Package 'KLINK'

August 3, 2026

Title Kinship Analysis with Linked Markers

Version 1.2.2

Description A 'shiny' application for forensic kinship testing, based on the 'pedsuite' R packages. 'KLINK' is closely aligned with the (non-R) software 'Familias' and 'FamLink', but offers several unique features, including visualisations and automated report generation. The calculation of likelihood ratios supports pairs of linked markers, and all common mutation models. The program is described in Vigeland and Gilfillan (2026) <doi:10.1016/j.fsigen.2026.103578>.

License GPL (>= 3)

URL https://github.com/magnusdv/KLINK,
https://doi.org/10.1016/j.fsigen.2026.103578

BugReports https://github.com/magnusdv/KLINK/issues

Depends R (>= 4.2)

Imports forrel (>= 1.9.0), gt, norSTR, openxlsx, pedFamilias, pedmut, pedprobr (>= 1.1.0), pedtools (>= 2.11.0), scales, shiny (>= 1.9.0), shinyBS, shinydashboard, shinyjs, verbalisr, xml2

Config/roxygen2/version 8.0.0

Encoding UTF-8

Language en-GB

LazyData true

NeedsCompilation no

Author Magnus Dehli Vigeland [aut, cre] (ORCID: <https://orcid.org/0000-0002-9134-4962>)

Maintainer Magnus Dehli Vigeland <m.d.vigeland@medisin.uio.no>

Repository CRAN

Date/Publication 2026-08-03 16:10:02 UTC

Contents

getLinkedPairs . . . 2
getTotals . . . 3
halfsib . . . 3
launchApp . . . 4
linkedLR . . . 5
loadFamFile . . . 6
loadMap . . . 7
map50 . . . 7
markerSummary . . . 8
parseXML . . . 8
paternity . . . 9
sibship . . . 9
writeResult . . . 10

Index **12**

`getLinkedPairs` *Identify linked marker pairs*

Description

This function returns a "maximal" set of disjoint pairs of linked markers, given a genetic marker map and a subset of the markers included in the map. The pairs are identified in a greedy manner, successively choosing the closest markers on each chromosome.

Usage

```
getLinkedPairs(markers, linkageMap = map50, maxdist = Inf)
```

Arguments

`markers`	A character vector containing marker names.
`linkageMap`	A data frame with columns including `Marker`, `Chr` and `cM`.
`maxdist`	A positive number indicating the maximum linkage distance (in cM). Markers further apart than this are considered unlinked.

Value

A list of character vectors, each containing two marker names.

Examples

```
# Example using the default map of 50 STR markers
map = norSTR::map50

getLinkedPairs(map$Marker, map, maxdist = 25)
```

getTotals *Get LR totals*

Description

Find, or add, the LR totals in a data frame with marker-wise LR values. The totals are found by multiplying the values in each column, after removing NA's.

Usage

```
getTotals(x, cols = c("LRlinked", "LRnolink", "LRnomut"))

addTotals(x, cols = c("LRlinked", "LRnolink", "LRnomut"))
```

Arguments

x	A data frame with LR results, typically the output of linkedLR().
cols	A vector of column names, by default c("LRlinked", "LRnolink", "LRnomut")

Details

Note that for the multiplication to respect linked markers, the input table should include LR values for only one marker in each linkage group, and NA's elsewhere. This is the format used by linkedLR().

Value

getTotals() returns a named numeric. addTotals returns a data frame equal to the input, but with a row of totals added at the bottom.

Examples

```
res = linkedLR(paternity)
getTotals(res)
addTotals(res)
```

halfsib *Dataset for a case involving a putative half sibling.*

Description

A list of two pedigrees forming opposing hypotheses about three individuals:

Usage

```
halfsib
```

Format

A list of two ped objects, named H1 and H2.

Details

- H1: A and B are full siblings, and C is their half brother
- H2: A and B are full siblings, and C is unrelated to them

The individuals are typed with 50 markers, but some genotypes are missing.

