



Uncertainty Quantification of a Physics-Informed Model based on Sparse Identification of a Thermal Energy Distribution System

September 2025

Changing the World's Energy Future

Paul RM Seurin, Linyu Lin



INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, LLC

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

Uncertainty Quantification of a Physics-Informed Model based on Sparse Identification of a Thermal Energy Distribution System

Paul RM Seurin, Linyu Lin

September 2025

**Idaho National Laboratory
Idaho Falls, Idaho 83415**

<http://www.inl.gov>

**Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517**

Uncertainty Quantification of a Physics-informed Model Based on Sparse Identification of a Thermal Energy Distribution System

Paul Seurin^{1,*}, Linyu Lin¹

¹Idaho National Laboratory, 1955 N. Fremont Avenue, Idaho Falls, ID 83415

ABSTRACT

Integrated energy systems (IES)s are crucial for enhancing the economy and efficiency of power generation sources (e.g., nuclear energy) necessary to unleash American energy dominance. These systems can be integrated with thermal energy storage (TES) and intermittent renewable energies to optimize overall energy use, peak-load regulation, and demand-side responses. However, the stabilization of energy generation, transport, and utilization introduces operational complexities that exceed the challenges of managing each sub-component individually. Currently, though IESs rely on human operators for efficiency and stability, reducing human error risk and enhancing performance through automation is highly desirable. Recent advances at Idaho National Laboratory have demonstrated successful control of the Thermal Energy Distributed System (TEDS). However, the automatic control system depends on a deterministic Sparse Identification of Nonlinear Dynamics with Control (SINDyC) model, which are trained based on simulation data from physics-based simulations. Because of uncertainties in physics-based simulation, SINDyC model results in large discrepancies against experimental data and can not be reliably used in automatic control. In this paper, we present an innovative approach to address these discrepancies by quantifying uncertainties and developing a more robust model. We first generated trajectories by using first-principles physics codes to encapsulate the experiment. Next, we trained thousands of models by randomly sampling these trajectories. We then collapsed all those models into one probabilistic SINDyC by fitting a multivariate Gaussian distribution onto the resulting coefficient's distribution. Despite its simplicity, our approach successfully produced 95% confidence intervals that captured the experimental trajectories. It even did so with a higher probability and better U-pooling score across six of the seven relevant quantities of interest (QoIs), as compared to other classical approaches. Ongoing research is focusing on generating new experimental trajectories to validate this approach, and on employing Bayesian calibration to refine parametric uncertainties and guide future model development efforts.

Keywords: Thermal Energy Distribution System, Sparse Identification of Nonlinear Dynamics with Control, uncertainty quantification, ensemble methods, MultiVariate Gaussian Fit SINDyC

*paul.seurin@inl.gov

NOMENCLATURE

α	sparse regression weight in the STLSQ algorithm
β	inclusion probability hard threshold to prune coefficient in the E-SINDyC algorithm
ϵ	hard threshold to prune coefficient in the STLSQ algorithm
AI	artificial intelligence
CDF	cumulative distribution function
DAE	differential-algebraic equations
E-SINDyC	Ensemble-SINDyC
FT	Friedman test
GHX	glycol heat exchanger
LASSO	Least Absolute Shrinkage and Selection Operator
max iter	maximum number of iterations in the STLSQ algorithm
MPC	Model Predictive control
MSE	mean squared error
MvG-SINDyC	MultiVariate Gaussian Fit SINDyC
NT	Nemenyi test
ODE	ordinary differential equation
PDE	partial differential equation
PI	proportional-integral
QoI	quantity of interest
ROM	reduced-order model
SINDyC	Sparse Identification of Nonlinear Dynamics with Control
STLSQ	Sequentially Thresholded Least Square Algorithm
TEDS	Thermal Energy Distributed System
TES	thermal energy storage
UQ	uncertainty quantification

1. INTRODUCTION

Integrated energy systems (IESs) are complex heterogeneous architectures designed to coordinate different assets (e.g., a nuclear power source and an hydrogen electrolyzer [1]) for improved overall economy in comparison to using each asset individually [2]. Their complexity arises due to the information and energy flow across components that perform different roles and entail different physical constraints, including temperatures, pressures, and flow rates [2]. One example is the Thermal Energy Distributed System (TEDS) in the Dynamic Energy Transport and Integration Laboratory at Idaho National Laboratory. It serves as a testbed for evaluating the performance and reliability of hybrid generation of electrical power and non-electrical products [3]. In a pre-experimental phase, a Modelica-based model of the system [4, 3] was developed to inform system design and potential control methodologies so that future IESs could be operated safely and effectively [3]. Currently, decoupled proportional-integral (PI) controllers manage the different TEDS components. Each PI controller adapts the signals to user-defined setpoints, including fluid temperatures at heater outlets, mass flow rates through pumps, and fluid temperatures at pump inlets. Supervisory controllers that integrate all these signals while also accounting for physical constraints and their coupled interactions are an active area of research [2]. These controllers are intended to improve the economic value proposition of IESs by effectuating actions more efficiently than can be done by humans, theoretically resulting in substantial cost savings.

It is in this regard that advanced control methodologies built around model predictive control (MPC) are currently being explored [2]. MPC proposes control actions based on expected transients of TEDS state variables in response to exogenous inputs and /or disturbances such as grid demand fluctuations or weather variability. Modelica and Dymola based physics-based simulations offer a power tool for capturing thermal hydraulics in an equation-based framework for IESs. For MPC use cases, Dymola can export a functional mock-up unit (FMU), which encapsulates the compiled solver and exposes model variables (states, inputs, and outputs) through a standardized API. References [5] and [6] demonstrates the use of Dymola FMU in a nonlinear MPC framework for controlling a vapor-compression system and a drum boiler system. However, real-time requirements and numerical-solvers issues can be challenging when physics-based models are directly coupled in the MPC system. For each MPC iteration, the differential-algebraic equations (DAE) systems must be integrated over the prediction horizon, which usually requires small timesteps to handle stiff dynamics and can be difficult to meet the real-time requirements. Besides, the black-box DAE solvers encapsulated by off-the-shelf FMUs are not easily differentiated or optimized over. As system size grows with multiple components and systems like in IES, the exploding DAE dimension will further slow down the real-time prediction and cause solver failures and convergence issues. To mitigate the slowness in practice, a high-fidelity physics model is used to generate simulation data, then a fast surrogate model is fitted over the relevant operating envelop.

However, this technology relies on the system model's accuracy remaining valid long enough to determine optimal control actions. Moreover, the model must respond very rapidly, as demand responses must be met within a matter of a few seconds while also considering both efficiency and safety. The model also needs to be interpretable, where model terms and forms should explicitly connected to dominant physical interpretations, and easily integrated with optimizers and differentiators for control action optimizations. Surrogate forms like Sparse Identification of Nonlinear Dynamics with Control (SINDyC) are inherently differentiable and can be reliably solved inside the MPC optimizer, simplifying gradient-based solution methods and enhancing the robustness of MPC control systems. These surrogates focus on the input-output mapping over the MPC's region of interest and ignore unexcited dynamics. As a result, the state dimensionality is drastically reduced and stiff subsystems that don't affect control can be removed, which offers tailored complexity and orders-of-magnitude faster than repeated DAE solves.

With the advent of artificial intelligence (AI) approaches and the achieved increases in computational power,

the past decade has seen the emergence of a range of foundation predictive models that are both fast and highly transferrable across a wide range of system dynamics. Foundation predictive models are deep neural networks trained on massive heterogeneous datasets to learn general-purpose representations that can be specialized to downstream tasks with minimal additional training. In physics-based simulations, foundation predictive models are trained based on data from a wide range of dynamical systems, sidestepping the need to derive or solve governing equations explicitly [7]. Reference [8] shows the use of multiscale operator transformer to learn solution operators of diverse differential equations with high sample efficiency and continuous-in-time inference capabilities. Negrini et al. [9] further fuses numerical inputs and scientific text descriptions to provide predictions and interpretability. Sun et al. [10] demonstrates the use of multi-modal neural network approach for encoding and decoding both numerical and symbolic datatypes. However, these methods often function as black boxes, making them difficult to interpret or trustfully deploy. In real-world applications such as IESs, it is crucial that techniques be interpretable, trustworthy, and safe [11]. In the highly regulated nuclear market, this poses a significant obstacle to the large-scale deployment of new AI-based technologies [12], thus accounting for the gap between AI's promise and its limited adoption within this field. Utilizing interpretable methods for IES control is therefore pivotal, and will bring AI technologies one step closer to application in real-world systems, resulting in substantial savings for the industry. This may be enabled by adopting methods from the realm of model identification.

Model identification or model discovery is a class of methods designed to generate low-order governing equations that capture essential physical phenomena through statistical optimization. While physics-informed methods for deep learning are burgeoning [11], the model identification methods are more trustworthy and interpretable because they work directly with known physical relationships and constraints, providing insights into the nature of the underlying dynamical system rather than fitting a black-box model. In the past 10 years, applications of these methods in physics have proliferated, including techniques such as nonlinear principal component analysis with nonlinear autoencoder [13], Koopman operator with extended dynamic mode decomposition [14, 15], neural operators [16, 17], symbolic regression [18], relevance vector machine [19], Gaussian process [20], flow map learning [21], and SINDyC [22, 23]. These methods have revealed powerful applications in system identification, forecasting, and optimization, with applications ranging from ocean mesoscale closure models [19], to thermal-hydraulic transient analysis [24], to nuclear core reload physical model prediction [18], to air separation unit scheduling optimization under time-varying electricity prices [13]. Arguably, the primary goal of modeling complex dynamics via low-order equations that outperform physics-based codes in terms of speed is to utilize the model for advanced control systems. This reduces overall system costs [25, 26, 27, 28, 15], and among the noticeable recent applications in this regard are nuclear reactor control [28, 29] and chemical engineering process control [25]. However, many methods require vast amounts of data and do not perform well under high noise [30]—something commonly found in real-world systems. Comparing to physics-based models, surrogate models are fast computing and easier to integrate with MPC. Comparing to foundation predictive model, surrogate models require less data than foundation models and thus excel in data-scarce environments [31]. Besides, comparing to black-box approaches [32], methods like SINDyC result in an analytical equations with explicit forms, resulting in better interpretability and feasible uncertainty quantification.

Of all these methods, SINDyC in particular has gained traction, thanks to its simplicity, performance, and interpretability, even for highly nonlinear systems. SINDyC methods can identify ordinary differential equations (ODEs) [22] and partial differential equations (PDEs) [23] with explicit analytical equations, allowing direct interpretation of dominant physical terms and facilitating analysis. Meanwhile, by enforcing sparsity, SINDyC requires relatively few data samples compared to black-box surrogates, especially when dynamics are intrinsically low-dimensional [31]. SINDyC-based methods have been used in applications such as plasma modeling [33], forward propagation of uncertainty [34], structural mechanics [35], control of heat-pipe microreactors under normal and abnormal conditions [29] and, most importantly, control of TEDS [36]. Despite their prevalence, these reduced-order models (ROMs) require substantial engineering

expertise in order to perform well. Increased system non-linearity can degrade ROM performance [24]. In SINDyC, this can be mitigated by using intricate higher-order terms in governing equations (called libraries, see Section 2.1), but there are no sound theories for optimal term selection. Therefore, use of efficient and robust SINDyC models under limited-data and high-noise conditions is crucial for enabling trustworthy control of real-world engineering processes, but adds a layer of complexity.

The present work presents a novel approach to develop a robust SINDyC-based model for control applications with limited experimental data, as applied to Idaho National Laboratory’s TEDS. We first generated data (i.e., trajectories) via the physics code Modelica, ensuring that the time series approached the real-world system’s evolution. However, this model has inherent inadequacies (e.g., mechanical latencies due to physical spacing between sub-components) that Modelica does not reproduce. These inadequacies were addressed via uncertainty quantification (UQ) of the SINDyC model’s parameters Ξ and Θ (explained in Section 2). The uncertainties were obtained by training thousands of SINDyC models and then aggregating them altogether with UQ to build a single probabilistic model. The resulting model provided a predictive distribution $p(x|\Xi, \Theta)$ —with 95% confidence intervals—that had to encapsulate the experiment along the trajectories. The model also had to obtain a high score with respect to other metrics (e.g., the U-pooling metric) that measure the similarity between two distributions [37] (i.e., the probabilistic model and the experimental trajectories and noise). Multiple approaches could then be taken in deploying the control, such as using the worst-case extrema, mean, median, or distribution mode for each parameter [30, 25]. The contributions of the present work are as follows:

1. Developing a robust SINDyC model using ensemble methods against uncertainties in training data when high-quality experimental data are limited.
2. Applying a novel probabilistic ensemble methods that accounts for the correlation between the model’s coefficients, physical interpretations, and uncertainties in physics-based models. This involves fitting a multivariate Gaussian for coefficients of a linear SINDyC model with dominant state variables and external control actions, here termed the Multivariate Gaussian Fit SINDyC (MvG-SINDyC).
3. Optimizing the performance of MvG-SINDyC trained with different numbers of trajectories or bootstrap.

Section 2 describes the theory behind the SINDyC algorithm (see Section 2.1) and the recent developments pertaining to UQ for SINDyC-based models (see Section 2.2). Section 3 describes TEDS (see Section 3.1) and the characteristics of the experimental and Modelica trajectories (see Section 3.2). Section 4 discusses the results, and Section 5 presents the conclusion, along with areas of future research.

2. SPARSE IDENTIFICATION OF NONLINEAR DYNAMICS WITH CONTROL (SINDyC)

2.1. Original Definition

SINDyC [22] is a model discovery method for identifying the ODEs and PDEs that govern observed data. The term “trajectories”, which we will employ throughout the remainder of this paper, is often used in PySINDy [38] to denote the data. The model to identify is typically that given in Equation 1:

$$\frac{d}{dt}X = f(X, U; \Xi, \theta) = \Xi\Theta(X, U)^T, \quad (X(0) = X_0, U(0) = U_0) \quad (1)$$

where $X \in \mathbb{R}^N$ is the state of the system, $U \in \mathbb{R}^M$ represents the exogenous or control inputs, $\Xi \in \mathbb{R}^{N \times K}$

represents the coefficients, and $\theta(X, U) = \begin{pmatrix} f_1(X, U) \\ f_2(X, U) \\ \dots \\ f_K(X, U) \end{pmatrix}$ represents a library of K transformations (e.g.,

$1, \sin(x), x^2, \dots$), which can also include cross terms between X and U . The choice of library is pivotal to the method's performance. Partial knowledge of the physics can be readily included by adding more terms, and the values of the coefficients in Ξ afford a natural assessment of each term's contributions to the evolution of the state X , thus making it easily interpretable. Because the coefficients of Ξ are usually sparse in most dynamical systems [22, 30], SINDyC models are derived via sparse regression, by solving Equation 2; parsimony also promotes generalizability to unknown trajectories [39]:

$$\Xi_k^* \in \underset{\Xi_k}{\operatorname{argmin}} \|\dot{X}_k - \Xi_k \Theta(X, U)^T\|_2 + R(\Xi_k) \quad (2)$$

where k indicates the k th row of Equation 1, and $R(\Xi_k)$ is a term that enforces parsimony. $R(\Xi_k) = \alpha \times \|\Xi_k\|_0$ for the sequentially thresholded least squares (STLS) algorithm or $R(\Xi_k) = \alpha \times \|\Xi_k\|_1$ was utilized for a convex regression with the Least Absolute Shrinkage and Selection Operator (LASSO) [40] in discovering ODEs [22, 30], with α being a trade-off factor for identifying the Pareto optimal model that best balances model complexity and accuracy. However, the LASSO approach did not perform well in the presence of highly correlated parameters, especially when dealing with PDEs [23]. An extension of it, termed the Sequential Thresholded Least Square Algorithm (STLSQ) [30], revised the parsimony component of the regression with $\epsilon \|\Xi_k\|_0 + \alpha \|\Xi_k\|_2$, which solves a ridge regression-like problem via hard thresholding. A tolerance level β is used to iteratively refine Ξ until a satisfying predictor can be found. The pseudo-code for fitting a SINDyC model with STLSQ is given in Algorithm 1. This method was implemented in PySINDy [38], the package that was used to obtain the SINDyC models in this work.

Algorithm 1 Pseudocode for STLSQ (adapted from [23, 30])

```

1: Input: state  $X$ , actuator  $U$ , libraries  $\Theta$ , sparsity-promoting coefficients  $\alpha$ , threshold  $\epsilon$ , maximum number of
   iterations max iter
2:
3: Output: Final coefficients for the SINDyC models  $\Xi$ 
4:
5: •Compute  $\Theta(X, U)^T$ 
6: •Compute time derivative  $X_t$  (e.g., finite difference)
7: for  $k = 1, \dots, N$  do
8:   •Initialize  $\Xi_k = \frac{\theta_k X_{t,k}}{\theta_k^T \theta_k + \alpha I}$ 
9:   for  $j=1, \text{max iter}$  do
10:    •Threshold small coefficients  $t_{ind} = (j, \Xi_{k,j} < \epsilon)$  and  $\Xi_k(t_{ind}) = 0$ 
11:    •Evaluate  $\tilde{t}_{ind} = (j, \Xi_{k,j} \geq \epsilon)$ 
12:    • $\Xi_k(\tilde{t}_{ind}) = \frac{\theta_k(\tilde{t}_{ind}) X_{t,k}}{\theta_k^T(\tilde{t}_{ind}) \theta_k(\tilde{t}_{ind}) + \alpha I}$ 
13:    if  $t_{ind} == \Xi_k.shape$  or  $\Xi_k = \Xi_k(\tilde{t}_{ind})$  then
14:      •break
15:    end if
16:   end for
17: end for
18: •Return  $\Xi$ 

```

2.2. Uncertainty Quantification with SINDyC Models

The viability and performance of methods developed to act upon systems (e.g., forecasting, optimization, and control) depend on the accuracy of the mathematical models used to represent the intricate dynamics and physical interactions that occur within them. In practice, while first-principles and physics-based models are derived based on thorough physical and mathematical assumptions, significant prediction errors can still occur between an ideal *in silico* model and a physical experiment. These errors can arise from various sources of uncertainty. Quantifying these uncertainties is pivotal to ensure robustness, performance, and safe model deployment. This process of quantifying uncertainties in the input parameters, along with their propagation to the outputs, is known as UQ [41].

