Mixture manifold model for inverse regression problems

Karthik Ramachandran, Simiao Ren, Jordan Malof, Ph.D. – Duke University, Department of Electrical and Computer Engineering

Abstract

In this project I look to build upon existing research surrounding tandem models as a means of solving ill-posed inverse machine learning problems that look to identify potential x inputs that correspond to a given y output. These inverse problems present several challenges. Primarily, x solutions can be non-unique, meaning that multiple x inputs result in the same y output. Standard regression methods converge to the average of these correct solutions, producing incorrect outputs. Additionally, standard regression techniques fail to preserve domain boundaries when going from the y- to the x-space. First, a forward model is trained in the x to y direction. Then, a novel mixture manifold model is used to train inverse models in the y to x direction, using the forward model and a boundary loss term to measure performance, and, for each y, select the best x proposed by the inverse models. In later experiments, a repulsion term is added to the loss calculation to force learning of new manifolds.

Tandem Models and Inverse Solutions

The goal of a tandem model is to reliably learn a single set of x inputs that correspond to a set of y-values [1]. Otherwise, training a backwards model directly on a dataset causes the model to become affected by single y-values that have multiple x-values, causing convergence to some mean x that doesn't correspond to the same y. An additional boundary term pushes the backwards model outputs to a defined domain to ensure only valid solutions are generated. The below figure illustrate the training and inference process of the tandem model, as well as the loss used during training.

$$L = \left\|\hat{f}(\hat{g}^2) - s\right\|_2^2 + 100 * ReLU(|\hat{x} - \mu_x| - \frac{1}{2}R_x)$$

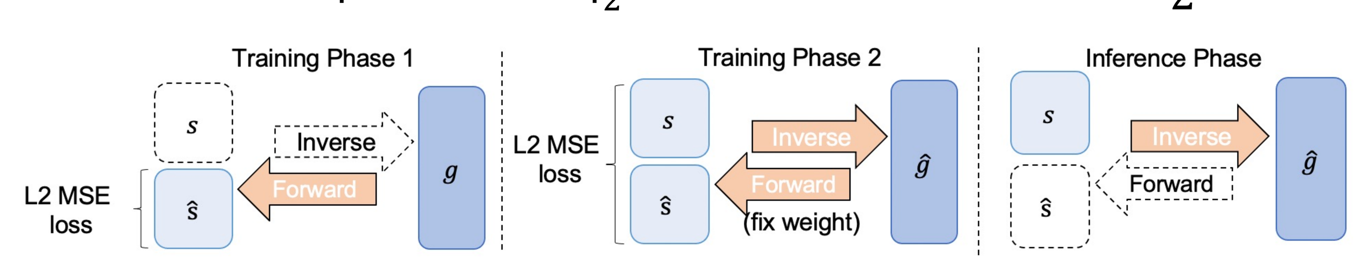

Figure 1: The loss (upper) used during phase 2 of a tandem model's training, with detailed steps shown (lower; adapted from [2])

Combination of Tandem Models

Each tandem model learns a single solution set. So, training multiple tandem models should result in learning several x solutions for each y. To select a single x, the below formula is used.

$$\hat{x} = argmin_{\hat{x}^m} \left\|\hat{f}(\hat{x}^m) - y\right\|_2^2$$

Figure 2: The selection process for the network's output at inference time.

Experimental Design

A forward model will be trained on 2-dimensional sinusoidal data of the form $y = \cos(3\pi x) + \sin(3\pi x)$. Then, k backwards networks will be trained, with k ranging from 1 to 4. This will be repeated across 10 trials and on a robotic arm dataset with a 4-dimensional input and 2-dimensional output. Next, a repulsion term will be added to force learning of new manifolds. The learned manifolds and model performances are plotted in the images below and to the right.

