FastVPINNs: A fast, versatile and robust variational PINNs framework for forward and inverse problems in science

Thivin Anandh, Divij Ghose, Sashikumaar Ganesan

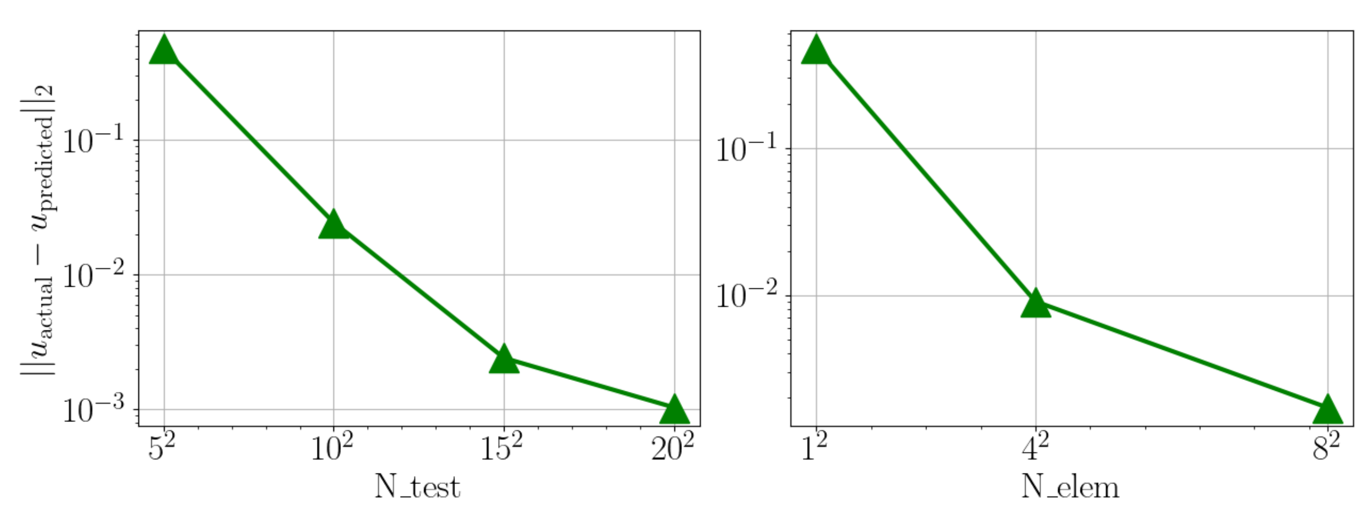
Hp-Variational PINNs

hp-Variational PINNs[1] uses the variational form of the loss function to train a Neural Network. For example, the variational loss formulation of a 2D-Steady state Poisson problem can be represented as

$$\sum_{k=1}^{N_{el}} \left[\int_{\Omega_k} \left(\frac{\partial u}{\partial x}\frac{\partial v_k}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v_k}{\partial y} \right) dx - \int_{\Omega_k} f v_k dx \right] + \tau \sum_{x,y \in \mathbf{B}} [u(x,y) - h(x,y)]^2$$

