Script para ajuste de regressão linear simples por grupos (chaves) no R

Sollano Rabelo Braga
Marcio Leles Romarco de Oliveira

março, 2016

Contents

1) Baixar e/ou carregar pacotes necessários **2**

2) Carregar os dados **2**

3) Cálculo das variáveis necessárias para a regressão **2**

4) Ajuste de uma equação por uma chave **3**
4.1) R base 3
4.2) dplyr 4
4.3) nlme 8
4.4) Comparar as tabelas geradas 10

5) Ajuste de uma equação por duas ou mais chaves **11**
5.1) R base 11
5.2) dplyr 11
5.3) nlme 12
5.4) Comparar as tabelas geradas 22

6) Exportar tabelas de coeficientes **22**

7) Junção dos dados originais e dados de regressão **22**
7.1) Importar dados 22
7.2) R base 22
7.3) dplyr 23

8) Exportar tabelas finais **24**

1) Baixar e/ou carregar pacotes necessários

```
# install.packages('nlme', dependencies = TRUE)
# install.packages('tidyverse', dependencies = TRUE)
# Para se realizar a instalação, basta remover o # e
# rodar o comando.
```

Para carregar os pacotes, pode-se utilizar do comando library:

```
library(nlme)
library(tidyverse)
library(broom)
```

2) Carregar os dados

```
dados_orig <- read.csv2("dados.csv")
```

Salva-se os dados em um objeto separado, removendo seus NAs

```
dados <- na.omit(dados_orig)
```

3) Cálculo das variáveis necessárias para a regressão

Este passo pode ser feito utilizando o R base ou o pacote dplyr R base

```
dados$DAP <- dados$CAP/pi
dados$INV_DAP <- 1/dados$DAP
dados$LN_HT <- log(dados$HT)
dados$LN_HD <- log(dados$HD)
head(dados)
```

```
##    TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD      DAP    INV_DAP
## 1    3654     101        301 51.1 28.4    Normal 30.45 16.26564 0.06147931
## 5    3654     101        302 51.3 29.0    Normal 30.45 16.32930 0.06123962
## 9    3654     101        204 53.2 28.8    Normal 30.45 16.93409 0.05905249
## 10   3654     101        405 59.0 30.2 Dominante 30.45 18.78028 0.05324733
## 15   3654     101        101 51.0 29.2    Normal 30.45 16.23380 0.06159986
## 16   3654     101        201 59.3 30.7 Dominante 30.45 18.87578 0.05297795
##       LN_HT    LN_HD
## 1  3.346389 3.416086
## 5  3.367296 3.416086
## 9  3.360375 3.416086
## 10 3.407842 3.416086
## 15 3.374169 3.416086
## 16 3.424263 3.416086
```

dplyr Com o dplyr pode-se criar diversas variáveis com um unico comando

```
dados <- mutate(dados, DAP = CAP/pi, INV_DAP = 1/DAP, LN_HT = log(HT),
    LN_HD = log(HD))
head(dados)
```

```
##   TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD      DAP    INV_DAP
## 1   3654     101        301 51.1 28.4    Normal 30.45 16.26564 0.06147931
## 2   3654     101        302 51.3 29.0    Normal 30.45 16.32930 0.06123962
## 3   3654     101        204 53.2 28.8    Normal 30.45 16.93409 0.05905249
## 4   3654     101        405 59.0 30.2 Dominante 30.45 18.78028 0.05324733
## 5   3654     101        101 51.0 29.2    Normal 30.45 16.23380 0.06159986
## 6   3654     101        201 59.3 30.7 Dominante 30.45 18.87578 0.05297795
##      LN_HT    LN_HD
## 1 3.346389 3.416086
## 2 3.367296 3.416086
## 3 3.360375 3.416086
## 4 3.407842 3.416086
## 5 3.374169 3.416086
## 6 3.424263 3.416086
```

4) Ajuste de uma equação por uma chave

Ln(Ht) = b0 + b1*((1/DAP) + b2* Ln(Hd)

4.1) R base

Cria-se um objeto que contem apenas as variáveis da regressão Primeiramente deve se inserir o y, e entao, x1, x2… xn

```
aux1 <- dados[, c("LN_HT", "INV_DAP", "LN_HD")]
```

Utilizando a função by, executa-se a regressão linear no primeiro argumento insere-se as variáveis, no segundo argumento uma variável classificatoria, e no terceiro argumento a função a ser executada (lm)

```
reg_rbase_talh <- by(aux1, dados$TALHAO, lm)
```

