

<https://doi.org/10.1038/s42005-024-01606-9>

Tutorial: a beginner's guide to building a representative model of dynamical systems using the adjoint method

Check for updates

Leon Lettermann ^{1,2}, Alejandro Jurado¹, Timo Betz ¹, Florentin Wörgötter^{1,3} & Sebastian Herzog ¹

Building a representative model of a complex dynamical system from empirical evidence remains a highly challenging problem. Classically, these models are described by systems of differential equations that depend on parameters that need to be optimized by comparison with data. In this tutorial, we introduce the most common multi-parameter estimation techniques, highlighting their successes and limitations. We demonstrate how to use the adjoint method, which allows efficient handling of large systems with many unknown parameters, and present prototypical examples across several fields of physics. Our primary objective is to provide a practical introduction to adjoint optimization, catering for a broad audience of scientists and engineers.

The interplay between theory and empirical is central to advancing our understanding of complex systems and will probably become even more important in the future¹. The synergy between theoretical constructs, such as model descriptions by systems of differential equations like ordinary differential equations (ODEs) and partial differential equations (PDEs), and empirical evidence obtained through experimentation forms the cornerstone of scientific progress in many fields^{2–6}. Theoretical models, firmly grounded in the mathematical elegance of ODEs and PDEs, provide a predictive framework that allows us to comprehend and interpret the observed phenomena within intricate systems. They not only serve as blueprints for our understanding but also offer the ability to hypothesize and test various scenarios^{1,7}.

However, the full potential of these mathematical models is only unlocked when they are brought into harmony with empirical data garnered from real-world experiments. This endeavour entails the estimation of the parameters within the mathematical model. Experimental data serve as the empirical foundation upon which theoretical models are constructed, refined, and validated. This dynamic relationship between theory and experiment is an iterative, virtuous cycle. Theoretical constructs guide experiments by suggesting testable hypotheses, while experimental results, in turn, enrich and often challenge the theoretical models.

Sadly, accurate mathematical modelling, tailored to real-world problems, is often hindered by a lack of empirical detail due to unobservable quantities, missing initial values, or other unknown parameters; this often requires some best-practice process for the estimation of multiple parameters that govern the behaviour of the model. As a result,

building a trustworthy mathematical representation of a complex system from empirical evidence remains a challenging problem. The domain of complex systems⁸ provides an in-depth review, illustrating possible applications and problems in the field, which includes, for example, modelling the signaling in heart muscle tissues, important for understanding arrhythmia or defibrillation⁹. To address these problems, one needs methods for accurate and robust estimation of multiple model parameters, which is pivotal for progress in both scientific and engineering domains. Ideally, such a method should, in the first place, allow us to directly assess the validity of underlying model assumptions by verifying if parameters can be found, for which the model equations reproduce the observed data. In addition, however, more sophisticated applications are desired, such as to discern latent variables from empirical data, augment extant models through the incorporation of novel variables to enhance their congruence with empirical evidence, and facilitate the creation of parsimonious predictive models characterized by reduced dimensionality. In many cases, these issues can be addressed using the so-called *adjoint method*^{1,8,10–17}, which is a powerful, reliable, and efficient method for parameter estimation to hundreds of thousands of parameters, offering the multifaceted set of capabilities mentioned above.

The primary objective of this tutorial is to offer a comprehensive introduction to the field of adjoint optimization pertaining to the optimization of multiple parameters within differential equations and to provide an easily accessible framework—including computer code—to apply it to users' problems. To achieve this, we will elucidate the practical utility of the adjoint method, specifically focusing on its basic capacity to critically

¹Third Institute of Physics - Biophysics, Georg-August Universität Göttingen, 37077 Göttingen, Germany. ²BioQuant & Institute for Theoretical Physics, Heidelberg University, 69120 Heidelberg, Germany. ³Bernstein Center for Computational Neuroscience, Friedrich-Hund Platz 1, 37077 Göttingen, Germany.

e-mail: sherzog3@gwdg.de

evaluate the soundness of foundational model assumptions, but also the more indirect use-cases outlined above.

In the following, we start by giving a general overview of the different methods available for parameter estimation. Afterward, we focus on the adjoint method, which is introduced in theoretical detail in the section “A detailed look at the adjoint method”. In the subsequent section, “Practical computations and challenges of applications”, we show how this mathematically cumbersome framework can be applied easily to a number of different physical systems and inference tasks while highlighting challenges and providing code examples. To facilitate accessibility, we present a straightforward implementation of the adjoint method, referred to as adoptODE.

Overview of methods for parameter estimation

Parameter estimation and the concept of system identification have a long tradition. It is important to note that both have invariably progressed in tandem. Parameter estimation encompasses deducing the values of latent variables within a mathematical model, contingent upon empirical observations, a classical challenge known from mathematical optimization, where good parameters are usually defined by minimizing a meaningful distance between the model and certain data. Parameter identification, on the other hand, is an exalted quest for ascertaining the unique and unequivocal determination of good model parameters predicated upon the corpus of available empirical data. Koopmans and Reiersol¹⁸ first described the concept of identification of structural characteristics in 1950, while in 1956, Fisher¹⁹ introduced the idea of identifiability and Berman and Schoenfeld²⁰ addressed the actual identifiability problem. From this point on, the field has developed rapidly^{21–29} and there have been many publications to date, a detailed review of which can be found in ref. 30.

From a high-level perspective, parameter estimation is a search problem in which an arbitrary search space is searched for optimal solutions so that the given model generates an answer that matches the user’s data. The search process should be both effective, meaning it can find optimal solutions, and efficient, meaning it can find them quickly and with minimal resources. Based on these fundamentals and many other advances in the field of optimisation, there are nowadays an immense number of optimisation methods that can be used to fit parameters of differential equations. The general structure, of which many variations exist, can be defined as

Definition 1. (*Parameter estimation*) Assume the configuration of the system in question is defined by a state y that is evolving according to an equation of motion (EOM) f . This EOM depends on additional parameters p . Supplemented with possibly inaccurate initial conditions y_0 the system is given as a typical ODE problem

$$\frac{d}{dt}y(t) = f(y(t), t, p), \quad y(t=0) = y_0. \quad (1)$$

The task is to find parameters p minimising a real-valued loss function

$$L(y_{(\hat{t}_0)}, \dots, y_{(\hat{t}_N)}), \quad (2)$$

which depends on the state y of the system at a finite number of ascending evaluation time points $\hat{t}_0, \hat{t}_1, \dots, \hat{t}_N$, i.e. the times at which an experimental system was observed. To be found are the parameters p and possibly initial conditions y_0 such that the loss is minimal.

The different methods can be roughly divided into several groups, the advantages and disadvantages of which are listed in Table 1 as follows:

Likelihood-based methods are a class of statistical techniques used to estimate the parameters of a statistical model. The central idea is to maximize the likelihood function, which measures how well the model explains the observed data. This allows utilizing known uncertainties in the data and estimating the confidence of the resulting parameters. Within this category, we have the classical Kalman Filter³¹ and ensembles of augmented Kalman

filters, where the estimated variables are described by multidimensional normal distributions. These methods are particularly adept at estimating unknown parameters entangled within the observation error covariance matrix³². The expectation-maximization algorithm³³ offers a method for extracting maximum likelihood or maximum a posteriori parameter estimates within statistical models where latent variables play a crucial role. This can also be applied to the estimation of model parameters if one succeeds in formulating the system as a statistical model. Bayesian estimation makes use of Bayes’ theorem to continually update the probabilities of hypotheses as more evidence or information becomes available. Likelihood-based approaches are well-suited for parameter estimation in systems characterized by stochastic dynamics unfolding over significant time scales. In such scenarios, the rapid, microscopic degrees of freedom can be effectively characterized as stochastic noise. An illustrative instance arises in the realm of (bio)chemistry, particularly when investigating the equilibrium of a series of molecular reactions³⁴. Nevertheless, when the system dynamics undergo pronounced changes within short time intervals, as often observed in chaotic systems, the applicability of likelihood-based methods diminishes³⁵. Additionally, the effectiveness of likelihood-based methods hinges on the availability of a substantial dataset to ensure accurate estimation of expected values without distortion from data noise. Consequently, in scenarios with limited data, as exemplified by Jain and Wang³⁶, likelihood-based methods may not be the most suitable choice. The least-squares estimate is equivalent to the maximum-likelihood estimate when the error terms in the model follow a Gaussian distribution. This way, least-squares methods can be viewed as a simplification of likelihood-based approaches. The benefit lies in avoiding the necessity to transform the physical model into a static one. However, this simplification comes at the cost of forfeiting the capability for uncertainty quantification³⁷.

Least-squares methods try to minimize the squared (euclidean) distance between parameter-dependent model predictions and given observations, in its earliest version, used by Gauss to predict the location of the asteroid Ceres after not being observed for some time. Within this category, we find the Gauss–Newton Method, an optimization algorithm designed to address non-linear least-squares problems efficiently. It is essentially a modification of Newton’s method. However, for large problems with several tens of thousands of parameters, the need to calculate a second derivative is often a limiting factor³⁸. Another notable technique is the Levenberg–Marquardt algorithm, which serves as a numerical method for minimizing functions with nonlinearity. The method combines the Gauss–Newton method with a regularisation technique that enforces descending function values, making it more suitable for estimating parameters within higher-order nonlinear differential equations³⁹. Least-squares methods are easier to calculate than likelihood-based methods and can be applied more generally. As shown in ref. 40 several conditions, like the normality of the errors, the linearity of the model, and the independence and homoscedasticity of the observations, need to be fulfilled such that the validity and optimality for the estimated parameters are given. If these conditions are not met, least-squares methods are more likely to lead to biased and inconsistent results⁴¹ and also can be sensitive to noise and require a priori knowledge of the system structure⁴². This prevents a direct application for more complex, non-linear or even chaotic systems. However, least-square methods can be found in powerful approaches, as shown in refs. 43,44, where a sparse regression is utilized for system identification based on a user-defined set of nonlinear functions. As Bock et al.⁴⁵ demonstrate, nonlinear least-squares problems are often solved by simply coupling an integrator with an optimization procedure, leading to shooting approaches.

