treecm: an introduction

Marco Bascietto

June 20, 2012

The centre of mass is a crucial data for arborists in order to consolidate a tree using static or dynamic cables. Given field-recorded data on branchiness of a tree, the package:

- computes and plots the centre of mass of the tree itself
- simulates the shift in CM position as branches are pruned
- computes branches slenderness coefficient in order to aid the arborist identify potentially dangerous branches
- computes the force acting on a ground plinth and its best position relating to the tree centre of mass, should the tree need to be stabilized by a steel cable

This vignette showcases some real-world cases where the package has been employed. Please notice that although the package is to be used as a quantitative aid to tree pruning and stabilization actions, the authors cannot take any responsability on the accuracy of the package results.

1 Usage examples

1.1 Plot centre of mass

We will make use of the data set bundled in the package to plot a basic view of masses of branches and logs of a stone pine (*Pinus pinea* L.) sampled by B. De Cinti and M. Bascietto (figure 1):

```
> library(treecm)
> data(stonePine1TreeData)
> print(stonePine1TreeData)
```

```
$fieldData
    azimuth dBase dTip length tipD height tilt toBePruned pathToTip     biomass
L1      275    73   41   10.2 2.50   0.00   80      FALSE      TRUE 1825.120033
L2      275    41   16    3.9 2.75  10.20   80      FALSE      TRUE  192.826834
B1      190    15    0     NA 7.95  10.10    0      FALSE     FALSE  123.160705
B2      200    22    0     NA 7.95  10.40    0      FALSE     FALSE  312.987817
```

```
B3      230    15    0     NA 7.95  10.40    0      FALSE     FALSE  123.160705
B4      200    18    0     NA 7.95  11.15    0      FALSE     FALSE  191.998003
B5      180     7    0     NA 7.95  11.30    0      FALSE     FALSE   19.249859
B6      150     6    0     NA 7.95  11.30    0      FALSE     FALSE   13.225032
B7      340    16    0     NA 7.95  11.30    0      FALSE     FALSE  144.121427
B8      220    13    0     NA 3.95  11.80    0      FALSE     FALSE   86.921601
B9      165    19    0     NA 7.95  11.80    0      FALSE     FALSE  219.017380
B10     280     8    0     NA 7.95  11.90    0      FALSE     FALSE   26.647181
B11     170     9    0     NA 3.95  11.90    0      FALSE     FALSE   35.499271
B12     265     8    0     NA 7.95  12.20    0      FALSE     FALSE   26.647181
B13      75     6    0     NA 3.95  12.20    0      FALSE     FALSE   13.225032
B14     180     6    0     NA 7.95  12.20    0      FALSE     FALSE   13.225032
B15     170     6    0     NA 7.95  12.60    0      FALSE     FALSE   13.225032
B16     120     5    0     NA 7.95  12.60    0      FALSE     FALSE    8.483444
B17      10    14    0     NA 3.95  13.00    0      FALSE     FALSE  104.112967
B18     180    13    0     NA 7.95  13.00    0      FALSE     FALSE   86.921601
B19     260    13    0     NA 7.95  13.20    0      FALSE     FALSE   86.921601
B20      75     6    0     NA 3.95  13.20    0      FALSE     FALSE   13.225032
B21      75    10    0     NA 3.95  13.75    0      FALSE     FALSE   45.882751
B22     215     7    0     NA 7.95  13.75    0      FALSE     FALSE   19.249859
B23     140     7    0     NA 7.95  13.75    0      FALSE     FALSE   19.249859
C       275    16    0    3.0 3.00  14.10   80      FALSE      TRUE  144.121427

$density
[1] 650

$allometryFUN
function (x, diameter)
{
    a <- 0.16843
    b <- 2.43523
    powerEquation(a, b, as.real(x[diameter]))
}
<environment: namespace:treecm>

$branchesCM
[1] 1
```

This dataset has been collected for a 17.1 metres tall stone pine whose stem was tilted approx. 20° from the vertical plane (or 80° from the horizontal plane).

The stem has been sectioned in two logs (`L1` and `L2`), and a final branch (`C`). These two logs and the final branch components have been defined in the field as the "main stem" of the tree, all the other components of the tree fall into the crown. The definition of the main stem is important only for the correct assessment of the position of the anchor on the tree, should the tree need stabilization with a steel cable. Main stem components get a `TRUE` value in the `pathToTip` column.

A component with `FALSE` or missing value in the `pathToTip` column is treated as it belonging to the crown. The crown was made up of 23 branches (`B1-B23`), all of them horizontal (*ie* tilted 0°).

The `stonePine1TreeData` dataset is a direct result of importing `stonePine1FieldData` with:

- wood fresh density: 650 $\frac{kg}{m^3}$

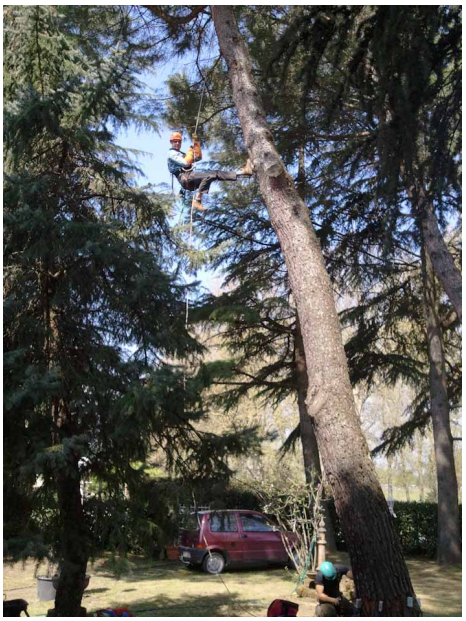


Figure 1: The stone pine measured in `stonePine1TreeData`, crown not visible. The tilted stem is clearly visible, as is the tree climber.

- allometry function for branch mass: `allometryABDC`
- position of the centre of mass: on the branch tip

Log biomass is computed by Smalian's formula (la Marca (2004)). It is important to choose the most appropriate allometric equation in order to yield trustworthy biomass figures and, as a result, appropriate centre of mass coordinates. Allometry equations functions are discussed in section 2.4, page 21.

The package recognizes rows that represent branches because their diameter at tip (`tipD`) is 0 (see more at page 25).

Let's get going and compute the centre of mass of this pine:

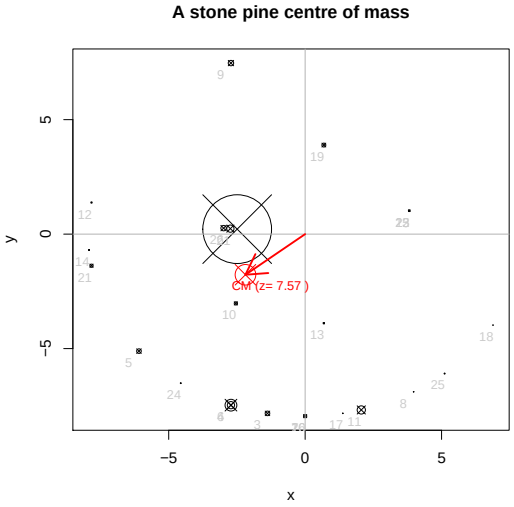
```
> vectors  <- treeVectors(stonePine1TreeData)
> CM       <- centreOfMass(vectors)
> summary(CM)
```

```
Coordinates of the centre of mass:
Cartesian (x/m, y/m, z/m): -2.19 , -1.77 , 7.57
Polar (angle/degrees, distance/m, height/m): 230 , 2.82 , 7.57
```

The core of the package is the `summary` method for `CM` class. The centre of mass for this stone pine lies 2.82 metres South-West of tree base (230° from magnetic North), 7.57 metres above ground. Cartesian coordinates are provided as well, though not so usefull as polar ones.

