Multilevel Multiplicative MLORAS

1 Two-level ORAS

We are trying to obtain two-level ORAS methods by learning interface values and the prolongation operator.

$$C = R_0^T \left(R_0 A R_0^T\right)^{-1} R_0 \tag{1}$$

$$\text{Error propagation operator}: \quad M_{\text{2-ORAS}} = (I - CA)(I - M_{\text{1-ORAS}} A) \tag{2}$$

1.1 New Loss Function

For the NeurIPS paper, we used the following loss function:

$$Y_K^{(\theta)} = \left\{ \left\| \left(T^{(\theta)}\right)^K x_1 \right\|, \left\| \left(T^{(\theta)}\right)^K x_2 \right\|, \ldots, \left\| \left(T^{(\theta)}\right)^K x_m \right\| \right\}, \tag{3}$$

$$\mathcal{L}^{(\theta)} = \max(Y^{(\theta)}). \tag{4}$$

As we will show in the results section, this loss function does not perform as we expect. Instead we consider $(Y_i^{(\theta)})^{\frac{1}{i}}$ for all i's from 1 to K, and we get the new loss function as follows:

$$Z^{(\theta)} = \left(\max((Y_1^{(\theta)})^1), \max((Y_2^{(\theta)})^{\frac{1}{2}}), ..., \max((Y_K^{(\theta)})^{\frac{1}{K}})\right). \tag{5}$$

$$\mathcal{L}^{(\theta)} = \langle \text{softmax}(Z^{(\theta)}) \cdot Z^{(\theta)} \rangle. \tag{6}$$

We use $K = 10$ for our experiments.

1.2 Learning Denser R_0

For learning R_0, one option is to just use the sparsity of the R_0 we get from RAS (with overlap and partition of unity). As we show in the results, that does not perform as we expect. On the other hand, we can learn denser R_0 by connecting each coarse node not only to its underlying subdomain node, but also its neighbors' subdomain nodes (Figure 1)

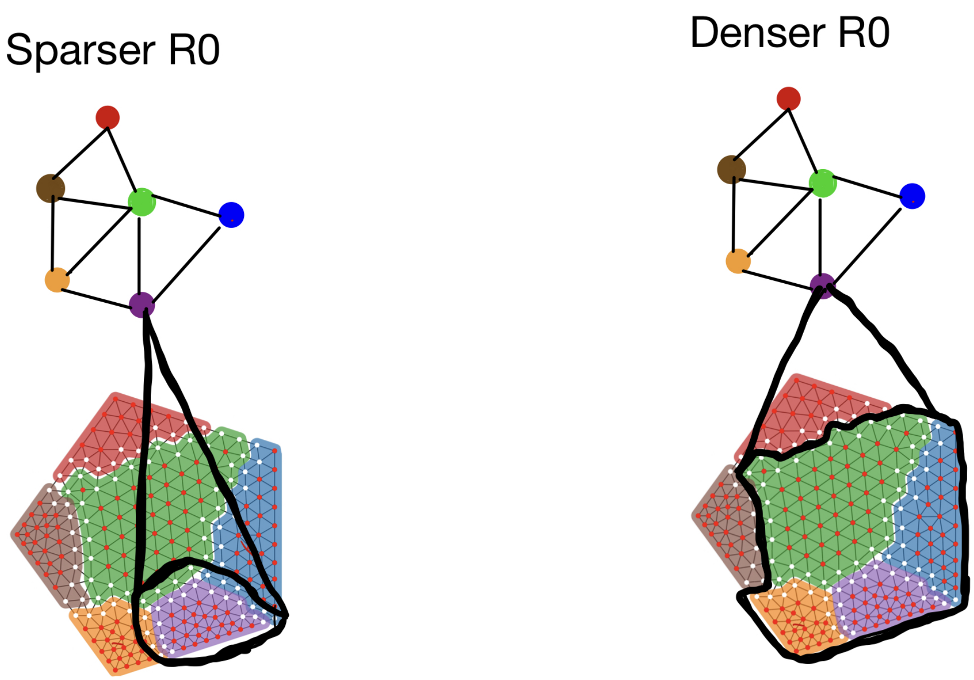

Figure 1: Learning denser R_0 within the HGNN architecture.

