Response to Reviewer RhAr

Comment — While the proposed method is interesting, the referee finds it hard to generalize to more challenging cases, and the literature review is not sufficient.

Reply: The basic idea of continuous dependence is a general requirement on the solutions to ODEs and PDEs. And the incorporation of this feature into the PINN framework is also straightforward, as we have demonstrated in the manuscript. In the revision, we have added more challenging examples, like a multiscale model for p53 activation. We have also presented a PDE example on the application of cd-PINN to the diffusion equaitons. These non-trivial examples show that our cd-PINN is capable for more challenging cases.

Towards the literature review, more new related works have been cited to provide the readers a more comprehensive view about the whole problem and recent advances.

Comment — The framework is designed and tested for ODEs, but with PDEs, can you reach comparable performance and speed-accuracy tradeoff?

Reply: The basic idea of continuous dependence is still applicable to PDEs, which is widely known as the well-posedness of PDEs. We found that by including the requirements of continuous dependence into the loss function, the generality of cd-PINN could be dramatically improved when compared to classical PINNs. Here we present an example of cd-PINN applied to the diffusion equations.

Consider the following Cauchy problem for the $1+1$ dimensional diffusion equation:

$$\begin{aligned} &\frac{\partial u}{\partial t} - D\frac{\partial^2 u}{\partial x^2} = 0,\ x \in [-10.0, 10.0],\ t \in [0.1, 1.1], \\ &u(x,0) = \frac{1}{\sqrt{2\pi}\sigma}\exp\Big(-\frac{(x-\mu)^2}{2\sigma^2}\Big) + \frac{1}{\sqrt{2\pi}\sigma_1}\exp\Big(-\frac{(x-\mu_1)^2}{2\sigma_1^2}\Big),\ x \in [-10.0, 10.0]. \end{aligned} \tag{1}$$

In practice, we set $\sigma_1 = 1.0$ and $\mu_1 = 5.0$, and use 20 real data points with $\sigma = 1.0, \mu = 0.0$ as labeled training data, along with 2^{14} unlabeled residual data points. To assess cd-PINN's generalization ability, we selected 10100 configurations of values for $\sigma \in [0.1, 10.0]$ and $\mu \in [-5.0, 5.0]$ to generate corresponding solutions as the test data set, with a total of 8080000 test data points.

Figure 1 shows the predicted results of cd-PINN under three different combinations of μ and σ. It can be seen that cd-PINN has good prediction results under the new μ and σ. Figure 2 shows the mean absolute error of PINN and cd-PINN under all μ and σ. It can be seen that cd-PINN has obvious advantages over PINN. Moreover, the NRMSE of PINN is 2.07×10^{-2}, while the NRMSE of cd-PINN is 5.24×10^{-3}.

