On the Role of Fixed Points of Dynamical Systems in Training Physics-Informed Neural Networks

Anonymous authors
Paper under double-blind review

Abstract

This paper empirically studies commonly observed training difficulties of Physics-Informed Neural Networks (PINNs) on dynamical systems. Our results indicate that fixed points which are inherent to these systems play a key role in the optimization of the in PINNs embedded physics loss function. We observe that the loss landscape exhibits local optima that are shaped by the presence of fixed points. We find that these local optima contribute to the complexity of the physics loss optimization which can explain common training difficulties and the resulting nonphysical predictions. Under certain settings, e.g., initial conditions close to fixed points or long simulations times, we show that those optima can even become better than that of the desired solution.

1 Introduction

Dynamical systems are governed by differential equations and are ubiquitous in many scientific disciplines such as economics, biology, physics and engineering. The upsurge in scientific machine learning has led to the development of deep learning approaches that are applicable to those systems and often superior to classical methods. State-of-the-art methods incorporate (at least) some part of the underlying physics, e.g., learned through data or embedded by design (Brunton et al., 2016; Sanchez-Gonzalez et al., 2020). Among those methods are *physics-informed neural networks* (PINNs) which are the prime paradigm of physics-informed machine learning (Raissi et al., 2019; Karniadakis et al., 2021). Their seamless integration of data and physical constraints has pushed PINNs into a vast number of applications on dynamical systems, including system identification (Raissi, 2018), hidden state inference (Raissi et al., 2020) and surrogate modeling (Sun et al., 2020).

Since PINNs are capable of solving differential equations in a fully mesh-free and time-continuous manner, one promising field of application is the numerical simulation of dynamical systems. In those applications, labeled training data is scarce and typically only used to specify the corresponding initial and boundary conditions (IC/BC). For a complete and unique definition of the forward problem, the IC/BC are either included in the loss function or explicitly enforced using specific network architectures. Both variants, however, rely on the optimization of the embedded physics loss function, i.e., on minimizing residuals on the governing differential equations, evaluated at collocation points which are randomly sampled inside the computational domain. Optimization success and accuracy thus particularly depend on the complexity of the studied dynamical system and the corresponding physics loss function.

1.1 Training Difficulties of Physics-Informed Neural Networks

In general, issues in the optimization of PINNs are manifold and often cause incorrectly predicted system dynamics. A complete description of all reported issues is exhaustive, hence we focus our discussion on relevant issues and proposed remedies that frequently appear in the numerical simulation of dynamical systems using PINNs.

Conflicting Objectives.

Several ongoing discussions address conflicts in the optimization of multiple objectives as one root cause of convergence issues in PINNs (Wang et al., 2021a). It has been shown that different weighting of the objectives, bound to physical, IC/BC and data constraints, is effective for PINNs and can accomplish a successful training. The weighting is conducted either by hand-tuned loss weights or with adaptive weighting schemes that adjust the weights during network training, such as in Maddu et al. (2021) or Jin et al. (2021). With a focus on coping with imbalanced gradients, those methods are generally used to improve the PINN's performance and to select an optimum point on the Pareto front (Rohrhofer et al., 2021). Furthermore, specially-designed network architectures enable hard encoding of IC/BC and physical constraints (Lu et al., 2021; Raissi et al., 2019). These approaches circumvent conflicts by reducing the overall number of competing objectives.

Propagation Failure.

Most relevant for our discussion are recent works that focus on a purported failure mode of PINNs in which the learned system dynamics does not represent the solution that is specified by the IC/BC. A reason for this, it has been argued, is that propagation of the solution from the enforced conditions to interior points is disrupted for a certain region in the computational domain, which often yields the trivial (zero) solution (Daw et al., 2022). To mitigate this issue, several remedies have been proposed. One focus lies in improving network initializations to reduce the bias towards flat output functions, e.g., by learning in sinusoidal space (Wong et al., 2021). Another type of methods propose reweighting (Wang et al., 2022) or resampling (Leiteritz & Pflüger, 2021) of collocation points. In those methods, importance or density of collocation points propagates from the enforced conditions to interior points during network training which, in a causality-respecting manner, promises to mitigate the propagation failure. Since it is generally argued that with an increasing domain size the physics loss optimization becomes more complex (Krishnapriyan et al., 2021), other methods focus on the extent of the computational domain. Often referred to as sequence-to-sequence learning or domain decomposition, those methods comprise approaches that divide the original spatio-temporal domain into smaller subdomains which are easier to solve (Jagtap & Karniadakis, 2021). Furthermore, approximation issues of PINNs in the presence of high-frequency or multi-scale features have been explained by the spectral bias of PINNs with proposed remedies found in Wang et al. (2021b).

1.2 Our Contribution

As shown in the last section, the literature is abound with techniques that try to mitigate commonly observed training difficulties of PINNs – but explanations why training on dynamical systems often fails seems incomplete to us: We suspect that not only the trivial zero solution, but also fundamental properties, e.g., fixed points of dynamical systems play a key role in training (failures) of PINNs. Based on this, our contribution in this paper will be as follows.

- We illustrate training difficulties of PINNs on a complex dynamical system, namely the Navier-Stokes equations, and hypothesize that the observed nonphysical prediction is affected by a fixed point inherent to the system. (Section 3).
- We then show on two simple dynamical systems that stable *and* unstable fixed points contribute to the optimization complexity of the physics loss function and influence the rate of training success. (Section 4.1)
- We empirically demonstrate that under certain settings, e.g., IC close to fixed points and long simulation times, nonphysical predictions become economical with better minima than that of the desired solution. (Section 4.2)
- We further show that even when the IC is far from fixed points, the physics loss landscape is still being shaped by these points which form local optima/saddle points that might prevent a successful training. (Section 4.3)

2 Background

2.1 Dynamical Systems

The state u of a dynamical system can be described by a differential equation of the form:

$$u_t = \mathcal{F}[u], \tag{1}$$

where $u = u(t, x)$ in general depends on time $t \in [0, T]$ and space $x \in \Omega \subseteq \mathbb{R}^n$, u_t denotes the (partial) derivative of u w.r.t. time, and $\mathcal{F}$ is an arbitrary, potentially nonlinear differential operator dictating the system dynamics.

In the numerical simulation of dynamical systems, IC are imposed to define the initial state of the system through

$$u(t_0 = 0, x) = u_0(x), \quad \forall x \in \Omega. \tag{2}$$

If $\mathcal{F}$ contains spatial derivatives, additional BC are required to guarantee the uniqueness of the solution:

$$\mathcal{B}[u] = g(x, t), \quad \forall x \in \partial\Omega, \tag{3}$$

where $\mathcal{B}$ denotes a boundary operator.

