# Processing forest inventory measurements

## Setup and connect to DB

```
library(tidyverse)
library(magrittr)
library(RSQLite)
DBname <- "RilieviDendrometria_v03.sqlite"
DBconn <- dbConnect(SQLite(), DBname)
dbListTables(DBconn)
```

```
##  [1] "AdS_Rilevat"       "Altezze"           "Boschi"
##  [4] "Cav_specie"        "Cavallettamento"   "Diradamento"
##  [7] "Rilevatori"        "Rilievi"           "Specie"
## [10] "V_Altezze"         "V_Cavallettamento" "prova"
## [13] "prova2"            "sqlite_sequence"
```

```
v_cav <- dbGetQuery(DBconn, 'select * from V_Cavallettamento')
v_heigths <-
  dbGetQuery(DBconn, 'select * from V_Altezze') %>%
  mutate(h_dendrometrica = as.numeric(h_dendrometrica))
dbDisconnect(DBconn)
```

## Synthesis tables

COULD not find the 'pivot function', is it special for EXPLORATORY?

```
{# r} v_cav %>%   pivot(Cod_bosco + AdS ~ sp, value = Id_ceppaia, fun.aggregate = length)
```

## Initial stand tables

```
v_cav %>%
  group_by(Cod_bosco, AdS, cod_specie) %>%
  summarise(n= n(), dg = sqrt(mean(d130^2))) %>%
  mutate(G = n * dg^2 * pi/40000)
```

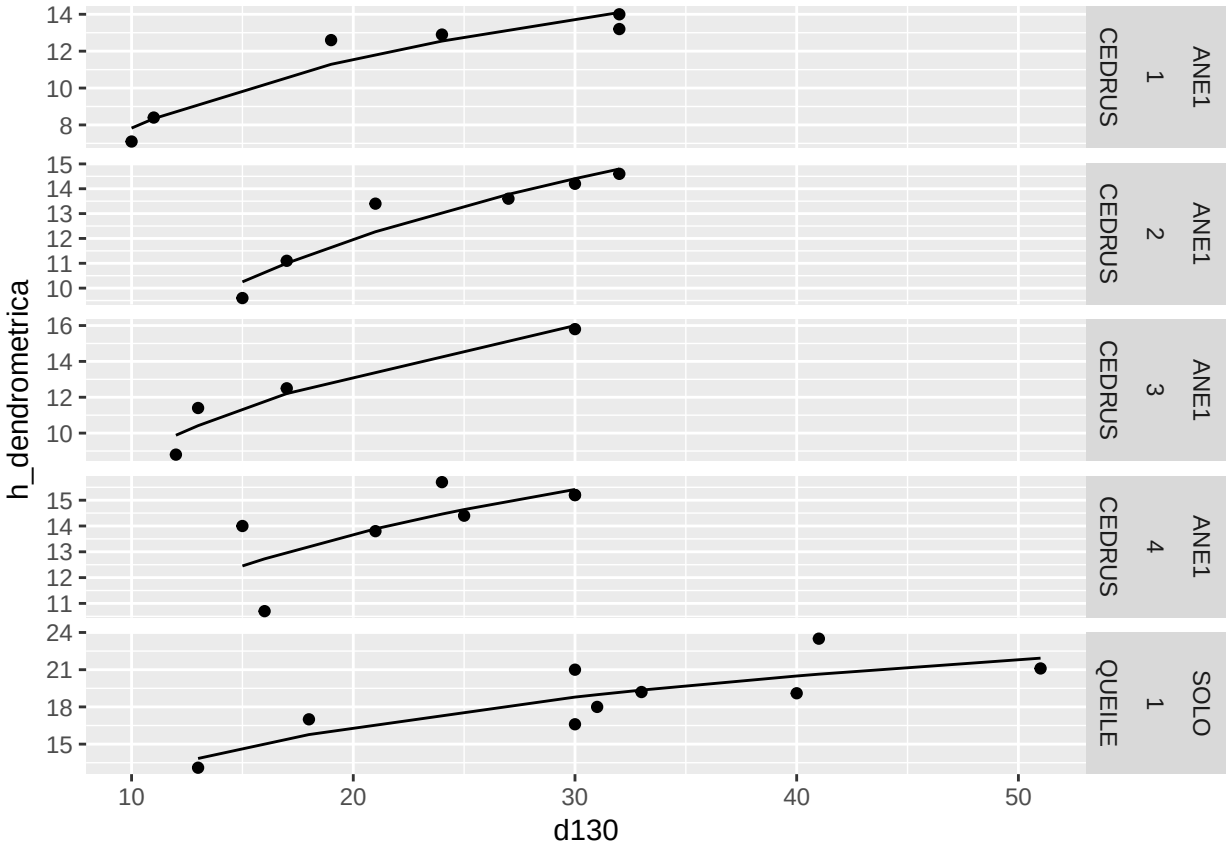
```
## # A tibble: 5 x 6
## # Groups: Cod_bosco, AdS [5]
##   Cod_bosco AdS   cod_specie     n    dg     G
##   <chr>     <chr> <chr>      <int> <dbl> <dbl>
## 1 ANE1      1     CEDRUS       158  18.4  4.22
## 2 ANE1      2     CEDRUS       154  17.8  3.82
## 3 ANE1      3     CEDRUS       153  17.5  3.68
## 4 ANE1      4     CEDRUS       170  20.6  5.69
## 5 SOLO      1     QUEILE        62  28.8  4.05
```

