Unrestricted Search Tutorial

Sam Payne

Winter 2007

This tutorial explains the unrestricted search option of Inspect (Tsur 2005, Tanner 2008), and assumes a completion of previous Inspect tutorials. An unrestricted search is a powerful tool for finding evidence of post-translational modifications in MS/MS spectra. Rather than identifying only specified modifications, an unrestricted search finds peptides with any PTM (up to a size limit). Using this feature properly requires an understanding of several practical and technical issues.

- Database
- Inspect Commands
- Inspect Run
- Post-Processing

This will not cover all the features of an unrestrictive search, but enough so that you can successfully run the search. Additional information can always be found in the documentation included in the Inspect folder “docs.”

Introduction:

There are now two papers explaining the theory of both the search and the post-processing for the unrestrictive searches. I suggest that you read both of those, cited at the end of the tutorial. There are a few things that I need to explain. First, the unrestrictive search is has the best power on large samples, more than 100,000 spectra. If you read the Tanner paper you will see that we do a lot of statistics based that are helped by big numbers. If you have a single dataset of 10,000 spectra (of which only 20-30% have annotations), then you might not find what you're looking for. Next, as you might expect, the search produces results for mass shifts that are may not be previously documented or commonly understood (that's the hope right?). This remains for the user to interpret. Finally, we have discovered that a large number of the modifications in any sample are due to *in vitro* handling damage, so understand that those are not biologically significant.

Step 1: New Database

The most significant difference in an unrestricted search is a substantial increase in running time. The algorithm used to consider a modification of any size on any residue takes much, much longer than a regular Inspect search. To partially ameliorate this problem, we use an abbreviated database containing only the proteins found in an unmodified search. The justification for using an abbreviated database is that proteins containing modifications will also have unmodified peptides (which can be found in a regular search). This assumption holds even for the most heavily modified protein samples (human lens crystallins).

Two scripts process the results of an unmodified Inspect search to make the abbreviated database. First, run the pvalue script as previously described. Next, run the Summary script with the '-b' option. In our lab, we include all proteins with at least two peptides and ten spectra.

```
> python PValue.py –r RawResults.txt –w PvaluedResults.txt –p 0.05

> python Summary.py –r PvaluedResults.txt –d Database\myDB.RS.trie –b
Database\AbbrevDB.trie –e 2 –m 10
```

Of course, here you will want to include a shuffled version of the abbreviated database. The summary script weeded most of the fake proteins. So the database to include in the input script might be named something like "AbbrevDB.RS.trie".

Step 2: Inspect Commands

The input file is changed to denote an unrestricted search. You must specify how many modifications per peptide. As previously stated, allowing multiple PTMs per peptide is not recommended. This is especially true for an unrestricted search. Almost any spectra can find a high-scoring match if given multiple unrestricted PTMs. Add the following lines to the infile:

```
blind,1
mods,1
```

Note, you cannot specify both the blind command, and the “mod” command to enumerate specific PTMs, unless they are fixed PTMs like C+57. The search modes are distinct and mutually exclusive.

Step 3: The Inspect run

The unrestricted search is a prime candidate for supercomputing. At our lab, we use several supercomputer grids to speed up the analysis. Again, we cannot provide scripts that will work on a particular grid. Please consult your grid administrator.

One suggestion that we can give, is that you might want to split up a single spectral file into multiple jobs, meaning that one job will process spectra 0 through 6000, and another will process 6000 through 12000, etc. Altering the input file as below will restrict the spectra searched.

```
spectra, ../spectraDir/16.mzXML,18000,24000
```

Step 4: Post processing

The PValue calculation as described in the first tutorial works very well for an unmodified searched. However, it does not work with unrestricted PTM searches. We've developed an effective filter (Tanner 2008), encoded by PTMAnalysis.py. I want you to get familiar with using the usage information of the script, so I’m largely omitting an explanation of the arguments here. I actually will explain only a few things. First, note that –B and –d are two different databases. The –B option is not the database used for the search in step 3. It is some huge database, like swiss prot. Read the paper for details. Second, when you get the output, sort by the column “SitePValue” and make your cutoffs where you feel comfortable.

```
> python PTMAnalysis.py
–r InspectResultsDir
–w FinalResults.txt
–d Database\AbbrevDB.FS.trie
–s ..\mzxmlDir
–S 0.5
–B Database\uniprot_sprot.trie
```

```
–k KnownPTM.txt
```

(For clarity, I have put each argument on a separate line; in reality you need to type them into the same line at the command prompt.) This script may take a while to complete. Advanced users can read the code-level documentation to figure out how to run portions of the script in parallel. Also users need to install the libsvm in their root C directory (or home dir for linux). Link at http://www.csie.ntu.edu.tw/~cjlin/libsvm/

References

Tsur et al (2005) Identification of post-translational modifications by blind search of mass spectra. Nat Biotechnol 23:1562-7

Tanner S, et al. (2008) Accurate Annotation of Peptide Modifications through Unrestrictive Database Search. J Proteome Res. 2008 Jan;7(1):170-81.