Using ReportingTools in an Analysis of Microarray Data

Jason A. Hackney and Jessica L. Larson

October 2, 2012

Contents

1 Introduction

The ReportingTools package is particularly useful in displaying results from a microarray experiment. In this vignette we show how to display results from differential gene expression, Gene Ontology (GO), protein families (PFAM) and gene set enrichment analyses. In the final section, we create an index page where the user can easily access any of these results.

2 Differential expression analysis using limma

For this vignette we will examine the ALL dataset. First we load our ReportingTools package and the data. This dataset is from a clinical trial in acute lymphoblastic leukemia (ALL) and is available from Bioconductor.

```
> library(ReportingTools)
> library(ALL)
> library(hgu95av2.db)
> library(genefilter)
> data(ALL)
```

We will compare the gene expression between the BCR/ABL and NEG samples. We use featureFilter to remove most of the unexpressed genes.

```
> ALL <- ALL[, ALL$mol.biol %in% c('NEG','BCR/ABL') &
+     !is.na(ALL$sex)]
> ALL$mol.biol <- factor(ALL$mol.biol,
+     levels = c('NEG', 'BCR/ABL'))
> ALL <- featureFilter(ALL)
```

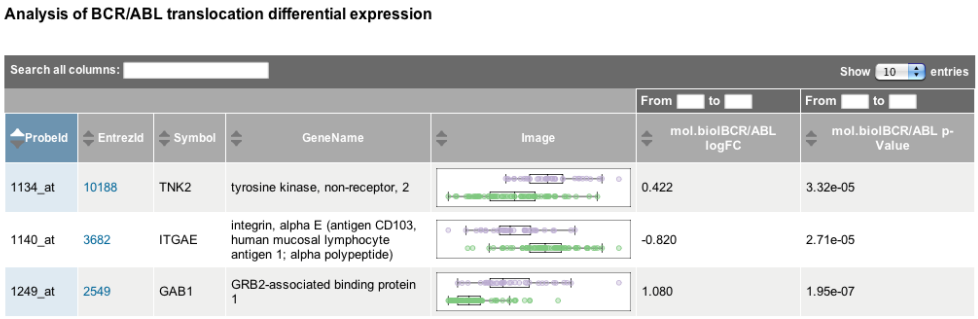
Next we use limma to find statistical evidence of differential genes.

```
> library(limma)
> model <- model.matrix(~mol.biol+sex, ALL)
> fit <- eBayes(lmFit(ALL, model))
```

With the limma output we can make our differential analysis report. To publish MArrayLM objects, we supply the eSet and factor used in our analysis.

```
> library(lattice)
> rep.theme <- reporting.theme()
> lattice.options(default.theme = rep.theme)
> deReport <- HTMLReport(shortName = 'de_analysis',
+     title = 'Analysis of BCR/ABL translocation differential expression',
+     reportDirectory = "./reports/",
+     baseUrl = "")
> publish(fit, deReport, eSet=ALL, factor=ALL$mol.biol, coef=2, n=100)
> finish(deReport)
```

The resulting output is displayed on an .html page and includes several statistics of interest as well as an image of the data.



Analysis of BCR/ABL translocation differential expression

Search all columns: Show 10 entries

ProbeId	EntrezId	Symbol	GeneName	Image	From ___ to ___ mol.biolBCR/ABL logFC	From ___ to ___ mol.biolBCR/ABL p-Value
1134_at	10188	TNK2	tyrosine kinase, non-receptor, 2		0.422	3.32e-05
1140_at	3682	ITGAE	integrin, alpha E (antigen CD103, human mucosal lymphocyte antigen 1; alpha polypeptide)		-0.820	2.71e-05
1249_at	2549	GAB1	GRB2-associated binding protein 1		1.080	1.95e-07

Figure 1: Resulting page created by `publish` for `fit` .

