

Day 2 PM Practical - Phasing and Imputation

Imputation can be used to help refine associated regions, or to increase power by generating a denser set of SNPs in common between different case/control cohorts, for subsequent meta-analysis.

From the chr22 dataset we analysed on DAY2 AM, we can carry out pre-phasing and imputation using **SHAPEIT2** and **IMPUTE2** to investigate the signal of association at a much denser set of SNPs. Also, we will prepare data for the **Michigan imputation server**

<https://imputationserver.sph.umich.edu/index.html#!> and process the resulting post-imputation files.

Phasing is best carried out on the combined set of cases and controls together. Earlier we created **PLINK**-format files that contained QCed data from cases and controls. These will need to be converted to Oxford format for use with **SHAPEIT2** and **IMPUTE2**.

First, use **PLINK** to *convert* the **PLINK**-format `casecon.qc` files that we created previously into the Oxford format (`casecon.qc.gen`, `casecon.qc.sample`):

```
plink --bfile casecon.qc \  
--allow-no-sex --recode oxford \  
--out casecon.qc
```

We can now run **SHAPEIT2** on the genotype and QC-ed data (`casecon.qc.gen`, `casecon.qc.sample`). **SHAPEIT2** is fast but it would still take too long to phase the whole of chr22 in the time we have for this practical so we first focus on a small region. Let us consider the region 22-23Mb that has already shown some evidence of a signal of association in the analysis we did earlier (Day 2 AM).

We can run **SHAPEIT2** on this region using the following command. Notice that we actually run the phasing on a slightly larger region from 21.5-23.5Mb.

```
shapeit_v2 \  
--input-gen casecon.qc.gen casecon.qc.sample \  
--input-map genetic_map_chr22_combined_b36.txt \  
--output-max casecon.qc.21.5-23.5Mb.haps casecon.qc.haps.sample \  
--thread 4 \  
--input-from 21500000 \  
--input-to 23500000
```

NOTES

- (a) The inclusion of 500Kb “buffer regions” on either side of the region of interest (specified by the `--input-from` and `--input-to` options) is needed to avoid any drop in imputation performance towards the edge of the region when carrying out imputation.
- (b) Notice also that we have used the `-thread 4` option. **SHAPEIT2** is multi-threaded which means it can take advantage of multiple processor cores when available. Using a value of 4 will make the command run around 4 times faster.
- (c) The `--input-map` option is needed to specify the recombination map across the chromosome that specifies the rate of switching in the HMM model that **SHAPEIT2** uses.

This command results in a set of haplotypes stored in the file `casecon.qc.21.5-23.5Mb.haps`

To understand more about the commands used by **SHAPEIT2** either

- (a) look at the **SHAPEIT2** webpage https://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#home
- (b) type `shapeit_v2 --help` for a full list of all options.

We can now impute genotypes at SNPs using the HapMap CEU reference panel using **IMPUTE2**. The command is as follows

```
impute_v2.3.2 \  
-use_prephased_g \  
-known_haps_g casecon.qc.21.5-23.5Mb.haps \  
-m genetic_map_chr22_combined_b36.txt \  
-h genotypes_chr22_CEU_r22_nr.b36_fwd_phased_by_snp \  
-l genotypes_chr22_CEU_r22_nr.b36_fwd_legend_by_snp \  
-int 22000000 23000000 \  
-Ne 15000 \  
-buffer 500 \  
-o casecon.qc.22-23Mb.imputed.gen
```

This produces an imputed file `casecon.qc.22-23Mb.imputed.gen`

NOTES

- (a) In this command we have used the `-int` option to specify that we are only interested in the region 22-23Mb. We use the option `-buffer 500` to specify a 500Kb buffer region.

- (b) The reference haplotypes are specified using the `-h` and `-l` options that provide the haplotypes file and the legend file respectively. Files like these for each chromosome and each reference panel can be downloaded from the **IMPUTE2** website.
- (c) As we are working with simulated data we do not have to worry about aligning strand so the `-s` flag is not used. On real data, this is an important issue that we must consider.

IMPUTE2 produces an assessment of its imputation accuracy, which is printed to the screen. An example is given below. In this example, it is estimated that 88.7% of genotypes are called with a probability above 0.9, and of these genotypes, 99.4% are correctly called (this is highlighted in **ORANGE** below).

