GTEx eQTL analysis notes

Gao Wang

Last updated: August 17, 2015

Chapter 1

General Information

1.1 Release V(?)

- GTEx raw and normalized expression and PEER factors data: ftp://ftp.broadinstitute.org/GTEx_Analysis_2014-06-13/expressionCovariateSandbox/
  - There is a README file in this folder that contains description of data
- GTEx summary statistics formatted to eqtlbma input, by Sarah: [Midway] /project/mstephens/gtex/preprocessing/june2014/sumstats
- GTEx summary statistics from eqtlbma by Sarah: [Midway] /project/mstephens/gtex/analysis/june2014/sumstats
- GTEx genotypes /project/mstephens/gtex/preprocessing/june2014/omni451.vcf.gz
- GTEx expression level [Midway] /project/mstephens/gtex/preprocessing/june2014/GTEx_Analysis_2014-06-13_expression_provisional/

1.2 Release V6

Input and summary statistics from MatrixEQTL: [PPS] /mnt/gluster/data/external_private_supp/GTEX_6_13/ncbi/44473/gtex/exchange/GTEx_phs000424/exchange/analysis_releases/GTEx_Analysis_2015-01-12

1.2.1 Why covariates not removed from expression values

"One reason for this is because this is how Matrix eQTL accepts input data. I realize that corrected expression values could be valuable to people and would certainly be willing to provide them, but I do not have a method. Although PEER does output a residuals file, I can only get PEER to run robustly by filtering low expressed genes and opting not to use explicit covariates as

input to PEER. The solution would be to use the transformation that Matrix eQTL uses internally. ..." [David DeLuca, Broad institute]

Chapter 2

GTEx Summary Statistics

Computations are done on the Midway cluster. `GTEx.bashrc` sets up the proper shell environment for this computational workflow:

```
source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
```

2.1 Formatting GTEx Summary Statistics V6

2.1.1 Convert tissue specific results to matrices

For ease of storage / access / numeric operations I convert the summary statistics from plain text file to HDF5 format.

ss.gz to HDF5 conversion

```
cd $AnlyDir/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs
for i in `ls *.gz`; do
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py ss_to_h5 \
     '"$i"' --action convert --output . --message "MatrixEQTL summary statistics of GTEx Release 2015.04.15"' |\
sbatch -J ss2h5_$i -o $LogDir/ss2h5_$i.o%j -e $LogDir/ss2h5_$i.e%j
done
```

Note

I ended up having to perform the entire data conversion on SSD on my desktop computer using. The `pytable` version of HDF5 implementation takes 2h to convert each tissue but on Midway when HDF5 file size grows to over 1GB the disk I/O is unacceptable. There must be tweaks but not looking into it for the moment. Implementations with `h5py` for the same purpose is way faster in data conversion (30min each file, performance on Midway is also acceptable) but the resulting data file size is 1.4 times of `pytables` implementation, even both set to zlib9 compression – thus I stick with `pytable`. Once converted, it takes almost zero seconds to query data from the HDF5 files.

2.1.2 Merge data across tissues

HDF5 merge, in 50 batches

```
source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
zcat $InputDir/tss_coords.bed.gz | cut -f4 | split -l 1127 -d - $TmpDir/tss_coords.bed.split.
k=0
for i in `ls $TmpDir/tss_coords.bed.split.*`; do
let k+=1
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py ss_to_h5 \
     $DataDir/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs/*.h5 \
     --action merge --output $DataDir/MatrixEQTLSumStats/MatrixEQTLSumStats.'"$k"' \
     --message "MatrixEQTL summary statistics of GTEx Release 2015.04.15" \
     --gene-list '"$i"'' |\
sbatch -J h5merge_"$k" -o $LogDir/h5merge_"$k".o%j -e $LogDir/h5merge_"$k".e%j
done
```

It takes on average 3 hours per batch to complete. Merged number of genes per batch see `log/2015-06-18-merger-log.txt`. There are a total of 38933 genes.

Note

I was suspicious to see only 38933 genes in v6. I decide to check against the input data as well as the intermediate HDF5 files I generated

BASH

```
for i in `ls $DataDir/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs/*.cis.eqtl.gz`; do
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     zcat '"$i"' | cut -f2 | sort -u | gzip --best > '"$i"'.uniqgenes.gz' |\
sbatch -J count_gene -o $LogDir/count_gene.o%j -e $LogDir/count_gene.e%j
done
for i in `ls $DataDir/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs/*.h5`; do
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py ss_to_h5 '"$i"' --action summary --output stdout | gzip --best > '"$i"'.h5genes.gz' |\
sbatch -J count_gene_h5 -o $LogDir/count_gene_h5.o%j -e $LogDir/count_gene_h5.e%j
done
```

Counting (`zcat *.xx.gz | sort -u | wc -l`) and comparing results from both sources, I found there are indeed only 38933 genes in the output. So we are good.

