Error estimation for physics-informed neural networks with implicit Runge-Kutta methods

Jochen Stiasny JBEST@DTU.DK

Spyros Chatzivasileiadis

SPCHATZ@DTU.DK

Department of Wind and Energy Systems Technical University of Denmark Kgs. Lyngby, DK-2800

Abstract

The ability to accurately approximate trajectories of dynamical systems enables their analysis, prediction, and control. Neural network (NN)-based approximations have attracted significant interest due to fast evaluation with good accuracy over long integration time steps. In contrast to established numerical approximation schemes such as Runge-Kutta methods, the estimation of the error of the NN-based approximations proves to be difficult. In this work, we propose to use the NN's predictions in a high-order implicit Runge-Kutta (IRK) method. The residuals in the implicit system of equations can be related to the NN's prediction error, hence, we can provide an error estimate at several points along a trajectory. We find that this error estimate highly correlates with the NN's prediction error and that increasing the order of the IRK method improves this estimate. We demonstrate this estimation methodology for Physics-Informed Neural Network (PINNs) on the logistic equation as an illustrative example and then apply it to a four-state electric generator model that is regularly used in power system modelling.

Keywords: Dynamical system, error estimation, physics-informed neural network, Runge-Kutta method

1. Introduction

Differential equations provide a powerful modelling tool to describe how systems change. They are ubiquitous in science and engineering and are used to formulate the behaviour of dynamical systems (Strogatz, 2015). While instantaneous changes are simple to evaluate, there, usually, exists no analytical solution to these differential equations. To predict future states of the system, we revert to numerical integration methods, see for example Hairer et al. (1993); Hairer and Wanner (1996); Iserles (2008) for detailed treatment of the topic. A key question is the estimation of the approximation error of an integration scheme—both for a single time step and multiple time steps. For established schemes such as Runge-Kutta (RK) methods, this estimation hinges around the local approximation error ε which is closely related to the step size h. This relation is expressed as the order p of a scheme $\varepsilon = \mathcal{O}\left(h^{p+1}\right)$ and it describes the error characteristic when h is sufficiently small (Hairer et al., 1993). To obtain absolute error estimates, an additional RK scheme can be constructed to form an embedded RK method (Hairer et al., 1993). The resulting error estimates are crucial in choosing time step sizes that yield a balance between a sufficiently accurate approximation and the required computational cost.

For dynamical systems found in power systems, the computational burden of solving the associated differential equations is significant—Konstantelos et al. (2017) provide an indication of the scales. Grid operators, who need to ensure system stability, rely heavily on these numerical integration methods to choose control actions, but the computational burden becomes a limiting factor. Therefore, the choice of the integration method and accurate error estimate are critical; this becomes even more relevant in light of the energy transition with additional dynamic phenomena (Hatziargyriou et al., 2021). Neural Network (NN)-based approaches such as in Cui et al. (2023); Moya and Lin (2023); Stiasny et al. (2023) could have upsides in terms of prediction speed, however, the question of estimating and controlling the approximation error poses a major challenge to their adoption. Motivated by this example, this work shall provide a novel error estimation procedure to understand the error characteristics of learned solutions for dynamical systems.

Neural network-based integration: With NN-based integration methods we *learn* an approximate solution function to a differential equation or a system of differential equations. This learned solution offers the upside of fast and explicit evaluations while providing sufficiently accurate results over large time steps, given appropriate training. These benefits come at the expense of the need to train the model in advance. As suggested in Lagaris et al. (1998) the training procedure can be set up as a collocation method. The NN constitutes the candidate function and is fit to match the differential equations at collocation points. This methodology has been revived in Raissi et al. (2018) under the term Physics-Informed Neural Network (PINN). A related approach, that can be seen as a generalisation of PINNs, is termed operator learning (Lu et al., 2021; Kovachki et al., 2023); we will subsequently use NN or PINN to refer to these kind of approaches. As these methods do not adhere to error characteristics by construction—unlike RK schemes—understanding and estimating the prediction errors is an open research question.

