A note on `esApply`

ExpressionSets are complex objects. `exprs(ExpressionSet)` produces $G \times N$, where $G$ is the number of genes on a chip and $N$ is the number of tissues analyzed, and `pData(ExpressionSet)` produces $N \times p$, where $p$ is the number of phenotypic or demographic, etc., variables collected.

Abstractly, we are often interested in evaluating functions $f(y; x)$ where $y$ is an $N$-vector of expression results for a specific gene and $x$ is an $N$-dimensional structure, coordinated with $y$, that distinguishes elements of $y$ for processing in the function $f$. A basic problem is to guarantee that the $j$th element of $y$ is correctly associated with the $j$th component of $x$.

As an example, let’s consider `sample.ExpressionSet`, which is an `ExpressionSet` supplied with Biobase. We will print a little report, then the first $N$-vector of gene expressions and some covariate data:

```r
> print(sample.ExpressionSet)
ExpressionSet (storageMode: lockedEnvironment)
assayData: 500 features, 26 samples
 element names: exprs, se.exprs
phenoData
 sampleNames: A, B, ..., Z (26 total)
 varLabels and varMetadata description:
    sex: Female/Male
    type: Case/Control
    score: Testing Score
featureData
 featureNames: AFFX-MurIL2_at, AFFX-MurIL10_at, ..., 31739_at (500 total)
 fvarLabels and fvarMetadata description: none
experimentData: use 'experimentData(object)'
Annotation: hgu95av2
```

```r
> print(exprs(sample.ExpressionSet)[1, ])

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>192.7420</td>
<td>85.7533</td>
<td>176.7570</td>
<td>135.5750</td>
<td>64.4939</td>
<td>76.3569</td>
<td>160.5050</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>65.9631</td>
<td>56.9039</td>
<td>135.6080</td>
<td>63.4432</td>
<td>78.2126</td>
<td>83.0943</td>
<td>89.3372</td>
</tr>
<tr>
<td></td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
</tr>
<tr>
<td></td>
<td>91.0615</td>
<td>95.9377</td>
<td>179.8450</td>
<td>152.4670</td>
<td>180.8340</td>
<td>85.4146</td>
<td>157.9890</td>
</tr>
<tr>
<td></td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>146.8000</td>
<td>93.8829</td>
<td>103.8550</td>
<td>64.4340</td>
<td>175.6150</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

```r
> print(pData(sample.ExpressionSet)[1:2, 1:3])
```

1
sex type score
A Female Control 0.75
B Male Case 0.40

Now let's see how expressions and a covariate are related:

```r
> print(rbind(exprs(sample.ExpressionSet[, ]), sex <- t(pData(sample.ExpressionSet)) [, ]))

    A         B         C         D         E
AFFX-MurIL2_at 192.742  85.7533 176.757 135.575  64.4939
              Female  Male  Male  Male  Female
F             G         H         I         J
AFFX-MurIL2_at 76.3569 160.505  65.9631 56.9039 135.608
              Male  Male  Female  Male  Male
K             L         M         N         O
AFFX-MurIL2_at 63.4432  78.2126  83.0943  89.3372  91.0615
              Male  Female  Male  Male  Female
P             Q         R         S         T
AFFX-MurIL2_at 95.9377 179.845 152.467 180.834  85.4146
              Female  Female  Male  Male  Female
U             V         W         X         Y
AFFX-MurIL2_at 157.989 146.82  93.8829 103.855  64.434
              Male  Female  Male  Male  Female
Z

A function that evaluates the difference in median expression across strata defined using an abstract covariate x is

```r
> medContr <- function(y, x) {
+  ys <- split(y, x)
+  median(ys[[1]]) - median(ys[[2]])
+}
```

We can apply this to a small ExpressionSet that gives back the data listed above:

```r
> print(apply(exprs(sample.ExpressionSet[, , drop = F]), 1, medContr, pData(sample.ExpressionSet)[["sex"]]))

AFFX-MurIL2_at
-12.7935
```
That’s a bit clumsy. This is where esApply comes in. We pay for some simplicity by following a strict protocol for the definition of the statistical function to be applied.

```r
> medContr1 <- function(y) {
+   ys <- split(y, sex)
+   median(ys[[1]]) - median(ys[[2]])
+ }
> print(esApply(sample.ExpressionSet, 1, medContr1)[1])

AFFX-MurIL2_at
 -12.7935
```

The manual page on esApply has a number of additional examples that show how applicable functions can be constructed and used. The important thing to note is that the applicable functions know the names of the covariates in the pData dataframe.

This is achieved by having an environment populated with all the variables in phenoData(ExpressionSet) put in as the environment of the function that will be applied. If that function already has an environment we retain that but in the second position. Thus, there is some potential for variable shadowing.

1 Session Information

The version number of R and packages loaded for generating the vignette were:

```
R version 2.7.0 (2008-04-22)
x86_64-unknown-linux-gnu
locale:
LC_CTYPE=en_US;LC_NUMERIC=C;LC_TIME=en_US;LC_COLLATE=en_US;LC_MONETARY=C;LC_MESSAGES=en_US
LC_PAPER=en_US;LC_NAME=C;LC_ADDRESS=C;LC_TELEPHONE=C;LC_MEASUREMENT=en_US;LC_IDENTIFICATION=C

attached base packages:
[1] tools stats graphics grDevices utils datasets methods base

other attached packages:
[1] Biobase_2.0.1
```