Advantages of hp-VPINNs

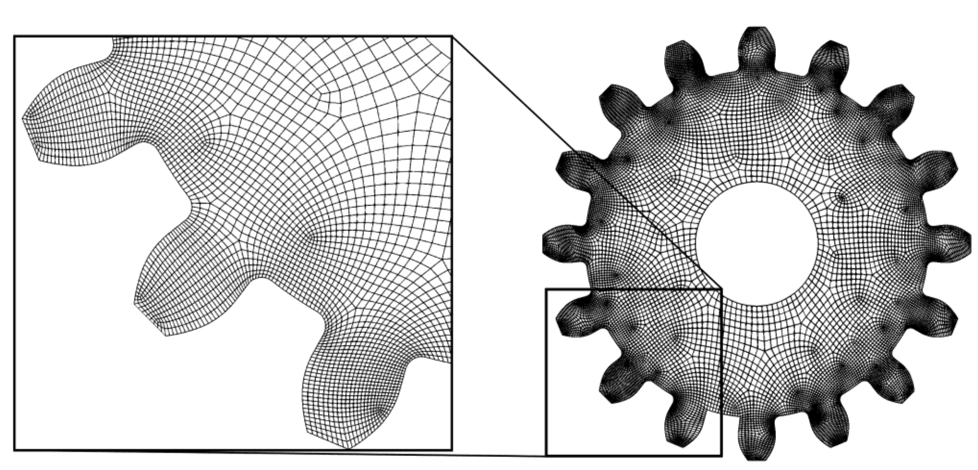
- The use of h- and p- refinement can be used to increase the accuracy of the solution
- The h-refinement results in restricting the shape functions to a smaller region, which enables the NN to capture high frequency solutions



Challenges in Existing implementation of hp-VPINNs

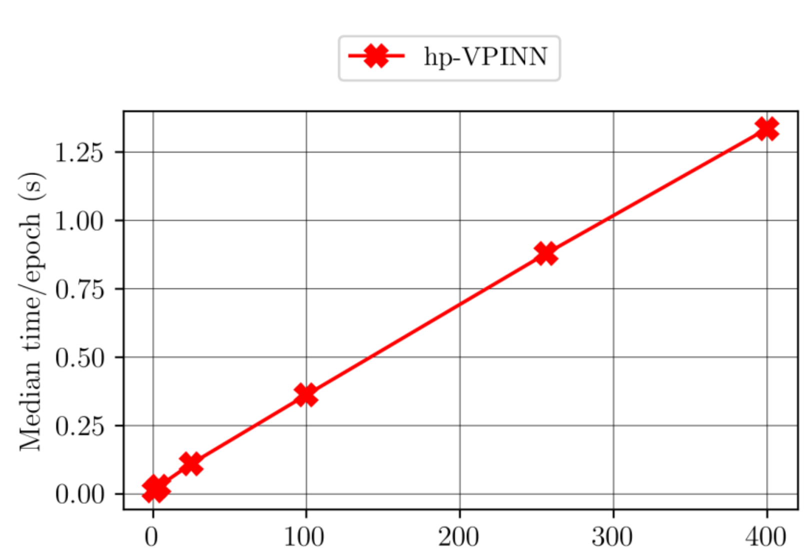
Handling Complex Geometries

- The Current implementation can handle only structured quadrilateral cells. This is because the transformation from actual cell to reference cell can be achieved by multiplying with a constant Jacobian value



Increased Execution Time with increase in number of cells

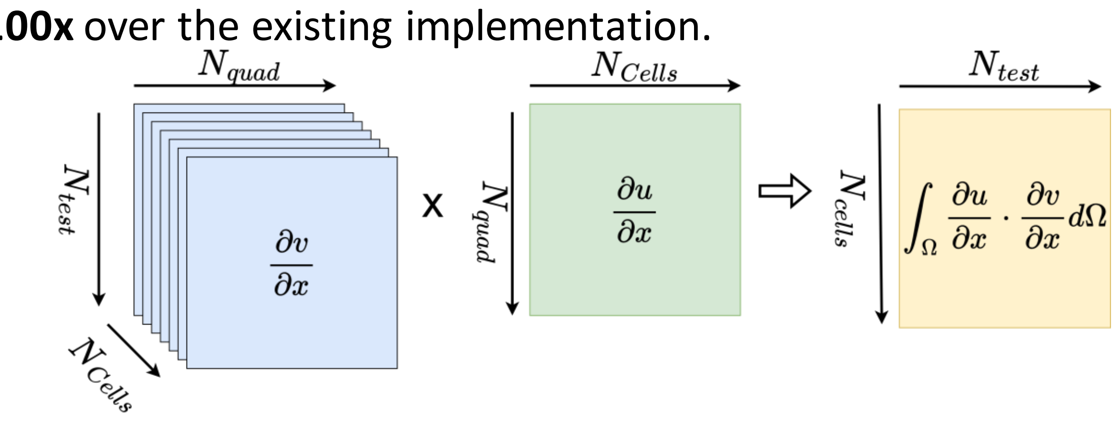
- The current implementation scales linearly with the number of elements within the domain, which makes this method computationally expensive for training domains with large number of elements.