Imputation accuracy assessment

The table below is based on an internal cross-validation that is performed during each IMPUTE2 run. For this analysis, the program masks the genotypes of one variant at a time in the study data (Panel 2) and imputes the masked genotypes by using the remaining study and reference data. The imputed genotypes are then compared with the original genotypes to produce the concordance statistics shown in the table. You can learn more about this procedure and the contents of the table at http://mathgen.stats.ox.ac.uk/impute/concordance_table_description.html.

In the current analysis, IMPUTE2 masked, imputed, and evaluated 242375 genotypes that were called with high confidence (maximum probability ≥ 0.90) in the Panel 2 input file (`-g` or `-known_haps_g`).

When the masked study genotypes were imputed with reference data from Panel 0, the concordance between original and imputed genotypes was as follows:

Interval	#Genotypes	%Concordance	Interval	%Called	%Concordance
[0.0-0.1]	0	0.0	[≥ 0.0]	100.0	97.0
[0.1-0.2]	0	0.0	[≥ 0.1]	100.0	97.0
[0.2-0.3]	0	0.0	[≥ 0.2]	100.0	97.0
[0.3-0.4]	0	0.0	[≥ 0.3]	100.0	97.0
[0.4-0.5]	1187	45.7	[≥ 0.4]	100.0	97.0
[0.5-0.6]	4900	58.3	[≥ 0.5]	99.5	97.2
[0.6-0.7]	4673	71.1	[≥ 0.6]	97.5	98.0
[0.7-0.8]	6482	82.3	[≥ 0.7]	95.6	98.6
[0.8-0.9]	10079	91.0	[≥ 0.8]	92.9	99.0
[0.9-1.0]	215054	99.4	[≥ 0.9]	88.7	99.4

Run **SNPTEST** on this imputed file

```
snptest_v2.5.2 \  
-data casecon.qc.22-23Mb.imputed.gen casecon.qc.haps.sample \  
-frequentist 1 \  
-bayesian 1 \  
-method score \  
-pheno phenotype \  
-o snptest_assoc_imp.txt
```

and plot the results in R

```
#read in the association results
```

```
assoc.imp.snptest = read.table("snptest_assoc_imp.txt",  
header = T,as.is=T)
```

```
#open .png file
```

```
png("impute2.png")
```

```
#set R output format for two plots, side by side
```

```
par(mfrow=c(1,2))
```

```
#plot log10 Bayes factors
```

```
plot(assoc.imp.snptest$pos, assoc.imp.snptest$bayesian_add_log10_bf, ylim = c(0, 10),  
main = "log10 BFs", xlab = "position",  
ylab = "log10 Bayes factor",  
col = 1 + 1*(assoc.imp.snptest[,1] == "---"), pch = 16)
```

```
#add legend (red points are the imputed SNPs and black points are  
#the typed SNPs)
```

```
legend("topleft", col = c(1,2), c("Typed", "Imputed"), pch = 16)
```

```
#plot -log10 p-values
```

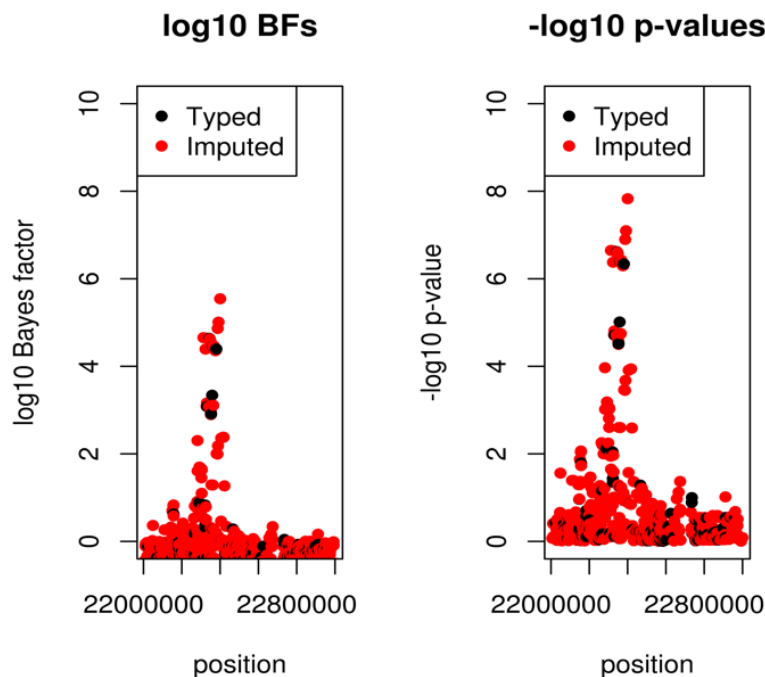
```
plot(assoc.imp.snptest$pos,  
-log10(assoc.imp.snptest$frequentist_add_pvalue),  
ylim = c(0, 10), main="-log10 p-values", xlab="position",  
ylab="-log10 p-value",  
col = 1 + 1*(assoc.imp.snptest[,1] == "---"), pch = 16)
```

```
#add legend
```

```
legend("topleft", col = c(1,2), c("Typed", "Imputed"), pch = 16)
```

```
#close .png file
```

```
dev.off()
```



