Scalable Identification of Partially Observed Systems with Certainty-Equivalent EM

Kunal Menda [* 1]   **Jean de Becdelièvre** [* 1]   **Jayesh K. Gupta** [* 1]
Ilan Kroo [1]   **Mykel J. Kochenderfer** [1]   **Zachary Manchester** [1]

Abstract

System identification is a key step for model-based control, estimator design, and output prediction. This work considers the offline identification of partially observed nonlinear systems. We empirically show that the certainty-equivalent approximation to expectation-maximization can be a reliable and scalable approach for high-dimensional deterministic systems, which are common in robotics. We formulate certainty-equivalent expectation-maximization as block coordinate-ascent, and provide an efficient implementation. The algorithm is tested on a simulated system of coupled Lorenz attractors, demonstrating its ability to identify high-dimensional systems that can be intractable for particle-based approaches. Our approach is also used to identify the dynamics of an aerobatic helicopter. By augmenting the state with unobserved fluid states, a model is learned that predicts the acceleration of the helicopter better than state-of-the-art approaches. The codebase for this work is available at https://github.com/sisl/CEEM.

1. Introduction

The performance of controllers and state-estimators for nonlinear systems depends heavily on the quality of the model of system dynamics (Hou & Wang, 2013). System-identification addresses the problem of learning or calibrating dynamics models from data (Ljung, 1999), which is often a time-history of observations of the system and control inputs. In this work, we address the problem of learning dynamics models of partially observed systems (shown
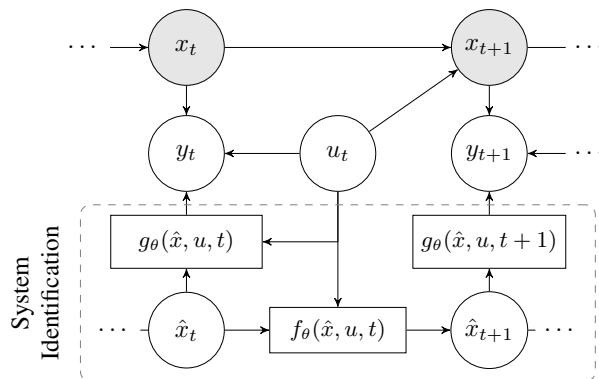
Figure 1. A graphical model representing a partially observed dynamical system. Gray-box identification algorithms attempt to search a model class of dynamics and observation models for the model that maximizes the likelihood of the observations.

in Figure 1) that are high-dimensional and nonlinear. We consider situations in which the system's state cannot be inferred from a single observation, but instead requires inference over time-series of observations.

The problem of identifying systems from partial observations arises in robotics (Punjani & Abbeel, 2015; Cory & Tedrake, 2008; Ordonez et al., 2017) as well as domains such as chemistry (Gustavsson, 1975) and biology (Sun et al., 2008). In many robotic settings, we have direct measurements of a robot's pose and velocity, but in many cases we cannot directly observe relevant quantities such as the temperature of actuators, the state the environment around the robot, or the intentions of other agents. For example, Abbeel et al. (2010) attempted to map the pose and velocity of an aerobatic helicopter to its acceleration. They found their model to be inaccurate when predicting aggressive maneuvers because of the substantial airflow generated by the helicopter that affected the dynamics. Since it is often impossible to directly measure the state of the airflow around a vehicle, identification must be with only partial observability.

System identification is a mature field with a rich history (Ljung, 1999; 2010). A variety of techniques have been proposed to learn predictive models from time-series

data. Autoregressive approaches directly map a time-history of past inputs to observations, without explicitly reasoning about unobserved states (Billings, 2013), and are the state-of-the-art approach to the aforementioned problem of modeling the aerobatic helicopter (Punjani & Abbeel, 2015). In contrast, state-space models (SSM) assume an unobserved state x_t that evolves over time and *emits* observations y_t that we measure, as shown in Figure 1. Recurrent Neural Networks (RNNs) (Bailer-Jones et al., 1998; Zimmermann & Neuneier, 2000) are a form of *black-box* nonlinear SSM that can be fit to observation and input time-series, and Subspace Identification (SID) methods (Van Overschee & De Moor, 1994) can be used to fit linear SSMs.

However, in many cases, prior knowledge can be used to specify structured, parametric models of the system (Gupta et al., 2019; 2020) in state-space form, commonly refered to as *gray-box models*. Such models can be trained with less data and used with a wider array of control and state-estimation techniques than black-box models (Gupta et al., 2019; 2020; Lutter et al., 2019b;a).

To identify partially observed gray-box models, the unobserved state-trajectory is often considered as missing data, and techniques based on Expectation-Maximization (EM) are used (Dempster et al., 1977; Schön et al., 2011; Kantas et al., 2015; Ghahramani & Roweis, 1999). The *smoothing* step (E-step) deals with state inference—the current system dynamics estimate is used to infer the distribution of unobserved state-trajectories conditioned on the observations $p(x_{1:T} \mid y_{1:T})$. This distribution is sometimes called the joint smoothing distribution in the literature, and it is used to estimate the expected log-likelihood of the observations. In the *learning* step (M-step), the system's dynamics estimate is updated such that it maximizes the expected log-likelihood. The *smoothing* step can typically be approached with particle approximations of $p(x_{1:T} \mid y_{1:T})$, but naive implementations of particle smoothers can be computationally intensive and become rapidly intractable in high dimensions. Across various fields and disciplines, numerous methods have been developed to alleviate this computational burden, several of which are discussed in Section 2.3.

This work is motivated by robotics applications, in which the following assumptions are often valid:

- Systems evolve nearly deterministically, which implies that the process noise is small and unimodal, and,
- The distribution of states conditioned on the observations $p(x_{1:T} \mid y_{1:T})$ is unimodal.

We study the benefits of making the *certainty-equivalent* approximation in the E-step of the EM procedure, and we refer to this approach as *CE-EM*. Specifically, we use nonlinear programming to tractably find the maximum-likelihood (ML) point-estimate of the unobserved states, and use it

in lieu of a fully characterized approximate distribution. The contributions of this paper are to describe an efficient implementation of CE-EM, and to test the approximation against state-of-the-art approaches on a variety of system-identification problems. We demonstrate on a system of Lorenz attractors that:

- CE-EM can be faster and more reliable than approaches using particle approximations,
- CE-EM scales to high-dimensional problems, and,
- CE-EM learns unbiased parameter estimates on deterministic systems with unimodal $p(x_{1:T} \mid y_{1:T})$.

