Package 'aroma.apd'

May 7, 2026

Version 0.7.1

Depends R (>= 3.0.0)

Imports R.methodsS3 (>= 1.7.1), R.oo (>= 1.23.0), R.utils (>= 2.2.0), R.huge (>= 0.9.0)

Suggests affxparser

SuggestsNote Recommended: affxparser

Title A Probe-Level Data File Format Used by 'aroma.affymetrix' [deprecated]

Description DEPRECATED. Do not start building new projects based on this package. (The (in-house) APD file format was initially developed to store Affymetrix probe-level data, e.g. normalized CEL intensities. Chip types can be added to APD file and similar to methods in the affxparser package, this package provides methods to read APDs organized by units (probesets). In addition, the probe elements can be arranged optimally such that the elements are guaranteed to be read in order when, for instance, data is read unit by unit. This speeds up the read substantially. This package is supporting the Aroma framework and should not be used elsewhere.)

License LGPL (>= 2.1)

URL https://www.aroma-project.org/,

https://github.com/HenrikBengtsson/aroma.apd

BugReports https://github.com/HenrikBengtsson/aroma.apd/issues

LazyLoad TRUE

biocViews Microarray, DataImport

NeedsCompilation no

Author Henrik Bengtsson [aut, cre, cph]

Maintainer Henrik Bengtsson <henrikb@braju.com>

Repository CRAN

Date/Publication 2025-04-06 17:20:01 UTC

Contents

`aroma.apd-package` *Package aroma.apd*

Description

This package has been deprecated. Do not start building new projects based on it.

DEPRECATED. Do not start building new projects based on this package. (The (in-house) APD file format was initially developed to store Affymetrix probe-level data, e.g. normalized CEL intensities. Chip types can be added to APD file and similar to methods in the affxparser package, this package provides methods to read APDs organized by units (probesets). In addition, the probe elements can be arranged optimally such that the elements are guaranteed to be read in order when, for instance, data is read unit by unit. This speeds up the read substantially. This package is supporting the Aroma framework and should not be used elsewhere.)

Requirements

This package requires the packages **R.huge** and **affxparser**.

Installation and updates

To install this package, see `https://www.braju.com/R/`.

To get started

To get started, see:

1. readApd(), readApdUnits(), readApdRectangle() - Reads APD files.
2. celToApd() - creates an APD file from a CEL file.
3. cdfToApdMap() - creates an APD read map from a CDF file.
4. findApdMap() - finds an APD read map.
5. createApd(), writeApd() - creates APD files.
6. updateApd(), updateApdUnits() - updates APD files.

Search paths

Typically you do not have to specify the pathname of the CDF file when reading APD files or similar. Instead, the chip type is obtained from the APD header and the corresponding APD file is search for in several predefined locations. For details how to specify the search path, see findCdf.

Some APD files uses a corresponding read map in order to read data faster. The findApdMap() method is used to find the corresponding APD map file containing the map vector. How to specify search paths for map files, see that method.

How to cite this package

Currently no information.

License

The releases of this package is licensed under LGPL version 2.1 or newer.

The development code of the packages is under a private licence (where applicable) and patches sent to the author fall under the latter license, but will be, if incorporated, released under the "release" license above.

References

[1] H. Bengtsson, *The R.oo package - Object-Oriented Programming with References Using Standard R Code*, In Kurt Hornik, Friedrich Leisch and Achim Zeileis, editors, Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003), March 20-22, Vienna, Austria. https://www.r-project.org/conferences/DSC-2003/Proceedings/

Author(s)

Henrik Bengtsson

cdfToApdMap *Generates an APD read map file from an Affymetrix CDF file*

Description

Generates an APD read map file from an Affymetrix CDF file.

