A more data-efficient way for discovery of the equation using PINNs

Research · December 2020

CITATIONS
0

READS
70

1 author:

Eunkyue Sohn
Pohang University of Science and Technology
3 PUBLICATIONS **0** CITATIONS

SEE PROFILE

A more data-efficient way for discovery of the equation using PINNs

EunKyue Sohn

December 2020

1 Example: the discovery of Burgers' equation via standard PINNs method

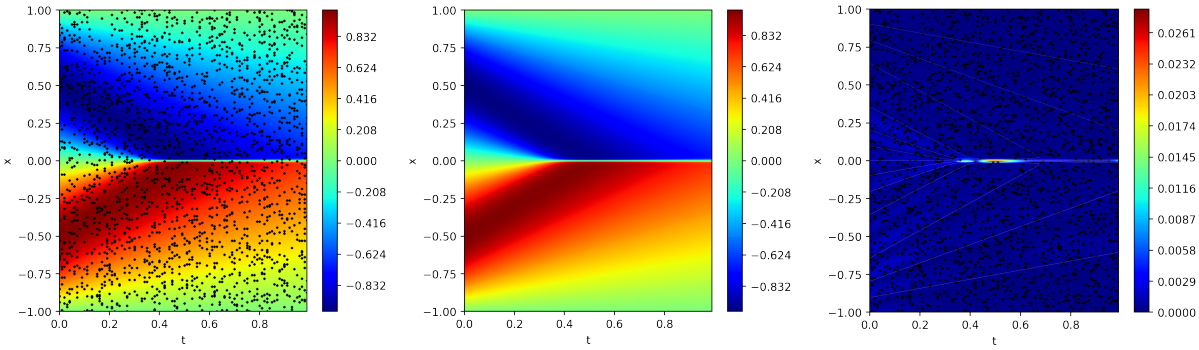
We recall here the process for discovery of 1-dimensional Burgers' equation in original work [1]. Suppose that observations $\{u(t_k, x_k) = y_k\}_{i=k}^{M}$ on the domain $[0, 1] \times [-1, 1]$ are given for the governing equation $u_t + \lambda_1 u u_x - \lambda_2 u_{xx} = 0$.
For the discovery of λ_1 and λ_2, we put λ_1 and λ_2 into the *loss* function (22) as parameters as well as weights and biases, namely,

$$\begin{aligned} loss(W, b, \lambda_1, \lambda_2) = &\frac{1}{M}\sum_{k=1}^{M}(\boldsymbol{n}(t_k, x_k) - y_k(t_k, x_k))^2 \\ &+ \frac{1}{M}\sum_{k=1}^{M}(\boldsymbol{n}_t(t_k, x_k) + \lambda_1 \boldsymbol{n}(t_k, x_k)\boldsymbol{n}_x(t_k, x_k) - \lambda_2 \boldsymbol{n}_{xx}(t_k, x_k))^2 \end{aligned} \tag{1}$$

We randomly pick $M = 2000$ points from the reference solution of example 3.2.1.1 as shown in figure 11. Note that in this case, correct values of λ_1 and λ_2 are $(1,\ 0.01/\pi) \simeq (1.0,\ 0.00318309886)$.

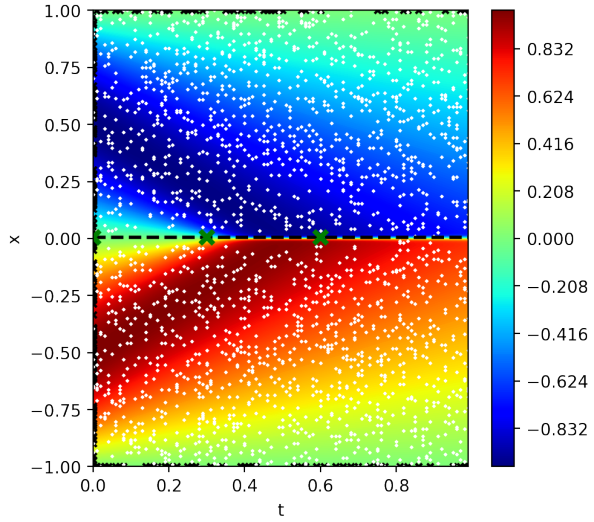
After learning process with L-BFGS-B optimizer, PINNs reveal almost exact values of λ_1 and λ_2, $\lambda_1 = 1.0002587, \lambda_2 = 0.00314385$. Note that percentage errors are 0.02587%, and percentage errors are 1.23319% for λ_1, λ_2, respectively.

