

Robust Physics Informed Neural Networks

Maciej Paszyński



Faculty of Computer Science
AGH University of Krakow



Marcin Łoś, Maciej Paszyński,
Robust Physics Informed Neural Networks **arXiv:2401.02300v2** (2024)

Outline

- **Motivation:** *Physics Informed Neural Networks (PINN)*
 - Poisson problem with sin-sin solution
 - Poisson problem with exp-sin solution
 - Advection-diffusion problem
- **Solution:** *Robust Physics Informed Neural Networks (RPINN)*
 - Poisson problem with sin-sin solution
 - Poisson problem with exp-sin solution
 - Advection-diffusion problem
- **Method:** *Abstract framework*
 - Weak formulation in discrete setup
 - Boundness and inf-suf stable property
 - Inner product, true error and Gram matrix
 - Relation between loss function and true error
 - Open source code in Google Colab

References

- [1] Marcin Łoś, Maciej Paszyński, *Robust Physics Informed Neural Networks*, **arXiv:2401.02300v2** (2024)
- [2] Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszynski, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024) 116904
- [3] Dongho Shin, John Stikwerda, *Inf-sup conditions for finite differences approximations of the Stokes equations*, **Journal of Australian Mathematical Society** (1997)
- [4] Paweł Maczuga, Maciej Skoczeń, Przemysław Rożnawski, Filip Tłuszcz, Marcin Szubert, Marcin Łoś, Witold Dzwiniel, Keshav Pingali, Maciej Paszyński, *Physics Informed Neural Network Code for 2D Transient Problems (PINN-2DT) Compatible with Google Colab*, **arXiv:2310.03755** (2024)

Poisson problem with sin-sin solution

Given $\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek the solution of

$$-\Delta u = f, \quad (1)$$

with zero Dirichlet b.c. In this problem we select the solution

$$u(x_1, x_2) = \sin(2\pi x_1) \sin(2\pi x_2). \quad (2)$$

We employ the manufactured solution technique

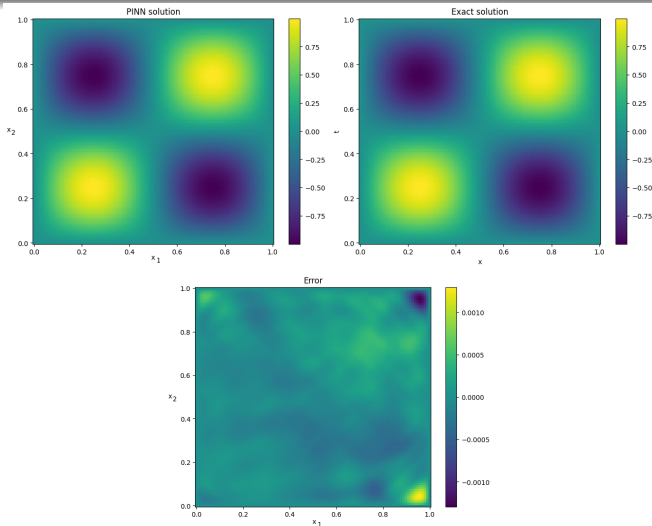
$$f_1(x_1, x_2) = -\Delta u(x_1, x_2) = 8\pi^2 \sin(2\pi x_1) \sin(2\pi x_2). \quad (3)$$

We define the following loss function for PINN

$$LOSS(\theta) = RES_1(\theta) = \sum_{\mathbf{x}} (\Delta u_{\theta}(\mathbf{x}) + f_1(\mathbf{x}))^2 \quad (4)$$

Physics Informed Neural Networks

Poisson problem with sin-sin solution



Physics Informed Neural Networks

Poisson problem with sin-sin solution



Poisson problem with sin-exp solution

Given $\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek the solution of

$$-\Delta u = f, \quad (5)$$

with zero Dirichlet b.c. In this problem we select the solution

$$u(x_1, x_2) = -e^{\pi(x_1 - 2x_2)} \sin(2\pi x_1) \sin(\pi x_2). \quad (6)$$

We employ the manufactured solution technique

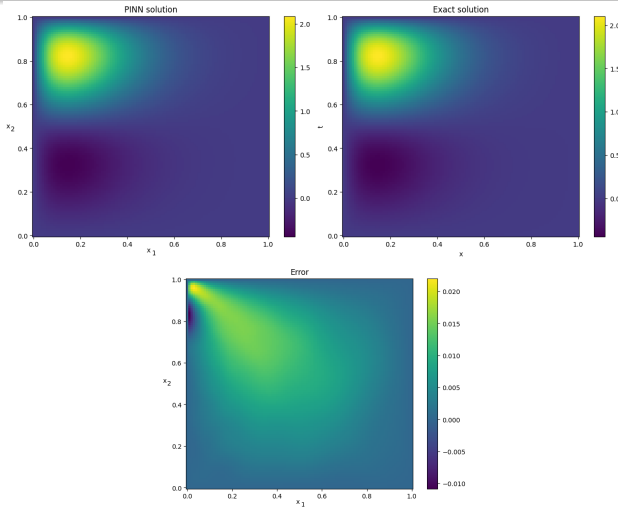
$$\begin{aligned} f_2(x_1, x_2) &= -\Delta u(x_1, x_2) = \\ &= \pi^2 e^{\pi(x-2y)} \sin(\pi y) (4 \cos(2\pi x) - 3 \sin(2\pi x)) \\ &\quad - \pi^2 e^{\pi(x-2y)} \sin(2\pi x) (4 \cos(\pi y) - 3 \sin(\pi y)) \end{aligned} \quad (7)$$

