

Genetic data in R and GDS format

Jon Chernus

Department of Human Genetics
School of Public Health
University of Pittsburgh

Document created: November 23, 2024

This slide set is called `genetic_data_in_r_gds.pdf` and is located in the “`28_genetics_data_in_R_gds`” folder of our Lectures repository.

How to load big genetics files into R

Without specialized packages?

- `read.table?`
- `data.table::fread?`

Advantages of specialized packages

- Process big files faster
- Integrate other data more easily
- Read-to-use analysis tools

Without specialized packages

It's possible to read in text-based files with non-genetics packages

- `read.table` (or other base R functions)
- `data.table::fread` (quite fast and flexible – try it)

Limitations

- Relatively slow/inefficient, especially for bigger data sets
- Stores the files in working memory
- Not specialized/suited for genetics data structures

PLINK and VCF files can be read in easily.

Advantages

- Some can handle very large data sets without having to load the entire file into memory
- More effective at handling genetics-specific data structures

The BEDMatrix package

- Easily read in PLINK files with the `BEDMatrix` function
- Read in VCF by first using `plink --recode vcf`

Several types exist, so you'll need to convert between them (explained later)

- The `SeqArray` package uses `SeqArray` GDS
- The `SNPRelate` package uses `SNP` GDS
- The `GWASTools` package has its own GDS formats

Some packages that use GDS

SeqArray has basic utilities for SeqArray GDS files.

It interfaces with other packages such as

- SeqVarTools , a tool set with more GDS utilities
- GWASTools, a tool set for GWAS data cleaning
- SNPRelate, a tool set for relatedness and PCA calculations
- GENESIS, a tool set for GWAS, relatedness, and PCA in family data

GDS files are fast, compact, and flexible

Compare the size of 200 billion genotypes in the 1000 Genomes Project:

File format	Approximate size (Gb)
.vcf.gz	14.4
.bcf	12.3
.gds	2.6-5.7

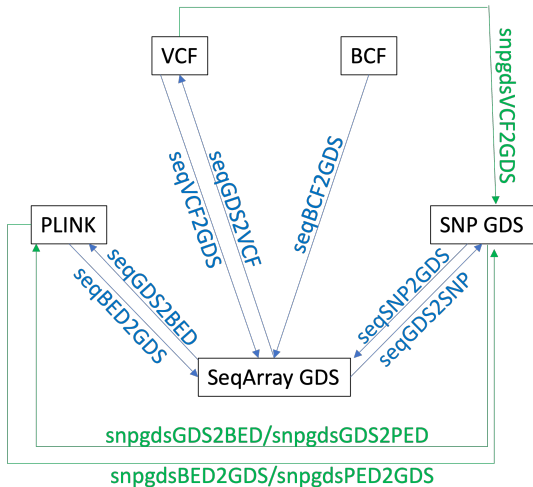
Some properties of GDS files

- Not human-readable
- Special accessor functions are required to interact with their fields
- Hierarchically and flexibly structured
- Customizable and can hold annotation like VCF
- Support both sample and subject annotation

How to convert to/from GDS

seqArray package

SNPRelate package



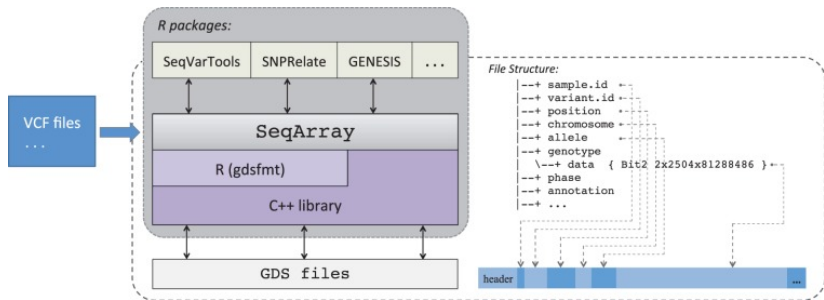


Figure 1: SeqArray framework (source: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5860110/>)

SeqArray basic utilities

There are several dozen commands, including the most important file conversion commands (mentioned above). Here are a few:

