Overcoming gradient pathologies in constrained neural networks

Paris Perdikaris
Department of Mechanical Engineering
University of Pennsylvania
email: pgp@seas.upenn.edu

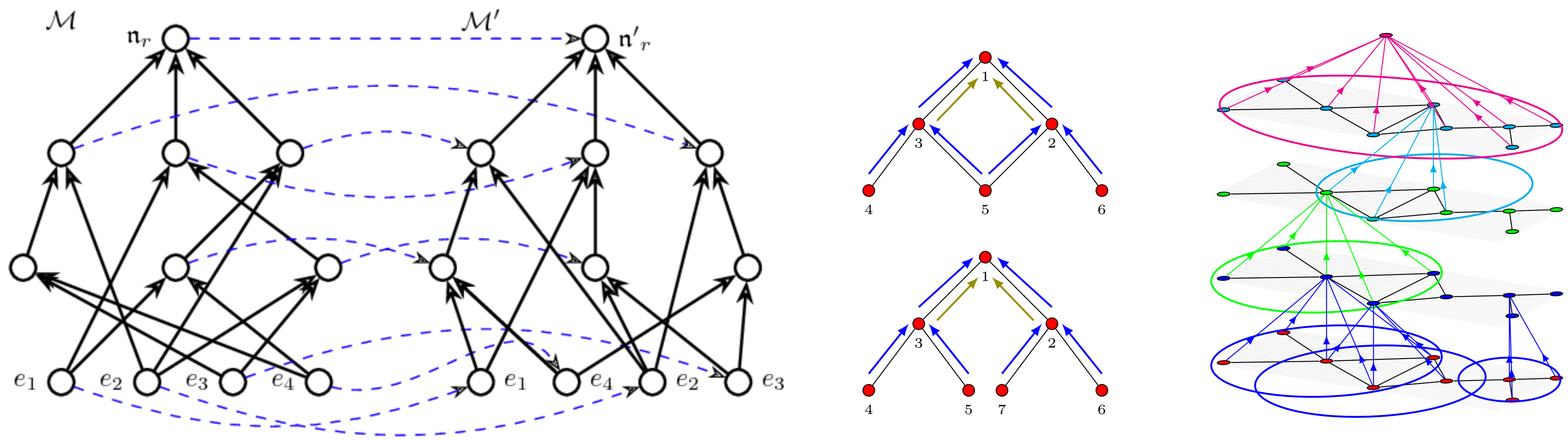
Sifan Wang
Graduate Program in Applied Mathematics
University of Pennsylvania
email: sifanw@sas.upenn.edu

Machine Learning for Physics and the Physics of Learning
Validation and Guarantees in Learning Physical Models (MLPWS3)
IPAM
October 29, 2019

Penn
UNIVERSITY *of* PENNSYLVANIA

Physics of AI: Two schools of thought

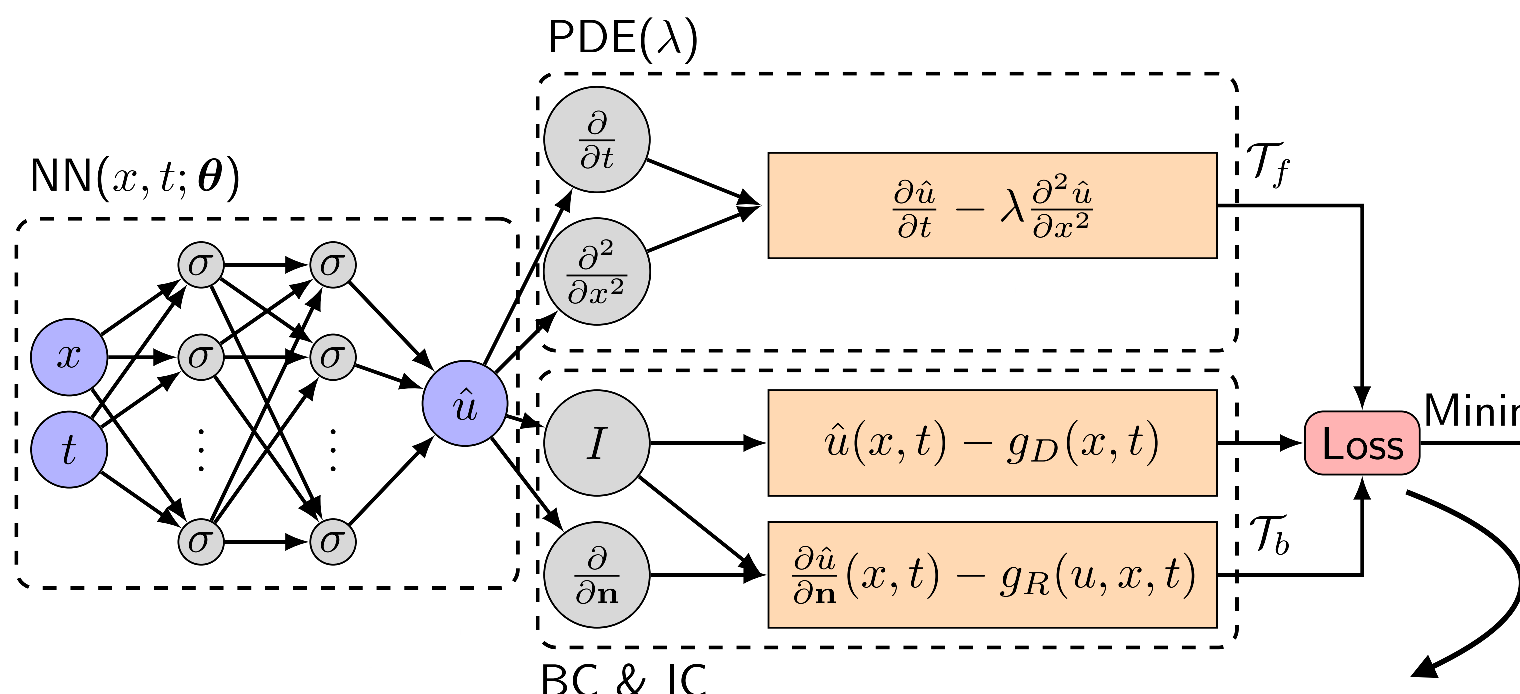
1. Physics is implicitly baked in specialized neural architectures with strong inductive biases (e.g. invariance to simple group symmetries).



**figures from Kondor, R., Son, H.T., Pan, H., Anderson, B., & Trivedi, S. (2018). Covariant compositional networks for learning graphs. arXiv preprint arXiv:1801.02144.*

2. Physics is explicitly imposed by constraining the output of conventional neural architectures with weak inductive biases.

Psichogios & Ungar, 1992
Lagaris et. al., 1998
Raissi et. al., 2019
Lu et. al., 2019
Zhu et. al., 2019

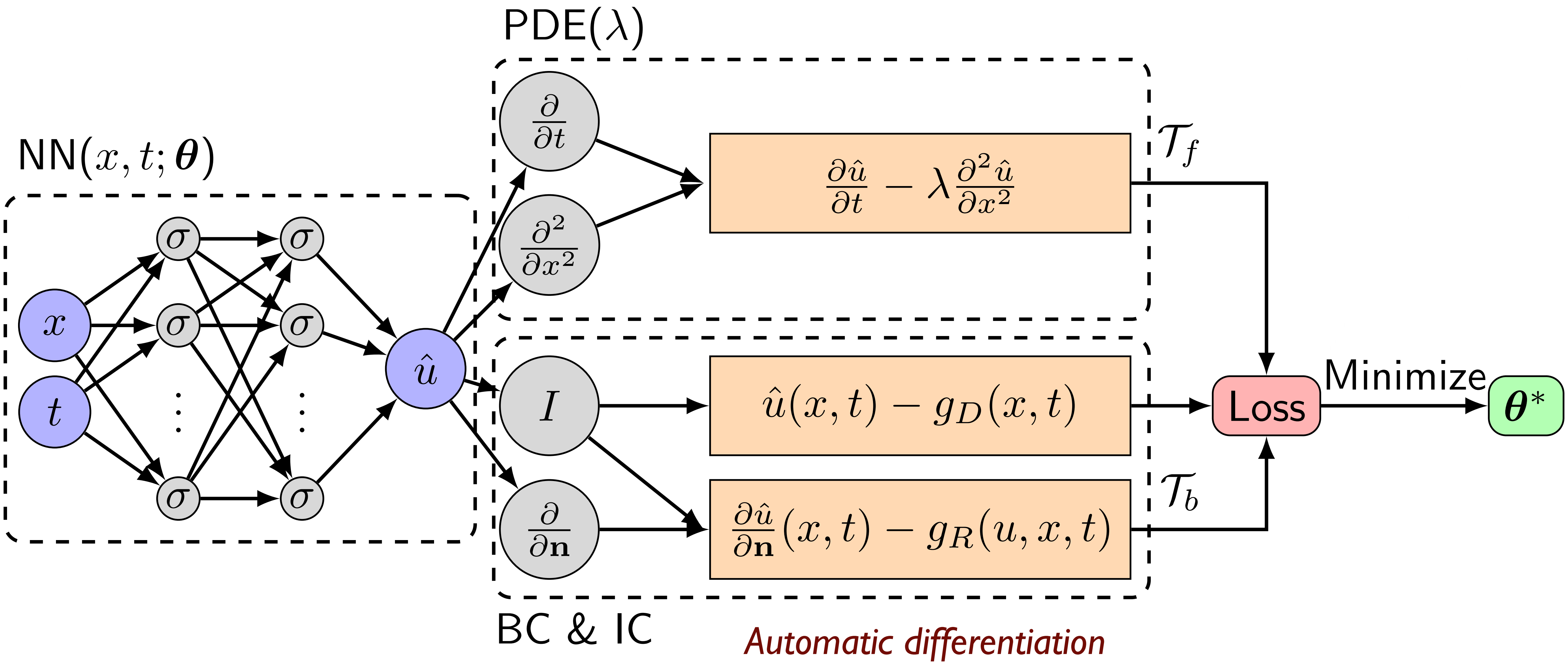


$$\mathcal{L}(\theta) := \underbrace{\frac{1}{N_u}\sum_{i=1}^{N_u}[\boldsymbol{u}_i - f_\theta(\boldsymbol{x}_i)]^2}_{\text{Data fit}} + \underbrace{\frac{1}{\lambda}\mathcal{R}[f_\theta(\boldsymbol{x})]}_{\text{Physics regularization}}$$

Physics-informed Neural Networks

$$f\left(\mathbf{x}; \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_d}; \frac{\partial^2 u}{\partial x_1 \partial x_1}, \ldots, \frac{\partial^2 u}{\partial x_1 \partial x_d}; \ldots; \boldsymbol{\lambda}\right) = 0, \quad \mathbf{x} \in \Omega,$$

$$\mathcal{B}(u, \mathbf{x}) = 0 \quad \text{on} \quad \partial\Omega,$$



Psichogios, D. C., & Ungar, L. H. (1992). A hybrid neural network–first principles approach to process modeling. AIChE Journal, 38(10), 1499-1511.

