R: Split/Apply/Combine using the Orange dataset

```
# clear workspace
rm(list=ls())
# Get info on "Orange" R dataset
help("Orange")
# Make a copy of the "Orange" template and save as my
orange <- as.data.frame(Orange)
```

R *Orange* dataset = original template

Growth of Orange Trees

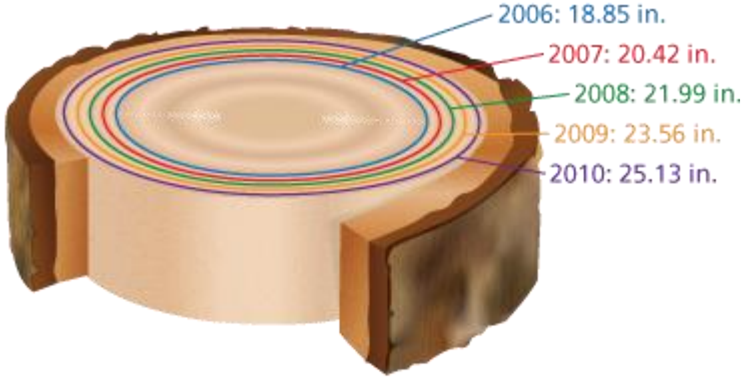
Description

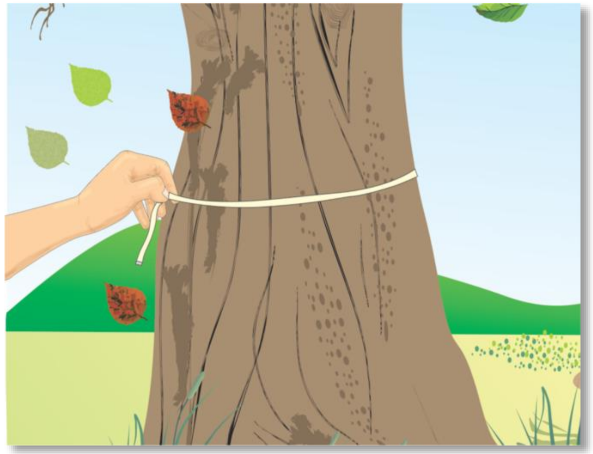
The Orange data frame has 35 rows and 3 columns of records of the growth of orange trees.

Columns:

- **Tree** - ordered factor indicating the tree on which the measurement is made.
- **age** - numeric vector giving the age of the tree (days since 1968/12/31)
- **circumference** - numeric vector of trunk circumferences (mm).

“If I had an hour to solve a problem, I'd spend 55 minutes thinking about the problem and 5 minutes thinking about solutions.”
— *Albert Einstein*





Data Exploration: "orange" data frame

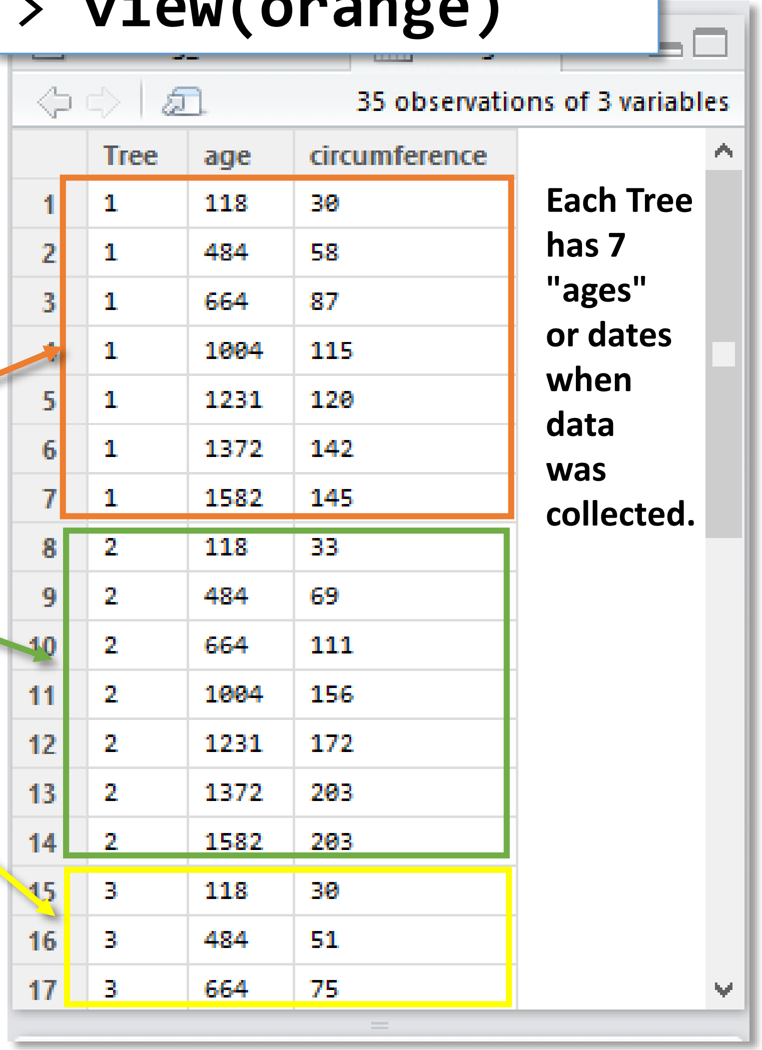
```
> str(orange) # Get structure
'data.frame':   35 obs. of  3 variables:
 $ Tree         : Ord.factor w/ 5 levels "3"<"1"<"5"<"2"<..:
 $ age          : num  118 484 664 1004 1231 ...
 $ circumference: num  30 58 87 115 120 142 145 33 69 111 ...
```

```
# What are values of 'Tree' factor?
> levels(orange$Tree)
[1] "3" "1" "5" "2" "4"
```

7 "ages" or data collection dates:

```
 Tree.age         Tree.date
1       118       1969-04-27
2       484       1970-04-28
3       664       1970-10-25
4      1004       1971-09-30
5      1231       1972-05-14
6      1372       1972-10-02
7      1582       1973-04-30
```

```
# R/Studio Viewer
> view(orange)
```



35 observations of 3 variables

	Tree	age	circumference
1	1	118	30
2	1	484	58
3	1	664	87
4	1	1004	115
5	1	1231	120
6	1	1372	142
7	1	1582	145
8	2	118	33
9	2	484	69
10	2	664	111
11	2	1004	156
12	2	1231	172
13	2	1372	203
14	2	1582	203
15	3	118	30
16	3	484	51
17	3	664	75

Each Tree has 7 "ages" or dates when data was collected.

