Results Math (2024) 79:43
© 2023 The Author(s)
1422-6383/24/010001-21
published online December 8, 2023
https://doi.org/10.1007/s00025-023-02074-3

Results in Mathematics

A Functional Approach to Interpreting the Role of the Adjoint Equation in Machine Learning

Imre Fekete, András Molnár, and Péter L. Simon

Abstract. The connection between numerical methods for solving differential equations and machine learning has been revealed recently. Differential equations have been proposed as continuous analogues of deep neural networks, and then used in handling certain tasks, such as image recognition, where the training of a model includes learning the parameters of systems of ODEs from certain points along their trajectories. Treating this inverse problem of determining the parameters of a dynamical system that minimize the difference between data and trajectory by a gradient-based optimization method presents the solution of the adjoint equation as the continuous analogue of backpropagation that yields the appropriate gradients. The paper explores an abstract approach that can be used to construct a family of loss functions with the aim of fitting the solution of an initial value problem to a set of discrete or continuous measurements. It is shown, that an extension of the adjoint equation can be used to derive the gradient of the loss function as a continuous analogue of backpropagation in machine learning. Numerical evidence is presented that under reasonably controlled circumstances the gradients obtained this way can be used in a gradient descent to fit the solution of an initial value problem to a set of continuous noisy measurements, and a set of discrete noisy measurements that are recorded at uncertain times.

Mathematics Subject Classification. 90C52, 68Q32, 34A55.

Keywords. Continuous backpropagation, adjoint equation, parameter learning.

1. Introduction

Machine learning has been recently connected to the field of differential equations by observing that numerical time integrators resemble formulae used in residual neural networks [9,14]. The fast development of deep learning algorithms has led to the study of analogues of deep neural networks, and that of the discretization of continuous dynamical systems [7,15]. The continuous analogue of backpropagation in deep residual neural networks is the adjoint equation, also used in optimal control, plays a crucial role in connecting the two fields. This recently revealed relation has been exploited along the following two directions.

(1) Certain tasks, typically handled by deep neural networks, such as image recognition, are treated by using the continuous analogue, that is, by learning the parameters of a system of ODEs.
(2) Learning the parameters of a dynamical system from certain points of its trajectories by using the continuous analogue of backpropagation, that is, by applying the gradient method by computing the derivative of the loss function by solving the adjoint equation.

Our work presented in this paper is mostly related to direction (2), hence we will deal with the two directions in the introduction as follows.

- Literature overview corresponding to direction (1).
- Problem description of direction (2).
- Literature overview corresponding to direction (2).
- Novelties and structure of our paper.

The analogue between a deep residual neural network and the numerical scheme corresponding to the discretization of an ODE is presented, and the proposed method is applied in image classification in [7,14]. The idea is also extended to learning neural ODE for stiff systems [8]. A linear multi-step architecture (LM-architecture) is introduced in [9] as a generalization of ResNet, inspired by the linear multi-step method for solving ordinary differential equations. It is shown that it achieves higher accuracy in image recognition than ResNet and other previous neural ODE (NODE) architectures. The representative power of NODE, that is, the class of functions that these can represent is investigated in [6]. The authors illustrate the limitations of this approach with some simple examples, and show how the class of representable functions can be extended by extending the state space via augmented variables.

We now turn to direction (2), and formulate the problem of learning the parameters of a dynamical system. Let us assmue that we are given a set of time points $\mathcal{T} \subseteq [0,1]$, and a sample from a trajectory of a differential equation evaluated at these points. We note that the choice of the unit interval is merely an aesthetic one, which can be made without loss of generality. This is typically either the time dependence of a trajectory component $y : [0,1] \to \mathbb{R}$, or a time series $y(\tau_1), y(\tau_2), \dots, y(\tau_n)$ obtained from it. The goal is to find an initial value problem, the solution of which fits the given data.

More precisely, given a family of right hand sides parameterized by a k-dimensional parameter $\theta \in \mathbb{R}^k$, a d-dimensional initial condition $x_0 \subseteq \mathbb{R}^d$, and a 1-dimensional initial time $t_0 \in \mathbb{R}$, we are looking for the best initial time, initial condition, parameter triple (t_0, x_0, θ) in some search space $\mathcal{S} \subseteq \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^k$.

That is, given the function $f : \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}^d$, we consider the solution x of the problem

$$\begin{cases} \dot{x}(t) &= f\big(t, x(t); \theta\big), \qquad t_0 < t < t_0 + 1 \\ x(t_0) &= x_0, \end{cases} \tag{1}$$

and try to find the value of (t_0, x_0, θ), for which the distance of the functions $t \mapsto x(t)$ and $t \mapsto y(t - t_0)$ is minimal in some sense.

To this end, we employ a learning process, which first constructs a differentiable loss function $\mathcal{L} \colon \mathcal{S} \to \mathbb{R}$, then, given an initial guess for the triple (t_0, x_0, θ), applies a gradient-descent based iterative method to minimize it. Efficient calculation of the gradients used during the iteration is made possible by the continuous backpropagation process based on the adjoint equation [5].

As an illustrative example, the reader may have in mind the $d = 1$ dimensional case. Then two simple possible loss functions are the following. Given a discrete sample, we may let

$$\mathcal{L}(t_0, x_0, \theta) = \frac{1}{n} \sum_{j=1}^{n} \left(x_{(t_0, x_0, \theta)}(t_0 + \tau_j) - y(\tau_j) \right)^2, \tag{2a}$$

while given the trajectory itself, we may pick

$$\mathcal{L}(t_0, x_0, \theta) = \int_{[0,1]} \left(x_{(t_0, x_0, \theta)}(t_0 + \tau) - y(\tau) \right)^2 \, d\tau, \tag{2b}$$

where we use the subscript (t_0, x_0, θ) to emphasize the solution's dependence on these parameters.

The minimization of this loss function is also referred to as parameter identification problem or system identification and goes back to the seventies. The unknown parameters are to be determined in such a way that the difference of the model output and the measurements is minimized. The papers [4,12] review the results and methods known and used at that time. The minimization of the loss function can be carried out by using the gradient method, hence an effective method for computing the gradient is desired. This has led to the use of the adjoint equation developed in optimal control theory and also used in sensitivity analysis. The first derivation of the adjoint equation goes back to the idea of using Lagrangian formalism, where λ, the variable in the adjoint equation, plays a similar role as a Lagrange multiplier. It turned out that solving the adjoint equation backward in time yields the gradient of the loss function. The derivation of the adjoint equation and its relation to the gradient of the loss function is carried out and shown in a non-precise way in several papers. In most of them, a formal linearization leads to the adjoint equation after neglecting higher order terms. In [11], the formal derivation using the Lagrange multiplier analogue can be found and a specific application to stereo tomography in geophysics is detailed. A biological application is shown in [13], where the dynamical process is governed by a parabolic partial differential equation that is approximated by an ODE. The loss function is defined as the integral of the norm of the difference between the output and the observation. The gradient of the loss function is derived formally by changing the parameter vector infinitesimally and neglecting higher order terms. The calculations are extended to implicit ODEs and to differential-algebraic systems in [3]. The adjoint method has also recently been related to system identification [1,10]. The loss function, in those papers, is defined as the sum of squares of the differences $x(\tau_j) - y(\tau_j)$ (in our terms), the adjoint equation is derived only formally. It is not verified precisely how the gradient of the loss function can be obtained from the backward solution of the adjoint equation. The adjoint equation helps to compute not only the gradient but also the Hessian of the loss function. The adjoint equation is derived in a formal way by linearizing and neglecting the higher order terms also in [5]. The identification problem, namely finding the best parameter values of a dynamical system to fit given data is related to optimal control and is investigated also in [15].

