Package 'archiDART'

December 22, 2015

Type Package

Title Plant Root System Architecture Analysis Using DART and RSML Files

Version 1.2

Date 2015-12-22

Author Benjamin M Delory, Caroline Baudson, Yves Brostaux, Guillaume Lobet, Patrick du Jardin, Loic Pages, Pierre Delaplace

Maintainer Benjamin M Delory <Benjamin.Delory@leuphana.de>

Description Analysis of complex plant root system architectures (RSA) using the output files created by Data Analysis of Root Tracings (DART), an open-access software dedicated to the study of plant root architecture and development across time series (Le Bot et al (2010) DART: a software to analyse root system architecture and development from captured images, Plant and Soil, DOI: 10.1007/s11104-009-0005-2), and RSA data encoded with the Root System Markup Language (RSML) (Lobet et al (2015) Root System Markup Language: toward a unified root architecture description language, Plant Physiology, DOI: 10.1104/pp.114.253625). More information can be found in Delory et al (2015) archiDART: an R package for the automated computation of plant root architectural traits, Plant and Soil, DOI: 10.1007/s11104-015-2673-4.

License GPL-2

Imports stats, utils, graphics, grDevices, XML

NeedsCompilation no

R topics documented:

archidraw 2
archigrow 5
architect 8
ch7lie 12
ch7rac 13
ch7tps 14
latdist 15
trajectory 18

Index **23**

archidraw *X-Y Plotting of Vectorized Root Systems*

Description

X-Y plotting of vectorized root systems for selected observation dates using Data Analysis of Root Tracings (DART) output files and RSA data encoded with the Root System Markup Language (RSML). More information can be found in *Delory et al (2015).*

Usage

```
archidraw(inputlie=NULL, inputrsml=NULL, res=NULL, unitlength="px", unitangle="d",
          rotation=0, rsml.connect=FALSE, numdate=NULL, finalscale=NULL,
          coldate=par("col"), ltydate=par("lty"), lwddate=par("lwd"),
          main=NULL, xlab=NULL, ylab=NULL, xlim=NULL, ylim=NULL, ...)
```

Arguments

`inputlie` A character string specifying the path to the folder containing the lie files created by DART. This argument could be the same as `inputrsml`.

`inputrsml` A character string specifying the path to the folder containing the rsml files. This argument could be the same as `inputlie`.

`res` **If images were acquired with a flatbed scanner**: a numeric value specifying the resolution of the images used to vectorize the root systems (resolution is expressed in dots/inch). **If images were acquired with a digital camera**: the ratio between the length of a reference object located on the image expressed in pixels and the actual length of the same object expressed in inches. This argument must be specified only if `unitlength="mm"` or `unitlength="cm"`.

`unitlength` A character string specifying the unit of length that must be used by the function to scale the X and Y axes of the graphical outputs. The value acceptable for this argument could be either “px” for pixels, “cm” for centimetres or “mm” for millimetres.

`unitangle` A character string specifying the unit that must be used by the function to express the calculated angles. The value acceptable for this argument could be either “d” for degrees (default value) or “r” for radians.

`rotation` A numeric value specifying the angle (expressed in `unitangle`) that must be used by the function to perform a clockwise rotation of the root system.

`rsml.connect` A logical value that must be specified when `is.null(inputrsml)=FALSE`. This argument will be used only for RSA data encoded with the Root System Markup Language. If `TRUE`, the lateral roots will be connected to their parent root if no connection information can be found in the rsml file(s) (see details). If `FALSE`, no connection will be made between daughter and mother roots if no connection information can be found in the rsml file(s).

`numdate` A numeric value or a vector of numeric values specifying the identification number of all the observation dates at which the root system architecture must be plotted (see details).

`finalscale` A logical value that must be specified when `is.null(numdate)=FALSE`. If `TRUE`, the root system architecture at the selected observation dates will be plotted with the same X and Y-axis scales to that used to draw the root system architecture at the last observation date. If `FALSE`, the root system architecture at each observation date will be plotted using different X and Y-axis ranges of coordinates (`xlim` and `ylim`).

`coldate` A vector specifying the colour(s) that must be used to draw the root system architecture at the selected observation dates (see details).

`ltydate` A vector specifying the line type(s) that must be used to draw the root system architecture at the selected observation dates (see details).

`lwddate` A vector specifying the line width(s) that must be used to draw the root system architecture at the selected observation dates (see details).

`main` A character string giving a main title for the plot. When `is.null(main)=TRUE`, the default title for each plot is the name of the corresponding lie/rsml file.

`xlab` A character string giving a label for the X axis. When `is.null(xlab)=TRUE`, the default label for the X axis is X (`unitlength`).

`ylab` A character string giving a label for the Y axis. When `is.null(ylab)=TRUE`, the default label for the Y axis is Y (`unitlength`).

`xlim` A numeric vector (length=2) giving the X limits of the plot using the same unit as `unitlength`.

`ylim` A numeric vector (length=2) giving the Y limits of the plot using the same unit as `unitlength`.

`...` Additional graphical parameters (see details).

Details

Le Bot et al (2010) presented a free and open-access software (Data Analysis of Root Tracings - DART) allowing the analysis of complex root system architectures from captured images, particularly across time series. Using this software, a user has to manually identify roots as a set of links. At the end of the vectorization process, a final dataset can be exported as a table file (lie file) containing several attributes for each point used to construct the links constituting the vectorized root system. Using these lie files created by DART, `archidraw` allows the X-Y plotting of vectorized root systems for selected observation dates.

The `archidraw` function will locate and read the lie files created by DART and plot the vectorization results at selected observation dates for each analysed root system. First, `archidraw` will check if all arguments have been written in the suitable mode. If `res` and `rotation` are specified, the function will check if the numeric values required by these arguments are positive. If `is.null(numdate)=FALSE`, the function will also automatically check if the numeric values in `numdate` are positive and sorted by increasing values. If an error occurs for one argument, the code stops and the corresponding error message will be returned by the function. Second, the function will use `inputlie` to locate and read the raw lie files created by DART. To run `archidraw` efficiently, DART files must have been saved with their appropriate extensions (.lie). Finally, `archidraw` will plot each vectorized root system located in `inputlie` at the observation dates specified by `numdate`. By default (when `is.null(numdate)=TRUE`), only the root system architecture at the last observation date will be plotted. If `is.null(numdate)=FALSE`, `archidraw` will plot only the root system architecture for the selected observation dates. If the number of elements specified in `coldate`/`ltydate`/`lwddate` is lower than `max(Date)` in the lie files (if `is.null(numdate)=TRUE`) or `max(numdate)` (if `is.null(numdate)=FALSE`), `archidraw` will replicate the elements stored in `coldate`/`ltydate`/`lwddate` to draw the vectorization results. When the number of elements specified in `coldate`/`ltydate`/`lwddate` is greater than `max(Date)` in the lie files (if `is.null(numdate)==TRUE`) or `max(numdate)` (if `is.null(numdate)=FALSE`), only the first corresponding elements in `coldate`/`ltydate`/`lwddate` will be used by `archidraw` to plot the root system architecture for each file located in `inputlie`.

