Collocation-based Robust Variational Physics Informed Neural Networks (CRVPINN)

Maciej Paszyński

Faculty of Computer Science
AGH University of Krakow

maciej.paszynski@agh.edu.pl
home.agh.edu.pl/paszynsk (without "i")

April 4, 2025

Collaborators




Marcin Łoś[1], Tomasz Służalec[1], Paweł Maczuga[1], Magdalena Pabisz[1]




Keshav Pingali[4],Askold Vilkha[1],Anna Paszyńska[2], Mateusz Dobija[2]




Carlos Uriarte[3], Judit Munoz-Matute[3,4], Sergio Rojas[5], David Pardo[3,6]

[1]AGH University of Science and Technology, Kraków, Poland,
[2]Jagiellonian University, Kraków, Poland
[3]University of The Basque Country, Bilbao, Spain,
[4]Oden Institute, The University of Texas at Austin, USA
[5]Pontifical Catholic University of Valparaiso, Chile
[6]Basque Center for Applied Mathematics, Bilbao, Spain

Outline

- How to solve PDEs with Physics Informed Neural Networks
- PINN loss is not robust[1]
- Mathematical tools to make the loss robust again[2]
- How to define robust loss
- Gallery of numerical examples[3]
- Conclusions

---

[1] Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024)

[2] Marcin łoś, Tomasz Służalec, Paweł Maczuga, Askold Vilkha, Carlos Uriarte, Maciej Paszyński, *Collocation-based Robust Variational Physics-Informed Neural Networks (CRVPINN)* **https://arxiv.org/abs/2401.02300** (2024) (submitted to **Computers & Structures**)

[3] 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)

Solving PDEs with Physics Informed Neural Networks

M. Raissi, P. Perdikaris, G.E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, **Journal of Computational Physics**, 378 (2019) 686-707

6865 citations in 5 years

Poisson problem with sin-exp solution

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

$$-\Delta u = f_1,$$

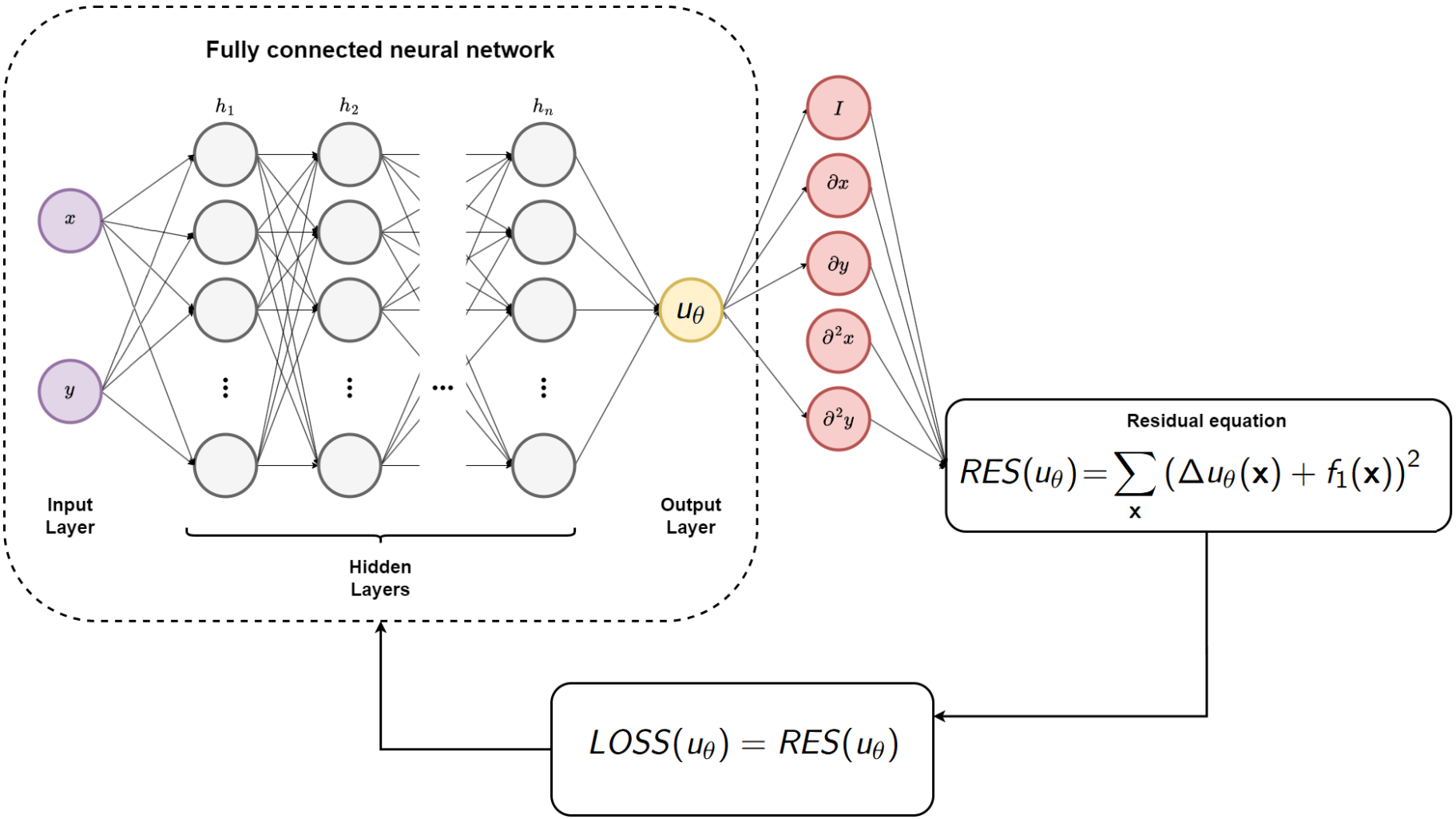
with zero Dirichlet b.c. In this problem we select the solution

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

We employ the manufactured solution technique

$$\begin{aligned} f_1(x_1, x_2) = -\Delta u(x_1, x_2) = \\ = \pi^2 e^{\pi(x-2y)} \sin(\pi y)(4\cos(2\pi x) - 3\sin(2\pi x)) \\ -\pi^2 e^{\pi(x-2y)} \sin(2\pi x)(4\cos(\pi y) - 3\sin(\pi y)) \end{aligned}$$

Solving PDEs with Neural Networks



$$u_\theta(x, y) = \mathcal{A}_4\sigma\left(\mathcal{A}_3\sigma\left(\mathcal{A}_2\sigma\left(\mathcal{A}_1\{x\ y\} + b_1\right) + b_2\right) + b_3\right) + b_4$$

$$\mathcal{A}_1 \in \mathcal{M}^{1\times 100};\ \mathcal{A}_2, \mathcal{A}_3 \in \mathcal{M}^{100\times 100};\ \mathcal{A}_4 \in \mathcal{M}^{100\times 2}$$

Programming of PINN loss

```
def residual_loss
(pinn:PINN,x:torch.Tensor,y:torch.tensor):

loss = f(pinn, x, y)
     + dfdx(pinn, x, y, order=2)
     + dfdy(pinn, x, y, order=2)

loss_val = loss.pow(2).sum() return loss_val
```

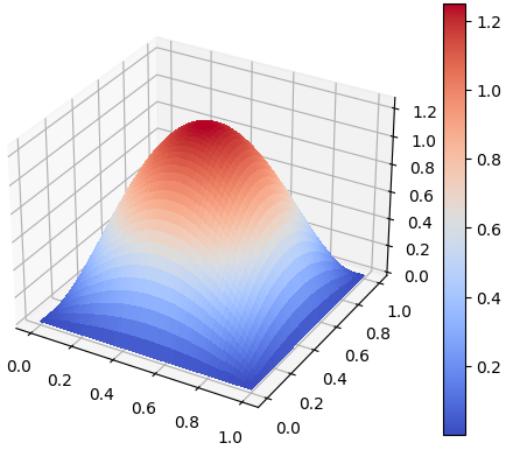
We define the following loss function for PINN

$$LOSS(u_\theta) = RES(u_\theta) = \sum_{x,y} \left(\frac{\partial^2 u_\theta(x,y)}{\partial x^2} + \frac{\partial^2 u_\theta(x,y)}{\partial y^2} + f_1(x,y) \right)^2$$

$$= \sum_{x,y} \left(\frac{\partial^2 PINN(x,y)}{\partial x^2} + \frac{\partial^2 PINN(x,y)}{\partial y^2} + f_1(x,y) \right)^2$$

$$u_\theta(x,y) \approx PINN(x,y) = \mathcal{A}_4 \sigma \left(\mathcal{A}_3 \sigma \left(\mathcal{A}_2 \sigma \left(\mathcal{A}_1 \begin{bmatrix} x \\ y \end{bmatrix} + b_1 \right) + b_2 \right) + b_3 \right)$$

Enforcing Dirichlet b.c. strongly




```
def zero_dirichlet(x, y): return 20*x*(1-x)*y*(1-y)

def dirichlet_bc(logits, x, y):
return logits*zero_dirichlet(x,y)
```

The Dirichlet boundary condition is enforced as a constraint

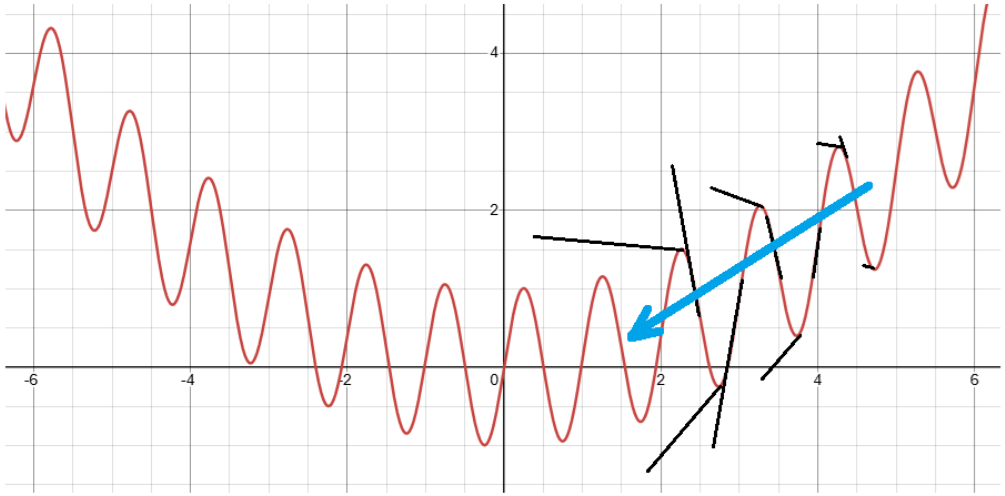
Training of neural network by SDG or ADAM

