

# Package ‘aRchi’

April 16, 2021

**Type** Package

**Title** Quantitative Structural Model ('QSM') Treatment for Tree Architecture

**Version** 1.0.5

**Author** Olivier Martin

**Maintainer** Olivier Martin <oli.martin@entymail.com>

**Description** Provides a set of tools to manipulate, visualize and compute metrics from quantitative structural model of trees (i.e the so-called 'QSM') . It can be used in various context of forest ecology (i.e biomass estimation) and tree architecture (i.e architectural metrics), see Martin-Ducup et al. (2020) <doi:10.1111/1365-2435.13678>. The package is based on a new S4 class called 'aRchi'.

**License** CeCILL

**Encoding** UTF-8

**URL** <https://github.com/umr-amap/aRchi>

**RoxygenNote** 7.1.1

**Imports** data.table, dplyr, plyr, rgl, methods, lidR, FNN, DiceKriging, circular, pkgcond, stringr, utils, stats, progress, gtools

**Collate** 'nullOrnumeric.R' 'nullOrlist.R' 'nullOrLASOrDatatable.R' 'nullOrDatatable.R' 'aRchiClass.R' 'BranchAngle.R' 'Compute\_A0.R' 'Compute\_Mf.R' 'DAL.R' 'ForkRate.R' 'LeonardoRatio.R' 'Make\_Node.R' 'Make\_Path.R' 'PathFraction.R' 'SelectinQSM\_3d.R' 'TreeBiomass.R' 'Truncate\_QSM.R' 'WBEparameters.R' 'WoodSurface.R' 'aRchi.R' 'add\_pointcloud.R' 'angle3d.R' 'build\_aRchi.R' 'clean\_QSM.R' 'get\_QSM.R' 'get\_nodes.R' 'get\_operations.R' 'get\_paths.R' 'get\_pointcloud.R' 'plot.R' 'read\_QSM.R' 'read\_aRchi.R' 'write\_aRchi.R'

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2021-04-16 15:50:09 UTC

**R topics documented:**

add_pointcloud	2
angle3d	3
aRchi-class	4
BranchAngle	4
build_aRchi	5
Clean_QSM	6
Compute_A0	7
Compute_Mf	8
DAI	9
ForkRate	10
get_nodes	11
get_operations	12
get_paths	13
get_pointcloud	14
get_QSM	14
LeonardoRatio	15
Make_Node	16
Make_Path	18
nullOrDatatable-class	19
nullOrLASOrDatatable-class	19
nullOrlist-class	19
nullOrnumeric-class	20
PathFraction	20
plot,aRchi,ANY-method	21
read_aRchi	22
read_QSM	23
SelectinQSM_3d	24
TreeBiomass	25
Truncate_QSM	26
WBEparameters	27
WoodSurface	28
write_aRchi	29
<b>Index</b>	<b>30</b>

---

add_pointcloud	<i>Add a point cloud to an object of class aRchi</i>
----------------	--

---

**Description**

Add a point cloud to an object of class aRchi

**Usage**

```
add_pointcloud(aRchi, point_cloud)

## S4 method for signature 'aRchi'
add_pointcloud(aRchi, point_cloud)
```

**Arguments**

aRchi	The object of class aRchi
point_cloud	The point cloud. Either a las or a data.frame

---

angle3d	<i>Calculate the zenith angle from xyz coordinates</i>
---------	--

---

**Description**

Calculate a zenith angle between two segments from 3d (i.e x,y,z) coordinates

**Usage**

```
angle3d(o, a, b)
```

**Arguments**

o	3d coordinates of the common point of the two segments
a	3d coordinates of the other point of segment a
b	3d coordinates of the other point of segment b

**Value**

The angle in degree

**Examples**

```
origin=c(0,0,0)
a=c(0,0,1)
b=c(1,0,0)

angle3d(o=origin,a=a,b=b)
```

---

aRchi-class	<i>aRchi</i>
-------------	--------------

---

### Description

Class containing files to compute and display in three dimensions tree architectural metrics at different level of organization

### Value

An object of class aRchi

### Fields

QSM a data.table containing QSM information according to read\_QSM function format  
 pointcloud a data.table containing the point cloud used to generated the QSM  
 Paths a data.table of Paths according to Make\_Path function (see [Make\\_Path](#))  
 Nodes Metrics computed at the node scale (see [Make\\_Node](#))  
 operations Record all the operations realized on the object.

---

BranchAngle	<i>Estimation of the tree branch angle from an aRchi file</i>
-------------	---

---

### Description

Estimate the branch angle of a QSM. Two methods are possible (see method argument)

### Usage

```
BranchAngle(aRchi, method = NULL, A0 = FALSE, level = "Tree")
```

```
## S4 method for signature 'aRchi'
```

```
BranchAngle(aRchi, method = NULL, A0 = FALSE, level = "Tree")
```

### Arguments

aRchi	an object of class aRchi with at least a QSM and a path table
method	character. SegmentAngle or codeKing98
A0	logical (default = FALSE). If TRUE the main axis to remove from the calculation is re-estimated using the <a href="#">Compute_A0</a> function. If false the default branch order 0 is kept.
level	character. The level at which the branch angle is computed. Tree for tree level; branching_order for branch order level; Axis one angle value per axis.

