WaveQTL

June 19, 2014

1 Overview

The `WaveQTL` software implements a wavelet-based approach for genetic association analysis of functional phenotypes (e.g. sequence data arising from high-throughput sequencing assays) described in [4]. A key factor that distinguishes a wavelet-based approach from typical association analysis is to better exploit high-resolution measurements provided by high-throughput sequencing assays (see [4] for motivations). The implemented wavelet-based methods aim to

- test for genetic association of functional phenotype;
- provide explanations for observed associations such as which parts and features of the data are driving the association;
- make applications to large-scale genetic association analyses computationally tractable.

Although wavelet methods were motivated primarily by genetic association studies for high-throughput sequencing data, they could also test for association between functional data and other covariates, either continuous or discrete. For example, in a genomics context, it could be used to detect differences in gene expression (from RNA-seq data) or TF binding (from ChIP-seq data) measured on two groups (e.g. treatment conditions or cell types). Or it could be used to associate a functional phenotype, such as chromatin accessibility, with a continuous covariate, such as "overall" expression of a gene. It could also be used for genome-wide association studies of functional phenotypes unrelated to sequencing.

We describe how to preprocess functional phenotypic data and produce inputs required by `WaveQTL` in Section 3, how to perform an analysis using `WaveQTL`, with description of input file formats and output files, in Section 4, and how to construct effect size estimate in data space from `WaveQTL` outputs in Section 5. As an example data, we use dsQTL data shown in Figure 2 of [4] containing DNase-seq data at chr17.10160989.10162012 and genotypes at 24 SNPs in 2kb cis-candidate region on 70 individuals. The dsQTL data (`chr17.10160989.10162012.pheno.dat`

and chr17.10160989.10162012.2kb.cis.geno) are provided in WaveQTL/data/dsQTL/ directory. In Section 6, we provide instructions on how to produce those two files from raw data files downloaded from http://eqtl.uchicago.edu/dsQTL_data/ (e.g., genotype data and DNase-seq data in the entire genome). One can use the same procedure to generate genotype data and phenotype data for other sites.

2 How to cite WaveQTL

Heejung Shim and Matthew Stephens (2014). Wavelet-based genetic association analysis of functional phenotypes arising from high-throughput sequencing assays. arXiv. 1307.7203.

3 Functional phenotypic data preprocessing

The WaveQTL software takes normalized wavelet coefficients (WCs) as an input phenotype. We provide multiple R functions in R/WaveQTL_preprocess_funcs.R that preprocess functional phenotypic data as described in [4] and produce inputs required by WaveQTL. Here, we describe how to use the main function WaveQTL_preprocess using dsQTL data from [4] as an example. R/WaveQTL_preprocess_example.R contains R scripts used in this section. For details of preprocessing procedure, please read our paper [4], and for details of the function WaveQTL_preprocess and other functions called by WaveQTL_preprocess, see the description of functions in R/WaveQTL_preprocess_funcs.R.

The main function WaveQTL_preprocess takes functional data ("Data") and optionally library read depth ("library.read.depth"), a list of covariates ("Covariates"), minimum read count for filtering ("meanR.thresh") as an input. If "library.read.depth" is provided, the function standardizes the functional data by the library read depth to account for different library read depths across individuals. Then, the function decomposes the (standardized) functional data into WCs using a wavethresh R package and normalizes the WCs. If "Covariates" are provided, the function corrects the WCs for the Covariates during the normalization. See the description of the function Normalize.WCs in WaveQTL_preprocess_funcs.R for details of normalization. Users can specify which type of wavelet transform should be applied by using "filter.number" and "family" in input arguments. They are arguments to the function wd in the R package wavethresh (for details, see their manual http://cran.r-project.org/web/packages/wavethresh/wavethresh.pdf). For now, the function doesn't allow users to specify the level of wavelet decomposition and the function uses the maximum level decomposition. In addition to wavelet transform, this function filters out some WCs that are computed based on low counts. The function considers WCs to have low count if the total counts used in their computation are less than or equal to "meanR.thresh" per individual on average. Finally, the function

`WaveQTL_preprocess` returns quantile transformed (standardized and covariates corrected) WCs ("WCs") and filtering status for each WCs ("filtered.WCs") as an output.

