

## Exercises – Spectral Library Building and Searching Using SpectraST

### Natalie Tasman

These exercises take you through the process of building a spectral library from sequence search results of the yeast SILAC dataset we have been using, searching the same dataset against it, validating the results using PeptideProphet, and comparing them to the original sequence search results.

#### Step 1. Extracting confident IDs from sequence search results and library building

In this tutorial, we will build a spectral library from the Tandem-K search results of the yeast SILAC dataset we have been using. In the first step, we extract the confident identifications from the PeptideProphet output (i.e., the interact.xml file you created in the PeptideProphet tutorial).

Log onto the Petunia interface, and select the **SpectraST Tools** link. Then click on the tab **SpectraST Library Import**. In the pane **1. Specify File Format**, select **.pepXML (Sequence Search Result)** from the drop-down menu.

Now, click **Add Files** in the pane **2. Specify files to import**, browse to the directory class/SpectraST. To save you the trouble of moving files, the interact.xml file that you have created in the PeptideProphet tutorial (for the semitryptic search), as well as the mzXML files containing the query spectra, are already copied here. Select the interact.pep.xml file. (Note that when building spectral libraries, the mzXML files must be in the same directory as the pepXML file.)

In the pane **3. General Options**, type in “raw” in the **Enter name of output file** box; this will be the name given to output file. In the **Specify a dataset identifier** box, type in “course”; this will allow you to keep track of the sample source of your consensus spectra later on. In the **Specify a minimum probability to import** box, check that the value of 0.9 is set as the default. This is the minimum probability for an identified spectrum to be extracted by SpectraST.

Leave all other options as their defaults. Go to the bottom of the page and click **Import Library Files**. It will take about 3-6 minutes to run. During this time, SpectraST scans through all the identifications contained in the interact.pep.xml file, selecting only those above the probability threshold, goes back to the respective mzXML files to extract the query spectra, and then creates a raw spectral library of them. While the program is running, monitor the progress in the **Output so far** pane. You may have to click the **UPDATE THE PAGE** link at the bottom to force the page to refresh every now and then.

When SpectraST is done, the **Output so far** pane should contain information about the newly created raw library. How many spectra are extracted into the library? (10,787) How many distinct peptide ions do they represent? (4,531) How many spectra are identified to fully tryptic peptides? (10,573)

Un-hide the **Command Status** pane, which should have turned orange by now, and click to **view log file and output files**. Click on the link **c:\inetpub\wwwroot\ISB\data\class\SpectraST** to browse the directory. You will see that 5 files are created: raw.splib (a binary-format library used by SpectraST to search), raw.sptxt (a text-format equivalent of raw.splib for human viewing), raw.spidx (a library index on the precursor m/z value), raw.pepidx (a library index on the peptide ion), and spectrast.log (a log file). Click the **View** links to see how they look like. In particular, notice some useful information presented at the beginnings of the files raw.sptxt, raw.pepidx and spectrast.log. You probably want to use the “Stop” button of your browser to stop loading the enormous raw.sptxt in its entirety.

### Step 2. Building a Consensus Library

The second step is to build a consensus library from the raw library you just created. Note that in the raw library, some peptide ions are represented by more than one raw spectrum; in consensus building, these “replicate” spectra will be combined into one.

Again click on the tab **SpectraST Library Import** under **SpectraST Tools**. From the drop-down menu under **1. Specify File Format**, select **.splib (perform join/build actions on SpectraST libraries)**. In the pane **2. Specify files to build/join**, remove the interact.pep.xml file, then browse to the directory class/SpectraST to add the raw.splib file that you just built in Step 1. In the **3. Select Actions** pane, select **Consensus** in the **Select Build Action** drop-down menu (leave “join action” as default). Type in “consensus” in the **Enter name of output file** box. Leave all other options as defaults. Scroll down to the bottom and hit the **Import Library Files** button.

It should take less than a minute to run. SpectraST will create a “consensus” spectrum for each peptide ion that has multiple replicates in the raw library raw.splib. For peptide ions with only a single replicate, SpectraST will also include them in the final library after some spectrum processing.