2.2 Physics-Informed Neural Networks

Fully-connected neural networks (FC-NNs) are most common among PINNs due to their good trade-off between simplicity and expressive power. Thus, we use FC-NNs to approximate the unknown solution function of (1) with $u(t, x) \approx u(t, x; \theta) =: u_\theta(t, x)$, where $\theta \in \mathbb{R}^{n_\theta}$ are the weights and bias terms of the network. Common activation functions are the hyperbolic tangent (tanh) or Sigmoid linear unit (SiLU, swish), which render the approximated solution function and derivatives smooth[1].

PINNs use automatic differentiation (AD) (Baydin et al., 2018) to obtain (partial) derivatives of the network's output with respect to its inputs. In order to retrieve the derivatives of the network solution, AD requires to pass discrete evaluation points, called collocation points, through the network in a feed-forward operation. The collocation points define the data set for penalizing residuals of the differential equation. The physics loss residual for a dynamical system (1) is given by:

$$f(t, x) := u_{\theta,t}(t, x) - \mathcal{F}[u_\theta(t, x)]. \tag{4}$$

Following the standard PINN formulation, the sum of squared residuals at all collocation points yields the physics loss function:

$$L_f(\theta) = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| f(t^i, x^i) \right|^2, \tag{5}$$

with the collocation points $\{t^i, x^i\}_{i=1}^{N_f}$ sampled from the entire computational domain $(t, x) \in [0, T] \otimes \Omega$. These points do not need any label and can be either fixed during PINN training or re-sampled before each training epoch. In the initial formulation of PINNs, additional data constraints are used to enforce the IC/BC by

$$L_u(\theta) = \frac{1}{N_u} \sum_{i=1}^{N_u} \left| u_\theta(t^i, x^i) - u(t^i, x^i) \right|^2, \tag{6}$$

where u is given by the r.h.s. of (2) and (3). Both losses (5) and (6) are combined by scalarization of a multi-objective optimization through

$$L(\theta) = \lambda L_u(\theta) + L_f(\theta), \tag{7}$$

where λ represents a weighting factor, which here by default is set to $\lambda = 1$, unless explicitly stated otherwise. As an alternative to this (vanilla) formulation of PINNs, special network architectures have been proposed that ensure IC/BC are satisfied explicitly. We refer to these PINNs as being *hard constrained*.

[1] Thus mesh-free and time-continuous in the context of differential equations.

2.3 Stability and Fixed Points

Fixed points u^* of a dynamical system are given by the roots of the nonlinear function $\mathcal{F}$ in equation (1):

$$\mathcal{F}[u^*] = 0. \tag{8}$$

In general, they can be either stable, asymptotically stable or unstable. For a stable fixed point, any trajectory close to it will stay close, whereas for an asymptotically stable fixed point close trajectories will further converge to it as $t \to \infty$. In contrast, an unstable fixed point is repulsive and even the smallest deviation will cause any close trajectory move away from it as $t \to \infty$.

Some fixed points of dynamical systems are trivial, e.g., the zero solution $u^* = 0$ is a fixed point for many dynamical systems, such as for harmonic oscillators or pendulums. For dynamical systems governed by partial differential equations, such as in fluid flow, fixed points are non trivial but generally given by steady-state solutions that do not change over time and thus for which equation (8) holds true.

3 Motivation: Complex Fluid Dynamics

We use this motivation example to show commonly observed training difficulties and nonphysical behavior of PINNs when simulating complex fluid flow. In particular, we select a well-known benchmark setting, known as vortex shedding, and demonstrate that the PINN prediction u_θ resembles a steady-state behavior, although the true system dynamics is given by transient, here periodic, fluid motion.

Navier-Stokes Equations.

The Navier-Stokes equations govern fluid flow and build a coupled system of nonlinear partial differential equations (PDEs). For the two-dimensional case, the unknown solution functions are $u(t, \vec{x})$, $v(t, \vec{x})$ and $p(t, \vec{x})$, representing the fluid velocity in x- and y-direction and pressure, respectively. The system equations for transient fluid flow are given by

$$u_t = -(uu_x + vu_y) - p_x + \mathrm{Re}^{-1}(u_{xx} + u_{yy}), \tag{9a}$$

$$v_t = -(uv_x + vv_y) - p_y + \mathrm{Re}^{-1}(v_{xx} + v_{yy}), \tag{9b}$$

where we set the Reynolds number Re to 100.

Experimental Setup.

We introduce a stream function $\psi(t, \vec{x})$ with $u = \psi_y$ and $v = -\psi_x$ to enforce continuity, i.e., conservation of mass for an incompressible fluid. A single 8x100 neural network with tanh activation functions is used to approximate $\psi_\theta(t, \vec{x})$ and $p_\theta(t, \vec{x})$. Software and hardware specifications in use are found in Appendix A.

We use a publicly available database of direct numerical simulation data, found in Boudina (2021). This labeled dataset is used as reference solution and to impose the initial sequence of the fluid motion in the time domain $t \in [0, 3]$ representing 50% of a vortex shedding period. Collocation points, however, are sampled in the time domain $t \in [0, 18]$, i.e., the PINN should continue the simulation beyond the domain of reference data and capture the system dynamics by minimizing residuals of (9). Further details on optimization and data settings, as well as the computational domain and BC in use, can be found in Appendix B.

PINN Approaches Steady State.

Results to this experiment are presented in Figure 3. We observe that outside the domain of reference data, the PINN's prediction (b) starts to substantially deviate from the reference solution (a). Evaluating the PINN prediction at the depicted spatial coordinate (c) further shows that the solution approaches a nonphysical steady state. This is interesting as the symmetrical steady-state solution to this problem resembles an unstable fixed point. The fixed point is unstable for this Reynolds number because any perturbation (typically caused by numerical instabilities in classical methods) will lead to the development of the periodic motion,

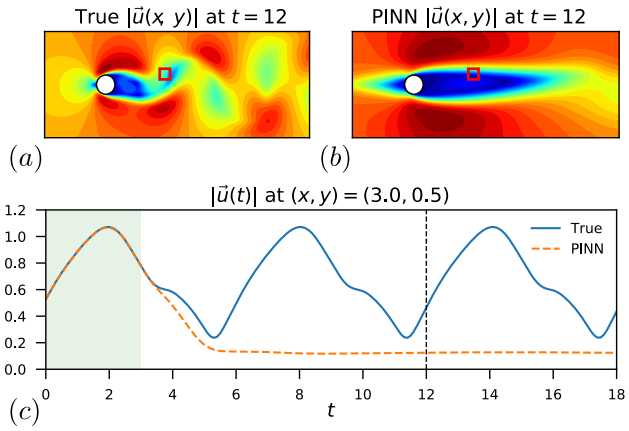

Figure 1: **Fluid Dynamics.** Substantial difference in velocity magnitudes $|\vec{u}|$ given by (a) reference data and predicted by the (b) PINN after two consecutive periods (vertical dashed line). (c) The time evolution of $|\vec{u}|$ at a spatial point (red rectangle) indicates that the PINN approaches a steady-state solution. Labeled training data was used for 50% of the first vortex shedding period (green shaded area).

known as vortex shedding. Similar PINN behavior has been reported in Chuang & Barba (2022) and similar issues can be resolved using truncated Fourier decomposition presented in Raynaud et al. (2022).

