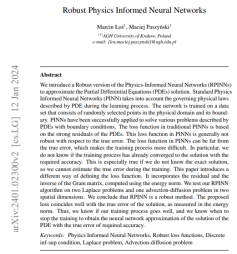
# Robust Physics Informed Neural Networks

**Maciej Paszyński**


Faculty of Computer Science
**AGH University of Krakow**


Robust Physics Informed Neural Networks

Marcin Łoś[1], Maciej Paszyński[1]

[1]AGH University of Krakow, Poland
e-mail: {los,maciej.paszynski}@agh.edu.pl

arXiv:2401.02300v2 [cs.LG] 12 Jan 2024

Abstract

We introduce a Robust version of the Physics-Informed Neural Networks (RPINNs) to approximate the Partial Differential Equations (PDEs) solution. Standard Physics Informed Neural Networks (PINN) takes into account the governing physical laws described by PDE during the learning process. The network is trained on a data set that consists of randomly selected points in the physical domain and its boundary. PINNs have been successfully applied to solve various problems described by PDEs with boundary conditions. The loss function in traditional PINNs is based on the strong residuals of the PDEs. This loss function in PINNs is generally not robust with respect to the true error. The loss function in PINNs can be far from the true error, which makes the training process more difficult. In particular, we do not know if the training process has already converged to the solution with the required accuracy. This is especially true if we do not know the exact solution, so we cannot estimate the true error during the training. This paper introduces a different way of defining the loss function. It incorporates the residual and the inverse of the Gram matrix, computed using the energy norm. We test our RPINN algorithm on two Laplace problems and one advection-diffusion problem in two spatial dimensions. We conclude that RPINN is a robust method. The proposed loss coincides well with the true error of the solution, as measured in the energy norm. Thus, we know if our training process goes well, and we know when to stop the training to obtain the neural network approximation of the solution of the PDE with the true error of required accuracy.

Keywords: Physics Informed Neural Networks, Robust loss functions, Discrete inf-sup condition, Laplace problem, Advection-diffusion problem

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 di erences 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 Dzwinel, 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 $\Omega = (0,1)^2 \quad \mathrm{R}^2$ we seek the solution of

$$-\Delta u = f, \tag{1}$$

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

$$u(x_1, x_2) = sin(2\pi x_1) sin(2\pi x_2). \tag{2}$$

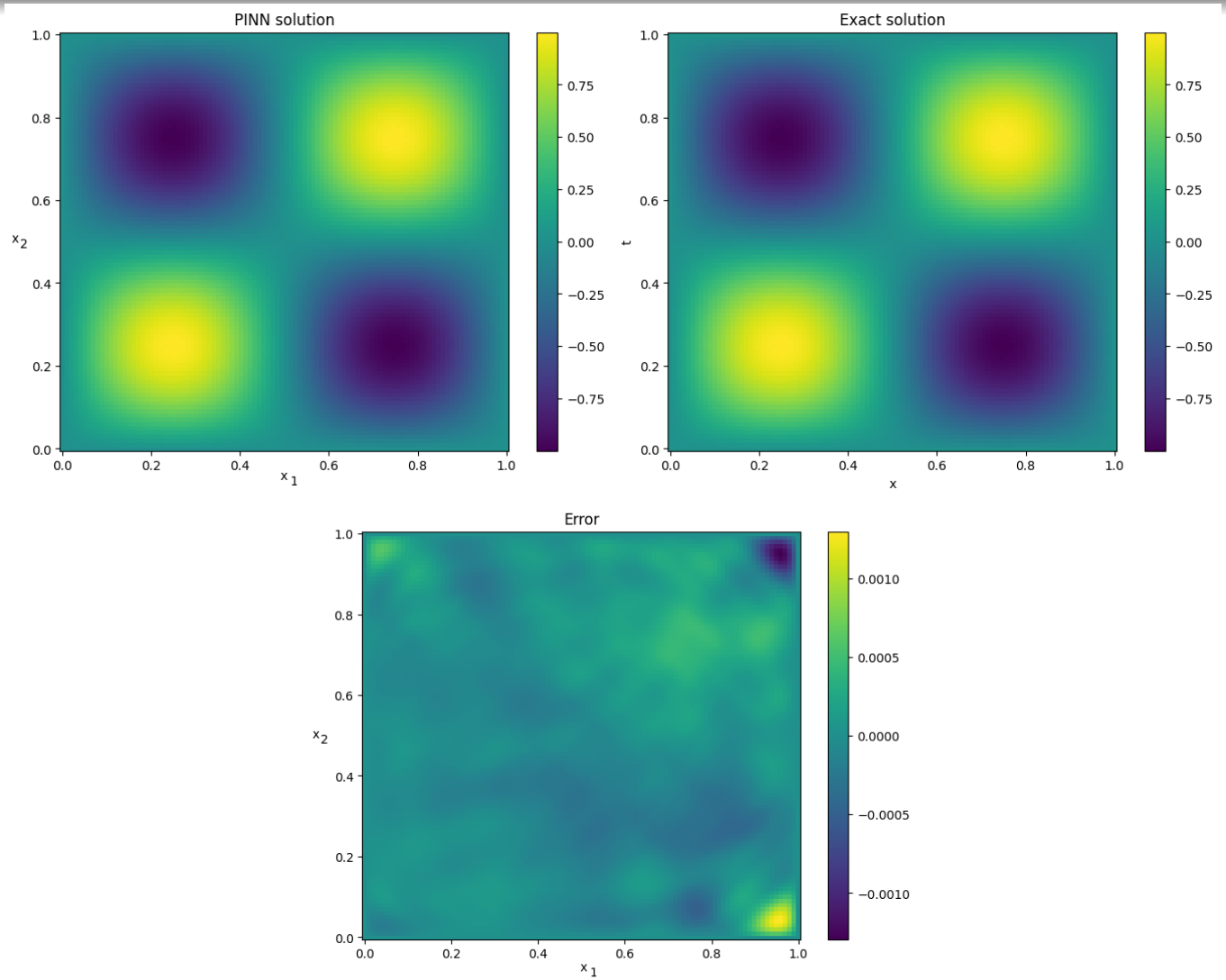
We employ the manufactured solution technique