1.3 Training

The model that works best has the label "Base" in the figures, and it uses two layers of HGNN with TAGConvs. As apparent from its name, it uses the denser R_0 from the previous subsection. Moreover, the row sum of the learned prolongation operator is equal to one since we found that it is necessary for desired performance. We also train this model with the new loss function 6. To ensure all the details mentioned in the previous subsections are necessary, we compare "Base" with the three variants, each of which is only different on one aspect from "Base":

- Base: Described in the paragraph above.
- Un-normalized R_0: The prolongation operator does not have the restriction of row sum = 1
- Sparser R_0: The prolongation operator has the same sparsity as RAS R_0
- Old loss: Trained with the old loss function 4.
- P-loss: Inspired by optimal interpolation paper [1], adding $\gamma \sum_{i \in \text{columns}} ||R_{0_i}||_A^2$ to the loss function

We train all the models on 100 grids of sizes 800-999 nodes, with batch size of 10, for 30 epochs, and with learning rate of 10^{-4}.

1.4 Stationary Results

Figure 2 shows the performance of the stationary algorithm of the models as well as that of RAS.

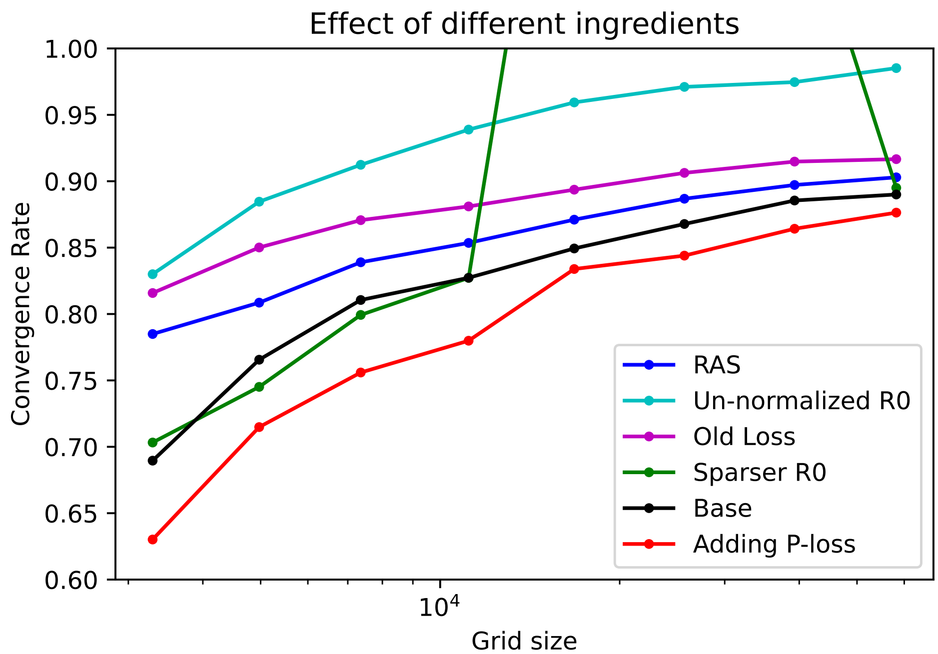

Figure 2: Stationary algorithm performance of different models and RAS.

1.5 Ablation study

We compare Hierarchical GNN model (used in above results) with Graph U-Nets. The training loss is shown in Figure 3, and the performance of the models on the test data set is shown in Figure 4.