If RSA data have been encoded with the Root System Markup Language (*Lobet et al, 2015*), the path to the folder containing the rsml files can be specified using the `inputrsml` argument. To run `archidraw` efficiently, rsml files must have been saved with their appropriate extensions (.rsml). If `is.null(inputrsml)=FALSE`, `archidraw` will check if `rsml.connect` is specified and written in the suitable mode. After reading the rsml files located in `inputrsml`, the RSA data associated with one root system in each rsml file will be converted into a data frame possessing the same structure as the lie files created by DART. The `archidraw` function will then use these data frames to plot the root system architectures encoded with the RSML format. As some rsml files do not provide connection information between daughter and mother roots, the `rsml.connect` argument can be used to connect the lateral roots to their corresponding mother root. If `rsml.connect=TRUE`, each point starting a lateral root will be connected to the nearest point located on its mother root. Using rsml files, only RSA data associated with roots possessing a branching order lower or equal to 5 will be used for the computation of graphical outputs.

Due to technical limitations, it is sometimes easier to take a picture of a root system if it is rotated before image acquisition. In this case, the vectorized root system will depict a rotation compared with the natural plant root system. In order to make a correction, one can use `rotation` to specify an angle value expressed in `unitangle` that will be used by the function to rotate the vectorized root system clockwise before plotting.

Additional graphical parameters can also be used to personalize the graphical outputs (see `par`).

Value

Returns a X-Y plot for each vectorized root system located in `inputlie`/`inputrsml`.

Author(s)

Benjamin M. Delory, Caroline Baudson, Yves Brostaux, Guillaume Lobet, Loic Pages, Patrick du Jardin, Pierre Delaplace

References

Delory B.M., Baudson C., Brostaux Y., Lobet G., du Jardin P., Pages L., Delaplace P. (2015) archiDART: an R package for the automated computation of plant root architectural traits, *Plant and Soil*, DOI: 10.1007/s11104-015-2673-4.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

Lobet G., Pound M.P., Diener J., Pradal C., Draye X., Godin C., Javaux M., Leitner D., Meunier F., Nacry P., Pridmore T.P., Schnepf A. (2015) Root System Markup Language: Toward a Unified Root Architecture Description Language, *Plant Physiology*, DOI: 10.1104/pp.114.253625.

See Also

`par`.

Examples

```
## Importation of an example dataset
data(ch7lie)
```

```
## Creation of a folder (called DART_archidraw) inside a temporary directory to contain lie
## example datasets created by DART
dirDART <- paste(tempdir(), "/DART_archidraw", sep="")
dir.create(dirDART)

## Definition of the number of lie files that will be saved inside dirDART
n <- 1

## Creation of lie files inside dirDART
for (i in 1:n) {
write.table(ch7lie, file=paste(dirDART,"/ch7_",i,".lie", sep=""), row.names=FALSE)}

## Use of archidraw to plot the root system architecture in black for the last observation date
archidraw(inputlie=dirDART, res=75, unitlength="cm", lwddate=2, las=1, bty="l", asp=1,
xaxp=c(0,30,3), yaxp=c(0,90,9))


## Use of archidraw to plot the root system architecture at the last observation date
## Use a specific colour for each link related to a single observation date
archidraw(inputlie=dirDART, res=75, unitlength="cm", coldate=rainbow(31), lwddate=2, las=1,
bty="l", asp=1, xaxp=c(0,30,3), yaxp=c(0,90,9))

## Use of archidraw to plot the root system architecture at selected observation dates
## Use the same X and Y-axis scales to that used for the last observation date and a specific colour
## for each observation date
archidraw(inputlie=dirDART, res=75, unitlength="cm", numdate=c(15,30), finalscale=TRUE,
coldate=rainbow(31), lwddate=2, las=1, bty="l", asp=1, xaxp=c(0,30,3), yaxp=c(0,90,9))

unlink(dirDART, recursive=TRUE)
```

archigrow *Computing Growth Rates and X-Y Plotting of Vectorized Root Systems*

Description

Calculation of growth rate matrices and X-Y plotting of vectorized root systems for selected observation dates using Data Analysis of Root Tracings (DART) output files. In the final plots, the colour code used for each link constituting the vectorized root systems depends on their corresponding growth rate value. More information can be found in *Delory et al (2015).*

Usage

```
archigrow(inputlie, inputtps, res=NULL, unitlength="px", unittime,
          unitangle="d", rotation=0, numdate=NULL, finalscale=NULL,
          coldyn, GRscale=NULL, main=NULL, xlab=NULL, ylab=NULL,
          xlim=NULL, ylim=NULL, ...)
```

Arguments

`inputlie`	A character string specifying the path to the folder containing the lie files created by DART. This argument could be the same as `inputtps`.
`inputtps`	A character string specifying the path to the folder containing the tps files created by DART. This argument could be the same as `inputlie`.

`res` **If images were acquired with a flatbed scanner**: a numeric value specifying the resolution of the images used to vectorize the root systems (resolution is expressed in dots/inch). **If images were acquired with a digital camera**: the ratio between the length of a reference object located on the image expressed in pixels and the actual length of the same object expressed in inches. This argument must be specified only if `unitlength="mm"` or `unitlength="cm"`.

`unitlength` A character string specifying the unit of length that must be used by the function both to calculate the growth rate matrices and to scale the X and Y axes of the graphical outputs. The value acceptable for this argument could be either “px” for pixels, “cm” for centimetres or “mm” for millimetres.

`unittime` A character string specifying the unit of time used to express the data in the tps files created by DART (all character strings are valid for this argument).

`unitangle` A character string specifying the unit that must be used by the function to express the calculated angles. The value acceptable for this argument could be either “d” for degrees (default value) or “r” for radians.

`rotation` A numeric value specifying the angle (expressed in `unitangle`) that must be used by the function to perform a clockwise rotation of the root system (see details).

`numdate` A numeric value or a vector of numeric values specifying the identification number of all the observation dates at which the root system architecture must be plotted (see details).

`finalscale` A logical value that must be specified when `is.null(numdate)=FALSE`. If `TRUE`, the root system architecture at the selected observation dates will be plotted with the same X and Y-axis scales to that used to draw the root system architecture at the last observation date. If `FALSE`, the root system architecture at each observation date will be plotted using different X and Y-axis ranges of coordinates (`xlim` and `ylim`).

`coldyn` A vector specifying the colour(s) that must be used to draw the root system architecture at the selected observation dates. The function will colour each link constituting a vectorized root system depending on its corresponding growth rate value. To do so, `archigrow` interpolates the colour(s) contained in `coldyn` between the minimum and the maximum values of the computed standardized growth rate matrix (if `is.null(GRscale)=TRUE`) or between `min(GRscale)` and `max(GRscale)` (if `is.null(GRscale)=FALSE`).

`GRscale` A numeric vector (length=2) specifying the minimum and the maximum growth rate values (expressed in `unitlength/unittime`) that must be used by the function to plot each vectorized root system.