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

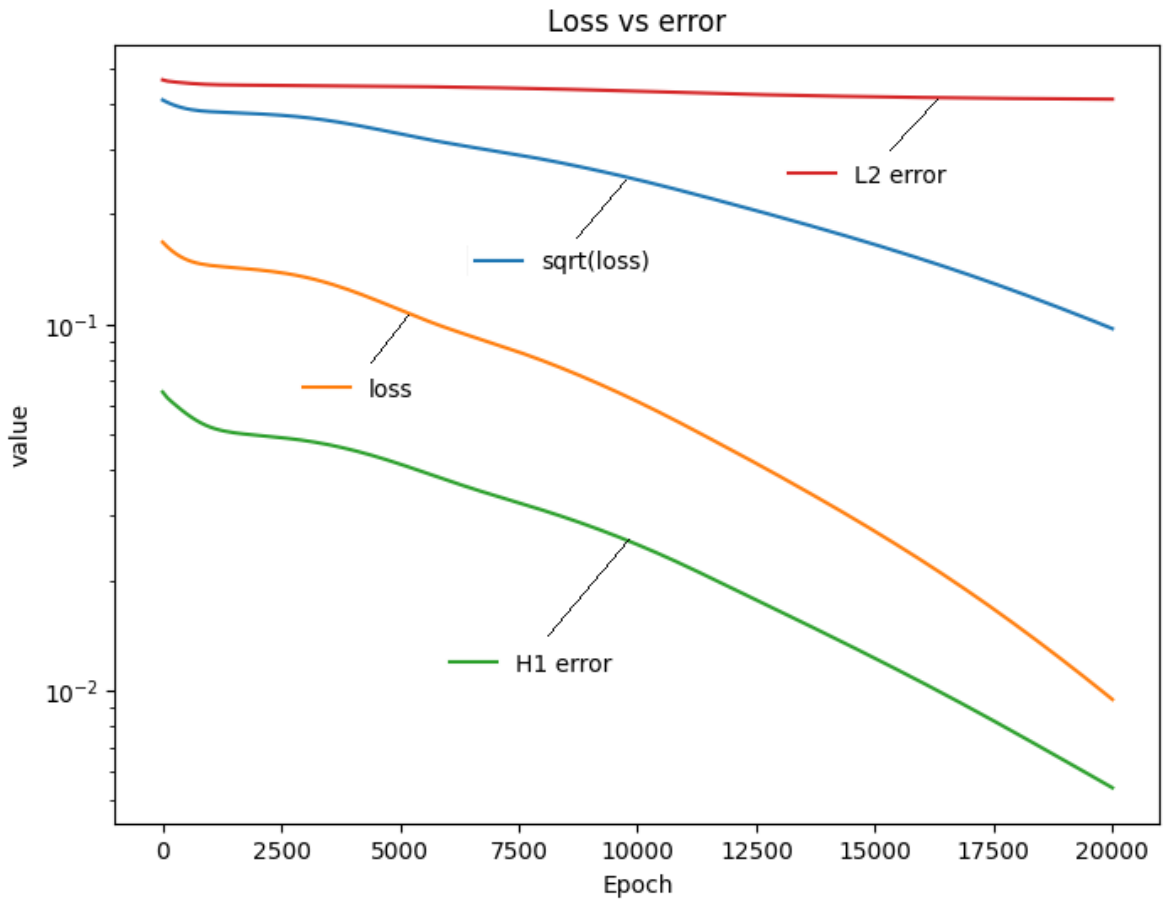
We define the following loss function for PINN

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

# 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 $\Omega = (0,1)^2 \quad \mathrm{R}^2$ we seek the solution of

$$-\Delta u = f, \tag{5}$$

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

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

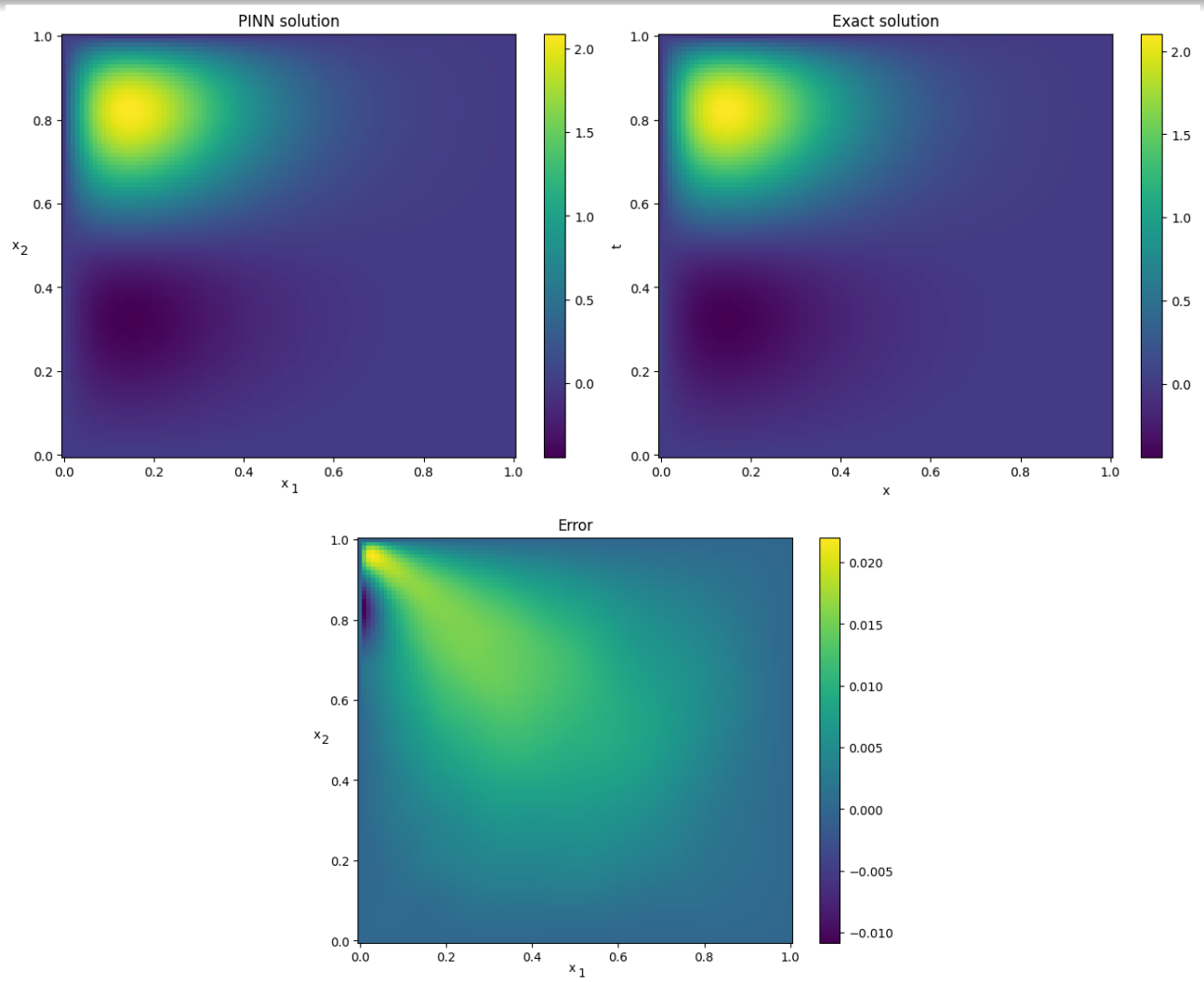
We employ the manufactured solution technique

