Generalization of System Identification Objective Functions Through Stochastic Hidden Markov Models for Regularization, Smoothness, and Uncertainty Quantification

by

Nicholas Galioto

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
University of Michigan
2023

Doctoral Committee:

Assistant Professor Alex Arkady Gorodetsky, Chair
Professor Dennis S. Bernstein
Assistant Professor Xun Huan
Associate Professor Necmiye Ozay

Nicholas Galioto
ngalioto@umich.edu
ORCID iD: 0000-0001-8213-2442

Dedication

To those who dedicate their lives to the pursuit of knowledge...

Acknowledgments

This work is the culmination of many years of diligent study and persevering effort that would not be possible without the support of numerous people. Notably, I would like to thank my adviser Alex Gorodetsky for his guidance, mentorship, and inspirational passion for research. My gratitude further extends to all of the Computational Autonomy lab members that I have had the pleasure of meeting and getting to know throughout the years. I would also like to express my appreciation to Dr. Harsh Sharma and Professor Boris Kramer of University of California San Diego for their collaboration in researching identification of nonseparable Hamiltonian systems. Last, but not least, I am grateful to my family and friends for their encouragement and support.

The funding for this research comes from the DARPA Physics of AI Program under the grant "Physics Inspired Learning and Learning the Order and Structure of Physics," Agreement No. HR00111890030, the DARPA Artificial Intelligence Research Associate under the grant "Artificial Intelligence Guided Multi-scale Multi-physics Framework for Discovering Complex Emergent Materials Phenomena," Agreement No. HR00111990028, and the AFOSR Computational Mathematics Program (P.M. Fariba Fahroo).

Table of Contents

Chapter

Appendices

List of Tables

List of Figures

List of Appendices

List of Abbreviations

DMD dynamic mode decomposition

DRAM delayed rejection adaptive Metropolis

ERA eigensystem realization algorithm

GLS generalized least squares

HMM hidden Markov model

IC initial condition

ID identification

KF Kalman filter

LS least squares

LTI linear time-invariant

MAP maximum *a posteriori*

MCMC Markov chain Monte Carlo

MMSE minimum mean squared error

MS multiple shooting

MSE mean squared error

NEM negative energy mode

NSSNN nonseparable symplectic neural network

ODE ordinary differential equation

PCA principal component analysis

PDE partial differential equation

pdf probability density function

QoI quantity of interest

RMSE root mean squared error

SINDy sparse identification of nonlinear dynamics

TDMD total dynamic mode decomposition

TV total variation

UKF unscented Kalman filter

Abstract

With the growing availability of computational resources, the interest in learning models of dynamical systems has grown exponentially over the years across many diverse disciplines. As a result of this growth, objective functions for model estimation have been rapidly developed independently across fields such as fluids, control, and machine learning. Theoretical justifications for these objectives, however, have lagged behind. In this dissertation, we provide a unifying theoretical framework for some of the most popular of these objectives, specifically dynamic mode decomposition (DMD), single rollout Markov parameter estimation, sparse identification of nonlinear dynamics (SINDy), and multiple shooting.

In this framework, we model a general dynamical system using a hidden Markov model and derive a marginal likelihood that can be used for estimation. The key difference between this and most existing likelihood estimators is that rather than simply modeling the estimation error in the output of the system, we additionally model the error in the dynamics through the inclusion of a process noise term. Not only does this process noise term provide the flexibility needed to generalize many existing objectives, but it also provides three significant advantages in the marginal likelihood. The first is that it generates an explicit regularization term that arises directly from the model formulation without the need for adding heuristic priors onto the parameters. Furthermore, this regularization term is over the output, rather than the parameters, of the model and is therefore applicable to any arbitrary parameterization of the dynamics. Secondly, the process noise term provides smoothing of the marginal likelihood optimization surface without having to discount the information in the data through tempering methods or abbreviated simulation lengths. Lastly, estimation of the process noise term can give a quantification of the uncertainty of the estimated model without necessarily requiring expensive Markov chain Monte Carlo (MCMC) sampling.

To evaluate this proposed marginal likelihood, we present an efficient recursive algorithm for linear-Gaussian models and an approximation to this algorithm for all remaining models. We discuss how simplifications to the approximate algorithm can be made when the noise is additive Gaussian and derive simplifications for when it is arbitrary additive/multiplicative noise. Next, we provide theoretical results proving that the considered objectives are special cases of a posterior that uses the proposed marginal likelihood. These results uncover the sets of assumptions needed to transform the negative log posterior into each of the objective functions that we consider. We then present numerical experiments that compare the (approximate) marginal likelihood to each of the considered objectives on a variety of systems. These experiments include linear, chaotic, partial differential equation, limit cycle, and Hamiltonian systems. Additionally, we include a novel comparison of Hamiltonian estimation using symplectic and non-symplectic dynamics propagators. This comparison uses uncertainty quantification both in the form of MCMC sampling and process noise covariance estimation to show that embedding the symplectic propagator into the objective delivers more precise estimates than embedding the objective with the non-symplectic propagator. Overall, the results of this dissertation demonstrate that the marginal likelihood is able to produce more accurate estimates on problems with high amounts of uncertainty in the forms of measurement noise, measurement sparsity, and model expressiveness than comparable objective functions.

CHAPTER I
Introduction

The study of dynamical systems is as old a pursuit as human civilization itself, beginning with attempts at predicting seasonal change by studying celestial bodies. In fact, lunar cycles were being noted on bones and caves as early as 25,000 years ago [63]. In ancient civilization, there were two competing approaches for explaining motion and change: (1) a geometric perspective that asserted that the cosmos moved in accordance with principles of mathematical order and (2) a mechanical perspective that sought to explain the underlying causes that lead to motion [103]. During this time, astronomy was a central area of study due to beliefs that astronomical events directly impacted human affairs and terrestial events. The geometric perspective was forefront in astronomy in ancient and medieval times, using proportions and circles to create some of the first mathematical models describing natural phenomena. In the 17th century, however, there was a critical paradigm shift in modeling the solar system caused by the publication of Newton's *Principia*, in which the discovery of the universal law of gravitation was presented. This law was the first mathematical decription of an underlying cause of motion and unified the phenomenon of gravity that had been observed on Earth with the observations of celestial bodies. Also in *Principia*, Newton published his laws of motion, creating the basis for classical mechanics as we know it today. The discovery of these laws of motion and gravitation allowed for mathematical models of mechanical systems to be derived directly from physical quantities such as force and mass. As the field of mechanics progressed over the following centuries, additional physical laws were discovered, such as the principle of least action, that created increasingly powerful modeling techniques.

As models of dynamical systems became more advanced and widespread, new uses for these models began to arise beyond only prediction. Control theory is one such example. The field is considered to have begun in 1868 with the publication of James Clark Maxwell's manuscript *On Governors*, which provided modeling and analysis of a centrifugal governor based on mechanics principles. Broadly speaking, control theory requires mathematical models of dynamical systems in order to design control laws that evoke a desired behavior from the system of interest. Another example is the qualitative theory of differential equations, which was born out of a series of memoirs published by Poincaré in 1881-1882 titled “On curves defined by differential equations.” These writings showed that many of the important dynamical properties of a system could be derived from its equations of motion without needing a closed-form solution. Poincaré’s work on dynamical systems served as the basis for study of many interesting dynamical behaviors such as chaos and bifurcations. Thus, having a model of a dynamical system at hand can allow for meaningful analysis and control of the system behavior.

Creating a model for a dynamical system of interest, however, is not always straightforward. Although the known laws of physics are able to provide simple, reliable, and interpretable equations of motion, their application requires significant knowledge on the system, such as the generalized coordinates, the full system state, or the external forces. Moreover, many systems whose dynamics are of interest are non-physical systems such as financial markets or traffic patterns that are not governed by known scientific laws. In these cases where physical laws cannot be applied, models must be estimated, either purely or partially, with data collected from the system. This process of estimating the dynamics of a system of interest using measurements is known as system identification (ID), and it has been used in a range of applications such as forecasting the weather and climate [18, 88], predicting traffic flow [61], and enabling adaptive control [57, 20].

The process of system ID begins with collecting a set of data $\mathcal{D} \subset \mathcal{Y}$ from the system of interest, where $\mathcal{Y}$ is the set containing all possible measurements. If the user has control over the experiment that generates these data, then the experimental design is sometimes considered to be part of the system ID procedure. For this dissertation, we will assume that the data have already been collected and given to us. The next step is to define a model class $\mathcal{M}$ of candidate models. To identify the best model within $\mathcal{M}$, an ordering of the models based on their quality must be induced by some objective or figure of merit. Letting $\mathbb{R}$ be the real line, an objective function $\mathcal{J} : \mathcal{Y} \times \mathcal{M} \mapsto \mathbb{R}$ assigns a value to each candidate model given a collection of data. For a given $(\mathcal{D}, \mathcal{M}, \mathcal{J})$ triple, the set of optimal models $\mathcal{M}^* \subseteq \mathcal{M}$ are those that yield the lowest objective value, i.e., $\mathcal{M}^* = \arg\min_M \mathcal{J}(\mathcal{D}, M)$. Once a model belonging to $\mathcal{M}^*$ is found, it can be used in various applications such as prediction, filtering, and control. This description of the system ID process is summarized in the flow chart in Fig. 1.1.

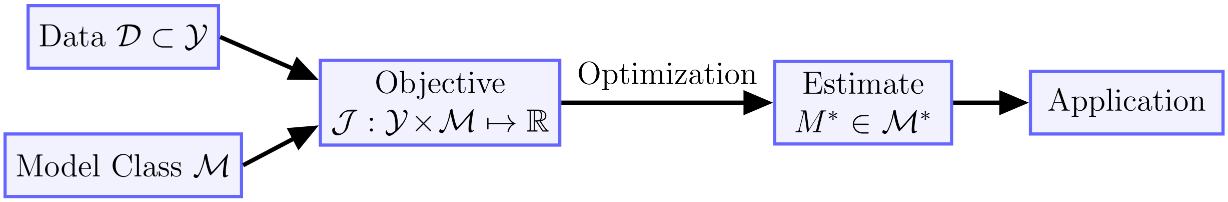

Data $\mathcal{D} \subset \mathcal{Y}$
Model Class $\mathcal{M}$
Objective $\mathcal{J} : \mathcal{Y} \times \mathcal{M} \mapsto \mathbb{R}$
Optimization
Estimate $M^* \in \mathcal{M}^*$
Application

Figure 1.1: System ID flow chart.

1.1 Design considerations in system identification

Broadly speaking, the goal of system ID is to find a mathematical model that can produce an estimate $\hat{\mathbf{y}} \in \hat{\mathcal{Y}}$ for any $\mathbf{y} \in \mathcal{Y}$ such that the expectation $\mathbb{E}_{\pi(\mathbf{y})}[\ell(\mathbf{y}, \hat{\mathbf{y}})]$ is minimized, where $\hat{\mathcal{Y}}$ is the range of the model, $\pi(\mathbf{y})$ is the probability density function (pdf) of $\mathbf{y}$, and $\ell : \mathcal{Y} \times \hat{\mathcal{Y}} \mapsto \mathbb{R}$ is a measure of model error. In the previous discussion, we described how the optimal model set $\mathcal{M}^*$ depends on the specification of the $(\mathcal{D}, \mathcal{M}, \mathcal{J})$ triple. Consequently, $\mathcal{M}$ and $\mathcal{J}$ must be properly designed such that $\mathcal{M}^*$ minimizes the expected model error as much as possible with the data $\mathcal{D}$ provided. In this section, we discuss important considerations that go into this design and identify the merits and drawbacks of some of the most common choices.

Before we begin, we must point out that the main challenge in minimizing the expected model error is that in all practical applications, the available training data are only a small subset of $\mathcal{Y}$. Therefore, the values of ℓ over the training data are not necessarily representative of the values of ℓ over unseen data. Consequently, one must be concerned not only with the training model fit, but also with the model generalizability. A model is said to "generalize" if the distribution of ℓ values over the training data is similar to that over unseen data. If the user is unduly concerned with only one of these aspects, it is typically the case that the other will suffer. Thus, system ID is often a tradeoff between the model's fit and its generalizability. This tradeoff is also known as the bias-variance tradeoff since good model fit and generalizability tend to yield low bias and variance, respectively. Managing this tradeoff is a key aspect of proper design of the model class and objective function.

1.1.1 Model class

The first design point within system ID that we discuss is the model class selection. At the heart of this design is the decision on how "complex" the candidate models should be. There are different ways of quantifying the complexity of a model, but the idea is that complexity refers to the diversity of output behavior from a given model or set of models. Often, one would like a model class that is broad enough to include the dynamics of the system of interest, but there are a number of reasons why making the model class as complex as possible may be undesirable. The first ties back to the bias-variance tradeoff. The more flexible a model is, the more likely it is to overfit and display unexpected behavior once it leaves the domain of the training data. Another consideration is the available computational resources to run the model. This becomes a concern if the model needs to be evaluated in time-constrained environments such as real-time control or if the system of interest is high-dimensional such as spatio-temporal systems. In the former case, the model class should contain only cheap to evaluate models, which are usually less expressive, and in the latter case, the model class is typically constrained to contain only reduced-order models. The last consideration is that models in the model class should not violate known information about the system. For example, if the system is known to behave as a mass-spring-damper system, then the model form can be fixed and only physical quantities such as stiffness need be estimated. Grey-box models such as this have the added benefit of being more interpretable and easier to analyze. Another example is if the system demonstrates certain physical phenomena such as stability or energy conservation, then models that do not share these characteristics would ideally be excluded from the model class.

A natural starting point when selecting a model class is the set of linear systems. This group of models comes with the benefit of decades, if not centuries, worth of theoretical research that make them interpretable and easy to work with. For example, it is straightforward to switch between continuous and discrete time, and the stability properties of the system can be found directly from the eigenvalues of the dynamics matrix. This interpretability also allows for restricting the model space to only stable models by projecting arbitrary matrices into the stable set using decomposition methods [62]. Additionally, linear models are cheap to evaluate and sometimes even to estimate. Evaluation requires only a matrix-vector multiplication, and for certain objectives, estimation only requires solving a linear least squares problem. A possible drawback to using linear models, however, is that their dynamic behavior is significantly limited. Chaos, limit cycles, and finite escape times are all examples of dynamic behavior that cannot be produced by linear systems. Common examples of linear systems used in system ID include autoregressive models, such as ARX [39] and ARMAX [5], and linear state-space models [94].

If more model flexibility is desired than that which is afforded by linear models, the next step up in complexity is a basis expansion. Basis expansions take the form $\sum_{i=1}^{p} \alpha_i \psi_i$, where $\alpha_i \in \mathbb{R}$ are unknown scalar coefficients and ψ_i are nonlinear basis functions. The left-hand side and the inputs to ψ_i are omitted as they are problem-dependent. These models have the option of tuning the model complexity by increasing or decreasing the number of basis functions p. They also allow for nonlinear behavior while remaining linear with respect to the unknown coefficients, which simplifies optimization for certain objective functions. The choice of basis function is left to the user, but common choices are polynomials [67] and radial basis functions [104].

Basis expansions unfortunately have the disadvantage of being increasingly sensitive to different inputs outside the training set as p increases. Therefore, they are not well-suited for complex system behavior when the training data do not adequately cover the operating regime. Instead, the usual choice for modeling complicated dynamics is a neural network [79, 109]. Neural networks can also have varying levels of complexity by altering the number of hidden layers and the number of nodes in each layer. Although they are the most expensive models to train and run out of the ones we have discussed, they can still be used in certain real-time applications [110, 45]. The dynamical properties, however, are difficult to interpret from the network parameters.

1.1.2 Objective function

The second design point that we consider is how to design an objective function that can properly discern "good" models within the chosen model class. Again, the main consideration is balancing the fit and generalizability of the model. To address the fitting concerns, every objective function includes at least one term that penalizes some notion of model error. Most often, this term is chosen to be the average model error over the training data, i.e., $\frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{y}^{(i)},\hat{\mathbf{y}}^{(i)})$, where i indexes the training data. The justification for this choice comes from the law of large numbers, which states that under certain conditions, the average value of a population of samples from a random variable will converge to the random variable's expected value as the number of samples approaches infinity. Therefore, the objective function can be seen as an approximation of the expected value $\mathbb{E}_{\pi(\mathbf{y})}[\ell(\mathbf{y},\hat{\mathbf{y}})]$ that the user seeks to minimize. However, in many problems, the assumptions of the law of large numbers are not met. One such assumption is that the samples should be independent and identically distributed. Since the state of a dynamical system depends on previous states, data collected from a single, uninterrupted experiment will not be independent. Secondly, the result of the law of large numbers only applies to the asymptotic behavior of the sample mean and makes no statement on how close the sample mean is to the expected value for finite samples. Therefore, the sample mean could be quite far from the expected value. If the variance of ℓ is bounded, then a probabilistic bound could be set on the deviation of the sample mean from the expected value using Chebyshev's inequality, but the number of samples would still need to be relatively large to achieve a meaningful bound. Thus, for small training sets, this objective function could be a very poor approximation of the expected model error.

Although fit is usually the primary concern, the generalizability of a model is often just as important. In fact, recent trends in the literature have moved toward more flexible model classes such as neural networks, and, as a result, generalization has increasingly become a concern for system ID. Addressing generalizability in the objective function involves incorporating a preference for "simpler" or less complex models through a process known as regularization. Regularization can either be explicit by adding a penalty or constraint directly into the objective function, or it can be implicit by any other means that avoids overfitting. In system ID, a common form of explicit regularization includes adding a norm on the parameter vector to penalize large or high-order parameter values. Examples include the L_0 norm as in sparse regression [36], the L_1 norm as in lasso regression [55], the L_2 norm as in ridge regression [15], and the Hankel nuclear norm for Markov parameter estimation [76]. Another form of explicit regularization is adding a physics-informed term as in physics-informed neural networks [81]. Examples of implicit regularization are ubiquitous in training neural networks and include stochastic optimization methods such as stochastic gradient descent [95], early stopping [72], and dropout [97]. Though these methods have shown to improve estimation on a variety of problems, they are largely *ad hoc* and lack rigorous theoretical justification.

In addition to these concerns relating to the bias-variance tradeoff and the lack of theoretical justification supporting many of the proposed solutions, there is also the practical consideration pertaining to the ease of optimization of the objective. Specifically, the objective surface should not trap optimizers in local minima, and the optimization should be computationally tractable. The most reliable way to ensure the objective can be optimized is by formulating the optimization as a linear least squares problem. In this case, a closed-form solution exists and can be computed simply by solving a linear system of equations. This approach is applicable whenever the parameters enter linearly into the objective and can therefore sometimes be used in nonlinear system ID as in sparse identification of nonlinear dynamics (SINDy). Also, L_2 regularization can be added to this approach while still preserving the existence of a closed-form solution. This formulation, however, is quite restrictive, and objective functions other than linear least squares are often needed.

1.2 Shortcomings of existing objectives

Now that we have discussed the high-level considerations of properly designing an objective function, we turn to a more detailed discussion of the specific challenges that are most commonly encountered in practice. Often, objective functions are designed with the optimal model in mind, but for practical purposes, the behavior of the objective on non-optimal models is just as important. The reason for this is that the objective function value of each model in $\mathcal{M}$ determines the topology of the objective surface over which we must optimize. In order for this surface to be amenable to efficient optimization, we would intuitively like for the objective to (1) assign a “proper” ordering of the models in $\mathcal{M}$, and (2) better distinguish between models as more data are added.

What constitutes as a “proper” ordering of the models will differ among users. From our perspective, we argue that models that deliver accurate short-term predictions but later become significantly inaccurate are preferable to models that consistently deliver mildly inaccurate predictions over any time period. This perspective originates from the fact that in many control and forecasting contexts, the estimated state can be adjusted as more measurements are acquired. Therefore, predictions at future times can also be adjusted as they draw closer in time. Moreover, virtually every model contains some amount of error, and the longer a model is simulated, the more time that error has to accumulate. As a result, the predictions of these models almost always degrade the farther they are into the future. The accuracy of these long-term predictions should not be given equal weight in evaluating a model's performance as predictions in the short-term. If models are ordered properly, then the optimizer will be making meaningful progress as it moves toward lower objective function values.

Secondly, measurements carry information on the system from which they are collected, and we would like an objective function that use this information as efficiently as possible. To tell if an objective function is effectively extracting information from new data, one must look at how its assignment of objective values changes with added data. A good objective should not only update its assignments with new data, but it should do so in a way that increasingly exaggerates the difference between good and bad models. If the relative difference between the objective values of good and bad models remains nearly constant, or in the worst case becomes smaller, with additional data, then the objective is not properly utilizing the data. Furthermore, how well an objective performs in this regard can have optimization implications. If the difference between good and bad models shrinks with more data, then an optimizer will have greater difficulty finding its way toward good models.

1.2.1 Fundamental objectives

Next, let us introduce two fundamental objectives so that we may assess their quality in regard to these considerations. By far, the most common objective function in system ID is the MSE objective defined as

$$\mathcal{J} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2, \tag{1.1}$$

where the data $\mathbf{y}_i \in \mathcal{D}$ come from a single trajectory, and i indexes time within this trajectory. This objective is ubiquitous throughout estimation communities, and many variations of it simply change how the estimate $\hat{\mathbf{y}}$ is evaluated. In system ID, $\hat{\mathbf{y}}$ tends to be evaluated in one of two ways. For expositional purposes, let us define a propagator $\Psi : \mathcal{Y} \mapsto \mathcal{Y}$ that takes an element in $\mathcal{Y}$ and maps it forward one step in time to another element in $\mathcal{Y}$. One approach takes an initial condition $\mathbf{y}_0 \in \mathcal{D}$ and repeatedly applies Ψ to estimate each $\mathbf{y}_i$. The objective for this approach is defined as

$$\mathcal{J} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \Psi^i(\mathbf{y}_0))^2, \tag{1.2}$$

where Ψ^i denotes i compositions of Ψ. This objective is used to train a number of machine learning models [16, 59, 80]. The error term in this objective is sometimes referred to as the simulation error, so we refer to this objective throughout the dissertation as the *simulation objective.*

The other common approach is to use the last measurement as the input to Ψ and only predict one step into the future. This objective is defined as

$$\mathcal{J} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \Psi(\mathbf{y}_{i-1}))^2, \tag{1.3}$$

and it is used within popular system ID algorithms [101, 10] and for training certain neural networks [35]. The error in models like this that use past data in their estimates is sometimes referred to as prediction error. Since this objective only uses the most recent data point, we refer to it throughout the dissertation with the more specific term *propagator objective.*

1.2.2 Quantitative assessment

To investigate the behavior of these two fundamental objectives, consider a simple setting where the underlying system is the sinusoid $\sin(t)$, for $t \in [0, \infty)$. Suppose our model is $\sin(\omega t)$, and our goal is to estimate ω. We have collected a set of noiseless data at times t_i for $i = 1, 2, \ldots, n$ separated by intervals $\Delta t = 0.1$. To assess the quality of a proposed value of ω, we define the following versions of the simulation and propagator objectives (in the time-domain[1]):

$$\mathcal{J}_{\text{sim}}(\omega) = \frac{1}{n}\sum_{i=1}^{n}\Big(\sin(t_i) - \sin(\omega t_i)\Big)^2, \tag{1.4}$$

and

$$\mathcal{J}_{\text{prop}}(\omega) = \frac{1}{n}\sum_{i=1}^{n}\Big(\sin(t_i) - \sin(t_{i-1} + \omega \Delta t)\Big)^2, \tag{1.5}$$

respectively.

For the sake of illustration, let us consider two candidate values: $\omega = 0.95$ and $\omega = 0.50$. With the knowledge that the true value is $\omega = 1.0$, we understand that $\omega = 0.95$ should clearly be the better model. The outputs of the two models are plotted in Fig. 1.2a, and the value of the two objectives are plotted over time in Fig. 1.2b. First, let us consider the short-term behavior of the two models. The $\omega = 0.95$ model begins very close to the truth, while the $\omega = 0.50$ model immediately begins to deviate. In our previous discussion, we argued that models with good short-term predictions should be preferred, and the performance of these two models clearly support that argument. Looking at the simulation MSE, we see that although the objective function for $\omega = 0.5$ quickly becomes much larger than that of $\omega = 0.95$, by the end of time span of the data, the objective value for $\omega = 0.95$ has actually surpassed that of $\omega = 0.5$! Because the simulation objective weighs all errors equally in time, it ends up assigning a better objective value to a much worse model. In contrast, the propagator objective only considers short-term, specifically one-step, prediction. Therefore, it can properly distinguish between the models regardless of how much time has passed.

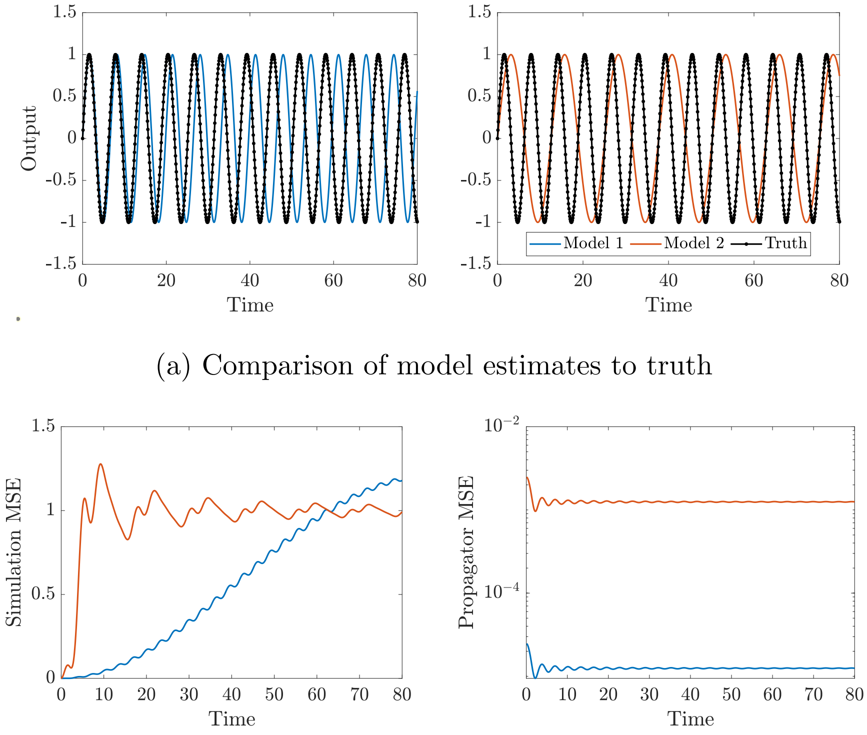
Next, let us consider the behavior of the objectives as the number of data increases. Recall that we would like an objective that becomes increasingly better at distinguishing between good and bad models as more data become available. In the simulation objective, we see that the opposite is the case. As more data arrive, the objective values of the two models come closer together. In fact, if we take the limit of the continuous-time simulation objective as $t \to \infty$, we see that the objective for every single model other than $\omega \equiv 1$ approaches 1. That is,

$$\lim_{t\to\infty} \frac{1}{t}\int_0^t (\sin(t) - \sin(\omega t))^2 \mathrm{d}t = 1, \quad \forall \omega \neq 1. \tag{1.6}$$

Although the propagator objective does not push the objective values of the two models closer together, it also does not push them farther apart. This observation reflects the fact

[1] For this linear problem, it is more appropriate to consider frequency-domain system ID, which would not encounter the problems described here. However, these types of time-domain system ID procedures using least squares-based regression/machine learning approaches are increasingly being used for complex nonlinear systems, and we seek to show that they can be limited in an extremely simple setting.

that the propagator objective does not assess long-term prediction, and it therefore receives no new information from increasing the time span of the data beyond a single period. Aside from this shortcoming, it may seem like the propagator objective can reliably distinguish good and bad models. However, we will later show that this objective quickly breaks down once noise is added to the data.



(a) Comparison of model estimates to truth

(b) Mean squared error vs. time

Figure 1.2: Illustration of how the simulation (1.2) and propagator (1.3) objectives behave on qualitatively different models.

1.2.3 Qualitative assessment

In addition to the ability to properly order the model space, we are also interested qualitatively in an objective function's optimization surface. To investigate the fundamental objectives' merits in this regard, let us consider the case of learning a continuous-time linear pendulum

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & \theta_1 \\ \theta_2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \tag{1.7}$$

where the true parameters are $\theta_1 = 1$ and $\theta_2 = -g/L$. Here, $g = 9.81$ is the acceleration due to gravity and $L = 1$ is the pendulum length. Data are obtained in 0.1 second increments with noise standard deviation of 0.1. In addition to the two fundamental objectives, we include an objective in the form of a negative log posterior in this comparison. This is the objective that will be presented in Chapter II and discussed thoroughly in this dissertation. The surfaces of the simulation, propagator, and negative log posterior objectives are shown in Fig. 1.3. The first column of this figure corresponds to the simulation objective, the second column corresponds to the propagator objective, and the third column corresponds to the negative log posterior. The rows of this figure from top to bottom correspond to 20, 40, and 80 data points, respectively, collected in 0.1 second increments.

In the left panel, we see the effect of being unable to properly order the model space. There are multiple local minima in the objective function, and as more data are added, the surface becomes flatter and even more local minima arise. In the middle column, we see that focusing only on one-step-ahead prediction can create a very smooth surface, but adding more data has almost no effect on the objective whatsoever. In this case, finding the minimum of the objective is fast, but the estimated model might not display the proper long-term behavior. Lastly, the third column represents the objective arising from the probabilistic approach discussed in the next chapter. In this objective, the surface is smooth like the propagator objective, but adding more data actually has a beneficial effect. As the number of data increases, the objective function becomes steeper in the direction of the minimum, which is advantageous for efficient optimization. This effect is a result of the posterior being able to better distinguish between the models. Lastly, the negative log posterior has a similar curved shape as the simulation objective, suggesting that it is still able to assess long-term behavior as well.

1.2.4 Discussion on state-of-the-art

The issue of creating an objective function that is amenable to optimization is a well-known problem in system ID [77]. As a result, many methods for system ID possess a component meant to smooth the objective in one form or another. As we saw in the previous sections, having a large number of data can complicate the objective function surface and exacerbate optimization difficulties. Consequently, many of the smoothing techniques center around discounting the data in one form or another. In this section, we will discuss the state-of-the-art approaches for objective function smoothing and why they are still insufficient for managing error accumulation.

The first example is simulated annealing [52], which smooths the posterior by scaling down the influence of the likelihood by a discount factor. As optimization progresses, the likelihood is gradually scaled up to its full weight according to a "cooling schedule." This algorithm can be difficult to use in practice since its effectiveness strongly depends on the cooling schedule, which must be chosen by the user and is largely problem-dependent. Furthermore, the algorithm addresses only the effects of error accumulation and not the underlying cause, and as a result, the posterior is still filled with local minima. A variation known as data annealing [34] starts with training data from a short time period and introduces more data over a gradually increasing time period. This has a similar "cooling" effect of incrementally increasing the influence of the likelihood on the posterior. Using data from a shorter training period at the early iterations can prevent the large error accumulation associated with long simulation times, but it still lacks a mechanism for handling error accumulation once the full dataset has been introduced.

Other algorithms [9] directly address error accumulation by simulating the system output at many different initial times and using only the data within a specified time horizon of the initial times to evaluate the fit of each trajectory. An example of an extreme case of this is standard and exact dynamic mode decomposition (DMD) [101]. In these algorithms, the Koopman operator is estimated by finding the best linear mapping of the data forward only one step in time, leading to a convex optimization problem. Algorithms that use a time horizon greater than one fall generally under the category of multiple shooting (MS). In such algorithms, the selected time horizon usually depends on the approximate time scale of the system. The main issue with these algorithms is that they do not allow for flexibility in the case that the state components have different time scales.

1.3 Contributions

The goal of this work is to present an objective function that can address many of the concerns stated in the previous sections. Specifically, we would like an objective function that incorporates regularization, properly manages error accumulation, and is derived directly from the dynamics model such that it is theoretically justifiable and interpretable within the perspective of the model formulation. Moreover, this objective should be general enough that it contains many other objectives as special cases, thereby lending a new perspective from which to understand these existing objectives. Lastly, we would like a quantification of the uncertainty of the estimates produced by this objective, further increasing interpretability by signalling when to be confident and when to be cautious in trusting the model predictions.

To create a general objective function requires modeling all sources of uncertainty of the estimation problem in the dynamics model formulation. We identify three such sources that enter into most system ID problems: model, measurement, and parameter uncertainty. Notably, we will show that accounting for model uncertainty can actually be a unifying solution for both the issues of regularization and error accumulation. Next, an objective can be derived directly from the dynamics formulation in the form of a negative log likelihood by using probabilistic first principles. Theoretical comparisons to other objective functions are then as straightforward as discovering what simplifying assumptions must be made in the dynamics model and what prior distributions must be placed on the parameters to yield a certain objective as its negative log posterior. Finally, the uncertainty of a given estimate can be found through the quantification of each individual source of uncertainty. For the model and measurement uncertainty, this usually takes the form of an estimated covariance matrix, and for parameter uncertainty, this requires sampling from the posterior distribution.

Following this approach, we arrive at a negative log likelihood that can be used in a wide array of system ID problems. The algorithm for evaluating this likelihood has been described in [21, 87], and its development is therefore not claimed by this work. Rather, the contributions of this dissertation center around the novel way in which we use the algorithm and the original insights into the likelihood's behavior that we provide. Specifically, the contributions are the following:

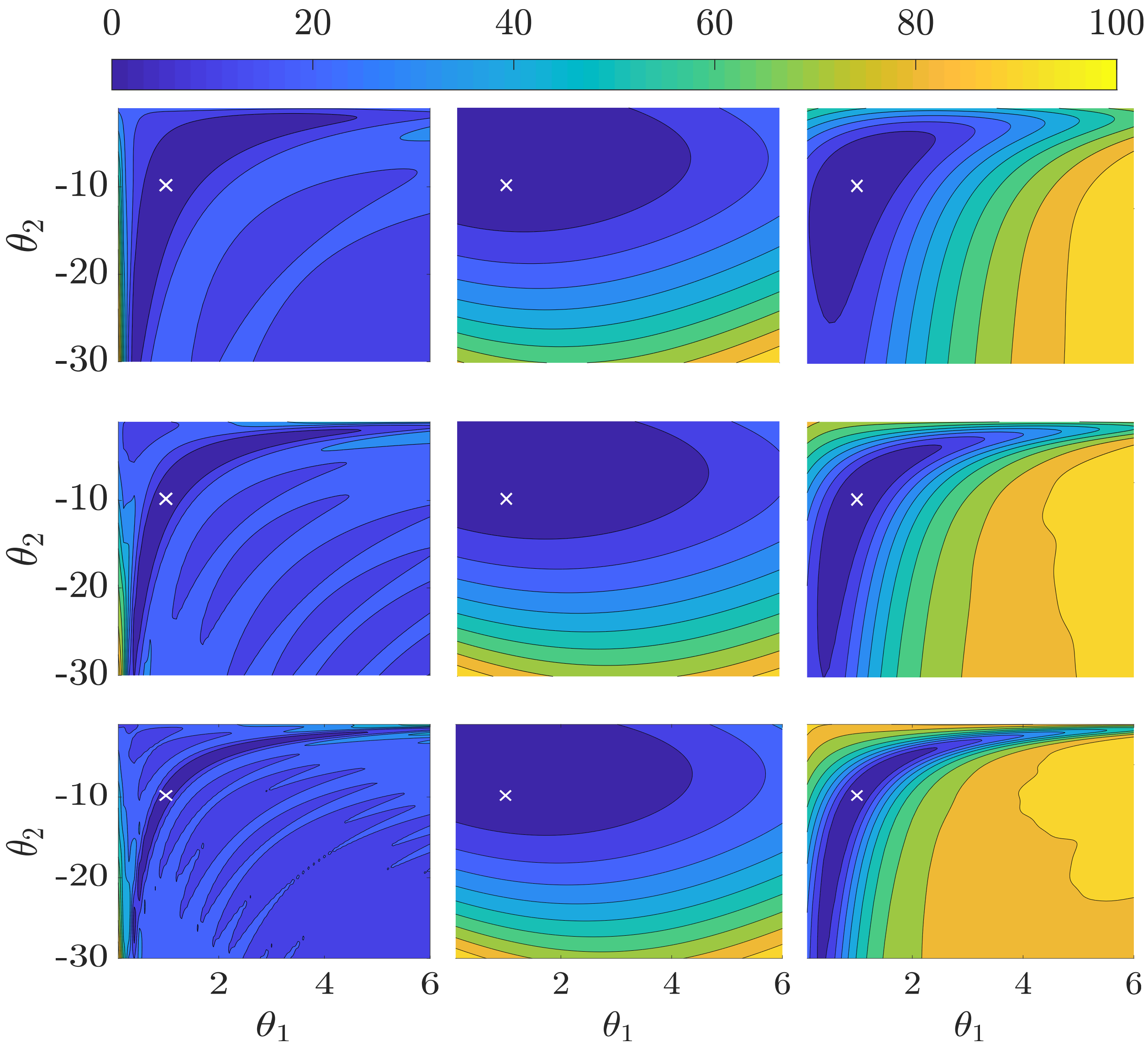
- A new perspective on the optimality of common objectives including DMD (Theorem 3), single rollout Markov parameter estimation (Proposition 2), SINDy (Theorem 4), and MS (Proposition 4) objectives by proving that they are special cases of the more general negative log posterior. This contribution is summarized in Fig. 1.4;
- Identification of explicit regularization that arises in the likelihood independent of model parameterization (Eq. (3.2)), and experimental evidence that this regularization is effective at preventing overfitting;
- Demonstration that the parameters used to specify the process noise can be used as hyperparameters for managing error accumulation, resulting in improved smoothness of the optimization surface (Fig. 1.3 and Section 4.1.2.2). Moreover, these hyperparameters can be continuous and tuned automatically, unlike the time horizon in MS;
- Novel use of the process noise for quantifying uncertainty in the model estimate without requiring simulation of the model itself (Section 5.4.1) and in the model predictions without requiring Markov chain Monte Carlo (MCMC) sampling (Fig. 5.13);
- Experimental analysis showing that enforcing symplecticity within the training process when learning Hamiltonian systems reduces data requirements and estimation uncertainty in Sections 5.4.1 and 5.4.2;

- Experimental evidence of the broad applicability of the likelihood across different model classes including linear models, partially-known models, and neural networks and across different types of noise including additive, multiplicative, Gaussian, and uniform in Sections 3.3, 4.2, and 5.4.

These contributions have led to a number of publications in peer-reviewed journals and conference proceedings. The relevant publications pertain to the following bibliography entries listed in reverse chronological order: [30, 92, 29, 27, 28].

1.4 Outline

The rest of this dissertation is organized into five additional chapters. In Chapter II, we present a general hidden Markov model (HMM) of dynamical systems and utilize probabilistic first principles to derive the marginal likelihood from this formulation. Then, we discuss an algorithm for computing the likelihood and give an analysis of its computational complexity. We finish this chapter by showing how the fundamental objectives are special cases of this marginal likelihood. Chapter III considers linear system ID. We describe two overarching approaches for estimating state-space realizations and give DMD and Markov parameter estimation as examples of each one. This chapter also gives theoretical analysis showing how each example is contained within the marginal likelihood. The chapter concludes by comparing the robustness of each approach for varying amounts of data and measurement noise. In Chapter IV, we consider more complicated systems that require nonlinear system ID methods for estimation. We specifically present SINDy and MS as nonlinear system ID methods and once again show them as special cases of the marginal likelihood. We conclude the chapter by comparing the methods to the likelihood on a number of numerical experiments that include chaos, limit cycles, and partial differential equations (PDEs). The final form of system ID we consider is learning Hamiltonians through physics-informed system ID in Chapter V. In this chapter, we present the basics of Hamiltonian mechanics and symplectic integrators for both separable and non-separable Hamiltonians. We discuss how knowledge of Hamiltonian systems can be embedded into the learning process and show how it leads to improved estimation. Then, we provide numerical experiments showing how the marginal likelihood is not only still applicable in these formulations, but can outperform approaches based on least squares objectives. This dissertation is brought to an end in Chapter VI with a discussion on our conclusions and directions for future work.



(a) Simulation objective

(b) Propagator objective

(c) Negative log posterior

Figure 1.3: Comparison of three optimization objectives for the identification of a linear pendulum. The rows correspond to the objective functions obtained after 20, 40, and 80 data points are taken at 0.1 second intervals from top to bottom. White crosses indicate true parameters. Neglecting process noise in the left column results in many local minima. Neglecting measurement noise in the middle column results in an objective insensitive to the number of data. The Bayesian approach in the right column results in the ideal scenario where the objective becomes steeper in the direction of the minimum as the amount of data increases.

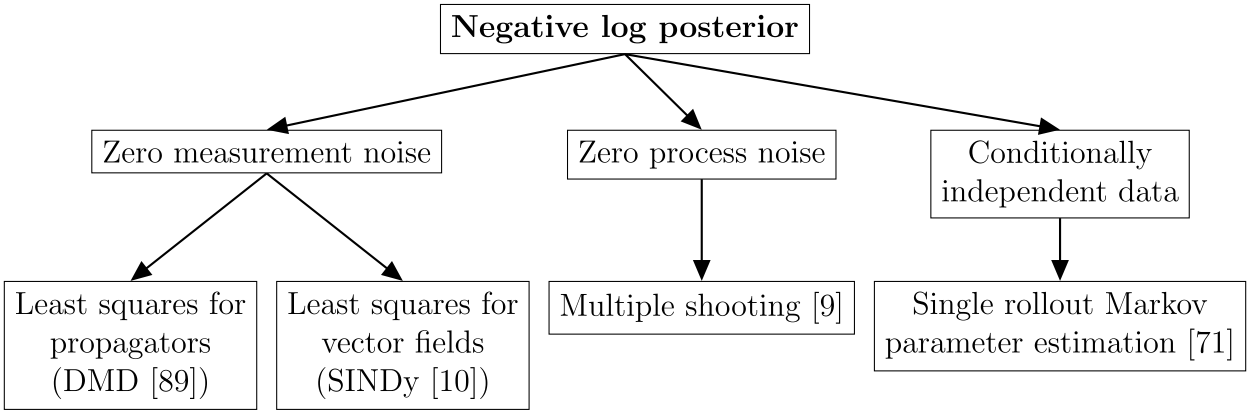

Negative log posterior
Zero measurement noise
Zero process noise
Conditionally independent data
Least squares for propagators (DMD [89])
Least squares for vector fields (SINDy [10])
Multiple shooting [9]
Single rollout Markov parameter estimation [71]

Figure 1.4: Flow chart showing how different assumptions on the negative log posterior can lead to popular system ID objectives. For brevity, this chart shows only the most notable assumptions and omits the remainder.

CHAPTER II
Probabilistic Problem Formulation

A Bayesian persepctive on system ID was first offered in 1981 by [74] in an effort to make the field "a consistent theory with formal structure." In that work, it was shown that many of the objective functions used in system ID could be formulated as negative log likelihoods or negative log posteriors through the use of probability and Bayesian inference. Notably, they showed that the simulation objective could be derived as a likelihood by modeling the entirety of the problem uncertainty as an additive Gaussian random variable in the output equation. As we saw in the last chapter, however, the simulation objective can be quite difficult to optimize. As a result, the popular set of objectives has started to trend away from this approach, and the recently developed objectives have evolved significantly. Despite this, the dominant likelihood used for Bayesian system ID largely remains unchanged [69], and Bayesian interpretations of objective functions have not kept pace.

In this section we present an alternative probabilistic problem formulation that has seen use in certain applications [50, 70, 24] but mainly remains neglected in the field of system ID. We will show in later chapters, however, that this formulation is flexible enough to include many of the system ID objectives that are currently popular and is therefore applicable to a wide range of problems. This formulation differs from the usual approach by including a process noise term in the dynamics model in addition to the usual measurement noise term. We derive the likelihood from this formulation and present a recursive algorithm for its evaluation. Then, we discuss approximations that can be made to save on computational expense and provide an analysis of the computational complexity of these approximations. Additionally, we include a discussion on our approach to quantifying the uncertainty in the parameters and considerations for selecting a point estimate for prediction. The chapter concludes by showing that the two fundamental objectives are the limiting cases of this proposed likelihood.

2.1 Notation

The following notation is used throughout this dissertation. Matrices are represented with uppercase and bold font $\mathbf{A}$ and vectors with lowercase and bold font $\mathbf{x}$. Matrices and vectors are indexed with square brackets, e.g., the (i, j)th element of $\mathbf{A}$ is denoted $\mathbf{A}[i, j]$. The norm of a vector $\mathbf{x}$ weighted by a positive definite matrix $\mathbf{W}$ is defined as $\|\mathbf{x}\|_{\mathbf{W}}^2 := \mathbf{x}^\top \mathbf{W}^{-1} \mathbf{x}$, where $^\top$ denotes the transpose. The L_2 norm of a vector and the induced L_2 matrix norm are denoted as $\|\cdot\|_2$. The norm $|\cdot|$ represents the element-wise absolute value.

The notation $\mathcal{N}(\mathbf{m}, \mathbf{P})$ denotes a normal distribution with mean $\mathbf{m}$ and covariance $\mathbf{P}$, and the notation $\mathcal{U}[\mathbf{a}, \mathbf{b}]$ denotes a uniform distribution with lower bound $\mathbf{a}$ and upper bound $\mathbf{b}$. If $\mathbf{y} = |\mathbf{x}|$ and $\mathbf{x}$ is distributed as $\mathcal{N}(\mathbf{0}, \mathbf{P})$, then $\mathbf{y}$ follows a half-normal distribution denoted as half–$\mathcal{N}(\mathbf{0}, \mathbf{P})$. Pdfs are usually represented as the function $\pi(\cdot)$ when confusion with the mathematical constant π is not possible.

