Using Categories defined by Chromosome Bands

D. Sarkar, S. Falcon, R. Gentleman

August 7, 2008

1 Chromosome bands

Typically, in higher organisms, each chromosome has a centromere and two arms. The short arm is called the p arm and the longer arm the q arm. Chromosome bands (see Figure 1) are identified by differential staining, usually with Giemsa-based stains, and many disease-related defects have been mapped to these bands; such mappings have played an important role in classical cytogenetics. With the availability of complete sequences for several genomes, there have been efforts to link these bands with specific sequence locations (Furey and Haussler, 2003). The estimated location of the bands in the reference genomes can be obtained from the UCSC genome browser, and data linking genes to particular bands can be obtained from a variety of sources such as the NCBI. This vignette demonstrates tools that allow the use of categories derived from chromosome bands, that is, the relevant categories are determined *a priori* by a mapping of genes to chromosome bands.

Figure 1 shows an ideogram of human chromosome 12, with the band 12q21 shaded. As shown in the figure, 12q21 can be divided into more granular levels 12q21.1, 12q21.2, and 12q21.3. 12q21.3 can itself be divided at an even finer level of resolution into 12q21.31, 12q21.32, and 12q21.33. Moving towards less granular bands, 12q21 is a part of 12q2 which is again a part of 12q. We take advantage of this nested structure of the bands in our analysis.

```
> library("Category")
> library("ALL")
> library("hgu95av2.db")
> library("annotate")
> library("genefilter")
> library("SNPchip")
> library("geneplotter")
> library("limma")
> library("lattice")
> library("graph")
```

2 Data Preparation

For illustration, we use a microarray dataset (Chiaretti et al., 2005) from a clinical trial in acute lymphoblastic leukemia (ALL). The data are described in Chapter 2 of Hahne et al.

```
NULL

NULL
```

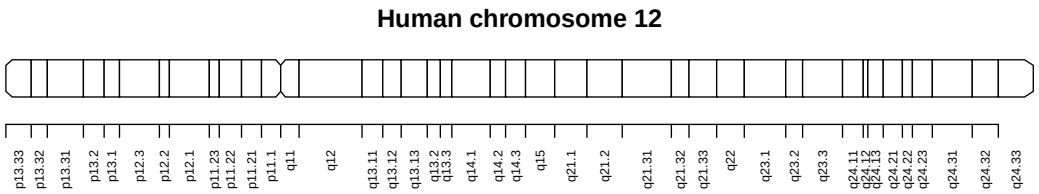

Figure 1: Ideogram for human chromosome 12. The p arm is on the left, the q arm is on the right, and the centromere is indicated by a notch. The shaded bands together represent 12q21. This band is composed of three sub-bands: 12q21.1, 12q21.2, and 12q21.3. The last of these is composed of sub-sub-bands 12q21.31, 12q21.32, and 12q21.33.

(2008). For the analysis presented here, we consider the comparison of two subsets: patients identified as having a BCR/ABL gene fusion present, typically as a result of a translocation of chromosomes 9 and 22 (labeled BCR/ABL), and those that have no observed cytogenetic abnormalities (labeled NEG). The full dataset is available in the ALL package, and the relevant subset of the data can be obtained by

```
> data(ALL, package="ALL")
> subsetType <- "BCR/ABL"
> Bcell <- grep("^B", as.character(ALL$BT))
> bcrAblOrNegIdx <- which(as.character(ALL$mol.biol) %in% c("NEG", subsetType))
> bcrAblOrNeg <- ALL[, intersect(Bcell, bcrAblOrNegIdx)]
> bcrAblOrNeg$mol.biol <- factor(bcrAblOrNeg$mol.biol)
```

We also create relevant annotation maps to go from feature names to Entrez ID, gene symbol, and chromosome band.

```
> annType <- c("db", "env")
> entrezMap <- getAnnMap("ENTREZID", annotation(bcrAblOrNeg),
+                        type=annType, load=TRUE)
> symbolMap <- getAnnMap("SYMBOL", annotation(bcrAblOrNeg),
+                        type=annType, load=TRUE)
> bandMap <- getAnnMap("MAP", annotation(bcrAblOrNeg),
+                      type=annType, load=TRUE)
```

