# Measuring the Risk of Solving Fluid Dynamics by Neural Nets

Dibyakanti Kumar, Department of Computer Science, The University of Manchester

This talk is based on the work "Investigating the ability of PINNs to solve Burgers' PDE near finite-time blowup" published at the IOP-MLST journal with Anirbit Mukherjee

DOI : https://doi.org/10.1088/2632-2153/ad51cd

Let $\boldsymbol{u}(\boldsymbol{x}, t)$ be the actual solution

**PDE**

$$\boldsymbol{u}_t + \mathcal{N}_{\boldsymbol{x}}[\boldsymbol{u}] = 0, \quad \boldsymbol{x} \in D, t \in [0, T]$$

**Initial Condition**

$$\boldsymbol{u}(\boldsymbol{x}, 0) = h(\boldsymbol{x}), \quad \boldsymbol{x} \in D$$

**Boundary Conditions**

$$\boldsymbol{u}(\boldsymbol{x}, t) = g(\boldsymbol{x}, t), \quad \boldsymbol{t} \in [0, T], \boldsymbol{x} \in \partial D$$

Let the neural solution be $\boldsymbol{u}_\theta(\boldsymbol{x}, t)$

**Functional Residual**

$$\mathcal{R}_{pde}(x, t) := \partial_t \boldsymbol{u}_\theta + \mathcal{N}_{\boldsymbol{x}}[\boldsymbol{u}_\theta(\boldsymbol{x}, t)]$$

**Conditional Residual (I.C.)**

$$\mathcal{R}_t(x) := \boldsymbol{u}_\theta(\boldsymbol{x}, 0) - h(\boldsymbol{x})$$

**Conditional Residual (B.C.)**

$$\mathcal{R}_b(x, t) := \boldsymbol{u}_\theta(\boldsymbol{x}, t) - g(\boldsymbol{x}, t)$$

# Physics Informed Neural Nets (PINNs)

Let the neural solution be $\boldsymbol{u}_\theta(\boldsymbol{x}, t)$

**Conditional Residual (I.C.)**

$$\mathcal{R}_t(x) := \boldsymbol{u}_\theta(\boldsymbol{x}, 0) - h(\boldsymbol{x})$$

**Functional Residual**

$$\mathcal{R}_{pde}(x, t) := \partial_t \boldsymbol{u}_\theta + \mathcal{N}_x[\boldsymbol{u}_\theta(\boldsymbol{x}, t)]$$

**Conditional Residual (B.C.)**

$$\mathcal{R}_b(x, t) := \boldsymbol{u}_\theta(\boldsymbol{x}, t) - g(\boldsymbol{x}, t)$$

**PINN Loss Function**

$$\mathcal{L}(\theta) := \frac{1}{N_{pde}} \sum_{i=1}^{N_{pde}} \mathcal{R}_{pde}(x_r^i, t_r^i)^2 + \lambda_1 \frac{1}{N_t} \sum_{i=1}^{N_t} \mathcal{R}_t(x_t^i)^2 + \lambda_2 \frac{1}{N_b} \sum_{i=1}^{N_b} \mathcal{R}_b(x_b^i, t_b^i)^2$$

# Why Physics Informed Neural Nets (PINNs) ?

PINNs have been achieving ever newer feats of solving complicated PDEs numerically while offering an attractive trade-off between **accuracy** and **speed** of inference.

In this work, we investigate the stability of PINNs near finite-time blow-ups from a rigorous theoretical viewpoint.

# Why Physics Informed Neural Nets (PINNs) ?

$$\mathcal{E}_G := \|\boldsymbol{u} - \boldsymbol{u}_\theta\|$$

$$\tilde{\mathcal{E}}_T := \left(\sum_{n=1}^{N} w_n |\mathcal{R}_\theta|^p\right)^{1/p}$$

## Risk Upperbound (Theorem 2.6)[1]

$$\mathcal{E}_G \leq C_{pde}\tilde{\mathcal{E}}_T + C_{pde} C_{quad}^{\frac{1}{p}} N^{-\frac{\alpha}{p}}$$

It was proven in the above mentioned theorem – for the first time – that one can minimize certain empirical errors to find provably good approximations to any PDE.

The strength of this result lies in its reliance on only **mild conditions** on the PDE.

[1]Mishra and Molinaro, "Estimates on the generalization error of physics-informed neural networks for approximating PDEs".

