

# Package ‘pegas’

October 7, 2009

**Version** 0.2

**Date** 2009-10-03

**Title** Population and Evolutionary Genetics Analysis System

**Author** Emmanuel Paradis

**Maintainer** Emmanuel Paradis <Emmanuel.Paradis@ird.fr>

**Depends** R (>= 2.6.0), ape (>= 2.4), adegenet

**ZipData** no

**Description** pegas provides functions for reading, writing, plotting, analysing, and manipulating allelic data, and for the analysis of population nucleotide sequences including coalescence analyses.

**License** GPL (>= 2)

**URL** <http://ape.mpl.ird.fr/pegas/pegas.html>

**Repository** CRAN

**Date/Publication** 2009-10-07 06:38:13

## R topics documented:

pegas-package . . . . .	2
amova . . . . .	2
as.loci . . . . .	4
Fst . . . . .	5
haploNet . . . . .	6
haplotype . . . . .	7
heterozygosity . . . . .	8
hw.test . . . . .	9
MMD . . . . .	10
nuc.div . . . . .	11
R2.test . . . . .	12
read.gtx . . . . .	13

read.loci . . . . .	14
site.spectrum . . . . .	15
summary.loci . . . . .	16
tajima.test . . . . .	17
theta.h . . . . .	18
theta.k . . . . .	19
theta.s . . . . .	20
theta.tree . . . . .	22
utilities . . . . .	23
write.loci . . . . .	24

<b>Index</b>	<b>26</b>
--------------	-----------

---

pegas-package	<i>Population and Evolutionary Genetics Analysis System</i>
---------------	---

---

### Description

**pegas** provides functions for the analysis of allelic data and of haplotype data from DNA sequences. It requires and complements two other R-packages: **ape** and **adegenet**.

The complete list of functions can be displayed with `library(help = pegas)`.

More information on **pegas** can be found at <http://ape.mpl.ird.fr/pegas/pegas.html>.

### Note

Currently, **pegas** is in development and most functions need to be tested more or less thoroughly.

### Author(s)

Emmanuel Paradis

Maintainer: Emmanuel Paradis

---

amova	<i>Analysis of Molecular Variance</i>
-------	---------------------------------------

---

### Description

This function performs a hierarchical analysis of molecular variance as described in Excoffier et al. (1992). This implementation accepts any number of hierarchical levels.

### Usage

```
amova(formula, data = NULL, nperm = 1000, is.squared = FALSE)
## S3 method for class 'amova':
print(x, ...)
```

**Arguments**

<code>formula</code>	a formula giving the AMOVA model to be fitted with the distance matrix on the left-hand side of the <code>~</code> , and the population, region, etc, levels on its right-hand side (see details).
<code>data</code>	an optional data frame where to find the hierarchical levels; by default they are searched for in the user's workspace.
<code>nperm</code>	the number of permutations for the tests of hypotheses (1000 by default). Set this argument to 0 to skip the tests and simply estimate the variance components.
<code>is.squared</code>	a logical specifying whether the distance matrix has already been squared.
<code>x</code>	an object of class "amova".
<code>...</code>	unused (here for compatibility).

**Details**

The formula must be of the form `d ~ A/B/...` where `d` is a distance object, and `A`, `B`, etc, are the hierarchical levels from the highest one. Any number of levels is accepted, so specifying `d ~ A` will simply test for population differentiation.

It is assumed that the rows of the distance matrix are in the same order than the hierarchical levels (which may be checked by the user).

**Value**

an object of class "amova" which is a list with a table of sums of square deviations (SSD), mean square deviations (MSD), and the number of degrees of freedom, and a vector of variance components.

**Note**

If there are more than three levels, approximate formulae are used to estimate the variance components.

**Author(s)**

Emmanuel Paradis

**References**

Excoffier, L., Smouse, P. E. and Quattro, J. M. (1992) Analysis of molecular variance inferred from metric distances among DNA haplotypes: application to human mitochondrial DNA restriction data. *Genetics*, **131**, 479–491.

**See Also**

[amova](#) in the **ade4** for an implementation of the original Excoffier et al.'s model; [adonis](#) for a general (multivariate) implementation of an ANOVA framework with distances.