We applied a non-specific filter to the dataset to remove probesets lacking the desired annotation as well as those with an interquartile range (IQR) below the median IQR, as probesets with little variation across samples are uninformative. We also ensured that each Entrez Gene identifier maps to exactly one probeset by selecting the probeset with the largest IQR when two or more probesets map to the same Entrez Gene ID.

```
> filterAns <- nsFilter(bcrAblOrNeg,
+                       require.entrez = TRUE,
```

```
+                         remove.dupEntrez = TRUE,
+                         var.func = IQR, var.cutoff = 0.5)
> nsFiltered <- filterAns$eset
```

We also remove probesets with no gene symbol, as well as those with no mapping to a chromosome band.

```
> hasSYM <- sapply(mget(featureNames(nsFiltered), symbolMap, ifnotfound=NA),
+                  function(x) length(x) > 0 && !is.na(x[1]))
> hasMAP <- sapply(mget(featureNames(nsFiltered), bandMap, ifnotfound=NA),
+                  function(x) length(x) > 0 && !is.na(x[1]))
> nsFiltered <- nsFiltered[hasSYM & hasMAP, ]
```

We define the *gene universe* to be the subset of genes that remain after this filtering.

```
> affyUniverse <- featureNames(nsFiltered)
> entrezUniverse <- unlist(mget(affyUniverse, entrezMap))
> names(affyUniverse) <- entrezUniverse
> if (any(duplicated(entrezUniverse)))
+     stop("error in gene universe: can't have duplicate Entrez Gene Ids")
```

We assessed differential expression between the BCR/ABL and NEG groups using an empirical Bayes approach, as implemented in the software package limma (Smyth, 2005), yielding an attenuated t-statistic for each gene.

```
> design <- model.matrix(~ 0 + nsFiltered$mol.biol)
> colnames(design) <- c("BCR/ABL", "NEG")
> contr <- c(1, -1) ## NOTE: we thus have BCR/ABL w.r.t NEG
> fm1 <- lmFit(nsFiltered, design)
> fm2 <- contrasts.fit(fm1, contr)
> fm3 <- eBayes(fm2)
> ttestLimma <- topTable(fm3, number = nrow(fm3), adjust.method = "none")
> ttestLimma <- ttestLimma[featureNames(nsFiltered), ]
> tstats <- ttestLimma$t
> names(tstats) <- entrezUniverse[rownames(ttestLimma)]
> ##
```

We used a p-value cutoff of 0.01 to identify a list of potentially differentially expressed genes.

```
> ttestCutoff <- 0.01
> smPV  <- ttestLimma$P.Value < ttestCutoff
> pvalFiltered <- nsFiltered[smPV, ]
> selectedEntrezIds <- unlist(mget(featureNames(pvalFiltered), entrezMap))
> ##
```

3 Methods

There are two important features of gene sets based on chromosome bands: (1) the bands are nested hierarchically, and (2) they almost form a partition (for most species almost all genes

appear in only one location in the genome). This naturally leads to two different dichotomies in approaches to testing: one is a top-down versus a bottom-up approach, and the other contrasts local tests with global tests. We use human chromosome 12 as an example to describe these approaches. Users will need to identify which of the approaches best align with their objectives and then make use of the software appropriately.

Conceptually one can start a sequential testing approach either at the most coarse level of organization (probably the level of the arm: p or q), or the most specific level (that of sub-sub-bands). In the top-down approach, one first tests the hypothesis of interest on a coarse level of organization, and if rejected, the next level of organization is considered. For example, we might first consider the band 12q21, and if that hypothesis is rejected, test each of the sub-bands 12q21.1, 12q21.2, and 12q21.3. If the hypothesis for 12q21.3 is rejected, then we may subsequently examine each of 12q21.31, 12q21.32, and 12q21.33.

The bottom-up approach differs in that we begin at the most granular level and move upward, amalgamating adjacent bands at each level. The bottom-up approach is easier to put into a conditional hypothesis testing framework. Our initial null hypotheses involve the smallest, or most granular bands, and if there is evidence that these are unusual (i.e., we reject the null hypothesis) then moving to a larger, or less granular, band requires additional information to declare it significant, over and above what we have used to identify the smaller band. In our example, we would first test 12q21.31, 12q21.32, and 12q21.33, and then move up and test 12q21.3. If one or more of the three sub-bands had been declared significant, we would exclude the evidence from genes annotated in those sub-bands when testing the coarser band.