Usage

```
## Default S3 method:
cdfToApdMap(filename, mapType=NULL, mapFile=NULL, mapPath=NULL, ..., verbose=FALSE)
```

Arguments

filename	The pathname of the CDF file.
mapType	A character string naming the type of the map. If NULL, the chip type will be used.
mapFile	The filename of the resulting APD map file. If NULL, the map type with filename extension apm will be used.
mapPath	An optional path where to the map file will be stored.
...	Additional arguments passed to readCdfUnitsWriteMap, e.g. units.
verbose	A Verbose object.

Value

Returns (invisibly) a list structure with elements:

pathname	The pathname of the generated APD map file.
mapType	The map type character string.
chipType	The chip type character string.
readMap	The generated read map integer vector.

Author(s)

Henrik Bengtsson

See Also

To read an APD map file, see readApdMap().

celToApd *Generates an APD file from an Affymetrix CEL file*

Description

Generates an APD file from an Affymetrix CEL file.

Usage

```
## Default S3 method:
celToApd(filename, apdFile=NULL, mapType="asChipType", writeMap=NULL, ..., verbose=FALSE)
```

Arguments

filename	The pathname of the CEL file.
apdFile	An optional pathname of the APD file, otherwise it will be the same as the CEL file with extension replaced with 'apd'.
mapType	The type of read map for the generated APD file. If NULL, no remapping of the cell indices is done. If "asChipType", the map type is the same as the chip type of the CEL file. If any other character string, it sets the map type to that string. Note that there must be a APD map file with this type that can be found by findApdMap().
writeMap	An optional *write* map integer vector used to remap the cell indices for optimal reading speed. If NULL, the write map may be obtained from the read map file specified by the mapType argument.
...	Additional arguments passed to writeApd().
verbose	A Verbose object.

Value

Returns (invisibly) the pathname of the written APD file.

Author(s)

Henrik Bengtsson

See Also

To create an APD map file from a CDF file, see cdfToApdMap(). To read an APD file, see readApd(). To read an APD map file, see readApdMap().

Examples

```
library("R.utils")  ## Arguments

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 1. Scan for existing CEL files
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# a) Scan for CEL files
files <- list.files(pattern="[.](cel|CEL)$")
files <- files[!file.info(files)$isdir]
if (length(files) > 0 && require("affxparser")) {
  cat("Create an optimal read map for CEL file:", files[1], "\n")
  cdffile <- findCdf(readCelHeader(files[1])$chiptype)
  res <- cdfToApdMap(cdffile)

  cat("Converting CEL file to APD file:", files[1], "\n")
  apdfile <- celToApd(files[1])
  cat("Created APD file:", apdfile, "\n")
  file.remove(apdfile)

  cat("Converting CEL file to APD file with an optimized read map:", files[1], "\n")
  apdfile <- celToApd(files[1], mapType=res$mapType)
  cat("Created APD file:", apdfile, "\n")

  writeMap <- invertMap(res$readMap)
  for (file in files[-1]) {
    cat("Converting CEL file to APD file with an optimized read map:", file, "\n")
    apdfile <- celToApd(file, mapType=res$mapType, writeMap=writeMap)
    cat("Created APD file:", apdfile, "\n")
  }
} # if (length(files) > 0)
```

---

createApd *Creates an Affymetrix probe data (APD) file*

Description

Creates an Affymetrix probe data (APD) file.

An Affymetrix probe data (APD) structure can hold a header and a numeric data vector. Since the APD structure is kept on file all the time, the number of elements in the data vector is only limited by the file system and not the amount of system memory available. For more details, see the FileVector class (and its superclass), which is used internally.

Usage

```
## Default S3 method:
createApd(filename, nbrOfCells, dataType=c("float", "double", "integer", "short",
  "byte"), chipType=NULL, mapType=NULL, ..., verbose=FALSE, .checkArgs=TRUE)
```

Arguments

filename	The filename of the APD file.
nbrOfCells	The number of cells (probes) data elements the APD file structure should hold.
dataType	The data type of the data elements.
chipType	An (optional) character string specifying the chip type.
mapType	An (optional) character string specifying the probe-index map. Use by findApdMap() to find read map.
...	Additional named arguments added to the header of the APD file structure.
verbose	See Verbose.
.checkArgs	If TRUE, arguments are checked, otherwise not.