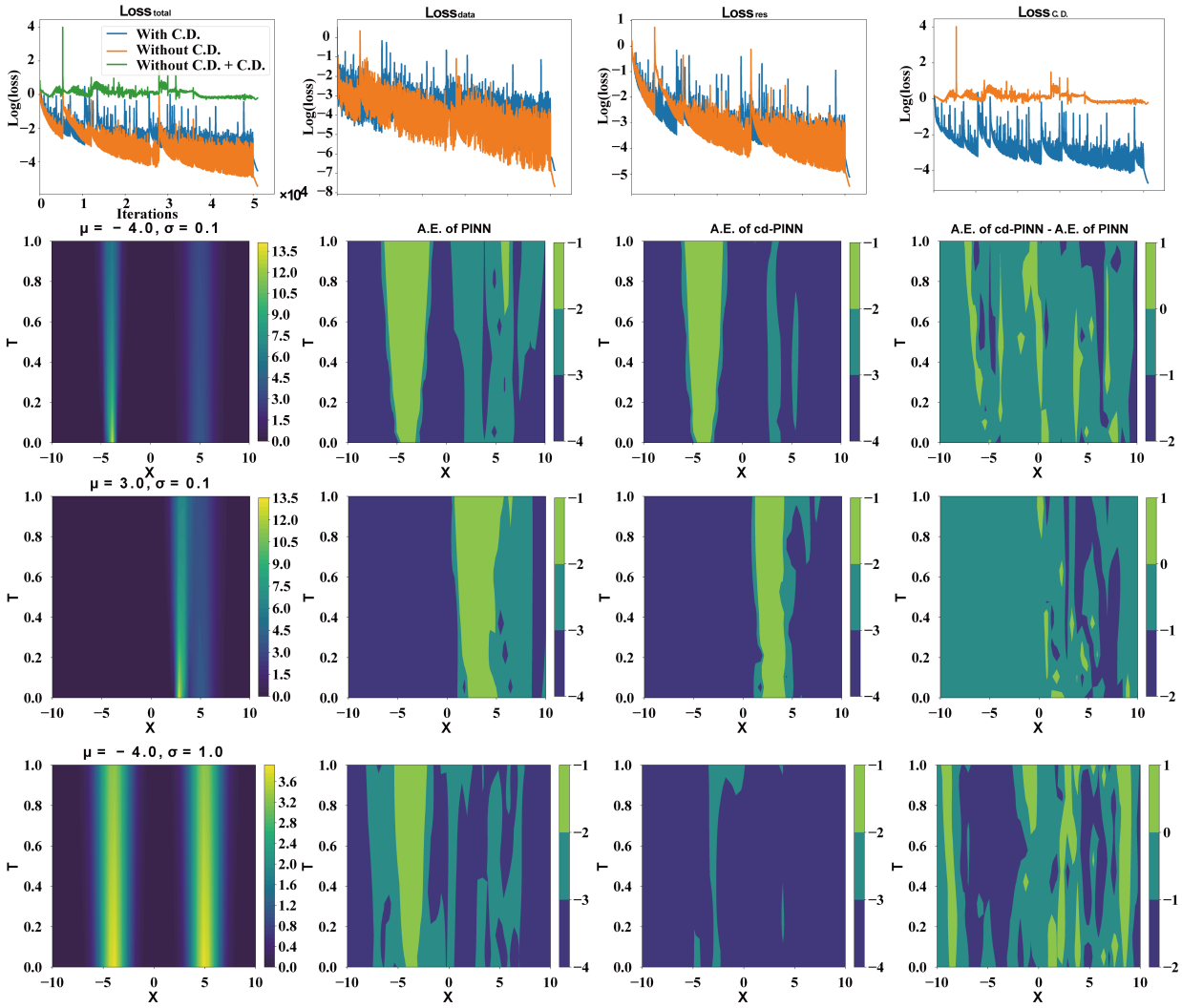

Figure 1: Result of two-dimensional diffusion equation. (a) shows the loss of each part during the model training process. Each column of (b), (c), and (d) shows the true solution, the absolute error of PINN predicted solution, the absolute error of the cd-PINN predicted solution, and the difference in the logarithmic value of the absolute error between the cd-PINN and PINN predicted solution. (a), (b) and (c) are the situations under three different combinations of μ and σ respectively.

We fully understood the significance of applying cd-PINN to PDE cases. However due to the scope of this paper, which focuses on the applications to ODEs, as well as page limits, we decided not to include the PDEs in the current study.

Comment — The parametric ODEs considered are not challenging enough. Say if the parameters induced singularity or multiscale of ODEs, can the proposed method still work?

Reply: In order to test whether our method is applicable to more complicated cases, here we look into a multiscale model for p53 activation, which is a key gene closely related to cancer development [2].

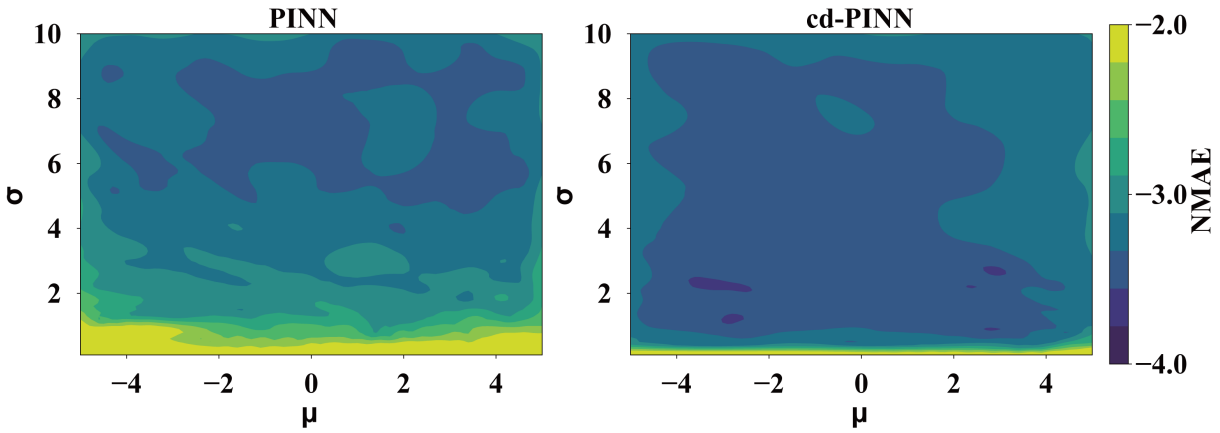

Figure 2: The mean absolute error of PINN and cd-PINN under 10100 configurations of values for $\sigma \in [0.1, 10.0]$ and $\mu \in [-5.0, 5.0]$.

