Problem Set 4: Sleep Stages and Hypnograms

Problem Set 4 will give you the opportunity to develop your Python programming skills as we explore electroencephalogram (EEG) data. In Problem Set 4, you will write code **to classify sleep stages and create hypnograms** showing how the sleep stage changes during the sleep epoch.

Why do neuroscientists create hypnograms?

Sleep researchers classify sleep into multiple stages based on various physiological characteristics including eye movements (detected by electrooculography, EOG), muscle tone (detected by electromyography, EMG), and neural activity (detected by EEG). These stages include both rapid eye movement (REM) and non-rapid eye movement (NREM) sleep. NREM sleep can be further divided into three or four categories based on specific EEG activity patterns.

Hypnograms are useful tools to visualize how sleep stages change throughout the night. In general, a night of sleep consists of multiple sleep cycles about 90 minutes in duration. Early cycles have the most slow wave sleep (SWS - the deepest levels of NREM sleep) while later cycles have the most REM sleep, and may even include periods of wakefulness. Hypnogram analysis may indicate disrupted sleep resulting from various medical conditions or disorders or as a side-effect from various drugs.

Hypnograms are traditional constructed by visual inspection of the physiological signals. Researchers classify sleep in 30 second epochs based on criteria original standardize by Rectshaffen and Kales in 1968[1] and revised by the American Academy of Sleep Medicine (AASM) in 2007[2]. This process is often completed manually, although there are some examples of algorithms that classify the physiological activity. In this problem set, you will create an algorithm that attempts to classify sleep state based on EEG activity. You will also generate visualizations of EEG data including hypnograms.

Why do neuroscientists use automatic classifiers?

Neuroscientists often seek to "decode" the neural signal recorded during experiments. The purpose of this decoding may be to understand how the neural activity relates to overall brain state or human behavior (as in this problem set), or the purpose of decoding may be to drive neuroprosthetic devices (as we learned about from Dr. Donoghue). Classifiers are simply algorithms that attempt to assign meaning (in our case, sleep stages) to activity.

In this problem set, we will guide you to complete a simple classifier. Because the classifier will be quite simple, its performance may not be that high. Neuroscientists often apply technically advanced machine learning techniques to build classifiers of neural activity. Some machine

[1] "A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects." Allan Rechtschaffen and Anthony Kales, editors. , 1968.

[2] "The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications". Iber, C; Ancoli-Israel, S; Chesson, A; Quan, SF, Westchester: American Academy of Sleep Medicine, 2007.

learning algorithms learn to classify data in a “supervised” way by learning from data that has been pre-classified (for example, by the human researchers who developed our data set.). The algorithm will create a classifier that can then be applied to new data. Other algorithms seek to find the different groups or clusters of data within the existing set in an “unsupervised” manner and then classify data into these groups. These types of machine learning techniques are outside the scope of this course, but if you have background experience with machine learning you are welcome to apply them here or in the final project for the course.

To complete this problem set, you will

- Examine examples of pre-classified sleep data.
- Write functions to predict sleep state.
- Write functions to visualize how sleep state changes with time.

At the end of this problem set, you will

- Complete a programming assignment submission which will evaluate that your functions can predict sleep state.
- Complete a peer assessment submission to confirm that you are clearly displaying power spectra and plotting hypnograms. (To receive credit for the peer assessment, you must complete your peer evaluation by the evaluation due date.)
- Complete some short questions to assess your ability to interpret the data.

Things to keep in mind while you are coding

- It’s really easy to make mistakes when you are writing code. It happens to everyone. Try not to get frustrated. Ask for help if you need it!
- Sometimes it is useful to print out the values of some of your variables to make sure things are going as you want them to (or you can use the debugger in Spyder – but printing is sometimes much easier!)
- It is often very useful to write out your logic with pencil and paper before you begin.

Data Set

This data set contains one channel of EEG data recorded from one subject during one night of sleep. The data has also been classified by researchers according to the Rectshaffen and Kales criteria (see above).

