Package 'visPedigree'

January 21, 2026

Type Package

Title Tidying and Visualizing Animal Pedigrees

Version 0.7.1

Description Built on graph theory and the high-performance 'data.table' framework, this package provides a comprehensive suite of tools for tidying, pruning, and visualizing animal pedigrees. By modeling pedigrees as directed acyclic graphs using 'igraph', it ensures robust loop detection, efficient generation assignment, and sophisticated hierarchical layouts. Key features include standardizing pedigree formats, flexible ancestry tracing, and generating legible vector-based PDF graphs. A unique compaction algorithm enables the visualization of massive pedigrees (e.g., in aquaculture selective breeding population) by grouping full-sib families, maintaining structural clarity without overcrowding.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports data.table (>= 1.14.0), igraph (>= 1.3.0), nadiv (>= 2.18.0)

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0), knitr, rmarkdown, devtools

URL https://github.com/luansheng/visPedigree, https://luansheng.github.io/visPedigree/

BugReports https://github.com/luansheng/visPedigree/issues

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Sheng Luan [aut, cre]

Maintainer Sheng Luan <luansheng@gmail.com>

Repository CRAN

Date/Publication 2026-01-21 20:10:03 UTC

Contents

`big_family_size_ped` *A large pedigree with big family sizes*

Description

A dataset containing a pedigree with many full-sib individuals per family.

Usage

```
big_family_size_ped
```

Format

A data.table with 7 columns:

Ind Individual ID

Sire Sire ID

Dam Dam ID

Sex Sex of the individual

Year Year of birth

IndNum Numeric ID for individual

SireNum Numeric ID for sire

DamNum Numeric ID for dam

deep_ped *A deep pedigree*

Description

A dataset containing a pedigree with many generations.

Usage

```
deep_ped
```

Format

A data.table with 4 columns:

Ind Individual ID

Sire Sire ID

Dam Dam ID

Sex Sex of the individual

inbreed *Calculate inbreeding coefficients*

Description

`inbreed` function calculates the inbreeding coefficients for all individuals in a tidied pedigree.

Usage

```
inbreed(ped, ...)
```

Arguments

`ped`	A `tidyped` object.
`...`	Additional arguments passed to `makeDiiF`.

Details

This function takes a pedigree tidied by the `tidyped` function and calculates the inbreeding coefficients using the `makeDiiF` function from the **nadiv** package. It prefers using numeric columns (**IndNum**, **SireNum**, **DamNum**) if available, which is faster and more robust.

Value

A `tidyped` object with an additional column **f**.

loop_ped *A pedigree with loops*

Description

A dataset containing a pedigree with circular mating loops.

Usage

```
loop_ped
```

Format

A data.table with 3 columns:

Ind Individual ID

Sire Sire ID

Dam Dam ID

plot.tidyped *Plot a tidy pedigree*

Description

Plot a tidy pedigree

Usage

```
## S3 method for class 'tidyped'
plot(x, ...)
```

Arguments

x	A tidyped object.
...	Additional arguments passed to visped.

Value

Invisibly returns a list of graph data from visped (node/edge data and layout components) used to render the pedigree; the primary result is the plot drawn on the current device.

print.summary.tidyped *Print method for summary.tidyped*

Description

Print method for summary.tidyped

Usage

```
## S3 method for class 'summary.tidyped'
print(x, ...)
```

Arguments

x	A summary.tidyped object.
...	Additional arguments (ignored).

Value

The input object, invisibly.

print.tidyped *Print method for tidyped pedigree*

Description

Print method for tidyped pedigree

Usage

```
## S3 method for class 'tidyped'
print(x, ...)
```

Arguments

x	A tidyped object
...	Additional arguments passed to the data.table print method

Value

The input object, invisibly.

simple_ped *A simple pedigree*

Description

A small dataset containing a simple pedigree for demonstration.

