# SPATIOTEMPORAL TILING OF THE KURAMOTO-SIVASHINSKY EQUATION

A Thesis
Presented to
The Academic Faculty

by

Matthew N. Gudorf

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Physics

Georgia Institute of Technology
December, 2020

# SPATIOTEMPORAL TILING OF THE KURAMOTO-SIVASHINSKY EQUATION

# ACKNOWLEDGEMENTS

I would like to first and foremost thank my advisor Predrag Cvitanović for his seemingly inexhaustible amount of patience and understanding regarding both my Physics research as well as my mental health. At the same time, I thank everyone who has assisted my research. I thank Xiong Ding and Burak Budanur for their introductions to spatiotemporal chaos, the Kuramoto-Sivashinsky equation, and computational Physics. I thank Mohammad Farazmand, Simon Berman, John Gibson, Ashley Willis, Pallantla Ravi Kumar, for discussions regarding nonlinear dynamics and chaos. I would like to also thank my thesis committee members Michael Schatz, Kurt Wiesenfeld, Flavio Fenton, Luca Dieci for their understanding during this tumultuous year of 2020. I thank my friends who helped me through the entire process of receiving this PhD. Lastly I need to thank my family who made this endeavor possible and less stressful with their endless support, I love you all.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# SUMMARY

Motivated by translational invariance and exponentially unstable dynamics, 'spatiotemporally chaotic' or 'turbulent' flows are recast as a (D+1)-dimensional spatiotemporal theory which treats space and time equally. Time evolution is replaced by a repertoire of spatiotemporal patterns taking the form of (D+1) dimensional invariant tori (periodic orbits). Our claim is that the entirety of spacetime can be described as the shadowing of a finite collection of 'fundamental orbits'; periodic orbits of small spatiotemporal extent. We demonstrate that not only can fundamental orbits be extracted from larger orbits, they can also be used as the 'building blocks' of turbulence. That is, they can be combined spatiotemporally to find both known and new solutions. In the future we aim to explain all of these results by constructing a (D+1)-dimensional symbolic dynamics whose alphabet is the set of fundamental orbits, however, in order to do so we must first confirm that all fundamental orbits have been found. These ideas are investigated in the context of the 1+1 dimensional spacetime of the Kuramoto-Sivashinsky equation using the independently developed Python package 'orbithunter'. This package intends to serve as a user friendly and powerful framework for solving chaotic nonlinear partial differential equations; currently only implemented for the Kuramoto-Sivashinsky equation, however. This package gives access to completely original spatiotemporal techniques, such as spatiotemporal clipping and spatiotemporal gluing of orbits. A short preview of these techniques and how to generalize them to other equations is also provided. By demonstrating robust and powerful spatiotemporal techniques this work hopes to inspire others to take up the spatiotemporal mantle and work towards a pattern based description of turbulence.

# CHAPTER I

# INTRODUCTION

Turbulence stands as one of the few remaining classical Physics problems yet to be fully explained. Turbulence, and other spatiotemporally chaotic processes, typically take the form of deterministic, nonlinear partial differential equations. The prototypical example of a chaotic nonlinear system is the Navier-Stokes (NS) equation

$$\mathbf{u}_t + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{\nabla p}{\rho} + \nu \nabla^2 \mathbf{u} + \mathbf{f}\,. \tag{1}$$

Using a 'dynamical systems formulation' [132], the time evolution prescribed by (1) can be posed as the traversal of an $\infty$-dimensional state space. While $\infty$-dimensional, dissipative and strongly contracting flows are typically contained within finite-dimensional attractors or 'inertial manifolds' [21, 38, 110, 126, 127] in non-trivial, nonlinear ways [28, 49, 50, 125, 143, 144]. For the Navier-Stokes equation (1) the existence of such a manifold has not yet been proven; however, spatiotemporal recurrences and dissipation seem to imply its existence. For specific geometries such as plane-Couette flow [47] and pipe flow [13], it has been shown that time invariant solutions: equilibria, periodic orbits, relative periodic orbits (often referred to as 'exact coherent structures' in the fluid dynamics community [137, 138, 141]) shape the geometry of the state space via their stable and unstable manifolds. These computational results and their impact cannot be understated. Nearly all of these computational successes have occurred on very small spatial domains, however. These domains, commonly referred to as *minimal cells*, are large enough to support turbulence yet small enough such that finding time invariant solutions remains computationally tractable [47, 48, 94, 135]. The inability to extend these successes to larger spatial domains is the primary motivation behind this body of work. The claim put forth here is that exponentially unstable dynamical equations, those which have positive Lyapunov exponents [84, 104], are not the way forward for turbulence research. Instead, we aim to explain turbulent flows as the shadowing of members of a finite set of admissible spatiotemporal patterns [23]. Infinite spacetime can then be described as a collection of such patterns: no dynamics required.

These ideas are explored in the context of the Kuramoto-Sivashinsky equation

$$u_t + u_{xx} + \nu u_{xxxx} + \frac{1}{2}(u^2)_x = 0\,. \tag{2}$$

a chaotic nonlinear reaction-diffusion equation [74, 117] often used as a numerical proving ground for new approaches to turbulence [69, 107]. In-depth discussion of the Kuramoto-Sivashinsky equation itself is reserved until sect. 1.1. The proposal is to study the infinite, 2-dimensional spacetime of the Kuramoto-Sivashinsky equation by developing a 2-dimensional symbolic 'dynamics' (for lack of a better word) whose alphabet consists of fundamental spatiotemporal patterns. Each of these patterns is a invariant 2-torus of minimal spatiotemporal extent, denoted *fundamental periodic orbits*. In this symbolic representation, the columns code admissible time itineraries and rows encode the admissible spatial profiles. The admissibility of these spatiotemporal patterns is determined by the *grammar* [24] of the corresponding symbolic alphabet. Our claim is that the entirety of spacetime is explained by the shadowing of these fundamental orbits. In other words, fundamental orbits are the so called 'building blocks' of turbulence. This is an attractive proposition, as there have been many attempts to find the fundamental structures necessary for sustaining turbulence in the Navier-Stokes equations, the 'self-sustaining process' proposed by Waleffe [136] serving as a prime example. It is possible that the proper structures have already been determined but they have simply not been utilized properly, that is, they have not been used in conjunction with a truly spatiotemporal theory.

Continuing with the description of fundamental orbits, the spatiotemporal dimensions are assumed to be on the order of the important physical scales of whichever equation is in question. Alternatively, it may be that the fundamental orbits and their spatiotemporal configurations *decide* upon the physical scales, not the other way around. Provocative statements such as this may be hard to digest for practitioners of dynamical systems theory, or more generally, physicists. While we do not want to discard all physics related knowledge of the Kuramoto-Sivashinsky equation, avoiding descriptions based on time dependent processes is necessary. The hypothesis that the most important orbits are the smallest originates from the theory of cycle expansions [2]. In these expansions, the dominant contributions are provided by the shortest cycles (smallest orbits). The higher order terms (larger orbits) of such expansions serve as curvature corrections [24].

Treatment of time and space in this manner is a new idea but analyzing fundamental geometrical shapes and how they combine is not. In fact, the idea of there being fundamental 'cells' or volumes is not new either; until time is included. In their manuscript on pattern formation, Cross and Hohenberg [22] discuss characteristic lengths and their relation to the fractal dimension of attractors; defining *extensive chaos* in the process. They conjecture that in the asymptotic limit of system size, the fractal dimension of attractors scale like $L^d$ where $d$ is the Euclidean dimension of the system. This, they claim, provides a different correlation length $\xi_f$ such that the system is comprised of cells of volume $\xi_f^d$ and the number of cells within the volume $(L/\xi_f)^d$ gives the fractal dimension of the attractor. The fundamental orbits have a similar flavor to this; except that now time is included as well, such that these cells are now spatiotemporal.

Study of important patterns occurs in other disciplines as well. There is the study of topological defects in the context of nematic liquid crystals [34] and cosmology [133] as well as the study of motifs in complex networks [87, 89], for a few examples. In the case of liquid crystals, defects carry an additional energy cost. In cosmology, topological defects are suspected as the possible source for the structures seen in the universe today [8]. Motifs in complex networks are important because they represent subgraphs which appear much more frequently than one would expect in randomized networks. Certain patterns seem to be specific to different categories of networks (biological, technological, etc.) [89]. Clearly these patterns and defects are special and deserve in-depth investigation; of course, in our context these take the form of fundamental orbits.

In order to make any progress the computation problem must be tractable. This is believed to be the case as the number of fundamental orbits is believed to be 'small in number'. More precisely, the quantity which is actually believed to be 'small in number' are the number of unique fundamental orbit *families*. Each element of these continuous families has its own corresponding group orbit produced by continuous and discrete spatiotemporal symmetries. Note that using the phrase 'small in number' might be somewhat disingenuous, as we are technically referring to an infinite set of solutions. These continuous families are actually very useful; especially in the description of spatiotemporal shadowing of fundamental orbits. Shadowing is not an exact process; each shadowing event is not identical, even if the underlying fundamental orbit being shadowed is believed to be the same. Similar, but not identical, regions of spacetime can be explained by the shadowing of different members of the same continuous family of orbits. This explanation of shadowing shares an intimate relationship with the namesake of this thesis work, *spatiotemporal tiling*, a concept which we believe is best presented visually. To begin, we note that all doubly periodic solutions are infinite spacetime solutions by definition. The 'tiling' of an orbit is generated by discrete translations of a compact fundamental domain. An example of such a tiling is displayed in figure 2. When we make the statement that a region of spacetime is being shadowed by an orbit, we mean that *locally*, the infinite tiling in figure 2 is being shadowed. In other words, if we were to superimpose figure 2 (or its translations) on top of figure 1, the regions of spacetime where 'overlaps' occur represent regions of spacetime where the tiling orbit is being shadowed. A crude demonstration of shadowing is displayed in figure 3. Clearly, the example tiling in figure 2 does not capture the complexity of figure 1. In other words a single orbit tiling cannot account for the majority of space time; however, by including more orbits in the description, more spacetime can be explained by shadowing. We can demonstrate this by comparing figure 3 and figure 4. In figure 3, only a single orbit is used, while in figure 4, multiple, symmetry related orbits are used. While this is a very crude calculation, we believe it not only shows that more space-time can be covered by including 'more orbits', but also that a single spatiotemporal pattern and its symmetry related copies are shadowed frequently (although not uniformly) throughout spacetime.

The goal is clear then: classify and enumerate all fundamental orbits and then use them to reconstruct the entirety of spacetime; simple, right? This is akin to the statement that all storms; past, present and future, can be derived by capturing the fundamental spatiotemporal cloud formations. Clearly this is no small task, but we proceed in the following manner: To begin, the Kuramoto-Sivashinsky equation is introduced and some motivating preliminary investigations are described. The Kuramoto-Sivashinsky equation is then formulated as a spatiotemporal system; the equation being rewritten as a system of differential algebraic equations and the symmetries recast using a spatiotemporal symmetry group. After the equations and its symmetries have been formulated, the numerical optimization techniques used to collect orbits are derived in chapter 3, followed by the new spatiotemporal 'clipping' method sect. 3.3 and spatiotemporal 'gluing' method sect. 3.5. Using the numerical optimization methods, a collection of orbits is created. With this collection, fundamental orbits are searched for, and found, using the combination of clipping and numerical optimization. Orbits and fundamental orbits can then be combined spatiotemporally to find progressively larger orbits via the spatiotemporal gluing method. Lastly, a demonstration of the `orbithunter` computational package and its generalization to other equations is presented in chapter 5.

In summary, the work presented here represents the beginning of a truly spatiotemporal theory of chaos and turbulence based upon spatiotemporal patterns. Starting from scratch in this manner has its share of disadvantages, as there is no large repertoire of work with which to compare to. In other words, there is no possibility of building upon the work of others, either computationally or theoretically. This constitutes a herculean endeavor, and as a result, only the first part of the spatiotemporal odyssey is completed here. That is, the work presented here focuses solely on new spatiotemporal numerical methods and techniques that will be required for a full description of spacetime via symbolic dynamics. These methods are packaged and presented as the open source code `orbithunter`.

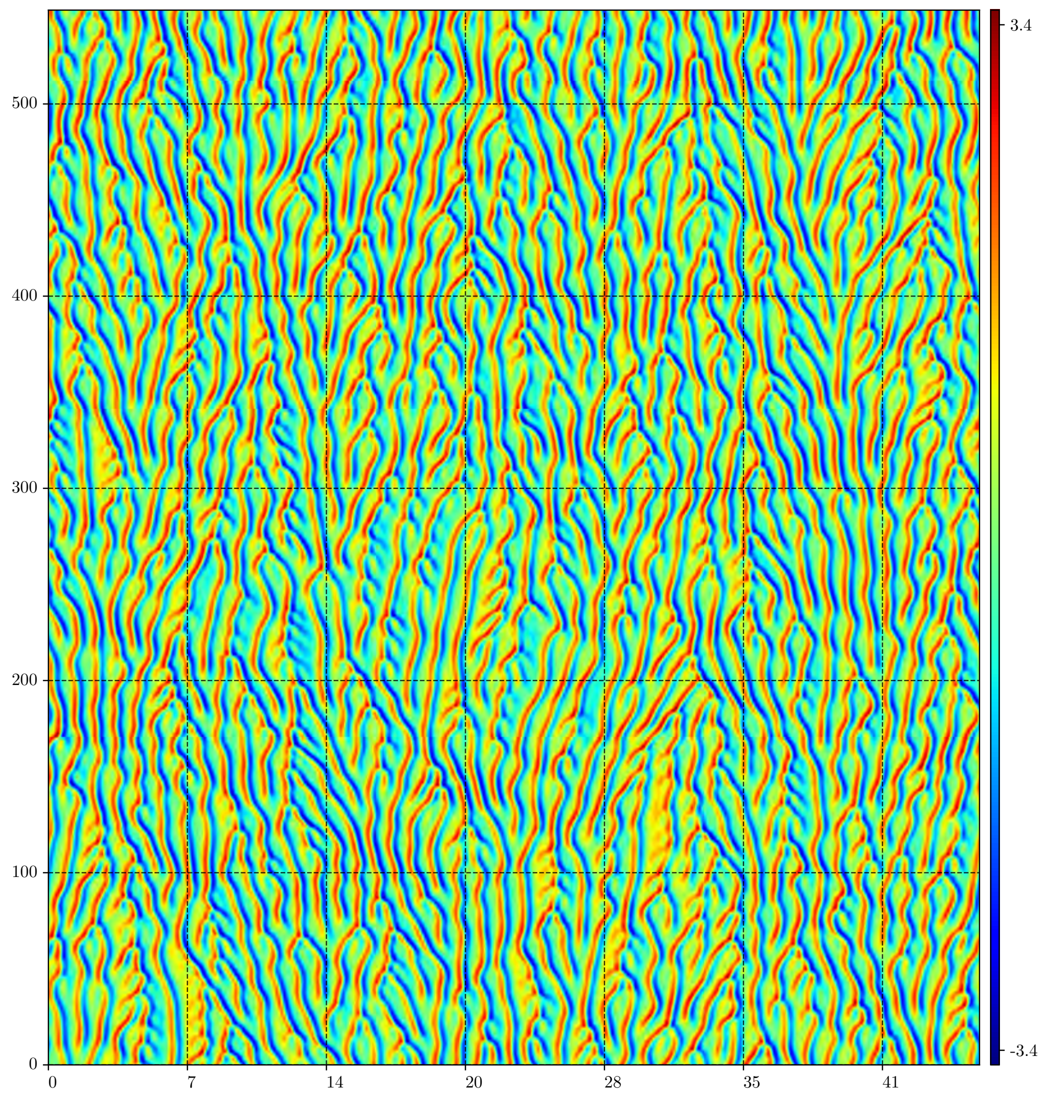

**Figure 1:** Simulation of "steady state turbulence" of the Kuramoto-Sivashinsky equation.

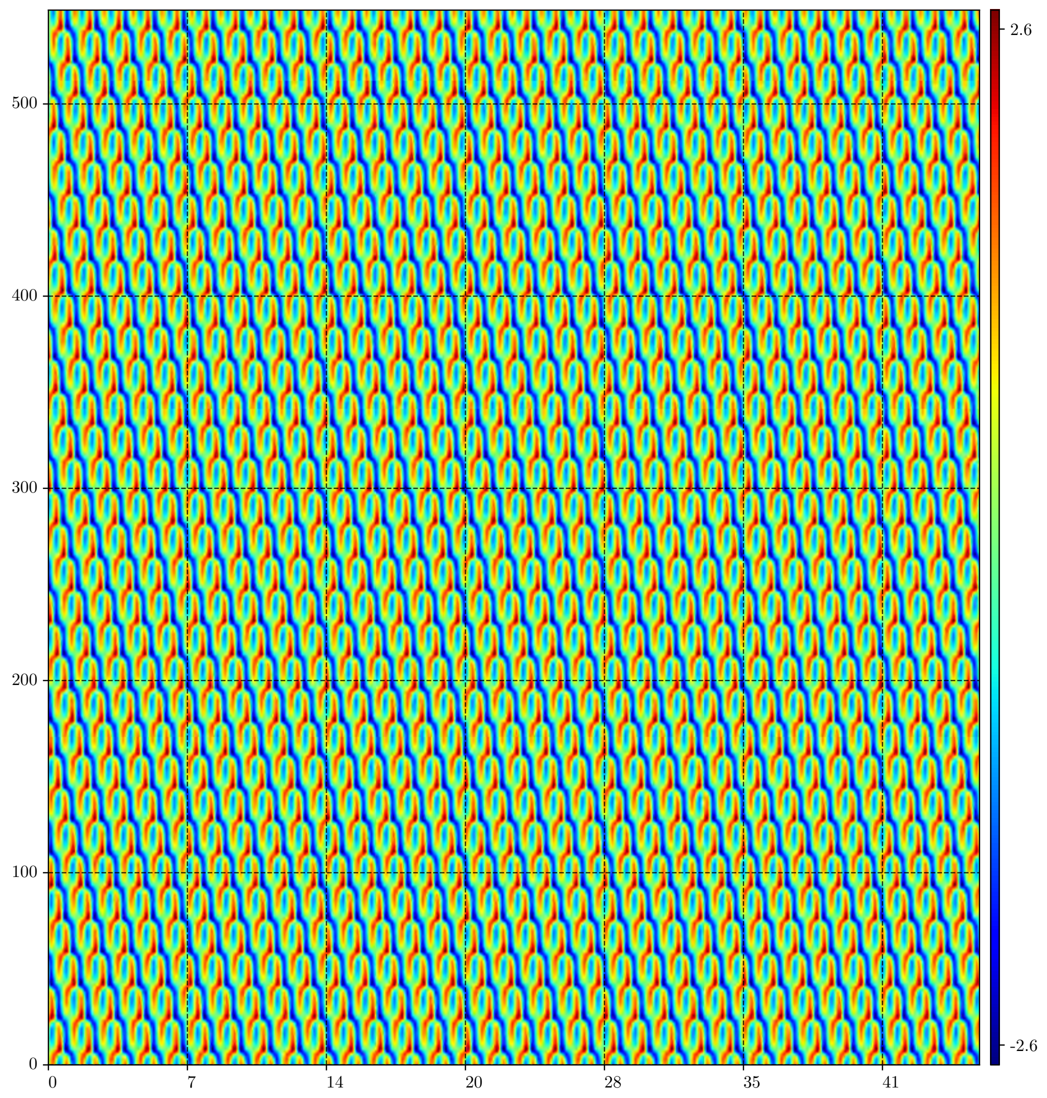

**Figure 2:** Spatiotemporal tiling of a relative periodic orbit.

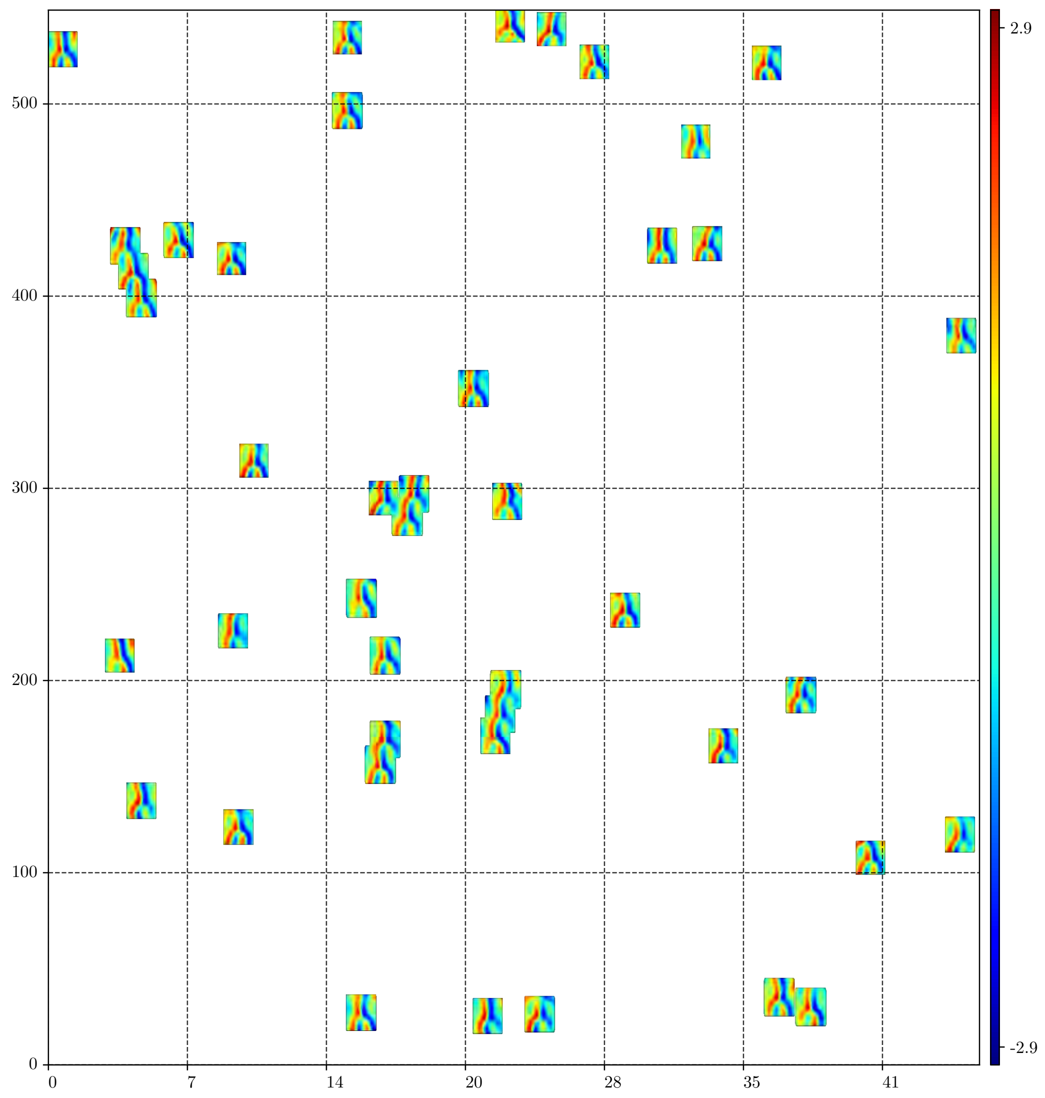

**Figure 3:** Qualitative demonstration of spatiotemporal shadowing using a single orbit.

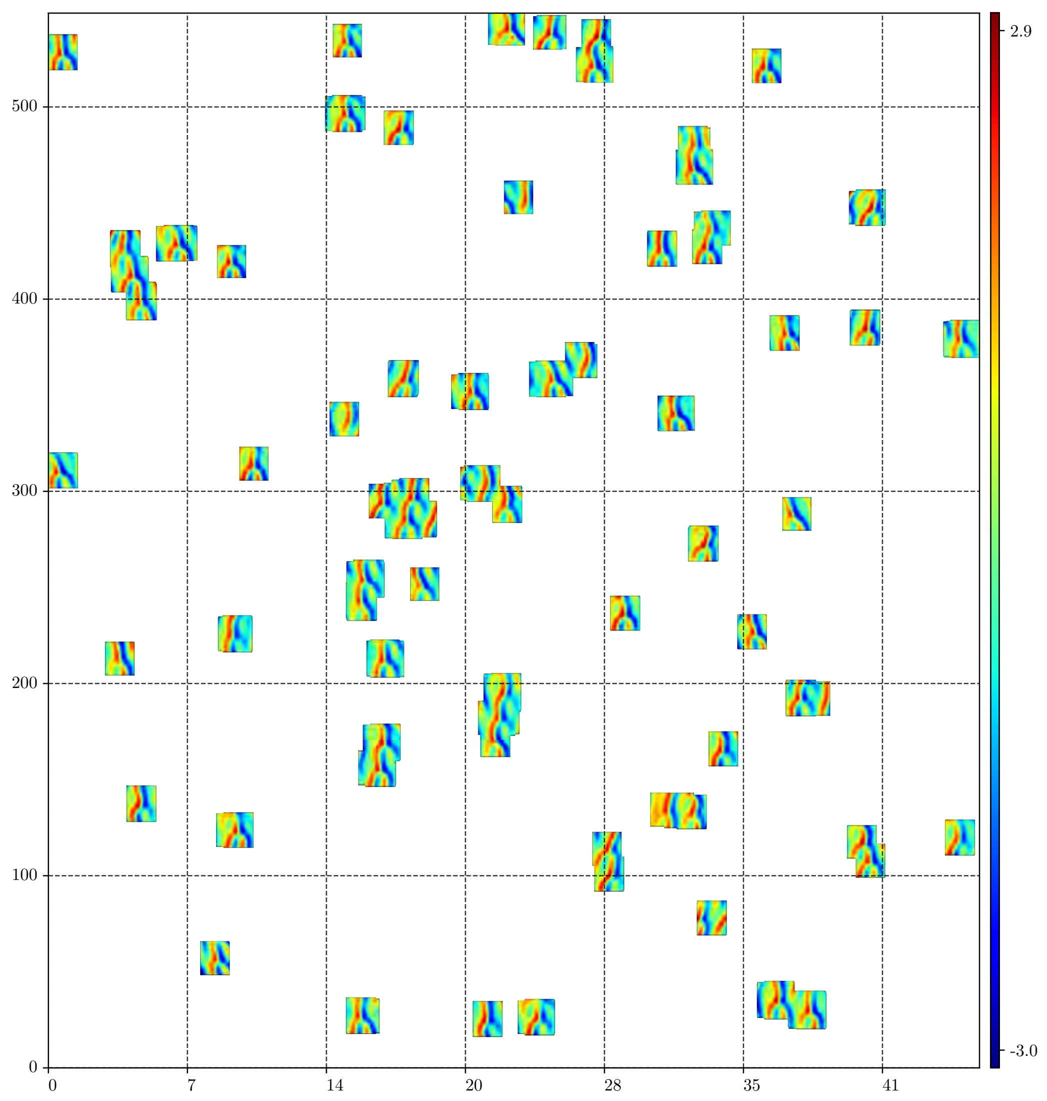

**Figure 4:** Qualitative demonstration of spatiotemporal shadowing using multiple, symmetry related orbits.

## *1.1 Kuramoto-Sivashinsky equation*

The Kuramoto-Sivashinsky equation and its variants have been used to model many different phenomena, including but not limited to: the dynamics of bright spots formed by self-focusing laser beams, oscillations of plasma particles trapped in magnetic wells created by the inhomogeneous magnetic field of a tokamak, Rayleigh-Bénard convection, flow of a viscous fluid down a vertical plane, nonlinear saturation of Rayleigh-Taylor instability in thin films for a few examples [64, 69, 79, 93, 107, 116, 118]. The Kuramoto-Sivashinsky equation also has connections with other partial differential equations such as the Navier-Stokes equation, the Kardar, Parisi, and Zhang (KPZ) equation and the stochastic Burgers equation [40, 66]. For our purposes, we consider the Kuramoto-Sivashinsky equation as a model for the velocity of a laminar flame front. The Kuramoto-Sivashinsky equation presents itself as an interesting case study into chaotic dynamical systems, as evidenced by investigations into the geometry of its state space [25] and the dimension of its inertial manifold [28]. The reason why the Kuramoto-Sivashinsky equation is used instead of the Navier-Stokes equation is for two reasons. First, its relative simplicity; second, the ease with which the two dimensional space-time velocity field can be visualized. This visualization makes our spatiotemporal arguments more compelling, easier to understand and also it allows us to quickly develop an intuition as to what patterns constitute fundamental orbits.

The general spatiotemporal Kuramoto-Sivashinsky equation on a doubly periodic spatiotemporal domain is given by

$$u_t + u_{xx} + \nu u_{xxxx} + \frac{1}{2}(u^2)_x = 0 \quad x \in [0, L]\,,\ t \in [0, T]\,. \tag{3}$$

The variable $u = u(t, x)$ represents a spatiotemporal velocity field. The subscripts $(\cdot)_x$ and $(\cdot)_t$ in (3) denote partial derivatives with respect to space and time. The roles of each term of this equation are: $u_{xx}$ is an "anti-diffusion" term, pumping energy into the system and feeding instabilities of large length scales, the 'hyper-viscous' term, $u_{xxxx}$, provides damping of small length scales, and lastly the nonlinear inertial term $1/2(u^2)_x$ transfers energy between the large and small scales. The "hyper-viscosity" parameter $\nu$ plays a role analogous to the role that the Reynolds number $Re$ plays in the Navier-Stokes equation;

instead of keeping this parameter we instead use the spatial dimension as a proxy, using the following transformations which provide a dimensionless form of (3) Specifically, $\nu$ is scaled out using with transformations $x \to x\nu^{1/2}, t \to t\nu, u \to u\nu^{-1/2}$ such that $x \in [0, L\nu^{-1/2}]$. The periods of periodic solutions are also rescaled following the relation: $T_p = \frac{T_p^*}{\nu}$. Possible avenues of study of the equation and its behavior include varying $L$ while keeping $\nu = 1$, or varying $\nu$ while keeping $L = 1$ or $2\pi$. The former of these two choices is utilized here. In these dimensionless units the form of the spatiotemporal Kuramoto-Sivashinsky equation utilized in this study is

$$u_t + u_{xx} + u_{xxxx} + \frac{1}{2}(u^2)_x = 0 \quad x \in [0, L]\,,\; t \in [0, T]\,. \tag{4}$$

When viewed as a dynamical system, an important physical scale is derived by linearizing about the trivial solution $u(t, x) = 0$ and noting that the eigenvectors of the resulting linearization are Fourier modes. This admits a spectrum defined by the polynomial $(\frac{2\pi k}{L})^2 - (\frac{2\pi k}{L})^4$ whose maximum on any given spatial domain occurs when $k = L/(2\pi\sqrt{2})$; in the discrete setting, the discrete value of $k$ closest to this will be referred to as the 'most unstable wavelength'. The spatial dimension of all figures are labeled in terms of this scale, which allows for quick interpretation and verification of figures. For comparison to other studies, $L = 22$ is the spatial domain size used in [11, 25, 28, 29, 45], which equals $L/(2\pi\sqrt{2}) \approx 2.5$ in wavelength units. It is noted that it is also common to restrict the investigation to symmetry invariant subspaces [1, 37, 64, 69, 78, 107]. We account for symmetries and exploit their benefits, but for the most part do not *require* such restrictions; making our results more generalizable.

## *1.2 Spatial integration*

The steady solutions of (4), i.e., where $u_t = 0$ have been investigated thoroughly [30, 51, 65, 88, 130]. By assuming $u_t = 0$ the Kuramoto-Sivashinsky (4) can be integrated with respect to space; which can be written as a three dimensional dynamical system; $u_x = v, v_x = w, w_x = u^2 - v - c$. One of the motivating factors behind our spatiotemporal study was similar in concept; except we did not assume $u_t = 0$. In other words, we investigated the dynamical system with periodic boundary conditions in time $u(t) = u(t+T)$, where $x$ plays the role of the dynamical 'time'. Using known periodic orbits we investigated whether it was possible to define a spatial dynamical system, such that any invariant 2-torus could be reproduced using only a single temporal strip and spatial integration. Similar to the three dimensional spatial system just mentioned, define

$$u^{(0)} \equiv u\,, \quad u^{(1)} \equiv u_x\,, \quad u^{(2)} \equiv u_{xx}\,, \quad u^{(3)} \equiv u_{xxx}\,. \tag{5}$$

The Kuramoto-Sivashinsky equation can then be written as a system of ordinary differential equations

$$\begin{aligned} u_x^{(0)} &= u^{(1)}\,, \\ u_x^{(1)} &= u^{(2)}\,, \\ u_x^{(2)} &= u^{(3)}\,, \\ u_x^{(3)} &= -u_t^{(0)} - u^{(2)} - u^{(0)}u^{(1)}\,. \end{aligned} \tag{6}$$

Given the time-periodic boundary condition, it is natural to expand the Kuramoto-Sivashinsky field $u(x,t_n) = u_n(x)$ as a temporal Fourier $u(x,t_n) = u_n(x)$ over $M$ points of a periodic temporal lattice $t_n = nT/M$, $n = 0, 1, \cdots, M-1$:

$$u^{(i)}(t,x) = \sum_{n=0}^{M-1} \tilde{u}^{(i)}(x)\, e^{i\omega_n t_n}\,, \quad \text{where } \omega_n = 2\pi n/T. \tag{7}$$

Now we write the equivalent expression in its Fourier representation, taking into consideration a truncated number of Fourier modes $N$.

$$\begin{aligned} \frac{\partial}{\partial x}\tilde{u}_k^{(0)} &= \tilde{u}_k^{(1)}\,, \\ \frac{\partial}{\partial x}\tilde{u}_k^{(1)} &= \tilde{u}_k^{(2)}\,, \\ \frac{\partial}{\partial x}\tilde{u}_k^{(2)} &= \tilde{u}_k^{(3)}\,, \\ \frac{\partial}{\partial x}\tilde{u}_k^{(3)} &= -i\omega_k\tilde{u}_k^{(0)} - \tilde{u}_k^{(2)} - \sum_{m=0}^{N/2-1} \tilde{u}_m^{(0)}\tilde{u}_{k-m}^{(1)}\,. \end{aligned} \tag{8}$$

Rewriting (5) in terms of temporal Fourier modes, we obtain a tower of ordinary differential equations, As a disclaimer, the equations and formulas written here will not be used in the remainder of the text due to different numerical choices between this preliminary investigation and the full spatiotemporal formulation.

The nonlinear term manifests as a convolutional sum in (8) however it is more practically evaluated using transforms; this constitutes a *pseudospectral* method [6, 15, 16, 100, 113, 120].

$$\sum_{m=0}^{N/2-1} \tilde{u}_m^{(0)} \tilde{u}_{k-m}^{(1)} = \mathcal{F}\left\{u^{(0)} \cdot u^{(1)}\right\} \tag{9}$$

The actual integration of these equations utilized MATLAB's ODE integrator for stiff equations `ode15s`. The spatial integration results, demonstrated in figure 5, indicated that our spatiotemporal formulation based on invariant 2-tori had merit. Spatial integration was performed in two segments utilizing reversibility of the spatial system which results from reflection symmetry. The error plot, figure 5, was half-cell shifted such that the boundaries are displayed at $x = L/2$ for comparison purposes. Unfortunately, it was found that the

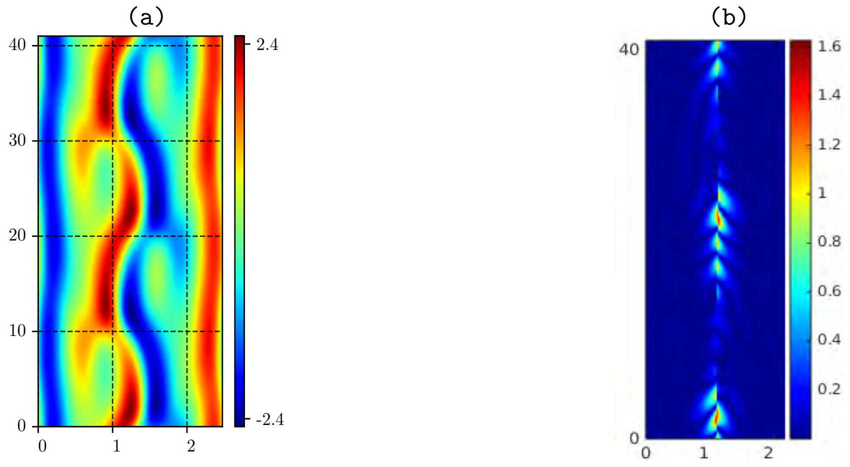

**Figure 5:** (a) Spatially integrated orbit, (b) difference between spatial and temporal integration.

trajectories diverge to infinity after very short 'times'; from the literature on 'chronotopic Lyapunov analysis' [44, 80, 81, 105], strange attractors in time 'generally' correspond to a strange repellers in space. Therefore it is believed that the instabilities and diverging results are not computational bugs, rather, features of the spatial system. As previously mentioned, due to spatial reflection symmetry the system of equations (8) is a reversible system with respect to space. Based upon this property, a possible remedy for these poor results might be the implementation of a symplectic or variational integrator; numerical integration schemes used in Hamiltonian systems wherein energy conservation is necessary [43, 72, 85]. However, (8) does not have 'Hamiltonian' structure; or at least we did not find any way to show that it does; therefore, we used these results mainly as motivation for the spatiotemporal formulation. Therefore, we moved past symplectic integrators and the spatial system (8) in the process, having derived more motivation for a spatiotemporal method. At the core of the spatiotemporal formulation is the ability to find invariant 2-tori of varying size. This is accomplished by deriving a system of differential algebraic equations, defined in terms of spatiotemporal Fourier modes. By variational techniques, we are able to solve the corresponding boundary value problem, which is posed as a least-squares minimization problem. The next chapter focuses on the tools to produce these equations such as Fourier transforms, symmetries, and lastly the differential algebraic equations themselves.

# CHAPTER II

# SPATIOTEMPORAL FORMULATION

## *2.1 Variational methods*

Historically there have been a large number of studies of the Kuramoto-Sivashinsky equation in the context of spatially periodic boundary conditions using Galerkin projections, transforming (4) into a system of time dependent ordinary differential equations, one for each spatial mode mode [1, 9, 64, 69, 96, 107]. Now, motivated by commuting space and time translational invariance, a natural choice is to utilize spatiotemporal Fourier modes; producing a set of differential algebraic equations. This is implemented using a pseudospectral formulation, the defining property of which uses Fourier transforms and a grid of collocation points to evaluate the nonlinear term [6, 15, 41, 100, 128]. Spectral methods are memory-minimizing and accurate [6]; the main disadvantages being the programming difficulty and implementation on irregular domains. Luckily for us, after some initial bumps and bruises, the power of these methods truly begins to shine. The spatiotemporal formulation is predicated upon the successful collection of invariant 2-tori; doubly periodic solutions to (4). For brevity, the term 'orbit' shall refer to the various names for solutions of the spatiotemporal formulation used so far: invariant 2-tori, doubly periodic orbits, periodic orbits, etc.

To begin, the spatiotemporal configuration manifold or *tile* on which an orbit is defined will be denoted by $\Omega \equiv (T, L)$ such that

$$\Omega = [0, T] \times [0, L] \subseteq \mathbb{R}^2 \,. \tag{10}$$

Due to translational invariance these intervals can always begin at 0 without loss of generality. Proofs regarding the existence and boundedness of spacetime solutions of the Kuramoto-Sivashinsky equation, are not derived here, instead we assume the results from [64, 96, 123] hold in order to satisfy these requirements. Continuing, the 'governing equation', the Kuramoto-Sivashinsky equation (4) in this case, will be represented by the function $f(v)$ where $v$ is referred to as the 'state vector'. The state vector contains all variables required to define an orbit

$$v \equiv [u, p_i]^\top . \tag{11}$$

where $u = u(t,x)$ represents a spatiotemporal velocity field and represents all relevant parameters $p_i$ (includes $T, L$). A finite dimensional state vector only formally exists after discretization, as solutions $u(t,x)$ are inherently infinitely dimensional; However, even though it is a slight abuse of notation, we favor simply writing $v$ in place of explicitly writing $(u, p_i)$. The orbits of $f$ are then defined as the subset of functions $u \in L^2$ which satisfy the doubly periodic boundary conditions and exist in the appropriate Sobolev space; $u \in \mathrm{H}^{1,4}_{f}(\Omega) \subset \mathrm{H}^{1,4}(\Omega)$ where $f(v) = 0$ is satisfied identically on the tile $\Omega$

$$\mathrm{H}^{1,4}_{f}(\Omega) \equiv \{u \in \mathrm{H}^{1,4}_{\text{per}}(\Omega) \mid f(u) = 0 \quad \forall (t,x) \in \Omega\} . \tag{12}$$

The superscripts $(1,4)$ indicate functions once differentiable in time and four times differentiable in space. A state $v$ whose velocity field exists in $\mathrm{H}^{1,4}_{\Omega}(\Omega)$. but does not satisfy $f = 0$ (prior to optimization, at least) will be referred to as an orbit approximation, guess orbit, orbit guess, initial condition, etc. It follows that the larger the values of $f$ (in magnitude) the further the guess is from an orbit. Additionally, implicit in the definition (12) is the dependence $\Omega = \Omega(T, L)$; in other words the spatiotemporal domains on which orbits exist are now functions of the spatiotemporal dimensions due to their inclusion as hyperparameters.

To find orbits, a variational formulation is used, one which poses the problem of solving $f(v)$ as a minimization problem. The derivation of the scalar cost functional or *cost function* [7] which defines the minimization problem begins with the Lagrangian density or *formal Lagrangian* [56]

$$\mathcal{L}(t, x, p_i, u, \lambda, u_t, u_x, u_{xx}, u_{xxxx}) = \frac{1}{2}\lambda[u_t + u_{xx} + u_{xxxx} + \frac{1}{2}(u^2)_x] , \tag{13}$$

which is a function of $u(t,x)$, its partial derivatives, the parameters $p_i$ and finally the co-state or adjoint variable $\lambda(t,x)$. In numerical contexts, $\lambda$ is often referred to as a 'test' or 'trial' function [6], it is assumed to exist in the appropriate Sobolev space, and satisfy the same boundary conditions as $u$; it is also essentially a Lagrange multiplier. The factor of

1/2 is simply for convenience later on. Defining the integral

$$S(\lambda, v) = \int_\Omega \mathcal{L}(t, x, p_i, u, \lambda, u_t, u_x, u_{xx}, u_{xxxx})\,. \tag{14}$$

Requiring $S = 0\,,\ \ \forall \lambda$ yields the *weak form*, *integral form* or *variational form* of the Kuramoto-Sivashinsky equation. This formulation is closely related to true spectral methods; for example, selecting the space of trigonometric polynomials as the space for $\lambda$ defines the Galerkin method [16]. The benefit of the weak formulation typically results from the application of integration-by-parts to (14); by doing so the differentiability requirements on $u$ can be reduced to only second order. The weak formulation is in fact not used to define the optimization problem here; it is mentioned, however, because of previous investigations into continuous symmetries and derivation of possible conservation laws associated with Lie-Bäcklund symmetries [57, 59, 62], relegated to appendix B. Unfortunately, such analysis proved fruitless but we admit this might be due to user error; the results are included for posterity.

Returning back to the variational formulation; the problem of finding solutions $f = 0$ is posed as finding the stationary points of the functional (14). To show that these stationary points are indeed the orbits we are looking for, let us derive the Euler-Lagrange equations of (14). These can be derived by the 'functional derivatives' of (13) with respect to $u$, $\lambda$. The general form of the functional derivative is defined by the operator [62]

$$\frac{d}{dw_i} = \frac{\partial}{\partial q_\ell} + \sum_{j=1}^{n} \sum_{\mu_1 \leq \cdots \leq \mu_j} (-1)^j \frac{\partial^j}{\partial q_{\mu_1} \ldots \partial q_{\mu_j}} \left( \frac{\partial}{\partial w_{i,\mu_1 \ldots \mu_j}} \right)\,, \tag{15}$$

where $d$ is used to indicates the total derivative and $\partial$ indicates partial derivatives. The variables $w_i$ representing $u$ and $\lambda$ and the configuration space variables $t, x$ represented by $q_\ell$. The bounds on the inner sum (15) simply take care of repeating terms arising from the commutation of derivatives. Application of (15) to (13) simply yields the generalization of the Euler-Lagrange equations to more variables and higher order derivatives; yielding the Kuramoto-Sivashinsky equation, its *adjoint equation* [62] as well as conditions on the partial derivatives with respect to parameters $p_i$.

$$\begin{aligned}
\frac{d\mathcal{L}}{du} &= \left(\frac{\partial}{\partial u} - \frac{\partial}{\partial x}\frac{\partial}{\partial u_x} - \frac{\partial}{\partial t}\frac{\partial}{\partial u_t} + \frac{\partial^2}{\partial x^2}\frac{\partial}{\partial u_{xx}} + \frac{\partial^4}{\partial x^4}\frac{\partial}{\partial u_{xxxx}}\right)\mathcal{L} \ , \\
&= -\lambda_t + \lambda_{xx} + \lambda_{xxxx} - u\lambda_x \ , \\
\frac{d\mathcal{L}}{d\lambda} &= u_t + u_{xx} + u_{xxxx} + \frac{1}{2}(u^2)_x \ , \\
\frac{d\mathcal{L}}{dp_i} &= \lambda\frac{\partial f}{\partial p_i} \ . \qquad (16)
\end{aligned}$$

After the derivation of the Euler-Lagrange equations (16) we require $\lambda = f$ such that the Lagrangian density becomes

$$\mathcal{L}(t, x, p_i, u, u_t, u_x, u_{xx}, u_{xxxx}) = \frac{1}{2}|f|^2 \ , \qquad (17)$$

and the action

$$S(f) = \int_\Omega \frac{1}{2}|f|^2 dt dx \ . \qquad (18)$$

Such that this specific choice for $\lambda$ admits a least-squares optimization problem wherein the the integral (18) only equals zero when $f = 0$ identically on the tile, the definition of strong solutions to (4). The Euler-Lagrange equations also change accordingly; as can be seen by substitution, all of the equations (16) are satisfied when $\lambda = f = 0$, the adjoint equation trivially so. The problem of finding orbits is reframed as the least-squares minimization problem: find $v$ such that $\phi(v) = 0$, where $\phi$ is the discretized form of the cost function, written in vector norm as

$$\phi(v) = \frac{1}{2}f^\top f = 0 \, . \qquad (19)$$

For everything that follows we shall assume the usage of an $L_2$ norm. It should be noted that the equation $\phi(v) = 0$ is actually not equivalent to $f(v) = 0$ numerically. Assuming machine precision is $\epsilon_c = 10^{-15}$, then there is no way of distinguishing a value $f^2 \leq 10^{-15}$, which makes all values of $f \leq 10^{-15/2}$, for all intents and purposes, equivalent. In this regard, we are losing precision due to the quadratic formulation of cost functional. With the specific form of $\phi$ given by (19), orbits will be defined as its minima $\phi = 0$, as $f = 0$ implies $\phi = 0$. Hereafter, it is specifically the discrete function (19) which will be referred to as the cost function; the value of the cost function corresponding to a particular state $v$ will be referred to as the *residual* of $v$. Before we move on, a disclaimer in regards to the usage of the terminology 'variational form' and 'variational formulation'. The authors of [16] specify that this name only applies to 'time independent problems with a spatially symmetric operator'. In [6], the authors do not mention any such requirement, however they do imply that the variational formulation represents the equations that result from applying integration by parts to (14). Due to the lack of dynamics and the fact that our numerical methods are defined by the variation of the cost function, making small corrections to reduce the residual, we still refer to our formulation as variational.