- Select points $(x, y) \in \Omega$ randomly
- Correct the weights using the loss derivatives

$$A^k_{i,j} = A^k_{i,j} - \eta \frac{\partial LOSS(x,y)}{\partial A^k_{i,j}} = A^k_{i,j} - \eta \frac{\partial \left(\Delta PINN(x,y) + f_1(x,y)\right)^2}{\partial A^k_{i,j}}$$

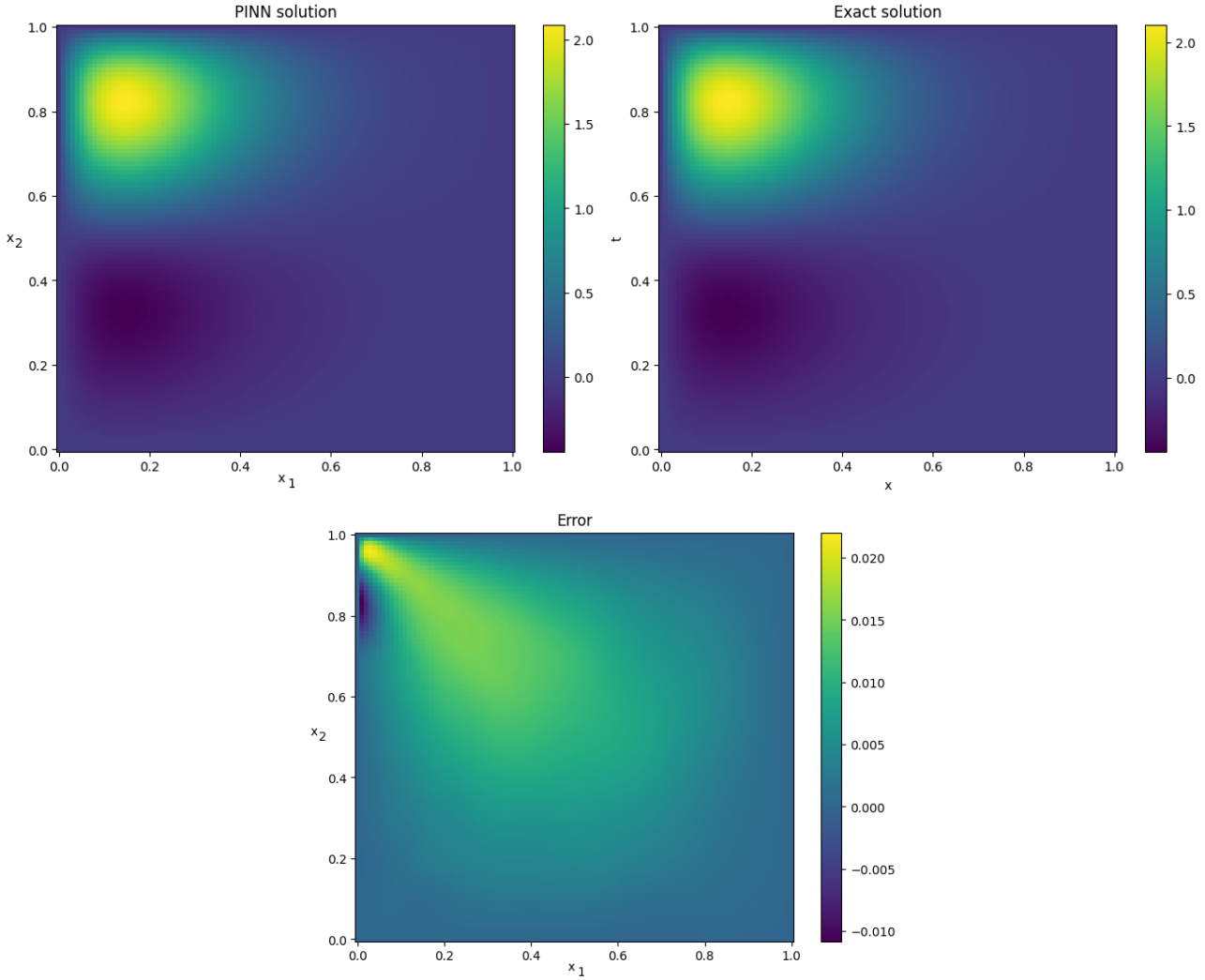
Repeat until $LOSS(x, y) \leq \delta$

Stochastic Gradient Descent method is replaced by ADAM method



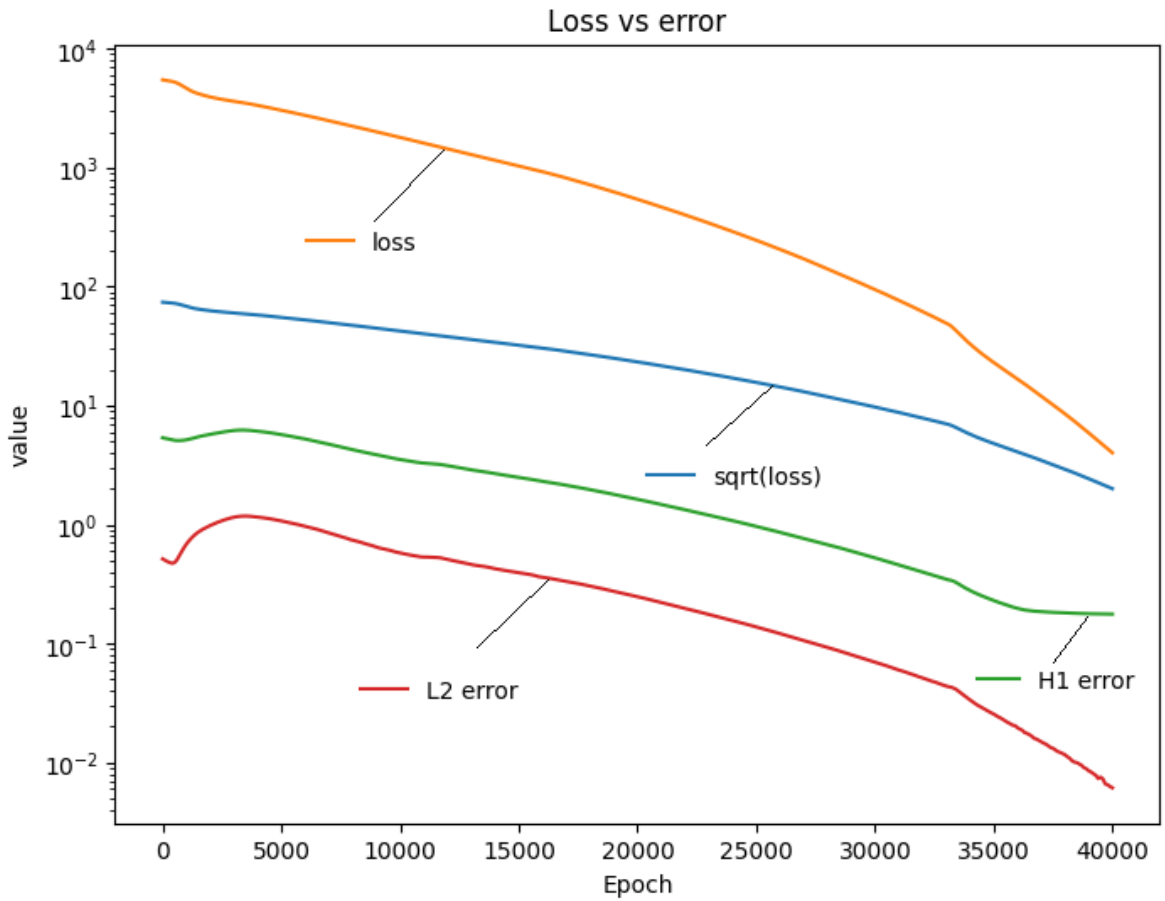
D. P. Kingma, J. Lei Ba, ADAM: A method for stochastic optimization, **arXiv:1412.6980** (2014)

Physics Informed Neural Networks
Poisson problem with sin-exp solution



We solve PDE by training neural network

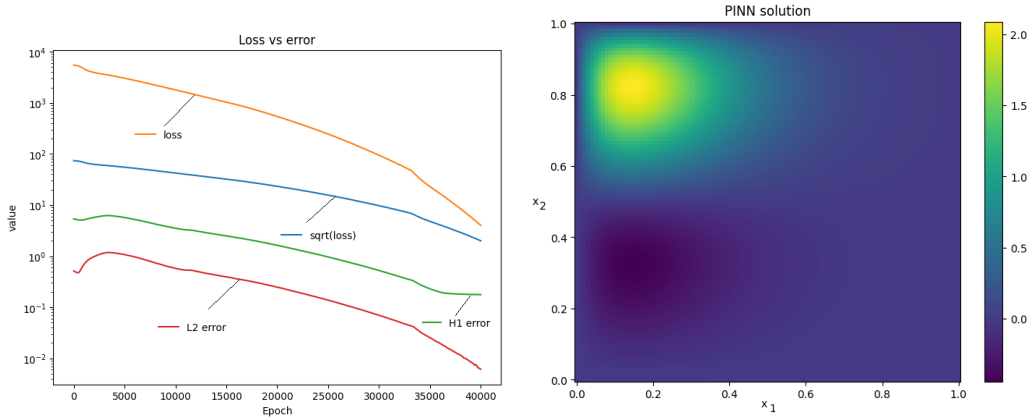
Physics Informed Neural Networks
Poisson problem with sin-exp solution