Getting started

Your first step is to open problem_set4.py in Spyder (or whatever Python environment you are choosing to use.) Again, you will want to open an IPython console in Spyder. Looking at the .py file, you’ll see that it starts with some comments, and some import commands. You’ll also see some functions (and skeletons of functions) that we will work through as we go through the problem set. Below all the functions, you can write lines of code to run. (We provide hints in the comments)

Step 1: Exploring the Provided Example

In the first part of the problem set, you will look at some provided EEG data that has already been classified. While you look at the data, you are trying to determine the "rules" you might use to classify sleep state.

In this part of the problem set, we pose some suggestions for exploration you might do or figures you might make, but these are all for your own benefit. You will not turn in anything for this part of the problem set – although you will use some of your observations to answer the homework questions.

Loading the example data

For this problem set, we have provided four ~5 minute examples of the four sleep states. You can access these with the `load_examples` function. You can run it as follows:

```
examples, srate = load_examples('example_stages.npz')
```

This reads in the example file and stores the data in a 2-dimensional array (`examples).` Each row in the array corresponds to 5 minutes of data from one sleep stage.

- Row 0: REM sleep
- Row 1: Stage 1 NREM sleep
- Row 2: Stage 2 NREM sleep
- Row 3: Stage 3 and 4 NREM sleep (for this problem set, we will combine stages 3 and 4, as in the AASM criteria.)

`srate` contains the sampling rate for the data.

What is that sampling rate for this data?

Look for differences in EEG between states

Now that we have these four examples, you can plot the power spectral density of the four sleep stages to look for differences between the EEG signals recorded during each of the these sleep epochs.

We have started the function `plot_example_psds` for you. You can use the `m.psd` function (NFFT values of 256 and 512 are both reasonable for this data set). You may find the function `plt.legend` helpful, as well as the `plt.xlim` function useful for constraining the plot to about 20 or 30 Hz. (If you don't constrain the plot, you will see a small peak at 60 Hz. 60 Hz is a common frequency for electrical noise in the United States, and researchers try to minimize it as much as possible).

Additionally, in your exploration of the neural data, you will likely want to normalize your plot (thus making the area under the curve that you display equal to 1.) You can easily accomplish this by dividing the y-values of your plot by the sum of all the y-values. This will be especially helpful if you choose to vary the time interval you look at.

We have some homework questions about the examples. (Listed here and at the end of this document.):

> *REM sleep looks most similar to which stage of NREM sleep?*
>
> (Because these stages look so similar, we will only ask you to classify NREM sleep stages for this homework assignment. REM sleep can be distinguished from NREM by also considering EOG to see the rapid eye movements and EMG which indicates the drop in muscle tone, but we are not including this data in our set.)
>
> *NREM stages 3 and 4 are dominated by which frequencies?*
>
> *Sleep spindles, typically between 11-15 Hz, are most likely prominent in which sleep stage?*

Looking at variability within the sleep stages

The plots we just made were over the entire 5 minute period, but researchers score sleep every 30 seconds (and you are welcome to play with window size during this problem set – although we will ultimately ask you to focus on 30 second bins). It is also useful to consider this variability before we begin classifying the data. We can easily see the variability using spectrograms. Remember that the spectrogram basically allows us to use a color plot to see how the power spectral density varies with time.

We have started the function `plot_example_spectrograms` for you. You can use the `plt.specgram` function with the same parameters you used for the psd plots. Here, you may find `plt.ylim` to be a useful function for constraining the figure.

You should see that the spectrograms also emphasize the same differences you observed with the psds, but you can see that the signal does indeed vary a lot. The spectrogram is created with smaller bins than the 30s bins used by researchers (for the NFFT values we suggested - the number of bins will decrease as you NFFT value increases).

Step 2: Determine the rules to "guess" the sleep stage

Now that you can see the differences between the states, you will formulate rules to guess the state of unknown EEG signal. It's time to use some of your own creativity! We haven't provided you with any specific functions here. This is really just an opportunity for you to delve deeper into the data you have just explored in an effort to quantify exactly what makes the three sleep states differ. In the problem set, will not be scoring any of the code or figures you use to generate your rules. This is all for your own discovery process.

There are many approaches you can take to find these rules, and you are not limited to using just the tools we have discussed.