## Examples

```
require(ape)
data(woodmouse)
d <- dist.dna(woodmouse)
g <- factor(c(rep("A", 7), rep("B", 8)))
p <- factor(c(rep(1, 3), rep(2, 4), rep(3, 4), rep(4, 4)))
amova(d ~ g/p) # 2 levels
amova(d ~ p) # 1 level
amova(d ~ g)

## 3 levels:
pop <- gl(64, 5, labels = paste("pop", 1:64))
region <- gl(16, 20, labels = paste("region", 1:16))
conti <- gl(4, 80, labels = paste("conti", 1:4))
dd <- as.dist(matrix(runif(320^2), 320))
amova(dd ~ conti/region/pop, nperm = 100)
```

---

as.loci

*Conversion Among Allelic Data Classes*


---

## Description

These functions do conversion among different allelic data classes.

## Usage

```
as.loci(x, ...)
## S3 method for class 'genind':
as.loci(x, ...)
genind2loci(x)
## S3 method for class 'data.frame':
as.loci(x, allele.sep = "/", col.pop = "none", col.loci = NULL, ...)
loci2genind(x)
## S3 method for class 'factor':
as.loci(x, allele.sep = "/", ...)
## S3 method for class 'vector':
as.loci(x, allele.sep = "/", ...)
```

## Arguments

x	an object of class "loci" or "genind", a data frame, a factor, or a vector.
allele.sep	the character(s) separating the alleles for each locus in the data file (a slash by default).
col.pop	specifies whether one of the column of the data file identifies the population; default "none", otherwise an integer giving the number of the column.
col.loci	a vector of integers specifying the indices of the columns that are loci. By default, all columns are taken as loci except one labelled "population", if present.
...	further arguments to be passed to or from other methods.

**Details**

`genind2loci(x)` and `as.loci(x)` are the same if `x` is of class "genind".

**Value**

a data frame with class `c("loci", "data.frame")` for `as.loci` and `genind2loci`; an object of class "genind" for `loci2genind`.

**Author(s)**

Emmanuel Paradis

**See Also**

[read.loci](#)

---

Fst

*F-Statistics*

---

**Description**

This function computes the  $F_{IT}$ ,  $F_{ST}$  and  $F_{IS}$  for each locus in the data.

**Usage**

```
Fst(x, pop = NULL)
```

**Arguments**

`x` an object of class "loci".

`pop` a vector or factor giving the population assignment of each row of `x`, or a single numeric value specifying which column of `x` to use as population indicator. By default, the column labelled "population" is used.

**Details**

The formulae in Weir and Cockerham are used for each allele, and then averaged within each locus over the different alleles as suggested by these authors.

**Value**

a matrix with genes (loci) as rows and the three  $F$ -statistics as columns.

**Note**

This function has not been tested thoroughly, so the results need to be taken with some caution.

**Author(s)**

Emmanuel Paradis

**References**

Weir, B. S. and Cockerham, C. C. (1984) Estimating  $F$ -statistics for the analysis of population structure. *Evolution*, **38**, 1358–1370.

---

 haploNet

*Haplotype Networks*


---

**Description**

haploNet computes an haplotype network. There is a plot method.

**Usage**

```
haploNet(h)
## S3 method for class 'haploNet':
plot(x, size = 1, col = "black", bg = "white",
      col.link = "black", lwd = 1, lty = 1, pie = NULL,
      labels = TRUE, scale.ratio = 1, legend = FALSE,
      fast = FALSE, ...)
```

**Arguments**

<code>h</code>	an object of class "haplotype".
<code>x</code>	an object of class "haploNet".
<code>size</code>	a numeric vector giving the diameter of the circles representing the haplotypes: this is in the same unit than the links and eventually recycled.
<code>col</code>	a character vector specifying the colours of the circles; eventually recycled.
<code>bg</code>	a character vector specifying the colours of the background of the circles; eventually recycled.
<code>col.link</code>	a character vector specifying the colours of the links; eventually recycled.
<code>lwd</code>	a numeric vector giving the width of the links; eventually recycled.
<code>lty</code>	idem for the line types.
<code>pie</code>	a matrix used to draw pie charts for each haplotype; it must have as many rows than there is haplotypes.
<code>labels</code>	a logical specifying whether to identify the haplotypes with Roman numerals (the default).
<code>scale.ratio</code>	the ratio of the scale of the links representing the number of steps on the scale of the circles representing the haplotypes. It may be needed to give a value greater than one to avoid overlapping circles.

legend	a logical specifying whether to draw the legend, or a vector of length two giving the coordinates where to draw the legend; FALSE by default. If TRUE, the user is asked to click where to draw the legend.
fast	a logical specifying whether to optimize the spacing of the circles; FALSE by default.
...	further arguments passed to <code>plot</code> .