Usage

```
simple_ped
```

Format

A data.table with 3 columns:

Ind Individual ID

Sire Sire ID

Dam Dam ID

small_ped *A small pedigree*

Description

A small dataset containing a pedigree with some missing parents.

Usage

```
small_ped
```

Format

A data.frame with 3 columns:

Ind Individual ID

Sire Sire ID

Dam Dam ID

summary.tidyped *Summary method for tidyped objects*

Description

Summary method for tidyped objects

Usage

```
## S3 method for class 'tidyped'
summary(object, ...)
```

Arguments

object	A tidyped object.
...	Additional arguments (ignored).

Value

A summary.tidyped object containing pedigree statistics.

tidyped *Tidy and prepare a pedigree using graph theory*

Description

This function takes a pedigree, checks for duplicate and bisexual individuals, detects pedigree loops using graph theory, adds missing founders, assigns generation numbers, sorts the pedigree, and traces the pedigree of specified candidates. If the cand parameter contains individual IDs, only those individuals and their ancestors or descendants are retained. Tracing direction and the number of generations can be specified using the trace and tracegen parameters.

Usage

```
tidyped(
  ped,
  cand = NULL,
  trace = "up",
  tracegen = NULL,
  addgen = TRUE,
  addnum = TRUE,
  inbreed = FALSE,
  ...
)
```

Arguments

ped	A data.table or data frame containing the pedigree. The first three columns must be **individual**, **sire**, and **dam** IDs. Additional columns, such as sex or generation, can be included. Column names can be customized, but their order must remain unchanged. Individual IDs should not be coded as "", " ", "0", "*", or "NA"; otherwise, they will be removed. Missing parents should be denoted by "NA", "0", or "*". Spaces and empty strings ("") are also treated as missing parents but are not recommended.
cand	A character vector of individual IDs, or NULL. If provided, only the candidates and their ancestors/descendants are retained.
trace	A character value specifying the tracing direction: "**up**", "**down**", or "**all**". "up" traces ancestors; "down" traces descendants; "all" traces the union of ancestors and descendants. This parameter is only used if cand is not NULL. Default is "up".
tracegen	An integer specifying the number of generations to trace. This parameter is only used if trace is not NULL. If NULL or 0, all available generations are traced.
addgen	A logical value indicating whether to generate generation numbers. Default is TRUE, which adds a **Gen** column to the output.
addnum	A logical value indicating whether to generate a numeric pedigree. Default is TRUE, which adds **IndNum**, **SireNum**, and **DamNum** columns to the output.
inbreed	A logical value indicating whether to calculate inbreeding coefficients. Default is FALSE. If TRUE, an **f** column is added to the output.
...	Additional arguments passed to inbreed.

Details

Compared to the legacy version, this function handles cyclic pedigrees more robustly by detecting and reporting loops, and it is generally faster for large pedigrees due to the use of sparse graph algorithms from the igraph package. Generation assignment is performed using a topological sorting-based algorithm that ensures parents are always placed in a generation strictly above their offspring (TGI algorithm).

Value

A tidyped object (which inherits from data.table). Individual, sire, and dam ID columns are renamed to **Ind**, **Sire**, and **Dam**. Missing parents are replaced with **NA**. The **Sex** column contains "male", "female", or NA. The **Cand** column is included if cand is not NULL. The **Gen** column is included if addgen is TRUE. The **IndNum**, **SireNum**, and **DamNum** columns are included if addnum is TRUE. The **f** column is included if inbreed is TRUE.