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

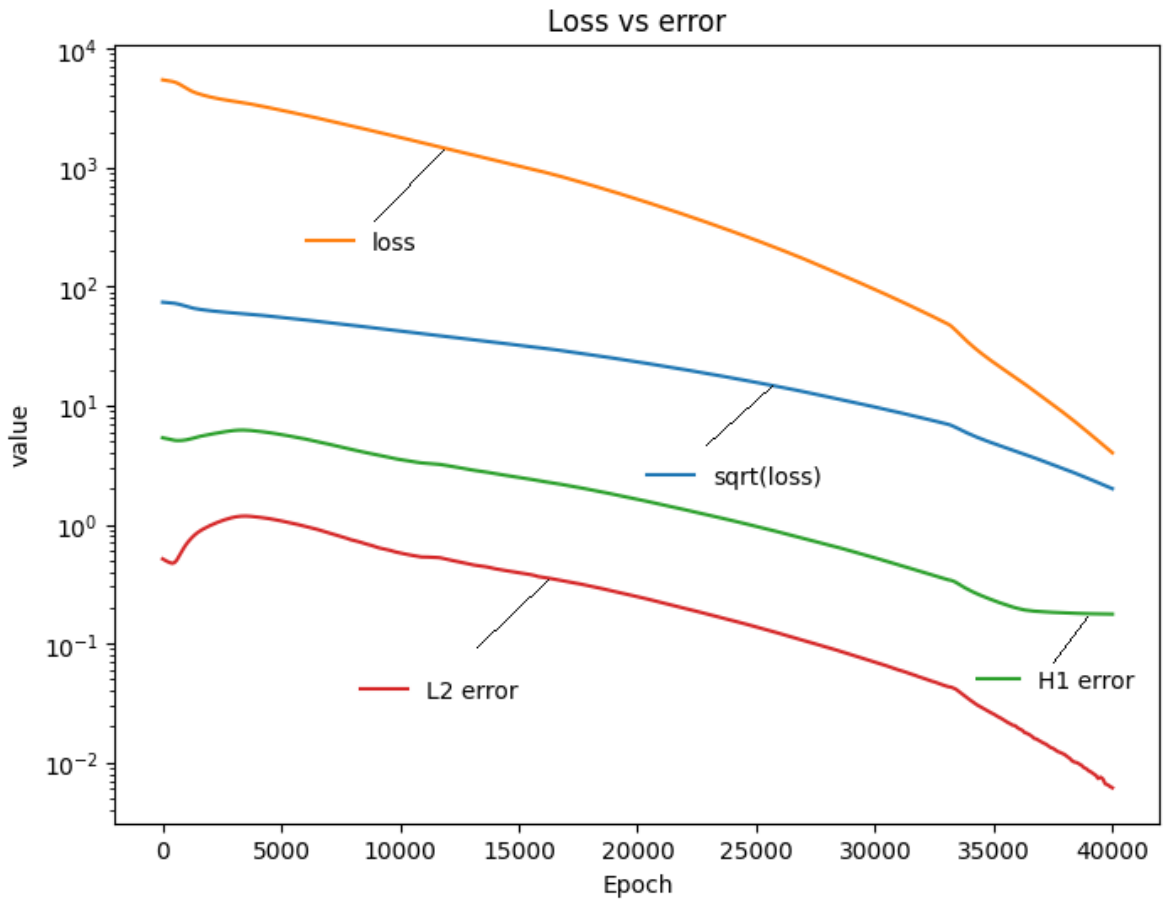
We define the following loss function for PINN

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

# Physics Informed Neural Networks
# Poisson problem with sin-exp solution



# Physics Informed Neural Networks
# Poisson problem with sin-exp solution



# Advection-diffusion problem

$\Omega = (0,1)^2$ $\mathrm{R}^2$ 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

$$\begin{aligned} \beta \cdot \quad u - \quad \Delta u &= 0 \quad \text{in } \Omega \\ u &= g \quad \text{over } \partial\Omega, \end{aligned} \tag{9}$$

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

$$g(0,x_2) = \sin(\pi x_2) \text{ for } x_2 \quad (0,1), \quad g(1,x_2) = 0 \text{ for } x_2 \quad (0,1)$$
$$g(x_1,0) = 0 \text{ for } x_1 \quad (0,1), \quad g(x_1,1) = 0 \text{ for } x_1 \quad (0,1)$$

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

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

# Advection-diffusion problem

We define the shift $u_{shift}$ such that

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

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

$u_0(x_1, x_2) = u(x_1, x_2) - u_{shift}(x_1, x_2) = 0$ for $(x_1, x_2)$ $\partial\Omega$.
Using the shift we reformulate the problem with zero Dirichlet b.c.:

