Question 1: Sample number of QTNs of your choice from the genetic markers used in homework2 and simulate QTN effects from a standard normal distribution. Assign genetic effects for each individual. Simulate normal distributed residual effects with appropriate variance to have a heritability of 0.75. Add residual effects to the genetic effects to create phenotypes. Use the GLM GWAS package you developed in homework 4 to perform association analyses with three PCs included as covariates. Count number of false positives for identifying half of your QTNs (20 points).

Hypothesis: The GWASxPCA_GLM package performs GWAS by GLM using PCA as covariates. Thus, I predict that this method will show improved performance at identifying the QTN compared to GLM without covariates.

Method: Phenotypes for each individual were simulated from genotypic data (n=281, m=3093) using the G2P function with heritability set to 0.75 and 10 QTNs included. Simulated phenotypes were used along with the genetic map to perform GWAS using the GWASXPCA_GLM package. Marker p values were extracted from the output file and the *sort* function was used to order them by ascending values before merging with the ordered SNP output. Value matching using *%in%* identified QTN with a TRUE value, all other SNPs received a FALSE value in a new column. TRUE (1) and FALSE (0) values were converted to binary data, and the *cumsums* function was used in a *for loop* to count each QTN until the middle QTN was identified. The –log(p value) for this QTN was used as the cutoff value on a Manhattan Plot. False positives were counted and reported.

Results: Ten QTN were used to perform phenotype simulation and GWAS. The fifth QTN occurred as the 13th most significant SNP, and 8 SNP (i.e. positives) were found to be more significant (Figure 1). The package was able to return three QTN in the top five most significant SNP, with 61.5% of SNP above the cutoff line occurring as false positives. The package is superior compared to GWAS by GLM thanks to the use of PCA as covariates, but there is much room for improvement.

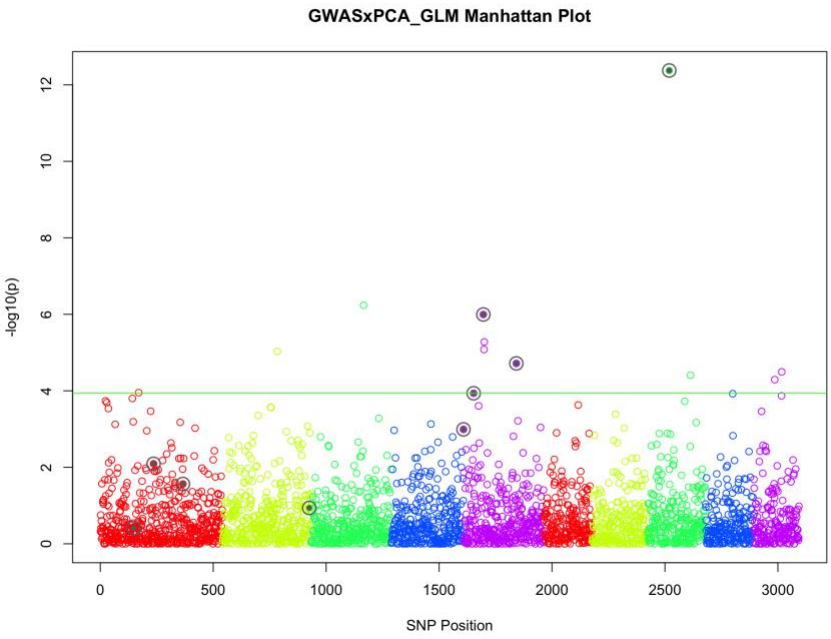

Figure 1: Manhattan Plot for GWASxPCA_GLM with 10 QTN illustrated as filled in shadow circles. Cutoff line is set to -log10(p) of the fifth QTN. Open circles above the cutoff line represent SNP occurring as false positives.

Question 2: Repeat simulation in (1) 30 times and exam statistical power vs. FDR at mapping resolution of 100,000 base pairs (20 points).

Hypothesis: I predict that power vs. FDR plots will be relatively uniform among the 30 replications of phenotype simulation and GWAS using GAPIT. At a resolution of 100,000 base pairs, I expect power and FDR to be relatively high. Based on what has been lectured on, I assume that as base pair resolution increase, the area under the curve will increase. Thus, a resolution of 100kb will be sufficient.