**Value**

`haploNet` returns an object of class "haploNet" which is a matrix where each row represents a link in the network, the first and second columns give the number of linked haplotypes, the third column, named "step", gives the number of steps in this link, and the fourth column, named "Prob", gives the probability of a parsimonious link as given by Templeton et al. (1992).

**Author(s)**

Emmanuel Paradis

**References**

Templeton, A. R., Crandall, K. A. and Sing, C. F. (1992) A cladistic analysis of phenotypic association with haplotypes inferred from restriction endonuclease mapping and DNA sequence data. III. Cladogram estimation. *Genetics*, **132**, 619–635.

**See Also**

[haplotype](#)

---

haplotype

*Haplotype Extraction and Frequencies*

---

**Description**

`haplotype` extracts the haplotypes from a set of DNA sequences. The result can be plotted with the appropriate function.

**Usage**

```
haplotype(x, labels = NULL)
## S3 method for class 'haplotype':
plot(x, ...)
```

**Arguments**

x	a set of DNA sequences (as an object of class "DNABin", or an object of class "haplotype").
labels	a vector of character strings used as names for the rows of the returned object. By default, Roman numbers are given.
...	further arguments passed to <code>barplot</code> .

**Value**

haplotype returns an object of class c("haplotype", "DNABin") which is an object of class "DNABin" with an attribute "index" identifying the index of each observation that share the same haplotype.

**Author(s)**

Emmanuel Paradis

**See Also**

[DNABin](#) for manipulation of DNA sequences in R.

---

heterozygosity

*Heterozygosity at a Locus Using Gene Frequencies*

---

**Description**

This function computes the mean heterozygosity from gene frequencies, and returns optionally the associated variance.

**Usage**

```
heterozygosity(x, variance = FALSE)
H(x, variance = FALSE)
```

**Arguments**

x	a vector or a factor.
variance	a logical indicating whether the variance of the estimated heterozygosity should be returned (TRUE), the default being FALSE.

**Details**

The argument x can be either a factor or a vector. If it is a factor, then it is taken to give the individual alleles in the population. If it is a numeric vector, then its values are taken to be the numbers of each allele in the population. If it is a non-numeric vector, it is a coerced as a factor.

The mean heterozygosity is estimated with:

$$\hat{H} = \frac{n}{n-1} \left( 1 - \sum_{i=1}^k p_i^2 \right)$$

where  $n$  is the number of genes in the sample,  $k$  is the number of alleles, and  $p_i$  is the observed (relative) frequency of the allele  $i$ .

**Value**

a numeric vector of length one with the estimated mean heterozygosity (the default), or of length two if the variance is returned `variance = TRUE`.

**Author(s)**

Emmanuel Paradis (Emmanuel.Paradis@mpl.ird.fr)

**References**

Nei, M. (1987) *Molecular evolutionary genetics*. New York: Columbia University Press.

**See Also**

[theta.s](#)

---

hw.test

*Test of Hardy–Weinberg Equilibrium*

---

**Description**

This function tests, for a series of loci, the hypothesis that genotype frequencies follow the Hardy–Weinberg equilibrium.

**Usage**

```
hw.test(x, B = 1000)
```

**Arguments**

`x` an object of class "loci".  
`B` the number of replicates for the Monte Carlo procedure.

**Details**

This test can be performed with any level of ploidy. Two versions of the test are available: the classical  $\chi^2$ -test based on the expected genotype frequencies calculated from the allelic frequencies, and an exact test based on Monte Carlo permutations of alleles. The latter version is available only for diploids. Set `B = 0` if you want to skip the latter test.

**Value**

a matrix with three or four columns with the  $\chi^2$ -value, the number of degrees of freedom, associated *P*-value, and possibly the *P*-value from the Monte Carlo test. The rows of this matrix are the different loci in `x`.

**Author(s)**

Emmanuel Paradis

**Description**

This function draws a histogram of the frequencies of pairwise distances from a set of DNA sequences.

**Usage**

```
MMD(x, xlab = "Distance", main = "", rug = TRUE,  
     legend = TRUE, lcol = "blue", ...)
```

**Arguments**

<code>x</code>	a set of DNA sequences (object of class "DNAbin").
<code>xlab</code>	the label for the x-axis.
<code>main</code>	the title (none by default).
<code>rug</code>	a logical specifying whether to add a rug of the distances on the horizontal axis (see <a href="#">rug</a> ).
<code>legend</code>	a logical specifying whether to draw a legend.
<code>lcol</code>	the colour used for the empirical density curve.
<code>...</code>	further arguments passed to <code>hist</code> .