Shooting approaches are numerical methods to solve boundary value problems of ordinary differential equations. The basic idea of these methods is to trace the problem back to the solution of an initial value problem. This can also be applied to the estimation of model parameters. The approach is similar to aiming and launching a projectile towards a distant target, from where it borrows its name. The projectile is released at a specific initial angle, and by adjusting the trajectory based on the observed misses, the target is eventually hit. Shooting approaches are often used in the field of dynamical

Table 1 | Overview of advantages and disadvantages of the different classes of parameter estimation techniques discussed

Method	Advantages	Disadvantages
Likelihood-based methods	<ul style="list-style-type: none"> • Probabilistic framework allows inference and uncertainty quantification. • Can handle nonlinear and non-Gaussian models. 	<ul style="list-style-type: none"> • Require the specification of a likelihood function. • Possibly identifiability issues or multiple local optima. • Can be computationally intensive / require approximation techniques.
Least-squares methods	<ul style="list-style-type: none"> • Simple and intuitive to implement and interpret. • Can handle linear and some nonlinear models efficiently. • Widely used and well-studied in various fields. 	<ul style="list-style-type: none"> • Deterministic framework does not account for uncertainty or noise. • Sensitive to outliers or heteroscedasticity. • Requires a loss function, which might not reflect the true model.
Shooting approaches	<ul style="list-style-type: none"> • Can handle boundary value problems and differential-algebraic equations. • Can exploit the structure (e.g. constraints) and properties of the problem. • Often easy to implement and parallelize. 	<ul style="list-style-type: none"> • Require the solution of an initial value problem, hence sensitive to initial conditions or numerical errors. • Possibly instability or convergence issues.
Smoothing-based approach	<ul style="list-style-type: none"> • Can handle noisy or incomplete data. • Can reduce the effects of outliers or measurement errors. • Often improve fit and generalization of the model. 	<ul style="list-style-type: none"> • Require the specification of a smoothing parameter, which may not be easy to choose or estimate. • May introduce bias or overfitting in the model. • May not preserve some features or characteristics of the data.
Interior-point methods	<ul style="list-style-type: none"> • Can handle nonlinear and constrained problems. • Can exploit sparsity and structure in the problem. • Often converge faster and more reliably than other methods. 	<ul style="list-style-type: none"> • Require the solution of a sequence of subproblems, which may be costly or ill-conditioned. • May not scale well to large-scale or high-dimensional problems. • May require tuning of parameters or regularization techniques.
Simulation methods	<ul style="list-style-type: none"> • Can handle stochastic and dynamic models. • Can generate synthetic data and scenarios. • Often useful for testing and validation purposes. 	<ul style="list-style-type: none"> • Require the specification of a simulation model, which may not always be realistic or accurate. • May be affected by randomness or variability. • May not provide analytical or exact solutions. • Possibly computationally demanding.
Artificial neural networks	<ul style="list-style-type: none"> • Can approximate complex nonlinear functions. • Often achieve high performance and accuracy in various tasks. 	<ul style="list-style-type: none"> • Require a large amount of data and computational resources to train and evaluate. • May suffer from overfitting or underfitting issues. • May not be interpretable or explainable.
Adjoint methods	<ul style="list-style-type: none"> • Can compute gradients or sensitivities of an ODE model efficiently. • Scales well with large problems (High dimension, many parameters). • Can exploit the adjoint or dual structure of the problem. • Often useful for optimization or inverse problems. 	<ul style="list-style-type: none"> • Require the derivation and implementation of the adjoint equations, which may be complicated or tedious. • May be affected by numerical errors or instabilities. • May not be applicable to some types of problems or models.

systems, like population dynamics and chemical reactions⁴⁶. The efficacy of shooting methods is intricately tied to the selection of initial guesses and the level of noise present. Simultaneously, their implementation is straightforward, demanding only the capacity to solve the system of equations. With well-considered time intervals for parameter evaluation, these methods can extend their applicability to estimating parameters in chaotic systems. However, meeting this requirement is not always a straightforward task⁴⁷. The extension of (single) shooting methods is multiple shooting approaches like^{48,49}, which find application in the estimation of parameters within systems characterized by nonlinear differential equations⁵⁰. Multiple shooting methods are more robust and efficient but require more computational resources and an even better partition of the time interval⁴⁷.

Smoothing-based approach: The generalized smoothing approach⁵¹ represents a method for parameter estimation within models defined by systems of nonlinear differential equations. This method relies on the utilization of noisy measurements applied to a subset of variables to effectively estimate the parameters of the system. By employing this approach, the model adeptly manages the challenges posed by noisy measurements that

specifically impact a designated subset of variables. The application of smoothing techniques is prevalent in diverse fields, such as computational neuroscience, the modelling of flare dynamics, and various other domains where encountering noisy data is commonplace. This is particularly valuable in scenarios where robustness in the face of model misspecification is crucial, as highlighted in ref. 51. Moreover, the incorporation of quasilinearization methods introduces a strategic smoothing component, which is governed by a penalty derived from a quasilinearized ODE basis⁵². Despite the evident advantages, certain challenges are associated with smoothing-based approaches. One notable issue is the selection of an appropriate smoothing parameter or bandwidth, which plays a pivotal role in striking a balance between bias and variance. Opting for a smoothing parameter that is too small may lead to overfitting and heightened variance, while choosing one that is too large may result in underfitting and elevated bias, as discussed in ref. 53. In general, smoothing-based approaches necessitate a judicious combination with other methodologies to yield optimal results for parameter estimation. However, the selection of complementary approaches is not straightforward, making these methods less readily applicable out of the

box. The intricate task of determining how to effectively integrate them adds a layer of complexity, emphasizing the need for careful consideration in order to harness the full potential of these techniques.

Interior-point methods (or barrier methods) are algorithms for solving linear and non-linear convex optimization problems. To transverse the interior (rather than the boundary) of the feasible region in the search space nonlinear programming and iterative methods are used. Interior-point methods are related to smoothing-based approaches because they can be used to smooth the objective function and the constraints, and also to l_1 -regularized least-square problems⁵⁴. Interior-point methods were used in many domains, like control problems in computational fluid dynamics⁵⁵. The incorporation of interior-point methods and decomposition strategies can lead to methods that can be applied to large-scale problems, as shown in ref. 56. One drawback, however, lies in the complexity of applying this method out of the box. The interior-point method, while a powerful tool for solving nonlinear optimization problems, may encounter challenges in converging to a local minimum, even when the problem adheres to the Mangasarian–Fromovitz constraint qualification and satisfies second-order sufficient optimality conditions⁵⁷. Additionally, the method faces difficulties when dealing with ill-conditioned problems, large-scale problems, and problems characterized by multiple local minima⁵⁸. Navigating the intricacies of the interior-point method in the realm of inverse modelling poses an additional layer of difficulty. The selection of the barrier parameter, regularization parameter, and stopping criterion becomes a challenging task in itself⁵⁹. These choices significantly impact the performance of the interior-point method, requiring careful consideration and tuning for effective application in the domain of inverse modelling.

Simulation-based approaches (or metaheuristics): This diverse group comprises techniques like particle swarm optimization, which iteratively seeks to enhance candidate solutions by drawing inspiration from collective behaviour seen in bird flocking and fish schooling. It is designed to find global and optimal solutions within a given function⁶⁰. The sequential Monte Carlo method offers simulation techniques for computing the posterior distribution of a sequence of relevant parameters, including parameter estimation within non-linear differential equations for complex systems⁶¹. Differential evolution, on the other hand, is utilized for optimizing multidimensional real-valued functions by iteratively improving candidate solutions with respect to a predefined measure of quality. Finally, genetic algorithms with α -selection are heuristic search algorithms well-suited for adaptive systems, incorporating α -selection in the process of parameter estimation for non-linear differential equations within complex systems⁶². Theoretically, these methods are capable of solving the most complex problems, but depending on the nature of the problem, the demand for computer resources is very high and, in the worst case, can reach the effort of a brute force approach.

Artificial neural networks (ANNs) constitute a prominent subset within the domain of machine learning, drawing inspiration from the architectural and functional attributes of biological neural networks. In essence, ANNs are characterized by their inherent non-linearity and their capacity for data-centric modelling. These networks can effectively address both regression and classification tasks, thereby offering a unified approach to a spectrum of predictive and discriminative challenges. Within the realm of parameter estimation⁶³, presents an application of a continuous Hopfield network for the purpose of least squares minimization. This approach illustrates the capacity of ANNs in enhancing parameter estimation processes. Additionally⁶⁴, introduces a novel decomposition approach, wherein an ANN model is derived from the dataset and subsequently utilized for parameter estimation. This strategy serves to streamline the parameter estimation problem, offering a more accessible route to parameter determination⁶⁵. puts forth a modular neural network architecture designed specifically for dynamic system parameter estimation. This approach excels in terms of speed and noise immunity, showcasing the substantial advantages of employing ANNs in this context. Furthermore⁶⁶, introduces a modified Hopfield network to facilitate robust parameter estimation within nonlinear models marked by unknown-but-bounded errors and

uncertainties⁶⁷. shows the impact of different ANN architectures and loss functions and applies domain adaptation techniques to improve the accuracy of the estimated parameters. These studies collectively underscore the efficacy of ANNs in the realm of complex system parameter estimation. Furthermore, numerous studies, including^{68–71}, contribute to the burgeoning field of ANNs in parameter estimation, further substantiating the value of this methodology. However^{72,73}, underscore the restrictions inherent to ANNs with regard to the interpretation of their parameters in the context of physical significance. Their analysis accentuates that, notwithstanding the fact that ANNs do not necessitate a causal model, the parameters derived from ANNs lack inherent physical interpretations, and, frequently, the quantity of parameters to be ascertained within ANNs significantly surpasses the quantity of data instances available within the training dataset. One of the challenges in applying ANNs to complex physical systems is that the parameters of the ANNs are often difficult to interpret or relate to the underlying physical phenomena. Therefore, the main goal of ANNs is to achieve high predictive accuracy rather than to provide physical insights. However, this may limit the generalizability and robustness of the ANNs, especially when the data are scarce or noisy. To address this issue, approaches called physics-informed neural networks (PINNs) have been proposed. PINNs are ANNs that incorporate the physical laws that govern the system into the training process, thus enabling the ANNs to learn from both data and prior knowledge. The prior knowledge can, for example, be expressed by differential equations⁷⁴ or by stochastic differential equations⁷⁵.

Adjoint methods: The adjoint method¹⁵ is derived from the Lagrangian dual problem⁷⁶, with the main ideas going back to Pontryagin⁷⁷ and his associates in the formulation of Pontryagin’s maximum principle. The adjoint method expresses the gradient of a function with respect to its parameters in the context of constrained optimization and is constructed around the adjoint state. This adjoint state is used to compute the gradients by a backward pass, where the actual system and the adjoint state are solved in a negative time direction, illustrated in Fig. 1. Leveraging the dual formulation of this optimization problem enables efficient gradient computation, particularly for large numbers of parameters. In the domain of parameter estimation within multidimensional media, Rodriguez-Fernandez et al.²⁴ present an algorithm grounded in the adjoint method, demonstrating its prowess in determining parameters effectively. Likewise, Bhat⁷⁸ shifts the focus to system identification, applying the adjoint method to nonlinear, non-autonomous systems. Their work showcases the adjoint method’s capability to estimate parameters and states from noisy observations, revealing its utility in this field. Subsequently, Chen et al.⁷⁹ showed how to training infinitesimally layered neural networks by using the adjoint method for finding gradients to optimize ANNs used as equation of motion (EOM) of an ODE, resulting in neural ordinary differential equations. This concept was further elaborated in ref. 80, where a class of models called universal differential equations, which are differential equations that contain an embedded universal approximator, such as a neural network, as part of the equation, were introduced. One of the major advantages of the adjoint method is its linear scaling with the number of parameters to be estimated, making it a favourable method, especially for optimization problems in high-dimensional parameter spaces⁸¹. Additionally, it can readily be adapted to be used with PDEs and even stochastic differential equations. Note that there are also many more cases with specific applications, including refs. 82–84, but detailing this is beyond the scope of this paper.

