Package ‘apex’

September 23, 2017

Title Phylogenetic Methods for Multiple Gene Data

Version 1.0.3

Description Toolkit for the analysis of multiple gene data. Apex implements the new S4 classes 'multidna', 'multiphyDat' and associated methods to handle aligned DNA sequences from multiple genes.

Depends R (>= 3.1.3), methods, ape, phangorn

Imports utils, graphics, stats, grDevices, adegenet

License GPL (>= 2)

LazyData true

URL https://github.com/thibautjombart/apex

BugReports https://github.com/thibautjombart/apex/issues


VignetteBuilder knitr

Suggests testthat, knitr, rmarkdown

RoxygenNote 6.0.1

NeedsCompilation no

Author Thibaut Jombart [aut, cre], Zhian Namir Kamvar [aut], Klaus Schliep [aut], Eric Archer [aut], Rebecca Harris [aut]

Maintainer Thibaut Jombart <thibautjombart@gmail.com>

Repository CRAN

Date/Publication 2017-09-23 14:19:41 UTC
Description

Accessors for slots in `multidna` and `multiphyDat` objects.

Usage

```r
getNumInd(x, ...)
```

```r
## S4 method for signature 'multidna'
getNumInd(x, ...)
```

```r
getNumLoci(x, ...)
```

```r
## S4 method for signature 'multidna'
getNumLoci(x, ...)
```

```r
## S4 method for signature 'multiphyDat'
getNumLoci(x, ...)
```
getLocusNames(x, ...)

## S4 method for signature 'multidna'
getLocusNames(x, ...)

## S4 method for signature 'multiphyDat'
getLocusNames(x, ...)

setLocusNames(x) <- value

## S4 replacement method for signature 'multidna'
setLocusNames(x) <- value

## S4 replacement method for signature 'multiphyDat'
setLocusNames(x) <- value

getNumSequences(x, ...)

## S4 method for signature 'multidna'
getNumSequences(x, exclude.gap.only = TRUE, loci = NULL, ...)

## S4 method for signature 'multiphyDat'
getNumSequences(x, exclude.gap.only = TRUE, loci = NULL, ...)

getSequenceNames(x, ...)

## S4 method for signature 'multidna'
getSequenceNames(x, exclude.gap.only = TRUE, loci = NULL, ...)

## S4 method for signature 'multiphyDat'
getSequenceNames(x, exclude.gap.only = TRUE, loci = NULL, ...)

genequences(x, ...)

## S4 method for signature 'multidna'
genequences(x, loci = NULL, ids = NULL, simplify = TRUE, exclude.gap.only = TRUE, ...)

## S4 method for signature 'multiphyDat'
genequences(x, loci = NULL, ids = NULL, simplify = TRUE, exclude.gap.only = TRUE, ...)
Arguments

x  
a multidna or multiphyDat object.

...  
further arguments passed on to other functions.

value  
a replacement value for the slot.

exclude.gap.only  
logical. Remove or ignore sequences containing all gaps?

loci  
a character, numeric, or logical vector identifying which loci to return.

ids  
a character, numeric, or logical vector identifying which sequences to return within a locus.

simplify  
logical. If FALSE, always return a list of DNAbin sequences. If TRUE and only one locus has been requested, return a single DNAbin object.

Details

getNumInd  Returns the number of individuals.

getNumLoci  Returns the number of loci.

getLocusNames  Returns the names of each locus.

setLocusNames  Sets the names of each locus.

getNumSequences  Returns the number of sequences in each locus.

getSequenceNames  Returns the names of individual sequences at each locus.

getSequences  Returns sequences of specified loci and individuals.

Description

In multidna and multiphyDat, some individuals may not be sequenced for all genes. The generic function add.gaps has method for both objects; it identifies the missing sequences, and adds gap-only sequences to the alignments wherever needed.

Usage

add.gaps(x, ...)

## S4 method for signature 'multidna'
add.gaps(x, ...)

## S4 method for signature 'multiphyDat'
add.gaps(x, ...)

Arguments

x  
a multidna or multiphyDat object.

