen and Darren J Lim. Doxing democracy: influencing elections via cyber voter interference. *Contemporary Politics*, 25(2):150–171, 2019.

[198] Johannes Bubeck, Kai Jäger, Nikolay Marinov, and Federico Nanni. Why Do States Intervene in the Elections of Others? *Available at SSRN 3435138*, 2019.

[199] Read the mueller report: Searchable document and index. *New York Times*, Apr 2019.

[200] Crispin W Gardiner. *Handbook of Stochastic Methods*, volume 3. Springer Berlin, 1985.

[201] Peter Jung and Peter Hänggi. Dynamical systems: a unified colored-noise approximation. *Physical Review A*, 35(10):4464, 1987.

[202] Peter Hänggi and Peter Jung. Colored noise in dynamical systems. *Advances in Chemical Physics*, 89:239–326, 1995.

[203] Richard Bellman. Dynamic programming and a new formalism in the calculus of variations. *Proceedings of the National Academy of Sciences of the United States of America*, 40(4):231, 1954.

[204] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[205] Rainer Buckdahn and Juan Li. Stochastic differential games and viscosity solutions of Hamilton–Jacobi–Bellman–Isaacs equations. *SIAM Journal on Control and Optimization*, 47(1):444–475, 2008.

[206] Triet Pham and Jianfeng Zhang. Two person zero-sum game in weak formulation and path dependent Bellman–Isaacs equation. *SIAM Journal on Control and Optimization*, 52(4):2090–2121, 2014.

[207] Michael G Crandall, Hitoshi Ishii, and Pierre-Louis Lions. User,Äôs guide to viscosity solutions of second order partial differential equations. *Bulletin of the American mathematical society*, 27(1):1–67, 1992.

[208] Hilbert J Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of statistical mechanics: theory and experiment*, 2005(11):P11011, 2005.

[209] Mark Kac. On distributions of certain wiener functionals. *Transactions of the American Mathematical Society*, 65(1):1–13, 1949.

[210] Yaming Chen, Adrian Baule, Hugo Touchette, and Wolfram Just. Weak-noise limit of a piecewise-smooth stochastic differential equation. *Physical Review E*, 88(5):052103, 2013.

[211] Julie Leifeld, Kaitlin Hill, and Andrew Roberts. Persistence of saddle behavior in the nonsmooth limit of smooth dynamical systems. *arXiv preprint arXiv:1504.04671*, 2015.

[212] Kiyoshi Kanazawa, Tomohiko G Sano, Takahiro Sagawa, and Hisao Hayakawa. Asymptotic derivation of Langevin-like equation with non-gaussian noise and its analytical solution. *Journal of Statistical Physics*, 160(5):1294–1335, 2015.

[213] Sofia H Piltz, Lauri Harhanen, Mason A Porter, and Philip K Maini. Inferring parameters of prey switching in a 1 predator–2 prey plankton system with a linear preference tradeoff. *Journal of Theoretical Biology*, 456:108–122, 2018.

[214] Peter Tanchak. The invisible front: Russia, trolls, and the information war against ukraine. *Revolution and War in Contemporary Ukraine: The Challenge of Change*, 161:253, 2017.

[215] Brandon C Boatwright, Darren L Linvill, and Patrick L Warren. Troll factories: The internet research agency and state-sponsored agenda building. *Resource Centre on Media Freedom in Europe*, 2018.

[216] Adam Badawy, Emilio Ferrara, and Kristina Lerman. Analyzing the digital traces of political manipulation: the 2016 Russian interference Twitter campaign. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 258–265. IEEE, 2018.

[217] Ryan L Boyd, Alexander Spangher, Adam Fourney, Besmira Nushi, Gireeja Ranade, James Pennebaker, and Eric Horvitz. Characterizing the internet research agency,Äôs social media operations during the 2016 us presidential election using linguistic analyses. 2018.

[218] Damian J Ruck, Natalie M Rice, Joshua Borycz, and R Alexander Bentley. Internet Research Agency Twitter activity predicted 2016 US election polls. *First Monday*, 24(7), 2019.

[219] Nicolas Guenon des Mesnards and Tauhid Zaman. Detecting influence campaigns in social networks using the Ising model. *arXiv preprint arXiv:1805.10244*, 2018.

[220] Jane Im, Eshwar Chandrasekharan, Jackson Sargent, Paige Lighthammer, Taylor Denby, Ankit Bhargava, Libby Hemphill, David Jurgens, and Eric Gilbert. Still out there: Modeling and Identifying Russian Troll Accounts on Twitter. *arXiv preprint arXiv:1901.11162*, 2019.

[221] Alexandria Volkening, Daniel F Linder, Mason A Porter, and Grzegorz A Rempala. Forecasting elections using compartmental models of infections. *arXiv preprint arXiv:1811.01831*, 2018.

[222] David Rothschild. Forecasting elections: Comparing prediction markets, polls,

and their biases. *Public Opinion Quarterly*, 73(5):895–916, 2009.

[223] Drew A Linzer. Dynamic Bayesian forecasting of presidential elections in the states. *Journal of the American Statistical Association*, 108(501):124–134, 2013.

[224] Wei Wang, David Rothschild, Sharad Goel, and Andrew Gelman. Forecasting elections with non-representative polls. *International Journal of Forecasting*, 31(3):980–991, 2015.

[225] Ramin Skibba. Pollsters struggle to explain failures of us presidential forecasts. *Nature News*, 539(7629):339, 2016.

[226] Ryan Neville-Shepard. Constrained by duality: Third-party master narratives in the 2016 presidential election. *American Behavioral Scientist*, 61(4):414–427, 2017.

[227] David Barber, A Taylan Cemgil, and Silvia Chiappa. *Bayesian Time Series Models*. Cambridge University Press, 2011.

[228] Matthew D Hoffman and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

[229] Andrew Gelman, Donald B Rubin, et al. Inference from iterative simulation using multiple sequences. *Statistical science*, 7(4):457–472, 1992.

[230] Stephen P Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455, 1998.

[231] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[232] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[233] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[234] Nassim Nicholas Taleb. Election predictions as martingales: an arbitrage approach. *Quantitative Finance*, 18(1):1–5, 2018.

[235] David Rushing Dewhurst, Michael Vincent Arnold, and Colin Michael Van Oort. Selection mechanisms affect volatility in evolving markets. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 90–98. ACM, 2019.

[236] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[237] Wenwen Dou, Xiaoyu Wang, Remco Chang, and William Ribarsky. Paralleltopics: A probabilistic approach to exploring document collections. In *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 231–240. IEEE, 2011.

Appendix A

Tables

Tables referenced in Chapter 3

Classification	Cusp shape	Words
Type I	Slow buildup, fast relaxation	rumble veterans dusty labour scattered hampshire #tinychat elected ballot selection labor entering beam phenomenon voters mamma anonymity republican #nowplaying indictment wages conservatives pulse knee grammy essays #tcot kentucky fml netherlands jingle valid whitman syracuse dems deposit bail tomb walker reader
Type II	Fast buildup, slow relaxation	xbox chained yale bombing holocaust connecticut #tinychat civilian jill turkish tsunami ferry #letsbehonest beam agreement riley ethics phenomenon harriet privacy israeli #nowplaying gun dub pulse killings herman enormous fbi dmc searched norman joan affected arthur sandra radiation army walker reader
Type III	Roughly symmetric	rumble memorial sleigh veterans costumes greeks britney separated father's shark grammys labor costume x-mas bunny commonwealth clause olympics olympic daylight cyber wrapping rudolph drowned re-election

Table A.1: Words for which at least one cusp segment was close in norm to a spatial mean cusp segment as detailed in Section 3.3. We display the distributions of "cusp points"—hypothesized deterministic maxima of the noisy mechanistically-generated time series—in Fig. 3.13.

Appendix B

Supplementary Figures

Figures referenced in Ch 4.

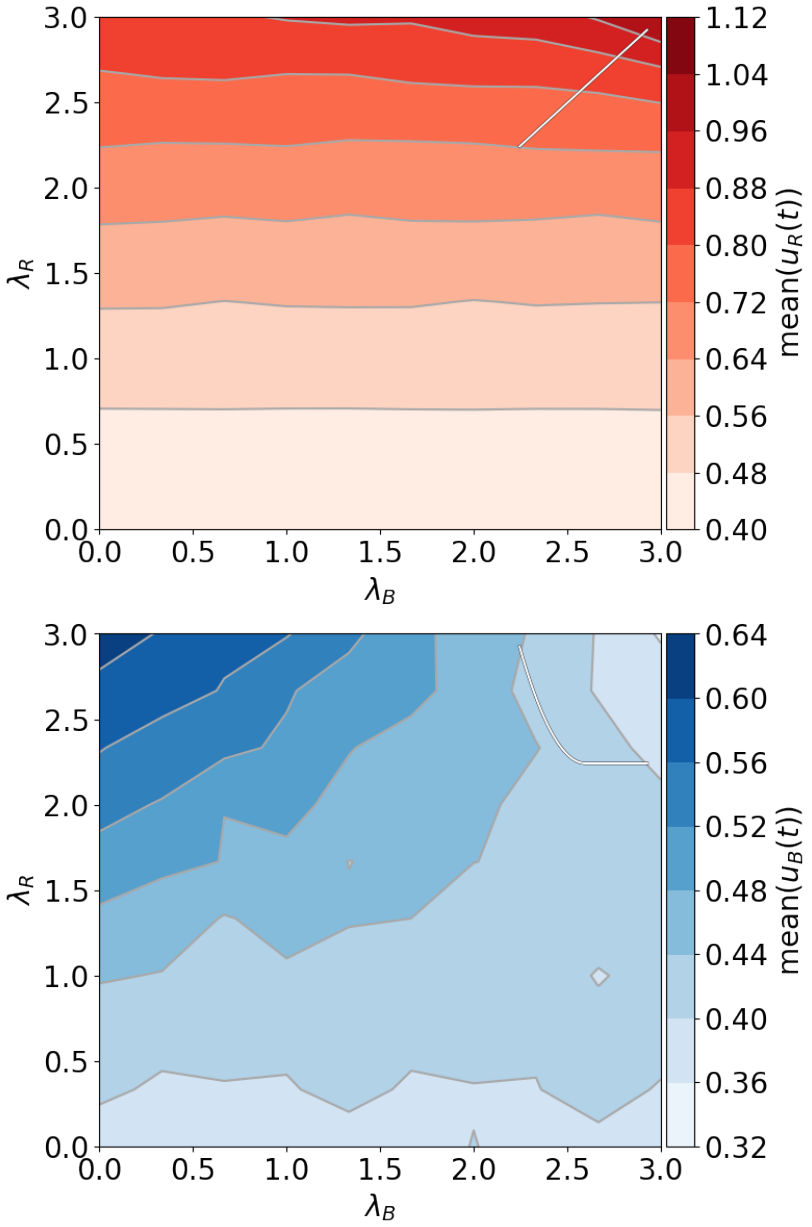

Figure B.1: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to mean of control policy.

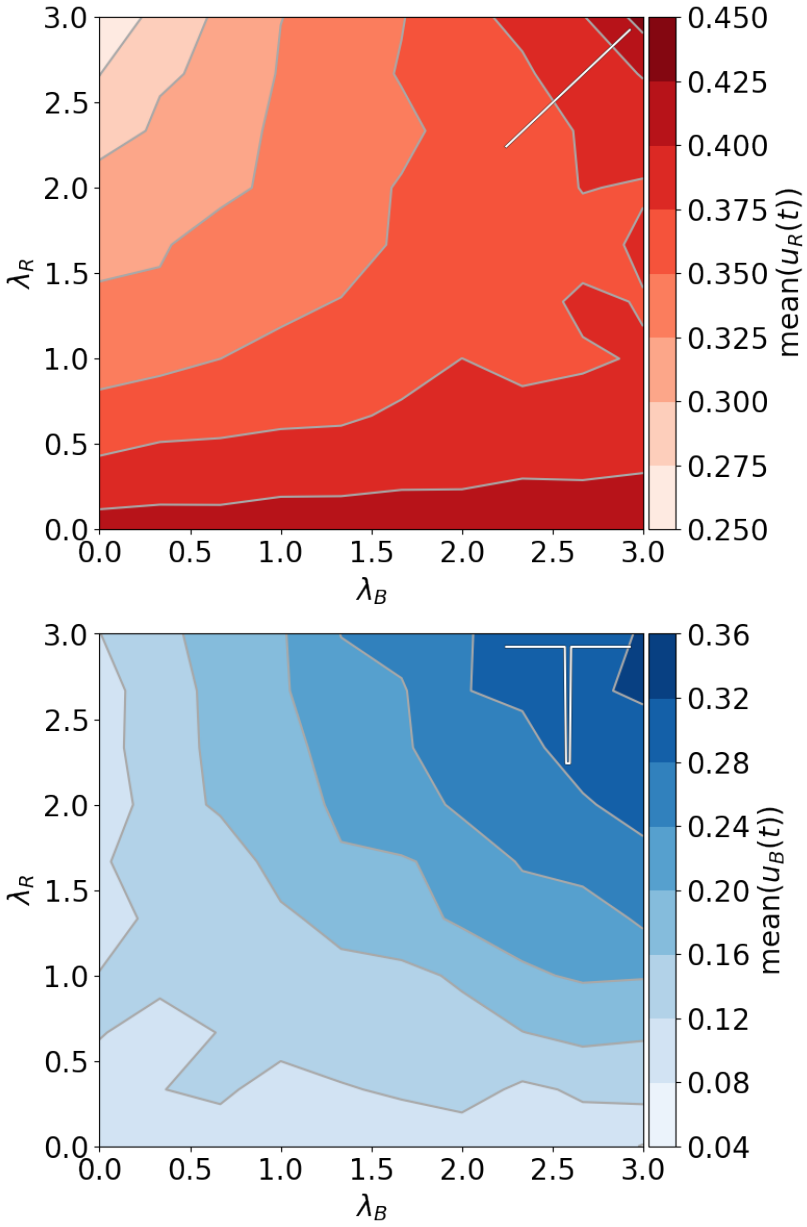

Figure B.2: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. Intensity of color corresponds to mean of control policy.

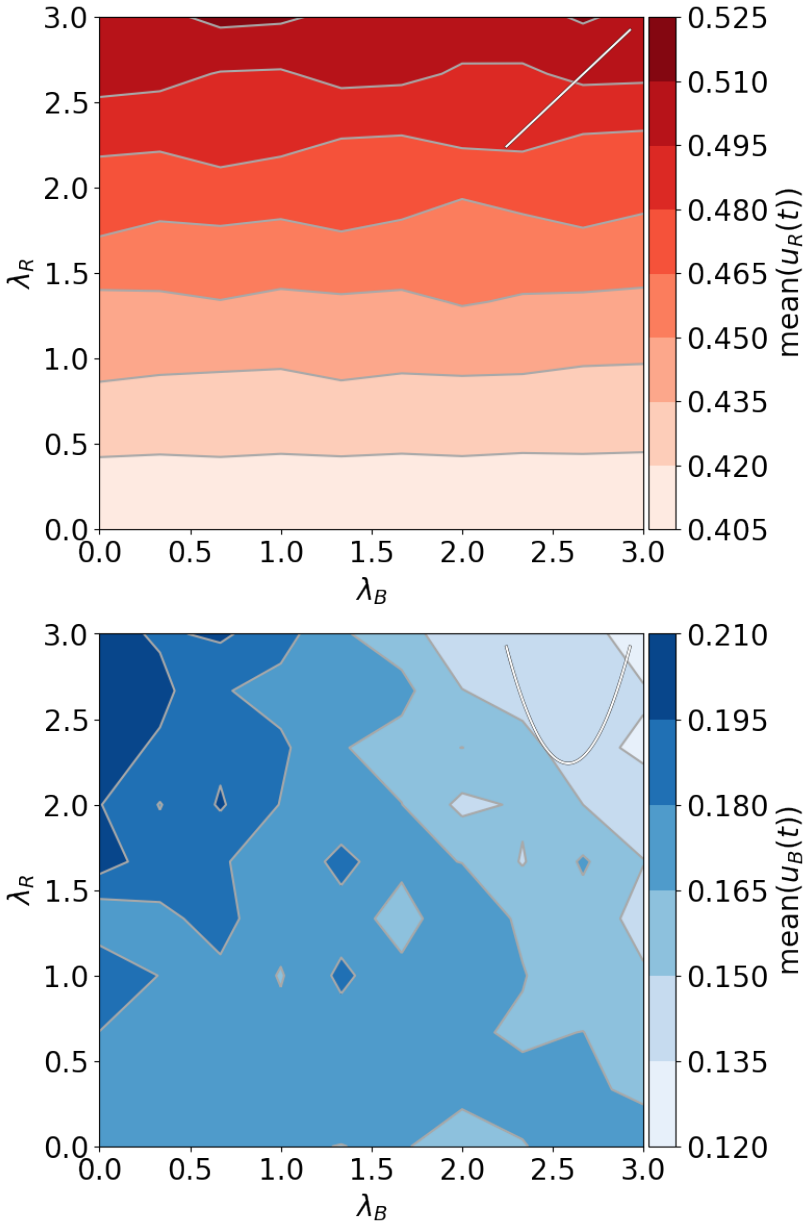

Figure B.3: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to mean of control policy.