Examples

```
library(pedtools)
plotPedList(halfsib, hatched = typedMembers)
markerSummary(halfsib)
linkedLR(halfsib)
```

---

launchApp *Launch KLINK*

Description

This launches the KLINK app. `runKLINK()` is a synonym for `launchApp()`, but with an additional argument `version`.

Usage

```
launchApp()

runKLINK(version = NULL)
```

Arguments

`version`	A character, e.g. "1.0.0". If the installed version of KLINK differs from this, the program aborts with an error.

Value

No return value, called for side effects.

Examples

```
## Not run:
launchApp()

## End(Not run)
```

linkedLR *LR with pairwise linked markers*

Description

This function does the main LR calculations of the KLINK app.

Usage

```
linkedLR(
  pedigrees,
  linkageMap = map50,
  linkedPairs = NULL,
  maxdist = Inf,
  markerData = NULL,
  mapfun = "Kosambi",
  lumpSpecial = TRUE,
  alleleLimit = 10,
  verbose = TRUE,
  debug = FALSE
)
```

Arguments

pedigrees	A list of two pedigrees.
linkageMap	A data frame with columns including Marker, Chr and cM. By default, a built-in map norSTR::map50 of 50 STR markers is used.
linkedPairs	A list of marker pairs. If not supplied, calculated as getLinkedPairs(markerData$Marker, linkageMap, maxdist = maxdist).
maxdist	A number, passed onto getLinkedPairs() if linkedPairs is NULL.
markerData	A data frame with marker data, typically the output of markerSummary(pedigrees).
mapfun	Name of the map function to be used; either "Haldane" or "Kosambi" (default).
lumpSpecial	A logical indicating if special lumping should be activated. This is strongly recommended in all cases with linked STR markers.
alleleLimit	A number, by default 10, passed on to pedprobr::likelihood2().
verbose	A logical, by default TRUE.
debug	A logical, by default FALSE.

Value

A data frame with detailed LR results.

Examples

```
linkedLR(paternity)

# Detailed messages, including reports on lumping
linkedLR(paternity, debug = TRUE)

# For testing
# .linkedLR(paternity, markerpair = c("SE33", "D6S474"))
```

`loadFamFile` *Load* `.fam` *file*

Description

Load `.fam` file

Usage

```
loadFamFile(path, fallbackModel = "equal", withParams = FALSE)
```

Arguments

`path`	The path to a `.fam` file.
`fallbackModel`	The name of a mutation model; passed on to `pedFamilias::readFam()`.
`withParams`	A logical indicating if the Familias parameters should be included in the output. (See `pedFamilias::readFam()`.)

Value

If `withParams = FALSE`, a list of two pedigree hypotheses in the form of `ped` objects. If `withParams = TRUE`, a list with components `peds` and `params`.

Examples

```
fam = system.file("extdata/halfsib-test.fam", package = "KLINK")
peds = loadFamFile(fam)
pedtools::plotPedList(peds)

loadFamFile(fam, withParams = TRUE)$params
```

loadMap *Load genetic map*

Description

Loads a genetic map from a tab-separated text file, or from the first sheet of an Excel file with extension `.xlsx`.

Usage

```
loadMap(path)
```

Arguments

`path`	Path to the map file, either as plain text or Excel.

Details

The file should contain columns `Marker`, `Chr` and `cM` (slight variations in column names are allowed). Chromosomes should be positive integers, optionally prefixed by "chr". Positions must be nonnegative numeric values. Both period and comma are accepted as decimal separators.

Value

A data frame with standardised column names and rows sorted by chromosome and position.

Examples

```
tmp = tempfile(fileext = ".map")
map1 = as.data.frame(norSTR::map50)
write.table(map1, tmp, sep = "\t", quote = FALSE, row.names = FALSE)

map2 = loadMap(tmp)
stopifnot(all.equal(map1, map2))
```

map50 *Genetic marker map*

Description

This object is re-exported from norSTR::map50.

