Predicting Solar Wind Streams from the Inner-Heliosphere to Earth via Shifted Operator Inference

Opal Issan[a,*], Boris Kramer[a]

[a]*Department of Mechanical and Aerospace Engineering, University of California San Diego, CA, United States*

November 7, 2022

Abstract

Solar wind conditions are predominantly predicted via three-dimensional numerical magnetohydrodynamic (MHD) models. Despite their ability to produce highly accurate predictions, MHD models require computationally intensive high-dimensional simulations. This renders them inadequate for making time-sensitive predictions and for large-ensemble analysis required in uncertainty quantification. This paper presents a new data-driven reduced-order model (ROM) capability for forecasting heliospheric solar wind speeds. Traditional model reduction methods based on Galerkin projection have difficulties with advection-dominated systems—such as solar winds—since they require a large number of basis functions and can become unstable. A core contribution of this work addresses this challenge by extending the non-intrusive operator inference ROM framework to exploit the translational symmetries present in the solar wind caused by the Sun's rotation. The numerical results show that our method can adequately emulate the MHD simulations and is more accurate than a reduced-physics surrogate model, the Heliospheric Upwind Extrapolation model.

Keywords: solar wind modeling, space weather prediction, magnetohydrodynamics, data-driven model reduction, scientific machine learning, operator inference

1. Introduction

Magnetohydrodynamic (MHD) modeling of coronal and interplanetary solar wind can significantly improve the prediction of catastrophic space weather events. Such space weather geomagnetic storms can have detrimental effects on spacecrafts, cause electric power outage, satellite collisions, telecommunication interruption, and expose astronauts to harmful radiation. Very recently, 40 out of 49 of SpaceX's starlink satellites failed to reach their low-Earth orbits presumably due to the effects of a geomagnetic storm around Feb. 2, 2022; the geomagnetic storm increased Earth's upper atmosphere density causing orbital drag [6]. This event further emphasises the need for real-time modeling of solar storms. The most substantial source of space weather events are coronal mass ejections, corotating interaction regions, and high-speed solar wind streams that reach the Earth's magnetosphere. Three-dimensional MHD solar wind models, such as the Magnetohydrodynamics Around a Sphere [55] model, Enlil [40], and the Space Weather Modeling Framework [63], can provide high-fidelity predictions. Apart from providing a global assessment of coronal and heliospheric properties, MHD modeling can connect *in situ* magnetic and plasma observations from one spacecraft to the other, providing crucial support to interplanetary missions [57]. Although MHD models are an important tool in understanding observed coronal and heliospheric dynamics, they require computationally intensive high-dimensional simulations. This renders them infeasible for time-sensitive predictions and large-ensemble methods such as quantifying forecast uncertainty and performing parameter sensitiv-

*Corresponding author
Email address: oissan@ucsd.edu (Opal Issan)

ity analysis. Thus, there is a need for computationally efficient surrogate models that are capable of reproducing MHD results with sufficient fidelity.

The present work addresses this challenge by proposing a new method to derive a data-driven reduced-order model (ROM) for solar wind predictions that particularly focuses on issues that arise from the advection-dominated nature of the problem. The proposed method efficiently learns predictive ROMs from high-fidelity simulations (or other data) of solar wind models, while simultaneously producing an interpretable model and physical model form. We subsequently review the related literature, both focusing on the mathematical aspects of surrogate modeling for advection-dominated systems and the application domain of efficient heliospheric modeling.

Solar wind predictions are produced by a chain of coupled models in different parts of the Sun-Earth domain, i.e., the solar surface, corona, and heliosphere. In the heliospheric domain, there are mainly two classes of surrogate solar wind models: reduced-physics (white-box) and data-driven (black-box) models. The first approach is based on physical simplifications of the MHD equations. The simplest reduced-physics model is the ballistic approximation which assumes that each solar wind parcel maintains a constant radial speed as it propagates in the heliosphere. This approximation is mainly used to map solar wind streams for short radial distances [61]. An improved kinematic model that bridges the gap between the ballistic mapping and global three-dimensional MHD modeling is the Heliospheric Upwind Extrapolation model, where each parcel speed is dependent on its adjacent parcel speed [56, 52, 42, 51]. A second approach is to build surrogate solar wind models via data-driven and statistical techniques [12]. Such methods mainly aim to forecast the solar wind at Earth's vicinity without computing the solar wind dynamics on the full heliospheric domain. Examples of data-driven models include an artificial neural network model [67], a gradient-boosting regression-based model [9], and a probability distribution function model based on past rotation solar wind observations [11]. As we will see later, our proposed reduced-order modeling methodology is data-driven, yet accounts for the physical properties of the solar wind rotation. It therefore serves as a hybrid gray-box approach that leverages available physical information while remaining computationally efficient.

From a methodological perspective, several ROM strategies have targeted advection-dominated scenarios. For background, traditional ROMs are derived via (Petrov-) Galerkin projection, where the full-order model (FOM) is projected onto a low-dimensional subspace, see [23, 7, 22]. This class of ROMs aims to identify a small set of basis functions that minimize a certain error metric. However, a well-reported issue with linear-subspace ROMs is that they fail to model advection-dominated problems due to poorly decaying Kolmogorov N-width [19, 41, 65] which results in a slow decay of the singular values, see, e.g., [59, 24, 39, 50, 34]. An accurate ROM would require a large number of basis functions, rendering it inefficient from a computational perspective. Additionally, a large number of global basis functions can lead to numerical instabilities. The efforts to address the challenge posed by advection-dominated systems can be roughly categorized into Lagrangian-based approaches and methods that leverage a transport-invariant coordinate frame. The first line of research leverages Lagrangian coordinate grids to build a ROM that propagates both the wave physics *and* the coordinate grid in time [39, 34, 33]. These methods work extremely well, yet require knowledge of the underlying equations to solve for the Lagrangian grid, limiting their range of applicability. The second line of research, which our work builds upon, is based on transforming the dynamics to a moving coordinate frame via a time-dependent shift that is added to the spatial coordinates. In the moving frame, the system dynamics are absent of advection. The shift function can be numerically learned in various ways. For instance, the shifted proper orthogonal decomposition method [50] proposes to detect the shift either through tracking peaks of the solution, or through an expansive SVD algorithm, where different candidate shifts are applied to the data matrix and then the SVD is computed. The shifts leading to the best singular value decay are then selected and a corresponding ROM is built via Galerkin projection. This strategy is computationally quite expensive due to the need for many SVDs of a large (yet rectangular) data matrix. The authors in [38] propose an implicit feature-tracking algorithm that is based on a minimal-residual ROM. This algorithm works well on complex geometries, however, finding the domain mapping can be quite expensive, too. Very recent work by [43] builds

on the trend of machine learning to derive two separate neural networks, one for detecting nonlinear shifts in the transport velocity, and a second for interpolating a shifted solution back to the reference frame. The method is fully data-driven, yet does not propose a predictive ROM that integrates the shift detection with a projection framework. Most closely related to our work is [37], which proposes an unsupervised learning method to aid the identification and low-dimensional modeling of systems with translational symmetries. The data-based method uses sparse regression and ridge detection to identify models with non-constant wave speeds inherent to the data. The method performs well on several examples, including multiple waves travelling in opposite directions. While the examples all demonstrate interesting wave phenomena, the governing equations were always known, allowing to derive solid intuition about the wave-speed library. Other approaches that have been developed that do not fit precisely into these categories include the work by [44] that updates the ROM basis online to avoid slowly decaying Kolmogorov N-width, and the work by [24] that applies two separate mechanisms to deal with advection-diffusion systems, namely a representation of translational features via advection modes, and then the subsequent residual (features that are not purely advective) via global modes. Building on the philosophy of inducing time-dependent shifts into the ROM framework, we extend the non-intrusive projection-based operator inference ROM framework [46] towards advection-dominated systems by transforming the dynamics to a moving coordinate frame. Standard operator inference has successfully been applied to diverse applications such as combustion [62, 36], chemical reactors [10], ocean flows [68], Hamiltonian systems [60], and general reaction systems in the presence of incomplete data [64]. Since operator inference learns the operators that would be obtained through intrusive Galerkin projection (which can be done exactly with additional data pre-processing, see [45]) it inherits problems that intrusive Galerkin ROMs face in the presence of strong advection.

We propose a new strategy for efficient data-driven heliospheric solar wind modeling. The method, *shifted operator inference (sOpInf)*, builds on standard operator inference and extends it towards the challenges faced in solar wind predictions. Our proposed method first determines a moving coordinate frame where the dynamics are absent of translation and rotation and subsequently transforms the system into the new time-dependent coordinates. Two methods for predicting the shift are proposed. It then performs model learning in the shifted coordinate systems and subsequently makes predictions with the sOpInf-ROM via interpretable ODE simulation. Our hypothesis aligns with the previously cited references, in that simple translational patterns in the data and model can (and should) be exploited in the ROM approach. Our proposed approach (1) speeds up the MHD simulation by several orders of magnitude, (2) preserves the solar wind spiral pattern created by the Sun's rotation, and (3) uncovers macroscopic coherent structures present in the evolution of solar wind streams by analyzing the velocity field modal decomposition. We present computationally-efficient data-driven ROMs for two heliospheric solar wind models: the MAS (Magnetohydrodynamics Around a Sphere) model and the HUX (Heliospheric Upwinding eXtrapolation) model.

This paper is organized as follows. Section 2 describes the MAS and HUX heliospheric solar wind models and in Section 3 we present the proposed method, shifted operator inference. In Section 4 we demonstrate the performance of sOpInf on the MAS and HUX solar wind speed simulated data. Section 5 then offers conclusions and an outlook to future work.

2. Solar Wind Models

This section introduces the solar wind models considered in this study. Section 2.1 presents the MAS model. Section 2.2 discusses an approximation to that model with similar physical attributes, the HUX model.

2.1. Spherical Magnetohydrodynamics: The MAS Model

The MAS (Magnetohydrodynamics Around a Sphere) model is the primary MHD model in the CORHEL (CORona-HELiosphere) software and is publicly available at NASA's community-coordinated modeling center [3]. The MAS model solves the time-dependent resistive MHD equations and has been

used to study coronal mass ejections [31], coronal dynamics [55], solar wind structure [54], and connect *in-situ* spacecraft observations [57]. Herein, we focus our effort on analyzing the MAS solar wind radial velocity results and therefore exclude the discussion of other plasma components such as the magnetic field, plasma temperature, density, pressure, etc. This is because many space weather operational forecast models for satellite control and Earth-based infrastructure are particularly interested in near-Earth solar wind speed. Predicting the solar wind speed is pivotal for assessing the risk of geomagnetic storms because (1) coronal mass ejections, which are the most fundamental source of space weather events, are modeled as perturbations to the ambient solar wind; (2) interaction regions between fast and slow solar wind, known as co-rotating interaction regions, mainly present during solar minimum, are a driver of moderate geomagnetic activity [54]; and (3) high-speed solar wind streams cause an additional acceleration of energetic electrons in the radiation belts [16].

2.1.1. Governing Equation

The MAS model solves a system of three-dimensional time-dependent resistive MHD equations in spherical coordinates (r, θ, ϕ), where r is the radial distance from the Sun, θ is Carrington latitude in heliographic (rotating) coordinate system (HG), ϕ is the Carrington longitude in the HG coordinate system. The governing equations are

$$\nabla \times \mathbf{B} = \frac{4\pi}{c}\mathbf{J}, \tag{1}$$

$$\nabla \times \mathbf{E} = -\frac{1}{c}\frac{\partial \mathbf{B}}{\partial t}, \tag{2}$$

$$\mathbf{E} + \frac{1}{c}\boldsymbol{v} \times \mathbf{B} = \eta \mathbf{J}, \tag{3}$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0, \tag{4}$$

$$\rho\left(\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v}\right) = \frac{1}{c}\mathbf{J} \times \mathbf{B} - \nabla p - \nabla p_w + \rho \mathbf{g} + \nabla \cdot (\nu \rho \nabla \boldsymbol{v}), \tag{5}$$

$$\frac{1}{\gamma - 1}\left(\frac{\partial T}{\partial t} + \boldsymbol{v} \cdot \nabla T\right) = -T \nabla \cdot \boldsymbol{v} + S. \tag{6}$$

and the initial and boundary condition are described in [54, 30, 53]. Here, $\mathbf{B}(r, \theta, \phi, t)$ is the magnetic field, $\mathbf{J}(r, \theta, \phi, t)$ is the current density, $\mathbf{E}(r, \theta, \phi, t)$ is the electric field, $\boldsymbol{v}(r, \theta, \phi, t)$ is the plasma velocity, $T(r, \theta, \phi, t)$ is the plasma temperature, $\rho(r, \theta, \phi, t)$ is the plasma mass density, and $p(r, \theta, \phi, t)$ is the plasma pressure, and $p_w(r, \theta, \phi, t)$ is the Alfvén wave pressure. The constant c denotes the speed of light in a vacuum and $\mathbf{g}(r) = -\frac{GM_s}{r^2}\hat{\mathbf{e}}_r$ is the gravitational acceleration, where $\hat{\mathbf{e}}_r$ is the unit vector in the radial direction, G is the universal gravitational constant, and $M_s = 1.99 \times 10^{30}\,\mathrm{kg}$ is the solar mass. For the simulations used in this study, the constant resistivity is set to $\eta = 4.6779 \times 10^{-5}\,\mathrm{s}$ and the kinematic viscosity $\nu = 3.3503 \times 10^{16}\,\mathrm{m^2 s^{-1}}$. In the energy equation described in Eq. (6), the thermodynamic approximation sets the ratio of specific heats to $\gamma = 5/3$. Moreover, the energy source terms are denoted by $S = S(T)$; for more details about the thermodynamic energy source term see the MAS user guide [53]. The MAS boundary conditions exploit photospheric magnetic field observations (e.g. data from the Wilcox Solar Observatory, the Global Oscillation Network Group, and the Solar Dynamics Observatory spacecraft), see [1, 53]. Here, we choose to analyze the scenario where the thermodynamic MAS results are driven by a synoptic map of the photospheric magnetic field as it reaches a dynamic steady state [55]. Most MHD models, e.g. MAS, solve for the ambient solar wind via time-dependent simulations and allow the solution to relax to steady state. While customary, this results in a large run time to compute the steady state. There are other MHD models, such as [47], that solve directly for the steady solar wind, yet they have their own set of numerical challenges as it is no longer straightforward to extract the physical variables from the flux variables.

2.1.2. Numerical Solver

The MAS model equations are numerically solved on a nonuniform logically-rectangular staggered grid using finite differences. The nonuniform mesh allows for adjustment of the grid point concentration based on transition and active regions. For more details about the numerical methods and their stability, see [32, 14]. The MAS model divides its computational domain to two distinct regions: the corona and heliosphere. The corona is the region between $1R_S$ to $30R_S$ and the heliosphere is the region between $30R_S$ to 1.1AU. The value R_S denotes solar radii unit of distance which is 695, 700km, and 1/215th of an astronomical unit (AU), which is equal to the distance from the Sun to Earth. Numerical results and implementation details are discussed in Section 4.2.

2.2. Solar Wind Speed: The HUX Model

The HUX (Heliospheric Upwinding eXtrapolation) model developed by [56, 52] is a two-dimensional time-stationary model that predicts the heliospheric solar wind speed. The HUX model has been incorporated into operational and ensemble-based space weather programs [29, 5] as an MHD surrogate model to study retrospective time periods as well as real-time predictions. It has also been used to map streams directly from *in-situ* spacecraft observations (e.g. Helios A/B) to Earth [25]. The HUX model is based on simplified physical assumptions of the fluid momentum equation. In contrast to the MAS model, where the velocity field is solved via the MHD equations, the HUX model constructs a kinematic mapping where each plasma parcel speed is governed by its adjacent parcel's speed.

We introduce the HUX model as a reduced-physics surrogate solar wind model that is capable of capturing the solar wind speed as it propagates away from the Sun. The HUX model shares many similarities with the MAS model, such as advection-dominated solutions, and can therefore be used to test our proposed method, shifted operator inference. Moreover, since both sOpInf and HUX are surrogate models to the MAS model, we will compare their capability to approximate the MAS results. This section describes the HUX governing equations along with their spatial discretization and implementation.

2.2.1. Reduced-Physics Equation

The HUX model [56, 61] is derived from the fluid momentum equation in the corotating frame of reference with the Sun by considering Eq. (5) in the absence of magnetic and viscous effects describing steady flow by replacing the time derivative ($\frac{\partial}{\partial t}$) with a spatial derivative ($-\Omega_{\text{rot}}\frac{\partial}{\partial \phi}$), i.e. the governing equations are

$$-\Omega_{\text{rot}}(\theta)\frac{\partial \boldsymbol{v}(r,\theta,\phi)}{\partial \phi} + \left[\boldsymbol{v}(r,\theta,\phi)\cdot\nabla\right]\boldsymbol{v}(r,\theta,\phi) = -\frac{1}{\rho(r,\theta,\phi)}\nabla p(r,\theta,\phi) + \mathbf{g}(r), \tag{7}$$

where $\boldsymbol{v} = [v_r(r,\theta,\phi), v_\theta(r,\theta,\phi), v_\phi(r,\theta,\phi)]$ is the solar wind proton velocity, $\rho(r,\theta,\phi)$ is the plasma density, and $\mathbf{g}(r)$ is the gravitational acceleration specified in Section 2.1.1. The term $\Omega_{\text{rot}}(\theta) = \frac{2\pi}{25.38} - \frac{2.77\pi}{180}\cos(\theta - \frac{\pi}{2})^2$ is the angular frequency of the Sun's rotation, i.e., a function of latitude [52]. The analysis in [56, 52, 25] justifies neglecting the pressure gradient and gravity terms in Eq. (7) and only taking into account variations of the velocity in the radial direction. As a result, Eq. (7) reduces to the two-dimensional nonlinear scalar homogeneous time-stationary equation

$$-\Omega_{\text{rot}}(\theta = \hat{\theta})\frac{\partial v_r(r,\phi)}{\partial \phi} + v_r(r,\phi)\frac{\partial v_r(r,\phi)}{\partial r} = 0, \tag{8}$$

where the independent variables are r and ϕ and the dependent variable is the velocity in the radial direction $v_r(r,\phi)$. The angular frequency of the Sun's rotation is evaluated at a constant Carrington latitude $\hat{\theta}$; here we consider the Sun's equatorial plane ($\hat{\theta} = 0$) so that $\Omega_{\text{rot}}(0) = \frac{2\pi}{25.38}1/\text{days}$ at the solar equator. The initial-boundary value problem (IBVP) is subject to the initial condition $v_r(r_0,\phi) = v_{r_0}(\phi)$ and is defined on the periodic domain $0 \le \phi \le 2\pi$ and $r \ge 30R_S$, where beyond $30R_S$, the solar wind travels along roughly radial trajectories, justifying the assumption of only considering the

velocity in the radial direction. Additionally, to account for the residual acceleration present in the inner heliosphere, the authors in [56] suggested adding an acceleration boost to the initial velocity profile described by

$$v_{\text{acc}}(r_0, v_{r_0}(\phi)) = \alpha[v_{r_0}(\phi)](1 - e^{-r_0/r_h}), \tag{9}$$

where $v_{r_0}(\phi)$ is the initial radial velocity, $\alpha = 0.15$ is the acceleration factor, and $r_h = 50R_S$ is the radial location at which the acceleration ends. Hence, the acceleration boost, $v_{\text{acc}}(r_0, v_{r_0}(\phi))$, is added to the initial velocity profile $v_{r_0}(\phi)$ prior to solving the HUX equation.