Data for Tree 1

Data for Tree 2

Data for Tree 3

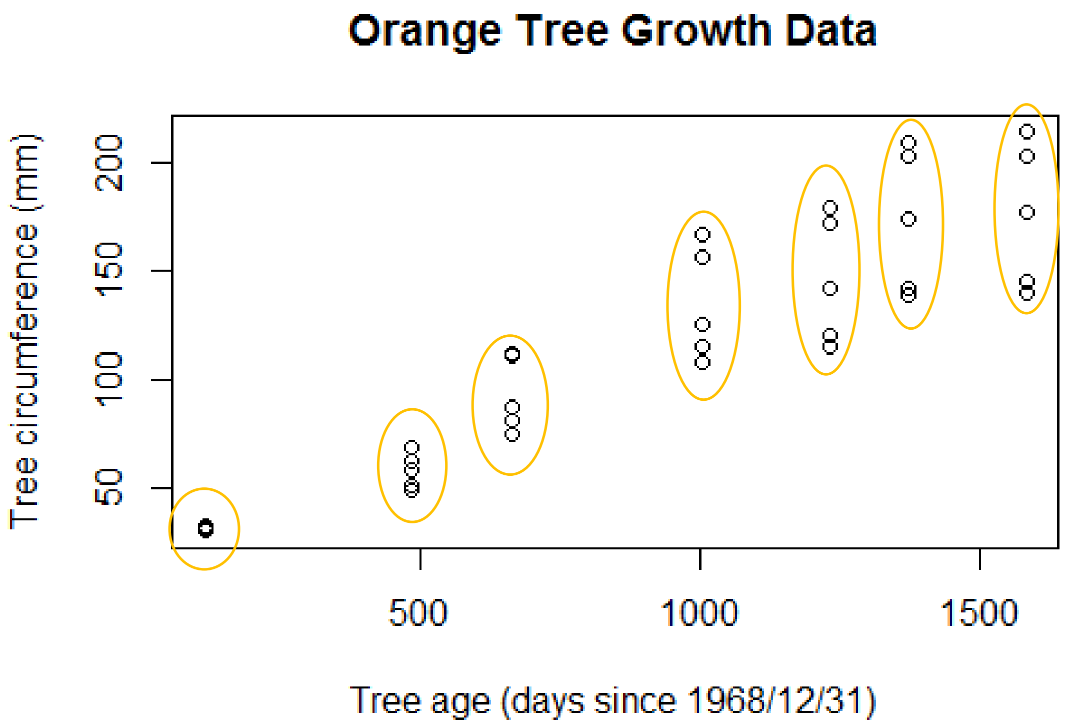
.

.

.

Data for Tree 5

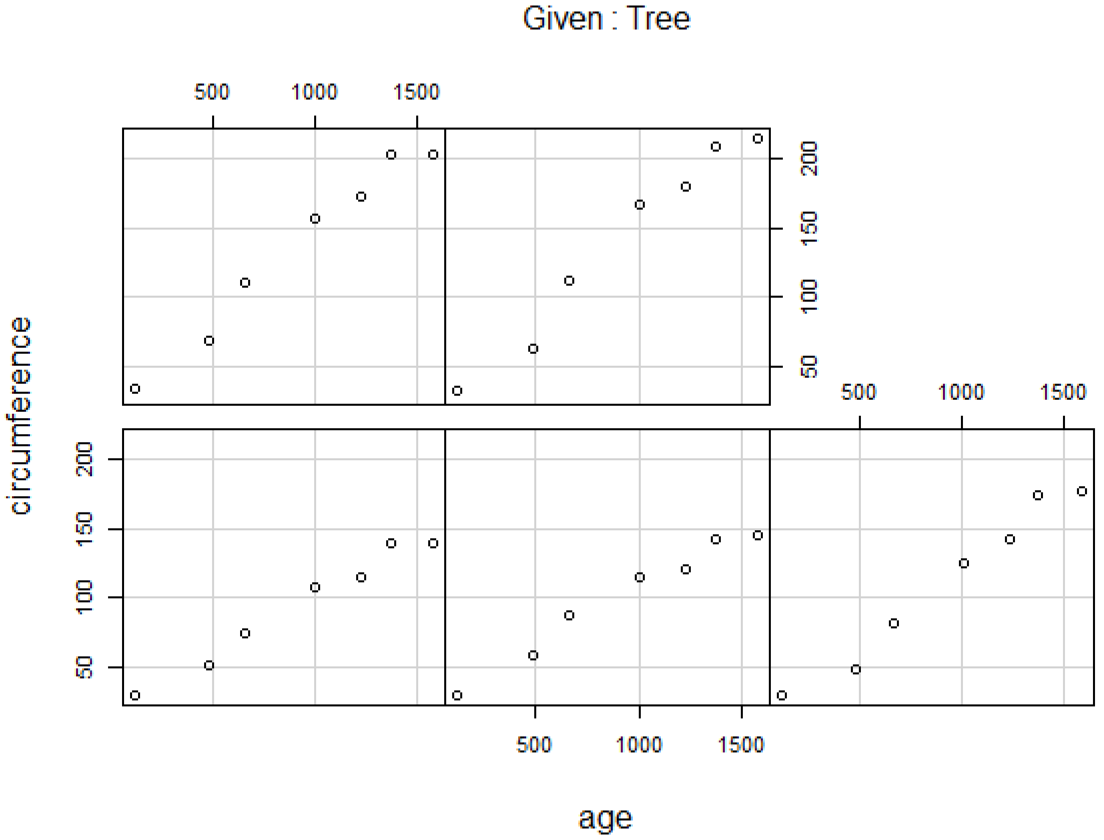
```
attach(orange)  # Can now skip orange$ prefix

# What are min/max values of 'age'?

# Get X-axis range for: age
Xrange <- range(age); Xrange

[1]  118 1582

# Get Y-axis range for: circumference
Yrange <- range(circumference); Yrange

[1]  30 214

# See help("plot"), example("plot")
plot(age, circumference,
  main="Orange Tree Growth",
  xlab="Tree age(days since 1968/12/31)",
  ylab="Tree circumference (mm)",
  xlim= Xrange, ylim= Yrange);
```



What does this mean?

