Experiment Result: Two Phase Flow

A Preprint

Jayjay, Tuna, Jason, Richard

2024-09-30

We now evaluate surrogate models in two different criteria, forward simulation and inverse problem.

1 Pipeline

- **FNO-NF.jl**: create two-phase flow dataset, eigenvector of FIM, and vJp
  - Now we differentiate each time step saturation, $S^1(K), \cdots S^8(K)$ with respect to K
  - Rather than differentiating $\{S^t(K)\}_{t=1}^8$ with respect to K and repeating it the 8 times.
- **Diff_MultiPhysics**: train (written in pytorch) and posterior estimation

2 Updates on training scheme: respecting the time dynamics of GCS PDE Equation

Before discussing what steps I took to compute $\tilde{K}$, our MLE estimate, I want to briefly go over new training scheme we tried.

Setting	Previous Experiment	Updated Experiment
Dataset	2000 pairs of $\{K, S^t(K)\}_{t=1}^8$ Train/Test split: [1800, 200]	1000 pairs of $\{K, S^t(K)\}_{t=1}^8$ Train/Test split: [800, 200]
FIM	Number of observations = 10 Number of eigenvectors = 1 For a single pair of datapoints, 1 FIM is obtained. And we repeat it for 8 times.	Number of observations = 2 Number of eigenvectors = 1 For a single pair of datapoints, 8 FIMs are obtained (for 8 time steps).
Likelihood	Difference between perturbed and true time series $\{S^t(K)\}_{t=1}^8$	Difference between perturbed and true single time step Saturation for instance, $S^1(K), \cdots S^8(K)$
Hyperparameter	Batch size = 100	Batch size = 100

Now, for the sake of clarity, I am going to call:

- eigenvector obtained from the full time series (or across all time steps), **static eigenvector** as it does not evolve over time.
- eigenvector obtained from each time step, **dynamic eigenvector** as it reflects how the system's dynamics evolve.

In case we need to recall how we computed FIM..

$$\{X_i\}_{i=1}^N \sim p_X(X),\ \epsilon \sim \mathcal{N}(0, \Sigma),\ \Sigma = I$$

For a single data pair, we generate multiple observations.

$$Y_{i,J} = F(X_i) + \epsilon_{i,J}, \quad where \left\{\epsilon_{i,J}\right\}_{i,J=1,1}^{N,M}$$

As we assumed Gaussian, we define likelihood as following.

$$p(Y_{i,J}|X_i) = e^{-\frac{1}{2}\|Y_{i,J}-F(X_i)\|_2^2}$$

$$log\, p(Y_{i,J}|X_i) \approx \frac{1}{\Sigma}\|Y_{i,J} - F(X_i)\|_2^2$$

A FIM for a single data pair i is:

$$FIM_i = \mathbb{E}_{Y_{i,\{J\}_{i=1}^m} \sim p(Y_{i,J}|X_i)} \left[\left(\nabla log\, p(Y_{i,J}|X_i)\right) \left(\nabla log\, p(Y_{i,J}|X_i)\right)^T \right]$$

3 Forward Simulation

So, now we compare how the learning becomes different when compared with

- that of static eigenvector
- that of dynamic eigenvector, respecting the time dynamics of GCS PDE equation.

Like before, we evaluate the training result of PBI model:

1. Loss behavior
2. Forward simulation
3. Inversion

3.1 How does changed eigenvector look like?

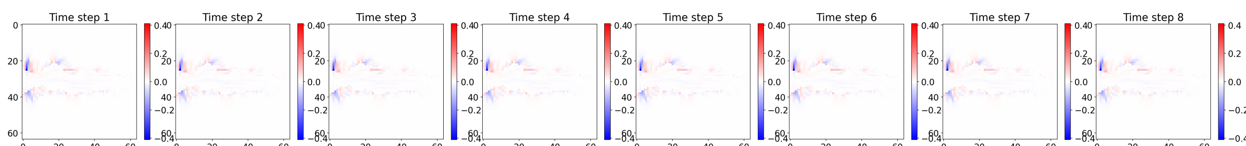


Figure 1: Static eigenvector

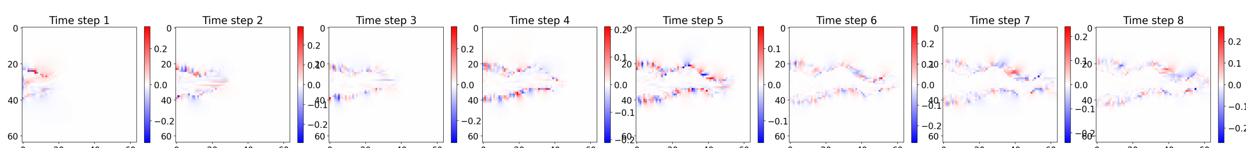
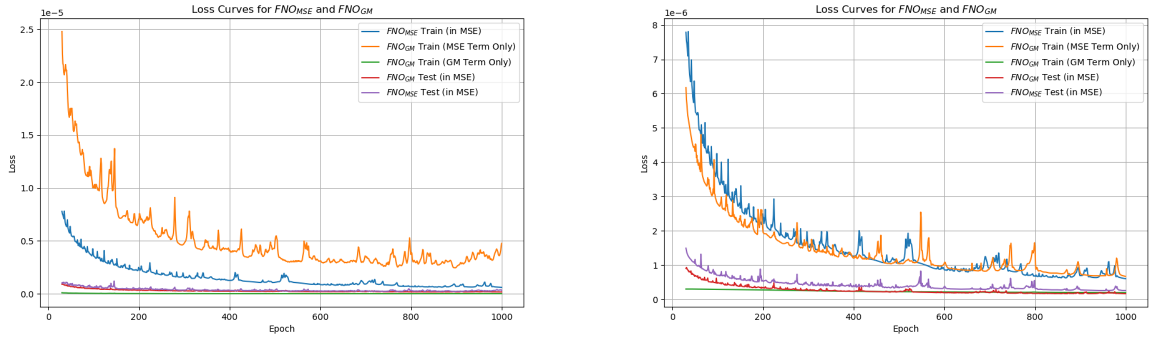


