

gwascat: structuring and querying the NHGRI GWAS catalog

VJ Carey*

March 31, 2012

Contents

1	Introduction	1
1.1	Installation	2
1.2	Attachment and access to documentation	2
1.3	Illustrations	2
2	SNP sets and trait sets	4
2.1	SNPs by name	4
2.2	Traits by genomic location	5
3	Counting alleles associated with traits	6
4	Imputation to unobserved loci	8
5	Formalizing disease traits with Disease Ontology	10
6	Appendix: Adequacy of location annotation	12

1 Introduction

NHGRI maintains and routinely updates a database of selected genome-wide association studies. This document describes R/Bioconductor facilities for working with contents of this database.

*Generous support of Robert Gentleman and the Computational Biology Group of Genentech, Inc. is gratefully acknowledged

1.1 Installation

The package can be installed using Bioconductor's *BiocInstaller* package, with the sequence

```
library(BiocInstaller)
biocLite("gwascat")
```

1.2 Attachment and access to documentation

Once the package has been installed, use `library(gwascat)` to obtain interactive access to all the facilities. After executing this command, use `help(package="gwascat")` to obtain an overview. The current version of this vignette can always be accessed at www.bioconductor.org, or by suitably navigating the web pages generated with `help.start()`.

Some noteworthy limitations: As of 2012.03.20, there are 213 records in the database for which no SNP is identified. There are 283 records for which no chromosomal position of the associated locus is given.

1.3 Illustrations

Available functions are:

```
> library(gwascat)
> objects("package:gwascat")

[1] "chklocs"           "elementMetadata"   "getRsids"
[4] "getTraits"         "locs4trait"        "obo2graphNEL"
[7] "ranges"            "riskyAlleleCount"  "subsetByChromosome"
[10] "subsetByTraits"    "topTraits"
```

The GRanges instance with all SNP-disease associations is:

```
> gwrngs
```

gwasloc instance with 7273 records and 34 attributes per record.

Extracted: 2012.03.20

Excerpt:

GRanges with 5 ranges and 3 elementMetadata cols:

	seqnames	ranges	strand	Disease.Trait	SNPs
	<Rle>	<IRanges>	<Rle>	<character>	<character>
[1]	chr9	[136149229, 136149229]	*	Duodenal ulcer	rs505922
[2]	chr8	[143761931, 143761931]	*	Duodenal ulcer	rs2294008
[3]	chr7	[30937178, 30937178]	*	Nephrolithiasis	rs1000597
[4]	chr13	[42754522, 42754522]	*	Nephrolithiasis	rs4142110

```
[5] chr5 [176798306, 176798306] * | Nephrolithiasis rs11746443
      p.Value
      <numeric>
[1] 1e-10
[2] 2e-33
[3] 2e-14
[4] 5e-09
[5] 9e-12
```

```
---
```

```
seqlengths:
```

```
chr1 chr10 chr11 chr12 chr13 chr14 ... chr4 chr5 chr6 chr7 chr8 chr9
NA NA NA NA NA NA ... NA NA NA NA NA NA
```

To determine the most frequently occurring traits:

```
> topTraits(gwrngs)
```

```

      Height      Type 2 diabetes      Multiple sclerosis
      304          160          127
Coronary heart disease      Crohn's disease      HDL cholesterol
      124          121          118
      Ulcerative colitis      Bipolar disorder      LDL cholesterol
      106          105          102
      Triglycerides
      98
```

For a given trait, obtain a GRanges with all recorded associations; here only three associations are shown:

```
> subsetByTraits(gwrngs, tr="LDL cholesterol")[1:3]
```

gwasloc instance with 3 records and 34 attributes per record.

