geuvStore: sharded storage for cis-association statistics

Vincent J. Carey, stvjc at channing.harvard.edu

February 2015

Contents

1 Introduction

The *geuvStore* package demonstrates an approach to management of large numbers of statistics generated in integrative genomic analyses. The specific use case demonstrated here is cis-eQTL discovery. The following considerations motivated the design used here.

- Cluster computing will typically be used to perform cis-eQTL searches. Scalable performance is greatly aided by the *BatchJobs* infrastructure, which will create an archive of results.
  - This archive includes a database that holds information on job status (including time and memory required to complete) and result location. We consider this information worth saving.
  - The collection of results is, by default, "sharded" into a reasonable number of folders holding serialized R objects. We find this approach useful for supporting parallelizable retrieval of results.
- It makes sense to store results of cis-association analyses so that queries based on genomic addresses are rapidly resolved. Thus all the results are stored in `GRanges` instances, and queries based on `GRanges` are efficiently resolvable if an optional index is prepared before use.

2 Illustration

2.1 Construction of mediator and indices

The most basic entity mediating access to the information is the `BatchJobs` registry object. This is typically not created in a portable format, but includes directory information that we modify during package installation.

```
suppressPackageStartupMessages(library(geuvStore))
prst = partialRegistry()
## NOTE: ids of 92 jobs available for this registry can be obtained with partialIds()
```

```
prst
## Job registry:  flatReg
##   Number of jobs:  2283
##   Files dir: /tmp/Rtmpqs60Nl/Rinst4efd6c2bd1c9/geuvStore/geuvStore
##   Work dir: /tmp/Rtmpqs60Nl/Rbuild4efdfd142d9/geuvStore/vignettes
##   Multiple result files: FALSE
##   Seed: 123
```

```
##   Required packages: BatchJobs, GenomeInfoDb, GenomicRanges, geuvPack, GGtools, Rsamtools, VariantAnnot
```

Note that the show method mentions 2283 "jobs". The job set was defined using a partition of 22830 genes into sets of 10 genes each. Association statistics were recorded between expression levels of each gene (as recorded in the GEUVADIS FPKM report) and all SNP with MAF $> .01$ lying within a radius of 1 million bp upstream or downstream from the gene region. This package provides access to a selection of 92 jobs.

We use the `ciseStore` class to mediate between the user and the results data. This includes optional mappings based on gene identifiers (in the case of this example, these are Ensembl gene IDs) and `GRanges`. We have stored the maps, but they can be computed in real time if need be.

```
library(gQTLBase)
# prstore = ciseStore(prst, addProbeMap=TRUE, addRangeMap=TRUE)
prstore = makeGeuvStore()
prstore
## ciseStore instance with 92 completed jobs.
## excerpt from job  1 :
## GRanges object with 1 range and 11 metadata columns:
##       seqnames                ranges strand |       paramRangeID
##          <Rle>             <IRanges>  <Rle> |           <factor>
##   [1]        1 [225418903, 225418903]      * | ENSG00000183814.10
##                  REF             ALT     chisq permScore_1 permScore_2
##       <DNAStringSet> <CharacterList> <numeric>   <numeric>   <numeric>
##   [1]              G               A 0.2972831   0.1150969    8.312895
##       permScore_3             snp        MAF            probeid   mindist
##         <numeric>     <character>  <numeric>       <character> <numeric>
##   [1]   0.1021324 snp_1_225418903 0.03246753 ENSG00000183814.10    999947
##   -------
##   seqinfo: 1 sequence from hg19 genome; no seqlengths
```