A simple visualization of tree centre of mass and its logs and branches is achieved simply by:

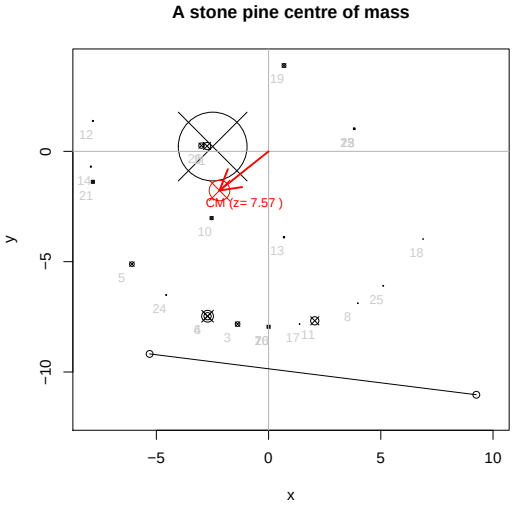
```
> plot(vectors, main = "A stone pine centre of mass")
> plot(CM)
```



In a cartesian coordinate system whose origin lies at tree base, the masses of logs and branches are plotted as vectors pointing inwards, towards the ground. Each circle represents a branch or log mass whose radius is proportional to its mass. Likewise, the centre of mass is plotted as a vector pointing inwards, in red colour. Its height component is written alongside its label as z coordinate. A red arrow approximates the direction the tree will follow should it break at its base.

It is important that, should the tree break, it does not fall onto buildings or cause damage to people. We can add buildings and other important points to the CM plot provided that we measured the polar coordinates of their relevant points, from the the tree base, using the `plotPolarSegment` function. Let's add a building face facing the tree:

```
> plot(vectors,
+   main = "A stone pine centre of mass",
+   xlim = c(-8, 10),
+   ylim = c(-12, 4)
+ )
> plot(CM)
> plotPolarSegment(210, 10.6, 140, 14.4)
```



1.2 Snow load

Snow may increase crown load substantially, sometimes breaking entire branches. As a side effect, snow-loaded crowns may alter tree centre of mass by moving it upwards and, in asymmetric crowns, towards the part of crown under heavier load.

Let's model a snow load that doubles the biomass of branches higher than 12 m:

```
> rows <- substr(row.names(stonePine1TreeData$fieldData), 1, 1)
> component <- substr(row.names(stonePine1TreeData$fieldData), 1, 1)
> stonePine1TreeData$fieldData <- within(stonePine1TreeData$fieldData,
+                                        biomass[height > 12 & component != "L"] <-  biomass[height > 12 & component != "L"] * 2
+                                        )
> rm("rows")
```

Let's recaltulate the vectors under snow load and plot the results:

```
> vectors  <- treeVectors(stonePine1TreeData)
> CM       <- centreOfMass(vectors)
> summary(CM)
```

```
Coordinates of the centre of mass:
Cartesian (x/m, y/m, z/m): -2.10 , -1.72 , 8.43
Polar (angle/degrees, distance/m, height/m): 230 , 2.71 , 8.43
```

```
> plot(vectors,
+   main = "A stone pine centre of mass under 2x snow load",
+   xlim = c(-8,10),
+   ylim = c(-12,4)
+ )
> plot(CM)
> plotPolarSegment(210, 10.6, 140, 14.4)
```

Tree centre of mass has clearly shifted upwards but the snow load would not increase the danger on the building, should the tree collapse.

1.3 Wind load

Winds may increase load on some sectors of the crown and decrease it in other sectors. We would like to model the effect of a prevailing Southbound wind that halves branches mass in the northern sector and doubles it in the southern sector.

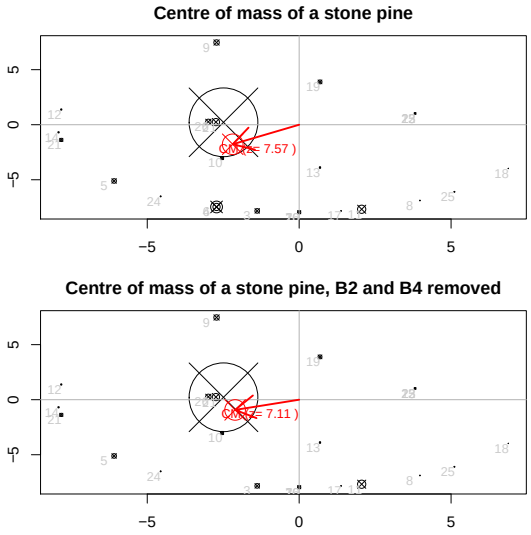
```
> data(stonePine1TreeData)
> rows <- substr(row.names(stonePine1TreeData$fieldData), 1, 1)
> stonePine1TreeData$fieldData <- within(
+   stonePine1TreeData$fieldData, {
+   biomass[((azimuth >= 270 | azimuth < 90) & rows != "L")] <- biomass[((azimuth >= 270 | azimuth < 90) & rows != "L")] / 2
+   biomass[((azimuth >= 90 | azimuth < 270) & rows != "L")] <- biomass[((azimuth >= 90 | azimuth < 270) & rows != "L")] * 2
+ })
> rm(rows)
> vectors  <- treeVectors(stonePine1TreeData)
> CM       <- centreOfMass(vectors)
> summary(CM)


Coordinates of the centre of mass:
Cartesian (x/m, y/m, z/m): -2.14 , -3.00 , 8.57
Polar (angle/degrees, distance/m, height/m): 215 , 3.69 , 8.57
```

Under a heavy southbound wind the CM of the tree will move considerably towards South and 1 metre farther away from tree base. Although too simplicistic a model the results lead to the conclusion that dynamic forces in prevailing wind conditions should be taken into account when assessing tree stability.