2.2.2. Discretization via the Upwind Scheme

This section describes the semi-discretization of Eq. (8) in longitude, which then results in a set of ODEs. To begin, we rewrite Eq. (8) in the hyperbolic conservation form

$$\frac{\partial}{\partial r} v_r(r, \phi) + \frac{\partial}{\partial \phi} f[v_r(r, \phi)] = 0, \tag{10}$$

where the physical flux function is $f[v_r(r, \phi)] = -\Omega_{\text{rot}}(\hat{\theta}) \ln[v_r(r, \phi)]$. We use the first-order conservative upwind method from [25], so to approximate the partial derivative of the flux function f with respect to ϕ by

$$\frac{\partial}{\partial \phi} f[v_r(r, \phi^{(j)})] \approx \frac{-\Omega_{\text{rot}}(\hat{\theta})}{\Delta\phi} \left(\ln[v_r(r, \phi^{(j+1)})] - \ln[v_r(r, \phi^{(j)})] \right), \tag{11}$$

where n_ϕ is the number of mesh points in longitude and $j = 1, 2, \ldots, n_\phi$ denotes the longitude grid index. We discretize the longitudinal direction uniformly with $\Delta\phi$ mesh spacing and denote the discretized state vector as $\mathbf{v}(r) = [v_r(r, \phi^{(1)}), v_r(r, \phi^{(2)}), \ldots, v_r(r, \phi^{(n_\phi)})]^\top \in \mathbb{R}^{n_\phi}$. From here, we obtain the semi-discrete system of ordinary differential equations

$$\frac{\mathrm{d}}{\mathrm{d}r} \mathbf{v}(r) = \mathbf{D} \ln[\mathbf{v}(r)] \tag{12}$$

with the sparse matrix

$$\mathbf{D} = \frac{\Omega_{\text{rot}}(\hat{\theta})}{\Delta\phi} \begin{bmatrix} -1 & 1 & 0 & & & & \\ 0 & -1 & 1 & & & & \\ & \ddots & \ddots & \ddots & \ddots & & \\ & & & -1 & 1 & 0 & \\ & & & & -1 & 1 & \\ 1 & & & & 0 & -1 & \end{bmatrix} \in \mathbb{R}^{n_\phi \times n_\phi}. \tag{13}$$

The initial condition $\mathbf{v}(r_0) \in \mathbb{R}^{n_\phi}$ is set as the MAS coronal solution, $\mathbf{v}^{\text{MAS}} \in \mathbb{R}^{n_\phi}$, evaluated at $r_0 = 30R_S$ along with adding the ad hoc acceleration boost described in Eq. (9), i.e.,

$$\mathbf{v}(r_0) = \mathbf{v}^{\text{MAS}}(r_0) \left[1 + \alpha(1 - e^{-r_0/r_h}) \right]. \tag{14}$$

3. Shifted Operator Inference: A Non-intrusive Reduced-Order Model Approach for Advection-Dominated Systems

This section proposes *shifted operator inference (sOpInf)*, a non-intrusive data-driven modeling framework that expands standard operator inference [44] towards the challenge of modeling solar wind, and more generally, advection-dominated systems. The proposed method ensures that the learned ROM is able to (i) capture the dynamics of the translational systems with only a few modes, (ii) retain the translation and rotation properties of the physical system, and (iii) accurately predict

the shift velocity in the testing regime. This is done by transforming the original coordinates to a moving coordinate frame where the dynamics are absent of translation and rotation. Section 3.1 describes the new method that leverages a coordinate shift to first transform the data and subsequently learn in the transformed coordinates. Section 3.2 proposes two alternative strategies for deriving this coordinate transformation. Lastly, Section 3.3 illustrates the sOpInf methodology on an introductory example: the one-dimensional inviscid Burgers' equation.

3.1. Shifted Operator Inference for Advection-Dominated Systems

To illustrate the proposed methodology, we consider a generic $k-$dimensional time-dependent partial differential equation (PDE) for the scalar function $u(x_1, x_2, \ldots, x_k, t)$ of the form

$$F\left(u, x_1, \ldots, x_k, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_k}, t\right) = 0, \tag{15}$$

where $x_1, x_2, \ldots, x_k \in \mathbb{R}$ denote the spatial coordinates and $t \in \mathbb{R}^+$. One may think of t as time, or, as described in the previous section, we will also consider the independent variable to be the radial distance from the Sun, r. The function F defines the equations of motion of the system which include advective terms. For simplicity, we focused the illustration on a purely convective case, but our method can account for higher derivatives (e.g. diffusion terms) as well, see Section 4 where we consider a model with viscous forces.

Our goal is to derive a data-driven ROM that can accurately predict the solutions to advection-dominated systems, i.e., that uses data of a semi-discretization of (15) and produces a predictive and efficient low-dimensional model. The proposed method proceeds in four steps as outlined next.

(I) Data collection and translation. The system (15) is in k dimensions and is typically solved via a spatial discretization scheme at fixed spatial locations $\mathbf{x}_i = [x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n_i)}] \in \mathbb{R}^{n_i}$, $i = 1, 2, \ldots, k$, which we refer to as the *original coordinates*. These are assembled into the following spatial grid

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & \ldots & x_1^{(n_x)} \\ \vdots & \ddots & \vdots \\ x_k^{(1)} & \ldots & x_k^{(n_x)} \end{bmatrix} \in \mathbb{R}^{k \times n_x} \qquad \text{and} \qquad \mathbf{x} = \text{vec}(\mathbf{X}) \in \mathbb{R}^n,$$

where $n_x = \Pi_{i=1}^k n_i$ is the number of spatial grid points and $n = k \cdot n_x$. Depending on the context, using $\mathbf{X}$ in matrix form or $\mathbf{x}$ in vector form may be more preferred. We collect data (for instance, solar wind speed data) from the numerical solver in the original coordinates at instances t_i, i.e., $\mathbf{u}_i \approx u(\mathbf{x}, t_i) \in \mathbb{R}^n$ with $0 = t_0 < t_1 < \cdots < t_K = T$. To account for the translational element in the data, we next shift each snapshot to a moving coordinate frame

$$\mathbf{u}_i \approx u(\mathbf{x}, t_i) \mapsto \tilde{u}(\tilde{\mathbf{x}}(\mathbf{x}, t_i), t_i) \approx \tilde{\mathbf{u}}_i \qquad \text{with} \qquad \tilde{\mathbf{x}}(\mathbf{x}, t) = \mathbf{x} + \mathbf{c}(t) \tag{16}$$

where $\tilde{\mathbf{x}}(\mathbf{x}, t_i)$ denotes the moving coordinate frame and $\mathbf{c}(t) \in \mathbb{R}^n$ represents the traveling wave speed. We evaluate $\tilde{\mathbf{u}}_i$ via piecewise linear interpolation, i.e.,

$$\tilde{\mathbf{u}}_i = \mathcal{P}_i^k \left[\mathbf{u}_i, \mathbf{x}, \tilde{\mathbf{x}}(\mathbf{x}, t_i)\right]$$

where $\mathcal{P}_i^k$ denotes the $k > 1$ dimensional piecewise linear interpolant of $\mathbf{u}_i$ in the original grid $\mathbf{x}$ evaluated on the moving coordinate frame grid $\tilde{\mathbf{x}}(\mathbf{x}, t_i)$. The piecewise linear interpolation is implemented in the Python package `scipy` under the function `scipy.interpolate.LinearNDInterpolator()`. Our method is not restricted to the type of interpolation, so higher order interpolation methods can be used as well (e.g. piecewise cubic interpolation). In the moving coordinate frame, the system no longer exhibits translational properties. Section 3.2 proposes two techniques to determine $\mathbf{c}(t)$. We then store the transformed data in the matrix

$$\tilde{\mathbf{U}} = [\tilde{\mathbf{u}}_1 \quad \ldots \quad \tilde{\mathbf{u}}_K] \in \mathbb{R}^{n \times K},$$

where in the applications that we consider, $n \gg K$, so the matrix $\tilde{\mathbf{U}}$ is tall and skinny.

(II) Data reduction via projection. Given high-dimensional data, we first identify the low-dimensional subspace in which to learn a ROM. In this work, we use the subspace spanned by the proper orthogonal decomposition (POD) modes [23], which is obtained by computing the economy-sized singular value decomposition of the snapshot matrix, i.e.,

$$\tilde{\mathbf{U}} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{W}^\top, \tag{17}$$

where $\mathbf{V} \in \mathbb{R}^{n\times K}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{K\times K}$ and $\mathbf{W} \in \mathbb{R}^{K\times K}$. The $\ell \ll n$ dimensional POD basis, $\mathbf{V}_\ell = [\mathbf{v}_1, \ldots, \mathbf{v}_\ell]$, is given by the first ℓ columns of $\mathbf{V}$. The basis dimensions can be chosen based on the cumulative energy criteria, e.g.,

$$\ell = \arg\min_{\hat{\ell}} \frac{\sum_{i=1}^{\hat{\ell}} \sigma_i}{\sum_{i=1}^{K} \sigma_i} > \epsilon_{\text{tol}}, \tag{18}$$

where $\sigma_i = \boldsymbol{\Sigma}_{ii}$ are the singular values, and ϵ_{tol} is commonly chosen to be $\epsilon_{\text{tol}} = 0.95$ or $\epsilon_{\text{tol}} = 0.99$ which encapsulate 95% and 99% of the energy in the data, respectively. Next, we project the state snapshot data onto the POD subspace spanned by the columns of $\mathbf{V}_\ell$ and obtain the reduced snapshot matrices

$$\widehat{\mathbf{U}} = \mathbf{V}_\ell^\top \tilde{\mathbf{U}} = [\hat{\mathbf{u}}_1 \quad \ldots \quad \hat{\mathbf{u}}_K] \in \mathbb{R}^{\ell\times K}, \qquad \text{and} \qquad \dot{\widehat{\mathbf{U}}} = [\dot{\hat{\mathbf{u}}}_1 \quad \ldots \quad \dot{\hat{\mathbf{u}}}_K] \in \mathbb{R}^{\ell\times K}, \tag{19}$$

where the columns of $\dot{\widehat{\mathbf{U}}}$ are computed from $\widehat{\mathbf{U}}$ using any time derivative approximation (see, e.g., [35, 27, 15]), or can be obtained—if available—by evaluating the right-hand-side of the governing equation (the residual) and projecting the resulting data.

(III) Model learning and prediction via operator inference. In this section we start with the assumption that the finite-dimensional data-generating model (a spatial discretization of Eq. (15) in the moving coordinate frame) is of the form of a polynomial nonlinear system of ODEs, written as

$$\frac{\mathrm{d}\tilde{\mathbf{u}}}{\mathrm{d}t} = \mathbf{A}\tilde{\mathbf{u}} + \mathbf{H}(\tilde{\mathbf{u}} \otimes \tilde{\mathbf{u}}) + \mathbf{C}(\tilde{\mathbf{u}} \otimes \tilde{\mathbf{u}} \otimes \tilde{\mathbf{u}}) + \mathbf{B} + \text{HOT}, \qquad \tilde{\mathbf{u}} \in \mathbb{R}^n \tag{20}$$

with matrices $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{H} \in \mathbb{R}^{n\times n^2}$ and $\mathbf{C} \in \mathbb{R}^{n\times n^3}$. The column-wise Kronecker product is denoted by $\otimes$. Boundary conditions can either be represented via the constant $\mathbf{B} \in \mathbb{R}^n$ or time-dependent BCs as $\mathbf{B}\eta(t)$. The abbreviation "HOT" in Eq. (20) denotes higher-order terms, and represents terms that are quartic and higher order. For example, in the case of Burgers' equation in Section 3.3 the term $u\frac{\partial u}{\partial x}$ is quadratic in the PDE state $u(x,t)$ and would therefore yield a discretized component $\mathbf{H}(\tilde{\mathbf{u}} \otimes \tilde{\mathbf{u}})$.

Approximating the high-dimensional state $\tilde{\mathbf{u}}$ in a low-dimensional basis $\mathbf{V}_\ell \in \mathbb{R}^{n\times\ell}$, with $\ell \ll n$, we write $\tilde{\mathbf{u}} \approx \mathbf{V}_\ell\hat{\mathbf{u}}$. Using a Galerkin projection, this yields the ROM of Eq. (20) as

$$\frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t} = \widehat{\mathbf{A}}\hat{\mathbf{u}} + \widehat{\mathbf{H}}(\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}}) + \widehat{\mathbf{C}}(\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}} \otimes' \hat{\mathbf{u}}) + \widehat{\mathbf{B}} + \text{HOT}, \qquad \hat{\mathbf{u}} \in \mathbb{R}^\ell \tag{21}$$

where $\otimes'$ is the compact Kronecker product, which removes redundant terms in the standard Kronecker product $\otimes$. For example, for $\hat{\mathbf{u}} = [\hat{u}_1, \hat{u}_2]^\top$ the standard Kronecker product yields $\hat{\mathbf{u}} \otimes \hat{\mathbf{u}} = [\hat{u}_1^2, \hat{u}_1\hat{u}_2, \hat{u}_2\hat{u}_1, \hat{u}_2^2]^\top$ and the compact Kronecker product yields $\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}} = [\hat{u}_1^2, \hat{u}_1\hat{u}_2, \hat{u}_2^2]^\top$, which uses only unique terms. For here on, we only use the compact Kronecker product for learned sOpInf ROMs. Consequently, the ROM operators and their dimensions are

$\widehat{\mathbf{A}} = \mathbf{V}_\ell^\top\mathbf{A}\mathbf{V}_\ell \in \mathbb{R}^{\ell\times\ell}$, $\widehat{\mathbf{H}} = \mathbf{V}_\ell^\top\mathbf{H}(\mathbf{V}_\ell \otimes' \mathbf{V}_\ell) \in \mathbb{R}^{\ell\times\frac{1}{2}\ell(\ell+1)}$, $\widehat{\mathbf{C}} = \mathbf{V}_\ell^\top\mathbf{C}(\mathbf{V}_\ell \otimes' \mathbf{V}_\ell \otimes' \mathbf{V}_\ell) \in \mathbb{R}^{\ell\times\frac{1}{6}\ell(\ell+1)(\ell+2)}$, and $\widehat{\mathbf{B}} = \mathbf{V}_\ell^\top\mathbf{B} \in \mathbb{R}^\ell$ is the reduced constant vector $\mathbf{B}$. We note again that projection preserves polynomial structure, that is, Eq. (21) has the same polynomial form as Eq. (20), but in the reduced subspace defined by $\mathbf{V}_\ell$.

To simplify notation, we continue from now on with a quadratic system, but note that all results carry over directly to cubic, quartic and all higher-order polynomial terms. Nevertheless, we note

that the number of elements in the ROM operators scales with ℓ^4 for the cubic operator, ℓ^5 for the quartic operator, etc., yet higher-order terms often exhibit significant block-sparsity that can be exploited in numerical implementations which limits the growth of computational cost to solve the ROM. For terms in the governing equations that are not in polynomial form, the introduction of variable transformations and auxiliary variables via the process of lifting [20, 28, 62, 48] can convert these terms to polynomial form.