There are three primary types of uncertainty: aleatoric uncertainties, which stem from experimental noise; model inadequacies, which refer to deficiencies in the model’s ability to capture the full physics of the system; and parametric uncertainties, which arise from uncertainty in the model parameters. Often the model inadequacy and parametric uncertainties are embedded under the umbrella term *epistemic uncertainties*. Additional trajectories can reduce the parametric uncertainty, but cannot reduce aleatoric uncertainties and model inadequacies, both of which require more effort to characterize. Similarly, the parametric uncertainty could also be improved by performing detailed hyper-parameter tuning of the model [42], while the other two remain unchanged again. Quantifying each type of uncertainty can help maximize resources and the information gained from new experiments, and can also help guide model development [43, 44, 30]. In machine-learning-based models, parametric uncertainties tend to dominate, due to the potentially large number of parameters to tune. In contrast, in physics-based models, which are usually built from first principles, model inadequacy typically dominates.

Dealing with uncertainty in SINDyC coefficients is a relatively recent challenge, with techniques primarily focused on addressing robustness to noise as well as compensating for a limited number of trajectories. While the sparsity helps focusing on core dynamics and improve the robustness of the models to noise and overfitting [45], SINDyC models are still highly sensitive to noise. This is particularly true in cases involving PDEs, as the noise can be greatly amplified when computing high-order derivatives. Similar to a neural network that tends to overfit to the underlying noise in the training dataset, various techniques were developed to enhance robustness to noise and reduce overfitting.

Because taking the derivative is known to be unstable and sensible to noise, approaches including integral [46] and weak formulations [47] were developed. They involved reformulated the PDE problem by multiplying it by weights $w_p(x, u, t)$ and integrating over a domain Ω_k , which can be repeated over different combinations of weights and domains to then solve a new system of equation with sparse regression. For instance in the integral case, the PDE $\dot{X}(t) = F(x, u)$, where $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ can be integrated and approximated with a fourth order Runge-Kutta (RK4) scheme such that $x(t_{k+1}) \simeq x(t_k) + \frac{1}{6}\Delta t_k(k_1 + 2k_2 + 2k_3 + k_4) = \mathcal{F}_{RK4}(x(t_k), \Delta t_k, f(\cdot))$, where $\Delta t_k = t_{k+1} - t_k$, $k \geq 0$ and $k_1 = f(x(t_k))$, $k_2 = f(x(t_k) + \Delta t_k \frac{k_1}{2})$, $k_3 = f(x(t_k) + \Delta t_k \frac{k_2}{2})$, $k_4 = f(x(t_k) + \Delta t_k k_3)$, and $f(\cdot)$ (equal to $\Xi\Theta(X, U)$ with the sparsity assumption) is a gradient field information [45], which lead to a loss function of the form $\sum_k \|x(t_{k+1}) - \mathcal{F}_{RK4}(x(t_k), \Delta t_k, \Xi\Theta(X, U))\|$. On the other hand, iNeural-SINDy [45], employs a neural network to learn an implicit representation of the data, which output provides a denoised version in the vicinity of the noisy data. Automatic differentiation was then utilized to obtain more accurate derivatives, while an integral form loss was added to improve the robustness to noise [45]. On top of that, these models can violate basic laws, noticeably because of incorrect derivatives, SINDy with Side Information (SINDy-SI) [48] was explored, where the authors injected physics-information, symmetries, and invariance in the shape of polynomial constraints using a form of a lifted Semi-Definite Programming problem.

Other methodologies closer to the work proposed include sub-sampling with co-teaching [25]. In sub-sampling, a subset of the trajectories, which can consist of snapshots in time, is sampled to train a model.

This process is repeated until a satisfactory number of models has been trained. The best model is then selected based on specific criteria such as the Akaike information criterion (see Equation 4), which balances error (e.g., the mean squared error (MSE) and sparsity (e.g., $\|\Xi\|_0$). Given the m^{th} SINDyC model M_m :

$$MSE_m = \frac{1}{T} \sum_{t=1}^T (M_m(t) - M_{exp}(t))^2 \quad (3)$$

$$AIC_m = T \times \log MSE + 2\|M_m\|_0 \quad (4)$$

When sub-sampling is coupled with co-teaching, several noise-free trajectories generated from an approximate first-principles model are also sampled, the rationale being that this will lower the error value by pushing the coefficients in basin of attractions away from the noise. However, despite the production of probabilistic forecast information being useful for more robust control, it is unavailable in all of these methods, which only provide what is the “best” fit.

To account for uncertainty in SINDyC model predictions, the essential approaches are Bayesian and non-Bayesian. An example of the former is sparse Bayesian inference [39, 49, 50], and an example of the latter would be an ensemble-based approach such as bagging and bragging [30] or dropout-SINDyC [25]. For simplicity, we term these approaches Ensemble-SINDyC (E-SINDyC). Bayesian approaches promote sparsity in the model realizations by using sparsity-inducing priors such as Laplace, spike-and-slab, and regularized horseshoe priors. Ensemble-based approaches aggregate the predictions of different models trained on different bootstrap samples of trajectories and libraries, forming the posterior distribution $p(X|\Xi, \Theta)$. There is a clear parallel with practices in deep learning methodologies such as Bayesian neural networks, dropout-based variational inference (also called Monte Carlo Dropout), and deep learning ensembles [51, 42]. The main limitation of Bayesian approaches is that they rely on computationally costly methods based on Markov Chain Monte Carlo in order to obtain the posterior distribution of the parameters, which may take hours or even days. In contrast, ensemble-based approaches can achieve the same result in a matter of seconds yet still retain a similar level of performance with an improved tolerance to noise [30]. Furthermore, ensemble-based methods are very simple to implement. Therefore, the present work focuses on the latter class of methods. It should be noted however, that the integral (e.g., in iNeural-SINDy [45]) and weak formulations (e.g. in E-SINDyC [30]) can be coupled to other techniques to improve the robustness of the overall approaches.

In the context of SINDyC, there are two types of levers to adjust during training: the trajectories and the libraries. In ensemble-bragging and dropout-SINDyC, the libraries $(\Theta_i)_{i \in \{1, \dots, K\}}$ are randomly added and removed, respectively, and one model is trained in this manner. In ensemble-bagging, sets of randomly sampled trajectories are utilized to train each model separately. The inclusion probability of each coefficient is then considered in order to prune the model even further. Inclusion probabilities for the coefficients can be obtained by evaluating, across the different models, the portion of non-zero occurrences for each parameter. A deterministic prediction can then be given by the mean, median, or mode of each coefficient based on their distribution across the different models, which was shown in [30] to surpass deterministic SINDy models for a MPC task applied to the Lorentz system in the high-noise, limited-trajectories region. Alternatively, for probabilistic forecasting, the sample mean $\mu(X) = \frac{1}{M} \sum_{m=1}^M M_m(X, U)$ and sample variance $\sigma^2(X) = \frac{1}{M} \sum_{m=1}^M (M_m(X, U) - \mu(X, U))^2$, where $(M_m(X, U))_{m \in \{1, \dots, M\}}$ represents M SINDyC models, can then be used to provide confidence intervals for the true value of X . The alternative method that we propose starts off in a manner similar to the ensemble method. However, once the distribution of coefficients is obtained, a multivariate Gaussian is fitted to the coefficients. We term this method MultiVariate Gaussian Fit SINDyC (MvG-SINDyC). The advantage of this approach is that it naturally computes a covariance matrix that accounts for the correlation between the coefficients of the model. Moreover, this distribution could be calibrated further via inverse UQ [41, 44] once more experimental trajectories are collected with limited parameters compared to the ensemble approach. Algorithm 2 summarizes the approaches.

Algorithm 2 Pseudocode for MvG-SINDyC and E-SINDyC models

```
1: Input: state  $X$ , actuator  $U$ , libraries  $\Theta$ , optimizer  $\text{optim}^*$ ,  $\text{optim args}^*$ , number of library terms  $l$ , number of
   bootstraps  $q$ , threshold  $\beta$ , method
2:
3: Output: Final coefficients for the SINDyC models  $(\Xi_i)_{i \in \{1, \dots, q\}}$  or mean  $\mu$  and covariance  $\Sigma$ 
4:
5: •Compute  $\Theta(X, U)^T$ 
6: •Compute time derivative  $X_t$ 
7: for  $k = 1, \dots, N$  do
8:   •Choose  $l$  libraries at random  $\Theta_{1, \dots, l}$ 
9:   •Choose  $q$  trajectories at random  $(X, U)_{1, \dots, q}$ 
10:  •Train a model  $\Xi_k = \text{optim}((X, U)_{1, \dots, q}, \Theta_{1, \dots, l}, *args)$ 
11: end for
12: •Evaluate inclusion probabilities  $(\chi(\Xi_k) \in [0, 1]^K)_{k \in \{1, \dots, N\}}$ 
13: •Threshold  $\Xi = \{\Xi_1, \dots, \Xi_q\}(\chi(\Xi_i) \geq \beta)$ 
14: if method == E-SINDyC then
15:   •return  $\Xi$ 
16: if method == MvG-SINDyC then
17:   • $(\mu, \Sigma) \sim \mathcal{N}.fit(\Xi)$ 
18:   •return  $(\mu, \Sigma)$ 
19: end if
20: end if
```

* The optimizer is, for instance, the STLSQ from Algorithm 1, and then the hyperparameters are the sparsity-promoting coefficients α , the threshold ϵ , and maximum number of iterations iter must be provided.

2.3. Generation of Trajectories for SINDyC

As with every data-driven model, SINDyC models are very sensitive to the trajectories on which they are trained. Multiple factors that affect the quality of the trajectories—and hence the performance of the SINDyC model—must be carefully considered. These factors include, but are not limited to, the sampling period used to record the trajectories, the controls or actuators U and inputs X used to generate and collect trajectories, and their initial conditions [25]. All these elements impact the system dynamics information contained in the trajectories. Indeed, the trajectories should cover a large operating range, and this involves avoiding long-idle steady-state periods. Similarly, it might be more challenging to sample high-gradient or unstable regions. These can be addressed through burst sampling [52], which means increasing the number of trajectories sampled locally and/or considering multiple short trajectories. Moreover, basic machine-learning best practices should be followed, including splitting the trajectories between training and validation to perform hyperparameter tuning.

Nevertheless, the particularity of this task is the use of simulations as a proxy for experimental trajectories. While more trajectories seem to provide more comprehensive information to the SINDyC solver, more is not necessarily better, as it introduces more system dynamics variations that may differ from real-world experimental conditions and be harder for the model to capture [25]. This can amplify the discrepancy between the physics code and the experiment. Thus, we must find a trade-off between the information provided to SINDyC and the real operating conditions. Overall, this process requires trial and error as well as engineering intuition. We explain our approach in Section 4.2.

3. DESIGN OF EXPERIMENTS

In this section, we describe the TEDS setup and the experimental and Modelica trajectories.

3.1. Thermal Energy Distributed System

The TEDS facility is composed of three primary components: a 200 kW Chromalox heater, a single-tank packed-bed thermal energy storage (TES) system filled with 0.125-in. alumina beads, and an ethylene-glycol-to-Therminol-66 glycol heat exchanger (GHX). Among the other sub-components are system piping, five control valves, and associated temperature and flow sensors [2, 3]. The experiment consisted of five modes—startup or warm-up, charging, standby, discharging, and cooldown—and terminated at 16,000 seconds [36]. Charging entailed sending hot fluid to the top of the TES tank and collecting cold fluid at the bottom, whereas discharging was the reverse. The present work focuses on the prediction of the state variables for the discharge (9,180 s to 14,640 s). A schematic plot of TEDS and the associated control systems is provided in Figure 1.

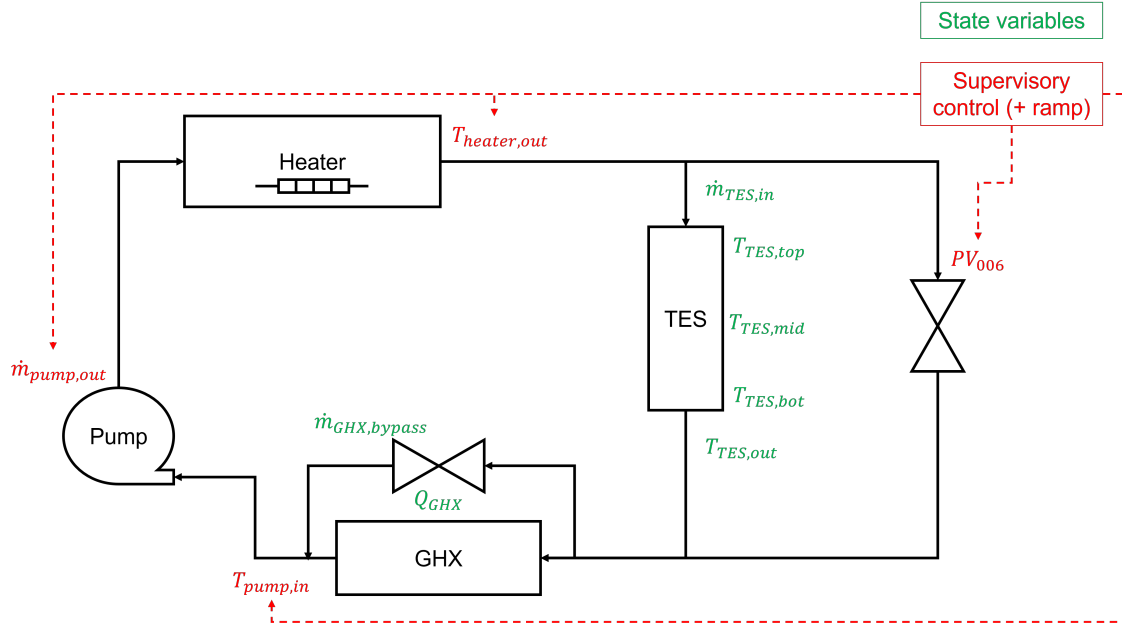


Figure 1. Simplified TEDS and control systems.

As described in Section 1, MPC requires a model of the system's dynamic over multiple time intervals in the future. In [2], a ROM of the TES and the GHX was developed that takes the following form:

$$\dot{x}_a = A_a x + B_a u \quad (5)$$

where a is either the TES or the GHX, and (A_a, B_a) represents the associated matrix found using SINDyC.

For the input, we get $x_{tes} = \begin{pmatrix} \dot{m}_{tes,in} \\ T_{tes,out} \\ T_{top} = T_{tes,node50} \\ T_{mid} = T_{tes,node100} \\ T_{bot} = T_{tes,node150} \end{pmatrix}$ and $x_{ghx} = \begin{pmatrix} \dot{m}_{ghx,bypass} \\ Q_{ghx} \end{pmatrix}$. For the control, we get

$$u_{tes} = u_{ghx} = \begin{pmatrix} PV_{006} \\ \dot{m}_{pump,out} \\ T_{pump,in} \\ T_{heater,out} \end{pmatrix}.$$

The temperature transient, flow rates, and TEDS power were then studied, and the MPC performance relied on finding optimal setpoints for the pump flow rates ($\dot{m}_{pump,out}$), GHX outlet temperatures ($T_{pump,in}$), heater outlet temperatures ($T_{heater,out}$), and pressure valve positions (PV_{006}). Another control lever is the time ramp (not represented in the figure), which accounts for the delay between control actions and their realization on the physical system—again, not considered in Modelica. An application program interface was implemented to exchange information between the control system and the Modelica model for TEDS. While the MPC takes action following the SINDyC-based ROM, the actions are also passed onto the Modelica models of the TES and GHX to ensure that the optimal actions leveraging the ROM formulation extend well to the Modelica-based system. While the performance on a simple test case was satisfactory, there remains room for improvement. For instance, the power slightly oscillated during the ramp, potentially resulting in unnecessary thermal load on the components. Also, the control was exclusively carried out in Modelica, with no incorporation of experiments. It is therefore difficult to trust in the model’s robustness when deployed on a physical system. As discussed in Sections 1 and 2.2, the model must consider the parametric uncertainties in the SINDyC model in order to bridge the gap between the control with SINDyC, with Modelica, and the real physical system. Section 3.2 describes the trajectories generated to achieve this.

3.2. Experimental and Modelica Trajectories

In this work, we had only one trajectory for each quantity of interest (QoI) arising from a single experiment. As explained in Section 2.3, we needed to provide trajectories that exhaustively captured the different operating ranges but were not so complex as to deviate from the experimental conditions. To achieve this, we expanded the possible ranges centered around the true actuators and used Latin hypercube sampling with the Sobol sampler from the SALib package [53]. We found 500 samples to be sufficient for obtaining dense actuator trajectories that encapsulated the experimental ones, as showcased by the dashed dark lines and blue lines in Figure 2.

Table I showcases the ranges utilized in order for the actual actuators to generate the trajectories. In the TEDS process modeled in this work, the discharge occurs between 9,180 and 14,640 seconds, and these ranges are applied to a portion of the discharge. Note, however, that the entire startup, charging, discharging, and cooldown process is perturbed to ensure consistent initial and boundary conditions. The perturbation that occurs before and after the discharge is the same for each trajectory. We then built a linear time-dependent function with fixed values for the other periods. PI controllers modified each component online in order to achieve these setpoints, which is why some of them are not strictly equal to their input values across the whole time series. $\dot{m}_{pump,out}$ was not modified from the input perspective, as we found it destabilized the Modelica model to too great an extent. However, it was indirectly perturbed via the other actuators and their interaction through the PI (see Figure 2 (B)). Lastly, the time ramp simulates the mechanistic delay between the actuator order and its realization in the physical system, as was explained in Section 3.1.

Table I. Ranges resulting from the Sobol sampler utilized for the actuators to generate the trajectories.

actuators	lower bound	upper bound
PV_{006} (-)	0	1
$\dot{m}_{pump,out}^*$ (kg/s)	0.039	0.81
$T_{heater,out}$ (K)	366	504
$T_{pump,in}$ (K)	372	584
time ramp (s)	100	150

* Fixed for this study to ensure the stability of all trajectories.