Figure 2: Dynamic eigenvector

3.2 How does it impact training?

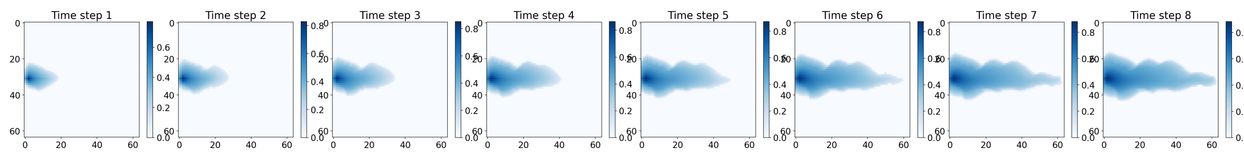
When we look at the test loss, we observe that unlike static model, dynamic model's test curve is always lower than that of MSE model.

	Epochs	λ	Train Loss	Test Loss
			MSE/GM	MSE
FNO-MSE	1000	N.A.	6.5207×10^{-8}	1.3088×10^{-7}
FNO-PBI	1000	1.0	8.3925×10^{-8}	1.3030×10^{-7}

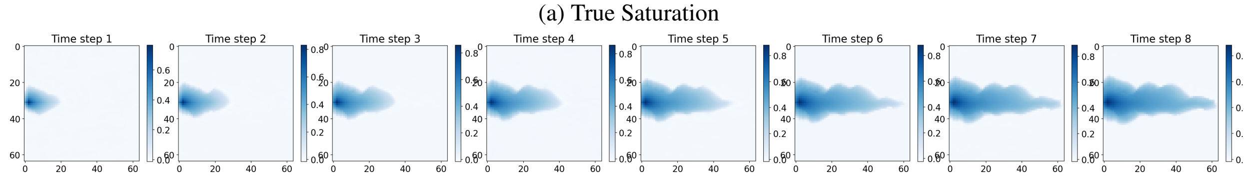


(a) Loss (static) (b) Loss (dynamic)

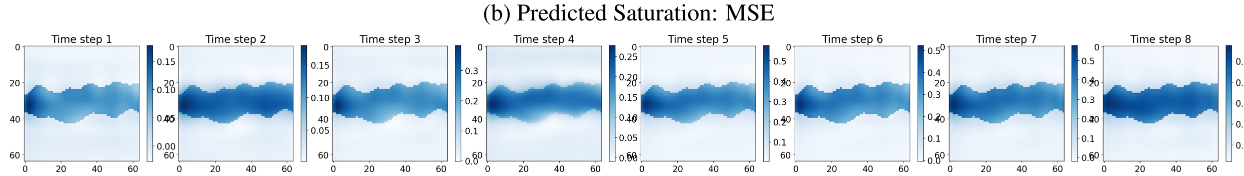
Figure 3: Loss plot static vs dynamic



(a) True Saturation



(b) Predicted Saturation: MSE



(c) Predicted Saturation: PBI (dynamic)

Figure 4: Example of Forward Prediction

3.3 Forward Simulation on Test dataset

3.4 Sanity check: how does vJp of MSE model and PBI model look like?

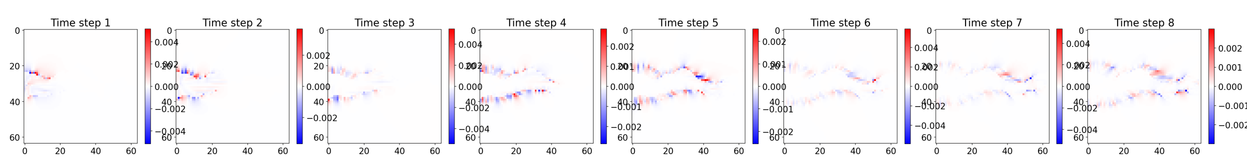


Figure 6: Dynamic vJp (True)