Lagaris, I. E., Likas, A., & Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. IEEE transactions on neural networks, 9(5), 987-1000.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). 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.

Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2019). DeepXDE: A deep learning library for solving differential equations. arXiv preprint arXiv: 1907.04502.

Physics-informed Neural Networks

Example: Burgers' equation in 1D

$$\begin{aligned} & u_t + uu_x - (0.01/\pi)u_{xx} = 0, \quad x \in [-1,1], \quad t \in [0,1], \\ & u(0,x) = -\sin(\pi x), \\ & u(t,-1) = u(t,1) = 0. \end{aligned} \tag{3}$$

Let us define $f(t,x)$ to be given by

$$f := u_t + uu_x - (0.01/\pi)u_{xx},$$

```python
def u(t, x):
    u = neural_net(tf.concat([t,x],1), weights, biases)
    return u
```

Correspondingly, the *physics informed neural network* $f(t,x)$ takes the form

```python
def f(t, x):
    u = u(t, x)
    u_t = tf.gradients(u, t)[0]
    u_x = tf.gradients(u, x)[0]
    u_xx = tf.gradients(u_x, x)[0]
    f = u_t + u*u_x - (0.01/tf.pi)*u_xx
    return f
```

Physics-informed Neural Networks

The shared parameters between the neural networks $u(t, x)$ and $f(t, x)$ can be learned by minimizing the mean squared error loss

$$MSE = MSE_u + MSE_f, \tag{4}$$

where

$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2,$$

and

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2.$$

Here, $\{t_u^i, x_u^i, u^i\}_{i=1}^{N_u}$ denote the initial and boundary training data on $u(t, x)$ and $\{t_f^i, x_f^i\}_{i=1}^{N_f}$ specify the collocations points for $f(t, x)$. The loss MSE_u corresponds to the initial and boundary data while MSE_f enforces the structure imposed by equation (3) at a finite set of collocation points.

Physics-informed Neural Networks

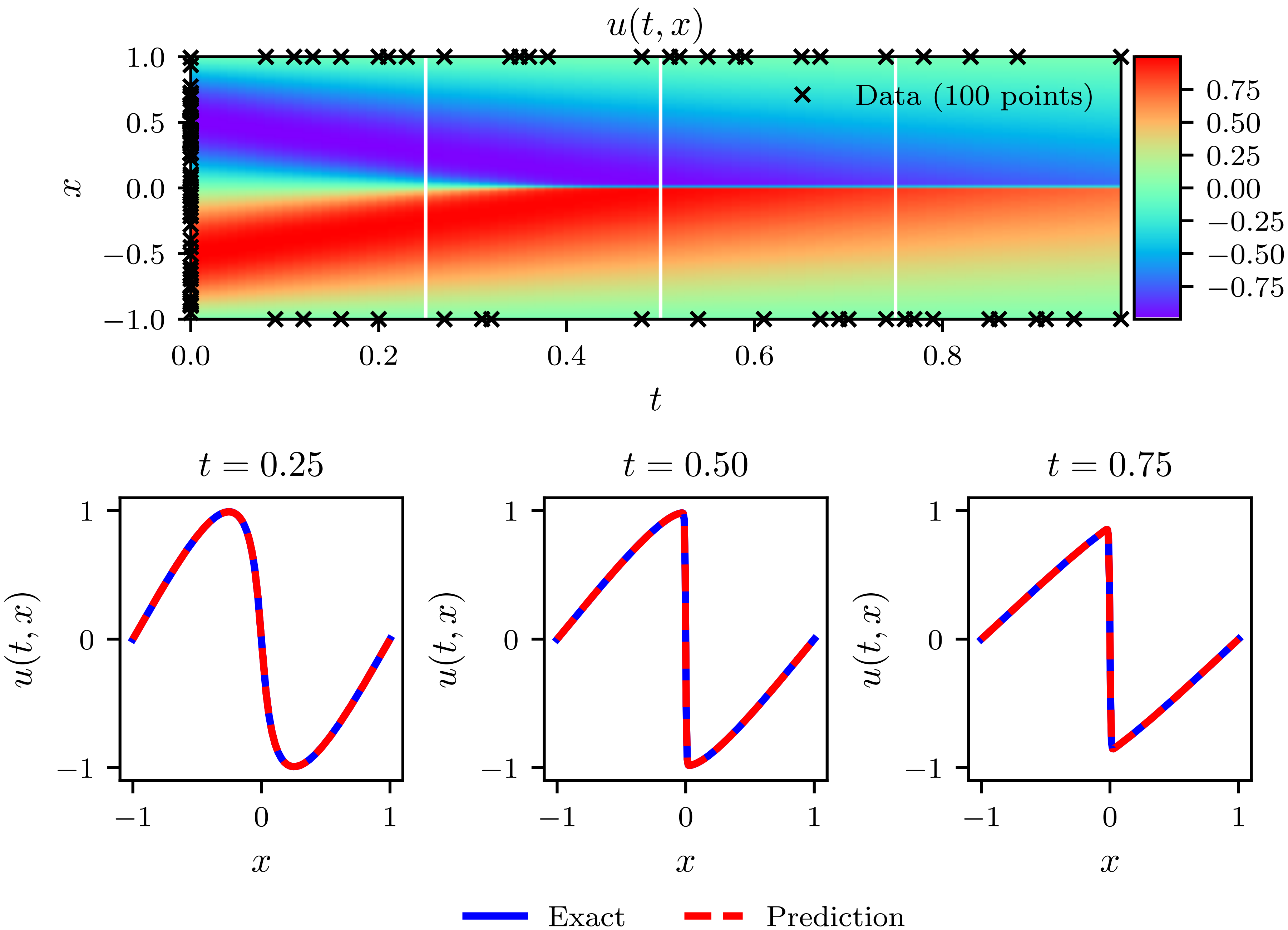
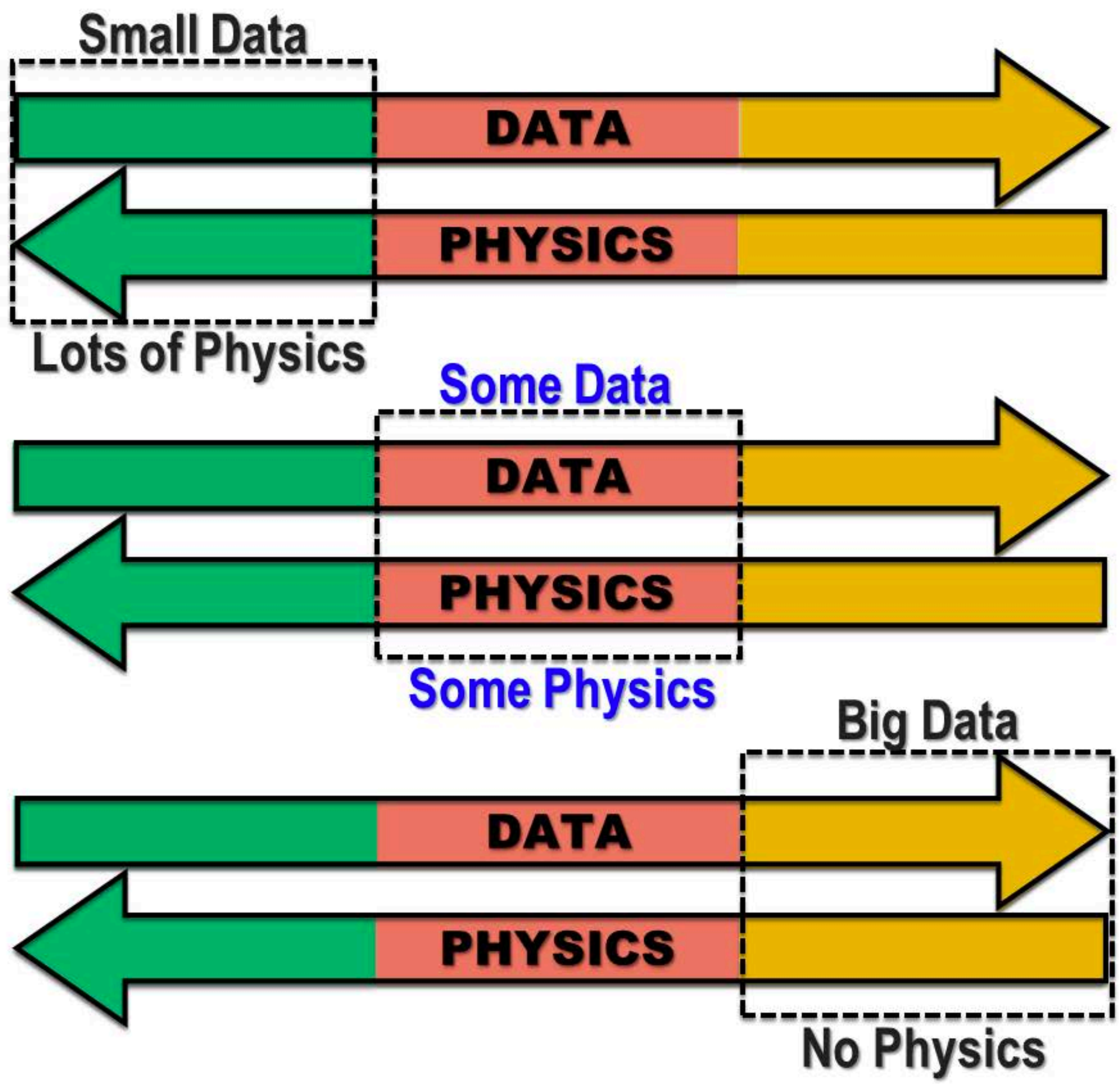
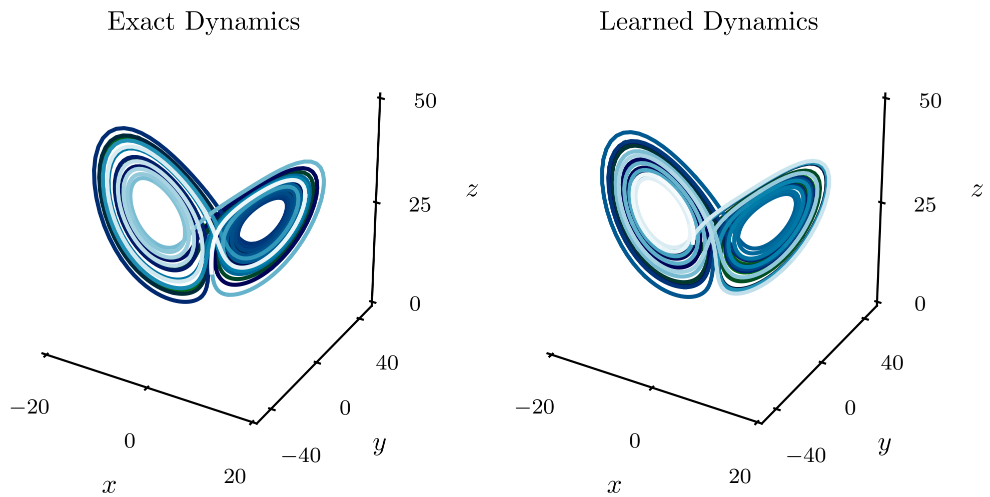