Having provided an overview with the aim of presenting previous approaches and contextualizing the present work, we now return to the latter and summarize its novelties. We argue that the novelties of the paper spring from the functional approach enabling us to treat this topic with what we feel is more precision and elegance, but more concretely, it enables us to do the following:

- to treat the case of discrete, and continuous samples together via a general loss function;
- to offer a succinct and self-contained derivation of the adjoint equation;
- to present a time-differentiable homotopy as the continuous analogue of backpropagation;
- to obtain the differential equation governing this homotopy, the solution of which yields the gradient of the loss function.

The spirit of the functional approach is preserved in implementation via the JAX package [2], offering composable transformations with applications that are not limited to machine learning.

The paper is structured as follows. In Sect. 2, we present the abstract approach, construct the general loss function from building blocks, and prove in Theorem 2, that the adjoint equation yields the gradient of these.

Then, in Sects. 3, and 4, the adjoint equation is formulated, and the gradient of the general loss function is derived for the case of single, and multiple time points, see Theorem 4. In Sect. 5, we turn to implementing the abstract approach. In practice, to obtain the aforementioned gradient, one can solve the initial value problem (13), which presents the computable form of the adjoint equation, and the suitable initial condition. Lastly, in Sect. 6 we show some numerical examples illustrating the feasibility of the method.

2. General Approach

We will use the following standard notation for the solution that enables us to denote more clearly its dependence on the initial condition and on the parameters. Let $\phi(t,s,p,\theta) = x(t)$ denote the value of the solution of (1) at time t satisfying the initial condition $x(s) = p$. Then the initial value problem (1) takes the form

$$\dot{x}_{(t_0,x_0,\theta)}(t) = \partial_t \phi(t,t_0,x_0,\theta) = f\big(t,\phi(t,t_0,x_0,\theta),\theta\big)$$

for $t_0 < t < t_0 + 1$. Moreover, we introduce the forward transfer operator family $\varphi(\tau) : \mathcal{S} \to \mathcal{S}$ by the formula

$$\varphi(\tau)(s,p,\theta) = \big(\tau + s, \phi(\tau+s,s,p,\theta),\theta\big). \tag{3}$$

In words, $\varphi(\tau)$ advances the lifted dynamical system by time τ.

The function φ defines a dynamical system on the search space $\mathcal{S}$ and satisfies an autonomous differential equation, the right hand side of which is the lifted version of f, namely $F : \mathcal{S} \to \mathcal{S}$, defined as

$$F(s,p,\theta) = \big(1, f(s,p,\theta), 0\big),$$

that is, the following proposition holds.

Proposition 1. *The function φ satisfies the group property $\varphi(t+\tau) = \varphi(t) \circ \varphi(\tau)$ and the autonomous differential equation*

$$\varphi'(\tau) = F \circ \varphi(\tau)$$

for all t.

Proof. The group property can be derived by using the group property of ϕ as follows.

$$\begin{aligned}\varphi(t)\big(\varphi(\tau)(s,p,\theta)\big) &= \varphi(t)\big(\tau+s,\phi(\tau+s,s,p,\theta),\theta\big)\\ &= \Big(t+\tau+s, \phi\big(t+\tau+s,\tau+s,\phi(\tau+s,s,p,\theta),\theta\big),\theta\Big)\\ &= \big(t+\tau+s,\phi(t+\tau+s,s,p,\theta),\theta\big) = \varphi(t+\tau)(s,p,\theta).\end{aligned}$$

The differential equation can be obtained by differentiating (3) with respect to τ.

$$\begin{aligned}\varphi'(\tau)(s,p,\theta) &= \big(1,\partial_t\phi(s+\tau,s,p,\theta),0\big)\\ &= \big(1,f(s+\tau,\phi(s+\tau,s,p,\theta),\theta),0\big)\\ &= \big(F\circ\varphi(\tau)\big)(s,p,\theta).\end{aligned}$$

□

We are now ready to construct the loss function. The input of this function will be the triple (t_0, x_0, θ) including both the initial condition and the parameters. This triple determines the solution of the initial value problem (1) uniquely on $[t_0, t_0 + 1]$. The value of the loss function compares the measurement $y(\tau)$ to the state $\phi(t_0 + \tau, t_0, x_0, \theta)$ for some time instants $\tau \in [0, 1]$.

To this end, we introduce the differentiable function $h(\tau) : \mathcal{S} \to \mathbb{R}$, that maps the state triple at time $t_0 + \tau$ to a scalar representing the error at this time.

One of the most typical error functions is the square of the difference, that is used in the $d = 1$ dimensional cases (2a), (2b) of Sect. 1. In that case, the function $h(\tau)$ takes the form of

$$h(\tau)(s, p, \theta) = \big(p - y(\tau)\big)^2.$$

To turn this into a function of the initial state, we compose it from the right by the function $\varphi(\tau)$, which advances the state by time τ. The result is the function

$$h(\tau) \circ \varphi(\tau) : \mathcal{S} \to \mathbb{R}.$$

In the case of the simple squared difference of (2a), (2b), we get

$$\big(h(\tau) \circ \varphi(\tau)\big)\big(t_0, x_0, \theta\big) = \big(\phi(t_0 + \tau, t_0, x_0, \theta) - y(\tau)\big)^2.$$

If we want to compare the solution to the measurement at several time instants $\tau \in [0, 1]$, and then aggregate the resulting differences, then we take a probability measure σ on $[0, 1]$ that is concentrated to those time instants and integrate the point-wise error $h(\tau) \circ \varphi(\tau)$ with respect to this measure, leading to the definition of the general loss function,

$$\mathcal{L} = \int_{[0,1]} h(\tau) \circ \varphi(\tau) \; d\sigma(\tau). \tag{4}$$

To emphasize the arguments of the loss function, this definition can be written in the form

$$\mathcal{L}(t_0, x_0, \theta) = \int_{[0,1]} \big(h(\tau) \circ \varphi(\tau)\big)\big(t_0, x_0, \theta\big) \; d\sigma(\tau).$$

We visualize the general loss function in Fig. 1.