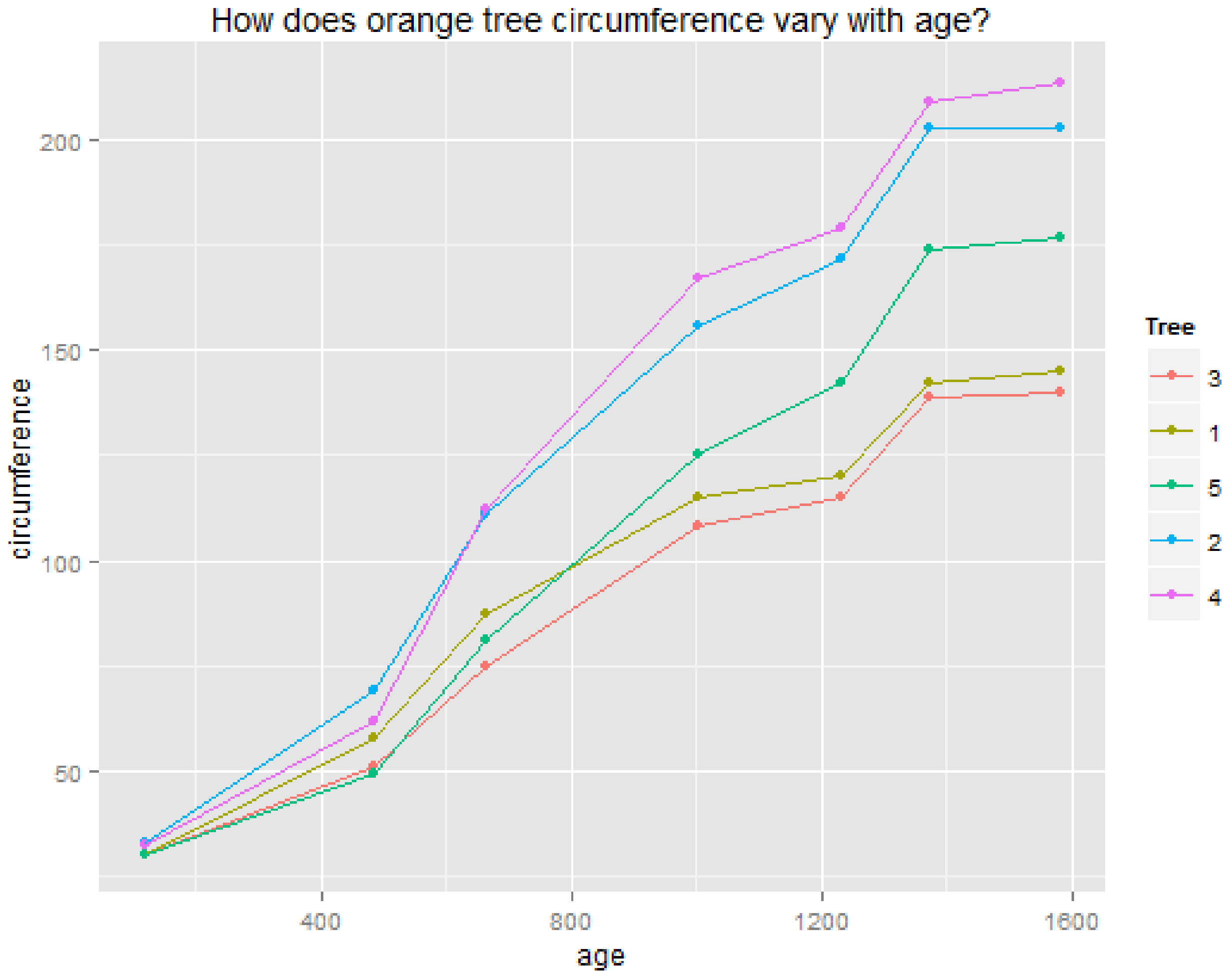
- Data was collected for each Tree at 7 "ages" : 118, 484, 664, 1004, 1231,1372, 1582.
- Each dot is the circumference measurement for a Tree.
- As age increases, circumference also increases.

Other plots of the same data…



From help("orange")

```
# scatter plots for each Tree
require(stats); require(graphics)
coplot(circumference ~ age | Tree,
     data = Orange, show.given = FALSE)
```



From ggplot2-tutorial

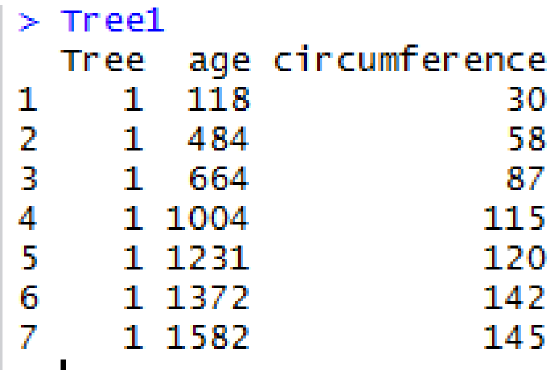
```
library(ggplot2)
qplot(age, circumference, data = Orange,
    geom = c("point", "line"), colour = Tree,
    main = "How does orange tree circumference vary with age?")
```

```
# Get rows from orange where Tree = “1”

# Method 1: Use subset()
Tree1 <-subset(orange, orange$Tree == "1")

# Method 2: Use sqldf()
library(sqldf)
Tree1 <- sqldf("SELECT * FROM orange
       WHERE Tree='1'")

# Compute correlation. See help("cor")
cor(Tree1$age,Tree1$circumference)
[1] 0.9854675
```

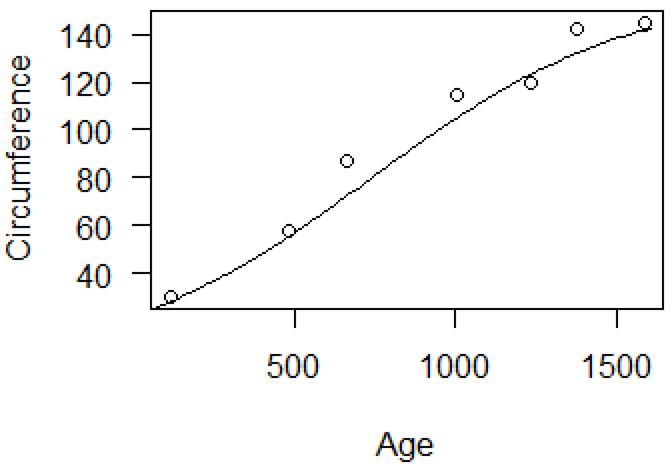
What does this mean?