1.4 Effect of pruning on CM

As far as static forces are concerned, in an effort to move centre of mass closer to tree base, we could prune a few heavy branches. Let's have a look how CM would move if we cut `B2` and `B4`:

```
> library(treecm)
> data(stonePine1TreeData)
> vectors  <- treeVectors(stonePine1TreeData)
> CM       <- centreOfMass(vectors)
> op <- par(mfrow = c(2, 1), mai = c(0.5,0.5,0.5,0.2))
> plot(vectors, main = "Centre of mass of a stone pine")
> plot(CM)
> component <- row.names(stonePine1TreeData$fieldData)
> stonePine1TreeData$fieldData$toBePruned[component %in% c("B2", "B4")] <- TRUE
> vectors  <- treeVectors(stonePine1TreeData)
> CM       <- centreOfMass(vectors)
> plot(vectors, main = "Centre of mass of a stone pine, B2 and B4 removed")
> plot(CM)
> par(op)
> rm(op)
```



CM has actually moved towards tree base, and farther away from the house. As a matter of facts, branch pruning has been a slight reasonable action towards a safer tree.

1.5 Slenderness ratio

The slenderness ratio of a tree is a pure number defined as $SR = \frac{h}{d}$ where h is the height of the tree trunk, and d is the diameter of the tree Mattheck et al. (1995). The SR is a measure of tree stability and is extensively used in tree stability measures carried out by Visual Tree Assessment (VTA). SR in the range $30 \leq SR \leq 70$ are considered optimal, whereas $SR > 70$ lead to consider the tree at risk of breaking due to its excessive slenderness. The authors have

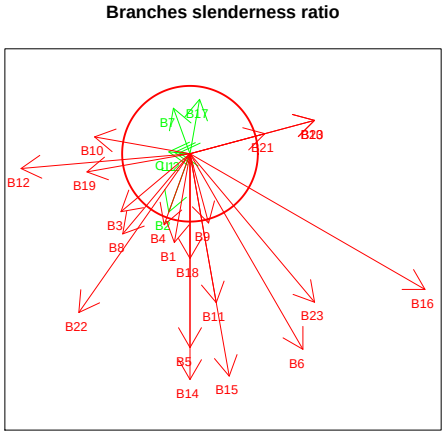
applied the same concept to tree branches as well. While SR in vertical trees has a physical meaning Mattheck et al. (1995), branches are not usually vertical. As the branch starts to deviate from the verticality (as most of the branches do) the arm of the moment gets longer, reaching a maximum limit in horizontal branches. The longer the arm, the higher the stress on the branch. In order to estimate the added stress imposed by branch angle we improved Mattheck's formula by adding a component proportional to branch tilt angle:

$SR_c = \frac{l}{d} \cdot (1 + cos\alpha)$

where α is branch tilt angle (i.e. 90° for a vertical branch, 0° for an horizontal branch). In vertical branches $SR_c = SR$, in horizontal branches $SR_c = 2SR$. As far as we know this is the first attempt to apply the slenderness ratio to branches. Optimal (safe) branches could be in the range $30 \leq SR_c \leq 70$.

When `stonePine1TreeData` object is filled with branches `length` (not to be confused with `tipD`, the length of branch projection on the ground, from the tree base to branch tip) than SR_c can be computed and plotted:

```
> data(stonePine1TreeData)
> # assign length to branches
> stonePine1TreeData$fieldData <- within(stonePine1TreeData$fieldData,
+ length[3:25] <- c(7, 7, 7, 7, 7, 7, 4, 7, 7, 4, 7, 7, 4, 7, 7, 7, 4, 7, 7, 4, 4, 7, 7)
+ )
> vectors <- treeVectors(stonePine1TreeData)
> SR      <- treeSR(stonePine1TreeData,vectors)
> plot(SR, main = "Branches slenderness ratio", xaxt='n', yaxt = 'n', xlab = "", ylab = "")
```



The 2D plot charts branches azimuth as arrows whose length is SR_c. The longer the arrows the more slender the branch. Arrows pointing inside the red circle are considered to be stable, whereas longer arrows are considered as risky ($SR_c \geq 70$). The plot may be a visual clue on the process of branch pruning selection.

1.6 Tree stabilization

Estimating the coordinates of the centre of mass of a tree is crucial to judge its static stability. The centre of mass of a perfectly balanced tree lies in between its trunk, that is the x and y coordinates of the CM lie inside the $\pi \cdot r^2$ surface where r is the radius of tree base.

The more distant the centre of mass from tree base the higher the constrains the tree poses on the ground through its roots. When concerns about tree stability are raised and the tree needs to be consolidated a proper cabling system has to be put in place. Knowing in advance the direction the tree would fall in case of overturning is necessary to properly engineer the cabling system.

One or two static steel cables properly linking the tree to a plinth on the ground may effectively and easily lock the tree into place should its roots loose connection to the ground. Static steel cable systems include single cables and two cables systems:

- **Single cable system**. A single cable is layed down just opposite of CM azimuth ($\pm 180°$). Although cheaper a solution this system has a major drawback: it does not take into consideration that trees move under wind and snow pressures, thereby shifting their CM azimuth in unpredictable ways. A single cable would not counter act tree movements and, should it break down, it would not hold the tree in place. Furthermore, as the tree grows taller, its CM shifts and the ground plinth should be moved accordingly. This is clearly not a feasible solution.

- **Double cable system** A system of two cables joined at the tree anchor is layed down at an even angle from the CM azimuth ($\pm 180°$). This system holds in position a collapsed tree while allowing it some degrees of movement under snow or wind. Further, it decreases the force on the cables and on the plinth, thereby allowing for increased safety.

As version 1.1, `treecm` may be used to design single cable systems.

Although putting in place the cable(s) and building an appropriate plinth on the ground are technically easy, a correct assessment of the masses into play is mandatory to design a proper system:

- The height of the anchor on the tree must be carefully chosen to be above its centre of mass (to prevent its turnover!)

- The height of the anchor on the tree must be well below its tip, to not allow the stem to flex, break and fall down

- The force acting on the cable(s) and on the plinth gets lower the farther away it is from tree base

As a rule of thumb, as far as safety is concerned, the higher on the tree the anchor is and the farther the plinth is from tree base the better. Due to the presence of other trees, buildings etc. in urban settings, scarcely ever it is possible to install very long cables, though.

Function `getPlinthForce` is designed to return the force on the cable and on the plinth given the length of the cable and the position of the anchor on the stem.

Its rationale lies in the comparison of the moment of the tree (applied to its centre of mass) and the moment of the anchor (applied on the anchor):

$$M_{tree} = M_{anchor} \tag{1}$$

Where:

$$M_{tree} = l_{cm} \cdot f_{cm} \cdot sin\alpha \tag{2}$$

and:

$$M_{anchor} = l_{anchor} \cdot f_{anchor} \cdot sin\beta \tag{3}$$

where:

- l_x are the moment arms
- f_x are the weight forces (masses by standard gravity)
- α is the angle on the centre of mass point, between the tree weight vector and the moment arm toward tree base
- β is the angle on the anchor point, between the steel cable and the vector toward tree base

