CS 165 Winter 2023	*Last updated:* February 18, 2023

Homework 3

Due Feb 27th, 2023

This assignment is a continuation of the neural operator bootcamp, where the goal is to use the neural operator package to simulate the Darcy equation and Navier-Stokes equation.

You will need to use GPU for this assignment. You use the provided Nvidia GPU instance, Google colab, or your personal desktop. If you opt for Nvidia GPU instance:

```
Sign up and enroll in the course:
https://courses.nvidia.com/courses/course-v1:DLI+S-OV-04+V1/
Click Enter Promo Code and enter:
DLITEACH0223_11_NWXB_32
Enter your billing address and submit.
```

Restarting the GPU instance will reset any changes you made, so save your work frequently. You can also request access to the Caltech cluster by contacting the TA.

1 Environment setup

First, please walk through the notebooks given in the Bootcamp repo https://github.com/NeuralOperator/bootcamp/: 1-setup.ipynb and 2-intro-FNO.ipynb. For an FNO layer with

`n_modes_height = n_modes_width = 12 and hidden_channels=32,`

what is the size of the weight tensors (parameters)? Print the tfno model as in 2-intro-FNO.ipynb.

2 Darcy Flow equation

The Darcy Flow equation is a 2nd-order Elliptic equation

$$\begin{aligned} -\nabla \cdot (a(x)\nabla u(x)) &= f(x) \qquad x \in (0,1)^2 \\ u(x) &= 0 \qquad x \in \partial(0,1)^2 \end{aligned} \tag{1}$$

This PDE has numerous applications including modeling the pressure of the subsurface flow, the deformation of linearly elastic materials, and the electric potential in conductive materials. We are interested in learning the operator mapping the diffusion coefficient a to the solution u. $G : a \mapsto u$. The force $f = 1$ is fixed with a zero Dirichlet boundary.

Walk through the bootcamp 3-darcy_flow.ipynb and 4-training-on-Darcy-Flow.ipynb. What is the training and testing error?

Could you modify the config in config/darcy_config.yaml and get testing l2 error on 64 resolution smaller than 5%?

3 Navier-Stokes equation.

Lastly, we'll tackle the Navier-Stokes equation, which is one of the hardest problems in scientific simulation. For this task we will train a FNO model to simulate the NS equation. You are free to use either the neural operator package https://github.com/neuraloperator/neuraloperator or the source code https://github.com/neuraloperator/neuraloperator/blob/master/fourier_2d.py.

Consider the 2-d Navier-Stokes equation for a viscous, incompressible fluid in vorticity form on the unit torus:

$$\begin{aligned} \partial_t w(x,t) + u(x,t)\cdot\nabla w(x,t) &= \nu\Delta w(x,t) + f(x), && x\in(0,1)^2, t\in(0,T] \\ \nabla\cdot u(x,t) &= 0, && x\in(0,1)^2, t\in[0,T] \\ w(x,0) &= w_0(x), && x\in(0,1)^2. \end{aligned}$$

where w is the vorticity field. We aim to learn the mapping of the vorticity field from one time frame to the next frame $G : w(:,t) \mapsto w(:,t+dt)$.

1. visualize the dataset Download the dataset from google drive https://drive.google.com/file/d/1BUMa8lczq4iK4MgOZ0kNfDPmCbM3k95C/view?usp=sharing. The dataset consists of 50 trajectories of simulation. Each trajectory has 501 time frames. Each time frame is represented as a 64×64 image (vorticity field). The file is about 800MB.

After downloading the dataset, print its shape and then plot the first trajectory's first four time frames, which should each be displayed as several images.

2. prepare the input and output dataset Since we want to learn the evaluation operator from the previous time step to the next, we can define the input as [50, 0:500, 64, 64] and let the output be one-frame off [50, 1:501, 64, 64].

Next, divide the dataset into two sets: 40 trajectories for training and 10 trajectories for validation. To convert the time series into an image-to-image mapping, mix the trajectory index and the time index. The resulting training input and output should have shapes of [40*500, 64, 64] and [40*500, 64, 64], respectively, while the validation set should have shapes of [10*500, 64, 64].

3. train the model With the dataset prepared, train the model (100 epochs) using the same pipeline as in the previous Darcy Flow problem. After training the model, save the model and report the training and testing errors.

4. visualize the output We have just trained the model for one-step mapping. Now it's time to generate a longer time series. To do this, use a for-loop that iteratively predicts the next time step as a recurrent model:

$$w(0)\mapsto w(1)\mapsto w(2)\mapsto w(2)\mapsto\cdots$$

Take the first frame $w(0)$ from the validation set and compare the later prediction against the ground truth. How does the error grow over time? Will it diverge? If so, explain why.