lilpPXF running guide

Here are the steps you need to run lilpPXF from start to finish, which will read your data, spatially bin the spectra, fit them with stellar templates, subtract these fits on the pixel level, re-bin based on an emission line and recompute emission line products from that.

The pipeline takes a parameter file, e.g. "pf.param". You can find a description of all parameters in parameterfiles/pf.param

There are also example batchfiles for some of these steps in batchfiles/

Step 1) Set up the python environment using the .yml file, and activate it.

```
conda env create –file pPXF_env.yml
conda activate pPXF_env
```

Step 2) Set up a python file.

Set up a file like "spectral_fitting.py", which will be how you call lilpPXF. Make sure the module loads correctly. Setting it up this way lets you call individual functions if you so desire.
You will run the pipeline by executing

```
python spectral_fitting.py pf.param
```

Where “pf.param” is one of the options described below, and as you will see, some tasks can be made parallel.

Step 3) Reading data
Navigate to parameterfiles/read_data.param
Set the read_data field to 'True', enter your datacube filepath, and set redshift and wavelength parameters.
Run by executing

```
python spectral_fitting.py parameterfiles/read_data.param
```

Step 4) Spatial binning
Navigate to parameterfiles/vorbin_SN40.param

Set your desired input and output directories. I recommend setting a different output to avoid overwriting anything and enabling all future analyses on this binset to be drawn from the same folder.
Set "run_vorbin" to True.
Set your binning parameters
Run by executing

```
python spectral_fitting.py parameterfiles/vorbin_SN40.param
```

Step 5) Stellar continuum fitting
Navigate to parameterfiles/SN40_stelkin.param
Set your desired input and output directories, again I recommend setting a different output to avoid overwriting anything and enabling easy comparison of different parameters.
Set run_fit to "True"
Set fitting parameters
Run by executing

```
python spectral_fitting.py parameterfiles/SN40_stelkin.param
```

This step can be run in parallel

```
mpirun -np 8 python spectral_fitting.py parameterfiles/SN40_stelkin.param
```

This will also create continuum-subtracted outputs on the pixel level by taking the best-fit spectrum for each bin, scaling it to the relative strength of the individual pixel spectrum, and subtracting it. It will also measure the emission lines.

Step 6) Spatial binning on an emission line
Navigate to parameterfiles/vorbin_contsub_Halpha.param
Set input and output dirs - input here is where you have measured Halpha emission lines in the last step. Output is where you want the new binned spectra and outputs to go.

Set your desired binning parameters.
Set binned_contsub to "True" to co-add the already continuum subtracted spectra from the last step.
Set contsub_level to 'bin' to produce maps and line measurements using the new spatial binning scheme, as desired
Run by executing

```
python spectral_fitting.py parameterfiles/vorbin_continuum_Halpha.param
```