The force on the cable and on the plinth is then easily derived as:

$$f_{anchor} = \frac{M_{tree}}{l_{anchor} \cdot sin\beta} \tag{4}$$

Let's look at a simple example using the `stonePine1TreeData` dataset we seek the force and plinth position by positioning the anchor at 10m along the main stem and for a 40m long steel cable:

```
> library(treecm)
> data(stonePine1TreeData)
> vectors <- treeVectors(stonePine1TreeData)
> CM      <- centreOfMass(vectors)
> ## We need to compute the tree moment
> treeMoment <- buildTreeMomentObject(
+   centreOfMassModulus(CM)
+   , treeTotalBiomass(stonePine1TreeData)
+   , centreOfMassAngle(CM)
+ )
> treeMoment <- calcMoment(treeMoment)
> ## We extract the logs belonging to the main stem
> mainStem <- logPathSelection(stonePine1TreeData)
> (plinth <- getPlinthForce(
+         l.stem = 10,
+         d = 40,
+         logs = mainStem,
+         treeMoment = getMoment(treeMoment),
+         CM = CM
+ ))
```

```
$force
[1] 11842.77

$distanceOnGround
[1] 37.03226

$anchorAlongStem
[1] 10

$cableLength
[1] 40

$anchorHeight
[1] 9.848078

$azimuth
[1] 50
```

A named list of six elements is returned:

1. `force` (11843 N) is the actual force on the steel cable and plinth. Conversion to kilogram-force is approximately done by dividing it by 10 as $1kg_F \approx 9.81N$ (*ie* 1184 kg_F)
2. `distanceOnGround` (37.03 m) is the distance between the plinth and tree base (assuming a flat terrain)
3. `anchorAlongStem` (10 m) is the distance between the anchor and tree base, following the tree main stem
4. `cableLength` (40 m)
5. `anchorHeight` (9.85 m) is the height above ground of the anchor, equal to `anchorAlongStem` only when the main stem is vertical (90° above ground)
6. `azimuth` (50°) is the azimuth relative to North where the plinth should be positioned (this is simply the CM azimuth ±180°)

We now have the polar coordinates for the position of the plinth (`distanceOnGround`, `azimuth`) and the force on it in order to engineer the cable width and the plinth accordingly.

What if we had constraints on the position of the plinth? It turns out that `getPlinthForce` is vectorized both to `l.stem` and `d`.

Let's examine the possible outcomes for a 15m to 50m long cable:

```
> plinth <- getPlinthForce(
+         l.stem = 10,
+         d = 15:50,
+         logs = mainStem,
+         treeMoment = getMoment(treeMoment),
+         CM = CM
+ )
> print(plinth$force)
```

```
 [1] 17170.94 16133.11 15378.09 14804.64 14354.82 13993.00 13696.03 13448.23
 [9] 13238.55 13059.03 12903.75 12768.23 12649.04 12543.47 12449.38 12365.05
[17] 12289.09 12220.34 12157.87 12100.87 12048.68 12000.73 11956.55 11915.73
[25] 11877.90 11842.77 11810.05 11779.53 11750.99 11724.25 11699.16 11675.56
[33] 11653.35 11632.39 11612.60 11593.87
```

There's almost a 30% decrease in the force to the plinth if we had the chance of setting the plinth 50m away from tree base. Let's now assess how force to the plinth varies when we move the anchor position along the stem. Remember that the tree centre of mass is 7.57m high, and that `l.stem` is the distance between tree base and the anchor, following the stem, not the anchor height on the ground:

```
> plinth <- getPlinthForce(
+         l.stem = seq(9, 12, 0.5),
+         d = 40,
+         logs = mainStem,
+         treeMoment = getMoment(treeMoment),
+         CM = CM
+ )
> print(plinth$force)
```

```
[1] 13014.34 12396.25 11842.77 11344.76 10894.78 10486.65 10115.24
```

There's almost a 20% decrease in the force to the plinth at 12m along the stem. If the tree trunk at its 12m was large enough we could position the anchor there.

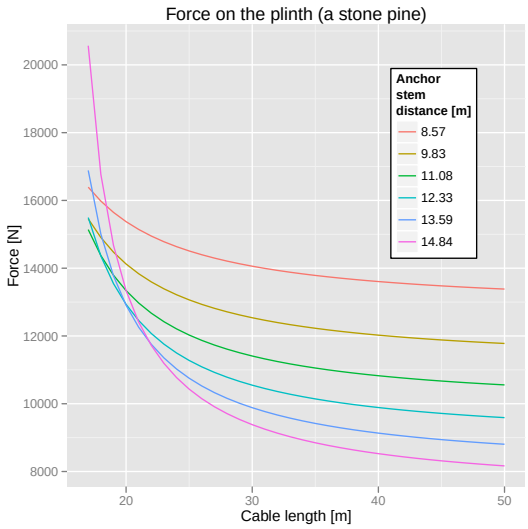
Let's have a closer look at how the force to the plinth reacts by letting vary both `l.stem` and `d`:

```
> aR <- anchorRange(mainStem, CM)
> l.stemSeq <- round(seq(aR[["z"]] + 1, aR[["hMax"]] - 2, length.out = 6), 2)
> plinth <- data.frame(
+         getPlinthForce(
+                 l.stemSeq,
+                 17:50,
+                 mainStem,
+                 getMoment(treeMoment),
+                 CM
+ ))
> head(plinth)
```

```
     force distanceOnGround anchorAlongStem cableLength anchorHeight azimuth
1 16391.09         13.26886            8.57          17     8.439802      50
2 15980.26         14.41057            8.57          18     8.439802      50
3 15647.68         15.53446            8.57          19     8.439802      50
4 15373.36         16.64384            8.57          20     8.439802      50
5 15143.56         17.74123            8.57          21     8.439802      50
6 14948.51         18.82857            8.57          22     8.439802      50
```

We make use of the `anchorRange` function to select the proper range along the stem (see page 9). We compute `getPlinthForce` for a range of distances anchor-tree base [8.57..14.84], each distance for a range of cable lengths [17..50]. Converting the list in a data frame let us plot it with `ggplot2`:

```
> library(ggplot2)
> ggplot(data = plinth, aes(x = cableLength, y = force)) +
+   geom_line(aes(color = factor(anchorAlongStem), group = anchorAlongStem)) +
+   ylab('Force [N]') +
+   xlab('Cable length [m]') +
+   labs(colour = "Anchor\nstem\ndistance [m]") +
+   opts(title = "Force on the plinth (a stone pine)",
+         legend.position = c(.8, .7),
+         legend.background = theme_rect(fill="white")
+   )
```