...  
further arguments passed to other methods (currently not used).
Description

Toolkit for the analysis of multiple gene data. Apex implements the new S4 classes multidna, multiphyDat and associated methods to handle aligned DNA sequences from multiple genes.

concatenate

Concatenate genes into a single matrix

Description

These functions concatenate separate DNA alignments into a single alignment matrix. concatenate is a generic with methods for:

- multidna: returns a DNAbin matrix
- multiphyDat: returns a phyDat object

Usage

concatenate(x, ...)

## S4 method for signature 'multidna'
concatenate(x, genes = TRUE, ...)

## S4 method for signature 'multiphyDat'
concatenate(x, genes = TRUE, ...)

Arguments

- x: a multidna or a multiphyDat object.
- genes: an optional vector indicating the genes to retain for the concatenation; any way to subset the list in x@dna is acceptable; by default, all genes are used.

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>
## Examples

```r
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
plot(x)
image(concatenate(x))
```

### Description

This function computes pairwise genetic distances between individuals using genes in a `multidna` object. By default, one distance matrix (dist object) is created for each each, but a single distance can be derived by pooling all genes (argument `pool=TRUE`)

### Usage

```r
dist.multidna(x, pool = FALSE, genes = TRUE, ...)
```

### Arguments

- `x`: a `multidna` object.
- `pool`: a logical indicating if all genes should be pooled (concatenated) to obtain a single distance matrix; defaults to FALSE.
- `genes`: an optional vector indicating the genes to retain for the concatenation; any way to subset the list in `x@dna` is acceptable; by default, all genes are used.
- `...`: further arguments passed to `dist.dna`.

### Value

A list of dist objects (pool=FALSE) or a single dist object (pool=TRUE)

### Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

### See Also

- `dist.dna`
getTree

Build phylogenies from multiple gene data

## Examples

```r
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
plot(x)

## get separate distance matrix and pooled one
ld <- dist.multidna(x)
d <- dist.multidna(x, pool=TRUE)

## get corresponding NJ trees
ltrees <- lapply(ld, nj)
tree <- nj(d)
par(mfrow=c(3,1))
for(i in 1:2) plot(ltrees[[i]], main=names(ltrees)[i])
plot(tree, main="Pooled distances")
```

---

### Description

This function builds separate phylogenetic trees for each gene in a multidna object, specifying a method for computing pairwise distances between individuals, and a method to build the tree from the distance matrix. By default, procedures from ape are used.

### Usage

```r
getTree(x, pool = FALSE, genes = TRUE, model = "N",
        pairwise.deletion = TRUE, method = nj, ladderize = TRUE,
        negative.branch.length = FALSE, ...)
```

### Arguments

- `x`: a multidna object.
- `pool`: a logical indicating if all genes should be pooled (concatenated) to obtain a single tree; defaults to FALSE.
- `genes`: an optional vector indicating the genes to retain for the concatenation; any way to subset the list in `x@dna` is acceptable; by default, all genes are used.
- `model`: a character string passed to `dist.dna` describing the model to be used to compute genetic distances; defaults to `"N"`, the absolute number of mutations separating sequences.
pairwise.deletion

  a logical passed to `dist.dna` indicating if pairwise deletions should be used; the alternative is to remove all sites for which at least one missing value is present.

method

  a function building a tree from a matrix of pairwise genetic distances.

ladderize

  a logical indicating if the tree should be ladderized; defaults to TRUE.

negative.branch.length

  a logical indicating if negative branch lengths should be allowed (e.g. in the case of Neighbor-Joining reconstruction), or not, in which case they are set to 0 (FALSE, default).

... further arguments passed to the tree reconstruction method defined by 'method'.

Value

  a `multiPhylo` object

Author(s)

  Thibaut Jombart <t.jombart@imperial.ac.uk>

See Also

  `dist.multidna`

Examples

```r
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
plot(x)

## make trees, default parameters
trees <- getTree(x)
trees
plot(trees, type="unrooted")

## make one single tree based on concatenated genes
tree <- getTree(x, pool=TRUE)
tree
plot(tree, type="unrooted")
```
**Description**

New `multidna` objects can be created using `new("multidna", ...)` where "..." are arguments documented below. The main input is a list of DNABin matrices. The constructor ensures that all matrices will be reordered in the same way, and as an option (setting `add.gaps=TRUE`, gap-only sequences ("...——...")) will be added wherever sequences are missing.