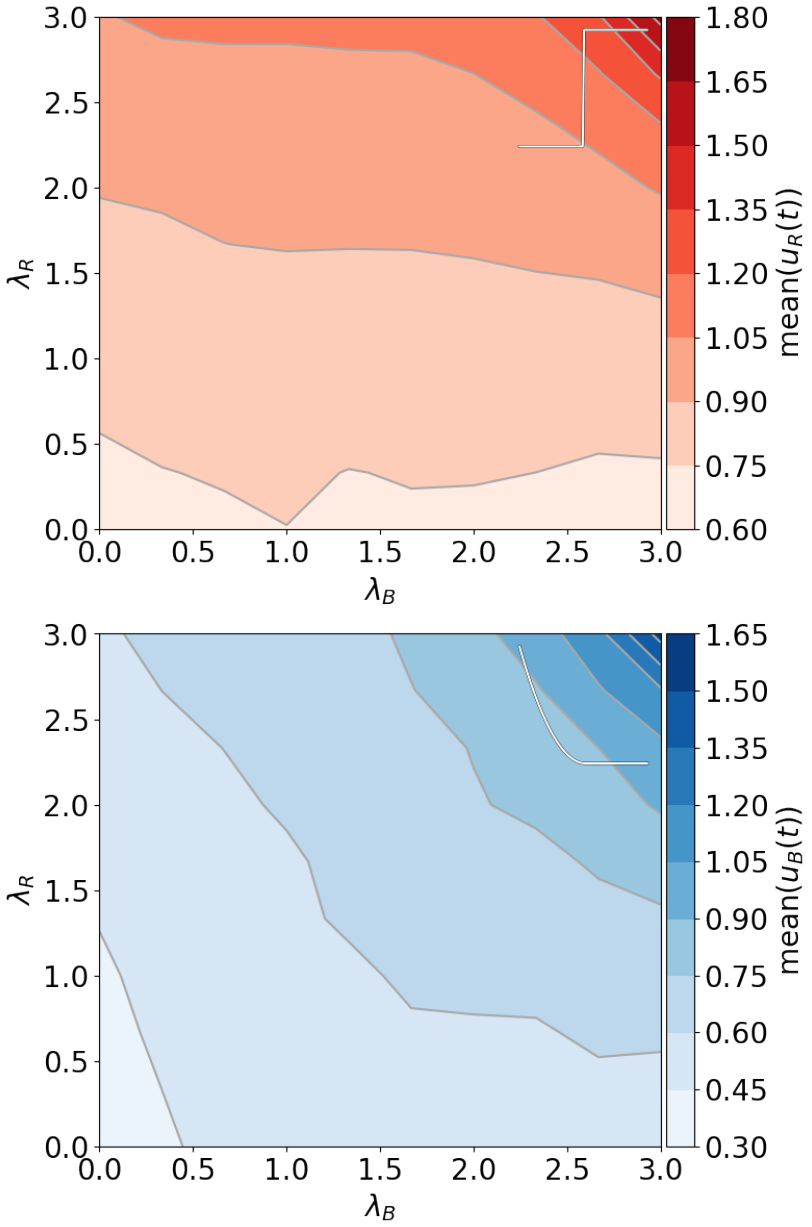

Figure B.4: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to mean of control policy.

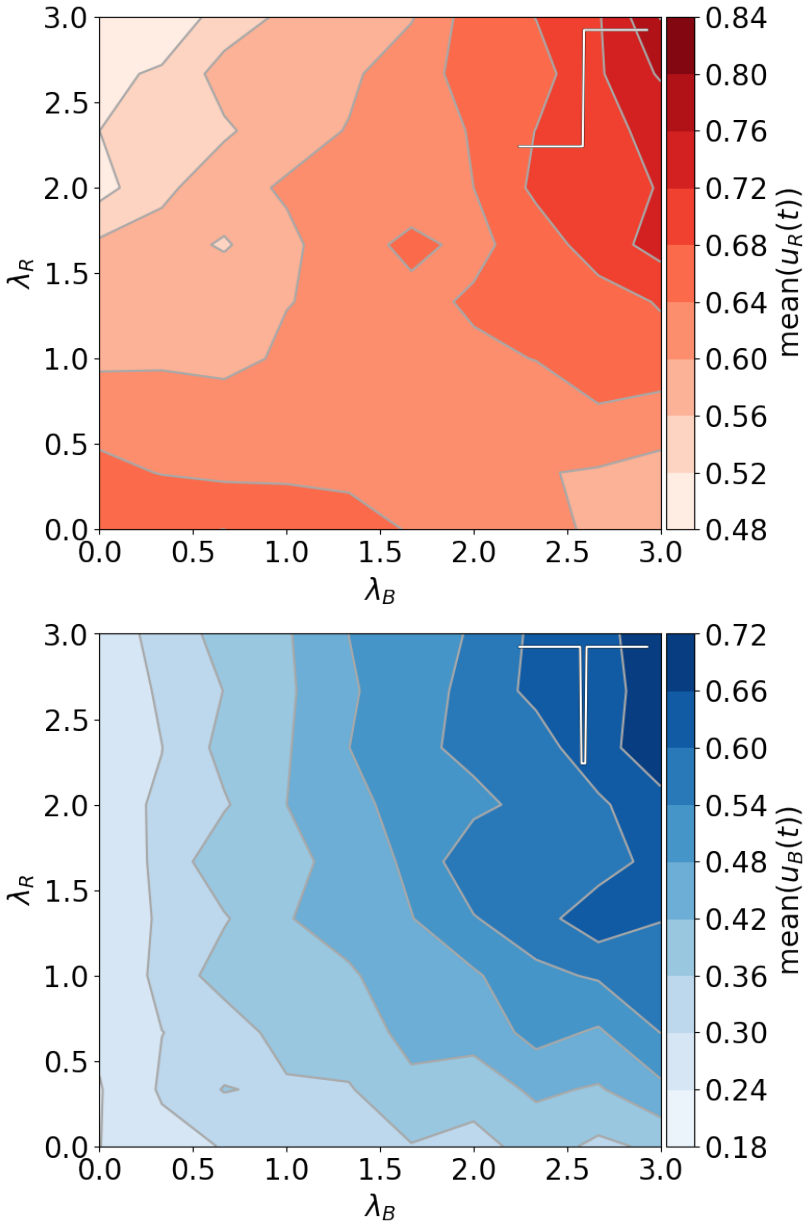

Figure B.5: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. Intensity of color corresponds to mean of control policy.

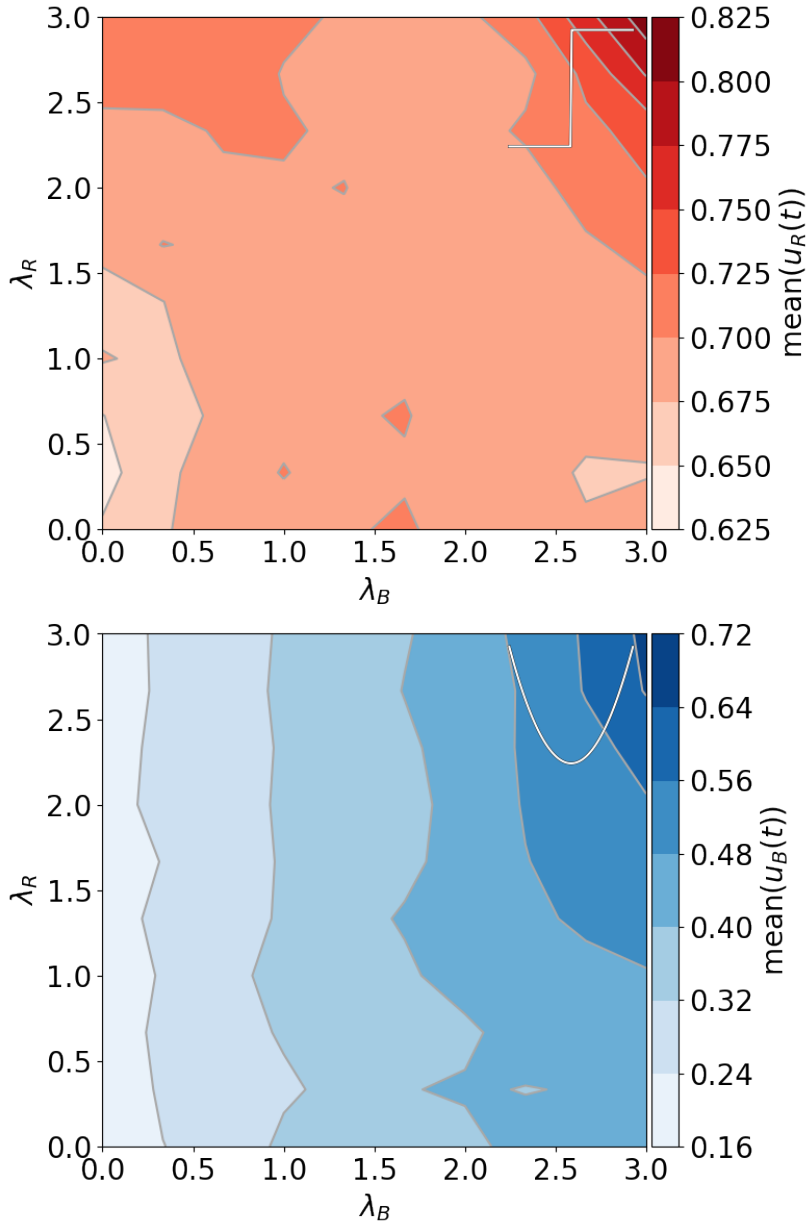

Figure B.6: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to mean of control policy.

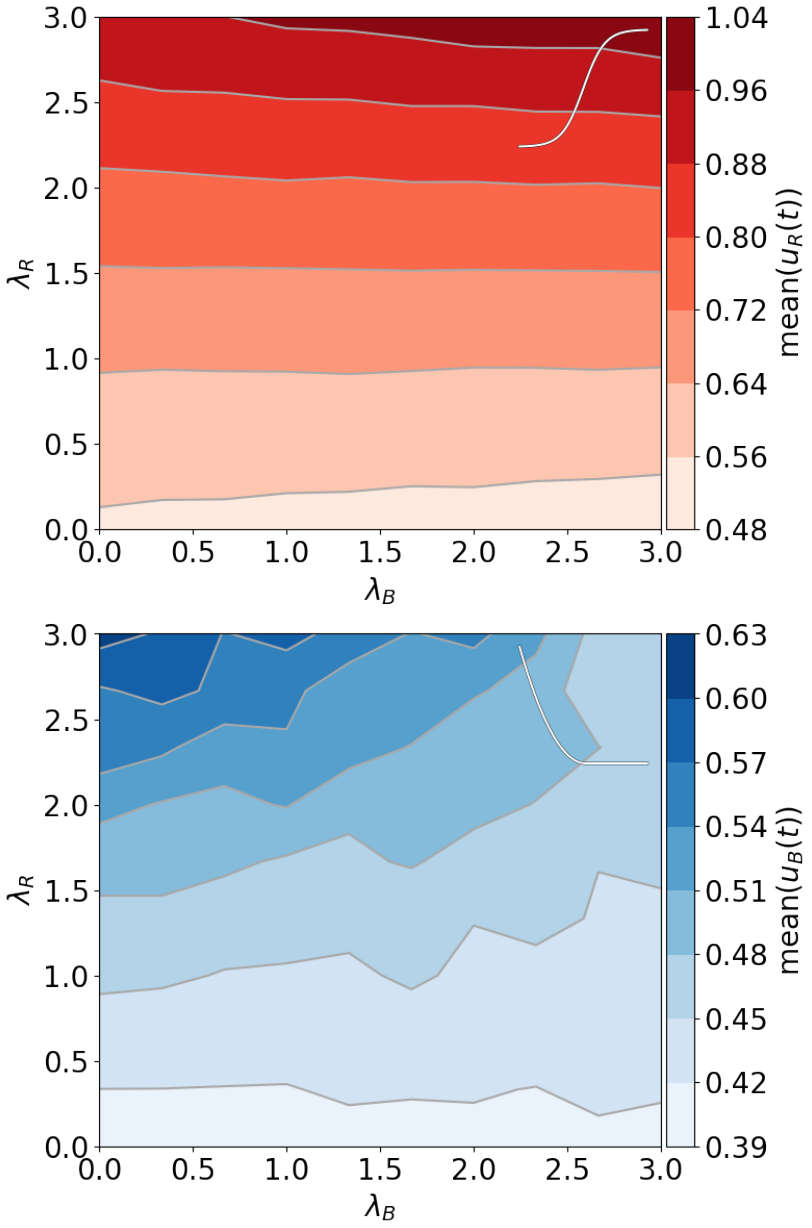

Figure B.7: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = \tanh(x)$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to mean of control policy.

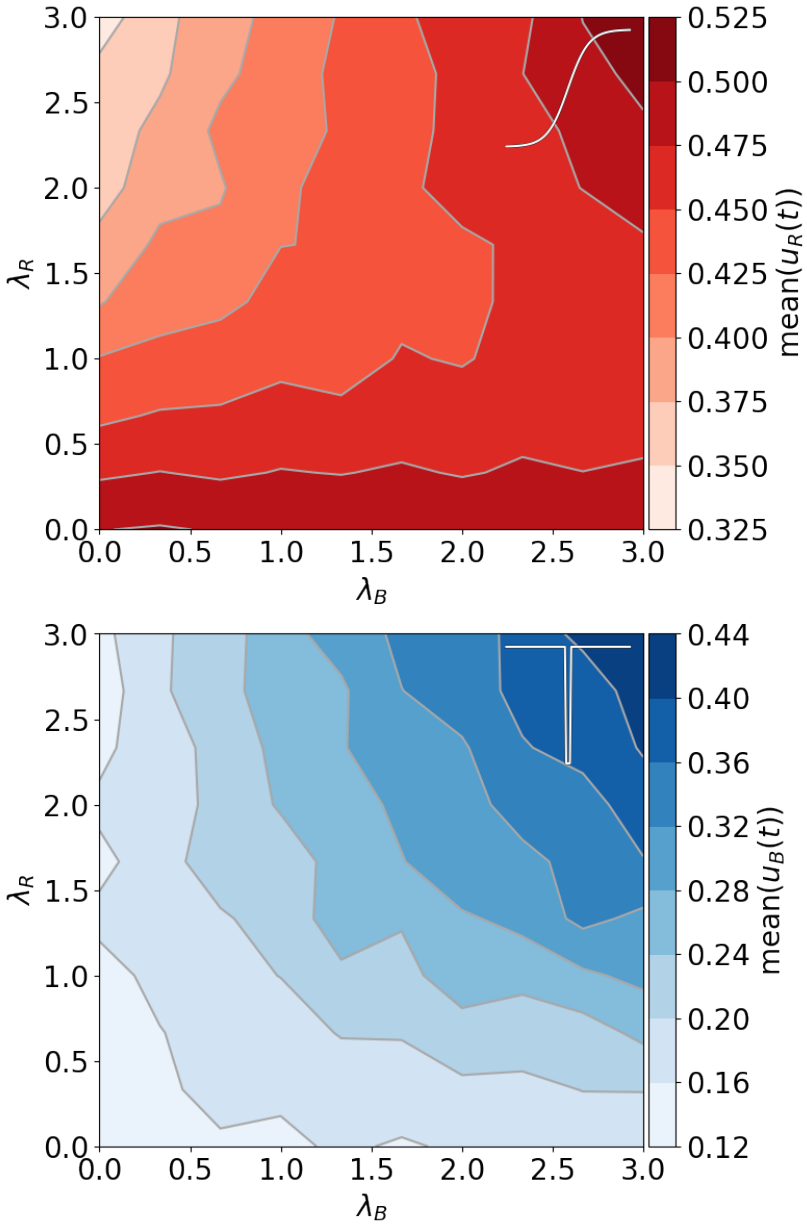

Figure B.8: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = \tanh(x)$ and Blue final condition $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. Intensity of color corresponds to mean of control policy.