Approaches for analysing the error of PINNs: The trivial approach consists in comparing the PINN's prediction with a highly accurate integration scheme that can be regarded as ground truth. However, as this assessment might involve significant computational cost, our interest lies on approaches for estimating the error characteristics without knowing the ground truth. For the simplest version of PINNs, we obtain a local approximation error $\mathcal{O}(1)$, meaning that the error is independent of the time step size h. Lagaris et al. (1998) proposed to change the NN's architecture to ensure numerical consistency for small h; reducing the time step size should reduce the error. However, achieving high orders p would require many additional function evaluations which reduces the attractiveness of PINNs from the standpoint of speed. Furthermore, these changes do not necessarily help with controlling the error for large time steps. This is undesirable as PINNs become an attractive alternative to RK schemes only for large time steps.

The following approaches focus on providing guaranteed error bounds to certify a PINN. Hillebrecht and Unger (2022) formulate error dynamics between the exact solution and the PINN approximation. These dynamics can be bounded using the Lipschitz constant of the dynamical system, but for non-linear systems this might not result in tight error bounds. Several works (Yarotsky, 2017; De Ryck et al., 2021; De Ryck and Mishra, 2022) aim to provide theoretical upper bounds on the approximation errors of NNs based on the function class that they describe, however, these results do not provide practical error estimates. Eiras et al. (2023) verify that the residual of the differential equations based on the learned approximation is bounded and they show empirically that this bound is related to the maximum approximation error.

Proposed approach: In this work, we combine PINN with the collocation method of Gauss-Legendre Runge-Kutta schemes, a special form of implicit RK schemes, to estimate the error, similar to embedded RK schemes. To this end, we evaluate the PINN at time instances (nodes) defined by the IRK scheme. For a high-order IRK scheme, the function values at the nodes coincide with intermediate values along the trajectory of interest. We then evaluate how closely these predictions match the implicit system of equations defined by the IRK scheme. Based on the mismatch, we can estimate the error at the nodes and hence along the trajectory. As a numerical example, we illustrate the methodology on the logistic equation. A power system generator model with four states serves as a second example to showcase the resulting error estimation on a relevant use case.

Paper structure: Section 2 briefly introduces the notation as well as IRK methods and PINNs as numerical integration methods. Section 3 explains the error estimation procedure. Sections 4 and 5 illustrate the methodology on the example of the logistic equation and the power system generator respectively. Section 6 concludes and discusses next steps.

2. Approximating the trajectory of a dynamical system

We assume an autonomous dynamical system governed by

$$\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t)\right) \tag{1}$$

with state $x \in \mathbb{R}^n$, time $t \in \mathbb{R}$ and the update function $f : \mathbb{R}^n \mapsto \mathbb{R}^n$. Given an initial condition $x_0 = x(t_0)$, the trajectory $x(t_0 + h)$ can be obtained by integration of (1)

$$\boldsymbol{x}(t_0 + h) = \boldsymbol{x}_0 + \int_{t_0}^{t_0 + h} \boldsymbol{f}(\boldsymbol{x}(t)) dt.$$
 (2)

We refer to $h = t - t_0$ as the time step size. Given that f is continuous and Lipschitz continuous, the existence and uniqueness of x(h) is ensured. Usually, no analytical form for (2) exists, hence, we revert to constructing an approximation $\hat{x}(h)$ of the trajectory. There is a great variety of established methods, see for example Hairer et al. (1993); Iserles (2008), in this work we focus on implicit RK methods (Section 2.1). In Section 2.2, we present PINNs as a NN-based integration method.

2.1. Implicit Runge-Kutta (IRK) methods

As described in Iserles (2008), (2) constitutes a quadrature problem and it can be approximated by a weighted sum of the function values at s nodes

$$x(t_0 + h) \approx x_0 + h \sum_{i=1}^{s} b_i f(x(t_0 + c_i h)).$$
 (3)

The location of each node is determined by the parameter c_i which usually ranges in [0,1] and it is associated with the positive weight b_i . As the quadrature requires $\boldsymbol{x}(t_0+c_ih)$ at the nodes, their values are approximated by $\boldsymbol{k}^{(i)} \approx \boldsymbol{x}(t_0+c_ih)$ and are subsequently referred to as the *i*-th stage of the RK scheme. By constructing a polynomial of order s that passes through the node values $\boldsymbol{k}^{(i)}$, a

set of conditions can be derived—we construct a collocation method. The conditions are described by the following system of implicit equations

$$\boldsymbol{k}^{(i)} = \boldsymbol{x}_0 + h \sum_{j=1}^{s} a_{ij} \boldsymbol{f} \left(\boldsymbol{k}^{(j)} \right). \tag{4}$$