Now merging the batches into one large HDF5 file

```
python $SrcDir/analysis_admin.py ss_to_h5 $DataDir/MatrixEQTLSumStats/*.h5 --action cat --output $DataDir/MatrixEQT\
LSumStats.h5
```

2.1.3 Select the "best" gene-snp pair

```
python $SrcDir/analysis_admin.py ss_to_h5 $DataDir/MatrixEQTLSumStats.h5 --action max --output $DataDir/MatrixEQTLS\
umStats.Portable.h5
```

2.1.4 Select some "null" gene-snp pairs

For each gene I select up to 3 null gene-snp pairs, for use of calibrating Sarah's model

```
python $SrcDir/analysis_admin.py ss_to_h5 $DataDir/MatrixEQTLSumStats.h5 --action null --output $DataDir/MatrixEQTL\
SumStats.Portable.h5 --gene-list <(zcat $DBDir/snp-gene-pairs/GTExV6Genes.gz)
```

2.2 Creating R Interface for the New Format

Since the release did not use rsID to name SNPs I create a database to match SNP ID with dbSNP names, to use with R to extract proper SNP-gene by rsID.

```
wget ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606_b144_GRCh37p13/VCF/All_20150605.vcf.gz
```

Build an annotation database matching SNP ID to rs ID as well as the cis-genes involved of up to $\pm 100,000$bp.

ID annotations, in 50 batches

```
source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
zcat $InputDir/snp_coords.bed.gz | split -l 205953 -d - $TmpDir/snp_coords.bed.split.
k=0
for i in `ls $TmpDir/snp_coords.bed.split.*`; do
let k+=1
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     cat '"$i"' | bash $SrcDir/FindrsIDcisGene.sh \
     $DBDir/dbSNP/All_20150605.vcf.gz $InputDir/tss_coords.bed.gz 100000 | gzip > \
     $DBDir/snp-gene-pairs.'"$k"'.gz' |\
sbatch -J snpid_"$k" -o $LogDir/snpid_"$k".o%j -e $LogDir/snpid_"$k".e%j
done
```

It takes on average 25 hours per batch to complete. There are 9794339 out of 10297646 (95.1%) SNPs with rsID in latest dbSNP (build 144). To merge them to one file and build a sqlite database:

```
zcat $DBDir/snp-gene-pairs.*.gz | gzip --best > $DBDir/snp-gene-pairs.gz
cd $DBDir; mkdir snp-gene-pairs; mv snp-gene-pairs.* snp-gene-pairs; cd snp-gene-pairs
wsqlite snp-gene.sqlite3 -i snp-gene-pairs.gz --as dbsnp144 -d '\t' --header coord rsid cisgenes
wsqlite snp-gene.sqlite3 "create index rsid_index on dbsnp144 (rsid)"
wsqlite snp-gene.sqlite3 "create index coord_index on dbsnp144 (coord)"
```

2.2.1 Run queries

Now we have the data `*.h5` and meta-data `*.sqlite` in place. I have made a separate note on how to use this data-set. See `SumstatsDB.pdf` in this repo.

2.3 Reproducing GTEx V6 Cis-eQTL Analysis

The idea now is to figure out how the `matrixEQTL` analysis is done exactly in v6 so that `eqtlbma` will be done on the same basis, i.e., input files and parameters. For this purpose I make a toy dataset for a randomly chosen tissue from GTEx consisting of five files for input to `matrixEQTL`:

genotype `SNP.txt`, expression `GE.txt`, a file `Covariates.txt` of covariates and files `geneloc.txt` and `snpsloc.txt` with gene and SNP location information.

```
Tissue=Adipose_Subcutaneous
```

2.3.1 Covariates

There are 40 covariates: "C1,C2,C3,InferredCov1, InferredCov2, ... InferredCov35, gender, Platform". Not all are included in every analysis – inclusion depends on sample size. See the `README` for details.

```
tar -zxvf GTEx_Analysis_2015-01-12_eQTLInputFiles_covariates.tar.gz "$Tissue"_Analysis.covariates.txt
wc -l "$Tissue"_Analysis.covariates.txt
cut -f1 "$Tissue"_Analysis.covariates.txt | tr "\n" ", "
mv "$Tissue"_Analysis.covariates.txt $TmpDir/Covariates.txt
```

2.3.2 SNP data

- **SNP matrix**

Input genotype data are large. Here I just take the first 500 SNPs from the tissue data for verification purpose.

```
tar -zxvf GTEx_Analysis_2015-01-12_eQTLInputFiles_snpMatrices.tar.gz "$Tissue"_Analysis.snps.txt -O |\
head -501 > $TmpDir/SNP.txt
```

Warning

SNP data released has a lot of imputed data. Cautions are to be taken when comparing with previous `eqtlbma` analysis (have to make sure the input data are the same).