2.2 Probabilistic perspective

Our approach to modeling the system dynamics is a probabilistic one in which we take the perspective that unknown quantities are best described by distributions rather than hidden values. To this end, we begin by defining the probability triple $(\Omega, \mathcal{F}, \mathbb{P})$, where the set Ω is a sample space, the collection of sets $\mathcal{F} \subseteq 2^\Omega$ is a σ-algebra, and the set function $\mathbb{P} : \mathcal{F} \mapsto [0, 1]$ is a probability measure. We model the states $\mathbf{x}_k \in \mathbb{R}^{d_x}$ and the outputs $\mathbf{y}_k \in \mathbb{R}^{d_y}$ as stochastic processes indexed by $k \in \mathbb{Z}_+ \cup \{0\}$ corresponding to time $t_k \in [0, \infty)$. Some systems also depend on control inputs, which we model as the deterministic process $\mathbf{u}_k \in \mathbb{R}^{d_u}$ with the same index set. Our modeling framework follows a discrete-time HMM [25]. However, the model may still be applied to continuous-time dynamics by discretizing differential equations in time through numerical integration. The model is defined generally as

$$\mathbf{x}_{k+1} = \mathcal{S}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}, \omega), \tag{2.1a}$$

$$\mathbf{y}_k = \mathcal{M}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}, \omega), \tag{2.1b}$$

where $\mathcal{S} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \times \mathbb{R}^{d_\theta} \times \Omega \mapsto \mathbb{R}^{d_x}$ and $\mathcal{M} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \times \mathbb{R}^{d_\theta} \times \Omega \mapsto \mathbb{R}^{d_y}$ are the state-transition and measurement functions, respectively, and are parameterized by the uncertain variable $\boldsymbol{\theta} \in \mathbb{R}^{d_\theta}$. Both operators are functions of $\omega \in \Omega$ to represent that their outputs are random variables. The system is therefore characterized by the sequences of transitional pdfs $\pi(\mathbf{x}_{k+1}|\mathbf{x}_k, \boldsymbol{\theta})$ and output pdfs $\pi(\mathbf{y}_k|\mathbf{x}_k, \boldsymbol{\theta})$ induced by $\mathcal{S}$ and $\mathcal{M}$, respectively. A visual representation of this model, in the form of a Bayesian network, is provided in Fig. 2.1.

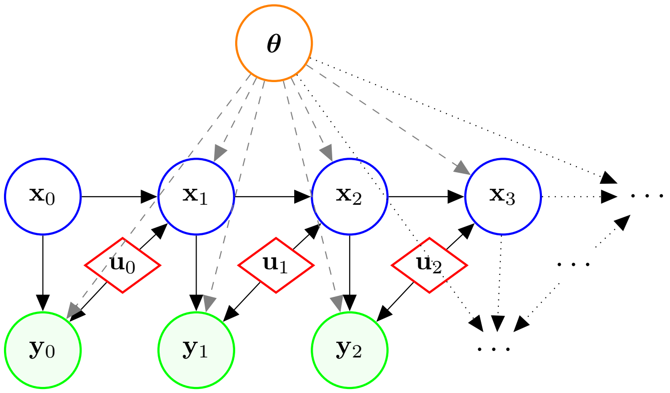


Figure 2.1: Bayesian network representation of the system ID problem.

2.3 Algorithm

The aim of Bayesian inference is to quantify the uncertainty in the parameters $\boldsymbol{\theta}$ given a collection of measurements $\mathcal{Y}_n \coloneqq (\mathbf{y}_0, \ldots, \mathbf{y}_n)$ through computation of the posterior distribution $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$. To begin, we factorize the posterior into the computable terms $\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) \coloneqq \pi(\mathcal{Y}_n|\boldsymbol{\theta})$ and $\pi(\boldsymbol{\theta})$ known as the likelihood and prior, respectively, and a normalizing constant $\pi(\mathcal{Y}_n)$ known as the evidence. This factorization is given by Bayes' rule

$$\pi(\boldsymbol{\theta}|\mathcal{Y}_n) = \frac{\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n)\pi(\boldsymbol{\theta})}{\pi(\mathcal{Y}_n)}. \tag{2.2}$$

The HMM, however, has the additional collection of random variables $\mathcal{X}_n = (\mathbf{x}_0, \ldots, \mathbf{x}_n)$ that does not appear in this factorization. Taking these variables into account yields the likelihood $\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n, \mathcal{X}_n) \coloneqq \pi(\mathcal{Y}_n, \mathcal{X}_n|\boldsymbol{\theta})$, which can be computed with the following factorization

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n, \mathcal{X}_n) = \pi(\mathbf{y}_0|\mathbf{x}_0, \boldsymbol{\theta})\pi(\mathbf{x}_0|\boldsymbol{\theta}) \prod_{k=1}^{n} \pi(\mathbf{y}_k|\mathbf{x}_k, \boldsymbol{\theta})\pi(\mathbf{x}_k|\mathbf{x}_{k-1}, \boldsymbol{\theta}). \tag{2.3}$$

This formulation introduces the states into the estimation problem, but we are only interested in learning the model parameters. We would therefore like to avoid the added difficulty that comes with estimating the parameters and states simultaneously. To do this requires marginalizing out $\mathcal{X}_n$ through evaluation of the integral $\int \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n, \mathcal{X}_n)\mathrm{d}\mathcal{X}_n$. On its face, this marginalization requires a costly $d_x(n+1)$-dimensional integration. However, the integral can be broken into $n+1$ integrals of the more manageable dimension d_x by the following decomposition

$$\mathcal{L}(\boldsymbol{\theta};\mathcal{Y}_n) = \pi(\mathbf{y}_0|\boldsymbol{\theta}) \prod_{k=1}^{n} \pi(\mathbf{y}_k|,\mathcal{Y}_{k-1},\boldsymbol{\theta}), \tag{2.4}$$

where $\pi(\mathbf{y}_k|,\mathcal{Y}_{k-1},\boldsymbol{\theta}) = \int \pi(\mathbf{y}_k,\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})\mathrm{d}\mathbf{x}_k$. Each term in this product can be efficiently computed using recursion, as shown in Algorithm 1 [87, Th. 12.3].

Algorithm 1 Recursive marginal likelihood evaluation [87, Th. 12.3]

```
Require: $\pi(\mathbf{x}_0|\boldsymbol{\theta})$, $\mathcal{Y}_n$
Ensure: $\mathcal{L}(\boldsymbol{\theta};\mathcal{Y}_n)$
Initialize $\pi(\mathbf{x}_0|\mathcal{Y}_{-1},\boldsymbol{\theta}) \coloneqq \pi(\mathbf{x}_0|\boldsymbol{\theta})$ and $\mathcal{L}(\boldsymbol{\theta};\mathcal{Y}_{-1}) \coloneqq 1$
for $k = 0, \dots n$ do
    Marginalize: $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta}) \leftarrow \int \pi(\mathbf{y}_k|\mathbf{x}_k,\boldsymbol{\theta})\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})\mathrm{d}\mathbf{x}_k$
                 $\mathcal{L}(\boldsymbol{\theta};\mathcal{Y}_k) \leftarrow \mathcal{L}(\boldsymbol{\theta};\mathcal{Y}_{k-1})\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$
    if $k < n$ then
        Update: $\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta}) \leftarrow \frac{\pi(\mathbf{y}_k|\mathbf{x}_k,\boldsymbol{\theta})}{\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})}\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$
        Predict: $\pi(\mathbf{x}_{k+1}|\mathcal{Y}_k,\boldsymbol{\theta}) \leftarrow \int \pi(\mathbf{x}_{k+1}|\mathbf{x}_k,\boldsymbol{\theta})\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})\mathrm{d}\mathbf{x}_k$
    end if
end for
```

Applying Algorithm 1 requires a computable form of the pdf $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$. One approach for computing this pdf is to use particle methods, which provide an approximation of the exact distribution [2], but this solution can be extremely costly. Another approach is to assume that the pdf is Gaussian and approximate its mean and covariance. This approach requires computing mean and covariance integrals, which is often also expensive, but numerical techniques can be used to approximate these integrals in an efficient manner. The usual techniques perform this approximation through linearization of $\mathcal{S}$ or $\mathcal{M}$ using Taylor series expansion or through Gaussian integration, e.g., the unscented transform or Gauss-Hermite cubature. Though these schemes yield (generally) biased estimates, they are nevertheless more computationally tractable than particle methods and have empirically shown good performance.

To use these integral approximation techniques on $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$ requires also keeping track of the mean and covariance of the distributions $\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})$ and $\pi(\mathbf{x}_{k+1}|\mathcal{Y}_k,\boldsymbol{\theta})$. Since $\pi(\mathbf{x}_{k+1}|\mathcal{Y}_k,\boldsymbol{\theta})$ is induced by the function $\mathcal{S}$, its first two moments can be found through integral approximation. These methods, however, cannot be applied to find the first two moments of the update distribution $\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})$ because a function mapping realizations of $\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$ to realizations of $\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})$ is not available. To address this issue, the cost-effective solution is to use the linear minimum mean squared error (MMSE) estimator, also known as the Kalman update. This approach requires only $\mathbb{E}\left[\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta}\right]$, $\mathbb{V}\mathrm{ar}\left[\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta}\right]$, and $\mathbb{C}\mathrm{ov}\left[\mathbf{x}_k,\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta}\right]$, which are all computable with integral approximation. The algorithm for computing a Gaussian approximation of the marginal likelihood $\hat{\mathcal{L}}(\boldsymbol{\theta};\mathcal{Y}_n)$ using these simplifying approximations is given in Algorithm 2. Although we keep this discussion general, we note that the integral approximation method that we use throughout this dissertation is the unscented transform in the form of the unscented Kalman filter (UKF) [49].

Algorithm 2 Recursive approximate marginal likelihood evaluation

```
Require: $\pi(\mathbf{x}_0|\boldsymbol{\theta})$, $\mathcal{Y}_n$
Ensure: $\hat{\mathcal{L}}(\boldsymbol{\theta};\mathcal{Y}_n)$
Initialize $\pi(\mathbf{x}_0|\mathcal{Y}_{-1},\boldsymbol{\theta}) := \pi(\mathbf{x}_0|\boldsymbol{\theta})$ and $\hat{\mathcal{L}}(\boldsymbol{\theta};\mathcal{Y}_{-1}) := 1$
for $k = 0,\dots n$ do
    Marginalize: $\boldsymbol{\mu}_k \leftarrow \mathbb{E}_{\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})}\left[\mathcal{M}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega)\right]$
                 $\mathbf{S}_k \leftarrow \mathbb{V}\mathrm{ar}_{\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})}\left[\mathcal{M}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega)\right]$
                 $\mathbf{U}_k \leftarrow \mathbb{C}\mathrm{ov}_{\pi(\mathbf{x}_k,\mathbf{x}_{k-1}|\mathcal{Y}_{k-1},\boldsymbol{\theta})}\left[\mathcal{S}(\mathbf{x}_{k-1},\mathbf{u}_k,\boldsymbol{\theta},\omega),\mathcal{M}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega)\right]$
                 $\hat{\mathcal{L}}(\boldsymbol{\theta};\mathcal{Y}_k) \leftarrow \hat{\mathcal{L}}(\boldsymbol{\theta}\mathcal{Y}_{k-1})\frac{1}{\sqrt{(2\pi)^{d_y}\det(\mathbf{S}_k)}}\exp\left(-\frac{1}{2}\left\|\mathbf{y}_k-\boldsymbol{\mu}_k\right\|_{\mathbf{S}_k}\right)$
    if $k < n$ then
        Update: $\mathbf{K}_k \leftarrow \mathbf{U}_k\mathbf{S}_k^{-1}$,
                $\mathbf{m}_k^+ \leftarrow \mathbf{m}_k + \mathbf{K}_k(\mathbf{y}_k - \boldsymbol{\mu}_k)$,    $\mathbf{P}_k^+ \leftarrow \mathbf{P}_k - \mathbf{K}_k\mathbf{U}_k^\top$
        Predict: $\mathbf{m}_{k+1} \leftarrow \mathbb{E}_{\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})}\left[\mathcal{S}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega)\right]$
                 $\mathbf{P}_{k+1} \leftarrow \mathbb{V}\mathrm{ar}_{\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})}\left[\mathcal{S}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega)\right]$
    end if
end for
```

2.3.1 Special noise forms

If more is known about the forms of $\mathcal{S}$ and $\mathcal{M}$, integral approximation can sometimes be replaced by closed-form solutions. To this end, it is useful to separate the dynamics and output functions into deterministic and stochastic components. Let $\Psi : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \times \mathbb{R}^{d_\theta} \mapsto \mathbb{R}^{d_x}$ and $h : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \times \mathbb{R}^{d_\theta} \mapsto \mathbb{R}^{d_y}$ represent deterministic dynamics and measurement functions, respectively, and let their stochastic counterparts be represented by stochastic processes $\boldsymbol{\xi}_k(\omega) \in \mathbb{R}^{d_x}$ and $\boldsymbol{\eta}_k(\omega) \in \mathbb{R}^{d_y}$, respectively. For simplicity, we assume that these stochastic processes are stationary.

Realizations of the random variable $\boldsymbol{\xi}_k$ are known as process noise. We interpret this term as accounting for all forms of model uncertainty such as numerical inaccuracies or insufficient model expressiveness, and it can also partially account for the uncertainty pertaining to the dynamics model parameters. Similarly, $\boldsymbol{\eta}_k$ is the measurement noise and can hold the modeling and parameter uncertainties of the observation model h. In addition, $\boldsymbol{\eta}_k$ can also contain sensor noise. We interpret this term as representing the measurement uncertainty. With our division of $\mathcal{S}$ and $\mathcal{M}$ into deterministic and stochastic components, the choice of how to model the process and measurement noise will fully determine the forms of the three distributions from Algorithm 1: $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$, $\pi(\mathbf{x}_k|\mathcal{Y}_k,\boldsymbol{\theta})$, and $\pi(\mathbf{x}_{k+1}|\mathcal{Y}_k,\boldsymbol{\theta})$. In the following sections, we look at how two noise forms in particular simplify the approximating computations in Algorithm 2.

2.3.1.1 Additive Gaussian noise

The first noise form that we consider is a widely used and highly versatile form that serves as the model for the majority of the numerical examples in this dissertation. This noise model is additive Gaussian noise, and the system model for this noise is defined as

$$\mathcal{S}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega) = \Psi(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta}_\Psi) + \boldsymbol{\xi}_k(\omega), \qquad \boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)), \tag{2.5a}$$

$$\mathcal{M}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta},\omega) = h(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta}_h) + \boldsymbol{\eta}_k(\omega), \qquad \boldsymbol{\eta}_k \sim \mathcal{N}(\mathbf{0},\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)), \tag{2.5b}$$

with uncertain initial condition $\mathbf{x}_0(\boldsymbol{\theta}_{\mathbf{x}_0})$. For ease of reference throughout the dissertation, the uncertain parameters are partitioned as $\boldsymbol{\theta} = \begin{bmatrix}\boldsymbol{\theta}_{\mathbf{x}_0}^\top & \boldsymbol{\theta}_\Psi^\top & \boldsymbol{\theta}_h^\top & \boldsymbol{\theta}_\Sigma^\top & \boldsymbol{\theta}_\Gamma^\top\end{bmatrix}^\top \in \mathbb{R}^{d_\theta}$ corresponding to the initial condition, dynamics, observation function, process noise covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{d_x \times d_x}$, and measurement noise covariance $\boldsymbol{\Gamma} \in \mathbb{R}^{d_y \times d_y}$, respectively. Although this noise form is often chosen for convenience, the choice of a Gaussian distribution to model $\boldsymbol{\xi}_k$ and $\boldsymbol{\eta}_k$ is theoretically justified by the Principle of Maximum Entropy. This principle states that for a given mean and covariance, the Gaussian distribution assumes the least amount of additional information beyond these first two moments [44]. The mean is assigned to be zero under the assumption that the optimal estimate will be close to unbiased, and the values of the covariances are left as part of the estimation problem.

For additive Gaussian noise, there are a number of simplifications that can be made in the equations of Algorithm 2. Many of these simplified equations require statistics of a function output, so for representational purposes, we denote $\Psi(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\theta})$ and $h(\mathbf{y}_k,\mathbf{u}_k,\boldsymbol{\theta})$ as Ψ_k and h_k, respectively. The first two moments of the three pdfs of interest for this noise model are given by the following equations. The mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ of $\pi(\mathbf{x}_k|\mathcal{Y}_{k-1},\boldsymbol{\theta})$:

$$\mathbf{m}_k = \mathbb{E}\left[\Psi_k\right], \tag{2.6a}$$

$$\mathbf{P}_k = \mathbb{V}\mathrm{ar}\left[\Psi_k\right] + \boldsymbol{\Sigma}. \tag{2.6b}$$

The mean $\boldsymbol{\mu}_k$ and covariance $\mathbf{S}_k$ of $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1}, \boldsymbol{\theta})$:

$$\boldsymbol{\mu}_k = \mathbb{E}\left[h_k\right], \tag{2.7a}$$

$$\mathbf{S}_k = \mathbb{V}\mathrm{ar}\left[h_k\right] + \boldsymbol{\Gamma}, \tag{2.7b}$$

$$\mathbf{U}_k = \mathbb{C}\mathrm{ov}\left[\Psi_k, h_k\right]. \tag{2.7c}$$

The mean $\mathbf{m}_k^+$ and covariance $\mathbf{P}_k^+$ of $\pi(\mathbf{x}_k|\mathcal{Y}_k, \boldsymbol{\theta})$:

$$\mathbf{K}_k = \mathbf{U}_k \mathbf{S}_k^{-1}, \tag{2.8a}$$

$$\mathbf{m}_k^+ \approx \mathbf{m}_k + \mathbf{K}_k(\mathbf{y}_k - \boldsymbol{\mu}_k), \tag{2.8b}$$

$$\mathbf{P}_k^+ \approx \mathbf{P}_k - \mathbf{K}_k \mathbf{U}_k^\top. \tag{2.8c}$$

These last three equations represent the linear MMSE estimator. Therefore, the right-hand sides are approximations of $\mathbf{m}_k^+$ and $\mathbf{P}_k^+$ and are only exact if $\mathbf{x}_k$ and $\mathbf{y}_k$ are jointly Gaussian.

We remark that this model form contains the special case of linear-Gaussian systems, which are defined by linear representations of Ψ and h and additive Gaussian forms of $\boldsymbol{\xi}_k$ and $\boldsymbol{\eta}_k$. Linear-Gaussian systems are notable because in such a formulation, all three pdfs of interest are Gaussian. As a result their means and covariances can be computed analytically with the Kalman filter (KF).

2.3.1.2 General additive/multiplicative noise

The other noise model that we consider is additive and multiplicative noise with arbitrary distributions. This noise model is defined as

$$\mathcal{S}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}, \omega) = \Psi(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}) \odot \mathbf{w}_k(\omega) + \boldsymbol{\xi}_k(\omega), \tag{2.9a}$$

$$\mathcal{M}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}, \omega) = h(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}) \odot \mathbf{v}_k(\omega) + \boldsymbol{\eta}_k(\omega), \tag{2.9b}$$

where $\odot$ represents element-wise multiplication. The $\odot$ operation is defined when the dimensions of the operands match or when the operands are a matrix and a vector whose length equals the number of matrix columns. In the latter case, $\odot$ multiplies the i-th column of the matrix by the i-th element of the vector. The multiplicative noise terms $\mathbf{w}_k$ and $\mathbf{v}_k$ are defined to have means $\bar{\mathbf{w}}$ and $\bar{\mathbf{v}}$ and covariances $\mathbf{Q}$ and $\mathbf{R}$, respectively. The additive noise terms $\boldsymbol{\xi}_k$ and $\boldsymbol{\eta}_k$ are defined to have zero means and covariances $\boldsymbol{\Sigma}$ and $\boldsymbol{\Gamma}$, respectively. Since we are interested in only the first two moments of the pdfs of interest, the higher order moments of these noise terms are not needed and can be arbitrary. However, it is worth noting that the closer the higher moments are to zero, the better the approximation of the marginal likelihood.

Similarly to the additive Gaussian noise case, there are a number of simplifications that can be made to the equations of Algorithm 2 when the noise is multiplicative. These equations were first derived in 1971 by [82] for linear systems with multiplicative scalar-valued Gaussian noise. Here, we extend these equations to the case of vector-valued noise and provide the results as follows. The mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ of $\pi(\mathbf{x}_k|\mathcal{Y}_{k-1}, \boldsymbol{\theta})$:

$$\mathbf{m}_k = \mathbb{E}\left[\Psi_k\right] \odot \bar{\mathbf{w}}, \tag{2.10a}$$

$$\mathbf{P}_k = \mathbb{E}\left[\Psi_k \Psi_k^\top\right] \odot \mathbf{Q} + \mathbb{V}\text{ar}\left[\Psi_k\right] \odot (\bar{\mathbf{w}}\bar{\mathbf{w}}^\top) + \boldsymbol{\Sigma}. \tag{2.10b}$$

The mean $\boldsymbol{\mu}_k$ and covariance $\mathbf{S}_k$ of $\pi(\mathbf{y}_k|\mathcal{Y}_{k-1}, \boldsymbol{\theta})$:

$$\boldsymbol{\mu}_k = \mathbb{E}\left[h_k\right] \odot \bar{\mathbf{v}}, \tag{2.11a}$$

$$\mathbf{S}_k = \mathbb{E}\left[h_k h_k^\top\right] \odot \mathbf{R} + \mathbb{V}\text{ar}\left[h_k\right] \odot (\bar{\mathbf{v}}\bar{\mathbf{v}}^\top) + \boldsymbol{\Gamma}, \tag{2.11b}$$

$$\mathbf{U}_k = \mathbb{C}\text{ov}\left[\Psi_k, h_k\right] \odot \bar{\mathbf{v}}_k. \tag{2.11c}$$

The mean $\mathbf{m}_k^+$ and covariance $\mathbf{P}_k^+$ of $\pi(\mathbf{x}_k|\mathcal{Y}_k, \boldsymbol{\theta})$ are given by the same Kalman update equations in Eq. 2.8.

Notice that $\mathbb{E}\left[\Psi_k \Psi_k^\top\right]$ and $\mathbb{E}\left[h_k h_k^\top\right]$ are required when multiplicative noise is present. These quantities could be computed with the usual integral approximation methods, but for computational efficiency, we assume that the function outputs are Gaussian. This assumption allows Eqs. (2.10b) and (2.11b) to be computed in the following respective forms:

$$\mathbf{P}_k = \mathbb{V}\text{ar}\left[\Psi_k\right] \odot (\mathbf{Q} + \bar{\mathbf{w}}\bar{\mathbf{w}}^\top) + (\mathbb{E}\left[\Psi_k\right]\mathbb{E}\left[\Psi_k\right]^\top) \odot \mathbf{Q} + \boldsymbol{\Sigma}, \tag{2.12a}$$

$$\mathbf{S}_k = \mathbb{V}\text{ar}\left[h_k\right] \odot (\mathbf{R} + \bar{\mathbf{v}}\bar{\mathbf{v}}^\top) + (\mathbb{E}\left[h_k\right]\mathbb{E}\left[h_k\right]^\top) \odot \mathbf{R} + \boldsymbol{\Gamma}. \tag{2.12b}$$

Once again, if Ψ or h are linear functions, then the expectation and variance terms containing Ψ_k or h_k, respectively, can be found in closed form.

2.3.2 Computational complexity

Now that we have the equations needed for computing Algorithm 2, we turn our attention towards the computational requirements of this algorithm. We will show in the following chapters that Algorithm 2 yields more flexible and robust estimators than competing system ID approaches, but this robustness will be at the cost of increased computational complexity. In this section, we assess the cost of the algorithm for the additive Gaussian noise model from Eq. (2.5), both in the linear and nonlinear cases, by counting the number of floating-point operations (flops) required by each algorithm. We conclude this subsection by tallying the additional number of flops required if multiplicative noise is added to the model as in Eq. (2.9).

For this analysis, addition, subtraction, multiplication, and division of two floating point numbers and the logarithm of one floating point number all count as one flop. The multiplication of an $d_y \times n$ matrix by an $n \times d_\theta$ matrix then counts as $d_y d_\theta(2n-1)$ flops because each of the $d_y d_\theta$ entries of the product matrix requires n multiplications and $n-1$ additions.[1] Similarly, the multiplication of an $d_y \times n$ matrix by an $n \times 1$ vector requires $n(2n-1)$ flops. Additionally, we approximate the cost of a Cholesky decomposition, matrix inversion, and determinant performed on an $n \times n$ matrix all to be $n^3/3$ flops. Furthermore, the complexity of these algorithms strongly depends on the complexity of the dynamical and measurement models used, which will vary from problem to problem. For the sake of generality, we define the computational complexity of the dynamical model Ψ and measurement model h to be denoted as F and H, respectively. Clearly in the linear case, these variables will not be needed as the dynamical and measurement models are matrices, and the number of flops can be calculated without loss of generality. The number of flops for each algorithm will be given in terms of the problem dimensions, so recall the following notation: d_x is the dimension of the state, d_y is the dimension of the measurements, d_θ is the number of parameters, and n is the total number of measurements available.

Although we need to evaluate the full posterior pdf for estimation, our analysis focuses entirely on the computation of the marginal likelihood since this is the dominant cost in the unnormalized posterior. The prior distribution is chosen by the user and its cost is typically orders of magnitude lower than that of the likelihood computation. Recall that the KF can be used when distributions are Gaussian and that we have chosen to use the unscented transform to approximate the mean and covariance of all other distributions. Therefore, the following analysis provides results for the Kalman filtering algorithm, the unscented Kalman filtering algorithm, and their prediction and update subcomponents. The full algorithms off which we base this analysis can be found in the appendix as Algorithms 4 and 5 for the KF and UKF, respectively. Table 2.1 shows the number of different types of operations required by each algorithm, and Table 2.2 shows the number of flops for each algorithm. Note that although the mean and covariance of the marginal likelihood are computed in the update step of the Bayesian algorithms, the computation of the log of this distribution is excluded from this step and is instead included only in the total. Lastly, the 18 flops outside the parentheses in the UKF total count comes from the formation of the weights, which is

[1] We only consider the naive matrix-multiplication scheme, not the asymptotically more optimal approaches such as Strassens algorithm.

Table 2.1: Tally of matrix and vector operations of Algorithms 4 and 5. VEW and MEW are element-wise vector and matrix operations, respectively, such as addition, subtraction, and element-wise multiplication and division. MV is a matrix-vector or vector-vector multiplication, and MM is matrix-matrix multiplication. Inv is a matrix inversion, Det a determinant, and Chol a Cholesky decomposition.

Algorithm	**VEW**	**MEW**	**MV**	**MM**	**Inv**	**Det**	**Chol**
KF Prediction	0	1	1	2	0	0	0
KF Update	2	2	3	6	1	0	0
KF Total	$4n$	$3n$	$6n$	$8n$	$2n$	n	0
UKF Prediction	$4d_x$	8	0	1	0	0	1
UKF Update	$4d_x + 2$	14	1	5	1	0	1
UKF Total	$(8d_x + 4)n$	$22n$	$3n$	$6n$	$2n$	n	$2n$

Table 2.2: Flop count of Algorithms 4 and 5

Algorithm	**Flop Count**
KF Prediction	${4d_x}^3 + {d_x}^2 - d_x$
KF Update	$2{d_x}^3 + \frac{1}{3}{d_y}^3 + 6{d_x}^2 d_y + 4d_x{d_y}^2 - {d_x}^2 - {d_y}^2 + 3d_x d_y - 1$
KF Total	$n(6{d_x}^3 + {d_y}^3 + 6{d_x}^2 d_y + 4d_x{d_y}^2 + {d_y}^2 + 3d_x d_y - d_x + 3d_y + 8)$
UKF Prediction	$\frac{13}{3}{d_x}^3 + 17{d_x}^2 + 4d_x + 2 + (2d_x + 1)F$
UKF Update	$\frac{1}{3}{d_x}^3 + \frac{1}{3}{d_y}^3 + 6{d_x}^2 d_y + 8d_x{d_y}^2 + 9{d_x}^2 + 4{d_y}^2 + 13d_x d_y +$ $2d_x + 6d_y + 2 + (2d_x + 1)H$
UKF Total	$n\Bigg(\frac{14}{3}{d_x}^3 + {d_y}^3 + 6{d_x}^2 d_y + 8d_x{d_y}^2 + 26{d_x}^2 + 6{d_y}^2 + 13d_x d_y +$ $6d_x + 9d_y + 13 + (2d_x + 1)(F + H)\Bigg) + 18$

required only once at the beginning of the algorithm.

The computational costs of the Bayesian algorithms are on the order $\mathcal{O}(n({d_x}^3 + {d_y}^3))$. Typically the dimension d_y of the observations is small, so this algorithm is primarily limited by the dimension d_x of the state vector. Furthermore, the dimension d_θ of the parameter vector only affects the evaluation of the prior, which is usually chosen so as to be easy to compute. Therefore, this algorithm is most efficient for problems where the state dimension is low and the parameter dimension is high, such as in nonlinear regression problems.

Next, we analyze the additional computational complexity of this filtering procedure when multiplicative noise is also considered. The equations used for this algorithm and their marginal flop counts are shown in Table 2.3. Because we assumed that the noise is stationary, the outer products $\bar{\mathbf{w}}\bar{\mathbf{w}}^\top$ and $\bar{\mathbf{v}}\bar{\mathbf{v}}^\top$ from Eqs. (2.12a) and (2.12b) need only be evaluated once. All other operations scale linearly with the number of data n. The order of the added expense is $\mathcal{O}(n({d_x}^2 + {d_y}^2))$, which is negligible in comparison to the complexity $\mathcal{O}(n({d_x}^3 + {d_y}^3))$ of the additive Gaussian system. Therefore, the additional computation required when multiplicative noise is added to the model does not affect the order of complexity of the overall algorithm.

Table 2.3: The computational complexity added by including multiplicative noise.

	Equation	Added flops
Dyn.	Eq. (2.10a)	nd_x
	Eq. (2.12a)	$5n{d_x}^2 + {d_x}^2$
Obs.	Eq. (2.11a)	nd_y
	Eq. (2.12b)	$5n{d_y}^2 + {d_y}^2$
	Eq. (2.11c)	nd_xd_y
Total:		$n(5{d_x}^2 + 5{d_y}^2 + d_xd_y + d_x + d_y) + {d_y}^2 + {d_x}^2$

2.3.3 Uncertainty quantification

The final aspect of the algorithm that we consider in this chapter is how to quantify the uncertainty in the model's parameters and its predictions. Traditional system ID approaches focus on finding a single, optimal point-estimate of the parameters. They must then rely on a validation subset of the available data disjoint from the training set to assess the quality of this singular model. In contrast, Bayesian approaches consider a set of parameter values that have a high probability with respect to the posterior distribution. The validity of the model's predictions can then be assessed through analysis of the distribution of outputs generated by the set of model parameters. If this output distribution is low-variance, then the model has high certainty, but if the model outputs are more spread out, then the model is unsure where the best estimate lies.

To utilize this Bayesian approach, we therefore require parameter samples from the posterior distribution. In general, this posterior is non-Gaussian and does not have a closed-form expression, which can make efficient sampling difficult. One approach is to approximate the posterior using easy-to-sample distributions such as Gaussians. This is the approach used in Laplace approximations and variational inference. The other approach is to draw samples from the exact posterior at the cost of additional computational expense. Certain forms of variational inference such as normalizing flows are theoretically able to sample from the exact posterior by transforming samples from a simple distribution – usually Gaussian – to samples from the posterior. The challenge, however, is learning a mapping between the two distributions, which includes all the difficulties involved with trying to learn complicated functions. Additionally, the samples generated are only from the exact posterior if the function mapping is perfectly accurate, which can almost never be achieved in practice. The other solution to sampling directly from the posterior is MCMC sampling. This approach constructs a Markov chain that converges to the posterior as its stationary distribution. One of the difficulties in this approach is achieving convergence of the Markov chain. Although asymptotic convergence is theoretically guaranteed, in practice, convergence can consume a large amount of time and computational expense. The other main difficulty of this approach is that the drawn samples are correlated, which can greatly increase the required number of samples needed to achieve a target effective sample size. Nevertheless, we opt for using MCMC sampling for its convergence guarantees at the cost of time and computation spent tuning the sampler and drawing large numbers of samples for the sake of convergence and effective sample size.

Every MCMC algorithm involves two steps: (1) draw a sample from a proposal distribution and (2) accept or reject that sample according to a probability that conserves the posterior as the chain's stationary distribution. Often, this second step uses the Metropolis-Hastings probability, which is defined as

$$\alpha = \min\left(1, \frac{\pi(\boldsymbol{\theta}^*|\mathcal{Y}_n)}{\pi(\boldsymbol{\theta}^{(k-1)}|\mathcal{Y}_n)} \frac{\varphi(\boldsymbol{\theta}^{(k-1)})}{\varphi(\boldsymbol{\theta}^*)}\right), \tag{2.13}$$

where φ is the proposal distribution, $\boldsymbol{\theta}^*$ is the proposed sample, and $\boldsymbol{\theta}^{(k-1)}$ is the current position of the Markov chain. Note that the acceptance probability only requires the ratio between the posterior pdf of two samples. Importantly, this allows us to circumvent the computation of the normalizing constant $\pi(\mathcal{Y}_n)$.

Next, if we insert Algorithm 2 into the described sampling scheme, we arrive at the approximate marginal MCMC scheme provided in Algorithm 3. The function $\hat{\mathcal{L}}(\boldsymbol{\theta}; \mathcal{Y}_n)$ is the likelihood estimator using the UKF, and we restate that pseudocode for its evaluation can be found in the appendix as Algorithm 5. In the case of linear-Gaussian systems, Algorithm 4 can be used to evaluate the exact likelihood using the KF, and as a result, Algorithm 3 becomes an exact marginal MCMC sampling scheme.

The final requirement to run this algorithm is a starting point for sampling. In MCMC sampling, it is good practice to start the sampler at a high probability point to reduce the convergence time. For this reason, we select the maximum *a posteriori* (MAP) estimate of the parameter posterior

$$\boldsymbol{\theta}^{\text{map}} = \arg\max_{\boldsymbol{\theta}} \pi(\boldsymbol{\theta}|\mathcal{Y}_n). \tag{2.15}$$

To find this point, we simply treat the negative log posterior as an objective function and use available optimization techniques to minimize it.

Algorithm 3 Approximate marginal MCMC for Bayesian inference

```
Require: Prior distribution $\pi(\boldsymbol{\theta})$
         UKF-based likelihood estimator $\hat{\mathcal{L}}(\boldsymbol{\theta}; \mathcal{Y}_n)$
         Proposal distribution $\varphi(\boldsymbol{\theta})$
         Initial sample $\boldsymbol{\theta}^{(0)}$
Ensure: Samples from stationary distribution $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$
Compute $\hat{z}^{(0)} = \hat{\mathcal{L}}(\boldsymbol{\theta}^{(0)}; \mathcal{Y}_n)$
for $k = 1$ to $N$ do
    $\boldsymbol{\theta}^* \sim \varphi$    Sample from proposal
    $z^* = \hat{\mathcal{L}}(\boldsymbol{\theta}^*; \mathcal{Y}_n)$    Compute estimated likelihood
    Compute acceptance probability

        $\alpha = \min\left(1, \frac{z^*\pi(\boldsymbol{\theta}^*)}{z^{(k-1)}\pi(\boldsymbol{\theta}^{(k-1)})} \frac{\varphi(\boldsymbol{\theta}^{(k-1)})}{\varphi(\boldsymbol{\theta}^*)}\right)$    (2.14)

    Accept $\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^*$ and $z^{(k)} = z^*$ with probability $\alpha$
    Otherwise $\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)}$ and $z^{(k)} = z^{(k-1)}$
end for
```

2.3.3.1 Sampling strategy

Even with a good starting point, however, achieving good mixing of the Markov chain can be challenging when the parameters are non-identifiable. A state-space dynamics model is at best unique up to a change of coordinates transformation, and in overparameterized cases, models are not even unique for a fixed coordinate frame. We attempt to address this issue by fixing the observation parameters $\boldsymbol{\theta}_h$ (and the control-input matrix parameters in linear time-invariant (LTI) models) at the MAP and then sampling the remaining parameters. The rationale behind this approach is to constrain the coordinate frame as a means to mitigate one of these sources of non-identifiability. Fixing parameters runs the risk of neglecting uncertainty, but we are primarily concerned with the uncertainty in the output behavior, not in the parameters. This constraint should theoretically not restrict the behavior of the model dynamics nor the uncertainty in the output, so we consider this an acceptable tradeoff.

To perform sampling, we use an MCMC within Gibbs sampling scheme. In the Gibbs sampler, the parameters are separated into groups $\{\boldsymbol{\theta}_{\mathbf{x}_0}\}$, $\{\boldsymbol{\theta}_{\Psi}\}$, and $\{\boldsymbol{\theta}_{\Sigma}, \boldsymbol{\theta}_{\Gamma}\}$ and sampled sequentially from their corresponding conditional distributions using the delayed rejection adaptive Metropolis (DRAM) algorithm [38]. To show how fixing $\boldsymbol{\theta}_h$ and using the DRAM within Gibbs approach improves sampling compared to basic DRAM, Fig. 2.2 shows samples drawn from the $\mathbf{A}$ matrix of the linear system in Section 3.3.3 using both approaches. Figs. 2.2a and 2.2b show the samples drawn when using DRAM without Gibbs and without fixing any parameters at the MAP. With this approach, 10^7 samples were drawn, the first

10^6 discarded as burn-in, and every 1,000th remaining sample was plotted. Figs. 2.2c and 2.2d show samples drawn with the DRAM within Gibbs approach with matrices $\mathbf{B}$ and $\mathbf{H}$ fixed at the MAP. For this method, 10^6 samples were drawn, 10^5 were discarded as burn-in, and every 1,000th remaining sample was plotted. Despite the former approach drawing 10 times as many samples, we observe that the latter approach covers more of the posterior. Moreover, the mixing of the chain in the DRAM within Gibbs approach appears much better.

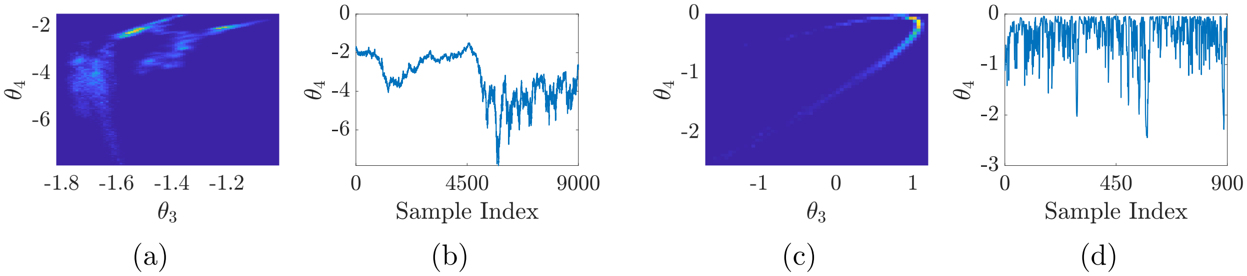


Figure 2.2: 2D marginal distributions and chains from MCMC sampling System (3.21). Figs. 2.2a and 2.2b show samples drawn using DRAM, and Figs. 2.2c and 2.2d show samples drawn using a DRAM within Gibbs procedure.

2.3.3.2 Prediction strategies

After completing sampling, we are now ready to make predictions. Whereas traditional system ID approaches define the problem through an optimization objective, the Bayesian approach separates learning and decision making. In effect, it provides a way of generating new optimization objectives and interpreting existing ones. Here, we briefly comment on the fact that this separation comes in the form of a two step procedure: (1) computing the posterior and (2) extracting a goal-oriented estimator through the specification of a loss function. For detailed discussion of these topics we refer the reader to [7].

First note that we have considered $\boldsymbol{\theta}$ to contain all uncertain parameters in the problem. For prediction, however, it is standard to make predictions into the future using deterministic models based on Ψ. As a result, we will only need to use the dynamics parameters $\boldsymbol{\theta}_\Psi$. To describe the evolution of the state $\mathbf{x}_k$, we introduce the following notation

$$\Psi^k(\mathbf{x}_0, \mathbf{u}_{0:k}, \boldsymbol{\theta}) := \Psi(\Psi(\cdots \Psi(\mathbf{x}_0, \mathbf{u}_0, \boldsymbol{\theta}), \mathbf{u}_1, \boldsymbol{\theta}) \ldots), \mathbf{u}_k, \boldsymbol{\theta}), \tag{2.16}$$

which denotes k compositions of the dynamics propagator. Next, we define the *posterior predictive* distribution of the states as an average over all possible values of the dynamics parameters conditioned on the observations

$$\pi(\mathbf{x}_k \mid \mathcal{Y}_n) = \int \pi(\mathbf{x}_k|\boldsymbol{\theta}_\Psi)\pi(\boldsymbol{\theta}_\Psi|\mathcal{Y}_n)\mathrm{d}\boldsymbol{\theta}_\Psi, \tag{2.17}$$

where we will use a deterministic prediction that discards the process noise

$$\pi(\mathbf{x}_k|\boldsymbol{\theta}_\Psi) = \delta_{\Psi^k(\mathbf{x}_0,\mathbf{u}_{0:k},\boldsymbol{\theta}_\Psi)}(\mathbf{x}_k). \tag{2.18}$$

This restriction is not explicitly necessary, but it is representative of how learned models are used in practice. We will sometimes refer to Eq. (2.18) as a deterministic simulation when we want to distinguish from a stochastic simulation in which realizations of process noise are added to the dynamics.

Then, we extract an estimator to use as the "point estimate" from the posterior. In this dissertation, we primarily rely on two different estimators unless otherwise noted. The mean estimator

$$\mathbf{x}_k^{\mathrm{avg}} = \mathbb{E}_{\pi(\boldsymbol{\theta}_\Psi|\mathcal{Y}_n)}\left[\pi(\mathbf{x}_k|\mathcal{Y}_n)\right], \tag{2.19}$$

and the parameter MAP estimator

$$\mathbf{x}_k^{\boldsymbol{\theta}^{\mathrm{map}}} = \delta_{\Psi^k(\mathbf{x}_0,\mathbf{u}_{0:k},\boldsymbol{\theta}_\Psi^{\mathrm{map}})}(\mathbf{x}_k). \tag{2.20}$$

The decision between these two estimators is problem-dependent, and we will note the choice we make in each experiment.

Finally, an evaluation metric must be selected to compare the chosen point estimate to other methods. The specific metric will depend on the system's characteristics and the desired utility of the model. More often than not, the model is intended to be used for prediction. In this case, the MSE of the deterministic simulation with respect to the system ground truth is typically used. Although we noted earlier that the simulation objective is not well-suited for the evaluation of long-time model performance, this metric can still be applicable if the evaluation time span is kept to a reasonable length, i.e., one in which small errors do not have a chance to heavily accumulate. For certain systems, however, finding a reasonable evaluation length is extremely challenging. As an example, in systems where errors grow rapidly, such as chaotic and unstable systems, predictions quickly diverge from the ground truth, and an evaluation window that ends before this divergence occurs is usually not large enough to meaningfully capture the system behavior. Therefore, alternative methods are needed to compare estimates of such systems. These methods tend to be more heavily problem-dependent and will therefore be introduced as needed throughout the dissertation.

2.4 Comparison to fundamental objective functions

Now we turn to comparing the marginal likelihood to the two fundamental objectives described in Section 1.2.1. In the literature, there have been several comparisons between the simulation and propagator objectives [21, 1, 83] that have shown that the simulation objective yields better estimates when measurement noise is present and process noise is omitted, and the propagator objective yields better estimates when process noise is present and measurement noise is omitted. In this section, we review the theory that explains these observations by specializing the marginal likelihood to two special cases. These cases are (1) zero process noise (model error is ignored) and (2) noiseless, invertible measurements (measurement error is ignored).

2.4.1 Simulation least squares

The simulation objective (1.2) uses a deterministic model, discarding process noise. This objective will be useful when discussing MS later on in Section 4.1.2. In this setting, the distribution $\pi(\mathbf{x}_k|\mathbf{x}_{k-1}, \boldsymbol{\theta}_\Psi, \boldsymbol{\theta}_\Sigma)$ reduces to the Dirac delta function $\delta_{\mathbf{x}_k}(\Psi(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \boldsymbol{\theta}_\Psi))$. The assumption of zero process noise leads to the marginal likelihood given in Theorem 1.