O arquivo gerado pela função by comporta-se como uma lista, por isso e necessário utilziar o cbind para unir os seus dados note que so e possivel pedir o sumario individual de cada observação

```
aux2 <- data.frame(cbind(reg_rbase_talh))
summary(aux2)
```

```
##  reg_rbase_talh.Length  reg_rbase_talh.Class  reg_rbase_talh.Mode
##  12     lm     list
##  12     lm     list
##  12     lm     list
##  12     lm     list
##  12     lm     list
##  12     lm     list
##  12     lm     list
##  12     lm     list
```

```
summary(aux2[1, 1]$`3654`)
```

```
##
## Call:
## FUN(formula = data[x, , drop = FALSE])
##
## Residuals:
```

```
##       Min        1Q    Median        3Q       Max
## -0.229947 -0.019780  0.000262  0.020359  0.103337
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.7599     0.3544   2.144   0.0335 *
## INV_DAP      -9.1600     0.6228 -14.709  < 2e-16 ***
## LN_HD         0.9190     0.1032   8.905 9.69e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03639 on 164 degrees of freedom
## Multiple R-squared:  0.6537, Adjusted R-squared:  0.6495
## F-statistic: 154.8 on 2 and 164 DF,  p-value: < 2.2e-16
```

Criação da tabela de coeficientes utiliza-se a função vapply, que e composta por 3 argumentos 1: matriz ou vetor em que se deseja aplicar uma função em cada elemento; 2: função a ser aplicada na matriz ou vetor; neste caso, cria-se uma função que extrái os coeficientes, o r quadrado e o erro do ajuste; 3: template que fornece os nomes das colunas a serem criadas pela função esta função gera uma matriz com duas linhas e n colunas, dependendo do numero de coeficientes. Entao transpoe-se o seu resultado com t()

```
tab_rbase_talh <- t(vapply(reg_rbase_talh, function(x) c(coef(x),
    summary(x)$adj.r.squared, summary(x)$sigma), c(b0 = 0,
    b1 = 0, b2 = 0, Rsqr = 0, Std.Error = 0)))
```

A função by salva a chave na linha do objeto; devido a isso, precisavos uni-lo novamente ao data frame: obs: e importante que um dos elementos desta uniao seja um dataframe, caso contrario todas as variáveis do objeto seram transformadas em fatores Isso ocorre provavelmente pelo fato de utilizar-se os nomes da linhas na função

```
tab_rbase_talh <- cbind(TALHAO = rownames(tab_rbase_talh),
    as.data.frame(tab_rbase_talh))
```

Remoção dos rownames

```
rownames(tab_rbase_talh) <- NULL

head(tab_rbase_talh)
```

```
##   TALHAO        b0        b1        b2      Rsqr  Std.Error
## 1   3654 0.7599015 -9.160042 0.9190132 0.6494627 0.03639290
## 2   3656 0.9734743 -4.431688 0.7659867 0.8193125 0.03018832
## 3   3661 0.7094903 -4.569761 0.8550774 0.5171755 0.03400913
## 4   3662 0.2873602 -7.155051 1.0318020 0.4372779 0.04119515
## 5   3663 0.2947413 -4.877883 0.9905392 0.7327276 0.04232639
## 6   3664 2.4153508 -4.769222 0.2967465 0.4336297 0.02810529
```

4.2) dplyr

Esta e uma forma mais direta de se realizar este procedimento, pois utiliza apenas um pacote, e nao cria nenhum objeto adicional com os dados originais, pode-se criar a tabela de coeficientes diretamente obs: em pipes do dplyr, "." representa o dataframe ate aquele ponto do codigo

```
reg_dplyr_talh <- dados %>% # definição do df a ser utilizado
  group_by(TALHAO) %>% # definição da chave. Uma vantagem e que pode-se utilizar mais de um grupo na fu
  do(Reg = lm(LN_HT ~ INV_DAP + LN_HD, data =.)) # modelo linear
```

