A Behavior-Driven Theory of Emergent Pattern and Structure in Complex Spatiotemporal Systems

By

Adam Thomas Rupe

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Physics

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Chair James P. Crutchfield

Richard Scalettar

David Doty

Committee in Charge

2020

To my grandparents, Ethel and Norman Rupe

Contents

List of Figures

List of Tables

Abstract

A Behavior-Driven Theory of Emergent Pattern and Structure in Complex Spatiotemporal Systems

Coherent structures form spontaneously in far-from-equilibrium spatiotemporal systems and are found at all spatial scales in natural phenomena from laboratory hydrodynamic flows and chemical reactions to ocean and atmosphere dynamics. Phenomenologically, they appear as key components that organize macroscopic dynamical behaviors. Unlike their equilibrium and near-equilibrium counterparts, there is no general theory to predict what patterns and structures may emerge in far-from-equilibrium systems. Each system behaves differently; details and history matter. The complex behaviors that emerge cannot be explicitly described mathematically, nor can they be directly deduced from the governing equations (e.g. what is the mathematical expression for a hurricane, and how can you derive it from the equations of a general circulation climate model?). It is thus appealing to bring the instance-based data-driven models of machine learning to bear on the problem. Supervised learning models have been the most successful, but they require ground-truth training labels which do not exist for far-from-equilibrium structures. Unsupervised models that leverage physical principles of self-organization are required.

The work developed in this thesis utilizes a notion of *intrinsic computation* to construct a physics-based machine learning model, the *local causal states*, to extract emergent pattern and structure in complex spatiotemporal systems. As a *behavior-driven* theory, it does so without requiring the governing equations or ground-truth training labels. After motivating the need for history-dependent, instance-based modeling for studying far-from-equilibrium phenomena, parallels between models of computation and complex dynamical system will be developed to argue for the use of machine learning models based on intrinsic computation. The mathematical foundations in symbolic dynamics and shift spaces is then given for the local causal states. Spacetime invariant sets are shown to be equivalent to spacetime symmetries in the local causal states. These behaviors, known as *domains*, capture pattern as generalized symmetries. Using this, the local causal states are used to

give a formal definition of coherent structures as spatially-localized, temporally-persistent deviations from generalized spacetime symmetries. The utility of the local causal states in capturing pattern and structure is demonstrated using cellular automata models and complex fluid flows (using both simulations and observations). The fluid flow results require high-performance computing; we will briefly describe our distributed implementation in Python and how we were able to process almost 90TB of data from the CAM5.1 climate model in under 7 minutes end-to-end on 1024 Intel Haswell nodes of the Cori supercomputer.

Acknowledgments

Fascination with the topics of chaos, nonlinear dynamics, nonequilibrium thermodynamics, and complexity lead me to pursue a PhD in physics. I am grateful to my PhD adviser Professor Jim Crutchfield for the inspiration and guidance he provided that enable me to succeed in this goal. His push for me to improve my precision of language and mathematical sophistication has been particularly invaluable. I could not have hoped for a more fitting and enjoyable thesis to do for my PhD research. I'd like to thank Professors Michael Marder and Harry Swinney for introducing me to these topics during my undergrad, and professor Marder for getting me started with research and programming. I also thank Professors Richard Scalettar and David Doty for their helpful comments and revisions for this thesis.

For my qualification examination I used the discovery of extreme weather events in large climate data sets as a motivating example for my work, and I am still astonished that I was able to get even preliminary results for such an ambitious pie-in-the-sky goal as part of my dissertation work. This was made possible through the DisCo collaboration, which was one of the academic partner projects for the NERSC-Intel Big Data Center. Mr. Prabhat (now Dr. Prabhat!) at NERSC was instrumental in establishing the Big Data Center and envisioning what would become Project DisCo. I am grateful to Prabhat for his mentorship and for introducing me to the exciting world of machine learning. I thank Karthik Kashinath for his friendship and infectious enthusiasm for data-driven approaches to dynamical systems. The distributed implementation of local causal state reconstruction presented in Chapter 7 would not be possible without Nalini Kumar, whom I thank for her friendship, patience, and for introducing me to high-performance computing. I also thank our other DisCo collaborators, who helped put all the pieces together to achieve the results presented in Chapter 7: Vlad Epifanov, Sacha Pavlyk, Frank Schlimbach, Mostofa Patwary, Sergey Maidanov, and Victor Lee.

My graduate school experience has been greatly enriched by discussion and interactions with Dmitry Shemetov, Greg Wimsatt, Grzegorz Muszynski, Paul Riechers, Alec Boyd, Cina Aghamohammadi, Sarah Marzen, Alex Jurgens, Mikhael Semaan, Fabio Anza,

Ariadna Venegas-Li, Sam Loomis, David Gier, Dany Masante, Jeff Emenheiser, Jordan Snyder, Xincheng Lei, Yao Liu, John Mahoney, Korana Burke, Ryan James, and many others. I am thankful to you all.

I thank Telluride Sciences Research Center, Institute for Pure and Applied Mathematics (IPAM), and National Energy Research Scientific Computing Center (NERSC) for their hospitality during visits. Thanks to Intel for their financial support of this work through the Intel Parallel Computing Center at UC Davis, as part of the NERSC-Intel Big Data Center.

I thank my father Ray and my brother Eric for their never ending support and encouragement. Special thanks to Nastya Salova for being awesome and keeping me sane during grad school. She has helped me become a better person. I'd also like to thank my cats Tycho and Kepler for their constant companionship and comfort throughout this journey. Finally, I must thank my grandparents Ethel and Norman, and my mother Beverly. Without their emotional and financial support this work would never have happened. I am forever grateful for all they've done for me.

Chapter 1

Pattern and Structure

What does it mean for a natural system to be structured? Through the entropy concept in statistical mechanics and information theory we have a means of formalizing the notion of randomness and disorder. But what about pattern and structure? It may be tempting to simply think of these as order, the opposite of randomness and disorder, as with the total regularity of a crystalline solid. What we find in nature however is rather more complicated than this single-dimension perspective. *Randomness can be structured; patterns can arise from chaos.* This can be seen, for example, with coherent structures that form in turbulent fluid flows. The recent images of Jupiter from the Juno spacecraft provide particularly dramatic examples. An image of Jupiter's Great Red Spot [7], taken by Juno, is shown in Figure 1.1.

This Chapter serves as an introduction to the problem of spontaneous self-organization in far-from-equilibrium systems and gives an overview of the tools developed and used in this thesis. The historical context and technical details outlined here are meant to provide more of a personal perspective motivating this work, rather than a comprehensive review. Sections 1.1 and 1.2 provide contextual background, and the remaining Sections provide an overview of the technical ideas presented in this thesis.

The terms "organized", "structured", and "patterned" are used somewhat loosely throughout, as one might colloquially use to describe the clouds of Jupiter shown in Figure 1.1. A more technical discourse on the notion of *pattern* is given in Section 3.1.6 as a generalization of exact symmetry. In Chapter 4 we expand this to define spatiotemporal

Figure 1.1. Color-enhanced compilation image of Jupiter's clouds captured by NASA's Juno Spacecraft, taken on April 1 2019. While locally turbulent and chaotic at the smallest scales, there is obvious large-scale organization. A few of the large horizontal zonal bands can be seen, including the strong jet streams that separate adjacent bands. Coherent vortices at all scales can be seen throughout, including the largest: Jupiter's famous Great Red Spot.

patterns as statistically-regular regions of generalized symmetries, which we refer to as *domains*. The definition of domain is then used in Chapter 6 to rigorously define *coherent structures* as persistent and localized deviations from these generalized spacetime symmetries. The formal definitions of domain patterns and coherent structures are seen as forms of macroscopic self-organization in the system.

1.1 In Nature

In contrast with much of modern physics, which concentrates on the very small (quantum mechanics) and the very large (cosmology), pattern and structure exist at the human scale. Perhaps in part because we experience these phenomena on a daily basis, it is easy to overlook that such collective organization is just as mysterious and unintuitive

as quantum mechanics and cosmology. When Boris Belousov discovered an oscillating chemical reaction in the early 1950s, he was unable to publish because the established wisdom at the time deemed this impossible, as it was seen as a violation of the 2nd Law of thermodynamics, even though he provided the recipe for others to reproduce his experiment [8]. It was perfectly acceptable for, say, clear chemical reagents to mix and turn opaque. It was thought impossible for these reagents to apparently un-mix and the solution turn clear again, as happens with the "chemical clock" discovered by Belousov. Around the same time, however, Alan Turing showed mathematically that such chemical oscillations can exist in open systems, for which the 2nd Law does not apply, and could in fact be derived from simple models of reaction-diffusion systems [9] (the subtleties of nonequilibrium processes in open systems which cannot be described by states that are instantaneously in equilibrium were not well understood at this time).

This mysterious and unintuitive phenomenon, now know as *pattern formation* and *spontaneous self-organization*, is not limited to chemical reactions. Patterns and structures abound in nonequilibrium systems across all spatial scales [10, 1], from galactic structures to planetary – such as Jupiter's famous Red Spot [11, 12] and similar climatological structures on Earth [13] – down to the microscopic scales of snowflakes [14] and bacterial [15] and crystal growth [16]. Fifty years prior to the chemical oscillations studied by Belousov and Turing, at the turn of the 20th century, the formal study of pattern formation and spontaneous self-organization began with Bénard's work on convective instabilities in fluids [17], followed soon after by Lord Rayleigh [18] and G.I. Taylor [19], and others later still [20, 21, 22, 23].

1.1.1 Near Equilibrium

To illustrate the basic concepts of pattern formation, consider the original Rayleigh-Bénard (RB) convection system, which is simply a box of fluid heated from below. See Figure 1.2. When the temperature gradient is low, heat is transported up through the fluid via conduction, and the fluid velocity is everywhere zero. As the temperature gradient is increased it will eventually reach a critical value and a moment of magic occurs. All of a sudden, and all at once, the fluid conspires to form convective columns in, for

example, a hexagonal lattice, as convective transport becomes more favorable than conduction. As seen in Figure 1.3, the patterns of the convective columns are clearly visible to the human eye and so the characteristic length scales of the patterned state is much, much (about a million times) larger than those of the constituent water molecules. The thermodynamic mechanism governing the critical point at which convective transport becomes more favorable depends on whether the fluid fully fills a closed tank, as depicted in Figure 1.2, or whether it is a liquid with a free surface open to atmosphere, as is the case with the convection cells shown in Figure 1.3. See Section 1.6.1 for further discussion.

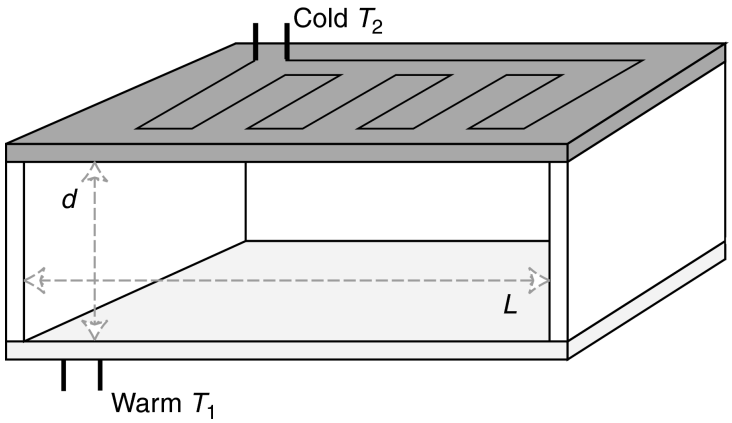


Figure 1.2. Diagram of the experimental setup for Raleigh-Bénard convection. A box of fluid is heated from below and cooled from above. Reproduced, with permission, from Ref. [1].

We will refer to the spontaneous onset of patterns out of equilibrium, like the situation just described, as *nonequilibrium phase transitions* [24, 25], due to the many similarities with the more familiar equilibrium phase transitions. While emergent patterns can form in equilibrium systems, such as strongly correlated electron materials [26, 27], we will focus solely on nonequilibrium systems for this thesis. A crucial distinction, described in more detail below, is the state selection criteria provided by the 2$^{\text{nd}}$ Law of Thermodynamics that governs pattern formation in equilibrium. That said, much of the work presented in this thesis concerns the nature of pattern and structure, regardless of their physical origins. For instance, macroscopic correlation lengths — the spatial scale of coordination among system constituents — are characteristic of pattern formation in and out of equilibrium, but arbitrarily large correlation lengths may occur during equilibrium

phase transitions which do not produce patterns [28]. Indeed, scale-invariant states with diverging correlation lengths must be distinguished from patterned states. We call it *patterned* exactly because there is some level of intricate *organization* that can not be captured by simple quantification of correlation length. Providing a formal mathematical accounting of this notion of organization is the main goal of this thesis.

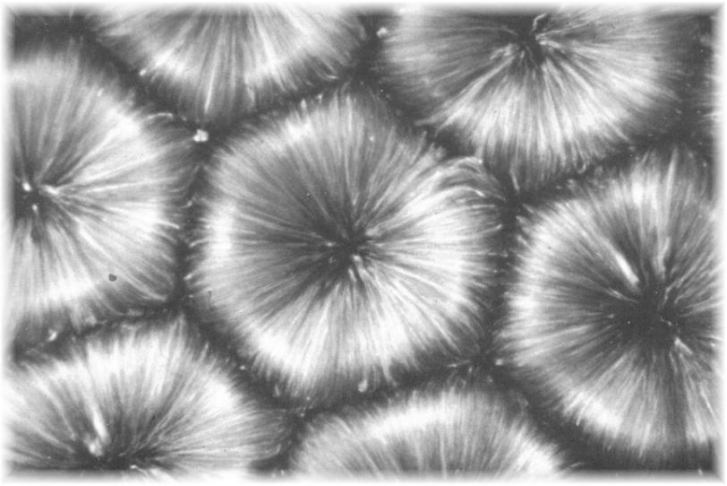


Figure 1.3. Close up image of a time lapse view of hexagonal cells in free surface Bénard convection. Aluminum flakes were used to trace the fluid velocity of the convective motion; the fluid moves up through the center of the cells and sinks at the edges. Image used with the permission of *The Parabolic Press* [2].

How does nonequilibrium pattern formation happen? While it is unknown how such a large number of molecules can all spontaneously organize collectively into intricate macroscopic patterns and structures, there is a lot that we do understand about the physics taking place, particularly in the near-equilibrium regime where patterns first emerge. We can understand the onset of patterns using *bifurcation theory* [29, 1, 30, 31], in close analogy to second-order equilibrium phase transitions.

Patterns are born out of conflict and compromise. There are two competing forces at play whenever patterns form: the inexorable pull towards thermodynamic equilibrium (due to the 2$^{\text{nd}}$ Law) and an external push away from equilibrium (gradients in intensive quantities drive fluxes in extensive quantities). We define a dimensionless *bifurcation parameter* that is the ratio of these two competing forces and measures the distance from

thermodynamic equilibrium

$$R \propto \frac{\text{driving}}{\text{dissipation}} \ . \tag{1.1}$$

This signifies that driving and dissipation are crucial ingredients for nonequilibrium pattern and structure (which is why they have sometimes been known as *dissipative structures* [32, 33, 34], though we will not use that term here). A constant supply of energy, and potentially other similar quantities, must be continuously pumped into the system and simultaneously dissipated away to maintain nonequilibrium pattern and structure. For RB convection, R is the Rayleigh number and is proportional to $\frac{\Delta T}{\nu\kappa}$, where ν is the kinematic viscosity and κ the thermal diffusivity (additional constants are included to non-dimensionalize R).

We assume the system is governed at the macroscopic level by some effective field theory, given as a set of nonlinear partial differential equations [1]

$$\partial_t U(x,t) = F\big(U(x,t), \partial_x U(x,t), ...; R\big) \ . \tag{1.2}$$

The state of the system is given by U and varies smoothly and continuously in space x and time t. Time evolution, as given by Equation (1.2), is governed by local interactions that are applied uniformly in space and time (i.e. do not have explicit space or time dependence) and depend on external conditions as encapsulated by the bifurcation parameter R. The local interactions are assumed to extend over a finite distance and thus are a function of finitely many spatial derivatives of U. For fluid flows, these are the Navier-Stokes equations, or some approximation thereof.

With fixed, gradient-free boundary conditions we have $R = 0$ and the system will reach thermodynamic equilibrium with U uniform in space and time. If the boundary conditions are fixed with non-zero gradients we have $R \neq 0$ and the system reaches a *nonequilibrium steady-state*. Close to equilibrium, with R small but non-zero, the nonequilibrium steady-state is known as the *base state*, which we denote U^*. The base state shares the symmetries of the boundary conditions, so if the boundary conditions are time-independent then the base state will also be time independent

$$\partial_t U^*(x,t) = 0 \ . \tag{1.3}$$

Below a critical value of R, R_c, the base state is stable. All infinitesimal perturbations, $\delta U(x,t) = Ae^{\sigma t}e^{ik\cdot x}$, exponentially decay. At R_c the growth rate of a perturbation at wavenumber k_c becomes zero, and for $R > R_c$ this perturbation grows. The base state becomes linearly unstable at R_c and the patterned state with wavenumber k_c becomes the stable nonequilibrium steady-state for $R > R_c$ [1, 35].

In the case of RB convection, $R = 0$ for zero temperature gradient and the resulting equilibrium state has a uniform temperature equal to the temperature of the boundary and fluid velocity zero everywhere. For small $R > 0$ the base state also has fluid velocity everywhere zero, but now a linear temperature profile in the vertical direction, as dictated by the Fourier heat law. The conduction state (the base state) becomes unstable at R_c and for $R > R_c$ the convection state with wavenumber k_c takes over and becomes the stable patterned state.

These three states, the equilibrium state, the base state, and the patterned state, characterize nonequilibrium phase transitions that pass through them. The key feature that distinguishes the three states is *symmetry*. The equilibrium state is defined by fixed, uniform boundary conditions that smooth out any gradients in the system, thus making the equilibrium state uniform in space and time. This is the state of *maximal symmetry*. Introduction of non-zero gradients on the boundary give rise to the base state. This breaks the system symmetry in the direction of the flux driven by the gradients. For RB convection the temperature gradient of the box induces a vertical heat flux, breaking the symmetry of the temperature field in the vertical direction. However, the temperature field in the horizontal direction remains uniform for a fixed vertical position. As the driving is increased further, and R increases past the critical value R_c, the patterned state emerges and is characterized by symmetry breaking in directions orthogonal to the driven fluxes. The hexagonal convection cells in the patterned state of RB convection tile the horizontal directions. We see that the onset of patterns in a nonequilibrium phase transition occurs through a sequence of progressively broken symmetries. The concept of *broken symmetry* is an integral component of the theory developed here, and we shall return to this idea several times throughout.

As mentioned above, there are some strong analogies between equilibrium phase transitions and nonequilibrium phase transitions. Both involve a rapid change in qualitative behavior of the system as some control parameter passes through a critical value. In both cases, the change in behavior involves breaking symmetry. A selection must be made when symmetry is broken in both cases, and the selection is dictated by random, microscopic fluctuations. When a spin system cools to the magnetic phase, the system can either have all the spins align up or all align down. Similarly, when convection sets in for the RB system, the convection columns have to organize so that adjacent columns flow in different directions, but which columns flow up and which flow down depends on the detailed conditions of the system leading up to the instability. This influence of small fluctuations becomes important in the far-from-equilibrium regime.

From renormalization theory, we understand there is a *universality* to equilibrium phase transitions [28]. The specific microscopic details of a system are inconsequential for determining its critical behavior. There is a similar universality for nonequilibrium phase transitions due to generic forms of bifurcation that can occur at R_c [29]. For example, static striped patterns can occur in RB convection and the Belousov-Zhabotinsky reaction-diffusion system, because the imaginary component of the temporal exponent of the critical mode is zero for both.

The most significant distinction between the two types of phase transitions is found in their names. Being in thermodynamic equilibrium, or not, makes a world of difference. In equilibrium, thermodynamic quantities are always well-defined and crucially the 2^{nd} Law applies. Measuring entropy or free energy values is so important in equilibrium thermodynamics because the 2^{nd} Law provides a *selection principle* of the unique equilibrium state in terms of these quantities: the equilibrium state is that which maximizes entropy or, equivalently, minimizes free energy, subject to boundary constraints. It can not be stressed enough that *there is no thermodynamic selection principle for nonequilibrium states* [36, 37]. Extremum principles based on entropy production have been proposed, being perhaps the most straightforward nonequilibrium generalization of the 2^{nd} Law, but it has been shown that extremizing entropy production can not provide a universal

selection criteria for nonequilibrium states [38, 39].

Selection of the patterned state in nonequilibrium phase transitions comes not from a thermodynamic state-selection principle, but as we've seen, from analysis of the dynamical equations of motion. Linear stability analysis shows that a critical mode begins to grow as the system moves through R_c, and perturbation theory (e.g. amplitude equations [40, 1]) shows how this growing mode saturates to create the patterned state. Which mode grows and how it saturates is dictated, in this close-to-equilibrium regime, by the geometry of the boundary conditions.

Notice that equilibrium thermodynamics is a static *theory of state* and what might cause one state to change into another state. Time is not a thermodynamic variable. Nonequilibrium, in contrast, is concerned with *process*. Even to determine time-independent nonequilibrium states, we must invoke the dynamical equations of motion. Prigogine describes it succinctly [41]: equilibrium is about *being*, while nonequilibrium is about *becoming*. If there is to be a nonequilibrium selection principle, it will be one of process, not of state. A leading candidate for such a theory employs information theory to define a time-dependent thermodynamic entropy [39, 42], or path entropy [43], from which to generalize the 2^{nd} Law: the nonequilibrium state at time t is the final state that results from the process that maximizes the time-dependent entropy from the initial state at time t_0, subject to macroscopic constraints of the process (e.g. thermal driving). While this information-theoretic framework, known as the Principle of Maximum Entropy, adds a satisfying logical foundation to equilibrium statistical mechanics and nicely generalizes to linear near-equilibrium thermodynamics, it does not make novel thermodynamic predictions outside of these already-understood regimes. Near-equilibrium thermodynamics is recovered using linear perturbation theory. Predictions for the far-from-equilibrium frontier are thus out of reach, as the necessary calculations are intractable. We will see that this kind of intractability is characteristic of the far-from-equilibrium regime where we find patterns and structure in the natural world.

1.1.2 Far From Equilibrium

Here we use the term nonequilibrium phase transition to refer to the *primary bifurcation* that occurs close to equilibrium. As described above, this is when the base state (the nonequilibrium steady-state closest to equilibrium) first loses stability. The new stable state that appears after this first instability is what we have referred to as the patterned state. This state however can also lose stability as R is increased further beyond R_c and subsequent bifurcations occur. Phenomenologically, the resulting states have a further reduction in symmetry and exhibit more intricate patterns.

Nonequilibrium phase transitions have been thoroughly studied because they provide a setting to study the onset of patterns that is amenable to mathematical analysis and experimental investigation. This is due to simplified boundary conditions and the simplicity of the base state, which is uniform in the orthogonal direction. However, most of the interesting pattern and structure we observe in the natural world, such as in the clouds of Jupiter seen above, occur far from equilibrium with $R >> R_c$. In the framework of bifurcation theory, we can understand natural pattern and structure as resulting from a series of many bifurcations that progressively reduce symmetry and increase complexity. In fact, most natural patterns are far enough from equilibrium that all semblance of symmetry in the observed patterns has been eliminated.

There is no general theory to predict what patterns and structures may emerge far from equilibrium. As stated by Harry Swinney, "Far beyond the primary instability, each system behaves differently. Details matter... There is no universality" [35]; and from Philip Ball, "... the patterns of a river network and of a retinal nerve are both the same and utterly different. It is not enough to call them both fractal, or even to calculate a fractal dimension. To explain a river network fully, we must take into account the complicated realities of sediment transport, of changing meteorological conditions, of the specific vagaries of the underlying bedrock geology–things that have nothing to do with nerve cells." [10]

Thinking of far-from-equilibrium patterns emerging after a sequence of bifurcations of the near-equilibrium base state gives us a conceptual framework for beginning to un-

derstand why *details and history matter* far from equilibrium. We saw above that when symmetry is broken during a bifurcation a selection must be made by the system when a patterned state grows and stabilizes (e.g. directions of convective columns in the RB system). Far from R_c, not only must selections be made for individual patterned states, but the selection of the new stable patterned state after the bifurcation is determined by microscopic fluctuations; unlike nonequilibrium phase transitions, a continuum of patterns may have positive growth rate when the current state becomes unstable during a bifurcation far beyond the initial instability. If the system goes through several symmetry-breaking bifurcations, many such compounding selections will be made. The resulting far-from-equilibrium state will depend on the details leading to each individual selection, as well as the history of all selections made.

This is not necessarily how far-from-equilibrium patterns form in nature. We know the patterns in Jupter's clouds did not arise from an irrotational ball of homogeneous gas initially isolated from the Sun, with radiative flux and angular velocity slowly turned up to their present values. Nonetheless, it is not hard to imagine that the emergence of the observed patterns in Jupiter's atmosphere similarly depended heavily on the details of Jupiter's formation and the specific history of its atmospheric dynamics following formation.

In either case, far-from-equilibrium patterns and structures present an enormous and unique challenge. Equilibrium and nonequilibrium phase transitions provide a nice framework for conceptualizing far-from-equilibrium patterns, but their technical tools breakdown and fail in this regime.

Historically, the primary objective in physics is discovering fundamental laws, given as mathematical equations. Newtons laws of motion and gravitation, Maxwell's equations of electro-magnetism (which unify several physical laws discovered prior), the Navier-Stokes equations of fluid dynamics, Einstein's laws of relativity, the standard model of particle physics. Pattern formation, particularly in fluid dynamics, shows that knowing the equations governing complex phenomena is insufficient for full understanding. Despite knowing the equations of fluid dynamics for over a century and intense interest in a pattern

forming system like Taylor-Couette flow, the full phase space of possible behaviors for this system is still poorly understood [35]. Even outside of pattern formation, it has been observed that knowing the equations of motion is not always sufficient. As Freeman Dyson said of general relativity, "It often happens that the understanding of the mathematical nature of an equation is impossible without a detailed understanding of its solutions. The black hole is a case in point. One could say without exaggeration that Einstein's equations of general relativity were understood only at a very superficial level before the discovery of the black hole." [44]

For far-from-equilibrium patterns, the situation is even more troubling. Knowledge of particular solutions of the system can still be insufficient for understanding. If analytic solutions can be obtained for complex behaviors, the solutions may be too complicated to provide much insight. This was the case for a solution of the constrained Euler beam problem that involved pages of elliptic integrals [35]. Numerical solutions from computer simulations are now an integral part of science that allow for the analysis of complex systems for which analytic solutions can not be obtained. Having particular numerical solutions can still not be enough. Fluid turbulence is a prime example. Despite knowing the Navier-Stokes equations for over a century and decades of countless numerical solutions, turbulence remains a persistent mystery.

For the purposes of this thesis, if a particular system behavior can not be deduced from the equations of motion governing the system, we refer to this behavior as *emergent*. We may observe that the governing equations produce some emergent behavior through simulation, but the current tools of physics can not tell us how the physical principles encapsulated by the governing equations give rise to the observed behavior. Further, an emergent behavior might be quite *complex* and thus difficult to even describe mathematically. Simple striped patterns that arise out of the primary instability in RB convection may be easily described with Fourier modes. This makes the linear instability analysis of this pattern tractable. Consider though the Great Red Spot of Jupiter. It is a dynamic and irregular structure with diffuse boundaries. Fully describing the Great Red Spot with Fourier modes is, in practice, infeasible. Section 1.6 gives a more detailed discussion of

complexity and emergence.

The work developed in this thesis is an attempt to make progress towards understanding natural patterns and structures in the challenging far-from-equilibrium regime. Giving a formal mathematical accounting of complex pattern and structure, particularly localized *coherent structures*, like the Great Red Spot of Jupiter, is the main emphasis. The tools developed for this purpose will hopefully, in the future, be able to help elucidate the physical and causal mechanisms that give rise to complex pattern and structure far from equilibrium.

While there is no general theory for systems far from equilibrium, there are some basic principles from what little is known about the physics of self-organization that we will make use of. Knowing the equations of motion or even specific solutions is not sufficient; they must be supplemented with new analysis tools. Because specific details for each system matter, the instance-based models of *machine learning* are appealing for this task as they learn from, and apply to, specific system instances. In the context of dynamical systems, we are interested in *behavior-driven* (sometimes also called data-driven or equation-free) modeling. Below we will give a brief overview of machine learning and its relevance for this work, then we will motivate our particular behavior-driven model based on *intrinsic computation* — an extension of statistical mechanics that uses computation theory to capture pattern and structure in dynamical behaviors.

1.2 In Data

A new data-driven paradigm is beginning to take shape, centered around machine learning (ML), which has shown to be useful for scientific applications where the governing equations are not known and we need to extract insight from noisy or imperfect data. Perhaps less appreciated is the potential of using ML to study complex systems with well known equations and high-fidelity simulation data, like cellular automata and fluid flows, as a means of circumventing the difficulties these systems pose to traditional scientific inquiry.

Machine learning has been remarkably successful in commercial applications, due to its unprecedented ability to "find patterns in data". If our interest is in finding patterns and structure in natural systems, can we do this by running data from natural systems through a machine learning algorithm? Unsurprisingly, it is not so easy.

While machine learning has already seen some success in scientific application, particularly in automated curation of large datasets [45, 46, 47], current ML models and techniques that do not incorporate physical insights are insufficient for scientific *discovery* and *understanding* [48, 49]. For a data-driven scientific paradigm[1] to stand equal alongside the hypothesis-driven paradigm, new data-driven methods are required that discover and mathematically describe complex emergent phenomena, uncover the physical and causal mechanisms underlying these phenomena, and are better able to predict these phenomena and how they evolve over time.

1.2.1 Machine Learning

Machine learning is a large, rapidly expanding field of study [51, 52]. Here we will try to give a brief overview of the relevant concepts; the details are largely unimportant for our development. The goal is to introduce some of the difficulties with scientific ML and give context for how the work developed in this thesis fits into the ML framework.

As a motivating example, consider the common task in computer vision to decide

[1] I do not embrace the "End of Theory" thesis [50]. Rather, data-driven science requires a new modality of "theory" and "modeling" to extract actionable insight and understanding directly from data. This is in contrast to the use of data just to verify and validate insight and understanding from theoretical models.

whether a given object is present in an image or not. For example, is there a cat in an RGB image that is 256 pixels by 256 pixels in size? We can formalize this by seeking a function $f : \mathbb{R}^{3\times256\times256} \to \{0, 1\}$, where f outputs a 1 if there is a cat in the input image and a 0 otherwise. Trying to design such a function by hand is a daunting task. Machine learning algorithms instead attempt to *learn* (approximations of) such functions. If we cannot write down an explicit mathematical expression for complex structures in far-from-equilibrium systems, perhaps we can design an algorithm to learn such an expression.

But what does it mean for an algorithm to learn? From Ref. [53], "A computer program is said to learn from *experience* E with respect to some class of *tasks* T and *performance* measure P, if its performance at tasks in T, as measured by P, improves with experience E".

1.2.1.1 Tasks

The most common task in ML, as with the cat-in-an-image example, is *classification*. That is, we seek a function that maps a high-dimensional input into some discrete categories or classes, typically just a handful: $f : \mathbb{R}^N \to \{1, ..., k\}$. Rather than just decide simply whether an object is in an image, we can ask our algorithm to tell us *where* an object is in an image. Bounding-box localization is the simplest; as the name suggests the algorithm tries to return the original image with a tight bounding box around the desired object if it is present in the image (i.e. find the smallest sub-image such that f on that sub-image still gives the object class label). The most complicated localization task is *segmentation*, which gives a pixel-level identification of the desired object if it is present. Formally, an image segmentation algorithm learns a function that maps each pixel in the input to a class label. A binary segmentation for the cat-in-an-image problem would map each pixel to a class of either `cat-pixel` or `not-cat-pixel`. The task of coherent structure discovery that we will deal with in this thesis can be framed as a spacetime segmentation problem: given a spacetime field input we would like a pixel-level identification of any coherent structures present in the field.

Another common task, *regression*, can be considered as a generalization of classification with a continuum of categories: $f : \mathbb{R}^N \to \mathbb{R}$. This is most commonly encountered in

science applications as curve fitting. There are many other types of tasks in machine learning which we will not discuss here.

1.2.1.2 Experiences

Most of the current commercial success of machine learning has come in the form of *supervised learning*, where the algorithm learns from ground-truth labels y given in a training dataset: $(\mathbf{x} \in \mathbb{R}^N, y \in \{1, ..., k\})$. For example, humans have hand-labeled thousands of images to generate ground-truth training data for image classification learning (e.g. a human labels a training image 1 if it contains a cat and 0 if it does not) [54]. This presents an immediate difficulty for scientific applications, where *ground-truth often does not exist*. In fact, the whole notion of scientific *discovery* precludes the notion of a ground-truth training dataset. Discovery is the act of learning something totally new and often unexpected.

Discovery in machine learning can only be achieved through *unsupervised learning*, where the algorithm learns just from unlabeled inputs $\mathbf{x} \in \mathbb{R}^N$. This then relies on exploiting some notion of structure in the data. Presently we are interested in discovering physical structure and pattern in spatiotemporal data. This is a fundamentally unsupervised problem; there is no ground-truth. In essence, we seek to *define* a ground-truth for coherent structure segmentation from physical principles.

1.2.1.3 Performance

Let's briefly return to supervised learning. Consider a parametric classifier model $f(\mathbf{x}, \vec{\theta})$ that gives a class label y for a given input $\mathbf{x}$ and fixed set of parameters $\vec{\theta}$. We can train the model using ground-truth labels in a training dataset by defining a *loss function* $L\big(f(\mathbf{x}, \vec{\theta}), y\big)$ and *error rate* $J(\vec{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim\hat{\mathrm{P}}_{\mathrm{data}}} L\big(f(\mathbf{x}, \vec{\theta}), y\big)$. The error rate quantifies how often the model gives the correct label for a given training input. Typically max likelihood, $J(\vec{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim\hat{\mathrm{P}}_{\mathrm{data}}} \log \mathrm{P}_{\mathrm{model}}(y|\mathbf{x})$, or mean-squared-error, $J(\vec{\theta}) = \frac{1}{2}\mathbb{E}_{(\mathbf{x},y)\sim\hat{\mathrm{P}}_{\mathrm{data}}} ||y - f(\mathbf{x}, \vec{\theta})||^2$, are used. Because $\hat{\mathrm{P}}_{\mathrm{data}}$ is the empirical distribution from the training set, $J(\vec{\theta})$ is the *training error*. The model is trained by tuning the parameters $\vec{\theta}$ to minimize training error. Typically $J(\vec{\theta})$ is differentiable so that the minimization can be done through gradient descent.

Machine learning is distinct from pure optimization because what we actually care about in machine learning is minimizing an objective function (e.g. error rate) according to the *data-generating distribution* $\mathrm{P}_{\mathrm{data}}$ that $\hat{\mathrm{P}}_{\mathrm{data}}$ is sampled from. This is known as *generalization* – the ability to perform well, after training, on previously unseen data. We want to fit a model that not only makes training error small, but makes the gap between training and test error small. Underfitting of an ML model occurs when training error is too large, and overfitting is when training error is small but the gap between training error and test error is too large. Generalization performance is often described in terms of a trade-off between *bias* and *variance*. Bias represents erroneous assumptions in the model that prevent it from finding relevant relations between inputs and outputs, and thus leads to underfitting. Variance represents the sensitivity of f to small differences in input that can cause the model to overfit to random fluctuations in the training data. See Ref. [55] for an in-depth discussion.

How can we train unsupervised algorithms without ground-truth labels to compare against? More troubling, how can we evaluate the effectiveness of a trained model? This is an extremely difficult problem and something we will grapple with throughout.

In some unsupervised problems the input data can actually be used to provide labels for itself and thus give a loss function that can be used to train parametric models. This is known as *self-supervision*, and as an example can be used for time series prediction where the prediction error provides a loss function. Self-supervision however can not be directly used in this way to train a parametric model to discover pattern and structure in physical data. (As we will see though, one of the physical insights that will be discussed further in Section 1.3 allows for something akin to the predictive self-supervision just discussed. While not done here, this opens the possibility of defining a loss function to train parametric models like artificial neural networks.) Without any kind of error metric to optimize, we must rely on physical insights to create nonparametric models.

1.2.2 Representation Learning

Much of the effort in machine learning comes in the form of learning good representations for the input data [56]. Intuitively, we can imagine that there are some key *latent features*

of a cat (e.g. whiskers, pointy ears, etc.) that makes them easily recognizable in an image. If an ML algorithm can identify these features, it is much easier to determine if there is a cat or not.

Let's state this idea of *feature extraction* a bit more formally. Consider data points $\mathbf{x} \in \mathbb{R}^N$ that are split into two classes y_1 and y_2. If one can insert a hyperplane into $\mathbb{R}^N$ such that all points in class y_1 are on one side of the hyperplane and all points in y_2 are on the other side, these classes and their labeled data points are said to be *linearly separable*. Such classification problems are easy, as algorithms like support vector machines [57] exist that are guaranteed to converge to an optimal classifier for linearly separable data. Most classification problems are not linearly separable, but if we can find a transformation $\varphi(\mathbf{x})$ of the data (known as a *feature map*) such that points in the transformed space (known as the *feature space* or *latent space*) are linearly separable, then the classification task becomes easy again.

To demonstrate, consider the toy example of binary circle classification in $\mathbb{R}^2$ where points inside the circle of radius R are in class y_1 and points outside the circle are y_2. There is no way to draw a straight line (a hyperplane in $\mathbb{R}^2$) that separates points in y_1 from points in y_2. Let $\mathbf{x}^i = (x_0^i, x_1^i)$ denote the i^{th} data point with components x_0^i and x_1^i. If we keep these two components in $\mathbb{R}^2$ as features, but add an additional radial feature, $(x_0)^2 + (x_1)^2$, then data points in the three-dimensional feature space of $\varphi(\mathbf{x}) = (x_0, x_1, x_0^2 + x_1^2)$ can be separated by a hyperplane at R in the third dimension. Points in class y_1 will lie below the plane and points in y_2 will be above.

In most real-world problems, an appropriate choice of the feature map $\varphi(\mathbf{x})$ is not so obvious. Kernel methods are a powerful class of ML models for which the feature map need not be given explicitly because the model only requires inner products in feature space. The inner products can be specified via a kernel function $k(\mathbf{x}^i, \mathbf{x}^j) = \langle \varphi(\mathbf{x}^i), \varphi(\mathbf{x}^j) \rangle$. If $k(\mathbf{x}^i, \mathbf{x}^j)$ is a positive definite kernel (i.e. satisfies Mercer's condition [58]) then it implicitly defines a feature map. In some cases, as with the Gaussian kernel $k(\mathbf{x}^i, \mathbf{x}^j) = \exp(-\frac{||\mathbf{x}^i - \mathbf{x}^j||^2}{2\sigma^2})$, the implicit feature space is infinite dimensional [59] and thus likely to linearize the problem.

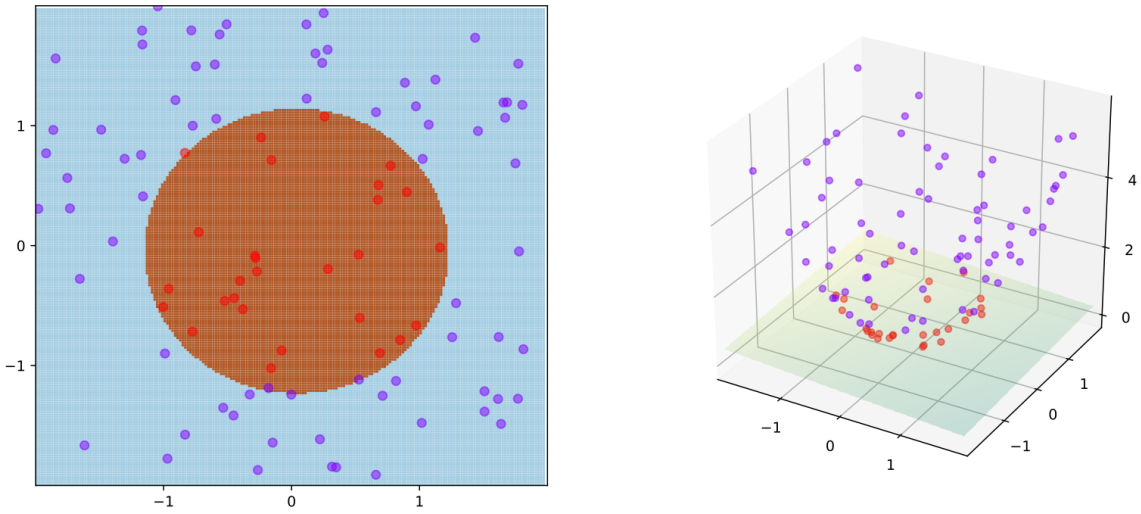


Figure 1.4. Depiction of the binary circle classification problem and its solution using the "kernel trick". On the left is the problem in the original observable space. Points inside the circle are colored red, and points outside colored purple. No hyperplane can separate these points. On the right are the points in the transformed latent space with an extra radial dimension. In this space a horizontal plane can be drawn at a height equal to the circle's radius that will separate points in the two classes. Credit: Shiyu Ji [3].

This "kernel trick" tends to work well for relatively simple problems like recognizing hand-written digits, but more advanced problems like identifying and localizing cats in images requires much more complicated features than kernel methods are capable of, even with infinite-dimensional feature spaces. For the circle classification example above we can see the utility of the kind of algebraic features used in kernel methods. It is perhaps not surprising that such transformations of a full image are not as effective for identifying cats in images. The dramatic success of deep learning [60] can largely be attributed to automated feature learning over a much larger class of complex features. Convolutional neural networks are particularly effective for computer vision tasks because of the translationally-invariant localizing features they are able to learn. The successive layers of a deep neural network learn transformations that extract abstract features like edges, textures, etc. Like kernel methods though, finding effective features makes the classification problem linear (or as close to linear as possible); the final layer of most deep

learning classifiers is a standard linear classifier that places hyperplanes in the learned feature space.

For methods like supervised deep learning, the user need not specify in any form what features the algorithm should try to learn. Given an expressive enough architecture and enough labeled training data, the network can figure it all out for itself. *Representation learning* is a subfield of ML that attempts to more directly specify or control the latent features or representations [56]. This is particularly useful in high-dimensional unsupervised learning problems where relatively few features are desired to help discover "patterns and structure" in the data. Such methods fall under the heading of *dimensionality reduction*. The canonical technique is *principle component analysis* (PCA) [61], in which the data is transformed into a linearly uncorrelated basis set by taking the leading eigenvectors of the data covariance matrix. PCA is often used in computational fluid dynamics (though typically called Proper Orthogonal Decomposition in that context); for example in *reduced order modeling* where expensive direct numerical simulations of the Navier-Stokes PDE are replaced with cheaper solutions of a finite number of ODEs that are Galerkin projections of POD modes [62].

While the features learned by convolutional neural networks are much more effective for identifying cats in pictures than the features learned by kernel methods, deep neural networks are inadequate for extracting physically-meaningful pattern and structure from physical data. Deep neural networks are surely expressive and flexible enough to express these features, but we emphasize again that ground truth for physical pattern and structure does not exist. Thus the problem for deep learning is not in expressing the features, but in how to learn the features. Our quest here is to give a rigorous and principled definition of pattern and structure in nature from physical principles, and deep learning will not be of help for this task.

Operator-theoretic methods have risen to prominence as physics-based representation learning for high-dimensional dynamical systems. Consider a discrete-time dynamical system,

$$\mathbf{x}_{t+1} = f(\mathbf{x_t}) \ . \tag{1.4}$$

The *Koopman operator* $\mathcal{K}$ [63, 64] gives the dynamics in feature space,

$$(\mathcal{K}\varphi)(\mathbf{x}) = \varphi\big(f(\mathbf{x})\big) \ . \tag{1.5}$$

Here the features (typically called *observables* in this context) are scalar functions on the state space, $\varphi : \mathbb{R}^N \to \mathbb{R}$, i.e. individual components of the feature maps described above. The Koopman operator acts on the Hilbert space of all possible observables. Analogous to the kernel trick, the boost to infinite dimensions linearizes the dynamics; $\mathcal{K}$ is a linear, infinite-dimensional operator. Any nonlinear dynamic f can be made linear through $\mathcal{K}$, but at the price of requiring infinite dimensions.

Finite-dimensional approximations of $\mathcal{K}$ have shown to be quite useful [64, 65]. The popular Dynamic Mode Decomposition [66, 67] is a simple linear method that approximates the infinite-dimensional Hilbert space with a finite-dimensional subspace spanned by linear monomials. DMD attempts to find the best linear approximation to f and is most effective in describing growing or decaying oscillatory modes. Nonlinear generalizations like Extended Dynamic Mode Decomposition [68] and Time-Lagged Independent Component Analysis [69] perform regression in a finite-dimensional subspace spanned by a given library of (potentially nonlinear) observable basis functions to approximate $\mathcal{K}$. These nonlinear methods can also be used to approximate the *Perron-Frobenius operator*, which is the dual of the Koopman operator and evolves distributions over the state space. Modes of finite-dimensional approximations to the Perron-Frobenius operator (and the related transfer operator) are used to extract slow modes (metastable states) [70] and almost-invariant sets [71, 72].

Operator-based methods are the most similar approaches to the physics-based representation learning methodology for spatially-extended dynamical systems that we develop here. While the physical principles we pursue for a general theory of pattern and structure will take us in a different direction, there are many similarities with operator methods. However, a detailed technical discussion of these similarities is outside the scope of this thesis.

To recap, we seek a theory of pattern and structure in far-from-equilibrium physical systems. Because these are emergent behaviors that can not be deduced from the govern-

ing equations we turn to data-driven methods, rather than traditional analytics, to study structured behaviors directly. That is, we seek a behavior-driven theory of pattern and structure. Machine learning methods like deep learning are capable of extracting complicated patterns and structure from data, but only if trained on ground-truth examples, which do not exist for this problem and are in fact what we want to define. Unsupervised operator-theoretic dimensionality reduction methods come closest to what we seek, and indeed are capable of capturing physically-meaningful structures in certain cases. The formal theory we develop captures a more general and abstract notion of pattern and structure, and because it is a behavior-driven theory it provides an unsupervised representation learning method to extract these features directly from spatiotemporal data.

To establish context for this theory we start with dynamics, where we find that nonlinearity is, on the one hand, required for the emergence of complex structures, but on the other an enormous challenge for traditional mathematical analysis.

1.3 In Theory

From the outset of modern dynamical systems theory Poincaré had run up against what we now call deterministic chaos and realized the predictive limitations for complex dynamical systems [73]. Not only are nonlinear systems generally not solvable, their exponential sensitivity to initial conditions can make solutions difficult or impossible to even reasonably approximate [74]. Initiated by Poincaré, the emphasis for such systems moved to qualitative analysis, e.g. classifying types of permitted orbits based on their stability and topological properties [75, 76]. Consider a dynamical system $\dot{\mathbf{x}} = F(\mathbf{x})$, $\mathbf{x} \in M$, where the phase space M is a manifold of dimension n and $F : M \to \mathbb{R}^n$ is a vector field. Rather than seek a general solution $\mathbf{x}(t)$ for any initial condition $\mathbf{x}_0 \in M$, we are interested in qualitative properties of the orbits traced by $\mathbf{x}(t)$. Does $\mathbf{x}(t)$ approach a fixed point or more general limit cycle as $t \to \infty$? For what values of $\mathbf{x}_0 \in M$ do the orbits reach this limit cycle?

When speaking of a system's behaviors we formally mean its orbits — the trajectories $\mathbf{x}(t)$ traced out by the time-evolution of an initial condition $\mathbf{x}_0$. This may seem like unnecessary terminology for low-dimensional systems, but it is rather useful in high-dimensional systems as very high or infinite dimensional spaces are difficult to conceptualize. When viewing video of a fluid flow it is much easier to track qualitative behavior of the flow, rather than track the evolution of every point in space over time. It should be noted that we are capable of tracking qualitative behavior precisely because the states and orbits of the fluid flow are, in some way, structured.

Qualitative dynamics seeks to answer questions about a system's behaviors, its allowed orbits, by analyzing specific solutions (rather than general solutions) of the equations of motion. For instance, under what conditions is a solution stable to small perturbations? This is feasible in low dimensions and for certain idealized cases in high dimensions. For more complex systems this is no longer feasible. We instead seek to use a system's behaviors to answer questions about the system [77]. Often in these systems the interest is in certain properties of the complex behaviors themselves and how the behaviors may undergo drastic spontaneous change, as with the fluid instabilities studied by Bénard,

Rayleigh, and Taylor. In particular, we are interested here in behaviors that have complex pattern and structure, how these behaviors may spontaneously arise, and how they evolve over time.

Our study of pattern and structure in a system's orbits has its foundations in the field of *symbolic dynamics* [78], which lies at the interface of dynamical systems and computation theory. The core physical principle of our physics-based machine learning approach is broken symmetry. The minimal machine presentations [79, 80, 81] of symbolic dynamics provide a mathematical formalism to define a notion of pattern as generalized symmetry. Building on this, our theory defines coherent structures in terms of locally-broken patterns (generalized symmetries). This development is outlined below. Symbolic dynamics is discussed further in Section 2.3. Machine presentations are discussed in Chapter 3, with the algebraic theory of patterns as generalized symmetries given in Section 3.1.6.

A natural simplification for dynamical systems is to discretize time. This can be done, for example, using Poincaré return maps [73, 82] which measure intersections of continuous orbits with a subspace of M that is transverse to the flow f. Qualitative behavior of orbits are persevered by Poincaré maps; (quasi-)periodic orbits of the continuous system lead to (qusi-)periodic orbits of the return map, etc. However time is discretized, we can generally consider a discrete-time dynamical system as a compact metric space M but now with a continuous function $f : M \to M$ that simply maps from the phase space to itself. Orbits are given as the iterates $\{\mathbf{x}_0, f(\mathbf{x}_0), f^2(\mathbf{x}_0), f^3(\mathbf{x}_0), ...\}$. Originating with Hadamard's study of geodesic flows on surfaces of negative curvature [83] (contemporary with Poincaré), symbolic dynamics makes the further simplification of discretizing the phase space M. Consider a measurement partition $g : M \to \mathcal{A}$ such that g uniquely maps each $\mathbf{x} \in M$ to a symbol in the finite set $\mathcal{A}$. Here the system's orbits are replaced with sequences of symbols from $\mathcal{A}$, and the potentially complicated dynamic f is replaced with a trivial shift dynamic σ that simply moves indices of each symbol in the sequence. System behaviors are now given as bi-infinite symbol sequences. The conversion of a continuous dynamical system into sequences of observed measurement symbols is referred to here as *dynamical structure modeling*, and is discussed further in Section 2.1.2, with

details of phase space discretization through measurement partitions given in Section 2.2. These symbol sequences are the objects of study of symbolic dynamics, and allow for the use of computation-theoretic models to formally capture the pattern and structure of the dynamical evolution of symbols.

The collection of all bi-infinite sequences of symbols from $\mathcal{A}$ is known as the *full-$\mathcal{A}$ shift* and is denoted as $\mathcal{A}^{\mathbb{Z}}$. A *shift space* is a compact shift-invariant subset of $\mathcal{A}^{\mathbb{Z}}$, denoted as $\mathcal{X}$. There are many ways to specify a particular shift space; the most common is through specification of a set $\mathcal{F}$ of *forbidden words*. The shift space $\mathcal{X}_{\mathcal{F}} \subseteq \mathcal{A}^{\mathbb{Z}}$ is the set of all bi-infinite symbol sequences that do not contain any finite sequences in $\mathcal{F}$. If $\mathcal{F}$ is finite then $\mathcal{X}_{\mathcal{F}}$ is a *shift of finite type*, originally coined *intrinsic Markov chains* as the support of any Markov chain is a shift of finite type. If $\mathcal{F}$ is a regular language [84] then $\mathcal{X}_{\mathcal{F}}$ is known as a *sofic shift*. The class of sofic shifts is the smallest collection of shift spaces that contains all shifts of finite type and is closed under continuous surjective maps [85].

The notion of forbidden words in system behavior starts us towards a theory of pattern and structure. If the set of forbidden words for a system has a high computational complexity then the system itself must be, in some way, highly complex. But how do we make this operational? How does specification of what *can not* happen induce structure in what *can* happen? This is where we need the notion of a *model* of system behavior, and more specifically, computation-theoretic models of algorithm and effective procedure [86]. One can think of a pattern as being a predictive regularity. Similarly, a model can be thought of as a compressed representation that predicts a system's allowed behaviors. Knowing what is not allowed enables a model to better predict what is allowed. Once we have a model of a system's behaviors we can ask how much computational resources the model needs — the number of bits the model has to store about the process — in order to make its predictions. There may be many models that optimally predict a given system's behaviors. However, there will always be some lower bound on the minimal computational resources required, and we are thus interested in minimal models. *For a model to optimally predict with minimal resources that model must capture pattern and structure present in the system's behaviors.*

Indicative of the interesting connections between complex dynamical systems and computation theory provided by symbolic dynamics, sofic shifts are those that can be defined by finite automata models. Every sofic shift $\mathcal{X}$ can be *presented* by a minimal finite automata that generates exactly the elements of $\mathcal{X}$. One important property to highlight here is that sofic shifts and their presenting automata have a defining *semi-group algebra* [85, 87]. Symmetry is perhaps the most important and useful idea in all of physics. In scientific machine learning, physics is most often incorporated through symmetry constraints, using group equivariance [88, 89, 90]. For pattern and structure, symmetry and its group algebra is too strict. The semi-group algebra of sofic shifts provides a mathematical formalism for capturing pattern and structure as *generalized symmetries* of the system.

Though symbolic dynamics originated as a tool for studying dynamical systems through measurement discretization, the field itself has largely been concerned with the mathematical properties of abstract shift spaces. In parallel to the study of shift spaces, the early work in dynamical structure modeling (also called nonlinear modeling) first introduced the quantitative information measures of Shannon's theory of communication [91] to dynamical systems using discretizing measurement partitions [92, 93, 94, 95, 96, 97, 98]. A more in-depth historical review can be found in Ref. [99]. Building on these foundations, a new generation of physicists grappling with the implications of deterministic chaos, most notably Wolfram [100, 101], Grassberger [102], and Crutchfield [103], first employed the machine presentations of shift spaces to analyze the *complexity* of dynamical systems. Crutchfield and his collaborators used information theory to synthesize symbolic dynamics and its computation-theoretic models with statistical mechanics to develop an operational framework of pattern and structure for complex natural systems. This body of theory is called *computational mechanics* [104].

1.3.1 Computational Mechanics

To capture pattern and structure in ensembles of behaviors computational mechanics employs a minimal, but now stochastic, model that optimally predicts the distribution over behaviors. It does so by relying on a weak notion of causality; the assumption that

the system is a channel [105] that communicates its past to its future through the present. This leads to the *causal equivalence relation*, which will be described in more detail below and in Chapter 3;

$$\text{past}_i \sim_\epsilon \text{past}_j \iff \Pr(\text{Future}|\text{past}_i) = \Pr(\text{Future}|\text{past}_j) \ . \tag{1.6}$$

The equivalence classes over pasts induced by the causal equivalence relation are known as the *causal states* of the system; they are the unique minimal sufficient statistic of the past for optimally predicting the future, and are analogous to the states of a presenting automaton for a sofic shift. The idea of internal states of machines being equivalent histories goes back to the origins of the physics of computation, as outlined in Section 2.1.1.

Notice the causal states and the causal equivalence relation they are built from do not reference any governing equations of motion. Computational mechanics is a behavior-driven theory; pattern and structure of a system's behaviors are discovered directly from the behaviors themselves. This is in contrast to "symbolic regression", in which one attempts to discover the equations of motion by fitting the observed data to some pre-specified function basis [106, 107, 108, 109]. The machine presentations discovered by computational mechanics provide a more general and abstract representation of the system's dynamics from which pattern and structure may be extracted. As we discuss more in Section 1.6, an alternative modeling paradigm like machine presentations is required for *emergent* behaviors that can not be deduced from the equations of motion. Hence we emphasize that our use of behavior-driven models is *not* because we just don't know the governing equations for the systems we are interested in; rather it is that the equations can not help us understand the complex emergent behaviors we are interested in (and so discovering the governing equations is similarly not helpful for this work).

Like the use of Koopman modes for dimensionality reduction, the latent causal state representations are accessed through an infinite-dimensional feature space. For the Koopman operator, this is the space of all observables; for causal states it is the joint space of infinite-length pasts and futures. Similarly, as all dynamics are linearized in the observable feature space of the Koopman operator, so are they linearized in the feature space of pasts and futures via the shift operator. The linear shift dynamic in the joint feature

space of pasts and futures is the *data-generating process* that gives the *data-generating distribution* Pr(Future|past) which defines the causal states.

Most of the development of computational mechanics has been for strictly temporal systems. Shalizi, along with Hanson and Crutchfield, developed the foundations for computational mechanics in spatiotemporal systems. The key step in moving to spacetime is to use *lightcones* as local notions of past and futures, so that two past lightcones ℓ_i^- and ℓ_j^- are causally equivalent if they have the same conditional distribution over future lightcones;

$$\ell_i^- \sim_\epsilon \ell_j^- \iff \Pr(\mathrm{L}^+|\ell_i^-) = \Pr(\mathrm{L}^+|\ell_j^-) \ . \tag{1.7}$$

The equivalence classes induced by (1.7) are called *local causal states* [110]. These are the main objects of interest for this thesis. We will use the local causal states to create an unsupervised physics-based representation learning method that extracts (potentially hidden) pattern and structure in spatiotemporal systems.

The ϵ-function, which generates the causal equivalence classes, is the feature-map from past lightcones to local causal states; $\epsilon : \ell^- \mapsto \xi$. Segmentation of a spatiotemporal system is achieved by mapping a spacetime field $\mathbf{x}$ to its associated local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$: every local point in spacetime $\mathbf{x}(\mathbf{r}, t)$ is mapped to its latent local causal state via its past lightcone $\xi = S(\mathbf{r}, t) = \epsilon\big(\ell^-(\mathbf{r}, t)\big)$. Crucially, this ensures the global latent variable field $\mathcal{S}$ maintains the same coordinate geometry of $\mathbf{x}$ so that localized structure in $\mathbf{x}$ can be identified via properties of $\mathcal{S}$ in the same region.

We discussed in Section 1.2 how deep neural networks are capable of learning complicated and powerful representations of image data when trained on ground-truth labels. But deep neural networks can also be used for unsupervised representation learning using the *autoencoder* framework. Autoencoders are feedforward neural networks that attempt to learn the identity function through a bottleneck latent space: an input X is encoded into a latent space $Z = \varphi(X)$, where $\dim(Z) << \dim(X)$, and then a decoding is learned from the latent space back to a reconstruction of the input $\overline{X} = \varphi^{-1}(Z)$. The intuition behind how autoencoders learn structural representations of their input is quite similar to that of the local causal states; to optimally decode the input from a minimal encoding,

that encoding must capture pattern and structure in the input. Autoencoders however have not been shown to be capable of capturing the kind of physical pattern and structure we are interested in here from snapshot images of spatiotemporal systems [111]. Given the physical insights of computational mechanics, this is not surprising. Autoencoders used in this way do not take dynamics and time evolution into account.

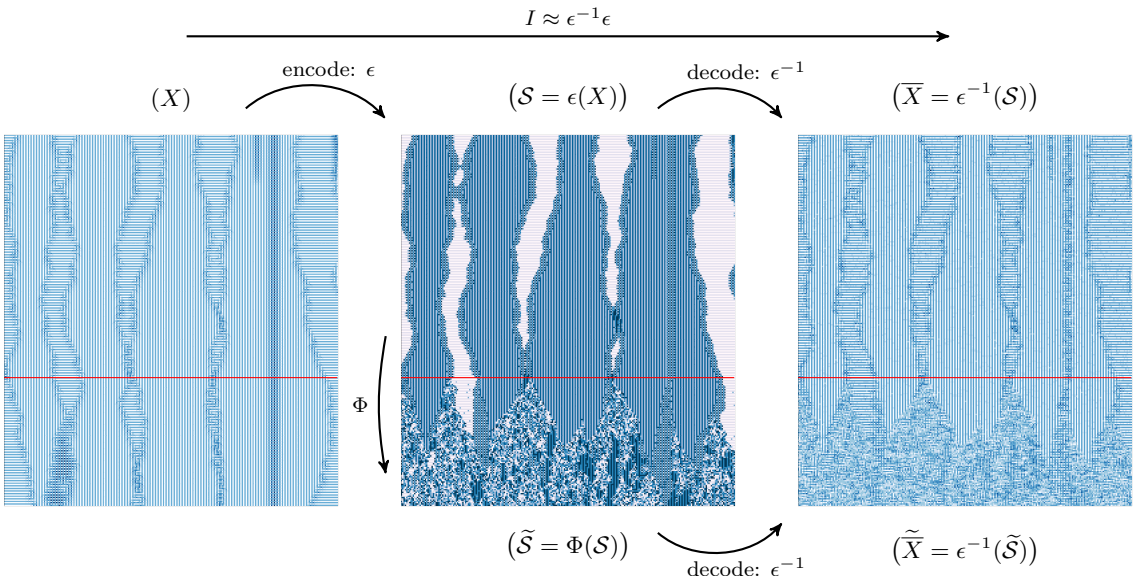


Figure 1.5. Local causal states as predictive spacetime autoencoders. An observable spacetime field X, up to time t (shown as a red horizontal line), is mapped to the local causal state field $S = \epsilon(X)$. Using, ϵ^{-1}, a reconstructed spacetime field can be created $\overline{X} = \epsilon^{-1}(S)$. With the inferred stochastic dynamic over local causal states, Φ, the states can be evolved forward in time to produce a forecasted local causal state field $\widetilde{S} = \Phi(S)$. The forecasted state field is then mapped to a forecasted observable field $\widetilde{\overline{X}} = \epsilon^{-1}(\widetilde{S})$.

With this said, we can formally connect local causal states with autoencoders. A stochastic inverse of the ϵ-function, ϵ^{-1}, can be defined that reconstructs an observable spacetime field from a local causal state field. From this, we can view the local causal states as a spacetime autoencoder; as shown in Figure 1.5 the ϵ-function encodes an observable spacetime field $\mathbf{x}$ to the compressed latent field $\mathcal{S}$ and ϵ^{-1} decodes to a reconstructed observable field $\overline{\mathbf{x}} = \epsilon^{-1}(\mathcal{S})$.

There are several points to note. First, unlike typical neural network autoencoders, the local causal states are nonparametric models and so rather than using the encoding and decoding together to train parameters like neural network autoencoders, the ϵ-map

and its inverse are learned directly by approximating the local causal equivalence relation from data. Second, as already mentioned, a crucial distinction is that the ϵ-map encoding is done locally so that the latent space and observable space share a spacetime coordinate geometry. The latent space of a neural network autoencoder does not share geometry with its inputs because of how the bottleneck is created. For local causal states approximated from real-valued spacetime data, the bottleneck comes from having a finite number of latent local causal states. Lastly, as mentioned, taking temporal evolution into account is critical. Because of this, viewing local causal states as spacetime autoencoders in this way is not particularly useful. However, a stochastic dynamic can be defined over the local causal states using Markov shielding (see Section 3.2.2.3 for Markov shielding). This, combined with the ϵ^{-1} decoding, allows for spacetime forecasting; infer the local causal states and their dynamics up to the present time, evolve the states forward in time, then decode to a predicted observable field. This view of the local causal states as *predictive spacetime autoencoders* is useful, as it provides an objective metric for optimal (temporal) prediction through a minimal latent space that can help deal with, and correct for, the approximations that are necessary for estimating causal equivalence from data. Having this performance metric also opens the path for self-supervised training of parametric neural network models (though we do not do so in this thesis).