We also demonstrate the algorithm on the problem of identifying the dynamics of an aerobatic helicopter. We show that a nonlinear SSM can be trained with CE-EM that outperforms various approaches including the most recent work done on this dataset (Punjani & Abbeel, 2015). A codebase implementing CE-EM and other supplementary material can be found at our website: `https://sites.google.com/stanford.edu/ceem/`.

2. Background

This section states the nonlinear system-identification problem with partial observations and discusses approaches that use EM to solve it.

2.1. Formal Problem Statement

In this work, we assume that we are given a batch of trajectories containing observations $y_{1:T} \in \mathbb{R}^{m \times T}$ of a dynamical system as it evolves over a time horizon T, possibly forced by some known input sequence $u_{1:T}$. We assume that this dynamical system has a state $x \in \mathbb{R}^n$ that evolves and generates observations according to the following equations,

$$\begin{aligned} x_{t+1} &= f(x_t, u_t, t) + w_t, \quad w_t \sim p_w(\cdot) \\ y_t &= g(x_t, u_t, t) + v_t, \quad v_t \sim p_v(\cdot) \end{aligned} \tag{1}$$

where w_t is referred to as the *process noise* and v_t as the *observation noise*. Both w_t and v_t are assumed to be additive for notational simplicity, but this is not a required assumption.[1] Without loss of generality, we can drop the dependence on u_t, absorbing it into the dependence on t.

We further assume that we are provided a class of parameterized models $f_\theta(x, t)$ and $g_\theta(x, t)$ for $\theta \in \Theta$ that approximate the dynamical system's evolution and observation processes. The goal of our algorithm is to find the parameters θ that maximize the likelihood of the observations. That

---

[1]In order to substitute w_t or v_t in the arguments of p_v and p_w, the only requirement is to be able to express them as functions of the other terms. Hence, the formulation is amenable to any method of injecting noise into the system, so long as we can estimate the probability of the noise given the other terms in their respective equations.

is, we seek to find:

$$\theta_{\text{ML}} = \arg\max_{\theta} p(y_{1:T} \mid \theta)$$

$$= \arg\max_{\theta} \int p(y_{1:T}, x_{1:T} \mid \theta) dx_{1:T}$$

$$= \arg\max_{\theta} \int p(y_{1:T} \mid x_{1:T}, \theta) p(x_{1:T} \mid \theta) dx_{1:T} \tag{2}$$

Using the graphical model shown in Figure 1, we can rewrite this integral as:

$$\theta_{\text{ML}} = \arg\max_{\theta} \int \prod_{t=1}^{T} p(y_t \mid x_t, \theta) \cdot$$
$$\prod_{t=1}^{T-1} p(x_{t+1} \mid x_t, \theta) p(x_1 \mid \theta) dx_{1:T} \tag{3}$$

Finally, using the notation chosen to describe a dynamical system in Equation (1), we obtain:

$$\theta_{\text{ML}} = \arg\max_{\theta} \int \left(\prod_{t=1}^{T} p_v(y_t - g_\theta(x_t, t)) \right) \cdot$$
$$\left(p(x_1) \prod_{t=1}^{T-1} p_w(x_{t+1} - f_\theta(x_t, t)) \right) dx_{1:T} \tag{4}$$

The Expectation-Maximization (EM) algorithm has been used in the literature to solve problems of this form.

2.2. Expectation-Maximization for SSM Identification

The EM algorithm is a two-step procedure that copes with the missing state information by forming an approximation $Q(\theta, \theta_k)$ of the joint state and observation likelihood $p_\theta(x_{1:T}, y_{1:T})$ at the kth iteration of the algorithm:

$$Q(\theta, \theta_k) = \int \log \ p(x_{1:T}, y_{1:T} \mid \theta) \cdot$$
$$p(x_{1:T} \mid y_{1:T}, \theta_k) dx_{1:T} \tag{5}$$

Classical proofs show that maximizing $Q(\theta, \theta^k)$ with respect to θ results in increasing $p_\theta(x_{1:T}, y_{1:T})$ (Dempster et al., 1977). The EM algorithm iteratively performs two steps:

1. Compute $Q(\theta, \theta_k)$
2. Update $\theta_{k+1} = \arg\max_{\theta} Q(\theta, \theta_k)$

The evaluation of $Q(\theta, \theta_k)$ in the E-step requires the estimation the $p(x_{1:T} \mid y_{1:T}, \theta_k)$, as well as the integration of the likelihood over this distribution. In the linear Gaussian case, EM can be performed exactly using Kalman smoothing (Rauch et al., 1965; Ghahramani & Hinton, 1996). In the more general case, there is no analytic solution, and previous work on approximating the E-step is summarized in the next section.

2.3. Related Work

There are general approaches that are based on particle representations of the joint states-observations likelihood that use Sequential Monte-Carlo (SMC) methods such as Particle Smoothing (PS) (Schön et al., 2011; Kantas et al., 2015). With enough particles, PS can handle any system and any joint distribution of states and observations. However, PS suffers from the *curse of dimensionality*, requiring an intractably large number of particles if the state space is high-dimensional (Snyder et al., 2008; Kantas et al., 2015). In the simplest form, both the E-step and the M-step can be quadratic in complexity with respect to the number of particles (Schön et al., 2011). An important body of work has attempted to alleviate some of this burden by using Nested SMC (Naesseth et al., 2015) and forward filtering-backward simulation (FFBSi), (Lindsten & Schön, 2013), and conditional particle filtering (Lindsten, 2013) in the E-step. PS can also require variance reduction techniques, such as full-adaptation, to perform reliably, and likelihood estimates in the E-step can be noisy (Svensson et al., 2018). Stochastic Approximation EM can be used to stabilize learning in the presence of this noise (Delyon et al., 1999; Svensson & Lindsten, 2018).