Value

Returns (invisibly) the pathname of the file created.

Data type

Valid data types are: byte (1 byte), short (2 bytes), integer (4 bytes), float (4 bytes), and double (8 bytes).

Note that in Affymetrix CEL files, the probe intensities as well as the standard deviations are stored as floats (4 bytes) and not doubles (8 bytes). This is why, the default data type is "float".

Author(s)

Henrik Bengtsson

See Also

updateApd() and readApd(). To find a map of a certain type, see findApdMap().

Examples

```
# Float precision
.Machine$float.eps <- (2^((8-4)*8)*.Machine$double.eps)
tol <- .Machine$float.eps ^ 0.5


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 1. Create an Affymetrix Probe Signal (APD) file for a
#    'Mapping50K_Hind240' with 1600-by-1600 probes.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
chipType <- "Mapping50K_Hind240"
nbrOfCells <- 1600^2

pathname <- paste(tempfile(), "apd", sep=".")
createApd(pathname, nbrOfCells=nbrOfCells, chipType=chipType)
```

```
# File size
cat("File name:", pathname, "\n")
cat("File size:", file.info(pathname)$size, "bytes\n")
cat("APD header:\n")
header <- readApdHeader(pathname)
print(header)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 2. Update the signals for a subset of probes
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
cells <- c(1, 1100:1120, 220:201, 998300:999302)
signals <- log(cells+1, base=2)  # Fake signals
updateApd(pathname, indices=cells, data=signals)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 3. Re-read the signals for a subset of probes
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
apd <- readApd(pathname, indices=cells)

# Signals in APD files are stored as floats (since this is
# the precision in CEL files).
stopifnot(all.equal(signals, apd$intensities, tolerance=tol))


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 4. Re-read the signals for a subset of probes
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
if (require("affxparser") && FALSE) {
  cdfFile <- findCdf(chipType)
  if (length(cdfFile) > 0) {

    apd <- readApdUnits(pathname, units=100:104)

    # Sample new data (with one decimal precision)
    apd2 <- lapply(apd, function(unit) {
      lapply(unit, function(groups) {
        n <- length(groups$intensities)
        values <- as.integer(runif(n, max=655350))/10
        list(intensities=values)
      })
    })

    # Update APD file with new data
    updateApdUnits(pathname, units=100:104, data=apd2)

    # Re-read data to verify correctness
    apd <- readApdUnits(pathname, units=100:104)

    # Signals in APD files are stored as floats (since this is
    # the precision in CEL files).
    stopifnot(all.equal(apd2, apd, tolerance=tol))
```

```
  } # if (length(cdfFile) > 0 ...)
} # if (require("affxparser"))


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 4. Clean up
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
file.remove(pathname)
```

`findApdMap` *Search for APD map files in multiple directories*

Description

Search for APD map files in multiple directories.

Usage

```
## Default S3 method:
findApdMap(mapType=NULL, paths=NULL, pattern="[.](a|A)(p|P)(m|M)$", ...)
```

Arguments

`mapType`	A `character` string of the map type to search for.
`paths`	A `character vector` of paths to be searched. The current directory is always searched at the beginning. If `NULL`, default paths are searched. For more details, see below.
`pattern`	A regular expression file name pattern to match.
`...`	Additional arguments passed to `findFiles`.

Details

Note, the current directory is always searched at the beginning. This provides an easy way to override other files in the search path. If `paths` is `NULL`, then a set of default paths are searched. The default search path is consituted of:

1. `"."`
2. `getOption("AFFX_APD_PATH")`
3. `Sys.getenv("AFFX_APD_PATH")`

One of the easiest ways to set system variables for R is to set them in an `.Renviron` file, see `Startup` for more details.

Value

Returns a `vector` of the full pathnames of the files found.