**Details**

Currently only the observed distribution of pairwise distances are plotted, as well as an empirical density estimate. Rogers and Harpending (1992) give formulae to compute the expected curves under some demographic models.

**Author(s)**

Emmanuel Paradis

**References**

Rogers, A. R. and Harpending, H. (1992) Population growth makes waves in the distribution of pairwise genetic-differences. *Molecular Biology and Evolution*, **9**, 552–569.

**Examples**

```
data(woodmouse)  
MMD(woodmouse, col = "grey")  
MMD(woodmouse, breaks = 20, legend = FALSE)
```

---

`nuc.div`*Nucleotide Diversity*

---

**Description**

This function computes the nucleotide diversity from a sample of DNA sequences.

**Usage**

```
nuc.div(x, variance = FALSE, pairwise.deletion = FALSE)
```

**Arguments**

`x` a matrix or a list which contains the DNA sequences.

`variance` a logical indicating whether to compute the variance of the estimated nucleotide diversity.

`pairwise.deletion` a logical indicating whether to delete the sites with missing data in a pairwise way. The default is to delete the sites with at least one missing data for all sequences.

**Details**

The nucleotide diversity is the sum of the number of differences between pairs of sequences divided by the number of comparisons (i.e.  $n(n - 1)/2$ , where  $n$  is the number of sequences).

The variance of the estimated diversity uses formula (10.9) from Nei (1987). This applies only if all sequences are of the same lengths, and cannot be used if `pairwise.deletion = TRUE`. A bootstrap estimate may be in order if you insist on using the latter option.

**Value**

A numeric vector with one or two values (if `variance = TRUE`).

**Author(s)**

Emmanuel Paradis (Emmanuel.Paradis@mpl.ird.fr)

**References**

Nei, M. (1987) *Molecular evolutionary genetics*. New York: Columbia University Press.

**See Also**

[base.freq](#), [GC.content](#), [theta.s](#), [seg.sites](#)

**Examples**

```
data(woodmouse)
nuc.div(woodmouse)
nuc.div(woodmouse, TRUE)
nuc.div(woodmouse, FALSE, TRUE)
```

---

R2.test

*Ramos-Onsins–Rozas Test of Neutrality*


---

**Description**

This function computes Ramos-Onsins and Rozas's test of neutrality for a set of DNA sequences.

**Usage**

```
R2.test(x, B = 1000, theta = 1, plot = TRUE, quiet = FALSE, ...)
```

**Arguments**

x	a DNA matrix as an object of class "DNAbin".
B	the number of replicates used for the simulation procedure.
theta	the value of the <i>theta</i> population parameter used in simulation procedure.
plot	a logical value specifying whether to plot the results (TRUE by default).
quiet	a logical value specifying whether to not display the progress of the simulations. The default is FALSE meaning that a progress bar is displayed by default.
...	further arguments passed to <code>hist</code> .

**Value**

a list with two elements: `R2` the value of the test statistic  $R_2$ , and `P.val` the associated *P*-value. If `B = 0` a single value, the test statistic, is returned

**Note**

The simulation procedure probably needs to be tested and improved. However the results make sense so far.

**Author(s)**

Emmanuel Paradis

**References**

Ramos-Onsins, R. and Rozas, R. (2002) Statistical properties of new neutrality tests against population growth. *Molecular Biology and Evolution*, **19**, 2092–2100.

Sano, J. and Tachida, G. (2005) Gene genealogy and properties of test statistics of neutrality under population growth. *Genetics*, **169**, 1687–1697.

**See Also**

[read.dna](#), [dist.dna](#)

---

read.gtx

*Read Genetix Data Files*

---

**Description**

This function reads allelic data from a Genetix file (\*.gtx).

**Usage**

```
read.gtx(file)
```

**Arguments**

`file` a file name specified by either a variable of mode character or a quoted string.

**Value**

a data frame with class `c("loci", "data.frame")`.

**Note**

The package **adegenet** has a similar function, but it returns an object of class "genind".

**Author(s)**

Emmanuel Paradis

**References**

Belkhir, K., Borsa, P., Chikhi, L., Raufaste, N. and Bonhomme, F. (1996–2004) GENETIX 4.05, logiciel sous Windows(TM) pour la genetique des populations. Laboratoire Genome, Populations, Interactions, CNRS UMR 5000, Universite de Montpellier II, Montpellier (France). <http://www.genetix.univ-montp2.fr/genetix/intro.htm>

**See Also**

[read.loci](#), [write.loci](#), [read.genetix](#)

**Examples**

```
require(adegenet)
(X <- read.gtx(system.file("files/nancycats.gtx", package = "adegenet"))
## compare with the example in ?read.genetix
```

read.loci

*Read Allelic Data Files***Description**

This function reads allelic data from a text file: rows are individuals, and each column is an allele. By default, the header of the file gives the locus names. If one column is labelled “population”, it is taken as a population variable.