We start with making a directory to save output from the function `WaveQTL_preprocess`.

```
cd WaveQTL
mkdir test
cd test
mkdir dsQTL
```

Now open `R` session, set working directory to `WaveQTL/R` and read functions in `WaveQTL_preprocess_funcs.R`.

```
> setwd("~/WaveQTL/R")
> source("WaveQTL_preprocess_funcs.R")
```

Then, set a path to the directory containing dsQTL data that is shown in Figure 2 of [4] and read data on 70 individuals.

```
> data.path = "../data/dsQTL/"
# read functional phenotypic data
> pheno.dat = as.matrix(read.table(paste0(data.path,
+ "chr17.10160989.10162012.pheno.dat")))
> dim(pheno.dat)
[1] 70 1024
# read library read depth
> library.read.depth = scan(paste0(data.path, "library.read.depth.dat"))
> length(library.read.depth)
[1] 70
# read Covariates
> Covariates = as.matrix(read.table(paste0(data.path, "PC4.dat")))
> dim(Covariates)
[1] 70 4
```

Each row of `pheno.dat` contains functional phenotypic data for each individual on a region of 1024bp (B = 1024). For example, this functional phenotypic data can be the number of reads starting at each location of the region. In our dsQTL analysis of [4], we obtained functional phenotypic data after applying multiple procedures into read counts to avoid potential biases. We describe how to replicate those procedures in Section 6. Each element of `library.read.depth` contains library read depth for each individual. `Covariates` contains multiple covariates to be

corrected for (here, four principal components we used to control for confounding factors in our dsQTL analysis of [4]).

We set two to "meanR.thresh" for filtering of low count WCs and run `WaveQTL_preprocess`. Here, we use default values for "filter.number" and "family" arguments, which corresponds to the Haar DWT.

```
> meanR.thresh = 2
> res = WaveQTL_preprocess(Data = pheno.dat, library.read.depth =
+ library.read.depth, Covariates = Covariates, meanR.thresh = meanR.thresh)
> str(res)
List of 2
 $ WCs         : num [1:70, 1:1024] -1.415 0.126 0.347 -1.323 0.674 ...
 $ filtered.WCs: num [1:1024] 1 1 1 1 1 1 1 1 0 0 ...
```

`WaveQTL_preprocess` returns a list of `WCs` and `filtered.WCs`. `WCs` is a matrix of the number of individuals (70) by the number of WCs (1024). Each row contains (standardized, covariates corrected, and quantile transformed) WCs for each individual. WCs are ordered from low resolution WC to high resolution WC. For example, with a Haar wavelet transform, the first column contains WC (precisely speaking, scaling coefficient) that corresponds to sum of data in the region. The second column contains WC that contrasts the data in the first half vs second half of the region. The last (1024) column contains WC that contrasts the data in the 1023-th base vs 1024-th base. `filtered.WCs`, a vector of length 1024, contains indicators to show whether t-th WC in output `WCs` is filtered (0) or not (1) due to low count.

Finally, set a path to the output directory we created at the beginning and save output as files.

```
> output.path = "../test/dsQTL/"
> write.table(res$WCs, file= paste0(output.path, "WCs.txt"), row.names=FALSE,
+ col.names = FALSE, quote=FALSE)
> cat(res$filtered.WCs, file = paste0(output.path, "use.txt"))
```

4 WaveQTL

4.1 Input file format for WaveQTL

`WaveQTL` requires two input files containing genotypes and phenotype (WCs), and optionally two input files containing filtering status of WCs and information on which WCs share the same hyper parameter π (see [4] for definition of π).