We define the following loss function for PINN

$$LOSS(\theta) = RES_2(\theta) = \sum_{\mathbf{x}} (\Delta u_{\theta}(\mathbf{x}) + f_2(\mathbf{x}))^2 \quad (8)$$

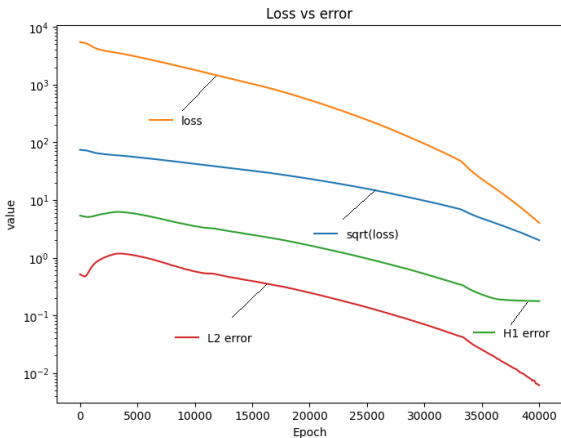
Physics Informed Neural Networks

Poisson problem with sin-exp solution



Physics Informed Neural Networks

Poisson problem with sin-exp solution



Advection-diffusion problem

$\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek solution of the Eriksson-Johnson problem
 Kenneth Eriksson, Claes Johnson. *Adaptive finite element methods for parabolic problems I: A linear model problem*. **SIAM Journal of Numerical Analysis** 1991

$$\begin{cases} \beta \cdot \nabla u - \epsilon \Delta u = 0 & \text{in } \Omega \\ u = g & \text{over } \partial\Omega, \end{cases} \quad (9)$$

with $\beta = (1, 0)$, $\epsilon = 0.1$, with g such that

$$\begin{aligned} g(0, x_2) &= \sin(\pi x_2) \text{ for } x_2 \in (0, 1), & g(1, x_2) &= 0 \text{ for } x_2 \in (0, 1) \\ g(x_1, 0) &= 0 \text{ for } x_1 \in (0, 1), & g(x_1, 1) &= 0 \text{ for } x_1 \in (0, 1) \end{aligned}$$

$$u_{\text{exact}}(x, y) = \frac{(e^{r_1(x-1)} - e^{r_2(x-1)})}{(e^{(-r_1)} - e^{(-r_2)})} \sin(\pi y), \quad (10)$$

$$r_1 = \frac{(1 + \sqrt{(1 + 4\epsilon^2\pi^2)})}{(2\epsilon)}, \quad r_2 = \frac{(1 - \sqrt{(1 + 4\epsilon^2\pi^2)})}{(2\epsilon)}.$$

Advection-diffusion problem

We define the shift u_{shift} such that

$$u(x_1, x_2) = u_0(x_1, x_2) + u_{shift}(x_1, x_2), \quad (11)$$

$$u_{shift}(x_1, x_2) = (1 - x_1) \sin(\pi x_2) \quad (12)$$

$u_0(x_1, x_2) = u(x_1, x_2) - u_{shift}(x_1, x_2) = 0$ for $(x_1, x_2) \in \partial\Omega$.

Using the shift we reformulate the problem with zero Dirichlet b.c.:

$$\begin{cases} \beta \cdot \nabla u_0 - \epsilon \Delta u_0 = -\beta \cdot \nabla u_{shift} + \epsilon \Delta u_{shift} & \text{in } \Omega \\ u = 0 & \text{over } \partial\Omega, \end{cases} \quad (13)$$

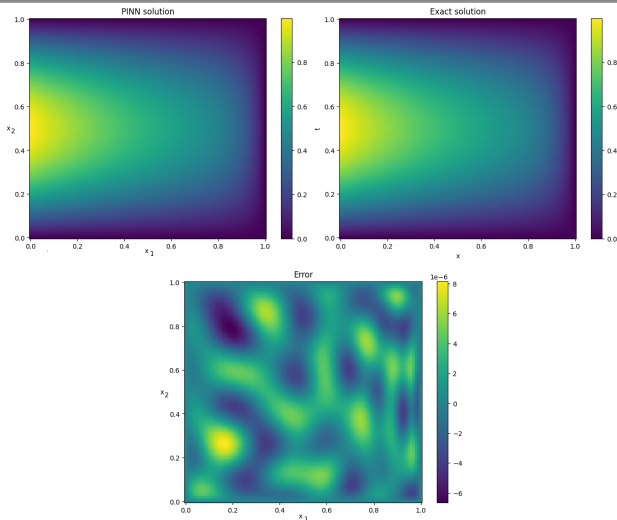
We define the following loss function for PINN

$$\begin{aligned} LOSS(\theta) = RES_3(\theta) = \\ \sum_{\mathbf{x}} (\beta \cdot \nabla u_0(\mathbf{x}) - \epsilon \Delta u_0(\mathbf{x}) + \beta \cdot \nabla u_{shift}(\mathbf{x}) - \epsilon \Delta u_{shift}(\mathbf{x}))^2 \end{aligned} \quad (14)$$

We enforce the zero Dirichlet b.c. on the NN in a strong way.