1.4 In Cellular Automata

In the strictly temporal case, a minimal model that optimally predicts the future from the past captures the structure of the system's temporal evolution. It is less clear, in the spatiotemporal case, whether local models that optimally predict future lightcones from past lightcones at each point in spacetime can capture collective organized structures over extended regions in space and how they evolve in time to produce spacetime structures.

To investigate the capacity of the local causal states to capture extended spacetime pattern and structure we start with the simplest mathematical models of pattern formation. Cellular automata (CA) are fully-discrete spatially-extended dynamical systems. Space is discretized into a regular lattice, and each site on the lattice takes values from a discrete alphabet $x \in \mathcal{A}$. The spatial lattice evolves in discrete time steps according to a local update rule; each site $\mathbf{x}_t^{\mathbf{r}}$ (subscripts denote time, superscripts sites) is updated according to a local update rule ϕ that is a deterministic function of the radius R neighborhood of $\mathbf{x}_t^{\mathbf{r}}$. Because each site takes discrete values, there is a finite number of possible neighborhood values. Thus ϕ is given as a *lookup table* that specifies the output for each possible neighborhood. The full spatial lattice is updated by synchronous application of ϕ at each site on the lattice. Spacetime fields $\mathbf{x}$ of a CA are the orbits of the evolving spatial lattice. CAs are described in more detail in Section 4.1.

This very simple form for the equations of motion, which can be carried out by hand, is capable of producing arbitrarily complex behaviors; many cellular automata support universal computation and thus in principle can be used to reproduce any scientific simulation. Even in the simple class of elementary cellular automata (ECA), one spatial dimension with $R = 1$ and $\mathcal{A} = \{0, 1\}$, at least one ECA can support universal computation and many others produce organized pattern and structure that spontaneously emerge from random initial conditions. As such, cellular automata, and ECAs in particular, have become flagship models of spontaneous self-organization. Cellular automata are also very important models of complex systems because of the dual role they play as self-organizing dynamical systems and models of distributed computation. As we will see in more detail in Chapter 2, there are many interesting connections between models of computation and

complex dynamical systems, and cellular automata comfortably play both roles.

An early indication that local causal states are indeed capable of capturing collective structure and organization in spacetime was Shalizi et al's use of the *local statistical complexity* — the point-wise entropy over local causal states — to outline coherent structures in cellular automata [112]. The local statistical complexity is used as a *diagnostic scalar field* [113] from which coherent structures are visually identified. However, we have found that local statistical complexity can produce both false negative and false positive identification of known coherent structures in CAs. Misidentifications occur due to a mechanism we have called local causal state *contamination*.

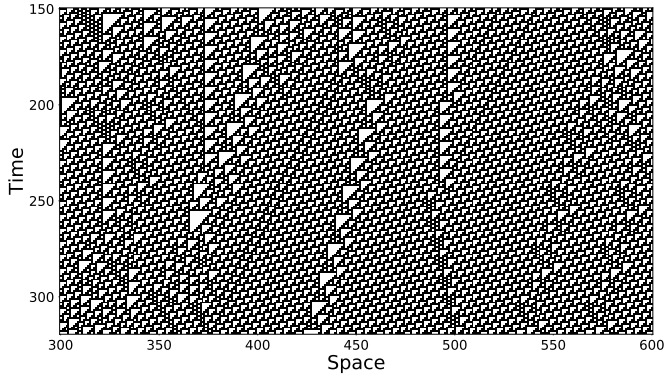
Here we seek not just an objective method of identifying coherent structures, but a formal and principled accounting of organized pattern and structure using the local causal states. To do so, we want to use algebraic and geometric properties of the local causal state fields $\mathcal{S} = \epsilon(\mathbf{x})$, rather than rely on the local statistical complexity. The subjective nature of coherent structure detection using local statistical complexity allows for some wiggle room in local causal state reconstruction. For our purposes, we need a more exact reconstruction technique with convergence guarantees. We created the *topological reconstruction* technique that enables exact reconstruction of finite approximations to the topological variants of the local causal states, with convergence guarantees using the number of unique (ℓ^-, ℓ^+) pairs seen during inference [114]. See Section 4.2 for more on topological reconstruction.

Using topological reconstruction, we show in Chapters 4 and 5 that invariant sets in the observable CA field $\mathbf{x}$ correspond to spacetime symmetries in the corresponding latent local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ [115, 114]. These spacetime invariant sets and associated regions in general CA spacetime fields are known as *domains* [5], in analogy with equilibrium statistical mechanics. Domains are formally defined in Section 4.4. They generalize the notion of spacetime pattern associated with regular symmetry. *Explicit symmetry* domains, like the hexagonal cells of RB convection, exhibit spacetime symmetry in the observable field, and hence also in the local causal state field. There are invariant sets of CAs that have spacetime symmetry in the local causal state field, but *not* in the

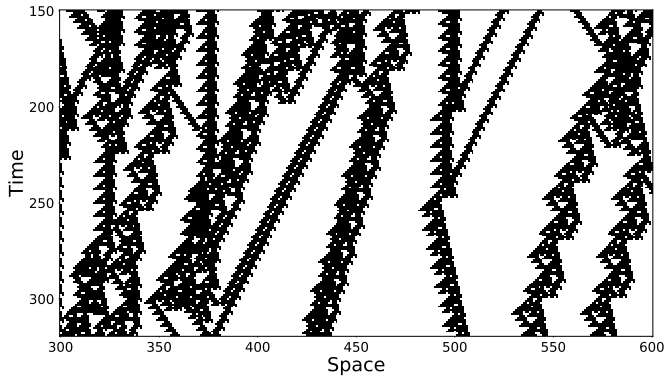
observable spacetime field. As we will see, although the observable fields of these *hidden symmetry* domains do not posses explicit symmetry, they do have a form of *stochastic symmetry*. Interestingly, domains of cellular automata appear to be related to *additive* dynamics of the CA, or more generally *permutive* dynamics [114], as detailed in Chapter 5.

Using domains as a formalization of generalized patterns in spacetime, in Chapter 6 we give a rigorous and principled definition of coherent structures in cellular automata as spatially localized, temporally persistent non-domain regions in spacetime [115]. This gives a formal definition to the notion of coherent structures as *locally broken symmetries*. Complications due to contamination arise in practice when trying to objectively identify coherent structures in CAs using this definition. Fortunately, topological reconstruction allows for a method of correcting for contamination that identifies CA structures that agrees well with an established method (known as *domain particle interaction decomposition* [5, 116, 4], described in Section 4.4.1) that filters out spatial regions incompatible with the invariant set of spatial configurations [115]. A sample coherent structure segmentation of ECA rule 110 using the local causal states is shown in Figure 1.6. The background domain states have been identified from their spacetime symmetry and all of these domain states in the local causal state field are mapped to the color `white` in the filter. All other non-domain states, which highlight the coherent structures and their interactions, are colored `black` in the filter. We emphasize that this kind of visual filtering is only possible due to the shared coordinate geometry of the observable spacetime field and the local causal state latent field. Other examples are given in more detail in Chapter 6.

Taken together, these results on cellular automata show that the local causal states, designed for optimal local prediction over lightcones, can, in a principled way, capture collective pattern and structure in spacetime. However, cellular automata are idealized mathematical models. Can these local causal state techniques be extended to the pattern and structure found in natural systems far from equilibrium? As a behavior-driven model, the local causal states require only spacetime fields, whether they are from cellular automata or general circulation climate models. The main practical difference between



(a) Raw spacetime field.



(b) Local causal state domain-nondomain filter.

Figure 1.6. ECA 110 structural segmentation: (a) A sample observable field evolved from a random initial configuration. (b) A local causal state coherent structure segmentation filter with domain sites in `white` and non-domain in `black`.

CAs and climate or fluid simulations is that CAs are discrete-valued fields, which greatly simplifies reconstruction and local causal state inference. For real-valued reconstruction more approximations are required and we no longer have convergence guarantees. This makes model evaluation much more difficult, especially because there is no ground-truth for the complex structures we are interested in. That being said, the local causal states still hold great promise for analyzing far-from-equilibrium pattern and structure.

1.5 In Complex Fluid Flows

Coherent structures [62, 117] form a "hidden skeleton" of complex fluid flows that heavily dictate material transport [118]. They are responsible for "Lévy flights" which lead to a probability distribution over tracer particles with a divergent second moment that can not be accounted for by normal diffusive processes [119]. Understanding these structures and being able to better discover them in real-world data will improve forecasting of damaging contaminants in globally significant events; volcanic ash from the Eyjafjallajökull eruption, oil from Deepwater Horizon, and radioactive contamination following the Fukushima reactor disaster are recent examples. Similarly, extreme weather events such as atmospheric rivers, cyclones, and blocking events play a crucial role in thermal and material transport in the global climate system. Detecting, classifying, and characterizing weather and climate patterns is a fundamental requirement to improve our understanding of extreme weather and climate events, their formation, and how they may change with global warming.

More generally, coherent structures in far-from-equilibrium systems can be understood as key organizing features that heavily dictate the dynamics of the full system. From a machine learning perspective, they provide a "natural" dimensionality reduction from the full high-dimensional system to relatively few collective features. They are crucial to understanding and predicting the full high-dimensional system. And, as with extreme weather events, the coherent structures are often the features of interest.

While the importance of coherent structures in complex fluid flows is widely understood, there is no accepted definition for what a coherent structure is, exactly, and no principled method for identifying coherent structures in specific flows. Again, there is no ground-truth. Recently though, the Lagrangian approach has gained popularity [71, 118, 117]. In this, coherent structures are seen as material surfaces with characteristic deformations induced by the Lagrangian particle flow. Each point of the surface at some initial time is viewed as a tracer particle, and the Lagrangian evolution of all tracer particles then acts as a continuous deformation of the surface through time. However, even in this Lagrangian framework there are many different approaches for specific

implementation of these ideas and there is no consensus on which is "correct" [113].

We make no claim that the local causal states will be the "correct" approach to coherent structures in complex fluid flows. However, they do provide a novel and interesting perspective on this difficult challenge. The algebraic interpretation of patterns as generalized symmetries, which are broken locally by coherent structures, provides an appealing physical basis for the local causal state approach.

While we are interested in using the local causal states to analyze the same phenomenological structures as the Lagrangian approaches, the local causal states are inferred from spacetime fields and thus are an Eulerian method. A frequent objection to (instantaneous) Eulerian approaches for coherent structures is that they will depend on the frame of reference. This is not the case, however, for the local causal states since they are local models which are constructed from lightcones. Because lightcones are defined purely in terms of distances, they are invariant under Euclidean isometries and thus independent of frame-of-reference. In addition to their much greater generality (i.e. they do not require any notion of Lagrangian flow), the local causal states are a promising approach due to the formal coherence principle of locally broken symmetry that they provide for defining coherent structures; although the clean picture of coherent structures on top of background domains found with cellular automata can get muddied when working with complex fluid flows. This is due both to the increased complexity of the flow behavior itself as well as the additional approximations required for local causal state reconstruction from real-valued spacetime fields.

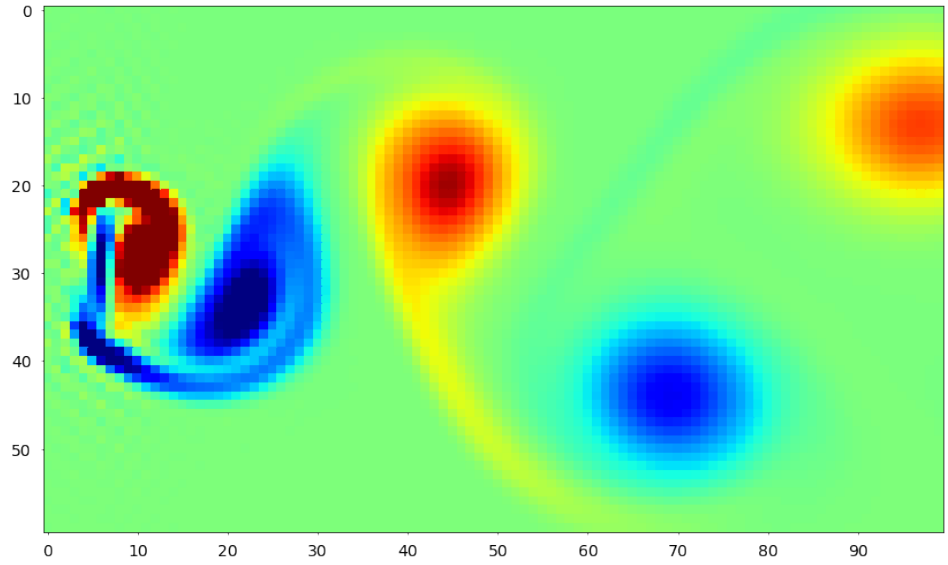
In Figure 1.7 we see a demonstration of the local causal states' ability to extract coherent structures in a complex fluid flow, the von Kármán vortex street [120]. This particular instance is vortex shedding from flow around a linear barrier, simulated using the Lattice Boltzmann algorithm [121]. We emphasize again the behavior-driven aspect of the local causal states. The behavior of a vortex street can be simulated from two rather different generative models, Lattice Boltzmann or Navier-Stokes. The local causal states are indifferent to the generative model; it is the only the generated behavior that matters. Figure 1.7 (a) shows a snapshot spatial image from the observable vorticity

field of the flow and (b) shows the corresponding local causal state field snapshot. The background flow is mapped to a single local causal state, colored `white`, signifying a Euclidean symmetry domain. All other states map to different colors and represent the coherent vortices and their internal structure. Considering states as only `white` (domain) and `not white` (non-domain) produces a coherent structure segmentation like that of ECA rule 110 in Figure 1.6.

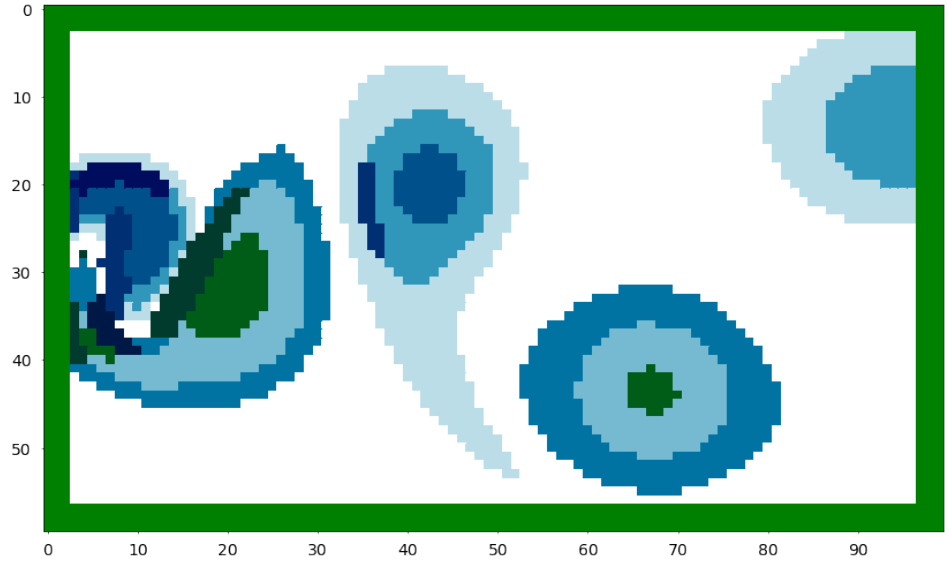
While the vortices in the above example are pretty evident by eye in the observable field, there is still no objective ground truth for what defines these objects. This is true for coherent structures in fluid flows generally. Without any ground truth to compare against, Ref. [113] compared several leading Lagrangian approaches on three benchmark data sets. In Section 7.3 we use the two more complex of these data sets to compare local causal state coherent structures with Lagrangian coherent structures [122]: a pseudo-spectral direct numerical simulation of two-dimensional turbulence in a doubly-periodic domain and interpolated video data of the clouds of Jupiter from the Nasa Cassini spacecraft. In Section 7.3.2 we find that the coherent vortices in the two-dimensional turbulence data can be identified as coherent structures from the local causal state definition given for cellular automata. With a particular choice of inference parameters, the background potential flow gets mapped to a single local causal state, signifying a Euclidean symmetry domain. The coherent vortices are mapped to a set of localized non-domain states that remain coherent over time.

Coherent vortices in Jupiter's atmosphere, such as the Great Red Spot and the String of Pearls, are identified in Section 7.3.3 with similar sets of coherent states. However, for Jupiter these states are not necessarily in contrast to some clear background domain. The question of what are the domain(s) of Jupiter's atmosphere highlights the contrast between the clean mathematical universe of cellular automata and the complications of far-from-equilibrium systems found in nature.

One way to think about the idea of domains in the clouds of Jupiter is to go back to the picture of sequential bifurcations. The equilibrium and base nonequilibrium states of Jupiter's atmosphere would have radial velocity everywhere zero. Here the entire system



(a) Vorticity observable field



(b) Local causal state latent field

Figure 1.7. Kármán vortex street structural segmentation: (a) Spatial snapshot of the vorticity observable field. (b) The corresponding spatial snapshot of the local causal state field. Each unique color represents a unique local causal state.

is viewed as a domain pattern with Euclidean symmetry. As with Taylor-Couette flow, we would imagine the patterned state that emerges from the primary bifurcation is one with horizontal stripes from the onset of convection rolls. At its current state, perhaps the most distinguishing features of Jupiter are in fact its zonal belts. Imagining that Jupiter actually did form from such a series of bifurcations, remnants of the symmetries of this

first patterned state are still present in the zonal belts. Thus we can view the zonal belts as the domain symmetries, though the actual symmetries have been long broken and the dynamics inside each belt is highly, and uniquely, structured and turbulent.

This leaves us with a question for local causal state coherent structure analysis. What features do we want to capture as locally-broken generalized symmetries? The Great Red Spot is an obvious structure of interest [12, 11]. If we want to capture it as a coherent structure using local causal states as done with cellular automata and two-dimensional turbulence we would hope to find each zonal belt mapped to its own state, signifying the domain symmetry, then a set of localized non-domain states associated with the Great Red Spot. However, as we've already said the bands are full of other turbulent structures besides the Great Red Spot. The above analysis would nicely capture the Great Red Spot at the expense of these other structures. For natural far-from-equilibrium systems it is not a clear case of capturing *the* patterns and structures present, but rather a choice must be made of capturing pattern and structure at a particular level of detail, or that persist for a particular length of time. As we will see in Section 7.1, the approximations required for local causal state reconstruction in real-valued fields are a strength, rather than a weakness. Inference parameters can tune the reconstructed states to capture varying levels of structural detail and coherence time in a satisfying way.

Having established the utility of local causal states for natural pattern and structure, there is one far-from-equilibrium system that is of particular interest; and challenge.

1.5.1 Extreme Weather and Climate Change

Extreme weather is one of the main mechanisms through which climate change will directly impact human society. Life across the globe has survived and thrived by adapting to its local weather, including extreme events such as strong winds and floods from cyclones, drought and heat waves from blocking events and large-scale atmospheric oscillations, and critically-needed precipitation from atmospheric rivers. Driven by an ever-warming climate, extreme weather events are changing in frequency and intensity at an unprecedented pace [123, 124]. We need to understand these events and their driving mechanisms to enable communities to continue to adapt and thrive.

High-resolution, high-fidelity global climate models are an indispensable tool for investigating climate change. A multitude of climate change scenarios are now being simulated, each producing hundreds of terabytes of data. Currently, climate change is assessed in these simulations using summary statistics such as mean global sea surface temperature. This is inadequate for answering detailed questions about the effects of climate change on extreme weather events. Due to the sheer size and complexity of these simulated data sets, it is essential to develop robust and automated methods that can provide the deeper insights we seek.

Recently, supervised Deep Learning (DL) techniques have been applied to address this problem [125, 126, 127, 128]. Further progress however has been stymied by two daunting challenges: reliance on labeled training data and interpretability of trained models. The DL models used in the above studies are trained using the automated heuristics of TECA [129] for proximate labels. This is necessary because, simply put, there currently is no ground truth for pixel-level identification of extreme weather events [130]. While the results in Ref. [125] show that DL can improve upon TECA, the results of Ref. [128] reach accuracy rates over 97% and thus essentially just reproduce the output of TECA. The supervised learning paradigm of optimizing objective metrics (e.g. training and generalization error) breaks down here [49]; TECA is not ground truth and we do not know how to train a DL model to disagree with TECA in just the right way to get closer to "ground truth".

To avoid this issue, a campaign is currently underway to generate expert-labeled training data [131]. Supervised DL models trained on this data will automate expert-level curation of large climate data sets for extreme weather detection. In this case there too will be challenges. Although an improvement over automated heuristics, expert-labeled data is still not an objective ground truth. Further, while human experts can debate the subtleties of physical characteristics of extreme weather events, the interpretability problem [132] prevents us from probing a trained DL model to determine exactly how and why it identifies (or misidentifies) specific events.

The local causal states present a promising unsupervised alternative to circumvent

these challenges of DL-based approaches. While not yet able to cleanly extract extreme weather events, they show promising results in outlining hurricanes and atmospheric rivers in the water vapor field of the CAM5.1 general circulation climate model [122].

1.6 Complexity, Emergence, and Computation

Quantitative science has relied and flourished on the reductionist hypothesis — that all systems, no matter how complex, are governed by the same fundamental laws. With this, however, has often come an erroneous assumption of what Philip Anderson called "*constructionism*" — the ability to start from the fundamental laws and reconstruct the universe [133]. For a sufficiently *complex* system, strong nonlinear dependence among system components can make it impossible to deduce *emergent* behaviors from the governing equations [134, 135]. Having sophisticated climate models does not mean we fully understand the physical processes that govern the dynamics of hurricanes produced in these simulations, and more notoriously, turbulence remains a persistent mystery despite knowing Navier-Stokes for almost 200 years.

Cellular automata provide perhaps the most striking example of emergent complexity arising from simple dynamical rules. As we have seen, the dynamics of CAs are given by local look up tables and can be iteratively computed by hand. Yet, as with the elementary CA rule 110, certain CAs can support universal computation [136] and can thus produce arbitrarily complex behaviors (recall that rule 110 is just a particular assignment of a radius-1 local dynamic over binary strings). Universal computation in rule 110 is achieved through the dynamics and interactions of emergent coherent structures. This is known only phenomenologically; it is currently out of reach to derive, starting from the lookup table (the equations of motion), that rule 110 produces emergent coherent structures with interactions that can be used to emulate a cyclic Tag machine (which is Turing complete). Such a derivation may even be impossible.

Complexity and emergence are notoriously tricky concepts and now typically avoided in the physical sciences and applied mathematics (though that is starting to change [137, 135, 138, 139]). In attempting to study pattern and structure in natural systems however, some general notions of complexity and emergence are inescapable. While it is not our goal here to attempt to at last pin down definitions, the example of rule 110 gives us the opportunity to make some formal statements using the theory of computation. In this, we will take seriously the physical Church-Turing thesis, described in more detail

in Section 2.1.1, and consider the implications of computational intractability and undecidability on physical systems. Rather than using computational considerations to place bounds on possible physical theories [140], we will examine how computational considerations limit our ability to *understand* physical phenomena.

One proposed definition of a complex system is one for which nontrivial properties of its limit set(s) are uncomputable [141]. This adds another level of unpredictability beyond deterministic chaos; it is not just uncertainty in what the system will do, it is uncertainty in what the system is even capable of. Another common, but less formal, definition of a complex system is one with arbitrarily long transient behavior. If the system states that comprise the limit set are uncomputable, it is impossible to distinguish transient states from recurrent states, and so the less formal definition is subsumed by the formal. This definition would classify rule 110 as complex, due to the Halting Problem [142]. Being Turing complete sets rule 110 apart from the other elementary cellular automata, but qualitatively rule 110 does not stand out as the only "complex" ECA. In fact, it can be shown that nontrivial limit sets for cellular automata are generally uncomputable [143]. From this definition, cellular automata, as a class of models, are complex dynamical systems.

The computational mechanics framework, as will be discussed in more detail in Section 3.1, has a measure of complexity based on how information from the past is processed to best inform what will happen in the future. This is known as the *statistical complexity*, and is computed using an invariant asymptotic distribution. As such, it seems at odds with the definition of a complex system just given; the statistical complexity is a nontrivial property of a system's limit set and thus would be uncomputable for a complex system. It is best to think of statistical complexity as a measure of the complexity of a *behavior*, rather than of a *system*. It may be taken as a given that the systems studied here – cellular automata, turbulent fluid flows, and climate – are complex systems, in the sense of having uncomputable limit sets. It would be useful, however, to quantify the level of complexity in the patterns and structures produced by these systems. This is provided by the statistical complexity. It is perhaps natural to consider the hexagonal cells in RB

convection as a more complex pattern than striped convection cells because the hexagons break symmetry in two directions while stripes break symmetry in just one direction. The quantification of pattern complexity by statistical complexity can be thought of as a generalization of this "degree of symmetry breaking" argument from RB convection.

Emergence is a more difficult subject, and no formal definition will be given here. Intuitively, we want to say a behavior is emergent if it can not be derived or deduced from the governing model [144, 134, 145, 135]. However, just because we do not yet know how to derive a particular behavior does not mean it is necessarily impossible. Thus this view of emergence is seen more as a statement of human limitations, rather than as a property of the system of interest. For the purposes of this thesis, the view of emergence as a statement of human limitation is actually what we are interested in. As described above in Section 1.1.2, it is often regarded in physics that understanding the fundamental constituents of a system and the interactions among them is what leads to an understanding of the system's behaviors; "the rest is stamp collecting", as purportedly said by Rutherford. What is not always appreciated is that the question of *how* constituents and interactions give rise to particular behaviors can also be of fundamental importance [133]. Perhaps less well appreciated is just how difficult this question can be to answer. In fact, as we now highlight, there are cases where it is provably impossible to derive emergent behaviors from the governing equations of motion.

We return to cellular automata, where we can again use computation theory to formalize the difficulty in understanding how equations of motion give rise to specific behaviors. Recall that the limiting behavior of CAs are uncomputable [143], starting from the lookup tables, which is already a formal limitation on our ability to understand how the dynamical rules specified by CA lookup tables produce CA behaviors. The situation can be more severe for certain CAs, for which finite-time behavior can also present difficulties. In particular, it has been shown that for several CAs [146], including rule 110 [147], the problem of predicting the state of the CA t time steps into the future is P-Complete.

To elaborate, a P-Complete problem is just as hard to solve as any problem that can be solved in polynomial time with a serial computer. The class of these problems

that can be solved in time that is polynomial in the size of the input is the computational complexity class P [148]. A subclass of P is the class NC of problems that can be solved by parallel computers in polylogarithmic time. If the time prediction problem of a particular CA has a closed-form analytic solution, that prediction problem would be computable in polylogarithmic time by a parallel computer and thus be in NC. It is widely believed, but not proven, that there are "inherently sequential" problems that are in P but *not* in NC. This is analogous to the more famous P vs. NP problem, for which it is also widely believed, but not proven, that there are problems in NP that are not in P [149]. If NC $\neq$ P, this implies that for a CA with P-Complete time prediction, the most efficient way to find the future state of the CA is by direct simulation. There can be no closed-form analytic solution.

As discussed in Ref. [146], the P-Completeness of the Majority-Vote CA has implications for the predictability of the Ising model. In addition, it has been shown that, in the infinite-size limit, the macroscopic thermodynamic behavior of the Ising model is uncomputable from the microscopic interactions [135]. These results for the Ising model show that physically-relevant models produce behaviors that are formally difficult to understand from the governing equations. While CAs can be seen as discrete analogues of partial differential equations (PDEs), it is not clear that the difficulty posed by nonlinear PDEs can be similarly formalized.

Yet, the difficulty of understanding the emergence of complex behaviors from nonlinear PDEs is crucial in the pursuit of far-from-equilibrium pattern and structure. This brings us to the central motivation of the work presented in this thesis: Organized pattern and structure in the far-from-equilibrium regime are emergent behaviors. Whether this is a matter of our current inability to derive these behaviors from governing laws, or whether it will ultimately be a proven difficulty as with the examples above, the immediate implication is the same: the traditional hypothesis-driven paradigm breaks down for emergent behaviors, and an alternative, perhaps data-driven, mode of scientific inquiry is required.

Key to the hypothesis-driven approach is the ability to deduce physical consequences of

a model (traditionally parametric equations of motion[2]) that provide testable predictions. If the model prediction is not falsified by experiment it provides a mechanistic hypothesis for the underlying physics that produces the observed phenomenon. For complex systems, the governing equations alone are insufficient for understanding the mechanisms underlying emergent behavior if the behavior can not be deduced from the equations [150]. Simulating particular solutions is similarly insufficient because it may be impossible to isolate individual mechanistic components in the tangled web of interactions that give rise to emergent behavior. This also makes it difficult, if not impossible, to test mechanistic hypotheses of emergent behavior in controlled experiments.

Below we give a short example of how the hypothesis-driven approach has successfully explained the physical mechanism underlying the near-equilibrium patterns seen in Bénard convection. We argue that analogous approaches break down far from equilibrium.

1.6.1 A Quick Example: Mechanisms of Instability in Bénard Convection

Above in Section 1.1.1 we discussed the convective instabilities first studied experimentally by Bénard in 1900, then theoretically by Rayleigh in 1916; and many others after. Recall that when a layer of fluid is heated from below heat is transferred through the fluid via conduction if the Rayleigh number (the strength of the temperature gradient, relative to the internal dissipation of the fluid) is low enough. At a critical value R_c of the Rayleigh number the conduction state becomes unstable and convection becomes thermodynamically more favorable than conduction. This is the onset of patterns, such as the Bénard cells shown in Figure 1.3.

As mentioned in Section 1.1.1, the thermodynamic mechanism that determines the critical value of temperature gradient needed for the conduction state to go unstable depends on whether the container holding the fluid is sealed on top or not. In fact, the analysis carried out by Rayleigh assumed a sealed container, while the experiments of Bénard used an open container; the critical value computed by Rayleigh did not agree with the value obtained experimentally by Bénard. For closed-surface convection (Rayleigh) the

[2]We use the term "parametric equations of motion" to indicate there are free parameters (e.g. thermal diffusivity, the coefficient of thermal expansion, and kinematic viscosity for the case of convection discussed below) that can be tuned to describe a particular physical instantiation or experimental setup.

instability is buoyancy driven, whereas for free-surface convection (Bénard) the instability is driven by surface tension. But how do we *know* these two physical mechanisms govern convective instabilities in the two different cases of open vs closed containers?

This well-studied problem of convective instability provides a quintessential success story of the hypothesis-driven scientific paradigm. First, a physical phenomenon was carefully observed and measured in experiment by Bénard. Then, a mechanistic hypothesis of buoyancy-driven instability was proposed by Rayleigh, who was able to quantitively deduce the experimentally-observed behavior (details given below). The behavior deduced from the mechanistic hypothesis could then be compared with the experimental findings. In this case, the initial hypothesis was not a good match with experiment. From here additional experiments are required to at least verify there was no critical error in the original experiment (even better if more precise measurements are made). If the original experiment is verified then an additional mechanistic hypothesis is required. In this case, additional experimental results continued to disagree with Rayleigh's theoretical analysis, and so a new mechanistic hypothesis that free-surface convective instability is driven by surface tension, rather than buoyancy, was proposed by M.J. Block [151]. This new hypothesis was quantified by J.R.A Pearson [152], who performed a linear stability analysis similar to that of Rayleigh, but this time on a different set of parametric equations corresponding to the surface tension hypothesis. After subsequent studies and more precise modern experiments, the critical value corresponding to the surface tension hypothesis is found to agree very well with free-surface convection experiments [153].

Let's clarify the distinction between these two mechanistic hypotheses: buoyancy driven vs. surface tension driven instability. Of course in both cases the basic laws of motion and thermodynamics apply. The parametric equations of motion that quantify these hypotheses are built from the Navier-Stokes equation, the heat equation, and the continuity equation. The key difference comes in the form of constitutive relations among thermodynamic variables, and how they are incorporated into the equations and boundary conditions. In particular, the dependence of density and surface tension on temperature. For Rayleigh's calculation, he assumed that density varies linearly with

temperature and that density variation is only significant in the buoyancy force – this is known as the Oberbeck-Boussinesq approximation. Rayleigh did not include surface tension effects. Pearson did the opposite; he did not include density effects, and instead included surface tension and its dependence on temperature.

We will not perform the linear stability analysis here, which is the derivation of the observed behavior of convective instability with the prediction each hypothesis makes about R_c that can be compared with experiment. Rayleigh's analysis of buoyancy driven instability [18] has been reproduced and elaborated upon by many others, see e.g. Refs. [20, 21, 30]. For the surface tension case, the linear stability analysis of Pearson [152] was elaborated upon with weakly nonlinear perturbation theory [154, 155]. In both cases, once the governing equations and conduction steady-state solution are established, small perturbations in dynamic variables (temperature and pressure in the buoyancy case; temperature and vertical velocity for surface tension) are introduced. Including the small perturbuations into the equations of motion and linearizing produces the equations governing the perturbations, from which the conditions under which certain perturbations first begin to grow, rather than die away, are derived. That is, the critical R_c at which the conduction state becomes unstable. In both cases, the perturbations that grow first represent oscillatory solutions, and hence convection rolls.

This linear stability analysis is tractable for idealized near-equilibrium systems, but not so for far-from-equilibrium systems. Moreover, the growing perturbation at the critical point in the simplified near-equilibrium cases like the one just discussed can be described with simple spatial oscillations using Fourier modes. Recall though that the central aim of this thesis is to provide a mathematical description for complex localized structures like hurricanes. Even if you could come up with perturbation equations for a pre-hurricane state of a region in the atmosphere, how would we mathematically represent a hurricane to show they are perturbations that grow and saturate out of this state? In fact, even using simplified means to represent the presence of a hurricane, it has been shown that finite-amplitude instability analysis is unsufficient to describe the formation of hurricanes [13]. Furthermore, how would you test various hypotheses about hurricane formation in care-

fully controlled experiments with actual hurricanes?

Chapter 2

Mathematical Preliminaries

The main goal of this chapter is to argue that organized pattern and structure can be mathematically formalized using intrinsic computation. This serves to motivate the use of the local causal states for far-from-equilibrium systems, as well as provide their technical mathematical development. We start by reviewing the physics of computation and then draw parallels between models of computation and complex dynamical systems. The interface of computation and dynamics provided by symbolic dynamics and shift spaces is particularly useful for developing a theory of pattern and structure. From there we will generalize to stochastic processes. This sets us up for the formal theory of intrinsic computation provided by computational mechanics, presented in Chapter 3.

2.1 Theory of Computation and Complex Dynamical Systems

It is instructive to turn back the clock to the late 1960's when the first textbooks on computation came out [86]. Computation had only been formalized three decades prior, in the pursuit of answering foundational questions at the heart of mathematical logic [156, 157]. Concepts that we take for granted now, like *machines* and *algorithms*, had to be explained from scratch. The discrete nature of computation, with a lack of integrals and derivatives, also needed to be explained. From a modern perspective, these early explanations of the math and physics of computation are immediately evocative of complex systems. That is, the difficulties faced when studying the capabilities and limitations of computing ma-

chines are very similar to the difficulties that complex systems present to the traditional tools of physics, as described above in Section 1.6. For example, "In developing a Theory of Computation we are trying to deal with systems composed of a great many parts, or very intricate structures. Classical mathematical methods can do this only in very special situations, and their limitations are very serious", and "...the system can be treated as individually and independently random– this is what happens in Statistical Thermodynamic theories. But it must be stated, explicitly and emphatically, that this is just what does *not* happen when, as in a computation system, the structure has a more organized, purposeful structure," from the Preface of Ref. [86].

It is not all that enlightening to view a computing machine as a complex dynamical system. If one adopts the uncomputability-of-limit-sets definition for complex systems described in Section 1.6, this follows immediately. Viewing a complex dynamical system as performing a computation is much more interesting. In Section 1.6 we found the language of computation theory to be quite helpful in attempting to make more formal statements about complexity and emergence. In this Chapter and Chapter 3 we go further, and show that the mathematical formalism of computation is useful for creating a theory of pattern and structure in complex dynamical systems. Throughout we will see many deep, perhaps even fundamental, connections between complex dynamical systems and computing machines.

2.1.1 Physics of Computation

Central to the theory of computation is the idea of an *effective procedure*, what we now call an *algorithm*: a finite number of finite instructions that always terminates after a finite number of steps and always produces the correct outcome. This idea was formalized by the λ-calculus of Church [156] and the machines of Turing [157]. The two definitions were shown to be equivalent, leading to the notion of *universal* computation: a model of computation is said to be universal, or Turing-complete, if it can emulate any other model of computation. The Church-Turing thesis states that any function on the natural numbers can be calculated through an effective procedure if and only if it can be computed by a Turing machine. There exist Turing machines, called universal Turing machines, that

can emulate all other Turing machines. From the Church-Turing thesis, these can emulate any model of computation and thus serve as a universal model for all computations. Thus *machine* is synonymous with computation; a machine is any formal instantiation of an algorithm and all algorithms can be carried out via a machine.

When interest in computation shifted from the foundations of mathematics to practical considerations of actual computing machines it was natural to extended the Church-Turing thesis to assert that an algorithm can be carried out by a *physical* machine if and only if it can be computed by a Turing machine. Thus abstract models of computation are useful for understanding the physical capabilities and limitations of actual computing machines. Beyond the immediate practical utility of this physical Church-Turing thesis, there are arguments for why it is a reasonable assertion to make [158, 159]. Here we will assume that machines are tethered to the laws of physics, as many did in the early development of computer science; "Perhaps one could even maintain the view that belief in an arithmetic statement is equivalent to the belief that certain machines, if properly built, will work. Thus, I know that the order of summation in addition is irrelevant. I can think of this as a property of abstract number, or as an empirical generalization from experience with counting, or as a necessary property of any machine which adds numbers correctly." [86, pg.6]

2.1.1.1 Machines and Internal States

What then, *is* a machine (often also called automaton)? We will follow the development in Ref. [86, Ch.2]. To start, consider a machine simply as a black box with an input channel and an output channel. As with cellular automata, we consider time to evolve in discrete steps and the inputs and outputs to take values from discrete symbol sets. The input channel S takes values in $\{s_1, \ldots, s_n\}$ and the output channel R $\{r_1, \ldots, r_m\}$. At each moment t an input signal $S(t)$ is sent from the environment E to the machine M. Similarly, at each moment the machine selects an output $R(t)$ that is sent to the environment. To describe the machine's behavior, we need to specify how its outputs depend on its inputs. In general, the output R at time t will depend on the history $H(t)$ up to time t. The history is a record of all the state of affairs concerning M, including

all the inputs M has so far received. At time t the machine receives input s_i and then responds at time $t+1$ with output r_j that depends on the input s_i as well as the internal state of affairs inside M. For the "deterministic" machines considered here, the internal state of M at time t is determined by the history $H(t)$. Thus the behavior of M can be described in terms of a momentary response function:

$$R(t+1) = F\big(H(t), S(t)\big) \ . \tag{2.1}$$

Because of the explicit dependence on the full infinite history, this relation is not directly of much use. Rather, we would prefer a more direct relation between the input $S(t)$ and the output $R(t+1)$. There are infinitely many possible histories, and similar to far-from-equilibrium systems, the internal state of the machine, and hence the output $R(t+1)$ is determined by the particular history leading up to time t. If all past events have separate, independent effects determining the internal state, then the machine requires infinite memory resources to fully specify its output behavior. It may be the case however that some histories produce the same behavior in the machine. Distinct histories may be equivalent with regards to machine behavior.

We define equivalent histories as follows. Imagine that there are two identical copies of a machine M. At time t machine M_1 has history $H_1(t)$ and machine M_2 has history $H_2(t)$. We say that H_1 and H_2 are *equivalent histories* with respect to M if, for every possible subsequent sequence of inputs $S(t), S(t+1), S(t+2), \ldots$ both M_1 and M_2 would yield the same sequence of outputs. That is, there is no way to distinguish machines M_1 and M_2 by testing them with input sequences and observing the corresponding output sequences. The simplest class of machines, which is the class we will be concerned with in this work, are the *finite-state machines* which can distinguish between a finite number of classes of possible histories. These classes are the *internal states* of the machine.

Denote the internal state of a machine M at time t as $Q(t)$, with individual states given as $\{q_1, \ldots, q_p\}$. The definition of internal states as equivalent histories removes the cumbersome history dependence from the response function of finite-state machines,

$$R(t+1) = F\big(Q(t), S(t)\big) \ . \tag{2.2}$$

The output at $t+1$ depends on the input and internal state at time t. What determines the internal state at time $t+1$? Because the internal state $Q(t)$ at time t is determined by the history $H(t)$ and the history $H(t+1)$ at the next time step differs from $H(t)$ by the single input $S(t)$, the internal state $Q(t+1)$ at time $t+1$ can only depend on $Q(t)$ and $S(t)$. We express this in terms of an internal state transition function,

$$Q(t+1) = G\big(Q(t), S(t)\big) \ . \tag{2.3}$$

The two functions F and G give a complete and finite description of the machine M and its behavior.

Note that internal states defined as equivalence classes of histories gives a *minimal description* of the machine. Additional internal states can be added that do not change the behavior of the machine, and are thus in a sense redundant. This will be emphasized more below, when discussing minimal machine presentations of shift spaces.

2.1.2 Dynamical Structure Modeling

At the surface level it may appear as though we have just shifted symbols around in going from Equation (2.1) to Equations (2.2) and (2.3). The distinction this change in symbols represents is enormous. Not only have we gone from a general dependence on infinite histories to finite internal states, we also gain the dynamical structure of state transitions. Beyond having a finite representation of the machine's behavior that we can actually work with, the internal dynamic specifies how the machine is organized to produce structured behavior. As we will see in Section 3.1.6, this structural organization can be represented visually, through machine graphs, as well as algebraically through defining semi-groups [160, 87, 161]. For our purposes this provides both a formal and practical mathematics with which to express pattern and structure in complex dynamical systems.

To make the connection with dynamical systems, let's re-examine the history dependence of machines. Equation (2.1) gives the deterministic dynamical evolution equations governing machine behavior, their equations of motion. Traditionally, deterministic dynamical systems, such as cellular automata and fluid flows, have no history dependence. The state at time $t+1$ is fully determined by the state at time t. The move to internal

states as equivalent histories is useful for removing the dependence on infinite histories, but what if there is no history dependence in the first place?

As we've seen, the nonlinear equations of motion that govern complex dynamical systems can be used to simulate emergent behaviors, but they obfuscate the mechanisms that give rise to these behaviors. It may be useful then to trade the impenetrability of the equations of motion for a history-dependent dynamical description amenable to treatment in terms of internal states and equivalent histories. We outline the basics of this move in Section 2.2 on measurement theory. In this, a continuous dynamical system is observed through the output of a finite-precision measurement device. This results in a structured stochastic process over observed symbol sequences. Symbolic dynamics is the study of such symbol sequences, and we will see how the move to finite-state machines captures the pattern and structure of these processes in Section 3.1.6. Historically, the procedure of studying continuous dynamical systems in terms of measured symbol sequences, shown in Figure 2.1, was referred to as "nonlinear modeling" [162], although this term is not commonly used these days and we will instead use *dynamical structure modeling* to avoid confusion with uses of the term nonlinear modeling in statistics.

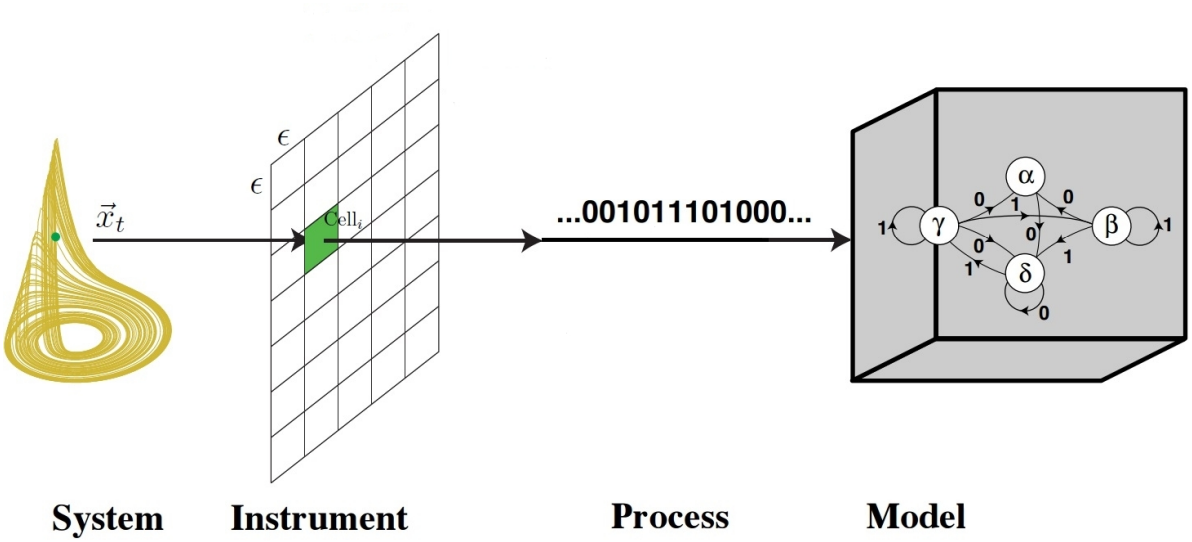


Figure 2.1. Schematic of dynamical structure modeling. A continuous dynamical system is observed using a finite-precision measuring devices to produce a discrete structured stochastic process. The set of allowed symbol sequences is presented by a finite-state machine that captures the structure of the process. Credit: James P. Crutchfield, with modification.

The conceptual use of finite-state machines in dynamical structure modeling is slightly different from what we have just presented for finite-state machines designed to perform useful computations. The mathematics of finite-state machines is used to describe the organization and structure of the dynamics over symbol sequences, without reference to inputs from an external environment. This is why the term *intrinsic computation* is used. (Finite-state machines are most commonly used for the task of language recognition [84], and you can frame the use of finite-state machines in symbolic dynamics in these terms. That is, the observed symbol sequence is seen as the input into a machine tasked with recognizing a particular language. But this is not really what we are interested in. A given process will have a single language of admissible symbol sequences and thus a single associated finite-state machine, see Section 3.1.4 below. It is not so much a question of "does this particular symbol sequence belong to this particular language?" but rather a question of "what is the language of all possible symbol sequences generated by this process, and what does that language tell us about the generating process?")

This very abstract setting is far removed from real physical systems, but allows for clean results and establishes a firm mathematical foundation. Moving to real-world systems, we have already seen how system history is important for far-from-equilibrium systems. For spatially-extended dynamical systems, the global evolution may be deterministic, but the dynamics of localized structures will depend on the history of interactions with the surrounding environment. Taking into account memory effects and history dependence will be necessary for behavior-driven modeling of far-from-equilibrium pattern and structure.

It is instructive to review dynamical structure modeling, as it harbors the origins and foundations of the ideas and tools we will actually use later, namely the local causal states. However, the local causal state modeling we will use does not fully follow this scheme. Constructing "good measuring devices", as described below in Section 2.2, is intractable for real-world systems [163]. In Chapter 7 we will introduce the approximations necessary to bypass measurement discretization and directly model continuous systems.

2.2 Measurement Theory

Let's now examine the relationship between continuum models of dynamical systems and observed symbol sequences that result from finite-precision measurements of the system. As the first step in dynamical structure modeling, this relates the discrete structured stochastic process studied next in Section 2.3 with dynamical systems. Further, the concept of generating partitions will provide a cautionary tale for data-driven approaches to dynamical systems. One must always keep in mind that conclusions drawn from data streams can be a reflection of not just the underlying system of interest, but also the measurement or observation procedure. There are good ways to measure the system, known as generating partitions, that faithfully encode dynamical properties in the observed symbol sequences. In particular, we will review the metric entropy and use it to show how chaos produces randomness, and that the observed randomness of symbol sequences is faithful for generating partitions. For a more in-depth technical review of entropy in dynamical systems, see Ref. [164].

2.2.1 Measurements of a Dynamical System

Consider an ergodic dynamical system $\dot{x} = f(x)$, $x \in M$, with invariant probability measure μ. The system will be measured in discrete time intervals δ, yielding a discrete time dynamic given by the flow map: $\Phi\big(x(t), \delta\big) = x(t) + \int_t^{t+\delta} f\big(x(\tau)\big) d\tau$. We will not be concerned with the details of this temporal discretization, so we will just work with discrete-time models on a continuous state space:

$$x_{n+1} = f(x_n) \ , \tag{2.4}$$

where $x \in M$ and $f : M \to M$ is now a continuous function from the state space to itself and there is still an invariant measure μ on f.

Our interest is in the effects of discretizing state space with finite-precision measurements. We formalize this with a measurement function $g : M \to \mathcal{A}$ that partitions the state space: $P_i \cap P_j = \emptyset$ and $\bigcup_{i=0}^{N} P_i = M$. Each partition element carries a unique measurement symbol $s \in \mathcal{A}$. It is common to set the measurement symbol to be the index of the corresponding partition element, $g(x \in P_i) = i$ and $\mathcal{A} = \{0, \ldots, N\}$, but

in general the symbols are arbitrary (given that each partition element carries a unique symbol). At any given time, the probability of a measurement outputting symbol i is given by $p_i = \int_{x \in P_i} \mu(x) dx$.

The ideal, infinite-precision measurement device would be the case where g is the identity. But for real measurement devices, data storage, and processing, measurements are finite precision (often very high precision, but still finite). Thus our measurements can only output a finite number of symbols.

Measurements of the evolution of dynamical system f with instrument g produces a sequence of symbols $s \in \mathcal{A}$. Starting with initial condition $x_0 \in M$ the instrument outputs the initial measurement $s_0 = g(x_0)$. At the next time step we have $x_1 = f(x_0)$ and the instruments reads $s_1 = g(x_1) = g\left(f\left(x_0\right)\right)$. At the n^{th} time step the instrument reads $s_n = g(x_n) = g\big(f^n(x_0)\big)$. Thus an orbit $x_0, f(x_0), f^2(x_0), \ldots$, of f corresponds to an infinite sequence of output symbols from the measurement instrument $s_0, s_1, s_2, \ldots = g(x_0), g\big(f(x_0)\big) g\big(f^2(x_0)\big), \ldots$.

Recall that g induces a partition P over M. Similarly $g \circ f$ also induces a partition $f^{-1}P$ over M. The elements $(f^{-1}P)_i$ of this partition are all the points $x \in M$ such that $g\big(f(x)\big) \in P_i$. Note that the evolved partitions are given by pre-images of the original partition under f because we are still partitioning M at the initial time, based on what symbol is output after one time step under f. Each time step induces a new partition $f^{-n}P$ whose elements are the points in $x \in M$ such that $g\big(f^n(x)\big) \in P_i$.

The evolved partitions are themselves not of particular interest, but rather the *dynamical refinements* of the initial partition P that they produce. For two partitions P and Q, the partition refinement $P \vee Q = \{P_i \cap Q_j \ : \ P_i \in P \text{ and } Q_j \in Q\}$, is also a partition. The first dynamical refinement of P under f is given as $P \vee f^{-1}P$ and the elements of this partition are sets of points in M that have the same output for $g(x)$ as well as $g\big(f(x)\big)$. The partition $P \vee f^{-1}P$ maps from M to two-symbol sequences $s_0 s_1$ in $\mathcal{A} \times \mathcal{A}$. The full dynamical refinement is $P \vee f^{-1}P \vee f^{-2}P \vee f^{-3}P \vee \cdots$ and maps from M to infinite-length symbol sequences in $\mathcal{A} \times \mathcal{A} \times \mathcal{A} \times \cdots$.

A *generating partition* is one for which there is a one-to-one correspondence between

an initial condition $x_0 \in M$ and the infinite-length symbol sequence $s_0 s_1 s_2 s_3 \cdots$ from measurements of the orbit of x_0, almost everywhere on M. Generating partitions are "good" measurement devices because arbitrarily high precision measurements of the initial condition can be recovered with long enough measurement sequences. In general this need not be the case; two distinct initial conditions may yield the same symbol sequence, at all lengths, for a "bad" partition. For example, if g is the constant function it will produce the same symbol sequence for all initial conditions. As we will see below, generating partitions are also good measurement devices because quantitative properties of the measured symbol sequences correspond to quantitative properties of the continuous model. A special case of generating partitions are the *Markov partitions* for which the stochastic process of measurement symbols is a Markov process. Note though that not all systems f admit a Markov partition.

It is mathematically convenient to consider bi-infinite symbol sequences $\cdots s_{-2} s_{-1} s_0 s_1 s_2 \cdots \in \mathcal{A}^{\mathbb{Z}}$. One can imagine the system f has been evolving for an infinite time in the past and will continue to evolve for an infinite time in the future, being measured with g for the full duration. In this setting, the dynamics $f : M \to M$ of the original continuous model correspond to a trivial linear *shift dynamic* $\sigma : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$ in measurement space. The linear shift operator σ simply moves indices in the bi-infinite symbol sequences. For two symbol sequences $x, y \in \mathcal{A}^{\mathbb{Z}}$, $y = \sigma(x) \iff y_{i+1} = x_i \; \forall i \in \mathbb{Z}$. This is the setting for *symbolic dynamics*, which we will examine in much more detail next in Section 2.3.

2.2.1.1 Example: Logistic Map

The logistic map is a discrete-time dynamical system defined as

$$x_{n+1} = f(x_n) = r x_n (1 - x_n) \; , \tag{2.5}$$

and is historically important in the study of deterministic chaos [165]. We will use it here, with $r = 4$, to demonstrate measurement partitions and their dynamical refinements. The state space for the logistic map is the unit interval, $M = [0, 1]$.

The binary partition P with $P_a = [0, \frac{1}{2}]$, $P_b = [\frac{1}{2}, 1]$ is shown in Figure 2.2. The partition boundary at $\frac{1}{2}$ is shown with the dashed blue line. The pre-images of the

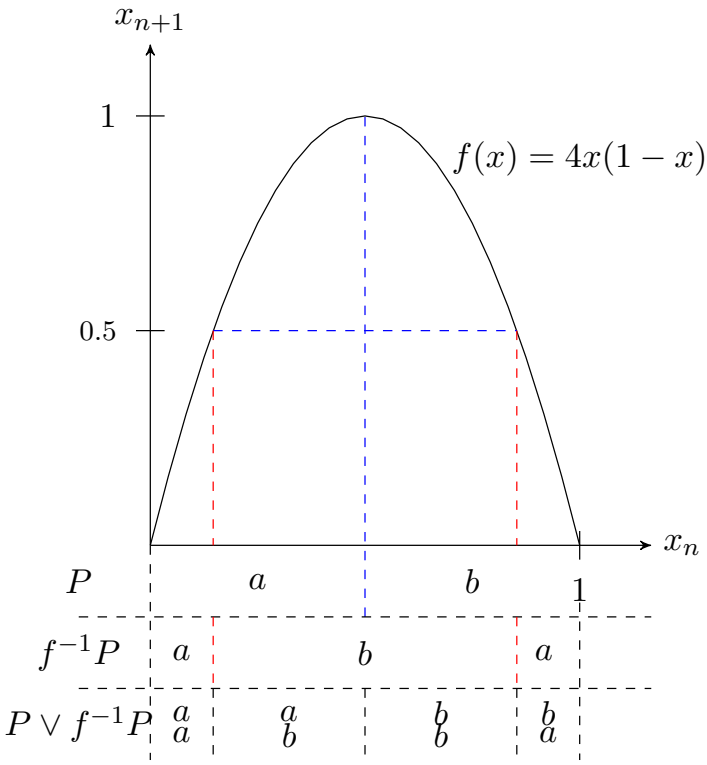


Figure 2.2. Generating partition of the logistic map with $r = 4$, and its first iterate.

partition boundary, $f^{-1}(\frac{1}{2}) = \frac{1}{2} \pm \frac{1}{2\sqrt{2}}$, shown with red dashed lines, give the partition boundaries for the first iterate $f^{-1}P$ of P. Recall that $f^{-1}P$ partitions M based on $g\big(f(x_0)\big)$. We can see that points outside the interval $[\frac{1}{2} - \frac{1}{2\sqrt{2}}, \frac{1}{2} + \frac{1}{2\sqrt{2}}]$ map to values less than $\frac{1}{2}$ under f and points inside the interval map to values greater than $\frac{1}{2}$. The set of intersections between P and $f^{-1}P$ gives the dynamical refinement $P \vee f^{-1}P$ which maps initial conditions in M to the two-symbol sequences they produce. We will not prove it here, but P is a generating partition of the logistic map [166].

2.2.1.2 Metric Entropy

In dynamical structure modeling, the underlying dynamical system is deterministic, but when we apply finite-precision measurements, the dynamics of observed symbols is stochastic. The probability of seeing a given symbol at any time is equal to the probability of the system being in the partition element corresponding to that symbol label: $\Pr(s_i) = \int_{x \in P_i} \mu(x) dx$. The probability of seeing the two-symbol sequence $s_i s_j$ is given by the probability of the system being in the corresponding first partition refinement cell:

$\Pr(s_i s_j) = \int_{x \in P_i \cap (f^{-1}P)_j} \mu(x) dx$, and similarly for length-L sequences.

The entropy of a partition P (induced by measurement device g) is given by

$$H(f, P) = -\sum_i \Pr(P_i) \log \Pr(P_i) \ . \tag{2.6}$$

This quantifies the amount of uncertainty in the single-time output of the measurement device. The *Shannon entropy rate*,

$$h_\mu(f, P) = \lim_{N \to \infty} \frac{1}{N} H\Big(\bigvee_{n=0}^{N} f^{-n}(P) \Big) \ , \tag{2.7}$$

quantifies the asymptotic symbol uncertainty, given all previously measured symbols.

If P is a Markov partition, the measured process is Markov and the entropy rate depends only on the last previously measured symbol. For a general partition, the stochastic process of measured symbols can be highly structured and the entropy rate can depend on symbols measured arbitrarily far back in the past.

Note that the entropy rate is a function of both the dynamical system f and the chosen partition P. A bad partition may yield a symbol sequence that is not at all faithful to the system f, and thus the entropy rate for this partition will not tell us anything meaningful about f. For instance, the constant function g that is the trivial partition $P = M$ will yield an entropy rate of 0 (every measured symbol is the same, so there is never any uncertainty).

To remove the dependence on the chosen partition, we take the supremum over all possible partitions

$$h_\mu(f) = \sup_P \ h_\mu(f, P) \ . \tag{2.8}$$

This is the *metric entropy* of f [94], also known as the *Kolmogorov-Sinai entropy*. It can be shown that this supremum is achieved for generating partitions: $h_\mu(f, P) = h_\mu(f)$ if P is a generating partition.

With some assumptions on the measure μ, the metric entropy is equal to the sum of the positive Lyapunov exponents of f [167],

$$h_\mu(f) = \sum_{\lambda_i > 0} \lambda_i \ . \tag{2.9}$$

For general measures, the metric entropy is bounded by the sum: $h_\mu(f) \leq \sum_{\lambda_i>0} \lambda_i$ [168]. Positive Lyapunov exponents measure the divergence rate of nearby orbits under f and as such serve as a kind of measure of chaos. Pesin's relation, Equation (2.9), shows how this spreading of orbits leads to randomness in measured symbol sequences. But again, this relation only holds for generating partitions.

2.3 Symbolic Dynamics

Measurement theory shows how to produce a discrete structured process from a continuous dynamical system. In the context of dynamical structure modeling, we can consider bi-infinite processes that result from taking measurements of a dynamical system that has been running for an infinite amount of time in the past and will continue evolving for an infinite amount of time in the future. The introduction of an entropy rate for these processes already alludes to non-trivial history dependence. But this is a measure of randomness, not of structure. Before moving into the formal theory of structure provided by computational mechanics in the next Chapter, here we give some basic definitions and properties of these discrete processes, both the metric (stochastic processes) and topological (shift spaces) variants. The use of finite-state machines first arose in symbolic dynamics through the study of minimal presentations of sofic shifts [79, 80, 81]. The computational mechanics literature [103, 169, 104] has largely focused on the metric case, employing minimal probabilistic machines, known as hidden Markov models, to study structure in stochastic processes. The two cases are closely related, and we will detail both for completeness and because both will be useful in the full development of the local causal states and their applications.

2.3.1 Shift Spaces

Consider an indexed bi-infinite sequence of symbols from a finite alphabet $\mathcal{A}$, denoted by $x_{:} = (x_i)_{i \in \mathbb{Z}}$

$$x_{:} = \cdots x_{-2} x_{-1} x_0 x_1 \cdots \ .$$

We denote a contiguous chain of l symbols as $x_{0:l} = x_0 x_1 \cdots x_{l-1}$. Left indices are inclusive; right, exclusive. We suppress indices that are infinite. The *full $\mathcal{A}$-shift* is the set of all bi-infinite sequences of symbols from $\mathcal{A}$, and is denoted by

$$\mathcal{A}^{\mathbb{Z}} = \{(x_i)_{i \in \mathbb{Z}} | x_i \in \mathcal{A} \text{ for all } i \in \mathbb{Z}\} \ .$$

Each sequence $x_{:} \in \mathcal{A}^{\mathbb{Z}}$ is called a *point* of the full shift. A *block*, or *word*, over $\mathcal{A}$ is a finite sequence of symbols from $\mathcal{A}$: $w = x_{[i,j]} = x_i x_{i+1} \cdots x_j$. For two finite words u and v we can *concatenate* them together to form a new word uv.

The *shift map* σ on the full shift $\mathcal{A}^{\mathbb{Z}}$ maps a point $x_:$ to the point $y_: = \sigma(x_:)$ whose ith coordinate is $y_i = x_{i+1}\ \forall\ i$ (i.e. shifts every element in $x_:$ one place to the left). The shift map is one-to-one and onto, so the inverse σ^{-1} exists and shifts the sequence one place to the right. k composition of σ with itself shifts sequences k places to the left, and similarly k compositions of σ^{-1} shifts the sequence k places to the right.

A *shift space* (or *subshift*, or simply *shift*) is a compact, shift-invariant subset of $\mathcal{A}^{\mathbb{Z}}$. The topology is given by the metric $d(x_:, y_:) = 2^{-k}$ if $x_: \neq y_:$ and k is maximal so that $x_{[-k,k]} = y_{[-k,k]}$, and is zero otherwise. Shift spaces, denoted as $\mathcal{X}$, are commonly specified by sets of *forbidden words*. If $x_: \in \mathcal{A}^{\mathbb{Z}}$ and w is a block over $\mathcal{A}$, we say that w *occurs in* $x_:$ if there are indices i and j so that $w = x_{[i,j]}$. Let $\mathcal{F}$ be a collection of forbidden words over $\mathcal{A}$. For any such $\mathcal{F}$, define $\mathsf{X}_{\mathcal{F}}$ to be the subset of sequences in $\mathcal{A}^{\mathbb{Z}}$ which do *not* contain any words in $\mathcal{F}$. A shift space is a subset $\mathcal{X}$ of a full shift $\mathcal{A}^{\mathbb{Z}}$ such that $\mathcal{X} = \mathsf{X}_{\mathcal{F}}$ for some collection $\mathcal{F}$ of forbidden words over $\mathcal{A}$. Since forbidden words in $\mathcal{F}$ are coordinate-free, it follows that they specify a shift-invariant subset of $\mathcal{A}^{\mathbb{Z}}$. Compactness, or closure, is less evident, but it can be shown to always hold.