The PINN loss is not robust

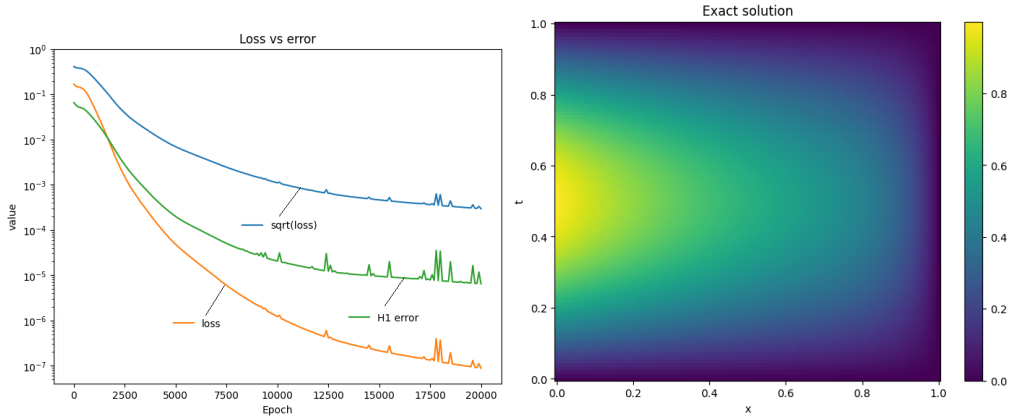
It is not equal to the true error $\|u_\theta - u_{EXACT}\|_{H^1(\Omega)}$

Unrobust losses of PINN



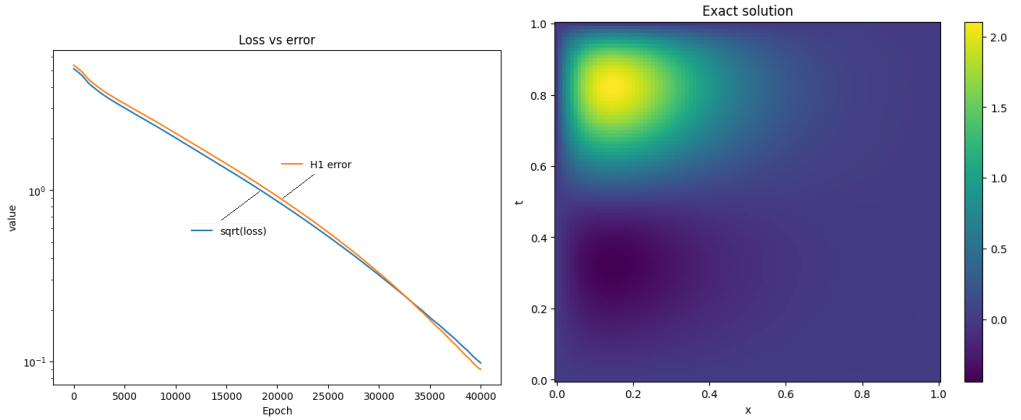
(a) Poisson problem with sin-exp forcing

$$-\Delta u = f_1;\ f_1(x_1, x_2) = -\Delta u(x_1, x_2) = \pi^2 e^{\pi(x-2y)} \sin(\pi y)$$
$$(4\cos(2\pi x) - 3\sin(2\pi x)) - \pi^2 e^{\pi(x-2y)} \sin(2\pi x)(4\cos(\pi y) - 3\sin(\pi y)).$$



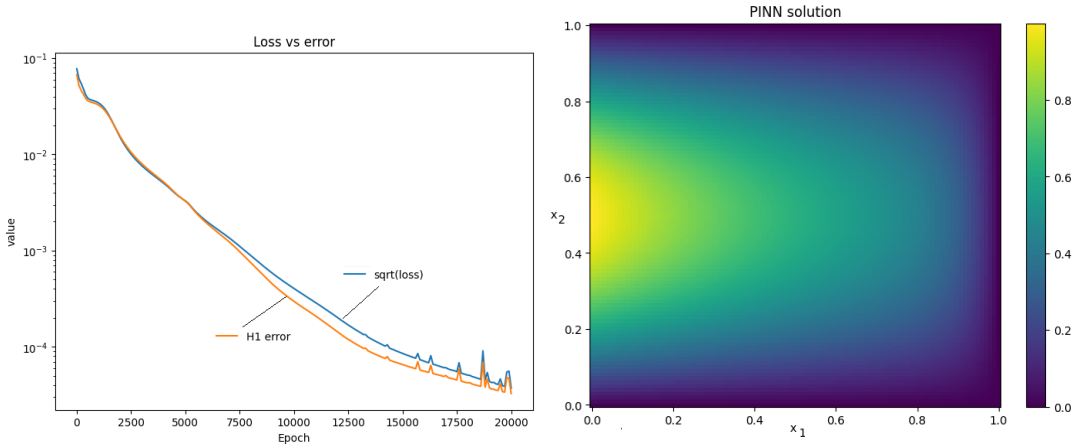
(b) Advection-diffusion problem $\begin{cases} \beta \cdot \nabla u - \epsilon \Delta u = 0 & \text{in } \Omega, \\ u = g & \text{over } \partial\Omega\,. \end{cases}$

Robust loss of CRVPINN



(a) Poisson problem with sin-exp forcing

$$-\Delta u = f_1;\ f_1(x_1, x_2) = -\Delta u(x_1, x_2) = \pi^2 e^{\pi(x-2y)} \sin(\pi y)$$
$$(4\cos(2\pi x) - 3\sin(2\pi x)) - \pi^2 e^{\pi(x-2y)} \sin(2\pi x)(4\cos(\pi y) - 3\sin(\pi y)).$$



(b) Advection-diffusion problem $\begin{cases} \beta \cdot \nabla u - \epsilon \Delta u = 0 & \text{in } \Omega, \\ u = g & \text{over } \partial\Omega\,. \end{cases}$

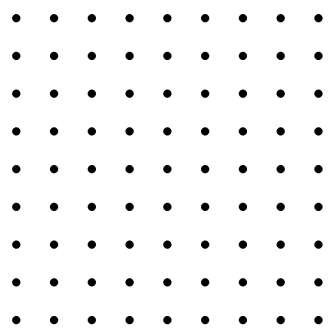
Mathematical tools to make the loss robust again

Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024) 116904

22 citations in 6 months (200 in 5 years ?)

Marcin łoś, Tomasz Służalec, Paweł Maczuga, Askold Vilkha, Carlos Uriarte, Maciej Paszyński, *Collocation-based Robust Variational Physics-Informed Neural Networks (CRVPINN)* **https://arxiv.org/abs/2401.02300** (2024)

Grid of points instead of continuous domain



Grid of $N \times N$ collocation points over a square domain

$$\Omega_h := \{(ih, jh) \in (0,1)^2 : 0 \leq i \leq N, 0 \leq j \leq N\},$$

where $h = 1/N$ denotes the discretization size. We consider

$$D_h := \{u : \Omega_h \longrightarrow \mathbb{R}\} \cong \mathbb{R}^{(N+1)^2}$$

Now, we follow the notation convention below for simplicity:

$$u_{i,j} := u(ih, jh), \qquad 0 \leq i, j \leq N.$$

Discrete norms and inner products defined at points

$$(u,v)_h := h^2 \sum_{p\in\Omega_h} u(p)v(p), \quad \|u\|_h^2 := (u,u)_h, \quad u,v \in D_h.$$

We consider finite-difference like gradient operations[4]

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

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

for $0 \leq i \pm 1, j \pm 1 \leq N$. As a result, we can define the following discrete inner product according to these gradient values:

$$(u,v)_{\nabla,h} := (\nabla_{x\pm} u, \nabla_{x\pm} v)_h + (\nabla_{y\pm} u, \nabla_{y\pm} v)_h$$

with corresponding induced norm

$$\|u\|_{\nabla,h}^2 := (u,u)_{\nabla,h} = \|\nabla_{x+} u\|_h^2 + \|\nabla_{y+} u\|_h^2.$$

[4] Dongho Shin, John Stikwerda, *Inf-sup conditions for finite differences approximations of Stokes equations*, **Australian Mathematical Society** (1997)

Discrete norms and inner products defined at points

We introduce a canonical orthonormal basis for D_h
$\delta_{i,j} : \Omega \longrightarrow \mathbb{R}$, $0 \leq i, j \leq N$, that behave as Kronecker deltas

$$\delta_{i,j}(x) = \begin{cases} 1 & \text{if } x = x_{i,j}, \\ 0 & \text{if } x \neq x_{i,j}. \end{cases}$$

We introduce the discretization space with homogeneous b.c.

$$D_{0,h} = \{u \in D_h : u|_{\partial\Omega} = 0\}.$$

Thus, $D_{0,h}$ is an $(N-1)^2$-dimensional space with basis $\{\delta_{i,j}\}_{0<i,j<N}$.

Rediscovering properties of discrete spaces

Lemma (Discrete integration by parts)

Given $u, v \in D_{0,h}$, it satisfies

$$(\nabla_{x+} u, v)_h = -(u, \nabla_{x+} v)_h, \qquad (\nabla_{y+} u, v)_h = -(u, \nabla_{y+} v)_h$$

Lemma (Discrete product rule)

Given $u, v \in D_{0,h}$, it satisfies

$$\nabla_{x+}(uv)_{i,j} = u_{i+1,j}(\nabla_{x+} v)_{i,j} + (\nabla_{x+} u)_{i,j} v_{i,j}.$$
$$\nabla_{y+}(uv)_{i,j} = u_{i,j+1}(\nabla_{y+} v)_{i,j} + (\nabla_{y+} u)_{i,j} v_{i,j}.$$

Lemma (Discrete norm equivalence - *Discrete Poincaré's inequality*)

There exist constants $0 < c < C$ such that

$$c\|u\|_{\nabla,h} \leq \|u\|_h \leq C\|u\|_{\nabla,h}, \qquad \forall u \in D_{0,h}.$$

Discrete weak variational reformulation

Advection-diffusion model problem: Find $u \in D_{0,h}$ such that

$$\beta_x \nabla_{x+} u + \beta_y \nabla_{y+} u - \epsilon \Delta_h u = f,$$

where $f, \beta_x, \beta_y \in D_{0,h}$ are given source and coefficient functions,

$$\Delta_h u_{i,j} := \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}.$$

Testing with $v \in D_{0,h}$,

$$(\beta_x \nabla_{x+} u + \beta_y \nabla_{y+} u - \epsilon \Delta_h u, v)_h = (f, v)_h,$$

applying Lemma 1 to Laplasian term, we obtain the following

Definition (Discrete weak variational reformulation)

Find $u \in D_{0,h}$ such that

$$\overbrace{(\beta_x \nabla_{x+} u + \beta_y \nabla_{y+} u, v)_h + \epsilon(\nabla_+ u, \nabla_+ v)_h}^{b(u,v)} = \overbrace{(f, v)_h}^{l(v)}, \quad \forall v \in D_{0,h},$$

b and l are the discrete bilinear/linear forms over $(D_{0,h})^2$ and $D_{0,h}$.

