cdfenvs and oligonucleotide arrays

Laurent Gautier

June 13, 2008

Contents

Introduction

This document describes briefly how the package gets (or tries to get) the needed cdfenvs. Various issues like security and configuring the options for the package *affy* are outlined. To some extend, what is developped here could[1] be applied to the package *oligochips*.

As usual, loading the package in your `R` session is required.

```
R> library(affy) ##load the affy package
```

Why and how

The *cdfenvs* are associative data structures to map efficiently a *probeset id* with indices for the corresponding probes. These indices are used to subset a matrix storing probe level intensities. Technically, a *cdfenv* is a **R** `environment`. The functions `get` and `multiget` (in *Biobase*) are a convenient way to access what is in an environment. Expert users only will consider modifying what is those environment.

In the case of *Affymetrix* data, the *cdfenvs* are built from the *.CDF* files. The package *makecdfenvs* is dedicated to the building of packages with *cdfenvs*. The simplest way to proceed is to have the package needed for the analysis installed. A number of *cdfenv* packages for *Affymetrix* chips are available for download on the *Bioconductor* data packages repository. This release (1.2.x) of the *affy* package has an option to allow automatic downloading and install of a *cdfenv* that would be found missing during an analysis. However, if you are using an *unconventional* chip, it is possible that *Bioconductor* has not created the appropriate package for your *.CDF* file. We recommend that you use the package *makecdfenvs* to create the appropriate source code for the *cdfenv* package you need. For Microsoft Windows binaries you need to do more, see `http://www.stats.ox.ac.uk/pub/R/rw-FAQ.html`. If you contribute the package to

[1] Currently this would require a bit of effort, but it should become very easy very soon

Bioconductor a windows binary will be made for you (and everybody else). The last section outlines briefly how to change the associated *cdfenv*.

The complete structure of the options for the package is not completely described, but one can refer the source code for the function `.setAffyOptions()`. The entry *probesloc* in the options details the path used to look for the corresponding *cdfenv*. We introduce a simple function to display the content of the options for the obtention of the information about probe locations:

```
> print.probesloc.opt <- function(affy.opt, fields) {
+     all.fields <- c("what", "where", "autoload", "repository",
+         "installdir")
+     if (sum(!(fields %in% all.fields)) > 0)
+         stop(paste("'fields' can only contain elements of:",
+             paste(all.fields, collapse = " ")))
+     l <- lapply(affy.opt$probesloc, function(x) x[fields])
+     l <- lapply(l, function(x) {
+         unk <- is.na(names(x))
+         x[unk] <- rep(list(unk = NA), sum(unk))
+         x <- lapply(x, function(y) if (is.null(y))
+             "NULL"
+         else y)
+         return(x)
+     })
+     ul <- as.character(unlist(l))
+     m <- t(matrix(ul, nr = length(fields)))
+     colnames(m) <- fields
+     print(m)
+ }
```

The default search path for *cdfenvs* will be:

```
> affy.opt <- getOption("BioC")$affy
> print.probesloc.opt(affy.opt, c("what", "where", "autoload"))
```

```
     what          where                 autoload
[1,] "environment" "<environment>"       "NA"
[2,] "libPath"     "NULL"                "NA"
[3,] "data"        "affy"                "NA"
[4,] "bioC"        "/tmp/Rinst536194906" "NA"
```

The option `autoload` is only relevant where `what` is equal to *package*. Having it set to `TRUE` means that an attempt will be made to download the package if it is not found in `.libPaths()` or in the search path shown above.

The following function returns a set of options with any existing automatic download deactivated :

```
> deactivate.autoload <- function(affy.opt) {
+     l <- lapply(affy.opt$probesloc, function(x) {
+         i <- names(x) == "autoload"
+         x[i] <- list(FALSE)
+         return(x)
+     })
+     affy.opt$probesloc <- l
+     return(affy.opt)
+ }
```

It can be used to deactivate any automatic download:

```
> affy.opt <- getOption("BioC")$affy
> affy.opt.noauto <- deactivate.autoload(affy.opt)
> .setAffyOptions(affy.opt.noauto)
```

Security

The autoload mecanism can be perceived as a security breach. It is the case, but not more than any **R** package you might install (unless you inspect carefully the source of every single package you install).

By default the *cdfenv* packages are downloaded from the bioconductor repository. This can be changed through the options for the package. One might want to check what it is like on his/her local installation:

```
> print.probesloc.opt(affy.opt, c("what", "autoload", "repository"))
```

```
     what          autoload repository
[1,] "environment" NA       NA
[2,] "libPath"     NA       NA
[3,] "data"        NA       NA
[4,] "bioC"        NA       NA
```

If you do not have install permissions, you might consider installing the *cdfenvs* packages to a particular place. For that you need to change the *installdir* (passed to the function `install.packages`).

```
> my.installdir <- "mydir/is/here"
> has.installdir <- unlist(lapply(affy.opt$probesloc, function(x) if ("installdir" %i
+     names(x)) grep("installdir", names(x)) else numeric(0)))
> l <- lapply(affy.opt$probesloc, function(x) {
+     if ("installdir" %in% names(x)) {
+         x$installdir <- my.installdir
```

```
+     }
+     return(x)
+ })
> affy.opt$probesloc <- l
> .setAffyOptions(affy.opt)
```

Note that a given *cdfenv* package is searched as specified in the argument *where*. Depending on your settings, you might prefer updating the `.libPaths()` of your **R** session, or the field *where* for the probelocs).

Changing the associated *cdfenv*

If you do not care about mapping *probeset id*s with indices and only want to deal with probe intensities, or if you would like to make a custom *cdfenv* from scratch, you will have to modify the association between your `AffyBatch` object(s) and *cdfenvs*.

If you want to experiment with this feature of the package, we would recommend to create a dummy *cdfenv* and associate your `AffyBatch` to it:

```
> data(affybatch.example)
> dummymap.name <- "dummymap"
> assign(dummymap.name, new.env())
> affybatch.example@cdfName <- paste(dummymap.name, "cdf", sep = "")
```

As explained in the first part, the *affy* package will then find a matching environment in the current environment (`.GlobaEnv`)and will not try to find a package. Note: the environment should still be in the section *probelocs* of the package options.