## 2.2 Spatiotemporal Fourier Transforms

With the optimization problem properly defined we now need a discrete representation of the governing equations and our orbit states $v$. Following the guide of J. P. Boyd [6], we let the geometry determine our set of basis functions. Due to translational invariance, periodic boundary conditions, and smoothness, a spatiotemporal Fourier basis is a natural choice for the discrete representation of $u(t, x)$.

The derivations that follow will be in terms of the (truncated) spatiotemporal discrete Fourier series; taking advantage of the results regarding the convergence of truncated Fourier series [16, 17] and other discretization related properties. This discrete formulation directly connects to the computational codes and hence is more representative of the work performed here than say, exposition regarding the tensor products of triginometric polynomials and the corresponding function space. The tile $\Omega$ is discretized by a set of spatiotemporal grid points, or *collocation points* [6, 15, 16], defined such that $\Omega_{nm} = (t_n, x_m)$

$$\begin{aligned} t_n &= \frac{nT}{N}, \quad n \in 0, 1, \ldots, N-1\ , \\ x_m &= \frac{mL}{M}, \quad m \in 0, 1, \ldots, M-1\ . \end{aligned} \tag{20}$$

For brevity, discretized tiles will also be referred to simply as tiles (10); in context it should be fairly obvious if we mean the continuous or discrete version. It directly follows that the discretized velocity field $u(t_n, x_m)$ is exactly the values of $u(t, x)$ interpolated at the collocation points; the interpolation determined by the values of the spatiotemporal Fourier coefficients, soon to be derived. By electing to compute the nonlinear term (4) as a product in physical space at the collocation points, as opposed to a convolution in spectral space, we can classify our formulation as *pseudospectral.* An orbit in this framework is then given by $u(t_n, x_m)$ such that $f(u(t_n, x_m)) = 0$ at every collocation point. The benefits of pseudospectral methods are numerous; most importantly they increase the accuracy of our calculations while also minimizing the amount of computational memory required [6].

Instead of using an overly verbose and repetitive verbiage for the coefficients of the Fourier basis function, i.e. *spatiotemporal Fourier modes* or *spatiotemporal Fourier coefficients*; they will simply be referred to as *modes*, unless specifically mentioned otherwise. This avoids excessive usage of 'spatiotemporal' and 'Fourier'; everything from here on out is in one way or another, spatiotemporal. Likewise, *discrete spatiotemporal real valued Fourier transform* (it can get even worse) will be cut short to either *Fourier transform* or simply 'the' transform of the field $u(t_n, x_m)$.

Prior to the development of the real-valued transforms, we note that the transforms are technically computed as

$$\begin{aligned} \tilde{u}_k(t_n) \quad &= \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} u(t_n, x_m) e^{-i(2\pi k m/M)} \ , \\ u(t_n, x_m) \quad &= \frac{1}{\sqrt{M}} \sum_{k=1}^{M/2-1} \tilde{u}_k(t_n) e^{i(2\pi k m/M)} \ , \end{aligned} \tag{21}$$

we are simply separating real and imaginary parts to retrieve a real-valued form. These transforms are computed via the numerical Python package SciPy [134], as they allow for simple importation of optimized Fourier transform algorithms. While the transforms are technically computed via (21), the Fourier transforms will be written in terms of a truncated basis of cosines and sines, as it is beneficial to work with the SO(2) real-valued representation of the modes. The primary reason being: if the computational degree of freedom are complex numbers, then constraints are needed to ensure the dimensions $(T, L)$ are maintained as real numbers during the optimization process. The second reason is that expanding in sines and cosines turns out to be very helpful for orbits with discrete spatiotemporal symmetries, due to the parity of said functions.

Moving on, the expressions of the Fourier modes and transforms are written such that evaluating the expressions on paper will yield *exactly* the same result (up to machine precision, of course) as the computational codes. Before continuing to the discrete transforms themselves, first define the spatiotemporal frequencies $q_k$ and $\omega_j$, defined by the discretization of the tile

$$\begin{aligned} q_k &= \frac{-2\pi k}{L} \quad \text{where, } k \in 1, \ldots \frac{M}{2} - 1 \,, \\ \omega_j &= \frac{-2\pi j}{L} \quad \text{where, } j \in 0, \ldots \frac{N}{2} - 1 \,. \end{aligned} \tag{22}$$

For real valued input the negative frequency and positive frequency modes of the Fourier transform are related by conjugation relations; hence only half of the spectrum ($j, k$ non-negative) are required to fully define $u(t_n, x_m)$. Also, it is noted that products of the form $q_k x_m = 2\pi km/M$, the form that routinely appears in the discussions of spectral and pseudospectral methods.

The spatiotemporal transform of $u$ into the set of modes $\tilde{u}$ is derived via the composition of one-dimensional spatial and temporal transforms. Again, it should be stressed that in the derivations that follow such that *everything is written in order to agree with output of the computational codes*; uneasiness towards any strange conventions is likely mutual. The numerically implemented spatial transform (and its inverse) for real valued input take the following form

$$\begin{aligned} e_k(t_n) &= \sqrt{\frac{2}{M}} \sum_{m=0}^{M-1} u(t_n, x_m) \cos(q_k x_m) \,, \\ f_k(t_n) &= \sqrt{\frac{2}{M}} \sum_{m=0}^{M-1} u(t_n, x_m) \sin(q_k x_m) \,, \\ u(t_n, x_m) &= \sqrt{\frac{2}{M}} \sum_{k=1}^{\frac{M}{2}-1} e_k(t_n) \cos(q_k x_m) + f_k(t_n) \sin(q_k x_m) \,. \end{aligned} \tag{23}$$

The $k = 0, \frac{M}{2}$ spatial modes are constrained to 0; hence why they are not included in (23). This choice is related to Galilean invariance and hence it is reserved for the section on spatiotemporal symmetries, sect. 2.3. The constraint on the Nyquist ($M/2$) mode is a numerical convention; numerically, this mode is returned as the sum of the corresponding negative and positive modes. This, in combination with the conjugation relation, means that this mode is purely real; this behavior, and the fact that the magnitude is typically small for sufficiently large discretizations, means that we exclude this mode from calculations (i.e. discarded after both space and time transforms).

$$
\begin{aligned}
a_{jk} &= \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} e_k(T - t_n) \cos(\omega_j t_n) \; , \\
b_{jk} &= \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} e_k(T - t_n) \sin(\omega_j t_n) \; , \\
c_{jk} &= \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} f_k(T - t_n) \cos(\omega_j t_n) \; , \\
d_{jk} &= \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} f_k(T - t_n) \sin(\omega_j t_n) \; , \\
e_k(T - t_n) &= \sqrt{\frac{2}{N}} \Big( \frac{a_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} a_{jk} \cos(\omega_j t_n) + b_{jk} \sin(\omega_j t_n) \Big) \; , \\
f_k(T - t_n) &= \sqrt{\frac{2}{N}} \Big( \frac{c_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} c_{jk} \cos(\omega_j t_n) + d_{jk} \sin(\omega_j t_n) \Big) \; . \qquad (24)
\end{aligned}
$$

The spatiotemporal transform follows by taking the temporal transform of the spatial modes $e_k(t_n), f_k(t_n)$ (23). Due to the numerical implementation, time is technically parameterized as $T - t_n$, as the first row of the array in which $u$ is stored corresponds to $t = T$ and the last row, $t = 0$. The expression (23) is still correct, but now that the time ordering matters, the time transform and its inverse must be written

$$
\begin{aligned}
a_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(T-t_n, x_m) \cos(q_k x_m) \cos(\omega_j t_n) \ , \\
b_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(T-t_n, x_m) \cos(q_k x_m) \sin(\omega_j t_n) \ , \\
c_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(T-t_n, x_m) \sin(q_k x_m) \cos(\omega_j t_n) \ , \\
d_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(T-t_n, x_m) \sin(q_k x_m) \sin(\omega_j t_n) \ , \\
u(T-t_n, x_m) &= \sqrt{\frac{4}{NM}} \sum_{k=1}^{\frac{M}{2}-1} \Bigg[ \Bigg( \frac{a_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} a_{jk} \cos(\omega_j t_n) + b_{jk} \sin(\omega_j t_n) \Bigg) \cos(q_k x_m) \\
&\quad + \Bigg( \frac{c_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} c_{jk} \cos(\omega_j t_n) + d_{jk} \sin(\omega_j t_n) \Bigg) \sin(q_k x_m) \Bigg] .
\end{aligned}
\tag{25}
$$

This is a technical detail very important when taking time derivatives; if overlooked then the time derivative would be off by a negative sign. The transform (24) can be rewritten, yielding

$$
\begin{aligned}
e_k(t_n) &= \sqrt{\frac{2}{N}} \Bigg( \frac{a_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} a_{jk} \cos(-\omega_j t_n) + b_{jk} \sin(-\omega_j t_n) \Bigg) \ , \\
f_k(t_n) &= \sqrt{\frac{2}{N}} \Bigg( \frac{c_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} c_{jk} \cos(-\omega_j t_n) + d_{jk} \sin(-\omega_j t_n) \Bigg) \ .
\end{aligned}
\tag{26}
$$

The convention is to use (24); this should only be viewed as a numerical parameterization of the field $u(t,x)$. This is not trying to imply anything regarding time reversal. As previously mentioned, this is accommodated by including an extra negative sign in time differentation, resulting from the negative sign accompanying $\omega_j$, i.e. $-\omega_j$ in (26). Finally, substitution of

(26) into (23) yields the spatiotemporal transform

$$
\begin{aligned}
a_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(t_n, x_m) \cos(q_k x_m) \cos(\tilde{\omega_j} t_n) \ , \\
b_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(t_n, x_m) \cos(q_k x_m) \sin(\tilde{\omega_j} t_n) \ , \\
c_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(t_n, x_m) \sin(q_k x_m) \cos(\tilde{\omega_j} t_n) \ , \\
d_{jk} &= \sqrt{\frac{4}{NM}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} u(t_n, x_m) \sin(q_k x_m) \sin(\tilde{\omega_j} t_n) \ , \\
u(t_n, x_m) &= \sqrt{\frac{4}{NM}} \sum_{k=1}^{\frac{M}{2}-1} \Bigg[ \bigg( \frac{a_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} a_{jk} \cos(\tilde{\omega_j} t_n) + b_{jk} \sin(\tilde{\omega_j} t_n) \bigg) \cos(q_k x_m) \\
&\qquad + \bigg( \frac{c_{0k}}{2} + \sum_{j=1}^{\frac{N}{2}-1} c_{jk} \cos(\tilde{\omega_j} t_n) + d_{jk} \sin(\tilde{\omega_j} t_n) \bigg) \sin(q_k x_m) \Bigg] \\
&\qquad \text{where, } \tilde{\omega_j} = -\omega_j \ . \qquad (27)
\end{aligned}
$$

Finally, we have arrived at a formula which defines $u(t_n, x_m)$ in terms of spatiotemporal Fourier modes. The expression (27) is very cumbersome, however, and so it will typically be represented using the operators $\mathcal{F}, \mathcal{F}^{-1}$.

For the numerical derivations that follow, two different representations are used. The first of the two representations defined is a tensor representation. The tensor arranges the four sets of modes of (27) into blocks based on frequencies

$$
\tilde{u} \equiv \begin{bmatrix} a_{0k} & c_{0k} \\ a_{jk} & c_{jk} \\ b_{jk} & d_{jk} \end{bmatrix} . \qquad (28)
$$

The zeroth $j = 0$ time modes are indicated separately because there are no analogous terms in $b_{jk}, d_{jk}$. This is an important factor to note for time differentiation, however, for simplicity, the awkwardness of the zeroth mode will be implicit in the mode tensor expression such that (28) will be simplified to

$$
\tilde{u} \equiv \begin{bmatrix} a_{jk} & c_{jk} \\ b_{jk} & d_{jk} \end{bmatrix} . \qquad (29)
$$

In the context of the tensor notation the sets of modes $\{a_{jk}, b_{jk}, c_{jk}, d_{jk}\}$ will be referred to as *mode blocks*. The shape of the mode tensor is fundamentally different when symmetries are considered; therefore, a notation which is uniform regardless of symmetry is implemented. Defining the set of indices as

$$\begin{aligned} k &\in \{1, ..., M/2-1\}\,, \\ j &\in \{1, ..., N/2-1\}\,. \end{aligned} \tag{30}$$

Each of the mode blocks (29) is labeled by indices $j, k$. The set of all indices in (29) is captured by the ordered pairs, defined via products of sets as

$$\{\mathbf{j}, \mathbf{k}\} \equiv \{\{\{0\}, \{j\}, \{j\}\} \times \{\{k\}, \{k\}\}\}\,, \tag{31}$$

where $\{j\}$ and $\{k\}$ are given by (30). This notation is simply to help keep track of the dimensions of the mode tensor (29). For example the dimensions of the mode tensor (for orbits without discrete symmetry) are

$$\begin{aligned} |\{\mathbf{j}, \mathbf{k}\}| &= |\{\{\{0\}, \{j\}, \{j\}\}| \times |\{\{k\}, \{k\}\}\}|\,, \\ &= (|\{0\}| + |\{j\}| + |\{j\}|) \times (|\{k\}| + |\{k\}|)\,, \\ &= (1 + (N/2-1) + (N/2-1)) \times (M/2 - 1 + M/2 - 1)\,, \\ &= (N-1) \times (M-2)\,. \end{aligned} \tag{32}$$

This notation is not perfect, but it allows us to represent the ordering of modes within each block, as well as the dimensions of the mode tensor. It is very important to get this ordering correct for differentiation and other operators later on.

In anticipation of future derivations, one final convention involving the dimensionality of the mode tensor and its indices (31) is introduced. Specifically, for the derivation of linear operators will use Kronecker products with identity matrices; the dimensions of which are given (32) or (30). Again, the following definitions allow for a catch-all notation without having to constantly spell out individual cases for orbits with different symmetries; for a tensor equivalent with shape (32) we define

$$\begin{aligned}
\mathbb{I}_j &\equiv \mathbb{I}_{N/2-1} \, , \\
\mathbb{I}_k &\equiv \mathbb{I}_{M/2-1} \, , \\
\mathbb{I}_{|\mathbf{k}|} &\equiv \mathbb{I}_{M-2} \, , \\
\mathbb{I}_{|\mathbf{j}|} &\equiv \mathbb{I}_{N-1} \, .
\end{aligned} \tag{33}$$

The last two identity matrices diagonals' have dimension equal to the total number of space and time indices, respectively. The sizes of the first two identities are the same across all classes of orbits, but the latter two are not; the total number of indices dependent on symmetry.

The second, vector representation requires two decisions to be made; how to order real and imaginary components with respect to one another and how to order space and time indices relative to each other. The convention here is to have the spatial index $k$ as the 'inner' index and $j$ as the outer index. For those familiar with programming parlance, this would be equivalent to two nested 'for-loops'. Alternatively, this can be viewed as taking the mode tensor (29) and 'stacking' the rows in a vertical manner thereby 'flattening' the tensor into a vector. To represent this visually, the vector can informally be written as

$$\Big[ \overbrace{\underbrace{a_{(0,k)}}_{\cos(q_k x_m)} \quad \underbrace{c_{(0,k)}}_{\sin(q_k x_m)} \quad \cdots}^{\cos(\omega_j t_n)} \quad \overbrace{\underbrace{b_{(1,k)}}_{\cos(q_k x_m)} \quad \underbrace{d_{(1,k)}}_{\sin(q_k x_m)} \quad \cdots}^{\sin(\omega_j t_n)} \Big]^{\top} . \tag{34}$$

The labelling by trigonometric functions is mainly a means of demonstrating the pattern with which indices are cycled through; no mathematical statements are being made in (34) other than the ordering of the modes.

As one might suspect, the vector representation utilizes matrix representations of the various operators relevant to the differential algebraic equations. For example, one of the numerical methods described in chapter 3 directly solves the least-squares Newton equations; requiring explicit construction of the Jacobian (93). As the final task of this section, the matrix representations of the Fourier transforms, $\mathcal{F}$, $\mathcal{F}^{-1}$ are derived. These matrices are derived by first constructing the matrix for a single instant in time or space, and then using a Kronecker product to extend it to spacetime, i.e. make a block diagonal matrix whose blocks are simply copies of one another. The matrix representation of the Fourier transforms can be constructed numerically by applying the Fourier transform to each column of an appropriately sized identity matrix; this yields a Vandemonde matrix wherein the elements are powers of the roots of unity. The matrix representation of the linear operator $\mathcal{F}_x$ over the complex numbers is

$$\mathcal{F}_x = \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & r & r^2 & r^3 & \dots & r^{(M-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & r^{(\frac{M}{2})} & r^{2(\frac{M}{2})} & r^{3(\frac{M}{2})} & \dots & r^{(\frac{M}{2})(M-1)} \end{bmatrix}, \tag{35}$$

with $r = \cos(\frac{2\pi}{M}) + i\sin(\frac{2\pi}{M})$. The matrix has dimensions $M/2 + 1 \times M$; the dimension resulting from the conjugacy relations for real-valued input. The remaining steps are: to account for the constraints on $k \in \{0, \frac{M}{2}\}$ and convert (37) into a real-valued form. The first of these two steps, accounting for the constraints, is handled by discarding the first and last rows; i.e. a projection which discards the constrained modes.

$$\mathcal{F}_x = \begin{bmatrix} 1 & r & r^2 & r^3 & \dots & r^{(M-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & r^{(\frac{M}{2}-1)} & r^{2(\frac{M}{2}-1)} & r^{3(\frac{M}{2}-1)} & \dots & r^{(\frac{M}{2}-1)(M-1)} \end{bmatrix}. \tag{36}$$

Substituting the identity $r^p = (\cos(\frac{2\pi}{M}) + i\sin(\frac{2\pi}{M}))^p = (\cos(\frac{2\pi p}{M}) + i\sin(\frac{2\pi p}{M}))$ into (36) this expression allows us to easily separate the real and imaginary components; concatenating the imaginary components of this matrix to the bottom of the real component and multiplying the entire matrix by a normalization factor

$$[\mathcal{F}_x] \equiv \sqrt{\frac{2}{M}} \begin{bmatrix} 1 & \cos(\frac{2\pi}{M}) & \cos(\frac{4\pi}{M}) & \cdots & \cos(\frac{2(M-1)\pi}{M}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \cos(\frac{2(\frac{M}{2}-1)\pi}{M}) & \cos(\frac{4(\frac{M}{2}-1)\pi}{M}) & \cdots & \cos(\frac{2(\frac{M}{2}-1)(M-1)\pi}{M}) \\ 0 & \sin(\frac{2\pi}{M}) & \sin(\frac{4\pi}{M}) & \cdots & \sin(\frac{2(M-1)\pi}{M}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \sin(\frac{2(\frac{M}{2}-1)\pi}{M}) & \sin(\frac{4(\frac{M}{2}-1)\pi}{M}) & \cdots & \sin(\frac{2(\frac{M}{2}-1)(M-1)\pi}{M}) \end{bmatrix}. \tag{37}$$

The inverse of (37) can be derived by taking the matrix transpose of (37), namely, $\mathcal{F}_x^{-1} = \mathcal{F}_x^\top$. Recall that (37) was constructed such that it returns the spatial modes defined at a single instant in time $u(t', x_m)$. The generalization to space time is therefore simply derived by taking the Kronecker outer product of (37) with the identity matrix whose diagonal is the same dimension as the time discretization. In other words, we are simply combining $N$ copies of the operator (37) into another, higher dimensional, operator $\mathbf{M}[\mathcal{F}_x] = \mathbb{I}_N \otimes \mathcal{F}_x$. and likewise for the matrix inverse. Note that the Kronecker product is not commutative. The matrix representation for the temporal transforms follows from the previous exposition (37), the only difference being the inclusion of the zeroth mode. The temporal transform matrix is then

$$[\mathcal{F}_t] \equiv \sqrt{\frac{2}{M}} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} & \dots & 1/\sqrt{2} \\ 1 & \cos(\frac{2\pi}{N}) & \cos(\frac{4\pi}{N}) & \cdots & \cos(\frac{2(N-1)\pi}{N}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \cos(\frac{2(\frac{N}{2}-1)\pi}{N}) & \cos(\frac{4(\frac{N}{2}-1)\pi}{N}) & \cdots & \cos(\frac{2(\frac{N}{2}-1)(N-1)\pi}{N}) \\ 0 & \sin(\frac{2\pi}{N}) & \sin(\frac{4\pi}{N}) & \cdots & \sin(\frac{2(N-1)\pi}{N}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \sin(\frac{2(\frac{N}{2}-1)\pi}{N}) & \sin(\frac{4(\frac{N}{2}-1)\pi}{N}) & \cdots & \sin(\frac{2(\frac{N}{2}-1)(N-1)\pi}{N}) \end{bmatrix}. \qquad (38)$$

The factors of $\sqrt{2}$ are written specifically such that the inverse is again given by transposition. The Kronecker product appears once again however with its position interchanged. This is now a generalization of a temporal operator to spacetime, where (37) was the generalization of a spatial operator. The identity matrix reflects the spatial dimension $\mathbf{M}[\mathcal{F}_\mathbf{t}] = [\mathcal{F}_t] \otimes \mathbb{I}_{|\mathbf{k}|}$. This concludes the definition of the spatial and temporal Fourier transform matrices. The spatiotemporal version is created simply via the product $M[\mathcal{F}_t]M[\mathcal{F}_x]$.

## *2.3 Spatiotemporal Symmetries*

One of the most important aspects of this problem, both in terms of finding and classifying solutions, is symmetry. The symmetries of the Kuramoto-Sivashinsky equation considered in the spatiotemporal formulation include Galilean invariance and the symmetry group $G \equiv \mathrm{SO}(2) \times \mathrm{O}(2)$. This spatiotemporal symmetry group provides a new perspective with which to view orbits and their symmetries. This in turn provides a new tool which greatly benefits the computations to follow. It is important to first make the distinction between equivariance and invariance. An orbit is *invariant* with respect to a symmetry operation $g$ if the discretized velocity field $u(t_n, x_m)$ is exactly the same at every collocation point (or equivalently the modes are exactly the same)

$$g \cdot \tilde{u} = \tilde{u} \,. \tag{39}$$

A solution is *equivariant* with respect to $g$ if $g \cdot \tilde{u} \neq \tilde{u}$ but still represents a solution

$$g \cdot f(\tilde{u}) = f(g \cdot \tilde{u}) = 0 \,. \tag{40}$$

Clearly, if an orbit is equivariant with respect to $g_1$ and $g_2$ then it is equivariant with respect to $g_1 \cdot g_2$; likewise for invariance. It logically follows that both equivariant symmetry operations and invariant symmetry operations exist in subgroups. Because invariance is a much stricter requirement, a special name is given to its subgroup: isotropy group. Note that the isotropy group is actually a subgroup of the equivariance subgroup. This distinction is visualized in figure 7 using a shift-reflection invariant orbit. Moving forward, the statements: ‘an orbit has a discrete symmetry’ and ‘an orbit with discrete symmetry’ are always statements of invariance, not equivariance. Likewise, saying an orbit ‘has no symmetry’ is equivalent to saying that its isotropy subgroup is the trivial subgroup; i.e. containing only the identity element. Much like everything else, the symmetry group SO(2) $\times$ O(2) is also affected by discretization. For example, orbits will not be equivariant under arbitrary rotations, as they require interpolations of the field. In other words, orbits can only by equivariant with respect to discrete rotations; making the largest possible equivariance subgroup equal to $\mathrm{C}_N \times \mathrm{D}_M$.

### 2.3.1 Discrete spatiotemporal symmetries

We consider orbits with the following discrete symmetries: spatial reflection invariance, spatiotemporal shift-reflection invariance (‘pre-periodic’ orbits) [29], and equilibria which are invariant under spatial reflection and time translation. For continuous symmetries, relative periodic orbits and relative equilibria are considered. The spatiotemporal symmetry group provides a new perspective on known symmetries, and it makes a connection to plane-Couette flow. Namely, the notion of 'pre-periodicity' can be replaced by spatiotemporal shift-reflection. To see how this presents itself, let us begin with the definition of a 'pre-periodic' orbit. A pre-periodic orbit is an orbit which can be decomposed into a set of 'fundamental domains', whose exact definition is dependent upon the symmetry in question. The importance of such domains is that a single fundamental domain can reproduce an entire orbit via discrete symmetry operations. For pre-periodic orbits, the fundamental domains are halves of the temporal extent. For a simple shift-reflection example, let us choose the halves $t_1 \in [0, T_p/2]$ and $t_2 \in [T_p/2, T_p]$. The fundamental domains are equivalent upon spatial reflection $u(t_1, x) = \sigma u(t_2, x)$. In the spatiotemporal formulation, the notion of pre-periodicity is replaced by spatiotemporal shift-reflection symmetry. This is analogous to the shift-reflection which occurs in plane-Couette flow [47]. To show this, constrain the plane-Couette shift-reflection to two dimensions; the correspondence between the shift reflection in each system is self evident when explicitly written

$$
\begin{aligned}
s_1 w(z,x) &= -w(L_z - z, x + \frac{L_x}{2}) \,, \\
\sigma \circ u(x,t) &= = -u(L - x, t + \frac{T}{2}) \,.
\end{aligned}
\tag{41}
$$

The real benefit of this formulation comes in the form of *selection rules*; constraints imposed on the modes (29) by discrete symmetry invariance. It follows logically that the first goal is then to derive these isotropy subgroups. First, let us define some discrete symmetry operations. Let $\sigma$ represent spatial reflection about $x = \frac{L}{2}$ such that

$$
\sigma u(t_n, x_m) = -u(t_n, L - x_m) \,, \tag{42}
$$

and let $\tau$ represent half-cell $(T/2)$ translations in time

$$
\tau u(t_n, x_m) = u(t_n + \frac{T}{2}, x_m) \,. \tag{43}
$$

Finally, define the spatiotemporal shift-reflection by the composition of these two operations $s = \sigma\tau$ such that

$$
\sigma\tau u(t_n, x_m) = -u(t_n + \frac{T}{2}, L - x_m) \,. \tag{44}
$$

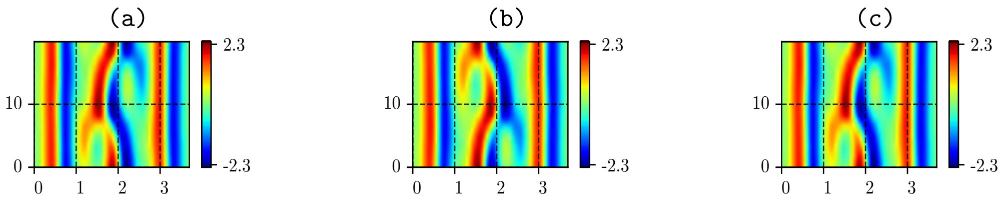

**Figure 6:** (a) An orbit, (b) its spatial reflection (equivariance) (c) its shift-reflection (invariance).

As mentioned previously, we want a spatiotemporal description of antisymmetric and shift-reflection orbits; by definition, these have isotropy subgroups $H_\sigma = \{e, \sigma\}$ and $H_{\sigma\tau} = \{e, \sigma\tau\}$, respectively. Both types of orbit are equivariant under time translations, spatial half-cell shifts and spatial reflection, i.e. the subgroup $\mathrm{C}_N \times \mathrm{D}_2$. It was realized that these isotropy groups are both subgroups of

$$G = \{e, \sigma, \tau, \sigma\tau\} = \mathrm{C}_2 \times \mathrm{Z}_2 \, , \tag{45}$$

consisting of the identity, spatial reflection, half-cell time translation, and spatiotemporal shift-reflection, from left to right. Additionally, $G$ is the isotropy subgroup of equilibria; therefore is captures all of the discrete symmetries that we wanted to include. Using this group, we find three ways to utilize and exploit symmetry invariance: a group theoretic derivation using projection operators and irreducible subspaces, matrix representations of said projection operators and substitution of symmetry operations into the invariance condition $g \cdot \tilde{u} = \tilde{u}$. The 'group theoretic' derivation was found to be relatively simple, but this likely does not generalize to larger groups. The advantage of these group theoretic calculations is that it results in a precise description of invariant subspaces and their relation to one another. The second method, construction of the matrix representations of the projection operators, provides a method of projecting other linear operators onto the invariant subspaces. Lastly, the invariance condition provides an efficient manner to derive the selection rules pertaining to each isotropy group; i.e. application of the symmetry operations (42) and (44) to the modes (29).

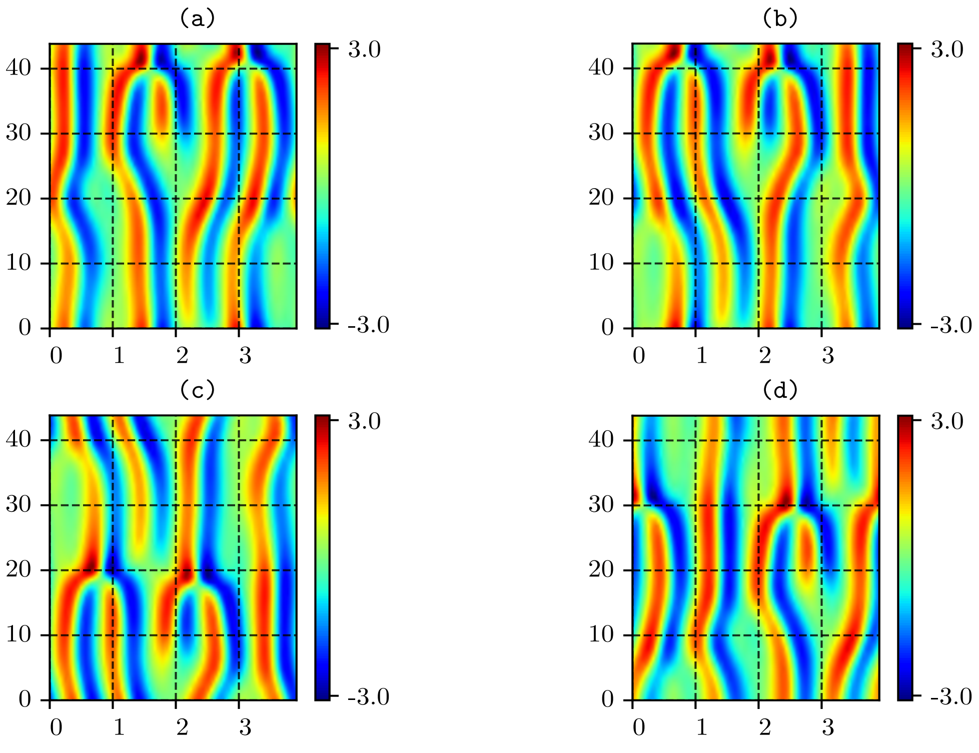

**Figure 7:** Equivariant orbits, (a) orbit with trivial isotropy group, (b) its spatial reflection, (c) its shift-reflection, (d) its rotation by a quarter period in both space and time.

### 2.3.2 Projection operators and spatiotemporal symmetry invariant subspaces

The group theoretic calculations are nearly identical to those in [47, 69]; however, the interpretation is necessarily different due to spatiotemporal nature of our group and the fact that there are no longer any dynamics. To begin, teh first step is to define the character table of (45) [54], shown in table 1. This character table leads to four linear projection operators [24]

$$\begin{aligned} P^{(++)} &= \frac{1}{4}(1+\sigma+\tau+\sigma\tau)\,, \\ P^{(-+)} &= \frac{1}{4}(1-\sigma+\tau-\sigma\tau)\,, \\ P^{(+-)} &= \frac{1}{4}(1+\sigma-\tau-\sigma\tau)\,, \\ P^{(--)} &= \frac{1}{4}(1-\sigma-\tau+\sigma\tau)\,. \end{aligned} \tag{46}$$

Using these projection operators, the solution space (with discrete symmetry) can be decomposed into the irreducible subspaces $\mathbb{U} = \mathbb{U}_{(++)} \oplus \mathbb{U}_{(+-)} \oplus \mathbb{U}_{(-+)} \oplus \mathbb{U}_{(--)}$. Applying the projection operators (46) to the modes in tensor notation yields the four irreducible subspaces of modes

$$\tilde{u}^{(--)} = \begin{bmatrix} a_{jk} & 0 \\ b_{jk} & 0 \end{bmatrix} \forall k, j \text{ odd} , \qquad \tilde{u}^{(-+)} = \begin{bmatrix} a_{jk} & 0 \\ b_{jk} & 0 \end{bmatrix} \forall k, j \text{ even} ,$$

$$\tilde{u}^{(+-)} = \begin{bmatrix} 0 & c_{jk} \\ 0 & d_{jk} \end{bmatrix} \forall k, j \text{ odd} , \qquad \tilde{u}^{(++)} = \begin{bmatrix} 0 & c_{jk} \\ 0 & d_{jk} \end{bmatrix} \forall k, j \text{ even} . \tag{47}$$

A convenient interpretation of these four subspaces are based on being symmetric or antisymmetric with respect to reflections about $L/2$ and half-cell time translations of $T/2$, the latter contributing to the conditions regarding $j$ taking odd or even values within each subspace. The effect of exchanging the projection operators (46) and spatial differentiation operator is derived; as this is directly relevant for the calculation of the nonlinear term in (4)

$$\begin{aligned} D_x P^{(++)} &= P^{(-+)} D_x \, , \\ D_x P^{(+-)} &= P^{(--)} D_x \, , \\ D_x P^{(-+)} &= P^{(++)} D_x \, , \\ D_x P^{(--)} &= P^{(+-)} D_x \, . \end{aligned} \tag{48}$$

Using these relations (48) we can produce the projections of the Kuramoto-Sivashinsky equation onto the different irreducible subspaces

$$\begin{aligned} P^{(++)} f &= u_t^{(++)} + u_{xx}^{(++)} + u_{xxxx}^{(++)} + \frac{1}{2}\partial_x (P^{(-+)} u^2) \, , \\ P^{(+-)} f &= u_t^{(+-)} + u_{xx}^{(+-)} + u_{xxxx}^{(+-)} + \frac{1}{2}\partial_x (P^{(--)} u^2) \, , \\ P^{(-+)} f &= u_t^{(-+)} + u_{xx}^{(-+)} + u_{xxxx}^{(-+)} + \frac{1}{2}\partial_x (P^{(++)} u^2) \, , \\ P^{(--)} f &= u_t^{(--)} + u_{xx}^{(--)} + u_{xxxx}^{(--)} + \frac{1}{2}\partial_x (P^{(+-)} u^2) \, . \end{aligned} \tag{49}$$

With this we can determine the *symmetry invariant subspaces* or simply *invariant subspaces*. This is similar to the notion of *flow invariant subspaces* except that there are no longer any dynamics; 'flow' does not apply here. One method of deriving the invariant subspaces is to simply assume a decomposition of $u$ and then show that the sum of the relevant equations

(49) commutes with combinations of the projection operators (49). For example, assume $u = P^{(++)}u = u^{(++)}$ and substitute into (49)

$$\begin{aligned} f(P^{(++)}u) &= u_t^{(++)} + u_{xx}^{(++)} + u_{xxxx}^{(++)} + \frac{1}{2}\partial_x((u^{(++)})^2) \,, \\ &= u_t^{(++)} + u_{xx}^{(++)} + u_{xxxx}^{(++)} + \frac{1}{2}\partial_x(P^{(-+)}u^2) \,, \\ &= u_t^{(++)} + u_{xx}^{(++)} + u_{xxxx}^{(++)} + \frac{1}{2}P^{(++)}\partial_x(u^2) \,, \\ &= P^{(++)}f(u) \,. \end{aligned} \tag{50}$$

Where the (non-obvious) fact that $(u^{(++)})^2$ exists in the $P^{(-+)}$ subspace was used. Therefore $\mathbb{U}_{(++)}$ constitutes an invariant subspace. This is the invariant subspace containing all equilibria. Using similar substitutions, we find that the symmetry invariant subspaces are $\mathbb{U}_{(++)}$, $\mathbb{U}_{(++)} \oplus \mathbb{U}_{(--)}$, $\mathbb{U}_{(++)} \oplus \mathbb{U}_{(+-)}$ and $\mathbb{U}_{(++)} \oplus \mathbb{U}_{(-+)}$. Addition of the corresponding projection operators (46) results in a more useful representation; three of which exactly correspond to the invariances which motivated this entire derivation

$$\begin{aligned} P_0 \equiv P^{(++)} &= \frac{1}{4}(1 + \sigma + \tau + \sigma\tau) \,, \\ P_\sigma \equiv P^{(++)} + P^{(+-)} &= \frac{1}{2}(1 + \sigma) \,, \\ P_\tau \equiv P^{(++)} + P^{(-+)} &= \frac{1}{2}(1 + \tau) \,, \\ P_{\sigma\tau} \equiv P^{(++)} + P^{(--)} &= \frac{1}{2}(1 + \sigma\tau) \,. \end{aligned} \tag{51}$$

The invariant subspaces trivially follow from the projection operators (51)

$$\begin{aligned} \mathbb{U}_0 &\equiv \{\tilde{u} \,|\, P_0 = \tilde{u} = \tilde{u}\} \,, \\ \mathbb{U}_\sigma &\equiv \{\tilde{u} \,|\, P_\sigma \tilde{u} = \tilde{u}\} \,, \\ \mathbb{U}_\tau &\equiv \{\tilde{u} \,|\, P_\tau \tilde{u} = \tilde{u}\} \,, \\ \mathbb{U}_{\sigma\tau} &\equiv \{\tilde{u} \,|\, P_{\sigma\tau} \tilde{u} = \tilde{u}\} \,, \end{aligned} \tag{52}$$

where $\tilde{u}$ are the modes of (11). $\mathbb{U}_0$ represents the subspace of antisymmetric equilibria, $\mathbb{U}_\sigma$ the spatial reflection invariant subspace, $\mathbb{U}_{\sigma\tau}$ the shift-reflection invariant subspace, and lastly $\mathbb{U}_\tau$ with solutions 'twice-repeating' solutions, i.e. those symmetric with respect to $T/2$; discussion deferred to the end of the symmetry discussion. The continuous spatial translations are eliminated from every subspace *except* $\mathbb{U}_{(++)} \oplus \mathbb{U}_{(-+)}$; shown by applying the generator of translations (spatial differentiation) to the non-vanishing modes. Naturally the definition of $f$ in the shift-reflection and reflection invariant subspaces comes from the substitution of $u^{(++)} + u^{(--)}$ and $u^{(++)} + u^{(+-)}$, or any equivalent expressions, into $f$. This is not a very useful representation of the equations; instead everything is described in terms of the the mode tensor components (29) in sect. 2.4.

**Table 1:** Character table for discrete spatiotemporal symmetry group.

| | $e$ | $\sigma$ | $\tau$ | $\sigma\tau$ |
|---|---|---|---|---|
| $E$ | 1 | 1 | 1 | 1 |
| $\Gamma_1$ | 1 | 1 | -1 | -1 |
| $\Gamma_2$ | 1 | -1 | 1 | -1 |
| $\Gamma_3$ | 1 | -1 | -1 | 1 |

### 2.3.3 Projection operator matrix representations

Using the invariance condition (39) is equivalent to applying the corresponding projection operator (51) and finding which modes survive. The invariance condition will be used to derive the selection rules in a functional form, but the matrix representations of the projection operators (51) are useful for applying the selection rules to other linear operators. The focus is only on reflection and shift-reflection invariance, such that the derivations are limited to the matrix representations of $P_\sigma$ and $P_{\sigma\tau}$. All derivations will utilize the conventions on the notation for identity matrices from (33). In what follows, 'primed' variables are simply temporary placeholders used in the construction of the final expression. For the real and imaginary components of a single spatial mode, the spatial reflection operator (with reflection axis $x = L/2$) is

$$\mathbf{M}[\sigma^{''}] = \text{diag}(-1, 1) \; . \tag{53}$$

This changes the sign of the real component of the mode. Generalizing to a set of $\frac{M}{2} - 1$ spatial modes

$$\mathbf{M}[\sigma^{'}] = \mathbf{M}[\sigma^{''}] \otimes \mathbb{I}_k = \text{diag}(-\mathbb{I}_k, \mathbb{I}_k) \; . \tag{54}$$

The spatiotemporal version of (54) is a diagonal matrix consisting of $N-1$ repeats of (54)

$$\mathbf{M}[\sigma] = \mathbb{I}_{(N-1)} \otimes \mathbf{M}[\sigma^{'}] = \text{diag}(\sigma_0^{'}, \sigma_1^{'}, \dots, \sigma_{N-1}^{'}) \ , \tag{55}$$

with (55) the projection operator $P_\sigma$ can be defined as

$$\mathbf{M}[P_\sigma] = \frac{1}{2}(\mathbb{I} + \sigma) = \text{diag}(0, \mathbb{I}_k, \dots) \, . \tag{56}$$

Which can be seen clearly keeps the spatially antisymmetric components of the modes with respect to the ordering (34). The projection onto the space of equilibria is simply the respect components of (56).

The shift-reflection projection operator requires both the temporal half-cell shift $\tau$ and the reflection operator (55). Half-cell shift is equivalent to rotation by $\pi$ such that (time equivalent of) the expression (71) (SO(2) rotations for $j$ modes) simplifies to

$$\mathbf{M}[\tau^{'}] = \text{diag}(1, (-1)^j, (-1)^j) \ . \tag{57}$$

This specific form of (57) is used to represent the fact that $j$ follows the tensor index convention defined in (31). The spatiotemporal version of (57) is achieved via another Kronecker product; again this is a diagonal matrix with $M-2$ copies of $\tau^{'}$ along its diagonal.

$$\mathbf{M}[\tau] = \mathbf{M}[\tau^{'}] \otimes \mathbb{I}_{(M-2)} = \text{diag}(\tau_0^{'}, \tau_1^{'}, \dots) \ . \tag{58}$$

Finally, using the Kronecker product identity $(A \otimes B)(C \otimes D) = (AC \otimes BD)$ (if the sizes are consistent) we have the projection operator as

$$\begin{aligned} \mathbf{M}[P_{\sigma\tau}] &= \frac{1}{2}(\mathbb{I} + \sigma\tau) \ , \\ &= \frac{1}{2}(\mathbb{I} + (\mathbb{I}_{(N-1)} \otimes \sigma^{'}) \cdot (\tau^{'} \otimes \mathbb{I}_{(M-2)})) \ , \\ &= \frac{1}{2}(\mathbb{I} + (\tau^{'} \otimes \sigma^{'}))) \ , \end{aligned} \tag{59}$$

where

$$\mathbf{M}[(\tau^{'} \otimes \sigma^{'})] \equiv \text{diag}((-1)^{j+1}\, \mathbb{I}_k, (-1)^j\, \mathbb{I}_k, \dots) \ . \tag{60}$$

The indices within (60) are ordered according to the vector representation (34). Substitution of (60) into (59) yields a diagonal matrix where the terms with $j+1$ survive for odd $j$, and the terms with $j$ survive for even $j$. This yields diagonal alternates between two identity matrix blocks and two zero blocks until while also abiding by the ordering of (31)

$$\mathbf{M}[P_{\sigma\tau}] \equiv \mathrm{diag}(0, \mathbb{I}_k, \mathbb{I}_k, 0, 0, \mathbb{I}_k, \mathbb{I}_k, 0, 0, \dots)\,. \tag{61}$$

It is important computationally to discard the modes whose projections are equal to zero. Discarding the vanishing modes requires a modification of the projection operators (56), (61). Note that the projection operators are diagonal matrices and that the vanishing modes correspond to where the diagonal equals zero. Therefore, the vanishing modes are properly discarded by removing these rows of the projection matrices. This concludes the derivation of the two projection operators for antisymmetric and shift-reflection orbits. While these explicit constructions are useful, the projections can be expressed more efficiently by direct substitution of symmetry operations into the invariance condition.