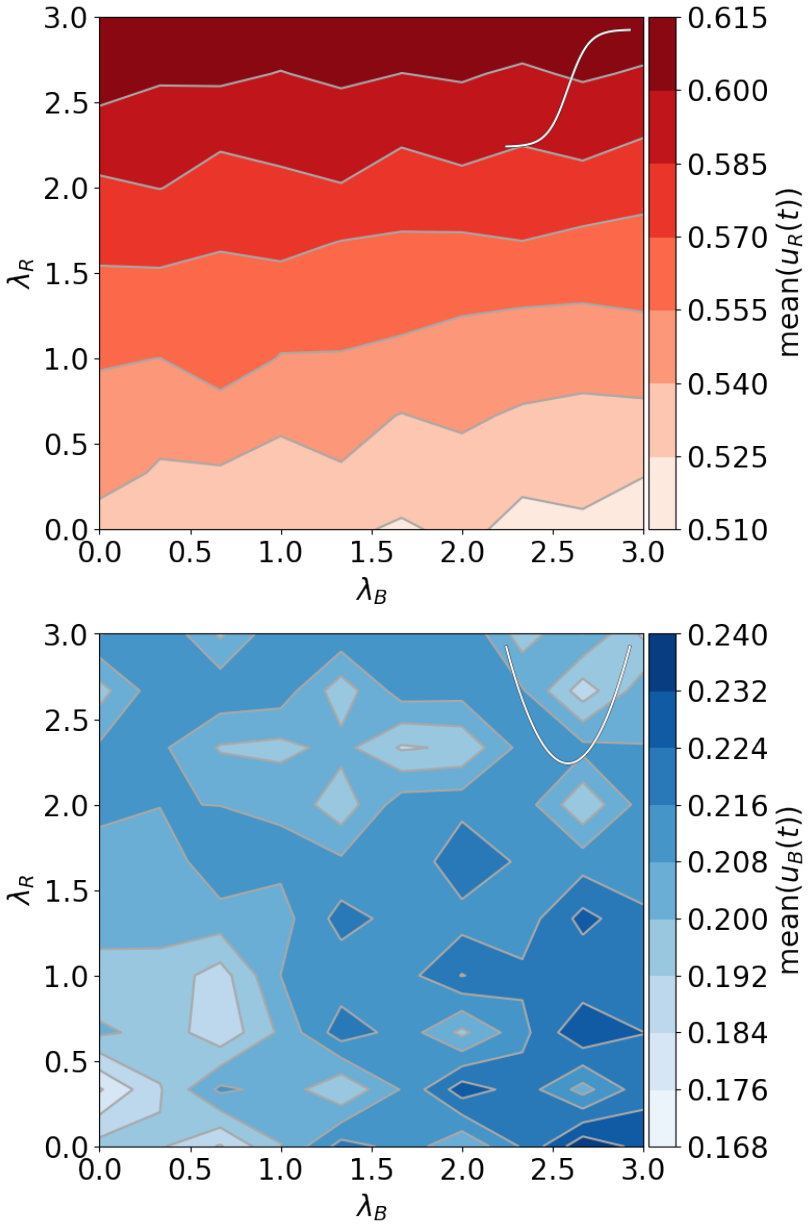

Figure B.9: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = \tanh(x)$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to mean of control policy.

We now present analogous figures for the standard deviations instead of the mean.

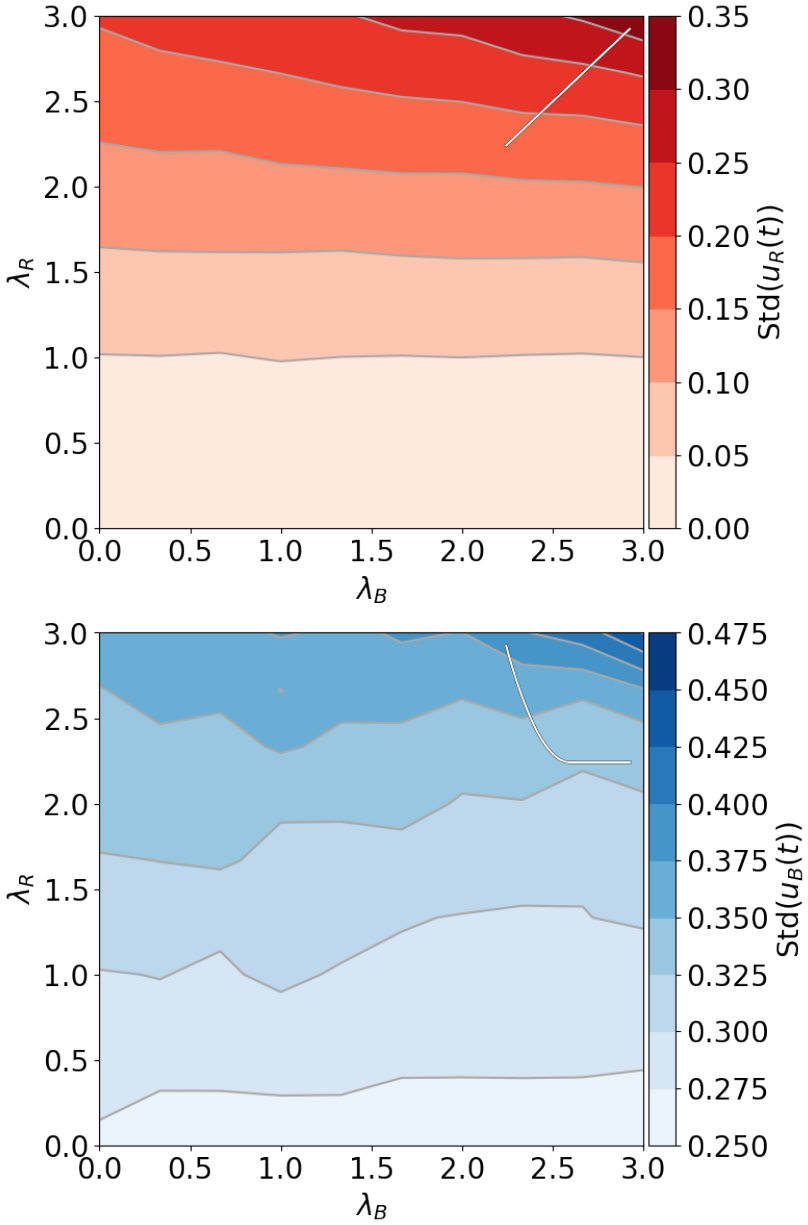

Figure B.10: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to std of control policy.

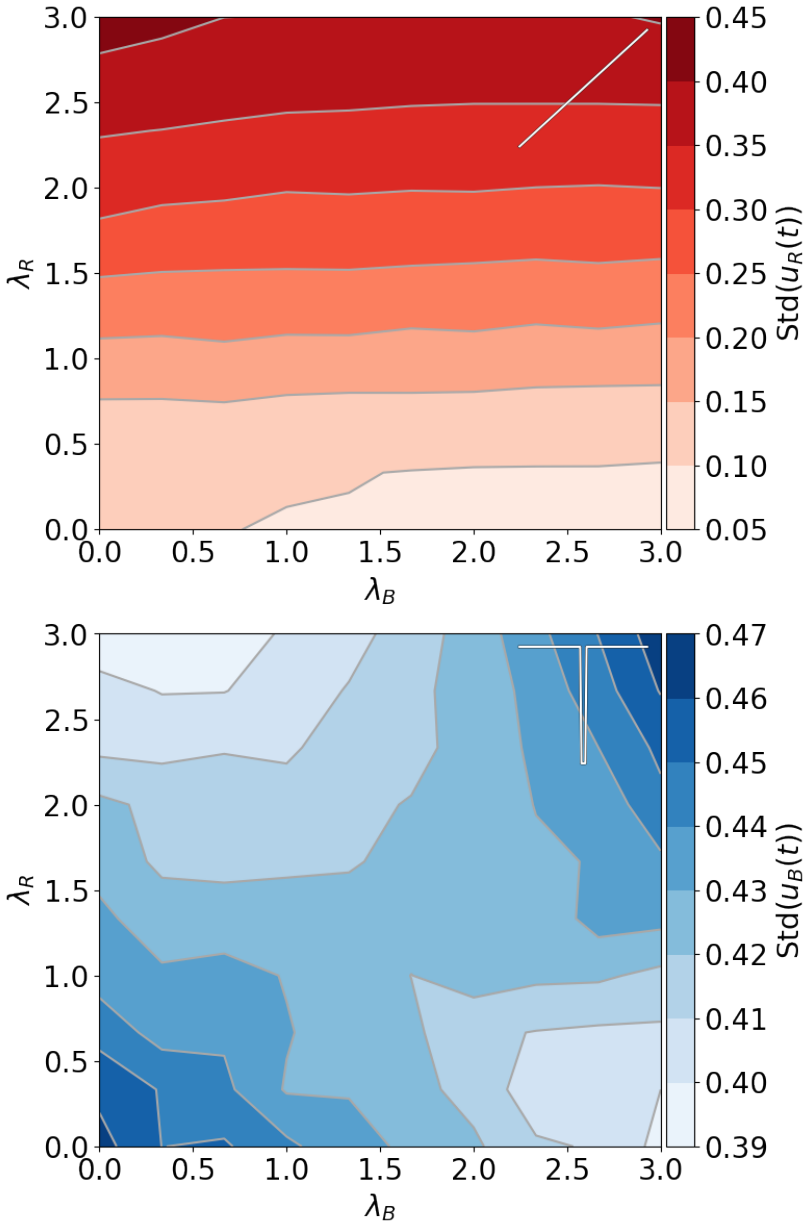

Figure B.11: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. Intensity of color corresponds to std of control policy.

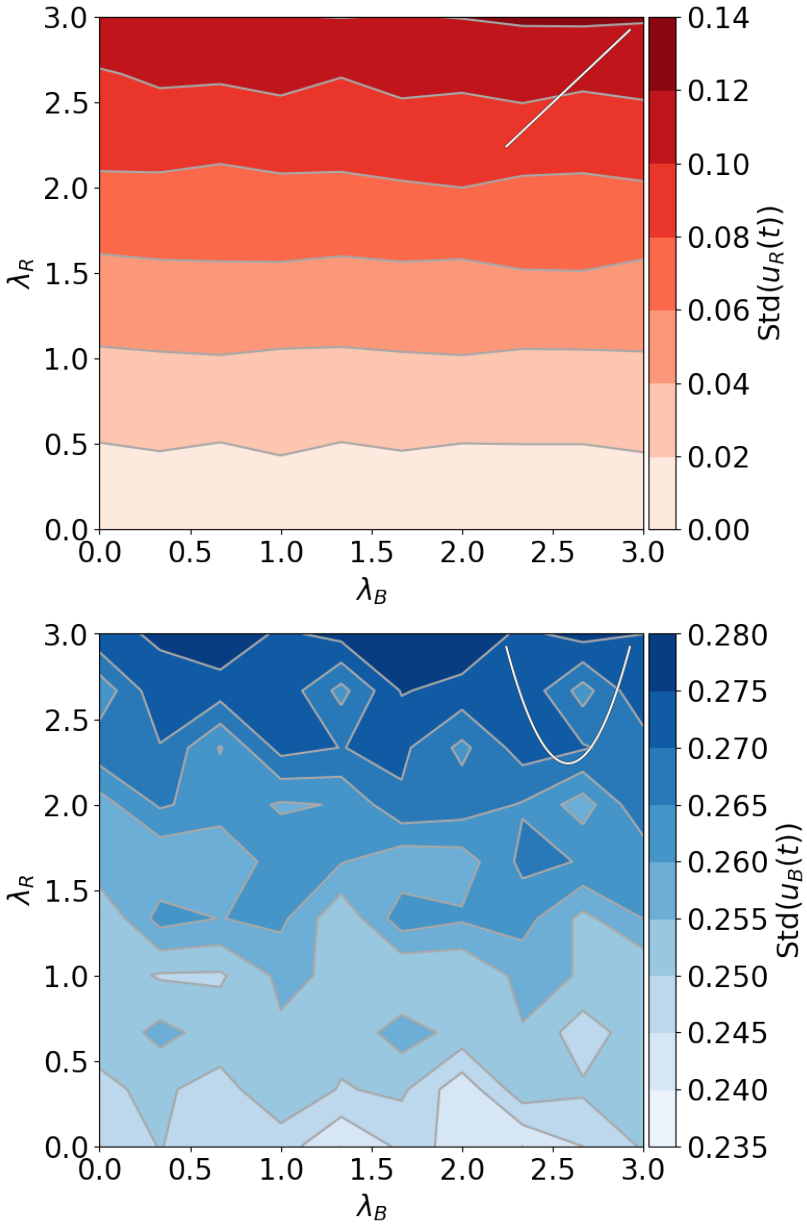

Figure B.12: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = x$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to std of control policy.

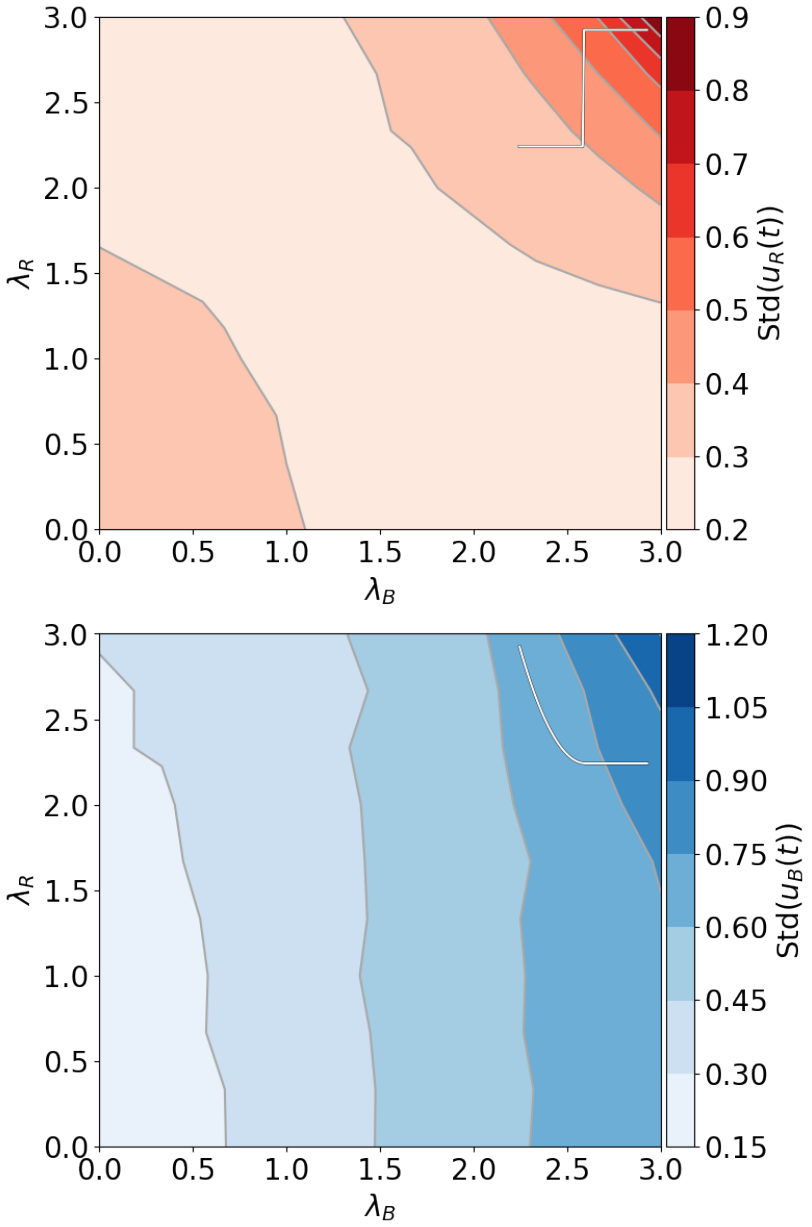

Figure B.13: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to std of control policy.

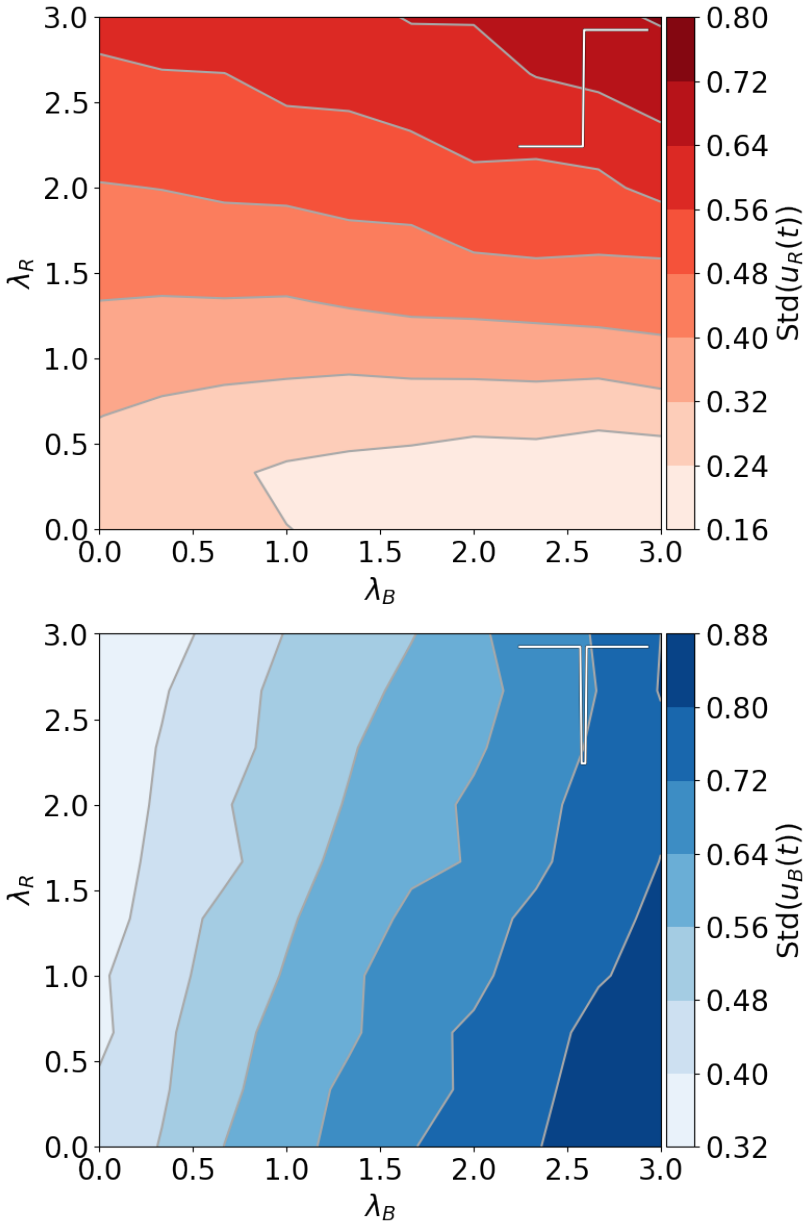

Figure B.14: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. Intensity of color corresponds to std of control policy.