Methods: The *replicate* function was used to perform phenotype simulation and GWAS via GAPIT thirty times. An output matrix (i.e. power.rep) was created using the *GAPIT.FDR.TYPEI* function for each rep, which includes a list of seven features. The power values are conserved for each replicate. The *seq* function was used to return a series of numbers starting at the row number for the target result (e.g. FDR, AUC, and false positive) and advancing by 7 to represent the next column for the target result for all thirty reps. These values were then used to index power.rep to return the value for each target result value for each rep. We then use *reduce* to average across the reps to return a mean FDR and false positive value for each QTN, and a single AUC value for FDR.

Results: Performing phenotype simulation and GWAS via GAPIT thirty times resulted in a mean of 0.4 false positives occurring as more significant than the fifth QTN. Compared to the package GWASxPCA_GLM, GAPIT returns fewer false positives at the same cutoff (i.e. fifth QTN). A mean of 0.64 was found for the AUC, which is greater than the values presented in lecture for bp resolutions of 1, 1K, 10K and 100K. Mean FDR values ranged from 0.0067 to 0.97 (Figure 2a), and power vs. FDR curves show high variability for each replicate (Figure 2b). It's possible that the "zig-zag" nature of the curves is from an error in the calculation of power using the *GAPIT.FDR.TYPEI* function.

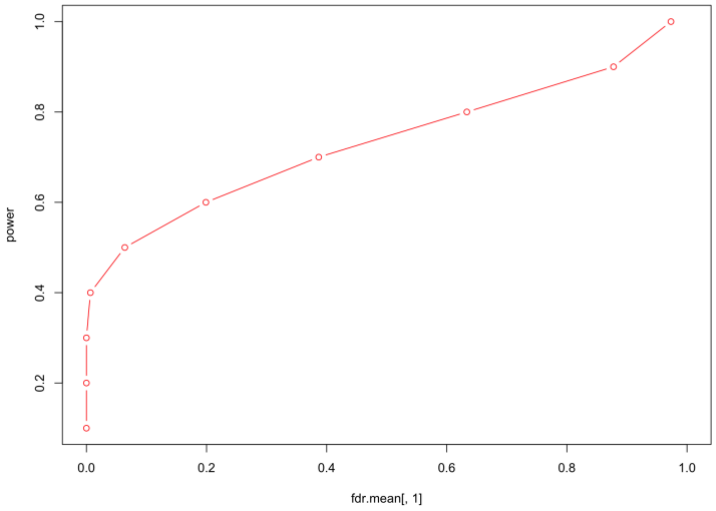

Figure 2a: Power vs mean FDR values calculated from 30 replications of phenotype simulation and GWAS by GAPIT.

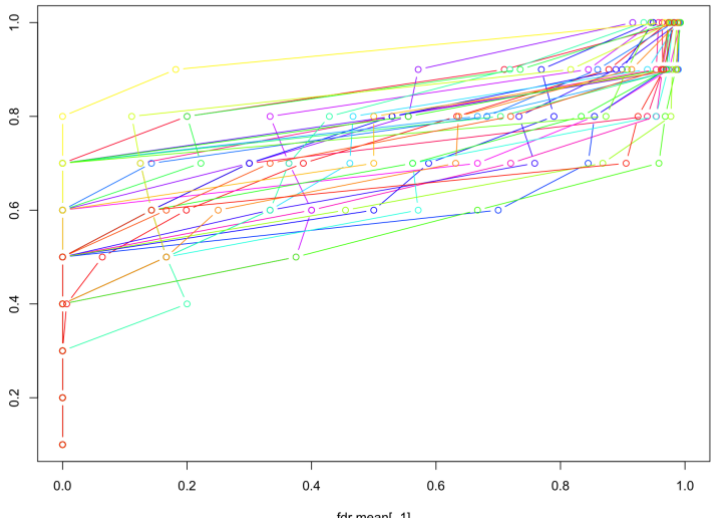

Figure 2b: Power vs FDR curves for each replicate of phenotype simulation and GWAS using GAPIT and mean FDR values.

Question 3: Repeat (2) with your package replaced by following packages:

a. PLINK: 20 points
b. FarmCPU: 20points
c. BLINK: 20 points

Hypothesis: I predict that each package will differ in performance due to inherent differences. FarmCPU will perform better compared to PLINK (i.e. greater power and FDR). FarmCPU uses QTN to build kinship, which is then used to calculate likelihood. FarmCPU also uses an iterative process to recalculate variance matrices to identify and account for the most significant SNP as pseudo QTN, thus controlling for the masking effects of these QTN and allowing for identification QTN of lesser effect. BLINK will perform the best, and will compute the fastest! In BLINK, LD is used to identify QTN by removing SNP in LD. FarmCPU optimizes bin size to increase power and FDR to identify QTN. Also, in BLINK kinship is replaced by QTN and this can cause overfitting and a penalty can be incurred based on the number of QTN.

