Gradient-free training of neural ODEs for system identification and control using ensemble Kalman inversion

Lucas Böttcher [1]

Abstract

Ensemble Kalman inversion (EKI) is a sequential Monte Carlo method used to solve inverse problems within a Bayesian framework. Unlike backpropagation, EKI is a gradient-free optimization method that only necessitates the evaluation of artificial neural networks in forward passes. In this study, we examine the effectiveness of EKI in training neural ordinary differential equations (neural ODEs) for system identification and control tasks. To apply EKI to optimal control problems, we formulate inverse problems that incorporate a Tikhonov-type regularization term. Our numerical results demonstrate that EKI is an efficient method for training neural ODEs in system identification and optimal control problems, with runtime and quality of solutions that are competitive with commonly used gradient-based optimizers.

1. Introduction

Already in 1988, two years after Rumelhart et al. (1986) have proposed backpropagation as a training method for multilayer perceptrons, Singhal & Wu (1988) used an extended Kalman filter (EKF) to train such neural networks. In a follow-up study, Singhal & Wu (1989) noted that "the Kalman algorithm converges in fewer iterations than backpropagation and obtains solutions with fewer hidden nodes in the network." While a more detailed comparison of backpropagation and EKFs showed that the latter may be associated with a substantially higher computational cost (Ruck et al., 1992), a more efficient variant of the EKF used by Singhal and Wu has been introduced by Puskorius & Feldkamp (1991) and been shown to converge towards desired solutions more rapidly than backpropagation in different learning tasks. Until 2019, when Kovachki & Stuart (2019) used a Monte Carlo approximation of the EKF, the so-called ensemble Kalman filter (EnKF) (Evensen, 1994), to train artificial neural networks (ANNs), most applications of Kalman filters to such training tasks were based on extended and unscented Kalman filters (Haykin & Haykin, 2001) that are known to suffer from computational and memory limitations. The EnKF avoids these limitations by propagating an ensemble of states that approximates the system state distribution and from which covariance matrix estimates are computed at every iteration. This method originated in the geosciences and has been successfully applied to many high-dimensional and non-linear data assimilation problems (Katzfuss et al., 2016). In addition to its application in data assimilation, the EnKF has been adapted to solve general inverse problems of the form

$$y = G(\theta) + \xi\,, \tag{1}$$

where one wishes to determine model parameters $\theta \in \mathcal{U}$ based on a known output variable $y \in \mathcal{Y}$ and model $G\colon \mathcal{U} \to \mathcal{Y}$ (Iglesias et al., 2013). The quantity $\xi \sim \mathcal{N}(0, \Gamma)$ denotes Gaussian noise with covariance Γ. The use of EnKF iterations to solve inverse problems has been dubbed "ensemble Kalman inversion" (EKI). A continuous-time limit of the discrete-time formulation of EKI has been derived by Schillings & Stuart (2017).

Ensemble Kalman inversion belongs to the class of sequential Monte Carlo methods that are used to solve inverse problems within a Bayesian framework (Idier, 2013; Dashti & Stuart, 2017). Kovachki & Stuart (2019) have cast supervised, semi-supervised, and online learning tasks into inverse problems of the form (1) and solved them using EKI. Unlike backpropagation, EKI is a gradient-free optimization method that only requires one to evaluate ANNs in forward passes. It can thus be easily parallelized.

Another variant of the EnKF has been used by Haber et al. (2018) to solve non-linear regression and image classification problems and an EKI-based sparse learning method has been proposed by Schneider et al. (2022) to use time-averaged statistics for data-driven discovery of differential equations. Ensemble Kalman methods have also been combined with auto-differentation approaches for parameter

[1]Department of Computational Science and Philosophy, Frankfurt School of Finance and Management, Frankfurt am Main, Germany. Correspondence to: Lucas Böttcher <l.boettcher@fs.de>.

Workshop on New Frontiers in Learning, Control, and Dynamical Systems at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023. Copyright 2023 by the author(s).

inference in dynamical systems and neural network models of partially or fully unknown dynamics (Chen et al., 2022).

In this work, we study the ability of EKI to efficiently train neural ordinary differential equations (neural ODEs) in system identification and control tasks. Neural ODEs have received renewed interest in recent years due to their diverse applications in dynamical systems identification, timeseries modeling (Wang & Lin, 1998; Chen et al., 2018) and optimal control (Asikis et al., 2022; Böttcher et al., 2022; Böttcher & Asikis, 2022; Cuchiero et al., 2020). Inverse problems (1) can be generalized to describe optimal control problems that involve an additional Tikhonov-type regularization term (Clason & Kaltenbacher, 2020). Complementing earlier work on Tikhonov EKI (Chada et al., 2020), we incorporate a regularization term in equation (1) to solve optimal control problems with EKI and neural ODEs.

2. Contributions

The main contributions of our work are as follows:

- Combining EKI with neural ODEs for system identification tasks.
- Formulating optimal control problems in terms of an inverse problem (1) that accounts for a control-energy regularization term.
- Formulating EKI updates for solving optimal control problems with neural ODEs.
- Comparing gradient-based and EKI-based optimization of neural ODEs in system identification and control tasks.

Our source codes are publicly available at (Böttcher, 2023).

3. Ensemble Kalman inversion

Ensemble Kalman inversion is a gradient-free optimization method that can be used to solve inverse problems (1) in an iterative manner. Because different neural-network optimization problems can be cast in the form (1) (Kovachki & Stuart, 2019), we will use EKI in this work to determine the optimal neural ODE parameters $\theta^* \in \mathcal{U}$ that minimize a given loss function. In Sections 4 and 5, we will reframe neural ODE-based system identification and optimal control problems as inverse problems and demonstrate how they can be solved using EKI. We compare the ability of EKI to efficiently identify solutions that are close to the desired optimum with algorithms that rely on backpropagation through time (BPTT) (Williams & Peng, 1990; Werbos, 1990; Feldkamp & Puskorius, 1993).