By solving for $k^{(i)}$, we can then calculate

$$\hat{\boldsymbol{x}}(t_0 + h) = \boldsymbol{x}_0 + h \sum_{i=1}^s b_i \, \boldsymbol{f}\left(\boldsymbol{k}^{(i)}\right). \tag{5}$$

The number of stages s and the choice of a_{ij}, b_i, c_i determine the accuracy of the scheme. It can be shown, that the choice of Gauss-Legendre polynomials for b_i, c_i leads to a scheme of numerical order 2s. This implies that the approximation error scales according to $\mathcal{O}\left(h^{2s}\right)$. This relationship allows IRK schemes to achieve highly accurate approximations for large time step sizes h and moderate numbers of stages s. The approximation error $\eta^{(i)} \in \mathbb{R}^n$

$$\boldsymbol{\eta}^{(i)} := \boldsymbol{k}^{(i)} - \boldsymbol{x}(t_0 + c_i h) \tag{6}$$

of the stages that result from solving the collocation scheme scales as $\mathcal{O}(h^s)$ (Butcher, 2009). We will use this relation in the error estimation in Section 3 as it provides insightful information at intermediate points along the trajectory.

2.2. Neural-Network-based approximation

In the previous section, the candidate solution for the collocation problem is a polynomial. Instead, we now choose a NN to approximate x

$$\hat{\boldsymbol{x}}(t_0 + h) = \boldsymbol{x}_0 + h \, \operatorname{NN}_{\theta}(h). \tag{7}$$

The modifications of multiplying NN $_{\theta}(h)$ with h and the addition of the initial condition x_0 can already be found in Lagaris et al. (1998). They ensure that the initial condition is satisfied for h=0 and the local approximation error will scale as $\mathcal{O}(h)$ if NN $_{\theta}(h)$ is bounded.

In this work, we use a simple feed-forward neural network NN $_{\theta}: \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ with L layers of size d_l . It describes a concatenation of affine transformations $\mathcal{A}_l: \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l}: z \mapsto W_l z + b_l$ with non-linear element-wise activation functions $\sigma: \mathbb{R} \to \mathbb{R}$

$$NN_{\theta}(h) = \mathcal{A}_{L} \circ \sigma \circ \mathcal{A}_{L-1} \circ \cdots \circ \sigma \circ \mathcal{A}_{1}(h).$$

The weight matrices W_l and bias vectors b_l of the L layers constitute the trainable parameters θ and for the present case, the time step size forms the input $(d_0 = 1)$ and the n states of the system form the output $(d_L = n)$. The training problem to determine θ becomes

$$\min_{\theta} \frac{1}{N} \sum_{j=1}^{N} \left\| \frac{d}{dt} \hat{\boldsymbol{x}} \left(t_0 + h^{(j)} \right) - \boldsymbol{f} \left(\hat{\boldsymbol{x}} \left(t_0 + h^{(j)} \right) \right) \right\|_2^2$$
 (8)

as introduced in Lagaris et al. (1998) and termed PINNs by Raissi et al. (2018). This can be seen as a collocation method for finding θ by evaluating the norm in (8) at N collocation points at time step sizes $h^{(j)}$. The temporal derivative $\frac{d}{dt}\hat{x}(t_0+h^{(j)})$ can be computed using automatic differentiation (AD) (Baydin et al., 2018), hence, this training procedure does not require simulated data points. We solve the optimisation problem in (8) with a gradient descent method, namely the limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) (Liu and Nocedal, 1989).

3. Error estimates of PINN predictions with IRK methods

The following error estimation approach is closely linked to the works in Hairer and Wanner (1996); de Swart and Söderlind (1997); Butcher (2009), however, we do not focus on the error estimation at the end of the time step $|\hat{x}(t_0 + h) - x(t_0 + h)|$. Instead, we aim for s error estimates at the s locations of the RK stages $t_0 + c_i h$ which are spread across the interval $[t_0, t_0 + h]$ as $c_i \in [0, 1]$.