Author(s)

Henrik Bengtsson

`gtypeCelToPQ` *Function to immitate Affymetrix' gtype_cel_to_pq software*

Description

Function to immitate Affymetrix' gtype_cel_to_pq software.

Usage

```
## Default S3 method:
gtypeCelToPQ(filename, units=NULL, ..., cdf=NULL, nbrOfQuartets=NULL, verbose=FALSE)
```

Arguments

`filename`	The name of a CEL file.
`units`	Indices of CDF units to be returned.
`...`	Arguments passed to `readCelUnits`.
`cdf`	A CDF `list` structure, the pathname of the CDF file, or `NULL`. If `NULL`, the CDF file corresponding to the chip type of the CEL file is searched for using `findCdf`.
`nbrOfQuartets`	The number of probe quartets in the returned `matrix`.
`verbose`	See `Verbose`.

Value

Returns an NxK `matrix` where N is the number of probesets (SNPs) and K=4*Q where Q is the number of probe quartets (PMA,MMA,PMB,MMB). The rownames corresponds to the probeset names.

Author(s)

Henrik Bengtsson

References

[1] Affymetrix, *Genotyping Probe Set Structure*, Developers' Network, White paper, 2005-2015.

See Also

`gtypeCelToPQ`(). `applyCdfGroups`.

Examples

```
# Scan for CEL files
files <- list.files(pattern="[.](cel|CEL)$")

# Convert each to RAW file
for (file in files) {
  rawFile <- gsub("[.][^.]*$", ".raw", file)
  file.remove(rawFile)
  cel <- gtypeCelToPQ(file, verbose=TRUE)
  write.table(file=rawFile, cel, sep="\t", quote=FALSE)
}
```

---

readApd *Reads an Affymetrix probe data (APD) file*

Description

Reads an Affymetrix probe data (APD) file.

Usage

```
## Default S3 method:
readApd(filename, indices=NULL, readMap="byMapType", name=NULL, ..., verbose=FALSE,
  .checkArgs=TRUE)
```

Arguments

filename	The filename of the APD file.
indices	An optional numeric vector of cell (probe) indices specifying what cells to read. If NULL, all are read.
readMap	A vector remapping cell indices to file indices. If "byMapType", the read map of type according to APD header will be search for and read. It is much faster to specify the read map explicitly compared with searching for it each time. If NULL, no map is used.
name	The name of the data field. If NULL, the APD header name is used. If not specified, it defaults to "intensities".
...	Not used.
verbose	See Verbose.
.checkArgs	If TRUE, arguments are validated, otherwise not.

Details

To read one *large* contiguous block of elements is faster than to read individual elements one by one. For this reason, internally more elements than requested may be read and therefore allocation more memory than necessary. This means, in worst case N elements may read allocation $N * 8$ bytes of R memory, although only two elements are queried. However, to date even with the largest arrays from Affymetrix this will still only require tens of megabytes of *temporary* memory. For instance, Affymetrix Mapping 100K arrays holds 2,560,000 probes requiring 20Mb of temporary memory.

Value

A named `list` with the two elements `header` and `data`. The header is in turn a `list` structure and the second is a `numeric vector` holding the queried data.

Remapping indices

Argument `readMap` can be used to remap indices. For instance, the indices of the probes can be reorder such that the probes within a probeset is in a contiguous set of probe indices. Then, given that the values are stored in such an order, when reading complete probesets, data will be access much faster from file than if the values were scatter all over the file.

Example of speed improvements. Reading all 40000 values in units 1001 to 2000 of an Affymetrix Mapping 100K Xba chip is more than 10-30 times faster with mapping compared to without.

File format

The file format of an APD file is identical to the file format of an `FileVector`.

Author(s)

Henrik Bengtsson

See Also

`createApd`() and `updateApd`(). See also `readApdHeader`(). To create a cell-index read map from an CDF file, see `readCdfUnitsWriteMap`.