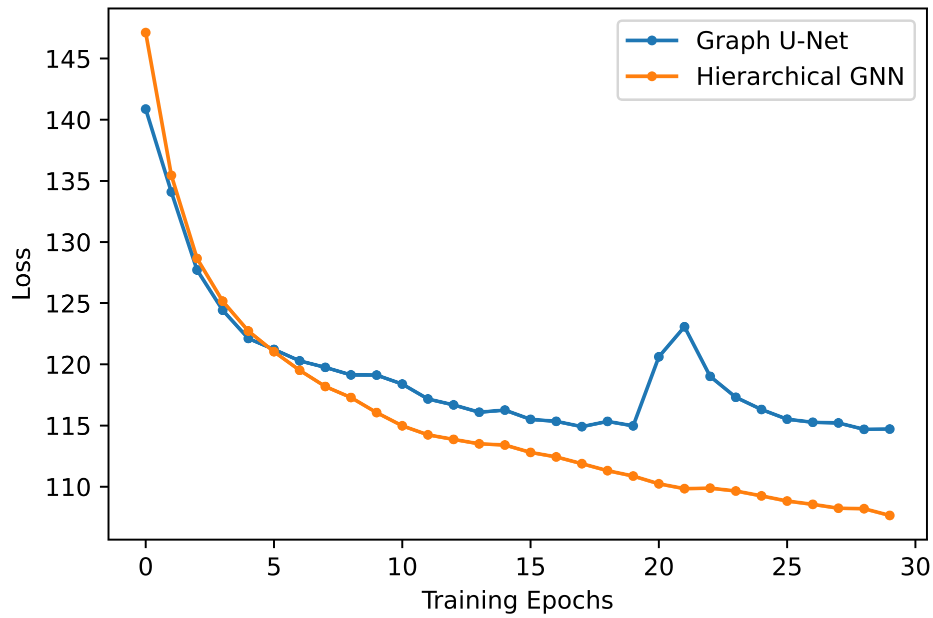

Figure 3: Training loss of Hierarchical GNN vs. Graph U-Net.

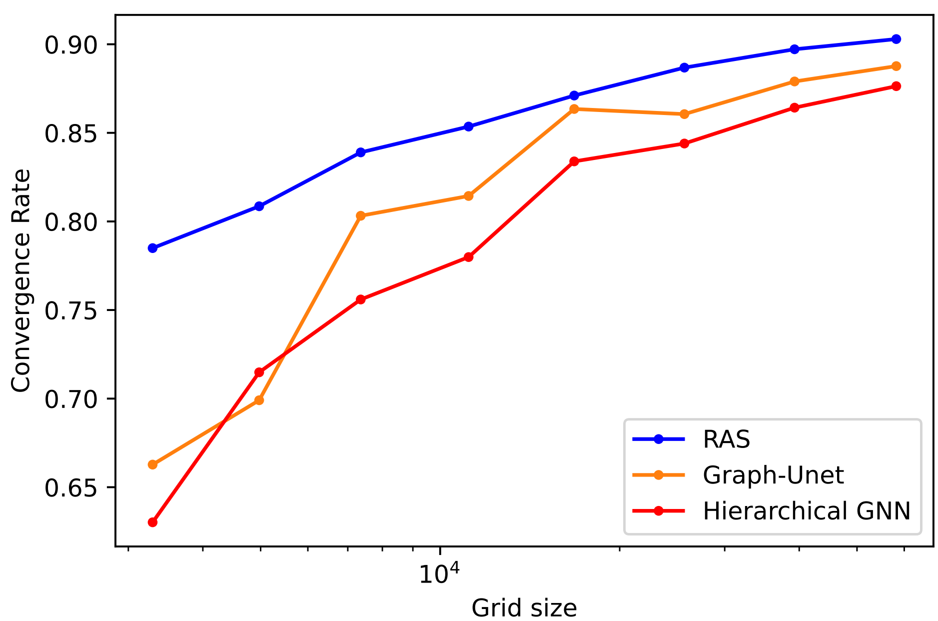

Figure 4: Test results of Hierarchical GNN vs. Graph U-Net.

1.6 Next step

- Moving to 3D/anisotropic problems?
- Still don't know how to make it work better than RAS as a preconditioner for FGMRES
- Try ASAP pool as another GNN baseline (does it have node assignment guarantees?)

References

[1] Luke N Olson, Jacob B Schroder, and Raymond S Tuminaro. A general interpolation strategy for algebraic multigrid using energy minimization. *SIAM Journal on Scientific Computing*, 33(2):966–991, 2011.