

Revised problems, selected solutions and additional  
material:  
ISMB2011 tutorial:  
genetics of gene expression with R/bioc

VJ Carey

October 11, 2011

## Contents

<b>1</b>	<b>Selected problems from the main notes, with some extensions</b>	<b>2</b>
<b>2</b>	<b>FDR and MAF</b>	<b>18</b>
<b>3</b>	<b>Master regulators in cis?</b>	<b>20</b>
<b>4</b>	<b>Session information</b>	<b>24</b>

# 1 Selected problems from the main notes, with some extensions

- Manual computation of `gwSnpTests` results. The basic result of interest is

```
> library(GGtools)
> library(ggtut)
```

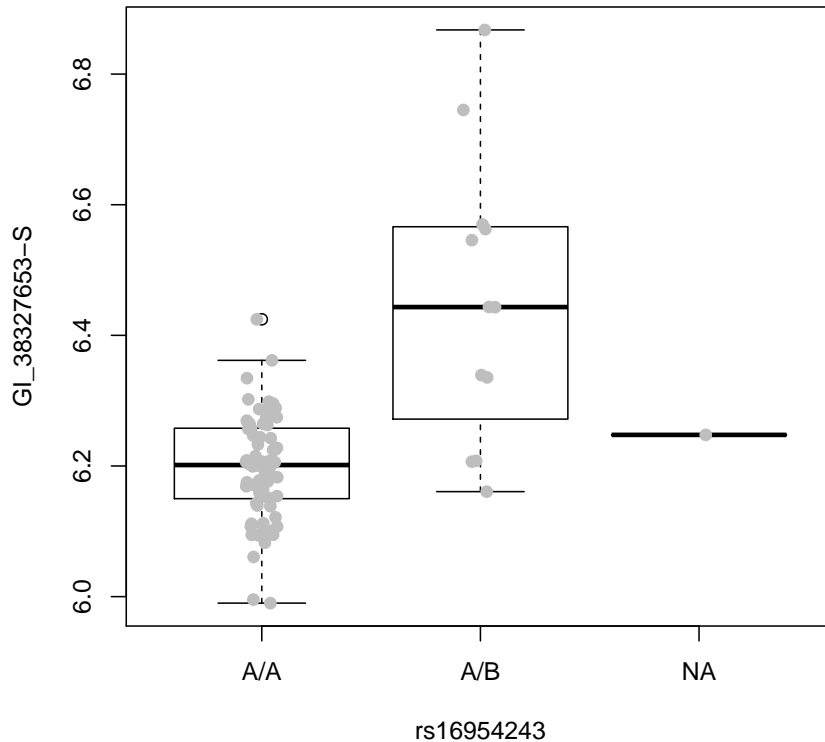
To get a tailored `smlSet`, use `getSS("hmyriB36", [chrvec])`  
available chromosomes are named 1, 10, ..., X, Y

```
> c17 = getSS("GGdata", "17", renameChrs = "chr17")
> t1 = gwSnpTests(genesym("CHRNE") ~ male, c17, chrnum("chr17"))
> topSnps(t1)
```

	p.val
rs16954243	2.925728e-09
rs7214776	7.564315e-09
rs8081611	7.564315e-09
rs2302321	4.838786e-08
rs8070572	2.505861e-07
rs7225684	4.087961e-07
rs2243093	8.156751e-07
rs16954257	9.001695e-07
rs2243100	9.273871e-07
rs8077875	1.474871e-06

The data configuration giving rise to the strongest association test is

```
> plot_EvG(probeId("GI_38327653-S"), rsid("rs16954243"), c17)
```



Of note is the sparsity of data on the A/B genotype.

To obtain the expression data serving as dependent variable,

```
> chrneExpr = as.numeric(exprs(c17[genesym("CHRNE"), ]))
```

The count of B alleles is obtained as

```
> num243 = as(smList(c17)[["chr17"]][, "rs16954243"], "numeric")
```

A simple approach to testing the additive genetic model with adjustment for gender is

```
> lm1 = lm(chrneExpr ~ num243 + male, data = pData(c17))
> summary(lm1)
```

Call:

```
lm(formula = chrneExpr ~ num243 + male, data = pData(c17))
```

Residuals:

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

-0.28055 -0.04835 -0.00189 0.06057 0.40423

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	6.18678	0.01639	377.553	< 2e-16 ***
num243	0.25449	0.03326	7.651	2.68e-11 ***
maleTRUE	0.02210	0.02274	0.972	0.334

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1072 on 86 degrees of freedom

(1 observation deleted due to missingness)

Multiple R-squared: 0.4097, Adjusted R-squared: 0.396

F-statistic: 29.84 on 2 and 86 DF, p-value: 1.434e-10

The computations in `gwSnpTests` occur via `snp.rhs.tests` in the `snpMatrix` package:

```
> c17$chrneExpr = chrneExpr
> lm2 = snp.rhs.tests(chrneExpr~male,
+ snp.data=smList(c17)[[1]], data= pData(c17),
+ family="gaussian")
> lm2["rs16954243",]
```

	Chi.squared	Df	p.value
rs16954243	35.23268	1	2.925728e-09

Comments:

- The least-squares based p-value and the `snp.rhs.tests` p-values differ as the latter is based on the score test of the additive genetic effect after adjustment for covariates.
- We have disregarded familial structure in the CEU data. One approach to coping with this involves a cluster-adjusted robust variance estimate in the score test, but in relatively small samples this seems to result in severe loss of power.

```
> table(table(c17$famid))
 3  6
10 10
> lm3 = snp.rhs.tests(chrneExpr~male+cluster(famid),
+ snp.data=smList(c17)[[1]], data=
+ pData(c17), family="gaussian")
> lm3["rs16954243",]
```

```

                Chi.squared Df    p.value
rs16954243      7.841172  1 0.00510695

```

If instead we allow a random effect of family, we have

```

> library(nlme)
> lme1 = lme(chrneExpr~num243+male, random=~1|famid,
+ data=pData(c17), na.action=na.omit)
> summary(lme1)