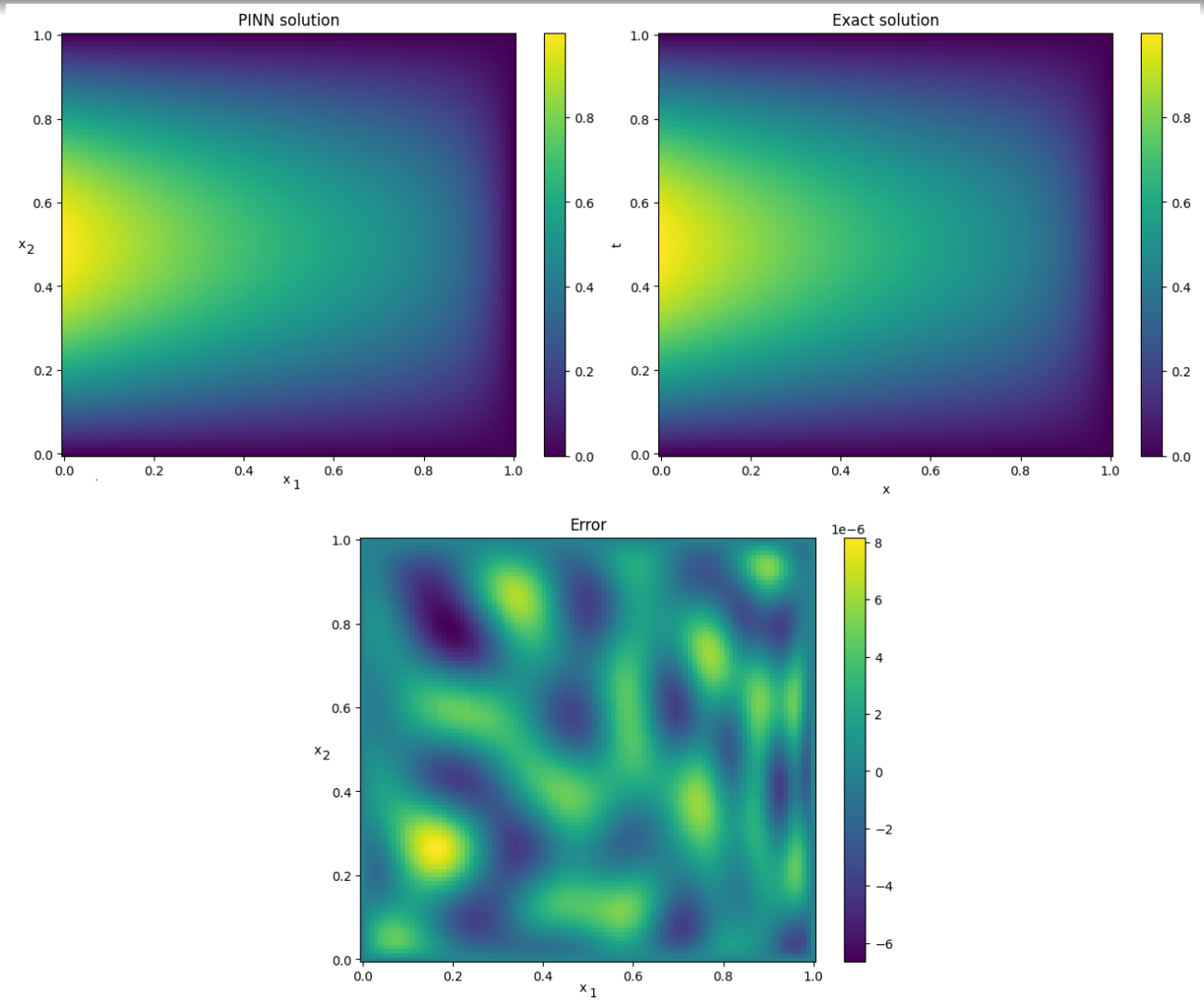
$$\begin{array}{rl} \beta \cdot u_0 - \Delta u_0 = -\beta \cdot u_{shift} + \Delta u_{shift} & \text{in } \Omega \\ u = 0 & \text{over } \partial\Omega, \end{array} \tag{13}$$

We define the following loss function for PINN

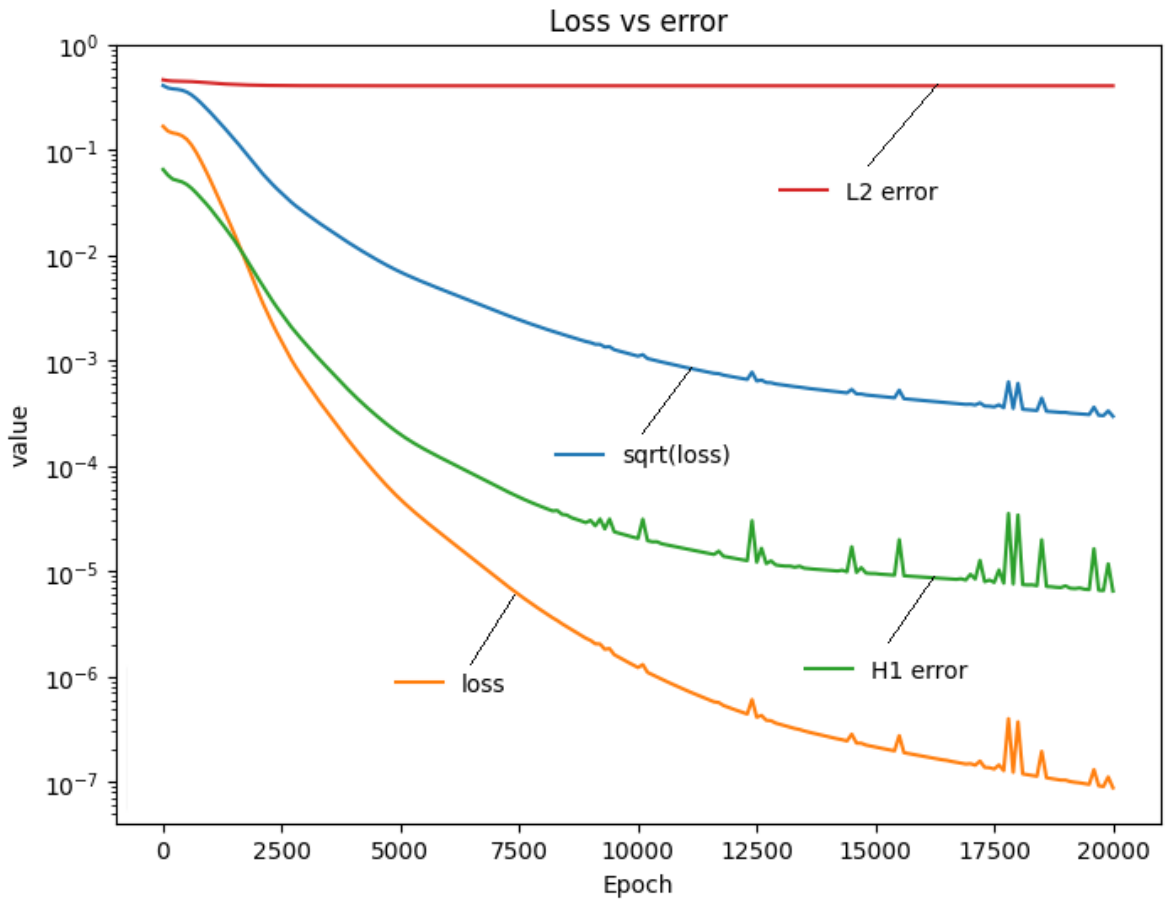
$$LOSS(\theta) = RES_3(\theta) = \sum_{\mathbf{x}} (\beta \cdot u_0(\mathbf{x}) - \Delta u_0(\mathbf{x}) + \beta \cdot u_{shift}(\mathbf{x}) - \Delta u_{shift}(\mathbf{x}))^2 \tag{14}$$