`main` A character string giving a main title for the plot. When `is.null(main)=TRUE`, the default title for each plot is the name of the corresponding lie file.

`xlab` A character string giving a label for the X axis. When `is.null(xlab)=TRUE`, the default label for the X axis is X (`unitlength`).

`ylab` A character string giving a label for the Y axis. When `is.null(ylab)=TRUE`, the default label for the Y axis is Y (`unitlength`).

`xlim` A numeric vector (length=2) giving the X limits of the plot using the same unit as `unitlength`.

`ylim` A numeric vector (length=2) giving the Y limits of the plot using the same unit as `unitlength`.

`...` Additional graphical parameters (see details)

Details

Le Bot et al (2010) presented a free and open-access software (Data Analysis of Root Tracings - DART) allowing the analysis of complex root system architectures from captured images, particularly across time series. Using this software, a user has to manually identify roots as a set of links. At the end of the vectorization process, a final dataset can be exported as a table file (lie file) containing several attributes for each point used to construct the links constituting the vectorized root system. A second table file (tps file) containing the root system age at each observation date can also be exported from DART. Using these files created by DART, `archigrow` allows the X-Y plotting of vectorized root systems for selected observation dates with a colour code depending on the growth rate value of each link constituting the vectorized root systems.

The `archigrow` function will successively locate and read the lie and tps files created by DART, compute a growth rate matrix and plot the vectorization results at selected observation dates for each analysed root system. First, `archigrow` will check if all arguments have been written in the suitable mode. If `res` and `rotation` are specified, the function will check if the numeric values required by these arguments are positive. If `is.null(numdate)=FALSE`, the function will also automatically check if the numeric values in `numdate` are positive and sorted by increasing values. If an error occurs for one argument, the code stops and the corresponding error message will be returned by the function. Second, the function will use `inputlie` and `inputtps` to locate and read the raw lie and tps files created by DART. To run `archigrow` efficiently, DART files must have been saved with their appropriate extensions (.lie or .tps). If `inputtps` contains a single tps file, it will be used by `archigrow` for each lie file located in `inputlie`. If `inputps` contains more than one tps file, the number of tps files in `inputtps` must be equal to the number of lie files in `inputlie` and corresponding lie and tps files must have the same name. If an error occurs at this step, a message will be returned by the function. Third, `archigrow` will compute a growth rate matrix containing for each root constituting the vectorized root system its corresponding growth rate value at each observation date. Finally, `archigrow` will plot each vectorized root system located in `inputlie` at the observation dates specified by `numdate`. By default (when `is.null(numdate)=TRUE`), only the root system architecture at the last observation date will be plotted. If `is.null(numdate)=FALSE`, `archigrow` will plot only the root system architecture for the selected observation dates.

Due to technical limitations, it is sometimes easier to take a picture of a root system if it is rotated before image acquisition. In this case, the vectorized root system will depict a rotation compared with the natural plant root system. To make a correction, one can use `rotation` to specify an angle value expressed in `unitangle` that will be used by the function to rotate the vectorized root system clockwise before plotting and root system architecture parameters calculation.

Additional graphical parameters can also be used to personalize the graphical outputs (see `par`). By default, a legend showing the colour gradient used to draw each vectorized root system is shown at the right side of the final plot.

Growth rates are calculated as follows:

- For the first observation date, it is calculated as the ratio of the root length to the root system age.
- For other observation dates (t), it is calculated as the difference between the root length at time t and t-1 divided by the difference between the root system age at time t and t-1.

Value

Returns the computed growth rate matrices in a list of data frames. Each element of the list is named as its corresponding lie file and contains the following columns: **Root** (the identification number of each root constituting a vectorized root system), **GR.DateX** (the calculated growth rate value of a single root at the observation date X expressed in `unitlength/unittime`).

Returns a X-Y plot for each vectorized root system located in `inputlie`.

Author(s)

Benjamin M. Delory, Caroline Baudson, Yves Brostaux, Loic Pages, Patrick du Jardin, Pierre Delaplace

References

Delory B.M., Baudson C., Brostaux Y., Lobet G., du Jardin P., Pages L., Delaplace P. (2015) archiDART: an R package for the automated computation of plant root architectural traits, *Plant and Soil*, DOI: 10.1007/s11104-015-2673-4.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

See Also

par.

Examples

```
## Importation of example datasets
data(ch7lie)
data(ch7tps)

## Creation of a folder (DART_archigrow) inside a temporary directory to contain the lie and tps
## example datasets created by DART
dirDART <- paste(tempdir(), "/DART_archigrow", sep="")
dir.create(dirDART)

## Definition of the number of lie and tps files that will be saved inside dirDART
n <- 1

## Creation of lie and tps files inside dirDART
for (i in 1:n) {
write.table(ch7lie, file=paste(dirDART,"/ch7_",i,".lie", sep=""), row.names=FALSE)
write.table(ch7tps, file=paste(dirDART,"/ch7_",i,".tps", sep=""), row.names=FALSE)}

## Use of archigrow to plot the root system architecture for selected observation dates
## Use the same X and Y-axis scales to that used for the last observation date
ch7_output_1 <- archigrow(inputlie=dirDART, inputtps=dirDART, res=75, unitlength="cm",
unittime="day", coldyn=c("blue", "orange", "red"), numdate=c(15,30), finalscale=TRUE,
las=1, bty="l", asp=1, xaxp=c(0,30,3), lwd=2)

unlink(dirDART, recursive=TRUE)
```

architect *Computing Traits Describing The Global Root System Architecture*

Description

Calculation of common root system architecture (RSA) traits from Data Analysis of Root Tracings (DART) output files and RSA data encoded with the Root System Markup Language (RSML). More information can be found in *Delory et al (2015)*.

Usage

```
architect(inputrac=NULL, inputtps=NULL, inputrsml=NULL, res=NULL, unitlength="px",
          rsml.date=NULL, rsml.connect=FALSE, rootdiv=1)
```

Arguments

`inputrac` A character string specifying the path to the folder containing the rac files created by DART. This argument could be the same as `inputtps`/`inputrsml`.

`inputtps` A character string specifying the path to the folder containing the tps files created by DART. This argument could be the same as `inputrac`/`inputrsml`.

`inputrsml` A character string specifying the path to the folder containing the rsml files. This argument could be the same as `inputrac`/`inputtps`.

`res` **If images were acquired with a flatbed scanner**: a numeric value specifying the resolution of the images used to vectorize the root systems (resolution is expressed in dots/inch). **If images were acquired with a digital camera**: the ratio between the length of a reference object located on the image expressed in pixels and the actual length of the same object expressed in inches. This argument must be specified only if `unitlength="mm"` or `unitlength="cm"`.

`unitlength` A character string specifying the unit of length that must be used by the function to express the results in the final data frame. The value acceptable for this argument could be either “px” for pixels, “cm” for centimetres or “mm” for millimetres.

`rsml.date` A numeric value that must be specified when `is.null(inputrsml)=FALSE`. This argument specifies the root system age (the unit of time is defined by the user) and will be used only for RSA data encoded with the Root System Markup Language.