Based on a discrete-time formulation of the EnKF for inverse problems (Iglesias et al., 2013), a corresponding continuous-time formulation has been derived by Schillings & Stuart (2017). As a starting point, we consider an ensemble $\{\theta^{(j)}\}_{j=1}^J \subset \mathcal{U}$ of neural-network parameters. In accordance with Schillings & Stuart (2017), the continuous-time evolution of the ensemble $\{\theta^{(j)}\}_{j=1}^J$ is described by

$$\dot{\theta}^{(j)} = -C^{\theta G}(\theta)\Gamma^{-1}\left(G(\theta^{(j)}) - y\right), \tag{2}$$

$$\theta^{(j)}(0) = \theta_0^{(j)}, \tag{3}$$

where the empirical cross-covariance matrix is

$$C^{\theta G}(\theta) = \frac{1}{J}\sum_{j=1}^{J}\left(\theta^{(j)} - \bar{\theta}\right) \otimes \left(G(\theta^{(j)}) - \bar{G}\right). \tag{4}$$

The ensemble means $\bar{\theta}$ and $\bar{G}$ are given by

$$\bar{\theta} = \frac{1}{J}\sum_{j=1}^{J}\theta^{(j)} \quad \text{and} \quad \bar{G} = \frac{1}{J}\sum_{j=1}^{J}G(\theta^{(j)}), \tag{5}$$

respectively.

For linear inverse problems where $G(\theta) = A\theta$, it has been shown by Schillings & Stuart (2017) that Eq. (2) can be rewritten as

$$\dot{\theta}^{(j)}(t) = -C(\theta)\nabla_\theta \Phi(\theta; y), \tag{6}$$

where

$$\Phi(\theta; y) = \frac{1}{2}\|y - A\theta\|_\Gamma^2 \equiv \frac{1}{2}\|\Gamma^{-1/2}(y - A\theta)\|_{\mathcal{Y}}^2 \tag{7}$$

and

$$C(\theta) = \frac{1}{J}\sum_{j=1}^{J}\left(\theta^{(j)} - \bar{\theta}\right) \otimes \left(\theta^{(j)} - \bar{\theta}\right). \tag{8}$$

For any symmetric, positive-definite operator $C\colon \mathcal{H} \to \mathcal{H}$, we use the notations $\|\cdot\|_C = \|C^{-1/2}\|_{\mathcal{H}}$ and $\langle\cdot,\cdot\rangle_C = \langle C^{-1/2}\cdot, C^{-1/2}\cdot\rangle_{\mathcal{H}}$ to indicate adapted versions of the norm $\|\cdot\|_{\mathcal{H}}$ and inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ associated with a Hilbert space $\mathcal{H}$.

Equation (6) shows that $\theta^{(j)} \to \theta^*$ in the limit $t \to \infty$ where θ^* minimizes $\Phi(\theta; y)$ within the subspace $\operatorname{span}\{\theta_0^{(j)} - \bar{\theta}_0\}_{j=1}^J$ of $\mathcal{U}$. Here, $\bar{\theta}_0$ is the mean of the initial ensemble $\{\theta_0^{(j)}\}_{j=1}^J$.

4. Learning dynamical systems

4.1. Neural ODEs

A neural ODE parameterizes the vector field $f\colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ of a dynamical system

$$\dot{x}(t) = f(x(t), t), \tag{9}$$

where $x(t) \in \mathbb{R}^n$ is the system state at time t. We use $x(0) = x_0 \in \mathbb{R}^n$ to denote the corresponding initial condition.

Specifically, a neural ODE is an artificial neural network $f_\theta(x(t), t)$ with parameters $\theta \in \mathbb{R}^N$ that is used to represent the right-hand side of Eq. (9). When numerically integrating the dynamical system (9) with vector field $f_\theta(x(t), t)$, we say that the underlying artificial neural network becomes *time-unfolded*. For example, considering a simple forward Euler scheme with time step Δt, we have

$$x_{k+1} = x_k + \Delta t f_\theta(x_k, t_k)\,, \tag{10}$$

where $x_k \equiv x(t_k)$ and $t_k = k\Delta t$ for some positive integer k. Equation (10) shows that integrating the dynamical system (9) produces a residual neural network (He et al., 2016) that satisfies

$$f_\theta(x_k, t_k) = f_\theta(x_{k-1} + \Delta t f_\theta(x_{k-1}, t_{k-1}), t_k)\,. \tag{11}$$

Given observations $\{\hat{x}(t_\ell)\}_{\ell \in \{1,\dots,M\}}$ at times $t_\ell \in [0, T]$, our goal is to learn $f_\theta(x(t), t)$. To do so, we use the mean squared error

$$\mathrm{MSE}(\theta) = \frac{1}{M}\sum_{\ell=1}^{M} \left[\hat{x}(t_\ell) - x(t_\ell; \theta)\right]^2 \tag{12}$$

as a loss function and train a neural ODE with a suitable optimizer that minimizes Eq. (12).

4.2. Numerical experiments

To study the ability of neural ODEs that are trained with EKI to efficiently learn a dynamical system based on given observation data, we first consider the dynamical system

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix} = \begin{pmatrix}-0.05 & 1\\ -1 & -0.05\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} \tag{13}$$

with initial condition $(x_1(0), x_2(0))^\top = (1, 0)^\top$. The solution of this initial value problem is

$$\begin{pmatrix}x_1(t)\\ x_2(t)\end{pmatrix} = \begin{pmatrix}e^{-t/20}\cos(t)\\ -e^{-t/20}\sin(t)\end{pmatrix}. \tag{14}$$

To train a neural ODE $f_\theta(x(t), t)$ to represent the vector field associated with Eq. (13), we use 100 reference points from a discretized solution of the initial value problem as training data. The discretized solution consists of 500 time points $t_\ell \in [0, 40]$ and the set of 100 training data points consists of 10 subsets of 10 points associated with consecutive time steps [see red dots in Fig. 1(a)]. The first points in each subset are selected uniformly at random without replacement from the set of 500 points.