Whole chromosome imputation with Michigan Server

The Michigan and Sanger Imputation Servers are useful for large-scale (whole-genome) imputation jobs, or if you want to use the HRC reference panel. (The TOPMed Imputation Server is another option which allows use of the TOPMed reference panel).

First, we prepare the input files for uploading to the Michigan imputation server, using steps (i) – (v) below:

(i) First compute allele frequencies for subsequent use in step (iii):

```
plink --bfile casecon.qc --allow-no-sex \
--freq --out casecon.qc.freq
```

This command produces an output file `casecon.qc.freq.frq` that contains frequencies of the minor alleles.

(ii) In R, extract a list of palindromic SNPs (SNPs with alleles on the positive strand that could match the alleles on the negative strand) for subsequent exclusion, if desired:

```
#read in casecon.qc.bim file
bim = read.table("casecon.qc.bim", header = F)

#get indices of A/T and G/C SNPs
w = which ((bim$V5=="A" & bim$V6=="T") |
           (bim$V5=="T" & bim$V6=="A") |
           (bim$V5=="C" & bim$V6=="G") |
           (bim$V5=="G" & bim$V6=="C"))

#extract A/T and G/C SNPs
at.cg.snps = bim[w,]

#save A/T and G/C SNPs into a file at-cg-snps.txt
write.table(at.cg.snps$V2, "at-cg-snps.txt", row.names = F, col.names = F, quote = F)
```

(iii) Run the following **PERL** script (available from Will Rayner <http://www.well.ox.ac.uk/~wrayner/tools/>) for validating the SNPs against the HRC reference panel for strand, id names, positions and alleles.

WARNING - this step can be very slow, as reads in information from the whole genome for the HRC reference panel. If it has not finished within a reasonable amount of time (e.g. 5-10 minutes), we suggest that you stop the script (by pressing **CTRL C**) and instead copy the files above from a directory where we have already created them for you:

DAY2_IMP_WillRayner_OUTPUT_FILES

with the following UNIX command

```
cp ~/DAY2_IMP_WillRayner_OUTPUT_FILES/* .
```

To run the script yourself, type:

```
perl HRC-1000G-check-bim-v4.3.0.pl -b \
casecon.qc.bim -f casecon.qc.freq.frq \
-r HRC.r1-1.GRCh37.wgs.mac5.sites.tab -h
```

The script produces the output files for subsequent use in step (v):

```
FreqPlot-casecon.qc-HRC.txt
LOG-casecon.qc-HRC.txt
ID-casecon.qc-HRC.txt
Position-casecon.qc-HRC.txt
Chromosome-casecon.qc-HRC.txt
Exclude-casecon.qc-HRC.txt
Strand-Flip-casecon.qc-HRC.txt
Force-Allele1-casecon.qc-HRC.txt
```

```
Run-plink.sh
```

(iv) Run the **Unix** script `Run-plink.sh` that contains **PLINK** commands, to create a VCF file `casecon.qc-updated-chr22.vcf`. (The first command below makes the script executable, while the 2nd command actually runs it). Note that, if we wanted, we could edit this script to add in a **PLINK** command to first remove all the palindromic SNPs that we identified earlier, whereas Will Rayner's tool only removes those whose allele frequencies are too close to 0.5 for it to be able to infer that they should be flipped.

```
chmod a+x Run-plink.sh
```

```
./Run-plink.sh
```

Note that this **Unix** script only creates a VCF file for chromosome 22 because we only provided real genotype data for chromosome 22 to Will Rayner's **PERL** script. If we had provided genome-wide data, the **Unix** script `Run-plink.sh` would have contained additional lines to deal with the other chromosomes.