# Why Finite-Time Blow-Ups ?

- Existing studies with PINNs (including studies of its failure modes) focus on cases where the true solution is “nice”.

# Why Finite-Time Blow-Ups ?

- Existing studies with PINNs (including studies of its failure modes) focus on cases where the true solution is "nice".
- A finite-time blow-up – a phenomenon where the solution $\boldsymbol{u}$ becomes infinite at some points as $t$ approaches a certain time $T < \infty$, while the solution is well-defined for all $0 < t < T$, $\sup_{x \in D} |\boldsymbol{u}(\boldsymbol{x}, t)| \to \infty \quad as \quad t \to T^{-}$
- $\frac{du}{dt} = u^2, \ u(0) = u_0, \ u_0 > 0$ is an ODE whose solution blows-up at $t = \frac{1}{u_0}$

# Why Finite-Time Blow-Ups ?

- Existing studies with PINNs (including studies of its failure modes) focus on cases where the true solution is “nice”.
- A finite-time blow-up – a phenomenon where the solution $\boldsymbol{u}$ becomes infinite at some points as $t$ approaches a certain time $T < \infty$, while the solution is well-defined for all $0 < t < T$, $\sup_{x \in D} |\boldsymbol{u}(\boldsymbol{x}, t)| \to \infty \quad as \quad t \to T^{-}$
- $\frac{du}{dt} = u^2,\ u(0) = u_0,\ u_0 > 0$ is an ODE whose solution blows-up at $t = \frac{1}{u_0}$
- There are multiple real-world phenomena whose PDE models have finite-time blow-ups and these singularities are known to correspond to practically relevant processes – such as in chemotaxis models, thermal-runoff models and combustion models.

# Why Finite-Time Blow-Ups ?

In a recent experimental studies with PINNs[2], experimental evidence was shown for PINNs potentially **discovering PDE solutions with blow-up** even when their **explicit descriptions are not known**.

[2] Wang et al., *Asymptotic self-similar blow up profile for 3-D Euler via physics-informed neural networks.*

# Why Finite-Time Blow-Ups ?

In a recent experimental studies with PINNs[2], experimental evidence was shown for PINNs potentially **discovering PDE solutions with blow-up** even when their **explicit descriptions are not known**.

Here we look to understand this interface from a rigorous viewpoint and show how well the theoretical risk bounds correlate to their experimentally observed values - in certain blow-up situations.

[2] Wang et al., *Asymptotic self-similar blow up profile for 3-D Euler via physics-informed neural networks.*

# Why is Burgers' PDE Interesting?

In order to assess the efficacy of PINNs in the vicinity of a blow-up phenomenon, we have chosen a very specific case - that of the Burgers' PDE where analytically exact solutions with finite-time blow-up are known - as is needed for a controlled study!

## Burgers' Equation (inviscid)

$$\partial_t \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = 0$$

# Main Results

We analyse the following PDE in the domain $D \subset \mathbb{R}^d$ and $t \in [-\frac{1}{\sqrt{2}} + \delta, \delta]$ for the following initial condition

**PDE**

$$\partial_t \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = 0$$

**Initial Condition**

$$\boldsymbol{u}(t = t_0) = \boldsymbol{u}_{t_0}, \; t_0 = -\frac{1}{\sqrt{2}} + \delta$$

Let the neural net we aim to train be $\boldsymbol{u}_\theta(\boldsymbol{x}, t) \in \mathbb{R}^d$.

Then the residual terms for this predictor are,

**Conditional Residual**

$$\mathcal{R}_{\mathrm{t}} := \boldsymbol{u}_\theta(t = t_0) - \boldsymbol{u}(t = t_0)$$

The residual term for the functional loss is

**Functional Residual**

$$\mathcal{R}_{\mathrm{pde}} := \partial_t \boldsymbol{u}_\theta + (\boldsymbol{u}_\theta \cdot \nabla)\boldsymbol{u}_\theta$$

# (d+1)-Burgers' PDE : $L^2$-Risk

The $L^2$ population risk of the predictor that we are interested in is,

$$\int_{\Omega} \|\boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{u}_\theta(\boldsymbol{x}, t)\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t$$

where $\boldsymbol{u}$ is the exact solution and $\boldsymbol{u}_\theta$ is the neural surrogate.

This definition of $L^2$-risk measures the distance of the neural surrogate from the true PDE solution.