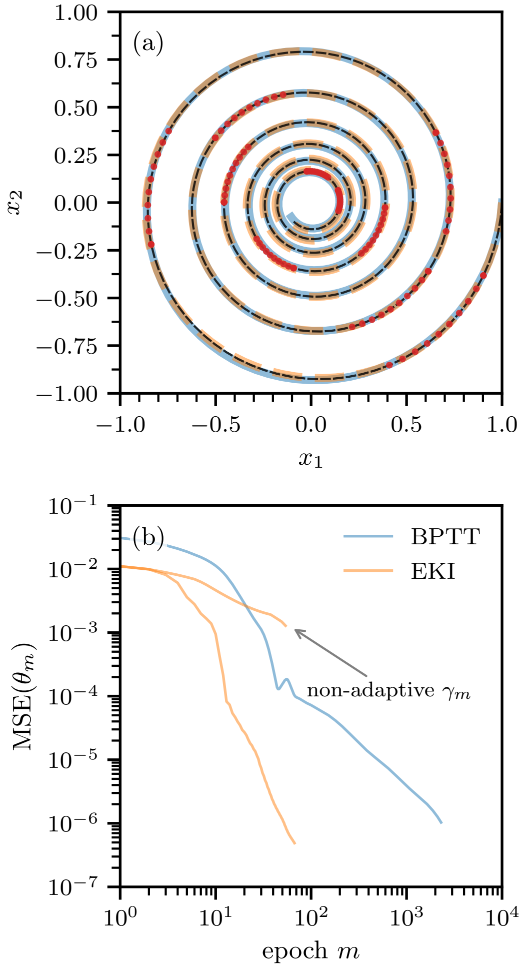

Figure 1. Learning a dynamical system from observation data using BPTT and EKI. (a) We employ a neural ODE to learn the dynamical system (13). Solid blue and orange lines correspond to solutions that are obtained with BPTT and EKI, respectively. The dashed black line indicates the solution of the original dynamical system for $(x_1(0), x_2(0))^\top = (1, 0)^\top$. Red dots represent observation data that is used to train the neural ODE. (b) Mean squared error (MSE) associated with BPTT (solid blue line) and EKI (solid orange line) as a function of the number of training epochs m [see Eq. (12)]. The neural ODE consists of one hidden layer with 10 tanh neurons. Its total number of parameters is 52. In the gradient-based optimization, we used the Adam optimizer and set the learning rate to $\eta = 0.01$. The EKI optimizer has 22 ensemble members and an exponential scheduler to adapt γ_m [see Eq. (15)]. For EKI, we also show the MSE evolution if no scheduler is used for γ_m. The EKI MSE is based on the minimum MSE in the whole ensemble. We stopped both training algorithms when their runtime exceeded one minute.

We represent $f_\theta(x(t), t)$ using a neural network with one hidden layer and 10 tanh neurons. The total number of parameters is 52. If not stated otherwise, weights and biases

	EKI	SGD ($\eta = 0.01$)	SGD ($\eta = 0.1$)	Adam ($\eta = 0.01$)	Adam ($\eta = 0.1$)
training error	$\mathbf{4.89 \times 10^{-7}}$	2.47×10^{-3}	5.02×10^{-5}	1.03×10^{-6}	5.79×10^{-7}
test error	$\mathbf{9.11 \times 10^{-4}}$	2.48×10^{-1}	5.47×10^{-3}	1.12×10^{-3}	7.86×10^{-3}

Table 1. Training and test errors associated with different optimizers. Training has been stopped after the training time exceeded 60 seconds. SGD and Adam results are based on the same initial neural-network parameters. For EKI, the initial ensemble size is 22 and the diagonal elements of the covariance matrix are updated according to the exponential scheduler (15).

	EKI	SGD ($\eta = 0.01$)	SGD ($\eta = 0.1$)	Adam ($\eta = 0.01$)	Adam ($\eta = 0.1$)
training error	4.00×10^{-7}	5.62×10^{-4}	4.57×10^{-5}	2.02×10^{-7}	$\mathbf{1.02 \times 10^{-7}}$
test error	$\mathbf{1.38 \times 10^{-5}}$	8.38×10^{-2}	1.19×10^{-3}	1.44×10^{-5}	1.97×10^{-5}

Table 2. Training and test errors associated with different optimizers. Training has been stopped after the training time exceeded 60 seconds. SGD and Adam results are based on the same initial neural-network parameters. For EKI, the initial ensemble size is 22 and the diagonal elements of the covariance matrix are updated according to the exponential scheduler (15).

are initialized from $\mathcal{U}(-\sqrt{d}, \sqrt{d})$, where d is the inverse of the number of input features. In all examples, we integrate neural ODEs using a Dormand–Prince method (Dormand & Prince, 1980; Hairer et al., 1993). For EKI, we use an ensemble size of $J = 22$ and a diagonal covariance matrix with entries $(\Gamma_m)_{ij} = \gamma_m \delta_{ij}$, where m is the current training epoch and δ_{ij} is the Kronecker delta function (*i.e.*, $\delta_{ij} = 1$ if $i = j$ and 0 otherwise). We initially set $\gamma_0 = 0.9$ and then reduce its value every two iterations using an exponential scheduler. That is, every two iterations, we set

$$\gamma_m = \gamma_0 e^{-\alpha m}\,, \tag{15}$$

where $\alpha > 0$ modulates the decrease of γ_0. We use an exponential decay in γ_m to map small differences between $G(\theta_m^{(j)})$ and y in Eq. (2) to noticeable updates in $\theta_m^{(j)}$. In our simulations, we set $\alpha = 0.35$.

Figure 1(b) shows that this adaptive EKI method is able to train the described neural ODE to represent the dynamical system (13) with initial condition $(x_1(0), x_2(0))^\top = (1, 0)^\top$. Without exponential scheduler, the MSE decreases more slowly as a function of training epochs. To compare the EKI-based solution with a solution that uses a gradient-based optimizer, we train the same neural ODE using the Adam optimizer (Kingma & Ba, 2015), an adaptive gradient-descent method, with a learning rate of $\eta = 0.01$.[1]

For an appropriate comparison of the two optimization methods, we stop the neural ODE training when the training time exceeds one minute on a single core of an Intel® Core™ i7-10510U CPU @ 1.80GHz × 8. The total numbers of training epochs of EKI and Adam are 66 (*i.e.*, ca. 1 per second) and 2277 (*i.e.*, ca. 38 per second), respectively. The training and test errors of EKI are 4.89×10^{-7} and 9.11×10^{-4}, respectively. The training error of Adam is 1.03×10^{-6}, about twice as large as that of EKI, while the test error of Adam is 1.12×10^{-3} and thus almost equivalent to that of EKI. We also performed numerical experiments for an additional learning rate ($\eta = 0.1$) and for stochastic gradient descent (SGD). The results are summarized in Table 1. While Adam can achieve a smaller training error for the learning rate $\eta = 0.1$, the corresponding test error is substantially larger than for the training with $\eta = 0.01$. The performance of SGD is inferior to Adam for the two tested learning rates. Overall, EKI can deliver a competitive performance against the adaptive learning method Adam.

