CSCI2951 HW3 | Ben Siranosian | 2014-11-12

Problem 1: Units of measuring LD

I have implemented algorithms to compute the solution to the dominating set and set cover problems. For the dominating set problem, I solve it through a brute force search of the subsets of the vertices of the graph. I use an adjacency matrix as the internal representation and search for a subset of vertices where the sum over every row is at least one, representing the dominating set in the graph. I solve the set cover problem through a conversion of the sets to an adjacency matrix and a reduction to the dominating set problem.

I have also implemented an algorithm to parse genotype data downloaded from HapMap and calculate pairwise r^2 values between SNPs. This algorithm discards any SNPs with ambiguous genotypes (at least one *N* present in the genotype of the individuals) and SNPs where the entire population of individuals is identical (all homozygous for one allele). To calculate r^2 when only the genotypes of individuals are known, I use the approximation found in (1) that *r* is approximated by the correlation between two genotypes:

$$r \approx C(Y,Z)/\sqrt{V_y V_z}$$

Where Y and Z are the vectors of genotypes for individuals at two SNPs, $C(Y,Z)$is the covariance and V_y is the variance. This approximation is accurate when populations are mating randomly, and close to accurate even when that assumption is violated. For some exploratory analysis, I downloaded the genotype calls for chromosome 1 from the HapMap project. The results of examining the first 10,000 SNPs can be found in figure 1. Out of 10,000 SNPs investigated, 612 did not have missing data and had at least one individual that was not homozygous for the major allele.

Converting real r^2 data to a useful measure of informativeness or a construct appropriate for LD select proved difficult. The majority of calculated r^2 values for real data are low and sparse, making it difficult to apply my algorithms for dominating set or set cover. Figure 2 shows the adjacency matrix for LD-select at different values of r^2. The matrix is very sparse for higher values of r^2.

Next, I studied the relationship between the threshold r^2 value for LD-select and informativeness. I defined the information threshold as the fraction of SNPs selected in a dominating set, computed over 100 replicates at randomized subsets of size 13 (I had to keep this number small because otherwise computation of the dominating set took too long) from the HapMap data processed earlier. The results from these calculations are in figure 3. I found that informativeness increased rapidly with increasing r^2 at first, until a value of r^2 = 0.1. After this, informativeness increased in a logarithmic fashion with increasing r^2, attaining a maximum value shortly before r^2 = 1.0. A second interesting quantity to plot was the proportion of calculated dominating sets that were of maximum size. These are sets where no reduction was possible – a smaller set of SNPs that captured the same information was impossible to find. This quantity seems to model a logistic function with increasing r^2. Code to produce the figures and all of the algorithms I developed can be found in 'dominating_set_cover.py'.

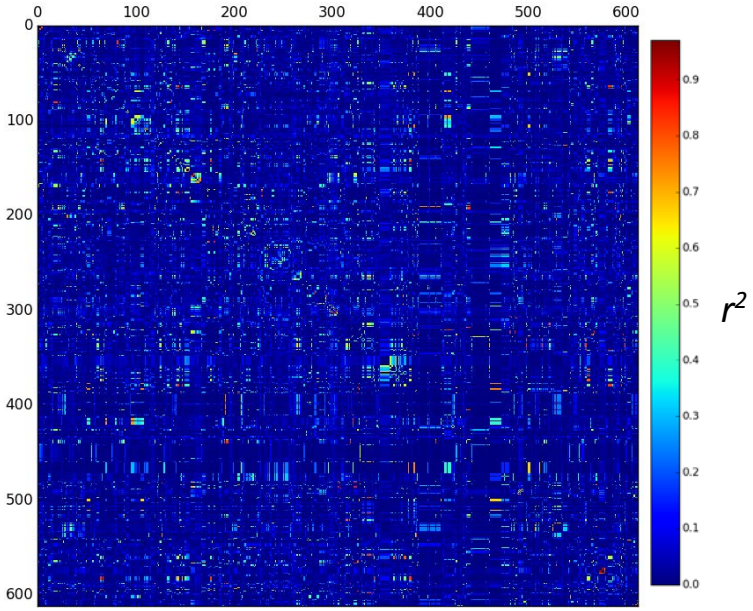

Figure 1: Pairwise r^2 values for the first 612 informative SNPs in chromosome 1.