Properties of the discrete weak variational formulation

Lemma (Boundedness of b)

For bounded coefficients β_x and β_y, there exists $\mu > 0$ such that

$$b(u, v) \leq \mu \, \|u\|_{\nabla,h} \, \|v\|_{\nabla,h}, \qquad \forall u, v \in D_{0,h}.$$

We have $\mu = (2C\|\beta\|_\infty + \epsilon)$, where $\|\beta\|_\infty = \max\{|\beta_x|, |\beta_y|\}$.

Lemma (Coercivity of b)

There exists $\alpha > 0$ such that

$$b(u, u) \geq \alpha \, \|u\|^2_{\nabla,h}, \qquad \forall u \in D_{0,h},$$

whenever $\epsilon > 2C\|\beta\|_\infty$, where $C > 0$ is the constant from Poincarè's inequality.

We have $\alpha = (-2C\|\beta\|_\infty + \epsilon)$

Well-posedness of discrete weak formulation

As a result, (sup-sup) continuity and (inf-sup) stability constants exist with values at most μ and at least α, respectively,

$$\sup_{u\neq 0}\sup_{v\neq 0}\frac{b(u,v)}{\|u\|_{\nabla,h}\|v\|_{\nabla,h}}\leq \mu,$$

$$\inf_{u\neq 0}\sup_{v\neq 0}\frac{b(u,v)}{\|u\|_{\nabla,h}\|v\|_{\nabla,h}}\geq \alpha,$$

$$\inf_{v\neq 0}\sup_{u\neq 0}\frac{b(u,v)}{\|u\|_{\nabla,h}\|v\|_{\nabla,h}}\geq \alpha.$$

Consequently, discreat weak variational problem:
Find $u \in D_{0,h}$ such that

$$\overbrace{(\beta_x \nabla_{x+} u + \beta_y \nabla_{y+} u, v)_h + \epsilon(\nabla_+ u, \nabla_+ v)_h}^{b(u,v)} = \overbrace{(f, v)_h}^{l(v)}, \quad \forall v \in D_{0,h},$$

admits a unique solution in $D_{0,h}$.

How to define robust loss

Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024) 116904

Marcin łoś, Tomasz Służalec, Paweł Maczuga, Askold Vilkha, Carlos Uriarte, Maciej Paszyński, *Collocation-based Robust Variational Physics-Informed Neural Networks (CRVPINN)* **https://arxiv.org/abs/2401.02300** (2024)

Towards Collocation-based Robust Variational Physics Informed Neural Networks (CRVPINN)

Theorem (Robustness)

Let $u \in D_{0,h}$ and let $LOSS(u) = RES(u)^T \mathbf{G}^{-1} RES(u)$. Then,

$$\frac{1}{\mu}\sqrt{LOSS(u)} \leq \|u - u_{EXACT}\|_{\nabla,h} \leq \frac{1}{\alpha}\sqrt{LOSS(u)},$$

where μ and α are the boundedness and coercivity constants of b.

Recall $\mu = (2C\|\beta\|_\infty + \epsilon)$, where $\|\beta\|_\infty = \max\{|\beta_x|, |\beta_y|\}$, and $\alpha = (-2C\|\beta\|_\infty + \epsilon)$

Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024)

Robust loss function

PINN loss

$$RES(u_\theta) = \sum_{\mathbf{x}} (\Delta u_\theta(\mathbf{x}) + f(\mathbf{x}))^2$$

Robust loss

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

The Gram matrix employs the Kronecker delta test functions

$$\mathbf{G}_{i,j;k,l} = h^{-2} \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j),(i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1),(i,j-1)\} \end{cases}$$

Remark. 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 bounded inf-sup stable.

Instead of the inverse $\mathbf{G}^{-1}$ we compute LU factorization $\mathbf{G} = \mathbf{LU}$.

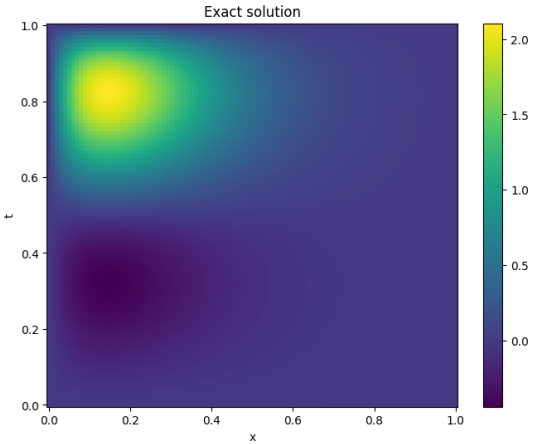
$$LOSS(u_\theta) = RES^T(u_\theta) q, \qquad \mathbf{L} q = z, \qquad \mathbf{U} z = RES(u_\theta).$$

Gallery of numerical problems

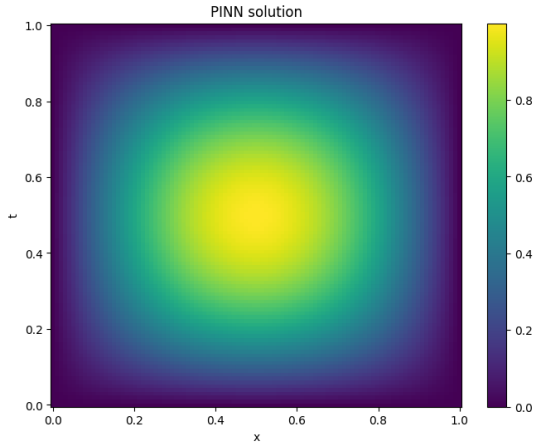
Sergio Rojas, Paweł Maczuga, Judit Muñoz-Matute, David Pardo, Maciej Paszyński, *Robust Variational Physics-Informed Neural Networks*, **Computer Methods in Applied Mechanics and Engineering** 425 (2024) 116904

Marcin łoś, Tomasz Służalec, Paweł Maczuga, Askold Vilkha, Carlos Uriarte, Maciej Paszyński, *Collocation-based Robust Variational Physics-Informed Neural Networks (CRVPINN)* **https://arxiv.org/abs/2401.02300** (2024)

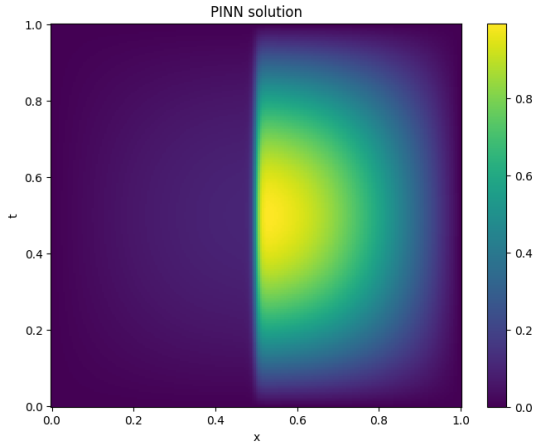
Gallery of problems (Poisson, Advection-diffusion, Stokes)



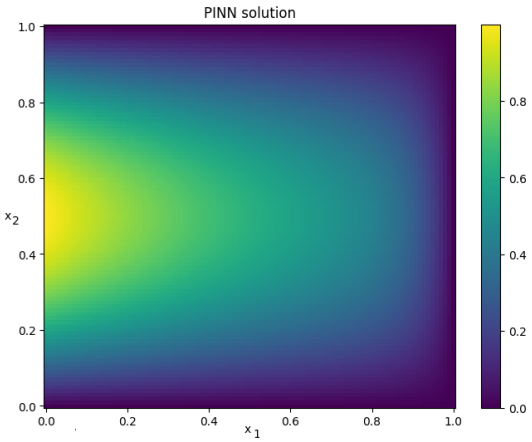
(a) $-\Delta u = f_1$.



(b) $-\nabla \cdot (\epsilon(x_2)\nabla u) = f_2$

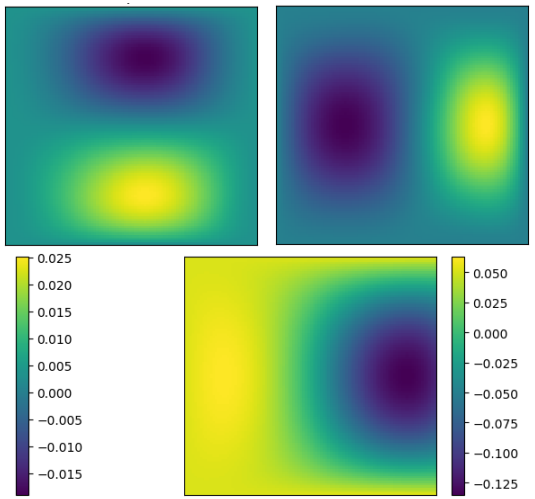


(c) $-\Delta u = f_3$.



(d)

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



(e)

$$-\Delta \mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega$$

$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega, \qquad \mathbf{u} = \mathbf{g} \text{ in } \Gamma$$

Poisson problems with zero Dirichlet b.c.