### 2.3.4 Discrete symmetry selection rules

To derive the selection rules for each discrete symmetry, the reflection, time translation and shift-reflection operations need to be defined in terms of the modes (29). Reflection and (half period) time translation come immediately; shift-reflection is less obvious and hence derived explicitly

$$\begin{aligned}
\sigma\tau\, u(t_n, x_m) = -\sum_{k,j} & \Big(a_{jk}\cos(\tilde{\omega}_j(t_n + T/2)) + b_{jk}\sin(\tilde{\omega}_j(t_n + T/2))\Big)\cos(q_k(L - x_m)) \\
& \Big(-c_{jk}\cos(\tilde{\omega}_j(t_n + T/2)) - d_{jk}\sin(\tilde{\omega}_j(t_n + T/2))\Big)\sin(q_k(L - x_m))\,, \\
= \sum_{k,j} & -\cos(\pi j)\Big((a_{jk}\cos(\tilde{\omega}_j t_n) + b_{jk}\sin(\tilde{\omega}_j t_n))\Big)\cos(q_k x_m) \\
& +\cos(\pi j)\Big(-c_{jk}\cos(\tilde{\omega}_j t_n) - d_{jk}\sin(\tilde{\omega}_j t_n)\Big)\sin(q_k x_m)\,, \\
= \sum_{k,j} & (-1)^{j+1}\Big((a_{jk}\cos(\tilde{\omega}_j t_n) + b_{jk}\sin(\tilde{\omega}_j t_n))\Big)\cos(q_k x_m) \\
& +(-1)^j\Big(-c_{jk}\cos(\tilde{\omega}_j t_n) - d_{jk}\sin(\tilde{\omega}_j t_n)\Big)\sin(q_k x_m)\,.
\end{aligned} \tag{62}$$

This results in the following tensor expressions for each operation

$$\sigma\tilde{u} = \begin{bmatrix} -a_{jk} & c_{jk} \\ -b_{jk} & d_{jk} \end{bmatrix} \ , \quad \tau\tilde{u} = \begin{bmatrix} (-1)^j a_{jk} & (-1)^j c_{jk} \\ (-1)^j b_{jk} & (-1)^j d_{jk} \end{bmatrix} \ , \quad \sigma\tau\tilde{u} = \begin{bmatrix} (-1)^{j+1} a_{jk} & (-1)^j c_{jk} \\ (-1)^{j+1} b_{jk} & (-1)^j d_{jk} \end{bmatrix} \quad . \tag{63}$$

which, when substituted into the invariance condition (39), yield the following selection rules. For the antisymmetric orbits, the selection rules and corresponding tensor notation are derived by simply requiring the real spatial component mode blocks to vanish

$$a_{jk},\ b_{jk} = 0 \quad \forall j\ , \forall k\ . \tag{64}$$

The non-vanishing modes can be represented in tensor form as

$$\tilde{u}_\sigma = \begin{bmatrix} c_{jk} \\ d_{jk} \end{bmatrix} \ .$$

The selection rules for equilibria follow trivially from this by selecting only the zeroth temporal modes,

$$\begin{aligned} a_{jk},\ b_{jk},\ d_{jk} &= 0\ , \\ c_{jk} &= 0 \quad \forall j \neq 0\ . \end{aligned} \tag{65}$$

Therefore, the tensor form consists of only a single row of modes

$$\tilde{u}_0 = [c_{0k}]\ . \tag{66}$$

The shift-reflection selection rules

$$\begin{aligned} a_{jk} = b_{jk} = 0 \ \text{ for } \ j \ \text{ even} \ \ , \\ c_{jk} = d_{jk} = 0 \ \text{ for } \ j \ \text{ odd} \ \ , \end{aligned} \tag{67}$$

seem harder to put into tensor format at first but luckily there is a trick that can be exploited to yield a tidier expression. This is most easily seen by explicitly applying the selection rules to (29).

$$\tilde{u}_{\sigma\tau} \equiv \begin{bmatrix} 0 & c_{0,k} \\ a_{1,k} & 0 \\ \vdots & \vdots \\ b_{1,k} & 0 \\ 0 & d_{2,k} \\ \vdots & \vdots \end{bmatrix} \equiv \begin{bmatrix} c_{0,k} \\ a_{1,k} \\ \vdots \\ b_{1,k} \\ d_{2,k} \\ \vdots \end{bmatrix} . \tag{68}$$

.

This staggered structure can be exploited by discarding the zeros and ‘shuffling’ the modes together. By formatting the modes in this way, note that the temporal frequencies are arranged in a manner identical to (29). In other words, no extra work is required to define time differentiation. The down side is that the modes need to be ‘unshuffled’ before the inverse transform can be applied, that is, the zeros present in (68) need to be reinserted. The selection rules are incorporated into the definition of the temporal transforms to yield a ‘symmetry invariant’ Fourier transform. By doing so, there is no possibility of forgetting to apply the selection rules.

Before proceeding to continuous symmetries, here is a summary on what has been covered for discrete symmetries. First, the distinction between equivariance and invariance was made. By doing so, very specific definitions of group orbits and isotropy groups were defined. It was realized that the isotropy subgroups of interest were both subgroups of a larger group which is equivalent to the Klein four group. Owing to the structure of this group, irreducible subspaces and their linear projection operators were defined. Four symmetry invariant subspaces were then derived by acting on the Kuramoto-Sivashinsky equation with combinations of these projection operators. Each of these subspaces has a very useful set of constraints denoted as selection rules (equivalent to projection onto the respective subspace). The selection rules were defined in both a functional form and as matrices. Finally, the rules were incorporated into the temporal transforms, referring to these new transforms as symmetry invariant Fourier transforms. For orbits within each invariant subspace the redefined Fourier transforms remain orthogonal transformations.

### 2.3.5 Continuous symmetries

There are two continuous symmetries that are accounted for, Galilean invariance and spatial translation symmetry which results in relative periodic orbits. The first of these two symmetries, invariance under Galilean transformations, results in the following equivariance: if $u(t,x)$is a solution, then $u(t,x-ct)-c$ is also a solution ($c$ being an arbitrary constant speed or 'mean flow'). This is handled with a simple constraint which constrains the mean flow with respect to space

$$\langle u\rangle(t) \, = \int_0^L dx\, u(x,t) = 0\,. \tag{69}$$

The practical method to enforce this is to simply discard the zeroth spatial modes, i.e. those with $k=0$. This is a conventional choice but not necessarily the best choice; there is to be a longer discussion on this later, but the general idea is that reducing the group orbit without purpose is not desirable. Finding orbits is the goal, not finding specific members of group orbits. It is believed that reducing the dimension of the group orbit may reduce the likelihood of finding that group orbit numerically. Regardless, the convention defined by (69) is enforced.

The last spatiotemporal symmetry to discuss is spatial translation symmetry which defines relative periodic orbits. To make a distinction between uniform rotations, this symmetry will be referred to as *spatial shift* symmetry. A relative periodic orbit is defined here as an orbit whose field $u(t_n,x_m)$ is only periodic after accounting for a 'spatial drift' or 'shift'. In other words, relative periodic orbits abide by the following relation

$$u(t,x) = g \circ u(t+T,x)\,, \tag{70}$$

where $g$ represents a spatial shift defined by the $f$. If this shift is not accounted for, the $u(t,x)$ will be discontinuous in time. To make the most out of the Fourier basis some intervention is required to generate a truly periodic representation of the field.

Two ideas come to mind: either utilize a comoving reference frame, or slice (quotient) the continuous symmetry [12]. The slicing method has a number of disadvantages and so only a comoving frame is used. Namely, there is the notion of 'in-slice' time which can induce sharp discontinuities in the field; a bad property for Fourier representations.

Additionally, applying the slicing condition to the equations spatiotemporally results is some quite confusing equations, much more than the comoving frame ansatz. The comoving frame and its spatial shift needs to be formulated in dimensionless spatial units as opposed to an angular quantity. This is important because $T$ and $L$ and $s$ are changing in the numerical optimization process and hence, they are coupled. The matrix representation of SO(2) group elements for a spatial shift by an amount $s$ is given by

$$g(s) \equiv \begin{bmatrix} \cos(q_k s) & \sin(q_k s) \\ -\sin(q_k s) & \cos(q_k s) \end{bmatrix} . \tag{71}$$

Note that there is an extra negative sign contained within $q_k$, again, kept in this form to match the computational expressions (27). The specific ordering of the modes results in a tri-band matrix, not a block diagonal matrix with blocks of dimension $2 \times 2$. In other words, the index $k$ indicate that each element (71) represents a diagonal matrix where $k$ takes values (30). This matrix applied to a set of spatial modes defined at a specific instant in time, shifts them by $s$ in the positive $x$ direction. The spatiotemporal version of this operator, that is, uniform spatial rotations of the entirety of $u(t_n, x_m)$, is a block diagonal matrix with $N$ blocks; one for each value of $t_n$. Both uniform spatial translations and comoving rotations are applied in the spatial mode basis, as it makes more sense considering the parameterization in time. The uniform shift (71) generalizes to spacetime via Kronecker product as seen before in (58)

$$\mathbf{M}[g(s)] = \mathbb{I}_N \otimes g(s) \ , \tag{72}$$

which, explicitly, equals

$$\mathbf{M}[g(s)] = \text{diag}(g(s), g(s), \dots, g(s)) \, . \tag{73}$$

The general idea originates in López *et al.* [82] where the field is kept in a comoving reference frame. The comoving reference frame transformation is a generalization of (73) where the previously uniform shift $s$ is replaced by a spatial shift linearly parameterized by time. That is, at every discrete time $t_n$, the field is shifted by an amount $\mathrm{S}t_n/T$ via the symmetry operation $g(\mathrm{S}t_n/T)$. For brevity let $S_n \equiv S(t_n) \equiv \mathrm{S}t_n/T$. Using (71) the matrix representation of the comoving frame transformation is as follows

$$\mathbf{M}[g(\phi_n)] = \mathrm{diag}(g(\phi_{N-1}), g(\phi_{N-2}), \dots)\,. \tag{74}$$

Note that the time is decreasing along the diagonal due to the conventions of the numerical implementation. Using (74), the ansatz for relative periodic solutions can be written as the modification of the spatiotemporal transform (27). The most succinct form of the comoving transformation is writing the expression in terms of operators for the translations and Fourier transforms.

$$u(t_n, x_m) = \mathcal{F}_x^{-1}\big(\mathbf{g}^{-1}(\mathrm{S}t_n/T)\mathcal{F}_t^{-1}(\tilde{u})\big)\,. \tag{75}$$

Where the action of $\mathbf{g}^{-1}(\mathrm{S}t_n/T)$ maps $x_m \to x_m - \mathrm{S}t_n/T$, a translation in the positive spatial direction. Equivalently, this can be written as the inverse Fourier transform of rotated spatial modes

$$\begin{aligned}\mathbf{g}\, u(t_n, x_m) = \sqrt{\frac{2}{M}}\sum_{k=1}^{\frac{M}{2}-1} \; &\big(e_k(t_n)\cos(q_k\mathrm{S}t_n/T) + f_k(t_n)\sin(q_k\mathrm{S}t_n/T)\big)\cos(q_k x_m) \\ &+\big(-e_k(t_n)\sin(q_k\mathrm{S}t_n/T) + f_k(t_n)\cos(q_k\mathrm{S}t_n/T)\big)\sin(q_k x_m)\;, \\ = \sqrt{\frac{2}{M}}\sum_{k=1}^{\frac{M}{2}-1} \; & e_k(t_n)\cos(q_k(x_m - \mathrm{S}t_n/T)) + f_k(t_n)\sin(q_k(x_m - \mathrm{S}t_n/T))\,. \end{aligned} \tag{76}$$

An example of this transformation is displayed in figure 8.

An important detail is that (76) reproduces the $u(t_n, x_m)$ which solves (4), *however*, numerically the field is kept always kept in the comoving frame; such that symmetry operation is not built into the temporal Fourier transform as it is for other symmetries. The reasoning for this choice is that numerically we desire the modes corresponding to the periodic comoving frame only; the price being an additional term in the differential algebraic equations. Note that this ansatz also accounts for relative equilibria, which are equilibria in the comoving frame such that the vanishing modes of relative equilibria are

$$a_{jk} = b_{jk} = c_{jk} = d_{jk} = 0 \quad \forall\, j > 0\;. \tag{77}$$

This concludes the discussion regarding continuous spatial translation symmetry, until the derivation of the differential algebraic equations. Before moving on, one last comment regarding the subspace corresponding to $P_\tau$ of (51). This subspace contains the 'twice-repeating' solutions of the Kuramoto-Sivashinsky equation. That is, any orbit has a representation which exists in this invariant subspace, created by simply taking a tiling of size $[0, 2T] \times [0, L]$. Note that one must use this representation of the orbit otherwise it would not be *invariant* under the transformation. This seems trivial at first, however, it *does* impose selection rules; namely, the modes with odd temporal index $j$ must vanish. One interpretation of this is that shadowing an orbit for two full periods is a much stricter requirement than shadowing for a single period; this strictness manifesting as selection rules. While not without its hypothetical uses, this subspace and its selection rules are currently not utilized. Hypothetically it is possible that spatiotemporal symmetry groups of higher order (i.e. higher order cyclic subgroups) would permit more interesting invariances than simply 'twice-repeating' invariance. Why? For example, two repeats of a shift reflection orbit is invariant under more combinations of symmetry actions. Compositions of quarter-cell and three-quarter cell shifts with reflection leave the orbit invariant. However, these quarter period shifts are not described by the discrete symmetry group used here; presenting open doors for future investigations.

The spatiotemporal formulation offers a new perspective on the symmetries of the Kuramoto-Sivashinsky equation. Previously, the reflection symmetry manifested as a flow-invariant subspace, unstable to perturbations; the shift-reflection symmetry as 'pre-periodic' orbits which are quite different from one another. Now, every discrete spatiotemporal symmetries manifests as a set of selection rules; constraints which require subsets of modes to vanish (equal zero). Additionally, this spatiotemporal description demonstrates that there is actually a relation between antisymmetric and shift-reflection orbits by way of invariant subspaces (52). In regards to continuous symmetries, relative periodicity is now described with comoving reference frames. This description benefits the numerical methods as it affords a periodic representation of the field. Therefore, in summary, the symmetry classes of orbits considered here include antisymmetric, shift-reflection, relative periodic, equilibrium and relative equilibrium orbits.

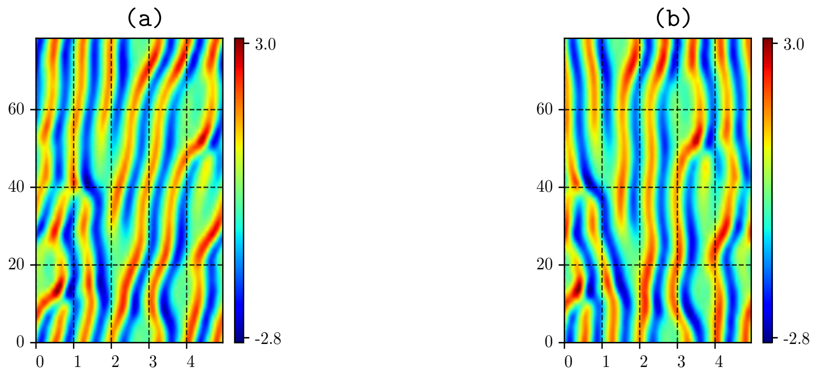

**Figure 8:** Relative periodic orbit $(T, L) \approx (78.36, 4.96(2\pi\sqrt{2}))$, (a) in the comoving reference frame, (b) in the physical reference frame. The spatial shift from (a) to (b) is $S_n = -1.37(2\pi\sqrt{2})t_n/T$.

## *2.4 Differential algebraic equations*

Now that the spatiotemporal Fourier transforms and spatiotemporal symmetries have been properly derived, the Kuramoto-Sivashinsky equation can now be written as a system of differential algebraic equations. These equations are derived by substitution of (27) into (4); using the pseudospectral product to compute the nonlinear term. That is, computing the elementwise product of the values of $u$ at the collocation points. Technically speaking, the equations are derived by substitution of the total Fourier series, using orthonormality relations and truncation to define our finite set of differential algebraic equations. We do not find it useful to provide such a derivation as it results in a convolution sum expression for the nonlinear term; which, to reiterate, is not computed in the pseudospectral formulation. The pseudospectral method requires operations involving tensors, vectors and matrices; These operations include element-wise products of tensors and Kronecker outer products; the latter only used for the explicit construction of linear operators. The element-wise multiplication is required for both the pseudospectral product and also *spectral differentiation*; element-wise products of modes and the corresponding powers of spatial or temporal frequencies. This is slightly more complicated in the real-valued, SO(2), representation as it requires tracking of an additional factor of $-1$ (equivalent to tracking the powers of $i$ for the complex representation) as well as some reordering of the modes for odd-ordered derivatives. These details can be represented explicitly via the differentiation matrices, i.e. the generators of

SO(2) rotations

$$\mathbf{M}[\partial_x] \equiv \mathbb{I}_{|\mathbf{j}|} \otimes \begin{bmatrix} 0 & q_k \\ -q_k & 0 \end{bmatrix} , \tag{78}$$

and for time

$$\mathbf{M}[\partial_t] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\omega_j \\ 0 & \omega_j & 0 \end{bmatrix} \otimes \mathbb{I}_{|\mathbf{k}|} \,. \tag{79}$$

The time differentiation is close to the same form as (78) except for the inclusion of the zeroth time modes and the additional factor of $-1$ due to the time ordering of $u$. We remind the reader to recall the conventions of the signs of the frequencies (22), in case there is any confusion regarding the signs of the terms in these generators. The derivatives of higher order can be derived from the appropriate powers of (78) and (79). In the tensor representation (29) the element-wise products for spatial derivatives up to fourth order are

$$\partial_x \tilde{u} = \begin{bmatrix} q_k c_{jk} & -q_k a_{jk} \\ q_k d_{jk} & -q_k b_{jk} \end{bmatrix} , \qquad \partial_{xx} \tilde{u} = \begin{bmatrix} -q_k^2 a_{jk} & -q_k^2 c_{jk} \\ -q_k^2 b_{jk} & -q_k^2 d_{jk} \end{bmatrix} ,$$

$$\partial_{xxx} \tilde{u} = \begin{bmatrix} -q_k^3 c_{jk} & q_k^3 a_{jk} \\ -q_k^3 d_{jk} & q_k^3 b_{jk} \end{bmatrix} , \qquad \partial_{xxxx} \tilde{u} = \begin{bmatrix} q_k^4 a_{jk} & q_k^4 c_{jk} \\ q_k^4 b_{jk} & q_k^4 d_{jk} \end{bmatrix} , \tag{80}$$

$$\partial_t \tilde{u} = \begin{bmatrix} -\omega_j b_{jk} & -\omega_j d_{jk} \\ \omega_j a_{jk} & \omega_j c_{jk} \end{bmatrix} . \tag{81}$$

Note that the zeroth $j = 0$ mode components in (81)are equal to zero due to the definition (79). These derivatives are all that are required for the linear terms of (4). Using the derivatives in tensor notation (81), (80) the linear component of the Kuramoto-Sivashinsky equation written in tensor representation is

$$(\partial_t + \partial_{xx} + \partial_{xxxx})\tilde{u} = \begin{bmatrix} (-q_k^2 + q_k^4)a_{jk} - \omega_j b_{jk} & (-q_k^2 + q_k^4)c_{jk} - \omega_j d_{jk} \\ (-q_k^2 + q_k^4)b_{jk} + \omega_j a_{jk} & (-q_k^2 + q_k^4)d_{jk} + \omega_j c_{jk} \end{bmatrix} . \tag{82}$$

For the nonlinear term: in operator notation the pseudospectral implementation of the

Kuramoto-Sivashinsky equation, the nonlinear term is written

$$f(v) \equiv \frac{1}{2}\partial_x \mathcal{F}((\mathcal{F}^{-1}(\tilde{u}) \cdot \mathcal{F}^{-1}(\tilde{u}))\,. \tag{83}$$

The product $u^2$ is equal to the products of each of its components, i.e. the products of sums within (27). The block wherein each product belongs is derived simply by noting the parity of the trigonometric polynomials. For example, the parity of the product $\mathcal{F}^{-1}(a_{jk}) * \mathcal{F}^{-1}(c_{jk})$ can be crudely derived by inspection of the parity of products of the form $\cos(\omega_j t_n)\cos(q_k x_m)\cos(\omega_{j'} t_n)\sin(q_{k'} x_m)$. For non-vanishing components, the products will be odd with respect to space and even with respect to time; the correspondence is then $\cos(\omega_j)\sin(q_k)$ hence it lies in the $\tilde{N}_{c_{jk}}$ mode block. Applying this logic to all other products yields an expression for the nonlinear component of (83); let $(\cdot)$ represent the element-wise product. Let $a = \mathcal{F}^{-1}(a_{jk}),\ b = \mathcal{F}^{-1}(b_{jk}),\ c = \mathcal{F}^{-1}(c_{jk}),\ d = \mathcal{F}^{-1}(d_{jk})$. The full expression of (83) in terms of the mode blocks is then

$$\begin{bmatrix} \tilde{N}_a & \tilde{N}_c \\ \tilde{N}_b & \tilde{N}_d \end{bmatrix} = \begin{bmatrix} q_k \mathcal{F}(a\cdot c + b\cdot d) & -\frac{q_k}{2}\,\mathcal{F}(a\cdot a + b\cdot b + c\cdot c + d\cdot d)) \\ q_k\,\mathcal{F}(a\cdot b + c\cdot d) & -q_k\,\mathcal{F}(a\cdot d + b\cdot c) \end{bmatrix}, \tag{84}$$

such that the Kuramoto-Sivashinsky equation (4) in terms of spatiotemporal modes is

$$f(v) = \begin{bmatrix} (-q_k^2 + q_k^4)a_{jk} - \omega_j b_{jk} + \tilde{N}_a & (-q_k^2 + q_k^4)c_{jk} - \omega_j d_{jk} + \tilde{N}_c \\ (-q_k^2 + q_k^4)b_{jk} + \omega_j a_{jk} + \tilde{N}_b & (-q_k^2 + q_k^4)d_{jk} + \omega_j c_{jk} + \tilde{N}_d \end{bmatrix}. \tag{85}$$

To be explicit (85) can be written as a system of equations as well, this is included simply to help the reader with interpretation of the tensor representation. Namely $f = 0$ is equivalent to the system of equations

$$\begin{aligned} 0 &= (-q_k^2 + q_k^4)a_{jk} - \omega_j b_{jk} + q_k \mathcal{F}(a\cdot c + b\cdot d) \\ 0 &= (-q_k^2 + q_k^4)b_{jk} + \omega_j a_{jk} + q_k\,\mathcal{F}(a\cdot b + c\cdot d) \\ 0 &= (-q_k^2 + q_k^4)c_{jk} - \omega_j d_{jk} - \frac{q_k}{2}\,\mathcal{F}(a\cdot a + b\cdot b + c\cdot c + d\cdot d)) \\ 0 &= (-q_k^2 + q_k^4)d_{jk} + \omega_j c_{jk} - q_k\,\mathcal{F}(a\cdot d + b\cdot c)\,. \end{aligned} \tag{86}$$

With the general expression (83) in place, the discrete symmetry variants of (85) can be derived. This requires a modification to the pseudospectral product. The reason being the nonlinear term (83) for orbits with discrete symmetry (52) would yield $P^i\mathcal{F}(\mathcal{F}^{-1}(\tilde{u}^2)) = 0$ for all mode components. This is not to say that the nonlinear term vanishes, rather, the spatial differentiation is required to map the modes from the complement back into the invariant subspace. This is simply the nature of the selection rules and their relation to spatial differentiation (48). The accommodation employed is to take the spatial derivative in the spatial mode basis, preceding the time transform (with its built in projection). With this modification we are now fully equipped to handle the discrete symmetry variants of (85). In the $\mathbb{U}_0$ subspace of antisymmetric equilibria the only non-zero modes are given by (66); the non-vanishing components of (85) are then

$$f_0(v) = \left[(-q_k^2 + q_k^4)c_{0k} - \tfrac{1}{2}\mathcal{F}_t(q_k\mathcal{F}_x(\mathcal{F}^{-1}(c_{0k}) \cdot \mathcal{F}^{-1}(c_{0k})))\right] \,. \tag{87}$$

Continuing on with the antisymmetric subspace $\mathbb{U}_\sigma$ using the selection rules (67)

$$f_\sigma(v) = \begin{bmatrix} (-q_k^2 + q_k^4)c_{jk} - \omega_j d_{jk} + \frac{1}{2}\mathcal{F}_t(q_k\mathcal{F}_x(c\cdot c + d\cdot d) \\ (-q_k^2 + q_k^4)d_{jk} + \omega_j c_{jk} + \mathcal{F}_t(q_k\mathcal{F}_x(c\cdot d)) \end{bmatrix} \,. \tag{88}$$

Unfortunately there is no useful simplification for the shift reflection subspace as each block has non-vanishing modes; however, it should be noted that the components of (85) also obey the selection rules (67).

For relative periodic orbits and relative equilibria, substitution of the comoving frame ansatz (76) into (4) generates an additional linear term via the time derivative of $x_m - \frac{St_n}{T}$. The result can explicitly be represented by an additional linear term of the form $-\frac{S}{T}\partial_x\tilde{u}$ such that

$$\begin{aligned} f_S(v) \;&=\; \begin{bmatrix} (-q_k^2 + q_k^4)a_{jk} - \omega_j b_{jk} + \tilde{N}_{a_{jk}} & (-q_k^2 + q_k^4)c_{jk} - \omega_j d_{jk} + \tilde{N}_{c_{jk}} \\ (-q_k^2 + q_k^4)b_{jk} + \omega_j a_{jk} + \tilde{N}_{b_{jk}} & (-q_k^2 + q_k^4)d_{jk} + \omega_j c_{jk} + \tilde{N}_{d_{jk}} \end{bmatrix} \\ &\quad - \begin{bmatrix} q_k\frac{S}{T}c_{jk} & -q_k\frac{S}{T}a_{jk} \\ q_k\frac{S}{T}d_{jk} & -q_k\frac{S}{T}b_{jk} \end{bmatrix} \end{aligned} \tag{89}$$

In order to explicitly construct the Jacobian of $f$ and its adjoint it will be helpful to construct (83) using matrices where possible. For brevity, the general case (i.e. no symmetry) is simply included in the discrete symmetry case; allowed because the spatial differentiation commutes with the time transform without projections; however, the comoving frame ansatz gets its own versions explicitly expressed. Note that the element-wise multiplication (pseudospectral product $u^2$) within the nonlinear term can be represented by multiplication of the vector form of $u$ with a diagonal matrix with $u$ along its diagonal. In spite of this, it benefits us to keep it in pseudospectral form, as later on the Jacobian will come more naturally. To accommodate orbits with discrete symmetries, the nonlinear term is written $\mathbf{M}[\mathcal{F}_t \partial_x \mathcal{F}_x] = \mathbf{M}[\mathcal{F}_t]\mathbf{M}[\partial_x]\mathbf{M}[\mathcal{F}_x]$ such that

$$f(v) = (\mathbf{M}[\partial_t] + \mathbf{M}[\partial_{xx}] + \mathbf{M}[\partial_{xxxx}])\tilde{u} + \frac{1}{2}\mathbf{M}[\mathcal{F}_t \partial_x \mathcal{F}_x](\mathcal{F}^{-1}(\tilde{u}) \cdot \mathcal{F}^{-1}(\tilde{u})) \,, \tag{90}$$

and

$$f_S(v) = f(v) - \frac{S}{T}\mathbf{M}[\partial_x]\tilde{u} \,. \tag{91}$$

Solving for the roots of (83) (and its symmetry subspace variants) is at the core of all spatiotemporal computations. To do so it is necessary to derive the Jacobian of $f$, and its action on vectors, with respect to all variables represented in $v$.

$$\mathbf{J} \equiv \frac{\partial f}{\partial v} = \left[\frac{\partial f}{\partial \tilde{u}}, \frac{\partial f}{\partial \mathrm{T}}, \frac{\partial f}{\partial \mathrm{L}}\right] ,$$

$$\mathbf{J}_0 \equiv \frac{\partial f_0}{\partial v} = \left[\frac{\partial f}{\partial \tilde{u}}, \frac{\partial f}{\partial \mathrm{L}}\right] ,$$

$$\mathbf{J}_S \equiv \frac{\partial f_S}{\partial v} = \left[\frac{\partial f_S}{\partial \tilde{u}}, \frac{\partial f_S}{\partial \mathrm{T}}, \frac{\partial f_S}{\partial \mathrm{L}}, \frac{\partial f_S}{\partial \mathrm{S}}\right] . \tag{92}$$

These Jacobian matrices are defined by $|\tilde{u}|$ equations (number of modes) and $|\tilde{u}| + p$ unknowns (modes and parameters). Using the matrix representation (90) the Jacobian can be easily defined using the 'direct-matrix' calculus [20]

$$(\mathbf{J})_{\tilde{u}} = \mathbf{M}[\partial_t] + \mathbf{M}[\partial_{xx}] + \mathbf{M}[\partial_{xxxx}] + \mathbf{M}[\mathcal{F}_t \partial_x \mathcal{F}_x] \,\mathrm{diag}[\mathcal{F}^{-1}(\tilde{u})]\, \mathbf{M}[\mathcal{F}^{-1}] \,, \tag{93}$$

and for relative periodic orbits

$$(\mathbf{J}_S)_{\tilde{u}} = \mathbf{J}_{\tilde{u}} - \frac{S}{T}\mathbf{M}[\partial_x] \,. \tag{94}$$

where $\mathrm{diag}[\mathcal{F}^{-1}(\tilde{u})]$ is a diagonal matrix with $u$ as its elements. The derivatives with respect to the parameters $T, L$ apply to the frequencies within (83). The dependence on $T$ and $L$ is always in the form of inverse powers, i.e. $T^{-1}$, $L^{-4}$, and so on. The convention is to write the partial derivatives in terms of the original factor; e.g. $(-4/L)L^{-4}$ instead of $(-4/L^{-5})$. Computationally it allows us to represent everything in terms of scalar multiplication of previously defined components. The partial derivative of $f$ with respect to $T$ is given by

$$\frac{\partial f}{\partial \mathrm{T}} = -\frac{1}{T}\partial_t \tilde{u}\,, \tag{95}$$

and likewise for the relative periodic orbits

$$\frac{\partial f_S}{\partial \mathrm{T}} = \frac{\partial f}{\partial \mathrm{T}} - \frac{1}{T}\Big(-\frac{S}{T}\partial_x \tilde{u}\Big)\,. \tag{96}$$

For the derivative with respect to the spatial period we have

$$\frac{\partial f}{\partial \mathrm{L}} = \Big(-\frac{2}{L}\partial_{xx} - \frac{4}{L}\partial_{xxxx}\Big)\tilde{u} - \frac{1}{2L}\mathcal{F}_t\partial_x\mathcal{F}_x\left(\mathcal{F}^{-1}(\tilde{u})\cdot\mathcal{F}^{-1}(\tilde{u})\right)\,, \tag{97}$$

and for relative periodic orbits

$$\frac{\partial f_S}{\partial \mathrm{L}} = \frac{\partial f}{\partial \mathrm{L}} - \frac{1}{L}\Big(-\frac{S}{T}\partial_x \tilde{u}\Big)\,. \tag{98}$$

Finally, the partial derivative with respect to the spatial shift parameter for relative periodic orbits

$$\frac{\partial f_S}{\partial \mathrm{S}} = -\frac{1}{T}\partial_x \tilde{u}\,. \tag{99}$$

Another component required for the numerical techniques is to derive the adjoint of (93). The general expression can be immediately derived by linearization of the adjoint equation (16) with respect to $\lambda$ and then using the matrix representation (90) such that

$$(\mathbf{J})^{\top}_{\tilde{u}} = -\mathbf{M}[\partial_t] + \mathbf{M}[\partial_{xx}] + \mathbf{M}[\partial_{xxxx}] - \mathbf{M}[\mathcal{F}]\,\mathrm{diag}(\mathcal{F}^{-1}(\tilde{u}))\,\mathbf{M}[\mathcal{F}_x^{-1}\partial_x\mathcal{F}_t^{-1}]\,, \tag{100}$$

and

$$(\mathbf{J}_S)^{\top}_{\tilde{u}} = (\mathbf{J})^{\top}_{\tilde{u}} + \frac{S}{T}\mathbf{M}[\partial_x]\,. \tag{101}$$

In (100) and (101) the orthogonal property of operators has been utilized; i.e. the transposition results in the inverse operators. Unlike the Jacobian (93), the adjoint (100) is never constructed explicitly. Instead, we require the matrix-vector product with the adjoint Jacobian. In conventional numerical studies, matrix-vector products most commonly arise as finite-difference approximations of tangent space evolution. In the context of the pseudospectral spatiotemporal method, we can compute these without such approximations, as the Jacobian of the differential algebraic equations (83) and its adjoint are no longer implicitly dependent on time *evolution.* In other words, this enables the pseudospectral computation of the matrix-vector product with the Jacobian, avoiding the usage of finite differences completely.

In summary, this chapter derived all of the necessary components to discretize and solve the Kuramoto-Sivashinsky equation using a spatiotemporal Fourier mode basis, including: the spatiotemporal Fourier transform (27), the differential algebraic equations (83), its Jacobian (93) and adjoint Jacobian (100). Additionally, the spatiotemporal formulation offers a new description of solutions of the Kuramoto-Sivashinsky equation with discrete symmetries, via the inception of a spatiotemporal symmetry group. With these results in place, the numerical optimization methods and other spatiotemporal techniques can be developed.

# CHAPTER III

# NUMERICAL METHODS

## *3.1 Optimization algorithms*

Nearly every result of the numerical studies presented here involves solving the minimization problem $\phi(v) = 0$ in one way or another. Therefore, it is imperative to develop robust numerical methods to accomplish just that. In this section we introduce the terminology and the optimization algorithms used to derive all results. Recently, many new methods have been made available in our computational codes, but they have not been tested enough to be commented on. Additionally, while included, a number of the following algorithms are unsuited to the Kuramoto-Sivashinsky equation. They are included, however, because as they may useful to others in the future, and the inclusion of these extra algorithms comes at no extra cost; each method comes in a 'class' as determined by SciPy, the computing package they originate from [134]. Because of how they are grouped, making a single algorithm available is equivalent to making all algorithms in that class available. SciPy and NumPy constitute the main computational Python packages used in the spatiotemporal numerical codes [53, 134]. To begin the discussion on numerical optimization let us first introduce some terminology.

As a blanket term 'the' optimization of a guess orbit denotes the process of using any one of the numerical algorithms in an attempt to monotonically decrease the residual (19). The optimization of a guess orbit $\tilde{v}$ has *converged* if the residual becomes smaller than some predetermined threshold or *tolerance*, denoted in the codes by `tol`. Currently only the simplest requirement is imposed on the numerical methods, namely, we require that regardless of the numerical method, the residual must be a strictly non-increasing function

$$\phi(v_n + \delta v_n) < \phi(v_n) \, , \tag{102}$$

where $\delta v_n$ is a 'state correction' produced by an optimization algorithm. If this monotonic behavior is violated then we deem the guess orbit a failure. Some common ways of reducing the likelihood of failure include introducing a line-search, trust-region, and backtracking techniques [27]. We elect to apply the latter, this takes the form of a damping parameter in class of linear least-squares solvers, and as a step size in the custom descent methods. In the solvers imported from other computational packages, there is less control and customization, unfortunately. The least-squares approaches the problem by either iteratively or directly solving the so-called ‘Newton equations’ [135]. The relevant Newton equations can be derived by linearizing about an orbit $v^* = v + \delta v$ where $f(v^*) = 0$.

$$f(v + \delta v) \approx f(v) + \mathbf{J}\delta v \; + \mathcal{O}(\delta v^2)\,. \tag{103}$$

Substitution of zero for the LHS yields

$$\mathbf{J}\delta v = -f\,. \tag{104}$$

The solution $\delta v$ of (104) will be referred to as a *Newton step* or *Newton correction.* Note that because $T, L, S,$ are kept as independent variables, (104) represents an under-determined system of equations, hence the reason why this is approached in a least squares manner. The system of equations (104) is solved in one of two ways: explicit construction of the Moore-Penrose pseudoinverse $\mathbf{J}^{+} = (\mathbf{J}^{\top}\mathbf{J})^{-1}\mathbf{J}^{\top}$, or applying iterative methods to the *normal equations*

$$\mathbf{J}^{\top}\mathbf{J}\delta v = -\mathbf{J}^{\top} f\,. \tag{105}$$

Newton’s method applied to a least-squares problem is commonly referred to as the Gauss-Newton method [7, 27, 111]. The iterative methods available for solving the normal equations of the Newton system (105) include: MINRES [102], Bi-CG [35], Bi-CGSTAB [131], GMRES [112], LGMRES [4], CG [55], CGS [119], QMR [42], and GCROT(m,k) [26]. The linear least-squares solvers include LSQR [103], LSMR [39], and LSTSQ [106].

The last detail before providing the actual algorithm is the addition of backtracking. Backtracking, sometimes categorized as an inexact line-search method [97], simply reduces the step size taken in the direction of the Newton step derived by either the direct or iterative methods’ solution to (105). The simplest form of backtracking is employed here, which includes a coefficient which damps the Newton step

$$v_{n+1} = v_n + \tau_k \delta v_n\,. \tag{106}$$

To ensure sufficient decrease in the residual, the step size $\tau_k$ is bounded from below by some minimal value, typically on the order of $10^{-4}$. The strategy for producing $\tau_k$ is again, as simple as possible; start with $\tau_0 = 1$ dividing it by a factor of two while the residual decrease condition is not met. Before defending this simplistic choice, it is worth mentioning that the plan is to eventually replace this crude condition with something more sophisticated such as the Goldstein or Wolfe conditions [97], but, for now, we stick to simplicity. The equations (105) are technically solved repetitively, generating a sequence of corrections; the $n$th correction given by the solution to

$$\mathbf{J}^\top(v_n)\mathbf{J}(v_n)\delta v_{n+1} = -\mathbf{J}^\top(v_n)f(v_n)\,. \tag{107}$$

i.e. as $\mathbf{J}$ and $f$ are functions of $v$, with every correction $\mathbf{J}(v), f(v)$ need to be updated; to better represent the actual algorithm, this detail is accounted for by overwriting the current state with the new state, and so the index notation of (107) is avoided.

**Algorithm 1** Gauss-Newton method with backtracking

```
Require: $N \geq 1, \mathrm{tol} > 0, v \in \mathrm{H}^{1,4}_{\mathrm{per}}(\Omega), 0 < \tau_{\min} < 1, \rho \in (0,1)$
  $n \Leftarrow 1$
  $b \Leftarrow -f(v)$
  $r \Leftarrow \frac{1}{2}b^2$
  while $r > \mathrm{tol}$ and $n < N$ do
    $\tau_0 = 1$
    $\mathbf{A} \Leftarrow \mathbf{J}(v)$
    {Solve $A\ \delta v = b$ for $\delta v$}
    while $\frac{1}{2}f(v + \tau_k \delta v)^2 > r$ do
      $\tau_k \Leftarrow \rho\tau_{k-1}$
      if $\tau_k < \tau_{\min}$ then
        return $v$
      end if
    end while
    $n \Leftarrow n + 1$
    $v \Leftarrow v + \tau\delta v$
    $b \Leftarrow -f$
    $r \Leftarrow \frac{1}{2}b^2$
  end while
  return $v$
```

The second main class of algorithms are the class of descent methods. The methods in this class that are available for use are: conjugate gradient method (CG), the Broyden-Fletcher-Goldfarb-Shanno method (BFGS), Newton-CG method [97], limited memory BFGS

(L-BFGS-B) [14], the truncated Newton algorithm (TNC) [95] and the previously mentioned adjoint descent algorithm [33]. These algorithms are characterized by their usage of local gradient information to find directions in which to 'descend', i.e. reduce the residual. These methods typically do not solve a least-squares system, i.e. (105), except for the 'Newton descent' method [33, 76, 77].

The (custom) descent algorithms are formulated by introducing a *fictitious time* $\tau$ such that $f(v(\tau)) \to 0$ as $\tau \to \infty$. The corrections take the form $\delta v = (\partial_\tau v)\delta\tau$ such that

$$\phi(v + (\partial_\tau v)\delta\tau) \leq \phi(v) \; , \tag{108}$$

where $v(\tau)$ is represented simply as $v$. The determine what we should choose for $\partial_\tau v$, we differentiate the cost function (19) with respect to $\tau$

$$\partial_\tau \phi = \left[\mathbf{J}\partial_\tau v\right]^\top f \, , \tag{109}$$

where $\mathbf{J} = \frac{\partial \mathbf{f}}{\partial v}$ is the Jacobian matrix (93). Therefore the minimization problem is defined by choosing $\partial_\tau v$ such that (109) is a decreasing function of $\tau$. The first such choice is to require $\partial_\tau v$ to satisfy

$$\mathbf{J}\partial_\tau v = -f \, , \tag{110}$$

as

$$\partial_\tau \phi = -|f|^2 \; \leq 0 \, . \tag{111}$$

This choice defines the *Newton descent* method . Solving (110) for $\delta v$ is exactly the same as solving the 'Newton system' or 'Newton equations' hence the moniker. The inclusion of 'descent' comes from the infinitesimal step size $\tau$. Algorithms which fall into the 'descent method' category are methods which use the cost function and local gradient information in order to produce a 'descent direction' in which numerically stepping reduces the residual. These conditions are similar but less precise than the Wolfe conditions or Armijo-Goldstein conditions; the optimization considered here does not concern itself with finding an optimal step length as defined by a line-searching procedure [97]. The alternative descent method is the *adjoint descent* method [33]. This adjoint descent method is derived by noting that

(109) can be written

$$\partial_\tau \phi = (\partial_\tau v)^\top \Big[\mathbf{J}^\top f\Big] \,, \tag{112}$$

such that

$$\partial_\tau v = -\mathbf{J}^\top f \,, \tag{113}$$

yields

$$\partial_\tau \phi = -\big[\mathbf{J}^\top f\big]^2 \le 0\,. \tag{114}$$

This formulation includes the adjoint of the Jacobian hence the name 'adjoint descent'. There are a number of important details worth noting. First, it should be noted for the set of differential algebraic equations, the Jacobian and its adjoint can be explicitly constructed; they are not implicitly dependent on time evolution as is the typical case [24, 29, 33]. Going even further, the adjoint itself actually does not even need to be explicitly constructed; all that is required is the ability to calculate the matrix-vector product, which can be done without the use of finite-difference approximations [10, 71]. The components of the matrix-vector product $\mathbf{J}^\top f$ come immediately from adjoint Jacobian equation (100) and the Euler-Lagrange equations for the parameters (16)

$$\begin{aligned} (\mathbf{J}^\top f)_{\tilde{u}} &= -f_t + f_{xx} + f_{xxxx} - u \cdot f_x \,, \\ (\mathbf{J}^\top f)_{p_i} &= f^\top \frac{\partial f}{\partial p_i} \,. \end{aligned} \tag{115}$$

As $f$ can be computed in a matrix free manner, it follows that (115) can be as well. Lastly it should be noted that the adjoint descent direction (113) is simply the negative gradient of the cost-function. Therefore, (113) can be directly interpreted as the gradient descent method. In practice the minimization problem was approached via the combination of the adjoint descent method followed by the 'LSTSQ' algorithm which explicitly constructs the pseudoinverse using a 'divide-and-conquer' singular value decomposition [134]. Therefore, these two methods will be the main focus of the derivations, as the other methods have been added very recently and no investigations have been made into their efficacy. Both of these algorithms have custom implementations, meaning that they are not simply imported from a computational package; hence, a more detailed description is readily available. In

**Algorithm 2** Adjoint descent method

```
Require: $N \geq 1, \mathrm{tol} > 0, v \in \mathrm{H}^{1,4}_{\mathrm{per}}(\Omega), 0 < \tau_{\min} < 1, \rho \in (0,1)$
  $n \leftarrow 1$
  $r \leftarrow \frac{1}{2}f^2$
  while $r > \mathrm{tol}$ and $n < N$ do
    $\partial_\tau v \leftarrow -\mathbf{J}^\top(v)f(v)$
    while $\frac{1}{2}f(v + \tau_k\partial_\tau v)^2 > r$ do
      $\tau_k \leftarrow \rho\tau_{k-1}$
      if $\tau_k < \tau_{\min}$ then
        return $v$
      end if
    end while
    $n \Leftarrow n + 1$
    $v \leftarrow v + \tau_k\partial_\tau v$
    $r \Leftarrow \frac{1}{2}f(v)^2$
  end while
  return $v$
```

fact customization of different algorithms is recommended because many of the 'built-in' methods do not take full advantage of the problem. For example, by virtue of having differential algebraic equations, the matrix-vector product of the Jacobian with an arbitrary vector can be written explicitly. In order to use these two numerical methods, given by algorithm 1 and algorithm 2, to solve the minimization problem, initial guesses for orbits are required.