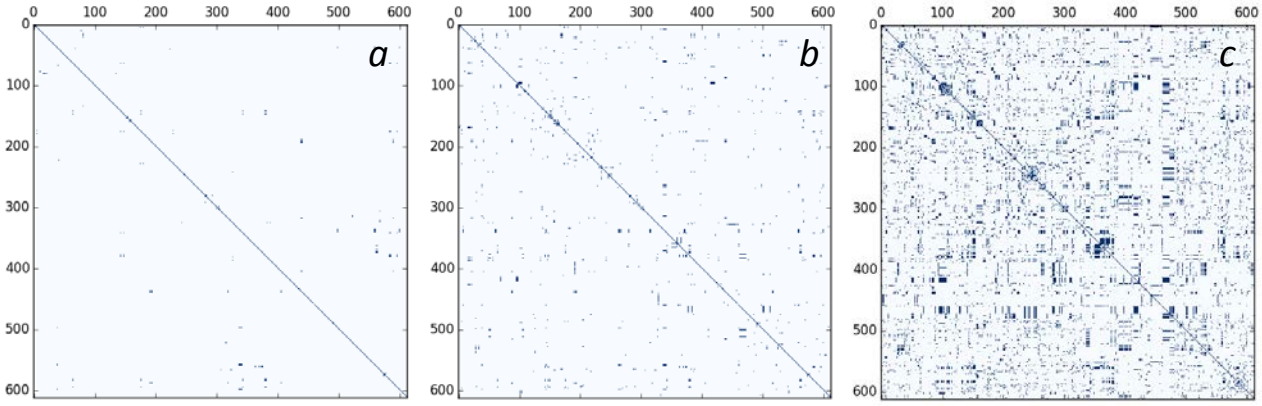

Figure 2: LD-select adjacency matrices for $r^2 = 0.75$ (a), $r^2 = 0.50$ (b), $r^2 = 0.15$ (c). Note the sparse nature of the matrix, even at small r^2 values.

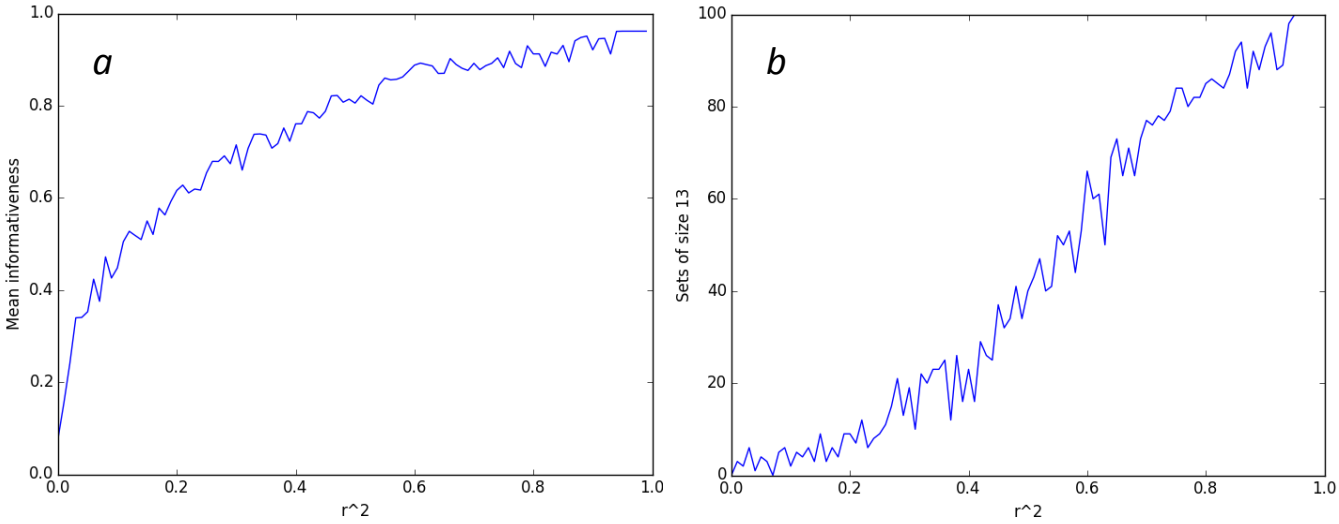

Figure 3: a) Relationship between informativeness and r^2 is monotonically increasing. Models a logarithmic function after $r^2 = 0.1$. b) Number of subsets where a dominating set cannot be found models a logistic function with increasing r^2.

Problem 2: GWAS presentation

I have chosen to do my GWAS presentation on atrial fibrillation. I have gathered information on the GWAS study for this disease presented in a publication (2). My outline interpretation of their analysis is as follows:

1. Experimental Design – Illumina Hap300 BeadChip on Icelandic population
    a. AF or atrial flutter (AFI)
    b. 550 patients, 4476 controls
    c. 316515 SNPs satisfying quality criteria
        i. Picked from phase 1 of HapMap
        ii. Some generated no data or yield <90%
        iii. Some monomorphic or nearly
        iv. Some showed distortion from HWE in the controls
        v. Some had genotyping problems
        vi. Samples with a call rate below 98% discarded
    d. Genotype calling verification step based on HapMap data
2. Associated SNPS
    a. 3 SNPs at significance threshold (Bonferoni correction)
        i. Association testing procedure
            1. Likelihood ratio statistic
                a. X^2 distribution under null if subjects unrelated
            2. Odds ratio calculating assuming multiplicative model
                a. Two chromosomes for each individual
            3. Multiple groups combined with a Mantel-Haenszel model
                a. Groups can have different allelic population frequencies, genotypes and haplotypes, same relative common risks
        ii. All within a single LD block, chr4q25
        iii. rs2200733 (odds ratio (OR) = 1.75; P < 1.6 E-10), rs2220427 (OR = 1.75; P < 1.9 E -10) and rs2634073 (OR = 1.60; P < 2.1 E-9
            1. First two SNPs perfect proxies for each other in HapMap, almost perfect in Icelandic (r^2 = 0.999)
        iv. adjusted for relatedness of individuals