To this end, we evaluate the approximation function \hat{x} defined by the PINN at the node locations $t_0 + c_i h$ that correspond to an IRK method of our choice. These predictions act as approximations

$$\hat{\boldsymbol{k}}^{(i)} = \hat{\boldsymbol{x}} \left(t_0 + c_i h \right) \tag{9}$$

of the IRK stages $k^{(i)}$ and will, generally speaking, lead to a mismatch $\delta^{(i)} \in \mathbb{R}^n$

$$\boldsymbol{\delta}^{(i)} := \hat{\boldsymbol{k}}^{(i)} - \boldsymbol{k}^{(i)},\tag{10}$$

that we will call the *stage error*. Using (6), (9) and (10) and $\eta^{(i)} = \mathcal{O}(h^s)$, we can formulate the desired error estimate for the NN-based function approximation

$$\hat{x}(t_0 + c_i h) - x(t_0 + c_i h) = \hat{k}^{(i)} - (k^{(i)} + \eta^{(i)}) = \delta^{(i)} + \mathcal{O}(h^s)$$
(11)

The subsequent paragraph describes the estimation of $\delta^{(i)}$ without explicitly solving for $k^{(i)}$.

Estimation of $\delta^{(i)}$: Due to the stage error $\delta^{(i)}$, the corresponding system of equations in (4) evaluated with $\hat{k}^{(i)}$ will also not be satisfied; a residual $\zeta^{(i)} \in \mathbb{R}^n$ for the system of equations for the *i*-th stage remains

$$\boldsymbol{\zeta}^{(i)} := \hat{\boldsymbol{k}}^{(i)} - \boldsymbol{x}_0 - h \sum_{j=1}^s a_{ij} \boldsymbol{f} \left(\hat{\boldsymbol{k}}^{(j)} \right). \tag{12}$$

We begin by replacing $\hat{k}^{(i)} = k^{(i)} + \delta^{(i)}$ and expanding $f\left(k^{(i)} + \delta^{(i)}\right)$ in a Taylor series expansion

$$\zeta^{(i)} = \mathbf{k}^{(i)} + \boldsymbol{\delta}^{(i)} - \left(\mathbf{x}(t_0) + h \sum_{j=1}^{s} a_{ij} \mathbf{f} \left(\mathbf{k}^{(j)} + \boldsymbol{\delta}^{(j)} \right) \right)$$

$$= \mathbf{k}^{(i)} + \boldsymbol{\delta}^{(i)} - \left(\mathbf{x}(t_0) + h \sum_{j=1}^{s} a_{ij} \left[\mathbf{f} \left(\mathbf{k}^{(j)} \right) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\mathbf{k}^{(j)}} \boldsymbol{\delta}^{(j)} + \mathcal{O} \left(\boldsymbol{\delta}^{(j)^2} \right) \right] \right)$$

$$= \boldsymbol{\delta}^{(i)} - h \sum_{j=1}^{s} a_{ij} \left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\mathbf{k}^{(j)}} \boldsymbol{\delta}^{(j)} + \mathcal{O} \left(\boldsymbol{\delta}^{(j)^2} \right) \right]$$
(13)

The last simplification follows from the definition of the IRK scheme evaluated on $k^{(i)}$ as it must equal 0. As (13) depends on $k^{(j)}$, we perform a Taylor series expansion of $\frac{\partial f}{\partial x}\Big|_{k^{(j)}}$ using $k^{(j)} = \hat{k}^{(j)} - \delta^{(j)}$. The relation between $\zeta^{(i)}$ and $\delta^{(i)}$ becomes

$$\boldsymbol{\zeta}^{(i)} = \boldsymbol{\delta}^{(i)} - h \sum_{j=1}^{s} a_{ij} \left. \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right|_{\hat{\boldsymbol{k}}^{(j)}} \boldsymbol{\delta}^{(j)} + \mathcal{O}\left(\boldsymbol{\delta}^{(j)^2}\right). \tag{14}$$

By collecting
$$\boldsymbol{\delta} = \begin{bmatrix} \boldsymbol{\delta}^{(1)} & \dots & \boldsymbol{\delta}^{(s)} \end{bmatrix} \in \mathbb{R}^{ns}$$
 and $\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\zeta}^{(1)} & \dots & \boldsymbol{\zeta}^{(s)} \end{bmatrix} \in \mathbb{R}^{ns}$ we obtain
$$\boldsymbol{\zeta} = (I - hJ) \, \boldsymbol{\delta} + \mathcal{O} \left(\boldsymbol{\delta}^2 \right) \tag{15}$$

with the identity matrix $I \in \mathbb{R}^{ns \times ns}$ and the Jacobian-like matrix $J \in \mathbb{R}^{ns \times ns}$ that is constructed based on the Jacobians $\frac{\partial f}{\partial x}\Big|_{\hat{k}^{(i)}} \in \mathbb{R}^{n \times n}$ and the weights of the IRK scheme a_{ij}

$$J = \begin{bmatrix} a_{11} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{k}}^{(1)}} & \dots & a_{1s} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{k}}^{(s)}} \\ \vdots & \ddots & \vdots \\ a_{s1} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{k}}^{(1)}} & \dots & a_{ss} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{k}}^{(s)}} \end{bmatrix}.$$