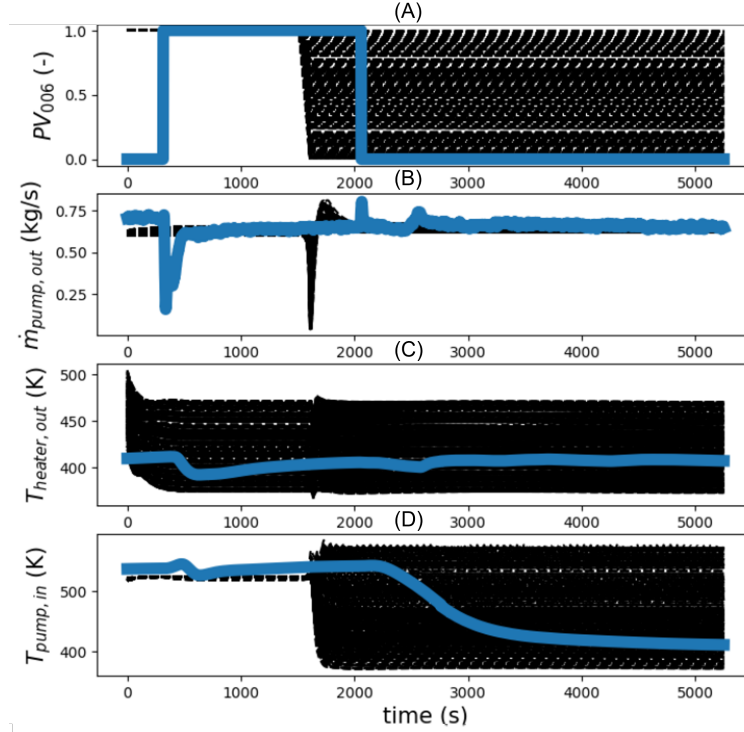


Figure 2. Experimental and Modelica actuators. The blue lines are experiments; the dashed dark lines are actuators.

As observed in Figures 3 (A) and 3 (B), for the TES and GHX systems, respectively, the resulting output signals do not faithfully reproduce the experiment. This is a result of the inadequate physics from Modelica. In Section 4.3, we describe how we will work around these limitations by utilizing multiple sets of different numbers of trajectories. Moreover, the higher discrepancy may warrant greater sparsity for the TES, as will be shown in Section 4.2 and Appendix B. Table II summarizes the characteristics of the training trajectories generated with Modelica. To avoid the problem of sharp gradients, we used an adaptive sampling period that refines the sampling in the presence or absence of sharp gradients within Modelica. To ensure that all the trajectories have the same number of points, we coarsened (or refined) them in order to obtain approximately the same number of points as in [2] (i.e., 5,251 points, as seen in the second column of Table II).

Table II. Summary of the characteristics of the training trajectories used for SINDyC model discovery.

sampling period (-)	number [*] data points in each trajectory (-)	number of ^{**} Modelica trajectories (-)	duration of the experiment (s)
adaptive	5,251	500	5,460

^{*} Though the sampling period was adaptive, we used segment fitting to coarsen (or refine) the time series so as to reduce (or increase) it to this number of points.

^{**} 500 trajectories if 1 sample is used to fit a Modelica model, or 500 sets of trajectories if multiple samples are used each time a model is fitted.

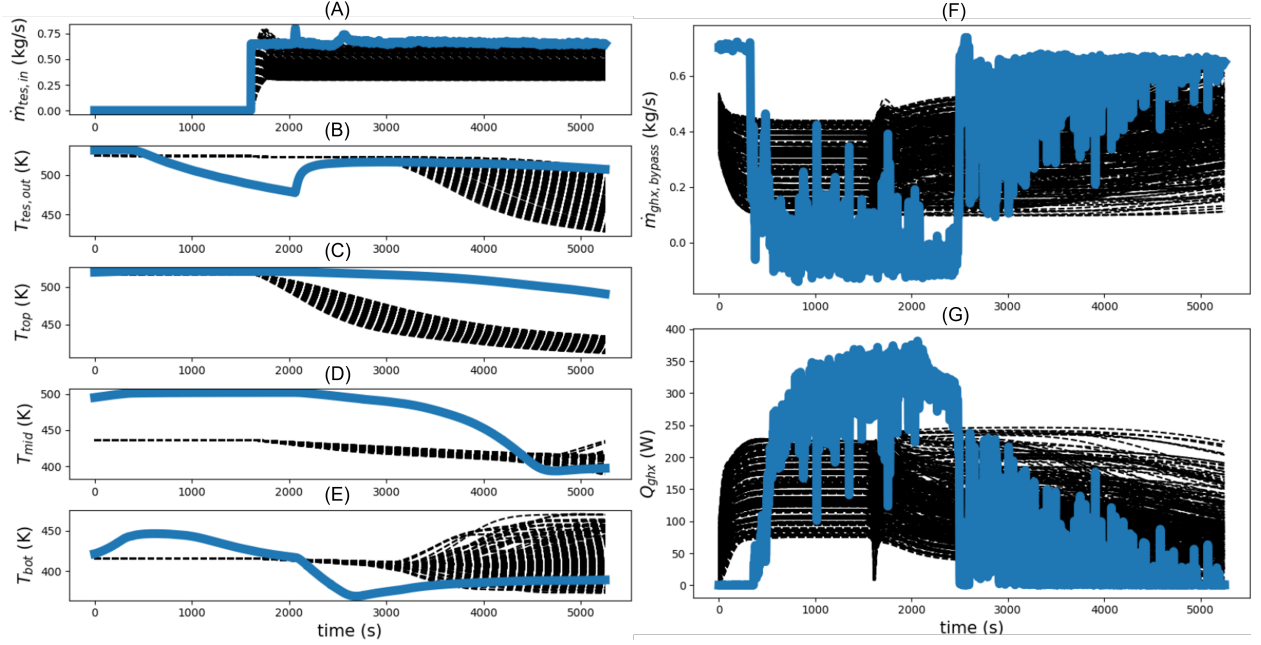


Figure 3. Experimental and Modelica trajectories outputs (or states) for the TES ((A) to (E)) and GHX ((F) to (G)).

4. RESULTS AND ANALYSIS

In this section, we present our approach to obtaining the SINDyC models and their performances (see 4.2); explain our UQ performed with the SINDyC models in 4.3; and summarize the results in 4.5.

4.1. Metrics for Comparing the Performances of the Probabilistic SINDyC Models

In this work, we attempt to build a computational model and validate it with a physical experiment. The validity of a computational model that estimates a real physical system must be rigorously assessed prior to its deployment. Many metrics exist for comparing deterministic data by using point estimates of the experiment and/or model output (e.g., MSE, Minkowski distance, and mean absolute difference [37]). There are also probabilistic metrics in which either the experiment or model output uncertainties are accounted for (e.g., normalized Euclidean distance and Mahalanobis distance [37]), or in which both are accounted for (e.g., Kullback-Leibler divergence, Jensen-Shannon divergence, Hellinger metric, Kolmogorov-Smirnov distance, and the U-pooling metric [37]). The latter are more difficult to interpret but are actually the most accurate—particularly when comparing probabilistic models such as those being built as part of the present research. Moreover, as there is no one-size-fits-all metric, more than one must be leveraged so as to enable a more rigorous comparison between different approaches for selecting the model [54, 37].

To compare the approaches in Section 2.2, we considered two metrics (described later) that account for the uncertainty in the model’s estimate and in the experiment: (1) the U-pooling metric and (2) the percentage of the experiment that falls within the 95% confidence interval. Ultimately, the former was chosen due to being more conducive to comparisons of time series [37].

Let us assume that, at each time-step, the model’s output observation stems from an underlying Gaussian distribution with mean μ_i and standard deviation σ_i , where μ_i is obtained by taking the mean of the

probabilistic model's outputs and σ_i is its standard deviation. The u value u_i is computed using $u_i = G_i(D_i)$, where G_i is the cumulative distribution functions (CDFs) of the latter distribution, and D_i is the experimental realization at timestep i . It is straightforward to see that if $D_i \sim \mathcal{N}(\mu_i, \sigma_i)$ (i.e., comes from the same distribution as the predictive model), the u_i should be uniformly distributed. The U-pooling metric is then the measure of the vertical and horizontal discrepancies between the u_i and F distributions, where F is the CDF of a unit uniform distribution $\mathcal{U}(0, 1)$. To evaluate it, the area metric is considered:

$$d_A(F, G) = \int_{-\infty}^{\infty} |F(x) - G(x)| dx \quad (6)$$

where F and G are CDFs. The U-pooling metric is often utilized in time-series anomaly detection to identify which distributions differ excessively from the regular signals.

The other metric is the confidence interval. For an arbitrary number of samples, the confidence interval that the true mean of a population μ based on n observations can be estimated via a student t-distribution is as follows [55]:

$$\mu \sim (\bar{x} - t_{\frac{\alpha}{2}, \nu} \frac{s}{\sqrt{n}}, \bar{x} + t_{\frac{\alpha}{2}, \nu} \frac{s}{\sqrt{n}}) \quad (7)$$

where the level of confidence is given by $C = 100(1 - \alpha)\%$, and $t_{\frac{\alpha}{2}, \nu}$ is the $1 - \frac{\alpha}{2}$ quantile of the t-distribution for $\nu = n - 1$ degrees of freedom. Asymptotically, the t-distribution approaches the normal distribution. Equation 7 states that “ μ is in this interval with probability C .” However, for robust control when the dynamic model is deployed, we need an uncertainty band on the worst-case scenario—not just the mean. Therefore, the uncertainty in the outputs' estimates will be interpreted as the 95% confidence interval of the output distribution. Assuming a normal distribution, the uncertainty band will be expressed as the 95% confidence interval of $\pm\sigma$. For our model's output, we can calculate it and verify that the experimental value lies within $[\mu - 2\sigma, \mu + 2\sigma]$. It is probably the most important metric, as it enables us to perform more robust control when the dynamic model based on SINDyC is deployed on the physical system.

The purpose of this paper is to develop a probabilistic model that affords a satisfactory trade-off between accuracy and speed. Thus, no threshold values will be kept in mind for discard one model or another. We are merely comparing the methods in light of the selected metrics. Further research will involve calibration of the model with more experimental data (which we do not currently possess), and optimization of the parametric uncertainties with respect to those metrics. But this is beyond the scope of the current study.

4.2. SINDyC Model Procurement and Performance

SINDyC models are very sensitive to the trajectories and the choice of libraries. In some instances, a poor choice of coefficients for SINDyC leads to unphysical relationships between input parameters [25], or to divergence of the prediction as a result of small variations in the coefficients. This makes it necessary that the basis functions be properly chosen, potentially entailing several rounds of manual tuning. In [2], it was decided to utilize a first-order model without cross terms for modeling the system. This simplified the control methodology because there are only two types of basis functions (i.e., 1 and x). However, the model inadequacy is rather large. Therefore, in this paper, the focus is on selecting the right trajectories for training the models, and not on modifying the libraries. Additionally, as explained in Section 2.3, the hyperparameters of the STLSQ (α , ϵ , and max iter) are important.

Table III summarizes the hyperparameters considered for generating the models. For the hyperparameters of the STLSQ, we performed a grid search by using the Modelica outputs as the training set. The number of models for which the prediction of the experiment converged, and the error between the predictions and the experiment trajectories, were used as validation metrics for the choice of hyperparameters. As the central

focus of this study is to build UQ for the SINDyC-based TEDS system, for brevity's sake we omit showing the results.

Table III. Hyperparameters for the STLSQ algorithm for the GHX and TES scenarios.

	α	ϵ	max iter	bootstraps (q)	# of models per set of trajectories
GHX	1e-3	1e-6	20	1,4,8,12,16,20	500
TES	1e-2	1e-3	20	1,4,8,12,16,20	500

It should also be noted that we found that having a dense set of coefficients is better for the GHX scenario, emphasizing the high importance of all the parameters; whereas for the TES, a sparse set was preferred (see Appendix B). We hypothesize this to be due to the larger discrepancies between the experiment and Modelica in the TES case, specifically for T_{top} and T_{mid} (see Figure 3 (A)).

To train the SINDyC models for each set of trajectories of size p , we randomly drew 500 subsets of $\binom{S}{p}$, where S (equal to 500) is the original number of trajectories generated. Here, we describe the results for the GHX, but, for the sake of readability and conciseness, we have moved the TES results to Appendix B. Figures 4 (A) through (C) and Figures 5 (A) through (C) showcase the predictive performance of each model on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for $\dot{m}_{ghx,bypass}$. Figures 4 (D) through (F) and Figures 5 (D) through (F) showcase the predictive performance of each model on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for Q_{ghx} . The blue-shaded areas represent the $\mu \pm 2 \times \sigma$ prediction of the resulting models, the red lines represents the mean of the outputs, the blue and orange curves are the max and min of all the predictions, respectively, the dark lines are the (noisy) experimental trajectories, the light green lines are the corresponding de-noised experimental trajectories filtered via a Savitzky-Golay filter, using the SciPy Python package [56], and the orange arrows were added to underscore the curvature of the mean of the outputs. From these figures focusing on the red curves and orange arrows, we can first observe that the more trajectories are considered for training, the smoother the prediction. This is particularly visible in the first few thousand seconds in Figures 4 (D) through (F) and Figures 5 (D) through (F), where the initial peak disappeared when training with more than 12 trajectories. We hypothesize that this arises from the second-order loss function utilized in SINDyC, which averages the predictive performance across trajectories with various curvatures. As a result, the model smoothed them out. The number of models that converges when predicting the experiment also increases. However, when the number increases too much, several sharp gradients are completely missed. This is particularly visible in Figures 4 (A) through (C) and Figures 5 (A) through (C) at 2,500 seconds, where the dip of $\dot{m}_{ghx,bypass}$ is overpredicted starting from the 16-trajectories case. Moreover, on these Figures we can see that the means are moving up, where for the $\dot{m}_{ghx,bypass}$ it is evolving towards an horizontal line, while the Q_{GHX} 's peak is being smoothed out and the curve is converging to some sort of a bilinear. There thus seems to be a compromise to be found. Table IV can help find the sweet spot. The table provides metrics including the percentage of the experimental trajectories residing within the 95% confidence interval of the posterior probabilistic distribution given by the ensemble of models trained with different number of trajectories and the U-pooling between the distributions and the experimental trajectories. Each metric may give a different story depending on the QoIs of interest, but the compromise seemed to be achieved by the 12-trajectories case, where the percentage of the experimental trajectories that lies within the confidence intervals is high, and the U-pooling are low for both QoIs.

Number of trajectories for training	$\dot{m}_{ghx,bypass}$	Q_{ghx}
1	0.765 / 0.980	0.912 / 0.634
4	0.926 / 1.009	0.405 / 1.209
8	0.895 / 0.914	0.467 / 1.104
12	0.759 / 0.908	0.477 / 1.059
16	0.595 / 0.904	0.435 / 1.010
20	0.588 / 0.890	0.419 / 1.027

Table IV. Percentage of the experimental trajectories residing within the 95% confidence intervals of the model's prediction / U-pooling for models trained with different number of trajectories in the GHX scenario. Highlighted in red is the selected case for further comparison.

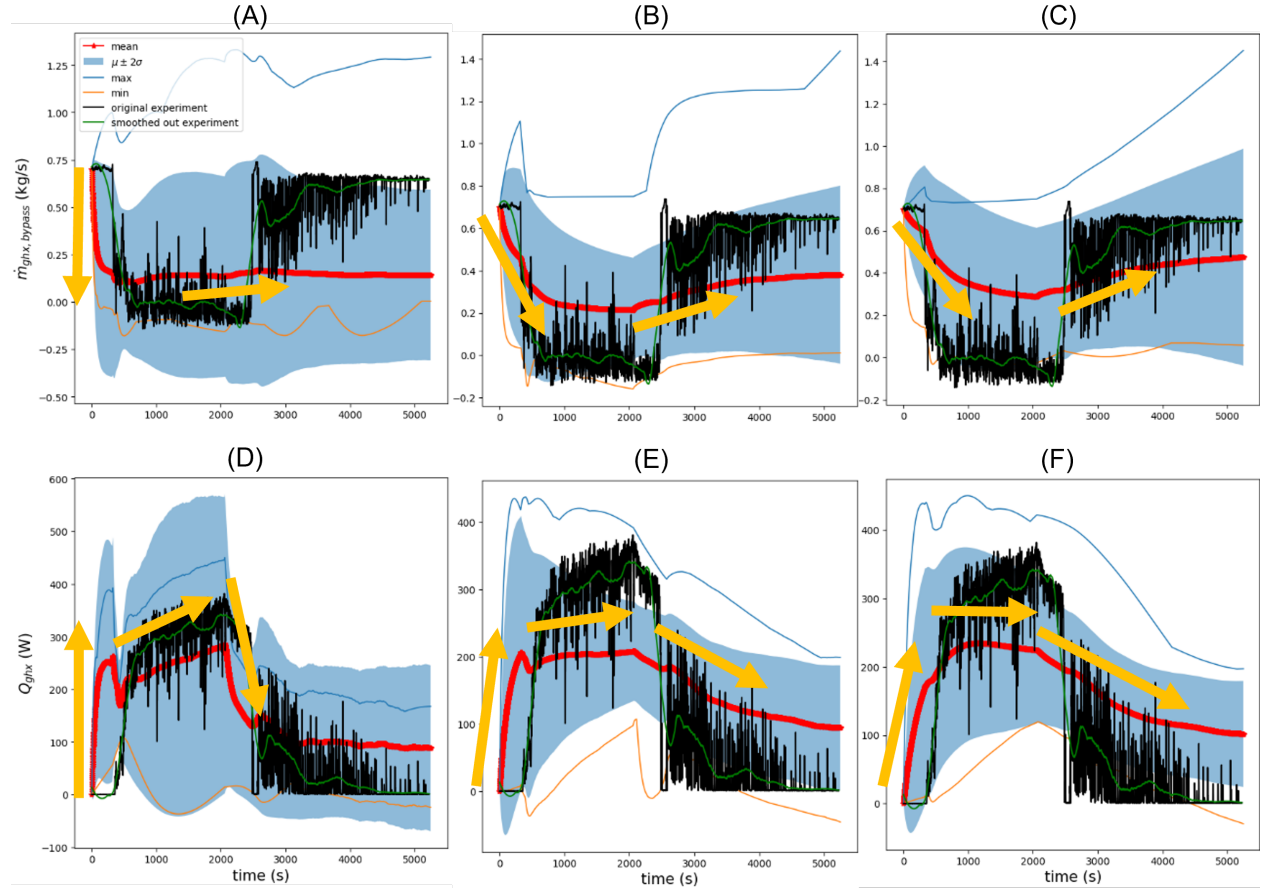


Figure 4. Prediction of the experiment with SINDyC models for $\dot{m}_{ghx,bypass}$ with (A) 1 trajectory, (B) 4 trajectories, and (C) 8 trajectories, and for Q_{ghx} with (D) 1 trajectory, (E) 4 trajectories, and (F) 8 trajectories. Orange arrows were added to underscore the trend of the mean as the number of trajectories to train the SINDyC models increases.