4.1.1 Genotype file format

WaveQTL takes genotype file in the BIMBAM format (http://stephenslab.uchicago.edu/software.html) [3]. To see details of the different BIMBAM genotype file formats and option for each format, please read the BIMBAM manual (http://www.haplotype.org/download/bimbam-manual.pdf). Among the BIMBAM genotype file formats, a "mean genotype file format" is particularly useful for imputed genotypes as well as for other covariates other than SNPs (e.g., control/treatment group indicator). The first column of the mean genotype file is the SNP id and the second and third columns are allele types with minor allele first. The remaining columns contain the (posterior) mean genotypes (numbers between 0 and 2) of each individual. An example of mean genotype file of 24 SNPs and 70 individuals are in WaveQTL/data/dsQTL/chr17.10160989.10162012.2kb.cis.geno. The first three rows and the first 23 columns in chr17.10160989.10162012.2kb.cis.geno are:

```
chr17.10159002 G A 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0.543 0 0.058 0
chr17.10159236 T C 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0.002 0 0.003 0
chr17.10160091 C T 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 2
```

One can use the following bash command (in one line) to generate BIMBAM mean genotype file from IMPUTE genotype file (http://www.stats.ox.ac.uk/~marchini/software/gwas/file_format.html) [2]:

```
cat [impute filename] | awk -v s=[number of samples/individuals]
'{ printf $2 "," $4 "," $5; for(i=1; i<=s; i++) printf "," $(i*3+3)*2+$(i*3+4);
printf "\n" }'
> [bimbam filename]
```

Notice that one may need to manually input the two quote symbols ‘ . Depending on the terminal, a direct copy/paste of the above line may result in bash: syntax error near unexpected token ‘(’ ” errors. This paragraph is copied from the GEMMA manual (http://www.xzlab.org/software/GEMMAmanual.pdf)

WaveQTL can take a non-genotype covariate (e.g., control/treatment group indicator) for association analysis. The input file can be prepared in the same format as mean genotype file described above, with arbitrary SNP id and allele coding using ACGT in the first three columns. User should use the "-notsnp" option to disable the minor allele frequency cutoff and to use any numerical values as covariates.

4.1.2 Phenotype file format

The phenotype input file contains multiple lines, each of with corresponds to each individuals in the same order as in the mean genotype file, and multiple columns, each of which corre-

sponds to each WCs. The phenotype input file can be prepared from functional phenotypic data by using the function `WaveQTL_preprocess`. This function lists (standardized, covariates corrected, and quantile transformed) WCs in the order from low resolution WC to high resolution WC (see Section 3 for details). The WCs can be ordered in different ways, but two additional input files containing filtering status of the WCs and information on which WCs share the same hyper parameter π should be prepared accordingly (see Section 4.1.3 and Section 4.1.4). The procedure described in Section 3 creates an example of phenotype file `WaveQTL/test/dsQTL/WCs.txt` of 1024 WCs on 70 individuals. The first three rows and the first six columns in `WaveQTL/test/dsQTL/WCs.txt` are:

```
-1.4147464 -1.8027431 -0.97699540 2.4499977 -0.7673774 -1.167875
 0.1256613 -1.4147464 -0.08964235 0.4241882  1.5197595  0.920823
 0.3470265 -0.6744898 -0.27188001 1.0364334 -0.4241882 -0.816375
```

4.1.3 Filtering status file format (optional)

One can filter out WCs from analyses by providing "filtering status file" with "-u" option. The filtering status file contains one line with multiple columns (the same number of columns in phenotype file), each of which indicates whether the corresponding WC in the phenotype file is filtered (0) or not (1). One can use the function `WaveQTL_preprocess` to generate filtering status of WCs based on low count threshold. The procedure described in Section 3 produces an example of the filtering status file `WaveQTL/test/dsQTL/use.txt` of 1024 WCs by using low count threshold of two. The first ten columns in `WaveQTL/test/dsQTL/use.txt` are:

```
1 1 1 1 1 1 1 1 0 0
```

where the 7th and 8th WCs are filtered out in analysis. By default, `WaveQTL` includes all WCs in analysis.

4.1.4 Group of WCs file format (optional)

One can provide information on which WCs share the same hyperparameter π in the model described in [4] with "-group" option. A group of WCs sharing the same π should be located next to each other in the phenotype file. The group of WCs file contains one line with multiple positive numbers, each of which indicates the start position of each group of WCs in the phenotype file. Suppose the group of WCs file contains

```
1 2 9 17
```

and the phenotype file contains 1024 columns. Then, `WaveQTL` partitions 1024 WCs into four groups. The first group has one WC that is in the first column of the phenotype file. The

second group has 7 WCs that locate from the second column to 8th column. WCs from 9th to 16th columns are in the third group and WCs from 17th to the end (1024th) columns are in the fourth group.