## Details

The method "SegmentAngle" compute the angle by considering the first and the last cylinder of each segment, mean is then used for the level of organization selected.

The method "King98" compute the angle by considering the first and the last cylinder of each axis mean is then used for the level of organization selected.

The main axis is always removed.

## Value

a numeric or data.table. The branch angle in degree at the selected level. with 0 a perfectly vertical branch angle, 90 a perfectly horizontal branch angle and >90 a downward branch angle

## References

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

## Examples

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Compute the branch angle at various level
BranchAngle(Tree1_aRchi,method="SegmentAngle")
BranchAngle(Tree1_aRchi,level="branching_order",method="SegmentAngle",A0=TRUE)
```

---

build_aRchi	<i>Build a an object of class aRchi</i>
-------------	---

---

## Description

Build an object of class aRchi

## Usage

```
build_aRchi(QSM, point_cloud, keep_original = FALSE)
```

## Arguments

QSM	A data.table obtained from <a href="#">read_QSM</a> function
point_cloud	A point cloud. Either a LAS or a data.table with at least three columns with 3d coordinates (i.e X,Y,Z)
keep_original	logical (Default = FALSE). Should the original branching order and axis be kept ? Otherwise, it is re-estimated.

**See Also**

[aRchi](#); [write\\_aRchi](#); [read\\_aRchi](#)

**Examples**

```
file_QSM=system.file("extdata","Tree_1_TreeQSM.txt",package = "aRchi")
file_pc=system.file("extdata","Tree_1_point_cloud.las",package = "aRchi")
QSM=read_QSM(file_QSM,model="treeQSM")
pc=lidR::readLAS(file_pc)
# Make an object of class aRchi
Tree1_aRchi=build_aRchi(QSM=QSM,point_cloud=pc)
```

---

Clean\_QSM

*Cleans a QSM*

---

**Description**

Cleans the QSM in an object of class `aRchi` by removing branches that have a disproportionate lower radius than their siblings.

**Usage**

```
Clean_QSM(aRchi, threshold = NULL, plotresult = FALSE)
```

```
## S4 method for signature 'aRchi'
Clean_QSM(aRchi, threshold = NULL, plotresult = FALSE)
```

**Arguments**

<code>aRchi</code>	an object of class <code>aRchi</code> with at least a QSM and a Paths table.
<code>threshold</code>	numeric. The proportion of the largest daughter diameter (between 0 and 1) under which a branch is removed.
<code>plotresult</code>	logical (default = FALSE). Show the results in a 3d plot if TRUE

**Details**

This cleaning is done by browsing the tree QSM from the base to the top. Each time a ramification point is encountered a daughter branch is removed if its radius is lower than a selected (i.e threshold) proportion of radius of the largest daughter. This allows removing small branches on large branches that can be for example traumatic or epicormic shoots or false branches due to noise in QSM. In [ForkRate](#) function the same approach is used with a threshold of 75% (i.e 0.75) to count the number of fork and compute the fork rate.

**Value**

An object of class `aRchi` with the cleaned QSM.

**See Also**

[ForkRate](#) to compute the fork rate; [Truncate\\_QSM](#) to truncate a QSM at a specific diameter threshold

**Examples**

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Clean the QSM: threshold of 0.5
Cleaned_Tree1_aRchi=Clean_QSM(Tree1_aRchi,threshold = 0.5,plotresult = TRUE)
# show the cleaned QSM data.table
get_QSM(Cleaned_Tree1_aRchi)
```

---

 Compute\_A0

*Find the principal axis of a tree*


---

**Description**

Find the principal axis of a tree (i.e A0) and add a column to the QSM of an aRchi object. This is alternative method to the default branch order proposed in a QSM for the principal axis only.

**Usage**

```
Compute_A0(aRchi, plotresult = FALSE)
```

```
## S4 method for signature 'aRchi'
Compute_A0(aRchi, plotresult = FALSE)
```

**Arguments**

**aRchi**            an object of class aRchi with at least a QSM and the Paths table  
**plotresult**       logical (default = FALSE). Show the results in a 3d plot if TRUE

**Details**

The method used to find the principal axis consist in finding the highest vertical path with consecutive segments of similar diameters and orientations. An index called A0 of the probability of being the principal axis is thus computed for each path of the tree and the path with the highest value is considered as the principal axis (see Martin-Ducup et al. 2020 for more information).

A0 ranges between 0 and 4 with 0 indicating a path with a low probability of being the principal axis and 4 indicating a high probability of being the principal axis, i.e. the highest vertical path with consecutive segments of similar diameters and orientations. The path with the maximum A0 value was selected as the principal axis.

The new column A0 of the QSM slot take the value 2 if the cylinder is part of the principal axis or 1 if not.

**Value**

The aRchi object with the QSM having a new column A0.

**References**

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

**See Also**

[DAI](#) to compute the dominance of a principal axis index that uses the A0 index.