Figure 1: *Burgers' equation: Top:* Predicted solution $u(t, x)$ along with the initial and boundary training data. In addition we are using 10,000 collocation points generated using a Latin Hypercube Sampling strategy. *Bottom:* Comparison of the predicted and exact solutions corresponding to the three temporal snapshots depicted by the white vertical lines in the top panel. The relative $\mathcal{L}_2$ error for this case is $6.7 \cdot 10^{-4}$. Model training took approximately 60 seconds on a single NVIDIA Titan X GPU card.


Small Data
DATA
PHYSICS
Lots of Physics
Some Data
DATA
PHYSICS
Some Physics
Big Data
DATA
PHYSICS
No Physics

Recent advances

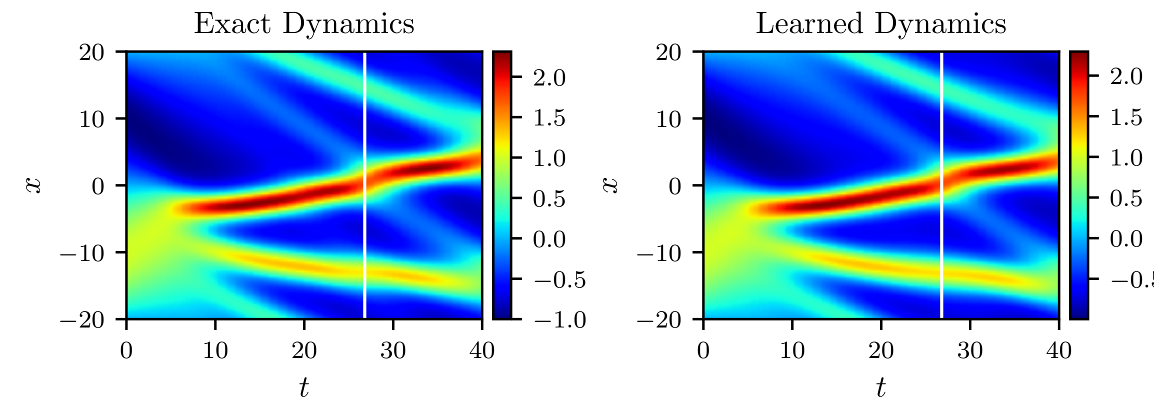
Discovery of ODEs



Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2018). Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems. arXiv preprint

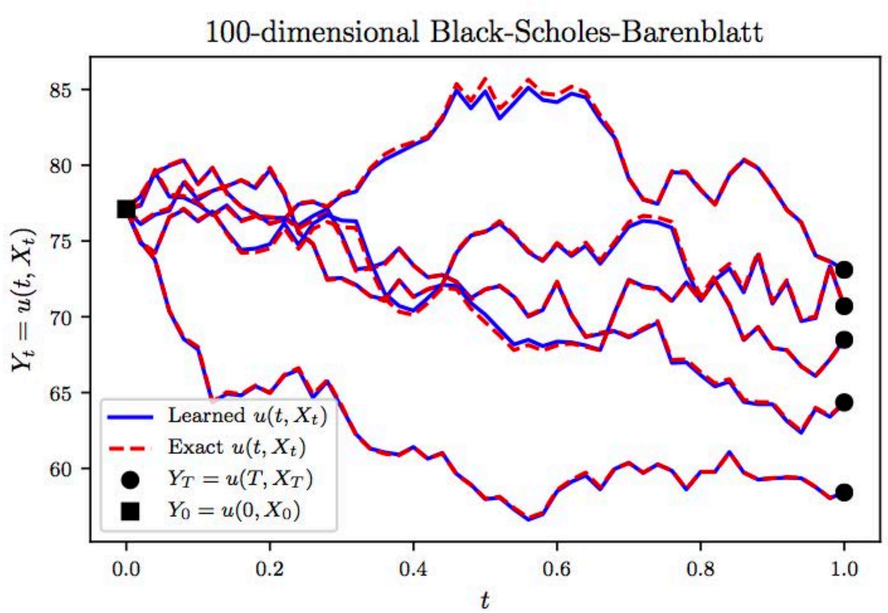
Discovery of PDEs

$$u_t = -uu_x - u_{xxx}$$



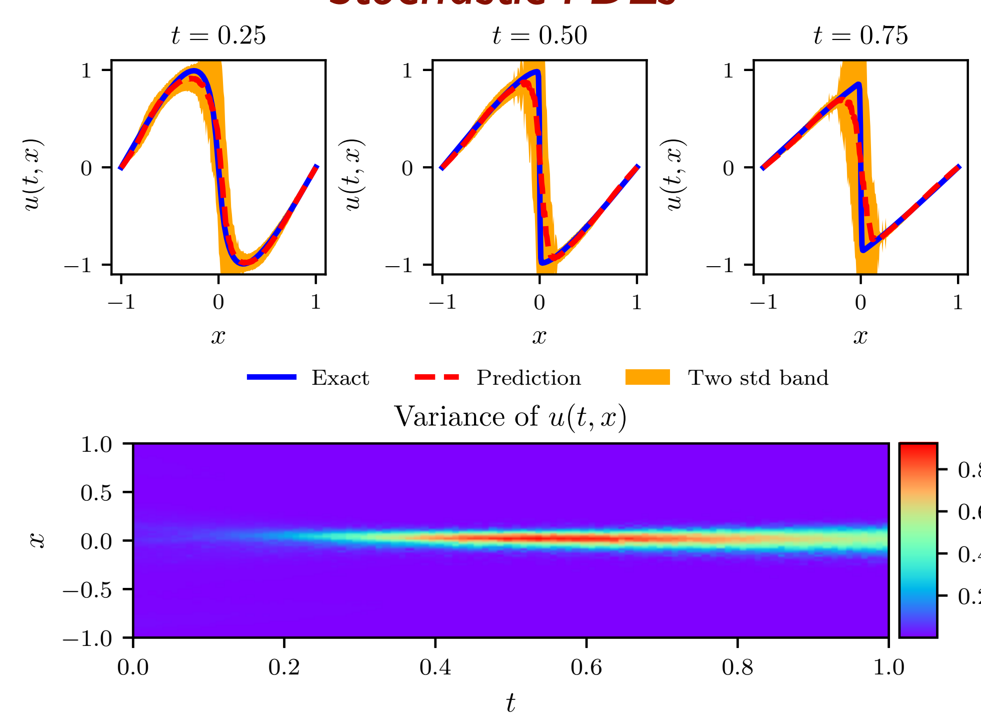
Raissi, M. (2018). Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations. arXiv preprint arXiv:1801.06637.

High-dimensional PDEs



Raissi, M. (2018). Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations. arXiv preprint arXiv: 1804.07010.

Stochastic PDEs



Yang, Y., & Perdikaris, P. (2019). Adversarial uncertainty quantification in physics-informed neural networks. Journal of Computational Physics.