Neglecting the higher order terms of δ and inverting (I - hJ), we obtain

$$\boldsymbol{\delta} = (I - hJ)^{-1} \boldsymbol{\zeta} \tag{16}$$

which can then be used in (11) to estimate the prediction error of the NN.

4. Logistic equation - illustration with an analytical solution

This section demonstrates the methodology on the logistic equation

$$\frac{d}{dt}x = x(1-x)$$

with the analytical solution for an initial conditions $x(t_0) > 0$

$$x(t_0 + h) = \frac{e^h}{c_0 + e^h};$$
 $c_0 = \frac{1}{x(t_0)} - 1.$

We use the simple but nonlinear logistic equation due to the ease of visualisation and the existence of an analytical solution. The arguments can be equally made for dynamical systems with multiple states as we demonstrate later on. To build intuition, we show in Section 4.1 the prediction accuracy of different integration schemes and in Section 4.2 we illustrate the interpretation of the IRK stages as approximate intermediate values $x(t_0 + c_i h)$. Section 4.3 then presents the error estimation.

4.1. Approximating a trajectory of the logistic equation

We apply different integration schemes to the logistic equation, starting with an initial condition $x_0=0.01$. Figure $\mathbf{1}(a)$ shows the trace of the predictions when a single time step of size h is applied. We first want to note that—despite the simplicity of the trajectory—an explicit RK scheme such as the well-known fourth-order RK4 scheme fails to predict large time steps. Hence, we revert to IRK schemes for predictions over long time steps. With three stages (s=3), we can still observe an error, for eight stages (s=8), the predicted values are indistinguishable from the analytical solution. Lastly, we plot the prediction of a trained PINN and it also closely matches the analytical solution over the entire range of the tested time step sizes.

Plotting the approximation error $|\hat{x}-x|$ against the time step size in Figure 1(b) provides further insights. By showing both axes on a logarithmic scale the order p of the different schemes become visible. For the PINN, such a convergence properties is not observed, in fact, the error is significantly higher for h=5 than for h=10. This example illustrates, that while PINNs can predict large time steps accurately, reducing the time step size can lead to larger errors. This creates an error characteristic that is more difficult to analyse and control than for a RK scheme.

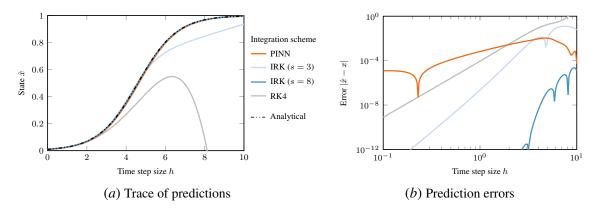


Figure 1: Predictions from different numerical integration schemes for a time step of size h.

4.2. Visualising the stages of an IRK scheme

We now perform a similar analysis for the stage value $k^{(i)}$ of the IRK schemes. We plot their values against the time c_ih that corresponds to the i-th node. For s=3, we can observe that the stages $k^{(2)}$ and $k^{(3)}$ in Figure 2(a) follow the analytical solution well at first, but then deviate. For the higher order IRK scheme (s=8) the stages $k^{(3)}$ and $k^{(5)}$ coincide closely with the analytical solution, that is the true trajectory. Figure 2(b) underlines this observation as we plot the stage approximation

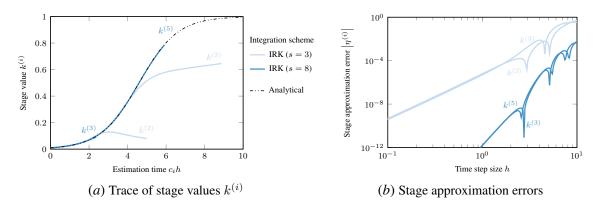


Figure 2: Analysis of the prediction accuracy of IRK stages for a time step of size h.

error $\eta^{(i)}$ against the time step size h. This error follows the order $\mathcal{O}(h^s)$, which illustrates that interpreting the stage values $k^{(i)}$ as approximations of the trajectory itself at intermediate points becomes very precise for high order schemes, even for large time step sizes h. The proposed error estimate exploits this fact.