When it is done running, check the **Output so far** pane to see the statistics. How many spectra are there in this library? (4,531) Note that this is the same as the number of unique peptide ions in the raw library, as the replicates are now combined. How many spectra are from single observations (see the NREPS line)? (1,830)

### Step 3. Evaluate the quality of the spectral library and applying quality filters

Usually, with a high probability cutoff of 0.9, most of the spectra contained in the consensus library are of decent quality, but occasionally there will be a few that are mis-identified by the sequence search engine in the first place, and/or highly impure. In this step, SpectraST will attempt to identify these spectra and (if you so choose) remove them from the library.

Go to the **SpectraST Library Import** page once again, and again select **.splib (perform join/build actions on SpectraST libraries)** as the file format. In the pane **2. Specify files to build/join**, remove the raw.splib file, and add the consensus.splib file. Select **Quality\_Filter** in the **Select Build Action** drop-down menu. Scroll down and you should notice a few more quality filter options. In this case we will use the defaults, so just go ahead and click **Import Library Files**.

SpectraST will subject the spectra to each of its quality filters, and determine if they fail any of them. In this case, we are asking SpectraST to keep all spectra regardless of quality, but indicate in the output library (“annotate”) which of these spectra have failed which filters, if any.

When it is done, un-hide the **Command Status** pane, and click to **view log file and output files**. Go to the class\SpectraST directory by clicking on the link **c:\inetpub\wwwroot\ISB\data\class\SpectraST**, and click to **View** the file spectrast.log. If you scroll down, you can see a log of what SpectraST has done in this session so far, including Step 1 through 3 in this tutorial. Towards the end there are information on the quality filters (prefixed with “QUALITY\_FILTER”) and at the very end there are some statistics. It tells you how many spectra would be left if you had selected various levels of quality.

How many spectra would be left if we set the quality level at 2? (4,221). How many would be left if we set the quality level at 3? (3,777) The level designations are as follows:

- Level 1: Remove impure spectra
- Level 2: Level 1 + spectra that have a spectrally similar counterpart in the library with a conflicting identification
- Level 3: Level 2 + spectra whose peptide sequence has no shared sub-sequence with any other peptides in the library
- Level 4: Level 3 + singleton spectra
- Level 5: Level 4 + inquerate spectra (with a user-defined quorum)

A utility called Lib2HTML can convert a SpectraST library to a web page for visualization. In the Petunia interface, you can run this program by going to the **SpectraST Tools** menu and clicking the tab **Lib2HTML**. Remove any other .splib file, add the newly created file consensus\_quality.splib, and hit **Convert Library Files**. When it is done, un-hide the **Command Status** pane, and click to view the html file. Here you can click on the links in the **LibID** column (far right) to see the spectra themselves. The **Status** column specifies the least stringent among the failed quality filters for that entry. (“Normal” implies the spectrum passes through all quality filters.)

Click on some “Singleton” spectra and some “Normal” spectra. Which type of spectra appears to be of high-quality (in terms of criteria such as long consecutive ion series, fewer noise peaks, fewer unassigned peaks)? (*Normal*)

Recall that in the original Tandem-K search, there are some identifications mapped to decoy sequences, which we can be sure are false positives. We have not explicitly removed them during our consensus building process. They are indicated by a “REV” prefix in the

## SpectraST -- Tutorial

protein name. Use the Find function of your browser to count the number of spectra identified to “REV” proteins. How many do you count? (22) How many library entries do you estimate to be false positives? ( $22 \times 2 = 44$ ) What is the estimated identification error rate of your library? ( $44 / 4,531 = 0.01$ )

A 1% error rate is certainly quite good, but we can do better. Of course, if we desire, we can first remove spectra identified to “REV” proteins, which we are certain to be false, and cut the error rate roughly in half. However, for this tutorial, let’s keep these spectra around for educational purpose.

Another way to reduce the false positives is by using SpectraST’s quality filters. Now, revisit the entries with “REV” proteins using the Find function of your browser. This time, determine how many of the 22 entries have a **Status** of “Impure” or “Conflicting\_ID.” (8 and 6, respectively) This implies that if we remove these questionable spectra, we can cut the number of falsely identified spectra in our library by about half. Click on a few of the “Impure” entries (the link in the LibID column) to get a feeling of what SpectraST considers an impure spectrum. Also click on a few of the “Conflicting\_ID” entries and see if their identifications look questionable to you.