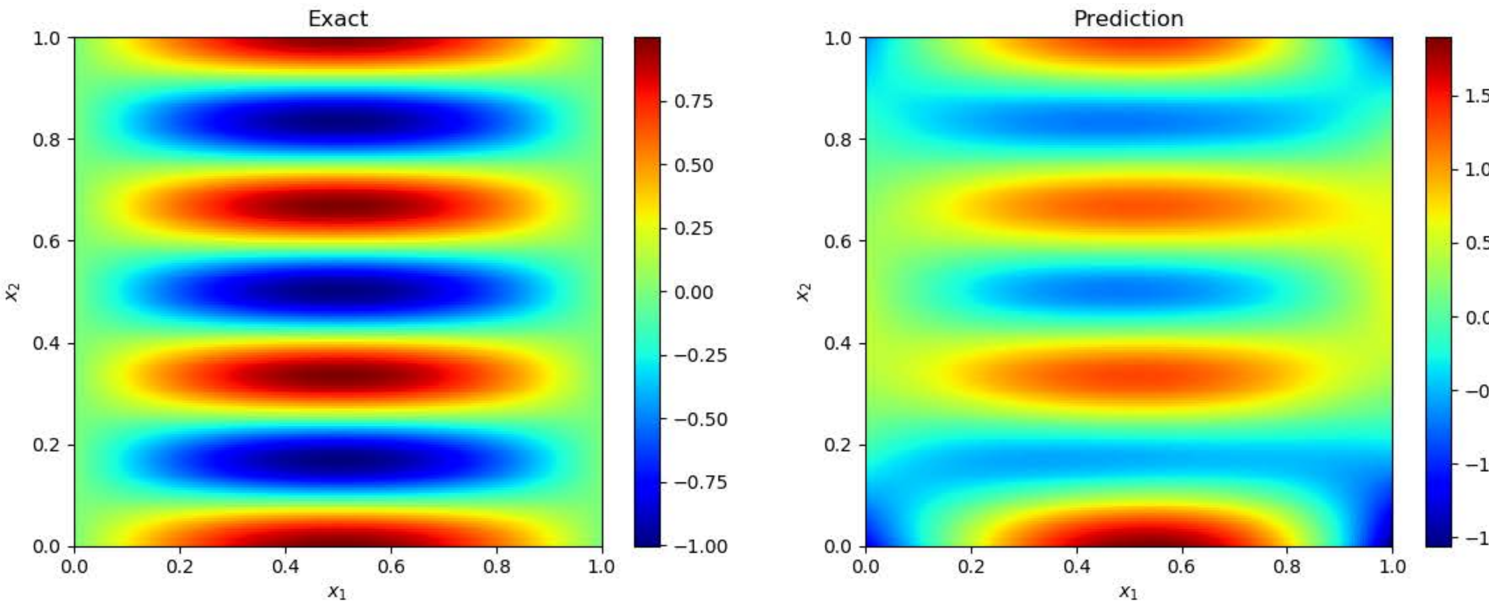
Physics as a regularizer/prior

$$\mathcal{L}(\theta) := \underbrace{\frac{1}{N_u}\sum_{i=1}^{N_u}[\boldsymbol{u}_i - f_\theta(\boldsymbol{x}_i)]^2}_{\text{Data fit}} + \underbrace{\frac{1}{\lambda}\mathcal{R}[f_\theta(\boldsymbol{x})]}_{\text{Physics regularization}}$$

Results showcase remarkable promise, but failure looms even for the simplest problems…

<u>*A simple benchmark:*</u>

$$\Delta u(x_1, x_2) = f(x_1, x_2)$$
$$u(x_1, x_2) = \sin(a_1 \pi x_1)\cos(a_2 \pi x_2)$$
$$f(x_1, x_2) = -(a_1^2\pi^2 + a_2^2\pi^2)u(x_1, x_2)$$



59% error in the prediction of a dense, 4-layer deep physics-informed neural network

An "unconventional" regularizer/prior that requires us to revisit standard deep learning practices:

- loss function
- network initialization
- data normalization
- optimization
- network architecture

→ This talk { Overcoming gradient pathologies in PINNs via:
- Adaptive learning rate strategies
- Resilient neural architectures

Gradient pathologies in physics-informed neural networks

$$\mathcal{L}(\theta) := \underbrace{\mathcal{L}_u(\theta)}_{\text{Data fit}} + \underbrace{\mathcal{L}_r(\theta)}_{\text{PDE residual}} + \underbrace{\mathcal{L}_{u_0}(\theta)}_{\text{ICs fit}} + \underbrace{\mathcal{L}_{u_b}(\theta)}_{\text{BCs fit}}$$

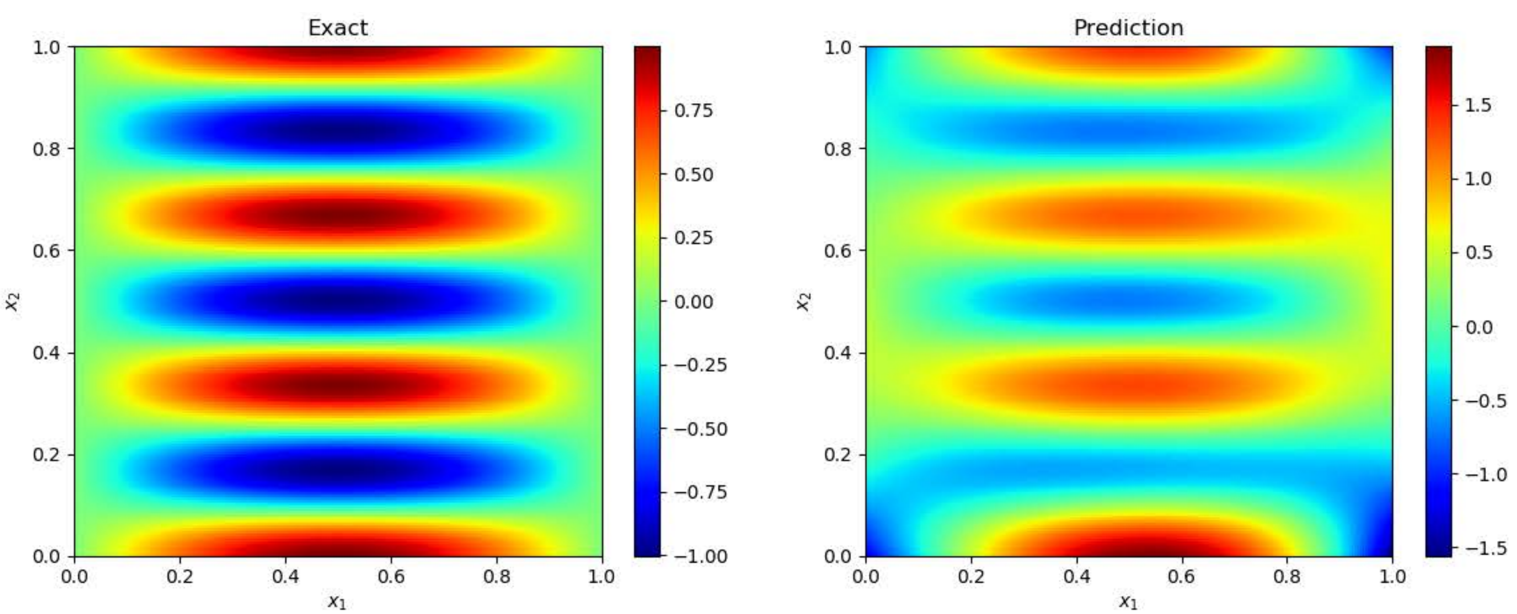
<u>Hypothesis:</u> Constraints alter the loss landscape of neural networks. Different terms in such composite loss functions may have different nature and magnitudes, leading to *imbalanced gradients* during back-propagation.

Gradient descent update:

$$\begin{aligned}\theta_{n+1} &= \theta_n - \eta \nabla_\theta \mathcal{L}(\theta_n) \\ &= \theta_n - \eta\{\nabla_\theta \mathcal{L}_u(\theta_n) + \nabla_\theta \mathcal{L}_r(\theta_n) + \nabla_\theta \mathcal{L}_{u_0}(\theta_n) + \nabla_\theta \mathcal{L}_{u_b}(\theta_n)\}\end{aligned}$$

A simple benchmark:

$$\begin{aligned}\Delta u(x_1, x_2) &= f(x_1, x_2) \\ u(x_1, x_2) &= \sin(a_1 \pi x_1)\cos(a_2 \pi x_2) \\ f(x_1, x_2) &= -(a_1^2 \pi^2 + a_2^2 \pi^2) u(x_1, x_2)\end{aligned}$$