Here is one approach:

- Create the PSD over each 30s epoch from each of the three sample NREM data sets.
- Visualize the power in the frequency bands that you expect to be most different between the three stages of NREM sleep. (Hint: Consider the answers to the questions posed above.)
    - You may want to consider the distribution of the power in your bands of interest over the 5 minutes of samples.
    - One way to visualize the distribution would be to use histograms. (Hint: Check out the parameters for `plt.hist`. Cumulative, normalized histograms drawn with `histtype= 'step'` can create a nice visual for you to consider.)
    - Examine these distributions yourself. (By this, we mean LOOK at the data – don't just rely on code to find the answers.) Do the three types of sleep have different distributions? What makes them different?
- Use the differences you've identified to create a set of rules to classify the data. Write down the rules for reference. We'll turn them into code in the next step!

There are many other approaches as well. For example, you may choose to break down your epoch into smaller bins. The rule for your entire epoch might choose the most popular classification from all the smaller bins – like an election, where each bin gets a "vote."

Step 3: Turn your rules into code

We would like you to create a classifier that scores an individual 30s epoch of EEG data and returns your best guess at the NREM sleep stage for that epoch. (Recall we will not be classifying REM sleep in this problem set.) In this part of the problem set, you will write your classifier, which will be evaluated as part of the programming assignment.

Writing your classifier

You will complete the function `classify_epoch(epoch,rate)` which takes an epoch of frequency data and the sampling rate, and returns the best classification for that epoch. (The returned value should be integers as follows:

1 for Stage 1 NREM

2 for Stage 2 NREM

3 for Stage 3/4 NREM

The body of this function should be the "rules" you determined in step 2.

The programming assignment for this problem set will test your `classify_epoch` code.

(Hint: This function may need to create a psd, normalize the psd, and sum over certain portions of the psd. Then, it might use one or more if statements to make a decision about sleep stage based on the sums obtained.)

Step 4: Test your classifier

Now that you have established these rules based on the example data, you want to make sure that rules hold. You will not submit anything for this part of the assignment – however, you may find you need to modify your `classify_epoch` code as a result of this step.

The first place to test your classifier is to make sure that the examples that you used to determine your rules are properly classified!

Run your classifier on the example stages data. The `classify_epoch` function doesn't depend on epoch size, so you can pass in all the data, or subdivide the data into smaller epochs (for example, 30 second bins). Because the data are already sorted by stage, you already know the answers you should be getting.

Writing this code yourself is excellent practice replicating what you would have to do if you had confronted data like this on your own. We recommend that you try that out! However, you can also run the provided function `test_examples(examples, srate)` to see your performance. You can look at the code to see what we did.

Classification is hard, especially with this limited data set. It's also theoretically possible that the researcher made errors when they classified the data. For these reasons, we are expecting you to reach a 75% percent level of success on the practice data. (You may get 100% correct with the examples set – but you used that data to set your rules. In the practice set, you will have data you haven't seen before.)

Now let's try that practice data set! It contains an hour of data. The practice data set is in a different format than the example set. With the example set, we gave single examples of each sleep stage. Here you have some real sleep data that will contain the different sleep stages (so just an array of EEG values and the sampling rate). Theoretically, the correct stages would be unknown – so you want to treat the data this way (meaning that you don't want to make new rules based on this data – you want to apply your rules here). For the problem set, we gave you the correct stages in the answers file so you can check your performance.

You can load the practice data set with the `load_eeg` and `load_stages` functions.

```
eeg, srate = load_eeg('practice_eeg.npz')
stages = load_stages('practice_answers.npz')
```

We have provided you with a `classify_eeg` function that takes the practice eeg file, breaks it into 30s epochs and runs `classify_epoch` (PLEASE DO NOT MODIFY THE `classify_eeg` FUNCTION – WE USE IT TO SCORE YOUR ASSIGNMENT. MAKE ALL MODIFICATIONS IN `classify_epoch`).