See Also

summary.tidyped

Examples

```
library(visPedigree)
library(data.table)

# Tidy a simple pedigree
tidy_ped <- tidyped(simple_ped)
head(tidy_ped)

# Trace ancestors of a specific individual within 2 generations
tidy_ped_tracegen <- tidyped(simple_ped, cand = "J5X804", trace = "up", tracegen = 2)
head(tidy_ped_tracegen)

# Trace both ancestors and descendants for multiple candidates
# This is highly optimized and works quickly even on 100k+ individuals
cand_list <- c("J5X804", "J3Y620")
tidy_ped_all <- tidyped(simple_ped, cand = cand_list, trace = "all")

# Check for loops (will error if loops exist)
try(tidyped(loop_ped))

# Example with a large pedigree: extract 2 generations of ancestors for 2007 candidates
cand_2007 <- big_family_size_ped[Year == 2007, Ind]

tidy_big <- tidyped(big_family_size_ped, cand = cand_2007, trace = "up", tracegen = 2)
summary(tidy_big)
```

`visped` *Visualize a tidy pedigree*

Description

`visped` function draws a graph of a full or compact pedigree.

Usage

```
visped(
  ped,
  compact = FALSE,
  outline = FALSE,
  cex = NULL,
  showgraph = TRUE,
  file = NULL,
  highlight = NULL,
  trace = FALSE,
  showf = FALSE,
  pagewidth = 200,
  symbolsize = 1,
```

```
  maxiter = 1000,
  ...
)
```

Arguments

ped	A tidyped object (which inherits from data.table). It is recommended that the pedigree is tidied and pruned by candidates using the tidyped function with the non-null parameter cand.
compact	A logical value indicating whether IDs of full-sib individuals in one generation will be removed and replaced with the number of full-sib individuals. For example, if there are 100 full-sib individuals in one generation, they will be replaced with a single label "100" when compact = TRUE. The default value is FALSE.
outline	A logical value indicating whether shapes without labels will be shown. A graph of the pedigree without individual labels is shown when setting outline = TRUE. This is useful for viewing the pedigree outline and identifying immigrant individuals in each generation when the graph width exceeds the maximum PDF width (500 inches). The default value is FALSE.
cex	NULL or a numeric value changing the size of individual labels shown in the graph. *cex* is an abbreviation for 'character expansion factor'. The visped function will attempt to estimate (cex=NULL) the appropriate cex value and report it in the messages. Based on the reported cex from a previous run, this parameter should be increased if labels are wider than their shapes in the PDF; conversely, it should be decreased if labels are narrower than their shapes. The default value is NULL.
showgraph	A logical value indicating whether a plot will be shown in the default graphic device (e.g., the Plots panel in RStudio). This is useful for quick viewing without opening a PDF file. However, the graph on the default device may not be legible (e.g., overlapping labels or aliasing lines) due to size restrictions. It is recommended to set showgraph = FALSE for large pedigrees. The default value is TRUE.
file	NULL or a character value specifying whether the pedigree graph will be saved as a PDF file. The PDF output is a legible vector drawing where labels do not overlap, even with many individuals or long labels. It is recommended to save the pedigree graph as a PDF file. The default value is NULL.
highlight	NULL, a character vector of individual IDs, or a list specifying individuals to highlight. If a character vector is provided, individuals will be highlighted with a purple border while preserving their sex-based fill color. If a list is provided, it should contain: • ids: (required) character vector of individual IDs to highlight. • frame.color: (optional) hex color for the border of focal individuals. • color: (optional) hex color for the fill of focal individuals. • rel.frame.color: (optional) hex color for the border of relatives (used when trace is not NULL). • rel.color: (optional) hex color for the fill of relatives (used when trace is not NULL).

For example: `c("A", "B")` or `list(ids = c("A", "B"), frame.color = "#9c27b0")`. The function will check if the specified individuals exist in the pedigree and issue a warning for any missing IDs. The default value is NULL.

`trace` A logical value or a character string. If TRUE, all ancestors and descendants of the individuals specified in `highlight` will be highlighted. If a character string, it specifies the tracing direction: "**up**" (ancestors), "**down**" (descendants), or "**all**" (union of ancestors and descendants). This is useful for focusing on specific families within a large pedigree. The default value is FALSE.

`showf` A logical value indicating whether inbreeding coefficients will be shown in the graph. If `showf = TRUE` and the column **f** exists in the pedigree, the inbreeding coefficient will be appended to the individual label, e.g., "ID (0.05)". The default value is FALSE.