- **SNP location**

This is required input for `matrixEQTL` but is not provided. Need to create it from SNP ID's.

```
echo -e "snp\tchr\tpos" > $TmpDir/snpsloc.txt
cut $TmpDir/SNP.txt -f1 | tail -n+2 | awk -F"_" '{print $0"\t""chr"$1"\t"$2}' >> $TmpDir/snpsloc.txt
```

2.3.3 Expression data

To focus only on genes paired with the 500 SNPs I get a list of genes via `bedtools closest` and only extract expression data for these genes.

Create `bedtools` input and run `bedtools` to search for genes within 1000bp range to the SNPs of interest.

```
tail -n+2 GTEx_Analysis_2015-01-12_eQTLInputFiles_genePositions.txt |\
awk '{print $2"\t"$3"\t"$4+1"\t"$1"\t"1"\t""+"}' > $TmpDir/geneloc.bed
cut $TmpDir/SNP.txt -f1 | tail -n+2 | awk -F"_" '{print $1"\t"$2"\t"$2+1"\t"$0"\t"1"\t""+"}' > $TmpDir/snpsloc.bed
bedtools closest -a $TmpDir/snpsloc.bed -b $TmpDir/geneloc.bed |\
awk '{if (($8-$2)<1000 && ($8-$2)>-1000) print $0}' > $TmpDir/gs-pairs.bed
```

Extract expression data for selected genes

```
cut -f10 $TmpDir/gs-pairs.bed | sort -u > $TmpDir/1.txt # 22 genes found
tar -zxvf GTEx_Analysis_2015-01-12_eQTLInputFiles_geneLevelNormalizedExpressionMatrices.tar.gz \
"$Tissue"_Analysis.expr.txt -O | head -1 > $TmpDir/GE.txt
tar -zxvf GTEx_Analysis_2015-01-12_eQTLInputFiles_geneLevelNormalizedExpressionMatrices.tar.gz \
"$Tissue"_Analysis.expr.txt -O | grep -wf $TmpDir/1.txt >> $TmpDir/GE.txt # 11 genes here
```

It ends up with 11 genes for the 500 SNPs

```
cut -f 1,2 $TmpDir/GE.txt
```

2.3.4 Gene location data

Location data is extracted for the selected genes.

```
cut -f1 $TmpDir/GE.txt | tail -n+2 > $TmpDir/1.txt
grep -wf $TmpDir/1.txt GTEx_Analysis_2015-01-12_eQTLInputFiles_genePositions.txt > $TmpDir/2.txt
head -1 GTEx_Analysis_2015-01-12_eQTLInputFiles_genePositions.txt | cat - $TmpDir/2.txt > $TmpDir/geneloc.txt
```

2.3.5 Find overlapping gene-snp pairs between this toy set and GTEx v6 summary statistics

BASH

```
for i in `cut -f1 $TmpDir/geneloc.txt | tail -n+2`; do
    echo $i
    python analysis_admin.py sumstat_query $InputDir/"$Tissue"_Analysis.h5 -g $i -s NULL > $TmpDir/1.txt
    if [[ -s $TmpDir/1.txt ]]; then
       grep -wf $TmpDir/1.txt $TmpDir/snpsloc.txt | cut -f1 > $TmpDir/"$i"_snps_to_check.txt
    fi
done
```

2.3.6 Run `matrixEQTL` on this toy set

Once the input is prepared it is straightforward to just download the example R script (below, also committed to git repo) and run it, just remember to set the correct data path and change emitted p-value threshold to 1 (`pvOutputThreshold=1`).

http://www.bios.unc.edu/research/genomic_software/Matrix_eQTL/R.html

To obtain value for one gene-snp pair:

R

```
source('MatrixEQTL.R')
ss = '1_714019_A_G_b37'
gg = 'ENSG00000237491.4'
me$all$eqtls[which(me$all$eqtls$snps == ss & me$all$eqtls$gene == gg ),]
```