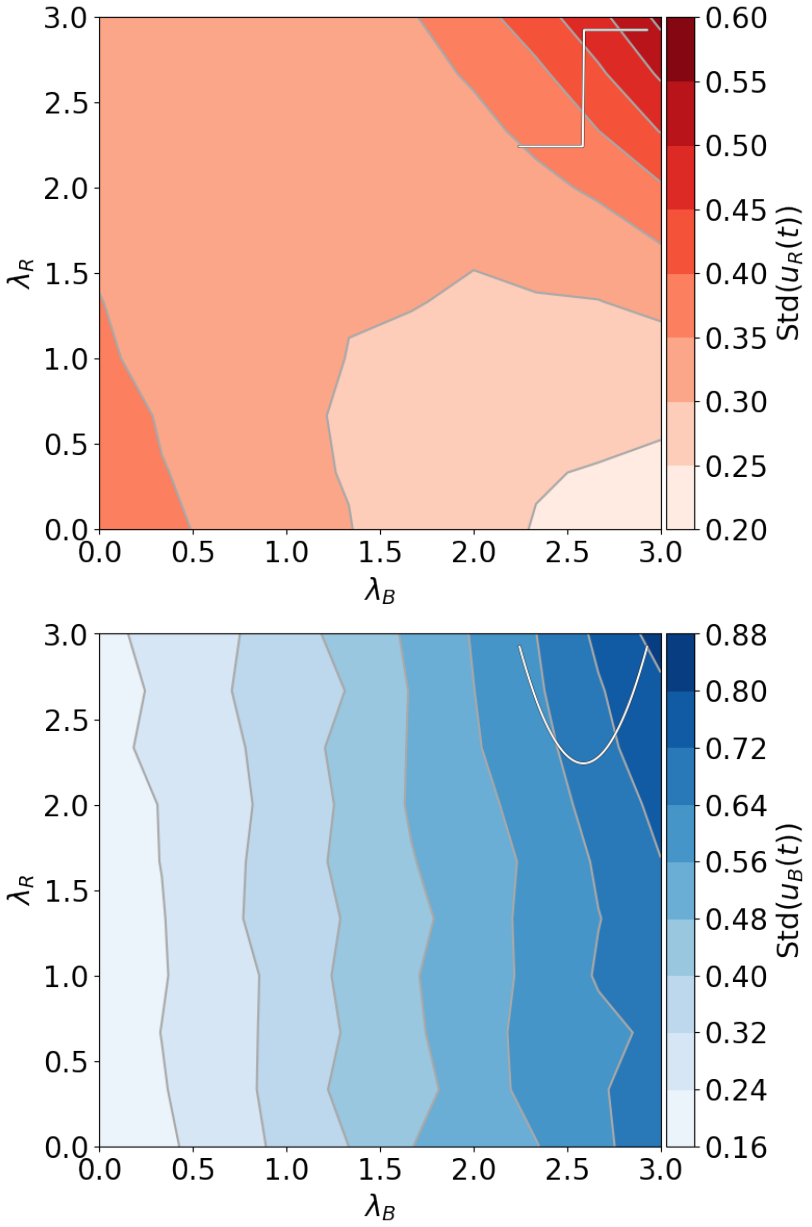

Figure B.15: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = 2[\Theta(x) - \Theta(-x)]$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to std of control policy.

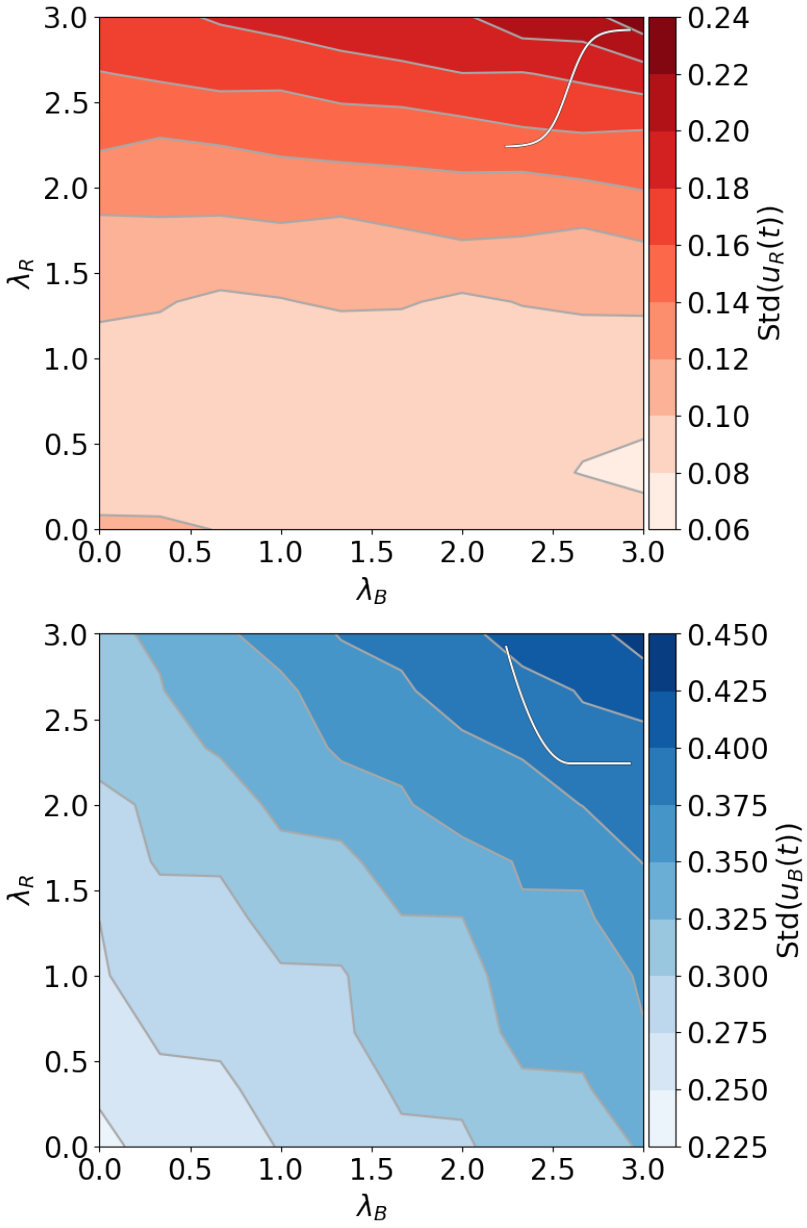

Figure B.16: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = \tanh(x)$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2\Theta(-x)$. Intensity of color corresponds to std of control policy.

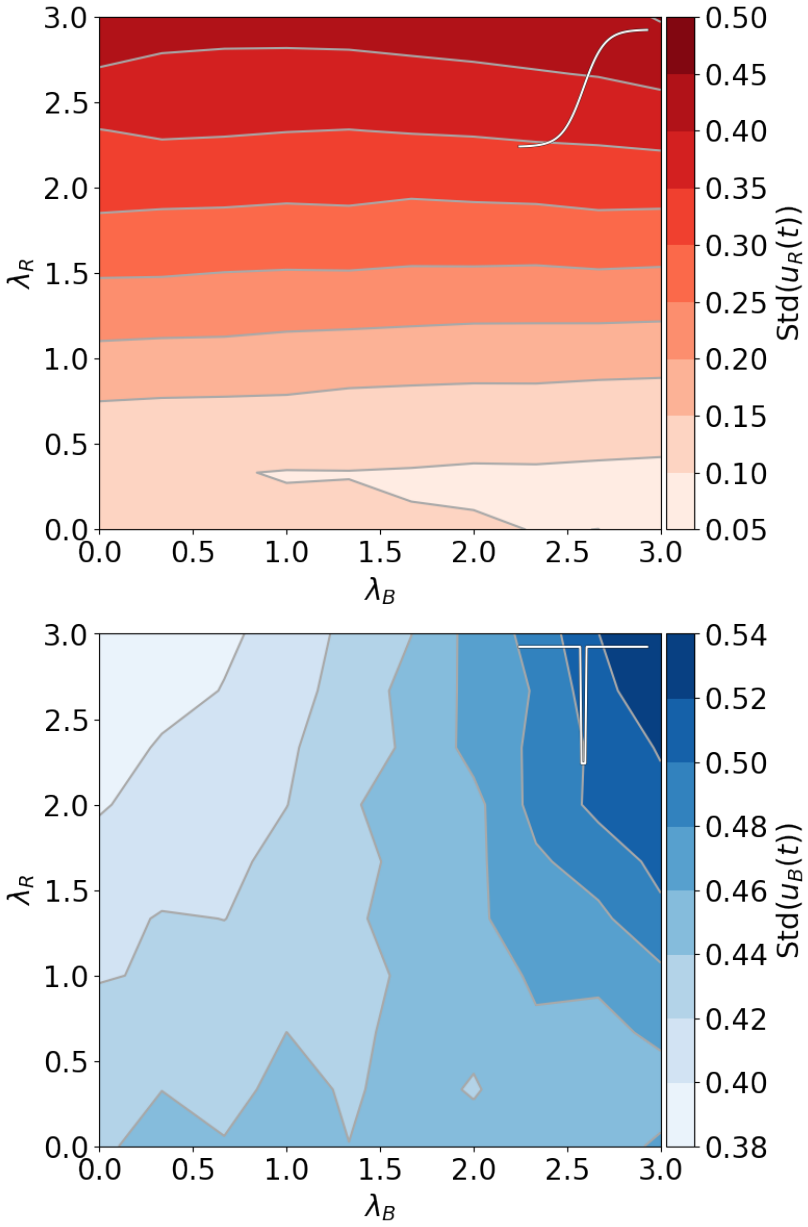

Figure B.17: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ *with Red final condition* $\Phi_{\mathrm{R}}(x) = \tanh(x)$ *and Blue final condition* $\Phi_{\mathrm{B}}(x) = 2[\Theta(|x| - 0.1) - \Theta(0.1 - |x|)]$. *Intensity of color corresponds to std of control policy.*

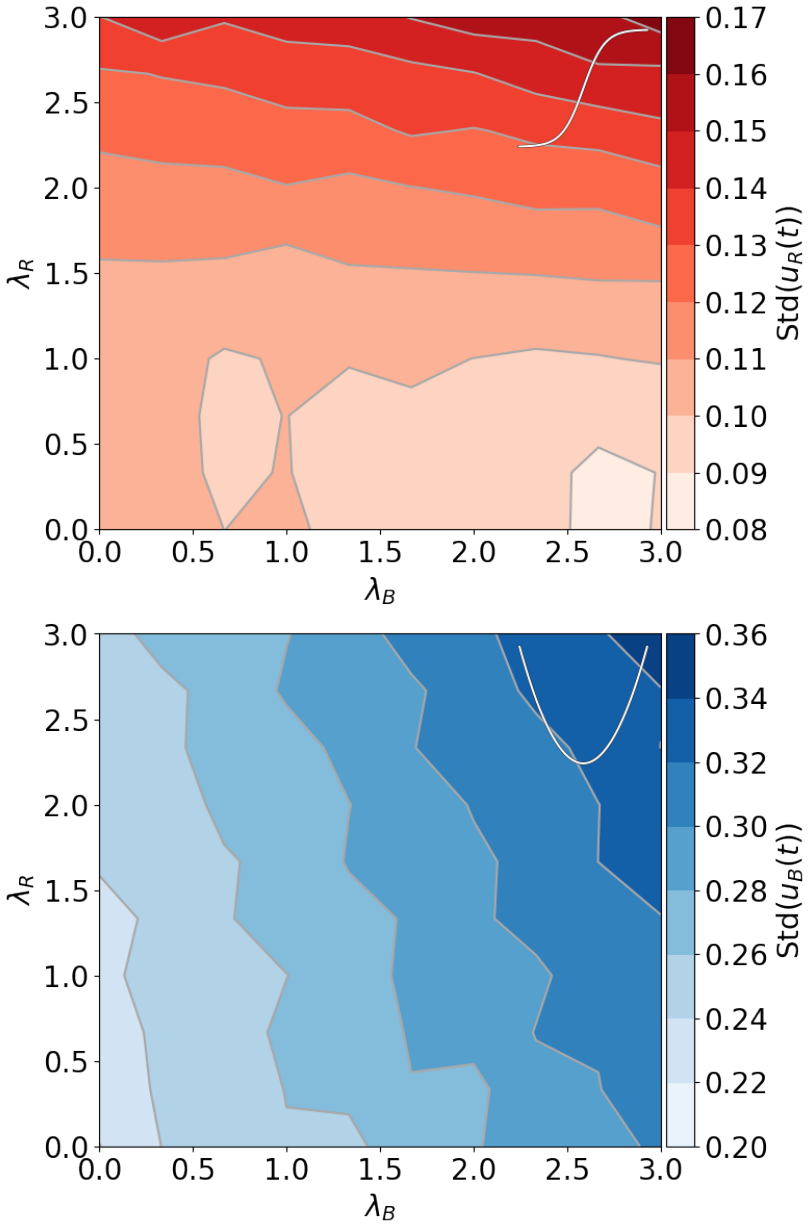

Figure B.18: Parameter sweep over coupling parameters $\lambda_{\mathrm{R}}, \lambda_{\mathrm{B}}$ with Red final condition $\Phi_{\mathrm{R}}(x) = \tanh(x)$ and Blue final condition $\Phi_{\mathrm{B}}(x) = \frac{1}{2}x^2$. Intensity of color corresponds to std of control policy.

Appendix C

Algorithmic details of neuroevolution trading agents

All code used to create the results of this paper is available online at https://gitlab.com/daviddewhurst/coco-neuro-trader-abm; code segments below are listed here only for convenience.

Algorithmic details of agents

All agents (except for neural network agents) submit a number of shares that is distributed Poisson with mean value $E[N] = 100$ by default.

- Zero intelligence: submit bid with probability $s_t \sim \text{Bernoulli}(p_{\text{bid},})$. We set $p_{\text{bid},=\frac{1}{2}}$. Order price is distributed uniformly around the last equilibrium price from the matching engine, $X_t^{(o)} = X_{t-1} + \nu u_t$, where $u_t \sim \mathcal{U}(-1, 1)$ and $\nu > 0$ is the so-called "micro-volatility" preference of the agent [235].

- Zero intelligence plus: the same as a zero-intelligence agent except they submit no order price.

- Momentum: These agents implement a very simple momentum algorithm based on the change in price between the current price and the last observed price:

```
dp = self.last - p  # p is the price; self.last is the last
    observed price
if dp > 0:
    side = 'ask'
    price = p + self.dx  # self.dx is a small price increment
elif dp < 0:
    side = 'bid'
    price = p - self.dx
else:
    side = np.random.choice(['ask', 'bid'])
    price = p
```

  We set the price increment $dx = 0.05$, but this is obviously arbitrary. In particular, one could set this to be a random variable that takes other information about market state as parameters.

- Fundamental-value: these agents have a "true valuation" estimate of what they think the traded asset is actually worth. If the price of the asset is below this value, they submit a bid, while if it is above this value, they submit an ask.

```
if p == orders.NaP:  # how we denote no price information returned
    by matching engine
    return []
# mean_price is the agent's true-valuation estimate
# price_tolerance is the agent's estimate of error around the true
    mean price
# that they are willing to accept
```

```
elif p > self.mean_price + self.price_tolerance:
    side = 'ask'
    price = p + self.dx + self.vol_pref * np.random.random() - self
        .vol_pref / 2  # vol_pref is nu from above
elif p < self.mean_price - self.price_tolerance:
    side = 'bid'
    price = p - self.dx + self.vol_pref * np.random.random() - self
        .vol_pref / 2
else:
    return []
```

As with any other agent with a `vol_pref` attribute, this algorithm can be made deterministic by setting `self.vol_pref` $= 0$.