3 GO analysis using GOstats

In this section, we show how to use `ReportingTools` to `publish` a GO analysis to an html file. First we select the top 100 differential genes and then run the `hyperGTest` from the `GOstats` package.

```
> library(GOstats)
> tt <- topTable(fit, coef = 2, n = 100)
> selectedIDs <- unlist(mget(tt$ID, hgu95av2ENTREZID))
> universeIDs <- unlist(mget(featureNames(ALL), hgu95av2ENTREZID))
> goParams <- new("GOHyperGParams",
+     geneIds = selectedIDs,
+     universeGeneIds = universeIDs,
+     annotation = annotation(ALL),
+     ontology = "BP",
+     pvalueCutoff = 0.01,
+     conditional = TRUE,
+     testDirection = "over")
> goResults <- hyperGTest(goParams)
```

With these results, we can then make the GO report. We must supply `publish` with the genes of interest and the species annotation for this dataset. The default p-value cutoff is 0.01 and the minimum category size is 10 genes.

```
> goReport <- HTMLReport(shortName = 'go_analysis',
+     title = 'GO analysis of BCR/ABL translocation',
+     reportDirectory = "./reports/",
+     baseUrl = "")
> publish(goResults, goReport, selectedIDs=selectedIDs, annotation.db="org.Hs.eg")
> finish(goReport)
```

The resulting output is displayed on an .html page and includes several statistics of interest as well as an image of the overlap for each category.

4 PFAM analysis

In this section, we show how to use `ReportingTools` to write a table of PFAM analysis results to an html file. First we run the `hyperGTest` from the `Category` package.

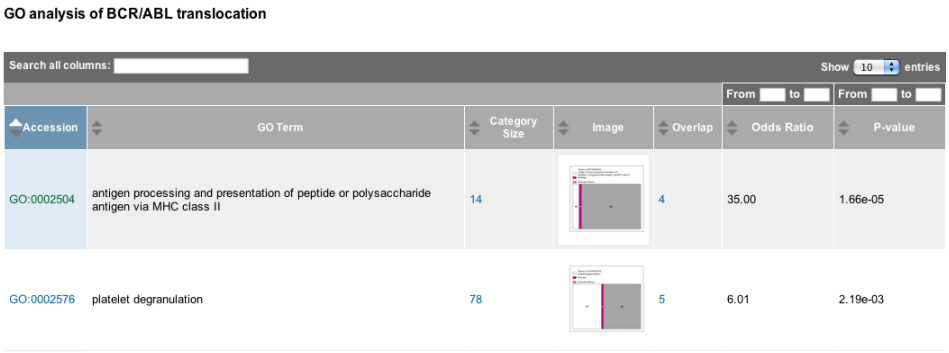


Figure 2: Resulting page created by `publish` for `goResults`.

```
> library(Category)
> pfamParams <- new("PFAMHyperGParams",
+     geneIds = selectedIDs,
+     universeGeneIds = universeIDs,
+     annotation = annotation(ALL),
+     pvalueCutoff = 0.01,
+     testDirection = "over")
> PFAMResults <- hyperGTest(pfamParams)
```

Then we make the PFAM report. Again we supply `publish` with the genes of interest and the species annotation for this dataset. We set the minimum category size to 3 genes.

```
> PFAMReport <- HTMLReport(shortName = 'pfam_analysis',
+     title = 'PFAM analysis of BCR/ABL translocation',
+     reportDirectory = "./reports/",
+     baseUrl = "")
> publish(PFAMResults, PFAMReport, selectedIDs=selectedIDs, annotation.db="org.Hs.eg",categorySize=3)
> finish(PFAMReport)
```

The resulting output is displayed on an .html page and includes several statistics of interest as well as an image of the overlap for each category.