This model is composed of seven coupled ordinary differential functions,

$$
\begin{aligned}
\frac{d[MDM2]}{dt} &= \frac{k_{MD_S}[S]}{K_{MD_S}+[S]} + \frac{k_{M_p}[p53]^{n_1}}{K_{M_p}^{n_1}+[p53]^{n_1}} + D_{MA}[MA] + \frac{k_{Dp4}[MDM2_p]}{K_{Mp}+[MDM2_p]} \\
&\quad - k_{MA}[MDM2][ARF] - \frac{k_{p_M}[Akt][MDM2]}{K_{Akt_M}+[MDM2]} - d_{Mdm2}[MDM2], \\
\frac{d[MDM2_p]}{dt} &= D_{MpA}[M_pA] + \frac{k_{p_M}[Akt][MDM2]}{K_{Akt_M}+[MDM2]} - k_{MpA}[MDM2_p][ARF] \\
&\quad - \frac{k_{Dp4}[MDM2_p]}{K_{Mp}+[MDM2_p]} - d_{Mp}[MDM2_p], \\
\frac{d[MA]}{dt} &= k_{MA}[MDM2][AFR] - D_{MA}[MA] - d_{MA}[MA], \\
\frac{d[M_pA]}{dt} &= k_{MpA}[MDM2_p][ARF] - D_{MpA}[M_pA] - d_{MpA}[M_pA], \\
\frac{d[p53]}{dt} &= k_{p53} - \frac{k_{M53}[MDM2][p53]}{K_{M53}+[p53]} - \frac{k_{Mp53}[MDM2_p][p53]}{K_{Mp53}+[p53]} - d_{p53}[p53], \\
\frac{d[PTEN]}{dt} &= k_{PTEN} + \frac{k_{P_p}[p53]^{n_2}}{K_{P_p}^{n_2}+[p53]^{n_2}} - d_{PTEN}[PTEN], \\
\frac{d[Akt]}{dt} &= \frac{k_{A_S}[S]}{K_{A_S}+[S]} \frac{[Akt]_t-[Akt]}{K_0+[Akt]_t-[Akt]} - \frac{k_{DP3}[Akt]}{K_{Akt}+[Akt]} - \frac{k_{A_p}[PTEN][Akt]}{K_{AP}+[Akt]},
\end{aligned}
\tag{2}
$$

with the initial values (ranging from 0.005 to 5) and model parameters (ranging from 0.05 to 50) directly taken from the cited paper [2].

In this task, we expect that the cd-PINN can correctly learn the solutions to the above model from time $t = 0$ to $t = 5$ with respect to arbitrary inputs of $[S] \in [0.1, 10.0]$ and $[ARF] \in [0.1, 1.5]$. For this purpose, we uniformly select 41×41 groups of $[S]$ and $[ARF]$ during the interval $[0.1, 10.0] \times [0.1, 1.5]$ to generate the test data. Meanwhile, the real data consists of 51 points corresponding to the solution with $[S] = 2.575$, $[ARF] = 0.45$ and 51 points corresponding to the solution with $[S] = 7.525$, $[ARF] = 0.695$. The training data set includes the real data and $N_t = N_0 = 2^{13}$ residual data points.

The final predictions of cd-PINN on this example are summarized in Figure 3, whose MSE is $3.32 \times$

10^{-4}, two order lower than the MSE of the model without C.D. constraints (5.38×10^{-2}). Therefore, it can be concluded that our cd-PINN is capable for handling more challenging situations.