4.3. Estimating the stage error

With this intuition in mind, we now apply the methodology presented in Section 3. The question we want to ask is how well the estimated error $\delta^{(i)}$ correlates with the prediction error $\hat{x}(t_0 + c_i h) - x(t_0 + c_i h)$. A strong correlation improves the usefulness of the estimate as it is more reliable. As the analytical solution exists, we compute the two errors for a time step of h = 5 (black dots) and h = 10 (red dots) and show the result in Figure 3. The subplots show the correlation for different

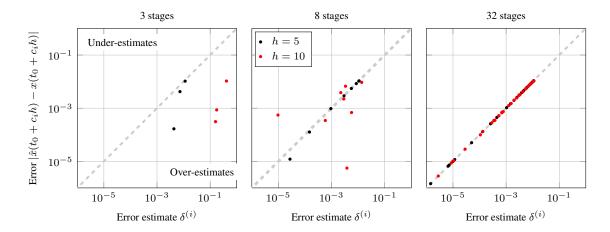


Figure 3: Correlation between the estimated stage error $\delta^{(i)}$ and the corresponding prediction error $|\hat{x}(t_0 + c_i h) - x(t_0 + c_i h)|$ for IRK schemes with 3, 8, and 32 stages. The black dots represent a time step size of h = 5 and the red dots a time step size of h = 10.

number of stages s. The dashed grey line indicates identity between the estimated and true predicted error. It is clearly visible that the larger s and the smaller time step size h improve the correlation which can be expected as the effect of the stage error $\eta^{(i)}$ reduces (see Section 4.2).

To identify the number of stages that is required for a reliable estimate, we plot the estimated error $\delta^{(i)}$ against the associated time value c_ih for different values of h, shown in Figure 4. We observe that for an IRK scheme with few stages (left plot), nearby values of c_ih have orders of magnitude different error estimates $\delta^{(i)}$. By increasing the number of stages, the error estimates become more consistent and approach the error characteristic based on the analytical solution. If

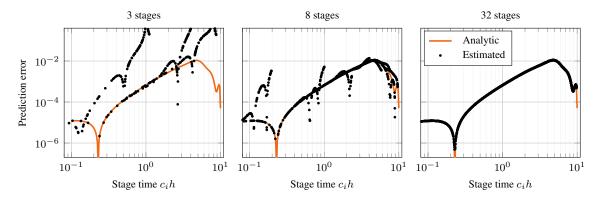


Figure 4: Consistency of stage prediction for different orders of the IRK schemes. The points indicate the error estimates $\delta^{(i)}$ for a range of values of h. The exact error $|\hat{x} - x|$ is shown in orange. For low order IRK schemes, the error estimates for a given time $c_i h$ clearly contradict each other. For a high-order IRK scheme all error estimates form a consistent estimate that is in line with the exact error characteristics.

the analytical solution is unknown, checking the consistency of the error estimate along a trajectory can be used for validating the estimation accuracy.

5. Error estimation for a power system generator model

As motivated in Section 1, NN-based approximations might be able to accelerate the simulation of dynamical phenomena in power systems. The PINNSim approach (Stiasny et al., 2023) relies on learning the dynamics of single components, such as generators, and then resolving their interactions. Being able to estimate errors of the learned dynamics will be an important step for future development of such approaches. As an example for the dynamics of a component, we use a two-axis generator model which is modelled by a state vector $x \in \mathbb{R}^4$ governed by

$$\frac{d}{dt} \begin{bmatrix} E'_q \\ E'_d \\ \delta \\ \Delta \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{T'_{do}} \left(-E'_q - (X_d - X'_d) I_d + E_{fd} \right) \\ \frac{1}{T'_{qo}} \left(-E'_d + (X_q - X'_q) I_q \right) \\ \omega_s \Delta \omega \\ \frac{1}{2H} \left(P_m - E'_d I_d - E'_q I_q - (X'_q - X'_d) I_d I_q - D\Delta \omega \right) \end{bmatrix}$$

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} R_s & -X_q \\ X_d & R_s \end{bmatrix}^{-1} \begin{bmatrix} E'_d - V \sin(\delta - \theta) \\ E'_q - V \cos(\delta - \theta) \end{bmatrix}.$$

We refer to Sauer and Pai (1998) for a detailed description of the modelling and the used parameters. The following results use the parameters associated with generator 2 in Sauer and Pai (1998, p. 167).