The goal at this stage is to learn a ROM that evolves the shifted and projected snapshots in time. Operator inference solves a least-squares problem to find the reduced operators that yield the ROM that best matches the projected snapshot data in a minimum residual sense. For a quadratic ROM (with $\widehat{\mathbf{C}}$ and HOT set to zero in Eq. (21)) operator inference solves the least-squares problem

$$\min_{\widehat{\mathbf{A}}\in\mathbb{R}^{\ell\times\ell},\widehat{\mathbf{H}}\in\mathbb{R}^{\ell\times\frac{1}{2}\ell(\ell+1)},\widehat{\mathbf{B}}\in\mathbb{R}^{\ell}} \left\| \left[\widehat{\mathbf{A}}\widehat{\mathbf{U}} + \widehat{\mathbf{H}}(\widehat{\mathbf{U}} \otimes' \widehat{\mathbf{U}}) + \widehat{\mathbf{B}}\,\mathbf{1}_K - \dot{\widehat{\mathbf{U}}}\right]^\top \right\|_{\mathrm{F}}^2,$$

where $\mathbf{1}_K \in \mathbb{R}^K$ is the length K row vector with all entries set to one. Note that this least-squares problem is linear in the coefficients of the unknown ROM operators $\widehat{\mathbf{A}}$, $\widehat{\mathbf{H}}$, and $\widehat{\mathbf{B}}$. The appeal of the operator inference approach comes from the ability to compute the ROM operators $\widehat{\mathbf{A}}$, $\widehat{\mathbf{H}}$, and $\widehat{\mathbf{B}}$ directly from data without needing explicit access to the original high-dimensional operators $\mathbf{A}$, $\mathbf{H}$, and $\mathbf{B}$. The unknown operators and known low-dimensional data are combined in the matrices

$$\mathbf{O} = [\widehat{\mathbf{A}} \quad \widehat{\mathbf{H}} \quad \widehat{\mathbf{B}}] \in \mathbb{R}^{\ell\times(\ell+\frac{1}{2}\ell(\ell+1)+1)} \qquad \text{and} \qquad \mathbf{D} = \left[\widehat{\mathbf{U}}^\top \quad (\widehat{\mathbf{U}} \otimes' \widehat{\mathbf{U}})^\top \quad \mathbf{1}_K\right] \in \mathbb{R}^{K\times(\ell+\frac{1}{2}\ell(\ell+1)+1)},$$

respectively. The unknown operators are then obtained as a solution to the minimization problem

$$\min_{\mathbf{O}\in\mathbb{R}^{\ell\times(\ell+\frac{1}{2}\ell(\ell+1)+1)}} \left\| \mathbf{D}\mathbf{O}^\top - \dot{\widehat{\mathbf{U}}}^\top \right\|_{\mathrm{F}}^2. \tag{22}$$

For $K > \ell + \frac{1}{2}\ell(\ell+1) + 1$ this overdetermined linear least-squares problem has a unique solution [18, Sec. 5.3]. It follows from linear algebra (and is noted in [46]) that Eq. (22) can be written as ℓ independent least-squares problems, each of the form

$$\min_{\mathbf{o}_i\in\mathbb{R}^{\ell+\frac{1}{2}\ell(\ell+1)+1}} \|\mathbf{D}\mathbf{o}_i - \mathbf{r}_i\|_2^2,$$

for $i = 1, \ldots, \ell$, where $\mathbf{o}_i$ is a column of $\mathbf{O}^\top$ (row of $\mathbf{O}$) and $\mathbf{r}_i$ is a column of $\dot{\widehat{\mathbf{U}}}^\top$. This makes the operator inference approach efficient and scalable.

To avoid overfitting and prevent potential instability of the learned ROMs, regularization becomes necessary, see [36] for a detailed regularization study of operator inference. In this work, we use an Tikhonov regularization penalty so that the least-squares problem becomes

$$\min_{\mathbf{O}\in\mathbb{R}^{\ell\times(\ell+\frac{1}{2}\ell(\ell+1)+1)}} \left\| \mathbf{D}\mathbf{O}^\top - \dot{\widehat{\mathbf{U}}}^\top \right\|_{\mathrm{F}}^2 + \left\|\boldsymbol{\Gamma}\mathbf{O}^\top\right\|_{\mathrm{F}}^2 \tag{23}$$

where $\boldsymbol{\Gamma} = \operatorname{diag}(\lambda_1\mathbf{I}_{(\ell)}, \lambda_2\mathbf{I}_{(\frac{1}{2}\ell(\ell+1))}, \lambda_1) \in \mathbb{R}^{(\ell+\frac{1}{2}\ell(\ell+1)+1)\times(\ell+\frac{1}{2}\ell(\ell+1)+1)}$ is the diagonal matrix used for regularization. The parameter λ_1 is the regularization parameter of the operators $\widehat{\mathbf{B}} \in \mathbb{R}^{\ell}$ and $\widehat{\mathbf{A}} \in \mathbb{R}^{\ell\times\ell}$ and λ_2 regularizes the operator $\widehat{\mathbf{H}} \in \mathbb{R}^{\ell\times\frac{1}{2}\ell(\ell+1)}$. The regularization parameters λ_1 and λ_2 are problem specific and should be chosen accordingly. We provide details in Section 4 and refer to [62, Sec. IV.B] for more implementation details of operator inference.

Having learned the ROM in Eq. (21) from the shifted data, we can make efficient predictions in that low-dimensional subspace that go beyond the training data into the fully predictive regime. We thus simulate Eq. (21) to obtain a solution $\hat{\mathbf{u}}(t)$, which we then lift to n dimensions to get the approximate ROM solution $\tilde{\mathbf{u}}^{\mathrm{ROM}}(t) = \mathbf{V}_\ell\hat{\mathbf{u}}(t) \approx \tilde{\mathbf{u}}(t)$. We use the Operator Inference Python package version 1.2.1 [4] to implement the model learning and prediction step of sOpInf.

(IV) Re-shifting predicted ROM data. The predicted ROM solutions $\tilde{\mathbf{u}}^{\mathrm{ROM}}(t)$ will be in the moving coordinate frame and require reverse translation to the original coordinates. We shift the ROM-predicted solutions back to the original coordinate system via interpolation

$$\mathbf{u}_i^{\mathrm{ROM}}(\mathbf{x}, t_i) = \mathcal{P}_i^k \left[\tilde{\mathbf{u}}_i^{\mathrm{ROM}}, \tilde{\mathbf{x}}(\mathbf{x}, t_i), \mathbf{x}\right]$$

where $\mathcal{P}_i^k$ denotes the $k > 1$ dimensional (here: piecewise linear) interpolant of $\tilde{\mathbf{u}}_i^{\mathrm{ROM}}$ in the moving coordinate frame grid $\tilde{\mathbf{x}}(\mathbf{x}, t_i)$ evaluated on the original grid $\mathbf{x}$; see part (I) in Section 3.1 for more details about the interpolation implementation. After shifting back to the original coordinates, the predicted and reconstructed sOpInf snapshots are columns of the matrix $\mathbf{U}^{\mathrm{ROM}} \in \mathbb{R}^{n\times(K+m)}$, such that $t_{K+m} > t_K$, which is the final output of the algorithm.

The previous steps (I)–(IV) are summarized in Algorithm 1, which is written for a quadratic system; yet extensions to cubic, quartic, and other polynomial systems are straightforward.

Algorithm 1 Shifted operator inference (sOpInf)

Input: $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K] \in \mathbb{R}^{n\times K}$ such that each column, $\mathbf{u}_i \in \mathbb{R}^n$, is a snapshot observed at t_i, SVD cumulative energy threshold $\epsilon_{\mathrm{tol}} > 0$, and regularization coefficients $\{\lambda_1, \lambda_2\}$.
Output: $\mathbf{U}^{\mathrm{ROM}} \in \mathbb{R}^{n\times(K+m)}$ sOpInf reconstructed and predicted snapshots, where $t_{K+m} > t_K$.
Begin:

1: Learn shift function $\mathbf{c}(t)$. ▷ Section 3.2
2: Shift snapshots to moving coordinate system $\mathbf{U}(\mathbf{x}, t_i) \mapsto \tilde{\mathbf{U}}(\tilde{\mathbf{x}}(\mathbf{x}, t_i), t_i)$ with $\tilde{\mathbf{x}}(\mathbf{x}, t) = \mathbf{x} + \mathbf{c}(t)$. ▷ Section 3.1$(I)$
3: Determine low-dimensional subspace matrix $\mathbf{V}_\ell$ by using threshold ϵ_{tol}. ▷ Section 3.1(II)
4: Project to low-dimensional subspace $\widehat{\mathbf{U}} = \mathbf{V}_\ell^\top \tilde{\mathbf{U}}$ and compute $\dot{\widehat{\mathbf{U}}}$. ▷ Section 3.1$(II)$
5: Solve the linear least-squares problem in Eq. (23) with regularization coefficients $\{\lambda_1, \lambda_2\}$ to obtain the sought ROM operators; here $\widehat{\mathbf{B}}, \widehat{\mathbf{A}}, \widehat{\mathbf{H}}$. ▷ Section 3.1$(III)$
6: Simulate the ROM in Eq. (21) to get $\hat{\mathbf{u}}(t)$ and lift to $\tilde{\mathbf{u}}^{\mathrm{ROM}}(t) = \mathbf{V}_\ell \hat{\mathbf{u}}(t)$. ▷ Section 3.1$(III)$
7: Shift ROM results to original coordinates $\tilde{\mathbf{u}}^{\mathrm{ROM}}(\tilde{\mathbf{x}}(\mathbf{x}, t_i), t_i) \mapsto \mathbf{u}^{\mathrm{ROM}}(\mathbf{x}, t_i)$. ▷ Section 3.1$(IV)$

3.2. Determination of Spatial Shift Velocity

There are various ways in which the traveling wave speed, $\mathbf{c}(t)$, can be discovered. For instance, the authors in [37] use sparse regression and spectral clustering to uncover the function $\mathbf{c}(t)$. We present two methods: the method of characteristics discussed in Section 3.2.1, an analytic approach that requires knowledge of the underlying equations (but not the discretization or computer code), and the cross-correlation extrapolation method described in Section 3.2.2, a purely data-driven approach. In both cases, the shift function $\mathbf{c}(t)$ is learned from the batch of training data and extrapolated in the testing regime.

3.2.1. Method of Characteristics

The method of characteristics can be applied to quasi-linear partial differential equations, in which along the characteristic curves the PDE can be transformed to a set of coupled ODEs. We exploit the method of characteristics to find the scalar shift function $c(t)$ for first-order PDEs describing the scalar quantity $u(x_1, x_2, \ldots, x_k, t) : [a_1, b_1] \times \ldots \times [a_k, b_k] \times [t_0, t_f] \mapsto \mathbb{R}^+$, whose dynamics are described by the following quasi-linear PDE:

$$\frac{\partial u(x_1, \ldots, x_k, t)}{\partial t} + \sum_{i=1}^{k} f_i\left[u(x_1, \ldots, x_k, t), t\right] \frac{\partial u(x_1, \ldots, x_k, t)}{\partial x_i} = g\left[u(x_1, \ldots, x_k, t), t\right], \tag{24}$$

subject to the initial condition $u(x_1, \ldots, x_k, t = 0) = u_0(x_1, \ldots, x_k)$ with periodic boundary conditions at each spatial coordinate boundaries $x_i = a_i$ and $x_i = b_i$ for $i = 1, 2, \ldots, k$. The transport speed

$f_i\left[u(x_1,\ldots,x_k,t),t\right]\in\mathbb{R}$ must be strictly positive or negative $\forall t\in[t_0,t_f]$ and $\forall x_i\in[a_i,b_i]$ to enforce uni-directional characteristics. The function $g\left[u(x_1,\ldots,x_k,t),t\right]\in\mathbb{R}$ is the source term. Then, by the method of characteristics, Eq. (24) can be written as a system of $k+1$ ODEs, i.e.

$$\frac{\mathrm{d}u(x_1(t),\ldots,x_k(t),t)}{\mathrm{d}t}=g\left[u(x_1(t),\ldots,x_k(t),t),t\right], \tag{25a}$$

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t}=f_i\left[u(x_1(t),\ldots,x_k(t),t),t\right],\quad i=1,\ldots,k. \tag{25b}$$

We solve Eq. (25a) first, which results in

$$u(x_1(t),\ldots,x_k(t),t)=G\left[u_0(x_1(0),\ldots,x_k(0)),t\right], \tag{26}$$

where $G\left[u_0(x_1(0),\ldots,x_k(0)),t\right]\in\mathbb{R}$ describes the scalar quantity u along the characteristic curves. Given this ansatz, we proceed to obtain the characteristic curves by solving Eq. (25b) via separation of variables, such that

$$x_i(t)=x_i(0)+\int_{t_0}^{t}f_i\left[G\left[u_0(x_1(0),\ldots,x_k(0)),t\right],t\right]\mathrm{d}t. \tag{27}$$

For most equations that arise from conservation laws, the ODEs in Eq. (25a)–(25b) can be solved analytically via Eq. (26)–(27). In the case when Eq. (25a) can not be solved analytically, the coupled system of ODEs in Eq. (25a)–(25b) can be solved numerically on a discrete grid; or in the case when $f_i[G,t]$ is a nonelementary antiderivative, we can approximate the function f_i using Taylor series, and integrate term-by-term.

By following the characteristic curves described in Eq. (27) we are able to discover a moving coordinate frame absent of advection. In fluid dynamics, the characteristic paths are referred to as the Lagrangian specification of the flow field, whereby the fluid motion is observed following an individual fluid parcel. We construct a moving coordinate frame, $\mathbf{x}+\mathbf{c}(t)$, that resembles the main direction of the Lagrangian frame of reference. The shift function $c_i(t)$ corresponding to the spatial coordinate x_i can be obtained by computing the *mean characteristic* emerging from a certain spatial domain. Let the spatial domain of x_i, for $i=1,2,\ldots,k$ be discretized on a grid with n_i points such that $\mathbf{x}_i=[x_i^{(1)},x_i^{(2)},\ldots,x_i^{(n_i)}]\in\mathbb{R}^{n_i}$. The mean characteristic (and hence the shift function $c_i(t)\in\mathbb{R}$ for $i=1,2,\ldots k$) emerging in the interval $[x_i^{(p)},x_i^{(q)}]$ with $1\le p<q\le n_i$ is given as

$$c_i(t)=\frac{1}{|x_i^{(p)}-x_i^{(q)}|}\int_{x_i^{(p)}}^{x_i^{(q)}}\int_{t_0}^{t}f_i\left[G\left[u_0(x_1(0),\ldots,x_k(0)),t\right],t\right]\mathrm{d}t\;\mathrm{d}x_i \tag{28}$$

$$\approx\frac{1}{(q-p)}\sum_{j=p}^{q}\int_{t_0}^{t}f_i\left[G\left[u_0(x_1(0),\ldots,x_i^{(j)}(0),\ldots,x_k(0)),t\right],t\right]\mathrm{d}t. \tag{29}$$

The spatial interval $[x_i^{(p)},x_i^{(q)}]$ can be set to the entire spatial domain, yet it is usually set to be a specific region of interest.

For problems with shock formation, the shift function $\mathbf{c}(t)$ is computed via the mean characteristic curve only before shock formation, i.e. before the characteristic lines first intersect, and approximated via the shock curve after shock formation, i.e.

$$c_i(t)=\begin{Bmatrix}\frac{1}{|x_i^{(p)}-x_i^{(q)}|}\int_{x_i^{(p)}}^{x_i^{(q)}}\int_{t_0}^{t}f_i\left[G\left[u_0(x_1(0),\ldots,x_k(0)),t\right],t\right]\mathrm{d}t\;\mathrm{d}x_i & \text{if } t_0<t<t_s\\ s_i(t)-s_i(t_s)+a & \text{if } t>t_s\end{Bmatrix} \tag{30}$$

where $a=\frac{1}{|x_i^{(p)}-x_i^{(q)}|}\int_{x_i^{(p)}}^{x_i^{(q)}}\int_{t_0}^{t_s}f_i\left[G\left[u_0(x_1(0),\ldots,x_k(0)),t\right],t\right]\mathrm{d}t\;\mathrm{d}x_i$. The time of shock formation is denoted by t_s, and $s_i(t)$ is the shock trajectory in x_i coordinate, see Section 3.3 for a one-dimensional

inviscid Burgers' equation example. For one-dimensional problems where $g = 0$, the shock trajectory can be computed via the entropy (Rankine-Hugoniot) condition or the Whitham's geometric equal area rule in which multi-valued regions of the solution are replaced with a discontinuity that satisfies conservation [66]. The shock location is approximated by locating a vertical line that splits the multi-valued curve into two regions with equal area.

In problems where there are multiple shock curves, such as shown in Section 4.3, we suggest to approximate $\mathbf{c}(t)$ by tracking only one shock curve. The selection of the shock is problem dependent, e.g. in Section 4.3 we choose to follow the first shock emerging. Although, if the choice of shock curve or spatial interval $[x^{(p)}, x^{(q)}]$ is ambiguous, computing the shock curve is unfeasible or computationally expensive, or the initial condition is noisy, we recommend to use the cross-correlation extrapolation technique (Section 3.2.2) to find $\mathbf{c}(t)$. Additionally, it can be non-trivial to find such characteristics in the case of more complex hyperbolic PDEs that are not in the form of Eq. (24), e.g. coupled systems such as Eqs. (1)–(6) with multiple dependent variables resulting in more than one characteristic curve emanating from a single spatial point, in which we recommend employing the data-driven cross-correlation extrapolation technique, which we present next.

3.2.2. Cross-Correlation Extrapolation Method

Cross-correlation is a mathematical operation that is commonly used in signal processing and pattern recognition to measure similarity of two signals. For two finite discrete signals $\mathbf{f}, \mathbf{g} \in \mathbb{C}^n$, the univariate discrete *circular cross-correlation* is defined as

$$(\mathbf{f} \star \mathbf{g})[\tau] \coloneqq \sum_{j=1}^{n} \overline{\mathbf{f}[j]}\mathbf{g}[(j+\tau)_{\mathrm{mod}_n}], \tag{31}$$

where $\overline{\mathbf{f}}$ denotes the complex conjugate of $\mathbf{f}$, the bracket $[j]$ denotes the jth element of the signal, and $\tau \in \mathbb{Z}$ is the discrete displacement. The discrete circular cross-correlation can be extended to the multi-variate case, for snapshots with $k \in \mathbb{Z}$ variables and tensor-valued $\mathbf{f}, \mathbf{g} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_k}$:

$$(\mathbf{f} \overset{k}{\star\cdots\star} \mathbf{g})[\boldsymbol{\tau}] \coloneqq \sum_{j_1=1}^{n_1} \cdots \sum_{j_k=1}^{n_k} \overline{\mathbf{f}[j_1, \ldots, j_k]}\mathbf{g}[(j_1+\tau_1)_{\mathrm{mod}_{n_1}}, \ldots, (j_k+\tau_k)_{\mathrm{mod}_{n_k}}], \tag{32}$$

where $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_k] \in \mathbb{Z}^k$ is the multi-variate discrete displacement. When the signals correlate, the value of $\mathbf{f} \star \mathbf{g}$ is maximized.