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 $\Omega = (0,1)^2 \quad \mathrm{R}^2$ we seek the solution of

$$-\Delta u = f, \tag{15}$$

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

$$u(x_1, x_2) = sin(2\pi x_1) sin(2\pi x_2). \tag{16}$$

We employ the manufactured solution technique

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

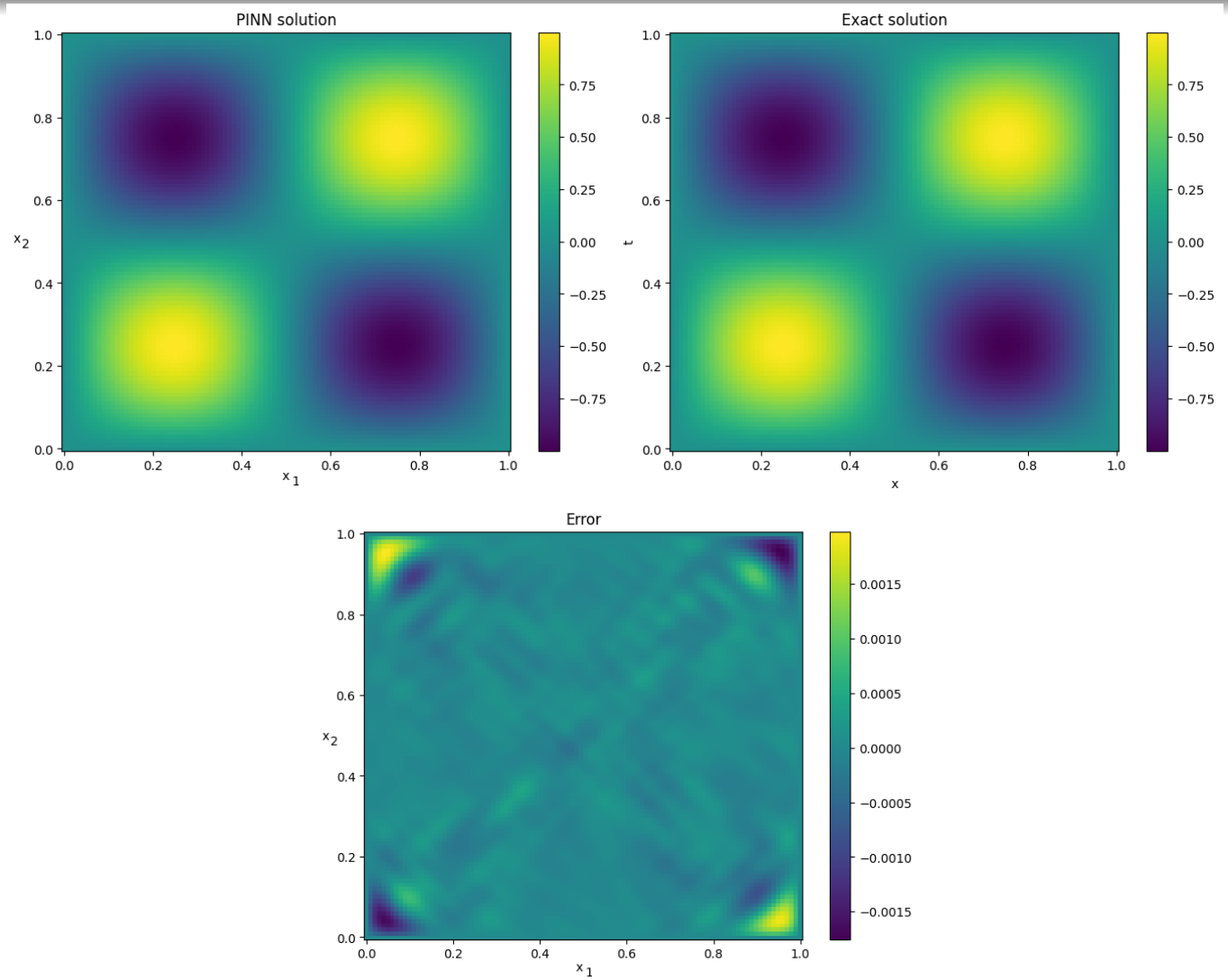
# Robust Physics Informed Neural Networks
## Poisson problem with sin-sin solution

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

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

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

# 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 $\Omega = (0,1)^2 \quad \mathrm{R}^2$ we seek the solution of

$$-\Delta u = f, \tag{21}$$

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

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

We employ the manufactured solution technique

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

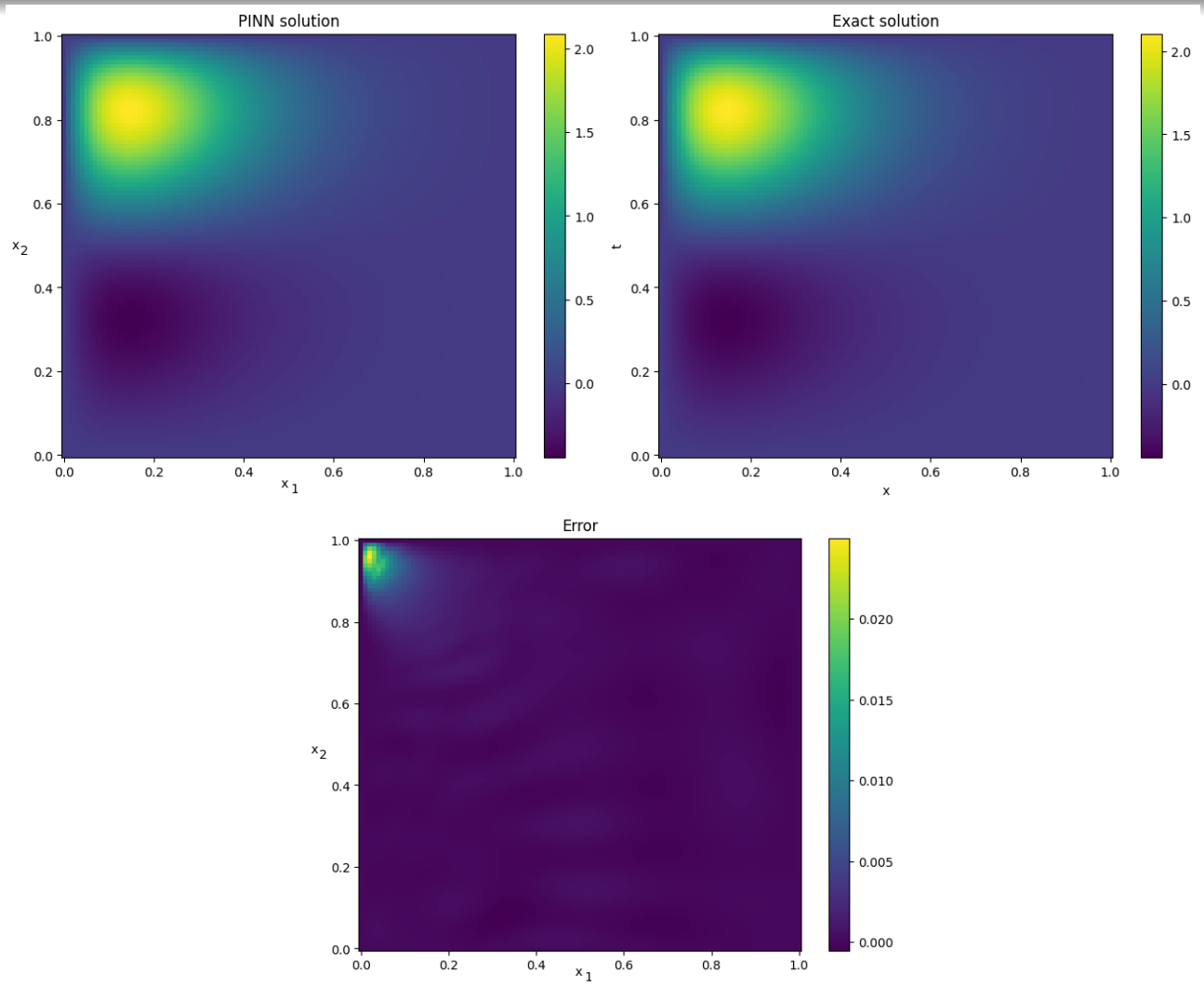
# Robust Physics Informed Neural Networks
## Poisson problem with sin-exp solution

