Lecture 4: Variational NNs and the Deep Ritz Method

Chris Budd[1]

[1]University of Bath

Seattle, June 2025

Some papers to look at

- Grossman et. al. *Can PINNS beat the FE method?*
- Shin et. al. *On the convergence of PINNS for linear elliptic and parabolic PDEs*
- E and Yu *The Deep Ritz Method*
- Appella, B, et. al. *Equidistribution based training of FKS and ReLU Neural Networks*

Motivation: Calculus of variations

Seek to solve PDE problems eg. of the form

$$-\epsilon^2 u_{xx} + u = 1, \quad u(0) = u(1) = 0, \quad \Omega = [0, 1] \quad (*)$$

In the previous lecture we tried to solve this PDE using a PINN with the PDE residual as the loss function

In this lecture we consider using ideas from the calculus of variations to find a loss function

We will also introduce some ideas which will be needed when we study **Neural Operators** in later lectures.

Functional minimisers

Consider the functional

$$F(u) = \int_\Omega \frac{\epsilon^2 u_x^2}{2} + \frac{u^2}{2} - u \, dx.$$

Claim $F(u)$ is minimised when u is the (unique) solution of the PDE (*)

Question: Over what space is $F(u)$ minimised.

Answer: Space is $H_0^1(\Omega)$

$$H_0^1(\Omega) = \{u : \int_\Omega u_x^2 \, dx < \infty, \quad u(0) = u(1) = 0.\}.$$

Approach 1: Finite Element Methodology

Express $u(x)$ as a Galerkin approximation:

$$u(x) \approx U(x) = \sum_{i=0}^{N} U_i(t)\, \phi_i(x)$$

with $\phi_i(x)$ **Not** globally differentiable *locally supported, piece-wise polynomial spline functions*

U is in the linear space $\mathcal{V}$ spanned by the functions $\phi_i(x)$.

- Require U to satisfy the *weak form* of the PDE.
- If the PDE is linear, this leads to a linear system for the coefficients U_i of the approximation
- Solve this systemusing (for example) a conjugate-gradient method.

Why Finite Element Methods are so powerful: Céas Lemma

Suppose that

$$-\Delta u = f(x), \quad x \in \Omega, \quad u = 0 \quad x \in \partial\Omega$$

On a mesh $\mathcal{T}$ we have the finite element approximation U and the interpolant Πu (See Lecture 2)

Theorem [Céa]

$$\|u - U\|_{H_0^1} \leq \|u - \Pi u\|_{H_0^1}.$$

This theorem gives a strong, and evaluable, upper bound on the FE error.
Once you have got an FE solution you know it is a good one!

Proof This relies on the fact that the FE approximation is linear

$$\|u-\Pi u\|^2 = \|u-U+U-\Pi u\|^2 = \|u-\Pi u\|^2+2\langle u-U, U-\Pi u\rangle+\|\Pi u-U\|^2.$$

But the weak form of the PDE is

$$\langle u, \phi\rangle = \langle U, \phi\rangle = \int_\Omega f\phi \, dx, \quad \forall \phi \in \mathcal{V}$$

Therefore

$$\langle u - U, \phi\rangle = 0, \quad \forall \phi \in \mathcal{V}$$

But

$$U - \Pi u \equiv \phi \in \mathcal{V}$$

Hence

$$\|u - \Pi u\|^2 = \|u - U + U - \Pi u\|^2 = \|u - \Pi u\|^2 + \|\Pi u - U\|^2.$$

□

Features of the FE method

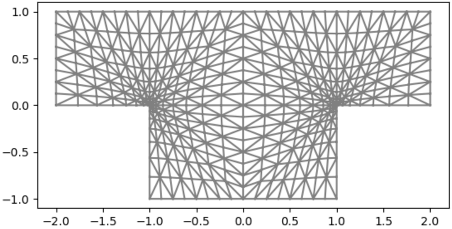
- Unique solution (if the PDE is linear)
- Have guaranteed error estimates (due to Céa's lemma) of the form

$$\|u - U\|_{H^1} < C(u)N^{-\alpha}$$

- Can reduce $C(u)$ and increase α using an adaptive approach.
- But .. Awkward in higher dimensions!

Meshes

Traditional PDE computations using Finite Element Methods use a computational mesh τ comprising mesh points and a mesh topology with $\phi_i(x)$ defined over the mesh



Mesh choice

Accuracy of the computation depends crucially on the choice and shape of the mesh

Mesh needs to be

- Fine Enough to capture (evolving) small scales/singular behaviour
- Coarse Enough to allow practical computations
- Able to resolve local geometry eg. re-entrant corners in non-convex domains
- Can inforce structure preserving elements eg. conservation laws.

Often very hard to find a good mesh, especially in higher dimensions!

The Deep Ritz Method [E + W??]

Lovely Idea!!!

