Solutions for chapter R and Bioconductor Introduction

Exercise 1

a
```
> apropos("plot")
```

```
  [1] ".__C__recordedplot"
  [2] ".__M__KEGGmnplot:annotate"
  [3] ".__M__MAplot:affy"
  [4] "..."
```

b
```
> help.search("mann-whitney")
```

c
```
> library("Biobase")
> openVignette("Biobase")
```

Exercise 2

`sessionInfo` prints version information about R and all loaded packages. This is helpful when posting on one of the R or Bioconductor mailing lists in order to provide detailed information about the software you are using.

```
> sessionInfo()
R version 2.7.0 Under development (unstable) (2007-10-16
 r43183)
i686-pc-linux-gnu

locale:
C

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

other attached packages:
 [1] geneplotter_1.17.3    lattice_0.17-2
 [3] annotate_1.17.3       xtable_1.5-2
 [5] AnnotationDbi_1.1.6   RSQLite_0.6-4
 [7] DBI_0.2-4             hgu95av2cdf_2.0.0
 [9] hgu95av2probe_2.0.0   matchprobes_1.11.0
```

```
[11] CLL_1.2.4             affy_1.17.3
[13] preprocessCore_1.1.3  affyio_1.7.6
[15] RColorBrewer_1.0-2    GO_2.0.1
[17] class_7.2-38          hgu95av2_2.0.1
[19] BiocCaseStudies_1.1.1 Biobase_1.17.5
[21] weaver_1.5.0          codetools_0.1-3
[23] digest_0.3.1

loaded via a namespace (and not attached):
[1] KernSmooth_2.22-21 grid_2.7.0
```

Exercise 3

a
```
> x = c(0.1, 1.1, 2.5, 10)
> y = 1:100
> z = y < 10
> pets = c(Rex="dog", Garfield="cat", Tweety="bird")
```

b Arithmetic expressions in R are vectorized. The operations are performed element by element. If two vectors of unequal length are used in the same expression, R recycles the shorter of the two vectors.

```
> 2 * x + c(1,2)
[1]  1.2  4.2  6.0 22.0
```

c Index vectors can be of type *logical*, *integer* and *character* (for the special case of named vectors).

```
> ##logical
> y[z]
[1] 1 2 3 4 5 6 7 8 9
> ## integer
> y[1:4]
[1] 1 2 3 4
> y[-(1:95)]
[1]  96  97  98  99 100
> ## character
> pets["Garfield"]
Garfield
   "cat"
```

Matrices and arrays can be indexed similar to vectors. Each dimension is separated by a comma in the square brackets.

```
> m = matrix(1:12, ncol=4)
> m[1,3]
[1] 7
```

d List items are selected using the `$` operator or the `[[` operator. The latter accepts all three types of index vectors, the former always interprets its right hand argument literally as a name. Note that `[` returns a list even if only one element is selected. You can use the `[[` operator to get to the content of a single list element. List are created using the `list` function.

```
> l = list(name="Paul", sex=factor("male"), age=35)
> l$name
[1] "Paul"
> l[[3]]
[1] 35
```

e A *matrix* is a rectangular table of elements of equal type. In a *data.frame*, each column may have different type. R matrices and arrays are implemented as vectors with a dimension attribute, data frames as a list of vectors that are all enforced to have the same length, but may be of different type.

Exercise 4

```
> ppc = function(x) paste("^", x, sep="")
```

Exercise 5

```
> myFindMap = function(mapEnv, which) {
      myg = ppc(which)
      a1 = eapply(mapEnv, function(x)
          grep(myg, x, value=TRUE))
      unlist(a1)
  }
```

Exercise 6

a
```
> theEnv = new.env(hash=TRUE)
> theEnv$locations = myFindMap(hgu95av2MAP, 18)
```

b
```
> theEnv$strip = function(x) gsub("18", "", x)
```

c
```
> myExtract = function(env)  env$strip(env$locations)
> myExtract(theEnv)[1:5]
```
```
  420_at 36469_at   808_at   862_at 35817_at
"p11.2"    "q12"  "q21.2"  "q21.3"    "q23"
```

Exercise 7

a
```
> class(pData)
```
```
[1] "data.frame"
```

b
```
> names(pData)
```
```
[1] "gender" "type"   "score"
```

c
```
> sapply(pData, class)
```
```
    gender      type     score
 "factor"  "factor" "numeric"
```

d
```
> pData[c(15, 20), c("gender", "type")]
```
```
  gender type
O Female Case
T Female Case
```
```
> pData[pData$score > 0.8,]
```
```
  gender    type score
E Female    Case  0.93
G   Male    Case  0.96
X   Male Control  0.98
Y Female    Case  0.94
```

Exercise 8

```
> plot(x=x, y=y, log="xy",
       xlab="gene expression sample #1",
       ylab="gene expression sample #3",
       main="scatterplot of expression intensities",
       pch=20)
> abline(a=0, b=1)
```

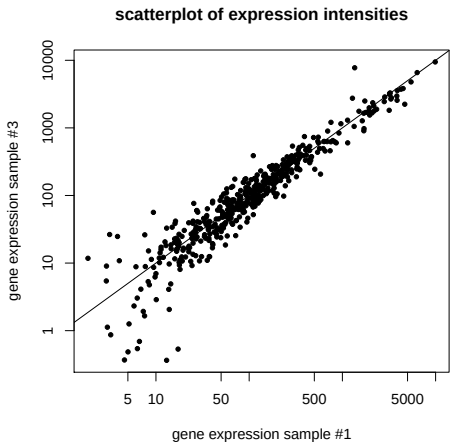

Figure 1. Scatter plot of expression intensities for two samples.

Exercise 9

```
> multiecdf(int ~ gc, data=subset(probedata, gc %in% gcUse),
      xlim=c(6, 11), col=colorfunction(12)[-(1:2)],
      lwd=2, main="", ylab="ECDF")
```

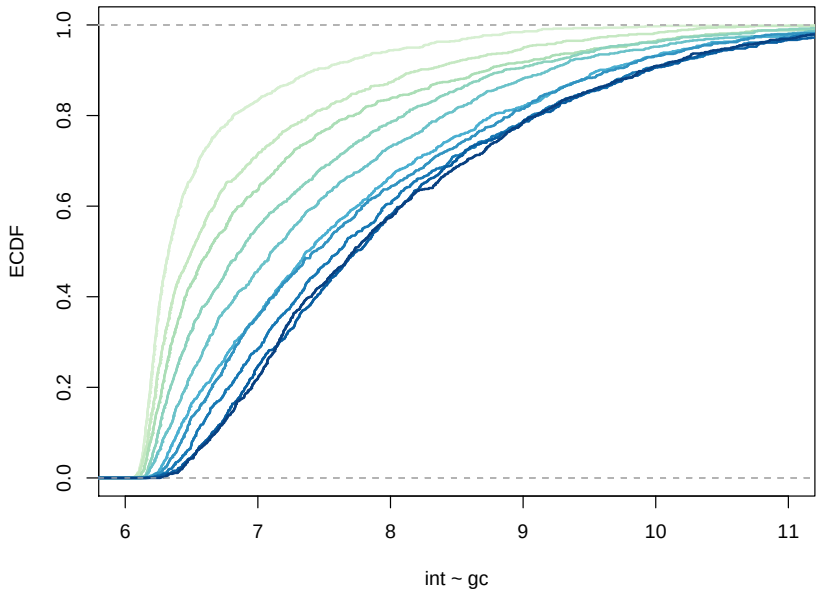

Figure 2. Scatter plot of expression intensities for two samples.