5 GSEA analysis

In this section we show how to use `publish` to display `GeneSetCollection` objects and their corresponding gene set enrichment statistics. For this example, we will randomly select our gene sets and create our collection.

```
> library(GSEAlm)
> library(GSEABase)
> mapped_genes <- mappedkeys(org.Hs.egSYMBOL)
> eidsAndSymbols <- as.list(org.Hs.egSYMBOL[mapped_genes])
> geneEids<-names(eidsAndSymbols)
> set.seed(123)
> set1<-GeneSet(geneIds=sample(geneEids,100, replace=FALSE), setName="set1",
+         shortDescription="This is set1")
```

PFAM analysis of BCR/ABL translocation

Search all columns: Show 10 entries

PFAM ID	PFAM Term	PFAM Size	Image	Overlap	From to (Odds Ratio)	From to (P-value)
PF00993	Class II histocompatibility antigen, alpha domain	5		3	133.00	1.46e-05
PF07654	Immunoglobulin C1-set domain	40		4	9.89	1.10e-03

Figure 3: Resulting page created by `publish` for `PFAMResults`.

```
> set2<-GeneSet(geneIds=sample(geneEids,10, replace=FALSE), setName="set2",
+         shortDescription="This is set2")
> set3<-GeneSet(geneIds=sample(geneEids,37, replace=FALSE), setName="set3",
+         shortDescription="This is set3")
> set4<-GeneSet(geneIds=sample(geneEids,300, replace=FALSE), setName="set4",
+         shortDescription="This is set4")
> geneSets<-GeneSetCollection(c(set1,set2,set3,set4))
```

We can now make a very simple `GeneSetCollection` html table with `ReportingTools` .

```
> geneSetsReport <- HTMLReport(shortName = "gene_sets",
+         title = "Gene Sets",
+         reportDirectory = "./reports",
+         baseUrl = "")
> publish(geneSets, geneSetsReport, annotation.db="org.Hs.eg")
> finish(geneSetsReport)
```

The resulting output is displayed on an .html page and includes the gene sets and links to pages listing the genes within the corresponding set.

Often, investigators would like more information about the enrichment of certain gene sets. Thus, we will proceed with gene set enrichment analysis (GSEA). To begin, we determine the overlap between our sets and our genes of interest by creating an incidence matrix.

```
> mat <- matrix(data=0, ncol=length(universeIDs),nrow=length(geneSets))
> for(i in 1:length(geneSets)){
+         geneIdEntrez<-unlist(geneIds(geneSets[[i]]))
+         mat[i,match(geneIdEntrez, universeIDs)] <- 1
+ }
> colnames(mat) <- universeIDs
> rownames(mat) <- sapply(geneSets, function(x) x@setName)
```

Now we can run the GSEA and obtain set-specific statistics and p-values.

```
> lm<-lmPerGene(ALL, ~mol.biol+sex, na.rm=TRUE)
> GSNorm<-GSNormalize(lm$tstat[2,], mat)
```

