A Strategy for the Efficient Identification of Modified Peptides
I want to discuss the most efficient way to get as many matches as possible from an LC-MS/MS run. That is, to find matches to peptides with unsuspected chemical or post-translational modifications, with minor sequence variants, such as SNPs, and peptides which are the products of non-specific cleavage.

This strategy is not applicable to identifying peptides from protein that have little similarity to those in the database. As the databases fill up, this is becoming less common. If you are in this situation, the primary tool has to be de novo of high quality MS/MS spectra

Neither is it applicable to investigations focused on a particular modification, such as phosphorylation. The key here would be a targeted experiment. Maybe using a neutral loss scan to identify and select the phosphorylated peptides, or an IMAC column to isolate the phosphopeptides
Search strategy

1. **Standard Mascot search**
   Returns the easy matches

2. **Error tolerant search**
   Returns additional matches, but only for proteins where we have at least one good peptide match already
   Limited to a single additional SNP or modification per peptide

3. **De novo**
   If data very high quality, can return novel full-length peptide sequences
   Use Blast to find likely parent proteins
   More often, returns partial / ambiguous peptide sequences

4. **Error tolerant tag search**
   To find matches to
   1. Isolated peptides that have a SNP or unsuspected modification
   2. Peptides with multiple SNPs or unsuspected modifications
   (No reason to expect additional matches from a standard tag search)

If you simply want to get as many identifications as possible, so as to minimise the number of unmatched spectra and maximise protein coverage, you might come up with a strategy similar to this. I’ll now go through the four steps in some detail. Step 1 is, of course, a standard Mascot MS/MS ions search.
1. Standard Mascot search

Why not try to get everything in a single search?

<table>
<thead>
<tr>
<th>plc dataset on dual processor 2.8 GHz P4</th>
<th>peptides</th>
<th>identity</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLE</td>
<td>tested</td>
<td>matches</td>
<td>threshold</td>
</tr>
<tr>
<td>trypsin</td>
<td>7.5E+07</td>
<td>399</td>
<td>41</td>
</tr>
<tr>
<td>semi-trypsin</td>
<td>1.2E+09</td>
<td>379</td>
<td>53</td>
</tr>
<tr>
<td>none</td>
<td>1.0E+10</td>
<td>299</td>
<td>62</td>
</tr>
</tbody>
</table>

Above timings are for Oxidation (M) as only variable modification. If also include Phospho (STY), Me-ester (DE), Me-ester (C-term), Acetyl (N-term), pyro-glu (N-term Q), pyro_gln (N-term E), the trypsin time goes from 10 mins to 1357 mins!

Should we try to get as many matches as possible in the first pass search? Well, let's look at some numbers for a typical ion trap dataset when we search using loose trypsin, semi-specific trypsin, and no enzyme specificity.

As you can see, the search time increases by an order of magnitude as we go from trypsin to semi-specific trypsin, and a further order of magnitude as we go to a completely non-specific search.

The reason is simple, the search space, that is the number of candidate peptides, is increasing by a factor of 10 each time. Having to wait 10 or 100 times as long for the results is bad enough. A more fundamental problem is that the significance threshold score is a function of the number of candidates, so this increase by 10 each time, and we lose marginal matches. Unless you have a high level of non-specific peptides in the sample, you lose more than you gain.

So, doing a no-enzyme search in Mascot is not a good idea unless there is a very high level of non-specific peptides. Semi-trypsin is almost always a better choice if the peptides came from a tryptic digest. Only use no enzyme if the peptides are not the products of a deliberate enzyme digest, e.g. MHC peptides or endogenous peptides.

Identical considerations apply to modifications. If we go from 1 variable mod to 7, the search time is even worse than for no enzyme. This is because of the combinatorial explosion. Having to test all the combinations and permutations of these variable mods.