**Examples**

```
# Read an aRchi file with at least the QSM and the paths table
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

Tree1_aRchi=Compute_A0(Tree1_aRchi,plotresult=TRUE)
```

---

Compute\_Mf

*Compute Moment of force*

---

**Description**

Compute the moment of gravity force Mf and the moment of gravity force relative to cylinder radius Mf\_r from an object of class aRchi.

**Usage**

```
Compute_Mf(aRchi, WoodDensity)

## S4 method for signature 'aRchi'
Compute_Mf(aRchi, WoodDensity = NULL)
```

**Arguments**

aRchi            an object of class aRchi with at least the QSM and the paths table.

WoodDensity    a numeric or a data.table. A single wood density value for the whole tree or one value per cylinder in kg/m3. If wood density is given for each cylinder a data.table with two column (i.e cyl\_ID and WoodDensity) must be given.



### Details

The moment of gravity force (i.e  $M_f$ ) is calculated at each cylinder position.  $M_f$  can be seen as a proxy of the mechanical loading history due to gravity at a given position of a tree. This quantity is defined by the following equation:  $M_f = R * FALSE$  where  $R$  is the lever arm, which is the norm of the horizontal vector between the position where  $M_f$  is measured (i.e a cylinder) and the position where the force is applied (i.e., the center of mass,  $G$ , of the whole structure upstream a cylinder: a subtree).

The mass of the cylinders are needed to calculate the center of mass and are estimated using their volume and the wood density provided in argument `WoodDensity`. Finally, `FALSE` is the weight of the subtree:  $FALSE = g * M$  with  $g$  the standard acceleration due to gravity ( $9.81 \text{ m.s}^{-2}$ ).

The moment of gravity force relative (i.e  $M_{f\_r}$ ) to cylinder radius  $r$  is also computed following the formula:  $M_{f\_r} = M_f / r^3$

### Value

The `aRchi` file with the QSM slot having three new columns: the biomass upstream the cylinder `sub_tree_biomass`, the moment of gravity force  $M_f$  and the moment of gravity force relative to cylinder radius  $M_{f\_r}$ .

### Examples

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# Compute the moment of force for each cylinder
Tree1_aRchi=Compute_Mf(Tree1_aRchi,WoodDensity=550)

# show the QSM data.table with the three new columns sub_tree_biomass, MF and Mf_r)
get_QSM(Tree1_aRchi)
```

---

 DAI

*Estimate the index of dominance of the principal axis: DAI.*

---

### Description

Estimate the index of dominance of the principal axis (DAI) from an `aRchi` object.

### Usage

```
DAI(aRchi)

## S4 method for signature 'aRchi'
DAI(aRchi)
```

**Arguments**

aRchi                    an object of class aRchi with at least the QSM and the Paths table

**Details**

The idea of DAI is to disentangle architectures of trees with a strong apical dominance, i.e. with a central main stem growing more strongly than other side axes (such as in most conifers, for instance), from those having a spread out branching pattern with similar axes and no obvious main stem.

The higher the index the more dominant the principal axis. DAI is computed based on the indices A0 which is an index of the probability of being the principal axis for each path of the tree (see function [Compute\\_A0](#)). More information are given in Martin-Ducup et al 2020

**Value**

Numeric. The value of DAI.

**References**

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

**See Also**

[Compute\\_A0](#) to identify the principal axis (i.e  $\max(A_0)$ ).

**Examples**

```
# Read an aRchi file with at least the QSM and the paths table
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

DAI(Tree1_aRchi)
```

---

ForkRate

*Compute the fork rate of a tree*

---

**Description**

Compute the fork rate from an aRchi object

**Usage**

```
ForkRate(aRchi)

## S4 method for signature 'aRchi'
ForkRate(aRchi)
```

**Arguments**

aRchi a file of class aRchi with at least a QSM and a Path table.

**Details**

The fork rate is the mean number of forks per meter of tree height. This metric is computed by browsing tree QSM from the base to the top. Each time a ramification point is encountered it is evaluated as a fork if at least one daughter had a radius not less than 75% of the diameter of the largest daughter. This threshold was chosen in order to exclude non-perennial structures, such as traumatic or epicormic shoots, or branches that will not last on the tree. If the ramification point is a fork, all retained daughter branches are browsed through until the next ramification point and further until the path end. If a daughter is rejected, it is removed as well as all the paths passing through it.

**Value**

a vector with two numeric value. The number of Fork and the fork rate.

**References**

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

**See Also**

[Clean\\_QSM](#) to clean a QSM based on a threshold of percentage of largest daughter's diameter.

**Examples**

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Compute the fork rate of Tree1
ForkRate(Tree1_aRchi)
```

---

get\_nodes

*Get the nodes from an object of class aRchi*

---

**Description**

Get the nodes from an object of class aRchi

**Usage**

```
get_nodes(aRchi)

## S4 method for signature 'aRchi'
get_nodes(aRchi)
```

**Arguments**

aRchi            The object of class aRchi

**See Also**

[get\\_QSM](#); [get\\_pointcloud](#); [get\\_paths](#)

**Examples**

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# get the nodes (a list of two data.table)
get_nodes(Tree1_aRchi)
```

---

get\_operations

*Get the nodes from an object of class aRchi*

---

**Description**

Show a list with all the operations (and their parameters) that have been performed on an object of class aRchi

**Usage**