```

Linear mixed-effects model fit by REML

```

Data: pData(c17)
      AIC      BIC  logLik
-120.7854 -108.5136 65.39268

```

Random effects:

```

Formula: ~1 | famid
      (Intercept) Residual
StdDev:  0.02655194 0.1038747

```

Fixed effects: chrneExpr ~ num243 + male

```

                Value Std.Error DF  t-value p-value
(Intercept) 6.185640 0.01705465 67 362.6952 0.0000
num243      0.259045 0.03281251 67  7.8947 0.0000
maleTRUE    0.023200 0.02210860 67  1.0493 0.2978

```

Correlation:

```

      (Intr) num243
num243  -0.247
maleTRUE -0.621 -0.014

```

Standardized Within-Group Residuals:

```

                Min      Q1      Med      Q3      Max
-2.39631647 -0.58167247  0.05750796  0.57956215  3.67920381

```

Number of Observations: 89

Number of Groups: 20

Other things being equal, the best way for dealing with familial correlation in this setting is to use an appropriate variance components model like that afforded by `lme`. We do not address this issue further, and have ignored the correlation to illustrate mechanics of analysis.

- Counting SNP that are monomorphic in the cohort.

```

> cs17 = col.summary(smList(c17)[[1]])
> dim(cs17)

```

```

[1] 89701      9
> cs17[1:5, ]
      Calls Call.rate Certain.calls      RAF      MAF      P.AA
rs6565733      90          1          1 0.8722222 0.12777778 0.01111111
rs1106175      90          1          1 0.1833333 0.18333333 0.67777778
rs17054921     90          1          1 0.9888889 0.01111111 0.00000000
rs8064924      90          1          1 0.8777778 0.12222222 0.00000000
rs8070440      90          1          1 0.9444444 0.05555556 0.00000000
      P.AB      P.BB      z.HWE
rs6565733 0.23333333 0.75555556 0.4439974
rs1106175 0.27777778 0.04444444 -0.6864313
rs17054921 0.02222222 0.97777778 0.1065936
rs8064924 0.24444444 0.75555556 1.3209514
rs8070440 0.11111111 0.88888889 0.5580490
> sum(cs17[, "MAF"] == 0)
[1] 28734

```

- Checking a finding in another population; sensitivity analysis. We supply the analogous expression and genotype information on the Yoruba cohort in the `hmyriB36` package.

```

> y17 = getSS("hmyriB36", "17", renameChrs = "chr17")
> y1 = gwSnpTests(genesym("CHRNE") ~ male, y17, chrnum("chr17"))
> topSnps(y1)
      p.val
rs9889685 0.0001236433
rs7212518 0.0001347252
rs9907560 0.0001406747
rs6501801 0.0001413512
rs2024498 0.0001431966
rs8069187 0.0001431966
rs9891980 0.0001431966
rs9303366 0.0001431966
rs1917997 0.0001461397
rs16974160 0.0001680648

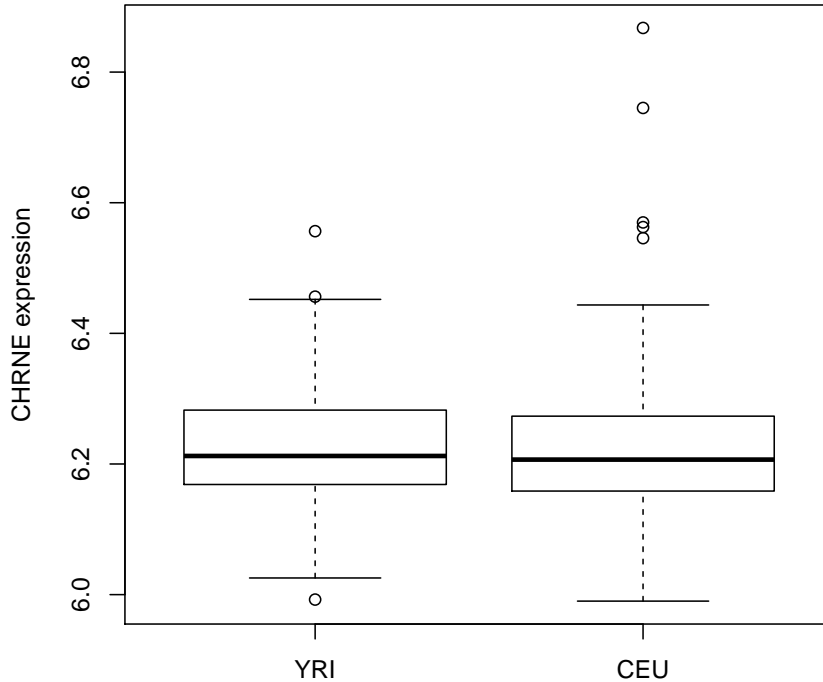
```

We check to see if the marginal distributions of expression in the two populations are similar:

```

> ych = exprs(y17[genesym("CHRNE"), ])
> cch = exprs(c17[genesym("CHRNE"), ])
> boxplot(list(YRI = ych, CEU = cch), ylab = "CHRNE expression")

```



These are reasonably similar distributions – there is a little more dynamic range for this gene in CEU. So we can't readily argue that the expression measures in YRI are somehow compromised for this gene, depressing power. A useful exercise would be to assess the expression measures in CEU for outlying values and check the sensitivity of the eQTL inference to these values. Winsorization might be considered.

```
> library(parody)
> calout.detect(chrneExpr, method = "GESD")

$ind
[1] 63 7 52 44 49

$val
[1] 6.867608 6.745111 6.569859 6.562725 6.545770
```

This procedure uses a Gaussian reference model and outward testing to objectively reason about the existence of outlying observations. We can repeat our manual test of association simply excluding the outliers

```
> summary(update(lm1, subset = chrneExpr < 6.5))
```

```
Call:
lm(formula = chrneExpr ~ num243 + male, data = pData(c17), subset = chrneExpr
    6.5)
```

```
Residuals:
      Min       1Q   Median       3Q      Max
-0.210840 -0.055186  0.006141  0.063266  0.223664
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.200913   0.012848 482.630 < 2e-16 ***
num243       0.106438   0.032817   3.243  0.00172 **
maleTRUE    -0.007301   0.018187  -0.401  0.68917
---

```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.08264 on 81 degrees of freedom
(1 observation deleted due to missingness)
Multiple R-squared:  0.1204,    Adjusted R-squared:  0.09872
F-statistic: 5.545 on 2 and 81 DF,  p-value: 0.005532
```

This shows the CEU result to be quite sensitive to the extreme expression values achieved for the heterozygous individuals. Another sensitivity analysis is Winsorization (see section 3.2.1 of V. Barnett and T. Lewis, *Outliers in statistical data*, for a discussion of Winsorization for mean estimation; Winsorization of covariates is not widely studied.)