Another group of methods is based on linearizing the dynamics around the current state estimates to obtain a time-dependant linear Gaussian dynamical system, to which we can apply Kalman smoothing techniques. Ghahramani & Roweis (1999) use Extended Kalman Smoothing (EKS) for a fast E-step and show that, when using radial basis functions to describe the system dynamics, no assumption is required in the M-step. The drawbacks of this approach are that the number of radial basis functions required to accurately represent a function grows exponentially with the input dimension, that the user cannot specify a gray-box SSM, and that EKS performance can vary dramatically with the hyperparameters. Goodwin & Aguero (2005) proposed a method called MAP-EM, which linearizes around the state-trajectory that maximizes joint state and observation likelihood, approximates process and observation noise as Gaussian, and performs a *local* version of EM using this linearization. In comparison, the method we consider assumes that the maximum likelihood state-trajectory estimate concentrates all of the probability mass. Goodwin & Aguero (2005) call this simplification Certainty Equivalent EM (CE-EM) and report worse results than their approach on simple examples.

This paper shows that CE-EM can actually be a fast, simple, and reliable approach. The method we use here is tailored to systems that are not dominated by process noise, in which the state-trajectory distribution is unimodal, and whose high dimensional state-space make other methods intractable. Such scenarios are regularly encountered in robotics, but perhaps less so in other applications domains of system

identification such as chemistry, biology, or finance.

3. Methodology

This section introduces the CE-EM approximation and presents an efficient algorithm to identify high-dimensional robotic systems.

3.1. Certainty-Equivalent Expectation Maximization

Under the certainty-equivalence condition, the distribution of states conditioned on observations is assumed to be a Dirac delta function. That is, we assume:

$$p(x_{1:T}|y_{1:T}, \theta) = \delta_{x_{1:T}^{\mathrm{ML}}(\theta)}(x_{1:T}) \tag{6}$$

where

$$
\begin{aligned}
x_{1:T}^{\mathrm{ML}}(\theta) &= \arg\max_{x_{1:T}} p(x_{1:T} \mid y_{1:T}, \theta) \\
&= \arg\max_{x_{1:T}} p(x_{1:T}, y_{1:T} \mid \theta)/p(y_{1:T} \mid \theta) \\
&= \arg\max_{x_{1:T}} p(x_{1:T}, y_{1:T} \mid \theta)
\end{aligned}
\tag{7}
$$

In other words, this assumption implies that there is only one state trajectory that satisfies the system dynamics, and is coherent with the data.

The E-step can be rewritten as:

$$
\begin{aligned}
Q(\theta, \theta_k) &= \int \log\, p(x_{1:T}, y_{1:T} \mid \theta) \cdot \\
&\qquad \delta_{x_{1:T}^{\mathrm{ML}}(\theta_k)}(x_{1:T}) dx_{1:T} \\
&= \log p(x_{1:T}^{\mathrm{ML}}(\theta_k), y_{1:T} \mid \theta)
\end{aligned}
\tag{8}
$$

By observing the maximization of the expression in Equation (7) as the E-step, and maximizing the expression in Equation (8) as the M-step, we see that CE-EM is simply block coordinate-ascent on the single joint-objective:

$$
\begin{aligned}
J(x_{1:T}, \theta) &= \log p(x_{1:T}, y_{1:T} \mid \theta) \\
&= \log p(x_1) + \sum_{t=1}^{T} \log p_v(y_t - g_\theta(x_t, t)) \\
&\quad + \sum_{t=1}^{T-1} \log p_w(x_{t+1} - f_\theta(x_t, t))
\end{aligned}
\tag{9}
$$

Jointly maximizing this objective over $x_{1:T}$ and θ yields θ_{ML}.

Though this objective can be optimized as is by a nonlinear optimizer, it is not necessarily efficient to do so since θ and $x_{1:T}$ are highly coupled, leading to inefficient and potentially unstable updates. For this reason, we still opt for the block coordinate-ascent approach similar to EM, where θ and $x_{1:T}$ are each sequentially held constant while the other

---

Algorithm 1. CE-EM Implementation

Input: observations $y_{1:T}$, control inputs $u_{1:T}$, stopping criterion tol
Initialize: model parameters θ
Initialize: hidden state estimates $x_{1:T}$
repeat
    $J^k \leftarrow J(x_{1:T}^k, \theta^k)$
    $x_{1:T}^{k+1} \leftarrow \arg\max\limits_{\tilde{x}_{1:T}} J(\tilde{x}_{1:T}, \theta^k) - \rho_x \|\tilde{x}_{1:T} - x_{1:T}^k\|_2^2$
    $\theta^{k+1} \leftarrow \arg\max\limits_{\tilde{\theta}} J(x_{1:T}^{k+1}, \tilde{\theta}) - \rho_\theta \|\tilde{\theta} - \theta^k\|_2^2 + \log p(\tilde{\theta})$
until $J^k - J(x_{1:T}^{k+1}, \theta^{k+1}) \leq tol$

---

is optimized. By viewing EM as block coordinate-ascent, we may also borrow good practices typically employed when running the algorithm (Shi et al., 2016). In particular, we employ trust-region regularization in order to guarantee convergence to a local optimum, even in non-convex settings (Grippo & Sciandrone, 2000).

3.1.1. SMOOTHING

At iteration k, we first perform *smoothing* by holding θ constant and finding the ML point-estimate for $x_{1:T}$ as follows:

$$x_{1:T}^{(k)} = \arg\max_{x_{1:T}} J(x_{1:T}, \theta^{(k-1)}) - \rho_x \|x_{1:T} - x_{1:T}^{(k-1)}\|_2^2 \tag{10}$$

where ρ_x scales a soft trust-region regularizer similar to damping terms found in Levenberg-Marquardt methods (Levenberg, 1944; Marquardt, 1963). We note that the Hessians of this objective with respect to $x_{1:T}$ are block-sparse, and as a result this step can be efficiently solved with a second-order optimizer in $\mathcal{O}(n^3 T)$.

If the cost function $J(x_{1:T}, \theta)$ is a sum of quadratic terms, i.e. if all stochasticity is Gaussian, then the smoothing can be solved with a Gauss-Newton method, where the Hessian matrix is approximated using first-order (Jacobian) information. In this case, the solution to the smoothing step is equivalent to iteratively performing EKS (Aravkin et al., 2017). However, EKS typically involves forward and backward passes along each trajectory, performing a Riccati-like update at each time step. Though square-root versions of the algorithm exist to improve its numerical stability (Bierman, 1981), solving the problem with batch nonlinear least-squares as opposed to iteratively performing a forward-backward method can improve stability for long time-series (Van Dooren, 1981).