Extracted: 2012.03.20

Excerpt:

GRanges with 3 ranges and 3 elementMetadata cols:

```

      seqnames      ranges strand | Disease.Trait      SNPs
      <Rle>      <IRanges> <Rle> | <character> <character>
[1] chr2 [20903015, 20903015] * | LDL cholesterol rs4971516
[2] chr2 [20903015, 20903015] * | LDL cholesterol rs4971516
[3] chr6 [16197194, 16197194] * | LDL cholesterol rs2142672
      p.Value
      <numeric>
[1] 2e-40
[2] 2e-52
```

```

[3]      2e-08
---
seqlengths:
  chr1 chr10 chr11 chr12 chr13 chr14 ... chr4 chr5 chr6 chr7 chr8 chr9
    NA   NA   NA   NA   NA   NA ...   NA   NA   NA   NA   NA   NA

```

2 SNP sets and trait sets

2.1 SNPs by name

We can regard the content of a SNP chip as a set of SNP, referenced by name. The `pd.genomewidesnp.6` package describes the Affymetrix SNP 6.0 chip. We can determine which traits are associated with loci interrogated by the chip as follows. We work with a subset of the 1 million loci for illustration.

The `locon6` data frame has information on 10000 probes, acquired through the following code (not executed here to reduce dependence on the `pd.genomewidesnp.6` package, which is very large).

```

> library(pd.genomewidesnp.6)
> con = pd.genomewidesnp.6@getdb()
> locon6 = dbGetQuery(con,
+   "select dbsnp_rs_id, chrom, physical_pos from featureSet limit 10000")

```

Instead use the serialized information:

```

> data(locon6)
> rson6 = as.character(locon6[[1]])
> rson6[1:5]
[1] "rs2887286" "rs1496555" "rs41477744" "rs3890745" "rs10492936"

```

We subset the GWAS ranges structure with rsids that are common to both the chip and the GWAS catalog. We then tabulate the diseases associated with the common loci.

```

> intr = gwrngs[ intersect(getRsids(gwrngs), rson6) ]
> sort(table(getTraits(intr)), decreasing=TRUE)[1:10]

```

Select biomarker traits	Height
3	2
Metabolic traits	Schizophrenia
2	2
Ulcerative colitis	Age-related macular degeneration
2	1
Aging traits	Alcohol dependence
1	1
Alzheimer's disease	Alzheimer's disease (late onset)
1	1

2.2 Traits by genomic location

We will assemble genomic coordinates for SNP on the Affymetrix 6.0 chip and show the effects of identifying the trait-associated loci with regions of width 1000bp instead of 1bp.

The following code retrieves coordinates for SNP interrogated on 10000 probes (to save time) on the 6.0 chip, and stores the results in a GRanges instance.

```
> gr6.0 = GRanges(seqnames=ifelse(is.na(locon6$chrom),0,locon6$chrom),
+               IRanges(ifelse(is.na(locon6$phys),1,locon6$phys), width=1))
> elementMetadata(gr6.0)$rsid = as.character(locon6$db SNP_rs_id)
> seqlevels(gr6.0) = paste("chr", seqlevels(gr6.0), sep="")
```

Here we compute overlaps with both the raw disease-associated locus addresses, and with the locus address ± 500 bp.

```
> ag = function(x) as(x, "GRanges")
> ovraw = subsetByOverlaps(ag(gwrngs), gr6.0)
> length(ovraw)
```

```
[1] 54
```

```
> ovaug = subsetByOverlaps(ag(gwrngs+500), gr6.0)
> length(ovaug)
```

```
[1] 78
```

To acquire the subset of the catalog to which 6.0 probes are within 500bp, use:

```
> rawrs = elementMetadata(ovraw)$SNPs
> augrs = elementMetadata(ovaug)$SNPs
> gwrngs[augrs]
```

gwasloc instance with 78 records and 34 attributes per record.

Extracted: 2012.03.20

Excerpt:

GRanges with 5 ranges and 3 elementMetadata cols:

	seqnames	ranges	strand	
	<Rle>	<IRanges>	<Rle>	
[1]	chr10	[64580575, 64580575]	*	
[2]	chr1	[70335682, 70335682]	*	
[3]	chr1	[84865230, 84865230]	*	
[4]	chr10	[49985110, 49985110]	*	
[5]	chr1	[167903079, 167903079]	*	