**Usage**

```
read.loci(file, header = TRUE, loci.sep = " ", allele.sep = "/",
          col.pop = "none", col.loci = NULL, skip = 0)
```

**Arguments**

file	a file name specified by either a variable of mode character, or a quoted string.
header	a logical specifying whether the first line of the data file gives the names of the loci (TRUE by default).
loci.sep	the character(s) separating the loci (columns) in the data file (a space by default).
allele.sep	the character(s) separating the alleles for each locus in the data file (a slash by default).
col.pop	specifies whether one of the column of the data file identifies the population; default "none", otherwise an integer giving the number of the column.
col.loci	a vector of integers specifying the indices of the columns that are loci. By default, all columns are taken as loci except one labelled "population", if present.
skip	an integer specifying the number of lines in the data file to be skipped before reading the data.

**Details**

The rownames of the returned object identify the individual genotypes; they are either taken from the data file if present, or given the values "1", "2", ...

In the returned object, alleles are separated by "/" , even if it is not the case in the data file.

**Value**

a data frame with class `c("loci", "data.frame")`. Details on the structure can be found in <http://ape.mpl.ird.fr/pegas/DefinitionDataClassesPegas.pdf>

**Author(s)**

Emmanuel Paradis

**See Also**

[read.gtx](#), [write.loci](#)

---

site.spectrum	<i>Site Frequency Spectrum</i>
---------------	--------------------------------

---

### Description

site.spectrum computes the (un)folded site frequency spectrum of a set of aligned DNA sequences.

### Usage

```
site.spectrum(x, folded = TRUE, outgroup = 1)
## S3 method for class 'spectrum':
plot(x, col = "red", main = NULL, ...)
```

### Arguments

x	a set of DNA sequences (as an object of class "DNABin", or an object of class "spectrum").
folded	a logical specifying whether to compute the folded site frequency spectrum (the default), or the unfolded spectrum if folded = FALSE.
outgroup	a single integer value giving which sequence is ancestral; ignored if folded = TRUE.
col	the colour of the barplot (red by default).
main	a character string for the title of the plot; a generic title is given by default (use main = "" to have no title).
...	further arguments passed to <code>barplot</code> .

### Details

Under the infinite sites model of mutation, mutations occur on distinct sites, so every segregating (polymorphic) site defines a partition of the  $n$  sequences (see Wakeley, 2009). The *site frequency spectrum* is a series of values where the  $i$ th element is the number of segregating sites defining a partition of  $i$  and  $n - i$  sequences. The *unfolded* version requires to define an ancestral state with an external (outgroup) sequence, so  $i$  varies between 1 and  $n - 1$ . If no ancestral state can be defined, the *folded* version is computed, so  $i$  varies between 1 and  $n/2$  or  $(n - 1)/2$ , for  $n$  even or odd, respectively.

If `folded = TRUE`, sites with more than two states are ignored and a warning is returned giving how many such sites were found.

If `folded = FALSE`, sites with an ambiguous state at the external sequence are ignored and a warning is returned giving how many such sites were found. Note that it is not checked if some sites have more than two states.

### Value

site.spectrum returns an object of class "spectrum" which is a vector of integers (some values may be equal to zero) with the attribute "folded" (a logical value) indicating which version of the spectrum has been computed.

**Author(s)**

Emmanuel Paradis

**References**

Wakeley, J. (2009) Coalescent Theory: An Introduction. Greenwood Village, CO: Roberts and Company Publishers.

**See Also**

[DNABin](#) for manipulation of DNA sequences in R, [haplotype](#)

**Examples**

```
require(ape)
data(woodmouse)
(sp <- site.spectrum(woodmouse))
plot(sp)
```

---

summary.loci

---

*Print and Summaries of Loci Objects*


---

**Description**

These functions print and summarize table of alleles and loci (objects of class "loci").