A detailed look at the adjoint method

Arguably, the adjoint method is one of the most powerful tools for parameter estimation in large systems. However, its complexity arises from intricate mathematical subtleties. Therefore, our initial focus will be on the overarching aspects of the theory presented in abstract terms to preserve generality. Despite the seemingly complicated nature of the general formalism, its practical application proves to be straightforward. In fact, we devote the later practical computations section to showcase the concrete parameters estimation in five very distinct physical systems. To this aim, we rely on a specific implementation and a ready-to-use tool, with the aim of

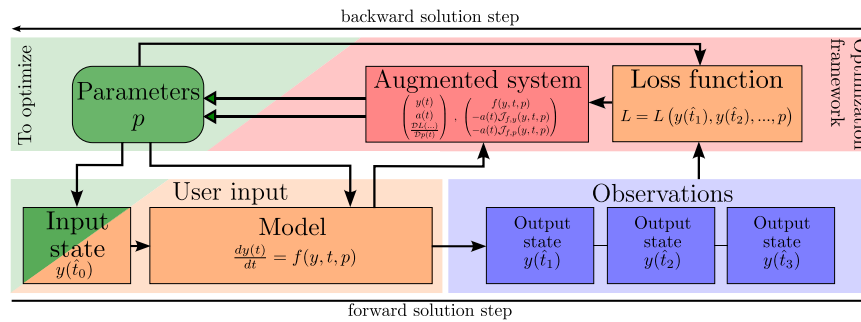


Fig. 1 | Diagram of the optimization procedure using the adjoint method. A prior-provided ordinary differential equation (ODE) model generates outputs depending on its parameters and the input state. By comparing to data and possibly additional constraints formulated in the loss function L , input states and parameters can be optimized. The augmented system (Def. 3) required for the optimisation of

the parameters is complicated but can be set up and solved automatically by frameworks such as adoptODE introduced below. The user input is coloured in orange, and the data used to find the desired parameters is constraining the model outputs coloured in blue.

providing a hands-on guide to parameter extraction accessible to non-specialists and provide a brief overview of the main ideas of the adjoint method here as an alternative to the following more technical introduction.

To optimize a loss function L that depends on the solution of a differential equation f evaluated at different times for parameters p of that equation (cf. Def. 1) is tricky; the parameters might influence the solution of f at different times in varying, non-obvious ways.

Starting with an initial guess of parameters, the differential equation f can be solved, and the loss function L evaluated, yielding the current loss value, which should now be minimised by adapting the parameters. First, the reaction of the loss function to changes in the solution of the equations needs to be understood. For the latest (in observation time) solution entering the loss function, this is a simple derivative. But if we change the solution at an earlier time, we have to account for subsequent changes at later times. This complex sensitivity of the loss function to changes in the solution is captured by the adjoint state (Def. 2). The observation that it is simple for the last observation time can be extended to a differential equation describing the adjoint state, but being solved backward in time, starting with the last observed time point.

Secondly, we need to describe how the solution of the differential equation f changes if parameters p are changed. This is essentially described by the Jacobians of equation f with respect to the parameters p . Because the adjoint state already includes how the change propagates to later times, we only need to consider how changing the parameter for a short time changes the solution at that time, which can be described by the jacobian.

The full response of the loss function to briefly changing parameters at a time can now be described by the product of jacobian and adjoint state at that time. Because we want to change the parameters at all times, we integrate over the time at which the parameters are changed, and to conveniently solve the adjoint state simultaneously (to save the memory of storing it), we do so backward in time as well, leading to the augmented system (Def. 3). A schematic of this optimization procedure is shown in Fig. 1.

Derivation of the adjoint method

Assume a problem set up as in Def. 1. Finding parameters to minimize the loss is non-trivial, because the state y , which enters the loss function, depends on parameters p implicitly by being a solution of Eq. (1). Furthermore, solutions at different times are connected. Therefore the different $y_{(t_i)}$ cannot be optimised independently. This problem has been studied in detail in ref. 77. To simplify notation, state y and parameters p are treated here as scalars. The generalisation to more dimensions is straightforward.

Defining two derivatives

To understand the adjoint method, it is helpful to distinguish two different ways of differentiating quantities depending on the solution of an ODE. Taking, for example, the loss function $L(y_{(t_0)}, \dots)$, its derivative by the state

of the system at time t can be understood in different ways:

$$\text{Momentary: } \frac{\partial L(y_{(t_0)}, \dots)}{\partial y(t)}, \tag{3}$$

$$\text{Lasting: } \frac{\mathcal{D}L(y_{(t_0)}, \dots)}{\mathcal{D}y(t)}. \tag{4}$$

The momentary derivative at t is the change in loss if the system state is varied, but only at time t , and for all future times reversed to its previous value, analogous to the functional derivative from variational calculus with the simplification of using a delta distribution in time as a test function. For the example of the loss function, this means that the momentary derivative by the state $y(t)$ is vanishing, except t is one of the observation times at which the loss function explicitly depends on the state. The lasting derivative, on the other hand changes $y(t)$ and keeps the introduced change for all following times. For the derivative of the loss function by the state $y(t)$, the change of y at t is propagated by the time evolution to all observation times $t_i > 0$, thereby entering the loss. In the case of deriving by the parameters p instead of the state, this means a change in parameters is performed at t , which induces a slight deviation in the state y , which for the momentary derivative is propagated with the old parameter value, and for the lasting derivative with the changed value of p . The lasting derivative with respect to the state y allows us to define the central quantity of the adjoint method:

Definition 2. (Adjoint state) The adjoint state $a(t)$ is defined as

$$a(t) = \frac{\mathcal{D}L(y_{(t_0)}, \dots, y_{(t_N)})}{\mathcal{D}y(t)}. \tag{5}$$

It describes the reaction in loss if, at time t , the system's state is changed, with the changed state afterward propagated to affect all later states entering L . Intuitively, if an external influence would perturb the system at time t , the subsequent time evolution would produce a different trajectory, changing the later states $y_{(t_i)}$ entering the loss function. The adjoint state measures this change in loss as a function of the time t and the perturbation direction.

Adjoint method

In this framework, the required gradients g to solve the optimisation problem can be understood as the lasting derivative of L with respect to the parameters p at time t_0 . Using an integral, we can express these gradients in

terms of the easier-to-handle momentary derivative,

$$g = \frac{\mathcal{D}L(y_{(\hat{t}_0)}, \dots)}{\mathcal{D}p(\hat{t}_0)} = \int_{\hat{t}_N}^{\hat{t}_0} dt \frac{\partial L(y_{(\hat{t}_0)}, \dots)}{\partial p(t)}. \quad (6)$$

This integral will later be evaluated backward in time. Conceptually, this is because we know what happens if the parameters are changed at the end point $t = \hat{t}_N$, namely nothing, as the system has no time anymore to react to a change. From this known start point, the gradients can be determined by moving the time of parameter change towards the initial time \hat{t}_0 . The momentary derivative can be expressed by the chain rule as the product of the adjoint state and a Jacobian. This reflects the fact that momentarily changing the parameters introduces a change in the system's state, which—after the parameters are changed back—propagates as defined for the lasting derivative. The Jacobian of the EOM f , $\partial f / \partial p = \mathcal{J}_{f,p}$, describes the change in state caused by the parameter change.

$$\frac{\partial L(\dots)}{\partial p(t)} = \frac{\mathcal{D}L(\dots)}{\mathcal{D}y(t)} \cdot \mathcal{J}_{f,p}(y, t, p) = a(t) \cdot \mathcal{J}_{f,p}(y, t, p) \quad (7)$$

Combining Eq. (6) and Eq. (7), we obtain an expression for the time derivative of the gradient g ,

$$\frac{d}{dt} g(t) = -a(t) \frac{\partial f(y, t, p)}{\partial p}. \quad (8)$$

The desired gradients are $g = g(t_0)$, and the initial condition for solving this backwards is $g(t_N) = 0$.

The missing piece is now the adjoint state, or an equation allowing us to derive it. For this, we first consider one step between t and $t + \epsilon$ with ϵ small, such that none of the evaluation times \hat{t}_i are passed within this step, as their influence has to be considered separately. Then the following holds:

$$\begin{aligned} \frac{\mathcal{D}L(\dots)}{\mathcal{D}y(t)} &= \frac{\mathcal{D}L(\dots)}{\mathcal{D}y(t + \epsilon)} \frac{\mathcal{D}y(t + \epsilon)}{\mathcal{D}y(t)} \\ &\Rightarrow a(t) = a(t + \epsilon) \frac{\mathcal{D}y(t + \epsilon)}{\mathcal{D}y(t)}. \end{aligned} \quad (9)$$

Elementary algebra on the differential quotient yields the desired evolution of $a(t)$:

$$\begin{aligned} \frac{d}{dt} a(t) &= \lim_{\epsilon \rightarrow 0} \frac{a(t + \epsilon) - a(t)}{\epsilon} \\ &= \lim_{\epsilon \rightarrow 0} \frac{a(t + \epsilon) - a(t + \epsilon) \frac{\mathcal{D}y(t + \epsilon)}{\mathcal{D}y(t)}}{\epsilon} \\ &= \lim_{\epsilon \rightarrow 0} \frac{a(t + \epsilon) \left(1 - \left(1 + \mathcal{J}_{f,y}(y, t, p) \epsilon \right) \right)}{\epsilon} \\ &= -a(t) \cdot \mathcal{J}_{f,y}(y, t, p). \end{aligned} \quad (10)$$

The significance of the condition of not passing the evaluation time lies with the adjoint state $a(t)$. Defined as the change of the loss function while changing the solution at time t , it can be expanded as

$$\begin{aligned} a(t) &= \frac{\mathcal{D}L(y_{(\hat{t}_0)}, \dots, y_{(\hat{t}_N)})}{\mathcal{D}y(t)} \\ &= \sum_{i=0}^N \frac{\partial L(y_{(\hat{t}_0)}, \dots, y_{(\hat{t}_N)})}{\partial y_{(\hat{t}_i)}} \frac{\mathcal{D}y_{(\hat{t}_i)}}{\mathcal{D}y(t)} \\ &= \sum_{i=0}^N \mathcal{J}_{L, y_{(\hat{t}_i)}}(y, \hat{t}_i, p) \frac{\mathcal{D}y_{(\hat{t}_i)}}{\mathcal{D}y(t)}. \end{aligned} \quad (11)$$

Because $\mathcal{D}y_{(\hat{t}_i)} / \mathcal{D}y_{(t)}$ is zero for $t > \hat{t}_i$ and becomes unity at $t = \hat{t}_i$, this discontinuity has to be manually incorporated into the $a(t)$ evolution, meaning $\mathcal{J}_{L, y_{(\hat{t}_i)}}(y, \hat{t}_i, p)$ has to be added when t passes \hat{t}_i . Additionally, we find $\mathcal{J}_{L, y_{(\hat{t}_N)}}(y, \hat{t}_N, p)$ as initial value $a(\hat{t}_N)$.