There is a HIGH POSITIVE correlation between age and circumference.

Method 1 & 2 result in the same data frame:



```
> Tree1
  Tree  age circumference
1    1  118            30
2    1  484            58
3    1  664            87
4    1 1004           115
5    1 1231           120
6    1 1372           142
7    1 1582           145
```



Tree1 fitted model

Problem: Define Covariance/Correlation Chart

Row	Tree	Covariance	Correlation
1	1	cov(age,circumference) where Tree="1"	cor(age,circumference) where Tree="1"
2	2	cov(age,circumference) where Tree="2"	cor(age,circumference) where Tree="2"
3	3	cov(age,circumference) where Tree="3"	cor(age,circumference) where Tree="3"
4	4	cov(age,circumference) where Tree="4"	cor(age,circumference) where Tree="4"
5	5	cov(age,circumference) where Tree="5"	cor(age,circumference) where Tree="5"

“A problem well defined
is a problem half-solved.”
– John Dewey

We'll try 3 methods:

- *Using **subset()** and **merge()**
- *Using **tapply()**
- Using **ddply()**

*Adapted from MyFirstRLesson.R
Patricia Hoffman, PhD.
http://patriciahoffmanphd.com/startr.php

UCSC Extension – Course 262
Data Mining and Machine Learning

SubProblems:

- Get **chart1** for orange
- Get **chart2** for oPlus10 (circumference +10)
- Get **chart3** for oX2 (circumference *2)
- Get **chart4** for oX_2 (circumference * -2)

How to do all of the above without cutting/pasting code for the same functionality but different inputs?

Solution:

Define & re-use function **covcor()**.

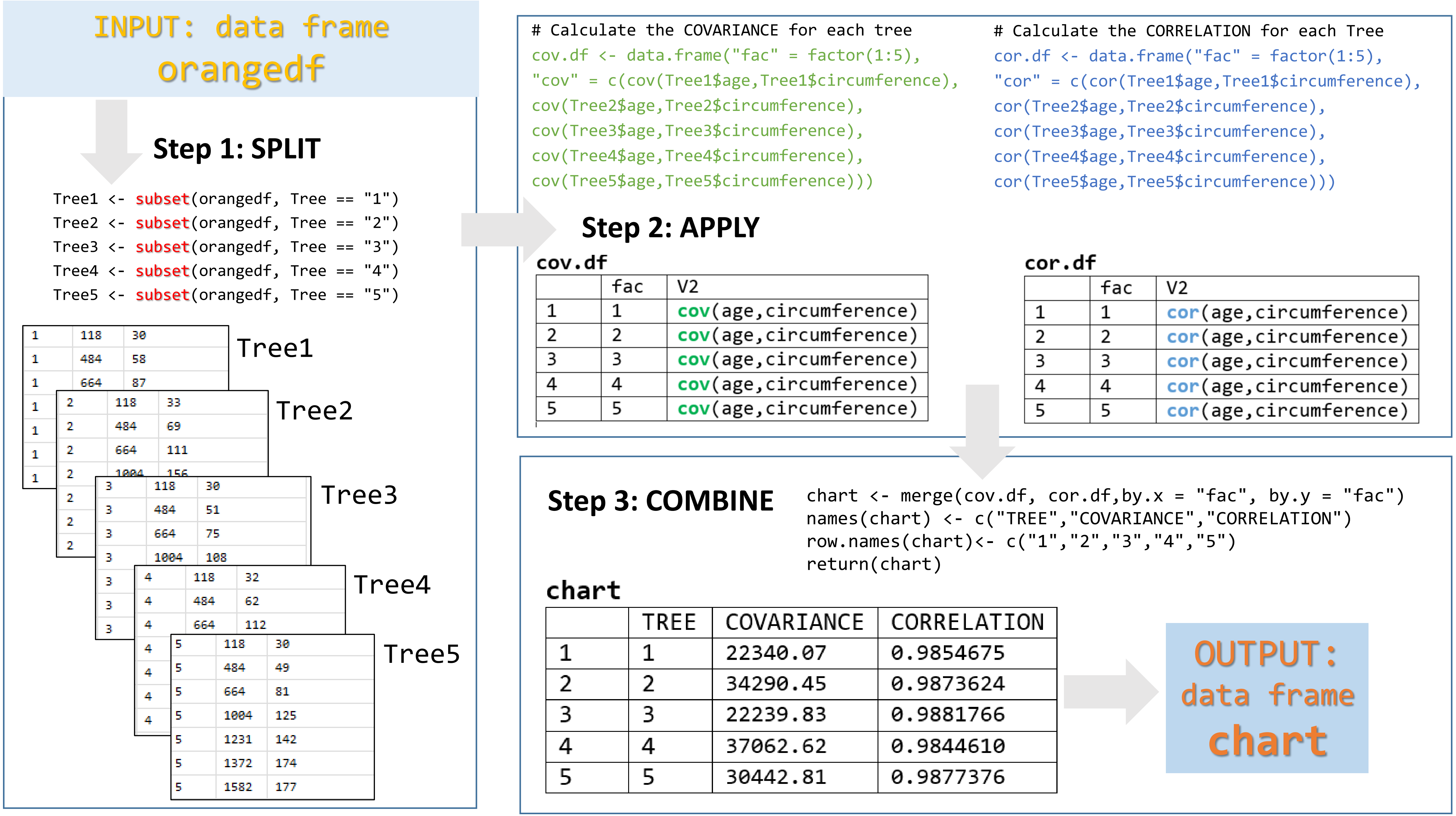
```
chart1 <- covcor(orange)

oPlus10 <-orange
oPlus10$circumference <- oPlus10$circumference+10
chart2 <- covcor(oPlus10)

oX2 <-orange
oX2$circumference <- oX2$circumference*(-2)
chart3 <- covcor(oX2)

oX_2 <-orange
oX_2$circumference <- oX_2$circumference*(-2)
chart4 <- covcor(oX_2)
```