```
get_operations(aRchi)

## S4 method for signature 'aRchi'
get_operations(aRchi)
```

**Arguments**

aRchi            The object of class aRchi

**See Also**

[get\\_QSM](#); [get\\_pointcloud](#); [get\\_paths](#)

**Examples**

```
## Not run:
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# Making some operations
Tree1_aRchi<-Make_Path(Tree1_aRchi)
```

```
Tree1_aRchi=Compute_A0(Tree1_aRchi)
Tree1_aRchi=Compute_Mf(Tree1_aRchi,WoodDensity = 550)
Tree1_aRchi=Clean_QSM(Tree1_aRchi,threshold = 0.5)

# Show the oprations and their parameters
get_operations(Tree1_aRchi)

## End(Not run)
```

---

get\_paths

*Get the paths from an object of class aRchi*

---

## Description

Get the paths from an object of class aRchi

## Usage

```
get_paths(aRchi)

## S4 method for signature 'aRchi'
get_paths(aRchi)
```

## Arguments

aRchi            The object of class aRchi

## See Also

[get\\_QSM](#); [get\\_pointcloud](#); [get\\_nodes](#);

## Examples

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# get the paths (a data.table)
get_paths(Tree1_aRchi)
```

---

get\_pointcloud            *Get the pointcloud from an object of class aRchi*

---

**Description**

Get the pointcloud from an object of class aRchi

**Usage**

```
get_pointcloud(aRchi)

## S4 method for signature 'aRchi'
get_pointcloud(aRchi)
```

**Arguments**

aRchi            The object of class aRchi

**See Also**

[get\\_QSM](#); [get\\_paths](#); [get\\_nodes](#)

**Examples**

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# get the pointcloud
get_pointcloud(Tree1_aRchi)
```

---

get\_QSM            *Get the QSM from an object of class aRchi*

---

**Description**

Get the QSM from an object of class aRchi

**Usage**

```
get_QSM(aRchi)

## S4 method for signature 'aRchi'
get_QSM(aRchi)
```

**Arguments**

aRchi            The object of class aRchi

**See Also**

[get\\_pointcloud](#); [get\\_paths](#); [get\\_nodes](#)

**Examples**

```
# Read an aRchi file with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

# show the QSM data.table
get_QSM(Tree1_aRchi)
```

---

LeonardoRatio	<i>Compute Leonardo's ratio</i>
---------------	---------------------------------

---

**Description**

Compute from an object of class aRchi the Leonardo's ratio (i.e R\_ratio) at node, axis, branch order or tree level.

**Usage**

```
LeonardoRatio(aRchi, level = "Tree", position = 10)
```

```
## S4 method for signature 'aRchi'
LeonardoRatio(aRchi, level = "Tree", position = 10)
```

**Arguments**

aRchi            an object of class aRchi with at least a QSM and the Nodes table (see function [Make\\_Node](#))

level            characters. At which level R\_ratio has to be estimated. Node for node level, Axis for the axis level, branching\_orde for branch order level and Tree for tree level (default).

position        At which position from the node R\_ratio had to be estimated. Either a numeric or a character. Use a numeric multiple of ten to select the distance from the node in cm where R ratio has to be estimated (e.g 10 for 10cm from the node). Use the % sign after a multiple of ten to select the distance from the node in percentage of the length of the parent and daughters segments (e.g 50% for an estimation at mid-length of the segments). Note that 0 is accepted and correspond to the closest position from the node.

**Details**

Details for Leonardo Da Vinci's ratio calculation are given in the details part of function [Make\\_Node](#).

**Value**

Data.table of the summary of R\_ratio for the selected level.

**See Also**

[Make\\_Node](#) for node metrics estimation; [WBEparameters](#) to estimates WBE parameters at different level;

**Examples**

```
# Read an aRchifile with a QSM and node tables.
file=system.file("extdata","Tree1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Leonardo'ratio at the branching order level estimated at midlength of the segments
LeonardoRatio(Tree1_aRchi,level="Tree", position="50%")
# Leonardo'ratio at the node level estimated at 30 cm from the node
LeonardoRatio(Tree1_aRchi,level="Node", position=30)
```

---

Make\_Node

*Make Node*

---

**Description**

Compute several node metrics (i.e Leonardo's ratio and WBE parameters) from an object aRchi at different distance from the node.

**Usage**