It is important to note that the top-down versus bottom-up approaches represent a fundamental trade-off between false positive and false negative errors. The bottom-up approach necessarily involves performing a larger number of tests, yielding a correspondingly larger absolute number of false positives for a given false positive rate at which each individual test is controlled. The top-down approach cuts down on the number of false positives by starting with fewer top-level tests, and performing further tests at sublevels only when a top-level test is rejected. The disadvantage to this approach is loss of power to detect real departures that are localized to a sub-level, a phenomenon commonly illustrated using Simpson's paradox (see, e.g., Wagner, 1982).

Whether a test is local or global is a different question, orthogonal to that of top-down or bottom-up. There are two distinct but potentially relevant questions that may be of interest. The first is whether genes in a particular gene set are "different" relative to all other genes under consideration. For a Hypergeometric test, this question may be formalized as whether the proportion of interesting genes in 12q21 is different from the proportion of interesting genes in the rest of the genome, or equivalently, whether membership in 12q21 is independent of being selected. Such tests are *global* in the sense that all genes in the gene universe are used to determine whether or not genes at a location are unusual or not. An alternative is to ask whether genes in a genomic location are different relative to other genes in some meaningfully defined neighbourhood. Such a test can be performed simply by restricting the gene universe to a suitable subset; for example, when testing 12q21, we may only consider genes in 12q. A more natural approach is to use a 2×3 contingency table to test the hypothesis that the proportion of interesting genes is the same in 12q21, 12q22, and 12q23. Both these tests are *local* in the sense that only nearby genes are used.

Contingency table tests are inherently local and although they do not naturally extend to conditional testing, we can use a top-down approach to test at various resolutions. Such tests can be performed by the `cb_contingency()` function, which we do not discuss in this vignette. Instead, we focus on the bottom-up approach, which allows for conditional testing.

4 Utility functions

We first define a few utility functions that we subsequently use in presentation. The `chrSortOrder()` function reorders rows of data frame for display in a natural order.

```
> chrSortOrder <- function(df) {
+     chrs <- sub("([^pq]+).*$", "\\1", rownames(df))
+     xyIdx <- chrs %in% c("X", "Y")
+     xydf <- NULL
+     if (any(xyIdx)) {
+         chrs <- chrs[!xyIdx]
+         xydf <- df[xyIdx, ]
+         df <- df[!xyIdx, ]
+     }
+     ord <- order(as.integer(chrs), rownames(df))
+     df <- df[ord, ]
+     if (!is.null(xydf))
+       df <- rbind(df, xydf)
+     df
+ }
```

The `gseaTstatStripplot()` function creates a comparative strip plot of the t-statistics for specified bands.

```
> gseaTstatStripplot <- function(bands, g, ..., include.all = FALSE)
+ {
+     chroms <- c(1:22, "X", "Y")
+     chromArms <- c(paste(chroms, "p", sep=""), paste(chroms, "q", sep=""))
+     egid <- lapply(nodeData(g, bands), "[[", "geneIds")
+     if (include.all) {
+         egid$All <-
+             unique(unlist(lapply(nodeData(g)[chromArms], "[[", "geneIds")))
+     }
+     tdf <- do.call(make.groups, lapply(egid, function(x) tstats[x]))
+     stripplot(which ~ data, tdf, jitter = TRUE, ...)
+ }
>
>
```