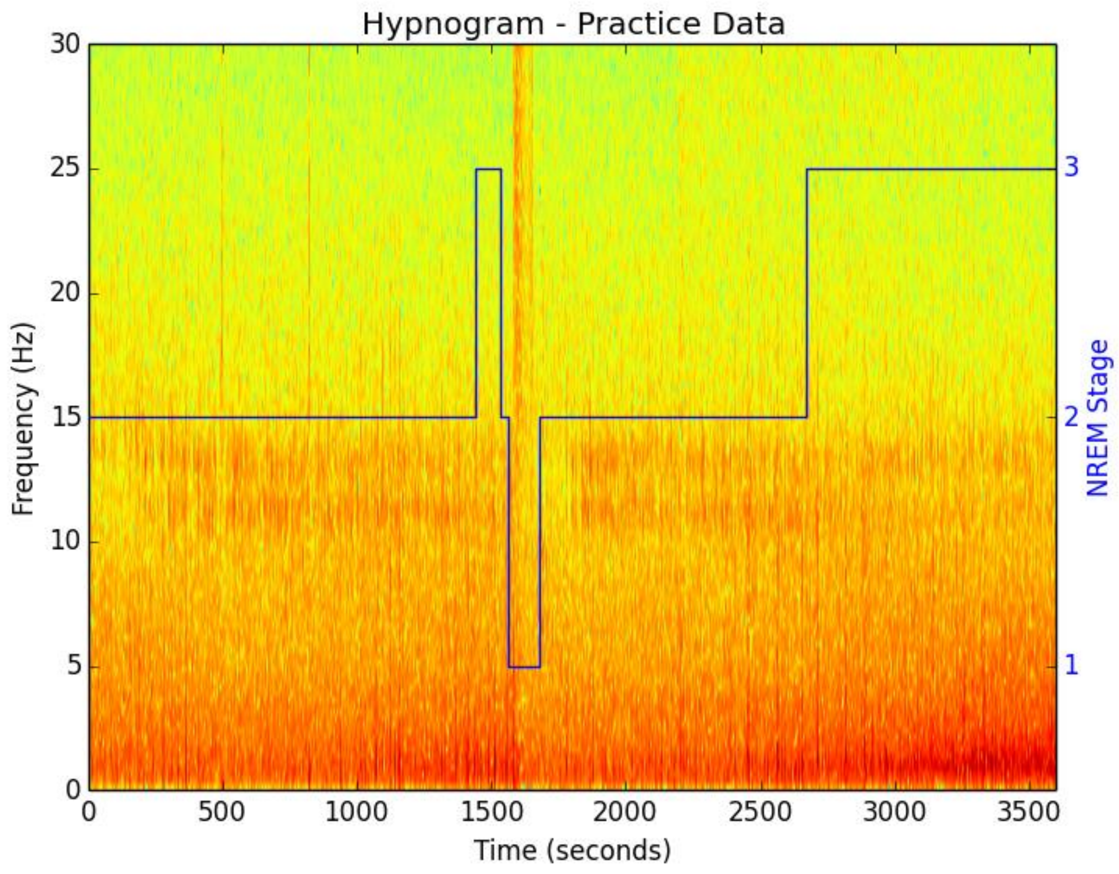
You can test your performance with the `classifier_tester` function. You should shoot for at least 75% success. You may need to tweak your classifier to improve your performance – but remember, you want to still classify the examples correctly!

There's nothing for you to submit here, but we'll run `classify_eeg` on the test data set (where you don't know the correct answers) to score your assignment.

Step 5: Displaying Your Results

Finally, we'd like you to create and plot your hypnogram. To really help the viewer understand your data, we'd like you to plot the hypnogram on top of the spectrogram. You will submit two plots as your Peer Assessment.

Below you see an example using the stages from the answers file. You should create your hypnogram with your classifier's values. Remember to convert your x-values to time (in seconds).



We've provided skeleton code in `plot_hypnogram` to help you use multiple y-axes on the same plot. For more information see: http://matplotlib.org/examples/api/two_scales.html

Hint: adding the parameter `drawstyle = 'steps'` to your plot command for the hypnogram will give you square steps for your plot.

Use your classifier on unknown data

We have also provided you with the `'test_eeg.npz'` dataset. This dataset contains an additional 50 minutes of data and can be loaded with the `load_eeg` function. For this data set, you will not be able to check your performance, but you will be able to create a visualization with the hypnogram over the spectrogram, as you did for the practice data set.