```
> winch = chrneExpr
> winch[winch >= 6.5] = 6.5
> summary(update(lm1, winch ~ .))
```

```
Call:
lm(formula = winch ~ num243 + male, data = pData(c17))
```

```
Residuals:
      Min       1Q   Median       3Q      Max
-0.22249 -0.05158  0.00559  0.06645  0.23023
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.194349   0.013435 461.044 < 2e-16 ***
num243       0.188876   0.027274  6.925 7.44e-10 ***
maleTRUE     0.006361   0.018641  0.341  0.734
---

```



Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.08787 on 86 degrees of freedom  
(1 observation deleted due to missingness)

Multiple R-squared: 0.3589, Adjusted R-squared: 0.344  
F-statistic: 24.07 on 2 and 86 DF, p-value: 4.984e-09

The lesson is that sensitivity analysis by excluding potentially aberrant values can be very costly; softer approaches are feasible, and in this case suggest that sensitivity of the eQTL finding to achieved high values of CHRNE expression in CEU may not be so great.

For now we conclude that data from the YRI cohort seem provide no evidence of an eQTL for CHRNE, in contrast to the CEU cohort.

– **Approximate FDR for distance-delimited tests (original notes exercise, p. 18).**

We saved the results of `cisProxScores` applied to the surveys before and after permutation. We can obtain the collections of delimited scores as follows:

```
> data(CPS17)
> data(PERMCP17)
> sb1 = scoresByGenes(CPS17, as.GRanges = FALSE)
> permsb1 = scoresByGenes(PERMCP17, as.GRanges = FALSE)
> summary(sapply(sb1, length))
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.0	52.0	71.0	86.3	96.0	696.0

Let's focus on the strongest association observed per gene within 50kb of its coding region. The permutation distribution has quantiles

```
> targp = c(0.95, 0.975, 0.99, 0.995)
> pq = quantile(sapply(permsb1, max), targp)
> pq
```

95%	97.5%	99%	99.5%
9.95100	11.27950	12.88570	13.56545

leading to counts of false eQTL claims

```
> fp = sapply(pq, function(x) sum(sapply(permsb1, max) > x))
> fp
```

95%	97.5%	99%	99.5%
24	12	5	3

The numbers of real tests passing these thresholds are

```
> op = sapply(pq, function(x) sum(sapply(sb1, max) > x))
> op
```

95%	97.5%	99%	99.5%
113	88	70	62

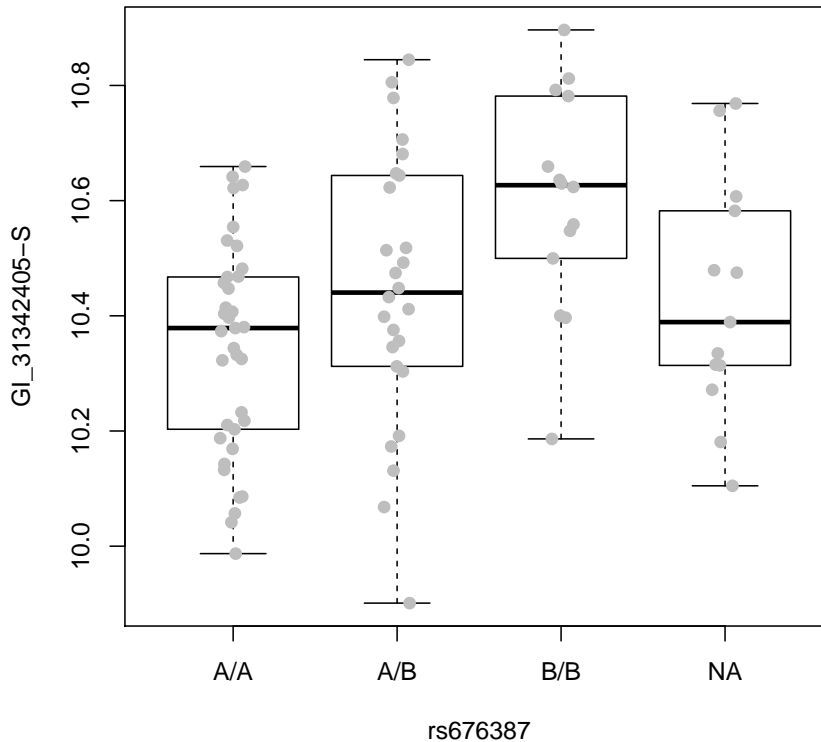
so a table of thresholds and approximate FDRs is

```
> fdrmat = cbind(pctile.null = targp * 100, thresh = pq, nfalse = fp,
+   nsig = op, fdr = fp/op)
> fdrmat
```

	pctile.null	thresh	nfalse	nsig	fdr
95%	95.0	9.95100	24	113	0.21238938
97.5%	97.5	11.27950	12	88	0.13636364
99%	99.0	12.88570	5	70	0.07142857
99.5%	99.5	13.56545	3	62	0.04838710

So out of our 498 genes on chromosome 17, we assert existence of 62 eQTL within 50kb at an FDR of about 5%. Let's look at the data configuration for the statistically weakest claim.

```
> at5p = which(sapply(sb1,max)>13.565) # find genes with
>                                     # sufficient scores
> sort(sapply(sb1[at5p],max))[1:5] # look at the weakest
GI_31342405-S  GI_4557308-S  GI_21687219-S  GI_4759187-S  GI_41327725-S
      13.59           13.83           13.90           13.99           14.25
>                                     # find strongest SNP
> sort(sb1[["GI_31342405-S"]], decreasing=TRUE)[1:5]
rs676387 rs10454087   rs6963   rs646123   rs665268
      13.59       12.97       12.76       12.18       12.18
> plot_EvG(probeId("GI_31342405-S"), rsid("rs676387"), c17)
```



Not bad. Does it hold up in YRI?

