UNIVERSIDAD CARLOS III DE MADRID
SCHOOL OF GRADUATE STUDIES
BIOENGINEERING AND AEROSPACE ENGINEERING DEPARTMENT

MASTER IN SPACE ENGINEERING

MiSE

Sparse Regression for the Breathing Mode Instability: Extracting Governing Equations from Hall Effect Thrusters Simulation Data

Master Thesis

Borja Bayón Buján

Supervisor: Mario Merino Martínez

Course 2022/2023

CONTENTS

***Abstract*—The development of efficient and reliable electric space propulsion systems relies on accurate modeling and identification of their underlying dynamics. Traditional approaches to model identification often involve intricate physical analysis or making extensive assumptions, limiting their applicability and scalability. In this thesis an algorithm for accurate modeling and identification of electric space propulsion systems is presented. The algorithm, based on sparse regression and model parsimony, allows automatic data-driven identification of models for space plasma thrusters. It incorporates statistical techniques, physical constraints, and trajectory-based information for robust system identification. The algorithm is demonstrated using PIC/fluid simulation data from a Hall Effect Thruster for several operating points. Models of varying complexity are obtained, focusing on physical explainability and coefficient variation with operating point. The resulting equations for average ion and neutral densities align well with existing models. Point-wise density models exhibit location dependency in the discharge chamber. The algorithm showcases its general applicability to other electric propulsion systems.**

"*All models are wrong, but some are useful.*" George E. P. Box

I. Introduction

Hall Effect Thrusters (HETs) have emerged as highly successful electric space propulsion systems. Typically, these systems employ an annular chamber that utilizes electron collisions to ionize an injected neutral gas, resulting in the formation of a plasma consisting of electrons and ions. This plasma is subsequently accelerated by an induced axial electric field, which is established between an internal anode (also serving as a propellant injector) and an external cathode. Despite their extensive usage in space and extensive research, certain aspects of HET physics remain incompletely understood, prompting ongoing investigations.

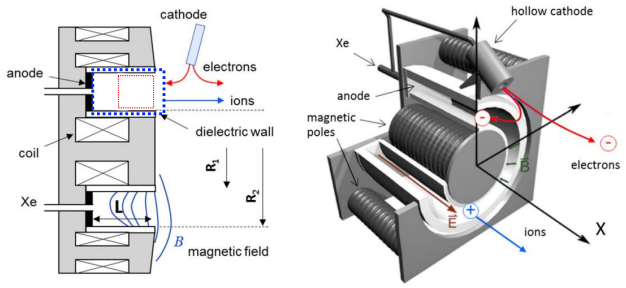

Figure 1: Lateral cut and 3D view of a Hall Effect Thruster. The areas delimited in the upper left correspond to the ones used for the global models (red) and pointwise analysis (blue). Taken from [1].

One intriguing phenomenon that merits attention is the presence of oscillations within the HET discharge. Their presence has been linked to enhanced transport and impacts the performance and efficiency of the device in a deleterious way [2]. Notably, the most prominent oscillation observed in HETs is the breathing mode. This mode entails significant low-frequency discharge current oscillations that primarily occur along the axial direction within the chamber. The physical mechanism responsible for these breathing oscillations is currently well-defined and typically described as an ionization/neutral-depletion instability.

A Breathing Mode cycle starts with neutrals filling the chamber from anodic injection and ion recombination in the walls. Ionization events occur due to neutral-electron collisions, slowly at first until either the densities or the temperature raise enough to start a cascading process which leads to a very pronounced increase in ion density. The ionization occurs in a radial front which starts in the downstream region and moves upstream inside the channel as the neutrals are depleted. The avalanche ionization stops and the instability saturates when most neutrals are depleted, and the ion density goes down until the cycle starts once more. This is made possible by the multi-scale nature of the phenomena, as neutral velocities are typically an order of magnitude below ion velocities and the injection and ionization processes also set their own scales.

In the context of electric propulsion research, the presented description elucidates a spatially-dependent complex process exhibiting similarities to a predator-prey relationship. However, the impetus for adopting simple description stems from the need to comprehend the global behavior of the instability and exploit the resultant models' simplicity. By isolating the system's dynamic characteristics, researchers are empowered to unravel critical insights into electric propulsion mechanisms.

Most models for the Breathing Mode in the literature are derived from the ion and neutral continuity equations

$$\dot{n}_i + \nabla \cdot n_i u_i = S_{ion} \tag{1a}$$
$$\dot{n}_n + \nabla \cdot n_n u_n = -S_{ion}, \tag{1b}$$

where $S_{ion} = R_{ion}(T_e) n_i n_n$ is the ionization source term proportional to the ionization rate R_{ion}, u_i and u_n are the ion and neutral velocities, assumed to be constant. Taking a volumetric average to go from the 3D equations to a generic, simple 0D global model, most of the literature obtains a model similar to

$$\dot{n}_i = S_{ion} - S_{walls} - n_i \frac{u_{iz}}{L} \tag{2a}$$
$$\dot{n}_n = -S_{ion} + S_{walls} + g_{inj} - n_n \frac{u_{nz}}{L}, \tag{2b}$$

Indeed, the very first 0D model [3] was able to replicate the non-linear self-sustained nature of the oscillations by replicating the predator-prey equations for ions and neutrals

$$\dot{n}_i = \zeta_{ion} n_n n_i - n_i \frac{u_{iz}}{L} \tag{3a}$$
$$\dot{n}_n = -\zeta_{ion} n_n n_i + n_n \frac{u_{nz}}{L}, \tag{3b}$$

However, linear stability analysis revealed that this model is metastable and cannot sustain these oscillations in a real thruster. Furthermore, the model fails to account for the constant neutral particle influx from the channel gas injector [4], but introducing this term directly yields decaying oscillations [5]. Follow-up works have tried to overcome these obstacles by introducing fluctuating neutral injection from anodic pre-ionization [6] or feedback coming from the moving ionization front [7], [8]. Others have introduced physically-motivated oscillations in the ionization rate through the electron temperature [9], ion velocity [9] or even the characteristic length [5] either by adding additional governing equations to the system or coming up with self-consistent dependencies on the immediate or delayed densities. Indeed, it seems clear that two fluctuating terms are needed to have self-sustained oscillations, but it is unclear which terms those should be. In this context the usage of data-driven techniques seems ideal, as they can help elucidate and expand existing models while minimizing the researcher's bias.

In one noticeable detail, many of the recent 0D models for the breathing mode justified the oscillation as the coupling between two zone within the thruster, but asides from the ionization zone it is not clear where this coupling occurs. Even when Hall Effect Thrusters are typically divided into distinct regions

defined by their dominant processes, these are known to overlap and clear limits cannot be defined unless arbitrary thresholds are defined. The problem of identifying subspaces of the spatiotemporal domain with distinct physics is not special to Hall Effect Thrusters, and refers to the more specific problem of Dominant Balance Analysis. However, complex geometries do not lend themselves to this analysis unless they are simplified. This is treated directly in fields and applications such as turbulence research [10], [11], geophysical flows [12], optical systems, bursting neurons and Rotating Detonation Engines [13]. These works segment the different zones with application-specific clustering algorithms based on expert knowledge, or unsupervised clustering based on post hoc interpretation. [13] uses data-driven techniques, but still depends on user-defined parameters and governing equations. A purely unsupervised alternative should be able to provide for the same results with less researcher work and bias.

In this line, previous work done by the author as part of the MiSE's Integral Project identified the Sparse Identification of Non-linear Dynamics framework [14] as a very suitable technique for interpretable system identification. Based on sparse regression on the system dynamics, SINDy returns models based on algebraic expressions from a user-provided library, where the algorithm is able to assertain which library features appear in the true dynamics, culling the rest. The aforementioned work implemented a Python library based on the basic SINDy formulation and expanded with an optimizer with oracle properties [15], automatic model selection based on parsimony principles [16], statistical resampling techniques for selection robustness [17] and a weak form [18] for noise robustness. This work has also highlighted limitations of the SINDy methodology, specifically its reliance on linear regression without accounting for temporal causality and the need for coupling the regression process among the different variables. To address these issues, this thesis develops a Integral Least Square model fine-tuning methodology, enabling direct regression on trajectory data after feature identification. This approach rectifies the temporal causality concerns and benefits from a smaller search space.

Additionally, in physical applications, incorporating underlying physical/mathematical knowledge is crucial for enhancing model performance and interpretability. Enforcing physical priors through regularization and constraints has been explored as a means to align predictions with the system's understanding. Recent advancements have been made to incorporate constraints into the SINDy algorithm, expanding its capabilities in the presence of energy-preserving quadratic non-linearities [19] or through new constraint-friendly optimizers such as the Sparse Relaxed Regularized Regression (SR3) [20] and Conditional Gradients based approaches [21]. Even newer approaches use Discrete optimization through Mixed-Integer Optimization (MIO) [22], which allows to express arbitrary linear, quadratic, and semidefinite constraints on both the coefficients and sparsity structure. However, some of these approaches are designed for high-dimensional and high-noise scenarios and sacrifice algorithmic simplicity and interpretability, thus reducing the core advantages offered by SINDy's versatile and interpretable algorithmic framework. In this work we propose a simpler alternative based on linear regression which allows to impose linear equality constraints.

The first part of this thesis explores an unsupervised perspective of obtaining 0D Breathing Mode global models for the downstream region, gradually introducing additional variables to assess their adjustment to expectations. By using the aforementioned fine-tuning techniques, the resulting models are improved for predictive purposes. In the second part of this thesis we conduct a parametric analysis to test the fidelity of the selected models, and test a novel implementation of data-driven Dominant Balance Analysis to identify dynamical regimes within the thruster. Overall, this research aims to provide a fresh perspective to the Breathing Mode and its spatial dynamics within Hall Effect Thrusters, and validate the algorithms for future use in electric propulsion research.

II. Methods

This sections intends to outline the Methods used within our implementation of the SINDy framework. A brief summary of the SINDy problem with the ALASSO optimizer, the Pareto knee selection criterion and the usage of Weak forms and data bootstrapping will be shown first, as their detailed descriptions can be found in the Integral Project, which can be found in APPENDIX 1.

The rest are methods developed within this Master thesis and which expand upon this core. Namely, introducing constraints, an Integral version of Least Squares and the basis of pointwise modeling will be covered afterwards in more detail. An explanation of how everything fits together within the algorithm is also outlined.

A. Sparse Identification of Nonlinear Dynamics (SINDy)

Most physical systems have only a few relevant terms that define their dynamics. For a dynamical system of state $\mathbf{X}(t)$ governed by a set of ordinary differential equations of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) = \dot{\mathbf{x}}(t) = f(\mathbf{x}(t)) \tag{4}$$

where we can construct a pair of matrices $\mathbf{X}$ and $\dot{\mathbf{X}}$ representing the state variables and their derivatives sampled at discretes points in time:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^T(t_1) \\ \mathbf{x}^T(t_2) \\ \vdots \\ \mathbf{x}^T(t_m) \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_k(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_k(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_k(t_m) \end{bmatrix}$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}^T(t_1) \\ \dot{\mathbf{x}}^T(t_2) \\ \vdots \\ \dot{\mathbf{x}}^T(t_m) \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \cdots & \dot{x}_k(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \cdots & \dot{x}_k(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \cdots & \dot{x}_k(t_m) \end{bmatrix}. \tag{5}$$

To determine the function $\mathbf{f}$ in a data-driven way, we can express the problem as a linear regression for each of the columns of (5), denoted from now on $\dot{X}_j$ where $j \in [1, ..., k]$. By either measuring the state derivatives directly or computing them numerically from the state measurements we can obtain said columns. Likewise we can measure or construct from state measurements a feature library $\mathbf{\Theta}$. Each of its columns represents a different feature/function $f_i(\mathbf{X})$ (a scalar function dependent on the state and is a candidate to be present in the dynamics $\mathbf{f}$), evaluated at each time-step i.e. $\mathbf{\Theta}(\mathbf{X}) = [f_1(\mathbf{X}), f_2(\mathbf{X}) \ ... \ f_n(\mathbf{X})]$. We then formulate the regression problem as in Equation (6) to solve for the coefficients β_j:

$$\dot{X}_j = \beta_j \cdot \mathbf{\Theta}(\mathbf{X}) \tag{6}$$

From the assumption that f is sparse we would expect for most of the $\beta_{j,i}$ in the true coefficients $\beta_j = [\beta_{j,0}, \beta_{j,1}, ..., \beta_{j,n}]$ to be near or equal to zero; however, this would not be the case if Ordinary Least Squares regression was to be applied to (6). Likewise, the brute-force approach of trying all possible $\mathbf{\Theta}$ and choosing the best-performing one can get intractable for high-dimensional systems and candidate libraries.

The Sparse Identification of Nonlinear Dynamics (SINDy) exploits the key idea that $\mathbf{f}$ is sparse by solving for (6) through regression but adding a regularization term to promote sparsity in the solution:

$$\beta_j = \min_{\beta_j} \left\{ \left\| \dot{X}_j - \beta_j \mathbf{\Theta}(\mathbf{X}) \right\|_2^2 + \lambda R(\beta_j) \right\} \tag{7}$$

where R(β_j) is a norm term that penalizes the presence of non-zero elements in the coefficient vector, and λ weights the regularization over the typical least squares term. It can be seen that by using $\lambda = 0$ in (7) the Ordinary Least Squares formulation is recovered.

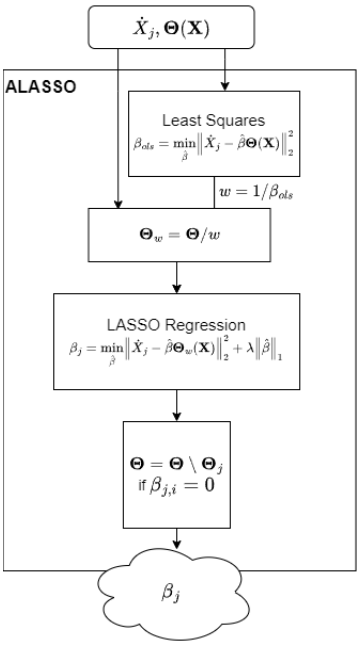

Figure 2: Schematic view of the ALASSO algorithmic implementation.

In a previous study we found the Adaptive Least Absolute Shrinkage Operator (ALASSO) method to be the most suitable for our study. As a variation of the traditional LASSO method, it uses the L1-norm but with each coefficient having a different weight in the regularization term based on their Ordinary Least Squares coefficients β_j^{ols}, such that in Equation 7:

$$R(\beta_j) = \sum_i \left\| w_i \cdot \beta_{j,i} \right\|_1 = \sum_i \left| \frac{1}{\left| \beta_{j,i}^{ols} \right|} \cdot \beta_{j,i} \right| \tag{8}$$

The algorithm can be seen in detail in Figure 2. After the correct features are selected, a final Ordinary Least Squares fit is done on the remaining features to ensure that the coefficients aren't biased. In the following sections some other options for additional fine-tuning of the coefficients will be explored.

B. Hyperparameter tuning: Pareto front Analysis

To select the model for $\dot{X}_j$ in an automatic, data-driven way we use the Pareto "knee" criteria. We plot every model obtained from running the minimization in Equation (7) with different λ values in the space of the two functions we want to minimize: model error and complexity. We select the model which minimizes a distance metric $d_{Pareto,i}$ defined for model i as:

$$d_{Pareto,i} = \sqrt{(1 - R_i^2)^2 + \left(\frac{n_{terms,i}}{n_{features}} \right)^2} \tag{9}$$

where R_i^2 stands for model coefficient of determination, $n_{terms,i}$ for the model number of non-zero terms and $n_{features}$ is the number of columns/functions in the library Θ. In previous work the Pareto knee was observed to be too strict in practice, leading to overly-sparse solutions. However, we decide to maintain this approach instead of using other options which tend to overfit the data.

C. Data bootstrapping and feature culling

Data bootstrapping and random feature culling have been proposed to address the inconsistent feature selection problems for noisy and correlated data. This approach involves repeating the model search process on multiple random subsamples of the data and dropping random features from our feature library at each search, introducing variability into the generated models. This was seen to improve the search when correlated features, few data, noise or outliers are present.