Problem 3: Hora and Tempus

To answer this question, I took to the process of simulation. I simplified the problem some by assuming Hora takes 100 steps to make a watch, while Tempus takes 125. This gives Hora an advantage at lower call volumes because his process doesn't require sub-assemblies! If Hora receives a call while assembling a watch, his progress is reset to 0. If Tempus receives a call, though, his progress is only subtracted by 10. I can then compare how many watches the two make in a given set of time as a function of the call frequency.

Simulating from call frequency = 0.0001 to 0.01 we see that Hora has an initial advantage, but Tempus takes over as calls become more frequent. Hora makes the most watches when business is slow, but can't keep up once the phone starts ringing off the hook. I probably could have solved this problem with some math (look at that linear relationship) but simulation seemed like the most fun since I was already in a programming mood. Code for this analysis can be found in hora_tempus.py.

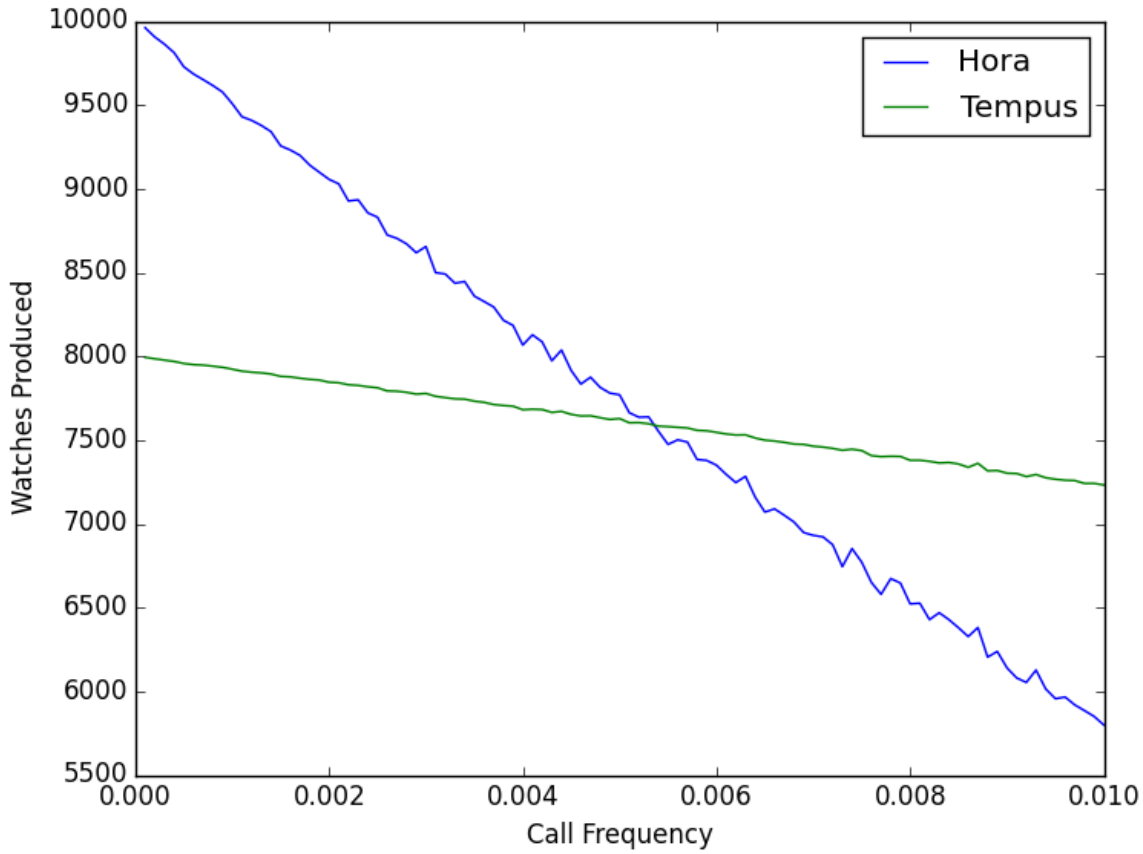

Figure 4: Hora and Tempus compete for the title of best watchmaker.

References

1. Rogers AR, Huff C: Linkage Disequilibrium Between Loci With Unknown Phase. Genetics 2009, 182:839–844.

2. Gudbjartsson DF, et al.: Variants conferring risk of atrial fibrillation on chromosome 4q25. Nature 2007, 448:353–357.