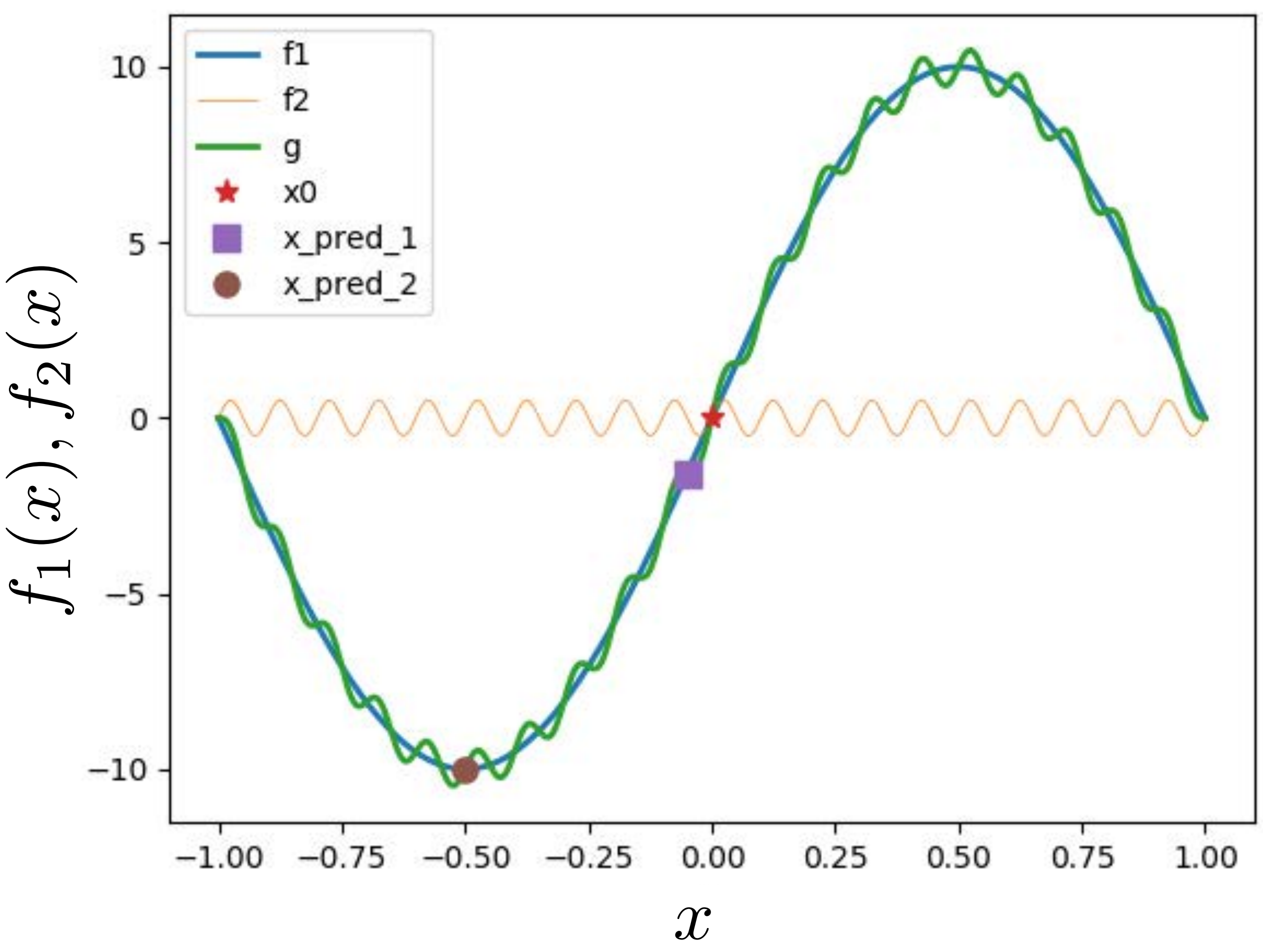


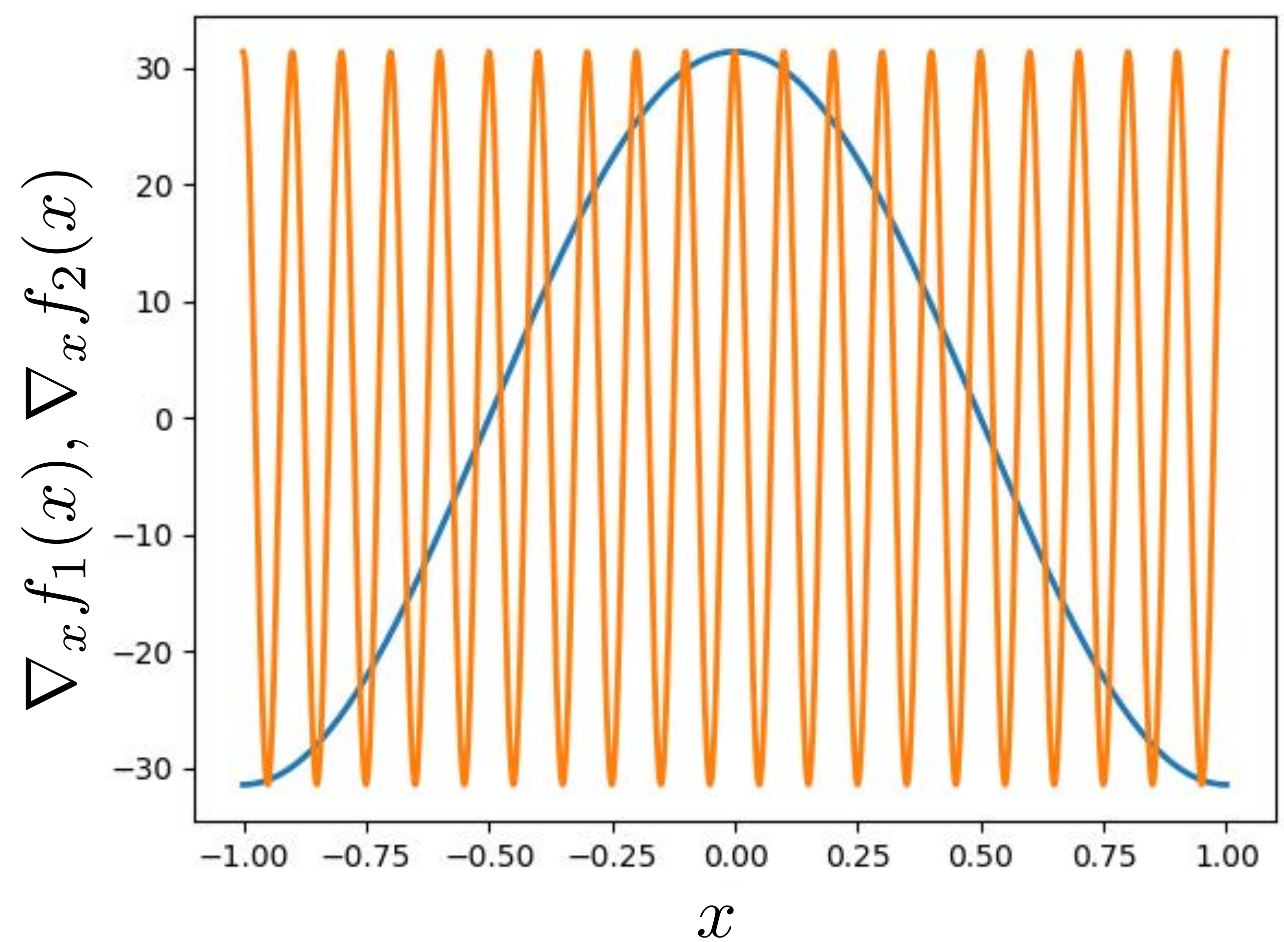
59% error in the prediction of a dense, 4-layer deep physics-informed neural network

Some intuition

A pedagogical example:
Minimization of an additive objective with multi-scale behavior: $\min f_1(x) + f_2(x)$

$$x_{n+1} = x_n - \eta\{\nabla_x f_1(x) + \nabla_x f_2(x)\}$$





Gradient descent is bound to get trapped in local suboptimal minima

Hypothesis: Adaptively selecting different learning rates that balance the interplay between the different loss terms can lead to improved solutions:

$$x_{n+1} = x_n - \eta_1^{(n)} \nabla_x f_1(x) - \eta_2^{(n)} \nabla_x f_2(x)$$

Gradient pathologies in physics-informed neural networks

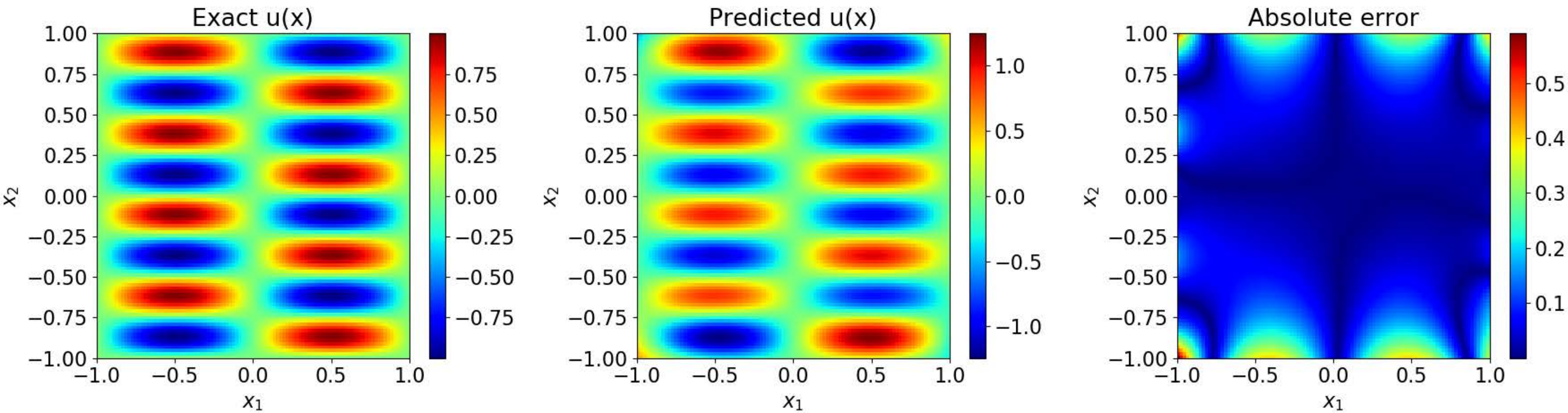
A simple benchmark (2D Helmholtz equation):