The `esetBWPlot()` function creates box-and-whisker plots for every gene in an *"ExpressionSet"*.

```
> esetBWPlot <- function(tmpSet, ..., layout=c(1, nrow(emat)))
+ {
+     emat <- exprs(tmpSet)
+     pd <- pData(tmpSet)
+     probes <- rownames(emat)
+     syms <-
+         sapply(mget(probes, hgu95av2SYMBOL, ifnotfound=NA),
+                function(x) if (all(is.na(x))) "NA" else as.character(x)[1])
+     selectedAffy <-
+         probes %in% affyUniverse[selectedEntrezIds]
+     symsSelected <- syms[selectedAffy]
+     symsWithStatus <-
+         paste(syms,
+               ifelse(selectedAffy, "*", ""),
+               sep = "")
+     pdat <-
+         cbind(exprs=as.vector(emat),
+               genes=factor(probes, levels = probes, labels = syms),
+               pd[rep(seq_len(nrow(pd)), each=nrow(emat)), ])
+     pdat <- transform(pdat, genes = reorder(genes, exprs))
+     panels.to.shade <- levels(pdat$genes) %in% symsSelected
+     bwplot(mol.biol ~ exprs | genes, data=pdat,
+            layout = layout,
+            auto.key=TRUE,
+            scales=list(x=list(log=2L)),
+            xlab="Log2 Expression",
+            panels.to.shade = panels.to.shade,
+            panel = function(..., panels.to.shade) {
+                if (panels.to.shade[packet.number()])
+                    panel.fill(col = "lightgrey")
+                panel.bwplot(...)
+            },
+            strip=FALSE,
+            strip.left=TRUE, ...)
+ }
> g1 <- makeChrBandGraph(annotation(nsFiltered), univ=entrezUniverse)
> ct <- ChrBandTreeFromGraph(g1)
> subsetByBand <- function(eset, ct, band) {
+     egIDs <- unlist(nodeData(ct@toChildGraph, n=band,
+                              attr="geneIds"), use.names=FALSE)
+     wantedProbes <- affyUniverse[as.character(egIDs)]
+     eset[intersect(wantedProbes, featureNames(eset)), ]
+ }
>
```

5 Hypergeometric Testing

We use a method similar to that described in Falcon and Gentleman (2007) to conditionally test for over-representation of chromosome bands in the selected gene list. A test is set up by creating a suitable object of class *"ChrMapHyperGParams"*. We first create an object to perform a standard Hypergeometric analysis, treating each chromosome band independently, and then modify a copy to represent a conditional Hypergeometric computation.

```
> params <- new("ChrMapHyperGParams",
+               conditional=FALSE,
+               testDirection="over",
+               universeGeneIds=entrezUniverse,
+               geneIds=selectedEntrezIds,
+               annotation="hgu95av2",
+               pvalueCutoff=0.05)
> paramsCond <- params
> paramsCond@conditional <- TRUE
```

The test computations are performed by

```
> hgans <- hyperGTest(params)
> hgansCond <- hyperGTest(paramsCond)
```

The results can be summarized by

```
> sumUn <- summary(hgans, categorySize=1)
> chrSortOrder(sumUn)
```

	ChrMapID	Pvalue	OddsRatio	ExpCount	Count	Size
1	6p23	0.002093524	33.628895	0.3300881	3	4
2	14q22.2	0.004913731	16.810198	0.4126101	3	5
3	7p15	0.005580562	5.623932	1.2378303	5	15
4	13q31	0.006792330	Inf	0.1650440	2	2
5	12q14.1	0.006792330	Inf	0.1650440	2	2
6	13q31.1	0.006792330	Inf	0.1650440	2	2
7	7q31.2	0.006792330	Inf	0.1650440	2	2
8	14q22	0.016425077	4.013024	1.5679184	5	19
9	9q21.13	0.019261736	22.355932	0.2475661	2	3
10	6q	0.020819912	1.984357	7.7570700	14	94
11	6q2	0.021995096	2.182100	5.6114975	11	68
12	2q32	0.032157970	5.597734	0.7426982	3	9
13	4q22	0.036429948	11.175141	0.3300881	2	4
14	9q21.1	0.036429948	11.175141	0.3300881	2	4
15	12p12.3	0.036429948	11.175141	0.3300881	2	4
16	7p15.3	0.036429948	11.175141	0.3300881	2	4
17	1p36	0.042251321	2.013832	5.4464534	10	66
18	9	0.042748920	1.621450	12.5433472	19	152

```
19    12q14 0.043192180  4.796843  0.8252202     3   10
20  11p15.4 0.043192180  4.796843  0.8252202     3   10
21 22q11.23 0.043192180  4.796843  0.8252202     3   10
```

```
> sumCond <- summary(hgansCond, categorySize=1)
> chrSortOrder(sumCond)
```

```
   ChrMapID      Pvalue OddsRatio   ExpCount Count Size
1      6p23 0.002093524 33.628895  0.3300881     3    4
2   14q22.2 0.004913731 16.810198  0.4126101     3    5
3   12q14.1 0.006792330       Inf  0.1650440     2    2
4   13q31.1 0.006792330       Inf  0.1650440     2    2
5    7q31.2 0.006792330       Inf  0.1650440     2    2
6   9q21.13 0.019261736 22.355932  0.2475661     2    3
7       6q2 0.021995096  2.182100  5.6114975    11   68
8      2q32 0.032157970  5.597734  0.7426982     3    9
9      4q22 0.036429948 11.175141  0.3300881     2    4
10  12p12.3 0.036429948 11.175141  0.3300881     2    4
11   7p15.3 0.036429948 11.175141  0.3300881     2    4
12     1p36 0.042251321  2.013832  5.4464534    10   66
13        9 0.042748920  1.621450 12.5433472    19  152
14  11p15.4 0.043192180  4.796843  0.8252202     3   10
15 22q11.23 0.043192180  4.796843  0.8252202     3   10
```