Examples

```
## Not run: #See ?createApd for an example.
```

readApdHeader *Reads the header of an Affymetrix probe data (APD) file*

Description

Reads the header of an Affymetrix probe data (APD) file.

Usage

```
## Default S3 method:
readApdHeader(filename, ..., verbose=FALSE, .checkArgs=TRUE)
```

Arguments

filename	The filename of the APD file.
...	Not used.
verbose	See Verbose.
.checkArgs	If TRUE, arguments are validated, otherwise not.

Details

The file format of an APD file is identical to the file format of an FileVector. Most elements of the APD header are stored in the comment character string of the file vector's header. The APD header nbrOfProbes is identical to the length of the file vector, and is *not* stored in the above comment string.

Value

A named list. The numeric element nbrOfProbes is the number of probe values available in the APD file. The optional character element name specifies the name of the APD vector. The optional character element chipType specifies the chip type, cf. the same field in readCelHeader. The optional character element maptype specifies the type of probe-index map for this APD file. Its value can be used to find the mapping file, see findApdMap() and readApdMap(). All other fields are optional and character values.

Author(s)

Henrik Bengtsson

See Also

readApd().

Examples

```
## Not run: #See ?createApd for an example.
```

readApdMap *Reads an APD probe map file*

Description

Reads an APD probe map file.

Usage

```
## Default S3 method:
readApdMap(filename, path=NULL, ...)
```

Arguments

filename	The filename of the APD file.
path	The path to the APD file.
...	Arguments passed to readApd().

Value

A named list with the two elements header and map. The header is in turn a list structure and the second is a numeric vector holding the probe map indices.

File format

The file format of an APD map file is identical to the file format of an APD file, see readApd(). The APD map file identified by the name of the data defaults to "map". If not, an error is thrown.

Author(s)

Henrik Bengtsson

See Also

To search for an APD map file, see findApdMap(). To create a cell index map from an CDF file, see readCdfUnitsWriteMap. Internally, readApd() is used.

readApdRectangle *Reads a spatial subset of probe-level data from Affymetrix APD files*

Description

Reads a spatial subset of probe-level data from Affymetrix APD files.

Usage

```
## Default S3 method:
readApdRectangle(filename, xrange=c(0, Inf), yrange=c(0, Inf), ..., asMatrix=TRUE)
```

Arguments

filename	The pathname of the APD file.
xrange	A numeric vector of length two giving the left and right coordinates of the cells to be returned.
yrange	A numeric vector of length two giving the top and bottom coordinates of the cells to be returned.
...	Additional arguments passed to readApd().
asMatrix	If TRUE, the APD data fields are returned as matrices with element (1,1) corresponding to cell (xrange[1],yrange[1]).

Value

A named list APD structure similar to what readApd(). In addition, if asMatrix is TRUE, the APD data fields are returned as matrices, otherwise not.

Author(s)

Henrik Bengtsson

See Also

The readApd() method is used internally.

Examples

```
# Local functions
rotate270 <- function(x, ...) {
  x <- t(x)
  nc <- ncol(x)
  if (nc < 2) return(x)
  x[,nc:1,drop=FALSE]
}

# Scan current directory for APD files
```

```
files <- list.files(pattern="[.](apd|APD)$")
files <- files[!file.info(files)$isdir]
if (length(files) > 0) {
  apdFile <- files[1]

  # Read APD intensities in the upper left corner
  apd <- readApdRectangle(apdFile, xrange=c(0,250), yrange=c(0,250))
  z <- rotate270(apd$intensities)
  sub <- paste("Chip type:", apd$header$chipType)
  image(z, col=gray.colors(256), axes=FALSE, main=apdFile, sub=sub)
  text(x=0, y=1, labels="(0,0)", adj=c(0,-0.7), cex=0.8, xpd=TRUE)
  text(x=1, y=0, labels="(250,250)", adj=c(1,1.2), cex=0.8, xpd=TRUE)
}
```

`readApdUnits` *Reads Affymetrix probe data (APD) as units (probesets)*

Description

Reads Affymetrix probe data (APD) as units (probesets) by using the unit and group definitions in the corresponding Affymetrix CDF file.