`rsml.connect` A logical value that must be specified when `is.null(inputrsml)=FALSE`. This argument will be used only for RSA data encoded with the Root System Markup Language. If `TRUE`, the lateral roots will be connected to their parent root if no connection information can be found in the rsml file(s) (see details). If `FALSE`, no connection will be made between daughter and mother roots if no connection information can be found in the rsml file(s).

`rootdiv` Either a numeric value or a vector of numeric values defining a class number or the class limits that must be used by the function to divide the first-order root in zones. Using this argument, `architect` will also compute the distribution of the secondary root number, length and density on the first-order root.

Details

Le Bot et al (2010) presented a free and open-access software (Data Analysis of Root Tracings - DART) allowing the analysis of complex RSA from captured images, particularly across time series. Using this software, a user has to manually identify roots as a set of links. After vectorization of a root system, a final dataset can be exported as a table file (rac file) containing several attributes for each individual root, notably the root length at each observation date. A second table file (tps file) containing the root system age at each observation date can also be exported from DART. These two datasets can finally be used to calculate derived RSA traits. In this context, `architect` is an R function allowing the calculation of common RSA traits at each observation date for each DART rac file: the total root system length, the length of the first-order root, the total number of lateral roots, the total lateral root length, the length and the number of lateral roots for each branching order, the mean lateral root length for each branching order, the density of secondary roots on the first-order root, the first-order and the lateral root growth rates, as well as the growth rate of the entire root system. When needed, the distribution of the secondary root number, length and density on the first-order root can also be calculated (via `rootdiv`).

The `architect` function will successively locate and read the rac and tps files created by DART and calculate RSA traits. First, `architect` will check if all arguments have been written in the suitable mode. If `res` is specified, the function will check if the numeric value required by this argument is positive. The function will also automatically check if the numeric elements in `rootdiv` are superior or equal to zero and sorted by increasing values. If an error occurs for one argument, the code stops and the corresponding error message will be returned by the function. Second, the function will use `inputrac` and `inputtps` to locate and read the raw rac and tps files created by DART, respectively. To run `architect` efficiently, DART files must have been saved with their appropriate extensions (.rac or .tps). Third, the function will create vectors and matrices for RSA traits calculation. If `inputtps` contains a single tps file, this will be used by `architect` for each rac file located in `inputrac`. If `inputtps` contains more than one tps file, the number of tps files in `inputtps` must be equal to the number of rac files in `inputrac` and corresponding rac and tps files must have the same name. If an error occurs at this step, a message will be returned by the function.

If RSA data have been encoded with the Root System Markup Language (*Lobet et al, 2015*), the path to the folder containing the rsml files can be specified using the `inputrsml` argument. To run `architect` efficiently, rsml files must have been saved with their appropriate extensions (.rsml). If `is.null(inputrsml)=FALSE`, `architect` will check if `rsml.date` and `rsml.connect` are specified and written in the suitable mode. After reading the rsml files located in `inputrsml`, the RSA data associated with one root system in each rsml file will be converted into a data frame possessing the same structure as the rac files created by DART. The `architect` function will then use these data frames to calculate RSA traits describing the global architecture of each root system. The argument `rsml.date` will be used to create a tps file for each root system located in `inputrsml`. As some rsml files do not provide connection information between daughter and mother roots, the `rsml.connect` argument can be used to connect the lateral roots to their corresponding mother root. If `rsml.connect=TRUE`, each point starting a lateral root will be connected to the nearest point located on its mother root. If `rsml.connect=FALSE` or `rsml.connect=TRUE`, the distance between a branching point and the parent root base (DBase) will be approximated by the distance between the point located on the mother root that is closest to the point starting a lateral root and the parent root base. Using rsml files, only RSA data associated with roots possessing a branching order lower or equal to 5 will be used for the computation of RSA traits.

Growth rates are calculated as follows:

- *First-order root growth rate*: for the first observation date, it is calculated as the ratio of the first-order root length to the root system age. For other observation dates (t), it is calculated as the difference between the first-order root length at time t and t-1 divided by the difference between the root system age at time t and t-1.
- *Lateral root growth rates*: for the first observation date, it is calculated as the ratio of the lateral root length to the root system age. For other observation dates (t), it is calculated as the difference between the lateral root length at time t and t-1 divided by the difference between the root system age at time t and t-1.
- *Growth rate of the root system*: for the first observation date, it is calculated as the ratio of the total root system length to the root system age. For other observation dates (t), it is calculated as the difference between the total root system length at time t and t-1 divided by the difference between the root system age at time t and t-1.

Value

Returns a data frame summarizing all RSA traits calculated for each rac/rsml file located in `inputrac`/`inputrsml`.

The data frame contains the following columns: **FileName** (the name of the rac/rsml files), **Time** (the root system age expressed in the same unit (=unittime) as in the tps file(s) (for rac files) or in the same unit as `rsml.date` (for rsml files)), **TRL** (the total root system length expressed in `unitlength`), **GRTR** (the growth rate of the root system expressed in `unitlength`/unittime), **L1R** (the first-order root length expressed in `unitlength`), **GR1R** (the growth rate of the first-order root expressed in `unitlength`/unittime), **TNLR** (the total number of lateral roots), **TLRL** (the total lateral root length expressed in `unitlength`), **NxLR** (the total number of lateral roots with a branching order equal to x), **LxLR** (the total length of lateral roots with a branching order equal to x expressed in `unitlength`), **MLxLR** (the mean length of lateral roots with a branching order equal to x expressed in `unitlength`), **GRxL** (the growth rate of lateral roots with a branching order equal to x expressed in `unitlength`/unittime), **D2LR** (the density of secondary roots on the first-order root expressed in root(s)/`unitlength`).

If `rootdiv` **is a numeric value > 1**, the following columns will be added to the data frame: **N2LR.LayerX** (the secondary root number in layer X), **L2LR.LayerX** (the total secondary root length in layer X expressed in `unitlength`), **D2LR.LayerX** (the secondary root density on the first-order root in the layer X expressed in root(s)/`unitlength`).

If `rootdiv` **is a vector of numeric values**, the following columns will be added to the data frame: **N2LR.XtoY** (the secondary root number in a layer delimited by X and Y on the first-order root), **L2LR.XtoY** (the total secondary root length in a layer delimited by X and Y on the first-order root expressed in `unitlength`), **D2LR.XtoY** (the secondary root density in a layer delimited by X and Y on the first-order root expressed in root(s)/`unitlength`).

Author(s)

Benjamin M. Delory, Caroline Baudson, Yves Brostaux, Guillaume Lobet, Loic Pages, Patrick du Jardin, Pierre Delaplace

References

Delory B.M., Baudson C., Brostaux Y., Lobet G., du Jardin P., Pages L., Delaplace P. (2015) archiDART: an R package for the automated computation of plant root architectural traits, *Plant and Soil*, DOI: 10.1007/s11104-015-2673-4.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

Lobet G., Pound M.P., Diener J., Pradal C., Draye X., Godin C., Javaux M., Leitner D., Meunier F., Nacry P., Pridmore T.P., Schnepf A. (2015) Root System Markup Language: Toward a Unified Root Architecture Description Language, *Plant Physiology*, DOI: 10.1104/pp.114.253625.