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

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

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

# Robust Physics Informed Neural Networks
# Poisson problem with sin-exp solution



# Robust Physics Informed Neural Networks
# Poisson problem with sin-exp solution

# Advection-di usion problem

$= (\ 0; 1)^2$ $R^2$ we seek solution of the Eriksson-Johnson problem

Kenneth Eriksson, Claes Johnson. Adaptive nite element methods for parabolic problems I: A linear model problem. SIAM Journal of Numerical Analysis 1991

$$\begin{cases} r\ u & u = 0 \quad \text{in} \\ & u = g \quad \text{over} @\ ; \end{cases} \tag{27}$$

with $= (\ 1; 0), \quad = 0{:}1$, with $g$ such that

$$g(0; x_2) = \sin(\ x_2) \text{ for } x_2 \ 2\ (0; 1); \quad g(1; x_2) = 0 \text{ for } x_2 \ 2\ (0; 1)$$

$$g(x_1; 0) = 0 \text{ for } x_1 \ 2\ (0; 1); \quad g(x_1; 1) = 0 \text{ for } x_1 \ 2\ (0; 1)$$

$$u_{exact}(x; y) = \frac{(e^{(r_1(x\ \ 1))} \quad e^{(r_2(x\ \ 1))})}{(e^{(\ r_1)} \quad e^{(\ r_2)})} \sin(\ y); \tag{28}$$

$$r_1 = \frac{(1 + \quad \sqrt{(1 + 4\ ^2\ ^2)})}{(2\ )}; r_2 = \frac{(1 \quad \sqrt{(1 + 4\ ^2\ ^2)})}{(2\ )}:$$

# Advection-di usion problem

We de ne the shift $u_{shift}$ such that

$$u(x_1; x_2) = u_0(x_1; x_2) + u_{shift}(x_1; x_2); \tag{29}$$

$$u_{shift}(x_1; x_2) = ( 1 \quad x_1) \sin( \quad x_2) \tag{30}$$

$u_0(x_1; x_2) = u(x_1; x_2) \quad u_{shift}(x_1; x_2) = 0$ for $(x_1; x_2)$ 2 @ .
Using the shift we reformulate the problem with zero Dirichlet b.c.:

$$\begin{cases} r\ u_0 \qquad u_0 = \qquad r\ u_{shift} + \quad u_{shift} & \text{in} \\ u = 0 \quad \text{over} @ \ ; \end{cases} \tag{31}$$

# Robust Physics Informed Neural Networks
# Advection-di usion problem

$$RES_3(\ ) = f \quad r\ u_0(x) \qquad u_0(x) + \qquad r\ u_{shift}(x) \qquad u_{shift}(x) g_x \tag{32}$$

$$G_{i;j;k;l} = \begin{cases} 4 & \text{for } (i;j) = (k;l) \\ 1 & \text{for } (k;l) \in f(i+1;j);(i\ \ 1;j)g \\ 1 & \text{for } (k;l) \in f(i;j+1);(i;j\ \ 1)g \end{cases} \tag{33}$$

$$LOSS(\ ) = RES_3(\ )^T G^{\ 1} RES_3(\ ) \tag{34}$$

# Robust Physics Informed Neural Networks
# Advection-di usion problem

# Robust Physics Informed Neural Networks
# Advection-di usion problem

# Discrete domain based on training points

$= (\ 0; 1)^2$, $h = \frac{1}{N}$ with spatial resolution $N\ 2\ N$
The discrete domain

$$_h = f(ih; jh)\ 2\ R^2 : 0 \quad i \quad N; 0 \quad j \quad Ng: \tag{35}$$

and its interior $\overset{0}{h} = \ _h n @$ ,

$$\overset{0}{_h} = f(ih; jh)g\ 2\ R^2 : 0 < i < N; 0 < j < Ng: \tag{36}$$

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

# Discrete $L^2$ norm and inner product

Space of discrete functions de ned over $_h$

$$L^2(\ _h) = f u : \ _h \ ! \ \ R g \tag{37}$$

The inner product

$$(u; v)_{L^2(\ _h)} = h^2 \sum_{p2\ _h} u(p)v(p): \tag{38}$$

The norm

$$kuk^2_{L^2(\ _h)} = (u; u)_{L^2(\ _h)} = h^2 \sum_{p2\ _h} u(p)^2: \tag{39}$$

We also de ne

$$\begin{aligned} X \ & L^2_0(\ _h) = f \ _h 3 \ (x) \ ! \ \ u(x) \ 2 \ R \ ; \\ & u^2(p) < 1 \ ; u(x) = 0; x \ 2 \ @ \ _h g: \end{aligned} \tag{40}$$
$$p2\ _h$$

# Discrete gradient operators

For $u: \quad_h ! \quad R$ we denote

$$u_{i;j} = u(x_{i;j}) = u(ih; jh): \tag{41}$$

We introduce the discrete gradient operators

$$r \ _+ u_{i;j} = (r \ _{x+} u_{i;j}; r \ _{y+} u_{i;j}) = \frac{u_{i+1;j} \quad u_{i;j}}{h}; \frac{u_{i;j+1} \quad u_{i;j}}{h} \ ; \tag{42}$$

$$r \quad u_{i;j} = (r \ _x \ u_{i;j}; r \ _y \ u_{i;j}) = \frac{u_{i;j} \quad u_{i \ 1;j}}{h}; \frac{u_{i;j} \quad u_{i;j \ 1}}{h} \ : \tag{43}$$

We de ne them equal to 0 in the points outside the boundary.