Let $\mathcal{X}$ be a shift space, and let $\mathcal{B}_n(\mathcal{X})$ denote the set of all length n words in $\mathcal{X}$. The *language* of $\mathcal{X}$ is the collection of all allowed words in $\mathcal{X}$, $\mathcal{B}(\mathcal{X}) = \bigcup_{n=0}^{\infty} \mathcal{B}_n(\mathcal{X})$. This is very similar to the use of language in computation theory [84]. Sets of finite strings recognized by finite-state machines are known as *regular languages*. However, there are two distinguishing properties of the languages of shift spaces. Let $\mathcal{X}$ be a shift space and $L = \mathcal{B}(\mathcal{X})$ be its language. If $w \in L$, then

(a) every subblock of w belongs to L, and

(b) there are nonempty blocks u and v in L such that $uwv \in L$.

The two properties, (a) and (b), characterize the languages of shift spaces. That is, if L is a collection of blocks over $\mathcal{A}$, then $L = \mathcal{B}(\mathcal{X})$ for some shift space $\mathcal{X}$ if and only if L satisfies (a) and (b). Moreover, the language of a shift space determines the shift space. In fact, for any shift space, $\mathcal{X} = \mathsf{X}_{\mathcal{F}}$ for $\mathcal{F} = \mathcal{B}^c(\mathcal{X})$. Thus two shift spaces are equal if and only if they have the same language.

A shift space $\mathcal{X}$ is *irreducible* if for every ordered pair of blocks $u, v \in \mathcal{B}(\mathcal{X})$ there is a block $w \in \mathcal{B}(\mathcal{X})$ so that $uwv \in \mathcal{B}(\mathcal{X})$.

Making a more direct connection with computation theory, a shift space whose language is a regular language is known as a *sofic shift*. (Note that not all regular languages can be the language of a shift space. Those that can, i.e. that satisfy properties (a) and (b) above, are known as *factorial, prolongable regular languages*. The finite-state machines of these languages have all states as accept states. The finite-state machines of irreducible sofic shifts additionally have all states as start states.) The sofic shifts are all those that can be represented by a finite-state machine, and thus will be our objects of interest. The simplest set of shift spaces are the *shifts of finite type* which are specified by a finite set of forbidden words $\mathcal{F}$. The set of sofic shifts is the smallest set that contains all shifts of finite type and is closed under surjective mappings [85].

Consider a transformation from a sequence $x_\cdot = \cdots x_{-1}x_0x_1\cdots$ over $\mathcal{A}$ to a new sequence $y_\cdot = \cdots y_{-1}y_0y_1\cdots$ over another alphabet $\mathcal{U}$ as follows. To compute the i^{th} coordinate y_i of the transformed sequence, we use a function ϕ that depends on the *window* of coordinates of $x_\cdot$ from $i-m$ to $i+n$. The mapping $\phi : \mathcal{B}_{m+n+1}(\mathcal{X}) \to \mathcal{U}$ is a fixed *block map*, called an $(m+n+1)$- block map, from allowed $(m+n+1)$-blocks in $\mathcal{X}$ to symbols in $\mathcal{U}$

$$y_i = \phi(x_{i-m}x_{i-m+1}\cdots x_{i+n}) = \phi(x_{[i-m,i+n]}) \ . \tag{2.10}$$

Let $\mathcal{X}$ be a shift space over $\mathcal{A}$, and $\phi : \mathcal{B}_{m+n+1}(\mathcal{X}) \to \mathcal{U}$ be a block map. Then the map $\Phi : \mathcal{X} \to \mathcal{U}^{\mathbb{Z}}$ defined as $y_\cdot = \Phi(x_\cdot)$ with y_i given by Equation (2.10) is called a *sliding block code* with memory m and anticipation n induced by ϕ. We denote the formation of Φ from ϕ by $\Phi = \phi_\infty^{[-m,n]}$. If $\mathcal{Y}$ is a shift space contained in $\mathcal{U}^{\mathbb{Z}}$ and $\Phi(\mathcal{X}) \subseteq \mathcal{Y}$, we write $\Phi : \mathcal{X} \to \mathcal{Y}$. If Φ is surjective (onto) it is called a *factor map*. If it is injective (one-to-one) it is called an *embedding* of $\mathcal{X}$ into $\mathcal{Y}$. If Φ is bijective (one-to-one and onto) it is called a *conjugacy*.

One of the most important results in symbolic dynamics is the Curtis-Hedlund-Lyndon Theorem, which will be useful for our studies of cellular automata. Let $(\mathcal{X}, \sigma_x)$ and $(\mathcal{Y}, \sigma_y)$ be shift dynamical systems, and $\Phi : \mathcal{X} \to \mathcal{Y}$ a (not necessarily continuous) mapping.

Then Φ is a sliding block code if and only if it commutes with the shift operators [170], $\Phi \circ \sigma_x = \sigma_y \circ \Phi$.

For an in-depth review of shift spaces, see Ref. [78]

2.3.2 Stochastic Processes

In the framework of dynamical structure modeling, the set of possible symbol sequences that result from discrete measurements of a continuous dynamical system form a shift space. If the dynamical system is ergodic, or measurements are taken from a single ergodic component (i.e. an invariant set), then the resulting shift space is irreducible and its invariant measure allows us to additionally assign probabilities to measured symbol sequences. Thus the shift space gives the set of all possible symbol sequences we can measure, while the *stochastic process* gives the probabilities of the symbol sequences.

A general stochastic process $\mathcal{P}$ is the distribution of all a system's behaviors or realizations $\cdots x_{-2}x_{-1}x_0x_1\cdots$ as specified by their joint probabilities $\Pr(\cdots X_{-2}X_{-1}X_0X_1\cdots)$. X_t is the random variable for the outcome of the measurement at time t, taking values x_t from a finite set $\mathcal{A}$ of all possible events. The support of the process (the set of all sequences with non-zero probability) is known as the *process language* $L(\mathcal{P})$. This is the set of all finite and infinite strings that can be realized by the process $\mathcal{P}$. We consider only *stationary* processes for which $\Pr\left(X_{t:t+l}\right) = \Pr\left(X_{0:l}\right)$ for all t and l.

A stationary stochastic process over a shift space $\mathcal{X}$ is an assignment of a shift-invariant probability measure $\mu(w)$ to the words w of $\mathcal{B}(\mathcal{X})$ such that each word satisfies prefix and suffix marginalization:

$$\mu(w) = \sum_{\{a:aw\in\mathcal{B}(\mathcal{X})\}} \mu(aw)$$

$$\mu(w) = \sum_{\{a:wa\in\mathcal{B}(\mathcal{X})\}} \mu(wa)$$

$X_{i:j}$ is the random variable distributed as $\mu(x_{i:j}), \quad x_{i:j} \in \mathcal{B}(\mathcal{X})$. Compactly, one can denote a stationary process generated this way as $\mathcal{P} = \mu(\mathcal{X})$. The process language of $\mathcal{P}$ is equal to the language of $\mathcal{X}$, $\mathcal{B}(\mathcal{X}) = L(\mathcal{P})$. Forbidden words of $\mathcal{X}$ are measure

zero in $\mathcal{P}$, and all allowed words have some finite probability. For more details on the measure-theoretic relations between shift spaces and stochastic processes, see Refs. [171] and [172].

2.4 Spatiotemporal Processes

So far we have considered the dynamical evolution of strictly temporal systems, with a single degree of freedom. Now though, we are interested in dynamical systems that have spatial extent, with some spatial configuration that is evolving in time. We give the necessary background for these spatiotemporal processes.

2.4.1 Topology of Configurations

The *state* x of a spatiotemporal system specifies the values $x^{\mathbf{r}}$ at *sites* $\mathbf{r}$ of a *lattice* $\mathcal{L}$. Assuming values lie in the finite set $\mathcal{A}$, a *configuration* $x \in \mathcal{A}^{\mathcal{L}}$ is the collection of values over the lattice sites.

A spatiotemporal process, in contrast to a purely temporal one, generates a sequence $\cdots x_{-1}x_0x_1\cdots$ consisting of the series of spatial fields x_t. (Subscripts denote time; superscripts sites.) A realization of a spatiotemporal process is known as a *spacetime field* $\mathbf{x} \in \mathcal{A}^{\mathcal{L}\times\mathbb{Z}}$, consisting of a time series $x_0, x_1, \ldots$ of spatial configurations $x_t \in \mathcal{A}^{\mathcal{L}}$. $\mathcal{A}^{\mathcal{L}\times\mathbb{Z}}$ is the orbit space of the process; that is, time is added onto the system's state space. The associated spacetime field random variable is $\mathbf{X}$. A *spacetime point* $\mathbf{x}_t^{\mathbf{r}} \in \mathcal{A}$ is the value of the spacetime field at coordinates $(\mathbf{r}, t)$—that is, at location $\mathbf{r} \in \mathcal{L}$ at time t. The associated random variable at that point is $\mathbf{X}_t^{\mathbf{r}}$. To illustrate, consider points $\mathbf{x}_t^{\mathbf{r}}$ from a spacetime field for a 1-D infinite lattice:

$$\begin{array}{ccccc}
 & \vdots & \vdots & \vdots & \\
\cdots & \mathbf{x}_{-1}^{1} & \mathbf{x}_{0}^{-1} & \mathbf{x}_{1}^{-1} & \cdots \\
\cdots & \mathbf{x}_{-1}^{0} & \mathbf{x}_{0}^{0} & \mathbf{x}_{1}^{0} & \cdots \\
\cdots & \mathbf{x}_{-1}^{1} & \mathbf{x}_{0}^{1} & \mathbf{x}_{1}^{1} & \cdots \\
 & \vdots & \vdots & \vdots &
\end{array}$$

Being interested in spatiotemporal systems that exhibit spatial translation symmetries, we narrow consideration to regular spatial lattices with topology $\mathcal{L} = \mathbb{Z}^d$. (As needed, the lattice will be infinite or periodic along each dimension.)

The collection of all spatial sites within radius R of a site $x^{\mathbf{r}}$, including $x^{\mathbf{r}}$ itself, is

known as the site's *neighborhood* $\eta(x^{\mathbf{r}})$:

$$\eta(x^{\mathbf{r}}) = \{x^{\mathbf{r}'} \ : \ ||\mathbf{r} - \mathbf{r}'|| \leq R; \ \mathbf{r}, \mathbf{r}' \in \mathcal{L}\} \ . \tag{2.11}$$

The neighborhood specification depends on the form of the lattice distance metric chosen. The two most common neighborhoods for regular lattice configurations are the Moore and von Neumann neighborhoods, defined by the Chebyshev and Manhattan distances in $\mathcal{L}$, respectively.

2.4.2 Dynamics

Rather than generalize dynamical structure modeling to measurements of continuous spatiotemporal systems, for simplicity we now consider discrete-time dynamics over a discrete lattice system.

The most general form of lattice dynamics is through a *global evolution operator* $\Phi : \mathcal{A}^{\mathbb{Z}^d} \to \mathcal{A}^{\mathbb{Z}^d}$ that maps full lattice configurations at time t to configurations at $t+1$:

$$\mathbf{x}_{t+1} = \Phi(\mathbf{x}_t) \tag{2.12}$$

A *local dynamic* uses a *local evolution operator* ϕ that updates individual sites based on their local neighborhoods:

$$\mathbf{x}^{\mathbf{r}}_{t+1} = \phi\big(\eta(\mathbf{x}^{\mathbf{r}}_t)\big) \tag{2.13}$$

The list of all neighborhoods η and their corresponding outputs $\phi(\eta)$ is known as the *lookup table*. Local dynamics are used to model the notion of *local interactions* in physical systems.

Global update Φ of a local dynamic is achieved by applying ϕ in parallel to all neighborhoods on the lattice. For one-dimensional spatial lattices given as shift spaces, this global update is a sliding block code. Cellular automata [173, 174] are examples of discrete lattice systems with local update rules that we will study, along with their symbolic dynamics, in Chapters 4, 5, and 6.

2.4.3 Shift Spaces in Spacetime

Local dynamics over infinite or periodic lattice systems leads very naturally to the extension of shift spaces to $d+1$ dimensions [175, 176, 177, 78, 178] by virtue of the Curtis-Hedlund-Lyndon theorem [170].

A full-$\mathcal{A}$ shift is now the collection of all spacetime fields of symbols from $\mathcal{A}$:

$$\mathcal{A}^{\mathbb{Z}^{d+1}} = \{\mathbf{x}_t^{\mathbf{r}} \ , \ (\mathbf{r}, t) \in \mathbb{Z}^{d+1} | \mathbf{x}_t^{\mathbf{r}} \in \mathcal{A} \text{ for all } (\mathbf{r}, t) \in \mathbb{Z}^{d+1}\}$$

The shift map is now indexed by the direction vector of the shift. Equivalently, one can think of each dimension of $\mathbb{Z}^{d+1}$ as having its own shift operator for shifts along that dimension, where shifts along different dimensions commute. Let σ_p denote the *temporal shift operator* that shifts a spacetime field $\mathbf{x}$ p steps along the time dimension. This translates a point $\mathbf{x}_t^{\mathbf{r}}$ in the spacetime field as: $\sigma_p(\mathbf{x})_t^r = \mathbf{x}_{t+p}^r$. Similarly, let σ^{s_n} denote the *spatial shift operator* that shifts a spacetime field $\mathbf{x}$ by s_n steps along the n^{th} spatial dimension. This translates a spacetime point $\mathbf{x}_t^{\mathbf{r}}$ as: $\sigma^{s_n}(\mathbf{x})_t^r = \mathbf{x}_t^{r'}$, where $r'_n = r_n + s_n$.

A $d+1$ *dimensional shift space* $\mathcal{X}$ (again also referred to as *subshift* or just *shift*) is a closed (under the Cantor metric), shift-invariant subset of $\mathcal{A}^{\mathbb{Z}^{d+1}}$:

$$\mathcal{X} = \sigma^{\mathbf{n}}(\mathcal{X}) \ \ \forall \ \ \mathbf{n} \in \mathbb{Z}^{d+1}$$

As with temporal shifts, we can equivalently define shift spaces based on lists of "forbidden motifs". A *shape* is a finite subset $F \subset \mathbb{Z}^{d+1}$. A *motif* f on shape F is a particular realization of symbols from $\mathcal{A}$ over F, $f : F \to \mathcal{A}^F$. Given a list $\mathcal{F}$ of forbidden motifs, we can define a shift space as:

$$\mathcal{X} = \mathsf{X}_{\mathcal{F}} = \{\mathbf{x} \in \mathcal{A}^{\mathbb{Z}^{d+1}} | \sigma^{\mathbf{n}}(\mathbf{x})_F \notin \mathcal{F} \ \ \forall \mathbf{n} \in \mathbb{Z}^{d+1} \ , \ \forall F\} \tag{2.14}$$

We say that a motif f on shape F *occurs* or is *admissible* in a shift space $\mathcal{X}$ if there is a field $\mathbf{x} \in \mathcal{X}$ such that the field restricted to shape F has motif f; $\mathbf{x}_F = f$.

There is no loss of generality in defining a $d+1$ dimensional shift space to specify all motifs in $\mathcal{F}$ over a fixed shape F'. Usually F' is taken to be a $d+1$ dimensional hypercube. This is the easiest way to generalize the notion of words. "Words" of length

l here are $d+1$ hypercubes of side length l. The set of all such words, or equivalently the set of all motifs, gives the language of a spacetime shift space.

Unlike their low-dimensional counterparts, high-dimensional shift spaces (such as spacetime shift spaces) are still largely shrouded in mystery. Many results from the low-dimensional case turn out to be undecidable for high-dimensional shifts. For example, given a finite list of motifs $\mathcal{F}$, it is undecidable if the high-dimensional shift of finite type $\mathcal{X} = \mathsf{X}_{\mathcal{F}}$ is nonempty [179, 180].

2.4.4 Spacetime Stochastic Processes

From a spacetime shift space we can create a *spacetime stochastic process* by assigning a shift-invariant measure over the admissible motifs of the shift space that satisfies the analogue of prefix and suffix marginalization. Points in the spacetime field now become random variables $\mathbf{X}_t^{\mathbf{r}} \sim \mu(\mathbf{x}_t^{\mathbf{r}})$. The spacetime stochastic process then is the joint distribution over this spacetime field of random variables.

Creating a spacetime process in this way guarantees the process has the generalized notion of stationarity (i.e. $\mu(f)$ is independent of where $\mathbf{x}_F = f$ is found in spacetime). Also, the temporal process over spatial configurations is guaranteed to be stationary from this construction.

It should be noted that this is an abstract generalization from the temporal setting. While a cellular automaton may well be described by a $d+1$ shift space, the problem of measures and how they are propagated by CAs is much trickier. Since a CA uses a local update rule that is applied uniformly in time and space, the set of motifs which may occur in the generated spacetime field is (asymptotically) independent of where you look in the field. However, the density of motifs may change over time. Typical CA behavior is, in fact, non-stationary in this way [100].

Chapter 3

Computational Mechanics

Having set up structured processes and how they can result from measurements of dynamical systems in Chapter 2, we now introduce the use of computational models, i.e. machines, to formally capture the *structure* of structured processes. Meant to evoke a computation-theoretic extension of statistical mechanics, this framework is known as *computational mechanics* [104].

A machine representation of a process is known as a *presentation*. We will start in the more general setting of presentations for stochastic processes (their support is always a shift space), since this will allow us to formalize the intrinsic randomness of a process, as well as its intrinsic structure. However, as structure is largely a topological property, we will also introduce machine presentations of shift spaces.

3.1 Temporal Presentations

The canonical presentation computational mechanics uses for a given stochastic process is a form of *hidden Markov model* (HMM) [181]: the *ϵ-machine*, which consists of a set $\mathbf{\Xi}$ of *causal states* and a transition dynamic T over the causal states [169]. ϵ-Machines satisfy three conditions: irreducibility, unifilarity, and probabilistically distinct states [182]. *Irreducibility* implies that the associated state-transition graph is strongly connected. *Unifilarity*, perhaps the most distinguishing feature, means for each state $\xi \in \mathbf{\Xi}$ and each observed symbol $x \in \mathcal{A}$ there is at most one outgoing transition from ξ labeled x. Critically, unifilarity enables one to directly calculate various process quantities, such

as conditional mutual informations, using properties of the hidden (causal) states. Notably, many of these quantities cannot be directly calculated using the states of general (nonunifilar) HMMs. Finally, an HMM has *probabilistically distinct states* when, for every pair of states ξ and ξ', there exists a word w such that the probability of observing w from each state is distinct: $\Pr(w|\xi) \neq \Pr(w|\xi')$. An HMM satisfying these three properties is an ϵ-machine.

We note that the scope of computational mechanics does extend beyond the ϵ-machine, depending on what questions you ask of a process. The ϵ-machine is the canonical presentation of a process in computational mechanics as it is often easily constructed and can tell you a lot about a process. It is centered around the task optimal prediction of the future from the past. That is, getting the correct conditional distribution over the future given the past. Other tasks, such as minimal (potentially nonunifilar) generation of a process, may require a different presentation [183].

3.1.1 Causal States and the Causal Equivalence Relation

The set of causal states of the ϵ-machine

$$\mathbf{\Xi}_t = X_{:t}/\sim_\epsilon \tag{3.1}$$

is the partition over the space of all pasts defined via the *causal equivalence relation*:

$$x_{:t} \sim_\epsilon x'_{:t} \iff \Pr(X_{t:}|X_{:t} = x_{:t}) = \Pr(X_{t:}|X_{:t} = x'_{:t}) \ . \tag{3.2}$$

In words, the causal equivalence relation says that two pasts are considered causally equivalent if they make the same prediction over futures. The distributions over futures (predictions) are known as *future morphs*. Each causal state $\xi \in \mathbf{\Xi}$ is an element of the coarsest partition of a process's pasts such that every $x_{:0} \in \xi$ has the same future morph $\Pr(X_{0:}|\cdot)$. Due to stationarity, each past has a well defined future morph, and thus every past has an associated causal state. The *ϵ-function* $\epsilon(x_{:t})$ is the map that takes pasts to their causal states $\epsilon : X_{:0} \to \xi$ and thus generates the partition defined by the causal equivalence relation. The causal states are the unique *minimal sufficient statistic* of the past to predict the future [169]. Causal equivalence and optimal prediction will be the springboard for generalizing to spatiotemporal systems.

Notice that causal equivalence is essentially a stochastic generalization of the equivalent histories definition of finite-state machines given in Section 2.1.1. Recall though that the use of machines in dynamical structure modeling is for intrinsic computation, with no reference to an external environment providing input; histories of past behavior of the system are equivalent if they lead to the same future behavior of the system. Thus the causal states can be thought of as the internal memory states of intrinsic computation for a process.

3.1.2 Causal State Transitions

The causal equivalence relation provides a natural unifilar dynamic over the causal states. At time t the ϵ-function maps the past $X_{:t}$ to its causal state ξ_t. At the next time step the new updated past is $X_{:t+1} = X_{:t}X_t$, which the ϵ-function maps uniquely to causal state ξ_{t+1}. Thus there is at most one causal state ξ_{t+1} that the causal state ξ_t can transition to upon seeing X_t. This guarantees the ϵ-machine is unifilar.

Moreover, this gives us the transition dynamic. One can think of ξ_t as an encapsulation of the past $X_{:t}$ and ξ_{t+1} as an encapsulation of the the past $X_{:t+1} = X_{:t}X_t$. These two pasts differ only by the observed symbol X_t; thus to get the full summary ξ_{t+1} one just needs the past $X_{:t}$ as summarized by ξ_t and the observed symbol X_t. The probability of this transition occurring is just the probability of seeing the observed symbol X_t given the ϵ-machine was in state ξ_t at time t. All such transitions are given mathematically as a *symbol-labeled transition matrix* $T^{(x)}$ which has elements that give the probability of transitioning from state ξ to ξ' on the symbol x:

$$T^{(x)}_{\xi\xi'} \equiv \Pr(X_t = x, \xi_{t+1} = \xi' | \xi_t = \xi) \ . \tag{3.3}$$

The overall internal state dynamics is governed by the stochastic matrix $T = \sum_x T^{(x)}$. Its unique left eigenvector π, associated with eigenvalue 1, gives the *asymptotic state probability* $\Pr(\xi)$. By extension, the transition matrix giving the probability of a word

$w = x_0 x_1 \cdots x_{\ell-1}$ of length ℓ is the product of transition matrices of each symbol in w:

$$\begin{aligned} T^{(w)} &\equiv \prod_{x_i \in w} T^{(x_i)} \\ &= T^{(x_0)} T^{(x_1)} \cdots T^{(x_{\ell-1})} \ . \end{aligned} \tag{3.4}$$

The set of causal states together with the set of symbol-labeled transition matrices gives the ϵ-machine. Defining the ϵ-machine in this way, using the causal equivalence relation, is equivalent to the first definition of an ϵ-machine machine given above as a hidden Markov model with the three properties of irreducibility, unifilarity, and probabilistically distinct states [182].

3.1.3 Basic Measures

The *statistical complexity* of a process [103] is the average amount of information about the past that must be stored to predict the future as well as possible. Since the causal states are minimal sufficient statistics for this future prediction, the complexity of a process is also the amount of information needed to specify its causal state [169]. We can thus calculate the statistical complexity of a process using the process's ϵ-machine:

$$\begin{aligned} C_\mu(\mathcal{P}) &\equiv H[\Xi] \\ &= -\sum_{\xi \in \Xi} \Pr(\xi) \log_2 \Pr(\xi) \ , \end{aligned} \tag{3.5}$$

where $\Pr(\xi)$ is the asymptotic state probability defined above.

Complexity of a process is to be contrasted with the intrinsic randomness of a process, as these two notions are often conflated. Statistical complexity is the uncertainty in the causal states of the process, while the *entropy rate* of the process is the uncertainty in observed symbols:

$$h_\mu(\mathcal{P}) = \lim_{L \to \infty} \frac{H[X_{t:t+L}]}{L} = \lim_{L \to \infty} H[X_t | X_{t-L:t}] \tag{3.6}$$

For stationary processes this quantity is the same for all t. The last expression in Equation (3.6) let's us calculate h_μ using the ϵ-machine for the process:

$$h_\mu(\mathcal{P}) = -\sum_{\xi \in \Xi} \Pr(\xi) \sum_{x \in \mathcal{A}, \xi' \in \Xi} T^{(x)}_{\xi\xi'} \log_2 T^{(x)}_{\xi\xi'} \tag{3.7}$$

It should be noted that these and other information-theoretic measures of processes are independent of the computational mechanics formalism [184, 185]. However, since ϵ-machines are mathematical representations of processes, these information measures can be computed using the machine, as demonstrated above. In fact, this ability to calculate many information quantities from the ϵ-machine makes it a particularly useful representation of a process. For more details on information theory and ϵ-machines, see [186, 187, 188, 189].

Recall that the entropy rate of a process is equal to the metric entropy only if the measurement partition is generating. In the dynamical structure modeling framework, the ϵ-machine is constructed directly from the measured process and as such is a reflection of the underlying dynamical system as well as how it is measured. The ϵ in ϵ-machine symbolizes this dependence on the measurement partition.

3.1.4 Topological Machines

Since shift spaces deal only with what words are allowed or not allowed, rather than distributions over those words, shift spaces are presented by labeled graphs (i.e. finite-state machines) instead of HMMs. These are graphs $G(E, V)$ where every edge $e \in E$ of the graph carries a label $\mathfrak{L}(e)$ from the alphabet $\mathcal{A}$. Let $\mathcal{W} = ...e_{-1}e_0e_1...$ be a bi-infinite walk on G and $\mathfrak{L}_\infty(\mathcal{W}) = ...\mathfrak{L}(e_{-1})\mathfrak{L}(e_0)\mathfrak{L}(e_1)...$ be the label of the walk. The set of all labels of bi-infinite walks on G is denoted by $\mathsf{X}_G = \{\mathfrak{L}_\infty(\mathcal{W}) : \mathcal{W} \in G\} = \mathfrak{L}_\infty(G)$. A labeled graph presents a shift $\mathcal{X}$ iff $\mathsf{X}_G = \mathcal{B}(\mathcal{X})$ [78].

A presenting graph G of a shift $\mathcal{X}$ is said to *unifilar* or *right-resolving* if for every vertex $v \in V$ and symbol $x \in \mathcal{A}$, there is at most one outgoing edge e from v labeled with the symbol x. As shown by Fischer [79, 80], an irreducible sofic shift $\mathcal{X}$ always has a unique, minimal, unifilar presenting graph which is strongly connected. This presentation is referred to as the (right) *Fischer cover* of $\mathcal{X}$.

A related presentation of a shift space $\mathcal{X}$ is the *future cover* or *Krieger cover* of $\mathcal{X}$ [182]. The *future* or *follower set* $F_\mathcal{X}(x_{:i})$ of a left-infinite sequence (past) $x_{:i}$ in $\mathcal{X}$ is the collection of all right-infinite sequences (futures) $x_{i:}$ such that $x_{:} = x_{:i}x_{i:} \in \mathcal{X}$. Define an

equivalence relation $\sim_K$ on the set of left-infinite pasts in $\mathcal{X}$ as:

$$x_{:i} \sim_K x_{:j} \iff F_{\mathcal{X}}(x_{:i}) = F_{\mathcal{X}}(x_{:j}) \tag{3.8}$$

where $x_{:} \in \mathcal{X}$, and i and j are arbitrary indices that may or may not be equal. Relation (3.8) can be thought of as a *topological causal equivalence relation*, and the follower sets as *topological morphs*.

The Krieger cover of $\mathcal{X}$ is the directed, edge-labeled graph G whose vertices are equivalence classes of pasts $x_{i:}$ under the relation $\sim_K$. There is a directed edge in G from vertex v to vertex v' labeled with symbol x, if for some past $x_{i:} \in v$ the past $x_{:i+1} = x_{:i}x \in v'$. As with the ϵ-machine and causal equivalence, construction by topological equivalence guarantees the Kreiger cover is unifilar.

The Kreiger cover is not necessarily irreducible, but it will always have a single recurrent, irreducible component that is isomorphic to the Fisher cover. As we will not be concerned with synchronization and the transient structure of Kreiger covers, we will refer to general "topological machines" in what follows, by which we mean the Fisher cover or, equivalently, the recurrent component of the Kreiger cover. We note though that our reconstruction algorithm for topological machines, see Section 4.2, utilizes the topological equivalence relation that defines the Kreiger cover.

For these topological machines the symbol-labeled transition matrices are replaced by adjacency matrices with binary entries to indicate whether the transition on that symbol is allowed or not. Statistical complexity and entropy rate are replaced by their topological counterparts, which can often be calculated by simply assuming a uniform distribution over states and transitions in Equations (3.5) and (3.7) (in general, distributions which maximize the entropy rate must be used). The topological complexity measures the amount of memory required to specify the language of $\mathcal{X}$ and the topological entropy measures the growth rate of allowed symbol sequences in $\mathcal{X}$.

3.1.5 An Example: The Even Shift

To briefly illustrate the strictly-temporal concepts just reviewed we now go through a canonical example of the Even Shift and the Even Process, with the associated topological

and ϵ-machines. The Even Shift [85] is a subshift of the Full Binary Shift $\mathcal{A}^{\mathbb{Z}}$ with $\mathcal{A} = \{0, 1\}$. It is named the Even Shift because only even blocks of 1s bounded by 0s are allowed. Thus the forbidden words are odd blocks of 1s bounded by 0s. There is a countable number of such forbidden words, so the Even Shift is not of finite type, but it is sofic since it can be presented by a finite-state machine, as shown in Figure 3.1 (a). This is the topological machine presentation of the Even Shift.

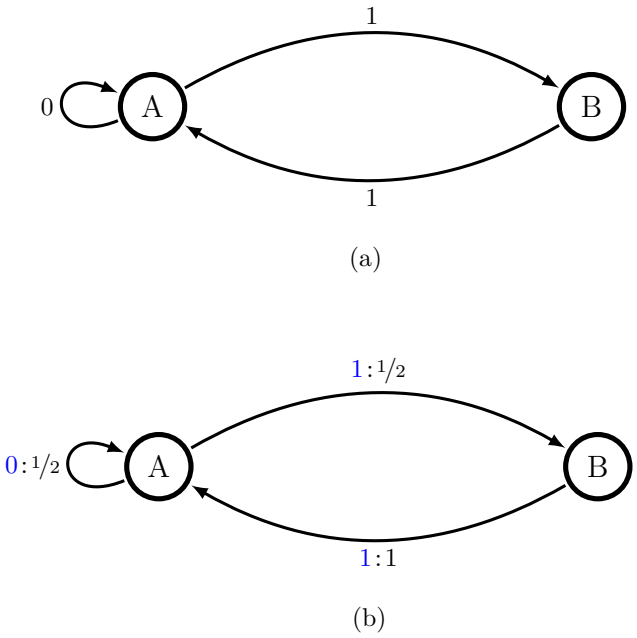


Figure 3.1. (a) Topological machine presentation of the Even Shift. (b) ϵ-machine presentation of the Even Process

To give context, note that the Full Binary Shift is presented by a machine with a single state that has two self-loops, one for each of the binary symbols. Recall the Full Binary Shift is the set of all binary strings, and is defined by $\mathcal{F}$ as the empty set; no words are forbidden. The Even Shift, in contrast, has infinitely many forbidden words. However, just a single additional state is required in its topological machine presentation to specify all these forbidden words. And, the machine graph provides a *semantic* interpretation for the Even Shift. Think of transitions between states in the machines as allowed concatenations of symbols onto allowed words. Starting in state A with the empty word, we can concatenate both a 1 and a 0, as these are allowed transitions leaving state A. If we concatenate a 0 we return to state A and can repeat. If we concatenate a 1, however, we

move to state B and from there we can *only* concatenate another 1 for the next symbol. This shows how the machine is *organized* to ensure that only pairs of 1s are ever produced, guaranteeing there will be no forbidden blocks of an odd number of 1s bounded by 0s.

A class of Even processes can be formed by applying a shift-invariant measure over the Even Shift. This is most commonly done by adding fixed probabilities to the transitions of the topological machine, resulting in the ϵ-machine for that process. Constructing a stationary process from a shift in this way guarantees the machine presentations will be topologically equivalent. Said another way, the support of the stochastic process will be equal to the shift. We note though that going the other direction can be more complicated. If one starts with a stationary stochastic process and then defines a shift as the support of that process, the topological machine of that shift is not necessarily topologically equivalent to the ϵ-machine of the starting process.

The Even Process typically refers to an assignment of uniform probability to the transitions of state A. As with the topological machine of the Even Shift, the semantics of the Even Process are easy to understand from the ϵ-machine, shown in Figure 3.1 (b). Starting in state A, the process is just flipping a fair coin, except that when a 1 is produced, the next symbol produced is guaranteed to also be a 1, then it goes back to a fair coin. We can see how this is a *structured* stochastic process, and how the ϵ-machine fully captures that structure. In the same way the finite presentation of the topological machine accounts for an infinite number of forbidden words, the finite presentation of the ϵ-machine captures the infinite Markov order of the Even Process. With strings of 1s the "phase" (whether that string is even or odd in length) must be remembered, and as strings of 1s may be arbitrarily long in the Even Process a Markov chain presentation would need to be infinite Markov order to keep track of that phase.

3.1.6 Algebraic Theory of Patterns as Generalized Symmetries

Unlike the use of group algebra to formalize perfect symmetry, there is no universally accepted formalism for pattern and structure. There is no way to rigorously *show* that machine presentations are the "correct" mathematical formalism. An argument must be made for this. We have given some arguments at the beginning of this chapter to justify the

introduction of machines. Ref. [169] gives a more thorough discourse on the general notion of pattern; "some object O has a pattern P — O has a pattern 'represented', 'described', 'captured', and so on by P — if and only if we can use P to predict or compress O". Further, they argue that the machine presentations of computational mechanics are the proper objects O to capture the pattern P in a process $\mathcal{P}$, as they satisfy five desiderata for pattern:

1. *Algebraic*, giving us an explicit breakdown or decomposition of pattern into its parts;
2. *Computational*, showing how the process stores and uses information;
3. *Calculable*, analytically or by systematic approximation;
4. *Causal*, telling us how instances of the pattern are actually produced; and
5. *Naturally stochastic*, not merely tolerant of noise but explicitly formulated in terms of ensembles.

Our approach here has been different from that of Ref. [169] in that we have argued for the use of machines from the start, and in so doing have encompassed, to some degree, all of the above points under the umbrella of computation. Let us summarize and elaborate upon these points in our context, and show how machines tie them all together.

As emphasized in the quote above, and as we emphasized in the previous Chapter, Section 1.3.1, *pattern derives from optimal prediction with minimal resources*. This is the computational aspect of Ref. [169]. The Kolmogorov-Chaitin complexity [190, 191, 192] is discussed as a particular computational approach to pattern. In this, the pattern of an *individual* object is captured by the shortest program, typically given as a universal Turing machine, that exactly reproduces the object. The length of this shortest program is the Kolmogorov-Chaitin complexity of the object. This measure is maximal for random sequences, and as such captures "degree of randomness" rather than "degree of organization". Moreover, the quantity is generally not computable, due to the halting problem. Thus, while Kolmogorov-Chaitin complexity clearly satisfies the computational aspect of pattern, it is not calculable nor naturally stochastic. In fact, if we wish to distinguish

structure from randomness, as we do in this work, then Kolmogorov-Chaitin complexity is a clear example of why pattern must be formulated in terms of ensembles. A finite-state machine presentation is a program, and from causal equivalence we want the shortest such program that optimally predicts. However, this is optimal prediction of ensemble behavior, rather than individual sequences. Take for instance the Full Binary Shift. For a generic sequence $x_{:} \in \mathcal{A}^{\mathbb{Z}}$ its symbols are entirely independent and so no program can compress it. However, the topological machine presentation has just a single state. As an ensemble, the Full Shift is minimally complex; nothing is forbidden and all concatenations of 0s and 1s are allowed. This is a short description. It is maximally random though, so any individual sequence must be fully described, symbol by symbol.

The organizational semantics of machines, as described in the Even Shift example above in Section 3.1.5, captures the causal aspect of Ref. [169]. Machine graphs show, in a visually interpretable manner, how the allowed symbol concatenations are structured to produce the allowed sequences of the shift space. Allowed concatenations lead us to the strongest argument for the use of machines to formalize pattern; the algebraic argument.

As mentioned before, finite-state machines (and thus sofic shifts) have a defining semi-group algebra [160, 85, 87, 161]. Recall that a group is a set of elements closed under an associative, invertible binary operation with an identity element. A semi-group generalizes this by dropping the requirements for invertibility and identity element. The elements of a machine's semi-group $G(M)$ are words (symbol blocks) and the binary operation is word concatenation. Crucially, there is an absorbing semi-group element $e \in G(M)$ such that $we = ew = e$ for all words $w \in G(M)$. This element defines the algebra of a particular machine through the following; for any two words u and v in the language of the machine $L(M)$ and their concatenation $w = uv$, if $w \in L(M)$ than $uv \neq e$ is an element of $G(M)$, and if w is not in $L(M)$ than $uv = e$ in $G(M)$. Thus the semi-group algebra of a machine sends two allowed words to the absorbing element e if and only if their concatenation produces a word forbidden by the language of the machine. We can relate the semi-group of a machine to its machine graph through the generators of $G(M)$. The generators are the single-symbol concatenations, along with e. The single-symbol concatenations are

the labelled edges of the machine graph; its state transitions. Any "missing" edge that could be drawn between two states, including a state to itself, that is not a copy of an existing edge, is an implicit transition that leads an absorbing "forbidden words state". All concatenations that lead to the forbidden word state map to e in $G(M)$.

An example for the Even Shift is shown in Figure 3.2. The forbidden state $\mathcal{F}$ cycles on the absorbing element e to depict that all transitions leading to $\mathcal{F}$ produce forbidden words and thus map to e. Recall that the minimal machine presentation shown here is best understood as a generative model; starting from any state and following transitions produces words in the language of the Even Shift (as long as they don't lead to $\mathcal{F}$). Non-asymptotic machines for both left and right concatenations that can be used to recognize sequences from the Even Shift are given in Ref. [87].

The Even Shift is described by the following semi-group relations [87]:

$$\begin{aligned} 010 &= e \\ 0^2 &= 0 \\ 1^3 &= 1 \\ 01^2 &= 1^2 0 = 0 \ . \end{aligned}$$

The smallest forbidden pattern of 'an odd number of 1s bounded by 0s' is 010. This is the irreducible forbidden pattern; thus the main defining relation of the Even Shift is $010 = e$. The relations $1^3 = 1$ and $01^2 = 1^2 0 = 0$ can be used to reduce longer sequences of odd-length blocks of 1s bounded by 0s to 010; e.g. $0111110 = 01^2 1^3 0 = 010 = e$. Finally, there are no restrictions on sequences of 0s in the Even Shift, which gives the relation $0^2 = 0$.

To close our arguments for the use of machines to formalize pattern and structure, let's re-examine exact symmetry through this lens. There is an appealing logic to the algebraic perspective. Groups capture symmetry; semi-groups generalize groups; machines are defined by semi-groups; so machines capture pattern as generalized symmetry. Ref. [169] introduced an identity element, the null symbol, to their machine algebra in an attempt to formalize this appeal. Only semi-groups with an identity, called monoids, can contain

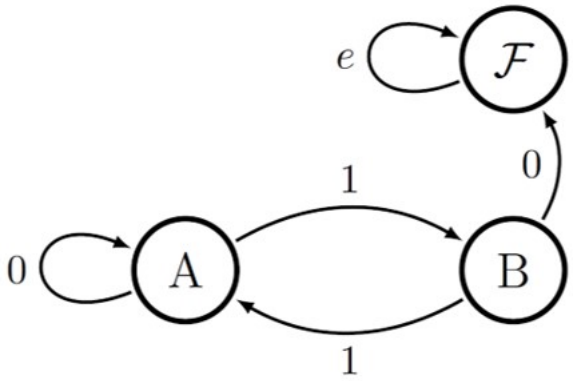


Figure 3.2. Minimal machine presentation of the Even Shift that includes a "forbidden word state" $\mathcal{F}$. All transitions that lead to this state produce forbidden words and thus the associated concatenations map to the absorbing semi-group element e in the machine algebra $G(M)$.

proper groups as sub-groups. However, the binary operation of machine semi-groups is string concatenation, and the inverse of string concatenation is string splitting, which can not be performed an arbitrary number of times for finite strings. Thus it can not be that a machine's semi-group can contain a symmetry group as a proper sub-group in this way.

That being said, we can still view the semi-group algebra of machines as generalizing the group algebra of symmetries, in the one-dimensional case, in the following way: all translation-invariant strings, $\sigma^p(x_:) = x_:$, define a shift space and thus can be presented by a machine. Translation-invariant strings are produced by semi-groups with the following asymptotic property[1]: for all allowed words w, there is one and only one generator $x \in \mathcal{A} \neq e$ such that $wx \neq e$. That is, for every word there is one and only one symbol that can be concatenated to produce an allowed word. For the machine graph, this means for every state there is one and only one transition leading to it and one and only one transition leaving it. From this point of view, we see exact symmetry as a highly restrictive property.

This generalization also adds a computational perspective, as described by desideratum 2. above, to symmetry groups. One may not initially think of symmetry groups as "storing and processing" information. The machine presentation of translation invariance makes this computational aspect of symmetry groups explicit. To elaborate, recall that a

[1] A translation-invariant string s can be described in terms of a minimal block b such that $s = \cdots bbb \cdots$. This is an asymptotic property because the word w must be larger than b for it to hold.

symmetry group is the group of all transformations that leaves an object unchanged. The quotient group of a symmetry thus contains all the transformations that *do* change an object, but sequentially lead to an overall transformation that brings the object back to itself. One can think of the elements of the quotient group of a symmetry as being memory states, or more accurately as the transitions between memory states. They capture the contextual information required for a regular pattern. The group relations specify how these pieces fit together to create the particular pattern. In the one-dimensional case of a periodic tiling, the generator and defining relation for the quotient group is a counter that tracks the phase of the periodic pattern. This can be regarded as memory because the starting point must be stored when traversing the pattern to specify when it repeats.

As an example, consider the shift of strings with the form:

$$\cdots 111000111000111000111000111000111000111000111000111 \cdots ,$$

with the repeating pattern of three 1s followed by three 0s (or vice versa). The minimal machine presentation of this shift is shown in Figure 3.3. This pattern requires six unit shifts to return back to itself, and thus the machine has six internal states, reflecting the quotient group of this symmetry. To ask whether a 1 is in the middle of a block of 1s, or on the right end of a block, is equivalent to asking whether that 1 was generated from state B or state C. It is in this way that the internal states act as a counter tracking the phase of the repeating pattern.

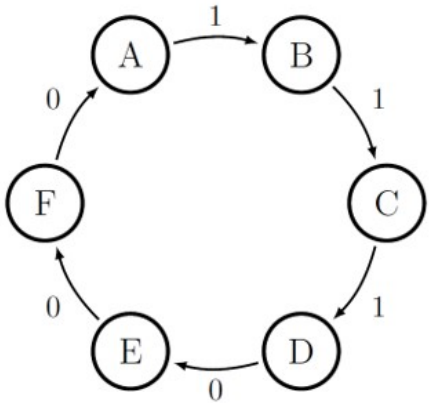


Figure 3.3. Minimal machine presentation of the shift space of translation-invariant strings $\cdots 111000111000111 \cdots$.

3.2 Spatiotemporal Presentations

Moving to systems with spatial dimensions introduces many conceptual and technical difficulties for generalizing computational mechanics. The global approach, outlined below, is the most straightforward way to apply computational mechanics to spatiotemporal systems, though it is essentially intractable in practice. The work of Hanson and Crutchfield [5, 193, 194, 116, 4], described further in Section 4.4.1, is an attempt at handling a global, configurational type approach in a manageable way. Instead of focusing on how all possible configurations evolve, focus on special sets of homogeneous, dynamically invariant configurations. While grounded in solid dynamical systems principles and highly successful in 1D cellular automata, this method has its own difficulties and shortcomings. Most notably in practical application of the method to higher dimensional systems (though still theoretically sound in any dimension).

It turns out to be quite fruitful to take a local, rather than global, perspective when moving to the spatially-extended setting. That is, instead of seeking a presentation for the global dynamic, try to find presentations for each site in spacetime that encode the rules for local transitions in time and space. Such a local spatiotemporal computational mechanics however is not so straightforward. At present, this local theory is incomplete. As mentioned for computational mechanics in its original temporal setting, there is not a single machine presentation for a process which answers all questions about that process. This situation is even worse for local spatiotemporal computational mechanics. For instance, the ϵ-machine is the presentation for optimal prediction of the future of a process, given the past. In the time series case, there is only one notion of past and future. But there may be different notions of a local future which one may want to predict in a spatiotemporal system. There can thus be a multitude of different local presentations for a spatiotemporal process [195]. Not just answers to different questions, but also there are now different ways to ask the same question in spatiotemporal settings (e.g. not just prediction vs generation, but now also *what* do you want to predict or generate).

First we outline the global approach and why it is infeasible in practice. Then we give a detailed review of the standard local approach, the *local causal states*, as first given in

Ref. [110]. Notice we will be using just the states, and not a full machine (i.e. states and their transitions). Similar to the intractability of high-dimensional shift spaces, regular languages and finite-state machines do not have straightforward generalizations to higher dimensions [196]. Though we do not give the details here, the transition structure of the local causal states outlined in Ref. [110] does not yield consistent presentations of their associated spacetime shifts [195]. Despite this, we will see in the following Chapters that the local causal states are nonetheless capable of capturing complex pattern and structure in spacetime.

3.2.1 Global ϵ-Machine

The most obvious way to tackle spatiotemporal systems with computation mechanics is to treat the system as a time series of evolving configurations. We can thus reduce the problem to the familiar time series setting of computational mechanics, only now with an alphabet size exponential in the size of the spatial lattice. While this can be done in theory, one quickly sees that this approach is intractable for all practical purposes. Not only would you need an immense amount of computation power and data to process in order to reconstruct a global machine, but even if you could get a result it would present structure in the dynamics at the full configuration level. Most often in spatially-extended systems there is interest in localized structures in space and how they evolve in time. Global causal states would only be able to code for full configurations. It would be difficult to access information on a particular localized structure since this structure may appear in various configurational contexts and it is only the full configurations that are being represented with this global spacetime ϵ-machine.

3.2.2 Local Causal States

The causal equivalence relation is the core of computational mechanics, and this remains the case when moving to spatiotemporal systems. What we need are notions of past and future in the spatiotemporal setting. The global machine uses the same notion of pasts and futures as the temporal setting but with full spatial configurations rather than a single observed symbol.

Since the global approach is intractable, we instead take a local approach. We seek not pasts and futures of the full spatial configuration, but rather local pasts and futures for each site on the spatial lattice. This is the opposite extreme from the global approach and, as we will see, has many advantages. Chief among them being tractability; the huge space of configurations is replaced by a smaller (though still high dimensional) space of local pasts and futures as discussed below. This approach also allows for clear and interpretable representations of localized structures that emerge in the spatial lattice. Since the single lattice site representations can be composed together to form larger sized representations (including the full spatial lattice), as discussed below, this approach can uniformly capture pattern and structure at all scales in the system.

3.2.2.1 Local Analogues

The simplest local approach is to look at the time series produced by the evolution of each site on the lattice

$$\ldots, \mathbf{x}^{\mathbf{r}}_{-1}, \mathbf{x}^{\mathbf{r}}_{0}, \mathbf{x}^{\mathbf{r}}_{0}, \ldots$$

This approach though ignores the explicit interactions between the sites, which is exactly what we are interested in as it is these interactions that produce complex patterns and structures in space and time.

Thus we want to take into account not just the pasts and futures of the individual lattice sites, but also the spacetime points that they could affect and could be affected by through local interactions. For systems that evolve according to a homogeneous local dynamic, influence can only propagate at a finite speed through the lattice, and this naturally leads to the use of *lightcones* as local pasts and futures.

Formally, the *past lightcone* $\mathtt{L}^-$ of a spacetime random variable $\mathbf{X}^{\mathbf{r}}_t$ is the set of all random variables at previous times that could possibly influence it. That is:

$$\mathtt{L}^-(\mathbf{r}, t) \equiv \left\{ \mathbf{X}^{\mathbf{r}'}_{t'} : \; t' \leq t \text{ and } ||\mathbf{r}' - \mathbf{r}|| \leq c(t - t') \right\} , \tag{3.9}$$

where c is the finite speed of information propagation in the system. Similarly, the *future lightcone* $\mathtt{L}^+$ is given as all the random variables at subsequent times that could possibly

be influenced by $\mathbf{X}_t^{\mathbf{r}}$:

$$\mathtt{L}^+(\mathbf{r}, t) \equiv \left\{ \mathbf{X}_{t'}^{\mathbf{r}'} : \; t' > t \text{ and } ||\mathbf{r}' - \mathbf{r}|| \leq c(t' - t) \right\} . \tag{3.10}$$

We include the *present* random variable $\mathbf{X}_t^{\mathbf{r}}$ in its past lightcone, but not its future lightcone. An illustration for one-space and time $(1 + 1\text{D})$ fields on a lattice with nearest-neighbor (or *radius*-1) interactions is shown in Fig 3.4. We use $\mathtt{L}^-$ to denote the random variable for past lightcones with realizations ℓ^-; similarly, $\mathtt{L}^+$ those with realizations ℓ^+ for future lightcones.

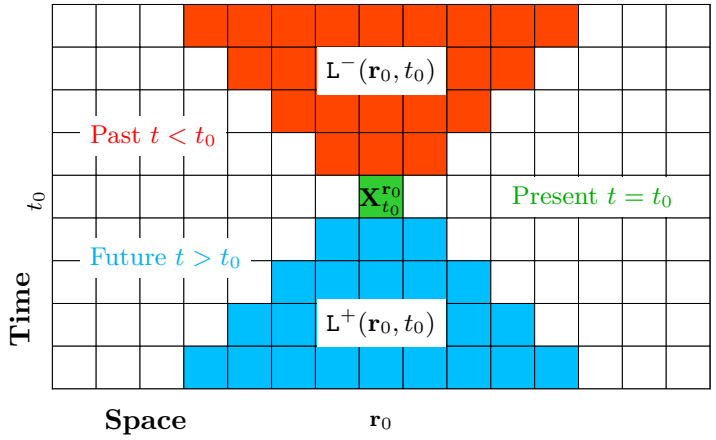


Figure 3.4. Lightcone random variable templates: Past lightcone $\mathtt{L}^-(r_0, t_0)$ and future lightcone $\mathtt{L}^+(r_0, t_0)$ for present spacetime point $\mathbf{X}_{t_0}^{r_0}$ in a $1 + 1$ D field with nearest-neighbor (or radius-1) interactions.

An aspect of past lightcones that highlights their importance and further justifies their use as local pasts is their relation to the governing local dynamics through *higher-order lookup tables*. The n^{th}-order lookup table ϕ^n maps the radius $r = n \cdot c$ neighborhood of a site to that site's value n time-steps in the future. Said another way, a spacetime point $\mathbf{x}_{t+n}^{\mathbf{r}}$ is completely determined by the radius $R = n \cdot c$ neighborhood n time-steps in the past according to $\phi^n\left(\eta^{n \cdot c}(\mathbf{x}_t^{\mathbf{r}})\right)$. To fill out the elements of ϕ^n, apply ϕ to all points of $\eta^{n \cdot c}$ to produce $\eta^{(n-1) \cdot c}$ and so on until $\eta^0 = x$ is reached. This is what we call the *lookup table cascade*, the elements of which are finite-depth past lightcones.

The choice of lightcone representations for both local pasts and futures is ultimately a weak-causality argument; influence and information propagate locally through a spacetime

site from its past lightcone to its future lightcone. By definition no spacetime site outside of the past lightcone of $\mathbf{x}_t^{\mathbf{r}}$ can causally influence $\mathbf{x}_t^{\mathbf{r}}$. Nor can $\mathbf{x}_t^{\mathbf{r}}$ causally influence any spacetime point outside of its future lightcone.

3.2.2.2 Local Causal Equivalence Relation

Having established the use of lightcones as local pasts and futures, generalizing the causal equivalence relation to spacetime is now straightforward. Two past lightcones are causally equivalent if they have the same distribution over future lightcones:

$$\ell_i^- \sim_\epsilon \ell_j^- \iff \Pr\left(\mathtt{L}^+|\ell_i^-\right) = \Pr\left(\mathtt{L}^+|\ell_j^-\right) . \tag{3.11}$$

This *local causal equivalence relation* over lightcones implements an intuitive notion of *optimal local prediction* [110]. At some point $\mathbf{x}_t^{\mathbf{r}}$ in spacetime, given knowledge of all past spacetime points that could possibly affect $\mathbf{x}_t^{\mathbf{r}}$—i.e., its past lightcone $\ell^-(\mathbf{r}, t)$—what might happen at all subsequent spacetime points that could be affected by $\mathbf{x}_t^{\mathbf{r}}$—i.e., its future lightcone $\ell^+(\mathbf{r}, t)$?

The equivalence relation induces a set $\boldsymbol{\Xi}$ of *local causal states* ξ. A functional version of the equivalence relation is helpful, as in the pure temporal setting, as it directly maps a given past lightcone ℓ^- to the equivalence class $[\ell^-]$ of which it is a member:

$$\begin{aligned}\epsilon(\ell^-) &= [\ell^-] \\ &= \{\ell^{-'} : \ell^- \sim_\epsilon \ell^{-'}\}\end{aligned}$$

or, even more directly, to the associated local causal state:

$$\epsilon(\ell^-) = \xi_{\ell^-} .$$

3.2.2.3 Properties

Closely tracking the standard development of temporal computational mechanics [169], a set of results for spatiotemporal processes parallels those of temporal causal states [110], which we summarize here.

- **Local Causal States Are Minimal Sufficient Statistics**

The starting point for this path of generalizing computational mechanics is again optimal (local) prediction. Formally, the notion of optimal prediction here means getting the correct conditional distribution $\Pr(\mathtt{L}^+|\ell^-)$. That is, the local causal states are minimal sufficient statistics of past lightcones for optimal prediction of future lightcones.

- **Local Causal States are Unique**

  In fact, the local causal states are the *unique* minimal sufficient statistics for this task of optimal local prediction.

- **Patch Composition**

  A *patch* is a connected subset of the spatial configuration at a particular time. These patches have past and future lightcones, similarly defined as all points in spacetime which can affect or be affected by that patch. The patch lightcones are just the unions of the single-site lightcones for all spatial sites in the patch. Optimal patch prediction then is getting the correct conditional distribution over future patch lightcones, given the past patch lightcone. Optimal patch predictors are the minimal sufficient statistics for optimal patch prediction, and are uniquely determined by the composition of all the local causal states within the patch.

- **Global Prediction**

  Consider the full spatial configuration as the largest possible patch for the system. Optimal prediction of this patch is then optimal prediction of the evolution of the full system. The minimal sufficient statistic for this global prediction are the configurational causal states, which are the states of the global ϵ-machine described above. Thus the global causal state at any time is uniquely determined by the collection of all local causal states in the spatial lattice at that time.

- **Markov Properties**

  The parents of the local causal state at $(\mathbf{r}, t)$ are the local causal states at all points at time $t-1$ which are inside the past lightcone $\ell^-(\mathbf{r}, t)$. That is, the local causal

states at all points inside the depth-1 past lightcone at $(\mathbf{r}, t)$. The local causal state at $(\mathbf{r}, t)$ is independent of the configuration in its past lightcone, $\ell^-(\mathbf{r}, t)$, given its parent states. Similarly, the local state at a point $(\mathbf{r}, t)$ is independent of the local causal states of points in its past lightcone, given its parents. This is known as *Markov shielding*.

It is conjectured that the spacetime field of local causal states is a Markov random field.

3.2.2.4 Causal state filtering

As in purely-temporal computational mechanics, the local causal equivalence relation Equation (3.11) induces a partition over the space of (infinite) past lightcones, with the local causal states being the equivalence classes. We will use the same notation for local causal states as was used for temporal causal states above, as there will be no overlap later: $\boldsymbol{\Xi}$ is the set of local causal states defined by the local causal equivalence partition, ξ denotes the random variable for a local causal state, and ξ for a specific causal state realization. The local ϵ-function $\epsilon(\ell^-)$ maps past lightcones to their local causal states $\epsilon : \ell^- \mapsto \xi$, based on their conditional distribution over future lightcones.

For spatiotemporal systems, a first step to discover emergent patterns applies the local ϵ-function to an entire spacetime field to produce an associated *local causal state field* $\mathcal{S} = \epsilon(\mathbf{x})$. Each point in the local causal state field is a local causal state $\mathcal{S}_t^{\mathbf{r}} = \epsilon\big(\ell^-(\mathbf{x}_t^{\mathbf{r}})\big) = \xi \in \boldsymbol{\Xi}$.

The central strategy here is to extract a spatiotemporal process' pattern and structure from the local causal state field. The transformation $\mathcal{S} = \epsilon(\mathbf{x})$ of a particular spacetime field realization $\mathbf{x}$ is known as *causal state filtering* and is implemented as follows. For every spacetime coordinate $(\mathbf{r}, t)$:

1. At $\mathbf{x}_t^{\mathbf{r}}$ determine its past lightcone $\mathtt{L}^-(\mathbf{r}, t) = \ell^-$;
2. Form its local predictive distribution $\Pr(\mathtt{L}^+|\ell^-)$;
3. Determine the unique local causal state $\xi \in \boldsymbol{\Xi}$ to which it leads; and
4. Label the local causal state field at point $(\mathbf{r}, t)$ with ξ: $\mathcal{S}_t^{\mathbf{r}} = \xi$.

Notice the values assigned to $\mathcal{S}$ in step 4 are simply the labels for the corresponding local causal states. Thus, the local causal state field is a *semantic field*, as its values are not measures of any quantity, but rather labels for equivalence classes of local dynamical behaviors as in the *measurement semantics* introduced in Ref. [197].

Though the local causal states are not a full machine presentations, they gain immense utility through causal filtering. Because observable spacetime fields are transformed locally, at each point in spacetime, the resulting local causal state field shares the same coordinate geometry of the observable field. This means pattern and structure of arbitrary shapes and sizes in the observable field can be defined through properties of the corresponding regions in the latent local causal state field. As we will see in Section 4.4.2, this includes algebraic properties that generalize exact symmetries in higher dimensions in an analogous manner described above, in Section 3.1.6, to machine presentations of temporal processes. The local causal states do not have a semi-group algebra generated by "word" concatenations, but the geometry of local causal state fields afforded by causal filtering provides relations among the local causal states that allow for algebras generated by spacetime shifts. As we will see in Chapters 4 and 5, the local causal states can capture quotient groups of generalized spacetime symmetries.

Causal filtering – through the shared coordinate geometry of $\mathbf{x}$ and $\mathcal{S}$ and the spacetime algebra over the local causal states– is the main theoretical contribution of this thesis. The applications to coherent structures in Chapters 6 and 7 are made possible using the shared coordinate geometry. Moreover, the spacetime algebra of local causal states provides the formal physical basis of coherent structures as locally broken (generalized) symmetries. Similarly, the analysis of CA patterns in Chapters 4 and 5, and their connection to spacetime invariant sets, is made possible only through causal filtering.

Chapter 4

Cellular Automata: Domain Patterns

We are ultimately interested in far-from-equilibrium pattern and structure in natural systems. In Chapter 1 we discussed the difficulties far-from-equilibrium systems pose for the traditional tools of physics. Emergent behaviors can not be deduced from governing equations nor isolated in carefully controlled experiments, making it seemingly impossible to test potential mechanistic hypotheses underlying these behaviors. Thus, in Chapter 3 we introduced the local causal states as a behavior-driven approach based on intrinsic computation, developed in Chapter 2, to circumvent these issues. There are three looming hurdles that prevent us from jumping straight in and applying the local causal states to natural systems. One is an imminently practical concern that we will return to next Chapter in Sections 7.1 and 7.2: the algorithmic and computational challenges of inferring the local causal states (what we call local causal state reconstruction) for natural systems. The other two issues are conceptual.

First, in Section 3.1.6 we argued for machine presentations as the appropriate mathematical formalism to capture pattern as generalized symmetry. However, machines can not be easily generalized to higher dimensions and as we saw in Section 3.2.2 the local causal state approach we will use, based on causal state filtering, does not utilize state transitions. Thus it remains to be seen whether the local causal states are useful for capturing pattern in higher dimensions in the way machines capture pattern in one

dimension.

Second, we again emphasize that there is no general theory, no ground-truth, for what actually constitutes an "organized structure" in far-from-equilibrium systems. Even if we do apply the local causal states to a natural system, how will we know whether they are telling us anything meaningful about the system? There are certainly many "know it when you see it" cases, like the Great Red Spot of Jupiter, but we would like to be more principled and systematic than that. Ideally, we would like to construct a *physical theory* of self-organization in far-from-equilibrium systems, based on the local causal states, that defines a ground-truth.

As a first step towards this lofty goal, we need to start in a simpler setting. Before we can use the local causal states to learn about natural systems, we need to learn more about the local causal states and their ability to capture pattern and structure in spacetime. Cellular automata will be our proving ground. They are fully discrete and thus more immediately amenable to computation-theoretic approaches like the local causal states. For the one-dimensional cellular automata that we will use, their spatial configurations can be studied using the tools and machine presentations of symbolic dynamics, thanks to the Curtis-Hedlund-Lyndon theorem (see Section 2.3).

Crutchfield and Hanson introduced a principled analysis of CA patterns and structures [103, 5, 193, 194, 116, 198, 4]. They defined *domain* patterns as dynamically invariant sets of spatially statistically stationary configurations with finite memory. This led to formal methods for proving that domains were spacetime shift-invariant and so dominant patterns for a given CA. Having identified these significant patterns, they created spatial transducers that decomposed a CA spacetime field into domains and nondomain structures, such as particles and particle interactions [199]. We refer to this analysis of CA structures as the *domain-particle-interaction decomposition* (DPID). The following extends DPID but, for the first time, uses local causal states to define domains and coherent structures. In this, domains are given by spacetime regions where the associated local causal states have time and space translation symmetries. Formally, this defines domain patterns as generalized spacetime symmetries. The local causal state extension is

crucial, as the dependence of DPID analysis on finite-state machines greatly complicates its application to natural systems.

Our exploration of the patterns and structures produced by cellular automata, as described by the local causal states, proceeds as follows. In Section 4.1 we review cellular automata. Then the topological reconstruction technique for inferring local causal states from discrete-valued spacetime fields is given in Section 4.2, and the automata-theoretic perspective of the global dynamics of CAs, the key ingredient of DPID, is given in Section 4.3. These two Sections establish the analysis tools that will be used in what follows. This sets us up to define the key notion of cellular automata domains in Section 4.4, which formalizes pattern in spatiotemporal systems as generalized spacetime symmetries. We then take a brief detour in Chapter 5 to relate domains to particular dynamical properties of cellular automata. Finally, in Chapter 6 we use domains to formalize coherent structures in cellular automata as localized deviations from generalized symmetries. Taken together, the results demonstrated in these Chapter show that the local causal states can capture meaningful spacetime pattern and structure in a principled manner.