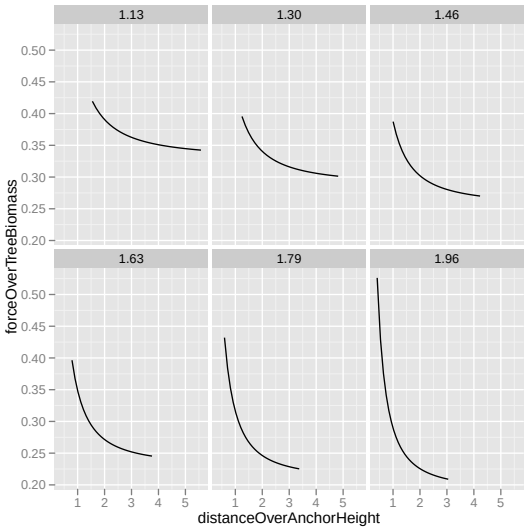


Expressing force as a ratio to tree biomass and expressing distances relative to tree CM height or distance anchor-tree base:

```
> plinth <- transform(plinth,
+                     distanceOverAnchorHeight = distanceOnGround / anchorAlongStem
+                     , heightOverAnchorHeight = round(anchorAlongStem / CM[["z"]], 2)
+                     , forceOverTreeBiomass   = force / treeTotalBiomass(stonePine1TreeData)/10
+                     )
> head(plinth[c("distanceOverAnchorHeight", "heightOverAnchorHeight", "forceOverTreeBiomass")])
```

```
  distanceOverAnchorHeight heightOverAnchorHeight forceOverTreeBiomass
1                 1.548292                    1.13            0.4193783
2                 1.681513                    1.13            0.4088669
3                 1.812656                    1.13            0.4003576
4                 1.942105                    1.13            0.3933389
5                 2.070156                    1.13            0.3874593
6                 2.197033                    1.13            0.3824687
```

```
> ggplot(data = plinth,
+         aes(x = distanceOverAnchorHeight, y = forceOverTreeBiomass)) +
+   geom_line() +
+   facet_wrap(~heightOverAnchorHeight)
```



It looks like positioning the anchor ≈ 1.6 times the height of the centre of mass, positioning the plinth 3 times the distance tree base-anchor, would expose the cable and the plith to 1/4 of the weight of the tree itself.

1.6.1 A word of caution

The cable system is engineered for the safety of people and their properties. Let's not forget that it should not add to the dangers of the tree breaking down! Practisioners should pay attention that:

- in the case of the tree breaking down it should immediatly lean on the anchor
- the anchor on the tree should not be strictly tightened to the trunk. The secondary growth of the trees would include the cable itself, resulting in a weakness point along the stem (figure 2)
- the anchor should be loose on the trunk, positioned just above a fork. The anchor should be loosened from time to time in order to avoid the previous effect
- the cable must not be extremely tightened so as to constrain the tree in its position so as to enable it its natural movements under wind or snow. At the other end a loosened cable would enable the tree to gain speed, should it break, before leaning on the anchor. A compromise should be reached between a tight cable and a loose one
- from time to time the cable must be checked for potential damages
- as soon as the tree grows taller and changes its crown layout, or as soon as a pruning is carried out, the anchor-cable-plinth system must be modified

(a) Damage to a tree by a tight collar

(b) Close up to the weakness point, notice the marks of the anchor (righthand side) and of a wooden stick (lefthand side) left deep in the wood

Figure 2: The anchor had been put into place too tight, the secondary tree growth had started to overgrow it. Luckily the anchor has been removed but the weakness point still remains

accordingly. Rememeber that the plinth, being a quasi-permanent structure should be engineered to resist to potentially higher forces and that the anchor position should be raised as the tree get taller and heavier.

2 Data collection

Data collection to estimation of the centre of mass is carried out in three steps:

1. Field measurements
2. Visual check for correctness of assumptions
3. Collection of correct allometric equation in order to estimate branch and foliage biomass

2.1 Field measurements

A few field measurements are needed to estimate centre of mass position at the stem level and at the branch level. Field data are easily recorded by climbing the tree using tree-climbing techniques or by hydraulic platforms. A few instruments are needed including:

- A forestry caliper to measure diameter of logs and branches
- A clinometer, or ipsometer or any other instrument to measure height of branches or logs
- A measuring tape to measure length of branch or log projections on the ground

2.1.1 Measurements on logs

The stem is ideally sectioned in logs in order to compute their volume and mass. The measurements to be taken on each log include:

- Diameter at the base of the log, in cm (`dBase`)
- Diameter at the top of the log, in cm (`dTip`)
- Length of the log, in m (`length`)
- Azimuth of the log (`azimuth`), in case it is not vertical, in degrees from North (0° North, 180° South)
- Distance between tree base and the projection of log tip on the ground (`tipD`), in m
- Height above ground of the base of the log, in m (`height`)
- Log tilt (`tilt`) from the horizontal plane (eg a vertical log is tilted by 90°, an horizontal log is tilted by 0°), in degrees (optional, default to 0 if missing)

2.1.2 Measurements on branches

Each branch contributes to the position of the centre of mass by means of their wooden component and their foliage component. Every part of a tree carrying foliage is considered to be a branch. This definition applies to tree tip as well, although some trees may have lost their tip or have it removed during topping operations. The measurements to be taken on each branch include:

- Diameter at the base of the branch, in cm (`dBase`)
- Diameter at the top of the log must be 0
- Length of the branch, from its fork on the stem to its tip, in m (`length`), only if slenderness ratio (SR, page 7) is to be computed
- Azimuth of the branch (`azimuth`), in degrees, usually measured with a compass (0° North, 180° South)
- Distance between tree base and the projection of branch tip on the ground (`tipD`), in m
- Height above ground of the branch insertion into the stem, in m (`height`)
- Branch tilt (`tilt`) from the horizontal plane (eg a vertical branch is tilted by 90°, an horizontal branch is tilted by 0°), in degrees (optional, default to 0 if missing)

2.1.3 Additional optional fields

Two more boolean fields are not strictly measured but they can initially be recorded in the field:

- Pruning status (`toBePruned`), which branches are going to be pruned? How would it affect CM? Optional, defaults to FALSE
- Main stem selection (`pathToTip`), which logs and branches make part of the "main stem" of the tree? Optional, defaults to FALSE

2.2 Visual check for correctness of assumptions

2.2.1 Relative position of centre of mass of branches and logs

The position of the centre of mass of a tree is computed taking into account the centre of mass of each branch and log. Pinpointing the centre of mass along a branch, taking into account branch form factor and the pattern of distribution of leaves biomass along it, would require many more field measures highering the time spent on it and the costs of the sampling.

Since the package aims to help engineering a consolidation system, the centre of mass is by default located at branches or logs tip. This leads to an estimate of the coordinates of the tree centre of mass that is farther away from the base than the actual one. This difference can be regarded as an inherent safety factor.

The package behaviour can be modified in order to let the position branches and logs centre of mass to get nearer to their base. The relative position of the centre of mass of branches and logs can be set as a real number ranging from 0.01 (base) to 1 (tip, the default behaviour). Setting can be done during import of field data using function `importField` and its parameter `bCM` or by using the setter function `setBranchesCM`.