By default `WaveQTL` assumes that WCs in the phenotype file are ordered from low resolution WC to high resolution WC, and that WCs in the same scale share the same π, which is equivalent to providing the group of WCs file as follows for 1024 WCs.

```
1 2 3 5 9 17 33 65 129 257 513
```

To group WCs in multiple scales together (only for consecutive scales), one can use the function `generate_Group` we provided in `R/WaveQTL_preprocess_funcs.R` (see its description in the file for details).

```
> setwd("~/WaveQTL/R")
> source("WaveQTL_preprocess_funcs.R")
> group.info = generate_Group(numWCs = 1024, group.scale = c(0, 1, 4, 5))
> group.info
[1]  1  2  9 17
```

In this example, `generate_Group` generates "the group of WCs file" to partition 1024 WCs into four groups: the first group contains WC in the "0-th" scale (precisely speaking scaling coefficient); the second group consists of WCs from the "1st" (coarsest) to 3rd scales; WCs in the "4th" scale are in the third group and WCs from the "5th" to the last (10th) scales are in the fourth group.

4.2 Performing an analysis using WaveQTL

If you followed the procedure in Section 3, you can find `WCs.txt` and `use.txt` files in `WaveQTL/test/dsQTL` directory. Let's go to the `WaveQTL/test/dsQTL` directory.

```
cd ~/WaveQTL/test/dsQTL
```

4.2.1 Testing the null hypothesis H_0: functional phenotype at a given site is unassociated with all near-by SNPs

To test the null hypothesis, H_0: DNase-seq data at the site is unassociated with all near-by SNPs (we took "near-by" to mean "within 2kb of the site" in [4]), one can run `WaveQTL` with the option `-fph 3`, using for example

```
../../WaveQTL -gmode 1 -g ../../data/dsQTL/chr17.10160989.10162012.2kb.cis.geno
-p WCs.txt -u use.txt -o test -f 1024 -numPerm 30 -fph 3
```

The genotype file (`../../data/dsQTL/chr17.10160989.10162012.2kb.cis.geno`) should be provided with the option `-g`. For the mean genotype file format, `-gmode 1` is required in addition to `-g` (please read the BIMBAM manual `http://www.haplotype.org/download/bimbam-manual.pdf`). The phenotype file (`WCs.txt`) should be provided with the option `-p`. The name of output files (`test`) is specified using the option `-o` and the number of WCs (`1024`) is specified using the option `-f`. Optionally, one can provide filtering status file (`use.txt`) with the option `-u`. The `-numPerm` option can be used to specify the maximum number of permutations (30 in this example, 10000 by default). Please read the supplementary material of [4] to see details of our permutation procedure.

The command with `-fph 3` generates five output files inside an output folder in the current directory (`WaveQTL/test/dsQTL`). The `test.fph.pval.txt` file should be checked first to see if there is a strong evidence for association.

```
chr17.10161485
30
0
+0.03226
```

The first line shows SNP id that is most strongly associated with DNase-seq data at the site. The second and the third lines contain the number of permutations performed and the number of permuted data sets with test statistic greater than or equal to the observed test statistic when the permutation procedure stops, respectively (see the supplementary material of [4] for details of those numbers). Finally, the fourth line shows p-value obtained by permutation.

The rest four files allow for more detailed investigation of association with each of all near-by SNPs. Thus, once users believe there is an evidence for association in the site, they could look at those files for further investigation. The `test.fph.logLR.txt` file contains SNP id in the first column, log likelihood ratio test statistic for each SNP ($\log \hat{\Lambda}(y, g)$ in [4]) in the second column, and $\log_{10}$ Bayes Factor for each WCs ($\mathrm{BF}_{\mathrm{sl}}$ in [4]) in the remaining columns. The first three rows and the first six columns in `WaveQTL/test/dsQTL/output/test.fph.logLR.txt` are:

```
chr17.10159002 +0.33495 +0.03456 +0.14826 -0.00001 -0.00676
chr17.10159236 +6.63939 +0.43781 -0.05384 +0.43085 +0.09587
chr17.10160091 +6.15850 +0.60985 -0.03028 +0.30198 +0.10745
```