Command	What it does
<code>seqVCF2GDS</code>	Convert from VCF to GDS
<code>seqBCF2GDS</code>	Convert BCF to GDS
<code>seqOpen</code>	Open the GDS file
<code>seqClose</code>	Close it
<code>seqGetData</code>	Get data from a SeqArray file
<code>seqSetFilter</code>	Define subsets of samples or variants
<code>seqSetFilterChrom</code>	Define subsets by regions
<code>seqResetFilter</code>	Reset filtering
<code>seqApply</code>	Apply user-defined functions across samples or variants

Opening a .gds file with SeqArray

```
library("SeqArray", quietly = TRUE, verbose = FALSE,
       warn.conflicts = FALSE)
gds_path <- paste0(.libPaths(), "/SeqArray/extdata/CEU_Exon.gds")
seq <- seqOpen(gds_path)
```

Object of class "SeqVarGDSClass"

File: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/SeqArray/extdata/CEU_Exon.gds (287.6K)

```
+ [ ] *
| -> description [ ] *
| -> sample.id { Str8 90 LZMA_ra(34.7%), 257B } *
| -> variant.id { Int32 1348 LZMA_ra(16.7%), 905B } *
| -> position { Int32 1348 LZMA_ra(64.4%), 3.4K } *
| -> chromosome { Str8 1348 LZMA_ra(4.39%), 157B } *
| -> allele { Str8 1348 LZMA_ra(16.6%), 901B } *
| -> genotype [ ] *
| | -> data { Bit2 2x90x1348 LZMA_ra(26.3%), 15.6K } *
| | -> ~data { Bit2 2x1348x90 LZMA_ra(29.2%), 17.3K } *
| | -> extra.index { Int32 3x0 LZMA_ra, 18B } *
| | \ -> extra { Int16 0 LZMA_ra, 18B }
| -> phase [ ]
| | -> data { Bit1 90x1348 LZMA_ra(0.86%), 137B } *
| | -> ~data { Bit1 1348x90 LZMA_ra(0.86%), 137B } *
| | -> extra.index { Int32 3x0 LZMA_ra, 18B } *
| | \ -> extra { Bit1 0 LZMA_ra, 18B }
| -> annotation [ ]
| | -> id { Str8 1348 LZMA_ra(38.3%), 5.5K } *
| | -> qual { Float32 1348 LZMA_ra(2.11%), 121B } *
| | -> filter { Int32,factor 1348 LZMA_ra(2.11%), 121B } *
| | | -> info [ ]
| | | | -> AA { Str8 1328 LZMA_ra(22.1%), 593B } *
| | | | -> AC { Int32 1348 LZMA_ra(24.1%), 1.3K } *
| | | | -> AN { Int32 1348 LZMA_ra(19.6%), 1.0K } *
| | | | -> DP { Int32 1348 LZMA_ra(47.7%), 2.5K } *
| | | | -> HM2 { Bit1 1348 LZMA_ra(145.6%), 253B } *
| | | | -> HM3 { Bit1 1348 LZMA_ra(145.6%), 253B } *
| | | | -> OR { Str8 1348 LZMA_ra(19.6%), 341B } *
| | | | -> GP { Str8 1348 LZMA_ra(24.3%), 3.8K } *
| | | | \ -> BN { Int32 1348 LZMA_ra(20.7%), 1.1K } *
| | \ -> format [ ]
| | | -> DP [ ] *
| | | | -> data { VL_Int 90x1348 LZMA_ra(70.8%), 115.2K } *
| | | | \ -> ~data { VL_Int 1348x90 LZMA_ra(65.1%), 105.9K } *
| -> sample.annotation [ ]
| | -> family { Str8 90 LZMA_ra(55.0%), 221B } *
```

Exploring a .gds file with SeqArray

```
# Extract some info
var.ids <- seqGetData(g, "variant.id")
samp.ids <- seqGetData(g, "sample.id")
chroms <- seqGetData(g, "chromosome")
rsids <- seqGetData(g, "annotation/id")

# Look at some of it
length(samp.ids)
```

```
[1] 90
```

```
head(samp.ids)
```

```
[1] "NA06984" "NA06985" "NA06986" "NA06989"
[5] "NA06994" "NA07000"
```

```
table(chroms)
```

```
chroms
 1  10  11  12  13  14  15  16  17  18  19  2
142 70  16 62  11 61  46 84 100 54 111 59
 20 21 22  3  4  5  6  7  8  9
 59 23 23 81 48 61 99 58 51 29
```

```
length(rsids)
```

```
[1] 1348
```

```
head(rsids)
```

```
[1] "rs111751804" "rs114390380" "rs1320571"
[4] "rs2760321" "rs2760320" "rs116230480"
```