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

  $f_1(x_1, x_2) = -\Delta u(x_1, x_2) = \pi^2 e^{\pi(x-2y)} \sin(\pi y)(4\cos(2\pi x)$
  $-3\sin(2\pi x)) - \pi^2 e^{\pi(x-2y)} \sin(2\pi x)(4\cos(\pi y) - 3\sin(\pi y))$

- $-\nabla \cdot (\epsilon(x_2) \nabla u) = f_2,\ \epsilon(x_2) = 2(x_2 + 1),$
  $u(x_1, x_2) = \sin(2\pi x_1) \sin(\pi x_2)$

  $f_2(x_1, x_2) = \pi \sin(\pi x_1)[\cos(\pi x_2) d\epsilon(x_2)/(dx_2) - 2\pi\epsilon(x_2) \sin(\pi x_2)]$

- $-\Delta u = f_3$
  $u(x_1, x_2) = (0.45 \tanh(100(x_2 - 0.5)) + 0.55) \sin(\pi x_1) \sin(\pi x_2)$

  $f_3(x_1, x_2) = \sin(\pi x_1)(\sin(\pi x_2)$
  $(-10.8566 - 8.88264 \tanh(100(-0.5 + x_2))) +$
  $1/(\cosh(100(-0.5 + x_2)))^2$
  $(282.743 \cos(\pi x_2) - 9000 \sin(\pi x_2) \tanh(100(-0.5 + x_2))))$

Advection-diffusion problem

$\Omega = (0,1)^2 \subset \mathbb{R}^2$ we seek solution[5]

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

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

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

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

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

[5] Kenneth Eriksson, Claes Johnson. *Adaptive finite element methods for parabolic problems I: A linear model problem.* **SIAM Journal of Numerical Analysis** (1991)

Stokes problem

$$-\Delta \mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega$$
$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega, \qquad \mathbf{u} = \mathbf{g} \text{ in } \Gamma$$

$$\begin{aligned}
u_1(x_1,x_2) =& 2e_1^x(-1+x_1)^2x_1^2(x_2^2+x_2)(-1+2x_2),\\
u_2(x_1,x_2) =& -e_1^x(-1+x_1)x_1(-2+x_1*(3+x_1))(-1+x_2)^2x_2^2,\\
p(x_1,x_2) =& (-424+156\cdot 2.718+(x_2^2-x_2)(-456+\\
& e_1^x(456+x_1^2(228-5(x_2^2-x_2))+2x_1(-228+(x_2^2-x_2))+\\
& 2x_1^3(-36+(x_2^2-x_2))+x_1^4*(12+(x_2^2-x_2))))),
\end{aligned}$$

$$f_1(x_1,x_2) = -\frac{\partial^2 u_1(x_1,x_2)}{\partial x_1^2} - \frac{\partial^2 u_1(x_1,x_2)}{\partial x_2^2} + \frac{\partial p}{\partial x_1},$$

$$f_2(x_1,x_2) = -\frac{\partial^2 u_2(x_1,x_2)}{\partial x_1^2} - \frac{\partial^2 u_2(x_1,x_2)}{\partial x_2^2} + \frac{\partial p}{\partial x_2},$$

$$g_1(x_1,x_2) = u_1(x_1,x_2), \quad (x_1,x_2) \in \partial\Omega$$
$$g_2(x_1,x_2) = u_2(x_1,x_2), \quad (x_1,x_2) \in \partial\Omega.$$

Gram matrices

- Poisson problem $-\Delta u = f$
- Advection-diffusion problem $\beta \cdot \nabla u - \epsilon \Delta u = f$

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

- Poisson problem with non-constant coefficients $-\nabla \cdot (\epsilon(x_2) \nabla u) = f$,

$$\hat{\mathbf{G}}_{\zeta_1,\zeta_2} = h(\epsilon \nabla \delta_{ij}, \delta_{kl}) = \begin{cases} 2\epsilon_{i,j} + \epsilon_{i-1,j} + \epsilon_{i,j-1} & (k,l) = (i,j) \\ -\epsilon_{i-1,j} & (k,l) = (i-1,j) \\ -\epsilon_{i,j} & (k,l) = (i+1,j) \\ -\epsilon_{i,j} & (k,l) = (i,j+1) \\ -\epsilon_{i,j-1} & (k,l) = (i,j-1) \end{cases}$$

Gram matrices of Stokes problem

$$
\begin{aligned}
-\Delta \mathbf{u} + \nabla p &= \mathbf{f} \text{ in } \Omega \\
\nabla \cdot \mathbf{u} &= 0 \text{ in } \Omega, \\
\mathbf{u} &= \mathbf{g} \text{ in } \Gamma
\end{aligned}
\qquad
G = \begin{bmatrix} G_{\sigma} & G_{\sigma \mathbf{u}} & G_{\sigma p} \\ G_{\sigma \mathbf{u}}^T & G_{\mathbf{u}} & 0 \\ G_{\sigma p}^T & 0 & G_p \end{bmatrix}, \quad G_p = K + M
$$

$$
G_{\sigma} = \begin{bmatrix} K_+^x & S_+^T & 0 & 0 \\ S_+ & K_+^y & 0 & 0 \\ 0 & 0 & K_+^x & S_+^T \\ 0 & 0 & S_+ & K_+^y \end{bmatrix} + 2 \begin{bmatrix} M & 0 & 0 & 0 \\ 0 & M & 0 & 0 \\ 0 & 0 & M & 0 \\ 0 & 0 & 0 & M \end{bmatrix}
$$

$$
G_{\mathbf{u}} = \begin{bmatrix} 2K_-^x + K_-^y & S_-^T \\ S_- & K_-^x + 2K_-^y \end{bmatrix} + \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix}
$$

$$
G_{\sigma \mathbf{u}}^T = \begin{bmatrix} A_-^x & A_-^y & 0 & 0 \\ 0 & 0 & A_-^x & A_-^y \end{bmatrix}, \quad G_{\sigma p}^T = -\begin{bmatrix} K_+^x & S_+ & S_+^T & K_+^y \end{bmatrix}
$$

$$
M \sim (f, g)_{L^2}, \quad K_{\pm} \sim (\nabla_{\pm} f, \nabla_{\pm} g)_{L^2}, \quad S_{\pm} \sim (\nabla_{x\pm} f, \nabla_{y\pm} g)_{L^2}
$$

$$
K_{\pm}^x \sim (\nabla_{x\pm} f, \nabla_{x\pm} g)_{L^2}, \quad K_{\pm}^y \sim (\nabla_{y\pm} f, \nabla_{y\pm} g)_{L^2}.
$$

$$
A_{\pm}^x \sim (\nabla_{x\pm} f, g)_{L^2}, \quad A_{\pm}^y \sim (\nabla_{y\pm} f, g)_{L^2}.
$$

Discrete weak formulations

- Poisson problems: We seek $u \in D_{0,h}$:

$$\overbrace{\epsilon(\nabla_+ u, \nabla_+ v)_h}^{b(u,v)} = \overbrace{(f, v)_h}^{l(v)}, \quad \forall v \in D_{0,h},$$

- Advection-diffusion problem: We seek $u \in D_{0,h}$:

$$\overbrace{(\beta_x \nabla_{x+} u + \beta_y \nabla_{y+} u, v)_h + \epsilon(\nabla_+ u, \nabla_+ v)_h}^{b(u,v)} = \overbrace{(f, v)_h}^{l(v)}, \quad \forall v \in D_{0,h},$$

- Stokes problem: We seek $u = (\boldsymbol{\sigma}, \mathbf{u}, p) \in D_h^\sigma \times D_{0,h}^2 \times D_h^p$:

$$(Au, v)_h = (-\nabla_+ \cdot \boldsymbol{\sigma} + \nabla_+ p, \mathbf{v})_h + (\nabla_- \cdot \mathbf{u}, q)_h + (\boldsymbol{\sigma} - \nabla_- \mathbf{u}, \boldsymbol{\tau})_h =$$
$$(f, \mathbf{v})_h \quad \forall (\boldsymbol{\tau}, \mathbf{v}, q) \in D_h^\sigma \times D_{0,h}^2 \times D_h^p$$

$$D_h^p = \left\{ p \in D_h : p|_{\Gamma_p} = 0, (p, 1)_h = 0 \right\}, D_h^\sigma = \left\{ \boldsymbol{\sigma} \in D_h^4 : \sigma_{ij}|_{\Gamma_\sigma^j} = 0 \right\}$$

$$\Gamma_p = \{(0, jh) : 0 \le j \le N\} \cup \{(ih, 0) : 0 \le i \le N\} \cup \{(1, 1)\} \subset \partial\Omega_h,$$
$$\Gamma_\sigma^1 = \{(0, jh) : 0 \le j \le N\}, \Gamma_\sigma^2 = \{(ih, 0) : 0 \le i \le N\}$$

Discrete inner products

- Poisson problems
- Advection-diffusion problem

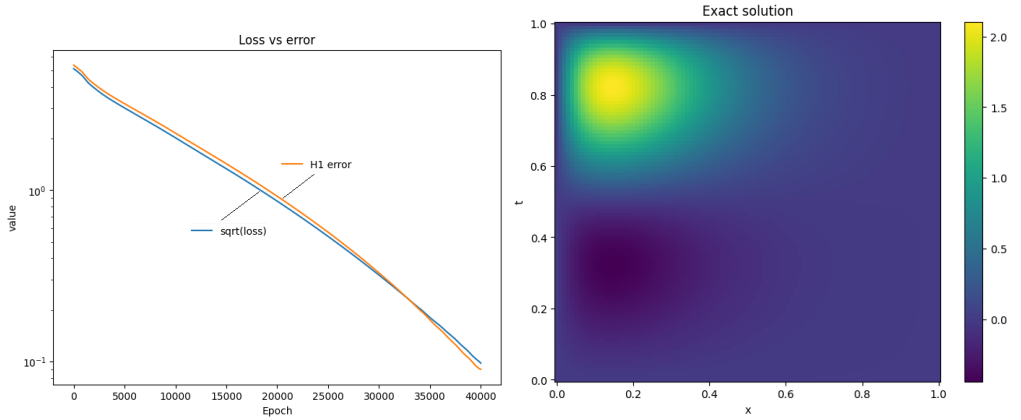
$$\begin{aligned}(u,v)_{\nabla,h} &:= (\nabla_{x+}u, \nabla_{x+}v)_h + (\nabla_{y+}u, \nabla_{y+}v)_h \\ &= (\nabla_{x-}u, \nabla_{x-}v)_h + (\nabla_{y-}u, \nabla_{y-}v)_h,\end{aligned}$$