(v) Compress the file `casecon.qc-updated-chr22.vcf` using `bgzip`. (Note that the SNPs first need to be sorted in order of BP position. **PLINK** should have done this automatically when creating `casecon.qc-updated-chr22.vcf`. If you instead wanted to do this yourself, you could also use a different utility such as `vcf-sort`).

```
bgzip -c casecon.qc-updated-chr22.vcf > casecon.qc.impute.vcf.gz
```

This command creates a compressed file `casecon.qc.impute.vcf.gz` that is ready for uploading to the Michigan imputation server. Once you upload it, the imputation job might take between a few hours and a few days to complete.

After the imputation job has finished, you will be sent a web link to download the results in a (compressed) `chr_22.zip` file and a password for unzipping it. After unzipping `chr_22.zip`, you will get 2 compressed files:

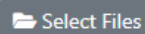
```
chr22.info.gz  
chr22.dose.vcf.gz
```

Here, we have provided these files for you (after running the imputation job ourselves on the Michigan server, selecting Minimac4 as the imputation engine, Eagle 2.4 phasing and HRC r 1.1 as the reference panel – see the screenshot below).

Reference Panel ([Details](#)) HRC r1.1 2016 (GRCh37/hg19) ▼

Input Files ([VCF](#))

 casecon.qc.impute.vcf.gz

 Select Files

Multiple files can be selected by using the **ctrl** / **cmd** or **shift** keys.

Array Build GRCh37/hg19 ▼

Please note that the final SNP coordinates always match the reference build.

rsq Filter off ▼

Phasing Eagle v2.4 (phased output) ▼

Population EUR ▼

Mode Quality Control & Imputation ▼

Processing post-imputation files

First, we need to filter out SNPs with information score lower than 0.5, to produce a file `casecon.qc.filtered.vcf` that contains only SNPs with info score ≥ 0.5 .

WARNING - this step can also be very slow. If you prefer not to wait (e.g. 5-10 minutes), we suggest that you skip it, and instead use a copy of the resulting output file that we have provided for you named `copy.casecon.qc.filtered.vcf`

You can use this file in the commands below instead of `casecon.qc.filtered.vcf`.

If you prefer to run the command yourself to generate the file `casecon.qc.filtered.vcf` you should type:

```
bcftools filter --exclude 'INFO/R2<0.5' \  
--output casecon.qc.filtered.vcf chr22.dose.vcf.gz
```


Sometimes the VCF files obtained from imputation contain duplicate SNPs at the same position. To make the subsequent analyses simpler, you may prefer not to keep these SNPs. The following commands scan the file `casecon.qc.filtered.vcf` (with respect to the second column that contains the SNP positions) and creates an output file `casecon.qc.filtered.pos.dups` with the duplicate positions listed:

```
cut -f 2 casecon.qc.filtered.vcf > positions.txt
```

```
sort positions.txt > sortedpositions.txt
```

```
uniq -d sortedpositions.txt > casecon.qc.filtered.pos.dups
```

We can now transform the `casecon.qc.filtered.vcf` file into PLINK format:

```
plink --vcf casecon.qc.filtered.vcf --allow-no-sex \
--make-bed --out casecon.qc.filtered.plinkformat
```

We can use the resulting `casecon.qc.filtered.plinkformat.bim` file, together with `casecon.qc.filtered.pos.dups`, to create a file with SNP IDs that have duplicate position values for subsequent removal. There are various ways to do this; we will do it in R:

```
filteredbim=read.table("casecon.qc.filtered.plinkformat.bim", header=F)
```

```
dupposns=read.table("casecon.qc.filtered.pos.dups", header=F)
```

```
mg=merge(dupposns, filteredbim, by.x="V1", by.y="V4")
```

```
write.table(mg[,3], file="casecon.qc.filtered.dups", col.names=F,
row.names=F, quote=F)
```