3.1.2. LEARNING

In the *learning* step, we hold $x_{1:T}$ constant and find:

$$\theta^{(k)} = \arg\max_\theta J(x_{1:T}^{(k)}, \theta) - \rho_\theta \|\theta - \theta^{(k-1)}\|_2^2 + \log p(\theta) \tag{11}$$

Again, ρ_θ scales a soft trust-region regularizer, and specifying $\log p(\theta)$ allows us to regularize θ toward a prior. The above optimization problem can be solved using any nonlinear optimizer. We find the Nelder-Mead (Gao & Han, 2012) scheme well-suited for small parameter spaces, and first-order schemes such as Adam (Kingma & Ba, 2015), or quasi-second-order schemes such as L-BFGS (Liu & Nocedal, 1989) suited for larger parameter spaces such as those of neural networks. The routine, summarized in Algorithm 1, iterates between the smoothing and learning steps until convergence.

It should be noted that making the certainty-equivalent approximation is generally known to bias parameter estimates (Celeux & Govaert, 1992), but can yield the correct solutions under the assumptions we make (Neal & Hinton, 1998).

4. Experiments

Readers of Goodwin & Aguero (2005) will likely be left with the incorrect impression that CE-EM is an inferior method that would perform poorly in practice. The objective of our experiments is to demonstrate that the CE-EM algorithm is capable of identifying high-dimensional nonlinear systems in partially observed settings. We do so in simulation by identifying the parameters of a system of partially observed coupled Lorenz attractors, as well as by identifying the dynamics of a real aerobatic helicopter. In the second experiment, we build on previous analysis of the dataset (Abbeel et al., 2010; Punjani & Abbeel, 2015) by attempting to characterize the interaction of the helicopter with the fluid around it, without having any direct observation of the fluid state. Instructions for reproducing all experiments are included in the supplementary material.

4.1. Identification of Coupled Lorenz Systems

In this experiment, we show that:

1. CE-EM learns unbiased parameter estimates of systems that are close to deterministic, and,
2. CE-EM scales to high-dimensional problems in which particle-based methods can be intractable.

To justify these claims, we use a system that is sufficiently nonlinear and partially observable to make particle-based smoothing methods intractable. We choose a system of coupled Lorenz attractors for this purpose, owing to their ability to exhibit chaotic behavior and their use in nonlinear atmospheric and fluid flow models (Bergé et al., 1984). Arbitrary increases in state dimensionality can be achieved by coupling multiple individual attractors. The state of a system with K coupled Lorenz attractors is $x \in \mathbb{R}^{3K} = \{\ldots, x_{1,k}, x_{2,k}, x_{3,k}, \ldots\}$. The dynamics of

Table 1. Mean parameter estimates and standard errors for a single Lorenz system simulated with various σ_w and σ_v.

σ_w	σ_v	σ	ρ	β
0.001	0.01	10.011(0.012)	28.000(0.001)	2.667(0.000)
0.010	0.01	10.017(0.012)	28.000(0.001)	2.668(0.001)
0.100	0.01	10.064(0.036)	27.996(0.013)	**2.676 (0.004)**
0.001	0.05	10.006(0.016)	27.998(0.002)	2.666(0.001)
0.001	0.10	9.998(0.022)	27.995(0.004)	2.665(0.001)

the system are as follows:

$$
\begin{aligned}
\dot{\tilde{x}}_{1,k} &= \sigma_k(x_{2,k} - x_{1,k}) \\
\dot{\tilde{x}}_{2,k} &= x_{1,k}(\rho_k - x_{3,k}) - x_{2,k} \\
\dot{\tilde{x}}_{3,k} &= x_{1,k}x_{2,k} - \beta_k x_{3,k} \\
\dot{x} &= \dot{\tilde{x}} + Hx
\end{aligned}
\tag{12}
$$

where H is an $\mathbb{R}^{3K \times 3K}$ matrix.

We nominally set the parameters $(\sigma_k, \rho_k, \beta_k)$ to the values $(10, 28, 8/3)$, and randomly sample the entries of H from a normal distribution to generate chaotic and coupled behavior between attractors, while avoiding self-coupling. These parameters are estimated during identification. In order to make the system partially observed, the observation $y \in \mathbb{R}^{(3K-2)}$ is derived from x as follows:

$$
y = Cx + v, \quad v \sim \mathcal{N}(0, \sigma_v^2 \mathbf{I})
\tag{13}
$$

where $C \in \mathbb{R}^{(3K-2) \times 3K}$ is a known matrix with full row-rank, and v is the observation noise sampled from a Gaussian with diagonal covariance $\sigma_v^2 \mathbf{I}$. The entries of C are also randomly sampled from a standard normal distribution. In the following experiments, we simulate the system for $T = 128$ timesteps at a sample rate of $\Delta t = 0.04$s, and integrate the system using a 4th-order Runge-Kutta method. Initial conditions for each trajectory are sampled such that $x_{1,k} \sim \mathcal{N}(-6, 2.5^2), x_{2,k} \sim \mathcal{N}(-6, 2.5^2), x_{3,k} \sim \mathcal{N}(24, 2.5^2)$.

4.1.1. UNBIASED ESTIMATION IN DETERMINISTIC SETTINGS

To test the conditions under which CE-EM learns unbiased parameter estimates, we simulate a single Lorenz system with $H = \mathbf{0}$ and known $C \in \mathbb{R}^{2 \times 3}$. We introduce and vary the process noise $w \sim \mathcal{N}(0, \sigma_w^2 \mathbf{I})$, and vary the observation noise coefficient σ_v, and then attempt to estimate the parameters (σ, ρ, β). Using initial guesses within 10% of the system's true parameter values, we run CE-EM on a single sampled trajectory. For each choice of σ_w and σ_v, we repeat this process for 10 random seeds.

Table 1 shows the mean and standard errors of parameter estimates for various σ_w and σ_v. We highlight in red the mean estimates that are not within two standard errors of

their true value. We see that σ and ρ are estimated without bias for all scenarios. However, the estimate of β appears to become biased as the process noise is increased, but not as the observation noise is increased. This supports the assumption that the objective used in CE-EM is sound when systems evolve close to deterministically, but can be biased if it is not.

4.1.2. COMPARISON TO PARTICLE BASED METHODS

In Section 2, we discussed methods for parameter estimation in state-space systems that are based on particle-filtering and smoothing (Kantas et al., 2015). Since these methods approximate the distribution over unobserved state-trajectories in their E-step, as opposed to only their point estimate, such methods can be asymptotically unbiased. However, for a finite number of particles, such methods can result in high-variance estimates. In this experiment, we compare the bias resulting from using CE-EM with the variance of using a state-of-the-art Particle EM algorithm.