While PINNs reveal the equation, PINNs also provide the prediction as well (see figure 1 - (c)). The results are depicted in figure 11.

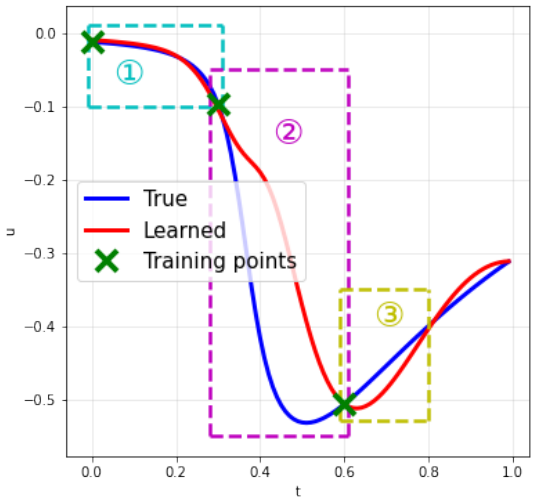


(a) Learned dynamics (b) True dynamics (c) Error

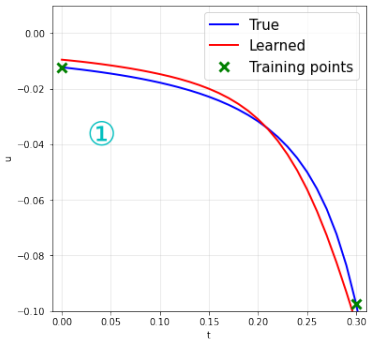
Figure 1: **Data-driven discovery of Burgers'equation**: *PINNs reveal the correct values of* λ_1, λ_2. *Percentage errors are* 0.00286%, 0.16974% *for* λ_1, λ_2, *respectively. The black points in subfigure (a) and (c) represent our observations.*



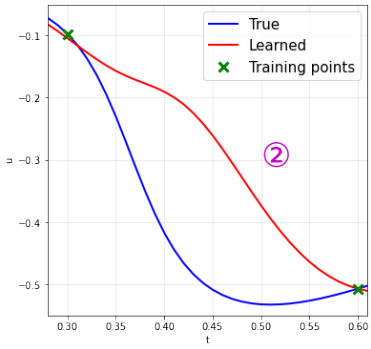
(a) True dynamics



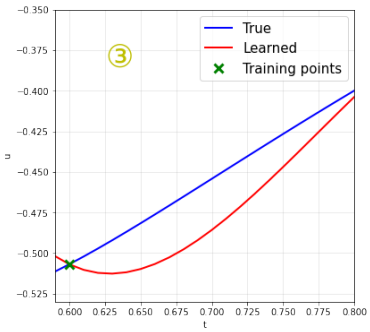
(b) Spatio-snapshot, $x = 0.00392157$



(c) Sub-picture of spatio-snapshot: ①



(d) Sub-picture of spatio-snapshot: ②



(e) Sub-picture of spatio-snapshot: ③

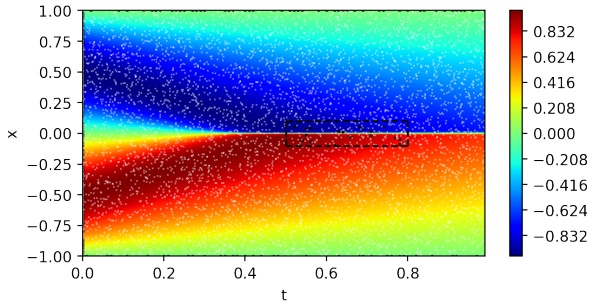
Figure 2: **A more data-efficient way for discovery of the equation**: ***(a)****: Green points represent data on $x = 0.00392157$ and black points are initial and boundary points. White points are 2000 interior observations.* ***(b)****: The spatio-snapshot at $x = 0.00392157$ which is depicted by horizontal black dashed line in subfigure (a).* ***(c), (d), (e)****: The magnified sub-picture of spatio-snapshot depicted by rectangles in subfigure (b).*