- Mean reversion: this agent anticipates a return to some rolling mean. If the price is higher than the rolling mean, the agent submits an ask order. If the price is lower than the rolling mean, it submits a bid order.

```
self.prices[self.current_ind] = p  # a list of recent prices
# implementation of rolling window since order they're stored doesn
    't matter
# self.window is number of prices that are stored
self.current_ind = (self.current_ind + 1) % self.window

self.mean_price = self.prices[self.prices > 0].mean()
if p == orders.NaP:
    return []
elif p > self.mean_price + self.price_tolerance:
    side = 'ask'
    price = p - self.vol_pref * np.random.random()
```

```
elif p < self.mean_price - self.price_tolerance:
    side = 'bid'
    price = p + self.vol_pref * np.random.random()
else:
    return []
```

We set `self.window` = 50 by default.

- Market making: this agent provides liquidity on both sides of a limit order book in an attempt to collect small profits that result from other market participants crossing the spread. Like the mean reverting agent, this agent keeps a rolling list of recent prices to which it refers when calculating likelihood of price movements.

```
self.prices[self.current_ind] = self.engines[eid].eq  # interfacing
    with one of possibly many matching engines
self.current_ind = (self.current_ind + 1) % self.window
p = self.prices[self.prices > 0].mean()

# React to inventory imbalance
# target_inventory_size is generally assumed to be a small number
divergence = (self.shares_held - self.target_inventory_size)
if divergence > self.inventory_tolerance:
    self.shift = max(0.01, self.shift - 0.01)  # shifts reference
        price
    self.spread += 0.01  # shifts how far on either side of the
        equilibrium the agent will place orders
elif divergence < -self.inventory_tolerance:
    self.shift += 0.01  # shift price up
    self.spread += 0.01
else:
```

```
15      self.shift = 0.
16      self.spread = max(0.01, self.spread - 0.01)
```

Unlike the other agents described, the market-making agent then submits two orders instead of only one.

```
buy_order = orders.Order(
    self.uid,
    f'{self.uid}-{self.order_number}',
    'bid',
    max(int(self.shares - divergence), 10),  # adaptively calculate
        number of shares to lower inventory imbalance
    np.round(p - self.spread + self.shift, 2),
    time_in_force=0,
)
self.order_number += 1

sell_order = orders.Order(
    self.uid,
    f'{self.uid}-{self.order_number}',
    'ask', max(int(self.shares + divergence), 10),
    np.round(p + self.spread + self.shift, 2),
    time_in_force=0,
)
self.order_number += 1
```

Marginalization procedure

We provide some more details on how we attempt to marginalize over unobserved market variables using analogues of those variables generated by the agent-based

model (ABM). Recall from the main paper that we can view the ABM as just a stochastic function $\mathcal{G}(\alpha, \mathcal{M})$ that, given an agent parameter vector α and evolutionary mechanism $\mathcal{M}$ (which may be the identity mechanism, i.e. no selection or mutation), generates orderbooks $D_b(x,t)|\alpha$, $D_a(x,t)|\alpha$ and price time series $X_t|\alpha$. Our first step is simply to jointly obtain many orderbooks and price time series by calling $\mathcal{G}(\alpha, \mathcal{M})$ many times for a variety of different α. We then have random fields of orderbooks $\mathcal{D}_b = \{D_b(x,t)|\alpha\}_\alpha$, $\mathcal{D}_a = \{D_a(x,t)|\alpha\}_\alpha$ and price time series $\mathcal{X} = \{X_t|\alpha\}_\alpha$ indexed by the vectors α that we draw from some joint distribution $p(\alpha)$. In our implementation, we just set $p(\alpha)$ to a multinomial distribution with equal probabilities of picking each agent type.

We called $\mathcal{G}(\alpha, \mathrm{id})$ once for each saved evolved neural network and added the evolved neural network to the simulation. (One simple modification of our work would be to *not* add the evolved neural networks to these marginalization simulations and see how this affects the performance of the networks on real data. We hypothesize that it may have a significant effect because the effects of the network itself will not be reflected in the marginalization data, which more closely simulates a real market in which the neural network has very little market power, but we are unsure of the direction in which the effect would be.) Given these random fields of orderbooks and price time series, we create the price difference $\Delta X = X_t - X_{t-1}$ and the total resting order volume time series calculated from the sequence of orderbooks, $\Delta\hat{V}^{(b)}$ and $\Delta\hat{V}^{(a)}$, defined analogously. The change in price does have an observable real market analogue—the actual change in price of the spot asset—while the change in resting order volume does not, since we did not have access to real market orderbook data. We store ΔX in a ball tree using the ℓ_1 distance metric. Given an observed ΔX^* from real market

data, we then query the ball tree for the indices $i_1, ..., i_k$ of the nearest k neighbors of ΔX^*, $\Delta X_{i_1}, ..., \Delta X_{i_k}$ and then extract $\Delta \hat{V}_{i_1}^{(b)}, ..., \Delta \hat{V}_{i_k}^{(b)}$ and $\Delta \hat{V}_{i_1}^{(a)}, ..., \Delta \hat{V}_{i_k}^{(a)}$. We then approximate the market $(\Delta V^{(b)}, \Delta V^{(a)})|\Delta X^*$ by the approximation to the conditional expectation given by the result of the nearest-neighbors query:

$$\Delta V^{(b)}|\Delta X^* \approx \frac{1}{k}\sum_{j=1}^{k} \Delta \hat{V}_{i_k}^{(b)}, \qquad \Delta V^{(a)}|\Delta X^* \approx \frac{1}{k}\sum_{j=1}^{k} \Delta \hat{V}_{i_k}^{(a)}. \tag{C.1}$$

The validity of this procedure rests entirely on the degree to which the mechanics and dynamics of the ABM simulate the true mechanics and dynamics of the real asset market. Nothing in the above procedure is guaranteed to produce anything close to the true, unobservable $\Delta V^{(b)}$ and $\Delta V^{(a)}$ at all unless the crucial assumption that $(\Delta V^{(b)}, \Delta V^{(a)})|\Delta X^*$ are distributed "similarly" to $(\Delta \hat{V}^{(b)}, \Delta \hat{V}^{(a)})|\Delta X^*$ is satisfied—and, in our work, we have been cavalier about what exactly "similarly" means. This is certainly another area on which future research could be conducted.

Risk management routines We implemented some simple risk management routines to halt the losses of poorly-performing algorithms. These routines come in three variants: two versions of the traders' adage "cut your losses but let your winners run," and one version that seeks to limit total leverage (net open position of spot contracts).

- "Cut losses" algorithm operating on price levels: halts trading if total profit is below a certain level.

```
1    def cut_losses(
2        profit_arr,
3        lower_limit=-0.5,
4        method='absolute',
5        roll_ind=100,
```

```
6           ):
7       """Implements the level-based adage "cut your losses and let
            your winners run."
8
9       If the level of profit is below a certain set level, halt
            trading. Otherwise, keep going.
10
11      :param profit_arr: array of profits so far
12      :type profit_arr: index-able
13      :param lower_limit: the level of profit below which trading
            should halt.
14      :type lower_limit: `float`
15
16      :returns : `bool`, whether or not trading should halt
17      """
18      if (method == 'absolute') or (len(profit_arr) <= roll_ind):
19          if profit_arr[-1] <= lower_limit:
20              return True
21          return False
22
23      elif method == 'rolling':
24          max_profit = np.max( profit_arr[-roll_ind:] )
25          if profit_arr[-1] - lower_limit < max_profit:
26              return True
27          return False
28
29      else:  # they didn't specify anything, better to halt rather
            than propagate their errors
30          return True
```

If `method == 'rolling'` and `roll_ind` $= 0$, this is equivalent to halting trading at time t if π_t is less than $\max_{t'=1,\ldots,t} \pi_{t'} -$ `lower_limit`. We used these settings of this algorithm during backtesting.

- "Cut losses" algorithm operating on changes in price level: halts trading if $\Delta\pi$ is less than some specified cutoff.

```
    def cut_losses_delta(
        profit_arr,
        lower_limit=-0.5,
        ):
    """Implements the flow-based adage "cut your losses and let
       your winners run."

    If the change in profit is below a certain set level, halt
       trading. Otherwise, keep going.

    :param profit_arr: array of profits so far
    :type profit_arr: index-able
    :param lower_limit: the level of delta profit below which
       trading should halt.
    :type lower_limit: `float`

    :returns : `bool`, whether or not trading should halt

    """
    if (len(profit_arr) >= 2) and (profit_arr[-1] - profit_arr[-2]
       <= lower_limit):
        return True
    return False
```

- Leverage limit: if the total number of spot contracts (long or short, i.e., positive or negative) is above a set threshold, halts trading.

```
    def limit_leverage(
        shares_arr,
        max_shares=150,
        ):
    if np.abs( shares_arr[-1] ) >= max_shares:
        return True
    return False
```

The existence of a risk management routine supervising an algorithm's execution will, in general, change the algorithm's profit distribution. As an example, suppose that a trading algorithm had zero average profit $E[\pi] = \int_{-\infty}^{\infty} \pi p(\pi)\ d\pi = 0$ when unsupervised by a risk management algorithm. *If* the risk management algorithm were "perfect" in the sense that it could guaranteed limit loss to no less than $-\pi^*$ for $\pi^* > 0$, then it is the case that the trading algorithm would have positive expected profit. This is equivalent to the common-sense statement that, if H_1 and H_2 are two people who are indistinguishable in every way except for their age, H_1 is 70 years old, and H_2 is 80 years old, then the life expectancy of H_2 is greater than that of H_1. Of course, no such "perfect" risk management algorithm exists, particularly when trading in real markets where execution and liquidity concerns become dominant.

Profitability of evolved agents

We tested the profitability of all evolved algorithms on the validation dataset, the currency pairs EUR/USD and GBP/USD from 1-1-2010 to 12-31-2015, and then tested the elite evolved algorithms—what we termed the top five most profitable algorithms on the validation dataset—on a test dataset, EUR/USD and GBP/USD

from 1-1-2016 to 7-1-2019 (the date we collected this data).

Below, we display the distributions of profit by the elite evolved algorithms on the test dataset. By total profit, we mean all profit the algorithm earned over all trading episodes (defined in the main paper) in the test dataset. The probability of profit, expected profit, and maximum *a-posteriori* profit are all measured on a per-trading-episode basis. The red curve superimposed on the histogram is a maximum-likelihood estimated lognormal pdf, which fits the observed data fairly well in some cases and not well in others. We fit a lognormal pdf as, if profit accumulates via random positive and negative percentages changes, the lognormal distribution is the appropriate limiting distribution by the multiplicative CLT (after shifting the distribution by the appropriate location and scale parameters).

