SNP-wise Weights

Stuart Brabbs

The goal here is to explore how to assign weights to the models produced by SNP-wise selection. The natural way explored here involves weighting by a sort of "posterior score." Since the form of the posterior is $p(\theta \mid y) \propto p(\theta)p(y \mid \theta)$, we can compute the posterior score from this. While we could specify the prior based on any prior knowledge, we will focus on the simple situation of a diffuse prior $\theta \sim Bernoulli(\pi)$, where $\pi = 1/p$, the p being the number of predictors.

The harder part is calculating the likelihood $p(y \mid \theta)$. We first experiment with calculating instead a Bayes Factor based on a slightly different prior, though also diffuse. This prior, D2, comes from the paper "Imputation-Based Analysis of Association Studies: Candidate Regions and Quantitative Traits" by Servin and Stephens. To calculate the Bayes Factor for an individual SNP based on this prior, the following function is given, which requires as inputs predictor vector g, phenotype vector y, and sigmaa and sigmad, standard deviations that are only used as a dividing factor and are thus not as important for our purposes. As in the paper, we take these to be 0.5, though they advise averaging over several values of these to get more accurate results. It outputs the log Bayes Factor:

```
logBF = function(g,y,sigmaa,sigmad) {
  subset = complete.cases(y) & complete.cases(g)
  y=y[subset]
  g=g[subset]
  n=length(g)
  X = cbind(rep(1,n), g, g==1)
  invnu = diag(c(0,1/sigmaa^2,1/sigmad^2))
  invOmega = invnu + t(X) %*% X
  B = solve(invOmega, t(X) %*% cbind(y))
  invOmega0 = n
  return(-0.5*log10(det(invOmega)) + 0.5*log10(invOmega0) - log10(sigmaa) -   log10(sigm
ad) - (n/2)*(log10(t(y- X %*% B) %*% y) - log10(t(y) %*% y - n*mean(y)^2)))
}
```

To test this function, we first simulate genotypes for 100 SNPs for 100 individuals, and then assign each individual a value for a continuous phenotype, with 4 SNPs (x1, x20, x50, x95) being true causal ones:

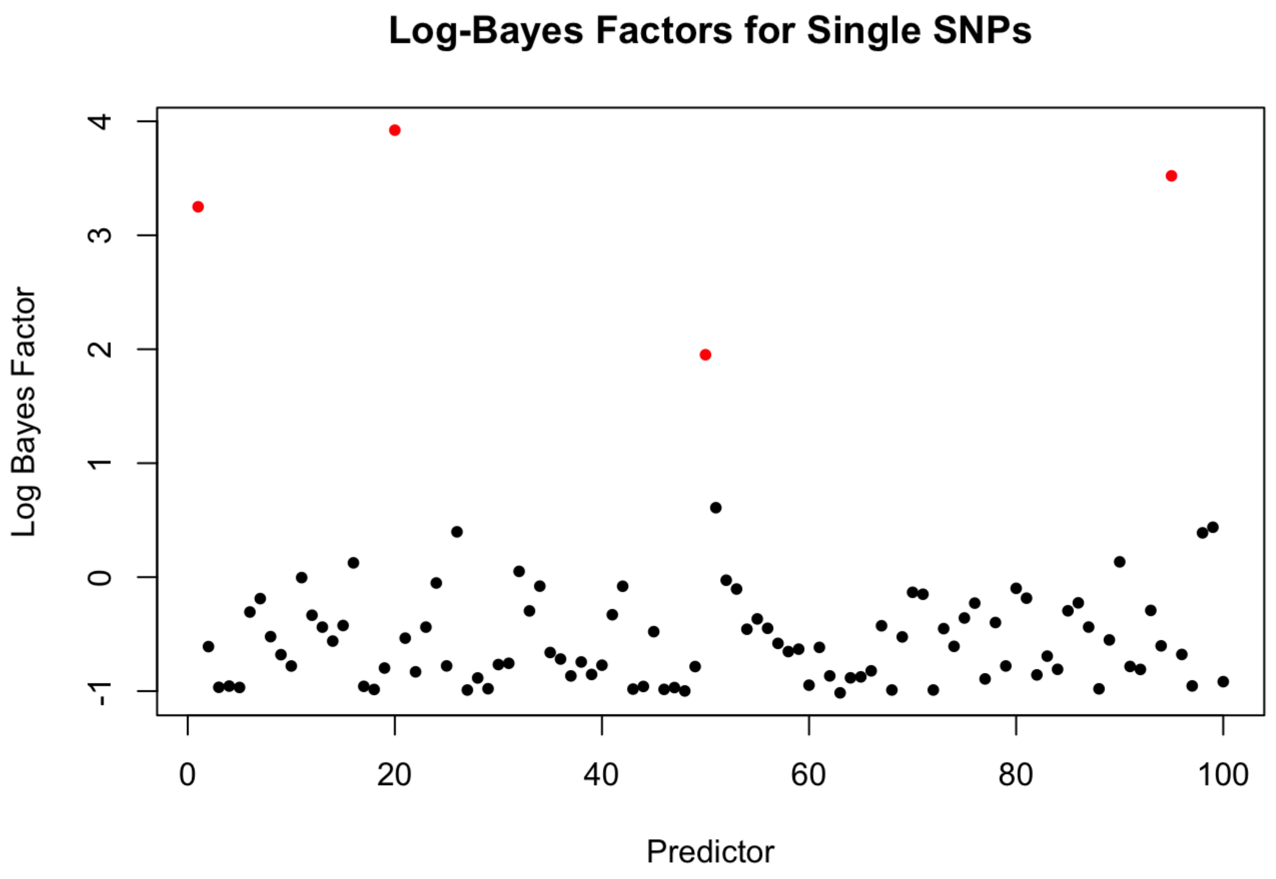
```
set.seed(1)
geno <- sample(0:2, 100, replace=TRUE)
for (i in c(2:100)) {
  g <- sample(0:2, 100, replace=TRUE)
  geno <- cbind(geno, g)
}

y <- geno[,1] + geno[,20] + geno[,50] + geno[,95] + rnorm(100)
```