We can now again transform the `casecon.qc.filtered.vcf` file into PLINK format using the following **PLINK** command that reads the file `casecon.qc.filtered.vcf`, removes the SNPs listed in the file `casecon.qc.filtered.dups` and outputs only biallelic SNPs with genotype probability ≥ 0.9 .

```
plink --vcf casecon.qc.filtered.vcf \
```

```
--exclude casecon.qc.filtered.dups \  
--vcf-min-gp 0.9 --biallelic-only --allow-no-sex \  
--make-bed --out casecon.qc.filtered
```

The command create files

```
casecon.qc.filtered.bim  
casecon.qc.filtered.fam  
casecon.qc.filtered.bed
```

Note that the file `casecon.qc.filtered.fam` does not contain the case/control values (its last column has values -9). We can update the case/control status using the pre-specified `casecon.qc.pheno.txt` file (provided) with the case/controls status for each individual. The following **PLINK** command updates the case/control status, and also filters the SNPs by MAF, HWE and missing rate. It creates `casecon.qc.imp.bim`, `casecon.qc.imp.fam` and `casecon.qc.imp.bed` output files.

```
plink --bfile casecon.qc.filtered \  
--allow-no-sex --pheno casecon.qc.pheno.txt \  
--maf 0.01 --geno 0.1 --hwe 1e-6 \  
--make-bed --out casecon.qc.imp
```

Association analysis for the imputed genotypes

We can now run the association test with **PLINK** on the imputed and typed SNPs. For the typed SNPs, we use the version with updated basepair positions (and chromosome assignments) that were used to inform the imputation, retaining only SNPs whose updated positions were still on chromosome 22.

```
plink --bfile casecon.qc.imp \  
--allow-no-sex --logistic beta --ci 0.95 \  
--out casecon.qc.imp.log
```

```
plink --bfile casecon.qc-updated --chr 22 \  
--allow-no-sex --logistic beta --ci 0.95 \  
--out casecon.qc.typed.log
```

and compare with the association results for the typed SNPs:

```
#read in the association test results for the typed SNPs
assoc.qc = read.table("casecon.qc.typed.log.assoc.logistic", header = T, as.is = T)

#read in the association test results for the imputed SNPs
assoc.qc.imp = read.table("casecon.qc.imp.log.assoc.logistic", header = T, as.is = T)

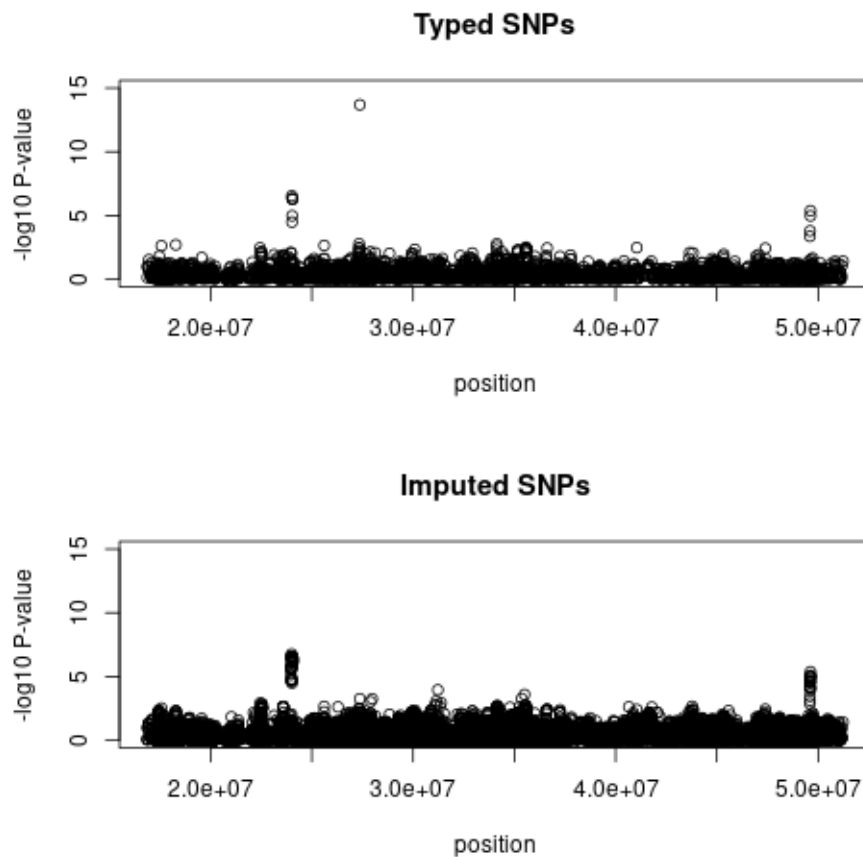
#open .png file
png("typed_imputed.png")

#set R output format for two plots
par(mfrow = c(2,1))

#Plot -log10 p-values for the typed SNPs
plot(assoc.qc$BP, -log10(assoc.qc$P), main = "Typed SNPs",
     ylab = "-log10 P-value", xlab = "position", ylim=c(0,15))

#Plot -log10 p-values after the imputation
plot(assoc.qc.imp$BP, -log10(assoc.qc.imp$P),
     main = "Imputed SNPs",
     ylab = "-log10 P-value", xlab = "position", ylim=c(0,15))

#close .png file
dev.off()
```