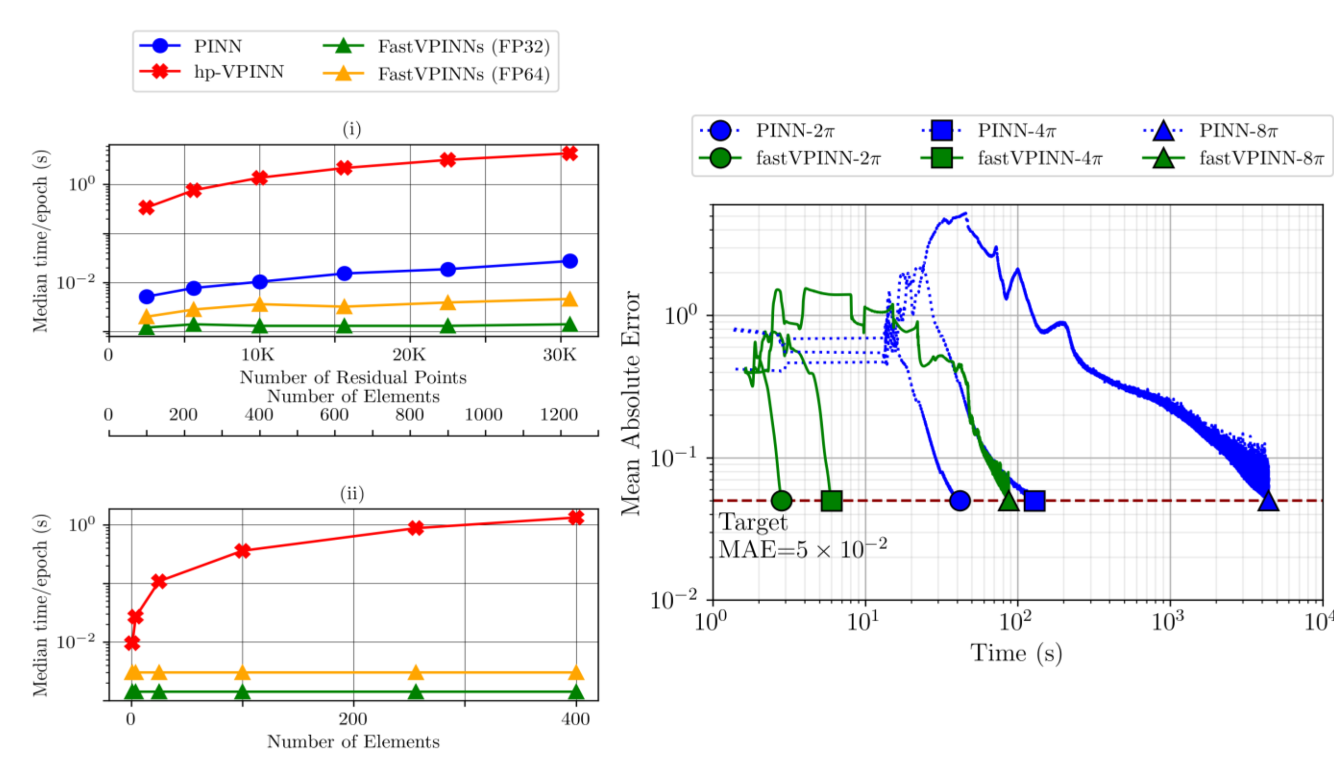
References: [1] hp-VPINNs: Ehsan Kharazmi et. al.