The goal of the learning process is to find a minimum of the loss function in the search space, i.e. to find the optimal values of the initial condition (t_0, x_0) and the parameter θ. To this end, the efficient calculation of the gradient of the loss function, denoted by $\mathcal{L}'$, is needed. Equation (4) shows that this gradient can be obtained from the derivative $(h(\tau) \circ \varphi(\tau))'$. It turns out that computing this derivative is numerically demanding, hence an alternative route using the so-called adjoint equations has been developed, see e.g. [5]. Below we show a general derivation of this equation and a new proof for the fact that the gradient of the loss function can be obtained from the adjoint equation.

$$\begin{tikzcd}
 & \cdots \arrow[r, dashed] \arrow[d, dotted, no head] & \cdot \arrow[dr, no head, bend left, "\int d\sigma(\tau)"] \arrow[d, dotted, no head] & \\
(t_0, x_0, \theta) \arrow[ur, bend left] \arrow[r, "\varphi(\tau)"] \arrow[dr, bend right] & \cdots \arrow[r, dashed, "h(\tau)"] \arrow[d, dotted, no head] & \cdot \arrow[r, phantom] \arrow[d, dotted, no head] & \mathcal{L}(t_0, x_0, \theta) \\
 & \cdots \arrow[r, dashed] & \cdot \arrow[ur, no head, bend right] &
\end{tikzcd}$$

$$(1 + d + k) \qquad (|\mathcal{T}|, 1 + d + k) \qquad (|\mathcal{T}|, 1) \qquad (1)$$

FIGURE 1. The loss function $\mathcal{L}$, which, in words, for each time $0 \le \tau \le 1$, transfers the initial state triple (t_0, x_0, θ) forward by time τ, assigns a scalar score to the resulting state triple using $h(\tau)$, and lastly aggregates these scores by integrating over [0, 1] with respect to the measure σ. The bottom row lists the dimensions, and shapes of the objects encountered, in a form related to implementation. These are, from left to right: a (row) vector, a matrix with the same number of columns, and $|\mathcal{T}|$ rows, that is, one for each time instant; a column vector with the same number of rows, and lastly a scalar

The main idea of this general approach is that calculating $h(\tau)' \circ \varphi(\tau)$ is relatively easy, and it is connected to the desired derivative $\big(h(\tau) \circ \varphi(\tau)\big)'$ by a differential equation, the adjoint equation.

In other words, we show that there exists a differential equation, such that its solution acts as a continuous transformation between the functions $\big(h(\tau) \circ \varphi(\tau)\big)'$ and $h(\tau)' \circ \varphi(\tau)$, much like a homotopy mapping one curve to another.

Indeed, given a time $0 \le t \le \tau$, let us define

$$\Lambda(\tau, t) = h(\tau) \circ \varphi(\tau - t),$$

and use the group property of φ to split the map $h(\tau) \circ \varphi(\tau)$ as

$$h(\tau) \circ \varphi(\tau) = h(\tau) \circ \varphi(\tau - t) \circ \varphi(t) = \Lambda(\tau, t) \circ \varphi(t).$$

Next, we introduce the desired homotopy $\lambda(\tau, t)$ as

$$\lambda(\tau, t) = \big(h(\tau) \circ \varphi(\tau - t)\big)' \circ \varphi(t) = \Lambda(\tau, t)' \circ \varphi(t).$$

Clearly, then $\lambda(\tau, \tau) = h(\tau)' \circ \varphi(\tau)$, and $\lambda(\tau, 0) = \big(h(\tau) \circ \varphi(\tau)\big)'$ hold, i.e. λ connects the two mappings. The time evolution of λ, that is the function $t \mapsto \lambda(\tau, t)$ satisfies a differential equation, that is generally called the adjoint equation. This is the statement of the following theorem.

Theorem 2. *The function $\lambda(\tau, \cdot)$ satisfies the differential equation*

$$\partial_t \lambda(\tau, t) = -\lambda(\tau, t) \cdot \big(F' \circ \varphi(t)\big) \qquad \textit{for} \quad 0 < t < \tau \le 1. \qquad (5)$$

Proof. By the group property, and the chain rule, we have that

$$\begin{aligned}\Lambda(\tau,t) &= \Lambda(\tau,t+s)\circ\varphi(s),\\ \Lambda(\tau,t)' &= \big(\Lambda(\tau,t+s)'\circ\varphi(s)\big)\cdot\varphi(s)'.\end{aligned}$$

Applying this to λ, we get that

$$\begin{aligned}\lambda(\tau,t) &= \Lambda(\tau,t)'\circ\varphi(t)\\ &= \big(\Lambda(\tau,t+s)'\circ\varphi(s)\circ\varphi(t)\big)\cdot\big(\varphi(s)'\circ\varphi(t)\big)\\ &= \lambda(\tau,t+s)\cdot\big(\varphi(s)'\circ\varphi(t)\big).\end{aligned}$$

Now we take the derivative with respect to s, and substitute $s=0$.

$$\begin{aligned}0 &= \partial_t\lambda(\tau,t+s)\cdot\big(\varphi(s)'\circ\varphi(t)\big)+\lambda(\tau,t+s)\cdot\frac{d}{ds}\left(\varphi(s)'\circ\varphi(t)\right)\Big|_{s=0}\\ &= \partial_t\lambda(\tau,t)+\lambda(\tau,t)\cdot\big(\varphi'(0)'\circ\varphi(t)\big)\\ &= \partial_t\lambda(\tau,t)+\lambda(\tau,t)\cdot\big(F'\circ\varphi(t)\big),\end{aligned}$$

where the last line uses

$$\frac{d}{d\tau}\left(\varphi(\tau)'\right)\Big|_{\tau=0} = \big(\varphi'(0)\big)' = \big(F\circ\varphi(0)\big)' = F'.$$

□

To summarize, the general approach is to solve the the differential equation (1), then the gradient of the loss function is obtained by solving the adjoint equation backward, from $t=t_0+\tau$ to $t=t_0$. So far we have obtained the derivative $\big(h(\tau)\circ\varphi(\tau)\big)'$. In the next two sections, we derive the gradient of the loss function when we have a single time point, i.e. the probability measure is concentrated on a single point, and when we have several time instants.

3. The Case of a Single Time Point

Let us first consider the case of a single measurement at a fixed time τ. This corresponds to the case where σ is concentrated on the single time instant τ. Then, the loss function is simply $h(\tau)\circ\varphi(\tau)$, which acts on $\mathcal{S}$ by the formula

$$\mathcal{L}(t_0,x_0,\theta) = h(\tau)\big(t_0+\tau,\phi(t_0+\tau,t_0,x_0,\theta),\theta\big). \tag{6}$$

For the sake of brevity, and exploiting that τ is now fixed, we introduce the functions $\bar{h}=h(\tau)$, and $\bar{\varphi}=\varphi(\tau)$, and we let $\xi_0=(t_0,x_0,\theta)$. With these, the loss function can be written as

$$\mathcal{L}(\xi_0) = \bar{h}\big(\bar{\varphi}(\xi_0)\big).$$

We are interested in calculating the gradient of this function using backpropagation, summarized in Fig. 2.