2 A more data-efficient way for discovery of the equation

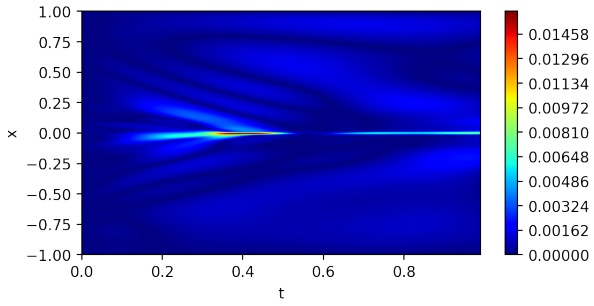
Recall the previous example. The PINNs in original paper already provides a data-efficient way to discover the equation, but getting $M = 2000$ points in the interior region may not be an easy task. In some cases, it may be preferable to take observations in initial and boundary condition, and in addition, it is a well-known fact that in most cases initial and boundary conditions play a very critical role in solving differential equations. Therefore, we take a look at a data-efficient way to make important use of initial and boundary conditions with some modifications.

Recall that the *loss* function in the previous example:

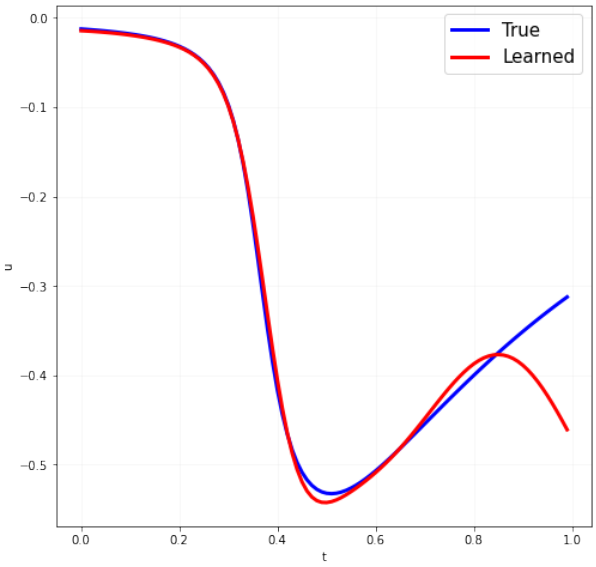
$$loss(W, b, \lambda_1, \lambda_2) = g(W, b) + h(W, b, \lambda_1, \lambda_2) \tag{2}$$



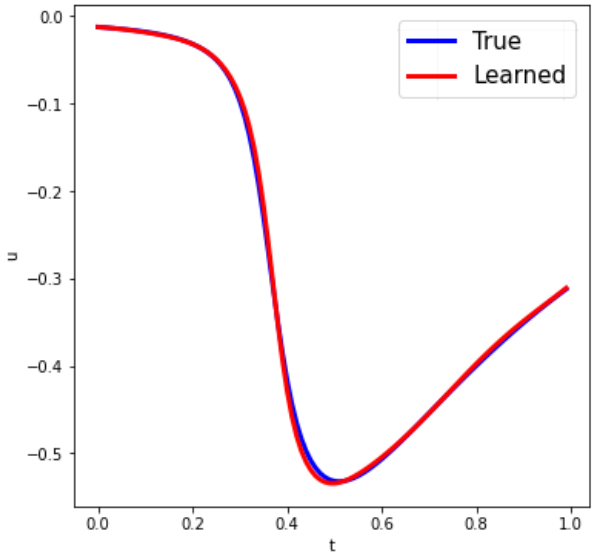
(a) Observation from true data



(b) error



(c) spatio-snapshot with 2000 interior observations.



(d) spatio-snapshot with the proposed work.