Methods: Each package was run using a phenotype simulation with 10 QTN and a heritability value of 0.75. Power, FDR, and AUC were calculated in a method similar to question 2. For PLINK, quality control parameters were set at 0.2 for missing rate and 0.5 for MAF.

Results: *PLINK*

One iteration of PLINK performed using iPAT and genotype simulation with GAPIT produced FDR values close to 0.98 (Figure 3a). Due to quality control settings, 2,633 SNP were returned. After thirty replications, mean FDR values ranged from 0.0167 at QTN 1 to 0.976 at QTN 10, mean AUC.FDR was 0.44 (Figure 3a). I am weary of these results, since I had many issues running PLINK and producing poor results. For example, I received a warning (unknown GUI for my working directory) and an error (error in if (file ==””) file <- stdout() else if (is.character(file)) …) while using iPAT.

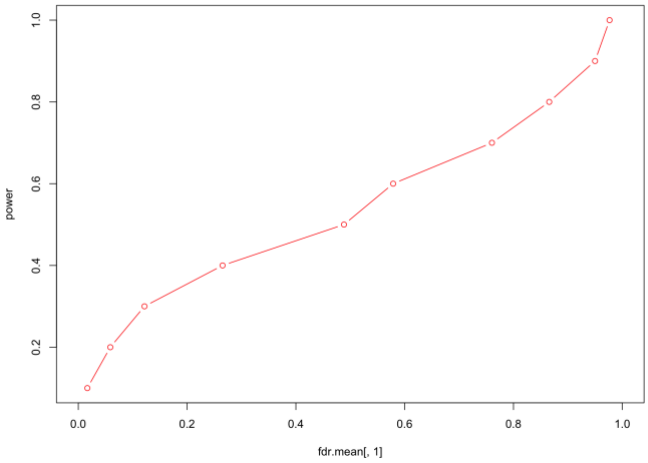

Figure 3a: Power vs. FDR for one GWAS using PLINK

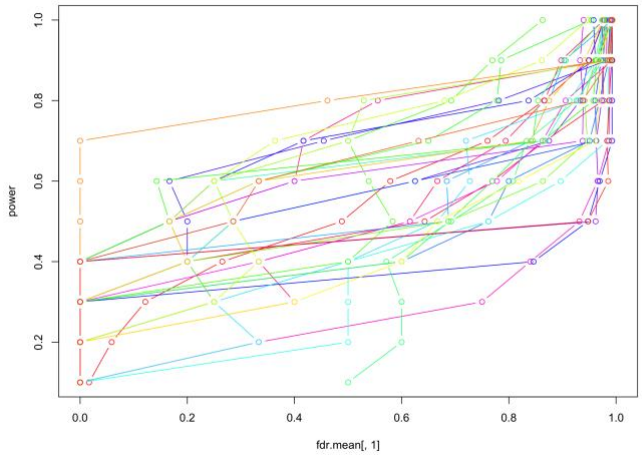

Figure 3b: Power vs. FDR for 30 replications of GWAS with PLINK

FarmCPU

After performing GWAS using FarmCPU one time, 8 QTN had FDR values of zero and two QTN had FDR values closer to 1.0 (Figure 4a). At QTN 9 and 10, 106 and 327 false positives were returned, respectively. FarmCPU computation time was noticeably longer than PLINK when performing thirty replications. GWAS by FarmCPU for 30 replications resulted in a mean AUC of 0.8275, and mean FDR values ranging from 0.0048 at QTN 6 to 0.605 at QTN 10 (Figure 4b). Compared to the package GWASxPCA_GLM, mean AUC for FarmCPU was greater than the package (0.64). Mean AUC for FarmCPU was
greater than PLINK, although given the poor
results I am hesitant to conclude that FarmCPU has greater performance.