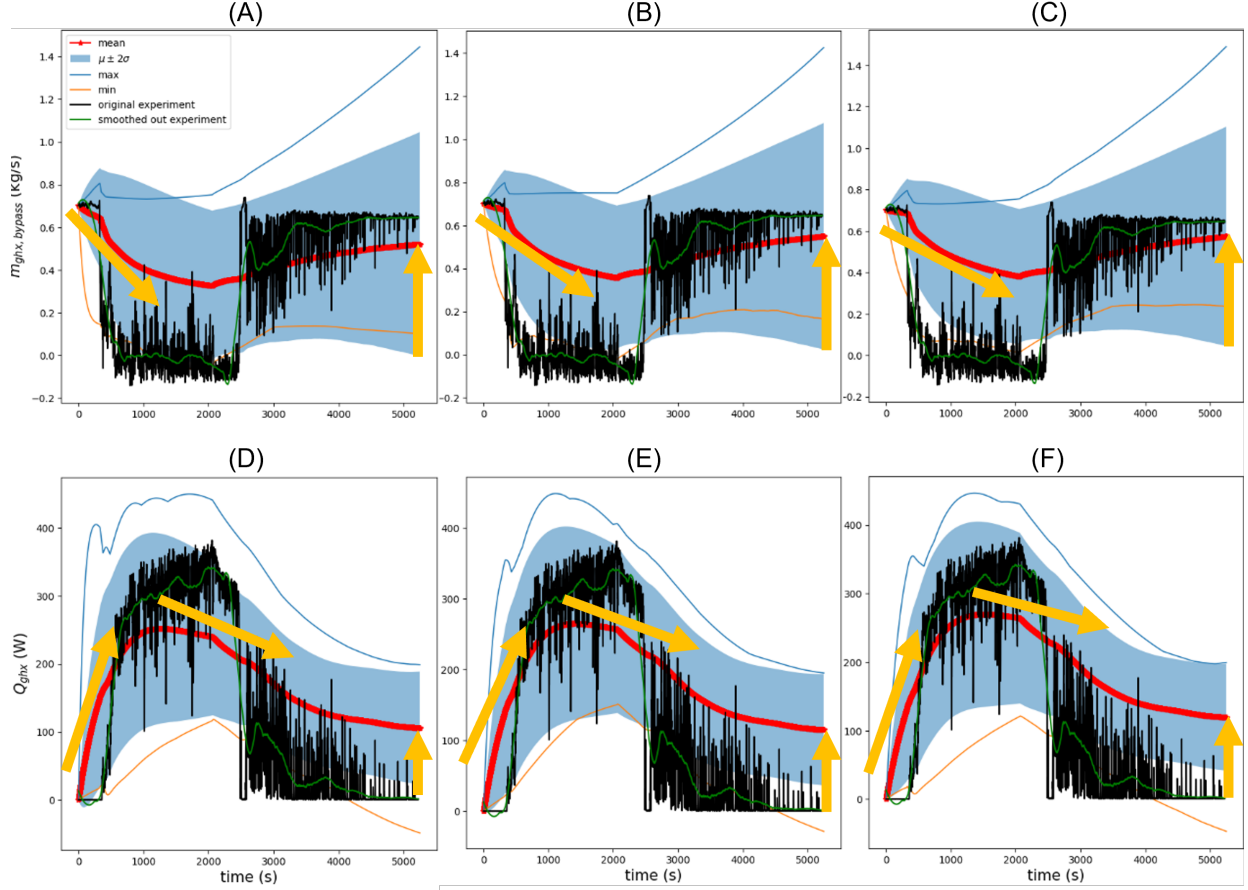


Figure 5. Prediction of the experiment with SINDyC models for $m_{ghx,bypass}$ with (A) 12 trajectories, (B) 16 trajectories, (C) and 20 trajectories, and for Q_{ghx} with (D) 12 trajectories, (E) 16 trajectories, and (F) 20 trajectories. Orange arrows were added to underscore the trend of the mean as the number of trajectories to train the SINDyC models increases.

It should be noted that the model's inadequacies which prevents the model to fit the experimental trajectories perfectly originate from the linear formulation of SINDyC. Nevertheless, there is nothing in the MvG-SINDyC approach that prevents us to develop a similar probabilistic model with higher-order terms. Utilizing higher-order terms in SINDyC could potentially reduce the original inadequacies, but it would subsequently slow-down the decision-making during control. A companion paper is currently being pursued, in which the impact of the MvG-SINDyC based on a linear formulation in control will be evaluated.

4.3. Uncertainty Quantification for SINDyC Models

In this section, we apply two approaches for UQ with SINDyC models: E-SINDyC [30] and MvG-SINDyC derived in this work following Algorithm 2, as explained in Section 2.2. We will also compare both approaches when using only 12 trajectories for E-SINDyC and MvG-SINDyC, which was the trade-off found in Section 4.2 to demonstrate the advantage of mixing models trained on different numbers of trajectories. To our knowledge, this is the first time that (1) a continuous probability distribution has been fit directly onto the SINDyC coefficient distribution, and that (2) a probabilistic model has been built with different numbers of trajectories. Again, we describe the results for the GHX scenario and move the TES results to Appendix

C. Table V summarizes the hyperparameters utilized for building the probabilistic SINDyC models. For the two E-SINDyC-based approaches, we used bootstrapping when sampling the model, meaning we randomly sampled 500 models each time, with replacement, before evaluating the metrics. For the MvG-SINDyC, we sampled from the distribution 500 times.

Table V. Hyperparameters used to obtain the probabilistic SINDyC models for the GHX and TES scenarios.

	Numbers of trajectories for fitting [*]	β	# of models aggregated to build the probabilistic SINDyC model
GHX	1,8,12,20	0	2,000
TES	4,8,12,16	0	2,000

^{*} In other words, the number of trajectories used to fit the (deterministic) SINDyC models that were aggregated to obtain the MvG-SINDyC and E-SINDyC probabilistic models.

First, Figure 6 showcases the inclusion probabilities for each coefficient, along with the correlation between them. These were obtained by considering all 2,000 models trained (see Tables III and V). Most coefficients have a high probability of being integrated, except for A_{11} ($\frac{d}{dt}\dot{m}_{ghx,bypass}, \dot{m}_{ghx,bypass}$) and A_{15} ($\frac{d}{dt}\dot{m}_{ghx,bypass}, T_{pump,in}$), which have probabilities of 0.444 and 0.304, respectively. In this instance, we are tempted to prune these two coefficients; however, we also see that most coefficients are highly correlated. Therefore, we will consider them all for producing predictive confidence intervals (i.e., β is equal to 0 in Algorithm 2). Additionally, the correlation plots in Figure 12 given in Appendix A re-emphasize the high correlation between the coefficients. Typically, there is a straight line with a negative slope between A_{10} and A_{15} , corresponding to a dark square on the correlation table in Figure 6. Moreover, in looking at the diagonal, we see that the distribution of each coefficient closely resembles a Gaussian. That fact, as well as the high correlation between the coefficients, motivated us to fit a multivariate Gaussian to the coefficients rather than use a classical ensemble approach, giving rise to MvG-SINDyC.

Coefficients (GHX)	Inclusion probabilities
A_{10}	1.000
A_{11}	0.444
A_{12}	0.998
A_{13}	1.000
A_{14}	0.817
A_{15}	0.304
A_{16}	1.000
A_{20}	1.000
A_{21}	1.000
A_{22}	1.000
A_{23}	1.000
A_{24}	1.000
A_{25}	1.000
A_{26}	1.000

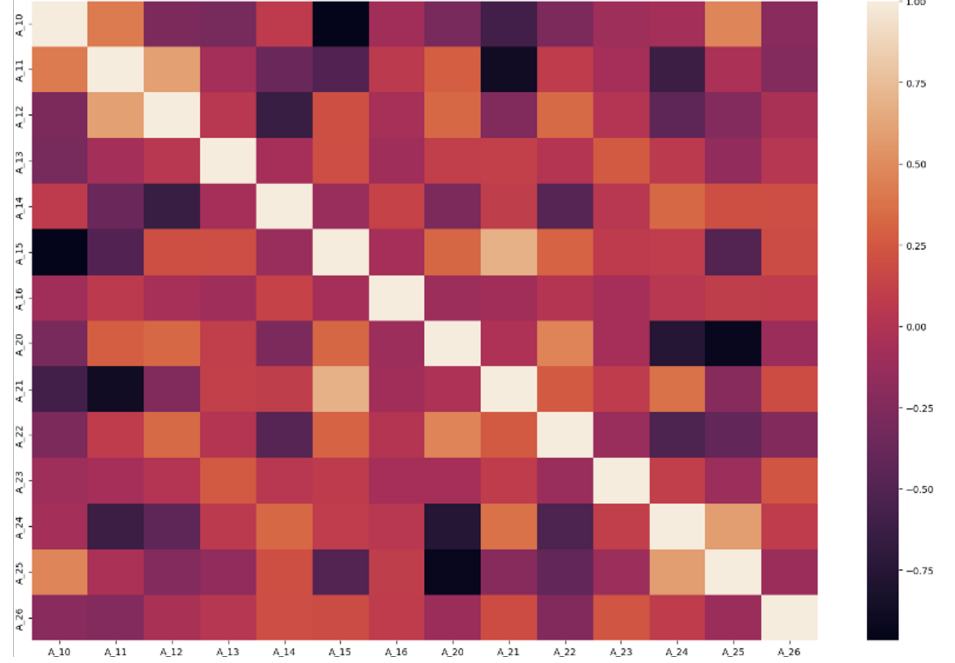


Figure 6. Inclusion probabilities for each coefficient in the SINDyC model, and the correlations between them. For the latter, the lighter color signifies higher positive correlations, while darker colors signifies higher negative correlations.

Figure 7 showcases example predictive confidence intervals for each approach when applied to a randomly sampled set of 500 predictions: Figures 7 (A) and (E) for MvG-SINDyC, Figures (B) and (F) for E-SINDyC, and Figures 7 (C) and (G) for using only 12 trajectories (MvG-SINDyC), for each QoI; Figures 7 (A) through (C) for $\hat{m}_{ghx,bypass}$; and Figures 7 (D) through (F) for Q_{ghx} . The blue-shaded areas represent the $\mu \pm 2 \times \sigma$ prediction of the resulting models. In each instance, our MvG-SINDyC approach surpasses the ensemble one by proposing intervals that better encapsulate the experimental trajectories. The Table VI and Figure 8 confirm these observations. The table provides the mean and standard deviation of the percentage of the experimental trajectories that lie within the 95% confidence intervals of the models' predictions, as averaged over 100 random realizations (i.e., 500 models randomly sampled 100 times for E-SINDyC, and 500 combinations of coefficients randomly sampled 100 times for MvG-SINDyC), while the figure visually represents the distribution of the results across the realizations. The table shows that, on average, over 90% of the experimental trajectories are captured by our approach's 95% confidence intervals prediction, against 89% and 51% for E-SINDyC for $\hat{m}_{ghx,bypass}$ and Q_{ghx} , respectively. Statistical tests could then be utilized to ensure that these observations were statistically significant. Following the suggestion in [54, 57], we used a Friedman test (FT) and a post-hoc Nemenyi test (NT). The p values of the FTs for both QoIs is equal to 0, meaning that at least one of the approaches statistically differs from the other three. The p values of the NTs for MvG-SINDyC, when compared against the other three approaches, is equal to 0, meaning that the latter is statistically different from the other. Considering it has the highest average, we can conclude our MvG-SINDyC approach to be statistically superior.

Moreover, the mean of the prediction, as represented by the red line, is more faithful to the true experimental distributions, re-emphasizing the superiority of our approach in performing UQ for SINDyC. Lastly, the advantage of models obtained from multiple numbers of trajectories rather than a single one is also emphasized by Table VI, with the performance (i.e., the percentage of the experimental trajectories within

the confidence intervals of the models' predictions) being increased from 80.6% and 35.0% to 99.0% and 93.3% for $\dot{m}_{ghx,bypass}$ and Q_{ghx} , respectively, for the 12 trajectories (MvG-SINDyC) and MvG-SINDyC, respectively. Similarly, in the E-SINDyC approaches, the performance increased from 74.9% and 47.2% to 88.8% and 51.2% for $\dot{m}_{ghx,bypass}$ and Q_{ghx} , respectively.

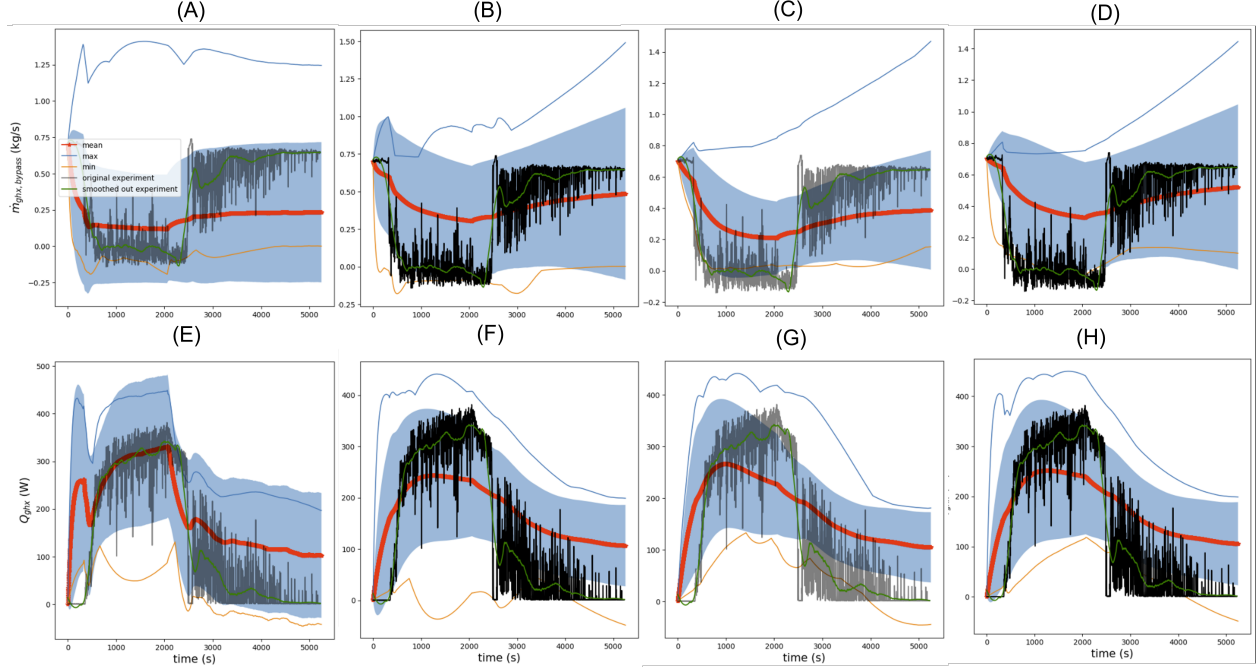


Figure 7. Comparison of the UQ-SINDyC approaches. (A) and (E) for MvG-SINDyC with different numbers of trajectories, (B) and (F) for E-SINDyC with different numbers of trajectories, (C) and (G) for MvG-SINDyC with 12 trajectories, and (D) and (H) for E-SINDyC with 12 trajectories.

Table VI. Percentage of the (filtered) experimental trajectories within the 95% confidence intervals of the models' predictions (i.e., between 2.5% and 97.5% of the mean). The results highlighted in red indicate the best approach.

Method	$\dot{m}_{ghx,bypass}$	Q_{ghx}
MvG-SINDyC	0.990/0.053	0.933 / 0.073
E-SINDyC	0.888 / 0.035	0.512 / 0.029
12 trajectories (MvG-SINDyC)	0.806 / 0.057	0.350 / 0.021
12 trajectories (E-SINDyC)	0.749 / 0.048	0.472 / 0.035

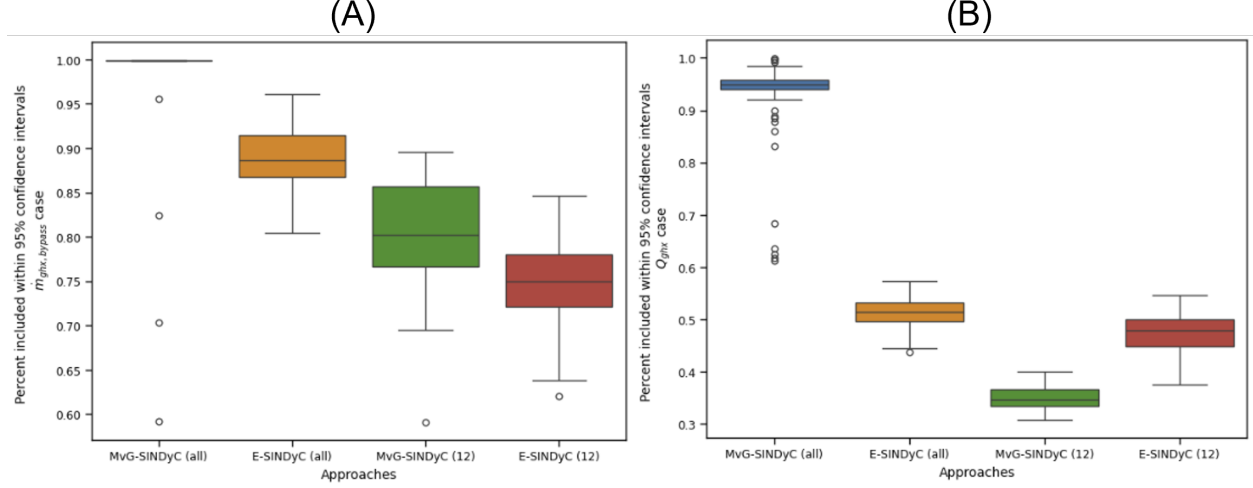


Figure 8. Distribution of the percentage of the experimental trajectories within the 95% confidence intervals of the models' predictions across 100 realizations for (A) $m_{ghx,bypass}$ and (B) Q_{ghx} .