$$\begin{tikzcd}
\xi_0 \arrow[dr, mapsto, dashed] & & & \bar{h}\big(\bar{\varphi}(\xi_0)\big) \\
& \bar{\varphi} \arrow[r, mapsto, dashed, bend left, "\bar{\varphi}(\xi_0)"] \arrow[dl, mapsto] & \bar{h} \arrow[ur, mapsto, dashed] \arrow[l, mapsto, bend left, "\bar{h}'\big(\bar{\varphi}(\xi_0)\big)"] & \\
(\bar{h} \circ \bar{\varphi})'(\xi_0) & & & 1 \arrow[ul, mapsto]
\end{tikzcd}$$

FIGURE 2. The forward, and the backward pass in the case of a single time point τ. The arrows representing the former are dashed. During the forward pass we start from ξ_0 and calculate $\bar{\varphi}(\xi_0)$, then $\bar{h}\big(\bar{\varphi}(\xi_0)\big)$. During the backward pass we take these values, and starting from $1 = \mathrm{id}'\big(\bar{h}(\bar{\varphi}(\xi_0))\big)$, we calculate $\bar{h}'\big(\bar{\varphi}(\xi_0)\big)$, and lastly $(\bar{h} \circ \bar{\varphi})'(\xi_0)$

We note, again, that in the simple case when $\bar{h}(s, p, \theta) = \big(p - y(\tau)\big)^2$, the loss function takes the form

$$\mathcal{L}(t_0, x_0, \theta) = \big(\phi(t_0 + \tau, t_0, x_0, \theta) - y(\tau)\big)^2.$$

Based on the result of the previous section, the gradient of the loss function can be calculated as follows.

Corollary 3. *Let the loss function be given by* (6)*. Then its gradient can be obtained as* $\mathcal{L}' = \lambda(\tau, 0)$*, where* $\lambda(\tau, \cdot)$ *is the solution of the adjoint equation* (5)*, solving it backward starting from the initial condition* $\lambda(\tau, \tau) = \bar{h}' \circ \bar{\varphi}$ *with* $\bar{h} = h(\tau)$*, and* $\bar{\varphi} = \varphi(\tau)$*.*

4. The Case of Multiple Time Points

Similarly to the single point case, we would like to find a way to transform the various $\lambda(\tau, \tau) = h(\tau)' \circ \varphi(\tau)$ functions, possibly scaled values of which are obtained during backpropagation, into the derivative of the loss function (4), that is, into $\mathcal{L}'$.

Given a t from the closed unit interval, let us consider how the loss function depends on the state at time t. During the forward pass, that is, the evaluation of the loss function $\mathcal{L}$, the initial value problem (1) is solved forward in time. This implies that the aforementioned state affects the states at later times, that is, those at time τ for all $t \le \tau \le 1$.

This effect is the following. First, the state is carried to time τ via $\varphi(\tau - t)$, then the resulting state is fed into $h(\tau)$, yielding the partial loss value belonging to time τ. Therefore, we form the composition of these two functions,

$$h(\tau) \circ \varphi(\tau - t)$$

for each $t \leq \tau \leq 1$, and aggregate the results using the measure σ to get the function

$$L(t) = \int_{[0,1]} \mathbb{I}(t \leq \tau) \cdot h(\tau) \circ \varphi(\tau - t)\, d\sigma(\tau),$$

which can be seen to be the $\tau-$aggregated version of $\Lambda(\tau, t)$. This becomes a proper loss function, in the sense that it will take the initial state to some loss value, if we compose it from the right by $\varphi(t)$. Indeed,

$$L(t) \circ \varphi(t)$$

is a family of loss functions that measure the loss encountered on the interval $[t, 1]$. Using that $\varphi(0)$ is the identity, equation (4) yields $L(0) = \mathcal{L}$.

We may now proceed analogously to the single point case, and define

$$\begin{aligned} l(t) &= L(t)' \circ \varphi(t) \\ &= \int_{[0,1]} \mathbb{I}(t \leq \tau) \cdot \big(h(\tau) \circ \varphi(\tau - t)\big)' \circ \varphi(t)\, d\sigma(\tau) \\ &= \int_{[0,1]} \mathbb{I}(t \leq \tau) \cdot \lambda(\tau, t)\, d\sigma(\tau), \end{aligned}$$

the $\tau-$aggregated version of $\lambda(\tau, t)$, which will act as the transformation between the functions

$$\begin{aligned} l(0) &= \mathcal{L}', \\ l(1) &= \sigma(\{1\}) \cdot h(1)' \circ \varphi(1). \end{aligned}$$

Let us describe now the time evolution of l. The case of the continuous and the discrete sample can be treated together by assuming that σ decomposes into the sum of an absolutely continuous and a discrete part, that is $\sigma = \sigma_c + \sigma_d$ with Radon-Nikodym derivatives ρ_c and ρ_d. Then we have that

$$l(t) = \int_0^1 \mathbb{I}(t \leq \tau) \cdot \lambda(\tau, t) \cdot \rho_c(\tau)\, d\tau + \sum_{j=1}^{n} \mathbb{I}(t \leq \tau_j) \cdot \lambda(\tau_j, t) \cdot \rho_d(\tau_j), \tag{7}$$

and the time evolution of this family is given by the following theorem.

Theorem 4.

$$l'(t) = -\lambda(t,t) \cdot \rho_c(t) - \sum_{j=1}^{n} \lambda(\tau_j, \tau_j) \cdot \rho_d(\tau_j) \cdot \delta_{\{\tau_j\}} - l(t) \cdot \big(F' \circ \varphi(t)\big) \qquad 0 < t < 1 \tag{8}$$

Proof. The idea of the proof is to differentiate (7), and apply Theorem 2. For the continuous part, we use the Leibniz rule.

$$\begin{aligned}
l'(t) &= -\lambda(t,t)\rho_c(t) + \int_t^1 \partial_t\lambda(\tau,t)\rho_c(\tau)\,d\tau \\
&\quad - \sum_{j=1}^{n} \lambda(\tau_j,\tau_j)\rho_d(\tau_j)\delta_{\{\tau_j\}} + \sum_{j=1}^{n} \mathbb{I}(t \le \tau_j)\cdot \partial_t\lambda(\tau_j,t)\rho_d(\tau_j) \\
&= -\lambda(t,t)\rho_c(t) - \int_t^1 \lambda(\tau,t)\cdot \big(F' \circ \varphi(t)\big)\rho_c(\tau)\,d\tau \\
&\quad - \sum_{j=1}^{n} \lambda(\tau_j,\tau_j)\rho_d(\tau_j)\delta_{\{\tau_j\}} - \sum_{j=1}^{n} \mathbb{I}(t \le \tau_j)\cdot \lambda(\tau,t)\cdot \big(F' \circ \varphi(t)\big)\rho_d(\tau_j) \\
&= -\lambda(t,t)\rho_c(t) - \sum_{j=1}^{n} \lambda(\tau_j,\tau_j)\rho_d(\tau_j)\delta_{\{\tau_j\}} \\
&\quad - \left(\int_t^1 \lambda(\tau,t)\rho_c(\tau)\,d\tau + \sum_{j=1}^{n} \mathbb{I}(t \le \tau_j)\cdot \lambda(\tau,t)\rho_d(\tau_j)\right)\cdot \big(F' \circ \varphi(t)\big) \\
&= -\lambda(t,t)\cdot\rho_c(t) - \sum_{j=1}^{n} \lambda(\tau_j,\tau_j)\cdot\rho_d(\tau_j)\cdot\delta_{\{\tau_j\}} - l(t)\cdot\big(F' \circ \varphi(t)\big)
\end{aligned}$$

□

We take a moment to underline yet again that $\lambda(t,t) = h(t)' \circ \varphi(t)$, and that $\lambda(t,t)$ are functions from which we obtain values during backpropagation.