- **Distinguishing variants in exons (continuation of problem on p. 18 of original notes).** The task here is to acquire the locations of the SNPs of interest. We'll focus on the highest-scoring SNP for each gene, restricting attention to associations with  $FDR < 5\%$ .

Retrieve the names of these most associated SNPs:

```
> snpsat5 = sapply(lapply(sb1[at5p], sort, decreasing=TRUE),
+   function(x)names(x[1]))
```

Get their locations:

```
> data(snpgr17)
> snpgr17[1:3]
```

GRanges with 3 ranges and 0 elementMetadata values:

	seqnames	ranges	strand
	<Rle>	<IRanges>	<Rle>
rs1106176	chr17	[6934, 6934]	*
rs6420494	chr17	[7214, 7214]	*
rs6420495	chr17	[7242, 7242]	*

```

---
seqlengths:
  chr17
    NA

```

Now find the proportion of these SNPs that are in exons:

```

> library(TxDb.Hsapiens.UCSC.hg18.knownGene)
> tx17 = TxDb.Hsapiens.UCSC.hg18.knownGene
> ex17 = exons(tx17, vals=list(exon_chrom="chr17"))
> mean(snpgr17[snpsat5] %in% ex17)

[1] 0.2903226

```

The not-yet-released VariantAnnotation package can provide more information. We need to provide allele details for our variants, which we have not recorded or retained. However, since these are documented SNP, we have access to the reference assignments.

We will use the hg18-associated dbSNP data:

```

> library(SNPlocs.Hsapiens.dbSNP.20090506)
> s117 = getSNPlocs("chr17")

```

We need to acquire the IUPAC codes for the SNPs of interest.

```

> AT5 = snpgr17[snpsat5]
> Nat5 = names(AT5)
> Nat5f = gsub("rs", "", Nat5)
> sig17 = s117[match(Nat5f, s117[, 1]), ]
> icodes = IUPAC_CODE_MAP[sig17[, "alleles_as_ambig"]]
> icodes[1:5]

```

```

  S   Y   M   K   R
"CG" "CT" "AC" "GT" "AG"

```

We will arbitrarily assign these as ref and variant for input to the predictCoding function of VariantAnnotation:

```

> c1 = sapply(strsplit(icodes, ""), "[", 1)
> c2 = sapply(strsplit(icodes, ""), "[", 2)
> elementMetadata(AT5)$ref = c1
> elementMetadata(AT5)$var = c2

```

We need the hg18 sequence:

```

> library(BSgenome.Hsapiens.UCSC.hg18)

```

Finally, we obtain the transcript database and call the predictCoding function to see which of the significant eQTL correspond to nonsynonymous modifications.

```

> library(VariantAnnotation)
> predictCoding(AT5, tx17, Hsapiens, "ref", "var")

DataFrame with 7 rows and 10 columns
  queryHits      txID      refSeq      varSeq      refAA
  <integer> <character> <DNAStrngSet> <DNAStrngSet> <AAStringSet>
1         22      52441          ACT          ACC          T
2         27      54786          CAG          CAG          Q
3         27      54787          CAG          CAG          Q
4         27      54788          CAG          CAG          Q
5         44      55846          CGG          CAG          R
6         49      52805          GTA          GAA          V
7         49      52807          GTA          GAA          V

  varAA      Consequence      cds_id      cds_name      exon_rank
  <AAStringSet>      <factor> <integer> <character> <integer>
1           T      synonymous      176944          NA          2
2           Q      synonymous      176945          NA          3
3           Q      synonymous      176946          NA          4
4           Q      synonymous      176947          NA          5
5           Q nonsynonymous      176948          NA          6
6           E nonsynonymous      176949          NA          7
7           E nonsynonymous      176950          NA          8

```

– **Systematic assessment of effect of reducing expression heterogeneity via PCA (p.28)**

We used `gwSnpTests` to illustrate, for one gene, the impact of employing principal components as ‘adjustments’ to eQTL test procedures. The motivations are detailed in papers by O. Stegle, Jeffrey Leek et al. Here we show how to introduce this approach in a chromosome-wide survey.

We obtain the expression and genotype data, compute PCs on the expression matrix, and then reduce the expression set to the filtered 498 genes.

```

> c17 = getSS("GGdata", "17", renameChrs="chr17")
> pct = prcomp(t(exprs(c17)))
> o17 = observed17ceu()
> pm = probesManaged(o17,1) # names of filtered probes
> c17f = c17[probeId(pm),] # subset

```

We add the first four PC as ‘phenoData’ to the `smlSet` instance.

```

> c17f$PC1 = pct$x[, 1]
> c17f$PC2 = pct$x[, 2]
> c17f$PC3 = pct$x[, 3]
> c17f$PC4 = pct$x[, 4]

```

On a 2-core MacBook Pro, the following takes only a few moments.

```

> library(multicore)
> if (!exists("eqtpc")) eqtpc = eqtlTests(c17f, ~male + PC1 + PC2 +
+   PC3 + PC4, targdir = "pc4", shortfac = 10, genegran = 2,
+   geneApply = mclapply)

```

Obtain the vector of best per-gene eQTL scores:

```

> adjtop = mclapply(pm,
+   function(x)topFeats(probeId(x), mgr=eqtpc, ffind=1))
> names(adjtop) = pm
> adjmax = sapply(adjtop, max)