```
Make_Node(aRchi, all_combination = FALSE)
```

```
## S4 method for signature 'aRchi'
Make_Node(aRchi, all_combination = FALSE)
```

**Arguments**

aRchi            a file of class aRchi containing at least a QSM

all\_combination    logical (default = FALSE). Should the node metrics be computed at each combination of distance from the node ? (see details).



## Details

The Nodes slot contains a list of two data.table (Absolute\_positions and Relative\_positions). Each data.tables contains for each node several values of Leonardo's rule ratio (R\_ratio), radius scaling exponent (alpha), length scaling exponent (beta) and the estimated metabolic rate (theta) which are parameters of the West Brown and Enquist metabolic theory (see Bentley et al. 2013, Lau et al 2019 and Martin-Ducup et al. 2020).

For a given node, each R\_ratio and alpha correspond to its value estimated with the branch radius (for alpha) or the cross section area (for R\_ratio) at a given position from the node for the parent (pos\_parent) and for the daughters (pos\_daughters). The positions (i.e pos\_parent and pos\_daughters) are the distances in meters (for Absolute\_position data.table) or in percentage (for Relative\_position data.table) from the node (i.e the ramification point) to the point of radius or cross section area estimation. These positions are given at a 10 cm step and cannot be higher than the length of the shorter segment among the couple daughters/parent for Absolute\_position and at a 10% step for the Relative\_position.

Kriging models are used to estimate radius (and thus cross section area) along the segment positions. For example pos\_parent = 0.2 and pos\_daughters = 0.2 for Absolute\_position data.table means that the parameters (i.e R\_ratio and alpha) have been estimated at 20cm from the node position for the parent and 20 cm from the node position for the daughters. For the Relative\_position data.table, this position is given in proportion to the total length of the segment. For example, pos\_parent = 0.5 and pos\_daughter = 0.5 means that the parameters are estimated at mid\_length of the segments for both the parent and the daughters).

beta values are repeated for each node as it depend on segment length only and not on radius position.

If all\_combination = TRUE all the possible combination for pos\_parent and pos\_daughter are computed (e.g pos\_parent = 0.3 and pos\_daughter = 0.5) but note that this processing might take several minutes and is FALSE by default.

## Value

The aRchi file with the Nodes slot filled.

## References

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

Lau, A. et al. Estimating architecture-based metabolic scaling exponents of tropical trees using terrestrial LiDAR and 3D modelling. *Forest Ecology and Management* 439, 132–145 (2019).

Bentley, L. P. et al. An empirical assessment of tree branching networks and implications for plant allometric scaling models. *Ecology Letters* 16, 1069–1078 (2013).

## See Also

[WBEparameters](#) to estimates WBE parameters at different level; [LeonardoRatio](#) to estimates Leonardo Da Vinci's ratio at different level.

**Examples**

```
# Read a QSM file
file=system.file("extdata","Tree_1_TreeQSM.txt",package = "aRchi")
QSM=read_QSM(file,model="treeQSM")
# Build an object of class aRchi
Tree1_aRchi<-build_aRchi(QSM=QSM)
Tree1_aRchi
# Make the node table
Tree1_aRchi<-Make_Node(Tree1_aRchi)
Tree1_aRchi
# WBE parameters at the tree level
LeonardoRatio(Tree1_aRchi)
```

---

Make\_Path

*Make the path of a QSM in an aRchi object*

---

**Description**

Identify and record the paths of a QSM in an object of class aRchi. The path are needed for several tree metrics estimation of the aRchi package.

**Usage**

```
Make_Path(aRchi)

## S4 method for signature 'aRchi'
Make_Path(aRchi)
```

**Arguments**

aRchi            a file of class aRchi

**Details**

A path is a continuous succession of cylinders from a terminal segment (i.e a branch tip) to the trunk base. Thus, there is as many path as terminal segments in a QSM.

This function fill the slot Paths of an object of class aRchi with a data.table of the paths. This data.table contains the same variables as a classic QSM plus an ID\_path column. This table is thus larger than the QSM table as each cylinder is repeated as many times as it appears in a path. For example, the first cylinder of the QSM (i.e the beginning of the trunk) is repeated N path times, with N path the number of path of the QSM.

Many function of the aRchi packages request a path table (check see also).

**Value**

The aRchi object with the table of paths

**See Also**

[BranchAngle](#); [Truncate\\_QSM](#); [Clean\\_QSM](#); [ForkRate](#); [PathFraction](#)

**Examples**

```
# Read a QSM file
file=system.file("extdata","Tree_1_TreeQSM.txt",package = "aRchi")
QSM=read_QSM(file,model="treeQSM")
# Build an object of class aRchi
Tree1_aRchi<-build_aRchi(QSM=QSM)
Tree1_aRchi
# Make the path table
Tree1_aRchi<-Make_Path(Tree1_aRchi)
Tree1_aRchi
PathFraction(Tree1_aRchi)
```

---

*nullOrDatatable-class*    *nullOrDatatable*

---

**Description**

Class union

---

*nullOrLASOrDatatable-class*  
*nullOrLASOrDatatable*

---

**Description**

Class union

---

*nullOrlist-class*    *nullOrlist*

---

**Description**

Class union

---

nullOrnumeric-class	<i>nullOrnumeric</i>
---------------------	----------------------

---

### Description

Class union

---

PathFraction	<i>Compute the path fraction</i>
--------------	----------------------------------

---

### Description

Compute from an object of class aRchi the path fraction

### Usage

```
PathFraction(aRchi)

## S4 method for signature 'aRchi'
PathFraction(aRchi)
```

### Arguments

aRchi            an object of class aRchi with at least the QSM and the Paths table

### Details

The path fraction is the ratio between the mean path length and the maximum path length.

### Value

The path fraction

### References

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

Smith, D. D. et al. Deviation from symmetrically self-similar branching in trees predicts altered hydraulics, mechanics, light interception and metabolic scaling. *New Phytologist* 201, 217–229 (2014).

### See Also

[Make\\_Path](#) to compute the paths table.

**Examples**