2.2 Extraction of content

For a vector of gene identifiers, all available results are extracted.

```
head(
extractByProbes(prstore,
   probeids=c("ENSG00000183814.10", "ENSG00000174827.9"))
)
## GRanges object with 6 ranges and 12 metadata columns:
##       seqnames                ranges strand |       paramRangeID
##          <Rle>             <IRanges>  <Rle> |           <factor>
##   [1]        1 [225418903, 225418903]      * | ENSG00000183814.10
##   [2]        1 [225419456, 225419456]      * | ENSG00000183814.10
##   [3]        1 [225419667, 225419667]      * | ENSG00000183814.10
##   [4]        1 [225420024, 225420024]      * | ENSG00000183814.10
##   [5]        1 [225420751, 225420751]      * | ENSG00000183814.10
##   [6]        1 [225420977, 225420977]      * | ENSG00000183814.10
##                  REF             ALT       chisq permScore_1 permScore_2
##       <DNAStringSet> <CharacterList>   <numeric>   <numeric>   <numeric>
##   [1]              G               A 0.297283124   0.1150969   8.3128951
##   [2]              T               C 0.001771913   2.4850662   0.2125758
##   [3]              G               C 0.055747393   3.4358410   0.6570552
##   [4]              G               A 0.028064071   3.5856677   0.3993631
##   [5]              C               T 0.024406613   0.4720181   0.4061977
```

```
##   [6]              C               T 0.088637603   3.8852537   1.5905848
##       permScore_3             snp         MAF            probeid   mindist
##         <numeric>     <character>  <numeric>        <character> <numeric>
##   [1]   0.1021324 snp_1_225418903 0.03246753 ENSG00000183814.10    999947
##   [2]   1.1020702 snp_1_225419456 0.28787879 ENSG00000183814.10    999394
##   [3]   0.7084834 snp_1_225419667 0.17532468 ENSG00000183814.10    999183
##   [4]   0.2936893 snp_1_225420024 0.05952381 ENSG00000183814.10    998826
##   [5]   2.1247086 snp_1_225420751 0.11038961 ENSG00000183814.10    998099
##   [6]   1.5894870 snp_1_225420977 0.02272727 ENSG00000183814.10    997873
##           jobid
##       <integer>
##   [1]         1
##   [2]         1
##   [3]         1
##   [4]         1
##   [5]         1
##   [6]         1
##   -------
##   seqinfo: 1 sequence from hg19 genome; no seqlengths
```

For a request based on genomic coordinates, a GRanges can be used to query. `findOverlaps` is used, and all results for genes whose regions overlap the query ranges are returned.

```
head(
extractByRanges(prstore, GRanges("1", IRanges(146000000, width=1e6)))
)
## GRanges object with 6 ranges and 12 metadata columns:
##       seqnames                ranges strand |      paramRangeID
##          <Rle>             <IRanges>  <Rle> |          <factor>
##   [1]        1 [146003411, 146003411]      * | ENSG00000227280.1
##   [2]        1 [146013533, 146013533]      * | ENSG00000227280.1
##   [3]        1 [146013577, 146013577]      * | ENSG00000227280.1
##   [4]        1 [146025251, 146025251]      * | ENSG00000227280.1
##   [5]        1 [146028147, 146028147]      * | ENSG00000227280.1
##   [6]        1 [146038943, 146038943]      * | ENSG00000227280.1
##                   REF             ALT        chisq permScore_1 permScore_2
##       <DNAStringSet> <CharacterList>    <numeric>   <numeric>   <numeric>
##   [1]              A               G 3.873386e-07 0.004976712  1.54592531
##   [2]              G               A 4.070230e-02 2.195225999  0.70197195
##   [3]              A               C 4.679088e-01 0.708770449  0.18964422
##   [4]              C               T 1.263953e-01 0.005527808  0.00379705
##   [5]              A               G 5.320439e-02 0.045255895  1.68327379
##   [6]              A               C 5.877874e-01 0.752969457  0.50468710
##       permScore_3             snp         MAF           probeid   mindist
##         <numeric>     <character>  <numeric>       <character> <numeric>
##   [1]  0.24055958 snp_1_146003411 0.03800475 ENSG00000227280.1    627857
##   [2]  0.03620320 snp_1_146013533 0.05225653 ENSG00000227280.1    637979
##   [3]  0.86766185 snp_1_146013577 0.36817102 ENSG00000227280.1    638023
##   [4]  0.35851185 snp_1_146025251 0.06532067 ENSG00000227280.1    649697
##   [5]  0.63502184 snp_1_146028147 0.22565321 ENSG00000227280.1    652593
##   [6]  0.04864164 snp_1_146038943 0.05819477 ENSG00000227280.1    663389
##           jobid
##       <integer>
##   [1]         1
```

```
##   [2]         1
##   [3]         1
##   [4]         1
##   [5]         1
##   [6]         1
##   -------
##   seqinfo: 1 sequence from hg19 genome; no seqlengths
```

2.3 Applicative programming

The `storeApply` function will be evaluated on all store elements. It dispatches to *BiocParallel* `bplapply`, and the registered 'BiocParallelParam' is used implicitly. Some list flattening may be required after use.

```
lens = storeApply(prstore, length)
summary(unlist(lens))
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   61950   74810   80110   82170   86330  168100
```

2.4 Visualization support

As of March 5, 2015 "biocLite('vjcitn/gQTLbrowser')" will acquire a package including an interactive visualization function. "example('gQTLbrowse')" will load a queriable interface into the browser, with tooltips on the Manhattan plot for the selected gene.