We attempt to identify the parameters (σ, ρ, β) of the same single Lorenz system as in Section 4.1.1. However, we introduce process noise $w \sim \mathcal{N}(0, 0.1^2\mathbf{I})$ and observation noise $v \sim \mathcal{N}(0, 0.5^2\mathbf{I})$. We use a training dataset of four trajectories sampled with conditions specified in Section 4.1.

The performance of Particle EM can vary substantially depending on implementation of the particle filter and smoother in the E-step, and implementation of the M-step. We use a fully-adapted particle filter with systematic and adaptive resampling, following the recommendations of Doucet et al. (2000); Hol et al. (2006). Furthermore, we use the FFBSi algorithm (Godsill et al., 2004; Lindsten & Schön, 2013) in order to generate iid samples of smoothed state-trajectories, while avoiding complexity that is quadratic in the number of particles experienced by the forward filter-backward smoother (FFBSm) approach (Doucet et al., 2000). We then use Stochastic Approximation EM (SAEM) (Delyon et al., 1999) to perform the M-step.[2] We use $N_p = 100$ particles for filtering, and sample $N_s = 10$ smoothed trajectories using FFBSi.

Figure 2 shows the estimated parameters versus EM epoch using CE-EM and Particle EM. We plot learning curves for 10 random seeds, each of which initializes parameter estimates to within 10% of their true value, and uses a different set of training trajectories. We see that CE-EM consistently converges to accurate parameter estimates in approximately 5 epochs. Estimates of Particle EM appear to initially diverge but in all but one case converge to a similar accuracy in 50 epochs. Furthermore, since the com-

Figure 2. Comparison of CE-EM and Particle EM on parameter estimation of a single Lorenz system.

plexity of FFBSi is $O(N_p N_s)$,[3] the runtime per epoch of Particle EM is $47\times$ more than that of CE-EM.[4] Since the variance of particle-based methods generally increases with the effective dimension of the system, the bias induced by CE-EM may be a worthwhile trade-off for fast and reliable parameter estimation.

4.1.3. CONVERGENCE OF CE-EM ON HIGH-DIMENSIONAL PROBLEMS

To demonstrate that CE-EM is capable of identifying high-dimensional systems, we show that we can estimate the dynamics of an 18 dimensional system of six coupled Lorenz attractors. Moreover, we test whether CE-EM converges to more accurate estimates when more trajectories are provided. To test these claims, we sample 2, 4, and 8 trajectories from a deterministic system with parameters θ_{true}, and $\sigma_v = 0.01$. We randomly initialize each element of the parameters being optimized ($\theta = [\sigma_{1:K}, \rho_{1:K}, \beta_{1:K}, H]$) to within 10% of the their value in θ_{true}. We then run CE-EM on each batch, tracking the error in the estimated dynamics as training proceeds. We measure this error, which we call $\epsilon(\theta)$, as follows:

$$\epsilon(\theta) = \mathbf{E}_{x \sim p(x_0)} \left[\|f_\theta(x) - f_{\theta_{\text{true}}}(x)\|_2 \right] \quad (14)$$

---

[2]We have found that using FFBSi with SAEM performs more reliably than FFBSm with the M-step recommended by (Schön et al., 2011), and both implementations can be found in the associated codebase.
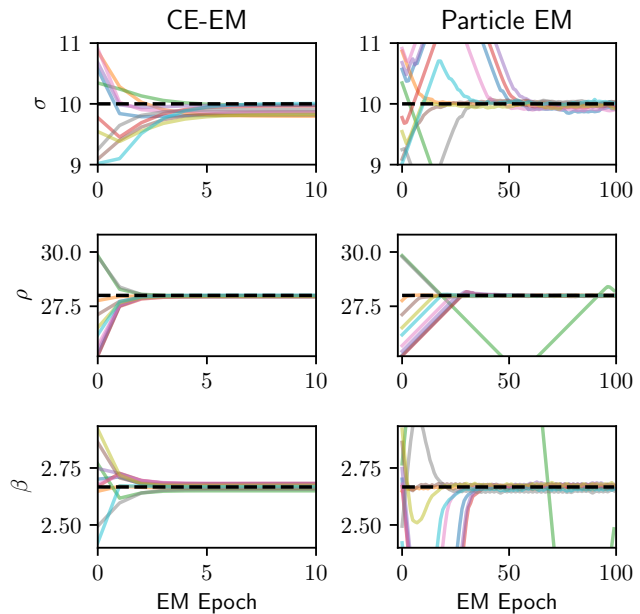
[3]Variants of FFBSi algorithm improve on the $O(N_p N_s)$ complexity (Lindsten & Schön, 2013).

[4]Experiments were run on a computer with an Intel® Core™ i7-6700K CPU @ 4.00GHz $\times 8$ processor and 32GB of memory. Operations are parallelized across trajectories for both implementations.
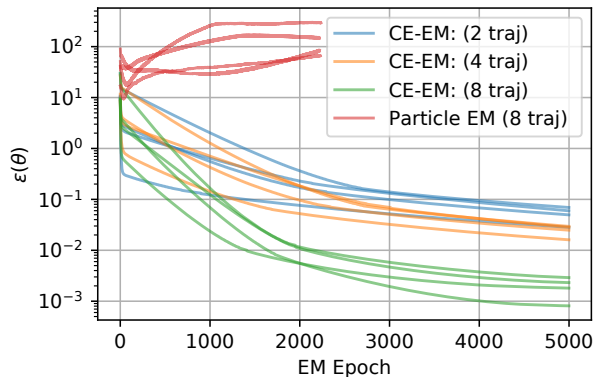
Figure 3. Error in estimated dynamics as CE-EM trains on a varied number of trajectories from a system of six coupled Lorenz attractors.

In the learning step, we do not regularize θ to a prior and set $\rho_\theta = 0$.

For comparison, we also run the same Particle EM implementation as in the previous experiment on this problem. We use the same N_p, N_s, and hyperparameters as before.

Figure 3 shows the results of this experiment for four random seeds for each batch size. We can see that, as the number of trajectories used in training increases, the error in the estimated dynamics tends toward zero. Furthermore, we see that CE-EM convergences monotonically to a local optimum in all cases. In contrast, Particle EM appears to initially improve but then converges to very poor parameter estimates.