```
# Read an aRchi file with at least the QSM and the paths table
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)

PathFraction(Tree1_aRchi)
```

---

plot,aRchi,ANY-method *Plot an object of class aRchi*

---

**Description**

Plot an object of class aRchi.

**Usage**

```
## S4 method for signature 'aRchi,ANY'
plot(
  x,
  y,
  transparency = 1,
  color = "white",
  bg = "black",
  lwd = 3,
  show_point_cloud = FALSE,
  skeleton = TRUE
)
```

**Arguments**

x	An aRchi object
y	Unused (inherited from R base)
transparency	The transparency of the cylinders
color	The color of the cylinders. Can be either a single color or a level of organization: "branching_order" for branching branching_order, "cylinder" to colorize each cylinder independently, "segment" to colorize the branch segments, "axis" to colorize the axis, "A0" to colorize only the main axis from <a href="#">Compute_A0</a> function
bg	The background color
lwd	line width of the skeleton
show_point_cloud	logical (Default = FALSE). Display the point cloud ?
skeleton	logical (Default is TRUE). Display the skeleton only (i.e segments). Faster than displaying the whole QSM with the fleshed cylinders.

## Details

Plot an object of class aRchi in a 3d device. The QSM can be plotted according to different level of organization and the point cloud can be displayed if available.

## Examples

```
# Read an aRchi file with at least a QSM
file=system.file("extdata","Tree1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Plot the QSM by coloring the branching order
plot(Tree1_aRchi,color="branching_order")
# Same with the fleshed cylinder and the point cloud
plot(Tree1_aRchi,color="branching_order",skeleton=FALSE,show_point_cloud=TRUE)
```

---

read\_aRchi

*Read an aRchi file*

---

## Description

Read an aRchi file

## Usage

```
read_aRchi(file)
```

```
## S4 method for signature 'character'
read_aRchi(file)
```

## Arguments

file            The directory to the .aRchi file.

## See Also

[write\\_aRchi](#)

---

read_QSM	<i>Read a QSM</i>
----------	-------------------

---

### Description

Read a QSM file generated with treeQSM, simpletree, simpleforest or pypetree.

### Usage

```
read_QSM(file, model)
```

### Arguments

file	The directory to the QSM file path
model	treeQSM, simpletree, simpleforest or pypetree depending on the algorithm used to generate the QSM

### Details

For treeQSM model, please respect the format with column order provided in the last version <<https://github.com/InverseTampere/TreeQSM/blob/master/README.md>>

### Value

a list containing a data.table with the QSM and a character with the model name. This list can be used to build an aRchi object (see function [build\\_aRchi](#))

### See Also

[aRchi](#) the aRchi class; [build\\_aRchi](#) to build an object of class aRchi

### Examples

```
file=system.file("extdata","Tree_1_TreeQSM.txt",package = "aRchi")
QSM=read_QSM(file,model="treeQSM")
```

---

SelectinQSM_3d	<i>SelectinQSM_3d</i>
----------------	-----------------------

---

### Description

Select interactively a sub-part of a QSM (cylinder, segment, node, axis, branch, subtree) in a 3d device and return its characteristics.

### Usage

```
SelectinQSM_3d(aRchi, skeleton = TRUE, level = "cylinder")
```

```
## S4 method for signature 'aRchi'
```

```
SelectinQSM_3d(aRchi, skeleton = TRUE, level = "cylinder")
```

### Arguments

aRchi	An object of class aRchi
skeleton	logical. Display the skeleton only. Default is TRUE. Faster than displaying the QSM with the fleshed cylinders.
level	character. cylinder (default), segment, node, axis, branch, subtree.

### Details

The selection is performed in two times: i) Identifying the zone of interest in the 3d device and zoom into it if needed. When identified, the user has to hit enter in the R console. At this point, it is impossible to rotate the displayed QSM anymore as the left button of the mouse is used for the selection. However translation are still possible with the right button. ii) Draw a rectangle with the left button of the mouse in the zone of interest.

Some details about the level of organization are given below.

"cylinder": return characteristics for the cylinders selected only

"segment:" return characteristics for the cylinders of the segments selected

"node": return the characteristics for the cylinders of the node selected. For a specific node, select the mother.

"axis": return the characteristics for the cylinders of the axis selected. An axis is a continuous succession of cylinder having a same branching order value.

"branch": return the characteristics for the cylinders of the branch selected. A branch is similar to an axis but regroup also everything that is upstream the axis (i.e all that the axis carries)

"Subtree": return the characteristics for the cylinders of the subtree selected. A subtree is similar to a branch but starting from the cylinder selected and not from the point of insertion of the selected axis. In other word, when the user draw a rectangle on a cylinder, the subtree selection return all that the cylinder carries. If several cylinders are selected, the subtree selection return all that the most downstream cylinder carries.



**Value**

a data.table with the cylinders characteristics at the requested level (i.e sub-part of the original QSM).

**Examples**