In Section 4, we learned a single trajectory. Here, we aim to predict a collection of trajectories starting from initial conditions $x_0 \in \mathcal{X}_0$ within the set \mathcal{X}_0 , hence, we adjust (7) to consider x_0 as an additional input to the PINN

$$\hat{x}(t_0 + h, x_0) = x_0 + h \, \text{NN}_{\theta}(h, x_0). \tag{17}$$

As a consequence, the approximation error will depend on h and x_0 . Therefore, we sample initial conditions from \mathcal{X}_0 and apply the proposed error estimation technique for each x_0 . As a result we obtain s error estimates along the trajectory of every tested x_0 . To report the error, we compute a norm over the prediction error $\hat{x} - x$ (evaluated at $c_i h$) and the estimated error $\delta^{(i)}$

$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2 = \sqrt{\sum_{j=1}^{n} \xi_j^2 (\hat{x}_j - x_j)^2}$$
 $\|\boldsymbol{\delta}^{(i)}\|_2 = \sqrt{\sum_{j=1}^{n} \xi_j^2 {\delta_j^{(i)}}^2}.$

We include a scaling factor ξ_j in the norm so that all state variations have similar magnitude, see Stiasny (2023) for more details. This factor was also employed when evaluating (8) during training the PINN model (3 layers with 64 neurons each). As for Figure 4, we collect the estimated errors at all s stages. Figure 5 shows the resulting error estimates $\|\boldsymbol{\delta}^{(i)}\|_2$ and the prediction errors evaluated with a differential equation solver with tolerance settings of 10^{-12} , as no analytical solution is available. To represent the dependency of the prediction error on the different initial conditions, we compute quantiles over small intervals of $c_i h$ which corresponds to the location of the stages.

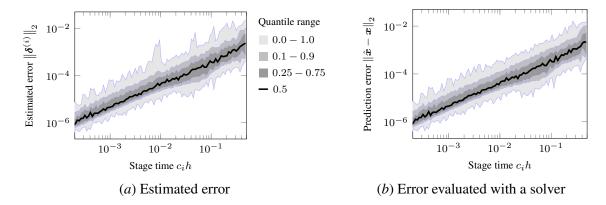


Figure 5: Error distribution of the PINN predictions with respect to the time step size h for a two-axis generator model. We compare the estimated errors $\|\boldsymbol{\delta}^{(i)}\|_2$ to the prediction error evaluated with a highly accurate solver. As the error depends also on the initial condition, the grey shadings correspond to the errors for 100%, 80%, and 50% of these trajectories.

Figure 5 uses s=8 and shows a great agreement of the estimated error with the true prediction error apart from a few instances where the proposed error estimation scheme over-estimates the maximum error, for example at $c_i h=0.01$ and $c_i h=0.05$. By increasing the number of stages, the estimation quality can be further improved.

6. Conclusions

The estimation of prediction errors poses a major challenge for Physics-Informed Neural Networks (PINNs). This work introduces a simulation-free and practical error estimation approach, which can remove the barrier for incorporating PINNs in established numerical solvers. Our methodology only requires the evaluation of an implicit Runge-Kutta (IRK) scheme (whose coefficients are usually pre-computed) and a matrix inversion, which leads to a simple and easy-to-implement estimation method, see (16). We demonstrate that the proposed approach can accurately estimate the prediction errors along a trajectory and is also applicable for dynamical systems with multiple states. We also show that it is straightforward to assess the quality of this estimate by analysing the consistency of the error estimates that stem from different time step sizes h.

Future work: From a theoretical perspective, we see the need to further investigate the influencing factors on the estimation accuracy. That is, first, the analysis of the impact of higher-order terms of $\delta^{(i)}$ which could become significant if the PINN's prediction is not very accurate. Furthermore, the impact of system characteristics such as stiffness and oscillatory behaviour on the required number of stages s should be investigated.