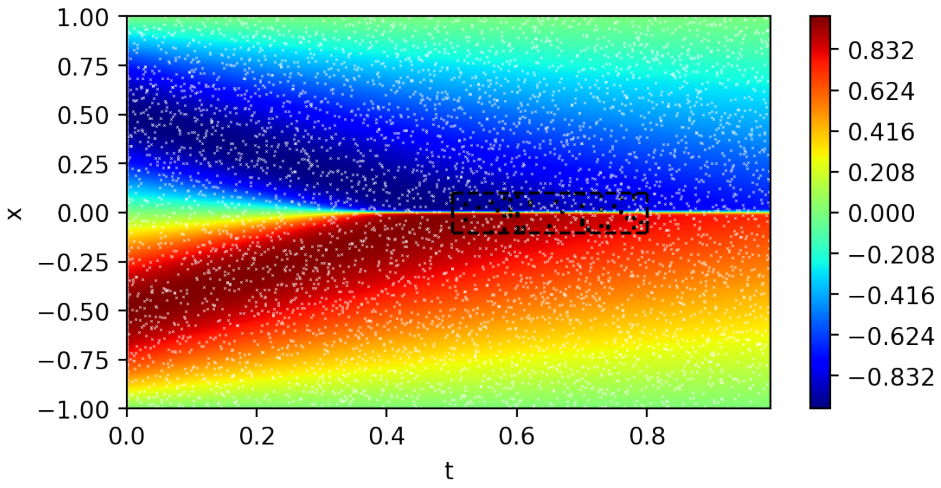
Figure 3: **A more data-efficient way for discovery of the equation**: *In subfigure (a), the rectangle marked with a black dashed line means the area in which 20 interior observation was sampled. The black points and white points represent observation and residual collocation points, respectively. Subfigure (c) is a spatial snapshot after learning without initial and boundary points and additional residual collocation points. Instead, as in the previous example in 3.2.2.1, 2000 training data were used. The subfigure (d) is a spatial snapshot learned by the proposed methodology. The proposed methodology provides accurate values of* λ_1 *and* λ_2*,* $(\lambda_1, \lambda_2) = (0.99800324, 0.00318106)$ *with 200 initial and boundary points and only 20 interior observations. The percentage errors are 0.1996% and 0.0641% for* λ_1 *and* λ_2*, respectively.*

, where

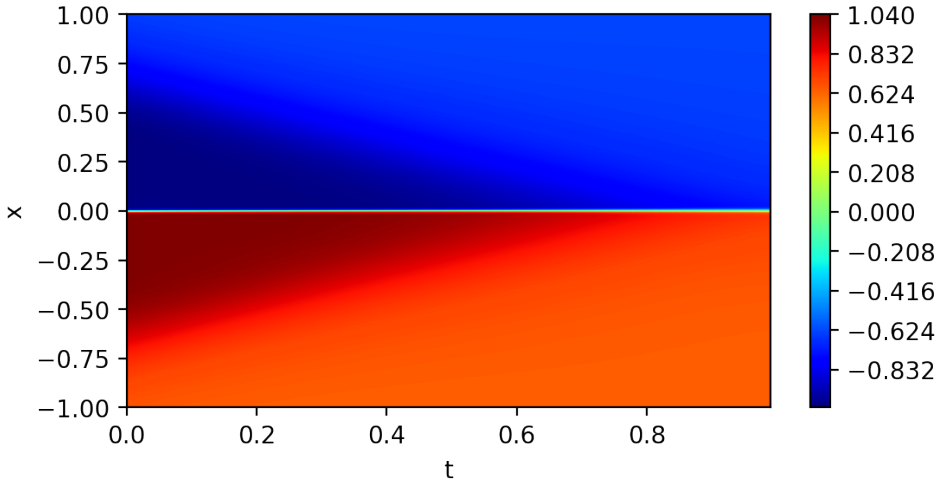
$$
\begin{aligned}
g(W,b) &= \frac{1}{M}\sum_{k=1}^{M}(\boldsymbol{n}(t_k,x_k) - y_k(t_k,x_k))^2 \\
h(W,b,\lambda_1,\lambda_2) &= \frac{1}{M}\sum_{k=1}^{M}(\boldsymbol{n}_t(t_k,x_k) + \lambda_1\boldsymbol{n}(t_k,x_k)\boldsymbol{n}_x(t_k,x_k) - \lambda_2\boldsymbol{n}_{xx}(t_k,x_k))^2
\end{aligned}
\tag{3}
$$

The key observation here is that the training data for g and h are the same as $\{(t_k, x_k)\}_{k=1}^{M}$. If we keep these frameworks completely intact, adding initial points or boundary points would not have a big effect.