As you are counting, you may notice that the rest of the entries with “REV” proteins are marked “Inquire\_Unconfirmed” and none of them “Normal.” In other words, if we are more conservative and choose to filter at quality level 3, we would have removed all of the known false positives, and by our assumption, all false positives as well. However, this comes with a price: we will also have to sacrifice some correctly identified spectra.

For this tutorial, let’s decide that quality level 2 is where we will strike our balance between coverage and quality. Perform the quality filter again, this time telling SpectraST to actually remove “Impure” and “Conflicting\_ID” spectra. To do so, go back to the **SpectraST Library Import** page again, and select **.splib (perform join/build actions on SpectraST libraries)** as the file format. In the pane **2. Specify files to build/join**, browse to select the file consensus.splib (remove any other files already there). Select **Quality\_Filter** in the **Select Build Action** drop-down menu. This time, name your output library “consensus\_Q2” in the **Enter name of output file** box in the **4. General Options** pane. In the pane **6. Quality Filter Options**, select **2: ...+spectra with look-alikes having conflicting IDs** in the drop-down menu **Quality Level to Remove**. This tells SpectraST to remove all spectra that are found to be impure. Click **Import Library Files**.

How many spectra are there in the resulting library consensus\_Q2.splib? (4,221) Note that this is the number promised at the end of the spectrast.log file. For the purpose of this course, it is didactic to be able to see the library spectra about to be removed, but in real-life library building, one does not always need to apply the quality filter in this two-step manner (flag everything first, determine a suitable quality level, then remove). For most applications, a quality level of 2 is generally suitable, although more advanced users may want to go through this process to find the right balance between coverage and quality.

### Step 4: Concatenating the yeast library to a decoy library

The spectral library you just created is extremely small and unsuitable for spectral searching. First, the number of candidates considered in each search is not large enough to form a reliable background statistically. Second, since we are about to search the same dataset we used to create the library against itself, we need to have some negative control. Put differently, we want to give the search engine some room to make mistakes and see if it makes them, thereby giving us a sense of how reliable our spectral search is.

To do so, we can concatenate our yeast library to a much larger human library. A “decoy” library of 16,649 human spectra that do not have any isobaric identical or homologous counterparts in the yeast library has been created for you.

To join your yeast library to the decoy library, go to the **SpectraST Library Import** page, select **.splib (perform join/build actions on SpectraST libraries)** again. In the **2. Specify files to build/join** pane, remove all previously added files, and add the files `consensus_Q2.splib` and `human_decoy.splib` from the directory `class\SpectraST`. In the **3. Select Actions** pane, select **None** in the **Select Build Action**, and **Union** in the **Select Join Action** menu. Type in “`consensus_Q2_plus_decoy`” in the **Enter name of output file** box. Scroll down and hit **Import Library Files**.

While you are waiting for it to run (it probably will take a couple minutes), a word of caution about the use of decoy libraries in spectral searching. Unlike in sequence searching, for which a common and accepted practice is to use decoy database searching to establish false discovery rates, we do not yet have a well-established method of decoy spectral searching that is proven. Until a thorough investigation of decoy spectral searching is done, using this approach to calculate accurate false discovery rate is not recommended. However, using it as a sanity check or as a rough estimate, as in this tutorial, is certainly not a bad idea.

### Step 5: Searching the original dataset against this spectral library

Now we are ready to re-search our original dataset against our newly built spectral library. To speed things up and to make it more interesting, we are going to search *only those spectra that were not identified in the original Tandem-K search (with probability no less than 0.9)*. These were also the spectra that we did not use to build our libraries. Two mzXML files: `OR20080317_S_SILAC-LH_1-I_01_FILTERED.mzXML` and `OR20080320_S_SILAC-LH_1-I_11_FILTERED.mzXML` are created for you that only contain spectra unidentified by Tandem-K.