As an example of a non-linear dynamical system, we consider the ODE

$$\ddot{x} = -\omega \sin(x)\,, \tag{16}$$

which describes the motion of a simple pendulum with natural frequency ω. In our numerical experiments, we set $\omega = 1$. As initial condition, we use $x(0) = \pi/4$ and $\dot{x}(0) = 0$. As in the previous example, the neural network parameterizing $f_\theta(x(t), t)$ has 10 $\tanh$ neurons.

To train a neural ODE $f_\theta(x(t), t)$ to represent the vector field associated with Eq. (16), we use 100 reference points from a discretized solution of the initial value problem as training data. The discretized solution consists of 200 time points $t_\ell \in [0, 20]$ and the set of 100 training data points consists of 10 subsets of 10 points associated with consec-

[1] In all numerical experiments that use the Adam optimizer, we set $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Here, β_1, β_2 are coefficients that are used to compute running averages of the gradient and its square. The parameter ϵ is used in the denominator of gradient updates to improve numerical stability.

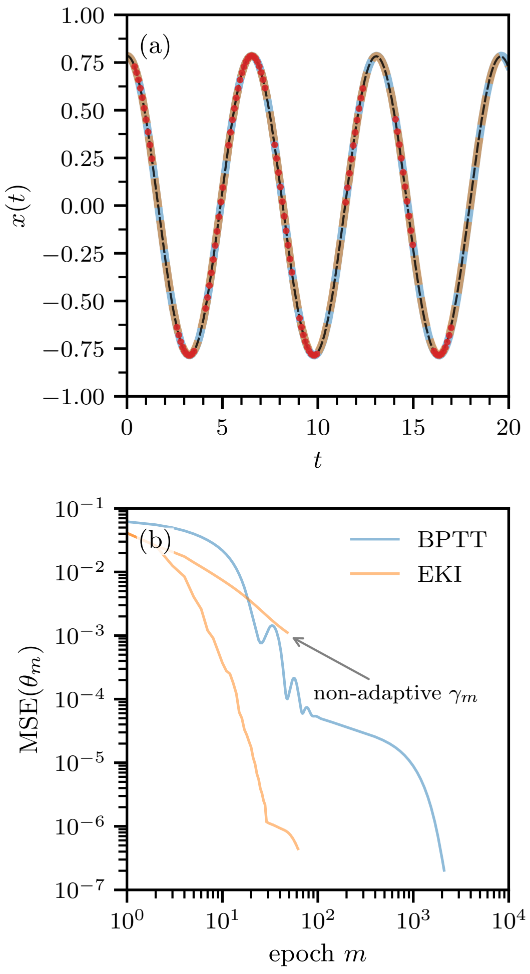

Figure 2. Learning a dynamical system from observation data using BPTT and EKI. (a) We employ a neural ODE to learn the dynamical system (16). Solid blue and orange lines correspond to solutions that are obtained with BPTT and EKI, respectively. The dashed black line indicates the solution of the original dynamical system for $x(0) = \pi/4$ and $\dot{x}(0) = 0$. Red dots represent observation data that is used to train the neural ODE. (b) Mean squared error (MSE) associated with BPTT (solid blue line) and EKI (solid orange line) as a function of the number of training epochs m [see Eq. (12)]. The neural ODE consists of one hidden layer with 10 tanh neurons. Its total number of parameters is 52. In the gradient-based optimization, we use the Adam optimizer and set the learning rate to $\eta = 0.01$. The EKI optimizer has 22 ensemble members and an exponential scheduler to adapt γ_m [see Eq. (15)]. For EKI, the MSE is based on the minimum MSE in the whole ensemble. We stopped both training algorithms when their runtime exceeded one minute.

utive time steps [see red dots in Fig. 2(a)]. The first points in each subset are selected uniformly at random without replacement from the set of 200 points.

We use $J = 22$ ensemble members and an exponential scheduler (15) with $\gamma_0 = 1.4$–2.6 and $\alpha = 0.4$. The adaptive γ_m again helps the EKI-based optimizer reach small loss values [see Fig. 2(b)]. We again stop training when the training time exceeds one minute. Figure 2(b) shows that the training loss associated with EKI is slightly larger than that associated with Adam (4.00×10^{-7} vs. 2.02×10^{-7}). We also performed additional numerical experiments using the Adam optimizer with a learning rate of $\eta = 0.1$ and SGD with $\eta = 0.1, 0.01$. The training and test errors are summarized in Table 2. The smallest test error has been achieved with EKI.

5. Optimal control

We will now focus on the gradient-free training of neural ODE controllers. As a starting point, we consider the boundary value problem

$$\dot{x} = f(x(t), u(t), t)\,, \quad x(0) = x_0\,, \quad x(T) = x^*\,, \quad (17)$$

where the vector field $f \colon \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^n$ describes the evolution of the system state $x(t) \in \mathbb{R}^n$ subject to a control function $u(t) \in \mathbb{R}^m$.[2]

In optimal control, one wishes to identify a Lebesgue-measurable control function $u(t)$ that satisfies the constraint (17) and minimizes the functional

$$\mathcal{J} = \phi(x(T), T) + \int_0^T L(x(t), u(t), t)\,\mathrm{d}t\,, \qquad (18)$$

where $\phi \colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ and $L \colon \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}$ (Speyer & Jacobson, 2010; Lewis et al., 2012). That is, one wishes to minimize $\mathcal{J}$ and steer the dynamical system as defined in Eq. (17) from its initial state $x_0 \in \mathbb{R}^n$ to a desired target state $x^* \in \mathbb{R}^n$ in finite time T. If we were to impose additional constraints on the control function $u(t)$, it has to be chosen from a corresponding set of admissible controls (Wang & Xu, 2018).