## 3.2 Orbit hunting

A surprisingly hard component of finding periodic orbits in chaotic systems is finding good candidates for initial conditions. Historically, initial conditions are produced by finding the local minima of the *recurrence function* [3, 18, 31, 122]. A recurrence function calculates the pairwise distance between points in a time series

$$R(t, t') = \frac{|u(t + t') - u(t)|}{|u(t)|} \,. \tag{116}$$

A trajectory segment nearly closing on itself, or 'close recurrence', manifests as a local minimum of the recurrence function. Therefore, finding these local minima yields guesses for periodic orbits, in terms of the state space point and approximate period $t'$.

There are many problems with such methods however, especially in the limit of large dimensions. The most obvious disadvantages are related to the computation of the recurrence function; it requires time integration and computations of all (or perhaps a subsample) pairwise distances. Both of these operations become more costly in the high dimensional limit. While this is nothing to scoff at, there are two more disadvantages that are arguably far worse. Firstly, the $L_2$ norm used in the computation of pairwise distances does not incorporate the geometry of the state space (this is not to imply these is a better or obvious choice of norm). In other words, two state-space points can be close in this norm, yet very far away from each other dynamically; i.e. on opposite sides of an unstable manifold. Therefore, even the minima if (116) are not guaranteed to yield 'good' initial conditions, i.e. those which converge to orbits after application of numerical optimization methods. This previous disadvantage, of course, is predicated on being able to find *any* minima of adequate quality; something which is not guaranteed in finite time integration.

These weaknesses of recurrence methods are well known but improvements are constantly being made. For instance, a recent development compares more than just the difference between state space points. Methods that minimize the distance between *segments* of state space trajectories should be much more robust. For example, in Page & Kerswell [101] dynamic mode decomposition (DMD) [114, 115] approach, the spatiotemporal data in form of a time series of spatial snapshots yields a low-dimensional approximation to the local tangent space. This, coupled with identification of repeated harmonics in the DMD eigenvalue spectrum, makes possible detection of periodic orbits in short time series, as short as a quarter of period of a given periodic orbit, i.e., much before any state space recurrence. While the replacement of state space points by state space trajectories is a much needed improvement over the pairwise point distances of traditional time-recurrence methods, the method still has its limitations, such as only permitting the variability of the temporal periods and, of course, using dynamics.

The downsides of close recurrences and potential improvements have been described, but unfortunately there are few, if any, alternatives afforded to the dynamical systems formulation. Fortunately, the robust nature of the spatiotemporal formulation and its numerical methods allows us to find orbits using initial conditions produced by simply initializing

Fourier modes to have random values; a concept which is so simple it is easy to overlook how impressive it actually is. The construction of initial conditions in this manner contains three steps: first, define a discretized tile, then initialize the values of the corresponding modes with randomly sampled values. Lastly, apply any other desired pre-processing steps, modifying the magnitude of the modes to attempt to emulate the physical scales of the Kuramoto-Sivashinsky equation. This method of generating orbit guesses is the crudest of all possible methods, while it does work it is not very efficient. The spatiotemporal techniques: clipping described in sect. 3.3 and gluing described in sect. 3.5 methods are much better alternatives. The issue, however, is that they require a collection of orbits to have already been found (technically clipping can be applied to trajectories, but this is avoided as it requires time integration).

The two requirements for defining a discretized tile are the dimensions of the tile $(T, L)$ and the corresponding dimensionality of the collocation grid $(N, M)$. Determining the dimension of the collocation points is important, however, only approximate guidelines have been developed. The one requirement is that the discretization size scales with the tile dimensions, motivated by the extensive chaos [32] of the Kuramoto-Sivashinsky equation; the dimension of its inertial manifold scales as a function of $L$ [37]. Therefore, it follows that in order to resolve all 'physical' modes [28, 144], the discretization dimension must also increase according to the dimension of the inertial manifold [36, 109]. When possible, powers of two are utilized to exploit the efficiency of the fast Fourier transform. Tiles of intermediate size are discretized following 'power-of-two' rules

$$\begin{aligned} N &= \max(2^{\lfloor \log_2(T)-1 \rfloor}, 16) \,, \\ M &= \max(2^{\lfloor \log_2(L) \rfloor}, 16) \,. \end{aligned} \tag{117}$$

Unfortunately, increasing the discretization by a factor of two is a nontrivial jump in computational memory requirements; this is becomes dangerous as the discretization becomes progressively larger. Therefore, it is useful to construct an alternative strategy which increases in small increments as a function of tile dimensions. This strategy is supported by the *microextensive* chaos of the Kuramoto-Sivashinsky equation [124]; the fractal dimension increases linearly even for small changes in $L$. This manifests as the strategy

$$\begin{aligned} N &= \max(\lfloor 4T^{1/2} \rfloor, 16) \, , \\ M &= \max(\lfloor 6L^{1/2} \rfloor, 16) \, . \end{aligned} \tag{118}$$

In order to ensure that the collocation grids have an even number of points in every dimension, they are floor divided and then multiplied by 2.

After a tile has been created the values of $\tilde{u}$ need to be initialized. The main method of producing initial conditions is to simply randomly initialize a set of modes and then modulate them in such a way that the resulting spectrum mimics the physical scales of the Kuramoto-Sivashinsky equation. This process of course also incorporates the selection rules (64), (67) for guesses which have discrete symmetry.

The first step in initializing the mode values is to simple draw from a random normal distribution. After this first step, a strategy for how to modulate the modes needs to be decided upon. The most logical strategy is to create a spectrum which mimics the relevant physical scales of the Kuramoto-Sivashinsky equation. The spatial physical scale is determined by the most unstable wavelength discussed in chapter 1. A natural characteristic time scale would be one derived from something intrinsic such as Lyapunov exponents. It should be noted that while these strategies make the most logical sense, it can also be beneficial to contradict our intuition and actually instantiate guesses with relatively large magnitude high spatial frequency modes. This only works in the context of preconditioned numerical methods, which effectively damp these modes and control the magnitude of the partial derivatives with respect to the tile dimensions. The only reason why these methods are believed to work is due to the variational formulation, the damping of the numerical methodssect. 3.1 and the fact that these types of guesses start far away from any local mimima.

The modulation strategies are described by the element-wise product of the random modes with mollifying functions. These mollifiers are designed to be functions of the mode indices (31) and possibly some extra parameters. As the mollifiers are identical for each mode block (29), they can be described in terms of the mode indices $j, k$. The Gaussian mollifier takes the form (119), an example guess orbit is displayed in figure 9

$$\psi_{jk} = \exp\Bigg( -\frac{(j-\mu_t)^2}{2\sigma_t^2} - \frac{(k-\mu_x)^2}{2\sigma_x^2}\Bigg)\,. \tag{119}$$

The 'Gaussian-in-time-exponential-in-space' (GTES) mollifier is given by (120)

$$\psi_{jk} = \exp\Bigg( -\frac{(j-\mu_t)^2}{2\sigma_t^2} - \frac{|k-\mu_x|}{\sigma_x^2}\Bigg)\,. \tag{120}$$

The two mollifiers (119), (120) are the only mollifiers dependent on $j$ in a manner other than simple truncation. The remainder of the mollifiers are the 'exponential'

$$\psi_{jk} = \begin{cases} \exp\big(-\frac{|k-\mu_x|}{\sigma_x^2}\big) & \forall k,\ j < \tilde{j} \text{ else,} \\ 0 & \end{cases} \tag{121}$$

and 'linear-exponential'

$$\psi_{jk} = \begin{cases} \exp\big(-|(q_k^2 - q_k^4) - (q_{\tilde{k}}^2 - q_{\tilde{k}}^4)|\big) & \forall k,\ j < \tilde{j} \text{ else,} \\ 0\,. & \end{cases} \tag{122}$$

and finally, the last mollifier simply consists of temporal mode truncation.

$$\psi_{jk} = \begin{cases} 1 & \forall k,\ j < \tilde{j} \text{ else,} \\ 0\,. & \end{cases} \tag{123}$$

and of course there is the option for no modulation, although this is highly discouraged. These modulations are not equally effective; many other functions have been tried for the sake of experimentation. For large tiles the gaussian in time exponential in space modulation seems to work well, but the smallest the residual has been made is order 1. It is therefore hard to provide a recommendation for a single or 'best' manner with which to produce orbit approximations. The numerical methods we employ do not seem to be interested in our desire to produce a physically motivated construction method drawn from our experience and intuition. Figure 9 and figure 14 represent initial guesses (with no symmetry) generated by application of the mollifiers just described. In each figure, the initial mode values are identical so that direct comparison of mollifiers (119)-(122) can be made.

Before moving on, we give some typical values for the previous initial condition creation methods in the original numerical hunt. In this search, orbits' tile dimensions were chosen

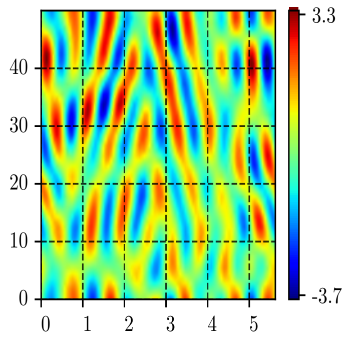

**Figure 9:** Guess orbit created with Gaussian modulation.

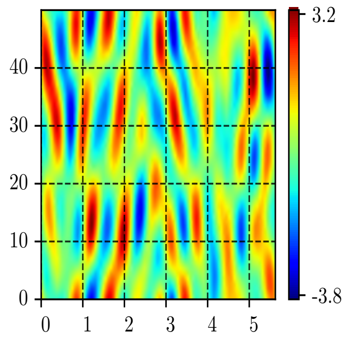

**Figure 10:** Guess orbit created with exponential modulation.

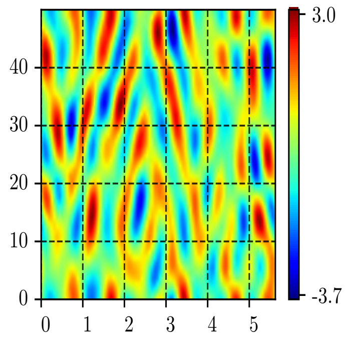

**Figure 11:** Guess orbit created with 'GTES' modulation.

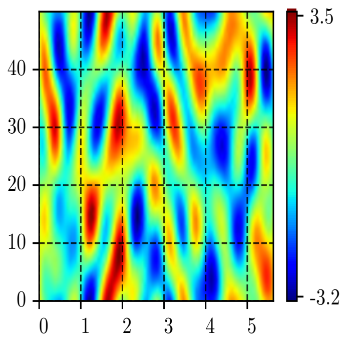

**Figure 12:** Guess orbit created with 'linear exponential' modulation.

from the ranges $T \in [20, 200]$ and $L \in [22, 88]$. The spatial periods were chosen such that the orbits' tile dimensions ranged from sizes comparable to other studies $L \approx 2.5(2\pi\sqrt{2}$ [25, 78] up to tiles deemed to be of 'intermediate' size $L \approx 7.5(2\pi\sqrt{2}$; beyond which finding orbits from random initial conditions is too inefficient. Most of the initial conditions were generated by the simple time truncation modulation strategy (123); reason being the other methods had not been developed yet. The discretization sizes abided by the powers of two strategy (117). The optimization process consisted of two stages: first, the adjoint descent method in algorithm 1 is applied to the orbit guess to bring the guess closer to an orbit, at which point the guess is passed to the Gauss-Newton with backtracking algorithm, algorithm 2. In the adjoint descent method the default tolerance was originally set to be $10^{-4}$, the maximum number of iterations dependent on the dimension of the tile; a value of $16NM$ being the most common choice. Originally the tolerance of the Gauss-Newton algorithm was set to be machine precision, i.e. $10^{-14}$ and the number of steps set to be quite large; typically in the hundreds, due to the presence of backtracking. If the error tolerance was met after application of the combination of these two algorithms, the corresponding orbit is saved and stored in the collection, to be used in the application of gluing and clipping in the future. The next numerical method, and the first original spatiotemporal method, is the clipping method; the method which enables the hunt for fundamental orbits.

### *3.3 Orbit clipping*

In order to find fundamental orbits, a numerical method which extracts small spatiotemporal subdomains from larger orbits or trajectories is required. This technique is denoted clipping

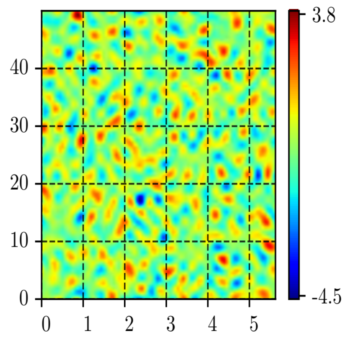

**Figure 13:** Guess orbit created without modulation.

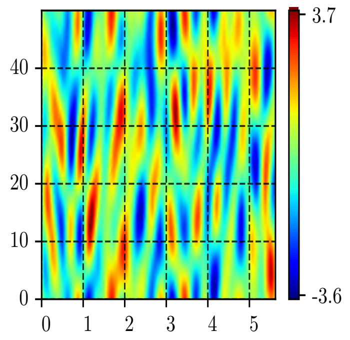

**Figure 14:** Guess orbit created with time truncation modulation.

and is defined as follows. Given any periodic orbit (or arbitrary spatiotemporal domain), a 'clipping' is a window of spacetime which is extracted from an orbit or trajectory. The process is exactly as the name suggests; first choose an orbit from which to clip. Then transform said orbit into the physical field basis. The clipping consists of the collocation points $\Omega'_{nm} = \{\Omega_{nm} \mid t^-_n \leq t_n \leq t^+_n, x^-_m \leq x_m \leq x^+_m\}$ and corresponding field values $u(t'_n, x'_m)$. In practice, the window boundaries are allowed to be specified as continuous values. These boundaries are provided in terms of the 'plotting coordinates'; this allows for clipping based upon visual inspection. To prevent any confusion or miscalculation, every dimension is rescaled to the interval $[0, 1]$. This is in anticipation of negative coordinates, i.e. $[-1, 1]$ for the wall-normal dimension in plane-Couette flow [47].

Iterating through each dimension, the clipping function finds and uses the nearest collocation points as the actual boundaries; therefore the tile dimension returned will be typically be different than what is provided; for this reason, we recommend using interpolation prior to clipping, if higher precision is desired. An important detail to note is that the new tile must be have an even number of points in each dimension. Let $n^+$ and $n^-$ represent the grid points closest to the provided boundaries. If both are even or both are odd, no additional operation is required. If one is odd and one is even, however, then $N' = n^+ - n^-$ is odd, which is not allowed (at least for the Kuramoto-Sivashinsky equation). The two solutions are to modify $n^+, n^-$ such that both are odd, or both are even. The former seemed more prone to possible mistakes, hence the latter is used. This is accomplished by dividing both values by two and taking the integer part, followed by multiplication by two.

By definition clippings do not obey the necessary boundary conditions or differentiability requirements; hence, they do not exist in the necessary space of functions (12). Additional processing steps are therefore required before using the clipping as an orbit guess. Specifically these extra steps consist of interpolation before the clipping and then truncation post clipping; the former being mentioned previously. Upon interpolation the original orbit will generally no longer be an orbit as determined by the residual at the collocation points; unless it was converged with a large enough number of points as to eliminate aliasing within numerical precision. There are two reasons for the interpolation however: first, increasing the resolution enables more control over the location of the clipping boundaries. Second, due to the discontinuity at the boundaries, the modes will be contaminated by the Gibbs phenomenon. Without interpolation, this error contaminates the low frequency modes due to aliasing. Therefore the combination of interpolation and truncation is essentially an anti-aliasing procedure; truncation of these high frequency modes essentially projects the clipping onto onto the space of trigonometric polynomials thereby satisfying the boundary conditions and differentiability requirements. The other method to improve a clipping is to simply choose better locations for the window boundaries; that is, boundaries that minimize the discontinuities. This process has not been automated (at least not yet) in order to ensure that the returned orbit instance best represents the user's desired clipping; however, one could imagine a 'flexible' clipping routine which searches for the lowest residual clipping by allowing the window boundaries to vary slightly. The last additional detail is to decide whether or not to impose any symmetries onto the clipping. This is discouraged for discrete symmetries unless it is glaringly obvious that the clipping exists in the antisymmetric or shift-reflection invariant subspaces. Even when this is case, it is still recommended to assume no symmetry. Instead, the recommended approach is to clip assuming no symmetry, apply numerical optimization, convert to a discrete symmetry class and then use optimizaiton once more.

In regards to continuous symmetry; if a certain pattern is being targeted and it is believed to shadow a relative periodic orbit, then the clipping must be initialized as such in order to capture this behavior. The 'guess' for the spatial shift $S$ being calculated by finding the shift value which minimizes the norm of the difference at the temporal boundary $u(0,x) - g \circ u(t_{N-1},x)$. The reason behind these contradictory approaches for discrete and continuous symmetries is that the selection rules can be satisfied by orbits assumed to have no symmetry. However, orbits without symmetry can not capture relative periodic orbits because it requires an extra degree of freedom; the spatial shift.

With these definitions, guess orbits can now be produced by clipping; with this, smaller orbits shadowed by larger orbits can be found. The guess orbits' tile dimensions depend on the clipping, begging the question, how do we account for finding orbits corresponding to the 'same' pattern that were clipped differently? In other words, a different initial condition will converge to a different orbit by virtue of the least-squares solving (105). This question and more are explained by the next numerical method to be described, numerical continuation.

## *3.4 Orbit continuation*

To explore the continuous families of orbits discovered later on, a practical continuation method is required; that is, a method of constraining a dimension to a new value and updating the modes and other dimensions via optimization. Such methods go by a handful of names including pseudo-arclength continuation, homotopy methods, practical continuation [97]. In the context of differential algebraic equations, this is accomplishing by simply imposing a constraint on the continuation dimension. Consequentially, the state vector's (11) dimension is reduced by 1; the linear systems (105) and descent directions (113) modified accordingly. The application of these types of constraints are recommended only for the sake of exploring continuous families of solutions. If we start with an orbit described by the state vector $v$ defined on $\Omega_{nm}$, continuation of the spatial dimension would be performed by setting $L \rightarrow L'$ such that $\Omega'_{nm} = (t_n, x'_m) = (t_n, \frac{mL'}{M})$. To continue a solution to have $L'$ where $|L'-L| >> 0$, the continuation must be subdivided into small steps, i.e. $L_i^{'} = L_i + \delta L$ where $i \in \{0, \dots, \frac{(L'-L)}{\delta L}\}$. For each 'step', the dimension is constrained to $L_i^{'}$ and then passed through the optimization methods until convergence or failure. If the continuation fails, it is assumed that the orbit does not exist on a tile of size $L_i$, not that the numerical methods have failed us. By 'incorrectly' changing the dimension in this way, the residual changes according to the partial derivatives, i.e. $\Delta F = \frac{\partial F}{\partial L}\delta L$, and likewise for time. Because of the form of these partial derivatives, it is recommended to make much smaller steps when continuing with respect to space rather than time. Also, the spatial continuation problem can actually, and likely should be, approached by continuation in time, if the temporal period corresponding desired spatial dimension is known. For example, continuation of an (idealized example) orbit defined on $(T, L) = (15, 15)$ can be continued to $(T, L) = (20, 20)$ by either continuation in time, constraining $T'$ until $T' + \delta T = 20$, or by constraining space; as the dimensions are dependent on one another. The former, constraining time, seems to perform much better due to the order of the spatial derivatives, hence the larger amount of error introduced by the constraint $L' = L + \delta L$. Therefore, it is recommended to either continue in time (if spatial period is known) or to take much smaller spatial steps.

The second type of continuation utilized is *discretization continuation*. It can be useful to increase and decrease the resolution of an orbit; while maintaining its status as an orbit (residual beneath a threshold). To do so, we can apply truncation or interpolation, changing the discretization size (the minimum increment being 2), followed by reapplication of the optimization methods. It is possible to increase and decrease the discretization by amounts larger than two; testing with respect to these step sizes has not been performed. For a general discretization continuation $(N, M) \rightarrow (N', M')$ this process is typically performed in one of two ways, either one dimension at a time $(N, M) \rightarrow (N', M) \rightarrow (N', M')$ (technically the order is determined by the efficiency, in a computational memory sense) or by 'cycling' through the axes, $(N, M) \rightarrow (N+2, M) \rightarrow (N+2, M+2) \rightarrow \dots$. This concludes the continuation definition, we will now move onto the final spatiotemporal method, gluing.

## *3.5 Orbit gluing*

The ultimate technique developed in the course of this research is referred to as gluing. This technique is simple and intuitive; especially in the context of the two-dimensional spacetime of the Kuramoto-Sivashinsky equation. The primary motivation for the development of the gluing method was to develop a technique that could combine spatiotemporal configurations of fundamental orbits. In practice, gluing can be applied to orbits of any size; if all orbits are comprised of fundamental orbits then gluing larger orbits together is by all means and purposes the same as gluing converged combinations of fundamental orbits together. Therefore, every orbit added to the collection provides new opportunities to find even more orbits. By alternating between gluing and optimization, progressively larger orbits can be found. The two main challenges with gluing result from differences in tile dimension and discontinuities along the gluing boundaries. The former of these issues is tackled in the context of arbitrarily sized *symbolic configurations.* A configuration is a symbolic representation of the guess orbit; it is paramount to understand and so an example configuration (124) and its resulting guess orbit figure 15 are demonstrated explicitly.

$$\begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} \equiv \begin{bmatrix} A & B \\ B & A \end{bmatrix} \tag{124}$$

. The easiest gluing method consists of crudely approximated all tiles as having the same size; applying interpolation or truncation such that all collocation grids have the same dimensions. The guess orbit is produced by simply joining all tiles and their corresponding field values in the manner prescribed by the spatiotemporal configuration. The dimensions of the guess orbit are set to be the average, multiplied by the dimension of the configuration in the respective dimension. For example, the dimensions of a three-by-three configuration $s_t \times s_x = 3 \times 3$ would equal $(\bar{T}, \bar{L}) = (s_t \sum_i T_i,\ s_x \sum_i L_i)$. This method implicitly assumes that all tiles have approximately equal dimensions.

In the case of disparate dimensions, the advice we provide is to reconsider even performing the gluing, or to apply the 'strip-wise' gluing correction. To explain the concept behind strip-wise gluing, first 'aspect-ratio correction' needs to be explained. Let us use

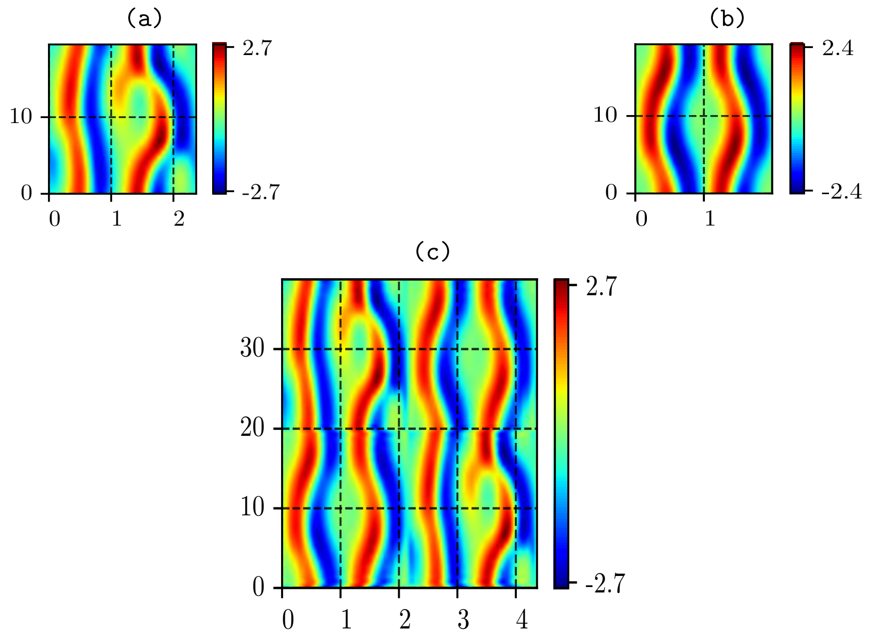

**Figure 15:** (a) Orbit 'A', (b) Orbit 'B', (c) the configuration 'AB_BA' in string notation, equivalent to (124).

the simplest possible gluing configuration as an example; two orbits undergoing one dimensional gluing, i.e. gluing a pair of orbits in space or time. For the sake of this example, this is chosen to be space. It can be generally assumed that the two orbits exist on tiles with differing dimensions $T' \neq T^{''}$, $L' \neq L^{''}$ (unless continuation was performed). In order for gluing to be well defined numerically, the collocation grids must have the same number of points along the dimensions which are transverse to the gluing. These two requirements imply that the grid spacings are necessarily different, i.e. because $T' \neq T^{''}$, setting $N^{'} = N^{''}$ implies that $t_n^{'} \neq t_n^{''}$ (other than at 0). If no modifications were made, the simple gluing previously described would result in the tile with dimensions $\bar{T} = \frac{T'+T^{''}}{2}$, $\bar{L} = L' + L^{''}$ and discretization sizes $M = M' + M'' = 2M'$, $N = N' = N^{''}$ such that $(T, L) = (\frac{n\bar{T}}{N}, \frac{m\bar{L}}{M})$. The resulting tiles' collocation grid spacing is necessarily different from the original (under the current assumptions). With the current strategy of averaging or summing depending on dimension the difference between original and final grid spacing scales with the difference of the original dimensions, i.e.

$$\begin{aligned}\Delta t'_n - \Delta\overline{t_n} &= = \frac{T^{'} - T^{''}}{2N}\,,\\ \Delta x'_m - \Delta\overline{x_m} &= = \frac{L^{'} - L^{''}}{2M}\,.\end{aligned} \tag{125}$$

The relations end up looking the same because the number of points in time is constant but the number of spatial points doubles in size. The deformations defined by (125) introduce error locally, however. By the nature of the global basis functions and the nonlinearity having dependence on $L$, the error is not so easily described.

The aspect-ratio correction aims to improve upon this by first rediscretizing the tiles such that their grid spacings in the gluing dimension are as close as possible prior to gluing, resulting in $\bar{x_m} \approx x'_m \approx x^{''}_m$. Assume that in the previous example $L' = 2L^{''}$; the aspect ratio correction corresponds to resizing by either interpolation or truncation such that $M' = 2M^{''}$, which ensures the grid spacings are the same, i.e. $\frac{L'}{M'} = \frac{L^{''}}{M''}$. In this idealized case, the periods were integer multiples of one another; in reality this is never the case, hence the new collocation grid is never error free with respect to the gluing dimension.

The 'strip-wise gluing' technique is essentially the extension of the aspect-ratio correction to arbitrarily sized (rectangular) spatiotemporal configurations. The complication arises from the fact that the discretization must be identical in all transverse dimensions during the gluing process. As the name suggests, strip-wise gluing is the decomposition of the $D$ dimensional gluing problem into many different 1-dimensional gluing problems. Assume we are given a spatiotemporal configuration with dimensions $(s_1, s_2, s_3, s_4)$, i.e. a 3+1 dimensional spacetime, and that all tiles start with identical discretizations. In the one-dimensional example, the aspect-ratio correction could freely resize the tiles to have an arbitary number of points in the gluing dimension. With the inclusion of more dimensions, the necessary modification is that the total discretization size is fixed. Instead of freely rediscretizing the gluing dimension the aspect-ratio correction in $D$ dimensions determines how to appropriately distribute the fixed number of points amongst the original tiles within the strip, prior to gluing. For example, given a strip within the four dimensional configuration, indexed in the four dimensional tensor as $(0, 0, s_3, 0)$, the total discretization would be (using identical sizes originally) $s_3 * N$. Based on their periods in this dimension, $s_3 * N$ is subdivided redistributed between the number of orbits ($s_3$), to yield a strip of 'aspect-ratio corrected' orbits, such that the sum of the new discretizations equal $s_3 * N$. Using the two-dimensional spacetime for a more concrete example, for clarity, imagine we start with four orbits $N = 32, M = 32$; The total discretization will be $N = 32, \bar{M} = 32 * 4 = 128$. Now assume that the spatial periods of the four orbits can be written as fractions of the total, $L_1 = \bar{L}/8$, $L_2 = \bar{L}/4$, $L_1 = 3\bar{L}/8$, $L_4 = \bar{L}/4$; the aspect ratio correction would rediscretize the orbits prior to gluing according to $M_1^{'} = \bar{M}/8$, $M_2^{'} = \bar{M}/4$, $M_3^{'} = 3\bar{M}/8$, $M_4^{'} = \bar{M}/4$, of course accounting for even valued $M_i$.

After this transformation, the corresponding tiles and their discretized fields are concatenated along the gluing dimension; resulting in a $D - 1$ spatiotemporal configuration. This continues until a stripwise gluing has been performed in all $D$ dimensions. The last detail concerns how many stripwise gluings are performed; This is determined by the spatiotemporal configuration; for a configuration of shape $(s_1, s_2, s_3, s_4)$, if $s_1$ is chosen as the first dimension, there would be $s_2 s_3 s_4$ strips, is this was followed up by strip-wise gluing in the $s_2$ dimension, then there would be $s_3 s_4$ strips, and so on and so forth. Note that due to the nature of strip-wise gluing, the gluing order is not a commutative process; therefore, it should be chosen according to the dimensions of the spatiotemporal configuration or the dimensions of the tiles therein.

The second main challenge of the gluing method results from the fact that there will be discontinuities along the gluing boundaries. While we can always apply the interpolation-truncation technique described in sect. 3.3 to glued orbits, it would be better to find a technique which attacks the error at the source, i.e. reducing the magnitude of the discontinuities. Any transformation will affect the tangents of the glued orbit; therefore the predisposition should be to favor simple, minor transformations. The crudest method which is employed is to glue tiles consisting of fundamental orbits padded by regions of zeros, i.e. the fundamental orbit fields' compact support being a subdomain of the tile. This technique is motivated by the fact that if all tiles are padded, then they will at least be continuous by definition; however, it is not well motivated from a mathematical standpoint. An example

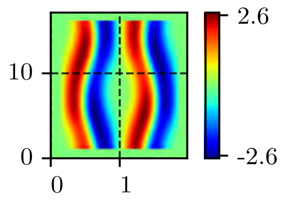

**Figure 16:** A fundamental orbit with numerically zero-padded 'border'.

of this 'padding' is shown in figure 16.

Only a single 'smart' technique has been developed so far; that is, a technique which helps with the discontinuity error while also remaining computationally cheap, relative to the optimization problem. This more expensive version of gluing explores the group orbits of each constituent; finding the combination of representatives which produce the guess with smallest residual. The original implementation of the gluing was designed such that it was a process which 'conserved' the symmetry of orbits. In other words, antisymmetric orbits could be glued to other antisymmetric orbits, and the resulting guess would be assumed to be antisymmetric. In order to impose this condition, it was actually the fundamental domains of each orbit with discrete symmetry that would be glued together. In the same spirit, only the discrete symmetry component of the group orbit was used in the gluing of orbits with discrete symmetries. The original motivation was that it was unknown whether or not different fundamental orbits existed in the different subspaces; in other words, with our original choice the result from gluing would automatically be in the corresponding subspace. However, upon further consideration it was determined that this is in fact not necessary; even though they do not live in the group orbit of an antisymmetric solution, rotations of antisymmetric orbits can be converged as orbits without symmetry. Therefore, the philosophy now is to allow possible symmetry operations in order to minimize discontinuities at the gluing boundaries. If so desired, then any discrete symmetry selection rules may be applied afterwards. Some final recommendations for gluing: gluing orbits together which have drastically different (transverse) dimensions is not recommended, even for pairwise gluing. For example, gluing an $L = 44$ and $L = 22$ orbit, *with respect to time*; as this would set $\bar{L} = 33$. Second, even when using orbits of approximately the same size, large gluing configurations are not recommended. Instead, the recommendation is to slowly build up the tiling by alternating between tiling and converging; producing progressively better local shadowings with which to glue.

A technique even more expensive than the group orbit search would include exploration of each orbits' continuous family; a detail to be discussed in detail in chapter 4. This is typically not a tractable solution unless the size of the configuration is sufficiently small. In actuality, it is not even recommended for pair-wise gluing, as each continuation would entail multiple optimization runs, making the gluing process more complex than the original goal of finding the glued orbit. However, if truly desired, the following is recommended: first perform continuation to generate a sampling of each continuous family and then query the saved members during the gluing process.

In summary, orbits can be found by applying numerical optimization methods sect. 3.1 to initial guesses created via the methods described in sect. 3.2. After orbits have been found, the original spatiotemporal techniques, clipping sect. 3.3 and gluing sect. 3.5, as well as numerical continuation sect. 3.4 can be applied. All three of these methods are used in conjunction with the numerical optimization methods; inspiring more improvements to be made in the future in chapter 6.

# CHAPTER IV

# NUMERICAL EXPERIMENTS

The goal of this next chapter is to demonstrate numerical results; the orbithunter codes were such a large endeavor that much of the actual physics is still under investigation. Therefore, while there are many unanswered questions, the following should be sufficient to display the power of the spatiotemporal methods developed in the previous chapter.

## *4.1 Preliminary testing*

Before we began searching for orbits in full, we first tested the efficacy of the spatiotemporal numerical methods using known periodic orbits [25]. The first such test was simply to determine that coarse discretizations could be used to accurately represent orbits of various dimensions. The predecessor to our work [75] states that they required upwards of 512 and 1024 points to discretize the temporal dimension; with our methods, orbits with comparable periods require far fewer points, presumably from the accuracy of the pseudospectral method. This is demonstrated in figure 17 wherein the same orbit is plotted (without interpolation, contrary to most figures). This is important because the main limiting factor for spatiotemporal methods is the number of computational degrees of freedom; the entire orbit must be kept in the computational memory. It is imperative that we be able to find periodic orbits with coarse resolutions, otherwise the problem is not computationally feasible, at least without appealing to more complex computational resources such as cluster based computing.

The first test demonstrated here simply rediscretizes a known solution to create an coarse initial guess and then the optimization algorithms are applied, determining whether or not the solution would converge to the 'same' orbit that it originated from. This rediscretization test is shown in figure 17, the main takeaway being that coarse discretizations, much coarser than the original, can be used to represent orbits. In fact, orbits can be 'compressed' in an informational sense by applying discretization continuation to find the minimal collocation

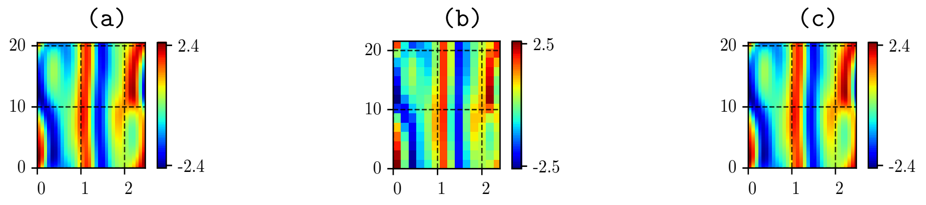

**Figure 17:** (a) Orbit with discretization $(N, M) = (128, 32)$, (b) the same orbit converged with $(N, M) = (14, 14)$, (c) the orbit resulting from increasing (b)'s discretization back to $(N, M) = (128, 32)$.

grid size which converges to the 'same' orbit. The usage of 'same' here is because the tile dimensions are changing and so typically not only will we find a different member of the continuous family but in addition, it will be a different group orbit member. The second of the preliminary numerical tests utilized the same orbit from figure 17, this time around, a large amount of random noise was added to the physical field. To take the worst case scenario, there was absolutely no pre-processing applied to the noise; the field in figure 18 simply consists of the sum of the original field and values numerically drawn from a random normal distribution. Additionally, the orbit was converged as an orbit without symmetry; even though it started in the shift-reflection subspace. Therefore, the noise provided contributions to the now re-included vanishing modes. By virtue of the robustness of the hybrid numerical method, the noisy construction converged to the same family of orbits figure 18, within the shift-reflection subspace. This seems to indicate that the variational nature of the problem is not only able to converge a poor initial condition, but also that there seems to be some memory of the original solution due to topology. Note that preconditioning chapter A is utilized in the first stage of the optimization process, meaning that the changes to tile size are in fact damped during the adjoint descent stage; this might provide enough of a constraint to 'remember' the original orbit; a comment included for transparency. One aspect of the numerical optimization which has not been tested thoroughly is the best way to scale $N, M$ as functions of $T, L$. The relationships (118) and (117) are suggested, however, due to the nature of the differential algebraic equations, $N$ and $M$ are not really independent of one another, contrary to what the guidelines suggest. In any case, the results displayed in figure 18 and figure 17 gave us

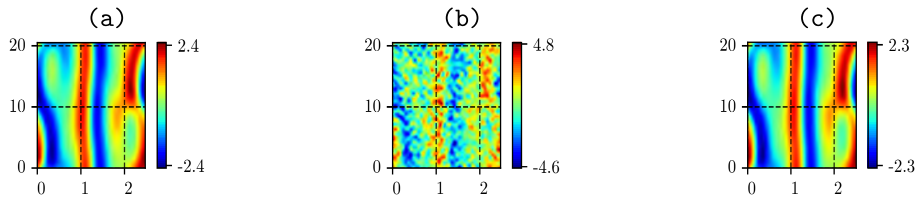

**Figure 18:** (a) The original orbit, (b) with additive noise (note the scale), (c) the converged noisy guess.

confidence that the numerical methods were robust and efficient enough to not only find orbits from random initial conditions; but also confidence that the clipping sect. 3.3 and gluing sect. 3.5 techniques would work, which we shall now demonstrate.

## *4.2 Hunting results*

The successful application of the optimization algorithms sect. 3.1 results in a collection of orbits of varied tile dimensions, symmetries, but most importantly; spatiotemporal patterns. In figure 19 and figure 20, the initial conditions from which the orbits were converged are also displayed to give the reader a sense of the capabilities of the optimization algorithms. The end goal is to search for the smallest and hopefully most frequently shadowed orbits; the fundamental orbits of the Kuramoto-Sivashinsky equation, and so we limit the size of the initial guesses created in this manner by placing a heuristic upper bound on the tile dimensions $(T, L)$, as described in sect. 3.2. The hypothesis is that if all fundamental orbits are captured then the infinite spacetime can be described by combinations of the set of fundamental orbits, as opposed to an infinite collection of orbits of increasing size.

The collection process continued until we were satisfied with our collection, determined by whether or not we thought all fundamental patterns were accounted for. This numbers around several thousand orbits with tile dimensions of intermediate size. The determination that most fundamental orbits were captured was a completely qualitative endeavor; the search for orbits continued simply until we stopped seeing ‘new’ spatiotemporal patterns. As a final example of their capabilities, we demonstrate the optimization techniques applied to modulated initial conditions defined on much larger tiles. Unfortunately, these

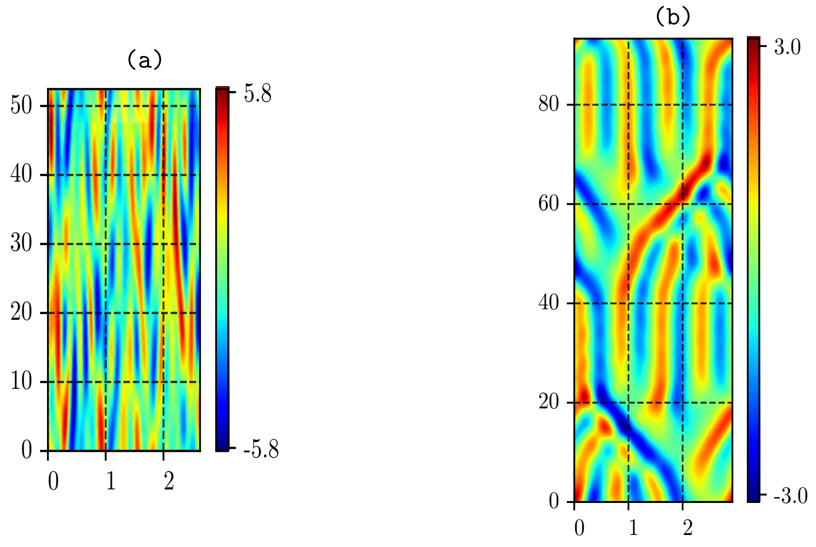

**Figure 19:** (a) Shift-reflection guess orbit, (b) converged orbit

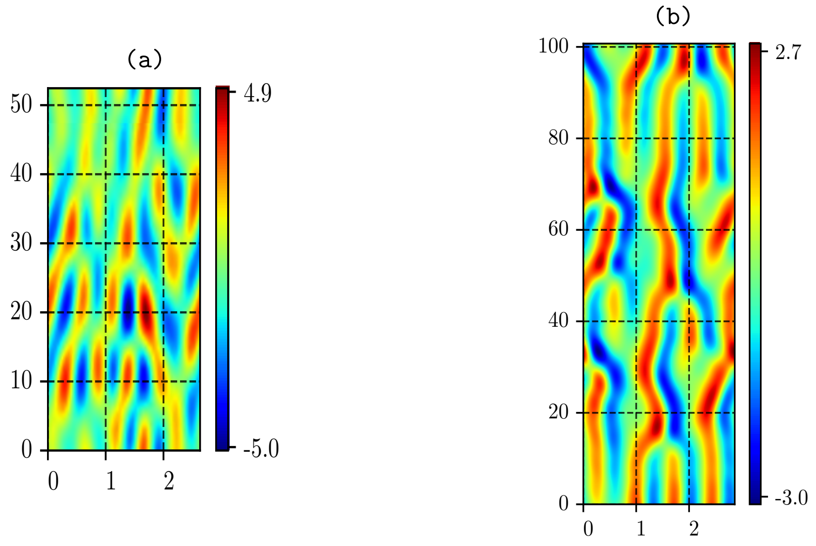

**Figure 20:** (a) Relative periodic guess orbit, (b) converged orbit.

results do not have very small residual values, and so they should not be hailed as complete victories. In spite of this, the point is to demonstrate that it is at least possible to perform optimization on large spatiotemporal guesses, constructed via modulation strategies sect. 3.2. The resulting orbit approximations at least exhibiting familiar structures (in actuality, fundamental orbits). Not every initial condition performs equally well, of course; this is demonstrated in figures 21 and 22 and figures 23 and 24 wherein two different modulation strategies are applied to the same modes. After adjoint descent, the initial tile figures 21 and 23 with dimensions $(T, L) \approx (266.7, 33.7(2\pi\sqrt{2}))$; figure 22 results in an orbit approximation with $\mathcal{O}(1)$ residual and tile dimensions $(T, L) \approx (289.08, 36.4(2\pi\sqrt{2}))$. Likewise, the orbit approximation displayed in figure 24 has $\mathcal{O}(1)$ residual and tile dimensions $(T, L) \approx (299.2, 37.02(2\pi\sqrt{2}))$. These approximations seem to be defined by large regions of relatively uniform local spatial shift velocity; this is easiest to see in figure 21 where the long, diagonal lines representing drifting wavelengths both in the positive and negative spatial direction. The reason for this is unknown; however, we believe that it indicates that there is a much deeper meaning or relationship between the reflection and continuous spatial shift symmetries.

With a collection of orbits in tow, we could begin looking for fundamental orbits, first deciding upon candidates by visual inspection, i.e. determining the most frequently occurring spatiotemporal patterns, following by application of spatiotemporal clipping and more numerical optimization.