The `test.fph.pi.txt` file contains the maximum likelihood estimate $\hat{\pi}$ for each SNP (see [4] for details of definition). By default, `WaveQTL` partitions 1024 WCs into 11 groups, so `test.fph.pi.txt` file have 12 columns with SNP id in the first column and $\hat{\pi}_s$ for each scale s in the remaining 11 columns (from coarse to fine scales). The first three rows and the first six

columns in
`WaveQTL/test/dsQTL/output/test.fph.pi.txt` are:

```
chr17.10159002 +0.92179 +0.99699 +0.49610 +0.49200 +0.50032
chr17.10159236 +0.99989 +0.02672 +0.99951 +0.99917 +0.97613
chr17.10160091 +0.99999 +0.10329 +0.99926 +0.99823 +0.97052
```

The `test.fph.mean.txt` and `test.fph.var.txt` files contain SNP id in the first column and posterior mean (`test.fph.mean.txt`) and variance (`test.fph.var.txt`) of effect sizes (β_{sl} in [4]) of each SNP on each WCs in the remaining columns. The first three rows and the first six columns in `WaveQTL/test/dsQTL/output/test.fph.mean.txt` are:

```
chr17.10159002 -0.108228 -0.213471 -0.0404823 0.0363575 -0.0118591
chr17.10159236 -0.306942 0.00137141 -0.30402 0.148124 -0.0474979
chr17.10160091 -0.399600 -0.00697998 -0.262387 0.15933 -0.0459835
```

The first three rows and the first six columns in `WaveQTL/test/dsQTL/output/test.fph.var.txt` are:

```
chr17.10159002 0.00993409 0.0153632 0.00579517 0.00533707 0.0034537
chr17.10159236 0.0129849 0.000173951 0.0128463 0.00619634 0.00393106
chr17.10160091 0.0201208 0.000920335 0.012013 0.00734072 0.0043809
```

4.2.2 testing the null hypothesis H_0: functional phenotype at a given site is unassociated with a single SNP

To test for association with each of near-by SNPs, `WaveQTL` performs permutation and provides p-value for each of near-by SNPs, which can be accomplished with the option `-fph 2`, using for example

```
../../WaveQTL -gmode 1 -g ../../data/dsQTL/chr17.10160989.10162012.2kb.cis.geno
-p WCs.txt -u use.txt -o test2 -f 1024 -numPerm 30 -fph 2
```

The command with `-fph 2` generates five output files inside an output folder in the current directory (`WaveQTL/test/dsQTL`). The `test2.fph.pval.txt` file contains more information while the other four files are identical to those generated by the command with `-fph 1`. The `test2.fph.pval.txt` file contains multiple columns each of which corresponds to each of near-by SNPs. The first five columns in `WaveQTL/test/dsQTL/output/test2.fph.pval.txt` are:

```
chr17.10159002 chr17.10159236 chr17.10160091 chr17.10160195 chr17.10160499
25 30 30 30 30
10 1 0 0 3
0.41097 +0.06452 +0.03226 +0.03226 +0.12903
```

The fist line contains SNP id, and the second and third lines contains the number of permutations performed and the number of permuted data sets with test statistic greater than or equal to the observed test statistic when the permutation stops, for each SNP, respectively. Finally, the fourth line shows p-value for each SNP.

4.2.3 running WaveQTL without permutation

One can run `WaveQTL` without permutation using the command with the option `-fph 1` for example

```
../../WaveQTL -gmode 1 -g ../../data/dsQTL/chr17.10160989.10162012.2kb.cis.geno
-p WCs.txt -u use.txt -o test1 -f 1024 -fph 1
```

The command with the option `-fph 1` generate four output files (`test1.fph.logLR.txt`, `test1.fph.pi.txt`, `test1.fph.mean.txt`, and `test1.fph.var.txt`) that are identical to those files produced by the command with the option `-fph 2` or `-fph 3`.