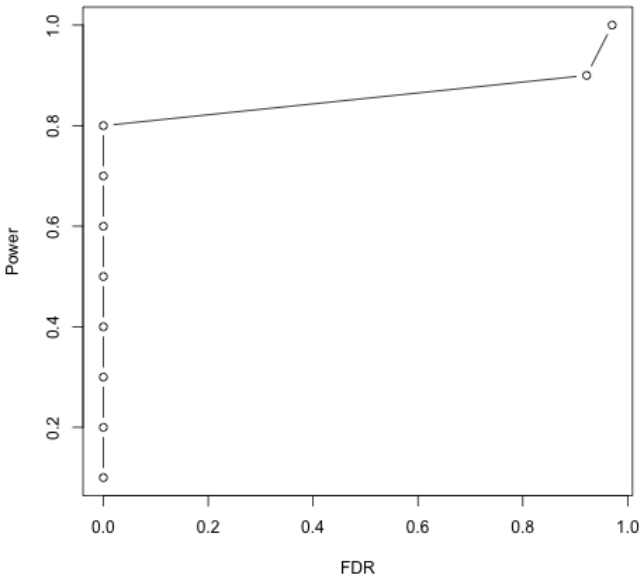

Figure 4a: Power vs. FDR curve for one iteration of GWAS using FarmCPU.

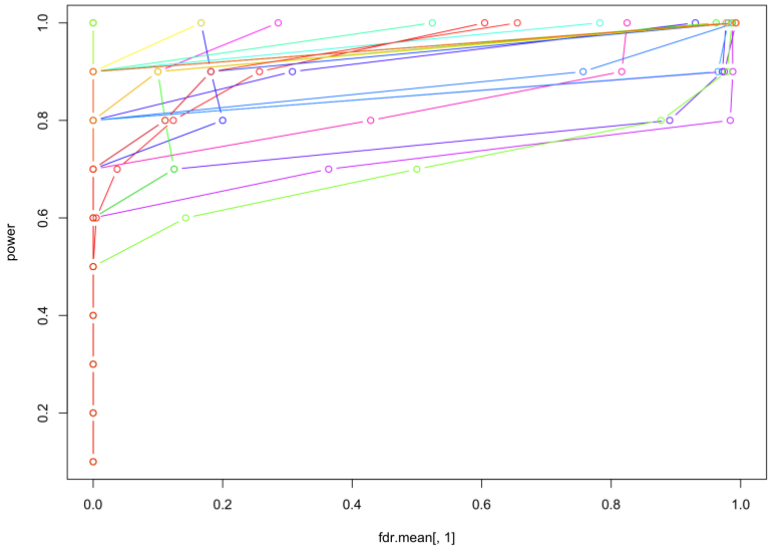

Figure 4b: Power vs. FDR curves for thirty replications of GWAS using FarmCPU.

BLINK

BLINK performed the best compared to GWASxPCA_GLM, GAPIT, and PLINK, and FARMCPU. Several linkes of evidence support this claim. Average AUC for BLINK after 30 replications was, and mean FRD values ranged from 0.0056 at the 5th QTN to 0.645 at the tenth QTN (Figure 4a&b). These values are slightly higher than FarmCPU; however, mean AUV for BLINK after 30 replications was 0.8465, compared to 0.8275 for FarmCPU after 30 replications. GWAS performed by BLINK was noticeably faster than the others methods.