4.1 Cellular Automata

A *cellular automaton* (CA) is a a fully-discrete spatially-extended dynamical system with a regular spatial lattice in d dimensions $\mathcal{L} = \mathbb{Z}^d$, consisting of local variables taking values from a discrete alphabet $\mathcal{A}$ and evolving in discrete time steps according to a *local dynamic* ϕ. Time evolution of the value at a site on a CA's lattice depends only on values at sites within a given *radius* R. The collection of all sites within radius R of a point $\mathbf{x}_t^{\mathbf{r}}$, including $\mathbf{x}_t^{\mathbf{r}}$ itself, is known as the point's *neighborhood* $\eta(\mathbf{x}_t^{\mathbf{r}})$:

$$\eta(\mathbf{x}_t^{\mathbf{r}}) = \{\mathbf{x}_t^{\mathbf{r}'} \ : \ ||\mathbf{r} - \mathbf{r}'|| \leq R; \ \mathbf{r}, \mathbf{r}' \in \mathcal{L}\} \ .$$

The neighborhood specification depends on the form of the lattice distance metric chosen. The two most common neighborhoods for regular lattice configurations are the Moore and von Neumann neighborhoods, defined by the Chebyshev and Manhattan distances on $\mathcal{L}$, respectively.

The *local* evolution of a spacetime point is given by:

$$\mathbf{x}_{t+1}^{\mathbf{r}} = \phi\big(\eta(\mathbf{x}_t^{\mathbf{r}})\big) \ ,$$

and the *global* evolution $\Phi : \mathcal{A}^{\mathcal{L}} \to \mathcal{A}^{\mathcal{L}}$ of the spatial field is given by:

$$\mathbf{x}_{t+1} = \Phi(\mathbf{x}_t) \ . \tag{4.1}$$

For example, this might apply ϕ in parallel, simultaneously to all neighborhoods on the lattice. Although, other local update schemes are encountered.

As noted, CAs are fully discrete dynamical systems. They evolve an initial spatial configuration $x_0 \in \mathcal{A}^{\mathcal{L}}$ according to Eq. (4.1)'s dynamic. This generates an *orbit* $\mathbf{x}_{0:t} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots \mathbf{x}_{t-1}\} \in \mathcal{A}^{\mathcal{L} \times \mathbb{Z}}$. Usefully, dynamical systems theory classifies a number of orbit types. Most basically, a *periodic orbit* repeats in time:

$$\begin{aligned} \mathbf{x}_{t+p} &= \Phi^p(\mathbf{x}_t) \\ &= \mathbf{x}_t \ , \end{aligned} \tag{4.2}$$

where p is its *period*—the smallest integer for which this holds. A *fixed point* has $p = 1$ and a *limit cycle* has finite $p > 1$. An *aperiodic orbit* has no finite p; a behavior that can occur only on infinite lattices.

Since CA states are spatial configurations an orbit $\mathbf{x}_{0:t}$ is a spacetime field. These orbits constitute the spatiotemporal processes of interest in this Chapter. Visualizing CA orbits as spacetime fields reveals the fascinating patterns and localized structures that CAs produce and how the patterns and structures evolve and interact over time.

4.1.1 Elementary Cellular Automata

The parameters $(\mathcal{A}, R)$ define a CA *class*. One simple but nontrivial class is that of the so-called *elementary* cellular automata (ECAs) [100] with a binary local alphabet $\mathcal{A} = \{0, 1\}$ and radius $R = 1$ local interactions $\eta(x_t^r) = x_t^{r-1} x_t^r x_t^{r+1}$. Due to their definitional simplicity and wide study, we mostly explore ECAs in our examples.

A local update rule ϕ is generally specified through a *lookup table*, which enumerates all possible neighborhood configurations η and their outputs $\phi(\eta)$. The lookup table for ECAs is given as:

η			$O_\eta = \phi(\eta)$
1	1	1	O_7
1	1	0	O_6
1	0	1	O_5
1	0	0	O_4
0	1	1	O_3
0	1	0	O_2
0	0	1	O_1
0	0	0	O_0

,

where each output $O_\eta = \phi(\eta) \in \mathcal{A}$ and the ηs are listed in lexicographical order. There are $2^8 = 256$ possible ECA lookup tables, as specified by the possible strings of output bits: $O_7 O_6 O_5 O_4 O_3 O_2 O_1 O_0$. A specific ECA lookup table is often referred to as an ECA *rule* with a *rule number* given as the binary integer $o_7 o_6 o_5 o_4 o_3 o_2 o_1 o_0 \in [0, 255]$. For example, ECA 172's lookup table has output bit string 10101100.

4.2 Topological Reconstruction

Being a behavior-driven technique, the local causal states are reconstructed from spacetime field realizations, rather than calculated from a system's equations of motion. As

the results presented in Sections 4.4 and 5 below rely on symmetries in local causal state fields, it is important to know whether one has a faithful local causal state reconstruction or not.

Using the local causal equivalence relation as given in Eq. (3.11) presents a challenge since the conditional distributions over lightcones must be inferred from finite realizations. To circumvent this, we instead use *topological reconstruction*. This replaces *probabilistic morphs* $\mathrm{morph}_{\mathcal{P}}(\ell_i^-) = \Pr\left(\mathtt{L}^+|\ell_i^-\right)$ with *topological morphs* $\mathrm{morph}_{\mathcal{T}}(\ell_i^-) = \{\text{all } \ell_j^+ \text{ occurring with } \ell_i^-\}$, which are the supports of the probabilistic morphs. Thus, two past lightcones are topologically (causally) equivalent if they lead to the same *set* of future lightcones. Contrast this with being probabilistically (causally) equivalent, if they have the same *distribution* over future lightcones.

Topological reconstruction is particularly convenient for fully-discrete systems such as CAs, since the topological morphs for finite-depth lightcones can be *exactly* reconstructed and the condition for topological equivalence is *exact*. (That is, are the topological morph sets the same or not?) Moreover, the number of unique past lightcone-future lightcone pairs seen in spacetime field data is monotone increasing, providing a measure of convergence for identifying topological morphs. In short, finite-depth approximations to the topological local causal states can be exactly reconstructed with confidence.

For concreteness, the topological reconstruction of local causal states in what follows uses past and future lightcone depths of 3 for explicit symmetry domains and past lightcone depth 8 and future lightcone depth 3 for hidden symmetry domains. (Domain types are defined in Section 4.4.3.)

4.3 Automata-Theoretic CA Evolution

Rather than study how a CA evolves individual configurations, it is particularly informative to investigate how CAs evolve sets of configurations [5, 199]. This allows for discovery of structures in the state space of a CA induced by Φ.

For cellular automata in one spatial dimension, configurations $x \in \mathcal{A}^{\mathbb{Z}}$ are strings over the alphabet $\mathcal{A}$. Sets of strings recognized by finite state machines are called *regular*

languages. Any regular language L has a unique minimal finite-state machine $M(L)$ that recognizes or generates it [84]. These automata are useful since they give a finite representation of a typically infinite set of regular-language configurations.

As described in more detail in Section 2.3, not all regular languages are appropriate as sets of spatial configurations for cellular automata; we consider only sets that are subword closed and prolongable in the sense that every word in the language can be extended to the left and the right to obtain a longer word in the language. In automata theory these are known as *factorial, prolongable regular languages*. In symbolic dynamics [78] these are the languages of *sofic shifts*—closed, shift-invariant subsets of $\mathcal{A}^{\mathbb{Z}}$ that are described by finite-state machines. The language L of a sofic shift $\mathcal{X} \subseteq \mathcal{A}^{\mathbb{Z}}$, a *sofic language*, is the collection of all words that occur in points $x \in \mathcal{X}$. (See Section 2.3 for details of sofic languages; sometimes called *process languages* in the computational mechanics literature.) Every state of the machine $M(L)$ for a sofic language L is both a start and end state.

For the remainder of our development any language L always refers to a sofic language, and the machine $M(L)$ refers to the unique minimal deterministic finite-state machine presentation of that language.

To explore how a CA evolves languages we establish a dynamic that evolves machines. This is accomplished via finite-state transducers. Transducers are a particular type of input-output machine that maps strings to strings [200]. This is exactly what a (one-dimensional) CA's global dynamic Φ does [101]. As a mapping from a configuration x_t at time t to one x_{t+1} at time $t+1$, Φ is also a map on a configuration set L_t from one time to the next L_{t+1}:

$$L_{t+1} = \Phi(L_t) \ . \tag{4.3}$$

The global dynamic Φ can be represented as a finite-state transducer T_Φ that evolves a set of configurations represented by a finite-state machine. This is the *finite machine evolution* (FME) operator [5]. Its operation composes the CA transducer T_Φ and finite-state machine $M(L_t)$ to get the machine $M_{t+1} = M(L_{t+1})$ describing the set L_{t+1} of

spatial configurations at the next time step:

$$M_{t+1} = \min\left(\mathrm{T}_\Phi \circ M(L_t)\right) \ . \tag{4.4}$$

Here, $\min(M)$ is the automata-theoretic procedure that minimizes the number of states in machine M. While not entirely necessary for language evolution, the minimization step is helpful when monitoring the complexity of L_t. The net result is that Eq. (4.4) is the automata-theoretic version of Eq. (4.3)'s set evolution dynamic. Analyzing how the FME operator evolves configuration sets of different kinds is a key tool in understanding CA emergent patterns.

4.4 CA Domains

Modern physics evolved to use group theory to formalize the concept of symmetry [201]. The successes in doing so are legion in twentieth-century fundamental physics. When applied to emergent patterns, though, group-theoretic descriptions formally describe only their exact symmetries. This is too restrictive for more general notions—naturally occurring patterns and structures that are an amalgam of strict symmetry and randomness. Thus, one appeals to semi-group theory [202, 87] to describe partial symmetries. This use of semigroup algebra is fundamental to automata as developed in early computation theory [203, 204]. In this, different classes of automata or "machines" formalize the concept of pattern; see Section 3.1.6. Through the connection with semi-group theory, pattern captured by machines can be seen as a system's generalized symmetries. The variety of computational model classes [84] then becomes an inspiration for understanding emergent natural patterns [203]. Alluding to condensed matter physics, we will refer to regions with this generalized symmetry as *domains*. They are the background organizations above which *coherent structures* are defined; see Chapter 6.

Below we give two distinct domain definitions; one in terms of dynamically-invariant sets, the other in terms of local causal state symmetries. For both, a formal notion of domain is used to discover and describe these and other emergent spacetime structures that form in CA spacetime fields.

4.4.0.1 Structure From Broken Symmetries

Structure is often described as arising from *broken symmetries* [133, 33, 10, 205, 206, 29, 30, 207]. Though key to our development, broken symmetry is a more broadly unifying mechanism in physics. Care, therefore, is required to precisely distinguish the nature of broken symmetries we are interested in. Specifically, our formalism seeks to capture coherent structures as *temporally-persistent, spatially-localized broken symmetries*.

Drawing contrasts will help delineate this notion of coherent structure from others associated with broken symmetries. Equilibrium phase transitions also arise via broken symmetries. There, the degree of breaking is quantified by an *order parameter* that vanishes in the symmetric state. A transition occurs when the symmetry is broken and the order parameter is no longer zero [206].

This, however, does not imply the existence of coherent structures. When the order parameter is global and not a function of space, symmetry is broken globally, not locally. And so, the resulting state may still possess additional global symmetries. For example, when liquids freeze into crystalline solids, continuous translational symmetry is replaced by a discrete translational symmetry of the crystal lattice—a global symmetry.

Similarly, the primary bifurcation exhibited in nonequilibrium phase transitions occurs when the translational invariance of an initial homogeneous field breaks [29, 30]. It is often the case, though, as in equilibrium, that this is a continuous-to-discrete symmetry breaking, since the cellular patterns that emerge have a discrete lattice symmetry. To be concrete, this occurs in the conduction-convection transition in Rayleigh-Bénard flow. The convection state just above the critical Rayleigh number consists of convection cells patterned in a lattice [17, 20]. In the language used here, the above patterns arise as a change of domain structure, not the formation of coherent structures. Coherent structures, such as topological defects [29, 208], form at higher Rayleigh numbers when the discrete cellular symmetries are locally broken.

Describing domains, their use as a baseline for coherent structures, and how their own structural alterations arise from global symmetry breaking transitions delineates what our coherent structures are not. To make positive headway, we move on to a direct

formulation, starting with how they first appeared in the original DPID and then turning to express them via local causal states. After establishing domain formalism, and taking a detour connecting domain behaviors to CA subdynamics in Section 5, we will return to the task of defining coherent structures and providing examples of CA structures in Section 6.

4.4.1 DPID Patterns: Spacetime Invariant Sets

Domains of one-dimensional cellular automata were defined in DPID pattern analysis [5, 193, 194, 116, 4] as sets of spatially (and statistically) homogeneous configurations that are invariant under a CA dynamic Φ.

Presently, we find it useful to restate and reinterpret these results using symbolic dynamics [78]. Recall from Section 2.3 that a *shift space*, or simply a *shift*, $\mathcal{X} \subseteq \mathcal{A}^{\mathbb{Z}}$ is a compact, shift-invariant subset of the full-$\mathcal{A}$ shift $\mathcal{A}^{\mathbb{Z}}$. A *point* $x = \cdots x_{-2}x_{-1}x_0x_1\cdots$ in a shift space is an indexed bi-infinite string of symbols in $\mathcal{A}$ and the *shift operator* σ increments the indices of points by one; if $y = \sigma(x)$ for $x \in \mathcal{X}$, then $y_i = x_{i+1}$ and by definition $y \in \mathcal{X}$. As the name suggests, a *sliding block code* $\Phi : \mathcal{X} \to \mathcal{Y}$ maps points from one shift space to another using a sliding-window function ϕ: $y_i = \phi(x_{i-m:i+n})$, where $x \in \mathcal{X}$, $y \in \mathcal{Y}$. We are particularly interested in surjective codes, also known as *factor maps*. The notational overlap with CA dynamics is intentional: CAs are uniform sliding block codes ($m = n = R$) that commute with σ [170].

We now give the spacetime invariant set definition of CA domains using this symbolic dynamics formalism. We consider sets of CA configurations given as shift spaces, and these are invariant sets of the CA if the CA dynamic Φ is a factor map from that shift space to itself.

Definition 1. Consider a CA Φ and a set $\Lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_{\widehat{p}}\}$ of shift spaces $\Lambda_i \subseteq \mathcal{A}^{\mathbb{Z}}$. Together, this set of shifts is a *domain* of Φ if the following hold:

1. *Spatial invariance*: Each $\Lambda_i \in \Lambda$ is an *irreducible sofic shift*. That is, the set of strings in each Λ_i is generated by a strongly-connected finite-state machine $M(\Lambda_i)$.
2. *Temporal invariance*: $\Phi : \Lambda_i \to \Lambda_{i+1 (\mathrm{mod}\, \widehat{p})}$ is a factor map from Λ_i to $\Lambda_{i+1 (\mathrm{mod}\, \widehat{p})}$.

Thus, $\Phi^{\widehat{p}} : \Lambda_i \to \Lambda_i$ is a factor from Λ_i to itself, for all $\Lambda_i \in \Lambda$.

Each distinct Λ_i is a *temporal phase* of the domain and the number $\widehat{p}$ of temporal phases is the *recurrence time* of the domain—the minimum number of time steps required for Φ to map a temporal phase back to itself. The size s of the minimal cycle in $M(\Lambda_i)$ is the *spatial period* of Λ_i. For all known examples, the spatial period of each Λ_i in a given Λ is the same; thus, making s the spatial period of the domain.

An ambiguity arises here between Λ's recurrence time $\widehat{p}$ and its *temporal period* p. For a certain class of CA domain (those with explicit symmetries, see Section 4.4.3) , the domain states $x \in \Lambda$ are periodic orbits of the CA, with orbit period equal to the domain period: $x = \Phi^p(x)$. It is less clear how to define the temporal period for domains in general using this formalism. The temporal period appears to be related more to the *spacetime shift spaces* of domain orbits that results from evolving domain spatial shift spaces under Φ. The spacetime shift spaces of hidden symmetry domains are more complicated objects than those of explicit symmetry domains. Notably, at present, beyond particular cases it is not known how to generally relate the domain spatial shifts to their resulting spacetime shifts.

Given a CA Φ, there are no general analytic solutions to $\Phi^{\widehat{p}}(\Lambda_i) = \Lambda_i$. However, given a candidate shift $\mathcal{X}$ it is computationally straightforward to find the factor $\mathcal{Y}$ of $\mathcal{X}$ under Φ using the FME operator. That is, we want to restrict the function-domain of Φ to $\mathcal{X}$ and then find the set $\mathcal{Y}$ of images $y = \Phi(x)$ for all pre-images $x \in \mathcal{X}$ so that Φ is a surjective map from $\mathcal{X}$ to $\mathcal{Y}$. This is exactly what the FME operator does. If $\mathcal{X}$ is an irreducible sofic shift and $\mathcal{X} = \Phi^{\widehat{p}}(\mathcal{X})$ for some $\widehat{p}$, then $\mathcal{X}$ is a domain temporal phase of Φ. Since the FME evolves machines, we technically look for $M(\mathcal{X}) = \Phi^{\widehat{p}}\big(M\left(\mathcal{X}\right)\big)$, where equality here is given by machine isomorphism [1]. If $\widehat{p} > 1$, the other temporal phases can be found using $\Lambda_{i+1(\mathrm{mod}\ \widehat{p})} = \Phi(\Lambda_i)$.

[1] Crutchfield and McTague implemented an efficient, but exhaustive search algorithm to solve the invariant equation using the enumerated library of machines of Ref. [209]. Reference [210] analyzed ECA 22 using the approach.

4.4.1.1 DPID Transducer Filters

Once a CA's domains $\Lambda^0, \Lambda^1, \ldots$ are discovered, they can be used to create a *domain transducer* τ that identifies which of configuration x's sites are in which domain and which are not in any domain [199]. For a given 1+1 dimension spacetime field $\mathbf{x}$, each of its spatial configurations $x = \mathbf{x}_t$ are scanned by the transducer, with output $T_t = \tau(x)$. Although the transducer maps strings to strings, the full spacetime field can be filtered with τ by collecting the outputs of each configuration in time order to produce the *domain transducer filter field* of $\mathbf{x}$: $T = \tau(\mathbf{x})$.

Sites $\mathbf{x}_t^r$ "participating" in domain Λ^i are labeled i in the transducer field. That is:

$$T_t^r = \tau(\mathbf{x}_t)^r = i \ .$$

Other sites are similarly labeled by the particular way in which they deviate from domain(s). One or several sites, for example, can indicate transitions from one domain temporal phase or domain type to another. If that happens in a way that is localized across space, one refers to those sites as participating in a CA *particle*. *Particle interactions* can also be similarly identified. Reference [5] describes how this is carried out.

In general, a stack automaton is needed to perform this domain-filtering task, but it may be efficiently approximated using a finite-state transducer [199].

This filter allows us not only to formally define CA domains, the transducer allows for site-by-site identification of domain regions and thus also sites participating in non-domain patterns. In this way and in a principled manner, one finds localized deviations from domains. In Section 6 we will use these as candidate coherent structures.

Originally, this was called *cellular automata computational mechanics*. Since then, other approaches to spatiotemporal computational mechanics developed, such as local causal states. We now refer to the above as DPID pattern analysis.

4.4.2 Local Causal State Symmetries

DPID pattern analysis formulates domains directly in terms of how a system's dynamic evolves spatial configurations. That is, domains are sets of structurally homogeneous spatial configurations that are invariant under Φ. While this is appealing in many ways,

the FME can not be applied to more complex spatiotemporal systems, such as turbulent fluid flows. But, the local causal states can.

Just as the causal states help discover structure from a temporal process, we would like to use the local causal states to discover pattern and structure directly from spacetime fields. To do so, we start with a precise formulation of domains in terms of local causal states [115, 114]. Since local causal states apply in arbitrary spatial dimensions, the following addresses general d-dimensional cellular automata. In this, index $n \in \{1, 2, \ldots, d\}$ identifies a particular spatial coordinate.

A simple but useful lesson from DPID is that domains are special (invariant) subsets of CA configurations. More formally, they are subshifts of the full shift, which is the set of all possible configurations. Since they are deterministically generated, a CA's spacetime field is entirely specified by the rule ϕ, the initial condition $\mathbf{x}_0$, and the boundary conditions. Here, in analyzing a CA's behavior, ϕ is fixed and we only consider periodic boundary conditions. This means for a given CA rule, the spacetime field is entirely determined by $\mathbf{x}_0$. If it belongs to a domain—$\mathbf{x}_0 \in \Lambda^i$—all subsequent configurations of the spacetime field will, by definition, also be in the domain—$\mathbf{x}_t = \Phi^t(\mathbf{x}_0) \in \Lambda^i$. In this sense a domain $\Lambda \subseteq \mathcal{A}^{\mathcal{L}}$ is a subset of a CA's allowed behaviors: $\Lambda \subseteq \Phi^t(\mathcal{A}^{\mathcal{L}})$, $t = 1, 2, 3, \ldots$. More formally, the domain spacetime fields that result from evolution of domain configurations form a spacetime subshift of the spacetime full shift, which is the set of all possible spacetime fields that CA can produce.

Lacking prior knowledge, if one wants to use local causal states to discover a CA's patterns, their reconstruction should be performed on *all* of a CA's spacetime behavior $\Phi^t(\mathcal{A}^{\mathcal{L}})$. This gives a complete sampling of spacetime field realizations and so adequate statistics for good local causal state inference. Doing so leaves one with the full set of local causal states associated with a CA. Since domains are a subset of a CA's behavior, they must be described by some special subset of the associated local causal states. What are the defining properties of this subset of states which define them as one or another domain?

The answer is quite natural. The defining properties of local causal states associated

with domains are expressed in terms of symmetries. For one-dimensional CAs these are time and space translation symmetries. In general, alternative symmetries may be considered as well, such as rotations, as appropriate to other settings. Such symmetries are directly accessed through causal filtering.

Consider a spatiotemporal process $\mathbf{X}$, the set $\mathbf{\Xi}$ of local causal states induced by the local causal equivalence relation $\sim_\epsilon$ over $\mathbf{X}$, and the local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ over the spacetime field realization $\mathbf{x}$. Let σ_p denote the *temporal shift operator* that shifts a spacetime field $\mathbf{x}$ p steps along the time dimension. This translates a point $\mathbf{x}_t^{\mathbf{r}}$ in the spacetime field as: $\sigma_p(\mathbf{x})_t^{\mathbf{r}} = \mathbf{x}_{t+p}^{\mathbf{r}}$. Similarly, let σ^{s_n} denote the *spatial shift operator* that shifts a spacetime field $\mathbf{x}$ by s_n steps along the n^{th} spatial dimension. This translates a spacetime point $\mathbf{x}_t^{\mathbf{r}}$ as: $\sigma^{s_n}(\mathbf{x})_t^r = \mathbf{x}_t^{r'}$, where $r'_n = r_n + s_n$.

Definition 2. A *pure domain field* $\mathbf{x}_\Lambda$ is a spacetime field such that σ_p and the set of spatial shifts $\{\sigma^{s_n}\}$ applied to $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$ form a symmetry group [201]. The generators of the symmetry group consist of the following translations:

1. *Temporal invariance*: For some finite time shift p the domain causal state field is invariant:

$$\sigma_p(\mathcal{S}_\Lambda) = \mathcal{S}_\Lambda \ , \tag{4.5}$$

   and:

2. *Spatial invariance*: For some finite spatial shift s_n in each spatial coordinate n the domain causal state field is invariant:

$$\sigma^{s_n}(\mathcal{S}_\Lambda) = \mathcal{S}_\Lambda \ . \tag{4.6}$$

The symmetry group is completed by including these translations' inverses, compositions, and the identity null-shift $\sigma_{\mathbf{0}}(\mathbf{x})_t^r = \mathbf{x}_t^{\mathbf{r}}$. The set $\mathbf{\Xi}_\Lambda \subseteq \mathbf{\Xi}$ is Λ's *domain local causal states*: $\mathbf{\Xi}_\Lambda = \{(\mathcal{S}_\Lambda)_t^{\mathbf{r}} : \ t \in \mathbb{Z}, \mathbf{r} \in \mathcal{L}\}$.

A *domain* Λ of $\mathbf{X}$ is the set of all spacetime field realizations $\mathbf{x}_\Lambda$ such that $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$ contains only local causal states from $\mathbf{\Xi}_\Lambda$ and has the defining spacetime symmetries. The set Λ is a *spacetime shift space*—a closed, spacetime-shift invariant subset of $\mathcal{A}^{\mathbb{Z}\times\mathcal{L}}$.

Note that the spacetime shift space property is a necessity, as two distinct domains may have the same local causal state symmetries. As an example, below in Section 5.1 we will see that particular CA rules, called additive CAs, produce only domain behavior. And, their domains all have the same local causal state symmetries. However, they are still distinct domains as the collective set of all their spacetime fields is not closed under spacetime shifts. (The easiest way to see this is to realize that elements of the CA lookup table, together with their updated site values, form spacetime motifs, as described in Section 2.4.3, and thus different CA rules produce distinct shifts of finite type with different lists $\mathcal{F}$ of forbidden motifs.) Therefore, local causal state symmetries are *characteristic* of domain spacetime shift spaces, but do not fully *specify* these shift spaces.

The smallest integer p for which the temporal invariance of Eq. (4.5) is satisfied is Λ's *temporal period*. The smallest s for which Eq. (4.6)'s spatial invariance holds is Λ's *spatial period*.

The domain's *recurrence time* $\widehat{p}$ is the smallest time shift that brings $\mathcal{S}_\Lambda$ back to itself when also combined with finite spatial shifts. That is, $\sigma^j \sigma_{\widehat{p}}(\mathcal{S}_\Lambda) = \mathcal{S}_\Lambda$ for some finite space shift σ^j. If $\widehat{p} > 1$, this implies there are distinct tilings of the spatial lattice at intervening times between recurrences. The distinct tilings then correspond to Λ's *temporal phases*: $\Lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_{\widehat{p}}\}$. For systems with a single spatial dimension, like the CAs we consider here, the spatial symmetry tilings are simply $(\mathcal{S}_\Lambda)_t = \cdots w \cdot w \cdot w \cdots = w^\infty$, where $w = (\mathcal{S}_\Lambda)_t^{i:i+s}$. Each domain phase Λ_i corresponds to a unique tiling w_i.

Consider a contiguous region $\mathcal{R}_\Lambda \subset \mathcal{L} \times \mathbb{Z}$ in $\mathcal{S} = \epsilon(\mathbf{x})$ for spacetime field $\mathbf{x}$ for which all points $\mathcal{S}_t^{\mathbf{r}}$ in the region are domain local causal states: $\mathcal{S}_t^{\mathbf{r}} \in \mathbf{\Xi}_\Lambda$, $(r, t) \in \mathcal{R}_\Lambda$. The space and time shift operators over the region obey the symmetry groups of pure domain fields. Such regions, over both $\mathbf{x}$ and $\mathcal{S} = \epsilon(\mathbf{x})$, are *domain regions*.

While Definitions 1 and 2 are independent definitions of CA domain that even differ in what mathematical objects are identified as domains, we have found a strong empirical correspondence between these two definitions, as demonstrated throughout the rest of this chapter. We formalize this correspondence as follows.

Consider a CA Φ and two domain sets Λ^1 and Λ^2. Λ^1 is a set of spatial shifts that satisfy Definition 1 for Φ: each $\Lambda^1_i \in \Lambda^1$ is an irreducible sofic shift such that $\Phi^{\widehat{p}}(\Lambda^1_i) = \Lambda^1_i$. Since Λ^1 is a set of shift spaces that are themselves sets of spatial configurations x_{Λ^1}, we can simply think of Λ^1 as the set of all configurations in the collection $\{x_{\Lambda^1} \,:\, x_{\Lambda^1} \in \Lambda^1_i \in \Lambda^1\}$ of invariant spatial shifts. Λ^2 is a spacetime shift space that satisfies Definition 2 for Φ: for each spacetime field orbit $\mathbf{x}_{\Lambda^2} \in \Lambda^2$ the local causal-state field $\mathcal{S}_{\Lambda^2} = \epsilon(\mathbf{x}_{\Lambda^2})$ is comprised of states from $\mathbf{\Xi}_{\Lambda^2}$ and is time- and space- translation invariant: $\sigma_p(\mathcal{S}_{\Lambda^2}) = \mathcal{S}_{\Lambda^2}$ and $\sigma^s(\mathcal{S}_{\Lambda^2}) = \mathcal{S}_{\Lambda^2}$. We conjecture the following bijective relationship between Λ^1 and Λ^2.

Conjecture 1. For each configuration $x_{\Lambda^1} \in \Lambda^1$, its orbit under Φ is in Λ^2 and each spacetime field $\mathbf{x}_{\Lambda^2} \in \Lambda^2$ is the orbit, under Φ, of a configuration in Λ^1. That is, first, for all $x_{\Lambda^1} \in \Lambda^1$, there is $\mathbf{x}_{\Lambda^2} \in \Lambda^2$ such that $\mathbf{x}_{\Lambda^2} = \{x_{\Lambda^1}, \Phi(x_{\Lambda^1}), \Phi^2(x_{\Lambda^1}), \Phi^3(x_{\Lambda^1}), \ldots\}$. And, second, for all $\mathbf{x}_{\Lambda^2} \in \Lambda^2$ there is $x_{\Lambda^1} \in \Lambda^1$ such that $\mathbf{x}_{\Lambda^2} = \{x_{\Lambda^1}, \Phi(x_{\Lambda^1}), \Phi^2(x_{\Lambda^1}), \Phi^3(x_{\Lambda^1}), \ldots\}$.

Taking this conjecture to be true, the following uses Λ and "domain" to refer both to sets of invariant spatial shifts (Λ^1) and the set of orbits of those spatial shifts (Λ^2). We will see this relationship between domain definitions in all examples given in this Section, with two particularly detailed examples given in Sections 6.2.1 and 6.3.1.

4.4.3 Domain Classification: Explicit vs Hidden Symmetry

CA domains fall into one of two classes: explicit symmetry or hidden symmetry. In the local causal state formulation, a domain Λ has *explicit symmetry* if the time and space shift operators σ_p and σ^s—that generate the domain symmetry group over $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$—also generate that same symmetry group over $\mathbf{x}_\Lambda$. That is, $\sigma_p(\mathbf{x}_\Lambda) = \mathbf{x}_\Lambda$ and $\sigma^s(\mathbf{x}_\Lambda) = \mathbf{x}_\Lambda$. From this, we see the following.

Lemma 1. Every explicit symmetry domain configuration $x_\Lambda \in \Lambda$ of a CA Φ generates a periodic orbit of that CA, with the orbit period equal to the domain temporal period.

Proof. This follows since time shifts of the spacetime field are essentially equivalent to applying the CA dynamic Φ: $\mathbf{x}_{t+p} = \sigma_p(\mathbf{x})_t$ and $x_{t+p} = \Phi^p(x_t)$. Thus, if x_Λ is any spatial

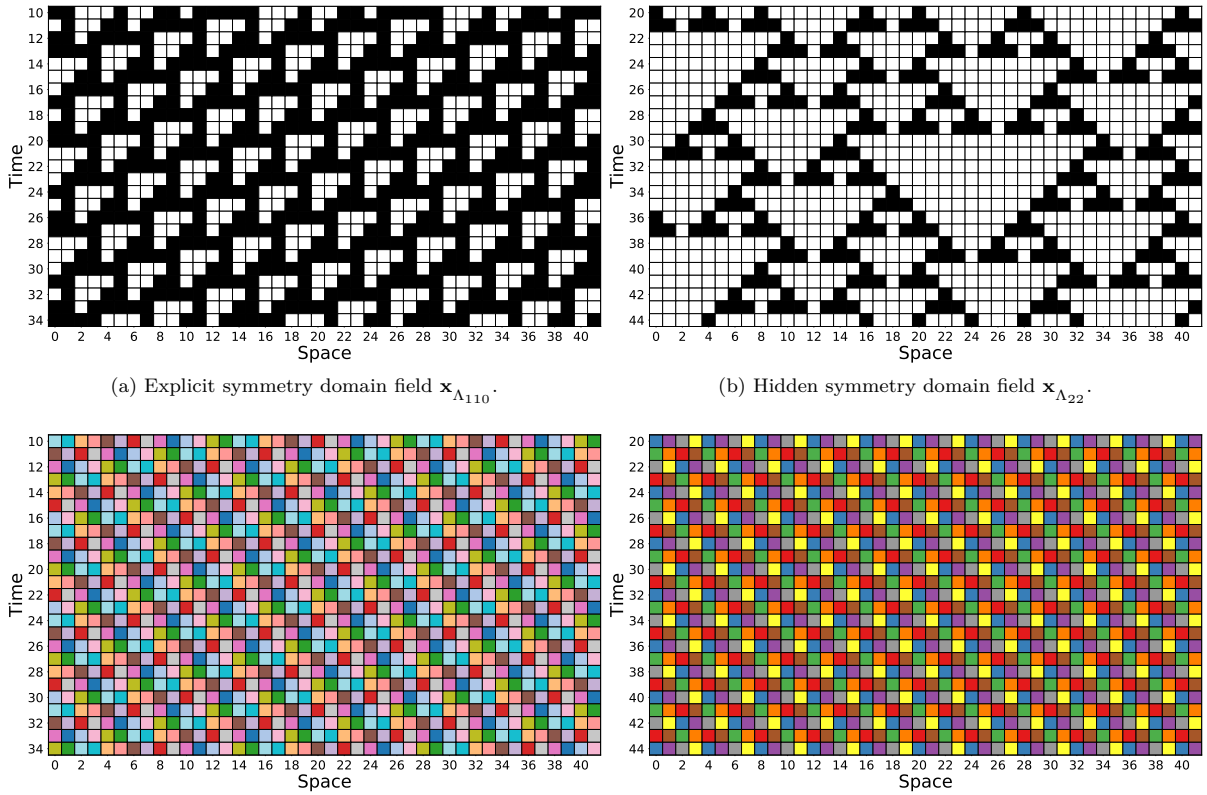
configuration of a domain spacetime field—$x_\Lambda = (\mathbf{x}_\Lambda)_t$, for any t—then $\Phi^p(x_\Lambda) = x_\Lambda$ if and only if $\sigma_p(\mathbf{x}_\Lambda) = \mathbf{x}_\Lambda$.

A *hidden symmetry domain* is one for which the time and space shift operators, which generate the domain symmetry group over $\mathcal{S}_\Lambda$, do not generate a symmetry group over $\mathbf{x}_\Lambda$: $\sigma_p(\mathbf{x}_\Lambda) \neq \mathbf{x}_\Lambda$ or $\sigma^s(\mathbf{x}_\Lambda) \neq \mathbf{x}_\Lambda$ or both.

Domain classification for the invariant-set formulation is similar. A domain Λ has explicit symmetry if its spatial configurations x_Λ are not just shift invariant but also translation invariant, so that $y = \sigma^s(x_\Lambda) = x_\Lambda$, for all $x_\Lambda \in \Lambda$. If Λ has hidden symmetry it is still shift invariant: $y = \sigma^s(x_\Lambda) \in \Lambda$, but it is not translation invariant: $y = \sigma^s(x_\Lambda) \neq x_\Lambda$. From the machine presentation, Λ is a hidden symmetry domain if $M(\Lambda)$ has any local branching in transitions between states; see Section 3.1.6. That is, if there is any state in $M(\Lambda)$ such that there is more than one transition leaving that state, then Λ has hidden symmetry.

Notably, hidden symmetry domains are associated with a level of stochasticity in their observable spacetime fields. We occasionally refer to these as *stochastic domains*. The algebraic formulation just given highlights the notion of hidden symmetry domains as patterns that generalize the exact spacetime symmetries of explicit symmetry domains.

Example domains from each category are shown in Figure 4.1. ECA 110 is given as the explicit symmetry example; a sample spacetime field $\mathbf{x}_{\Lambda_{110}}$ of its domain is shown in Figure 4.1(a). The associated local causal state field $\mathcal{S}_{\Lambda_{110}}$ is shown in Figure 4.1(c). Each unique color corresponds to a unique local causal state. The local causal state field clearly displays the domain's translation symmetries. ECA 110's domain has spatial period $s = 14$ and temporal period $p = 7$. These are gleaned by direct inspection of the spacetime diagram. Pick any color in $\mathcal{S}_{\Lambda_{110}}$ and one must go through 13 other colors moving through space to return to the original color and, likewise, 6 other colors in time before returning. One can also see that at every time step $\mathcal{S}_{\Lambda_{110}}$ has a single spatial tiling w of the 14 states. Thus, the recurrence time is $\widehat{p} = 1$. Finally, notice from Figure 4.1(a) that spatial configurations of $\mathbf{x}_{\Lambda_{110}}$ are periodic orbits of Φ_{110}, with orbit period equal to the domain period, $p = 7$.



(a) Explicit symmetry domain field $\mathbf{x}_{\Lambda_{110}}$.

(b) Hidden symmetry domain field $\mathbf{x}_{\Lambda_{22}}$.

(c) Explicit symmetry state field $\mathcal{S}_{\Lambda_{110}} = \epsilon(\mathbf{x}_{\Lambda_{110}})$.

(d) Hidden symmetry state field $\mathcal{S}_{\Lambda_{22}} = \epsilon(\mathbf{x}_{\Lambda_{22}})$.

Figure 4.1. Pure domain spacetime fields for explicit symmetry and hidden symmetry domains shown in (a) and (b) for ECA 110 and ECA 22, respectively. Associated local causal state fields fully display these symmetries in (c) and (d), with each unique color corresponding to a unique local causal state. For ECA 110, lightcone horizons $h^- = h^+ = 3$ were used and for rule 22 $h^- = 10$ and $h^+ = 4$.

For a prototype hidden symmetry domain, ECA 22 is used. Crutchfield and McTague used DPID analysis to discover this ECA's domain in an unpublished work [210] that we used here to produce the domain spacetime field $\mathbf{x}_{\Lambda_{22}}$ shown in Figure 4.1(b). The associated causal state field $\mathcal{S}_{\Lambda_{22}}$ is shown in Figure 4.1(d). Unlike ECA 110's domain, it is not clear from $\mathbf{x}_{\Lambda_{22}}$ what the domain symmetries are. It is not even clear there *are* symmetries present from the observable spacetime field. However, the causal state field $\mathcal{S}_{\Lambda_{22}}$ is immediately revealing. Domain translation symmetries are clear. The domain is period 4 in both space and time: $p = s = 4$. There are eight unique local causal states in $\mathcal{S}_{\Lambda_{22}}$ and, as the spatial period is 4, the eight states come in two distinct spatial tilings w_1 and w_2, each consisting of 4 states. And so, the recurrence time for ECA 22 is $\widehat{p} = 2$. Shortly, we examine hidden symmetries in more detail to illustrate how the local causal

states lend a new semantics that illuminates stochastic symmetries.

Figure 4.1 shows how the local causal states, through causal filtering, capture patterns in spacetime as generalized symmetries. For observable spacetime fields with space and time translation symmetries, i.e. explicit symmetry domains like Λ_{110}, the local causal states capture the quotient groups of these spacetime symmetries. An observable field that is not space or time translation invariant may still posses a generalized symmetry, i.e. a pattern, if the corresponding local causal state field has space and time translation symmetries. Because the local causal states are still translation invariant, they still capture the quotient groups and internal organization of these generalized symmetries.

Chapter 5

Cellular Automata: Domain Subdynamics

Early systematic studies of cellular automata focused largely on phenomenology [211, 212, 213, 214] and algebraic properties of *additive* cellular automata [215, 216]. Analysis was possible due to the linear superposition exhibited by additive CAs. One challenge was to extend the analytic techniques appropriate for linear CAs to nonlinear CAs, which produce much richer and more physically-relevant behaviors.

The first identification of what we now call a CA domain came from the observation that the *nonlinear* rule 18, when evolving certain configurations, emulates the *linear* rule 90 [217]. As described in more detail below, we now know that the set of configurations over which rule 18 is equivalent to rule 90 is in fact the *domain* of rule 18. Thus, in the case of the nonlinear rule 18, its domain represents a subset of behavior that is actually linear.

This approach was elaborated upon by Refs. [216, 100] and expanded upon by Ref. [218], which proved nonlinear rules 126 and 146 can be reduced to the linear rule 90. The latter also showed, under certain conditions, that more general orbits of the nonlinear rules can be mapped to orbits of rule 90, then mapped back; further extending the use of linear analytics on nonlinear rules. Reference [219] considered additional algebraic structures to generalize additive CAs into a larger set of *quasilinear* CAs that do not obey superposition, but whose spacetime evolutions can still be predicted in less time then general nonlinear

CAs. More recently, Ref. [220] defined a special set of *quasiadditive* spatial configurations for the nonlinear rule 22 that emulate rule 146 at all even time steps. (Ref. [218] had already established that rule 146 emulates linear rule 90.)

In this Chapter we utilize the tools just developed in Chapter 4 to further investigate connections between domain behaviors of CAs and the local update rules that produce these behaviors. We will demonstrate that all of the results just mentioned of linear behaviors embedded in nonlinear CAs are examples of domains. In particular, we will show that all of the above cases actually correspond to lookup table vacancies that result from invariant (proper) subshifts of the linear ECA rule 90. In addition, we will show that the exact symmetries of explicit symmetry domains relate them to the linear dynamic of the identity rule, ECA 204. We also give a counterexample demonstrating that domain behaviors are not always created by additive subdynamics. First though, we define additive CA dynamics and introduce the formalism of CA subdynamics.

5.1 Additive CAs

A CA is *additive* if the output $\phi(\eta)$ may be written as a linear combination of the neighborhood entries:

$$x_{t+1}^r = \phi\big(\ \eta(x_t^r)\ \big) = \sum_{i=-R}^{R} a_i x_t^i \ (\mathrm{mod}\ |\mathcal{A}|)\ , \tag{5.1}$$

where $a_i \in \mathcal{A}$.

CAs are referred to as *linear* if they obey a linear superposition principle. For any N radius-R neighborhood configurations η_i, the local update rule ϕ satisfies:

$$\sum_{i=1}^{N} \phi(\eta_i) = \phi\left(\sum_{i=1}^{N} \eta_i\right)\ , \tag{5.2}$$

where $\sum$ denotes addition modulo $|\mathcal{A}|$ over each site in the neighborhoods [215, 218]. Additivity and linearity are distinct properties. For CAs with an alphabet $\mathcal{A}$ that has a prime ring structure, including the binary-alphabet CAs considered here, additivity and linearity are equivalent (each implies the other) [221]. And so, we use both terms interchangeably.

Linear superposition enables powerful analytic techniques. For example, representing spatial configurations via characteristic polynomials gives a linear time-evolution via multiplication of a second polynomial representing Φ [215]. Or, one can employ the ergodic theorem for commuting transformations together with Fourier analysis [216]. Similarly, ergodic theory interprets additive CAs as endomorphisms of a compact Abelian group, which have been thoroughly investigated [222].

For general one-dimensional CAs with alphabet $\mathcal{A}$ and neighborhood radius R, there are $|\mathcal{A}|^{|\mathcal{A}|^{2R+1}}$ update rules. However, only $|\mathcal{A}|^{2R+1}$ are linear. Though there are cases where nonlinear rules may be mapped onto linear rules [218], the analytic tools just described apply to a vanishingly small subset of CA rules; hence the interest in extensions to a wider class of CAs.

In our analysis we examine the powers of the lookup table, also called *higher-order lookup tables*. The n^{th}-order lookup table ϕ^n maps the radius $n \cdot R$ neighborhood of a site to that site's value n time steps in the future. Said another way, a spacetime point x^r_{t+n} is completely determined by the radius $n \cdot R$ neighborhood n time-steps in the past according to:

$$x^r_{t+n} = \phi^n\left(\eta^n(x^r_t)\right) \ .$$

5.2 CA Subdynamics

Laying the groundwork for a CA's subdynamics requires clarity on what is meant by the CA dynamic itself. The global dynamic Φ is implemented through synchronous, parallel application of the local dynamic ϕ. And so, Φ and ϕ are closely related, but they should not be conflated. In what follows, *dynamic* refers to ϕ and, more specifically, to its lookup table. However, since ϕ is the building block of Φ, restrictions on ϕ induce restrictions on Φ. Specifically, these constrain the configurations that we consider evolving under Φ.

A subdynamic of ϕ then is determined by a subset of elements from its lookup table. (This need not be a proper subset, we consider the full dynamic ϕ to be a subdynamic.) To formalize this we must recast ϕ's lookup table (LUT) as a set. We do this by considering elements of the set $\text{LUT}(\phi)$ as tuples of neighborhood values and their outputs under ϕ:

$\mathrm{LUT}(\phi) \equiv \{\big(\eta, \phi(\eta)\big) : \text{ for all } \eta \in \mathcal{A}^{2R+1}\}$. We also need to consider higher-order lookup tables. The generalization to $\mathrm{LUT}(\phi^n)$ is straightforward:

$$\mathrm{LUT}(\phi^n) \equiv \{\big(\eta^n, \phi^n(\eta^n)\big) : \text{ for all } \eta^n \in \mathcal{A}^{2nR+1}\} \ ,$$

where the η^n runs through all length $2nR+1$ words in $\mathcal{A}$.

We consider two methods for removing elements from $\mathrm{LUT}(\phi^n)$ to create a subdynamic of ϕ^n. The first is related to additive dynamics.

5.2.1 Lookup Table Linearizations

Recall that an additive local dynamic may be written in the form of Equation (5.1). For each $x^i,\ i \in \{-R, -R+1, \ldots, R-1, R\}$, in neighborhood η there is a coefficient $a_i \in \mathcal{A}$ such that the output of ϕ for that neighborhood is given by Equation (5.1). Every linear rule then is specified by a length $2R+1$ vector $\mathbf{a}$ of these coefficients. For example, ECA rule 90 is given by $\mathbf{a}_{90} = (1, 0, 1)$, since $x^i_{t+1} = \phi_{90}(x^{i-1}_t x^i_t x^{i+1}_t) = 1x^{i-1}_t + 0x^i + 1x^{i+1}_t \pmod 2$.

For a linear rule given as a coefficient vector, the vector $\mathbf{o} = \mathbf{a}\mathbf{N}^{\mathrm{T}}$ gives the lookup table outputs, where $\mathbf{N}$ is the matrix of neighborhood values, each row of $\mathbf{N}$ is a neighborhood η, and these are given in lexicographical order. Thus, we can refer to linear rules via their coefficient vector $\mathbf{a}$, which will allow us to easily enumerate every additive CA in a given class.

Consider an arbitrary lookup table $\mathrm{LUT}(\phi_\alpha)$ and an additive lookup table $\mathrm{LUT}(\phi_\beta)$ in the same CA class.

Definition 3. Construct the *ϕ_β-linearization of ϕ_α*, denoted $\phi_{\alpha\leftrightarrow\beta}$, by removing elements $(\eta, \phi_\alpha(\eta))$ from $\mathrm{LUT}(\phi_\alpha)$ if and only if $\phi_\alpha(\eta) \neq \phi_\beta(\eta)$. That is, we only keep elements in $\mathrm{LUT}(\phi_\alpha)$ that are also in $\mathrm{LUT}(\phi_\beta)$.

Being linear, ϕ_β has a coefficient vector $\mathbf{a}_\beta$, which is now also associated with $\phi_{\alpha\leftrightarrow\beta}$. For each element $(\eta, \phi_{\alpha\leftrightarrow\beta}(\eta)) \in \mathrm{LUT}(\phi_{\alpha\leftrightarrow\beta})$ the output $\phi_{\alpha\leftrightarrow\beta}(\eta)$ is given by $\mathbf{a}_\beta \cdot \eta$, treating the neighborhood η as a vector and the dot product sum is performed mod $|\mathcal{A}|$. β may be given as a CA rule number or as a coefficient vector; e.g., $\mathrm{LUT}(\phi_{18\leftrightarrow 90})$ is the same as $\mathrm{LUT}(\phi_{18\leftrightarrow(1,0,1)})$.

Generalizing to higher-order lookup tables, $\mathrm{LUT}(\phi^n_{\alpha\leftrightarrow\beta})$ denotes keeping only elements in $\mathrm{LUT}(\phi^n_\alpha)$ if and only if $\phi^n_\alpha(\eta^n) = \phi^n_\beta(\eta^n)$. At higher powers, there may be linearizations that are not powers of a linear rule in the CA class. For example, take ϕ_α to be an ECA—CA class ($\mathcal{A} = \{0,1\}, R = 1$). At the second power, we may want a linearization with coefficient vector $\mathbf{a} = (0,1,1,1,1)$, which is not the second power of a linear ECA. However, we still want this as a possible linearization of $\mathrm{LUT}(\phi^2_\alpha)$. In such cases, the coefficient vector will be used for β: $\mathrm{LUT}(\phi^2_{\alpha\leftrightarrow(0,1,1,1,1)})$.

Notice that the construction of $\mathrm{LUT}(\phi_{\alpha\leftrightarrow\beta})$ is symmetric between $\mathrm{LUT}(\phi_\alpha)$ and $\mathrm{LUT}(\phi_\beta)$ (via set intersection), and so $\mathrm{LUT}(\phi_{\alpha\leftrightarrow\beta}) = \mathrm{LUT}(\phi_{\beta\leftrightarrow\alpha})$. That being said, $\mathrm{LUT}(\phi_{\alpha\leftrightarrow\beta})$ *linearizes* ϕ_α to ϕ_β, while the opposite is not true; $\mathrm{LUT}(\phi_{\beta\leftrightarrow\alpha})$ does not nonlinearize ϕ_β. Any subset of an additive lookup table is necessarily also additive.

5.2.2 Language-Restricted Lookup Tables

Lookup table linearization, as described, is a procedure for defining a particular CA subdynamic by focusing purely on the lookup table for that CA. Such a subdynamic may not be realizable on a nontrivial set of spatial configurations, though. For example, ECA rule 204 is the identity rule, which is linear with coefficient vector $\mathbf{a}_{204} = (0,1,0)$. If we reduce rule 18 to rule 204, the only neighborhoods in $\mathrm{LUT}(\phi_{18\leftrightarrow204})$ are 000 and 101. The only configurations (longer than 3) with only these neighborhoods are all-0 configurations.

The motivation for the second subdynamic-construction method derives directly from the *configurational consistency* lacking in the $\phi_{18\leftrightarrow204}$ example. While lookup table linearization subdynamics are constructed by considering the outputs $\phi_\alpha(\eta)$, language-restricted subdynamics are constructed employing the neighborhoods η themselves.

Consider a sofic language L (Section 4.3), its machine $M(L)$, and an arbitrary CA rule ϕ_α.

Definition 4. Construct the *L-restriction of ϕ_α*, denoted $\phi_{\alpha|L}$, by removing elements $(\eta, \phi_\alpha(\eta))$ from $\mathrm{LUT}(\phi_\alpha)$ if and only if $\eta \notin L$.

That is, consider each neighborhood η as a length $2R+1$ word and only keep the neighborhoods that belong to the language L. Operationally, if $\eta \in L$, there exists a path

in $M(L)$ such that concatenation of output symbols along that path give the word η.

This easily generalizes to higher powers of LUT(ϕ_α), as we can consider the neighborhoods η^n as words of length $2nR+1$, keeping elements $(\eta^n, \phi_\alpha^n(\eta^n))$ in LUT(ϕ_α^n) if and only if $\eta^n \in L$.

5.3 Domain-Restricted Lookup Tables and Their Linearizations

When a CA lookup table is restricted to one of its domain languages, *temporal consistency* is then added to the configurational consistency of the language-restricted subdynamics. Surprisingly, restricting a nonlinear CA to a domain can result in an additive subdynamic, thus embedding realizable linear behavior in a generally nonlinear system. More surprising, we demonstrate below that every example of an ECA domain found in the literature corresponds to an additive subdynamic. This correspondence between domains and additive subdynamics does not hold generally, however, as we demonstrate with a counterexample from the CA class ($\mathcal{A} = \{0,1\}, R = 2$).

In the absence of a general correspondence, why is it that so many ECA domains *do* embed an additive subdynamic? It may be that domain-restricted lookup tables generally do correspond to *permutive* subdynamics, rather than additive subdynamics. These are the *partially permutive* CAs of Ref. [223]. Permutive CAs are a superset of additive CAs (all additive CAs are permutive, but not all permutive CAs are additive). Further investigations are required to show whether or not CA domains generally embed permutive subdynamics. Even if there is such a general correspondence with permutive subdynamics, there still remains the question of why so many ECA domains have not just a permutive subdynamic, but an additive subdynamic. Given the historical interest in embeddings of linear behaviors in nonlinear CAs, we address this particular question in the remainder of Chapter 5.

Embedding of linear behavior in a nonlinear CA is certainly an interesting property of the nonlinear CA, but the mechanism for this behavior is actually best understood through the linear CA that the nonlinear CA emulates over its domain. We will start by

showing that, for binary-alphabet CAs, additive dynamics produce *only* domain behavior. That is, the full binary shift $\mathcal{A}^{\mathbb{Z}}$ is invariant under evolution of an additive CA Φ_β. If any proper subshift $\mathcal{X} \subset \mathcal{A}^{\mathbb{Z}}$ is also invariant under an additive Φ_β, it is by definition linear (for prime ring alphabet CAs). When a proper subshift is invariant under an additive Φ_β it specifies an additive subdynamic. This leaves vacancies in the full lookup table that can be filled in to produce a non-additive lookup table that supports the domain additive subdynamic. We find invariant proper subshifts of the well-studied additive ECA rule 90 to be the mechanism that generates additive subdynamics in the known cases of linear behavior embedded in nonlinear ECAs.

Before beginning, a few comments about implementing the analysis tools are in order. ECA lookup tables and their subdynamics are easily analyzed by hand, but higher powers of the lookup tables quickly become unmanageable. Fortunately, the subdynamic formalism just developed lends itself to automation. In the following, this was implemented in Python, allowing for exact symbolic exploration of higher-order lookup-table linearizations. For example, given the set $\mathrm{LUT}(\phi^n_{\alpha|L})$ we can enumerate all possible $2nR + 1$ linearizations $\mathbf{a}$ to check whether $\mathrm{LUT}(\phi^n_{\alpha|L}) \subseteq \mathrm{LUT}(\phi^n_{\alpha \leftrightarrow \mathbf{a}})$. If so, ϕ^n_α linearizes to $\mathbf{a}$ over language L.

Similarly, topological reconstruction (see Section 4.2) of the local causal states for each example was also implemented in Python. Since the local causal state labels in causal filtering are arbitrary, we use a simple, but arbitrary alphabetical labeling. To avoid confusion, we emphasize that for different CAs there is no connection between states labeled the same. For instance, we give causal filterings of ECAs 90 and 150 in Figure 5.1, and in each of these there are local causal states labeled A. There is no relation, however, between state A of rule 90 and state A of rule 150.

5.4 Additive CAs Produce Only Domains

We start with linear CAs and find that *all* behaviors they produce are domains. From the invariant set perspective of domain, we state this formally.

Theorem 1. Every nonzero linear CA Φ_β with $\mathcal{A} = \{0, 1\}$ is a factor map from the full-$\mathcal{A}$

shift to itself:

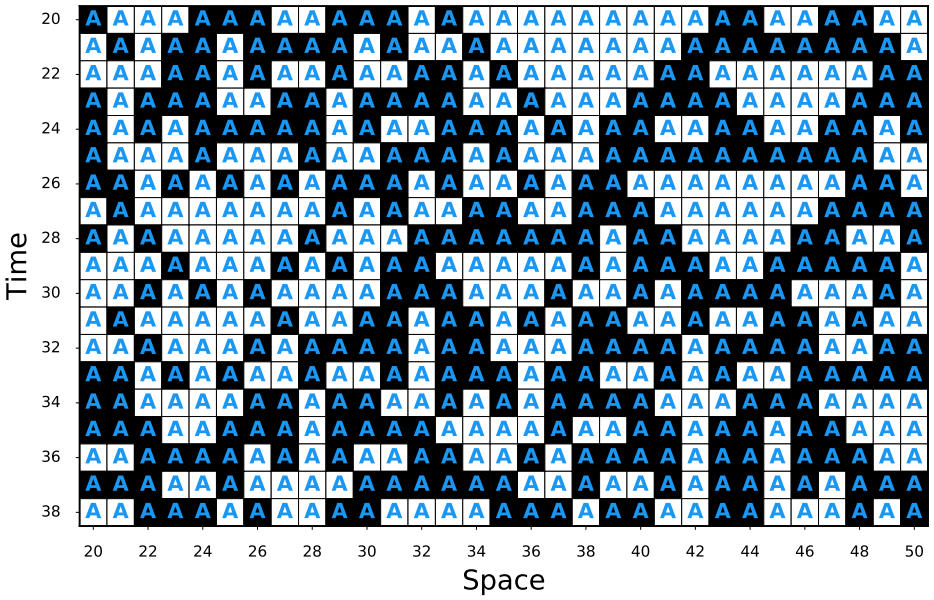
$$\Phi_\beta : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}} \ .$$

Proof sketch Using linear superposition, Equation (5.2), and additivity, Equation (5.1), a right inverse $\widetilde{\Phi}_\beta$ can be constructed for any linear Φ_β so that $\widetilde{\Phi}_\beta\big(\Phi_\beta\left(x\right)\big) = x$, for all $x \in \mathcal{A}^{\mathbb{Z}}$. See the Appendix in Section 5.9.1 for the detailed proof.

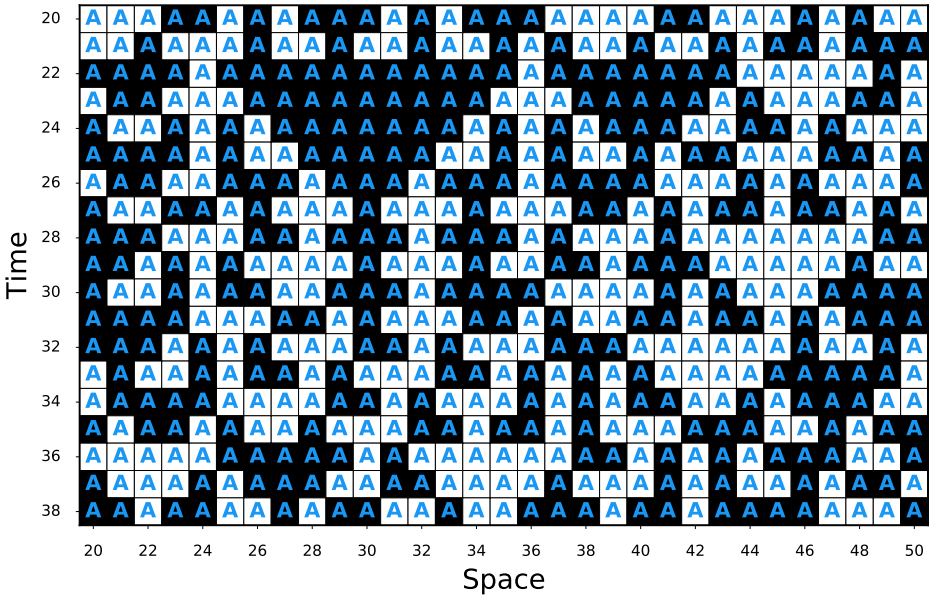
From Theorem 1 we see that every orbit under Φ_β lies in the domain invariant set since the invariant set is the set $\mathcal{A}^{\mathbb{Z}}$ of all spatial configurations. This means Φ_β induces no spatial restrictions on its images. Though we must note this result is for bi-infinite spatial configurations $x \in \mathcal{A}^{\mathbb{Z}}$. If a linear CA evolves configurations using other global topologies (most commonly a finite ring topology) then it may not be surjective. For example, finite spatial configurations with an odd number of sites with value 1 are not reachable by the linear ECA Φ_{90}; there is no y such that $x = \Phi_{90}(y)$ for finite x with an odd number of 1s [215]. Such unreachable states are also known as *Garden of Eden* states.

What is the local causal state signature of additive CAs that only produce domain behaviors? The answer is remarkably simple. When spacetime fields are filtered with the local causal-equivalence relation, there is only a single local causal state. That is, for *every* spacetime field $\mathbf{x}$ produced by Φ_β, the associated local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ consists of a single state. Thus, the filtered field has trivial time and space translation symmetries. Moreover, since there is only one local causal state, this symmetry can never be broken. And so, all spacetime fields of Φ_β are necessarily pure domain fields.

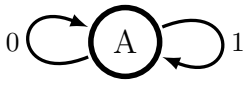
Figure 5.1 displays spacetime fields $\mathbf{x}$ generated by rule 90, $\mathbf{a}_{90} = (1, 0, 1)$, and rule 150, $\mathbf{a}_{150} = (1, 1, 1)$, with the associated local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ superimposed on top. The white and black squares are the site values 0 and 1, respectively, of the CA spacetime field. While the blue letters denote the local causal states for each site in the field. Such diagrams, with the local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ superimposed over its associated CA spacetime field $\mathbf{x}$, are called *local causal state overlay diagrams*.



(a) Rule 90 spacetime field and associated local causal states.



(b) Rule 150 spacetime field and associated local causal states.



(c) Machine $M(\mathcal{A}^{\mathbb{Z}})$ of domain $\Lambda = \mathcal{A}^{\mathbb{Z}}$ for $\phi = 90$ and $\phi = 150$.

Figure 5.1. Causally-filtered spacetime fields of (a) rule 90 and (b) rule 150, evolved from random initial conditions. White and black squares represent 0 and 1 CA site values, respectively. While blue letters are the associated local causal state label for that site. (c) Their domains $\Lambda_{90} = \mathcal{A}^{\mathbb{Z}}$ and $\Lambda_{150} = \mathcal{A}^{\mathbb{Z}}$ are described by the single-state machine $M(\mathcal{A}^{\mathbb{Z}})$.

5.4.1 Causal asymmetry of Rule 60

While $\mathcal{A}^{\mathbb{Z}}$ is invariant under the additive rule 60, the local causal state analysis reveals an interesting subtlety. Using standard lightcones, as depicted in Figure 3.4, local causal state inference was run over rule 60 using topological reconstruction. The resulting causal filtering is shown in the overlay diagram of Figure 5.2(a). As can be seen, there are multiple causal states (cf. rules 90 and 150 above with one) and there are no obvious spacetime translation symmetries in $\mathcal{S}$.

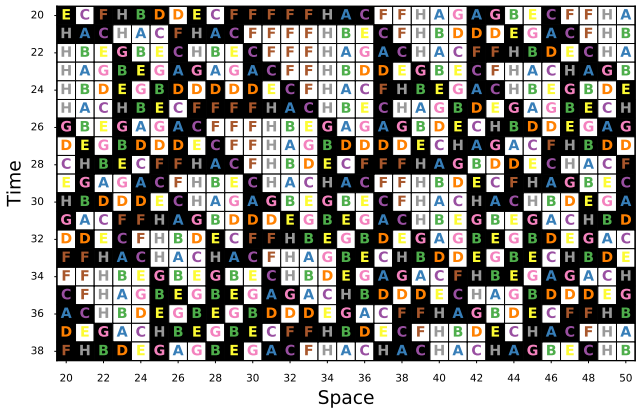
This occurs since rule 60 breaks the causal symmetry assumption implicit in standard lightcones. Since rule 60, $\mathbf{a}_{60} = (1, 1, 0)$, is the sum mod 2 of the center and left bit in the radius-1 neighborhood the right neighbor is irrelevant to the dynamic. One takes this left-skewed causal asymmetry into account by modifying the lightcone shape used in local causal equivalence. The new, seemingly appropriate lightcone is depicted in Figrue. 5.2(c). Applying local causal state filtering using these half-lightcones yields the overlay diagram of Figure 5.2(b): A single local causal state is revealed. Thus, taking into account the causal asymmetry, local causal state analysis in fact demonstrates that rule 60 produces only pure-domain fields. This holds similarly for rule 102, $\mathbf{a}_{102} = (0, 1, 1)$, when the mirror-symmetry right-skewed half-lightcones are used.

Since it demonstrates the consequences (and power) of the weak-causality argument for using lightcones as local pasts and futures, the result here is significant. Tracking weak-causality—how information locally propagates through points in spacetime—is necessary for relating emergent behavior to properties of the system that generated the behavior. Here, we know rule 60 is additive and we know $\mathcal{A}^{\mathbb{Z}}$ is invariant under Φ_{60}, so we know it should have a single local causal state. However, a single local causal state is properly inferred only if we account for rule 60's inherent causal asymmetry.