The U-pooling score for each approach is given in Table VII, and the distributions across the 100 realizations are shown in Figure 9. For this metric, the score is lower for MvG-SINDyC as well. The FTs succeeded and the NTs demonstrated that all post-hoc pairwise tests were significant for the best cases (except between MvG-SINDyC and E-SINDyC trained on multiple trajectories), with a p value of 0.91. Lastly, we can observe once again that injecting different numbers of trajectories is beneficial for obtaining better performance with respect to the U-pooling metric: the score decreased from 1.220 and 1.208 to 0.959 and 0.767 for $m_{ghx,bypass}$ and Q_{ghx} , respectively, for the 12 trajectories (MvG-SINDyC) and MvG-SINDyC, respectively. Similarly, as regards the E-SINDyC approaches, the performance decreased from 0.995 and 1.114 to 0.965 and 1.052 for $m_{ghx,bypass}$ and Q_{ghx} , respectively.

Table VII. U-pooling metric for each approach. The lower the better. The results highlighted in red indicate the best approach.

Method	$m_{ghx,bypass}$	Q_{ghx}
MvG-SINDyC	0.959 / 0.039	0.767 / 0.081
E-SINDyC	0.965 / 0.012	1.052 / 0.029
12 trajectories (MvG-SINDyC)	1.220 / 0.045	1.209 / 0.029
12 trajectories (E-SINDyC)	0.995 / 0.011	1.114 / 0.029

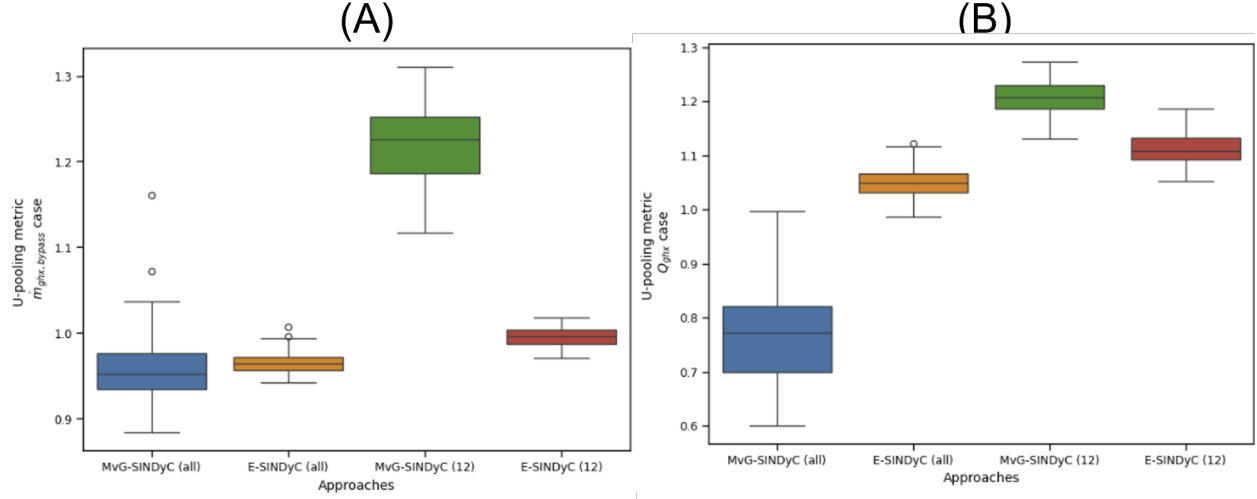


Figure 9. Distribution of the U-pooling metric for the models' predictions across 100 realizations for (A) $\dot{m}_{glx,bypass}$ and (B) Q_{glx} .

4.4. Application to Additional Use Cases: Charging of TEDS

In this section, we focus on the charging portion of our transient—in particular for the TES, as the GHX is closed during this period of time. The charge starts at 2,220 s and ends at 8,080 s. Because the physic is not the same (the TES is charging, hence the flow is reversed in comparison to before), we anticipate that the coefficients needed in the resulting SINDyC model will differ from those pertaining to the discharge. For the sake of fairness of comparison, we will use the same hyperparameters as before. The actuators (which ranges are given in Table VIII) and resulting trajectories are given in Figure 10.

Table VIII. Ranges for the actuators to generate trajectories in the charge case.

actuators	lower bound	upper bound
PV_{pv006} (-)	0	1
$\dot{m}_{pump,out}$	0.62	1.26
$T_{heater,out}$ (K)	303	573
$T_{pump,in}$ (K)	303	574
time ramp (s)	100	150

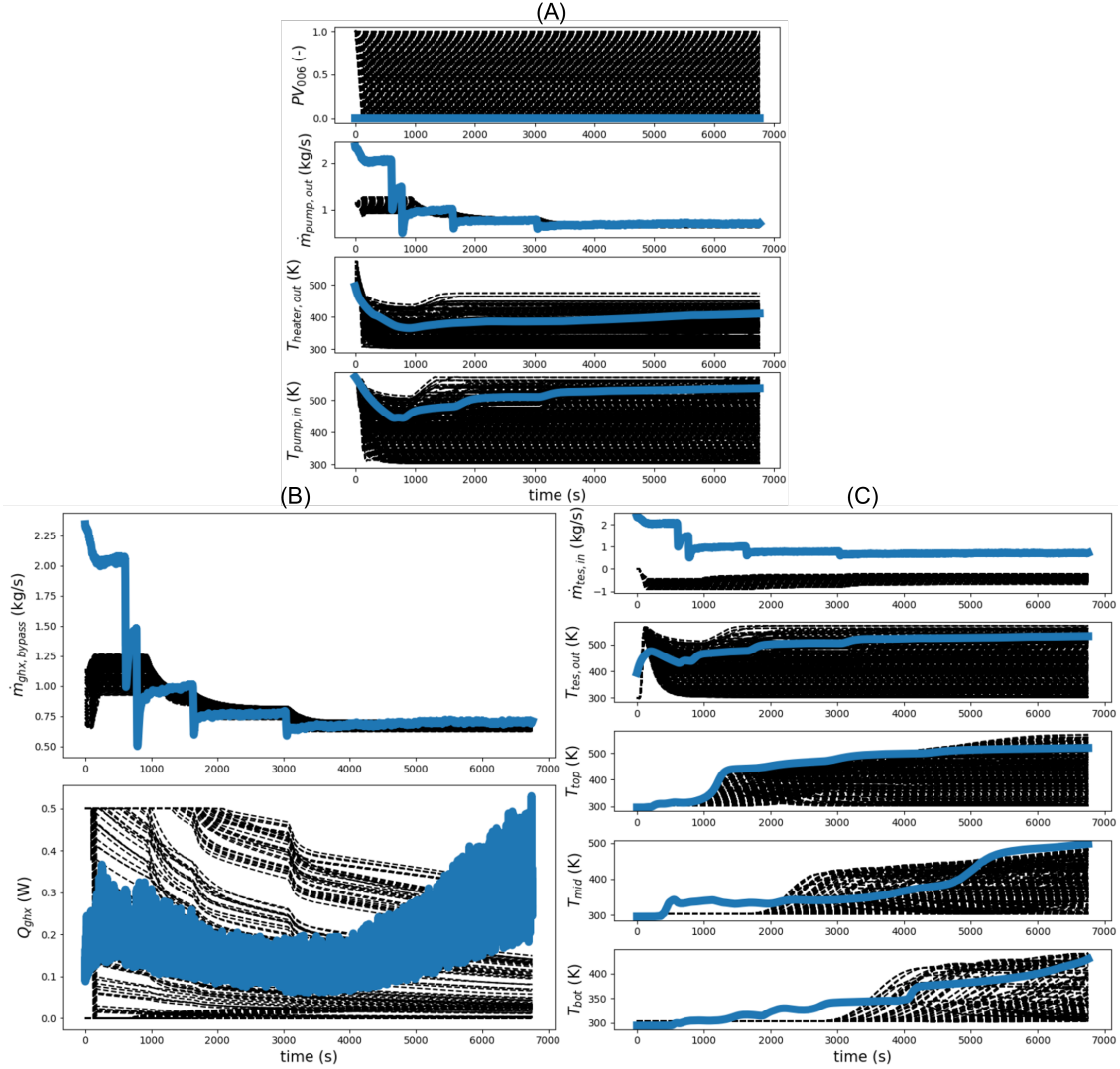


Figure 10. (A) Actuators and trajectories for the (B) GHX and (C) TES discharge.

The correlation plots for the GHX are compared in Figure 13 in Appendix A. We see that, for the most part, the coefficient distributions are very different. In particular, the coefficients for Q_{ghx} (i.e., $(A_{2i})_{i \in \{0,6\}}$) vary far more in the discharge case, as represented by the wider spread of the orange dots in the plots. This demonstrates that we cannot expect a similar model to cover similar transients, even on the same system. In this instance, in first-principles models such as computational fluid dynamics models, closure relations changed depending on the regime (e.g., bubbly flow versus annular flow). The SINDyC model affords us a principled approach to readily characterize these changes without being constrained by the physics, but to do so via a purely data-driven methodology that simplifies the tedious process. The predictive confidence intervals are given in Figures 11 (A) and (B) and Figures 11 (C) through (G) for the GHX and TES scenarios, respectively. Apart from Q_{ghx} , the linear MvG-SINDyC model captures the experiment within the confidence interval really well. The mean (red curves in the figures) also follow the experiment reasonably well. We hypothesize that the error in the Q_{ghx} case is due to the Modelica data poorly reproducing the charge, as exemplified by Figure 10 (B). We recommend a thorough study of the Modelica

model, specifically in this region of the trajectories space. Such a study is currently being performed in parallel with this work. Lastly, for brevity's sake, we omit the study of the full trajectory, but the difference in the coefficients when comparing the two “phenomena” essentially reinforces the intuition that it is simply impossible to deploy a single compelling ROM across the full trajectory.

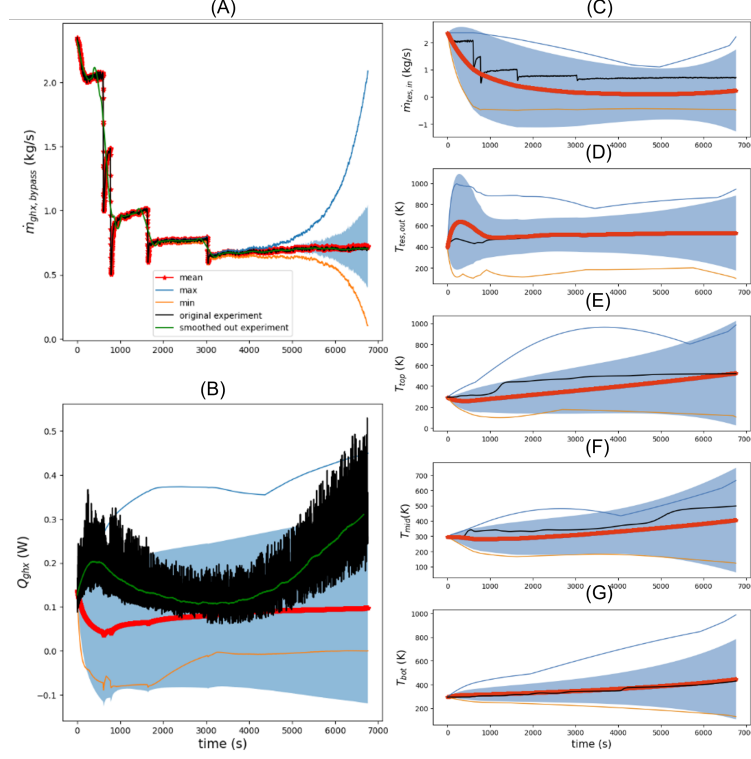


Figure 11. Probabilistic predictive intervals given by the SINDyC model in the discharge case for the GHX ((A) and (B)) and TES ((C) to (G)) scenarios.

4.5. Summary of the Studies Conducted

In Section 4.2 and Appendix B, we explain our approach to obtaining SINDyC models by using different trajectories and hyperparameters for the STLSQ algorithm. We found it was better not to encourage sparsity for the GHX scenario, but to encourage it for the TES. We hypothesize that this is due to the higher model discrepancy between Modelica and the experiment in the TES scenario. We also showed that the more trajectories are utilized to train SINDyC models, the smoother the prediction. This is attributed to the training approach using second-order regression. This was more evident for the GHX, where local gradients during discharge can be high. As a result, we found there should be a trade-off in the number of trajectories to capture the intricacies of the experimental trajectories (e.g., high gradients) while also mitigating the model discrepancies between the physical experiment and Modelica. Per our estimates, 12 trajectories for training achieved the proper trade-off in both instances.

In Section 2.2 and Appendix C, we applied the approaches described in Algorithm 2, namely MvG-SINDyC and E-SINDyC. We also applied these approaches with samples of only 12 trajectories to demonstrate the superiority of utilizing multiple sets of various numbers of trajectories. We then randomly sampled 500 models (for E-SINDyC) and 500 combinations of coefficients (for MvG-SINDyC) to obtain predictive

confidence intervals for the experiment. Subsequently, we repeated this to obtain statistics (i.e., the mean and standard deviation) across 100 realizations and applied FTs and post-hoc NTs to ensure the statistical superiority of one approach over the others. We found MvG-SINDyC with multiple sets of trajectories to be statistically superior, consistently outperforming E-SINDyC in terms of capturing the experimental trajectories within the 95% predictive confidence intervals (see Tables VI and XI for the GHX and TES, respectively) for all but $\hat{m}_{tes,in}$. In terms of the U-pooling metric, MvG-SINDyC surpassed E-SINDyC six out of seven times (for all but T_{top} , as showcased in Table XII), while being statistically similar to $\hat{m}_{ghx,bypass}$ with E-SINDyC. Apart from these two instances, E-SINDyC performed significantly worse than MvG-SINDyC trained with sets of different numbers of trajectories, thus demonstrating the superiority of our approach. Table IX summarizes their performance.

Table IX. Summary of the performance of each approach for UQ with SINDyC models. Only MvG-SINDyC trained with various numbers of trajectories achieved excellent performance on both the GHX and TES scenarios.

	multivariate fit	E-SINDyC	best number of trajectory fit (here 12)	best number of trajectory ensemble (here 12)
GHX	Excellent*	poor	poor	poor
TES	Excellent	good	good	poor

* There are three levels of performance: (1) excellent, if the predictive interval and U-pooling is consistently good across every QoI; (2) good, if it at least once surpasses the other methods, even if it is inconsistent; and (3) poor, if otherwise.

We then extended the application of MvG-SINDyC to the charge portion of the trajectories in Section 4.4. We found that, with different SINDyC coefficients, good predictive confidence intervals could be obtained for all but Q_{ghx} . However, we noticed that the discrepancy between Modelica and the experiment was too large, thus excessively biasing our model. This will motivate future Modelica model development to obtain legitimate intervals.

5. CONCLUSION

IESs are complex heterogeneous architectures that aim to coordinate different assets to improve the overall economy in comparison to using each asset individually. They are particularly interesting for nuclear developers trying to improve their economic value proposition by expanding into unique markets and not just selling electricity on the grid. Supervisory control methodologies must be deployed to efficiently orchestrate the interactions among the subsystems and dynamically adjust sub-system's setpoints to meet external demands. However, due to a lack of high-quality experimental data and the strict requirements on computational speed, the development and deployment of such methodologies relies on surrogate models that are trained primarily by simulation data from physics-based models with uncertainties. In the present work, we develop an approach using probabilistic ensemble to building robust, interpretable, fast-to-evaluate models so as to enable development and eventual deployment of efficient and reliable control algorithms for IES applications.

First, we generated trajectories by using a physics code (i.e., Modelica) that encapsulated the physical experiment while also considering the limited adequacy of the physics code (see Section 3.2). We then used an advanced surrogate model identification method, named SINDyC, to build rapidly computing models (see Section 4.2 and Appendix B for the GHX and TES, respectively). We aggregated these models to estimate the true values with uncertainty that the coefficients must be equal to in order to better approximate the experiment. We then fit a multivariate Gaussian to obtain one probabilistic SINDyC model (see Section

2.2 and Appendix C for the TES and GHX, respectively) and demonstrated its superiority at capturing the experiment, within its predictive confidence bounds, in comparison to a classical approach (i.e., E-SINDyC [30]). Our approach was coined “MvG-SINDyC.”

We found that the characteristics of the trajectories, as well as the number of trajectories generated in the simulation used to train the SINDyC models, were pivotal for correctly reproducing the experiment, within reasonable confidence bounds. Moreover, higher discrepancies between the experiment and Modelica warranted a higher degree of sparsity in SINDyC. More trajectories resulted in smoother predictions with a loss of information, especially in high-gradient regions. On the other hand, an insufficient number of trajectories could cause certain physics to be overlooked, or perhaps even be biased toward the model inadequacy of the physics code. We optimized the selection of trajectories using different sets of trajectories such that our approach can encapsulate the experimental trajectories with a desired balance of smoothness and resolutions. Note that the choice of these sets requires several iterations of tuning. As a result, our approach was able to encapsulate the experimental trajectories over 90% of the time for five different QoIs, 84% for $\dot{m}_{tes,in}$, and 39% for T_{mid} on average, which is consistently higher than all the other approaches except for $\dot{m}_{tes,in}$ (see Tables VI and XI for the GHX and TES, respectively). While the confidence interval is the most important metric when it comes to robust control, MvG-SINDyC also surpassed the E-SINDyC approaches on the U-pooling metric six out of seven times (see Tables VII and XII), while being statistically similar to $\dot{m}_{ghx,bypass}$ with E-SINDyC, even if our approach obtained a better score when averaged over 100 realizations. To our knowledge, this is the first time that (1) a continuous probability distribution has been fit directly onto the SINDyC coefficient distribution accounting for correlations among coefficients, and that (2) a probabilistic model has been built with optimal combinations of trajectories with respect to smoothness and resolutions.