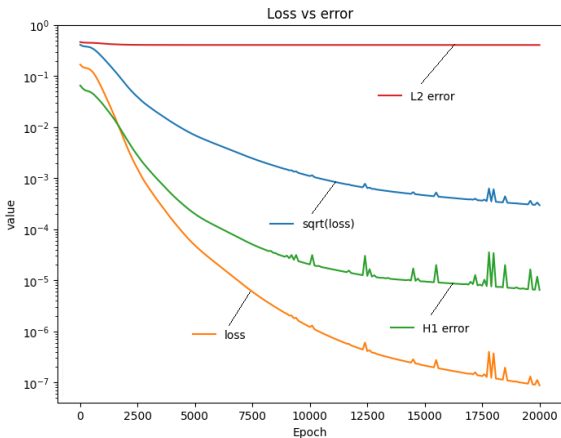
Physics Informed Neural Networks

Advection-diffusion problem



Physics Informed Neural Networks

Advection-diffusion problem



Poisson problem with sin-sin solution

Given $\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek the solution of

$$-\Delta u = f, \quad (15)$$

with zero Dirichlet b.c. In this problem we select the solution

$$u(x_1, x_2) = \sin(2\pi x_1) \sin(2\pi x_2). \quad (16)$$

We employ the manufactured solution technique

$$f_1(x_1, x_2) = -\Delta u(x_1, x_2) = 8\pi^2 \sin(2\pi x_1) \sin(2\pi x_2). \quad (17)$$

Robust Physics Informed Neural Networks

Poisson problem with sin-sin solution

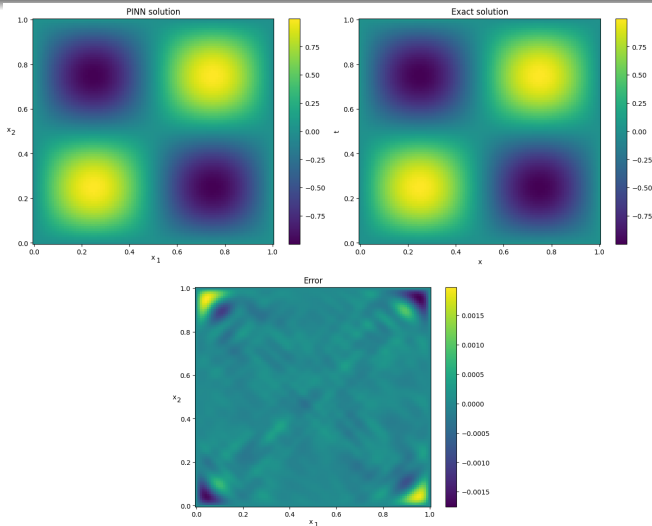
$$RES_1(\theta) = \{\Delta u_\theta(\mathbf{x}) + f_1(\mathbf{x})\}_x \quad (18)$$

$$G_{i,j;k,l} = \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j), (i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1), (i,j-1)\} \end{cases} \quad (19)$$

$$LOSS(\theta) = RES_1(\theta)^T G^{-1} RES_1(\theta) \quad (20)$$

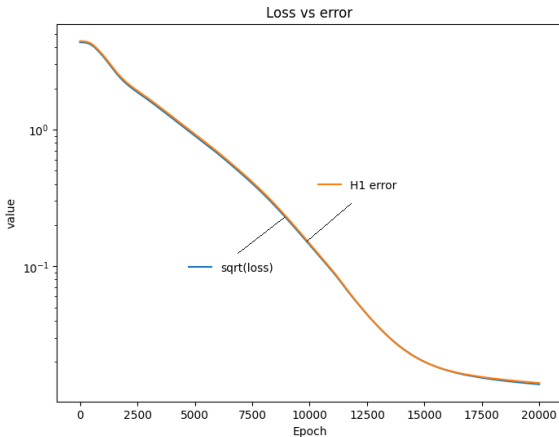
Robust Physics Informed Neural Networks

Poisson problem with sin-sin solution



Robust Physics Informed Neural Networks

Poisson problem with sin-sin solution



Poisson problem with sin-exp solution

Given $\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek the solution of

$$-\Delta u = f, \quad (21)$$

with zero Dirichlet b.c. In this problem we select the solution

$$u(x_1, x_2) = -e^{\pi(x_1-2x_2)} \sin(2\pi x_1) \sin(\pi x_2). \quad (22)$$

We employ the manufactured solution technique

$$\begin{aligned} f_2(x_1, x_2) &= -\Delta u(x_1, x_2) = \\ &= \pi^2 e^{\pi(x-2y)} \sin(\pi y) (4 \cos(2\pi x) - 3 \sin(2\pi x)) \\ &\quad - \pi^2 e^{\pi(x-2y)} \sin(2\pi x) (4 \cos(\pi y) - 3 \sin(\pi y)) \end{aligned} \quad (23)$$