```
# Read an aRchi file with at least a QSM
if(interactive()){
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Select a branch
SelectinQSM_3d(Tree1_aRchi,level="branch")
# Same with the fleshed cylinder and keep the branch QSM in an object
My_branch=SelectinQSM_3d(Tree1_aRchi,level="branch",skeleton=FALSE)
My_branch
# Compute the moment of force
Tree1_aRchi=Compute_Mf(Tree1_aRchi,WoodDensity=550)
#Select a cylinder to return the moment of force at his position
SelectinQSM_3d(Tree1_aRchi,skeleton=FALSE)
}
```

---

TreeBiomass

*Tree biomass estimation of the woody part*


---

**Description**

Compute the tree wood biomass at different level of organization

**Usage**

```
TreeBiomass(aRchi, WoodDensity, level = "Tree")
```

```
## S4 method for signature 'aRchi'
```

```
TreeBiomass(aRchi, WoodDensity = NULL, level = "Tree")
```

**Arguments**

aRchi	an object of class aRchi with at least a QSM
WoodDensity	a numeric or a data.table. A single wood density value for the whole tree or one value per cylinder in kg/m <sup>3</sup> . If wood density is given for each cylinder a data.table with two column (i.e cyl_ID and WoodDensity) must be given.
level	character. The level at which the wood biomass is computed. Tree, branching_order or Axis.

**Value**

a numeric or data.table. The wood biomass in Kg at the requested level

**Examples**

```
# Read an aRchi file with at least a QSM
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Compute the whole tree wood biomass.
TreeBiomass(Tree1_aRchi,WoodDensity=550)
```

---

Truncate\_QSM

*Truncate a QSM*


---

**Description**

Truncate a QSM at a radius threshold

**Usage**

```
Truncate_QSM(
  aRchi,
  threshold = NULL,
  Keepdaughters = FALSE,
  plotresult = FALSE
)

## S4 method for signature 'aRchi'
Truncate_QSM(
  aRchi,
  threshold = NULL,
  Keepdaughters = FALSE,
  plotresult = FALSE
)
```

**Arguments**

aRchi	an object of class aRchi with at least a QSM and a Paths table.
threshold	numeric. The radius threshold in meter.
Keepdaughters	logical (default = FALSE). Keep the daughters of the last segment retained even if they are lower than the threshold.
plotresult	logical (default = FALSE). Show the results in a 3d plot if TRUE

**Details**

The threshold is applied to a whole segments. In other word, if a segment has at least one cylinder lower than the threshold it is removed as well as everything upstream (except the direct daughters if Keepdaughters=TRUE).

**Value**

An aRchi file with the QSM truncated

**See Also**

[Clean\\_QSM](#) to clean a QSM of an object aRchi.

**Examples**

```
# Read an aRchifile with a QSM and paths tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# Truncate the QSM: 5cm radius threshold
Truncated_Tree1_aRchi=Truncate_QSM(Tree1_aRchi,plotresult = TRUE,threshold = 0.05)
```

---

WBEparameters

*Compute WBE parameters*


---

**Description**

Compute from an object of class aRchi the West, Brown and Enquist (WBE) scaling exponent (alpha and beta) and the subsequent estimated metabolic rate (theta) at node, branch, branch order or tree level.

**Usage**

```
WBEparameters(aRchi, level = "Tree", position = 10)
```

```
## S4 method for signature 'aRchi'
```

```
WBEparameters(aRchi, level = "Tree", position = 10)
```

**Arguments**

aRchi	an object of class aRchi with at least a QSM and the Nodes table (see function <a href="#">Make_Node</a> )
level	characters. At which level R_ratio has to be estimated. Node for node level, Axis for axis level, branching_order for branch order level and Tree for tree level (default).
position	At which position from the node WBE parameters have to be estimated. Either a numeric or a character. Use a numeric multiple of ten to select the distance from the node in cm where R ratio has to be estimated (e.g 10 for 10cm from the node). Use the % sign after a multiple of ten to select the distance from the node in percentage of the length of the parent and daughters segments (e.g 50% for an estimation at mid-length of the segments). Note that 0 is accepted and correspond to the closest position from the node.

**Details**

Details for WBE parameters calculation are given in the details part of function [Make\\_Node](#).

**Value**

Data.table of the summary (median and mean) of WBE parameters at the selected level. The median should be used according to Bentley et al. 2013.

**References**

Martin-Ducup, O. et al. Terrestrial laser scanning reveals convergence of tree architecture with increasingly dominant crown canopy position. *Functional Ecology* (2020).

Lau, A. et al. Estimating architecture-based metabolic scaling exponents of tropical trees using terrestrial LiDAR and 3D modelling. *Forest Ecology and Management* 439, 132–145 (2019).

Bentley, L. P. et al. An empirical assessment of tree branching networks and implications for plant allometric scaling models. *Ecology Letters* 16, 1069–1078 (2013).

**See Also**

[Make\\_Node](#) for node metrics estimation; [LeonardoRatio](#) to estimates Leonardo Da Vinci's ratio at different level.

**Examples**