**Usage**

```r
## S4 method for signature 'multidna'
initialize(.Object, dna = NULL, ind.info = NULL,
          gene.info = NULL, add.gaps = TRUE, quiet = FALSE, ...)
```

**Arguments**

- `.Object` the object skeleton, automatically generated when calling `new`.
- `dna` a list of DNABin matrices (1 per gene); rows should be labelled and indicate individuals, but different individuals and different orders can be used in different matrices.
- `ind.info` an optional data.frame containing information on the individuals, where individuals are in rows.
- `gene.info` an optional data.frame containing information on the genes, where genes are in rows.
- `add.gaps` a logical indicating if gap-only sequences should be used where sequences are missing; defaults to TRUE.
- `quiet` a logical indicating if messages should be shown; defaults to FALSE.
- `...` further arguments to be passed to other methods

**Author(s)**

Thibaut Jombart <t.jombart@imperial.ac.uk>

**See Also**

- the `multidna` class
- `read.multidna` and `read.multidna`
Examples

```r
## empty object
new("multidna")

## simple conversion with nicely ordered output
data(woodmouse)
gen <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
image(woodmouse)
image(x@dna[[1]])
image(x@dna[[2]])

## trickier conversion with missing sequences / wrong order
gen <- list(gene1=woodmouse[,1:500], gene2=woodmouse[c(5:1,14:15),501:965])
x <- new("multidna", genes)
x
image(x@dna[[1]])
image(x@dna[[2]])
```

Description

New `multiphyDat` objects can be created using `new("multiphyDat", ...)` where "..." are arguments documented below. The main input is a list of phyDat matrices. The constructor ensures that all matrices will be reordered in the same way, and genes with missing individuals will be filled by sequences of gaps ("-").