Disease.Trait

SNPs

p.Value

		<character>	<character>	<numeric>
[1]	Ewing sarcoma	rs224278		4e-17
[2]	Hypertension risk in short sleep duration	rs2226284		3e-08
[3]	Response to hepatitis C treatment	rs12144715		7e-06
[4]	Response to antidepressant treatment	rs10857636		2e-07
[5]	Schizophrenia	rs10489202		1e-08

seqlengths:

chr1	chr10	chr11	chr12	chr13	chr14	...	chr4	chr5	chr6	chr7	chr8	chr9
NA	NA	NA	NA	NA	NA	...	NA	NA	NA	NA	NA	NA

Relaxing the intersection criterion in this limited case leads to a larger set of traits.

```
> setdiff( getTraits(gwrngs[augrs]), getTraits(gwrngs[rawrs]) )
```

```
[1] "Response to hepatitis C treatment"
[2] "Response to antidepressant treatment"
[3] "Bipolar disorder"
[4] "Phospholipid levels (plasma)"
[5] "Endometrial cancer"
[6] "Neuroblastoma"
[7] "MRI atrophy measures"
[8] "Menarche (age at onset)"
[9] "Self-rated health"
[10] "Neonatal lupus"
[11] "Crohn's disease"
[12] "Optic disc size (cup)"
[13] "Response to statin therapy"
[14] "Tanning"
[15] "Obesity"
[16] "Osteonecrosis of the jaw"
[17] "Hip geometry"
[18] "Parkinson's disease"
```

3 Counting alleles associated with traits

We can use `riskyAlleleCount` to count risky alleles enumerated in the GWAS catalog. This particular function assumes that we have genotyped at the catalogued loci. Below we will discuss how to impute from non-catalogued loci to those enumerated in the catalog.

```
> data(gg17N) # translated from GGdata chr 17 calls using ABmat2nuc
> gg17N[1:5,1:5]
```

	rs6565733	rs1106175	rs17054921	rs8064924	rs8070440
NA06985	"G/G"	"A/G"	"C/C"	"G/G"	"G/G"
NA06991	"G/G"	"A/A"	"C/C"	"G/G"	"G/G"
NA06993	"G/G"	"A/A"	"C/C"	"G/G"	"G/G"
NA06994	"A/G"	"A/G"	"C/C"	"A/G"	"G/G"
NA07000	"G/G"	"A/A"	"C/C"	"G/G"	"G/G"

This function can use genotype information in the A/B format, assuming that B denotes the alphabetically later nucleotide. Because we have direct nucleotide coding in our matrix, we set the `matIsAB` parameter to false in this call.

```
> h17 = riskyAlleleCount(gg17N, matIsAB=FALSE, chr="ch17")
> h17[1:5,1:5]
```

	rs7217319	rs12150338	rs1231206	rs216172	rs10852932
NA06985	0	0	1	1	1
NA06991	0	0	0	0	2
NA06993	0	0	1	1	1
NA06994	0	0	2	2	0
NA07000	0	0	2	2	0

```
> table(as.numeric(h17))
```

```
0    1    2
8190 3345 2235
```

It is of interest to bind the counts back to the catalog data.

```
> gwr = gwrngs
> gwr = gwr[colnames(h17),]
> elementMetadata(gwr) = cbind(elementMetadata(gwr), DataFrame(t(h17)))
> sn = rownames(h17)
> gwr[,c("Disease.Trait", sn[1:4])]
```

gwasloc instance with 153 records and 5 attributes per record.

Extracted: 2012.03.20

Excerpt:

GRanges with 5 ranges and 5 elementMetadata cols:

	seqnames	ranges	strand	Disease.Trait	NA06985
	<Rle>	<IRanges>	<Rle>	<character>	<integer>
[1]	chr17	[38924, 38924]	*	AIDS progression	0
[2]	chr17	[1634104, 1634104]	*	Serum calcium	0
[3]	chr17	[2125605, 2125605]	*	Coronary heart disease	1
[4]	chr17	[2126504, 2126504]	*	Coronary heart disease	1

```

[5]      chr17 [2143460, 2143460]      * |      Aortic root size      1
      NA06991      NA06993      NA06994
      <integer> <integer> <integer>
[1]          0          0          0
[2]          0          0          0
[3]          0          1          2
[4]          0          1          2
[5]          2          1          0
---
seqlengths:
  chr1 chr10 chr11 chr12 chr13 chr14 ... chr4 chr5 chr6 chr7 chr8 chr9
    NA   NA   NA   NA   NA   NA ...   NA   NA   NA   NA   NA   NA

```

Now by programming on the `elementMetadata`, we can identify individuals with particular risk profiles.