## Estimation of dbh-heigth models

```
h_mods <- v_heigths %>%
  group_by(Cod_bosco, AdS, cod_specie) %>%
    do(fit = lm(h_dendrometrica ~ log(d130), .))

v_heigths %>%
  group_by(Cod_bosco, AdS, cod_specie) %>%
  nest() %>%
  inner_join(h_mods, .) %>%
  mutate(avg_h = list(augment(fit, newdata = data))) %>%
  unnest(avg_h) %>%
  rename(avg_h = .fitted, avg_h.se = .se.fit) %>%
  ggplot() +
  geom_point(aes(d130, h_dendrometrica)) +
  geom_line(aes(d130, avg_h)) +
  facet_grid(Cod_bosco + AdS + cod_specie ~ ., scales = "free")
```

```
## Joining, by = c("Cod_bosco", "AdS", "cod_specie")
```




## Stand tables summary by dbh class

and Estimation of average heigths (by class)

```r
library(broom)
acl <- 3 # dbh class intervals width

standTables1 <- v_cav %>%
  mutate(dbh = d130, d130 = acl*floor(.5 + d130/acl)) %>%
  group_by(Cod_bosco, AdS, cod_specie, d130) %>%
  summarise(frq = n()) %>%
  nest() %>%
  inner_join(h_mods, .) %>%
  mutate(avg_h = list(augment(fit, newdata = data))) %>%
  unnest(avg_h) %>%
  rename(avg_h = .fitted, avg_h.se = .se.fit)
```

```
## Joining, by = c("Cod_bosco", "AdS", "cod_specie")
```

## Estimation of wood resource using INFC volume (tables) functions

[Species matching Quil for QUEILE Piab for CEDRUS]

```r
# Funzione che converte un fattore nel vettore dei corrispondenti livelli
#  (quindi in un vettore di stringhe 'chr', "character")
factor2chr <- function(x) levels(x)[x]


library(ForIT)
#Elenco delle SPECIE e dei corrispondenti codici
INFCspecies <- INFCstats %$%
  unique(data.frame(spg = factor2chr(spg), specie = factor2chr(specie))) %>%
  mutate(rownames = NULL)

lkup <- setNames(c('Quil', 'Piab'), c('QUEILE','CEDRUS'))

standTables2 <- standTables1 %>%
  group_by(Cod_bosco, AdS) %>%
  nest() %>%
  mutate(v = map(data,
                 function(x) as.data.frame(x) %$%
                   INFCvpe(lkup[cod_specie], d130 , avg_h,
                           mod='v', frq, aggr=FALSE)[['mainData']])) %>%
  unnest()
```

```
## Warning in (D_0[1, ] %*% mvc %*% t(t(D_0[1, ]))) + (sa2 * d2h^2): Recycling array of length 1 in arra
##   Use c() or as.vector() instead.

## Warning in (D_0[1, ] %*% mvc %*% t(t(D_0[1, ]))) + (sa2 * d2h^2): Recycling array of length 1 in arra
##   Use c() or as.vector() instead.

## Warning in (D_0[1, ] %*% mvc %*% t(t(D_0[1, ]))) + (sa2 * d2h^2): Recycling array of length 1 in arra
##   Use c() or as.vector() instead.

## Warning in (D_0[1, ] %*% mvc %*% t(t(D_0[1, ]))) + (sa2 * d2h^2): Recycling array of length 1 in arra
##   Use c() or as.vector() instead.
```

```
## Warning in (D_0[1, ] %*% mvc %*% t(t(D_0[1, ]))) + (sa2 * d2h^2): Recycling array of length 1 in arra
##   Use c() or as.vector() instead.

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

```
standTables2 %T>%
  {if (any(.$in.range == 'n')) warning("There are OUTofRANGE values")} %<>%
  filter(!in.range == 'n') %>%
  group_by(Cod_bosco, AdS, cod_specie) %>%
  summarise(n= sum(frq), dg = sqrt(mean(d130^2)), Vol = sum(T_0)/10000) %>%
  mutate(G = n * dg^2 * pi/40000) %>%
  select(Cod_bosco, AdS, cod_specie, n, dg, G, Vol)
```

```
## Warning in function_list[[i]](value): There are OUTofRANGE values

## # A tibble: 5 x 7
## # Groups: Cod_bosco, AdS [5]
##   Cod_bosco AdS   cod_specie     n    dg     G   Vol
##   <chr>     <chr> <chr>      <int> <dbl> <dbl> <dbl>
## 1 ANE1      1     CEDRUS       114  19.4  3.37  2.13
## 2 ANE1      2     CEDRUS       117  19.4  3.46  1.98
## 3 ANE1      3     CEDRUS       118  19.4  3.48  2.08
## 4 ANE1      4     CEDRUS       127  18.3  3.35  3.61
## 5 SOLO      1     QUEILE        60  32.2  4.88  3.85
```