Mission 1

Objectives

Understand the given data (basically plotting), and obtain Φ_{res} field from ω_{c_T} and $|\nabla c_T|$ fields. The most basic example code is given and mission is yours.

You will learn

- Get familiar with Google Colab - make sure everyone can use it (Teach each other)
- Plotting of simulation data
- Some post-processing of data (thresholding)

Data explanation

- "wtemp-slice-B1-0000080000" is Q/c_p, where Q is a heat release rate and c_p is a specific heat capacity at constant pressure.
- "nablatemp-slice-B1-0000080000" is $|\nabla T|$, where T is a temperature.

"slice" means $x-y$ plane at $z = 5$ mm, "B1" means case B1, "0000080000" means timestep$= 80000$ (basically a single timestep data of a simulation). $x \approx 0$ region might have crazy values due to boundary condition effect, you will see the effect when you plot.

Tasks

1. Start Google colab environment. (Open the given codes in google colab and upload the data into the google colab by your own research)
2. Make ω_{c_T} field, $\omega_{c_T} = Q/c_p/(T_b - T_u)$, where $T_u = 1500.00$ K and $T_b = 1623.47$ K
3. Make $|\nabla c_T|$ field, $|\nabla c_T| = |\nabla T|/(T_b - T_u)$
4. Plot the both ω_{c_T} and $|\nabla c_T|$ fields to check if they look good.
5. Make $\omega^*_{c_T}$ and $|\nabla c_T|^*$, where superscript "$*$" means $x^* = x/max(x)$ (so value of x becomes between 0 and 1). Maximum of each fields are (for case B1): $max(\omega_{c_T}) = 1996.8891$ and $max(|\nabla c_T|) = 3931.0113$. I will give you the unit of the fields later :)
6. Check $\omega^*_{c_T}$ and $|\nabla c_T|^*$ fields are between 0 to 1.
7. Use Equation 1 in the paper "Data driven analysis and ..." to obtain Φ_{res}, and plot it to compare with Fig.7.

If finished

1. Make the figure beautiful (important in later stage)
2. Try to write a Gaussian filter code for $\overline{\omega}_{c_T}$ for given filter size (as a parameter), and visualize it to see if it is working and have fun with it.
3. Report any mistake found in this document (just a list of corrections - for improvement)

Mission 2

Objectives

Learn the filtering operation (Gaussian filter in this study). Apply the filter for Φ_{res} and ω_{c_T} to obtain $\overline{\Phi}_{res}$ and $\overline{\omega}_{c_T}$. After that, input the $\overline{\omega}_{c_T}$ and $|\nabla\tilde{c}_T|$ into the neural network to predict $\overline{\Phi}_{NN}$ and compare it with $\overline{\Phi}_{res}$.

Note: Regarding the $|\nabla c_T|$, Favre filter is used (density-weighted filtering), which uses ρ field. However, data for ρ is not available :(, so $|\nabla\tilde{c}_T|$ is given for different filter sizes.

You will learn

- Learn how filter works and implement in code (Teach each other)
- Learn to use trained NN
- Deal with some comparison methods (Pearson correlation coefficients)

Data explanation

- "tilde-nablatemp-slice-B1-0000080000-025" is $|\nabla T|$, where T is a temperature.

"tilde" means Favre-filtered field, "slice" means $x-y$ plane at $z = 5$ mm, "B1" means case B1, "0000080000" means timestep= 80000 (basically a single timestep data of a simulation), "025" is a filter size in cell size (basically "025" means $25 * \delta x$, where $\delta x = l_x/n_x = 10mm/384$).

Tasks

When following the task, do it with only one filter size (maybe $\Delta = 0.5\delta_{\text{th}}$), in order to avoid complication. Plus, your filtered data might look different than that of the original paper, because you will be filtering 2D data, while original paper did 3D filter on 3D data (effect from the z direction is lost in 2D data).

1. Get $\overline{\omega}_{c_T}$ for one filter size. (recommendation: $\Delta = 0.5\delta_{\text{th}}$)
2. Apply same filter to get $\overline{\Phi}_{res}$, using Φ_{res} obtained in the previous mission.
3. Prepare inputs, $\overline{\omega}^+_{c_T} = \overline{\omega}_{c_T}/(\rho_r S_L/\delta_{th})$ and $|\nabla\tilde{c}_T|^+ = |\nabla\tilde{c}_T|/(1/\delta_{th})$ and $\Delta^+ = \Delta/2\delta_{th}$ for the NN (Normalization). You can find the constants in the code. ρ_r - reactant gas density, S_L - laminar flame speed, δ_{th} - laminar flame thickness.
4. Use the inputs for the NN to get $\overline{\Phi}_{NN}$.
5. Compare the results: $\overline{\Phi}_{NN}$ vs $\overline{\Phi}_{res}$. Maybe use a Pearson correlation coefficients or MSE.
6. Do same things for the other filter sizes $\Delta = 0.50\delta_{th}, 0.75\delta_{th}, 1.00\delta_{th}, 1.25\delta_{th}, 1.50\delta_{th}, 1.75\delta_{th}, 2.00\delta_{th}$.