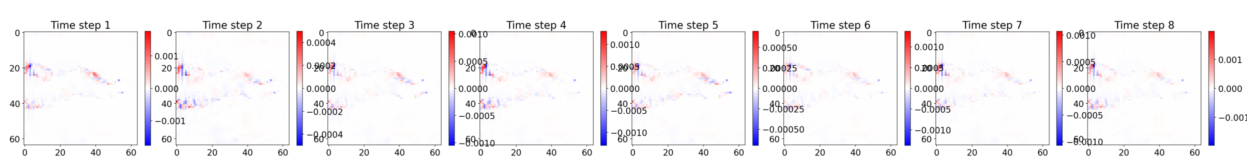
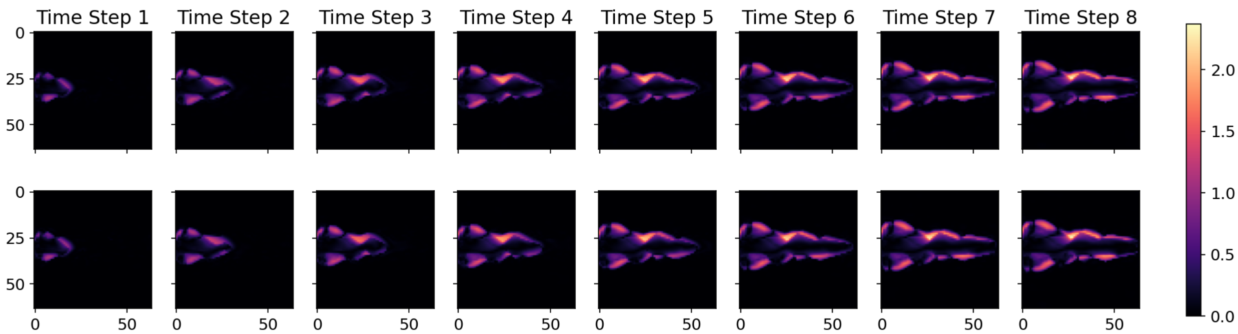
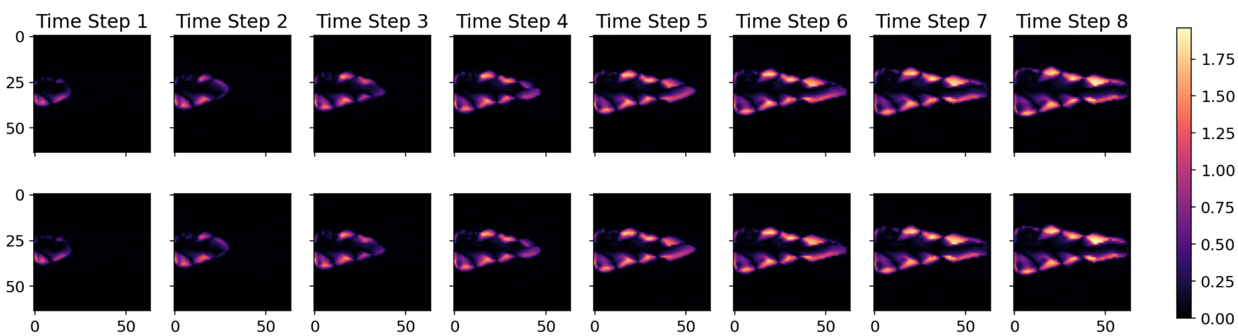


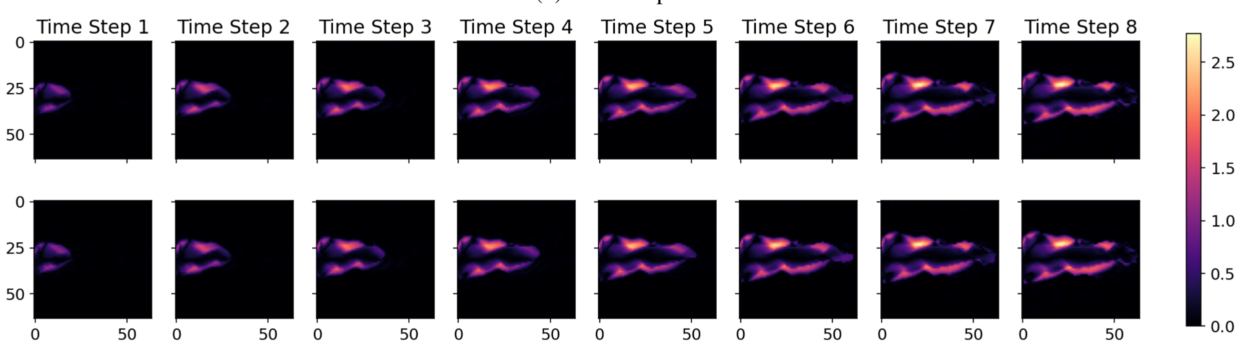
Figure 7: Learned vJp (MSE)



(a) Test Sample 1



(b) Test Sample 2



(c) Test Sample 3

Figure 5: Absolute Difference (x 5) plot of test samples

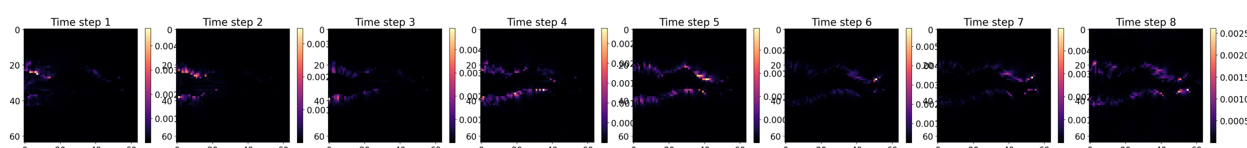


Figure 8: Abs Diff in vJp (MSE)

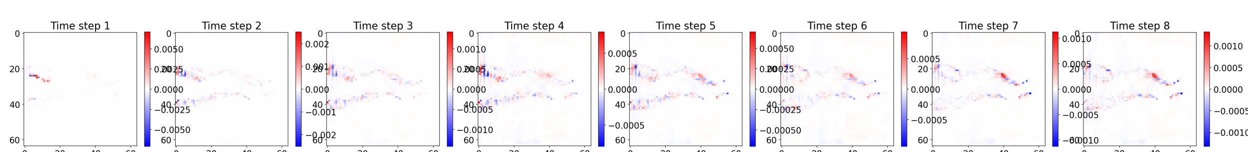


Figure 9: Learned vJp (PBI)

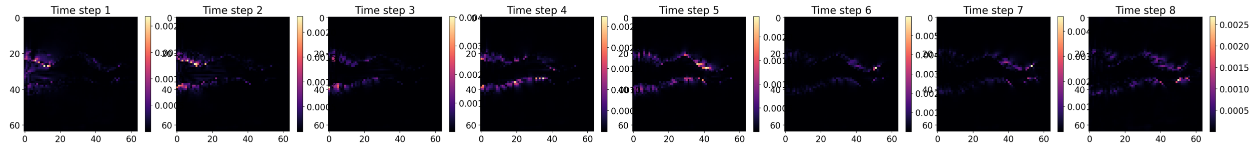


Figure 10: Abs Diff in vJp (PBI)

4 Inverse

Previously, we showed MLE estimate of $\tilde{K}$.

