Very Small EQTL Analysis

R script:

```r
library(gQTLstats)
library(ldblock)
library(VariantAnnotation)
library(geuvPack)
library(batchtools)
data(geuFPKM)

ss <- stack1kg()
v17 <- ss@files[[17]]    # handle to VCF for chromosome 17

someGenes <- c("ORMDL3","GSDMB", "IKZF3", "MED24", "CSF3", "ERBB2",
               "GRB7", "MIEN1", "GSDMA", "THRA", "MSL1")
se17 <- geuFPKM[ which(rowData(geuFPKM)$gene_name %in% someGenes),]

n <- 10
vr0 <- 39.5e6
vr1 <- 40.5e6
v <- seq(vr0, vr1, length=n+1)
vl <- zipup(v[1:n],v[2:(n+1)]-1)

run.job <- function(v, se, vcf)  {
  vr <- GRanges("17", IRanges(start=v[1], end=v[2], names=c("range")))
  results <- AllAssoc(se, vcf, vr)
}
concat.job <- function(gr1, gr2)  {
  gr <- c(gr1,gr2)
}

system("rm -Rf ./registry")
reg <- makeRegistry(file.dir="./registry", conf.file = "./batchtools.conf.R")
batchMap(fun = run.job, as.list(vl), more.args = list(se=se17, vcf=v17))
submitJobs()
waitForJobs()
all.results <- reduceResults(fun = concat.job)
save(all.results, file="all.results.Rdat")
```

1. Run on ERISOne under LSF:

```r
# batchtools.conf.R
cluster.functions = makeClusterFunctionsLSF("/PHShome/sp128/EQTL/simple.tmpl")
```

Template file:

```
<%
    jt <- paste(file.dir, "/bout", sep="")
    if (!dir.exists(jt))  {
      dir.create(jt, recursive = TRUE)
    }
    jb <- fp(file.dir, "bout", sprintf("%s.bout", job.hash))

    # rscript to run JobCollection
    rscript = paste(
            "library(batchtools)",
            "library(GenomicRanges)",
            "library(gQTLstats)",
            sprintf("jc = readRDS('%s')", uri),
            sprintf("batchtools:::doJobCollection(jc, output = '%s')", log.file),
          sep=";")

    queue = "short"
%>

#BSUB-o <%= jb %>
#BSUB-q <%= queue %>

Rscript -e "<%= rscript %>"
```

2. Run on Channing under SGE:

```
# batchtools.conf.R
cluster.functions = makeClusterFunctionsSGE("/udd/resjp/EQTL/simple.tmpl")
```

Template file:
This is a very convoluted template designed to overcome two difficulties:
1. Very hard to pass arguments on the command line to qsub. qsub wants only a shell script name.
2. Rscript not available. Very hard to inline R code with R CMD BATCH because of quotation mark confusion.
This is a knock off of the Channing homemade utility qbR; qbR is a shell script which makes a shell script which runs an R script.

```
<%

    # make two additional directories in registry
    rdir <- paste(file.dir, "/rexe", sep="")
    if (!dir.exists(rdir))  {
      dir.create(rdir, recursive = TRUE)
    }

    jdir <- paste(file.dir, "/exe", sep="")
    if (!dir.exists(jdir))  {
      dir.create(jdir, recursive = TRUE)
    }

    # make R script for job to run - rscript that runs JobCollection
    rscript = paste(
            "library(batchtools)",
            "library(gQTLstats)",
            "library(GenomicRanges)",
            sprintf("jc = readRDS('%s')", uri),
            sprintf("batchtools:::doJobCollection(jc, output = '%s')", log.file),
          sep=";")

    rf <- fp(file.dir, "rexe", sprintf("%s.R", job.hash))
    cat(rscript,file=rf,sep="\n")
    Sys.chmod(rf, mode = "0777")

    # make script to run job
    jscript <- fp(file.dir, "exe", sprintf("%s.sh", job.hash))
    # cat(c("#!/bin/bash", "echo 'Hello, world'"),file=jscript,sep="\n")
    cat(paste0("/local/bin/R CMD BATCH ", rf),file=jscript,sep="\n")
    Sys.chmod(jscript, mode = "0777")

%>

#$ -N <%= job.name %>
#$ -j y
#$ -S /bin/bash
#$ -V
#$ -o <%= log.file %>
#$ -l R=true
#$ -cwd

<%= jscript %>
```

3. Running on tiny two-node AWS slurm cluster.
Note: setwd to /scratch so all nodes can read from and write to the registry

```
#batchtools.conf.R
cluster.functions = makeClusterFunctionsSlurm("/scratch/simple.tmpl")
```

Template file:

```
#!/bin/bash
<%

    # make directory in registry
    rdir <- paste(file.dir, "/rexe", sep="")
    if (!dir.exists(rdir))  {
      dir.create(rdir, recursive = TRUE)
    }

    # make R script for job to run - rscript that runs JobCollection
    rsrc = paste(
          "library(batchtools)",
          "library(gQTLstats)",
          "library(GenomicRanges)",
          sprintf("jc = readRDS('%s')", uri),
          sprintf("batchtools:::doJobCollection(jc, output = '%s')", log.file),
          sep=";")

    rscript <- fp(file.dir, "rexe", sprintf("%s.R", job.hash))
    cat(rsrc,file=rscript,sep="\n")
    Sys.chmod(rscript, mode = "0777")

%>

#SBATCH --job-name=<%= job.name %>
#SBATCH --output=<%= log.file %>
#SBATCH --error=<%= log.file %>
#SBATCH -p debug
#SBATCH -N 1
#SBATCH -n 1
#SBATCH -t 1:00

R CMD BATCH --no-save --no-restore "<%= rscript %>" /dev/stdout
```