**Theorem 1**

Let $\boldsymbol{u} \in C^1(D \times [t_0, T])$ be the unique solution of the $(d+1)-$dimensional Burgers' PDE. Then for any $C^1$ surrogate solution, say $\boldsymbol{u}_\theta$, the $L^2$-risk with respect to the true solution is bounded as,

$$\log\left(\int_\Omega \|\boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{u}_\theta(\boldsymbol{x}, t)\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t\right) \leq \log\left(\frac{C_1 C_2}{4}\right) + \frac{C_1}{\sqrt{2}} \tag{1}$$

where,

$$C_1 = d^2\|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} + 1 + d^2\|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)}$$

$$C_2 = \int_D \|\mathcal{R}_t\|_2^2 \,\mathrm{d}\boldsymbol{x} + \int_\Omega \left\|\mathcal{R}_{pde}\right\|_2^2 \mathrm{d}\boldsymbol{x}\,\mathrm{d}t + d^2\|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}_\theta\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t$$
$$+ d^2\|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t$$

# (d+1)-Burgers' PDE: An Upper Bound for the $L^2-$Risk

$$C_1 = d^2 \|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} + 1 + d^2 \|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)}$$

$$C_2 = \int_D \|\mathcal{R}_t\|_2^2 \,\mathrm{d}\boldsymbol{x} + \int_\Omega \|\mathcal{R}_{pde}\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t + d^2 \|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}_\theta\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t$$
$$+ d^2 \|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t$$

Note the following key points about this bound,

- This can be estimated just by knowing an upper bound on $\|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)}$ and $\|\boldsymbol{u}\|_2$.

# (d+1)-Burgers' PDE: An Upper Bound for the $L^2-$Risk

$$
\begin{aligned}
C_1 &= d^2\|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} + 1 + d^2\|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)} \\
C_2 &= \int_D \|\mathcal{R}_t\|_2^2 \,\mathrm{d}\boldsymbol{x} + \int_\Omega \|\mathcal{R}_{pde}\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t + d^2\|\nabla \boldsymbol{u}_\theta\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}_\theta\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t \\
&\quad + d^2\|\nabla \boldsymbol{u}\|_{L^\infty(\Omega)} \int_\Omega \|\boldsymbol{u}\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t
\end{aligned}
$$

Note the following key points about this bound,

- This has explicit dependence on the norm of the surrogate (through $C_2$) and its spatial gradient (through both $C_1$ and $C_2$). Hence providing a theoretical foundation for the role of **functional regularization** in PINN training[3].

[3] Wang et al., *Asymptotic self-similar blow up profile for 3-D Euler via physics-informed neural networks.*

# (1+1)-Burgers' PDE Near Finite-Time Blow Up

In one dimension, for Burgers's PDE we can get a more interesting bound! Towards that we consider working in the domain $x \in [-1, 1]$ and $t \in [-1 + \delta, \delta)$, parameterized by $\delta$.

**PDE**

$$u_t + uu_x = 0$$

**Initial Conditions**

$$u(x, -1 + \delta) = \frac{x}{-2 + \delta}$$

**Boundary Conditions**

$$u(-1, t) = \frac{1}{1 - t}$$

$$u(1, t) = \frac{1}{t - 1}$$

## (1+1)-Burgers' PDE Near Finite-Time Blow Up

In one dimension, for Burgers's PDE we can get a more interesting bound! Towards that we consider working in the domain $x \in [-1, 1]$ and $t \in [-1 + \delta, \delta)$, parameterized by $\delta$.

**PDE**

$$u_t + uu_x = 0$$

**Initial Conditions**

$$u(x, -1 + \delta) = \frac{x}{-2 + \delta}$$

**Boundary Conditions**

$$u(-1, t) = \frac{1}{1 - t}$$

$$u(1, t) = \frac{1}{t - 1}$$

These initial and boundary conditions correspond to an exact solution to the Burgers' PDE i.e. $u(x, t) = \frac{x}{t-1}$

# (1+1)-Burgers' PDE Near Finite-Time Blow Up : Residuals

As earlier, for the neural net $u_\theta(x,t)$ attempting to solve this PDE, we define the following residuals,