For the standard test, the structure of the chromosome band graph is ignored and a Hypergeometric test is applied to each band independently. For the conditional test, the hierarchical relationship among the bands as represented by the graph is used in the computation. The highest-resolution bands (those with no children in the graph) are tested first. Testing proceeds with the bands whose children (sub-bands) have already been tested. For these bands, the gene annotations that are inherited from significant child nodes (children with p-value smaller than the specified cutoff) are removed prior to testing to yield a conditional test.

The effect of the conditional test is illustrated by examining the results for 14q and its sub-bands. In the standard test, we see that 14q22 and 14q22.2 both have a significant p-value. In the conditional test, only 14q22.2 remains. The conclusion is that there is not enough additional evidence beyond that provided by 14q22.2 to mark 14q22 as significant.

6 GSEA using linear models

GSEA is a popular method that can be used to assess whether or not particular gene sets are associated with a phenotype of interest (Subramanian et al., 2005; Tian et al., 2005; Jiang and Gentleman, 2007). Most applications of this method do not explicitly deal with structure of the gene sets, but when analyzing chromosomal location such methods are desirable. We present a simple approach that is similar in spirit to traditional GSEA, and generalizes nicely to accommodate nested categories. Consider the situation where gene i has an associated measure of differential expression y_i; for example, an attenuated t-statistic derived from a differential

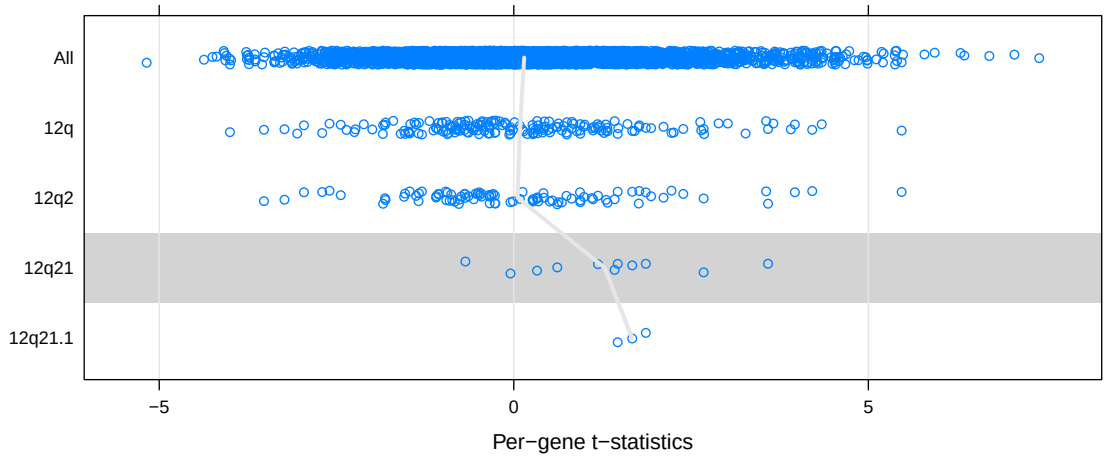

Figure 2: Per-gene t-statistics, as computed by limma, for selected chromosome bands. The points are jittered vertically to alleviate overlap. A thick grey line joins the mean value within each group. A GSEA test would compare the genes in 12q21 with those in 12q, or the entire gene universe, by fitting a linear model with per-gene t-statistics as the response, and a term indicating membership in 12q21 as the predictor. If 12q21.1 is declared as significant, a conditional GSEA test would include an additional term for 12q21.1.