We now calculate the Bayes Factor for each, and plot to ensure the highest values are for the causal SNPs:

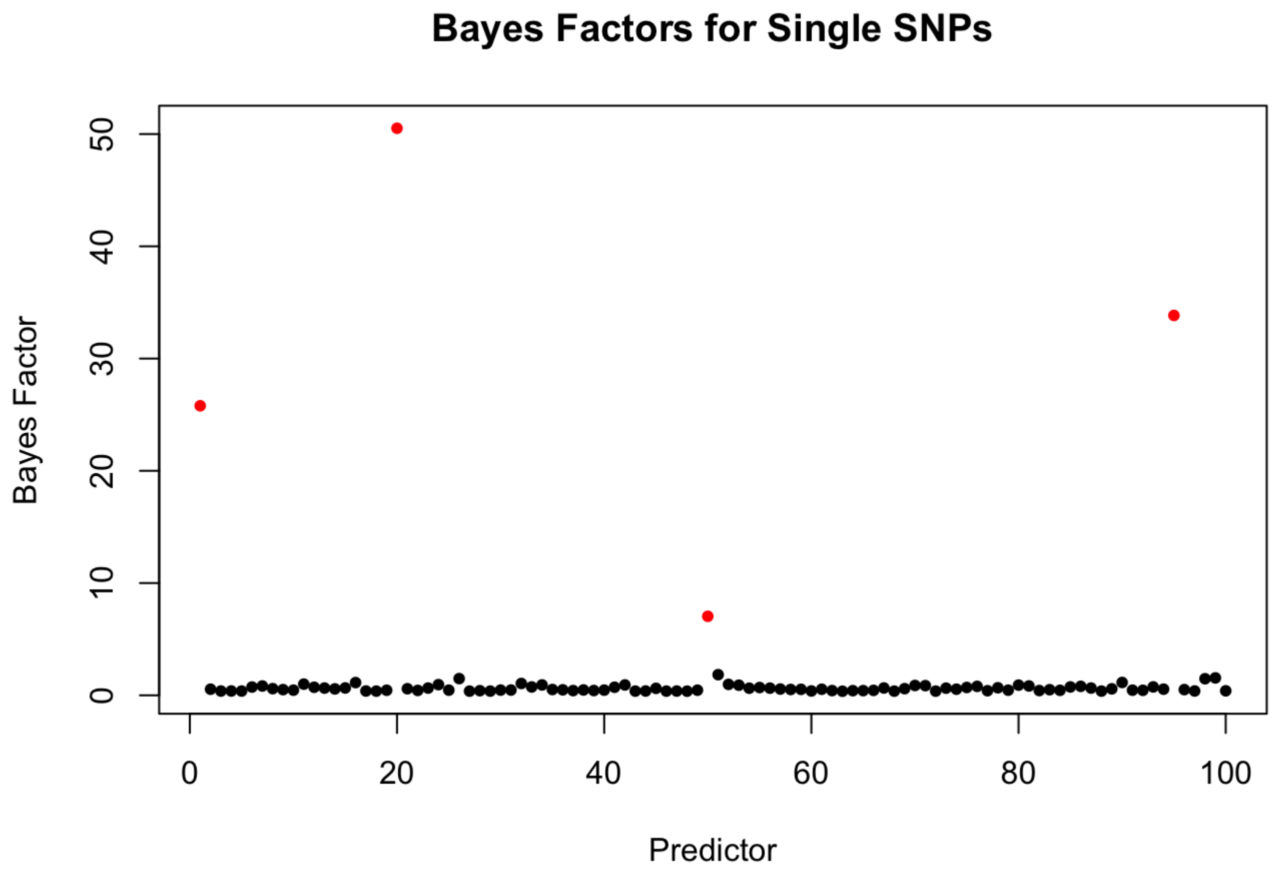
```r
bfs <- c()
for (i in c(1:100)) {
  newbf <- logBF(geno[,i], y, 0.5, 0.5)
  bfs <- c(bfs, newbf)
}

index <- c(1:100)
signals <- c(1, 20, 50, 95)
plot(bfs, pch = 20, xlab = "Predictor", ylab = "Log Bayes Factor", main = "Log-Bayes Fac
tors for Single SNPs", col = ifelse(index %in% signals, "red", "black"))
```



In the above plot we have the log Bayes Factors of the SNPs under single-SNP regression. We can see that the four causal SNPs (in red) are the ones with the highest Bayes Factors. We plot log Bayes Factors because it is easier to visualize the variation - a plot of the Bayes Factors themselves is below:

```r
plot(exp(bfs), pch = 20, xlab = "Predictor", ylab = "Bayes Factor", main = "Bayes Factor
s for Single SNPs", col = ifelse(index %in% signals, "red", "black"))
```



The next step is to identify the normalizing constant to use on the weights to get them to sum to one. We would, of course, use the Bayes Factors themselves (not log) as weights, because this is more natural and the log factors are often negative.