Robust Physics Informed Neural Networks

Poisson problem with sin-exp solution

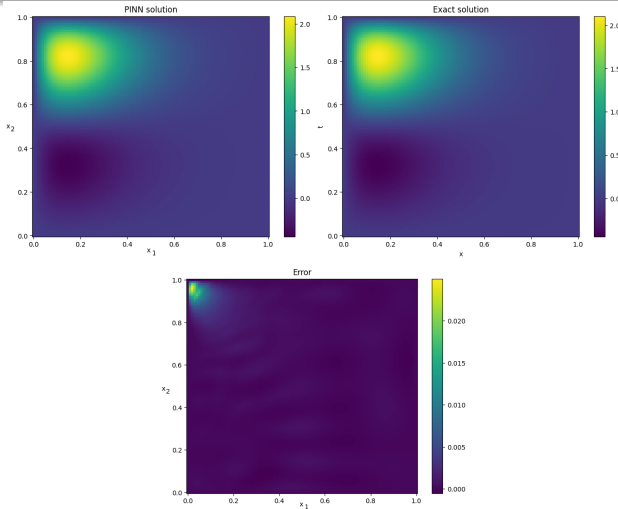
$$RES_2(\theta) = \{\Delta u_\theta(\mathbf{x}) + f_1(\mathbf{x})\}_x \quad (24)$$

$$G_{i,j;k,l} = \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j), (i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1), (i,j-1)\} \end{cases} \quad (25)$$

$$LOSS(\theta) = RES_2(\theta)^T G^{-1} RES_2(\theta) \quad (26)$$

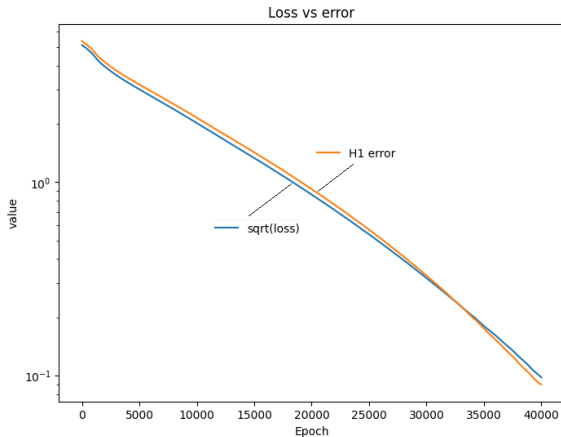
Robust Physics Informed Neural Networks

Poisson problem with sin-exp solution



Robust Physics Informed Neural Networks

Poisson problem with sin-exp solution



Advection-diffusion problem

$\Omega = (0, 1)^2 \subset \mathbb{R}^2$ we seek solution of the Eriksson-Johnson problem
 Kenneth Eriksson, Claes Johnson. *Adaptive finite element methods for parabolic problems I: A linear model problem*. **SIAM Journal of Numerical Analysis** 1991

$$\begin{cases} \beta \cdot \nabla u - \epsilon \Delta u = 0 & \text{in } \Omega \\ u = g & \text{over } \partial\Omega, \end{cases} \quad (27)$$

with $\beta = (1, 0)$, $\epsilon = 0.1$, with g such that

$$\begin{aligned} g(0, x_2) &= \sin(\pi x_2) \text{ for } x_2 \in (0, 1), & g(1, x_2) &= 0 \text{ for } x_2 \in (0, 1) \\ g(x_1, 0) &= 0 \text{ for } x_1 \in (0, 1), & g(x_1, 1) &= 0 \text{ for } x_1 \in (0, 1) \end{aligned}$$

$$u_{\text{exact}}(x, y) = \frac{(e^{r_1(x-1)} - e^{r_2(x-1)})}{(e^{(-r_1)} - e^{(-r_2)})} \sin(\pi y), \quad (28)$$

$$r_1 = \frac{(1 + \sqrt{(1 + 4\epsilon^2\pi^2)})}{(2\epsilon)}, \quad r_2 = \frac{(1 - \sqrt{(1 + 4\epsilon^2\pi^2)})}{(2\epsilon)}.$$

Advection-diffusion problem

We define the shift u_{shift} such that

$$u(x_1, x_2) = u_0(x_1, x_2) + u_{shift}(x_1, x_2), \quad (29)$$

$$u_{shift}(x_1, x_2) = (1 - x_1) \sin(\pi x_2) \quad (30)$$

$$u_0(x_1, x_2) = u(x_1, x_2) - u_{shift}(x_1, x_2) = 0 \text{ for } (x_1, x_2) \in \partial\Omega.$$

Using the shift we reformulate the problem with zero Dirichlet b.c.:

$$\begin{cases} \beta \cdot \nabla u_0 - \epsilon \Delta u_0 = -\beta \cdot \nabla u_{shift} + \epsilon \Delta u_{shift} & \text{in } \Omega \\ u = 0 & \text{over } \partial\Omega, \end{cases} \quad (31)$$