Interestingly the “suspicious” genotyped SNP does not look very significant in the imputed version of the data. We can check this in R by typing:

```
assoc.qc[assoc.qc$P < 1E-08,]
```

```
assoc.qc.imp[assoc.qc.imp$BP == 27370273,]
```

It seems that the imputed version of this SNP is much less significant (p-value = 0.0005571), suggesting that perhaps there were genotyping errors at this SNP that have been corrected when imputing it, by incorporating information from the surrounding SNPs.

Association analysis for the imputed genotypes in the replication data

Similarly, we can impute the replication data set. Here, we have provided the imputed, QCed, replication data set in PLINK format:

```
rep.imp.bed  
rep.imp.bim  
rep.imp.fam
```

Run the association test with **PLINK** on the imputed SNPs:

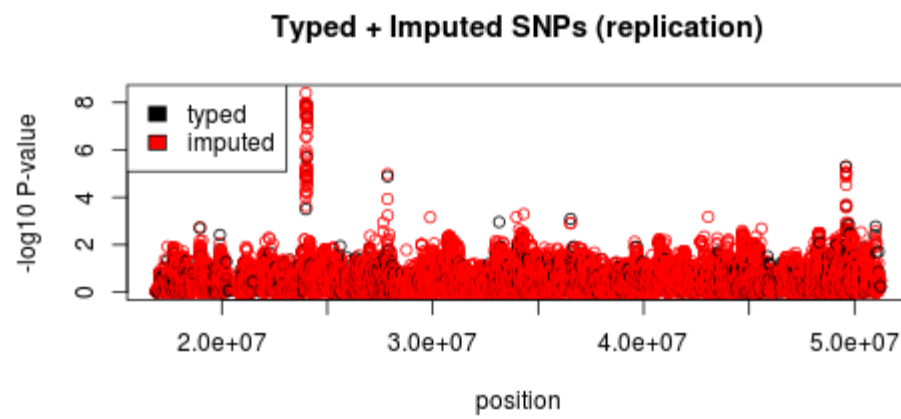
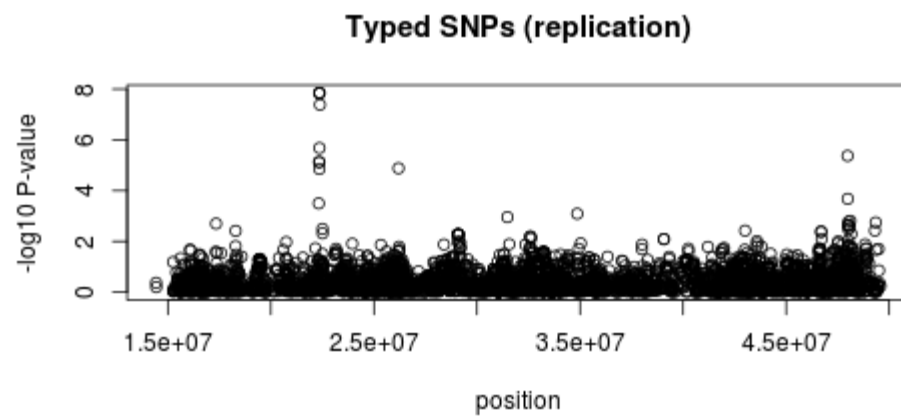
```
plink --bfile rep.imp \
--allow-no-sex --logistic beta --ci 0.95 \
--out rep.imp.log
```