2.3 Wood density

Log mass is estimated by converting its volume (as measured in the field) to fresh mass. The conversion factor is usually referred to as fresh density. Wood density is usually quite conservative among individual of the same tree species. Density values are commonly found in published literature. The dataset `Dst` (Niklas et al. (2010)) can be a useful starting point to assess wood density, should it be unavailable, (density in $\frac{kg}{m^3}$, measured at 50% moisture content):

```
> library(treecm)
> data(Dst)
> print(Dst)
```

	species	group	density
1	Abies alba	conifer	545
2	Abies alba	conifer	577
3	Abies balsama	conifer	529
4	Abies grandis	conifer	449
5	Abies procera	conifer	465
6	Agathis vitiensis	conifer	673
7	Araucaria angustifolia	conifer	689
8	Chaemaecyparis lawsoniana	conifer	497
9	Larix decidua	conifer	673
10	Larix eurolepis	conifer	577
11	Larix kaempferi	conifer	609
12	Picea abies	conifer	497
13	Picea alba	conifer	529
14	Picea omorika	conifer	497
15	Picea sitchensis	conifer	481
16	Picea sitchensis	conifer	529
17	Pinus caribaea	conifer	977
18	Pinus contorta	conifer	593
19	Pinus holfordiana	conifer	513
20	Pinus nigra	conifer	609
21	Pinus nigra	conifer	705
22	Pinus pinaster	conifer	609
23	Pinus ponderosa	conifer	561
24	Pinus radiata	conifer	577
25	Pinus radiata	conifer	641
26	Pinus strobus	conifer	433
27	Pinus sylvestris	conifer	625
28	Podocarpus sp.	conifer	641
29	Podocarpus guatemalensis	conifer	657
30	Pseudotsuga menziesii	conifer	625
31	Pseudotsuga menziesii	conifer	673
32	Thuja heterophylla	conifer	545

33	Thuja heterophylla	conifer	593
34	Thuja heterophylla	conifer	609
35	Thuja plicata	conifer	465
36	Aesculus hippocastanum	dicot	657
37	Acacia mollissima	dicot	897
38	Acer psedudoplatanus	dicot	721
39	Afzelia quanzensis	dicot	1137
40	Alnus glutinosa	dicot	675
41	Alstonia boonei	dicot	497
42	Anthocephalus chinensis	dicot	567
43	Aspidosperma sp.	dicot	993
44	Autranella congolensis	dicot	1144
45	Berlinia confusa	dicot	849
46	Betula sp.	dicot	801
47	Brachstegia nigerica	dicot	865
48	Brachylaena hutchinsii	dicot	1153
49	Byrsonima coriacea	dicot	865
50	Calophyllum brasiliense	dicot	817
51	Canarium schweinfurthii	dicot	593
52	Carpinus betulus	dicot	865
53	Cassispourea malasana	dicot	897
54	Castanea sativa	dicot	657
55	Cedrela odorata	dicot	433
56	Celtis sp.	dicot	961
57	Ceratopetalum apetalum	dicot	733
58	Chlorophora excelsa	dicot	817
59	Cordia millenii	dicot	545
60	Cullenia ceylanica	dicot	769
61	Cyanomeria alexandri	dicot	1121
62	Cylicodiscus gabunensis	dicot	1185
63	Dipterocarpus sp.	dicot	929
64	Dipterocarpus acutangulus	dicot	913
65	Dipterocarpus caudiferus	dicot	753
66	Dipterocarpus zeylanicus	dicot	977
67	Drychalanops beccarii	dicot	865
68	Drychalanops keithii	dicot	902
69	Drychalanops lanceolata	dicot	865
70	Entandrophragma angolense	dicot	705
71	Entandrophragma cylindricum	dicot	833
72	Entandrophragma utile	dicot	833
73	Eperua sp.	dicot	1169
74	Erythrophteum sp.	dicot	1362
75	Eucalyptus piluaris	dicot	897
76	Eucalyptus marginata	dicot	1009
77	Eucalyptus microcorys	dicot	1234
78	Eucalyptus paniculata	dicot	1346
79	Eucalyptus versicolor	dicot	1041
80	Eusideroxylon zwageri	dicot	1282
81	Fagus sylvatica	dicot	833
82	Fraxinus excelsior	dicot	801
83	Gmelina arborea	dicot	625
84	Gonystylus macrophyllum	dicot	785
85	Gossweilerodendron balsamiferum	dicot	641
86	Guarca excelsa	dicot	689
87	Guarca thompsonii	dicot	817
88	Heritiera simplicifolia	dicot	753
89	Heritiera simplicifolia	dicot	801

90	Hevea brasiliensis	dicot	865
91	Hopea sengal	dicot	817
92	Khaya anthotheca	dicot	657
93	Khaya grandiflora	dicot	817
94	Khaya ivorensis	dicot	641
95	Khaya nyascia	dicot	705
96	Koordersiodendron pinnatum	dicot	1089
97	Loniciocarpus castillo	dicot	1169
98	Lophira alata	dicot	1292
99	Lovoa trichilioides	dicot	673
100	Maesopsis veminii	dicot	609
101	Mansonia altissima	dicot	801
102	Mora excelsa	dicot	1137
103	Muragne sp.	dicot	689
104	Nauclea diderrichii	dicot	945
105	Nectrandra sp.	dicot	689
106	Newtonia buchaneni	dicot	705
107	Nothofagus sprocera	dicot	561
108	Ocotea rodiaei	dicot	1250
109	Ocotea usambarensis	dicot	769
110	Octomeles sumatrana	dicot	481
111	Olea hochstetteri	dicot	1121
112	Oxystigma oxyphyllum	dicot	801
113	Parashorea sp.	dicot	705
114	Parashorea malaanonan	dicot	641
115	Parashorea tomentelia	dicot	577
116	Peltogyne sp	dicot	1105
117	Pericopsis elata	dicot	977
118	Pipradeniostrum africanum	dicot	849
119	Platanus hybrida	dicot	785
120	Populus canadensis	dicot	529
121	Populus x canescens	dicot	577
122	Populus x canescens	dicot	481
123	Protium decendrum	dicot	801
124	Prunus avium	dicot	753
125	Pseudosindora palustris	dicot	833
126	Pterocarpus angolensis	dicot	881
127	Pterygota bequaertii	dicot	849
128	Pterygota macrocarpa	dicot	705
129	Qualea sp.	dicot	897
130	Quercus sp.	dicot	833
131	Quercus cerris	dicot	929
132	Quercus rubra	dicot	865
133	Ricinodendron rautanenii	dicot	224
134	Salix x alba	dicot	529
135	Salix alba var. coerulea	dicot	513
136	Salix fragilis	dicot	529
137	Sclerocarpa sp.	dicot	657
138	Scottellia coriacea	dicot	849
139	Shorea acuminatissima	dicot	609
140	Shorea faguetiana	dicot	673
141	Shorea gibbosa	dicot	625
142	Shorea guiso	dicot	993
143	Shorea hakeifolia	dicot	689
144	Shorea leptoclados	dicot	545
145	Shorea macrophylla	dicot	449
146	Shorea parviflora	dicot	513

```
147               Shorea pauciflora   dicot     689
148                Shorea smithiana   dicot     513
149                  Shorea superba   dicot     945
150                  Shorea superba   dicot    1057
151                 Shorea waltonii   dicot     529
152              Staudtia stipitata   dicot    1139
153               Sterculia oblonga   dicot     913
154           Sterculia rhinopetala   dicot     961
155           Swartzia leiocalycine   dicot    1298
156           Symphonia globulifera   dicot     881
157           Syncarpia glomulifera   dicot    1025
158                Tarrietia utilis   dicot     817
159                 Tectona grandis   dicot     801
160                 Tectona grandis   dicot     817
161            Terminalia amazonica   dicot     961
162            Tieghemelia heckerii   dicot     801
163                   Tilia vulgaris   dicot     657
164        Triplochiton scleroxylon   dicot     465
165                    Ulmus glabra   dicot     753
166                Ulmus hollandica   dicot     641
167                   Ulmus procera   dicot     641
168                 Virola koschnyi   dicot     657
169                    Vochvsia sp.   dicot     657
170            Vochysia hondurensis   dicot     577
```

Please note that the dataset provides density figures for wood at 50% moisture content, this is not fresh density (100% moisture content) as needed by `treecm`. A conversion factor shall be applied by the user. As an example, a maritime pine density at 50% moisture content is:

```
> data(Dst)
> with(Dst, density[species == "Pinus pinaster"])
```

```
[1] 609
```

The datasets for stone pine provided with the package have been built taking into account a ≈ 1.07 conversion factor from 50% to 100% moisture content density.