Using SeqArray to explore a .gds file (con't.)

```
# Look at alleles  
alleles <- seqGetData(g, "allele")  
length(alleles)
```

```
[1] 1348
```

```
head(alleles)
```

```
[1] "T,C" "G,A" "G,A" "T,C" "G,C" "C,T"
```

```
# Look at allele counts  
allele_counts <- seqGetData(g, "annotation/info/AC")  
length(allele_counts)
```

```
[1] 1348
```

```
head(allele_counts)
```

```
[1] 4 1 6 128 13 1
```

```
# Look at sample family IDs  
sample_annot <- seqGetData(g, "sample.annotation/family")  
str(sample_annot)
```

```
chr [1:90] "1328" "" "13291" "1328" "1340" ...
```

Using SeqArray to look at genotypes

Genotypes are stored in a 3D array - a little unwieldy

```
# Look at genotypes  
genotypes <- seqGetData(g, "genotype")  
str(genotypes)
```

```
int [1:2, 1:90, 1:1348] NA NA NA NA 0 0 NA NA NA NA ...  
- attr(*, "dimnames")=List of 3  
  ..$ allele : NULL  
  ..$ sample : NULL  
  ..$ variant: NULL
```

```
dim(genotypes)
```

```
[1]    2   90 1348
```

Using SeqArray to look at genotypes (con't.)

Instead, extract genotypes with `$dosage`:

```
# Now it's not 3D; look at first 10 samples and  
# variants  
genotypes <- seqGetData(g, "$dosage")  
str(genotypes)
```

```
int [1:90, 1:1348] NA NA 2 NA NA 2 2 2 2 2 ...  
- attr(*, "dimnames")=List of 2  
..$ sample : NULL  
..$ variant: NULL
```

```
genotypes[1:10, 1:10]
```

```
      variant  
sample [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]  
[1,]   NA   NA    2    1    2    2    2    2    2    2  
[2,]   NA   NA    2    0    2    2    2    2    2    2  
[3,]    2    2    2    0    2    1    2    2    1    1  
[4,]   NA   NA    2   NA    2    2    2    2    1    1  
[5,]   NA   NA    2   NA    2    2    2    2    2    2  
[6,]    2    2    2    0    1    2    2    2    2    1  
[7,]    2    2    2    0    2    2    2    2    2    2  
[8,]    2    2    2    0    2    2    2    2    2    2  
[9,]    2    1    2    0    2    2    2    2    2    2  
[10,]   2    2    2    0    2    2    1    2    2    2
```


Subsetting a GDS file with seqArray

```
seqGetData(g, "sample.id")[10] # Get sample 10's ID
```

```
[1] "NA07346"
```

```
seqGetData(g, "variant.id")[5] # Get SNP 5's ID
```

```
[1] 5
```

```
seqGetData(g, "annotation/id")[5] # Get SNP 5's rsID
```

```
[1] "rs2760320"
```

```
seqGetData(g, "allele")[5] # Get REF,ALT alleles for the SNP
```

```
[1] "G,C"
```

```
seqGetData(g, "$dosage")[10, 5] # Get sample 10's genotype at SNP 5
```

```
[1] 2
```

Filtering a GDS file with seqArray

```
seqResetFilter(g) # How many samples/variants to start?
```

```
# of selected samples: 90  
# of selected variants: 1,348
```

```
seqSetFilter(g, sample.sel = c(1, 10:14, 20)) # Select 7 samples
```

```
# of selected samples: 7
```

```
seqSetFilter(g, variant.sel = c(1, 5, 25, 32)) # Select variants
```

```
# of selected variants: 4
```

```
seqGetData(g, "$dosage") # Look at genotypes
```

```
      variant  
sample [,1] [,2] [,3] [,4]  
[1,]   NA    2    2    2  
[2,]    2    2    2    2  
[3,]    2    2    2    2  
[4,]    2    2    2    2  
[5,]   NA    2    2    2  
[6,]    2    2    2    2  
[7,]   NA    2    2    2
```

```
seqResetFilter(g) # Back to 90 samples and 1348 variants
```

```
# of selected samples: 90  
# of selected variants: 1,348
```