Theorem 1 (Marginal likelihood – zero process noise)**.** *Let the dynamics model be deterministic. Then, the marginal likelihood* (2.4) *is defined recursively as*

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_k) = \frac{\exp\left(-\frac{1}{2}\|\mathbf{y}_k - h(\Psi^k(\mathbf{x}_0, \mathbf{u}_{0:k}, \boldsymbol{\theta}_\Psi), \boldsymbol{\theta}_h)\|^2_{\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)}\right)}{\sqrt{2\pi}^{d_y}|\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)|^{\frac{1}{2}}} \tag{2.21}$$

for $k = 1, \dots, n$. Moreover the log marginal likelihood becomes

$$\begin{aligned}\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \sum_{k=1}^{n} &\left(-\frac{1}{2}\|\mathbf{y}_k - h(\Psi^k(\mathbf{x}_0, \mathbf{u}_{0:k}, \boldsymbol{\theta}_\Psi), \boldsymbol{\theta}_h)\|^2_{\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)}\right) \\ &- \frac{nd_y}{2}\log 2\pi - \frac{n}{2}\log|\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)|.\end{aligned} \tag{2.22}$$

Proof. The proof follows from the fact that a deterministic system must follow a fixed trajectory defined entirely by the parameters. In other words, we have $\pi(\mathcal{X}_n|\boldsymbol{\theta}_\Psi, \mathcal{Y}_n) = \pi(\mathcal{X}_n|\boldsymbol{\theta}_\Psi) = \delta_{\Psi(\mathbf{x}_0,\mathbf{u}_0,\boldsymbol{\theta}_\Psi),\dots,\Psi^n(\mathbf{x}_0,\mathbf{u}_{0:n},\boldsymbol{\theta}_\Psi)}(\mathcal{X}_n)$. As a result, the second term of the joint likelihood 2.3 drops out, and we are left with

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{X}_n, \mathcal{Y}_n) = \prod_{k=1}^{n} \frac{\exp\left(-\frac{1}{2}\|\mathbf{y}_k - h(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}_h)\|^2_{\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)}\right)}{\sqrt{2\pi}^{d_y}|\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)|^{\frac{1}{2}}}.$$

Because the dynamics are deterministic, we have $\mathbf{x}_k = \Psi^k(\mathbf{x}_0, \mathbf{u}_{0:k}, \boldsymbol{\theta}_\Psi)$. Thus the likelihood no longer depends on the states other than the known initial state, and what remains is the marginal likelihood as stated

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \prod_{k=1}^{n} \frac{\exp\left(-\frac{1}{2}\|\mathbf{y}_k - h(\Psi^k(\mathbf{x}_0, \boldsymbol{\theta}_\Psi), \boldsymbol{\theta}_h)\|^2_{\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)}\right)}{\sqrt{2\pi}^{d_y}|\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)|^{\frac{1}{2}}}.$$

Taking the log of this expression completes the proof. □

2.4.2 Propagator least squares

Next, we consider the ramifications on the posterior of assuming no measurement noise. We will later show that the DMD and SINDy objectives correspond to this case in Sections 3.2.1 and 4.1.1.

Consider an invertible observation operator so that the states are uniquely determined $\mathbf{x}_k = h^{-1}(\mathbf{y}_k)$. Using this assumption in System (2.5) leads to a Markovian system for the system observables

$$\mathbf{y}_{k+1} = h\left(\Psi\left(h^{-1}(\mathbf{y}_k), \mathbf{u}_k, \boldsymbol{\theta}_\Psi\right) + \boldsymbol{\xi}_k, \boldsymbol{\theta}_h\right) \tag{2.23}$$

for $k = 1, \ldots, n-1$ where $\boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma))$.

This assumption yields the marginal likelihood given in Theorem 2 below.

Theorem 2 (Marginal likelihood – noiseless, invertible observations)**.** *Let h be an invertible operator and the measurements be noiseless. Then, the marginal likelihood* (2.4) *is defined recursively as*

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_k) = |\nabla h^{-1}(\mathbf{y}_k)| \frac{\exp\left(-\frac{1}{2}\|h^{-1}(\mathbf{y}_k) - \Psi\left(h^{-1}\left(\mathbf{y}_{k-1}\right), \mathbf{u}_{k-1}, \boldsymbol{\theta}_\Psi\right)\|^2_{\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)}\right)}{\sqrt{2\pi}^{d_x}|\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)|^{\frac{1}{2}}} \tag{2.24}$$

for $k = 2, \ldots, n$ and

$$\begin{aligned}\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_1) = \log \int \exp\left(\|h^{-1}(\mathbf{y}_1) - \Psi(\mathbf{x}_0, \mathbf{u}_0, \boldsymbol{\theta}_\Psi)\|^2_{\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)}\right) \pi(\mathbf{x}_0|\boldsymbol{\theta}) \mathrm{d}\mathbf{x}_0 \\ - \frac{d_x}{2}\log 2\pi - \frac{1}{2}\log|\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)|.\end{aligned} \tag{2.25}$$

Together, the log marginal likelihood becomes

$$\begin{aligned}\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \sum_{k=2}^{n} \bigg(& \log|\nabla h^{-1}(\mathbf{y}_k)| - \frac{1}{2}\|h^{-1}(\mathbf{y}_k) - \Psi\left(h^{-1}\left(\mathbf{y}_{k-1}\right), \mathbf{u}_{k-1}, \boldsymbol{\theta}_\Psi\right)\|^2_{\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)}\bigg) \\ & - \frac{nd_x}{2}\log 2\pi - \frac{n}{2}\log|\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)| + \log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_1).\end{aligned} \tag{2.26}$$

Proof. Noiseless observations $\mathbf{y}_k = h(\mathbf{x}_k, \boldsymbol{\theta}_h)$ imply that $\|\mathbf{y}_k - h(\mathbf{x}_k, \boldsymbol{\theta}_h)\|^2_{\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)} = 0$ such that the first term of the joint likelihood 2.3 drops out. Second, we notice that we can rewrite the second term in terms of $\mathbf{y}_k$ rather than $\mathbf{x}_k$ by using the change of variables formula

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \prod_{k=1}^{n} |\nabla h^{-1}(\mathbf{y}_k)| \frac{\exp\left(-\frac{1}{2}\|h^{-1}(\mathbf{y}_k) - \Psi(h^{-1}(\mathbf{y}_{k-1}), \mathbf{u}_{k-1}, \boldsymbol{\theta}_\Psi)\|^2_{\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)}\right)}{\sqrt{2\pi}^{d_x}|\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)|^{\frac{1}{2}}}.$$

The likelihood no longer depends on the states and thus the given result is the marginal likelihood. □

As we intuitively expected, there is no marginalization over states under this assumption because the learning problem effectively "resets" after every data point. After the reset, the states are at their true value, and optimization progresses to ensure that the residual of propagation between true values is small.

Remark 1 (Data on initial condition)**.** *If the initial condition is treated as beginning when the data are obtained, then the log likelihood for the first data point becomes independent of the parameters and we can set it to an arbitrary constant.*

2.5 Summary

In this chapter, we have presented a proposed probabilistic formulation of the system ID problem. In contrast to the dominant modeling approach, we included process noise in the dynamics to account for the model uncertainty present during estimation. We then derived a marginal likelihood from this model and provided a recursive algorithm for evaluating this likelihood. We showed how the cost of the algorithm can be reduced through certain approximations and discussed how the algorithm can be simplified further for certain noise forms. Then we discussed how to use the algorithm for uncertainty quantification and identified useful estimators for prediction. Finally, we provided derivations showing how the simulation and propagator objectives are contained within this broader marginal likelihood.

CHAPTER III
Linear Time-Invariant System Identification

Linear models of dynamical systems are desirable for a number of reasons. They are cheap to evaluate, admit closed-form solutions, and analyzing their qualitative behavior is as straightforward as finding their eigenvalues and eigenvectors. Furthermore, there is a rich pool of control theory dedicated to the control of linear systems that can be utilized for such models.

In this chapter, we concern ourselves strictly with learning state-space representations of linear systems. The two primary avenues for this are either to estimate a realization of the state-space matrices directly or to estimate the Markov parameters and use the eigensystem realization algorithm (ERA) to find a state-space realization. Within the state-space matrices approach, we consider the DMD algorithm, and within the Markov parameter approach, we consider one such algorithm that estimates the Markov parameters from a single trajectory. We present the methodology for each algorithm and then prove that they fit inside the marginal likelihood formulation under certain assumptions. The chapter concludes with a set of numerical experiments in which the marginal likelihood is compared to these algorithms on datasets with varying numbers of data and amounts of measurement noise. These experiments will demonstrate the marginal likelihood's robustness to various amounts of uncertainty and reveal where the other algorithms begin to break down.

3.1 Marginal likelihood evaluation of LTI systems

For certain systems, the marginal likelihood is analytically tractable. Here we show the approach in the context of LTI models defined as

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}(\boldsymbol{\theta})\mathbf{x}_k + \mathbf{B}(\boldsymbol{\theta})\mathbf{u}_k + \boldsymbol{\xi}_k, \quad \boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})), \\ \mathbf{y}_k &= \mathbf{H}(\boldsymbol{\theta})\mathbf{x}_k + \mathbf{D}(\boldsymbol{\theta})\mathbf{u}_k + \boldsymbol{\eta}_k, \quad \boldsymbol{\eta}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}(\boldsymbol{\theta})). \end{aligned} \tag{3.1}$$

In this system, if $\mathbf{x}_0$ is either given or Gaussian-distributed, then the system is linear-Gaussian, and the equations in Algorithm 1 have closed-form solutions. Next, we consider two different, but technically equivalent, approaches for evaluating the closed-form marginal likelihood.

3.1.1 State-space approach

The first approach uses the state-space models with a KF to evaluate the marginal likelihood. Following [87], let $\mathbf{m}_k(\boldsymbol{\theta})$ and $\mathbf{P}_k(\boldsymbol{\theta})$ denote the mean and covariance of the Gaussian distribution $\pi(\mathbf{x}_k \mid \mathcal{Y}_{k-1}, \boldsymbol{\theta})$, i.e., $\pi(\mathbf{x}_k \mid \mathcal{Y}_{k-1}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{m}_k(\boldsymbol{\theta}), \mathbf{P}_k(\boldsymbol{\theta}))$, at time t_k. The value of the mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ can be found via a KF. Then, each term in Eq. (2.4) becomes $\pi(\mathbf{y}_k \mid \mathcal{Y}_{k-1}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}_k(\boldsymbol{\theta}), \mathbf{S}_k(\boldsymbol{\theta}))$, where $\boldsymbol{\mu}_k(\boldsymbol{\theta}) = \mathbf{H}(\boldsymbol{\theta})\mathbf{m}_k(\boldsymbol{\theta}) + \mathbf{D}(\boldsymbol{\theta})\mathbf{u}_k$ and $\mathbf{S}_k(\boldsymbol{\theta}) = \mathbf{H}(\boldsymbol{\theta})\mathbf{P}_k(\boldsymbol{\theta})\mathbf{H}(\boldsymbol{\theta})^\top + \boldsymbol{\Gamma}(\boldsymbol{\theta})$. The log marginal likelihood from line 3 of Algorithm 1 becomes

$$\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = -\frac{1}{2}\sum_{k=0}^{n}\Big(\|\mathbf{y}_k - \boldsymbol{\mu}_k(\boldsymbol{\theta})\|^2_{\mathbf{S}_k(\boldsymbol{\theta})} + \log\det\left(\mathbf{S}_k(\boldsymbol{\theta})\right) + d_y \log(2\pi)\Big). \tag{3.2}$$

The form of the marginal likelihood (3.2) resembles a least squares (LS) metric plus a regularization term $\log\det(\mathbf{S}_k)$. The inclusion of this term differs from the typical approach in which regularization is introduced through a prior distribution or heuristic penalty placed on the parameters (e.g., the L_2 norm used in ridge regression). Here, the regularization term has arisen in the likelihood directly from the probabilistic model of the dynamical system and does not require any assumptions on the parameters. The effect of this additional term is a penalty on systems where the estimated output has a large covariance. A large $\mathbf{S}_k$ can arise when the output is sensitive to the model parameters, which is undesirable as it is commonly a sign of overfitting. The regularization term $\log\det(\mathbf{S}_k)$ encodes this dispreference automatically in the likelihood.

3.1.2 Input-output Markov parameter approach

The other common approach to LTI system ID is to first estimate the Markov parameters and then use the ERA [48] to extract a realization of the system matrices $(\mathbf{A}, \mathbf{B}, \mathbf{H}, \mathbf{D})$. The ERA procedure is provided in Appendix B. Here we discuss how the Markov parameters can be obtained from data.

The Markov parameters are obtained by rewriting the linear system (3.1) in a form that removes the states through recursive substitution into the observation equations. Let the Markov parameters be $\mathbf{G}_0 = \mathbf{D}$ and $\mathbf{G}_k = \mathbf{H}\mathbf{A}^{k-1}\mathbf{B}$ for $k = 1, 2, \ldots$, and define a new random variable $\boldsymbol{\nu}_k = \sum_{i=1}^{k} \mathbf{H}\mathbf{A}^{i-1}\boldsymbol{\xi}_{k-i}$. Then we obtain

$$\mathbf{y}_k = \mathbf{H}\mathbf{A}^k\mathbf{x}_0 + \sum_{i=0}^{k} \mathbf{G}_i\mathbf{u}_{k-i} + \boldsymbol{\nu}_k, \boldsymbol{\nu}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}_k), \tag{3.3}$$

where $\boldsymbol{\Lambda}_k = \sum_{i=1}^{k} \mathbf{H}\mathbf{A}^{i-1}\boldsymbol{\Sigma}(\mathbf{H}\mathbf{A}^{i-1})^\top + \boldsymbol{\Gamma}$ if $k > 0$, and $\boldsymbol{\Lambda}_k = \boldsymbol{\Gamma}$ if $k = 0$. The $\boldsymbol{\nu}_k$ are not independent due to their sharing of the process noise $\boldsymbol{\xi}_k$. The covariance is defined as $\boldsymbol{\Lambda}_{j,k} \coloneqq \mathbb{C}\text{ov}[\boldsymbol{\nu}_j, \boldsymbol{\nu}_k]$ for $0 < j < k$. If $j = 0$ and $j \neq k$, then $\boldsymbol{\Lambda}_{j,k} = \mathbf{0}$. Lastly, if $k < j$, then $\boldsymbol{\Lambda}_{j,k} = \boldsymbol{\Lambda}_{k,j}^\top$.

The task then is to learn the Markov parameters for use within the ERA. One can use Bayesian inference again to learn a posterior over the Markov parameters. Assuming $\mathbf{x}_0 = \mathbf{0}$, the likelihood model implied by Eq. (3.3) is

$$p(\mathcal{Y}_n \mid \mathbf{G}_{0:n}, \mathbf{U}_{0:n}) = \mathcal{N}\left(\mathbf{G}_{0:n}\mathbf{U}_{0:n}, \boldsymbol{\Lambda}\right), \tag{3.4}$$

where $\mathbf{G}_{0:n} = \begin{bmatrix}\mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_n\end{bmatrix}$ and

$$\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_0 & \boldsymbol{\Lambda}_{0,1} & \cdots & \boldsymbol{\Lambda}_{0,n} \\ & \boldsymbol{\Lambda}_1 & \cdots & \vdots \\ & & \ddots & \vdots \\ \text{Sym} & & & \boldsymbol{\Lambda}_n \end{bmatrix}, \mathbf{U}_{0:n} = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_n \\ \mathbf{0} & \mathbf{u}_0 & \cdots & \mathbf{u}_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{u}_0 \end{bmatrix}. \tag{3.5}$$

The log of this likelihood (3.4) is equivalent to the state-space log likelihood (3.2) in the following sense: if we evaluate (3.4) with matrices $\mathbf{G}_{0:n}$ and $\boldsymbol{\Lambda}$ determined by a set of state-space matrices, the result equals the evaluation of (3.2) using that same set of state-space matrices.

For comparisons in Section 3.2.2, it is useful to consider a maximum likelihood estimate (MLE) obtained as:

$$\hat{\mathbf{G}}_{0:n} = \underset{\mathbf{G}_{0:n}}{\arg\min} \left\|\text{vec}\left(\mathbf{Y}_{0:n} - \mathbf{G}_{0:n}\mathbf{U}_{0:n}\right)\right\|_{\boldsymbol{\Lambda}}^2, \tag{3.6}$$

where vec$(\cdot)$ denotes the vectorization of a matrix, and $\mathbf{Y}_{0:n} = \begin{bmatrix}\mathbf{y}_0 & \mathbf{y}_1 & \cdots & \mathbf{y}_n\end{bmatrix}$. The generalized LS solution is $\text{vec}(\hat{\mathbf{G}}_{0:n}) = (\mathbf{V}^\top\boldsymbol{\Lambda}^{-1}\mathbf{V})^\dagger\mathbf{V}^\top\boldsymbol{\Lambda}^{-1}\text{vec}(\mathbf{Y}_{0:n})$, where † denotes the pseudo-inverse, and $\mathbf{V}^\top \coloneqq \mathbf{U}_{0:n} \otimes \mathbf{I}_{d_y}$, where $\otimes$ is the Kronecker product.

There are two main issues with the MLE approach: (1) the covariance matrix $\boldsymbol{\Lambda}$ depends on the unknown state-space matrices and is therefore itself unknown, and (2) the Markov parameters are dramatically overparameterized since they are direct functions of the system matrices $(\mathbf{A}, \mathbf{B}, \mathbf{H}, \mathbf{D})$. The usual solutions for the first issue include either removing the weighting and using the standard L_2 norm or estimating a realization of the state-space matrices directly. The second issue of overparameterization arises when $nd_yd_u > {d_x}^2 + d_xd_u + d_xd_y + d_yd_u$, as is typical. As a result, the number of data points (nd_y) is typically smaller than the number of unknowns (nd_yd_u), – except in the case where $d_u = 1$ – and the optimum is not unique. Moreover, any optimum found with this approach, including when $d_u = 1$, will necessarily overfit the data when there is noise.

This second issue motivates approaches that either use multiple trajectories/rollouts with differing inputs[1] to increase the number of data points, or break a single trajectory into multiple ones to decrease the effective number of Markov parameters. Nevertheless, we will show that these existing works use implicit simplifying assumptions and are still at risk of underdetermination for certain input signals. The Bayesian state-space approach, on the other hand, makes all assumptions explicit and is viable regardless of the type of control inputs. These details are futher described in Section 3.2.2.

3.2 Theoretical foundations and analysis

In this section, we analyze two popular system ID methods. The first takes the state-space approach and is known as DMD. The other takes the input-output Markov parameter approach for single-trajectory data. After introducing each method, we derive assumptions that can be placed on the HMM formulation such that the negative log marginal likelihood reduces to that methods's objective function. We additionally include a numerical comparison between the marginal likelihood and the Markov parameter approach that shows the accuracy of each method in estimating the Markov parameters at varying noise and number of data.

3.2.1 Dynamic mode decomposition

DMD is a data-driven method for system ID that is used to identify the 'dynamic modes' of a dynamical system [89]. These modes reveal characteristics such as unstable growth modes, resonance, and spectral properties [78]. DMD is favorable when the system at hand is high dimensional but has some hidden low-dimensional structure, as is the case in many fluids problems. Although extensions of DMD to systems with control inputs have been made, the standard DMD does not consider control inputs. We therefore restrict the analysis of this algorithm to autonomous systems.

The process of DMD starts with organizing a series of measurements at regular time

[1] The inputs must differ by more than a scalar multiplier to avoid underdetermination.

intervals into two matrices: $\mathbf{Y}_{0:n-1}$ and $\mathbf{Y}_{1:n}$. Then, it seeks a linear operator $\mathbf{A}$ which maps the observables from one time step to the next i.e., $\mathbf{Y}_{1:n} = \mathbf{A}\mathbf{Y}_{0:n-1}$. To find $\mathbf{A}$, one simply minimizes the Frobenius norm of $\mathbf{A}\mathbf{Y}_{0:n-1} - \mathbf{Y}_{1:n}$ by solving the least squares problem

$$\mathbf{A} = \arg\min_{\tilde{\mathbf{A}}} \sum_{k=1}^{n} \|\mathbf{y}_k - \tilde{\mathbf{A}}\mathbf{y}_{k-1}\|^2. \tag{3.7}$$

The solution is given by $\mathbf{A} = \mathbf{Y}_{1:n}\mathbf{Y}_{0:n-1}^{\dagger}$, where † denotes the pseudo-inverse.

The method given above may at first appear only applicable to linear systems, but [84] showed that in the nonlinear case, the approximated operator $\mathbf{A}$ and its corresponding modes are approximations to the linear but infinite-dimensional Koopman operator and Koopman modes respectively, thus revealing its applicability to nonlinear systems.

Next we show the LS procedure for DMD can also be *derived* directly from the general probabilistic system (2.5) under certain assumptions.

Theorem 3 (DMD as a maximum likelihood of system (2.5))**.** *Assume a linear model* $\Psi(\mathbf{x}_k, \boldsymbol{\theta}_\Psi) = \boldsymbol{\theta}_\Psi \mathbf{x}_k$*; identity observation operator* $h = \mathbf{I}_{d_x}$*; noiseless measurements* $\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma) = \mathbf{0}$*; and identity process noise* $\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma) = \mathbf{I}_{d_x}$*. Then, the maximum marginal likelihood estimator corresponding to System* (2.5) *is equivalent to the LS objective of the DMD problem* (3.7)*.*

Proof. This result uses a straightforward application of Theorem 2. Without loss of generality, we use the fact that the first measurement is of the initial condition, and therefore we can ignore $\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_0)$. Here, we have an identity observation operator, and therefore the inverse and Jacobian are also the identity. The dynamics are linear and unknown so we can write $\mathbf{A} \equiv \boldsymbol{\theta}_\Psi$. With these substitutions, the log marginal likelihood (2.26) becomes

$$\begin{aligned}\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \sum_{k=1}^{n} \Bigg(\log|\mathbf{I}_{d_x}| - \quad & \frac{1}{2}\|\mathbf{y}_k - \mathbf{A}\mathbf{y}_{k-1}\|^2_{\mathbf{I}_{d_x}} \Bigg) - \\ \frac{nd_x}{2}\log 2\pi - \frac{n}{2}\log|\mathbf{I}_{d_x}|. &\end{aligned} \tag{3.8}$$

After evaluating $\log|\mathbf{I}_{d_x}| = 0$, we arrive at our stated result

$$\log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = -\sum_{k=1}^{n} \frac{1}{2}\|\mathbf{y}_k - \mathbf{A}\mathbf{y}_{k-1}\|^2 - \frac{nd_x}{2}\log 2\pi. \tag{3.9}$$

Clearly, the maximizer of this function is equivalent to the minimizer of (3.7). □

While the invertible measurement operator is not a restrictive assumption because DMD is only concerned with mapping observables and not underlying states, Theorem 3 shows why DMD may not be appropriate for cases where the observations are noisy. This fact has been recognized in the literature and several procedures for rectifying this issue have been proposed. For instance, [41] showed that total least squares is a more appropriate algorithm to identify $\mathbf{A}$ when measurement noise is present, a method known as total DMD (total dynamic mode decomposition (TDMD)). For a full analysis of the total least squares problem, see [32, 102]. We will empirically compare TDMD to our approach in Section 3.3, where we see that it also performs worse than the posterior predictive mean.

In [98], another connection between the Bayesian approach to DMD was developed that infers the Koopman modes and eigenfunctions of the Koopman operator directly, rather than learning the dynamical operator itself. That work showed that when the measurements are noiseless, the maximum likelihood estimate of their Bayesian model, TDMD, and DMD all provide the same estimate. In contrast, here we have provided our result in terms of the underlying hidden state dynamics rather than explicitly assuming observation dynamics.

One benefit of the analysis in our context is that our use of an underlying state-space model makes the framework valid even when the observations cannot be written using a Markovian (zero-lag) model as in Eq. (2.23), which was required for the approach developed in [98]. In fact, this result can be interpreted to indicate that zero-lag DMD is most effective if the observation operator is invertible.

3.2.2 Markov parameter estimation

Next, we consider the Markov parameter estimation approach to learning state-space realizations of stochastic LTI systems, including different approaches for addressing the challenges raised in Section 3.1.2. First, we analyze how existing algorithms approach the problem of unknown covariance $\mathbf{\Lambda}$. Second, we describe how single and multiple rollout resolve the issue of overparameterized Markov parameters that leads to the aforementioned underdetermined system. Finally, we demonstrate how learning the system matrices rather than the Markov parameters leads to faster convergence.

Many existing approaches avoid knowledge of $\mathbf{\Lambda}$ by minimizing an LS objective with an unweighted L_2 norm. Here, we explain this approach as assuming conditionally independent data – given parameters, inputs, and initial conditions – within the setup provided in Section 3.1.2. This assumption sets $\mathbf{\Lambda}_{j,k} = \mathbf{0}$ for $j \neq k$. The resulting estimator is provided below.

Proposition 1. *Assume that $\mathbf{x}_0 = \mathbf{0}$, the inputs $\mathbf{u}_k$ are known, and the outputs $\mathbf{y}_k$ are conditionally independent given $\mathbf{G}_{0:n}$ and $\mathbf{U}_{0:n}$, $\forall k = 0, 1, \ldots, n$. Then, the MLE of an LTI* *system's Markov parameters is*

$$\hat{\mathbf{G}}_{0:n} = \underset{\{\mathbf{G}_i\}_{i=0}^n}{\arg\min} \sum_{k=0}^{n} \left\| \mathbf{y}_k - \sum_{i=0}^{k} \mathbf{G}_i \mathbf{u}_{k-i} \right\|_{\mathbf{\Lambda}_k}^2 . \tag{3.10}$$

Proof. Conditional independence implies $\pi(\mathcal{Y}_n | \mathbf{G}_{0:n}, \mathbf{U}_{0:n}) = \prod_{k=0}^{n} \pi(\mathbf{y}_k | \mathbf{G}_{0:n}, \mathbf{U}_{0:n})$. With $\mathbf{x}_0 = \mathbf{0}$, the marginal distributions follow from the input-output relation in Eq. (3.3) as $\pi(\mathbf{y}_k | \mathbf{G}_{0:n}, \mathbf{U}_{0:n}) = \mathcal{N}(\sum_{i=0}^{k} \mathbf{G}_i \mathbf{u}_{k-i}, \mathbf{\Lambda}_k)$. Then, taking the negative log yields the MLE of the Markov parameters (3.10). □

The independence assumption, however, is not sufficient to convert Eq. (3.6) to the LS objectives commonly used in the literature. The additional assumption that $\mathbf{\Lambda}_k \propto \mathbf{I}$ is needed to convert the weighted norm to a scalar multiple of the L_2 norm. This assumption holds trivially if $d_y = 1$. Otherwise, there is no reasonable assumption to enable $\mathbf{\Lambda}_k \propto \mathbf{I}$. Nevertheless, if one considers an approximate objective where this is assumed to be so, one obtains

$$\hat{\mathbf{G}}_{0:n} = \underset{\{\mathbf{G}_i\}_{i=0}^n}{\arg\min} \sum_{k=0}^{n} \left\| \mathbf{y}_k - \sum_{i=0}^{k} \mathbf{G}_i \mathbf{u}_{k-i} \right\|_2^2 . \tag{3.11}$$

This approximate objective no longer requires knowledge of the system matrices and is used as the basis for a number of approaches [100, 86, 111] for both single and multiple rollout data. This work considers learning from single trajectories, so we focus only on single rollout.

For the single rollout procedure, the data are divided into K overlapping subtrajectories of length $\bar{n}$ such that $n = \bar{n} + K - 2$. To address underdetermination, one must also require $\bar{n} < \frac{n+1}{d_u}$. After dividing the single trajectory into multiple trajectories, the final output of each subtrajectory follows the same form as Eq. (3.3),

$$\mathbf{y}_k = \mathbf{H}\mathbf{A}^{\bar{n}-1}\mathbf{x}_{k-\bar{n}+1} + \sum_{i=0}^{\bar{n}-1} \mathbf{G}_i \mathbf{u}_{k-i} + \boldsymbol{\nu}_k, \tag{3.12}$$

for $k = \bar{n}, \ldots, n$. Assuming that the inputs are zero-mean, then the expected value of each $\mathbf{x}_{k-\bar{n}+1}$ is zero with respect to the inputs and noise variables. This is sometimes used as justification to eliminate $\mathbf{x}_{k-\bar{n}+1}$ from the estimation problem [71], and we therefore also adopt this ansatz. Adding the approximation that $\mathbf{\Lambda}_k \propto \mathbf{I}$ yields the following optimization problem

$$\hat{\mathbf{G}}_{0:\bar{n}-1} = \underset{\{\mathbf{G}_i\}_{i=0}^{\bar{n}-1}}{\arg\min} \sum_{k=\bar{n}}^{n} \left\| \mathbf{y}_k - \sum_{i=0}^{\bar{n}-1} \mathbf{G}_i \mathbf{u}_{\bar{n}-i} \right\|_2^2 . \tag{3.13}$$

The number of unknown Markov parameters is now only $\bar{n}$ rather than $n+1$, mitigating the problem of underdetermination. The LS solution is $\hat{\mathbf{G}}_{0:\bar{n}-1} = \mathbf{Y}_{\bar{n}:n}\bar{\mathbf{U}}^{\dagger}_{\bar{n}:n}$, where

$$\bar{\mathbf{U}}_{\bar{n}:n} = \begin{bmatrix} \mathbf{u}_{\bar{n}} & \mathbf{u}_{\bar{n}+1} & \cdots & \mathbf{u}_n \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_K \end{bmatrix}. \tag{3.14}$$

This formulation is equivalent to the slightly different form provided by [71][2]. Although the system of equations now has more equations than unknowns for proper choice of $\bar{n}$, the system can still suffer from underdetermination for certain input signals. For example, sinusoidal inputs generate a $\bar{\mathbf{U}}_{\bar{n}:n}$ with rank of only 2. Additionally, each estimated data point requires exactly $\bar{n}$ inputs. Consequently, the outer sum skips the first $\bar{n}$ data points since the inputs $\mathbf{u}_0, \mathbf{u}_{-1}, \ldots$ are typically assumed unknown.

This L_2 optimization problem (3.13) is equivalent to the weighted L_2 optimization problem (3.10) under additional assumptions stated in Proposition 2.

Proposition 2. *Assume the assumptions of Proposition 1 are met and additionally that* $\sum_{i=\bar{n}}^{k} \mathbf{G}_i \mathbf{u}_{k-i} = \mathbf{0}$, $\mathbf{A}^k \mathbf{\Sigma}(\mathbf{A}^k)^\top = \mathbf{0}$ *for* $k \geq \bar{n}$, *and* $\mathbf{\Lambda}_{\bar{n}} \propto \mathbf{I}$. *If the first* $\bar{n}$ *outputs are discarded, then the MLE in Eq.* (3.10) *is equivalent to the estimator in Eq.* (3.13).

Proof. If $\sum_{i=\bar{n}}^{k} \mathbf{G}_i \mathbf{u}_{k-i} = \mathbf{0}$, then in Eq. (3.10) $\sum_{i=0}^{k} \mathbf{G}_i \mathbf{u}_{k-i} = \sum_{i=0}^{\bar{n}-1} \mathbf{G}_i \mathbf{u}_{k-i}$. Lastly if $\mathbf{A}^k \mathbf{\Sigma}(\mathbf{A}^k)^\top = \mathbf{0}$ for $k \geq \bar{n}$, then $\mathbf{\Lambda}_k = \sum_{i=1}^{\bar{n}} \mathbf{H}\mathbf{A}^{i-1}\mathbf{\Sigma}(\mathbf{H}\mathbf{A}^{i-1})^\top + \mathbf{\Gamma} = \mathbf{\Lambda}_{\bar{n}}$, for $k \geq \bar{n}$. By assumption, $\mathbf{\Lambda}_{\bar{n}} \propto \mathbf{I}$, so the weighted norm is equivalent to the standard L_2 norm. □

The assumptions $\sum_{i=\bar{n}}^{k} \mathbf{G}_i \mathbf{u}_{k-i} = \mathbf{0}$ and $\mathbf{A}^k \mathbf{\Sigma}(\mathbf{A}^k)^\top = \mathbf{0}$ for $k \geq \bar{n}$ can be satisfied if the system has finite impulse response. Alternatively, these two assumptions can be achieved asymptotically under the much weaker assumption that $\rho(\mathbf{A}) < 1$, where $\rho(\cdot)$ denotes the spectral radius of a matrix. Such a result is given in Proposition 3.

Proposition 3. *Let* $\rho(\mathbf{A}) < 1$, *the inputs* $\mathbf{u}_k$ *be independent realizations of a real-valued random variable, and* $\mathbf{\Lambda}_{\bar{n}} \propto \mathbf{I}$. *As* $\bar{n} \to \infty$, *the negative log likelihood of Eq.* (3.10) *approaches the single rollout LS objective of Eq.* (3.13) *with probability 1. Moreover, it converges at least linearly.*

The proof is in Appendix C.

For systems where $\mathbf{\Lambda}_k \propto \mathbf{I}$ approximately holds for $k > \bar{n}$, this result implies that even if $\mathbf{\Lambda}_k$ varies, a good approximation can be achieved for reasonably small $\bar{n}$. However, for systems where $\rho(\mathbf{A}) \geq 1$, the conditions of this proposition no longer hold, e.g., periodic systems have $\rho(\mathbf{A}) = 1$.

[2] The unlabeled equation following Eq. 5 in [71].

3.2.2.1 Numerical comparison

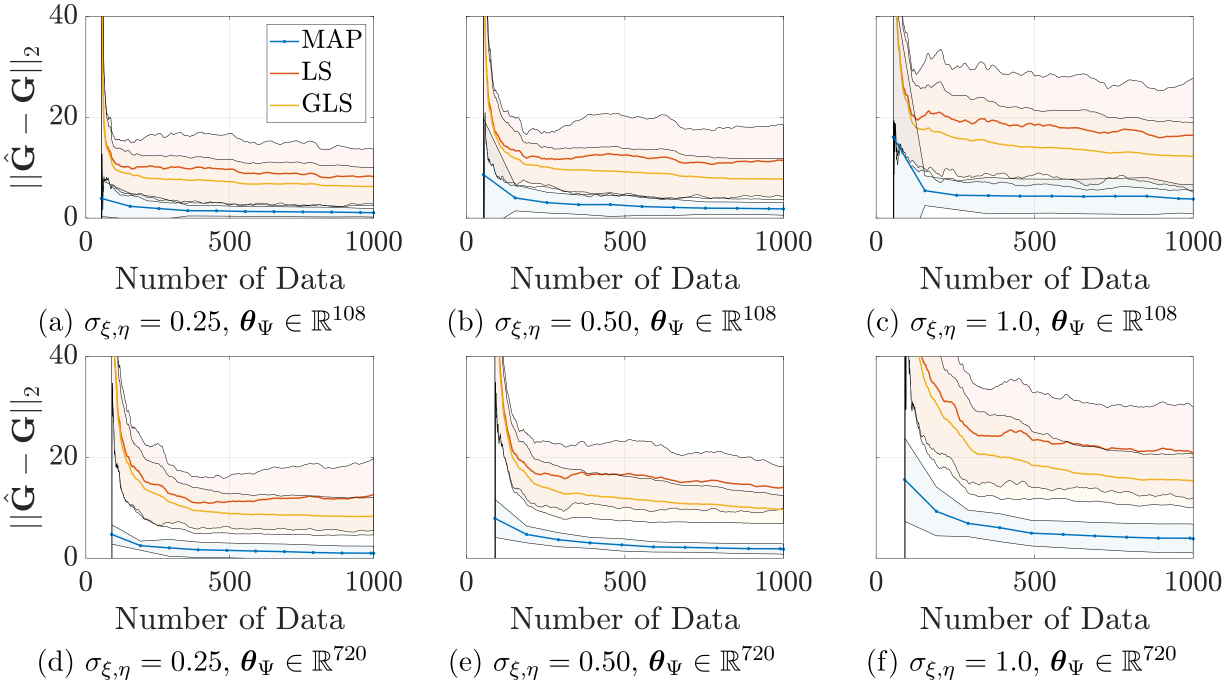
We now perform a comparison between three approaches using the same numerical experiment from [71]. The first approach is the LS approach (3.13) used in single rollout in [71]. In the second approach, we assume that $\mathbf{\Lambda}_k$ is given, e.g., by an oracle, so we minimize the same objective as Eq. (3.13), but with a different weighted norm

$$\hat{\mathbf{G}}_{0:\bar{n}-1} = \underset{\{\mathbf{G}_i\}_{i=0}^{\bar{n}-1}}{\arg\min} \sum_{k=\bar{n}}^{n} \left\| \mathbf{y}_k - \sum_{i=0}^{\bar{n}-1} \mathbf{G}_i \mathbf{u}_{k-i} \right\|_{\mathbf{\Lambda}_k}^2 . \tag{3.15}$$

This objective is henceforth referred to as generalized least squares (GLS). Finally, we compare with the MAP estimate of the Bayesian approach described in Section 3.1. The state-space approach is used numerically, though it is theoretically equivalent to the input-output approach.

This experiment was performed as follows. A random state-space system was generated by independently sampling the entries of $\mathbf{H}$ and $\mathbf{D}$ from $\mathcal{N}(0, 1/d_y)$ and of $\mathbf{B}$ from $\mathcal{N}(0, 1/d_x)$. The dimensions of the system were $d_y = 2$, $d_x = 5$, and $d_u = 3$, and the noise covariances were $\mathbf{\Sigma} = \sigma_\xi^2 \mathbf{I}$ and $\mathbf{\Gamma} = \sigma_\eta^2 \mathbf{I}$. To highlight the difference between LS and GLS, $\mathbf{A}$ was set as the identity matrix to ensure that $\rho(\mathbf{A}) = 1$ and consequently that the covariance $\mathbf{\Lambda}_k$ would vary significantly over time. To avoid having priors give the Bayesian algorithm an edge, improper uniform priors were placed on the state-space matrices, and only weakly informative priors of half-$\mathcal{N}(0, 1)$ were placed on the parameters σ_ξ and σ_η to enforce positivity and improve convergence as recommended in [31]. Then, data were generated by simulating the system with inputs sampled from a standard normal, i.e., $\mathbf{u}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. For this experiment, a subtrajectory length of $\bar{n} = 18$ was considered for a total of $d_y d_u \bar{n} = 108$ parameters. The Markov parameters were estimated using the first K subtrajectories of the simulated trajectory where $K = d_u \bar{n}, \ldots, 2000$. Optimizing over the posterior is significantly more expensive than solving a linear LS problem, so for computational feasibility, this optimization was only performed at $K = d_u \bar{n}, d_u \bar{n} + 100, \ldots, 2000$. Since the LS methods do not use the first $\bar{n}$ data points, these data were also removed from MAP estimation for consistency.

To assess the accuracy of the estimate, the spectral norm of the estimation error $\|\hat{\mathbf{G}}_{0:\bar{n}-1} - \mathbf{G}_{0:\bar{n}-1}\|_2$ was evaluated. This experiment was repeated 50 times, and the top row of Fig. 3.1 shows the average error norm plotted as a solid line with a shaded region representing plus-minus one standard deviation against the number of data used in each estimate. The figure also compares the LS, GLS, and MAP estimates at various noise levels $\sigma_{\xi,\eta} = 1/4, 1/2, 1$. The weighting used by GLS yielded lower error mean and variance at all noise levels compared to the LS estimate. The MAP estimate produced the lowest mean error and error variance of all by a significant margin. The same experiment was repeated with larger dimensions of $d_y = 8$, $d_x = 10$, and $d_u = 5$ for a total 720 parameters, and the results are shown in the bottom row of Fig. 3.1. Again, the ranking of the performance of the estimates is the same, but in the larger system, the improvement achieved by the MAP estimate is even greater. We also observe very little degradation of the MAP estimate when the system dimensions increase other than in convergence rate, which can be attributed to the greater number of parameters. The results of these experiments illustrate the performance costs incurred by adding simplifying assumptions into the objective.



(a) $\sigma_{\xi,\eta} = 0.25$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{108}$
(b) $\sigma_{\xi,\eta} = 0.50$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{108}$
(c) $\sigma_{\xi,\eta} = 1.0$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{108}$

(d) $\sigma_{\xi,\eta} = 0.25$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{720}$
(e) $\sigma_{\xi,\eta} = 0.50$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{720}$
(f) $\sigma_{\xi,\eta} = 1.0$, $\boldsymbol{\theta}_\Psi \in \mathbb{R}^{720}$

Figure 3.1: A comparison of the spectral norm of the Markov parameters estimation error for $\bar{n} = 18$ using the LS, GLS, and MAP estimates at varying noise levels. The lines represent mean error values and the shaded regions represent plus-minus one standard deviation.

3.3 Numerical experiments

In this section, we compare the negative log marginal likelihood to DMD and Markov parameter estimation methods on a number of numerical experiments. Specifically, we will consider three different versions of a pendulum with varying numbers of data and levels of noise. The first example is a linear pendulum, the second example is a nonlinear pendulum, and the third example is a linear pendulum with control inputs. In the first two examples, we compare to DMD and in the third example, we compare to the Markov parameter approach. In each example, we will show that while these other methods break down quickly when measurement noise is added, the estimates produced by the marginal likelihood degrade more gradually as different sources of uncertainty are added such as measurement noise, model mismatch, and sparse data. The code for these experiments can be found at https://github.com/ngalioto/BayesID.

3.3.1 Linear pendulum, linear model

For this experiment, we consider learning a linear model under an identity observation operator $h = \mathbf{I}$ when the truth model is also linear. We show that that the proposed probabilistic approach is more robust to sparse observations and measurement noise than the LS-based DMD and TDMD.

Consider the linear model (1.7) for which the exact propagator is

$$\mathbf{x}_k = \exp\left(\begin{bmatrix} 0 & 1 \\ -\frac{g}{L} & 0 \end{bmatrix} \Delta t\right)\mathbf{x}_{k-1}, \qquad \mathbf{x}_0 = \begin{bmatrix} 0.1 \\ -0.5 \end{bmatrix} \tag{3.16}$$

where $g = 9.81$ is the acceleration due to gravity and $L = 1$ is the length of the pendulum.

We are learning an unknown linear model $\mathbf{A}(\boldsymbol{\theta}_\Psi)$, and we assume that the process noise and measurement noise is also uncertain. Under this setting, System (2.5) becomes

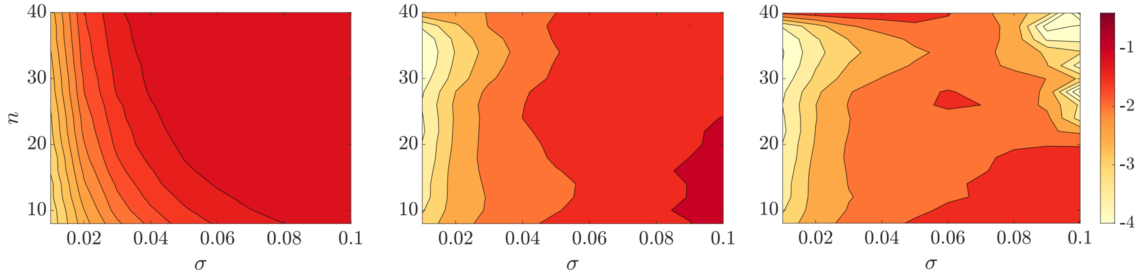
$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}(\boldsymbol{\theta}_\Psi)\mathbf{x}_{k-1} + \boldsymbol{\xi}_k, \quad \boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)) \\ \mathbf{y}_k &= \mathbf{x}_k + \boldsymbol{\eta}_k, \quad \boldsymbol{\eta}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)), \end{aligned} \tag{3.17}$$

for $k = 1, \ldots, n$ where

$$A(\boldsymbol{\theta}_\Psi) = \begin{bmatrix} \theta_1 & \theta_2 \\ \theta_3 & \theta_4 \end{bmatrix}, \quad \boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma) = \theta_5\mathbf{I}_2, \quad \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma) = \theta_6\mathbf{I}_2. \tag{3.18}$$

Because this setup is precisely the one corresponding to DMD, we seek to compare the performance of our approach to DMD and TDMD. Our comparison takes the form of average performance over 500 different realizations of the datasets for different combinations of training data sizes n and true measurement noise standard deviation σ. The data points are spread out over a simulation period of four seconds, so increasing n indicates increasing density of data per time.

The results, shown in Fig. 3.2, provide ($\log_{10}$) ratios of the expected error of the posterior predictive mean (computed with 1000 posterior samples) to the (T)DMD estimators. The squared errors were calculated only at the times of observations, and the largest MSE from each dataset for each algorithm was discarded to prevent biasing from outliers. We see that the biggest gains in using the probabilistic Bayesian approach come in the low noise regime. At first this seems surprising, but in the low noise regime, this is likely the result of the scale of the errors being so small. As the noise increases, we see the ratio increasing even though we would expect DMD to break down much more quickly than the Bayesian approach. The reason this occurs is because DMD predictions decay to zero after a certain level of noise (shown in Fig. 3.2a), effectively placing an upper bound on the MSE of the algorithm. Regardless, the contour plots show that the Bayesian algorithm outperforms both DMD and TDMD at every measurement frequency and noise pair considered.

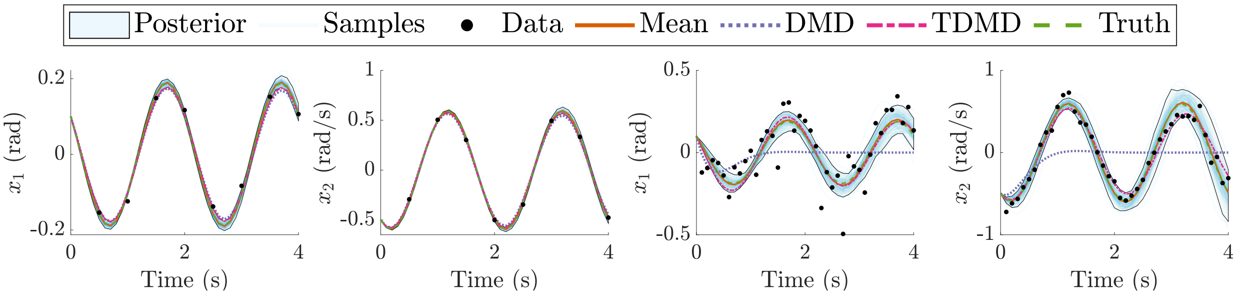


(a) DMD Pred. MSE (b) Bayes/DMD Pred. (c) Bayes/TDMD Pred.