In Section 4.4, we extended our approach to the charging period of the trajectories, demonstrating a seamless extension for all but Q_{ghx} . However, the poor quality of the data coming from Modelica was deemed the reason behind this drop in performance. Ongoing research is focusing on including more experimental data and deriving better Modelica models—which could also be informed by our uncertainty estimates [30, 44]—to confirm the validity of our approach.

Additionally, the UQ intervals (see Figures 7 and 17) could benefit from further refinement, but will be limited by the SINDyC models’ inadequacies and the experimental noise. Inverse UQ (also known as Bayesian calibration) is a classical approach to estimating the latter two while also providing better parametric uncertainty for the model coefficients [41, 44], as well as to guiding future model development [43] and is currently being explored.

Lastly, it is sometimes useful to prune several coefficients whose inclusion probabilities are small (see Algorithm 2); however, we decided not to do this, due to the high correlations of these coefficients with other non-zero ones. A potential route would be to study the performance achieved after pruning them. This would affect the TES model more, as 19 of the 50 coefficients have an inclusion probability of 50% or less (compared to only 2 out of 14 for the GHX). A study of the effect of pruning coefficients could be worthwhile, but the performance we obtained was already sufficient, so we leave it for future endeavors.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Paul Seurin: Conceptualization, Methodology, Software, formal analysis & investigation, Writing— original draft preparation, Visualization. **Linyu Lin:** Conceptualization, Investigation, Writing— review & editing, Supervision, Funding acquisition. All authors have read and agreed to the published version of the manuscript.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGEMENTS

Work supported through the INL Laboratory Directed Research& Development (LDRD) Program under DOE Idaho Operations Office Contract DE-AC07-05ID14517. We would like to thank Dr. Junyung Kim and Dr. Daniel Mikkelsen for developing the Modelica models. We would also like to thank Dr. Mohammad Abdo and John Shaver for proof-reading the manuscript and John Merickel for the discussions about using appropriate metrics for comparing stochastic time-series.

APPENDIX A. CORRELATION PLOTS BETWEEN THE SINDYC MODELS' PARAMETERS

Figure 12 showcases the correlation plots between the model's parameters of MvG-SINDyC for the GHX. While some of the coefficients such as A_{14} , A_{15} , and A_{20} are not Gaussian, we have still decided to fit a multivariate gaussian distribution to the latter. A strict multivariate gaussian, by definition, should have every marginal be Gaussian. However, the objective of this paper is to build a compact, correlated ensemble that faithfully captures the bulk of the simulation-induced uncertainty that is readily expandable to downstream model predictive control, optimizer, or broader UQ workflows.

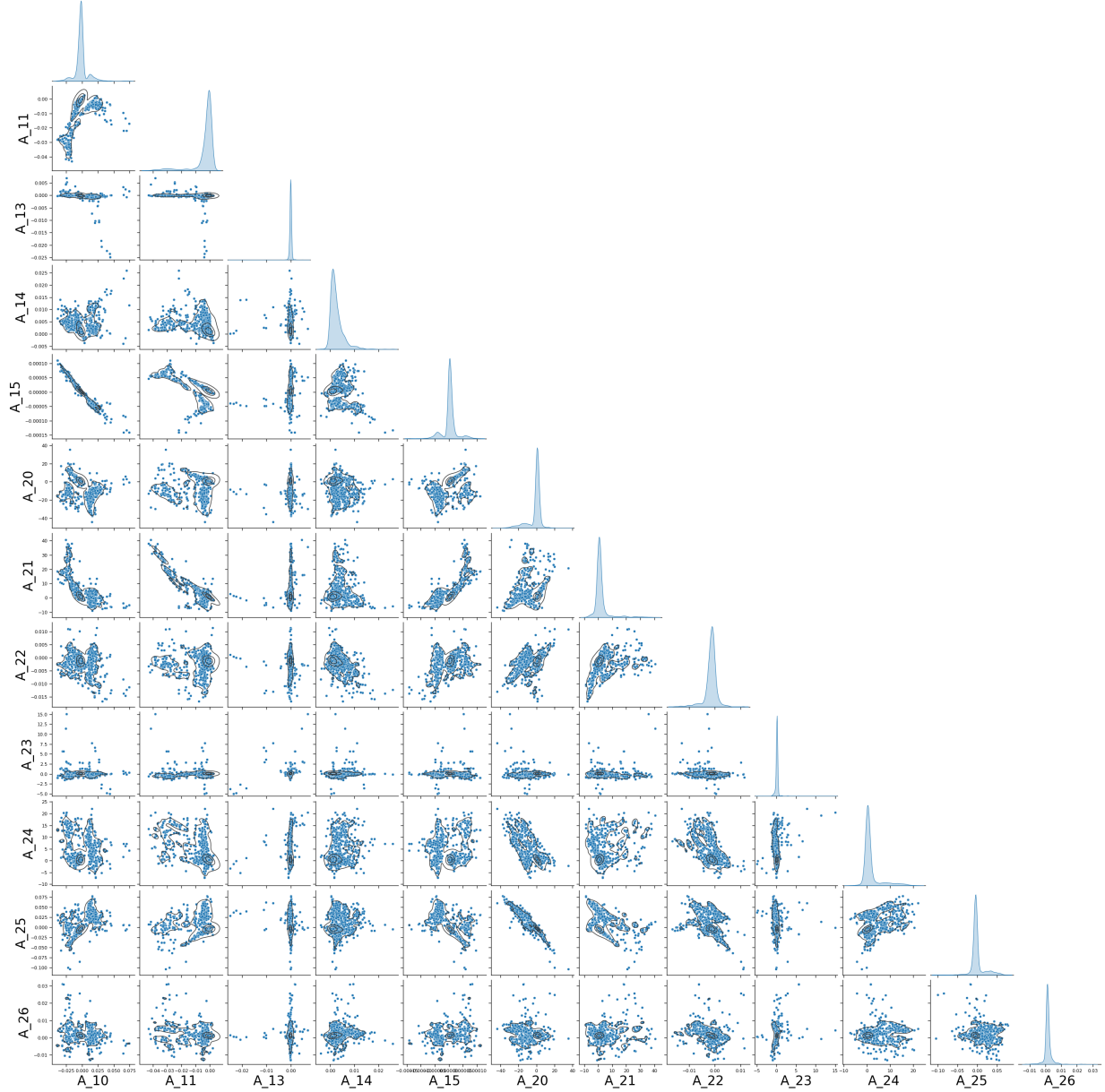


Figure 12. Correlation plots between the coefficients utilized for building the MvG-SINDyC model.

The correlation plots for the model's parameters used to obtain the MvG-SINDyC of the GHX for the charge and the discharge are given in Figure 13. Here, we see the large discrepancies between the parameter's distributions, meaning that different dynamics are excited depending on the scenario. This impels us to build different models depending on the regime studied even with the same component.

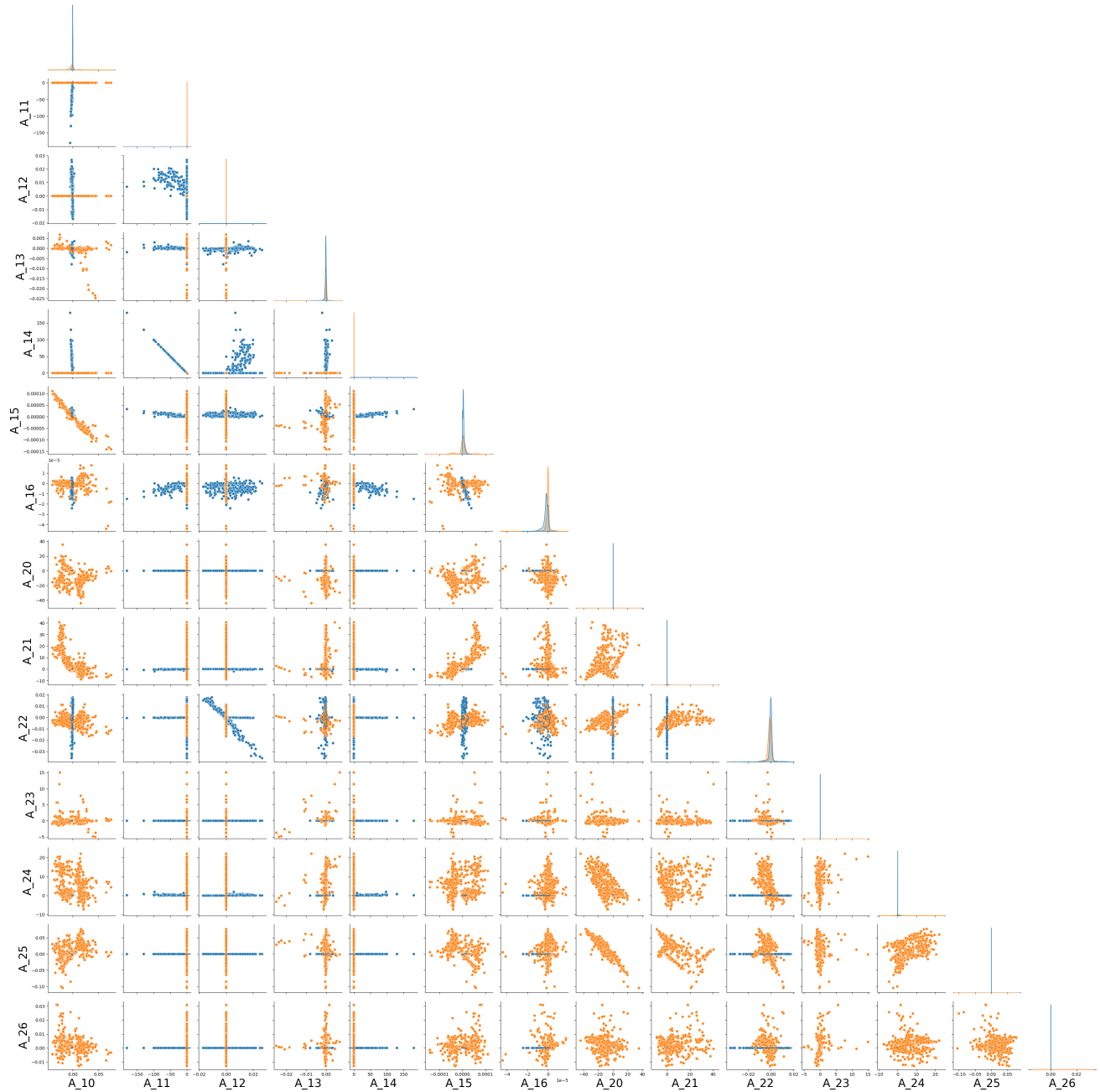


Figure 13. Correlation plots between the coefficients utilized for building the MvG-SINDyC model in the GHX scenario for both the charge (blue dots) and the discharge (orange dots).

APPENDIX B. SINDYC MODELS PROCUREMENT AND PERFORMANCES (TES)

This section follows Section 4.2 in the main text, where the model’s performance for the GHX were presented. The hyperparameters for the STLSQ are showcased in Table III in the latter section. Figures 14 (A) through (C) and Figures 15 (A) through (C) showcase each model’s predictive performance on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for $\dot{m}_{tes,in}$. Figures 14 (D) through (F) and Figures 15 (D) through (F) showcase each model’s predictive performance on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for $T_{tes,out}$. Figures 14 (G) through (I) and Figures 15 (G) through (I) showcase each model’s predictive performance on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for T_{top} . Figures 14 (J) through (L) and Figures 15 (J) through (L) showcase each model’s predictive performance on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for T_{mid} . Figures 14 (M) through (O) and Figures 15 (M) through (O) showcase each model’s predictive performance on the experiment, for models trained with 1, 4, 8, 12, 16, and 20 trajectories, respectively, for T_{bot} . The blue-shaded areas represent the $\mu \pm 2 \times \sigma$ prediction of the resulting models, the red lines represents the mean of the outputs, the blue and orange curves are the max and min of all the predictions, respectively, the dark lines are the (noisy) experimental trajectories, and the light green lines are the corresponding de-noised experimental trajectories filtered with a Savitzky-Golay filter. As in the cases involving the GHX, the more trajectories that are included, the smoother the prediction, whereas using only one trajectory reproduces the Modelica’s inadequacy. On the other hand, for models trained on a higher number of trajectories, the regions around 4,700 s and 3,700 s for T_{mid} and T_{bot} , respectively, are missed. As a result, while the performance in many cases were equivalent as showcased in Table X. It should be noted however that the case for 1 trajectory is particular because only few trajectories, the ones closer in shape compared to the experimental trajectories converged. This resulted into a very competitive metrics in every case, but it is biased towards the fact that Modelica trajectories further away from the experimental ones were therefore not conserved. Moreover, we do not want to overfit to the experimental data. Hence, the case with 1 trajectory cannot be utilized standalone for comparison. For these reasons and consistency with the GHX scenario, we decided to pursue with 12 trajectories.

Number of trajectories for training	$\dot{m}_{tes,in}$	$T_{tes,out}$	T_{top}	T_{mid}	T_{bot}
1	0.828 / 0.951	0.988 / 0.771	1.000 / 2.128	0.733 / 0.893	0.866 / 0.785
4	0.873 / 1.247	0.998 / 1.030	1.000 / 3.149	0.386 / 1.014	0.813 / 0.725
8	0.868 / 1.297	0.998 / 1.660	1.000 / 3.954	0.343 / 1.081	0.833 / 0.763
12	0.882 / 1.259	0.998 / 2.482	1.000 / 3.883	0.326 / 1.113	0.686 / 0.779
16	0.799 / 1.324	0.998 / 3.186	1.000 / 3.966	0.308 / 1.144	0.449 / 1.032
20	0.782 / 1.354	0.998 / 2.804	1.000 / 3.897	0.301 / 1.157	0.520 / 0.978

Table X. Percentage of the experimental trajectories residing within the 95% confidence intervals of the model’s prediction / U-pooling for models trained with different number of trajectories in the GHX scenario. Highlighted in red is the selected case for further comparison.

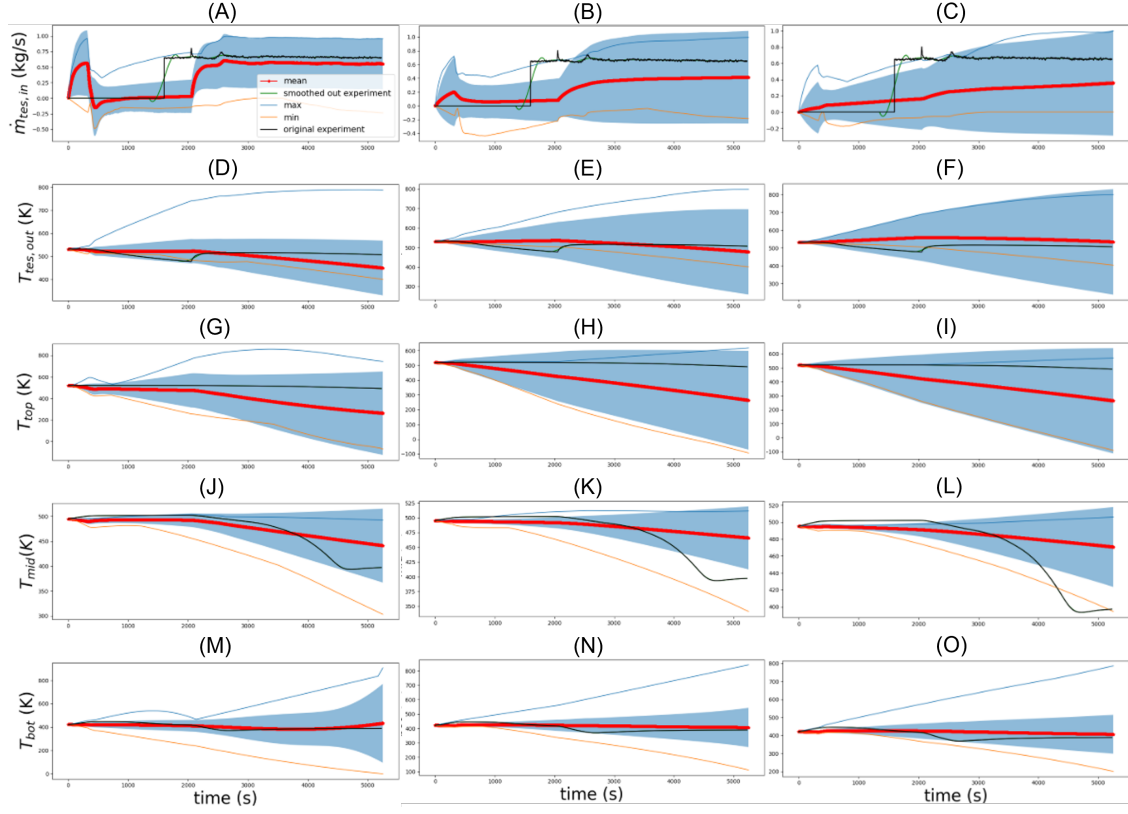


Figure 14. Prediction of the experiment with the SINDyC models: $\dot{m}_{tes,in}$ with (A) 1 trajectory, (B) 4 trajectories, and (C) 8 trajectories; $T_{tes,out}$ with (D) 1 trajectory, (E) 4 trajectories, and (F) 8 trajectories; T_{top} with (G) 1 trajectory, (H) 4 trajectories, and (I) 8 trajectories; T_{mid} with (J) 1 trajectory, (K) 4 trajectories, and (L) 8 trajectories; and T_{bot} with (M) 1 trajectory, (N) 4 trajectories, and (O) 8 trajectories.