FastVPINNs – A Tensor based approach

- We address both of the existing challenges in our proposed framework FastVPINNs, which uses bilinear transformation to handle skewed quadrilateral cells and uses a novel tensor-based loss calculation which eliminates cell looping and there by resulting in a speedup of **100x** over the existing implementation.



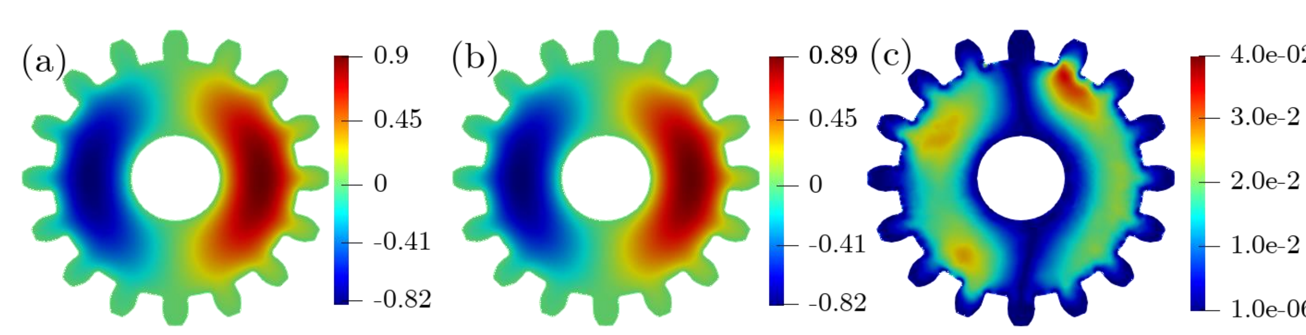
Performance of FastVPINNs framework



FastVPINNs on Complex Geometries

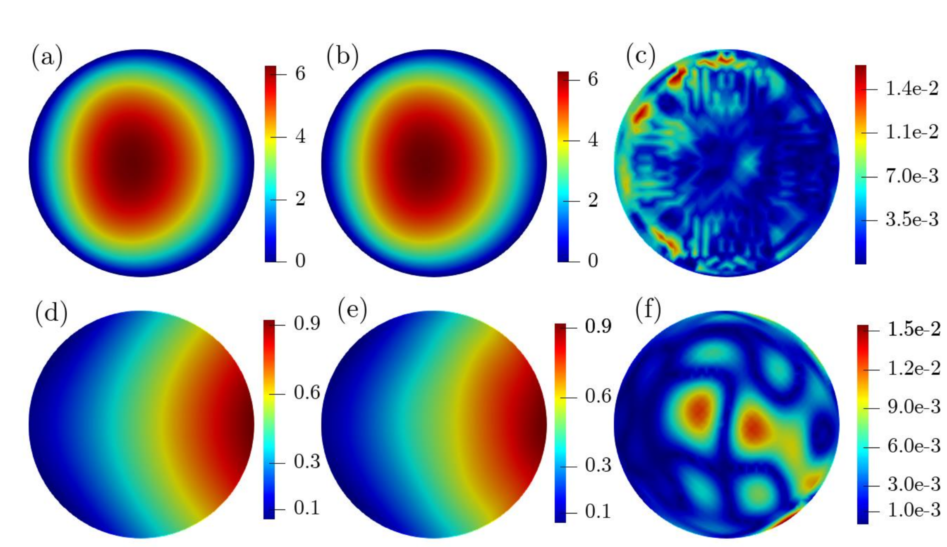
Forward Problems (CD2D)

- The gear mesh has close to 14,192 cells and solved with 350k quadrature points



Inverse Problems

- The inverse problem in trying to estimate the spatially varying diffusion parameter was solved on a complex mesh with 1024 cells
- FastVPINNs was able to complete a training of 100k iteration within 200 seconds (18 ms per iteration)



ArXiV version of the full paper can be found at : https://arxiv.org/abs/2404.12063