We propose to find the optimal discrete displacement, $\boldsymbol{\tau}^* \in \mathbb{Z}^k$, by maximizing the cross-correlation between the two discrete signals (or snapshots), which amounts to solving

$$\boldsymbol{\tau}^* \coloneqq \underset{\boldsymbol{\tau} \in \mathbb{Z}^k}{\arg\max} (\mathbf{f} \overset{k}{\star\cdots\star} \mathbf{g})[\boldsymbol{\tau}]. \tag{33}$$

Once the shift is computed for all training snapshots by applying the circular discrete cross-correlation between each snapshot $\mathbf{u}_i \in \mathbb{R}^n$ and the initial condition $\mathbf{u}_0 \in \mathbb{R}^n$, we obtain the the multivariate discrete displacement for each time-step, i.e.,

$$\boldsymbol{\tau}^*(t_i) \coloneqq \underset{\boldsymbol{\tau} \in \mathbb{Z}^k}{\arg\max} \{\mathbf{u}_0 \overset{k}{\star\cdots\star} \mathbf{u}_i\}[\boldsymbol{\tau}]. \tag{34}$$

Then, the shift function $c_j(t), \forall j \in \{1, 2, \ldots, k\}$ is found via least squares polynomial curve fitting to the data points between the time increments and corresponding spatial location of the shift, such that the shift function is of the form

$$c_j(t) = \sum_{m=0}^{d} a_m t^m$$

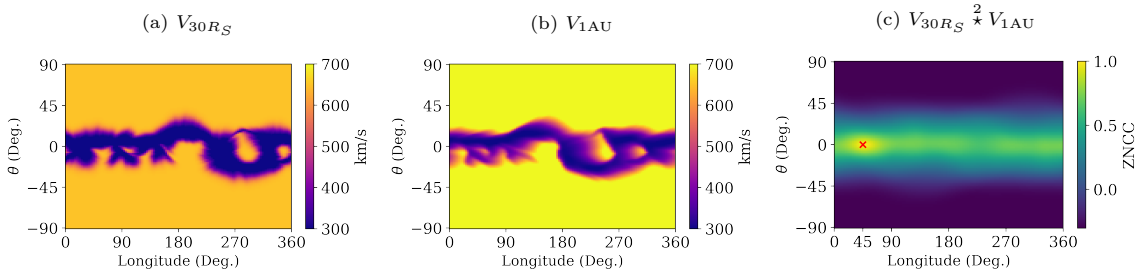

Figure 1: An illustration of discrete circular cross-correlation between the MAS CR2210 velocity results at (a) the initial condition ($30R_S$) and (b) at Earth (1AU). The bi-variate zero-normalized (circular) cross-correlation (ZNCC) between the two MAS velocity results at $30R_S$ and 1AU is shown in Graphic (c). The ZNCC is described by Eq. (32) along with normalizing the two signals by subtracting their mean and dividing by their standard deviations. The shift is found by the maximum of $V_{30R_S} \overset{2}{\star} V_{1\mathrm{AU}}$, which is 45° in longitude and 0° in latitude (purely longitudinal translation).

where d is the degree of the polynomial approximation. To find the vector of real coefficients $\mathbf{a} = [a_0, a_1, \ldots, a_d] \in \mathbb{R}^{d+1}$, we solve the minimization problem

$$\min_{\mathbf{a}\in\mathbb{R}^d} \|\mathbf{T}\mathbf{a} - \mathbf{b}\|_2^2$$

where $\mathbf{b} = [x_{j,\tau_j^\star(t_1)}, \ldots, x_{j,\tau_j^\star(t_K)}] \in \mathbb{R}^K$ is the vector of corresponding spatial location of the shift, and the Vandermonde matrix $\mathbf{T}$ is defined as

$$\mathbf{T} = \begin{bmatrix} 1 & t_1 & \ldots & t_1^d \\ 1 & t_2 & \ldots & t_2^d \\ \vdots & \vdots & & \vdots \\ 1 & t_K & \ldots & t_K^d \end{bmatrix} \in \mathbb{R}^{K\times(d+1)}.$$

To make predictions outside the training interval, we approximate the shift by polynomial extrapolation of $\mathbf{c}(t) = [c_1(t), c_2(t), \ldots, c_k(t)] \in \mathbb{R}^k$. As an illustration, Figure 1 shows the bi-variate discrete circular cross-correlation applied to the MAS CR2210 snapshot at $30R_S$ (the initial condition) and at 1AU. Here, since the flow is steady, the independent variable is the radial distance from the Sun, r, (instead of time t). For this case, the convective shift is 45° in longitude and 0° in latitude. Figure 1 confirms that the translation in the solar wind is purely longitudinal due to the rotation of the Sun. The main idea behind the cross-correlation extrapolation technique is similar to the template-fitting technique studied in [59, 58] where the data is periodic and the template is set to be the initial condition.

3.3. Illustrative Example: Shifted Operator Inference for the Inviscid Burgers' Equation

The one-dimensional inviscid Burgers' equation is of the form of Eq. (24) describing the scalar quantity $u(x,t) : [a,b] \times [0,T] \mapsto \mathbb{R}^+$ with $f[u(x,t),t] = u(x,t)$ and $g[u(x,t),t] = 0$, such that

$$\frac{\partial u(x,t)}{\partial t} + u(x,t)\frac{\partial u(x,t)}{\partial x} = 0 \tag{35}$$

subject to the initial condition $u(x, t=0) = u_0(x)$ with appropriate boundary conditions at $x = a$ and $x = b$. In the moving coordinate frame defined by

$$\tilde{x}(x,t) = x + c(t) \qquad \text{and} \qquad u(x,t) = \tilde{u}(\tilde{x}(x,t),t) \tag{36}$$

Burgers' equation (35) becomes

$$\frac{\partial \tilde{u}(\tilde{x},t)}{\partial t} + \left(\tilde{u}(\tilde{x},t) + \frac{\mathrm{d}c}{\mathrm{d}t}\right)\frac{\partial \tilde{u}(\tilde{x},t)}{\partial \tilde{x}} = 0. \tag{37}$$

This can be written in conservative form as

$$\frac{\partial \tilde{u}(\tilde{x},t)}{\partial t} + \frac{\partial}{\partial \tilde{x}}\left(\frac{1}{2}\tilde{u}(\tilde{x},t)^2 + \frac{\mathrm{d}c}{\mathrm{d}t}\tilde{u}(\tilde{x},t)\right) = 0. \tag{38}$$

Then, by the conservative first-order upwind scheme, we approximate the spatial derivative by

$$\frac{\partial}{\partial \tilde{x}}\left(\frac{1}{2}\tilde{u}(\tilde{x}^{(j)},t)^2 + \frac{\mathrm{d}c}{\mathrm{d}t}\tilde{u}(\tilde{x}^{(j)},t)\right) = \frac{1}{2\Delta\tilde{x}}\left[\tilde{u}(\tilde{x}^{(j)},t)^2 - \tilde{u}(\tilde{x}^{(j-1)},t)^2\right] + \frac{1}{\Delta\tilde{x}}\frac{\mathrm{d}c}{\mathrm{d}t}\left[\tilde{u}(\tilde{x}^{(j)},t) - \tilde{u}(\tilde{x}^{(j-1)},t)\right], \tag{39}$$

where $j = 1, 2, \ldots, n$ denotes the grid index in $\tilde{x}$. Based on the discretization scheme in Eq. (39), we can write the dynamics of Eq. (38) in vector form as

$$\frac{\mathrm{d}\tilde{\mathbf{u}}(t)}{\mathrm{d}t} = \mathbf{A}(t)\tilde{\mathbf{u}}(\mathbf{t}) + \mathbf{H}\left[\tilde{\mathbf{u}}(t) \otimes \tilde{\mathbf{u}}(t)\right] \tag{40}$$

where $\otimes$ denotes the Kronecker product and $\tilde{\mathbf{u}}(t) = [\tilde{\mathbf{u}}(\tilde{x}^{(1)},t), \tilde{\mathbf{u}}(\tilde{x}^{(2)},t), \ldots, \tilde{\mathbf{u}}(\tilde{x}^{(n)},t)]^\top \in \mathbb{R}^n$ denotes the state vector discretized over n spatial points at time t. Here, $\mathbf{H} \in \mathbb{R}^{n \times n^2}$ is the quadratic operator that corresponds to the discrete term $\frac{1}{2\Delta\tilde{x}}\left[\tilde{u}(\tilde{x}^{(j)},t)^2 - \tilde{u}(\tilde{x}^{(j-1)},t)^2\right]$ from Eq. (39). Moreover, $\mathbf{A}(t) \in \mathbb{R}^{n \times n}$ is the linear time-dependent operator corresponding to the discrete term $\frac{1}{\Delta\tilde{x}}\frac{\mathrm{d}c}{\mathrm{d}t}\left[\tilde{u}(\tilde{x}^{(j)},t) - \tilde{u}(\tilde{x}^{(j-1)},t)\right]$ in the moving coordinate frame. In the case where $c(t) \in \mathbb{R}$ is a linear function (so $\frac{\mathrm{d}c}{\mathrm{d}t}$ =const.), meaning the wave is traveling at constant speed, the linear operator is time independent, i.e., $\mathbf{A}(t) \equiv \mathbf{A}$.

The traveling wave speed $c(t)$ can be estimated by the method of characteristics (see Section 3.2.1), where we can rewrite Eq. (35) as the two coupled ODEs

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = u(x(t),t) \qquad \text{and} \qquad \frac{\mathrm{d}u(x(t),t)}{\mathrm{d}t} = 0. \tag{41}$$

Hence, along the characteristic lines the quantity $u(x(t),t)$ remains constant, which can be verified by

$$\frac{\mathrm{d}}{\mathrm{d}t}u(x(t),t) = \frac{\partial}{\partial t}u(x(t),t) + \frac{\mathrm{d}x(t)}{\mathrm{d}t}\frac{\partial}{\partial x}u(x(t),t) = \frac{\partial}{\partial t}u(x(t),t) + u(x(t),t)\frac{\partial}{\partial x}u(x(t),t) = 0. \tag{42}$$

Then, by integrating Eq. (41) the characteristic curves are linear before shock formation. Let the spatial domain of x be discretized on a uniform grid with n points such that $\mathbf{x} = [x^{(1)}, x^{(2)}, \ldots, x^{(n)}] \in \mathbb{R}^n$. From here, we can approximate the shift function via Eq. (30) as a piecewise continuous function:

$$c(t) = \begin{cases} \frac{1}{q-p}\sum_{j=p}^{q} u(x^{(j)}, t=0)t & \text{if } 0 < t < t_s \\ s(t) - s(t_s) + a & \text{if } t > t_s \end{cases} \tag{43}$$

where $t_s \in \mathbb{R}$ is the time of shock formation, $a = \frac{1}{q-p}\sum_{j=p}^{q} u(x^{(j)}, t=0)t_s$, and $s(t) \in \mathbb{R}$ is the shock trajectory.

To demonstrate the sOpInf method, we now consider a specific spatial domain $x \in [0,3]$ and time domain $t \in [0,2]$ along with a Gaussian initial condition $u_0(x) = 0.8 + 0.5\exp\left(-(x-1)^2/0.1\right)$ and periodic boundary conditions. Equation (35) in the regular coordinates is solved numerically via the forward Euler method in time and the conservative upwind scheme in space on an equidistant computational grid with 500 discretization points in space, and 1000 points in time. With this choice, the CFL condition $u(x,t)\frac{\Delta t}{\Delta x} \leq 1$ is satisfied, where Δx and Δt denote the grid spacing in x and t, respectively. The training dataset consists of 80% of the snapshots and the testing dataset consists of 20% of the snapshots. The shock is formed at $t_s = \min_x \left(-\mathrm{d}u_0(x)/\mathrm{d}x\right)^{-1} \approx 0.737$ and the shock trajectory $s(t)$ is computed by Whitham's equal-area rule [66], in which the area of each lobe in the mutli-valued solution is numerically approximated via the trapezoidal rule.

The cross-correlation extrapolation method (Section 3.2.2) approximated $c(t)$ by least-squares linear fitting to the cross-correlation between the training snapshots and the initial condition $u_0(x)$,

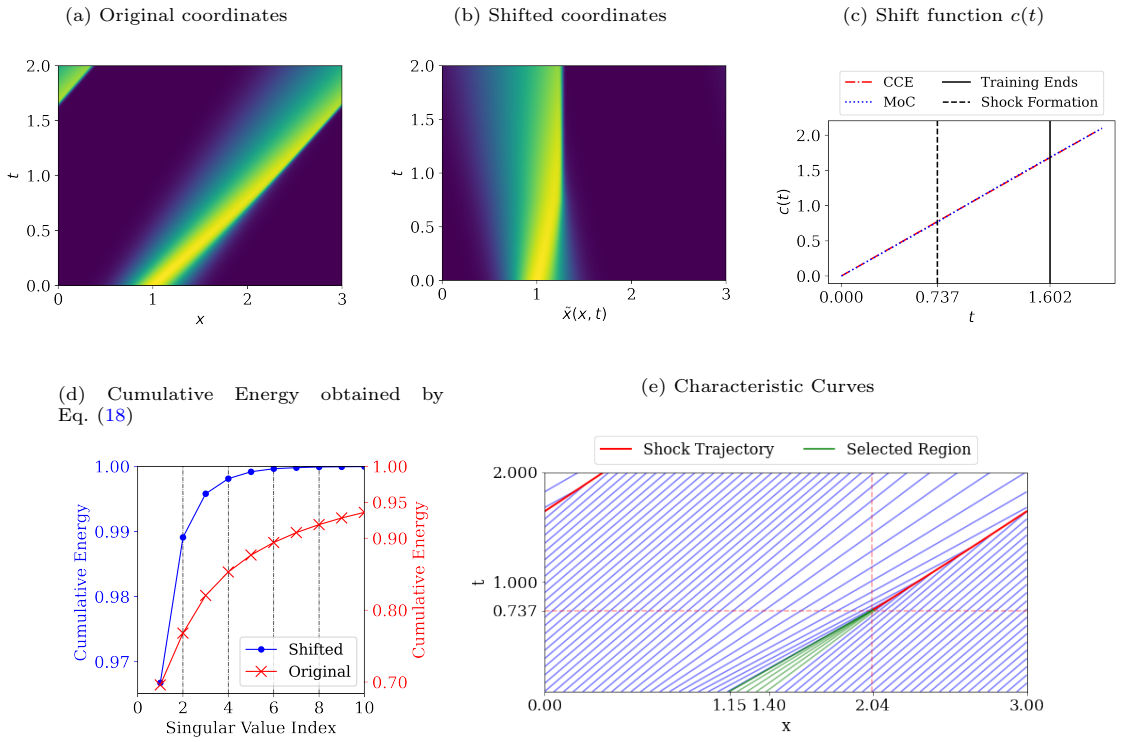

Figure 2: One-dimensional inviscid Burgers' equation with Gaussian initial condition and periodic boundary conditions. The numerical solutions are shown in the (a) original coordinates (x, t) and the (b) shifted coordinates $\tilde{x}(x, t)$. Graphic (c) presents a comparison between finding the wave speed function $c(t)$ using the method of characteristics (MoC) and the cross-correlation extrapolation method (CCE). The results are visually indistinguishable. Graphic (d) shows the cumulative energy computed via the SVD of $\mathbf{U}$ (original coordinates) vs. $\tilde{\mathbf{U}}$ (shifted coordinates), illustrating that the data can be much better approximated in the shifted than in the original coordinates. The characteristic curves including the shock trajectory are shown in Graphic (e).

resulting in $c(t) = 1.05t$. Whereas, via the method of characteristics (Section 3.2.1) the shift function, $c(t)$ described in Eq. (43), is linear only before shock formation, where we compute the mean of characteristics emanating from the spatial interval $[1.15, 1.4]$ before shock formation, see the green characteristic curves in Figure 2e. By the Rankine-Hugoniot jump condition [66], the shock curve $s(t)$, and hence the shift function after shock formation, is non-linear, see the red curve in Figure 2e. However, the absolute difference between the shift function $c(t)$ obtained by the method of characteristics vs. the cross-correlation extrapolation is less than 10^{-2} for all $t \in [0, 2]$, indicating there is not a significant difference between the two methods, see Figure 2c for a visual comparison. With that, we found that using the method of characteristics to find $c(t)$ resulted in up to 2% more accurate ROM results in comparison to the results obtained by the cross-correlation extrapolation linear shift function. By aligning the wave discontinuity at one spatial point, the ROM modes can better approximate the shock. We continue the Burgers' ROM analysis using the shift function computed by the method of characteristics.

Figure 2a shows the inviscid Burgers' equation results on the original coordinate system (x, t) while Figure 2b shows the inviscid Burgers' equation results on the shifted coordinates $(\tilde{x}, t)$. It is apparent from Figure 2b that the dynamics are largely absent of translation in the shifted coordinate frame. The evolution of the initial wave depends solely on shock formation as the discontinuity in the wave sharpens. Figure 2d compares the singular value decay of the solution data matrix $\mathbf{U}$ on the original coordinates and of $\tilde{\mathbf{U}}$ on the shifted coordinates. For the former, the singular value decay is very slow due to the translation properties in the system; a ROM in those coordinates would require a large number of basis functions. In contrast, once the dynamics are absent of translation on the shifted coordinates the system can be approximated using only a few modes.