Let $y(x)$ be the output of a parametrised NN

$$y(x) = NN(x)$$

with Y the (nonlinear) set of functions parametrised by θ

Then set

$$y^* = \min_Y F.$$

Allowing for the boundary conditions

- Assume that $y(\mathbf{x})$ has enough regularity for F to be defined at any point x eg. $y(x) \in H^1$
- *ReLU* is OK, but may prefer $ReLU^3$.
- Differentiate $y(\mathbf{x})$ once, exactly using the inbuilt chain rule
- Calculate $F(y)$ by using quadrature at quadrature points $\mathbf{X}_i$ (chosen to be uniformly spaced, or random)
- Train the neural net to minimise a loss function combining F and the boundary and (if needed) initial conditions
- Include known point values if available.

In more detail:

FE: linear

- Limited expressivity, reduced accuracy
- Adaptive only with effort
- Not equivariant
- Need a complex mesh data structure
- Convex with guarantees of uniqueness for many problems (and direct calculation using linear algebra)
- Work on saddle-point problems (eg. most problems)
- Good (a-priori and a-posteriori) guaranteed error bounds :

  Cea's Lemma: Bounds solution error by interpolation error on the FE space

DRM:nonlinear

- Very expressive (potential high accuracy for a small number of degrees of freedom)
- Self adaptive
- Equivariant
- Don't need a complex mesh data structure
- Don't work on saddle-point problems (eg. most problems)
- Don't have Céas Lemma

Comparing a DRM to an adaptive finite element method

If $\sigma(z) = ReLU(z)$ then

$$y(\mathbf{x}) = \sum_{i=0}^{W-1} c_i (a_i \mathbf{x} + \mathbf{b}_i)_+$$

In d-dimensions this is a **piece-wise linear function**
SO .. in principle a ReLU network has the same expressivity as an adaptive Finite Element Method and should deliver the same error estimates if correctly trained.

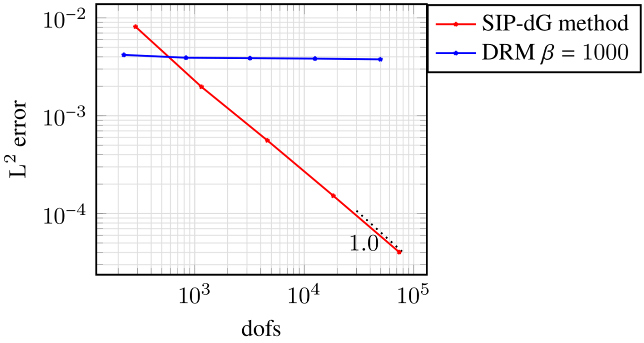
Compare with a traditional linear spline (used in FE) which is often a **piecewise linear Galerkin approximation to a function with a fixed mesh.** Good convergence, but often much slow than an adaptive FE method and hence a well trained PINN

BUT do we ever see this in practice?

Example: Poisson equation in 2D: 1

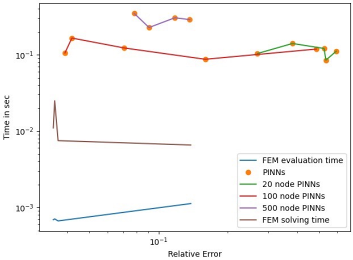
DRM method works well for small DOF

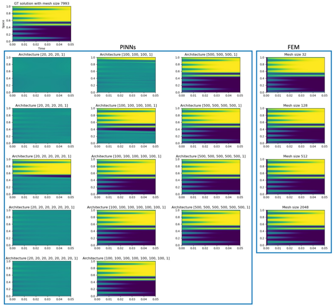
dG Finite Eelement Method is **much** better for more DOF



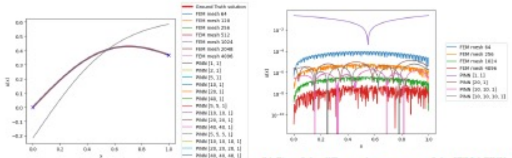
This convergence pattern is seen in many other examples

(Solution of the Allen-Cahn Equations





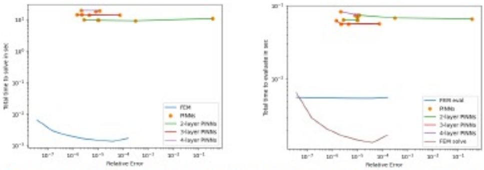
Results by Grossman et. el. 2



(a) Plot of the PDE ground truth solution and the FEM and PINNs solution approximations.

(b) Plot of the difference between some of the FEM / PINNs solution approximations to the ground truth solution on a log scale.

Figure 1: Plot for 1D Poisson equation solution.



(a) Plot of time to solve FEM and train PINN in sec versus ℓ^2 relative error.