# Discrete $H_0^1$ inner product and norm

$$H_0^1(\ _h) = f\ _h\ 3\ x_{i;j}\ !\ \ u_{i;j}\ 2\ R\ ;\quad u\ 2\ L_0^2(\ _h);\ r\ _{x+}u\ 2\ L_0^2(\ _h);\ r\ _{y+}u\ 2\ (L_0^2(\ _h)g: \tag{44}$$

$$(u;v)_{H_0^1(\ _h)} = (\ r\ _{x+}u;r\ _{x+}v)_{L^2(\ _h)} + (\ r\ _{y+}u;r\ _{y+}v)_{L^2(\ _h)} \tag{45}$$

$$= \sum (u_{i+1;j}\ \ u_{i;j})(v_{i+1;j}\ \ v_{i;j}) + \tag{46}$$

$$0\ \ i<N,\ 0\ \ j\ \ N$$

$$+ \sum (u_{i;j+1}\ \ u_{i;j})(v_{i;j+1}\ \ v_{i;j}): \tag{47}$$

$$0\ \ i\ \ N,\ 0\ \ j<N$$

$$kuk^2_{H_0^1(\ _h)} = (\ u;u)_{H_0^1(\ _h)}: \tag{48}$$

# Discrete test functions

We introduce discrete test functions $_h$ 3 x 7! $_{i;j}(x)$ 2 R

$$_{i;j}(x) = \begin{cases} 1 & \text{for } x = x_{i;j} \\ 0 & \text{otherwise} \end{cases} : \tag{49}$$

Then B = f $_{i;j}$ g$_{0<i;j<N}$ constitutes a basis of H$_0^1$( $_h$).

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 $G_{i;j;k;l} = (\ _{i;j};\ _{k;l})_{H_0^1(\ _h)} =$

$$= \sum_{\substack{0 \le m < N \\ 0 \le n \le N}} (\ _{i;j}(x_{m+1;n}) - \ _{i;j}(x_{m;n}))(\ _{k;l}(x_{m+1;n}) - \ _{k;l}(x_{m;n})) +$$
$$\sum_{\substack{0 \le m \le N \\ 0 \le n < N}} (\ _{i;j}(x_{m;n+1}) - \ _{i;j}(x_{m;n}))(\ _{k;l}(x_{m;n+1}) - \ _{k;l}(x_{m;n}))$$

# Gram matrix derivation

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

$$= \sum_{\substack{0 \le m < N \\ 0 \le n \le N}} \left( \frac{\partial}{\partial_{i;j}}(x_{m+1;n}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right)\left( \frac{\partial}{\partial_{i;j}}(x_{m+1;n}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right) +$$

| | $\left( \frac{\partial}{\partial_{i;j}}(x_{m+1;n}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right)$ | $\left( \frac{\partial}{\partial_{i;j}}(x_{m+1;n}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right)$ |
|---|---|---|
| (m,n)==(i,j) | -1 | -1 |
| (m+1,n)==(i,j) | 1 | 1 |
| (m,n+1)==(i,j) | 0 | 0 |

$$+ \sum_{\substack{0 \le m \le N \\ 0 \le n < N}} \left( \frac{\partial}{\partial_{i;j}}(x_{m;n+1}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right)\left( \frac{\partial}{\partial_{i;j}}(x_{m;n+1}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right) = 4$$

| | $\left( \frac{\partial}{\partial_{i;j}}(x_{m;n+1}) - \frac{\partial}{\partial_{i;j}}(x_{m;n}) \right)$ | $\left( \frac{\partial}{\partial_{i;j}}(x_{m;n+1}) - \frac{\partial}{\partial_{j;j}}(x_{m;n}) \right)$ |
|---|---|---|
| (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 $G_{i,j;k,l} =$

$$= \square \quad \underset{\substack{0\le m<N \\ 0\le n\le N}}{} (\delta_{i,j}(x_{m+1,n}) - \delta_{i,j}(x_{m,n}))(\delta_{i+1,j}(x_{m+1,n}) - \delta_{i+1,j}(x_{m,n})) +$$

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

$$+ \square \quad \underset{\substack{0\le m\le N \\ 0\le n<N}}{} (\delta_{i,j}(x_{m,n+1}) - \delta_{i,j}(x_{m,n}))(\delta_{i+1,j}(x_{m,n+1}) - \delta_{i+1,j}(x_{m,n})) = -1$$

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

# Gram matrix derivation

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

**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.

# Disrete weak formulation

Our advection-diffusion problem can be reformulated in the discrete weak form as:
Find $u \quad H_0^1(\Omega_h)$ such that

$$b(u,v) = l(v) \qquad v \quad H_0^1(\Omega_h) \tag{51}$$

$$b(u,v) = (\beta_x \quad {}_{x+}u, v)_{L^2(\Omega_h)} + \tag{52}$$

$$(\beta_y \quad {}_{y+}u, v)_{L^2(\Omega_h)} + \ ( \quad {}_{+}u, \quad {}_{+}v)_{L^2(\Omega_h)},$$

$$l(v) = (f, v)_{L^2(\Omega_h)}.$$

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 $u \quad H_0^1(\Omega_h)$ we have*

$$u_{L^2(\Omega_h)} \leq C \; u_{H_0^1(\Omega_h)} \tag{53}$$

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

$$u_{L^2(\Omega_h)} \leq 2 \; u_{H_0^1(\Omega_h)} \tag{54}$$

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 $\mu$ independent of $N$ such that*

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

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

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

Here $C > 0$ is the constant from discrete Poincare Lemma,
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 $2\alpha > MC^2$.*
*Here $C > 0$ is the constant from discrete Poincare Lemma, and $M = \max_{i,j} |\alpha_{x+}\beta_x + \alpha_{y+}\beta_y|$.*

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

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

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

# Relation between robust loss and true error

**Theorem**

*Let $u \quad H_0^1(\Omega_h)$ and $\varphi$ be its residual representative. Then*

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

*where $\mu$, $\alpha$ 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 Paszynski, *Robust Variational Physics-Informed Neural Networks*, **arXiv:2308.16910** (2023) ]