In this work, we focus on the special case where $\phi(x(T), T) \equiv 0$ and $L(x(t), u(t), t) \equiv L(u(t)) = \|u(t)\|_2^2$. The condition $\phi(x(T), T) \equiv 0$ means that the endpoint cost is zero. Because $L(u(t)) = \|u(t)\|_2^2$, the running (or integrated) cost is positive for a non-zero control signal. The corresponding cost function is given by the control energy

$$E_T[u] = \int_0^T \|u(t)\|_2^2\,\mathrm{d}t \qquad (19)$$

The outlined optimal control problem aims at finding the control signal $u^*(t)$ that is associated with the smallest

[2]To adhere to the standard notation in control theory, we use m to denote the dimension of the vector space associated with the control signal $u(t)$. Therefore, it is important to distinguish m from the number of epochs.

control energy and satisfies the constraint (17). That is,

$$u^*(t) = \arg\min_{u(t)} E_T[u]\,. \tag{20}$$

subject to the constraint (17).

Using Pontryagin's maximum principle (Pontryagin, 1987), a necessary condition for optimal control, we can find the control that satisfies Eqs. (17) and (20) by minimizing the control Hamiltonian

$$H(x(t), u(t), \lambda(t), t) = \lambda(t)^\top f(x(t), u(t), t) + \|u(t)\|_2^2\,, \tag{21}$$

at every time point t. The time-dependent components of the Lagrange multiplier vector $\lambda(t) \in \mathbb{R}^n$ are called the adjoint (or costate) variables of the system (Speyer & Jacobson, 2010; Bertsekas, 2012).

5.1. Neural ODE controllers

While Pontryagin's maximum principle provides a necessary condition for optimal control (Pontryagin, 1987), the Hamilton–Jacobi–Bellman equation offers both necessary and sufficient conditions for optimality (Zhou, 1990; Bellman & Dreyfus, 1962). Nonlinear optimal control problems are usually solved through indirect and direct numerical methods, and recently transformation methods have been proposed to convert non-linear control problems into linear ones (Kaiser et al., 2021). Indirect optimal control solvers involve different kinds of shooting methods (Oberle & Grimm, 2001) that use the maximum principle and a control Hamiltonian to construct a system of equations that describe the evolution of state and adjoint variables. On the other hand, direct methods involve parameterizing state and control functions and solving the resulting optimization problem. Possible function parameterizations include piecewise constant functions and other suitable basis functions (Bock & Plitt, 1984). Over the past two decades, pseudospectral methods have been a successful approach to solving nonlinear optimal control problems, with applications in aerospace engineering (Gong et al., 2006). However, it has been shown that certain pseudospectral methods are incapable of solving standard benchmark control problems (Fahroo & Ross, 2008). Here, we parameterize and learn control functions $u(t)$ using neural ODEs $u_\theta(t)$ with parameters $\theta \in \mathbb{R}^N$ (Asikis et al., 2022; Böttcher et al., 2022; Böttcher & Asikis, 2022). The boundary value problem (17) hence becomes

$$\dot{x} = f(x(t), u_\theta(t), t)\,, \quad x(0) = x_0\,, \quad x(T) = x^*\,. \tag{22}$$

Before applying EKI to optimal control problems, we have to adapt both the underlying inverse problem and ensemble iterations. Recall that the standard EKI algorithm as summarized in Section 3 aims at finding a solution to the inverse problem (1) by minimizing the functional

$$\Phi(\theta; y) = \frac{1}{2}\|y - G(\theta)\|_\Gamma^2 \equiv \frac{1}{2}\|\Gamma^{-1/2}(y - G(\theta))\|_{\mathcal{Y}}^2\,. \tag{23}$$

To account for the additional regularization term in optimal control problems, we extend the inverse problem (1) using

$$z = F(\theta) + \xi\,, \tag{24}$$

where $F\colon \mathcal{U} \times \mathcal{U} \to \mathcal{Z}$, $z = (y, 0)^\top \in \mathcal{Z}$, $\xi = (\xi_1, \xi_2)^\top$, $\xi \sim \mathcal{N}(0, \Sigma)$ and

$$\Sigma = \begin{pmatrix} \Gamma & 0 \\ 0 & \mu^{-1}\Gamma' \end{pmatrix}\,. \tag{25}$$

The first element of the function $F(\theta) = (G(\theta), H(\theta))^\top$, $G(\theta)$, describes the evolution of the controlled system state $x(t)$ associated with $f(x(t), u_\theta(t), t)$ while the second element, $H(\theta)$, accounts for the control energy term $E_T[u_\theta]$.

As in the unregularized EKI method that we described in Section 3, the evolution of the ensemble $\{\theta^{(j)}\}_{j=1}^J$ associated with the inverse problem (24) is given by

$$\dot{\theta}^{(j)} = -B^{\theta F}(\theta)\Sigma^{-1}\left(F(\theta^{(j)}) - z\right)\,, \tag{26}$$

$$\theta^{(j)}(0) = \theta_0^{(j)}\,, \tag{27}$$

where the empirical cross-covariance matrix, $B^{\theta F}(\theta)$, and the ensemble mean, $\bar{F}$, are given by

$$B^{\theta F}(\theta) = \frac{1}{J}\sum_{j=1}^J \left(\theta^{(j)} - \bar{\theta}\right) \otimes \left(F(\theta^{(j)}) - \bar{F}\right)\,. \tag{28}$$

and

$$\bar{F} = \frac{1}{J}\sum_{j=1}^J F(\theta^{(j)})\,, \tag{29}$$

respectively. The associated loss function is

$$\tilde{\Phi}(\theta; z) = \frac{1}{2}\|\Sigma^{-1/2}(z - F(\theta))\|_{\mathcal{Z}}^2\,. \tag{30}$$

We now identify y in $z = (y, 0)^\top$ with the target state x^*, and we set $G(\theta) = x(T;\theta)$ and $H(\theta) = E_T[u_\theta]^{1/2}$. This yields the loss function

$$\tilde{\Phi}(\theta; x^*) = \frac{1}{2}\|x(T;\theta) - x^*\|_\Gamma^2 + \frac{\mu}{2\Gamma'}E_T[u_\theta]\,. \tag{31}$$

The above formulation of optimal control problems is similar in its mathematical structure to Tikhonov EKI that has been introduced by Chada et al. (2020) to regularize the parameter vector θ while solving inverse problems using EKI.