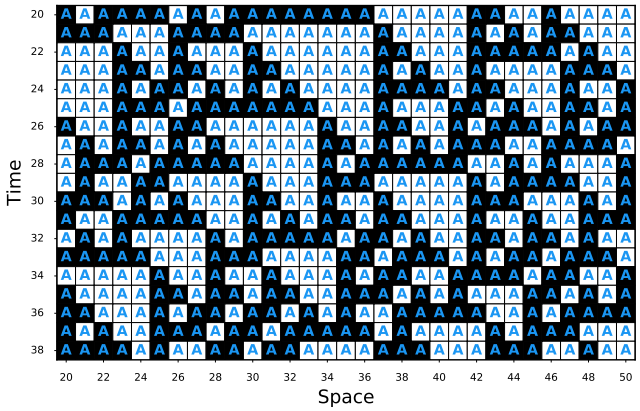
5.5 Explicit Symmetry Domains

Before moving on to the main results concerning invariant subshifts of ECA Rule 90, let's quickly look at explicit symmetry domains through the lens of additive subdynamics.

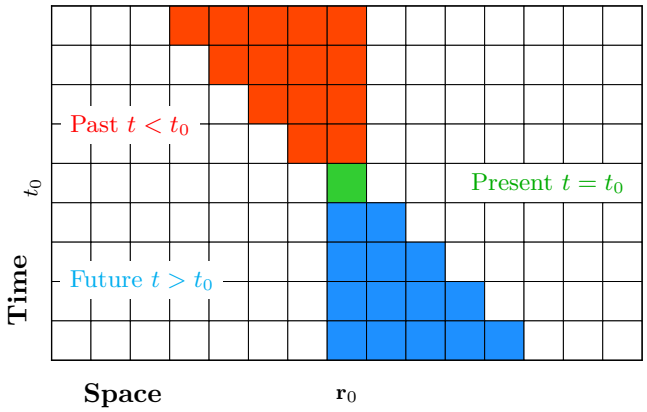
Lemma 1 established that explicit symmetry domains are periodic orbits of their CA.



(a) Local causal state field of rule 60 using full lightcones.



(b) Local causal state field of rule 60 using half-lightcones.



(c) Half-lightcones for left-skewed causal asymmetry.

Figure 5.2. Filtered spacetime field of rule 60, evolved from random initial conditions. White and black squares represent 0 and 1 site values, respectively. Colored letters denote the associated local causal state label for a site. (a) The result of using full lightcones, as depicted in Figure 3.4, for local causal state filtering. (b) Local causal state filtering appropriate to the left-skewed causal asymmetry of rule 60, using half-lightcones depicted in (c). This filtering recovers the single-state, trivially symmetric, local causal state field expected for additive CAs.

Due to this, explicit symmetry domains linearize to the identity rule, which for ECAs is rule 204, $\mathbf{a}_{204} = (0, 1, 0)$. For general $\mathcal{A} = \{0, 1\}$ CAs we denote the identity rule as $\mathbf{a}_{\mathrm{I}}$, whose coefficient vector has a single 1 for the center bit and all other elements 0.

Consider a CA Φ_α with an explicit symmetry domain Λ_α with temporal period p.

Theorem 2. The lookup table of ϕ_α, restricted to the domain Λ_α, linearizes to the identity rule at integer multiples of the domain temporal period p; that is:

$$\mathrm{LUT}(\phi^{np}_{\alpha|L(\Lambda_\alpha)}) \subseteq \mathrm{LUT}(\phi^{np}_{\alpha\leftrightarrow\mathbf{a}_{\mathrm{I}}}) \ ,$$

for $n = 1, 2, 3, \ldots$.

Proof. From Lemma 1, any configuration x_{Λ_α} in the domain produces a periodic orbit of ϕ_α, with orbit period p: $\Phi^p_\alpha(x_{\Lambda_\alpha}) = x_{\Lambda_\alpha}$. Since the full configuration returns after p time steps, so do all the individual sites of the configuration: $(x_{\Lambda_\alpha})^{r_0}_{t_0+p} = (x_{\Lambda_\alpha})^{r_0}_{t_0}$. Moreover, $(x_{\Lambda_\alpha})^{r_0}_{t_0+np} = (x_{\Lambda_\alpha})^{r_0}_{t_0}$, for $n = 1, 2, 3, \ldots$. Thus:

$$\begin{aligned}
(x_{\Lambda_\alpha})^{r_0}_{t_0+np} &= \phi^{np}_\alpha\left(\eta^{np}((x_{\Lambda_\alpha})^{r_0}_{t_0})\right) \\
&= (x_{\Lambda_\alpha})^{r_0}_{t_0} \\
&= \mathbf{a}_{\mathrm{I}} \cdot \left(\eta^{np}((x_{\Lambda_\alpha})^{r_0}_{t_0})\right) \ .
\end{aligned}$$

This is an equivalent statement to $\mathrm{LUT}(\phi^{np}_{\alpha|L(\Lambda_\alpha)}) \subseteq \mathrm{LUT}(\phi^{np}_{\alpha\leftrightarrow\mathbf{a}_{\mathrm{I}}})$.

■

For ϕ_α with an explicit symmetry domain Λ_α, more linearizations are possible, based on the spacetime symmetries of Λ_α. If Λ_α's recurrence time is smaller than its temporal period p, it takes fewer time translations than p to return all sites to themselves if also paired with spatial translations. As the local causal states fully capture the spatiotemporal symmetries of Λ_α, they are particularly convenient for expressing this.

Consider the local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ of a pure-domain field $\mathbf{x}$. From the definition of domain, $\mathcal{S}^{r_0}_{t_0+p} = \mathcal{S}^{r_0}_{t_0}$. And, for an explicit symmetry domain this means $\mathbf{x}^{r_0}_{t_0+p} = \mathbf{x}^{r_0}_{t_0}$, for all (r_0, t_0), which provides the connection to the identity rule $\mathbf{a}_{\mathrm{I}}$ stated above. From the definition of recurrence time, we have $\mathcal{S}^{r_0+\delta}_{t_0+\widehat{p}} = \mathcal{S}^{r_0}_{t_0}$, for some spatial

translation δ. For explicit symmetry domains this again implies the same invariance for $\mathbf{x}$. There may be other spacetime translation symmetry generators such that $\mathcal{S}_{t_0+i}^{r_0+j} = \mathcal{S}_{t_0}^{r_0}$. If $\mathcal{S}_{t_0}^{r_0}$ lies within the past lightcone of $\mathcal{S}_{t_0+i}^{r_0+j}$, there is a linearization of $\mathrm{LUT}(\phi_{\alpha|\Lambda_\alpha})$ at the i^{th} power, with linear coefficient vector $\mathbf{a} = (a_{-iR}, a_{-iR+1}, \ldots, a_{-1}, a_0, a_1, \ldots, a_{iR-1}, a_{iR})$ such that only $a_j = 1$ and all other coefficients are zero.

For example, Λ_{110} has recurrence time $\widehat{p} = 1$, temporal period $p = 7$, and spatial period $s = 14$, as can be seen in Figure 5.3(c). Thus, we know $\mathrm{LUT}(\phi^7_{110|L(\Lambda_{110})}) \subseteq \mathrm{LUT}(\phi^7_{110\leftrightarrow 204})$. However, we also see in Figure 5.3(c) that $\mathcal{S}_{t_0-3}^{r_0-2} = \mathcal{S}_{t_0}^{r_0}$. If we pick any site in the field, it will have some local causal state label; e.g., A. Shifting up three sites and left two will always take one to same local causal state label. Exact symbolic calculations showed that $\mathrm{LUT}(\phi^3_{110|L(\Lambda_{110})})$ linearizes to $\mathbf{a} = (0, 1, 0, 0, 0, 0, 0)$. Similarly, $\mathcal{S}_{t_0-4}^{r_0+2} = \mathcal{S}_{t_0}^{r_0}$ and $\mathrm{LUT}(\phi^4_{110|L(\Lambda_{110})})$ linearizes to $\mathbf{a} = (0, 0, 0, 0, 0, 0, 1, 0, 0)$.

There may be yet further linearizations. These again derive from Λ_α's symmetries, but the resulting linearizations are not related to the identity rule. Let's start with an example. Consider again rule 110 and its domain, as shown in Figure 5.3(c). We find several linearizations of $\mathrm{LUT}(\phi^7_{110|L(\Lambda_{110})})$, including $\mathbf{a}_{\mathrm{I}} = (0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0)$, the identity rule discussed at the section's beginning. Another linearization we find at the 7^{th} power is $\mathbf{a} = (0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1)$. To understand this one, and others like it, it is again useful to refer to spacetime diagrams of Λ_α (Λ_{110} in this case).

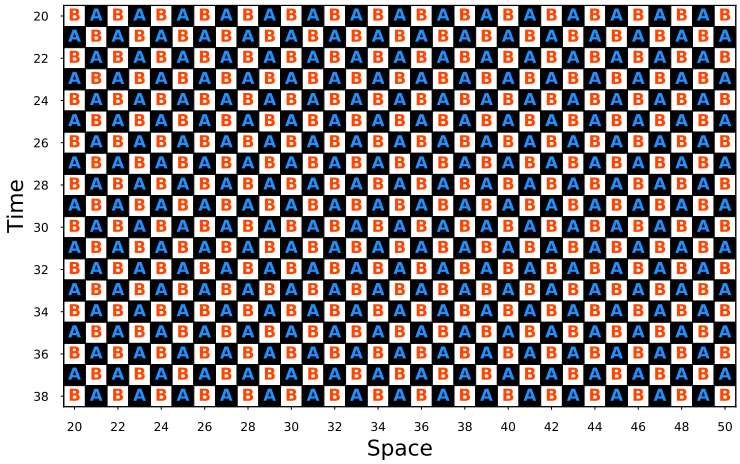
A linearization $\mathbf{a}$ means a spacetime point $\mathbf{x}_t^r \in \Lambda_\alpha$ is 0 if $\eta^n(\mathbf{x}_t^r) \cdot \mathbf{a}$ is even and $\mathbf{x}_t^r$ is 1 if the dot product is odd, where we treat the neighborhood as a vector. This must be satisfied at *every* spacetime point in a pure domain field $\mathbf{x}_{\Lambda_\alpha}$. If:

$$\mathbf{x}_{t_0}^{r_0} = \eta^n(\mathbf{x}_{t_0}^{r_0}) \overset{(\mathrm{mod}\ 2)}{\cdot} \mathbf{a} \ , \tag{5.3}$$

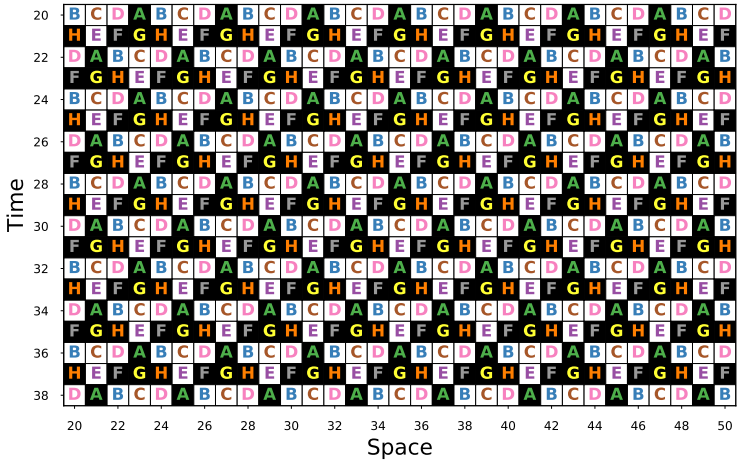
then for an explicit symmetry domain:

$$\mathbf{x}_{t_0+ip}^{r_0+js} = \eta^n(\mathbf{x}_{t_0+ip}^{r_0+js}) \overset{(\mathrm{mod}\ 2)}{\cdot} \mathbf{a}$$

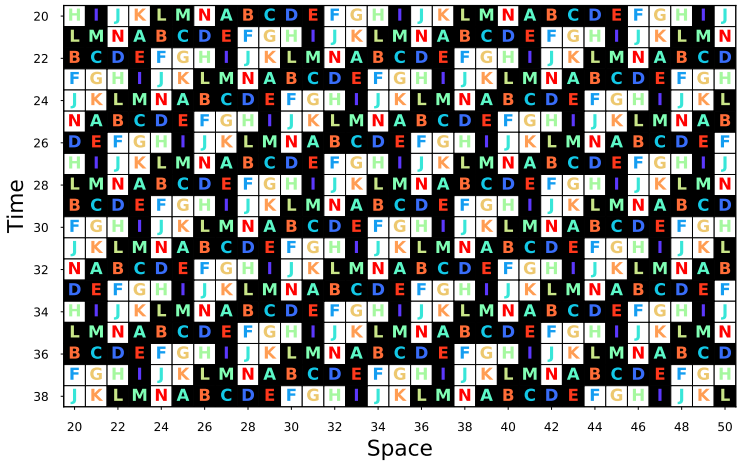
for $i, j \in \mathbb{Z}$. Thus, there is only a relatively small number of neighborhoods $\eta^n(\mathbf{x}_t^r)$ in spacetime fields of explicit symmetry domains that must satisfy Equation (5.3). Said



(a) Domain of rule 58 with associated local causal states.



(b) Domain of rule 54 with associated local causal states.



(c) Domain of rule 110 with associated local causal states.

Figure 5.3. Filtered spacetime fields from the explicit symmetry domains of rules 58 (a), 54 (b), and 110 (c), with the associated local causal states superimposed on top.

another way, the languages of explicit symmetry domains are very restrictive. This means there are relatively few entries in $\mathrm{LUT}(\phi^n_{\alpha|L(\Lambda_\alpha)})$ that must obey additivity.

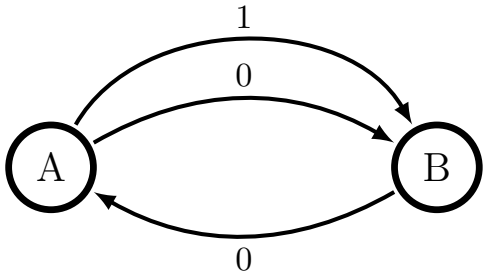
To further illustrate this linearization type, contrast the explicit symmetry domains shown in Figure 5.3: (a) the rule 58 domain Λ_{58} with characteristics $\widehat{p} = 1$, $p = 2$, and $s = 2$; and (b) the rule 54 domain Λ_{54} with characteristics $\widehat{p} = 2$, $p = 4$, and $s = 4$. Now, compare their linearizations at the 4^{th} power, this being a multiple of the temporal period p for both domains.

The 4^{th} power reveals more linearizations for Λ_{58} than for Λ_{54} not related to the identity rule—those with more than one nonzero element in the coefficient vector $\mathbf{a}$. Specifically, there are 29 linearizations for Λ_{54} and 123 for Λ_{58}. This is expected, as Λ_{58} is generated by smaller translations s and p than Λ_{54} (i.e. Λ_{58} has more symmetries than Λ_{54}). This means at a given power n, there are fewer distinct neighborhoods η^n in Λ_{58} than in Λ_{54}, and so there can be more linearizations $\mathbf{a}$ that can satisfy Equation (5.3) everywhere in the field.

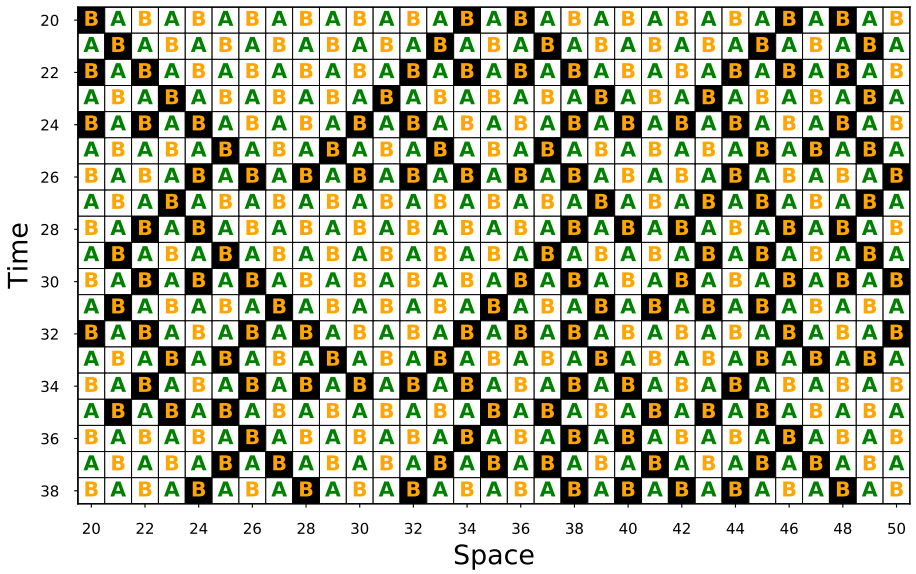
Before moving on to hidden symmetry domains, in closing we should highlight the role of spacetime symmetries for linearizations of explicit symmetry domains. Notice that the domain's spatial languages themselves were never directly needed. Rather, the symmetries of the spacetime field orbits and their orbit period (i.e., the domain temporal period) that were key. This reflects the local causal-state perspective of domain, as in Definition 2, coming into play.

5.6 Hidden Symmetry Domains and ECA Rule 90

In contrast, as we now show, linearizations of hidden symmetry domains are based on the domain spatial languages and their recurrence times, as in Definition 1. Moreover, unlike explicit symmetry domains, we find that not every hidden symmetry domain has a linearization. After going through detailed examples of ECA domains, we will close with an $R = 2$ CA that has a "nonlinear" domain.



(a) Finite-state machine $M(\Lambda_{18})$ of invariant set language Λ_{18}.



(b) Sample spacetime field $\mathbf{x}_{\Lambda_{18}}$ of Λ_{18} and associated local causal state field $\mathcal{S}_{\Lambda_{18}} = \epsilon(\mathbf{x}_{\Lambda_{18}})$.

Figure 5.4. (a) Finite-state machine $M(\Lambda_{18})$ for the invariant set language $L(\Lambda_{18})$ of the rule 18 domain. (b) Filtered spacetime field $\mathbf{x}_{\Lambda_{18}}$ (white and black squares) of the rule 18 domain Λ_{18} with the associated local causal state field $\mathcal{S}_{\Lambda_{18}} = \epsilon(\mathbf{x}_{\Lambda_{18}})$ (green and orange letters) superimposed.

5.6.0.1 Rule 18

We start by examining in detail the original observations [217, 218, 5] concerning the domain in the nonadditive CA rule 18 and the domain's linearization to rule 90, $\mathbf{a}_{90} = (1, 0, 1)$. In words, ϕ_{90} updates lattice sites according to the sum mod 2 of that site's left and right neighbors:

$$\begin{aligned} \mathbf{x}_{t+1}^r &= \phi_{90}\big(\eta(\mathbf{x}_t^r)\big) \\ &= \mathbf{x}_t^{r-1} + \mathbf{x}_t^{r+1} \pmod 2 \ . \end{aligned}$$

Table 5.1 gives rule 90's lookup table.

Rule 18 is not additive, as can be seen from its lookup table also given in Table 5.1. However, there are special behaviors produced by rule 18 that were originally noted due to the equivalence between these behaviors of rule 18 and rule 90. More importantly, they suggested that a *nonlinear rule is capable of producing linear behaviors*. From Refs. [5, 193, 194, 116, 4], we know the special behaviors of rule 18 that emulate rule 90 are, in fact, rule 18's domain behaviors.

Rule 18's domain is the set of spatial configurations that is invariant under Φ_{18} and their spacetime field orbits. This invariant set is the single sofic shift $\Lambda_{18} = \{\mathcal{X}_{(0,\Sigma)}\}$, where Σ represents wildcard-sites that can be either 0 or 1. Its domain language is $L(\Lambda_{18}) = (0\Sigma)^* + (\Sigma 0)^*$. The set's finite-state machine $M(\Lambda_{18})$ is shown in Figure 5.4(a). Since the machine states lie in a single recurrent component, i.e., it has a single temporal phase, the recurrence time of Λ_{18} is $\widehat{p} = 1$. Its spatial period is $s = 2$, since this is the size of the minimal cycle of $M(\Lambda_{18})$.

Evolving spatial configurations $x \in \Lambda_{18}$ creates spacetime fields—their orbits $\mathbf{x}_{\Lambda_{18}}$. Applying the causal equivalence relation over these fields yields two local causal states, corresponding to the fixed-0 and wildcard sites. A sample spacetime field $\mathbf{x}_{\Lambda_{18}}$ of Λ_{18} and its causal filtering $\mathcal{S}_{\Lambda_{18}} = \epsilon(\mathbf{x}_{\Lambda_{18}})$ are shown as a local causal state overlay diagram in Figure 5.4 (b). State A corresponds to the fixed-0 sites and state B the wildcard states. These states appear in a checkerboard tiling in the field, displaying the defining spacetime symmetry of Λ_{18}. At each time step the same two states tile the spatial lattice, giving

η			$\phi_{90}(\eta)$	$\phi_{18}(\eta)$	$\phi_{18\leftrightarrow 90}(\eta)$	$\phi_{18\vert L(\Lambda_{18})}(\eta)$
1	1	1	0	0	0	–
1	1	0	1	0	–	–
1	0	1	0	0	0	0
1	0	0	1	1	1	1
0	1	1	1	0	–	–
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	0	0	0

Table 5.1. Lookup tables for rule 90 (ϕ_{90}) and rule 18 (ϕ_{18}) as well as for rule 18 linearized to rule 90 ($\phi_{18\leftrightarrow 90}$) and rule 18 restricted to its domain ($\phi_{18|L(\Lambda_{18})}$). The leftmost column gives all ECA neighborhood values in lexicographical order, and each subsequent column is the output of the neighborhoods for the specified dynamic or subdynamic. Symbol – indicates a lookup table element excluded from the respective subdynamic.

the recurrence time $\widehat{p} = 1$. The spatial period $s = 2$ and temporal period $p = 2$ are found from $\mathcal{S}_{\Lambda_{18}}$'s space and time translation invariance.

Having defined and described rule 18's domain Λ_{18} allows us to explain its linearization to rule 90: Keeping only elements of LUT(ϕ_{18}) that are also in LUT(ϕ_{90}) gives the linearization LUT($\phi_{18\leftrightarrow 90}$) of rule 18 to rule 90. This is shown in Table 5.1. In contrast, keeping only elements of LUT(ϕ_{18}) if the neighborhood η_i of that element belongs to the 0-Σ language of Λ_{18} gives the restriction LUT($\phi_{18|L(\Lambda_{18})}$) of ϕ_{18} to its domain. As shown in Table 5.1, this subdynamic excludes the neighborhoods $\eta \in \{111, 110, 011\}$. Table 5.1 shows that the only two elements that differ between LUT(ϕ_{90}) and LUT(ϕ_{18}) have neighborhoods $\eta \in \{110, 011\}$. Thus, LUT($\phi_{18|L(\Lambda_{18})}$) $\subset$ LUT($\phi_{18\leftrightarrow 90}$). Moreover, as Λ_{18} is a stochastic domain with recurrence time $\widehat{p} = 1$, we find that rule 18 restricted to its domain linearizes to rule 90 at all powers of the lookup tables:

$$\text{LUT}(\phi^n_{18|L(\Lambda_{18})}) \subseteq \text{LUT}(\phi^n_{18\leftrightarrow 90}) \ ,$$

for $n = 1, 2, 3, \ldots$.

η			ϕ_{90}	ϕ_{18}	ϕ_{26}	ϕ_{82}	ϕ_{146}	ϕ_{154}	ϕ_{210}	ϕ_{218}
1	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	1	1	1	1
1	1	0	1	0	0	1	0	0	1	1
0	1	1	1	0	1	0	0	1	0	1

Table 5.2. Lookup tables for rule 90 (ϕ_{90}) and the seven nonlinear rules, ϕ_α, $\alpha \in \{18, 26, 82, 146, 154, 210, 218\}$, that also have the $\Lambda_{0,\Sigma}$ domain invariant set. The first five rows correspond to the neighborhoods that belong to the domain language $L(\Lambda_{0,\Sigma})$. Since all eight rules have $\Lambda_{0,\Sigma}$ as a domain, the output for these five neighborhoods in $L(\Lambda_{0,\Sigma})$ are the same. The bottom three rows are the neighborhoods not in $L(\Lambda_{0,\Sigma})$. The eight rules in this table are all possible 2^3 output assignments for these three remaining neighborhoods.

5.6.1 Invariant Subshifts of Rule 90

Historically the 0-Σ domain was of interest because the nonlinear rule 18 exhibits linear behavior over $\Lambda_{0,\Sigma}$, since it emulates the linear rule 90 over $\Lambda_{0,\Sigma}$. However, we now know that $\Lambda_{0,\Sigma}$ is an invariant set of rule 18 and $\mathrm{LUT}(\phi_{18|L(\Lambda_{0,\Sigma})}) \subset \mathrm{LUT}(\phi_{18\leftrightarrow 90})$. This means $\Lambda_{0,\Sigma}$ is also an invariant set of rule 90.

Since $\mathrm{LUT}(\phi_{90})$ is additive, so is $\mathrm{LUT}(\phi_{90|L(\Lambda_{0,\Sigma})})$. And, as described above, there are three neighborhoods $\eta \in \{111, 110, 011\}$ in $\mathrm{LUT}(\phi_{90})$ that are not in $\mathrm{LUT}(\phi_{90|L(\Lambda_{0,\Sigma})})$. So, starting from $\mathrm{LUT}(\phi_{90|L(\Lambda_{0,\Sigma})})$, which is additive, there are three unconstrained outputs, one for each $\eta \in \{111, 110, 011\}$, to fill in to create an ECA lookup table that has $\Lambda_{0,\Sigma}$ as a linear invariant set. This is shown graphically in Table 5.2. The neighborhoods are ordered there so that the top five neighborhoods are those in $L(\Lambda_{0,\Sigma})$ and the bottom three are those that are not.

Rule 18 is just one of seven nonlinear rules that have $\Lambda_{0,\Sigma}$ as an invariant set that linearizes to rule 90: $\mathrm{LUT}(\phi^n_{\alpha|L(\Lambda_{0,\Sigma})}) \subseteq \mathrm{LUT}(\phi^n_{\alpha\leftrightarrow 90})$, for $n = 1, 2, 3, \ldots$ and $\alpha \in \{18, 26, 82, 146, 154, 210, 218\}$. Though Rule 146 emulating rule 90 over $\Lambda_{0,\Sigma}$ was pointed out in Ref. [218], to our knowledge the analysis here is the first identification of

η			$\phi_{90}(\eta)$	$\phi_{126}(\eta)$	$\phi_{126\leftrightarrow 90}(\eta)$	$\phi_{126\vert L(\Lambda_{\text{even}})}(\eta)$
1	1	1	0	0	0	0
1	1	0	1	1	1	1
1	0	1	0	1	–	–
1	0	0	1	1	1	1
0	1	1	1	1	1	1
0	1	0	0	1	–	–
0	0	1	1	1	1	1
0	0	0	0	0	0	0

Table 5.3. Lookup tables for rule 90 (ϕ_{90}) and rule 126 (ϕ_{126}) as well as for rule 126 linearized to rule 90 ($\phi_{126\leftrightarrow 90}$) and rule 126 restricted to its domain ($\phi_{126|L(\Lambda_{\text{even}})}$). Same format as in Table 5.1.

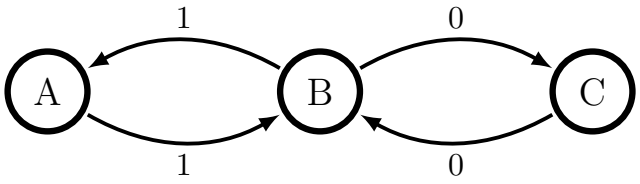
the linear $\Lambda_{0,\Sigma}$ domain in the nonlinear ECAs 26, 82, 154, 210, and 218. This is likely because $\Lambda_{0,\Sigma}$ does not appear to be a dominant behavior of rules 26, 82, 154, 210, and 218 from random initial conditions. In fact, simple stationary or oscillatory behaviors seem to be the dominant attractors for these rules.

Reference [218] also reported on the nonlinear rule 126 emulating rule 90. This was not over $\Lambda_{0,\Sigma}$ though. Instead it was over a different invariant set, that we call the *even* domain Λ_{even}. This domain also consists of a single temporal phase, which is the sofic shift that contains only even blocks of 1s and 0s. The machine $M(\Lambda_{\text{even}})$ for this domain is shown in Figure 5.5. A sample domain spacetime field $\mathbf{x}_{\Lambda_{\text{even}}}$, evolved from a domain configuration initial condition, is shown in Figure 5.5 with the associated local causal state field $\mathcal{S}_{\Lambda_{\text{even}}} = \epsilon(\mathbf{x}_{\Lambda_{\text{even}}})$ superimposed. Interestingly, though Λ_{even} and $\Lambda_{0,\Sigma}$ have different invariant spatial shifts, the resulting spacetime shifts of their orbits have the same generalized symmetries, as captured by the local causal states.

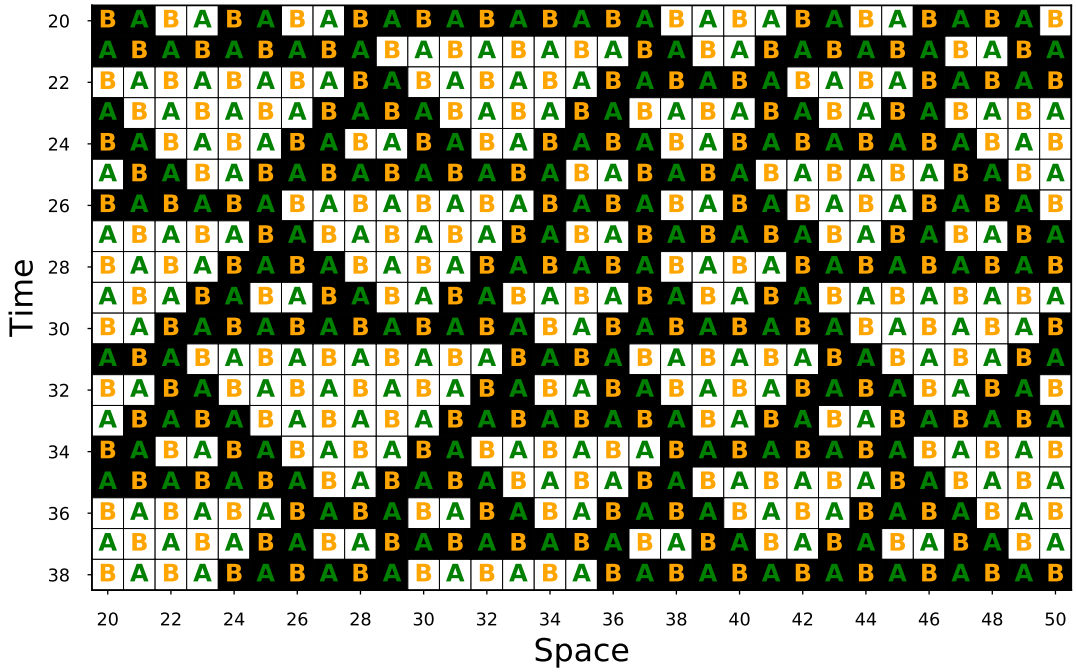
The lookup table for rule 126 is compared with that of rule 90 in Table 5.3, as well as its linearization to rule 90 and its restriction to Λ_{even}. From this we can see that $\text{LUT}(\phi_{126|L(\Lambda_{\text{even}})}) = \text{LUT}(\phi_{126\leftrightarrow 90})$. As with $\Lambda_{0,\Sigma}$ and rule 18, this means Λ_{even} is also an invariant set of rule 90. Thus, rule 126 linearizes to rule 90 over Λ_{even} at all powers:

$$\text{LUT}(\phi^n_{126|L(\Lambda_{\text{even}})}) \subseteq \text{LUT}(\phi^n_{126\leftrightarrow 90}) \ ,$$

for $n = 1, 2, 3, \ldots$.



(a) Machine $M(\Lambda_{\text{even}})$ of invariant language $L(\Lambda_{\text{even}})$.



(b) A spacetime field $\mathbf{x}_{\Lambda_{126}}$ of Λ_{126} and associated filtered local causal state field $\mathcal{S}_{\Lambda_{126}} = \epsilon(\mathbf{x}_{\Lambda_{126}})$.

Figure 5.5. (a) Machine $M(\Lambda_{\text{even}})$ for the invariant language $L(\Lambda_{\text{even}})$ of the rule 126 even domain. (b) Sample spacetime field $\mathbf{x}_{\Lambda_{126}}$ (black and white squares) of the even domain Λ_{even} of rule 126 with the associated local causal state field $\mathcal{S}_{\Lambda_{126}} = \epsilon(\mathbf{x}_{\Lambda_{126}})$ (green and orange letters) superimposed.

Following the same combinatorics as with $\Lambda_{0,\Sigma}$, we see from Table 5.3 that there are only two neighborhoods not in $L(\Lambda_{\text{even}})$. And so, there are 2^2 ECA rules with Λ_{even} as a domain invariant set. Two of these are rule 90 and rule 126; the other two are rule 94 and rule 122. Rule 122 is qualitatively similar to rule 126 over random initial conditions, while rule 94 generically settles into a fixed-point orbit.

Before moving on, clarification is in order. We said that $\Lambda_{0,\Sigma}$ and Λ_{even} are domain invariant sets of Rule 90. Formally, from Definition 1, Φ_{90} is a factor map from Λ to Λ for both of these domains. More specifically, this is true for *every* power of Φ_{90}: $\Phi_{90}^n : \Lambda \to \Lambda$ for all $n = 1, 2, 3, \ldots$. This is also holds for all the nonlinear rules just discussed that also have one of these domain invariant sets. Thus, these rules *emulate* rule 90 over their domain. That is, for all of these nonlinear ϕ_α, any orbit starting from an initial configuration $\widehat{x}$ in the appropriate domain Λ is the same if evolved under Φ_α or Φ_{90}:

$$\{\widehat{x}, \Phi_\alpha(\widehat{x}), \Phi_\alpha^2(\widehat{x}), \Phi_\alpha^3(\widehat{x}), \ldots\} = \{\widehat{x}, \Phi_{90}(\widehat{x}), \Phi_{90}^2(\widehat{x}), \Phi_{90}^3(\widehat{x}), \ldots\} \ .$$

5.6.2 Rule 22

The next example we explore is the enigmatic Rule 22 [224], which exhibits a more general notion of linearization. Using symbolic manipulation methods, Crutchfield and McTague used the FME analysis to discover this ECA's domain [210].

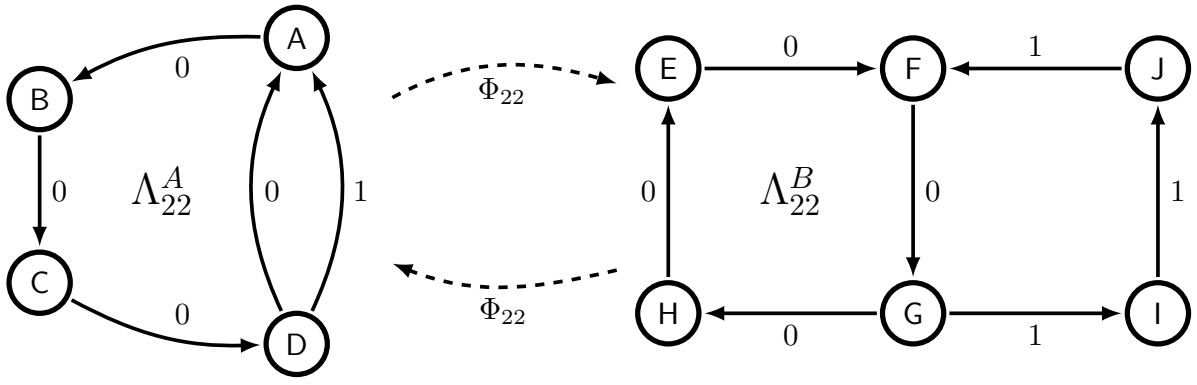


Figure 5.6. Machine $M(\Lambda_{22})$ for rule 22's domain Λ_{22}, which has two distinct temporal phases: $\Lambda_{22}^A = \Phi_{22}(\Lambda_{22}^B)$ and $\Lambda_{22}^B = \Phi_{22}(\Lambda_{22}^A)$.

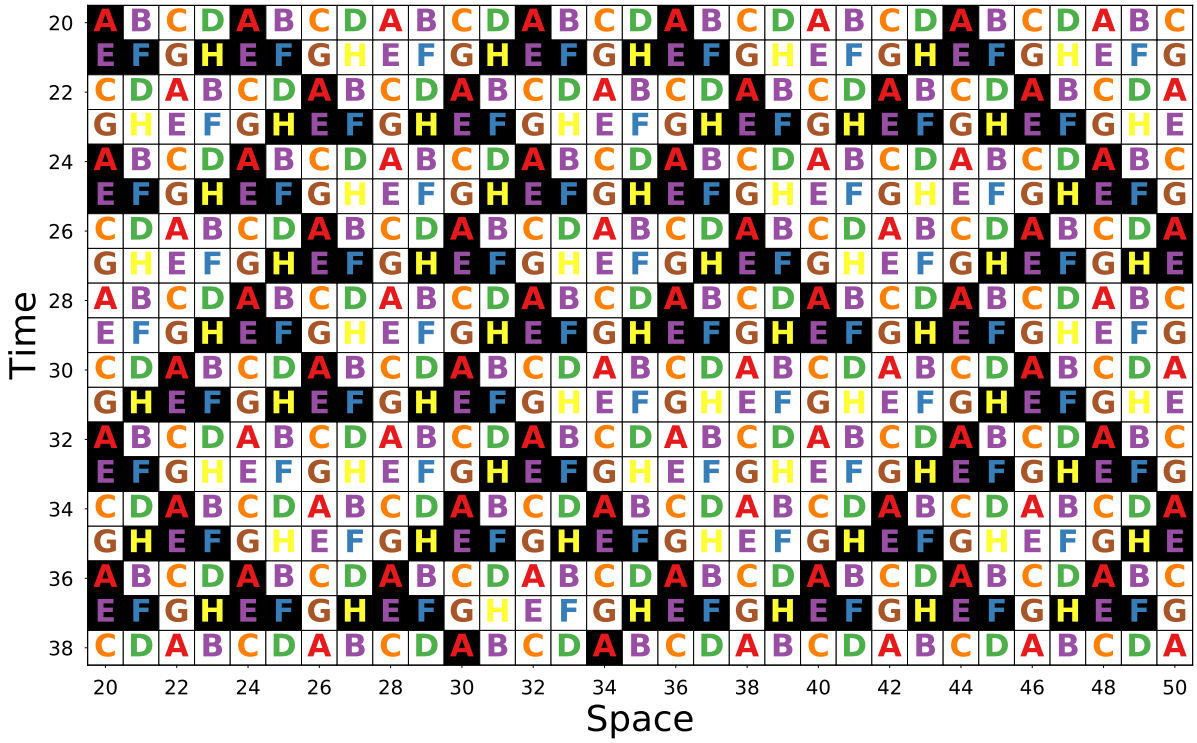


Figure 5.7. Filtered spacetime field $\mathbf{x}_{\Lambda_{22}}$ (white and black squares) of the rule 22 domain Λ_{22} with the associated local causal state field $\mathcal{S}_{\Lambda_{22}} = \epsilon(\mathbf{x}_{\Lambda_{22}})$ (colored letters) superimposed.

Rules dominated by a stochastic symmetry domain, such as rule 22, are sometimes referred to as "chaotic" CAs. As such, it is typically challenging to extract meaningful structures purely from visually inspecting spacetime fields. So, while not visually apparent, the domain underlying rule 22 is rather more complex than the previous ones. Much of rule 22's mystery stems from its complex domain.

The domain of rule 22 is comprised of two temporal phases, $\Lambda_{22} = \{\Lambda_{22}^A, \Lambda_{22}^B\}$. The machine presentation $M(\Lambda_{22})$ of Λ_{22} is shown in Figure 5.6. The two components correspond to the irreducible sofic shifts Λ_{22}^A and Λ_{22}^B. A sample spacetime field $\mathbf{x}_{\Lambda_{22}}$ of Λ_{22} is shown in Figure 5.7, with the associated local causal state field $\mathcal{S}_{\Lambda_{22}} = \epsilon(\mathbf{x}_{\Lambda_{22}})$ superimposed on top. There are two distinct spatial tilings of the local causal states, ABCD and EFGH, associated with Λ_{22}^A and Λ_{22}^B, respectively, giving a recurrence time $\widehat{p} = 2$. The spacetime translation invariance of $\mathcal{S}_{\Lambda_{22}}$ gives a spatial period of $s = 4$ and temporal period $p = 4$.

Since Λ_{22} has two temporal phases, care must be taken when discussing its invariance

and linearization. Reference [210]'s FME analysis established that $\Lambda_{22}^A = \Phi_{22}(\Lambda_{22}^B)$ and $\Lambda_{22}^B = \Phi_{22}(\Lambda_{22}^A)$. Thus, Φ_{22} is a factor map from each phase to itself only at the second power: $\Phi_{22}^2 : \Lambda_{22}^A \to \Lambda_{22}^A$ and $\Phi_{22}^2 : \Lambda_{22}^B \to \Lambda_{22}^B$. It is not surprising then that $\mathrm{LUT}(\phi_{22|L(\Lambda_{22})})$ is not additive at the first power, but at the second power. Specifically, $\mathrm{LUT}(\phi^2_{22|L(\Lambda_{22})})$ linearizes again to rule 90. In fact, exact symbolic calculation finds this linearization occurs at all even powers:

$$\mathrm{LUT}(\phi^{2n}_{22|L(\Lambda_{22})}) \subseteq \mathrm{LUT}(\phi^{2n}_{22\leftrightarrow 90}) \ ,$$

for $n = 1, 2, 3, \ldots$.

To be clear, $\mathrm{LUT}(\phi_{22|L(\Lambda_{22})})$ is constructed by keeping only neighborhoods that are in the language $L(\Lambda_{22})$, which is the union $L(\Lambda_{22}) = L(\Lambda_{22}^A) \cup L(\Lambda_{22}^B)$ of the temporal phase languages. Since $\mathrm{LUT}(\phi^{2n}_{22|L(\Lambda_{22})})$ is additive, then its subsets $\mathrm{LUT}(\phi^{2n}_{22|L(\Lambda_{22}^A)})$ and $\mathrm{LUT}(\phi^{2n}_{22|L(\Lambda_{22}^B)})$ are also additive. That is, $\mathrm{LUT}(\phi^{2n}_{22|L(\Lambda_{22}^A)}) \subseteq \mathrm{LUT}(\phi^{2n}_{22\leftrightarrow 90})$ and similarly for phase B. Therefore Λ_{22}^A and Λ_{22}^B are invariant sets of Φ_{90}^2. However, we cannot say Λ_{22} is a domain of rule 90 because $\Lambda_{22}^A \neq \Phi_{90}(\Lambda_{22}^B)$ and $\Lambda_{22}^B \neq \Phi_{90}(\Lambda_{22}^A)$. Thus, rule 22 over its domain does not fully emulate rule 90, they only agree every other time step. Given similar terminology used elsewhere, we may call Λ_{22} a *quasidomain* of rule 90.

Table 5.4 gives the 2^{nd} power of the rule 22 lookup table as well as that for rule 90. It also gives the linearization to rule 90 as well as the restriction of rule 22 to Λ_{22} at the 2^{nd} power, where the first linearization of this subdynamic occurs.

As with $\Lambda_{0,\Sigma}$ and Λ_{even}, rule 22's domain linearizes to the additive rule 90. Rule 90 produces the mod-2 Pascal triangle spacetime patterns characteristic of many chaotic CAs. It is well known that the sum-mod-2 of the neighborhood outer bits is the mechanism that generates these patterns. Since $\Lambda_{0,\Sigma}$ and Λ_{even} are a subset of rule 90's behaviors they also exhibit the mod-2 Pascal triangles, as can be seen in Figures 5.4, 5.5. Since the discovery of rule 22's domain, it was known that it produces similar Pascal triangle patterns. Though these are not exactly the same as rule 90's, since $\phi_{22|L(\Lambda_{22})}$ only emulates rule 90 every other time step. However, as far as we are aware, rule 22's domain linearization to rule 90 at every even power here is the first report of such a mechanism for the production of Pascal traingle patterns in rule 22.

η^2					$\phi^2_{90}(\eta^2)$	$\phi^2_{22}(\eta^2)$	$\phi^2_{22\leftrightarrow 90}(\eta^2)$	$\phi^2_{22\vert L(\Lambda_{22})}(\eta^2)$
1	1	1	1	1	0	0	0	–
1	1	1	1	0	1	0	–	–
1	1	1	0	1	0	0	0	0
1	1	1	0	0	1	1	1	1
1	1	0	1	1	0	0	0	0
1	1	0	1	0	1	1	1	–
1	1	0	0	1	0	0	0	–
1	1	0	0	0	1	1	1	1
1	0	1	1	1	0	0	0	0
1	0	1	1	0	1	0	–	–
1	0	1	0	1	0	1	–	–
1	0	1	0	0	1	0	–	–
1	0	0	1	1	0	0	0	–
1	0	0	1	0	1	0	–	–
1	0	0	0	1	0	0	0	0
1	0	0	0	0	1	1	1	1
0	1	1	1	1	1	0	–	–
0	1	1	1	0	0	0	0	0
0	1	1	0	1	1	0	–	–
0	1	1	0	0	0	1	–	–
0	1	0	1	1	1	1	1	–
0	1	0	1	0	0	0	0	–
0	1	0	0	1	1	0	–	–
0	1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1
0	0	1	1	0	0	1	–	–
0	0	1	0	1	1	0	–	–
0	0	1	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1
0	0	0	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1
0	0	0	0	0	0	0	0	0

Table 5.4. Second-order lookup tables for rule 90 (ϕ^2_{90}) and rule 22 (ϕ^2_{22}), as well as for ϕ^2_{22} linearized to ϕ^2_{90} ($\phi^2_{22\leftrightarrow 90}$) and ϕ^2_{22} restricted to its domain ($\phi^2_{22|L(\Lambda_{22})}$). The leftmost column gives all second-order ECA neighborhood values (that is, all radius-2 neighborhood values) in lexicographical order. Each subsequent column is the output of the neighborhoods for the specified dynamic or subdynamic. The symbol – indicates that lookup table element is excluded from the respective subdynamic.

5.7 A Non-Additive Domain

Now we come to an example in the class ($\mathcal{A} = \{0, 1\}, R = 2$) that breaks the connection between domains and additive subdynamics we have seen so far. It possesses a domain that admits no linearizations.

The CA in question is radius-2 rule 2614700074, named according to the same numbering scheme used for ECAs. This was previously studied by Crutchfield and Hanson [116]. They designed it to have the $\Lambda_{0,\Sigma}$ domain along with another structurally distinct domain—the $\Lambda_{1,1,0,\Sigma}$ domain. This domain has a single temporal phase consisting of the sofic shift $\mathcal{X}_{1,1,0,\Sigma}$ with strings of the form $\cdots 110\Sigma110\Sigma110\Sigma\cdots$, where Σ is a wildcard that can be either 1 or 0. The machine for $\Lambda_{1,1,0,\Sigma}$ is shown in Figure 5.8(c).

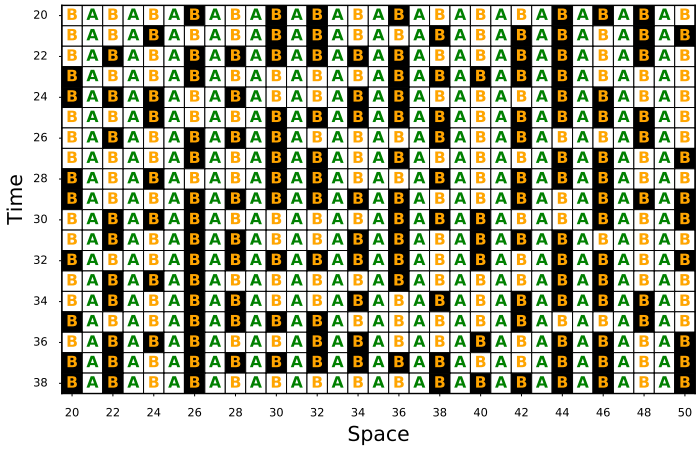
Reference [116] showed that $\Lambda_{0,\Sigma}$ and $\Lambda_{1,1,0,\Sigma}$ have distinct statistical signatures. Here, we investigate these differences via local causal states. Filtered spacetime fields for the $\Lambda_{0,\Sigma}$ and $\Lambda_{1,1,0,\Sigma}$ domains are shown in Figure 5.8(a) and (b), respectively. In each, as above, the colored letters represent the local causal state label at each site. These local causal state overlay diagrams clearly demonstrate that the domains have different spacetime symmetry groups. $\Lambda_{1,1,0,\Sigma}$ has recurrence time $\widehat{p} = 1$, temporal period $p = 2$, and spatial period 4, while $\Lambda_{0,\Sigma}$ has recurrence time $\widehat{p} = 1$, temporal period $p = 1$, and spatial period $s = 2$ for rule 2614700074.

Recall that $\Lambda_{0,\Sigma}$ is a domain invariant set of ECA rule 90. It is thus also a domain invariant set of Φ^2_{90}, which is itself a CA in the class ($\mathcal{A} = \{0, 1\}, R = 2$): $\mathbf{a} = (1, 0, 0, 0, 1)$. From this we can understand the $\Lambda_{0,\Sigma}$ domain of rule 2614700074 from the same combinatorial perspective as $\Lambda_{0,\Sigma}$ with rule 18. The restriction $\mathrm{LUT}(\phi^2_{90|L(\Lambda_{0,\Sigma})})$ leaves several output values unconstrained for assignment to construct a full CA lookup table. One such assignment gives rule 2614700074. As such, we find that rule 2614700074 linearizes to $\mathbf{a} = (1, 0, 0, 0, 1)$ at all powers:

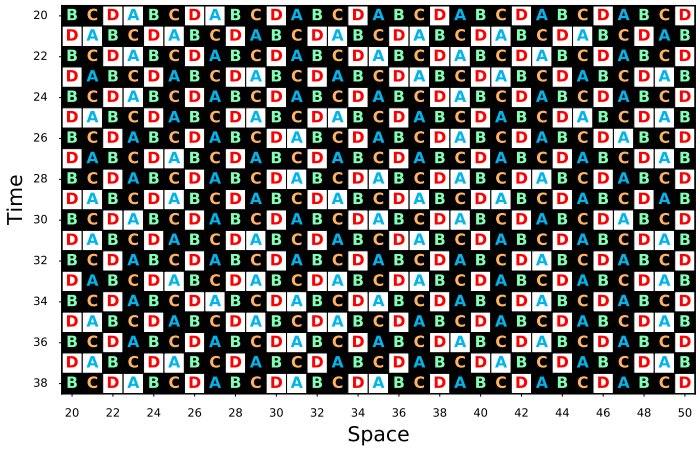
$$\mathrm{LUT}(\phi^n_{2614700074|L(\Lambda_{0,\Sigma})}) \subset \mathrm{LUT}(\phi^n_{2614700074 \leftrightarrow \mathbf{a}=(1,0,0,0,1)}),$$

for $n = 1, 2, 3, \ldots$.

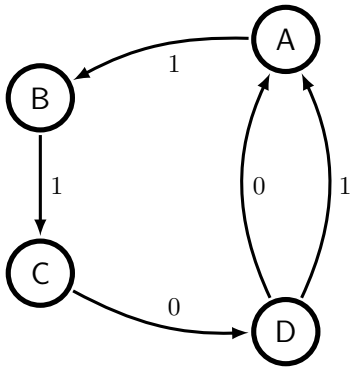
This connection to rule 90 also explains why $\Lambda_{0,\Sigma}$ has temporal period $p = 2$ for rule 18, but temporal period $p = 1$ for CA 2614700074. The local causal state field of $\Lambda_{0,\Sigma}$ for



(a) Filtered $\Lambda_{0,\Sigma}$ domain with associated local causal states.



(b) Filtered $\Lambda_{1,1,0,\Sigma}$ domain with associated local causal states.



(c) Machine for the domain invariant language $L(\Lambda_{1,1,0,\Sigma})$.

Figure 5.8. Filtered spacetime fields of the two domains of the $(\mathcal{A} = \{0,1\}, R = 2)$ CA rule 2614700074. Format and notation similar to previous such diagrams. (a) The $\Lambda_{0,\Sigma}$ domain has the same invariant set language of ECA rule 90. Its machine is shown in Figure 5.4(a)). (b) The $\Lambda_{1,1,0,\Sigma}$ domain. (c) Machine for $L(\Lambda_{1,1,0,\Sigma})$.

η					$\phi_{2614700074\|L(\Lambda_{1,1,0,\Sigma})}(\eta)$
1	1	1	0	1	0
1	1	1	0	0	1
1	1	0	1	1	1
1	1	0	0	1	1
1	0	1	1	1	1
1	0	0	1	1	1
0	1	1	1	0	0
0	1	1	0	1	1
0	1	1	0	0	0
0	0	1	1	0	0

Table 5.5. First-order lookup table of ($\mathcal{A} = \{0,1\}, R = 2$) CA 2614700074 restricted to its domain $\Lambda_{1,1,0,\Sigma}$. For simplicity, all elements of LUT($\phi_{2614700074}$) not in LUT($\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}$) are not shown.

rule 18 has a checkerboard symmetry. If one starts in state A at some spacetime point and moves forward one time step (i.e., applying Φ_{90}) one arrives at state B. However, starting in state A and moving forward two time steps (i.e., applying Φ_{90}^2) one ends in state A again.

The $\Lambda_{1,1,0,\Sigma}$ domain of rule 2614700074, in contrast to all other examples we have seen, does not appear to have any linearizations. From all examples we have seen and know of so far, we would expect rule 2614700074 to linearize over $\Lambda_{1,1,0,\Sigma}$ at its first power since it is a hidden symmetry domain with recurrence time $\widehat{p} = 1$: $\Phi_{2614700074} : \mathcal{X}_{1,1,0,\Sigma} \to \mathcal{X}_{1,1,0,\Sigma}$. Below we prove that the domain-restricted subdynamic of LUT($\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}$) cannot be additive. We also algorithmically checked for all possible linearizations of LUT($\phi^n_{2614700074|L(\Lambda_{1,1,0,\Sigma})}$) for $n = 1, 2, 3, 4$, finding none. For each $n \in \{1, 2, 3, 4\}$ we constructed the linearization LUT($\phi^n_{2614700074 \leftrightarrow \mathbf{a}}$) for all possible length $2nR + 1$ coefficient vectors $\mathbf{a}$ and found LUT($\phi^n_{2614700074|L(\Lambda_{1,1,0,\Sigma})}$) to be a subset of none of the linearizations.

Reference [116] showed, using the FME, that $\Lambda_{1,1,0,\Sigma}$ is a domain invariant set of rule 2614700074. First, we check that all 5-block words, i.e., all $R = 2$ neighborhoods η, are in the language $L(\Lambda_{1,1,0,\Sigma})$. To do so, consider the string $\cdots 110\Sigma110\Sigma110\Sigma\cdots$ and a sliding 5-block window over this string. This yields the 5-blocks $110\Sigma1$, $10\Sigma11$,

$0\Sigma110$, and $\Sigma110\Sigma$. Replacing each Σ with realizations 0 and 1 gives the neighborhoods in $L(\Lambda_{1,1,0,\Sigma})$, from which we can create $\mathrm{LUT}(\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})})$, shown in Table 5.5.

Now we show that there is no additivity assignment $\mathbf{a} = (a_{-2}, a_{-1}, a_0, a_1, a_2)$ such that $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(\eta)$ is given by $\mathbf{a} \overset{(\mathrm{mod}\ 2)}{\cdot} \eta$. From the first two rows of Table 5.5 we see that $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(11101) = 0$ and $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(11100) = 1$.
If $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(\eta)$ is additive, this shows the right-most entry of η must contribute to the additivity sum, and so we must have $a_2 = 1$. Similarly, from
$\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(01100) = 0$ and $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(11100) = 0$ we would need $a_{-2} = 1$. The neighborhoods 11001 and 11011 have the same output, giving $a_1 = 0$. Similarly, 10011 and 10111 have the same output, giving $a_0 = 0$, and 00110 and 01110 have the same output giving $a_{-1} = 0$.

Therefore, if there is an additivity assignment $\mathbf{a}$, it must be $\mathbf{a} = (1, 0, 0, 0, 1)$. However, we can see that $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(11011) = 1$, for example, and $(1, 0, 0, 0, 1) \overset{(\mathrm{mod}\ 2)}{\cdot} (1, 1, 0, 1, 1) \neq 1$. And so, $\phi_{2614700074|L(\Lambda_{1,1,0,\Sigma})}(\eta)$ cannot be additive.

■

5.8 Conclusion

The most basic ingredient of a cellular automaton, its lookup table, could not be simpler—a finite number of possible inputs are enumerated with their outputs explicitly specified. However, the overlapping interactions that occur when applying this simple lookup table synchronously for simultaneous global update of spatial configurations conspires to produce arbitrarily complex behaviors. The emergent complexity enshrouds a cellular automaton's simplicity, making it difficult to answer seemingly basic questions. Specifying a lookup table ϕ determines the global update Φ. Given a lookup table for ϕ, and hence Φ, what invariant sets are induced by Φ in the state space $\mathcal{A}^{\mathbb{Z}}$? In contrast with low-dimensional dynamical systems, the states $x \in \mathcal{A}^{\mathbb{Z}}$ of spatially-extended dynamical systems like CAs possess internal structure and live in infinite dimensions. For a given invariant set $\Lambda \subseteq \mathcal{A}^{\mathbb{Z}}$, is there a unifying structure in the states $x \in \Lambda$? Moreover, is there spacetime structure in the orbits of sequential states in Λ?

Our investigations provide several inroads to these questions, but they also highlight the challenge presented by complex spatially-extended dynamical systems and their emergent behaviors. While domain invariant sets appear to strongly correspond to the spacetime symmetries revealed by the local causal states in the orbit flows along the invariant sets, relating the invariant spatial shift spaces of Definition 1 with the resulting spacetime shift spaces of Definition 2 and their generalized symmetries remains unsolved. Since Φ is deterministic, the spacetime shift space that results from a given spatial shift space $\mathcal{X}$ is uniquely determined by Φ. This is not to say, though, as is often assumed, that the spacetime shift space trivially follows from Φ applied to $\mathcal{X}$. In fact, we still do not know how to fully characterize the spacetime shift space of orbits that follow from hidden symmetry domain invariant spatial shift spaces. In particular, we do not know how to properly define the domain temporal period for these spaces from their invariant spatial shift spaces. Such difficulties in understanding the spacetime shift spaces of hidden symmetry domains is perhaps one of the clearest examples of the fallacy of the "constructionist" hypothesis that often accompanies reductionism [133]. Tackling this will be a focus of subsequent investigations.

Beyond characterizing the spacetime shift spaces of domains, there remains the challenge of connecting domains, both the invariant spatial shift spaces and their resulting spacetime shift space of orbits, to the equation of motion Φ and the lookup table ϕ that generates it. Why does the particular assignment of lookup table outputs that form $\mathrm{LUT}(\phi_{18})$ generate the invariant set $\Lambda_{0,\Sigma}$? Why should $\mathrm{LUT}(\phi_{18|L(\Lambda_{0,\Sigma})})$ be additive when $\mathrm{LUT}(\phi_{18})$ is not? We have shown that this linear behavior of the nonlinear rule 18 actually follows from the combinatorics of $\Lambda_{0,\Sigma}$ being an invariant set of the additive rule 90. In fact, this is the mechanism behind every known stochastic ECA domain, including the enigmatic rule 22. Is this due to historical focus on rule 90? Or, is rule 90 particularly special? Beyond ECAs, we know from the $\Lambda_{1,1,0,\Sigma}$ domain of the $R = 2$ rule 2614700074 that this is not the only mechanism for generating hidden symmetry domains. Hidden symmetry domains need not be associated with an additive subdynamic. One possible path forward could be through the *partial permutivity* outlined in Ref. [223]; the per-

mutive subalphabets of examples 1.2 and 1.3 in Ref. [223] correspond to the domains of ECAs 18 and 22, respectively.

Perhaps in the sixteenth and seventeenth centuries — the burgeoning days of celestial mechanics — one could take for granted that knowing the equations of motion was tantamount to knowing the system and its behavior. Those days have long passed. Today, we appreciate that having the Navier-Stokes equations of hydrodynamics in hand does not translate into understanding emergent coherent structures, such as fluid vortices. The preceding recounted this lesson yet again. Even systems as "simple" as elementary cellular automata continue to hold surprises.

5.9 Appendices

5.9.1 Proof of Theorem 1

Theorem 1. Every nonzero linear CA Φ_β with $\mathcal{A} = \{0, 1\}$ is a factor map from the full-$\mathcal{A}$ shift to itself: $\Phi_\beta : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$.

Proof. For any $x \in \mathcal{A}^{\mathbb{Z}}$ we want to show that there exists a $y \in \mathcal{A}^{\mathbb{Z}}$ such that $x = \Phi_\beta(y)$. In other words, for a linear Φ_β there is always a right inverse $\widetilde{\Phi}_\beta$ such that $\Phi_\beta\big(\widetilde{\Phi}_\beta\,(x)\big) = x$.

First, decompose the string x into several 'basis' strings as follows; for each index i in x such that $x_i = 1$, create a basis string x^i for which $x^i_i = 1$ and $x^i_j = 0$ for all $j \neq i$. With this we have $x = \sum_i x^i$, which is shorthand for performing $x_j = \sum_i x^i_j \pmod 2$ at each index j of x.

If we can show the basis strings x^i always have a pre-image y^i, $x^i = \Phi_\beta(y^i)$, then it follows from linear superposition that $y = \sum_i y^i$ is the pre-image of x. If $x = \sum_i x^i = \sum_i \Phi_\beta(y^i)$, then using superposition, Equation (5.2), we have $\sum_i \Phi_\beta(y^i) = \Phi_\beta(\sum_i y^i)$. Thus, for any x there is a y such that $\Phi_\beta(y) = \Phi(\sum_i y^i) = \sum_i \Phi_\beta(y^i) = \sum_i x^i = x$.

We now need to show that for any basis string x^i with a single 1 there is always a pre-image y^i. This follows from additivity of ϕ_β. Since we are considering nonzero linear CAs there is at least one coefficient $a_i = 1$ in the additivity coefficient vector $\mathbf{a}$. In fact, it simplifies things to only consider CAs that have one or both of the outer bits of the neighborhood with a nonzero coefficient: $a_{-R} = 1$ or $a_R = 1$ or both. Any rule where this

is not the case is equivalent to one that is; e.g., the radius $R = 2$ CA $\mathbf{a} = (0, 1, 0, 1, 0)$ is equivalent to the radius $R = 1$ CA $\mathbf{a} = (1, 0, 1)$. (The only exception is the identity rule with $a_0 = 1$ and $a_i = 0$ for all $i \in \{-R, \ldots, 0, \ldots, R\} \backslash \{0\}$, but clearly every image is its own pre-image for the identity rule, so it is surjective).