Corollary 5. *Consider the general loss function* (4)*. Its gradient is* $\mathcal{L}' = l(0)$*, where* l *is the solution of the adjoint equation* (8)*, which we solve backward in time starting from the initial condition* $l(1) = \sigma(\{1\}) \cdot h(1)' \circ \varphi(1)$*.*

5. Application of the General Theory

In this section, we turn to the application of the general theory presented above. As the initial setting, we are given the input to $\mathcal{L}$, namely the triple (t_0, x_0, θ).

During the forward pass, the initial value problem (1) is solved to produce a solution $x_{(t_0,x_0,\theta)}$, which we denote simply by x, for the sake of brevity. This is then fed into the functions $h(\tau)$ point-wise, the results of which are aggregated via integration by the measure σ on $[0,1]$.

During the backward pass, we use x, a result of the forward pass, and solve another initial value problem backwards in time to backpropagate the gradient obtained in the form of a function g. We note that if we have a finite number of time points, then g is really just a finite dimensional vector.

5.1. The Case of a Single Time Point

First, we illustrate how to apply the general theory in the case of a single time point τ. To simplify matters as much as possible, we consider a differential equation with a $d = 1$ dimensional phase space and a $k = 1$ dimensional parameter. Moreover, we pick the squared difference error function $h(\tau)(s, p, \theta) = \big(p - y(\tau)\big)^2$. In this case, the loss function maps $\mathbb{R}^3$ to $\mathbb{R}$ following the formula

$$\mathcal{L}(t_0, x_0, \theta) = \big(\phi(t_0 + \tau, t_0, x_0, \theta) - y(\tau)\big)^2,$$

which is consistent with (2a), assuming $n = 1$ observation(s).

According to Corollary 3, the derivative of the loss function is $\mathcal{L}' = \lambda(\tau, 0)$, where $\lambda(\tau, \cdot)$ is the solution of the adjoint equation (5) satisfying the initial condition $\lambda(\tau, \tau) = h(\tau)' \circ \varphi(\tau)$.

The adjoint equation (5) is in a functional form. Applying both the left and the right-hand-sides to a point (t_0, x_0, θ) leads to a linear system of three differential equations. Let us now expand on these. First, we introduce the function that is going to satisfy this linear differential equation as

$$\big(a_1(t), a_2(t), a_3(t)\big) = a(t) = \lambda(\tau, t)(t_0, x_0, \theta),$$

where components a_i are now real-valued functions.

Then the adjoint equation itself is the non-autonomous linear differential equation of the form

$$\dot{a}(t) = -a(t)A(t),$$

where the coefficient matrix is $A(t) = F'\big(\varphi(t)(t_0, x_0, \theta)\big)$. Elaborating on this, we note that since $\varphi(t)(t_0, x_0, \theta) = \big(t_0 + t, x(t_0 + t), \theta\big)$, where $x(t_0 + t) = \phi(t_0 + t, t_0, x_0, \theta)$, and

$$F'(s, p, \theta) = \begin{pmatrix} 0 & 0 & 0 \\ \partial_t f(s, p, \theta) & \partial_x f(s, p, \theta) & \partial_\theta f(s, p, \theta) \\ 0 & 0 & 0 \end{pmatrix},$$

we have that

$$A(t) = \begin{pmatrix} 0 & 0 & 0 \\ \partial_t f\big(t_0 + t, x(t_0 + t), \theta\big) & \partial_x f\big(t_0 + t, x(t_0 + t), \theta\big) & \partial_\theta f\big(t_0 + t, x(t_0 + t), \theta\big) \\ 0 & 0 & 0 \end{pmatrix}.$$

Therefore, multiplication leads us to the expanded version of the adjoint equation,

$$\dot{a}_1(t) = -a_2(t)\partial_t f\big(t_0 + t, x(t_0 + t), \theta\big), \tag{9}$$

$$\dot{a}_2(t) = -a_2(t)\partial_x f\big(t_0 + t, x(t_0 + t), \theta\big), \tag{10}$$

$$\dot{a}_3(t) = -a_2(t)\partial_\theta f\big(t_0 + t, x(t_0 + t), \theta\big). \tag{11}$$

Thus, we need to solve the second equation for a_2, first, and then a_1 and a_3 can be obtained by simple integration.

Lastly, we derive the initial conditions for the unknown functions a_i. The abstract initial condition takes the form $\lambda(\tau,\tau) = h(\tau)' \circ \varphi(\tau)$, and we have that $a(\tau) = \lambda(\tau,\tau)(t_0, x_0, \theta)$. Differentiating $h(\tau)(s,p,\theta) = \big(p - y(\tau)\big)^2$ yields

$$h(\tau)'(s,p,\theta) = \Big(0, 2\big(p - y(\tau)\big), 0\Big).$$

Using $\varphi(\tau)(t_0, x_0, \theta) = \big(t_0 + \tau, x(t_0 + \tau), \theta\big)$, we obtain

$$a(\tau) = h(\tau)'\big(\varphi(\tau)(t_0, x_0, \theta)\big) = \Big(0, 2\big(x(t_0 + \tau) - y(\tau)\big), 0\Big),$$

leading to the initial condition

$$a_1(\tau) = 0, \qquad a_2(\tau) = 2\big(x(t_0 + \tau) - y(\tau)\big), \qquad a_3(\tau) = 0. \tag{12}$$

Thus, the gradient of the loss function can be obtained as

$$\mathcal{L}'(t_0, x_0, \theta) = a(0),$$

where $a(t) = \big(a_1(t), a_2(t), a_3(t)\big)$ is the solution of system (9)–(11) subject to the initial condition (12).

For the interested Reader, it might be useful to consider the case $f(p,\theta) = p\theta$, when system (9)–(11) can be solved analytically as

$$a(t) = 2\big(\mathrm{e}^{\theta\tau} x_0 - y(\tau)\big)\big(0, \mathrm{e}^{\theta(\tau - t)}, \mathrm{e}^{\theta\tau} x_0 (\tau - t)\big),$$

leading to

$$\mathcal{L}'(t_0, x_0, \theta) = a(0) = 2\big(\mathrm{e}^{\theta\tau} x_0 - y(\tau)\big)\big(0, \mathrm{e}^{\theta\tau}, \tau \mathrm{e}^{\theta\tau} x_0\big).$$

In this special case, the gradient of the loss function can also simply be obtained by direct differentiation of

$$\mathcal{L}(t_0, x_0, \theta) = \big(\mathrm{e}^{\theta\tau} x_0 - y(\tau)\big)^2.$$

5.2. The Case of Multiple Time Points

The case of multiple time points can be treated similarly to the single point case, seen in the previous subsection.