```
# Read an aRchifile with a QSM and node tables.
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")
Tree1_aRchi=read_aRchi(file)
# WBE parameters at the branching order level estimated at midlength of the segments
LeonardoRatio(Tree1_aRchi,level="Tree", position="50%")
# WBE parameters at the tree level estimated at 10 cm from the node
LeonardoRatio(Tree1_aRchi)
```

---

WoodSurface

*Estimate the unrolled wood surface from a QSM*

---

**Description**

Estimate the unrolled wood surface from a QSM by summing the area of all cylinders at several level of organization.

**Usage**

```
WoodSurface(aRchi, level = "Tree")

## S4 method for signature 'aRchi'
WoodSurface(aRchi, level = "Tree")
```

**Arguments**

aRchi	a file of class aRchi
level	text. The level at which the wood surface is computed. Tree, branching_order or Axis.

**Value**

a numeric or data.table. The wood surface in m2

**Examples**

```
# Read an aRchi file with at least a QSM.  
file=system.file("extdata","Tree_1_aRchi.aRchi",package = "aRchi")  
Tree1_aRchi=read_aRchi(file)  
# Compute the wood surface at branching order level.  
WoodSurface(Tree1_aRchi,level="branching_order")
```

---

write_aRchi	<i>Write an aRchi file</i>
-------------	----------------------------

---

**Description**

write an aRchi file

**Usage**

```
write_aRchi(aRchi, file)  
  
## S4 method for signature 'aRchi,character'  
write_aRchi(aRchi, file)
```

**Arguments**

aRchi	the object of class aRchi to write
file	The directory where to write the .aRchi file.

**See Also**

[read\\_aRchi](#)

# Index

- add\_pointcloud, [2](#)
- add\_pointcloud, aRchi-method  
(add\_pointcloud), [2](#)
- angle3d, [3](#)
- aRchi, [6](#), [23](#)
- aRchi (aRchi-class), [4](#)
- aRchi-class, [4](#)
  
- BranchAngle, [4](#), [19](#)
- BranchAngle, aRchi-method (BranchAngle),  
[4](#)
- build\_aRchi, [5](#), [23](#)
  
- Clean\_QSM, [6](#), [11](#), [19](#), [27](#)
- Clean\_QSM, aRchi-method (Clean\_QSM), [6](#)
- Compute\_A0, [4](#), [7](#), [10](#), [21](#)
- Compute\_A0, aRchi-method (Compute\_A0), [7](#)
- Compute\_Mf, [8](#)
- Compute\_Mf, aRchi-method (Compute\_Mf), [8](#)
  
- DAI, [8](#), [9](#)
- DAI, aRchi-method (DAI), [9](#)
  
- ForkRate, [6](#), [7](#), [10](#), [19](#)
- ForkRate, aRchi-method (ForkRate), [10](#)
  
- get\_nodes, [11](#), [13–15](#)
- get\_nodes, aRchi-method (get\_nodes), [11](#)
- get\_operations, [12](#)
- get\_operations, aRchi-method  
(get\_operations), [12](#)
- get\_paths, [12](#), [13](#), [14](#), [15](#)
- get\_paths, aRchi-method (get\_paths), [13](#)
- get\_pointcloud, [12](#), [13](#), [14](#), [15](#)
- get\_pointcloud, aRchi-method  
(get\_pointcloud), [14](#)
- get\_QSM, [12–14](#), [14](#)
- get\_QSM, aRchi-method (get\_QSM), [14](#)
  
- LeonardoRatio, [15](#), [17](#), [28](#)
  
- LeonardoRatio, aRchi-method  
(LeonardoRatio), [15](#)
  
- Make\_Node, [4](#), [15](#), [16](#), [16](#), [27](#), [28](#)
- Make\_Node, aRchi-method (Make\_Node), [16](#)
- Make\_Path, [4](#), [18](#), [20](#)
- Make\_Path, aRchi-method (Make\_Path), [18](#)
  
- nullOrDatatable-class, [19](#)
- nullOrLASOrDatatable-class, [19](#)
- nullOrlist-class, [19](#)
- nullOrnumeric-class, [20](#)
  
- PathFraction, [19](#), [20](#)
- PathFraction, aRchi-method  
(PathFraction), [20](#)
- plot, aRchi, ANY-method, [21](#)
  
- read\_aRchi, [6](#), [22](#), [29](#)
- read\_aRchi, character-method  
(read\_aRchi), [22](#)
- read\_QSM, [5](#), [23](#)
  
- SelectinQSM\_3d, [24](#)
- SelectinQSM\_3d, aRchi-method  
(SelectinQSM\_3d), [24](#)
  
- TreeBiomass, [25](#)
- TreeBiomass, aRchi-method (TreeBiomass),  
[25](#)
- Truncate\_QSM, [7](#), [19](#), [26](#)
- Truncate\_QSM, aRchi-method  
(Truncate\_QSM), [26](#)
  
- WBEparameters, [16](#), [17](#), [27](#)
- WBEparameters, aRchi-method  
(WBEparameters), [27](#)
- WoodSurface, [28](#)
- WoodSurface, aRchi-method (WoodSurface),  
[28](#)
- write\_aRchi, [6](#), [22](#), [29](#)

write\_aRchi, aRchi, character-method  
(write\_aRchi), [29](#)