and compare the results with the results you obtained previously for the genotyped SNPs using the following **R** commands:

```
#read in the association test results for the typed replication #data
assoc.rep = read.table("rep.log.assoc.logistic", header = T, as.is = T)
```

```
#read in the association test results for the imputed replication #data
assoc.rep.imp = read.table("rep.imp.log.assoc.logistic",
header = T,as.is = T)
```

```
#open .png file
png("imputed_typed_replication.png")
#set R output format for two plots
par(mfrow = c(2,1))
#Plot -log10 p-values before the imputation
plot(assoc.rep$BP, -log10(assoc.rep$P),
main = "Typed SNPs (replication)",
ylab = "-log10 P-value", xlab = "position")
#Colour the imputed SNPs in red. Use the column "Genotyped" in
#the rep.imp.info.gz file to identify imputed SNPs.
rep.info = read.table(gzfile("rep.imp.info.gz"), header = T, as.is = T)
rep.imputed = rep.info$SNP[rep.info$Genotyped == "Imputed"]
rep.w = which (assoc.rep.imp$SNP %in% rep.imputed)
rep.colour = rep(1, times = nrow(assoc.rep.imp))
rep.colour[rep.w] = 2
#Plot -log10 p-values after the imputation
plot(assoc.rep.imp$BP, -log10(assoc.rep.imp$P),
main = "Typed + Imputed SNPs (replication)", col = rep.colour, ylab = "-log10 P-value",
xlab = "position")
#add legend
legend("topleft", legend = c("typed", "imputed"), fill = 1:2)
#close .png file
dev.off()
```



Meta-analysis of the imputed data

To carry out a meta-analysis of the imputed data sets, first use R to prepare **META** input files:

(i) Get the non-effect allele from the sixth column of the .bim files:

```
casecon.qc.imp.bim = read.table("casecon.qc.imp.bim", as.is = T)
casecon.qc.imp.A2 = casecon.qc.imp.bim$V6
```

```
rep.imp.bim = read.table("rep.imp.bim", as.is = T)
rep.imp.A2 = rep.imp.bim$V6
```

(ii) Get INFO scores:

```
#Get INFO scores for the original imputed file
#####
#read in info file
info = read.table(gzfile("chr22.info.gz"), header = T, as.is = T)

#filter out info score <0.5
info.good = info[info$Rsq >= 0.5,]

#find indices of the relevant snps
m = match(casecon.qc.imp.bim$V2, info.good$SNP)

#info for the relevant snps
casecon.qc.imp.info = info.good[m,]

#Get INFO scores for the replication
#####
rep.info = read.table(gzfile("rep.imp.info.gz"), header = T, as.is = T)
rep.info.good = rep.info[rep.info$Rsq >= 0.5,]
m = match(rep.imp.bim$V2, rep.info.good$SNP)
rep.imp.info = rep.info.good[m,]
```

(iii) Create the META input files:

#META input file for the imputed original data

```
assoc.qc.imp = read.table("casecon.qc.imp.log.assoc.logistic", header = T, as.is = T)
casecon.qc.meta = data.frame(chr = assoc.qc.imp$CHR,
rsid = assoc.qc.imp$SNP,
pos = assoc.qc.imp$BP, allele_A = casecon.qc.imp.A2,
allele_B = assoc.qc.imp$A1, P_value = assoc.qc.imp$P,
info = casecon.qc.imp.info$Rsq, beta = assoc.qc.imp$BETA,
se = assoc.qc.imp$SE)
```

```
write.table(casecon.qc.meta, "casecon.qc_impute_meta.txt", row.names = F,
col.names = T, quote = F)
```

#META input file for the imputed replication data

```
assoc.rep.imp = read.table("rep.imp.log.assoc.logistic", header = T, as.is = T)
rep.imp.meta = data.frame(chr = assoc.rep.imp$CHR,
rsid = assoc.rep.imp$SNP,
pos = assoc.rep.imp$BP, allele_A = rep.imp.A2,
allele_B = assoc.rep.imp$A1,
P_value = assoc.rep.imp$P, info = rep.imp.info$Rsq,
beta = assoc.rep.imp$BETA,
se = assoc.rep.imp$SE)
```

```
write.table(rep.imp.meta, "rep_impute_meta.txt", row.names = F, col.names = T,
quote = F)
```

Now run a meta-analysis of the imputed data:

```
meta_v1.7 \
--cohort casecon.qc_impute_meta.txt rep_impute_meta.txt \
--method 1 --output meta.imp.txt
```