We start by considering the general loss function $\mathcal{L}$ as defined in (4). According to Corollary 3, its derivative is calculable as $\mathcal{L}' = l(0)$, where l is the solution of the adjoint equation (8), satisfying the initial condition $l(1) = \sigma(\{1\}) \cdot h(1)' \circ \varphi(1)$.

We now take (8) in its functional form, and apply its functions to the input triple (t_0, x_0, θ). Given a t from the unit interval, the three functions that we need to evaluate are $l(t), \lambda(t,t)$, and $F' \circ \varphi(t)$. In doing so, we will freely use that $\varphi(\tau)(t_0, x_0, \theta) = \big(t_0 + \tau, x(t_0 + \tau), \theta\big)$. We start with $l(t)$, and define the function that is to satisfy the adjoint equation as

$$a(t) = \big(a_1(t), a_2(t), a_3(t)\big) = l(t)(t_0, x_0, \theta) \qquad\qquad \in \mathbb{R}^{1+d+k}.$$

Then, we consider source term $\lambda(t,t) = h(t)' \circ \varphi(t)$, which might be considered the input gradient during the backpropagation step, and define the corresponding function

$$g(t) = \big(h(t)' \circ \varphi(t)\big)(t_0, x_0, \theta) = h(t)'\big(t_0 + t, x(t_0+t), \theta\big) \qquad \in \mathbb{R}^{1+d+k}.$$

Then, we mimic the previous subsection and let

$$A(t) = \big(F' \circ \varphi(t)\big)\big(t_0, x_0, \theta\big) = F'\big(t_0 + t, x(t_0+t), \theta\big) \quad \in \mathbb{R}^{(1+d+k)\times(1+d+k)}.$$

Lastly, we define

$$J(t) = f'\big(t_0 + t, x(t_0+t), \theta\big) \qquad\qquad \in \mathbb{R}^{d\times(1+d+k)},$$

and note that

$$a \cdot A = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ J \\ 0 \end{bmatrix} = a_2 \cdot J.$$

Still following Theorem 4, we are ready to state the initial value problem to be solved backward in time. Indeed, we plug in the recently defined functions to get

$$\begin{cases} \dot a(t) = -g(t)\rho_c(t) - \sum\limits_{j=1}^{n} g(\tau_j)\rho_d(\tau_j)\delta_{\{\tau_j\}} - a_2(t)\cdot J(t), & 0 < t < 1 \\ a(1) = g(1)\rho_d(1), \end{cases} \tag{13}$$

where the initial value follows from the formula

$$a(1) = \sigma(\{1\}) \cdot \big(h(1)' \circ \varphi(1)\big)\big(t_0, x_0, \theta\big) = \rho_d(1) \cdot g(1),$$

where we have used that $\sigma = \sigma_d + \sigma_c$, and $\sigma_c(\{1\}) = 0$ by its absolute continuity.

To summarize, given the values $\{g(t) : t \in \mathcal{T}\}$, the gradient of the loss function can be obtained as

$$\mathcal{L}'(t_0, x_0, \theta) = a(0),$$

where a is the solution of the initial value problem (13).

We take a moment to state that the ith component of (13) for $i = 1, 2, 3$ is

$$\begin{cases} \dot a_i(t) = -g_i(t)\rho_c(t) - \sum\limits_{j=1}^{n} g_i(\tau_j)\rho_d(\tau_j)\delta_{\{\tau_j\}} - a_2(t)\partial_{\xi_i} f\big(t_0 + t, x(t_0+t), \theta\big), & 0 < t < 1 \\ a_i(1) = g_i(1)\rho_d(1), \end{cases}$$

where $(\xi_1, \xi_2, \xi_3) = (t, x, \theta)$. We note that this involves a nontrivial differential equation only for $i = 2$, therefore having solved that first, the rest of the components a_1, and a_3 may be found by integration.

We note that using a discrete set of observations in a continuous world has its price, namely the Dirac delta terms $\delta_{\{\tau_j\}}$ mean that that a has jumps of possibly nonzero magnitude at times τ_j. In practice, this means that the numerical algorithm used to solve problem (13) has to be able to introduce artificial bumps in the solution it is producing. Alternatively, we may introduce the bumps by solving initial value problems on each subinterval $[\tau_n, 1], \dots, [\tau_{j-1}, \tau_j], \dots [0, \tau_1]$, and bumping the solution a through the initial conditions.

To make the latter argument more precise, we firstly let $\tau_{n+1} = 1$, and $\tau_0 = 0$, without introducing new time instants, and define $a^{n+1} \equiv 0$. Then, for each $j = n, \dots, 0$, we recursively introduce a sequence of functions

$$a^j : [\tau_j, \tau_{j+1}] \to \mathbb{R}^{1+d+k},$$

as the solutions to the sequence of initial value problems

$$\begin{cases} \dot{a}^j(t) = -g(t)\rho_c(t) - a_2^j(t) \cdot J(t), & \tau_j < t < \tau_{j+1} \\ a^j(\tau_{j+1}) = g(\tau_{j+1})\rho_d(\tau_{j+1}) + a^{j+1}(\tau_{j+1}), \end{cases} \tag{14}$$

solving all of which in succession, we arrive at $a^0(\tau_0) = a^0(0) = \mathcal{L}'(t_0, x_0, \theta)$. We note that the $g(\tau_{j+1})\rho_d(\tau_{j+1})$ terms get added with a positive sign, since a jump in forward time becomes the same jump, but negated, when looking at it in reversed time.

Lastly, we underline three important special cases.

The case of continuous data assumes that continuous data is available on the whole unit interval, that is, when $y(\tau)$ is defined for each τ from $[0, 1]$. We do not wish to highlight any single time instant in particular, therefore we let $\rho_d \equiv 0$, and we set the continuous weights to be uniform, that is, $\rho_c \equiv 1$. In other words, σ is the Lebesgue-measure on $[0, 1]$. In this case, the loss function is

$$\mathcal{L}(t_0, x_0, \theta) = \int_0^1 h(\tau)\big(t_0 + \tau, x(t_0 + \tau), \theta\big)\ d\tau,$$

and (13) becomes

$$\begin{cases} \dot{a}(t) = -g(t) - a_2(t) \cdot J(t), & 0 < t < 1 \\ a(1) = 0, \end{cases} \tag{15}$$

since $\rho_d \equiv 0$.