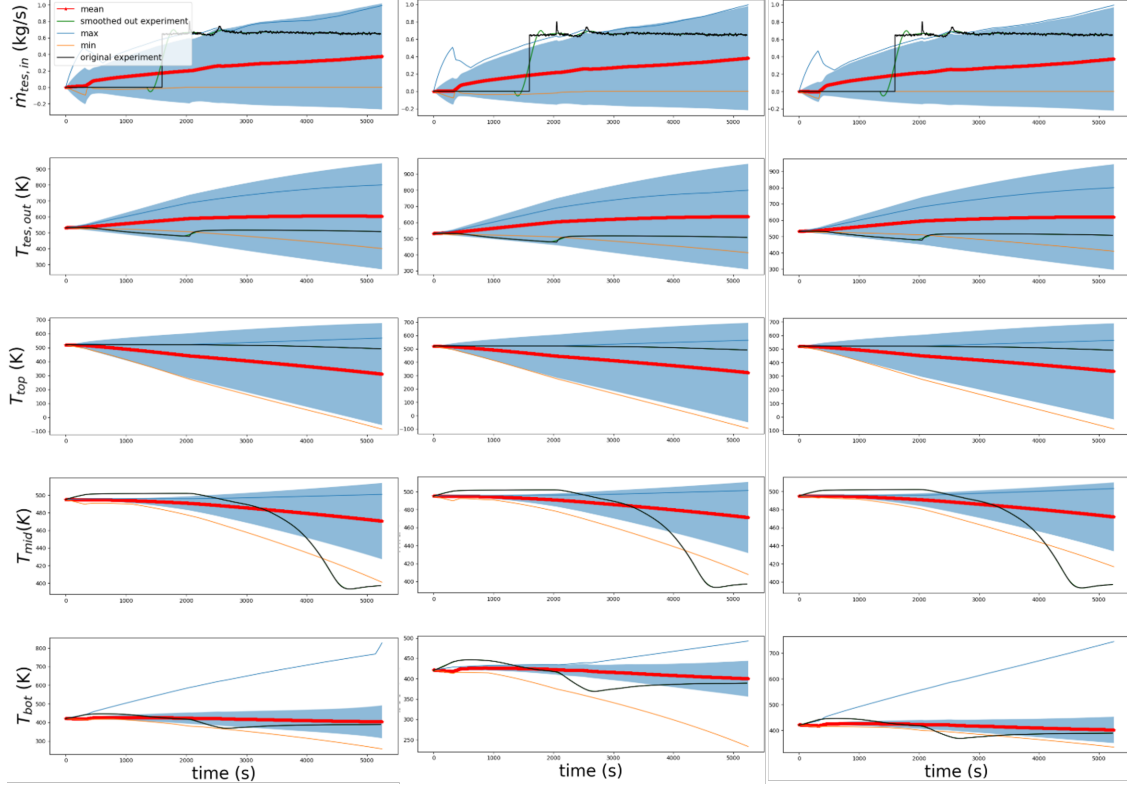


Figure 15. Prediction of the SINDyC models starting from experimental initial conditions; $\dot{m}_{tes,in}$ with (A) 12 trajectories, (B) 16 trajectories, and (C) 20 trajectories; $T_{tes,out}$ with (D) 12 trajectories, (E) 16 trajectories, and (F) 20 trajectories; T_{top} with (G) 12 trajectories, (H) 16 trajectories, and (I) 20 trajectories; T_{mid} with (J) 12 trajectories, (K) 16 trajectories, and (L) 20 trajectories; and T_{bot} with (M) 12 trajectories, (N) 16 trajectories, and (O) 20 trajectories.

APPENDIX C. UNCERTAINTY QUANTIFICATION FOR SINDYC MODELS (TES)

This section follows Section 4.3 in the main text, where we applied two methodologies for UQ with SINDyC models: E-SINDyC [30], following Algorithm 2, and our MvG-SINDyC. Figure 16 provides the inclusion probability of each coefficient for the TES model and the associated correlation matrix. For improved readability, we removed those coefficients whose inclusion probabilities were less than 50%. Here, many coefficients have an inclusion probability of 0. This is the result of our choice of sparsity, with ϵ being equal to $1e-2$. Indeed, compared to the GHX, we found the TES model to be more sensitive when predicting experimental trajectories. We hypothesize that this is caused by the larger number of coefficients and the greater Modelica model inadequacy, specifically for T_{top} and T_{mid} (see Section 3.2). As with the GHX, we also observe a large correlation between the coefficients, as evidenced by large positive and negative values in the correlation matrix in Figure 16.

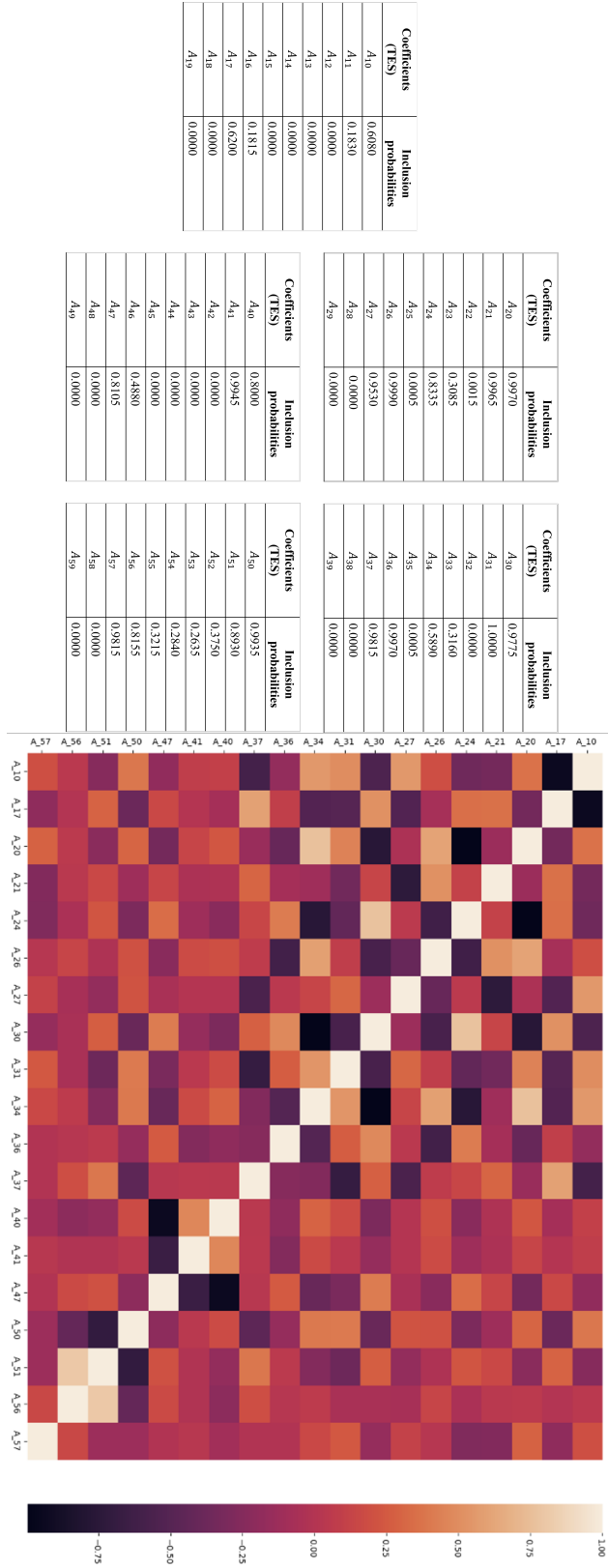


Figure 16. Inclusion probabilities for the SINDyC model in the TES case, and the associated correlation matrix (for non-thresholded coefficients).

Figure 17 showcases the predictive confidence intervals for each approach: Figures 17 (A), (E), (I), (M), and (Q) for MvG-SINDyC; Figures 17 (B), (F), (J), (N), and (R) for E-SINDyC; Figures 17 (C), (G), (K), (O), and (S) for using only 12 trajectories for MvG-SINDyC; and Figures 17 (D), (H), (L), (P), and (T) for using only 12 trajectories for E-SINDyC, for each QoI. This is followed by Figures 17 (A) through (D) for $\dot{m}_{tes,in}$, Figures 17 (E) through (H) for $T_{tes,out}$, Figures 17 (I) through (L) for T_{top} , Figures 17 (M) through (P) for T_{mid} , and Figures 17 (Q) through (T) for T_{bot} . The light blue line represents the $\mu \pm 3 \times \sigma$ prediction of the resulting models. Our approach surpasses the ensemble one by proposing confidence intervals that better encapsulate the experimental trajectories, apart from $\dot{m}_{tes,in}$, where MvG-SINDyC underpredicts the experiment at between 1,500 to 2,500 s. Table XI and Figure 18 confirms these observations, particularly in the case of T_{top} , where nearly 100% of the experimental trajectories are captured by the 95% confidence intervals of prediction within our approach, as compared against only 67% when using E-SINDyC. Moreover, Table XI also shows once again that using multiple numbers of trajectories for both MvG-SINDyC and E-SINDyC is superior to using only one.

Statistical tests could then be utilized to ensure that these observations were statistically significant, as proposed in Section 2.2. The p values of the FTs for the five QoIs was equal to 0, meaning that at least one of the approaches statistically differed from the other three. The p values of the NTs for MvG-SINDyC, when compared against the other E-SINDyC-based approaches, was equal to 0 for all. For all but $T_{tes,out}$ and T_{top} , MvG-SINDyC trained with multiple sets of trajectories proved statistically superior to its counterpart. Considering that MvG-SINDyC achieves the highest score, on average, four out of five times, we can conclude this approach to be statistically superior.

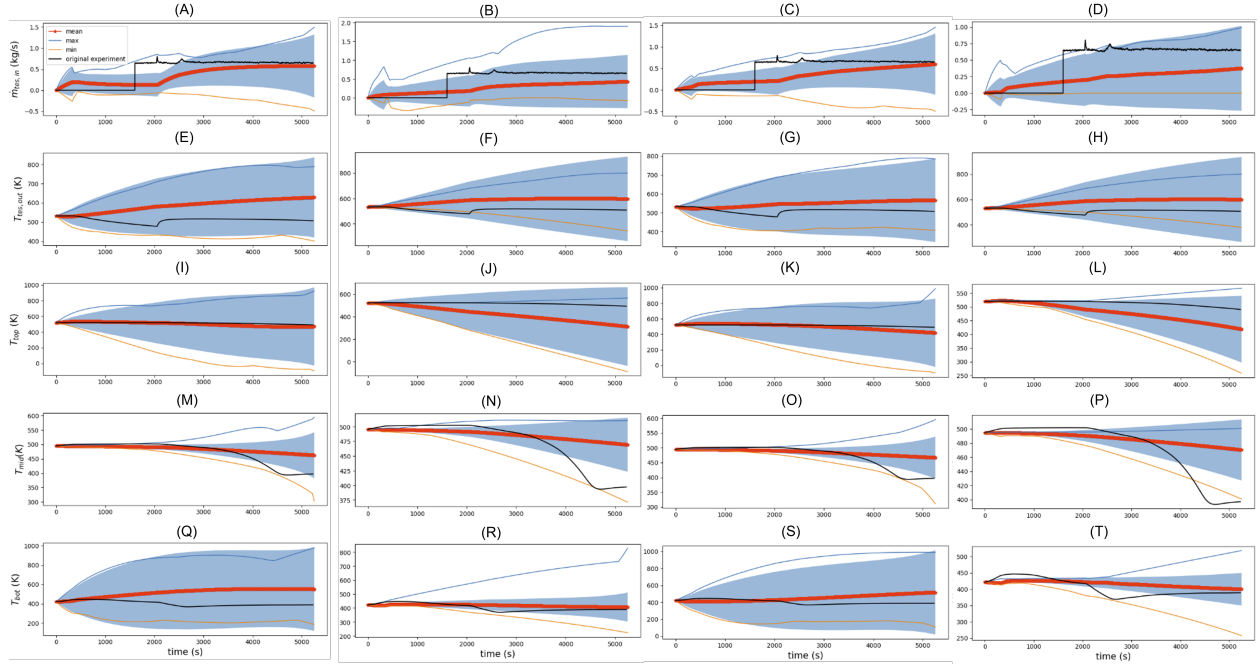


Figure 17. Comparison of the UQ-SINDyC approaches: Figures (A), (E), (I), (M), and (Q) for MvG-SINDyC with different numbers of trajectories; Figures (B), (F), (J), (N), and (R) for E-SINDyC with different numbers of trajectories; Figures (C), (G), (K), (O), and (S) for MvG-SINDyC with 12 trajectories; and Figures (D), (H), (L), (P), and (T) for E-SINDyC with 12 trajectories.

Table XI. Percentage of the (filtered) experimental trajectories within the 95% confidence intervals of the models' predictions (i.e., between 2.5% and 97.5% of the mean). The higher the better. Highlighted in red is the best approach.

Method	$\dot{m}_{tes,in}$	$T_{tes,out}$	T_{top}	T_{mid}	T_{bot}
MvG-SINDyC	0.844 / 0.013	0.999 / 0.000	1.000 / 0.000	0.391 / 0.019	0.999 / 0.000
E-SINDyC	0.876 / 0.005	0.998 / 0.000	1.000 / 0.000	0.341 / 0.007	0.674 / 0.086
12 trajectories (MvG-SINDyC)	0.799 / 0.020	0.999 / 0.000	1.000 / 0.000	0.335 / 0.012	0.999 / 0.000
12 trajectories (E-SINDyC)	0.876 / 0.018	0.998 / 0.000	1.000 / 0.000	0.328 / 0.005	0.665 / 0.076

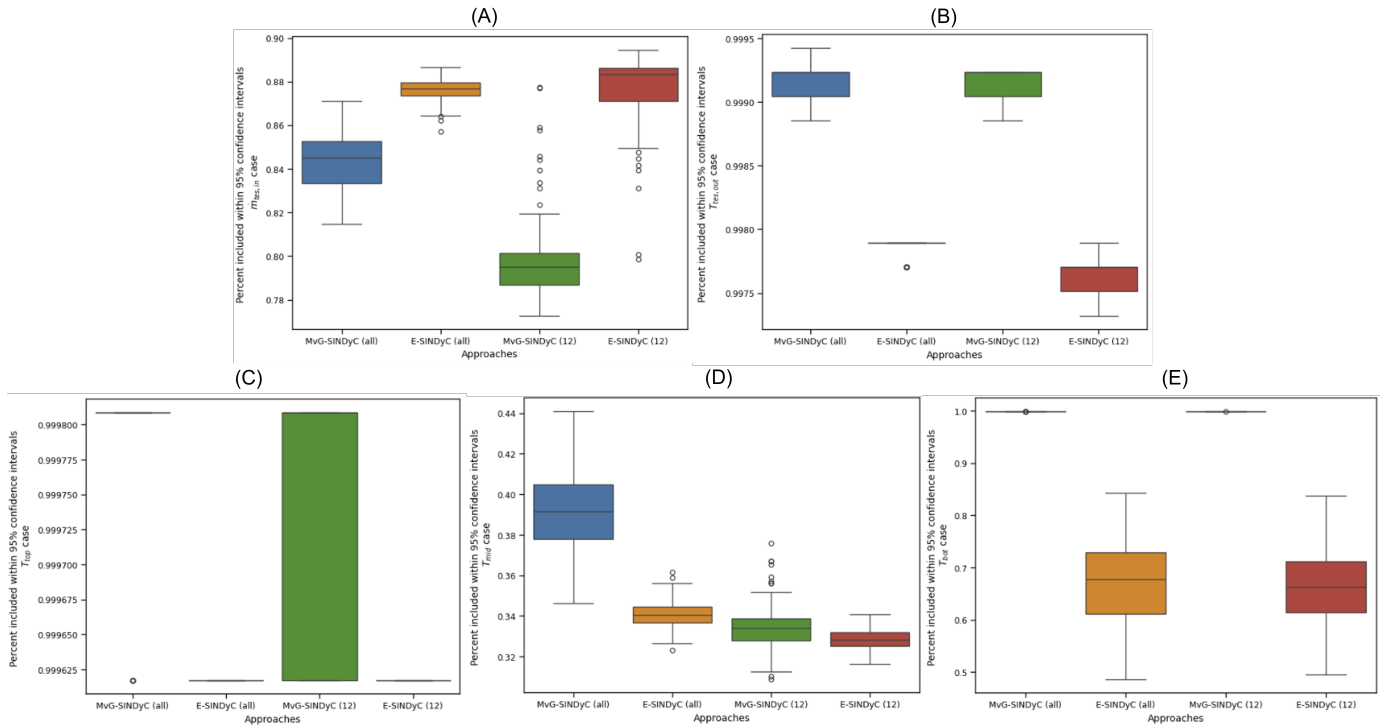


Figure 18. Distribution of the percentage of the experimental trajectories within the 95% confidence intervals of the models' predictions across 100 realization for (A) $\dot{m}_{tes,in}$ and (B) $T_{tes,out}$, (C) T_{top} , (D) T_{mid} , and (E) T_{bot} .

Similarly, Table XII and Figure 19 provides the performance of the approaches related to the U-pooling metric. For this metric, the MvG-SINDyC-based approaches surpassed the E-SINDyC ones four out of five times. On top of that, when the MvG-SINDyC with 12 trajectories proved better than the one trained with many trajectories (cases $T_{tes,out}$ and T_{top}), the post-hoc pairwise NT tests failed (with p-values of 0.1569 and 0.098, respectively) meaning that both MvG approaches were statistically equivalent. Against E-SINDyC, the NT tests are always significant. For these reasons also, we can consider MvG-SINDyC trained with different sets of trajectories to be superior.

Table XII. U-pooling metric for each approach. The lower the better. Highlighted in red is the best approach.