$$\Delta u + k^2 u = q(x, y) \quad (x, y) \in [-1, 1]$$

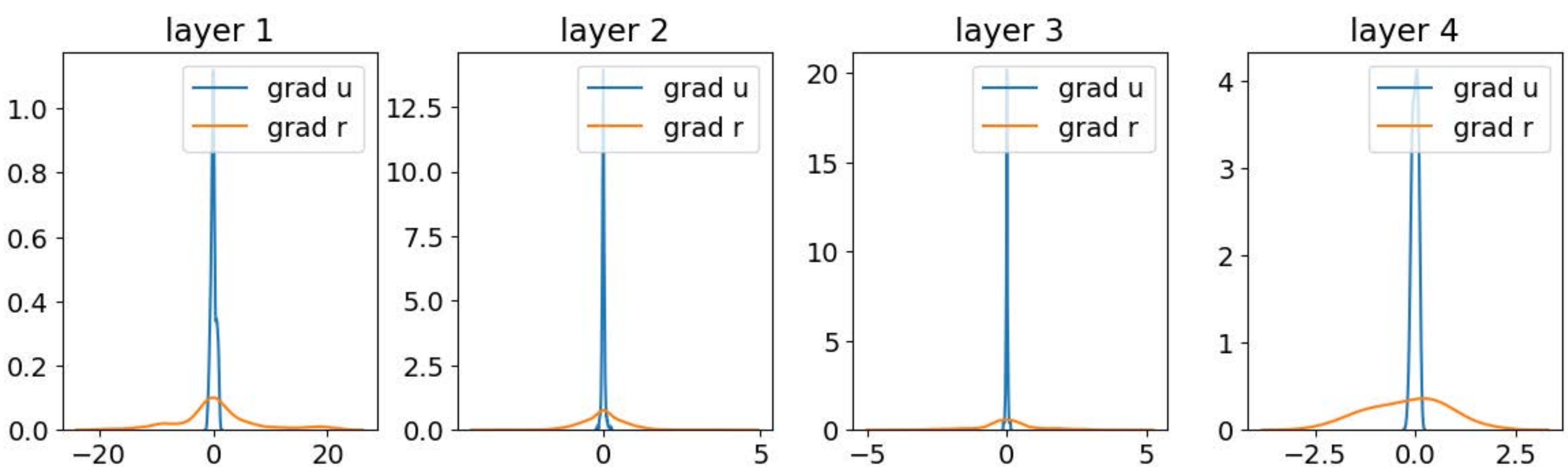
$$u(x, y) = (x + y)\sin(\pi x)\sin(6\pi y)$$

Loss function:

$$\mathcal{L}(\theta) := \underbrace{\mathcal{L}_r(\theta)}_{\text{PDE residual}} + \underbrace{\mathcal{L}_{u_b}(\theta)}_{\text{BCs fit}}$$



Prediction of a fully connected 4-layer deep physics-informed neural network (10% relative error)



Histograms of back-propagated gradients $\nabla_\theta \mathscr{L}_{u_b}(\theta), \nabla_\theta \mathscr{L}_r(\theta)$ *at each hidden layer*

Gradient pathologies in physics-informed neural networks

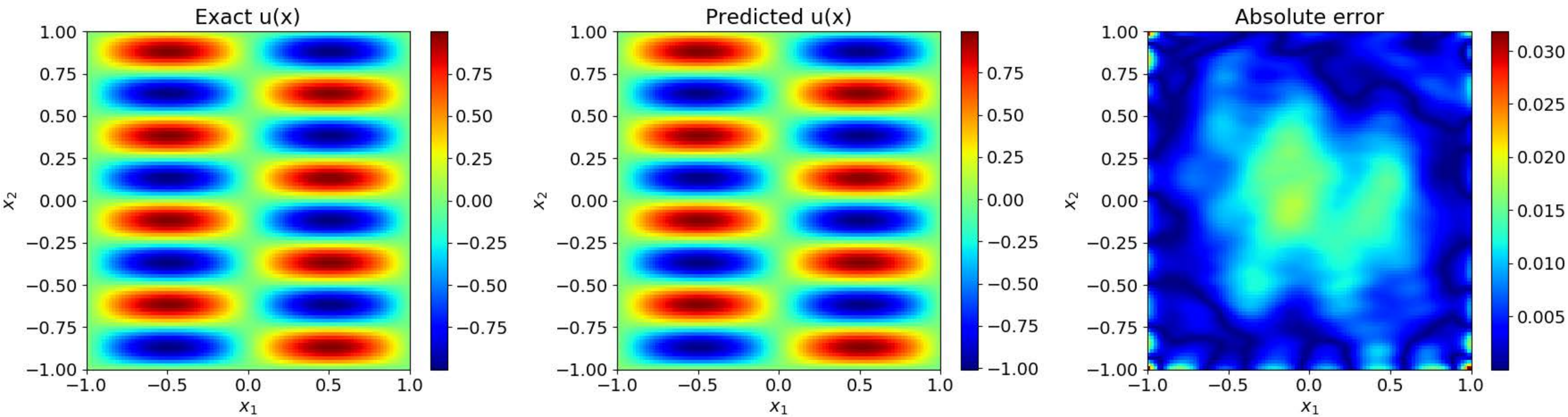
A simple benchmark (2D Helmholtz equation):

$$\Delta u + k^2 u = q(x,y) \quad (x,y) \in [-1,1]$$

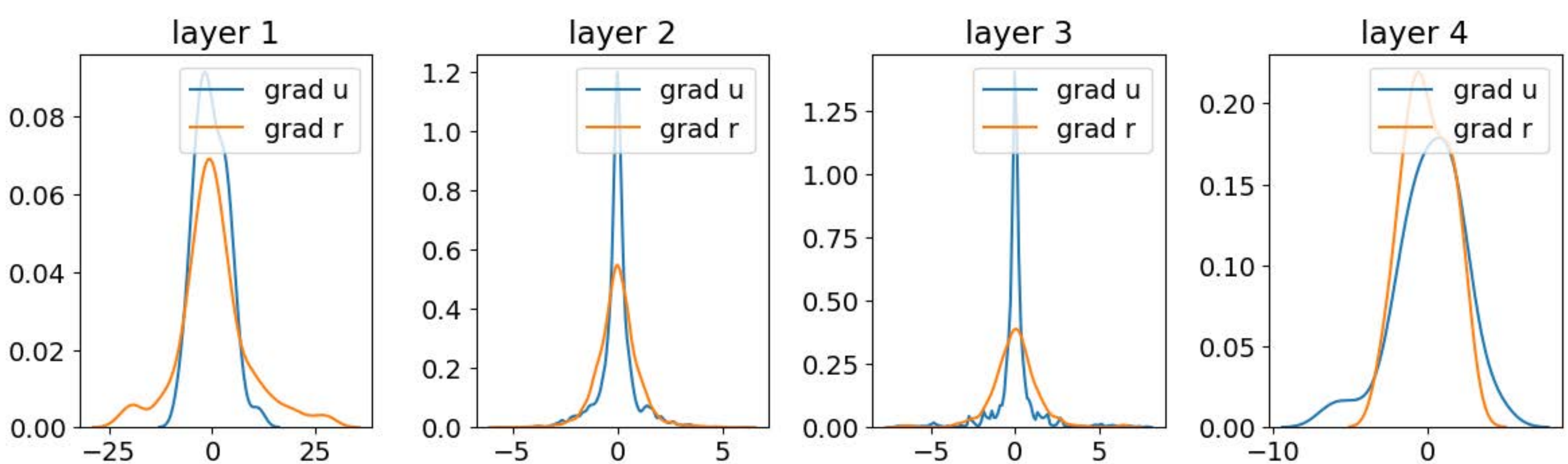
$$u(x,y) = (x+y)\sin(\pi x)\sin(6\pi y)$$

Loss function:

$$\mathcal{L}(\theta) := \lambda_1 \underbrace{\mathcal{L}_r(\theta)}_{\text{PDE residual}} + \lambda_2 \underbrace{\mathcal{L}_{u_b}(\theta)}_{\text{BCs fit}}$$



Prediction of a fully connected 4-layer deep physics-informed neural network (0.5% relative error)



Histograms of back-propagated gradients $\nabla_\theta \mathscr{L}_{u_b}(\theta), \nabla_\theta \mathscr{L}_r(\theta)$ at each hidden layer

…but how to choose the weights/learning rates?

$$\mathcal{L}(\theta) := \lambda_1 \underbrace{\mathcal{L}_u(\theta)}_{\text{Data fit}} + \lambda_2 \underbrace{\mathcal{L}_r(\theta)}_{\text{PDE residual}} + \lambda_3 \underbrace{\mathcal{L}_{u_0}(\theta)}_{\text{ICs fit}} + \lambda_4 \underbrace{\mathcal{L}_{u_b}(\theta)}_{\text{BCs fit}}$$

Adaptive moment estimation

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

…i.e. use the gradient statistics during training to adaptively adjust the learning rate.

A learning rate annealing algorithm for PINNs

Algorithm 1: Learning rate annealing for physics-informed neural networks

Consider a physics-informed neural network $f_\theta(\boldsymbol{x})$ with parameters θ. and a loss function

$$\mathcal{L}(\theta) = \mathcal{L}_r(\theta) + \sum_{i=1}^{M} \lambda_i \mathcal{L}_i(\theta),$$

where $\mathcal{L}_r(\theta)$ denotes the PDE residual loss, the $\mathcal{L}_i(\theta)$ correspond to data-fit terms (e.g., measurements, initial or boundary conditions, etc.), and $\lambda_i = 1$, $i = 1, \ldots, M$.

Use N steps of a gradient descent algorithm to update the parameters θ as:

for $n = 1, \ldots, N$ **do**

(a) Compute $\hat{\lambda}_i$ by

$$\hat{\lambda}_i = \frac{\max_\theta \{\nabla_\theta \mathcal{L}_r(\theta_n)\}}{|\nabla_\theta \mathcal{L}_i(\theta_n)|_{.k}}, \quad i = 1, \ldots, M,$$

where $|\nabla_\theta \mathcal{L}_i(\theta_n)|_{.k}$ denotes the k-th percentile of the set $\{|\nabla_\theta \mathcal{L}_i(\theta_n)|\}$.

(b) Update the weights λ_i using a moving average of the form

$$\lambda_i = (1 - \alpha)\lambda_i + \alpha \hat{\lambda}_i, \quad i = 1, \ldots, M.$$

(c) Update the parameters θ via gradient descent

$$\theta_{n+1} = \theta_n - \eta \nabla_\theta \mathcal{L}_r(\theta_n) - \eta \sum_{i=1}^{M} \lambda_i \nabla_\theta \mathcal{L}_i(\theta_n)$$

end

The recommended hyper-parameter values are: $\eta = 10^{-3}$, $k = 0.05$ and $\alpha = 0.9$.