Examples

```
## Importation of example datasets
data(ch7rac)
data(ch7tps)

## Creation of a folder (DART_architect) inside a temporary directory to contain the rac and tps
## example datasets created by DART
dirDART <- paste(tempdir(), "/DART_architect", sep="")
dir.create(dirDART)

## Definition of the number of rac and tps files that will be saved inside dirDART
n <- 2
```

```
## Creation of rac and tps files inside dirDART
for (i in 1:n) {
write.table(ch7rac, file=paste(dirDART,"/ch7_",i,".rac", sep=""), row.names=FALSE)
write.table(ch7tps, file=paste(dirDART,"/ch7_",i,".tps", sep=""), row.names=FALSE)}

## Use of architect to compute RSA parameters from DART output files
## rootdiv=1
ch7_output_2 <- architect(inputrac=dirDART, inputtps=dirDART, res=75, unitlength="cm")

## Use of the rootdiv argument to compute the distribution of the secondary root number, length
## and density in the [0-15[, [15-30[, [30-45[ and [45-60[ first centimeters of the first-order
## root
ch7_output_3 <- architect(inputrac=dirDART, inputtps=dirDART, res=75, unitlength="cm",
rootdiv=c(0,15,30,45,60))

unlink(dirDART, recursive=TRUE)
```

ch7lie *LIE Data for the Vectorized Root System of a Young Oak Tree*

Description

This dataset obtained with Data Analysis of Root Tracings summarizes the attributes of each point used to construct the links constituting the interpreted root system of a young oak tree (*Quercus robur* L.) growing in a vertical rhizobox.

Usage

```
ch7lie
```

Format

A data frame containing the following columns:

- **Num**: the identification number of each point (i.e. the junctions between two consecutive links) used to construct the vectorized root system.
- **Date**: the identification number coding for the date at which a point was seen for the first time.
- **Bran**: the branching state ('true' denotes the start of a branching link while 'false' denotes the start of a continuing link).
- **Apic**: the apical state ('true' denotes a point ending a root while 'false' is related to a point starting a new link).
- **Prec**: the identification number of the previous point used to construct a link.
- **Suiv**: the identification number of the following point used to contruct a link.
- **X**: the X coordinate of each point used to construct the vectorized root system.
- **Y**: the Y coordinate of each point used to construct the vectorized root system.

Source

www6.paca.inra.fr/psh/Modeles-et-logiciels/Dart

References

Pages L., Serra V. (1994) Growth and branching of the taproot of young oak trees - a dynamic study, *Journal of Experimental Botany*, **45**, 1327–1334.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

---

ch7rac *RAC Data for the Vectorized Root System of a Young Oak Tree*

Description

For each observation date, this dataset obtained with Data Analysis of Root Tracings summarizes the length of each root constituting the vectorized root system of a young oak tree (*Quercus robur* L.) growing in a vertical rhizobox.

Usage

```
ch7rac
```

Format

A data frame containing the following columns:

- **Root**: the identification number of each root constituting the vectorized root system.
- **Mother**: the parent root identification number (a first-order root having no parent root, its value in this column is set to -1).
- **Ord**: the branching order (equal to 1 for the first-order root, 2 for a secondary root (or a first-order lateral root), 3 for a tertiary root (or a second-order lateral root), etc).
- **DBase**: the distance between the branching point to the parent root base.
- **DApp**: the date of emergence.
- **Lengths**: the root length at each observation date (in pixels). The number of column "Lengths" is equal to the number of observation dates.

Source

www6.paca.inra.fr/psh/Modeles-et-logiciels/Dart

References

Pages L., Serra V. (1994) Growth and branching of the taproot of young oak trees - a dynamic study, *Journal of Experimental Botany*, **45**, 1327–1334.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

ch7tps *TPS Data for the Vectorized Root System of a Young Oak Tree*

Description

This dataset obtained with Data Analysis of Root Tracings gives the correspondence between the observation numbers, dates and colour codes used to perform a root system architecture analysis of a young oak tree (*Quercus robur* L.) growing in a vertical rhizobox using DART.

Usage

```
ch7tps
```

Format

A data frame containing the following columns:

- **Num**: the identification number of each observation date.
- **Date**: the root system age at each observation date (the unit of time is defined by the experimenter). In ch7tps, data are expressed in days.
- **CoulR**: the red component of the colour chosen for the vectorization of the root system at each observation date.
- **CoulG**: the green component of the colour chosen for the vectorization of the root system at each observation date.
- **CoulB**: the blue component of the colour chosen for the vectorization of the root system at each observation date.

Source

www6.paca.inra.fr/psh/Modeles-et-logiciels/Dart

References

Pages L., Serra V. (1994) Growth and branching of the taproot of young oak trees - a dynamic study, *Journal of Experimental Botany*, **45**, 1327–1334.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

`latdist` *Computing Lateral Root Length and Density Distribution*

Description

Calculation of local lateral root lengths and densities, as well as distances between neighbouring lateral roots according to the distance between the branching point and the parent root base (DBase) using Data Analysis of Root Tracings (DART) output files and RSA data encoded with the Root System Markup Language (RSML). More information can be found in *Delory et al (2015)*.

Usage

```
latdist(inputrac=NULL, inputrsml=NULL, output=c("lrd","dtp"), res=NULL, unitlength="px",
        int.length=NULL, interpol=NULL, rsml.date=NULL, rsml.connect=FALSE)
```

Arguments

`inputrac` A character string specifying the path to the folder containing the rac files created by DART. This argument could be the same as `inputrsml`.

`inputrsml` A character string specifying the path to the folder containing the rsml files. This argument could be the same as `inputrac`.

`output` A character string specifying the type of output that must be returned by the function. Two values are acceptable for this argument: “lrd” and “dtp”. If `output="lrd"` (default value), the function will compute lateral root lengths and densities according to DBase for each mother root of each vectorized root system. If `output="dtp"`, the function will compute the distances between neighbouring lateral roots according to DBase for each mother root of each analysed root system (see details).

`res` **If images were acquired with a flatbed scanner**: a numeric value specifying the resolution of the images used to vectorize the root systems (resolution is expressed in dots/inch). **If images were acquired with a digital camera**: the ratio between the length of a reference object located on the image expressed in pixels and the actual length of the same object expressed in inches. This argument must be specified only if `unitlength="mm"` or `unitlength="cm"`.

`unitlength` A character string specifying the unit of length that must be used by the function to express the results in the final list. The value acceptable for this argument could be either “px” for pixels, “cm” for centimetres or “mm” for millimetres.

`int.length` A numeric value specifying the interval length (expressed in `unitlength`) that must be used by the function to calculate local lateral root lengths and densities. This value must be specified only when `output="lrd"` (see details).