Figure 3.2: $\log_{10}$ ratio of the MSE obtained by the proposed Bayesian approach to that obtained by (T)DMD for the linear pendulum model. In all cases, this value is less than zero signifying that our proposed approach outperforms (T)DMD in all cases considered. Also observe in the high noise regime, TDMD can begin to lose stability.

Next we provide a detailed look at two specific points on these contour plots to demonstrate the mechanism by which DMD/TDMD decline. The first case is a low-noise/sparse-data case of $\sigma = 10^{-2}$ and $n = 8$, and the second case is for a higher noise case $\sigma = 10^{-1}$ with more data $n = 40$.

The reconstruction results for each state are compared in Fig. 3.3. The prediction (forecasting) results for just the second state are shown in Fig. 3.4. The mean here refers to the posterior predictive mean given in 2.19. The shaded area represents the region between the 97.5th and 2.5th quantiles of the Bayesian posterior. In the low noise case, we see that all three algorithms perform essentially equally – though the DMD-based approaches slightly underestimate the amplitude. In other words, even in the case for which DMD was designed to perform well, the Bayesian approach performs slightly better. In the high noise case, we see that the TDMD predictions become completely out of phase with increasingly small amplitude, and the DMD estimator smooths out the data too much and rapidly converges to zero. Not only does the Bayesian approach provide the most accurate estimate, but it also gives a quantification of the certainty of its estimate in the form of its posterior, which (T)DMD is unable to provide.

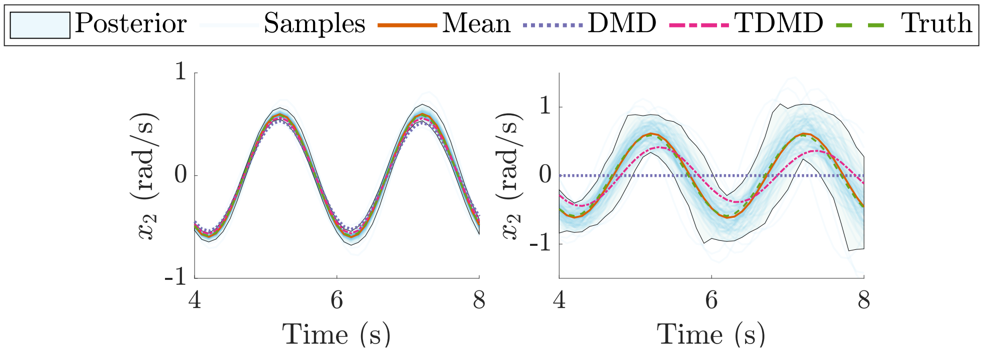


(a) $\sigma = 10^{-2}, n = 8$ (b) $\sigma = 10^{-2}, n = 8$ (c) $\sigma = 10^{-1}, n = 40$ (d) $\sigma = 10^{-1}, n = 40$

Figure 3.3: Comparison of reconstruction error amongst the Bayesian and (T)DMD algorithms for the linear pendulum truth model. Top row corresponds to a low-noise/sparse-data case and the bottom row corresponds to a high-noise/dense-data case. Left column corresponds to the first state (angular position) and right column corresponds to the second state (angular velocity). For low-noise, the algorithms perform similarly; however, the (T)DMD approaches underestimate the amplitude. For the high-noise case, DMD fails and TDMD misfits the amplitude. The Bayesian approach is able to recognize greater uncertainty for the high-noise case.

Fig. 3.5 shows the estimated eigenvalues of the system by the Bayesian and (T)DMD algorithms. In the low noise case, Fig. 3.5a shows that the Bayesian approach is slightly more accurate than the (T)DMD approaches, though they all perform well. For the high noise case, Fig. 3.5b shows that DMD is unable to provide a reasonable estimate of the eigenvalues. TDMD gives a close estimate, but the estimated eigenvalues are too far in the left-hand plane, causing the gradual decay seen in Fig. 3.4. The Bayesian estimate lies almost exactly on top of the truth.

Finally, Fig. 3.6 shows the marginal and joint distributions of the process and measurement noise variances for these two cases. The process noise is very close to zero because we are using a linear model for a linear system, and thus the system learns that the dynamics can be captured exactly. These plots also indicate that we have learned the measurement noise, as the mode aligns closely with the true value shown in red. Note also that the joint distribution in this figure shows that the two noise variances are negatively correlated, conveying the fact that the estimator does not yet have enough data to determine if the model is off and the measurements are accurate, or if the model is accurate and the measurements are noisy. As more data are collected, however, one of these scenarios can usually be ruled out and the distribution becomes unimodal.



(a) $x_2, \sigma = 10^{-2}, n = 8$ (b) $x_2, \sigma = 10^{-1}, n = 40$

Figure 3.4: Comparison of prediction error amongst the Bayesian and (T)DMD algorithms for the linear pendulum truth model. Left panel corresponds to a low-noise/sparse-data case and the right panel corresponds to a high-noise/dense-data case. Both panels show the angular velocity of the pendulum. For low-noise, the algorithms perform similarly. For the high-noise case, DMD fails and TDMD can be seen to be out of phase and have a smaller amplitude. The Bayesian approach is able to recognize greater uncertainty for the high-noise case.

3.3.2 Nonlinear pendulum, linear model

Next we consider a problem where the model class within which we are learning does not encompass the true underlying dynamical system. This is the most realistic situation that would be encountered in practice, and avoids the so-called "inverse crime" [105].

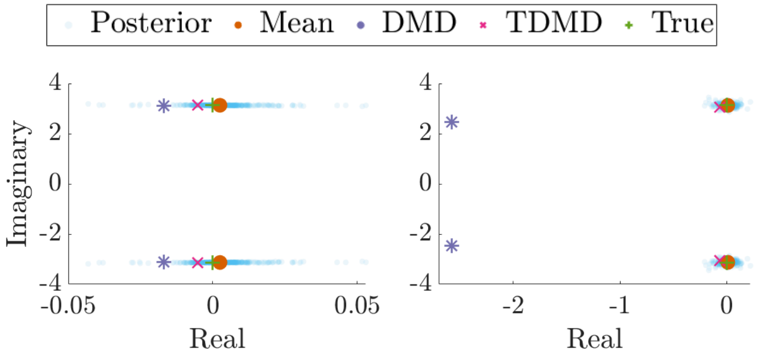
Consider a nonlinear pendulum

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{L}\sin(x_1) \end{bmatrix}, \quad x_0 = \begin{bmatrix} 2.5 \\ 0 \end{bmatrix} \tag{3.19}$$

to be the truth model. We have changed the initial condition to ensure that we are operating in the nonlinear regime.

The learning setup is identical to that provided in Section 3.3.1; we learn a linear model, and the same validation experiments are performed. These experimental results are shown in Fig. 3.7. We are able to clearly see here that, although the three algorithms are comparable in the low noise regime, the strength of the Bayesian approach increases with the measurement noise. A discussion on why (T)DMD may outperform the mean estimator from the Bayesian approach in the low noise regime is provided later in Section 3.3.2.1.

We again present more detailed results for two representative cases. Both cases have $n = 24$ data points, but the first case is a low noise case of $\sigma = 10^{-1}$ and the second case is a higher noise case of $\sigma = 1$.

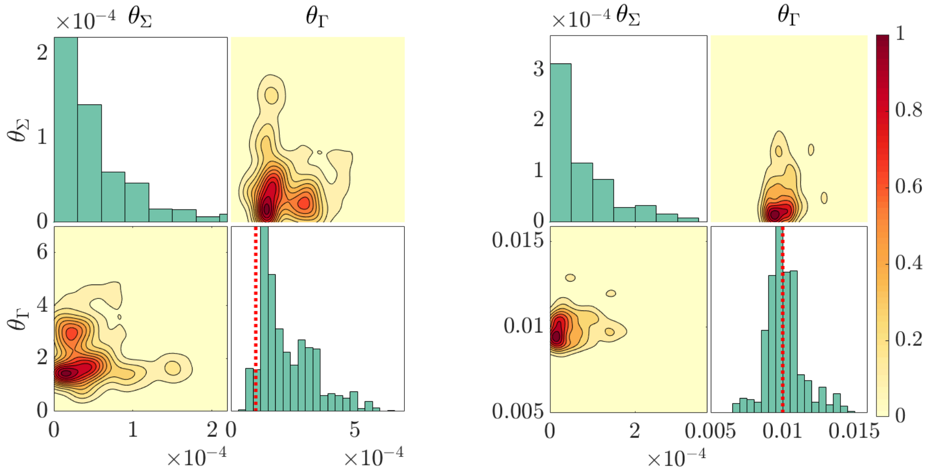


(a) Lower n, σ (b) Higher n, σ

Figure 3.5: Eigenvalue distributions for the estimators of the linear pendulum. The mean value here represents the mean of the eigenvalues. All three algorithms come very close to learning the true eigenvalues in the low noise case, but Bayes is able to outperform the other two in both the high and low noise cases. DMD achieves significant error when the data are noisy.

The resulting reconstructions are shown in Fig. 3.8, and the predictions are given in Fig. 3.9. Note that the variances of the posterior distributions in both cases grow much more quickly than in either of the linear pendulum examples as a consequence of increased model uncertainty (process noise). The posterior distribution can therefore be used to qualitatively assess not only how informative the data are, but also how appropriate the chosen model is for the system at hand. In the low noise case, the performances of the three estimates are virtually indistinguishable, once again demonstrating that even in systems that are ideal for (T)DMD, there is no loss of performance when using the Bayesian estimator. In the high noise case, DMD struggles with noisy measurements and settles on quickly decaying to zero, similar to what we observed in the linear case. TDMD, on the other hand, comes closer but is noticeably out of phase with the truth. The Bayesian approach is able to reconstruct the signal very closely, at least within the constraints imposed by using a linear model.

Next we investigate what the Bayesian approach learns for the process and measurement noise in the case where there is a model error. The marginal and joint posterior distributions for both measurement noise cases are shown in Fig. 3.10. We observe that in the low noise case 3.10a, the joint distribution is bimodal. The smaller mode corresponds to a model with low process noise and high measurement noise, and the larger mode corresponds to a model with high process noise and low measurement noise. The algorithm has effectively uncovered that the data can be explained in one of two ways: either the model fits the true system well, but the data are very noisy, or the measurement noise is low and the model is not capable of properly capturing the dynamics. In this case, the latter is true and is also the much more likely option based on posterior density. For the high noise case 3.10b, the joint distribution



(a) Lower n, σ

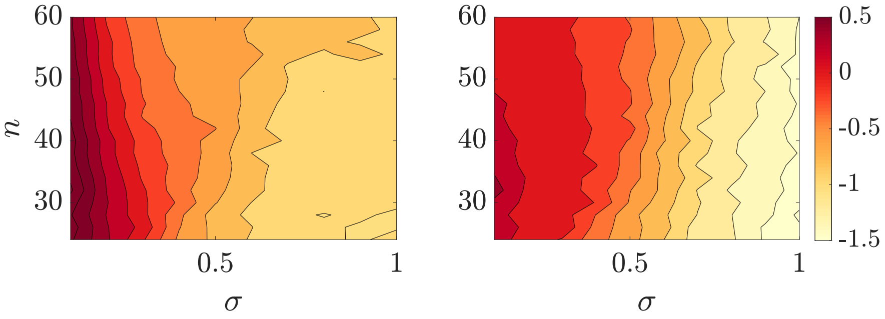
(b) Higher n, σ

Figure 3.6: Marginal and joint posterior distributions of the process and measurement noise variance parameters during the recovery of the linear pendulum. In the left panel, 8 measurements are not enough for the Bayesian estimator to unambiguously determine the measurement noise, but its best guess (the mode) aligns with the truth. On the right, we see that 40 measurements are enough to define a distinct mode within the joint distribution, which also aligns with the truth.

is unimodal, conveying the possibility of only one process-measurement noise pairing. Once again, the modes of both the measurement noise marginal distributions align closely with the truth shown in red. Finally, we see that the process noise magnitudes in both cases are much larger than those seen in the linear pendulum examples (Fig. 3.6) as a consequence of trying to capture nonlinear dynamics with a linear model.

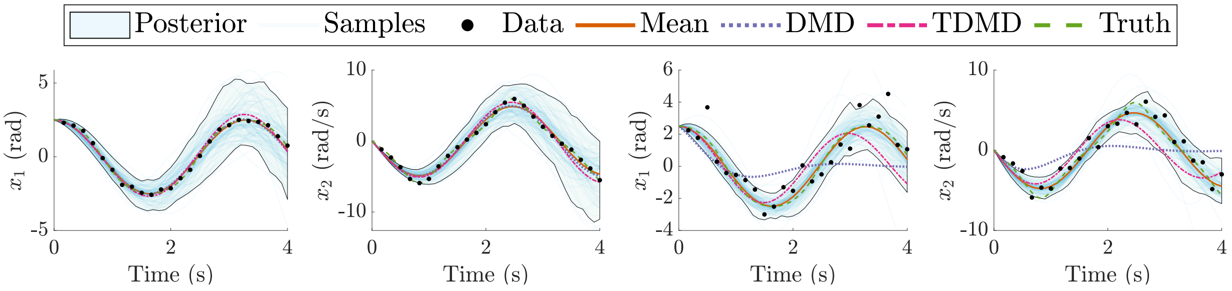
3.3.2.1 Discussion on diagnostics

One of the strengths of the Bayesian approach is that it separates the learning stage from the decision making stage, so if the initial decision rule yields an unsatisfactory estimate, one can go back and analyze the posterior distribution to devise an improved decision rule. It was noted earlier in Fig. 3.7 that the average MSE of (T)DMD is lower than that of the average MSE of the Bayesian estimator over 500 datasets when the measurement noise is low. This observation likely implies that there is a better decision rule that can be used to achieve performance at least as strong as DMD. To understand how to best select a point from the posterior to be our estimate, we first look at the posterior over the states. Fig. 3.11 shows samples from the posterior predictive distribution for a single dataset containing $n = 26$ measurements with noise standard deviation of $\sigma = 0.1$. The mean deviates from the truth near the peaks and valleys of the trajectory between about 2.5 and 4s. This is the same



(a) Bayes/DMD Pred. MSE (b) Bayes/TDMD Pred. MSE

Figure 3.7: Contours of the ratios from the nonlinear pendulum experiments. The experiment is the same as in Fig. 3.2. A detailed explanation for the low noise regime where it appears (T)DMD outperforms Bayes is given in Section 3.3.2.1.

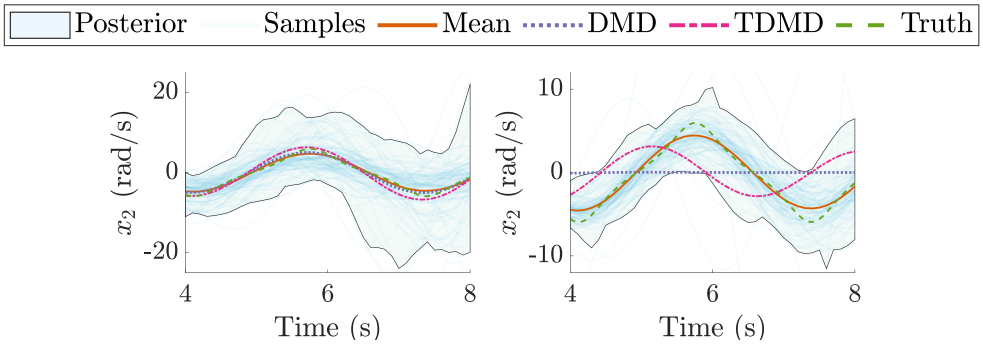


(a) $n = 24$, $\sigma = 10^{-1}$ (b) $n = 24$, $\sigma = 1$

Figure 3.8: Reconstruction performance for low-noise (top row) and high-noise (bottom row) datasets for the nonlinear pendulum using a linear model. All three estimates capture the truth closely in the low noise case, but only the Bayesian algorithm performs well (it is in phase and approximately the correct amplitude) for the high noise case.

location in which the posterior appears to be significantly spread in possible predictions. This presence of significant outliers is a result of the bimodal noise distribution previously discussed. Furthermore, it is clear that the mean is not a good estimator in the case of bimodal distributions; however, we see that there exists a mode in alignment with the truth. Upon this realization, we can then craft a decision rule that selects this mode rather than the mean for improved performance. In this case, the mode-based rule would result in the Bayesian approach being 1.3 times better than the TDMD estimator. Moreover, this entire analysis can be done *a posteriori* and therefore uses no additional assumptions or requirements on our approach.

We also note that the effect this has on the MSE ratio appears more strongly in this nonlinear case for two reasons. The first reason is that the higher process noise due to the model error and low measurement noise can create a bimodal distribution because of the alternate possibility of a good model with noisy data as shown earlier. The second reason is



(a) $n = 24$, $\sigma = 10^{-1}$ (b) $n = 24$, $\sigma = 1$

Figure 3.9: Comparison shown here is the same as in Fig. 3.4, but this time for a nonlinear pendulum truth model. In the low noise case, the estimates are all visually aligned with the truth. In the high noise case, DMD fails and TDMD falls out of phase, but the Bayesian algorithm remains robust and produces an accurate estimate.

that the ratio of process noise to measurement noise is higher than that in the linear case. As we have shown in Theorem 3, the (T)DMD approaches effectively assume the existence of process noise but no measurement noise. In cases where the linear and nonlinear models are mismatched this becomes a better assumption.

In summary, for cases where the model error can be significant, a non-mean estimator should be extracted from the Bayesian posterior. This estimator should be chosen by considering the bimodality of the learned process/measurement noise estimator, and can often be the peak of one of the modes. If this is done (it is an *a posteriori* procedure), we have seen that it yields improved performance compared to (T)DMD.

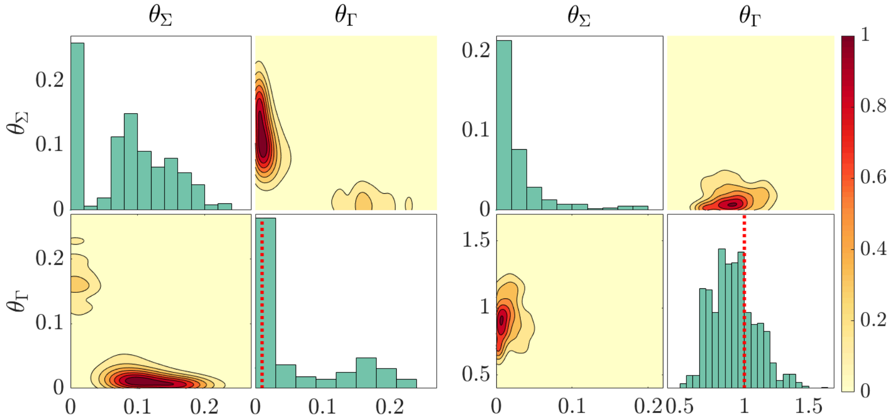
3.3.3 Linear pendulum with control

This last example demonstrates that as the noisiness and sparsity of data increase, the performance of the Bayesian method decays at a slower rate than the LS+ERA method from Section 3.2.2.

Consider a pendulum with unit length and mass with damping and random inputs. The dynamics of such a system are given as

$$\begin{aligned} \mathbf{x}_{k+1} &= \text{expm}\left(\begin{bmatrix} 0 & 1 \\ -\frac{g}{L} & -1 \end{bmatrix} \Delta t\right) \mathbf{x}_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k, \\ y_k &= \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}_k + \eta_k; \qquad \mathbf{x}_0 = \mathbf{0}, \end{aligned} \tag{3.20}$$

where expm is the matrix exponential and the inputs are Gaussian-distributed as $u_k \sim \mathcal{N}(0, \Delta t)$. The damping term is included to ensure that the $\mathbf{A}$ matrix is asymptotically stable



(a) True $\sigma_\Gamma = 0.1$ (b) True $\sigma_\Gamma = 1.0$

Figure 3.10: Marginal and joint posterior distributions of the process and measurement noise variance parameters during the recovery of the nonlinear pendulum. In the left panel, the joint distribution is bimodal, offering two possible models with the true case being strongly preferred. In the right panel, all of the distributions are unimodal and in alignment with the truth.

at all Δt considered. This damping term ensures $\rho(\mathbf{A}) < 1$, so the theoretical requirements in [71] for the LS+ERA are met.

Data were collected from this system over a 20s training period at various timesteps and noise levels. For this experiment, timesteps of $\Delta t = 0.10, 0.15, \ldots, 0.50$ and noise ratios of $\sigma = 0.00, 0.025, \ldots, 0.200$ were considered. Here, the noise ratio is defined as $\sigma := \sigma_{\boldsymbol{\eta}} / \max(\mathbf{x}[1])$, where $\sigma_{\boldsymbol{\eta}}$ is the standard deviation of the measurement noise. For each noise-timestep pair, 100 realizations of data were generated, and every realization was trained on separately such that each method estimated a set of 100 models per pair. Note that since the inputs are random, the system behavior is also random and the noise ratio of each dataset therefore varies, even within a given noise-timestep pair.

The model parameterization is

$$\begin{aligned} \mathbf{x}_0 = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}; \;\; \mathbf{x}_{k+1} &= \begin{bmatrix} \theta_3 & \theta_5 \\ \theta_4 & \theta_6 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} \theta_7 \\ \theta_8 \end{bmatrix} u_k + \boldsymbol{\xi}_k, \qquad & \boldsymbol{\xi}_k &\sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \theta_{11} & 0 \\ 0 & \theta_{12} \end{bmatrix}\right) \\ y_k &= \begin{bmatrix} \theta_9 & \theta_{10} \end{bmatrix} \mathbf{x}_k + \eta_k, & \eta_k &\sim \mathcal{N}(0, \theta_{13}). \end{aligned} \tag{3.21}$$

The priors half-$\mathcal{N}(0, 10^{-6})$ and half-$\mathcal{N}(0, 1)$ were placed on $\boldsymbol{\theta}_\Sigma$ and $\boldsymbol{\theta}_\Gamma$, respectively, and an improper uniform prior was placed on the remaining 10 parameters. The LS+ERA and MAP estimates were compared with respect to the $\log_{10}$ of the average MSE at each noise-timestep pair. For a given value of σ and Δt, let $i = 1, \ldots, 100$ index the data realizations.

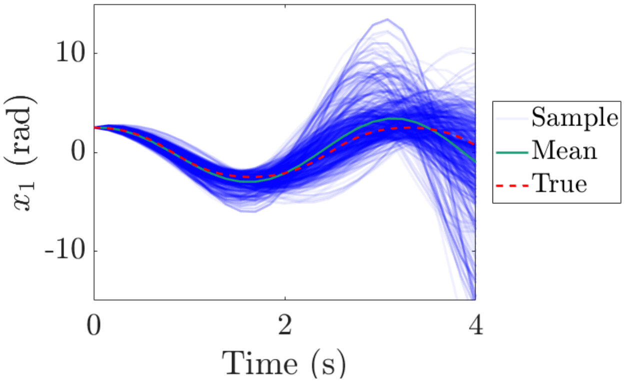
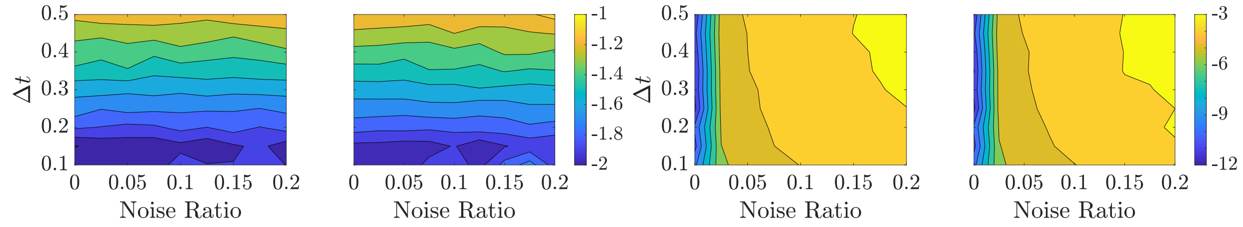


Figure 3.11: Posterior samples from the dataset with $n = 26$, $\sigma = 0.1$ that produced the worst mean estimate out of the 500 with respect to MSE. This figure illustrates that the mean deviates from the truth at the extrema of the curve where samples are skewed toward larger magnitudes. Using a decision rule that selects the mode here would give a much better estimate.

Then, the MSE on the ith dataset is defined as $\frac{1}{n}\sum_{k=1}^{n}(x_k[1] - \hat{y}_k(\boldsymbol{\theta}_i))^2$, where $\boldsymbol{\theta}_i$ represents the parameter estimate on the given dataset. Additionally, the MSE on a testing period of 20s beyond the training data was also calculated over time indices $k = n+1, \ldots, 2n$. There were a handful of outliers in the LS+ERA MSE that significantly skewed the average value, so only the lowest 99 MSE values were used to compute the average MSE of the LS+ERA estimate. The average MSE of the MAP estimate retains all 100 MSE values.

Contour plots of the $\log_{10}$ average MSE of the LS+ERA and MAP estimates are given in Fig. 3.12. We observe that the LS+ERA performance degrades most significantly as the timestep increases as a consequence of having fewer data available for estimation. The MAP estimate, on the other hand, appears to degrade more as the noise ratio is decreased, but its degradation due to increasing timestep is of similar magnitude. The slower degradation along the timestep axis suggests that the Bayesian approach has low data requirements, especially when the data are low-noise. Based on the colorbars of each set of plots, the MAP estimate gives at least an order of magnitude of improvement over the LS+ERA estimate.

To get a better understanding of how each method performs when the data are noisy or sparse, two datasets from different (noise-timestep) pairs were chosen on which to examine the estimated output from each method. The selected pairs are the high noise, low sparsity case (0.20, 0.1), and the low noise, high sparsity case (0.00, 0.5). These two pairs were chosen so that the effect of high noise and high sparsity could be studied separately. Within each pair, the data realization chosen was the dataset for which the LS+ERA method had the lowest training MSE so that its peformance was fairly represented. The estimated outputs of the two estimates are shown in Figs. 3.13a and 3.13b for the high noise and high sparsity



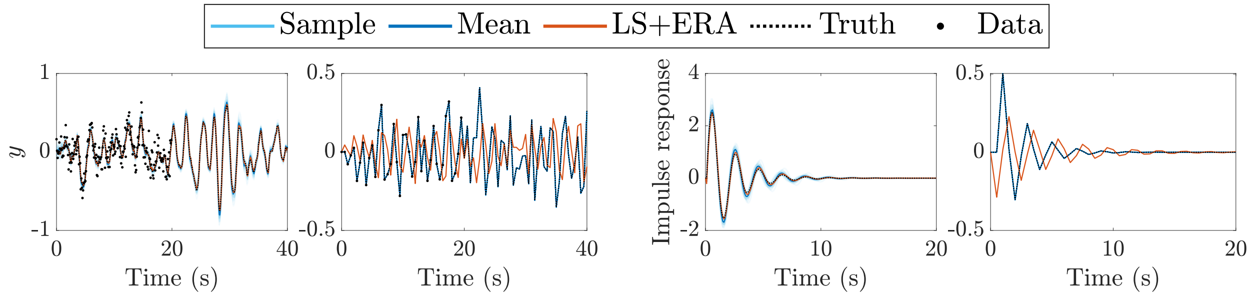
(a) LS+ERA Training (b) LS+ERA Testing (c) MAP Training (d) MAP Testing

Figure 3.12: Contour plots of the $\log_{10}$(MSE) of the LS+ERA and MAP estimates on System 3.20. Fig. 3.12a and 3.12c are the training MSE and Figs. 3.12b and 3.12d are the testing MSE.

cases, respectively. Furthermore, we examine the impulse response to check whether the model only learned how to produce sinuoids at a given frequency or if the estimated system actually approximates a realization of the state-space matrices as desired. The impulse response for the models in the high noise and high sparsity cases are shown in Figs. 3.13c and 3.13d, respectively.

To represent the posterior predictive distribution, 10^6 samples were drawn using the DRAM within Gibbs procedure described earlier in Section 2.3.3.1, the first 10^5 were discarded as burn-in, and 100 samples selected at regular intervals were simulated and plotted. The blue 'mean' line indicates the mean of these posterior predictive samples.

In the high noise, low sparsity case (Figs. 3.13a and 3.13c), the mean estimate and LS+ERA estimate both appear to fit the truth fairly well, despite the noisiness of the data. The MSE of the LS+ERA estimate over the full 40s is 2.86×10^{-4} and the MSE of the mean estimate is 4.74×10^{-4}. This was the only dataset on which the LS+ERA MSE was less than that of the MAP estimate. In fact, the next lowest LS+ERA training MSE was 4.68×10^{-4}, which is larger than the average MAP training MSE of 4.47×10^{-4}. Furthermore, the standard deviations of the LS+ERA training and testing MSE over all 100 datasets are 1.72×10^{-2} and 2.58×10^{-2} respectively, while the training and testing standard deviations of the MAP are 3.65×10^{-4} and 5.56×10^{-4}, respectively. The fact that the standard deviation of the LS+ERA MSE is about 100 times greater than that of the MAP MSE indicates that there are far worse LS+ERA estimates than the one presented here, but the MAP estimates likely all resemble the one shown in the figure. In the impulse response, the LS+ERA estimate is also closer to the truth than the mean estimate, but the posterior is wide and encompasses the truth. Therefore, the Bayesian method is 'aware' of the error and gives a reasonable quantification of the estimate uncertainty. In the low noise, high sparsity case (Figs. 3.13b and 3.13d), the posterior is so narrow that it visually appears as a single line, indicating low uncertainty. In both the output and impulse response plots, the mean is directly on top of the truth, and the LS+ERA estimate has large discrepancies between its output and the truth. In contrast to the Bayesian estimate, the LS+ERA method has no way to identify this larger error/uncertainty.



(a) $\sigma = 0.2$, $\Delta t = 0.1$ (b) $\sigma = 0.0$, $\Delta t = 0.5$ (c) $\sigma = 0.2$, $\Delta t = 0.1$ (d) $\sigma = 0.0$, $\Delta t = 0.5$

Figure 3.13: The Bayesian estimate is compared to the LS+ERA with $\bar{n} = 18$. The top row shows the LS+ERA estimate, deterministic simulations of 100 posterior samples, and the mean of the sample outputs. The bottom row shows the impulse response of each of these estimates. The left column shows the high noise, low sparsity case. The right column is the low noise, high sparsity case.

3.4 Summary

In this chapter, we considered the problem of learning state-space realizations of linear systems. Specifically, we considered DMD and single rollout Markov parameter estimation algorithms. We showed that the marginal likelihood reduces down to the DMD objective if we assume that there is no measurement noise and that the process noise covariance matrix is proportional to the identity matrix. We also showed that the single rollout Markov parameter estimation objective is equivalent to the marginal likelihood under the assumption of conditionally independent data. Finally, we provided numerical experiments showing how the flexibility of the marginal likelihood allows it to outperform these other two objectives at varying levels of data noise and sparsity.

CHAPTER IV
Nonlinear System Identification

In nonlinear system ID, there are certain behaviors that the models and underlying systems can exhibit that make many LS objectives unsuitable for estimation. For example, many nonlinear models can have their states go to infinity in finite time depending on the model parameters. Therefore, if the objective requires the model to be simulated over a relatively long period of time, the optimizer will likely run into this diverging behavior often. When the states diverge, the objective function value and gradients are undefined, which makes optimization especially difficult. Another example is chaos. In a chaotic system, arbitrarily small perturbations to the state grow exponentially over time. Thus, even small errors in the model's simulated state can quickly lead to large errors, making it difficult to discern good models. Mechanically, these issues lead to objectives that are multi-modal and difficult to optimize.

In this chapter, we consider two popular methods for addressing these optimization difficulties. The first approach uses a basis expansion to model a vector field and then poses the estimation problem as a linear least squares problem to find the basis coefficients. When this approach additionally includes threshholding as a form of sparsification, it is known as SINDy [10]. The second approach introduces a simulation length into the objective as a hyperparameter as a means to avoid the excessive error growth common in nonlinear systems. Using such an objective is known as MS [9], and it has been shown to create smoother, and therefore easier to optimize, objective function surfaces.

4.1 Theoretical foundations and analysis

We begin by presenting the objective functions for SINDy and MS. Similarly to the previous chapter, we will then derive the assumptions required for the marginal likelihood to take the form of each objective. This section concludes with a numerical comparison showing that, much like the time horizon parameter in MS, the process noise variance parameter in the marginal likelihood can deliver similar smoothing effects to the objective function surface.

4.1.1 Regularized regression for nonlinear models

For computational efficiency and ease of optimization, it is often convenient to formulate nonlinear system ID as a linear least squares problem. If a parsimonious solution is desired, as is often the case, regularization can also be added. One such approach that uses sparsity-enhancing regularization is the method of sparse regression or SINDy. These approaches organize a library of candidate functions (linear and nonlinear) into a matrix. They then aim to approximate the time derivative, or vector field, in the span of this library. For instance,

$$\dot{x} = f(x) \approx \begin{bmatrix} 1 & x & x^2 & \dots & x^{d_\theta} \end{bmatrix} \begin{bmatrix} \theta_1 & \theta_2 & \dots & \theta_{d_\theta} \end{bmatrix}^\top . \tag{4.1}$$

This example uses monomial candidate functions, but any basis (wavelets, orthogonal polynomials, empirical bases) can be used.

Suppose that the general dictionary of terms is given by $\Xi : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$ so that the deterministic portion of some continuous-time autonomous dynamics can be written as a linear system with respect to the parameters/coefficients of the functions in the dictionary $\dot{\mathbf{x}} = \Xi(\mathbf{x})\boldsymbol{\theta}_\Psi$. If direct data were available on the states and derivatives, one might then try to solve a (regularized) linear least squares problem for the parameters

$$\boldsymbol{\theta}_\Psi = \arg\min_{\tilde{\boldsymbol{\theta}}} \sum_{k=0}^{n} \|\dot{\mathbf{x}}_k - \Xi(\mathbf{x}_k)\tilde{\boldsymbol{\theta}}\|_2^2 + \lambda \|\tilde{\boldsymbol{\theta}}\|, \tag{4.2}$$

where λ is a regularization weight, and the norm can be chosen by the user. If the L_1 norm is chosen, this becomes a sparse regression problem.

Practical applications, however, do not have data on the derivative of each state. As a result, various numerical approximations can be made, and this is the approach taken by the SINDy algorithm. Here, we will consider one type of numerical approximation to the derivative, but our analysis can be extended to others. If a forward-difference approximation to the time derivative is taken, then the SINDy objective function is

$$\boldsymbol{\theta}_\Psi = \arg\min_{\tilde{\boldsymbol{\theta}}} \sum_{k=1}^{n} \left\| \frac{\mathbf{y}_k - \mathbf{y}_{k-1}}{\Delta t} - \Xi(\mathbf{y}_{k-1})\tilde{\boldsymbol{\theta}} \right\|_2^2 + \lambda \|\tilde{\boldsymbol{\theta}}\|_1. \tag{4.3}$$

Notice that this approach requires direct observation of the states. Next we show that this estimate is equivalent to the MAP of our target conditional distribution under more strict assumptions.

Theorem 4 (SINDy as a MAP estimate of system (2.5))**.** *Let* $\Xi(\mathbf{x}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$ *denote a library of candidate functions for continuous time drift dynamics. Let* $\Psi(\mathbf{x}; \boldsymbol{\theta}_\Psi)$ *denote the resulting discrete-time operator that uses a forward-Euler integration scheme*

$$\Psi(\mathbf{x}, \boldsymbol{\theta}_\Psi) = \mathbf{x} + \Delta t \Xi(\mathbf{x}) \boldsymbol{\theta}_\Psi. \tag{4.4}$$

Furthermore, assume an identity observation operator $h = \mathbf{I}$*; noiseless measurements* $\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma) = \mathbf{0}$*; identity process noise* $\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma) = \mathbf{I}$*; and a Laplace prior* $\pi(\boldsymbol{\theta}_\Psi) \propto \exp\left(-\tilde{\lambda}|\boldsymbol{\theta}_\Psi|\right)$*. Then, the MAP estimate of the posterior distribution* (2.2) *is equivalent to the SINDy estimator obtained by minimizing* (4.3)*.*

Proof. This proof is again a straightforward application of Theorem 2. Recall that the data are taken on the initial condition, and note that we have $\mathbf{y}_k = \mathbf{x}_k$. The log marginal likelihood (2.26) is then

$$\begin{aligned} \log \mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n) = \sum_{k=1}^{n} \Big(-\frac{1}{2} \left\|\mathbf{y}_k - (\mathbf{y}_{k-1} + \Delta t \Xi(\mathbf{y}_{k-1}) \boldsymbol{\theta}_\Psi)\right\|^2_{\mathbf{I}_{d_x}} \\ + \log|\mathbf{I}_{d_x}| \Big) - \frac{n d_x}{2} \log 2\pi - \frac{n}{2} \log|\mathbf{I}_{d_x}| \end{aligned} \tag{4.5}$$

$$= -\frac{\Delta t}{2} \sum_{k=1}^{n} \left\| \frac{\mathbf{y}_k - \mathbf{y}_{k-1}}{\Delta t} - \Xi(\mathbf{y}_{k-1}) \boldsymbol{\theta}_\Psi \right\|_2^2 - \frac{n d_x}{2} \log 2\pi. \tag{4.6}$$

Then we can drop the parameter-independent term and add the log prior to obtain a posterior that is proportional to

$$\log \pi(\boldsymbol{\theta}; \mathcal{Y}_n) \propto -\frac{\Delta t}{2} \sum_{k=1}^{n} \left\| \frac{\mathbf{y}_k - \mathbf{y}_{k-1}}{\Delta t} - \Xi(\mathbf{y}_{k-1}) \boldsymbol{\theta}_\Psi \right\|_2^2 - \tilde{\lambda}|\boldsymbol{\theta}_\Psi| \tag{4.7}$$

$$= -\frac{\Delta t}{2} \left(\sum_{k=1}^{n} \left\| \frac{\mathbf{y}_k - \mathbf{y}_{k-1}}{\Delta t} - \Xi(\mathbf{y}_{k-1}) \boldsymbol{\theta}_\Psi \right\|_2^2 + \frac{2\tilde{\lambda}}{\Delta t} |\boldsymbol{\theta}_\Psi| \right). \tag{4.8}$$

Maximizing the posterior is equivalent to minimizing the term in the parentheses. By setting $\lambda \equiv \frac{2\tilde{\lambda}}{\Delta t}$, we see that this is the exact form of the SINDy objective (4.3). □

4.1.2 Multiple shooting objective

Next, we turn to an approach that does not admit a closed-form solution, but instead tries to balance ease of optimization with noise-robust estimation: MS. In MS, the output trajectory is divided into L disjoint subtrajectories with initial times $\{t_{\ell_i}\}_{i=1}^{L}$ such that the ith subtrajectory has length $\Delta\ell_i \coloneqq \ell_{i+1} - \ell_i$. Then the output at time t_k contained within the ith subtrajectory is estimated as $\hat{\mathbf{y}}_k = f(\mathbf{x}_{\ell_i}, \mathbf{u}_{\ell_i:k}, t_k; \boldsymbol{\theta})$. Such an objective function requires the estimation of the set of subtrajectory initial conditions $\mathcal{Z}_L \coloneqq \{\mathbf{x}_{\ell_i}\}_{i=1}^{L}$, which can be done by adding the initial conditions as parameters [83], training an encoder [66, 6], or, if the system is fully observed, simply using the data $\mathbf{x}_{\ell_i} = \mathbf{y}_{\ell_i}$ [112]. In effect, this method introduces additional parameters (the initial conditions) as the cost for an improved estimate.

The MS objective function is defined as

$$\mathcal{J}(\boldsymbol{\theta}) = \sum_{i=1}^{L} \sum_{k=\ell_i}^{\ell_{i+1}-1} \|\mathbf{y}_k - \hat{\mathbf{y}}_k\|_2^2, \tag{4.9}$$

where $\ell_{L+1} \coloneqq n+1$. Oftentimes, a constant length of $T = \Delta\ell_i$ for all $i = 1, \ldots, L$ is used for simplicity. In the case $T = n$, the objective is equivalent to the deterministic LS, and if $T = 1$, the objective is equivalent to the propagator LS. Therefore, when $1 < T < n$, MS can be seen as a combination of these two objectives. In the original paper [9], the objective additionally had the constraint that $\mathbf{x}_{\ell_{i+1}} = \Psi^{\Delta\ell_i}(\mathbf{x}_{\ell_i}, \mathbf{u}_{\ell_i:\ell_{i+1}}, \boldsymbol{\theta}_\Psi)$, but this constraint is sometimes removed to simplify optimization. We will distinguish between the objectives with and without the constraints by referring to them as the constrained and unconstrained MS objectives, respectively.

4.1.2.1 Relation to probabilistic approach

From a probabilistic perspective, the MS objective amounts to a joint parameter-state estimation problem. Rather than estimating the state at every timestep, however, only the subset of subtrajectory initial conditions $\mathcal{Z}_L \subseteq \mathcal{X}_n$ are estimated. The posterior of such a problem can be factorized with Bayes' rule as

$$\pi(\mathcal{Z}_L, \boldsymbol{\theta} | \mathcal{Y}_n) \propto \mathcal{L}(\boldsymbol{\theta}, \mathcal{Z}_L; \mathcal{Y}_n)\pi(\mathcal{Z}_L, \boldsymbol{\theta}). \tag{4.10}$$

Factorizing further, the likelihood can be decomposed as $\mathcal{L}(\boldsymbol{\theta}, \mathcal{Z}_L; \mathcal{Y}_n) = \prod_{k=\ell_i}^{\ell_{i+1}-1} \pi(\mathbf{y}_k|\mathbf{x}_{\ell_i}, \boldsymbol{\theta})$ and the prior as $\pi(\mathcal{Z}_L, \boldsymbol{\theta}) = \pi(\boldsymbol{\theta}) \prod_{i=1}^{L} \pi(\mathbf{x}_{\ell_i}|\mathbf{x}_{\ell_{i-1}}, \boldsymbol{\theta})$, where $\pi(\mathbf{x}_{\ell_1}|\mathbf{x}_{\ell_0}, \boldsymbol{\theta}) \coloneqq \pi(\mathbf{x}_{\ell_1}|\boldsymbol{\theta})$. Each term in the likelihood $\pi(\mathbf{y}_k|\mathbf{x}_{\ell_i}, \boldsymbol{\theta})$ can still be evaulated with Algorithm 1 using data from only a single trajectory.

The most significant difference is that the prior has the added terms $\pi(\mathbf{x}_{\ell_i}|\mathbf{x}_{\ell_{i-1}}, \boldsymbol{\theta})$, which can be evaluated as $\pi(\mathbf{x}_{\ell_i}|\mathbf{x}_{\ell_{i-1}}, \boldsymbol{\theta}) = \int \prod_{k=\ell_{i-1}+1}^{\ell_i} \pi(\mathbf{x}_k|\mathbf{x}_{k-1}) \times \mathrm{d}\mathbf{x}_{\ell_i-1}\mathrm{d}\mathbf{x}_{\ell_i-2} \ldots \mathrm{d}\mathbf{x}_{\ell_{i-1}+1}$. This equation shows that $\pi(\mathbf{x}_{\ell_i}|\mathbf{x}_{\ell_{i-1}}, \boldsymbol{\theta})$ can be seen as representing the probability of $\mathbf{x}_{\ell_i}$ averaged over all trajectories that start at $\mathbf{x}_{\ell_{i-1}}$ with dynamics determined by $\boldsymbol{\theta}$. Alternatively, this term can be viewed as a soft constraint enforcing the estimated initial conditions to be connected by a single trajectory with initial condition $\mathbf{x}_{\ell_1}$. Under certain conditions, this constraint is equivalent to the MS constraints. Furthermore, estimators based on the posterior (4.10) are equivalent to estimators using the (un)constrained MS objective (4.9) under certain assumptions. This result is stated in Proposition 4.

Proposition 4. *Assume an improper uniform prior distribution on the parameters $\boldsymbol{\theta}$ and zero process noise $\boldsymbol{\Sigma} = \mathbf{0}$. Then, the negative log marginal likelihood of the joint parameter-state estimation problem* (4.10) *is equivalent to the unconstrained MS objective* (4.9)*. Moreover, the negative log posterior is equivalent to the constrained MS objective.*

Proof. According to Theorem 2 in [28], each term $\pi(\mathbf{y}_k|\mathbf{x}_{\ell_i}, \boldsymbol{\theta})$ in the marginal likelihood is equivalent to a deterministic LS objective when $\boldsymbol{\Sigma} = \mathbf{0}$. Then, taking the negative log of this product gives the unconstrained MS objective. When the prior over the states is added, each term $\pi(\mathbf{x}_{\ell_i}|\mathbf{x}_{\ell_{i-1}}, \boldsymbol{\theta})$ approaches the Dirac delta function $\delta_{\Psi^{\Delta_{\ell_i}}(\mathbf{x}_{\ell_{i-1}}, \mathbf{u}_{\ell_{i-1}:\ell_i}, \boldsymbol{\theta}_\Psi)}(\mathbf{x}_{\ell_i})$ as $\boldsymbol{\Sigma} \to \mathbf{0}$. These delta functions are equivalent to the constraints in the constrained MS objective. The prior over the parameters is constant, so taking the negative log of the posterior yields the constrained MS objective. □

4.1.2.2 Comparison of smoothing effects

Now that it is understood that the MS objective is equivalent to an objective of a joint parameter-state estimation problem, we demonstrate empirically that the additional expense of inferring the subtrajectory initial conditions is not computationally necessary. More specifically, we demonstrate that the proposed marginal likelihood improves the optimization surface in a similar fashion to the MS objective, and we provide a brief discussion on the advantages of the proposed approach.