We note that evaluating the physics loss for the depicted PINN prediction and for a solution obtained by training a PINN on data from the entire computational domain, i.e., on $t \in [0, 18]$, shows that the nonphysical steady-state prediction achieves a lower physics loss than that capturing the correct system dynamics (see Appendix B.4). This behavior suggests that the PINN training is affected by the underlying fixed point, and we hypothesize that this partly explains the commonly observed slow convergence or attraction to nonphysical predictions.

4 On the Role of Fixed Points in Training Physics-Informed Neural Networks

We now perform experiments to test the hypothesis that fixed points play a key role in the training of PINNs. For the sake of simplicity and tractability, we consider two simple ordinary differential equations (ODEs): the undamped pendulum dynamics and a simple toy example. These examples are chosen because they exhibit stable (pendulum), asymptotically stable (toy example), and unstable (both) fixed points. We further try to solve these systems using either vanilla (i.e., multi-objective, for the pendulum) and hard constrained (for the toy example) PINNs. For both examples we now stick to the convention of ODEs and denote the unknown solution function by $y(t)$.

Undamped Pendulum Equation.

The undamped pendulum dynamics are given by a second-order ODE

$$\ddot{y} = -\frac{g}{l} \sin(y), \tag{10}$$

with l and g representing the length of the rod and magnitude of the gravitational field, respectively. Here $l = 1$ and $g = 9.81$. This system exhibits two fixed points, a stable fixed point $y^* = 0°$ at the pendulum's natural rest position, and an unstable fixed point $y^* = 180°$ at the upright position. For comparison in our experiments, we create reference solutions using a Runge-Kutta fourth-order method.

Toy Example Equation.

The toy example is defined as a one-dimensional system with a single ODE given by

$$\dot{y} = y\left(1 - y^2\right). \tag{11}$$

Table 1: **Undamped Pendulum.** Rate of training success across different system settings (T and y_0) and network architectures (size and activation function). Triplets in the main table represent in percentage (%) and in the respective order, cases of successful training, attracted by stable fixed point, and unstable fixed point. Bold triplets represent a low ($< 5\%$) success rate.

T		2.5			5			7.5		
y_0		25°	100°	175°	25°	100°	175°	25°	100°	175°
4x50	tanh	98/2/0	100/0/0	100/0/0	**0/100/0**	90/10/0	**0/100/0**	**0/100/0**	**0/100/0**	**0/61/39**
	swish	100/0/0	100/0/0	98/0/2	56/44/0	80/20/0	**0/100/0**	**0/100/0**	**1/99/0**	**0/91/9**
	sin	100/0/0	100/0/0	100/0/0	65/35/0	93/7/0	**0/93/7**	**2/98/0**	24/75/1	**0/94/6**
8x100	tanh	100/0/0	100/0/0	99/0/1	100/0/0	97/0/3	92/0/8	49/31/20	84/0/16	**1/29/70**
	swish	100/0/0	100/0/0	100/0/0	100/0/0	100/0/0	57/42/1	100/0/0	100/0/0	**1/92/7**
	sin	100/0/0	100/0/0	98/0/2	55/0/45	60/0/40	39/15/46	32/0/68	29/0/71	**0/26/74**

This system exhibits three fixed points, two of which located at $y^* = \pm 1$ are asymptotically stable, and one at $y^* = 0$ which is unstable (see Figure 3). For this system, the analytical solution exists and is given in Appendix D.1. The simplicity of this toy example further allows for hard constraining the IC by setting

$$\hat{y}(t) = y_0 + t \cdot y_\theta(t), \tag{12}$$

with y_θ denoting the network's output.

4.1 Rate of Training Success in the Presence of Fixed Points

We now study the success rate of training PINNs on the above mentioned systems. We declare training successful if the L_2 relative error $\|y_\theta(t) - y(t)\|_2 / \|y(t)\|_2$ is below 15%, which allows a clear separation of training outcomes[2]. For the toy example nonphysical predictions are exclusively influenced by the unstable fixed point at $y^* = 0$ (see Figure 3(c)). For the undamped pendulum, however, we further classify whether an unsuccessful training, subsequently the nonphysical prediction, violates the physics by either being attracted by the stable ($y^* = 0°$) or the unstable fixed point ($y^* = 180°$). Examples for those cases are found in Figure 2 which also visually demonstrates the attraction in phase space.

For both systems, we train 100 randomly initialized PINNs with a different IC y_0 and simulation time T. The IC for the undamped pendulum are enforced using the multi-objective loss (7) (vanilla PINN) with a zero initial angular velocity, i.e., $\dot{y}_0 = 0$. Training is performed for 50k epochs using the Adam optimizer with default settings for the moment estimates. We repeat training for different setups, including network architectures and optimizer settings (see Appendix C.1 for details). Software and hardware specifications in use are found in Appendix A.

Table 1 shows the rate of training success across different network architectures for the experiments on the undamped pendulum and an initial learning rate of $\alpha = 0.001$. The outcome of experiments using different optimization settings and for the toy example can be found in Appendix C.1 and D.2. In general, we observe severe training difficulties across all experiments with only a minor number of cases leading to a success rate of 100%.

Influence of Initial Condition y_0.

From the results it is apparent that PINNs are sensitive to the choice of initial conditions. We observe that, in general, IC close to fixed points lead to a lower rate of training success compared to those that start far. For the undamped pendulum this is evident by comparing in Table 1 the rates for $y_0 = 100°$ with that of $y_0 = 25°$ and $y_0 = 175°$. We made similar observations for the toy example that also shows a lower rate of success for IC close to the unstable fixed point (see Appendix D.2).

[2]Results for further thresholds are provided in Appendix C.2 and suggest similar qualitative conclusions.