```

Repeat the process for one permutation of expression against genotype:

```

> set.seed(1234);
> if (!exists("eqtpcPERM"))
+ eqtpcPERM = eqtlTests(permEx(c17f),
+   ~male+PC1+PC2+PC3+PC4, targdir="pc4_perm",
+   shortfac=10, genegran=2, geneApply=mclapply)
> adjtop_PERM = mclapply(pm,
+   function(x)topFeats(probeId(x), mgr=eqtpcPERM, ffind=1))
> names(adjtop_PERM) = pm
> adjmax_PERM = sapply(adjtop_PERM, max)

```

Now we tabulate the significance claims:

```

> pcts = c(0.95, 0.975, 0.99, 0.995)
> permq = quantile(adjmax_PERM, pcts)
> nfals = sapply(permq, function(x) sum(adjmax_PERM > x))
> nsig = sapply(permq, function(x) sum(adjmax > x))
> options(digits = 3)
> cbind(PPCT = 100 * pcts, thresh = permq, nfals = nfals, nsig = nsig,
+   fdr = nfals/nsig)

```

	PPCT	thresh	nfals	nsig	fdr
95%	95.0	23.4	25	46	0.543
97.5%	97.5	26.5	13	29	0.448
99%	99.0	29.3	5	23	0.217
99.5%	99.5	30.8	3	21	0.143

Compare to the unadjusted analysis:

```

> o17 = observed17ceu()
> p17 = onePerm17ceu()
> unadjtop_PERM = mclapply(pm, function(x) topFeats(probeId(x),
+   mgr = p17, ffind = 1))
> unadjmax_PERM = sapply(unadjtop_PERM, max)
> unadjtop = mclapply(pm, function(x) topFeats(probeId(x), mgr = o17,

```

```

+     ffind = 1))
> unadjmax = sapply(unadjtop, max)
> pcts = c(0.95, 0.975, 0.99, 0.995)
> permq = quantile(unadjmax_PERM, pcts)
> nfals = sapply(permq, function(x) sum(unadjmax_PERM > x))
> nsig = sapply(permq, function(x) sum(unadjmax > x))
> options(digits = 3)
> cbind(PPCT = 100 * pcts, thresh = permq, nfals = nfals, nsig = nsig,
+       fdr = nfals/nsig)

```

	PPCT	thresh	nfals	nsig	fdr
95%	95.0	23.5	25	45	0.556
97.5%	97.5	26.3	13	30	0.433
99%	99.0	30.7	5	22	0.227
99.5%	99.5	32.2	3	20	0.150

We see that this very simple adjustment may substantially affect power of this approach to eQTL detection. Application of SVA for a systematic survey seems more laborious. The PEER method of Stegle et al. should also be investigated.

– **Working with trans tests.**

On p. 33 the exercise is: Find a locus giving a  $\chi^2(1)$  statistic exceeding 11.8 for association with FDPS; for this locus FDPS is the fourth-largest association score. Display the expression-vs- genotype plot. How would you perform a more comprehensive check for FDPS without restricting to  $MAF > 10\%$ ?

We are given

```

> tr17 = tr17_1_9()
> tr17

```

transManager instance, created Tue Oct 11 10:24:22 2011

dimension of scores component:

number of loci checked: 44638; genes retained: 20

the call was:

```

transScores(smpack = "GGdata", snpchr = "17", rhs = ~male, targdirpref = "twfi
geneApply = mclapply, chrnames = c("chr1", "chr9"), wrapperEndo = function
low = 0.1))

```

```

> getClass(class(tr17))

```

Class "transManager" [package "GGtools"]

Slots:

Name: base

Class: list

```

> names(tr17@base)
[1] "scores"  "inds"    "guniv"   "K"       "smsanno" "snpsnames" "call"
[8] "date"    "shortfac"

```

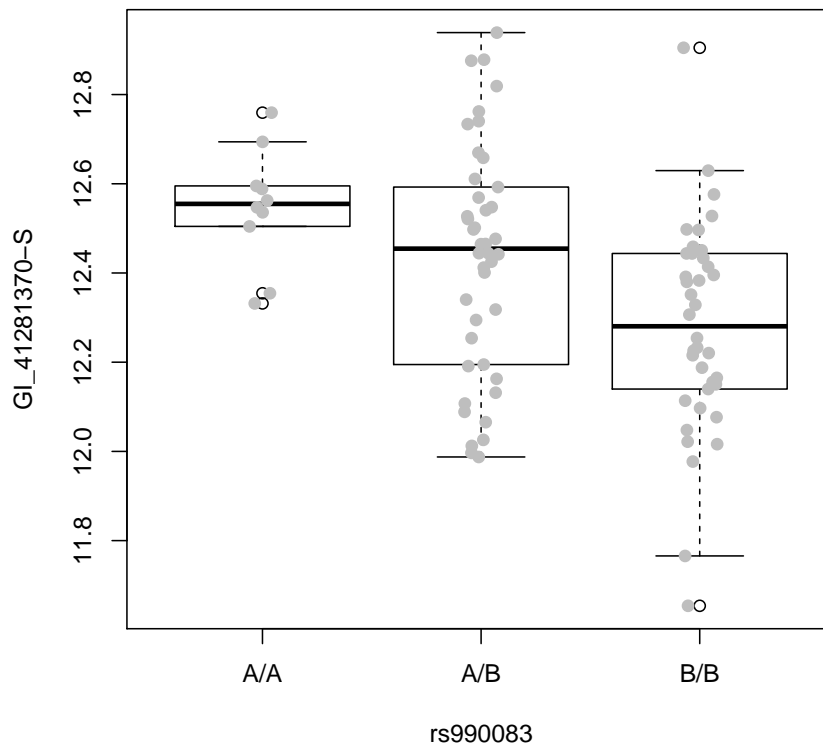
The exercise concerns working effectively with this fairly complicated data structure. The `scores` element has rows corresponding to SNP on chr17, and columns corresponding to ordered eQTL scores. The `i, j` element of `scores` is the `j`th highest score achieved for locus `i`. The `inds` element maps from scores to genes, through the `guniv` element. `guniv[ inds[i, j] ]` is the identifier for the gene giving rise to the `j`th highest score for locus `i`, or, more precisely the locus named by `snpsnames[i]`.