5 Estimation of effect size in the data space

We tested for genetic association by using quantile transformed WCs to make tests robust to deviations from normality, but this quantile transform can make estimated effect sizes in the data space more difficult to interpret. Therefore, once one identifies sites with strong association by permutation with the option `-fph 2` or `-fph 3`, we advise rerunning `WaveQTL` on WCs without quantile transform using the option `-fph 1` to estimate effect sizes in the data space. One can obtain (standardized and covariates corrected) WCs from functional phenotype data using the function `WaveQTL_preprocess` with `no.QT = TRUE` option.

```
> setwd("~/WaveQTL/R")
> source("WaveQTL_preprocess_funcs.R")
> data.path = "../data/dsQTL/"
> pheno.dat = as.matrix(read.table(paste0(data.path,
+ "chr17.10160989.10162012.pheno.dat")))
> library.read.depth = scan(paste0(data.path, "library.read.depth.dat"))
> Covariates = as.matrix(read.table(paste0(data.path, "PC4.dat")))
```

```
> set.seed(1)
> meanR.thresh = 2
> res.noQT = WaveQTL_preprocess(Data = pheno.dat, library.read.depth =
+ library.read.depth , Covariates = Covariates, meanR.thresh = meanR.thresh, no.QT = TRUE)
> output.path = "../test/dsQTL/"
> write.table(res.noQT$WCs, file= paste0(output.path, "WCs.no.QT.txt"), row.names=FALSE,
+ col.names = FALSE, quote=FALSE)
```

Then, one can run `WaveQTL` on these only standardized and covariates corrected WCs using the option `-fph 1`.

```
cd ~/WaveQTL/test/dsQTL
../../WaveQTL -gmode 1 -g ../../data/dsQTL/chr17.10160989.10162012.2kb.cis.geno
-p WCs.no.QT.txt -u use.txt -o test.no.QT -f 1024 -fph 1
```

Now posterior mean and variance of effect sizes on the WCs are in `test.no.QT.fph.mean.txt` and `test.no.QT.fph.var.txt` files, and posterior mean and variance of effect sizes in data space can be obtained by transforming them back to the data space as follows. R scripts used here are in `R/get_effectSizeinDataSpace.R`.

```
> setwd("~/WaveQTL/R")
## Read Haar DWT matrix
> Wmat_1024 = read.table("../data/DWT/Wmat_1024",as.is = TRUE)
> W2mat_1024 = Wmat_1024*Wmat_1024
```

We provide the Haar Discrete Wavelet Transform (DWT) matrix on 512, 1024, and 2048 bp in `WaveQTL/data/DWT/`. Please read Section 5.1 to see how to produce those or other DWT matrices. Let's set path to files containing posterior mean and variance of effect sizes in wavelet space. We focus on 11th SNP in genotype file that is most strongly associated with DNase-seq data at the site (i.e., it has largest log likelihood ratio).

```
beta_mean_path= "../test/dsQTL/output/test.no.QT.fph.mean.txt"
beta_var_path = "../test/dsQTL/output/test.no.QT.fph.var.txt"
## We'll look at effect size of 11th SNP in genotype file
sel_geno_IX = 11
```

Read posterior mean in wavelet space and transform them back to the data space, yielding 1024 posterior mean at each locations.

```
beta_mean = as.numeric(read.table(beta_mean_path)[sel_geno_IX,2:1025])
beta_dataS = as.vector(-matrix(data=beta_mean, nr = 1, nc = 1024)%*%as.matrix(Wmat_1024))
```