expression analysis such as limma (Smyth, 2005). Given a particular category, GSEA asks whether the distribution of y_i-s restricted to the category of interest is "unusual". Thus, in Figure 2, we might be interested in knowing whether the distribution of y_i values for genes in 12q21 is different from that of the genes in 12q (for a local test) or of all genes in the gene universe (for a global test). Figure 2 is produced by

```
> gseaTstatStripplot(c("12q21.1", "12q21", "12q2", "12q"),
+                    include.all = TRUE,
+                    g = g1,
+                    xlab = "Per-gene t-statistics",
+                    panel = function(...) {
+                        require(grid, quietly = TRUE)
+                        grid.rect(y = unit(2, "native"),
+                                  height = unit(1, "native"),
+                                  gp =
+                                  gpar(fill = "lightgrey",
+                                       col = "transparent"))
+                        panel.grid(v = -1, h = 0)
+                        panel.stripplot(...)
+                        panel.average(..., fun = mean, lwd = 3)
+                    })
```

We fit a factorial model to see whether the distribution of y_i is associated with category membership. Specifically, for category j, we fit the model

$$y_i = \beta_0 + \beta_1 a_{ij} + \varepsilon_i \tag{1}$$

where $a_{ij} = 1$ if gene i belongs to category j, and 0 otherwise. The index i may range over the full gene universe, or a subset, depending on whether one wishes to perform global or local tests. The null hypothesis of no association is represented by $H_0 : \beta_1 = 0$. The model nominally assumes that the y_i-s are Normally distributed with equal variance, but in practice the results are robust against mild deviations. The presence of an intercept term allows nonzero overall mean, which can be important in many situations, especially for local tests. We expect the test to be fairly insensitive to distributional assumptions on the y_i-s.

We can fit (1) by least squares and test $H_0 : \beta_1 = 0$ to obtain a marginal test for each category j; in this case, each chromosome band. The procedure also generalizes to incorporate the nesting structure of chromosome bands. Specifically, if band j_2 (e.g., 12q21.1) is nested within a coarser band j_1 (e.g., 12q21) and the more granular band j_2 is significant, then the effect of membership in j_1 over and above the effect attributable to membership in j_2 can be tested by fitting the model

$$y_i = \beta_0 + \beta_1 a_{ij_1} + \beta_2 a_{ij_2} + \varepsilon_i \tag{2}$$

and testing the null hypothesis $H_0 : \beta_1 = 0$. Multiple significant sub-bands and multiple levels of nesting can be incorporated by including further terms in the model. The complete process can be summarized as follows: Start by marginally testing each band which has no sub-bands. For all other bands, first test all sub-bands, then test the current band using a linear model that includes a term for each *significant* sub-band.

We apply this procedure to perform global tests using per-gene t-statistics as a measure of differential expression in BCR/ABL relative to NEG samples. As with the Hypergeometric tests, we start by creating objects of class *"ChrMapLinearMParams"*.

```
> params <- new("ChrMapLinearMParams",
+               conditional = FALSE,
+               testDirection = "up",
+               universeGeneIds = entrezUniverse,
+               geneStats = tstats,
+               annotation = "hgu95av2",
+               pvalueCutoff = 0.01,
+               minSize = 4L)
> params@graph <- makeChrBandGraph(params@annotation, params@universeGeneIds)
> params@gsc <- makeChrBandGSC(params@graph)
> paramsCond <- params
> paramsCond@conditional <- TRUE
```