**Conditional Residuals**

$$R_{tb,\theta}(x) = u_\theta(x, -1+\delta) - \frac{x}{-2+\delta}$$

$$R_{sb,-1,\theta}(t) = u_\theta(-1,t) - \frac{1}{1-t}, \quad R_{sb,1,\theta}(t) = u_\theta(1,t) - \frac{1}{t-1}$$

**Functional Residual**

$$R_{int,\theta}(x,t) = \partial_t u_\theta(x,t) + u_\theta(x,t) \cdot \partial_x u_\theta(x,t)$$

**Theorem 2**

Let $u \in C^1((-1+\delta, \delta) \times (-1, 1))$ be the unique solution of the (1+1)-D Burgers' equation for any $k \geq 1$ and $u^* = u_{\theta^*}$ be the neural surrogate. Then the population risk of it is bounded as,

$$
\begin{aligned}
&\left(\int_{-1+\delta}^{\delta}\int_{-1}^{1} |u(x,t) - u_\theta(x,t)|^2 \, dxdt\right)^{\frac{1}{2}} \\
&\leq \left[1 + Ce^C\right]\left[\int_{-1}^{1} \mathcal{R}_{tb,\theta^*}(x)dx + \int_{-1+\delta}^{\delta}\int_{-1}^{1} \mathcal{R}^2_{int,\theta^*}(x,t)dxdt\right. \\
&+ 2C_{2b}\left(\int_{-1+\delta}^{\delta} \mathcal{R}^2_{sb,-1,\theta^*}(t)dt + \int_{-1+\delta}^{\delta} \mathcal{R}^2_{sb,1,\theta^*}(t)dt\right) \\
&\left.+ 2C_{1b}\left(\left(\int_{-1+\delta}^{\delta} \mathcal{R}^2_{sb,-1,\theta^*}(t)dt\right)^{\frac{1}{2}} + \left(\int_{-1+\delta}^{\delta} \mathcal{R}^2_{sb,1,\theta^*}(t)dt\right)^{\frac{1}{2}}\right)\right]
\end{aligned}
\tag{2}
$$

# (1+1)-Burgers' PDE Near Finite-Time Blow Up : An Upper Bound for $L^2-$Risk

where $C = 1 + 2C_{u_x}$, with

$$C_{u_x} = \|u_x\|_{L^\infty} = \left\|\frac{1}{t-1}\right\|_{L^\infty([-1+\delta,\delta])} = \frac{1}{1-\delta}$$

$$C_{1b} = \|u(1,t)\|_{L^\infty}^2 = \left\|\frac{1}{1-t}\right\|_{L^\infty([-1+\delta,\delta])}^2 = \frac{1}{(1-\delta)^2}$$

$$C_{2b} = \|u_{\theta^*}(1,t)\|_{L^\infty([-1+\delta,\delta])} + \frac{3}{2}\left\|\frac{1}{t-1}\right\|_{L^\infty([-1+\delta,\delta])} = \|u_{\theta^*}(1,t)\|_{L^\infty([-1+\delta,\delta])} + \frac{3}{2}\left(\frac{1}{1-\delta}\right)$$

# (1+1)-Burgers' PDE Near Finite-Time Blow Up : Stability

Even though we set up the initial and boundary conditions to reflect a finite-time singularity, our bound for the $L_2$ population risk indicates the stability of the PINN risk, defined as,

$$\mathbb{E}[|\mathcal{R}_{int,\theta}(x,t)|^2] + \mathbb{E}[|\mathcal{R}_{tb,\theta}|^2] + \mathbb{E}[|\mathcal{R}_{sb,-1,\theta}|^2] + \mathbb{E}[|\mathcal{R}_{sb,1,\theta}|^2]$$

The "stability"[4] here means that if this PINN risk is found to be $\mathcal{O}(\epsilon)$ then that would directly imply that the $L_2$ population risk is also $\mathcal{O}(\epsilon)$.

[4] Wang et al., "Is $L^2$ Physics Informed Loss Always Suitable for Training Physics Informed Neural Network?"