Method	$\dot{m}_{tes,in}$	$T_{tes,out}$	T_{top}	T_{mid}	T_{bot}
MvG-SINDyC	0.978 / 0.044	1.817 / 0.182	2.393 / 0.292	1.171 / 0.032	1.469 / 0.374
E-SINDyC	1.569 / 0.044	2.184 / 0.184	4.048 / 0.282	1.225 / 0.015	0.919 / 0.067
12 trajectories (MvG-SINDyC)	1.070 / 0.027	1.722 / 0.088	2.174 / 0.188	1.243 / 0.017	1.308 / 0.218
12 trajectories (E-SINDyC)	1.526 / 0.048	2.721 / 0.283	3.911 / 0.592	1.250 / 0.010	0.919 / 0.058

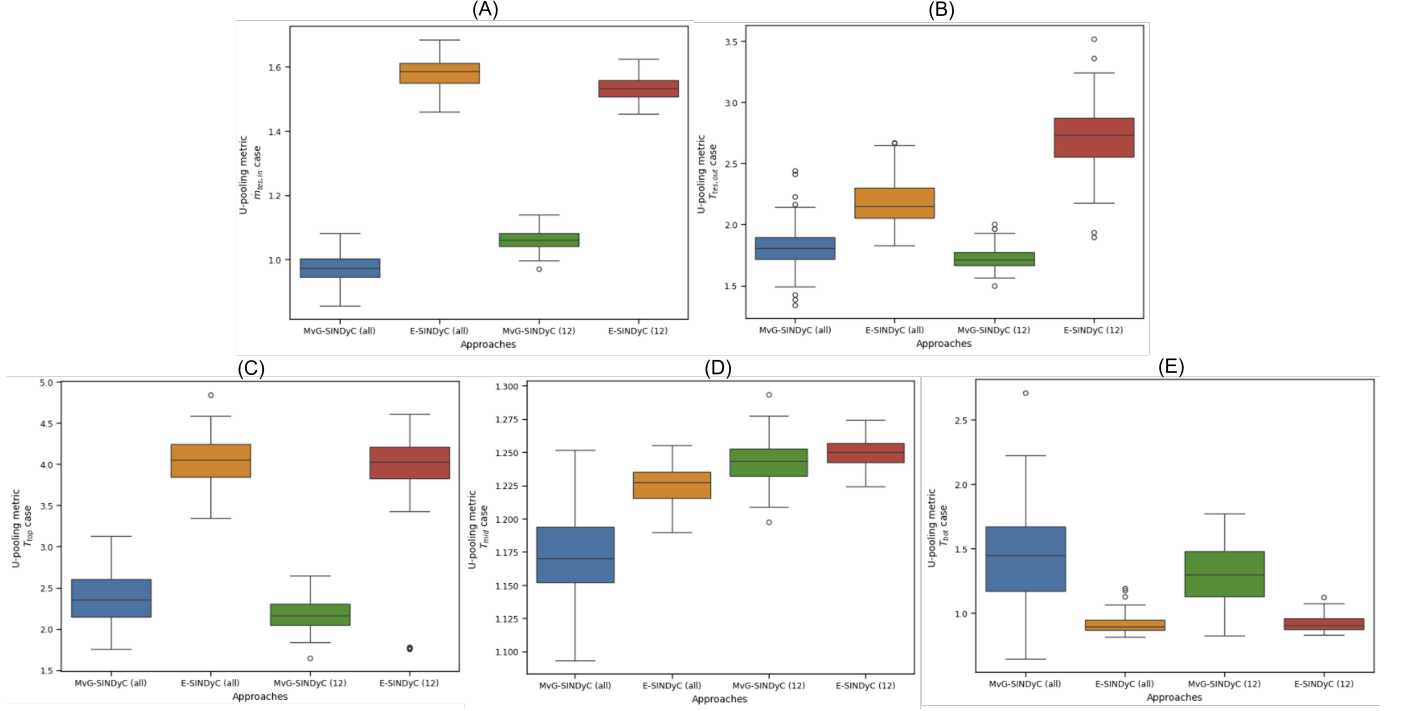


Figure 19. Distribution of the U-pooling metric for the models' predictions across 100 realizations for (A) $\dot{m}_{tes,in}$ and (B) $T_{tes,out}$, (C) T_{top} , (D) T_{mid} , and (E) T_{bot} .

REFERENCES

- [1] P. Seurin, O. Olabanjo, J. Wiggins, L. Pratt, L. Rana, R. Yasaei, and G. Renard. “Chapter 9 - Natural Language Processing for Earth resource management: a case of H2 Golden Retriever research.” *Data Analytics and Artificial Intelligence for Earth Resource Management*, pp. 157–183 (2025). URL <https://doi.org/10.1016/B978-0-443-23595-5.00009-7>.
- [2] L. Lin. “Development of Supervisory Control System for Thermal Energy Distribution System.” In *2024 Pacific Basin Nuclear Conference, PBNC 2024*, pp. 435–444. American Nuclear Society (2024).
- [3] K. L. Frick, S. M. Bragg-Sitton, and C. Rabiti. “Development of the INL Thermal Energy Distribution System (TEDS) in the Modelica Eco-System for Validation and Verification (INL/EXT-20-59195-Rev000).” Technical report, Idaho National Laboratory (2020).
- [4] C. Kral. *Modelica - object oriented modeling of polyphase electric machines (in German)*. München: Hanser Verlag (2018). URL DOI:10.3139/9783446457331.
- [5] M. Gräber, C. Kirches, D. Scharff, and W. Tegethoff. “Using functional mock-up units for nonlinear model predictive control.” In *9th International Modelica Conference*, pp. 781–790 (2012).
- [6] R. Franke, M. Walther, N. Worschech, W. Braun, and B. Bachmann. “Model-based control with FMI and a C++ runtime for Modelica.” *Proceedings of the 11th International Modelica Conference, Versailles, France, September 21-23* (2015).
- [7] M. Ziegler, A. F. Posada-Moreno, F. Solowjow, and S. Trimpe. “On Foundation Models for Dynamical Systems from Purely Synthetic Data.” *arXiv preprint arXiv:241200395* (2024).
- [8] M. Herde, B. Raonic, T. Rohner, R. Käppeli, R. Molinaro, E. de Bézenac, and S. Mishra. “Poseidon: Efficient foundation models for pdes.” *Advances in Neural Information Processing Systems*, **volume 37**, pp. 72525–72624 (2024).
- [9] E. Negrini, Y. Liu, L. Yang, S. J. Osher, and H. Schaeffer. “A Multimodal PDE Foundation Model for Prediction and Scientific Text Descriptions.” *arXiv preprint arXiv:250206026* (2025).
- [10] J. Sun, Y. Liu, Z. Zhang, and H. Schaeffer. “Towards a foundation model for partial differential equations: Multioperator learning and extrapolation.” *Physical Review E*, **volume 111**(3), p. 035304 (2025).
- [11] P. Seurin and K. Shirvan. “Physics-informed Reinforcement Learning optimization of PWR core loading pattern.” *Annals of Nuclear Energy*, **volume 208**, p. 110763 (2024). URL <https://doi.org/10.1016/j.anucene.2024.110763>.
- [12] A. Hall and V. Agarwal. “Barriers to adopting artificial intelligence and machine learning technologies in nuclear power.” *Progress in Nuclear Energy*, **volume 175**, p. 105295 (2024 Oct). URL doi: 10.1016/j.pnucene.2024.105295.
- [13] C. Tsay, A. Kumar, J. Flores-Cerrillo, and M. Baldea. “Optimal demand response scheduling of an industrial air separation unit using data-driven dynamic models.” *Comput Chem Eng*, **volume 126**, pp. 22–34 (2019). URL <https://api.semanticscholar.org/CorpusID:127849103>.
- [14] J. L. Proctor, S. L. Brunton, and J. N. Kutz. “Generalizing Koopman Theory to Allow for Inputs and Control.” *SIAM J APPLIED DYNAMICAL SYSTEMS*, **volume 17**, p. 909–930 (2018).

- [15] P. Bevanda, S. Sosnowski, and S. Hirche. “Koopman operator dynamical models: Learning, analysis and control.” *Annual Reviews in Control*, **volume 52**, pp. 197–212 (2021). URL <https://www.sciencedirect.com/science/article/pii/S1367578821000729>.
- [16] Q. Cao, S. Goswami, and G. Karniadakis. “Laplace neural operator for solving differential equations.” *Nat Mach Intell*, **volume 6**, p. 631–640 (2024). URL <https://doi.org/10.1038/s42256-024-00844-4>.
- [17] L. Lu, J. Pengzhan, P. Guofei, Z. Zhang, and G. E. Karniadakis. “Learning nonlinear operators via deeponet based on the universal approximation theorem of operators.” *Nature Machine Intelligence*, **volume 3**, p. 218–229 (2021).
- [18] M. Filho, A. M. M. Lima, and V. H. C. Pinheiro. “Multi-gene genetic programming of critical boron concentration and power peak factor for nuclear reactor fuel reload calculations.” *Progress in Nuclear Energy*, **volume 180**, p. 105596 (2025). URL <https://doi.org/10.1016/j.pnucene.2024.105596>.
- [19] L. Zanna and T. Bolton. “Data-driven equation discovery of ocean mesoscale closures.” *Geophysical Research Letters*, **volume 47** (2020). URL <https://doi.org/10.1029/2020GL088376>.
- [20] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Machine learning of linear differential equations using gaussian processes.” *Journal of Computational Physics*, **volume 348**, p. 683–693 (2017).
- [21] V. Churchill. “Principal component flow map learning of PDEs from incomplete, limited, and noisy data.” *Journal of Computational Physics*, **volume 524**, p. 113730 (2025). URL <https://doi.org/10.1016/j.jcp.2025.113730>.
- [22] S. Brunton, J. Proctor, and J. Kutz. “Sparse identification of nonlinear dynamics with control (SINDYc).” *IFAC-PapersOnLine*, **volume 49**(18), pp. 710–715 (2016).
- [23] e. a. Rudy, S. H. “Data-driven discovery of partial differential equations.” *Sci Adv*3 (2017). URL DOI:10.1126/sciadv.1602614.
- [24] Y. S. Bang, J. Heo, H. N. Na, and K. Kang. “Reduced order models for thermal hydraulic transient analysis in nuclear engineering.” *Annals of Nuclear Energy*, **volume 211**, p. 111002 (2025).
- [25] F. Abdullah and P. D. Christofides. “Data-based modeling and control of nonlinear process systems using sparse identification: An overview of recent results.” *Computers & Chemical Engineering*, **volume 174** (2023). URL <https://doi.org/10.1016/j.compchemeng.2023.108247>.
- [26] S. Peitz, S. E. Otto, and C. W. Rowley. “Data-driven Model Predictive Control using Interpolated Koopman Generators.” *arXiv preprint arXiv: 200307094v1* (2020).
- [27] M. Korda and I. Mezic. “Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control.” *Automatica*, **volume 149**, pp. 149–160 (2018).
- [28] A. J. Dave, T. Lee, R. Ponciroli, and R. B. Vilim. “Design of a supervisory control system for autonomous operation of advanced reactors.” *Annals of Nuclear Energy*, **volume 182**, p. 109593 (2023). URL <https://doi.org/10.1016/j.anucene.2022.109593>.
- [29] J. Oncken, L. Lin, and V. Agarwal. “Adaptive Model Predictive Control for Heat Pipe–Cooled Microreactors Under Normal and Heat Pipe Failure Conditions.” *Nuclear Technology*, **volume 210**(12), p. 2274–2289 (2024). URL <https://doi.org/10.1080/00295450.2024.2342206>.

- [30] U. Fasel, J. N. Kutz, B. W. Brunton, and S. L. Brunton. “Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control.” *Proc R Soc A*, **volume 478**, p. 20210904 (2022). URL <http://doi.org/10.1098/rspa.2021.0904>.
- [31] E. Kaiser, J. N. Kutz, and S. L. Brunton. “Sparse identification of nonlinear dynamics for model predictive control in the low-data limit.” *Proceedings of the Royal Society A*, **volume 474**(2219), p. 20180335 (2018).
- [32] Y. Che, J. Yurko, P. Seurin, and K. Shirvan. “Machine learning-assisted surrogate construction for full-core fuel performance analysis.” *Annals of Nuclear Energy*, **volume 168** (2022). URL <https://doi.org/10.1016/j.anucene.2021.108905>.
- [33] F. Alves, E. P. and Fiuza. “Data-driven discovery of reduced plasma physics models from fully kinetic simulations.” *Phys Rev Research*, **volume 4**, p. 033192 (9 September, 2022). URL <https://doi.org/10.1103/PhysRevResearch.4.033192>.
- [34] D. Saibal, R. E. Jones, and H. Koll. “Accurate data-driven surrogates of dynamical systems for forward propagation of uncertainty.” *International Journal for Numerical Methods in Engineering*, **volume 125** (2024). URL [10.1002/nme.7576](https://doi.org/10.1002/nme.7576).
- [35] M. Haywood-Alexander, W. Liu, K. Bacsá, Z. Lai, and E. Chatzi. “Discussing the spectrum of physics-enhanced machine learning: a survey on structural mechanics applications.” *Data-Centric Engineering*, **volume 5** (2024). URL [doi:10.1017/dce.2024.33](https://doi.org/10.1017/dce.2024.33).
- [36] L. Lin, J. Kim, D. Mikkelsen, V. Novotny, and N. Dinh. “Uncertainty Quantification for Digital Twins in Thermal Energy Distribution System.” In *2024 Pacific Basin Nuclear Conference, PBNC 2024*, pp. 425–434. American Nuclear Society (2024).
- [37] K. A. Maupin, L. P. Swiler, and N. W. Porter. “Validation Metrics for Deterministic and Probabilistic Data.” *ASME J Verif Valid Uncert September 2018*, **volume 3**(3), p. 031002 (January 22, 2019). URL <https://doi.org/10.1115/1.4042443>.
- [38] A. A. Kaptanoglu, B. M. de Silva, U. Fasel, K. Kaheman, J. L. Callahan, C. B. Delahunt, K. Champion, J. C. Loiseau, J. N. Kutz, and S. L. Brunton. “Pysindy: A comprehensive python package for robust sparse system identification.” *arXiv preprint arXiv:211108481* (2021).
- [39] S. M. Hirsh, D. A. Barajas-Solano, and J. N. Kutz. “Sparsifying Priors for Bayesian Uncertainty Quantification in Model Discovery.” (*arXiv:210702107*) (2021). URL <http://arxiv.org/abs/2107.02107>.
- [40] R. Tibshirani. “Regression shrinkage and selection via the lasso.” *J of the Royal Statistical Society B*, p. 267–288 (1996).
- [41] X. Wu, T. Kozłowski, and H. Meidani. “Kriging-based inverse uncertainty quantification of nuclear fuel performance code BISON fission gas release model using time series measurement data.” *Reliability Engineering & System Safety*, **volume 169**, pp. 422–436 (2018). URL <https://doi.org/10.1016/j.res.2017.09.029>.
- [42] Z. Abulawi, R. Hu, P. Balaprakash, and Y. Liu. “Bayesian Optimized Deep Ensemble for Uncertainty Quantification of Deep Neural Networks: a System Safety Case Study on Sodium Fast Reactor Thermal Stratification Modeling.” In *arXiv preprint arXiv:241208776v1* (2024).

- [43] G. Pastore, L. P. Swiler, J. D. Hales, S. R. Novascone, D. M. Perez, B. W. Spencer, L. Luzzi, P. Van Uffelen, and R. L. Williamson. “Uncertainty and sensitivity analysis of fission gas behavior in engineering-scale fuel modeling.” *Journal of Nuclear Materials*, **volume 456**, pp. 398–408 (2015). URL <https://doi.org/10.1016/j.jnucmat.2014.09.077>.
- [44] S. L. N. Dhulipala, A. Toptan, Y. Che, D. Schwen, R. T. Sweet, J. D. Hales, and S. R. Novascone. “Bayesian uncertainty quantification of tristructural isotropic particle fuel silver release: Decomposing model inadequacy plus experimental noise and parametric uncertainties.” *Journal of Nuclear Materials*, **volume 588**, p. 154790 (2024). URL <https://doi.org/10.1016/j.jnucmat.2023.154790>.
- [45] A. Forootani, P. Goyal, and P. Benner. “A robust sparse identification of nonlinear dynamics approach by combining neural networks and an integral form.” *Engineering Applications of Artificial Intelligence*, **volume 149**, p. 110360 (2025). URL <https://doi.org/10.1016/j.engappai.2025.110360>.
- [46] H. Schaeffer and S. McCalla. “Sparse model selection via integral terms.” *Phys Rev E* **96**, p. 023302 (2017). URL [doi:10.1103/PhysRevE.96.023302](https://doi.org/10.1103/PhysRevE.96.023302).
- [47] P. Reinbold, D. Gurevich, and R. Grigoriev. “Using noisy or incomplete data to discover models of spatiotemporal dynamics.” *Phys Rev E*, **volume 101**, p. 010203 (2020). URL [doi:10.1103/PhysRevE.101.010203](https://doi.org/10.1103/PhysRevE.101.010203).
- [48] G. F. Machado and J. Morgan. “Sparse identification of nonlinear dynamics with side information (SINDy-SI).” *2024 American Control Conference ACC*, p. 2879–2884 (2024).
- [49] R. K. Niven, A. Mohammad-Djafari, M. Cordier, L. Abel, and M. Quade. “Bayesian identification of dynamical systems.” *In Multidisciplinary Digital Publishing Institute Proceedings*, **volume 33** (2020).
- [50] A. Joshi, P. Thakolkaran, Y. Zheng, M. Escande, M. Flaschel, L. De Lorenzis, and S. Kumar. “Bayesian-EUCLID: Discovering hyperelastic material laws with uncertainties.” *Computer Methods in Applied Mechanics and Engineering*, **volume 398**, p. 115225 (2022). URL <https://doi.org/10.1016/j.cma.2022.115225>.
- [51] P. Seurin, A. Halimi, and K. Shirvan. “Impact of including fuel performance as part of core reload optimization: Application to power uprates.” *Nuclear Engineering and Design*, **volume 433**, p. 113844 (2025). URL <https://doi.org/10.1016/j.nucengdes.2025.113844>.
- [52] K. Champion, S. Brunton, and J. Kutz. “Discovery of nonlinear multiscale systems: Sampling strategies and embeddings.” *SIAM J Appl Dyn Syst*, **volume 18**, p. 312–333 (2019).
- [53] T. Iwanaga, W. Usher, and J. Herman. “Toward SALib 2.0: Advancing the accessibility and interpretability of global sensitivity analyses.” *Socio-Environmental Systems Modelling*, **volume 4**, p. 18155 (2022). URL <https://sesmo.org/article/view/18155>.
- [54] P. Seurin and K. Shirvan. “Assessment of reinforcement learning algorithms for nuclear power plant fuel optimization.” *Appl Intell*, **volume 54**, p. 2100–2135 (2024). URL <https://doi.org/10.1007/s10489-023-05013-5>.
- [55] W. L. Oberkampf and F. B. Matthew. “Measures of agreement between computation and experiment: Validation metrics.” *Journal of Computational Physics*, **volume 217**(1), pp. 5–36 (2006). URL <https://doi.org/10.1016/j.jcp.2006.03.037>.

- [56] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.” *Nature Methods*, **volume 17**, pp. 261–272 (2020).
- [57] P. Seurin and K. Shirvan. “Surpassing Legacy Approaches to PWR Core Reload Optimization with Single-Objective Reinforcement Learning.” *Nuclear Science and Engineering*, p. 1–32 (2025). URL <https://doi.org/10.1080/00295639.2025.2488702>.