OUTPUT

```
               snps              gene statistic       pvalue          FDR
241 1_714019_A_G_b37 ENSG00000237491.4  5.392883 1.576441e-07 3.597687e-06
         beta
241 0.7902205
```

and to verify value for gene-snp pairs:

```
echo $Tissue
```

R

```
source('/project/mstephens/gtex/scripts/SumstatQuery.R')
dat <- GetFlatSS('ENSG00000237491.4', '/project/mstephens/gtex/analysis/april2015/eqtl_data/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs/Adipose_Subcutaneous_Analysis.h5')
print(dat["1_714019_A_G_b37", ])
```

OUTPUT

```
        beta       t-stat      p-value
7.902205e-01 5.392883e+00 1.576441e-07
```

I verified a few other gene-SNP pairs, all agree. So I can claim I've figured out the input data and parameters used for the GTEx analysis!

2.4 Update `eqtlbma` Analysis for Some Gene-snp Pairs and Verify with GTEx `matrixEQTL`

Here is the command for generating summary statistics via `eqtlbma_bf`:

BASH

```
Analysis=join
Tissue=Test_Tissue
echo -e "$Tissue\tSNP.txt.gz" > list_geno.txt
echo -e "$Tissue\tGE.txt" > list_expl.txt
echo -e "$Tissue\tCovariates.txt" > list_covar.txt
eqtlbma_bf --geno list_geno.txt --exp list_expl.txt --covar list_covar.txt \
           --scoord snpsloc.bed  --gcoord geneloc.bed \
           --out ToyEQTLBma --analys $Analysis --outss --bfs sin \
           --gridL /project/mstephens/gtex/analysis/june2014/types_v1/grid_phi2_oma2_general.txt.gz \
           --gridS /project/mstephens/gtex/analysis/june2014/types_v1/grid_phi2_oma2_with-configs.txt.gz \
           --error uvlr --thread 4
```

Check a couple of output from `eqtlbma_bf` with `matrixEQTL`

```
zcat ToyEQTLBma_sumstats_Test_Tissue.txt.gz | head | cut -f1,2,7,9
```

OUTPUT

```
gene    snp     betahat.geno    betapval.geno
ENSG00000177757.1       1_662622_G_A_b37        -2.588170e-04   9.985439e-01
ENSG00000177757.1       1_676127_C_T_b37        7.151412e-02    7.744839e-01
ENSG00000177757.1       1_691541_AT_A_b37       -9.077870e-02   4.383949e-01
```

```
ENSG00000177757.1       1_693625_T_C_b37        -5.757815e-02   7.415218e-01
ENSG00000177757.1       1_693731_A_G_b37        -1.080375e-02   9.255429e-01
ENSG00000177757.1       1_697411_G_GA_b37       5.502301e-01    2.327584e-02
ENSG00000177757.1       1_701835_T_C_b37        7.243649e-01    3.168885e-03
ENSG00000177757.1       1_704367_T_C_b37        1.898470e-01    3.756234e-01
ENSG00000177757.1       1_705882_G_A_b37        -1.595642e-01   3.141656e-01
```

Check with `MatrixEQTL`

R

```r
gg = 'ENSG00000177757.1'
for (ss in c('1_662622_G_A_b37', '1_676127_C_T_b37', '1_691541_AT_A_b37', '1_693625_T_C_b37', '1_693731_A_G_b37')) print(me$all$eqtl
s[which(me$all$eqtls$snps == ss & me$all$eqtls$gene == gg ),][,c(2,1,6,4)])
```

OUTPUT

```
                  gene             snps          beta    pvalue
5493 ENSG00000177757.1 1_662622_G_A_b37 -0.000258817 0.9985439
                  gene             snps       beta    pvalue
4593 ENSG00000177757.1 1_676127_C_T_b37 0.07151412 0.7744839
                  gene              snps       beta    pvalue
3047 ENSG00000177757.1 1_691541_AT_A_b37 -0.0907787 0.4383949
                  gene             snps        beta    pvalue
4454 ENSG00000177757.1 1_693625_T_C_b37 -0.05757815 0.7415218
                  gene             snps        beta    pvalue
5200 ENSG00000177757.1 1_693731_A_G_b37 -0.01080375 0.9255429
```

Thus `eqtlbma_bf` agrees with `MatrixEQTL` when input agree. Good! We'll have to update the analysis using new input data.