Usage

```r
## S4 method for signature 'multiphyDat'
initialize(.Object, seq = NULL, type = character(0),
    ind.info = NULL, gene.info = NULL, add.gaps = TRUE, quiet = FALSE,
    ...)```

Arguments

- `.Object` the object skeleton, automatically generated when calling `new`.
- `seq` a list of phyDat matrices (1 per gene); rows should be labelled and indicate individuals, but different individuals and different orders can be used in different matrices.
- `type` a character string indicating the type of the sequences stored: "DNA" for DNA sequences, "AA" for amino-acids.
multidna-class

ind.info an optional data.frame containing information on the individuals, where individuals are in rows.
gene.info an optional data.frame containing information on the genes, where genes are in rows.
add.gaps a logical indicating if gap-only sequences should be used where sequences are missing; defaults to TRUE.
quiet a logical indicating if messages should be shown; defaults to FALSE.
... further arguments to be passed to other methods

Author(s)

Klaus Schliep <klaus.schliep@gmail.com>
Thibaut Jombart <t.jombart@imperial.ac.uk>

See Also

- the multiphyDat class
- read.multiphyDat

Examples

data(Laurasiatherian)

# empty object
new("multiphyDat")

## simple conversion with nicely ordered output
## Not run:
genes <- list(gene1=subset(Laurasiatherian,, 1:1600, FALSE),
        gene2=subset(Laurasiatherian,, 1601:3179, FALSE))
x <- new("multiphyDat", genes)

## End(Not run)

## trickier conversion with missing sequences / wrong order
genes <- list(gene1=subset(Laurasiatherian, 1:40),
        gene2=subset(Laurasiatherian, 8:47))
x <- new("multiphyDat", genes)

x
multidna-class

Description

This formal (S4) class is used to store multiple DNA alignments. Sequences are stored as a (possibly named) list, with each element of the list being a separate DNA alignment stored as a DNAbin matrix. The rows of the separate matrices all correspond to the same individuals, ordered identically.

Slots

dna a list of DNAbin matrices; empty slot should be NULL
labels a vector of labels of individuals
n.ind the number of individuals
n.seq the total number of sequences (pooling all genes), including gap sequences
n.seq.miss the total number of gap-only sequences
ind.info a data.frame containing information on the individuals, where individuals are in rows; empty slot should be NULL
gene.info a data.frame containing information on the genes, where genes are in rows; empty slot should be NULL

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

Examples

## empty object
new("multidna")

## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
image(woodmouse)
image(x@dna[[1]])
image(x@dna[[2]])

## trickier conversion with missing sequences / wrong order
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[c(5:1,14:15),501:965])
x <- new("multidna", genes)
x
image(x@dna[[1]])
image(x@dna[[2]])
multidnaRalignment

Convert from multidna into alignment (seqinr)

Description
The functions multidnaRalignment and multiphyDatRalignment concatenates separate sequences and return an alignment object of the seqinr package.

Usage
multidnaRalignment(x, genes = TRUE)
multiphyDatRalignment(x, genes = TRUE)

Arguments
  x              a multidna or multiphyDat object.
  genes          an optional vector indicating the genes to retain for the concatenation; any way to subset the list in x@dna or x@seq is acceptable; by default, all genes are used.

Value
  a alignment object

Author(s)
Thibaut Jombart <t.jombart@imperial.ac.uk>, Zhian N. Kamvar, Klaus Schliep

See Also
  • concatenate
  • as.alignment to convert single DNAin objects.

Examples
## not run:
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
y <- multidnaRalignment(x)
y
x2 <- multidnaRmultiphyDat(x)
z <- multiphyDatRalignment(x2)
## End(Not run)
multidna2genind  
Convert multidna into genind

Description
The functions multidna2genind and multiphyDat2genind concatenates separate DNA alignments, and then extracts SNPs of the resulting alignment into a genind object.

Usage
multidna2genind(x, genes = TRUE, mlst = FALSE, gapIsNA = FALSE)
multiphyDat2genind(x, genes = TRUE, mlst = FALSE, gapIsNA = FALSE)

Arguments
x a multidna or multiphyDat object.
genes an optional vector indicating the genes to retain for the concatenation; any way to subset the list in x@dna or x@seq is acceptable; by default, all genes are used.
mlst if TRUE, each gene will result in a single locus in the genind object. (Default to FALSE)
gapIsNA if TRUE and mlst = TRUE, sequences that consist entirely of gaps will be considered as NAs. (Default to FALSE)

Value
a genind object

Author(s)
Thibaut Jombart <t.jombart@imperial.ac.uk>, Zhian N. Kamvar, Klaus Schliep

See Also
• concatenate
• DNAbin2genind to convert single DNAbin objects.

Examples
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
y <- multidna2multiphyDat(x)
multidna2multiphyDat

Description

The functions `multidna2multiphyDat` and `multiphyDat2multidna` are used to convert data between `multidna` and `multiphyDat` classes.

Usage

`multidna2multiphyDat(x)`

`multiphyDat2multidna(x)`

Arguments

`x` a `multidna` or `multiphyDat` object.

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>, Zhian N. Kamvar, Klaus Schliep

See Also

- `concatenate`
- `DNAbin2genind` to convert single DNAbin objects.

Examples

```r
## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x

## conversion multidna -> multiphyDat
y <- multidna2multiphyDat(x)
y

## check round trip
identical(x, multiphyDat2multidna(y))
```
multiphyDat-class

multiphyDat: class for multiple gene data

Description

This formal (S4) class is identical to `multidna`, except that DNA sequences are stored using `phyDat` objects from the `phangorn` package. Sequences are stored as a (possibly named) list, with each element of the list being a separate DNA alignment stored as a `phyDat` object. The rows of the separate matrices all correspond to the same individuals, ordered identically.

Slots

- `seq`: a list of `phyDat` objects; empty slot should be NULL
- `type`: a character string indicating the type of the sequences stored: "DNA" for DNA sequences, "AA" for amino-acids.
- `labels`: a vector of labels of individuals
- `n.ind`: the number of individuals
- `n.seq`: the total number of sequences (pooling all genes), including gap sequences
- `n.seq.miss`: the total number of gap-only sequences
- `ind.info`: a data.frame containing information on the individuals, where individuals are in rows; empty slot should be NULL
- `gene.info`: a data.frame containing information on the genes, where genes are in rows; empty slot should be NULL

Author(s)

Klaus Schliep <klaus.schliep@gmail.com>
Thibaut Jombart <t.jombart@imperial.ac.uk>

Examples

data(Laurasiatherian)