You will submit both figures for peer assessment.

Step 6: Complete the Assignment

Assignment problems:
To complete the assignments, we'd like you to create the final figure for the two data files, `practice_eeg.npz` and `test_eeg.npz.` You will submit the results of your classification code using the `submit_problem_set4.py` code (see **Submitting your code** below) and upload the results of your plotting routines using the Peer Assignment tab in Coursera (see **Submitting your figures for Peer Assessment** below). You will also complete some very short questions **Problem Set 4: Homework Questions** under the Quizzes tab in Coursera.

Submitting your code:
Once you have updated problem_set4.py to include your `classify_epoch()` function, you can submit this part of your problem set by opening the file `submit_problem_set4.py` in Spyder. [The submit code will call the `classify_eeg()` function (so don't modify it!) which calls `classify_epoch().]` With `submit_problem_set4.py` you can run it (press the green play button or hit F5) and it will prompt you for your email address and a "one time password". You get this one time password from Coursera. On the Programming Assignments page, at the top, you should see your own *Submission Password*. This is NOT your Coursera password.

If you have logged in correctly, the submit script will run your code on the `test_eeg` data set.

Submitting your figures for Peer Assessment:

To complete the problem set, you will also need to complete the **Problem Set 4 Peer Assessment.** Here you will be asked to upload the two final figure files (from `practice_eeg` and `test_eeg` data sets.). Please read the question prompts carefully to ensure that you submit the correct figure in the correct locations! You will need to save the figures (for inline figures, right click on the figure and select "Save As", for figures in their own window, just choose the save icon). You can save the files as png, jpg, gif or pdf for peer assessment.

The purpose of the peer assessment is to ensure that you are presenting your data in a meaningful way that can be easily interpreted by others. Therefore, these assessments are based on the format of your figure, including axes labels and the figure title. Each figure will be assessed in the following categories (1 - 3 pts possible per category): Please note, there are multiple ways to receive full credit – only examples are given:

The x-axis is properly labeled. The x-axis should be labeled and include units. The x-axis should be labeled with "Time" or something similar, with units of seconds or minutes. If seconds are used, the x-axis should span from 0 – 3600 seconds for the practice data set, and from 0 – 3000 seconds for the test data set. If minutes are used, the x-axis should span from 0 – 60 minutes for the practice data set, and from 0 – 50 minutes for the test data set.

0: The x-axis is unlabeled or both the label and units provided are incorrect.

1: Either the label or the units are incorrect or missing.

2: The x-axis is correctly labeled (as described above).

The y-axes are properly labeled:

- The y-axes should be labeled and include units (where appropriate).
- One y-axis should be labeled with "Frequency" or something similar, with units of Hz, cycles/sec, or something similar.
- One y-axis should be labeled with "NREM Sleep Stage" or something similar. No units are necessary.
- The values on the y-axis should correspond with the correct label. That is, the frequency values should range from 0 – 20 Hz (or more), and the sleep stage values should range from 1-3 (or be slightly larger, e.g. 0 – 4, to allow the hypnogram to be more easily seen.) The sleep stage y labels should not include half values, like 1.5 and 2.5

0: The y-axes are unlabeled or both the labels and units provided are incorrect.

1: One of the y-axes has missing or incorrect labels or units are incorrect or not provided (when applicable.)

2: The y-axis labels are correct but they do not match with the y-values (either the labels are swapped, or half values are included, or both.)

3: The y-axes are correctly labeled (as described above) and correspond to the correct values.

The figure is properly titled. There are lots of options for figure titles, they just need to make sense. For example, "Hypnogram", "Hypnogram and Spectrogram", "Sleep Stage", "Sleep Stage for Practice Data" are all acceptable titles.

0: The figure title is either missing or irrelevant.

1: The figure is descriptively titled.

Spectrogram data are clearly displayed. The range of frequencies displayed is appropriate (the maximum frequency displayed should be 50 Hz or less).

0: The spectrogram is unclear or has an inappropriate range.