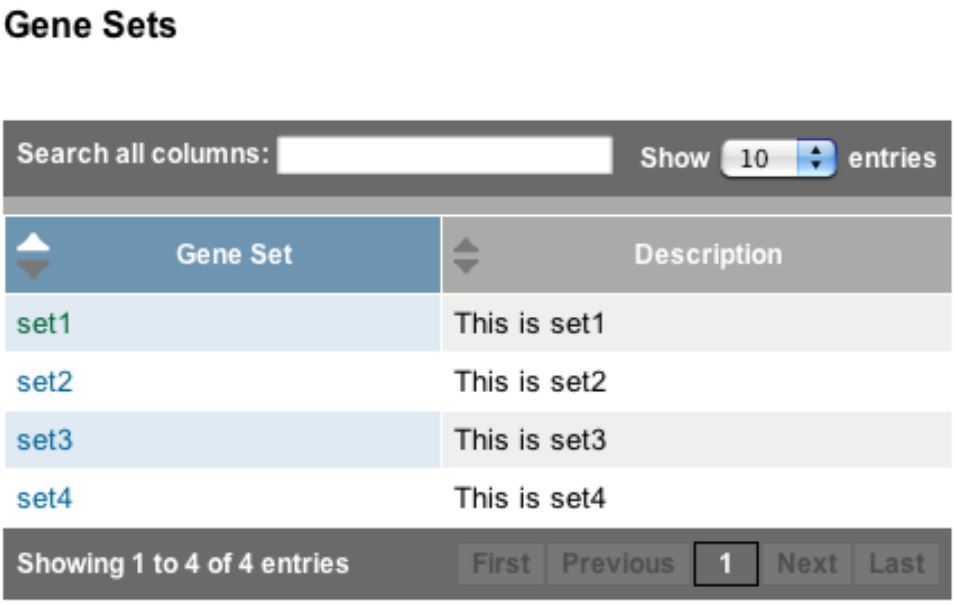


Figure 4: Resulting page created by `publish` for `geneSets`.

```
> #one-sided p-values
> pVals<-gsealmPerm(ALL,~mol.biol+sex, mat, nperm=100)
> bestPval<-apply(pVals,1, min)
```

We can add these statistics to our report page.

```
> gseaReport <- HTMLReport(shortName = "gsea_analysis",
+         title = "GSEA analysis",
+         reportDirectory = "./reports",
+         baseUrl = "")
> publish(geneSets, gseaReport, annotation.db="org.Hs.eg", setStats=GSNorm, setPValues=2*bestPval)
> finish(gseaReport)
```

The resulting output is displayed on an .html page and includes our set statistics and p-values. Links to set-specific pages are also created.

6 Putting it all together

We now make an index page to put all the output together.

```
> indexPage <- HTMLReport(shortName = "index",
+     title = "Analysis of ALL Gene Expression",
+     reportDirectory = "./reports",
+     baseUrl = "")
> publish(c(deReport, goReport, PFAMReport, gseaReport), indexPage)
> finish(indexPage)
```

GSEA analysis

Search all columns: Show 10 entries

		From to	From to
Gene Set	Description	Gene Set Statistic	Gene Set P-value
set1	This is set1	-1.540	0.4360
set2	This is set2	1.970	0.0792
set3	This is set3	0.811	0.4360
set4	This is set4	-0.520	0.8510

Showing 1 to 4 of 4 entries First Previous 1 Next Last

Figure 5: Resulting updated page created by `publish` for `geneSets` after we include the set statistics and p-values.

Genes in set2 -- This is set2

Search all columns: Show 10 entries

EntrezId	Symbol	GeneName
100418508	LOC100418508	MAS1 oncogene-like pseudogene
124842	TMEM132E	transmembrane protein 132E
220296	HEPACAM	hepatic and glial cell adhesion molecule
283416	C12orf61	chromosome 12 open reading frame 61
2841	GPR18	G protein-coupled receptor 18
392232	LOC392232	transient receptor potential cation channel, subfamily A, member 1 pseudogene
400645	RBM22P1	RNA binding motif protein 22 pseudogene 1
767561	SNORD113-1	small nucleolar RNA, C/D box 113-1
80129	C6orf97	chromosome 6 open reading frame 97
85388	SNORD14B	small nucleolar RNA, C/D box 14B

Showing 1 to 10 of 10 entries First Previous 1 Next Last

Figure 6: The set-specific page.

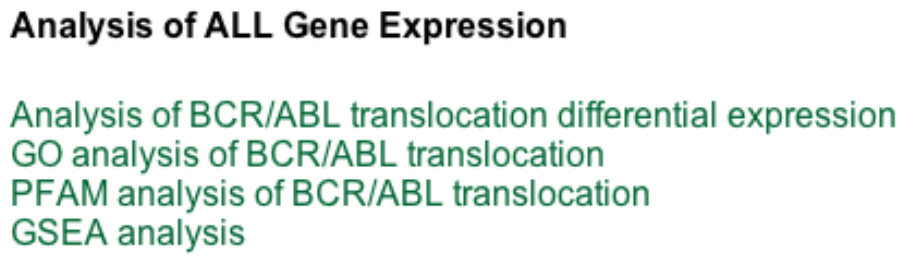


Figure 7: The page created from calling `publish` on all of our previous pages.