## *4.3 Clipping results*

To demonstrate the flexibility of the clipping method, we first demonstrate how orbits can be found by clipping out of arbitrarily large spatiotemporal trajectories. This is shown in figure 25 and figure 27, where a guess orbit with dimensions $(T, L) \approx (80, 2.02(2\pi\sqrt{2}))$ is clipped from a large spatiotemporal trajectory, without special care or planning other than a quick visual inspection. This guess orbit is then sent to optimization, converging to an orbit with dimensions $(T, L) \approx (113.2, 2.1(2\pi\sqrt{2}))$, demonstrating that clippings and initial conditions can be procured from time integration, if so desired; also demonstrating the

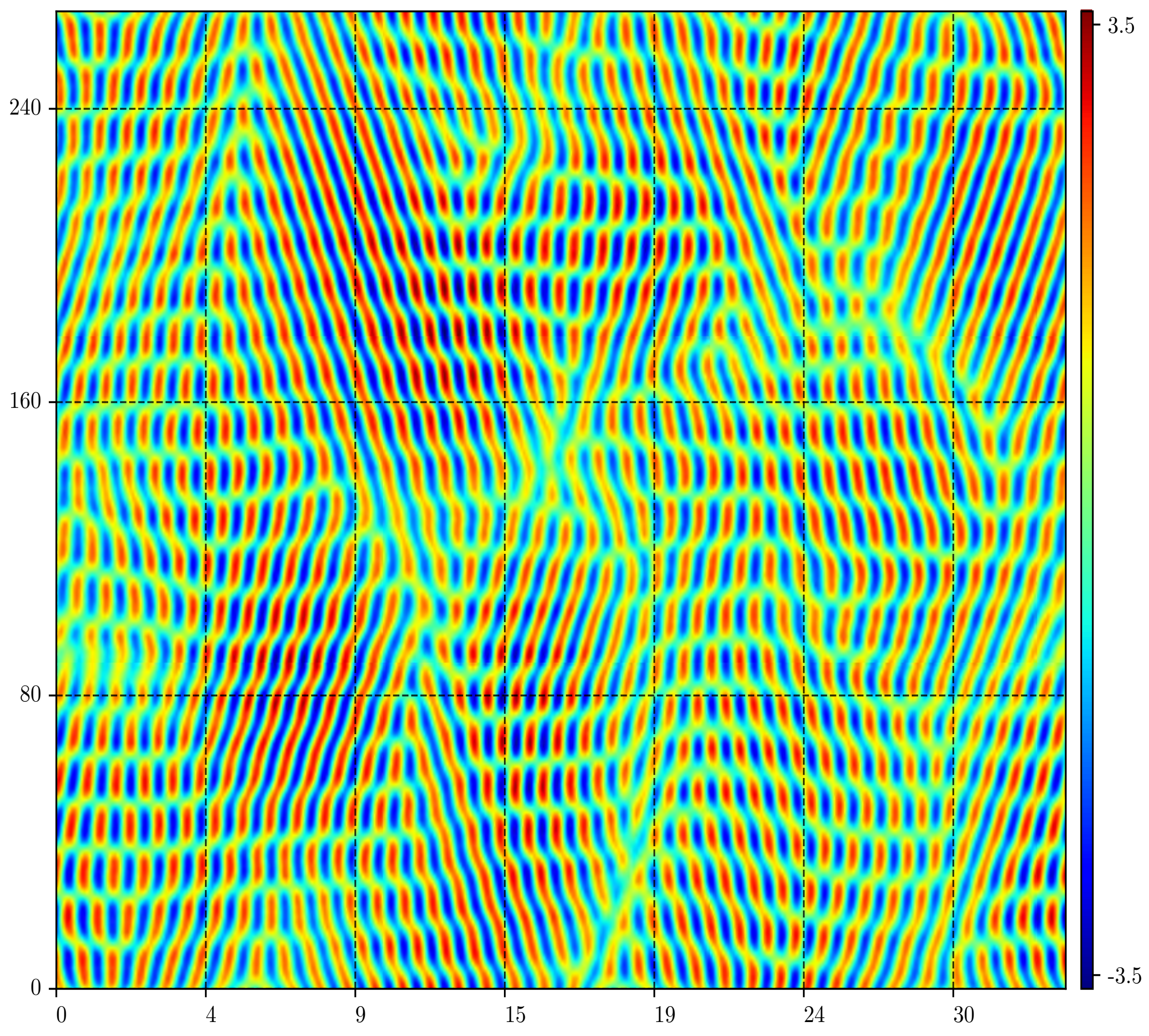

**Figure 21:** Large guess orbit generated with Gaussian modulation.

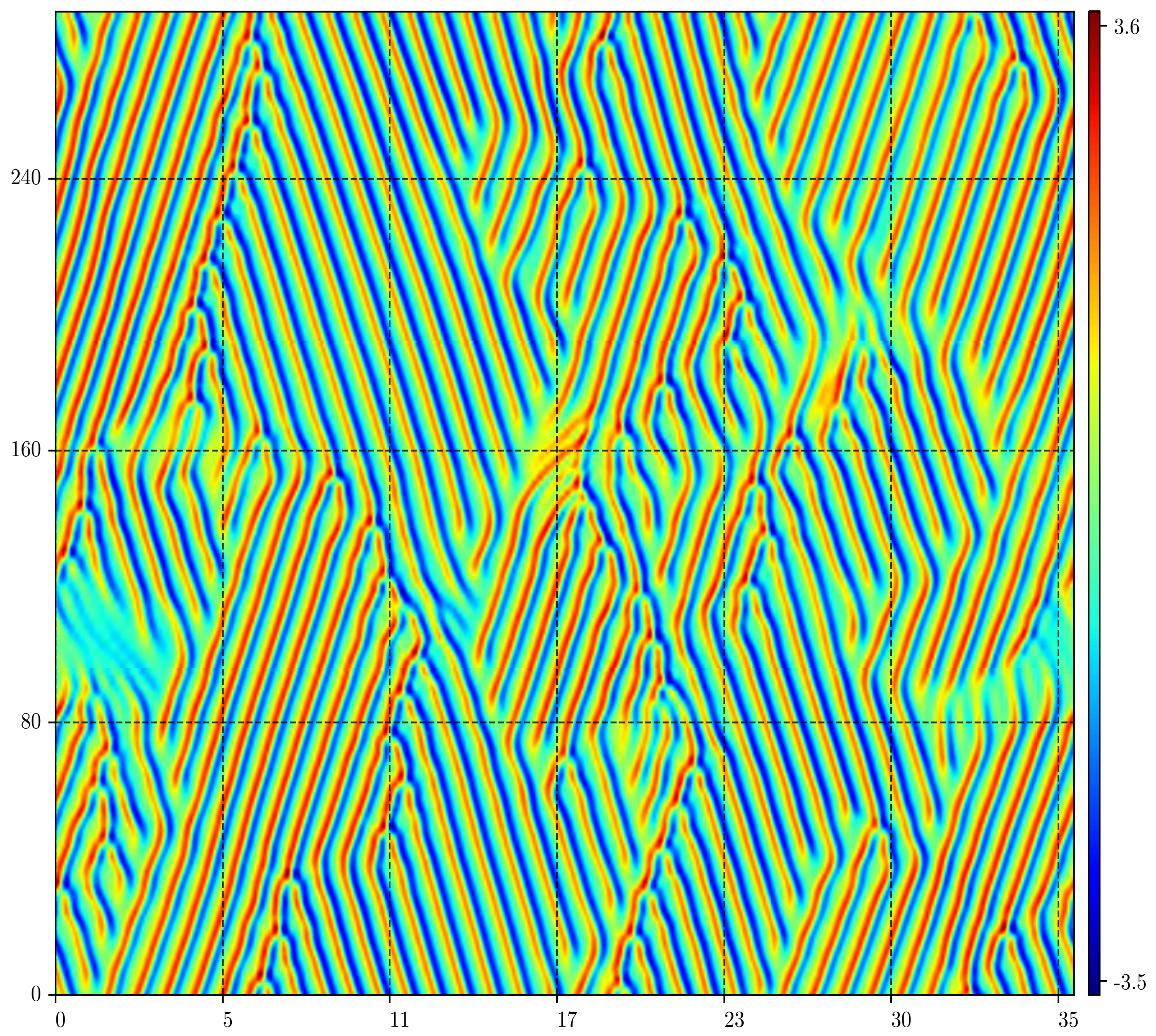

**Figure 22:** Result of applying adjoint descent to guess orbit created with Gaussian modulation.

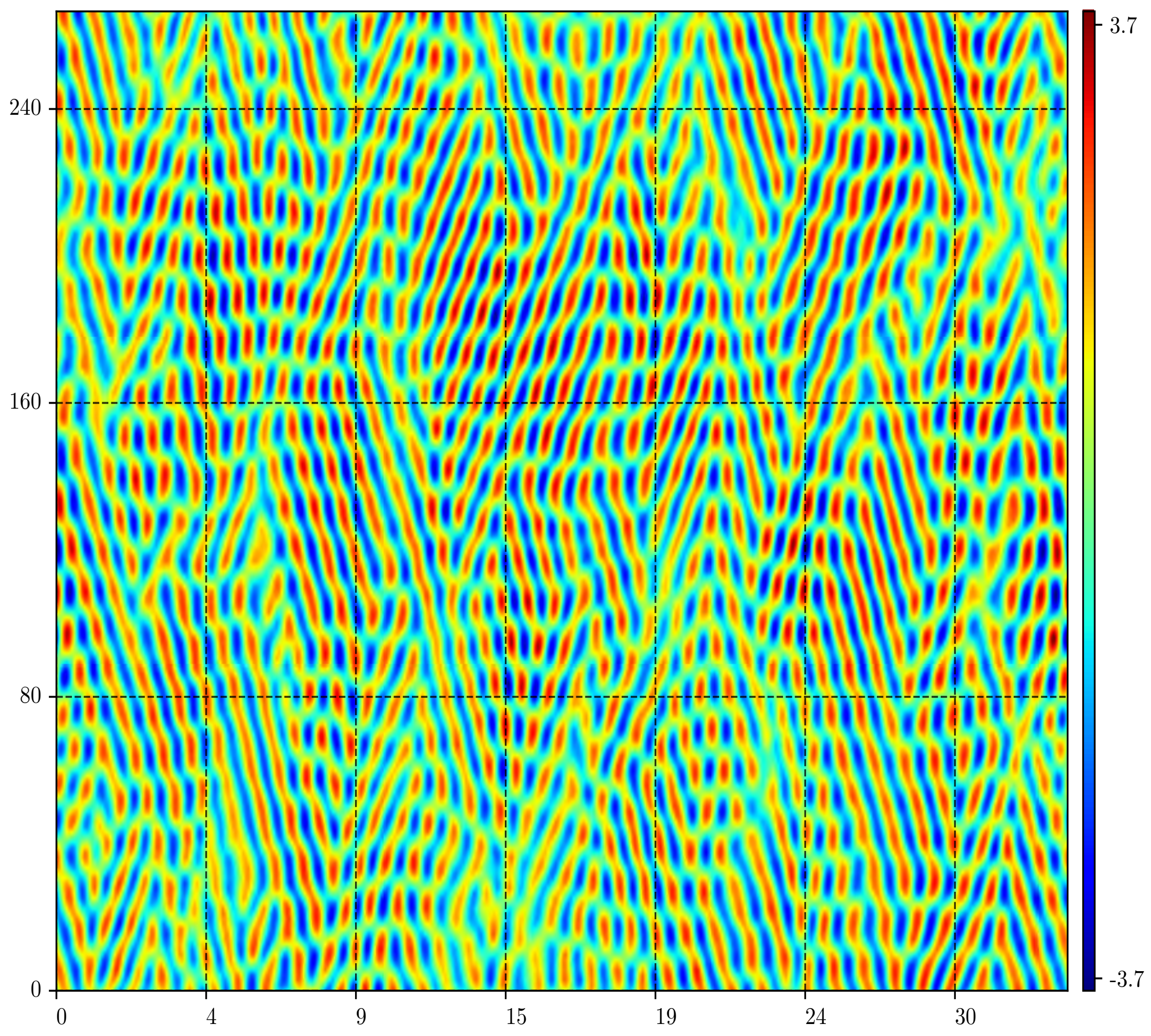

**Figure 23:** Large guess orbit generated with GTES modulation.

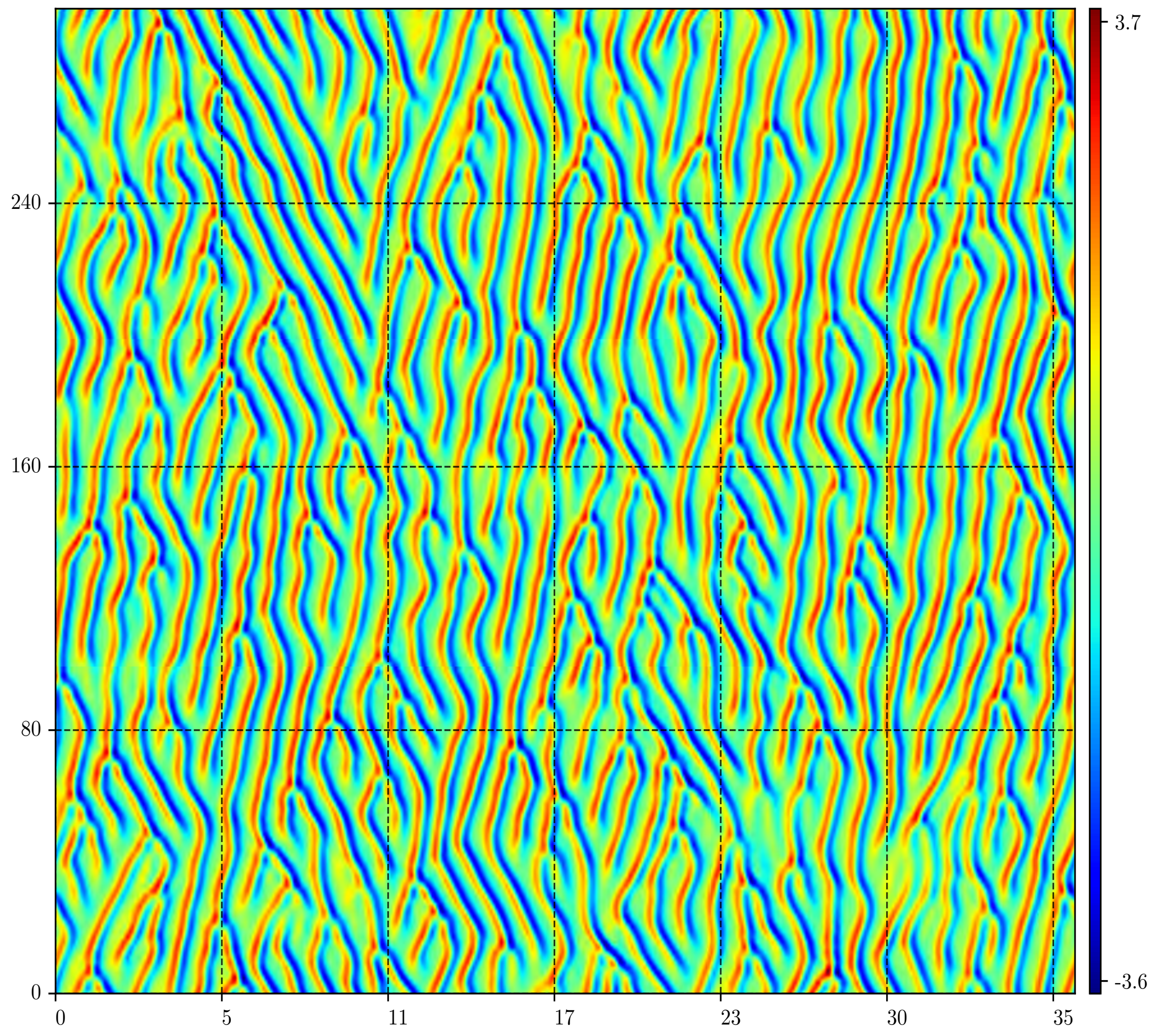

**Figure 24:** Result of applying adjoint descent to guess orbit created with GTES modulation.

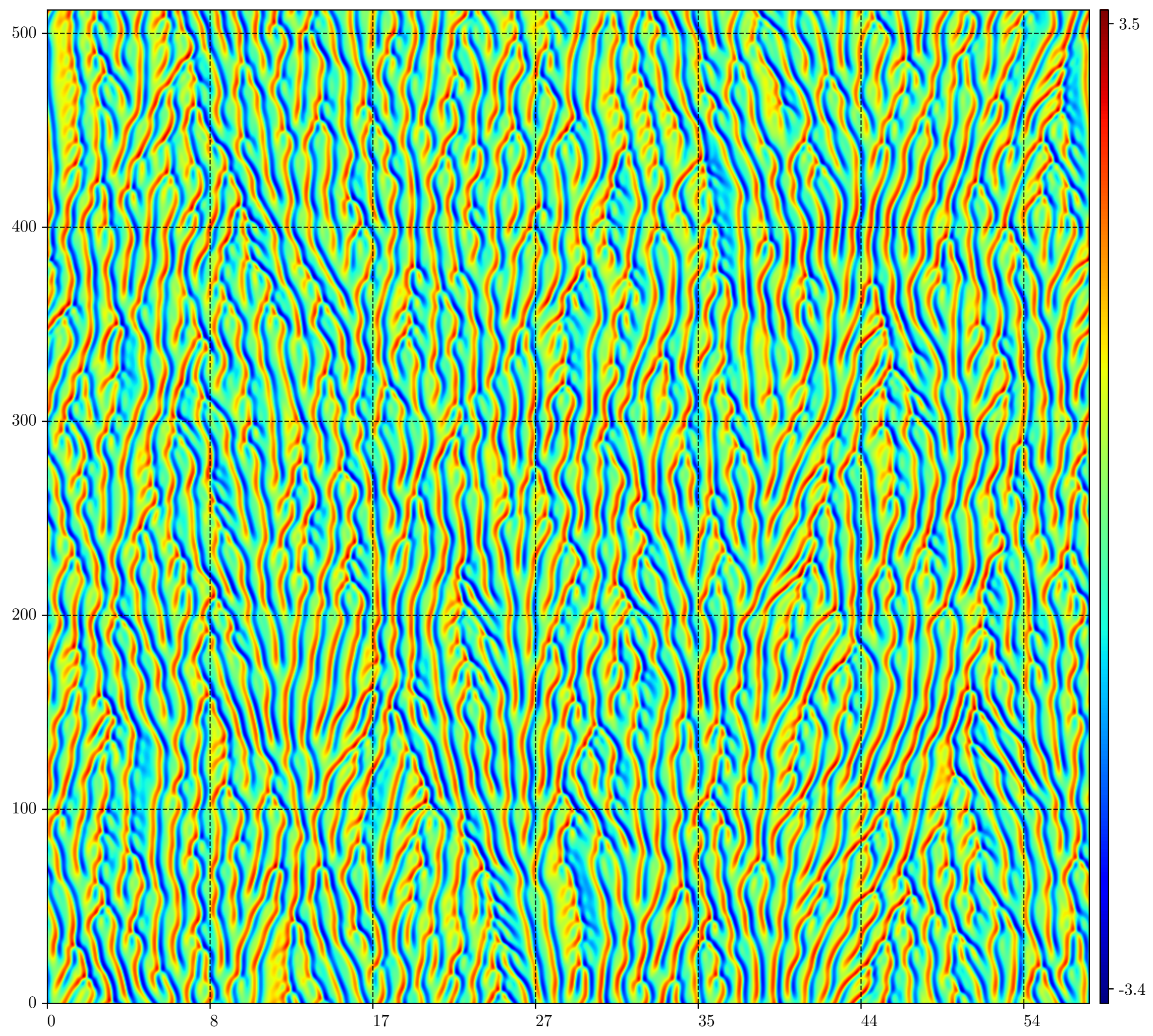

**Figure 25:** Large spatiotemporal trajectory with dimensions $(T, L) \approx (512, 57.6(2\pi\sqrt{2}))$ generated by time integration; used for clipping.

concept that spacetime consists of shadowing events. We encourage the reader to compare the patterns within the orbit figure 27(b) with the three fundamental orbits in figure 35. The clipping was procured simply by a quick visual inspect of the larger spatiotemporal trajectory. There was no care in chosing where the clipping boundaries were placed. The two different discretization strategies were employed, however; the power-of-two strategy (117) did not converge while (118) did. This preliminary result indicates that clipping can work when the clipping is extracted from large spacetime; we now proceed with the primary goal of clipping: extracting fundamental orbits from orbits. In order to begin the description of spacetime as a collection of fundamental orbit shadowings, we need, of

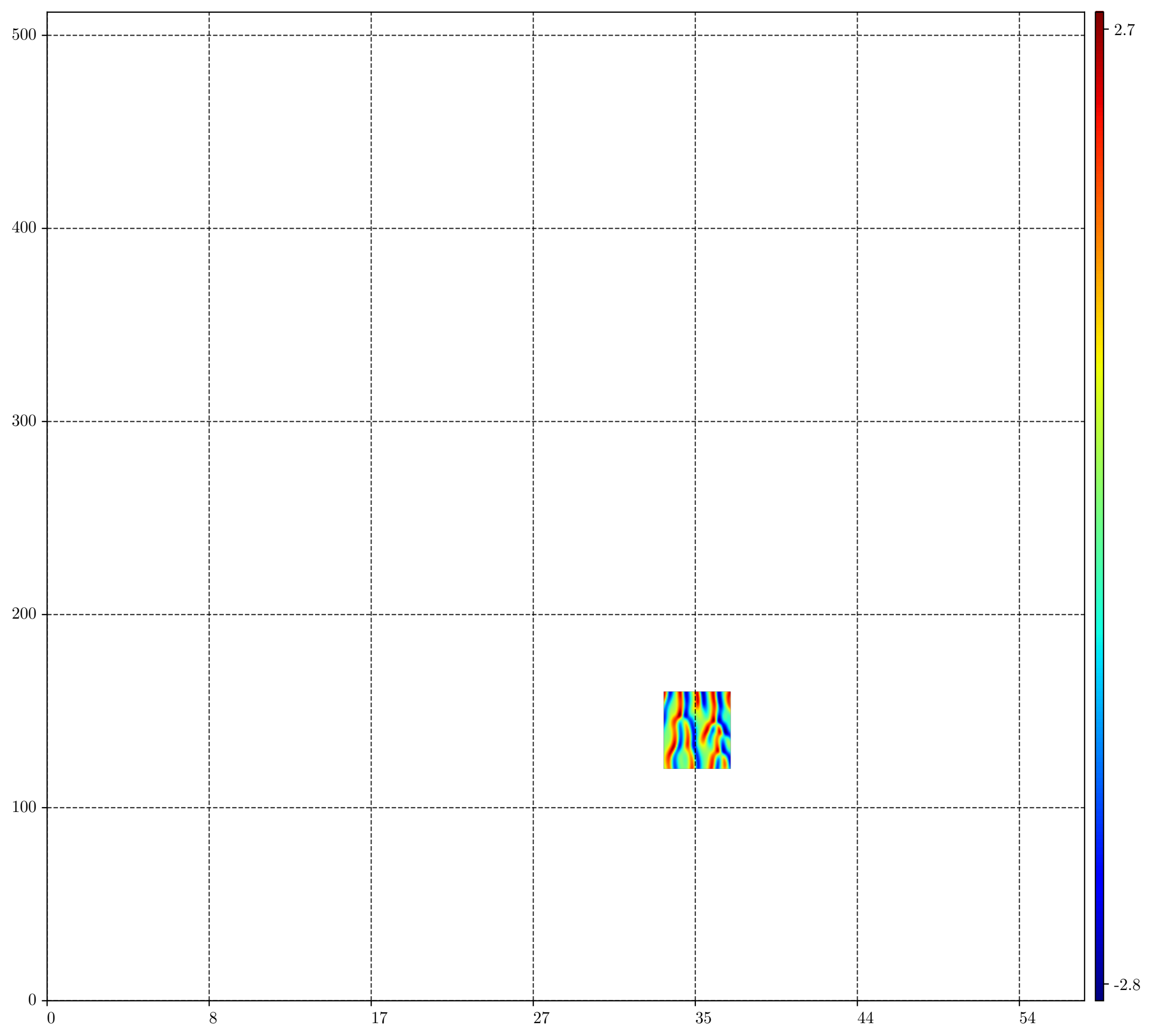

**Figure 26:** Clipping, $(T, L) \approx (40, 3.71(2\pi\sqrt{2}))$.

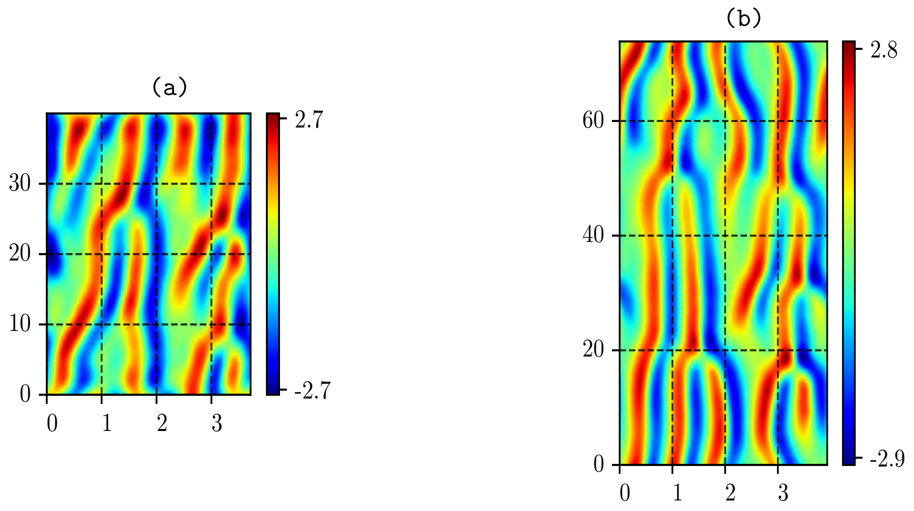

**Figure 27:** (a) Clipping $(T, L) \approx (40, 3.7(2\pi\sqrt{2}))$, (b) converged orbit $(T, L) \approx (73.8, 3.93(2\pi\sqrt{2}))$.

course, a collection of the supposed fundamental orbits. We will refer to them by shorthand names that capture their general behavior and makes a connection to the spatiotemporal pattern they represent. This type of classification is relatively common in the study of fluid flows [63, 90, 136]. Historically speaking, the very first application of clipping was not figure 27, rather, it was the iterative clipping displayed in figure 28. This, very cautious, version of clipping used multiple 'small steps', alternating between clipping and converging. It was unknown how careful the clipping process needed to be, that is, the magnitude of the difference in dimensions of a clipping and its originating orbit. The clipping proceeded according to the alphabetical order of figure 28, as the orbits' dimensions suggest. This turned out to be much too conservative, and in our experience, clipping is typically more flexible than gluing. For example, it is shown in figure 29 that the result of the iterative clipping can be reproduced with only a single clipping step.

The result shown in figure 29(d) was the first ever fundamental orbit guess, tentatively dubbed the 'wiggle' or 'gap' fundamental orbit. It was identified by noticing that it repeats twice within figure 28(a)-(b). The spatiotemporal wiggle can be converged in the antisymmetric subspace, however for the purpose of its description we find it beneficial to ignore its symmetry because the two-wavelength pattern appears more frequently than the antisymmetric fundamental domain. The spatiotemporal wiggle orbit consists of two bent

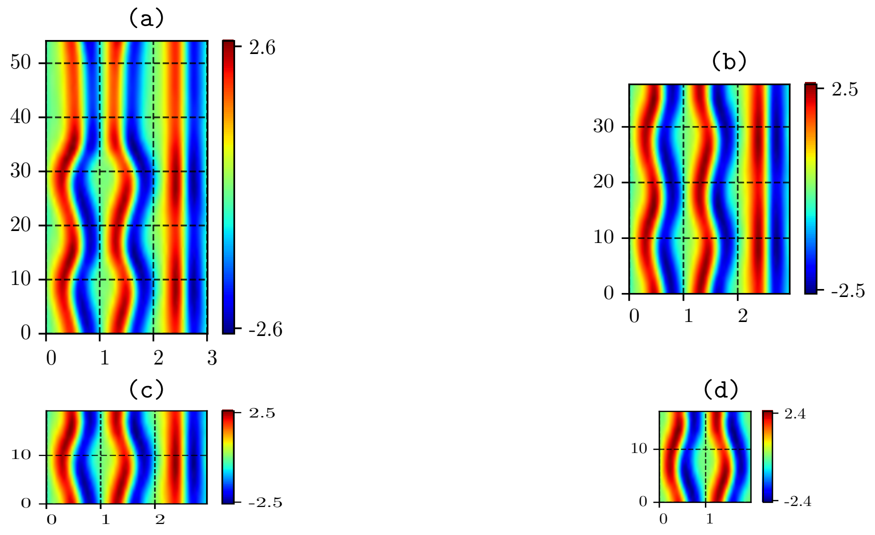

**Figure 28:** Example of iterative clipping. (a) Original orbit, (b) first converged clipping, (c) second converged clipping, (d) converged fundamental orbit.

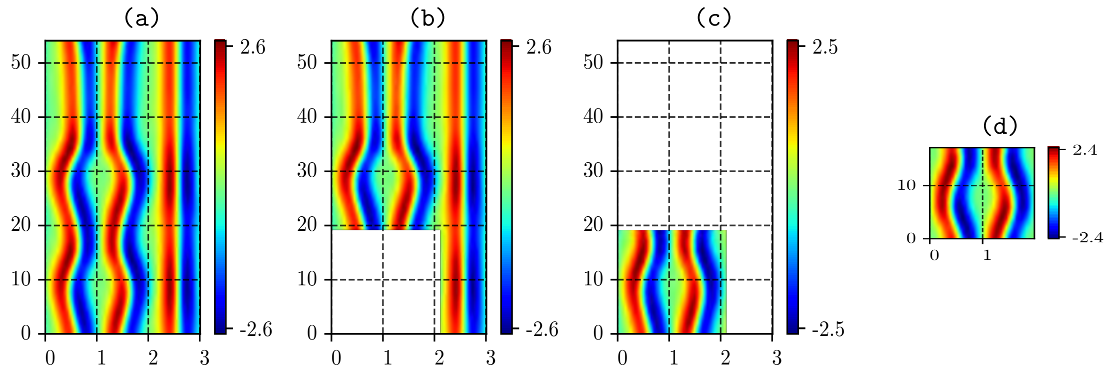

**Figure 29:** (a) Original orbit, (b) clipping exterior, (c) clipping interior, (d) converged clipping.

or 'wiggling' wavelengths occurring in an (exactly) antisymmetric manner. In the region of spacetime between the wiggles the magnitude of the spatiotemporal velocity field is very small, essentially near zero. The appearance of this small gap between the wiggles is explored via continuation sect. 4.4. The collection of the spatiotemporal wiggle not only provided us with our first fundamental orbit but also provided us with an obvious guess for the second fundamental orbit. If we look back at the iterative clipping procedure that we just performed, in figure 28(c), we see that the spatiotemporal wiggle is spatially adjacent to an additional, single wavelength. Therefore it follows that our second fundamental orbit candidate was the single wavelength equilibrium, as there is no other way to explain

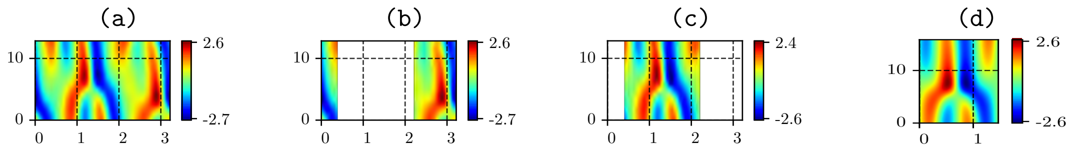

**Figure 30:** (a) Original orbit, (b) clipping exterior, (c) clipping interior, (d) converged clipping.

the spatiotemporal configuration of figure 28(c). Indeed, by clipping this single wavelength equilibrium from the original periodic orbit another fundamental orbit was found, which we now refer to as the spatiotemporal streak.

After collection of the spatiotemporal wiggle and spatiotemporal streak, the fundamental orbit collection was yet incomplete as the pattern emphasized in the introduction in figure 1 had yet to be found. In the spirit of topological defects [34, 70, 87] we named this fundamental orbit the spatiotemporal defect; it was found via the clipping demonstrated in figure 30. The spatiotemporal defect epitomizes two very important mechanisms of the Kuramoto-Sivashinsky equation: fluctuations in global wavelength count and local spatial drift velocity. Both of these processes can be observed in any sufficiently large spatiotemporal simulation or orbit. The effect of the wavelength merger is that it decreases the global wavelength count by one; however, due to linear instability, the region of spacetime evacuated by the merger is soon filled by another wavelength; the net result is approximately a quarter-cell (or equivalently, half wavelength) spatial drift. Therefore, the defect also describes local spatial shift velocity; the magnitude of which is determined by the spatiotemporal defect family member being shadowed; this is explored in sect. 4.4. Neither the spatiotemporal streak nor the spatiotemporal wiggle can account for this spatial shift behavior, as they both exist in the antisymmetric subspace. These two properties highlight the importance of the spatiotemporal defect. Due to the spatial shift it is clear that the spatiotemporal defect was assumed to be a relative periodic orbit. Locating the spatiotemporal defect proved to be much more difficult than both the spatiotemporal wiggle and spatiotemporal streak, in part due to the extra computational degree of freedom from the spatial shift, and its sensitivity to where the temporal clipping boundaries are placed; an

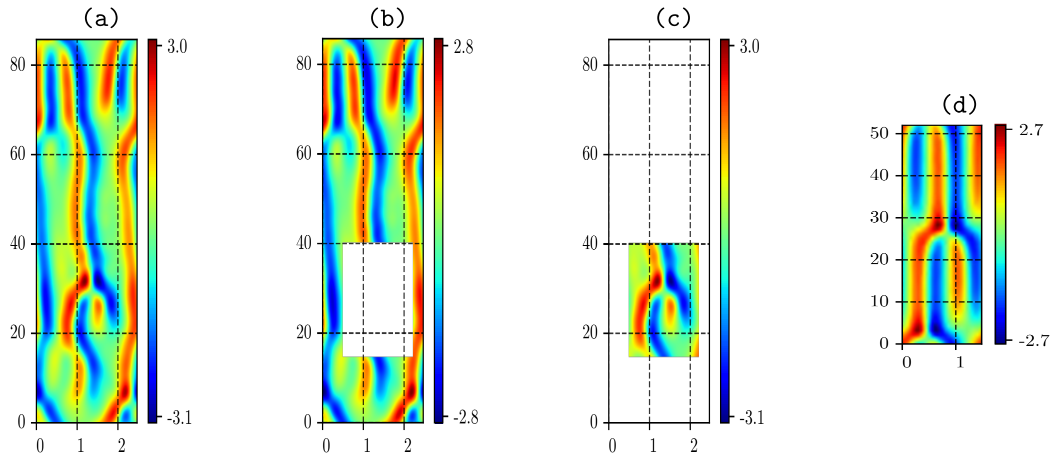

**Figure 31:** (a) Original orbit, (b) clipping exterior, (c) clipping interior, (d) converged clipping.

issue that should be improved in the future. With the collection of the spatiotemporal defect, the three fundamental orbits used in gluing had been found, their final versions displayed in figure 35. At the time, however, the hunt for fundamental orbits continued even after the collection of the spatiotemporal defect, as other fundamental orbits were still thought to exist. The results were somewhat surprising; nearly every guess converged to the spatiotemporal streak, spatiotemporal wiggle or spatiotemporal defect(or repeats thereof). Examples of redundant clippings are shown in figures 31 and 33, which both converge to two repeats of the spatiotemporal defect. The third example, seen figure 32, is not at all obvious; until numerical continuation is performed, that is. A collection of these repeats is displayed in figure 34, where it can be seen that orbits with the same pattern (or multiples thereof) were found.

In summary, there seem to be three main fundamental orbits, originally found to exist on tiles with approximate dimensions $(0, \approx 0.7)$ for the spatiotemporal streak, $\approx (15.85, 1.5)$ for the spatiotemporal defect and lastly $\approx (17.15, 2)$ for the spatiotemporal wiggle. As the error in gluing is highly dependent on the tile dimensions, it would be highly beneficial to have fundamental orbits which exist on identically sized tiles; especially with respect to the spatial dimension, due to the higher order derivatives and the error induced by incorrect approximation of $L$. This, and the existence of continuous families of fundamental orbits are investigated in the next section, which describes the results from numerical continuation.

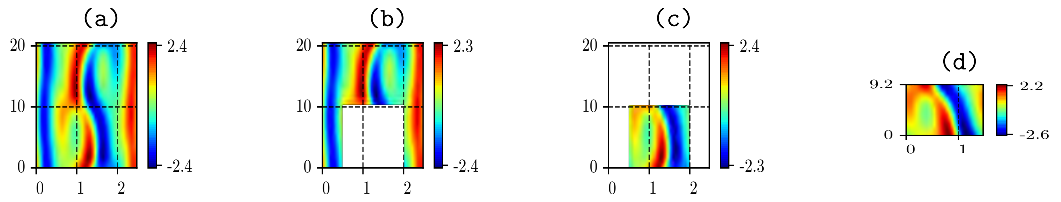
Figure 32: (a) Original orbit, (b) clipping exterior, (c) clipping interior, (d) converged clipping.

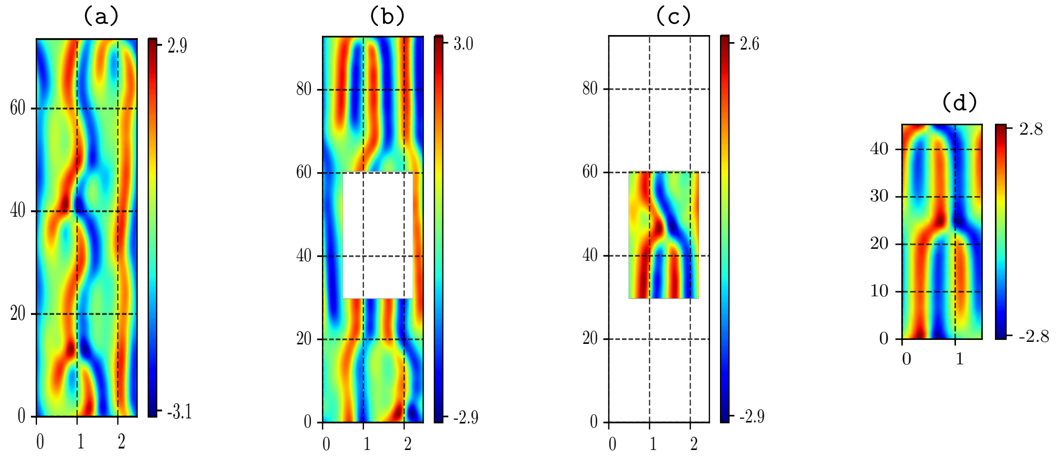
Figure 33: (a) Original orbit, (b) clipping exterior, (c) clipping interior, (d) converged clipping.

## *4.4 Continuation results*

After our first search for fundamental orbits concluded, we actually believed that there were *four* unique fundamental orbits, contradicting what was previously stated. Upon further review two of these four fundamental orbits, displayed in figure 30 and figure 32 looked quite similar visually, leading us to believe that they were related. In figure 38, this is shown by numerical continuation; namely figure 38(c) is the continuation of figure 38(b) to the exact spatial domain size that figure 32(d) exists on; they are likely not the same exact orbit numerically, but it is entirely believable they exist within the same group orbit, judging based on the spatiotemporal patterns.

This brought about a revelation regarding fundamental orbits that had been previously overlooked; fundamental orbits exist in continuous families. This has significant meaning in the context of shadowing. Two shadowings of the same fundamental orbit will look similar

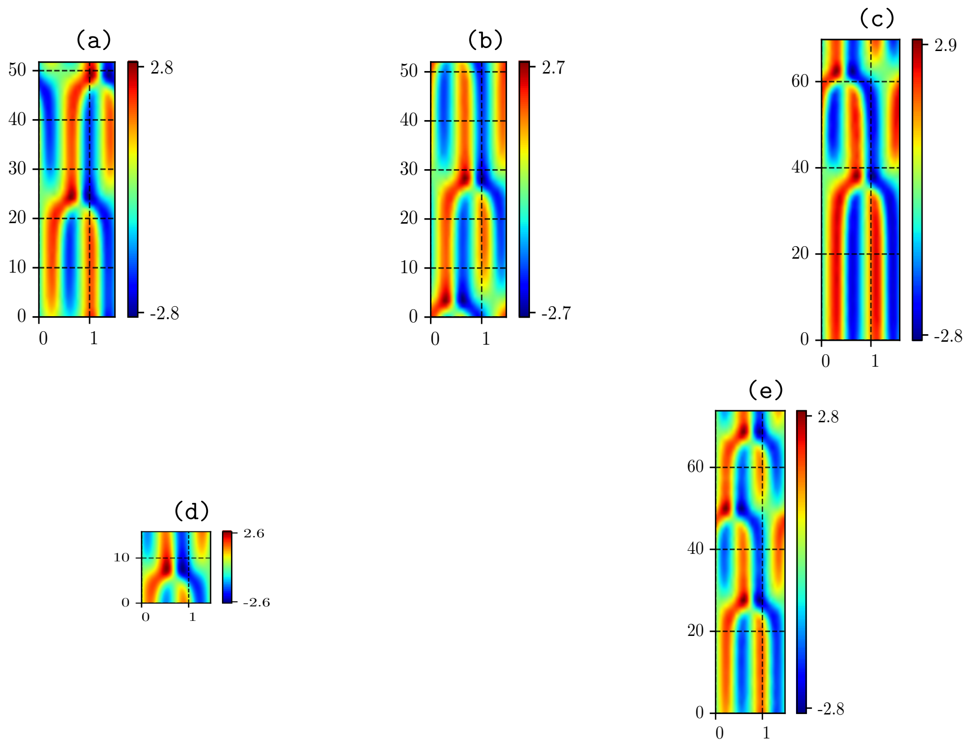

**Figure 34:** Repetitive results from the hunt for fundamental orbits.

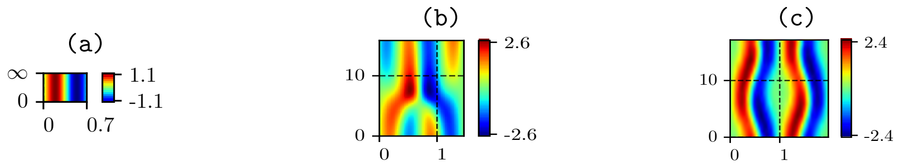

**Figure 35:** The (a) ‘streak’, (b) ‘defect’, (c) ‘wiggle’ fundamental orbits.

but not identical in spacetime; differences which can now be quantified by the approximate dimensions of the shadowing region. This property is precisely captured by continuous families of fundamental orbits, wherein all members of the family are related by continuous deformations of both tile and field.

Application of numerical continuation to the three fundamental orbits in figure 35 are shown in figure 36, figure 37, and figure 38. Each of these figures displays three or four members of the continuous family, such that the spatial dimension size increases from left to right. With fundamental orbits existing in families, and all admissible orbits existing as configurations of fundamental orbits, it might be accurate to say that all orbits exist in continuous families because fundamental orbits exist in continuous families. It of course is not known if *every* orbit exists in a continuous family, however, preliminary testing indicates that this is the case, although the families might be ‘small’, relative to the intervals of tile sizes that they exist on. This initially betrayed our intuition; in hyperbolic systems, periodic orbits are isolated solutions by virtue of their unstable manifolds [24]. The meaning of this and its effect on the symbolic dynamical grammar is currently unexplored, however, it has dramatic implications in regards to the symbolic dynamics. Namely, it means that the alphabet of this symbolic dynamics is not a rigid collection of symbols; each symbol, that is, each fundamental orbit, is parameterized by its tile sizes. The implication is that the symbolic alphabet is ‘rubbery’ which provides great boons, and banes. As the symbolic dynamics has not been formalized yet, this is left to the discussion of future work in chapter 6.

Each continuous family seems to exist on a finite interval of spatial periods, punctuated by what are presumed to be bifurcations. The problem with such a description is that the normal bifurcation analysis via linear stability is unavailable; without dynamics, we do not currently know how to provide evidence of bifurcation. Also, it may be the case that there has simply been a failure to track the family correctly. Therefore, the descriptions which follow cover only the qualitative behaviors of the orbits believed to be members of the same family. The spatiotemporal defect family is qualitatively described by three members displayed in figure 38. In its large space limit, it changes into a relative equilibrium approximately at $L \approx 1.5(2\pi\sqrt{2})$. In the limit of small spatial dimension, the temporal period can be stretched to ridiculous amounts $T$ in the hundreds, shadowing what is believe to be a heteroclinic connection.

In the limit of small spatial size, continuation of the spatiotemporal wiggle, figure 37 finds the two-wavelength equilibrium solution; a member of the spatiotemporal streak family. The implications of this are to be discussed shortly hereafter. In the limit of large spatial dimension, the 'stretching' which occurred in the defect family seems to occur again. Currently there is no explanation of why this occurs and what these orbits even represent. Finally, the spatiotemporal streak family, displayed in figure 36, simply represents the $n$-cell solutions of [51].

As described in the description of the spatiotemporal gluing method, gluing tiles of disparate dimension introduces substantial amounts of error. It is beneficial, therefore, to use numerical continuation to resize each fundamental orbit until they are of similar size. Unfortunately it seems impossible to resize the fundamental orbits while also maintaining their unique properties. To demonstrate this, the spatiotemporal wiggle and spatiotemporal streak were numerically continued to the same spatial dimension as the spatiotemporal defect, as it originally had the median spatial dimension prior to continuation.

The continuations of the streak and wiggle, show in figure 36 (b) and figure 37(a), respectively, are nearly identical, after a quarter cell shift of either orbit; i.e. they are within the same group orbit. This can be approximately shown by taking the the $L_2$ norm of the difference of the discretized fields; $|u_s - u_w| \approx 10^{-5}$, the non-zero magnitude accredited to interpolation error. The spatiotemporal wiggle's temporal derivative at this spatial domain size has a norm on the order of machine precision, $|u_t| \approx 10^{-14}$; meaning that we have converged to an equilibrium. Therefore, if we are to maintain a trinary symbolic alphabet, the spatiotemporal wiggle cannot be continued to the same domain size as the spatiotemporal defect, as it no longer is distinct from the spatiotemporal streak. It should be noted that the numerical continuation of the wiggle seems to be irreversible, if the equilibrium solution from the spatiotemporal wiggle family figure 37 has its spatial domain size increased, it does not return to the spatiotemporal wiggle family rather, it follows the

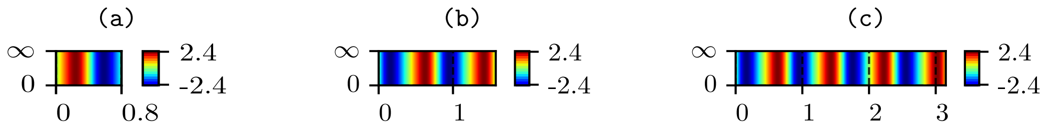

**Figure 36:** Members of the streak fundamental orbit family.

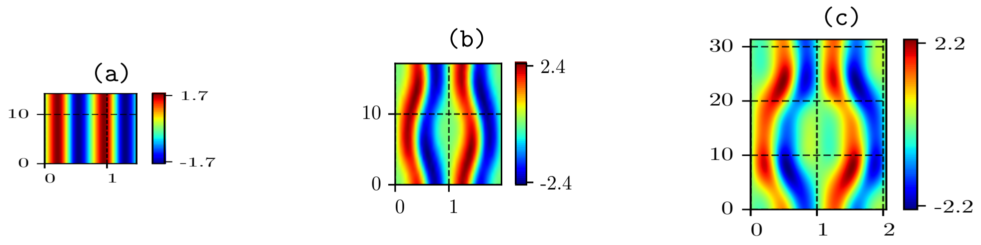

**Figure 37:** Members of the 'gap/wiggle' fundamental orbit family.

same pattern as the spatiotemporal streak fundamental orbit, its continuation producing the $n$-cell equilibrium solution.