Chapter 3

Computational Steps for Calculating Summary Statistics Under Quasi-Poisson Model

3.1 First Attempt

Here is the command which breaks the data into 2,000 batches and perform analysis under quasi-Poisson model.

2000 batches analysis

```
nBatches=2000
Model=quasipoisson
for i in `seq $nBatches`; do
echo '#!/bin/bash
     source $HOME/GIT/type-model/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py eqtlbma_batch \
     -g $InputDir/tss_coords.bed.gz \
     -s $InputDir/snp_coords.bed.gz \
     -n '"$nBatches"' -b '"$i"' -e ~/software/bin/eqtlbma_bf \
     -a $ConfDir/eqtlbma.'"$Model"'.txt' |\
sbatch -J eqtlbma_$Model_$nBatches_$i -o $LogDir/eqtlbma_$Model_$nBatches_$i.o%j --mem-per-cpu=10000
done
```

3.1.1 Result

Computation is heavy and not completed (see next section). Here taking a sample batch out of the 2000 batches above, the following is summarized from `eqtlbma` log file

- Input batch has 28 genes and 113 SNPs
- Genotype and expression of 44 tissues are loaded, vary in sample size, from >70 to 450.
- Tissue specific genes "to keep" vary, but are around 26, 27, 28.
- Number of SNPs (from `omni451.vcf.gz`) drop greatly after discarding missing values and filter by MAF < 0.05. There are only 10 out of 113 SNPs left.

- Test for each gene-snp pair are done for every batch. There are only a handful such pairs for 2000 batches.

When 5000 batches are created in another test run, some batch may end up fail after loading the data because there will be no qualified SNP left for that batch, from our sample data.

3.1.2 A note on computational time

There will be a 20min "overhead" for each job (loading / extracting data) regardless of per batch size, i.e., we will "waste" 0.3N CPU hours for running in N batches. It is thus best to find a batch neither too small nor to large (if we do not optimize the way data is loaded).

In my test job when only 10 SNPs and <28 genes are analyzed in effect, it takes only 1.38 sec to complete for normal model, evaluating for each tissue about 200 gene-SNP pairs.

However for the same job under quasi-Poission it takes **significantly** computation time. For example 5 hours (10:30 am to 3:30 pm) have passed for the 10 SNPs and 28 genes. Still the progress bar is at 3.75%. I have tried `--thread` option of `eqtlbma_bf` and reserve corresponding number of nodes per job on Midway. Somehow the computational node still only use one thread for a job. May have to figure it out but I doubt it's going to help now that we can easily create arbitrary batches.

Here is the `eqtlbma_bf` command in action for this one batch. For this batch `genes_2000_85_112.bed.gz` has 28 lines and `snps_2000_85_112.txt.gz` has 113 lines.

Quasi-Poisson eqtlbma_bf command

```
/home/gaow/software/bin/eqtlbma_bf --geno /project/mstephens/gtex/analysis/june2014/types_v1/list_geno.txt \
                                   --exp /project/mstephens/gtex/analysis/june2014/types_v1/list_explevels.txt \
                                   --out eqtlbma_bf_quasipoisson_2000_85_112/eqtlbma_bf_quasipoisson_2000_85_112 \
                                   --anchor TSS --cis 100000 --lik quasipoisson --analys join --outss --maf 0.05 \
                                   --gridL /project/mstephens/gtex/analysis/june2014/types_v1/grid_phi2_oma2_general.txt.gz \
                                   --gridS /project/mstephens/gtex/analysis/june2014/types_v1/grid_phi2_oma2_with-configs.txt.gz \
                                   --bfs sin --error uvlr --fiterr 0.5 -v 1 \
                                   --gcoord genes_2000_85_112.bed.gz --snp snps_2000_85_112.txt.gz
```

3.2 What Next?

After discussion with Matthew and William on June 22, 2015 we decide to pause this analysis (from experience of Pritchard's lab the gain from using GLM does not worth it's computational cost).

