Package: makecdfenv (via r-universe)

November 1, 2026

Version 1.88.0

Date 2006-03-06

Title CDF Environment Maker

Author Rafael A. Irizarry <rafa@jhu.edu>, Laurent Gautier <laurent@cbs.dtu.dk>, Wolfgang Huber <w.huber@dkfz-heidelberg.de>, Ben Bolstad <bmb@bmbolstad.com>

Maintainer James W. MacDonald <jmacdon@u.washington.edu>

Depends R (>= 2.6.0), affyio

Imports Biobase, affy, methods, stats, utils

Description This package has two functions. One reads a Affymetrix chip description file (CDF) and creates a hash table environment containing the location/probe set membership mapping. The other creates a package that automatically loads that environment.

License GPL (>= 2)

LazyLoad yes

biocViews OneChannel, DataImport, Preprocessing

Config/pak/sysreqs zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:30:06 UTC

RemoteUrl https://github.com/bioc/makecdfenv

RemoteRef RELEASE_3_23

RemoteSha 1ae1a3cf4b27c9035ae57f9d435c1692e6415836

Contents

Cdf-class 2
getInfoInFile 3
make.cdf.env 4
make.cdf.package 6
pmormm 7
read.cdffile 8

Index **10**

`Cdf-class` *Class Cdf*

Description

A class describing the content a Chip Description File.

Details

The class `Cdf` was designed to store the content of Affymetrix's Chip Definition Files (CDF). This early class is no longer widely used in the package affy. Environments (called `cdfenvs`) are preferred to have an efficient mapping between probe set identifiers and indexes (needed to access/subset particular probe intensities). Unless one needs to access every information contained in a CDF file, the `cdfenvs` will be preferred.

The following comments are only relevant to someone with interest in what is in a CDF file. The name associated to each probe is not unique, it corresponds to the gene name. It is very common to have a name repeated 40 times (20 perfect matches and 20 mismatches). Probes can be grouped by pairs: a perfect match (PM) probe has its mismatch (MM) counterpart. The two probes in a pair differ by one base (usually located in the middle of the sequence). The information relative to this particular base are stored in `pbase` and `tbase`. At a given position (x,y), having pbase[x,y] == tbase[x,y] means having a MM while having pbase[x,y] != tbase[x,y] means having a perfect mismatch if and only if one of the bases is A while the other is TRUE, or one is G and the other is C. The function `pmormm` returns `TRUE` for PMs and `FALSE` for MMs.

To know more about the file structure of a CDF file, one has to refer to the parsing code (mostly in C).

Creating Objects

```
new('Cdf',
cdfName = ...., # Object of class character
name = ...., # Object of class matrix
name.levels = ...., # Object of class character
pbase = ...., # Object of class matrix
pbase.levels = ...., # Object of class character
tbase = ...., # Object of class matrix
tbase.levels = ...., # Object of class character
atom = ...., # Object of class matrix
)
```

Slots

`cdfName`**:** the CDF name tag. Used to link with the right CEL files

`name`**:** Object of class "matrix" of 'factors' for the gene names corresponding to the probes.

`name.levels`**:** Object of class "character" containing the levels corresponding to `name`.

`pbase`**:** Object of class "matrix" of pbase levels.

`pbase.levels`: Object of class "character" containing the levels corresponding to `pbase`.

`tbase`: Object of class "matrix" of tbase levels.

`tbase.levels`: Object of class "character" containing the levels corresponding to `tbase`.

`atom`: Object of class "matrix" of atom or probe numbers.

Methods

atom (Cdf): An accesor function for the `ato,` slot.

atom<- (Cdf): A replacement function for `atom` slot.

name.levels (Cdf): An accesor function for the `name.levels` slot.

name.levels<- (Cdf): A replacement function for `name.levels` slot.

pbase (Cdf): An accesor function for the `pbase` slot.

pbase<- (Cdf): A replacement function for `pbase` slot.

pbase.levels (Cdf): An accesor function for the `pbase.levels` slot.

pbase.levels<- (Cdf): A replacement function for `pbase.levels` slot.

show (Cdf): renders information about the `Cdf` object in a concise way on stdout.

tbase (Cdf): An accesor function for the `tbase` slot.

tbase<- (Cdf): A replacement function for `tbase` slot.

tbase.levels (Cdf): An accesor function for the `tbase.levels` slot.

tbase.levels<- (Cdf): A replacement function for `tbase.levels` slot.