So, the answer is no, do not try to get as many matches as possible in the first pass search. It just makes the search very slow and very insensitive.
Search strategy

1. Standard Mascot search
   Returns the easy matches

2. Error tolerant search
   Returns additional matches, but only for proteins where we have at least
   one good peptide match already
   Limited to a single additional SNP or modification per peptide

3. De novo
   If data very high quality, can return novel full-length peptide sequences
   Use Blast to find likely parent proteins
   More often, returns partial / ambiguous peptide sequences

4. Error tolerant tag search
   To find matches to
   1. Isolated peptides that have a SNP or unsuspected modification
   2. Peptides with multiple SNPs or unsuspected modifications
      (No reason to expect additional matches from a standard tag search)

If you want to get as many identifications as possible, as efficiently as possible, the first
pass search must be kept simple. Usually, strict or loose trypsin. Zero or one variable
modifications. Certainly not more than two unless you know for sure they really are present.
Step 2 of our strategy is an error tolerant search. This is the efficient way to find unusual
modifications, as well as variations in the primary sequence and peptides from non-specific
cleavage
2. Error tolerant search

First pass - simple search of entire database
- Minimal modifications
- Enzyme specificity

Second pass - exhaustive search of selected protein hits
- Wide range of modifications
- Look for SNPs
- Relax enzyme specificity

All the protein hits found in the first pass search are selected for an exhaustive second pass search.

Because only a handful of entries are being searched, search time is not an issue.

For modifications, an error tolerant search looks for one unsuspected modification per peptide in addition to those mods specified as fixed or variable. This is sufficient because it will be very, very rare to get two unsuspected mods on a single peptide.

The error tolerant search also looks for sequence variants, such as single nucleotide polymorphisms (SNPs) or sequencing errors.

You can remove enzyme specificity completely, but you have to ask yourself whether you would believe a match that was doubly non-specific.
I think in most cases the answer is no. Our experience is that the levels of non-specific peptides are very low, less than 3%, unless there is something seriously wrong with the trypsin or the protocol. This is also the conclusion of a very careful study by Matthias Mann’s group. So, in general, I prefer to use semi-trypsin in an error tolerant search.
In the current version of Mascot, an error tolerant search is literally a second pass search. You have to select the protein hits you want to search by manually checking them off in the results report.
In Mascot 2.2, we will make the process integrated and automatic. You just have to check the Error tolerant box on the search form. This will perform a first pass search using the enzyme and modifications you specify in the search form. It will then automatically perform an error tolerant search on all of the proteins that contain significant peptide matches. David will say more about this in a later talk.
We need to do more work on trying to filter out the unlikely matches. One rule we plan to introduce is that you can’t have an unsuspected modification in a non-specific peptide. Another is to ignore a modification if it only gives a tiny increase in score over an unmodified and specific peptide. It will still be necessary to decide between alternative assignments of observed mass differences.
The good news about the error tolerant search is that it substantially increases the number of matches. In this particular hit, from 14 to 22.
The bad news is that the sequence coverage hardly changes, 41% to 42%
If we look more closely, we see that this is just an additional 5 residues here, KYMFR. However, when we look at the matches, there is only one match spanning this peptide and it is a weak and dubious match, requiring two mods and a SNP. In all honesty, I would not want to accept this match, so the coverage is actually unchanged. For this particular protein, the error tolerant search just gives us additional matches to the same peptides we saw in the standard search.
If you don’t want to spend a lot of time studying the matches from an error tolerant search and deciding which you accept, you can use it as a quick way of spotting whether there are modifications which should be included in the first pass search as variable mods. Here’s a nice example. Lots of matches for a modification of 43 Da, almost certainly carbamylation.
Adding this in as a variable mod increases the number of peptide matches for hit 1 from 57
To 92. However, coverage only increases from 57% to 64%. Basically, just one new peptide, that was only present in carbamylated form.
This observation is in agreement with a very careful study by Ken Parker and colleagues. As you go deeper, you tend to find modified versions of peptides that you already identified. They did detailed manual validation of matches to non-specific peptides and found approximately 2% were semi-tryptic and zero were fully non-specific.
## Search strategy

