Meeting Agendas

Damien

June 23, 2023

June 14, 2023

Participants: Amanda and Damien

Agenda

Today, we are discussing the following three papers

- Multifidelity PINN: `https://arxiv.org/pdf/1903.00104.pdf`
- Multifidelity DeepONets: `https://arxiv.org/pdf/2204.09157.pdf`
- Multifidelity Continual Learning: `https://arxiv.org/pdf/2304.03894.pdf`

Questions

Multifidelity PINNs

Q We do not have to train the final final neural network that takes in y_H and outputs f_e, correct? This is simply used for penalization?

A This is correct.

Q Does penalizing the gradient force the solution to converge to an approximation that is locally at least first order?

A They penalize the gradient because it trains better.

Q Can you discuss in more detail how each sub-network is trained?

A Training is done separately generally for each network separately. However, linear and nonlinear are trained simultaneously. The low-fidelity method can be trained separately. Training can be done all simultaneously?

Q How is data used for training in Figure 3? Did they train on data not in Figure 3a?

A Figure 3a contains all the training data. There is probably more testing data.

Multifidelity DeepONets

Q What are the essential differences between PINNs and DeepONets?

- For DeepONets you DO need to train the last network since you do not know the functional form of the operator, correct?

A PINNs train for one initial condition. Deep operator networks attempt to learn how all initial conditions map to solutions. The operator form is known, so the last part is known. We do not need to train the operator.

- What are the key differences in how PINNs and ONets are trained?

A Need smaller time steps and more data for DeepONets as opposed to PINNs.

Q How are M_L and P_L related in Figure 1?

A M_L contains the points/data needed to get a unique solution of the differential equation. P_L are simply data that we interpolate.

Continual Learning

Q You talked about this a bit yesterday, but what are the assumptions we make as far as knowledge about the low-fidelity model we are given?

- Even if we do not know the data the low-fidelity model was trained on, do we know the domain that data was in?

A We assume that the low-fidelity network is correlated with the solution on the new data set. We do not necessarily even know what domain the previous network was trained on.

- If the previous answer is yes, then why are we treating $\mathcal{NN}_{i-1}$ as a low-fidelity predictor on Ω_{i-1}?

Q Why do we need to train a single fidelity and a multifidelity PINN on Ω_1?

A The results were better when you initialize the multifidelity approach in this manner.

General Questions

Q Have you performed experiments with noise? Are these methods robust to noisy data?

A There is error added to the low-fidelity data in Burger's equation in the ONet paper. The high-fidelity data helps compensate for this noise.

June 22, 2023

Participants: Amanda and Damien

Agenda

I am having a hard time running the code on Marianas. The pendulum code works fine on my local machine, but there is something wrong still.

Comments

- The code works when I run it on my machine, so this has to be a problem with the environment.
- I have tried running two different slurm scripts: `mybatch.slurm` and `batch.slurm`. Both `.slurm` files give the same output.

Questions

Q Once I have run the lines listed below, do I need to run them again every time? Do the lines following `conda create --name jax-cuda` create a permanent environment that I can just load with `conda activate jax-cuda` from now on? Do I need to run the following lines again?

- `module purge`
- `module load cuda/11.4`
- `module load python/miniconda3.9`
- `source /share/apps/python/miniconda3.9/etc/profile.d/conda.sh`
- `conda create --name jax-cuda`
- `conda activate jax-cuda`
- `conda install scipy=1.10.0`
- `pip install --upgrade pip`
- `pip install --upgrade "jax[cuda11_pip]" -f https://storage.googleapis.com/jax-releases/jax_cuda`
- `conda install tqdm`
- `pip install torch`
- `conda install cudnn`

A No, you do not need to re-install the packages in the environment every time.

Q I put the above lines into a .sh file and the output when I run the .sh file is different than when I run the code otherwise. Is there a reason for this? I feel like it should be the same?

A Just put lines directly into the terminal since you do not need to run the code every time.

Q What is `SF_script.py`? Should I have access to this file?

Amanda's Comments:

- Possible reasons for error:
  - The version of cuDNN may not be compatible with what I need to do.
  - Try to re-install jax in the CUDA environment
  - This could be a memory error, so run on dl and not dl-shared
  - If it is a memory error, then it could be that the system is pre-allocating memory that does not exist

  - Take the line: `os.environ["XLA_PYTHON_CLIENT_PREALLOCATE"]="false"`. Put it in the script between when you import all the packages and when you start the code
  - If the code is not running on Marianas, developing locally is a good thing to do.

Multi-fidelity PINN Stuff:

- Amanda sent me an Overleaf for the multi-fidelity domain decomposition.
- Currently the code is not smart about how it selects points.
- There will be a running list of problems to work on in the Overleaf.