o objeto gerado pelo dplyr e mais organizado, e contem a sua chave como primeira coluna, e as regressoes como segunda coluna note que so e possivel pedir o sumario individual de cada observação

```
reg_dplyr_talh
```

```
## Source: local data frame [8 x 2]
## Groups: <by row>
##
## # A tibble: 8 × 2
##   TALHAO      Reg
## *  <int>   <list>
## 1   3654 <S3: lm>
## 2   3656 <S3: lm>
## 3   3661 <S3: lm>
## 4   3662 <S3: lm>
## 5   3663 <S3: lm>
## 6   3664 <S3: lm>
## 7   3665 <S3: lm>
## 8   3666 <S3: lm>
```

```
reg_dplyr_talh$Reg
```

```
## [[1]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      0.7599      -9.1600       0.9190
##
##
## [[2]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      0.9735      -4.4317       0.7660
##
##
## [[3]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      0.7095      -4.5698       0.8551
##
##
## [[4]]
##
## Call:
```

```
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      0.2874      -7.1551       1.0318
##
##
## [[5]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      0.2947      -4.8779       0.9905
##
##
## [[6]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      2.4154      -4.7692       0.2967
##
##
## [[7]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      1.1948      -2.7994       0.6639
##
##
## [[8]]
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Coefficients:
## (Intercept)      INV_DAP        LN_HD
##      1.4981      -4.8679       0.6236
```

```r
summary(reg_dplyr_talh$Reg)
```

```
##      Length Class Mode
## [1,] 12     lm    list
## [2,] 12     lm    list
## [3,] 12     lm    list
## [4,] 12     lm    list
## [5,] 12     lm    list
## [6,] 12     lm    list
## [7,] 12     lm    list
## [8,] 12     lm    list
```

```
summary(reg_dplyr_talh$Reg[[1]])
```

```
##
## Call:
## lm(formula = LN_HT ~ INV_DAP + LN_HD, data = .)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.229947 -0.019780  0.000262  0.020359  0.103337
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.7599     0.3544   2.144   0.0335 *
## INV_DAP      -9.1600     0.6228 -14.709  < 2e-16 ***
## LN_HD         0.9190     0.1032   8.905 9.69e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03639 on 164 degrees of freedom
## Multiple R-squared:  0.6537, Adjusted R-squared:  0.6495
## F-statistic: 154.8 on 2 and 164 DF,  p-value: < 2.2e-16
```

Criação da tabela de coeficientes

```
tab_dplyr_talh <- reg_dplyr_talh %>%
  mutate(b0=coef(Reg)[1], # a variável reg, criada anteriormente, possui os coeficientes na ordem,
         b1=coef(Reg)[2], # por isso os extrai-se com [], na ordem b0(1), b1(2)...bn(n+1)
         b2=coef(Reg)[3],
         Rsqr=summary(Reg)[[9]], # extrai-se r quadrado ajustado do summario de Reg
         Std.Error=summary(Reg)[[6]]) %>% # extrai-se o erro do summario de Reg
  select(-Reg) # agora que extrai-se as variáveis de interesse, remove-se a variável com os ajustes
tab_dplyr_talh
```

```
## Source: local data frame [8 x 6]
## Groups: <by row>
##
## # A tibble: 8 × 6
##   TALHAO        b0        b1        b2      Rsqr  Std.Error
##    <int>     <dbl>     <dbl>     <dbl>     <dbl>      <dbl>
## 1   3654 0.7599015 -9.160042 0.9190132 0.6494627 0.03639290
## 2   3656 0.9734743 -4.431688 0.7659867 0.8193125 0.03018832
## 3   3661 0.7094903 -4.569761 0.8550774 0.5171755 0.03400913
## 4   3662 0.2873602 -7.155051 1.0318020 0.4372779 0.04119515
## 5   3663 0.2947413 -4.877883 0.9905392 0.7327276 0.04232639
## 6   3664 2.4153508 -4.769222 0.2967465 0.4336297 0.02810529
## 7   3665 1.1947932 -2.799395 0.6639431 0.5812417 0.02827865
## 8   3666 1.4981113 -4.867886 0.6236117 0.6075428 0.02618013
```