Method 1: Define function **covcor()**

Using **subset()** and **merge()**

```
# FUNCTION covcor:
# INPUT: data frame with Tree, age, circumference
# OUTPUT: data frame with Tree,
Correlation/Covariance

covcor <- function(orangedf)
{
# SPLIT using subset()
Tree1 <- subset(orangedf, Tree == "1")
Tree2 <- subset(orangedf, Tree == "2")
Tree3 <- subset(orangedf, Tree == "3")
Tree4 <- subset(orangedf, Tree == "4")
Tree5 <- subset(orangedf, Tree == "5")

# APPLY functions cov() and cor()
# Calculate COVARIANCE for each Tree
cov.df <- data.frame("fac" = factor(1:5),
"cov" = c(cov(Tree1$age,Tree1$circumference),
cov(Tree2$age,Tree2$circumference),
cov(Tree3$age,Tree3$circumference),
cov(Tree4$age,Tree4$circumference),
cov(Tree5$age,Tree5$circumference)))
```

```
# Calculate CORRELATION for each Tree
cor.df <- data.frame("fac" = factor(1:5),
"cor" = c(cor(Tree1$age,Tree1$circumference),
cor(Tree2$age,Tree2$circumference),
cor(Tree3$age,Tree3$circumference),
cor(Tree4$age,Tree4$circumference),
cor(Tree5$age,Tree5$circumference)))
# COMBINE using merge()
# "Join" cov.df and cor.df using the fac column
chart <- merge(cov.df, cor.df,by.x = "fac", by.y =
"fac")
# Assign column names
names(chart) <- c("TREE","COVARIANCE","CORRELATION")
# Assign row names
row.names(chart)<- c("1","2","3","4","5")
return(chart)
}
```



INPUT: data frame
orangedf

Step 1: SPLIT

```
Tree1 <- subset(orangedf, Tree == "1")
Tree2 <- subset(orangedf, Tree == "2")
Tree3 <- subset(orangedf, Tree == "3")
Tree4 <- subset(orangedf, Tree == "4")
Tree5 <- subset(orangedf, Tree == "5")
```

Tree1

1	118	30
1	484	58
1	664	87
1		
1		
1		
1		

Tree2

2	118	33
2	484	69
2	664	111
2	1004	156
2		
2		
2		

Tree3

3	118	30
3	484	51
3	664	75
3	1004	108
3		
3		
3		

Tree4

4	118	32
4	484	62
4	664	112
4		
4		
4		
4		

Tree5

5	118	30
5	484	49
5	664	81
5	1004	125
5	1231	142
5	1372	174
5	1582	177

```
# Calculate the COVARIANCE for each tree
cov.df <- data.frame("fac" = factor(1:5),
"cov" = c(cov(Tree1$age,Tree1$circumference),
cov(Tree2$age,Tree2$circumference),
cov(Tree3$age,Tree3$circumference),
cov(Tree4$age,Tree4$circumference),
cov(Tree5$age,Tree5$circumference)))
```

```
# Calculate the CORRELATION for each Tree
cor.df <- data.frame("fac" = factor(1:5),
"cor" = c(cor(Tree1$age,Tree1$circumference),
cor(Tree2$age,Tree2$circumference),
cor(Tree3$age,Tree3$circumference),
cor(Tree4$age,Tree4$circumference),
cor(Tree5$age,Tree5$circumference)))
```

Step 2: APPLY

cov.df

	fac	V2
1	1	cov(age,circumference)
2	2	cov(age,circumference)
3	3	cov(age,circumference)
4	4	cov(age,circumference)
5	5	cov(age,circumference)

cor.df

	fac	V2
1	1	cor(age,circumference)
2	2	cor(age,circumference)
3	3	cor(age,circumference)
4	4	cor(age,circumference)
5	5	cor(age,circumference)

Step 3: COMBINE

```
chart <- merge(cov.df, cor.df,by.x = "fac", by.y = "fac")
names(chart) <- c("TREE","COVARIANCE","CORRELATION")
row.names(chart)<- c("1","2","3","4","5")
return(chart)
```

chart

	TREE	COVARIANCE	CORRELATION
1	1	22340.07	0.9854675
2	2	34290.45	0.9873624
3	3	22239.83	0.9881766
4	4	37062.62	0.9844610
5	5	30442.81	0.9877376

OUTPUT:
data frame
chart

Method 2: Define function **covcor()**

Using **tapply(index(array), column, function)+ merge()**

```
covcor <- function(orangedf)
{
# SPLIT/APPLY
agg.cor <-tapply(1:nrow(orangedf),
        orangedf$Tree, FUN=function(x)
        cor(orangedf$age[x],
        orangedf$circumference[x]))
agg.cov <-tapply(1:nrow(orangedf),
        orangedf$Tree, FUN=function(x)
        cov(orangedf$age[x],
        orangedf$circumference[x]))
# Convert arrays to tables, then dfs
agg.cor <-(as.data.frame(as.table(agg.cor)))
agg.cov <-(as.data.frame(as.table(agg.cov)))
# COMBINE
chart <-merge(agg.cov, agg.cor, by=c("Var1"))
colnames(chart) <- c("TREE", "COVARIANCE",
                         "CORRELATION")
return(chart)
}
```

- tapply applies function to elements in index(array) and groups results by column.
- Outputs an array, which must be converted to table, then data frame for merge().
- Similar to SQL "Group By " concept:

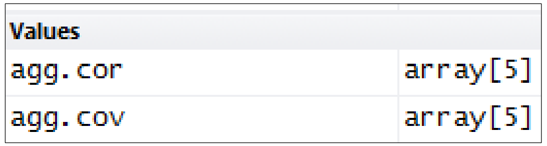
```
CREATE VIEW aggcov AS (SELECT Tree,
*COVAR_POP(age,circumference) AS COVARIANCE FROM
orangedf GROUP BY Tree);

CREATE VIEW aggcor AS (SELECT Tree,
*CORRELATION(age,circumference) AS CORRELATION
FROM orangedf GROUP BY Tree);

CREATE VIEW chart as (SELECT aggcov.Tree as TREE,
COVARIANCE, CORRELATION FROM aggcov, aggcor WHERE
aggcov.Tree = aggcor.Tree);
```

*Currently no built-in CORRELATION functions in SQL.
See: http://www.xarg.org/2012/07/statistical-functions-in-mysql/

INPUT:
data
frame
orangedf

Step 1: SPLIT/APPLY

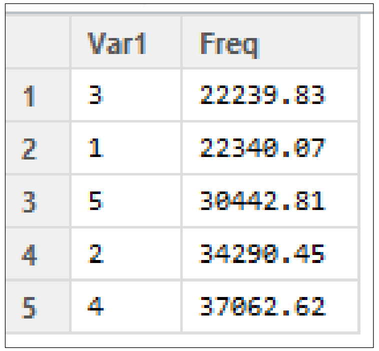
```
agg.cov <-tapply(1:nrow(orangedf),
orangedf$Tree, FUN=function(x)
cov(orangedf$age[x],
orangedf$circumference[x]))
```