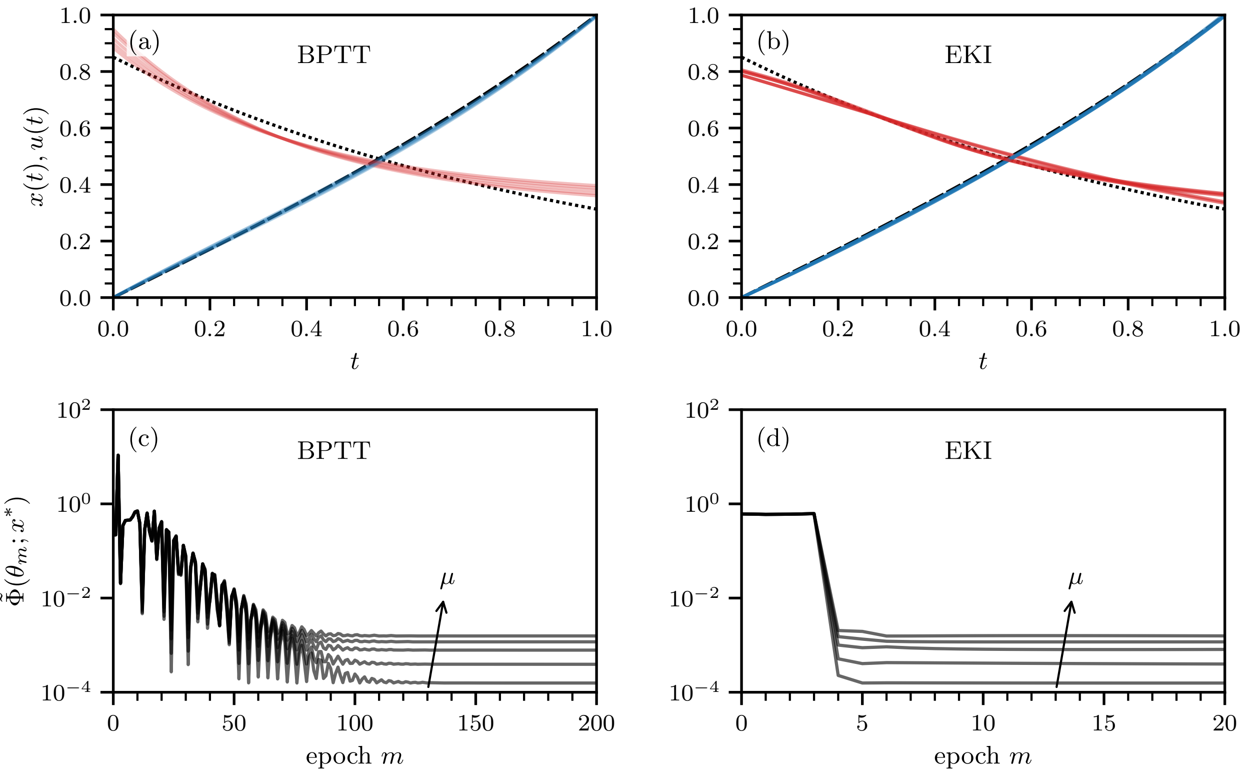

Figure 3. Controlling linear dynamics with neural ODEs. The neural ODE controller $u_\theta(t)$ has been trained with BPTT and EKI in panels (a,c) and (b,d), respectively. (a,b) Evolution of the system state $x(t)$ and control function $u(t)$. Dashed and dotted lines show solutions $x^*(t) = \sinh(t)/\sinh(1)$ and $u^*(t) = \exp(-t)/\sinh(1)$ associated with the optimal control problem (32) and (33). We set $x_0 = 0$ and $T = a = b = x^* = 1$ in all simulations. Numerical solutions in panels (a,b) are indicated by colored lines. Different lines correspond to different values of the control energy regularization parameter $\mu \in \{0.001, 0.0025, 0.005, 0.0075, 0.01\}$. The neural ODE controller $u_\theta(t)$ consists of three hidden layers with five exponential linear units (ELU) neurons each. Its total number of parameters is 106. In the gradient-based optimization, we used the Adam optimizer and set the learning rate to $\eta = 0.175$. (c,d) The loss function $\tilde{\Phi}(\theta_m; x^*)$ as a function of the number of training epochs m. For EKI, we set $\Gamma = 0.3, \Gamma' = 0.01$ and show the minimum loss in the whole ensemble. After three iterations, we set $\Gamma = 0.15$. For BPTT, ensemble variances are not relevant and we thus optimize the loss function for which $\Gamma = \Gamma' = 1$.

5.2. Numerical experiments

We now examine if neural ODE controllers that are trained with EKI are able to learn effective control functions. To do so, we consider the optimal control problem that aims at identifying

$$u^*(t) = \arg\min_{u(t)} E_T[u] \tag{32}$$

subject to

$$\dot{x} = ax + bu\,, \quad x(0) = x_0\,, \quad x(T) = x^*\,. \tag{33}$$

The mathematical structure of this optimal control problem is similar to that of control problems encountered in regulating temperature within a room (Lewis et al., 2012).

We use Pontryagin's maximum principle to derive $u^*(t)$, which we will assume to be the optimal control signal associated with Eqs. (32) and (33). This calculation yields

$$u^*(t) = \frac{a e^{-at}}{b \sinh(aT)} \left(x^* - x_0 e^{aT}\right)\,. \tag{34}$$

For $a > 0$, the magnitude of $u^*(t)$ decays exponentially with t because the value of the system state of the uncontrolled dynamics $\dot{x} = ax$ can be influenced more effectively for small values of t than for large ones. The opposite holds for $a < 0$.

The system state $x(t)$ under the influence of $u^*(t)$ is

$$x^*(t) = x_0 e^{at} + \frac{\sinh(at)}{\sinh(aT)} \left(x^* - x_0 e^{aT}\right)\,. \tag{35}$$

Finally, the control energy associated with the optimal control signal (34) is

$$E_T[u^*] = \int_0^T u^*(t)^2 \, \mathrm{d}t = \frac{a(1-e^{-2aT})(x^* - x_0 e^{aT})^2}{2b^2 \sinh^2(aT)} \,. \tag{36}$$

In the following numerical experiments, we consider the optimal control problem (32) and (33) for which $x_0 = 0$ and $T = a = b = x^* = 1$. The corresponding optimal control signal, system state, and control energy are $u^*(t) = \exp(-t)/\sinh(1)$, $x^*(t) = \sinh(t)/\sinh(1)$, and $E_T[u^*] = 2/(e^2-1) \approx 0.313$. The neural ODE controller $u_\theta(t)$ that we employ consists of three hidden layers with five exponential linear units (ELU) neurons each. Its total number of parameters is 106.