1: The spectrogram is clear and has an appropriate range.

Hypnogram data are clearly displayed. All hypnogram stage values can be easily seen in the hypnogram, and the hypnogram spans the same x-values as the spectrogram.

0: The hypnogram is unclear, does not span all x-values, and not all stages can be seen.

1: Either the hypnogram does not span all x-values or not all stages can be seen.

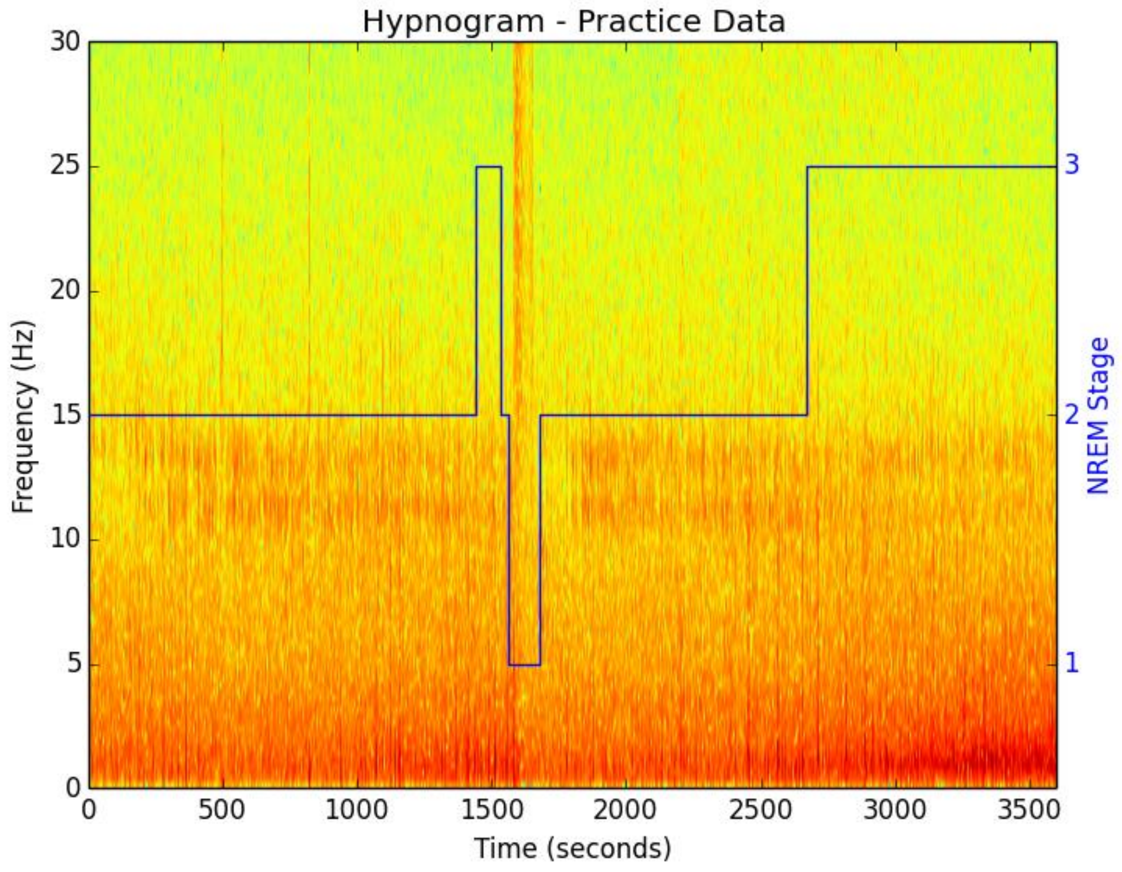
2: The hypnogram is clear, spans all x-values, and all stages can be seen.

The hypnogram and spectrogram data are displayed on the same plot.