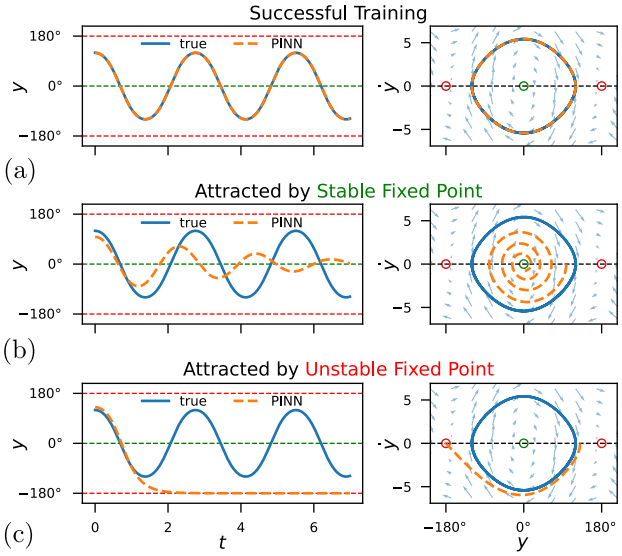

Figure 2: **Undamped Pendulum.** Representative examples of training cases from Table 1. Training outcomes are classified as either (a) successful training, (b) attracted by stable fixed point, or (c) unstable fixed point. *Left*: predicted trajectories as function of physical time. *Right*: predicted trajectories in phase space. Green and red lines/circles represent stable and unstable fixed points, respectively.

Influence of Simulation Time T.

Across different settings and for both systems, we observe that training becomes more difficult as the simulation time is increased. Likewise it can be seen that reducing the simulation time accounts for a higher success rate. The influence of the simulation time in terms of the physics loss complexity will be further analyzed in Section 4.3.

Influence of Network and Optimization Settings.

We observe a slight improvement in terms of a higher success rate as the network size is increased. Still, none of the tested network architectures could resolve the training issues at long simulation times and IC close to the (unstable) fixed point. Thus, we conclude that the observed training difficulties are bound to the optimization complexity, rather than insufficient expressive power of the network. As reported in Appendix C.1, we also perform an ablation study using different optimization settings in terms of the learning rate, number of collocation points, loss weighting and network initialization. In comparison to the baseline model (4x50, tanh, from Table 1) no optimization setting yields notable changes to the rate of training success.

4.2 Fixed Points Becoming Economical Solutions

Next, we demonstrate that, when the IC is very close to a fixed point, training may result in PINN predictions that approach these fixed points, even if the resulting behavior is nonphysical. Furthermore, we show that those nonphysical predictions can even become better minima than that of the desired optimum. The latter further renders convergence to the true solution unfeasible. To rule out effects from multi-objective optimization, we focus in this and the following subsection on the toy example implemented by a PINN with hard constraints. Similar observations on the undamped pendulum can be found in Appendix C.3.

For this experiment, we use a 4x50 network architecture with tanh activation and choose a simulation time of $T = 10$. To show that our chosen PINN architecture has sufficient expressive power to learn the true solution, we implemented the following, *data-guided* PINN as control: First, 10 labeled data points

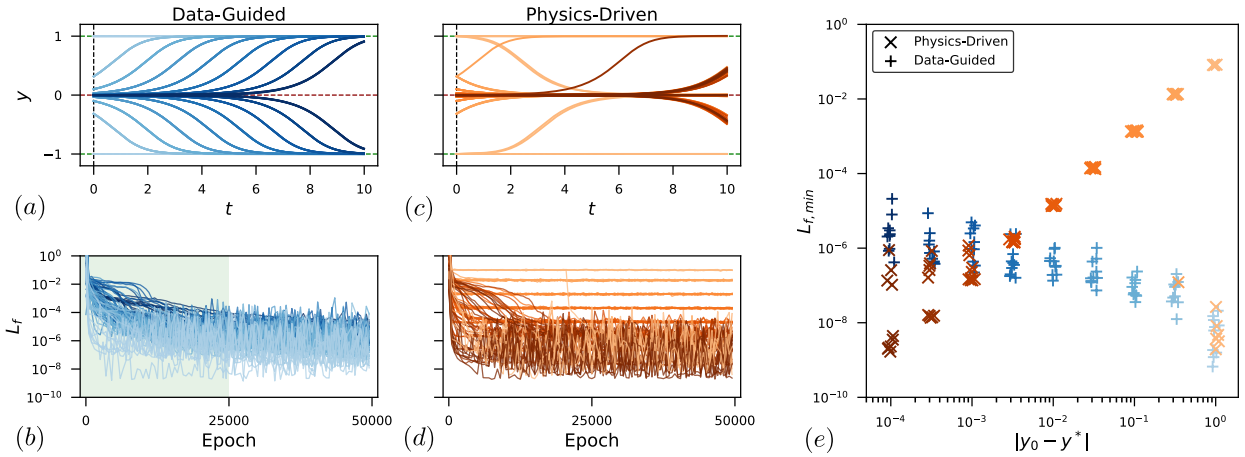

Figure 3: **Toy Example.** (a),(b) *Data-guided* PINN: predicted trajectories and learning curves with green shaded area representing the support of labeled training examples through (7). (c),(d) *Physics-driven* PINN: predicted trajectories and learning curves. (e) Minimal physics loss across all epochs vs. absolute distance between IC and the unstable fixed point. Five PINN instances were trained for each initial value (sequential color code) and approach (crosses and pluses). In the figure, markers were randomly shifted horizontally to reduce overlap.

are sampled from the analytical solution, equidistantly from the computational domain. In the first 25k epochs of training, these 10 points support training via (7), while in the remaining 25k epochs, the PINN is trained using the physics loss only. The first training phase thus guides the gradient-based optimization into the basin of attraction of the analytical solution, while the second training phase ensures that the physics loss is minimized. Indeed, as shown in Figure 3(b), the data-guided strategy successfully learns the analytical solution. We compare this approach to training with the physics loss only, i.e., without the use of labeled data, which we refer to as the *physics-driven* PINN. For each approach and IC we train five uniquely initialized PINN instances with the same optimization settings found in 4.1.

Influence of Initial Condition y_0.

For the toy example, Figure 3(c)-(e) show that the PINN predictions impacted by the unstable fixed point ($y^* = 0$) achieve lower physics losses as the IC gets closer to it. Indeed, for small values of y_0, the physics loss can become even smaller than the physics loss achieved by the data-guided PINN (see Figure 3(e)), i.e., the prediction impacted by the unstable fixed point seems to become a better minimum for PINN optimization than the true solution. This renders finding the true solution unfeasible, even for optimization methods that are not based on gradients (e.g., PSO-PINNs as in Davi & Braga-Neto (2022) or else).

4.3 Fixed Points Affect the Optimization Landscape and Slow Down Convergence.

Finally, we investigate the effect of fixed points on the physics loss landscape, and argue that our observations can partly explain commonly observed slow convergence. In Figure 3(d) we observe that most PINN instances suffer from slow convergence, indicated by the presence of plateaus in the learning curves. Only a few examples show a successful escape from those undesired optima which is evident by a distinct drop in the learning curves. Those cases are primarily reported at IC that are comparably far from the unstable fixed point at $y^* = 0$. To obtain a better understanding of this phenomenon, we plot the physics loss landscape and PINN prediction for a instance that barely manages to escape from its suboptimal location.