porém, a grande vantagem do dplyr e que este processo pode ser feita de forma direta, por meio dos pipes

```
tab_dplyr_talh <- dados %>% # definição do df a ser utilizado
  group_by(TALHAO) %>% # definição da chave
  do(Reg = lm(LN_HT ~ INV_DAP + LN_HD, data =.)) %>% # modelo linear
  mutate(b0=coef(Reg)[1], # a variável reg, criada anteriormente, possui os coeficientes na ordem,
         b1=coef(Reg)[2], # por isso os extrai-se com [], na ordem b0(1), b1(2)...bn(n+1)
         b2=coef(Reg)[3],
         Rsqr=summary(Reg)[[9]], # extrai-se r quadrado ajustado do summario de Reg
         Std.Error=summary(Reg)[[6]]) %>% # extrai-se o erro do summario de Reg
  select(-Reg) # agora que extrai-se as variáveis de interesse, remove-se a variável com os ajustes
tab_dplyr_talh
```

```
## Source: local data frame [8 x 6]
## Groups: <by row>
##
## # A tibble: 8 × 6
##   TALHAO        b0        b1        b2      Rsqr  Std.Error
##    <int>     <dbl>     <dbl>     <dbl>     <dbl>      <dbl>
## 1   3654 0.7599015 -9.160042 0.9190132 0.6494627 0.03639290
## 2   3656 0.9734743 -4.431688 0.7659867 0.8193125 0.03018832
## 3   3661 0.7094903 -4.569761 0.8550774 0.5171755 0.03400913
## 4   3662 0.2873602 -7.155051 1.0318020 0.4372779 0.04119515
## 5   3663 0.2947413 -4.877883 0.9905392 0.7327276 0.04232639
## 6   3664 2.4153508 -4.769222 0.2967465 0.4336297 0.02810529
## 7   3665 1.1947932 -2.799395 0.6639431 0.5812417 0.02827865
## 8   3666 1.4981113 -4.867886 0.6236117 0.6075428 0.02618013
```

4.3) nlme

a função lnList comporta-se de maneira similiar a lm, porém pode-se inserir uma chave apos a fórmula; a fórmula e a chave são separadas por | no proximo argumento insere-se o dado a ser utilizado.

```
reg_nlme_talh <- lmList(LN_HT ~ INV_DAP + LN_HD | TALHAO,
    dados)
```

Note que o objeto gerado aqui e o mais organizado dentre as 3 alternativas; Se apenas o objeto for chamado, ele fornece várias informações adicionais, como modelo ajustado, dados, graus de liberdade, e erro do ajuste é possível pedir o sumário do objeto como um todo, ou de observações:

```
reg_nlme_talh
```

```
## Call:
##   Model: LN_HT ~ INV_DAP + LN_HD | TALHAO
##    Data: dados
##
## Coefficients:
##      (Intercept)   INV_DAP     LN_HD
## 3654   0.7599015 -9.160042 0.9190132
## 3656   0.9734743 -4.431688 0.7659867
## 3661   0.7094903 -4.569761 0.8550774
## 3662   0.2873602 -7.155051 1.0318020
## 3663   0.2947413 -4.877883 0.9905392
## 3664   2.4153508 -4.769222 0.2967465
## 3665   1.1947932 -2.799395 0.6639431
## 3666   1.4981113 -4.867886 0.6236117
##
## Degrees of freedom: 1398 total; 1374 residual
## Residual standard error: 0.03409883
```

```r
summary(reg_nlme_talh)
```

```
## Call:
##   Model: LN_HT ~ INV_DAP + LN_HD | TALHAO
##    Data: dados
##
## Coefficients:
##    (Intercept)
##       Estimate Std. Error   t value     Pr(>|t|)
## 3654 0.7599015  0.3320262 2.2886795 2.224918e-02
## 3656 0.9734743  0.1428826 6.8131056 1.424742e-11
## 3661 0.7094903  0.2567286 2.7635806 5.793421e-03
## 3662 0.2873602  0.4149363 0.6925407 4.887149e-01
## 3663 0.2947413  0.1218988 2.4179188 1.573895e-02
## 3664 2.4153508  0.2779620 8.6894984 1.017107e-17
## 3665 1.1947932  0.2192492 5.4494763 5.980398e-08
## 3666 1.4981113  0.3474366 4.3118979 1.734095e-05
##    INV_DAP
##       Estimate Std. Error    t value     Pr(>|t|)
## 3654 -9.160042  0.5835032 -15.698358 3.306891e-51
## 3656 -4.431688  0.3470734 -12.768735 2.298794e-35
## 3661 -4.569761  0.5251068  -8.702535 9.122204e-18
## 3662 -7.155051  0.6389707 -11.197776 6.483511e-28
## 3663 -4.877883  0.5206944  -9.368034 2.923198e-20
## 3664 -4.769222  0.5457056  -8.739550 6.692023e-18
## 3665 -2.799395  0.5690715  -4.919232 9.736666e-07
## 3666 -4.867886  0.5352853  -9.094003 3.251870e-19
##    LN_HD
##       Estimate Std. Error   t value      Pr(>|t|)
## 3654 0.9190132 0.09669963  9.503792  8.662664e-21
## 3656 0.7659867 0.04363268 17.555345  2.092675e-62
## 3661 0.8550774 0.07763475 11.014106  4.277573e-27
## 3662 1.0318020 0.12401014  8.320303  2.090422e-16
## 3663 0.9905392 0.03693235 26.820364 8.817295e-128
## 3664 0.2967465 0.08780367  3.379659  7.460984e-04
## 3665 0.6639431 0.06785637  9.784537  6.679739e-22
## 3666 0.6236117 0.10497563  5.940538  3.594625e-09
##
## Residual standard error: 0.03409883 on 1374 degrees of freedom
```

```r
summary(reg_nlme_talh$`3654`)
```

```
##
## Call:
## lm(formula = object, data = dat, na.action = na.action)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.229947 -0.019780  0.000262  0.020359  0.103337
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.7599     0.3544   2.144   0.0335 *
## INV_DAP      -9.1600     0.6228 -14.709  < 2e-16 ***
## LN_HD         0.9190     0.1032   8.905 9.69e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03639 on 164 degrees of freedom
## Multiple R-squared:  0.6537, Adjusted R-squared:  0.6495
## F-statistic: 154.8 on 2 and 164 DF,  p-value: < 2.2e-16
```