- The inversion result looked too good to be true.
- This is because initial K_0 is unperturbed true K, so there was nothing to optimize upon.
- So now we perturbed K_0 like Francis did.

4.1 Setting

We wanted to evaluate surrogate model's performace in MLE/posterior estimation quickly, so for now, we kept inversion method as simple as possible. (least squares method)

$$min_K \|S_\theta(K) - S(K)\|_2^2$$

where:

- $K_0 = H(K)$
- S_θ: Neural Network model

- We obtain 100 $\{S^t(K)\}_{t=1}^8$ from test data.
- We generate $H(K_0)$ by averaging over all K_0 where H is observation operator.

Now we look at two different cases:

1. static
2. dynamic

4.2 Loss

With dynamic eigenvector, the loss during inversion falls under that of MSE model.

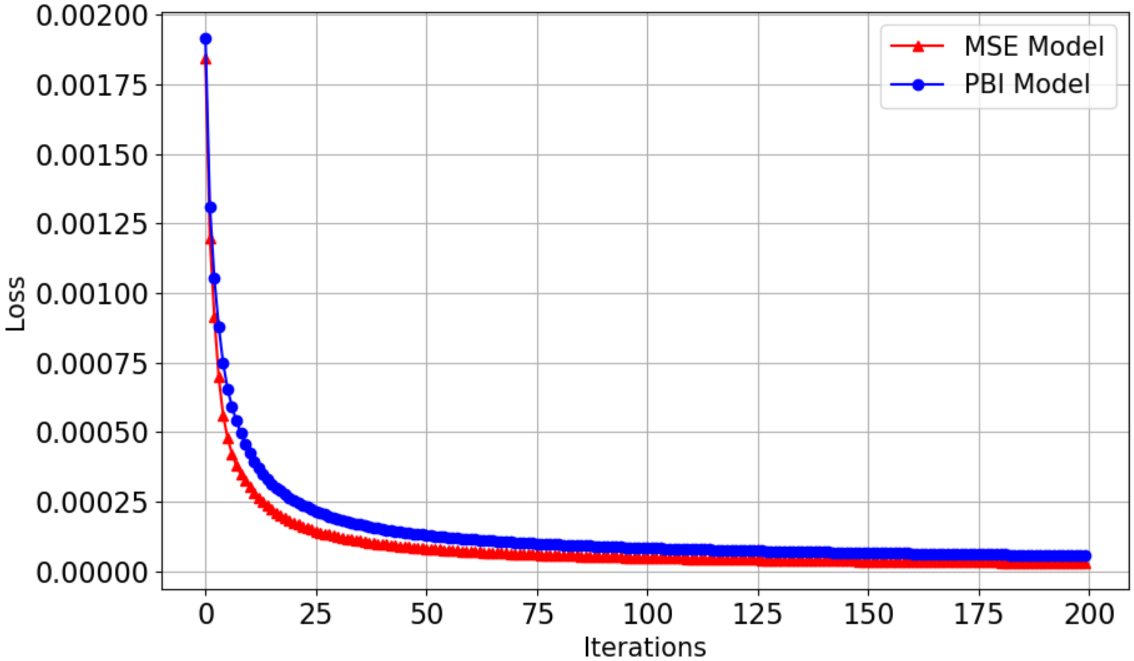


Figure 11: Loss (static)

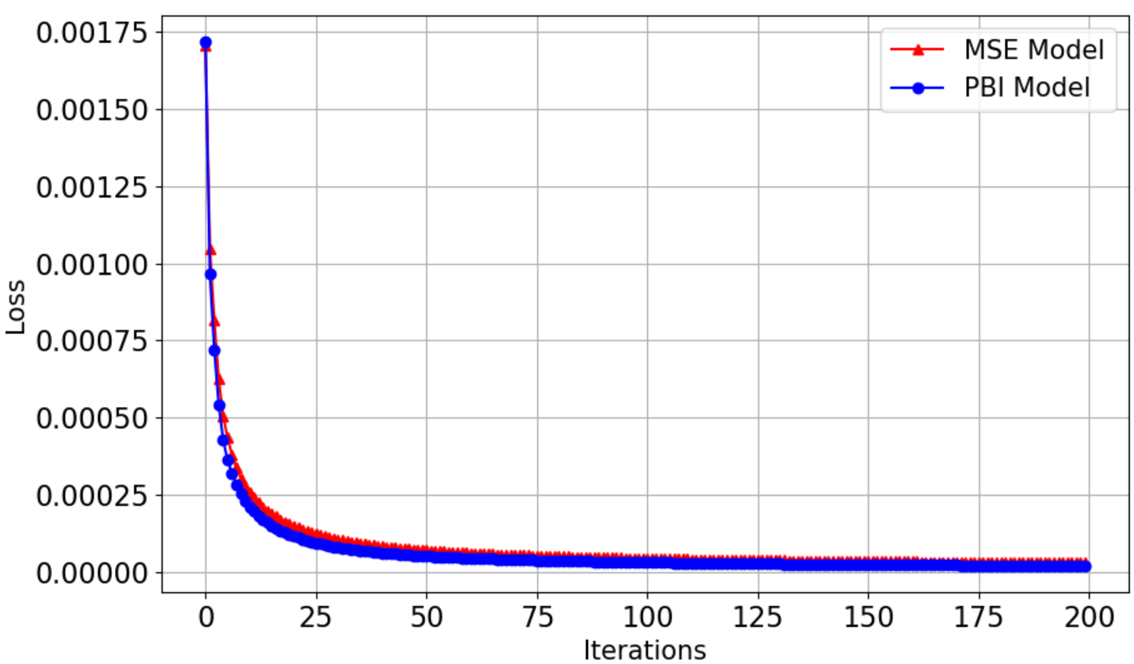


Figure 12: Loss (dynamic)

4.3 Ablation test: finding optimal lambda and number of epoch

4.3.1 Choosing the best parameters for MLE optimization

We conduct hyperparameter search for the λ. The number of epochs chosen were based on the loss plot convergence. If it converged, we stopped training.