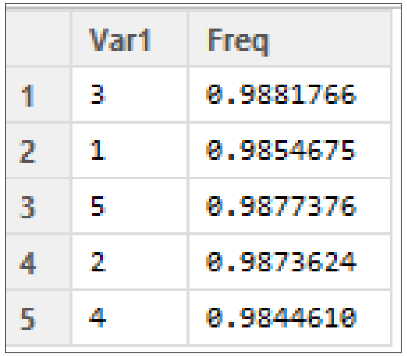
Values	
agg.cor	array[5]
agg.cov	array[5]

```
agg.cor <-tapply(1:nrow(orangedf),
   orangedf$Tree, FUN=function(x)
   cor(orangedf$age[x],
   orangedf$circumference[x]))
```

```
agg.cov <-
(as.data.frame
(as.table(agg.cov)))
```



	Var1	Freq
1	3	22239.83
2	1	22340.07
3	5	30442.81
4	2	34290.45
5	4	37062.62



	Var1	Freq
1	3	0.9881766
2	1	0.9854675
3	5	0.9877376
4	2	0.9873624
5	4	0.9844610

```
agg.cor <-
(as.data.frame
(as.table(agg.cor)))
```

Step 2: COMBINE

```
chart <-merge(agg.cov, agg.cor, by=c("Var1"))
colnames(chart) <- c("TREE","COVARIANCE","CORRELATION")
```

chart

	TREE	COVARIANCE	CORRELATION
1	1	22340.07	0.9854675
2	2	34290.45	0.9873624
3	3	22239.83	0.9881766
4	4	37062.62	0.9844610
5	5	30442.81	0.9877376

OUTPUT:
data frame
chart

Method 3: Define function **`covcor()`**
Using **`ddply(data frame, column, function1, function2)`**

```
library(plyr)
# Define 2 new functions to be used with ddply
ncor <- function(newdf) {chart <- cor(newdf[, 2],
        newdf[, 3]); return(chart)}
ncov <- function(newdf) {chart <- cov(newdf[, 2],
        newdf[, 3]);return(chart)}

# Now use ddply to compute BOTH ncov and ncor:
covcor <- function(odf) {
   chart <- ddply(odf, .(Tree), c("ncov", "ncor"))
   colnames(chart) <- c("TREE", "COVARIANCE", "CORRELATION")
   return(chart) }

# That's it!  no subsetting or merging required!
orange <- as.data.frame(Orange)
o.chart <- covcor(Orange)
```

From help("plyr"):

- **`ddply:`** Split data frame, apply function(s), and return results in a data frame.
- **`plyr`** is a set of tools that implement the **split-apply-combine (SAC)** pattern.

SAC is essential in data analysis. To solve a problem:

- Break it down into smaller pieces.
- Do something to each piece.
- Combine the results back together again.

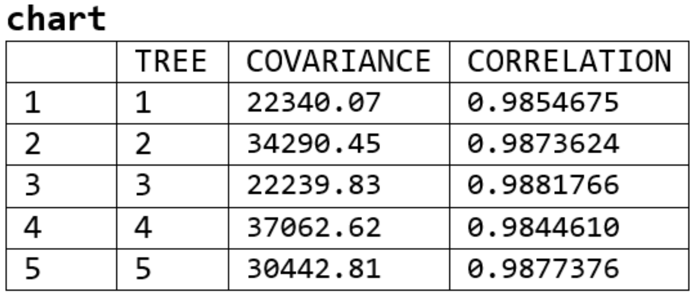
INPUT:
data
frame
odf

Step 0: Define new functions

```
library(plyr)
# Define 2 new functions to be used with ddply
ncor <- function(newdf) {chart <- cor(newdf[, 2], newdf[, 3]); return(chart)}
ncov <- function(newdf) {chart <- cov(newdf[, 2], newdf[, 3]);return(chart)}
```

Step 1: SPLIT/APPLY/COMBINE

```
# Now use ddply to compute BOTH ncov and ncor:
covcor <- function(odf)
   {chart <- ddply(odf, .(Tree), c("ncov", "ncor"))
   colnames(chart) <- c("TREE", "COVARIANCE", "CORRELATION")
   return(chart)}

# That's it!  no subsetting or merging required!
orange <- as.data.frame(Orange)
chart <- covcor(orange)
```

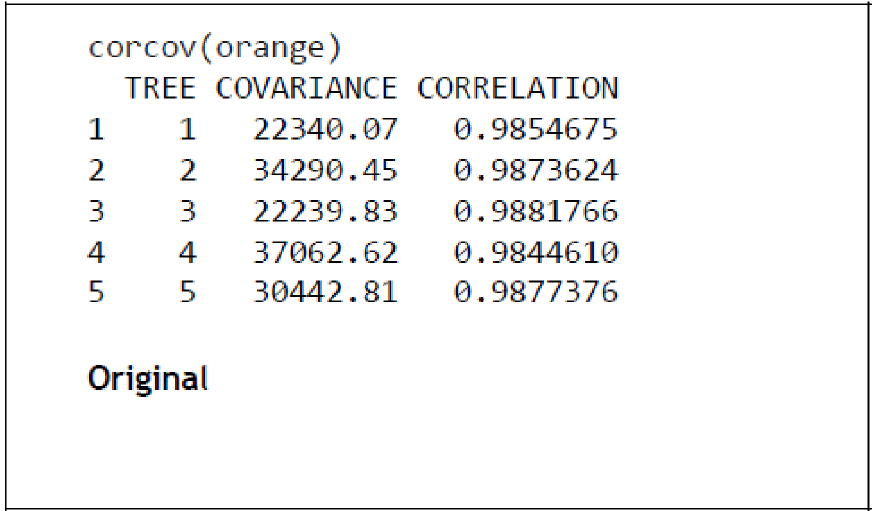


chart

	TREE	COVARIANCE	CORRELATION
1	1	22340.07	0.9854675
2	2	34290.45	0.9873624
3	3	22239.83	0.9881766
4	4	37062.62	0.9844610
5	5	30442.81	0.9877376