- Stokes problem[6]

$$\begin{aligned}(u,v)_{\text{graph}} &= (\nabla_+ \cdot \boldsymbol{\sigma} - \nabla_+ p, \nabla_+ \cdot \boldsymbol{\tau} - \nabla_+ q)_h + (\nabla_- \cdot \mathbf{u}, \nabla_- \cdot \mathbf{v})_h \\ &+ (\boldsymbol{\sigma} + \nabla_- \mathbf{u}, \boldsymbol{\tau} + \nabla_- \mathbf{v})_h + (\boldsymbol{\sigma}, \boldsymbol{\tau})_h + (\mathbf{u}, \mathbf{v})_h + (p, q)_h \\ &= (\nabla_+ \cdot \boldsymbol{\sigma}, \nabla_+ \cdot \boldsymbol{\tau})_h + 2\,(\boldsymbol{\sigma}, \boldsymbol{\tau})_h \\ &+ (\nabla_- \cdot \mathbf{u}, \nabla_- \cdot \mathbf{v})_h + (\nabla_- \mathbf{u}, \nabla_- \mathbf{v})_h + (\mathbf{u}, \mathbf{v})_h \\ &+ (\nabla_+ p, \nabla_+ q)_h + (p, q)_h \\ &+ (\nabla_- \mathbf{u}, \boldsymbol{\tau})_h + (\boldsymbol{\sigma}, \nabla_- \mathbf{v})_h \\ &+ (-\nabla_+ p, \nabla_+ \cdot \boldsymbol{\tau})_h + (\nabla_+ \cdot \boldsymbol{\sigma}, -\nabla_+ q)_h\end{aligned}$$

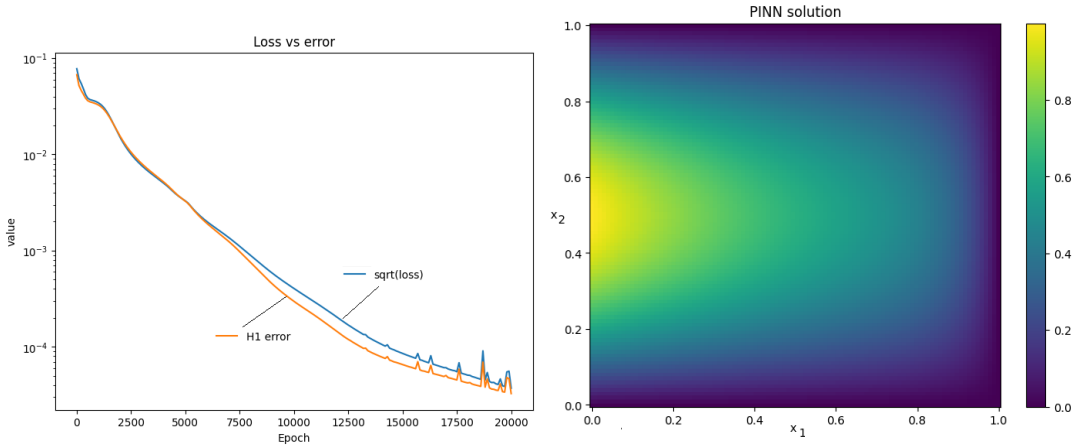
[6] Norbert Roberts, Thai Bui-Thanh, Leszek Demkowicz, *The DPG method for the Stokes problem*, **Computers & Mathematics with Applications** 67 (2014).

Robust loss of CRVPINN



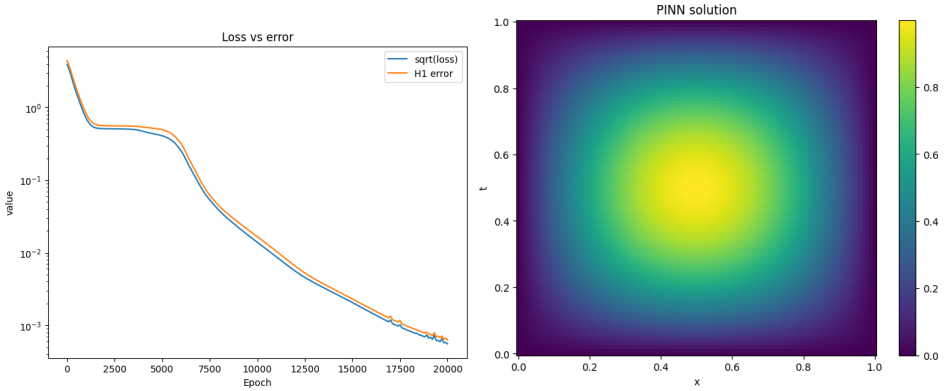
(a) Poisson problem with sin-exp forcing

$$-\Delta u = f_1;\ f_1(x_1, x_2) = -\Delta u(x_1, x_2) = \pi^2 e^{\pi(x-2y)} \sin(\pi y)$$
$$(4\cos(2\pi x) - 3\sin(2\pi x)) - \pi^2 e^{\pi(x-2y)} \sin(2\pi x)(4\cos(\pi y) - 3\sin(\pi y)).$$



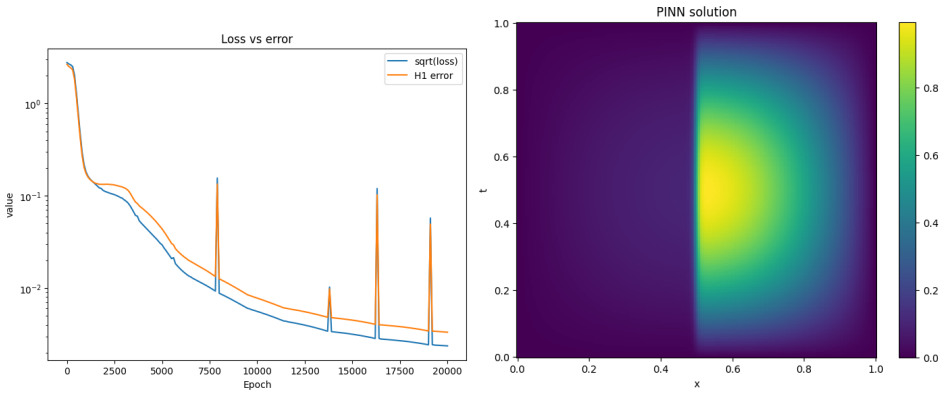
(b) Advection-diffusion problem $\begin{cases} \beta \cdot \nabla u - \epsilon \Delta u = 0 & \text{in } \Omega, \\ u = g & \text{over } \partial\Omega\,. \end{cases}$

Robust losts of CRVPINN



(a) Poisson problem with varying diffusion

$$\begin{cases} -\nabla \cdot (\epsilon(x_2)\nabla u) = f_2, \qquad \epsilon(x_2) = 2(x_2+1), \\ f_2(x_1,x_2) = \pi \sin(\pi x_1)[\cos(\pi x_2) d\epsilon(x_2)/(dx_2) - 2\pi\epsilon(x_2)\sin(\pi x_2)] \end{cases}$$



(b) Poisson problem with a jump

$$\begin{cases} -\Delta u = f_3, \quad f_3(x_1,x_2) = \sin(\pi x_1)(\sin(\pi x_2)(-10.8566 - 8.88264\tanh(100(-0.5+x_2))) + \\ \qquad 1/(\cosh(100(-0.5+x_2)))^2(282.743\cos(\pi x_2) - 9000\sin(\pi x_2)\tanh(100(-0.5+x_2)))) \end{cases}$$

Robust loss function for Stokes problem

We define the following robust loss for Stokes problem

$$RES(u_\theta) = \begin{bmatrix} RES_{6a}(u_\theta) \\ RES_{6b}(u_\theta) \\ RES_{6c}(u_\theta) \\ RES_{6d}(u_\theta) \\ RES_{6e}(u_\theta) \\ RES_{6f}(u_\theta) \\ RES_{6g}(u_\theta) \end{bmatrix} = \begin{bmatrix} \frac{\partial u_1}{\partial x_1} - w_1 \\ \frac{\partial u_1}{\partial x_2} - w_2 \\ \frac{\partial u_2}{\partial x_1} - z_1 \\ \frac{\partial u_2}{\partial x_2} - z_2 \\ -\frac{\partial w_1}{\partial x_1} - \frac{\partial w_2}{\partial x_2} + \frac{\partial p}{\partial x_1} - f_1, \\ -\frac{\partial z_1}{\partial x_1} - \frac{\partial z_2}{\partial x_2} + \frac{\partial p}{\partial x_2} - f_2 \\ \frac{\partial u_1(x_1,x_2)}{\partial x_1} + \frac{\partial u_2(x_1,x_2)}{\partial x_2} \end{bmatrix}$$

$$LOSS(u_\theta) = RES(u_\theta)^T \begin{bmatrix} G_{\boldsymbol{\sigma}} & G_{\boldsymbol{\sigma}\mathbf{u}} & G_{\boldsymbol{\sigma}p} \\ G_{\boldsymbol{\sigma}\mathbf{u}}^T & G_{\mathbf{u}} & 0 \\ G_{\boldsymbol{\sigma}p}^T & 0 & G_p \end{bmatrix}^{-1} RES(u_\theta)$$