To show the similarity of the smoothing effects in MS and the marginal likelihood, the logistic map example from [83] is considered. The logistic map is defined as

$$y_{k+1} = \theta y_k(1 - y_k). \tag{4.11}$$

This system exhibits chaotic behavior when θ is within the range [3.57, 4]. For this example, θ was set to 3.78, an initial condition of $y_0 = 0.5$ was used, and 200 noiseless data points were collected. To show how the objective surfaces vary with θ, all other parameters must be fixed. For the MS objective, the initial conditions were set to the true values, and for the marginal likelihood, we set $\mathbf{\Gamma} = 10^{-16}$ to maintain positive definiteness. Then, the objectives were compared at different time horizons T and variance ratios $\mathbf{\Sigma}/\mathbf{\Gamma}$. The ratio $\mathbf{\Sigma}/\mathbf{\Gamma}$ is used because in this problem, the shape of the objective did not appear to change for a fixed ratio, regardless of the $\mathbf{\Sigma}$ and $\mathbf{\Gamma}$ values. Note that the validity of using this ratio is only possible since the full state is observed, meaning that $\mathbf{\Sigma}$ and $\mathbf{\Gamma}$ are represented in the same coordinate frame.

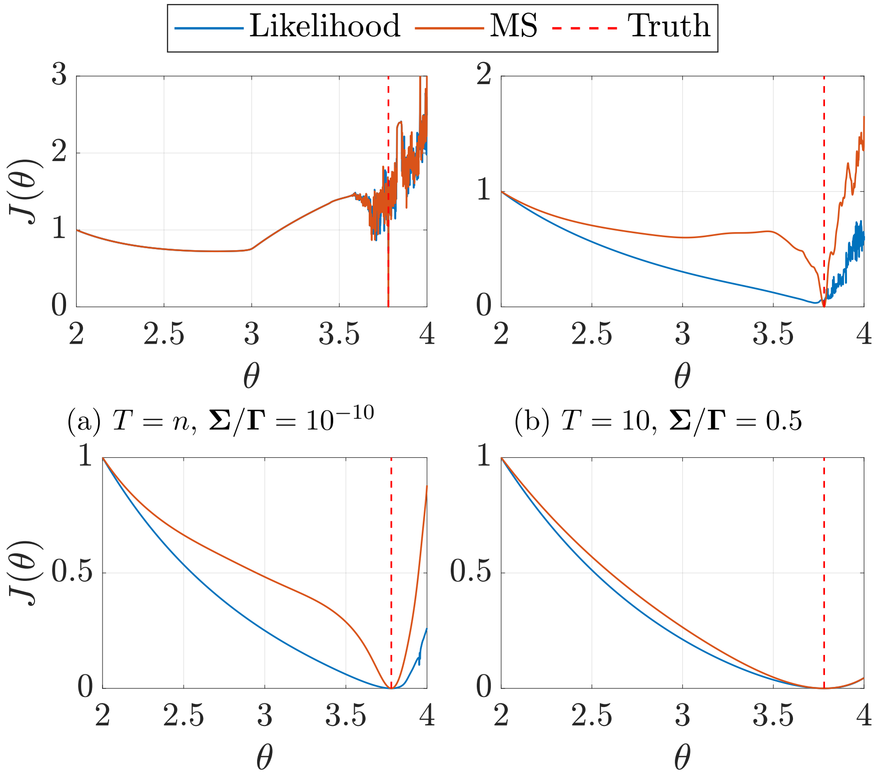
Both surfaces were normalized to equal 1.0 at $\theta = 2$, and the results are shown in Fig. 4.1. There is no exact mapping between T and $\mathbf{\Sigma}/\mathbf{\Gamma}$, so the values of $\mathbf{\Sigma}/\mathbf{\Gamma}$ were chosen such that the smoothness of the marginal likelihood roughly matched that of the MS objective by visual comparison. Fig. 4.1a shows values of T and $\mathbf{\Sigma}/\mathbf{\Gamma}$ where MS is equivalent to and the marginal likelihood approximates the deterministic LS. Due to the chaotic nature of this system, the deterministic LS objective is filled with local minima that make optimization extremely difficult. As T is decreased and $\mathbf{\Sigma}$ is increased, both surfaces show increasing smoothness, demonstrating the similar effect these variables have on their respective objectives. An important difference to note between T and $\mathbf{\Sigma}$ is that T is a discrete scalar variable, whereas $\mathbf{\Sigma}$ is a positive definite matrix of continuous values. Therefore, $\mathbf{\Sigma}$ gives the user greater flexibility when tuning the marginal likelihood, including the ability to use different variance values for different components of the state.

4.2 Numerical experiments

We now seek to compare SINDy and MS objectives on a number of different numerical experiments that include chaotic systems, PDEs, and physical data. We will show that SINDy can be very sensitive to measurement noise and timestep since it relies on time derivative estimation. The marginal likelihood, however, is robust to noise and can outperform SINDy even when the model form is known. Then, we show that that although MS can achieve lower MSE on training data than the marginal likelihood, it usually performs worse on testing data. This is a consequence of the fact that it has no method of regularization, whereas the $\log\det \mathbf{S}_k$ in the marginal likelihood can effectively prevent overfitting. The code for these experiments can be found at https://github.com/ngalioto/BayesID.

4.2.1 Vector field estimation

In the first set of numerical experiments, our aim is to estimate the set of differential equations, or vector field, governing the system of interest. Modeling the vector field of a system is often more general than learning a discrete mapping since it can be applied to different



(a) $T = n$, $\mathbf{\Sigma}/\mathbf{\Gamma} = 10^{-10}$

(b) $T = 10$, $\mathbf{\Sigma}/\mathbf{\Gamma} = 0.5$

(c) $T = 5$, $\mathbf{\Sigma}/\mathbf{\Gamma} = 0.7$

(d) $T = 2$, $\mathbf{\Sigma}/\mathbf{\Gamma} = 1.0$

Figure 4.1: Comparison of the log marginal likelihood and MS objective as $\mathbf{\Sigma}$ and T, respectively, vary.

time discretizations. Differential equations also have the added benefit of being more interpretable than discrete maps. In fact, models derived from physics are typically found in the form of differential equations. As a result, grey-box models often fall under this category of vector field estimation.

The first system that we consider is the Van der Pol oscillator, which displays nonlinear behavior in the form of a limit cycle. We then consider two known model forms in which we attempt to estimate the unknown parameters. The first of these systems is the chaotic Lorenz 63 system, and the second is a reaction-diffusion PDE. Since we are attempting to estimate a vector field, we will compare to SINDy in all experiments in this section.

4.2.1.1 Van der Pol oscillator

Consider the nonlinear Van der Pol oscillator

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \mu(1 - x_1^2)x_2 - x_1 \end{bmatrix}, \qquad \mathbf{x}_0 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \tag{4.12}$$

where $\mu = 3$. For both the Bayesian and SINDy algorithms, we consider a subspace of right hand sides that is spanned by a set of candidate functions. We choose monomial candidates up to third degree and their interacting terms. As a result, each algorithm seeks to learn 20 dynamics parameters (10 for each state). The Bayesian algorithm is additionally tasked with learning the covariance matrices parameterized as follows:

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma) = \begin{bmatrix} \theta_{21} & 0 \\ 0 & \theta_{22} \end{bmatrix}, \quad \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma) = \theta_{23}\mathbf{I}_2. \tag{4.13}$$

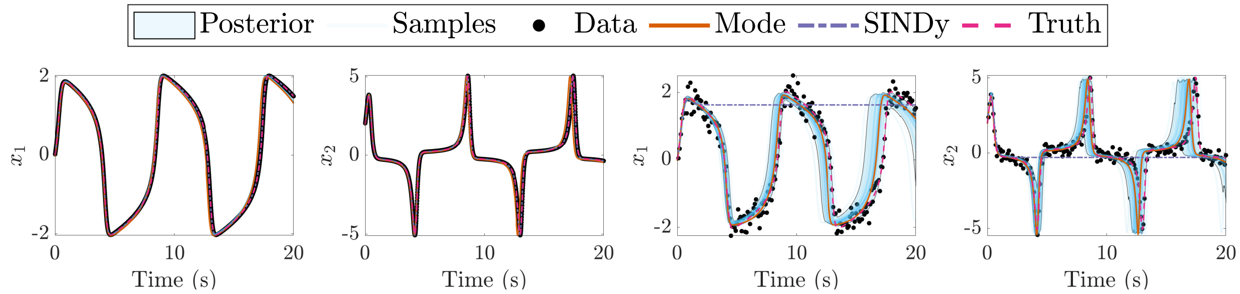
The priors on the dynamics parameters are Laplace distributions with zero mean, and the variance parameters priors are once again half-normal distributions.

We consider two cases: one where SINDy shows strong performance, and one in which SINDy struggles, and we show that the Bayesian algorithm yields an accurate estimate in both cases. The case in which SINDy excels is frequent and low noise data. Here, $n = 2000$ measurements were taken over the course of 20s with measurement noise standard deviation of $\sigma = 10^{-3}$. In the opposing case, we collect only $n = 200$ measurements over 20s with measurement noise standard deviation of $\sigma = 2.5 \times 10^{-1}$.

The reconstructions from these experiments are shown in Fig. 4.2, predictions are given in Fig. 4.3, and the phase plots over 200s are given in Fig. 4.4. Here, the mode represents the mode of the posterior predictive distribution. In the low noise case, we see that the Bayesian algorithm and SINDy both capture the dynamics very closely. We see that SINDy agrees slightly more closely with the trajectory as a result of its hard threshold regularization. Note that the posterior in this case is very small because the high number of data points and low measurement noise gives us high certainty in our estimate. In the high noise case, we see that SINDy gives a similar result to what DMD gave when the measurements were noisy: the trajectory immediately flatlines. When the data are noisy like this, the procedure for SINDy is to denoise the data using total variation (TV) regularization [14] before executing the algorithm. However, the increased timestep between data makes it difficult to accurately denoise the data, and when the TV regularization is performed, SINDy ends up giving an unstable estimate. The Bayesian approach, however, is still able to identify the dynamics of the Van der Pol system. The posterior in this high noise case is wider, signifying that the estimate holds more uncertainty than the low noise and frequent measurements case.

4.2.1.2 Lorenz 63

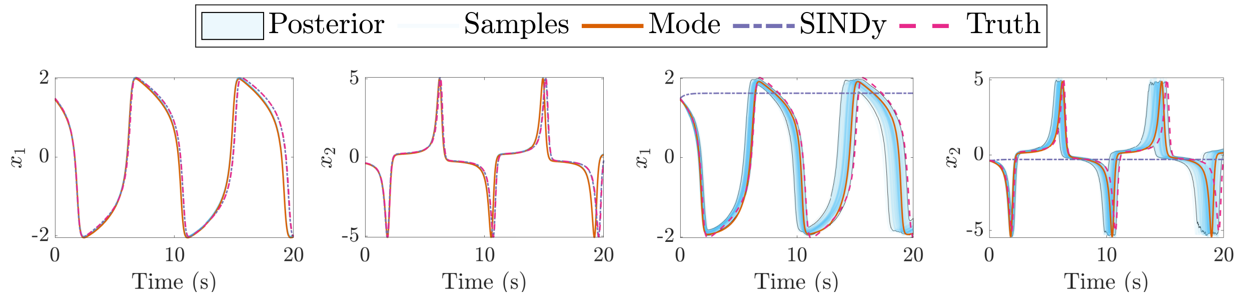
Next, we consider the case where the model form is known, for instance from physical laws, but the parameters are uncertain. This is the classical inference setting and has seen



(a) Reconstruction

(b) Reconstruction

Figure 4.2: Comparison of reconstruction error amongst the Bayesian and SINDy algorithms for the Van der Pol system. Top row corresponds to a low-noise/dense-data case, and the bottom row corresponds to a high-noise/sparse-data case. Left column corresponds to the first state (position), and right column corresponds to the second state (velocity). The Bayesian estimator is able to accurately reconstruct the dynamics, even in the presence of high noise.



(a) Prediction

(b) Prediction

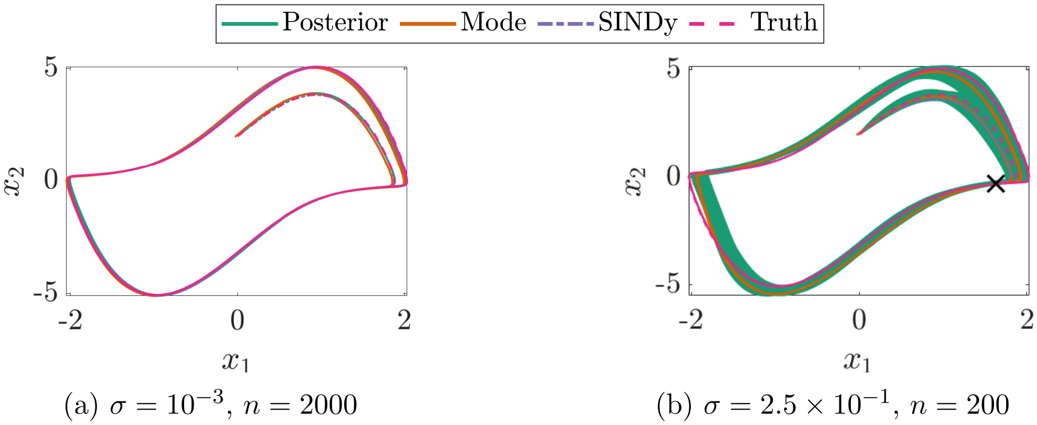
Figure 4.3: Comparison of prediction error amongst the Bayesian and SINDy algorithms for the Van der Pol system. The meaning of the figures is the same as described in Fig. 4.2. The model learned by the Bayesian estimator is still accurate at a different initial condition.

a lot of development [53, 22, 65, 64], including in the computational physics community. However, much of this literature either only considers deterministic dynamics according to some variation of Eq. (1.2) or only static problems. In this section, we show that we are able to effectively learn chaotic dynamics by imposing process noise in the estimation with a much smaller amount of data than observed in the literature.

Consider the chaotic Lorenz 63 system [60]

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}\theta_1(x_2 - x_1)\\ x_1(\theta_2 - x_3) - x_2\\ x_1x_2 - \theta_3x_3\end{bmatrix}, \qquad \mathbf{x}_0 = \begin{bmatrix}2.0181\\ 3.5065\\ 11.8044\end{bmatrix}. \tag{4.14}$$

The initial condition of this system was chosen so as to sit on the attractor. We attempt



(a) $\sigma = 10^{-3}$, $n = 2000$

(b) $\sigma = 2.5 \times 10^{-1}$, $n = 200$

Figure 4.4: Phase-diagram reconstruction for the Van der Pol oscillator under the two indicated data conditions. In the low-noise and frequent data domain, both the Bayesian and SINDy estimates lie directly on the truth. In the high noise-case, the Bayesian posterior is wider, but is still visually aligned with the truth. The SINDy estimate is unable to recover the limit cycle, and the large "x" marks the equilibrium point to which SINDy converges, as shown in Fig. 4.3.

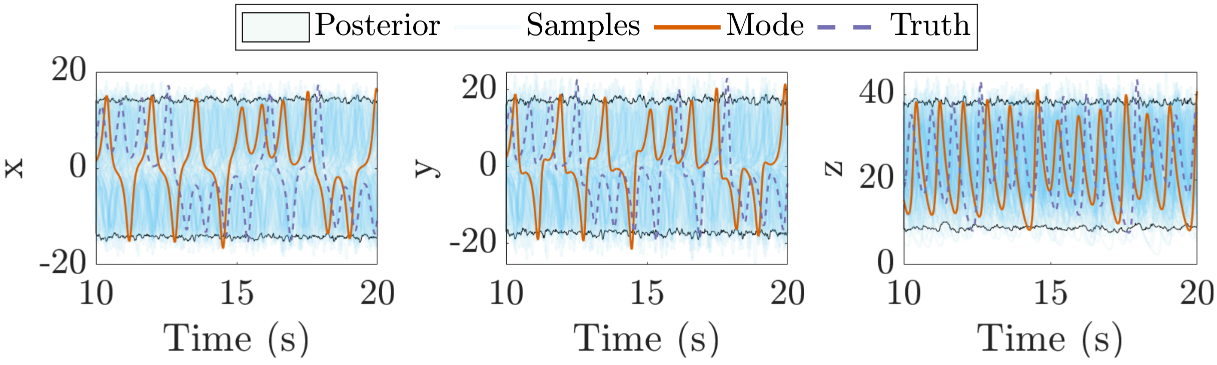
only to learn the parameters $\boldsymbol{\theta}_\Psi = (\theta_1, \theta_2, \theta_3)$, where the ground truth values are $\boldsymbol{\theta}_\Psi = (10, 8/3, 28)$. Computing the likelihood for chaotic systems can be difficult. Since filtering chaotic systems is well-known to be challenging, it may seem that our approach would breakdown. Here we show that our Gaussian filtering approach is still able to learn an approximate dynamical system without resorting to more complicated likelihood building processes, e.g., using correlation integrals [37, 96].

The priors on the dynamics parameters are once again improper and uniform. In addition to learning the model parameters in this example, we also learn the process noise variance for each state and the measurement noise variance for a total of seven parameters. The parameterizations of the covariance matrices are shown:

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma) = \begin{bmatrix} \theta_4 & 0 & 0 \\ 0 & \theta_5 & 0 \\ 0 & 0 & \theta_6 \end{bmatrix}, \quad \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma) = \theta_7 \mathbf{I}_3, \tag{4.15}$$

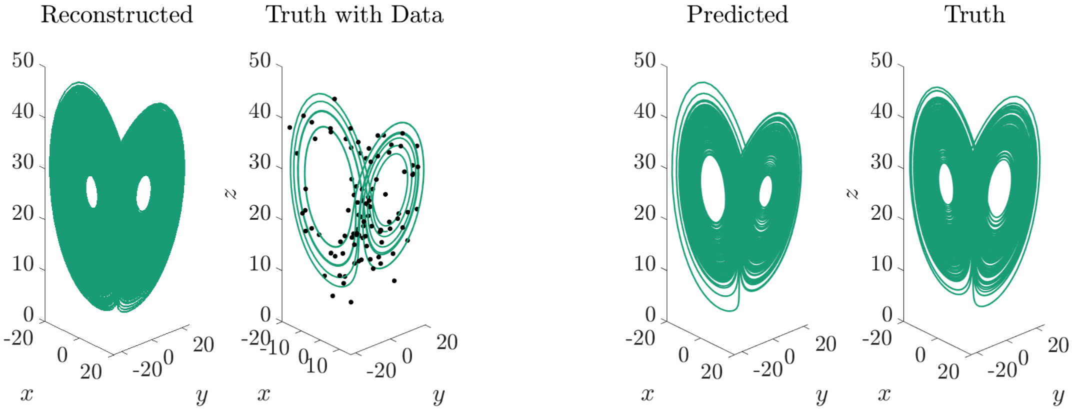
with half-normal priors as before.

We collect 100 data points uniformly spaced over 10s with a true measurement noise standard deviation of 2.0. The predicted state trajectories after 10s of simulation using the parameter posterior mode are shown in Fig. 4.5. Similar to the Van der Pol oscillator, the dynamics exist on a low-dimensional attractor in phase space, and the wide, but constant, posterior distribution once again reflects this fact. Fig. 4.6 shows the reconstructed and predicted attractors from the Bayesian algorithm. These figures show that while we cannot accurately capture the state, indeed all methods would eventually break down due to the chaotic nature of the system, we do predict a qualitatively similar attractor. As such, one would expect that most post-processing of these attractors, e.g., for control, would yield similar results.



(a) Prediction of x (b) Prediction of y (c) Prediction of z

Figure 4.5: Lorenz 63 prediction posteriors. Although the trajectories become misaligned rather quickly due to the chaotic nature of the system, the posterior phase diagram 4.6 reveals that the algorithm has discovered that the dynamics exist on a low-dimensional attractor.

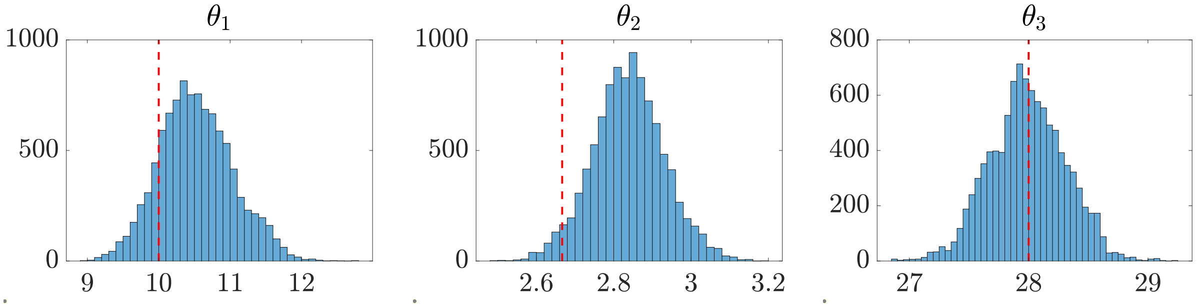


(a) Reconstruction (b) Prediction

Figure 4.6: Reconstruction and prediction of the Lorenz 63 attractor. The right panel compares the predicted and true trajectories up to 200s using the mode of the parameter posterior distribution. The proposed approach is able to successfully discover the Lorenz attractor from sparse, noisy data.

In addition to assessing the model's predictive ability, we can also use the posterior samples to estimate the value of the parameters themselves. Fig. 4.7 shows the marginal distributions of the parameter posterior with the truth values marked by the dashed red line. These distributions are relatively wide as a result of the large amount of noise in the data but are still able to encompass the truth values, demonstrating the robustness of the approach. We can also infer from these distributions that the model is more sensitive to the value of θ_2 since the marginal distribution of this parameter is much more narrow than the other two.

Similar to the eigenvalues of the linear pendulum in Section 3.3.1, the samples collected here can be used to investigate the probability distribution of any dynamical quantity of interest. Fig. 4.8 shows the estimation of the three Lyapunov exponents of the Lorenz system. Lyapunov exponents are a measure of the exponential growth rates of generic perturbations of a system. A positive Lyapunov exponent implies that an arbitrarily small perturbation will grow exponentially large over time. Such systems are considered to be chaotic [75]. Here, the Lyapunov exponents are computed using a function from MATLAB file exchange [33] that uses the algorithm proposed in [106]. The red line denotes the approximated value of the Lorenz system's Lyapunov exponents using the truth values of the parameters. When approximating the Lyapunov exponents of a system, the growth of the intial perturbation is dominated by the largest Lyapunov exponents, making the smaller Lyapunov exponents more difficult to estimate precisely. This fact is reflected in the distribution of λ_3, which is much wider than the other two. We see that for each exponent, the truth value is contained in its respective distribution at a relatively high probability value.



(a) θ_1 Estimate

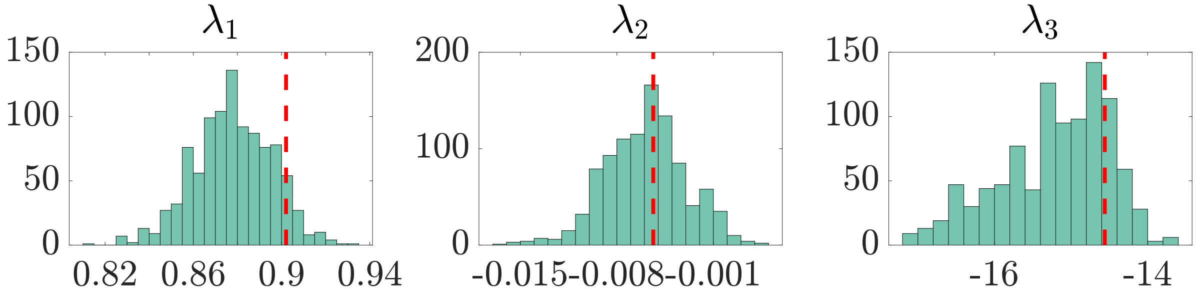
(b) θ_2 Estimate

(c) θ_3 Estimate

Figure 4.7: Posterior of the Lorenz 63 dynamics parameters. The distributions are relatively wide due to the high amount of noise in the data but still encompass the truth values, indicated by the dashed red line.

4.2.1.3 Reaction diffusion

In the final vector field example, we aim to show that the marginal likelihood is applicable to spatial problems. This experiment considers both a PDE and a case where the measurement



(a) λ_1 Estimate (b) λ_2 Estimate (c) λ_3 Estimate

Figure 4.8: Posterior of the Lyapunov exponent estimation of the Lorenz 63 system. The distribution of λ_3 is wider than the other two because the behavior of the system is dominated by the first two exponents, making the third difficult to estimate with high certainty.

operator h is not the identity. The reaction diffusion PDE is given by

$$\begin{aligned} \frac{\partial C_1}{\partial t} &= \theta_1 \frac{\partial^2 C_1}{\partial x^2} + 0.1 - C_1 + \theta_3 C_1^2 C_2, \\ \frac{\partial C_2}{\partial t} &= \theta_2 \frac{\partial^2 C_2}{\partial x^2} C_2 + 0.9 - C_1^2 C_2, \end{aligned} \tag{4.16}$$

where C_1 and C_2 specify the concentrations. A one-dimensional spatial grid was selected to have regular intervals of 0.4 units between boundaries of -40 and 40 for a total of 201 grid points for each of the two states. The boundary conditions at $x = \pm 40$ are

$$\frac{\partial C_1}{\partial x} = \frac{\partial C_2}{\partial x} = 0, \tag{4.17}$$

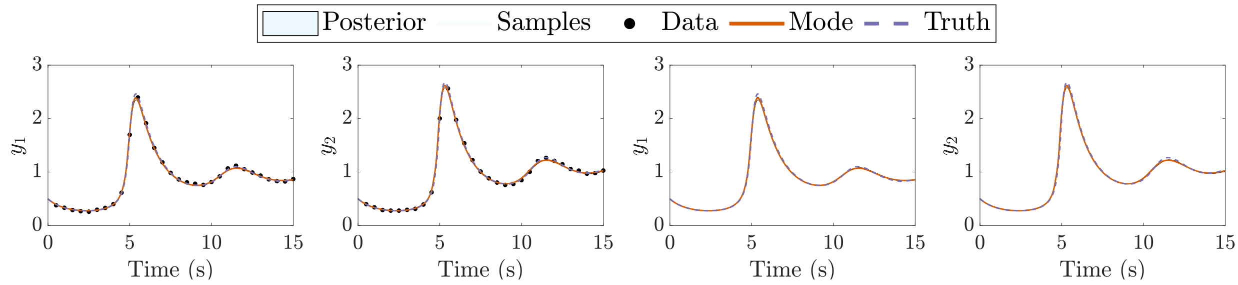
and the initial condition of the system was drawn from a uniform distribution as shown

$$(C_i)_j \sim \mathcal{U}[0.4, 0.6], \quad \text{for } t = 0;\ \forall i = 1, 2;\ \forall j = 1, \ldots, 201. \tag{4.18}$$

Similar to the Lorenz example, we attempt to learn only the model parameters θ_1, θ_2, and θ_3 rather than the complete model. The measurement covariance matrix is assumed to be known, and the process noise covariance is fixed to be 1e-8 such that the total number of parameters that we are learning remains only three. The observation operator indirectly measures the concentration through only the first two moments of the concentration of the first species at certain time intervals:

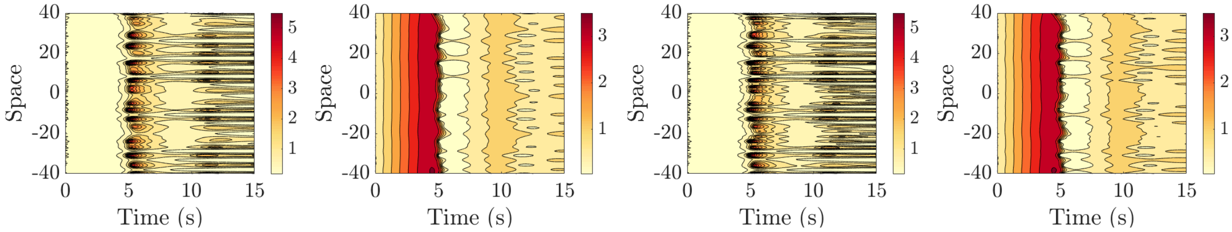
$$
\begin{aligned}
y_1(t) &= \int_{-40}^{40} C_1(t)\,\mathrm{d}x, \\
y_2(t) &= \int_{-40}^{40} C_1^2(t)\,\mathrm{d}x.
\end{aligned}
\tag{4.19}
$$

We collect measurements every 0.5s for 15s with noise standard deviation of 10^{-2}. The reconstructions and predictions of the moments from these data using the mode of the parameter posterior distribution are shown in Fig. 4.9. Additionally, the true and reconstructed contours of C_1 and C_2 are shown in Fig. 4.10. The Bayesian estimate shows close agreement with the truth.



(a) $n = 30$, $\sigma = 10^{-2}$ (b) Prediction

Figure 4.9: Reconstruction and prediction of the observables of the reaction diffusion system. The top row shows the reconstruction, and the bottom row shows the prediction for an alternate initial condition. The left column is the first measurement state (first moment), and the right column is the second measurement state (second moment). The estimates are very close to the truth, demonstrating the generality of the learned model.



(a) True (b) Reconstructed

Figure 4.10: The experiment is the same as in Fig. 4.9. The top row shows the true contours of C_1 and C_2. The bottom row shows the contours of C_1 and C_2 reconstructed using the mode of the parameter posterior distribution. Visually, the two rows appear very similar, reflecting the strong performance of the Bayesian algorithm.

4.2.2 Discrete-time neural network mappings

The goal of the second set of experiments is to estimate a discrete-time mapping of the system dynamics using a neural network. Working with discrete-time mappings rather than vector fields can be advantageous for the sake of numerical stability and efficiency. No numerical integration is needed, so one does not need to worry about stiffness or numerical error (other than machine precision). Also, only one evaluation of the estimated model is needed to step forward in time.

We use a similar neural network architecture throughout these experiments, changing only the dimensions of the input and output layers to match the dimensions of the system at hand. Following the approach of [6], the neural network has a single hidden layer with 15 nodes and tanh activation functions. Additionally, a linear transformation from the input of the network directly to the output is included such that the network form is

$$\mathbf{z}_{out} = \mathbf{A}_1(\boldsymbol{\theta}) \tanh(\mathbf{A}_2(\boldsymbol{\theta})\mathbf{z}_{in} + \mathbf{b}_2(\boldsymbol{\theta})) + \mathbf{A}_3(\boldsymbol{\theta})\mathbf{z}_{in} + \mathbf{b}_3(\boldsymbol{\theta}), \tag{4.20}$$

where $\mathbf{z}_{in} = \begin{bmatrix}\mathbf{x}_k^\top & u_k\end{bmatrix}^\top$. If the network parameterizes the dynamics function Ψ, then $\mathbf{z}_{out} = \mathbf{x}_{k+1}$, and if it parameterizes the observation function h, then $\mathbf{z}_{out} = \mathbf{y}_k$.

In this section, we will consider learning a Wiener-Hammerstein system using data collected from a physical system. This dataset is considered to be a benchmark problem in system ID. Then we consider the forced Duffing oscillator in the chaotic regime. Lastly, we consider learning the dynamics of a quantity of interest from a PDE system. In each example, we compare the negative log marginal likelihood to the MS objective.

4.2.2.1 Wiener-Hammerstein benchmark

First, experimental data collected from a nonlinear system is considered. For this example, the proposed Bayesian method is tested on the Wiener-Hammerstein benchmark [90], which is a standard dataset that has been used to compare the performances of different nonlinear system ID methods. The benchmark dataset is composed of 188,000 low-noise input-output data points, with a suggested training/testing split of 100,000/88,000. The best performance to date on this benchmark to the authors' knowledge comes from [6], which uses the MS objective. In this experiment, the method of [6] will be compared to the Bayesian method.

Because the dataset has such a high number of data points with low measurement noise, the advantages of the Bayesian approach are not nearly as evident. It will be shown, however, that methods that work well with a large amount of low-noise data are not necessarily best-suited for estimation when the data are few and noisy. To this end, only the first 1,000 data points of the original 100,000 point training set were used for training. Furthermore, zero-mean Gaussian noise with standard deviation $\sigma = 0.0178$ was added to these training data. This standard deviation is equal to 1% of $(\mathbf{y}_{max} - \mathbf{y}_{min})$.

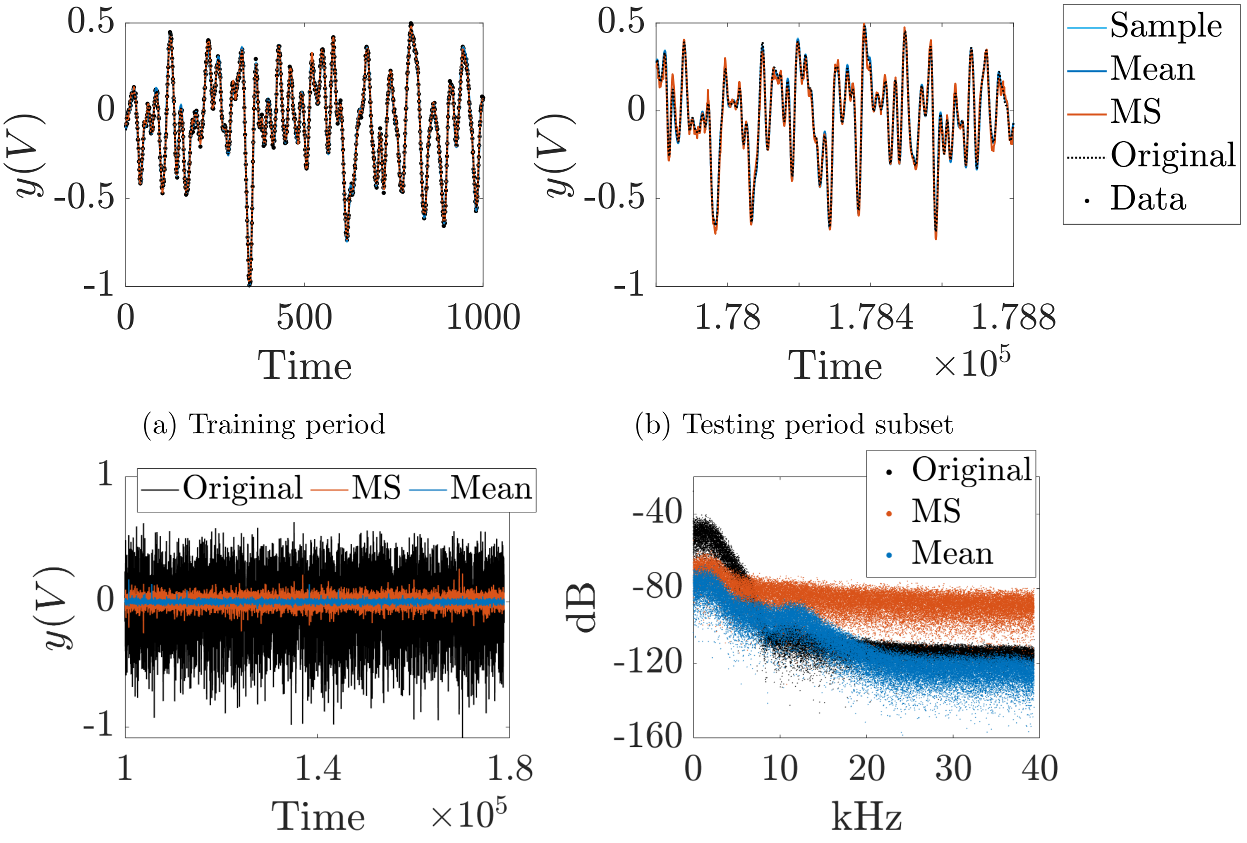
Both the dynamics model Ψ and measurement model h are parameterized by neural networks with the form in Eq. 4.20. The latent dimension of the state space is $d_x = 6$, which results in a total number of 401 parameters in Ψ and h combined. The one difference between our model and that of [6] is that rather than learning an encoder function to estimate the current state, the initial condition is estimated directly. The priors used for this model were half–$\mathcal{N}(0, 10)$ on $\boldsymbol{\theta}_\Sigma$, half–$\mathcal{N}(0, 0.01)$ on $\boldsymbol{\theta}_\Gamma$, and $\mathcal{N}(0, 0.2)$ on the remaining parameters.

Before training, the input and output data were both normalized to have zero means and standard deviations of one. The comparison method was trained with Adam batch optimization using the available code in the repository linked by [6]: https://github.com/GerbenBeintema/SS-encoder-WH-Silver. The batch size was reduced from 1,024 to 256 to handle the smaller dataset, but the number of epochs was kept at 100,000. The time horizon was also kept at $T = 80$ since it was chosen according to the time scale of the system, which does not change. The Bayesian method was trained for 10,000 iterations. Then, 10^5 samples were drawn from the posterior, and 2×10^4 were discarded as burn-in.

Figs. 4.11a and 4.11b show the estimated output of the Bayesian and MS estimates in the time domain during the training period and during the last 1,000 iterations of the testing period, respectively. The posterior predictive distribution is represented by 100 samples drawn at regular intervals from the collected samples and simulated deterministically. 'Mean' refers to the mean of these 100 posterior predictive samples. In Fig. 4.11a, the estimates look nearly identical. In Fig. 4.11b, some noisiness has appeared in the MS estimate indicative of overfitting, but the Bayesian estimate remains smooth due to its inherent regularization. Figs. 4.11c and 4.11d show the errors during the testing period in the time and frequency domain, respectively. The MSE values of the MS and mean estimates on the testing data are 1.0948×10^{-3} and 1.2546×10^{-4}, respectively. The posterior predictive mean MSE is over 8.7 times lower than that of MS, demonstrating that even state-of-the-art methods can rapidly degrade in the presence of noise.

4.2.2.2 Forced Duffing oscillator

Next, the utility of the proposed Bayesian method for learning chaotic behavior will be demonstrated. For this example, the Duffing oscillator is considered. The governing equation



(a) Training period

(b) Testing period subset

(c) Time domain error

(d) Freq. domain error

Figure 4.11: The top row shows the trajectory estimates of the MS and Bayesian methods over the duration of the training data in Fig. 4.11a and over the last 1,000 testing data in Fig. 4.11b. The bottom row shows the error between the unaltered data and the estimates in the time, Fig. 4.11c, and frequency, Fig. 4.11d, domains.

of the Duffing oscillator is

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \alpha & \delta \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \beta \begin{bmatrix} 0 \\ x^3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \gamma \cos(\omega t). \tag{4.21}$$

Depending on the values of the parameters, the solution of this system can be periodic or chaotic. In this example, the parameter values are $\alpha = 1$, $\delta = -0.3$, $\beta = -1$, $\omega = 1.2$, and $\gamma = 0.65$ following an example in [47] that yields chaotic behavior. To generate the data for this problem, an initial condition of $(x, \dot{x}) = (0, 0)$ is used, and the system is simulated for 600s before data collection to eliminate any initial transient behavior. After this initial period, the position x of the system is measured every $\Delta t = 0.25$s for 300s for a total of 1,200 data points, each with additive noise drawn from $\mathcal{N}(0, 10^{-6})$. The observation operator h and measurement noise covariance Γ are assumed to be known, and the dynamics model is the neural network architecture (4.20) from the previous example with latent space dimension $d_x = 2$. The priors are half–$\mathcal{N}(0, 10^{-4})$ on $\boldsymbol{\theta}_\Sigma$, and $\mathcal{N}(0, 0.2)$ on the remaining parameters. For sampling, 10^6 samples are drawn and half are discarded as burn-in.

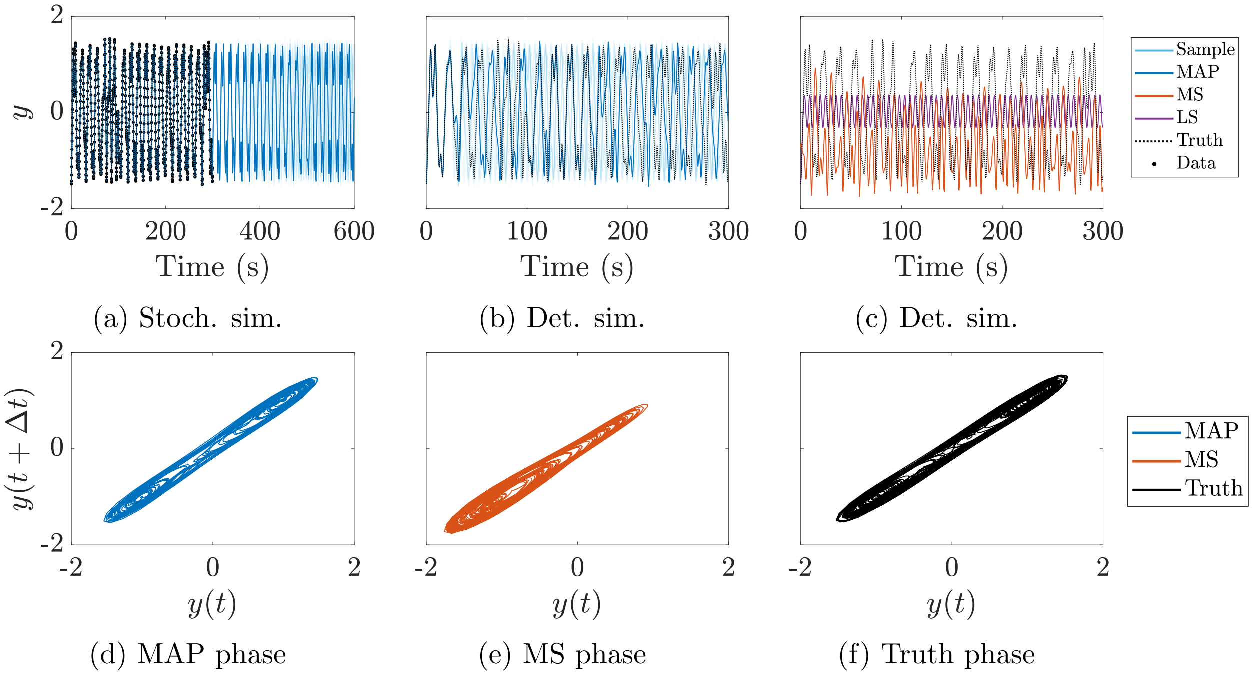
The Bayesian posterior is compared to the deterministic LS and MS objectives, and the results are shown in Fig. 4.12. For the MS objective, a time horizon of $T = 200$ was used. Smaller values of T in the range [30, 80] were tried but were found to give worse estimates. Figs. 4.12a and 4.12b show 25 posterior samples and the estimated MAP point simulated stochastically and deterministically, respectively. Fig. 4.12c shows the LS and MS estimates. The truth is plotted alongside the estimates in each figure for comparison. The LS estimate is clearly the worst in these figures out of the three, but the MSE of the LS estimate is actually lower than that of the MAP estimate. The LS estimate has an MSE of 0.7419, and the MAP estimate has an MSE of 1.2791. This shows that for certain system ID problems, especially ones including chaotic behavior, the LS objective induces a nonsensical ranking within the model space.

Although it is difficult to identify any sort of structure in the time domain of a chaotic system, the Duffing oscillator possesses an invariant set known as an attractor in phase space. Therefore, one way to assess how similar a model is to the underlying system is by comparing the phase space of the two systems. Figs. 4.12d, 4.12e, and 4.12f show the phase space of the MAP model, MS model, and truth system, respectively, over 600s. The MS and MAP models have similar shapes to the truth attractor, and both have foci near $\pm(1, 1)$ around which their outputs rotate. The MS model's attractor, however, becomes larger around its $-(1, 1)$ focus compared to both its $+(1, 1)$ focus and the truth attractor. In contrast, the structure of the MAP model visibly appears consistent with the truth attractor, suggesting that it is a more accurate representation of the truth despite its high MSE.

4.2.2.3 Allen-Cahn equation with forcing

In certain applications involving PDEs, one is not interested in the full-field solution but only in certain statistics of the full field [68, 13]. In this experiment, the goal is to learn a dynamical model of a PDE quantity of interest (QoI) that can be used for forecasting. Continuing with a focus on non-autonomous systems, we consider an example of the Allen-Cahn equation with forcing that was used in [23]. The system uses Neumann boundary conditions, and its dynamics are given as

$$\frac{\partial}{\partial t} w(\xi, t) = \sigma \frac{\partial^2 w}{\partial \xi^2} + w(1 - w^2) + \chi_\delta(\xi) u(t), \tag{4.22}$$



(a) Stoch. sim.

(b) Det. sim.

(c) Det. sim.