4 Imputation to unobserved loci

If we lack information on a specific locus s , but have reasonably dense genotyping on a subject, population genetics may allow a reasonable guess at the genotype at s for this subject. Many algorithms for genotype imputation have been proposed. Here we use a very simple approach due to David Clayton in the *snpStats* package.

We use the “low coverage” 1000 genomes genotypes for the CEU (central European) HapMap cohort as a base for constructing imputation rules. We focus on chromosome 17 for illustration.

The base data are

```

> data(low17)
> low17

A SnpMatrix with 60 rows and 196327 columns
Row names: NA06985 ... NA12874
Col names: chr17:1869 ... chr17:78654554

```

A somewhat sparser set of genotypes (HapMap phase II, genomewide 4 million loci) on chromosome 17 is archived as `g17SM`. This has a compact `SnpMatrix` encoding of genotypes.

```

> data(g17SM)
> g17SM

A SnpMatrix with 90 rows and 89701 columns
Row names: NA06985 ... NA12892
Col names: rs6565733 ... rs4986109

```


For a realistic demonstration, we use the subset of these loci that are present on the Affy 6.0 SNP array.

```
> data(gw6.rs_17)
> g17SM = g17SM[, intersect(colnames(g17SM), gw6.rs_17)]
> dim(g17SM)
```

```
[1]    90 20359
```

The base data were used to create a set of rules allowing imputation from genotypes in the sparse set to the richer set. Some rules involve only a single locus, some as many as 4. The construction of rules involves tuning of modeling parameters. See `snp.imputation` in `snpStats` for details.

```
> data(rules_6.0_1kg_17)
> rules_6.0_1kg_17[1:5,]
```

```
chr17:1869 ~ rs9915268+rs11247571+rs9895105+rs6598837 (MAF = 0.06666667, R-squared = 
chr17:2220 ~ rs4790867+rs10454094+rs2586238+rs7207284 (MAF = 0.125, R-squared = 0.706
chr17:6689 ~ rs4424950+rs4790867+rs7225087+rs11658347 (MAF = 0.125, R-squared = 0.592
rs34663111 ~ rs11658079+rs1609550+rs4985594+rs9788983 (MAF = 0.1166667, R-squared = 0
rs62054999 ~ rs17609440+rs2740351+rs2589492+rs16956017 (MAF = 0.125, R-squared = 0.26
```

The summary of rules shows the degree of association between the predictors and predictands in terms of R^2 . Many potential targets are not imputed.

```
> summary(rules_6.0_1kg_17)
```

	SNPs used				
R-squared	1 tags	2 tags	3 tags	4 tags	<NA>
[0,0.1)	655	785	276	56	0
[0.1,0.2)	7	664	926	868	0
[0.2,0.3)	0	158	916	3054	0
[0.3,0.4)	0	28	411	5104	0
[0.4,0.5)	0	20	203	6365	0
[0.5,0.6)	0	21	121	6052	0
[0.6,0.7)	0	29	104	5623	0
[0.7,0.8)	0	54	108	6330	0
[0.8,0.9)	0	141	225	9506	0
[0.9,0.95)	652	700	572	8056	0
[0.95,0.99)	7274	1689	1388	6158	0
[0.99,1]	33660	1353	2326	10152	0
<NA>	0	0	0	0	53601

The overlap between the 6.0-resident g17SM loci and the catalog is

```
> length(intersect(colnames(g17SM), values(gwrngs)$SNPs))
```

```
[1] 65
```

The new expected B allele counts are

```
> exg17 = impute.snps(rules_6.0_1kg_17, g17SM)
```

The number of new loci that coincide with risk loci in the catalog is:

```
> length(intersect(colnames(exg17), values(gwrngs)$SNPs))
```

```
[1] 89
```