**Usage**

```
## S3 method for class 'loci':
print(x, details = FALSE, ...)
## S3 method for class 'loci':
summary(object, ...)
## S3 method for class 'summary.loci':
print(x, ...)
## S3 method for class 'loci':
x[i, j]
## S3 method for class 'summary.loci':
plot(x, loci, what = "both", layout = 1, col = c("blue", "red"), ...)
```

**Arguments**

<code>x</code> , <code>object</code>	an object of class "loci" or "genind", a data frame, a factor, or a vector.
<code>details</code>	a logical value: if TRUE the data are printed as a data frame; the default is FALSE.
<code>i</code> , <code>j</code>	indices of the rows and/or columns to select or to drop. They may be numeric, logical, or character (in the same way than for standard R objects).
<code>loci</code>	the loci (genes) to be plotted. By default, all loci are plotted.

what	the frequencies to be plotted. Three choices are possible: "alleles", "genotypes", and "both" (the default), or any unambiguous abbreviations.
layout	the number of graphs to be plotted simultaneously.
col	the colours used for the barplots.
...	further arguments to be passed to or from other methods.

### Details

Genotypes not observed in the data frame are not counted.

When using the `[]` method, if only one column is extracted or if the returned data frame has no 'loci' column, then the class "loci" is dropped.

Note that an object of class "loci" can be displayed in the R spreadsheet editor with, e.g., `fix(x)`.

### Value

`summary.loci` returns a list with the genes as names and each element made a list with two vectors "genotype" and "allele" with the frequencies (numbers) of genotypes and alleles, respectively. The names of these two vectors are the observed genotypes and alleles.

`print` and `plot` methods return NULL.

### Author(s)

Emmanuel Paradis

### See Also

[read.loci](#), [getAlleles](#)

### Examples

```
require(adegenet)
data(nancycats)
x <- as.loci(nancycats)
s <- summary(x)
plot(s, layout=20, las=2)
layout(1)
```

---

tajima.test

*Test of the Neutral Mutation Hypothesis*

---

### Description

This function tests the neutral mutation hypothesis with Tajima's *D*.

**Usage**

```
tajima.test(x)
```

**Arguments**

`x` a set of DNA sequences (object of class "DNABin").

**Value**

a list with three numeric values:

<code>D</code>	Tajima's $D$ statistic.
<code>Pval.normal</code>	the p-value assuming that $D$ follows a normal distribution with mean zero and variance one.
<code>Pval.beta</code>	the p-value assuming that $D$ follows a beta distribution after rescaling on $[0, 1]$ (Tajima, 1989).

**Author(s)**

Emmanuel Paradis

**References**

Tajima, F. (1989) Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics*, **123**, 595–595.

**Examples**

```
require(ape)
data(woodmouse)
tajima.test(woodmouse)
```

---

theta.h

*Population Parameter THETA using Homozygosity*

---

**Description**

This function computes the population parameter THETA using the homozygosity (or mean heterozygosity) from gene frequencies.

**Usage**

```
theta.h(x, standard.error = FALSE)
```

## Arguments

`x` a vector or a factor.  
`standard.error` a logical indicating whether the standard error of the estimated theta should be returned (TRUE), the default being FALSE.

## Details

The argument `x` can be either a factor or a vector. If it is a factor, then it is taken to give the individual alleles in the population. If it is a numeric vector, then its values are taken to be the numbers of each allele in the population. If it is a non-numeric vector, it is coerced as a factor.

The standard error is computed with an approximation due to Chakraborty and Weiss (1991).

## Value

a numeric vector of length one with the estimated theta (the default), or of length two if the standard error is returned (`standard.error = TRUE`).

## Author(s)

Emmanuel Paradis (Emmanuel.Paradis@mpl.ird.fr)

## References

Zouros, E. (1979) Mutation rates, population sizes and amounts of electrophoretic variation at enzyme loci in natural populations. *Genetics*, **92**, 623–646.

Chakraborty, R. and Weiss, K. M. (1991) Genetic variation of the mitochondrial DNA genome in American Indians is at mutation-drift equilibrium. *American Journal of Human Genetics*, **86**, 497–506.

## See Also

[heterozygosity](#), [theta.s](#), [theta.k](#), [theta.tree](#)

---

theta.k

*Population Parameter THETA using Expected Number of Alleles*

---

## Description

This function computes the population parameter THETA using the expected number of alleles.

## Usage

```
theta.k(x, n = NULL, k = NULL)
```

**Arguments**

x	a vector or a factor.
n	a numeric giving the sample size.
k	a numeric giving the number of alleles.

**Details**

This function can be used in two ways: either with a vector giving the individual genotypes from which the sample size and number of alleles are derived (`theta.k(x)`), or giving directly these two quantities (`theta.k(n, k)`).