This is unconstrained.

4.3.1.1 Unconstrained (static)

	Epochs	λ	Loss (MSE)	SSIM
FNO-PBI	400	20.0	8.9021×10^{-5}	0.5550
FNO-PBI	300	50.0	6.1867×10^{-5}	0.5555
FNO-PBI	200	100.0	5.5757×10^{-5}	0.5709
FNO-MSE	400	20.0	5.3901×10^{-5}	0.5235
FNO-MSE	300	50.0	3.4602×10^{-5}	0.5428
FNO-MSE	200	100.0	2.9429×10^{-5}	0.5564

4.3.2 Updated Result

Out of all 100 test samples, I brought some interesting cases. Some looks good, some looks questionable. (Test sample 3, 5). Does SSIM values make sense here?

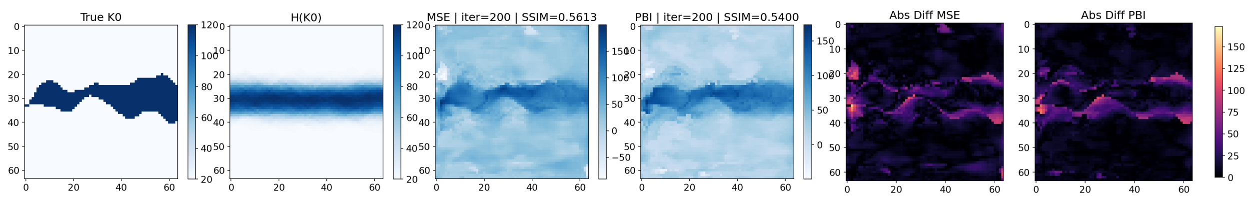


Figure 13: Test sample 1 (static)

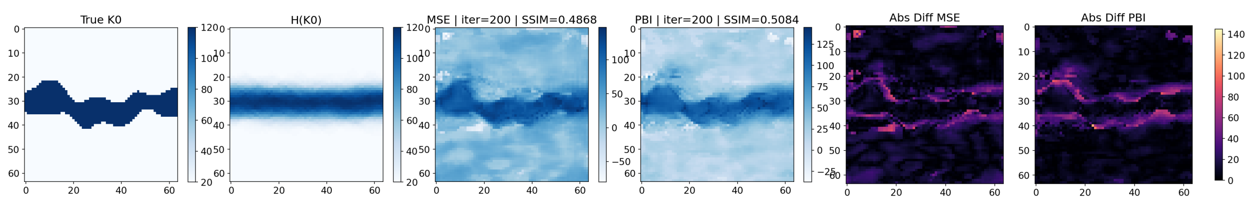


Figure 14: Test sample 2 (static)

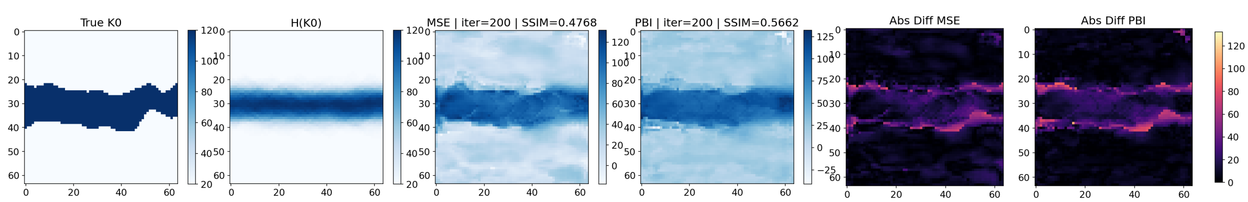


Figure 15: Test sample 3 (static)

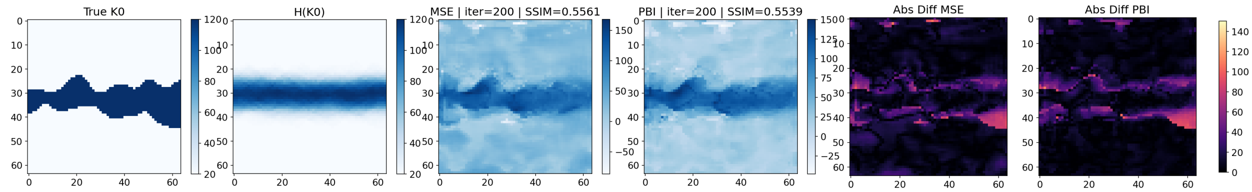


Figure 16: Test sample 4 (static)

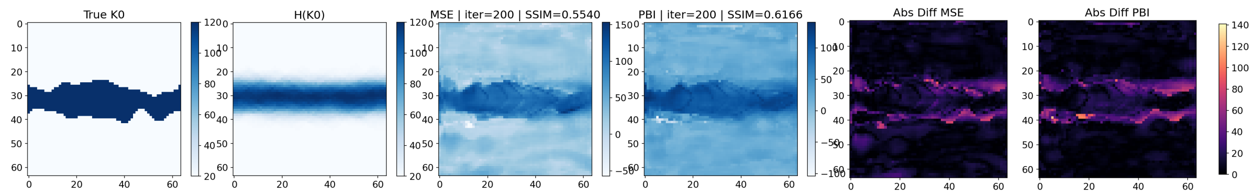


Figure 17: Test sample 5 (static)

4.3.3 What other things can be evaluated in terms of forward simulation?

1. **Stability**: predict longer saturation evolution 9th to 16th.
2. **Generalization**: test with out of distribution test samples.
3. **Towards learning true governing PDE equation**: One step prediction rather than multi-step prediction

- Current one is time discretized.