`pagewidth` A numeric value specifying the width of the PDF file in inches. This controls the horizontal scaling of the layout. The default value is 200.

`symbolsize` A numeric value specifying the scaling factor for node size relative to the label size. Values greater than 1 increase the node size (adding padding around the label), while values less than 1 decrease it. This is useful for fine-tuning the whitespace and legibility of dense graphs. The default value is 1.

`maxiter` An integer specifying the maximum number of iterations for the Sugiyama layout algorithm to minimize edge crossings. Higher values (e.g., 2000 or 5000) may result in fewer crossed lines for complex pedigrees but will increase computation time. The default value is 1000.

`...` Additional arguments passed to `plot.igraph`.

Details

This function takes a pedigree tidied by the `tidyped` function and outputs a hierarchical graph for all individuals in the pedigree. The graph can be shown on the default graphic device or saved as a PDF file. The PDF output is a legible vector drawing that is legible and avoids overlapping labels. It is especially useful when the number of individuals is large and individual labels are long.

Rendering is performed using a Two-Pass strategy: edges are drawn first to ensure center-to-center connectivity, followed by nodes and labels. This ensures perfect visual alignment in high-resolution vector outputs. The function also supports real-time ancestry and descendant highlighting.

This function can draw the graph of a very large pedigree (> 10,000 individuals per generation) by compacting full-sib individuals. It is highly effective for aquatic animal pedigrees, which usually include many full-sib families per generation in nucleus breeding populations. The outline of a pedigree without individual labels is still shown if the width of a pedigree graph exceeds the maximum width (500 inches) of the PDF file.

In the graph, two shapes and three colors are used. Circles represent individuals, and squares represent families. Dark sky blue indicates males, dark goldenrod indicates females, and dark olive green indicates unknown sex. For example, a dark sky blue circle represents a male individual; a dark goldenrod square represents all female individuals in a full-sib family when `compact = TRUE`.

Value

No returned values. The graph will be plotted directly on graphic devices.

Note

Isolated individuals (those with no parents and no progeny, assigned Gen 0) are automatically filtered out and not shown in the plot. A message will be issued if any such individuals are removed.

Examples

```
library(visPedigree)
library(data.table)
# Drawing a simple pedigree
simple_ped_tidy <- tidyped(simple_ped)
visped(simple_ped_tidy)

# Highlighting an individual and its ancestors and descendants
visped(simple_ped_tidy, highlight = "J5X804", trace = "all")

# Showing inbreeding coefficients in the graph
simple_ped_tidy_inbreed <- tidyped(simple_ped, inbreed = TRUE)
visped(simple_ped_tidy_inbreed, showf = TRUE)

# Adjusting page width and symbol size for better layout
# Increase pagewidth to spread nodes horizontally
# Increase symbolsize for more padding around individual labels
visped(simple_ped_tidy, pagewidth = 100, symbolsize = 1.2)

# Highlighting multiple individuals with custom colors
visped(simple_ped_tidy,
       highlight = list(ids = c("J3Y620", "J1X971"),
                        frame.color = "#4caf50",
                        color = "#81c784"))

# Handling large pedigrees: Saving to PDF is recommended for legibility
# The 'trace' and 'tracegen' parameters in tidyped() help prune the graph
cand_labels <- big_family_size_ped[(Year == 2007) & (substr(Ind,1,2) == "G8"), Ind]

big_ped_tidy <- tidyped(big_family_size_ped, cand = cand_labels, trace = "up", tracegen = 2)
# Use compact = TRUE for large families
visped(big_ped_tidy, compact = TRUE, file = tempfile(fileext = ".pdf"))

# Use outline = TRUE if individual labels are not required
visped(big_ped_tidy, compact = TRUE, outline = TRUE, file = tempfile(fileext = ".pdf"))
```

Index