D. Weak SINDy

Weak SINDy applies integration by parts and places the derivative on analytical test functions instead of the possibly-noisy data. A weak form of (6) has been obtained from multiplying both sides by $\int_a^b \phi(u)\mathrm{d}u$, where $\phi(u)$ is the analytical test function. In our case, $\phi(u) = 1$ similar to [18], [23] so that we arrive to

$$\int_{t_0}^{t_1} \dot{X}_j dt = \int_{t_0}^{t_1} \beta_j \mathbf{\Theta}(\mathbf{X}) \mathrm{d}t \tag{10}$$

$$\Rightarrow X_j(t_1) - X_j(t_0) \approx \Delta t \, \beta_j \sum_{t=t_0+\Delta t}^{t_1} \mathbf{\Theta}(\mathbf{X}) \tag{11}$$

In previous work this was confirmed to yield good results for noisy data but depend strongly on its parameters $n_{windows}$ and n_{points} per window.

E. Physics-informed model constraints

Another way of improving the performance of the resulting equations is through the introduction of physics-based constraints, coming from conservation laws (mass, momentum, energy) or previous knowledge of the system. Within the SINDy framework there exist several ways of introducing constraints depending on the optimizer. Most are equivalent to the usage of Lagrange multipliers by including the constraint as an additional penalty term in Equation (7):

$$\beta_j = \min_{\beta_j} \left\{ \left\| \dot{X}_j - \beta_j \mathbf{\Theta}(\mathbf{X}) \right\|_2^2 + \lambda R(\beta_j) + \alpha C(\beta_j) \right\} \tag{12}$$

However, this introduces an additional hyperparameter α which must be tuned, can turn the optimization non-convex and does not guarantee that the constraints will be satisfied exactly.

Given some initial knowledge of the model form, either from prior information or from a preliminary run of the SINDy algorithm, it is possible to formulate a constrained linear regression problem which incorporates equality constraints. The linear constraints can be incorporated into a matrix C

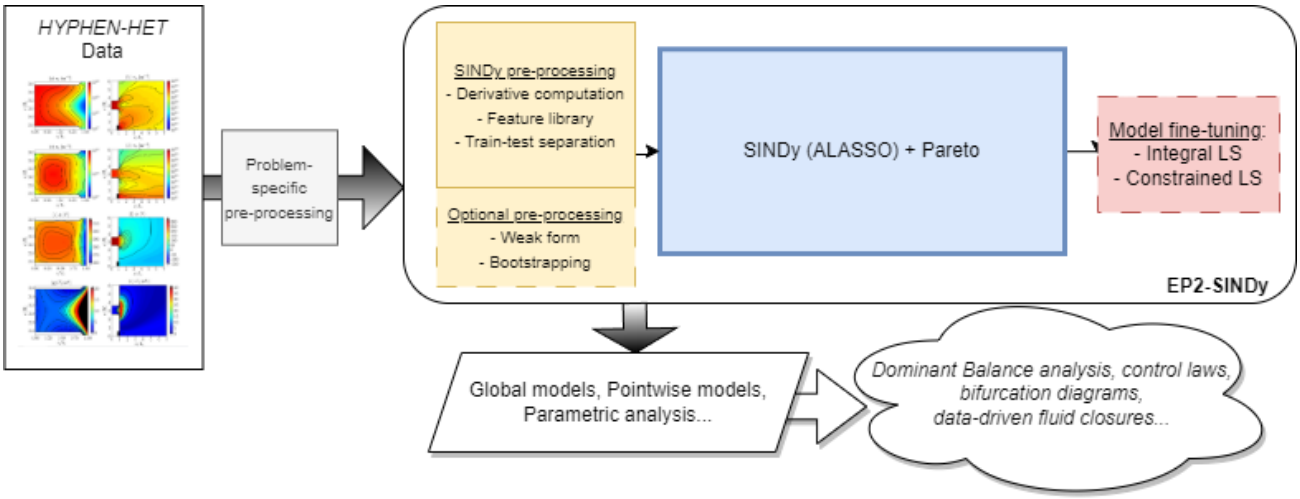

Figure 3: Summary of the algorithm's workflow, with the outputs and desired objectives outlined.

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1k} \\ c_{21} & c_{22} & \cdots & c_{1k} \\ \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nk} \end{bmatrix} \tag{13}$$

We need to couple the original regression problem to solve for the entire dynamics of $\dot{\boldsymbol{X}}$ simultaneously. To do this we take the original $\dot{\boldsymbol{X}}$ matrix and stack all its columns $\dot{X}_j$ into a single column. Similarly, we take the original Theta (now including only the features which we know will be in our model) and form an expanded Theta matrix where each of its columns is made of a repetition of the original column to match the size of the newXdot

$$\dot{\hat{X}} = \widehat{\boldsymbol{\Theta}}(X) \odot C\mathbf{b} \tag{14}$$

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \\ \vdots \\ \dot{X}_n \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{X}) & f_2(\mathbf{X}) & \cdots & f_k(\mathbf{X}) \\ f_1(\mathbf{X}) & f_2(\mathbf{X}) & \cdots & f_k(\mathbf{X}) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(\mathbf{X}) & f_2(\mathbf{X}) & \cdots & f_k(\mathbf{X}) \end{bmatrix} \odot C \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix} \tag{15}$$

where $\odot$ represent element-wise multiplication, such that for each element $\vartheta_{i,j}$ of the new $\widehat{\boldsymbol{\Theta}}$, we have $\vartheta_{i,j} = f_i(\boldsymbol{X}) \cdot c_{i,j}$. This solves the system identication problem simultaneously for all state variables through a shared coefficient vector b. The expanded coefficient matrix, can be recovered from the reduced coefficients vector b by multiplying by the corresponding column in C:

$$\beta_j = C_j \cdot b \tag{16}$$

which will exactly satisfy the equality constraints. Note that with this method we are able to impose linear relationships between the elements of β or make them equal to zero, but we cannot impose specific values of the coefficient.

F. Integral Least Squares

Another way to introduce coupling between the variable dynamics is through integration. Once the correct model form is identified through SINDy+Pareto and having the resulting coefficient values as an initial guess, we can optimize these values by minimizing the difference between the original trajectories and the model outputs. This implies including a model integration step where the model is integrated for n_{points} over $n_{windows}$. We then have the following equations

$$\beta = \min_{\beta} \left\{ \sum_{n_{windows}} \left\| \mathbf{X} - \int \boldsymbol{\Theta}(\mathbf{x}(t)) \right\|_2^2 \right\} \tag{17}$$

In this case the problem cannot be formulated as a linear regression problem, as the integration step introduces non-linearities. However, the same local minimization algorithm (Nelder-Mead) that was used in the model search step can be used, given that we start from an initial estimate β_0 which we expect lies close to the fine-tuned solution. This keeps the computation time small compared to global optimizers like Evolutionary Algorithms or Simulated Annealing. Another benefit of having an initial estimate is that we optimize a vector multiplying β_0, such that even if the problem has coefficients ranging between many orders of magnitude (as is our case), the optimization is done for a vector with values close to 1, which aids convergence.

From our experience with the Weak formulation we expect this formulation to also depend on the window number and window length. In this case, short window length could lead to overfit. On the other hand, fitting too long of a window would not only lead to increased computational cost and perhaps overfitting, but also leading to the all-zeros trivial solution as it is the one that best averages oscillating trajectories if the real model can't be found.

A sweet-spot must be found, in our case through trial and error. Similar to the weak case, a big number of integration windows could serve to regularize and minimize overfitting, but increasing this number would gradually decreasing returns while increasing computational cost significantly.

During integration the unmodelled variables are treated as forcing terms and passed to the function at every time-step from an input array.

G. Pointwise modelling

By taking advantage of the automatic nature of the SINDyEP2 model discovery workflow, we can pass the time series of the variables for every point in the thruster to obtain a 0D model in each. In this way we are able to mimic the results of [13] with fewer hyperparameters and using a simpler algorithm. However, there are some key drawbacks: the approach presented here skips the domain segmentation by unsupervised clustering step and is

unable to promote sparsity in the number of dynamic domains directly, as done with Sparse Principal Components Analysis.

However, we would expect from the continuous nature of physical phenomena that the transitions from one dynamic domain to another should be smooth, with the strictness of the Pareto selection criterion making this transition clear; expert assessment can do the rest. We also don't share the problem of redundant clusters, so by having a reduced "equation space" (feature library) we should expect few dynamic regimes. Furthermore, uncertainty estimates similar to those of the Gaussian Mixture Models seen in [13] could also be obtained from using the data bootstrapping module, but this is not explored in this work.

Another reason to use reduced libraries instead of a fully generalized one (as in the original model search) is due to the quality of the data within real systems and the computational cost of running the algorithm for large libraries. For a general pointwise search we recommend building a library with the terms that predominantly appear in the Pareto front of a preliminary model search.

In our eyes this pointwise analysis serves a dual purpose: determining the spatial extent of applicability for candidate models and helping to provide meaning to unconventional terms appearing in the equations.

H. Implementation in the algorithm

In this project we have implemented the previously explained methods within a Python framework denoted EP2-SINDy. Our implementation can be structured in four main parts:

- Data Pre-processing: the original time series data enters; the output is the target and feature arrays, which can be in their Weak and/or bootstrapped versions. The numerical derivatives are computed by the Smoothed Finite Difference method, while the weak version of Theta is computed by numerical integration by the trapezoidal rule. The feature library itself is computed as all polynomial combinations of the system variables up to degree n. To generate each bootstrap we sample the data with replacement, randomly selecting between 20 and 80 percent of the available data. The number of features dropped per bootstrap is set by a parameter n_{culled}. Both target and features can be normalized.
- Model identification: taking the target and feature arrays, it outputs a collection of candidate models obtained using the SINDy algorithm with the optimizer of choice. The desired length and number of points used in the hyperparameter sweep can also be adjusted.
- Model selection: taking the candidate models, it selects the optimal one by computing the Pareto distances and returning the model with the minimal one.
- Post-processing: plots the Pareto front in the $error$ vs n_{terms} space, as well as printing a list with the algebraic shape of the models within for further analysis.
- Extensions: once the correct features have been selected, the coefficients can be fine-tuned by running a Least Squares on the differential or weak forms of the dynamics with and without constraints, or directly on the trajectory data through the Integral Least Squares method. As an additional option, the resulting feature selection or reduced library can be passed to the Pointwise analysis tool over a selected spatial domain, or fitted to several datasets to assess model robustness and extrapolability.

All of this is implemented within a core function, while all pre-processing which depends on the specific characteristics of the system (like selecting which features to model and which to include in the feature library, filtering, removing outliers...) has to be done outside of the core function.

The entire workflow is represented in Figure 3.

III. Data generation

In this section we give a brief overview of the code used to generate the data, and describe the data characteristics.

A. HYPHEN-HET Simulation

The 2D axisymmetric hybrid PIC/fluid HET simulator named HYPHEN-HET [24] was used to generate the data for this study, which is available for public use through a Zenodo dataset [25]. The code uses a fluid description for electrons but treats ions and neutrals kinetically, using a time-marching sequential loop to couple the modules. It includes models for complex phenomena within the thruster like inter-particle collisions, plasma-wall interactions, the formation of plasma sheaths on the thruster walls and a empirical model for anomalous electron transport, linked to turbulence.

A dedicated sheath module is used to model the transition from the bulk quasineutral plasma to the non-neutral plasma near the walls using a planar, unmagnetized, collisionless, kinetic model including Secondary Electron Emission and retaining non-Maxwellian distributions. The sheath is solved with the appropiate conditions both for dielectric and metalic walls. For plasma-wall interactions, neutral quasispecular scattering and neutralization and ion recombination are also modelled.

Both elastic and inelastic (excitation, ionization) collisions are modelled, with cross-sections taken from the BIAGI database [26] for single-ion generation and the Drawin model [27] for double-ion generation. The code is also able to simulate Charge Exchange slow ions and fast neutral populations, but this is not included in the current simulations.

The SPT-100 is simulated by modeling its specific geometry, magnetic topology and cathode position. We chose to use data obtained for a previous data-driven study [28] where stable oscillations appear with a dominant breathing mode. In this thesis we will use the nominal case used in said study, operating with Xenon with a discharge voltage of $V_D = 300$ V and an anode mass flow rate of $\dot{m}_A = 5$ mg/s. Other off-nominal cases (with the turbulent transport term not fitted with experimental data) will be used to perform a parametric analysis.

The data is obtained from 41×49 points in the axial-radial (z-r) simulation domain by time-averaging every 100 simulation steps, resulting in a sampling time of 1.5 μs for a total of 12001 points. From the simulations we extract the time series for the neutral density, singly-charged density, electron temperature and singly-charged ion velocities (both axial and radial) at all points in the discharge channel of the thruster. We interpolate the ionization rate for singly-charged ions and neutrals at each point by using the same look-up table used within HYPHEN which relate the electron temperature and ionization rates. From now on we will omit specifying the singly-charged nature of the ions and their variables for brevity; we do not select data from the doubly-charged ions as their population is much fewer, and defining an ion velocity with two populations can result ambiguous. For the pointwise models we take the data straight from each point without spatial averaging. For the global models, we average spatially over an area downstream the discharge chamber.

IV. Models for the Breathing Mode

We now finally turn to the application of the algorithm for obtaining 0D models for the ion and neutral density dynamics downstream. The models are obtained by running the ALASSO optimizer with $n_{bootstraps} = 32$ on the differential form of the inputs, with progressively more complex feature libraries. We plot the Pareto front for the simplest and more complex cases to give an outlook on the process behind model selection.

A. Simple model

In this section we review the model discovery done for the simple case during the Integral Project. For a first application of the algorithm in the real scenario we tried to find a self-consistent model by using a library featuring all polynomial combinations of n_i and n_n up to degree n=2.

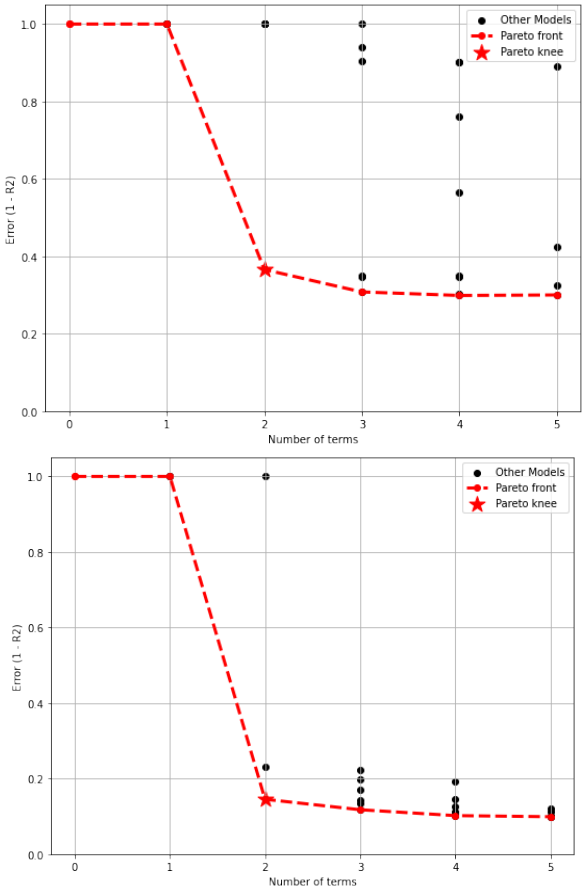

Figure 4: Pareto front of both ions (top) and neutrals (bottom).

bluePoli!40 score (R^2)	n_{terms}	d_{Pareto}	model
-8.46	1	9.46	$\dot{n}_i = 1.89 \cdot 10^{-15} n_n^2$
0.63	2	**0.49**	$\dot{n}_i = -1.45 \cdot 10^5 n_i + 4.12 \cdot 10^{-14} n_i n_n$
0.69	3	0.59	$\dot{n}_i = -2 \cdot 10^5 n_i + 4.31 \cdot 10^{-14} n_i n_n + 2.13 \cdot 10^{-14} n_i^2$
-2.56	1	3.57	$\dot{n}_n = 2.04 \cdot 10^{-14} n_i n_n$
0.85	2	**0.36**	$\dot{n}_n = 4.58 \cdot 10^4 n_n - 4.89 \cdot 10^{-14} n_i n_n$
0.88	3	0.51	$\dot{n}_n = -7 \cdot 10^4 n_n - 5.02 \cdot 10^{-14} n_i n_n - 4.16 \cdot 10^{-15} n_n^2$

Table I: Some of the models obtained along the Pareto front for the ion (top) and neutral (bottom) dynamics, along with their respective metrics. In bold, the minimal Pareto distance for each variable.