5 Conclusion

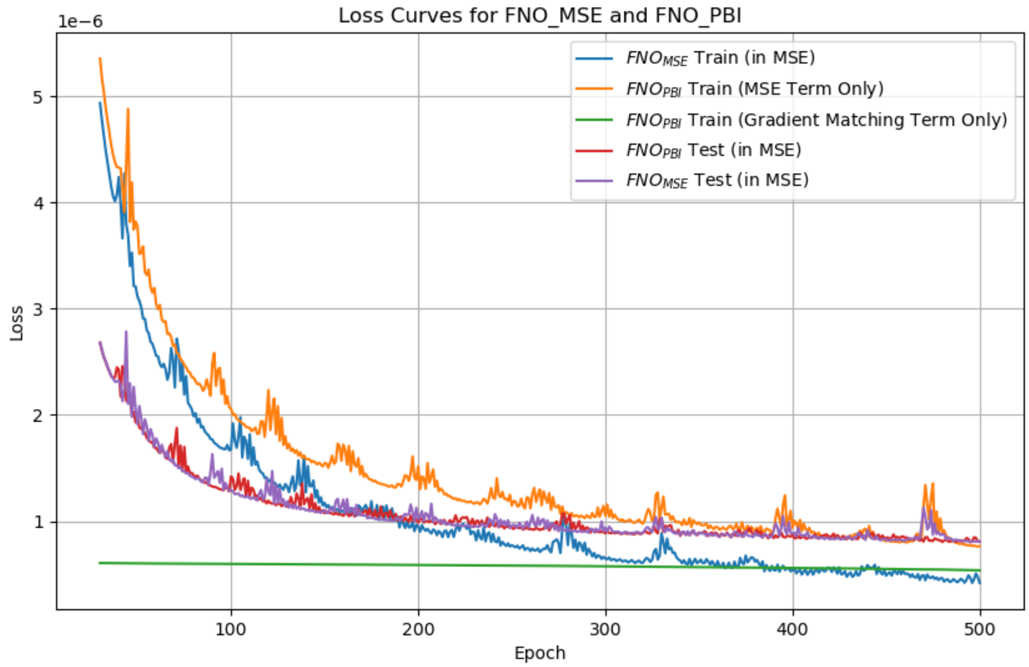
- As of right now, we don't see significant difference between MSE and PBI model in terms of posterior estimate.
  - It is likely undertrained.

6 Updates:

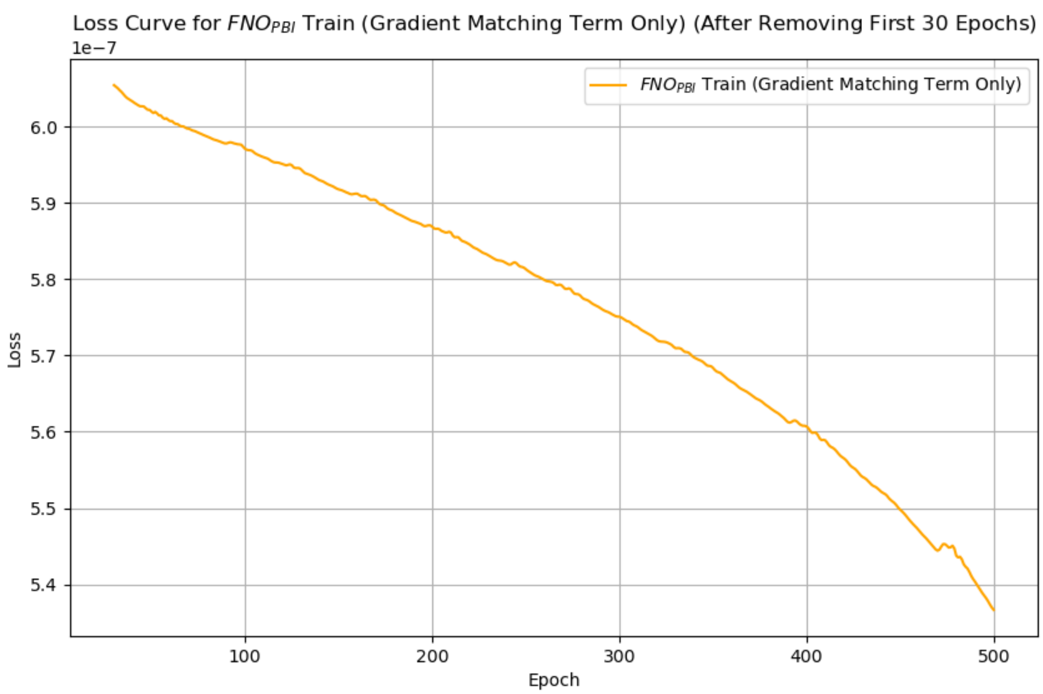
- To train FNO with multiple eigenvectors, have been generating dataset. For 1000 data points, we are obtaining the first 20 eigenvectors.
- However, number of observation is 20 (before it was 2) to get the FIM and we call Zygote.pullback 20 times per sample to get vJp, so it takes some time.
- We also had some debugged some code issues.
- So right now, tested with
  - 100 training sample,
  - 50 test samples
  - 500 epochs.
- And we show preliminary results with 3 different scenarios: when number of vector is 1, 3, 5.

