Notes on Rayleigh-Benard convection script in Dedalus

I've attached the following scripts:

rayleigh_benard.py
This is my main dedalus script for running the model and reads in parameters from a run_param_file.py.
Saves data into a folder called raw_data, along with 3 subfolders containing different outputs.

- snapshots contains the solver state (u, w, s, etc.) at a frequency specified in run_param_file.py.
- analysis contains all of my analysis tasks such as the various fluxes (L_cond, L_conv, etc.) values for E, along with a domain integrated KE and Reynolds number. Save frequency also specified in run_param_file.py
- run_parameters simply saves a large number of useful parameters (essentially everything specified in run_param_file.py) at a frequency of 1e20 seconds... so once at the start of the simulation and then never again. Probably a better way of doing this, but this is how I've been doing it as it at least works and saves all your parameters into a .h5 file!

run_param_file.py
As the name suggests, contains all the relevant input parameters. They shouldn't need to vary anything in rayleigh_benard.py if they want to simply run the simulation again with new inputs.

merge.py
This is unchanged from the script provided with dedalus. Compiles the .h5 files produced from each processor into a single .h5 file per output (still restricted by the max_writes values specified in rayleigh_benard.py). Still needs to be run if students only run rayleigh_benard.py on a single core, as will leave the files in the folder structure expect by merge_single.py.

merge_single.py
My own script which takes the multiple .h5 files left over by merge.py and merges them so that there is only a single .h5 for snapshots, analysis, and run_parameters (3 files in total).

plotting_snapshots.py
Produces a folder called figs, which then contains within it

- A plot of average KE vs time
- A 4 panel plot of the final state showing the entropy, vertical and horizontal components of velocity, and again the average KE over time.
- A folder called fluxes that has time averaged plots of the two different flux decompositions (obviously for the theta=0 boussinesq case the L_diss, L_buoy, L_visc, and L_KE components are all zero as each of these has a factor of theta associated with them) and a 3 panel plot showing again the two flux decompositions and a KE vs time plot with the range of the time average indicated.
- A folder called snapshots which includes multiple 4 panel plots showing the evolution of u, w, s and KE at different points in the simulation. These are the plots I usually make the videos from.

I tried to have the plotting_snapshots.py script simply make the video using matplotlib but wasn't successful, and think it might be easier to simply have the students download Fiji (https://imagej.net/Fiji/Downloads) which is a piece of image processing software that's installed in the undergrad mac labs. It's really lightweight and easy to use, and it's what I use to convert the folder of plots into a video.

Typical running order of all the above would look something like this:

```
python3 rayleigh_benard.py
python3 merge.py raw_data/snapshots --cleanup
python3 merge.py raw_data/analysis --cleanup
python3 merge.py raw_data/run_parameters --cleanup
```

Then give the simulation some name in merge_single.py, say like "test1"

```
python3 merge_single.py
```

Check the plotting_snapshots.py script is pointing towards the location of the data (only need to change the variable "direc" to point towards the folder that contains the snapshots, analysis, and run_parameter folders), and then change the run_name variable to whatever you named the simulation in merge_single.py.

```
python3 plotting_snapshots.py
```

Should then produce all the plots, and can make a video using Fiji.