Mixture Tandem Model Visualization in X-Space

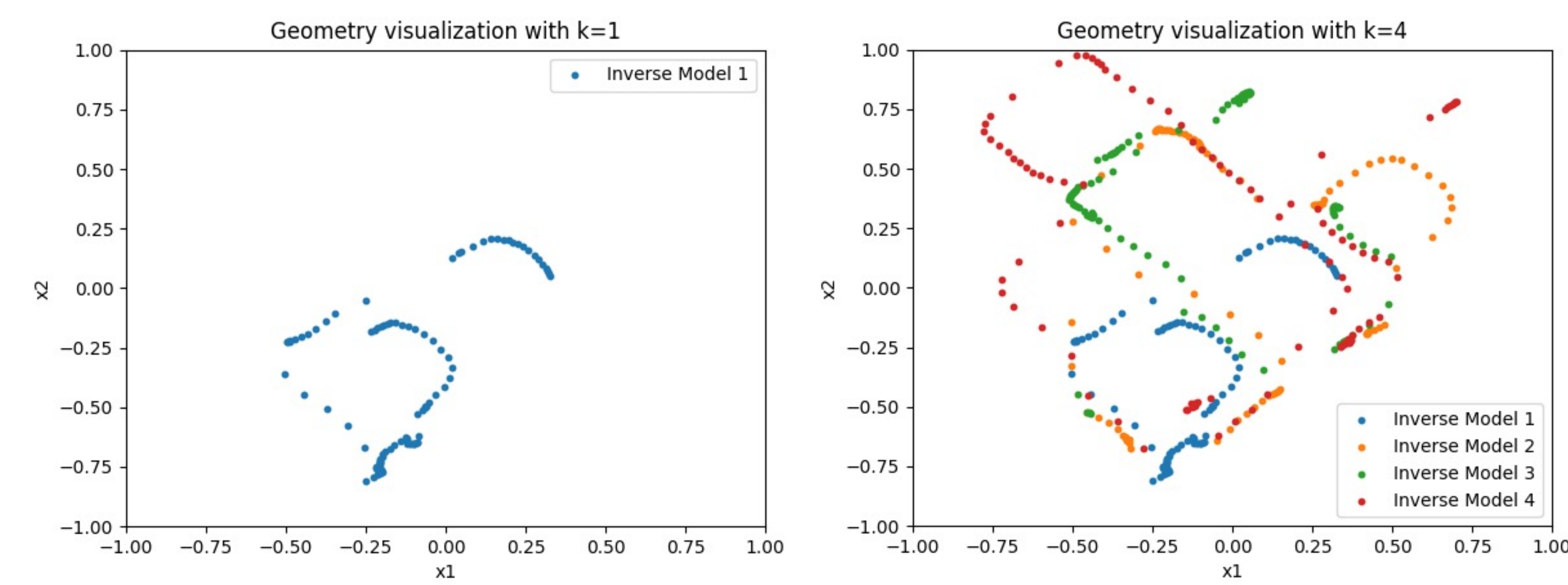

Figure 3: Sine wave geometries learned by tandem model with k=1 (left) and k=4 (right)

Mean Squared Error in Y-Space

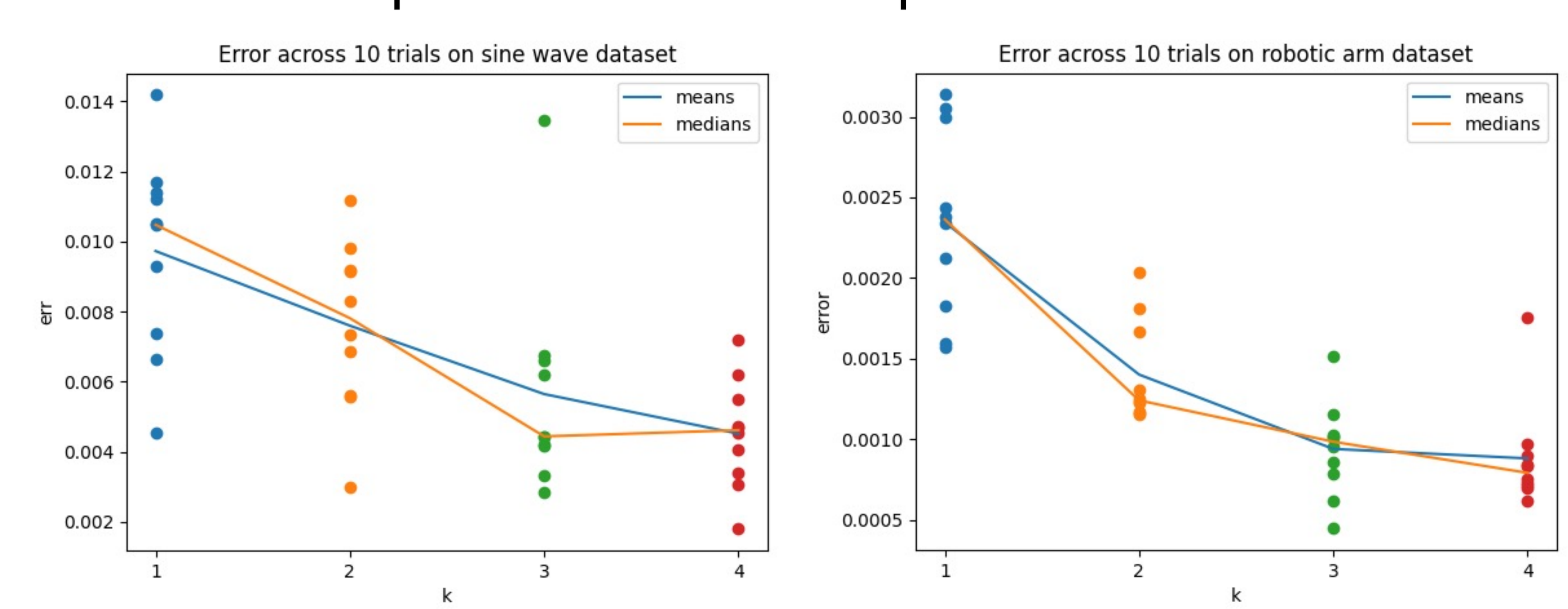

Figure 4: Model performance by k for sine wave (left) and robotic arm (right) problems