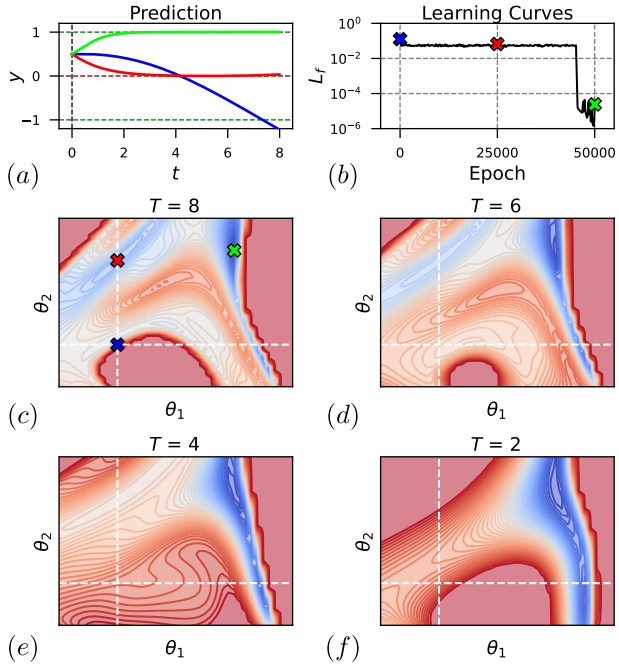

Figure 4: **Toy Example.** (a) PINN prediction and (b) corresponding learning curves for a training example that barely manages to escape from its suboptimal location. (c)-(f) Physics loss landscape for different T. Crosses denote (blue) initial, (red) intermediate, and (green) final step of the network training.

Influence of Simulation Time T.

Figure 4 shows the loss landscape for two directions θ_1 and θ_2 for an IC that is far from the unstable fixed point (visualization details are given in Appendix E). Additionally, in panel (a) and (b) a prediction and training sequence is given for $T = 8$ where the gradient-based optimization first gets trapped in a local optimum. After approximately 42,000 epochs, the gradient-based optimization manages to converge to the correct solution. We clearly detect the global minimum which corresponds to this true solution in the upper right region (green cross).

We further observe that for long simulation times, i.e. in panel (c)-(e), a local minimum or saddle point (red cross) forms, which seems to be attractive to the gradient-based optimization (cf. Figure 3) and apparently conforms to a nonphysical prediction. With decreasing T, this local optimum gradually vanishes and the global optimum becomes easier to reach due to the better (in terms of optimization) shape of the loss landscape. Thus, we argue that (I) fixed points have a large impact on the shape of the physics loss landscape, even when the IC is far from the (unstable) fixed point, and (II) simulating for a longer time, i.e., being further away from (possibly) enforced IC, results in a higher chance that the optimization gets trapped in local optima which conform to nonphysical predictions.

5 Discussion and Limitations

Our study and results suggest that fixed points of dynamical systems, irrespective of whether they are stable or unstable, lead to the formation of attractive optima in the physics loss landscape. This was demonstrated on a series of experiments using two dynamical systems described by ODEs, and we believe that some of our results also carry over to more complicated systems described by PDEs (Section 3).

On the Role of Fixed Points.

As discussed in Section 2.3, fixed points of dynamical systems are given by the roots of the nonlinear function $\mathcal{F}$ in Equation (1). Physics loss residuals (4) are thus small by definition since $\mathcal{F}[u]$, and thus u_t, is close to zero in the vicinity of fixed points. These properties seem to affect training of PINNs: True fixed point solutions, i.e., solutions that are constant or steady-state, trivially fulfill the physics loss function and, thus, the fixed points yield local minima in the physics loss landscape which are attractive to the gradient descent optimization, as we could show in our experiments. Additionally, the fact that residuals are small in the vicinity of fixed points may make it more economical for the PINN to violate prescribed system dynamics locally, for the sake of settling at a "simpler" solution. Effectively, fixed points, together with the L_2 losses in (7), allow PINNs to trade between severely violating physics locally or approximating it inaccurately on the entire computational domain.

Limitations and Further Work.

One may argue that the trade-off inherent in PINNs, or the multi-objective nature of their vanilla formulation (7), should be vacuous, as the true solution satisfies both physics and the IC/BC. Thus, nonphysical predictions should never represent a better optimum than the desired, physical solution. This is true, however, only for PINNs with unlimited expressive power. In practice, the expressivity of a neural network is always limited by its (necessarily) finite size. Therefore, the mentioned trade-off is effective, and we have reason to believe that there are settings where the desired solution does not correspond to a global optimum (cf. Figure 3). Future work shall investigate this aspect from a more theoretical perspective, instantiating approximation theorems for neural networks for PINNs.

Further, one may argue that some of the observed nonphysical predictions simply appear because the PINN has not sufficiently converged. In other words, training was not long enough to depart from the (flat) IC, which may correspond to a trivial solution of the differential equation (Wong et al., 2021; Leiteritz & Pflüger, 2021). A large part of the literature on propagation failures (see Section 1.1) points in this direction. Further, such a statement is supported by the fact that the physics loss of, e.g., the solution approaching the stable fixed point in Figure 2 is high, and by the late transition to the correct solution in Figure 4. Indeed, we do not claim that minima of the physics loss formed by the presence of fixed points *always* correspond to (good) minima of the full loss (7) – these minima may disappear entirely (e.g., for small computational domains), turn into saddle points that slow down convergence, or achieve a wider basin of attraction and/or smaller loss than the minimum corresponding to the true solution, in the extreme case where ICs are very close to unstable fixed points.

Finally, our investigations are based on two simple ODEs, with additional evidence by a benchmark PDE problem. We chose these systems because they are intuitive to understand, yet still exhibit nontrivial dynamics. Moreover, the simplicity of these systems allowed us to separate the effect of different types of fixed points and to at least partly exclude other explanations for the training difficulties of PINNs. Future work shall be devoted to studying fixed points and steady-state solutions, and to the wider spectrum of asymptotic properties of solutions to dynamical systems.

6 Conclusion

In this paper, we studied the physics loss optimization in PINNs when applied to dynamical systems governed by differential equations. Our results revealed that nonphysical predictions appear as attractive optima in the physics loss landscape and seem to stem from the presence of fixed points inherent to dynamical systems. These minima or saddle points potentially disrupt and trap the gradient descent optimization, leading to commonly observed convergence issues in PINNs. Reducing the computational domain yielded a greater rate of training success and, in general, reduced the complexity of the physics loss optimization. In the future, we believe that interdisciplinary research that includes advances in deep learning, stability theory and/or a further understanding of the underlying physics may improve physics-informed machine learning or benefit from it.