The experiments conducted thus far have demonstrated that CE-EM can learn unbiased parameter estimates of nearly-deterministic systems, and can scale to high-dimensional problems for which particle-based methods are intractable. In the next experiment, we use CE-EM to characterize the effect of unobserved states on the dynamics of an aerobatic helicopter.

4.2. Characterizing Aerobatic Helicopter Dynamics

Characterizing the dynamics of a helicopter undergoing aggressive aerobatic maneuvers is widely considered to be a challenging system-identification problem (Abbeel et al., 2010; Punjani & Abbeel, 2015). The primary challenge is that the forces on the helicopter depend on the induced state of the fluid around it. The state of the fluid cannot be directly observed and its dynamics model is unknown. Merely knowing the state of the helicopter and the control commands at a given time does not contain enough information to accurately predict the forces that act on it.

In order to address this issue, (Punjani & Abbeel, 2015) use an approach based on Takens theorem, which suggests that

a system's state can be reconstructed with a finite number of time-lagged observations of it (Takens, 1981). Instead of attempting to estimate the unobserved fluid state, they directly learn a mapping from a $0.5\,\mathrm{s}$ history of observed state measurements and control commands to the forces acting on the helicopter.

This approach is equivalent to considering the past $0.5\,\mathrm{s}$ of observations as the system's state. However, it can require a very large number of lagged observations to represent complex phenomena. In reality, the characteristic time of unsteady flows around helicopters can easily be up to tens of seconds. Having such a high dimensional state can make the control design and state-estimation more complicated. To avoid large input dimensions, a trade-off between the duration of the history and sample frequency is necessary. This trade-off will either hurt the resolution of low-frequency content or will alias high-frequencies. We attempt to instead explicitly model the unobserved states affecting the system.

4.2.1. OBJECTIVE AND DATASET

The objective of this learning problem is to predict y_t, the helicopter's acceleration at time t, from an input vector u_t containing the current measured state of the helicopter (its velocity and rotation rates) and the control commands.

We use data collected by the Stanford Autonomous Helicopter Project (Abbeel et al., 2010). Trajectories are split into $10\,\mathrm{s}$ long chunks and then randomly distributed into train, test, and validation sets according to the established protocol (Abbeel et al., 2010; Punjani & Abbeel, 2015) and summarized in Appendix A.1. The train, test, and validation sets respectively contain 466, 100, and 101 trajectories of 500 time-steps each.

A simple success metric on a given trajectory is the root-mean-squared prediction error,

$$\mathrm{RMSE} = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \left\| y_t^{(\text{measured})} - y_t^{(\text{pred})} \right\|_2^2} \quad (15)$$

where $y_t^{(\text{measured})}$ is the measured force from the dataset, $y_t^{(\text{pred})}$ is the force predicted by the model, and T is the number of time-steps in each trajectory.

4.2.2. PREVIOUS WORK AND BASELINES

Naive: We first consider a naive baseline that does not attempt to account for the time-varying nature of the fluid-state. We train a neural-network to map only the current helicopter state and control commands to the accelerations: $y_t = \mathrm{NN}_{\theta_n}(u_t)$, where NN_{θ_n} is a neural-network with parameters θ_n. We refer to this model as the *naive model*.

H25: We also compare to the work of Punjani & Abbeel (2015). They predict y_t using a time-history $u_{t-H:t}$ of
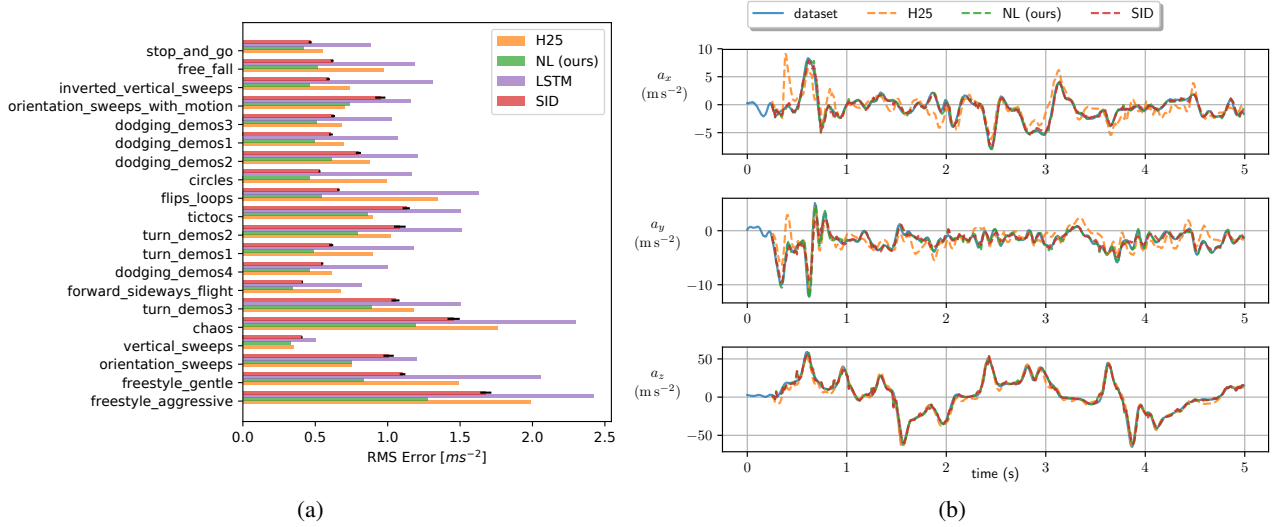
(a)

(b)

Figure 4. (a): Test performance of optimized models on various trajectories. Error bars on *SID* represent the standard deviation of performance of the 10 trained models. (b): Predicted acceleration along axis x, y and z in the body frame for one of the harder test set trajectories. Larger versions of these plots, including rotational accelerations, can be found in Appendix A.3.

$H = 25$ lagged observations of the helicopter's measured state and control commands. This input is passed through a ReLU-activated neural network with a single hidden-layer combined with what they call a Quadratic Lag Model. As a baseline, we reproduce their performance with a single deep neural network $y_t = \text{NN}_{\theta_h}(u_{t-H:t})$ with parameters θ_h. We call this neural network model the *H25 model*. Both of these models can be trained via stochastic gradient descent to minimize the Mean-Squared-Error (MSE) of their predictions for y. The optimization methodology for these models is described in Appendix A.2.