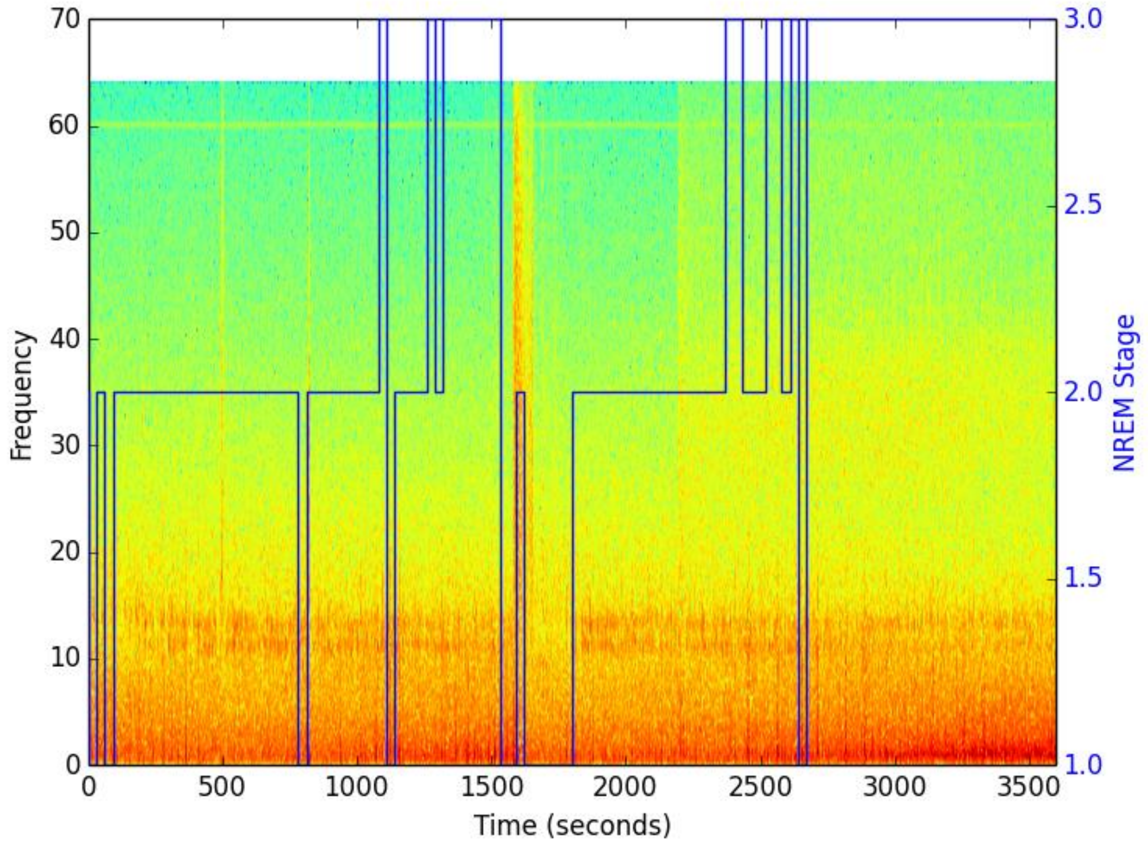
0: The hypnogram and spectrogram are displayed in separate figures or as two subplots.

1: The hypnogram and spectrogram appear on the same plot.

Here are some examples (these show the actual answers from the practice file, so your hypnogram should look different!):