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, 2018.

Mouad Boudina. Numerical simulation data of a two-dimensional flow around a fixed circular cylinder, June 2021. URL `https://doi.org/10.5281/zenodo.5039610`.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113 (15):3932–3937, 2016.

Pi-Yueh Chuang and Lorena A Barba. Experience report of physics-informed neural networks in fluid simulations: pitfalls and frustration. *arXiv preprint arXiv:2205.14249*, 2022.

Caio Davi and Ulisses Braga-Neto. Pso-pinn: Physics-informed neural networks trained with particle swarm optimization. *arXiv preprint arXiv:2202.01943*, 2022.

Arka Daw, Jie Bu, Sifan Wang, Paris Perdikaris, and Anuj Karpatne. Rethinking the importance of sampling in physics-informed neural networks. *arXiv preprint arXiv:2207.02338*, 2022.

Ameya D Jagtap and George E Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. In *AAAI Spring Symposium: MLPS*, 2021.

Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 426:109951, 2021.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

Raphael Leiteritz and Dirk Pflüger. How to avoid trivial solutions in physics-informed neural networks. *arXiv preprint arXiv:2112.05620*, 2021.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

Lu Lu, Raphael Pestourie, Wenjie Yao, Zhicheng Wang, Francesc Verdugo, and Steven G Johnson. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021.

Suryanarayana Maddu, Dominik Sturm, Christian L Müller, and Ivo F Sbalzarini. Inverse Dirichlet Weighting Enables Reliable Training of Physics Informed Neural Networks. *Machine Learning: Science and Technology*, 2021.

Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

Gaétan Raynaud, Sebastien Houde, and Frederick P Gosselin. Modalpinn: an extension of physics-informed neural networks with enforced truncated fourier decomposition for periodic flow reconstruction using a limited number of imperfect sensors. *Journal of Computational Physics*, pp. 111271, 2022.

Franz M Rohrhofer, Stefan Posch, and Bernhard C Geiger. On the Pareto Front of Physics-Informed Neural Networks. *arXiv preprint arXiv:2105.00862*, 2021.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pp. 8459–8468. PMLR, 2020.

Luning Sun, Han Gao, Shaowu Pan, and Jian-Xun Wang. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. *Computer Methods in Applied Mechanics and Engineering*, 361:112732, 2020.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021a.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021b.

Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*, 2022.

Jian Cheng Wong, Chinchun Ooi, Abhishek Gupta, and Yew-Soon Ong. Learning in sinusoidal spaces with physics-informed neural networks. *arXiv preprint arXiv:2109.09338*, 2021.

A Software and Hardware Specifications

All code in our experiments is implemented in Python version 3.8 and TensorFlow version 2.9.1. Computations are performed on a Nvidia Tesla T4 GPU with a memory size of 16 GB.

B Motivation: Complex Fluid Dynamics

In this section we provide further information to the experimental setup of the motivation example. We use a publicly available database of direct numerical simulation data, found in Boudina (2021). The computational domain for our experiment is restricted to $(x, y) \in \Omega := [-5, 15] \otimes [-10, 10]$.

B.1 Boundary Conditions

We show the BC in Figure 5. Here, the top/bottom boundary is considered as a moving wall with no-slip conditions. We use zero-gradient conditions for the outlet. We note that we have also tested different BC for the top/bottom boundary (symmetry wall, zero-gradient), and for the outlet (zero pressure), but none of the considered settings led to the desired vortex shedding motion.

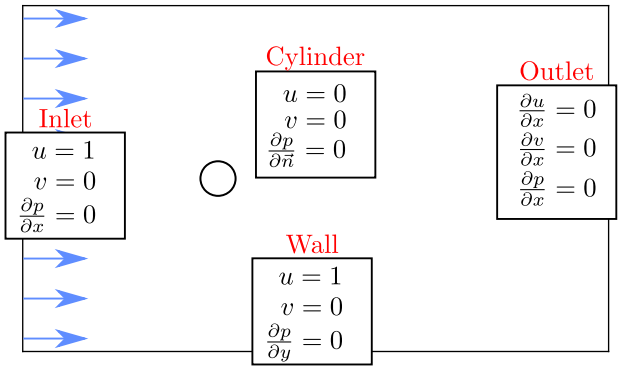

Figure 5: **Fluid Dynamics.** Boundary conditions for vortex shedding.

B.2 Training Data

Training data for this experiment is used to impose the initial sequence from the reference data, the respective BC, as well as the physical constraints (9) through a set of collocation points. The training data set for the initial sequence comprises in total $N_{IC} = 100.000$ data points which are sampled randomly during network training with a batch size of $N_{IC,\text{batch}} = 1024$. At each batch iteration, collocation points, and training data for the BC are sampled anew with sizes $N_{col} = 1024$, $N_{\text{Inlet}} = 128$, $N_{\text{Outlet}} = 128$, $N_{\text{Wall}} = 256$ and $N_{\text{Cylinder}} = 128$.

B.3 Loss Function and Optimization Settings

The overall loss function in this experiment is composed of the respective loss functions for the IC/BC and physical constraints:

$$L(\theta) = L_{IC}(\theta) + L_{\text{Inlet}}(\theta) + L_{\text{Outlet}}(\theta)$$
$$+L_{\text{Wall}}(\theta) + L_{\text{Cylinder}}(\theta) + L_{f,x}(\theta) + L_{f,y}(\theta),$$

where $L_{f,x}$ and $L_{f,y}$ are the physics loss functions for (9a) and (9b), respectively. As already stated in the main part of this work, we introduce a stream function $\psi(t, \vec{x})$ with $u = \psi_y$ and $v = -\psi_x$ to enforce