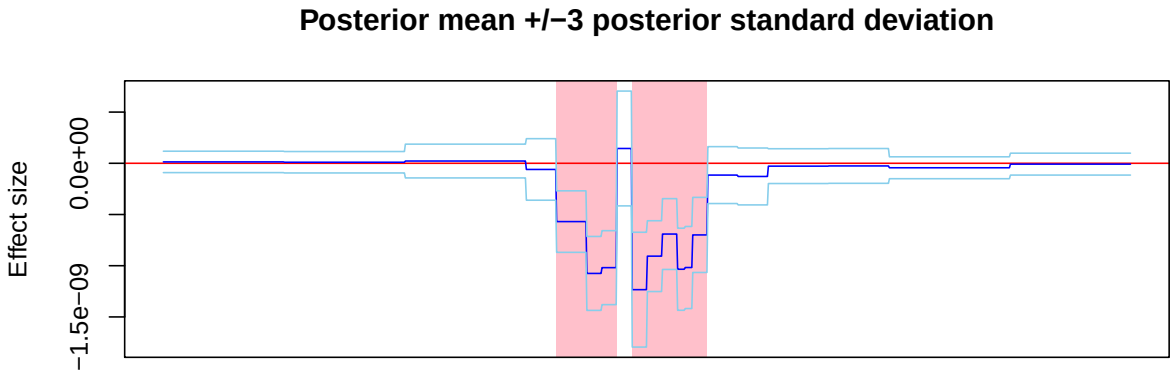

Figure 1: **Estimated effect sizes in data space**

```
length(beta_dataS)
[1] 1024
beta_dataS[1:6]
[1] 1.352119e-11 1.352119e-11 1.352119e-11 1.352119e-11 1.352119e-11
[6] 1.352119e-11
```

Read posterior variance in wavelet space, transform them back to the data space, and get standard deviation at each locations.

```
beta_var = as.numeric(read.table(beta_var_path)[sel_geno_IX,2:1025])
beta_var_dataS = as.vector(matrix(data=beta_var, nr=1, nc=1024)%*%as.matrix(W2mat_1024))
beta_sd_dataS = sqrt(beta_var_dataS)
length(beta_sd_dataS)
[1] 1024
beta_sd_dataS[1:6]
[1] 3.482833e-11 3.482833e-11 3.482833e-11 3.482833e-11 3.482833e-11
[6] 3.482833e-11
```

You can plot effect sizes in data space (see `R/get_effectSizeinDataSpace.R` for R script to produce this figure).

5.1 Generation of DWT matrix

We describe how to produce the Haar DWT matrices on 512, 1024, 2048 bp, provided in `WaveQTL/data/DWT/`, and other DWT matrix. R scripts used here are in `R/get_Wmat.R`.

```
> setwd("~/WaveQTL/R")
> source("WaveQTL_preprocess_funcs.R")
## Haar DWT for region of 512 bp
> InputD = diag(512)
> res = FWT(InputD)
> Wmat_512 = t(res$WCs)
## Haar DWT for region of 1024 bp
> InputD = diag(1024)
> res = FWT(InputD)
> Wmat_1024 = t(res$WCs)
## Haar DWT for region of 2048 bp
> InputD = diag(2048)
> res = FWT(InputD)
> Wmat_2048 = t(res$WCs)
## for different DWT for region of 512 bp
> InputD = diag(512)
> res = FWT(InputD, filter.number = 4, family ="DaubExPhase")
> Wmat_512.v2 = t(res$WCs)
```

The function `FWT` uses the Haar DWT with `filter.number`=1 and `family`="DaubExPhase" by default. Other DWT matrices can be produced by using `filter.number` and `family` in input arguments. They are arguments to the function `wd` in the R package `wavethresh` (for details, see their manual `http://cran.r-project.org/web/packages/wavethresh/wavethresh.pdf`).

6 Preparing genotype and functional phenotypic data for dsQTL analysis in [4]

You need to download genotype files (mean genotype format), DNase-seq data (as in hdf5 format), and mappability information (as in hdf5 format) from `http://eqtl.uchicago.edu/dsQTL_data/`. Here, we show how to generate `chr17.10160989.10162012.pheno.dat` and `chr17.10160989.10162012.2kb.cis.geno` in `WaveQTL/data/dsQTL/` directory from the downloaded files, and genotype and phenotype files for other sites can be easily produced by the same procedure (with different location information).