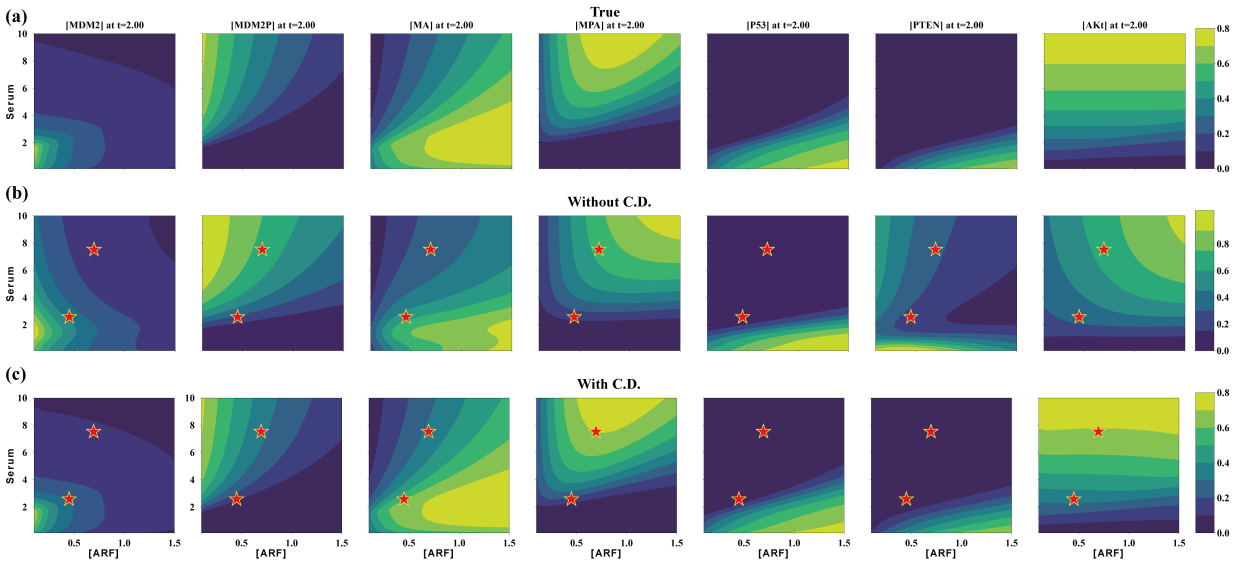

Figure 3: Phase diagram for the solutions of p53 activation model at time $t = 2$. The expression levels for seven genes are calculated by (a) the ODE solver, (b) PINN without C.D. constraints, and (c) cd-PINN separately in comparison.

Changes: See Section 3.4 and Appendix B.4 – A Multiscale Model for P53 Activation for details.

Comment — There has been work in the literature incorporating derivative (smoothness) loss into the loss function and finding that the nns identify smoother functions. I was wondering how the proposed continuity loss compares?

Reply: Thank you for raising this insightful question. Indeed, there has been works in the literature which incorporate derivative (smoothness) loss into the loss function. These works mainly focused on improving the general smoothness and robustness of neural networks. For example, [3] proposed methods to constrain the Lipschitz constants of neural networks, while [1] developed networks with adaptive Lipschitz constants to achieve smoother outputs in reinforcement learning.

In contrast, our continuity loss $\mathcal{L}_{cd}$ is specifically designed based on the mathematical properties of ODE solutions. Instead of enforcing general smoothness, it explicitly incorporates the continuous dependence of ODE solutions on the parameters $\boldsymbol{\mu}$ and initial values $\boldsymbol{u}_0$.

- **Parameter dependence term:** $\left\| \left(\frac{\partial^2 \hat{\boldsymbol{u}}}{\partial \boldsymbol{\mu} \partial t} - \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \frac{\partial \hat{\boldsymbol{u}}}{\partial \boldsymbol{\mu}} - \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\mu}} \right)(t_i, \boldsymbol{u}_{0_i}, \boldsymbol{\mu}_i) \right\|_2^2$.
- **Initial value dependence term:** $\left\| \left(\frac{\partial^2 \hat{\boldsymbol{u}}}{\partial \boldsymbol{u}_0 \partial t} - \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \frac{\partial \hat{\boldsymbol{u}}}{\partial \boldsymbol{u}_0} \right)(t_i, \boldsymbol{u}_{0_i}, \boldsymbol{\mu}_i) \right\|_2^2$.

These terms ensure the mathematical consistency of solutions with respect to different initial values or parameters rather than imposing general smoothness.

Changes: See Page 5 for changes.

"In the literature, there are works which incorporate derivative (or smoothness) loss into the loss function. For example, Virmaux et al. [3] proposed methods to constrain the Lipschitz constants of neural networks, while Song et al. [1] developed networks with adaptive Lipschitz constants to achieve smoother outputs in reinforcement learning. These works mainly focus on improving the smoothness and robustness of neural networks, which make a clear distinction from our method."

Comment — Please use more challenging examples.

Reply: Thank you for the suggestion. In the revision, we have added more challenging examples, like a multiscale model for p53 activation. We have also presented a PDE example on the application of cd-PINN to the diffusion equaitons. These non-trivial examples show that our cd-PINN is capable for more challenging cases.

Changes: See Section 3.4 and Appendix B.4 – A Multiscale Model for P53 Activation for details.

References

[1] Xujie Song, Jingliang Duan, Wenxuan Wang, Shengbo Eben Li, Chen Chen, Bo Cheng, Bo Zhang, Junqing Wei, and Xiaoming Simon Wang. Lipsnet: a smooth and robust neural network with adaptive lipschitz constant for high accuracy optimal control. In International Conference on Machine Learning, pages 32253–32272. PMLR, 2023.

[2] Xinyu Tian, Bo Huang, Xiao-Peng Zhang, Mingyang Lu, Feng Liu, José N Onuchic, and Wei Wang. Modeling the response of a tumor-suppressive network to mitogenic and oncogenic signals. Proceedings of the National Academy of Sciences, 114(21):5337–5342, 2017.

[3] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. Advances in Neural Information Processing Systems, 31, 2018.