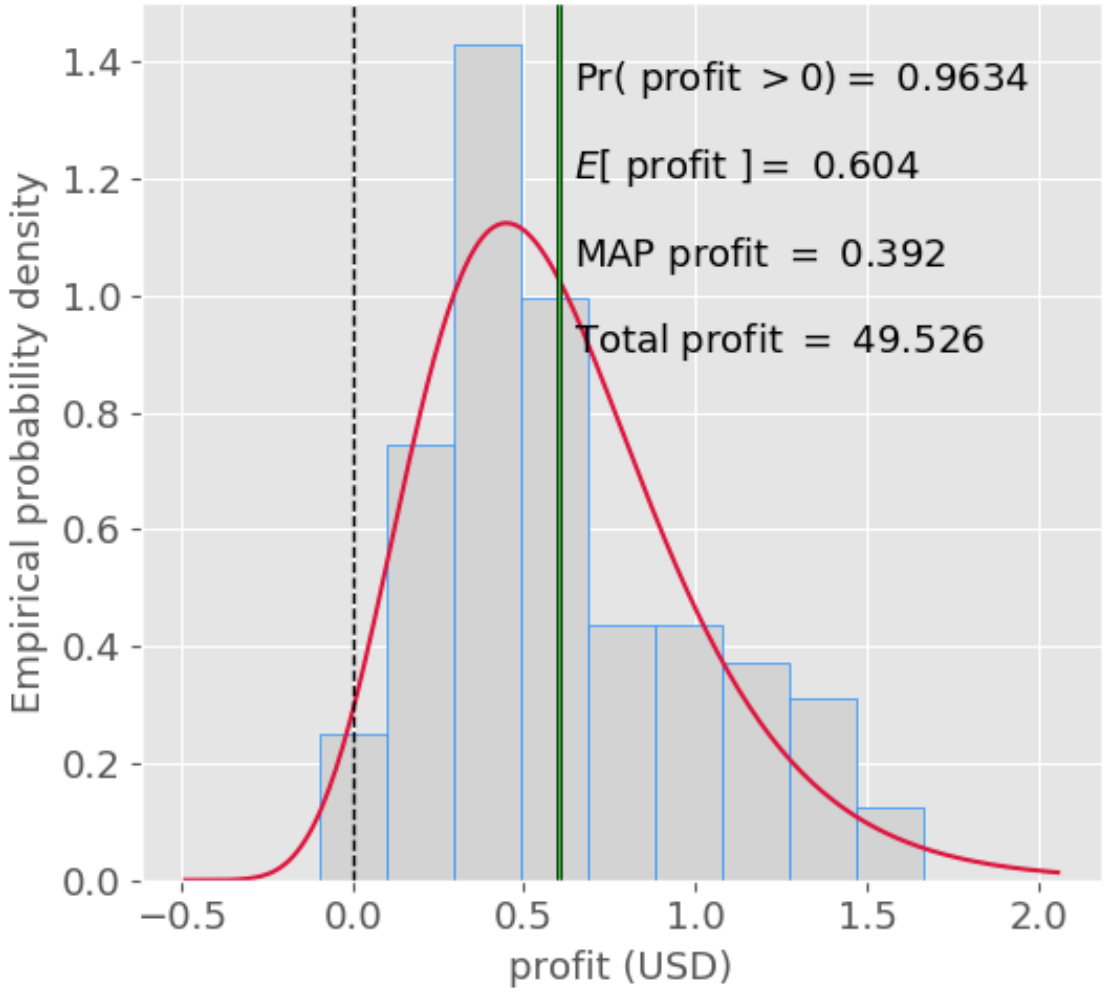

Figure C.1: $g = 10$, independent simulation 21

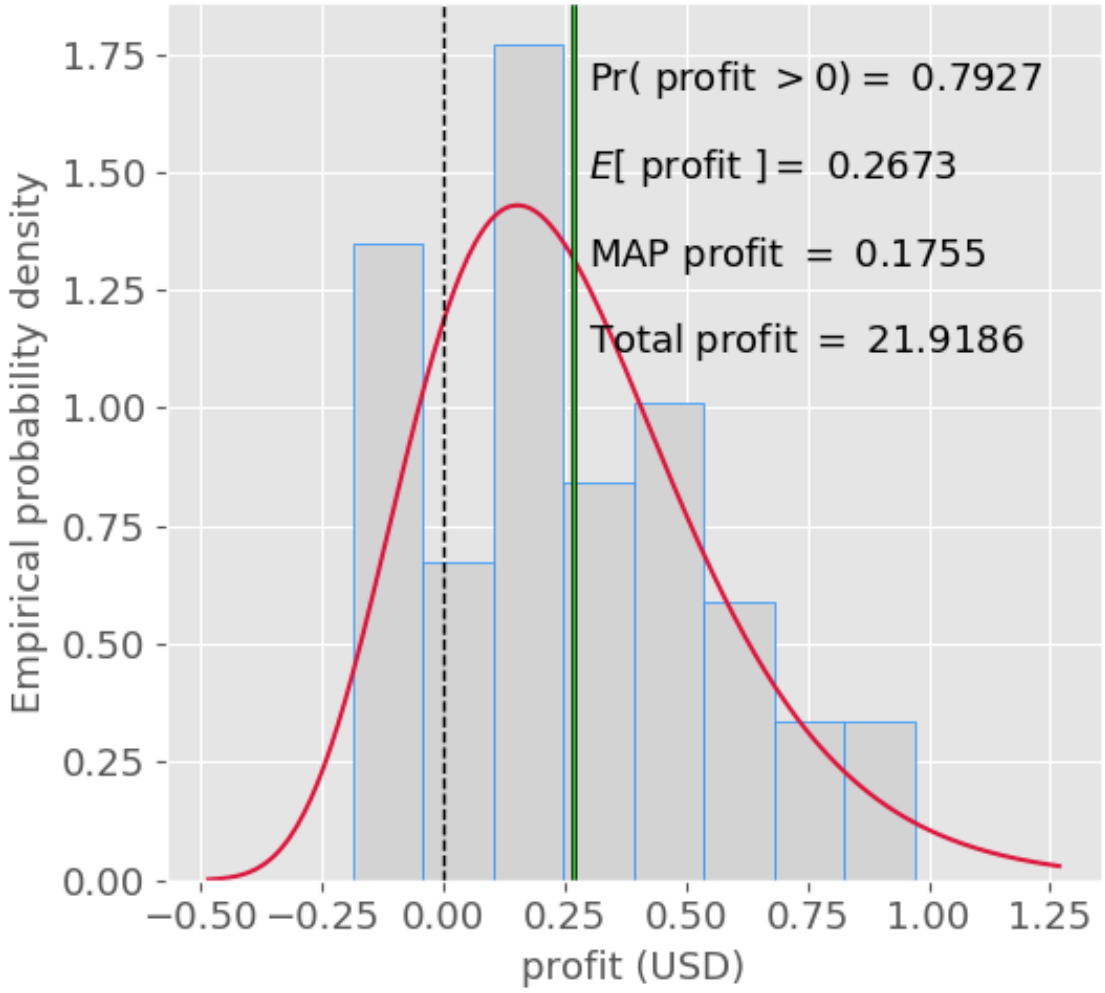

Figure C.2: $g = 10$, independent simulation 1

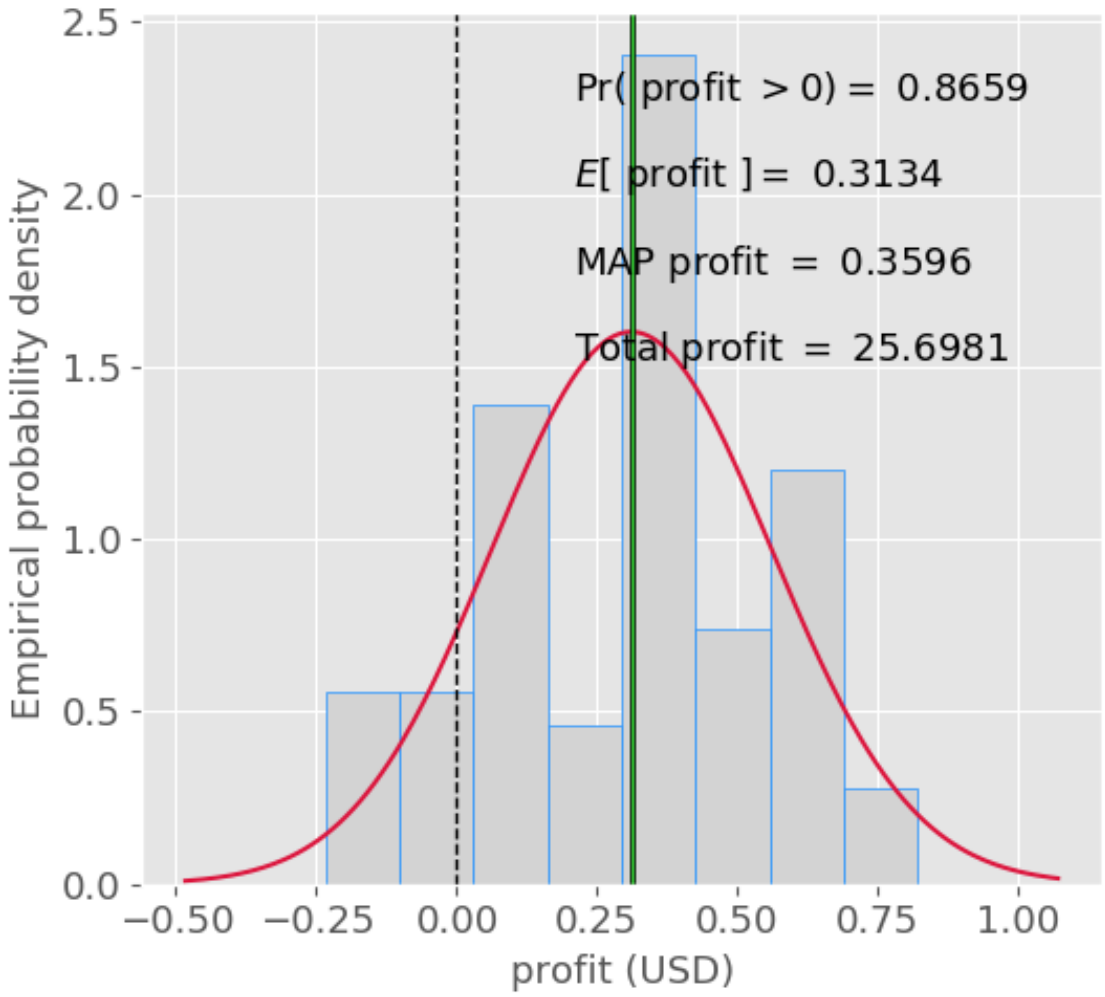

Figure C.3: $g = 10$, independent simulation 93

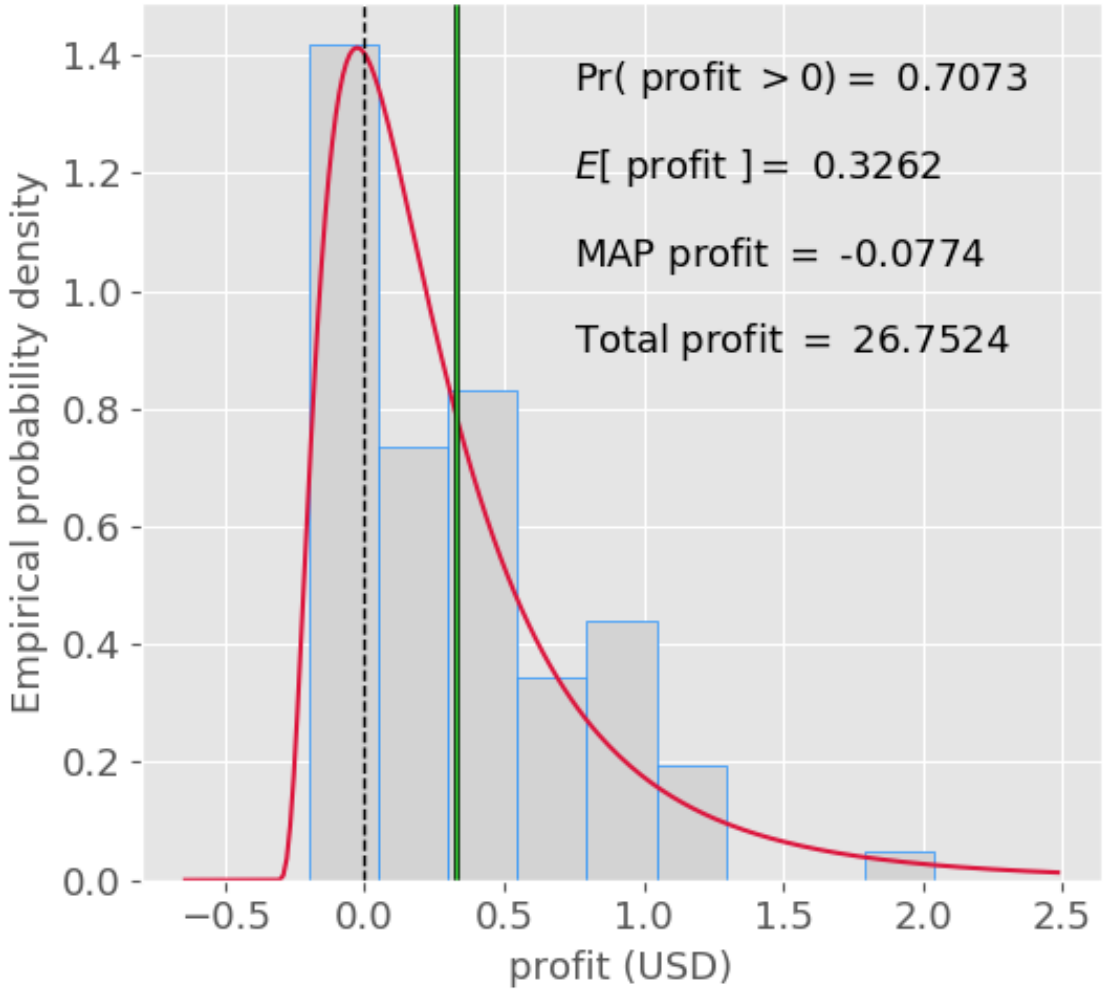

Figure C.4: $g = 10$, independent simulation 79

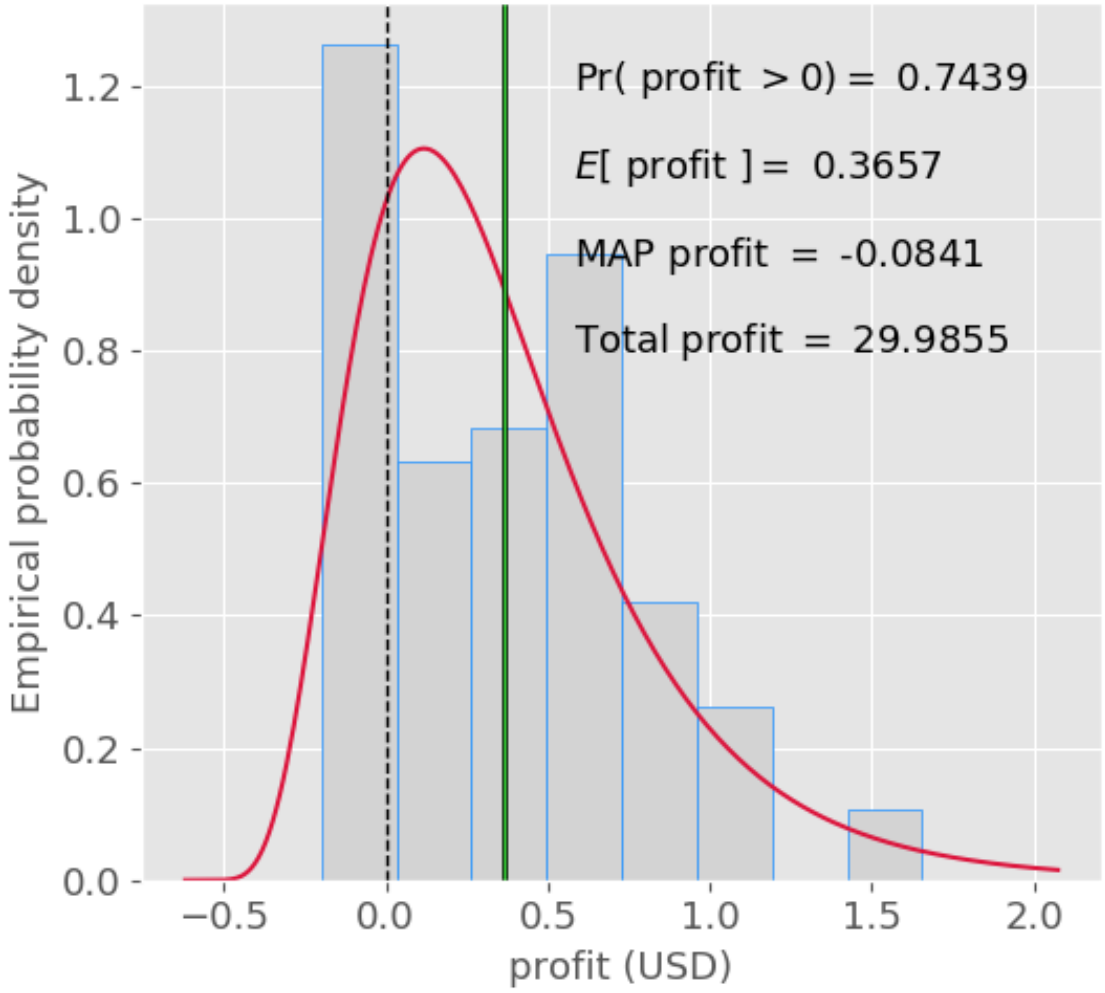

Figure C.5: $g = 10$, independent simulation 38

Appendix D

Technical details of the discrete shocklet transform

D.1 Document-free topic networks

An important application of the DST is the partial recovery of context- or document-dependent information from aggregated time series data. In natural language processing, many models of human language are statistical in nature and require original documents from which to infer values of parameters and perform estimation [236, 237]. However, such information can be both expensive to purchase and require a large amount of physical storage space. For example, the tweet corpus from which the labMT rank dataset used throughout this paper was originally derived is not inexpensive and requires approximately 55 TB of disk space for storage[1]. In contrast, the dataset used here is derived from the freely-available LabMT word set and is less than

[1]The dataset is available for purchase from Twitter at http://support.gnip.com/apis/firehose/overview.html. The on-disk memory statistic is the result of `du -h <dirname> | tail -n 1` on the authors' computing cluster and so may vary by machine or storage system

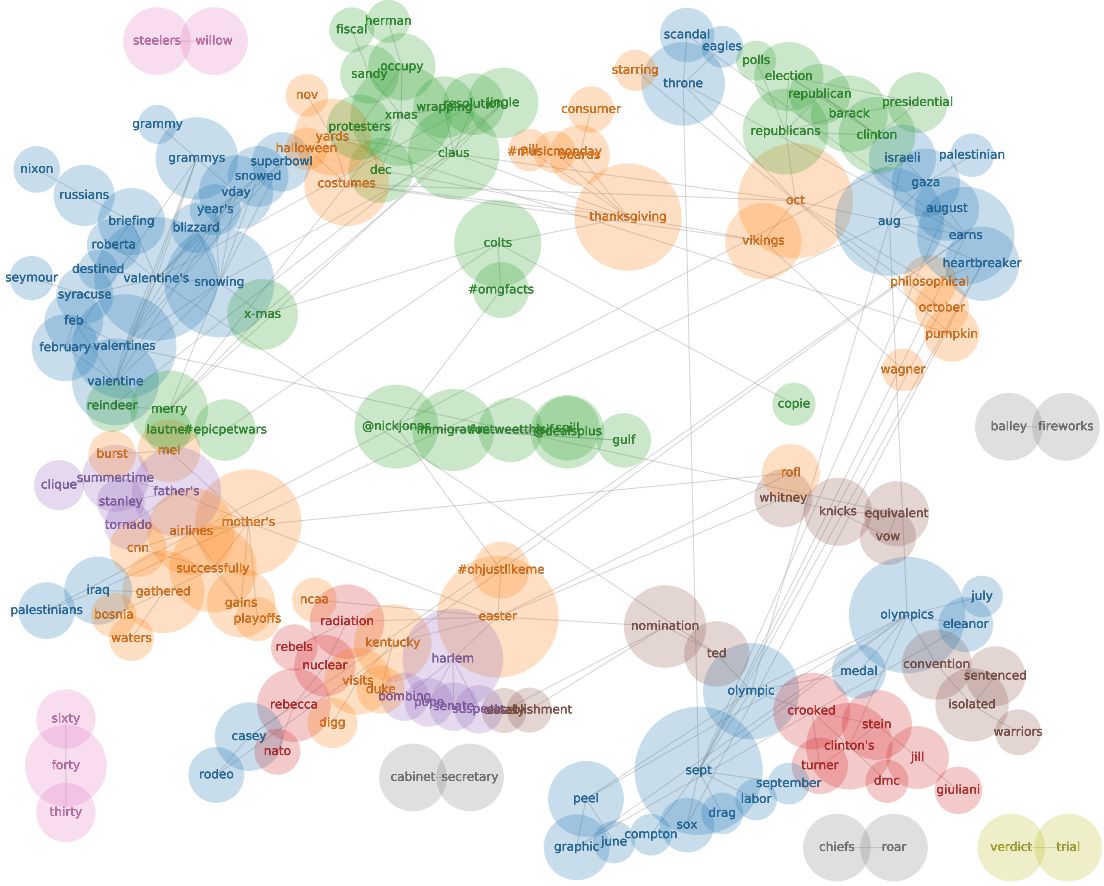

Figure D.1: Topic network inferred from weighted shock indicator functions. At each point in time, words are ranked according to the value of their weighted shock indicator function and the top k words are taken and linked pairwise for an upper bound of $\binom{k}{2}$ additional edges in the network; if the edge between words i and j already exists, the weight of the edge is incremented. The edge weight increment at time t is given by $w_{ij,t} = \frac{R_{i,t}+R_{j,t}}{2}$, the average of the weighted shock indicator for words i and j, with the total edge weight thus given by $w_{ij} = \sum_t w_{ij,t}$. After initial construction, the backbone of the network is extracted using the method of Serrano et al. [5]. The network is pruned further by retaining only those nodes i, j and edges e_{ij} for which w_{ij} is above the p-th percentile of all edge weights in the backboned network. The network displayed here is constructed by setting $k = 20$ and $p = 50$, where size of the node indicates normalized page rank. Topics are associated with distinct communities, found using the modularity algorithm of Clauset et al. [6].

400 MB in size. If topics of relatively comparable quality can be extracted from this smaller and less expensive dataset, the potential utility to the scientific community

at large, could be high.

