Astro 585 Project Proposal: Mock Simulations for HETDEX LAE Extraction to determine Completeness and Contamination

Codename: The Maxwell-Jüttner Distribution

February 21, 2014

1 Goal

The Hobby Eberly Telescope Dark Energy Experiment (HETDEX) is coming on-line in the fall, and it will detect on the order of 800 000 Lyman-α emitting galaxies (LAEs) at redshifts $\sim 2 < z <\sim 3.5$ to measure the equation of state of dark energy. Ly-α emits at 1216 Å in the rest frame, which gets shifted to the range $2500 - 5000$ Å at these redshifts.

In the local universe at redshifts $z < 0.5$, the forbidden line doublett [OII]3727 gets shifted into the same range. Indeed, there is a significant population of OII-emitters in the local universe. This introduces contamination of the sample of LAEs on the order of 20% if no careful classification is done.

It is therefore important to simulate the observations to test different algorithms that classify galaxies according to LAEs at high redshift versus OII emitters in the local universe.

This project proposal is about inserting fake LAEs and OII emitters into real images, and extracting their photometry for others to test their classification algorithms. A rudimentary code mostly written in python already exists.

Parallelization will come into play in two ways. First, this can be done embarassingly parallel for $\sim 1\,000\,000$ galaxies by distributing this among several computing nodes.

However, another possibly interesting parallelization technique will be to insert the fake LAEs and OII emitters into an image using GPUs. This will involve drawing from a distribution of possible galaxy morphological shapes. With a GPU the same morphology could then be placed into multiple images. Of course, multiple galaxies can also be placed into one image, as long as there aren't too many to step on each other.

2 Inputs and Outputs

The input is provided as a ASCII text file with RA, Dec, z for each LAE and another ASCII text file for the OII emitters. They have been created to simulate the large-scale structure of the universe form a given power spectrum.

We will also need to draw from the luminosity function of LAEs and OII emitters, which is provided by a ASCII file for each type of galaxy.

Lastly, there is a collection of images to use as a background testbed. These are stored in FITS file format.

The output for each LAE or OII emitter will be its continuum magnitudes for whatever continuum bands we have (K is being observed, g is planned) and Ly-α line flux.

The final product after applying a classification algorithm should be RA, Dec, continuum magnitudes, line flux, flux error, equivalent width (assuming it is an LAE), probability of being an LAE, and probability of being an OII emitter. The probability calculations are beyond the scope of this proposal, and is done by collaborators at Rutgers. The result will then be used to calculate the power spectrum, and compared with the original inputs (also by collaborators, one of which is coming to Penn State at the end of March).

3 Testing plan

This is a testing plan. Parts of it will be used for the actual experiment. To test the testing pipeline, we will start with the already existing pipeline as a reference. However, some error estimations are not yet implemented. In particular, HETDEX will give the position of a potential LAE to about 1 arcsecond. This error in the position will need to be taken into account when measuring the photometry of the LAE candidate. Mastering the force with YODA[1] has already been achieved. YODA is made of C++ code. We will then use this as a testbed for a more complex analysis that takes the error in the position into account. The precise algorithm has yet to be developed.

A rudimentary, unoptimized pipeline already exists. It is largely written in Python, but uses YODA for the object detection and photometry. First, we will perform unit tests on drawing randomly from the luminosity functions. The error can be estimated, so a unit test may sometimes fail with a predetermined frequency. The we will create another unit test for inserting a fake galaxy into an image. This will be an empty image, a slightly more populated image, and a realtively crowded image. Finally, unit tests will be developed to see how well YODA performs the extraction.

The serial implementation will function as a reference for the parallelization.

4 Problem size

As previously mentioned, HETDEX will detect about 800 000 LAEs, and a few million OII emitters. We are therefore planning on simulating on the order of one million galaxies to test the extraction pipeline, and determine our fraction of contaminants as well as incompleteness.

We will not need a large variety of background images, perhaps on the order of a few tens. The pipeline is therefore not very memory intensive. However, extraction and error analysis will required substantial computational resources.

5 Target Architecture, Programming Language, Libraries, Parallel Programming Paradigm

Several approaches are possible here. The simplest would be to just run the pipeline concurrently on multiple processors. This would not require many changes. However, there seems to be much more potential for running parts of the pipeline on a GPU.

[1] Yet another Object Detection Application, `http://www.as.utexas.edu/~drory/yoda/index.html`. YODA works similarly to SExtractor, but is better with multi-band imaging.

Much of the existing pipeline is written in python. As there is considerable momentum for python in the collaboration, we most likely we will continue using python to a large extent. However, core algorithms that require speed will be done in plain C. The state python bindings to GPU is unclear to the author at this stage. However, C libraries do exist that can be taken advantage of.