To solve the problem:

```

> fdid = get("FDPS", revmap(illuminaHumanv1SYMBOL))
> fdind = which(tr17@base$guniv == fdid)
> posslocs = which(tr17@base$inds[, 4] == fdind)
> possloc = tr17@base$snpsnames[33649]
> plot_EvG(probeId("GI_41281370-S"), rsid(possloc), c17)

```



- Visualizing an eQTL landscape in R



In the main notes we have emphasized passage of test results to the genome browser as this approach delivers better pan and zoom capabilities in a metadata-rich context. However, it can be useful to create visualizations in R, as they are readily customized.

```

> mplot = function (css, sloc, ...)
+ {
+   pv = p.value(css@.Data[[1]])
+   sn = css@.Data[[1]]@snp.names
+   names(pv) = sn
+   slnames = names(sloc)
+   pv = -log10(pv[intersect(slnames, names(pv))])
+   sloc = sloc[names(pv)]
+   plot(start(sloc), pv, xlab = "genomic coordinates",
+         ylab = "-10 log p", pch="", ...)
+   segments(start(sloc), rep(0,length(pv)), start(sloc), pv)
+   invisible(list(sloc = sloc, pv = pv))
+ }
> geneTick = function(css, useSym=TRUE) {
+   annstr = css@annotation
+   require(annstr, character.only=TRUE)
+   anne = get(paste(gsub(".db", "", annstr), "CHRLOC", sep=""))
+   anneend = get(paste(gsub(".db", "", annstr), "CHRLOCEND", sep=""))
+   gsanne = get(paste(gsub(".db", "", annstr), "SYMBOL", sep=""))
+   ann = get
+   axis(3, at=abs(get(css@psid, anne)[1]),
+         labels=get(css@psid, gsanne)[1], col="green")
+   axis(3, at=abs(get(css@psid, anneend)[1]), col="red", labels="")
+ }

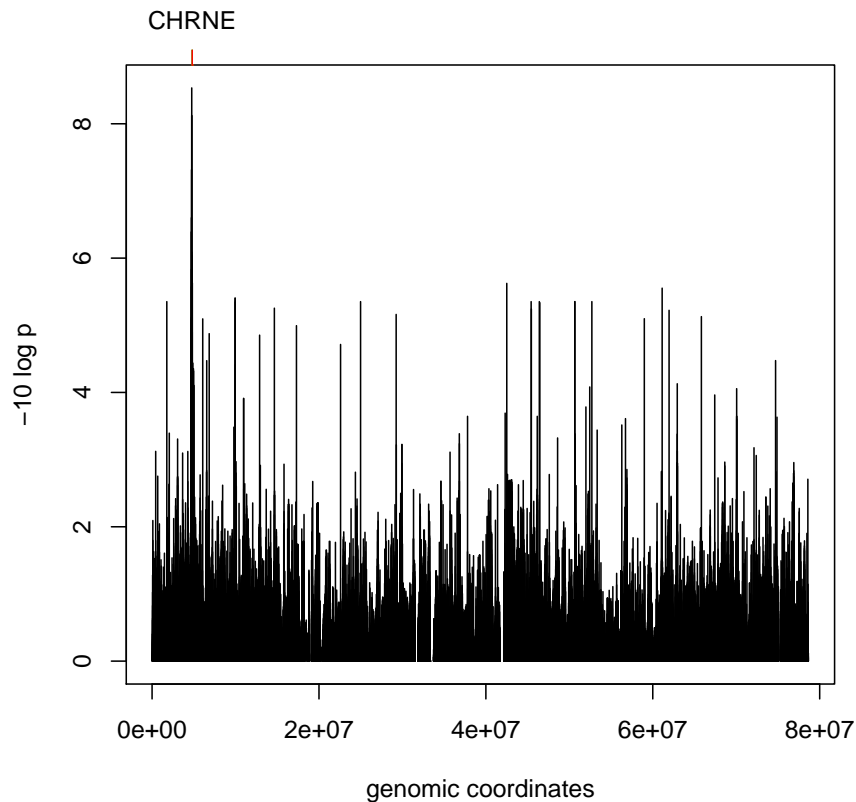
```

Here's an application.

```

> data(snpgr17)
> mplot(t1, snpgr17)
> geneTick(t1)

```



## 2 FDR and MAF

```
> library(ggtut)
> if (!exists("perm17")) perm17 = onePerm17ceu()
> perm17
```

```
eqtlTestsManager  computed Fri May 6 16:06:04 2011
```

```
gene annotation: illuminaHumanv1.db
```

```
There are 1 chromosomes analyzed.
```

```
some genes (out of 498): GI_10190685-S GI_10835020-S ... hmm23927-S hmm5188-S
```

```
some snps (out of 60967): rs6565733 rs1106175 ... rs7502145 rs4986109
```

We will obtain a realization of best per-gene eQTL scores under permutation of expression against genotype.

```
> probes = probesManaged(perm17,1)
> if (!exists("topperm"))
+ topperm = lapply(probes,
```

```

+ function(x)topFeats(probeId(x), mgr=perm17, ffind=1, n=1))
> if (!exists("toppermAt.1"))
+ toppermAt.1 = lapply(probes,
+ function(x)topFeats(probeId(x), mgr=perm17, ffind=1, n=150, minMAF=.1))

```

We need 150 candidates for the MAF-restricted run because these are sparse; topFeats will return an empty list if none of the top n scores meet the MAF threshold.

```

> if (!exists("obs17")) obs17 = observed17ceu()
> obs17