5 Formalizing disease traits with Disease Ontology

The Disease Ontology project [?] formalizes a vocabulary for human diseases. Bioconductor's DO.db package is a curated representation.

```
> library(DO.db)
```

```
> DO()
```

Quality control information for DO:

This package has the following mappings:

DOANCESTOR has 6241 mapped keys (of 6242 keys)

DOCHILDREN has 1775 mapped keys (of 6242 keys)

DOOBSOLETE has 2365 mapped keys (of 2365 keys)

DOOFFSPRING has 1775 mapped keys (of 6242 keys)

DOPARENTS has 6241 mapped keys (of 6242 keys)

DOTERM has 6242 mapped keys (of 6242 keys)

Additional Information about this package:

DB schema: DO_DB

DB schema version: 1.0

All tokens of the ontology are acquired via:

```
> alltob = unlist(mget(mappedkeys(DOTERM), DOTERM))
```

```
> allt = sapply(alltob, Term)
```

```
> allt[1:5]
```

D0ID:0000000	D0ID:0000405	D0ID:0001816
"gallbladder disease"	"vascular tissue disease"	"angiosarcoma"
D0ID:0002116	D0ID:0014667	
"pterygium"	"disease of metabolism"	

Direct mapping from disease trait tokens in the catalog to this vocabulary succeeds for a modest proportion of records.

```
> cattra = elementMetadata(gwrngs)$Disease.Trait
> mat = match(tolower(cattra), tolower(allt))
> catDO = names(allt)[mat]
> catDO[1:50]
```

```
[1] NA          NA          "D0ID:585"   "D0ID:585"   "D0ID:585"
[6] NA          "D0ID:10763" "D0ID:10763" "D0ID:10763" NA
[11] NA          NA          NA          NA          NA
[16] "D0ID:1612" "D0ID:1612" "D0ID:1612" NA          NA
[21] NA          NA          NA          NA          NA
[26] NA          NA          NA          NA          NA
[31] NA          NA          NA          NA          NA
[36] NA          NA          NA          NA          "D0ID:3121"
[41] "D0ID:3121" "D0ID:3121" "D0ID:3121" "D0ID:3121" NA
[46] NA          NA          "D0ID:3393" "D0ID:3393" "D0ID:3393"
```

```
> mean(is.na(catDO))
```

```
[1] 0.73271
```

Approximate matching of unmatched tokens can proceed by various routes. Some traits are not diseases, and will not be mappable. However, consider

```
> unique(cattra[is.na(catDO)])[1:20]
```

```
[1] "Duodenal ulcer "
[2] "Response to statin therapy"
[3] "Vitamin B12 levels"
[4] "Body mass index"
[5] "Cortical structure"
[6] "Facial morphology"
[7] "Cardiac repolarization"
[8] "Response to statin therapy (LDL-C)"
[9] "Ewing sarcoma"
[10] "Hypertension risk in short sleep duration"
[11] "Treatment response for severe sepsis "
```

```

[12] "Stroke"
[13] "Infantile hypertrophic pyloric stenosis"
[14] "Type 1 diabetes"
[15] "Type 2 diabetes"
[16] "Glycated hemoglobin levels"
[17] "Lipid metabolism phenotypes"
[18] "Serum metabolites"
[19] "Lymphocyte counts"
[20] "Inflammatory biomarkers"

```

```
> nomatch = cattra[is.na(catDO)]
```

Manual searching shows that a number of these have very close matches.

6 Appendix: Adequacy of location annotation

A basic question concerning the use of archived SNP identifiers is durability of the association between asserted location and SNP identifier. The `chklocs` function uses a current Bioconductor `SNPlocs` package to check this.

For example, to verify that locations asserted on chromosome 20 agree between the Bioconductor dbSNP image and the gwas catalog,

```
> if ("SNPlocs.Hsapiens.dbSNP.20110815" %in% installed.packages()[,1])
+ suppressWarnings(chklocs("20"))
```

```
[1] TRUE
```

This is not a fast procedure but has succeeded for all chromosomes 1-22 when checked off line.