The first three models of the Pareto front can be seen in Table I. The Pareto optimal combination results in:

$$\dot{n}_i = -1.45 \cdot 10^5 n_i + 4.12 \cdot 10^{-14} n_i n_n \quad (18a)$$
$$\dot{n}_n = 4.58 \cdot 10^4 n_n - 4.89 \cdot 10^{-14} n_i n_n \quad (18b)$$

The optimal model corresponds exactly with Equation (3), featuring an ionization term $n_i n_n$ (a source for the ions, a sink for the neutrals), an ion outflow term $-n_i$ and neutral inflow n_n. Examining the resulting Pareto front, a sharp drop in error can be seen for the 2-term models. In the neutral front we see a significant number of models very similar in terms of error, which is a dangerous hint of the strongly correlated nature of our library; for our unfiltered data it would not be possible to assert if the models in the Pareto front for $n_{terms} > 2$ are physically relevant. Beyond this, we can also see how the error does not decrease much further for increasing number of terms and saturates at around 25% (10%) for the ions (neutrals). This can be related to the higher-frequency dynamics observed in the original data which cannot be modelled by our low order library. We compare the coefficients in Equation (18) with those computed from average values within the downstream area:

$$\begin{aligned} u_i/L \approx 5500m\ s^{-1} \,/\, 2.5cm = 2.20 \cdot 10^5 s^{-1} \\ u_n/L \approx 260m\ s^{-1} \,/\, 2.5cm = 1.04 \cdot 10^4 s^{-1} \\ R_{ion}(T_e) \approx 7.17 \cdot 10^{-14} m^3 s^{-1} \end{aligned} \quad (19)$$

which lie pretty close to the observed one, another confirmation of the correct physical meaning assigned to each coefficient.

B. Adding the dependence on ionization rate

To obtain a more complex model we add the ionization rate R_{ion} to our n=3 degree library. This comes motivated by the observation by some authors that oscillations were recovered for constant neutral injection models once the ionization rate dependence on a changing electron temperature was introduced through $n_i n_n R_{ion}$.

bluePoli!40 score (R^2)	n_{terms}	d_{Pareto}	model
-8.46	1	9.46	$\dot{n}_i = 1.89 \cdot 10^{-15} n_n^2$
0.92	2	**0.41**	$\dot{n}_i = -2.66 \cdot 10^5 n_i + 1.16\, n_i n_n R_{ion}$
0.95	3	0.60	$\dot{n}_i = 1.83\, n_i n_n R_{ion} - 5.45 \cdot 10^{-14} n_i n_n + -3.14 \cdot 10^{18} n_i R_{ion}$
-0.69	1	1.70	$\dot{n}_n = -3.81 \cdot 10^{-14} n_i^2$
0.88	2	**0.42**	$\dot{n}_n = 1.83 \cdot 10^{23} - 0.77\, n_i n_n R_{ion}$
0.94	3	0.60	$\dot{n}_n = 1.16 \cdot 10^{23} - 0.81\, n_i n_n R_{ion} - 1.99 \cdot 10^4 n_n$

Table II: Some of the models obtained along the Pareto front for the ion (top) and neutral (bottom) dynamics, along with their respective metrics. In bold, the minimal Pareto distance for each variable.

Looking at Table II we obtain the following Pareto-optimal model:

$$\dot{n}_i = -2.66 \cdot 10^5 n_i + 1.16\, n_i n_n R_{ion} \quad (20a)$$
$$\dot{n}_n = 1.83 \cdot 10^{23} - 0.77\, n_i n_n R_{ion} \quad (20b)$$

Indeed, the ionization rate appears as part of the ionization term S_{ion}. Interestingly, the neutral dynamics change from having a proportional neutral influx to a constant one. This is aligned with what was modelled in the literature, but the decision on the correct term for the equation was done in an unsupervised way. We can see that the ionization term coefficient is now approximately equal to 1, albeit still different for the two equations.

C. Adding the dependence on the ion velocities

In this last increase in complexity we use a library featuring all polynomial combinations of n_i, n_n, R_{ion}, $u_{i,z}$ and $u_{i,r}$ up to degree n=2, and an additional column for the term $n_i n_n R_{ion}$. We do this to avoid the computational cost and high degree of correlation that would come from doing a bootstrapped model search with a library including 35 features.

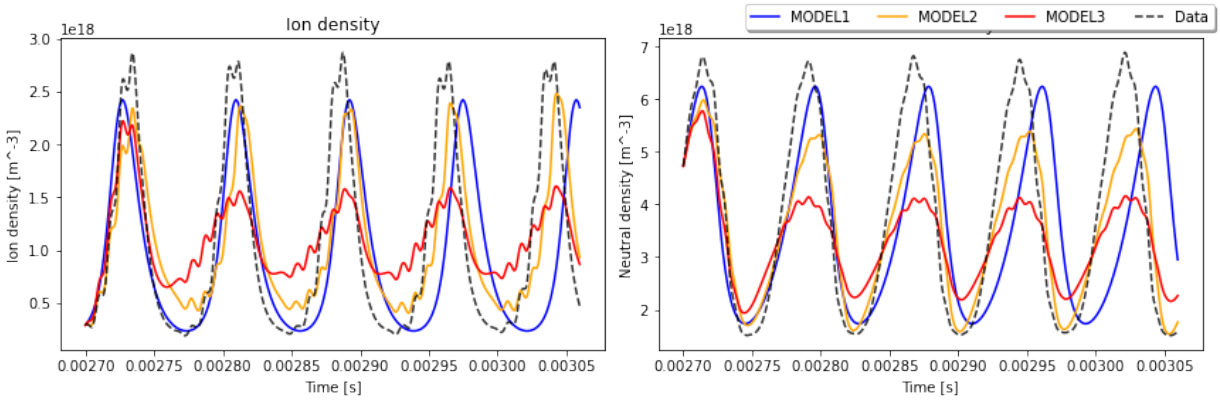

Figure 5: Integration of the three identified models, compared to the real trajectory.

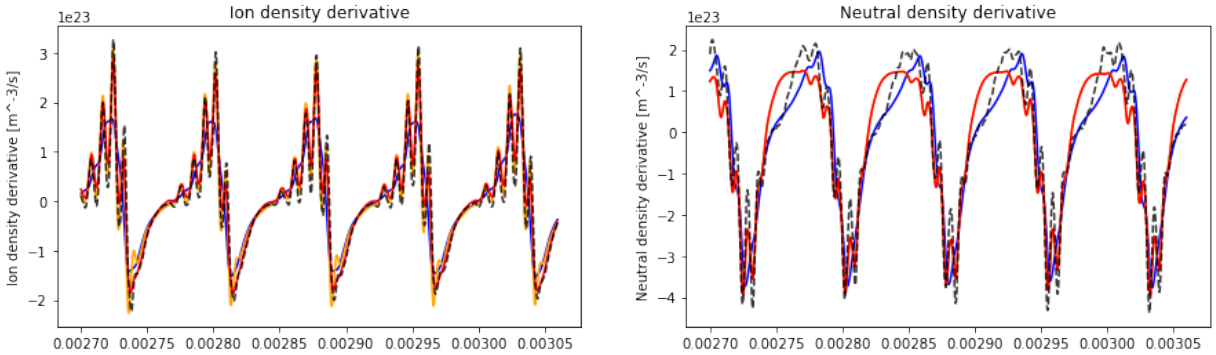

Figure 6: Comparison of the derivatives obtained from numerical differentiation of the data and evaluating the different models on the training data.

bluePoli!40 score (R^2)	n_{terms}	d_{Pareto}	model
-8.46	1	9.46	$\dot{n}_i = 1.89 \cdot 10^{-15} n_n^2$
0.95	2	**0.40**	$\dot{n}_i = -41.2\, n_i u_{i,z} + 0.82\, n_i n_n R_{ion}$
0.99	3	0.60	$\dot{n}_i = -47.7\, n_i u_{i,z} + 1.26\, n_i n_n R_{ion} - 2.09 \cdot 10^{-14} n_i n_n$
-0.69	1	1.70	$\dot{n}_n = -3.81 \cdot 10^{-14} n_i^2$
0.88	2	**0.42**	$\dot{n}_n = 1.83 \cdot 10^{23} - 0.77\, n_i n_n R_{ion}$
0.94	3	0.60	$\dot{n}_n = 2.55 \cdot 10^{23} - 0.89\, n_i n_n R_{ion} - 2.23 \cdot 10^{30} R_{ion} u_{i,z} u_{i,r}$

Table III: Some of the models obtained along the Pareto front for the ion (top) and neutral (bottom) dynamics, along with their respective metrics. In bold, the minimal Pareto distance for each variable.

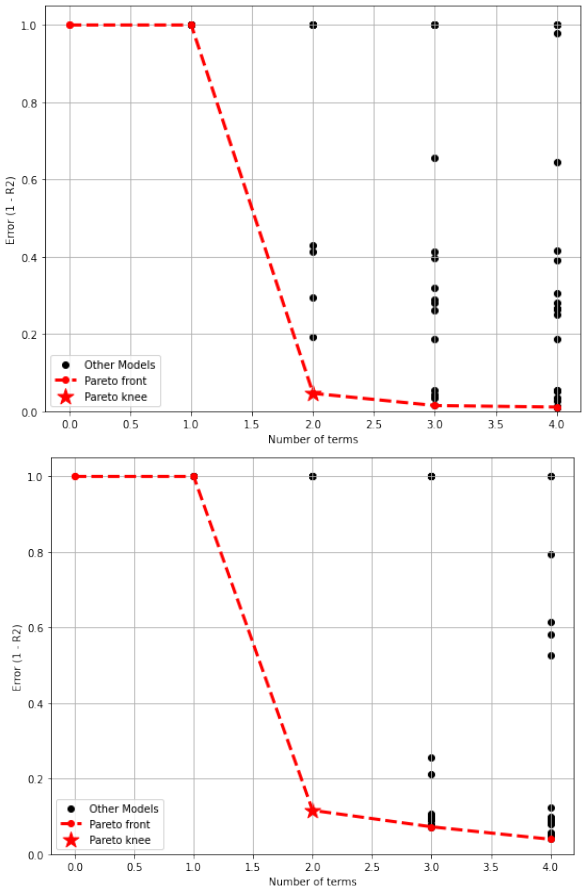

Figure 7: Pareto front of both ions (top) and neutrals (bottom).

Looking at the resulting Pareto fronts in Figure 7 we still see the pronounced drop indicative of correct identification. It can also be seen that beyond the 2-term solution there are plenty of

plausible models with the same number of terms and very close in accuracy. From Table III we have the following Pareto-optimal equations

$$\dot{n}_i = -41.2\, n_i u_{i,z} + 0.82\, n_i n_n R_{ion} \tag{21a}$$
$$\dot{n}_n = 1.83 \cdot 10^{23} - 0.77\, n_i n_n R_{ion} \tag{21b}$$

The ion axial velocity appears in the ion equation as part of the convection term. However, there is no term going with $u_{i,r}$, representing radial transport with a source or sink S_{walls}. For the neutrals, the dynamics stay the same as in the previous case. A 3-term model displaying the radial velocity is next in the Pareto front (Table III), but no apparent meaning can be given to it.

Integrating Equations (18), (20) and (21) in Figure 5 we can visually observe the performance of each model compared to the original data. The original data has the characteristic sharp increases and smoother decreases corresponding with the ions in the start, saturation and replenishment of the instability. The simple predator-prey model follows the correct shape and amplitude, but has a visibly lower frequency. The model accounting for the ionization rate has the correct frequency, but a lower amplitude for the neutrals. Surprisingly, the most complete and better-scoring of the three models performs the worst, having the right frequency but the wrong shape and amplitude.

The reason for this mismatch is two-fold, and can be seen from Figure 6: on the one hand, the constant injection model for the neutral dynamics has a better R^2 score by 3% (see Tables I and II, for instance), but the shape seems to miss the shoulder-like shape of the original dynamics. On the other hand, as was mentioned in the Introduction, the regression is done on pre-computed trajectories, very different from the one that the system follows once the model is integrated self-consistently.

V. Extensions

In the previous section we have observed the same limitations of the standard SINDy approach as was concluded during the Integral Project. In this section we aim to address these limitations by first applying the weak form of the inputs for model discovery, and then, by using the methods covered in Sections II-E and II-F. For simplicity we apply the methods to the predator-prey model.

A. Solution with Physical constraints

We use a reduced library based on the remaining functionals in Equation (18), $\boldsymbol{\Theta}(\boldsymbol{X}) = [n_i, n_n, n_i n_n]$ which will be used to build the stacked library matrix. Owing to the fact that we know the ionization term is shared and of opposite sign, while the convective term only appear in their respective equations, we build the following constraint matrix:

$$C = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix} \tag{22}$$

which constraints the ionization coefficients to be equal for both equations, but with opposite sign. Running the constrained Least Squares procedure returns the following model:

$$\dot{n}_i = -1.64 \cdot 10^5 n_i + 4.66 \cdot 10^{-14} n_i n_n \tag{23a}$$
$$\dot{n}_n = 4.39 \cdot 10^4 n_n - 4.66 \cdot 10^{-14} n_i n_n \tag{23b}$$

which results in the red trajectory seen in Figure 9. Just by adding this simple constraint we are able to correct the oscillation frequency while keeping the oscillation shape and frequency.

However, we have attempted to introduce the same constraint in Models 2 and 3, but the resulting trajectories did not change significantly. This may be a pointer that either the model or the constraint are either incorrect or too restrictive for those cases.

B. Weak form solution

In a previous study we have seen how using the weak forms improved selection in noisy cases, but the formulation had a strong dependence on the integration window hyperparameters. We are interested in see how it behaves for our system of interest, where instead of noise we have non-stochastic high-frequency dynamics superimposed to the low-frequency Breathing Mode. Setting $n_{windows} = 10000$ and changing n_{points} from 10, 50, 70 and 300 to cover within one window, respectively: a glimpse of the trajectory, half an oscillation, a full oscillation and many oscillation. The results can be seen in Figure 8

For a small number of integration points the solutions (and resulting model) tends towards the differential form one, as would be expected; for an intermediate amount around 70 points we recover a trajectory similar to the constrained and integral models, although the frequency still differs from the one of the data. For higher number of windows the frequency spectrum seems correct, but the trajectory lags behind. For even higher numbers the frequency is again incorrect. Remarkably, the correct choice of the parameter, $n_{windows} \approx 70$, leading to model:

$$\dot{n}_i = -1.51 \cdot 10^5 n_i + 4.21 \cdot 10^{-14} n_i n_n \tag{24a}$$
$$\dot{n}_n = 4.93 \cdot 10^4 n_n - 5.17 \cdot 10^{-14} n_i n_n \tag{24b}$$

also leads a trajectory very similar to the one resulting from using constraints, but without the necessity to know the real system.

C. Integral Least Squares solution

We have seen that introducing constraints has improved the solution by correcting the mismatch in frequency without altering the shape of the oscillations. We are left to wonder if using information from the trajectory to improve our initial model could yield the same result. We start from the solution given by the base algorithm to obtain the correct model shape (and reduce our feature library to $\Theta = [n_i, n_n, n_i n_n]$) and an initial estimate for the coefficients. We then run the ILS algorithm as explained in Section II-F, with $n_{windows} = 30$, $w_{size} = 100$.

Previous work from the Weak formulation has seen that the solution depends on the selection of these parameters

VI. Parametric study

After finding promising models for the nominal operating point we are interested in studying the dependence of the model coefficients with the two input parameters of the discharge: the discharge voltage V_D and the mass flow rate $\dot{m}_A$. We then pick five additional operating points where the breathing mode is apparent and perform a least squares fit to the optimal support. We only do this analysis for the model depending on the ionization rate and ion velocities as the meaning of the coefficients is the most clear.