Chose i to be either $-R$ or R such that $a_i = 1$. The neighborhood that has $x_i = 1$ and $x_j = 0$ for $j \in \{-R, \ldots, 0, \ldots, R\}$ and $j \neq i$ will necessarily output 1. To find the string that outputs the 3-block 010 there are three neighborhoods to consider, η_{-1}, η_0, and η_1. For the central neighborhood η_0, chose the same as above to still output 1. Neighborhood η_{-1} is to the left of η_0 and η_1 to the right. The outer neighborhoods overlap with η_0 and thus share all but one entry: $\eta_{-1}[-(R-1), \ldots, R] = \eta_0[-R, \ldots, R-1]$ and $\eta_1[-R, \ldots, R-1] = \eta_0[-(R-1), \ldots, R]$. Thus, we need only fill in $\eta_{-1}[-R]$ and $\eta_1[R]$. If $i = -R$, then $\eta_0[-R] = 1$ and $\eta_{-1}[-(R-1)] = 1$. And so, for η_{-1} to output a 0 we set $\eta_{-1}[-R] = 1$ if $a_{-R+1} = 1$ or $\eta_{-1}[-R] = 0$ if $a_{-R+1} = 0$. Meanwhile $\eta_1[R-1] = 0$, so for η_1 to output a 0 we must set $\eta_1[R] = 0$. If $i = R$ perform the symmetric operation: set $\eta_{-1}[-R] = 0$ and $\eta_1[R] = 1$.

To output the 5-block 00100 we similarly have two new neighborhoods to consider, η_{-2} and η_2. As above, we only need to fill in $\eta_{-2}[-R]$ and $\eta_2[R]$. If $i = -R$ we again just set $\eta_2[R] = 0$. Now, simply set $\eta_{-2}[-R]$ to either 0 or 1 so that $\mathbf{a} \cdot \eta_{-2} = 0$. Perform the similar symmetric construction if $i = R$. We can continue in this way to extend out to output arbitrary blocks $\cdots 000 \cdots 1 \cdots 000 \cdots$ with a single 1. Thus, we showed how to construct a pre-image y^i for any basis string $x^i = \cdots 000 \cdots 1 \cdots 000 \cdots$.

All that remains is the case of the all-0 string as an image. Clearly, though, from additivity, Equation (5.1), the all-0 string is its own pre-image for all linear CAs.

If Φ_β is surjective over all finite blocks, as we just showed, then Φ_β is necessarily surjective over $\mathcal{A}^{\mathbb{Z}}$. Thus, $y = \sum_i y^i = \widetilde{\Phi}_\beta(x)$ for all $x \in \mathcal{A}^{\mathbb{Z}}$.

■

Chapter 6

Cellular Automata: Coherent Structures

While much progress has been made in understanding the instability mechanisms driving pattern formation and the dynamics of the patterns themselves in idealized nonequilibrium phase transitions [29, 1, 30, 31], many challenges remain, especially with wider classes of real world patterns. In particular, the inescapable inhomogeneities of systems found in nature give rise to relatively more localized patterns, rather than the cellular patterns captured by simple Fourier modes. We refer to these localized patterns as *coherent structures*. There has been intense interest recently in coherent structures in fluid flows, including structures in geophysical flows [225, 117], such as hurricanes [13, 226], and in more general turbulent flows [62].

A principled universal description of the organization of such structures does not exist. So, while we can exploit vast computing resources to simulate models of ever-increasing mathematical sophistication, analyzing and extracting insights from such simulations becomes highly nontrivial. Indeed, given the size and power of modern computers, analyzing their vast simulation outputs can be as daunting as analyzing any real physical experiment [48]. Finally, there is no unique, agreed-upon approach to analyzing and predicting coherent material structures in fluid flows, for instance [113]. Even today ad hoc thresholding is often used to identify extreme weather events in climate data, such as cyclones and atmospheric rivers [227, 228, 229]. Developing a principled, but general mathematical

description of coherent structures is our focus.

Parallels with contemporary machine learning are worth noting, given the increasing overlap between these technologies and the needs of the physical sciences. Imposing Fourier modes as templates for cellular patterns is the mathematical analog of the technology of (supervised) *pattern recognition* [230]. Patterns are given as a finite number of classes and learning algorithms are trained to assign inputs into these classes by being fed a large number of labeled training data, which are inputs already assigned to the correct pattern class.

Computational mechanics, in contrast, makes far fewer structural assumptions [104]. As outlined in Section 3.2.2, for discrete spatially extended systems it makes only modest yet reasonable assumptions about the existence and conditional stationarity of lightcones in the orbit space of the system. In so doing, it facilitates identifying representations that are *intrinsic* to a particular system. This is in contrast with subjectively imposing a descriptional basis, such as Fourier modes, wavelets, or engineered pattern-class labels. We say that our subject here is not simply pattern recognition, but (unsupervised) *pattern discovery*.

This Chapter introduces the computational mechanics of coherent structures. The theory builds off the conceptual foundation laid out by DPID in which structures, such as particles and their interactions, are seen as deviations from spacetime shift-invariant domains. The new local causal state formulation differs from DPID in how domains and their deviations are formally defined and identified. The two distinct approaches to the same conceptual objective complement and inform one another, lending distinct insight into the patterns and regularity captured by the other.

Generalizing DPID particles, coherent structures are then formally defined as particular deviations from domains. Specifically, coherent structures are defined through *latent semantic fields*; either the local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ or the DPID domain-transducer filter, introduced in Section 4.4.1.1. For both of these, the latent field is endowed with the same coordinate geometry of the corresponding observable field, and each local point in spacetime carries a latent semantic variable (i.e. not a quantitative value).

CA coherent structures defined via the DPID domain-transducer filter are DPID particles. Defining particles using local causal states, in contrast, extends domain-particle-interaction analysis to a broader class of spatiotemporal systems for which DPID transducers do not exist. Due to this improvement, in the local causal states analysis we adopt the terminology of "coherent structures" over "particles".

Similar approaches using local causal states have been pursued by others [112, 231, 232, 233, 234]. However, as will be elaborated upon in future work, these underutilize computational mechanics, developing only a qualitative filtering tool—local statistical complexity—that assists in subjective visual recognition of coherent structures. Moreover, they provide no principled way to describe structures and thus cannot, to take one example, distinguish two distinct types of structures from one another. There have also been other unsupervised approaches to coherent structure discovery in cellular automata using information-theoretic measures [235, 236, 237, 238]. Recent critiques of employing such measures to determine information storage and flow and causal dependency [239, 240] indicate that these uses of information theory for CAs are still in early development and have some distance to go to reach the structure-detection performance levels presented here.

6.1 Structures as Domain Deviations

With domain regions and their symmetries established in Section 4.4, we now define coherent structures in spatiotemporal systems as spatially localized, temporally persistent broken symmetries. For clarity, the following definition is given for a single spatial dimension, but the generalization to arbitrary spatial dimensions is straightforward.

Definition 5. A *coherent structure* Γ is a contiguous nondomain region $\mathcal{R} \subset \mathcal{L} \times \mathbb{Z}$ of an observable spacetime field $\mathbf{x}$ such that $\mathcal{R}$ has the following properties in the latent semantic fields of $\mathcal{S} = \epsilon(\mathbf{x})$ or $T = \tau(\mathbf{x})$:

1. *Spatial locality*: Given a spatial configuration $\mathbf{x}_t$ at time t, Γ occupies the spatial region $\mathcal{R}_t = [i : j]$ if $\mathcal{S}_t^{i:j}$ is bounded by domain states on its exterior and contains nondomain states on its interior, $\mathcal{S}_t^{i-1} \in \Lambda$, $\mathcal{S}_t^i \notin \Lambda$, $\mathcal{S}_t^j \notin \Lambda$, and $\mathcal{S}_t^{j+1} \in \Lambda$.
2. *Lagrangian temporal persistence*: Given Γ occupies the localized spatial region $\mathcal{R}_t$ at time t, Γ persists to the next time step if there is a spatially localized set of nondomain states in $\mathcal{S}$ at time $t+1$ occupying a contiguous spatial region $\mathcal{R}_{t+1}$ that is within the depth-1 future lightcone of $\mathcal{R}_t$. That is, for every pair of coordinates $(r, t) \in \mathcal{R}_t$ and $(r', t+1) \in \mathcal{R}_{t+1}$, $||\mathbf{r}' - \mathbf{r}|| \leq c$.

For simplicity and generality we gave coherent structure properties in terms of local causal state fields. For CAs, to which the FME operator may be applied, the DPID transducer filter may similarly be used to identify coherent structures. However, the condition for temporal persistence is less strict: the regions $\mathcal{R}_{t+1}$ and $\mathcal{R}_t$, when given over T rather than $\mathcal{S}$, must have finite overlap. That is, there exists at least one pair of coordinates $(r, t) \in \mathcal{R}_t$ and $(r', t+1) \in \mathcal{R}_{t+1}$ such that $r = r'$. Coherent structures in CAs identified in this way are DPID particles. Both notions of temporal persistence are referred to as Lagrangian since they allow Γ to move through space over time.

Since local causal states are assigned to each point in spacetime, coherent structures of all possible sizes can be described. The smallest scale possible is a single spacetime point and the structure is captured by a single local causal state. Larger structures are given as

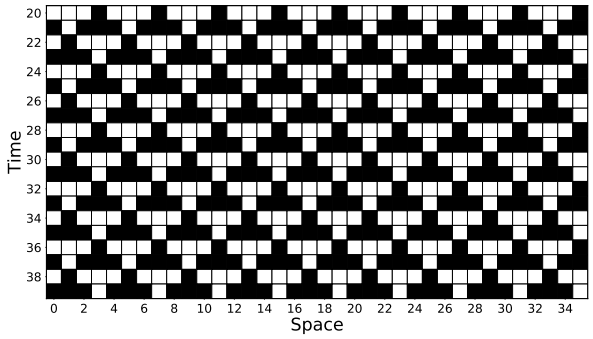
a set of states localized at the corresponding spatial scale. Such sets may be arbitrarily large and have (almost) arbitrary shape. In this way, the local causal states allow us to discover complex structures, without imposing external templates on the structures they describe. This leaves open the possibility of discovering novel structures that are not readily apparent from a observable spacetime field or do not fit into known shape templates.

We now apply the theory of domains and coherent structures to discover patterns in the spacetime fields generated by elementary cellular automata. For each domain class we analyze one exemplar ECA in detail. We begin describing the ECA's domain(s) and coherent structures generated by the ECA, from both the DPID and local causal state perspectives.

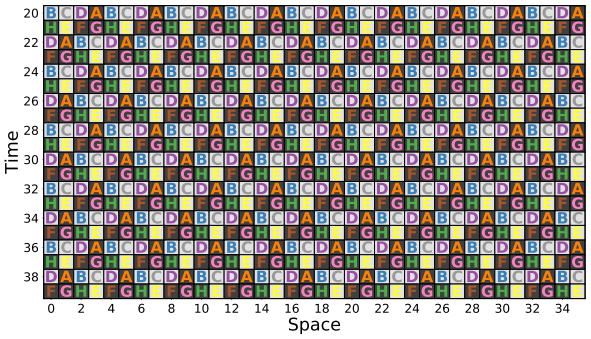
The analysis of domains and structures gives a sense of the correspondence between DPID and the local causal states; Conjecture 1. Though the CA dynamic Φ is not directly used to infer local causal states, the correspondence between DPID and local causal state domains shows that local causal states incorporate detailed dynamical features and they can be used to discover patterns and structures that can be defined directly from Φ using DPID.

6.2 Explicit Symmetry CAs

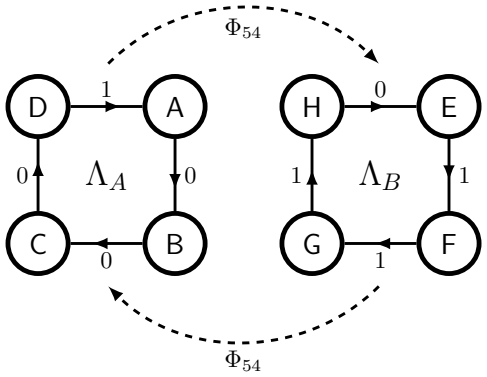
We start with a detailed look at ECA 54, whose domains and structures were worked out in detail via DPID [4]. ECA 54 was said to support "artificial particle physics" and this emergent "physics" was specified by the complete catalog of all its particles and their interactions. Here, we analyze the domain and structures using local causal states and compare. Since the particles (structures) are defined as deviations from a domain that has explicit symmetries, the resulting higher-level particle dynamics themselves are completely deterministic. As we will see later, this is not the case for hidden symmetry systems; stochastic domains give rise to stochastic structures.



(a) Pure domain spacetime field $\mathbf{x}_{\Lambda_{54}}$.



(b) Local causal state field $\mathcal{S}_{\Lambda_{54}} = \epsilon(\mathbf{x}_{\Lambda_{54}})$ atop $\mathbf{x}_{\Lambda_{54}}$.



(c) Finite-state machine $M(\Lambda_{54})$ of DPID domain language Λ_{54}.

Figure 6.1. ECA 54 domain: A sample pure domain spacetime field $\mathbf{x}_\Lambda$ is shown in (a). This field is repeated with the associated local causal states $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$ added in (b). Lightcone horizons $h^- = h^+ = 3$ were used. The DPID spacetime invariant set language is shown in (c). (Reprinted from Ref. [4] with permission.)

6.2.1 ECA 54's Domain

A pure-domain spacetime field $\mathbf{x}_\Lambda$ of ECA 54 is shown in Figure 6.1(a). As can be seen, it has explicit symmetries and is period 4 in both time and space. From the DPID perspective, though, it consists of two distinct spatial-configuration languages, $\Lambda_A = (0001)^*$ and $\Lambda_B = (1110)^*$, that map into each other under Φ_{54}; see Figure 6.1(c). This gives a recurrence time of $\widehat{p} = 2$. The finite-state machines, $M(\Lambda_A)$ and $M(\Lambda_B)$, shown there for these languages each have four states, reflecting the period-4 spatial translation symmetry: $s = 4$. Although the domain's recurrence time is $\widehat{p} = 2$, the raw states $\mathbf{x}_t$ are period 4 in time due to a spatial phase slip that occurs during their evolution: $p = 4$. This is shown explicitly in the spacetime machine given in Ref. [4]. We can see that the machine in Figure 6.1(c) fully describes the domain field in Figure 6.1(a). At some time t, the system is either in $(0001)^*$ or $(1110)^*$ and at the next time step $t + 1$ it switches,

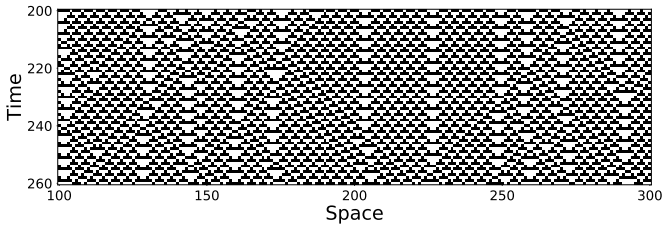
then back again at $t+2$, and so on.

Let's compare this with the local causal state analysis. The corresponding local causal state field $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$ was generated from the pure domain field $\mathbf{x}_\Lambda$ of Figure 6.1(a) via causal filtering; see Figure 6.1(b). We reiterate here that this reconstruction in no way relies upon the invariant set languages of Λ_{54} identified in DPID. Yet we see that the local causal states correspond exactly to $M(\Lambda_{54})$'s states. In total there are eight states, and these appear as two distinct tilings in the field. These tilings correspond to the two temporal phases of Λ_{54}: $w_A = [\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}] = \Lambda_A$ and $w_B = [\mathsf{E}, \mathsf{F}, \mathsf{G}, \mathsf{H}] = \Lambda_B$. At any given time t, a spatial configuration is tiled by only one of these temporal phases, which each consist of 4 states, giving a spatial period $s = 4$. And, at the next time $t+1$ there are only states from the other tiling. Then back to previous tiling, and so, the evolution continues. Thus, we can see the recurrence time is $\widehat{p} = 2$. In contrast, the actual local causal states are temporally period $p = 4$, which is also the orbit period of configurations in $\mathbf{x}_\Lambda$, as can be seen in Figure 6.1(a). This is in agreement with DPID's invariant set analysis, shown in Figure 6.1(c). As noted before and as will be emphasized, there is a strong correspondence between DPID's dynamically invariant sets of spatially homogeneous configurations and the local causal state description, both for coherent structures and the domains from which they are defined.

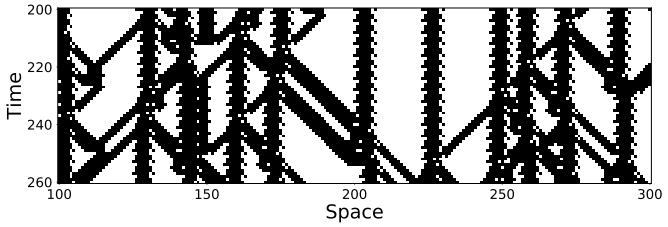
6.2.2 ECA 54's Structures

Let's examine the structures (particles) supported by ECA 54 and their interactions. Rule 54 organizes itself into domains and structures when started with random initial conditions. A sample spacetime field $\mathbf{x}$ produced by evolving a random binary configuration under Φ_{54} is shown in Figure 6.2(a). We first give a qualitative comparison of the structures in this field from both the DPID and local causal state perspectives.

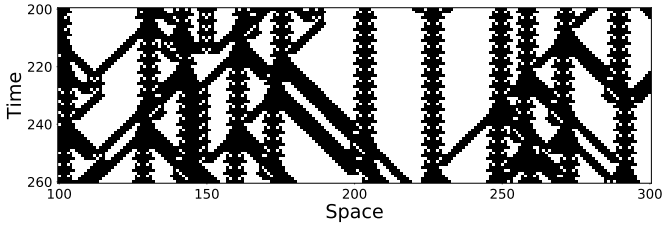
From the DPID side, a simple domain-nondomain filter is used with binary outputs that flag sites in transducer filter field $T = \tau(\mathbf{x})$ as either domain (white) or not domain (black). Applying this filter to the spacetime field of Figure 6.2(a) generates the diagram shown in Figure 6.2(b). Similarly, a domain-nondomain filter built from local causal states when applied to Figure 6.2(a) gives the output shown in Figure 6.2(c). For this filter, the



(a) Raw spacetime field.



(b) DPID domain-nondomain filter.



(c) Local causal state domain-nondomain filter.

Figure 6.2. Overview of ECA 54 structures: (a) A sample spacetime field evolved from a random initial configuration. (b) A filter that outputs white for cells participating in domains and black otherwise, using the DPID definition of domain. (c) The analogous domain-nondomain filter that uses the local causal state definition of domain. Lightcone horizons $h^- = h^+ = 3$ were used.

eight domain local causal states in $\mathcal{S} = \epsilon(\mathbf{x})$ are in white and all other local causal states black. While domain-nondomain detections differ site-by-site, we see that in aggregate there is again strong agreement on the structures identified by the two filter types.

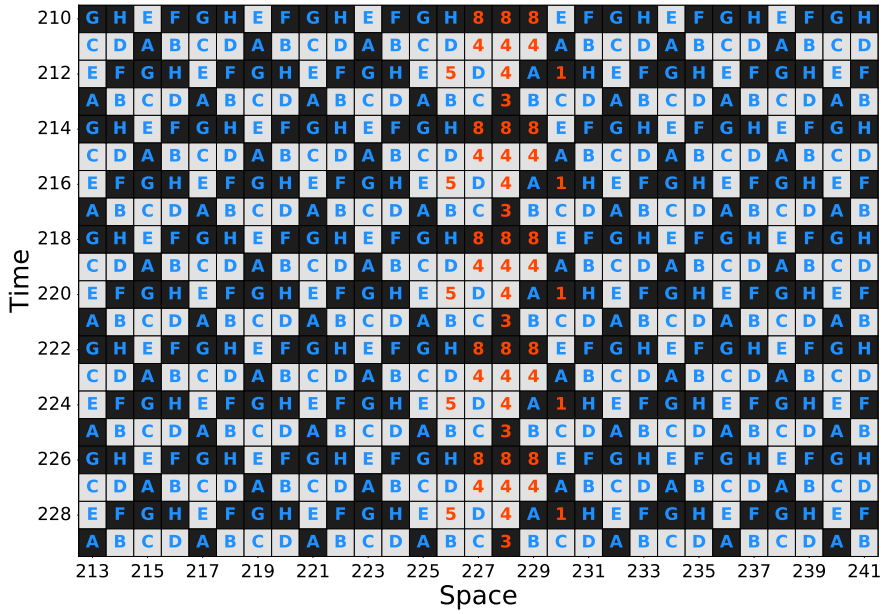
There are four types of particles found in ECA 54 [4], which we can now examine in detail. Before doing so, we must make a comment about the domain transducer τ used by DPID to identify structures. As mentioned, a stack automaton is generally required, but may be well-approximated with a finite-state transducer [199]. A trade-off is made with the transducer, however, since it must choose a direction to scan configurations—left-to-right or right-to-left. To best capture the proper spatial extent of a particle, an interpolation may be done by comparing right and left scans. This was done in the

domain-nondomain filter of Figure 6.2(b). The bidirectional interpolation used does not capture fine details of domain deviations. For the particle analysis that follows, a single direction (left to right) scan is applied to produce each $T_t = \tau(\mathbf{x}_t)$ in $T = \tau(\mathbf{x})$. A noticeable side-effect of the single direction scan is that it covers only about half of any given particle's spatial extent. (This scan-direction issue simply does not arise in local causal state filtering.)

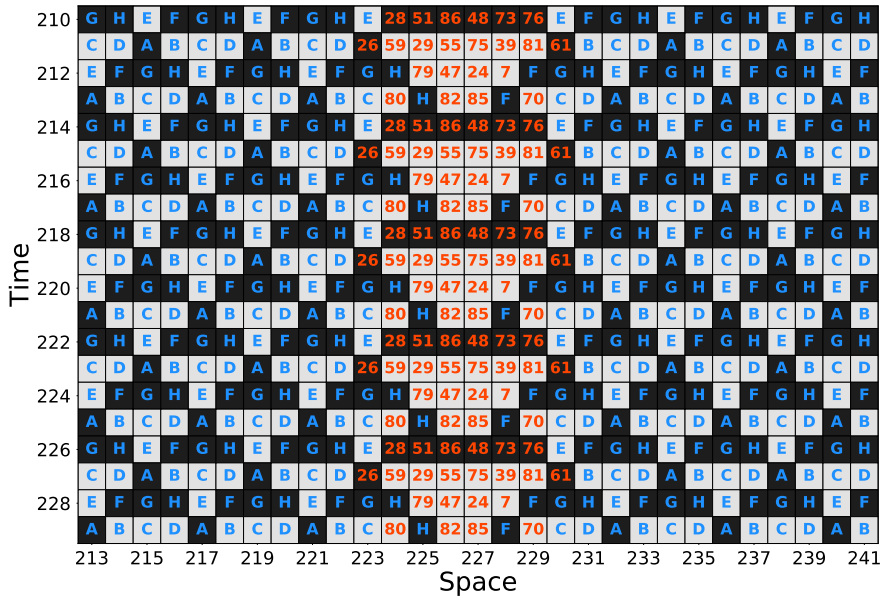
The first structure we analyze is the large stationary α particle, shown in Figure 6.3. For both overlay diagrams the white and black squares represent the values 0 and 1, respectively, of the observable ECA field $\mathbf{x}_\alpha$. Overlaid blue letters and red numbers are the latent semantic fields. In Figure 6.3(a) these come from the DPID domain transducer filtered field $T = \tau(\mathbf{x}_\alpha)$. In Figure 6.3(b) they come from the local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\alpha)$.

For the DPID domain transducer filtered field in Figure 6.3(a), overlaid blue letters are sites flagged as participating in domain by the transducer τ, with the letter representing the spatial phase of the domain as given by $M(\Lambda_{54})$. Red numbers correspond to sites flagged as various deviations from domain [4]. Here, the collection of such deviations outlines the α particle's structure; though, as stated above, the unidirectional transducer only identifies about half of the particle's spatial extent. The main feature to notice is that the particle has a period-4 temporal oscillation. As the α is recognizable by eye from the observable field values, one can see this period-4 structure is intrinsic to the observable spacetime field and not an artifact of the domain transducer. However, the period-4 temporal structure is clearly displayed by the DPID domain transducer description of α.

Figure 6.3(b) displays the local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\alpha)$; the eight domain states are given as blue letters, following Figure 6.1(b), and all other nondomain states, which outline the α, are red numbers. We see the local causal states fill out the α's full spatial extent. Since the numeric labels for each state are arbitrarily assigned during reconstruction, the α's spatial reflection symmetry that is clearly present does not appear in the local causal state labels. However, the underlying lightcones that populate the equivalence classes of these states *do* exhibit this symmetry. As with the DPID domain transducer description



(a) DPID domain transducer $T = \tau(\mathbf{x}_\alpha)$ atop $\mathbf{x}_\alpha$.



(b) Local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\alpha)$ atop $\mathbf{x}_\alpha$.

Figure 6.3. ECA 54's α particle: In both (a) and (b) white (0) and black (1) squares display the underlying ECA spacetime field $\mathbf{x}_\alpha$. (a) The DPID domain transducer filter $T = \tau(\mathbf{x}_\alpha)$ output is overlaid atop the spacetime field values of $\mathbf{x}_\alpha$. Blue letters are sites participating in domain and red numbers are particular deviations from domain. (b) The local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\alpha)$. The eight domain states are given by blue letters, all others by red numbers. In both diagrams, the non-domain sites outline the α particle of rule 54, according to the two different semantic fields. Lightcone horizons $h^- = h^+ = 3$ were used.

though, the local causal states properly capture the α's temporal period-4.

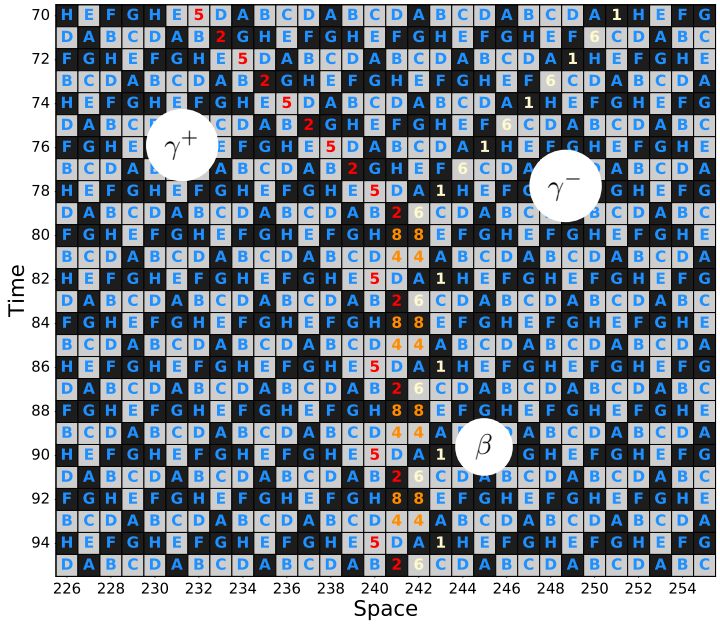
We emphasize that coherent structures are behaviors of the underlying system and, as such, they exist in the system's observable spacetime field. The latent semantic fields are formal methods that identify sites in the observable spacetime field which participate in a particular structure. This is made possible through the shared coordinate geometry between the observable field and the latent semantic fields, and is how overlay diagrams, like Figure 6.3, derive their utility.

We discuss the three remaining structures of ECA 54 by examining an interaction among them; the left-traveling γ^- particle can collide with the right-traveling γ^+ particle to form the β particle. This interaction is displayed with overlay diagrams in Figure 6.4. The values of the underlying field $\mathbf{x}_\beta$ are given by white (0) and black (1) squares. The DPID domain transducer filter field $T = \tau(\mathbf{x}_\beta)$ is overlaid over top of $\mathbf{x}_\beta$ in Figure 6.4(a) and the local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\beta)$ atop $\mathbf{x}_\beta$ in Figure 6.4(b).

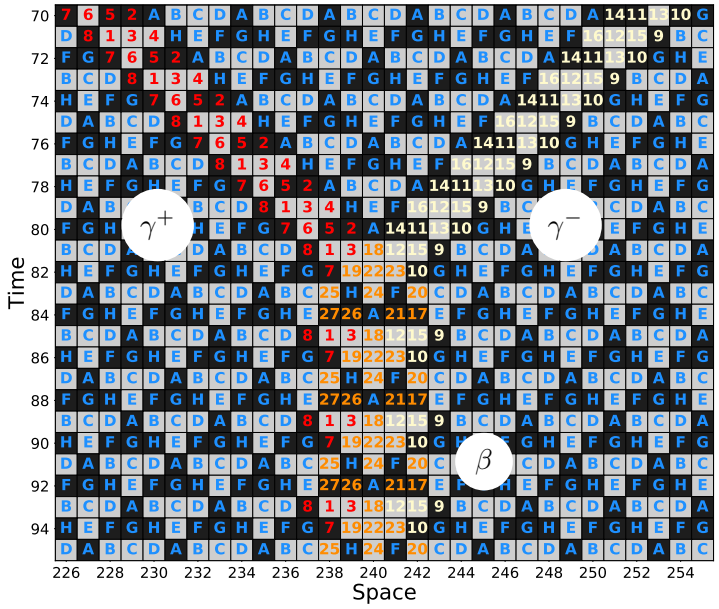
In both cases, the color scheme is as follows. Sites identified by the semantic fields as participating in a domain are colored blue, with the letters specifying the particular phase of the domain. In Figure 6.4(a) the domain phases are specified by T and in Figure 6.4(b) they are specified by S. And, as we saw in Figure 6.1 and can see here, these specifications of Λ_{54} are identical. For both Figures 6.4(a) and 6.4(b), nondomain sites participating in the γ^+ are flagged with red, those participating in the γ^- with yellow, and those uniquely participating in the β with orange.

As with the α particle, the local causal state description better covers the particles' spatial extent, but both filters agree on the temporal oscillations of each particle. Both γs are period 2 and β is period 4. Unlike the α and β, the γ particles are not readily identifiable by eye. They arise as a result of a phase slip in the domain. For example, a spatial configuration with a γ present is of the form $\Lambda_A\ \gamma\ \Lambda_B$.

Related to this, we point out here an observation about this interaction that illustrates how our methods uncover structures in spatiotemporal systems. At the top of each diagram in Figure 6.4 the spatial configurations are of the form $\Lambda_A\ \gamma^+\ \Lambda_B\ \gamma^-\ \Lambda_A$. At each subsequent time step, the domains change phase $A \to B$ and $B \to A$ and the



(a) DPID domain transducer $T = \tau(\mathbf{x}_\beta)$ atop $\mathbf{x}_\beta$.



(b) Local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\beta)$ atop $\mathbf{x}_\beta$.

Figure 6.4. ECA 54's $\gamma^+ + \gamma^- \rightarrow \beta$ interaction: In both diagrams the white (0) and black (1) squares display the underlying ECA spacetime field $\mathbf{x}_\beta$. (a) The DPID domain transducer filter $T = \tau(\mathbf{x}_\beta)$ output is overlaid atop the spacetime field values of $\mathbf{x}_\beta$. Blue letters are sites identified by T as participating in the domain. Colored numbers are sites identified as participating in one of the three remaining structures. The γ^+ particle is outlined only by red numbers, γ^- by yellow numbers, and β by a combination of red, yellow, and orange. (b) The local causal state field $\mathcal{S} = \epsilon(\mathbf{x}_\beta)$ is overlaid atop $\mathbf{x}_\beta$. The eight domain states are in blue, and the other nondomain states are colored the same as in (a). Lightcone horizons $h^- = h^+ = 3$ were used.

intervening domain region shrinks as the γs move towards each other. The intervening domain disappears when the γs finally collide. Then we have local configurations of the form $\Lambda_A\ \beta\ \Lambda_A$. However, there is an indication that a phase slip between these domain regions still happens "inside" the β particle. Notice in Figure 6.4 there are several spatial configurations (horizontal time slices) in which domain states appear inside the β that are the opposite phase of the bordering domain phases, indicating a phase slip. Also, the states constituting the γs are found as constituents of the β. For the DPID domain transducer τ, each γ consists of just two states, and all four of these states (two for each γ) are found in the β. In the local causal state field S, each γ is described by eight local causal states. Not all of these show up as states of the β, but several do. Those γ states that do show up in the β appear in the same spatiotemporal configurations they have in the γs.

These observations tell us about the underlying ECA's behavior and so can be gleaned from the observable spacetime field itself. That said, the discovery that the β particle is a "bound state" of two γs and that it contains an internal phase slip of the bordering domain regions is not at all obvious from inspecting observable spacetime fields. That is, $\gamma^+\ \Lambda\ \gamma^- \rightarrow \beta$. Such structural discovery, however, is greatly facilitated by the coherent structure analysis. To emphasize, these insights concern the intrinsic organization embedded in the spacetime fields generated by the ECA. No structural assumptions, beyond the very basic definitions of local causal states, are required.

Let's recapitulate the correspondence between the independent DPID and local causal state descriptions of the ECA 54 domain and structures. From the DPID perspective, the ECA 54 domain Λ_{54} consists of two homogeneous spatial phases that are mapped into each other by Φ_{54}. In contrast, Λ_{54} is described by a set of local causal states with a spacetime translation symmetry tiling. The two descriptions agree completely, giving a spatial period 4, temporal period 4, and recurrence time of 2. On the one hand, for ECA 54's structures DPID directly uses domain information to construct a transducer filter $T = \tau(\mathbf{x})$ that identifies structures as groupings of particular domain deviations. On the other, the local causal states are assigned uniformly to spacetime field sites via

causal filtering $\mathcal{S} = \epsilon(\mathbf{x})$. Domains and sites participating in a domain are found by identifying spatiotemporal symmetries in the local causal states. Coherent structures are then localized deviations from these symmetries. Though the agreement is not exact as with the domain, DPID and the local causal states still agree to a large extent on their descriptions of ECA 54's four particles and their interactions.

6.2.3 ECA 110

As the most complex explicit symmetry ECA, rule 110 is worth another mention (see Section 1.6 for further discussion on the significance of ECA rule 110). It is the only ECA proven to support universal computation (on a specific subset of initial configurations) and implements this using a subset of the ECA's coherent structures [136]. This was shown by mapping ECA 110's particles and their interactions onto a cyclic tag system that emulates a Post tag system which, in turn, emulates a universal Turing machine. A domain-nondomain filter reveals several of ECA 110's particles used in the implementation; see Figure. 1.6. The ECA 110 domain was displayed in Figures. 4.1(a) and 4.1(c), as the example for explicit symmetry domains. The domain has a single phase, rather than two phases like ECA 54's, and requires 14 states, as opposed to ECA 54's combined 8. The ECA 110's highly complex behavior surely derives from the heightened complexity of its domain. Exactly how, though, remains an open problem.

6.3 Hidden Symmetry CAs

Our attention now turns to ECAs with hidden symmetries and stochastic domains. These are the so-called "chaotic" ECAs. Since the structure of an ECA's domain heavily dictates the overall behavior, stochastic domains give rise to stochastic structures and hence, in combination, to an overall stochastic behavior. To be clear, since all ECA dynamics are globally deterministic—the evolution of spatial configurations is deterministic—the stochasticity here refers to local structures rather than global configurations. In contrast to explicit symmetry ECAs whose structures are largely identifiable from the observable spacetime field, the structures found in stochastic-domain ECAs are often not at all apparent. In this case the ability of our methods to facilitate the discovery and description

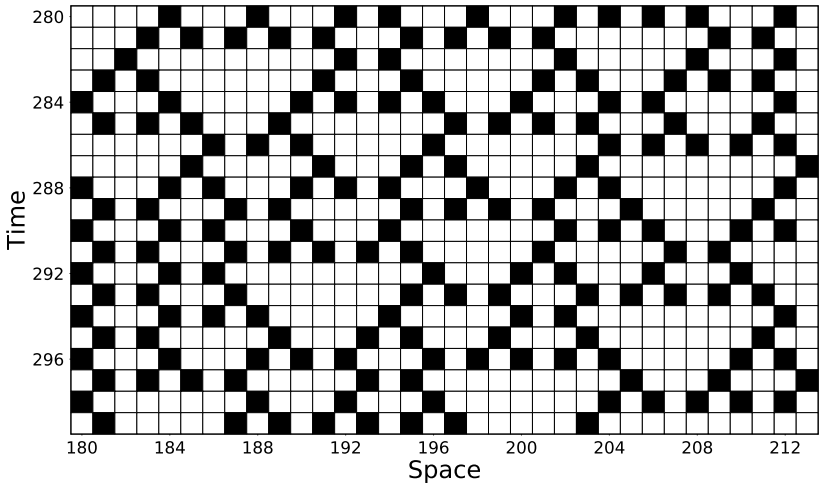
of such hidden structures is all the more important and sometimes even necessary. While the distinction between stochastic and explicit symmetry domains does not make a difference when determining DPID's spacetime invariant sets, local causal state inference is relatively more difficult with stochastic domains, usually requiring large lightcone depths and an involved domain-structure analysis.

Here, we examine ECA 18 in detail, as its stochastic domain is relatively simple and well understood. An empirical domain-structure analysis of ECA 18 was first given in Ref. [217] and then more formally in Refs. [216, 241, 242, 243], which notes the domain's temporal invariance. It was not until the FME was introduced in Ref. [5] that this was rigorously proven and shown to follow within the more general DPID framework. The distinguishing feature of ECA 18's domain observed in the early empirical analysis was that the lookup table ϕ_{18} becomes *additive* when restricted to domain configurations. Specifically, when restricted to domain, ϕ_{18} is equivalent to ϕ_{90}, which is the sum mod 2 of the outer two bits of the local neighborhood; $\mathbf{x}_{t+1}^r = \phi_{90}(\mathbf{x}_t^{r-1}\mathbf{x}_t^r\mathbf{x}_t^{r+1}) = \mathbf{x}_t^{r-1} + \mathbf{x}_t^{r+1} \pmod 2$. This was detailed above in Chapter 5.

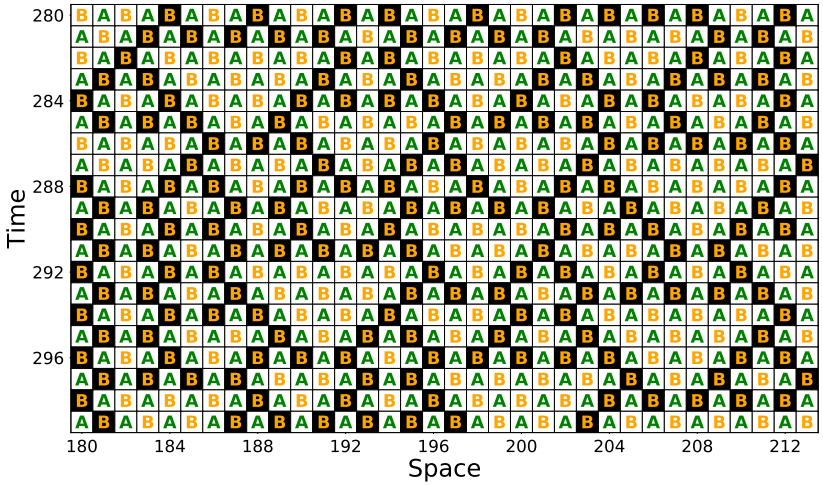
ECA 18's structures illustrate additional complications of local causal state analysis with stochastic symmetry systems. Nondomain states of ECA 54 and other explicit symmetry ECAs always indicate a particle or particle interaction, after transients. This is not the case with chaotic ECAs, and our formal definition is needed to identify ECA 18's coherent structures.

6.3.1 ECA 18's Domain

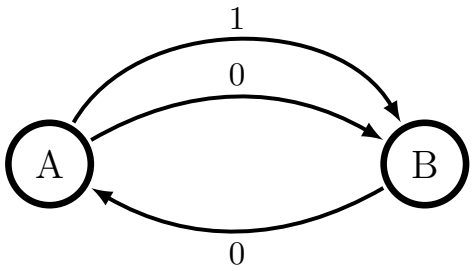
Iterates of a pure domain spacetime field $\mathbf{x}_{\Lambda_{18}}$ for the ECA 18 domain Λ_{18} is shown in Figure 6.5(a). White and black cells represent site values 0 and 1, respectively. A symmetry is not apparent (or indeed present) in the observable spacetime field. One noticeable pattern, though, is that 1s (black cells) always appear in isolation, surrounded by 0s on all four sides. This still does not reveal symmetry, since neither time nor space shifts match the original field. When scanning along one dimension, making either timelike or spacelike moves (vertically or horizontally), one sees that every other site is always a 0 and the sites in between are *wildcards*—they can be either 0 or 1. Making this



(a) Pure domain spacetime field $\mathbf{x}_\Lambda$.



(b) Local causal state field $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$.



(c) Finite-state machine $M(\Lambda_{18})$ of DPID domain language Λ_{18}.

Figure 6.5. ECA 18 domain: (a) Iterates of a sample pure domain spacetime field $\mathbf{x}_\Lambda$, white and black are values 0 and 1, respectively. (b) The same domain field with the local causal state field $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$ overlaid. Lightcone horizons $h^- = 8$ and $h^+ = 3$ were used. (c) The finite-state machine $M(\Lambda_{18})$ of the DPID invariant set language of the ECA 18 domain Λ_{18}. (Reprinted with permission from Ref [5].)

identification finally reveals the (stochastic) symmetry in the ECA 18 domain [5].

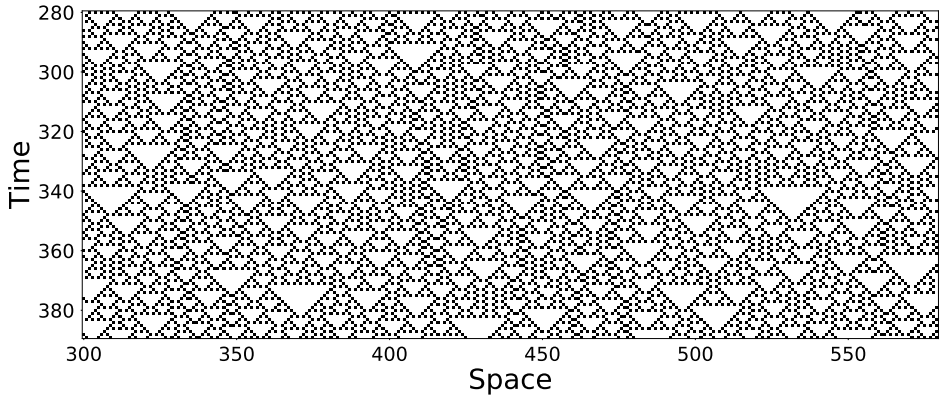
In contrast to this ad hoc description, the 0-wildcard pattern is clearly and immediately identified in the local causal state field $\mathcal{S}_\Lambda = \epsilon(\mathbf{x}_\Lambda)$, shown in Figure 6.5(b). State A occurs on the fixed-0 sites and state B on the wildcard sites. And, these states occur in a checkerboard symmetry that tiles the spacetime field. An interesting observation of this symmetry group is that it has rotational symmetry, in addition to the time and space translation symmetries. This is a rotation, though, in spacetime. While unintuitive at first, the above discussion shows this spacetime rotational symmetry is not just a coincidence. The 0-wildcard semantics applies for *both* spacelike and timelike scans through the field.

The DPID invariant-set language for this domain is given in Figure 6.5(c). Not surprisingly, this is the 0-wildcard language. It is easy to see that ϕ_{18} creates a tiling of 0-wildcard local configurations. Also, note the transition branching (the wildcard) leaving state A indicates a lack of translation symmetry. This identifies Λ_{18} as a stochastic symmetry domain. We again see a clear correspondence between the local causal state identification of the domain and that of DPID. Both give spatial period $s = 2$, temporal period $p = 2$, and recurrence time $\widehat{p} = 1$, as there is a single local causal state tiling and a single DPID spatial language, both corresponding to the 0-wildcard pattern.

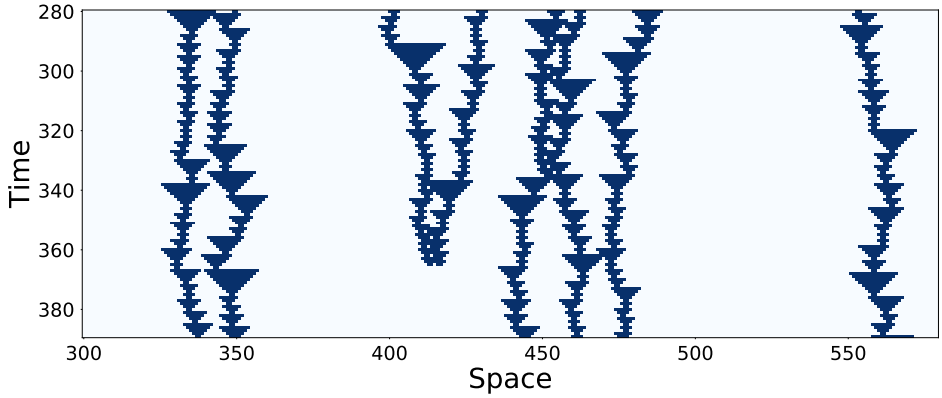
6.3.2 ECA 18's Structures

ECA 18's two-state domain Λ_{18} supports a single type of coherent structure—the α particle that appears as a phase-slip in the spatial period-2 domain and consists of local configurations $10^{2k}1$, $k = 0, 1, 2, \ldots$. The domain's stochastic nature drives the αs in an unbiased left-right random-walk. When two collide they pairwise annihilate; resolving each α's spatial phase shift. (To clarify, the α of ECA 18 has no relation to the α of ECA 54.)

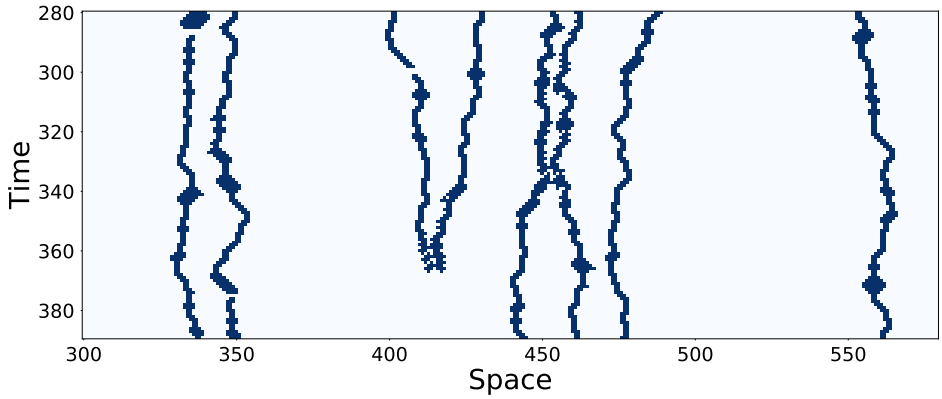
Figure 6.6 shows these structures as they evolve from a random initial configuration under Φ_{18}. The observable spacetime field is given in Figure 6.6(a) with the DPID transducer domain-nondomain filter (bidirectional scan interpolation) in Figure 6.6(b) and the local causal state domain-nondomain filter in Figure 6.6(c). With the aid of these domain



(a) Raw spacetime field



(b) DPID transducer domain-nondomain filter



(c) Local causal-state domain-nondomain filter

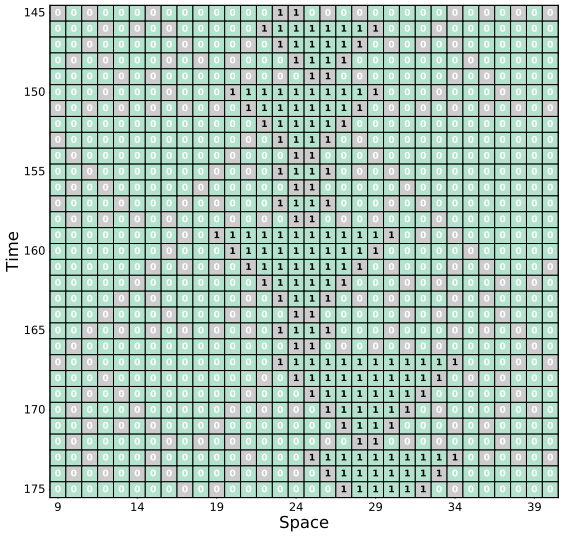
Figure 6.6. ECA 18 structures: (a) Sample spacetime field evolved under ECA 18 from a random initial configuration. (b) Spacetime field after filtering with domain regions in white and coherent structures in blue, using the DPID domain transducer. (c) Spacetime field filtered with domain regions in white and structures in blue, using local causal states. The occasional gap in the structures is an artifact of using finite-depth lightcones during reconstruction of local causal states. Lightcone horizons $h^- = 8$ and $h^+ = 3$ were used.

filters, visual inspection shows that ECA 18's structures are, in fact, pairwise annihilating random-walking particles. This was explored in detail by Ref. [194].

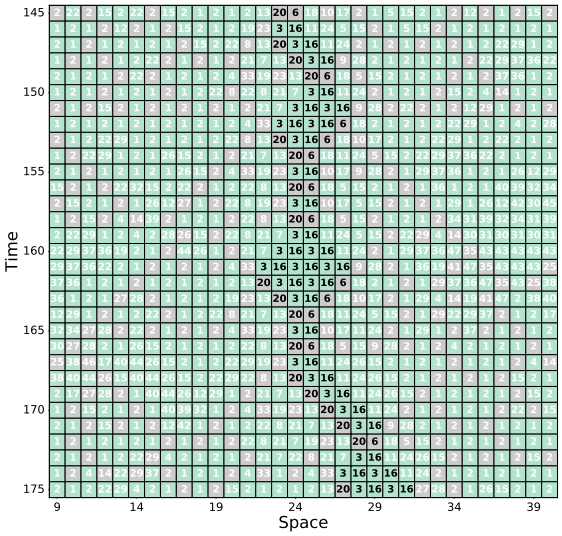
As noted above, the domain-structure local causal state analysis for stochastic domain systems is generally more subtle. In the DPID analysis, ECA 18 consists solely of the single domain and random-walking α particle structures. Thus, using the DPID transducer to filter out sites participating in domains leaves only α particles, as done in Figure 6.6(b). The situation is more complicated in the local causal state analysis. As described in more detail shortly, filtering out domain states leaves behind more than the structures. Why exactly this happens is the subject of future work. The field shown in Figure 6.6(c) was produced from a coherent structure filter, rather than from a domain-nondomain filter. There, local causal states that fit the coherent structure criteria are colored blue and all others are colored white.

To illustrate the more involved local causal state analysis let's take a closer look at the α particle. This also highlights a major difference between DPID and local causal state analyses. As the DPID transducer is strictly a spatial description it can identify structures that grow in a single time step to arbitrary size. One artifact of this is that the spatial growth can exceed the speed of local information propagation and thus make structures appear acausal. The local causal states, however, are constructed from lightcones and so naturally take into account this notion of causality. They cannot describe such acausal structures. Accounting for this, though, there is a strong agreement between the two descriptions.

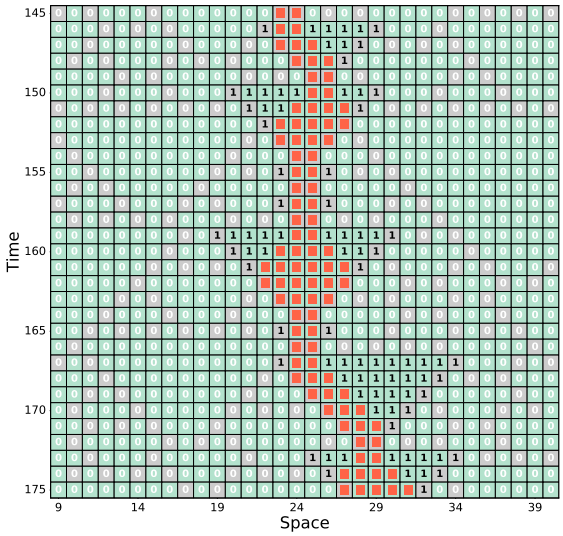
From the perspective of the DPID domain transducer τ ECA 18's α particles are simple to understand. From the domain language in Figure 6.5(c), the domain-forbidden words are those in the regular expression $1(00)^*1$. That is, pairs of ones with an even number of 0s (including no 0s) in between. This is the description of α particles at the spatial configuration level. The DPID bidirectional scan interpolation domain transducer perfectly captures α described this way; see Figure 6.7(a). To aid in visual identification we employed a different color scheme for Figure 6.7: the underlying ECA field values are given by green (0) and gray (1) squares. For the DPID transducer filtered field $T = \tau(\mathbf{x})$



(a) DPID domain transducer filter $T = \tau(\mathbf{x})$.



(b) Coherent structure filter over $\mathcal{S}$.



(c) Structure described by $\mathcal{S}$ compared to structure described by T.

Figure 6.7. Comparative analysis of ECA 18's α particle: In all three spacetime diagrams, the underlying ECA field values of 0 and 1 are represented as green and gray squares, respectively. (a) DPID domain transducer filtered field $T = \tau(\mathbf{x})$ with bidirectional scan interpolation. Domain sites are identified with white 0s and particle sites with black 1s. (b) A coherent structure causal filter; local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$ with nondomain local causal states states satisfying the coherent structure definition are colored black with all other states colored white. Lightcone horizons $h^- = 8$ and $h^+ = 3$ were used. (c) Comparison of the structures from the two methods: The DPID transducer filter of (a) with sites that have local causal states identified as the coherent structure in (b) given a red square label.

in Figure 6.7(a), overlaid white 0s identify domain sites and black 1s identify particle sites. Every local configuration identified as an α is of the form $1(00)^*1$. As noted above, however, αs described in this way can grow in size arbitrarily in a single time step as the number of pairs of zeros in $1(00)^*1$ is unbounded.

Local causal state inference—whether topological (Section 4.2) or probabilistic [110]—is *unsupervised* in the sense that it uses only raw spacetime field data and no other external information such as the CA rule used to create that spacetime data. Once states are inferred, further steps are needed for coherent structure analysis.

The first step is to identify domain states in the local causal state field $\mathcal{S} = \epsilon(\mathbf{x})$. They tile spacetime regions, i.e., domain regions. For explicit-symmetry domain ECAs this step is sufficient for creating a domain-structure filter. Tiled domain states can be easily identified and all other states outline ECA structures or their interactions. The situation is more subtle, however, for ECAs with stochastic domains.

For our purposes here, it suffices to strictly apply the definition of coherent structures after this first "out of the box" unsupervised causal filter. The initial unsupervised filtered spacetime diagram identifies a core set of states that are spatially localized and temporally persistent. A coherent structure filter then isolates these states by coloring them black and all other states white in the local causal state field $\mathcal{S}$. The output of this filter is shown in Figure 6.7(b). The growth rate of the structures identified in this way—by the local causal states—is limited by the speed of information propagation, which for ECAs is unity. Applying this growth-rate constraint on the DPID structure transducer, one again finds strong agreement. A comparison is shown in Figure 6.7(c). It shows the output of the DPID filter applied to the spacetime field of Figure 6.7(a) and, in red, sites corresponding to the structure according to the local causal states in Figure 6.7(b).

6.4 Remarks

Having laid out our coherent structure theory and illustrating it in some detail, it is worth looking back, as there are subtleties worth highlighting. The first is our use of the notion of *semantics*, which derives from the *measurement semantics* introduced in Ref. [197].

Performing causal filtering $\mathcal{S} = \epsilon(\mathbf{x})$ may at first seem counterproductive, especially for binary fields like those generated by ECAs, as the state space of the system is generally *larger* in $\mathcal{S}$ than in $\mathbf{x}$. As the local state space of ECAs is binary, complexity is manifest in how the sites interact and arrange themselves. Not all sites in the field play the same role. For instance, in ECA 110's domain, Figure 4.1(a), the 0s in the field group together to form a triangular shape. This triangle has a bottom-most 0 and a rightmost 0, but they are both still 0s. To capture the semantics of "bottom-most" and "rightmost" 0 of that triangle shape, a larger local state space is needed. And, indeed, this is exactly the manner in which the local causal states capture the semantics of the underlying field. We saw a similar example with the fixed-0 and wildcard semantics of Λ_{18}.

The values in the fields $\mathcal{S} = \epsilon(\mathbf{x})$ and $T = \tau(\mathbf{x})$ are not measures of some quantity, but rather semantic labels. For the local causal states, they are labels of equivalence classes of local dynamical behaviors. For the DPID domain transducer, they label sites as being consistent with the domain language Λ or else as the particular manner in which they deviate from that language.

This, however, is only the first level of semantics used in our coherent structure theory. While the filtered fields $\mathcal{S} = \epsilon(\mathbf{x})$ and $T = \tau(\mathbf{x})$ capture semantics of the original field $\mathbf{x}$, to identify coherent structures a new level of semantics on top of these filtered fields is needed. These are semantics that identify sites as domain or coherent structure using $\mathcal{S}$ and T. For the DPID domain transducer T, the domain semantics are by construction built into T. Our coherent structure definition adds the necessary semantics to identify collections of nondomain sites as participating in a coherent structure.

For the local causal states, one may think of the field $\mathcal{S} = \epsilon(\mathbf{x})$ as being the semigroup level of semantics. That is, they represent pattern and structure as generalized symmetries of the underlying field $\mathbf{x}$. This is the same manner in which the ϵ-machine captures pattern and structure of a stochastic process with semigroup algebra; see Section 3.1.6. The next level of semantics, used to identify domains, requires finding explicit symmetries in $\mathcal{S}$. Thus, domain semantics are the group-theoretic level of semantics, since domains are identified by spacetime translation symmetry groups over $\mathcal{S}$. With states participating

in those symmetry groups identified, our coherent structure definition again provides the necessary semantics to identify structures in $\mathbf{x}$ through $\mathcal{S}$. These remarks hopefully also clarify the interplay between group and semigroup algebras in our development.

Lastly, we highlight the distinction between a CA's local update rule ϕ and its global update Φ—the CA's equations of motion. For many CAs, as with ECAs, Φ is constructed from simultaneous synchronous application of ϕ across the lattice. In a sense, then, there is a simple relation between ϕ and Φ. However, as demonstrated by many ECAs, most notably the Turing complete ECA 110, the behaviors generated by Φ can be extraordinarily complicated, even though ϕ is extraordinarily simple. This is why complex behaviors and structures generated by ECAs are said to be *emergent*.

This point is worth emphasizing here due to the relationship between past lightcones and ϕ^i for CAs. Since the local causal states are equivalence classes of past lightcones, they are equivalence classes of the elements of ϕ^i for CAs. Thus, the system's local dynamic is directly embedded in the local causal states. As we saw, the local causal states are capable of capturing emergent behaviors and structures of CAs and so, in a concrete way, they provide a bridge between the simple local dynamic ϕ and the emergent complexity generated by Φ. Moreover, the correspondence between the local causal state and DPID domain-structure analysis shows the particular equivalence relation over the elements of ϕ^i used by the local causal states captures key dynamical features of Φ, used explicitly by DPID.

The relationship, though, between ϕ^i and Φ captured by the local causal states is not entirely transparent, as most clearly evidenced by the need for behavior-driven reconstruction of the local causal states. Given a CA lookup table ϕ, one may pick a finite depth i for the past lightcones and easily construct ϕ^i. It is not at all clear, however, how to use Φ to generate the equivalence classes over the past lightcones of ϕ^i that have the same conditional distributions over future lightcones. The only known way to do this is by brute-force simulation and reconstruction, letting Φ generate past lightcone-future lightcone pairs directly.

6.5 Conclusion

Two distinct, but closely related, approaches to spatiotemporal computational mechanics were reviewed: DPID and local causal states. From them, we developed a theory of pattern and structure in fully discrete dynamical field theories. Both approaches identify patterns as statistically-regular regions of a system's spatiotemporal behavior—its domains—generalizing patterns from exact symmetries. We then defined coherent structures as localized deviations from domains; i.e., coherent structures are locally-broken domain (generalized) symmetries.

The DPID approach defines domains as sets of homogeneous spatial configurations that are temporally invariant under the system dynamic. In 1+1 dimension systems, dynamically important configuration sets can be specified as particular types of regular language, i.e. sofic languages. Once these domain patterns are identified, a domain transducer τ can be constructed that filters spatial configurations $T_t = \tau(\mathbf{x}_t)$, identifying sites that participate in domain regions or that are the unique deviations from domains. Finding a system's domains and then constructing domain transducers requires much computational overhead, but full automation has been demonstrated. Once acquired, the domain transducers provide a powerful tool for analyzing emergent structures in discrete, deterministic 1+1 dimension systems. The theory of domains as dynamically invariant homogeneous spatial configurations is easily generalizable beyond this setting, but practical calculation of configuration invariant sets in more generalized settings presents enormous challenges.

The local causal state approach, in contrast, generalizes well. Both in theory and in practice, under a caveat of computational resource scaling (see Section 7.2). It is a more direct generalization of computational mechanics from its original temporal setting. The causal equivalence relation over pasts based on predictions of the future is the core feature of computational mechanics from which the generalization follows. Local causal states are built from a local causal equivalence relation over past lightcones based on predictions of future lightcones. Local causal states provide the same powerful tools of domain transducers, and more. Being equivalence classes of past lightcones, which in the

deterministic setting are the system's underlying local dynamic, local causal states offer a bridge between emergent structures and the underlying dynamic that generates them.

In both, patterns and structures are discovered rather than simply recognized. No external bias or template is imposed, and structures at all scales may be uniformly captured and represented. These representations greatly facilitate insight into the behavior of a system, insights that are intrinsic to a system and are not artifacts of an analyst's preferred descriptional framework. ECA 54's $\gamma^+ + \Lambda + \gamma^- \rightarrow \beta$ interaction exemplifies this.

DPID domain transducers utilize full knowledge of a system's underlying dynamic and, thus, perfectly capture domains and structures. Local causal states are built purely from spacetime fields and not the equations of motion used to produce those fields. Yet, the domains and structures they capture are remarkably close to the dynamical systems benchmark set by DPID. This is highly encouraging as the local causal states can be uniformly applied to a much wider array of systems than the DPID domain transducers, while at the same time providing a more powerful analysis of coherent structures.

Looking beyond cellular automata, recent years witnessed renewed interest in coherent structures in fluid systems [117, 65, 62]. There has been particular emphasis on Lagrangian methods, which focus on material deformations generated by the flow. The local causal states, in contrast, are an Eulerian approach, as they are built from lightcones taken from spacetime fields and do not require material transport in the system. A frequent objection raised against Eulerian approaches to coherent structures is that such approaches are not "objective"—they are not independent of an observer's frame of reference. This applies for instantaneous Eulerian approaches, however. And so, does not apply to local causal states. In fact, lightcones and the local causal equivalence relation over them are preserved under Euclidean isometries. This can be seen from Equations. (3.9) and (3.10) that define lightcones in terms of distances only and so they are independent of coordinate reference frame. Local causal states are objective in this sense.

Methods in the Lagrangian coherent structure literature fall into two main categories: diagnostic scalar fields and analytic approaches utilizing one or another mathematical

coherence principle. Previous approaches to coherent structures using local causal states relied on the local statistical complexity [112, 231]. This is a diagnostic scalar field and comes with all the associated drawbacks of such approaches [113]. The coherent structure theory presented here, in contrast, is the first principled mathematical approach to coherent structures using local causal states.

With science producing large-scale, high-dimensional data sets at an ever increasing rate, data-driven analysis techniques like the local causal states become essential. Standard machine learning techniques, most notably deep learning methods, convolutional neural nets, and the like are experiencing increasing use in the sciences [244, 245]. Unlike commercial applications in which deep learning has led to surprising successes, scientific data is highly complex and typically unlabeled. Moreover, interpretability and detecting new mechanisms are key to scientific discovery. With these challenges in mind, we offer local causal states as a unique and valuable tool for discovering and understanding emergent structure and pattern in spatiotemporal systems.