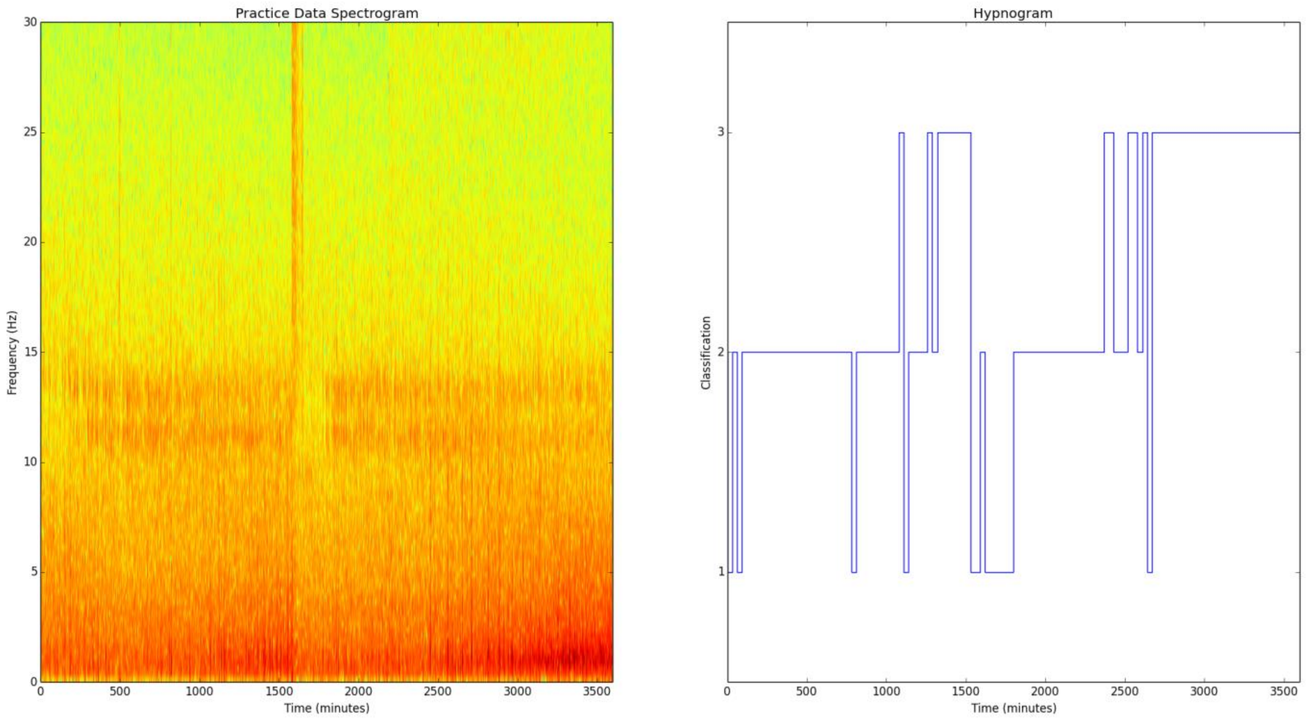


This figure would receive full credit. The axes are properly labeled and contain units (where needed). The title is descriptive. The spectrogram is clear. The hypnogram is clear, spans the x-values, and all stages can be easily seen. Both plots are on the same figure. Overall, the figure is clear.



This figure would receive 5/10 points:

x-axis:	2 pts.	The x-axis is properly labeled with units
y-axes:	1 pt.	The left y-axis does not contain units.
Title:	0 pt.	No title is provided.
Spectrogram:	0 pt.	The displayed frequency range is inappropriate.
Hypnogram:	1 pt.	The hynogram is not visible for stages 1 and 3 because they overlap the edges of the figure.
Same plot:	1pt.	The data are on the same plot.
		Total Score: 5 pts.



This figure would receive 6/10 points:

x-axis:	1 pts.	The units are incorrect on the x-axis ("minutes" but given as seconds).
y-axes:	1 pt.	The y-axis for the hypnogram is not descriptive.
Title:	1 pt.	Titles are descriptive
Spectrogram:	1 pt.	The spectrogram is clear and over an appropriate range of frequencies.
Hypnogram:	2 pt.	The hypnogram is clear and all values can be seen.
Same plot:	0 pt.	The data are on separate plots.
		Total Score: 6 pts.

Submitting your Homework Questions:

You will have one opportunity to correctly answer the homework questions by following the link on the Coursera page for Problem Set 4 Homework Questions. For that reason, we have provided you with the questions here so you can finalize your answers before entering the questions page on Coursera. It is probably best to complete and submit the programming assignment before answering the questions because some of the answers depend on your code working properly.

Each question is worth 1 point.

1. What was the sampling rate for the practice data set? (Please only provide the numeric value, and assume the units are samples/sec)

2. Based on the provided example data, REM sleep looks most similar to which stage of NREM sleep?

3. NREM stages 3 and 4 are dominated by frequencies in which band?

4. Sleep spindles, typically between 11-15 Hz, are most likely prominent in which sleep stage?

5. Which sleep stage was most prevalent during the test data set?