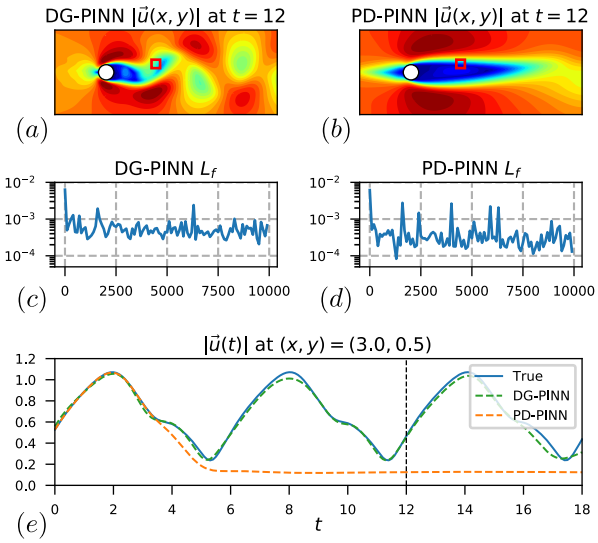

Figure 6: **Fluid Dynamics.** Comparison between a data-guided (DG) and standard physics-driven (PD) PINN. (a), (b) Predicted velocity magnitudes $|\vec{u}|$ after two consecutive periods (vertical dashed line). (c), (d) Learning curves for the physics loss L_f. (e) Time evolution of $|\vec{u}|$ at a spatial point (red rectangle). While the DG-PINN achieves a minimal physics loss of $L_{f,\text{min}} = 2.07 \cdot 10^{-4}$, that of the PD-PINN is $L_{f,\text{min}} = 8.28 \cdot 10^{-5}$.

continuity, i.e., conservation of mass for an incompressible fluid. A single 8x100 neural network with tanh activation functions is then used to approximate $\psi_\theta(t, \vec{x})$ and $p_\theta(t, \vec{x})$.

Optimization is performed using Adam with default settings for the moment estimates and a total number of 10k epochs. The initial learning rate is set to $\alpha = 0.001$ and an exponential decay with rate 0.9 and step 1000 is applied. Training for this setting and for the in Section A listed software/hardware took about $7h$.

B.4 Comparison to Data-Guided PINN

In the main part of this work, we claimed that evaluating the physics loss for the physics-driven PINN (presented in Figure 3) and for a solution obtained by training a data-guided PINN on data from the entire computational domain, i.e., on $t \in [0, 18]$, shows that the nonphysical steady-state prediction achieves a lower physics loss than that capturing the correct system dynamics.

Results to this claim can be found in Figure 6. We note that both PINN instances use the same physics loss function $L_f = L_{f,x} + L_{f,y}$, evaluated at collocation points in the full domain $t \in [0, 18]$. We observe in Figure 6(c)-(d) that the minimal physics loss value across all epochs is lower for the physics-driven PINN compared to that of the data-guided PINN: While the physics-driven prediction that approaches a steady-state solution yields a minimal physics loss of $L_{f,\text{min}} = 8.28 \cdot 10^{-5}$, that of the data-guided PINN indeed resembles the vortex shedding motion with a minimal physics loss of $L_{f,\text{min}} = 2.07 \cdot 10^{-4}$.

C Additional Content to Undamped Pendulum

This section provides further results in addition to the in Section 4.1 and 4.2 presented experiments.

C.1 Rate of Training Success - Optimization Settings

In Table 1, all PINN instances use an initial learning rate $\alpha = 0.001$, number of collocation points $N_c = 64$, loss weighting factor $\lambda = 1$ and as network initialization the Glorot uniform initializer. In addition to this,

Table 2: **Undamped Pendulum.** Rate of training success across different system and optimization settings using the 4x50 network architecture with tanh activation. Triplets in the main table represent in percentage (%) and in the respective order, cases of successful training, attracted by stable fixed point, and unstable fixed point. The tested optimization settings are learning rate (α), number of collocation points (N_c), loss weighting factor (λ) and network weights initialization (*Init.*) with He denoting the *He uniform* initialization. Baseline model (see Table 1) uses $\alpha = 0.001$, $N_c = 64$, $\lambda = 1$ and Glorot uniform initialization. Bold triplets represent a low ($< 5\%$) success rate.

T		2.5			5			7.5		
y_0		25°	100°	175°	25°	100°	175°	25°	100°	175°
Baseline		98/2/0	100/0/0	100/0/0	**0/100/0**	90/10/0	**0/100/0**	**0/100/0**	**0/100/0**	**0/61/39**
α	0.01	37/0/63	36/0/64	60/0/40	16/0/84	19/0/81	7/0/93	**0/7/93**	13/0/87	**0/0/100**
	0.001	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**
N_c	16	96/4/0	100/0/0	100/0/0	**0/100/0**	**4/96/0**	**0/100/0**	**0/100/0**	**0/100/0**	**0/100/0**
	256	100/0/0	100/0/0	100/0/0	**0/100/0**	100/0/0	35/64/1	**0/99/1**	**0/100/0**	**0/89/11**
λ	0.1	**0/100/0**	61/39/0	93/7/0	**0/100/0**	**0/99/1**	**0/85/0**	**0/100/0**	**0/100/0**	**0/100/0**
	10	100/0/0	100/0/0	100/0/0	11/89/0	42/0/0	6/85/9	**0/99/1**	**0/100/0**	**0/35/65**
Init.	He	100/0/0	98/0/2	100/0/0	**0/98/2**	93/7/0	**0/98/2**	**0/98/2**	**0/98/2**	**0/73/27**

we also test different optimization settings using the 4x50 network architecture with tanh activation as base model.

Results to this experiment can be found in Table 2, where we included the baseline model with default optimization settings from Table 1 as reference. In general, we observe that none of the tested optimization settings yields substantial improvement for IC close to fixed points and long simulation times.

C.2 Rate of Training Success - Further Thresholds

In the main part of the paper, we use for the classification of successful training a threshold of 15% in terms of the L_2 relative error. To demonstrate that our particular choice of this threshold does not contradict qualitative conclusions made in the main part, we further provide results using different thresholds. In particular, we validate the in Table 1 presented training cases now with a threshold of 5% and 25% in terms of the L_2 relative error.

The results can be found in Table 3. For compactness, we only show the results for the 4x50 and 8x100 architecture with tanh activation. As apparent in the table, no substantial differences in the classified training outcomes can be observed when using different thresholds.

C.3 Fixed Points Becoming Economical Solutions

In the main part of this work, Figure 3 shows for the toy example that nonphysical predictions can become better minima than that of the desired solution when the IC is close to a fixed point. To demonstrate similar qualitative observations for the undamped pendulum, we perform an experiment similar to that in Section 4.2.

In particular, we use a 8x100 network architecture with tanh activation and choose a simulation time of $T = 10$. Since for this simulation time physics-driven PINN instances will hardly converge to the true solution, we implement the same data-guided strategy as presented in Section 4.2: We include a total number of 100 labeled training points in the first half of the training. The data is sampled from the Runge-Kutta solution, equidistantly in the computational domain. In the second half, the training continues with the physics loss optimization only. Indeed, as show in Figure 7(b), the data-guided strategy successfully converges to the true solution, representing successful outcomes and sufficient expressive power for the chosen PINN architecture. We again compare this approach to physics-driven training, i.e., without the use of labeled training data. We repeat for each IC the experiment with 10 uniquely initialized PINN instances per training strategy. We note that none of the physics-driven instances converges to the true solution

Table 3: **Undamped Pendulum.** Rate of training success using differently set thresholds in terms of the L_2 relative error for the classification of successful and unsuccessful training. Triplets in the main table represent in percentage (%) and in the respective order, cases of successful training, attracted by stable fixed point, and unstable fixed point. Bold triplets represent a low ($< 5\%$) success rate.