The argument `x` can be either a factor or a vector. If it is a factor, then it is taken to give the individual alleles in the population. If it is a numeric vector, then its values are taken to be the numbers of each allele in the population. If it is a non-numeric vector, it is coerced as a factor.

Both arguments `n` and `k` must be single numeric values.

**Value**

a numeric vector of length one with the estimated theta.

**Note**

For the moment, no standard-error or confidence interval is computed.

**Author(s)**

Emmanuel Paradis (Emmanuel.Paradis@mpl.ird.fr)

**References**

Ewens, W. J. (1972) The sampling theory of selectively neutral alleles. *Theoretical Population Biology*, **3**, 87–112.

**See Also**

[theta.h](#), [theta.s](#), [theta.tree](#)

---

theta.s

*Population Parameter THETA using Segregating Sites*

---

**Description**

This function computes the population parameter THETA using the number of segregating sites `s` in a sample of `n` DNA sequences.

**Usage**

```
theta.s(s, n, variance = FALSE)
```

**Arguments**

s	a numeric giving the number of segregating sites.
n	a numeric giving the number of sequences.
variance	a logical indicating whether the variance of the estimated THETA should be returned (TRUE), the default being FALSE.

**Value**

a numeric vector of length one with the estimated theta (the default), or of length two if the standard error is returned (`variance = TRUE`).

**Note**

The number of segregating sites needs to be computed beforehand, for instance with the function `seg.sites` (see example below).

**Author(s)**

Emmanuel Paradis (Emmanuel.Paradis@mpl.ird.fr)

**References**

Watterson, G. (1975) On the number of segregating sites in genetical models without recombination. *Theoretical Population Biology*, **7**, 256–276.

Tajima, F. (1989) Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics*, **123**, 585–595.

**See Also**

[theta.h](#), [theta.k](#), [seg.sites](#), [nuc.div](#), [theta.tree](#)

**Examples**

```
data(woodmouse)
s <- length(seg.sites(woodmouse))
n <- nrow(woodmouse)
theta.s(s, n)
theta.s(s, n, variance = TRUE)
```

---

 theta.tree

 Population Parameter THETA Using Genealogy
 

---

### Description

This function estimates the population parameter  $\theta$  from a genealogy (coded as a phylogenetic tree) under the coalescent.

### Usage

```
theta.tree(phy, theta, fixed = FALSE, log = TRUE)
```

### Arguments

phy	an object of class "phylo".
theta	a numeric vector.
fixed	a logical specifying whether to estimate theta (the default), or to return the likelihoods for all values in theta.
log	a logical specifying whether to return the likelihoods on a log scale (the default); ignored if fixed = FALSE.

### Details

The tree `phy` is considered as a genealogy, and therefore should be ultrametric. By default,  $\theta$  is estimated by maximum likelihood and the value given in `theta` is used as starting value for the minimisation function (if several values are given as a vector the first one is used). If `fixed = TRUE`, then the [log-]likelihood values are returned corresponding to each value in `theta`.

### Value

If `fixed = FALSE`, a list with two elements:

theta	the maximum likelihood estimate of $\theta$ ;
logLik	the log-likelihood at its maximum.

If `fixed = TRUE`, a numeric vector with the [log-]likelihood values.

### Author(s)

Emmanuel Paradis

### References

Kingman, J. F. C. (1982) The coalescent. *Stochastic Processes and their Applications*, **13**, 235–248.  
 Wakeley, J. (2009) Coalescent Theory: An Introduction. Greenwood Village, CO: Roberts and Company Publishers.

**See Also**

[theta.h](#), [theta.s](#), [theta.k](#)

**Examples**

```
tr <- rcoal(50) # assumes theta = 1
theta.tree(tr, 10)
## profile log-likelihood:
THETA <- seq(0.5, 1.5, 0.01)
logLikelihood <- theta.tree(tr, THETA, fixed = TRUE)
plot(THETA, logLikelihood, type = "l")
```

---

utilities

*Utily Functions for pegas*

---

**Description**

The first three functions extract information on loci. `expand.genotype` creates a table of all possible genotypes given a set of alleles.

**Usage**

```
getPloidy(x)
getAlleles(x)
getGenotypes(x)
expand.genotype(n, alleles = NULL, ploidy = 2, matrix = FALSE)
```

**Arguments**

<code>x</code>	an object of class "loci".
<code>n</code>	an integer giving how many alleles to consider (ignored if <code>alleles</code> is used).
<code>alleles</code>	the allele names as a vector of mode character.
<code>ploidy</code>	an integer giving the ploidy level (either 2 or 4 for the moment).
<code>matrix</code>	a logical specifying whether to return the genotypes in a matrix or as a character vector.