Augmented system

With the obtained results, we can now approach the gradients g . After solving the equation for the state $y(t)$ forward, the gradients can be computed by solving first Eq. (10) for the adjoint state and then Eq. (8), both backward. However, this requires storing the system state y and the adjoint state a at a large number of intermediate times to have them available while solving Eq. (8), which is impractical due to high memory consumption. It is much more efficient (in terms of memory) to compute all three quantities simultaneously in a single backward pass of what is called the augmented system:

Definition 3. (*Augmented system*) The adjoint sensitivity method constructs the gradients required for directed optimization, $\mathcal{D}L(\dots) / \mathcal{D}p(t)$, within a larger system, called the augmented system with state $s(t)$, including the adjoint state $a(t) = \mathcal{D}L(\dots) / \mathcal{D}y(t)$.

$$s(t) = \begin{pmatrix} y(t) \\ a(t) \\ \frac{\mathcal{D}L(\dots)}{\mathcal{D}p(t)} \end{pmatrix}, \quad (12)$$

$$\frac{d}{dt} s(t) = \begin{pmatrix} f(y, t, p) \\ -a(t) \mathcal{J}_{f,y}(y, t, p) \\ -a(t) \mathcal{J}_{f,p}(y, t, p) \end{pmatrix}, \quad (13)$$

where the initial condition follows from the aforementioned as

$$s(\hat{t}_N) = \begin{pmatrix} y_{(\hat{t}_N)} \\ a(\hat{t}_N) = \mathcal{J}_{L, y_{(\hat{t}_N)}}(y_{(\hat{t}_N)}, \hat{t}_N, p) \\ \mathcal{J}_{L, p}(y_{(\hat{t}_N)}, \hat{t}_N, p) = 0 \end{pmatrix}. \quad (14)$$

This system can be solved backward by solving the different co-dependent quantities y , a and L simultaneously, which is possible as their respective EOMs are only dependent on each other evaluated at the current time t . An exception is the system states at observation times \hat{t}_i , which are required to update the adjoint state at observation times. They can be saved during the forward pass, which is required anyways to obtain $y_{(\hat{t}_N)}$. As only the system state and only at the observation times have to be saved, the augmented system significantly reduces memory consumption in comparison to saving the state and the adjoint state at all the intervals required by the ODE solver. Another advantage is that at the final value of the adjoint state, $a(\hat{t}_0)$, gradients are available to optimize the initial conditions.

Properties of the adjoint method

In the form of the augmented system the adjoint method provides a powerful tool for solving the complex problem of obtaining the gradients g . The memory cost is linear in the dimension of state y and parameters p . This is crucial when dealing with complex systems because evaluating the sensitivity of the cost function to each parameter individually can be computationally expensive. The adjoint method provides a way to compute these sensitivities simultaneously, significantly reducing computational costs.

The main computation necessary is to solve the ODE of the augmented system (Eq. (12)), which—between the observation times—can be done by calling any standard ODE solver. Besides utilising a large base of well-developed ODE solvers in general, some allow tuning their accuracy, offering major performance increases if mediocre accuracy is sufficient and vice versa. Additionally, fitting solvers and their accuracy can be chosen for challenging numerical integrations appearing in many complex systems. Care has to be taken at each observation time \hat{t}_i , as the update of the adjoint

state is a non-smooth operation, which has to be performed manually outside of the ODE solver. Additionally, solving the system itself backward can introduce numerical problems, such as instabilities, not present in the forward pass, as is the case for diffusive systems. An easy workaround at the cost of more memory is to save additional backup states during the forward pass, as discussed in the Supplementary Methods.

Practical computations and challenges of applications

A neat theory often faces an uphill climb of practical inconveniences to be useful in any real-world application. A straightforward, literal realization of the theory developed above includes the following steps:

1. Implementing and solving the EOM f to solve the forward problem and obtain the estimated solutions at the observation times, $y_{(i_0)}, \dots, y_{(i_N)}$.
2. Implementing the loss function L , dependent on the observed data and its gradients $DL(\dots)/Dy(\dot{t}_i)$ which can be evaluated at the previously computed estimated solutions to generate the initial conditions and updates for the adjoint state a .
3. Implementing the Jacobians $\mathcal{J}_{f,y}(y, t, p)$, $\mathcal{J}_{f,p}(y, t, p)$ necessary for the dynamic equation of the adjoint state.
4. Piecewise solving of the augmented system in the negative time direction, updating at observation times, delivering at $t = t_0$ the gradients for the parameters and as the adjoint state $a_{(i_0)}$ gradients to the initial conditions.
5. Updating the estimate of parameters and initial conditions by a gradient decent scheme of choice.

Once all the implementing is done, this provides a clear, iterative scheme to minimize the loss by directed optimization. However, even for simple systems, deriving and implementing the required gradient and Jacobian functions is complicated and a source of error, whereas, for larger systems, it becomes completely unpractical. Fortunately, the tools of auto-differentiation, advanced by a large interest from the artificial intelligence and machine learning community, can be utilized to internally arrive at fast, efficient methods for computing the gradients and Jacobians, requiring only the EOM and loss function as user input! To facilitate accessibility, we present a straightforward implementation of the adjoint method, referred to as adoptODE. AdoptODE uses the library JAX⁸⁵ and its auto-differentiation capability in that way and additionally handles the error-prone details of the backward solution execution, including updating the adjoint state and other subtleties. At its heart, as default, it uses the JAX integrated odeint routine, which uses a mixed 4th/5th order Runge–Kutta scheme with Dormand–Prince adaptable step sizing⁸⁶, but other JAX compatible solvers (as, for example, provided by diffrax⁸⁷) can be passed as option. An additional capability of adoptODE concerning the uncertainty in estimated parameters is to easily simulate data for a given system, which can be used to gauge the reliability of the recovery.

For the example of adoptODE, we present capabilities, arising challenges and workarounds for increasingly complex physical systems. The code of all these examples is provided, both as a starting point for adopting the method to new problems and as a reference for details and parameter choices we will not extensively discuss. We emphasise that the code used for this is rather simple and can easily be adapted to address other specific problems too. (The complete framework adoptODE can be found at: <https://gitlab.gwdg.de/sherzog3/adoptODE>, also, a hands-on guide to parameter estimation with adoptODE is provided in Box 1).

The examples considered here start with simple ODE problems, where we first address interactive N-body systems (A) and, in a second step, the Lotka–Volterra model (B) as an example of a simple Kolmogorov model, where the influence of wrong model assumptions and noise are reviewed. Afterward, we move towards problems involving partial differential equations, where the relevant fields have to be discretized to apply the adjoint method, leading to significantly larger state spaces. Here, we first consider the Bueno–Orovio–Cherry–Fenton

(BOCF) model for parameter recovery (C), and afterward, Rayleigh–Bénard Convection (D), where an unknown field, the temperature, can be extracted from velocity data only. Lastly, we move on to new experimental data of Zebrafish embryogenesis (E): This is a case where we extend existing models with a discretized continuous field as parameters to recover, by which novel unexpected biophysical results could be obtained. We define for all scenarios the initial state, the assumed EOM with its parameters, the input data, ask the system to solve a specific task and outline the practical challenge this problem poses.

Interactive N-body problem

The N-body problem, as a first example, is a dynamical system consisting of particles, which influence each other by physical forces. Here we consider the case where N point masses interact via some pairwise force F .

State: Positions and velocities of all bodies.

EOM :

$$\frac{d^2x}{dt^2} = \frac{1}{m_i} \sum_j F(x_i - x_j).$$

Challenge: The parameters can only be estimated by their action on the system. In this example, we illustrate extreme cases: First, in gravitational interactions, most masses only enter perturbatively over long times. In contrast, for colliding spheres, the interaction is concentrated at the very short collision events.

Sub-problem 1: Gravitational systems: The first problem is based on Newton's law of gravity, which is used to describe the orbits of the planets in the solar system.

Parameters: Masses of the planets up to Saturn and of the sun.

Input data: From NASA's Horizon Systems⁸⁸, the trajectories of the sun, the planets up to Saturn and as additional force probes, the asteroids Apophis and Eros, sampled every 20 days throughout the 20th century.

Task: Infer model parameters, here mass, from trajectories.

Results: As the solar system is dominated by the sun's mass, the planetary masses can only be inferred from the perturbations they cause. Recovery is still successful, as shown in Fig. 2a, with the largest deviation for Mars at a relative error of 0.3 and <1% error for Saturn, Jupiter, and the sun.

Sub-problem 2: Repulsive systems: The second example demonstrates repulsive interactions of spherical particles with different effective radii, described by the Lennard–Jones potential⁸⁹.

Parameters: Mass and radius of each particle.

Input data: For every number of bodies, random radii and masses were drawn from the intervals [0.2,0.7] and [1,10], respectively. The spheres were randomly assigned positions on a grid (to avoid initial overlaps in high-density runs) with initial velocity drawn from a truncated normal distribution over [−2,2].

Task: As above, but also: infer non-observable variables from observable data. Here masses and radii are obtained from the trajectories, where (different from the gravitational interaction above) this information is only contained in the collision events.

Results: In Fig. 2b, it can be seen that for all but two masses at the largest system, the recovery of masses and radii was within 1% accuracy, taking the smallest value for masses and radii as a reference.

Lotka–Volterra model

The Lotka–Volterra model^{90,91} represents a system of two first-order non-linear, coupled differential equations. It describes the interaction of two species, one prey and one predator, which influence the time evolution of their respective populations by predator–prey interactions controlled by four parameters, $\alpha, \beta, \delta, \gamma$. Additionally, wrong model terms that are not necessary to explain the generated data are included to demonstrate that the correct terms can be identified. The Lotka–Volterra model is an example of

Box 1 | The AdoptODE Cookbook

0. Install AdoptODE and JAX

AdoptODE and its dependencies are required to run the following notebook, together with an updated version of the JAX library. The package enables GPU calculations, provided a supported CUDA driver is required. An installation guide is provided in the git-repository, <https://gitlab.gwdg.de/sherzog3/adoptODE.git>. In the repository, the executable of this cookbook and further resources are available.