Chapter 7

Coherent Structures in Complex Fluid Flows

Over the last decade, the Data Deluge [246] brought dramatic progress across all of science [45, 46, 247, 47, 248]. For data-driven science to flourish by extracting meaningful scientific insights [48, 49], new methods are required that (i) discover and mathematically describe complex emergent phenomena, (ii) uncover the underlying physical and causal mechanisms, and (iii) accurately predict the occurrence and evolution of these phenomena over time. Increasingly, scientists lean on machine learning (ML) [249, 250, 251, 252, 253] and, more recently, deep learning (DL) [254, 255, 256, 257, 258] to fill this role.

While these techniques show great promise, serious challenges arise when they are applied to scientific problems. To better elucidate the challenges of scientific application of DL methods, we will focus on a particular problem of utmost and imminent importance that necessitates data-driven discovery - detection and identification of extreme weather events in climate data [259, 129, 130]. Driven by an ever-warming climate, extreme weather events are changing in frequency and intensity at an unprecedented pace [123, 124]. Scientists are simulating a multitude of climate change scenarios using high-resolution, high-fidelity global climate models, producing 100s of TBs of data per simulation. Currently, climate change is assessed in these simulations using summary statistics (e.g. mean global sea surface temperature) which are inadequate for analyzing the full impact of climate change. Due to the sheer size and complexity of these simulated

data sets, it is essential to develop robust and automated methods that can provide the deeper insights we seek.

Recently, supervised DL techniques have been applied to address this problem [125, 127, 128] including one of the 2018 Gordon Bell award winners [126]. However, there is an immediate and daunting challenge for these supervised approaches: ground-truth labels do not exist for pixel-level identification of extreme weather events [130]. The DL models used in the above studies are trained using the automated heuristics of TECA [129] for proximate labels. While the results in [125] qualitatively show that DL can improve upon TECA, the results in [128] reach accuracy rates over 97%, essentially reproducing the output of TECA. The supervised learning paradigm of optimizing objective metrics (e.g. training and generalization error) breaks down here [49] since TECA is not ground truth and we do not know how to train a DL model to disagree with TECA in just the right way to get closer to "ground truth".

With the absence of ground-truth labels, many scientific problems are fundamentally unsupervised problems. Rather than attempt to adapt unsupervised DL approaches to a problem like extreme weather detection, we instead take a behavior-driven approach and start from physical principles to develop a novel physics-based representation learning method for discovering structure in spatiotemporal systems directly from unlabeled data.

Having introduced the local causal states and their mathematical foundations in Chapters 2 and 3, and demonstrated their ability to capture meaningful pattern and structure in cellular automata in Chapters 4, 5, and 6, we are now ready to investigate the potential utility of the local causal states for data-driven scientific discovery on real-world problems. First, in Section 7.1 we discuss the approximations required to reconstruct the local causal states from real-valued spacetime fields. The main barriers to analyzing natural systems like fluid flows, however, arises in the required computational resources. In Section 7.2 we detail our distributed, high-performance computing (HPC) implementation of local causal state reconstruction that removes this barrier [122].

For the problem of unsupervised segmentation of extreme weather events in climate data, we view these events as particular cases of more general hydrodynamic coherent

structures. Atmospheric dynamics, and hydrodynamic flows more generally, are highly structured and largely organized around a low-dimensional skeleton of collective features, referred to as *coherent structures* [62]. In Section 7.3 we use the Lagrangian approach to coherent structures in fluid flows [117] as a benchmark for assessing the potential of using the local causal state coherent structure analysis (Chapter 6) for discovering these structures. Finally, in Section 7.4 we present preliminary results on the use of local causal states for unsupervised segmentation analysis of extreme weather events in high-resolution general circulation climate model datasets.

7.1 Reconstruction and Approximations

As discussed in Section 2.2, it is not known how to find "good" measurement devices (i.e. generating partitions) for natural systems like fluid flows. Thus, rather than reconstructing a machine presentation from a discretized measurement data stream, like in the dynamical structure modeling paradigm described in Section 2.1.2, we must perform reconstruction directly from observed spacetime fields. This was first done by Jänicke et. al. in Ref. [231]. In order to estimate empirical morph distributions $\Pr(\mathtt{L}^+|\ell^-)$ some kind of discretization is required. Rather than discretize the observable values, clustering over lightcones was done to discretize lightcone space. Once this is done, the reconstruction algorithm follows in the same manner as for discrete-valued observable fields, as first outlined in Ref. [110].

The reconstruction algorithm for real-valued fields was further elaborated upon by the LICORS work [233, 234, 260]. In this, they proved some statistical properties of the algorithm, including convergence results, and gave a more general description of the algorithm. The most significant contribution, from the perspective of the present work, is the *continuous histories* assumption that provides a theoretical justification for the lightcone clustering step. We will discuss this in more detail below.

The reconstruction algorithm used in the rest of this Chapter largely follows the basic LICORS algorithm given in Ref. [233]. That said, we provide a more thorough mathematical formalism behind the algorithm and its approximations. The main modification we make to the actual algorithm is the introduction of a *lightcone distance* for use in lightcone clustering, as opposed to the Euclidean distance used by the predecessor algorithms.

7.1.1 Setup

Assume there is a *data-generating process* $\mathcal{P}$. Since this thesis concerns spatiotemporal systems, we present our reconstruction formalism in that setting, but everything given here also applies for reconstruction of causal states from real-valued time series. As usual, denote past lightcones as ℓ^- with random variable $\mathtt{L}^-$. The space of all possible past lightcones is $\mathfrak{L}^-$. Similar notation for future lightcones, their random variables, and the space of future lightcones is ℓ^+, $\mathtt{L}^+$, and $\mathfrak{L}^+$, respectively. We will find a functional form

of the causal equivalence relation useful in what follows, given as:

$$\ell_i^- \sim_\epsilon \ell_j^- \iff \Pr(\mathtt{L}^+|\mathtt{L}^- = \ell_i^-) = \Pr(\mathtt{L}^+|\mathtt{L}^- = \ell_j^-) \iff \epsilon(\ell_i^-) = \epsilon(\ell_j^-) \ . \tag{7.1}$$

Optimality and uniqueness of causal states are only guaranteed in the infinite-length limit of pasts and futures. In this case, both $\mathfrak{L}^-$ and $\mathfrak{L}^+$ are uncountable and the morphs $\Pr(\mathtt{L}^+|\mathtt{L}^- = \ell^-)$ are probability densities, which we will now denote as $\mathrm{pr}(\mathtt{L}^+|\mathtt{L}^- = \ell^-)$. Moreover, they are probability densities conditioned on measure-zero events $\mathtt{L}^- = \ell^-$. If $\mathcal{P}$ satisfies conditional stationarity (which is satisfied for the systems considered in this work, with fixed local dynamics that are applied uniformly through space and time) then the morph densities are well-defined and thus so are the causal states. But how can we access and work with these objects empirically when we only ever have access to finite samplings of $\mathcal{P}$? There is an uncountable number of pasts ℓ^- and for each there is a continuous probability density $\mathrm{pr}(\mathtt{L}^+|\mathtt{L}^- = \ell^-)$ over an uncountable number of futures ℓ^+. Each ℓ^- is measure zero, so one can never hope to sample an individual morph density more than once.

7.1.2 Reconstruction Formalism

Clearly approximations are needed to replace the measure-zero objects ℓ^- with those of finite measure. As is usual, we need elements of a σ-algebra $\Sigma_{\mathfrak{L}^-}$ on $\mathfrak{L}^-$. In particular, we would like to partition $\mathfrak{L}^-$ using *finite-measure events* $\mathfrak{p}^- \in \Sigma_{\mathfrak{L}^-}$ so that the following densities can actually be sampled from finite data;

$$\mathrm{pr}(\mathtt{L}^+|\mathfrak{P}^- = \mathfrak{p}^-) = \int_{\ell^- \in \mathfrak{p}^-} \mathrm{pr}(\mathtt{L}^+|\mathtt{L}^- = \ell^-) d\mu \ , \tag{7.2}$$

where $\bigcup_i \mathfrak{p}_i^- = \mathfrak{L}^-$, $\mathfrak{p}_i^- \cap \mathfrak{p}_j^- = \emptyset$, and $\mu(\mathfrak{p}^-) \neq 0$ for all $\mathfrak{p}^-$.

Let $\gamma : \ell^- \mapsto \mathfrak{p}^-$ be the mapping from past lightcones to these finite-measure partition elements, which we will refer to simply as *pasts*. The γ mapping from past lightcones to pasts (sets of past lightcones) generates the following equivalence relation,

$$\ell_i^- \sim_\gamma \ell_j^- \iff \ell_i^- \in \mathfrak{p}_a^- \text{ and } \ell_j^- \in \mathfrak{p}_a^- \iff \gamma(\ell_i^-) = \gamma(\ell_j^-) \ . \tag{7.3}$$

For a given process $\mathcal{P}$, how do we construct this partitioning into finite-measure elements $\mathfrak{p}^-$ of a σ-algebra $\Sigma_{\mathfrak{L}^-}$? For discrete-valued processes constructing γ is quite natural

given that a finite-horizon cutoff h^- must be used for reconstruction from finite data: $\Sigma_{\mathfrak{L}^-}$ is the cylindrical σ-algebra and the $\mathfrak{p}^-$ are the length-l cylinder sets of $\mathfrak{L}^-$ (past lightcones that share the same first l symbols).

If $\mathcal{P}$ is real-valued, then the space of finite-length pasts $\mathfrak{L}_l^-$ is also uncountable (similar for futures) and so finite-length pasts are still measure-zero objects. The insight of the LICORS approach is that if two past lightcones are close, according to some metric, then their morphs must necessarily be close. This is know as the assumption of *continuous histories* [233, Assumption 3.1]. Thus for real-valued systems we can partition $\mathfrak{L}_l^-$ into elements of the Borel algebra using distance-based clustering and the resulting clusters give the finite set of pasts $\{\mathfrak{p}_i^-\}$. Two past lightcones are γ-equivalent in this case if they are assigned to the same distance-based cluster, and the γ-function maps past lightcones to their cluster assignment.

All prior work has used Euclidean distance for lightcone clustering. This gives uniform weight to all points within the finite time horizon of the lightcone, and no weight to all points outside. To smooth this step discontinuity, we introduce a *lightcone distance* with an exponential *temporal decay*. Consider two finite lightcones given as flattened vectors $\mathbf{a}$ and $\mathbf{b}$, each of length l;

$$\mathrm{D}_{\mathrm{lightcone}}(\mathbf{a}, \mathbf{b}) \equiv \sqrt{(a_1 - b_1)^2 + \ldots + \mathrm{e}^{-\tau d(l)}(a_l - b_l)^2} \ , \tag{7.4}$$

where τ is the *temporal decay rate* and $d(i)$ is the temporal depth of the lightcone vector at index i. Intuitively, when comparing lightcones we care more about similarities close to the present than similarities in the distant past. Euclidean distance over lightcones gives equal weight to these. As a side note, this decayed lightcone distance is similar to the *echo state property* that allows for effective reservoir computing and for which the reservoir will asymptotically wash out any information from initial conditions [261].

Using distance-based clustering for γ-equivalence, the empirical densities $\mathrm{pr}(\mathtt{L}^+|\mathfrak{p}^-) = \int_{\mathfrak{p}^-} \mathrm{pr}(\mathtt{L}^+|\mathtt{L}^- = \ell^-)d\mu$ can then be sampled. Alternatively, as we will do in this Chapter, the space of future lightcones $\mathfrak{L}_l^+$ can similarly be partitioned into a finite set of clusters (called *futures*) $\{\mathfrak{p}_i^+\}$ using distance-based clustering. Causal equivalence is then approximated using the discrete distributions $\Pr(\mathfrak{P}^+|\mathfrak{p}^-)$.

In either case, γ-equivalence is seen as a pre-partitioning for causal equivalence. If two past lightcones are close, their future morphs must necessarily be close, and thus they are approximately causally equivalent under the ϵ-map. We thus state a more actionable version of the continuous histories assumption, which is implicitly used, but not formally stated, in Ref. [233]: if two past lightcones are γ-equivalent, they are assumed to be ϵ-equivalent,

$$\gamma(\ell_i^-) = \gamma(\ell_j^-) \implies \epsilon(\ell_i^-) = \epsilon(\ell_j^-) \ . \tag{7.5}$$

This is slightly different from the continuous histories assumption, which is a pair-wise statement. Assumption 7.5 above states that a *group* of past lightcones are assumed to be causally equivalent if they are assigned to the same distance-based cluster. While practically implementable, it is also dependent on the chosen clustering algorithm, including clustering parameters.

Note that I have used the formalism of γ-equivalence for both the discrete-valued case and the real-valued case, but the continuous histories assumption does not hold for discrete processes. It is used for finite-length, real-valued pasts and the intuition behind it relies on the presumed continuity and smoothness of the dynamical system that produces $\mathcal{P}$.

Using the empirical morphs, $\mathrm{pr}(\mathtt{L}^+|\mathtt{p}^-)$ or $\Pr(\mathfrak{P}^+|\mathtt{p}^-)$, which can be sampled from finite data, we can define an empirical causal equivalence over the finite-measure $\mathtt{p}^- \in \Sigma_{\mathfrak{L}^-}$. If two empirical morphs are close, according to some empirical test, then their pasts are ψ-equivalent:

$$\mathtt{p}_i^- \sim_\psi \mathtt{p}_j^- \iff \Pr(\mathfrak{P}^+|\mathtt{p}_i^-) \approx \Pr(\mathfrak{P}^+|\mathtt{p}_j^-) \iff \psi(\mathtt{p}_i^-) = \psi(\mathtt{p}_j^-) \ . \tag{7.6}$$

We use the empirical distributions $\Pr(\mathfrak{P}^+|\mathtt{p}^-)$ rather than the empirical densities $\mathrm{pr}(\mathtt{L}^+|\mathtt{p}^-)$ because we have defined future lightcones to *not* include the present site, and so our future lightcones vectors (with finite horizon $h^+ > 0$) are never dimension 1. This is what is done in the LICORS work, so that they can perform standard density estimates and tests over univariate future lightcones. We thus use the further approximation of discretizing

future lightcones to avoid difficulties with multivariate densities (e.g. multivariate kernel density estimation [262]).

We can now formally state the causal equivalence reconstruction approximation as:

$$\epsilon(\ell^-) \approx \psi\big(\gamma(\ell^-)\big) \ . \tag{7.7}$$

7.1.3 Reconstruction Algorithm

We now use the above formalism to describe our reconstruction algorithm for real-valued systems. The objective of the algorithm is to reconstruct the empirical approximation to the ϵ-function, Approximation (7.7). The core reconstruction parameters are the past lightcone horizon, future lightcone horizon, and speed of information propagation: (h^-, h^+, c). These define the lightcone template, which is shown in Figure 7.1 for systems with two spatial dimensions, as will be considered in this chapter. In what follows, all lightcones are considered as finite vectors defined by this template.

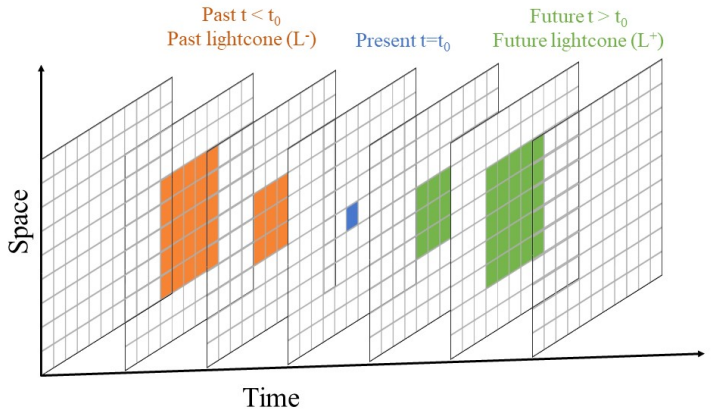


Figure 7.1. 2+1 D lightcone template with past horizon $h^- = 2$, future horizon $h^+ = 2$, and speed of information propagation $c = 1$. Credit: Nalini Kumar

7.1.3.1 Lightcone extraction

The main task in local causal state reconstruction is estimation of the empirical morphs, $\Pr(\mathfrak{P}^+|\mathfrak{p}^-)$. Ultimately this comes down to counting past lightcone - future lightcone pairs, (ℓ^-, ℓ^+). Thus the first step, shown in Algorithm 1, is to extract all such lightcone

pairs from the given spacetime field(s) $\mathbf{x}$ and store them in the paired lists ([plcs], [flcs]). Lightcones are stored as flattened vectors with dimension $dim(\ell^{\pm}) = \sum_{d=0}^{h^{\pm}}(2dc+1)^2$. Note that there will be points along the boundary of $\mathbf{x}$ where full lightcones are not present. The collection of such points is called *the margin* of $\mathbf{x}$.

Algorithm 1: Lightcone extraction

Data: spacetime field $\mathbf{x}$

Result: lists [plcs] and [flcs]

for *spacetime coordinates* $(t, y, x) \in \mathbf{x}$ *not in the margin* **do**

  read values of $\mathtt{L}^-(t, y, x)$ in canonical order;

  write values to flattened array ℓ^-;

  add ℓ^- to [plcs];

  read values of $\mathtt{L}^+(t, y, x)$ in canonical order;

  write values to flattened array ℓ^+;

  add ℓ^+ to [flcs];

end

7.1.3.2 Cluster lightcones

For real-valued systems, like the fluid flows considered here, unique (ℓ^-, ℓ^+) pairs will never repeat and so we implement γ-equivalence using distance-based clustering over lightcones. Because we will use the empirical distribution $\Pr(\mathfrak{P}^+|\mathfrak{P}^- = \mathfrak{p}^-)$, and not empirical densities $\mathrm{pr}(\mathtt{L}^+|\mathfrak{P}^- = \mathfrak{p}^-)$, we will independently cluster over both past lightcones and future lightcones, as shown in Algorithm 2; $\gamma^- : \ell^- \mapsto \mathfrak{p}^-$ is the mapping from past lightcones to assignments of clusters over past lightcones, and similarly $\gamma^+ : \ell^+ \mapsto \mathfrak{p}^+$ is the mapping from future lightcones to their cluster assignments.

7.1.3.3 Build morphs

After clustering [plcs] and [flcs] to produce [pasts] and [futures], respectively, we can build the empirical morphs $\Pr(\mathfrak{P}^+|\mathfrak{P}^- = \mathfrak{p}^-)$. They are found as rows of the joint distribution matrix D, where $D_{i,j} = \Pr(\mathfrak{P}_i^-, \mathfrak{P}_j^+)$. To get D we simply count occurrences of pairs $(\mathfrak{p}^-, \mathfrak{p}^+)$ in ([pasts], [futures]). This is shown in Algorithm 3. Note that we

Algorithm 2: Lightcone clustering

Data: lists [plcs] and [flcs]

Result: list [pasts] of past lightcone cluster assignment labels and list [futures] of future lightcone cluster assignment labels

begin

Input: [plcs];

perform distance-based clustering using $\mathrm{D}_{\mathrm{lightcone}}(\ell^-_{\mathbf{i}}, \ell^-_{\mathbf{j}})$;

write cluster assignment labels to [pasts] such that;

each $\mathfrak{p}^-_i = [\texttt{pasts}]_i = \gamma^-(\ell^-_i)$: $\ell^-_i = [\texttt{plcs}]_i$;

Output: [pasts];

end

begin

Input: [flcs];

perform distance-based clustering using $\mathrm{D}_{\mathrm{lightcone}}(\ell^+_{\mathbf{i}}, \ell^+_{\mathbf{j}})$;

write cluster assignment labels to [futures] such that;

each $\mathfrak{p}^+_i = [\texttt{futures}]_i = \gamma^+(\ell^+_i)$; $\ell^+_i = [\texttt{flcs}]_i$;

Output: [futures];

end

use integer labels $i \in \{0, 1, \ldots, N^-\}$ for the past lightcone cluster assignments (pasts) $\mathfrak{p}^-_i$, where N^- is the total number of past clusters $|\mathfrak{P}^-|$. Similarly for futures $\mathfrak{p}^+_i$ with $N^+ = |\mathfrak{P}^+|$.

7.1.3.4 Causal equivalence

With the empirical morphs $\Pr(\mathfrak{P}^+|\mathfrak{P}^- = \mathfrak{p}^-)$ in hand we can reconstruct causal equivalence of pasts, i.e. ψ-equivalence, using a two-sample test. We reconstruct ψ-equivalence using hierarchical agglomerative clustering, as shown in Algorithm 4. Distribution similarity $\Pr(\mathfrak{P}^+|\mathfrak{p}^-_i) \approx \Pr(\mathfrak{P}^+|\xi_a)$ is evaluated using a chi-squared test with p-value 0.05.

The resulting equivalence classes are the approximated local causal states, and the approximation of $\epsilon(\ell^-)$ is given as Approximation 7.7 using the reconstructed γ and ψ

Algorithm 3: Build morphs

Data: [pasts] and [futures]

Result: joint distribution matrix D

Initialize D as N^- by N^+ array of zeros;

for $(\mathfrak{p}^-, \mathfrak{p}^+)$ *in ([pasts], [futures])* **do**

  increment $D_{\mathfrak{p}^-,\mathfrak{p}^+}$ by 1 ;

end

Algorithm 4: Causal equivalence

Data: joint distribution D

Result: approximated local causal states and ϵ-map

Initialize empty list [states] of local causal states;

for $\Pr(\mathfrak{P}^+|\mathfrak{p}_i^-) = D_i$ *in* D **do**

  for ξ_a *in [states]* **do**

    if $\Pr(\mathfrak{P}^+|\mathfrak{p}_i^-) \approx \Pr(\mathfrak{P}^+|\xi_a)$ **then**

      add $\mathfrak{p}_i^-$ to ξ_a; $\mathfrak{p}_i^- \in \xi_a$;

      update $\psi(\mathfrak{p}_i^-) = \xi_a$;

      break;

    end

  end

  else

    initialize new state as $\xi_b = \{\mathfrak{p}_i^-, \}$;

    add ξ_b to [states];

    update $\psi(\mathfrak{p}_i^-) = \xi_b$

  end

end

functions: $\epsilon(\ell^-) \approx \psi\big(\gamma(\ell^-)\big)$.

7.1.3.5 Causal filter

Using the approximated ϵ-map we can perform spacetime segmentation of $\mathbf{x}$ though causal filtering. The γ-function has already been applied to produce [`pasts`] by clustering [`plcs`]. We then apply the learned ψ-function from Causal equivalence, Algorithm 4, to [`pasts`] to produce [`states`]. Because all these lists are in spacetime order, we simply reshape [`states`] to get the approximated local causal state field $\mathcal{S} \approx \psi\big(\gamma(\mathbf{x})\big)$. Causal filtering is shown in Algorithm 5.

Algorithm 5: Causal filter

Data: spacetime field $\mathbf{x}$

Result: local causal state field $S = \epsilon(\mathbf{x})$

Initialize empty local causal state field S with same dimensions as $\mathbf{x}$;

for *spacetime coordinates* $(t, y, x) \in \mathbf{x}$ **do**

- read past lightcone: $\ell^- = \mathtt{L}^-(t, y, x)$;
- get local causal state: $\epsilon(\ell^-) \approx \psi\big(\gamma(\ell^-)\big) = \xi_a$;
- write local causal state label in S: $S(t, y, x) = a$;

end

7.1.4 Distributed Reconstruction Pipeline

As will be discussed further next in Section 7.2, a distributed implementation of local causal state reconstruction is required for application to real-world systems. Here we briefly describe how to distribute the reconstruction algorithm across multiple compute processes. In Section 7.2 below, we give further detail on our actual implementation of this algorithm in Python.

1. *Data loading*: Stripe the spacetime data so that the spatial fields for each time-step in $\mathbf{x}$ are stored individually to allow for parallel I/O. Let `workset` be the time-steps that each process will extract lightcones from. Because lightcones extend in time, each process must load extra time-steps (`halos`), h^- at the beginning of `workset` and h^+ at the end. Each process loads its `workset + halos` in parallel.

2. *Lightcone extraction*: The temporal haloing removes any need for communication during lightcone extraction, Algorithm 1, which proceeds independently for each process.

3. *Communication barrier*: Ensure all processes have their local [`plcs`] and [`flcs`] lists before proceeding.

4. *Cluster lightcones*: A distributed version of Algorithm 2 is executed across all processes. First cluster the past lightcones across all processes. Store the cluster assignments labels locally, in order. Then do the same for future lightcones.

5. *Build local morphs*: Each process counts $(\mathfrak{p}^-, \mathfrak{p}^+)$ pairs in its local ([`pasts`], [`futures`]) to build D_{local}. Algorithm 3 is executed locally for each process.

6. *Communication barrier*: Wait for all processes to build D_{local}.

7. *Build global morphs*: Execute an all-reduce sum of all D_{local} to yield D_{global} across all processes.

8. *Causal equivalence*: Since each process has D_{global}, they can independently reconstruct the approximated local causal states and ϵ-map according to Algorithm 4

9. *Causal filter*: Each process independently applies the ϵ-map to their `workset` to produce S_{local}; Algorithm 5 is executed locally for each process.

10. *Write output*: Each process independently saves S_{local} with time-order labels so that $S = \epsilon(\mathbf{x})$ can be constructed from all S_{local}.

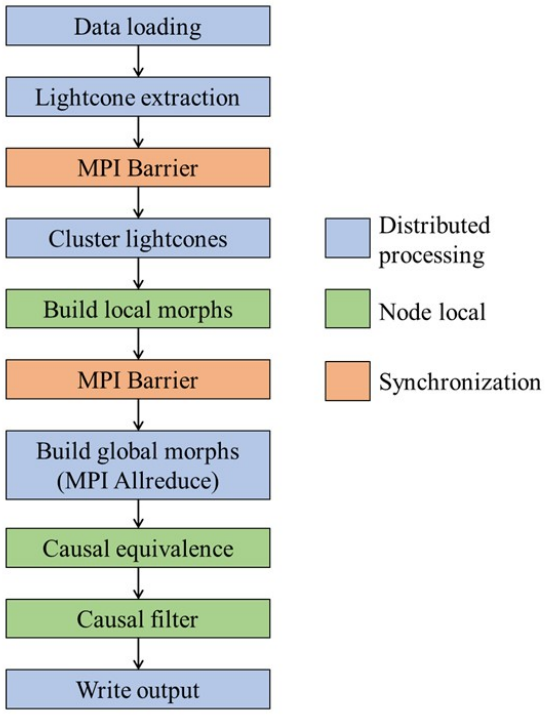


Figure 7.2. Distributed reconstruction pipeline. Credit: Nalini Kumar

7.2 DisCo – HPC Implementation in Python

Before beginning we note that the work presented in this Chapter, and in this Section in particular, was performed as part of ***Project DisCo*** [122]; the goal of which was to create a high-performance computing implementation of the local causal state analysis for application to extreme weather events in large climate simulation data sets. The collaborators for this Project, who contributed greatly to the work described below, are: Nalini Kumar (Intel), Vladislav Epifanov (Intel), Karthik Kashinath (LBNL), Oleksandr Pavlyk (Intel), Frank Schlimbach (Intel), Mostofa Patwary (Baidu Research), Sergey Maidanov (Intel), Victor Lee (Intel), Prabhat (LBNL), and James P. Crutchfield (UC Davis).

Theoretical developments in behavior-driven theories have far outpaced their implementation and application to real science problems due to significant computational demands. Theorists typically use high-productivity languages like Python, which often incur performance penalties, only for prototyping their method and demonstrating its use on small idealized data sets. Since these prototypes aren't typically optimized for production

level performance, their use in science applications with big datasets is limited. To solve real science problems, domain scientists often have to rewrite applications, or portions of, in programming languages like C, C++, and Fortran [263].

Making high-productivity languages performant and scalable on HPC systems requires highly optimized platform-specialized libraries with easy-to-use APIs, seamlessly integrated distributed-memory processing modes with popular Python libraries (like scikit-learn), efficient use of Just In Time (JIT) compilers like Numba etc. In Project DisCo, we use all these techniques to enable optimized Python code from prototype development to production deployment on more than 1000 nodes of an HPC system. This brings us closer to bridging the performance and productivity disconnect that typically exists in HPC, and streamlining the process from theoretical development to deployment at scale for science applications.

A challenge specific to DisCo is the need for distance-based clustering of lightcone data structures (described above in Section. 7.1). Compared to traditional clustering datasets, lightcones are very high-dimensional objects. Though lightcone dimensionality depends on reconstruction parameters, even the baseline lower bound of $O(100)$ dimensions is already very high for typical implementations of clustering methods. To facilitate discovery, our experiments used lightcones with dimension as high as 4495. Also, creation of lightcone vectors increases the on-node data by $O(lightcone\_dimension * 2)$. In our largest run, we process 89.5 TB of lightcone data, which is several orders of magnitude larger than previously reported lightcone-based methods.

To enable novel data-driven discovery at the frontiers of domain science with the ability to process massive amounts of high-dimensional data, we created a highly parallel, distributed-memory, performance optimized implementation of DisCo software including two specialized clustering methods (K-Means [264] and DBSCAN [265]). In keeping with our goal of maintaining scientists' software development productivity, the libraries use standard Python APIs (scikit-learn). These distributed implementations will be upstreamed to benefit the growing community of Python developers.

While unsupervised methods like K-Means [264] and DBSCAN [265] do not place

theoretical limitations on their use with high-dimensional data, the implementations are typically optimized for data with very small dimensions. Performing clustering, especially density-based clustering, in such high dimensions at such data scale has largely been left unexplored until DisCo. Moreover, due to the large amount of both raw and lightcone data, DisCo requires this high-dimensional clustering to be done over multi-node distributed data sets. We developed distributed implementations of K-Means and DBSCAN with similar API calls as scikit-learn and optimized. Our DBSCAN implementation was developed specially for high-dimensional data like the lightcones. In the following, we evaluate the scaling performance with both K-Means and DBSCAN for this large-dataset high-dimensional clustering problem in Project DisCo.

7.2.1 Contributions

Project DisCo makes the following contributions:

- First distributed-memory implementation of local causal state reconstruction allowing unprecedented capability on large scientific data sets.
- Performs unsupervised coherent structure segmentation that qualitatively outperforms state-of-the-art methods for complex realistic fluid flows.
- Demonstrates good single-node, weak scaling, and strong scaling performance up to 1024 nodes.
- Distributed implementation of K-Means and DBSCAN clustering methods for high-dimensional data using standard Python APIs.
- Achieves high performance while maintaining developer productivity by using newly developed optimized Python library functions and efficiently using parallelizing compilers.

7.2.2 Related Work

The basic algorithm for real-valued local causal state reconstruction used by DisCo largely follows that of LICORS [233, 260]. Without an HPC implementation, LICORS focused

on statistical properties of the algorithm, e.g. convergence, and small proof-of-concept experiments. Further, this work used the point-wise entropy over local causal states for coherent structure filters [112], but this approach cannot produce objective segmentation masks. In contrast, our method readily generates objective segmentation masks.

The first real-valued local causal state reconstruction was done in [231], which also analyzed complex fluid flows and climate simulations. They were able to work with these data sets due to efficient data reuse and data sub-sampling from a low-productivity single-node implementation written from scratch. Even with these optimizations in their implementation, DisCo produces much higher resolution results with our high-productivity HPC optimized implementation. Compare the bottom row of Fig. 5 in [231] with Fig. 7.5 in Section. 7.3. They also used the local causal state entropy, and so were also not capable of a structural segmentation analysis.

The structural segmentation methods for fluid flows most relevant to DisCo fall under the heading of Lagrangian Coheret Structures (LCS). These are a collection of approaches grounded in nonlinear dynamical systems theory that seeks to describe the most repelling, attracting, and shearing material surfaces that form the skeletons of Lagrangian particle dynamics [117]. Ref. [113] gives a survey of LCS methods, including two benchmark data sets we use here. This provides us a key point of comparison to the state-of-the-art for method validation, given in Section 7.3.

DisCo's segmentation semantics are built on a structural decomposition provided by the local causal states. Such a decomposition is similar to empirical dimensionality reduction methods, such as PCA [266] and DMD [67]. These methods are used extensively in fluid dynamics [62] and climate analytics [267].

The current state-of-the-art for detecting coherent weather and climate patterns are expert-designed empirical heuristics. These combine the wealth of experience that meteorologists have systematically gathered over decades and simple physical theories of weather and climate events [129]. However, there is often no consensus, even amongst the experts, on the most appropriate, accurate and precise definition for any given weather or climate event. Hence large projects are organized to bring scientists together to com-

pare different methods of detecting and tracking weather and climate patterns [130]. In recent years, scientists have made significant progress in applying methods from machine learning and deep learning to tackle the problems of event classification, detection, segmentation and tracking [244, 111, 256]. The success of these supervised learning methods, however, is strongly limited by the availability of reliable and accurate ground truth training data [131]. Semi-supervised and fully unsupervised segmentation are emerging areas of research with significant challenges [268].

The key step in the DisCo pipeline requires an unsupervised clustering method. We focus on the popular K-Means [264] method and the density-based DBSCAN [265]. Further discussion of clustering in the DisCo pipeline is given in Sections 7.1, 7.2.3, and 7.3.4.

Several distributed implementations of K-Means have been developed over the years. Ref. [269] is a C-based implementation that uses both MPI and OpenMP for parallelization. It evenly partitions the data to be clustered among all processes and replicates the cluster centers. At the end of each iteration, global-sum reduction for all cluster centers is performed to generate the new cluster centers. Ref. [270] is an extension of this work for larger datasets of billions of points, and Ref. [271] optimizes K-Means performance on Intel KNC processors by efficient vectorization. The authors of Ref. [272] propose a hierarchical scheme for partitioning data based on data flow, centroids(clusters), and dimensions. Our K-Means implementation partitions the problem based on data size, since its application to climate data is a weak scaling problem. We process much larger datasizes, though Ref. [272] showcases good performance for much higher dimensionality, up to $O(10^6)$, and clusters $O(10^6)$ than our use case. For comparison, in our capstone problem, in the K-Means stage of DisCo workflow we process $\sim 70 \times 10^9$lightcones ($\sim 70 \times 10^6$ per node) of 84 dimensions into 8 clusters in 2.32 s/iteration on Intel E5-2698 v3 (vs. 2.5E6 samples of 68 dimensions into 10,000 clusters in 2.42 s/iteration on 16 nodes of Intel i7-3770K processors in Ref. [272]). We also use a custom distance metric for applying temporal decay (described in Section 3.2.2) which doubles the number of floating point operations.

Several distributed implementations have been developed for density-based algorithms.

BD-CATS is a distributed DBSCAN implementation using union-find data structures that scales to 8000 nodes on Cori [273]. POPTICS [274] is a distributed implementation of the OPTICS algorithm using the disjoint-set data structure and scaled to 3000 nodes. HDBSCAN from Petuum analytics [275] uses the NN-Descent algorithm and approximates the k-NN graph. While their method has been shown to work for high-dimensional data, it has not been shown to work at scale. Other implementations such as PDBSCAN [276], PSDBSCAN [277], PDSDBSCAN [278], HPDBSCAN [279], etc. have been shown to scale well, but they use specialized indexing structures like k-d trees or ball-trees, which are sub-optimal for clustering high-dimensional data. To the best of our knowledge, this is the first implementation to demonstrate clustering to $O(100)$ dimensional data at this data scale (56 million points per node and 57 billion points in total).

7.2.3 Challenges of Lightcone Clustering

The most significant step, both computationally and conceptually, in the DisCo pipeline is the discretization of lightcone-space via distance-based clustering. While there are many choices for distance-based clustering, we focus on two of the most popular clustering algorithms in the scientific community: K-Means [264] and DBSCAN [265].

The use of clustering in a structural decomposition pipeline, along with the need to conform to the continuous histories assumption (see Section 7.1), would seem to favor a density-based method like DBSCAN over a prototype-based method like K-Means. Density-connectivity should ensure nearby lightcones are clustered together, whereas K-Means must return K clusters and therefore may put cuts in lightcone-space that separate nearby past lightcones, violating the continuous histories assumption.

Since we should avoid imposing geometric restrictions on the structures captured by local causal states, the ability of DBSCAN to capture arbitrary cluster shapes seems preferable to K-Means, which only captures convex, isotropic clusters. Furthermore, the restriction to K clusters, as opposed to an arbitrary number of cluster with DBSCAN, puts an upper bound on the number of reconstructed local causal states. To test these hypotheses we experimented with both K-Means and DBSCAN at scale to evaluate their parallel scaling performance and the quality of clustering in the DisCo pipeline on real-

world data sets. These experiments and results are discussed in Sections 7.2.4, 7.2.5 and 7.3.

7.2.3.1 Distributed K-Means

We developed a distributed K-Means implementation which will be upstreamed to daal4py [280], a Python package similar in usage to scikit-learn. Daal4Py provides a Python interface to a large set of conventional ML algorithms highly tuned for Intel® platforms. In contrast to other distributed frameworks for ML in Python, daal4py uses a strict single program, multiple data (SPMD) approach, and so assumes the input data to be pre-partitioned. All communication within the algorithms is handled under the hood using MPI.

Our single-node K-Means implementation performs one iteration of the algorithm in the following way: all data points are split into small blocks to be processed in parallel. For each block, distances from all points within the block to all current centroids are computed. Based on these distances, points are reassigned to clusters and each thread computes the partial sums of coordinates for each cluster. At the end of the iteration the partials sums are reduced from all threads to produce new centroids. We use Intel® AVX2 or Intel® AVX512 instructions, depending on the hardware platform, for vectorizing distance computations.

Our multi-node K-Means implementation follows the same general pattern: on each iteration current centroids are broadcast to all nodes, each node computes the assignments and partial sums of coordinates for each centroid, and then one of the nodes collects all partial sums and produces new centroids. We use MPI4Py for collecting partial sums. We utilize various routines for finding the initial set of K centroids – first K feature vectors, K random feature vectors, and K-Means++ [264] – provided by Intel® DAAL (Data Analytics Acceleration Library).

7.2.3.2 Distributed DBSCAN

We developed both single-node and multi-node implementations of DBSCAN optimized for use with high-dimensionality lightcone data.

The single-node DBSCAN implementation computes neighborhoods without using

indexing structures, like k-d tree or ball-tree, which are less suitable for high-dimensional data. The overall algorithmic complexity is quadratic in the number of points and linear in feature size (lightcone dimension). Neighborhood computation for blocks of data points is done in parallel without use of pruning techniques. We use Intel® AVX2 or Intel® AVX512 instructions, depending on the hardware platform, to compute distances between points, giving a 2-2.5x speed-up compared to the non-vectorized version.

For multi-node DBSCAN clustering, the first step is geometric re-partitioning of data to gather nearby points on the same node, inspired by the DBSCAN implementation of Ref. [273]. It is performed using the following recursive procedure: for a group of nodes we choose some dimension, find an approximate median of this dimension from all points currently stored on a node, split the current group of nodes into two halves (with value of chosen dimension lesser/greater than the median) and reshuffle all points so that each node contains only points satisfying the property above.

Next, do geometric re-partitioning recursively for the two resulting halves (groups) of nodes. Then each node gathers from other nodes any extra points that fall into its bounding box (extended by the epsilon in each direction) similar to [278]. Using these gathered points the clustering is performed locally on each node (single node implementation) and the results from all the nodes are then merged into a single clustering.

Because we use an approximate value of the median, the geometric partitions can sometimes have imbalanced sizes. This can impact the overall performance since different nodes will complete local clustering at different times and no node can proceed further until every node has finished. Also, the number of extra points for some geometric partitions lying in low and high density regions of the data set may be different, which may also cause some load imbalance among nodes.

7.2.4 Experimental Setup

Here we describe the data sets used for both the science results and scaling measurements. We also describe the HPC system – Cori – on which these computations were performed.

7.2.4.1 Description of the Cori System

All of our experiments were run on the Cori system at the National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (LBNL). Cori is a Cray XC40 system featuring 2,004 nodes of Intel® Xeon™ Processor E5-2698 v3 (Haswell) and 9,688 nodes of Intel® Xeon Phi™ Processor 7250 (KNL). Both Haswell and KNL nodes were used.

Haswell compute nodes have two 16-core Haswell processors. Each processor core has a 32 KB private L1 instruction cache, 32 KB private L1 data and a 256 KB private L2 cache. The 16 cores in each Haswell processor are connected with an on-die interconnect and share a 40-MB L3 cache. Each Haswell compute node has 128 GB of DDR4-2133 DRAM.

Each KNL compute node has a single KNL processor with 68 cores (each with 4 simultaneous hardware threads and 32 KB instruction and 32 KB data in L1 cache), 16 GB of on-package MCDRAM, and 96 GB of DDR4-2400 DRAM. Every two cores share an 1MB L2 (with an aggregate of 32MB total). The 68 cores are connected in a 2D mesh network. All measurements on KNL reported in this paper are performed with the MCDRAM in "cache" mode (configured as a transparent, direct-mapped cache to the DRAM).

Compute nodes in both the Haswell and KNL partitions are connected via the high-speed Cray Aries interconnect. Cori also has a Sonnexion 2000 Lustre filesystem, which consists of 248 Object Storage Targets (OSTs) and 10,168 disks, giving nearly 30PB of storage and a maximum of 700GB/sec IO performance.

7.2.4.2 Libraries and Environment

The DisCo application code is written in Python using both open-source and vendor optimized library packages. We use `Intel® Distribution Of Python (IDP) 3.6.8`. IDP incorporates optimized libraries such as Intel® MKL and Intel® DAAL for machine learning and data analytics operations to improve performance on Intel platforms. We also use `NumPy (1.16.1), SciPy (1.2.0), Numba (0.42.1), Intel® TBB (2019.4), Intel® DAAL (2019.3) and Intel® MKL (2019.3)` from Intel® Anaconda channels. `MPI4Py`

`(3.0.0)` is built to use the Cray MPI libraries.

For all K-Means experiments, our optimized implementation was built from source with Cray MPI and ICC (18.0.1 20171018). These will be contributed back to Intel® daal4py. We compile our DBSCAN implementation with Intel® C/C++ Compilers (ICC 18.0.1 20171018) and without the `-fp-model strict` compiler switch which can impact the vectorization performance. Both K-Means and DBSCAN are linked to Cray MPI binaries as well.

For scaling tests we installed the `conda` environments on the Lustre filesystem to improve Python package import times for large runs on Cori [263]. For K-Means experiments, we run the code with 1 MPI process per Haswell socket and limit the number of TBB threads to 32 on a node with `-m tbb -p 32` flags to the Python interpreter. For DBSCAN experiments we run the code with 1 MPI process per node and 68 tbb threads on KNL, and 1 MPI process per node with 32 threads on Haswell nodes.

7.2.4.3 Datasets

Two benchmark data sets – 2D turbulence and clouds of Jupiter – were chosen for validation against a survey of LCS methods from Ref. [113] and a simulated climate data set was chosen to demonstrate scientific and scaling performance on a real-world scientific application, as in [126].

The Jupiter data is interpolated RGB video taken over a period of 10 days by the NASA Cassini spacecraft and converted to integer grayscale [281]. The 2D turbulence data set is the vorticity field from direct numerical solutions of 2-dimensional Navier-Stokes equations using pseudo-spectral methods in a periodic domain [282]. The climate data set, used for scaling experiments, is simulated data of Earth's climate from the 0.25-degree Community Atmospheric Model (CAM5.1) [283]. Climate variables are stored on an 1152 x 768 spatial grid (float32), with a temporal resolution of 3 hours. Over 100 years of simulation output is available as NetCDF files. Our hero run processed 89.5 TB of lightcone data (obtained from 580 GB of simulated climate data).

7.2.5 Performance Results

We performed both K-Means and DBSCAN weak-scaling and strong-scaling experiments with the CAM5.1 climate dataset. For weak-scaling the data size per node is held fixed as the number of nodes increases (i.e. the total amount of data processed increases), while for strong-scaling the data size aggregated over all the nodes is held fixed as the number of nodes increases (hence the data size per node varies for strong-scaling). K-Means experiments are run on Cori Haswell nodes and DBSCAN experiments are run on both Cori Haswell and Cori KNL nodes. The performance of each stage of the pipeline as well as the total wall-clock time to solution (including synchronization) for an end-to-end single run is measured in seconds. These measurements capture the end-to-end capability of the system and software, including the single node optimizations, efficiency of the distributed clustering methods, and interconnect subsystems.

7.2.5.1 K-Means: Single-Node Performance

Table 7.1 shows the breakdown of execution time of different stages of the DisCo pipeline developed from scratch on one Haswell and one KNL node.

The *data read* stage simply reads the spacetime field data into memory which is then processed in *extract* to generate lightcone vectors. This involves reading spatiotemporal neighbors of each point in the field and flattening them into an n-dimensional vector. These are serial read/write operations that are unavoidable, but the memory access pattern can be optimized. Using Numba decorators for JIT optimization to improve caching and vectorization performance, we obtained a 64x speedup on Haswell and 134x speedup on KNL node resulting in overall speedup of 16.9x on Haswell and 62x on KNL over the baseline implementation inspired by [231]. For the *cluster_lc* stage, we compare our optimized K-Means implementation which gives ∼20x better performance than stock scikit-learn [284]. The other three stages take only a small fraction of the execution time and have little to gain from directed optimization.

7.2.5.2 K-Means: Multi-Node Scaling

All experiments were conducted with $h^{\pm} = 3$, $c = 1$, number of clusters `K=8`, and `iterations=100` for K-Means clustering. The results are shown in Figure 7.3. *Extract* is

Table 7.1. Single-node performance of the different stages of the DisCo pipeline before and after optimization

Stage	Haswell, time(s)			KNL, time(s)		
	Baseline	Opti-mized	**Speed up**	Baseline	Opti-mized	**Speed up**
data read	3.32	3.29	1	8.44	7.01	1.2
extract	519.64	7.85	65	4713.47	35.13	134
cluster_lc	399.63	19.03	21	513.63	26.34	19
morphs	0.85	0.86	1	6.19	6.16	1
equivalence	0.002	0.002	1	0.56	0.02	26
causal filter	0.14	0.14	1	0.49	0.49	1
Total	923.58	31.20	**29.6**	5242.78	74.93	**70**

embarrassingly parallel and thus, shows excellent scaling.

For weak scaling on Haswell, we used 220MB/node of raw data (80 timesteps of 1152 x 768 spatial fields). After lightcone extraction (84 dimension vectors of float32 data), the size of input to the clustering stage increases to 87.44GB/node. We achieved weak-scaling efficiency of 91% at 1024 nodes, measured against performance at 8 nodes. This is expected from increased amounts of time spent in communication at the end of each K-Means iteration as node concurrency increases.

For strong scaling experiments on Haswell, we used 64 timesteps per node on 128 nodes, 32 timesteps per node on 256 nodes, 16 timesteps per node on 512 nodes, and 8 timesteps per node on 1024 nodes. After lightcone extraction the total size of input data to the clustering stage is 54GB. We achieved 64% overall scaling efficiency and 81% clustering efficiency at 1024 nodes. At 1024 nodes, the amount of local computation workload on a node is small compared to the number of synchronization steps within K-Means and in the end-to-end pipeline.

7.2.5.3 DBSCAN: Single-Node Performance

We used the pipeline optimized for K-Means results, shown in Table 7.1. In the *cluster_lc* stage, we use our implementation of the DBSCAN algorithm discussed in Section

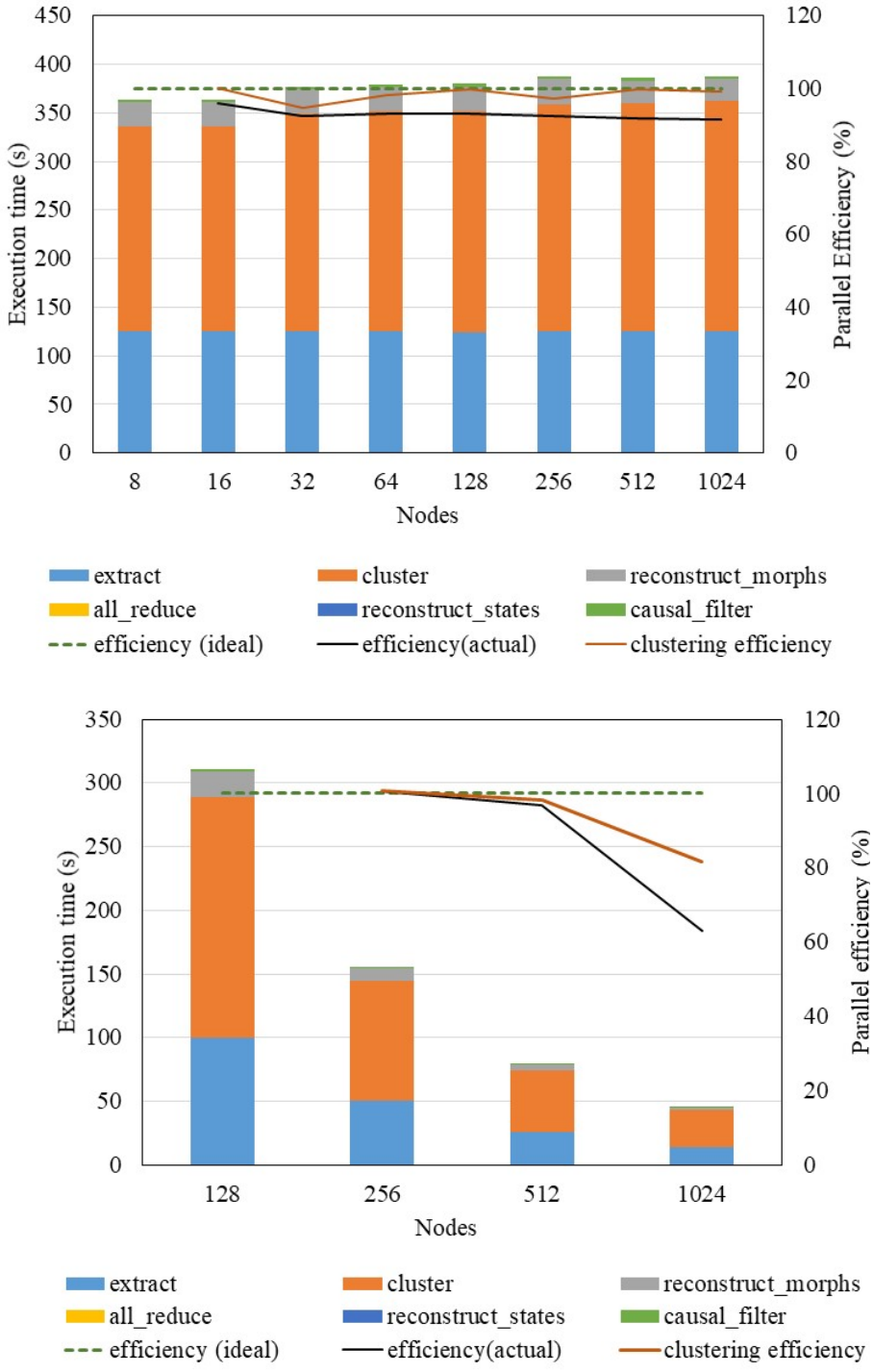


Figure 7.3. Breakdown of execution time spent in various stages of the DisCo on Haswell nodes with K-Means. Top : weak scaling and Bottom: strong scaling. Parallel efficiency are plotted on the secondary axis. Credit: Nalini Kumar

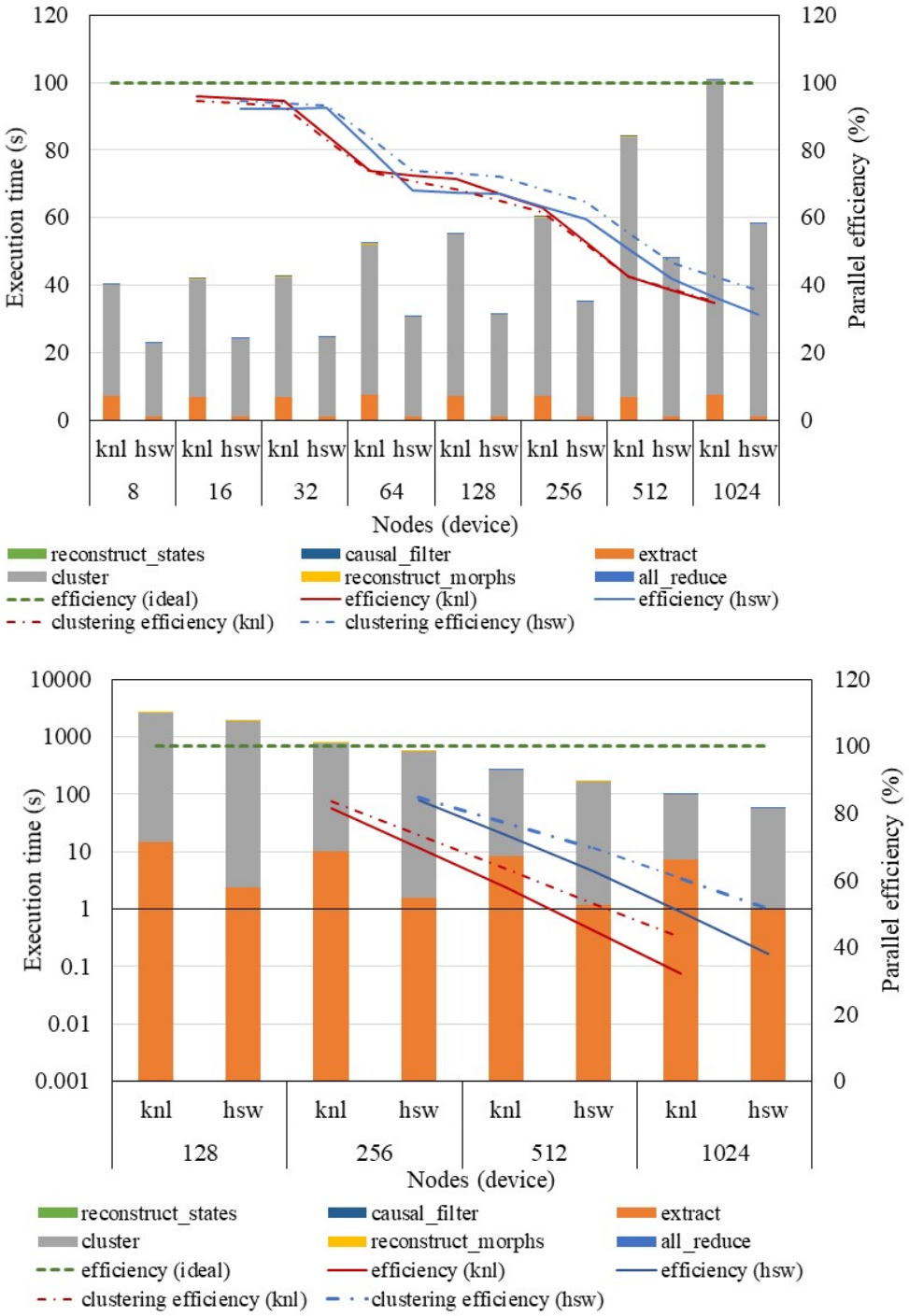


Figure 7.4. Breakdown of execution time spent in various stages of the DisCo on Haswell and KNL nodes with DBSCAN. Top : weak scaling and Bottom: strong scaling. Parallel efficiency are plotted on the secondary axis. Credit: Nalini Kumar

7.2.3. Designed for high-dimensional clustering, it does not use indexing data structures for nearest neighbor calculations. On the 2D turbulence data set, the scikit-learn DBSCAN with brute-force neighborhood computation is more than 3x faster than the default scikit-learn DBSCAN, which uses k-d trees, while producing reasonable results (less than 20% noise points). In turn, our DBSCAN implementation is more than 3x faster than the scikit-learn brute implementation (same clustering parameters) due to better on-node parallelization and use of AVX2 and AVX512 vector instructions for computing neighborhoods and distances.

7.2.5.4 DBSCAN: Multi-Node Scaling

We performed DBSCAN weak scaling and strong scaling runs using the climate data set on both Haswell and KNL nodes. All experiments were conducted with `minpts = 10` and `eps=0.05` for DBSCAN clustering. The results are shown in Figure 7.4.

For weak scaling on Haswell and KNL, we split a single timestep of the 1152 x 768 spatial field across 8 nodes. At 1024 nodes, we achieved a scaling efficiency of 34.6%. The poor scaling efficiency can be attributed to several causes. One, as discussed in Section 7.2.3, distributed DBSCAN uses geometric partitioning to gather adjacent points on the same node. Then, at each step, every node clusters its local data subset before merging results among different nodes. Two, since we didn't use indexing data structures to perform local clustering in DBSCAN, the complexity of each step is $O(dimensionality * |size\ of\ the\ partition|^2)$. Third, the total clustering time is equal to the running time of the slowest node, which is the node containing the largest data partition. As the number of nodes increases, it leads to an increase in imbalance in number of points between nodes and increased total running time, as can be seen in Figure 7.4. We are exploring ways of better partitioning the initial data to resolve the load imbalance issue while maintaining the scalability with increasing number of dimensions.

For strong scaling on Haswell and KNL, we used a single timestep of the 1152 x 768 spatial field per node for the 128-nodes run; one timestep across 2 nodes for the 256-nodes run; one timestep across 4 nodes for the 512-nodes run; and one timestep across 8 nodes for the 1024-nodes run. We achieved an overall scaling efficiency of 38% and

Table 7.2. Load distribution from geometric partitioning in DBSCAN

Nodes	128	256	512	1024
Min	87392	81960	84963	70824
Max	130867	130147	154574	172377
Average	105156	105156	105156	105156
Median	104378	104759	103854	103666

clustering efficiency of 52% on 1024 Haswell nodes. Increasing the number of nodes, while preserving the total input data size, results in a proportional decrease of partition sizes gathered per node. From the quadratic dependency on the number of points mentioned earlier, reducing the sizes of the partitions by $2x$, decreases the execution time by $4x$. However, since the partitions are not balanced, the obtained efficiency from increasing the number of nodes is marginally lower than the expected $4x$ reduction in execution time.

7.2.6 Hero Run

Our largest run processed 89.5 TB of lightcone data (obtained from 580 GB of simulated climate data) with distributed K-Means clustering on 1024 Intel Haswell nodes with 2 MPI ranks/node and 32 tbb threads/processor. We do not use Cori burst buffer. 580GB of climate data is read from the Cori /cscratch/ filesystem for generating nearly 90TB of lightcone data, after `extract`, which resides in the on-node memory. The left column of Figure 7.3 shows execution times for this run. The total time to solution was 6.6 minutes with a weak scaling efficiency of 91%. This suggests that further scaling may be possible.

7.2.7 Intel Legal Disclaimers

"Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to www.intel.com/benchmarks.

Performance results are based on testing as of April 10, 2019 and may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.

Configurations: Testing on Cori (see 7.2.4) was performed by NERSC, UC Davis, and Intel, with the spectre_v1 and meltdown patches.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. Check with your system manufacturer or retailer or learn more at [intel.com].

Intel, the Intel logo, Intel Xeon, Intel Xeon Phi, Intel DAAL are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. *Other names and brands may be claimed as the property of others."

7.3 Lagrangian Coherent Structures

With the theory and HPC implementation in place, we now present preliminary results demonstrating the ability of the local causal states to capture emergent coherent structures in complex spatiotemporal systems. In particular, we bring the local causal state analysis to bear on the Lagrangian coherent structure (LCS) problem in complex fluid flows.

The local causal state fields that are the direct output of DisCo, without additional semantic analysis (described further below), can be considered a "structural decomposition" of the flow. Incorporating the physics of local interactions to generalize the computational mechanics theory of structure (Section 3.1.6) to spatiotemporal systems, the local causal states are a more principled and well-motivated decomposition approach compared to empirical dimensionality reduction methods such as PCA and DMD (see Sec. 7.2.2), or automated heuristics like TECA.

But does the structural decomposition of the local causal states capture meaningful "structure"? What constitutes physically meaningful structure in complex fluid flows is an incredibly challenging open problem [62, 113]. Even something as seemingly obvious as a fluid vortex does not have a generally accepted rigorous definition [285]. This is to say that it is impossible to give a quantitative assessment of how close our method gets to ground truth because ground truth for this problem currently does not exist. Validating results that are otherwise not externally well-defined is a challenge faced by any unsupervised method.

In the absence of a quality metric to compare different methods against, the community standard is to qualitatively compare methods against each other. In particular, the Lagrangian approach to coherent structures in complex fluids is gaining wide acceptance and Ref. [113] surveys the current state-of-the-art Lagrangian Coherent Structure methods (see Sec. 7.2.2). We directly compare our results with the geodesic and LAVD approaches (described below) on the 2D turbulence data set from [113] and the Jupiter data set from [113] and [286].

There are three classes of flow structures in the LCS framework; elliptic LCS are rotating vortex-like structures, parabolic LCS are generalized Lagrangian jet-cores, and hyperbolic LCS are tendril-like stable-unstable manifolds in the flow [117]. The geodesic approach [117, 286] is the state-of-the-art method designed to capture all three classes of LCS and has a nice interpretation for the structures it captures in terms of characteristic deformations of material surfaces. The Lagrangian-Averaged Vorticity Deviation (LAVD) [287] is the state-of-the-art method specifically for elliptic LCS, but is not designed to capture parabolic or hyperbolic LCS.

The local causal states are not a Lagrangian method (they are built from spacetime fields, not Lagrangian particle flow) so they are not specifically designed to capture these structures. However, LCS are of interest because they heavily dictate the dynamics of the overall flow, and so signatures of LCS should be captured by the local causal states. As we will see in the results and comparisons in

Sections 7.3.2 and 7.3.3, this is indeed the case.

Snapshot images for our structural decomposition results on the three fluid flow data sets using K-Means clustering in the DisCo pipeline are shown in Figure 7.5. DBSCAN results are discussed in Section 7.3.4. We emphasize that DisCo produces a *spacetime* segmentation; the images shown are single-time snapshots taken from spacetime videos. The left image of each row in Figure 7.5 – (a), (d), and (f) – are snapshots from the unlabeled "training" data used for local causal state reconstruction. As an unsupervised method, nothing else is needed beyond the raw observable spacetime data. The other image(s) in each row are corresponding snapshots from the latent local causal state decomposition fields, which are the output of Algorithm 5: causal filtering, the final stage in the DisCo pipeline. Parameters used for reconstruction are given in the caption of Figure 7.5. Full segmentation videos are available at the DisCo YouTube channel [6].

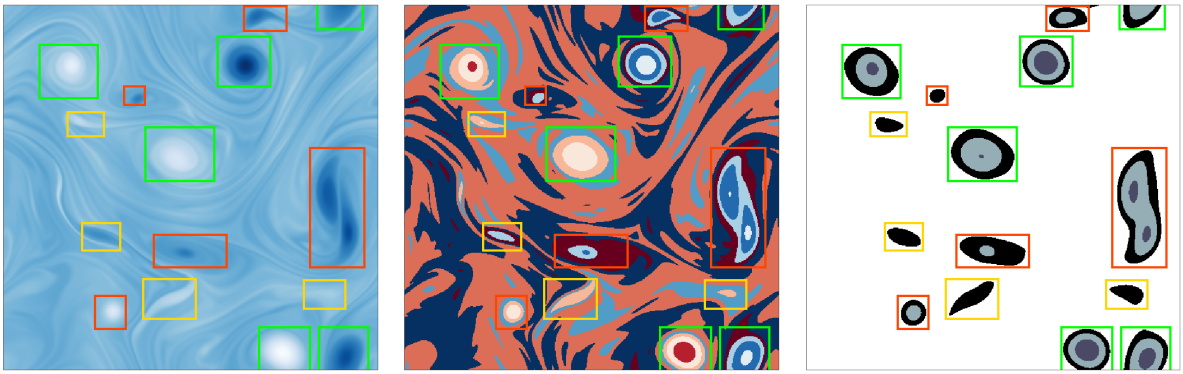
Recall from Section 1.2.1 that for segmentation each point in spacetime (pixel in the video) is assigned a class label. The extreme weather event segmentation masks shown in [126] have the following semantics: `cyclone`, `atmospheric river`, and `background`. In contrast, the segmentation classes of DisCo are the local causal states. Each unique color in the segmentation images – Figure 7.5 (b), (c), (e), and (g) – corresponds to a unique local causal state. Further post-processing is needed to assign semantic labels such as `cyclone` and `atmospheric river` to sets of local causal states. We will discuss this further in Section. 7.3.2 and Section 7.4.