Forcing the structure of the neural network where the training data $\{(t_k, x_k)\}_{k=1}^{M}$ exist (i.e., $g \to 0+$) does not always guarantee that we can obtain the correct values of the derivatives. To illustrate this, we add 200 randomly chosen initial and boundary points to the problem in the previous example (see



(a) Observation from true data



(b) a solution for ill-posed problem

Figure 4: **A more data-efficient way for discovery of the equation**: *In sub-figure (a), the rectangle indicated by the black dashed line represents the space where the data was sampled. Black points and white points are the observations and the residual collocation points, respectively. subfigure (b) shows the solution of ill-posed problem.*

figure 2). As one can see from the figure, if the gap between data is relatively large, one can rarely get the precise derivatives. In the gap between the data, the neural network could be an unsuitable smooth surface. Thus in order to get accurate results, we usually close the gap between the data by increasing the number of points, and get a kind of 'mean value' over the whole domain area.

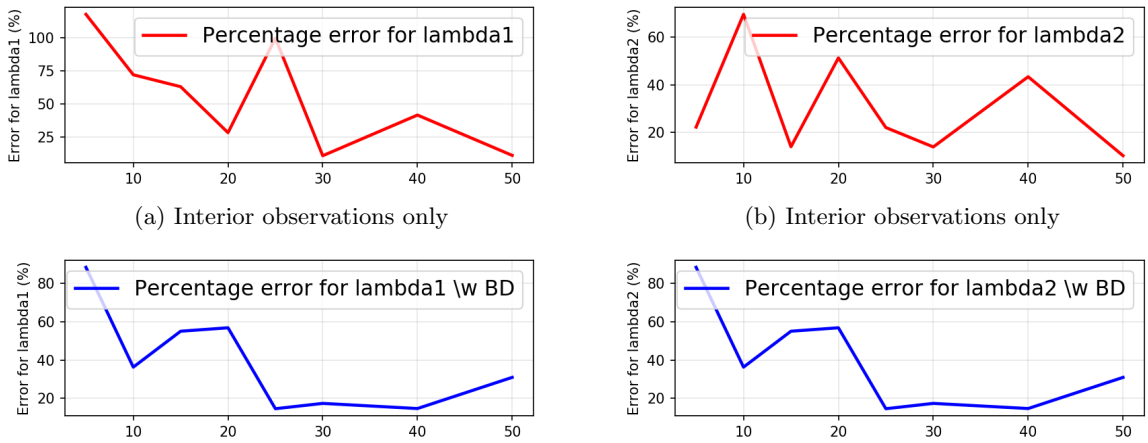
This method already gives reasonable results, but to further improve the accuracy, let us introduce additional residual collocation points into the *loss* function of the equation (2):

$$loss(W, b, \lambda_1, \lambda_2) = g(W, b) + h(W, b, \lambda_1, \lambda_2) + w(W, b, \lambda_1, \lambda_2) \tag{4}$$

,where

$$w(W, b, \lambda_1, \lambda_2) = \frac{1}{N_f} \sum_{k=1}^{N_f} (\boldsymbol{n}_t(t_f^k, x_f^k) + \lambda_1 \boldsymbol{n}(t_f^k, x_f^k) \boldsymbol{n}_x(t_f^k, x_f^k) - \lambda_2 \boldsymbol{n}_{xx}(t_f^k, x_f^k))^2 \tag{5}$$

This means that the structure learned from the given observations is propagated to the whole area. In other words, the structure learned in some sub-domain is forced to the neural network in the entire



(a) Interior observations only

(b) Interior observations only

(c) With initial and boundary points

(d) With initial and boundary points

Figure 5: **A more data - efficient way for discovery of the equation**: *Subfigure (a) and (b) show the relationship between the number of observations and the percentage error when there are only observations in the interior region. Subfigure (c) and (d) show the relationship between the number of observations and the percentage error when 200 initial and boundary points are added to the observations in the interior region. Even if the number of observations is increased, the percentage error does not fit well within 10%.*

domain. This modification will give 'connectivity' to the data-to-data gap. Neural networks no longer have freedom in the empty space between data and data [1].