Author(s)

L. Gautier <laurent@cbs.dtu.dk>

See Also

`read.cdffile`, `make.cdf.env`, `make.cdf.package`

`getInfoInFile` *get information from Affymetrix data files*

Description

Get specific information stored in Affymetrix data files (CEL or CDF).

Usage

```
getInfoInFile(filename, type, unit, property, compress = NULL)
```

Arguments

filename	a file name
type	"CEL" or "CDF" are the only known types.
unit	the 'unit' to find the information
property	the 'property' of interest
compress	a boolean

Details

The data files seem to have a structure of 'units'. A 'unit' start with something between square brackets (ex: "[UNIT1]"). The parameter unit let one specify in which unit the information of interest is located. The 'properties' are constituted of an identifier, the sign '=' and the value for the property (ex: "numCells=12000").

If you are planning to use this function, I assume you know the what is in CEL and CDF files in details and that you know what are doing.

Value

a character with the value of the 'property'.

Note

The code goes through the file until the rigth 'unit' is found. Then it looks for the right 'property' further down (eventually going through the next units if the property is not found).

See Also

read.celfile, read.cdffile, whatcdf

---

make.cdf.env *CDF Environment Maker*

Description

Reads an Affymetrix chip description file (CDF) and creates an environment used as a hash table for the probe set mapping to location.

Usage

```
make.cdf.env(filename,
             cdf.path = getwd(),
             compress = FALSE,
             return.env.only = TRUE,
             verbose = TRUE)
```

Arguments

filename	Character. Filename of the CDF file - **without** the path prefix!
cdf.path	Character. Path to the CDF file.
compress	Logical. If TRUE, CDF file is compressed.
return.env.only	Logical. If TRUE (the default), then the function returns an environment. Otherwise, a list with two elements, the first being the environment, and the second being a data structure that contains additional information needed for the package builder (see details, and vignette).
verbose	Logical. If TRUE, messages are shown.

Details

Normally, this function should not be called directly. The preferred way to handle CDF information is to use make.cdf.package to build a package, and to install it into R. The CDF information can then either be invoked automatically by the package affy, or can be loaded manually by calling, for example, library(hgu133a).

Some R installations (typically on Windows) do not offer all the tools that are necessary for package building. In such situations, this function may be called directly. Please see the vignette for details - type:

```
openVignette("makecdfenv")
```

Return values: env is an environment, used as a hash table. For every probe set name we have a matrix with 2 columns. The first column contains the PM locations and the second column the MM locations. For PM only chips the MM column will have NAs.

syms is a list that contains chip-specific (i.e., CDF-file specific) information that can be used in the construction of the help files for the CDF package.

Value

Depending on the argument return.env.only, either the environment, or a list with two elements, env and syms. See details.

Author(s)

Rafael A. Irizarry, Wolfgang Huber

See Also

make.cdf.package

Examples

```
env <- make.cdf.env("Hu6800.CDF.gz",
         cdf.path=system.file("extdata", package="makecdfenv"),
         compress=TRUE)
length(ls(env))
get("U53347_at", env)
```

make.cdf.package *CDF Environment Package Maker*

Description

This function reads an Affymetrix chip description file (CDF) and creates an R package that when loaded has the CDF environment available for use.

Usage

```
make.cdf.package(filename,
  packagename  = NULL,
  cdf.path     = getwd(),
  package.path = getwd(),
  compress     = FALSE,
  author       = "The Bioconductor Project",
  maintainer   = "Biocore Package Maintainer <maintainer@bioconductor.org>",
  version      = packageDescription("makecdfenv", fields ="Version"),
  species      = NULL,
  unlink       = FALSE,
  verbose      = TRUE)
```

Arguments

filename	Character. Filename of the CDF file - **without** the path prefix!
packagename	Character. Name wanted for the package.
cdf.path	Character. Path to the CDF file.
package.path	Character. Path where the package will be created.
compress	Logical. If TRUE, CDF file is compressed.
author	Character. What to put in the author field of the package.
maintainer	Character. What to put in the maintainer field of the package.
version	Character. What to put in the version field. Should be a of the form x.x.x.
species	Character. Must be specified using the format e.g., Homo_sapiens
unlink	Logical. If TRUE, and a package directory exists already in package.path, that is overwritten.
verbose	Logical. If TRUE messages are shown.

Details

The function is called for its side effect, creating a package. By default the package name will be the name of the CDF file made lower case and with special characters removed (i.e. only alpha-numeric).