SID: As a third baseline, we compare with subspace-identification methods (Van Overschee & De Moor, 1994). We let $\tilde{y}_t = y_t - \text{NN}_{\theta_n}(u_t)$ be the prediction errors of the trained naive model. We use the MATLAB command `n4sid` to fit a linear dynamical system of the following form:

$$x_{t+1} = A_s x_t + B_s u_t; \qquad \tilde{y}_t = C_s x_t + D_s u_t \qquad (16)$$

Here, $x \in R^d$ is the unobserved state with arbitrary dimension d. The learned parameters are $\theta_s = [A_s, B_s, C_s, D_s]$. We use a state dimension of 10 and call this model the *SID model*. The `n4sid` algorithm scales super-linearly with the amount of data supplied, and thus we train on 10 randomly sampled subsets of 100 trajectories each, and report the distribution in prediction performance. This approach fits a linear system to the *residual error* of the naive model, therefore the prediction of y_t obtained from the *SID model* is a nonlinear function of the states x_t.

LSTM: We also train an LSTM (Hochreiter & Schmidhuber, 1997) on the residual time-series $\tilde{y}_{1:T}$ and $u_{1:T}$.

4.2.3. NONLINEAR UNOBSERVED STATE MODEL

We use CE-EM to train a nonlinear SSM. Similar to the parameterization used for subpace-identification, we fit the prediction errors of the naive model using the following dynamical system:

$$
\begin{aligned}
x_{t+1} &= A_{\text{NL}} x_t + B_{\text{NL}} u_t \\
\tilde{y}_t &= C_{\text{NL}} x_t + D_{\text{NL}} u_t + \text{NN}_{\bar{\theta}_{\text{NL}}}(x_t, u_t)
\end{aligned}
\qquad (17)
$$

where $\text{NN}_{\bar{\theta}_{\text{NL}}}$ is a neural network, and $\theta_{\text{NL}} = [A_{\text{NL}}, B_{\text{NL}}, C_{\text{NL}}, D_{\text{NL}}, \bar{\theta}_{\text{NL}}]$ are the learned parameters. We introduce nonlinearity only in the observation function because it is known from Koopman theory that a nonlinear system can be approximated by a high-dimensional linear system provided the correct nonlinear mapping between them (Brunton et al., 2016).

While learning, we assume that both process and observation noise are distributed with diagonal Gaussian covariance matrices $\sigma_w \mathbf{I}$ and $\sigma_v \mathbf{I}$ respectively. The values of σ_w and σ_v are treated as hyperparmeters of CE-EM, and are both set to 1. Here as well, we use a state dimension of 10 and call this model the *NL model*. The optimization methodology for this model is described in Appendix A.2.

It should be noted that the system we learn need not actually correspond to an interpretable model of the fluid-state, but only of time-varying hidden-states that are useful for predicting the accelerations of the helicopter. Expert knowledge of helicopter aerodynamics could be used to further inform a gray-box model trained with CE-EM.

4.2.4. EVALUATION METHODOLOGY

The test RMSE of the *naive*, *H25*, and *LSTM* models can be evaluated directly on the test trajectories using next-step prediction. However, the *SID* and *NL* models require an estimate of the unobserved state before making a prediction. The natural analog of next-step prediction is extended Kalman filtering (EKF), during which states are recursively predicted and corrected given observations. At a given time-step, a prediction of $\tilde{y}_t$ is made using the current estimate of x_t, and is used in the computation of RMSE. The state-estimate is then corrected with the measured $\tilde{y}_t$.

4.2.5. RESULTS

Figure 4a shows the RMSE of the compared models on trajectories in the test-set. We see that the *NL* model is able to consistently predict the accelerations on the helicopter with better accuracy than any of the other models. The naive model performs on average 2.9 times worse than the *H25* model, and its results can be found in Appendix A.3. The *LSTM* model also performs poorly, on average 1.4 times worse than the *H25* model. The *SID* model notably outperforms the state-of-the-art *H25* model, suggesting that a large linear dynamical system can be used to approximate a nonlinear and partially observable system (Korda & Mezić, 2018). However, introducing nonlinearity as in the *NL* model noticeably improves performance.

Figure 4b depicts the errors in prediction over a sample trajectory in the test-set. Here, we also see that the *NL* model is able to attenuate the time-varying error present in predictions made by the *H25*, suggesting that it has accurately characterized the dynamics of unobserved, time-varying states.

This experiment validates the effectiveness of CE-EM to identify a nonlinear dynamical model of unobserved states that affect the forces acting an aerobatic helicopter.

5. Conclusions

This paper presented an algorithm for system identification of nonlinear systems given partial state observations. The algorithm optimizes system parameters given a time history of observations by iteratively finding the most likely state-history, and then using it to optimize the system parameters. The approach is particularly well suited for high-dimensional and nearly deterministic problems.

In simulated experiments on a partially observed system of coupled Lorenz attractors, we showed that CE-EM can perform identification on a problem that particle-based EM methods are ill-suited for. However, we also find that CE-EM yields biased parameter estimates in the presence of large process noise. This bias can be partially mitigated

by locally approximating the posterior state-marginals as Gaussian, as is done by MAP-EM (Goodwin & Aguero, 2005). We then used the algorithm to model the time-varying hidden-states that affect the dynamics of an aerobatic helicopter. The model trained with CE-EM outperforms state-of-the-art methods because it is able to fit large nonlinear models to unobserved states.

Numerous system-identification problems can be studied using CE-EM. Recently, there have been tremendous efforts to characterize predictive models for the spread of COVID-19 (Fanelli & Piazza, 2020). Limited capacity for testing the prevalence of the disease makes relevant states partially observed, and thus CE-EM may be useful for its modeling. We also hope to apply CE-EM to very high-dimensional systems with sparsely coupled dynamics using general-form consensus optimization (Boyd et al., 2011).

Acknowledgments

References

Abbeel, P., Coates, A., and Ng, A. Y. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

Aravkin, A., Burke, J. V., Ljung, L., Lozano, A., and Pillonetto, G. Generalized Kalman smoothing: Modeling and algorithms. *Automatica*, 86:63–86, 2017.

Bailer-Jones, C. A., MacKay, D. J., and Withers, P. J. A recurrent neural network for modelling dynamical systems. *Network: Computation in Neural Systems*, 9(4):531–547, 1998.