# Experiment and Analysis

# (2+1)-Burgers' PDE with Finite-Time Blow Up

Consider solving the $(2+1)-$Burgers' PDE on $t \in [-\frac{1}{\sqrt{2}} + \delta, \delta]$ where $\delta \in [0, \frac{1}{\sqrt{2}})$

## Initial Conditions

$$u_{1,t_0} = \frac{(1+\sqrt{2}-2\delta)x_1 + x_2}{2\delta(\sqrt{2}-\delta)} \ ; \ u_{2,t_0} = \frac{x_1 - (1-\sqrt{2}+2\delta)x_2}{2\delta(\sqrt{2}-\delta)}$$

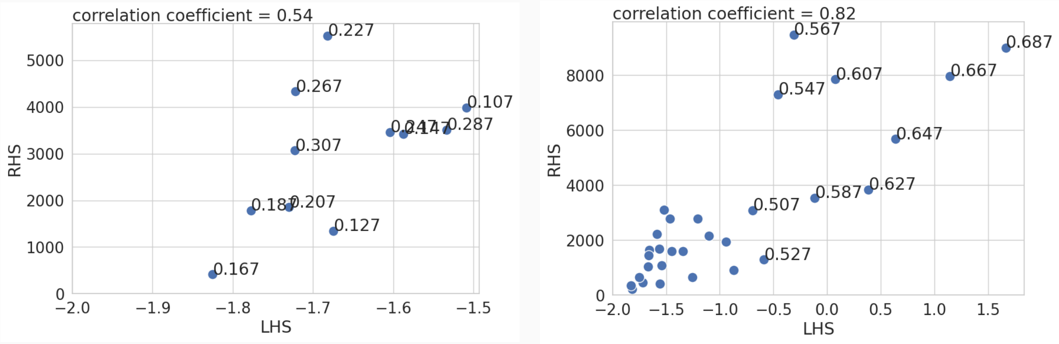
## Boundary Conditions

$$u_1(x_1 = 0) = \frac{x_2}{1 - 2 \cdot t^2} \ ; \quad u_1(x_1 = 1) = \frac{1 + x_2 - 2 \cdot t}{1 - 2 \cdot t^2}$$

$$u_2(x_2 = 0) = \frac{x_1}{1 - 2 \cdot t^2} \ ; \quad u_2(x_2 = 1) = \frac{x_1 - 1 - 2 \cdot t}{1 - 2 \cdot t^2}$$

These I.C. and B.C. correspond to this exact solution $u_1 = \frac{x_1 + x_2 - 2x_1 t}{1 - 2t^2}$, $u_2 = \frac{x_1 - x_2 - 2x_2 t}{1 - 2t^2}$

# (2+1)-Burgers' PDE with Finite-Time Blow Up : LHS vs RHS Plots for Theorem 1




**(a)** width=30 **(b)** width=100

**Figure 1:** This figure shows the RHS vs LHS plot of Equation (1) from Theorem 1 for different values of $\delta$ for PINNs of 2 different widths. (Recall that here the blow up is at $\delta \sim 0.7$)

# Future Directions

# Future Directions

- Can modifications to the PINN formalism be made to make them more sensitive to the local structure of the solutions and thus help detect the existence of finite-time blow-ups in PDEs?

[5] Wang et al., "Is $L^2$ Physics Informed Loss Always Suitable for Training Physics Informed Neural Network?"
[6] Gopalani and Mukherjee, "Global Convergence of SGD On Two Layer Neural Nets".

## Future Directions

- Can modifications to the PINN formalism be made to make them more sensitive to the local structure of the solutions and thus help detect the existence of finite-time blow-ups in PDEs?
- Are there any PINN losses for the $(d+1)$-dimensional Burgers – or for Navier-Stokes in general – that is "stable"[5], as was shown to be true in our $(1+1)$-dimensional Burgers' PDE setup?

[5] Wang et al., "Is $L^2$ Physics Informed Loss Always Suitable for Training Physics Informed Neural Network?"
[6] Gopalani and Mukherjee, "Global Convergence of SGD On Two Layer Neural Nets".

# Future Directions

- Can modifications to the PINN formalism be made to make them more sensitive to the local structure of the solutions and thus help detect the existence of finite-time blow-ups in PDEs?
- Are there any PINN losses for the $(d+1)$-dimensional Burgers – or for Navier-Stokes in general – that is "stable"[5], as was shown to be true in our $(1+1)$-dimensional Burgers' PDE setup?
- Do PINN loss functions for fluid PDEs satisfy the Villani condition[6], thereby establishing that Langevin Monte Carlo can effectively minimize PINN losses?

[5] Wang et al., "Is $L^2$ Physics Informed Loss Always Suitable for Training Physics Informed Neural Network?"
[6] Gopalani and Mukherjee, "Global Convergence of SGD On Two Layer Neural Nets".