GWAS practical

Andrea Manica

26 April 2018

A simple example of GWAS: genetic determinants of diabetes

We will use the library GenABEL to run GWAS (both simple, and more complex). Let's start by installing the package in your R environment. Type:

```
install.packages("GenABEL")
```

You will be asked to choose a mirror, select any that is close to you. After installation, we can load the library, so that we can use it (in the future, you won't need to reinstall the library, but you will need to load it every time you start R):

```
library("GenABEL")
```

For this tutorial, we will use data available with the library. They are simulated, keeping it simple in terms of ethical concerns for publishing data with phenotypes. We will immediately rename this dataset to something easy to remember (today, we will focus on diabetes, so let's use that as a name)

```
data(ge03d2ex)
ge03d2ex->diabetes
```

We can get a feel for the genetic data by typing:

```
head(summary(diabetes))
```

For each SNP, we have: chromosome, map position, allele coding, number of observed genotypes, allelic frequency, genotypic distribution, P-value of the exact test for HWE, Fmax (estimate of deviation from HWE, allowing meta-analysis) and LRT P-value for HWE test.

We can summarise the genetic data with

```
descriptives.marker(diabetes)
```

Our data look pretty good in terms of missing data (successful genotypes by person and per SNP). However, note that the cumulative distribution of SNPs out of HWE is in excess of what we expect at the different alpha values; more specifically, we have too many sites out of HWE. This could be due to stratification, possibly as a consequence of problems with genotyping quality or population structure. We will deal with this issue in a little while.

Let's explore what phenotypic info is available in this dataset:

```
names(phdata(diabetes))
```

(technically, we extracted the phenotypic data from the object, and listed the names)

We can now get some summaries with

```
descriptives.trait(diabetes)
```

We will focus on type 2 diabetes (dm2 in the list of traits). Let's check whether controls for that condition better match the expected null distribution. So, let's get a summary of our markers for control cases (dm2==0)

```
descriptives.marker(diabetes, ids=(phdata(diabetes)$dm2==0))
```

This looks promising. The controls are at HWE, suggesting that the deviation was due to stratification within the cases. Let's quickly confirm that the cases are not at HWE. We can get just table 2 from the summary by subsetting the output:

```
descriptives.marker(diabetes, ids=(phdata(diabetes)$dm2==1)) [2]
```

Now that we have explored our data, let's run a first, very simple GWAS:

```
dm2.simple.gwas<- qtscore(dm2, diabetes, trait="binomial")
```

We can see the results of our analysis by calling the object;

```
dm2.simple.gwas
```

We have some information on the number of individual and SNPs, as well as a summary of results for the first ten SNPs. effB corresponds to the (approximate) Odds Ratio estimate for the SNP, and chi2.1df and P1df give the appropriate test. Further in the table, you have equivalent information for a test in which we consider specific genotypes (we won't worry about it in this practical, as in many cases we don't have enough instances of each possible genotype combination for proper testing).

So, let's plot our hits as a Manhattan plot

```
plot(dm2.simple.gwas)
```

Earlier on, we did a coarse check for stratification in the data by checking HWE in controls. A more formal test of the extent to which our data match the assumption of no stratification in the data is to estimate lambda, which give the degree of inflation (if lambda>1) or deflation (if lambda<1) in the distribution of deviations from the null distribution. Looking back at the output from the summary of our gwas object:

```
dm2.simple.gwas
```

We can see that lambda has been estimated as 1.033. Whilst this number might seem small, lambda increases linearly with sample size, so given our small sample, the deviation is somewhat concerning. We can visualize that deviation by plotting

```
estlambda(dm2.simple.gwas[, "P1df"], plot=TRUE)
```

A lambda=1 would imply a 1:1 relationship between observed and expected chisquare values, but we can see that the relationship gets a bit messy at large values. It is possible to correct for this inflation, and that is what we have in the last column of the results section from the summary of the gwas object:

```
summary(dm2.simple.gwas)
```

Column Pc1df gives us the adjusted probabilities accounting for the measured lambda. Let's replot our hits and then superimpose the adjusted p values in a different colour

```
plot(dm2.simple.gwas)
add.plot(dm2.simple.gwas,df="Pc1df",col=c("red","orange"))
```

As you can see, the adjusted p-values are generally lower than the raw values (as one would expect), but not by a huge amount (again, as expected given the not overly large deviation from lambda equal to 1). Whilst this statistical correction attempts to reshape the distribution of chisquare2 values to the expected one, it is only effective at fixing small levels of stratification. We will see in the next section of this practical how to deal with more substantial stratification. We can now get a list of the top hits of our GWAS with

```
descriptives.scan(dm2.simple.gwas, sort="Pc1df")
```

Data quality can influence a lot your hits. The GenABEL package has a user friendly function that performs a few passes on the data until some basic quality checks all pass. There are a number of quality threshold switches that can be tweaked, and they should be tailored to the data that you are using (generally a function of the technology used to generate the data, Olivier covered these issues on the first day). In this case, let's

run the quality control routine with defaults, except for ignoring HWE deviations (we want to explore what qc filtering does to it):

```
diabetes.qc<-check.marker(diabetes, p.level=0)
```

As you can see, each pass removed some markers/individuals, until the last pass had no further problematic instances. We can get a summary of our clean up with:

```
summary(diabetes.qc)
```

We can now generate a clean dataset as

```
diabetes.clean<-diabetes[diabetes.qc$idok, diabetes.qc$snpok]
```

We can remove any cases of heterozygosity on the X chromosome in males with

```
diabetes.clean<-Xfix(diabetes.clean)
```