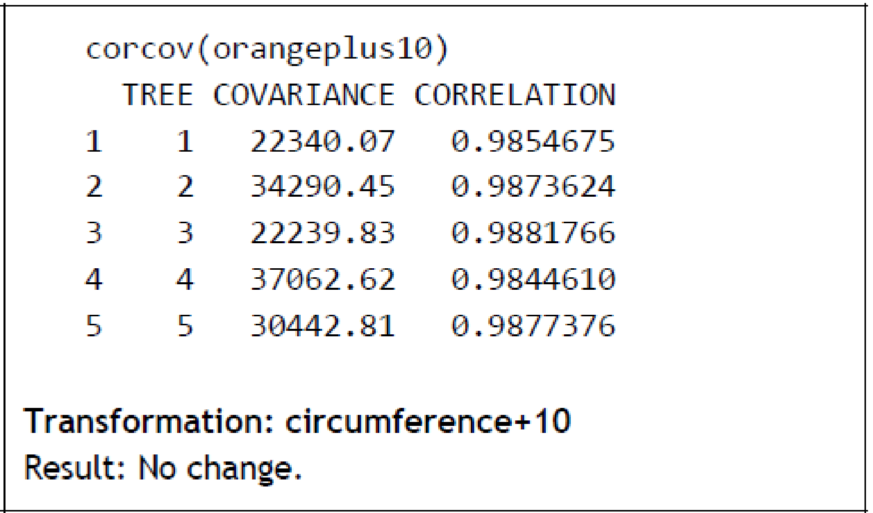
OUTPUT:
data frame
chart

DATA TRANSFORMATIONS: What happens when we change `circumference?`



```
corcov(orange)
  TREE COVARIANCE CORRELATION
1    1   22340.07   0.9854675
2    2   34290.45   0.9873624
3    3   22239.83   0.9881766
4    4   37062.62   0.9844610
5    5   30442.81   0.9877376
```

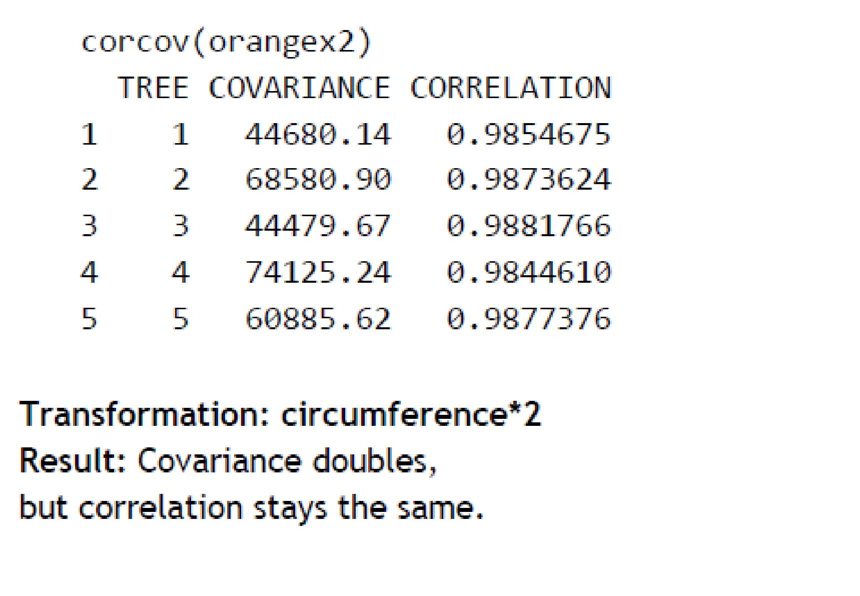
Original



```
corcov(orangeplus10)
  TREE COVARIANCE CORRELATION
1    1   22340.07   0.9854675
2    2   34290.45   0.9873624
3    3   22239.83   0.9881766
4    4   37062.62   0.9844610
5    5   30442.81   0.9877376
```

Transformation: circumference+10

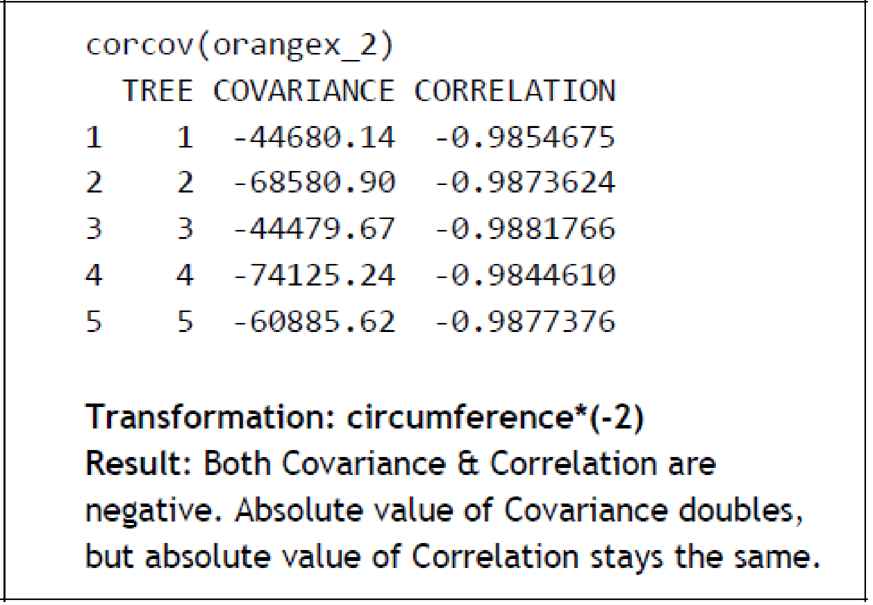
Result: No change.



```
corcov(orangex2)
  TREE COVARIANCE CORRELATION
1    1   44680.14   0.9854675
2    2   68580.90   0.9873624
3    3   44479.67   0.9881766
4    4   74125.24   0.9844610
5    5   60885.62   0.9877376
```

Transformation: circumference*2

Result: Covariance doubles,
but correlation stays the same.



```
corcov(orangex_2)
  TREE COVARIANCE CORRELATION
1    1  -44680.14  -0.9854675
2    2  -68580.90  -0.9873624
3    3  -44479.67  -0.9881766
4    4  -74125.24  -0.9844610
5    5  -60885.62  -0.9877376
```

Transformation: circumference*(-2)

Result: Both Covariance & Correlation are negative. Absolute value of Covariance doubles, but absolute value of Correlation stays the same.