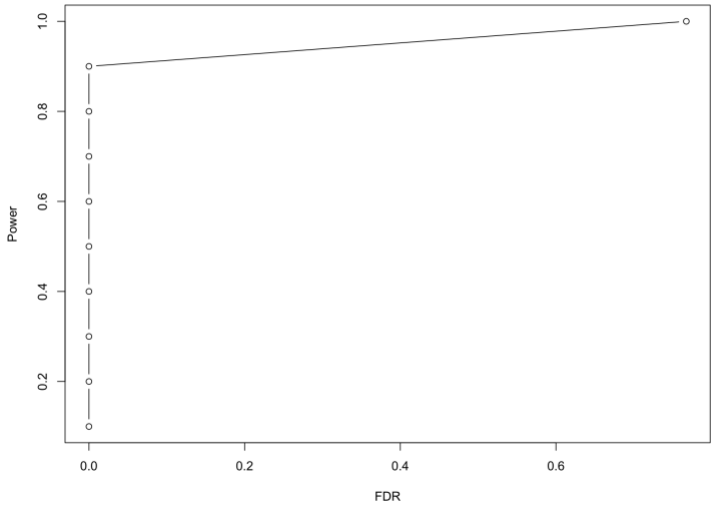

Figure 5a: Power vs FDR curve for one iteration of GWAS using BLINK

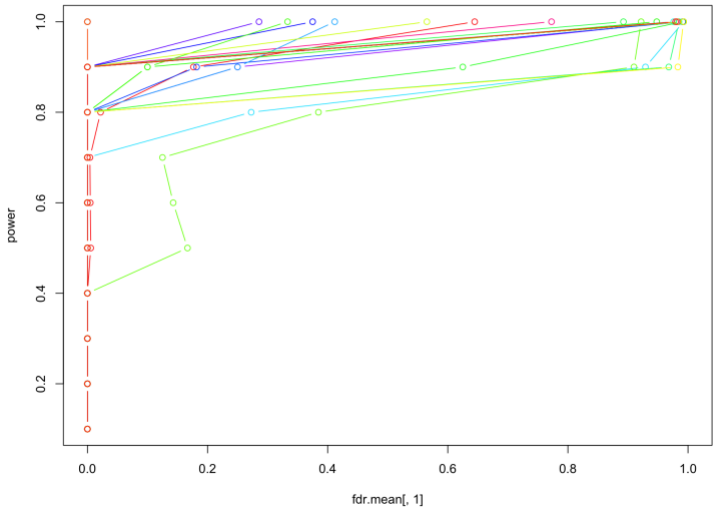

Figure 5b: Power vs FDR curves for thirty replications of GWAS using BLINK.

Summary

Due to the inconsistent results produced for PLINK, it is difficult to definitively to declare the "best" method for GWAS. Moreover, the "best" method should be judged by several factors, computing time, memory usage, and accuracy. We did not quantify average computing times for each method, nor did we quantify the average amount of memory used. Using the power and FDR curves produced for each method, and the mean value for AUC after thirty replications, we can make identify which methods are superior based on this metric. I would rank BLINK over FarmCPU, and both methods over GAPIT and the GWASxPCA_GLM package. GWAS by GLM, which both GAPIT and GWASxPCA_GLM use, appears to produce similar results regardless of the method. I would assume that PLINK ranks near BLINK and FarmCPU, but due uncertainty in performing analysis using this method, I cannot produce a definitive ranking. However, using mean AUC after 30 replications as a metric for ranking, I would rank:
BLINK (0.86) > **FarmCPU** (0.81) > **GAPIT** (0.64) > **PLINK** (0.48)

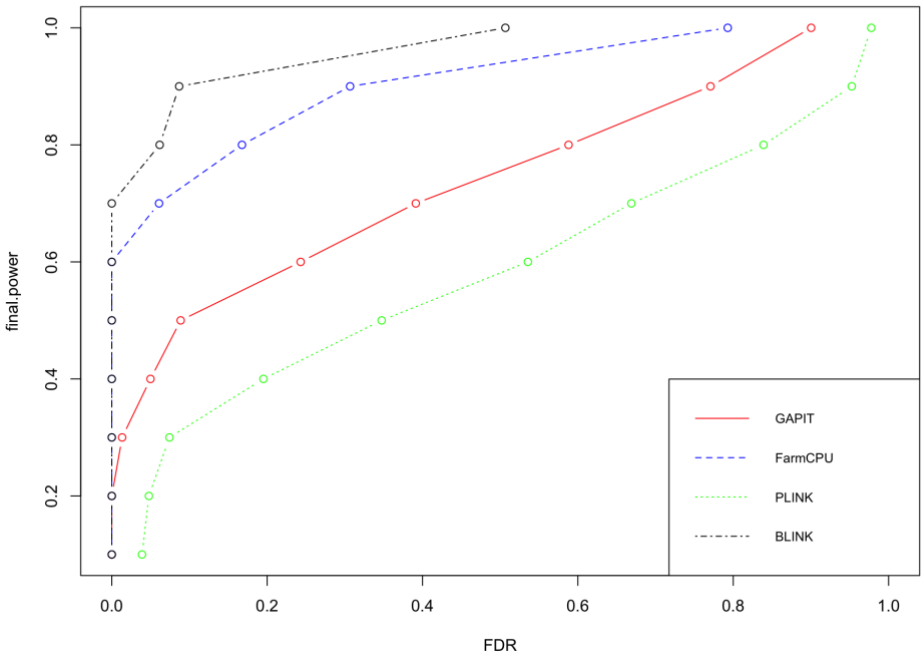

Figure 6: Power vs. curves summarizing each method investigated