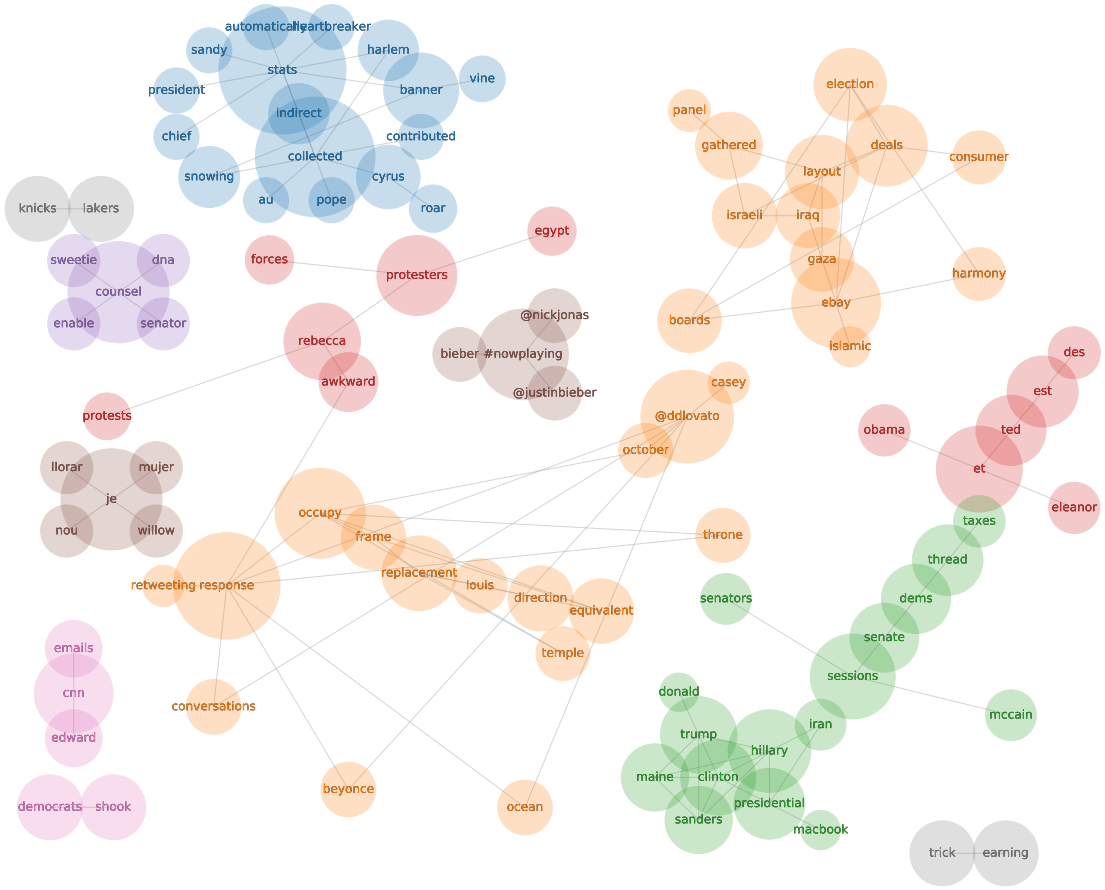

Figure D.2: Topic network inferred from weighted shock indicator functions. At each point in time, words are ranked according to the value of their weighted shock indicator function and the top k words are taken and linked pairwise for an upper bound of $\binom{k}{2}$ additional edges in the network; if the edge between words i and j already exists, the weight of the edge is incremented. The edge weight increment at time t is given by $w_{ij,t} = \frac{R_{i,t}+R_{j,t}}{2}$, the average of the weighted shock indicator for words i and j, with the total edge weight thus given by $w_{ij} = \sum_t w_{ij,t}$. After initial construction, the backbone of the network is extracted using the method of Serrano et al. [5]. The network is pruned further by retaining only those nodes i, j and edges e_{ij} for which w_{ij} is above the p-th percentile of all edge weights in the backboned network. The network displayed here is constructed by setting $k = 20$ and $p = 50$, where size of the node indicates normalized page rank. Topics are associated with distinct communities, found using the modularity algorithm of Clauset et al. [6].

We demonstrate that a reasonable topic model for Twitter during the time period

of study can be inferred from the panel of rank time series alone. This is accomplished via a multi-step meta-algorithm. First, the weighted Shock Indicator Function R_i is calculated for each word i. At each point in time t, words are sorted by their respective shock indicator functions as in Fig. **??**. At time step t, the top k words are taken and linked pairwise for an upper bound of $\binom{k}{2}$ additional edges in the network; if an edge already exists between word i and j, it is incremented by the mean of the words' respective weighted Shock Indicator Function $\frac{R_i+R_j}{2}$. Performing this process for all time periods results in a weighted network of related words. The weights $w_{ij} = \sum_t \frac{R_{i,t}+R_{j,t}}{2}$ are large when the value of a word's weighted shock indicator function is large or when a word is frequently in the top k, even if it is never near the top. The resulting network can be large; to reduce its size, its backbone is extracted using the method of Serrano *et al.* [5] and further pruned by retaining only those nodes and edges for which the corresponding edge weights are at or above the p-th percentile of all weights in the backboned network. Topics are associated with communities in the resulting pruned networks, found using the modularity algorithm of Clauset *et al.* [6].

Fig. D.1 and Fig. D.2 display the result of this procedure for $k = 20$ and $p = 50$. Unique communities (topics) are indicated by node color. In the co-shock network (Fig. D.1), topics include, among others:

- Winter holidays and events ("valentines", "superbowl", "vday",...);
- U.S. presidential elections ("republicans", "barack", "clinton", "presidential",...);
- Events surrounding the 2016 U.S. presidential election in particular ("clinton's", "crooked", "giuliani", "jill", "stein",...);

while the co-shock network displays topics pertaining to:

- popular culture and music ("bieber", "#nowplaying", "@nickjonas", "@justinbieber");
- U.S. domestic politics ("clinton", "hillary", "trump", "sanders", "iran", "sessions",...);
- and conflict in the Middle East ("gaza", "iraq", "israeli", "gathered")

The predominance of U.S. politics at the exclusion of politics of other nations is likely because the labMT dataset contains predominantly English words.

D.2 Statistical details

In this appendix we will outline some statistical details of the DST and STAR algorithm that are not necessary for a qualitative understanding of them, but could be useful for more in-depth understanding or efforts to generalize them.

We first give an illustrative example of how a sociotechnical time series can differ substantially from two null models of time series that have some similar statistical properties, displayed in Fig. D.3 (a more information-rich version of Fig. 3.5, displayed in the main body), panels A and B. In panel A, we display an example sociotechnical time series in the red curve, usage rank of the word "bling" within the LabMT subset of words on Twitter (denoted by r_t), and σr_t, a randomly shuffled version of this time series. We denote $\sigma \in \mathcal{S}_T$, the symmetric group on T elements, and draw σ from the uniform distribution over $\mathcal{S}_T$. It is immediately apparent that the structure of r_t and

σr_t are radically different in autocorrelation (both in levels and differences) and we do not investigate this admittedly-naïve null model any further.

We next consider a random walk null model constructed as follows: first differencing r_t to obtain $\Delta r_t = r_t - r_{t-1}$, we apply random elements $\sigma_i \in \mathcal{S}_T$ and integrate, displaying the resulting $r_{\sigma_i t} = \sum_{t' \leq t} \sigma_i \Delta r_t$ in panel C of Fig. D.3. Visual inspection (i.e., the "eye test") also demonstrates that these time series do not replicate the behavior displayed by the original r_t; they become negative, have a dynamic range that is almost an order of magnitude larger, and are more highly autocorrelated. We contrast the results of the DST on r_t and draws from this random walk null model in panels D – G of Fig. D.3. In panel D we display the DST of r_t, while in panels E – G we display the DST of three random $\sigma_i r_t$. The DSTs of the draws from the random walk model are more irregular that the DST of r_t, displaying many time-domain fluctuations between large positive values and large negative values. In contrast, the DST of r_t is relatively constant except near August of 2015, where it exhibits a large positive fluctuation across a wide range of W. The underlying dynamics for this fluctuation were driven by the release of a popular song called "Hotline Bling" on July 31st, 2015.

As a couterpoint to the DST, we computed the discrete wavelet transform (DWT) of r_t and the same $\sigma_i r_t$. We computed the wavelet transform using the Ricker wavelet,

$$\psi(\tau, W) = \frac{2}{\sqrt{3W\pi^{1/2}}} \left[1 - \left(\frac{\tau}{W}\right)\right] e^{-\tau^2/(2W^2)}. \tag{D.1}$$

We chose to compare the DST with the DWT because these transforms are very similar in many respects: they both depend on two parameters (a location parameter

τ and a scale parameter W); they both output a matrix of shape $T \times N_W$ (N_W rows, one for each value W, and T columns, one for each value of τ). There are some key difference between these transforms, however. The "kernels" of the wavelet transform—the kernels—have unique properties not shared by our shock-like kernels: wavelets $\psi(t)$ are defined on all of $\mathbb{R}$, satisfy $\lim_{t\to\pm\infty}\psi(t) = 0$, and are orthonormal. Our shock-like kernels do not satisfy any of these properties; they are defined on a finite interval $[-W/2, W/2]$, do not vanish at the endpoints of this interval, and are not orthogonal functions. Hence, differences in the DST and DWT of a time series are due primarily to the choice of convolution function—shock-like kernel in the case of the DST and wavelet in the case of the DWT. We display the DWT of r_t and the same $\sigma_i r_t$ in panels H – K of Fig. D.3. Comparing these transforms with the DSTs displayed in panels D – G, we see that the DST has increased time-localization over the DWT in time intervals during which the time series exhibit shock-like dynamics. As we note in Sec. 3.3.1 (there when comparing STAR to Twitter's ADV anomaly detection algorithm), this observation should not be construed as equivalent to the statement that the DST is in some way superior to the DWT or should supersede the DWT for general time series processing tasks; rather, it is evidence that the DST is a superior transform than the DWT for the purpose of finding shock-like dynamics in sociotechnical time series—a task for which it was designed and the DWT was not.

We finally note an analytical property of the DST that, while likely not useful in practice, is a fact that should be recorded and may be useful in constructing theoretical extensions of the DST. The DST is defined in Eq. 3.11, which we record here for ease in reference:

$$\mathrm{C}_{\mathcal{K}^{(\cdot)}}(t, W|\theta) = \sum_{-\infty}^{\infty} x(t+\tau)\mathcal{K}^{(\cdot)}(\tau|W,\theta), \tag{D.2}$$

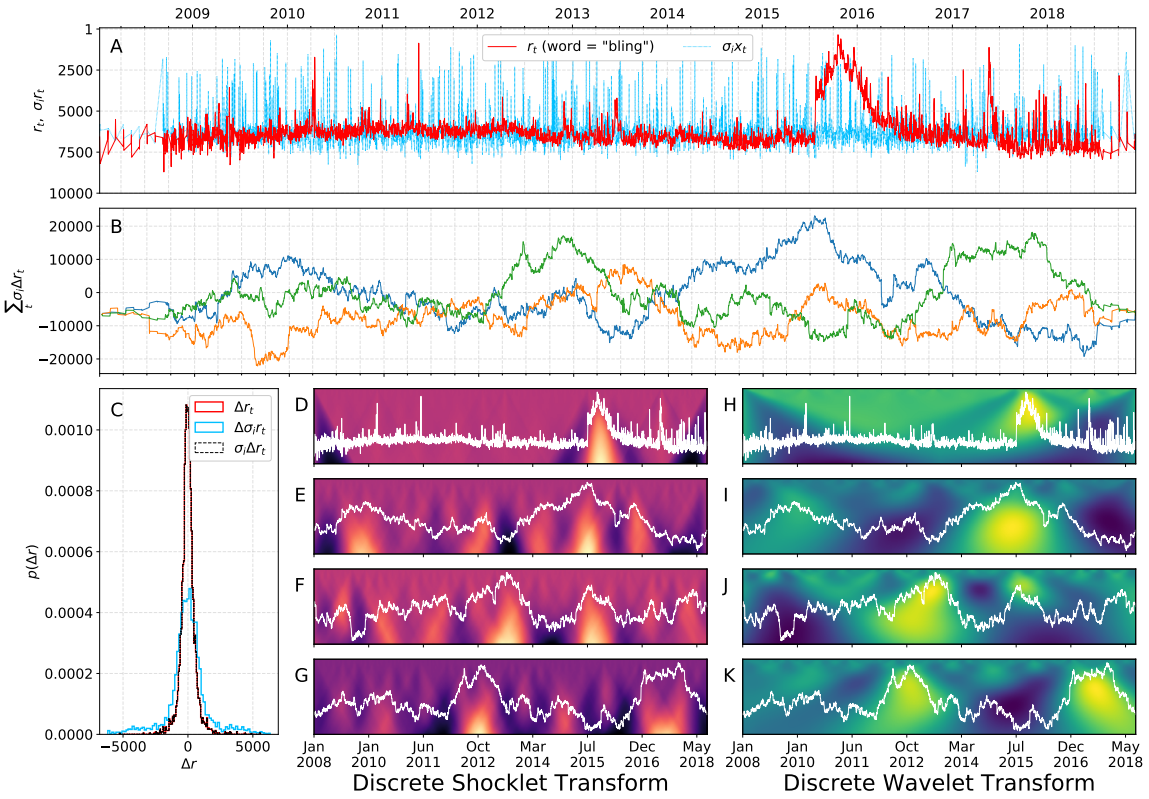

Figure D.3: Intricate dynamics of sociotechnical time series. Sociotechnical time series can display intricate dynamics and extended periods of anomalous behavior. The red curve shows the time series of the ranks down from top of the word "bling" on Twitter. Until 2015/10/31, the time series presents as random fluctuation about a steady trend that is nearly indistinguishable from zero. However, the series then displays a large fluctuation, increases rapidly, and then decays slowly after a sharp peak. The underlying mechanism for these dynamics was the release of a popular song titled "Hotline Bling" by a musician known as "Drake". Returns $\Delta r_t = r_{t+1} - r_t$ are calculated and their histogram is displayed in panel C. To demonstrate the qualitative difference of the "bling" time series from other time series with an identical returns distribution, elements of the symmetric group $\sigma_i \in \mathcal{S}_T$ are applied to the returns of the original series, $\Delta r_t \mapsto \Delta r_{\sigma_i t}$, and the resultant noise is integrated and plotted as $r_{\sigma_i t} = \sum_{t' \leq t} \Delta r_{\sigma_i t}$. The bottom-left panel (C) displays time-decoupled probability distributions of the returns of the plotted time series. The distributions of Δr_i and $\sigma \Delta r_i$ are identical, as they should be, but the integrated series have entirely different spectral behavior and dynamic ranges. Panels [D-G] display the discrete shocklet transform of the original series and the random walks $\sum_{t' \leq t} \Delta r_{\sigma_i t}$, showing the responsiveness of the DST to nonstationary local dynamics and its insensitivity to dynamic range. The right-most column of panels [H-k] displays the discrete wavelet transform of the original series demonstrating its comparatively less-sensitive nature to local anomalous dynamics.

defined for each t. The function $\mathcal{K}^{(\cdot)}$ is the shock kernel that is non-zero on $\tau \in [-W/2+t, W/2+t]$. For $t \in [-T, T]$, this can be rewritten equivalently as

$$\mathbf{C}_{\mathcal{K}^{(\cdot)}}(W|\theta) = \boldsymbol{K}(W|\theta)\boldsymbol{x}, \tag{D.3}$$

where $\boldsymbol{K}(W|\theta)$ is a $(2T+1) \times (2T+1)$ W-diagonal matrix, $\mathbf{C}_{\mathcal{K}^{(\cdot)}}(W|\theta)$ is the W-th row of the cusplet transform matrix, and $\boldsymbol{x}$ is the entire time series $x(t)$ considered as a vector in $\mathbb{R}^{2T+1}$. The matrix $\boldsymbol{K}(W|\theta)$ is just the convolution matrix corresponding to the cross-correlation operation with $\mathcal{K}^{(\cdot)}$. If $\boldsymbol{K}(W|\theta)$ is invertible, then it is clear that

$$\boldsymbol{x} = \boldsymbol{K}(W|\theta)^{-1}\mathbf{C}_{\mathcal{K}^{(\cdot)}}(W|\theta), \tag{D.4}$$

for any $1 < W < T$ and hence also

$$\boldsymbol{x} = \frac{1}{N_W}\sum_{W} \boldsymbol{K}(W|\theta)^{-1}\mathbf{C}_{\mathcal{K}^{(\cdot)}}(W|\theta). \tag{D.5}$$

This is an inversion formula similar to the inversion formulae of overcomplete transforms such as the DWT and discrete chirplet transform.