The tests are performed, and the results summarized by

```
> lmans <- linearMTest(params)
> lmansCond <- linearMTest(paramsCond)
> chrSortOrder(summary(lmans))
```

```
     ChrMapID       Pvalue    Effect Size Conditional TestDirection
53       12q21 6.880695e-03 1.1357271   11       FALSE            up
75          14 8.982563e-03 0.2920338  159       FALSE            up
76         14q 8.982563e-03 0.2920338  159       FALSE            up
80        14q2 2.647195e-04 0.6160399   75       FALSE            up
82       14q22 1.060929e-04 1.2994190   19       FALSE            up
110       16q1 4.973303e-03 0.8229480   23       FALSE            up
132        18p 1.165917e-03 0.8811387   28       FALSE            up
133       18p1 1.165917e-03 0.8811387   28       FALSE            up
134      18p11 1.165917e-03 0.8811387   28       FALSE            up
160        1p2 2.259325e-03 0.7942724   30       FALSE            up
213          2 3.756307e-04 0.3117384  292       FALSE            up
224         2q 1.190715e-04 0.4463934  164       FALSE            up
225        2q1 5.966671e-03 0.6610370   34       FALSE            up
234        2q3 8.122015e-03 0.3807922   95       FALSE            up
240          3 6.535031e-03 0.2502859  243       FALSE            up
253         3q 9.199609e-04 0.4402270  120       FALSE            up
259        3q2 6.694722e-03 0.3859373   98       FALSE            up
262       3q25 1.855496e-03 1.1458453   15       FALSE            up
267          4 1.554677e-03 0.3658692  158       FALSE            up
273         4q 3.169932e-03 0.4061720  108       FALSE            up
277       4q21 7.925223e-03 0.9528781   15       FALSE            up
287          5 5.104314e-04 0.3596184  204       FALSE            up
293         5q 3.151809e-03 0.3227621  174       FALSE            up
299        5q2 4.092325e-03 0.9283902   19       FALSE            up
304       5q31 7.092957e-03 0.5877231   41       FALSE            up
307          6 4.088163e-05 0.3872333  256       FALSE            up
317         6q 2.685345e-06 0.7233527   94       FALSE            up
321        6q2 2.996466e-06 0.8435741   68       FALSE            up
384     9q21.1 9.362534e-05 2.8511066    4       FALSE            up
620    14q22.2 1.794762e-04 2.4366868    5       FALSE            up
916       3q28 2.449730e-03 1.7549582    6       FALSE            up
980     5q31.1 9.425306e-03 0.9276824   15       FALSE            up
1001      6p23 1.487763e-04 2.7611697    4       FALSE            up
1066      8p22 1.268854e-03 2.0621705    5       FALSE            up
```

```
> chrSortOrder(summary(lmansCond))
```

```
     ChrMapID       Pvalue    Effect Size Conditional TestDirection
53       12q21 6.880695e-03 1.1357271   11        TRUE            up
80        14q2 4.124863e-03 0.4854303   75        TRUE            up
110       16q1 4.973303e-03 0.8229480   23        TRUE            up
134      18p11 1.165917e-03 0.8811387   28        TRUE            up
160        1p2 2.259325e-03 0.7942724   30        TRUE            up
225        2q1 5.966671e-03 0.6610370   34        TRUE            up
234        2q3 8.122015e-03 0.3807922   95        TRUE            up
```

```
262      3q25 1.855496e-03 1.1458453   15        TRUE            up
267         4 9.750430e-03 0.3032366  158        TRUE            up
277      4q21 7.925223e-03 0.9528781   15        TRUE            up
299       5q2 4.092325e-03 0.9283902   19        TRUE            up
321       6q2 2.996466e-06 0.8435741   68        TRUE            up
384    9q21.1 9.362534e-05 2.8511066    4        TRUE            up
620   14q22.2 1.794762e-04 2.4366868    5        TRUE            up
916      3q28 2.449730e-03 1.7549582    6        TRUE            up
980    5q31.1 9.425306e-03 0.9276824   15        TRUE            up
1001     6p23 1.487763e-04 2.7611697    4        TRUE            up
1066     8p22 1.268854e-03 2.0621705    5        TRUE            up

>
> ##
```

These examples only test for consistently upregulated categories; similar calls with `testDirection = "down"` can be used to test for downregulation. As we see, the GSEA approach picks out many more bands as significant, but there is some concordance with the Hypergeometric approach. For example, 7q31, 8p22, and 14q22.2 come up in both analyses. Figure 3 shows box-and-whisker plots of genes in one category (1p36.2) that is declared as significant by the Hypergeometric test, but not by GSEA. It is produced by

```
> tmpSet <- subsetByBand(nsFiltered, ct, "1p36.2")
> esetBWPlot(tmpSet, ylab="1p36.2", layout = c(2, 8),
+            par.strip.text = list(cex = 0.8))
```

References

S. Chiaretti, X. Li, R. Gentleman, A. Vitale, K. S. Wang, F. Mandelli, R. Foa, and J. Ritz. Gene expression profiles of b-lineage adult acute lymphocytic leukemia reveal genetic patterns that identify lineage derivation and distinct mechanisms of transformation. *Clinical Cancer Research*, 11:7209–7219, 2005.