7.3.1 Reconstruction Parameters

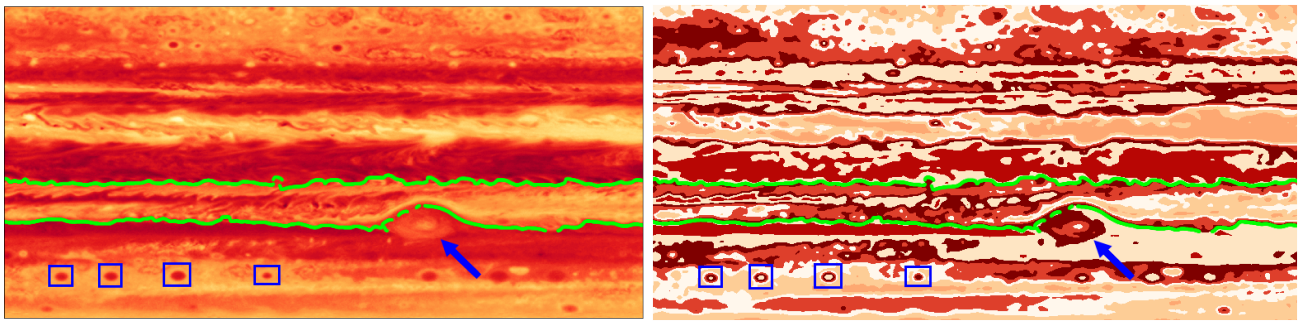
Before we examine the LCS results in detail, a discussion of how reconstruction parameters affect the local causal state structural decomposition is in order. The complex fluid flows of interest are multi-scale phenomena and so the question of how they are structured may not have a single answer. Different notions of structure may exist at different length and time scales. With this in mind, we found that essentially all reconstruction parameters yield a physically valid structural decomposition. Varying parameters adjusts the structural details captured in a satisfying way.

Larger values of K in K-Means produce refinements of structural decompositions from smaller values of K, capturing finer levels of detail. The speed of information propagation c controls the spatial-scale of the structural decomposition and the decay-rate τ controls the temporal coherence scale. Because uniqueness and optimality of local causal states are asymptotic properties, lightcone horizons should be set as large as is computationally feasible. Though some finite cutoff must always be used. The lightcone horizon creates a discrete cut in the amount information of local pasts that is taken into account, as opposed to the smooth drop-off of the temporal decay.

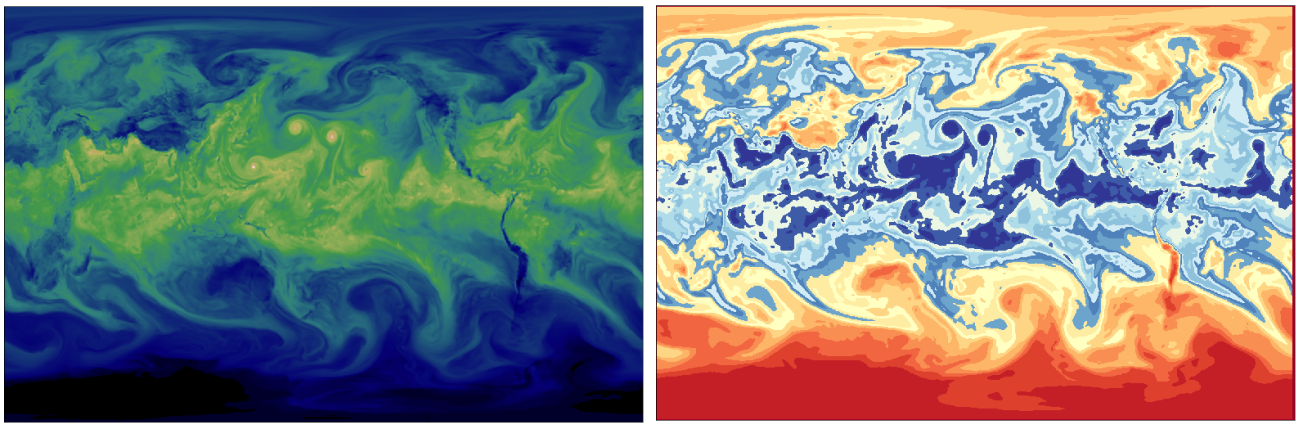
The local causal states, using different parameter values, provide a suite of tools for analyzing structure in complex, multi-scale spatiotemporal systems at various levels of description. Finally, we note that the $\tau \to \infty$ (or, equivalently the $h^{\pm} \to 0$) limit produces a standard K-Means image segmentation, which



(a) Turbulence vorticity field (b) Turbulence state field, fine structure (c) Turbulence state field, coarse structure



(d) NASA Cassini Jupiter cloud data in grayscale (e) Jupiter local causal state field



(f) Water vapor field of CAM5.1 climate model simulation (g) Climate local causal state field

Figure 7.5. Structural segmentation results for the three scientific data sets using K-Means lightcone clustering. The leftmost image of each row shows a snapshot from the data spacetime fields, and the other image(s) in the row show corresponding snapshots from the reconstructed local causal state spacetime fields. Reconstruction parameters given as (h^-, h^+, c, K^-, τ): (b) - (14, 2, 1, 10, 0.8), (c) - (14, 2, 1, 4, 0.0), (e) - (3, 3, 3, 8, 0), (g) - (3, 3, 1, 16, 0.04). $K^+ = 10$ and 0.05 for chi-squared significance level were used for all reconstructions. Full segmentation videos are available on the DisCo YouTube channel [6]

captures instantaneous structure and does not account for coherence across time and space.

7.3.2 2D Turbulence

While still complex and multi-scale, the idealized 2D turbulence data provides the cleanest Lagrangian Coherent Structure analysis using our DisCo structural decomposition. Figure 7.5 (a) shows a snapshot of the vorticity field, and (b) and (c) show corresponding snapshots from structural decompositions using different reconstruction parameter values. Both use the same lightcone template with $h^- = 14$, $h^+ = 2$, and $c = 1$. To reveal finer structural details that persist on shorter time scales, Figure 7.5 (b) uses $\tau = 0.8$ and $K = 10$. To isolate the coherent vortices, which persist at longer time scales, Figure 7.5 (c) was produced using $\tau = 0.0$ and $K = 4$. As can be seen in (b), the local causal states distinguish between positive and negative vortices, so for (c) we removed this asymmetry by reconstructing from the absolute value of vorticity.

All three images are annotated with color-coded bounding boxes outlining elliptic LCS to directly compare with the geodesic and LAVD LCS results from Figure 9, (k) and (l) respectively, in [113]. Green boxes are vortices identified by both the geodesic and LAVD methods and red boxes are additional vortices identified by LAVD but not the geodesic. Yellow boxes are new structural signatures of elliptic LCS discovered by DisCo. Because the background potential flow is mapped to a single local causal state state, colored white, in (c), all objects with a bounding box can be assigned a semantic label of `coherent structure` since they satisfy the local causal state definition given in Section 6.1 as spatially localized, temporally persistent deviations from generalized spacetime symmetries (i.e. local causal state symmetries). Significantly, DisCo has discovered vortices in (c) as coherent structures with this very general interpretation as locally broken symmetries. Further, this structural decomposition was performed with large past lightcones $h^- = 14$ and that have no temporal decay $\tau = 0.0$. Thus vortices are found to be the most long-lived coherent objects in this flow, and all shorter-lived fluctuations in the background potential flow get mapped to a Euclidean-symmetry domain (i.e. a single-state domain).

In the finer-scale structural decomposition of (b) we still have a unique set of states outlining the coherent vortices, as we would expect. If they persist on longer time scales, they will also persist on the short time scale. The symmetry of the background potential flow is broken further, revealing additional organization that largely follows the hyperbolic LCS stable-unstable manifolds. Because they act as transport barriers, they partition the flow on either side and these partitions are given by two distinct local causal states with the boundary between them running along the hyperbolic LCS in the unstable direction. For example, the narrow dark blue-colored state in the upper right of (b) indicates a narrow flow channel squeezed between two hyperbolic LCS.

7.3.3 Clouds of Jupiter

Figure 7.5 (d) shows a snapshot from the Jupiter cloud data, with corresponding structural decomposition snapshot in (e). The Great Red Spot, highlighted with a blue arrow, is the most famous structure in Jupiter's atmosphere. As it is a giant vortex, the Great Red Spot is identified as an elliptic LCS by both the geodesic and LAVD methods [286, 113]. While the local causal states in (e) do not capture the Great Red Spot as cleanly as the vortices in (b) and (c), it does have the same nested state structures as the turbulence vortices. There are other smaller vortices in Jupiter's atmosphere, most notably the "string of pearls" in the Southern Temperate Belt, four of which are highlighted with blue bounding boxes. We can see in (e) that the pearls are nicely captured by the local causal states, similar to the turbulence vortices in (b).

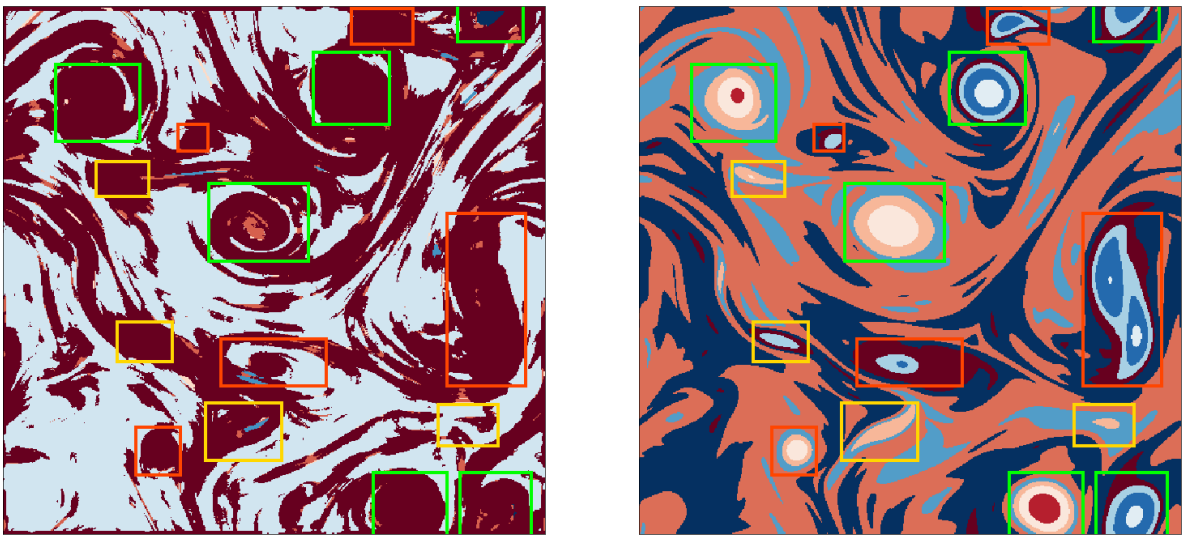
Perhaps the most distinctive features of Jupiter's atmosphere are the zonal belts. The east-west zonal jet streams that form the boundaries between bands are of particular relevance to Lagrangian Coherent Structure analysis. Figure 11 in [286] uses the geodesic method to identify these jet streams as shearless parabolic LCS, indicating they act as transport barriers that separate the zonal belts. The particular decomposition shown in (e) captures a fair amount of detail inside the bands, but the edges of the bands have neighboring pairs of local causal states with boundaries that extend contiguously in the east-west direction along the parabolic LCS transport barriers. Two such local causal state boundaries are highlighted in green, for comparison with Figure 11 (a) in [286]. The topmost green line, in the center of (d) and (e), is the southern equatorial jet, shown in more detail in Figure 11 (b) and Figure 12 of [286]. Its north-south meandering is clearly captured by the local causal states.

7.3.4 Lightcone Clustering Revisited

So what about DBSCAN, which was expected to be the more appropriate clustering method for the DisCo pipeline? For DBSCAN we found that most typical outcomes either classify most points as noise or into one single cluster. Some density parameters give $O(1000)$ clusters, but most contain $O(1)$ lightcones. All cases do not yield physically-meaningful segmentation output.

As a density-based method, when compared to K-Means, DBSCAN faces the curse of dimensionality [288, 289]. Distinguishing density-separated regions becomes exponentially difficult in high-dimensional spaces, such as the lightcone spaces used in our applications. Further, a limiting assumption of DBSCAN is that all clusters are defined by a single density threshold. Adaptive methods like OPTICS [290] and HDBSCAN [291] may perform better for complex data. The results we observe from experiments with DBSCAN suggests that the lightcone-spaces of complex fluid flows do not contain clear density-separated regions, and thus do not yield to a density-based discretization.

Figure 7.6 shows snapshots of spacetime segmentation results of the turbulence data set using both DBSCAN (a) and K-Means (b). The K-Means result in Figure 7.6 (b) is copied from Figure 7.5 (b) for easier visual comparison with the DBSCAN results in Figure 7.6 (a), which used reconstruction



(a) Turbulence local causal state field using DBSCAN

(b) Turbulence local causal state field using K-Means

Figure 7.6. Comparison of structural segmentation results on 2D turbulence using DBSCAN (a) and K-Means (b) for lightcone clustering. The K-Means results in (b) are the same as Figure 7.5 (b), repeated here for easier comparison. The DBSCAN results shown in (a) use reconstruction parameters $(h^-, h^+, c) = (3, 2, 1)$, $\tau = 0.0$, `eps` $= 0.0$, and `minpts` $= 10$.

parameters $(h^-, h^+, c) = (3, 2, 1)$, $\tau = 0.0$, `eps`=0.0005, and `minpts` $= 10$. We can see that the DBSCAN segmentation does outline structure in the flow, but it is all detailed structure of the background potential flow. None of our experiments with different parameter values were able to isolate vortices with a unique set of local causal states, which, as demonstrated above, is possible with K-Means. This inability to isolate vortices, along with the patchwork pattern of decomposition in parts of the background flow suggest that a single density threshold of lightcone space, which is assumed by DBSCAN, is incapable of coherent structure segmentation.

Similarly, our DBSCAN experiments on the climate data typically yielded outcomes that either classify most points as noise or into one single cluster. Some density parameters give $O(1000)$ clusters, but most contain $O(1)$ lightcones. All of these cases do not yield physically meaningful segmentation output, which again points to the shortcoming of a single density threshold. As can be seen in Figure 7.5 (f), the CAM5.1 water vapor data is very heterogeneous in space; the water vapor values are much more uniform towards the poles (the top and bottom of the image) than in the tropics (center). The polar regions will contribute a relatively small, but very dense, region in lightcone space compared to contributions from the tropics.

From experiments it appears that K-Means attempts to uniformly partition lightcone-space, which is consistent with these spaces not being density-separated. If this is in fact the case, the convex clusters that uniformly separate lightcone-space which result from K-Means are actually the most natural discretization

of lightcone-space concordant with the continuous histories assumption used during reconstruction, as described in Section 7.1.

Despite prior assumptions and intuitions, K-Means appears to be a much more effective clustering method for hydrodynamic coherent structure segmentation using our DisCo pipeline. That being said, there are plenty of other applications for which density-based clustering using DBSCAN is the appropriate choice. Our DBSCAN implementation has now made this algorithm available for large, high-dimensional applications, with the same easy-to-use Python API as found in scikit-learn.

7.4 Extreme Weather Events

Finally, we return to climate science. Figure 7.5 (g) shows the local causal state decomposition of the CAM5.1 water vapor field shown in Figure 7.5 (f). While our decomposition of the turbulence and Jupiter data align nicely with the LCS analysis in [113] and [286], we are not yet able to use the climate decomposition to construct segmentation masks of weather patterns.

However, given the promising decomposition in Figure 7.5 (g), we believe this is achievable. The climate decomposition shown here was performed solely on the water vapor field, and not on all physical variables of the CAM5.1 climate model, like was done in [126]. While we see signatures of cyclones and atmospheric rivers outlined in Figure 7.5 (g), it is not surprising that these structures are not uniquely identified from the water vapor fields alone; this would be akin to describing hurricanes as just local concentrations of water vapor. More contextual spacetime information is needed. This includes additional physical fields, and the use of larger lightcones in reconstruction.

A key distinguishing feature of hurricanes is their rotation. While rotation has its signature in the water vapor field, the three timesteps used for the lightcones in our local causal state reconstruction cannot capture much of this rotation. The vorticity field, as used for the 2D turbulence data, gives instantaneous local circulation. From Figure 7.5 (b) and (c) we see that vortices are easier to capture from vorticity. Additionally, hurricanes have distinctive patterns in temperature and pressure, for example a warm, low pressure inner core. Including the vorticity, temperature, and pressure fields into the climate decomposition will help yield a distinct set of states that identify hurricanes as high rotation objects with a warm, low pressure core that locally concentrate water vapor..

Similarly, atmospheric rivers (ARs) have distinguishing characteristics in other physical fields, most notably water vapor transport, that will help identify ARs when added to the decomposition. The use of larger lightcones should be particularly helpful for creating AR masks, as their geometry is crucial for their identification, which extends over a much larger spatial scale than can be captured by the depth-3 lightcones used in the reconstruction here.

Although our structural segmentation requires information from multiple physical observables to identify extreme weather events, the generality of the local causal states will allow us to do this. Lagrangian

approaches to coherent structures in fluid flows are based on Lagrangian particle flow dynamics. Such methods can not be applied to thermodynamic aspects of climate, like temperature and water vapor. The local causal states, which require only spacetime field data, can be applied to thermodynamic fields, as we have shown here.

References

[1] M.C. Cross and H.S. Greenside. *Pattern Formation and Dynamics in Nonequilibrium Systems*. Cambridge University Press, Cambridge, United Kingdom, 2009.

[2] M. Van Dyke. *An Album of Fluid Motion*. Parabolic Press, Stanford, California, 1982.

[3] S. Ji. Kernel trick. https://commons.wikimedia.org/wiki/File:Kernel_trick_idea.svg#filelinks, 2017. Accessed: 2020-01-21.

[4] J. E. Hanson and J. P. Crutchfield. Computational mechanics of cellular automata: An example. *Physica D*, 103:169–189, 1997.

[5] J. E. Hanson and J. P. Crutchfield. The attractor-basin portrait of a cellular automaton. *J. Stat. Phys.*, 66:1415 – 1462, 1992.

[6] Project disco segmentation videos. https://www.youtube.com/channel/UCwKTJloOOFQHVHDwkpqIdYA, Accessed: 2019-04-10.

[7] NASA, JPL-Caltech, SwRI, MSSS, G. Eichstädt, and S. Doran. Jupiters Great Red Spot, spotted. https://www.nasa.gov/image-feature/jpl/pia21985/jupiter-s-great-red-spot-spotted, 2019.

[8] A. T. Winfree. The prehistory of the Belousov-Zhabotinsky oscillator. *Journal of Chemical Education*, 61(8):661, 1984.

[9] A. M. Turing. The chemical basis of morphogenesis. *Trans. Roy. Soc., Series B*, 237:5, 1952.

[10] P. Ball. *The Self-Made Tapestry: Pattern Formation in Nature*. Oxford University Press, New York, 1999.

[11] J. Sommeria, S. D. Meyers, and H. L. Swinney. Laboratory simulation of Jupiter's Great Red Spot. *Nature*, 331(6158):689–693, 1988.

[12] P. S. Marcus. Numerical simulation of Jupiter's Great Red Spot. *Nature*, 331(6158):693–696, 1988.

[13] K. A. Emanuel. The theory of hurricanes. *Ann. Rev. Fluid Mech.*, 23(1):179–196, 1991.

[14] U. Nakaya. *Snow Crystals: Natural and Artificial*. Harvard University Press, Boston, Massachusetts, 1954.

[15] E. Ben-Jacob, I. Cohen, O. Shochet, I. Aranson, H. Levine, and L. Tsimring. Complex bacterial patterns. *Nature*, 373:556–567, 1995.

[16] I. V. Markov. *Crystal Growth for Beginners: Fundamentals of Nucleation, Crystal Growth and Epitaxy*. World Scientific, Singapore, third edition, 2017.

[17] H. Bénard. *Les Tourbillons Cellulaires dans une nappe Liquide Propageant de la Chaleur par Convection: en Régime Permanent*. Gauthier-Villars, 1901.

[18] Lord Rayleigh. On convection currents in a horizontal layer of fluid, when the higher temperature is on the under side. *Phil. Mag. (Series 6)*, 32(192):529–546, 1916.

[19] G.I. Taylor. Stability of a viscous liquid contained between two rotating cylinders. *Phil. Trans. Roy. Soc. Lond. A*, 223:289–343, 1923.

[20] S. Chandrasekhar. *Hydrodynamic and Hydromagnetic Stability*. Oxford, Clarendon Press, 1968.

[21] F. H. Busse. Non-linear properties of thermal convection. *Reports on Progress in Physics*, 41(12):1929, 1978.

[22] P.R. Fenstermacher, H.L. Swinney, and J.P. Gollub. Dynamical instabilities and the transition to chaotic Taylor vortex flow. *J. Fluid Mech.*, 94(1):103–128, 1979.

[23] V. Steinberg, G. Ahlers, and D. S. Cannell. Pattern formation and wave-number selection by Rayleigh-Bénard convection in a cylindrical container. *Physica Scripta*, T9:97, 1985.

[24] L.E. Reichl. *A Modern Course in Statistical Physics*. Wiley-VCH, Weinheim, Germany, 2016.

[25] H. Haken. *Information and Self-Organization*. Springer, New York, 2016.

[26] S. A. Kivelson, E. Fradkin, and V. J. Emery. Electronic liquid-crystal phases of a doped Mott insulator. *Nature*, 393(6685):550–553, 1998.

[27] B.-X. Zheng, C.-M. Chung, P. Corboz, G. Ehlers, M.-P. Qin, R. M. Noack, H. Shi, S. R. White, S. Zhang, and G. K.-L. Chan. Stripe order in the underdoped region of the two-dimensional Hubbard model. *Science*, 358(6367):1155–1160, 2017.

[28] N. Goldenfeld. *Lectures On Phase Transitions And The Renormalization Group*. Westview Press, New York, 1992.

[29] M. C. Cross and P. C. Hohenberg. Pattern formation outside of equilibrium. *Rev. Mod. Phys.*, 65(3):851–1112, 1993.

[30] R. Hoyle. *Pattern Formation: An Introduction to Methods*. Cambridge University Press, New York, 2006.

[31] M. Golubitsky and I. Stewart. *The Symmetry Perspective: From Equilibrium to Chaos in Phase Space and Physical Space*, volume 200. Birkhäuser, New York, 2003.

[32] P. Glansdorff and I. Prigogine. Thermodynamic theory of structure, stability and fluctuations. 1971.

[33] G. Nicolis and I. Prigogine. *Self-Organization in Nonequilibrium Systems*. Wiley, New York, 1977.

[34] D. Kondepudi and I. Prigogine. *Modern Thermodynamics: From Heat Engines to Dissipative Structures*. John Wiley & Sons, 2014.

[35] H. L. Swinney. Emergence and evolution of patterns. In *AIP Conference Proceedings*, volume 501, pages 3–22. American Institute of Physics, 2000.

[36] J. Keizer and R.F. Fox. Qualms regarding the range of validity of the glansdorff-prigogine criterion for stability of non-equilibrium states. *Proceedings of the National Academy of Sciences*, 71(1):192–196, 1974.

[37] R.F. Fox. The excess entropy around nonequilibrium steady states,$(\delta^2 s)_{ss}$, is not a liapunov function. *Proceedings of the National Academy of Sciences*, 77(7):3763–3766, 1980.

[38] R. Landauer. Inadequacy of entropy and entropy derivatives in characterizing the steady state. *Physical Review A*, 12(2):636, 1975.

[39] E T Jaynes. The minimum entropy production principle. *Annual Review of Physical Chemistry*, 31(1):579–601, 1980.

[40] A.C. Newell. Envelope equations. *Lectures in Applied Mathematics*, 15(157):4, 1974.

[41] I. Prigogine. From being to becoming: Time and complexity in the physical sciences. *Philosophy of Science*, 51(2):355–357, 1980.

[42] W.T. Grandy. *Entropy and The Time Evolution of Macroscopic Systems*, volume 10. Oxford University Press, 2008.

[43] P. Attard. *Non-Equilibrium Thermodynamics and Statistical Mechanics: Foundations and Applications*. Oxford, 2012.

[44] F. J. Dyson. *The scientist as rebel*. New York Review of Books, 2006.

[45] Tom Chivers. How big data is changing science. https://mosaicscience.com/story/how-big-data-changing-science-algorithms-research-genomics/, 2018. Accessed: 2019-04-07.

[46] Marc Chahin. How big data advances physics. https://www.elsevier.com/connect/how-big-data-advances-physics, 2017. Accessed: 2019-04-07.

[47] Terrence J Sejnowski, Patricia S Churchland, and J Anthony Movshon. Putting big data to good use in neuroscience. *Nature Neuroscience*, 17(11):1440, 2014.

[48] J. P. Crutchfield. The dreams of theory. *WIRES Comp. Stat.*, 6(March/April):75–79, 2014.

[49] James H Faghmous and Vipin Kumar. A big data guide to understanding climate change: The case for theory-guided data science. *Big data*, 2(3):155–163, 2014.

[50] C. Anderson. The end of theory: The data deluge makes the scientific method obsolete. https://www.wired.com/2008/06/pb-theory/, 2008. Accessed: 2019-04-07.

[51] V. Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[52] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[53] T. M. Mitchell. Machine learning. *McGraw Hill*, 45(37):870–877, 1997.

[54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[55] P. Mehta, M. Bukov, C.-H. Wang, A. G.R. Day, C. Richardson, C. K. Fisher, and D. J. Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics reports*, 2019.

[56] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[57] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[58] J. Mercer. Functions of positive and negative type, and their connection the theory of integral equations. *Philosophical Transactions of the Royal Society A.*, 209(441-458):415–446, 1909.

[59] M. Eigensatz. Insights into the geometry of the gaussian kernel and an application in geometric modeling. Master's thesis, Swiss Federal Institute of Technology Zürich, Switzerland, 2006.

[60] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[61] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[62] P. Holmes, J.L. Lumley, G. Berkooz, and C.W. Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, Cambridge, United Kingdom, 2012.

[63] B. O. Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the national academy of sciences*, 17(5):315, 1931.

[64] M. Budišić, R. Mohr, and I. Mezić. Applied Koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4):047510, 2012.

[65] I. Mezić. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.

[66] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[67] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[68] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

[69] L. Molgedey and H. G. Schuster. Separation of a mixture of independent signals using time delayed correlations. *Physical review letters*, 72(23):3634, 1994.

[70] G. Pérez-Hernández, F. Paul, T. Giorgino, G. De Fabritiis, and F. Noé. Identification of slow molecular order parameters for markov model construction. *The Journal of chemical physics*, 139(1):07B604_1, 2013.

[71] G. Froyland and K. Padberg. Almost-invariant sets and invariant manifolds connecting probabilistic and geometric descriptions of coherent structures in flows. *Physica D: Nonlinear Phenomena*, 238(16):1507 – 1523, 2009.

[72] S. Klus, B. E. Husic, M. Mollenhauer, and F. Noé. Kernel methods for detecting coherent structures in dynamical data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(12):123112, 2019.

[73] H. Poincaré and D. Goroff (ed.). *New Methods of Celestial Mechanics*. American Institute of Physics, 1993.

[74] E. N. Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, 20:130, 1963.

[75] S. Smale. Differentiable dynamical systems. *Bull. Amer. Math. Soc.*, 73:797–817, 1967.

[76] J. D. Meiss. *Differential Dynamical Systems*. Society for Industrial and Applied Mathematics, 2007.

[77] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw. Geometry from a time series. *Phys. Rev. Let.*, 45:712, 1980.

[78] D. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, New York, 1995.

[79] R. Fischer. Sofic systems and graphs. *Monastsh. Math*, 80:179, 1975.

[80] R. Fischer. Graphs and symbolic dynamics. *Coll. Math. Soc. Janos Bolyai, 16. Topics in Information Theory*, 1975.

[81] W. Krieger. On sofic systems i. *Israel Journal of Mathematics*, 48(4):305–330, 1984.

[82] S. H. Strogatz. *Nonlinear Dynamics and Chaos: with applications to physics, biology, chemistry, and engineering*. Addison-Wesley, Reading, Massachusetts, 1994.

[83] J. Hadamard. Les surfaces à courbures opposées et leurs lignes géodésique. *Journal de Mathematiques Pures et Appliqué*, 4:27–73, 1898.

[84] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Prentice-Hall, New York, third edition, 2006.

[85] B. Weiss. Subshifts of finite type and sofic systems. *Monastsh. Math.*, 77:462, 1973.

[86] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[87] B. Kitchens and S. Tuncel. Semi-groups and graphs. *Israel. J. Math.*, 53:231, 1986.

[88] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

[89] N. Thomas, T. Smidt, S. M. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds. *ArXiv*, abs/1802.08219, 2018.

[90] A. Salova, J. Emenheiser, A. Rupe, J. P. Crutchfield, and R. M. DSouza. Koopman operator and its approximations for systems with symmetries. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(9):093128, 2019.

[91] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Champaign-Urbana, 1962.

[92] A. N. Kolmogorov. A new metric invariant of transient dynamical systems and automorphisms in Lebesgue spaces. *Dokl. Akad. Nauk. SSSR*, 119:861, 1958. (Russian) Math. Rev. vol. 21, no. 2035a.

[93] A. N. Kolmogorov. Entropy per unit time as a metric invariant of automorphisms. *Dokl. Akad. Nauk. SSSR*, 124:754, 1959. (Russian) Math. Rev. vol. 21, no. 2035b.

[94] Ya. G. Sinai. On the notion of entropy of a dynamical system. *Doklady of Russian Academy of Sciences*, 124:768, 1959.

[95] J. Milnor and W. Thurston. On iterated maps of the interval. *Springer Lecture Notes*, 1342:465–563, 1988.

[96] A. A. Brudno. The complexity of the trajectories of a dynamical system. *Russ. Math. Surv.*, 33(1):197–198, 1978. [Russian] Uspekhi Mat. Nauk 33:1 (1978), 207-208.

[97] A. A. Brudno. Entropy and the complexity of the trajectories of a dynamical system. *Trans. Moscow Math. Soc.*, 44:127–151, 1983. [Russian] Tr. Mosk. Mat. Obs., Moscow State University, Moscow, 1982, 124–149.

[98] P. Collet, J. P. Crutchfield, and J. P. Eckmann. Computing the topological entropy of maps. *Comm. Math. Phys.*, 88:257, 1983.

[99] J. P. Crutchfield and N. H. Packard. Symbolic dynamics of one-dimensional maps: Entropies, finite precision, and noise. *Intl. J. Theo. Phys.*, 21:433, 1982.

[100] S. Wolfram. Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, 55:601, 1983.

[101] S. Wolfram. Computation theory of cellular automata. *Comm. Math. Phys.*, 96:15, 1984.

[102] P. Grassberger. Toward a quantitative theory of self-generated complexity. *Intl. J. Theo. Phys.*, 25:907, 1986.

[103] J. P. Crutchfield and K. Young. Inferring statistical complexity. *Phys. Rev. Let.*, 63:105–108, 1989.

[104] J. P. Crutchfield. Between order and chaos. *Nature Physics*, 8(January):17–24, 2012.

[105] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, 1991.

[106] J. P. Crutchfield and B. S. McNamara. Equations of motion from a data series. *Complex Systems*, 1:417 – 452, 1987.

[107] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[108] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[109] S.-M. Udrescu and M. Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020.

[110] C.R. Shalizi. Optimal nonlinear prediction of random fields on networks. *Discrete Mathematics & Theoretical Computer Science*, 2003.

[111] E. Racah, C. Beckham, T. Maharaj, S. E. Kahou, Prabhat, and C. Pal. ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In *Advances in Neural Information Processing Systems*, pages 3402–3413, 2017.

[112] C.R. Shalizi, R. Haslinger, J.-B. Rouquier, K.L. Klinkner, and C. Moore. Automatic filters for the detection of coherent structure in spatiotemporal systems. *Phys. Rev. E*, 73(3):036104, 2006.

[113] A. Hadjighasem, M. Farazmand, D. Blazevski, G. Froyland, and G. Haller. A critical comparison of Lagrangian methods for coherent structure detection. *Chaos*, 27(5):053104, 2017.

[114] A. Rupe and J. P. Crutchfield. Spacetime symmetries, invariant sets, and additive subdynamics of cellular automata. *arXiv preprint arXiv:1812.11597*, 2018.

[115] A. Rupe and J. P. Crutchfield. Local causal states and discrete coherent structures. *Chaos*, 28(7):1–22, 2018.

[116] J. P. Crutchfield and J. E. Hanson. Turbulent pattern bases for cellular automata. *Physica D*, 69:279 – 301, 1993.

[117] G. Haller. Lagrangian coherent structures. *Ann. Rev. Fluid Mech.*, 47:137–162, 2015.

[118] T. Peacock and G. Haller. Lagrangian coherent structures: The hidden skeleton of fluid flows. *Physics Today*, 66(2):41–47, 2013.

[119] E. R. Weeks, J. S. Urbach, and H. L. Swinney. Anomalous diffusion in asymmetric random walks with a quasi-geostrophic flow example. *Physica D: Nonlinear Phenomena*, 97(1-3):291–310, 1996.

[120] T. Von Kármán. *Aerodynamics*, volume 9. McGraw-Hill New York, 1963.

[121] S. Chen and G. D.. Doolen. Lattice boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1):329–364, 1998.

[122] A. Rupe, N. Kumar, V. Epifanov, K. Kashinath, O. Pavlyk, F. Schlimbach, M. Patwary, S. Maidanov, V. Lee, Prabhat, and J. P. Crutchfield. Disco: Physics-based unsupervised discovery of coherent structures in spatiotemporal systems. In *2019 IEEE/ACM Workshop on Machine Learning in High Performance Computing Environments (MLHPC)*, pages 75–87. IEEE, 2019. arXiv:1909.11822 [physics.comp-ph].

[123] K. A. Emanuel. The dependence of hurricane intensity on climate. *Nature*, 326(6112):483, 1987.

[124] Peter J Webster, Greg J Holland, Judith A Curry, and H-R Chang. Changes in tropical cyclone number, duration, and intensity in a warming environment. *Science*, 309(5742):1844–1846, 2005.

[125] M. Mudigonda, S. Kim, A. Mahesh, S .Kahou, K. Kashinath, D. Williams, V. Michalski, T. OBrien, and Prabhat. Segmenting and tracking extreme climate events using neural networks. In *Deep Learning for Physical Sciences (DLPS) Workshop, held with NIPS Conference*, 2017.

[126] T. Kurth, S. Treichler, J. Romero, M. Mudigonda, N. Luehr, E. Phillips, A. Mahesh, M. Matheson, J. Deslippe, M. Fatica, Prabhat, and M. Houston. Exascale deep learning for climate analytics. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, page 51. IEEE Press, 2018.

[127] M. J. Chiyu, J. Huang, K. Kashinath, Prabhat, P. Marcus, and M. Niessner. Spherical CNNs on unstructured grids. In *International Conference on Learning Representations*, 2019.

[128] T. Cohen, M. Weiler, B. Kicanaoglu, and M. Welling. Gauge equivariant convolutional networks and the icosahedral CNN. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1321–1330, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[129] Prabhat, O. Rbel, S. Byna, K. Wu, F. Li, M. Wehner, W. Bethel, et al. TECA: A parallel toolkit for extreme climate analysis. In *Third Worskhop on Data Mining in Earth System Science (DMESS)*, 2012.

[130] C. A. Shields, J. J. Rutz, L.-Y. Leung, F. M. Ralph, M. Wehner, B. Kawzenuk, J. M. Lora, E. McClenny, T. Osborne, A. E. Payne, P. Ullrich, A. Gershunov, N. Goldenson, B. Guan, Y. Qian, A. M. Ramos, C. Sarangi, S. Sellars, I. Gorodetskaya, K. Kashinath, V. Kurlin, K. Mahoney, G. Muszynski, R. Pierce, A. C. Subramanian, R. Tome, D. Waliser, D. Walton, G. Wick, A. Wilson, D. Lavers, Prabhat, A. Collow, H. Krishnan, G. Magnusdottir, and P. Nguyen. Atmospheric river tracking method intercomparison project (ARTMIP): project goals and experimental design. *Geoscientific Model Development*, 11(6):2455–2474, 2018.

[131] Prabhat, Karthik Kashinath, Mayur Mudigonda, Kevin Yang, Jiayi Chen, Annette Grenier, and Benjamin Toms. ClimateNet: bringing the power of deep learning to the climate community via open datasets and architectures. https://www.nersc.gov/research-and-development/data-analytics/big-data-center/climatenet/, 2018.

[132] C. Olah, A. Satyanarayan, I. Johnson, S. Carter, L. Schubert, K. Ye, and A. Mordvintsev. The building blocks of interpretability. *Distill*, 2018. https://distill.pub/2018/building-blocks.

[133] P. W. Anderson. More is different. *Science*, 177(4047):393–396, 1972.

[134] S. Wolfram. Undecidability and intractability in theoretical physics. *Physical Review Letters*, 54(8):735, 1985.

[135] M. Gu, C. Weedbrook, A. Perales, and M. A. Nielsen. More really is different. *Physica D: Nonlinear Phenomena*, 238(9-10):835–839, 2009.

[136] M. Matthew. Universality in elementary cellular automata. *Complex Systems*, 15(1):1–40, 2004.

[137] R. S. MacKay. Nonlinearity in complexity science. *Nonlinearity*, 21(12):T273, 2008.

[138] J. Ladyman, J. Lambert, and K. Wiesner. What is a complex system? *European Journal for Philosophy of Science*, 3(1):33–67, 2013.

[139] S. Aaronson, S. M. Carroll, and L. Ouellette. Quantifying the rise and fall of complexity in closed systems: The coffee automaton. *arXiv preprint arXiv:1405.6903*, 2014.

[140] S. Aaronson. NP-complete problems and physical reality. *ACM Sigact News*, 36(1):30–52, 2005.

[141] C. Moore. Unpredictability and undecidability in dynamical systems. *Phys. Rev. Lett.*, 64:2354, 1990.

[142] C. Moore and S. Mertens. *The nature of computation*. Oxford, 2011.

[143] J. Kari. Rice's theorem for the limit sets of cellular automata. *Theoretical computer science*, 127(2):229–254, 1994.

[144] S. Wolfram. Cellular automata as models of complexity. *Nature*, 311:419, 1984.

[145] N. Israeli and N. Goldenfeld. Computational irreducibility and the predictability of complex physical systems. *Physical review letters*, 92(7):074105, 2004.

[146] C. Moore. Majority-vote cellular automata, Ising dynamics, and P-completeness. *Journal of Statistical Physics*, 88(3-4):795–805, 1997.

[147] T. Neary and D. Woods. P-completeness of cellular automaton Rule 110. In *International Colloquium on Automata, Languages, and Programming*, pages 132–143. Springer, 2006.

[148] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[149] Scott Aaronson. $P \stackrel{?}{=} NP$. In *Open problems in mathematics*, pages 1–122. Springer, 2016.

[150] H. Ye, R. J. Beamish, S. M. Glaser, S. C.H. Grant, C.-H. Hsieh, L. J. Richards, J. T. Schnute, and G. Sugihara. Equation-free mechanistic ecosystem forecasting using empirical dynamic modeling. *Proceedings of the National Academy of Sciences*, 112(13):E1569–E1576, 2015.

[151] M. J. Block. Surface tension as the cause of Bénard cells and surface deformation in a liquid film. *Nature*, 178(4534):650–651, 1956.

[152] J.R.A. Pearson. On convection cells induced by surface tension. *Journal of fluid mechanics*, 4(5):489–500, 1958.

[153] M. F. Schatz, S. J. VanHook, W. D. McCormick, J.B. Swift, and H. L. Swinney. Onset of surface-tension-driven Bénard convection. *Physical review letters*, 75(10):1938, 1995.

[154] J.W. Scanlon and L.A. Segel. Finite amplitude cellular convection induced by surface tension. *Journal of Fluid Mechanics*, 30(1):149–162, 1967.

[155] A. Cloot and G. Lebon. A nonlinear stability analysis of the Bénard–Marangoni problem. *Journal of Fluid Mechanics*, 145:447–469, 1984.

[156] A. Church. A note on the entscheidungsproblem. *J. Symbolic Logic*, 1:40–41, 1936.

[157] A. M. Turing. On computable numbers, with an application to the entsheidungsproblem. *Proc. Lond. Math. Soc. Ser. 2*, 42:230, 1936.

[158] R. Gandy. Church's thesis and principles for mechanisms. In *Studies in Logic and the Foundations of Mathematics*, volume 101, pages 123–148. Elsevier, 1980.

[159] O. Copeland, B. J.and Shagrir. Physical computation: How general are gandys principles for mechanisms? *Minds and Machines*, 17(2):217–231, 2007.

[160] A. Ginzburg. *Algebraic theory of automata*. Academic Press, 1968.

[161] K. Young. *The Grammar and Statistical Mechanics of Complex Physical Systems*. PhD thesis, University of California, Santa Cruz, 1991. published by University Microfilms Intl, Ann Arbor, Michigan.

[162] M. Casdagli and S. Eubank, editors. *Nonlinear Modeling*, SFI Studies in the Sciences of Complexity, Reading, Massachusetts, 1992. Addison-Wesley.

[163] N. Rubido, C. Grebogi, and M. S. Baptista. Entropy-based generating Markov partitions for complex systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(3):033611, 2018.

[164] L.-S. Young. Entropy in dynamical systems. *Entropy*, 313, 2003.

[165] R. M. May. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467, 1976.

[166] P. Collet and J.-P. Eckmann. *Maps of the Unit Interval as Dynamical Systems*. Birkhauser, Berlin, 1980.

[167] Y. B. Pesin. Characteristic lyapunov exponents and smooth ergodic theory. *Russian Mathematical Surveys*, 32(4):55–112, 1977.

[168] D. Ruelle. An inequality for the entropy of differentiable maps. *Boletim da Sociedade Brasileira de Matemática-Bulletin/Brazilian Mathematical Society*, 9(1):83–87, 1978.

[169] C. R. Shalizi and J. P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *J. Stat. Phys.*, 104:817–879, 2001.

[170] G. A. Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Theory of Computing Systems*, 3(4):320–375, 1969.

[171] B. Kitchens and S. Tuncel. Finitary measures for subshifts of finite type and sofic systems. *Memoirs of the AMS*, 58:no. 338, 1985.

[172] M. Boyle and K. Petersen. Hidden Markov processes in the context of symbolic dynamics. *arXiv preprint arXiv:0907.1858*, 2009.

[173] J.L Schiff. *Cellular Automata: a Discrete View of the World*, volume 45. John Wiley & Sons, 2011.

[174] S. Wolfram. *Theory and Applications of Cellular Automata*. World Scientific Publishers, Singapore, 1986.

[175] T. Ceccherini-Silberstein and M. Coornaert. *Cellular Automata and Groups*. Springer Science & Business Media, 2010.

[176] D. Lind. Multi-dimensional symbolic dynamics. In *Symbolic Dynamics and its Applications*, volume 60, pages 61–80. American Mathematical Society, 2004.

[177] K. Schmidt. *Dynamical systems of algebraic origin*. Springer Science & Business Media, 1995.

[178] M. Hochman. Multidimensional shifts of finite type and sofic shifts. In *Combinatorics, Words and Symbolic Dynamics*, pages 296–359. Cambridge University Press, 2016.

[179] R. Berger. *The undecidability of the domino problem*. Number 66. American Mathematical Soc., 1966.

[180] R. M. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones mathematicae*, 12(3):177–209, 1971.

[181] D. R. Upper. *Theory and Algorithms for Hidden Markov Models and Generalized Hidden Markov Models*. PhD thesis, University of California, Berkeley, 1997. Published by University Microfilms Intl, Ann Arbor, Michigan.

[182] N. Travers and J. P. Crutchfield. Equivalence of history and generator ϵ-machines. *Physical Review E*, page in press, 2014. SFI Working Paper 11-11-051; arxiv.org:1111.4500 [math.PR].

[183] J. Ruebeck, R. G. James, John R. Mahoney, and J. P. Crutchfield. Prediction and generation of binary Markov processes: Can a finite- state fox catch a markov mouse? *CHAOS*, 28:013109, 2018.

[184] J. P. Crutchfield and D. P. Feldman. Regularities unseen, randomness observed: Levels of entropy convergence. *CHAOS*, 13(1):25–54, 2003.

[185] R. G. James, C. J. Ellison, and J. P. Crutchfield. Anatomy of a bit: Information in a time series observation. *CHAOS*, 21(3):037109, 2011.

[186] J. P. Crutchfield, C. J. Ellison, and J. R. Mahoney. Time's barbed arrow: Irreversibility, crypticity, and stored information. *Phys. Rev. Lett.*, 103(9):094101, 2009.

[187] J. P. Crutchfield, C. J. Ellison, J. R. Mahoney, and R. G. James. Synchronization and control in intrinsic and designed computation: An information-theoretic analysis of competing models of stochastic computation. *CHAOS*, 20(3):037105, 2010. Santa Fe Institute Working Paper 10-08-015; arxiv.org:1007.5354 [cond-mat.stat-mech].

[188] J. P. Crutchfield, P. Riechers, and C. J. Ellison. Exact complexity: Spectral decomposition of intrinsic computation. *Phys. Lett. A*, 380(9-10):998–1002, 2015.

[189] P. M. Ara, R. G. James, and J. P. Crutchfield. The elusive present: Hidden past and future dependence and why we build models. *Phys. Rev. E*, 93(2):022143, 2016. SFI Working Paper 15-07-024; arxiv.org:1507.00672 [cond-mat.stat-mech].

[190] A. N. Kolmogorov. Three approaches to the concept of the amount of information. *Prob. Info. Trans.*, 1:1, 1965.

[191] G. Chaitin. On the length of programs for computing finite binary sequences. *J. ACM*, 13:145, 1966.

[192] M. Li and P. M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, New York, 1993.

[193] J. P. Crutchfield. Discovering coherent structures in nonlinear spatial systems. In A. Brandt, S. Ramberg, and M. Shlesinger, editors, *Nonlinear Ocean Waves*, pages 190–216, Singapore, 1992. World Scientific. also appears in Complexity in Physics and Technology, R. Vilela-Mendes, editor, World Scientific, Singapore (1992).

[194] J. P. Crutchfield and J. E. Hanson. Attractor vicinity decay for a cellular automaton. *CHAOS*, 3(2):215–224, 1993.

[195] A. Rupe and J. P. Crutchfield. Spatiotemporal computational mechanics. 2020. in preparation.

[196] K. Lindgren, C. Moore, and M. Nordahl. Complexity of two-dimensional patterns. *J. Stat. Phys.*, 91:909–951, 1998.

[197] J. P. Crutchfield. Semantics and thermodynamics. In M. Casdagli and S. Eubank, editors, *Nonlinear Modeling and Forecasting*, volume XII of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 317 – 359, Reading, Massachusetts, 1992. Addison-Wesley.

[198] J. P. Crutchfield and M. Mitchell. The evolution of emergent computation. *Proc. Natl. Acad. Sci.*, 92:10742–10746, 1995.

[199] C. S. McTague and J. P. Crutchfield. Automated pattern discovery—An algorithm for constructing optimally synchronizing multi-regular language filters. *Theo. Comp. Sci.*, 359(1-3):306–328, 2006.

[200] J. G. Brookshear. *Theory of computation: Formal languages, automata, and complexity*. Benjamin/Cummings, Redwood City, California, 1989.

[201] W.-K. Tung. *Group Theory in Physics: an Introduction to Symmetry Principles, Group Representations, and Special Functions in Classical and Quantum Physics*. World Scientific Publishing Co Inc, Philidelphia, 1985.

[202] M.V. Lawson. *Inverse Semigroups: the Theory of Partial Symmetries*. World Scientific, 1998.

[203] J. Rhodes. *Applications of Automata Theory and Algebraic via the Mathematical Theory of Complexity to Biology, Physics, Psychology, Philosophy, Games, and Codes*. University of California, Berkeley, California, 1971. C. Nehaniv, editor, World Scientific Publishing Company, Singapore (2009).

[204] W. M. L. Holcombe. *Algebraic Automata Theory*. Cambridge University Press, Cambridge, 1982.

[205] H. Haken. *Synergetics, An Introduction*. Springer, Berlin, third edition, 1983.

[206] J. P. Sethna. Order parameters, broken symmetry, and topology. *arXiv:9204009*.

[207] M. Livio. Physics: Why symmetry matters. *Nature*, 490(7421):472–473, 2012.

[208] E.D. Siggia and A. Zippelius. Dynamics of defects in rayleigh-bénard convection. *Physical Review A*, 24(2):1036, 1981.

[209] B. D. Johnson, J. P. Crutchfield, C. J. Ellison, and C. S. McTague. Enumerating finitary processes. arxiv.org:1011.0036.

[210] J. P. Crutchfield and C. S. McTague. Unveiling an enigma: Patterns in elementary cellular automaton 22 and how to discover them. 2002. Santa Fe Institute Technical Report.

[211] J. D. Farmer, T. Toffoli, and S. Wolfram, editors. *Cellular Automata, Proceedings of an Interdisciplinary Workshop*, Amsterdam, 1984. North-Holland Publishing Company.

[212] N. H. Packard and S. Wolfram. Two-dimensional cellular automata. *J. Stat. Physics*, 38(5–6):901–946, 1985.

[213] T. Toffoli and N. Margolis. *Cellular Automata Machines: A New Environment for Modeling*. MIT Press, Cambridge, Massachusetts, 1987.

[214] H. Gutowitz. *Cellular Automata: Theory and Experiment*. Special Issues of Physica D. Bradford Books, Cambridge, Massachusetts, 1991.

[215] O. Martin, A. Odlyzko, and S. Wolfram. Algebraic properties of cellular automata. *Commun. Math. Phys.*, 93:219, 1984.

[216] D. A. Lind. Applications of ergodic theory and sofic systems to cellular automata. *Physica*, 10D:36, 1984.

[217] P. Grassberger. New mechanism for deterministic diffusion. *Phys. Rev. A*, 28:3666, 1983.

[218] E. Jen. Exact solvability and quasiperiodicity of one-dimensional cellular automata. *Nonlinearity*, 4:251, 1990.

[219] C. Moore. Quasilinear cellular automata. *Physica D: Nonlinear Phenomena*, 103(1-4):100–132, 1997.

[220] J. Gravner and D. Griffeath. The one-dimensional Exactly 1 cellular automaton: Replication, periodicity, and chaos from finite seeds. *J. Stat. Physics*, 142(1):168–200, 2011.

[221] L. Le Bruyn and M. Van den Bergh. Algebraic properties of linear cellular automata. *Linear Algebra and its Applications*, 157:217–234, 1991.

[222] D. A. Lind. The structure of skew products with ergodic group automorphisms. *Israel J. Math.*, 28(3):205–248, 1977.

[223] K. Eloranta. Partially permutive cellular automata. *Nonlinearity*, 6(6):1009, 1993.

[224] P. Grassberger. Long-range effects in an elementary cellular automaton. *J. Stat. Physics*, 45(1-2):27–39, 1986.

[225] M. R. Allshouse and T. Peacock. Lagrangian based methods for coherent structure detection. *Chaos*, 25(9):097617, 2015.

[226] T. Sapsis and G. Haller. Inertial particle dynamics in a hurricane. *J. Atmos. Sci.*, 66(8):2481–2492, 2009.

[227] F. Vitart, J.L. Anderson, and W.F. Stern. Simulation of interannual variability of tropical storm frequency in an ensemble of GCM integrations. *J. Climate*, 10(4):745–760, 1997.

[228] K. Walsh and I.G. Watterson. Tropical cyclone-like vortices in a limited area model: Comparison with observed climatology. *J. Climate*, 10(9):2240–2259, 1997.

[229] Prabhat, S. Byna, V. Vishwanath, E. Dart, M. Wehner, W. D. Collins, et al. TECA: Petascale pattern recognition for climate science. In *International Conference on Computer Analysis of Images and Patterns*, pages 426–436. Springer, 2015.

[230] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

[231] H. Jänicke, A. Wiebel, G. Scheuermann, and W. Kollmann. Multifield visualization using local statistical complexity. *IEEE Trans. Vis. Comp. Graphics*, 13(6):1384–1391, 2007.

[232] H. Jänicke and G. Scheuermann. Towards automatic feature-based visualization. In *Dagstuhl Follow-Ups*, volume 1. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2010.

[233] G.M Goerg and C.R. Shalizi. LICORS: Light cone reconstruction of states for non-parametric forecasting of spatio-temporal systems. *arXiv:1206.2398*.

[234] G.M. Goerg and C.R. Shalizi. Mixed LICORS: A nonparametric algorithm for predictive state reconstruction. In *Artificial Intelligence and Statistics*, pages 289–297, 2013.

[235] J.T. Lizier, M. Prokopenko, and A.Y. Zomaya. Local information transfer as a spatiotemporal filter for complex systems. *Phys. Rev. E*, 77(2):026110, 2008.

[236] J. Lizier, M. Prokopenko, and A. Zomaya. Information modification and particle collisions in distributed computation. *CHAOS*, 20(3):037109, 2010.

[237] B. Flecker, W. Alford, J. M. Beggs, P. L. Williams, and R. D. Beer. Partial information decomposition as a spatiotemporal filter. *CHAOS*, 21(3):037104, 2011.

[238] J.T. Lizier, B. Flecker, and P.L. Williams. Towards a synergy-based approach to measuring information modification. In *Artificial Life (ALIFE), 2013 IEEE Symposium on*, pages 43–51. IEEE, 2013.

[239] R. G. James, N. Barnett, and J. P. Crutchfield. Information flows? A critique of transfer entropies. *Phys. Rev. Lett.*, 116(23):238701, 2016.

[240] R. G. James and J. P. Crutchfield. Multivariate dependence beyond shannon information. *Entropy*, 19:531, 2017.

[241] K. Eloranta and E. Nummelin. The kink of cellular automaton rule 18 performs a random walk. *J. Stat. Phys.*, 69:1131–1136, 1992.

[242] N. Boccara, J. Nasser, and M. Roger. Particlelike structures and their interactions in spatiotemporal patterns generated by one-dimensional deterministic cellular-automaton rules. *Phys. Rev. A*, 44:866, 1991.

[243] K. Eloranta. The dynamics of defect ensembles in one-dimensional cellular automata. *J. Stat. Phys.*, 76(5/6):1377–1398, 1994.

[244] Y. Liu, E. Racah, Prabhat, J. Correa, A. Khosrowshahi, D. Lavers, K. Kunkel, M. Wehner, and W. Collins. Application of deep convolutional neural networks for detecting extreme weather in climate datasets. *arXiv:1605.01156*.

[245] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature Comm.*, 5:4308, 2014.

[246] Gordon Bell, Tony Hey, and Alex Szalay. Beyond the data deluge. *Science*, 323(5919):1297–1298, 2009.

[247] Jonathan T. Overpeck, Gerald A. Meehl, Sandrine Bony, and David R. Easterling. Climate data challenges in the 21st century. *Science*, 331(6018):700–702, 2011.

[248] L. C. Gillet, A. Leitner, and R. Aebersold. Mass spectrometry applied to bottom-up proteomics: Entering the high-throughput era for hypothesis testing. *Annual Review of Analytical Chemistry*, 9:449–472, 2016.

[249] Eric Mjolsness and Dennis DeCoste. Machine learning for science: State of the art and future prospects. *Science*, 293(5537):2051–2055, 2001.

[250] Pedro Larranaga, Borja Calvo, Roberto Santana, Concha Bielza, Josu Galdiano, Inaki Inza, José A Lozano, Rubén Armananzas, Guzmán Santafé, Aritz Pérez, and Victor Robles. Machine learning in bioinformatics. *Briefings in Bioinformatics*, 7(1):86–112, 2006.

[251] Keith T Butler, Daniel W Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547, 2018.

[252] Nicola Jones. How machine learning could help to improve climate forecasts. *Nature News*, 548(7668):379, 2017.

[253] Jordan Venderley, Vedika Khemani, and Eun-Ah Kim. Machine learning out-of-equilibrium phases of matter. *Phys. Rev. Lett.*, 120:257204, Jun 2018.

[254] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5):851–869, 2017.

[255] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, 2017.

[256] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and Prabhat. Deep learning and process understanding for data-driven Earth system science. *Nature*, 566(7743):195, 2019.

[257] Wahid Bhimji, Steven Andrew Farrell, Thorsten Kurth, Michela Paganini, Prabhat, and Evan Racah. Deep neural networks for physics analysis on low-level whole-detector data at the LHC. In *Journal of Physics: Conference Series*, volume 1085, page 042034. IOP Publishing, 2018.

[258] Amrita Mathuriya, Deborah Bard, Peter Mendygral, Lawrence Meadows, James Arnemann, Lei Shao, Siyu He, Tuomas Kärnä, Diana Moise, Pennycook, Kristyn Maschhoff, Jason Sewall, Nalini Kumar, Shirley Ho, Michael F. Ringenburg, Prabhat, and Victor Lee. CosmoFlow: Using deep learning to learn the universe at scale. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 819–829. IEEE, 2018.

[259] Michael F Wehner, G Bala, Phillip Duffy, Arthur A Mirin, and Raquel Romano. Towards direct simulation of future tropical cyclone statistics in a high-resolution global atmospheric model. *Advances in Meteorology*, 2010, 2010.

[260] George D Montanez and Cosma Rohilla Shalizi. The LICORS cabinet: Nonparametric algorithms for spatio-temporal prediction. *arXiv preprint arXiv:1506.02686*, 2015.

[261] H. Jaeger. The echo state approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2001.

[262] T. A. OBrien, K. Kashinath, N. R. Cavanaugh, W. D. Collins, and J. P. OBrien. A fast and objective multidimensional kernel density estimation method: fastKDE. *Computational Statistics & Data Analysis*, 101:148 – 160, 2016.

[263] Zahra Ronaghi, Rollin Thomas, Jack Deslippe, Stephen Bailey, Doga Gursoy, Theodore Kisner, Reijo Keskitalo, and Julian Borrill. Python in the nersc exascale science applications program for data. In *Proceedings of the 7th Workshop on Python for High-Performance and Scientific Computing*, page 4. ACM, 2017.

[264] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[265] Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[266] Ian Jolliffe. *Principal Component Analysis*. Springer, 2011.

[267] Alexis Tantet, Fiona R. van der Burgt, and Henk A. Dijkstra. An early warning indicator for atmospheric blocking events using transfer operators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(3):036406, 2015.

[268] X. Xia and B. Kulis. W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint arXiv:1711.08506*, 2017.

[269] Parallel k-means data clustering. http://www.ece.northwestern.edu/~wkliao/Kmeans/index.html, Accessed: 2019-04-08.

[270] Parallel k-means data clustering for large data sets. http://www.ece.northwestern.edu/~wkliao/Kmeans/kmeans_int64.html, Accessed: 2019-04-08.

[271] Sunwoo Lee, Wei-keng Liao, Ankit Agrawal, Nikos Hardavellas, and Alok Choudhary. Evaluation of k-means data clustering algorithm on intel xeon phi. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 2251–2260. IEEE, 2016.

[272] Liandeng Li, Teng Yu, Wenlai Zhao, Haohuan Fu, Chenyu Wang, Li Tan, Guangwen Yang, and John Thomson. Large-scale hierarchical k-means for heterogeneous many-core supercomputers. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 160–170. IEEE, 2018.

[273] Md Mostofa Ali Patwary, Suren Byna, Nadathur Rajagopalan Satish, Narayanan Sundaram, Zarija Lukić, Vadim Roytershteyn, Michael J Anderson, Yushu Yao, Pradeep Dubey, et al. Bd-cats: big data clustering at trillion particle scale. In *SC'15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12. IEEE, 2015.

[274] Mostofa Ali Patwary, Diana Palsetia, Ankit Agrawal, Wei-keng Liao, Fredrik Manne, and Alok Choudhary. Scalable parallel optics data clustering using graph algorithmic techniques. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 49. ACM, 2013.

[275] Inc. Petuum. Scalable clustering for exploratory data analysis. https://medium.com/@Petuum/scalable-clustering-for-exploratory-data-analysis-60b27ea0fb06, Accessed: 2019-04-08.

[276] Yike Guo and Robert Grossman. A fast parallel clustering algorithm for large spatial databases, high performance data mining. 2002.

[277] Xu Hu, Jun Huang, Minghui Qiu, Cen Chen, and Wei Chu. Ps-dbscan: An efficient parallel dbscan algorithm based on platform of ai (pai). *arXiv preprint arXiv:1711.01034*, 2017.

[278] Md Mostofa Ali Patwary. Pdsdbscan source code. *Web page http://users. eecs. northwestern. edu/˜ mpatwary/Software. html*, 2015.

[279] Markus Götz, Christian Bodenstein, and Morris Riedel. Hpdbscan: highly parallel dbscan. In *Proceedings of the workshop on machine learning in high-performance computing environments*, page 2. ACM, 2015.

[280] Intel. Fast, scalable and easy machine learning with daal4py. https://intelpython.github.io/daal4py/index.html, Accessed: 2019-04-08.

[281] NASA. Jupiter cloud sequence from cassini. https://svs.gsfc.nasa.gov/cgi-bin/details.cgi?aid=3610, Accessed: 2019-03-14.

[282] M. Farazmand. An adjoint-based approach for finding invariant solutions of navier-stokes equations. *J. Fluid Mech.*, 795:278–312, 2016.

[283] Michael F. Wehner, Kevin Reed, Fuyu Li, Prabhat, Julio Bacmeister, Cheng-Ta Chen, Chris Paciorek, Peter Gleckler, Ken Sperber, William D. Collins, Andrew Gettelman, and Christiane Jablonowski. The effect of horizontal resolution on simulation quality in the community atmospheric model, cam5.1. *Journal of Modeling the Earth System*, 06:980–997, 2014.

[284] Sergey Maidanov. Performing numerical analysis and data analytics with python at scale. https://www.ixpug.org/documents/1526053887xpug-maidanov-scalablescience.pdf, Accessed: 2019-04-08.

[285] Brenden Epps. Review of vortex identification methods. In *55th AIAA Aerospace Sciences Meeting*, page 0989, 2017.

[286] Alireza Hadjighasem and George Haller. Geodesic transport barriers in Jupiter's atmosphere: Video-based analysis. *Siam Review*, 58(1):69–89, 2016.

[287] G. Haller, A. Hadjighasem, M. Farazmand, and F. Huhn. Defining coherent vortices objectively from the vorticity. *Journal of Fluid Mechanics*, 795:136–173, 2016.

[288] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[289] R. B. Marimont and M. B. Shapiro. Nearest Neighbour Searches and the Curse of Dimensionality. *IMA Journal of Applied Mathematics*, 24(1):59–70, 08 1979.

[290] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. OPTICS: Ordering points to identify the clustering structure. In *ACM Sigmod record*, volume 28, pages 49–60. ACM, 1999.

[291] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205, 2017.