Systematic comparison

M1: Baseline PINN model (Raissi et. al., 2019)
M2: PINN with the proposed learning rate annealing

Architecture	M1	M2
30 units / 3 hidden layers	2.44E-01	**3.98E-02**
50 units / 3 hidden layers	1.06E-01	**1.58E-02**
100 units / 3 hidden layers	9.07E-02	**2.39E-03**
30 units / 5 hidden layers	2.47E-01	**8.91E-03**
50 units / 5 hidden layers	1.40E-01	**8.08E-03**
100 units / 5 hidden layers	1.15E-01	**3.25E-03**
30 units / 7 hidden layers	3.10E-01	**7.86E-03**
50 units / 7 hidden layers	1.98E-01	**3.66E-03**
100 units / 7 hidden layers	8.14E-02	**2.57E-03**

Relative prediction error (L2 norm) averaged over 10 independent trials for the 2D Helmholtz benchmark.

Soft physics-informed learning, a recap

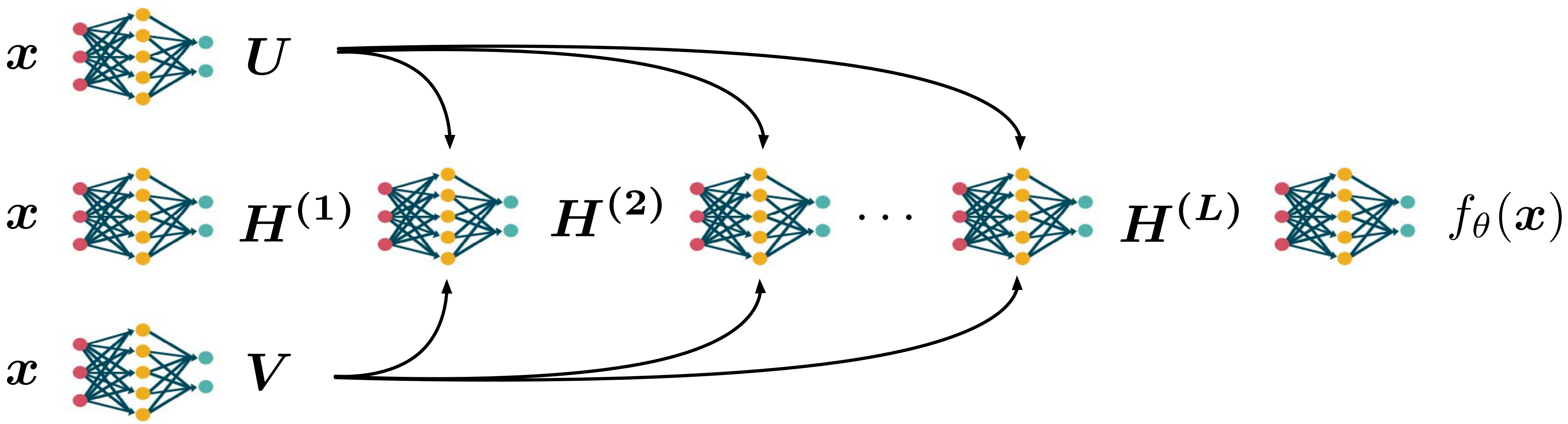
$$\mathcal{L}(\theta) := \underbrace{\frac{1}{N_u}\sum_{i=1}^{N_u}[\boldsymbol{u}_i - f_\theta(\boldsymbol{x}_i)]^2}_{\text{Data fit}} + \underbrace{\frac{1}{\lambda}\mathcal{R}[f_\theta(\boldsymbol{x})]}_{\text{Physics regularization}}$$

An "unconventional" regularizer/prior that requires us to revisit standard deep learning practices:

- loss functions (e.g., square residual, *variational principle, Hamiltonian, etc.?*)
- network initialization (e.g., Glorot, *adaptive?*)
- normalization (e.g., zero-mean/unit-variance, *PDE solution bounds?*)
- optimization (e.g., Adam, *adaptive learning rates, proximal algorithms, meta-learning?*)
- network architecture (e.g., fully connected, *residual/recurrent/convolutional layers, attention?*)

An improved neural architecture

$$U = \phi(W^1 \vec{x} + b^1), \;\; V = \phi(W^2 \vec{x} + b^2)$$
$$H^{(1)} = \phi(W^{z,1} \vec{x} + b^{z,1})$$
$$Z^{(k)} = \phi(W^{z,k} H^{(k)} + b^{z,k}), \;\; k = 1, \ldots, L$$
$$H^{(k+1)} = (1 - Z^{(k)}) \odot U + Z^{(k)} \odot V, \;\; k = 1, \ldots, L$$
$$f(x;\theta) = W H^{(L+1)} + b$$



$$\theta = \{W^1, b^1, W^2, b^2, (W^{z,l}, b^{z,l})_{l=1}^{L}, W, b\}$$

Key points:

- Account for multiplicative interactions of the inputs, similar to attention mechanisms.
- Residual connections improve resilience against vanishing gradient pathologies.

Systematic comparison

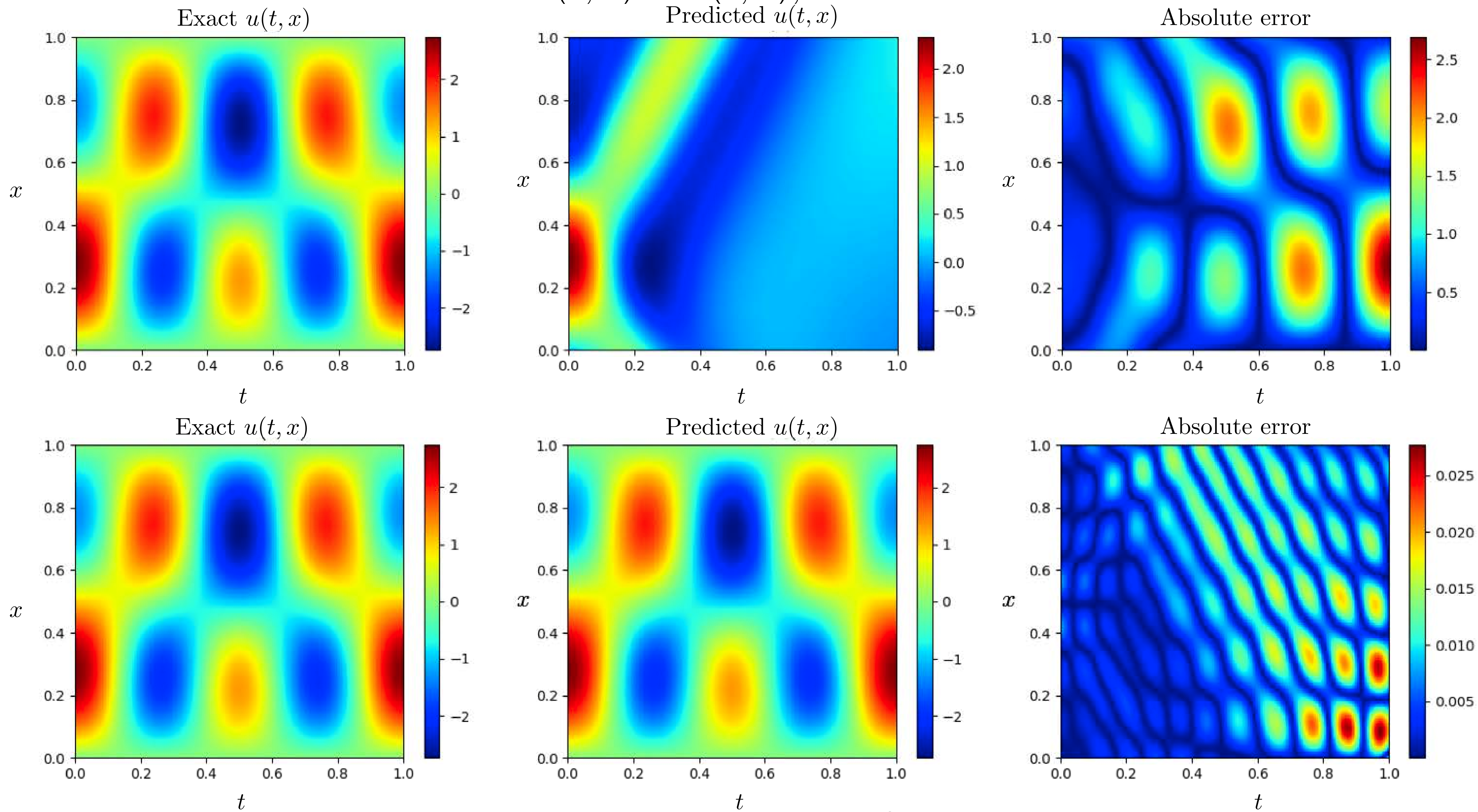
M1: Baseline PINN model (Raissi et. al., 2019)
M2: PINN with the proposed learning rate annealing
M3: PINN with the proposed neural architecture
M4: PINN with the proposed learning rate annealing and improved neural architecture

Architecture	M1	M2	M3	M4
30 units / 3 hidden layers	2.44E-01	3.98E-02	5.31E-02	**2.56E-03**
50 units / 3 hidden layers	1.06E-01	1.58E-02	2.46E-02	**1.81E-03**
100 units / 3 hidden layers	9.07E-02	2.39E-03	1.17E-02	**1.28E-03**
30 units / 5 hidden layers	2.47E-01	8.91E-03	4.12E-02	**1.96E-03**
50 units / 5 hidden layers	1.40E-01	8.08E-03	1.97E-02	**1.86E-03**
100 units / 5 hidden layers	1.15E-01	3.25E-03	1.08E-02	**1.22E-03**
30 units / 7 hidden layers	3.10E-01	7.86E-03	3.17E-02	**1.98E-03**
50 units / 7 hidden layers	1.98E-01	3.66E-03	2.37E-02	**1.54E-03**
100 units / 7 hidden layers	8.14E-02	2.57E-03	9.36E-03	**1.40E-03**

Relative prediction error (L2 norm) averaged over 10 independent trials for the 2D Helmholtz benchmark.