Convergence of CRVPINN: Stokes problem

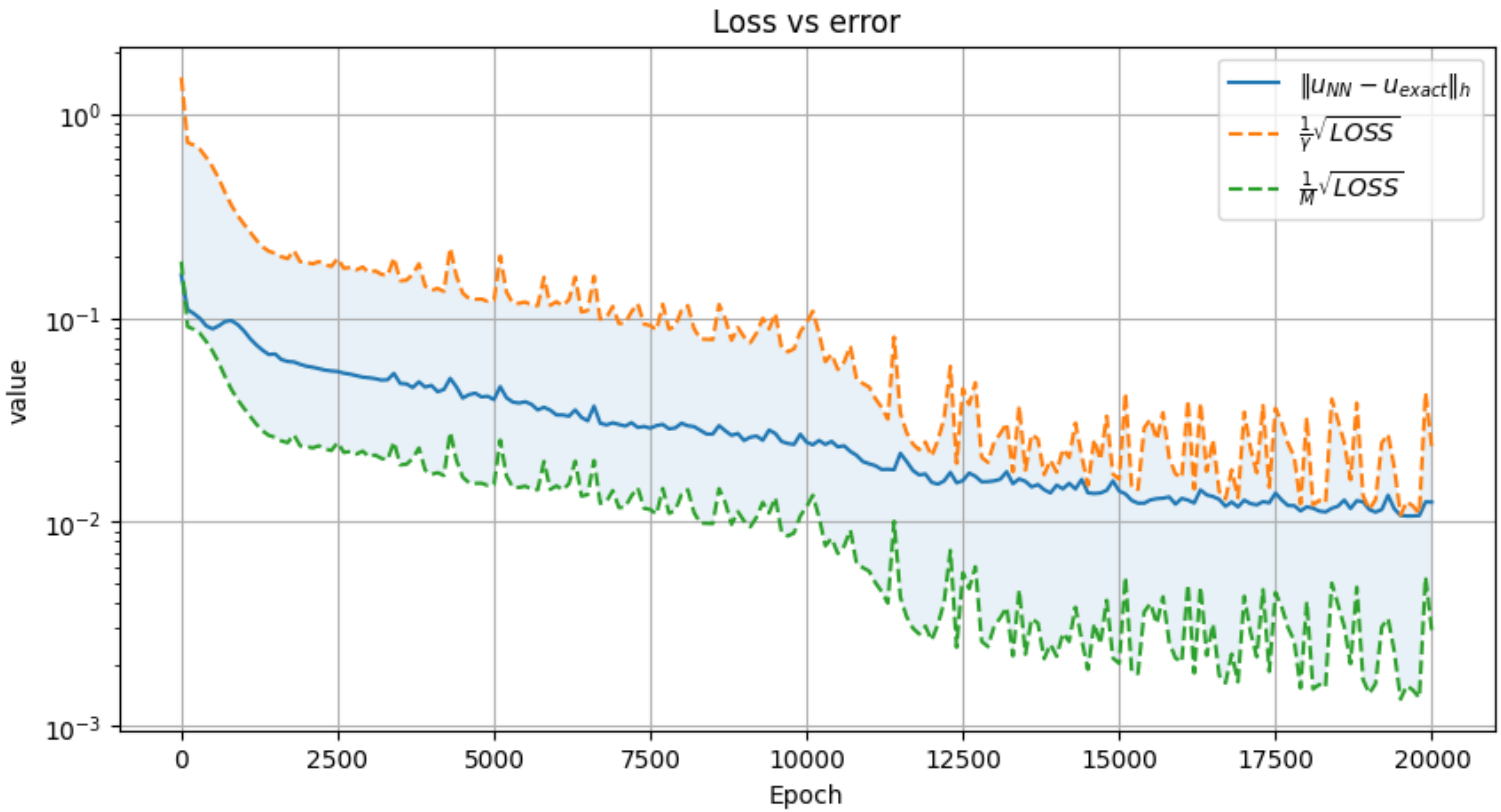


Figure: Robust loss and the true error coincides with theoretical estimates for the continuity constant $\frac{1}{\mu} = 1$ and inf-sup constant $\frac{1}{\alpha} = 8$

$$1\sqrt{LOSS(u_\theta)} < \|u_\theta - u_{exact}\| < 8\sqrt{LOSS(u_\theta)}$$

(CRVPINN): Stokes problem)

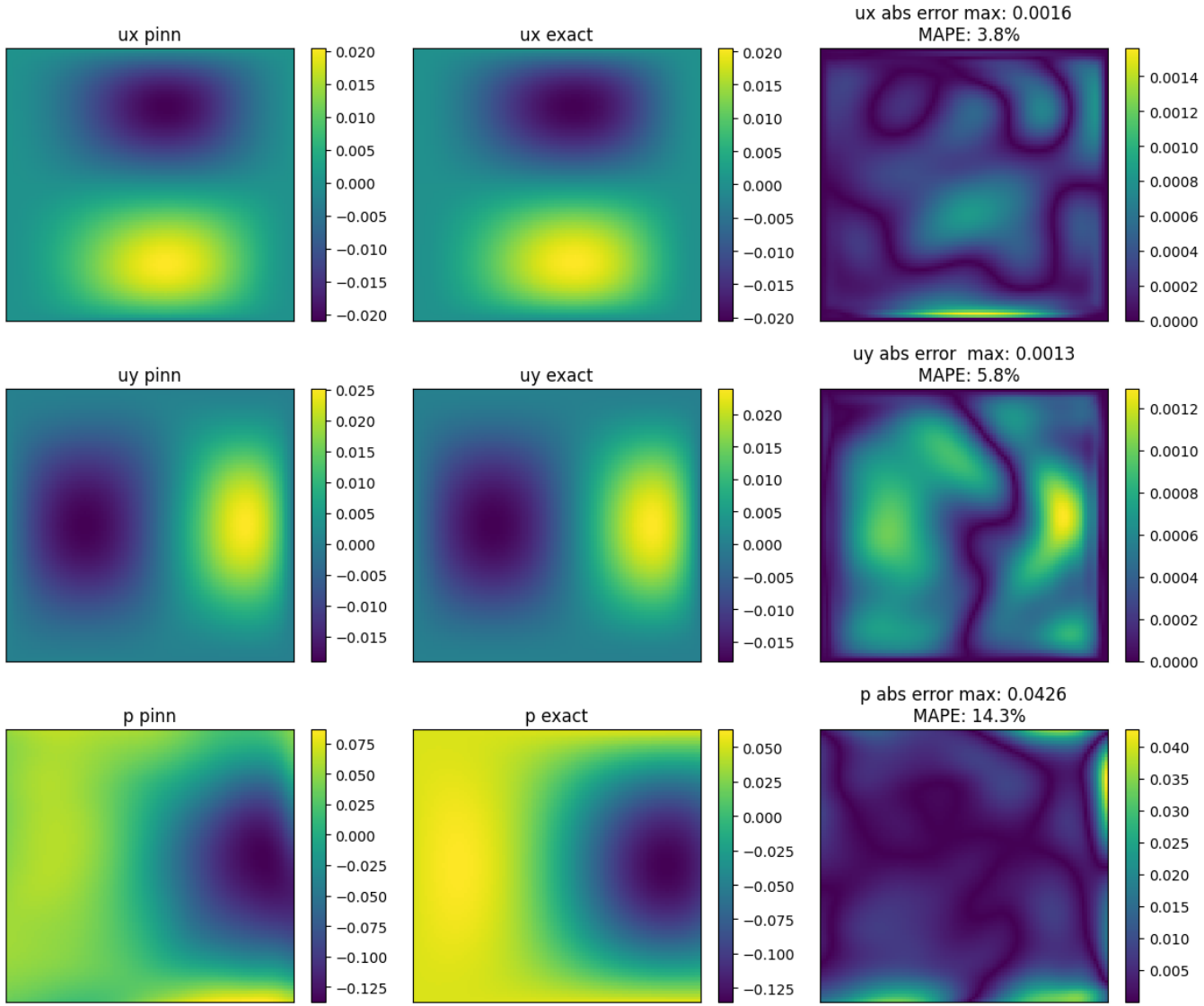


Figure: Comparison between the solution generated by CRVPINN and the exact solution of the Stokes problem with manufactured solution.

(CRVPINN): Non-linear stationary Navier-Stokes)

We add the non-linear term $u \cdot \nabla u$

$$\begin{cases} u \cdot \nabla u - \Delta \mathbf{u} + \nabla p = f & \text{in } \Omega, \\ \nabla \cdot \mathbf{u} = 0 & \text{in } \Omega, \\ \mathbf{u} = 0 & \text{in } \Gamma \end{cases}$$

We employ the Gram matrix and the loss function as for the Stokes.

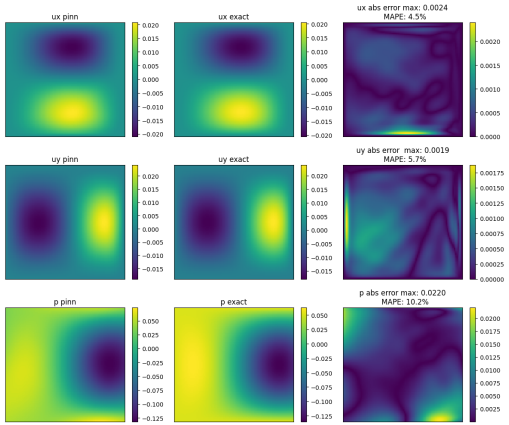
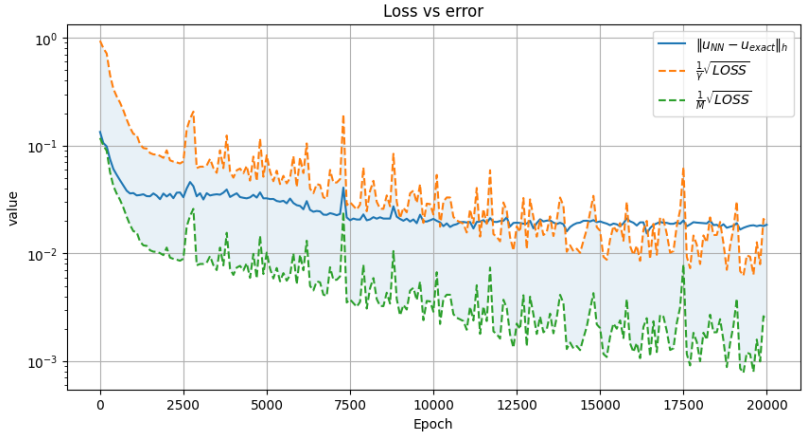
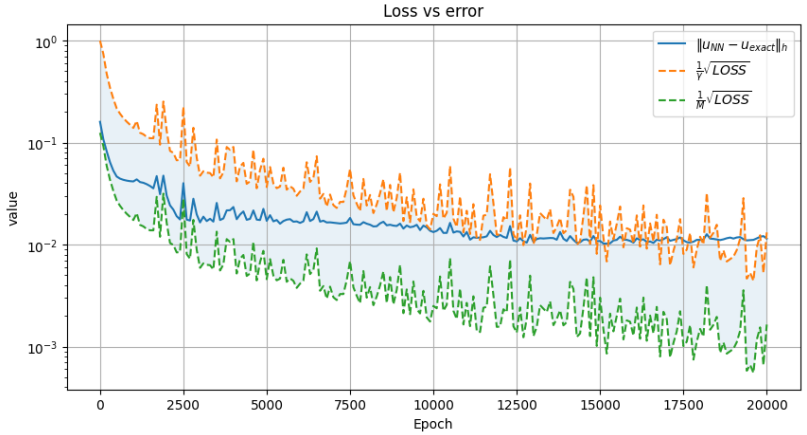


Figure: CRVPINN solution of the non-linear stationary Navier-Stokes.