**Value**

`getPloidy` returns the ploidy level of all loci in an object of class "loci" as a numeric vector. `getAlleles` and `getGenotypes` return the alleles and genotypes, respectively, observed in all loci in an object of class "loci" as a list.

`expand.genotype` returns a character vector (the default) or a matrix where the rows are the genotypes and the columns are the alleles. The matrix is numeric by default, or character if the argument `alleles` is given.

**Author(s)**

Emmanuel Paradis

**Examples**

```
expand.genotype(2)
expand.genotype(2, LETTERS[1:3])
expand.genotype(3, ploidy = 4)
```

---

`write.loci`*Write Allelic Data Files*

---

**Description**

This function writes allelic data into a text file.

**Usage**

```
write.loci(x, file = "", loci.sep = " ", allele.sep = "/", ...)
```

**Arguments**

<code>x</code>	an object of class "loci".
<code>file</code>	a file name specified by either a variable of mode character, or a quoted string. By default, the data are printed on the console.
<code>loci.sep</code>	the character(s) use to separate the loci (columns) in the file (a space by default).
<code>allele.sep</code>	the character(s) used to separate the alleles for each locus in the file (a slash by default).
<code>...</code>	further arguments passed to <code>write.table</code> .

**Value**

NULL

**Author(s)**

Emmanuel Paradis

**See Also**[read.loci](#)

**Examples**

```
require(adeigenet)
data(nancycats)
x <- as.loci(nancycats)
## print a small subset of the data:
write.loci(x[1:10, 1:3])
## use of '...':
write.loci(x[1:10, 1:3], quote = FALSE, col.names = FALSE)
```

# Index

## \*Topic **IO**

- as.loci, 4
- read.gtx, 13
- read.loci, 14
- write.loci, 24

## \*Topic **hplot**

- haploNet, 6
- haplotype, 7
- MMD, 10
- site.spectrum, 15
- summary.loci, 16

## \*Topic **htest**

- Fst, 5
- hw.test, 9
- R2.test, 12
- tajima.test, 17

## \*Topic **manip**

- haplotype, 7
- heterozygosity, 8
- nuc.div, 11
- site.spectrum, 15
- summary.loci, 16
- theta.h, 18
- theta.k, 19
- theta.s, 20
- utilities, 23

## \*Topic **models**

- amova, 2
- haploNet, 6
- theta.tree, 22

## \*Topic **package**

- pegas-package, 1

## \*Topic **univar**

- heterozygosity, 8
- nuc.div, 11
- theta.h, 18
- theta.k, 19
- theta.s, 20

[.loci (*summary.loci*), 16

adonis, 3  
amova, 2, 3  
as.loci, 4

barplot, 7, 15  
base.freq, 11

dist.dna, 13  
DNAbin, 8, 16

expand.genotype (*utilities*), 23

Fst, 5

GC.content, 11  
genind2loci (*as.loci*), 4  
getAlleles, 17  
getAlleles (*utilities*), 23  
getGenotypes (*utilities*), 23  
getPloidy (*utilities*), 23

H (*heterozygosity*), 8  
haploNet, 6  
haplotype, 7, 7, 16  
heterozygosity, 8, 19  
hw.test, 9

loci2genind (*as.loci*), 4

MMD, 10

nuc.div, 11, 21

pegas (*pegas-package*), 1  
pegas-package, 1  
plot.haploNet (*haploNet*), 6  
plot.haplotype (*haplotype*), 7  
plot.spectrum (*site.spectrum*), 15  
plot.summary.loci (*summary.loci*),  
16

print.amova (*amova*), 2

`print.loci(summary.loci)`, 16  
`print.summary.loci`  
    (`summary.loci`), 16

`R2.test`, 12  
`read.dna`, 13  
`read.genetix`, 13  
`read.gtx`, 13, 14  
`read.loci`, 4, 13, 14, 17, 24  
`rug`, 10

`seg.sites`, 11, 21  
`site.spectrum`, 15  
`summary.loci`, 16

`tajima.test`, 17  
`theta.h`, 18, 20, 21, 23  
`theta.k`, 19, 19, 21, 23  
`theta.s`, 9, 11, 19, 20, 20, 23  
`theta.tree`, 19–21, 22

`utilities`, 23

`write.loci`, 13, 14, 24