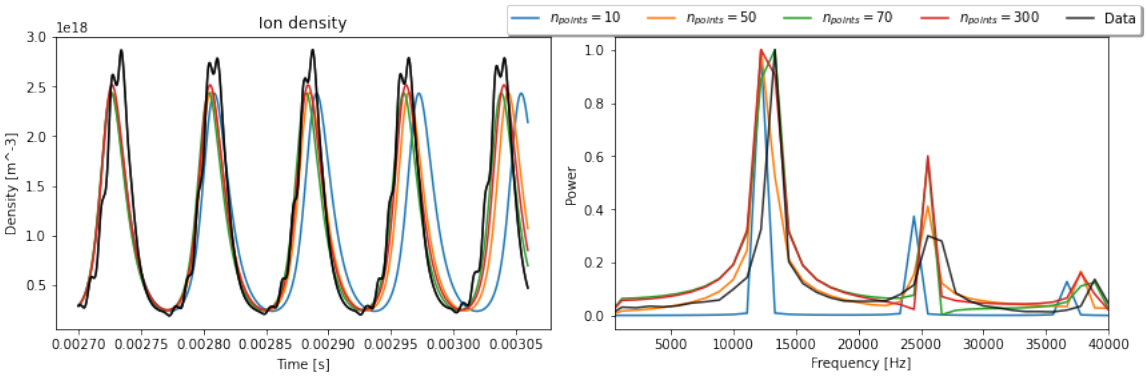

Figure 8: Comparison between true data and weak form predictions for the trajectories, phase space, derivative and Fourier spectrum. The weak forms depend on the number of points used per integration window.

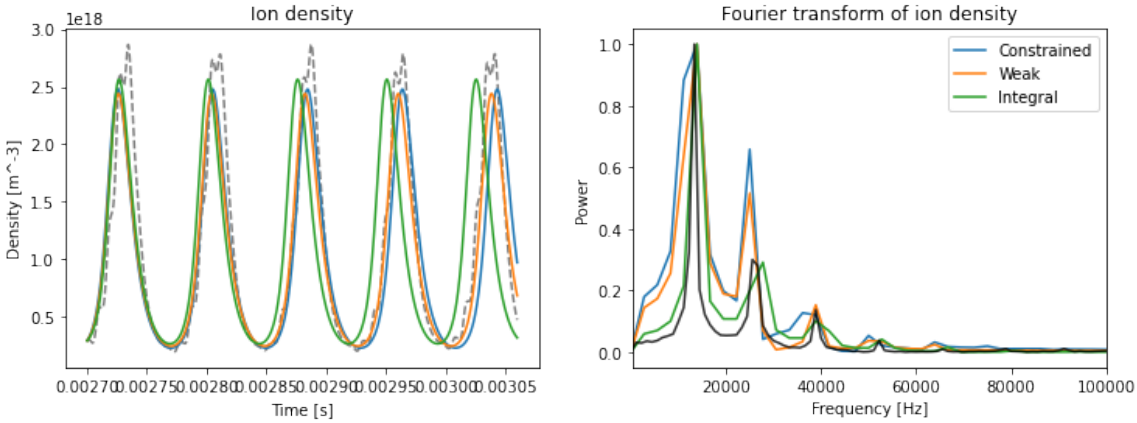

Figure 9: Comparison between true data and the constrained, weak and integral fine-tuned model predictions, with the Fourier power spectrum on the right.

After finding promising models for the nominal operating point we are interested in studying the dependence of the model coefficients with the two input parameters of the discharge: the discharge voltage V_D and the mass flow rate $\dot{m}_A$. We then pick five additional operating points where the breathing mode is apparent and apply our algorithm. We compare the results with and without the constraint on the ionization rate. We enumerate the resulting coefficients in Table IV and Table V, were taken within the equations to correspond with Equation 25.

$$\dot{n}_i = n_i u_{i,z}/L + \epsilon_{ion,n_i}\, n_i n_n R_{ion} \tag{25a}$$

$$\dot{n}_n = g_{inj} - \epsilon_{ion,n_n}\, n_i n_n R_{ion} \tag{25b}$$

where $\epsilon_{ion,n_i} = \epsilon_{ion,n_n} \equiv \epsilon_{ion}$ for the constrained case. The trends with mass flow rate for the cases at $V_D = 300V$ can be seen in Figure 10. We had too few point to set trends with the discharge voltage, so we refrain from visual analysis.

$V_D(V)$	$\dot{m}_A$ (mg/s)	L (cm)	$g_{inj}(m^{-3}/s)$	ϵ_{ion,n_i}	ϵ_{ion,n_n}
200	5	0.018	$2.36 \cdot 10^{23}$	1.06	1.06
300	2	0.020	$7.62 \cdot 10^{22}$	0.97	0.67
300	4	0.023	$1.56 \cdot 10^{23}$	0.84	0.59
300	5	0.024	$1.89 \cdot 10^{23}$	0.82	0.64
300	6	0.026	$2.23 \cdot 10^{23}$	0.77	0.61
400	5	0.026	$1.70 \cdot 10^{23}$	0.84	0.77

Table IV: Characteristic values for the terms appearing in Equation (25) for each of the operating points studied.

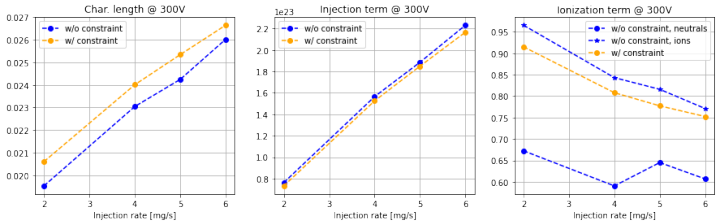

Figure 10: Trend of the characteristic length, injection and ionization terms for increasing mass flow rates, with and without constraints.

$V_D(V)$	$\dot{m}_A$ (mg/s)	L (cm)	$g_{inj}(m^{-3}/s)$	ϵ_{ion}
200	5	0.016	$2.72 \cdot 10^{23}$	1.01
300	2	0.021	$7.31 \cdot 10^{22}$	0.91
300	4	0.024	$1.53 \cdot 10^{23}$	0.81
300	5	0.025	$1.85 \cdot 10^{23}$	0.78
300	6	0.027	$2.17 \cdot 10^{23}$	0.75
400	5	0.028	$1.57 \cdot 10^{23}$	0.76

Table V: Characteristic values for the terms appearing in Equation (25) for each of the operating points studied, to make $\epsilon_{ion,n_i} = \epsilon_{ion,n_n} \equiv \epsilon_{ion}$

The linear increase of the injection term with injection rate was to be expected, but we see that both the characteristic length and ionization term depend on $\dot{m}_A$. Given that using the ionization rate directly inside the ionization term already accounts for changes in electron temperature for different mass flow rates, this is surprising. Likewise, many studies link the characteristic length in Equation (25) to the fixed thruster channel length. One reason for this change could be that it is compensating for the changing ionization term.

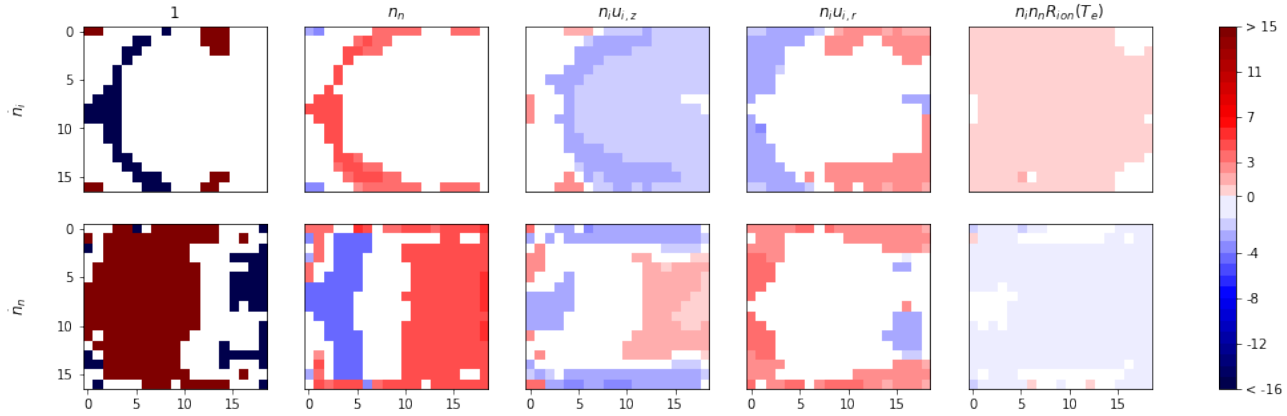

Figure 11: Pointwise distribution of coefficient values inside the discharge chamber, with the terms appearing in the ion dynamics at the top row and the neutral dynamics at the bottom. The color bar shows the magnitude of the exponent while the sign indicates whether the term is positive or negative within the equation, i.e. $val = sign(\beta_{j,i}) \cdot |log_{10}(|\beta_{j,i}|)|$.

In the case of $\epsilon_{ion,i}$, we see decreasing ionization values for increasing mass flow rate. Because of this and the fact that the value strays from 1, we would expect that within the ionization term there are other physical process being modelled; for instance, the decrease in the coefficient could be linked to a sink term proportional to $n_i n_n R_{ion}$ from ion recombination to the walls; however, this simple analysis is not sufficient to discern this, but only to obtain trends. In the case of the unconstrained ϵ_{ion}, we could say that a trend is not clear, and differences in value could be due to the numerical uncertainty in the coefficients; i.e. the term stays practically constant, and we could say that either the processes captured within the term cancel each-other out or are independent of mass flow rate.

For the constrained cases we can see how the trends remain, and the ionization term coefficient, now common to both equations, mimics the trend of ϵ_{ion,n_i}. A spatial analysis of the coefficient evolution might be able to shed more details into the meaning of the coefficients.

VII. Pointwise Models

Using the same model from the previous Section, we establish a reduced library including only the terms which could shed light into some of the unknowns seen in this thesis: $\Theta = [1, n_n, n_i u_{i,z}, n_i u_{i,r}, n_i n_n R_{ion}(T_e)]$; based on this library, we ran a model search for each point within our expanded spatial domain covering the upstream and downstream channel areas. In this way we can study the range of application of the previous section models and the spatial dependence of certain terms. The results of this analysis can be seen in Figure 11.

Some general insight should be provided first: we see that the ionization terms are present in practically all the domain and do not change their magnitude or sign significantly; likewise, the axial ion convection term is applicable to most of the channel, and only dissapears upstream near the backstreaming region, where ion velocity goes to zero and reverses direction towards the anode. This hints to a global ion behaviour, with a global model only slighty modified by extra terms in more complex areas.

This is the case near the walls, where the radial convection term finally appears unlike in the bulk plasma. Interestingly, it acts as a source term for the ions on the lateral walls, and a sink near the anode; for the neutrals it is always a source term.

For the neutrals, the neutral inflow term can be seen to go from a proportional predator-prey-like injection downstream to a constant injection near the anode; this seem to match our intuition, as further downstream there should be a strong effect from the neutrals coming from ion recombination to the walls.

Overall, trying to obtain more than general conclusions from this analysis appears harder than expected, as there is a strong spatial overlap between terms, and some visually consistent patterns do not have clear explanations. After all, some of this patterns could be related to the quality of the data, a specially important factor when no averaging or filtering of any type is performed. However, away from the walls it can be seen that the ions follow uniform dynamics while the neutral dynamic model changes the nature of its injection term over the domain.

VIII. Conclusions

Within this thesis we have reviewed and completed the implementation of data-driven framework for system identification. By expanding the SINDy algorithm with components which lead to robust model selection and model fine-tuning we have obtained a general-purpose algorithm which nonetheless shows promise in its application to electric propulsion systems.

Global models of increasing complexity were obtained which aligned pretty well with the ones found in the literature. The predator-prey model was again found to be the simplest model able to capture the Breathing Mode dynamics. More complex models, including one accounting for a time-varying ionization rate and ion velocities, were shown to be preferred even though their modelling of the neutral dynamics does not seem to match the one observed in the data. The more complex models capture well the frequency but not the amplitude of the oscillations. This highlighted that improved performance as measured in the dynamics does not imply better performance when integrating the model, and naturally required the use of advanced techniques.

In that line, a contrained and an integral variant of Least Squares on the selected features were also shown to yield improvements over the initial model. Using the weak form of the inputs could also improve the model, but precise knowledge of the right parameters is needed.

A parametric study allowed to validate the consistency of the model and expanded the meaning of its terms, as we were able to suggest that the ionization term in our advanced models may account not only for ionization, but also many other processes whose physical terms get reduced into a single, simple description from the sparsity criterion. While this effective

approximation is desired, it was seen that models with more terms along the Pareto front cannot be directly related with more detailed approximations, as previous works suggested [23].

Pointwise models pointed our attention to the spatial dependence of the neutral inflow term, while the presence of different areas heavily influenced by the wall and injector were also made clear. Outlining a precise number of clusters was not possible due to many overlaps between the dynamic subsets, but smooth transitions between regions could be observed.

Compared with experimental data, which cannot be measured in the channel of a HET and is limited in temporal and spatial resolution, HYPHEN has lent us the ability to get fully resolved spatio-temporal data. Indeed, having a algorithm capable of processing high-dimensional datasets coming from HYPHEN simulations can present for powerful modeling capabilities. Here we have limited ourselves to study a well-known phenomena in one of the most conventional thruster types, but this could be expanded to more advanced designs, as long as the simulations are a good representation of the ground truth.

Further work should seek to find data-driven models for the ionization rate/electron temperature and ion velocity, although these models do not seem to be sparse or easily expressed by polynomials. Other options for expansion could include increasing the training dataset to discover data-driven control laws, obtain bifurcation diagrams for mode transitions in HETs [29], obtain models for the oscillations in EPTs... From the algorithmic point of view, the bootstrapping use could be expanded to provide uncertainty estimates, autoencoders could be used to simultaneously learn Reduced Order Models and equations [30]. However, we must be wary of its limitations, as it acts as no substitute to the researcher but should be complemented and guided by it.

REFERENCES

[1] Boeuf, J., "Tutorial: Physics and modeling of Hall thrusters," *J. Applied Physics*, Vol. 121, No. 1, 2017, pp. 011101.

[2] Choueiri, E., "Plasma oscillations in Hall thrusters," *Physics of Plasmas*, Vol. 8, No. 4, 2001, pp. 1411–1426.

[3] Fife, J. M., *Hybrid-PIC modeling and electrostatic probe survey of Hall thrusters*, Ph.D. thesis, Massachusetts Institute of Technology, 1998.

[4] Barral, S. and Ahedo, E., "Low-frequency model of breathing oscillations in Hall discharges," *Physical Review E*, Vol. 79, 2009, pp. 046401.

[5] Dale, E. T., Jorns, B. A., and Hara, K., "Numerical investigation of the stability criteria for the breathing mode in Hall effect thrusters," *35th International Electric Propulsion Conference*, IEPC 2017-265, Atlanta, GA, 1017.

[6] Dale, E. T. and Jorns, B. A., "Two-zone Hall thruster breathing mode mechanism, Part I: Theory," *36th International Electric Propulsion Conference*, IEPC 2019-354, Vienna, Austria, 1019.

[7] Barral, S. and Peradzynski, Z., "A new breath for the breathing mode," 2009.

[8] Wang, C., Wei, L., and Yu, D., "A Basic Predator-Prey Type Model for Low Frequency Discharge Oscilations in Hall Thrusters," *Contributions to Plasma Physics*, Vol. 51, No. 10, 2011, pp. 981–988.

[9] Hara, K., Sekerak, M. J., Boyd, I. D., and Gallimore, A. D., "Perturbation analysis of ionization oscillations in Hall effect thrusters," *Physics of Plasmas*, Vol. 115, No. 20, 2014, pp. 203304.

[10] Portwood, G. D., de Bruyn Kops, S. M., Taylor, J. R., Salehipour, H., and Caulfield, C. P., "Robust identification of dynamically distinct regions in stratified turbulence," *Journal of Fluid Mechanics*, Vol. 807, 2016.