The case of a single observation assumes that we have a single observation at time τ. In this case, σ is concentrated on τ, that is, the continuous part is zero, $\rho_c \equiv 0$, while the discrete part is zero everywhere except at τ, where $\rho_d(\tau) = 1$. We can consider three cases based on the value of $\tau \in [0, 1]$. If $\tau = 0$, then there is no need to solve any initial value problem. If $\tau = 1$, then (13) becomes

$$\begin{cases} \dot{a}(t) = -a_2(t) \cdot J(t), & 0 < t < 1 \\ a(1) = g(1), \end{cases}$$

where the right hand side doesn't show the Dirac delta term that sits at $\tau = 1$, since it is outside of the interval where this differential equation is solved. This is a terse version of the single point case outlined in the previous subsection. If $0 < \tau < 1$, then (13) becomes

$$\begin{cases} \dot{a}(t) = -g(\tau)\delta_{\{\tau\}} - a_2(t) \cdot J(t), & 0 < t < 1 \\ a(1) = 0, \end{cases}$$

which is a homogeneous linear system on $(\tau, 1)$, and consequently, its solution there is zero, because of the initial condition $a(1) = 0$. At time τ, a has a jump of $g(\tau)$, and from that point, the homogeneous differential equation can transfer the now non-zero state to something other than zero. This process amounts to the solution of the initial value problem

$$\begin{cases} \dot{a}(t) & = -a_2(t) \cdot J(t), \qquad \tau > t > 0 \\ a(\tau) & = g(\tau), \end{cases}$$

which is, again, what the treatment of the single point case of the previous subsection predicted.

The case of finitely many observations extends the previous case, allowing for more than one, but still only finitely many observations taken at times $\tau_1, \dots, \tau_n$. In this case, σ is concentrated on these points, and consequently $\rho_c \equiv 0$. For those interested in this setting, (14) is the formula to turn to, which, given the context, becomes

$$\begin{cases} \dot{a}^j(t) & = -a_2^j(t) \cdot J(t), \qquad \tau_j < t < \tau_{j+1} \\ a^j(\tau_{j+1}) & = g(\tau_{j+1})\rho_d(\tau_{j+1}) + a^{j+1}(\tau_{j+1}), \end{cases}$$

where, as previously, we assume that $\tau_0 = 0, \tau_{n+1} = 1, a^{n+1} \equiv 0$, and the functions

$$a^j : [\tau_j, \tau_{j+1}] \to \mathbb{R}^{1+d+k} \qquad j = n \dots 0$$

have to be determined by solving the corresponding initial value problems in succession, to lastly arrive at the gradient of the loss function, $a^0(\tau_0) = a^0(0) = \mathcal{L}'(t_0, x_0, \theta)$.

6. Numerical Experiments

In this section, we present the results of numerical experiments as evidence in support of Theorem 4. We demonstrate that a gradient descent that obtains the necessary gradients via (15) as outlined in this paper is able to lessen small perturbations in an optimal parameter triple $\xi_0 = (t_0, x_0, \theta)$.

The experiments proceed as follows. To obtain our input data we solve an initial value problem (1) parameterized by ξ_0, and sample the first component of the resulting trajectory. We consider two cases.

In the first, continuous case, we assume that the entirety of this component is available to the optimization process. To mimic measurement errors, each time this component is evaluated, the result contains an additive error term that is normally distributed. In this case, the function family h is the square of the difference between the first component of the state of the dynamical system and the sample y.

In the second, discrete case, we uniformly divide the unit interval into subintervals. We then generate a discrete sample by considering the input data of the previous case and sampling it at a time instant from each subinterval, where these time instants are drawn from truncated normal distributions that are centered at the intervals' midpoints. Our input data y will then be a piecewise constant function, which takes the sampled value on each subinterval. We modify the h of the continuous case by multiplying it with a weight function, which is, on each subinterval, the probability density function of the time instant where the trajectory component has been sampled.

Then we construct the computational graph, or loss function, using our input data y, the vector field of the initial value problem f, and the loss function components h. Lastly, we apply a small random normal perturbation to the true parameter triple ξ_0, and initiate a gradient descent starting from the perturbed triple, in order to reduce the loss value.

As initial value problems, we consider the SI model with a fixed population of 10

$$\begin{cases} \dot{S} &= -\frac{\beta I S}{10} \\ \dot{I} &= \frac{\beta I S}{10} - \gamma I \end{cases} \qquad t_0 = 0 \qquad \begin{cases} S(t_0) &= 9 \\ I(t_0) &= \frac{1}{2} \end{cases} \qquad \begin{bmatrix} \beta & \gamma \end{bmatrix} = \begin{bmatrix} 10 & 3 \end{bmatrix}, \tag{16}$$

and the Lotka–Volterra equations

$$\begin{cases} \dot{u} &= (a - bv)u \\ \dot{v} &= (du - c)v \end{cases} \qquad t_0 = 0 \qquad \begin{cases} u(t_0) &= \frac{1}{2} \\ v(t_0) &= \frac{1}{2} \end{cases} \qquad \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 10 & 10 \\ 10 & 10 \end{bmatrix}. \tag{17}$$

We have ran the experiment for each set of input data, for each initial value problem. We have repeated each experiment 4 times, so as to get a better idea of the loss values encountered during the iteration. The results of the $2 \times 2 \times 4$ experiments are summarized in Fig. 3.

The experiments have been implemented in JAX [2]. The implementation tries to mimic the mathematics presented in this paper. In particular, it has not been optimized for computational efficiency. In practice, calculating the gradients requires the numerical solution of an initial value problem, and further numerical integration. This implies that the amount of work required for each gradient descent step depends on the numerical tolerances one specifies, with looser tolerances implying faster iteration. On the other hand, looser tolerances imply less precise gradients. It is unclear how these tolerances should be chosen, perhaps even varied during the iteration, to render the computational process more efficient in terms of the decrement of the loss value per unit work.