1. Define your System

Our example system is $\frac{d}{dt}pop = a \cdot pop + b$, where pop is some scalar population and a and b are the parameters we want to find. We assume the initial population, a and b to be bounded below by zero and above by some maximum specified in `kwargs_sys`.

```
import numpy as np
import jax.numpy as jnp
from jax import jit
def define_system(**kwargs_sys):
    p_max = kwargs_sys['p_max']
    a_max = kwargs_sys['a_max']
    b_max = kwargs_sys['b_max']

    def gen_y0():
        ini_pop = np.random.rand()*p_max
        return {'population':ini_pop}

    def gen_params():
        a = np.random.rand()*a_max
        b = np.random.rand()*b_max
        return {'a':a, 'b':b}, {}, {}

@jit
def eom(y, t, params, iparams, exparams):
    pop = y['population']
    a, b = params['a'], params['b']
    return {'population':a*pop+b}

@jit
def loss(ys, params, iparams,
        exparams, targets):
    pop = ys['population']
    t_pop = targets['population']
    return jnp.mean((pop-t_pop)**2)

return eom, loss, gen_params, gen_y0, {}
```

The second and third dictionary of `gen_params` are `iparams` and `exparams` we do not have in this simple example. The first two functions

can be arbitrary, the `eom` and `loss` functions have to be implemented using the JAX libraries.

2. Set up a simulation

To set up a simulation we define the dictionaries `kwargs_sys` and `kwargs_NODE` as well as the times `t_evals` at which we assume to observe our system. The keyword '`N_sys`' gives the number of copies in terms of multi-experiment fitting, here we consider only one system.

```
from adoptODE import
    simple_simulation, train_adoptODE
kwargs_sys = {'p_max': 2, 'a_max': 1,
              'b_max': 3, 'N_sys': 1}
kwargs_adoptODE = {'lr': 3e-2, 'epochs': 100}
t_evals = np.linspace(0, 5, 10)
dataset = simple_simulation(define_system,
                            t_evals,
                            kwargs_sys,
                            kwargs_adoptODE)
```

3. Train a simulation

The easy following command trains our simulation and prints the true params in comparison to the found ones:

```
_ = train_adoptODE(dataset)
print('True params: ', dataset.params)
print('Found params: ', dataset.params_train)
```

4. Including Data

To include data, we bring it in the same form as the shape of the state given by `gen_y0()`, but with two additional leading axes. The first counts the different experiments, and has length one here, the second runs over time points and has the same length as `t_evals`.

```
from adoptODE import dataset_adoptODE
data # Observation of population, shape (10,)
targets = {'population':data.reshape((1,10))}
dataset2 = dataset_adoptODE(define_system,
                             targets,
                             t_evals,
                             kwargs_sys,
                             kwargs_adoptODE)
```

The new `dataset2` generated by the `dataset_adoptODE` method can be trained with analogous commands as in point 3 above. However, in this case, we don't have access to the true parameters for comparison, as in the usual application the true parameters are not known by different approaches, and only accessible by parameter estimation techniques.

the Kolmogorov model class to characterise continuous-time Markov processes.

State: Current population of prey (x) and predators (y).

EOM:

$$\frac{dx}{dt} = ax - \beta xy [+c_1 x^2], \quad \frac{dy}{dt} = -\gamma y + \delta xy [+c_2 y^2],$$

where the c_i terms are additional terms added to the pure model in recovery to demonstrate that adoptODE can be used to identify correct terms in model development.

Challenge: The true model is unknown in many real-world applications. Here, we include additional higher-order terms in the EOM, the weights of which have to be recovered to zero in order to recreate the correct system.

Parameters: $\alpha, \beta, \gamma, \delta \in \mathbb{R}^+$ and $c_1, c_2 \in \mathbb{R}$.

Input data: 500 trajectories with random parameter values and initial populations drawn from $[0.1, 1]$ were generated at each noise level with the pure model ($c_1 = c_2 = 0$). Each observed datapoint was perturbed by random noise with amplitude proportional to its value and the given relative noise amplitude.

Task: Infer model parameters, here $\alpha, \beta, \gamma, \delta$, and the initial conditions for x, y from observed noisy trajectories. The added terms should be quantified as insignificant for the system by finding close to zero values for the parameters c_1, c_2 .

Results: The mean absolute errors of parameter recovery are, on average, <0.1 for all noise amplitudes (Fig. 2c), indicating that parameter recovery was successful. At large noise levels, the error sometimes exceeded 1, signifying parameters out of range. In practice, such noise levels will likely not play any major role. The values for the added parameters c_1, c_2 are

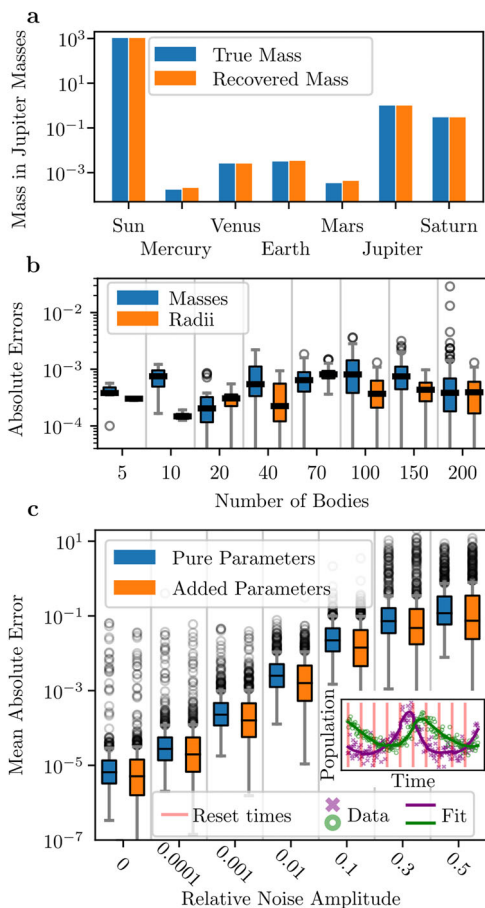


Fig. 2 | Parameter recovery for simple ODE systems. **a** The solar system up to Saturn is an example of attractive N-body interactions, planet masses unknown, truth in blue and recovery in orange. **b** Colliding spheres in a box, masses (blue) and radii (orange) of the spheres unknown. **c** The Lotka–Volterra model, a simple predator–prey system, showing statistics over 500 randomly generated systems for different noise levels. The absolute errors were averaged over the four parameters of the pure model (blue) and the two added parameters (orange). For the added parameters, the MAE is the mean absolute value, as the ground truth is always zero. The inset shows an example trajectory with reset times, at which the system is reset to new, simultaneously trained initial conditions.

similar to the errors of the pure parameters, meaning they are orders of magnitude smaller than the pure parameters for lower noise ratios, allowing to reject the respective terms as irrelevant for the model.

Bueno–Orovio–Cherry–Fenton model

The Bueno–Orovio–Cherry–Fenton (BOCF) model is a minimal ionic model for human ventricular cells in the heart. It shall be considered as an example of excitable media. In general, an excitable medium is a nonlinear dynamical system to describe the propagation of some waves, which can influence each other, but where one wave cannot propagate through the other. Such systems are used for modelling of many bio-chemical domains. In the context of the BOCF model, the field u serves to characterize the transmembrane potential, giving rise to propagating waves of action potentials. The remaining three fields, termed gating fields, delineate the tissue’s condition, e.g. the intracellular calcium level. The parameters of these gating fields encode essential features, like the refractory period, during which a specific location becomes inactive after the passage of an action potential. This characteristic is pivotal for the formation of patterns like spiral waves.

State: Four fields (voltage and three gating variables), each sampled at a 512×512 square lattice, resulting in one million coupled ODEs.

EOM: The equations of the BOCF model⁹².

Challenge: Many systems containing, for example, diffusion become heavily unstable if solved in a negative time direction, as necessary for the backward pass of the adjoint method. This can be countered by additional backup states generated during forward passes but must be carefully managed for large systems to control memory usage⁸¹. Here, we use a scheme where some points are saved during the forward pass, but additional backup states can be generated during the backward pass by forward propagation from the closest stored point. Additionally, the parameters of the BOCF model span many orders of magnitude, hence recovery benefits from logarithmic rescaling.

Parameters: The parameters of the BOCF model control the interactions between the membrane potential and the gating fields, allowing for example, to match the shape of the propagating action potentials. However, the parameters influence each other, so different sets of parameters can cause the same dynamics of the model. To ensure comparability between the ground truth parameters used to generate our data, and the recovered parameters, we first identified the 10 most influential parameters by separately perturbing them by a factor of 1.5 and comparing the resulting mean absolute errors. The 10 parameters out of the 28 parameters with the highest resulting mean absolute error were then selected for the recovery task. These 10 parameters were then perturbed by random factors between 1/2 and 3/2 and recovered by adoptODE.

Input data: A procedure of repeated excitation generates a random initial state showing spiral-wave patterns. From this initial state, a trajectory with 100 time points over 50 ms is generated.

Task: (1) infer non-observable variables from observable data, (2) validate model assumptions. Here: Find parameter values to best describe a given trajectory.

Additionally, we show how to use adoptODE to reproduce the BOCF model excitation field using the simpler Aliev–Panfilov (AP) model⁹³, a model with only six free parameters and two fields. Because the form of the excitation front and the single gating field in the AP model are different from the BOCF model and its three gating fields, both initial fields have to be reconstructed simultaneously with the parameters, adding $\approx 5 \times 10^5$ degrees of freedom to the optimization. Taking this into account, adoptODE finds parameters for which AP reproduces the BOCF dynamics, even if the prediction breaks down faster than the BOCF recovery.

Results: To demonstrate the versatility of the recovery for the BOCF system, we analyse the growing discrepancy between the true and the recovered system with time, where the recovery was once calculated using the BOCF model, and again with the AP model where additionally the initial fields had to be recovered. Note that for the investigated parameters, the BOCF model exhibits chaotic behaviour (maximum Lyapunov exponent $\lambda_{\max} \approx 0.00276$), i.e. even small deviations lead to an exponentially increasing error. For the BOCF model, only a very short time (50 ms) had to be observed for a meaningful recovery. Measured by the mean absolute error (Fig. 3c) and the mean error of peak occurrence averaged over all pixels (Fig. 3d), recovery was largely valid for up to 2 s. The errors at 1.5 s were mostly small differences in the propagation of individual excitation fronts and only in the upper right area already larger deviations (Fig. 3a and b) were found. For the AP model, 0.5 s were used for training. Over this period the recovery is surprisingly good, considering the AP model cannot truly capture the BOCF dynamic. Here, the optimisation probably uses the fit of the initial conditions for the gating field to compensate for the different models, leading to a quick divergence after the observed time (cf. Supplementary Fig. 1 displaying the resulting initial fields for the AP model). The recovered parameters for the BOCF model had a mean absolute deviation of 0.76% and a median absolute deviation of 0.4%, the recovered values are shown in Supplementary Table 1.