	T	2.5			5			7.5		
	y_0	25°	100°	175°	25°	100°	175°	25°	100°	175°
4x50	$L_2 < 5\%$	98/2/0	100/0/0	100/0/0	**0/100/0**	90/10/0	**0/100/0**	**0/100/0**	**0/100/0**	**0/61/39**
	$L_2 < 15\%$	98/2/0	100/0/0	100/0/0	**0/100/0**	90/10/0	**0/100/0**	**0/100/0**	**0/100/0**	**0/61/39**
	$L_2 < 25\%$	99/1/0	100/0/0	100/0/0	**0/100/0**	90/10/0	34/66/0	**0/100/0**	**0/100/0**	**0/61/39**
8x100	$L_2 < 5\%$	43/37/20	84/0/16	**0/29/71**	100/0/0	97/0/3	43/49/8	43/37/20	84/0/16	**0/29/71**
	$L_2 < 15\%$	100/0/0	100/0/0	99/0/1	100/0/0	97/0/3	92/0/8	49/31/20	84/0/16	**1/29/70**
	$L_2 < 25\%$	100/0/0	100/0/0	99/0/1	100/0/0	97/0/3	92/0/8	55/25/20	84/0/16	**1/29/70**

(see Figure 7(b)). The unsuccessful training outcomes are further classified into whether the subsequent nonphysical prediction violates the physics by either being attracted by the stable or the unstable fixed point (see Figure 2).

In Figure 7(a) we report the minimal physics loss values across all epochs for each training outcome. We observe, similar to the behavior in the toy example, that the PINN predictions attracted by the unstable fixed point ($y^* = 180°$) achieve lower physics losses as the IC gets closer to it. Furthermore, for $y_0 = 175°$ the nonphysical solution becomes a better optimum than that of the desired solution. As a direct consequence, any physics-driven instance converges to it.

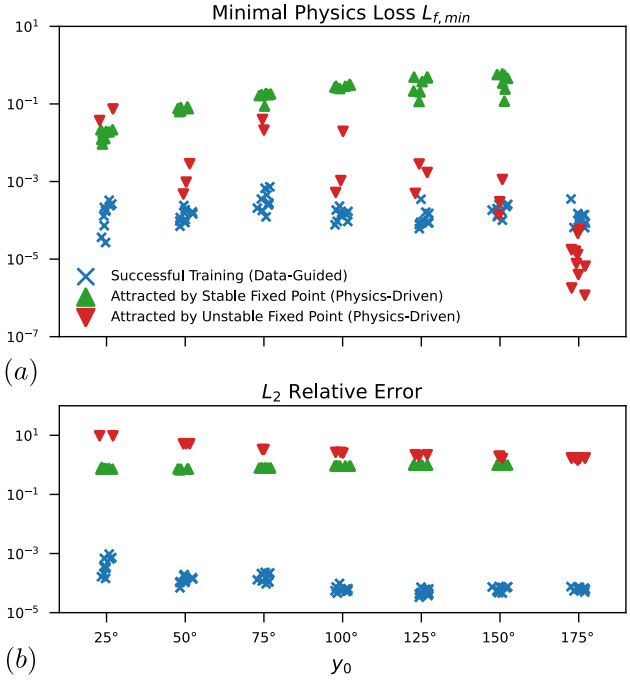

Figure 7: **Undamped Pendulum.** (a) Minimal physics loss across all epochs. (b) L_2 relative error. While the data-guided PINNs converge to the true solution (blue), the physics-driven PINNs yield incorrect system dynamics (large L_2 relative errors) by being either attracted by the stable (green) or unstable (red) fixed point (see Figure 2). For $y_0 = 175°$ the nonphysical solution becomes a better optimum than that of the desired solution. In the figure, markers were randomly shifted horizontally to reduce overlap.

T		2.5			5			7.5		
y_0		0.001	0.01	0.1	0.001	0.01	0.1	0.001	0.001	0.1
4x50	tanh	100	100	100	**1**	**1**	12	**1**	**2**	14
	swish	100	100	100	69	72	100	**0**	**0**	**2**
	sin	100	100	100	13	5	61	**3**	**4**	12
8x100	tanh	100	100	100	**0**	**0**	**3**	**0**	**1**	10
	swish	100	100	100	91	100	100	**0**	**0**	**0**
	sin	100	100	100	**0**	**1**	13	**0**	**3**	14

Table 4: **Toy Example.** Rate of training success across different system settings (T and y_0) and network architectures (size and activation function). Numbers in the main table represent in percentage (%) cases of successful training. Bold numbers represent a low ($< 5\%$) success rate.

D Additional Content to Toy Example

D.1 Analytical Solution

The analytical solution to (11) is given by

$$y(t) = \begin{cases} \left(1 + \left(\frac{1}{y_0^2} - 1\right) e^{-2t}\right)^{-1/2} & \text{for } 1 \geq y_0 > 0, \\ 0 & \text{for } y_0 = 0, \\ -\left(1 + \left(\frac{1}{y_0^2} - 1\right) e^{-2t}\right)^{-1/2} & \text{for } 0 > y_0 \geq -1. \end{cases}$$

D.2 Rate of Training Success

As defined in Section 4.1, we declare training successful if the L_2 relative error is below 15%. We show the rate of training success for the toy example using different network architectures in Table 4. Similar to observations on the undamped pendulum (see Table 1), we observe a low rate of training success for IC close to the unstable fixed point ($y^* = 0$) and long simulation times.

E Visualizing the Physics Loss Landscape

Visualization of the loss landscape is based on the work of Li et al. (2018) and is a two-dimensional projection (for the 4x50 architecture $n_\theta = 481$). We plot the loss landscape $L(\theta_1, \theta_2)$ with two specific directions θ_1 and θ_2. Here, θ_1 points in the direction of the network weights after the first half of the training ($\theta_1 = \theta^{25k} - \theta^0$), and θ_2 in their direction after full training ($\theta_2 = \theta^{50k} - \theta^0$). A basic Gram-Schmidt process is used to obtain an orthonormalized set of the two directions, and intermediate positions are projected onto them (see markers in Figure 4(c)). The loss landscape for different simulation times T is obtained by evaluating the physics loss function on a total number of 1024 collocation points, sampled from the time domain $t \in [0, T]$. Furthermore, loss values greater than $L(\theta_1, \theta_2) > 0.2$ were truncated to highlight the interesting domain in Figure 4(c)-(f).