Obtem-se estas informações a mais pois o objeto criado pela função lmList possui uma classe especial:

```
class(reg_nlme_talh)
```

```
## [1] "lmList"
```

Caso o objetivo seja obter apenas os coeficientes, somente a função lmList ja e suficiente

caso o objetivo seja uma tabela com informações de r2 e erro por grupo, pode-se obter da seguinte forma

Criação da tabela de coeficientes

aqui a procedimento se repete como nos metodos anteriores:

```
tab_nlme_talh <- t(vapply(reg_nlme_talh, function(x) c(coef(x),
    summary(x)$adj.r.squared, summary(x)$sigma), c(b0 = 0,
    b1 = 0, b2 = 0, Rsqr = 0, Std.Error = 0)))
tab_nlme_talh <- cbind(TALHAO = rownames(tab_nlme_talh),
    as.data.frame(tab_nlme_talh))

rownames(tab_nlme_talh) <- NULL

tab_nlme_talh
```

```
##   TALHAO        b0        b1        b2      Rsqr  Std.Error
## 1   3654 0.7599015 -9.160042 0.9190132 0.6494627 0.03639290
## 2   3656 0.9734743 -4.431688 0.7659867 0.8193125 0.03018832
## 3   3661 0.7094903 -4.569761 0.8550774 0.5171755 0.03400913
## 4   3662 0.2873602 -7.155051 1.0318020 0.4372779 0.04119515
## 5   3663 0.2947413 -4.877883 0.9905392 0.7327276 0.04232639
## 6   3664 2.4153508 -4.769222 0.2967465 0.4336297 0.02810529
## 7   3665 1.1947932 -2.799395 0.6639431 0.5812417 0.02827865
## 8   3666 1.4981113 -4.867886 0.6236117 0.6075428 0.02618013
```

4.4) Comparar as tabelas geradas

Pode-se testar se as tabelas geradas são iguais Primeiro argumento Target, segundo argumento Current

```
all.equal(tab_nlme_talh, tab_rbase_talh)
```

```
## [1] TRUE
```

Aqui recebos um comando TRUE, ou seja, as tabelas são idênticas

```
all.equal(tab_nlme_talh, tab_dplyr_talh)
```

```
## [1] "Attributes: < Names: 1 string mismatch >"
## [2] "Attributes: < Length mismatch: comparison on first 2 components >"
## [3] "Attributes: < Component \"class\": Lengths (1, 4) differ (string compare on first 1) >"
## [4] "Attributes: < Component \"class\": 1 string mismatch >"
## [5] "Attributes: < Component 2: Modes: numeric, logical >"
## [6] "Attributes: < Component 2: Lengths: 8, 1 >"
## [7] "Attributes: < Component 2: target is numeric, current is logical >"
## [8] "Component \"TALHAO\": 'current' is not a factor"
```

Aqui recebe-se o aviso de que a variável TALHAO possui a classe diferente Os demais dados são os mesmos, ou seja, os valores estimados são idênticos

5) Ajuste de uma equação por duas ou mais chaves

Ln(Ht) = b0 + b1*((1/DAP)

