INSPECT

The exposition is based on the following sources, which are all recommended reading:

1. Tanner et al.: InsPecT: Fast and accurate identification of post-translationally modified peptides from tandem mass spectras, Bioinformatics, 2001, 17, 1, 13-21

INSPECT

Reliable identification of post-translational modifications is key to understanding various cellular regulatory processes. InsPecT identifies post-translational modifications using tandem mass spectrometry data. It constructs database filters that proved to be very successful in genomics searches. Given an MS/MS spectrum S and a database D, a database filter selects a small fraction of database D that is guaranteed (with high probability) to contain a peptide that produced S.

InsPecT uses peptide sequence tags as efficient filters that reduce the size of the database by a few orders of magnitude while retaining the correct peptide with very high probability.

INSPECT (2)

In addition to filtering, InsPecT also uses novel algorithms for scoring and validating in the presence of modifications, without explicit enumeration of all variants. Inspect identifies modified and unmodified peptides with better or equivalent accuracy than other database search tools while being two orders of magnitude faster than SEQUEST, and substantially faster than X!TANDEM on complex mixtures.

INSPECT (3)

Post-translational modifications make the interpretation of MS/MS spectra hard. Even a single change in the peptide sequence due to post-translational modifications and mutations shift spectral peaks significantly. Enumerating all possible mutations and modifications in the database makes the database prohibitively large, but does not seriously affect de novo approaches.

Consequently, the current approach to identifying modifications is based upon a virtual *on-the-fly* enumeration of modifications of all candidate peptides. This approach is still computationally expensive, so that a search for modifications remains limited to smaller databases, or spectra of particular interest.

INSPECT (4)

The problem is similar to one faced by the genomics community in their search for sequence similarities. In order to find distant homologs, the alignment scoring had to be more sophisticated (and computationally expensive) thus making the database search too expensive to be a routine tool in the laboratory. The problem was solved using database filters that quickly eliminated much of the database, while retaining all true hits

InsPecT uses filtration to identify modified peptides using database search. The aggressive (but accurate) filtration allows InsPecT to apply more sophisticated and computationally intensive scoring to the few remaining candidates. Indeed, if the database is reduced to a few peptides, one can afford to consider a rich set of post-translational modifications (PTMs) for every peptide in the filtered database. Also, the decreased number of candidates reduces the possibility of a high score being achieved by chance.

INSPECT (5)

InsPecT has the following features:

1. Tag-based filters using a novel de novo interpretation algorithm that works in the presence of modifications and poor spectral quality.

2. A fast trie-based search for scanning the database with sequence tags.

3. A dynamic programming technique to identify candidate peptides with modifications without explicit enumeration of peptides.

4. A scoring algorithm that reflects peptide fragmentation patterns, and a novel quality score based on several complementary features.

Tag generation

Tag generation is a modified (but not simpler) version of de novo spectrum interpretation. We construct a directed acyclic graph where each node corresponds to a prefix residue mass (PRM), and each edge corresponds to a (possibly modified) amino acid. A path in the graph is a possible tag.

Given a peak of mass M for a spectrum of parent mass P, we produce a node at mass $M - |H|$ (for b fragments) and at $P - M$ (for y fragments). Two additional nodes are placed at zero mass and at the parent residue mass.

Scoring parameters are based upon odds empirically derived from an annotated set of spectra. The PRM range of a spectrum is divided into three equal sectors, and tagging parameters are derived for each sector and parent ion charge individually. A PRM nodes score, $S(N)$, is derived from the following three parameters, scored as log-odds ratios:

Tag generation (2)

Intensity rank: Peaks are ranked from the most intense to the least. Nodes receive a score based on the odds that a peak of a given rank is a *b* or *y* ion. These odds differ significantly by sector and charge. For instance, the odds that a rank-3 peak from the first or third sector of a charge-2 spectrum is a *b* ion are 69% and 14%, respectively. We take the log ratio of these odds with the odds that a randomly chosen peak matches a *b* or *y* ion, and obtain score $S_1(N)$.

Tag generation (3)

PRMsupport: For each PRM, we enumerate the ion types that support the node by checking for peaks at expected masses. The set of ion types identified is a witness set. Each node receives a score based upon empirical probability that its witness set represents a true peak. The optimal (and largest) witness set includes b, y, $b - H_2O$, $y - H_2O$, $b - NH_3$, $y - NH_3$, a, $a - H_2O$, $a - NH_3$, $b2$ (doubly-charged b), and $y2$ (doubly-charged y).

Witness set odds are affected by charge and sector. For instance, $b2$ and $y2$ ions are more abundant in charge-3 spectra (21% odds of appearing, versus only 8% in charge-2 spectra). We compare witness set odds with the odds that a randomly chosen peak matches a b or y ion; the log odds ratio is $S_2(N)$.

Tag generation (4)

Isotope pattern: For each atomic mass up to 1750, we compute the expected relative intensity of the first heavy isotopic peak based upon isotope frequencies of atoms C, H, O, N, and S. Each spectral peak is classified as a primary isotopic peak (with "child" peak at $+1Da$), secondary isotopic peak (with "parent" peak at $-1Da$), or lone peak. Using secondary peaks in a path is undesirable, particularly since there are pairs of amino acids whose masses differ by roughly $1Da$. For example, a peak classified as a primary isotopic peak is over five times more likely to be a y ion than is a peak classified as a secondary isotopic peak. The log odds ratio for this feature gives us score $S_3(N)$.