If more than one APD file is read, all files are assumed to be of the same chip type, and have the same read map, if any. It is not possible to read APD files of different types at the same time.

Usage

```
## Default S3 method:
readApdUnits(filenames, units=NULL, ..., transforms=NULL, cdf=NULL,
 stratifyBy=c("nothing", "pmmm", "pm", "mm"), addDimnames=FALSE, readMap="byMapType",
  dropArrayDim=TRUE, verbose=FALSE)
```

Arguments

`filenames`	The filenames of the APD files. All APD files must be of the same chip type.
`units`	An `integer vector` of unit indices specifying which units to be read. If `NULL`, all units are read.
`...`	Additional arguments passed to `readApd`().
`transforms`	A `list` of exactly `length(filenames)` `function`s. If `NULL`, no transformation is performed. Values read are passed through the corresponding transform function before being returned.
`cdf`	A `character` filename of a CDF file, or a CDF `list` structure. If `NULL`, the CDF file is searched for by `findCdf` first starting from the current directory and then from the directory where the first APD file is.
`stratifyBy`	Argument passed to low-level method `readCdfCellIndices`.
`addDimnames`	If `TRUE`, dimension names are added to arrays, otherwise not. The size of the returned APD structure in bytes increases by 30-40% with dimension names.

readMap	A vector remapping cell indices to file indices. If "byMapType", the read map of type according to APD header will be search for and read. It is much faster to specify the read map explicitly compared with searching for it each time. If NULL, no map is used.
dropArrayDim	If TRUE and only one array is read, the elements of the group field do *not* have an array dimension.
verbose	See Verbose.

Value

A named list where the names corresponds to the names of the units read. Each element of the list is in turn a list structure with groups (aka blocks).

Speed

Since the cell indices are semi-randomized across the array and with units (probesets), it is very unlikely that the read will consist of subsequent cells (which would be faster to read). However, the speed of this method, which uses FileVector to read data, is comparable to the speed of readCelUnits, which uses the Fusion SDK (readCel) to read data.

Author(s)

Henrik Bengtsson

See Also

To read CEL units, readCelUnits. Internally, the readApd() method is used for read probe data, and readApdMap(), if APD file has a map type specified and the read map was not given explicitly.

Examples

```
library("R.utils") # Arguments

verbose <- Arguments$getVerbose(TRUE)

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 1. Scan for existing CEL files
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# a) Scan current directory for CEL files
files <- list.files(pattern="[.](cel|CEL)$")
files <- files[!file.info(files)$isdir]

if (length(files) > 0 && require("affxparser")) {
  # b) Corresponding APD filenames
  celNames <- files
  apdNames <- gsub(pattern, ".apd", files)

  # - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  # 1. Copy the probe intensities from a CEL to an APD file
  # - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  for (kk in 1) {
```

```
  verbose && enter(verbose, "Reading CEL file #", kk)
  cel <- readCel(celNames[kk])
  verbose && exit(verbose)

  if (!file.exists(apdNames[kk])) {
    verbose && enter(verbose, "Creating APD file #", kk)
    chipType <- cel$header$chiptype
    writeApd(apdNames[kk], data=cel$intensities, chipType=chipType)
    verbose && exit(verbose)
  }

  verbose && enter(verbose, "Verifying APD file #", kk)
  apd <- readApd(apdNames[kk])
  verbose && exit(verbose)
  stopifnot(identical(apd$intensities, cel$intensities))
}


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 2. Read a subset of the units
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
units <- c(1, 20:205)
cel <- readCelUnits(celNames[1], units=units)
apd <- readApdUnits(apdNames[1], units=units)
stopifnot(identical(apd, cel))


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 3. The same, but stratified on PMs and MMs
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
apd <- readApdUnits(apdNames[1], units=units, stratifyBy="pmmm",
                                              addDimnames=TRUE)
} # if (length(files) > 0)
```

---

updateApd *Updates an Affymetrix probe data (APD) file*

Description

Updates an Affymetrix probe data (APD) file.