6.1 Preparing genotype data for a given site

The genotype files from `http://eqtl.uchicago.edu/dsQTL_data/` contain mean genotypes on 209 individuals. The list of 70 individuals included in the analysis of [4] is

WaveQTL/data/Shim_2014_etc/DNaseI.individuals.oneline.txt. Thus, first you need to exclude those individuals who are not in the list from the mean genotype files. One can use perl script in WaveQTL/scripts/take_70ind.pl (you need to change paths in the script). Then, for a given site (e.g., chr17.10160989.10162012), you need to find a set of SNPs within 2kb of the site (e.g., from 10158989 to 10164012 on chr17) from chr17.YRI.snpdata.txt file and extract genotypes only on those SNPs from YRI.70.mean.genotype.txt. One can use python script in WaveQTL/scripts/extract.py (you need to change a path in the script).

```
cd WaveQTL/scripts
./extract.py 17 10158989 10164012
```

Then, it creates chr17.10158989.10164012.YRI.mean.genotype.txt and chr17.10158989.10164012.YRI.snpdata.txt in the current working directory, which are the same as chr17.10160989.10162012.2kb.cis.noMAFfilter.geno and chr17.10160989.10162012.2kb.cis.noMAFfilter.info provided in the /WaveQTL/data/dsQTL/ directory. chr17.10160989.10162012.2kb.cis.noMAFfilter.geno contains genotypes at 31 SNPs within 2kb of the site and chr17.10160989.10162012.2kb.cis.noMAFfilter.info contains information from chr17.YRI.snpdata.txt for the 31 SNPs (those information is used when we prepare phenotype data. See Section 6.2). We excluded SNPs with MAF < 0.05 as [1] did in their analysis. This can be accomplished by

```
> in_path = "~/WaveQTL/data/dsQTL/chr17.10160989.10162012.2kb.cis.noMAFfilter.geno"
> out_path =  "~/WaveQTL/data/dsQTL/chr17.10160989.10162012.2kb.cis.geno"
> geno_in = read.table(in_path, as.is = TRUE)
> GenoTypes <- as.matrix(geno_in[,-(1:3)])
> MAF_our = apply(GenoTypes,1, sum)/(70*2)
> wh_sel = which((MAF_our > 0.05) & (MAF_our < 0.95))
> geno_out = geno_in[wh_sel,]
> write.table(geno_out, file = out_path, quote= FALSE, row.names = FALSE, col.names = FALSE)
```

6.2 Preparing functional phenotypic data for a given site

For the given site (chr17.10160989.10162012), WaveQTL/R/prepare_functional_phenotype.R shows how to produce the functional phenotypic data at the site (WaveQTL/data/dsQTL/chr17.10160989.10162012.pheno.dat) from the downloaded files, containing

- how to read DNase-seq data at the site on 70 individuals from files in hdf5 format;
- how to read mappability information at the site from file in hdf5 format;

- how to mask 5bp surrounding any SNP (i.e., the SNP position and 2bp on either side) to eliminate biases stemming from DNase I sequence preference (see the supplementary material of [1] for details);
- how to combine DNase-seq data from two strands while taking mappability into account.

To mask 5bp surrounding any SNP at the site, we need positions of SNPs that are located at the site. Thus, we used `WaveQTL/data/dsQTL/chr17.10160989.10162012.2kb.cis.noMAFfilter.info` which contains information (including positions) on SNPs within 2kb of the site (extracted from `chr17.YRI.snpdata.txt.gz` file).

References

[1] Jacob F Degner, Athma a Pai, Roger Pique-Regi, Jean-Baptiste Veyrieras, Daniel J Gaffney, Joseph K Pickrell, Sherryl De Leon, Katelyn Michelini, Noah Lewellen, Gregory E Crawford, Matthew Stephens, Yoav Gilad, and Jonathan K Pritchard. DNase I sensitivity QTLs are a major determinant of human expression variation. *Nature*, 482(7385):390–4, February 2012.

[2] Bryan N Howie, Peter Donnelly, and Jonathan Marchini. A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. *PLoS genetics*, 5(6):e1000529, June 2009.

[3] Bertrand Servin and Matthew Stephens. Imputation-based analysis of association studies: candidate regions and quantitative traits. *PLoS genetics*, 3(7):e114, July 2007.

[4] H. Shim and M. Stephens. Wavelet-based genetic association analysis of functional phenotypes arising from high-throughput sequencing assays. *ArXiv e-prints*, July 2013.