Figure 3 illustrates the predicted solutions from sOpInf compared to the FOM results for the inviscid Burgers' equation. The sOpInf model is of the form $\dot{\hat{\mathbf{u}}} = \hat{\mathbf{A}}\hat{\mathbf{u}} + \hat{\mathbf{H}}(\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}})$ with $\ell = 9$ modes. The regularization coefficients of the operators $\hat{\mathbf{A}} \in \mathbb{R}^{\ell \times \ell}$ and $\hat{\mathbf{H}} \in \mathbb{R}^{\ell \times \frac{1}{2}\ell(\ell+1)}$ are $\lambda_1 = 1$ and $\lambda_2 = 1$, respectively. The sOpInf results in 0.9999958 Pearson correlation coefficient (PCC) and 1.204×10^{-4} mean relative error (MRE) in comparison to the inviscid Burgers' results, indicating that the sOpInf model is able to capture well the evolution of the high-fidelity Burgers' simulation with only a few modes. On the shifted coordinates we are able to construct a ROM with only 9 modes, meanwhile, in the original coordinates, we would need 89 modes to achieve the same projection error. This demonstrates that the shifting procedure produces significant computational speedups.

We next assess how the amount of training data affects the accuracy of the resulting ROM by training sOpInf on 50%, 60%, 70%, and 80% of the total snapshots, see Figure 4. The numerical results show that—as anticipated—increasing the amount of training data improves the ROMs accuracy. Additionally, the relative error measured in the L_2-norm is bounded by 10% for all four simulations, resulting in an adequate ROM using as low as 50% of the total snapshots for training. To analyze the framework's robustness to noise, we tested the sOpInf methodology on the inviscid Burgers' simulated data with added noise. The results are presented in Appendix A and demonstrate that the method seems to be robust to the addition of moderate levels of Gaussian noise.

4. Numerical Results for MAS and HUX Solar Wind Models

We apply the sOpInf methodology to learn low-dimensional models for the ambient solar wind radial velocity predicted by the HUX and MAS models described in Section 2. We consider a specific event of interest and give some background on that event in Section 4.1. In Section 4.2, we give details on the data and implementation. Section 4.3 and Section 4.4 present the sOpInf two-dimensional steady results trained on HUX and MAS equatorial plane data, respectively. Section 4.5 presents sOpInf three-dimensional steady results trained on the full-Sun MAS results. Section 4.6 compares sOpInf and HUX reconstruction of MAS equatorial plane streamlines. Lastly, Section 4.7 discusses the choice of ROM model form. The public repository `https://github.com/opaliss/Space-Weather-ROM` contains a collection of Jupyter notebooks in Python 3.9 containing the code and data used in this study.

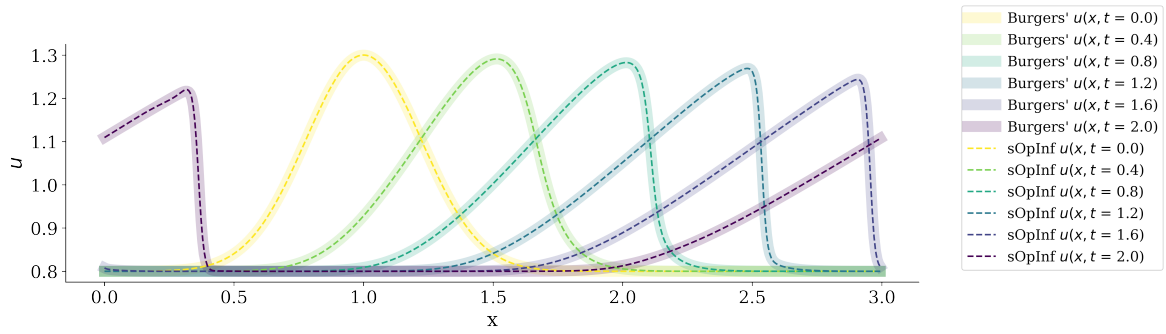

Figure 3: Solutions from the sOpInf model of the form $\dot{\hat{\mathbf{u}}} = \hat{\mathbf{A}}\hat{\mathbf{u}} + \hat{\mathbf{H}}(\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}})$ with $\ell = 9$ modes show very good agreement with the high-fidelity solutions for the one-dimensional inviscid Burgers' equation with Gaussian initial condition and periodic boundary conditions.

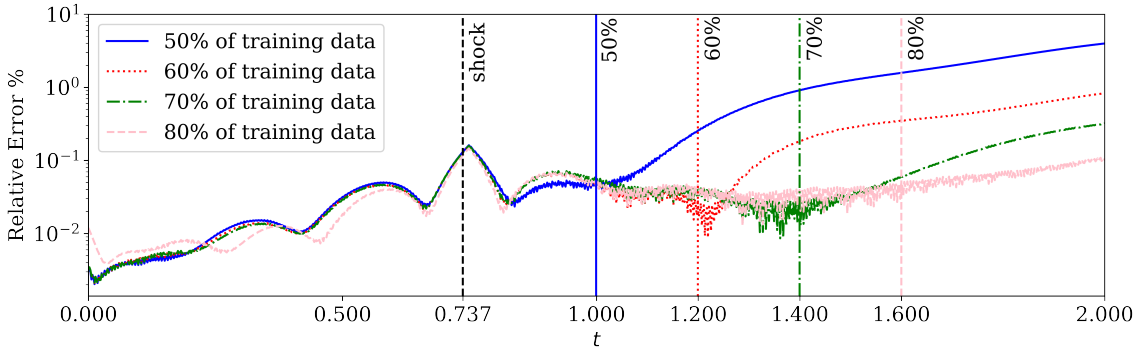

Figure 4: The sensitivity of sOpInf ROM to the amount of training data with 50%, 60%, 70%, and 80% of the total snapshots used for training. The relative error in the L_2-norm is shown as a function of time. The numerical results show that the sOpInf methodology is able to learn adequate ROMs with as low as 50% of the total snapshots utilized for training.

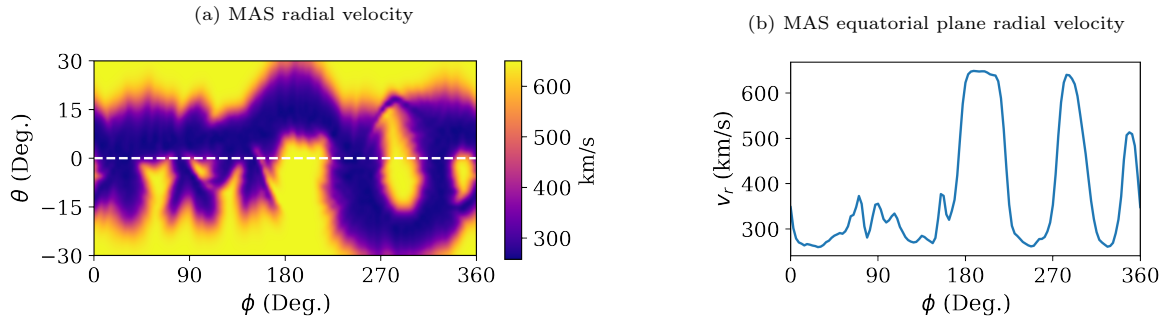

Figure 5: The MAS solar wind radial velocity solution at heliocentric distance $r = 30R_S$ during CR2210 (a) ranging from $\theta \in [-30°, 30°]$ in latitude and (b) the equatorial ($\theta_{e.p.} = 0°$) velocity profile. The solar wind radial velocity at the equator has three main peaks, originating from coronal holes, which make this period a great candidate to study large-scale structure in the solar wind.

4.1. Physical Relevance of Data

We focus on Carrington Rotation (CR) 2210, which occurred from 26 October to 23 November 2018, during a solar minimum. Figure 5a shows the heliospheric solar wind radial velocity MAS results for CR2210 in the latitude region of $\theta \in [-30°, 30°]$ and Figure 5b presents the equatorial solar wind radial velocity profile. As observed in Figure 5, during CR2210, the solar wind radial velocity at the equator has three main peaks, essentially making this period a great candidate to study large-scale structure in the solar wind. Although not shown here, the origin of the fast wind (ranging from 500 to 700 km/s) is from equatorial coronal holes located at approximately $180° - 210°$, $280° - 300°$, and $330° - 360°$ in longitude at the source surface (see [8, Fig. 8(a)] for a synoptic view at $2R_S$ of the coronal hole regions). Although fast streams commonly originate in large coronal holes, slow streams come from various coronal sources (e.g., coronal hole boundaries, coronal loops, etc.). Another reason to consider this data is that this Carrington period is well-studied in literature due to Parker Solar Probe (PSP) reaching its first perihelion pass of $35.7R_S$ after its first Venus gravity assist on 6 November 2018, as it broke records by becoming the closest spacecraft to the Sun. Moreover, the authors in [57] showed that the thermospheric MAS solar-wind speed results highly match the observations made by PSP during CR2210. Thus, while we do not present a comparison with *in-situ* solar-wind observations we can treat the MAS results as a physically meaningful representation of the solar wind.

4.2. Data and Implementation Details

4.2.1. MAS Data

The MAS model is described in detail in Section 2.1. The MAS model solar wind velocity results are time-stationary in spherical coordinates. The data covers the entire domain in longitude $0° \leq \phi \leq 360°$, latitude $-90° \leq \theta \leq 90°$, and radial axis (a.k.a. heliocentric distance) $0.14\text{AU} \leq r \leq 1.1\text{AU}$. The MAS simulation results are on a rectangular grid with $n_\phi = 128$ uniformly spaced points in Carrington longitude, $n_\theta = 111$ uniformly spaced points in heliographic latitude, and $n_r = 140$ points on a non-uniformly spaced grid in the radial axis. Hence, the snapshot data dimension is $n_x = n_\phi \times n_\theta = 14,208$ where we treat r as the independent variable. A convergence check with a higher-resolution grid found that this resolution is sufficient for physical accuracy. The MAS coronal and heliospheric models (implemented in FORTRAN) were run with medium and high resolution by the authors of [1], which took approximately four and 28 hours of wall-clock time using four NVIDIA RTX 2080Ti GPUs, respectively [13]. We derive data-driven ROMs from the medium-resolution MAS heliospheric simulation which is publicly available at PSI's web page [2]. The MAS training datasets contain 70% of the snapshots, with 98 and 42 snapshots for training and testing, respectively. The training domain is from 0.14 to 0.82AU and the testing domain is from 0.82 to 1.1AU.

4.2.2. HUX Data

The HUX model is described in detail in Section 2.2. To simulate the HUX model at the heliographic solar equatorial plane we numerically solve the ODE in Eq. (12) via the forward explicit Euler's method in time and setting the initial condition to the MAS velocity profile results at $30R_S \approx 0.14$AU. The HUX dataset is two-dimensional with $n_\phi = 128$ and $n_r = 140$ (satisfying the CFL condition) on the same domain as listed above for the MAS results. On the equatorial plane ($\theta_{\text{e.p}} = 0$), the angular frequency of the Sun is $\Omega_{\text{rot}}(\theta_{\text{e.p}}) = \frac{2\pi}{25.38}1/\text{days}$. The HUX dataset took 0.03s to simulate on a MacBook Pro 2.3 GHz Quad-Core Intel Core i7 processor with 16 GB RAM. We note that the HUX model is already computationally efficient. We train a ROM for HUX merely as a introductory example since it produces the same physics (advection-dominated solutions) that we expect in much more expensive solvers, and not to show any computational improvements. The HUX training and testing domains are identical to the MAS equatorial training domain, see details in Section 4.2.1.

4.2.3. ROM Implementation

The MAS and HUX numerical results are on a non-uniformly spaced grid in the radial axis, consequently, we approximate the derivatives of the training data with respect to r, i.e. $\frac{\mathrm{d}}{\mathrm{d}r}\hat{\mathbf{u}}$ in Eq. (19), via a second-order accurate central difference in the interior points and second-order accurate one-sided (forward and backward) difference at the boundaries. For the case of uniform grid meshing, e.g. Burgers' equation ROM presented in Section 3.3, we used a sixth-order finite difference scheme. We simulate the ROM via an implicit multi-step variable method based on a backward differentiation formula using the `scipy.integrate.solve_ivp()` Python function. The ROM regularization coefficients, λ_1 and λ_2, are chosen from the logarithmically spaced set, i.e. $\{10^0, 10^1, 10^2, \ldots, 10^{10}\}$, such that the best coefficients minimizes the relative error measured via the L_∞-norm over the training regime.

4.3. HUX Equatorial Plane Numerical Results

We apply the sOpInf framework to learn a ROM of the HUX CR2210 equatorial plane radial velocity. As derived in Section 2.2.1, the HUX underlying equation is

$$-\Omega_{\text{rot}}\,(0)\,\frac{\partial v_r(r,\phi)}{\partial \phi} + v_r(r,\phi)\frac{\partial v_r(r,\phi)}{\partial r} = 0,$$

where r, ϕ are the independent variables. To begin, we shift the HUX dynamics to a moving coordinate frame defined by

$$\tilde{\phi}(r,\phi) = \phi + c(r) \qquad \text{and} \qquad v_r(r,\phi) = \tilde{v}_r(\tilde{\phi}(r,\phi), r).$$

The shift function $c(r)$ can be learned via either the method of characteristics (Section 3.2.1) or the cross-correlation extrapolation method (Section 3.2.2), in particular, the circular uni-variate cross-correlation method described in Eq. (31). The linear-fit cross-correlation shift function resulted in $c(r) = -51.711^\circ r + 7.032^\circ$. Our numerical studies found that there is no substantial differences between the numerical results of the two methods. Following the steps described in Section 3.2.1, the HUX characteristic curves are derived by the following two coupled ODEs:

$$\frac{\mathrm{d}}{\mathrm{d}r}v_r(\phi(r), r) = 0 \quad \text{and} \quad \frac{\mathrm{d}\phi(r)}{\mathrm{d}r} = -\frac{\Omega_{\text{rot}}(0)}{v_r(\phi(r), r)}. \tag{44}$$

Then, by integration of Eq. (44), the characteristics before shock formation are straight lines described by

$$\phi(r) = \phi - \frac{\Omega_{\text{rot}}(0)}{v_{r_0}(\phi)}(r - r_0),$$

where $r_0 = 0.14$AU. The HUX characteristic curves are also called the *ballistic* approximation, which assumes that each spiral field line of plasma continues at a constant speed throughout the

Table 1: Comparison of the sOpInf ROM performance for the test case CR2210. Given are the mean/median/maximum relative error (RE) measured in percent and the Pearson correlation coefficient (PCC) comparing the ROM with the respective high-fidelity models (HUX, MAS-2D, MAS-3D) for both training and testing datasets.

Model	Regime	RE mean	RE median	RE max.	PCC
HUX Equatorial Plane (2D)	Training	0.286	0.143	3.379	0.99991
	Testing	0.526	0.385	4.114	0.99956
MAS Equatorial Plane (2D)	Training	0.449	0.229	5.564	0.99980
	Testing	1.264	0.849	8.235	0.99901
MAS Full Sun (3D)	Training	0.451	0.334	7.731	0.99969
	Testing	0.539	0.353	20.492	0.99964

heliosphere [52]. After obtaining the characteristic curves, we are able to approximate the shift function by

$$c(r) = \begin{cases} \frac{1}{q-p}\sum_{j=p}^{q} \frac{-\Omega_{\text{rot}}(0)}{\mathbf{v}(\phi_j, r_0)}(r - r_0) & \text{if } r_0 < r < r_s \\ s(r) - s(r_s) + a & \text{if } r > r_s \end{cases}$$

so that r_s is the radial position where the characteristics first intersect, $s(r)$ is the shock trajectory, and $a = \frac{1}{q-p}\sum_{j=p}^{q} \frac{-\Omega_{\text{rot}}(0)}{\mathbf{v}(\phi_j, r_0)}(r_s - r_0)$. The indices p, q can include the whole spatial domain or instead a bounded spatial interval to track specific regions of the initial wave. For CR2210, we limit p, q to include the characteristics emanating from a main equatorial high solar wind peak, originating from an equatorial coronal hole, on the longitudinal interval $[180°, 260°]$. There are three main shock curves, and we choose to follow the first shock curve that emerged at $r_s = 0.344$AU and $\phi_s = 211.737°$. The learned sOpInf ROM is of the form

$$\dot{\hat{\mathbf{v}}} = \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}), \qquad \hat{\mathbf{H}} \in \mathbb{R}^{\ell \times \frac{1}{2}\ell(\ell+1)}$$

and it is able to sufficiently model the dynamics of the HUX model with only $\ell = 4$ modes. A practical implementation of operator inference requires regularization, and for the least-squares fitting we found $\lambda = 10^3$ to give good results. The comparison between sOpInf and HUX velocity profiles is provided in Figure 6. The figure shows that the advective solutions are well approximated both in the training regime until $r = 0.82$AU and in the testing regime, where the ROM is fully predictive. This conclusion is also supported by Table 1 where we provide the mean/median/maximum relative error and the Pearson correlation coefficient comparing the HUX solutions with the ROM solutions, both in the testing and training regime. The error measures show that a sOpInf ROM can sufficiently predict the HUX dynamics while reducing the dimensionality of the problem from $n_\phi = 128$ to $\ell = 4$, i.e., a factor of 32 reduction of state-space dimension.

4.4. MAS Equatorial Plane Numerical Results

We apply the sOpInf framework to learn a ROM for the MAS CR2210 equatorial plane velocity field. Since the MAS Eqs. (1)–(6) are not in the form of Eq. (24), the method of characteristics can not be applied to approximate the shift function $c(r)$; instead, we use the cross-correlation extrapolation method, which resulted in $c(r) = -54.98°r + 7.39°$, where r is measured in AU. Figure 7 shows the MAS equatorial plane heliospheric results on the original and shifted polar coordinates along with the cumulative singular value energy criteria described in Eq. (18) in each coordinate system. Evidently, we see that shifting the snapshots to a moving coordinate frame creates a faster singular value decay, leading to an accurate representation of the shifted data with far fewer modes. This observation is in line with the results we showed for Burgers' equation in Figure 2d above.