S Falcon and R Gentleman. Using GOstats to test gene lists for GO term association. *Bioinformatics*, 23(2):257–8, 2007.

T. S. Furey and D. Haussler. Integration of the cytogenetic map with the draft human genome sequence. *Human Molecular Genetics*, 12:1037–1044, 2003. URL `DOI:10.1093/hmg/ddg113`.

F. Hahne, W. Huber, R. Gentleman, and S. Falcon. *Bioconductor Case Studies*. Springer, New York, 2008.

Z. Jiang and R. Gentleman. Extensions to gene set enrichment. *Bioinformatics*, 23:306–313, 2007.

Gordon K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions using R and Bioconductor*, pages 397–420. Springer, New York, 2005.

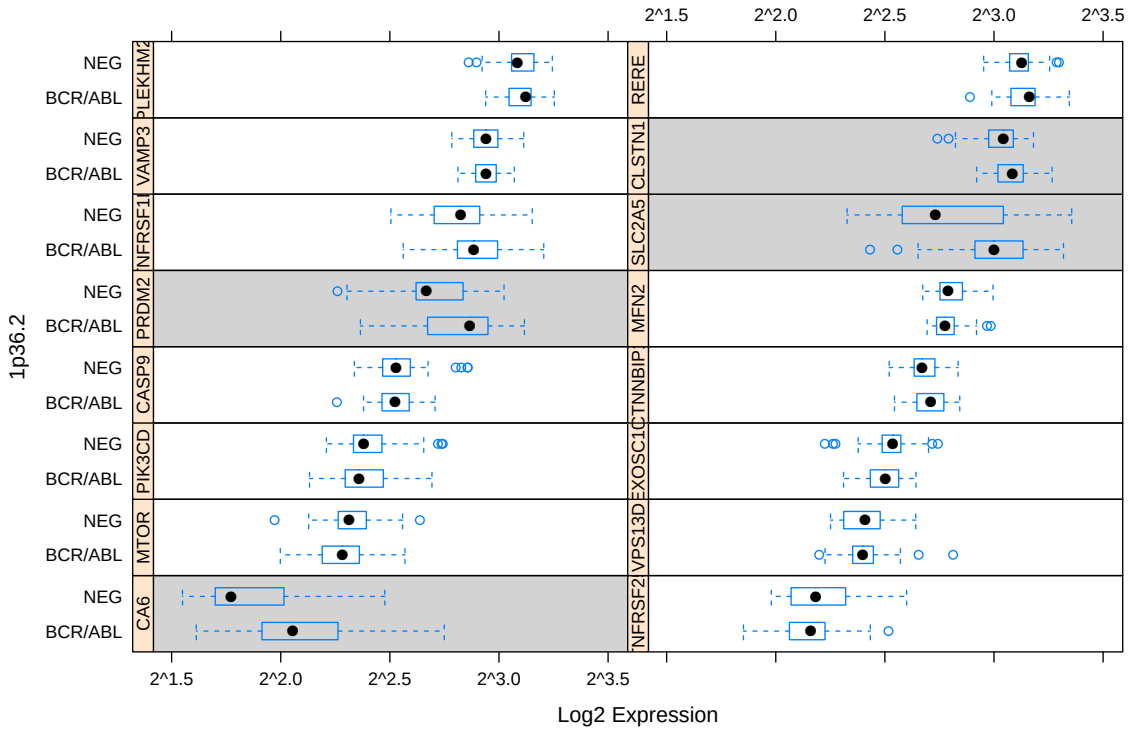

Figure 3: Expression values for the BCR/ABL and NEG samples for the genes located within 1p36.2, which is declared as significant by the Hypergeometric test, but not by GSEA. Genes in the selected list are highlighted with a shaded background. For these genes, The NEG samples, those with no known cytogenetic abnormalities, have significantly lower expression than the BCR/ABL samples. However, the direction is reversed (albeit mildly) for many of the other genes.

A. Subramanian, P. Tamayo, V. K. Mootha, et al. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

L. Tian, S. A. Greenberg, S. W. Kong, et al. Discovering statistically significant pathways in expression profiling studies. *Proceedings of the National Academy of Sciences*, 102(38): 13544–13549, 2005.

Clifford H. Wagner. Simpson's paradox in real life. *The American Statistician*, 36(1):46–48, 1982. ISSN 00031305.