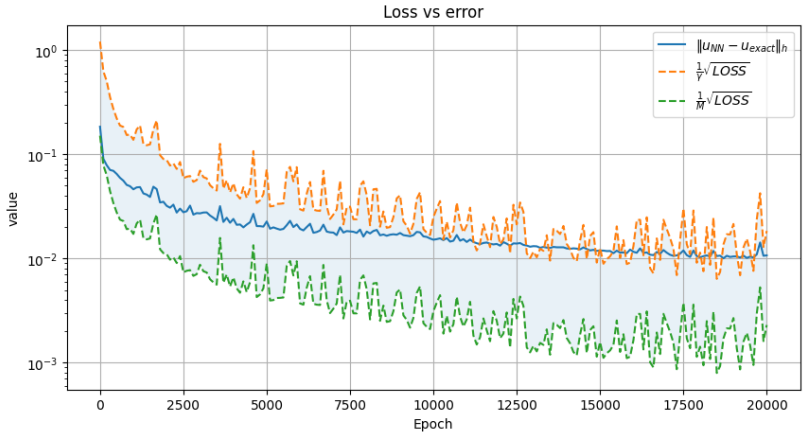
(CRVPINN): Non-linear stationary Navier-Stokes)



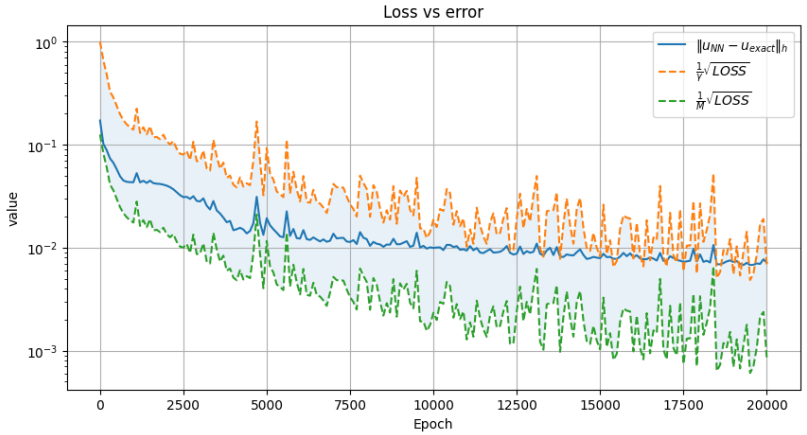
(a) CRVPINN training for 20x20 points.



(b) CRVPINN training for 30x30 points.



(c) CRVPINN training for 40x40 points.



(d) CRVPINN training for 50x50 points.

(CRVPINN): Circular wind problem)

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

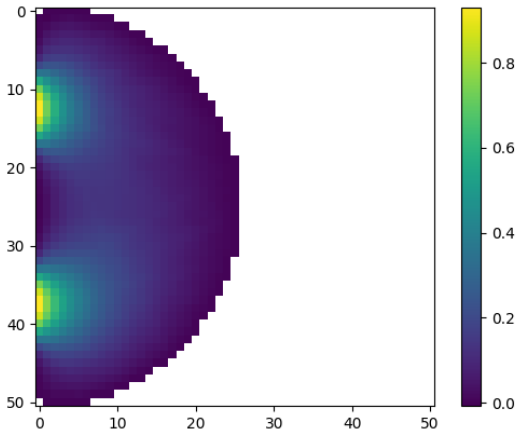
The problem is defined over the rectangular domain $\Omega = (0,1) \times (-1,1)$, with zero right-hand side $f = 0$, and with the advection vector $\beta = (-y, -x)$ modeling the rotational flow. We introduce the boundary with the inflow and the outflow

$$\Gamma_1 = \{(x,y) : x = 0.0, 0.5 \leq y \leq 1.0\}$$
$$\Gamma_2 = \{(x,y) : x = 0.0, 0.0 \leq y \leq 0.5\}$$

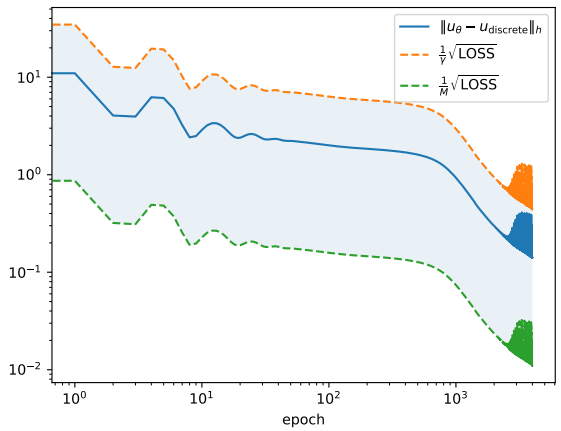
We introduce the Dirichlet b.c. for the inflow and the outflow

$$g(x_1,x_2) = \frac{1}{2}\left(\tanh((|x_2| - 0.35)\frac{b}{\epsilon}) + 1\right), \text{ for } (x_1,x_2) \in \Gamma_2$$
$$g(x_1,x_2) = \frac{1}{2}\left(\tanh((0.65 - (|x_2|))\frac{b}{\epsilon}) + 1\right), \text{ for } (x\_1,x_2) in \Gamma_2$$
$$g(x_1,x_2) = 0, \text{ for } (x_1,x_2) \in \Gamma \setminus \Gamma_1 \cup \Gamma_2$$

(CRVPINN): Circular wind problem)



(a) CRVPINN solution for $\epsilon = 0.1$.



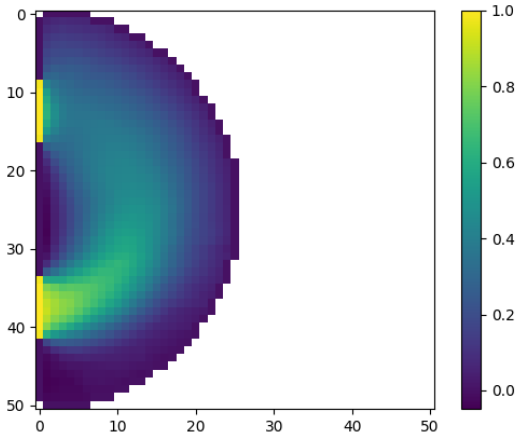
(b) Exact solution for $\epsilon = 0.1$.

The convergence of the CRVPINN training depends on the value of the diffusion coefficient ϵ, since $\alpha = \epsilon$ and $\mu = \epsilon + 2C = 4 + \epsilon$.
(Theorem 1) $\frac{1}{\mu}\sqrt{LOSS(u)} \leq \|u - u_{\text{EXACT}}\|_{\nabla,h} \leq \frac{1}{\alpha}\sqrt{LOSS(u)}$
For $\epsilon = 0.1$ we have

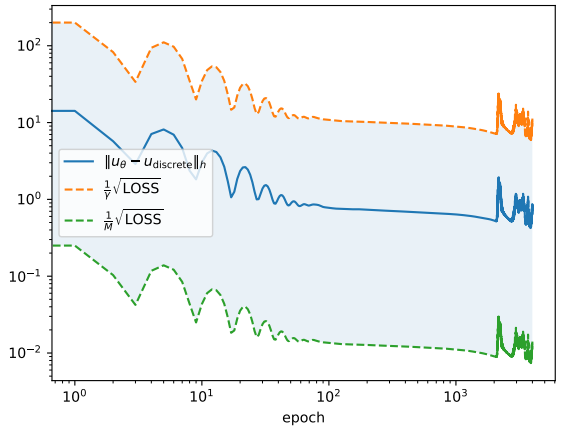
$$0.25\sqrt{LOSS(u)} \leq \|u - u_{\text{EXACT}}\|_{\nabla,h} \leq 10\sqrt{LOSS(u)}$$

These limits they define the range where the true error stays during the training process.

(CRVPINN): Circular wind problem)



(a) CRVPINN solution for $\epsilon = 0.05$.



(b) Exact solution for $\epsilon = 0.05$.

The convergence of the CRVPINN training depends on the value of the diffusion coefficient ϵ, since $\alpha = \epsilon$ and $\mu = \epsilon + 2C = 4 + \epsilon$.
(Theorem 1) $\frac{1}{\mu}\sqrt{LOSS(u)} \leq \|u - u_{\text{EXACT}}\|_{\nabla,h} \leq \frac{1}{\alpha}\sqrt{LOSS(u)}$
For $\epsilon = 0.05$ we have

$$0.25\sqrt{LOSS(u)} \leq \|u - u_{\text{EXACT}}\|_{\nabla,h} \leq 200\sqrt{LOSS(u)}$$

These limits they define the range where the true error stays during the training process.

Conclusions - Key points

Our CRVPINN Robust loss

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

is better than PINN loss

$$RES(u_\theta) = \sum_{x,y} (\Delta u_\theta(x,y) + f(x,y))^2$$

Gram matrix constructed with inner product of Kronecker deltas $(\delta_{ij}, \delta_{kl})_{\nabla,h}$(Poisson, advection-diffusion) or $(\delta_{ij}, \delta_{kl})_{graph}$(Stokes), for the norm that discrete weak form is bounded inf-sup stable.

Do not invert $\mathbf{G}^{-1}$. Compute LU factorization $\mathbf{G} = \mathbf{LU}$ instead.

The Gram matrix is sparse like finite difference method

$$\mathbf{G}_{i,j;k,l} = h^{-2} \begin{cases} 4 & \text{for } (i,j) = (k,l) \\ -1 & \text{for } (k,l) \in \{(i+1,j),(i-1,j)\} \\ -1 & \text{for } (k,l) \in \{(i,j+1),(i,j-1)\} \end{cases}$$