Chapter 4

Analyzing GTEx Using `eqtlbma`

4.1 Pre-processing

4.1.1 Extract input data from archive

BASH

```
for i in GTEx_Analysis_2015-01-12_eQTLInputFiles_covariates.tar.gz GTEx_Analysis_2015-01-12_eQTLInputFiles_snpMatrices.tar.gz GTEx_A\
nalysis_2015-01-12_eQTLInputFiles_geneLevelNormalizedExpressionMatrices.tar.gz GTEx_Analysis_2015-01-12_eQTLInputFiles_snpMatricesSu\
pplement.tar.gz; do
for j in `tar -tf $i`; do
echo $i $j
mkdir -p $(basename $i .tar.gz)
echo -e '#!/bin/bash\ntar -zxvf '"$i"' '"$j"' -O | gzip --best > '"$(basename $i .tar.gz)"'/'"$j"'.gz\necho complete!' | sbatch -J E\
xtract."$i"."$j" -o $LogDir/Extract."$i"."$j".o%j
done
done
```

4.1.2 Gene TSS coordinates file

```
GetTSSCoords "$DataPrefix"eQTLInputFiles_genePositions.txt.gz $InputDir/tss_coords.bed.gz
```

4.1.3 SNP coordinates file

Unfortunately the release does not come with a list of SNPs (union) involved. Getting such a list is quite a heavy duty. Takes 10min to extract ID's in parallel and 50min to concatenate them into a unique list in bed format:

```
GetSNPCoords "$DataPrefix"eQTLInputFiles_snpMatrices $InputDir/snp_coords.bed.gz 1
GetSNPCoords "$DataPrefix"eQTLInputFiles_snpMatrices $InputDir/snp_coords.bed.gz 2
```

Note

In Sarah's analysis based on an earlier version of data, there are 55,993 genes and 6,856,776 SNPs provided in gene/snp lists. In the v6 release there are 56,318 genes and 10,297,646 SNPs listed; although the sumstats of v6 only has ~39,000 genes.

4.1.4 Input file lists

BASH

```
rm -rf $InputDir/list_geno.txt
DFolder="$DataPrefix"eQTLInputFiles_snpMatrices
for i in `ls $DFolder`; do
    j=`basename $i _Analysis.snps.txt.gz`
    echo -e "$j\t$DFolder/$i" >> $InputDir/list_geno.txt
done
#
rm -rf $InputDir/list_expr.txt
DFolder="$DataPrefix"eQTLInputFiles_geneLevelNormalizedExpressionMatrices
for i in `ls $DFolder`; do
    j=`basename $i _Analysis.expr.txt.gz`
    echo -e "$j\t$DFolder/$i" >> $InputDir/list_expr.txt
done
#
rm -rf $InputDir/list_covar.txt
DFolder="$DataPrefix"eQTLInputFiles_covariates
for i in `ls $DFolder`; do
    j=`basename $i _Analysis.covariates.txt.gz`
    echo -e "$j\t$DFolder/$i" >> $InputDir/list_covar.txt
done
```

4.2 The Configuration Model

4.2.1 Run `eqtlbma_bf` analysis in batches

`analysis_admin eqtlbma_batch` generates batches on the fly and perform `eqtlbma_bf` analysis in (embarrassing) parallel fashion on batches of genes. To use it,

```
python analysis_admin.py eqtlbma_batch -h
```

eqtlbma_batch

```
usage: analysis_admin.py eqtlbma_batch [-h] -g GENE_COORDS -s SNP_COORDS
                                       [-w N] [-n N_BATCHES] [-b BATCH_ID] -a
                                       ARGS_FILE [--seed N] -e EQTLBMA_PATH
                                       [--dry-run] [--clean]

optional arguments:
  -h, --help       show this help message and exit
  -g GENE_COORDS   Gene (TSS) coordinate file, in bed.gz format. (default:
                   None)
  -s SNP_COORDS    SNP coordinate file, in bed.gz format. (default: None)
  -w N             Window size. (default: 100000)
  -n N_BATCHES     Total number of batches. (default: 1000)
  -b BATCH_ID      Execute the i-th batch. Program will quit if invalid ID is
                   provided. (default: 1)
  -a ARGS_FILE     Path to file containing additional eqtlbma command
                   arguments. (default: None)
  --seed N         If specified, a random number will be generated using (N +
                   batch_id) as seed, and the eqtlbma command will be appended
```

```
                  a "--seed" argument with the number generated here.
                  (default: None)
  -e EQTLBMA_PATH  Path to an eqtlbma_* executable. (default: None)
  --dry-run        Only generate and save the batch data & commands without
                   performing analysis. (default: False)
  --clean          Remove batch gene / snp coords file upon finishing the
                   analysis. (default: False)
```