```

```

eqtlTestsManager  computed Fri May  6 16:05:50 2011
gene annotation: illuminaHumanv1.db
There are 1 chromosomes analyzed.
some genes (out of 498): GI_10190685-S GI_10835020-S ... hmm23927-S hmm5188-S
some snps (out of 60967): rs6565733 rs1106175 ... rs7502145 rs4986109

```

```

> if (!exists("topobs"))
+ topobs = lapply(probes,
+ function(x)topFeats(probeId(x), mgr=obs17, ffind=1, n=1))
> if (!exists("topobsAt.1"))
+ topobsAt.1 = lapply(probes,
+ function(x)topFeats(probeId(x), mgr=obs17, ffind=1, n=150, minMAF=.1))

```

Plug-in estimates of FDR corresponding to quantiles of the “permutation distribution” of the test statistic can be obtained as follows.

```

> topps = unlist(topperm)
> topos = unlist(topobs)
> permQ = quantile(topps, c(.95, .975, .99, .995))
> nfalse = sapply(permQ, function(x)sum(topps>x))
> nsig = sapply(permQ, function(x)sum(topos>x))
> options(digits=4)
> cbind(threshold=permQ,
+ permptile=c(.95,.975,.99,.995)*100, nfalse, nsig, fdr=nfalse/nsig)

```

	threshold	permptile	nfalse	nsig	fdr
95%	23.53	95.0	25	45	0.5556
97.5%	26.25	97.5	13	30	0.4333
99%	30.75	99.0	5	22	0.2273
99.5%	32.24	99.5	3	20	0.1500

If we confine attention to SNP with MAFs exceeding some small quantity, we can reduce the FDR.

```

> topps = sapply(toppermAt.1, max)
> topos = sapply(topobsAt.1, max)
> permQ = quantile(topps, c(.95, .975, .99, .995))
> nfalse = sapply(permQ, function(x)sum(topps>x))
> nsig = sapply(permQ, function(x)sum(topos>x))
> cbind(threshold=permQ,
+   permptile=c(.95,.975,.99,.995)*100, nfalse, nsig, fdr=nfalse/nsig)

```

	threshold	permptile	nfalse	nsig	fdr
95%	19.15	95.0	25	78	0.3205
97.5%	20.55	97.5	13	53	0.2453
99%	22.04	99.0	5	36	0.1389
99.5%	24.02	99.5	3	22	0.1364

### 3 Master regulators in cis?

Here we consider the prospect of identifying SNP that are associated with more than one gene on the same chromosome.

The following code runs on a single core laptop in about 20 minutes; the results are saved for you.

```

> sn = snpsManaged(obs17, 1)

> bestGenesPerSNP_obs <- lapply(sn,
+   function(x) topFeats(rsid(x), mgr=obs17, useSym=FALSE, ffind=1))
> bestGenesPerSNP_perm <- lapply(sn,
+   function(x) topFeats(rsid(x), mgr=perm17, useSym=FALSE, ffind=1))

```

We obtain the stored results via

```

> data(bestGenesPerSNP_perm)
> data(bestGenesPerSNP_obs)

```

Each of these objects is a list with 60967 elements; each element is the SNP-specific 10 best association scores over all 498 genes on chr17 filtered for use in the tutorial.

```

> bestGenesPerSNP_obs[1:3]

$rs6565733
  GI_4506130-S GI_42476207-S GI_44889413-S GI_37544934-S GI_33636720-S
      11.54         10.40         9.59         9.45         8.70
GI_38142463-S GI_21389406-S GI_20149678-S GI_4503894-S GI_22094078-S
      8.01         7.97         7.91         7.84         7.78

```

```

$rs1106175
GI_21389406-S GI_22202610-S GI_13375890-A GI_4503894-S GI_31377553-S
      9.13      9.08      8.62      8.45      8.32
GI_4506130-S GI_5453631-S GI_12545398-S GI_21237796-A GI_37544934-S
      7.82      7.67      7.55      7.55      7.52

```

```

$rs17054921
GI_30795213-S GI_20070234-S GI_5902001-S GI_39780587-S GI_37544011-S
      11.21      5.77      5.70      5.37      5.34
GI_17939343-S GI_24415399-S GI_10835020-S GI_41281472-S GI_21361528-S
      4.41      4.10      4.03      3.50      3.47

```

To proceed with a simple assessment, we examine our permutation-based realization from the null distribution of scores for the second-highest scoring gene per SNP.

```

> p2 = sapply(bestGenesPerSNP_perm, "[", 2)
> quantile(p2, c(0.99, 0.995))

```

```

99% 99.5%
14.50 15.71

```

```

> o2 = sapply(bestGenesPerSNP_obs, "[", 2)
> quantile(p2, c(0.999, 0.9999))

```

```

99.9% 99.99%
22.09 26.64

```

```

> sum(o2 > 26.6)

```

```

[1] 10

```

```

> sum(p2 > 26.6)

```

```

[1] 7

```

This is not a good situation. Again, inclusion of SNP with very low MAF is problematic. To filter them away, in the absence of high-level functions, we make use of summaries that are computed with `eqtlTests`.

```

> MAF = function(mgr, ffind) mgr@summaryList[[ffind]][, "MAF"]/mgr@shortfac
> MAFS = MAF(perm17, 1)
> MAFS[1:5]

```

```

rs6565733 rs1106175 rs17054921 rs8064924 rs8070440
      0.12      0.18      0.01      0.12      0.05

```

```

> lowmaf = names(MAFS)[MAFS < 0.1]
> p2rs = sapply(strsplit(names(p2), "\\."), "[[", 1)
> mm = match(lowmaf, p2rs)
> quantile(p2[-mm], c(0.999, 0.9999))

```

```

99.9% 99.99%
16.18 18.08

```

```

> quantile(o2[-mm], c(0.999, 0.9999))

```

```

99.9% 99.99%
18.72 20.23

```

```

> sum(p2[-mm] > 18.08)

```

```

[1] 5

```

```

> sum(o2[-mm] > 18.08)

```

```

[1] 66

```

So we can assert that there are about 66 SNP loci that are associated with at least 2 different genes (probes) at an FDR of about 8%. Let's look at one:

```

> sort(o2[-mm][o2[-mm] > 18.08], decreasing = TRUE)[1:5]