[11] Lee, J. and Zaki, T. A., "Detection algorithm for turbulent interfaces and large-scale structures in intermittent flows," *Computers Fluids*, Vol. 175, 2018, pp. 142–158.

[12] Sonnewald, M., Wunsch, C., and Heimbach, P., "Unsupervised learning reveals geography of global ocean dynamical regions," *Earth and Space Science*, Vol. 6, No. 5, 2019, pp. 784–794.

[13] Callaham, J. L., Koch, J. V., Brunton, B. W., Kutz, J. N., and Brunton, S. L., "Learning dominant physical processes with data-driven balance models," *Nature Communications*, Vol. 12, No. 1, 2021.

[14] Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, Vol. 113, No. 15, 2016, pp. 3932–3937.

[15] Zou, H., "The Adaptive Lasso and its Oracle Properties," *Journal of the American Statistical Association*, Vol. 101, No. 476, 2006, pp. 1418–1429.

[16] Cortiella, A., Park, K.-C., and Doostan, A., "Sparse identification of nonlinear dynamical systems via reweighted L1-regularized least squares," *Computer Methods in Applied Mechanics and Engineering*, Vol. 376, 2021, pp. 113620.

[17] Fasel, U., Kutz, J. N., Brunton, B. W., and Brunton, S. L., "Ensemble-sindy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and Control," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 478, No. 2260, 2022.

[18] Schaeffer, H. and McCalla, S. G., "Sparse model selection via integral terms," *Physical Review E*, Vol. 96, No. 2, 2017.

[19] Loiseau, J.-C. and Brunton, S. L., "Constrained Sparse Galerkin regression," *Journal of Fluid Mechanics*, Vol. 838, 2018, pp. 42–67.

[20] Zheng, P., Askham, T., Brunton, S. L., Kutz, J. N., and Aravkin, A. Y., "A unified framework for sparse relaxed regularized regression: SR3," *IEEE Access*, Vol. 7, 2019, pp. 1404–1423.

[21] Carderera, A., Pokutta, S., Schütte, C., and Weiser, M., "CINDy: Conditional gradient-based identification of non-linear dynamics – noise-robust recovery," Apr 2021.

[22] Bertsimas, D. and Gurnee, W., "Learning sparse nonlinear dynamics via mixed-integer optimization," *Nonlinear Dynamics*, Vol. 111, No. 7, 2023, pp. 6585–6604.

[23] Alves, E. P. and Fiuza, F., "Data-driven discovery of reduced plasma physics models from fully kinetic simulations," *Physical Review Research*, Vol. 4, No. 3, 2022.

[24] Domínguez-Vázquez, A., *Axisymmetric simulation codes for Hall effect thrusters and plasma plumes*, Ph.D. thesis, Universidad Carlos III de Madrid, Leganés, Spain, 2019.

[25] Maddaloni, D., Vázquez, A. D., Terragni, F., and Merino, M., "Data from: Data-driven analysis of oscillations in Hall thruster simulations," March 2022.

[26] Stephen Francis Biagi, "Cross sections extracted from PROGRAM MAGBOLTZ, VERSION 7.1 JUNE 2004," June 2004, [Online; accessed 5-July-2021].

[27] Mitchner, M. and Kruger Jr., C., *Partially ionized gases*, John Wiley and Sons, Hoboken, NJ, 1973.

[28] Maddaloni, D., Domínguez-Vázquez, A., Terragni, F., and Merino, M., "Data-driven analysis of oscillations in Hall thruster simulations," *Plasma Sources Science and Technology*, Vol. 31, No. 4, apr 2022, pp. 045026.

[29] Sekerak, L., Longmier, B., Gallimore, A., Huang, W., Kamhawi, H., Hofer, R., Jorns, B., and Polk, J., "Mode Transitions in Magnetically Shielded Hall Effect Thrusters," *50th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, 2014, Cleveland, Ohio.

[30] Champion, K., Lusch, B., Kutz, J. N., and Brunton, S. L., "Data-driven discovery of coordinates and governing equations," *Proceedings of the National Academy of Sciences*, Vol. 116, No. 45, 2019, pp. 22445–22451.

APPENDIX

UNIVERSIDAD CARLOS III DE MADRID
SCHOOL OF GRADUATE STUDIES
BIOENGINEERING AND AEROSPACE ENGINEERING DEPARTMENT

MASTER IN SPACE ENGINEERING

MiSE

Preliminary Study of a Data-driven System Identification Algorithm in Electric Propulsion

Integral Project

Borja Bayón Buján

Supervisor: Mario Merino Martínez

Course 2022/2023

CONTENTS

***Abstract*—The development of efficient and reliable electric space propulsion systems relies on accurate modeling and identification of their underlying dynamics. Traditional approaches to model identification often involve intricate physical analysis or making extensive assumptions, limiting their applicability and scalability. This project presents and benchmarks an algorithm based on sparse regression and model parsimony for the automatic data-driven identification of models for space plasma thrusters. Several alternatives of the algorithm are outlined and tested against a benchmark of four synthetic dynamical systems and real data from a Hall Effect Thruster simulation. The results obtained demonstrate the algorithm's ability to identify simple models from complex datasets displaying relevant features and relationships, as well as its limitations.**

I. Introduction

Hall Effect Thrusters (HETs) have emerged as highly successful electric space propulsion systems. Typically, these systems employ an annular chamber that utilizes electron collisions to ionize an injected neutral gas, resulting in the formation of a plasma consisting of electrons and ions. This plasma is subsequently accelerated by an induced axial electric field, which is established between an internal anode (also serving as a propellant injector) and an external cathode. Despite their extensive usage in space and extensive research, certain aspects of HET physics remain incompletely understood, prompting ongoing investigations.

One intriguing phenomenon that merits attention is the presence of oscillations within the HET discharge. Their presence has been linked to enhanced transport and impacts the performance and efficiency of the device in a deleterious way [1]. Notably, the most prominent oscillation observed in HETs is the breathing mode. This mode entails significant low-frequency discharge current oscillations that primarily occur along the axial direction within the chamber. The physical mechanism responsible for these breathing oscillations is currently well-defined and typically described as an ionization/neutral-depletion instability resembling a predator-prey relationship [2], [3], [4]. However, a consistent model that comprehensively explains the physics of this instability, including its onset criteria and growth rate, is still missing. In this context, machine learning and data-driven techniques offer the potential to uncover hidden insights from high-dimensional data, enabling the discovery of patterns without the limitations imposed by researcher bias.

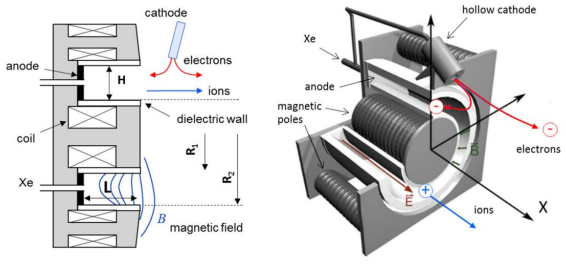

Figure 1: Lateral cut and 3D view of a Hall Effect Thruster. Taken from [5].

Several such techniques have already been employed for modeling and analysis in Hall Effect Thrusters. Examples include the use of Dynamic Mode Decomposition [6] to isolate the Breathing Mode from other dynamics, Neural Networks [7] for obtaining scaling laws, Gaussian Process regression [8] to model anomalous transport, and Stochastic system identification [9] to model discharge oscillations. However, the success of these models has been limited thus far due to lacking interpretability, requiring excessive amounts of data and/or computations or requiring a-priori model specifications which either constrain the search or lead to overfitting to the training dataset, limiting physical insights. Symbolic regression [10] has also been employed to obtain algebraic equations for the anomalous transport term while penalizing complex expressions to prevent overfitting, with much better results. In the broader context of plasma physics, sparse regression and its algorithmic implementation, the Sparse Identification of Non-linear Dynamics (SINDy) framework [11], have proven successful in deriving algebraic equations that capture system dynamics in low-pressure discharges [12], fusion [13], and various theoretical settings [14], [15] but have not yet been tested for modeling dynamics of space propulsion plasmas. The SINDy framework offers notable advantages over Symbolic regression, including reduced computational costs, easier implementation, and an abundance of variants that expand and strengthen its applications, such as weak formulations [16], [17], [18], different optimizers [19], [20] or by use of statistical techniques [21].

Beyond avoiding researcher bias, unsupervised (data-driven) system identification offers another significant advantage—it enables exploration of otherwise unapproachable model searches in high-dimensional parametric spaces. In such spaces, the parameters can encompass variables that define operating points, noise characteristics, or even spatial dependencies of the dynamics. However, a major challenge arises from the heavy reliance of these techniques on external parameters known as hyperparameters, which are not determined by the algorithm itself. Typically, these hyperparameters are manually fine-tuned through a trial-and-error process, involving the execution of the algorithm with various combinations and the selection of the best-performing combination based on error metrics. This may not be feasible for the aforementioned cases, or when there is limited understanding about system under study.

To overcome these limitations, advanced methods for hyperparameter tuning have been utilized with SINDy, including the utilization of Information Theory metrics (such as AIC and BIC) [22] and Pareto Analysis [20]. These methods aim to strike a balance between model complexity and accuracy and have demonstrated superior performance compared to alternatives like k-fold Cross-Validation in terms of both results and computational efficiency. By leveraging these techniques, it becomes possible to enhance the effectiveness of unsupervised system identification, enabling the discovery of optimal hyperparameter configurations that yield accurate and robust models in high-dimensional parametric spaces.

The main objective of this project is to evaluate some of the most promising methods employed within SINDy and assess their potential applicability to electric propulsion systems. Specifically, we wish to compare the performance of two different optimizers (STLSQ and ALASSO), two formulations (Differential and Weak), the use of statistical techniques and automatic hyperparameter tuning to improve system identification robustness. We first implemented the algorithms within a general framework, denoted EP2-SINDy and written in Python. In the second part of the project we benchmarked the algorithms by measuring performance metrics on data coming from four synthetic data-sets and from a Hall Effect Thruster simulation exhibiting Breathing Mode oscillations. The conclusions obtained from this project set the ground for further

expansions and applications as part of the author's Final Master's Thesis.

II. Methods

A. Sparse Identification of Nonlinear Dynamics (SINDy)

Most physical systems have only a few relevant terms that define their dynamics. For a dynamical system of state $\mathbf{X}(t)$ governed by a set of ordinary differential equations of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) = \dot{\mathbf{x}}(t) = f(\mathbf{x}(t)) \tag{1}$$

where we can construct a pair of matrices X and $\dot{\mathrm{X}}$ representing the state variables and their derivatives sampled at discretes points in time:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^T(t_1) \\ \mathbf{x}^T(t_2) \\ \vdots \\ \mathbf{x}^T(t_m) \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_n(t_m) \end{bmatrix}$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}^T(t_1) \\ \dot{\mathbf{x}}^T(t_2) \\ \vdots \\ \dot{\mathbf{x}}^T(t_m) \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \cdots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \cdots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \cdots & \dot{x}_n(t_m) \end{bmatrix}. \tag{2}$$

To determine the function $\mathbf{f}$ in a data-driven way, we can express the problem as a linear regression for each of the columns of matrix 2, denoted from now on $\dot{X}_j$. By either measuring the state derivatives directly or computing them numerically from the state measurements we can obtain said columns. Likewise we can measure or construct from state measurements a feature library $\mathbf{\Theta}$, where each of its columns represents a different feature/function of the state $\mathbf{f}_i(\mathbf{X})$ candidate to be present in the dynamics f, evaluated at each time-step i.e. $\mathbf{\Theta}(\mathbf{X}) = [\mathbf{f}_1(\mathbf{X}), \mathbf{f}_2(\mathbf{X}) \ ... \ \mathbf{f}_k(\mathbf{X})]$. We then formulate the problem as in Equation (3), where we have to solve for the coefficients β_j:

$$\dot{X}_j = \beta_j \cdot \mathbf{\Theta}(\mathbf{X}) \tag{3}$$

From the assumption that f is sparse we would expect for most of the $\beta_{j,i}$ in the true coefficients $\beta_j = [\beta_{j,0}, \beta_{j,1}, ..., \beta_{j,k}]$ to be near or equal to zero; however, this would not be the case if Ordinary Least Squares regression was to be applied to Equation (3). Likewise, the brute-force approach of trying all possible $\mathbf{\Theta}$ and choosing the best-performing one can get intractable for high-dimensional systems and candidate libraries.

The Sparse Identification of Nonlinear Dynamics (SINDy) exploits the key idea that $\mathbf{f}$ is sparse by solving for (3) through linear regression but adding a regularization term to promote sparsity in the solution:

$$\beta_j = \min_{\hat{\beta}} \left\| \dot{X}_j - \hat{\beta}\mathbf{\Theta}(\mathbf{X}) \right\|_2^2 + \lambda R(\hat{\beta}) \tag{4}$$

where R(β) is a norm term that penalizes the presence of non-zero terms in the coefficient vector, and λ weights the regularization over the typical least squares term. It can be seen that by using $\lambda = 0$ in (4) the Ordinary Least Squares formulation is recovered. There are several common options for R(β). The ideal R(β) is the L0 norm, because it penalizes the amount of model terms without penalizing their magnitude; however, minimizing on this norm leads to non-convex optimization problems. The L1 and L2 norms act as proxies of the first, penalizing the absolute value and the square root of the coefficients, respectively. Depending on the form of R(β), whether the optimization is iterative or done in a single step and other factors, many optimizer variants can be defined varying in their definition of (4). In this work we focus and compare two of the most successful optimizers: STLSQ and ALASSO, which are explained in detail at the end of this section.

Assuming a proper selection of R(β) and given the correct sparsity setting, SINDy is able to find the features relevant to the dynamics and set the coefficients of the rest equal to zero. This approach has several advantages compared to other system identification methods:

- The optimization problem is formulated in a linear way, even if $\mathbf{\Theta}(\mathbf{X})$ can contain arbitrarily complex non-linear functions.
- In theory, an arbitrary number of candidate functions can be included in $\mathbf{\Theta}(\mathbf{X})$ without hampering finding the true model.
- The optimization problem is uncoupled among the state variables of the dynamical system, as it only depends on their time-series.
- By not requiring a priori specification of a model asides from the selection of library candidate terms, sparse regression isn't as heavily affected by human bias, or unknown gaps in domain knowledge. The algorithm so far is purely data-driven, as its minimization is based solely on the data.

The second point is only true for entirely orthogonal features, due to increasing ill-possened of the problem owing to feature correlation as the number of features increases. Furthermore, the formulation is based on several large assumptions: namely, that the true first-order dynamics of each state variable are sparse, that the true functionals present in the real dynamics are represented in $\mathbf{\Theta}(\mathbf{X})$ and that Equation 3 holds (which is not the case for noisy data). Most of these assumptions can be relaxed by either looking for approximate models or extending the SINDy formulation with additional expansions of the method. Overall, the ease of adaptation and implementation is another big advantage of using SINDy.

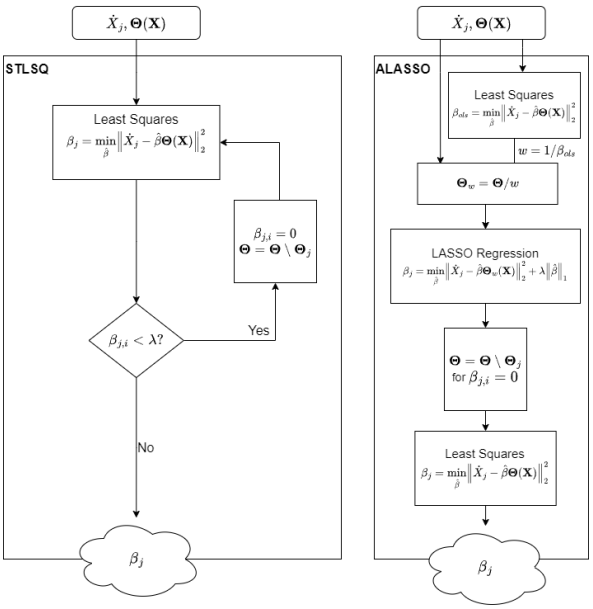

Figure 2: Schematic view of the STLSQ and ALASSO algorithmic implementation.