Rayleigh–Bénard convection

Rayleigh–Bénard convection is an example of self-organising nonlinear chaotic systems in the field of fluid thermodynamics. A fluid enclosed in a volume, the bottom of which is heated, creates convective

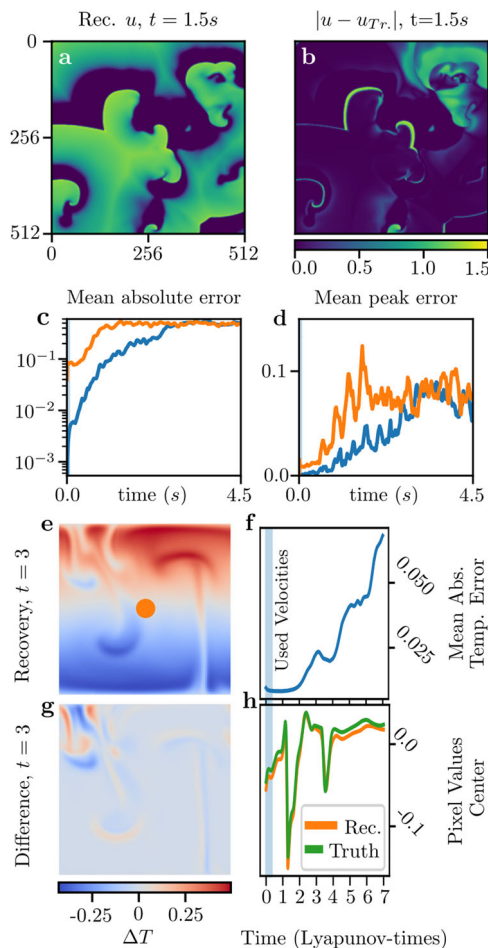


Fig. 3 | Bueno-Orovio–Cherry–Fenton (BOCF) model and Rayleigh–Bénard convection as examples for applying the adjoint method to discretized partial differential equation (PDE) systems. Recovery of parameters (in the BOCF model, a–d) and initial conditions of fields (in Rayleigh–Bénard convection, e–h). a, b Excitation field u , value given by colour bar below (b). a Recovered BOCF field after 1.5 s. b Difference between recovery in a and target. c Mean absolute error of the recovery using the BOCF model (blue) and the simpler Aliev–Panfilov (AP) model (orange) on a logarithmic scale, the length of the input sequence (50 ms) used to optimise the parameters via adoptODE is shown in blue. d Error in peak occurrence time averaged over all pixels. See Supplementary Movie 1. e Temperature perturbation. f Mean absolute error between recovered and true temperature perturbation. The length of the input sequence used to optimise the parameters via adoptODE is shown in blue. g Difference from ground truth, see also Supplementary Movie 2. h Pixel value of centre pixel for recovery (orange) and ground truth (green).

flow patterns conveying heat from the lower to the upper plate. Such convection cells are an example for the formation of dissipative structures far from thermodynamic equilibrium. The structure of these convection patterns is governed by the dimensionless Rayleigh and Prandtl numbers.

State: Temperature perturbation T and 2D velocity field \vec{u} , sampled at a 100×100 square lattice with periodic boundary conditions from right to left.

EOM: Equations for Rayleigh–Bénard Convection in the Oberbeck–Boussinesq approximation^{94,95}, including a prescription to obtain the pressure field p necessary in every step to maintain an incompressible flow⁹⁶.

Challenge: In many real-world scenarios, part of the model relevant quantities are not observed. Here we demonstrate that the adjoint method can recover such missing information. In the concrete case, the temperature field is recovered by reconstructing its initial condition only using

information on the velocity field. Due to incompressibility, the velocity depends in a complicated, non-local manner on the temperature perturbation, posing a challenging problem.

Parameters: Rayleigh and Prandtl numbers are assumed to be known, initial temperature field is unknown.

Input data: Trajectories obtained from the simulation at Prandtl-Number 7.0 and Rayleigh-Number 10^7 , but only the velocity field is available to the system for training, as would be the case in data from Particle Image Velocimetry (PIV⁹⁷) experiments. Measured in Lyapunov-times τ estimated from the same data, 0.35τ was used for training.

Task: (1) infer non-observable variables from observable data, (2) validate model assumptions. Here: Reconstruct the temperature field from the velocity field.

Results: Similarly to the BOCF Model, we validate the practicality of the recovered temperature field (Fig. 3e) by its power to predict the system longer than observed for training. The difference between the ground truth data and the recovered temperature field is shown in (Fig. 3g). We find the recovery to be accurate for around 2τ (Fig. 3f) and still qualitatively correct at 3τ (Fig. 3f and h) before it begins to more strongly deviate from the truth at 7τ . This behaviour is best seen in Supplementary Movie 2.

Zebrafish embryogenesis

The case of zebrafish embryogenesis shows the use of adoptODE on real experimental data. Zebrafish embryos, genetically modified to show fluorescent nuclei, were imaged in a light-sheet microscope from 5 to 7.5 h after fertilisation (hpf). The lightsheet scans the whole embryo once every 180 s and generates a full 3D image from which the position of every cell can be extracted. Afterwards, cells can be tracked from one frame to the next using Gaussian mixture models implemented in the library TGMM⁹⁸, resulting in branching trees of trajectories, as cells do actively divide during the time observed. The trajectories obtained in this way are not perfect and contain noise due to the experimental setup. To apply a continuous hydro-dynamical flow model supplemented with active stresses, from the trajectories, a mean velocity field was approximated by binning differences in cell positions between frames. The embryo develops on one pole of its spherical yolk cell. Around 5 hpf this cap spreads around the yolk cell in a process called epiboly, which coincides with and is required for the development of the germ layers⁹⁹. To study the influence of active processes on this spreading, a simplified two-dimensional Navier–Stokes flow on a sphere is assumed for the embryo on-top of the yolk and supplemented by a 2D time-dependent vector field of active forces.

State: 2D velocity field on a spherical shell, representing the mean tissue movement of the embryo along the yolk (cf. Fig. 4b).

EOM: A Navier–Stokes model for the flow in curved geometry, including a field of active forces interpolated between certain positions and times.

Challenge: Often, there are known blanks in models, where an additional, but not precisely known effect influences the system. In an extreme case, this effect can be varying in time and space, as the active forces generated by cells in the tissue of a zebrafish embryo in this example. The favourable scaling of the adjoint method with large numbers of parameters allows using a model-free, interpolation based description of such missing effects by discretizing them in time and space. Thereby, unbiased estimates of the additional effects can be obtained. As a consequence, the complexity and large number of degrees of freedom introduced by this additional field require some form of regularisation, especially if one wants to work with experimental data and a common problem for many complex systems. This is possible in the adjoint method and the implementation in adoptODE, too, because the loss function can be freely augmented by the required regularisation terms. In the case of the zebrafish embryo, an example for such a term is the L2-Norm of the fitted stresses, to prevent the usage of large stresses to fit noise in the data.

Parameters: The 2D active force vector at every position and time used for interpolation, in this case, around 5000 parameters. Additional

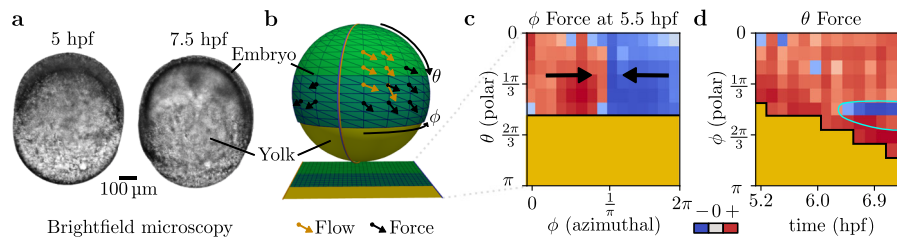


Fig. 4 | Describing active forces in zebrafish embryogenesis. **a** Brightfield images of different development stages (measured in hours past fertilization, hpf) of the embryo (dark) in the process of enveloping the yolk. Scale bar 100 μm . **b** Schematics of the model, the embryo is described on a two-dimensional spherical cap. The state is given by the flow at each grid point, which is determined by the forces encapsulated in the trained parameters. **c** and **d** Colour maps of force components, red positive,

blue negative, the black line marks the border between embryo and yolk (yellow). **c** A snapshot of the ϕ component of the force at 5.5 hpf over the spherical surface of the embryo. Black arrows illustrate the force direction of red and blue regions. **d** θ component of the force averaged over the azimuthal angle ϕ mapped against the polar position θ and time. Turquoise highlights a region of negative, hence up toward the pole-directed force at later times.

parameters per measurement allow the optimisation to rotate and time-shift measurements with respect to each other.

Input data: Mean velocity field of 12 measured zebrafish embryos with 50 time points each. All fields are aligned in order to facilitate a universal stress tensor field describing all different measurements simultaneously.

Task: (1) infer non-observable variables from observable data, (2) validate model assumptions, (3) extend existing models with new variables to better fit the data to the model assumptions.

Results: In this experimental setup, where the data was collected by Brightfield microscopy (an example is shown in Fig. 4a), the ground truth is unknown. Hence the results are compared to well-known details of zebrafish embryogenesis. However, the force distributions, as illustrated in Fig. 4b), obtained from the simple model not only plausibly explain the spreading via a strong force at the leading embryo edge (Fig. 4d) but also suggest two known deviations from a uniform spreading: Firstly, the ϕ force shows a strong dipole where cells converge at azimuthal position $\phi = \pi$ (Fig. 4c), yielding a thickening of the spreading embryo tissue known as shield formation⁹⁹. Secondly, the θ distribution (Fig. 4d) shows at later times partly negative values, in contrast to mostly positive values corresponding to a force spreading the embryo downward around the yolk. This corroborates the established invagination of cells on the inside of the spreading embryonic tissue sheet moving back towards the pole⁹⁹. However, a model including the missing radial dimension is necessary to conclusively resolve the invagination.

Outlook

This tutorial is meant to provide the reader with adequate insights enabling them to employ the adjoint method to their own scientific issues, but it also serves as an exploration of the adjoint method, elucidating its robust, dependable, and efficient nature in the context of multi-parameter estimation for differential equations. The method, as demonstrated through different examples, is not just a computational workhorse but also holds significant implications for advancing scientific enquiry, as mathematical modelling tools are increasingly applied to ever more complex systems. There, the versatility of the adjoint method becomes apparent in addressing various objectives, including the inference of non-observable variables from observable data, the validation of model assumptions, the expansion of existing models to accommodate new variables for better alignment with observed data, and the construction of parsimonious predictive models.

The utility of the adjoint method extends beyond mere parameter estimation, encompassing a spectrum of applications that contribute to the refinement and validation of mathematical models. By enabling the extraction of non-observable variables, the method facilitates a deeper understanding of complex systems, allowing researchers to validate the assumptions inherent in their models. Furthermore, the incorporation of new variables into existing models enhances their adaptability, enabling a more nuanced representation of the intricate relationships within the data.

In the spirit of fostering accessibility and encouraging broader utilization of the adjoint method, readers are encouraged to consider two

avenues of implementation. Direct implementation of the adjoint sensitivity method for nuanced problem-solving. Alternatively, for those with a less pronounced mathematical background, the adoptODE framework is presented as an accessible and user-friendly tool. This framework, available at <https://gitlab.gwdg.de/sherzog3/adoptODE>, is meticulously designed to cater to non-specialists, facilitating the seamless application of the adjoint method to a diverse array of scientific problems. Additionally, the supplied notebooks demonstrate the application of adoptODE to the examples presented in this tutorial, and should serve as a starting point for the adaptation to other problems.