Here is the command which breaks the data into 100 batches and perform analysis

100 batches analysis

```
nBatches=100
Model=normal
for i in `seq $nBatches`; do
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py eqtlbma_batch \
     -g $InputDir/tss_coords.bed.gz \
     -s $InputDir/snp_coords.bed.gz \
     -n '"$nBatches"' -b '"$i"' -w 100000 \
     --seed 10086 -e ~/software/bin/eqtlbma_bf \
     -a $ConfDir/eqtlbma.'"$Model"'.txt' |\
sbatch -J eqtlbma_"$Model"_"$nBatches"_"$i" -e $LogDir/eqtlbma_"$Model"_"$nBatches"_"$i".e%j \
       -o $LogDir/eqtlbma_"$Model"_"$nBatches"_"$i".o%j --mem-per-cpu=10000 --time=36:00:00
sleep 1
done
```

4.2.2 Failure, Tue Jun 16 08:14:48 CDT 2015

None of the jobs submitted was complete. Here is summary of completion from progress bar (percent of completion), also see `2015-06-15-progressbar.txt`:

OUTPUT

```
Read 102 items
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.21   50.34   63.78   57.94   75.14   99.54
```

Unfortunately `eqtlbma` will not produce any results unless 100% complete. Looking into the error log there are 4 types of issues, also see `2015-06-15-errors.txt`:

- 80 out of 100 jobs exceeded memory limit, e.g.:
  - *slurmstepd-midway106: error: Job 14865254 exceeded memory limit (9250836 > 9216000), being killed*
- 13 out of 100 jobs has early-on (at <1% progressbar) GSL error, e.g.:
  - *gsl: svd.c:286: ERROR: svd of MxN matrix, M<N, is not implemented. Default GSL error handler invoked.*
- 5 out of 100 jobs happened to not having any cis SNPs (or maybe have issues when running `bedtools`) so they are not analyzed.

- 2 out of 100 jobs failed due to `eqtlbma` error on duplicated SNPs (recall there is SNP duplication issue in GTEx input), e.g.:
  - *SNP 10_12000000_A_T_b37 is duplicated in file /project/mstephens/gtex/analysis/april2015/eqtl-_data/GTEx_Analysis_2015-01-12_eQTLInputFiles_snpMatrices/Adipose_Subcutaneous_Analysis.snps.t*
  - *SNP 12_48000000_T_C_b37 is duplicated in file /project/mstephens/gtex/analysis/april2015/eqtl-_data/GTEx_Analysis_2015-01-12_eQTLInputFiles_snpMatrices/Adipose_Subcutaneous_Analysis.snps.t*

So firstly I'll have to figure out if there is a way to reduce ram usage (without changing `eqtlbma` code). I bet simply using more batches will help because it seems to me that the program holds all the result output in RAM and write them out all at once when all is complete. This is not good when the summary statistic data itself is large.

Then I'll have to reproduce and fix or bypass the gsl error.

Finally I will have to instead of fixing GTEx file I'll just exclude these two SNPs in question out of the picture, if I do not change the behavior of `eqtlbma` (it seems to me that this should be a warning not an error).

4.2.3 Lessons, Wed Jul 22 12:05:27 CDT 2015

A couple of issues with `eqtlbma` were fixed based on suggestions from Tim. There should be no more RAM issue or gsl error. There are, however, some bad SNPs:

```
cd $DataDir/GTEx_Analysis_2015-01-12_MatrixEQTL_allCisSNPGenePairs
cat *.error | cut -f3 -d":" | awk '{print $1}' | sort -u
```

To bypass the 5 bad SNPs found so far, it is most straightforward to exclude both of them (because I do not know which ones to keep!):

```
cd $InputDir
mv snp_coords.bed.gz snp_coords.bed.dup.gz
mv snp_coords.bed.gz.tbi snp_coords.bed.dup.gz.tbi
zgrep -vP "10_12000000_A_T_b37|12_48000000_T_C_b37|1_105000000_C_CATT_b37|14_81000000_TC_T_b37|2_72000000_G_A_b37" \
snp_coords.bed.dup.gz | bgzip > snp_coords.bed.gz
tabix -p bed snp_coords.bed.gz
```

Also from what I learned, it is impossible to analyse the data with 10,000 permutations. Even QBF procedure using `--nperm 250` it will take months to complete. The plan now is to run without permutation for the entire analysis just to harvest sumstats and raw BFs (and compare with MatrixEQTL); then while running type model with this result on half of midway nodes I also rerun the BF step with `--nperm 250` on the other half of nodes.