1) STLSQ: The Sequentially Thresholded Least Squares (STLSQ) algorithm was presented in the original SINDy paper

[11], and has been shown to have guaranteed local convergence [23], often outperforming convex variants such as the LASSO. The algorithm works as shown in Figure 2: by iteratively doing Ordinary Least Squares regression with $R(\beta) = 0$, and then culling from the initial library the features whose coefficient is below a set threshold. At each iteration the Least Squares procedure is repeated on the remaining features, until all the non-zero coefficients are above the given threshold. The thresholding step has been shown to be an effective proxy of L0-norm regularization [20], but requires that either all the coefficients are of the same order of magnitude or that the target and features are normalized before inputting them into the algorithm.

Another complication of the STLSQ algorithm comes from a decrease in accuracy of model selection for increasing noise levels. Recent variations of the algorithm add L2-norm regularization to the regression problem (STRidge [24]) but this adds one additional parameter related to the regularization strength and biases the coefficients towards small values. Moreover, the same paper that gives it local convergence guarantees warns about the danger of inconsistent feature selection due to early culling of important features [23].

2) ALASSO: One method which drops the thresholding step while acting as an unbiased and convex proxy of the L0-norm is the Adaptive Least Absolute Shrinkage Operator (ALASSO) method. As a variation of the traditional LASSO method, it uses the L1-norm but with each coefficient having a different weight in the regularization term based on their Ordinary Least Squares coefficients β^{ols}, such that Equation 4 becomes:

$$R(\beta_j) = \sum_i \|w_i \cdot \beta_{j,i}\|_1 = \sum_i \left| \frac{1}{|\beta^{ols}_{j,i}|} \cdot \beta_{j,i} \right| \tag{5}$$

This has been shown to have oracle properties, in the sense that it performs as well as if the true underlying model were given in advance [25]. By using a weighted L1 norm it is able to place a stronger penalty on the coefficients $\beta_{j,i}$ that are anticipated to be small (or zero) on top of avoiding penalizing big coefficients values. In a more formal sense, the ALASSO gives a consistent estimator β in terms of variable selection and parameter estimation in the limit of sample size $N \to \infty$, meaning that it gives an unbiased estimate corresponding to the true model. Furthermore, because of the use of the L1 norm instead of an L2 norm, the algorithm leads the coefficients to zero and no thresholding step is needed. Our implementation of the ALASSO algorithm can be seen in Figure 2, where applying the weights to the feature library is equivalent to doing so to the coefficients, and allows to re-use the Ridge regression implementation of the Sklearn Python library. Unlike the STLSQ algorithm in which it is done automatically, a final Ordinary Least Squares fit is done on the remaining features to ensure that the coefficients aren't biased.

B. Hyperparameter tuning: Pareto front Analysis

One of the limitations of SINDy is the need to find the correct sparsity level, set by the hyperparameters α in ALASSO and λ in STLSQ as seen in the previous section. There is no known method for the optimal choice of such regularization parameters [26], but there are several methods [27] to approximate it:

- Supervised methods (requiring knowledge of the problem or of the noise variance): heuristic (rule of thumb) methods, Morozov's discrepancy principle.
- Unsupervised (data-driven) methods: (Generalized) Cross-validation, Information criteria like the AIC/BIC, L-curve/Pareto front analysis.

Among the second kind, using GCV has been observed to easily lead to overfitting while the definition of the AIC/BIC can be somewhat arbitrary. For this reason, in this work we choose to explore Pareto front analysis.

By considering a wide range of hyperparameter values, from zero to infinity, various sets of solutions to the optimization problem can be obtained. These solutions span from the inaccurate all-zero solutions for $\lambda = \infty$ to the better-scoring but non-sparse least square solutions for $\lambda = 0$. The key is to find a sweet spot where the model strikes a balance between overfitting and underfitting the data. This principle is known as the Pareto "knee" criteria. Plotting the error versus complexity for each model allows us to identify the best models at different levels of complexity, forming an L-shaped curve within the solution space known as the Pareto front. The knee of this curve likely represents an appropriate trade-off point, where both metrics are minimized effectively.

In our case we define the complexity exactly as the L0-norm of the model, normalized by the total number of features in the library. We define the error as $1–R^2$, where R^2 is the coefficient of determination on the whole dataset. For ideal cases a clear inflection region in terms of error would appear as model complexity is increased representing the addition of all terms necessary to minimally represent the dynamics; this is precisely the definition of the knee-point and Pareto-optimal model. However, real system exhibiting noise, secular terms (as will be seen in the Underdamped Oscillator model) or represented in a non-sparse basis may not display this feature. To locate the knee-point we instead use the geometrical property that the knee is the point closer to the ideal solution, in this case one with zero complexity and zero error. To find the knee we define an Euclidean distance in the normalized space:

$$d_{Pareto,i} = \sqrt{(1 - R_i^2)^2 + \left(\frac{n_{terms,i}}{n_{features}} \right)^2} \tag{6}$$

such that the knee corresponds to the model which minimizes this distance. We take this method from Multi-objective Optimization's Weighted Metric Method [28], although in our case we use this method to choose one optimal solution from the already-existing solution space, instead of iteratively locating all Pareto front solutions.

C. Data pre-processing

1) Train-test set separation: Train-test data separation is crucial in Machine Learning as it provides us with an estimate of how well the model will perform on unseen data, on top of avoiding model overfits to the training data. In our case, to obtain the test set we sample 25% of the data without replacement. We use the train set to find and fit the models, while the test set is only used to evaluate the scores for the Pareto front selection.

2) Computing the derivatives: If the data are smooth in the sense of lacking abrupt or irregular fluctuations, then Finite Difference methods give accurate derivative approximations. When the data are noisy, they give derivative estimates with more noise than the original data. To compute the numerical derivatives we use the Smoothed Finite Difference method, which first uses a Savitzky-Golay filter to smooth the data, then takes Finite Differences.

3) Weak SINDy: The subtraction operation in numerical differentiation amplifies the noise present in the original time-series. Applying filtering can smooth out fast dynamics and introduce artifacts, hampering the feature selection process. For this reason some authors have proposed the use of Weak versions of Equation 3 [17], [18], obtained from multiplying

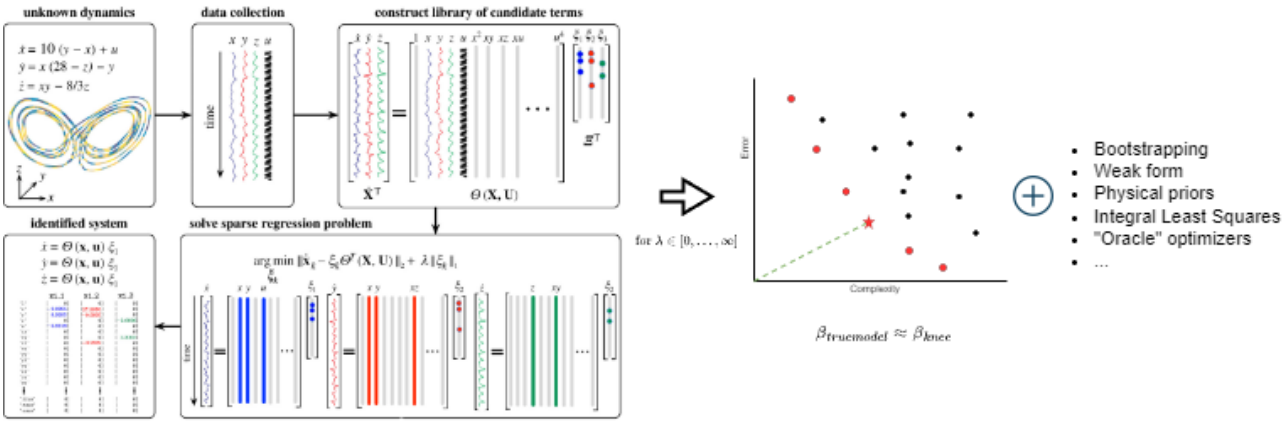

Figure 3: Summary of the algorithm: first SINDy is applied to obtain the candidate models for the dynamics of every modelled variable by sweeping a range of the algorithm hyperparameters. Then the candidates are plotted in a Pareto front and the optimal one is chosen based on the "knee" criterion. Expansions of the framework are enumerated on the left; some are present in this study while others are planned to be implemented in the future. Modified from [XXX].

both sides by $\int_a^b \phi(u)\mathrm{d}u$, where $\phi(u)$ is an analytical test function, such that

$$\int_{t_0}^{t_1} \phi(t)\dot{X}_j(t)\mathrm{d}t = \int_{t_0}^{t_1} \beta_j \phi(t)\boldsymbol{\Theta}(\mathbf{X}(t))\mathrm{d}t \tag{7}$$

By using integration by parts and a $\phi(u)$ with an analytically defined derivative we can completely remove the derivative of the data from the left hand side, i.e.

$$\int_{t_0}^{t_1} \phi(t)\dot{X}_j(t)\mathrm{d}t = \\ = \phi(t_1)X_j(t_1) - \phi(t_0)X_j(t_0) - \int_{t_0}^{t_1} \dot{\phi}(t)X_j(t)\mathrm{d}t \tag{8}$$

In this work we take $\phi(u) = 1$ similar to [16], [15], and arrive to the integral equation of the dynamics by substituting in 8, which can be approximated numerically

$$\int_{t_0}^{t_1} \dot{X}_j dt = \int_{t_0}^{t_1} \beta_j \boldsymbol{\Theta}(\mathbf{X})\mathrm{d}t \tag{9}$$

$$\Rightarrow X_j(t_1) - X_j(t_0) \approx \Delta t\, \beta_j \sum_{t=t_0+\Delta t}^{t_1} \boldsymbol{\Theta}(\mathbf{X}) \tag{10}$$

Note that the coefficients β_j obtained through this method are the same as for the differential formulation, but the minimization of equation (4) is done directly on the trajectory data, i.e. $\hat{X}_j = X_j(t_1) - X_j(t_0)$ and $\hat{\boldsymbol{\Theta}}(X) = \int_{t_0}^{t_1} \beta_j \boldsymbol{\Theta}(\mathbf{X})\mathrm{d}t$ computed by numerically integrating the function library evaluated on the data (in our implementation, using the trapezoidal rule). In the limit of $t_1 = t_0 + 1 * \Delta t$ we would expect this formulation to be identical to the differential one, which indeed is the case. This has served as a test of the algorithm implementation.

We repeat the procedure for several (t_0, t_1) to obtain many pairs of $(\hat{X}_j, \hat{\boldsymbol{\Theta}}(X))$ to stack for the regression. Within our study we will the proposed improvement in model accuracy and the effect in the choice of the number of windows and the number of points per window.

4) Building the feature library: While other options are possible, to build the feature library used in the Right Hand Side of the equations we build libraries composed of all polynomial combinations of the system variables up to degree n, i.e. for three variables

$$\boldsymbol{\Theta}(\mathbf{X}) = \left[1, x, y, z, xy, xz..., x^i y^j z^k\right] \tag{11}$$

such that i+j+k = n. We usually select a degree as low or close to the expected dynamics as possible, as high degree polynomials can present heavily statistical correlation and complicate feature selection.

5) Data bootstrapping and feature culling: The convex formulation of ALASSO and convergence guarantees of STLSQ ensure that each optimization run produces a unique solution. However, consistent feature of these algorithms has already been questioned, specially when finite sample sizes and highly correlated features are used to construct the library. In such cases, the algorithm may yield alternating solutions (selecting one of each pair of the correlated functions) when the data or the features present in the library are modified slightly, potentially overlooking a more accurate model with the same number of terms.

Data bootstrapping and random feature culling have been proposed to address the variability in solution outcomes. This approach involves repeating the optimization on multiple random subsamples of the data and examining the variations in the resulting models. This allows to assess the stability of the algorithm on the given problem and the uncertainty in each term, and has been shown to be equivalent in performance to Bayesian techniques [21], [29] offering important improvements over the base optimizers where it is applied. As for our current implementation we use bootstrapping solely for model identification, although future expansions could implement it for uncertainty quantification through model ensembles. To generate each bootstrap we sample the data with replacement, randomly selecting between 20% and 80% of the available data. Additionally, we drop a random feature from our feature library, introducing variability into the bootstrapped models.

D. The algorithm

In this project we have implemented the previously explained methods within a Python framework denoted EP2-SINDy. Our implementation can be structured in four main parts:

- Data Pre-processing: the original time series data enters; the output is the target and feature arrays, which can be in their Weak and/or bootstrapped versions. During pre-processing one can choose to normalize both target and features, which passes the inputs normalized by their column-wise L2-norm into the optimizer, but still does the

last Ordinary Least Squares fit on the unnormalized inputs. The bootstrapping step is implemented through a function which returns a bootstrap of the data with n_{culled} columns of $\mathbf{\Theta}(\mathbf{X})$ randomly dropped out. The Weak version of inputs is also obtained through a function which returns the inputs with columns of length $n_{windows}$ instead of $n_{timesteps}$.

- Model identification: taking the target and feature arrays, it outputs a collection of candidate models obtained using the SINDy algorithm with the optimizer of choice. The desired length and number of points used in the hyperparameter sweep can also be adjusted.
- Model selection: taking the candidate models, it selects the optimal one by computing the Pareto distances and returning the model with the minimal one.
- Post-processing: plots the Pareto front in the $error$ vs n_{terms} space, as well as printing a list with the algebraic shape of the models within for further analysis.

All of this is implemented within a core function, while all pre-processing which depends on the specific characteristics of the system (like selecting which features to model and which to include in the feature library, filtering, removing outliers...) has to be done outside of the core function.

Notice that up to now we have used the uncoupled nature of the problem to formulate the regression in term of $\dot{X}_j$ and not the entire $\dot{X}_j$. We accentuate this as the main core of the algorithm takes as inputs $\dot{X}_j$ and $\mathbf{\Theta}(X)$ or their Weak versions $\hat{X}_j$ and $\hat{\mathbf{\Theta}}(X)$, so the algorithm has to be re-run for every $j \in [1, 2...n_{var}]$, as will be done when we want to identify a whole dynamical system.

E. Data

In this section we outline the characteristics of the data generation. The Lorenz attractor, Van der Pol oscillator, underdamped harmonic oscillator and Lotka-Volterra system equations will be displayed. What these systems mainly have in common is that they can be represented by a low-dimensional set of first-order Ordinary Differential Equations (ODEs) composed of polynomial combinations of their variables. Additionally, data from a Hall Effect Thruster simulation, representative of the problem motivating the use of the algorithms, was also obtained.

1) Damped Harmonic Oscillator: An oscillator exhibiting damping,

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} - 2\zeta\omega_0\frac{\mathrm{d}x}{\mathrm{d}t} + \omega_0^2 x = 0 \tag{12}$$

where x is the position of the oscillating mass and ω_0 its natural frequency. The trajectories are the balance of the restoring force and the frictional force proportional to the velocity. The second order differential equation can be transformed into a system of first order differential equations by using $v = dx/dt$,

$$\dot{x} = v \tag{13a}$$
$$\dot{y} = -2\xi\omega_0 v - \omega_0^2 x, \tag{13b}$$

We simulate the system with $\xi = 0.1$, $\omega_0 = 3$ for 30 s at a sample rate of 100Hz. Its use in the benchmark is motivated by the simplicity of the system and the secular effects of very small damping in the dynamics. Finally, we build the polynomial feature library with $n = 2$.

2) Lotka-Volterra System: The Lotka-Volterra equations describe the dynamics of two interacting populations: a predator population and a prey population. Prey's growth rate is constant (they find enough food at all times) and their death rate is proportional to the number of predators. Predator's death rate of the predators is constant and their growth rate is proportional to the prey number.

$$\dot{x} = ax - bxy \tag{14a}$$
$$\dot{y} = cxy - dy, \tag{14b}$$

The difference in predator and prey time-scales and the fact that ions and neutrals seem to follow similar dynamics in the Breathing Mode oscillations motivates its use in the benchmark. We used $a = 2/3$, $b = 1$, $c = 1$, $d = 1/3$ and simulated the system for 100 s at 10 Hz sampling frequency at several different initial conditions. We build the polynomial feature library with $n = 2$.