To demonstrate the effectiveness and the ability to impart 'connectivity' of the proposed methodology, we randomly take a total of 200 initial and boundary points, and take only 20 points in the area far from there, namely $t \in [0.5, 0.8]$, $x \in [-0.1, 0.1]$ (see figure 3-(a)). And in order to enforce the overall structure, additional 5000 residual collocation points are selected by the LHS method.

In summary, we make the *loss* function as follows:

$$\begin{aligned} loss(W, b, \lambda_1, \lambda_2) = &\frac{1}{200}\sum_{k=1}^{200}(\boldsymbol{n}(t_u^k, x_u^k) - y_u^k)^2 + \frac{1}{20}\sum_{k=1}^{20}(\boldsymbol{n}(t_{int}^k, x_{int}^k) - y_{int}^k)^2 \\ &+ \frac{1}{220}\sum_{k=1}^{220}(\boldsymbol{n}_t(t_w^k, x_w^k) + \lambda_1 \boldsymbol{n}(t_w^k, x_w^k)\boldsymbol{n}_x(t_w^k, x_w^k) + \lambda_2 \boldsymbol{n}_{xx}(t_w^k, x_w^k))^2 \\ &+ \frac{1}{5000}\sum_{k=1}^{5000}(\boldsymbol{n}_t(t_f^k, x_f^k) + \lambda_1 \boldsymbol{n}(t_f^k, x_f^k)\boldsymbol{n}_x(t_f^k, x_f^k) + \lambda_2 \boldsymbol{n}_{xx}(t_f^k, x_f^k))^2 \end{aligned} \tag{6}$$

, where

$$\begin{aligned} &\{(t_u^k, x_u^k), y_u^k\}_{k=1}^{200} = \{(t_i^k, x_i^k), y_i^k\}_{k=1}^{N_i} \cup \{(t_b^k, x_b^k), y_b^k\}_{k=1}^{N_b} \ : \textit{Initial and boundary points} \\ &\{(t_{int}^k, x_{int}^k), y_{int}^k\}_{k=1}^{20} \ : \textit{Interior observations} \\ &\{(t_w^k, x_w^k), y_w^k\}_{k=1}^{220} = \{(t_u^k, x_u^k), y_u^k\}_{k=1}^{200} \cup \{(t_{int}^k, x_{int}^k), y_{int}^k\}_{k=1}^{20} \\ &\{(t_f^k, x_f^k), y_f^k\}_{k=1}^{5000} \ : \textit{Residual collocation points} \end{aligned} \tag{7}$$

Surprisingly, the proposed work gives the precise values of λ_1 and λ_2, $(\lambda_1, \lambda_2) = (0.99800, 0.00318)$. The percentage errors are 0.1996% and 0.0641% for λ_1 and λ_2, respectively. With the methodology presented,

[1] This does not mean the freedom in rigorous sense. It is usually determined by the structure of the neural network or the behavior of the optimizer.

one can solve the inverse problem and the forward problem simultaneously with high accuracy from initial and boundary conditions and a relatively small number of interior observations.
Note that the initial and boundary conditions in this process are critical. If the proposed methodology proceeds without initial and boundary conditions, it becomes an ill-posed problem, and appropriate learning fails with a high probability (see figure 4).

And in the process of the proposed methodology, additional collocation points also play a very important role. For example, increasing the number of points only in the interior area of the domain without additional collocation points, or increasing the number of points in the interior area with given 200 initial and boundary points does not give accurate results. Observation by increasing the number of data points in the interior region shows that when there are additional residual collocation points, the discovery of the equation is very stable with only 20 points. On the other hand, if the inverse problem proceeds without additional residual collocation points, the percentage error does not go well within 10% even if the number of points is increases (See figure 5).

References

[1] Maaziar Raissi, Paris Perdikari, and George Em Karniadakis, Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 2019.