Detailed description of the pIon code

pIon Workflow for PDM Candidate Identification

Input: Spectra, Database, Diagnostic_ion (default 126.12773), Threshold (default 0.7)
Output: List of filtered PDM candidates

Algorithm 1 pIon Workflow for PDM Candidate Identification

Input: Spectra, Database, Diagnostic_Ion (default 126.12773), Threshold (default 0.7)
Output: List of filtered PDM candidates

PSMs ← `Blind_Database_Search(Spectra, Database)`
`Mod_Type, Mod_PSMS, Unmod_PSMS` ←`Clustering_calibration(PSMs)`
`Mod_Type, Mod_PSMS` ← `Modification_location_evaluation(Mod_Type, Mod_PSMS)`
Unmodified_Vector←`Get_Abundance_Vector(Unmod_PSMS, Diagnostic_Ion)`
Scores ← `[]`
for each Mod_Type, Mod_PSMS **do**
    Mod_Vector←`Get_Abundance_Vector(Mod_PSMS, Diagnostic_ion)`
    Cosine_Similarity←`Get_Cosine_Similarity(Unmodified_Vector, Mod_Vector)`
    Score ← `Score_Function(Cosine_Similarity, Mod_PSMS)`
    `Scores.append((Mod_Type, Score))`
end for
Sorted_Scores ← `Sort(Scores, by=Score, descending=True)`
return `Filter(Sorted_Scores, Threshold)`

The following provides a detailed explanation of the steps in Algorithm 1, which outlines the pIon workflow for Probe-Derived Modification (PDM) candidate identification from mass spectrometry data.

1. Blind Database Search (Step 1)

The first step in the process is to perform a blind database search using the experimental mass spectrometry spectra (`Spectra`) and a reference `Database` of known peptide sequences. This step utilizes the `Blind_Database_Search` function, which identifies peptide-spectrum matches (PSMs). These PSMs are potential peptide candidates derived from the experimental data. The result is a list of PSMs that will be used in subsequent steps to evaluate possible modifications.

2. Modification Clustering, Calibration (Step 2)

The `clustering_calibration` function takes the PSMs identified in Step 1 and clusters them based on potential modification types. This step also involves the calibration and refinement of modification candidates, adjusting for experimental errors and ensuring that the modifications are consistent with the data. The function outputs: - `Mod_Type`: The unknown candidate modifications with accurate mass. - `Mod_PSMS`: The PSMs associated with each modification type. - `Unmod_PSMS`: The PSMs corresponding to unmodified peptides, used as a reference in the subsequent analysis.

3. Modification Location Evaluation (Step 3)

The `Modification_location_evaluation` function evaluates the precise location of the modifications within the peptides. It refines the identification of which amino acids are modified, which is important for accurate interpretation. This step updates the modification types and their associated PSMs, refining the data used in subsequent steps. pSite and hypothesis testing methods have been used. Details can be found in the paper [1].

4. Compute Abundance Vector for Unmodified Spectra (Step 4)

The `Get_Abundance_Vector` function computes an abundance vector for the unmodified PSMs (`Unmod_PSMS`). This vector represents the relative abundance (or intensity) of the diagnostic ion (typically defined by its mass-to-charge ratio, `Diagnostic_Ion`). This vector serves as a baseline for comparing modified spectra in later steps.

5. Scoring Modified PSMs (Steps 5 to 10)

In this phase, the algorithm computes a similarity score for each modification type by comparing the abundance vector of the unmodified PSMs with that of the modified PSMs:

1. For each modification type (`Mod_Type`) and the associated modified PSMs (`Mod_PSMS`), the algorithm computes a new abundance vector (`Mod_Vector`) using the `Get_Abundance_Vector` function.

2. The cosine similarity (`Cosine_Similarity`) between the unmodified and modified abundance vectors is calculated using the `Get_Cosine_Similarity` function. This similarity measure quantifies how closely the intensity distributions of the diagnostic ion in the unmodified and modified spectra align.

3. The score for each modification is calculated using the `Score_Function`, which combines the cosine similarity with other factors (e.g., the number of PSMs) to produce a final score. The scores are stored in a list (`Scores`).

6. Sorting and Filtering Candidates (Steps 11 and 12)

Once all modification types have been scored, the `Scores` list is sorted in descending order based on the score values. This ensures that the highest-scoring (most likely) modification candidates appear at the top of the list. The algorithm then filters the sorted list based on a predefined threshold score (`Threshold`).

7. Final Output

The final output is a list of filtered PDM candidates that meet the required threshold for further analysis or validation.

References

[1] Ji-Xiang He, Zheng-Cong Fei, Ling Fu, Cai-Ping Tian, Fu-Chu He, Hao Chi, and Jing Yang. A modification-centric assessment tool for the performance of chemoproteomic probes. *Nature Chemical Biology*, 18:904–912, 2022.