Calculating allele frequencies two ways:

```
# Calculate allele frequencies 'manually'
CalcFreq <- function(x) {
  mean(x == 0, na.rm = TRUE)
}
af <- seqApply(gdsfile = g, var.name = "genotype",
  as.is = "double", margin = "by.variant", FUN = CalcFreq)
head(af)
```

```
[1] 0.9649123 0.9905660 0.9610390 0.1232877
[5] 0.9269663 0.9943820
```

```
# Do it again with built-in function (same
# result)
af2 <- seqAlleleFreq(gdsfile = g)
head(af2)
```

```
[1] 0.9649123 0.9905660 0.9610390 0.1232877
[5] 0.9269663 0.9943820
```

SeqVarTools

Expands on SeqArray for dealing with GDS sequencing data.
(Examples that follow are from package vignette.)

```
library("SeqVarTools", quietly = TRUE, verbose = FALSE,
       warn.conflicts = FALSE)
vcffile <- seqExampleFileName("vcf")
gdsfile <- "./data/tmp.gds"
seqVCF2GDS(vcffile, gdsfile, verbose = FALSE)
gds <- seqOpen(gdsfile)
gds
```

Object of class "SeqVarGDSClass"

```
File: /Users/jonathanchernus/Documents/Teaching/2024f/HUGEN2071/lectures/live_lectures_github/HuGen2071-Lectures/28_genetics_data_in_R_gds/data/tmp.gds (163.0
+ [ ] *
|---+ description [ ] *
|---+ sample.id { Str8 90 LZMA_ra(34.7%), 257B } *
|---+ variant.id { Int32 1348 LZMA_ra(16.7%), 905B } *
|---+ position { Int32 1348 LZMA_ra(64.4%), 3.4K } *
|---+ chromosome { Str8 1348 LZMA_ra(4.39%), 157B } *
|---+ allele { Str8 1348 LZMA_ra(16.6%), 901B } *
|---+ genotype [ ] *
| |---+ data { Bit2 2x90x1348 LZMA_ra(26.3%), 15.6K } *
| |---+ extra.index { Int32 3x0 LZMA_ra, 18B } *
| | \---+ extra { Int16 0 LZMA_ra, 18B }
|---+ phase [ ]
| |---+ data { Bit1 90x1348 LZMA_ra(0.86%), 137B } *
| |---+ extra.index { Int32 3x0 LZMA_ra, 18B } *
| | \---+ extra { Bit1 0 LZMA_ra, 18B }
|---+ annotation [ ]
| |---+ id { Str8 1348 LZMA_ra(38.3%), 5.5K } *
| |---+ qual { Float32 1348 LZMA_ra(2.11%), 121B } *
| |---+ filter { Int32_factor 1348 LZMA_ra(2.11%), 121B } *
| |---+ info [ ]
| | |---+ AA { Str8 1328 LZMA_ra(22.1%), 593B } *
| | |---+ AC { Int32 1348 LZMA_ra(24.1%), 1.3K } *
| | |---+ AN { Int32 1348 LZMA_ra(19.6%), 1.0K } *
| | |---+ DP { Int32 1348 LZMA_ra(47.7%), 2.5K } *
| | |---+ HM2 { Bit1 1348 LZMA_ra(145.6%), 253B } *
| | |---+ HM3 { Bit1 1348 LZMA_ra(145.6%), 253B } *
| | |---+ OR { Str8 1348 LZMA_ra(19.6%), 341B } *
| | |---+ GP { Str8 1348 LZMA_ra(24.3%), 3.8K } *
| | |---+ BH { Int32 1348 LZMA_ra(20.7%), 1.1K } *
| |---+ format [ ]
| | \---+ DP [ ] *
```

SeqVarTools (con't.)

```
head(refChar(gds)) # Look at REF alleles
```

```
[1] "T" "G" "G" "T" "G" "C"
```

```
head(altChar(gds)) # Alt alleles
```

```
[1] "C" "A" "A" "C" "C" "T"
```

```
# Is everything bi-allelic? Investigate  
table(mAlleles(gds))
```

```
  2  3  
1346 2
```

```
multi.allelic <- which(mAlleles(gds) > 2)  
altChar(gds)[multi.allelic]
```

```
[1] "T,CT" "T,AT"
```

```
altChar(gds, n = 1)[multi.allelic]
```

```
[1] "T" "T"
```

```
altChar(gds, n = 2)[multi.allelic]
```

```
[1] "CT" "AT"
```

```
# Which are SNVs vs indels?  
table(isSNV(gds))
```

```
FALSE TRUE  
  2 1346
```

```
isSNV(gds)[multi.allelic]
```

```
[1] FALSE FALSE
```

SeqVarTools (con't.)