Tag generation (5)

We connect two nodes of mass m_1 and m_2 with an edge if the distance $(m_2 - m_1)$ is within ϵ (by default,$\epsilon = 0.5$) of a legal jump. Legal jumps include all amino acid masses, and (if PTMs are allowed) all singly modified amino acids.

We assign a score, $S(E)$, to each such edge based upon its skew (the difference between the edge length and the expected mass). In addition, edges containing PTMs receive a penalty derived from the expected frequency of that PTM. Tags are scored additively. Given a path connecting nodes $N_0, \ldots . N_n$ via edges $E_0, \ldots, E_n - 1$, with node scores $S(N_k)$ and edge scores $S(E_k)$, assign the tag a score of $\sum_{k=1}^{n}(S_1(N_k) + S_2(N_k) + S_3(N_k)) + \sum_{k=1}^{n-1} S(E_k)$.

Tag generation (6)

The three score factors S_1, S_2 and S_3 are listed in order of significance; they have Fisher criterion scores (FCS) of 1.63, 1.03, and 0.37 respectively.

Currently, InsPecT generates tags of length 3, and retains up to $50-100$ top-scoring tags for subsequent database searching. While this is effective, the authors have found that the top $10-25$ tags already show high sensitivity.

Database search and candidate generation

InsePecT searches the database for sequence tags using a trie-based data structure. After the construction of the trie automaton, all matches to any tag in the trie can be computed in a single scan of the database, independent of the number of tags in the automaton.

With the automaton, we must scan the entire database at least once. However, the cost is easily amortized by combining tags from multiple spectra into a single automaton. Each node of our trie contains zero or more tags; leaf nodes all contain at least one tag.

Database search and candidate generation (2)

InsPecT scans the database with the trie, looking for any peptide that matches a tag. When such a peptide is found (an *initial match*, or *tag hit*), we attempt to extend it to a full match by finding flanking sequences which match the tag's prefix and suffix masses, allowing for potential modifications. Different combinations of allowed modifications generate many different molecular masses

Database search and candidate generation (3)

Define a *decoration* to be the mass of a possible set of post-translational modifications. For instance, if our search allows up to two phosphorylations and two methylations, we generate a total of 9 decorations (including the "empty decoration"), $\{0, 14(1\ \text{methylation}, 0\ \text{phosphorylation}), 28, ...\}$).

We order the decorations according to increasing mass, and generate an array *DM* to store the mass values. This is done once, at the beginning of the search. Consider a tag with suffix value *S*. The tag matches a peptide in the database with decoration *d* if the peptide contains the tag, and has a suffix of mass *R*, such that $|S - (R + DM[d])| < \epsilon$.

The following algorithm takes time linear in the size of the array DM.

Database search and candidate generation (4)

Extend_tag_hit(S, N)

$R = 0$; $p = n$; $d = MaxMod$; //max. index of DM

$DONE = false$;

while (!$DONE$)

  do

  while $(R > S - DM[d] + \epsilon)$

    do $d = d - 1$;

    if $d < 0$ **then** *exit*; **fi**

  od

  if $(\mid S - R - DM[d] \mid < \epsilon)$ **then** *match*; **fi**

  $p = p + 1$;

  $R = R + M[p]$;

  if $(d < 0)$ **then** $DONE = true$; **fi**

od

Database search and candidate generation (5)

Prefix extension is performed similarly. Successful extensions are filtered to remove unfeasible decorations (for instance, oxidation requires the presence of a methionine).

Next, given a candidate peptide, the optimal attachment positions of PTMs on the flanking regions are determined by dynamic programming. This technique allows us to determine the optimal attachment positions without explicit enumeration and scoring of all possibilities. We consider a set of attachment sites to be optimal if the resulting set of PRM nodes has the highest possible score, as assigned during tag generation.

Database search and candidate generation (6)

We select attachment sites by computing $S[d, j]$, which is the optimal score attainable by attaching decoration d to the first j residues $r_1, \ldots, r_j$ of the candidate peptide. For an amino acid r, let $D(r)$ represent the set of all possible decorations that can be applied to that residue. Let $M[d, j]$ be the PRM created by attaching decoration d to the first j residues. Then, $S[d, j]$ is computed by the following recurrence

$$S[d, j] = \mathit{NodeScore}(M[d, j]) + \max d' \leq d, d' \in D(r_j)\{S[d - d', j - 1]\}$$

The corresponding dynamic program is used to compute the optimal attachment points. $\mathit{NodeScore}(M[d, j])$ equals the score $S(N_k)$ of the PRM node at that mass, or a penalty if no PRM node was generated nearby. Multiple PTMs at one amino acid - such as double oxidation of a methionine residue - are permitted where chemically reasonable. Note that $D(r)$ remains a small set. The peptide, with appropriately attached modifications, forms a candidate peptide that must now be scored.

Scoring

Finally InsPecT computes a probabilistic score similar to SCOPE which we will not give in detail.