Implicit in this discourse is the conviction that the widespread adoption of the adjoint method holds the potential to enhance interdisciplinary collaboration within the scientific community. By bridging the gap between theoretical constructs and experimental data, this method stands poised to elevate the synergy between theory and experimentation, thereby contributing to the collective advancement of scientific knowledge.

Data availability

The experimentally obtained flow fields for the zebrafish as well as the planet trajectories taken from the NASA Horizons System⁸⁸ are available in the git repository, Supplementary Movies 1 and 2 are available on the Communication Physics website.

Code availability

All results presented in this review were obtained using the adoptODE package. The package, notebooks with the implementation for the specific examples, as well as the reference data used are available in the git repository <https://gitlab.gwdg.de/sherzog3/adoptODE> and the version used to produce the results in this publication is tagged as <https://gitlab.gwdg.de/sherzog3/adoptODE/-/archive/Published/adoptODE-Published.zip>.

Received: 1 December 2023; Accepted: 25 March 2024;

Published online: 15 April 2024

References

- Bianconi, G. et al. Complex systems in the spotlight: next steps after the 2021 Nobel Prize in Physics. *J. Phys.* **4**, 010201 (2023).
- Wikle, C. K. & Berliner, L. M. A Bayesian tutorial for data assimilation. *Physica D* **230**, 1–16 (2007).
- Asch, M., Bocquet, M. & Nodet, M. *Data Assimilation* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2016).
- Nadler, P., Arcucci, R. & Guo, Y.-K. Data assimilation for parameter estimation in economic modelling. In *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)* 649–656 (IEEE, Sorrento, Italy, 2019).
- Bettencourt, L. M. A., Ribeiro, R. M., Chowell, G., Lant, T. & Castillo-Chavez, C. Towards real time epidemiology: data assimilation, modeling and anomaly detection of health surveillance data streams. *Intell. Secur. Inform.* **4506**, 79–90 (2007).

6. Blum, J., Dimet, F.-X. L. & Navon, I. M. Data assimilation for geophysical fluids. In *Handbook of Numerical Analysis*, Vol. 14 of *Special Volume: Computational Methods for the Atmosphere and the Oceans* (eds Temam, R. M. & Tribbia, J. J.), 385–441 (Elsevier, 2009).
7. Pospiech, G. & Fischer, H. E. Physical–mathematical modelling and its role in learning physics. In *Physics Education, Challenges in Physics Education* (eds Fischer, H. E. & Girwidz, R.) 201–229 (Springer International Publishing, Cham, 2021).
8. Ji, P. et al. Signal propagation in complex networks. *Phys. Rep.* **1017**, 1–96 (2023).
9. Alonso, S., Bär, M. & Echebarria, B. Nonlinear physics of electrical wave propagation in the heart: a review. *Rep. Prog. Phys.* **79**, 096601 (2016).
10. Arora, J. S. & Haug, E. J. Methods of design sensitivity analysis in structural optimization. *AIAA J.* **17**, 970–974 (1979).
11. Belegundu, A. D. Lagrangian approach to design sensitivity analysis. *J. Eng. Mech.* **111**, 680–695 (1985).
12. Errico, R. M. What is an adjoint model? *Bull. Am. Meteorol. Soc.* **78**, 2577–2592 (1997).
13. Jurovics, S. A. & McIntyre, J. E. The adjoint method and its application to trajectory optimization. *ARS J.* **32**, 1354–1358 (1962).
14. Lotkin, M. & Browne, H. N. On the accuracy of the adjoint method of differential corrections. *Am. Math. Mon.* **63**, 97–105 (1956).
15. Plessix, R.-E. A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. *Geophys. J. Int.* **167**, 495–503 (2006).
16. Tomović, R. *Sensitivity Analysis of Dynamic Systems. Electronic Sciences Series* (McGraw-Hill, 1963).
17. Tomović, R. The role of sensitivity analysis in engineering problems. In *Sensitivity Methods in Control Theory* (ed. Radanovic, L.) 103–109 (Pergamon, 1966).
18. Koopmans, T. C. & Reiersol, O. The identification of structural characteristics. *Ann. Math. Stat.* **21**, 165–181 (1950).
19. Fisher, F. M. Generalization of the rank and order conditions for identifiability. *Econometrica* **27**, 431–447 (1959).
20. Berman, M. & Schoenfeld, R. Invariants in experimental data on linear kinetics and the formulation of models. *J. Appl. Phys.* **27**, 1361–1370 (1956).
21. Yue, H. et al. Insights into the behaviour of systems biology models from dynamic sensitivity and identifiability analysis: a case study of an NF- κ B signalling pathway. *Mol. Biosyst.* **2**, 640–649 (2006).
22. Vilela, M., Vinga, S., Maia, M. A. G. M., Voit, E. O. & Almeida, J. S. Identification of neutral biochemical network models from time series data. *BMC Syst. Biol.* **3**, 47 (2009).
23. Thakker, K. M. *Compartmental models and their application*, Keith Godfrey. Academic Press Inc., London, 1983. No. of pages: 293, Price: \$50.00. *Biopharm. Drug Dispos.* **6**, 357–358 (1985).
24. Rodriguez-Fernandez, M., Egea, J. A. & Banga, J. R. Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. *BMC Bioinform.* **7**, 483 (2006).
25. Raue, A. et al. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics* **25**, 1923–1929 (2009).
26. Cobelli, C. & DiStefano, J. J. Parameter and structural identifiability concepts and ambiguities: a critical review and analysis. *Am. J. Physiol.* **239**, R7–24 (1980).
27. Cobelli, C., Finkelstein, L. & Carson, E. R. Mathematical modelling of endocrine and metabolic systems: Model formulation, identification and validation. *Math. Comput. Simul.* **24**, 442–451 (1982).
28. Bellmann, K. & Jacquez, J. A. *Compartmental analysis. Biology and Medicine*. Elsevier Publ. Co., Amsterdam, New York 1972. XIV, 237 S., 93 Abb., 1 Tab., \$24.35. *Biom. Z.* **16**, 537–537 (1974).
29. Anderson, D. H. Structural properties of compartmental models. *Math. Biosci.* **58**, 61–81 (1982).
30. Miao, H., Xia, X., Perelson, A. S. & Wu, H. On Identifiability of Nonlinear ODE Models and Applications in Viral Dynamics. *SIAM Rev.* **53**, 3–39 (2011).
31. Kalman, R. E. A new approach to linear filtering and prediction problems. *J. Basic Eng.* **82**, 35–45 (1960). **Estimating unknown quantities from integrating repeated measurements, Kalman filters are a main tool in studying dynamical systems.**
32. Frei, M. *Ensemble Kalman Filtering and Generalizations*. Doctoral Thesis, ETH Zurich (2013).
33. Meng, X.-L. & Van Dyk, D. The EM Algorithm—an old folk-song sung to a fast new tune. *J. R. Stat. Soc.: Ser. B (Methodological)* **59**, 511–567 (1997). **The widely-used Expectation-Maximization algorithm allows to abstract the parameters which are the most likely given a set of observations and an underlying statistical model.**
34. Vroylandt, H., Goudenège, L., Monmarché, P., Pietrucci, F. & Rotenberg, B. Likelihood-based non-Markovian models from molecular dynamics. *Proc. Natl Acad. Sci. USA* **119**, e2117586119 (2022).
35. Judd, K. Failure of maximum likelihood methods for chaotic dynamical systems. *Phys. Rev. E* **75**, 036210 (2007).
36. Jain, R. B. & Wang, R. Y. Limitations of maximum likelihood estimation procedures when a majority of the observations are below the limit of detection. *Anal. Chem.* **80**, 4767–4772 (2008).
37. Genschel, U. & Meeker, W. Q. A comparison of maximum likelihood and median-rank regression for Weibull estimation. *Qual. Eng.* **22**, 236–255 (2010).
38. Pes, F. & Rodriguez, G. A doubly relaxed minimal-norm Gauss–Newton method for underdetermined nonlinear least-squares problems. *Appl. Numer. Math.* **171**, 233–248 (2022).
39. Eberl, H., Kheil, A. & Wilderer, P. Multiple data parameter identification for nonlinear conceptual models. *Water Sci. Technol.* **36**, 61–68 (1997).
40. Forbes, A. B. Parameter estimation based on least squares methods. In *Data Modeling for Metrology and Testing in Measurement Science, Modeling and Simulation in Science, Engineering and Technology* (eds Pavese, F. & Forbes, A. B.) 1–30 (Birkhäuser, Boston, 2009).
41. Wei, C. Least squares estimation for a class of uncertain Vasicek model and its application to interest rates. *Stat. Pap.* <https://link.springer.com/article/10.1007/s00362-023-01494-1#citeas> (2023).
42. Cimpoesu, E. M., Ciubotaru, B. D. & Stefanoiu, D. Fault detection and diagnosis using parameter estimation with recursive least squares. In *2013 19th International Conference on Control Systems and Computer Science* 18–23 (IEEE, Bucharest, Romania, 2013).
43. Brunton, S. L., Proctor, J. L. & Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113**, 3932–3937 (2016).
44. Lejarza, F. & Baldea, M. Data-driven discovery of the governing equations of dynamical systems via moving horizon optimization. *Sci. Rep.* **12**, 11836 (2022).
45. Bock, H. G., Kostina, E. & Schlöder, J. P. Numerical methods for parameter estimation in nonlinear differential algebraic equations. *GAMM-Mitteilungen* **30**, 376–408 (2007).
46. Calver, J., Yao, J. & Enright, W. Using shooting approaches to generate initial guesses for ODE parameter estimation. In *Recent Developments in Mathematical, Statistical and Computational Sciences, Springer Proceedings in Mathematics & Statistics* (eds Kilgour, D. M., Kunze, H., Makarov, R., Melnik, R. & Wang, X.) 267–276 (Springer International Publishing, Cham, 2021).
47. George Mason University, Hamilton, F. Parameter estimation in differential equations: a numerical study of shooting methods. *SIAM Undergrad. Res. Online* **4**, 16–31 (2011).
48. Bock, H. G. & Plitt, K. J. A multiple shooting algorithm for direct solution of optimal control problems*. *IFAC Proc. Vol.* **17**, 1603–1608 (1984).