`interpol` A numeric value specifying the number of points that must be used by the function on each mother root to calculate local lateral root lengths and densities. By default, `interpol=NULL` means that the function will calculate local lateral root lengths and densities at each branching point of each mother root. A numeric value can be specified when `output="lrd"` (see details).

`rsml.date` A numeric value that must be specified when `is.null(inputrsml)=FALSE`. This argument specifies the root system age (the unit of time is defined by the user) and will be used only for RSA data encoded with the Root System Markup Language.

`rsml.connect` A logical value that must be specified when `is.null(inputrsml)=FALSE`. This argument will be used only for RSA data encoded with the Root System Markup Language. If `TRUE`, the lateral roots will be connected to their parent root if no connection information can be found in the rsml file(s) (see details). If `FALSE`, no connection will be made between daughter and mother roots if no connection information can be found in the rsml file(s).

Details

Le Bot et al (2010) presented a free and open-access software (Data Analysis of Root Tracings - DART) allowing the analysis of complex root system architectures from captured images, particularly across time series. Using this software, a user has to manually identify roots as a set of links. After vectorization of a root system, a final dataset can be exported as a table file (rac file) containing several attributes for each individual root, notably the root length at each observation date and the distance between each branching point to the parent root base (DBase). This R function allows the calculation of either local lateral root lengths and densities or distances between neighbouring lateral roots according to DBase on each mother root constituting a vectorized root system.

The `latdist` function will successively locate and read the rac files created by DART and calculate RSA parameters. First, `latdist` will check if all arguments have been written in the suitable mode. If `res`, `interpol` and `int.length` are specified, the function will check if the numeric values required by these arguments are positive. If an error occurs for one argument, the code stops and the corresponding error message will be returned by the function. Second, the function will use `inputrac` to locate and read the raw rac files created by DART. To run `latdist` efficiently, the DART files must have been saved with their appropriate extensions (.rac). Third, depending on the `output` value, the function will compute RSA parameters. If `output="lrd"`, the function will use `interpol` to select the appropriate DBase values for which the calculation of RSA parameters should be performed. Then, a lateral root density and a total lateral root length will be calculated on intervals possessing a length equal to `int.length` and centred on each DBase value. If `output="dtp"`, only the distances between neighbouring lateral roots on each mother root will be computed by the function.

If RSA data have been encoded with the Root System Markup Language (*Lobet et al, 2015*), the path to the folder containing the rsml files can be specified using the `inputrsml` argument. To run `latdist` efficiently, rsml files must have been saved with their appropriate extensions (.rsml). If `is.null(inputrsml)=FALSE`, `latdist` will check if `rsml.date` and `rsml.connect` are specified and written in the suitable mode. After reading the rsml files located in `inputrsml`, the RSA data associated with one root system in each rsml file will be converted into a data frame possessing the same structure as the rac files created by DART. The `latdist` function will then use these data frames to compute lateral root length and density distribution. As some rsml files do not provide connection information between daughter and mother roots, the `rsml.connect` argument can be used to connect the lateral roots to their corresponding mother root. If `rsml.connect=TRUE`, each point starting a lateral root will be connected to the nearest point located on its mother root. If `rsml.connect=FALSE` or `rsml.connect=TRUE`, the distance between a branching point and the parent root base (DBase) will be approximated by the distance between the point located on the mother root that is closest to the point starting a lateral root and the parent root base. Using rsml files, only RSA data associated with roots possessing a branching order lower or equal to 5 will be used for the computation of RSA traits.

Value

Returns a list including:

`root` A list of data frames. Each element of the list is named as its corresponding rac/rsml file and contains the following columns: **Root** (the identification number of each root constituting the vectorized root system in the corresponding rac/rsml file), **Ord** (the branching order), **LatRootNum** (the number of lateral roots), **FinalRootLength** (the root length at the last observation date expressed in `unitlength`), **LatRootDensity** (the lateral root density calculated as LatRootNum/FinalRootLength and expressed in root(s)/`unitlength`).

`res` A list of secondary lists. Each element of the primary list (res) is named as its corresponding rac/rsml file. The secondary lists contain as much elements as roots constituting a vectorized root system. Each element of the secondary lists could be either a `NULL` value if a root do not have lateral roots or if no DBase value is comprised in [`int.length`/2, FinalRootLength-(`int.length`/2)], or a data frame made of two or three columns depending on the `output` value.

If `output="lrd"`: **DBase** (the distance between the branching point and the parent root base expressed in `unitlength`), **LRD** (Local lateral root density expressed in root(s)/`unitlength`), **LRL** (Local lateral root length expressed in `unitlength`/`unitlength`).

If `output="dtp"`: **DBase** (the distance between the branching point and the parent root base expressed in `unitlength`), **DTP** (the distance to the previous lateral root expressed in `unitlength`).

Author(s)

Benjamin M. Delory, Guillaume Lobet, Loic Pages

References

Delory B.M., Baudson C., Brostaux Y., Lobet G., du Jardin P., Pages L., Delaplace P. (2015) archiDART: an R package for the automated computation of plant root architectural traits, *Plant and Soil*, DOI: 10.1007/s11104-015-2673-4.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

Lobet G., Pound M.P., Diener J., Pradal C., Draye X., Godin C., Javaux M., Leitner D., Meunier F., Nacry P., Pridmore T.P., Schnepf A. (2015) Root System Markup Language: Toward a Unified Root Architecture Description Language, *Plant Physiology*, DOI: 10.1104/pp.114.253625.

Examples

```
## Importation of an example dataset
data(ch7rac)

## Creation of a folder (DART_latrootdensity) inside a temporary directory to contain the rac
## example datasets created by DART
dirDART <- paste(tempdir(), "/DART_latrootdensity", sep="")
dir.create(dirDART)

## Definition of the number of rac files that will be saved inside dirDART
n <- 1

## Creation of rac files inside dirDART
for (i in 1:n) {write.table(ch7rac, file=paste(dirDART,"/ch7_",i,".rac", sep=""), row.names=FALSE)}

## Use of latdist to compute RSA parameters from DART output files
## Plotting the results for the primary root
```

```
ch7_output_4 <- latdist(inputrac=dirDART, output="lrd", res=75, unitlength="cm",
int.length=5)
plot(ch7_output_4$res$ch7_1[[1]]$DBase, ch7_output_4$res$ch7_1[[1]]$LRD, pch=16,
xlab="DBase (cm)",ylab="Lateral root density (roots/cm)", main="LRD-interpol=NULL-int.length=5",
las=1, bty="l", xaxp=c(0,90,9))
plot(ch7_output_4$res$ch7_1[[1]]$DBase, ch7_output_4$res$ch7_1[[1]]$LRL, pch=16,
xlab="DBase (cm)",ylab="Lateral root length (cm/cm)", main="LRL-interpol=NULL-int.length=5",
las=1, bty="l", xaxp=c(0,90,9))

ch7_output_5 <- latdist(inputrac=dirDART, output="lrd", res=75, unitlength="cm",
int.length=5, interpol=1000)
plot(ch7_output_5$res$ch7_1[[1]]$DBase, ch7_output_5$res$ch7_1[[1]]$LRD, pch=16,
xlab="DBase (cm)", ylab="Lateral root density (roots/cm)", main="LRD-interpol=1000-int.length=5",
las=1, bty="l", xaxp=c(0,90,9))
plot(ch7_output_5$res$ch7_1[[1]]$DBase, ch7_output_5$res$ch7_1[[1]]$LRL, pch=16,
xlab="DBase (cm)", ylab="Lateral root length (cm/cm)", main="LRL-interpol=1000-int.length=5",
las=1, bty="l", xaxp=c(0,90,9))

ch7_output_6 <- latdist(inputrac=dirDART, output="dtp", res=75, unitlength="cm")
plot(ch7_output_6$res$ch7_1[[1]]$DBase, ch7_output_6$res$ch7_1[[1]]$DTP, pch=16,
xlab="DBase (cm)", ylab="Distance to the previous root (cm)", main="DTP", las=1, bty="l",
xaxp=c(0,90,9))

unlink(dirDART, recursive=TRUE)
```