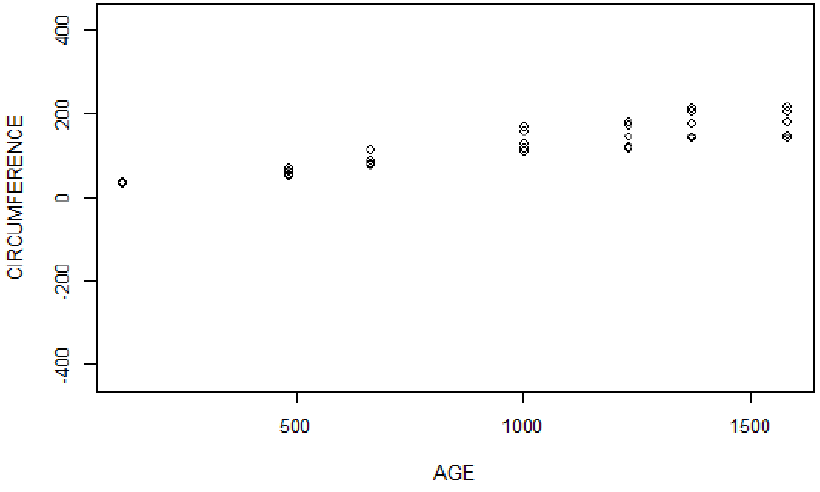
Conclusions:

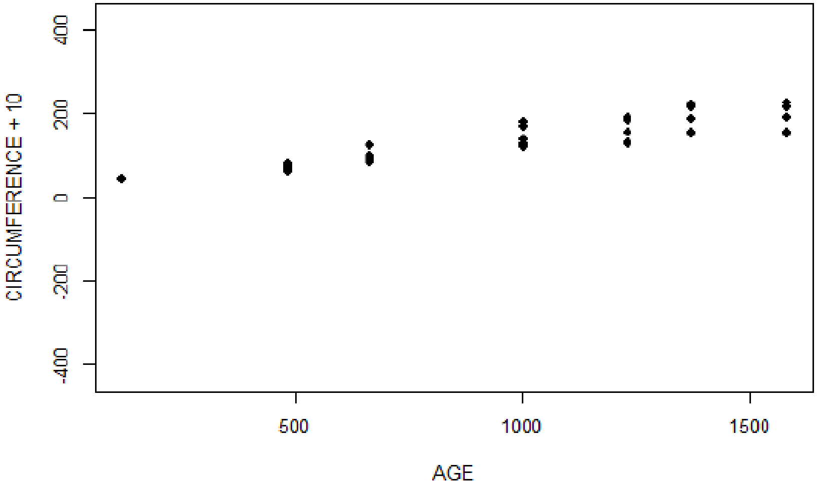
- **Covariance is affected by scale.** Doubling circumference increases covariance (data points "stretch out" to fill more space) but doesn't affect correlation (direction is still positive/upward).

- **Negative correlation/ covariance reverses the direction of the relationship**. I.e., as age increases, the tree circumference "shrinks" instead of getting larger.

DATA VISUALIZATION: What happens when we change `circumference?`

Original

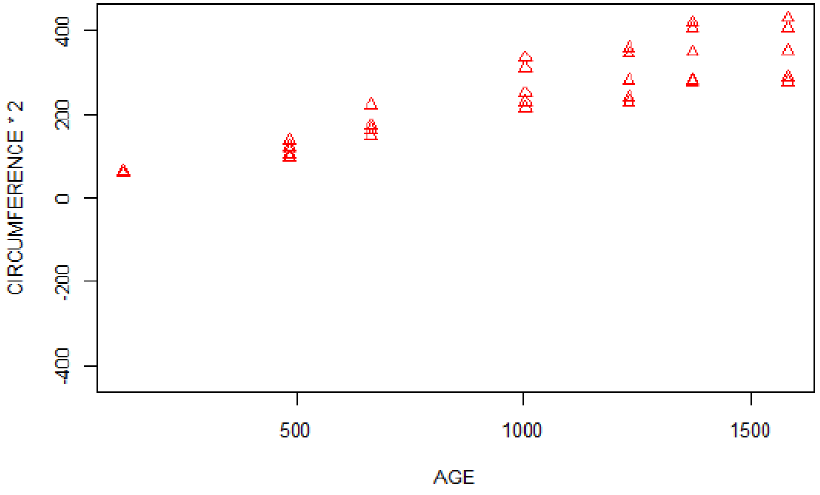


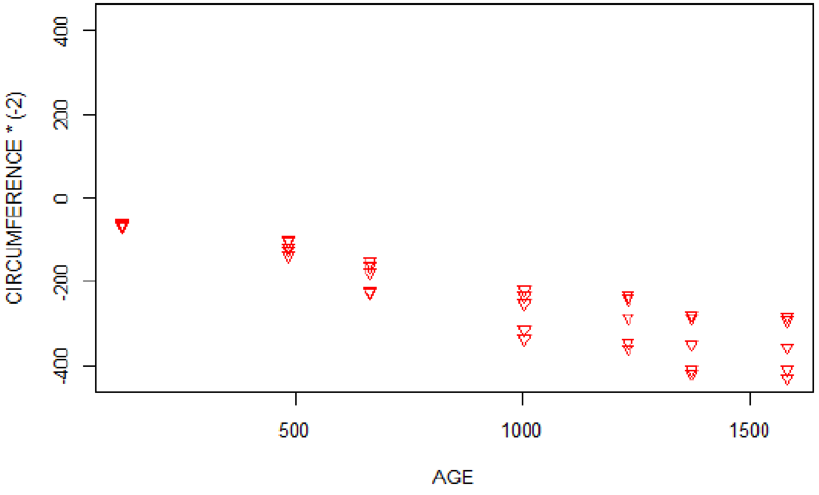


+10:
Minimal change.

X 2:

- **Covariance doubles** (points spread out).
- **Correlation stays the same** (positive relationship.)





X -2:

- **Covariance doubles** (points spread out).
- **Correlation reverses** (negative relationship).

BTW....

7 "ages" or sample dates:

	Tree.age	Tree.date
1	118	1969-04-27
2	484	1970-04-28
3	664	1970-10-25
4	1004	1971-09-30
5	1231	1972-05-14
6	1372	1972-10-02
7	1582	1973-04-30

```
# convert 'age' to a date
secs.per.day <- 24 * 60 * 60
orange$age.sec <- orange$age * secs.per.day
orange$Date <- as.POSIXct(orange$age.sec,
origin = "1968-12-31", tz = "EST")
# Get YYYY-MM-DD
orange$age.date <-substr(orange$Date,1,10)

# Can use getYear, getMonth, getDay,
# getHour, getMin, getSec.
library(gdata)
getYear(orange$Date)
```