The results shown in this section are for the ROM model form

$$\dot{\hat{\mathbf{v}}} = \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}), \qquad \hat{\mathbf{H}} \in \mathbb{R}^{\ell \times \frac{1}{2}\ell(\ell+1)}$$

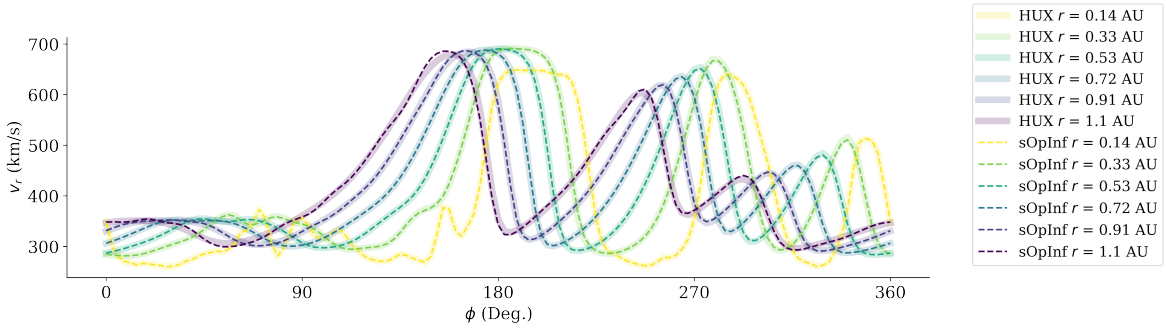

Figure 6: The HUX solar wind radial velocity results at the heliographic equatorial plane for CR2210 along with sOpInf quadratic ROM results. The sOpInf ROM aligns very well with the data past the training interval; the snapshots at $r = 0.91$AU and $r = 1.1$AU are testing data where the ROM is fully predictive.

(a) Original Coordinates

(b) Shifted Coordinates

(c) Cumulative Energy obtained by Eq. (18)

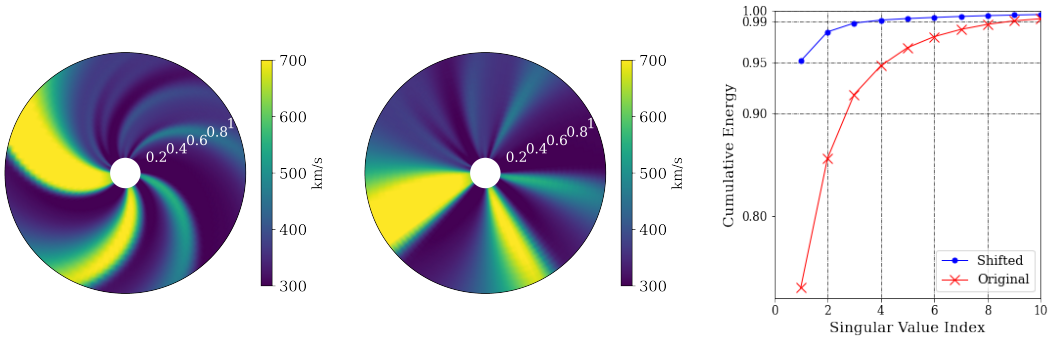

Figure 7: (a) The MAS solar wind radial velocity results at the equatorial plane for CR2210 from 0.14AU to 1.1AU. (b) The solar wind data in shifted coordinates eliminating the translational properties caused by the Sun's rotation. (c) The Singular value cumulative energy of the data in the original and shifted coordinates. The singular values decay more rapidly in the shifted coordinates, indicating that the ROM will require less modes in the shifted coordinates.

with $\ell = 9$ modes. This model form provided the best overall results in training and extrapolation, compared to other model combinations including linear and constant terms, see Section 4.7 for further discussion. The regularization coefficient for computing the operator $\hat{\mathbf{H}}$ is $\lambda = 10^5$. Figure 8 visually demonstrates that sOpInf is capable of accurately approximating the MAS equatorial results, where the snapshots at $r = 0.91$AU and $r = 1.1$AU are in the fully predictive regime of the ROM. We highlight that the MAS data contains more complex features than the HUX data, specifically in the region $0° \leq \phi \lesssim 120°$ where small localized wave structures exist. Nevertheless, sOpInf covers those equally well as the HUX data in the previous section. Figure 9 presents are more qualitative assessment of the relative error between the sOpInf ROM and MAS velocity fields, which shows that the relative error is less than 8.3% in the entire domain. The mean/median/maximum relative error and PCC in the testing and training regime are again provided in Table 1 above.

4.5. 3-D Full-Sun MAS Numerical Results

We showcase sOpInf trained on MAS three-dimensional steady-state velocity results. As mentioned previously, the MAS Eqs. (1)–(6) are not in the form of Eq. (24), hence, the method of characteristics is not a valid choice in approximating the shift function $\mathbf{c}(r) \in \mathbb{R}^2$. Therefore, we turn to the cross-correlation extrapolation method described in Section 3.2.2. More specifically, the full-Sun MAS

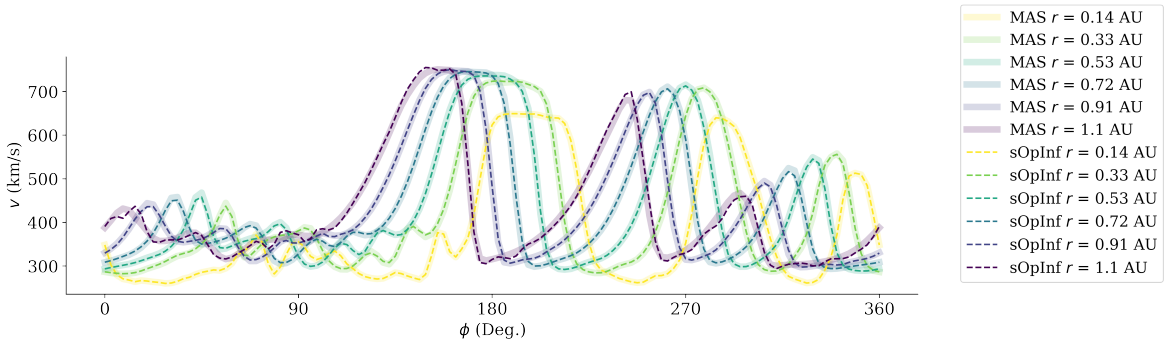

Figure 8: The MAS solar wind radial velocity results at the heliographic equatorial plane for CR2210 along with sOpInf quadratic ROM results. The sOpInf ROM aligns very well with the data past the training interval; the snapshots at $r = 0.91$AU and $r = 1.1$AU are testing data where the ROM is fully predictive.

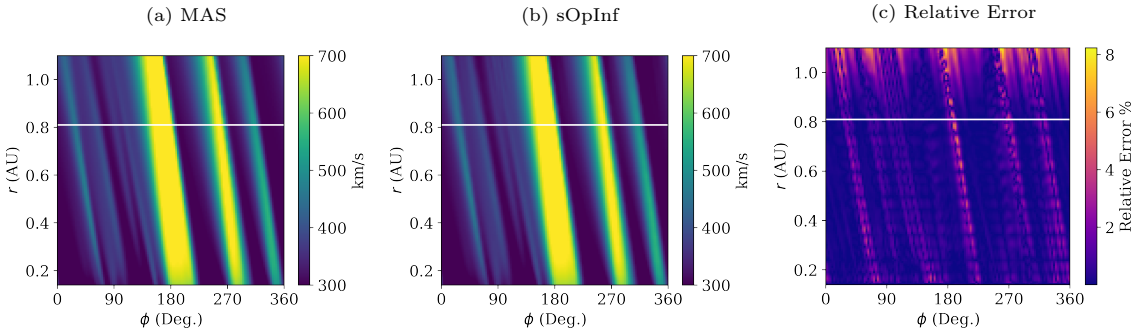

Figure 9: A comparison between (a) the MAS solar wind radial velocity solution at the heliographic equatorial plane for CR2210 and (b) the learned quadratic sOpInf ROM results with $\ell = 9$ basis modes. The relative error between MAS and sOpInf results is illustrated in sub-figure (c).

snapshots in matrix form, $\mathbf{v}(r) \in \mathbb{R}^{n_\phi \times n_\theta}$, are bi-variate in Carrington longitude (ϕ) and latitude (θ), with n_ϕ, n_θ mesh points in longitude and latitude, respectively. The shift function $\mathbf{c}(r) \in \mathbb{R}^2$ is found via the bi-variate circular cross-correlation defined by Eqs. (32)–(33) with $k = 2$. The bi-variate circular cross-correlation is applied between each snapshot and the velocity profile at the initial condition ($30R_S$). Since the translation in the MAS solar wind results is due to the Sun's rotation, the shift function can be expressed as $\mathbf{c}(r) = c(r)\hat{\mathbf{e}}_\phi$, where $\hat{\mathbf{e}}_\phi$ is the unit vector in longitude direction, and $c(r) \in \mathbb{R}$ is the shift in longitude, where we computed $c(r) = -52.44°r + 6.94°$ with r measured in AU. The results in this section are for a ROM of the form

$$\dot{\hat{\mathbf{v}}} = \hat{\mathbf{A}}\hat{\mathbf{v}} + \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}) + \hat{\mathbf{B}}$$

where $\hat{\mathbf{B}} \in \mathbb{R}^\ell$, $\hat{\mathbf{A}} \in \mathbb{R}^{\ell \times \ell}$, $\hat{\mathbf{H}} \in \mathbb{R}^{\ell \times \frac{1}{2}\ell(\ell+1)}$ with only $\ell = 8$ modes. The regularization coefficient for computing the operator $\hat{\mathbf{B}}$ and $\hat{\mathbf{A}}$ is $\lambda_1 = 10^4$ and the regularization coefficient for $\hat{\mathbf{H}}$ is $\lambda_2 = 10^8$. Figure 10a shows the relative error between the two velocity fields and the mean/median/max relative error in the testing and training regime is shown in Table 1. For a visual comparison, Figure 10b shows a comparison between the sOpInf reconstructed and predicted two-dimensional full-Sun snapshots; the visual comparison and error estimates indicate that sOpInf can successfully reproduce the high fidelity full-Sun MAS dataset, where $n_x = n_\phi \times n_\theta = 14,208$ with only $\ell = 8$ modes, leading to a substantial reduction in the model's dimensionality.

In practice, the sOpInf framework can be employed to speed up the MAS computational time by setting the MAS heliospheric outer boundary condition to 0.82AU (which is 70% of the current computational domain) and run the time-dependent MAS simulation until it relaxes to steady-state. Then, the steady-state MAS snapshots are used as training data for sOpInf, which takes 0.355 seconds to simulate from $30R_s$ up to 1.1AU on a MacBook Pro 2.3 GHz Quad-Core Intel Core i7 processor with 16 GB RAM (for 3-D full-Sun simulation). If we assume that running MAS on 70% of the computational domain would take 70% of the MAS current computational time, then we would be able to speed up the MAS computational time by approximately 1.2 hours and 8.4 hours for the medium and high resolution runs, respectively (i.e. save 30% of the MAS computational time). Another important sOpInf speed-up contribution can be in the case when one is interested in studying the solar wind dynamics much further in the heliosphere (e.g. conditions in the vicinity of Mars or Jupiter). In this case, one can use the MAS simulation up to 1.1AU for training sOpInf and use the reduced model to predict up to 5AU, resulting in a more significant speed up. It is important to mention that the MAS model solves for several plasma flow quantities, i.e. Eqs. (1)–(6), and sOpInf is currently only solving for the radial velocity component. Thus, a direct comparison of their run-time can be equivocal.

4.6. A Comparison of Surrogate Model Accuracy via Equatorial Plane Streamlines

Given the two surrogate models of MAS, namely HUX (a reduced-physics approximation) and sOpInf (a data-driven ROM trained on MAS), we are interested in comparing their accuracy as surrogates of MAS. We do so by drawing the equatorial streamlines for each model. The streamlines of a flow field are curves that are tangential to the local velocity vector. In Figure 11(a-c), the streamlines are mapped from the inner-heliosphere at 0.14AU to 1.1AU. The shape of the streamlines depends on the velocity field, such that the fast solar wind results in less tightly wound lines than the slow solar wind. At regions where the streamlines interact, a *compression wave* is formed, whereas, at regions where the streamlines are distant there is a *rarefaction wave*. At these regions of compression and rarefaction, the solar wind streams go through substantial changes in density and flow speed [26]. For a better understanding of the mapped streamline accuracy, Figure 11(e-f) presents a histogram of the mapped streamlines longitude difference at 1.1AU along with plotting the cumulative distribution function (CDF) in Figure 11d of the streamlines longitude difference at 1.1AU for each surrogate model: HUX and sOpInf. The MAS in comparison to HUX and sOpInf trained on MAS mean/median/maximum and standard deviation (SD) of the streamline longitude error are presented in Table 2. The streamline mean longitude absolute error at 1.1AU of the HUX model is a factor of 12 larger than the ones associated with sOpInf. The numerical results show that the sOpInf model is

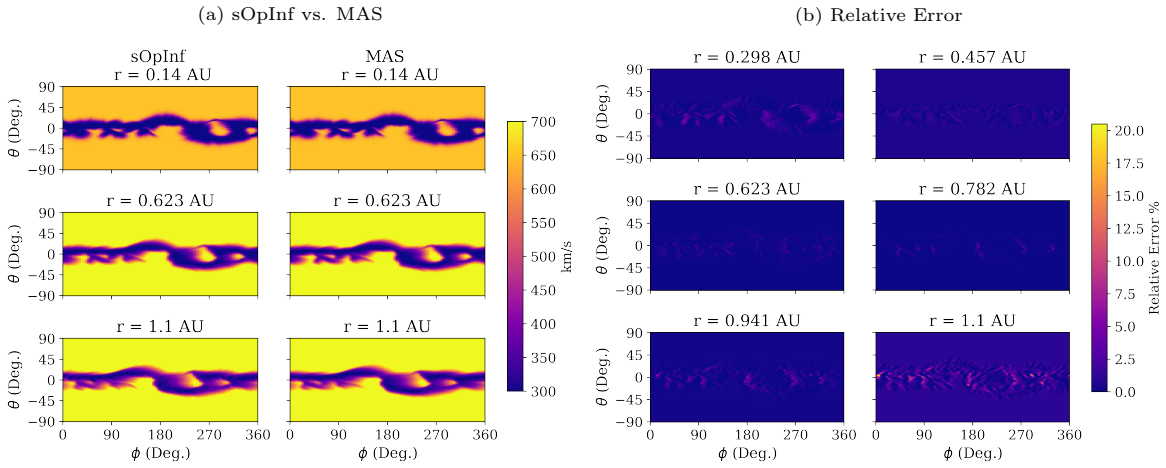

Figure 10: Graphic (a) shows the MAS solar wind velocity results (right column) and sOpInf model of the form $\dot{\hat{\mathbf{v}}} = \hat{\mathbf{A}}\hat{\mathbf{v}} + \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}) + \hat{\mathbf{B}}$ results (left column) with $\ell = 8$ modes. The training ends at 0.82AU, hence, the velocity profile at $r = 1.1$AU (last row) is in the purely predictive regime. Graphic (b) presents the relative error of learned full-Sun sOpInf ROM vs. MAS. The sOpInf ROM results at $r = 0.968$AU and $r = 1.1$AU (last row) are in the purely predictive regime. The ROM shows good qualitative and quantitative agreement with the MAS solution, yet can be evaluated at a much lower cost.

Table 2: Streamline longitude absolute error (AE), measured in degrees, at 1.1AU of the surrogate models HUX and sOpInf in comparison to MAS. That snapshot is in the fully predictive regime of the ROM, showing that the sOpInf ROM can predict well outside the training interval and provides a better surrogate model than the reduced-physics HUX model.

Model Comparison	AE mean	AE median	AE max.	AE SD
HUX vs. MAS	2.186°	1.977°	5.047°	1.463°
sOpInf vs. MAS	0.172°	0.155°	0.457°	0.113°

a more accurate approximation of the MAS model in comparison to the reduced-physics HUX model. Moreover, the sOpInf framework can account for the solar wind dynamics in three-dimensional space, whereas HUX is strictly two-dimensional.