Wave equation

$$u_{tt} = 4u_{xx}, \quad (t,x) \in [0,1]$$
$$u(0,x) = h(x),$$
$$u(t,0) = u(t,1),$$



Top: Imbalanced gradients in a dense, 5-layer deep physics-informed neural network lead to large prediction errors (76%).

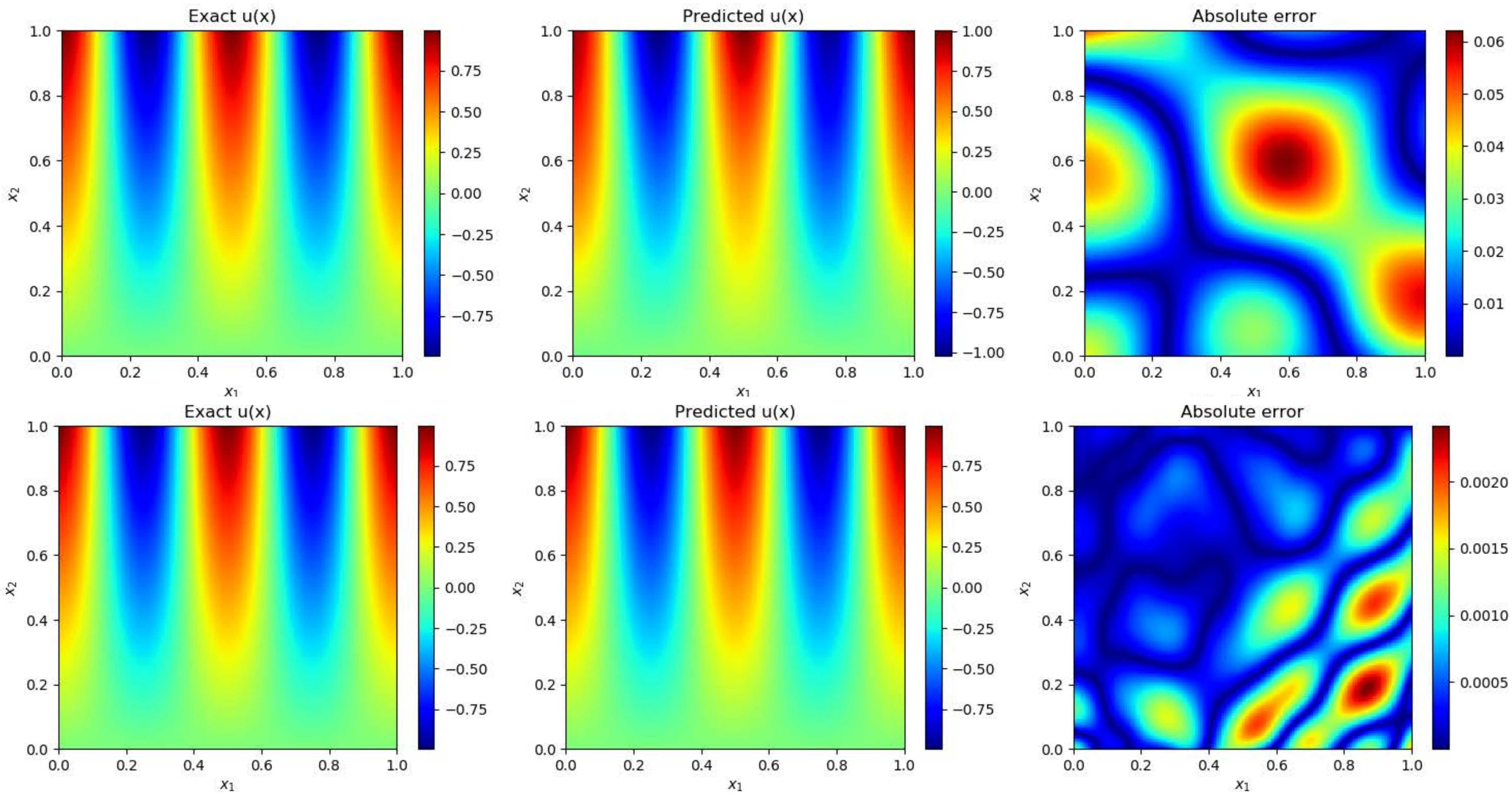
Bottom: Accurate predictions can be obtained using the proposed learning rate annealing and improved neural architecture strategy (relative prediction error: 0.6%).

Klein Gordon equation

$$u_{tt} + \alpha u_{xx} + \beta u + \gamma u^k = f(x,t), \ \ (x,t) \in \Omega \times [0,T]$$

$$u(x,0) = g_1(x), \ \ x \in \Omega \qquad u_(x,t) = h(x,t), \ \ (x,t) \in \partial\Omega \times [0,T]$$

$$u_t(x,0) = g_2(x), \ \ x \in \Omega$$



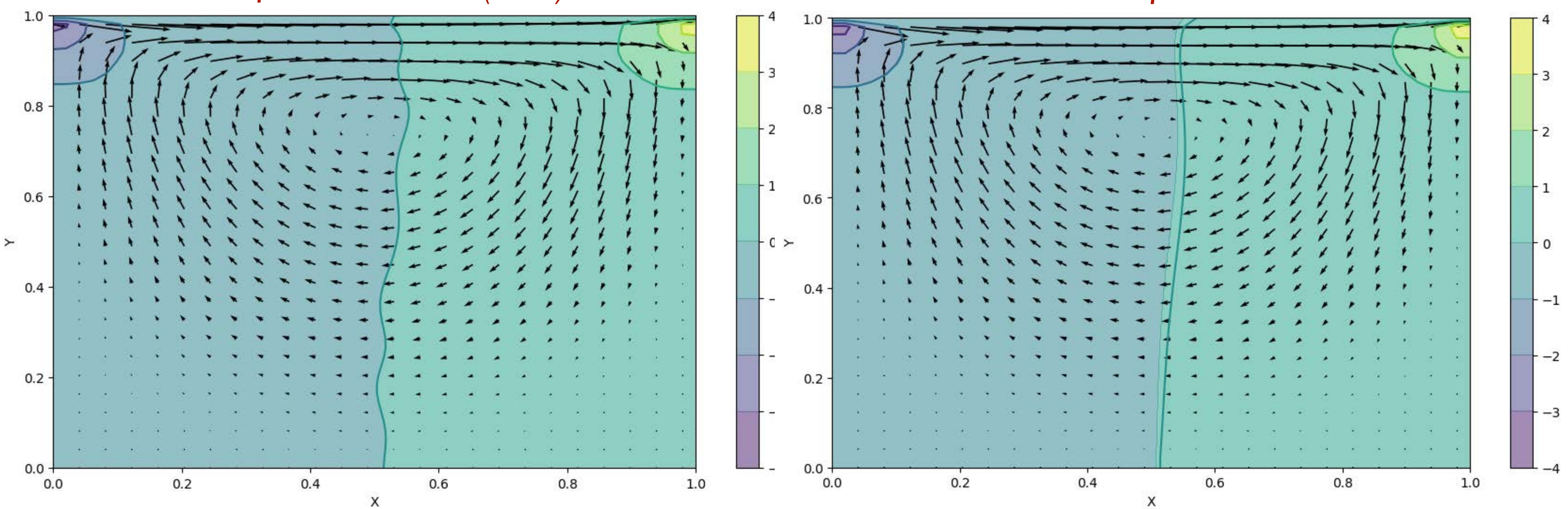
Top: Imbalanced gradients in a dense, 5-layer deep physics-informed neural network lead to considerable prediction errors (6.7%).

Bottom: Accurate predictions can be obtained using the proposed learning rate annealing and improved neural architecture strategy (relative prediction error: 0.1%).

Lid-driven cavity flow in 2D

$$\begin{aligned}
&\partial_t U + (U \cdot \Delta)U + \Delta p - \frac{1}{Re}\Delta U = 0 && \text{in } (0,T)\times\Omega \\
&\nabla \cdot U = 0, && \text{in } (0,T)\times\Omega \\
&U(t,x,y) = (1,0) && \text{on } (0,T)\times\Gamma_1 \\
&U(t,x,y) = (0,0) && \text{on } (0,T)\times\Gamma_0 \\
&U(0,x,y) = (0,0) && \text{in } \Omega
\end{aligned}$$

Reference solution (FVM) *PINNs prediction*



Relative prediction error (L2 norm) is ~1% for the velocity field and pressure.

Summary

- Function space constraints in introduce "unconventional" regularizers/priors that requires us to revisit standard deep learning practices.

- Constraints alter the loss landscape of neural networks. Different terms in such composite loss function may have different nature and magnitudes, leading to imbalanced gradients during back-propagation.

- Adaptive annealing of learning rates can balance the interplay between different terms in a constrained loss function and lead to improved solutions.

- Novel architectures can also safe-guard against gradient-related pathologies and lead to improved solutions.

- Using the proposed workflow we have observed consistent improvements in the predictive accuracy of physics-informed neural networks by a factor of 50-100x across a range of problems in computational physics.

- Despite some progress, we are still at the very early stages of understanding the capabilities and limitations of such models.

Acknowledgements:

Sifan Wang (UPenn)

U.S. DEPARTMENT OF ENERGY

DARPA

Wang, S., & Perdikaris, P. (2019). Fixing a broken PINN: Overcoming gradient pathologies in physics-informed neural networks (to appear on arXiv soon).

Penn
UNIVERSITY of PENNSYLVANIA

Email: pgp@seas.upenn.edu