In the gradient-based optimization (*i.e.*, for BPTT), we do not use an ensemble of ANN parameters and thus minimize the loss function (31) for which $\Gamma = \Gamma' = 1$. We use the Adam optimizer and set the learning rate to $\eta = 0.175$. We evaluated different learning rates within the range of 0.1 to 0.2 and chose the learning rate that produced solutions that closely resembled the optimal ones.

For EKI, the initial number of ensemble members is $J = 2$ and we set $\Gamma = 0.3, \Gamma' = 0.01$. After three iterations, we add 20 new ensemble members and set $\Gamma = 0.15$. We observe that a small number of ensemble members yields a good initial performance. Adding new ensemble members helps EKI learn good parameters as the optimization progresses. These observations on expanding the initial ensemble are consistent with related work (Kovachki & Stuart, 2019).

To investigate how the control energy regularization parameter μ [see Eq. (31)] affects the learned system state and control function, we vary μ within the range of 0.001 to 0.01 in the loss function $\tilde{\Phi}(\theta; x^*)$. The dashed and dotted black lines in Fig. 3(a,b) represent the optimal solutions $x^*(t)$ and $u^*(t)$. Solid red lines show the control solutions $u_\theta(t)$ learned by neural ODE controllers. Solid blue lines indicate the corresponding system state evolution. As we show in Fig. 3(a,b), variations in μ produce variations in the learned solutions of the optimal control problem. Furthermore, the shown simulation results suggest that both the gradient-based optimizer and EKI are capable of training neural ODE controllers to learn control functions that closely approximate the optimal solution.

We calculated the MSE between the learned and optimal control solution to compare the quality of the solutions learned using BPTT and EKI. We find the the MSE is about 1–1.4×10^{-3} (BPTT) and 0.4–0.6×10^{-3} (EKI). The MSE associated with EKI is thus about 50% of that associated with the BPTT-based solution.

Figure 3(c,d) shows the loss function $\tilde{\Phi}(\theta_m; x^*)$ associated with BPTT and EKI as a function of the number of training epochs m. Different loss trajectories correspond to different control energy regularization parameters μ. After approximately 90-100 epochs, Adam-based optimization stabilizes at certain loss values, although it may achieve small loss values earlier in the training process. In contrast, EKI is able to achieve similar results within only 4-5 epochs. In the studied control example, EKI and Adam-based optimization achieve about 1 and 10 optimization steps per second.

Our results show that EKI can account for a control energy regularization term and be employed to solve an optimal control problem in a gradient-free manner.

6. Discussion and conclusion

We studied the ability of ensemble Kalman inversion (EKI) as an alternative method to backpropagation for training neural ODEs in system identification and optimal control tasks. Ensemble Kalman inversion is a gradient-free optimization method that solves general inverse problems within a Bayesian framework. It only requires one to evaluate artificial neural networks in forward passes, making backward passes unnecessary.

After providing an overview of the basic formalism of EKI, we applied this method to system identification problems associated with linear and non-linear dynamics. Our results showed that EKI performs well with respect to gradient-based optimization methods such as SGD and Adam. We also applied EKI to optimal control problems that involve an additional control energy regularization term to keep the integrated square norm of the control signal small [see Eqs. (19) and (31)]. We reformulated EKI iterations to account for such a regularization term and applied this adapted method to an optimal control problem with an underlying linear vector field. The EKI approach that we use for solving optimal control problems is similar in its mathematical structure to Tikhonov EKI (Chada et al., 2020).

In summary, our results suggest that EKI can serve as an effective alternative to gradient-based optimization techniques in training neural ODEs for system identification problems. Additionally, we extended the use of EKI to optimal control problems, providing a new perspective on solving inverse problems that arise in control theory using EKI.

There are several promising avenues for future research. For instance, it would be interesting to examine EKI's effectiveness in higher-dimensional system identification and control tasks, and contrast its performance with other gradient-based techniques that have been utilized in training neural ODEs (Chen et al., 2018; Ainsworth et al., 2021). Moreover, future research may study the geometric properties of the optima found by gradient-based methods and EKI, providing insights into the similarities and differences between these optimization approaches. Such an analysis can broaden our understanding of optimization landscapes (Böttcher & Wheeler, 2022) and help guide the development of optimization algorithms that are both robust and efficient. Additionally, future work may focus on EKI's capacity to train neural ODEs using noisy observation data or other types of observation data that make the use of gradient-based methods more challenging. Finally, investigating the use of EKI in training physics-informed neural networks and their corresponding controllers (Mowlavi & Nabi, 2022) would be a valuable area of investigation as well.

References

Ainsworth, S. K., Lowrey, K., Thickstun, J., Harchaoui, Z., and Srinivasa, S. S. Faster policy learning with continuous-time gradients. In Jadbabaie, A., Lygeros, J., Pappas, G. J., Parrilo, P. A., Recht, B., Tomlin, C. J., and Zeilinger, M. N. (eds.), *Proceedings of the 3rd Annual Conference on Learning for Dynamics and Control, L4DC 2021, 7-8 June 2021, Virtual Event, Switzerland*, volume 144 of *Proceedings of Machine Learning Research*, pp. 1054–1067. PMLR, 2021.

Asikis, T., Böttcher, L., and Antulov-Fantulin, N. Neural ordinary differential equation control of dynamics on graphs. *Physical Review Research*, 4(1):013221, 2022.

Bellman, R. E. and Dreyfus, S. E. *Applied Dynamic Programming*. Princeton University Press, Princeton, MA, 1962.

Bertsekas, D. *Dynamic Programming and Optimal Control, 4th edition*. Athena Scientific, St Belmont, MA, 2012.

Bock, H. G. and Plitt, K.-J. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984.

Böttcher, L. Gitlab repository. `https://gitlab.com/ComputationalScience/eki-neural-ode`, 2023.