4.7. Additional Considerations when Choosing the Operator Model Form

4.7.1. Comparing Models With the Same Number of Modes

The choice of polynomial ROM form for the HUX and MAS dataset is an approximation of the governing equations (unlike the inviscid Burgers' example in Section 3.3) since both models have nonpolynomial terms. The above Sections 4.3–4.5 showcase the ROM model form that performed the best in the testing regime, i.e. purely-linear, purely-quadratic, or a combination thereof. Here, we present the results of a detailed investigation of how other polynomial ROM forms performed on each dataset. Figure 12 compares the state error in the fully predictive (testing) regime for each model: (12a) shows the HUX-2D equatorial plane results, (12b) shows the MAS-2D equatorial plane results, and (12c) shows the MAS-3D full-Sun results. In each case, we consider strictly-linear, strictly-quadratic, and linear-quadratic plus a constant term model forms in our analysis. Figure 12a shows a comparison of three different model forms for HUX dataset, in which strictly-quadratic and linear-quadratic plus constant term ROMs perform better than the strictly-linear ROM. There is not a significant difference between the two quadratic forms, yet since the strictly-quadratic ROM has fewer model parameters and resulted in a slightly better relative error in the testing regime, we choose to employ a strictly-quadratic ROM form. For the MAS-2D dataset, the results in Figure 12b show that

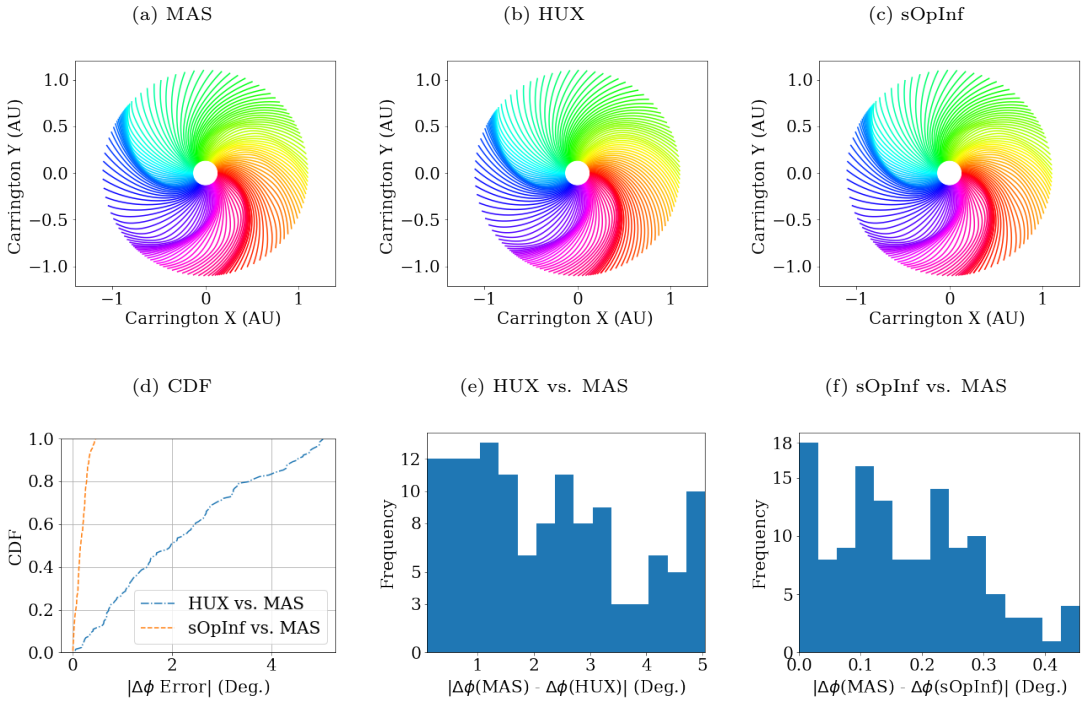

Figure 11: The solar wind streamlines (or Parker spiral) for the CR2210 equatorial plane is shown using the velocity results of (a) MAS, (b) HUX, and (c) sOpInf trained on MAS. Sub-figure (d) shows the cumulative distribution function (CDF) of the streamlines longitude absolute difference at 1.1AU, where sOpInf outperformed HUX in approximating the MAS solar wind streamlines. The longitude absolute difference between the streamlines are shown for (e) HUX vs. MAS and (f) sOpInf vs. MAS.

the strictly-quadratic model outperformed the other two model forms in the testing regime. Lastly, for the MAS-3D dataset, Figure 12c shows that the quadratic model with linear and constant terms outperforms the strictly-linear and strictly-quadratic ROMs with the same amount of modes.

4.7.2. Comparing Models with Similar Computational Cost

The main question we seek to answer: *Is it better to have a linear ROM with larger ℓ or a quadratic ROM with smaller ℓ?* The cost of simulating the sOpInf ROM depends on the number of ROM parameters in the matrices on the right-hand side of Eq. (21). That number of parameters is determined by both the model form and the reduced basis dimension ℓ and determines the models computational cost. As mentioned in Section 3.1(III), we use the compact Kronecker product in the sOpInf ROM. Consequently, the number of model parameters $d(\ell)$ (i.e., parameters in the system matrices) is

$$d(\ell) = \begin{cases} \ell \times \ell & \text{if } \dot{\hat{\mathbf{v}}} = \hat{\mathbf{A}}\hat{\mathbf{v}} \\ \ell \times \frac{1}{2}\ell(\ell+1) & \text{if } \dot{\hat{\mathbf{v}}} = \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}) \\ \ell \times (\ell + \frac{1}{2}\ell(\ell+1) + 1) & \text{if } \dot{\hat{\mathbf{v}}} = \hat{\mathbf{A}}\hat{\mathbf{v}} + \hat{\mathbf{H}}(\hat{\mathbf{v}} \otimes' \hat{\mathbf{v}}) + \hat{\mathbf{B}} \end{cases} \tag{45}$$

for the different ROM forms we investigated. We thus compare the quadratic model forms chosen in Section 4.3–4.5 to a linear ROM with a comparable number of model parameters $d(\ell)$ in Figure 13. That figure shows results for (13a) HUX-2D ROM presented in Section 4.3, (13b) MAS-2D ROM presented in Section 4.4, and (13c) MAS-3D ROM presented in Section 4.5. As seen in all three examples, the chosen quadratic model forms perform better than the purely-linear model form in the

(a) HUX-2D ROM Form Comparison

(b) MAS-2D ROM Form Comparison

(c) MAS-3D Full-Sun ROM Form Comparison

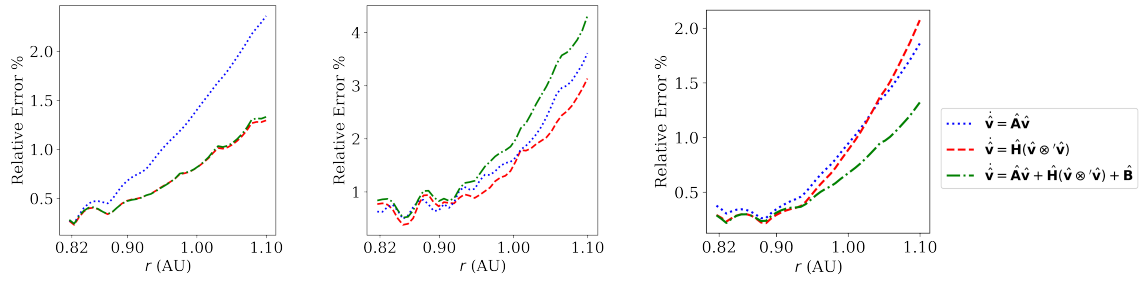

Figure 12: A comparison of three ROM model forms, i.e. purely linear, purely quadratic, and quadratic with linear and constant term, relative error measured via the L_2-norm in the testing regime. The three models are trained on 70% of the (a) HUX-2D equatorial data, (b) MAS-2D equatorial data, and (c) MAS-3D full-Sun data.

testing regime using a comparable number of model parameters (and hence comparable computational cost), more specifically:

- For the HUX-2D example, we set $\ell = 4$ with a strictly-quadratic model, so a linear model with $\ell \approx 6$ would have a comparable number of model parameters. The quadratic model performs better with approximately the same number of model parameters.
- For the MAS-2D example, we set $\ell = 9$ with a strictly-quadratic model, so a linear model with $\ell \approx 20$ would have a comparable number of model parameters. The quadratic model is more accurate with approximately the same number of model parameters.
- For the MAS-3D example, we set $\ell = 8$ with a linear-quadratic plus constant term model, so a linear model with $\ell \approx 19$ would have a comparable number of model parameters. The quadratic model performs better with approximately the same number of model parameters.

The above numerical evidence highlights the importance of including quadratic nonlinearity in the sOpInf ROM. From another perspective, adding a quadratic term to the model allows us to have a lower ROM dimension ℓ than if only linear terms were present.

5. Conclusion

We proposed a reduced-order modeling strategy that uses simulated data to learn low-dimensional models for efficient solar wind predictions. The method leverages physical knowledge in that it first seeks to detect a spatial shift in the data/model (arising from advection) either through the method of characteristics or the fully data-based cross-correlation method. Given that shift, the system is then transformed into a moving coordinate frame, where a ROM can efficiently be learned via operator inference. The numerical results showed that for the full-Sun MAS simulations, a ROM with $\ell = 8$ modes was sufficient to accurately predict the solar winds, yet produced significant computational speedup compared to the full-order model. From a surrogate modeling perspective, we investigated and compared the accuracy of two surrogates for the MAS model: a reduced-physics approximation (HUX) and the proposed sOpInf ROM approximation. We found that the latter is a much more accurate model than HUX; therefore, it is worth investigating ROM approaches for solar physics applications. Although we developed the sOpInf methodology to learn ROMs from simulated solar wind data, we found that the sOpInf framework is robust to moderate levels of noise, which is promising as one hopes to use sOpInf for noisy observational data. Additionally, when considering which sOpInf ROM model form should be chosen, we found that smaller quadratic ROMs perform better than

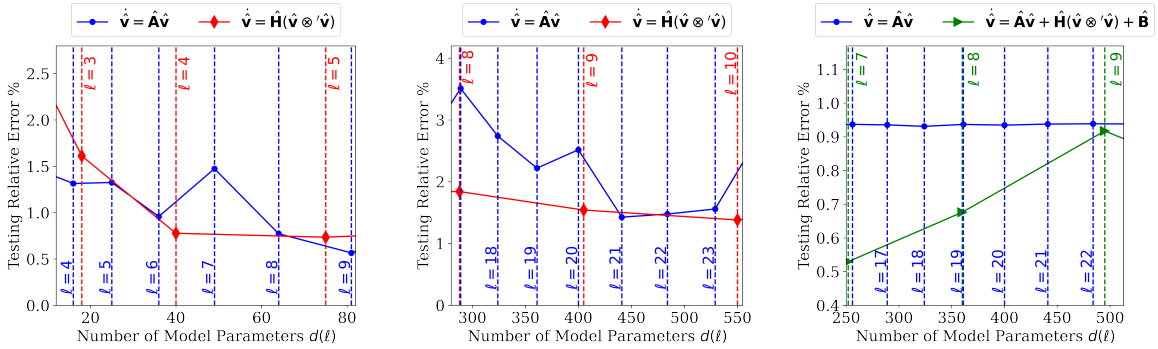

Figure 13: A comparison of quadratic ROM forms, i.e. purely-quadratic and linear-quadratic plus constant term, to purely-linear ROM form via the testing relative error measured by the Frobenius norm. The numerical results for (a) HUX-2D, (b) MAS-2D, and (c) MAS-3D, show that the quadratic ROM forms are generally more accurate than the strictly-linear ROMs learned with a comparable number of model parameters.

larger, purely-linear ROMs in the testing regime (with comparable number of model parameters), highlighting the importance of quadratic terms in the ROM. While applied to use cases where solar wind velocities are most relevant, our methodology is applicable to forecasting additional solar wind quantities such as the density, pressure, etc. These models are our focus of future work. Moreover, while most quadratic forms of the ROM were sufficient to represent the physics with good accuracy, we expect more nonlinearly behaving systems to benefit from additional variable transformations (and lifting approaches similar to [28, 49]).

A long-term goal of our project is to solve uncertainty quantification problems, by efficiently assessing the impact of model input uncertainties, such as uncertain model coefficients, boundary conditions, and initial conditions, for which we anticipate our ROMs to be very useful. The ROMs can substantially accelerate ensemble methods, e.g. the most direct approach of Monte Carlo simulations and Bayesian inference, which are highly valuable in space weather operational forecasting.

6. Acknowledgement

This research was partially supported by the National Science Foundation under Award 2028125 for "SWQU: Composable Next Generation Software Framework for Space Weather Data Assimilation and Uncertainty Quantification".

Appendix A. Extending Shifted Operator Inference to Applications with Noisy Data

We developed the sOpInf methodology to predict the ambient solar wind from simulated data, yet the sOpInf framework extends to a wide class of advection-dominated systems described on a periodic domain and can be trained directly from observational data which is always noise-corrupted. We thus tested the sOpInf sensitivity to noise on the inviscid Burgers' equation example described in Section 3.3 with Gaussian noise added to each snapshot entry. Following the work by [21], the noise is drawn from a $\mathcal{N}(0, \nu^2)$ with $\nu = \zeta \cdot (\max_x u(x,0) - \min_x u(x,0)) = \zeta \cdot (1.3 - 0.8) = \zeta/2$, where the coefficient ζ is the noise level. The same Gaussian initial condition, periodic boundary conditions, and first-order finite difference numerical solver described in Section 3.3 are used to generate the training and testing snapshots, where 80% of the total snapshots are used for training.

We generate noisy snapshots with various levels of noise $\zeta = 1, 2, \ldots, 20\%$, and for each noise level compute the shift function $c(t)$ via the cross-correlation extrapolation technique fitting a linear

polynomial (see Section 3.2.2). Figure A.14a shows the cross-correlation extrapolation linear shift function $c(t)$ slope as we vary the level of noise $\zeta = 1, 2, \ldots, 20\%$. We found that the cross-correlation technique is robust to noise as the shift function remained within $c(t) = (1.05 \pm 0.012)t$ for $\zeta = 1, 2, \ldots, 20\%$. Note that $c(t) = 1.05t$ is the linear shift function for the noiseless data as mentioned in Section 3.3.

Figure A.14b shows the singular values of the noisy and noiseless Burgers' training snapshots on the shifted and original coordinate frame for $\zeta = 2\%$. In agreement with Figure 2d, we see in Figure A.14b that the singular values decay faster in the shifted coordinates (in comparison to the original coordinates). Next, we compute the POD basis $\mathbf{V}_\ell = [\mathbf{v}_1, \ldots, \mathbf{v}_\ell] \in \mathbb{R}^{n \times \ell}$, shown in Figure A.14c. Inspecting the POD modes, we choose to keep $\ell = 5$ POD basis functions, since after the fifth mode the POD modes are polluted with noise. This choice of ROM dimension $\ell = 5$ is consistent with the singular values of the noisy data in the shifted coordinates, which plateau after $\ell = 5$. The projected snapshots $\hat{\mathbf{U}} = \mathbf{V}_\ell^\top \tilde{\mathbf{U}}$, where the rows of $\hat{\mathbf{U}} \in \mathbb{R}^{\ell \times K}$ are denoted by $\hat{\mathbf{U}}_{i,:} \in \mathbb{R}^K$ for $i = 1, 2, \ldots, \ell$ are the temporal ROM coefficients and are shown in Figure A.14d. Since the temporal coefficients are polluted with noise, using a uniform sixth-order finite difference scheme as we did in Section 3.3 will be highly inaccurate. Instead, the time derivative $\dot{\hat{\mathbf{U}}}$ of the reduced states is computed via a simple factor method based on bimodal kernels of [17].

Figure A.15 shows the noisy snapshots for noise level $\zeta = 2\%$. Correspondingly, the sOpInf ROM results with $\ell = 5$, $\lambda_1 = 1$ and $\lambda_2 = 10^4$ trained on noisy (noise level $\zeta = 2\%$) snapshots are shown in Figure A.16. The numerical results illustrate that the sOpInf framework successfully reconstructs and predicts the dynamics of the inviscid Burgers' equation from noisy snapshots. The results show small oscillations near the shock in the testing regime. We suspect this behavior is due to the low number of modes ($\ell = 5$). The mean relative error is 1.1416×10^{-3} and the Pearson correlation coefficient is 0.99987 in comparison to the noiseless snapshots. The numerical results show that the sOpInf methodology is robust to noise and can be potentially extended to applications with noisy observational data.

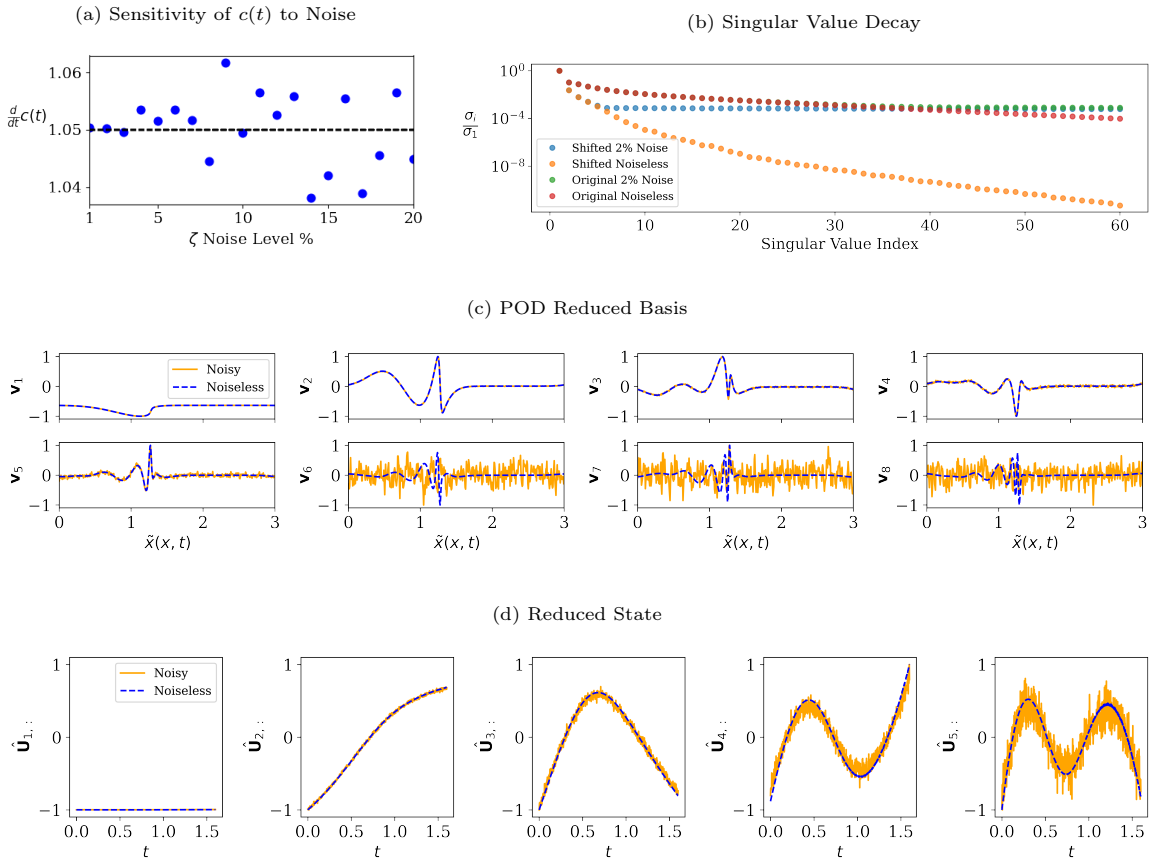

Figure A.14: Graphic (a) shows the slope $\frac{\mathrm{d}}{\mathrm{d}t}c(t)$ of the linear shift function as the noise level $\zeta = 1, 2, \ldots, 20\%$ varies. The results show that the cross-correlation extrapolation technique is robust to noise as the shift function slope remains relatively close to 1.05 as we increase the noise level. Graphic (b) shows the singular value decay of the noisy (with $\zeta = 2\%$) and noiseless Burgers' equation snapshots on the original and shifted coordinates. Graphic (c) shows the first eight POD modes normalized between $[-1, 1]$ of the noisy (noise level $\zeta = 2\%$) and noiseless inviscid Burgers' training snapshots. Of those, the first five POD modes of the noisy dataset are able to filter most of the noise, suggesting to learn a ROM with $\ell = 5$. Graphic (d) shows the first five temporal ROM coefficients (normalized to $[-1, 1]$) in the training regime for noisy and noiseless snapshots.