Robust Physics Informed Neural Networks

Advection-diffusion problem

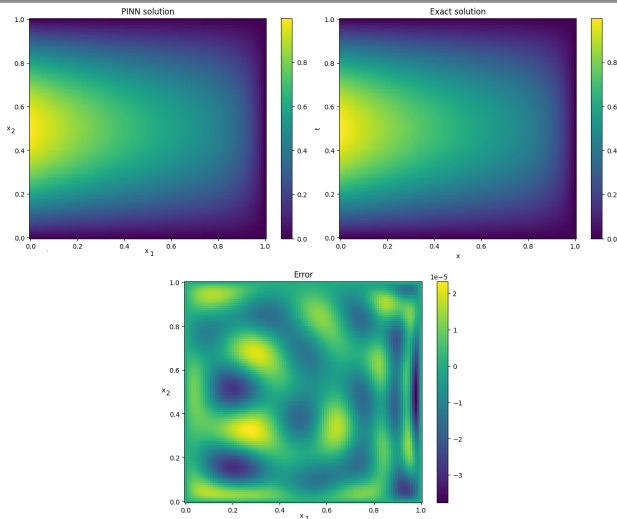
$$RES_3(\theta) = \{\beta \cdot \nabla u_0(\mathbf{x}) - \epsilon \Delta u_0(\mathbf{x}) + \beta \cdot \nabla u_{shift}(\mathbf{x}) - \epsilon \Delta u_{shift}(\mathbf{x})\}_{\mathbf{x}} \quad (32)$$

$$G_{i,j;k,l} = \epsilon \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j), (i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1), (i,j-1)\} \end{cases} \quad (33)$$

$$LOSS(\theta) = RES_3(\theta)^T G^{-1} RES_3(\theta) \quad (34)$$

Robust Physics Informed Neural Networks

Advection-diffusion problem



Robust Physics Informed Neural Networks

Advection-diffusion problem



Discrete domain based on training points

$\Omega = (0, 1)^2$, $h = \frac{1}{N}$ with spatial resolution $N \in \mathbb{N}$

The discrete domain

$$\Omega_h = \{(ih, jh) \in \mathbb{R}^2 : 0 \leq i \leq N, 0 \leq j \leq N\}. \quad (35)$$

and its interior $\Omega_h^0 = \Omega_h \setminus \partial\Omega$,

$$\Omega_h^0 = \{(ih, jh) \in \mathbb{R}^2 : 0 < i < N, 0 < j < N\}. \quad (36)$$

Remark 1. The points employed by the training of the PINN for evaluation of the PDE residual are selected from the Ω_h space, including the boundary points $\partial\Omega_h$.

Discrete L^2 norm and inner product

Space of discrete functions defined over Ω_h

$$L^2(\Omega_h) = \{u: \Omega_h \rightarrow \mathbb{R}\} \quad (37)$$

The inner product

$$(u, v)_{L^2(\Omega_h)} = h^2 \sum_{p \in \Omega_h} u(p)v(p). \quad (38)$$

The norm

$$\|u\|_{L^2(\Omega_h)}^2 = (u, u)_{L^2(\Omega_h)} = h^2 \sum_{p \in \Omega_h} u(p)^2. \quad (39)$$

We also define

$$L_0^2(\Omega_h) = \{\Omega_h \ni (x) \rightarrow u(x) \in \mathcal{R}, \sum_{p \in \Omega_h} u^2(p) < \infty; u(x) = 0, x \in \partial\Omega_h\}. \quad (40)$$

Discrete gradient operators

For $u: \Omega_h \rightarrow \mathbb{R}$ we denote

$$u_{i,j} = u(x_{i,j}) = u(ih, jh). \quad (41)$$

We introduce the discrete gradient operators

$$\nabla_+ u_{i,j} = (\nabla_{x+} u_{i,j}, \nabla_{y+} u_{i,j}) = \left(\frac{u_{i+1,j} - u_{i,j}}{h}, \frac{u_{i,j+1} - u_{i,j}}{h} \right), \quad (42)$$

$$\nabla_- u_{i,j} = (\nabla_{x-} u_{i,j}, \nabla_{y-} u_{i,j}) = \left(\frac{u_{i,j} - u_{i-1,j}}{h}, \frac{u_{i,j} - u_{i,j-1}}{h} \right). \quad (43)$$

We define them equal to 0 in the points outside the boundary.

