**GWAS code – using Pyseer**

This document shows the commands used to run Pyseer for each type of test.

**Logistic variant test**

*python scripts/phylogeny\_distance.py --lmm [whole tree] > phylogeny\_dists\_whole.tsv* **#Generates the phylogenetic matrix**

*pyseer --lmm --phenotypes known\_nonsyn\_phenotype\_res.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz --similarity phylogeny\_dists\_whole.tsv --output-patterns vcf-patterns\_whole.txt --cpu 8 > dapto\_genes\_whole.txt*

*python scripts/count\_patterns.py vcf-patterns\_whole.txt* **#Printing threshold**

*cat <(head -1 dapto\_genes\_whole.txt) <(awk '$4<2.07E-06 {print $0}' dapto\_genes\_whole.txt) > sig\_dap\_whole.txt* **#Print significant variants**

**Logistic variant test with elastic net regression (alpha=1)**

*pyseer --phenotypes known\_nonsyn\_phenotype\_res.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz --wg enet –save-vars output/ma\_snps –save-model dapto\_vars\_whole.lasso --cpu 4 –alpha 1 > selected\_vars\_whole.txt*

**Linear variant test**

*pyseer --lmm --phenotypes pheno\_linear.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz --similarity phylogeny\_dists\_whole.tsv --output-patterns vcf-patterns\_linear\_var\_whole.txt --cpu 8 > lin\_dapto\_genes\_whole.txt*

*python scripts/count\_patterns.py vcf-patterns\_linear\_var\_whole.txt*

*cat <(head -1 lin\_dapto\_genes\_whole.txt) <(awk '$4<1.29E-06 {print $0}' lin\_dapto\_genes\_whole.txt) > sig\_lin\_var\_whole.txt*

**Linear variant test with elastic net regression (alpha=1)**

*pyseer --phenotypes pheno\_linear.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz --wg enet –save-vars output/lin\_dapto\_vars –save-model lin\_dapto\_vars\_whole.lasso --cpu 4 –alpha 1 > lin\_selected\_vars\_whole.txt*

**Logistic burden test**

* Demonstrated here with a binary phenotype file and the VCF file filtered by mutation effect and allele frequency < 0.1

*Pyseer –lmm –phenotypes known\_nonsyn\_phenotype\_res.txt –vcf efm\_dap\_dataset\_v1.snippy.fun.rf.aff\_0.1.vcf.gz –burden efm\_genes.csv –similarity phylogeny\_dists\_whole.tsv –output-patterns filter\_patterns\_log\_burden.txt –cpu8 > log\_burden\_filter.txt*

**Linear burden test**

* Demonstrated here using the whole genome

*pyseer --lmm --phenotypes pheno\_linear.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz –burden efm\_genes.csv --similarity phylogeny\_dists\_whole.tsv --output-patterns vcf-patterns\_linear\_var\_burden\_whole.txt --cpu 8 > lin\_dapto\_genes\_whole.txt*

*python scripts/count\_patterns.py vcf-patterns\_linear\_var\_burden\_whole.txt*

*cat <(head -1 lin\_dapto\_genes\_whole.txt) <(awk '$4<2.05E-05 {print $0}' lin\_dapto\_genes\_whole.txt) > sig\_lin\_var\_whole.txt*

**Logistic burden test with elastic net regression (alpha=1)**

* Demonstrated here using the whole genome

*pyseer --phenotypes known\_nonsyn\_phenotype\_res.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz –burden efm\_genes.csv --wg enet –save-vars output/log\_burden\_whole –save-model log\_selected\_burden\_whole.lasso --cpu 4 –alpha 1 > log\_selected\_burden\_whole.txt*

**Linear burden test with elastic net regression (alpha=1)**

* Demonstrated here using the whole genome

*pyseer --phenotypes pheno\_linear.txt --vcf efm\_dap\_dataset\_v1.snippy.vcf.gz –burden efm\_genes.csv --wg enet –save-vars lin\_burden\_whole –save-model lin\_selected\_burden\_whole.lasso --cpu 4 –alpha 1 > lin\_selected\_burden\_whole.txt*