trajectory *Computing Root Growth Directions and Trajectories*

Description

Calculation of root system architecture (RSA) parameters describing the directions and the trajectories of the root system growth using Data Analysis of Root Tracings (DART) output files and RSA data encoded with the Root System Markup Language (RSML). More information can be found in *Delory et al (2015).*

Usage

```
trajectory(inputrac=NULL, inputlie=NULL, inputtps=NULL, inputrsml=NULL, res=NULL,
        unitlength="px", unitangle="d", rotation=0, l.brangle, l.curv, l.tipangle,
          rsml.date=NULL)
```

Arguments

inputrac
: A character string specifying the path to the folder containing the rac files created by DART. This argument could be the same as `inputlie/inputtps/inputrsml`.

inputlie
: A character string specifying the path to the folder containing the lie files created by DART. This argument could be the same as `inputrac/inputtps/inputrsml`.

inputtps
: A character string specifying the path to the folder containing the tps files created by DART. This argument could be the same as `inputrac/inputlie/inputrsml`.

inputrsml
: A character string specifying the path to the folder containing the rsml files. This argument could be the same as `inputrac/inputlie/inputtps`.

`res` **If images were acquired with a flatbed scanner**: a numeric value specifying the resolution of the images used to vectorize the root systems (resolution is expressed in dots/inch). **If images were acquired with a digital camera**: the ratio between the length of a reference object located on the image expressed in pixels and the actual length of the same object expressed in inches. This argument must be specified only if `unitlength="mm"` or `unitlength="cm"`.

`unitlength` A character string specifying the unit of length that must be used by the function to express the results. The value acceptable for this argument could be either “px” for pixels, “cm” for centimetres or “mm” for millimetres.

`unitangle` A character string specifying the unit that must be used by the function to express the calculated angles. The value acceptable for this argument could be either “d” for degrees (default value) or “r” for radians.

`rotation` A numeric value specifying the angle (expressed in `unitangle`) that must be used by the function to perform a clockwise rotation of the root system (see details).

`l.brangle` A numeric value specifying the minimum root length that must be used by the function to calculate the branching angle of a daughter root on its corresponding mother root (see details).

`l.curv` A numeric value specifying the distance that must be used by the function to place equidistantly spaced interpolated points along each root before calculating the angles between the direction vectors of the successive links constituting a vectorized root (see details).

`l.tipangle` A numeric value specifying the distance between the root tip and an interpolated point located along a root. Each interpolated point will be used by the function to calculate a direction vector following the root growth direction near the apex prior to the calculation of the root tip angle relative to a vertical direction vector at each observation date.

`rsml.date` A numeric value that must be specified when `is.null(inputrsml)=FALSE`. This argument specifies the root system age (the unit of time is defined by the user) and will be used only for RSA data encoded with the Root System Markup Language.

Details

Le Bot et al (2010) presented a free and open-access software (Data Analysis of Root Tracings - DART) allowing the analysis of complex RSA from captured images, particularly across time series. Using this software, a user has to manually identify roots as a set of links. After vectorization of a root system, three final datasets can be exported from DART as table files: (1) a rac file containing several attributes for each individual root, notably the root length at each observation date, (2) a lie file containing several attributes for each point used to construct the links constituting the vectorized root system, and (3) a tps file containing the root system age at each observation date. This R function allows the calculation of the branching angle of each daughter root on its corresponding mother root, statistical parameters describing the curvature of each root constituting a vectorized root system, and the root tip angles relative to a vertical direction vector at each observation date.

The `trajectory` function will successively locate and read the rac, lie and tps files created by DART and calculate RSA traits. First, `trajectory` will check if all arguments have been written in the suitable mode. If `res`, `rotation`, `l.brangle`, `l.curv` and `l.tipangle` are specified, the function will check if the numeric values required by these arguments are positive. If an error occurs for one argument, the code stops and the corresponding error message will be returned by the function. Second, the function will use `inputrac`, `inputlie` and `inputtps` to locate and read the raw rac, lie and tps files created by DART, respectively. To run `trajectory` efficiently, the DART files must have been saved with their appropriate extensions (.rac, .lie or .tps). If `inputtps` contains a single tps file, it will be used by `trajectory` for each lie/rac file located in `inputlie`/`inputrac`. Moreover, the number of lie files in `inputlie` must be equal to the number of rac files in `inputrac` and corresponding rac and lie files must have the same name. If `inputps` contains more than one tps file, the number of tps files in `inputtps` must be equal to the number of rac/lie files in `inputrac`/`inputlie` and corresponding rac/lie and tps files must have the same name. If an error occurs at this step, a message will be returned by the function. Third, the function will compute RSA traits.

If RSA data have been encoded with the Root System Markup Language (*Lobet et al, 2015*), the path to the folder containing the rsml files can be specified using the `inputrsml` argument. To run `trajectory` efficiently, rsml files must have been saved with their appropriate extensions (.rsml). If `is.null(inputrsml)=FALSE`, `trajectory` will check if `rsml.date` is specified and written in the suitable mode. After reading the rsml files located in `inputrsml`, the RSA data associated with one root system in each rsml file will be converted into two data frames possessing the same structure as the rac and lie files created by DART, respectively. The `trajectory` function will then use these data frames to compute RSA traits describing root growth directions and trajectories. The argument `rsml.date` will be used to create a tps file for each root system located in `inputrsml`. By default, if no connection information between daughter and mother roots exists in a rsml file, the function will connect the lateral roots with their correponding mother root. To do so, each point starting a lateral root will be connected to the nearest point located on its mother root. The distance between a branching point and the parent root base (DBase) will be approximated by the distance between the point located on the mother root that is closest to the point starting a lateral root and the parent root base. Using rsml files, only RSA data associated with roots possessing a branching order lower or equal to 5 will be used for the computation of RSA traits.

Due to technical limitations, it is sometimes easier to take a picture of a root system if it is rotated before image acquisition. In this case, the vectorized root system will depict a rotation compared with the natural plant root system. To make a correction, one can use `rotation` to specify an angle value expressed in `unitangle` that will be used by the function to rotate the vectorized root system clockwise before RSA parameters calculation.