Discrete H_0^1 inner product and norm

$$H_0^1(\Omega_h) = \{\Omega_h \ni x_{i,j} \rightarrow u_{i,j} \in \mathcal{R},$$
$$u \in L_0^2(\Omega_h), \nabla_{x+} u \in L_0^2(\Omega_h), \nabla_{y+} u \in (L_0^2(\Omega_h))\}. \quad (44)$$

$$(u, v)_{H_0^1(\Omega_h)} = (\nabla_{x+} u, \nabla_{x+} v)_{L^2(\Omega_h)} + (\nabla_{y+} u, \nabla_{y+} v)_{L^2(\Omega_h)} \quad (45)$$

$$= \sum_{\substack{0 \leq i < N \\ 0 \leq j \leq N}} (u_{i+1,j} - u_{i,j})(v_{i+1,j} - v_{i,j}) + \quad (46)$$

$$+ \sum_{\substack{0 \leq i \leq N \\ 0 \leq j < N}} (u_{i,j+1} - u_{i,j})(v_{i,j+1} - v_{i,j}). \quad (47)$$

$$\|u\|_{H_0^1(\Omega_h)}^2 = (u, u)_{H_0^1(\Omega_h)}. \quad (48)$$

Discrete test functions

We introduce discrete test functions $\Omega_h \ni x \mapsto \delta_{i,j}(x) \in \mathbb{R}$

$$\delta_{i,j}(x) = \begin{cases} 1 & \text{for } x = x_{i,j} \\ 0 & \text{otherwise} \end{cases}. \quad (49)$$

Then $\mathcal{B} = \{\delta_{i,j}\}_{0 < i,j < N}$ constitutes a basis of $H_0^1(\Omega_h)$.

Remark 2. In PINN method the discrete test functions represent the discrete Dirac deltas of the training points.

Gram matrix derivation

We derive the formula for the Gram matrix using the inner product.

The Gram matrix $G_{i,j;k,l} = (\delta_{i,j}, \delta_{k,l})_{H_0^1(\Omega_h)} =$

$$\begin{aligned}
 &= \sum_{\substack{0 \leq m < N \\ 0 \leq n \leq N}} (\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n}))(\delta_{k,l}(x_{m+1,n}) - \delta_{k,l}(x_{m,n})) + \\
 &\quad \sum_{\substack{0 \leq m \leq N \\ 0 \leq n < N}} (\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n}))(\delta_{k,l}(x_{m,n+1}) - \delta_{k,l}(x_{m,n}))
 \end{aligned}$$

Gram matrix derivation

For $(i, j) = (k, l)$ we have $G_{i,j;k,l} =$

$$= \sum_{\substack{0 \leq m < N \\ 0 \leq n \leq N}} (\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n})) (\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n})) +$$

	$(\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n}))$	$(\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n}))$
$(m,n) == (i,j)$	-1	-1
$(m+1,n) == (i,j)$	1	1
$(m,n+1) == (i,j)$	0	0

$$+ \sum_{\substack{0 \leq m \leq N \\ 0 \leq n < N}} (\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n})) (\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n})) = 4$$

	$(\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n}))$	$(\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n}))$
$(m,n) == (i,j)$	-1	-1
$(m+1,n) == (i,j)$	0	0
$(m,n+1) == (i,j)$	1	1

Gram matrix derivation

For $(i+1, j) = (k, l)$ we have $G_{i,j;k,l} =$

$$= \sum_{\substack{0 \leq m < N \\ 0 \leq n \leq N}} (\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n})) (\delta_{i+1,j}(x_{m+1,n}) - \delta_{i+1,j}(x_{m,n})) +$$

	$(\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n}))$	$(\delta_{i+1,j}(x_{m+1,n}) - \delta_{i+1,j}(x_{m,n}))$
$(m,n) == (i,j)$	-1	1
$(m+1,n) == (i,j)$	1	0
$(m,n+1) == (i,j)$	0	0

$$+ \sum_{\substack{0 \leq m \leq N \\ 0 \leq n < N}} (\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n})) (\delta_{i+1,j}(x_{m,n+1}) - \delta_{i+1,j}(x_{m,n})) = -1$$

	$(\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n}))$	$(\delta_{i+1,j}(x_{m,n+1}) - \delta_{i+1,j}(x_{m,n}))$
$(m,n) == (i,j)$	-1	0
$(m+1,n) == (i,j)$	0	0
$(m,n+1) == (i,j)$	1	0

Gram matrix derivation

$$G_{i,j;k,l} = \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j), (i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1), (i,j-1)\} \end{cases} \quad (50)$$

Remark 3. The inner product selected for the Gram matrix is induced by the norm for which the form $b(u, v)$ of the discrete weak form of the PDE is a bounded inf-sup stable.

Discrete weak formulation

Our advection-diffusion problem can be reformulated in the discrete weak form as:

Find $u \in H_0^1(\Omega_h)$ such that

$$b(u, v) = l(v) \quad \forall v \in H_0^1(\Omega_h) \quad (51)$$

$$b(u, v) = (\beta_x \nabla_{x+} u, v)_{L^2(\Omega_h)} + (\beta_y \nabla_{y+} u, v)_{L^2(\Omega_h)} + \epsilon (\nabla_+ u, \nabla_+ v)_{L^2(\Omega_h)}, \quad (52)$$
$$l(v) = (f, v)_{L^2(\Omega_h)}.$$

The residual of the advection-diffusion problem is

$$r(u, v) = b(u, v) - l(v).$$

We can form a vector of residuals $RES(\theta) = \{r(u_\theta, \delta_{i,j})\}_{0 < i,j < N}$.

