Howto get pretty html output for my gene list

October 28, 2003

1 Overview

This document demonstrates how you can get a nice–ish annotated html (web–page) output for the genes or probes that you have selected. To do this you will need both the *biobase* and *annotate* packages. You will also need to obtain annotation data for the experiment itself. In the example here the data come from Affymetrix U95A chips and that annotation is supplied by the Bioconductor project[1].

First we load the *annotate* package (this will automatically load *biobase* as well) and then obtain our test data. We have arbitrarily selected 15 genes of *interest*, you will have obtained your list in some meaningful way!

```
> library(annotate)
> data(eset)
> igenes <- geneNames(eset)[245:260]
> igenes
```

```
 [1] "31484_at"   "31485_at"   "31486_s_at" "31487_at"   "31488_s_at"
 [6] "31489_at"   "31490_at"   "31491_s_at" "31492_at"   "31493_s_at"
[11] "31494_at"   "31495_at"   "31496_g_at" "31497_at"   "31498_f_at"
[16] "31499_s_at"
```

Now, given this set of genes (or Affymetrix identifiers) we would like to provide some meaningful output.

The example data in `eset` has three (made–up) covariates.

```
> eset
```

```
Expression Set (exprSet) with
        500 genes
        26 samples
                 phenoData object with 3 variables and 26 cases
         varLabels
                cov1: Covariate 1; 2 levels
                cov2: Covariate 2; 2 levels
                cov3: Covariate 3; 3 levels
```

[1] If you have other needs please contact us.

We will use `cov2`. The function `esApply` takes an *exprSet* and applies a function to it. So, suppose we are interested in differences in the mean expression for these genes, between the two groups defined by `cov2`. We can obtain the t–tests by using `esApply` First, we make up an function that can be used by `esApply` to do what we want[2].

```
> library(ctest)
> mytt <- function(y) {
+     ys <- split(y, cov2)
+     t.test(ys[[1]], ys[[2]])
+ }
> ttout <- esApply(eset[245:260, ], 1, mytt)
> length(ttout)

[1] 16

> ttout[1]

$"31484_at"

        Welch Two Sample t-test

data:  ys[[1]] and ys[[2]]
t = -0.4822, df = 18.601, p-value = 0.6353
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -43.19992  27.04101
sample estimates:
mean of x mean of y
 145.8687  153.9482

> igenes.pvals <- sapply(ttout, function(x) x$p.value)
> igenes.pvals

  31484_at   31485_at 31486_s_at   31487_at 31488_s_at   31489_at   31490_at
 0.6352895  0.8922614  0.4468717  0.3111451  0.8643903  0.8732372  0.7667428
31491_s_at   31492_at 31493_s_at   31494_at   31495_at 31496_g_at   31497_at
 0.6059114  0.7363639  0.8204156  0.7730418  0.7143153  0.4551546  0.3538565
31498_f_at 31499_s_at
 0.4399743  0.4475066

> igenes.gp1mean <- sapply(ttout, function(x) x$estimate[1])
> igenes.gp2mean <- sapply(ttout, function(x) x$estimate[2])
```

And we see that `ttout` contains the output for the t–tests for the 16 genes. Notice that none of the p–values or test statistics are unusual – that is because

[2]See the `esApply` man pages and HOWTO's for more details on this function

we simply choose genes at random. You should see small p–values and large test statistics (we hope).

Now, to assemble our web page we need to get the annotation data and to then associate the annotation with the t–test results. We first obtain gene symbol and the LocusLink identifier data by loading the appropriate environments using `data`. We extract the data from the environment using either `get` or `multiget`.

```
> data(hgu95All)
> data(hgu95Asym)
> igenes.ll <- multiget(igenes, env = hgu95All)
> igenes.sym <- multiget(igenes, env = hgu95Asym)
```

And now we are ready to wrap this all up using `ll.htmlpage`. This function takes the LocusLink identifiers and wraps them in an HTML anchor that is linked to the appropriate LocusLink web page.

```
> ll.htmlpage(igenes.ll, "HOWTO.igenes", "Genes seleced in an arbitrary way",
+     list(igenes.sym, igenes, round(igenes.gp1mean, 3), round(igenes.gp2mean,
+         3), round(igenes.pvals, 3)))
```

And you can load this page up in your favorite browser. The genes are clickable. An `NA` in the output for LocusLink represents a gene or EST that we have not yet successfully mapped to a LocusLink identifier.