(d) MAP phase

(e) MS phase

(f) Truth phase

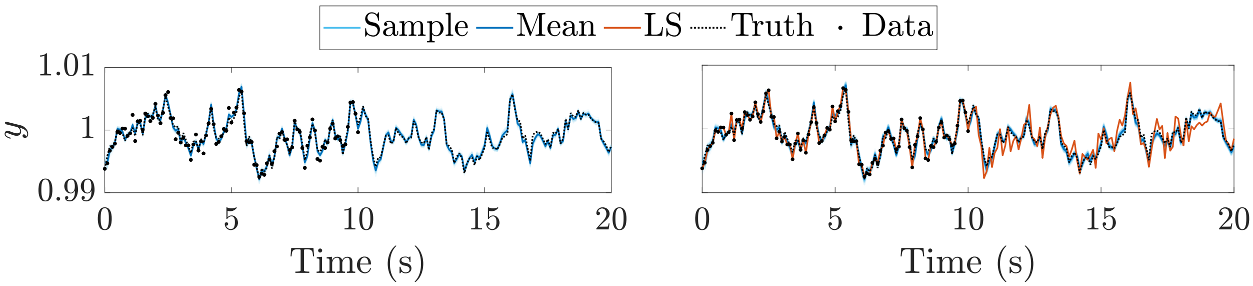
Figure 4.12: Figs. 4.12a and 4.12b compare the posterior predictive distribution and MAP to the truth, where the posterior samples and MAP are generated using stochastic and deterministic simulation respectively, and Fig. 4.12c shows deterministic simulations of the MS and LS estimates. Figs. 4.12d, 4.12e, and 4.12f show the phase space of the MAP, MS, and truth, respectively, over 600s.

where w is the flow, $\xi \in [-1,1]$ is the spatial coordinate, $t \in [0,\infty)$ is the time coordinate, u is the control input, and χ is an indicator function that takes the value one when $\xi \in \delta = [-0.5, 0.2]$ and zero otherwise. The control inputs are sampled as $u(t_k) \sim \mathcal{N}(0, 10^{-2})$ for $k = 0, \dots, n$, and a zero-order hold is assumed for intermediate time values. To generate the data for this system, a spatial mesh with 256 cells and a time discretization with $\Delta t = 0.1$s are used. Then at each timestep t_k, Eq. (4.22) is solved for $w(\xi_i, t_k)$ at each vertex ξ_i using the `solve` function in FEniCS [58]. The output of this system is the approximated second moment of the flow

$$y_k = \frac{1}{257}\sum_{i=0}^{256} w^2(\xi_i, t_k) + \eta_k, \tag{4.23}$$

where $\eta_k \sim \mathcal{N}(0, 0.2^2)$ represents sensor noise. The system is simulated using an initial condition of $w(\xi_i, 0) = 0$, $\forall i = 0, \dots, 256$, and training data collection begins at $t = 20$s since the first 20s contain a transient period where the system moves toward a stable equilibrium at ±1. After this initial period, data are collected for 10s at $\Delta t = 0.1$s intervals for a total 101 data points. The next 10s following data collection are used as a testing period.

The posterior predictive distribution is compared to a model trained using the deter-



(a) Stochastic simulation

(b) Deterministic simulation

Figure 4.13: The estimates of the Allen-Cahn QoI over the 10s training period and the subsequent 10s testing period.

ministic LS objective in Fig. 4.13. The dynamics model used by both algorithms is the neural network (4.20) used in previous examples with $d_x = 8$. The neural network has 350 parameters and there are only 101 data points, which puts many system ID algorithms at risk of overfitting. The observation operator is fixed as $\mathbf{H} = \begin{bmatrix} 1 & \mathbf{0}_{1\times 7} \end{bmatrix}$. The priors are half–$\mathcal{N}(0, 10^{-6})$ on $\boldsymbol{\theta}_\Sigma$, half–$\mathcal{N}(0,1)$ on $\boldsymbol{\theta}_\Gamma$, and $\mathcal{N}(0,4)$ on the remaining parameters. The data were normalized before training to have zero mean and standard deviation of one. In this experiment, 10^5 samples were drawn from the posterior, and half were discarded as burn-in.

Figs. 4.13a and 4.13b show 100 samples simulated stochastically and deterministically, respectively, and the 'mean' represents the mean of these sample trajectories. The LS estimate matches the data closely during the training period, but performs poorly beyond this period due to overfitting. In constrast, the mean provides a good estimate of the truth throughout the 20s time period.

The root mean squared error (RMSE) values of the mean and LS estimates on the training data and on the noiseless QoI values during the training and testing periods are given in Table 4.1. The LS estimate has a training data RMSE two orders of magnitude smaller than that of the mean estimate, but the noiseless training QoI RMSE of the LS estimate is actually worse than that of the mean estimate, indicating overfitting. Moreover, the noiseless testing QoI RMSE of the LS estimate is an order of magnitude worse than the mean estimate. Also note that the RMSE of the mean estimate on the noiseless training and testing QoIs are very similar, indicating good generalizability of the estimate. Recall that the prior used on the dynamics parameters was only weakly informative, so the improved generalizability of the mean estimate over the LS estimate comes nearly entirely from the inclusion of process noise in the likelihood. The LS objective, on the other hand, does not account for model uncertainty and implicitly assumes that the model that most closely fits the data is the best, making it prone to overfit when the model form is very expressive, as is the case here.

Table 4.1: RMSE values of the posterior predictive mean (Bayes) and deterministic LS estimate (LS) on the training data and on the noiseless QoI values during the training/testing periods.

	Training data RMSE	Training QoI RMSE	Testing QoI RMSE
Bayes	7.10×10^{-4}	3.74×10^{-4}	3.77×10^{-4}
LS	3.43×10^{-6}	8.18×10^{-4}	1.46×10^{-3}

4.3 Summary

In this chapter, we considered the identification of nonlinear systems, which can display a wide range of interesting dynamical behavior. We presented the SINDy and MS objectives and proved that they can be contained within the marginal likelihood formulation under certain assumptions. Specifically, the likelihood is equivalent to SINDy if we assume zero measurement noise and use a numerical integrator, and it is equivalent to MS if we assume zero process noise and perform joint parameter-state estimation. Notably, we showed that the smoothness properties of MS that often motivate its use can also be achieved in the marginal likelihood through proper tuning of the process noise covariance term. Moreover, the process noise covariance is a continuous and multi-dimensional parameter that therefore allows for greater flexibility in tuning.

Lastly, we showed improved estimation using the marginal likelihood over these other two objectives on a set of numerical experiments. These experiments included chaotic systems where we showed that the marginal likelihood could properly account for the exponential growth of errors, whereas SINDy could not. MS was able to account for error accumulation to a certain extent, but did not have the same flexibility as the likelihood to deliver accurate estimates. Additionally, we considered neural network parameterizations where, in some cases, there were more parameters than data points. In these examples, we demonstrated that the regularization that arises in the marginal likelihood is effective at resisting overfitting, whereas LS objectives have no means to avoid overfitting.

CHAPTER V
Hamiltonian System Identification

In some cases, we have more information available to us on the system than just the training data, and we would like to find a way to incorporate that added information into the system ID procedure. Doing so restricts the model space and improves the generalizability of estimated systems on data outside of the training set. One example of a class of systems on which we have substantial knowledge is Hamiltonian systems. These systems are defined by a Hamiltonian function from which the system's vector field can be derived. Hamiltonian systems arise as models in many science and engineering applications such as multibody dynamics and control in robotics [91], the Kozai-Lidov mechanism in astrophysics [56], particle accelerators in accelerator physics [26], 3D vortex dynamics in fluid mechanics [85], and the nonlinear Schrödinger equation in quantum mechanics [19]. Although these systems display complex nonlinear behavior, they possess an underlying highly structured geometry encoded by a Hamiltonian. Uncovering a system's Hamiltonian can reveal key insights into its physical properties such as mass or energy conservation. Learning Hamiltonian models directly from data is becoming increasingly important in diverse areas such as astrophysics, robotics, fluid dynamics, plasma physics, and quantum mechanics where first-principle modeling can yield highly complex model structures or such models are not available.

In this chapter, we consider the problem of learning Hamiltonian systems using both training data and physical knowledge. We begin by presenting the background knowledge available on these systems including Hamiltonian mechanics and their symplectic structure. Then, we cover special integrators that are able to conserve important physical properties in the evolution of a Hamiltonian system. Next, we discuss how to incorporate this background knowledge into the learning procedure and demonstrate how this knowledge improves the accuracy and precision of estimation. Additionally, we apply the marginal likelihood to a deep learning context and exhibit its robustness to different types of noise, specifically multiplicative and uniform noise.

5.1 Hamiltonian mechanics

Here we introduce the definition of a Hamiltonian system and identify numerous physical phenomena that such systems display. For a mechanical system, the Hamiltonian is defined as the total energy in the system, and it is a function of the generalized position $\mathbf{q} \in \mathbb{R}^d$, generalized momentum $\mathbf{p} \in \mathbb{R}^d$, and possibly time t. Because we are interested here in conservative systems, the Hamiltonian will be independent of time and take the form

$$\mathcal{H}(\mathbf{q}, \mathbf{p}) = \frac{1}{2}\mathbf{p}^\top \mathbf{M}^{-1}(\mathbf{q})\mathbf{p} + U(\mathbf{q}, \mathbf{p}), \tag{5.1}$$

where $\mathbf{M}$ is the inertia matrix and U the potential energy. Time derivatives of the position and momentum can be derived from the Hamiltonian according to Hamilton's equations

$$\dot{\mathbf{q}} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}}, \qquad \dot{\mathbf{p}} = -\frac{\partial \mathcal{H}}{\partial \mathbf{q}}. \tag{5.2}$$

These governing equations possess physically meaningful geometric properties that can be described in the form of reversibility, symplecticity, first integrals, and energy conservation [3]. Preservation of these qualitative features in a numerical simulation is crucial for accurate long-time prediction of Hamiltonian dynamics.

5.2 Explicit symplectic integrators

The goal of our estimation is to learn discrete-time propagators that conserve the aforementioned physical properties. As the name suggests, reversible symplectic integrators preserve reversibility and symplecticity and will serve as one basis of our learning approach. Using ideas from geometric mechanics, the field of geometric numerical integration has developed a variety of structure-preserving time integrators for Lagrangian/Hamiltonian systems, e.g., [40, 93]. We first focus on a basic model structure that can be integrated with the particularly simple leapfrog integration method, which enables us to focus on the contributions of our proposed learning approach. Then, we move to the more general case of nonseparable Hamiltonians and show similar results using the explicit symplectic integrator developed in [99].

5.2.1 Leapfrog integration

The model structure we use here assumes an autonomous system, a constant and known inertia matrix, and a potential energy function independent of the momentum. Without loss of generality, we set the inertia matrix to be the identity. These assumptions admit Hamiltonians of the form

$$\mathcal{H}(\mathbf{q}, \mathbf{p}) = \frac{1}{2}\mathbf{p}^\top \mathbf{p} + U(\mathbf{q}), \tag{5.3}$$

allowing the following simplification of Hamilton's equations:

$$\dot{\mathbf{q}} = \mathbf{p}, \qquad \dot{\mathbf{p}} = -\frac{\partial U}{\partial \mathbf{q}}. \tag{5.4}$$

Hamiltonians of this form are *separable* and therefore compatible with leapfrog integration. The leapfrog method is traditionally defined in half-step intervals for the momentum, but here we provide the equations at integer steps, which is more useful to our setting:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \Delta t \mathbf{p}_k - \frac{\Delta t^2}{2} \left.\frac{\partial U}{\partial \mathbf{q}}\right|_{\mathbf{q}_k}, \tag{5.5}$$

$$\mathbf{p}_{k+1} = \mathbf{p}_k - \frac{\Delta t}{2} \left(\left.\frac{\partial U}{\partial q}\right|_{\mathbf{q}_k} + \left.\frac{\partial U}{\partial \mathbf{q}}\right|_{\mathbf{q}_{k+1}} \right). \tag{5.6}$$

These update equations form a symplectic integrator that is also reversible [43]. As a result, they ensure that an approximate (there is an error due to discretization) Hamiltonian is conserved along the the trajectory of the dynamics.

5.2.2 Tao's explicit symplectic integrator

Many Hamiltonian systems of interest to engineers and scientists (e.g., [91, 56, 26, 85, 19]), however, do not admit this separable form. Such Hamiltonian systems are said to be *nonseparable*. Unlike the separable Hamiltonian systems, the governing equations in (5.2) cannot be further simplified for nonseparable Hamiltonians.

While explicit symplectic integration has been extensively studied for separable Hamiltonian systems and specific subclasses of nonseparable Hamiltonian systems, explicit symplectic approximations of general nonseparable Hamiltonian systems $\mathcal{H}(\mathbf{q}, \mathbf{p})$ is an active research topic. Recently, explicit symplectic integrators for arbitrary nonseparable Hamiltonian systems were presented in [99]. The derivation of these integrators is based on the idea of an extended phase space. For an arbitrary nonseparable Hamiltonian $\mathcal{H}(\mathbf{q}, \mathbf{p})$, we first introduce fictitious position $\tilde{\mathbf{q}}$ and fictitious momentum $\tilde{\mathbf{p}}$ corresponding to $\mathbf{q}$ and $\mathbf{p}$, respectively. Next, we define an augmented Hamiltonian

$$\bar{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \tilde{\mathbf{q}}, \tilde{\mathbf{p}}) \coloneqq \underbrace{\mathcal{H}(\mathbf{q}, \tilde{\mathbf{p}})}_{\mathcal{H}_a} + \underbrace{\mathcal{H}(\tilde{\mathbf{q}}, \mathbf{p})}_{\mathcal{H}_b} + \omega \cdot \underbrace{\left(\|\mathbf{q} - \tilde{\mathbf{q}}\|_2^2/2 + \|\mathbf{p} - \tilde{\mathbf{p}}\|_2^2/2\right)}_{\mathcal{H}_c}, \tag{5.7}$$

where $\mathcal{H}_a := \mathcal{H}(\mathbf{q}, \tilde{\mathbf{p}})$ and $\mathcal{H}_b := \mathcal{H}(\tilde{\mathbf{q}}, \mathbf{p})$ correspond to two copies of the original nonseparable Hamiltonian system with mixed-up positions and momenta; $\mathcal{H}_c$ is an artificial restraint; and ω is a constant that controls the binding of the two copies. The governing equations for the augmented Hamiltonian system (5.7) are

$$\begin{aligned} \dot{\mathbf{q}} &= \nabla_{\mathbf{p}} \mathcal{H}(\tilde{\mathbf{q}}, \mathbf{p}) + \omega(\mathbf{p} - \tilde{\mathbf{p}}), \quad & \dot{\mathbf{p}} &= -\nabla_{\mathbf{q}} \mathcal{H}(\mathbf{q}, \tilde{\mathbf{p}}) - \omega(\mathbf{q} - \tilde{\mathbf{q}}), \\ \dot{\tilde{\mathbf{q}}} &= \nabla_{\tilde{\mathbf{p}}} \mathcal{H}(\mathbf{q}, \tilde{\mathbf{p}}) + \omega(\tilde{\mathbf{p}} - \mathbf{p}), \quad & \dot{\tilde{\mathbf{p}}} &= -\nabla_{\tilde{\mathbf{q}}} \mathcal{H}(\tilde{\mathbf{q}}, \mathbf{p}) - \omega(\tilde{\mathbf{q}} - \mathbf{q}). \end{aligned}$$

The introduction of fictional variables $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{p}}$ along with a specific choice for the augmented Hamiltonian in (5.7) decouples the position $\mathbf{q}$ and $\mathbf{p}$ in the extended phase space, i.e., $\dot{\mathbf{q}}$ and $\dot{\mathbf{p}}$ are independent of $\mathbf{q}$ and $\mathbf{p}$, respectively. Unlike the original nonseparable Hamiltonian $\mathcal{H}(\mathbf{q}, \mathbf{p})$, the augmented Hamiltonian $\bar{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \tilde{\mathbf{q}}, \tilde{\mathbf{p}})$ is amenable to explicit symplectic integration. In this work, we use second-order explicit symplectic method $\psi^{\Delta t}$ based on Strang splitting

$$\psi^{\Delta t} := \psi_{\mathcal{H}_a}^{\Delta t/2} \circ \psi_{\mathcal{H}_b}^{\Delta t/2} \circ \psi_{\omega \mathcal{H}_c}^{\Delta t} \circ \psi_{\mathcal{H}_b}^{\Delta t/2} \circ \psi_{\mathcal{H}_a}^{\Delta t/2}, \tag{5.8}$$

where $\psi_{\mathcal{H}_a}^{\Delta t}, \psi_{\mathcal{H}_b}^{\Delta t}$, and $\psi_{\omega \mathcal{H}_c}^{\Delta t}$ are the time-Δt flow of $\mathcal{H}_a, \mathcal{H}_b$, and $\omega \mathcal{H}_c$. This allows us to obtain an explicit symplectic integrator for the augmented Hamiltonian $\bar{\mathcal{H}}$ via composition of explicit symplectic Euler substeps with step size $\Delta t/2$. Explicit update equations for these individual flows can be written as

$$\begin{aligned} \psi_{\mathcal{H}_a}^{\Delta t} &: \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \\ \tilde{\mathbf{q}} \\ \tilde{\mathbf{p}} \end{bmatrix} \to \begin{bmatrix} \mathbf{q} \\ \mathbf{p} - \Delta t \mathcal{H}_{\mathbf{q}}(\mathbf{q}, \tilde{\mathbf{p}}) \\ \tilde{\mathbf{q}} + \Delta t \mathcal{H}_{\tilde{\mathbf{p}}}(\mathbf{q}, \tilde{\mathbf{p}}) \\ \tilde{\mathbf{p}} \end{bmatrix}, \\ \psi_{\mathcal{H}_b}^{\Delta t} &: \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \\ \tilde{\mathbf{q}} \\ \tilde{\mathbf{p}} \end{bmatrix} \to \begin{bmatrix} \mathbf{q} + \Delta t \mathcal{H}_{\mathbf{p}}(\tilde{\mathbf{q}}, \mathbf{p}) \\ \mathbf{p} \\ \tilde{\mathbf{q}} \\ \tilde{\mathbf{p}} - \Delta t \mathcal{H}_{\tilde{\mathbf{q}}}(\tilde{\mathbf{q}}, \mathbf{p}) \end{bmatrix}, \\ \psi_{\omega \mathcal{H}_c}^{\Delta t} &: \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \\ \tilde{\mathbf{q}} \\ \tilde{\mathbf{p}} \end{bmatrix} \to \frac{1}{2} \begin{bmatrix} \begin{pmatrix} \mathbf{q} + \tilde{\mathbf{q}} \\ \mathbf{p} + \tilde{\mathbf{p}} \end{pmatrix} + \mathbf{R}(\Delta t) \begin{pmatrix} \mathbf{q} - \tilde{\mathbf{q}} \\ \mathbf{p} - \tilde{\mathbf{p}} \end{pmatrix} \\ \begin{pmatrix} \mathbf{q} + \tilde{\mathbf{q}} \\ \mathbf{p} + \tilde{\mathbf{p}} \end{pmatrix} - \mathbf{R}(\Delta t) \begin{pmatrix} \mathbf{q} - \tilde{\mathbf{q}} \\ \mathbf{p} - \tilde{\mathbf{p}} \end{pmatrix} \end{bmatrix}, \end{aligned}$$

where $\mathbf{R}(\Delta t) := \begin{bmatrix} \cos(2\omega \Delta t)\mathbf{I} & \sin(2\omega \Delta t)\mathbf{I} \\ -\sin(2\omega \Delta t)\mathbf{I} & \cos(2\omega \Delta t)\mathbf{I} \end{bmatrix}$. Given a nonseparable Hamiltonian $\mathcal{H}(\mathbf{q}, \mathbf{p})$ with initial condition $[\mathbf{q}(0)^\top, \mathbf{p}(0)^\top]^\top = [\mathbf{q}_0^\top, \mathbf{p}_0^\top]^\top$, we obtain explicit symplectic approximations of the dynamics by integrating the augmented Hamiltonian $\bar{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \tilde{\mathbf{q}}, \tilde{\mathbf{p}})$ with $[\mathbf{q}(0)^\top, \mathbf{p}(0)^\top, \tilde{\mathbf{q}}(0)^\top, \tilde{\mathbf{p}}(0)^\top]^\top = [\mathbf{q}_0^\top, \mathbf{p}_0^\top, \mathbf{q}_0^\top, \mathbf{p}_0^\top]^\top$. This integration method will henceforth be referred to as Tao's integrator.

5.3 Probabilistic learning of Hamiltonian systems

Next, we describe an algorithm that adapts the probabilistic approach of Chapter II to the context of Hamilton's equations. As previously mentioned, the motivation is to encode additional structure into the learning problem, when it is available, to both reduce data requirements and enforce physical constraints on the predictions.

Our goal is to design a propagator $\Psi(\boldsymbol{\theta}_\Psi)$ used in the objective that conserves certain phenomena in our problem. We achieve this goal primarily through two methods: (1) parameterizing the Hamiltonian rather than the ordinary differential equations (ODEs) so that the form of all models in our search space conserves the Hamiltonian properties, and (2) embedding the propagator with a explicit symplectic integrator so as not to violate these properties during the numerical integration necessary for training.

5.3.1 Parameterizing the Hamiltonian

First we consider how to utilize prior knowledge of the underlying system to inform the parametrization of a model. Recall from Section 5.1 that a Hamiltonian system is defined by a function of the generalized position and momentum known as the Hamiltonian. To leverage this fact, we assume that the generalized coordinate frame is known and attempt to directly learn the Hamiltonian. By estimating a model of the Hamiltonian and deriving the dynamics from this model according to (5.2), we guarantee that the predicted flow will also be Hamiltonian. This modeling choice makes the predictions more physically meaningful and also serves to restrict the model search space, which can aid optimization.

As we previously mentioned, we first restrict ourselves in this work to considering Hamiltonians of the form $\mathcal{H}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta}_\Psi) = \frac{1}{2}\mathbf{p}^\top\mathbf{p} + U(\mathbf{q}, \boldsymbol{\theta}_\Psi)$, where only the parameters of the potential energy function must be learned. Then, we remove this simplification and consider nonseparable Hamiltonians. In this case, the Hamiltonian is parameterized as a general function $\mathcal{H}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta})$. In this section, we use polynomial and neural network parameterizations, but any other nonlinear approximation approach can also be applied. We do not intend to advocate for a specific type of parameterization but rather to show how any approach can be embedded in our proposed framework.

5.3.1.1 Polynomial approximations

For separable Hamiltonians, we need only parameterize the potential energy function. Here we parameterize the potential as $U(q, \boldsymbol{\theta}_\Psi) = f_\ell(q; \boldsymbol{\theta}_\Psi)$, where f is a polynomial up to order ℓ – terms whose exponents add to greater than ℓ are not included – and $\boldsymbol{\theta}_\Psi$ are the term coefficients. Although the choice of polynomial sequence does not affect the representational capacity of a polynomial, it does affect the numerical conditioning of the estimation problem. For this reason, orthogonal polynomials are commonly chosen to express high degree polynomials [4].

Let us denote the approximate Hamiltonian by the function $\tilde{\mathcal{H}}(\mathbf{p}, \mathbf{q}, \boldsymbol{\theta})$. In the separable case, this function becomes

$$\tilde{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta}_\Psi) = \frac{1}{2}\mathbf{p}^\top \mathbf{p} + f_\ell(\mathbf{q}; \boldsymbol{\theta}_\Psi) + C, \tag{5.9}$$

where C is an arbitrary additive constant. Differentiating the potential energy to obtain the negative generalized forces we obtain $\frac{\partial U(q, \boldsymbol{\theta}_\Psi)}{\partial \mathbf{q}_i} = \frac{\partial f_\ell(\mathbf{q}; \boldsymbol{\theta}_\Psi)}{\partial \mathbf{q}_i}$.

In the general case, the approximate Hamiltonian becomes

$$\tilde{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta}_\Psi) = \Phi^\top(\mathbf{q}, \mathbf{p})\boldsymbol{\theta}_\Psi + C, \tag{5.10}$$

where $\Phi(\mathbf{q}, \mathbf{p}) \in \mathbb{R}^N$ is a vector whose ith component is the differentiable basis function $\phi_i : \mathbb{R}^{2d} \to \mathbb{R}$ for $i = 1, \ldots, N$. The gradient of the Hamiltonian follows

$$\nabla\tilde{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta}_\Psi) = \nabla\Phi^\top(\mathbf{q}, \mathbf{p})\boldsymbol{\theta}_\Psi, \tag{5.11}$$

where $\nabla\Phi(\mathbf{q}, \mathbf{p}) \in \mathbb{R}^{N \times 2d}$ is a matrix where the ith column is the gradient $\nabla\phi_i$ with respect to the state $[\mathbf{q}^\top, \mathbf{p}^\top]^\top$.

This parameterization is also considered in [107]. We provide their methodology here so that we may compare it to the marginal likelihood later on. Similarly to SINDy, this method estimates the basis coefficients by solving the linear least squares problem

$$\min_{\mathbf{c} \in \mathbb{R}^N} \|\mathbf{A}\mathbf{c} - \mathbf{b}\|_2, \tag{5.12}$$

where each element of $\mathbf{A}$ and $\mathbf{b}$ is defined as

$$a[i,j] = \frac{1}{K}\sum_{k=1}^{K}\Big(\nabla\phi_i(\mathbf{q}_k,\mathbf{p}_k)\cdot\nabla\phi_j(\mathbf{q}_k,\mathbf{p}_k)\Big), \tag{5.13}$$

$$b[i] = \frac{1}{K}\sum_{k=1}^{K}\Big(\begin{bmatrix}\dot{\mathbf{p}}_k^\top & -\dot{\mathbf{q}}_k^\top\end{bmatrix}^\top\cdot\nabla\phi_i(\mathbf{q}_k,\mathbf{p}_k)\Big), \tag{5.14}$$

for $1 \leq i,j \leq N$. Typically, data of $\dot{\mathbf{q}}$ and $\dot{\mathbf{p}}$ are not available and must therefore be numerically approximated. Following the approach of [107], we use a second-order finite difference method for this approximation. Since this method solves a linear least-squares problem, we will henceforth refer to it in this chapter as the LS method.

5.3.1.2 Nonseparable symplectic neural networks

As mentioned earlier, our learning approach is compatible with arbitrary model parameterizations. We now present a neural network architecture known as a nonseparable symplectic neural network (NSSNN) that was presented in [108] for learning nonseparable Hamiltonians from data. In that work, the Hamiltonian model was parameterized using a neural network, allowing the position and momentum time derivatives to be computed easily and efficiently with the automatic differentiation capabilities of PyTorch [73]. The idea of leveraging automatic differentiation for gradient evaluation of a Hamiltonian has previously been applied in other inference [8] and also Hamiltonian Monte Carlo [11] algorithms. The NSSNN distinguishes itself from these other methods by embedding the neural network within Tao's integrator to allow for inference of nonseparable Hamiltonians.

The NSSNN architecture and training procedure used in this work follows that which is described in the original paper. The neural network parameterizing the Hamiltonian is composed of six linear layers with sigmoid activation functions following all but the last layer. The training data is a set of input-target pairs. The inputs $(\mathbf{q}_0^{(j)},\mathbf{p}_0^{(j)})$ are a collection of measurements at times $t_0^{(j)}$, and the targets $(\mathbf{q}^{(j)},\mathbf{p}^{(j)})$ are measurements at times $t^{(j)} = t_0^{(j)} + T$ for $j = 1,\dots n$. Since the integrator returns both the physical and auxiliary position and momentum, the loss function contains a term corresponding to each of these variables. An L_1 loss is used since it was empirically shown to yield better results:

$$\mathcal{J} = \frac{1}{n}\sum_{j=1}^{n}\left\|\mathbf{q}^{(j)}-\hat{\mathbf{q}}^{(j)}\right\|_1 + \left\|\mathbf{p}^{(j)}-\hat{\mathbf{p}}^{(j)}\right\|_1 + \left\|\mathbf{q}^{(j)}-\hat{\mathbf{x}}^{(j)}\right\|_1 + \left\|\mathbf{p}^{(j)}-\hat{\mathbf{y}}^{(j)}\right\|_1. \tag{5.15}$$

To train the NSSNN, the weights of each layer are initialized with Xavier initialization, and the Adam optimizer is used to minimize the loss. The optimizer's hyperparameters are problem-dependent, and the selected values will be discussed for each experiment in Section 5.4.

5.3.2 Embedding learning with explicit symplectic integrators

To complete our learning setup, we incorporate the parameterized Hamiltonian into the Bayesian system ID framework. The key idea is to exploit knowledge of Hamiltonian systems and their structure-preserving time integrators to inform the design of the state propagator $\Psi(\mathbf{q}_k, \mathbf{p}_k; \boldsymbol{\theta}_\Psi)$. For separable systems, we do this by combining the leapfrog integrator equations (5.5) and (5.6) with the Hamiltonian parameterization (5.9) to obtain

$$\Psi(\mathbf{q}_k, \mathbf{p}_k; \boldsymbol{\theta}_\Psi) = \begin{bmatrix} \mathbf{q}_k + \Delta t \mathbf{p}_k - \frac{\Delta t^2}{2} \left.\frac{\partial U(\mathbf{q},\mathbf{p},\boldsymbol{\theta}_\Psi)}{\partial \mathbf{q}}\right|_{\mathbf{q}_k} \\ \mathbf{p}_k - \frac{\Delta t}{2}\left(\left.\frac{\partial U(\mathbf{q},\mathbf{p},\boldsymbol{\theta}_\Psi)}{\partial \mathbf{q}}\right|_{\mathbf{q}_k} + \left.\frac{\partial U(\mathbf{q},\mathbf{p},\boldsymbol{\theta}_\Psi)}{\partial \mathbf{q}}\right|_{\mathbf{q}_{k+1}} \right) \end{bmatrix}, \tag{5.16}$$

where each time that Ψ needs to be evaluated, the derivative of $U(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta}_\Psi)$ must be evaluated at both the current and the updated coordinate. Eq. (5.16) essentially embeds the learning problem within a structure that implicitly enforces that the learned model will be conservative, symplectic, and reversible – thus consistent with the true process.

For nonseparable systems, we apply the second-order explicit symplectic integrator $\psi^{\Delta t}$ from (5.8) to the parameterized nonseparable Hamiltonian (5.10) to obtain

$$\psi_{\boldsymbol{\theta}}^{\Delta t} := \psi_{\tilde{\mathcal{H}}_a(\boldsymbol{\theta})}^{\Delta t/2} \circ \psi_{\tilde{\mathcal{H}}_b(\boldsymbol{\theta})}^{\Delta t/2} \circ \psi_{\omega\tilde{\mathcal{H}}_c}^{\Delta t} \circ \psi_{\tilde{\mathcal{H}}_b(\boldsymbol{\theta})}^{\Delta t/2} \circ \psi_{\tilde{\mathcal{H}}_a(\boldsymbol{\theta})}^{\Delta t/2}, \tag{5.17}$$

where $\tilde{\mathcal{H}}_a(\boldsymbol{\theta}) := \tilde{\mathcal{H}}(\mathbf{q}, \tilde{\mathbf{p}}, \boldsymbol{\theta})$ and $\tilde{\mathcal{H}}_b(\boldsymbol{\theta}) := \tilde{\mathcal{H}}(\tilde{\mathbf{q}}, \mathbf{p}, \boldsymbol{\theta})$ correspond to two copies of the parameterized nonseparable Hamiltonian model with mixed-up positions and momenta, and $\omega\tilde{\mathcal{H}}_c$ is an artificial restraint that controls the binding of the two parameterized copies. We use $\psi_{\boldsymbol{\theta}}^{\Delta t}$ to encode Hamiltonian structure into the propagator, i.e.,

$$\Psi(\mathbf{q}_k, \mathbf{p}_k; \boldsymbol{\theta}_\Psi) := L^\dagger \circ \psi_{\boldsymbol{\theta}}^{\Delta t} \circ L, \tag{5.18}$$

where $L : [\mathbf{q}_k^\top, \mathbf{p}_k^\top]^\top \to [\mathbf{q}_k^\top, \mathbf{p}_k^\top, \mathbf{q}_k^\top, \mathbf{p}_k^\top]^\top$ duplicates the state to obtain the augmented state, and $L^\dagger : [\mathbf{q}_k^\top, \mathbf{p}_k^\top, \tilde{\mathbf{q}}_k^\top, \tilde{\mathbf{p}}_k^\top]^\top \to [\mathbf{q}_k^\top, \mathbf{p}_k^\top]^\top$ operates on the augmented state to yield the state in the original phase space. These physics-informed propagators in Eqs. (5.16) and (5.18) for separable and nonseparable Hamiltonians, respectively, ensures that the learned models will respect the geometric properties of the true process, i.e., the learned model will be a canonical Hamiltonian system.

5.3.3 Discussion and analysis

In this section, we comment and analyze the effect of the structure of Eqs. (5.16) and (5.18) on the observed dynamics. First and foremost, these propagators are structurally designed so that they are symplectic and reversible. Specifically, the learning is embedded within a symplectic integrator. This directly contrasts with similar approaches [107, 35, 8] that focus on learning a Hamiltonian but do not actually account for data generated by a conservative process. In our setting, the integrators are symplectic and reversible, and therefore, so too are the propagators within the objectives. This proves the following main result.

Theorem 5 (Symplectic and reversible)**.** *Consider the propagator Ψ parameterized according to Eq.* (5.16)*. Then the mapping $\mathbf{x}_k = (\mathbf{q}_k, \mathbf{p}_k) = \Psi^k(\mathbf{q}_0, \mathbf{p}_0; \boldsymbol{\theta}_\Psi) = \Psi^k(\mathbf{x}_0; \boldsymbol{\theta}_\Psi)$ is symplectic and reversible.*

The ramification of this result is that the objective can more accurately assess the performance of the model because the propagator enforces phenomena that are also enforced in nature.

Next, it is interesting to consider how the chosen propagator affects the marginal likelihod calculation. Primarily, it enters through the prediction step in Algorithm 1, where the probability of a future state $\mathbf{x}_{k+1}$ is the probability of its deviation from $\Psi(\mathbf{x}_k, \boldsymbol{\theta}_\Psi)$ weighted by the previous probability $\pi(\mathbf{x}_k|\boldsymbol{\theta}, \mathcal{Y}_k)$. The deviations from $\Psi(\mathbf{x}_k, \boldsymbol{\theta}_\Psi)$ are weighted by the process covariance $\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)$. This process covariance represents model error; in our setting this model error is a combination of the integration error of the numerical integrator and the error of the estimated Hamiltonian $\tilde{\mathcal{H}}$.

In another black-box approach where some non-symplectic integrator was chosen [28], there would be an additional source of error whereby the propagator would not respect the solution manifold. This source of error would grow with time, and therefore yield significant deviations causing an overestimation of the process noise covariance. In our approach, this source of deviation does not exist, and therefore we are able to learn a much smaller process noise when the model form $\tilde{\mathcal{H}}(\mathbf{q}, \mathbf{p}, \boldsymbol{\theta})$ is appropriate. We show this feature in Section 5.4.

5.4 Numerical experiments

In this section, we draw comparisons between the marginal likelihood with the physics-informed propagator and other methods for learning Hamiltonian systems. These comparison methods include the marginal likelihood with a propagator that is not physics-informed, the LS method of Section 5.3.1.1, and the NSSNN with the L_1 objective (5.15) from Section 5.3.1.2. The first example is the separable Hénon-Heiles system, in which we show that we achieve greater estimation accuracy and precision using the marginal likelihood with the physics-informed propagator compared to without. Then we consider the nonseparable Cherry problem in which we show a similar result using Tao's integrator. On this system, we also show that the marginal likelihood can outperform the LS method. Additionally, we demonstrate that we can achieve robust estimation with the additive Gaussian noise model of Eq. (2.5) even when the actual noise form is not additive Gaussian.

On the final two experiments, we consider multiplicative uniform noise and implement the filtering equations for the more general noise model of Eq. (2.9). The first of these final experiments is a toy problem that we refer to as Tao's problem. For this example, we compare using the L_1 and marginal likelihood objectives for learning a NSSNN. We will show that although the two objectives are comparable in the low noise case, the marginal likelihood delivers much better estimates once noise is introduced. Finally, we compare the L_1 and marginal likelihood objectives on the chaotic double pendulum. Despite the complexity of the system, we show the marginal likelihood can still accurately identify dynamics that evolve on the system manifold.

5.4.1 Hénon-Heiles system

The first system that we provide numerical results for is the canonical Hénon-Heiles Hamiltonian system. The Hamiltonian of this system is separable, and we use this simplified form to highlight the value of embedding physical knowledge about the data into the learning process. We will compare the approach in the previous section to a 'physics-ignorant' approach where the propagator is equipped with a non-symplectic integrator that does not conserve the Hamiltonian. This approach is akin to the derivative reconstruction approaches used by, for example, Hamiltonian neural networks [35]. We modify this physics-ignorant approach to use a classical second-order Runge-Kutta integrator to match our second-order accurate leapfrog integrator. We will sometimes refer to these approaches as the symplectic and non-symplectic approaches, respectively. We also point out that one can learn a Hamiltonian with the physics-ignorant approach and then predict with it using a symplectic integrator. Indeed this is the approach that seems to be espoused by a majority of the literature earlier in this chapter. Here we will show that this approach leads to a poorer reconstruction of the Hamiltonian because it neglects the symplectic process actually generating the data.

For generating the "true" data in each simulation, we use a leapfrog integrator with conservatively fine timestep $\Delta t = 10^{-3}$ to minimize integration error. For training, we use a large timestep of $\Delta t_t = 0.25$ to reduce computation time and demonstrate the robustness of our approach. In the results that follow, we will refer to the timestep $\Delta t_t = 0.25$ used for learning as the 'learning timestep,' and the finer timestep $\Delta t_f = 0.01$ used for prediction after learning as the 'fine timestep.'[1] Using a smaller timestep during testing ensures that the learned models have captured the continuous dynamics of the system and not only a discrete mapping at a single timestep [54].

The Hénon and Heiles [42] potential is given as

$$U(q_1, q_2) = \frac{1}{2}q_1^2 + \frac{1}{2}q_2^2 + q_1^2 q_2 - \frac{1}{3}q_2^3. \tag{5.19}$$

After differentiation, we obtain the momentum derivatives

$$\dot{p}_1 = -q_1 - 2q_1 q_2, \qquad \dot{p}_2 = -q_2 - q_1^2 + q_2^2. \tag{5.20}$$

For the initial condition, we select the point $(q_1, q_2, p_1, p_2)(0) = \left(\frac{1}{5}, \frac{-1}{4}, 0.3, -\frac{-1}{4}\right)$.

During learning, we parameterize the unknown potential with monomials up to total order three such that Hamilton's equations for the momenta in the learning system are

$$\dot{p}_1 = \theta_1 + 2\theta_3 q_1 + \theta_4 q_2 + 3\theta_6 q_1^2 + 2\theta_7 q_1 q_2 + \theta_8 q_2^2, \tag{5.21}$$

$$\dot{p}_2 = \theta_2 + \theta_4 q_1 + 2\theta_5 q_2 + \theta_7 q_1^2 + 2\theta_8 q_1 q_2 + 3\theta_9 q_2^2. \tag{5.22}$$

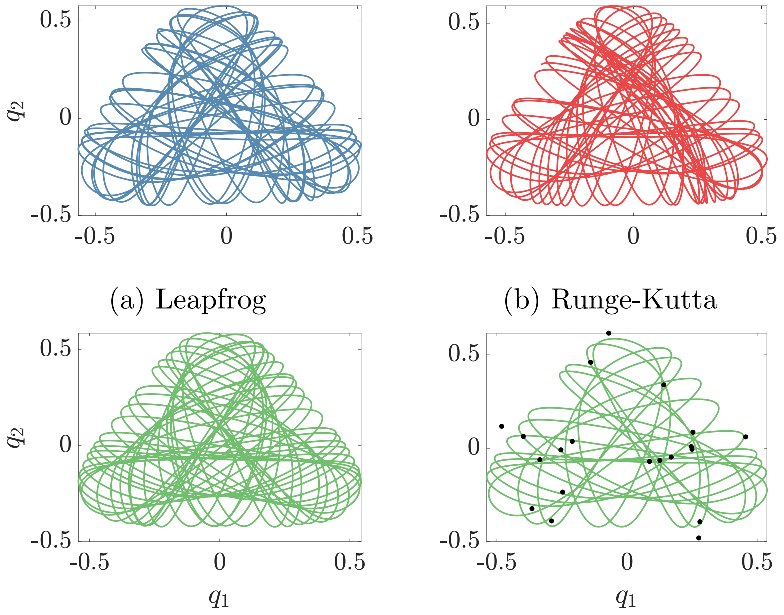
Note that these equations share parameters because they are derived from a single parameterized potential function. We also parameterize the process and measurement noise as $\mathbf{\Sigma}(\theta) = \theta_{10}\mathbf{I}_4$ and $\mathbf{\Gamma}(\theta) = \theta_{11}\mathbf{I}_4$. Finally, data are collected on the full state $n = 20$ times at regular intervals over 100 seconds with noise standard deviation of $\sigma = 0.05$.

The MAP points from each posterior are used to simulate the system for 200 seconds using the leapfrog integrator with the fine timestep, and the resulting phase plots are shown alongside the truth and the data points in Fig. 5.1. We see that both approaches learn that the system evolves on a manifold because the parameterization discussed in 5.3 restricts the model search space to Hamiltonian systems, each of which possesses a corresponding manifold.

From visual inspection, it first appears that the manifolds learned by both the symplectic and non-symplectic approaches are virtually identical. Fig. 5.2a, however, reveals that the system learned by the non-symplectic approach exists on a lower energy manifold than that learned by the symplectic approach. The true Hamiltonian is 0.1227, and the Hamiltonian learned by the symplectic approach is 0.1219 and by the non-symplectic approach is 0.1211.

The explanation for this discrepancy is shown in Fig. 5.2b, where we see that the Hamiltonian of the system grows very quickly when the Runge-Kutta integrator with the learning

[1] After learning a Hamiltonian, any relevant numerical scheme can be used for prediction.



(a) Leapfrog

(b) Runge-Kutta

(c) True

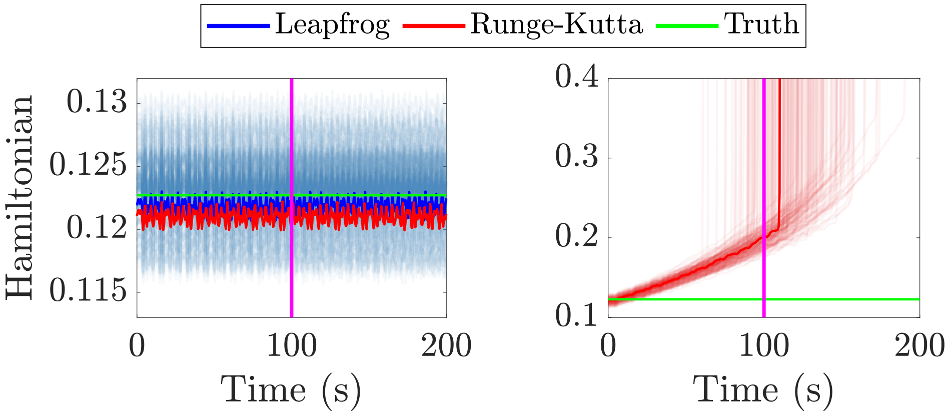
(d) Position Data

Figure 5.1: Phase diagrams of the Hénon-Heiles system. The top row shows the models learned by the approaches equipped with the leapfrog and Runge-Kutta integrators. All trajectories were integrated with the leapfrog method with the fine timestep for 200s, except for the bottom right, which was only integrated for the 100s period of data collection.

timestep is used. To compensate for this growth, the non-symplectic approach learns a lower energy system such that in the time it takes to integrate from one data point to the next, the energy added by the integrator has now pushed the Hamiltonian up to its true value. Overall, the Hamiltonian learned by the symplectic approach is closer to the truth than that learned by the non-symplectic approach.

In addition to accuracy, the second main benefit to using a symplectic integrator is a reduction of uncertainty in one's estimate. To quantify the uncertainty in our estimates, we run MCMC sampling on each posterior for 200,000 iterations and discard the first 100,000 samples as a conservative burn-in period. Fig. 5.3 shows the estimates of the trajectory of the system's first state and their corresponding posteriors. For the sake of direct comparison, each trajectory is integrated using the leapfrog method with the fine timestep. In this figure, the region between the 2.5% and 97.5% quantiles of the posterior is shaded and 100 individual samples have been overlaid. The posterior from the non-symplectic approach grows much more quickly than the posterior from the symplectic approach as the system evolves, reflecting greater uncertainty in the estimate of the non-symplectic approach.

To understand this discrepancy in the variance of the estimates, we turn to the shape of the log posterior. The inability of the Runge-Kutta integrator to preserve the Hamiltonian



(a) Leapfrog Integrator

(b) Runge-Kutta Integrator

Figure 5.2: The learned Hénon-Heiles Hamiltonians. The lighter blue and red lines are samples from the posteriors from the symplectic and non-symplectic learning approaches. The dark lines represent the MAP of the respective parameter posterior, and the subcaptions denote the integrator used to integrate each solution post-learning, e.g. the two dark red lines are the same model integrated with different methods. The vertical magenta line indicates the end of the data collection period.

of the system introduces error into the system learning process not present in the process using the leapfrog integrator. This additional error increases the process noise of the system, shown in Fig. 5.4, and, as discussed in [28], this order of magnitude difference in the process noise can introduce significant bias. Indeed, we see in Fig. 5.3 that the MAP estimate from the symplectic approach aligns closely with the truth for about 150 seconds, while the non-symplectic approach can only last about 20 seconds.

