Julia Yuan

Anna Ritz

Bio 131

14 May 2020

Cyclopeptide Sequencing

The biological question my project seeks to address is how to sequence cyclic proteins not coded in DNA. According to the central dogma, all proteins are originally found in DNA, where the DNA must first be transcribed into RNA and then translated into proteins by ribosomes. However, there exist proteins that do not adhere to the central dogma, known as non-ribosomal peptides (NRPs), which are synthesized by NRP synthetase. Many of these NRPs are cyclic, presenting an even greater challenge when sequencing. As a result, one way to sequence NRPs is to fragment them and obtain the masses of those fragments with a mass spectrometer. With the masses of all possible peptide fragments, and known amino acid masses, it is theoretically possible to piece together the amino acid sequence of the original protein.

The very first computational step was to generate the theoretical spectrum of a linear peptide. Given a linear peptide, we must be able to computationally produce the theoretical experimentally produced mass spectroscopy spectrum of all possible fragment weights. The algorithm for this step is shown below:

```
LinearSpectrum(Peptide, AminoAcid, AminoAcidMass)
    PrefixMass(0) ← 0
    for i ← 1 to |Peptide|
        for j ← 1 to 20
            if AminoAcid(j) =  i-th amino acid in Peptide
                PrefixMass(i) ← PrefixMass(i − 1) + AminoAcidMass(j)
    LinearSpectrum ← a list consisting of the single integer 0
    for i ← 0 to |Peptide| − 1
        for j ← i + 1 to |Peptide|
            add PrefixMass(j) − PrefixMass(i) to LinearSpectrum
    return sorted list LinearSpectrum
```

Though I converted the two lists of amino acids and their corresponding masses to a dictionary, the general method I used was the same. Peptide masses are increasingly added to the *PrefixMass* list, so that the final value is the mass of the whole peptide, and then all values are subtracted to get all possible fragment weights. The next step was to modify this code for cyclic peptides, which was done by subtracting all smaller sections from the total mass of the peptide.

The final step was to actually generate a cyclopeptide with a theoretical spectrum matching an ideal spectrum. Using a branch and bound algorithm, I was able to generate every proposed amino acid string with a spectrum that matched a given spectrum. The pseudocode is as follows:

```
CyclopeptideSequencing(Spectrum)
  CandidatePeptides ← a set containing only the empty peptide
  FinalPeptides ← empty list of strings
  while CandidatePeptides is nonempty
    CandidatePeptides ← Expand(CandidatePeptides)
    for each peptide Peptide in CandidatePeptides
      if Mass(Peptide) = ParentMass(Spectrum)
        if Cyclospectrum(Peptide) = Spectrum and Peptide is not in FinalPeptides
          append Peptide to FinalPeptides
        remove Peptide from CandidatePeptides
      else if Peptide is not consistent with Spectrum
        remove Peptide from CandidatePeptides
  return FinalPeptides
```

The "branch" component of the branch and bound algorithm is seen in the *Expand* function, which expands *CandidatePeptides* to include every single peptide of length 1. At every additional step, *Expand* will create a number of new peptides equal to the number of amino acids by adding every possible amino acid to the end of each current peptide. These peptides are all individually confirmed to fit the given spectrum or not, and either accepted or discarded—the "bound" component.

Though my results were not run on real data, all of my tests ran with the exception of the cyclopeptide sequencing. Unfortunately, I was not able to get it to function on larger datasets, though it worked well with the examples. In the interest of decreasing runtime, I modified the expand function to only add amino acids found in the given spectrum, rather than all possible amino acids, but further improvements can likely be made. Some additional functions were written to make my outputs compatible with Rosalind.

Ok with report being posted online.