Advanced R Programming for Bioinformatics

The site for this course is

`http://faculty.washington.edu/kenrice/sisg-adv/`

… please make sure you can access this page. Bookmarking it may also be helpful.

Exercises for Session 1: Effective Coding, Timing

With data on many SNPs, a common task is to compute the 'r-squared' between each pair of SNPs. Coding the genotypes for each SNP as 0/1/2 copies of an allele, r-squared for a pair of SNPs can be computed as the square of the Pearson correlation between the genotypes. (In R, the Pearson correlation of two vectors is given by the `cor()` function.)

1. For the SNP data in the *moderately* large `amd` dataset, write code which loops over pairs of SNPs, and explicitly calculates r-squared for each pair of SNPs. (Hint: try your code on a small number of SNPs before scaling up.)

   If you eliminate redundant calculations, such as computing r-squared between a SNP and itself, by how much can you speed up this code?

2. Use the profiler commands to see which operations take most time.

3. By looking up the documentation for `cor()`, implement all the pairwise calculations with a single R command. How much faster is this approach, compared to the previous calculations?

4. Yet another way to work out all the r-squareds is to compute the cross-product of the scaled matrix of genotypes with itself, divided by *n-1*, i.e.

   ```
   crossprod(scale(genotypes))/(n-1)
   ```

   Verify that this works, for appropriately-defined objects `genotypes` and `n-1`. How does this approach's runtime compare with the Q3 approach?