The fact that continuation of the spatiotemporal wiggle finds the two wavelength equilibrium solution raises an interesting question, could the spatiotemporal defect and spatiotemporal wiggle orbit families be recovered by manipulating the single wavelength spatiotemporal streak solution? It turns out that, yes, this is indeed possible and is discussed with the future work in chapter 6, displayed in figure 56. This seemingly throws a wrench into what we think of as being fundamental at first. However, it is not simply that the streak is the only fundamental pattern; rather, it is much more intimately involved with symmetry. Perhaps it could be argued that the spatiotemporal streak family and symmetry breaking represents the set of fundamental periodic orbits; currently this is an important but open question.

Before moving onto gluing, let us quickly summarize what we know about continuous families of our three proposed fundamental orbits. The spatiotemporal formulation is predicated upon the existence of a finite set of fundamental orbits, the set of fundamental orbits being the set of orbits which can ostensibly be used to construct and describe all other orbits solutions. We believe that this collection consists of three unique fundamental orbit continuous families. Each of these families was explored via continuation in $L$, resulting in a collection of family members on at least some finite interval of $L$. To describe the

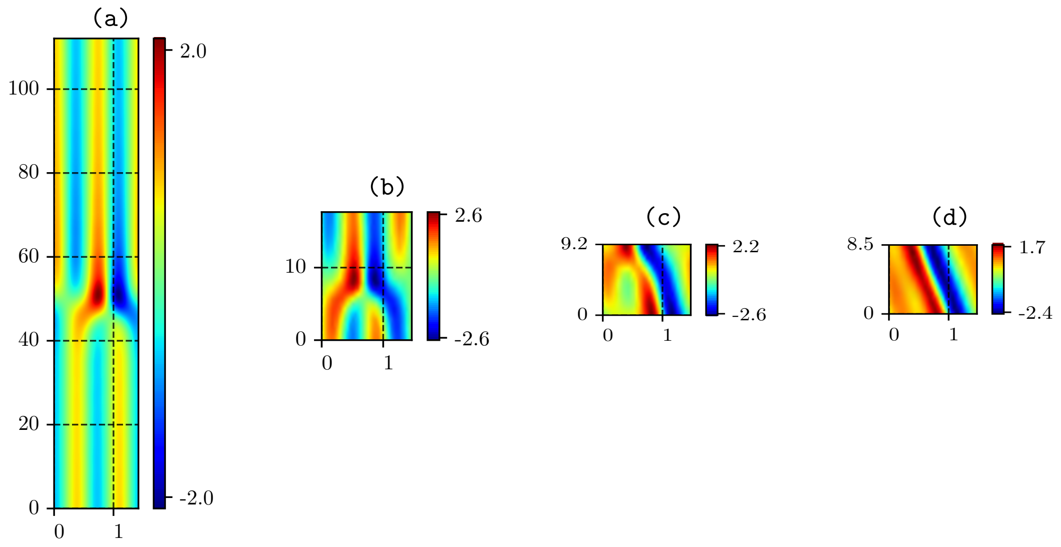

**Figure 38:** Members of the 'defect' fundamental orbit family.

families (other than the streak, because it was relatively boring), we looked at the large and small $L$ limits of the spatiotemporal defect and spatiotemporal wiggle families. Each of these families seems to have some deep connection with symmetry, however, we do not know how to describe this yet. With the existence of these families, we now have a rubbery symbolic alphabet with which to probe the grammar of the purported symbolic dynamics. In other words, even though there are only 'three' fundamental orbits, we can describe all spatiotemporal tile sizes by virtue of the different tile sizes within each family.

Currently, the formulation of the symbolic dynamics has not been completed, however, we can test our ideas by virtue of the gluing method, described in sect. 3.5. That is, by creating spatiotemporal combinations of fundamental orbits, and trying to find the orbits being shadowed thereby. As we shall see, the three families current within our alphabet is indeed sufficient to reproduce known solutions and produce new solutions, providing empirical evidence that we are on the right track.

## *4.5 Gluing results*

The gluing technique developed in sect. 3.5 is tested with progressively harder test cases. First, we apply it to one-dimensional gluing in space and time. Next, small spatiotemporal combinations are glued together both with and without the 'strip-wise' modification. Lastly,

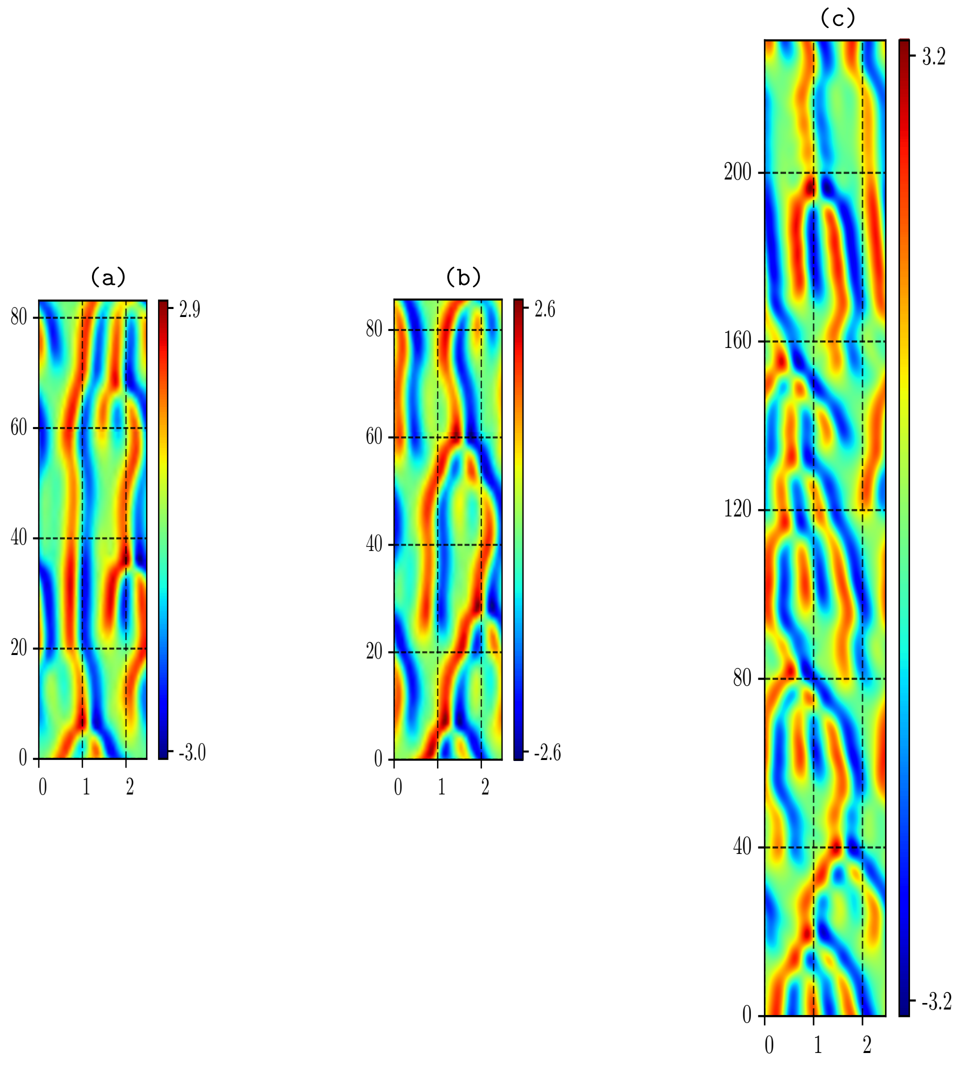

**Figure 39:** (a)-(b) Relative periodic orbits, (c) their temporal gluing.

we attempt to apply these ideas to arbitrarily large spatiotemporal configurations.

The first test case is one dimensional temporal configurations of orbits, the primary example is displayed in figure 39 wherein two relative periodic orbits defined on tiles with approximate dimensions $(T_1, L_1) \approx (83.11, 2.5(2\pi\sqrt{2}))$ and $(T_2, L_2) \approx (85.66, 2.5(2\pi\sqrt{2}))$. The resulting orbit has dimensions $(T, L) = (233.05, 2.5(2\pi\sqrt{2}))$; defined on a collocation grid with dimensions $(N, M) = (68, 32)$ (included for this one example to give a sense of how things scale). Next up is one-dimensional spatial gluing. For one-dimensional spatial gluing, the emphasis is placed on the ability to perform the gluing repeatedly; this opens the door to the possibility of slowly building up orbits spatially; possibly providing

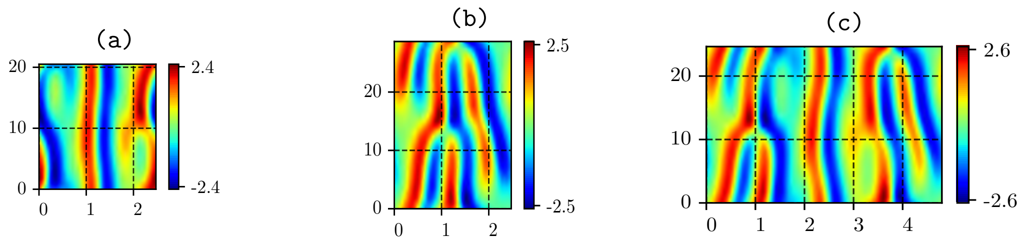

**Figure 40:** (a)-(b) Shift-reflection orbits, (c) their spatial gluing.

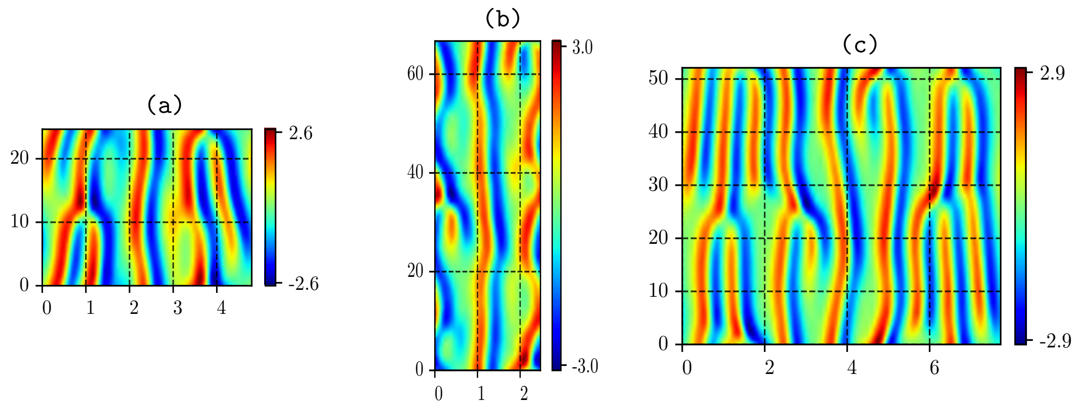

**Figure 41:** (a)-(b) Shift-reflection orbits, (c) their spatial gluing.

access to progressively larger spatial geometries, one of the primary motivations for all of this work. This is shown in two steps; in figure 40, two shift reflection orbits with dimensions $(T_1, L_1) \approx (20.50, 2.5(2\pi\sqrt{2}))$ and $(T_2, L_2) \approx (28.66, 2.5(2\pi\sqrt{2}))$ are spatially glued, resulting in a shift-reflection orbit with dimensions $(T_{12}, L_{12}) \approx (24.65, 4.8(2\pi\sqrt{2}))$. This result is then glued again with another shift-reflection orbit with dimensions $(T_3, L_3) \approx (66.70, 2.5(2\pi\sqrt{2}))$ resulting in yet another shift-reflection orbit figure 41 with dimensions $(T_{123}, L_{123}) \approx (45.75, 7.4(2\pi\sqrt{2}))$. In this specific example, shift-reflection symmetry was maintained in every step; showing how even discrete symmetries can be utilized in the gluing process. Finally, we arrive at the culmination of all of the research and computational codes developed for the spatiotemporal formulation, spatiotemporal gluing. We begin with the triplet of fundamental orbits as our tentative tiling alphabet figure 35. Mentioned at the end of sect. 3.5, arbitrary searches of the continuous families of fundamental orbits is not viable; instead, the usage of different 'tile sets' is being investigated. These sets represent numerical manipulations used to minimize the discontinuities at the boundaries through various means; zero padding, rescaling, usage of different continuous family members, different group orbit members, etc. Examples of these are displayed in figure 43, figure 42, and figure 35; which are the 'resized', 'spatially padded' and 'original' tile sets respectively. The defining quality of the spatially padded tile set is that the spatiotemporal fields are zero-padded with respect to space, as the name implies. In the resized tile-set the streak is numerically continued to a tile with larger spatial dimension, the defect numerically continued to have the same $T$ as the wiggle. Note that these examples only represent a subset of all tile sets used for experimentation. These fundamental orbits are glued together in spatiotemporal configurations, befitting a symbolic description as shown in figure 44. Note that usage of different tile sets could be represented by including the correponding tile size in figure 44. Beginning with a spatiotemporal configuration of the smallest possible spatiotemporal shape $(2, 2)$, we see that in figure 45, a converged orbit is indeed found, however, the defect in the bottom right of the configuration seems to not be present in the final orbit. In figure 46, the components of the inital spatiotemporal configuration can be visually observed; if a liberal interpretation is allowed that is. meaning, that the two streaks in the top left, $t \approx [15, T], x \approx [0, 2(2\pi\sqrt{2})]$ seem to configure themselves into a wiggle. This type of visual inspection points out a huge hole in the construction of the symbolic dynamics. Namely, we may have numerical convergence, however, there is an inability to identify the original spatiotemporal configuration within the resulting orbit. This presents us with a difficult challenge; namely, how do we validate that our gluing results accurately represent the spatiotemporal configuration we started with? Currently we have no good answer for this, however we are exploring some possibilities, described in chapter 6. Regardless of these complications, the results in figure 45 and figure 46 demonstrate that orbits can indeed be found via the gluing of fundamental orbits. So, it may be a stretch to claim that these are 'the' building blocks of spatiotemporal chaos for the Kuramoto-Sivashinsky equation, however, we will claim that these fundamental orbits can be used as building blocks; an exciting development.

It is important to investigate the different gluing options, their efficacy, and their potential. Towards this end, two examples are provided which attempt to stress the effect

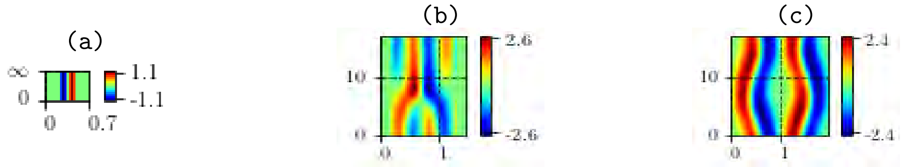

**Figure 42:** The (a) 'streak', (b) 'defect', (c) 'wiggle' fundamental orbits in the 'spatially padded' tile set.

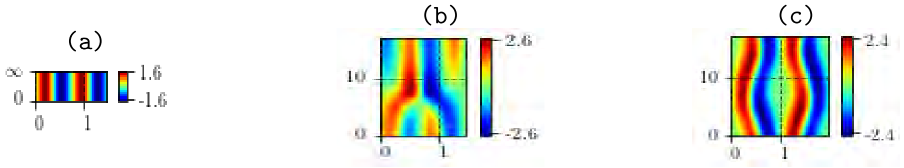

**Figure 43:** The (a) 'streak', (b) 'defect', (c) 'wiggle' fundamental orbits in the 'resized' tile set.

of different tile sets and different gluing strategies. In figure 47 and figure 48 the same spatiotemporal configuration (using the same tile set) is constructed, the difference being whether or not the strip-wise correction was applied. The residuals of the results are both $\mathcal{O}(-1)$, however, the opinion taken here is the strip-wise result in figure 48 appears to be a much better orbit approximation, determined by the visualization, not the residual. While it did perform better, this hints at future issues for the strip-wise gluing technique, namely, as the spatiotemporal configurations become progressively larger, the order in which the dimensions are glued has a larger effect. Additionally, the strip-wise technique breaks down in this limit, as the relative sizes of fundamental orbits between strips becomes distorted by differences in strip dimensions. The next example is similar; however, instead of demonstrating the difference between strip-wise and regular gluing, demonstrates how different tile sets sect. 3.5 can result in wildly different orbit approximations. The initial guesses are displayed in figure 49 and figure 51; the results in figure 50 and figure 52, respectively. The approximations actually have nearly the same $\mathcal{O}(10^{-1})$ residual; however, the result figure 52 seems much less like a 'typical' solution than figure 50.

Lastly, we include an excessively large spatiotemporal gluing, using a 30-by-30 spatiotemporal configuration of different family members of the fundamental orbit families. The idea behind using different members of the fundamental orbit families in order to instill the large spatiotemporal domain with local spatial shift velocity, using figure 23 as

(a)

$$\begin{bmatrix} 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 2 & 0 \\ 0 & 0 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 0 \end{bmatrix}$$

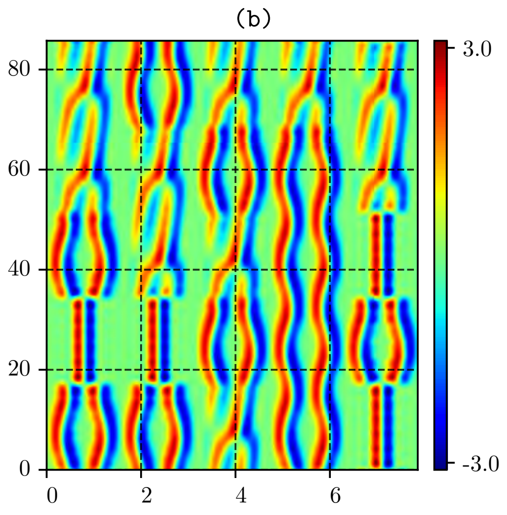

**Figure 44:** (a) A symbolic configuration, (b) a guess orbit manifestation.

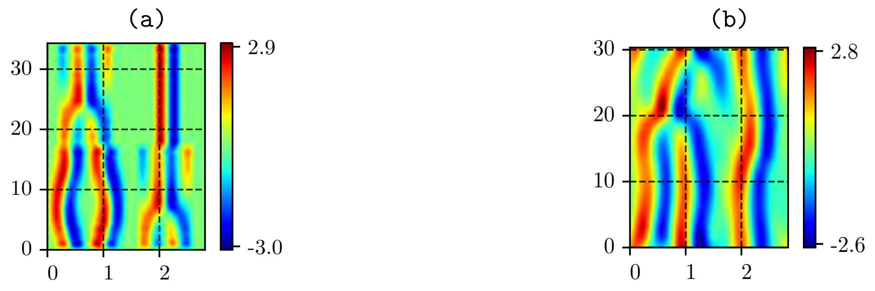

**Figure 45:** (a) Two-by-two spatiotemporal configuration (b) converged orbit.

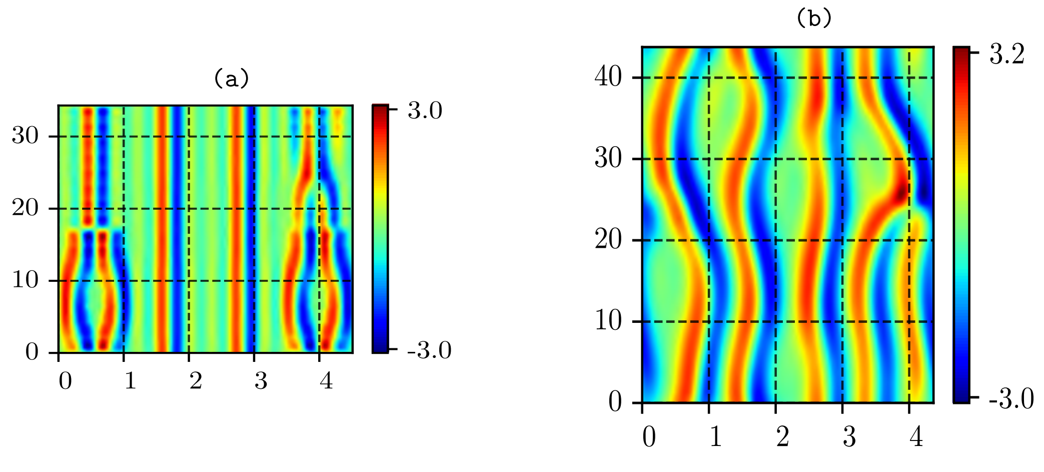

**Figure 46:** (a) Two-by-four spatiotemporal configuration (b) converged orbit.

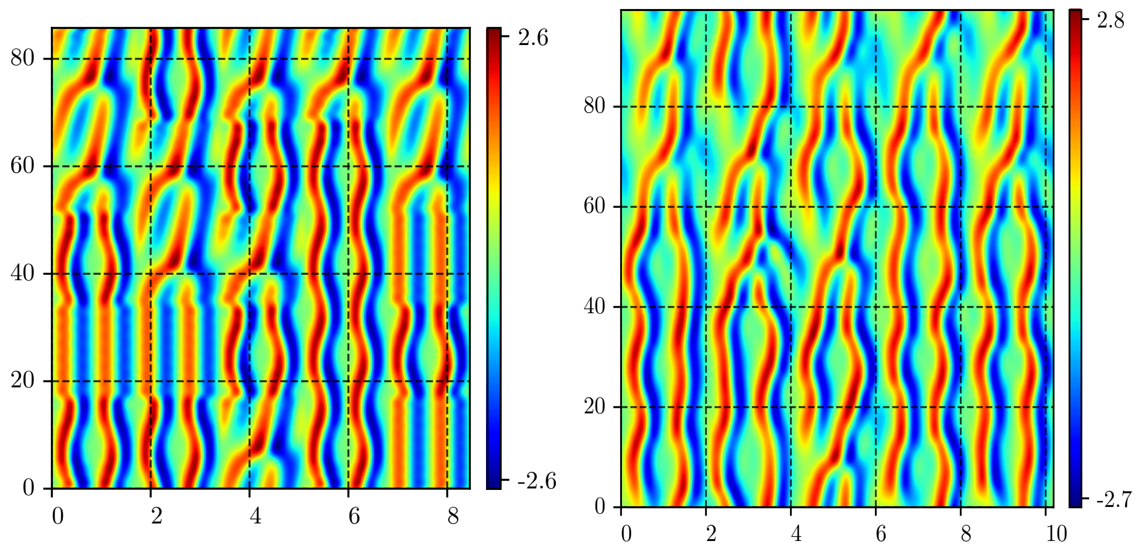

**Figure 47:** (a) 5-by-5 fundamental orbit gluing $(85.7544, 8.4(2\pi\sqrt{2}))$ (b) post-adjoint descent orbit approximation $(99.29, 10.1(2\pi\sqrt{2})$.

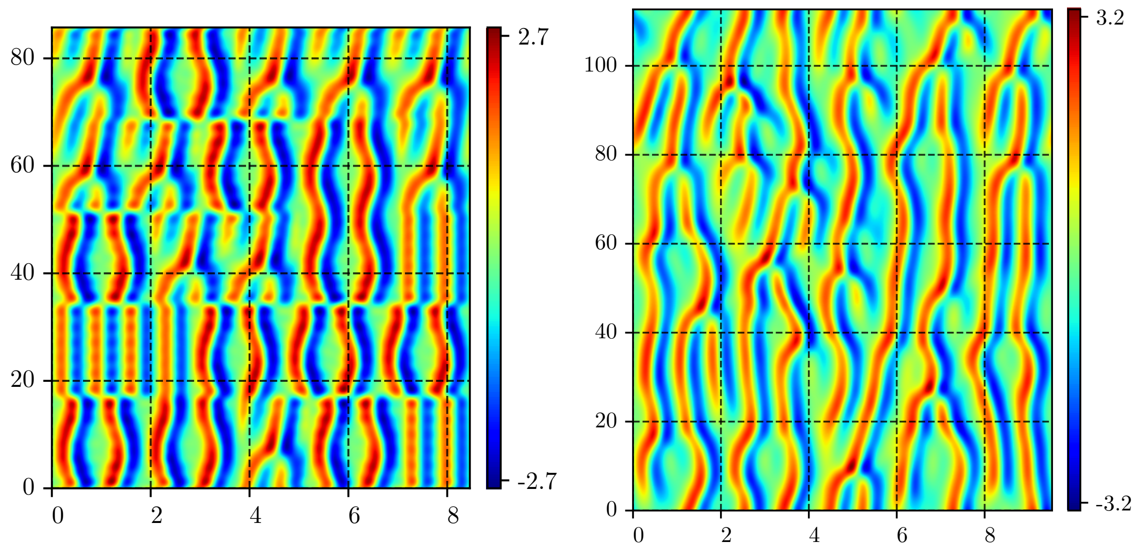

**Figure 48:** (a) 5-by-5 fundamental orbit gluing $(85.7544, 8.4(2\pi\sqrt{2}))$ (b) post-adjoint descent orbit approximation $(112.6, 8.6(2\pi\sqrt{2})$.

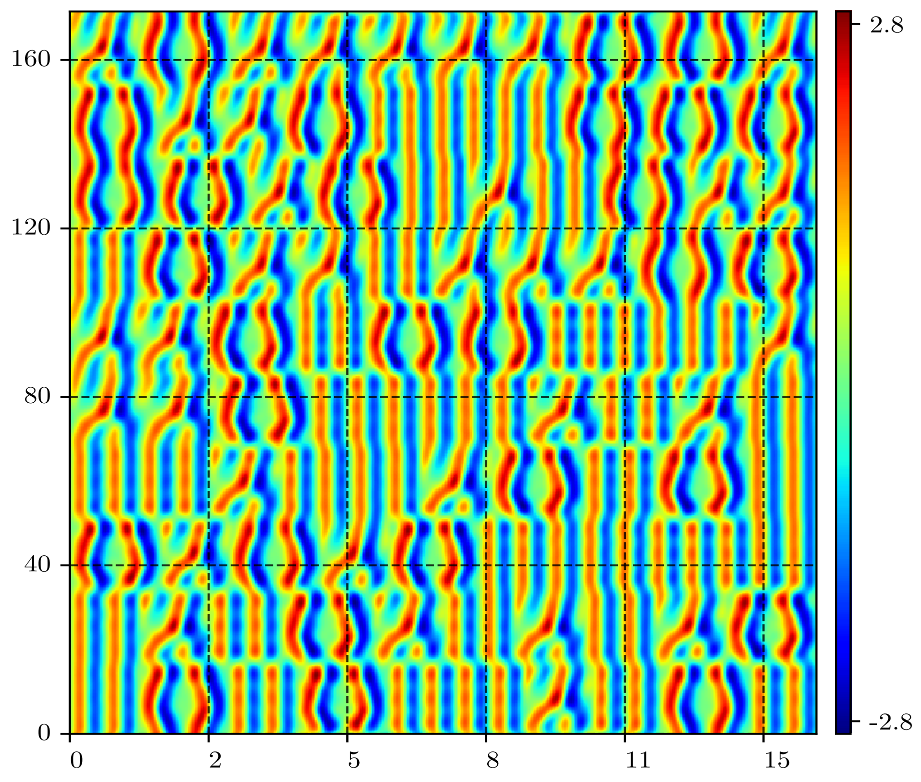

**Figure 49:** 10-by-10 symbolic configuration gluing using the 'resized' tile set, $(T, L) \approx (172, 16.1(2\pi\sqrt{2}))$.

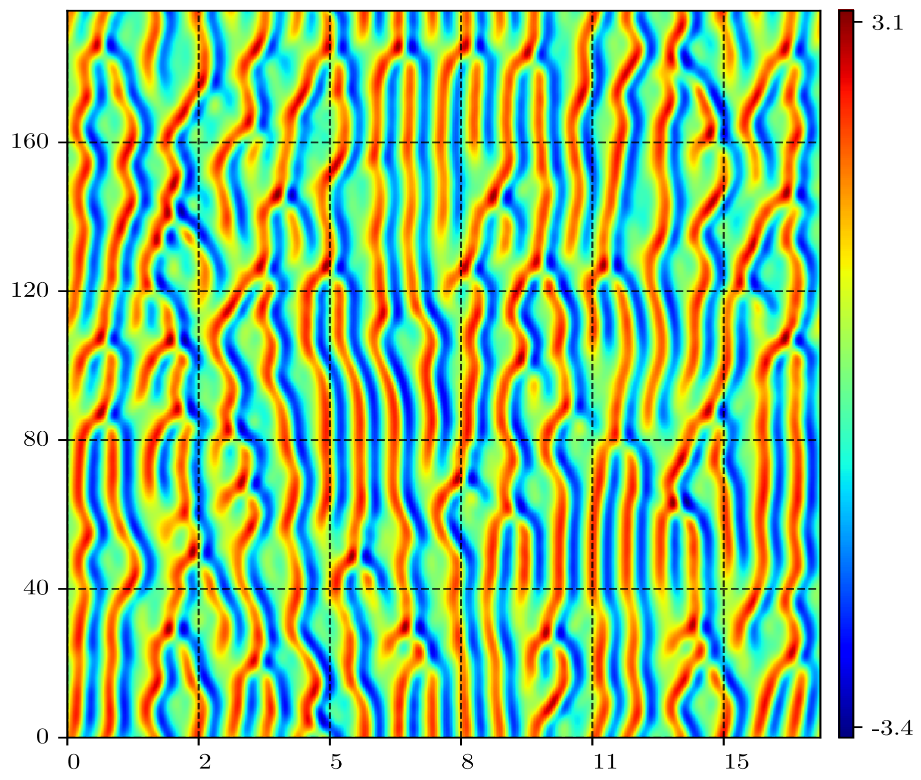

**Figure 50:** Orbit approximation, generated by applying adjoint descent to figure 49; $(T, L) \approx (195, 17.2(2\pi\sqrt{2}))$.

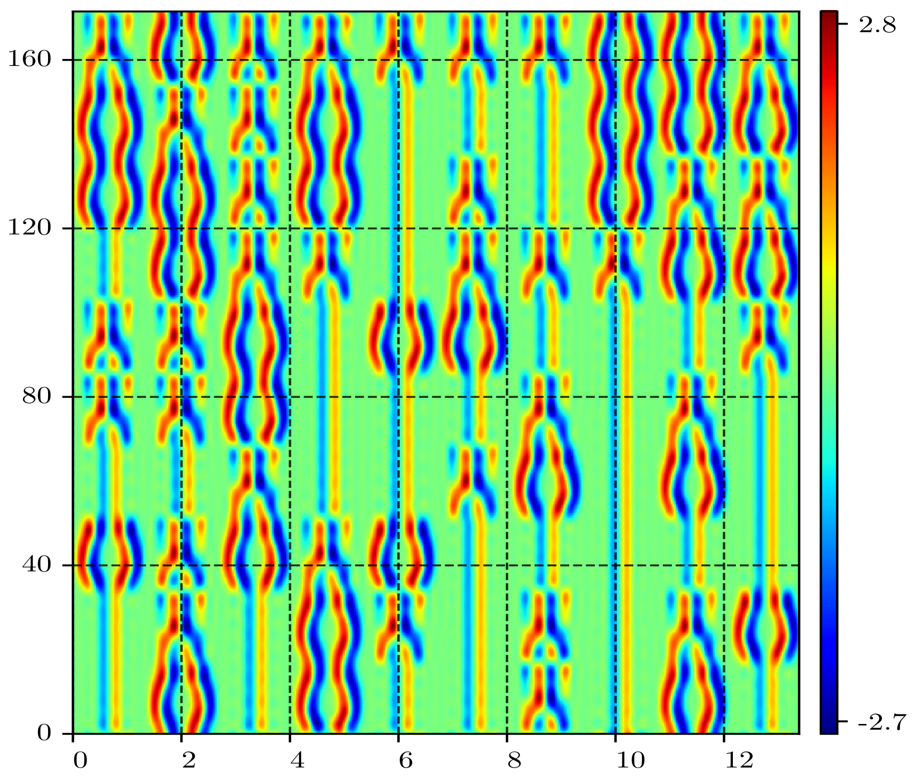

**Figure 51:** 10-by-10 symbolic configuration gluing using the 'spatially padded' tile set $(T, L) \approx (172, 13.3(2\pi\sqrt{2}))$. This is the same symbolic array as in figure 49

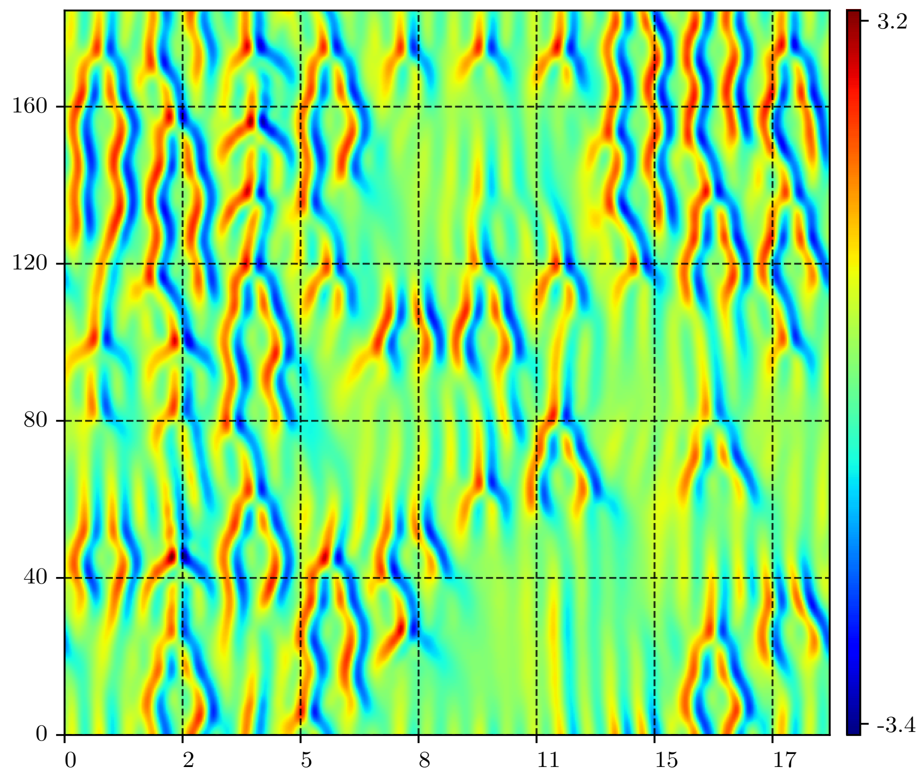

**Figure 52:** (Disclaimer: this is considered an example of a poor result). Orbit approximation resulting of application of adjoint descent to the 10-by-10 spatiotemporal gluing using the spatially padded tile set; $(T, L) \approx (184, 19.3(2\pi\sqrt{2}))$.

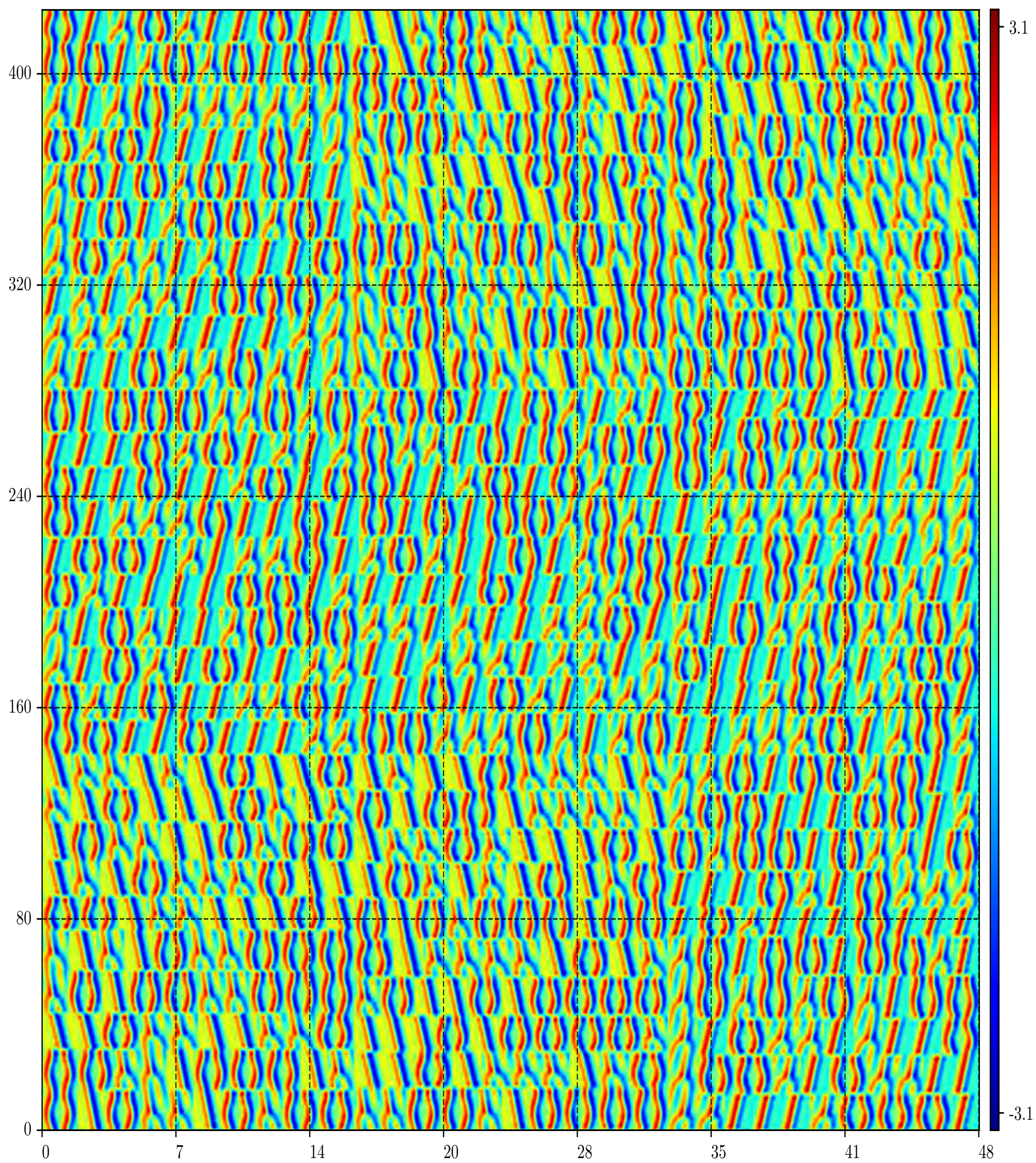

**Figure 53:** 30-by-30 symbolic configuration gluing using an experimental tile set, including local spatial translation velocity; $(T, L) \approx (424, 49(2\pi\sqrt{2})$, $\mathcal{O}(10^5)$ residual.

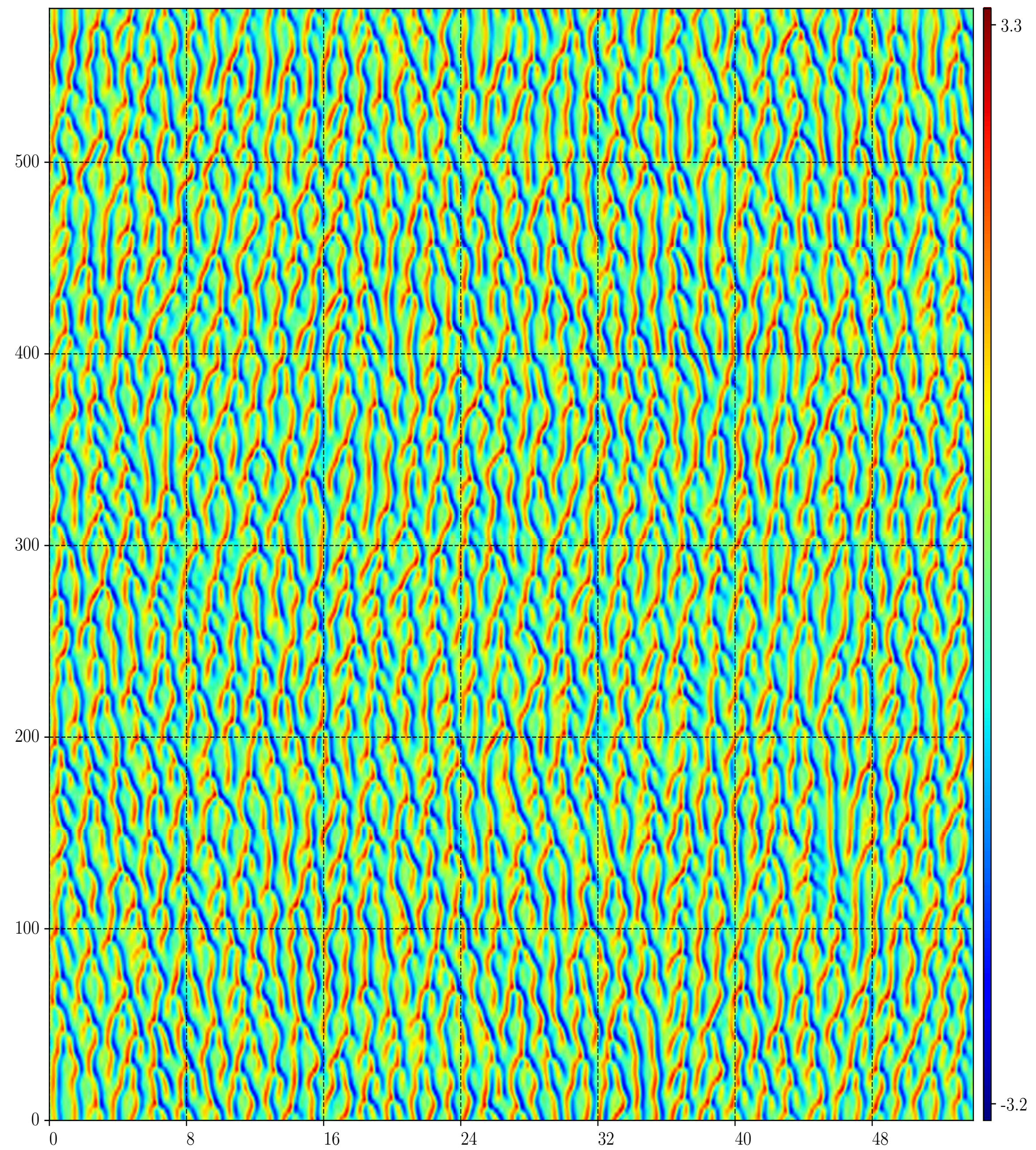

**Figure 54:** Orbit approximation resulting from 30-by-30 spatiotemporal configuration of fundamental orbits; $(T, L) \approx (580, 54(2\pi\sqrt{2}))$, $\mathcal{O}(1)$ residual.

motivation. By doing so, we produce a very large, approximate orbit defined on a spatiotemporal tile with dimensions $\approx (580, 54(2\pi\sqrt{2}))$. The resulting orbit approximation *still* does not accurately reproduce the variety of patterns embedded in the large spatiotemporal simulations such as figure 25. The residual is $\mathcal{O}(1)$, equally as bad as the much smaller orbit approximation figure 48; indicating that large spatiotemporal domains can now at least be approached numerically.

We pause to highlight the significance of these results; it is easy to forget that we are not simply cutting and pasting images together. We have successfully used unstable periodic orbits as building blocks to describe larger regions of spacetime; in the form of finding larger orbits: displayed in figure 45 and figure 46. If the same could be said for the Navier-Stokes equations then this would likely be revolutionary. In addition, this has allowed us to approach very large spatiotemporal domains numerically. For some comparison, example calculations from others have used $L = 22 \approx 2.5(2\pi\sqrt{2})$ and $L = 38.5 \approx 4.33(2\pi\sqrt{2})$ in the antisymmetric subspace [12, 78]. These domain sizes are blown out of the water by figure 54; which is an example of a calculation on a tile with spatial dimension $L \approx 54(2\pi\sqrt{2})$.

These numerical results indicate to us that the spatiotemporal formulation has merit. The path forward is now to determine all admissible spatiotemporal configurations; not an easy prospect. While this process has yet to be formalized, we can work towards finding all admissible combinations of fundamental orbits, based on numerical convergence. With that said, the next chapter demonstrates some example codes; hopefully serving as an advertisement for others to adopt the orbithunter framework.