When $T \to \infty$ (that is, when the signal $x(t)$ is turned on in the infinite past and continues into the infinite future), this equation becomes the formal operator equation

$$\mathrm{C}_{\mathcal{K}^{(\cdot)}}(t, W|\theta) = \boldsymbol{K}(W|\theta)[x(t)], \tag{D.6}$$

and hence (as long as the operator inverses are well-defined),

$$x(t) = \frac{1}{N_W}\sum_{W} \boldsymbol{K}(W|\theta)^{-1}[\mathrm{C}_{\mathcal{K}^{(\cdot)}}(t, W|\theta)]. \tag{D.7}$$

These inversion formulae are, in our estimation, of relatively little utility in practical application. Whereas inverting a wavelet transform is a common task—it may be desirable to decompress an image that is initially compressed using the JPEG 2000 algorithm, which uses the wavelet transform for compact representation of the image—we estimate the probability of being presented with some arbitrary shocklet transform and needing to recover the original signal from it to be quite low; the shocklet transform is designed to amplify features of signals to which we already have access, not to recreate time-domain signals from their representations in other domains.

D.3 STAR and ADV comparison figures

We display comparisons between windows in which STAR indicates shock-like behavior and sets (at times, continuous windows) in which Twitter's ADV algorithm indicates anomalous behavior in the time series. We denote STAR windows with blue shading and ADV sets with red shading (and, hence, overlap with purple shading).

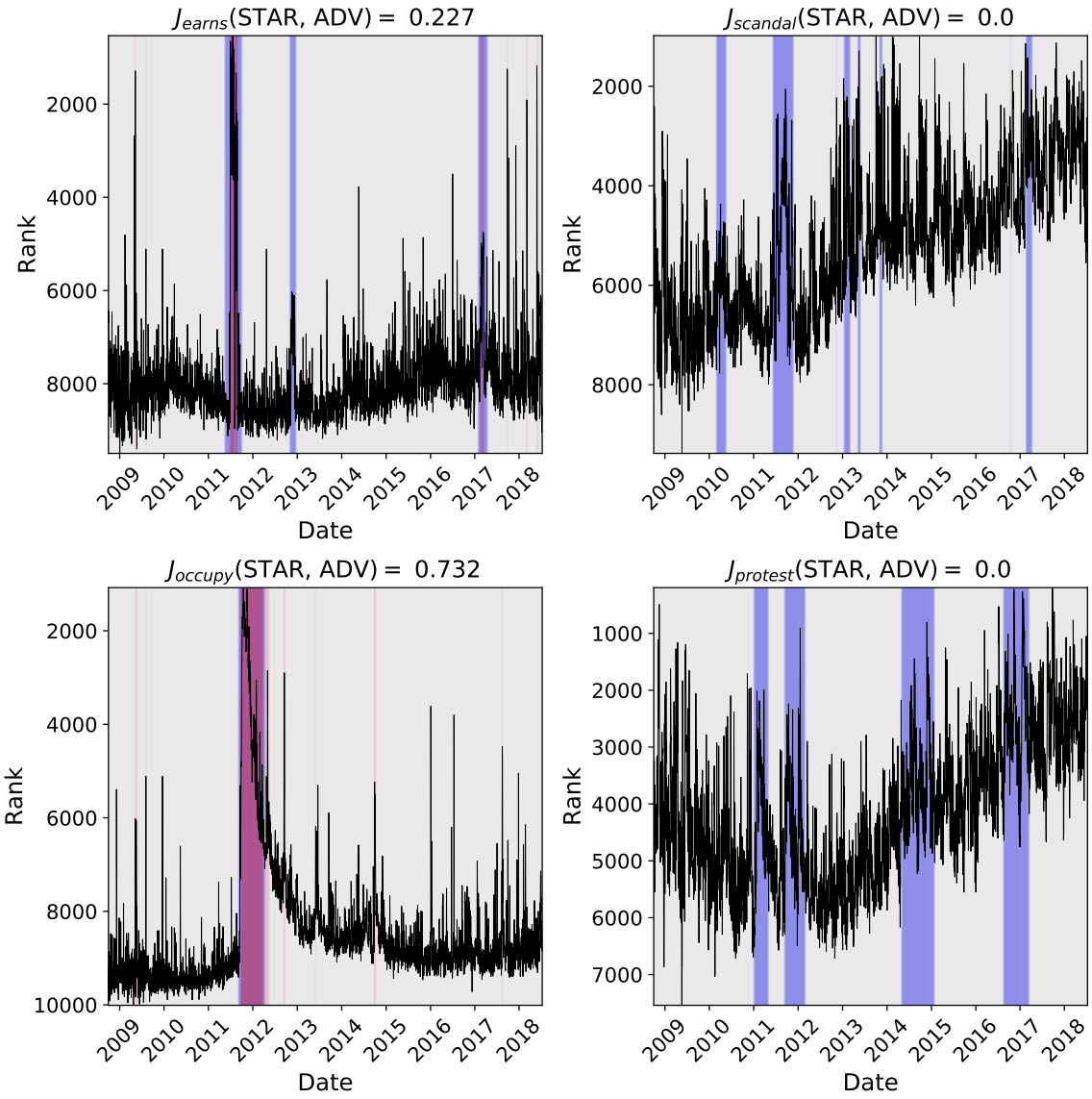

Figure D.4: Comparison of STAR and ADV indicator windows for some words surrounding the "Occupy Wall Street" movement during 2010.

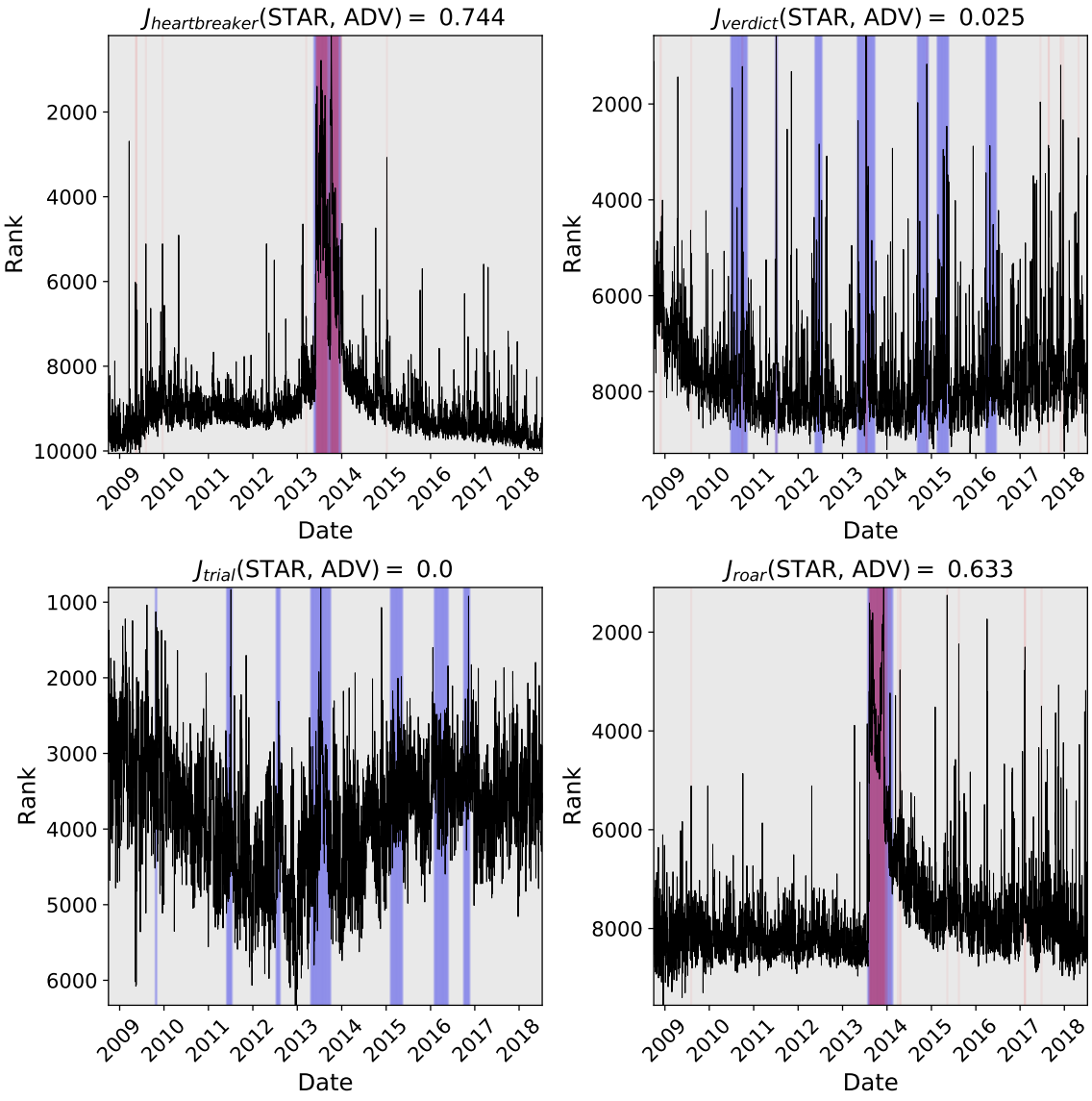

Figure D.5: Comparison of STAR and ADV indicator windows for some words surrounding popular events (the release of a song called "Heartbreaker" by Justin Bieber and "Roar" by Katy Perry) and criminal justice-related events (the trial and acquittal of George Zimmerman).

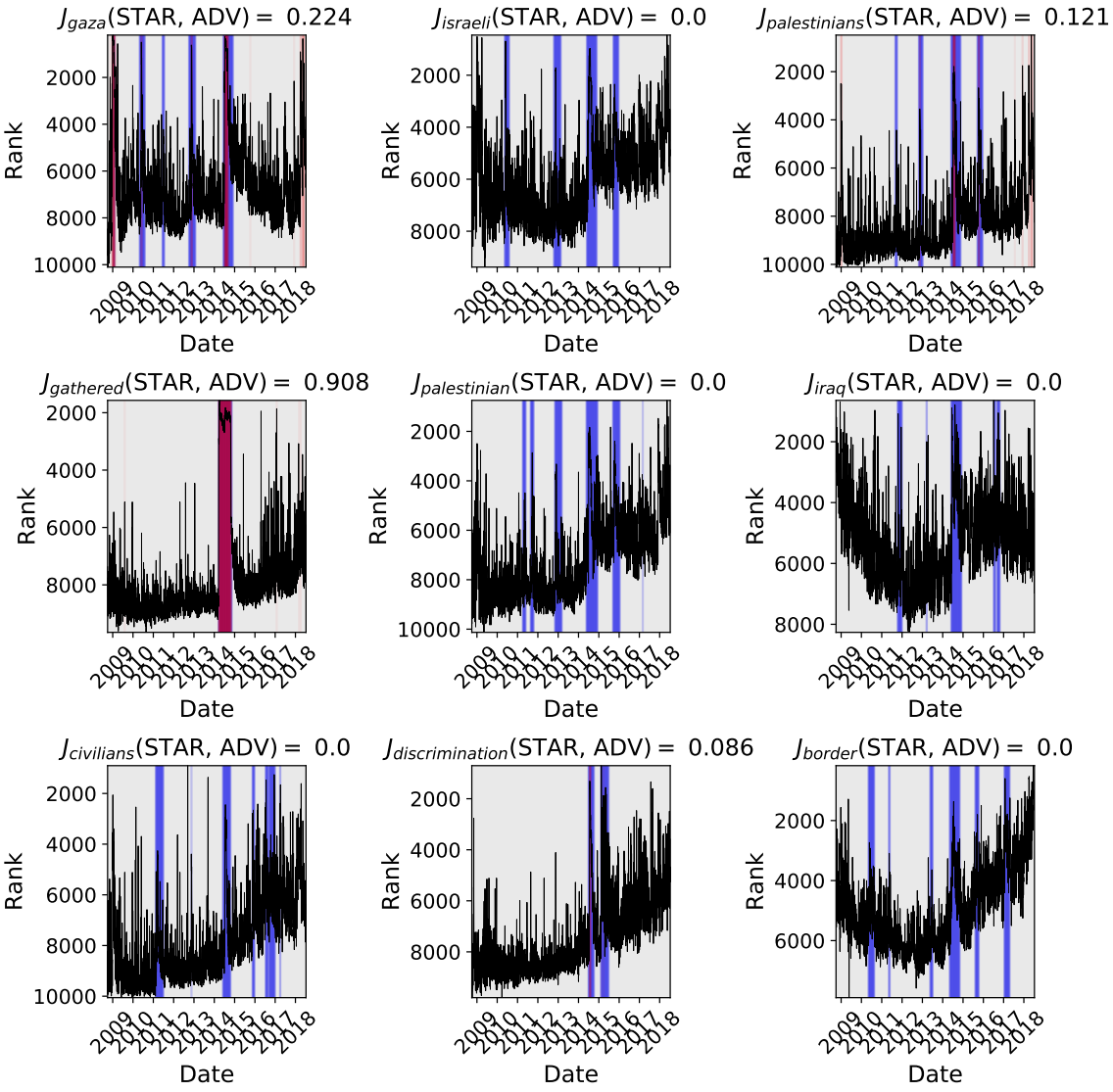

Figure D.6: Comparison of STAR and ADV indicator windows for some words surrounding the Gaza conflict of 2014.

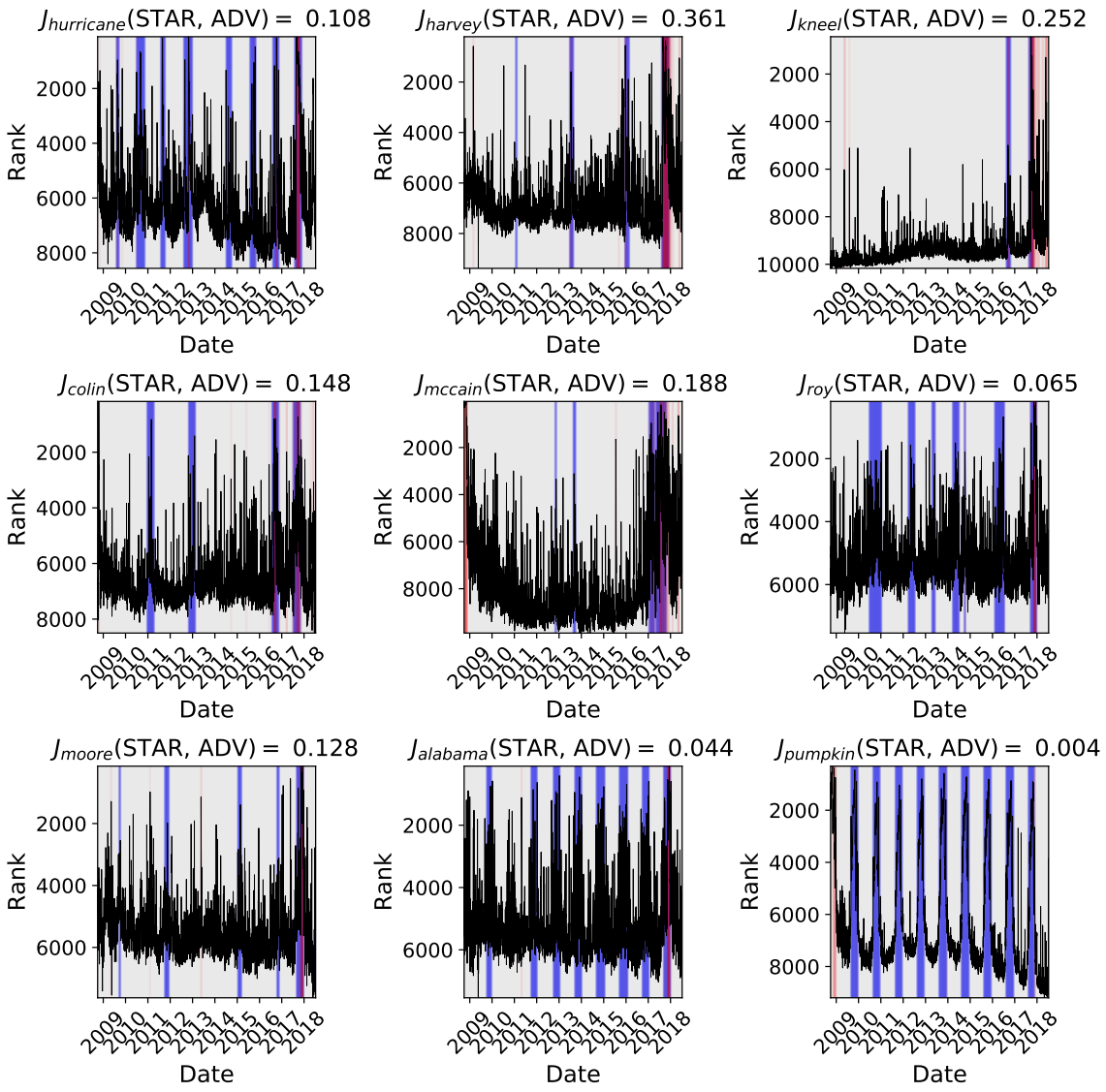

Figure D.7: Comparison of STAR and ADV indicator windows for some words surrounding the autumn of 2017, including Hurricane Harvey, Colin Kaepernick's kneeling protests, John McCain, the electoral campaign of Roy Moore in the U.S. state of Alabama, and pumpkins (a traditional gourd symbolic of autumn in the U.S.)