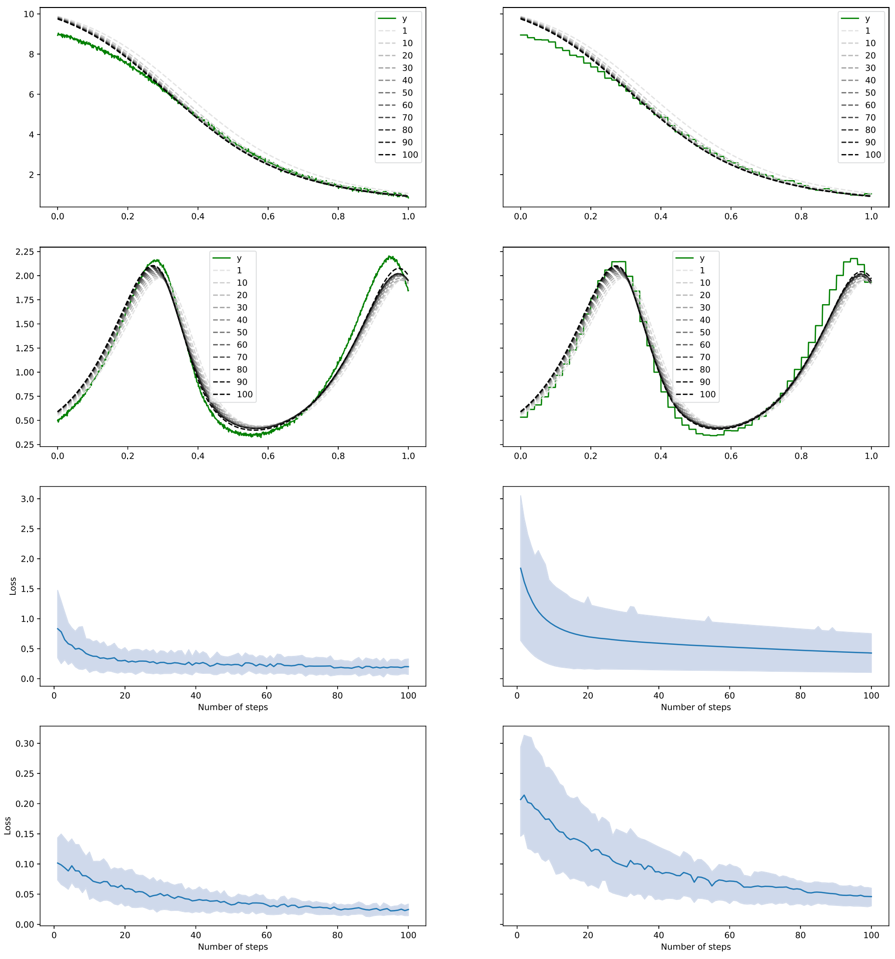

FIGURE 3. The two quadruples depict the results of 100 gradient descent steps starting from a slightly perturbed initial value problem parameter triple (t_0, x_0, θ). In each quadruple, the first row belongs to the case of the SI model (16), while the second to that of the Lotka–Volterra equations (17). The first column shows the case of continuous input, the second that of discrete input. The upper quadruple shows the input data y, and how the current best estimate of the underlying trajectory component varies during the iteration. The lower quadruple shows the loss values encountered during the same time. The latter are based on 4 repetitions of each experiment

In the continuous case, increasing the amount of noise, the integrals become harder to evaluate, which results in increased computation time and decreased accuracy. In the discrete case, taking samples from each subinterval according to a truncated normal distribution implies that as the temporal uncertainty goes to zero, the value of the weight function at the midpoints goes to infinity, which corresponds to the discrete part of (13).

The evaluation of the loss function, that is, that of the final integral, is not necessary for the calculation of the gradients, and time may be saved by only evaluating it when necessary.

In the examples of this section, the parameter triple the gradient descent starts from is not far from the one which yields the input data. When the initial parameter triple is further, then the true and the predicted trajectories can be different enough qualitatively for the iterative process to get stuck. In these cases, one may mimic the idea of the stochastic gradient descent by replacing σ with a random measure for each gradient descent step. We have had success using random normal distributions that were modified so that the expected measure was approximately uniform on the unit interval. This uniformity appears important in making sure that on average, the stochastic choice of measure does not interfere with how the errors at each time instant are weighted.

Funding Open access funding provided by Eötvös Loránd University. I.F. was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. This research has been implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the ELTE TKP 2021-NKTA-62 funding scheme. The authors acknowledge support from the Hungarian Scientific Research Fund, OTKA (Grant No. 135241) and from the Ministry of Innovation and Technology NRDI Office within the framework of the Artificial Intelligence National Laboratory Programme.

Data Availibility Statement The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests The authors have no relevant financial or non-financial interests to disclose.

References

[1] Bhat, H.S.: System identification via the adjoint method. In: 2021 55th Asilomar Conference on Signals, Systems, and Computers, pp. 1317–1321 (2021). https://doi.org/10.1109/IEEECONF53345.2021.9723391

[2] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs, version 0.3.13 (2018)

[3] Cao, Y., Li, S., Petzold, L., Serban, R.: Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution. SIAM J. Sci. Comput. **24**(3), 1076–1089 (2003). https://doi.org/10.1137/S1064827501380630

[4] Chavent, G.: Identification of distributed parameter systems: about the output least square method, its implementation, and identifiability. In: IFAC Proceedings Volumes, 12(8):85–97 (1979) issn: 1474-6670. https://doi.org/10.1016/S1474-6670(17)65413-2. 5th IFAC Symposium on Identification and System Parameter Estimation, Darmstadt, Germany, 24–28 Sept

[5] Chen, R.T.Q., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 31. Curran Associates, Inc. (2018)

[6] Dupont, E., Doucet, A., Teh, Y.W.: Augmented neural odes. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019)

[7] Haber, E., Ruthotto, L.: Stable architectures for deep neural networks. Inverse Prob. **34**(1), 014004 (2017). https://doi.org/10.1088/1361-6420/aa9a90

[8] Kim, S., Ji, W., Deng, S., Ma, Y., Rackauckas, C.: Stiff neural ordinary differential equations. Chaos Interdiscip. J. Nonlinear Sci. **31**(9), 093122 (2021). https://doi.org/10.1063/5.0060697

[9] Lu, Y., Zhong, A., Li, Q., Dong, B.: Beyond finite layer neural networks: bridging deep architectures and numerical differential equations. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning, Volume 80 of Proceedings of Machine Learning Research, pp. 3276–3285. PMLR (2018)

[10] Nandi, S., Singh, T.: Adjoint based hessians for optimization problems in system identification. In: IEEE Conference on Control Technology and Applications,

CCTA 2017, Mauna Lani Resort, HI, USA, Aug 27–30, 2017, pp. 626–631. IEEE (2017). isbn: 978-1-5090-2182-6. https://doi.org/10.1109/CCTA.2017.8062532

[11] Plessix, R.-E.: A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. Geophys. J. Int. **167**(2), 495–503 (2006). https://doi.org/10.1111/j.1365-246X.2006.02978.x

[12] Polis, M., Goodson, R.: Parameter identification in distributed systems: a synthesizing overview. Proc. IEEE **64**(1), 45–61 (1976). https://doi.org/10.1109/PROC.1976.10066

[13] Raffard, R.L., Amonlirdviman, K., Axelrod, J.D., Tomlin, C.J.: An adjoint-based parameter identification algorithm applied to planar cell polarity signaling. IEEE Trans. Autom. Control **53**(Special Issue), 109–121 (2008). https://doi.org/10.1109/TAC.2007.911362

[14] Ruthotto, L., Haber, E.: Deep neural networks motivated by partial differential equations. J. Math. Imaging Vis. **62**(3), 352–364 (2020). https://doi.org/10.1007/s10851-019-00903-1

[15] Weinan, E.: A proposal on machine learning via dynamical systems. Commun. Math. Stat. **5**(1), 1–11 (2017). https://doi.org/10.1007/s40304-017-0103-z

Imre Fekete, András Molnár and Péter L. Simon
Department of Applied Analysis and Computational Mathematics, Institute of Mathematics
ELTE Eötvös Loránd University
Budapest
Hungary
e-mail: `imre.fekete@ttk.elte.hu`;
`andras.molnar@ttk.elte.hu`;
`peter.simon@ttk.elte.hu`

and

HUN-REN-ELTE Numerical Analysis and Large Networks Research Group
Budapest
Hungary

Received: March 5, 2023.

Accepted: November 12, 2023.