Looking at genotypes is easier:

```
geno <- getGenotype(gds)
dim(geno)
```

```
[1] 90 1348
```

```
geno[1:10, 1:5]
```

```
      variant
sample 1     2     3     4     5
NA06984 NA   NA   "0/0" "1/0" "0/0"
NA06985 NA   NA   "0/0" "1/1" "0/0"
NA06986 "0/0" "0/0" "0/0" "1/1" "0/0"
NA06989 NA   NA   "0/0" NA   "0/0"
NA06994 NA   NA   "0/0" NA   "0/0"
NA07000 "0/0" "0/0" "0/0" "1/1" "1/0"
NA07037 "0/0" "0/0" "0/0" "1/1" "0/0"
NA07048 "0/0" "0/0" "0/0" "1/1" "0/0"
NA07051 "0/0" "1/0" "0/0" "1/1" "0/0"
NA07346 "0/0" "0/0" "0/0" "1/1" "0/0"
```

```
geno <- getGenotypeAlleles(gds)
geno[1:10, 1:5]
```

```
      variant
sample 1     2     3     4     5
NA06984 NA   NA   "G/G" "C/T" "G/G"
NA06985 NA   NA   "G/G" "C/C" "G/G"
NA06986 "T/T" "G/G" "G/G" "C/C" "G/G"
NA06989 NA   NA   "G/G" NA   "G/G"
```

Some other functions:

Command	What it does
<code>refDosage</code>	Get matrix of dosage for REF allele
<code>altDosage</code>	Get matrix of dosage for ALT allele
<code>variantInfo</code>	Get data frame of variant info
<code>hwe</code>	Hardy-Weinberg equilibrium test
<code>titv</code>	Calculate transition/transversion ratio
<code>missingGenotypeRate</code>	Calculate missing genotype rate by variant or by sample
<code>heterozygosity</code>	Calculate heterozygosity (by variant/ or sample)
<code>pca</code>	Principal component analysis
<code>mendelErr</code>	Detect Mendelian errors
<code>SeqVarData</code>	Combine GDS with annotation data
<code>regression</code>	Linear/logistic regression on variants

SeqVarTools examples

```
titv(gds) # Entire dataset
```

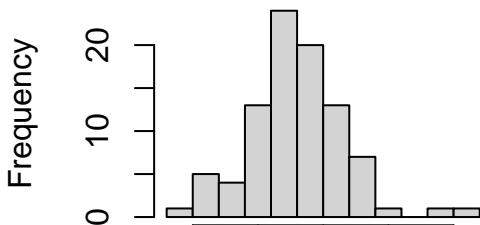
```
[1] 3.562712
```

```
titvs <- titv(gds, by.sample = TRUE) # For each sample  
head(titvs)
```

```
[1] 4.352941 3.791667 3.439394 3.568966 3.750000  
[6] 3.646154
```

```
hist(titvs)
```

Histogram of titvs



- Use `showfile.gds(closeall=TRUE, verbose=TRUE)` to close any/all open gds files
- You must close a gds file before opening it a second time
- Using `rm(list=ls())` will not close gds files
- Other functions in `seqArray` and `seqVarTools` can be used to close gds files, too

GWASTools is an R package for cleaning GWAS data.

.gds files in this context can include

- Raw chip intensity data
- Genotype calls
- SNP annotation
- Sample annotation

GWASTools has special data formats and functions for streamlining the cleaning process and reformatting files (see the exercises accompanying this lecture).

In HUGEN 2072, you'll see GDS files can be used in the GENESIS package for

- Generation of principal components of ancestry:
`GENESIS::pcair`
- Generation of kinship matrices: `GENESIS::pcrelate`
- Association testing, including mixed models:
`GENESIS::assocTestSingle`

Try running:

- `browseVignettes("SeqArray")`
- `browseVignettes("SeqVarTools")`
- `browseVignettes("GENESIS")`
- `browseVignettes("GWASTools")`
- `browseVignettes("SNPRelate")`