Böttcher, L. and Asikis, T. Near-optimal control of dynamical systems with neural ordinary differential equations. *Machine Learning: Science and Technology*, 3(4):045004, 2022.

Böttcher, L. and Wheeler, G. Visualizing high-dimensional loss landscapes with Hessian directions. *arXiv preprint arXiv:2208.13219*, 2022.

Böttcher, L., Antulov-Fantulin, N., and Asikis, T. AI Pontryagin or how artificial neural networks learn to control dynamical systems. *Nature Communications*, 13(1):333, 2022.

Chada, N. K., Stuart, A. M., and Tong, X. T. Tikhonov regularization within ensemble Kalman inversion. *SIAM Journal on Numerical Analysis*, 58(2):1263–1294, 2020.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 6572–6583, 2018.

Chen, Y., Sanz-Alonso, D., and Willett, R. Autodifferentiable ensemble Kalman filters. *SIAM Journal on Mathematics of Data Science*, 4(2):801–833, 2022.

Clason, C. and Kaltenbacher, B. Optimal control and inverse problems. *Inverse Problems*, 36(6):060301, 2020.

Cuchiero, C., Larsson, M., and Teichmann, J. Deep neural networks, generic universal interpolation, and controlled ODEs. *SIAM Journal on Mathematics of Data Science*, 2 (3):901–919, 2020.

Dashti, M. and Stuart, A. M. *The Bayesian approach to inverse problems*, pp. 311–428. Springer International Publishing, Cham, 2017.

Dormand, J. R. and Prince, P. J. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980.

Evensen, G. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, 99(C5):10143–10162, 1994.

Fahroo, F. and Ross, I. M. Advances in pseudospectral methods for optimal control. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 7309, 2008.

Feldkamp, L. and Puskorius, G. Neural network control of an unstable process. In *Proceedings of 36th Midwest Symposium on Circuits and Systems, Detroit, MI, USA, August 16-18, 1993*, pp. 35–40. IEEE, 1993.

Gong, Q., Kang, W., and Ross, I. M. A pseudospectral method for the optimal control of constrained feedback linearizable systems. *IEEE Transactions on Automatic Control*, 51(7):1115–1129, 2006.

Haber, E., Lucka, F., and Ruthotto, L. Never look back-A modified EnKF method and its application to the training of neural networks without back propagation. *arXiv preprint arXiv:1805.08034*, 2018.

Hairer, E., Nørsett, S. P., and Wanner, G. *Solving Ordinary Differential Equations I, Nonstiff Problems*. Springer Berlin, Heidelberg, 1993.

Haykin, S. S. and Haykin, S. S. *Kalman filtering and neural networks*. John Wiley & Sons, Inc., Hoboken, NJ, 2001.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016.

Idier, J. *Bayesian approach to inverse problems*. John Wiley & Sons, Hoboken, NJ, 2013.

Iglesias, M. A., Law, K. J., and Stuart, A. M. Ensemble Kalman methods for inverse problems. *Inverse Problems*, 29(4):045001, 2013.

Kaiser, E., Kutz, J. N., and Brunton, S. L. Data-driven discovery of Koopman eigenfunctions for control. *Machine Learning: Science and Technology*, 2(3):035023, 2021.

Katzfuss, M., Stroud, J. R., and Wikle, C. K. Understanding the ensemble Kalman filter. *The American Statistician*, 70(4):350–357, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Kovachki, N. B. and Stuart, A. M. Ensemble Kalman inversion: a derivative-free technique for machine learning tasks. *Inverse Problems*, 35(9):095005, 2019.

Lewis, F. L., Vrabie, D., and Syrmos, V. L. *Optimal Control*. John Wiley & Sons, Hoboken, NJ, 2012.

Mowlavi, S. and Nabi, S. Optimal control of PDEs using physics-informed neural networks. *Journal of Computational Physics*, pp. 111731, 2022.

Oberle, H. J. and Grimm, W. *BNDSCO: a program for the numerical solution of optimal control problems*. Institut für Angewandte Mathematik der Universität Hamburg, Germany, 2001.

Pontryagin, L. S. *Mathematical theory of optimal processes*. CRC Press, Boca Raton, FL, 1987.

Puskorius, G. V. and Feldkamp, L. A. Decoupled extended Kalman filter training of feedforward layered networks. In *IJCNN-91-Seattle International Joint Conference on Neural Networks, Seattle, WA, USA, July 08-12, 1991*, volume 1, pp. 771–777. IEEE, 1991.

Ruck, D. W., Rogers, S. K., Kabrisky, M., Maybeck, P. S., and Oxley, M. E. Comparative analysis of backpropagation and the extended Kalman filter for training multilayer perceptrons. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 14(06):686–691, 1992.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Schillings, C. and Stuart, A. M. Analysis of the ensemble Kalman filter for inverse problems. *SIAM Journal on Numerical Analysis*, 55(3):1264–1290, 2017.

Schneider, T., Stuart, A. M., and Wu, J.-L. Ensemble Kalman inversion for sparse learning of dynamical systems from time-averaged data. *Journal of Computational Physics*, 470:111559, 2022.

Singhal, S. and Wu, L. Training multilayer perceptrons with the extended Kalman algorithm. In Touretzky, D. (ed.), *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.

Singhal, S. and Wu, L. Training feed-forward networks with the extended Kalman algorithm. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '89, Glasgow, Scotland, May 23-26, 1989*, pp. 1187–1190. IEEE, 1989. doi: 10.1109/ICASSP.1989.266646.

Speyer, J. L. and Jacobson, D. H. *Primer on optimal control theory*. SIAM, Philadelphia, PA, 2010.

Wang, L. and Xu, Y. Admissible controls and controllable sets for a linear time-varying ordinary differential equation. *Mathematical Control and Related Fields*, 8(3&4): 1001, 2018.

Wang, Y.-J. and Lin, C.-T. Runge-Kutta neural network for identification of dynamical systems in high accuracy. *IEEE Transactions on Neural Networks*, 9(2):294–307, 1998.

Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

Williams, R. J. and Peng, J. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990.

Zhou, X. Maximum principle, dynamic programming, and their connection in deterministic control. *Journal of Optimization Theory and Applications*, 65(2):363–373, 1990.