1. **Standard Mascot search**  
   Returns the easy matches

2. **Error tolerant search**  
   Returns additional matches, but only for proteins where we have at least one good peptide match already  
   Limited to a single additional SNP or modification per peptide

3. **De novo**  
   If data very high quality, can return novel full-length peptide sequences  
   Use Blast to find likely parent proteins  
   More often, returns partial / ambiguous peptide sequences

4. **Error tolerant tag search**  
   To find matches to  
   1. Isolated peptides that have a SNP or unsuspected modification  
   2. Peptides with multiple SNPs or unsuspected modifications  
   (No reason to expect additional matches from a standard tag search)

---

Where can we go from here? Well, maybe there are some peptides which can’t be picked up by the error tolerant search. Maybe a peptide that spans a splice site or a peptide with a modification that is not in our list of modifications. The next step is de novo.
De Novo - Issues

Database : NCBInr 20060518 (3647739 sequences)
Enzyme : Trypsin/P
Fixed modifications : Carbamidomethyl (C)
Variable modifications : Acetyl (N-term), Oxidation (M)
Peptide Mass Tolerance : ± 0.8 Da
Fragment Mass Tolerance : ± 0.4 Da
Max Missed Cleavages : 2

<table>
<thead>
<tr>
<th>Peptide mass</th>
<th>Database search</th>
<th>De novo</th>
</tr>
</thead>
<tbody>
<tr>
<td>~ 1000 Da</td>
<td>~ 6 x 10^5</td>
<td>~ 1 x 10^8</td>
</tr>
<tr>
<td>~ 2000 Da</td>
<td>~ 5 x 10^5</td>
<td>~ 4 x 10^19</td>
</tr>
</tbody>
</table>

The problem with de novo is that the search space is huge. If we assume tryptic specificity, the bigger the peptide, the fewer the candidates in a database search. With de novo, the number of candidate sequences grows geometrically with peptide length. In reality, things aren’t so bad. Any practical de novo algorithm explores only a small portion of this search space. Nevertheless, you cannot expect to get de novo solutions from large peptides unless the signal to noise and mass accuracy are both very good.
Another important factor is coverage. It is hard to over emphasise the importance of getting both N-term and C-term matches for a stretch of de novo sequence to be reliable. This is a particular problem with large peptides, where the spectrum is often only good at (say) the low mass end. If the C-term ladder and the N-term ladder do not overlap, this is a much, much less constrained situation.
De novo is implemented in Mascot Distiller, because it requires very reliable peak picking. The starting point can be any MS/MS scan that has been processed to create a peak list. Right click the peak list node in Dataset Explorer and choose ‘de novo Search’, or choose the de novo button from the toolbar when a Summed Scans node is selected.
Good signal to noise and good mass accuracy are critical for successful de novo sequencing; much more so than in database searching. GIGO (garbage in - garbage out) is guaranteed.

In a de novo solution, i always represents I or L. q represent Q or K, when the mass tolerance does not allow these residues to be distinguished, although K is assumed at the C terminus of a peptide when tryptic specificity applies. f represents F or Met-Ox.

Ambiguity is indicated by a dash in the sequence. The tooltip shows details of the ambiguity in square brackets, using pipe symbols to separate alternatives. Note that the order of the pairs and triplets is undefined, so that SP could also be PS.

Although the example shown here looks very different to the Mascot database match, they are actually in perfectly agreement. Some uncertainty is unavoidable in de novo, because the search space is so very much larger. For example, the score hardly changes when DA is replaced by SV.
If we look at details of these two matches, we can see why. There is a y ion peak at 488.214 to support the sequence being DA, but there is also a y ion peak at 516.192 to support the alternative. This is just the nature of de novo on non-ideal data.
To de novo sequence a complete peak list collection, or the peak lists in the currently displayed TIC range, use the context menu obtained by right-clicking the root (world) node.
You can then browse down the tree, looking for cases where the database search failed and de novo has a high score.

This looks like a promising case. The Mascot search didn't get a significant match, but de novo has a very high score.