```r
## empty object
new("multiphyDat")
```

```r
## simple conversion with nicely ordered output
## Not run:
genes <- list(gene1=subset(Laurasiatherian,,1:1600, FALSE),
              gene2=subset(Laurasiatherian,,1601:3179, FALSE))
x <- new("multiphyDat", genes)
x
```

```r
## End(Not run)
```

```r
## trickier conversion with missing sequences / wrong order
```
genes <- list(gene1=subset(Laurasiatherian,1:40),
gene2=subset(Laurasiatherian,8:47))
x <- new("multiphyDat", genes)
x

plot, multidna, ANY-method

Display multidna objects

Description

Default printing for multidna objects

Usage

## S4 method for signature 'multidna,ANY'
plot(x, y, rows = TRUE, ask = FALSE, ...)

Arguments

x a multidna object
y an integer vector indicating the genes to plot
rows a logical indicating if different genes should be displayed in separate rows
ask a logical indicating if the user should be prompted between graphs
... arguments passed to image.DNAbin

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>
Thibaut Jombart <t.jombart@imperial.ac.uk>

Examples

## simple conversion with nicely ordered output
data(woodmouse)
genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
plot(x)
Description

These functions read multiple DNA alignments and store the output in a multidna object. They are relying on ape’s original functions read.dna and read.FASTA.

Usage

read.multidna(files, add.gaps = TRUE, ...)
read.multiFASTA(files, add.gaps = TRUE)
read.multiplyDat(files, add.gaps = TRUE, ...)

Arguments

files a vector of characters indicating the paths to the files to read from.
add.gaps a logical indicating if gap-only sequences should be added wherever sequences are missing; defaults to TRUE.
... further arguments to be passed to the functions read.dna and read.FASTA.

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>
Klaus Schliep <klaus.schliep@gmail.com>

See Also

read.dna
read.FASTA
read.phyDat

Examples

## get path to the files
files <- dir(system.file(package="apex"), pattern="patr", full=TRUE)
files

## read files
x <- read.multiFASTA(files)
x
plot(x)

y <- read.multiplyDat(files, format="fasta")
y
**Description**

In `multidna` and `multiphyDat`, some individuals may not be sequenced for all genes, resulting in gap-only sequences for missing data. The generic function `rm.gaps` has method for both objects; it identifies the missing sequences, and removes gap-only sequences from the alignments wherever needed.

**Usage**

```r
rmgaps(x, ..., 
## S4 method for signature 'multidna'
rm.gaps(x, ...) 
## S4 method for signature 'multiphyDat'
rm.gaps(x, ...)
```

**Arguments**

- `x` a `multidna` or `multiphyDat` object.
- `...` further arguments passed to other methods (currently not used).

**Description**

Default printing for multidna objects

**Usage**