Usage

```
map50
```

markerSummary *Generate table of marker data*

Description

Generate table of marker data

Usage

```
markerSummary(pedigrees, replaceNames = FALSE)
```

Arguments

pedigrees	A list of 2 pedigrees.
replaceNames	A logical, indicating if IDs should be changed to Person1, Person2, ...

Value

A data frame.

Examples

```
markerSummary(paternity)
```

parseXML *Parse* XML *file associated with* .fam *file*

Description

Parse XML file associated with .fam file

Usage

```
parseXML(xml)
```

Arguments

xml	Path to a file with extension .xml.

Examples

```
# (No example included)
```

paternity *Dataset for a paternity case*

Description

A list of two pedigrees forming the hypotheses in a paternity case: H1 (AF is the father of CH) and H2 (unrelated). AF and CH are genotyped with 11 markers, with allele frequencies from norSTR::norwayDB.

Usage

```
paternity
```

Format

A list of two pedigrees, named H1 and H2.

Examples

```
pedtools::plotPedList(paternity, marker = "SE33")
markerSummary(paternity)

forrel::kinshipLR(paternity)
```

sibship *Dataset for a full vs half sibship case*

Description

A list of two pedigrees forming opposing hypotheses about two individuals NN1 and NN2: FS (full siblings) vs. HS (half siblings). Both individuals are genotyped with the 23 markers included in the Fusion 6C kit. Among the 23 markers, four pairs are linked:

Usage

```
sibship
```

Format

A list of two ped objects, named FS and HS.

Details

- TPOX and D2S441 (88.8 cM, 9 & 13 alleles)
- D5S818 and CSF1PO (27.8 cM, 9 & 11 alleles)
- vWA and D12S391 (11.9 cM, 12 & 23 alleles)
- D21S11 and Penta D (44.7 cM, 26 & 24 alleles)

Examples

```
library(pedtools)
plotPedList(sibship, hatched = typedMembers)
markerSummary(sibship)
linkedLR(sibship)
```

`writeResult` *Write data and results to Excel*

Description

This function produces an Excel document containing the genotype data and various LR tables.

Usage

```
writeResult(
  resultTable,
  pedigrees,
  linkageMap,
  markerData,
  outfile,
  notes = NULL,
  famname = NULL,
  hideEmpty = FALSE,
  settings = NULL,
  XML = NULL
)
```

Arguments

`resultTable`	A data frame.
`pedigrees`	A list of two ped objects.
`linkageMap`	A data frame.
`markerData`	A data frame.
`outfile`	The output file name.
`notes`	A character vector.
`famname`	The name of the input `.fam` file.
`hideEmpty`	A logical, indicating if untyped markers should be dropped.
`settings`	A list of KLINK settings to be included in the output
`XML`	Optional data from .xml file.

Examples

```
# Built-in dataset `paternity`
peds = paternity
map = norSTR::map50
mdata = markerSummary(peds)

# Result table
LRtab = linkedLR(pedigrees = peds, linkageMap = map, markerData = mdata, verbose = FALSE)

# Write to excel
tmp = paste0(tempfile(), ".xlsx")
writeResult(LRtab,
            pedigrees = peds,
            linkageMap = map,
            markerData = mdata,
            outfile = tmp)

# openxlsx::openXL(tmp)
```

Index

∗ **datasets**
halfsib, 3
paternity, 9
sibship, 9

addTotals *(*getTotals*)*, 3

getLinkedPairs, 2
getTotals, 3

halfsib, 3

launchApp, 4
linkedLR, 5
linkedLR(), *3*
loadFamFile, 6
loadMap, 7

map50, 7
markerSummary, 8

norSTR::map50, *7*

parseXML, 8
paternity, 9
pedFamilias::readFam(), *6*
pedprobr::likelihood2(), *5*

runKLINK *(*launchApp*)*, 4

sibship, 9

writeResult, 10