2.4 Choosing a correct allometric equation in order to estimate branch and foliage biomass

It is not possible to weight the branches of a living tree. As a result branch and foliage biomass has to be estimated, this is usually done using branch diameter at its base. Models relating size or biomass to diameter of trees or branches are known as allometric equations. They usually take the form of $Y = a \cdot X^b$ where Y is branch biomass, X is branch diameter, a and b are parameters estimated on a sample of branches (eg during a pruning process).

When sampling is not possible one should rely on published allometric equations and feed them to `treecm`. Currently `treecm` ships with four allometric equations:

- `allometryABDC`, tested on stone pine branches, 5-16 cm diameter, returns fresh weight
- `allometryAsca2011`, tested on stone pine branches, 8-16 cm diameter, predominantly from the lower layers of the crown, returns fresh weight, it is based on a subsample of `allometryABDC` and it must be considered as deprecated
- `allometryCutini2009`, tested on stone pine trees (not on branches), 24+ cm diameter, returns biomass, dry weight, Cutini et al. (2009), its results should be increased by the estimated amount of water present in the branches
- `allometryPorte2002`, tested on maritime pine branches, 10- cm diameter, returns biomass (dry weight), Porté et al. (2002), its results should be increased by the estimated amount of water present in the branches

The proper allometric equation to be used must be fed to `treecm` when importing field data using function `importFieldData`, parameter `branchesAllometryFUN`. Please notice that:

- one should pick an allometric equation that yields fresh mass of branches in order to get results as closer as possible to the real tree centre of mass
- choosing the allometric equation is crucial to a correct estimation of masses. An allometric equation must be choosen according mainly to the tree species, and to the range of branch diameters it has been fitted on

2.4.1 A real example

Let's have a look at how the allometric equation may affect the estimate of branch masses in a stone pine. Dataset `stonePine2FieldData` holds field recorded value for a $\approx$ 11m tall stone pine, having only a few very big branches. We will compare the estimate of whole tree mass carried out applying two allometric equations: `allometryCutini2009` ($Y = -198 + 0.620 \cdot X^2$) and `allometryABDC` ($Y = 0.17 \cdot X^{2.4}$).

```
> csvFile <- system.file("data/stonePine2FieldData.csv.gz", package = "treecm")
> sP.Cutini <- treeBiomass(importFieldData(csvFile, 650, allometryCutini2009))
> head(sP.Cutini$fieldData)
```

```
   azimuth dBase dTip length tipD height tilt toBePruned pathToTip  biomass
L1       0    67   40    6.8  0.0    0.0   90      FALSE      TRUE 1056.886
B1     250    40    0     NA  7.8    6.8    0      FALSE     FALSE  793.764
B2     240    32    0     NA  8.9    7.8    0      FALSE     FALSE  436.644
B3      55    25    0     NA  9.0    9.0    0      FALSE     FALSE  189.264
B4     260    10    0     NA  5.5   10.0    0       TRUE     FALSE -136.236
B5      80    36    0     NA  8.2   10.6    0      FALSE     FALSE  605.284
```

```
> sP.ABDC   <- treeBiomass(importFieldData(csvFile, 650, allometryABDC))
> head(sP.ABDC$fieldData)
```

```
   azimuth dBase dTip length tipD height tilt toBePruned pathToTip    biomass
L1       0    67   40    6.8  0.0    0.0   90       FALSE       TRUE 1056.88596
B1     250    40    0     NA  7.8    6.8    0       FALSE      FALSE 1342.15910
B2     240    32    0     NA  8.9    7.8    0       FALSE      FALSE  779.48153
B3      55    25    0     NA  9.0    9.0    0       FALSE      FALSE  427.29213
B4     260    10    0     NA  5.5   10.0    0        TRUE      FALSE   45.88275
B5      80    36    0     NA  8.2   10.6    0       FALSE      FALSE 1038.42226
```

```
> biomassRaw <- data.frame(
+   cutini = sP.Cutini$fieldData$biomass
+   , ABDC = sP.ABDC$fieldData$biomass
+   , code = rownames(sP.ABDC$fieldData)
+   , diameter = sP.ABDC$fieldData$dBase
+   )
> library(reshape2)
> biomass <- melt(
+   biomassRaw
+   , measure.vars = c("ABDC", "cutini")
+   , value.name = "biomass"
+   , variable.name = "allometry"
+ )
> rm(biomassRaw)
> (treeBiomass <- with(biomass, tapply(biomass, allometry, sum)))
```

```
    ABDC   cutini
6547.653 3993.114
```

`allometryABDC` function estimate for the biomass of the tree is 1.6 times higher than `allometryCutini2009` function! This huge gap is accounted for by the way the allometric equations were designed:

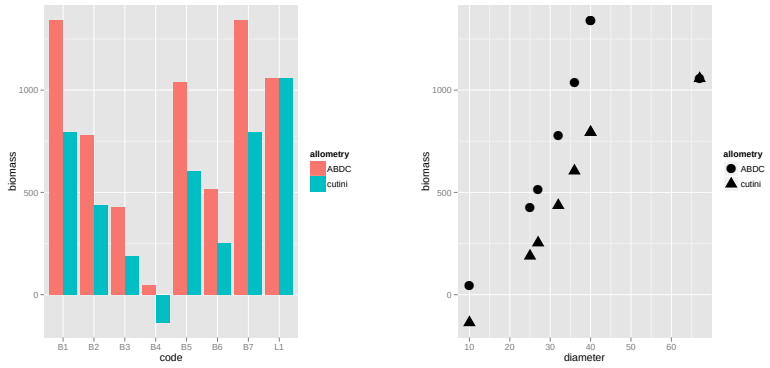
1. `allometryABDC` has been fitted on small-diameter branches, as a result it may predict unreliable mass figures for out of range branches

2. `allometryCutini2009` has been fitted on 24+ cm trees, as a result it properly describes the relationship between mass and diameter as far as diameter is concerned. We must assume that the relationship tree diameter-tree mass holds for branches as well, though. Further the equation yields biomass at 0% moisture content, a figure much lower than fresh biomass.