```r
show(object)
```

**Arguments**

- `object` a multidna object
show,multiphyDat-method

Display multiphyDat objects

Description
Default printing for multiphyDat objects

Usage
## S4 method for signature 'multiphyDat'
show(object)

Arguments

object a multiphyDat object

Author(s)
Thibaut Jombart <t.jombart@imperial.ac.uk>

[,multidna,ANY,ANY,ANY-method

Subset multidna objects

Description
Individuals in a multidna or multiphyDat object can be subsetted like the rows of a matrix, with the form x[i,]. Genes can be subsetted like the columns of a matrix, i.e. with the form x[,j].

Usage
## S4 method for signature 'multidna,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'multiphyDat,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]

Arguments

x the multidna object to subset.
i a vector of logical, integers or characters to subset data by individuals; characters will be matched against individual labels.
j a vector of logical, integers or characters to subset data by genes; characters will be matched against gene names labels.
... further arguments to be passed to other methods; currently ignored.
drop present for compatibility with the generic; currently not used.
Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

Examples

data(woodmouse)
 genes <- list(gene1=woodmouse[,1:500], gene2=woodmouse[,501:965])
x <- new("multidna", genes)
x
plot(x)

## keep only the first 5 individuals
x[1:5,]
plot(x[1:5,])

## keep individuals 2,4,6 and the second gene
x[c(2,4,6),2]
plot(x[c(2,4,6),2])
Index

[,multidna, ANY, ANY, ANY-method, 20
[,multidna-method
  ([,multidna, ANY, ANY, ANY-method), 20
[,multiplyDat, ANY, ANY, ANY-method
  ([,multidna, ANY, ANY, ANY-method), 20
accessors, 2
add.gaps, 4
add.gaps,multidna-method (add.gaps), 4
add.gaps,multiplyDat-method (add.gaps), 4
apex, 5
apex-package (apex), 5
as.alignment, 13
concatenate, 5
concatenate,multidna-method
  (concatenate), 5
concatenate,multiplyDat-method
  (concatenate), 5
dist.dna, 6–8
dist.multidna, 6, 8
DNAbin2genind, 14, 15
genind, 14
getLocusNames (accessors), 2
getLocusNames,multidna (accessors), 2
getLocusNames,multidna-method
  (accessors), 2
getLocusNames,multiplyDat (accessors), 2
getLocusNames,multiplyDat-method
  (accessors), 2
getNumInd (accessors), 2
getNumInd,multidna-method (accessors), 2
getNumInd,multiplyDat (accessors), 2
ggetNumInd,multiplyDat-method
  (accessors), 2
ggetNumLoci (accessors), 2
ggetNumLoci,multidna (accessors), 2
ggetNumLoci,multidna-method (accessors), 2
ggetNumLoci,multiplyDat (accessors), 2
ggetNumLoci,multiplyDat-method
  (accessors), 2
ggetNumSequences (accessors), 2
ggetNumSequences,multidna (accessors), 2
ggetNumSequences,multidna-method
  (accessors), 2
ggetNumSequences,multiplyDat
  (accessors), 2
ggetNumSequences,multiplyDat-method
  (accessors), 2
ggetSequenceNames (accessors), 2
ggetSequenceNames,multidna (accessors), 2
ggetSequenceNames,multidna-method
  (accessors), 2
ggetSequenceNames,multiplyDat
  (accessors), 2
ggetSequenceNames,multiplyDat-method
  (accessors), 2
ggetSequences (accessors), 2
ggetSequences,multidna (accessors), 2
ggetSequences,multidna-method
  (accessors), 2
ggetSequences,multiplyDat (accessors), 2
ggetSequences,multiplyDat-method
  (accessors), 2
ggetTree, 7
image, DNAbin, 17
initialize,multidna-method, 9
initialize,multidna-methods
  (initialize,multidna-method), 9
initialize,multiplyDat-method, 10
initialize,multiplyDat-methods
  (initialize,multiplyDat-method), 10
multidna, 2, 4–7, 9, 13–16, 18–20
multidna (multidna-class), 11
multidna-class, 11
multidna2alignment, 13
multidna2genind, 14
multidna2multiplyDat, 15
multiplyDat, 2, 4, 5, 10, 11, 13–15, 19, 20
multiplyDat (multiplyDat-class), 16
multiplyDat-class, 16
multiplyDat2alignment
  (multidna2alignment), 13
multiplyDat2genind (multidna2genind), 14
multiplyDat2mulidna
  (multidna2multiplyDat), 15

new.multidna
  (initialize, multidna-method), 9
new.multiplyDat
  (initialize, multiplyDat-method), 10

plot, multidna, ANY-method, 17
plot, multidna-method
  (plot, multidna, ANY-method), 17

read.dna, 18
read.FASTA, 18
read.multidna, 9, 18
read.multiFASTA (read.multidna), 18
read.multiplyDat, 11
read.multiplyDat (read.multidna), 18
read.phyDat, 18
rm.gaps, 19
rm.gaps, multidna-method (rm.gaps), 19
rm.gaps, multiplyDat-method (rm.gaps), 19

setLocusNames<- (accessors), 2
setLocusNames<-, multidna (accessors), 2
setLocusNames<-, multidna-method (accessors), 2
setLocusNames<-, multiplyDat (accessors), 2
setLocusNames<-, multiplyDat-method (accessors), 2
show, multidna-method, 19
show, multiplyDat-method, 20