5.4.2 Cherry problem

Next, we apply the Bayesian learning method to the Cherry problem [17], which has a nonseparable Hamiltonian of the form

$$\mathcal{H}(q_1, q_2, p_1, p_2) = \frac{1}{2}(q_1^2 + p_1^2) - (q_2^2 + p_2^2) + \frac{1}{2}p_2(p_1^2 - q_1^2) - q_1 q_2 p_1. \tag{5.23}$$

This four-dimensional dynamical system is a challenging example because it possesses a negative energy mode (NEM) that leads to an explosive nonlinear growth of perturbations for arbitrarily small disturbances. These NEMs occur in several important infinite-dimensional dynamical systems, e.g., gravitational instability of interpenetrating fluids [12] and magnetosonic waves in the solar atmosphere [46].

Similar to the Hénon-Heiles example, we will begin by comparing the symplectic and

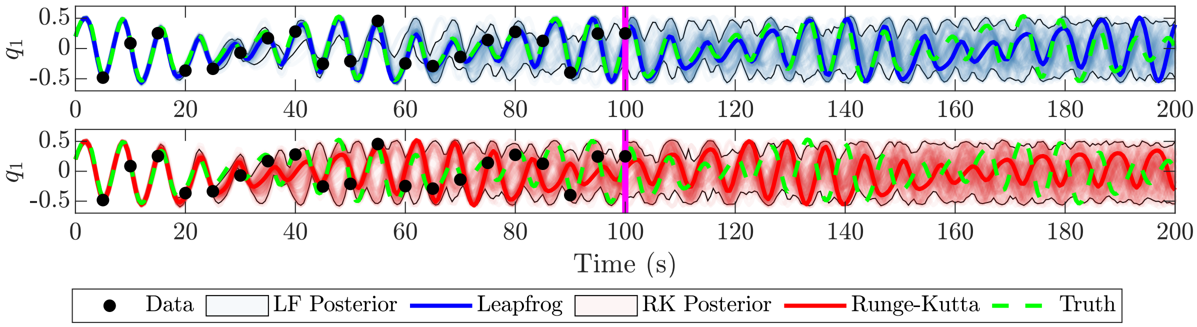
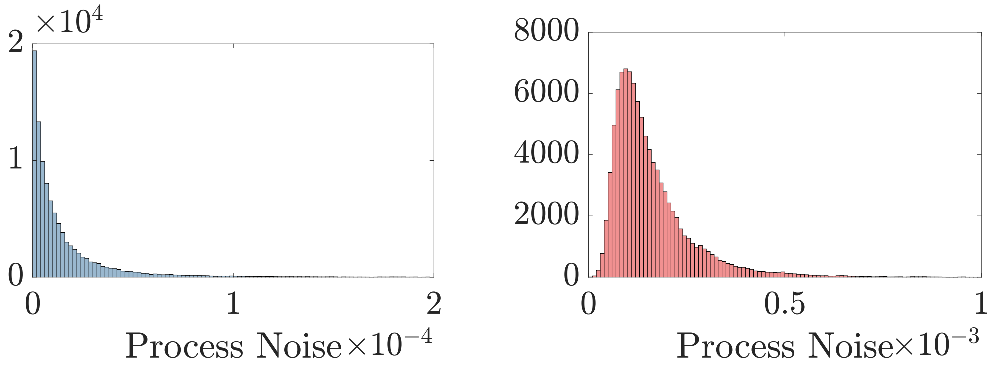


Figure 5.3: Hénon-Heiles reconstructed trajectories using the MAP point and samples from the posterior. The top row shows the results from the symplectic approach and the bottom row shows results from the non-symplectic approach, each integrated with leapfrog. The vertical magenta line indicates the end of the data collection period.



(a) Leapfrog

(b) Runge-Kutta

Figure 5.4: Marginal posteriors of the process noise. Due to the symplectic nature of the leapfrog integrator, the process noise is reduced by an order of magnitude.

non-symplectic Bayesian approaches, where this time, the symplectic integrator is now Tao's integrator. Since this integrator is also second-order, we keep the second-order Runge-Kutta integrator for the non-symplectic approach. The results of this comparison are provided in Section 5.4.2.1 and show the improvements in accuracy gained by using a structure-preserving symplectic integrator over a non-symplectic integrator of equivalent order accuracy. Section 5.4.2.2, gives a comparison between the proposed method and the LS method described in 5.3.1.1. This section demonstrates the greater robustness of the proposed approach to noisy data.

In both sections, we parameterize the Hamiltonian with polynomials up to total order three for a total of $N = 34$ basis functions. The type of polynomials used in each example is given in the corresponding section. Each covariance matrix is parameterized as an identity matrix scaled by an unknown parameter: $\mathbf{\Sigma}(\boldsymbol{\theta}_\Sigma) = \theta_{35}\mathbf{I}_4$ and $\mathbf{\Gamma}(\boldsymbol{\theta}_\Gamma) = \theta_{36}\mathbf{I}_4$. Once again, we place a Laplacian prior on $\boldsymbol{\theta}_\Psi$ for sparsity and half-normal priors on $\boldsymbol{\theta}_\Sigma$ and $\boldsymbol{\theta}_\Gamma$. The observation operator h is the identity such that we measure the full state. The data and predictions are both generated using the explicit symplectic integrator using a time step of $\Delta t_f = 0.01$.

5.4.2.1 Training with a single initial condition

In this study, we consider the case where noisy data from a single initial condition are used to learn the model, and we compare numerical results far outside the training data regime to demonstrate the generalizability of the presented Bayesian learning algorithm.

We select the initial condition $[q_1(0), q_2(0), p_1(0), p_2(0)]^\top = [0.15, 0.1, -0.05, 0.1]^\top$ for training and integrate the nonseparable Hamiltonian system up to $t = 8$. We collect the data on the full state every 0.4 time units for a total of 21 data points with zero-mean Gaussian noise with standard deviation of $\sigma = 0.01$. For the learning problem, we parameterize the nonseparable Hamiltonian using monomial basis functions. To ensure proper convergence, we draw 2×10^5 samples, and discard the first 10^5 samples as a conservative burn-in period.

Both symplectic and non-symplectic approaches in this study use a learning time step of $\Delta t_t = 0.05$ (five times the time step used for prediction) to reduce computational time. Despite the large learning step, the symplectic approach gives accurate predictions far outside the training data regime, see Fig. 5.6a. For the non-symplectic approach, we use a second-order Runge-Kutta integrator that does not preserve the underlying Hamiltonian structure.

The MAP parameters of each posterior are used to simulate for $t = 16$ (100 % outside the training data regime) using the explicit symplectic integrator. Fig. 5.5 shows the learned Hamiltonian values for the MAP point and their corresponding posteriors for both symplectic and non-symplectic approaches. The true Hamiltonian is -0.78×10^{-2}, and the value of the Hamiltonian learned by the symplectic approach is approximately -0.68×10^{-2}. The Hamiltonian learned by the non-symplectic approach is approximately -0.81×10^{-2}. Compared to the posterior from the symplectic approach in Fig. 5.5a, the posterior from the non-symplectic approach in Fig. 5.5b reflects greater uncertainty in the Hamiltonian estimate.

Fig. 5.6 compares the reconstructed trajectories of the system's first state using the MAP point and the samples from the posterior. For a fair comparison between the symplectic and non-symplectic approaches, each trajectory is integrated using the explicit symplectic integrator.[2] We see in Fig. 5.6 that the MAP estimate from the symplectic approach aligns

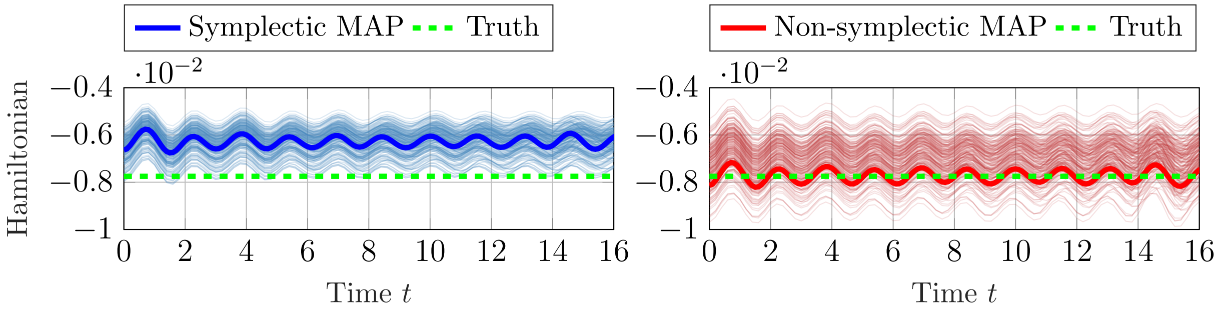
[2] We assume that a non-symplectic learning approach would still integrate the learned model with a symplectic integrator. This setting is similar to the approaches considered in [35] or [112] where the authors use non-symplectic integrators for training structure-preserving neural networks and then integrate the learned

closely with the truth for the entire length of the simulation, while the MAP point from the non-symplectic approach gives inaccurate results after the end of the training time interval at $t = 8$.

We also compute the following relative state error for both the training data regime and testing data regime

$$e(i:j) = \frac{\|\hat{\mathbf{x}}_{i:j} - \mathbf{x}_{i:j}\|_F}{\|\mathbf{x}_{i:j}\|_F} \tag{5.24}$$

for positive integers $i \leq j$ where $\mathbf{x}_{i:j} \in \mathbb{R}^{2d_x \times (j-i+1)}$ is the true state trajectory at times $t_i, \dots, t_j$, and $\hat{\mathbf{x}}_{i:j}$ is the estimate of this trajectory. In the training data regime ($t = 0$ to $t = 8$), the relative state error, $e(1:n)$, for the symplectic approach is 0.0877 whereas the relative state error for the non-symplectic approach is 0.1307. In the testing data regime ($t = 8$ to $t = 16$), the relative state error, $e(n+1:2n)$, for the symplectic approach is 0.2173 whereas the relative state error for the non-symplectic approach is 0.3326. Thus, the symplectic approach achieves approximately 30% reduction in state error over the non-symplectic approach.



(a) Symplectic approach

(b) Non-symplectic approach

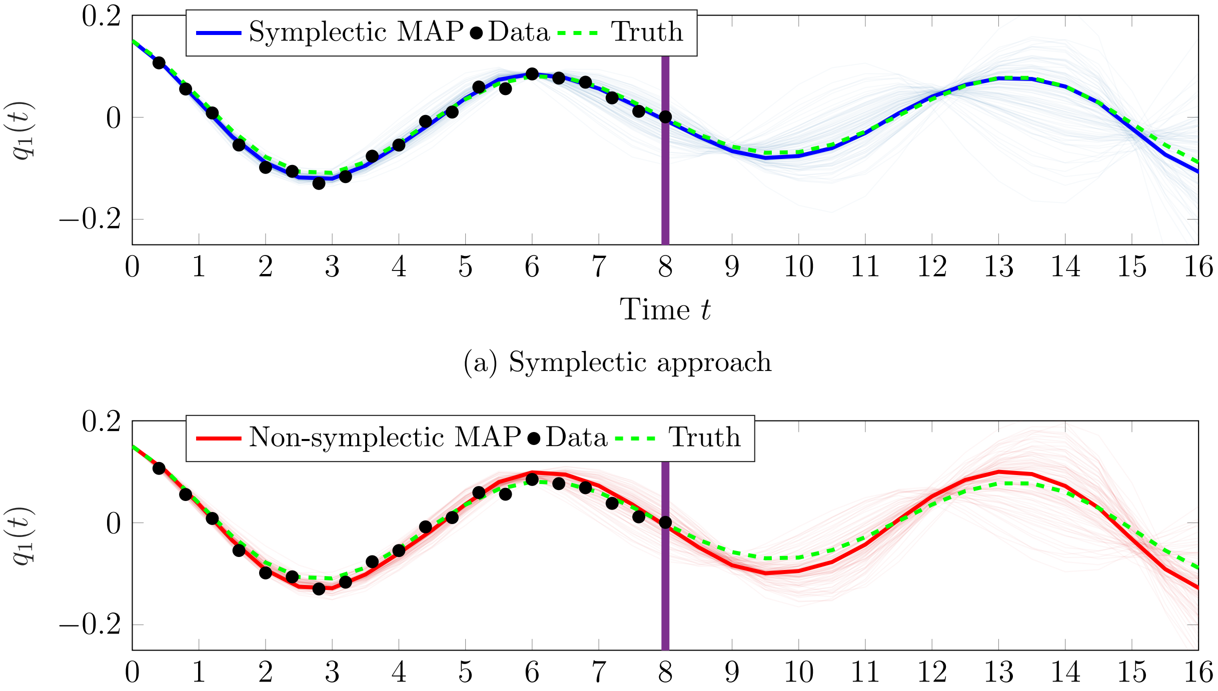
Figure 5.5: The learned Hamiltonians. The lighter blue and red lines in (a) and (b) are the samples from the posteriors from the symplectic and non-symplectic learning approaches. Both approaches estimate the true Hamiltonian accurately with an absolute energy error of less than 10^{-3}. However, the non-symplectic approach has much greater uncertainty.

5.4.2.2 Training with multiple initial conditions

In this subsection, we consider the case where data are collected from a number of different trajectories with independent initial conditions (ICs), and our goal is to estimate a model of the system that can give good prediction for an arbitrary IC.

To gather the training data, we first randomly sample $M = 5$ ICs from a Gaussian centered at the testing IC with standard deviation 0.05. The testing IC is the same IC used

Hamiltonian models using symplectic integrators.



(a) Symplectic approach

(b) Non-symplectic approach

Figure 5.6: Reconstruction error comparison for the training with single IC case. The MAP estimate for the symplectic approach captures the nonseparable Hamiltonian dynamics accurately at $t = 16$ which is 100% outside the training data whereas the MAP estimate for the non-symplectic approach yields inaccurate predictions outside the training data. The posterior for both approaches grows as the system evolves further in time. The purple line indicates the end of the data collection period.

in the previous example. For each training IC, we integrate the system for up to $t = 8$ and measure the state every 0.4 time units for a total of 21 data points per trajectory including the IC. Any trajectories that diverged during this time period had their IC resampled. To complete the training dataset, we add 10% relative noise, i.e., $\mathcal{Y}_k = \mathcal{X}_k(1 + u_k)$ where $u_k \sim \mathcal{U}[-0.10, 0.10]$ and $\mathcal{U}$ is the uniform distribution. This is the same noise form used by [107]. Note that although we are adding multiplicative noise here, we are still using the additive Gaussian model from Eq. (2.5) to show the algorithm's robustness in the event that the actual noise does not match our model.

We parameterize the Hamiltonian for the learning problem with Legendre polynomials as basis functions and integrate with the symplectic integrator with $\Delta t_t = 0.01$ during training. To find the MAP, we use the LS estimate as the optimization starting point for $\boldsymbol{\theta}_\Psi$ and the starting point 10^{-6} for variance parameters $\boldsymbol{\theta}_\Sigma$ and $\boldsymbol{\theta}_\Psi$. Then we draw 10^4 samples from the posterior.

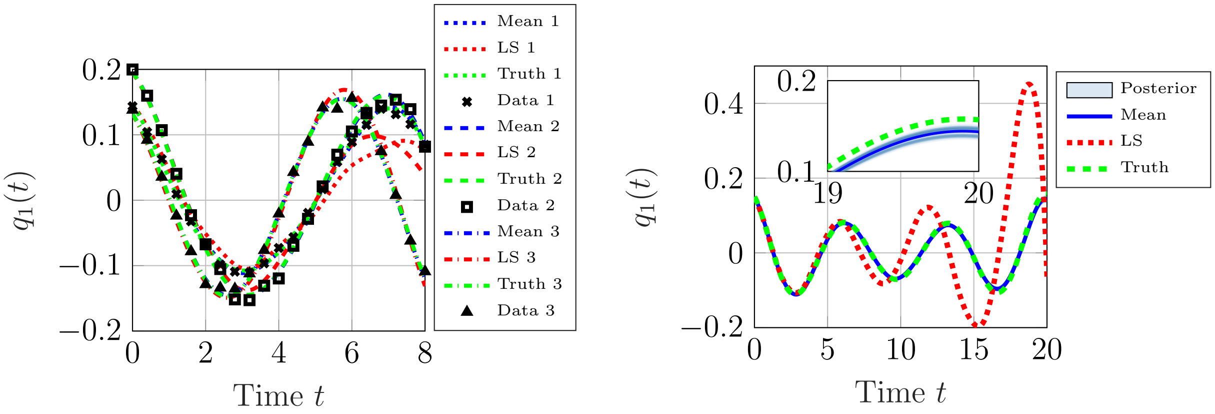
We compare the proposed structure-preserving learning method to the LS method and present the results in Fig. 5.7, where every learned model was integrated with the explicit symplectic integrator. Note that although [107] suggested a method for denoising the data, we found that using the raw data gave a better LS estimate in this problem. The training data and ground truth are shown alongside the point estimates from the LS and Bayesian methods in Fig. 5.7a. Even though the training ICs are all near the testing IC, we observe that the testing trajectory in Fig. 5.7b differs significantly from the training trajectories. Fig. 5.7a also shows that the mean estimate is able to fit the training data fairly well, whereas the LS estimate struggles due to the noisiness/sparsity of the data. The fact that the mean estimate can fit the training data even when the measurement noise is non-Gaussian further demonstrates the robustness of the proposed approach.

Next, we illustrate the posterior predictive distribution on the test IC in Fig. 5.7b by plotting the trajectory estimate from every 200th MCMC sample for a total of 50 samples. We see that as time increases, the posterior starts to widen, reflecting the growing uncertainty in the state with time. We also observe that although both the LS and posterior predictive mean match the truth very closely through $t = 4$, the LS estimate deviates from the truth much earlier and much more drastically than the mean estimate. Fig. 5.7b demonstrates that the Bayesian method is able to find models that generalize well outside of the training set of the data.

To compare the methods quantitatively, consider the relative error (5.24) over the first k time steps, i.e., $e(1 : k)$. The relative error of the mean estimate on the test IC stays under 10% for $t = 18.22$, while the LS estimate stays under this threshold for only $t = 1.49$. The Bayesian method therefore yields better long-time prediction performance.

5.4.3 Tao's example

We now turn to neural network parameterizations. Due to their large numbers of parameters, deep neural networks are often at risk of overfitting training data. This risk is especially problematic when the size of the training dataset is small and the data are significantly noisy. In the previous chapters, it was shown that Algorithm 1 can help counteract overfitting by penalizing models that have large output covariances. In contrast, objectives that do not account for a model's output covariance, like Eq. (5.15), have no way of distinguishing between high and low variance models without a sufficiently large number of data. In this experiment, we assess the potential of the negative log posterior for extending the applicability of the NSSNN into problems where the available data are few and noisy. Similar to the previous example, we will demonstrate the negative log posterior's robustness to non-



(a) Training set

(b) Testing set

Figure 5.7: Left: The training data from 3 different ICs are shown alongside the point estimates from the LS [107] and Bayesian methods. The Bayesian method is more robust to the noisiness/sparsity of the data. Right: The posterior predictive distribution and its mean are compared to the LS point on a trajectory outside the training set. The Bayesian method gives a good prediction over $t = 20$, while the LS estimate deteriorates rapidly after about $t = 5$.

Gaussian noise by using measurements with uniformly-distributed noise. In this experiment, however, we use the multiplicative noise model given in Eq. (2.9) and its associated filtering equations.

For this analysis, we would like to to eliminate as many sources of difficulty for the training as possible in an effort to isolate the effects that the noise and dataset size have on training. We therefore consider a Hamiltonian with one of the simplest nonseparable forms [99]:

$$\mathcal{H}(q,p) = \frac{1}{2}(q^2+1)(p^2+1). \tag{5.25}$$

Next, we generate the training data using Tao's integrator with initial condition $\mathbf{x}_0 = \begin{bmatrix} 0 & -3 \end{bmatrix}^\top$ and timesteps $\Delta t_f = 10^{-3}$ and $\Delta t_t = 10^{-2}$. The number of data points n takes the values $n = 300, 400, \ldots, 1000$. The period for this system is approximately 3.26, so at this Δt_t, the dataset timespans range from slightly under a single period to just over three periods. Then, we corrupt the datasets with multiplicative noise $\mathbf{v}_k \sim \mathcal{U}[1-a, 1+a]$ for $a = 0.00, 0.01, \ldots, 0.10$. The mean and variance of this noise are $\bar{\mathbf{v}} = \mathbf{1}$ and $\mathbf{R} = \frac{a^2}{3}\mathbf{I}_{2d}$. The final collection of datasets includes every (a, n) combination for a total 88 datasets.

To train the NSSNN, we use the Adam optimizer [51] with an initial learning rate of 0.05 and beta values $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Each model is trained for 400 epochs with the learning rate multiplied by 0.8 every 20 epochs. Following the approach presented in the original NSSNN paper, a batch size of 512 is used with the L_1 objective whenever applicable.

For the negative log posterior, the process noise variance is parameterized as $\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \theta_q \mathbf{I}_{2d_x}$, and the output model and measurement noise variance are assumed to be known. The parameter θ_q is initialized at a value of 10^{-3}, and a learning rate of $0.5\theta_q$ and beta values $\beta_1, \beta_2 = 0.1$ are used for optimization. To encourage small values, the prior half-$\mathcal{N}(0, 10^{-12})$ is placed on θ_q. The positivity constraint is enforced by setting θ_q to be 0.9 times its previous value whenever the optimizer places θ_q below zero. For the sake of direct comparison, we place an improper uniform prior on the dynamics model parameters.

Once training is completed, each model is simulated starting at $\mathbf{x}_0$ for twice the timespan of the training data using timestep Δt_f. The accuracy of the model is then assessed by computing the MSE defined as

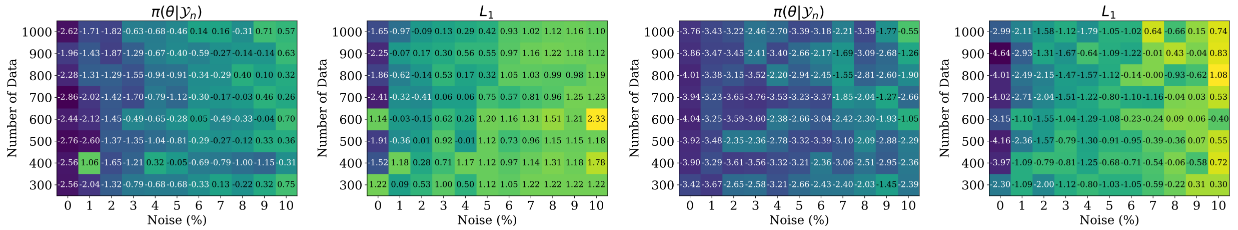
$$\text{MSE} = \frac{1}{2n} \sum_{k=0}^{2n-1} (\hat{q}_k - q_k)^2 + (\hat{p}_k - p_k)^2, \tag{5.26}$$

where the hat $\hat{}$ denotes an estimated value.

The results of the training can vary due to the randomness in the initialization of the NSSNN parameters, in the mini-batching procedure, and in the measurement noise. Sometimes, the optimizer can even get stuck at poor model estimates that yield very high MSE. To mitigate the effect of these outliers in our analysis, 20 models are trained at each (a, n) pair, and we compare the two methods using the median and minimum MSEs. Each of these 20 rounds of training has different realizations of initial parameters, mini-batches, and measurement noise.

The $\log_{10}$ of the median and minimum MSEs are presented as heatmaps in Figs. 5.8a and 5.8b, respectively. We see that using the negative log posterior to train the NSSNN results in lower median and minimum MSE at nearly every (a, n) pair. The primary exception is the minimum MSE in the 0% noise column in which the posterior yields lower MSE at roughly half of the n values. Moreover, in both figures the MSE of models trained with L_1 loss increases much more steeply once noise is added compared to the MSE of models trained with the posterior. These results demonstrate that the posterior is much more robust to noise than the L_1 loss. Additionally, the median MSE shows that even in the low noise case, the posterior can more reliably find good model estimates, especially when a smaller number of data are available.

To get a sense of how the model behavior varies with the noise and number of data, we plot the estimated trajectories of models corresponding to the four corners of the heatmaps. The estimated output of the median and minimum MSE models are shown in Figs. 5.9 and 5.10, respectively. The fuchsia line corresponds to the time t_{2n} at which the MSE evaluation ends. Visually, we see that the estimate produced with the negative log posterior can accurately

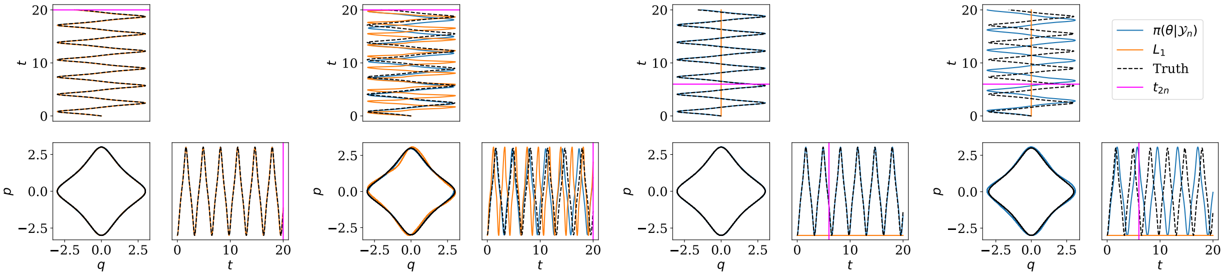


(a) Median (b) Minimum

Figure 5.8: $\log_{10}$ MSE of models trained using $-\log \pi(\boldsymbol{\theta}|\mathcal{Y}_n)$ and the L_1 norm as objective functions.

reconstruct the phase manifold in all eight of the figures. Where this estimate noticeably struggles is in the oscillation frequency of its median MSE estimates when the noise is high. However, the minimum MSE estimates at high noise do not have this issue.

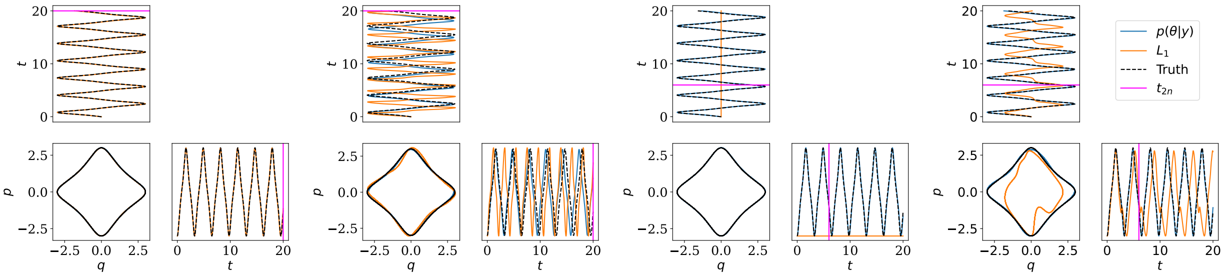
The L_1 objective struggles in two main areas: identifying good models with high noise data and reliable optimization with low number of data. We see that in the high noise case, both the median and minimum MSE estimates produced by the L_1 objective struggle to accurately reconstruct the phase manifold. In the low number of data case, the minimum MSE estimate matches the true output closely when there is not any noise. However, the median MSE estimate produces a flat line in the time domain. The fact that there is such a big difference in the median and minimum MSE estimates suggests that although the L_1 objective is able to produce good estimates with a small number of data, optimization is challenging and will likely take multiple tries.



(a) $a = 0\%$, $n = 1000$ (b) $a = 10\%$, $n = 1000$ (c) $a = 0\%$, $n = 300$ (d) $a = 10\%$, $n = 300$

Figure 5.9: Estimated trajectories from the median MSE models.

In Fig. 5.9b, we noted that the posterior estimate appears to match the true output closely in phase space but produces large errors in the time domain. One way to assess how closely the estimates match the system behavior in phase space is by looking at the



(a) $a = 0\%$, $n = 1000$ (b) $a = 10\%$, $n = 1000$ (c) $a = 0\%$, $n = 300$ (d) $a = 10\%$, $n = 300$

Figure 5.10: Estimated trajectories from the min. MSE models.

Hamiltonian MSE of the models defined as

$$\frac{1}{2n}\sum_{k=0}^{2n-1}\Big(\mathcal{H}(\hat{q_k},\hat{p_k}) - \mathcal{H}(q_0,p_0)\Big)^2, \tag{5.27}$$

where $\mathcal{H}$ is the Hamiltonian function (5.25). This metric gives a quantification for how close the energy levels of the estimates are to the true system's energy. Since some of the median MSE estimates, such as that in Fig. 5.9c, remain extremely close to the initial point, we will only asses the minimum MSE estimates with this metric. The $\log_{10}$ values of the Hamiltonian deviation for the minimum MSE estimates are shown in as a heatmap in Fig. 5.11. From this figure, we can see a clear trend that the Hamiltonian deviation grows as noise increases and as the number of data decreases in both objective's estimates. The estimates produced by the L_1 objective outperform those produced by the posterior at almost all n values when the data are noiseless – the notable exception being when the number of data is smallest. Once noise is added, however, we once again see a sharp increase in the MSE from the estimates found with the L_1 objective. The MSE values of the estimates produced with the negative log posterior, in contrast, increase gradually as the noise increases.

Overall, we conclude that if the data are truly noiseless, then the L_1 objective is likely the better option because it gives time domain MSE comparable to the negative log posterior at a lower computational cost. When the number of data is low, however, the negative posterior could be more cost efficient since it appears to require fewer training restarts than when the L_1 objective is used. For data that are noisy, the negative log posterior is the clear choice. Although the cost is higher, the negative log posterior produces significantly more accurate estimates when the data have as little as 1% noise.

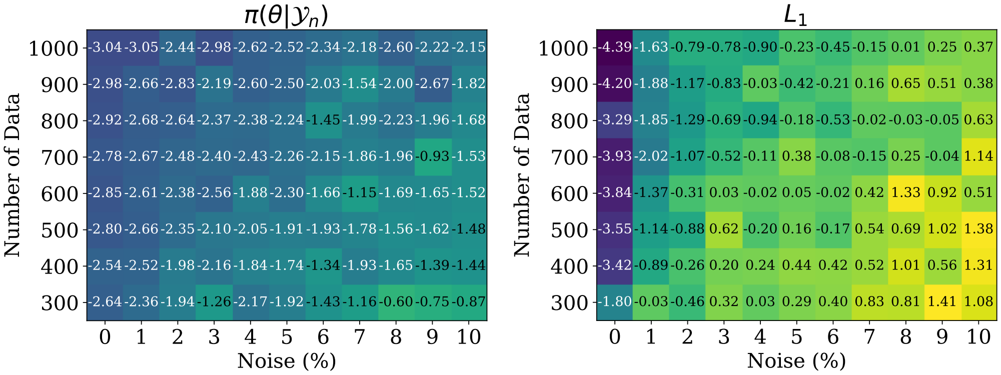


Figure 5.11: $\log_{10}$ Hamiltonian MSE of the minimum MSE models learned by the $-\log \pi(\boldsymbol{\theta}|\mathcal{Y}_n)$ and L_1 norm objectives.

5.4.4 Double pendulum

The next system that we consider is the double pendulum, which has the following Hamiltonian:

$$\begin{aligned}\mathcal{H}(\mathbf{q},\mathbf{p}) = &\frac{m_2\ell_2^2p_1^2 + (m_1+m_2)\ell_1^2p_2^2 - 2m_2\ell_1\ell_2p_1p_2\cos(q_1-q_2)}{2\ell_1\ell_2m_2\left(m_1+m_2\sin^2(q_1-q_2)\right)} \\ &- (m_1+m_2)g\ell_1\cos(q_1) - m_2g\ell_2\cos(q_2).\end{aligned} \tag{5.28}$$

We choose this system because it pertains to a physical mechanical system and therefore holds relevance to fields that design and study mechanical systems such as robotics. Additionally, the Hamiltonian is much more complex, and the behavior displayed by the double pendulum is chaotic in certain regions of the phase space. These two aspects make the system much more challenging to learn and allow us to test the limits of the NSSNN and the two objective functions.

For this experiment, we use an initial condition of $\mathbf{x}_0 = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^\top$ and collect $n = 2000$ measurements of the full state using timesteps $\Delta t_f = 10^{-3}$ and $\Delta t_t = 10^{-2}$. Then, we corrupt the measurements using multiplicative noise $\mathbf{v}_k \sim \mathcal{U}[0.99, 1.01]$. The training procedure uses 1,000 epochs with an initial learning rate of 0.05 that is multiplied by 0.8 every 50 epochs. To learn the process noise covariance, we use the parameterization $\mathbf{\Sigma} = \text{diag}(\begin{bmatrix}\theta_{q1} & \theta_{q2} & \theta_{q3} & \theta_{q4}\end{bmatrix})$. The learning rate for each parameter equals the parameter value and is also multiplied by 0.8 every 50 epochs. Any remaining aspects of the training follow the procedure described in the previous example.

First, we examine the time-domain estimates, plotted in Fig. 5.12. Since the double pendulum is a chaotic system, long-term prediction is incredibly difficult. We see that neither estimate can follow the truth for more than about 5s. We also examine the log posterior estimate with a quantification of uncertainty. Quantifying the uncertainty in deep neural networks is notoriously difficult, but with an estimated process noise available, we can run a stochastic simulation that gives us uncertainty quantification without the immense expense of MCMC sampling. The stochastic simulation is shown in Fig. 5.13. We see that the spread of samples begins to grow as the MAP estimate begins to deviate from the truth. This suggests that the process noise can give a reliable estimate of how the uncertainty in an estimated model changes over time. Notably, non-Bayesian approaches have no such resource for assessing the reliability of their estimates.

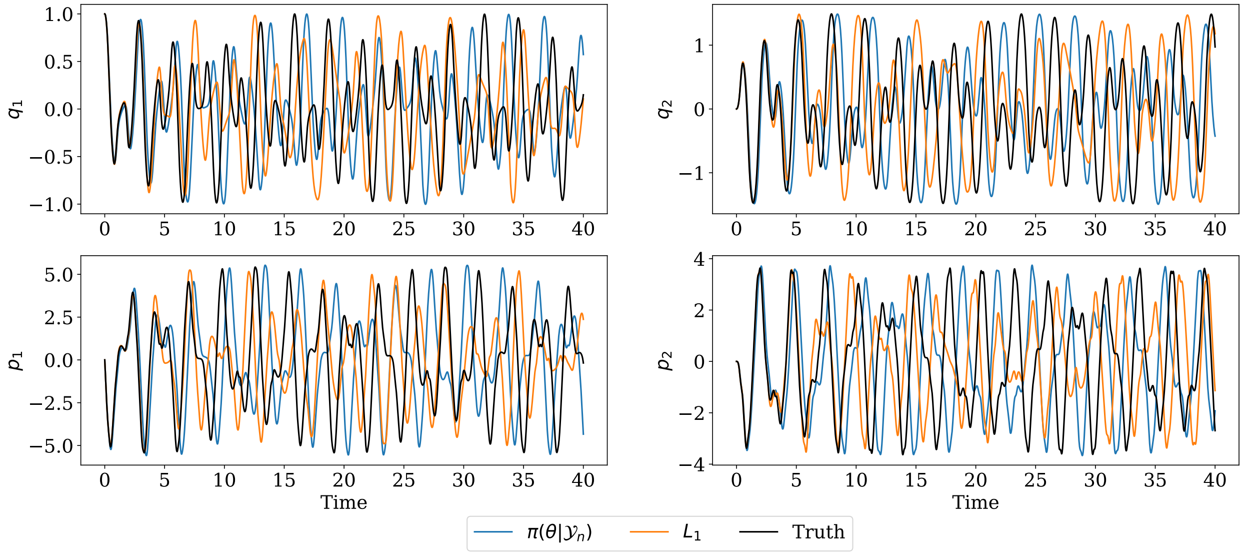


Figure 5.12: Comparison of the MAP and L_1 estimates on the double pendulum.

Next, we look at the accuracy of the two estimates in phase space. Specifically, we quantify this accuracy by evaluating the absolute Hamiltonian error defined as

$$\frac{1}{2n}\sum_{k=0}^{2n-1}\left|\mathcal{H}(\hat{\mathbf{q}_k},\hat{\mathbf{p}_k})-\mathcal{H}(\mathbf{q}_0,\mathbf{p}_0)\right|. \tag{5.29}$$

Fig. 5.14a shows the estimates in phase space. The color of the line denotes the value of the absolute Hamiltonian error at that point. The absolute Hamiltonian error is also plotted over time for reference in Fig. 5.14b. We see that both qualitatively and quantitatively, the MAP estimate is much closer to the true Hamiltonian. Moreover, the absolute Hamiltonian error of the MAP estimate is lower on average and does not display sudden spikes in error like the L_1 estimate. This is possibly due to the inherent regularization in the marginal likelihood that penalizes large output covariance. Overall on this example, we see that the negative log posterior is better able to capture the underlying Hamiltonian manifold of the

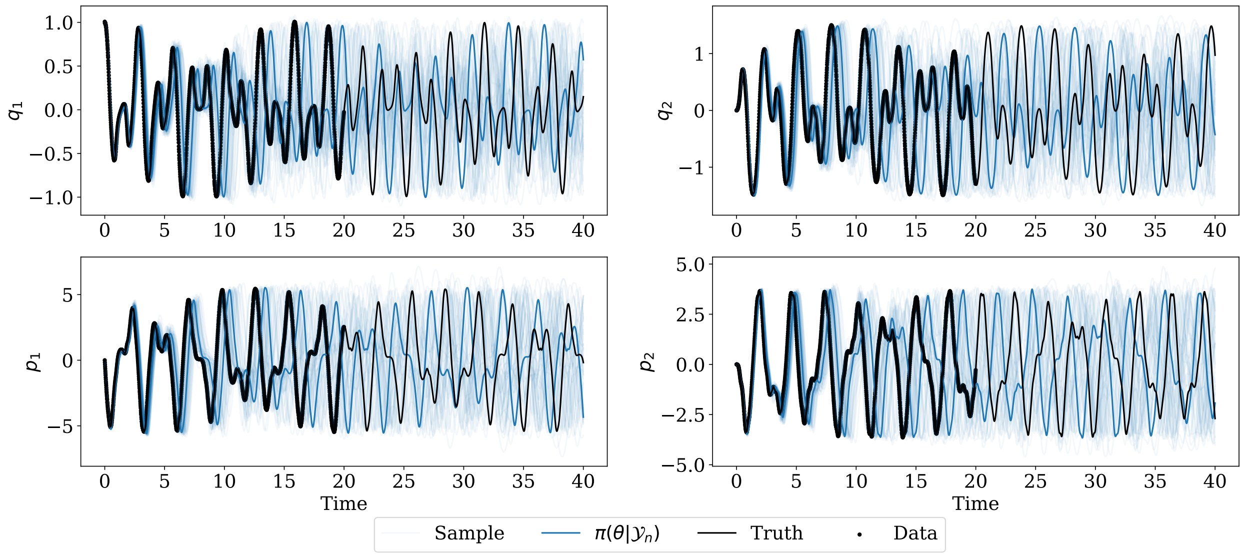


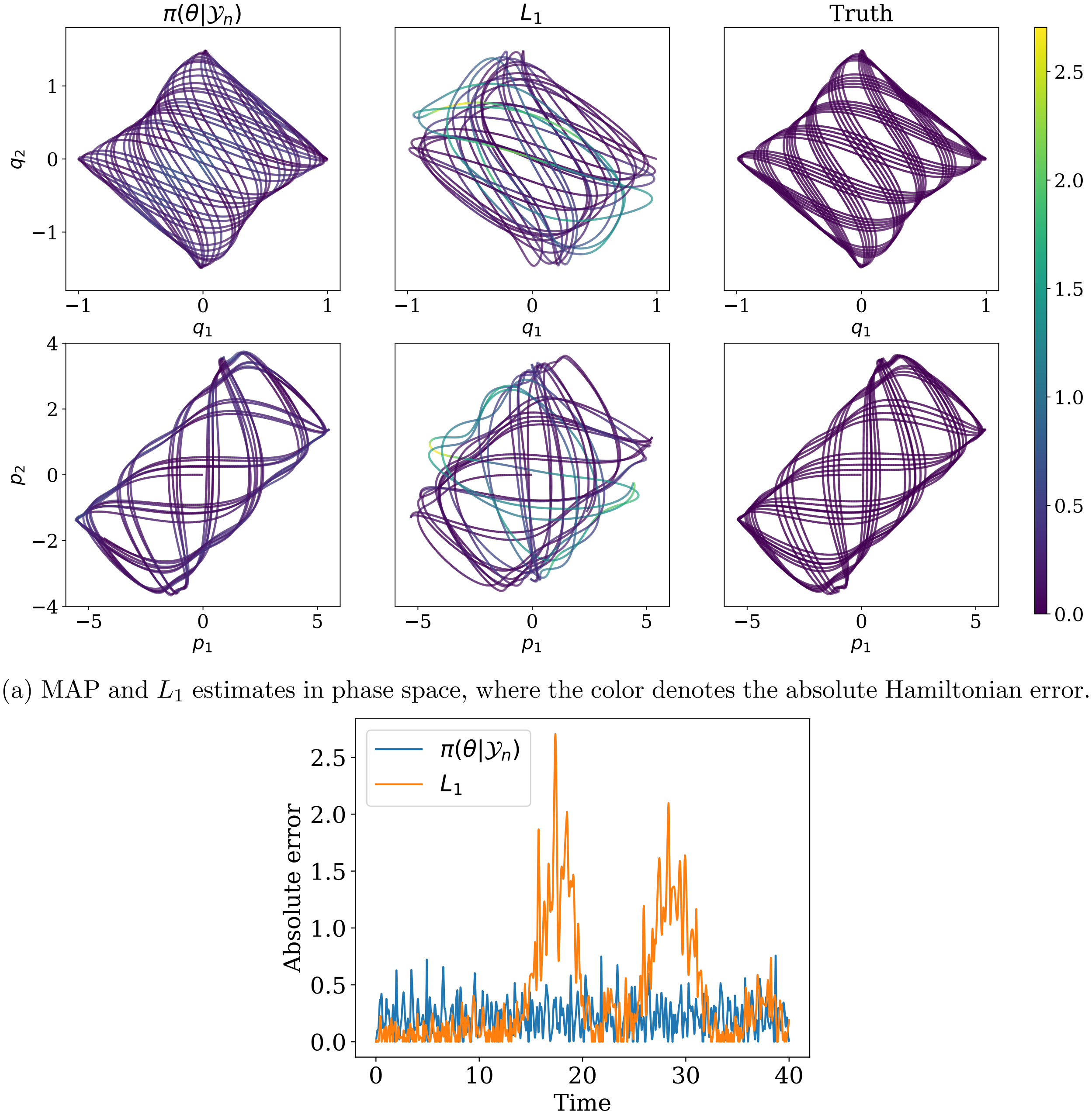
Figure 5.13: Realizations of the MAP dynamics with the nominal MAP estimate, truth, and data overlaid.

chaotic double pendulum.

5.5 Summary

In this chapter, we considered learning Hamiltonian systems using not only data, but also prior knowledge from physics. We began by introducing background on Hamiltonians that explained that such systems conserve a function of the generalized position and momentum, known as the Hamiltonian, from which the time derivatives can be derived. As a result, we are able to restrict the model space to Hamiltonian systems by parameterizing the Hamiltonian directly. Then we presented two numerical integration schemes that preserve symplecticity – one for separable and one for nonseparable Hamiltonians. We discussed how many learning frameworks for Hamiltonians will use non-symplectic integrators during training and then switch to a symplectic integrator during testing. Then, we proposed embedding the marginal likelihood with a symplectic integrator and compared this approach to the marginal likelihood without a symplectic integrator. This comparison differed from previous comparisons in that we ensured both integrators had the same order of accuracy to isolate the benefits of preserving symplecticity. This comparison showed that by enforcing symplecticity during training, the resulting estimates were more accurate and precise compared to estimates that were found using a non-symplectic integrator. The improved precision manifested itself in lower process noise values, demonstrating how the process noise can be used to quantify the uncertainty in an estimated model without requiring simulation.

Then, we turned to demonstrating the robustness of the marginal likelihood for noise forms other than additive Gaussian noise. We showed that even when the actual noise is not additive Gaussian, the additive Gaussian noise model can still deliver acceptable estimates, even outperforming a LS method. After this comparison, we implemented the filter for multiplicative noise described in Section 2.3.1.2. We compared the marginal likelihood with this filter to the L_1 objective for learning NSSNNs on data with uniform, multiplicative noise. This comparison showed that the marginal likelihood still demonstrates strong noise robustness with this other filter compared to the L_1 objective. Lastly, we applied the marginal likelihood to learning the chaotic double pendulum. This example showed that although long-term prediction on chaotic systems is extremely difficult, the marginal likelihood was able to identify a model whose solution remained close to the true manifold of the system.



(a) MAP and L_1 estimates in phase space, where the color denotes the absolute Hamiltonian error.

(b) Absolute Hamiltonian error over time.

Figure 5.14: Absolute Hamiltonian error in phase and time domains.