(b) Plot of time to interpolate FEM and evaluate PINN in sec versus ℓ^2 relative error.

Figure 2: Plot for 1D Poisson equation of time in sec versus ℓ^2 relative error.

Start of a convergence theory for DRMs

[Jiao, Lai, Lo, Wang, Yang]

- PINN error is a combination of approximation error and training/optimization error
- Show that a PINN (depth L width W) can be constructed with low approximation error which reduces as the complexity of the PINN increases.
- Hope that the optimization error can be reduced to acceptable levels.
- Try things out on simple problems

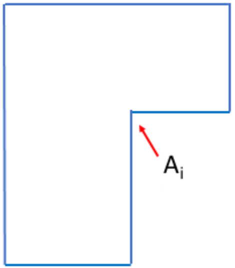
BUT

- **Non convex (no guarantee of uniqueness or convergence of training)**
- PINNs are nonlinear function approximators. No equivalent of Cea's lemma giving a bound on the solution error.
- Solutions can sometimes have no connection to reality!

Example (again) Poisson Problem on an L-shaped domain

Problem to solve:

$$
\begin{aligned}
-\Delta u &= f \text{ in } \Omega \\
u &= u_D \text{ on } \Gamma_D \\
\nabla u \cdot \vec{n}_\Omega &= g \text{ on } \Gamma_N .
\end{aligned}
$$



Singular solution

- Solution $u(\vec{x})$ has a gradient singularity at the interior corner A_i
- If the interior angle is ω and the distance from the corner is r then

$$u(r,\theta) \sim r^{\alpha} f(\theta), \quad \alpha = \frac{\pi}{\omega}$$

  where $f(\theta)$ is a regular function of θ
- Corner problem

$$u(r,\theta) \sim r^{2/3}, \quad r \to 0.$$

Numerical results: **random** quadrature points

Solve $\Delta u(x) = 0$ on $\Omega_L \quad u(r,\theta) = r^{2/3} sin(2\theta/3)$ on $\Gamma = \partial\Omega_L$

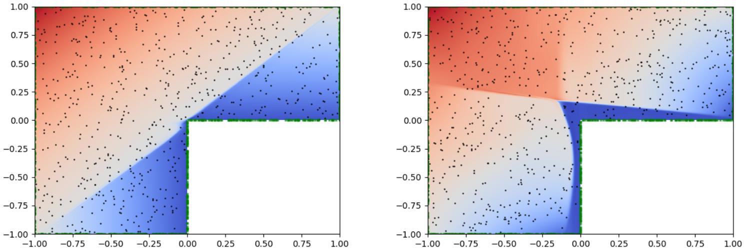


Figure: Left: PINN Right: DRM

Can we improve the accuracy by a better choice of collocation/quadrature points?

Optimal collocation points for the L-shaped domain

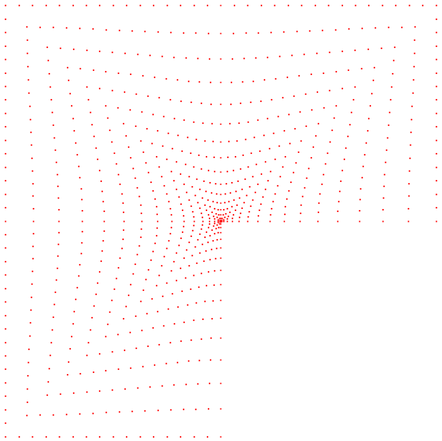


Figure: Optimal points for interpolating $u(r,\theta) \sim r^{2/3}$

Optimal points and PINN/Deep Ritz

Solutions with Optimal quadrature points

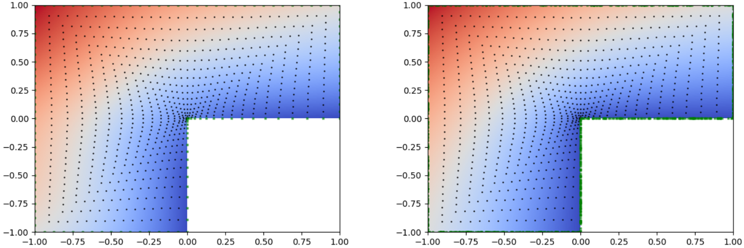


Figure: L^2 error - randomly sampled points: 0.468 | Optimal: 0.0639

Left: PINN, Right: Deep Ritz

Good choice of quadrature points makes a big difference, but still problems with pre-conditioning

Summary

- PINNS and NOs both show promise as a quick way of solving PDEs but have only really been tested on quite simple problems so far
- PINS not (yet) competitive with FE in like-for-like comparisons
- PINNs need careful meta-parameter tuning to work well
- NOs proving more promising. Now used for weather forecasting!
- Long way to go before we understand PINNS or NOs completey and have a satisfactory convergence theory for them in the general case.
- Lots of great stuff to do!