But, is it right? And, how do we resolve the ambiguity at the N-terminus?
Search strategy

1. Standard Mascot search
   Returns the easy matches
2. Error tolerant search
   Returns additional matches, but only for proteins where we have at least
   one good peptide match already
   Limited to a single additional SNP or modification per peptide
3. De novo
   If data very high quality, can return novel full-length peptide sequences
   Use Blast to find likely parent proteins
   More often, returns partial / ambiguous peptide sequences
4. Error tolerant tag search
   To find matches to
   1. Isolated peptides that have a SNP or unsuspected modification
   2. Peptides with multiple SNPs or unsuspected modifications
      (No reason to expect additional matches from a standard tag search)

This brings us to step 4 of our strategy. As long as we are not dealing with an un-sequenced genome, the best way to test a de novo solution is an error tolerant tag search. This can often get a match even when there are multiple differences between the analyte peptide and the database sequence.
Here’s another example, from the Orbitrap data, where the Mascot database search has failed to find a match.

The de novo solution is not a great score, and there’s ambiguity at each terminus.
Right click the solution and choose Mascot search from the context menu. Note that we have already toggled the tag type to error tolerant.
Distiller populates the query field with the tags taken from the non-ambiguous parts of the de novo solution. We submit the search …
And back comes the result. Note that the results from this most recent search have replaced the original database search. You can switch back to the previous results by selecting them on the searches tab.

This match looks very promising. It’s a high score, and it’s a protein to which we already have other good matches. Notice that the de novo solution wasn’t bad, but it was reversed. TTYDiPEi should have been iEPiDYTT. Unless the de novo manages to reach a terminus, there’s a 50:50 chance that it will be the wrong way round.

If we right click and choose to view the full Mascot report in a browser …
The reason we didn’t get a match from Mascot is that there is a modification, giving a delta of 634 Da. The peptide forms the protein N-terminus.

If we click on the hyperlink to see the peptide view …
The match was obtained by placing this modification delta on or close to the N-terminus. Remember that this peptide forms the N-terminus of the protein, so in all probability, the initiator Met is lost in the mature protein, making our actual mass delta 765.33. To further complicate the picture, the annotations for this protein report that the N-term Alanine is normally acetylated. If so, our unknown mod is actually 723.32.

None of these deltas correspond to anything in Unimod, which is why this match wasn’t picked up by the error tolerant search. This looked like a very solid match to me, so I started trying to figure out what the mod might be. Then the penny dropped. The delta is awfully close to the m/z value, which immediately suggests a precursor charge error.
If we go back to the survey scan, this is what we find. A 3+ peptide at 723.6803 and a 2+ peptide at 722.8604. The instrument thought it was going for 723.99, so Distiller used the 3+ peptide, which is both the closest and the more intense. If we take the correct mass and charge …
We get the correct match from Mascot and even the de novo falls straight out. The initiator Met has indeed been removed and the Alanine acetylated. Not an unknown modification after all, but nice to get to the bottom of a small mystery. The take home message is that de novo plus sequence tag can often take you further than an error tolerant search of the uninterpreted data.
Search strategy

1. **Standard Mascot search**
   - Returns the easy matches

2. **Error tolerant search**
   - Returns additional matches, but only for proteins where we have at least one good peptide match already
   - Limited to a single additional SNP or modification per peptide

3. **De novo**
   - If data very high quality, can return novel full-length peptide sequences
     - Use Blast to find likely parent proteins
   - More often, returns partial / ambiguous peptide sequences

4. **Error tolerant tag search**
   - To find matches to
     1. Isolated peptides that have a SNP or unsuspected modification
     2. Peptides with multiple SNPs or unsuspected modifications
   - (No reason to expect additional matches from a standard tag search)

So, there we have it. Four powerful tools to help us find modified peptides. The challenge going forward is to make the workflow more integrated. It is still a bit too manual for large data sets. The other thing we need to address is the speed of an error tolerant tag. It would be great if we could find a way to speed this up so as to allow them to be fired off automatically for every de novo solution