With respect to practical applications, the proposed estimation methodology can be used in monitoring the training progress, assessing the learned model, or providing estimates during prediction tasks. We plan to explore these use cases to develop heuristics for this method. Furthermore, the methodology could inform an additional term in the loss function similar to Raissi et al. (2018); Stiasny (2023). Lastly, the error estimates could also be employed at the evaluation stage to correct the PINN's prediction.

Acknowledgments

This work was supported by the ERC Project VeriPhIED, funded by the European Research Council, Grant Agreement No: 949899. The authors would like to thank Samuel Chevalier, Ignasi Ventura Nadal, and Petros Ellinas for their feedback on the manuscript.

References

- Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018.
- J. C. Butcher. Practical Runge–Kutta methods for scientific computation. *The ANZIAM Journal*, 50 (3):333–342, 2009.
- Wenqi Cui, Weiwei Yang, and Baosen Zhang. A Frequency Domain Approach to Predict Power System Transients. *IEEE Transactions on Power Systems*, pages 1–13, 2023.
- Tim De Ryck and Siddhartha Mishra. Generic bounds on the approximation error for physics-informed (and) operator learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in neural information processing systems*, volume 35, pages 10945–10958. Curran Associates, Inc., 2022.
- Tim De Ryck, Samuel Lanthaler, and Siddhartha Mishra. On the approximation of functions by tanh neural networks. *Neural Networks*, 143:732–750, 2021.
- Jacques J.B de Swart and Gustaf Söderlind. On the construction of error estimators for implicit Runge-Kutta methods. *Journal of Computational and Applied Mathematics*, 86(2):347–358, 1997.
- Francisco Eiras, Adel Bibi, Rudy Bunel, Krishnamurthy Dj Dvijotham, Philip Torr, and M. Pawan Kumar. Provably correct physics-informed neural networks. arXiv preprint arXiv:2305.10157, 2023.
- Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II*, volume 14 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.
- Ernst Hairer, Gerhard Wanner, and Syvert P. Nørsett. *Solving Ordinary Differential Equations I*, volume 8 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.
- Nikos Hatziargyriou, Jovica Milanovic, Claudia Rahmann, Venkataramana Ajjarapu, Claudio Canizares, Istvan Erlich, David Hill, Ian Hiskens, Innocent Kamwa, Bikash Pal, Pouyan Pourbeik, Juan Sanchez-Gasca, Aleksandar Stankovic, Thierry Van Cutsem, Vijay Vittal, and Costas Vournas. Definition and Classification of Power System Stability Revisited & Extended. *IEEE Transactions on Power Systems*, 36(4):3271–3281, 2021.

- Birgit Hillebrecht and Benjamin Unger. Certified machine learning: A posteriori error estimation for physics-informed neural networks. In *International Joint Conference on Neural Networks*, *IJCNN 2022, Padua, Italy, July 18-23, 2022*, pages 1–8. IEEE, 2022.
- Arieh Iserles. A First Course in the Numerical Analysis of Differential Equations. Cambridge University Press, 2 edition, November 2008.
- Ioannis Konstantelos, Geoffroy Jamgotchian, Simon H. Tindemans, Philippe Duchesne, Stijn Cole, Christian Merckx, Goran Strbac, and Patrick Panciatici. Implementation of a massively parallel dynamic security assessment platform for large-scale grids. *IEEE Transactions on Smart Grid*, 8 (3):1417–1426, 2017.
- Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.
- I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.
- Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.
- Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.
- Christian Moya and Guang Lin. DAE-PINN: a physics-informed neural network model for simulating differential algebraic equations with application to power networks. *Neural Computing and Applications*, 35(5):3789–3804, 2023.
- Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378(C), 2018.
- Peter W. Sauer and M. A. Pai. *Power system dynamics and stability*. Prentice Hall, Upper Saddle River, N.J, 1998.
- Jochen Stiasny. *Physics-Informed Neural Networks for Power System Dynamics*. PhD thesis, Technical University of Denmark, 2023.
- Jochen Stiasny, Baosen Zhang, and Spyros Chatzivasileiadis. PINNSim: A simulator for power system dynamics based on physics-informed neural networks. arXiv preprint arXiv:2303.10256, 2023.
- Steven H. Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering.* Westview Press, a member of the Perseus Books Group, Boulder, CO, second edition edition, 2015.
- Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.