```

```

rs6607284.GI_4758563-S rs12602297.GI_6031190-S rs4790175.GI_40255061-S
                21.39                      20.90                      20.64
rs781839.GI_40255061-S rs781825.GI_40255061-S
                20.23                      20.23

```

```

> 1 - pchisq(21, 1)

```

```

[1] 4.593e-06

```

```

> bestGenesPerSNP_obs[["rs6607284"]]

```

```

GI_31343475-S  GI_4758563-S  GI_13376244-S  GI_45505146-S  GI_15147332-S
                22.41          21.39          19.40          18.99          18.37
GI_33519473-S  GI_14249549-S  GI_34850073-S  GI_34147469-S  GI_8923770-S
                17.70          17.54          17.04          16.89          16.52

```

```

> c17 = getSS("GGdata", "17", renameChrs = "chr17")

```

```

> par(mfrow = c(2, 2))

```

```

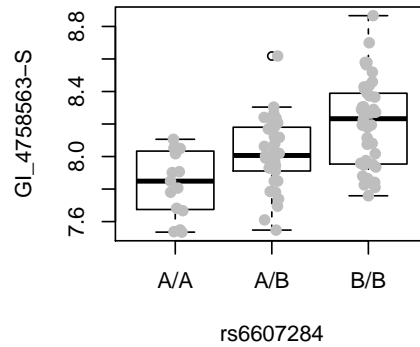
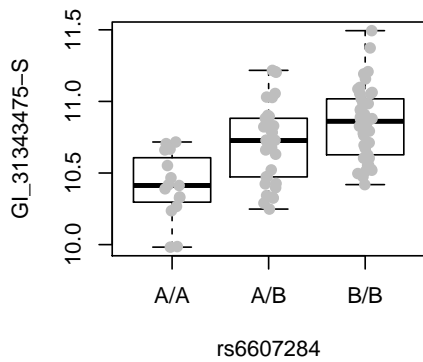
> plot_EvG(probeId("GI_31343475-S"), rsid("rs6607284"), c17)

```

```

> plot_EvG(probeId("GI_4758563-S"), rsid("rs6607284"), c17)

```



Some annotation:

```
> library(illuminaHumanv1.db)
> get("GI_31343475-S", illuminaHumanv1GENENAME)
[1] "guanine nucleotide binding protein (G protein), alpha 13"
> get("GI_4758563-S", illuminaHumanv1GENENAME)
[1] "ubiquitin specific peptidase 6 (Tre-2 oncogene)"
> get("GI_31343475-S", illuminaHumanv1CHRLLOC)
      17
-63005408
> get("GI_4758563-S", illuminaHumanv1CHRLLOC)
      17
5031686
```

```
> get("GI_31343475-S", illuminaHumanv1PATH)
```

```
[1] "04270" "04730" "04810"
```

```
> get("GI_4758563-S", illuminaHumanv1PATH)
```

```
[1] NA
```

## 4 Session information

```
> sessionInfo()
```

```
R version 2.14.0 alpha (2011-10-03 r57154)  
Platform: x86_64-apple-darwin10.8.0/x86_64 (64-bit)
```

```
locale:
```

```
[1] en_US.US-ASCII/en_US.US-ASCII/en_US.US-ASCII/C/en_US.US-ASCII/en_US.US-ASCII
```

```
attached base packages:
```

```
[1] grid      splines    stats      graphics  grDevices datasets  utils  
[8] tools     methods    base
```

```
other attached packages:
```

```
[1] multicore_0.1-7  
[2] VariantAnnotation_0.99.28  
[3] BSgenome.Hsapiens.UCSC.hg18_1.3.17  
[4] TxDb.Hsapiens.UCSC.hg18.knownGene_2.6.2  
[5] parody_1.11.0  
[6] nlme_3.1-102  
[7] ggtut_0.0.32  
[8] hmyriB36_0.99.11  
[9] SNPlocs.Hsapiens.dbSNP.20090506_0.99.6  
[10] cheung2010_0.0.20  
[11] hgfocus.db_2.6.3  
[12] Rsamtools_1.5.75  
[13] ChIPpeakAnno_2.0.5  
[14] gplots_2.10.1  
[15] KernSmooth_2.23-6  
[16] caTools_1.12  
[17] gdata_2.8.2  
[18] gtools_2.6.2  
[19] limma_3.9.21
```



[20] GO.db\_2.6.1  
[21] BSgenome.Ecoli.NCBI.20080805\_1.3.17  
[22] BSgenome\_1.21.5  
[23] Biostrings\_2.21.11  
[24] multtest\_2.9.1  
[25] biomaRt\_2.9.3  
[26] GenomicFeatures\_1.5.27  
[27] GGdata\_1.0.16  
[28] illuminaHumanv1.db\_1.10.0  
[29] ff\_2.2-3  
[30] bit\_1.1-7  
[31] GGtools\_3.99.34  
[32] GenomicRanges\_1.5.49  
[33] org.Hs.eg.db\_2.6.3  
[34] rtracklayer\_1.13.17  
[35] RCurl\_1.6-10  
[36] bitops\_1.0-4.1  
[37] IRanges\_1.11.29  
[38] annotate\_1.31.1  
[39] AnnotationDbi\_1.15.34  
[40] GGBase\_3.13.8  
[41] genefilter\_1.35.0  
[42] RSQLite\_0.10.0  
[43] DBI\_0.2-5  
[44] snpStats\_1.3.4  
[45] Matrix\_1.0-0  
[46] lattice\_0.19-33  
[47] survival\_2.36-10  
[48] Biobase\_2.13.10  
[49] BiocInstaller\_1.1.28  
[50] weaver\_1.19.2  
[51] codetools\_0.2-8  
[52] digest\_0.5.1

loaded via a namespace (and not attached):

[1] MASS\_7.3-15 XML\_3.4-3 xtable\_1.5-6 zlibbioc\_0.1.8