49. Diehl, M., Bock, H., Diedam, H. & Wieber, P.-B. Fast direct multiple shooting algorithms for optimal robot control. In *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control, Lecture Notes in Control and Information Sciences* (eds Diehl, M. & Mombaur, K.) 65–93 (Springer, Berlin, Heidelberg, 2006).
50. Horbelt, W., Müller, T., Timmer, J., Melzer, W. & Winkler, K. Analysis of nonlinear differential equations: parameter estimation and model selection. In *Medical Data Analysis Vol. 1933* (eds Goos, G., Hartmanis, J., Van Leeuwen, J., Brause, R. W. & Hanisch, E.) 152–159 (Springer, Berlin, Heidelberg, 2000).
51. Ramsay, J. O., Hooker, G., Campbell, D. & Cao, J. Parameter estimation for differential equations: a generalized smoothing approach. *J. R. Stat. Soc. Ser. B* **69**, 741–796 (2007). **Offering a systematic way to deal with noise in observations by smoothing over a controllable scale, smoothing-based approaches as this are an efficient way to treat noise present in most experiments.**
52. Frasso, G., Jaeger, J. & Lambert, P. Estimation and approximation in nonlinear dynamic systems using quasilinearization. Preprint at arXiv:1404.7370 (2014).
53. Zeng, W. & Liu, G. R. Smoothed finite element methods (S-FEM): an overview and recent developments. *Arch. Comput. Methods Eng.* **25**, 397–435 (2018).
54. Kim, S.-J., Koh, K., Lustig, M., Boyd, S. & Gorinevsky, D. An interior-point method for large-scale ℓ_1 -regularized least squares. *IEEE J. Sel. Top. Signal Process.* **1**, 606–617 (2007).
55. Ulbrich, M. & Ulbrich, S. Primal-dual interior-point methods for PDE-constrained optimization. *Math. Program.* **117**, 435–485 (2009).
56. Zavala, V. M., Laird, C. D. & Biegler, L. T. Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems. *Chem. Eng. Sci.* **63**, 4834–4845 (2008).
57. Zorkal'tsev, V. I. Interior point method: history and prospects. *Comput. Math. Math. Phys.* **59**, 1597–1612 (2019).
58. Santos, L.-R., Villas-Bôas, F., Oliveira, A. R. L. & Perin, C. Optimized choice of parameters in interior-point methods for linear programming. *Comput. Optim. Appl.* **73**, 535–574 (2019).
59. Andrei, N. Interior-Point Methods. In *Modern Numerical Nonlinear Optimization* (ed. Andrei, N.) *Springer Optimization and Its Applications* 599–645 (Springer International Publishing, Cham, 2022).
60. Shamieh, F. & Xu, C. Generation of optimal functions using particle swarm method over discrete intervals. In *NAFIPS 2009—2009 Annual Meeting of the North American Fuzzy Information Processing Society*, 1–5 (IEEE, Cincinnati, OH, USA, 2009).
61. Chronopoulou, A. & Spiliopoulos, K. Sequential Monte Carlo with parameter learning for non-markovian state-space models. arXiv:1508.02651v2 [stat.ME] (2015).
62. Neubauer, A. Theory of the simple genetic algorithm with α -selection, uniform crossover and bitwise mutation. *WTOS* **9**, 989–998 (2010).
63. Loh, A. & Lee, T. Parameter estimation using artificial neural nets. *IFAC Proc. Vol.* **24**, 81–83 (1991).
64. Dua, V. An Artificial Neural Network approximation based decomposition approach for parameter estimation of system of ordinary differential equations. *Comput. Chem. Eng.* **35**, 545–553 (2011).
65. Materka, A. Intelligent modular network for dynamic system parameter estimation. In *Proc. International Conference Signal Processing and Application Technology* 1353–1357 (Miller Freeman, Boston, MA, USA, 1996).
66. Nunes da Silva, I., Arruda, L. V. R. & Caradori do Amaral, W. A neural network for robust estimation and uncertainty intervals evaluation of nonlinear models. *IFAC Proc. Vol.* **32**, 5141–5146 (1999).
67. Jamal, S. A., Corpetti, T., Tiede, D., Letard, M. & Lague, D. Estimation of physical parameters of waveforms with neural networks. Preprint at arXiv:2312.10068 (2023).
68. Samad, T. & Mathur, A. Parameter estimation for process control with neural networks. *Int. J. Approx. Reason.* **7**, 149–164 (1992).
69. Morshed, J. & Kaluarachchi, J. J. Parameter estimation using artificial neural network and genetic algorithm for free-product migration and recovery. *Water Resour. Res.* **34**, 1101–1113 (1998).
70. Raol, J. & Madhuranath, H. Neural network architectures for parameter estimation of dynamical systems. *IEE Proc. — Control Theory Appl.* **143**, 387–394 (1996).
71. Chon, K. & Cohen, R. Linear and nonlinear ARMA model parameter estimation using an artificial neural network. *IEEE Trans. Biomed. Eng.* **44**, 168–174 (1997).
72. Tompos, A., Margitfalvi, J. L., Tfirst, E. & Héberger, K. Predictive performance of “highly complex” artificial neural networks. *Appl. Catal. A: Gen.* **324**, 90–93 (2007).
73. Sha, W. Comment on the issues of statistical modelling with particular reference to the use of artificial neural networks. *Appl. Catal. A: Gen.* **324**, 87–89 (2007).
74. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707 (2019).
75. O’Leary, J., Paulson, J. A. & Mesbah, A. Stochastic physics-informed neural ordinary differential equations. *J. Comput. Phys.* **468**, 111466 (2022).
76. McNamara, A., Treuille, A., Popović, Z. & Stam, J. Fluid control using the adjoint method. *ACM Transactions on Graphics* **23**, 449–456 (2004).
77. Pontryagin, L. S. *Mathematical Theory of Optimal Processes* (CRC Press, 1987). **Formulating Pontryagins Maximum Principle, in this book not only the adjoint method itself, but also a large part of its rigorous mathematical boundaries are established.**
78. Bhat, H. S. System identification via the adjoint method. In *2021 55th Asilomar Conference on Signals, Systems, and Computers* 1317–1321 (IEEE, Pacific Grove, CA, USA, 2021).
79. Chen, R. T. Q., Rubanova, Y., Bettencourt, J. & Duvenaud, D. K. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, Vol. 31 (eds Bengio, S. et al.) (Curran Associates, Inc., 2018). **The most prominent application in recent years, this paper presents the since influential idea of a neural network with a continuous equation of motion instead of discrete layers (Neural Ordinary Differential Equation, NODE), where the adjoint method is used for finding the gradients relevant for training.**
80. Rackauckas, C. et al. Universal differential equations for scientific machine learning. Preprint at arXiv:2001.04385 (2021).
81. Ma, Y., Dixit, V., Innes, M. J., Guo, X. & Rackauckas, C. A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. In *2021 IEEE High Performance Extreme Computing Conference (HPEC)* 1–9 (IEEE, Waltham, MA, USA, 2021). **Using multiple benchmark systems, the capabilities of the adjoint method are compared to other approaches, i.e. autodifferentiation through the solver.**
82. Banks, H. T. & Lamm, P. K. D. Estimation of delays and other parameters in nonlinear functional differential equations. *SIAM J. Control Optim.* **21**, 895–915 (1983).
83. Hao, W. A homotopy method for parameter estimation of nonlinear differential equations with multiple optima. *J. Sci. Comput.* **74**, 1314–1324 (2018).
84. Rizk-Allah, R. M. & Hassanien, A. E. A hybrid Harris Hawks–Nelder–Mead optimization for practical nonlinear ordinary differential equations. *Evol. Intell.* **15**, 141–165 (2022).
85. Bradbury, J. et al. JAX: Composable transformations of Python + NumPy programs (2018). **The python library JAX allows for efficient computation, including autodifferentiation, a crucial**

- capability allowing to automatize obtaining the otherwise cumbersome to compute Jacobians necessary.
86. Shampine, L. F. Some practical runge-kutta formulas. *Math. Comput.* **46**, 135–150 (1986).
 87. Kidger, P. On neural differential equations. Ph.D. thesis, University of Oxford (2021).
 88. Giorgini, J. D. Status of the JPL horizons ephemeris system. *IAU General Assembly* **29**, 2256293 (2015).
 89. Jones, J. E. On the determination of molecular fields—I. From the variation of the viscosity of a gas with temperature. *Proc. R. Soc. Lond. Ser. A* **106**, 441–462 (1924).
 90. Lotka, A. J. Contribution to the theory of periodic reactions. *J. Phys. Chem.* **14**, 271–274 (1910).
 91. Volterra, V. Fluctuations in the abundance of a species considered mathematically. *Nature* **118**, 558–560 (1926).
 92. Bueno-Orovio, A., Cherry, E. M. & Fenton, F. H. Minimal model for human ventricular action potentials in tissue. *J. Theor. Biol.* **253**, 544–560 (2008).
 93. Aliev, R. R. & Panfilov, A. V. A simple two-variable model of cardiac excitation. *Chaos Solitons Fractals* **7**, 293–301 (1996).
 94. Oberbeck, A. Über die Wärmeleitung der Flüssigkeiten bei Berücksichtigung der Strömungen infolge von Temperaturdifferenzen. *Ann. Phys.* **243**, 271–292 (1879).
 95. Boussinesq, J. *Theorie Analytique de La Chaleur Mise En Harmonie Avec La Thermodynamique et Avec La Theorie Mcanique de La Lumire: Refroidissement et Chauffage Par Rayonnement, Conductibilit Des Tiges, Lames et Masses Cristallines, Courants de Convection, Theorie Mcanique de La Lumire*. 1903. Xxxii, 625,[1] p Vol. 2 (Gauthier-Villars, 1903).
 96. Rempfer, D. On boundary conditions for incompressible Navier–Stokes problems. *Appl. Mech. Rev.* **59**, 107–125 (2006).
 97. Grant, I. Particle image velocimetry: a review. *Proc. Inst. Mech. Eng. Part C* **211**, 55–76 (1997).
 98. Amat, F. et al. Fast, accurate reconstruction of cell lineages from large-scale fluorescence microscopy data. *Nat. Methods* **11**, 951–958 (2014).
 99. Kimmel, C. B., Ballard, W. W., Kimmel, S. R., Ullmann, B. & Schilling, T. F. Stages of embryonic development of the zebrafish. *Dev. Dyn.* **203**, 253–310 (1995).

Acknowledgements

The authors would like to thank the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project-ID 454648639—SFB 1528 for funding this work. The instrument used for the zebrafish data was funded by the DFG—Project-ID 455153711. S.H. would also like to thank Ulrich Parlitz and Christian Tetzlaff for their continued support and inspiring scientific discussions. L.L. would like to thank Ulrich S. Schwarz, Falko Ziebert and Oliver M. Drozdowski for their continuous support and many scientific discussions.

Author contributions

L.L., S.H., T.B. and F.W. wrote the paper; S.H. came up with the application idea. L.L., S.H., T.B., A.J. and F.W. designed the numerical experiments; T.B. and A.J. contributed the experimental data; L.L., S.H., A.J. and T.B. performed the experiments and analysed the data; L.L. implemented adoptODE and provided the numerical simulations. S.H. supervised the implementation of adoptODE and the numerical simulations. All authors read and approved the manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s42005-024-01606-9>.

Correspondence and requests for materials should be addressed to Sebastian Herzog.

Peer review information *Communications Physics* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2024