GenomicScores: efficient storage and retrieval of genomewide position-specific scores

Robert Castelo
robert.castelo@upf.edu @robertclab

joint work with
Pau Puigdevall
pau.puigdevall@upf.edu

Dept. of Experimental and Health Sciences
Universitat Pompeu Fabra
Barcelona, Spain

BioC 2017 - Developer Day
Boston, USA
July 26, 2017
Genomewide position-specific scores are ubiquitous in genomic analyses, specially for the filtering and interpretation of single nucleotide variants.

Some of the most popular score sets are:


The size of some of them, e.g., (≈ 2.5Gb phastCons, ≈ 80Gb CADD), derived from storing double-precision numbers for millions of nucleotides along the genome, makes it difficult to use them interactively or integrate them into R workflows.
Sometimes, measurements and statistical models generate false precision, i.e., values that are meaningless or not that useful from the scientific point of view (sometimes this is application-dependent).

Using **lossy compression**, also known as **quantization**, we can trade off precision for compression without compromising the scientific integrity of the data (Zender, 2016).

Lossy compression leads to a subset of **quantized** values, much smaller than the original set of genomic scores.

Quantized values often lead to runs of identical values along the genome that can be further compressed with run-length encoding (RLE) vectors.
The GenomicScores package

- Efficient storage and retrieval of genomewide position-specific scores.

- Supports annotation packages such as phastCons100way.UCSC.hg19, but can be also used to fetch further score sets as AnnotationHub resources.

- Defines the \textit{GScores} class of objects, inspired by the former \textit{SNPlocs} class, and some of its accessors are (see help page for full list):

  - scores(object, ranges, scores.only=FALSE, summaryFun=mean, quantized=FALSE, caching=TRUE)
  - name(x): name of the set of scores, e.g., phastCons100way.UCSC.hg19.
  - type(x): type of scores, e.g., phastCons100way.
  - provider(x): provider of the score data, e.g., UCSC.
  - providerVersion(x): version of the data given by the provider.
  - organism(x): organism on which the scores are defined.
  - seqinfo(x): information about the genome sequence.
  - qfun(x): quantization function.
  - dqfun(x): dequantization function.
  - citation(x): \textit{bibentry} object on how to cite these data.
> library(phastCons100way.UCSC.hg19)
> gsco <- phastCons100way.UCSC.hg19
> gsco

GScores object
# organism: Homo sapiens (UCSC, hg19)
# provider: UCSC
# provider version: 09Feb2014
# download date: Mar 17, 2017
# loaded sequences: chr19_g1000208_random
# maximum abs. error: 0.05

> scores(gsco, GRanges(seqnames="chr7", IRanges(start=117232380, width=1)))

GRanges object with 1 range and 1 metadata column:
  seqnames ranges strand | scores
    <Rle>     <IRanges> <Rle> | <numeric>
[1]  chr7 [117232380, 117232380] * | 0.8

-------
seqinfo: 1 sequence from an unspecified genome; no seqlengths

> gsco

GScores object
# organism: Homo sapiens (UCSC, hg19)
# provider: UCSC
# provider version: 09Feb2014
# download date: Mar 17, 2017
# loaded sequences: chr19_g1000208_random, chr7
# maximum abs. error: 0.05
> library(GenomicScores)  
> availableGScores()

snapshotDate(): 2017-07-11

[1] "cadd.v1.3.hg19"  "fitCons.UCSC.hg19"
[3] "mcap.v1.0.hg19"  "phastCons100way.UCSC.hg19"
[5] "phastCons100way.UCSC.hg38" "phastCons60way.UCSC.mm10"
[7] "phastCons7way.UCSC.hg38" "phyloP100way.UCSC.hg19"
[9] "phyloP100way.UCSC.hg38"

> cadd <- getGScores("cadd.v1.3.hg19")

> citation(cadd)


> makeGScoresPackage(cadd, maintainer="me", author="me <me@example.com>", version="1.0.0")

Creating package in ./cadd.v1.3.hg19
## Future directions

- Current compression ratios, are:

<table>
<thead>
<tr>
<th>Score set</th>
<th>Original</th>
<th>Compressed</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>fitCons.UCSC.hg19</td>
<td>76 Mb</td>
<td>25 Mb</td>
<td>≈ 3</td>
</tr>
<tr>
<td>phyloP100way.UCSC.hg19</td>
<td>5.1 Gb</td>
<td>1.2 Gb</td>
<td>≈ 4</td>
</tr>
<tr>
<td>phastCons100way.UCSC.hg19</td>
<td>2.5 Gb</td>
<td>233 Mb</td>
<td>≈ 10</td>
</tr>
<tr>
<td>mcap.v1.0.hg19</td>
<td>729 Mb</td>
<td>61 Mb</td>
<td>≈ 12</td>
</tr>
<tr>
<td>cadd.v1.3.hg19</td>
<td>80 Gb</td>
<td>716 Mb</td>
<td>≈ 114</td>
</tr>
</tbody>
</table>

Can they be improved? Do we need different lossy compression for different applications?

- Current *GScores* class is based on the “older” *SNPlocs* class. This should probably change to the newer *ODLT_SNPlocs* class.
Future directions

- Should we integrate the \textit{MafDb} class, as a subclass of \textit{GScores}?

```
> library(MafDb.gnomAD.r2.0.1.hs37d5)
> mafdb <- MafDb.gnomAD.r2.0.1.hs37d5
> mafdb

Minor allele frequency Db (MafDb) object
# organism: Homo sapiens
# provider: BroadInstitute
# provider version: r2.0.1
# download date: Apr 10, 2017
# loaded sequences (SNVs): none
# loaded sequences (nonSNVs): none
# loaded populations (SNVs): none
# loaded populations (nonSNVs): none
# nr. of variants: 241056551

> populations(mafdb)
[1] "AF"    "AF_AFR" "AF_AMR" "AF_ASJ" "AF_EAS" "AF_Female"
[7] "AF_FIN" "AF_Male" "AF_NFE" "AF_OTH"

> mafByOverlaps(mafdb, "15:28356859", populations(mafdb))

GRanges object with 1 range and 10 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>AF</th>
<th>AF_AFR</th>
<th>AF_AMR</th>
<th>AF_ASJ</th>
<th>AF_EAS</th>
<th>AF_Female</th>
<th>AF_FIN</th>
<th>AF_Male</th>
<th>AF_NFE</th>
<th>AF_OTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
</tr>
<tr>
<td>[1] 15</td>
<td>[28356859, 28356859]</td>
<td>*</td>
<td>0.44</td>
<td>0.13</td>
<td>0.22</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AF_ASJ</td>
<td>AF_EAS</td>
<td>AF_Female</td>
<td>AF_FIN</td>
<td>AF_Male</td>
<td>AF_NFE</td>
<td>AF_OTH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td>&lt;numeric&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[1] 0.46</td>
<td>0.001</td>
<td>0.42</td>
<td>0.12</td>
<td>0.47</td>
<td>0.2</td>
<td>0.32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

seqinfo: 1 sequence from an unspecified genome; no seqlengths
Comments, bugs, issues and acknowledgments

- Comments to robert.castelo@upf.edu

- Bugs and issues to
  https://github.com/rcastelo/GenomicScores/issues

- Acknowledgments to:
  - Valerie Obenchain and Martin Morgan for their help to set up the AnnotationHub resources.
  - Michael Lawrence and Hervé Pages for useful discussions on how to store and retrieve score and minor allele frequency data.
  - Funding: TIN2015-71079-P (MINECO/FEDER, UE).