3) The Van der Pol System: Non conservative, self-oscillating system with non-linear damping derived to model oscillations in a vacuum tube triode circuit, represented by equation

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} - \mu(1 - x^2)\frac{\mathrm{d}x}{\mathrm{d}t} + x = 0 \tag{15}$$

Applying the Liénard transformation $y = x - x^3/3 - \dot{x}/\mu$ the Van der Pol oscillator can be written in its two-dimensional normal form.

$$\dot{x} = \mu(x - x^3/3 - y) \tag{16a}$$
$$\dot{y} = x/\mu, \tag{16b}$$

Having the presence of a third-degree nonlinearity motivates its use in the benchmark. We simulate the system for 20 s at a sample rate of 50 Hz for the same initial condition of [1, 1] but with values of the non-linear term μ going from 10 to 0.1. In this case we build the polynomial feature library with $n = 3$.

4) The Lorenz System: Very simplified model for atmospheric instability composed of a system of three variables (the stream function, the change in temperature and the deviation in linear temperature) and three dimensionless parameters (the Prandl number, the Reyleigh number and a ratio of the fluid dimensions). This system of equations exhibits chaotic nonperiodic trajectories which form a strange attractor in 3D space.

$$\dot{x} = \sigma(y - x) \tag{17a}$$
$$\dot{y} = x(\rho - z) - y, \tag{17b}$$
$$\dot{z} = xy - \beta z, \tag{17c}$$

The chaotic nature of the trajectories motivates its use in the benchmark. We simulate the system with $\rho = 28$, $\sigma = 10$ and $\beta = 8/3$ for 10 s at a sample rate of 300 Hz for several starting conditions. We build the polynomial feature library with $n = 2$.

5) SPT-100 Hall Effect Thruster: The 2D axisymmetric hybrid PIC/fluid HET simulator named HYPHEN-HET [30] was used to generate the data for this study, which is available for public use through Zenodo [XXX]. The code uses a fluid description for electrons but treats ions and neutrals kinetically. It includes model for plasma-wall interactions such as ion recombination, the formation of plasma sheaths on the thruster walls and a empirical model for anomalous electron transport, linked to turbulence.

The SPT-100 is simulated by modeling its specific geometry, magnetic topology and cathode position. We chose to use data obtained for a previous data-driven study [6] where stable oscillations appear with a dominant breathing mode. We choose the nominal case used in said study, operating with Xenon with a discharge voltage of $V_D = 300$ V and an anode mass flow rate of $\dot{m}_A = 5$ mg/s.

The data is obtained from 41×49 points in the axial-radial (z-r) simulation domain by time-averaging every 100 simulation steps,

resulting in a sampling time of 1.5 μs for a total of 12001 points. From the simulations we extract the time series for the neutral density and ion density at all points in the discharge channel of the thruster. For obtaining the models, we average spatially over an area downstream the discharge chamber.

Most models in the literature are derived from the ion and neutral continuity equations

$$\dot{n}_i + \nabla \cdot n_i \vec{u}_i = S_{ion} \tag{18a}$$
$$\dot{n}_n + \nabla \cdot n_n \vec{u}_n = -S_{ion}, \tag{18b}$$

where $S_{ion} = R_{ion}(T_e) n_i n_n$ is the ionization source term proportional to the ionization rate R_{ion}, u_i and u_n are the ion and neutral velocities, assumed to be constant. Taking a volumetric average to go from the 3D equations to the 0D global model, most of the literature [31], [32] obtains a model similar to

$$\dot{n}_i = S_{ion} - S_{walls} - n_i \frac{u_{iz}}{L} \tag{19a}$$
$$\dot{n}_n = -S_{ion} + S_{walls} + g_{inj} - n_n \frac{u_{nz}}{L}, \tag{19b}$$

The meaning of the characteristic length L and the shape of the terms g_{inj} and S_{wall} are debated in the literature, mainly coming from uncertainties in the plasma-wall interactions and differences in the extension of the modelled domain. Data-driven techniques are ideal here since having the general form of the equations enables verification of algorithm usage, while terms with known meaning but unknown symbolic form provide the chance to expand or validate existing knowledge without researcher biases. Based on the theoretical model in this project we used $n = 2$ to obtain a self-consistent model for the densities.

III. Benchmark Results

In this section we display the results of the application of the algorithms to synthetic data. We define four variants of the SINDy formulation to try to isolate the effects of the optimizers, formulations and statistical techniques, as can be seen in Table I. ALASSO was chosen as the algorithm of the Bootstrapped and Weak variants after the optimizer comparison, but for clarity the results for the four variants will be displayed together.

	Optimizer	Formulation	Bootstrapping?
"STLSQ"	STLSQ	Differential	No
"ALASSO"	ALASSO	Differential	No
"BALASSO"	ALASSO	Differential	Yes
"wALASSO"	ALASSO	Weak	No

Table I: Algorithmic components of the four variants.

In this benchmark we will first characterize the performance of the Weak algorithm under noise for varying window number and length to find how much the result depends on its parameters. Afterwards the four variants will be compared in terms of the minimal sample rate and time needed for correct system identification. Finally, the four variants will be compared in their ability to find and select the correct models, by considering both the presence of the correct model within the Pareto front as well as the model selected by using the knee criterion outlined in Section II-B.

A. Weak SINDy performance

Our implementation of the Weak formulation depends on two parameters: the number of integration windows $n_{windows}$ (which equals the number of points used in the regression) and the number of points used per window n_{points}. We use data coming from the Lotka-Volterra with increasing levels of noise. We assume the correct model form is known in advance to isolate the effect of the hyperparameters on the resulting coefficients and do an Ordinary Least Squares fit to the correct model with the Weak form of the inputs as explained in Section II-C3. We measure the error in the coefficients, defined as

$$error(\beta) = \frac{\|\beta - \beta_{true}\|_2}{\|\beta_{true}\|_2} \tag{20}$$

for four different number of windows and points per window at every noise level, averaged over 50 realizations. We compare this with the coefficient error from doing the fit on the Differential form of the inputs. The results can be seen in Figure 5 compared to the Differential formulation results.

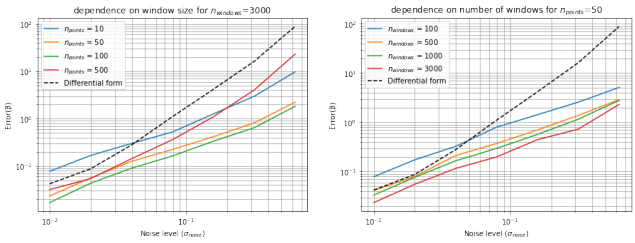

Figure 5: Dependance of the Weak SINDy algorithm performance on the number and size of the integration windows in terms of the coefficient error for increasing levels of noise with a comparison with the use of the Differential form. Analysis done for the Lotka-Volterra model with $\Delta t = 0.033s$, sampling frequency $f_s = 30Hz$

.

In general, it can be seen that using the Weak formulation clearly reduces coefficient error when compared with the Differential formulation. There also seems to be optimal choice of parameters, in this case $n_{points} \approx 100$ and $n_{windows} \approx 3000$. However, there seems to be sub-optimal results for too large or too small window sizes, while increasing the number of points (windows) on which the regression takes place has increasingly marginal returns. This can be linked to the fact that using the trapezoidal rule acts as a low-pass filter [16], where selecting the wrong cut-off frequency can introduce artifacts or smooth out important dynamics. Indeed we would suspect and have tested, the choice of optimal parameters depends on the dynamical system under study, related to the characteristic frequency of the system.

B. Dependence on sampling characteristics

The success of system identification is also expected to be related to the sampling rate and sampling time of the training trajectory. A low sampling rate may result in information loss and less accurate models. Similarly, at a constant sampling rate smaller sampling times enable the observation of fast dynamics, while larger times can smooth out important transient behaviors. Again, we would expect concrete limit values to have a dependency on the frequencies of the system under study, too.

We define an identification frontier as the set of lowest sample rates for which the optimizer is still able to correctly identify the system, for every sample time. In our case we are mainly interested in how this frontier varies for the different variants, so we simulate the Lotka-Volterra system varying the two parameters and run the SINDy algorithm with the 4 different variants. If equations (14) are present within the resulting Pareto sweeps, we count it as a correct identification. We perform the analysis

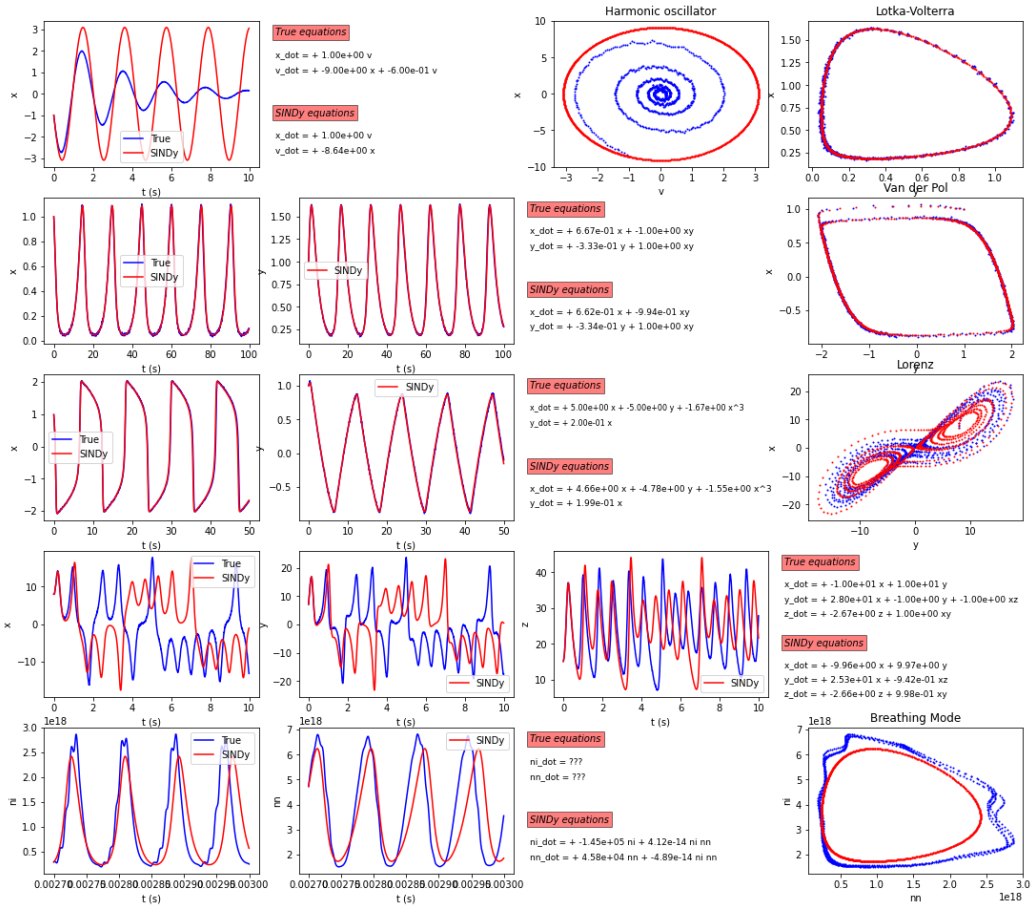

Figure 4: Display of the true and simulated dynamics next to the original and identified equations for the 4 synthetic systems and the Breathing Mode data. The real and simulated dynamic attractors can be seen on the right. 5% noise was added to the synthetic data to better differentiate among the trajectories and display EP2SINDy's capabilities.

for a single, noise-less trajectory (same initial conditions for the integration at each sample rate and size setting).

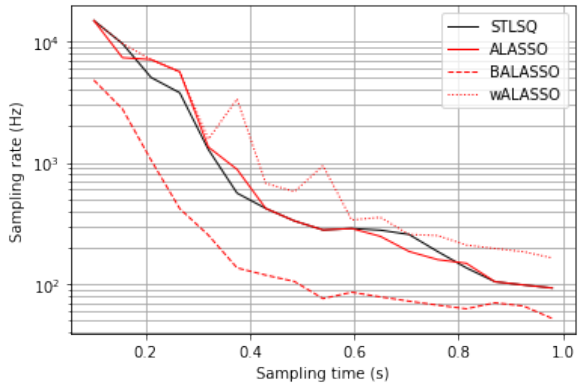

Figure 6: Point plot displaying the correct (green) or incorrect (red) identification of the correct model in the case of the Lotka-Volterra system for three of the variants. ALASSO extends the identification boundary a bit, but data bootstrapping allows to extend it to very low sampling rates and times.

The resulting frontiers of identification are plotted in Figure 6, where the biggest improvement can be seen for the Bootstrapped ALASSO; this could be explained by the better approximation of population statistics for sparse data by resampling with replacement. For the other variants, the detection frontier is very similar, with a slight worsening for the Weak ALASSO variant. It should be remarked that neither the Weak formulation nor the oracle properties Adaptive LASSO seem to improve the frontier. For the later, this is a good reminder that true model selection is only guaranteed for infinite sample sizes.

C. System identification for noisy data

We now test the four variants on the data from the four synthetic systems outlined in Section II-E, with the addition of the "knee" selection step to select the optimal model among the Pareto front. We use 100 realizations of randomly selected initial conditions (or the μ parameter in the case of the Van der Pol oscillator) at three different noise levels. In this and the following sections the noise will be generated from a Normal distribution with zero mean and variance dictated by a user-defined noise level σ_{noise} and the Root Mean Squared Error of the data, i.e.

$$Normal(\mu = 0, \sigma = \sigma_{noise} \cdot \text{RMSE}(data)) \qquad (21)$$

and summed to the original time series. For each system, variant and noise level we measure the number of times the correct model appears in the Pareto front (see Table II), the percentage of times the Pareto optimal model corresponds to the correct one (Table III) and the average number of terms of the Pareto optimal model correctly identified (Table IV).

We can see that, in terms of finding the correct model, ALASSO has a better performance in general than STLSQ, with the exception being the Van der Pol system. Furthermore, the bootstrapped and Weak variants generally outperform the base algorithm, with remarkable identification numbers.

For higher noise levels the Bootstrapped variant significantly outperforms the rest, while the Weak variant performance is variable depending on the dynamical system, and only slightly better than the basic ALASSO; this may be due to what was observed in Section III-A, as the window parameters were kept constant between systems at $n_{points} = 100$ and $n_{windows} = 3000$ instead of finding the optimal ones for each. The fact that the Bootstrapped variant performs better when sampling rates and times are way above the lower limits seen in Section III-B indicates that the base optimizers are inconsistent in their feature selection for the noisy cases. This makes Bootstrapping essential for any real-world application, as even the slightest level of noise for finite sample sizes can make the optimizer miss the true model.

When it comes to the accuracy of the Pareto criterion for model selection, some odd things are visible; for some cases, correct selection improves for higher levels of noise. Moreover, because the model scoring is based on the same data for the STLSQ, ALASSO and BALASSO variants, we would expect their model selection scores to be similar if not equal, but this is not the case. The first irregularity could be related to a significant drop in scores for all models except for the correct one when noise is added, but the second one seems to hint that our selection method is very sensible to the shape of the Pareto front in general, and not only the presence of the true model.

For the wALASSO variant, scoring is done in terms of the weak versions of the inputs; this new scoring given near-perfect selection rates for the Lotka-Volterra system independent of the noise level, while the contrary is true for the Underdamped Oscillator.

The heavy dependence of the results on the dynamical system under study can be explained from looking at Table IV or Figure 4. In the case of the Lorenz system, the correct model is identified within the Pareto front many times, but it seems that it always dropped for a simpler one 6-term model with the y term in $\dot{y}$ missing. This is due to the little weight it has on the dynamic equation and the use of the parsimony criterion. This also seems to be the case for the noise-less selections in the Underdamped Oscillator, as the small damping term is the one being dropped out. However, we know from integrating the models in Figure 4 that these terms have a strong influence on the trajectory, either because the system is chaotic (Lorenz) or because of secular effects (underdamped oscillator).