6.1 Forward Simulation on Test Dataset

	number of $\vec{x}$	λ	Train Loss	Test Loss
			MSE/GM	MSE
FNO-MSE	N.A.	N.A.	2.0915×10^{-7}	8.08192×10^{-7}
FNO-PBI	1	0.65	3.8140×10^{-7}	8.0472×10^{-7}
FNO-PBI	3	0.65	3.0985×10^{-7}	8.2933×10^{-7}
FNO-PBI	5	0.65	2.6275×10^{-7}	8.2738×10^{-7}

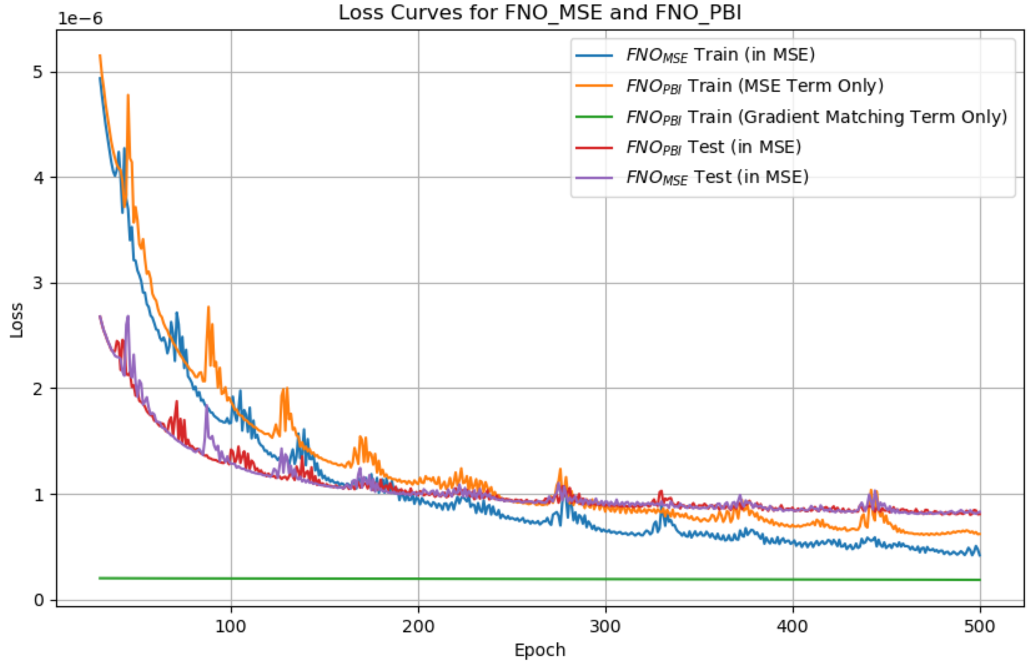


(a) All

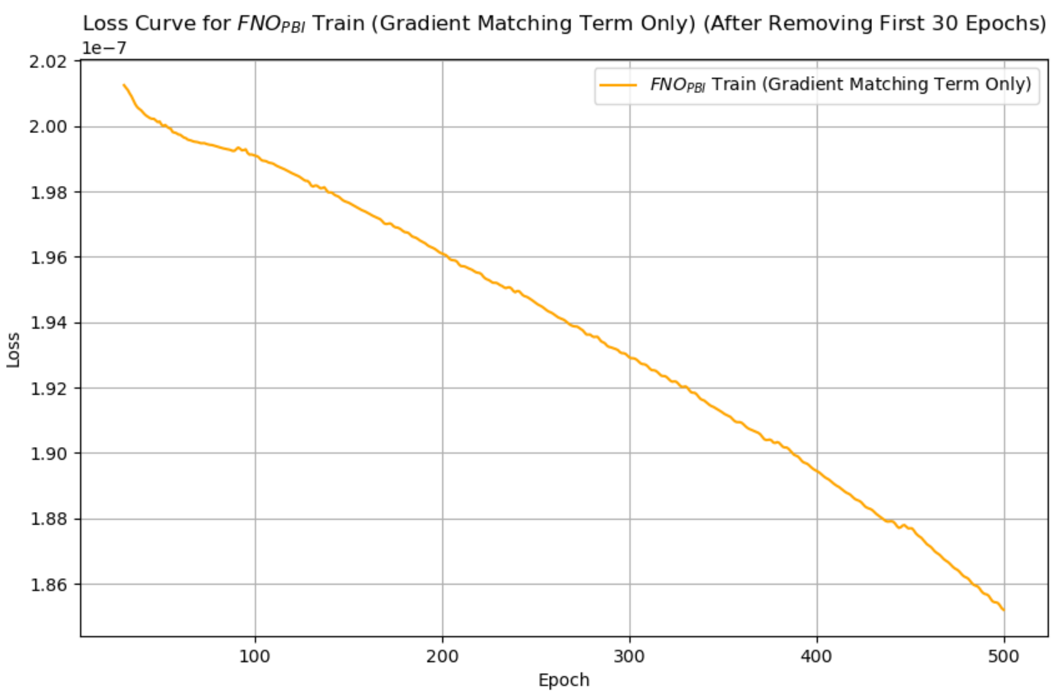


(b) GM term

Figure 18: When number of eigenvector = 1

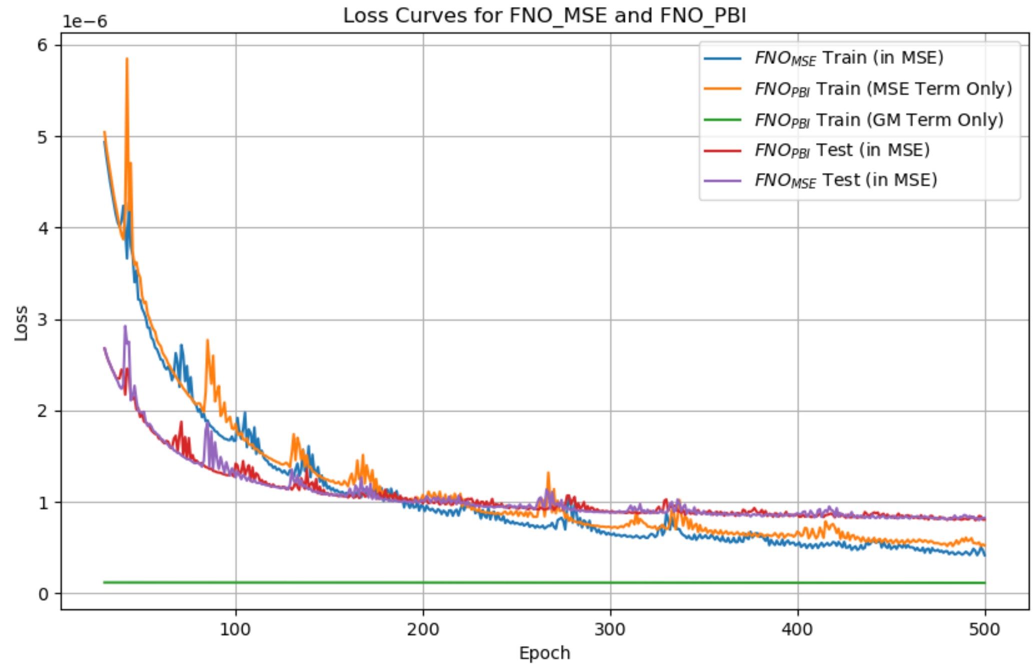


(a) All

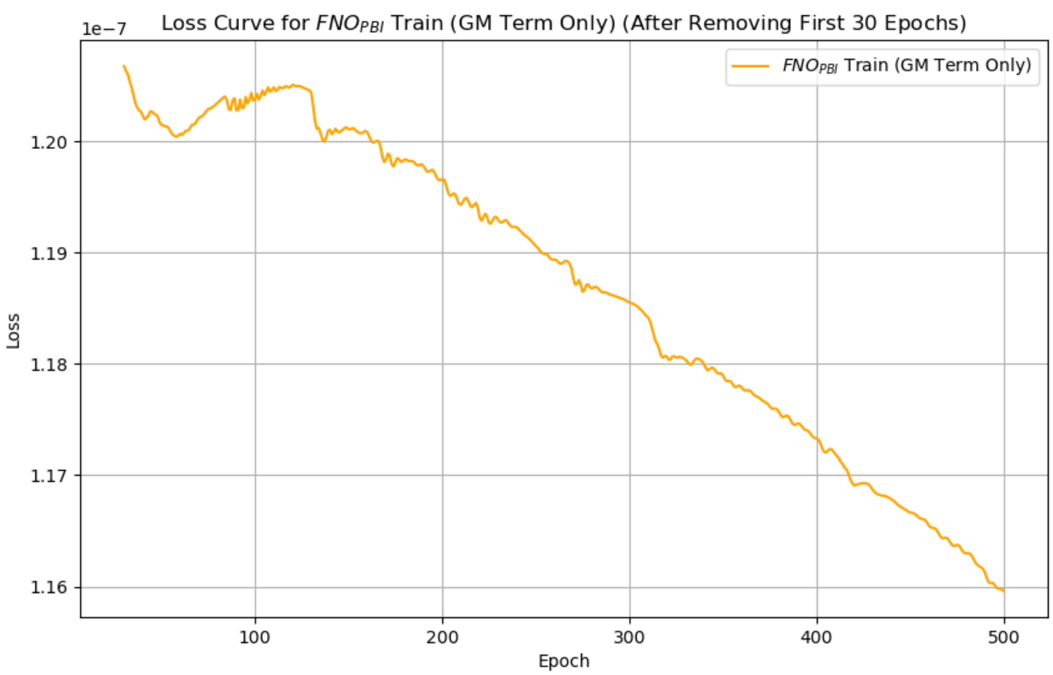


(b) GM term

Figure 19: When number of eigenvector = 3



(a) All



(b) GM term

Figure 20: When number of eigenvector = 5

6.2 MLE Estimate vs Number of Eigenvector

	number of $\vec{x}$	SSIM	Forward Loss	MSE
FNO-MSE	N.A.	0.7644	4.9598×10^{-5}	698.5997
FNO-PBI	1	0.7667	4.6305×10^{-5}	667.3286
FNO-PBI	3	0.7670	4.7811×10^{-5}	676.3323
FNO-PBI	5	0.7650	4.7420×10^{-5}	684.8894

6.3 Some comments on these mediocre results

1.

6.4 Side note: testing with one step prediction

6.5 Other comment

Will try to finish draft for ML4seismic presentation by Saturday.

6.6 Future Step

1. TODO: Debug NS eigenvector and vjp.
2. TODO: Want to generate the full dataset for Francis' dataset (which might take 1 or 2 days).
3. TODO: Try it on Jason's dataset (Now that we fixed the problem with FIM computation, we are optimistic about the experiment, so we want to try it again.)