# CHAPTER V

# ORBITHUNTER

While still undergoing the final stages of development (packaging and setup tools), one of the main deliverables of this thesis work is the numerical codes which have produced the results described in chapter 4. Across the course of this research, many different rewrites of the same integration routine, namely, the ETDRK4 routine [67] were encountered. These mostly redundant rewrites of the same code indicated to me that some universal framework was desperately needed, not only for the Kuramoto-Sivashinsky equation but for nonlinear chaotic partial differential equations in general; unfortunately, no one else is aboard the spacetime train as of yet. The goal of this framework is to allow researchers to immediately compare results as well as incorporate others' work directly into their own. Additionally, it seemed like having open source code was the more honest and open approach, contradicting the all too common occurrence of researchers having secret research codes that no one else can use. Another motivation was simply the challenge or inability to utilize or adapt others' codes. For these reasons the spatiotemporal techniques developed in chapter 3 and put on display in chapter 4 have been packaged into what I now refer to as the Python computing package 'orbithunter'. This development required far more effort than the development of the original research codes. The main goal was maximization of user friendliness, modularity and to enable nimble calculations via notebooks. Modularity implies easy incorporation of modules written for other equations; meaning the spatiotemporal techniques clipping, gluing, etc. needed to be written in a manner agnostic of equation. While these tools have not been tested for other equations, it is believed that they are at the very least well positioned for generalization. As a disclaimer orbithunter's development was entirely an independent endeavor. The codes are written to be efficient, Pythonic and as simple as possible; however, I can almost assuredly say there are improvements that could be made. Note that the actual source code for the various spatiotemporal functions is not included in the description that follows; it is much more compactly represented on Github [52], and it was decided that hundreds of pages of source code separated by page breaks was simply not conducive to understanding. It it worth mentioning, however, that the source code follows the Python conventions, except when Physics conventions take precedent. For example, the convention for variables is to always write them in lowercase; however, the spatiotemporal dimension is $T$ not written in lowercase because $t$ is used for the dimensionless time variable, not the period.

## *5.1 Examples using the Kuramoto-Sivashinsky equation*

The computational codes use object oriented programming based approach, based on the custom `Orbit` class. This class serves as the general scaffolding for orbits of all possible equations. It defines all functions required for the spatiotemporal techniques either as placeholders which depend on the equation, or as general functions which do not depend at all on equation. Therefore this class serves only as a template for other equations' orbits and their subclasses, and is never used explicitly. The current implementation only supports the Kuramoto-Sivashinsky equation, whose base class is `OrbitKS`; the different symmetry classes follow logically as subclasses; for spatiotemporal orbits this includes: `RelativeOrbitKS`, `AntisymmetricOrbitKS`, and `ShiftReflectionOrbitKS`. Equilibria are special cases of spatial reflection symmetry; relative equilibria are special cases of relative periodic orbits; it follows naturally then that `EquilibriumOrbitKS` and `RelativeEquilibriumOrbitKS` exist as subclasses of `AntisymmetricOrbitKS` and `RelativeOrbitKS`, respectively. Each of these subclasses has overloaded expressions for symmetry specific calculations; this becomes very convenient because general expressions such as the differential algebraic equation need only be written once, even though operations such as spatial differentiation are defined differently for different symmetry classes. The remainder of this section is dedicated to providing examples of the entire spatiotemporal process as laid out in chapter 4. The code snippets are placed at the end of the section for formatting reasons.

The first step towards finding orbits or in fact any other computation, is the instantiation of an class object. To do so, one need only call the `OrbitKS` as a function. This process is very flexible; control over it depends upon the amount of information specified by the user. If the discretized state is provided (along with the basis it represents) using the keyword arguments `state` and `basis`, then the discretization dimensions can be inferred from the shape of the array, e.g. (32). Essentially, whenever necessary information is not provided can be randomly generated. For example, if the state is not provided, then it will be randomly generated; utilizing the modulation strategies described in sect. 3.2. The code in listing 5.1 demonstrates various methods of guess orbit construction (lines with '#' indicate comments). These examples are not mutually exclusive; it is possible to provide the state and allow the parameters to be generated randomly, for example.

Numerical operations such as differentiation, calculation of the differential algebraic equations, Fourier transforms, plotting, etc. are all accessed via methods and attributes of the class instances. Most of these methods have their own set of keyword arguments which specify various options; their description is left to the `orbithunter` documentation available on Github [52]. A number of example computations using instance methods are displayed in listing 5.2. These expressions convey how numerical expressions can be evaluated with ease. It does not, however, demonstrate how easy it is to chain together operations, providing the ability to perform a very complex calculation in a single line of code. This power is demonstrated in listing 5.3, wherein the following operations are performed: first, an example orbit is generated. Then, this orbit's state is reflected with respect to space, translated 10 dimensionless units in time, differentiated with respect to space, rediscretized to 64 points in time and space, and finally the field is plotted. This has not included the fact that binary operations $+, -, *, **$, addition, subtraction, (scalar) multiplication, and exponentiation are also defined. The second example in listing 5.3 computes the $L_2$ norm of the difference between the squares of the evaluations of the differential algebraic equations. Note that the chained calculations listing 5.3 do not really mean anything; they are simply a random assortment of operations that can be chained together. Up to this point, all operations have been described by either binary operators or instance methods, the spatiotemporal techniques, however, are written as functions which take orbit instances (or a collection thereof), perform some operation and return yet another orbit. The four main algorithms are optimization, clipping, continuation, and gluing. These techniques are accessed via the functions: `converge` for optimization, `clip` for clipping, `tile` and `glue` for gluing, and lastly for continuation, `dimension_continuation` and `discretization_continuation`. The description of these functions is presented in terms of the base `Orbit` class even though it is never used, simply to be less verbose.

The first of these spatiotemporal applications, the `converge` function, provides access to all numerical optimization methods described in chapter 3. This function's only requirement is a class instance, typically a guess orbit. If the residual of said orbit is already smaller than the provided residual tolerance, the optimization routine is terminated. If this is not the case, then the orbit instance is passed to a function dependent on the numerical method specified by the `method` keyword argument. There are also a number of keyword arguments which can be passed to tailor the numerical optimization process, such as the residual tolerance, number of maximum iterations, etc. There is also a set of default values based on numerical results that can be accessed through the key words `precision` and `comp_time`. Another useful key word is `verbose` which takes a boolean value. When true, the function will print its progress, indicating iteration number, residual and parameter values. The `converge` function returns another custom construct, an `OrbitResult`. This object is essentially a copy of the SciPy's OptimizeResult [134]. It contains the optimization statistics such as targeted residual tolerance, residual at each iteration, number of iterations, current status, etc. The default settings of the optimization function utilizes adjoint descent. This function has by far the most options, for more details it is best to read the documentation stored on Github [52]; this isn't even considering the incredible amount of options that the SciPy functions take. Most importantly, `OrbitResult` has the post-optimization orbit instance, accessed by the `orbit` attribute. The benefit of this construct is that it takes any user defined statistic as a keyword argument, so custom statistics can be incorporated relatively easily. The issue is that this behavior hasn't been implemented into `converge` because its currently too vague to implement.

The `clip` function requires three arguments: an `Orbit` instance, the dimensions of the clipping, and the class with which to initialize the clipping. If the clipping's dimensions are chosen such that the clipping contains an approximately antisymmetric field, then it follows logically to think to initialize the clipping as an `AntisymmetricOrbitKS`; however, the safer bet is to assume no discrete symmetry, due to the approximative nature of clipping. The dimensions of the clipping are provided as a tuple of tuples, each representing the subinterval in each dimension which defines the clipping ('tuple' is a built-in Python data type). For example, in the Kuramoto-Sivashinsky equation this represents a pair of tuples, the first indicates which subinterval of time to clip, and the second is the subinterval of space. Because the clipping is often based off of visual inspection, the dimensions provided should match the conventions utilized in plotting. In the example of clipping from a large spatiotemporal trajectory figure 25, the corresponding code is displayed in listing 5.5.

There are two functions which provide access to spatiotemporal gluing; `glue` and `tile`. The `glue` function takes an array of `Orbit` instances and the orbit class constructor (`Orbit` instead of `Orbit()`, note the parentheses). The only other arguments indicate whether or not to use the strip-wise gluing strategy; if `stripwise=True`, then the order of the strip-wise gluing axes can also be specified via `gluing_order` which is a tuple of integers. Often times it is more convenient and useful to provide an array of symbols, and a Python dictionary which translates these symbols into orbits. This is the reason why the `tile` function exists; it is literally just the `glue` function which takes symbolic configurations and a Python dictionary as input. The tile simply maps the array of symbols into an array of orbits, which is then passed to `glue`. For a visual example of such a mapping look to figure 44. For the coding example, two equivalent methods of producing the same orbit are given, one using `tile` and the other using `glue`. While not previously mentioned, a useful utility for gluing and tiling is `read_fpo_set`. This function is used to load in a set of predefined fundamental orbits included as part of the package. There are multiple sets to choose from, representative of the fact that the investigation is still ongoing. The equation (which defaults to the Kuramoto-Sivashinsky equation) can be specified as well as any fundamental orbit dictionary relevant key word arguments. For example, the tile set used in figure 44 used key word arguments, `tileset='extra_space_padded'`, `comoving=True`, and `rescaled=True`. This tile set is defined by zero-padding of the streak and defect orbits so that the relative spatial sizes are respected without having to apply strip-wise gluing. The rescaling and comoving flags rescale the magnitude of each orbits' fields to be the same, and use the comoving representation of the defect, respectively. This rescaling is motivated only by minimizing the discontinuities at the tile gluing boundaries.

Finally, the last pair of functions implement the two different forms of continuation described in chapter 3. The more familiar type of continuation, using constraints to incrementally change tile size, hoping to find the corresponding member of the orbit family, is handled by the function `dimension_continuation`. This function takes an orbit instance, the new dimension value, the corresponding array axis, and lastly the step size for incrementing the dimension. Additionally, there is a flag variable `save` which, when true, saves all intermediate family members. If false, then only the last output of the continuation is returned, the intermediate states being overwritten. Lastly, because the continuation requires repeated application of `converge`, key word arguments of the aforementioned function are also accepted. The second type of continuation, represented by the function `discretization_continuation`, requires an orbit instance, the desired shape of the new collocation grid, and a boolean flag variable, `cycle` which defaults to false. There are two strategies for how to change the discretization; one dimension at a time (`cycle=False`) or to cycle through the axes (`cycle=True`). The goal of this process is to find the 'same' orbit converged on the tile with the new discretization size.

**Listing 5.1:** OrbitKS instantiation

```
from orbithunter import *
# Create an orbit instance with randomly generated state and
# parameters.
example_orbit_0 = OrbitKS()

# The same as above except a random seed for reproducibility
# (gives the same state and parameters with every call)
example_orbit_1 = OrbitKS(seed=4)

# Deciding upon the various random initial orbit parameters;
# ranges for random variables decided by the 'min' and 'max'
example_orbit_2 = OrbitKS(spectrum='gtes', T_min=60, T_max=80,
                          L_min=22, L_max=44, N=32, M=32)

# An example of how cast user provided data as an OrbitKS
example_orbit_3 = OrbitKS(state=user_provided_numpy_array,
                          basis='field', parameters=(100, 100, 0))
```

**Listing 5.2:** Evaluation of numerical expressions

```
from orbithunter import *


# Instantiate a class instance.
example_orbit = OrbitKS()


# Compute the second order spatial derivative
example_orbit_dx2 = example_orbit.dx(order=2)


# Compute the differential algebraic equations
example_orbit_dae = example_orbit.dae()


# Compute the adjoint descent direction, Jacobian adjoint times dae.
example_orbit_adj = example_orbit.rmatvec(example_orbit.dae())


# Spatial reflection of the example orbit
example_orbit_refl = example_orbit.reflection()
```

**Listing 5.3:** Numerical operations

```
from orbithunter import *
# Instantiate an orbit
example_orbit = OrbitKS(N=32, M=32)


# Binary operations
new_orbit = ((4 * example_orbit**2)
             - (example_orbit + OrbitKS(N=32, M=32))**2)


# Chaining numerical operation methods, plot the result
example_orbit.reflection().rotate(10, axis=0).dx().plot()
```

**Listing 5.4:** Clipping from large spatiotemporal trajectory

```
from orbithunter import *

# Read in the integrated orbit saved for reproducibility
integrated_trajectory = read_h5('OrbitKS_trajectory.h5')

# The dimensions of the subdomain to be clipped
clipping_window = ((120, 160), (34.25, 38))

# The clipped subdomain stored in a RelativeOrbitKS instance
clipping_rpo = clip(integrated_trajectory, clipping_window,
                    clipping_class=RelativeOrbitKS)
```

**Listing 5.5:** Clipping from large spatiotemporal trajectory

```
from orbithunter import *

integrated_trajectory = read_h5('OrbitKS_trajectory.h5')

clipping_window = ((120, 160), (34.25, 38))

clipping = clip(integrated_trajectory, clipping_window,
                    clipping_class=RelativeOrbitKS)
```

**Listing 5.6:** Tile and glue

```
from orbithunter import *
import numpy as np

# Load the default tile set
tiling_dictionary = read_fpo_set()

# Create a random symbolic array
np.random.seed(0)
symbol_array = (3*np.random.rand(5, 5)).astype(int)

# Convert symbolic array into array of orbits
array_of_orbit_instances = np.array([tiling_dictionary[symbol] for
                                          symbol in symbol_array.ravel()]
                                         ).reshape(*symbol_array.shape)

# Glue the array of orbits, resulting in RelativeOrbitKS instance
glued_rpo = glue(array_of_orbit_instances, RelativeOrbitKS,
                 stripwise=True, gluing_order=(1,0))

# Use tile instead, takes symbols and dict as input
tiled_rpo = tile(symbol_array, tiling_dictionary, RelativeOrbitKS,
                 stripwise=True, gluing_order=(1,0))
```

**Listing 5.7:** Continuation

```
from orbithunter import *

# Load the default tile set
example_orbit = OrbitKS(N=32, M=32)

# Try to converge orbit on a tile with larger time extent
larger_T_orbit = dimension_continuation(example_orbit,
                                        example_orbit.T + 1.0,
                                        axis=0, step_size=0.5)

# Try to increase its discretization size, save the intermediates
# Note that the previous operation might have failed completely,
# or at some intermediate point between T and T+1.
larger_NM_orbit = discretization_continuation(larger_T_orbit,
                                              (64, 64), cycle=True)
```

## 5.2 Generalization to other equations

Due to the modularity property, inclusion of new equations can be accomplished by developing and including small portions of code at a time. The following serves as a guideline for the development of these new modules; a process which takes a surprisingly small amount of work.

The general process is as follows: first, develop methods to parse input and initialize the set of attributes necessary to fully describe an orbit instance. Next, define the transform methods and numerical operations required to evaluate the differential algebraic equations, matrix-vector products and potentially the Jacobian matrix depending on the numerical method desired. It is recommended to pause development and test the performance of the code at this point; visualization is the recommended method for this testing as it allows quick interpretation of results.If the numerical optimization is providing valid results and is relatively efficient then the clipping, continuation and gluing methods are viable. Regarding naming conventions; the documentation and naming conventions should follow the other orbithunter modules; the base class for a new equation should consist of 'Orbit' followed by an abbreviated equation name, i.e. `OrbitKS`. The names of all classes and subclasses should use 'CamelCase' and represent the symmetry without being overly verbose. For example, `RelativeOrbitKS` is used instead of `RelativePeriodicOrbitKS`; everything is periodic hence the redundant verbiage can be removed.

The basic attributes which required for `Orbit` instances are the 'state', 'basis' and 'parameters' attributes. The 'state' attribute contains all state information in the basis defined by the 'basis' attribute. Even for discrete systems where it does not quite make sense, the default basis is 'field'. Likewise, 'parameters' contain all parameters required to define the orbit. In other words, the required variables are those which are required to define the state vector (11). Parsing the input is performed by the `_parse_state`, `_parse_parameters` methods. The underscore prefix is simply a python convention indicating the function is only intended for internal use. The state parsing determines the dimensionality of the collocation grid from a state provided as a NumPy array and the basis it is in. Likewise, the parameter parsing essentially validates the values provided. Lastly, the `_random_initial_condition` method generates a state when none is provided; clearly if we do not have a state then we cannot do anything else. Other requirements are the somewhat unaesthetic book-keeping variables, `field_shape`, `dimensions`, `parameter_labels`, `dimension_labels`. The first two are properties, the second are static methods; that means they do not need to be provided to instantiate an orbit, but they do need to be defined for the class. In order, these return the shape of the collocation grid, the tile dimensions (i.e. subset of parameters), the string labels given to the parameters and dimensions, respectively.

Once an orbit instance can be properly initialized, the next step is to develop the necessary transforms, handled by the `transform` method. This method is a wrapper for all possible transformations, i.e. all possible initial and final bases. The actual transforms themselves are written by the user and can be named whatever the user desires. The recommended approach is to write the transforms themselves as internal methods (underscore prefix) and to label them by the dimensions, e.g. _space_transform and _inv_space_transform for the Kuramoto-Sivashinsky equation. Orbithunter does not require the transforms to be written for each dimension individually, rather, discrete symmetries and their selection rules typically require this decomposition into 1-D transforms.

The output of the transform method is an orbit instance whose state is in the new basis, specified by the keyword argument 'to'; e.g. orbit.transform(to='modes'). The only basis expected by orbithunter is the 'field' basis, as that is the basis in which the data is saved.

The next step is to develop the requirements of the numerical optimization methods. The orbithunter philosophy is to write general expressions in the base equation class, e.g. OrbitKS, in terms of symmetry specific components. An example is given in terms of ShiftReflectionOrbitKS and AntisymmetricOrbitKS and the evaluation of the differential algebraic equations (83). These two subclasses have equations which share the same form; it is only the selection rules which are different. This affects differentiation and Fourier transform methods, however, regardless of the selection rules spatial differentiation is still performed by multiplication by spatial frequencies. In other words, the dae method can be written in terms of general differentiation methods, and likewise the differentiation methods can be written generally as well, the only difference we can write the spatial derivative functions generally, and delegate the specifics to the array of spatial frequencies. This is very beneficial, there is only a need to rewrite the functions affected by selection rules as opposed to all functions.

The methods required for optimization are the differential algebraic equations, the cost function and the corresponding derivatives. dae, matvec, rmatvec, jacobian, state_vector, from_numpy_array, residual, cost_function_gradient. The dae method returns an orbit instance whose state is the evaluation of the differential algebraic equations. The matvec and rmatvec methods both return matrix-vector products in the form of orbit instances. The difference is that the former uses the Jacobian $\mathbf{J}$, the latter, adjoint Jacobian. The names of these methods is motivated by SciPy linear algebra conventions. The jacobian method returns the Jacobian matrix (93) as a two dimensional NumPy array.

The `state_vector` method simply returns the current state and parameters in their vector representation (11). The residual returns a float whose value is the given by (19) and finally `cost_function_gradient` returns an orbit instance containing the matrix-vector product of adjoint Jacobian and differential algebraic equations; in other words, the adjoint descent direction. The method `from_numpy_array` parses the state vectors in NumPy array format into orbit instances, taking constraints into account.

For clipping, the only extra requirements are the ability to reference the tile dimensions in plotting and non-plotting units (which of course may be the same); all other operations utilize the physical field basis and slices of the corresponding NumPy array.

For gluing, the actual concatenation of the orbit states is handled by NumPy. The extra requirement consists solely of the method `glue_parameters`. This function simply returns the dimensions of the glued tile, given the dimensions of the constituent orbits. The constituents' dimensions are provided as the tuple of zipped (Python built-in) values, i.e. $((T_1, T_2, ...), (L_1, L_2, ...))$. The zipping is handled in the gluing function itself; `glue_parameters` need only be able to parse this form of data.

Lastly, the continuation functions require the ability to constrain the various dimensions, i.e. an attribute called `constraints` which contains a dictionary of booleans, and a method `constrain` which changes the values from false to true when the specified axis is to be constrained.

The following functions are not technically required, but likely necessary for gluing and clipping to function properly. The functions include `plot`, `reshape`, and `rescale`. Visualization is very useful for checking results; it can be less than straight forward in higher dimensions, however. Accommodation of fundamental domains is also recommended, but often times plotting the full solution is acceptable. Reshaping is (typically) necessary for clipping and gluing. The `reshape` method requires `_pad` and `_truncate` methods, which handle rediscretization by zero padding and truncation respectively. Rescaling has been found to be useful but is definitely the least important of these three methods.

To recap, the spatiotemporal numerical codes have been bundled into a computational Python package called orbithunter. The goal of this package is to make these spatiotemporal techniques accessible to all and also create a framework which allows for easy generalization of these techniques to other equations. Examples of the main techniques such as clipping, gluing, continuation, etc. are given in sect. 5.1. Likewise, a small walk for how to generalize orbithunter to other equations is provided in sect. 5.2. The techniques, results, and computational codes have all been detailed; our last endeavor is to come to a conclusion and discuss future work.

# CHAPTER VI

# CONCLUSIONS AND FUTURE WORK

We have detailed a spatiotemporal formulation of turbulence which treats all continuous dimensions with translational invariance equally and explains spatiotemporal solutions as collections of fundamental orbits. By transforming the Kuramoto-Sivashinsky equation (4) to a spatiotemporal Fourier mode basis (83) and solving the corresponding optimization problem (19) sect. 3.1, we were able to create a collection of orbits with varying sizes and symmetries. Once these solutions were found, the new spatiotemporal techniques developed by this work can be applied. These new techniques allow us to extract small orbits from larger orbits, denoted 'clipping', developed in sect. 3.3 and build larger orbits from by combining smaller orbits, denoted 'gluing', in sect. 3.5. The combination of these two methods enables an algorithmic way of building up or breaking down solutions, providing access to any tile size, theoretically. In other words, using these techniques have granted us the opportunity to explore the space of solutions. Both of these techniques rely on the robustness and efficiency of the optimization methods; therefore improvements should always be searched for. Although the symbolic dynamics has yet to be formalized, the combination of these techniques and methods provides a numerical foundation with which to investigate. Specifically, by gluing members of the three continuous families of fundamental orbits we can begin to probe the grammar by searching for admissible orbits. Before we can do so, however, we need to ensure that we have collected all fundamental orbits. While we have acted so far as if there are only three fundamental orbit families, there are examples of patterns which either represent new fundamental orbits or are some type of emergent phenomenon resulting from certain spatiotemporal configurations; this has not yet been determined. This is concerning, as we know from the work on cycle expansions that short cycles have the greatest importance [19]; if we miss any of these we will not be able to progress. Two examples are displayed in figure 55, wherein a higher order 'defect' and an ensemble of streaks into some larger coherent structure are displayed. Likewise, it is unknown if the local spatial shift velocities which occur in large simulations such as figure 25 manifest due to shadowing of different defect family members, or if some new fundamental orbit is required. How do we test whether or not all important fundamental orbits have been captured? One method, which currently can only be approached approximately, is to reconstruct a very large spatiotemporal simulation using gluing. This would likely be a much more involved construction than, for example, the huge tiling of figure 54. The idea is to cover the large trajectory with the set of fundamental orbits; any large spatiotemporal regions which cannot be covered would imply the existence of currently uncollected fundamental orbits. Not knowing whether or not we have captured all fundamental orbits begs the question; what quantitatively makes the fundamental orbits, fundamental? We previously argued that it is the 'irreducibility' of minimal spatiotemporal orbits which constitute this set. However, even the defect, streak and wiggle fundamental orbits can be decomposed into combinations of streaks and symmetry operations; quite literally, as displayed in figure 56. In other words, the fundamental orbits may need a more in depth description in terms of symmetry or perhaps even symmetry breaking; the spatiotemporal wiggle is a 2-cell streak solution which has broken temporal translation invariance. Likewise, the spatiotemporal defect might be the 2-cell streak solution with broken time translational invariance *and* broken spatial reflection symmetry.

In regards to the computational scaling of spatiotemporal methods, they should capitalize on advances in computing, specifically the increase in number of computational cores and parallel computing [139], better than the dynamical systems counterparts. Our spatiotemporal formulation is in the same spirit but unfortunately more work is required as the current spectral methods are global; the division into subdomains that Wang *et al.* [139] use is not afforded to us. More specifically, their idea is that by subdividing spatiotemporal domains into subdomains, the equations being solved locally on each subdomain, in parallel, and then combined together at their boundaries. The first idea that comes to mind is to use Chebyshev spectral methods with different boundary conditions [128].

The most important open question is how to incorporate continuous families into the

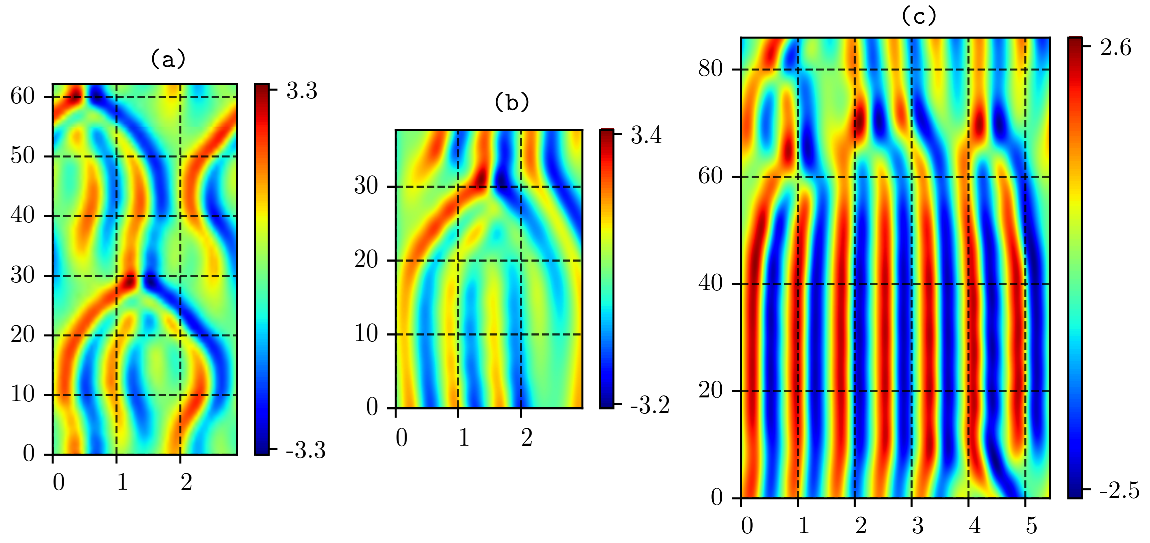

**Figure 55:** Potential fundamental orbits, (a)-(b) higher-order defects, (c) 'streak patch'.

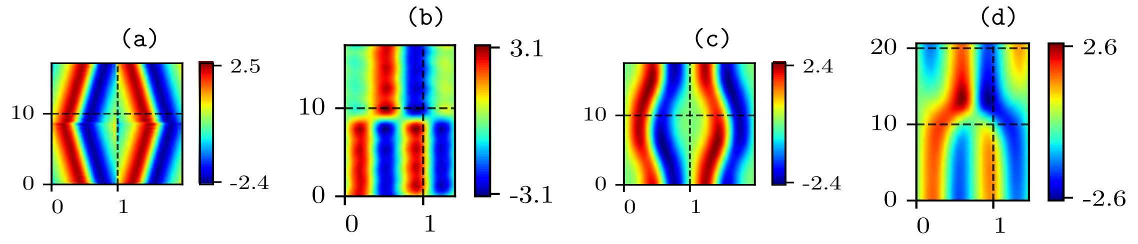

**Figure 56:** (a)-(b) Initial orbit guesses constructed from only streaks and symmetry operations, (c)-(d) the orbits they converge to, respectively.

proposed 2-dimensional spatiotemporal symbolic represention. The existence of continuous families makes the determination of the symbolic represention's grammar particularly difficult, as determining the grammar is ultimately an empirical process in this context. The admissibility of every periodic orbit is dependent upon the convergence of the optimization problem, which in turn depends on the quality of the guess orbit constructed via gluing as well as the employed numerical algorithm. The former is directly dependent upon which fundamental orbit family members are used in the construction of the initial guess orbit. In either case, if numerically the guess does not converge, but the orbit is admissible, we have a false negative. The course of action is to then improve the optimization and gluing methods, with respect to their frequency of convergence. The most obvious gluing improvements include 'proper' usage of the fundamental orbit continuous families and group orbits. As previously mentioned, continuous families will need to be incorporated into the symbolic dynamics, therefore they should clearly be included in the gluing process. In other words, instead of using three static representatives of the families we would need to reference a sample of family members and their group orbits during the gluing. The goal being to minimize discontinuities at the gluing boundaries as well as differences between tile dimensions. The problem with such an implementation is that the number of combinations of family members and their group orbits quickly grows out of hand, making the gluing much more complex than the original optimization problem, without knowing if this is even required. Related to the inclusion of the group orbits during gluing; historically, when performing simulations on a single spatially periodic domain, Galilean invariance has been invoked to constrain the mean value of the velocity field to zero [24, 67]. This does not mean, however, that the *local* Galilean velocity should be zero; This detail could be included in each gluing such that the local Galilean velocity of each fundamental orbit would be included as a free parameter. This adds yet another factor into the already complicated gluing process; however, by doing so we can theoretically construct better guesses by increasing the agreement between fundamental orbits at their boundaries. In regards to improving the numerical methods, searching for different algorithms, improving current algorithms, and investigating the properties of the systems of equations are all possible avenues of study.

It should be no surprise that the other type of error, false positives also occur; in other words, numerical convergence does not always indicate admissibility; why? Imagine that we construct a spatiotemporal configuration, assumed to be a spatiotemporal orbit without symmetry. Then imagine that this converges to an equilibrium solution. Clearly, the original spatiotemporal configuration is not represented by the converged solution. Therefore, in addition to convergence we require some other means of validating the results; currently this is limited to visual inspection. Visual inspection is not viable due to how poorly it scales with the exponentially growing number of spatiotemporal configurations. The two primary ideas for automating this validation process are convolutional neural networks and persistent homology, both of which are to be included in the computational codes.

In regards to other equations, the generalization of orbithunter was discussed in chapter 5. By virtue of this framework, this should take considerably less effort for others, as the clipping, gluing, and numerical optimization methods have been written in a manner as to be agnostic of equation. Therefore, one of the main goals is to simply advertise and attract collaborators to either work on the Kuramoto-Sivashinsky equation or on new equations. Currently, the next equation planned is Kolmogorov flow with periodic boundary equations; some general examples of Kolmogorov flow are refs. [83, 122]. With this, our exploration into a new spatiotemporal theory, and its computational successes comes to a close. Thank you for reading this work.

# APPENDIX A

# NUMERICAL EXTRAS

The following serves as a collection of small numerical details and definitions. Specifically, to perform the tensor and matrix-vector based calculations two operations are required: element-wise multiplication or *element-wise product*, and the Kronecker product between two tensors.

The element-wise product goes by many names such as component-wise product, Schur product, or Hadamard product [20, 121]. Typically these are defined and utilized with matrices, but the generalization of this operation to tensors is straightforward. Given a pair of tensors with dimensions $j, k$; the element-wise product is represented by the operator $\cdot$ and is defined as

$$\begin{bmatrix} m_{11} & \dots & m_{1k} \\ \vdots & \ddots & \vdots \\ m_{j1} & \dots & m_{jk} \end{bmatrix} \cdot \begin{bmatrix} p_{11} & \dots & p_{1k} \\ \vdots & \ddots & \vdots \\ p_{j1} & \dots & p_{jk} \end{bmatrix} = \begin{bmatrix} m_{11}p_{11} & \dots & m_{1k}p_{1k} \\ \vdots & \ddots & \vdots \\ m_{j1}p_{j1} & \dots & m_{jk}p_{jk} \end{bmatrix}. \tag{126}$$

The product (126) is only well defined for tensors with the same number of elements; element-wise multiplication is only defined for orbits whose tiles' discretizations have the same number of collocation points. Stated differently, the product $u \cdot u'$ implies $u(t_n, x_m) \cdot u'(t'_n, x'_m)$, however in order to be defined mathematically we only require that $N = N', M = M'$. This is not the same as requiring the tile dimensions $(T, L)$ to be identical; although this is nearly always the case. The second operation is the Kronecker product $\otimes$ between two matrices. For two matrices $\mathbf{A}, \mathbf{B}$ of dimensions $N \times N$ (for simplicity) the Kronecker product results in a $N^2 \times N^2$ via multiplication of each element of $A$ by the matrix $B$

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1k}B \\ \vdots & \ddots & \vdots \\ a_{j1}B & \dots & a_{jk}B \end{bmatrix}. \tag{127}$$

The Kronecker product is used in chapter 2 to generalize the finite matrix representation of a spatial operator or temporal operator to spacetime, via Kronecker product with the appropriately sized identity matrix. Put very simply, the 'generalization to spacetime' just means creating a block diagonal matrix whose blocks are copies of the original operator that acts on either on a single space or time point.

The second important numerical detail is why I felt justified to use a fully aliased pseudospectral method ref. [15]. My argument is that while aliasing can be devastating for temporal evolution due to the contamination of the higher Fourier mode components by their aliases, $(k' = k + N)$ where $N$ is the number of collocation points, or discrete period, the spatiotemporal problem has no dynamics and so the notion of 'propagation of error' does not exist. To support these claims, I looked towards the comparison of aliased and de-aliased calculations of the periodic, multidimensional Navier-stokes equations. Useful discussions may be found in [41, 68, 91, 100]. All of these authors conclude that with sufficient resolution, aliased calculations are quite acceptable. This is not without pushback; Moser, Moin and Leonard caution against aliased calculations [92]. They present a single, poorly resolved, aliased calculation of Taylor-Couette flow and compare it with three de-aliased calculations, one poorly resolved, one moderately resolved, and one well resolved. Their single aliased result is certainly much worse than their well resolved, de-aliased case, but their poorly resolved, de-aliased case is no better than the aliased one. The interpretation of these results is that any computation needs to have sufficient resolution. My interpretation is that because they did not include a high resolution aliased calculation, their claims are moot.

More motivation from Canuto, Hussaini, Quateroni and Zhang [15] are their fig. 7.1 and fig. 7.4, where the fully aliased but more resolved terms seem to beat out even the dealiased computations in energy conservation of the Korteweg-de Vries (KdV) equation for fig. 7.1. Their fig. 7.4 is a reproduction of the effects of aliasing in the transition to turbulence in channel flow by Krist and Zang [73]. Only the high resolution (aliased) seems to be physically representative of the actual solution, and even the dealiased computation on a coarser discretization, while better than an aliased computation of same dimension, still does not prevent artificial oscillations. Currently the status of the spatiotemporal formulation simply needs tools for computation of orbits within specified tolerances. The `discretization_continuation` tool included in orbithunter was added for this exact reason; if there is ever the need for very accurate computation of a specific orbit, then the discretization dimension can be increased (theoretically) until the orbit is 'fully resolved' in terms of the convergence of Fourier modes. This flexibility in dimensionality has not been thoroughly tested, unfortunately, and so it is unknown whether the results from figure 17 generalize.

Regarding the discretization sizes of orbit guesses; while not included in orbithunter, it would be possible to survey the space of discretization sizes via interpolation, to find the 'best' discretization size, determined by the minimal residual value (pre-optimization). This process had been tested in the past and went under the name 'residual guided rediscretization', however, no conclusions were ever made. Likewise, the same idea can be applied to converged orbits in order to compress orbit information into a minimal number of modes, the idea being to attempt to nip computational memory problems in the bud by finding the smallest dimensional representations of each orbit. The issue with this method is that it currently requires visual inspection of each result to ensure the same orbit is still being converged to.

One aspect that must be taken into consideration whenever a linear system is being solved numerically is whether or not the results are accurate. One quantitative measure of this is the condition number of the corresponding matrix [129]. The condition number is calculated by taking the ratio of the largest and smallest singular values computed by singular value decomposition. Broadly speaking, the condition number indicates how sensitive the system is to error or perturbation. If the condition number large the system is said to be "ill-conditioned" which in turn greatly affects the accuracy of solutions. For chaotic initial value problems the magnitude of the condition number depends on the maximal Lyapunov exponent [140]. For our purposes, the quality of computational results is quantified by the residual of the cost function, but this does not mean that the numerical methods cannot be improved. For the newly implemented iterative methods, this can be

accomplished by applying what is known as *preconditioning* [108]. Deciding on a specific *preconditioner* to use is a dark art, but some general guidelines are that the preconditioner should not be expensive to compute and it should approximately equal the inverse of the ill-conditioned linear operator. As mentioned before, preconditioners should be cheap and rescale the components in such a manner as to be of the same magnitude [97]. The stiffness of the Kuramoto-Sivashinsky equation
rfks05com comes from the linear term; therefore the approximate inverse of the linear term is used, such that the preconditioning can be represented by the matrix

$$P = \mathrm{diag}\Big((|\omega_j| + q_k^2 + q_k^4)^{-1}\Big)\,. \tag{128}$$

Although written in matrix form, the form of (128) allows it to be computed using element-wise multiplication. Its inclusion (or at least the specific implementation here) can be described in either a formal or practical sense. Introducing this into the definition of the cost function (19) is akin to changing the norm used in computation of the residual. In addition to rescaling the modes, it can also be useful to rescale the changes to parameters, typically chosen to satisfy

$$\begin{aligned} P \cdot \delta T &= \frac{\delta T}{T}\,, \\ P \cdot \delta L &= \frac{\delta L}{L^4}\,. \end{aligned} \tag{129}$$

The preconditioning given by (129) is mainly an attempt to imitate the effect of (128) for the parameters $(L, T)$.

When and why is this preconditioning used? Preconditioning of the form given by (128) and (129) is typically only ever applied to random initial conditions created with strategies that do not damp the higher order spatial frequency modes. In our experience, the behavior of such initial conditions is that the changes to the spatial dimension become absurdly large; far too large to be properly resolved by the number of modes. Therefore, to prevent this rapid 'expansion', preconditioning is applied simply as a means of handling 'poor' initial conditions.

Other set of operations used in the derivation of the matrix form of the Jacobian, in the context of the elementwise product, are 'direct-matrix calculus' operations [20]. For an arbitrary matrix $\mathbf{A}$ (independent of $\tilde{u}$), functions $\mathbf{F}(\tilde{u}), \mathbf{G}(\tilde{u})$, the relevant chain and product rules are

$$\begin{aligned}
\frac{\partial}{\partial \tilde{u}}(\mathbf{A}\tilde{u}) &= \mathbf{A} \,, \\
\frac{\partial}{\partial \tilde{u}}(\mathbf{F}(\tilde{u}) \cdot \mathbf{G}(\tilde{u})) &= \text{diag}[\mathbf{G}(\tilde{u})]\frac{\partial \mathbf{F}(\tilde{u})}{\partial \tilde{u}} + \text{diag}[\mathbf{F}(\tilde{u})]\frac{\partial \mathbf{G}(\tilde{u})}{\partial \tilde{u}} \,, \\
\frac{\partial}{\partial \tilde{u}}(\mathbf{A}\mathbf{F}(\tilde{u})) &= \mathbf{A}\frac{\partial \mathbf{F}(\tilde{u})}{\partial \tilde{u}} \,. \qquad (130)
\end{aligned}$$

It can be helpful to derive expressions using matrix calculus operations (130) such as to create a second expression with which to compare to.

In regards to computational memory requirements of the spatiotemporal method; demonstrated in figure 17, orbits can be compressed into a small number of modes. Other previous studies that used spatiotemporal discretizations (fundamentally different in nature, however) utilized finite differences and therefore required upwards of 512 to 1024 discrete time points, *for short orbits* [75]; clearly an unscalable requirement. Orbits of the same temporal period (but not the same orbit) can sometimes be represented with as few as 16 temporal modes. This reduction in the number of extra computational degree of freedom is how we argue that these methods have a chance to be extended to higher dimensions. As a preliminary test of these ideas, one of the shortest periodic orbits from plane-Couette flow [46] was discretized using 16 time points and then a spatiotemporal Fourier transform was applied. Inspection of the magnitude of the spatiotemporal modes showed that while the discretization had increased by a factor of 16, the number of modes above machine precision $10^{-14}$ in magnitude was only equal to one quarter of this number. Further investigation is required, but this seems to demonstrate that the increase in computational memory requirements may not be nearly as bad as suspected.

One last comment regarding discretizations; can these coarse discretizations be compared to results from dynamical systems' formulation? The answer is yes; using interpolation, a converged orbit had its collocation grid increased in size from $[64, 32]$ to $[4096, 512]$. After a relatively small number of adjoint descent steps, a spatial strip was taken from the approximate solution and integrated in time. This integration fully reproduced the high-resolution orbit approximation within some errors. The orbit chosen had temporal period on order of $T = 80$. This demonstrates that the aliased, coarse simulations and numerical results are valid and at least slightly protected from others criticisms.

# APPENDIX B

# DISCRETE LAGRANGIAN METHODS

The general benefit of the weak formulation is that by using integration by parts the highest order derivative can be made to be two. the differentiability requirements on $u$ from fourth to second order. This can be of great benefit in particular circumstances where shocks and compressibility play a role. Applying integration by parts to (13), the weak form of the Lagrangian density can be written

$$\mathcal{L} = \frac{1}{2}(vu_t - uv_t) - u_x v_x + u_{xx}v_{xx} + \frac{u}{3}(uv_x - vu_x) \; , \tag{131}$$

whose Euler-Lagrange equations result in the same Euler-Lagrange equations as (13); by virtue of this property, we can use (131) to explore possible non-trivial continuous symmetries of the spatiotemporal formulation [99]. The problem is that the machinery is typically applied in the context of dynamical systems; not our spatiotemporal formulation. Specifically, the dynamics of $\lambda$ play a key role in relating symmetries of the formal Lagrangian to conservation laws satisfied by $u$. Ibragimov [56, 57, 58] developed conservation laws of arbitrary differential equations by extending the Noether theorem to formal Lagrangians for the extended system $(u, \lambda)$, which can be restricted to the original system provided that it is possible to express the solution of the adjoint variable $\lambda$ in terms of $u$; which if interpreted correctly, is covered by the choice $\lambda = f$. Others have also applied such techniques to dissipative equations [142]. Motivated by continuous families and our variational formulation, these techniques were applied to our system; however, no conclusive or useful results were ever derived. While this is the case, these derivations are included for posterity purposes, so that others may benefit or be able to correctly use them.

The main idea, and reason why we are not sure if these computations are correct, is that it requires the Euler-Lagrange equations (16) to be at least 'nonlinearly self-adjoint'. If the substitution $\lambda = u$ turns an adjoint equation into the original then the equation is said to be *self adjoint.* Other than self-adjointness there are also the notions of quasi-self adjoint equations and nonlinearly adjoint equations refs. [59, 60, 61, 142]. The substitutions that define quasi self-adjoint equations and nonlinearly self-adjoint equations are functions $v = \psi(u)$ and $v = \psi(x^i, u, u^{(n)})$ that transform the adjoint equation into (4) (the term $u^{(n)}$ represents all derivatives up to order $n$). Therefore, making the substitution $\lambda = \psi(x^i, u, u^{(n)})$ and solving the set of equations is similar to the determining equations which produced the generators of the Lie algebra of infinitesimal symmetries.

Noether's Theorem tells us that symmetries impart the Euler-Lagrange equations with conservation laws [98]; therefore, if we can derive new non-trivial symmetries of our spatiotemporal system then a conservation law can be derived, perhaps explaining our continuous families. We already know the equations with which we intend to work with so we may begin after introducing some notation. In order to derive conserved quantities using the machinery of [57] we first need to find the vector fields that span the Lie algebra of infinitesimal symmetries of our equations. Our description follows Theorem 2.36 from [99] in terms of notation.