Usage

```
## Default S3 method:
updateApd(filename, indices=NULL, data, writeMap=NULL, ..., verbose=FALSE,
  .checkArgs=TRUE)
```

Arguments

filename	The filename of the APD file.
indices	A numeric vector of cell (probe) indices specifying which cells to updated.
data	A numeric vector of data elements to be assigned.
writeMap	A vector of indicies used to change the order how data elements are *written* (by default).
...	Not used.
verbose	See Verbose.
.checkArgs	If TRUE, arguments are checked, otherwise not.

Value

Returns (invisibly) the pathname of the file updated.

Author(s)

Henrik Bengtsson

See Also

createApd() and updateApd().

Examples

```
## Not run: #See ?createApd for an example.
```

---

updateApdHeader *Updates the header of an Affymetrix probe data (APD) file*

Description

Updates the header of an Affymetrix probe data (APD) file.

Usage

```
## Default S3 method:
updateApdHeader(filename, path=NULL, ..., verbose=FALSE)
```

Arguments

filename	The filename of the APD file.
path	The path to the APD file.
...	A set of named header values to be updated/added to the header. A value of NULL will be removed from the current header.
verbose	See Verbose.

Value

Returns (invisibly) the pathname of the file updated.

Author(s)

Henrik Bengtsson

See Also

createApd() and updateApd().

Examples

```
## Not run: #See ?createApd for an example.
```

---

updateApdUnits *Updates an Affymetrix probe data (APD) file by units (probesets)*

Description

Updates an Affymetrix probe data (APD) file by units (probesets) by using the unit and group definitions in the corresponding Affymetrix CDF file.

Usage

```
## Default S3 method:
updateApdUnits(filename, units=NULL, data, ..., cdf=NULL, stratifyBy=c("nothing", "pmmm",
  "pm", "mm"), verbose=FALSE)
```

Arguments

filename	The filename of the APD file.
units	An integer vector of unit indices specifying which units to be read. If NULL, all units are updated.
data	A numeric vector of data elements to be assigned.
...	Additional arguments passed to updateApd(), e.g. writeMap.
cdf	A character filename of a CDF file, or a CDF list structure. If NULL, the CDF file is searched for by findCdf.
stratifyBy	Argument passed to low-level method readCdfCellIndices.
verbose	See Verbose.

Value

Returns nothing.

Author(s)

Henrik Bengtsson

See Also

readApdUnits() to read unit by units. updateApd() to update cell by cell.

Examples

```
## Not run: #See ?createApd for an example.
```

---

writeApd *Writes an APD probe data file*

Description

Writes an APD probe data file.

Usage

```
## Default S3 method:
writeApd(filename, data, ..., writeMap=NULL)
```

Arguments

filename	The filename of the APD file.
data	A numeric vector of elements to be written.
...	Arguments passed to createApd(), e.g. chipType, mapType etc.
writeMap	A vector of indicies used to change the order how data elements are *written* (by default).

Value

Returns (invisibly) the pathname to the created file.

Author(s)

Henrik Bengtsson

See Also

To create an APD map file, see readApdMap().

`writeApdMap` *Writes an APD probe map file*

Description

Writes an APD probe map file.

Usage

```
## Default S3 method:
writeApdMap(filename, path=NULL, map, ...)
```

Arguments

`filename`	The filename of the APD file.
`path`	The path to the APD file.
`map`	A `vector` of indicies.
`...`	Additional arguments passed to `writeApd`().

Value

Returns (invisibly) the pathname to the create file.

Author(s)

Henrik Bengtsson

See Also

To read an APD map file, see `readApdMap`().

Index