The following output will be produced:

```
META v1.7
=====

==> Standard output used
==> Method being used: inverse-variance method (fixed effects
model)
==> OUTPUT: meta.imp.txt

Reading the data ...
==> 2 cohorts being used:
```



```
Cohort 1: casecon.qc_impute_meta.txt, 45978 out of 45978 (100%)
SNPs are used (threshold >= 0.5)
Cohort 2: rep_impute_meta.txt, 52285 out of 52285 (100%) SNPs are
used (threshold >= 0.5)
```

```
There are 59115 SNPs in the union list.
0 SNPs have had their alleles flipped to make strand consistent
952 SNPs have had the order of their alleles normalized
0 SNPs have been removed as their alleles are not consistent
across cohorts
```

```
Writing the data ...
```

```
DONE!
```

NOTE: For 952 SNPs, the effect and the non-effect alleles were different across the two cohorts, e.g. have a look at SNP 22:17316555

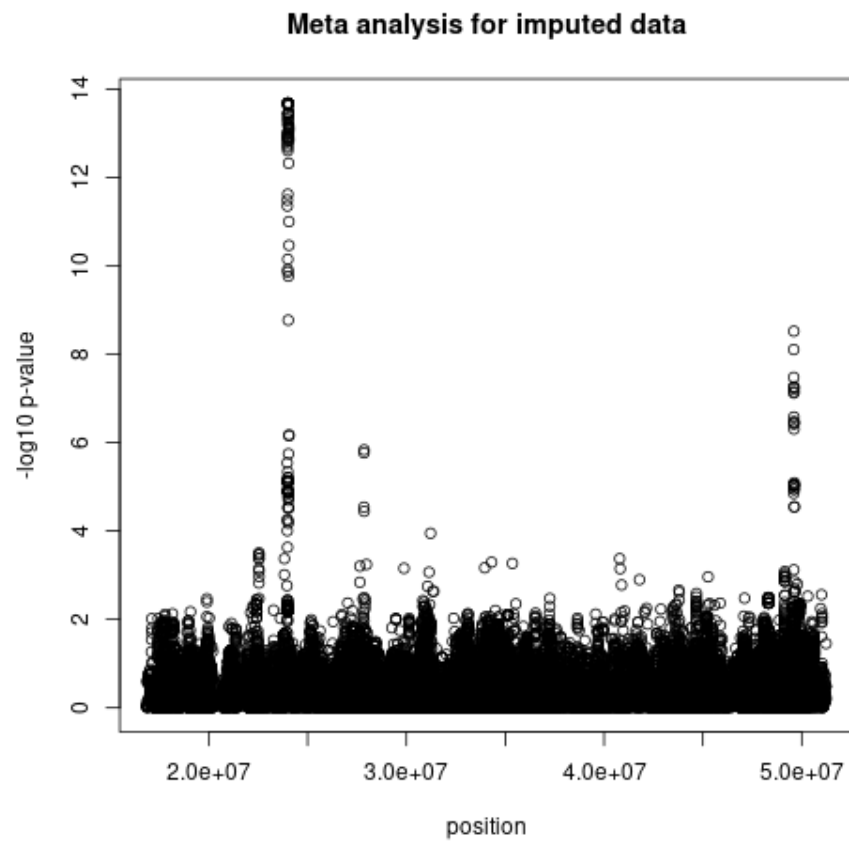
```
grep 17316555 casecon.qc.imp.bim
22      22:17316555:C:T 0      17316555      T      C
grep 22:17316555 rep.imp.bim
22      22:17316555      0      17316555      C      T
```

META takes this into account by changing the coefficients of the logistic regression accordingly.

You can read the results into **R** and make a plot:

```
#read in the results file
meta.imp = read.table("meta.imp.txt", header=T, as.is=T)
#open .png file
png("meta_imputed.png")

#set the output format
par(mfrow = c(1,1))
#plot -log10 p-values for the meta analysis
plot(meta.imp$pos, -log10(meta.imp$P_value),
main = "Meta analysis for imputed data", xlab = "position",
ylab = "-log10 p-value")
#close .png file
dev.off()
```



How does this compare to the meta-analysis of the genotyped SNPs that you did in the previous (Day 2 AM) exercise?