4.2.4 Success, configuration model without permutation

Job submission

`No permutation for configuration model`

```
nBatches=100
Model=normal.perm0
for i in `seq $nBatches`; do
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     python $SrcDir/analysis_admin.py eqtlbma_batch \
     -g $InputDir/tss_coords.bed.gz \
     -s $InputDir/snp_coords.bed.gz \
     -n '"$nBatches"' -b '"$i"' -w 100000 \
     --seed 10086 -e ~/software/bin/eqtlbma_bf \
     -a $ConfDir/eqtlbma.'"$Model"'.txt' |\
sbatch -J eqtlbma_"$Model"_"$nBatches"_"$i" -e $LogDir/eqtlbma_"$Model"_"$nBatches"_"$i".e%j \
       -o $LogDir/eqtlbma_"$Model"_"$nBatches"_"$i".o%j --mem-per-cpu=10000 --time=36:00:00
sleep 1
done
```

Run completed under 24 hours, yielding to 45GB output file in various batches. For storage purpose it is best idea to first convert these files in different batches to HDF5 and merge them to one file & archive. Skipping this step for now as I want to move on to the HM and BMA steps. These output are archived as is, though:

```
tar -cvf eqtlbma_bf.normal.perm0.tar *
```

4.2.5 List of summary statistics

The following script produces sumstats lists for all results previously computed:

make sumstats lists

```
SumstatsDir=$ArchiveDir/eqtlbma_bf/July2015
for i in `find $SumstatsDir -maxdepth 1 -name "eqtlbma_bf_normal_*" -type d`; do
    for j in `ls $i/*_sumstats_*.txt.gz`; do
        echo `echo $j | sed 's/\(.*sumstats_\)\(.*\).txt.gz/\2/g'` $j >> $i.ss.list
    done
done
```

Future analysis can be based on summary statistics instead.

4.3 The Type Model

The raw BF data are fed to the type-model version of the hierarchical model EM algorithm to estimate hyperparameters. Data is 28GB compressed text which may exceed 256GB when all are loaded into RAM (which is what `eqtlbma_hm` does!). Trying to run this on midway large RAM node and see if it crashes that node ...

type model job submission

```
cd $AnalysisDir
mkdir -p C.eqtlbma_hm/RawBFs; cd C.eqtlbma_hm
j=0; for i in `find $ArchiveDir/eqtlbma_bf/July2015 -name "*abf*"`; do let j=j+1; ln -s $i RawBFs/$j.gz; done
echo '#!/bin/bash
     source $HOME/GIT/gtex-eqtls/conf/GTEx.bashrc
     eqtlbma_hm --data "RawBFs/*.gz" --nsubgrp 44 --dim 5 \
     --ngrid 10 --model types --out eqtlbma_hm_normal.txt.gz \
     >& eqtlbma_hm.log' |\
sbatch -J eqtlbma_hm -e $LogDir/eqtlbma_hm.e%j \
```

```
    -o $LogDir/eqtlbma_hm.o%j --partition=bigmem --ntasks=1 \
    --cpus-per-task=1 --mem-per-cpu=256000
```

Note

It hangs very long (or infinitely) in the queue for job with such big memory. I'm currently trying `bigmem03` on PPS cluster.

Tue July 31 11:28:21 CDT 2015

Even with large RAM the computation is not getting anywhere. Two solutions to this:

- Implement an on-line version of the current EM algorithm e.g., Cappe & Mouline (2009) [1]
- Use current EM but only compute from read subset of the data and see what it gives.

These can be done in parallel. Naturally 2) can be done by just taking a few SNPs from each gene. Data is first reorganized to HDF5 format `analysis_admin.py bf_to_h5 --action convert` and subsetting was done via `analysis_admin.py bf_to_h5 --action null`

bf_to_h5

```
mkdir RawBF; cd RawBF
for i in `ls $BFDir/*l10abfs_raw.txt.gz`; do
    echo python $SrcDir/analysis_admin.py bf_to_h5 $i --output . --action convert
done
#python £SrcDir/analysis_admin.py bf_to_h5 *.h5 --output eqtlbma_bf_normal_l10abfs_raw --action cat
```

bf_subset

```
# Takes 45 sec to process per file single process. Multiple commands on same HD will not work due to I/O issues.
for i in `ls *.h5`; do
    j=$(basename "$i").subset.txt
    python $SrcDir/analysis_admin.py bf_to_h5 $i --output $j --action null
done
cat header.txt *.subset.txt | gzip --best > eqtlbma_bf_normal_l10abfs_raw.subset.gz
```

Fri Aug 14 13:55:18 CDT 2015

Even only taking a few SNPs per gene is prohibitively slow when confidence intervals (CI) are required. As a result a parallel implementation of CI computation was done to `eqtlbma_hm`.

[1] Cappe & Mouline (2009) http://cs.stanford.edu/~pliang/papers/online-naacl2009.pdf