To begin, take an arbitrary vector field defined on the space $X \times U$ which contains the independent variables $x^i$ and dependent variables $u^\alpha$ such that

$$\mathbf{v} = \xi^i(x^i, u^\alpha)\frac{\partial}{\partial x^i} + \psi_\alpha(x^i, u^\alpha)\frac{\partial}{\partial u^\alpha}\,, \tag{132}$$

where summation is implied by repeated indices. To create a vector field applicable to our equations, we need to "prolong" (132) or perform a "jet prolongation" [86] to the jet space of the same order as our equations, $n$. Informally this just means extending the definition (132) to the same order as the equations being studied. The general formula for the prolongation to the $n$th jet bundle is

$$\mathrm{pr}^{(n)}v = v + \psi_\alpha^J \frac{\partial}{\partial u_J^\alpha}\,, \tag{133}$$

where the coefficients $\psi_\alpha^J$ are given by

$$\psi_\alpha^J(x, u^{(n)} = D_J(\psi_\alpha - \xi^i u_i^\alpha) + \xi^i u_{J,i}^\alpha. \tag{134}$$

We can now begin to apply this to our equation of interest, the Kuramoto-Sivashinsky equation, (4). Starting with the prolongation of the general vector field; we need the fourth prolongation which seems like a lot of work (there is a coefficient for every combination of partial derivatives, and each higher order coefficient becomes more involved due to increased numbers of differentiation operations). Luckily, we already know that we are going to apply the vector field to the Kuramoto-Sivashinsky equation such that instead of calculating all $2 + 4 + 8 + 16$ jet prolongation coefficients (all combinations of $t, x$ derivatives of order one, two, three and four) we only need the coefficients which accompany vectors $\frac{\partial}{\partial u_J}$ which appear in the Kuramoto-Sivashinsky equation. Namely, $\{\psi^J\} = \{\psi^t, \psi^x, \psi^{xx}, \psi^{xxxx}\}$, which in turn creates the vector field specific to the Kuramoto-Sivashinsky equation

$$\begin{aligned}\mathrm{pr}\, v^{(4)} &= \epsilon(x,t,u)\frac{\partial}{\partial x} + \tau(x,t,u)\frac{\partial}{\partial t} + \psi(x,t,u)\frac{\partial}{\partial u} + \psi^t(x,t,u^{(1)})\frac{\partial}{\partial u_t} \\ &+ \psi^x(x,t,u^{(1)})\frac{\partial}{\partial u_x} + \psi^{xx}(x,t,u^{(2)})\frac{\partial}{\partial u_{xx}} + \psi^{xxxx}(x,t,u^{(4)})\frac{\partial}{\partial u_{xxxx}}\,. \quad (135)\end{aligned}$$

All other higher order terms will annihilate when acting on the Kuramoto-Sivashinsky equation. Note that the higher the "order" of the coefficient, the higher the order of the jet bundle that the coefficients depend on. Now that we have the general form of the vector field we can begin to derive the *infinitesimal generators* which span the Lie algebra. To accomplish this, we will derive the *determining equations* which are produced by applying (132) to the system of differential equations and equating to zero, that is

$$\mathrm{pr}\, v^{(4)}(G(u^{(\alpha)}(x,t), u^{(\alpha)}_{(1)}(x,t), \ldots, u^{(\alpha)}_{(n)}(x,t))) = 0\,. \quad (136)$$

Performing this operation yields

$$\psi^t + \psi^{xx} + \psi^{xxxx} + u\psi_x + \psi u_x = 0\,. \quad (137)$$

We finally are forced to derive the coefficients $\psi^J$ and to include as many details as possible we will write the exact formulas needed to derive them as well as the long form expressions that they are equal to

$$\begin{aligned}
\psi^t &= D_t(\psi(x,t,u) - \epsilon(x,t,u)u_x - \tau(x,t,u)u_t) + \tau(x,t,u)u_t t + \epsilon(x,t,u)u_x t \\
\psi^x &= D_x(\psi(x,t,u) - \epsilon(x,t,u)u_x - \tau(x,t,u)u_t) + \tau(x,t,u)u_t x + \epsilon(x,t,u)u_x x \\
\psi^{xx} &= D_x^2(\psi(x,t,u) - \epsilon(x,t,u)u_x - \tau(x,t,u)u_t) + \tau(x,t,u)u_{txx} + \epsilon(x,t,u)u_{xxx} \\
\psi^{xxxx} &= D_x^4(\psi(x,t,u) - \epsilon(x,t,u)u_x - \tau(x,t,u)u_t) + \tau(x,t,u)u_{txxx} + \epsilon(x,t,u)u_{xxxx}\,, \quad (138)
\end{aligned}$$

the long form expressions from each of these are given by substitution into (139). Separating the terms by coefficients of monomials yields the determining equations, as previously mentioned

$$
\begin{aligned}
\psi^t &= u_t^2\left(-\tau_u\right)-\tau_t u_t-u_t u_x \epsilon_u-\epsilon_t u_x+u_t \psi_u+\psi_t \\
\psi^x &= -u_t \tau_u u_x-u_t \tau_x+u_x^2\left(-\epsilon_u\right)-u_x \epsilon_x+u_x \psi_u+\psi_x \\
\psi^{\mathrm{xx}} &= -u_t u_x^2 \tau_{\mathrm{uu}}-2 u_t u_x \tau_{\mathrm{xu}}-u_t \tau_u u_{\mathrm{xx}} \\
&- u_t \tau_{\mathrm{xx}}+u_x^3\left(-\epsilon_{\mathrm{uu}}\right)+u_x^2 \psi_{\mathrm{uu}} \\
&- 2 \tau_u u_x u_{\mathrm{xt}}-2 u_{\mathrm{xt}} \tau_x-2 u_x^2 \epsilon_{\mathrm{xu}} \\
&+ 2 u_x \psi_{\mathrm{xu}}-3 u_x u_{\mathrm{xx}} \epsilon_u-u_x \epsilon_{\mathrm{xx}} \\
&- 2 u_{\mathrm{xx}} \epsilon_x+u_{\mathrm{xx}} \psi_u+\psi_{\mathrm{xx}} \\
\psi^{\mathrm{xxxx}} &= -4 u_t u_x u_{\mathrm{xxx}} \tau_{\mathrm{uu}}-3 u_t u_{\mathrm{xx}}^2 \tau_{\mathrm{uu}}-6 u_t u_x^2 u_{\mathrm{xx}} \tau_{\mathrm{uuu}}-u_t u_x^4 \tau_{\mathrm{uuuu}} \\
&- 12 u_t u_x u_{\mathrm{xx}} \tau_{\mathrm{xuu}}-4 u_t u_x^3 \tau_{\mathrm{xuuu}}-6 u_t u_x^2 \tau_{\mathrm{xxuu}}-4 u_t u_x \tau_{\mathrm{xxxu}}-4 u_t u_{\mathrm{xxx}} \tau_{\mathrm{xu}} \\
&- 6 u_t u_{\mathrm{xx}} \tau_{\mathrm{xxu}}-u_t \tau_u u_{\mathrm{xxxx}}-u_t \tau_{\mathrm{xxxx}}-12 u_x u_{\mathrm{xt}} u_{\mathrm{xx}} \tau_{\mathrm{uu}} \\
&- 15 u_x u_{\mathrm{xx}}^2 \epsilon_{\mathrm{uu}}-6 u_x^2 u_{\mathrm{xxt}} \tau_{\mathrm{uu}}-10 u_x^2 u_{\mathrm{xxx}} \epsilon_{\mathrm{uu}}+4 u_x u_{\mathrm{xxx}} \psi_{\mathrm{uu}} \\
&+ 3 u_{\mathrm{xx}}^2 \psi_{\mathrm{uu}}-4 u_x^3 u_{\mathrm{xt}} \tau_{\mathrm{uuu}}-10 u_x^3 u_{\mathrm{xx}} \epsilon_{\mathrm{uuu}}+6 u_x^2 u_{\mathrm{xx}} \psi_{\mathrm{uuu}} \\
&+ u_x^5\left(-\epsilon_{\mathrm{uuuu}}\right)+u_x^4 \psi_{\mathrm{uuuu}}-12 u_x^2 u_{\mathrm{xt}} \tau_{\mathrm{xuu}}-12 u_x u_{\mathrm{xt}} \tau_{\mathrm{xxu}} \\
&- 12 u_x u_{\mathrm{xxt}} \tau_{\mathrm{xu}}-16 u_x u_{\mathrm{xxx}} \epsilon_{\mathrm{xu}}-24 u_x^2 u_{\mathrm{xx}} \epsilon_{\mathrm{xuu}}+12 u_x u_{\mathrm{xx}} \psi_{\mathrm{xuu}} \\
&- 4 u_x^4 \epsilon_{\mathrm{xuuu}}+4 u_x^3 \psi_{\mathrm{xuuu}}-18 u_x u_{\mathrm{xx}} \epsilon_{\mathrm{xxu}}-6 u_x^3 \epsilon_{\mathrm{xxuu}}+6 u_x^2 \psi_{\mathrm{xxuu}} \\
&- 4 \tau_u u_x u_{\mathrm{xxxt}}-4 u_{\mathrm{xxxt}} \tau_x-4 u_x^2 \epsilon_{\mathrm{xxxu}}+4 u_x \psi_{\mathrm{xxxu}}-5 u_x u_{\mathrm{xxxx}} \epsilon_u-u_x \epsilon_{\mathrm{xxxx}} \\
&- 4 u_{\mathrm{xxxx}} \epsilon_x-12 u_{\mathrm{xt}} u_{\mathrm{xx}} \tau_{\mathrm{xu}}-4 \tau_u u_{\mathrm{xt}} u_{\mathrm{xxx}}-4 u_{\mathrm{xt}} \tau_{\mathrm{xxx}} \\
&- 12 u_{\mathrm{xx}}^2 \epsilon_{\mathrm{xu}}+4 u_{\mathrm{xxx}} \psi_{\mathrm{xu}}-6 \tau_u u_{\mathrm{xx}} u_{\mathrm{xxt}}-6 u_{\mathrm{xxt}} \tau_{\mathrm{xx}} \\
&+ 6 u_{\mathrm{xx}} \psi_{\mathrm{xxu}}-10 u_{\mathrm{xx}} u_{\mathrm{xxx}} \epsilon_u-6 u_{\mathrm{xxx}} \epsilon_{\mathrm{xx}}-4 u_{\mathrm{xx}} \epsilon_{\mathrm{xxx}} \\
&+ u_{\mathrm{xxxx}} \psi_u+\psi_{\mathrm{xxxx}} \qquad (139)
\end{aligned}
$$

$$\psi_t + \psi_{\mathrm{xx}} + \psi_{\mathrm{xxxx}} = 0$$
$$-4\tau_x = 0$$
$$-6\tau_{\mathrm{xx}} = 0$$
$$-2\tau_x - 4\tau_{\mathrm{xxx}} = 0$$
$$-4\epsilon_x + \tau_t + \tau_{\mathrm{xx}} + \tau_{\mathrm{xxxx}} = 0$$
$$4\psi_{\mathrm{xu}} - 6\epsilon_{\mathrm{xx}} = 0$$
$$-4\tau_u = 0$$
$$4\tau_{\mathrm{xu}} = 0$$
$$-2\epsilon_x - 4\epsilon_{\mathrm{xxx}} + \tau_t + \tau_{\mathrm{xx}} + \tau_{\mathrm{xxxx}} + 6\psi_{\mathrm{xxu}} = 0$$
$$-6\tau_u = 0$$
$$-12\tau_{\mathrm{xu}} = 0$$
$$6\tau_{\mathrm{xxu}} = 0$$
$$4\tau_{\mathrm{xu}} - 10\epsilon_u = 0$$
$$-12\epsilon_{\mathrm{xu}} + 6\tau_{\mathrm{xxu}} + 3\psi_{\mathrm{uu}} = 0$$
$$3\tau_{\mathrm{uu}} = 0$$
$$3\tau_{\mathrm{uu}} = 0$$

$$\psi - \epsilon_t - \epsilon_{\mathrm{xx}} - \epsilon_{\mathrm{xxxx}} + 2\psi_{\mathrm{xu}} + 4\psi_{\mathrm{xxxu}} = 0$$
$$-4\tau_u = 0$$
$$-12\tau_{\mathrm{xu}} = 0$$
$$-2\tau_u - 12\tau_{\mathrm{xxu}} = 0$$
$$-4\epsilon_u + 2\tau_{\mathrm{xu}} + 4\tau_{\mathrm{xxxu}} = 0$$
$$4\psi_{\mathrm{uu}} - 16\epsilon_{\mathrm{xu}} = 0$$
$$4\tau_{\mathrm{uu}} = 0$$
$$-2\epsilon_u - 18\epsilon_{\mathrm{xxu}} + 2\tau_{\mathrm{xu}} + 4\tau_{\mathrm{xxxu}} + 12\psi_{\mathrm{xuu}} = 0$$
$$-12\tau_{\mathrm{uu}} = 0$$
$$12\tau_{\mathrm{xuu}} = 0$$
$$4\tau_{\mathrm{uu}} = 0$$
$$12\tau_{\mathrm{xuu}} - 15\epsilon_{\mathrm{uu}} = 0$$
$$-2\epsilon_{\mathrm{xu}} - 4\epsilon_{\mathrm{xxxu}} + \psi_{\mathrm{uu}} + 6\psi_{\mathrm{xxuu}} = 0$$
$$-6\tau_{\mathrm{uu}} = 0$$
$$-12\tau_{\mathrm{xuu}} = 0$$

(140)

$$\begin{array}{cc}
 & \psi_x = 0 \\
\tau_{\text{uu}} + 6\tau_{\text{xxuu}} = 0 & \tau_x = 0 \\
-10\epsilon_{\text{uu}} = 0 & \tau_x = 0 \\
-24\epsilon_{\text{xuu}} + \tau_{\text{uu}} + 6\tau_{\text{xxuu}} + 6\psi_{\text{uuu}} = 0 & -\epsilon_x + \tau_t + \tau_{\text{xx}} + \tau_{\text{xxxx}} = 0 \\
6\tau_{\text{uuu}} = 0 & 4\tau_{\text{xu}} = 0 \\
6\tau_{\text{uuu}} = 0 & 6\tau_{\text{xxu}} = 0 \\
-\epsilon_{\text{uu}} - 6\epsilon_{\text{xxuu}} + 4\psi_{\text{xuuu}} = 0 & 3\tau_{\text{uu}} = 0 \\
-4\tau_{\text{uuu}} = 0 & 2\tau_{\text{xu}} + 4\tau_{\text{xxxu}} = 0 \\
4\tau_{\text{xuuu}} = 0 & 4\tau_{\text{uu}} = 0 \\
4\tau_{\text{xuuu}} - 10\epsilon_{\text{uuu}} = 0 & 12\tau_{\text{xuu}} = 0 \\
\psi_{\text{uuuu}} - 4\epsilon_{\text{xuuu}} = 0 & \tau_{\text{uu}} + 6\tau_{\text{xxuu}} = 0 \\
\tau_{\text{uuuu}} = 0 & 6\tau_{\text{uuu}} = 0 \\
\tau_{\text{uuuu}} = 0 & 4\tau_{\text{xuuu}} = 0 \\
-\epsilon_{\text{uuuu}} = 0 & \tau_{\text{uuuu}} = 0 \\
 & \tau_x = 0
\end{array} \tag{141}$$

While initially intimidating, these equations can be solved by noticing the lower order equations such as $\tau_x = \tau_u = 0$ which means that $\tau$ can only be a function of $t$. Following this reasoning we find that in fact

$$\begin{aligned}
\tau(x,t,u) &= \tau = c_1 \\
\epsilon(x,t,u) &= \epsilon(t) = c_3 t + c_1 \\
\psi(x,t,u) &= \psi = c_3 \, ,
\end{aligned} \tag{142}$$

such that the Lie algebra of infinitesimal symmetries is spanned by

$$\begin{aligned}
v_1 &= \partial_x \\
v_2 &= \partial_t \\
v_3 &= t\partial_x + \partial_u \, ,
\end{aligned} \tag{143}$$

which are the generators of space and time translations, and Galilean transformations. This is not surprising, as these symmetries have been previously described [12]. The reason why this calculation was pursued in the first place was to see if there were any "hidden" continuous symmetries afforded by a spatiotemporal formulation that were not present when the problem was viewed as a dynamical system. This is true for *discrete symmetries*, but unfortunately not so for continuous symmetries. To carry the calculation through to finality we need to know the prolongations of (143) and their extensions to the adjoint variables, as the Lie algebra needs to be extended to account for both Euler-Lagrange equations.

The prolongations of (143) result in

$$\begin{aligned} \mathrm{pr}\, v_1 &= y_1 = \partial_x \\ \mathrm{pr}\, v_2 &= y_2 = \partial_t \\ \mathrm{pr}\, v_3 &= y_3 = \partial_x + \partial_u - u_x \partial_{u_t}\,. \end{aligned} \tag{144}$$

We can now derive the extended versions of (144) such that we can apply them to the formal Lagrangian (131). Once again we deploy the machinery of Ibragimov to extend (144) to the adjoint variables. Unfortunately it seems that the symmetries were too simple to actually have extensions to the adjoint variables, but we can still go forward with the conservation law calculations regardless. Both Ibragimov [57] and Olver [99] work through the proof that there is a conserved vector (as Ibragimov names it) such that its divergence provides a conservation law (technically infinite number of conservation laws because they are equations involving PDE solutions). The components of the conserved vector (one for each independent variable) are given by

$$\begin{aligned} C^i &= \xi^i \mathcal{L} + W^\alpha [\frac{\partial \mathcal{L}}{\partial u^\alpha_i} - D_j \frac{\partial \mathcal{L}}{\partial u^\alpha_{ij}} + D_j D_k \frac{\partial \mathcal{L}}{\partial u^\alpha_{ijk}} - + D_j D_k D_l \frac{\partial \mathcal{L}}{\partial u^\alpha_{ijkl}}] \\ &+ D_j(W^\alpha)[\frac{\partial \mathcal{L}}{\partial u^\alpha_{ij}} - D_k \frac{\partial \mathcal{L}}{\partial u^\alpha_{ijk}} + D_k D_l \frac{\partial \mathcal{L}}{\partial u^\alpha_{ijkl}}] \\ &+ D_j D_k(W^\alpha)[\frac{\partial \mathcal{L}}{\partial u^\alpha_{ijk}} - D_l \frac{\partial \mathcal{L}}{\partial u^\alpha_{ijkl}} + D_k D_j \frac{\partial \mathcal{L}}{\partial u^\alpha_{ikj}} - + D_k D_j D_l \frac{\partial \mathcal{L}}{\partial u^\alpha_{ikjl}}] \\ &+ D_j D_k D_l(W^\alpha)[\frac{\partial \mathcal{L}}{\partial u^\alpha_{ijkl}}]\,, \end{aligned} \tag{145}$$

where the equation has been extended to include all possible non-zero terms in the context of the Kuramoto-Sivashinsky equation, $W^\alpha$ is shorthand for $\psi^\alpha + \xi^i u_i^\alpha$. Applying this to our generators yields one unique conservation law which we shall now detail.

For the Galilean transformation generator $\operatorname{pr} v_3 = t\partial_x + \partial_u - u_x\partial_{u_t}$ the components equal

$$\begin{aligned}
C^x &= t * \mathcal{L} + W[\frac{\partial \mathcal{L}}{\partial u_x} - D_x \frac{\partial \mathcal{L}}{\partial u_{xx}} - D_x^3 \frac{\partial \mathcal{L}}{\partial u_{xxxx}}] \\
&+ D_x(W)[\frac{\partial \mathcal{L}}{\partial u_{xx}} + D_x^2 \frac{\partial \mathcal{L}}{\partial u_{xxxx}}] \\
&+ D_x^2(W)[-D_x \frac{\partial \mathcal{L}}{\partial u_{xxxx}}] \\
&+ D_x^3(W)[\frac{\partial \mathcal{L}}{\partial u_{xxxx}}] \\
C^t &= 0 * \mathcal{L} + (1 - tu_x)[\frac{\partial \mathcal{L}}{\partial u_t}] \, . \qquad (146)
\end{aligned}$$

Both expressions simplify to

$$\begin{aligned}
C^x &= t(u_t v + u_x v_x + u_x v_{xxx} + u_{xxx} v_x + u_{xx} v_{xx}) + uv - v_x - v_{xxx} \\
C^t &= (1 - tu_x)v \, , \qquad (147)
\end{aligned}$$

such that the conservation law is given by the divergence

$$\begin{aligned}
D_x(C^x) + D_t(C^t) &= 0 \\
&= v_t - vu_x - u_x v_t t + uv_x + vu_x - v_{xx} - v_{xxxx} \\
&\quad + t(v_x(u_t + u_{xx} + u_{xxxx}) + u_x(v_{xx} + v_{xxxx}) + 2D_x(u_{xx}v_{xx})) \\
&= t(v_x(u_t + u_{xx} + u_{xxxx}) + u_x(-v_t + v_{xx} + v_{xxxx}) + 2D_x(u_{xx}v_{xx})) \\
&= 2D_x(u_{xx}v_{xx}) \, . \qquad (148)
\end{aligned}$$

This analysis is only relevant if there are non-trivial solutions to the adjoint equation, because otherwise one does not know how to evaluate (148). One component which may contain mistakes is the derivation of solutions to the adjoint equation; in this context only allowed to be constant solutions by virtue of the form of $\psi$ implying that the conserved vector (148) is unfortunately a trivial conservation law. There are a number of reasons why we believe this analysis fails to yield anything useful, the most obvious being that the Lie algebra of infinitesimal symmetries seems too simple to be correct.

We still want a method of analysis for our variational formulation. There are two avenues of pursuit towards this endeavor. The first is known as Hill's formula [5]. It discusses how the determinant of a finite matrix of the Hessian of an action functional of a discrete Lagrangian system, can be related to the eigenvalues of the monodromy matrix corresponding to a critical point of said action functional, which represents a critical point. Now, how to extend this to our formal Lagrangian, or if it even can be extended, is still unknown.

## References

[1] Armbruster, D., Guckenheimer, J., and Holmes, P., "Kuramoto-Sivashinsky dynamics on the center-unstable manifold," *SIAM J. Appl. Math.*, vol. 49, pp. 676–691, 1989.

[2] Artuso, R., Aurell, E., and Cvitanović, P., "Recycling of strange sets: I. Cycle expansions," *Nonlinearity*, vol. 3, pp. 325–359, 1990.

[3] Auerbach, D., Cvitanović, P., Eckmann, J.-P., Gunaratne, G., and Procaccia, I., "Exploring chaotic motion through periodic orbits," *Phys. Rev. Lett.*, vol. 58, pp. 2387–2389, 1987.

[4] Baker, A. H., Jessup, E. R., and Manteuffel, T., "A technique for accelerating the convergence of restarted GMRES," *SIAM J. Matrix Anal. Apply.*, 2005.

[5] Bolotin, S. V. and Treschev, D. V., "Hill's formula," *Russ. Math. Surv.*, vol. 65, p. 191, 2010.

[6] Boyd, J. P., *Chebyshev and Fourier Spectral Methods.* New York: Dover, 2 ed., 2000.

[7] Boyd, S. and Vandenberghe, L., *Convex Optimization.* Cambridge: Cambridge Univ. Press, 2004.

[8] Brandenberger, R. H., "Topological defects and structure formation," *Int. J. Mod. Phys. A.*, vol. 9, pp. 2117–2189, 1994.

[9] Brown, H. S. and Kevrekidis, I. G., "Modulated traveling waves for the Kuramoto-Sivashinsky equation," in *Pattern Formation: Symmetry Methods and Applications* (Benest, D. and Froeschlé, C., eds.), pp. 45–66, Providence, RI: AMS, 1996.

[10] Brown, P. N., "A local convergence theory for combined inexact-Newton/finite-difference projection methods," *SIAM J. Numer. Anal.*, 1987.

[11] Budanur, N. B. and Cvitanović, P., "Unstable manifolds of relative periodic orbits in the symmetry-reduced state space of the Kuramoto-Sivashinsky system," *J. Stat. Phys.*, vol. 167, pp. 636–655, 2015.

[12] Budanur, N. B., Cvitanović, P., Davidchack, R. L., and Siminos, E., "Reduction of the SO(2) symmetry for spatially extended dynamical systems," *Phys. Rev. Lett.*, vol. 114, p. 084102, 2015.

[13] Budanur, N. B., Short, K. Y., Farazmand, M., Willis, A. P., and Cvitanović, P., "Relative periodic orbits form the backbone of turbulent pipe flow," *J. Fluid Mech.*, vol. 833, pp. 274–301, 2017.

[14] Byrd, R., Lu, P., Nocedal, J., and Zhu, C., "A limited memory algorithm for bound constrained optimization," *SIAM J. Sci. and Stat. Comput.*, 1995.

[15] Canuto, C., Hussaini, M. Y., Quateroni, A., and Zhang, T. A., *Spectral Methods in Fluid Dynamics.* New York: Springer, 1988.

[16] Canuto, C., Hussaini, M. Y., Quateroni, A., and Zhang, T. A., *Spectral Methods: Fundamentals in Single Domains.* New York: Springer, 2006.

[17] Carleson, L., "On convergence and growth of partial sums of Fourier series.," *Acta Math*, 1966.

[18] Chandler, G. J. and Kerswell, R. R., "Invariant recurrent solutions embedded in a turbulent two-dimensional Kolmogorov flow," *J. Fluid Mech.*, vol. 722, pp. 554–595, 2013.

[19] Christiansen, F., Cvitanović, P., and Putkaradze, V., "Spatiotemporal chaos in terms of unstable recurrent patterns," *Nonlinearity*, vol. 10, pp. 55–70, 1997.

[20] Chu, K. T., "A direct matrix method for computing analytical Jacobians of discretized nonlinear integro-differential equations," *J. Comput. Phys.*, vol. 228, pp. 5526–5538, 2009.

[21] Constantin, P., Foias, C., Nicolaenko, B., and Témam, R., *Integral Manifolds and Inertial Manifolds for Dissipative Partial Differential Equations.* New York: Springer, 1989.

[22] Cross, M. C. and Hohenberg, P. C., "Pattern formation outside of equilibrium," *Rev. Mod. Phys.*, vol. 65, pp. 851–1112, 1993.

[23] Cvitanović, P., "Recurrent flows: The clockwork behind turbulence," *J. Fluid Mech. Focus Fluids*, vol. 726, pp. 1–4, 2013.

[24] Cvitanović, P., Artuso, R., Mainieri, R., Tanner, G., and Vattay, G., *Chaos: Classical and Quantum.* Copenhagen: Niels Bohr Inst., 2018.

[25] Cvitanović, P., Davidchack, R. L., and Siminos, E., "On the state space geometry of the Kuramoto-Sivashinsky flow in a periodic domain," *SIAM J. Appl. Dyn. Syst.*, vol. 9, pp. 1–33, 2010.

[26] de Sturler, E., "Truncation strategies for optimal Krylov subspaces methods," *SIAM J. Numer. Anal.*, 1999.

[27] Dennis, J. E. and Schnabel, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Philadelphia: SIAM, 1996.

[28] Ding, X., Chaté, H., Cvitanović, P., Siminos, E., and Takeuchi, K. A., "Estimating the dimension of the inertial manifold from unstable periodic orbits," *Phys. Rev. Lett.*, vol. 117, p. 024101, 2016.

[29] Ding, X. and Cvitanović, P., "Periodic eigendecomposition and its application in Kuramoto-Sivashinsky system," *SIAM J. Appl. Dyn. Syst.*, vol. 15, pp. 1434–1454, 2016.

[30] Dong, C. and Lan, Y., "Organization of spatially periodic solutions of the steady Kuramoto-Sivashinsky equation," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, pp. 2140–2153, 2014.

[31] Duguet, Y., Pringle, C. C. T., and Kerswell, R. R., "Relative periodic orbits in transitional pipe flow," *Phys. Fluids*, vol. 20, p. 114102, 2008.

[32] Egolf, D. A. and Greenside, H. S., "Relation between fractal dimension and spatial correlation length for extensive chaos," *Nature*, vol. 369, pp. 129–131, 1994.

[33] Farazmand, M., "An adjoint-based approach for finding invariant solutions of Navier-Stokes equations," *J. Fluid M.*, vol. 795, pp. 278–312, 2016.

[34] Fernández-Nieves, A., Vitelli, V., Utada, A. S., Link, D. R., Márquez, M., Nelson, D. R., and Weitz, D. A., "Novel defect structure in nematic liquid crystal shells," *Phys. Rev. Lett.*, vol. 99, p. 157801, 2007.

[35] Fletcher, R., "Conjugate gradient methods for indefinite systems," *Lecture Notes in Math.*, pp. 73–89, 1976.

[36] Foias, C., Nicolaenko, B., Sell, G. R., and Témam, R., "Inertial manifold for the Kuramoto-Sivashinsky equation," *C. R. Acad. Sci. Paris, Ser. I*, vol. 301, pp. 285–288, 1985.

[37] Foias, C., Nicolaenko, B., Sell, G. R., and Témam, R., "Inertial manifolds for the Kuramoto-Sivashinsky equation and an estimate of their lowest dimension," *J. Math. Pure Appl.*, vol. 67, pp. 197–226, 1988.

[38] Foias, C., Sell, G. R., and Témam, R., "Inertial manifolds for nonlinear evolutionary equations," *J. Diff. Equ.*, vol. 73, pp. 309–353, 1988.

[39] Fong, D. C.-L. and Saunders, M. A., "LSMR: an iterative algorithm for sparse least-squares problems," *SIAM J. Sci. Comput.*, 2011.

[40] Forster, D., Nelson, D. R., and Stephen, M. J., "Large-distance and long-time properties of a randomly stirred fluid," *Phys. Rev. A*, 1977.

[41] Fox, D. G. and Orszag, S. A., "Pseudospectral approximation to two-dimensional turbulence," *J. Comput. Phys.*, vol. 11, pp. 612–619, 1973.

[42] Freund, R. W. and Nachtigal, N. M., "QMR: a quasi-minimal residual method for non-Hermitian linear systems," *Numer. Math.*, 1991.

[43] Gerlach, E., Eggl, S., and Skokos, C., "Efficient integration of the variational equations of multi-dimensional Hamiltonian systems: Application to the Fermi-Pasta-Ulam lattice," *Int. J. Bifurcation Chaos*, vol. 22, p. 1250216, 2012.

[44] Giacomelli, G., Lepri, S., and Politi, A., "Statistical properties of bidimensional patterns generated from delayed and extended maps," *Phys. Rev. E*, vol. 51, pp. 3939–3944, 1995.

[45] Giannakis, D., Ourmazd, A., Slawinska, J., and Zhao, Z., "Spatiotemporal pattern extraction by spectral analysis of vector-valued observables," *J. Nonlin. Sci.*, pp. 1–61, 2019.

[46] Gibson, J. F., "Channelflow: A spectral Navier-Stokes simulator in C++," tech. rep., U. New Hampshire, 2019. `Channelflow.org`.

[47] Gibson, J. F., Halcrow, J., and Cvitanović, P., "Visualizing the geometry of state-space in plane Couette flow," *J. Fluid Mech.*, vol. 611, pp. 107–130, 2008.

[48] Gibson, J. F., Halcrow, J., and Cvitanović, P., "Equilibrium and traveling-wave solutions of plane Couette flow," *J. Fluid Mech.*, vol. 638, pp. 243–266, 2009.

[49] Ginelli, F., Chaté, H., Livi, R., and Politi, A., "Covariant Lyapunov vectors," *J. Phys. A*, vol. 46, p. 254005, 2013.

[50] Ginelli, F., Poggi, P., Turchi, A., Chaté, H., Livi, R., and Politi, A., "Characterizing dynamics with covariant Lyapunov vectors," *Phys. Rev. Lett.*, vol. 99, p. 130601, 2007.

[51] Greene, J. M. and Kim, J.-S., "The steady states of the Kuramoto-Sivashinsky equation," *Physica D*, vol. 33, pp. 99–120, 1988.

[52] Gudorf, M., "orbithunter : framework for nonlinear chaotic PDEs," 2020.

[53] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'io, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E., "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sept. 2020.

[54] Harter, W. G., *Principles of Symmetry, Dynamics, and Spectroscopy.* New York: Wiley, 1993.

[55] Hestenes, M. R. and Stiefel, E., "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards*, 1952.

[56] Ibragimov, N. H., "Integrating factors, adjoint equations and Lagrangians," *J. Math. Anal. Appl.*, vol. 318, pp. 742–757, 2006.

[57] Ibragimov, N. H., "A new conservation theorem," *J. Math. Anal. Appl.*, vol. 333, pp. 311–328, 2007.

[58] Ibragimov, N. H., "Quasi-self-adjoint differential equations," *Arch. ALGA*, vol. 4, p. 55–60, 2007.

[59] Ibragimov, N. H., "Nonlinear self-adjointness and conservation laws," *J. Phys. A*, vol. 44, p. 432002, 2011.

[60] Ibragimov, N. H., "Nonlinear self-adjointness in constructing conservation laws," *Arch. ALGA*, vol. 7, 2011.

[61] Ibragimov, N. H., "Conservation laws and non-invariant solutions of anisotropic wave equations with a source," *Nonlinear Anal. Real World Appl.*, vol. 40, pp. 82–94, 2018.

[62] Ibragimov, N. H. and Kolsrud, T., "Lagrangian approach to evolution equations: symmetries and conservation laws," *Nonlin. Dyn.*, vol. 36, pp. 29–40, 2004.

[63] Jimenez, J. and Simens, M. P., "Low-dimensional dynamics of a turbulent wall flow," *J. Fluid. Mech*, vol. 435, pp. 81–91, 2001.

[64] Jolly, M. S., Kevrekidis, I. G., and Titi, E. S., "Approximate inertial manifolds for the Kuramoto-Sivashinsky equation: Analysis and computations," *Physica D*, vol. 44, pp. 38–60, 1990.

[65] Jones, J., Troy, W. C., and MacGillivary, A. D., "Steady solutions of the Kuramoto-Sivashinsky equation for small wave speed," *J. Diff. Eqn.*, vol. 96, pp. 28–55, 1992.

[66] Kardar, M., Parisi, G., and Zhang, Y., "Dynamic scaling of growing interfaces," *Phys. Rev. Lett.*, 1985.

[67] Kassam, A.-K. and Trefethen, L. N., "Fourth-order time-stepping for stiff PDEs," *SIAM J. Sci. Comput.*, vol. 26, pp. 1214–1233, 2005.

[68] Kerr, R. M., "Higher-order derivative correlations and the alignment of small-scale structure in isotropic numerical turbulence," *J. Fluid. Mech.*, vol. 153, pp. 31–58, 1985.

[69] Kevrekidis, I. G., Nicolaenko, B., and Scovel, J. C., "Back in the saddle again: a computer assisted study of the Kuramoto-Sivashinsky equation," *SIAM J. Appl. Math.*, vol. 50, pp. 760–790, 1990.

[70] Kleman, M., "Defects in liquid crystals," *Rep. Prog. Phys.*, vol. 52, pp. 555–654, 1989.

[71] Knoll, D. and Keyes, D., "Jacobian-free Newton-Krylov methods: a survey of approaches and applications," *J. Comput. Phys.*, vol. 193, pp. 357–397, 2004.

[72] Kraus, M. and Maj, O., "Variational integrators for nonvariational partial differential equations," *Physica D*, vol. 310, pp. 37–71, 2015.

[73] Krist, S. and Zang, T. A., "Numerical simulation of channel flow transition: Resolution requirements and the structure of hairpin vortices," tech. rep., NASA, 1987.

[74] Kuramoto, Y. and Tsuzuki, T., "Persistent propagation of concentration waves in dissipative media far from thermal equilibrium," *Progr. Theor. Phys.*, vol. 55, pp. 356–369, 1976.

[75] Lan, Y., *Dynamical Systems Approach to* $1-d$ *Spatiotemporal Chaos – A Cyclist's View.* PhD thesis, School of Physics, Georgia Inst. of Technology, Atlanta, 2004.

[76] Lan, Y., Chandre, C., and Cvitanović, P., "Variational method for locating invariant tori," *Phys. Rev. E*, vol. 74, p. 046206, 2006.

[77] Lan, Y. and Cvitanović, P., "Variational method for finding periodic orbits in a general flow," *Phys. Rev. E*, vol. 69, p. 016217, 2004.

[78] Lan, Y. and Cvitanović, P., "Unstable recurrent patterns in Kuramoto-Sivashinsky dynamics," *Phys. Rev. E*, vol. 78, p. 026208, 2008.

[79] LaQuey, R. E., Mahajan, S. M., Rutherford, P. H., and Tang, W. M., "Nonlinear saturation of the trapped-ion mode," *Phys. Rev. Lett.*, vol. 34, pp. 391–394, 1974.

[80] Lepri, S., Politi, A., and Torcini, A., "Chronotopic Lyapunov analysis. I. A detailed characterization of 1D systems," *J. Stat. Phys.*, vol. 82, pp. 1429–1452, 1996.

[81] Lepri, S., Politi, A., and Torcini, A., "Chronotopic Lyapunov analysis. II. Towards a unified approach," *J. Stat. Phys.*, vol. 88, pp. 31–45, 1997.

[82] López, V., Boyland, P., Heath, M. T., and Moser, R. D., "Relative periodic solutions of the complex Ginzburg-Landau equation," *SIAM J. Appl. Dyn. Syst.*, vol. 4, pp. 1042–1075, 2006.

[83] Lucas, D. and Kerswell, R. R., "Spatiotemporal dynamics in 2D Kolmogorov flow over large domains," *J. Fluid Mech.*, vol. 750, pp. 518–554, 2014.

[84] Lyapunov, A. M., "The general problem of the stability of motion," *Int. J. Control*, vol. 55, pp. 531–534, 1992.

[85] Marsden, J. E. and West, M., "Discrete mechanics and variational integrators," *Acta Numerica*, vol. 10, pp. 357–514, 2001.

[86] Marsden, J., Pekarsky, S., Shkoller, S., and West, M., "Variational methods, multisymplectic geometry and continuum mechanics," *J. Geom. Phys.*, vol. 38, pp. 253–284, 2001.

[87] Mermin, N. D., "The topological theory of defects in ordered media," *Rev. Mod. Phys.*, vol. 51, pp. 591–648, 1979.

[88] Michelson, D., "Steady solutions of the Kuramoto-Sivashinsky equation," *Physica D*, vol. 19, pp. 89–111, 1986.

[89] Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U., "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, pp. 824–827, 2002.

[90] Moehlis, J., Faisst, H., and Eckhardt, B., "Periodic orbits and chaotic sets in a low-dimensional model for shear flows," *SIAM J. Appl. Dyn. Syst.*, vol. 4, pp. 352–376, 2005.

[91] Montigny-Rannou, F., "Effect of "aliasing" on spectral method solution of Navier-Stokes equations," *Rech. Aerosp.*, vol. 2, pp. 365–41, 1982.

[92] Moser, R. D., Moin, P., and A., L., "A spectral numerical method for the Navier-Stokes equations with applications to Taylor-Couette flow," *J. Comput. Phys.*, vol. 52, pp. 524–544, 1983.

[93] Münkel, M. and Kaiser, F., "An intermittency route to chaos via attractor merging in the Laser-Kuramoto-Sivashinsky equation," *Physica D*, 1996.

[94] Nagata, M., "Three-dimensional traveling-wave solutions in plane Couette flow," *Phys. Rev. E*, vol. 55, pp. 2023–2025, 1997.

[95] Nash, S. G., "Newton-type minimization via the Lanczos method.," *SIAM J. Num. Anal.*, 1984.

[96] Nicolaenko, B. and Scheurer, B., "Remarks on the Kuramoto-Sivashinsky equation," *Physica*, 1984.

[97] Nocedal, J. and Wright, S. J., *Numerical Optimization.* New York: Springer, 2 ed., 2006.

[98] Noether, E., "Der Endlichkeitssatz der Invarianten endlicher Gruppen," *Math. Ann.*, vol. 77, pp. 89–92, 1915.

[99] Olver, P. J., *Applications of Lie Groups to Differential Equations.* New York: Springer, 1998.

[100] Orszag, S. A., "Computation of pseudospectral and spectral approximations," *Stud. Appl. Math.*, vol. 51, pp. 253–259, 1972.

[101] Page, J. and Kerswell, R. R., "Searching turbulence for periodic orbits with dynamic mode decomposition," *J. Fluid Mech.*, vol. 886, p. A28, 2020.

[102] Paige, C. C. and Saunders, M. A., "Solution of sparse indefinite systems of linear equations," *SIAM, J. Numer. Anal.*, 1975.

[103] Paige, C. C. and Saunders, M. A., "LSQR: An algorithm for sparse linear equations and sparse least squares," *ACM TOMS*, 1982.

[104] Pesin, Y. B., "Characteristic Lyapunov exponents and smooth ergodic theory," *Russian Math. Surveys*, vol. 32, pp. 55–114, 1977.

[105] Politi, A., Torcini, A., and Lepri, S., "Lyapunov exponents from node-counting arguments," *J. Phys. IV*, vol. 8, p. 263, 1998.

[106] Press, W.H., T. S. A. and Flannery, B. P., *Numerical recipes: the art of scientific computing.* Cambridge University Press, 2007.

[107] Rempel, E. L., Chian, A. C., and Miranda, R. A., "Chaotic saddles at the onset of intermittent spatiotemporal chaos," *Phys. Rev. E*, vol. 76, p. 056217, 2007.

[108] Reynolds, D., Samtaney, R., and Woodward, C., "Operator-based preconditioning of stiff hyperbolic systems," *SIAM J. Sci. Comput.*, vol. 32, pp. 150–170, 2010.

[109] Robinson, J. C., "Inertial manifolds for the Kuramoto-Sivashinsky equation," *Phys. Lett. A*, vol. 184, pp. 190–193, 1994.

[110] Robinson, J. C., "Finite-dimensional behavior in dissipative partial differential equations," *Chaos*, vol. 5, pp. 330–345, 1995.

[111] Saad, Y., *Iterative methods for sparse linear systems.* SIAM, 2nd ed. ed., 2003.

[112] Saad, Y. and Schultz, M. H., "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 856–869, 1986.

[113] Schamel, H. and Elsässer, K., "The application of the spectral methods to nonlinear wave propagation," *J. Comput. Phys.*, vol. 22, pp. 501–516, 1976.

[114] Schmid, P. J., "Dynamic mode decomposition of numerical and experimental data," *J. Fluid Mech.*, vol. 656, pp. 5–28, 2010.

[115] Schmid, P. J., "Application of the dynamic mode decomposition to experimental data," *Exp. Fluids.*, vol. 50, pp. 1123–1130, 2011.

[116] Shlang, T. and Sivashinsky, G. I., "Irregular flow of a liquid film down a vertical column," *J. Physique*, vol. 43, pp. 459–466, 1982.

[117] Sivashinsky, G. I., "Nonlinear analysis of hydrodynamical instability in laminar flames - I. Derivation of basic equations," *Acta Astronaut.*, vol. 4, pp. 1177–1206, 1977.

[118] Sivashinsky, G. I. and Michelson, D. M., "On irregular wavy flow of a liquid down a vertical plane," *Progr. Theor. Phys.*, pp. 2112–2114, 1980.

[119] Sonneveld, P., "Cgs: a fast Lanczos-type solver for nonsymmetric linear systems," *Siam J. Sci. Statist. Comput.*, 1989.

[120] Spalart, P. R., Moser, R. D., and Rogers, M. M., "Spectral methods for the Navier-Stokes equations with one infinite and two periodic directions," *J. Comput. Phys.*, vol. 96, pp. 297–324, 1990.

[121] Steeb, W.-H. and Hardy, Y., *Matrix Calculus and Kronecker Product - A Practical Approach to Linear and Multilinear Algebra.* Singapore: World Scientific, 2 ed., 2011.

[122] Suri, B., Tithof, J., Grigoriev, R. O., and Schatz, M. F., "Unstable equilibria and invariant manifolds in quasi-two-dimensional Kolmogorov-like flow," *Phys. Rev. E*, vol. 98, p. 023105, 2018.

[123] Tadmor, E., "The well-posedness of the Kuramoto-Sivashinsky equation," *SIAM J. Math. Anal.*, 1986.

[124] Tajima, S. and Greenside, H. S., "Microextensive chaos of a spatially extended system," *Phys. Rev. E*, vol. 66, p. 017205, 2002.

[125] Takeuchi, K. A., Yang, H. L., Ginelli, F., Radons, G., and Chaté, H., "Hyperbolic decoupling of tangent space and effective dimension of dissipative systems," *Phys. Rev. E*, vol. 84, p. 046214, 2011.

[126] Témam, R., "Inertial manifolds," *Math. Intelligencer*, vol. 12, pp. 68–74, 1990.

[127] Témam, R., *Infinite-Dimensional Dynamical Systems in Mechanics and Physics.* New York: Springer, 2 ed., 2013.

[128] Trefethen, L. N., *Spectral Methods in MATLAB.* Philadelphia: SIAM, 2000.

[129] Trefethen, L. N. and Bau, D., *Numerical Linear Algebra.* Philadelphia: SIAM, 1997.

[130] Troy, W. C., "The existence of steady solutions of the Kuramoto-Sivashinsky equation," *J. Diff. Eqn.*, vol. 82, pp. 269–313, 1989.

[131] van der Vorst, H. A., "Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems.," *SIAM J. Sci. and Stat. Comput.*, 1992.

[132] van Veen, L., *Computational modelling of bifurcations and instabilities in fluid dynamics*, ch. A brief history of simple invariant solutions in turbulence, pp. 217–231. Springer, Cham, 2019.

[133] Vilenkin, A. and Shellard, E. P. S., *Cosmic Strings and other Topological Defects.* Cambridge UK: Cambridge Univ. Press, 2000.

[134] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[135] Viswanath, D., "Recurrent motions within plane Couette turbulence," *J. Fluid Mech.*, vol. 580, pp. 339–358, 2007.

[136] Waleffe, F., "On a Self-Sustaining Process in shear flows," *Phys. Fluids*, vol. 9, pp. 883–900, 1997.

[137] Waleffe, F., "Exact coherent structures in channel flow," *J. Fluid Mech.*, vol. 435, pp. 93–102, 2001.

[138] Waleffe, F., "Exact coherent structures and their instabilities: Toward a dynamical-system theory of shear turbulence," in *Proceedings of the International Symposium on "Dynamics and Statistics of Coherent Structures in Turbulence: Roles of Elementary Vortices"* (Kida, S., ed.), pp. 115–128, National Center of Sciences, Tokyo, Japan, 2002.

[139] Wang, Q., Gomez, S. A., Blonigan, P. J., Gregory, A. L., and Qian, E. Y., "Towards scalable parallel-in-time turbulent flow simulations," *Phys. Fluids*, vol. 25, p. 110818, 2013.

[140] Wang, Q., Hu, R., and Blonigan, P. J., "Least Squares Shadowing sensitivity analysis of chaotic limit cycle oscillations," *J. Comput. Phys.*, vol. 267, pp. 210–224, 2014.

[141] Wedin, H. and Kerswell, R. R., "Exact coherent structures in pipe flow," *J. Fluid Mech.*, vol. 508, pp. 333–371, 2004.

[142] Wei, L. and Wang, Y., "Symmetry analysis, conserved quantities and applications to a dissipative DGH equation," *J. Diff. Equ.*, vol. 266, pp. 3189–3208, 2019.

[143] Wolfe, C. L. and Samelson, R. M., "An efficient method for recovering Lyapunov vectors from singular vectors," *Tellus A*, vol. 59, pp. 355–366, 2007.

[144] Yang, H. L., Takeuchi, K. A., Ginelli, F., Chaté, H., and Radons, G., "Hyperbolicity and the effective dimension of spatially extended dissipative systems," *Phys. Rev. Lett.*, vol. 102, p. 074102, 2009.

# VITA

Matthew N. Gudorf was born in 1990 in Buffalo, New York and raised in Rochester, Michigan. He obtained a bachelor's degree in 2013 from The University of Michigan, Ann Arbor, specializing in Physics and Mathematical Physics. In August 2015 he enrolled in the PhD. program in the School of Physics at Georgia Institute of Technology where he worked on nonlinear dynamics and chaos until his graduation in late 2020.