The difference in the estimates may be clearly appreciated in figure 3.

Notice that `cutini`'s biomass estimate for `B4` is unreasonably negative (figure 3a. This is due to the small diameter branch (10 cm), far below the 24cm lower limit. Also notice consitently lower estimates from `cutini`'s function.

Which function to use? The answer is constrained by the fact that `ABDC` function cannot be reasonably applied since it has not been fitted on the range of branch diameters our pine shows. How shall we adapt `cutini`'s estimates? First of all we need to fix the biomass estimate for `B4`, we shall use the `ABDC` biomass figure for it. Then we shall convert 0% moisture biomass to fresh biomass. We will use a 1.5 expansion factor assuming there's a 50% decrease in mass from fresh to dry state.



(a) Bar plot (b) Dot plot biomass *Vs.* diameter

Figure 3: Bar plot and dot plot comparing biomass estimates of `allometryABDC` and `allometryCutini2009` functions to `stonePine2FieldData`

```
> biomass <- within(biomass, {
+   biomass[allometry == "cutini" & code != "L1"] <- biomass[allometry == "cutini" & code != "L1"] * 1.5
+   biomass[allometry == "cutini" & code == "B4"] <- biomass[allometry == "ABDC" & code == "B4"]
+ })
> with(biomass, tapply(biomass, allometry, sum))
```

```
    ABDC   cutini
6547.653 5711.465
```

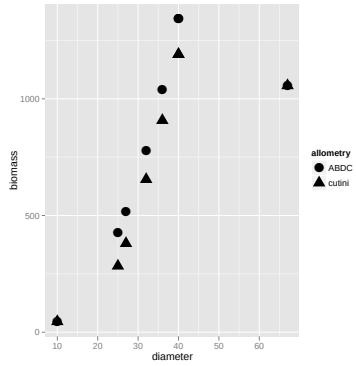


Figure 4: Dot plot comparing biomass estimates of `allometryABDC` and `allometryCutini2009` functions to `stonePine2FieldData` after adjusting the latter

This is just an attempt to raise awarness on the complexities of choosing a correct allometric equation. When we reckon our data are correct we just have to plug them into a proper `fieldData` data frame:

```
> sP.Cutini$fieldData$biomass <- biomass$biomass[biomass$allometry == "cutini"]
> head(sP.Cutini$fieldData)
```

```
   azimuth dBase dTip length tipD height tilt toBePruned pathToTip    biomass
L1       0    67   40    6.8  0.0    0.0   90       FALSE       TRUE 1056.88596
B1     250    40    0     NA  7.8    6.8    0       FALSE      FALSE 1190.64600
B2     240    32    0     NA  8.9    7.8    0       FALSE      FALSE  654.96600
B3      55    25    0     NA  9.0    9.0    0       FALSE      FALSE  283.89600
B4     260    10    0     NA  5.5   10.0    0        TRUE      FALSE   45.88275
B5      80    36    0     NA  8.2   10.6    0       FALSE      FALSE  907.92600
```

3 Correct layout of CSV file

A sample CSV data file is provided in the `data` directory. Function `importFieldData` loads and stores CSV files and along with needed data. CSV files are made up of 10 columns. The first row has to hold column headers. Headers are case sensitive. Each row holds individual log or branch data. Headers include:

1. **code** a simple code assigned to each log or branch
2. **azimuth** orientation, ie: compass bearing in degrees
3. **dBase** diameter of log or branch basal section, in cm
4. **dTip** diameter of log or branch tip (always 0 for branches), in cm
5. **length** log length, in m; also branch length if slenderness ratio (SR, page 7) is to be computed, leave it empty otherwise
6. **tipD** distance of the tip of the log or branch to tree base (different from branch length when tree stem is not vertical)
7. **height** height of log basal section of height of branch insertion on stem
8. **tilt** log or branch tilt from the horizontal plane (eg a vertical branch is tilted by 90°, an horizontal branch is tilted by 0°), in degrees (optional, only useful to estimate z coordinate of centre of mass)
9. **toBePruned** boolean to simulate branch pruning
10. **pathToTip** boolean to point out the "main stem"

3.1 Rules to layout a correct CSV file

Please notice that some rules have to be followed in order to record sound data in the field:

- the diameter of the tip of `L1` is equal to the diameter of the base of `L2`. `L2` tip diameter is, in turn, equal to `C` base diameter. Height figures must match between consecutive logs, as well as diameter measures do

- the distance of the tip of the branch (`tipD`) is not the length of the branch but the distance between tree base (the origin of the cartesian plot) and the branch tip
- note that `length` has been only recorded for the `C` branch (not considering logs) as it is the only branch not being horizontal. Non horizontal branches affect tree CM z-coordinate. When non-horizontal branches are present, and if one is interested in the z-coordinate of CM, than branch length and its angle from the horizonatl plane (`tilt`) should also be recorded. Otherwise branch `length` is not needed.

4 Contribute!

`treecm` is an ongoing project hosted on GitHub (`http://mbask.github.com/treecm/`). Many areas need to be expanded including:

- branch biomass estimation; allometric equations are used to estimate fresh branch and foliage biomass. So far only branch biomass for stone pine and maritime pine have been developed or integrated into the software from published data. We need to expand further the number of species represented (allometric equations relating branches fresh weight and their diameter, or raw data), particularly for the common species in urban areas such as cedars, magnolias, oaks
- The package does not estimate the position of the centre of mass of tree branches. This position may vary according to foliage mass and its distribution along the branchs, branch tapering, quantity of water in leaves (*ie* shaded or lit leaves) *etc.* The position must be fed to the package during data loading, as the variable `bCM`. Although going for the safe road, setting a branch centre of mass position on its tip may not be sufficiently precise should one assess wood quality as a function of load balance. Work is under way in order to to model branch load balance
- As far as postion of centre of mass, the package does not tell branches and logs apart. The position of CM in logs follows branches CM position settings, though not realistic

5 Future enhancements

- `getPlinthForce` should also account for the case where two cables could be laid down from the same anchor, to enhance safety
- `getPlinthForce` could also be used to test the engineering of a laid down system

References

Cutini, A. and Hajny, M. and Gugliotta, O. and Manetti, M. and Amorini, E. 2009, Effetti della struttura del popolamento sui modelli di stima del volume e della biomassa epigea (Pineta di Castelfusano - Roma) *Forest@*, **6**, 75–84

la Marca, O. 2004, *Elementi di dendrometria*. Patron Editore (Bologna), p. 119

Niklas, K. J. and Spatz, H.-C., 2010, Worldwide correlations of mechanical properties and green wood density *American Journal of Botany*, **97**, 1587–1594

Porté, A. and Trichet, P. and Bert, D. and Loustau, D. 2002, Allometric relationships for branch and tree woody biomass of Maritime pine (*Pinus pinaster* Ait.) *Forest Ecology and Management*, **158**, 71–83

Mattheck, C. and Breloer, H., 1995, *The Body Language of Trees: A Handbook for Failure Analysis (Research for Amenity Trees)*. HMSO (London).