Discrete Poincaré

Lemma (Discrete Poincaré)

There exists a constant $C > 0$ independent of N , such that for all $u \in H_0^1(\Omega_h)$ we have

$$\|u\|_{L^2(\Omega_h)} \leq C \|u\|_{H_0^1(\Omega_h)} \quad (53)$$

Following [Marcin Łoś, Maciej Paszyński, *Robust Physics Informed Neural Networks* [arXiv:2401.02300v2](https://arxiv.org/abs/2401.02300) (2024)]

$$\|u\|_{L^2(\Omega_h)} \leq 2 \|u\|_{H_0^1(\Omega_h)} \quad (54)$$

independently of the mesh size N .

Discrete boundedness

Lemma (Discrete boundedness)

The form $b(u, v)$ is a bounded bilinear form in the norm $\|u\|_{H_0^1(\Omega_h)}$,
This means that there exists μ independent of N such that

$$b(u, v) \leq \mu \|u\|_{H_0^1(\Omega_h)} \|v\|_{H_0^1(\Omega_h)}, \forall u, v \in H_0^1(\Omega_h)$$

Following [Marcin Łoś, Maciej Paszyński, *Robust Physics Informed Neural Networks* [arXiv:2401.02300v2](https://arxiv.org/abs/2401.02300v2) (2024)]

$$b(u, v) \leq \underbrace{(\epsilon + 2C\|\beta\|_\infty)}_{\mu > 0} \|u\|_{H_0^1(\Omega_h)} \|v\|_{H_0^1(\Omega_h)}$$

Here $C > 0$ is the constant from discrete Poincare Lemma,
 ϵ is the diffusion constant, $\beta = (\beta_x, \beta_y)$, $\|\beta\|_\infty = \max\{|\beta_x|, |\beta_y|\}$.

Discrete inf-sup

Lemma (Discrete inf-sup)

The form $b(u, v)$ is inf-sup stable bilinear form in the norm $\|u\|_{H_0^1(\Omega_h)}$, if $2\epsilon > MC^2$.

Here $C > 0$ is the constant from discrete Poincare Lemma, and $M = \max_{i,j} |\nabla_{x+\beta_x} + \nabla_{y+\beta_y}|$.

$$\sup_{0 \neq v \in V} \frac{b(u, v)}{\|v\|_{H_0^1(\Omega_h)}} \geq \alpha \|u\|_{H_0^1(\Omega_h)}, \forall u \in H_0^1(\Omega_h)$$

Following [Marcin Łoś, Maciej Paszyński, *Robust Physics Informed Neural Networks* [arXiv:2401.02300v2](https://arxiv.org/abs/2401.02300) (2024)]

$$b(u, u) \geq \underbrace{\left(\epsilon - \frac{1}{2}MC^2 \right)}_{\alpha > 0} \|u\|_{H_0^1(\Omega_h)}^2$$

Relation between robust loss and true error

Theorem

Let $u \in H_0^1(\Omega_h)$ and ϕ be its residual representative. Then

$$\frac{1}{\mu} \|\phi\|_{H_0^1(\Omega_h)} \leq \|u - u_{EXACT}\|_{H_0^1(\Omega_h)} \leq \frac{1}{\alpha} \|\phi\|_{H_0^1(\Omega_h)}$$

where μ , α are the boundedness and inf-sup constants of b .

The general idea of this proof is based on the similar considerations for the continuous case of Robust Variational Physics Informed Neural Networks introduced in

[Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **arXiv:2308.16910** (2023)]

Relation between robust loss and true error

Remark 5. For a neural network solution u_θ , the norm for which the form $b(u, v)$ of the discrete weak formulation of the PDE is a bounded inf-sup stable, for u_{EXACT} the exact solution of the discrete weak formulation, and for the robust loss function $LOSS(\theta)$

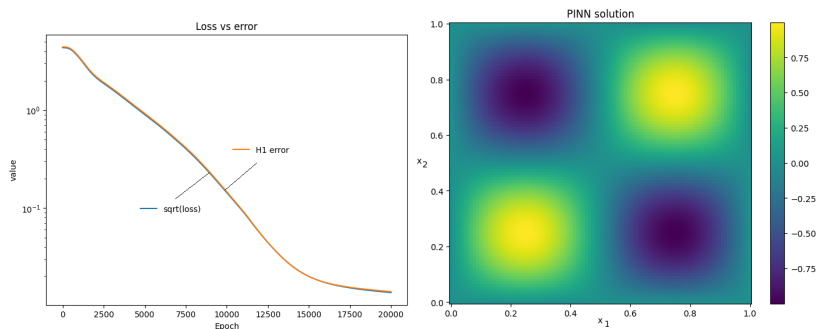
$$LOSS(\theta) = RES^T(\theta) \times \mathbf{G}^{-1} \times RES(\theta) \quad (55)$$

it holds

$$\frac{\sqrt{LOSS(\theta)}}{\mu} \leq \|u_\theta - u_{EXACT}\|_{H_0^1(\Omega_h)} \leq \frac{\sqrt{LOSS(\theta)}}{\alpha} \quad (56)$$

For the Laplace problems, $\mu = \alpha = 1$.