CHAPTER VI
Conclusions and Future Work

In this dissertation, we have proposed a framework with the potential for unifying many system ID objective functions. This framework considers a HMM that not only models the output error, but also models the dynamics error in the form of process noise. From this formulation, we derived a marginal likelihood that, when paired with a prior distribution, can be shown to contain many other popular objectives. Specifically, we proved that the following four objective functions can be viewed as special cases of the negative log posterior that uses this proposed marginal likelihood: DMD (Theorem 3), single rollout Markov parameter estimation (Proposition 2), SINDy (Theorem 4), and MS (Proposition 4).

In addition to increased generality, the process noise term was shown to also provide three main benefits. The first is adding regularization into the marginal likelihood directly from the model formulation (Eq. (3.2)). Since this regularization is placed over the output space of the system, it is more interpretable than heuristic parameter priors, and it is applicable to arbitrary model parameterizations. The second benefit is that it can smooth the optimization surface to improve ease of optimization (Fig. 1.3). It was shown that this smoothing is similar to that touted by the MS objective (Section 4.1.2.2). The process noise covariance, however, has the added advantages of being continuous and multi-dimensional for greater flexibility and being able to be tuned automatically. Lastly, we showed that the estimated process noise term can be used as a proxy for estimating the quantity of uncertainty in a model estimate (Section 5.4.1). This is useful when one wants to avoid additional simulations of the model or to avoid the cost of MCMC sampling (Section 5.3.1.2).

We also tested the marginal likelihood on many numerical experiments. We showed that the marginal likelihood is robust to different forms of noise (Section 5.4.2), even forms that do not match the modeling assumptions (Sections 5.4.3 and 5.3.1.2). Additionally, we provided a novel analysis using process noise covariance and MCMC sampling to show how physics-informed estimation can improve the accuracy and precision of estimates (Section 5.4.1).

Despite all of these advantages, the method still has a number of limitations that can be addressed through future work. For example, the Bayesian algorithm is significantly more expensive than other system ID approaches, making it infeasible for computationally intensive problems. Therefore, lowering its computation time could facilitate its application to larger, more complicated problems and could even enable online inference. The computational cost of the algorithm can be prohibitively expensive in two main scenarios: (1) the system state or the measurements are high-dimensional, and (2) the model parameters are unidentifiable, resulting in complex relationships between parameters that significantly extend the convergence time of MCMC sampling.

This first scenario commonly arises in the study of spatio-temporal systems such as fluid flow and time-dependent materials phenomena. These systems are infinite-dimensional and discretizations of the spatial domain often use thousands of nodes. For many LS objectives, the dominant cost of the objective computation for these systems is evaluation of the forward model. In comparison, the marginal likelihood computation not only needs to evaluate the forward model, but it must also compute time-varying covariance matrices, which incurs additional cost. As was shown in Section 2.3.2, this additional cost scales cubically with the dimensions of the state and measurements. Therefore, the computational cost of marginal likelihood evaluation can become prohibitive as state or output dimension increases.

One approach to lowering the cost of marginal likelihood evaluation for high-dimensional systems is to reduce the dimension of the model. Often, the dynamics of a spatio-temporal system can be modeled on a low-dimensional manifold while retaining relatively high representation accuracy. Common methods for dimension reduction include principal component analysis (PCA), kernel PCA, and diffusion maps. By replacing the high-dimensional data in the marginal likelihood with low-dimensional data, the negative log marginal likelihood becomes a feasible option for an objective function. It is important to note, however, that the reduced data do not represent the system behavior as accurately as the full-dimensional data, so the mapping to the low-dimensional space will add greater error/uncertainty into the system ID problem. Proper quantification of this uncertainty will significantly impact the accuracy of the learned model and should be a central area of focus within this research direction.

The second scenario pertains to the cost of MCMC sampling. Even when the cost of marginal likelihood evaluation is reasonable, the computational resources required for quantifying parameter uncertainty through MCMC sampling can be exorbitant in some cases. Specifically, when the parameters are correlated through complex relationships, it can be very difficult to efficiently sample from the posterior distribution. These complicated relations often arise due to non-identifiability in the parameters, but for black-box modeling, the parameter values are not of interest. Instead, the uncertainty in the model outputs is the target of sampling. As a consequence of the non-uniqueness of parameter values, the parameter posterior can theoretically be greatly simplified without any loss in the representation accuracy of the output posterior. Thus, approximate inference approaches for the parameter posterior have the potential to drastically reduce computational requirements for uncertainty quantification with minimal loss in the accuracy of the output posterior. One example of such an approach is variational inference, which is commonly used for efficient uncertainty quantification, usually in return for lower representational accuracy. Successful approximation of the posterior through variational inference can allow for fast, independent sampling and drastically reduce the expense of the most costly aspect of the system ID algorithm.

A final possible direction for future work is developing methods for tuning the process and measurement noise covariance matrices. These hyperparameters determine the topology of the marginal likelihood surface and are therefore critically important for efficient optimization and sampling. In fact, the success of the optimizer is often sensitive to the initial values of and priors placed on the covariance parameters. Achieving successful optimization usually requires significant hand-tuning of these parameters by the user, which can take a substantial amount of time given the computational requirements of the algorithm discussed earlier. An algorithm that could automatically tune these parameters for the user would be beneficial for the sake of time and ease of application. There are many objectives in the literature that balance fit and regularization with a tuning parameter, and the approaches used in these works could be potentially applicable to the marginal likelihood. One example of such a technique is the homotopy analysis method. Alternatively, there are also adaptive methods in the filtering literature that seek to properly tune the process and measurement noise covariances.

In conclusion, the algorithm presented in this dissertation was shown to be a powerful method for system ID problems that possess substantial amounts of uncertainty, but issues of computational expense and tuning difficulties can significantly impede its widespread application. Addressing the method's limitations through the research directions discussed in this chapter can augment the algorithm's power and flexibility, leading to the subsumption of ever more challenging problems into its domain of applicability.

APPENDIX A
Pseudocode

In this appendix we provide the pseudocode for both the linear Kalman filter and nonlinear unscented Kalman filter algorithms. In the UKF algorithm, α and κ are parameters that determine the spread of the sigma points around the mean, β is a parameter used for incorporating prior information on the distribution of $\mathbf{x}$, and the notation $[\cdot]_i$ denotes the i-th row of the matrix [87].

Algorithm 4 Kalman filtering for evaluating $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$ (exact for linear models)

```
Require: System parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_\Psi, \boldsymbol{\theta}_h, \boldsymbol{\theta}_\Sigma, \boldsymbol{\theta}_\Gamma)$;
         Prior distribution $\pi(\boldsymbol{\theta})$;
         Distribution on initial condition $\mathbf{m}_0$, $\mathbf{P}_0$;
         Linear dynamical model parameterization $\mathbf{A}(\boldsymbol{\theta}_\Psi)$;
         Linear observation model parameterization $\mathbf{H}(\boldsymbol{\theta}_h)$;
         Covariance matrices $\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)$ and $\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)$
Ensure: Posterior evaluation $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$
Compute the prior $\pi(\boldsymbol{\theta}|\mathcal{Y}_0) = \pi(\boldsymbol{\theta})$
for $k = 1$ to $n$ do
    Predict $\pi(\mathbf{x}_k|\boldsymbol{\theta}, \mathbf{y}_{k-1}) = \mathcal{N}(\mathbf{m}_k^-, \mathbf{P}_k^-)$
        $\mathbf{m}_k^-(\boldsymbol{\theta}) = \mathbf{A}(\boldsymbol{\theta}_\Psi)\mathbf{m}_{k-1}$
        $\mathbf{P}_k^-(\boldsymbol{\theta}) = \mathbf{A}(\boldsymbol{\theta}_\Psi)\mathbf{P}_{k-1}\mathbf{A}^\top(\boldsymbol{\theta}_\Psi) + \boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)$
    Compute the Evidence $\pi(\mathbf{y}_k|\boldsymbol{\theta}, \mathcal{Y}_{k-1}) = \mathcal{N}(\boldsymbol{\mu}_k, \mathbf{S}_k)$
        $\boldsymbol{\mu}_k(\boldsymbol{\theta}) = \mathbf{H}(\boldsymbol{\theta}_h)\mathbf{m}_k^-$
        $\mathbf{S}_k(\boldsymbol{\theta}) = \mathbf{H}(\boldsymbol{\theta}_h)\mathbf{P}_k^-\mathbf{H}^\top(\boldsymbol{\theta}_h) + \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)$
    Update $\pi(\mathbf{x}_k|\boldsymbol{\theta}, \mathcal{Y}_k) = \mathcal{N}(\mathbf{m}_k, \mathbf{P}_k)$
        $\mathbf{m}_k(\boldsymbol{\theta}) = \mathbf{m}_k^- + \mathbf{P}_k^-\mathbf{H}^\top(\boldsymbol{\theta}_h)\mathbf{S}_k^{-1}(\mathbf{y}_k - \boldsymbol{\mu}_k)$
        $\mathbf{P}_k(\boldsymbol{\theta}) = \mathbf{P}_k^- - \mathbf{P}_k^-\mathbf{H}^\top(\boldsymbol{\theta}_h)\mathbf{S}_k^{-1}\mathbf{H}(\boldsymbol{\theta}_h)\mathbf{P}_k^-$
    Update $\pi(\boldsymbol{\theta}|\mathcal{Y}_k) \propto \pi(\mathbf{y}_k|\boldsymbol{\theta}, \mathcal{Y}_{k-1})\pi(\boldsymbol{\theta}|\mathcal{Y}_{k-1})$
end for
```

Algorithm 5 Unscented Kalman filtering algorithm for approximating $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$

```
Require: System parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_\Psi, \boldsymbol{\theta}_h, \boldsymbol{\theta}_\Sigma, \boldsymbol{\theta}_\Gamma)$;
         Prior distribution $\pi(\boldsymbol{\theta})$;
         Distribution on initial condition $\mathbf{m}_0$, $\mathbf{P}_0$;
         Dynamical model parametrization $\Psi(\boldsymbol{\theta}_\Psi)$;
         Observation model parameterization $h(\boldsymbol{\theta}_h)$;
         Covariance matrices $\boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)$ and $\boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)$;
         UKF parameters $\alpha$, $\kappa$, $\beta$
Ensure: Approximate evaluation of the posterior $\pi(\boldsymbol{\theta}|\mathcal{Y}_n)$
Calculate $\lambda = \alpha^2(d_x + \kappa) - d_x$
Compute the weights
   $W_0^{(m)} = \frac{\lambda}{d_x+\lambda}$
   $W_0^{(c)} = \frac{\lambda}{d_x+\lambda} + (1 - \alpha^2 + \beta)$
   $W_i^{(m)} = W_i^{(c)} = \frac{1}{2(d_x+\lambda)}, \quad \forall i = 1, \ldots, 2d_x$
Compute the prior $\pi(\boldsymbol{\theta} \mid \mathcal{Y}_0) = \pi(\boldsymbol{\theta})$
for $k = 1$ to $n$ do
    Predict $\pi(\mathbf{x}_k|\boldsymbol{\theta}, \mathcal{Y}_{k-1}) \approx \mathcal{N}(\mathbf{m}_k^-, \mathbf{P}_k^-)$
    Form the sigma points
       $\mathcal{X}_{k-1}^{(0)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1}$
       $\mathcal{X}_{k-1}^{(i)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1} + \sqrt{d_x + \lambda}\left[\sqrt{\mathbf{P}_{k-1}}\right]_i$
       $\mathcal{X}_{k-1}^{(i+d_x)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1} - \sqrt{d_x + \lambda}\left[\sqrt{\mathbf{P}_{k-1}}\right]_i, \quad \forall i = 1, \ldots, d_x$
    Propagate the sigma points through the dynamical model
       $\hat{\mathcal{X}}_k^{(i)}(\boldsymbol{\theta}) = \Psi(\mathcal{X}_k^{(i)}, \boldsymbol{\theta}_\Psi), \quad \forall i = 0, \ldots, 2d_x$
    Compute the mean and covariance
       $\mathbf{m}_k^-(\boldsymbol{\theta}) = \sum_{i=0}^{2d_x} W_i^{(m)} \hat{\mathcal{X}}_k^{(i)}$
       $\mathbf{P}_k^-(\boldsymbol{\theta}) = \sum_{i=0}^{2d_x} W_i^{(c)} (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)(\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^\top + \boldsymbol{\Sigma}(\boldsymbol{\theta}_\Sigma)$
    Compute the Evidence $\pi(\mathbf{y}_k|\boldsymbol{\theta}, \mathcal{Y}_{k-1}) \approx \mathcal{N}(\boldsymbol{\mu}_k, \mathbf{S}_k)$
    Update the sigma points
       $\mathcal{X}_{k-1}^{(0)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1}$
       $\mathcal{X}_{k-1}^{(i)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1} + \sqrt{d_x + \lambda}\left[\sqrt{\mathbf{P}_{k-1}}\right]_i$
       $\mathcal{X}_{k-1}^{(i+d_x)}(\boldsymbol{\theta}) = \mathbf{m}_{k-1} - \sqrt{d_x + \lambda}\left[\sqrt{\mathbf{P}_{k-1}}\right]_i, \quad \forall i = 1, \ldots, d_x$
    Propagate the sigma points through the observation model
       $\hat{\mathcal{Y}}_k^{(i)}(\boldsymbol{\theta}) = h(\mathcal{X}_k^{(i)}, \boldsymbol{\theta}_h), \quad \forall i = 0, \ldots, 2d_x$
    Compute the mean and covariance
       $\boldsymbol{\mu}_k(\boldsymbol{\theta}) = \sum_{i=0}^{2d_x} W_i^{(m)} \hat{\mathcal{Y}}_k^{(i)}$
       $\mathbf{S}_k(\boldsymbol{\theta}) = \sum_{i=0}^{2d_x} W_i^{(c)} (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k^-)(\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k^-)^T + \boldsymbol{\Gamma}(\boldsymbol{\theta}_\Gamma)$
    Update $\pi(\mathbf{x}_k|\boldsymbol{\theta}, \mathbf{y}_k) \approx \mathcal{N}(\mathbf{m}_k, \mathbf{P}_k)$
       $\mathbf{U}_k(\boldsymbol{\theta}) = \sum_{i=0}^{2d_x} W_i^{(c)} (\mathcal{X}_k^{(i)} - m_k^-)(\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^\top$
       $\mathbf{m}_k(\boldsymbol{\theta}) = \mathbf{m}_k^- + (\mathbf{U}_k \mathbf{S}_k^{-1})(\mathbf{y}_k - \boldsymbol{\mu}_k)$
       $\mathbf{P}_k(\boldsymbol{\theta}) = \mathbf{P}_k^- - (\mathbf{U}_k \mathbf{S}_k^{-1})\mathbf{S}_k^{-1}(\mathbf{U}_k \mathbf{S}_k^{-1})^\top$
    Update $\pi(\boldsymbol{\theta}|\mathcal{Y}_k) \propto \pi(\mathbf{y}_k|\boldsymbol{\theta}, \mathcal{Y}_{k-1})\pi(\boldsymbol{\theta}|\mathcal{Y}_{k-1})$
end for
```

APPENDIX B
Eigensystem Realization Algorithm

The ERA is a subspace identification algorithm that is commonly paired with Markov parameter estimation methods, such as those described in Section 3.2.2, to procure an estimated realization of the state-space matrices. The implementation of this algorithm is detailed here.

Assume that a subset of the Markov parameters $\{\mathbf{G}_i\}_{i=0}^{n}$ are available and the state dimension d_x is known. The first step of the ERA is to form these $n+1$ Markov parameters into a Hankel matrix as shown

$$\mathbf{E} = \begin{bmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_{d_2} \\ \mathbf{G}_1 & \mathbf{G}_2 & \cdots & \mathbf{G}_{d_2+1} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{G}_{d_1-1} & \mathbf{G}_{d_1} & \cdots & \mathbf{G}_{d_1+d_2} \end{bmatrix}, \tag{2.1}$$

where $d_1, d_2 \in \mathbb{N}$ determine the shape of the Hankel matrix and must satisfy the inequalities $d_1 + d_2 \leq n$ and $\min\{d_y d_1, d_u d_2\} \geq d_x$. Choosing balanced dimensions $d_y d_1 \approx d_u d_2$ can possibly improve noise robustness.

The ERA uses this Hankel matrix to construct an estimate of the system's observability and controllability matrices from which a realization of the state-space matrices can be extracted. Define the observability and controllability matrices

$$\mathbf{O} = \begin{bmatrix} \mathbf{H}^\top & (\mathbf{HA})^\top & \cdots & (\mathbf{HA}^{d_2})^\top \end{bmatrix}^\top, \quad \mathbf{C} = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \cdots & \mathbf{A}^{d_1-1}\mathbf{B} \end{bmatrix},$$

respectively. Define the Hankel submatrices $\mathbf{E}^- = \mathbf{E}[:, 1 : d_2 d_u]$ and $\mathbf{E}^+ = \mathbf{E}[:, d_u + 1 : (d_2 + 1)d_u]$. Taking the SVD of the Hankel submatrix $\mathbf{E}^-$ yields the decomposition $\mathbf{E}^- = \mathbf{U}\mathbf{S}\mathbf{V}^\top$. The rank-$d_x$ approximation of $\mathbf{E}^-$ can be decomposed as $\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^\top$, where $\tilde{\mathbf{U}} = \mathbf{U}[:, 1 : d_x]$, $\tilde{\mathbf{S}} = \mathbf{S}[1 : d_x, 1 : d_x]$, and $\tilde{\mathbf{V}} = \mathbf{V}[:, 1 : d_x]$. Next, $\mathbf{O}$ and $\mathbf{C}$ can be estimated as $\hat{\mathbf{O}} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}^{\frac{1}{2}}$ and $\hat{\mathbf{C}} = \tilde{\mathbf{S}}^{\frac{1}{2}}\tilde{\mathbf{V}}^\top$. The ERA estimate of the state-space matrices are then $\hat{\mathbf{B}} = \hat{\mathbf{C}}[:, 1 : d_u]$,

$\hat{\mathbf{H}} = \hat{\mathbf{O}}[1 : d_y, :]$, and $\hat{\mathbf{A}} = \hat{\mathbf{O}}^\dagger \hat{\mathbf{E}}^+ \hat{\mathbf{C}}^\dagger$.

Algorithm 6 Least squares + eigensystem realization algorithm (LS+ERA) [71]

```
Require: Observations $\mathbf{Y}$
         Inputs $\mathbf{U}$
         State-space dimension $d_x$
         Hankel shape parameters $d_1$,$d_2$
Ensure: $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, $\hat{\mathbf{H}}$
Estimate Markov parameters: $\hat{\mathbf{G}} = (\mathbf{U}^\dagger \mathbf{Y})^\top$
$\hat{\mathbf{E}}^- = \hat{\mathbf{G}}[:, 1 : d_2 d_u]$ and $\hat{\mathbf{E}}^+ = \hat{\mathbf{G}}[:, (d_u + 1) : (d_2 + 1)d_u]$
$\mathbf{U}, \mathbf{S}, \mathbf{V} = \mathrm{SVD}(\hat{\mathbf{E}}^-)$
$\tilde{\mathbf{U}} = \mathbf{U}[:, 1 : d_x]$, $\tilde{\mathbf{S}} = \mathbf{S}[1 : d_x, 1 : d_x]$, and $\tilde{\mathbf{V}} = \mathbf{V}[:, 1 : d_x]$
$\hat{\mathbf{O}} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}^{\frac{1}{2}}$ and $\hat{\mathbf{C}} = \tilde{\mathbf{S}}^{\frac{1}{2}}\tilde{\mathbf{V}}^\top$
$\hat{\mathbf{B}} = \hat{\mathbf{C}}[:, 1 : d_u]$, $\hat{\mathbf{H}} = \hat{\mathbf{O}}[1 : d_y, :]$, and $\hat{\mathbf{A}} = \hat{\mathbf{O}}^\dagger \hat{\mathbf{E}}^+ \hat{\mathbf{C}}^\dagger$
```

APPENDIX C
Proof of Proposition 3

Our goal is to show that the assumptions introduced in Proposition 2 that lead to equivalency hold asymptotically. That is, we want to show $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}\mathbf{G}_i\mathbf{u}_{k-i} = \mathbf{0}$ and $\lim_{\bar{n}\to\infty}\mathbf{A}^k\boldsymbol{\Sigma}(\mathbf{A}^k)^\top = \mathbf{0}$ for $k \geq \bar{n}$. Let $N = k - \bar{n} + 1$ be the number of terms in the sum. We assume N is bounded such that it cannot grow arbitrarily large.

To show $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}\mathbf{G}_i\mathbf{u}_{k-i} = \mathbf{0}$, it suffices to show $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}|\mathbf{G}_i[j,:]\mathbf{u}_{k-i}| = 0$ for any $j \in \{1,\dots,d_y\}$. By assumption, the inputs $\mathbf{u}_i[j] \in \mathbb{R}$ are independent realizations of the random variable $\mathbf{u}$ for $i = 0, 1, \dots$ and $j = 1, \dots, d_u$. Recall that for any real-valued random variable z, $\lim_{a\to\infty}\int_{-a}^{a}\pi(z)\mathrm{d}z = 1$, where the integral represents the probability that $|z| < a$. This implies that for any $0 < \varepsilon < 1$, $\exists \mathbf{a} \in \mathbb{R}^{d_u}$ such that $|\mathbf{u}[j]| < \mathbf{a}[j]$ for $j = 1, \dots, d_u$ with probability (w.p.) $1 - \varepsilon$. Then, upper and lower bounds can be established as

$$0 \leq \sum_{i=\bar{n}}^{k}|\mathbf{G}_i[j,:]\mathbf{u}_{k-i}| < \sum_{i=\bar{n}}^{k}|\mathbf{G}_i[j,:]|\,\mathbf{a}, \tag{3.1}$$

w.p. at least $(1-\varepsilon)^N$. Next we prove that the upper bound goes to zero as $\bar{n} \to \infty$ regardless of $\mathbf{a}$ by showing that $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}\mathbf{G}_i[j,:] = \mathbf{0}$. Recall that when $\rho(\mathbf{A}) < 1$, the LTI system is exponentially stable. Specifically, there exist constants $c > 0$ and $\lambda \in (0,1)$ such that $\|\mathbf{A}^k\mathbf{x}_0\|_2 \leq c\lambda^k\|\mathbf{x}_0\|_2$, $\forall \mathbf{x}_0 \in \mathbb{R}^{d_x}$. We can therefore bound the norm of the columns of $\mathbf{G}_i$ as follows:

$$\begin{aligned}
\|\mathbf{G}_i[:,j]\|_2 &= \|\mathbf{H}\mathbf{A}^{i-1}\mathbf{B}[:,j]\|_2 && (3.2a)\\
&\leq \|\mathbf{H}\|_2\|\mathbf{A}^{i-1}\mathbf{B}[:,j]\|_2 && (3.2b)\\
&\leq \|\mathbf{H}\|_2 c\lambda^{i-1}\|\mathbf{B}[:,j]\|_2. && (3.2c)
\end{aligned}$$

Noting that the quantity $c\|\mathbf{H}\|_2\|\mathbf{B}[:,j]\|_2$ is constant, we see that $\|\mathbf{G}_i\|_2$ is bounded above by an exponentially decaying function of timestep i. This leads us to the following bound on the norm of the sum:

$$\left\|\sum_{i=\bar{n}}^{k}\mathbf{G}_i[:,j]\right\|_2 \leq \sum_{i=\bar{n}}^{k}\|\mathbf{G}_i[:,j]\|_2 \tag{3.3a}$$

$$\leq \sum_{i=\bar{n}}^{k} c\lambda^{i-1}\|\mathbf{H}\|_2\|\mathbf{B}[:,j]\|_2 \tag{3.3b}$$

$$\leq Nc\lambda^{\bar{n}-1}\|\mathbf{H}\|_2\|\mathbf{B}[:,j]\|_2. \tag{3.3c}$$

Since $0 < \lambda < 1$, $\lim_{\bar{n}\to\infty}\|\sum_{i=\bar{n}}^{k}\mathbf{G}_i[:,j]\|_2 = 0$, and consequently $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}\mathbf{G}_i[:,j] = \mathbf{0}$ as well, each w.p. at least $(1-\varepsilon)^N$. The variable ε is arbitrary, so $\lim_{\bar{n}\to\infty}\sum_{i=\bar{n}}^{k}\mathbf{G}_i\mathbf{u}_{k-i} \to \mathbf{0}$ w.p. 1. Moreover, the rate of convergence of the upper bound can be found as follows:

$$\lim_{\bar{n}\to\infty}\frac{Nc\lambda^{\bar{n}}\|\mathbf{H}\|_2\|\mathbf{B}[:,j]\|_2}{Nc\lambda^{\bar{n}-1}\|\mathbf{H}\|_2\|\mathbf{B}[:,j]\|_2} = \lim_{\bar{n}\to\infty}\frac{\lambda^{\bar{n}}}{\lambda^{\bar{n}-1}} = \lambda. \tag{3.4}$$

Therefore, the upper bound converges to zero linearly with rate λ, and the sum $\sum_{i=\bar{n}}^{k}\mathbf{G}_i\mathbf{u}_{k-i}$ must converge at least as fast. When $\mathbf{A}$ is diagonalizable, λ can be chosen to be $\rho(\mathbf{A})$ such that the convergence rate is bounded above by the maximum eigenvalue of $\mathbf{A}$.

The proof for the covariance term is similar. To show $\lim_{\bar{n}\to\infty}\mathbf{A}^{\bar{n}}\mathbf{\Sigma}(\mathbf{A}^{\bar{n}})^\top = \mathbf{0}$, we begin by decomposing the covariance matrix $\mathbf{\Sigma} = \mathbf{\Sigma}^{\frac{1}{2}}(\mathbf{\Sigma}^{\frac{1}{2}})^\top$. Then, the norm $\mathbf{A}^{\bar{n}}\mathbf{\Sigma}(\mathbf{A}^{\bar{n}})^\top$ is bounded above as follows:

$$\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}(\mathbf{A}^{\bar{n}})^\top\|_2 \leq \left\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}\right\|_2\left\|\left(\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}\right)^\top\right\|_2 = \left\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}\right\|_2^2. \tag{3.5}$$

It suffices to show $\lim_{\bar{n}\to\infty}\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}[:,j] = \mathbf{0}$ for columns $j \in \{1,\ldots,d_x\}$. To begin, we derive an upper bound:

$$\left\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}[:,j]\right\|_2^2 \leq \left(c\lambda^{\bar{n}}\left\|\mathbf{\Sigma}^{\frac{1}{2}}[:,j]\right\|_2\right)^2. \tag{3.6}$$

The fact that $\lim_{\bar{n}\to\infty}c^2\lambda^{2\bar{n}}\|\mathbf{\Sigma}^{\frac{1}{2}}[:,j]\|_2^2 = 0$ implies that $\lim_{\bar{n}\to\infty}\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}[:,j] = \mathbf{0}$. Lastly, the rate of convergence of the upper bound is as follows:

$$\lim_{\bar{n}\to\infty}\frac{c^2\lambda^{2\bar{n}}\left\|\mathbf{\Sigma}^{\frac{1}{2}}[:,j]\right\|_2^2}{c^2\lambda^{2(\bar{n}-1)}\left\|\mathbf{\Sigma}^{\frac{1}{2}}[:,j]\right\|_2^2} = \lim_{\bar{n}\to\infty}\frac{\lambda^{2\bar{n}}}{\lambda^{2\bar{n}-2}} = \lambda^2. \tag{3.7}$$

Then $\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}^{\frac{1}{2}}\|_2^2$, and consequently $\|\mathbf{A}^{\bar{n}}\mathbf{\Sigma}(\mathbf{A}^{\bar{n}})^\top\|_2$, converges at least as fast. Therefore, we have shown that $\sum_{i=\bar{n}}^{k}\mathbf{G}_i\mathbf{u}_{k-i}$ and $\mathbf{A}^{\bar{n}}\mathbf{\Sigma}(\mathbf{A}^{\bar{n}})^\top$ both converge to zero as $\bar{n}\to\infty$ with rate no greater than λ.

Bibliography

[1] Luis A Aguirre, Bruno HG Barbosa, and Antônio P Braga. Prediction and simulation errors in parameter estimation for nonlinear systems. *Mechanical Systems and Signal Processing*, 24(8):2855–2867, 2010.

[2] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[3] Vladimir Igorevich Arnol'd. *Mathematical methods of classical mechanics*, volume 60. Springer Science & Business Media, 2013.

[4] Richard Askey and James Arthur Wilson. *Some basic hypergeometric orthogonal polynomials that generalize Jacobi polynomials*, volume 319. American Mathematical Soc., 1985.

[5] Shivam Bang, Rajat Bishnoi, Ankit Singh Chauhan, Akshay Kumar Dixit, and Indu Chawla. Fuzzy logic based crop yield prediction using temperature and rainfall parameters predicted through arma, sarima, and armax models. In *2019 Twelfth International Conference on Contemporary Computing (IC3)*, pages 1–6. IEEE, 2019.

[6] Gerben Beintema, Roland Toth, and Maarten Schoukens. Nonlinear state-space identification using deep encoder networks. In *Learning for dynamics and control*, pages 241–250. PMLR, 2021.

[7] James O Berger. *Statistical decision theory and Bayesian analysis.* Springer Science & Business Media, 2013.

[8] Tom Bertalan, Felix Dietrich, Igor Mezić, and Ioannis G Kevrekidis. On learning Hamiltonian systems from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(12):121107, 2019.

[9] Hans Georg Bock and Karl-Josef Plitt. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984.

[10] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[11] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1), 2017.

[12] ARR Casti, PJ Morrison, and EA Spiegel. Negative energy modes and gravitational instability of interpenetrating fluids. *Annals of the New York Academy of Sciences*, 867(1):93–108, 1998.

[13] Julio E Castrillon-Candas, Fabio Nobile, and Raul F Tempone. Analytic regularity and collocation approximation for elliptic pdes with random domain deformations. *Computers & Mathematics with Applications*, 71(6):1173–1197, 2016.

[14] Rick Chartrand. Numerical differentiation of noisy, nonsmooth data. *International Scholarly Research Notices*, 2011, 2011.

[15] Lennart Ljung Tianshi Chen. What can regularization offer for estimation of dynamical systems? *IFAC Proceedings Volumes*, 46(11):1–8, 2013.

[16] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[17] Thomas Macfarland Cherry. V. On periodic solutions of Hamiltonian systems differential equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 227(647-658):137–221, 1928.

[18] Mohammad Amin Chitsazan, M Sami Fadali, and Andrzej M Trzynadlowski. Wind speed and wind direction forecasting using echo state network with nonlinear functions. *Renewable energy*, 131:879–889, 2019.

[19] James Colliander, Markus Keel, Gigiola Staffilani, Hideo Takaoka, and Terence Tao. Transfer of energy to high frequencies in the cubic defocusing nonlinear Schrödinger equation. *Inventiones Mathematicae*, 181(1):39–113, 2010.

[20] Rongxin Cui, Chenguang Yang, Yang Li, and Sanjay Sharma. Adaptive neural network control of auvs with control input nonlinearities using reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(6):1019–1029, 2017.

[21] Perry De Valpine and Alan Hastings. Fitting population models incorporating process noise and observation error. *Ecological Monographs*, 72(1):57–76, 2002.

[22] Socrates Dokos and Nigel H Lovell. Parameter estimation in cardiac ionic models. *Progress in biophysics and molecular biology*, 85(2-3):407–431, 2004.

[23] Sergey Dolgov, Dante Kalise, and Karl K Kunisch. Tensor decomposition methods for high-dimensional hamilton–jacobi–bellman equations. *SIAM Journal on Scientific Computing*, 43(3):A1625–A1650, 2021.

[24] Christopher Drovandi, Richard G Everitt, Andrew Golightly, and Dennis Prangle. Ensemble MCMC: accelerating pseudo-marginal MCMC for state space models using the ensemble Kalman filter. *Bayesian Analysis*, 17(1):223–260, 2022.

[25] Robert J Elliott, Lakhdar Aggoun, and John B Moore. *Hidden Markov models: estimation and control*, volume 29. Springer Science & Business Media, 2008.

[26] Etienne Forest. Geometric integration for particle accelerators. *Journal of Physics A: Mathematical and General*, 39(19):5321, 2006.

[27] Nicholas Galioto and Alex A Gorodetsky. Bayesian identification of hamiltonian dynamics from symplectic data. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1190–1195. IEEE, 2020.

[28] Nicholas Galioto and Alex Arkady Gorodetsky. Bayesian system ID: optimal management of parameter, model, and measurement uncertainty. *Nonlinear Dynamics*, 102(1):241–267, 2020.

[29] Nicholas Galioto and Alex Arkady Gorodetsky. A new objective for identification of partially observed linear time-invariant dynamical systems from input-output data. In *Learning for Dynamics and Control*, pages 1180–1191. PMLR, 2021.

[30] Nicholas Galioto and Alex Arkady Gorodetsky. Likelihood-based generalization of Markov parameter estimation and multiple shooting objectives in system identification. *arXiv preprint arXiv:2212.13902*, 2022.

[31] Andrew Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, 1(3):515–533, 2006.

[32] Gene H Golub and Charles F Van Loan. An analysis of the total least squares problem. *SIAM journal on numerical analysis*, 17(6):883–893, 1980.

[33] Vasiliy Govorukhin. Calculation lyapunov exponents for ode. MATLAB File Exchange, 2020. Retrieved June 29, 2020.

[34] Peter L Green. Bayesian system identification of a nonlinear dynamical system using a novel variant of simulated annealing. *Mechanical Systems and Signal Processing*, 52:133–146, 2015.

[35] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019.

[36] Yuantao Gu, Jian Jin, and Shunliang Mei. $l\_\{0\}$ norm constraint lms algorithm for sparse system identification. *IEEE Signal Processing Letters*, 16(9):774–777, 2009.

[37] Heikki Haario, Leonid Kalachev, and Janne Hakkarainen. Generalized correlation integral vectors: A distance concept for chaotic dynamical systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(6):063102, 2015.

[38] Heikki Haario, Marko Laine, Antonietta Mira, and Eero Saksman. Dram: efficient adaptive mcmc. *Statistics and computing*, 16:339–354, 2006.

[39] Baya Hadid, Eric Duviella, and Stéphane Lecoeuche. Data-driven modeling for river flood forecasting based on a piecewise linear arx system identification. *Journal of Process Control*, 86:44–56, 2020.

[40] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, volume 31. Springer Science & Business Media, 2006.

[41] Maziar S Hemati, Clarence W Rowley, Eric A Deem, and Louis N Cattafesta. Debiasing the dynamic mode decomposition for applied koopman spectral analysis of noisy datasets. *Theoretical and Computational Fluid Dynamics*, 31:349–368, 2017.

[42] Michel Hénon and Carl Heiles. The applicability of the third integral of motion: some numerical experiments. *Astronomical Journal, Vol. 69, p. 73 (1964)*, 69:73, 1964.

[43] Roger W Hockney and James W Eastwood. *Computer simulation using particles*. crc Press, 2021.

[44] Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

[45] Shang Jiang and Jian Zhang. Real-time crack assessment using deep neural networks with wall-climbing unmanned aerial system. *Computer-Aided Civil and Infrastructure Engineering*, 35(6):549–564, 2020.

[46] PS Joarder, VM Nakariakov, and B Roberts. A manifestation of negative energy waves in the solar atmosphere. *Solar Physics*, 176(2):285–297, 1997.

[47] Dominic Jordan and Peter Smith. *Nonlinear ordinary differential equations: an introduction for scientists and engineers*. OUP Oxford, 2007.

[48] Jer-Nan Juang and Richard S Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *Journal of guidance, control, and dynamics*, 8(5):620–627, 1985.

[49] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. Spie, 1997.

[50] Mohammad Khalil, Abhijit Sarkar, Sondipon Adhikari, and Dominique Poirel. The estimation of time-invariant parameters of noisy nonlinear oscillatory systems. *Journal of Sound and Vibration*, 344:81 – 100, 2015.

[51] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[52] Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[53] G. A. Kivman. Sequential parameter estimation for stochastic systems. *Nonlinear Processes in Geophysics*, 10(3):253–259, 2003.

[54] Aditi S Krishnapriyan, Alejandro F Queiruga, N Benjamin Erichson, and Michael W Mahoney. Learning continuous models for continuous physics. *arXiv preprint arXiv:2202.08494*, 2022.

[55] Zhilu Lai and Satish Nagarajaiah. Sparse structural system identification method for nonlinear dynamic systems with hysteresis/inelastic behavior. *Mechanical Systems and Signal Processing*, 117:813–842, 2019.

[56] Gongjie Li, Smadar Naoz, Matt Holman, and Abraham Loeb. Chaos in the test particle eccentric Kozai–Lidov mechanism. *The Astrophysical Journal*, 791(2):86, 2014.

[57] Yan-Jun Liu, Jing Li, Shaocheng Tong, and CL Philip Chen. Neural network control-based adaptive learning design for nonlinear systems with full-state constraints. *IEEE transactions on neural networks and learning systems*, 27(7):1562–1571, 2016.

[58] Anders Logg, Kent-Andre Mardal, and Garth Wells. *Automated solution of differential equations by the finite element method: The FEniCS book*, volume 84. Springer Science & Business Media, 2012.

[59] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. Pde-net: Learning pdes from data. In *International conference on machine learning*, pages 3208–3216. PMLR, 2018.

[60] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

[61] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865–873, 2014.

[62] Giorgos Mamakoukas, Orest Xherija, and Todd Murphey. Memory-efficient learning of stable linear dynamical systems for prediction and control. *Advances in Neural Information Processing Systems*, 33:13527–13538, 2020.

[63] Alexander Marshack. *The roots of civilization: The cognitive beginnings of man's first art, symbol and notation*. Moyer Bell Limited, 1991.

[64] Youssef M Marzouk and Habib N Najm. Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems. *Journal of Computational Physics*, 228(6):1862–1902, 2009.

[65] Youssef M Marzouk, Habib N Najm, and Larry A Rahn. Stochastic spectral methods for efficient bayesian solution of inverse problems. *Journal of Computational Physics*, 224(2):560–586, 2007.

[66] Daniele Masti and Alberto Bemporad. Learning nonlinear state–space models using autoencoders. *Automatica*, 129:109666, 2021.

[67] Alexandre Mauroy and Jorge Goncalves. Koopman-based lifting techniques for nonlinear systems identification. *IEEE Transactions on Automatic Control*, 65(6):2550–2565, 2019.

[68] Giovanni Migliorati, Fabio Nobile, Erik von Schwerin, and Raúl Tempone. Approximation of quantities of interest in stochastic pdes by the random discrete lˆ2 projection on polynomial spaces. *SIAM Journal on Scientific Computing*, 35(3):A1440–A1460, 2013.

[69] Brett Ninness and Soren Henriksen. Bayesian system identification via markov chain monte carlo techniques. *Automatica*, 46(1):40–51, 2010.

[70] Sanha Noh. Posterior inference on parameters in a nonlinear DSGE model via Gaussian-based filters. *Computational Economics*, pages 1–47, 2019.

[71] Samet Oymak and Necmiye Ozay. Non-asymptotic identification of lti systems from a single trajectory. In *2019 American control conference (ACC)*, pages 5655–5661. IEEE, 2019.

[72] Shaowu Pan and Karthik Duraisamy. Long-time predictive modeling of nonlinear dynamical systems using neural networks. *Complexity*, 2018:1–26, 2018.

[73] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[74] Václav Peterka. Bayesian approach to system identification. In *Trends and Progress in System identification*, pages 239–304. Elsevier, 1981.

[75] Arkady Pikovsky and Antonio Politi. *Lyapunov exponents: a tool to explore complex dynamics*. Cambridge University Press, 2016.

[76] Gianluigi Pillonetto, Tianshi Chen, Alessandro Chiuso, Giuseppe De Nicolao, and Lennart Ljung. Regularized linear system identification using atomic, nuclear and kernel-based norms: The role of the stability constraint. *Automatica*, 69:137–149, 2016.

[77] Luigi Piroddi and William Spinelli. An identification algorithm for polynomial narx models based on simulation error minimization. *International Journal of Control*, 76(17):1767–1781, 2003.

[78] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[79] Tong Qin, Kailiang Wu, and Dongbin Xiu. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 395:620–635, 2019.

[80] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

[81] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[82] PK Rajasekaran, N Satyanarayana, and MD Srinath. Optimum linear estimation of stochastic signals in the presence of multiplicative noise. *IEEE Transactions on Aerospace and Electronic Systems*, AES-7(3):462–468, 1971.

[83] Antônio H Ribeiro, Koen Tiels, Jack Umenberger, Thomas B Schön, and Luis A Aguirre. On the smoothness of nonlinear system identification. *Automatica*, 121:109158, 2020.

[84] Clarence W Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S Henningson. Spectral analysis of nonlinear flows. *Journal of fluid mechanics*, 641:115–127, 2009.

[85] Rick Salmon. Hamiltonian fluid mechanics. *Annual Review of Fluid Mechanics*, 20(1):225–256, 1988.

[86] Tuhin Sarkar, Alexander Rakhlin, and Munther A Dahleh. Finite time lti system identification. *The Journal of Machine Learning Research*, 22(1):1186–1246, 2021.

[87] Simo Särkkä. *Bayesian filtering and smoothing*. Cambridge University Press, 2013.

[88] Sebastian Scher. Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning. *Geophysical Research Letters*, 45(22):12–616, 2018.

[89] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[90] Johan Schoukens and Lennart Ljung. Wiener-hammerstein benchmark, 2009.

[91] Diana Serra, Fabio Ruggiero, Alejandro Donaire, Luca Rosario Buonocore, Vincenzo Lippiello, and Bruno Siciliano. Control of nonprehensile planar rolling manipulation: A passivity-based approach. *IEEE Transactions on Robotics*, 35(2):317–329, 2019.

[92] Harsh Sharma, Nicholas Galioto, Alex A Gorodetsky, and Boris Kramer. Bayesian identification of nonseparable hamiltonian systems using stochastic dynamic models. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 6742–6749. IEEE, 2022.

[93] Harsh Sharma, Mayuresh Patil, and Craig Woolsey. A review of structure-preserving numerical methods for engineering applications. *Computer Methods in Applied Mechanics and Engineering*, 366:113067, 2020.

[94] Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In *Conference On Learning Theory*, pages 439–473. PMLR, 2018.

[95] Samuel L Smith, Benoit Dherin, David Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021.

[96] Sebastian Springer, Heikki Haario, Vladimir Shemyakin, Leonid Kalachev, and Denis Shchepakin. Robust parameter estimation of chaotic systems. *Inverse Problems & Imaging*, 13(6):1189, 2019.

[97] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[98] Naoya Takeishi, Yoshinobu Kawahara, Yasuo Tabei, and Takehisa Yairi. Bayesian dynamic mode decomposition. In *IJCAI*, pages 2814–2821, 2017.

[99] Molei Tao. Explicit symplectic approximation of nonseparable Hamiltonians: Algorithm and long time performance. *Physical Review E*, 94(4):043303, 2016.

[100] Anastasios Tsiamis and George J Pappas. Finite sample analysis of stochastic system identification. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3648–3654. IEEE, 2019.

[101] Jonathan H Tu, Clarence W Rowley, Dirk M Luchtenburg, Steven L Brunton, and J Nathan Kutz. On dynamic mode decomposition: theory and applications. *Journal of Computational Dynamics*, 1(2), 2014.

[102] Sabine Van Huffel and Joos Vandewalle. Analysis and properties of the generalized total least squares problem $ax \approx b$ when some or all columns in a are subject to error. *SIAM Journal on Matrix Analysis and Applications*, 10(3):294, 1989.

[103] Richard S Westfall. *The construction of modern science: Mechanisms and mechanics.* Cambridge University Press, 1977.

[104] Matthew O Williams, Maziar S Hemati, Scott TM Dawson, Ioannis G Kevrekidis, and Clarence W Rowley. Extending data-driven koopman analysis to actuated systems. *IFAC-PapersOnLine*, 49(18):704–709, 2016.

[105] Armand Wirgin. The inverse crime. *arXiv preprint math-ph/0401050*, 2004.

[106] Alan Wolf, Jack B. Swift, Harry L. Swinney, and John A. Vastano. Determining lyapunov exponents from a time series. *Physica D: Nonlinear Phenomena*, 16(3):285–317, 1985.

[107] Kailiang Wu, Tong Qin, and Dongbin Xiu. Structure-preserving method for reconstructing unknown Hamiltonian systems from trajectory data. *SIAM Journal on Scientific Computing*, 42(6):A3704–A3729, 2020.

[108] Shiying Xiong, Yunjin Tong, Xingzhe He, Shuqi Yang, Cheng Yang, and Bo Zhu. Nonseparable symplectic neural networks. In *International Conference on Learning Representations*, 2021.

[109] Enoch Yeung, Soumya Kundu, and Nathan Hodas. Learning deep neural network representations for koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pages 4832–4839. IEEE, 2019.

[110] Jinghui Yuan, Mohamed Abdel-Aty, Yaobang Gong, and Qing Cai. Real-time crash risk prediction using long short-term memory recurrent neural network. *Transportation research record*, 2673(4):314–326, 2019.

[111] Yang Zheng and Na Li. Non-asymptotic identification of linear dynamical systems using multiple trajectories. *IEEE Control Systems Letters*, 5(5):1693–1698, 2020.

[112] Yaofeng Desmond Zhong, Biswadip Dey, and Amit Chakraborty. Symplectic ode-net: Learning hamiltonian dynamics with control. In *International Conference on Learning Representations*, 2019.