Additional Repulsion Loss to Encourage Diversity

To push the model away from already-learned manifolds, the below term is added to the loss. This shows the repulsion loss for a k=2 network; as k increases, an additional term is added for each previous manifold.

$$L_{repulsion} = -\lambda\left(1 - \exp\left(-\frac{\left\|\hat{g}^2(s) - \hat{g}^1(s)\right\|_2^2}{\sigma^2}\right)\right)$$

Figure 5: The repulsion loss term for k > 1

Ablation Visualization for Repulsion Loss

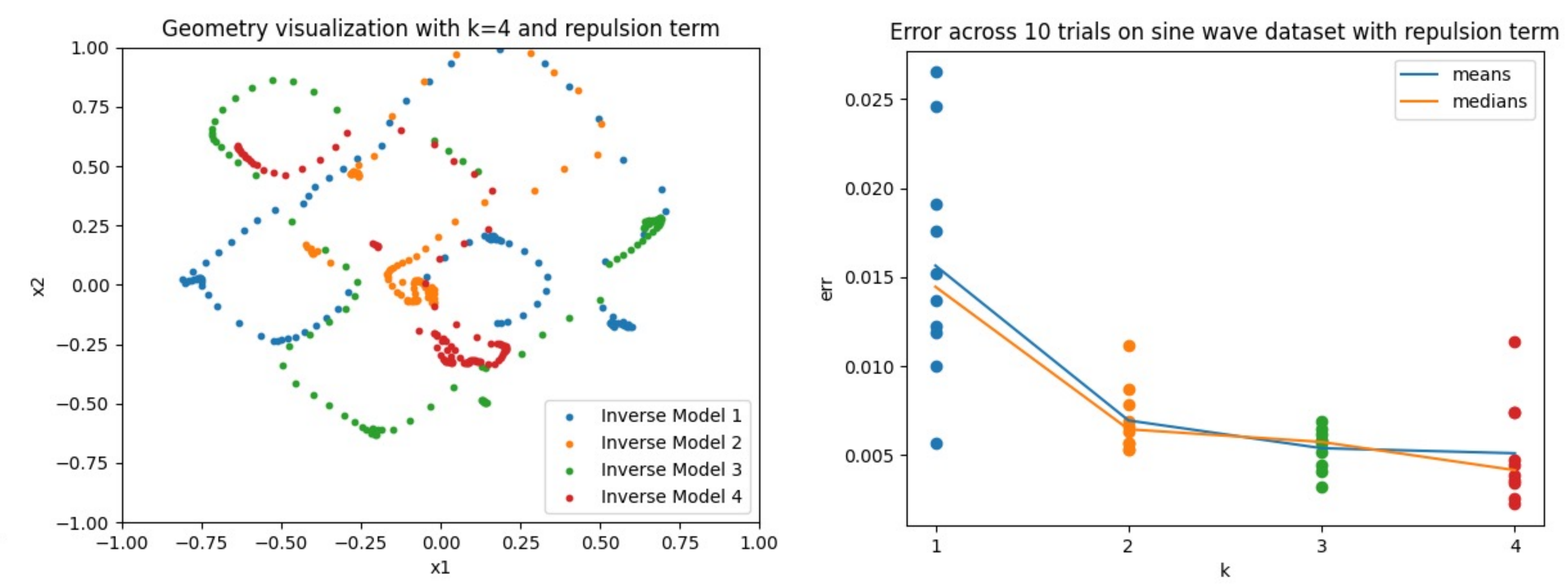

Figure 6: Repulsion term on k=4 network results in larger set of solutions that more distinctly map out multiple manifolds (left). This comes at no cost to overall model performance even as k increases (right).

Discussion and next steps

Using a combination of tandem networks clearly improves model performance across multiple datasets, with diminishing returns being seen for k > 3. The addition of a repulsion term successfully forces the model to learn additional valid x-solutions for single y-values without increasing error.

Immediate next steps include applying this approach to increasingly complex ill-posed problems and further understanding the effect to hyperparameters λ and σ^2 in the repulsion loss term. One additional avenue for exploration includes using new data synthesized by the forward model to train the backwards networks rather than using the same dataset to train the forward and backwards models.

This research has promise for applications that require quick inverse learning of multi-dimensional problems, including metamaterial discovery using known spectra. Further understanding of the inference speed of these solutions relative to existing methods, like Neural Adjoint [2], is also required.

References and code

[1] D. Liu, Y. Tan, E. Khoram, and Z. Yu, "Training deep neural networks for the inverse design of nanophotonic structures," ACS Photonics, vol. 5, no. 4, pp. 1365–1369, 2018.

[2] Ren, Simiao & Padilla, Willie & Malof, Jordan. (2020). Benchmarking deep inverse models over time, and the neural-adjoint method.

All code can be found at github.com/karthikr13/Tandem