# Relation between robust loss and true error

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

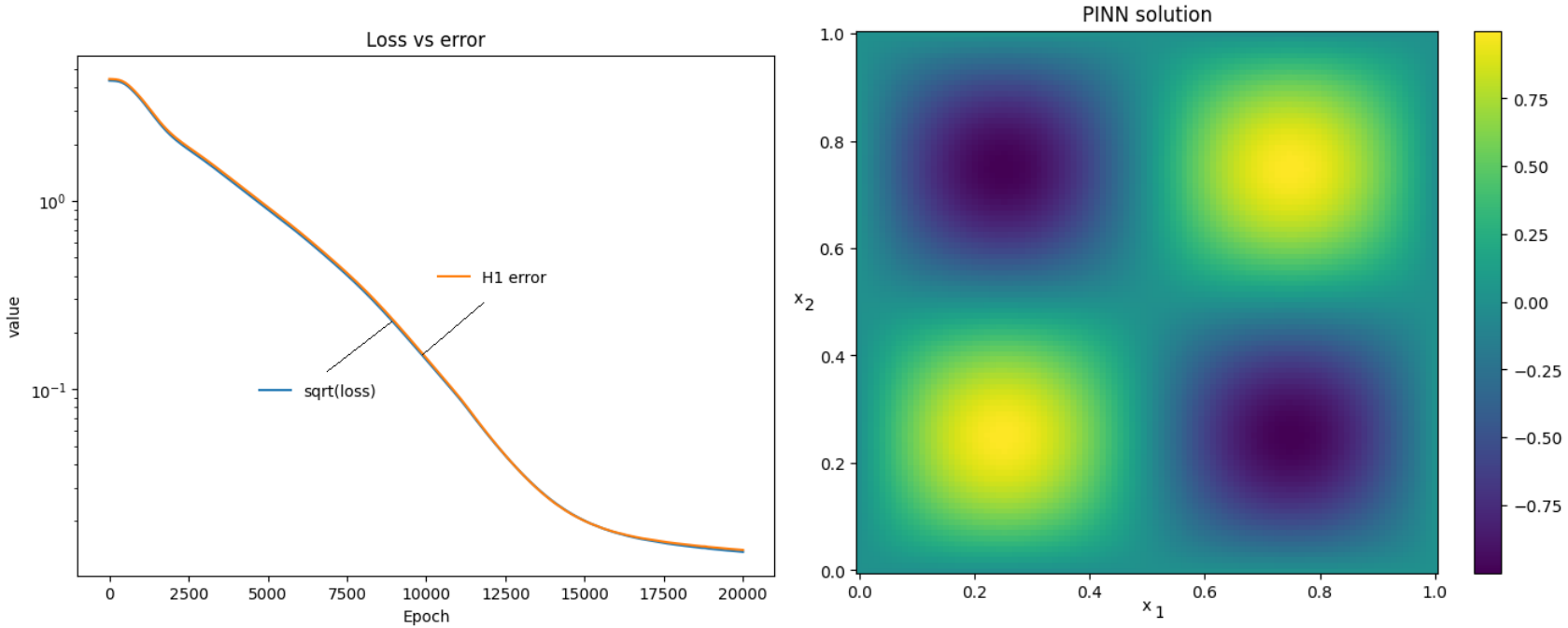
$$LOSS(\theta) = RES^T(\theta) \times \mathbf{G}^{-1} \times RES(\theta) \tag{55}$$

it holds

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

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

# Poisson problem with sin-sin solution

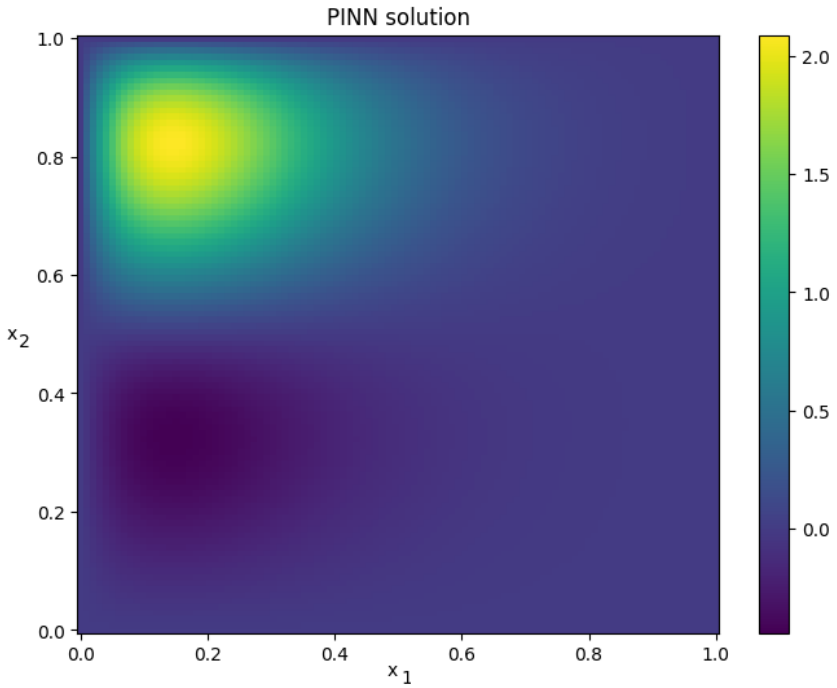



We can see that our loss is robust and equal to the true error computed in $H_0^1(\Omega_h)$ norm.

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

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

# 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

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

# Advection-diffusion problem

$\mu = \square(\ + 2C) = (0.1 + 2 \times 2) = 4.1$, and $\alpha = \square\ = 0.1$. So we have $\frac{1}{4.1}\ \square\overline{LOSS(\theta)} \leq \ u_{EXACT} - u_\theta\ _{H_0^1(\Omega_h)} \leq \frac{1}{0.1}\ \square\overline{LOSS(\theta)}$.
Multiplying by $\ = 0.1$, we have
$\frac{1}{41}\ \square\overline{LOSS(\theta)} \leq 0.1 \times \ u_{EXACT} - u_\theta\ _{H_0^1(\Omega_h)} \leq \ \square\overline{LOSS(\theta)}$. This implies plots agreement if we measure the error in $H_0^1(\Omega_h)$ 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 $(u,v)_{H_0^1(\Omega_h)} = (\partial_x u, \partial_x v)_{L^2(\Omega_h)} + (\partial_y u, \partial_y v)_{L^2(\Omega_h)}$.
- 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 di erences 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 Dzwinel, Keshav Pingali, Maciej Paszyński, *Physics Informed Neural Network Code for 2D Transient Problems (PINN-2DT) Compatible with Google Colab*, **arXiv:2310.03755** (2023)