Hereafter, we will consider that the normal vector that is orthogonal to a direction vector (a,b) will have the following coordinates: (b,-a).

A daughter root will be considered to grow at the left side of the mother root if the scalar product between a direction vector (=vector A) going from the branching point (X0) to the following point (Xd) on the daughter root and a vector (=vector B) normal to a direction vector (=vector C) going from the branching point (X0) to the following point on the mother root (Xm) is positive. If the scalar product between A and B is negative, the daughter root will be considered to grow at the right side of the mother root. If the scalar product between A and B is equal to zero, the calculation of the scalar product between A and B will be performed again using a vector A going from the branching point (X0) to a point located on the daughter root at Xd<-Xd+1 until the calculated scalar product is different from zero. If A is always normal to B, a random lateral root growth orientation (either left or right) will be defined.

The branching angle of a daughter root on its mother root is approximated by the angle between two direction vectors going from the branching point to an interpolated point located at a distance `l.brangle` from the branching point on the mother or on the daughter root, respectively. The distance `l.brangle` between a branching point and an interpolated point is measured along the mother and the daughter roots. A first-order root having no mother root, a `NA` value will be returned for Branching.Angle. If a root has a final length lower than `l.brangle`, no branching angle can be calculated and the function will return a `NA` value for Branching.Angle.

The curvature of each root is evaluated by the mean and the standard deviation of the local angles formed by the direction vectors of the successive links constituting a vectorized root. First, the function will use `l.curv` to interpolate equidistantly spaced points along each root constituting a vectorized root system. Second, the angles between the direction vectors of the successive links of each root are calculated. Then, the mean and the standard deviation of the calculated angles are determined for each root. If a root has a final length lower than 2*`l.curv`, the function will return a `NA` value for Mean.Curv and SD.Curv. If a root has a final length lower than 3*`l.curv`, the function will return a `NA` value for SD.Curv as no standard deviation can be calculated on a single angle value.

Value

Returns a list including:

`root` A list of data frames. Each element of the list is named as its corresponding rac/lie/rsml file and contains the following columns: **Root** (the identification number of each root constituting the vectorized root system), **Mother** (the parent root identification number), **Ord** (the branching order), **DBase** (the distance between the branching point to the parent root base expressed in `unitlength`), **DApp** (the date of emergence expressed in the same unit as the root system age in the tps file(s) (for rac/lie files) or in rsml.date (for rsml files)), **FinalRootLength** (the root length at the last observation date expressed in `unitlength`), **Tortuosity** (the ratio of the final root length to the Euclidean distance between the branching point and the apex of the root), **Orientation** (the direction of the lateral root growth), **Branching.Angle** (the branching angle of a daughter root on its mother root expressed in `unitangle`), **Mean.Curv** (the mean of the local angle values calculated between the direction vectors of the successive links constructed using equidistantly spaced interpolated points along each root of a vectorized root system; the calculated mean is expressed in `unitangle`), **SD.Curv** (the standard deviation of the local angle values calculated between the direction vectors of the successive links constructed using equidistantly spaced interpolated points along each root of a vectorized root system; the calculated standard deviation is expressed in `unitangle`.

`tip` A list of data frames. Each element of the list is named as its corresponding rac/lie/rsml file and contains the following columns: **Root** (the identification number of each root constituting the vectorized root system), **Ang.DateX** (the calculated root tip angle relative to a vertical direction vector at the observation date X expressed in `unitangle`).

Author(s)

Benjamin M. Delory, Guillaume Lobet, Loic Pages

References

Delory B.M., Baudson C., Brostaux Y., Lobet G., du Jardin P., Pages L., Delaplace P. (2015) archiDART: an R package for the automated computation of plant root architectural traits, *Plant and Soil*, DOI: 10.1007/s11104-015-2673-4.

Le Bot J., Serra V., Fabre J., Draye X., Adamowicz S., Pages L. (2010) DART: a software to analyse root system architecture and development from captured images, *Plant and Soil*, DOI: 10.1007/s11104-009-0005-2.

Lobet G., Pound M.P., Diener J., Pradal C., Draye X., Godin C., Javaux M., Leitner D., Meunier F., Nacry P., Pridmore T.P., Schnepf A. (2015) Root System Markup Language: Toward a Unified Root Architecture Description Language, *Plant Physiology*, DOI: 10.1104/pp.114.253625.

Examples

```
## Importation of example datasets
data(ch7rac)
data(ch7lie)
data(ch7tps)

## Creation of a folder (DART_trajectory) inside a temporary directory to contain the rac and lie
## example datasets created by DART
dirDART <- paste(tempdir(), "/DART_trajectory", sep="")
dir.create(dirDART)

## Definition of the number of rac, tps and lie files that will be saved inside dirDART
n <- 1

## Creation of rac, tps and lie files inside dirDART
for (i in 1:n) {
write.table(ch7rac, file=paste(dirDART,"/ch7_",i,".rac", sep=""), row.names=FALSE)
write.table(ch7lie, file=paste(dirDART,"/ch7_",i,".lie", sep=""), row.names=FALSE)
write.table(ch7tps, file=paste(dirDART,"/ch7_",i,".tps", sep=""), row.names=FALSE)}

## Use of trajectory to compute RSA parameters from DART output files
ch7_output_7 <- trajectory(inputrac=dirDART, inputlie=dirDART, inputtps=dirDART, res=75,
unitlength="cm", l.brangle=1, l.curv=1, l.tipangle=0.5)

## Plotting the results for ch7_1
## Distribution of the calculated insertion angles
hist(ch7_output_7$root$ch7_1$Branching.Angle, breaks=seq(from=0, to=180, by=5),
main="ch7_1-Branching angle distribution", xlab="Branching angle (d)", las=1,
xaxp=c(0,150,15), xlim=c(0,150))

## Evolution of the root tip angle for the primary root (black) and the two longest lateral roots
## (green and red)
date<-c(1:31)
plot(date, ch7_output_7$tip$ch7_1[1,2:ncol(ch7_output_7$tip$ch7_1)], type="l", lwd=2, bty="l",
las=1, ylim=c(0,180), ylab="Root tip angle (d)", xlab="Time (Num)", col="black",
main="Root tip angle", yaxp=c(0,180,18))
lines(date, ch7_output_7$tip$ch7_1[206,2:ncol(ch7_output_7$tip$ch7_1)], lwd=2, col="green")
lines(date, ch7_output_7$tip$ch7_1[221,2:ncol(ch7_output_7$tip$ch7_1)], lwd=2, col="red")
abline(h=90)

unlink(dirDART, recursive=TRUE)
```

Index

∗Topic **datasets**
    ch7lie, 12
    ch7rac, 13
    ch7tps, 14

archidraw, 2
archigrow, 5
architect, 8

ch7lie, 12
ch7rac, 13
ch7tps, 14

latdist, 15

par, *4*, *7*, *8*

trajectory, 18