Instructor Notes: Data Science in Radio Astronomy II

Module Overview

In Data Science in Radio Astronomy II, we build on what was covered in Part I of the sequence (the 21 cm Hydrogen line module). In this module, we focus on the Voyager 1 signal. This module is divided into two parts:

- **Part 1:** Using GNU Radio, students visualize the Voyager 1 signal and save the raw data file.
- **Part 2:** Students import that saved data into Python (via a Jupyter Notebook) to carry out advanced spectral analysis, including FFT, peak detection, and SNR computation.

The notes below are designed to help you guide your students through each part, providing clear instructions, discussion prompts, and troubleshooting tips.

Part 1: Signal Visualization in GNU Radio

Overview and Objectives

In Part 1, students work with GNU Radio to visualize the Voyager 1 signal. This is a hands-on introduction to the data: Voyager 1's weak carrier signal is recorded at approximately 8.4 GHz. They will build a flowgraph, observe spectral and waterfall displays, and enhance the visibility of the signal.

Key points to mention:

- Ensure the data file `Voyager1DATA.ci8` is downloaded and accessible.
- Explain the purpose of each block in the flowgraph and how it contributes to visualizing the signal.

Troubleshooting for Part 1

If students run into issues:

- **File not found:** Confirm the file path.
- **Unexpected signal behavior:** Check that the sample rate is set to 2929690 and that the block configurations match the provided schematic.

Flowgraph Construction and Configuration

Guide students to:

- Open GNU Radio Companion and create a new flowgraph.
- Add and configure blocks such as the File Source (set to `Voyager1DATA.ci8`), IChar To Complex, Frequency Xlating FIR Filter (using decimation, proper taps, and center frequency settings), and both the QT GUI Frequency and Waterfall Sinks (using FFT size 4096 and center frequency 8431e6).

Encourage them to run the flowgraph and confirm that they see displays similar to those in the lab manual. Remind them that adjustments like averaging and ROI selection can help clarify the weak signal amidst noise.

Discussion Points for Part 1

Ask:

- What do you observe in the raw spectrum?
- How does averaging help in making the signal more visible?
- Can you identify the central carrier and the sidebands, and what might these indicate?

Part 2: Spectral Analysis and SNR Computation in Python

Overview and Objectives

In Part 2, students transition from visualization to deeper analysis in Python. They will load the Voyager 1 data (saved from Part 1), compute the frequency axis, generate the full power spectrum via FFT, zoom in on a region of interest (ROI), perform automated peak detection, and finally estimate the noise floor and compute the SNR.

Key objectives:

- Load the saved data file in `complex64` format.
- Compute the FFT and visualize the complete power spectrum.
- Define a ROI, detect peaks, and calculate SNR.

Lab Walkthrough

Walk through the following steps:

1. **Data Loading and Frequency Axis Computation:** Explain how to load the data in Python and compute the frequency axis using parameters like center frequency, sample rate, and FFT size. Emphasize the use of `fftshift` to center the spectrum.
2. **Plotting the Entire Spectrum:** Show students how to segment the data, perform FFT on each segment, average the power spectrum, and plot the result. Ask them to observe any dominant features in the overall spectrum.
3. **Zooming into the ROI:** Guide them in selecting the ROI by defining frequency boundaries and creating a mask. They should plot the zoomed spectrum and note details like the carrier and sidebands.
4. **Peak Detection:** Demonstrate using SciPy's `find_peaks` function. Explain parameters such as the peak height offset, minimum distance, and prominence. Encourage students to adjust these settings and see the impact.
5. **SNR Computation:** Explain how to choose quiet regions in the spectrum, compute the noise floor, and then calculate the SNR as the difference (in dB) between the signal power and noise floor. Introduce the exercise where students complete or modify the `compute_snr()` function.

Troubleshooting for Part 2

If issues arise:

- **Data Loading Errors:** Verify the file path and format.
- **Incorrect Frequency Axis:** Recheck the center frequency, sample rate, and FFT size.
- **Unexpected FFT Output:** Ensure that data segmentation and averaging are done correctly.
- **Peak Detection Challenges:** Suggest fine-tuning the parameters (height offset, minimum distance, prominence) if peaks are not detected as expected.

Discussion Points for Part 2

Encourage questions such as:

- How do the observed spectral features relate to what was visualized in Part 1?
- What adjustments improved the clarity of the zoomed spectrum?
- How does the choice of noise estimation method (mean vs. median) affect the SNR?
- What insights does the automated peak detection provide about the signal?

General Guidance and Concluding Remarks

When running the lab, please ensure that:

- You remind students to document their observations either in markdown cells within the notebook or in a separate lab notebook.
- They compare results from GNU Radio (Part 1) and Python analysis (Part 2) to gain a holistic view of the signal.
- Each student experiments with parameter variations, noting the effects on spectral resolution, peak detection, and SNR.

Encourage a discussion at the end of the lab where students share their experiences and insights. Ask them to summarize their findings in a short report covering:

- Observations from the full spectrum and ROI.
- The impact of parameter adjustments on the analysis.
- Their approach to noise estimation and SNR calculation.