Let's see how we are doing with HWE now:

```
descriptives.marker(diabetes.clean)[2]
```

We still have an excess of SNPs out of HWE, not much better than the original dataset:

```
descriptives.marker(diabetes)[2]
```

The next step is to investigate sub-structure in the dataset. We start by building a matrix of genome-wide IBD:

```
diabetes.ibd<-ibs(diabetes.clean[,autosomal(diabetes.clean)],weight="freq")
```

We can now use this IBD matrix to perform MDS (after transforming it into a distance matrix)

```
diabetes.mds<-cmdscale(as.dist(0.5-diabetes.ibd))
```

Plot the first two dimensions of our MDS:

```
plot(diabetes.mds)
```

We can spot that there are four individuals that form a cluster separated from the other data (as highlighted by the first dimension). We can formally identify these two clusters using a clustering algorithm:

```
diabetes.kclust<-kmeans(diabetes.mds,centers=2,nstart=100)
```

Let's figure out which cluster is the smaller one (i.e. the one with the outliers), and get the labels for those individuals:

```
outlier.cluster<- which.min(table(diabetes.kclust$cluster))
outliers<-names(which(diabetes.kclust$cluster==outlier.cluster))
```

We can now remove these outliers with

```
diabetes.clean2<-diabetes.clean[names(which(diabetes.kclust$cluster!=outlier.cluster)),]
```

Now,let's check that we have improved our HWE:

```
descriptives.marker(diabetes.clean2)[2]
```

Great, we now have decreased the SNPs out of HWE, showing that the excess was due to those outliers. For good measure, we should do a final clean up of the data, allowing for HWE checking

```
diabetes.qc2<-check.marker(diabetes.clean2)
diabetes.clean3<-diabetes.clean2[diabetes.qc2$idok, diabetes.qc2$snpok]
```

Now let's rerun the GWAS analysis with the clean dataset:

```
dm2.clean3.gwas<- qtscore(dm2, diabetes.clean3, trait="binomial")
```

And finally we can compare the top hits with this clean dataset compared to the original data:

```
descriptives.scan(dm2.clean3.gwas, sort="Pc1df")
descriptives.scan(dm2.simple.gwas, sort="Pc1df")
```

We can easily ask how many of our top 10 SNPs were in the original list

```
sum(rownames(descriptives.scan(dm2.clean3.gwas, sort="Pc1df")[2]) %in% rownames(descriptives.scan(dm2.s
```

Only 7 out 10, so cleaning your data can change things quite a bit. The final step is to estimate genome-wide significance for the GWA scan (we can't use the p values from the function above, as they fail to account for multiple testing). To do so (using a randomisation approach), we run:

```
dm2.clean3.gwas.emp<- qtscore(dm2, diabetes.clean3, trait="binomial",times=200)
descriptives.scan(dm2.clean3.gwas.emp, sort="Pc1df")
```

Note that none of our top hits are actually significant once we compute appropriate genome-wide p-values. In this simple example, given the small sample size, it made sense to remove one of the clusters as it was only composed of 4 individuals. But in larger datasets, we might have multiple large clusters (e.g. different ethnic groups), and removing any of them might not be desirable. There are a number of approaches to test for association whilst accounting for stratification. Let us go back to our dataset before we removed the outliers (diabetes.clean). We can run a stratified analysis simply specifying the argument strata in qtscore:

```
dm2.strat.gwas <- qtscore(dm2, diabetes.clean, trait="binomial",strata= diabetes.kclust$cluster)
```

Let's compare the hits for the original data, after removing outliers, and when accounting for stratification:

```
par(mfcol=c(3,1))
plot(dm2.simple.gwas,ylim=c(1,4))
plot(dm2.clean3.gwas,ylim=c(1,4))
plot(dm2.strat.gwas,ylim=c(1,4))
par(mfcol=c(1,1))
```

The differences are not overly large (some differences visible especially in the middle of chr 2 and in chr 3), but note that the results between the dataset without the outliers and the dataset analysed with stratification are virtually identical (in this case, since the removed cluster was very small, we don't get much additional information including it, but we get different results if we include it whilst ignoring stratification). Now let's see how we could correct for stratification using PCA of the kinship matrix.

```
dm2.pca.gwas<-egscore(dm2, data=diabetes.clean, kin= diabetes.ibd)
```

We can compare the results between stratifying the analysis by cluster and using the full kinship matrix:

```
par(mfcol=c(2,1))
plot(dm2.strat.gwas,ylim=c(1,4))
plot(dm2.pca.gwas,ylim=c(1,4))
par(mfcol=c(1,1))
```

Whilst the results are broadly similar, note that, on the X chromosome, using the full kinship matrix removes some extreme values (which therefore were likely to be due to some cryptic stratification within the clusters). Finally, we can explore what would happen if we include covariates (sex and age), thus accounting for possible differences among the subjects:

```
dm2.cov.gwas<- egscore(dm2~sex+age, data=diabetes.clean, kin= diabetes.ibd)
par(mfcol=c(2,1))
plot(dm2.pca.gwas,ylim=c(1,4))
plot(dm2.cov.gwas,ylim=c(1,4))
```

```
par(mfcol=c(1,1))
```

In this dataset, the effect is limited, but note that the landscape of hits on chromosome 2 does change somewhat.

If you were interested in more advanced applications of GenABEL for GWAS analysis, there is a very extensive tutorial document available on the web: http://www.genabel.org/tutorials (you will see that this practical was heavily inspired by some of the sections in that document).