Study guide for Exam 1

This study guide is intended to help you to review for exams. This is not an exhaustive list of the topics covered in the class and there is no guarantee that these questions are representative of the questions on the exam. You should also review your class notes, the notes and readings on the syllabus, and your homework assignments.

Pairwise sequence alignment

- Terminology: Alphabet, sequence, string, subsequence, substring.
- Dynamic programming algorithms for *local, global* and *semiglobal* alignment.
  - Be familiar with the basic components of these algorithms: initialization, recursion, optimal score, traceback. What is the computational complexity of alignment with dynamic programing?
  - How do the basic algorithmic components differ for *local, global* and *semiglobal* alignment? What types of scoring functions are (un)suitable for each of these? Do any of the three types of alignment impose more restrictive criteria on the scoring function used? If so, what is the rationale for these criteria?
- Scoring functions
  - Edit distance. What are the required properties of distance functions for sequence alignment?
  - Similarity scoring. What are the required properties of simple similarity functions for sequence alignment?
  - You should be able to explain how changing a scoring function will influence the nature of optimal alignments obtained with respect to that scoring function.
- Applications: Given a particular sequence analysis scenario (e.g., sequence assembly, identifying introns, etc.), you should be able to state which type of alignment is most appropriate and why.

Multiple sequence alignment using classical approaches

- The formal definition of a multiple sequence alignment, which is a direct extension of the formal definition of a pairwise alignment.
- You should understand sum-of-pairs (SP) scoring, the most common approach to scoring columns in an MSA. SP scoring is easy to work with mathematically, but overestimates the number of mutations that gave rise to each site. Why?
- You should understand the relationship between a pair of sequences in an MSA, and a pairwise alignment of those sequences:

  - A multiple alignment induces pairwise alignments
  - A column in the induced pairwise alignment may contain all gaps, even though no column in the MSA contains all gaps. Why?
  - The pairwise alignment of two sequences induced by the optimal multiple alignment will not, in general, be the same as the optimal pairwise alignment of those sequences. Why? Further, the induced pairwise alignment may be more biologically realistic even though it is suboptimal.

- The dynamic programming algorithm for obtaining a global alignment of k sequences is a direct extension of the dynamic programming algorithm for obtaining a global alignment of 2 sequences. You should understand the initialization and recursion steps for the global multiple alignment algorithm and be able to write it down for 3 sequences.

- Global MSA is NP-complete. Given k sequences of length n, the computational complexity of the dynamic programming algorithm for global multiple alignment is $O(n^k 2^k k^2)$. You should understand how this expression is related to the steps in the multiple alignment algorithm. What part of the computation is associated with each term in the complexity?

- Because of its computational complexity, the exact alignment algorithm is not recommended for $n \gtrapprox 500$ or $k \gtrapprox 10$. For larger problem sizes, heuristics are used.

- Many heuristics approaches are based on the idea of a "progressive alignment".
  - The basic strategy of progressive alignment: First, pairwise alignments are constructed for all pairs of sequences. This yields a $k \times k$ matrix of pairwise distances, from which a "guide tree" is constructed. A multiple alignment is constructed by repeatedly merging sub-alignments. The order in which sequences/alignments are merged is determined by the guide tree. Frequently, but not always, the most similar sequence pairs are merged first.
  - Sub-alignments are merged using "profile alignment". A *profile* (i.e., an alignment) of k sequences drawn from alphabet Σ is treated as though it were a single sequence of symbols from a larger alphabet, $\hat{\Sigma}$. For example, when $k = 2$ where $\hat{\Sigma} = (\Sigma \times \Sigma) \setminus \{\substack{-\\-}\}$. Treating profiles as though they were sequences makes it possible to align profiles using the pairwise dynamic programming alignment algorithm. You should understand how profile alignment works.
  - Progressive alignment follows the "once a gap, always a gap" rule. Once an alignment of a subset of the sequences is formed, it cannot be rearranged to obtain a better alignment with new sequences as they are merged into the alignment. In other words, if a bad decision is made in an early stage of the alignment process, it cannot be corrected later. As a result, progressive alignment is not guaranteed to give the optimal alignment. This policy is also the reason that progressive alignment has better time complexity than dynamic programming.

  - The order in which profiles are merged determines the final multiple alignment. If the merging order is changed, a different alignment may result.
  - The computational complexity of progressive alignment is $O(k^2n^2)$. You should understand how this expression is related to the steps in the progressive alignment method.

- The performance of MSA programs is typically evaluated using benchmarks based on curated or automated structural alignments and/or simulated sequences. Various benchmarks are designed to mimic properties of different types of data sets encountered in practice, especially those that are challenging to align:
  - Highly divergent sequences, e.g., $< 50\%$ or $< 30\%$ identity.
  - A set of closely related sequences combined with several outliers, or "orphan" sequences.
  - Related sequences that differ due to large N or C terminal extensions or large internal insertions or deletions.

Markov chains

- Definitions and terminology Markov chains
  - States, the state probability distribution at time t, the initial state probability distribution.
  - The transition probability matrix. What requirements must a matrix satisfy to be a valid transition probability matrix?
  - What is the Markov property?
  - What are absorbing states, reflecting states, periodic states.
- We discussed finite-state, discrete-time, time-homogeneous Markov chains. You should understand each of these terms.
- Higher-order Markov chains: Given a transition matrix for 1 time step, you should understand how to construct a transition matrix for n time steps.
- Stationary state distributions. What is the formal definition of a stationary distribution? How can you verify that a given distribution is the stationary distribution? What Markov chain properties are required for a Markov chain to have a unique stationary distribution? What properties prevent a Markov chain from having a stationary distribution?
- Colloquially speaking, what does it mean for a Markov chain to be irreducible? When is a Markov chain aperiodic? You should be able to inspect simple Markov chain models and determine whether or not they are periodic and/or irreducible.
- What is the time reversibility property and how do you test whether a Markov chain is time reversible? Why is this property important for working with sequence evolution models?
- Simple random walk models. What are they?