Click on the **Home** link at the top. Then in the **Analysis Pipeline** pane, select **SpectraST** in the drop-down menu. Click on the **Analysis Pipeline** link at the top, and click on the **SpectraST Search** tab. To set up the search, first click **Add Files** in the first pane **1. Specify mzXML Files**, browse to the directory `class\SpectraST`, and select the 2 filtered files: `OR20080317_S_SILAC-LH_1-I_01_FILTERED.mzXML` and `OR20080320_S_SILAC-LH_1-I_11_FILTERED.mzXML`. In the second pane **2. Specify Library File**, select `consensus_Q2_plus_decoy.splib` in the same directory. In the pane **3. Specify a sequence**

## SpectraST -- Tutorial

**database to be printed to the output file for downstream processing**, browse to the directory `class\database`, and select `yeast_orfs_all_REV.20060126.fix.fasta`.

We are using all default options, so we are all set to go. Scroll down to the bottom of the page and hit **Run SpectraST**. Monitor the progress by viewing the **Output so far** pane.

The search should take about a minute or two. In the meantime, check out the page <http://www.peptideatlas.org/specilib/>. This website links to all the spectral searching for proteomics projects that we are aware of. You will find download links to spectral libraries and spectral library searching tools here, so be sure you check back for updates. We will be posting various spectral libraries derived from the PeptideAtlas project here.

When the search is done, run PeptideProphet as you normally would. Click on the **Analyze Peptide** tab, and add the two pepXML files containing the search results: `OR20080317_S_SILAC-LH_I-I_01_FILTERED.xml` and `OR20080320_S_SILAC-LH_I-I_11_FILTERED.xml`. Name the output file `interact-spec.pep.xml` (so as not to overwrite `interact.pep.xml`). Check the option **Use accurate mass binning**. Then go ahead and hit **Run XInteract** at the bottom of the page. Un-hide the **Command Status** pane, and **View** `interact-spec.shtml` when it is done.

Filter the identifications for probabilities above 0.9. How many hits remained? (*1,700*) Click on a few of them and see if they look good to you. Recall that all of these identifications were *missed* by Tandem-K in the previous search. How does the quality of these spectra compare to what you are used to seeing in previous sessions? What does this say about the sensitivity of spectral searching compared to sequence searching?

Feel free to play with the displaying options to the left of the spectra. The coloring scheme for the spectrum viewer is as follows. In the library (top) spectrum: *Red lines* = peaks assigned to known ions; *Blue lines* = unassigned peaks; *Red peak labels* = the ion assignment (you can customize what ion types to display on the left panel); red color indicates that this peak is present in the query spectrum as well; *Black peak labels* = indicates that this peak is missing in the query spectrum. In the query (bottom) spectrum: *Red lines* = peaks that match assigned peaks in the library spectrum; *Black lines* = peaks that do not match any assigned peaks in the library spectrum. In the ion table underneath the spectra: *Red colored boxes* = ions that are present in both spectra; *Pink-colored boxes* = ions that are present in the library spectrum only; *White colored boxes* = ions that are not present in either spectra.

Visualize the PeptideProphet models by clicking on any of the probabilities. Do they look reasonable? What is the sensitivity and error rates are a probability cutoff of 0.9? (*Sensitivity = 0.739, Error = 0.019*) What is the expected number of incorrect identifications among your positives (hits with probability  $\geq 0.9$ )? ( $1,700 \times 0.019 = 32$ )

As a sanity check, let's see if SpectraST finds any positive identification from the decoy (human) library. Because we only specified a yeast database for subsequent TPP processing, TPP will fail to map human peptides in the decoy library to any proteins. With the probability filter still on, filter the SpectraST hits for "NON EXISTENT" proteins. How many did you find? (*0*)

## SpectraST -- Tutorial

Given the decoy (human) library is about 4 times larger than the target (yeast) library, what does it say about the accuracy of SpectraST? Does it give you additional confidence in these identifications that SpectraST found but Tandem-K missed?

Lastly, recall that we deliberately left some known false positives in the spectral library. These are spectra identified to “REV” proteins in the original Tandem-K search. With the probability filter still on, filter the SpectraST hits for “REV” proteins. How many did you find? (6) These identifications were of course wrong, but we cannot blame the spectral search engine for them. Here, we are simply propagating the error we made at the sequence searching step. That’s why it is important to use quality filters when we build spectral libraries to minimize such errors.