If finished

1. Make the figure beautiful again again again (it is never enough)
2. I GAVE YOU ALL DATA :)
3. Report any mistake found in this document (just a list of corrections - for improvement)

Mission 0

Objectives

This mission should have been done, even before the NN implementation :).

Learn the gradient calculation operation ("fourth-order central difference" method in this study). Apply the gradient calculation operation for $\tilde{c}_T$ field to obtain $\nabla\tilde{c}_T$, then calculate the norm $|\nabla\tilde{c}_T|$, which will be used as an input for the NN.

Note: I thought we have time, so lets calculate the $|\nabla\tilde{c}_T|$ by ourselves (for fun?). However, the data we have is 3D and gradient in z direction is not available. Lets just use x and y information. It does not affect much. I believe ...

You will learn

- Learn how gradient calculation works in computer. (Teach each other)

Data explanation

- "tilde-temp-slice-B1-0000080000-025" is $\tilde{T}$, where T is a temperature.

"tilde" means Favre-filtered field, "slice" means $x - y$ plane at $z = 5$ mm, "B1" means case B1, "0000080000" means timestep= 80000 (basically a single timestep data of a simulation), "025" is a filter size in cell size (basically "025" means $25 * \delta x$, where $\delta x = l_x/n_x = 10mm/384$).

Tasks

When following the task, do it with only one filter size (maybe $\Delta = 0.5\delta_{\text{th}}$), in order to avoid complication. Then, do for other filter sizes. Plus, your calculated gradient might look different than that of the original data, because you will be calculating for 2D data, while original paper did the same on 3D data (effect from the z direction is lost in 2D data).

1. Calculate the gradient of $\tilde{T}$ and get the norm of it, $|\nabla\tilde{T}|$.
2. Compare the obtained $|\nabla\tilde{T}|$ in step "1." to the one from the original study ("tilde-nablatemp-slice-B1-0000095000-025.raw"), to make sure the gradient calculation is working.
3. Obtain the $\nabla\tilde{c}_T$ field. (Note: $c_T = \frac{T-T_u}{T_b-T_u}$, so $|\nabla\tilde{c}_T| = |\nabla\widetilde{\frac{T-T_u}{T_b-T_u}}| = |\nabla\widetilde{\frac{T}{T_b-T_u}}| - |\nabla\widetilde{\frac{T_u}{T_b-T_u}}| = |\widetilde{\frac{\nabla T}{T_b-T_u}}| - 0 = \frac{|\nabla\tilde{T}|}{T_b-T_u}$)
4. Do same things for the other filter sizes $\Delta = 0.50\delta_{th}, 0.75\delta_{th}, 1.00\delta_{th}, 1.25\delta_{th}, 1.50\delta_{th}, 1.75\delta_{th}, 2.00\delta_{th}$.

If finished

1. Maybe you can play with different gradient calculation methods, change order of the current scheme etc.
2. Understand how gradient calculation is working or how it is derived. Maybe ChatGPT???
3. Prepare if you have any questions.
4. Report any mistake found in this document (just a list of corrections - for improvement)

Mission 3

Objectives

Learning of the convolutional neural network (CNN) implementation with simple example. The objective is to train a CNN so that it can detect stars in given figures.

Note: Anh Khoa gave me this example case.

You will learn

- Train your own CNN? or dragon?

Data explanation

- Figures in "Stars/x" directory are the input data. They consist of some random background and star figures.
- Figures in "Stars/y" directory are the output data (target). They consist of corresponding star figures, without any background images. For example, image "Stars/y/0001.png" is the star-only figure of "Stars/x/0001.png".

Tasks

When following the task, do it with only one filter size (maybe $\Delta = 0.5\delta_{\mathrm{th}}$), in order to avoid complication. Then, do for other filter sizes. Since CNN accepts only image data, we cannot give it an extra parameter that includes filter size information (can we?? if so, teach me how). Thus, probably you will end up having separate models for each filter size.

1. Upload the "CNN.ipynb" into google colab.
2. Upload the "Stars.zip" into "/content/". Just upload without unzipping it. There is a unzip line in the code itself.
3. RUN ALL and check the result.

Basically, code does:

1. Unzips given Stars.zip
2. Loads the data (x: input & y: corresponding target or label data) and divides them into training set and test set.
3. Defining CNN model
4. Training
5. Drawing loss function on train and test data
6. Prediction images by the CNN for some samples to check if the CNN is working.

If finished

1. Understand what CNN actually doing. (Maybe some youtube videos)
2. Discuss how to implement CNN for the Φ_{CNN} prediction within the group.
3. Prepare if you have any questions.
4. Report any mistake found in this document (just a list of corrections - for improvement)