CM124 Programming Assignment Report

The Haplophaser2.m (Octave compatible code) and Haplophaser3.m (original MATLAB code) functions are renditions of the expectation-maximization algorithm, with some tweaks to improve speed in the face of large amounts of SNPs to be phased. They take in input file, output file, and partition size as inputs.

The first tweak to improve performance comes at the start of the code, using the user-input partition size. The code performs the expectation-maximization algorithm in partitions and ligates the phased partitions together to obtain the final phased population. The smaller the partition size is, the faster the algorithm will run, but this comes at the cost of accuracy. For test_data_1.txt, a partition size of 14 was used. For test_data_2.txt, a partition size of 13 was used.

For each partition, the code first determines the potential haplotype pairs for each of the 50 individuals. Initial frequencies are also assigned for each of these haplotype pairs with a uniform distribution. All unique haplotypes are stored as well. The initial potential haplotype pair frequencies for each genotype are then used to calculate unique haplotype frequencies. The code then enters a while loop in which potential haplotype pair frequencies are calculated for each genotype again, and then calculation of unique haplotype frequencies is repeated. This while loop has two conditions, each representative of a tweak included in the code to improve runtime.

The first condition is simply that the loop will stop when the maximum difference between old and new unique haplotype frequencies is less than 0.1%. This threshold can be decided within the code, with smaller thresholds improving accuracy at the cost of speed.

The second condition involves blacklisting individuals who have converged to a single haplotype pair. No more calculations need to be done for these individuals. This condition is rarely met, but the existence of a blacklist speed up the runtime of the code. Furthermore, a threshold is placed on the potential haplotype pair frequencies; once passed, this threshold results in that haplotype pair being assumed to be the correct one. A lower threshold is also placed so that any potential haplotype pair with a frequency of under 1% is assumed to no longer be in consideration. These leaps speed up the algorithm, but again at the cost of accuracy.

Once the while loop concludes, the phased partition is constructed from the stored indices of most likely haplotype pair for each individual and added on to the phased population, and the process is repeated for the next partition.