<u>Project Readme</u>

As always start by running the imports cell, it imports all the files and functions used in this project to the Jupyter notebook.

In order to load the datasets, use the function getLoader – returns the train_loader, val_loader, test_loader, dictionary with the properties of the real star. We used our best models with <u>seed=1.</u>

To load a model, you use the function: **load_model_from_mode** that receives: The model type: 'spectra' or 'doppler', the parameter <u>with_drop</u>, and the dropout value (if <u>with_drop</u> is True), the learning rate of the optimizer used to train the model, and a parameter <u>name</u> which is used to give another identifier for files with the same parameters but on different sets or train methods (it's just a string). In order to save a model use the (familiar from previous exercises) function: **<u>save_model_with_checkpoint</u>** which receives, the model, the learning it trained on, its type: 'spectra' or 'doppler', and the optional parameter <u>name</u>.

<u>Train & Test:</u>

- In order to train SpectraNet you need to used the function train_spectra1 (there is a train_spectra2 but it didn't give a good result so we didn't updated it over time) the function receives: the SpectraNet model, an optimizer, a learning rate scheduler, train_loader, val_loader, epochs amount – best results with 15.
- In order to train DopplerNet you have 2 options as stated in our report:
    1. "Together" with SpectraNet – "real" way to train the whole model Using the function: <u>train_doppler_together25</u>, that receives: a SpectraNet – pre-trained, a DopplerNet (untrained), an optimizer, a learning rate scheduler, train_loader, val_loader, epochs amount – best results with 15.
    2. "Separate" without dependency in SpectraNet – a way to train DopplerNet when you have the real spectras of the stars. Using the function: <u>train_doppler_seperate25</u>, that receives: a DopplerNet (untrained), an optimizer, a learning rate scheduler, train_loader, val_loader, epochs amount – best results with 15.
- When you want to test your models there are 4 options:

1. Test only SpectraNet: using the function check_spectra_model that receives: a SpectraNet, a data loader, a parameter plot_amount which if bigger than 0 the function will plot the first 'plot_amount' spectras (each spectra with its respectable ground truth), and the standard_vector – it's the x axis values in the plot.
2. Test only DopplerNet: using the function check_doppler_model that receives a DopplerNet, a data loader, and a plot_amount
3. Test both of them together (reality): using the function check_all_forward that receives a SpectraNet, a DopplerNet, a data loader, the parameter plot_amount, and the standard_vector. This function first finds the closest spectra to each star and by that coupling chooses the coupling of the velocities (if spectra1-output was closer to spectra2-ground truth, then velocity1-output will be compared to velocity2-ground truth).
4. Test both of them together optimally (not reality): using the function check_all_forward_optimal that receives a SpectraNet, a DopplerNet, a data loader, the parameter plot_amount, and the standard_vector. This function basically combines the 1st two functions mentioned in this section, and can be used as a lower bound of the networks.

Real star testing:

- To get the real star data first you need to use the function get_real_loader That needs as input the std_dict from the getLoader function.

- In order to test SpectraNet on it you need to use the function test_real_star_spectra that receives a SpectraNet, and the dictionary of the real star data
- In order to test DopplerNet on it (separately or together) you need to run the cell under the label "parametrs for plotting the results", and then use the functions:
    1. Separate: test_real_star_doppler_seperate, receives: a DopplerNet, the dictionary of the real star data, times vector, the repeat_time value and the velocities vector of the star.
    2. Together: test_real_star_together, receives: a SpectraNet, a DopplerNet, the dictionary of the real star data, times vector, the repeat_time value, velocities vector of the star, and the dictionary of the real star data.

files are at the repository: https://github.com/dschein1/Resolving-SB2-DL