IV. Breathing Mode Results

Based on the previous results we choose to use the BALASSO variant to look for models in the scenario of interest. We use 32 bootstraps with one feature randomly culled per bootstrap. The results can be seen in Table V and Figure 7.

score (R^2)	n_{terms}	d_{Pareto}	model
-8.46	1	9.46	$\dot{n}_i = 1.89 \cdot 10^{-15} n_n^2$
0.63	2	**0.49**	$\dot{n}_i = -1.45 \cdot 10^{5} n_i + 4.12 \cdot 10^{-14} n_i n_n$
0.69	3	0.59	$\dot{n}_i = -2 \cdot 10^{5} n_i + 4.31 \cdot 10^{-14} n_i n_n + 2.13 \cdot 10^{-14} n_i^2$
-2.56	1	3.57	$\dot{n}_n = 2.04 \cdot 10^{-14} n_i n_n$
0.85	2	**0.36**	$\dot{n}_n = 4.58 \cdot 10^{4} n_n - 4.89 \cdot 10^{-14} n_i n_n$
0.88	3	0.51	$\dot{n}_n = -7 \cdot 10^{4} n_n - 5.02 \cdot 10^{-14} n_i n_n - 4.16 \cdot 10^{-15} n_n^2$

Table V: Some of the models obtained along the Pareto front for the ion (top) and neutral (bottom) dynamics, along with their respective metrics. In bold, the minimal Pareto distance for each variable.

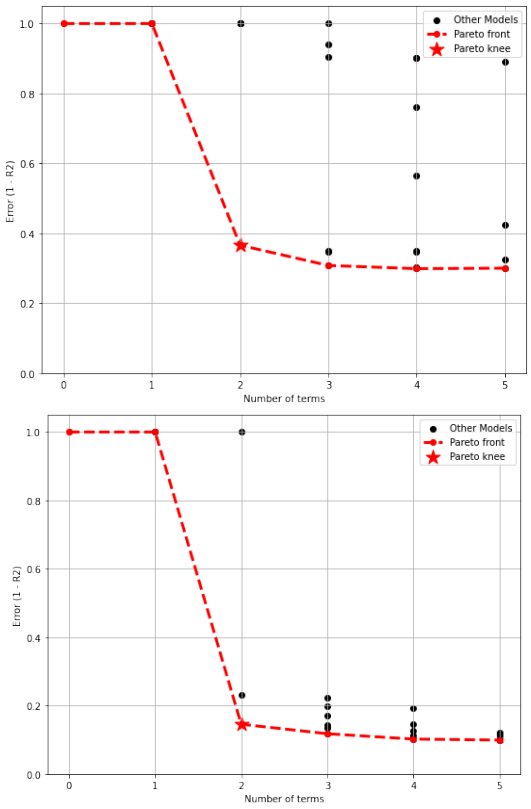

Figure 7: Pareto front of both ions (top) and neutrals (bottom).

Examining the resulting Pareto front first, we can see the signs of correct identification from the sharp drop in error when for the 2-term models. We can also see how the error does not decrease much further for increasing number of terms and saturates at around 25% (10%) for the ions (neutrals). This can be related to the higher-frequency dynamics observed in the original data which cannot be modelled by our low order library.

Looking at Table V, the 2-term models follow the exact form of the predator-prey or Lotka-Volterra model, with the neutrals acting as prey and the ions as predators. We integrate the equations and compare their trajectories with the original data to see the correct amplitude and shape of oscillation, but at a slightly different frequency as plotted in Figure 4. Looking at the system attractor, we see how SINDy filtered the high frequency components coming from other type of oscillations, and the result matches the Lotka-Volterra limit-cycle attractor.

We can compare the coefficients with values expected from (19), with typical values for Hall Effect Thruster magnitudes taken from their average values over the averaged downstream area and compared to the ones appearing in the optimal models of Table V:

$$\begin{aligned} u_i/L &\approx 5500m\ s^{-1}\ /\ 2.5cm = 2.20 \cdot 10^5 s^{-1} \\ u_n/L &\approx 260m\ s^{-1}\ /\ 2.5cm = 1.04 \cdot 10^4 s^{-1} \\ R_{ion}(T_e) &\approx 7.17 \cdot 10^{-14} m^3 s^{-1} \end{aligned} \tag{22}$$

where $R_{ion}(T_e)$ stands for the ionization rate, the term going with $n_i n_n$. The computed values lie pretty close to the observed one. Along with an order of magnitude difference between ion and neutral convection terms we find ionization terms that, although different, share the same magnitude. The neutral injection follows a proportional model, which comes unexpected from intuition but may be due to the simplifications needed to make the 0D model work. Guided by this, it would be interesting to see if adding the electron temperature and/or ion velocities and their respective governing equations would change the model.

Finally, it should be highlighted that the model was discovered successfully even with differences of tenths of orders of magnitude between the coefficients. This is mainly thanks to the normalization and renormalization steps, which lets us obtain results in meaningful SI units of the system unlike other implementations of the algorithm.

V. Conclusions

In this study we have seen that STLSQ and ALASSO have comparable performance in most cases. Contrary to what was to be expected, they both exhibit unstable feature selection as can be seen by the improvement in detection performance by using Bootstrapping.

The automatic hyperparameter tuning was tested in challenging scenarios where it was shown how it missed very relevant terms of the dynamics (dissipation, terms of little effect on the train trajectory…) due to their strictly-secular contribution. While this didn't hamper the identification of the predator-prey dynamics, it has been shown that for future use it will be necessary to either use manual hyperparameter selection, use many trajectories for validation or change the model selection criteria. The last one seems promising, as we have seen in the benchmark that SINDy is able to identify the correct models within its hyperparameter sweep.

Another identified intrinsic limitation of the SINDy + Pareto methodology is its sole reliance on regressing to the system dynamics. As we have seen, even the Weak formulation which uses the trajectory data directly does not incorporate information about temporal causality. Moreover, the regression process is uncoupled for each variable, while the quality of the integrated model depends on the coupling between variables. This seems to lead to obtaining models with the wrong frequency in the Breathing Mode case. One solution would be to include integration within the optimization loop to regress directly on causal trajectory data; a sort of Integral Least Square done once the correct features are found. This can also open the possibility of keeping the Pareto knee criteria but using an Integral score for each model, instead of the one based on the dynamics. We could also include knowledge from the real system as physical constraints.

Practically, the use of the Adaptive LASSO with Bootstrapping in the differential formulation was shown to be the most robust in our case of application; for data with a significant degree of noise the same algorithm with the coupled use of Bootstrapping + Weak formulation could yield better results. While the Weak formulation was tested to lower coefficient error and improve model identification in some cases, its use is conditioned to the use of optimal parameters for the window size and length. Judging this may not be possible if the true equations are not known in advance. Finally, in the current implementation automatic selection is not recommended unless required, but future expansions of the Pareto analysis or alternative approaches might change this outcome.

Based on all this and looking into the future, for the Final Master's Thesis we plan to:

- Try other model selection criteria (AIC/BIC).
- Develop an Integral Least Squares method to fit the most promising features coming from the SINDy regression on the dynamics to the trajectory data, through an optimization loop involving model integration at several windows.
- Extend our application of the algorithm in the real system of interest
  - Include the electron temperature and ion velocities into the model search
  - Incorporate physical constraints in the model search process.
  - Refine the existing models by performing a final Integral Least Squares step.
  - Perform a parametric analysis of the coefficients by fitting the model to several operating points, with the possibility of finding a control law.
  - Exploit the automatic nature of the algorithm to perform a pointwise model search, to highlight the expected spatial term dependences.

Overall the algorithm developed here shows much promise to model many plasma phenomena observed in electric propulsion with simple models, but we must be wary of its limitations and use our insight as researchers to supervise the algorithm.

References

[1] Choueiri, E., "Plasma oscillations in Hall thrusters," *Physics of Plasmas*, Vol. 8, No. 4, 2001, pp. 1411–1426.

[2] Fife, J. M., *Hybrid-PIC modeling and electrostatic probe survey of Hall thrusters*, Ph.D. thesis, Massachusetts Institute of Technology, 1998.

[3] Boeuf, J. and Garrigues, L., "Low frequency oscillations in a stationary plasma thruster," *J. Applied Physics*, Vol. 84, No. 7, 1998, pp. 3541–3554.

[4] Barral, S. and Ahedo, E., "Low-frequency model of breathing oscillations in Hall discharges," *Physical Review E*, Vol. 79, 2009, pp. 046401.

[5] Boeuf, J., "Tutorial: Physics and modeling of Hall thrusters," *J. Applied Physics*, Vol. 121, No. 1, 2017, pp. 011101.

[6] Maddaloni, D., Domínguez-Vázquez, A., Terragni, F., and Merino, M., "Data-driven analysis of oscillations in Hall thruster simulations," *Plasma Sources Science and Technology*, Vol. 31, No. 4, apr 2022, pp. 045026.

[7] Plyashkov, Y. V., Shagayda, A. A., Kravchenko, D. A., Lovtsov, A. S., and Ratnikov, F. D., "On scaling of hall-effect thrusters using neural nets," *Journal of Propulsion and Power*, Vol. 38, No. 6, 2022, pp. 935–944.

[8] Shashkov, A., Tyushev, M., Lovtsov, A., Tomilin, D., and Kravchenko, D., "Machine learning-based method to adjust electron anomalous conductivity profile to experimentally measured operating parameters of hall thruster," *Plasma Science and Technology*, Vol. 24, No. 6, 2022, pp. 065502.

[9] Lee, M., Kim, D., Lee, J., Kim, Y., and Yi, M., "A data-driven approach for analyzing Hall thruster discharge instability leading to plasma blowoff," *Acta Astronautica*, Vol. 206, 2023, pp. 1–8.

[10] Jorns, B., "Predictive, data-driven model for the anomalous electron collision frequency in a Hall effect thruster," *Plasma Sources Science and Technology*, Vol. 27, No. 10, 2018, pp. 104007.

[11] Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, Vol. 113, No. 15, 2016, pp. 3932–3937.

[12] Thakur, B., Sen, A., and Chaubey, N., "Data Driven Discovery of a model equation for anode-glow oscillations in a low pressure plasma discharge," *Physics of Plasmas*, Vol. 29, No. 4, 2022, pp. 042112.

[13] Lore, J., De Pascuale, S., Laiu, P., Russo, B., Park, J.-S., Park, J., Brunton, S., Kutz, J., and Kaptanoglu, A., "Time-dependent SOLPS-iter simulations of the tokamak plasma boundary for model predictive control using SINDy," *Nuclear Fusion*, Vol. 63, No. 4, 2023, pp. 046015.

[14] Dam, M., Brøns, M., Juul Rasmussen, J., Naulin, V., and Hesthaven, J. S., "Sparse identification of a predator-prey system from simulation data of a convection model," *Physics of Plasmas*, Vol. 24, No. 2, 2017, pp. 022310.

[15] Alves, E. P. and Fiuza, F., "Data-driven discovery of reduced plasma physics models from fully kinetic simulations," *Physical Review Research*, Vol. 4, No. 3, 2022.

[16] Schaeffer, H. and McCalla, S. G., "Sparse model selection via integral terms," *Physical Review E*, Vol. 96, No. 2, 2017.

[17] Messenger, D. A. and Bortz, D. M., "Weak Sindy: Galerkin-based data-driven model selection," *Multiscale Modeling amp;amp; Simulation*, Vol. 19, No. 3, 2021, pp. 1474–1497.

[18] Messenger, D. A. and Bortz, D. M., "Weak sindy for partial differential equations," *Journal of Computational Physics*, Vol. 443, 2021, pp. 110525.

[19] Zheng, P., Askham, T., Brunton, S. L., Kutz, J. N., and Aravkin, A. Y., "A unified framework for sparse relaxed regularized regression: SR3," *IEEE Access*, Vol. 7, 2019, pp. 1404–1423.

[20] Cortiella, A., Park, K.-C., and Doostan, A., "Sparse identification of nonlinear dynamical systems via reweighted L1-regularized least squares," *Computer Methods in Applied Mechanics and Engineering*, Vol. 376, 2021, pp. 113620.

[21] Fasel, U., Kutz, J. N., Brunton, B. W., and Brunton, S. L., "Ensemble-sindy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and Control," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 478, No. 2260, 2022.

[22] Mangan, N. M., Kutz, J. N., Brunton, S. L., and Proctor, J. L., "Model selection for dynamical systems via sparse regression and information criteria," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 473, No. 2204, 2017, pp. 20170009.

[23] "On the convergence of the Sindy algorithm," *Multiscale Modeling & Simulation*, Vol. 17, No. 3.

[24] Rudy, S. H., Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Data-driven discovery of partial differential equations," *Science Advances*, Vol. 3, No. 4, 2017.

[25] Zou, H., "The Adaptive Lasso and its Oracle Properties," *Journal of the American Statistical Association*, Vol. 101, No. 476, 2006, pp. 1418–1429.

[26] Mueller, J. L. and Siltanen, S., *Linear and nonlinear inverse problems with practical applications*, SIAM, 2012.

[27] Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B., "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," *2011 31st International Conference on Distributed Computing Systems Workshops*, 2011.

[28] Keller, A. A., *Multi-objective optimization in theory and practice I: Classical methods*, Bentham Science Publishers, 2017.

[29] Titterington, D. M. and Wang, B., "Convergence properties of a general algorithm for calculating variational Bayesian estimates for a normal mixture model," *Bayesian Analysis*, Vol. 1, No. 3, 2006.

[30] Domínguez-Vázquez, A., *Axisymmetric simulation codes for Hall effect thrusters and plasma plumes*, Ph.D. thesis, Universidad Carlos III de Madrid, Leganés, Spain, 2019.

[31] Hara, K., Sekerak, M. J., Boyd, I. D., and Gallimore, A. D., "Perturbation analysis of ionization oscillations in Hall effect thrusters," *Physics of Plasmas*, Vol. 115, No. 20, 2014, pp. 203304.

[32] Dale, E. T. and Jorns, B. A., "Two-zone Hall thruster breathing mode mechanism, Part I: Theory," *36th International Electric Propulsion Conference*, IEPC 2019-354, Vienna, Austria, 1019.

	UnderdOsc			L-V			VdP			Lorenz		
Noise level	0%	5%	20%	0%	5%	20%	0%	5%	20%	0%	5%	20%
STLSQ	100	100	95	100	81	37	99	55	0	97	0	0
ALASSO	100	100	100	100	92	60	100	18	1	100	21	0
BALASSO	100	100	100	100	100	97	100	60	5	100	61	15
wALASSO	100	100	100	100	94	86	100	41	1	100	51	0

Table II: Number of realizations where the correct model was identified for the four algorithms under study.

	UnderdOsc			L-V			VdP			Lorenz		
Noise level	0%	5%	20%	0%	5%	20%	0%	5%	20%	0%	5%	20%
STLSQ	100	53.0	57.9	99.0	100	16.2	100	96.3	N/A	0	N/A	N/A
ALASSO	0	47.0	46.0	92.0	98.9	1.7	73.0	100	0	0	0	N/A
BALASSO	0	0.11	0.11	92.0	90.0	1.0	78.0	83.3	0	0	0	0
wALASSO	0	0	0	99.0	100	98.8	32.0	22.0	0	N/A	N/A	N/A

Table III: Percentage of correct model selections. Only the cases where the correct model was present within the Pareto front were accounted for. Not Applicable cases correspond to those where there was not a single realization where the correct model was within the Pareto front.

	UnderdOsc (3)			L-V (4)			VdP (4)			Lorenz (7)		
Noise level	0%	5%	20%	0%	5%	20%	0%	5%	20%	0%	5%	20%
STLSQ	3.00	2.17	2.17	3.99	3.83	3.66	3.99	3.72	2.11	6.01	4.75	4.53
ALASSO	2.00	2.0	2.0	3.92	3.88	3.81	3.50	2.95	2.13	6.00	5.08	5.88
BALASSO	2.00	2.71	2.49	3.92	3.83	3.43	3.50	3.00	2.16	6.00	5.28	5.1
wALASSO	2.00	1.91	1.18	3.99	3.91	3.88	2.64	2.39	1.95	6.00	5.81	5.26

Table IV: Average number of terms from the correct model identified within the Pareto optimal model.