Poisson problem with sin-sin solution

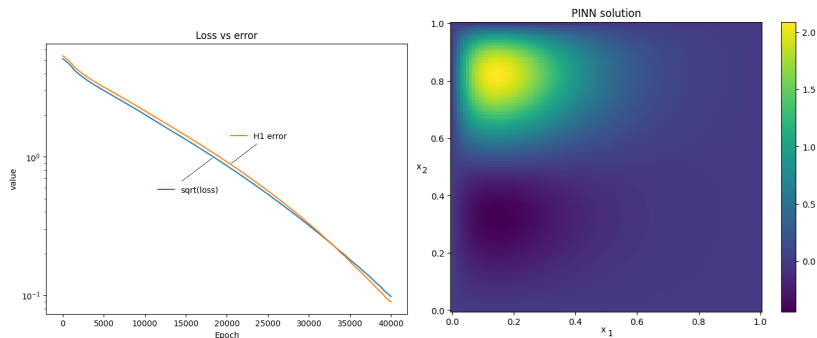


We can see that our loss is robust and equal to the true error computed in $H_0^1(\Omega_h)$ norm.

This is because for the Laplace problem we have $\mu = \alpha = 1$ so

$$\sqrt{\text{LOSS}(\theta)} = \sqrt{\text{RES}^T(\theta) \times \mathbf{G}^{-1} \times \text{RES}(\theta)} = \|u_{\text{EXACT}} - u_\theta\|_{H_0^1(\Omega_h)}.$$

Poisson problem with sin-exp solution

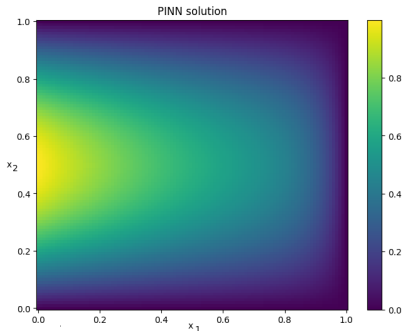


We can see that our loss is robust and equal to the true error computed in $H_0^1(\Omega_h)$ norm.

This is because for the Laplace problem we have $\mu = \alpha = 1$ so

$$\sqrt{LOSS(\theta)} = \sqrt{RES^T(\theta) \times \mathbf{G}^{-1} \times RES(\theta)} = \|u_{EXACT} - u_\theta\|_{H_0^1(\Omega_h)}.$$

Advection-diffusion problem



$\mu = (\epsilon + 2C) = (0.1 + 2 \times 2) = 4.1$, and $\alpha = \epsilon = 0.1$. So we have

$$\frac{1}{4.1} \sqrt{LOSS(\theta)} \leq \|u_{EXACT} - u_{\theta}\|_{H_0^1(\Omega_h)} \leq \frac{1}{0.1} \sqrt{LOSS(\theta)}.$$

Multiplying by $\epsilon = 0.1$, we have

$$\frac{1}{41} \sqrt{LOSS(\theta)} \leq 0.1 \times \|u_{EXACT} - u_{\theta}\|_{H_0^1(\Omega_h)} \leq \sqrt{LOSS(\theta)}.$$

This implies plots agreement if we measure the error in $\epsilon H_0^1(\Omega_h)$ norm.

The robust loss function and the true error are close to each other.

Conclusions

We numerically show the robustness of our loss function.

The construction of the robust loss function can be summarized as:

- We select PDE, e.g. Poisson or the advection-diffusion problem, and we derive its discrete weak formulation
- We seek the inner product for which the form $b(u_{i,j}, v_{i,j})$ of the discrete weak formulation is a bounded inf-sup stable bilinear form in the induced norm. In our case

$$(u, v)_{H_0^1(\Omega_h)} = (\nabla_{x+} u, \nabla_{x+} v)_{L^2(\Omega_h)} + (\nabla_{y+} u, \nabla_{y+} v)_{L^2(\Omega_h)}.$$

- We select the test functions corresponding with the points selected for training $\{\delta_{i,j}(x)\}_{i,j}$.
- We compute the Gram matrix $G_{i,j;k,l} = (\delta_{i,j}, \delta_{k,l})_{H_0^1(\Omega_h)}$.
- We invert the sparse Gram matrix.
- The robust loss function is defined as

$$LOSS(\theta) = RES(\theta)^T \mathbf{G}^{-1} RES(\theta)$$

References

- [1] Marcin Łoś, Maciej Paszyński, *Robust Physics Informed Neural Networks*, **arXiv:2401.02300v2** (2024)
- [2] Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszynski, *Robust Variational Physics-Informed Neural Networks*, **arXiv:2308.16910** (2023)
- [3] Dongho Shin, John Stikwerda, *Inf-sup conditions for finite differences approximations of the Stokes equations*, **Journal of Australian Mathematical Society** (1997)
- [4] Paweł Maczuga, Maciej Skoczeń, Przemysław Rożnawski, Filip Tłuszcz, Marcin Szubert, Marcin Łoś, Witold Dzwiniel, Keshav Pingali, Maciej Paszyński, *Physics Informed Neural Network Code for 2D Transient Problems (PINN-2DT) Compatible with Google Colab*, **arXiv:2310.03755** (2023)