Bergé, P., Pomeau, Y., and Vidal, C. *Order within Chaos: Towards a Deterministic Approach to Turbulence*. Wiley & Sons, 1984.

Bierman, G. J. A new computationally efficient fixed-interval, discrete-time smoother. In *IEEE Conference on Decision and Control (CDC)*, pp. 1054–1060, 1981.

Billings, S. A. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Wiley, 2013.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the

alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Brunton, S. L., Brunton, B. W., Proctor, J. L., and Kutz, J. N. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2), 2016.

Celeux, G. and Govaert, G. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14(3):315–332, 1992.

Cory, R. and Tedrake, R. Experiments in fixed-wing UAV perching. In *AIAA Guidance, Navigation and Control Conference*, 2008.

Delyon, B., Lavielle, M., Moulines, E., et al. Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics*, 27(1):94–128, 1999.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Doucet, A., Godsill, S., and Andrieu, C. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

Fanelli, D. and Piazza, F. Analysis and forecast of COVID-19 spreading in China, Italy and France. *Chaos, Solitons & Fractals*, 134:109761, 2020.

Gao, F. and Han, L. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Computational Optimization and Applications*, 51(1):259–277, 2012. doi: 10.1007/s10589-010-9329-3.

Ghahramani, Z. and Hinton, G. E. Parameter estimation for linear dynamical systems. Technical report, CRG-TR-96-2, University of Totronto, Dept. of Computer Science, 1996.

Ghahramani, Z. and Roweis, S. T. Learning nonlinear dynamical systems using an EM algorithm. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 431–437, 1999.

Godsill, S. J., Doucet, A., and West, M. Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99(465):156–168, 2004.

Goodwin, G. C. and Aguero, J. C. Approximate em algorithms for parameter and state estimation in nonlinear stochastic models. In *IEEE Conference on Decision and Control (CDC)*, pp. 368–373, 2005.

Grippo, L. and Sciandrone, M. On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. *Operations research letters*, 26(3):127–136, 2000.

Gupta, J. K., Menda, K., Manchester, Z., and Kochenderfer, M. J. A general framework for structured learning of mechanical systems. *arXiv preprint arXiv:1902.08705*, 2019.

Gupta, J. K., Menda, K., Manchester, Z., and Kochenderfer, M. J. Structured mechanical models for robot learning and control. In *Conference on Learning for Dynamics and Control (L4DC)*, 2020.

Gustavsson, I. Survey of applications of identification in chemical and physical processes. *Automatica*, 11(1):3–24, 1975.

Hochreiter, S. and Schmidhuber, J. LSTM can solve hard long time lag problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 473–479, 1997.

Hol, J. D., Schon, T. B., and Gustafsson, F. On resampling algorithms for particle filters. In *IEEE Nonlinear Statistical Signal Processing Workshop*, pp. 79–82, 2006.

Hou, Z.-S. and Wang, Z. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013.

Jones, E., Oliphant, T., Peterson, P., et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/.

Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351, 2015.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Korda, M. and Mezić, I. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, July 2018. doi: 10.1016/j.automatica.2018.03.046.

Levenberg, K. A method for the solution of certain nonlinear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, July 1944. doi: 10.1090/qam/10666.

Lindsten, F. An efficient stochastic approximation em algorithm using conditional particle filters. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6274–6278, 2013.

Lindsten, F. and Schön, T. B. Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends® in Machine Learning*, 6(1):1–143, 2013.

Liu, D. C. and Nocedal, J. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

Ljung, L. System identification. *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–19, 1999.

Ljung, L. Perspectives on system identification. *Annual Reviews in Control*, 34:1–12, 2010.

Lutter, M., Listmann, K., and Peters, J. Deep Lagrangian networks for end-to-end learning of energy-based control for under-actuated systems. In *International Conference on Intelligent Robots and Systems (IROS)*, 2019a.

Lutter, M., Ritter, C., and Peters, J. Deep Lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations (ICLR)*, 2019b.

Marquardt, D. W. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, June 1963. doi: 10.1137/0111030.

Menda, K., de Becdelièvre, J., Gupta, J. K., Kroo, I., Kochenderfer, M. J., and Manchester, Z. Normalized Stanford helicopter dataset, February 2020. URL https://doi.org/10.5281/zenodo.3662987.

Naesseth, C. A., Lindsten, F., and Schön, T. B. Nested sequential Monte Carlo methods. *International Conference on Machine Learning (ICML)*, 2015.

Neal, R. M. and Hinton, G. E. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pp. 355–368. Springer, 1998.

Ordonez, C., Gupta, N., Reese, B., Seegmiller, N., Kelly, A., and Collins Jr, E. G. Learning of skid-steered kinematic and dynamic models for motion planning. *Robotics and Autonomous Systems*, 95:207–221, 2017.

Punjani, A. and Abbeel, P. Deep learning helicopter dynamics models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3223–3230, 2015.

Rauch, H. E., Tung, F., and Striebel, C. T. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965.

Schön, T. B., Wills, A., and Ninness, B. System identification of nonlinear state-space models. *Automatica*, 47(1): 39–49, 2011.

Shi, H.-J. M., Tu, S., Xu, Y., and Yin, W. A primer on coordinate descent algorithms. *arXiv preprint arXiv:1610.00040*, 2016.

Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J. Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640, 2008.

Sun, X., Jin, L., and Xiong, M. Extended kalman filter for estimation of parameters in nonlinear state-space models of biochemical networks. *PLOS One*, 3(11):e3758, 2008.

Svensson, A. and Lindsten, F. Learning dynamical systems with particle stochastic approximation em. *arXiv preprint arXiv:1806.09548*, 2018.

Svensson, A., Lindsten, F., and Schön, T. B. Learning nonlinear state-space models using smooth particle-filter-based likelihood approximations. *IFAC-PapersOnLine*, 51(15):652–657, 2018.

Takens, F. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence*, pp. 366–381. Springer, 1981.

Van Dooren, P. A generalized eigenvalue approach for solving riccati equations. *SIAM Journal on Scientific and Statistical Computing*, 2(2):121–135, 1981.

Van Overschee, P. and De Moor, B. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93, 1994.

Zimmermann, H. and Neuneier, R. Modeling dynamical systems by recurrent neural networks. *WIT Transactions on Information and Communication Technologies*, 25: 557–566, 2000.