In general one would want to use the name given in by `cleancdfname(abatch@cdfName)` with `abatch` an `AffyBatch` object obtained, for example, using `ReadAffy`. This is the package name that the `affy` package looks for by default.

If the user has a CEL file, called `filename`, the recommended package name for the environment is `cleancdfname(whatcdf(filename))`. This usually coincides with the default.

Please see the vignette for more details.

Value

If success, the function returns the name of the created package.

Author(s)

Rafael A. Irizarry, Wolfgang Huber

See Also

`make.cdf.env`

Examples

```
pkgpath <- tempdir()
make.cdf.package("Hu6800.CDF.gz",
         cdf.path=system.file("extdata", package="makecdfenv"),
         compress=TRUE, species = "Homo_sapiens",
  package.path = pkgpath)
dir(pkgpath)
```

`pmormm` *PM or MM*

Description

Determing if intensities on an array from the cdffile are PM, MM, or neither.

Usage

```
pmormm(cdf)
```

Arguments

`cdf`	A `Cdf` object

Details

The intensities on a chip fall in three categories: perfect match (PM), mismatch (MM) or unknown (No information about them is contained in the CDF file, yet they were found informative (details to come)). PM are coded as TRUE, MM as FALSE and unknown as NA.

Value

A matrix (see section 'details').

Author(s)

L. Gautier <laurent@cbs.dtu.dk>

`read.cdffile` *Read a CDF file*

Description

Read the data contained in a CDF file

Usage

```
read.cdffile(file, compress=FALSE)
```

Arguments

`file`	the name of the CDF file
`compress`	whether the file is compressed or not

Details

This function is intended for use by `make.cdf.env`, which in turn is called by `make.cdf.package`. User may not have much benefit from calling this function directly.

In order to save memory, the name corresponding to each value in the CEL is a factor. As in R `factor` objects cannot be also of type `matrix`, the names corresponding to the indices were stored in a vector of type `character` called `name.levels`. The same thing was done with the pbase and cbase information (more for consistency than by conviction that some memory could be saved here).

Value

Returns a `Cdf-class` object.

Note

A `Cdf` object is not a cdf environment, which will be needed for the computation of expression values from the probe intensities in a `AffyBatch`.

Author(s)

Laurent Gautier (laurent@cbs.dtu.dk)

See Also

`Cdf-class`

Examples

```
fn <- system.file("extdata", "Hu6800.CDF.gz", package="makecdfenv")
mycdf <- read.cdffile(fn, compress=TRUE)
mycdf
```

Index

∗ **file**
    read.cdffile, 8
∗ **manip**
    getInfoInFile, 3
    make.cdf.env, 4
    make.cdf.package, 6
    pmormm, 7
∗ **methods**
    Cdf-class, 2

AffyBatch, *7*
atom *(*Cdf-class*)*, 2
atom,Cdf-method *(*Cdf-class*)*, 2
atom<- *(*Cdf-class*)*, 2
atom<-,Cdf-method *(*Cdf-class*)*, 2

Cdf-class, 2

getInfoInFile, 3

make.cdf.env, *3*, 4, *7, 8*
make.cdf.package, *3*, *5*, 6, *8*

name.levels *(*Cdf-class*)*, 2
name.levels,Cdf-method *(*Cdf-class*)*, 2
name.levels<- *(*Cdf-class*)*, 2
name.levels<-,Cdf-method *(*Cdf-class*)*, 2

pbase *(*Cdf-class*)*, 2
pbase,Cdf-method *(*Cdf-class*)*, 2
pbase.levels *(*Cdf-class*)*, 2
pbase.levels,Cdf-method *(*Cdf-class*)*, 2
pbase.levels<- *(*Cdf-class*)*, 2
pbase.levels<-,Cdf-method *(*Cdf-class*)*, 2
pbase<- *(*Cdf-class*)*, 2
pbase<-,Cdf-method *(*Cdf-class*)*, 2
pmormm, *2*, 7

read.cdffile, *3*, 8
ReadAffy, *7*

show,Cdf-method *(*Cdf-class*)*, 2

tbase *(*Cdf-class*)*, 2
tbase,Cdf-method *(*Cdf-class*)*, 2
tbase.levels *(*Cdf-class*)*, 2
tbase.levels,Cdf-method *(*Cdf-class*)*, 2
tbase.levels<- *(*Cdf-class*)*, 2
tbase.levels<-,Cdf-method *(*Cdf-class*)*, 2
tbase<- *(*Cdf-class*)*, 2
tbase<-,Cdf-method *(*Cdf-class*)*, 2