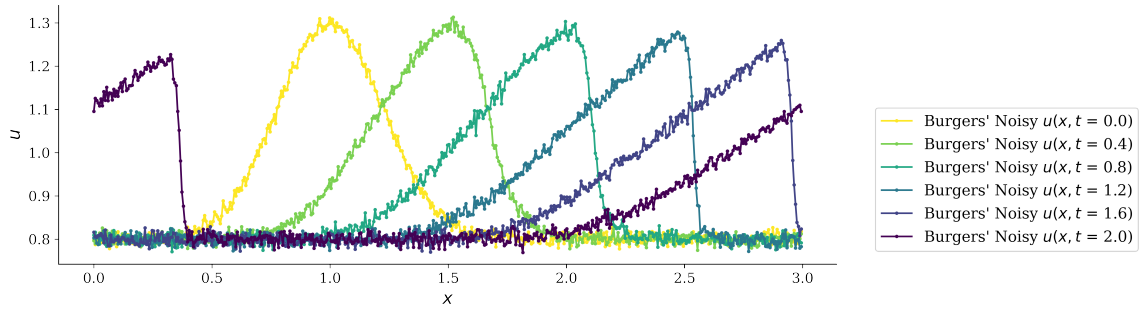

Figure A.15: The inviscid Burgers' equation simulated snapshots with added Gaussian noise (noise level $\zeta = 2\%$).

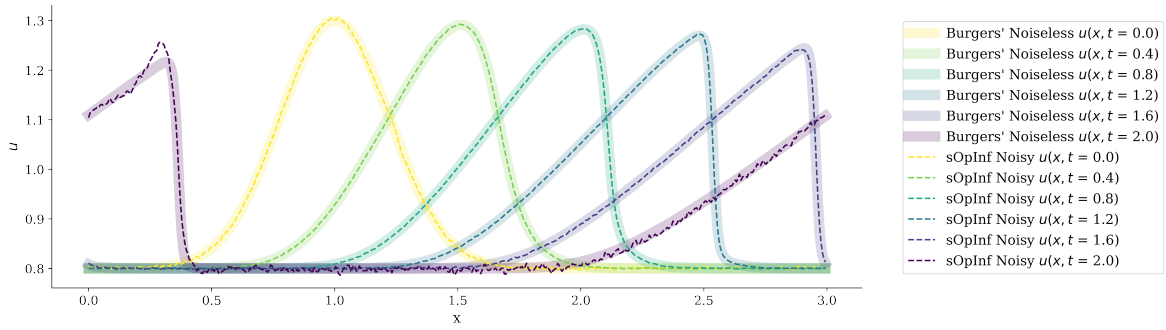

Figure A.16: Solutions from the sOpInf model of the form $\dot{\hat{\mathbf{u}}} = \hat{\mathbf{A}}\hat{\mathbf{u}} + \hat{\mathbf{H}}(\hat{\mathbf{u}} \otimes' \hat{\mathbf{u}})$ with $\ell = 5$ modes trained on noisy (noise level $\zeta = 2\%$) inviscid Burgers' snapshots. The results show good agreement with the noiseless snapshots.

References

[1] MAS model. https://www.predsci.com/MAS/. Accessed: 2022-01-25.

[2] MHD web. http://www.predsci.com/mhdweb/home.php. Accessed: 2022-02-14.

[3] NASA/NSF Community Coordinated Modeling Center. https://ccmc.gsfc.nasa.gov/. Accessed: 2022-01-25.

[4] Operator Inference package. https://github.com/Willcox-Research-Group/rom-operator-inference-Python3. Accessed: 2022-03-05.

[5] T. Amerstorfer, J. Hinterreiter, M. A. Reiss, C. Mostl, J. A. Davies, R. L. Bailey, A. J. Weiss, M. Dumbovic, M. Bauer, U. V. Amerstorfer, and R. A. Harrison. Evaluation of CME arrival prediction using ensemble modeling based on heliospheric imaging observations. *Space Weather*, 19(1):e2020SW002553, 2021.

[6] R. G. Andrews. Solar storm destroys 40 new spacex satellites in orbit. *The New York Times*, Feb 2022.

[7] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. Advances in Design and Control. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2005.

[8] S. T. Badman, S. D. Bale, J. C. M. Oliveros, O. Panasenco, M. Velli, D. Stansby, J. C. Buitrago-Casas, V. Réville, J. W. Bonnell, A. W. Case, T. D. de Wit, K. Goetz, P. R. Harvey, J. C. Kasper, K. E. Korreck, D. E. Larson, R. Livi, R. J. MacDowall, D. M. Malaspina, M. Pulupa, M. L. Stevens, and P. L. Whittlesey. Magnetic connectivity of the ecliptic plane within 0.5 au: Potential field source surface modeling of the first parker solar probe encounter. *The Astrophysical Journal Supplement Series*, 246(2):23, Feb 2020.

[9] R. L. Bailey, M. A. Reiss, C. N. Arge, C. Möstl, C. J. Henney, M. J. Owens, U. V. Amerstorfer, T. Amerstorfer, A. J. Weiss, and J. Hinterreiter. Using gradient boosting regression to improve ambient solar wind model predictions. *Space Weather*, 19(5):e02673, 2021.

[10] P. Benner, P. Goyal, B. Kramer, P. Peherstorfer, and K. Willcox. Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms. *Computer Methods in Applied Mechanics and Engineering*, 372:113433, 2020.

[11] C. D. Bussy-Virat and A. J. Ridley. Predictions of the solar wind speed by the probability distribution function model. *Space Weather*, 12(6):337–353, 2014.

[12] E. Camporeale. The challenge of machine learning in space weather: Nowcasting and forecasting. *Space Weather*, 17(8):1166–1207, 2019.

[13] R. M. Caplan, J. A. Linker, Z. Mikić, C. Downs, T. Török, and V. S. Titov. GPU acceleration of an established solar MHD code using OpenACC. *Journal of Physics: Conference Series*, 1225(1):012012, may 2019.

[14] R. M. Caplan, Z. Mikić, J. A. Linker, and R. Lionello. Advancing parabolic operators in thermodynamic MHD models: Explicit super time-stepping versus implicit schemes with Krylov solvers. *Journal of Physics: Conference Series*, 837:012016, May 2017.

[15] R. Chartrand. Numerical differentiation of noisy, nonsmooth, multidimensional data. In *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 244–248. IEEE, 2017.

[16] S. R. Cranmer, S. E. Gibson, and P. Riley. Origins of the ambient solar wind: Implications for space weather. *Space Science Reviews*, 212(3-4):1345–1384, Oct 2017.

[17] K. De Brabanter, J. De Brabanter, B. De Moor, and I. Gijbels. Derivative estimation with local polynomial fitting. *Journal of Machine Learning Research*, 14(1):281–301, 2013.

[18] G. H. Golub and C. F. Van Loan. Matrix computations. 1996. *Johns Hopkins University, Press, Baltimore, MD, USA*, pages 374–426, 1996.

[19] C. Greif and K. Urban. Decay of the Kolmogorov N-width for wave problems. *Applied Mathematics Letters*, 96:216–222, 2019.

[20] C. Gu. QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(9):1307–1320, 2011.

[21] M. Guo, S. A. McQuarrie, and K. E. Willcox. Bayesian operator inference for data-driven reduced-order modeling. *Computer Methods in Applied Mechanics and Engineering*, page 115336, 2022.

[22] J. S. Hesthaven, G. Rozza, and B. E. Stamm. *Certified reduced basis methods for parametrized partial differential equations*, volume 590. Springer, 2015.

[23] P. Holmes, J. L. Lumley, and G. Berkooz. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge Monographs on Mechanics. Cambridge University Press, 1996.

[24] A. Iollo and D. Lombardi. Advection modes by optimal mass transfer. *Physical Review E*, 89(2):022923, 2014.

[25] O. Issan and P. Riley. Theoretical refinements to the Heliospheric Upwind eXtrapolation technique and application to in-situ measurements. *Frontiers in Astronomy and Space Sciences*, 8:245, 2022.

[26] M. G. Kivelson and C. T. Russell. *Introduction to Space Physics*. Cambridge University Press, 1995.

[27] I. Knowles and R. J. Renka. Methods for numerical differentiation of noisy data. *Electron. J. Differ. Equ*, 21:235–246, 2014.

[28] B. Kramer and K. Willcox. Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition. *AIAA Journal*, 57(6):2297–2307, 2019.

[29] S. Kumar, A. Paul, and B. Vaidya. A comparison study of extrapolation models and empirical relations in forecasting solar wind. *Frontiers in Astronomy and Space Sciences*, 7:92, 2020.

[30] J. A. Linker, Z. Mikić, D. A. Biesecker, R. J. Forsyth, S. E. Gibson, A. J. Lazarus, A. R. Lecinski, P. Riley, A. Szabo, and B. J. Thompson. Magnetohydrodynamic modeling of the solar corona during whole sun month. *Journal of Geophysical Research*, 104:9809–9830, 1999.

[31] R. Lionello, C. Downs, J. A. Linker, T. Török, P. Riley, and Z. Mikić. Magnetohydrodynamic simulations of interplanetary coronal mass ejections. *The Astrophysical Journal*, 777(1):76, Oct 2013.

[32] R. Lionello, Z. Mikić, and J. A. Linker. Stability of algorithms for waves with large flows. *Journal of Computational Physics*, 152(1):346–358, 1999.

[33] H. Lu and D. Tartakovsky. Dynamic mode decomposition for construction of reduced-order models of hyperbolic problems with shocks. *Journal of Machine Learning for Modeling and Computing*, 2(1):1–29, 2021.

[34] H. Lu and D. M. Tartakovsky. Lagrangian dynamic mode decomposition for construction of reduced-order models of advection-dominated phenomena. *Journal of Computational Physics*, 407:109229, 2020.

[35] J. R. Martins and J. T. Hwang. Review and unification of methods for computing derivatives of multidisciplinary computational models. *AIAA Journal*, 51(11):2582–2599, 2013.

[36] S. A. McQuarrie, C. Huang, and K. E. Willcox. Data-driven reduced-order models via regularised operator inference for a single-injector combustion process. *Journal of the Royal Society of New Zealand*, 51(2):194–211, 2021.

[37] A. Mendible, S. L. Brunton, A. Y. Aravkin, W. Lowrie, and J. N. Kutz. Dimensionality reduction and reduced-order modeling for traveling wave physics. *Theoretical and Computational Fluid Dynamics*, 34(4):385–400, 2020.

[38] M. A. Mirhoseini and M. J. Zahr. Model reduction of convection-dominated partial differential equations via optimization-based implicit feature tracking. *arXiv preprint arXiv:2109.14694*, 2021.

[39] R. Mojgani and M. Balajewicz. Lagrangian basis method for dimensionality reduction of convection dominated nonlinear flows. *arXiv preprint arXiv:1701.04343*, 2017.

[40] D. Odstrcil. Modeling 3-D solar wind structure. *Advances in Space Research*, 32(4):497–506, Aug. 2003.

[41] M. Ohlberger and S. Rave. Reduced basis methods: Success, limitations and future challenges. *Proceedings of the Conference Algoritmy*, pages 1–12, 2016.

[42] M. Owens, M. Lang, L. Barnard, P. Riley, M. Ben-Nun, C. J. Scott, M. Lockwood, M. A. Reiss, C. N. Arge, and S. Gonzi. A computationally efficient, time-dependent model of the solar wind for use as a surrogate to three-dimensional numerical magnetohydrodynamic simulations. *Solar Physics*, 295(3):1–17, 2020.

[43] D. Papapicco, N. Demo, M. Girfoglio, G. Stabile, and G. Rozza. The neural network shifted-proper orthogonal decomposition: A machine learning approach for non-linear reduction of hyperbolic equations. *Computer Methods in Applied Mechanics and Engineering*, 392:114687, 2022.

[44] B. Peherstorfer. Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling. *SIAM Journal on Scientific Computing*, 42(5):A2803–A2836, 2020.

[45] B. Peherstorfer. Sampling low-dimensional Markovian dynamics for preasymptotically recovering reduced models from data with operator inference. *SIAM Journal on Scientific Computing*, 42(5):A3489–A3515, 2020.

[46] B. Peherstorfer and K. Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Computer Methods in Applied Mechanics and Engineering*, 306:196–215, 2016.

[47] V. J. Pizzo. A three-dimensional model of corotating streams in the solar wind: 3. magnetohydrodynamic streams. *Journal of Geophysical Research: Space Physics*, 87(A6):4374–4394, 1982.

[48] E. Qian, B. Kramer, A. Marques, and K. Willcox. Transform & learn: A data-driven approach to nonlinear model reduction. In *AIAA Aviation and Aeronautics Forum and Exposition*, Dallas, TX, June 2019.

[49] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox. Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406:132401, 2020.

[50] J. Reiss, P. Schulze, J. Sesterhenn, and V. Mehrmann. The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena. *SIAM Journal on Scientific Computing*, 40(3):A1322–A1344, 2018.

[51] M. A. Reiss, P. J. MacNeice, K. Muglach, C. N. Arge, C. Möstl, P. Riley, J. Hinterreiter, R. L. Bailey, A. J. Weiss, M. J. Owens, T. Amerstorfer, and U. Amerstorfer. Forecasting the ambient solar wind with numerical models. ii. an adaptive prediction system for specifying solar wind speed near the sun. *The Astrophysical Journal*, 891(2):165, Mar 2020.

[52] P. Riley and O. Issan. Using a Heliospheric Upwinding eXtrapolation technique to magnetically connect different regions of the heliosphere. *Frontiers in Physics*, 9:268, 2021.

[53] P. Riley, J. Linker, R. Lionello, Z. Mikic, and J. Wijaya. A rough guide to the MAS code. https://www.predsci.com/mas/doc/User-Guide.pdf, 05 2009.

[54] P. Riley, J. A. Linker, R. Lionello, and Z. Mikić. Corotating interaction regions during the recent solar minimum: The power and limitations of global MHD modeling. *Journal of Atmospheric and Solar-Terrestrial Physics*, 83:1–10, July 2012.

[55] P. Riley, J. A. Linker, and Z. Mikić. An empirically-driven global MHD model of the corona and inner heliosphere. *Journal of Geophysical Research*, 106:15889, 2001.

[56] P. Riley and R. Lionello. Mapping Solar Wind Streams from the Sun to 1 AU: A Comparison of Techniques. *Solar Physics*, 270:575–592, June 2011.

[57] P. Riley, R. Lionello, R. M. Caplan, C. Downs, J. A. Linker, S. T. Badman, and M. L. Stevens. Using Parker Solar Probe observations during the first four perihelia to constrain global magnetohydrodynamic models. *Astronomy & Astrophysics*, 650:A19, 2021.

[58] C. Rowley, I. Kevrekidis, J. Marsden, and K. Lust. Reduction and reconstruction for self-similar dynamical systems. *Nonlinearity*, 16(4):1257–1275, 2003.

[59] C. W. Rowley and J. E. Marsden. Reconstruction equations and the Karhunen-Loéve expansion for systems with symmetry. *Physica D: Nonlinear Phenomena*, 142(1):1–19, 2000.

[60] H. Sharma, Z. Wang, and B. Kramer. Hamiltonian operator inference: Physics-preserving learning of reduced-order models for Hamiltonian systems. *Physica D: Nonlinear Phenomena*, 431:133122, 2022.

[61] C. W. Snyder and M. Neugebauer. The relation of Mariner-2 plasma data to solar phenomena. In R. J. Mackin and M. Neugebauer, editors, *The Solar Wind*, page 25, Oxford, 1966. Permanon Press.

[62] R. Swischuk, B. Kramer, C. Huang, and K. Willcox. Learning physics-based reduced-order models for a single-injector combustion process. *AIAA Journal*, 58:6:2658–2672, 2020.

[63] G. Tóth, I. V. Sokolov, T. I. Gombosi, D. R. Chesney, C. R. Clauer, D. L. De Zeeuw, K. C. Hansen, K. J. Kane, W. B. Manchester, R. C. Oehmke, K. G. Powell, A. J. Ridley, I. I. Roussev, Q. F. Stout, O. Volberg, R. A. Wolf, S. Sazykin, A. Chan, B. Yu, and J. Kóta. Space weather modeling framework: A new tool for the space science community. *Journal of Geophysical Research: Space Physics*, 110(A12), 2005.

[64] W. I. T. Uy and B. Peherstorfer. Operator inference of non-Markovian terms for learning reduced models from partially observed state trajectories. *Journal of Scientific Computing*, 88(3):1–31, 2021.

[65] G. Welper. Interpolation of functions with parameter dependent jumps by transformed snapshots. *SIAM Journal on Scientific Computing*, 39(4):A1225–A1250, 2017.

[66] G. B. Whitham. *Linear and Nonlinear Waves*. John Wiley & Sons, 1999.

[67] Y. Yang, F. Shen, Z. Yang, and X. Feng. Prediction of solar wind speed at 1 AU using an artificial neural network. *Space Weather*, 16(9):1227–1244, 2018.

[68] S. Yıldız, P. Goyal, P. Benner, and B. Karasözen. Learning reduced-order dynamics for parametrized shallow water equations from data. *International Journal for Numerical Methods in Fluids*, 93(8):2803–2821, 2021.