5.1) R base

O procedimento é o mesmo do anterior, porém deve-se unir as chaves em uma unica variável

```
dados2 <- dados
dados2$TALHAO_PAR <- paste(dados2$TALHAO, dados2$PARCELA,
    sep = "_")

aux1 <- dados[, c("LN_HT", "INV_DAP")]
reg_rbase_talh_par <- by(aux1, dados2$TALHAO_PAR, lm)
aux2 <- data.frame(cbind(reg_rbase_talh_par))
```

Criação da tabela de coeficientes Como agora utiliza-se um modelo com 2 coeficientes, o terceiro argumento o função vapply e alterado, contendo 2 coeficientes tambem

```
tab_rbase_talh_par <- t(vapply(reg_rbase_talh_par, function(x) c(coef(x),
    summary(x)$adj.r.squared, summary(x)$sigma), c(b0 = 0,
    b1 = 0, Rsqr = 0, Std.Error = 0)))
tab_rbase_talh_par <- cbind(TALHAO_PAR = rownames(tab_rbase_talh_par),
    as.data.frame(tab_rbase_talh_par))
```

Remoção dos nomes das linhas

```
row.names(tab_rbase_talh_par) <- NULL

head(tab_rbase_talh_par)
```

```
##   TALHAO_PAR       b0         b1         Rsqr  Std.Error
## 1   3654_101 3.803258  -7.269544  0.678026756 0.01744684
## 2   3654_102 4.174023 -13.750464  0.690768894 0.02398376
## 3   3654_103 3.851391  -8.439370  0.360676662 0.04460684
## 4   3654_104 3.656796  -5.195365  0.172643512 0.03145268
## 5   3654_105 3.720361  -6.194041  0.631096864 0.01824305
## 6   3654_106 3.465344  -1.870015 -0.008799552 0.02687367
```

5.2) dplyr

Nao ha a necessidade de criar uma chave adicional, basta informar as chaves desejadas na função group_by() Isto torna este metodo bem pratico

```
tab_dplyr_talh_par <- dados %>%
  group_by(TALHAO, PARCELA) %>%
  do(Reg = lm(LN_HT ~ INV_DAP, data =.)) %>%
  mutate(b0=coef(Reg)[1], # a variável reg, criada anteriormente, possui os coeficientes na ordem,
         b1=coef(Reg)[2], # por isso os extrai-se com [], na ordem b0(1), b1(2)...bn(n+1)
         Rsqr=summary(Reg)[[9]], # extrai-se r quadrado ajustado do summario de Reg
         Std.Error=summary(Reg)[[6]]) %>% # extrai-se o erro do summario de Reg
  select(-Reg) # agora que extrai-se as variáveis de interesse, remove-se a variável com os ajustes
tab_dplyr_talh_par
```

```
## Source: local data frame [153 x 6]
## Groups: <by row>
##
## # A tibble: 153 × 6
##    TALHAO PARCELA       b0         b1         Rsqr  Std.Error
##     <int>   <int>    <dbl>      <dbl>        <dbl>      <dbl>
## 1    3654     101 3.803258  -7.269544  0.678026756 0.01744684
## 2    3654     102 4.174023 -13.750464  0.690768894 0.02398376
## 3    3654     103 3.851391  -8.439370  0.360676662 0.04460684
## 4    3654     104 3.656796  -5.195365  0.172643512 0.03145268
## 5    3654     105 3.720361  -6.194041  0.631096864 0.01824305
## 6    3654     106 3.465344  -1.870015 -0.008799552 0.02687367
## 7    3654     107 4.099547 -13.005030  0.589929762 0.03919004
## 8    3654     109 3.913942 -10.446481  0.747098296 0.03393732
## 9    3654     110 3.818471  -7.115864  0.871018715 0.01441336
## 10   3654     111 3.975365 -11.204591  0.972232836 0.01176976
## # ... with 143 more rows
```

5.3) nlme

Assim como no R base, deve-se criar uma chave que represente as demais

```
dados2 <- dados
dados2$TALHAO_PAR <- paste(dados2$TALHAO, dados2$PARCELA,
    sep = "_")

reg_nlme_talh_par <- lmList(LN_HT ~ INV_DAP | TALHAO_PAR,
    dados2)

reg_nlme_talh_par
```

```
## Call:
##   Model: LN_HT ~ INV_DAP | TALHAO_PAR
##    Data: dados2
##
## Coefficients:
##          (Intercept)      INV_DAP
## 3654_101    3.803258  -7.26954436
## 3654_102    4.174023 -13.75046406
## 3654_103    3.851391  -8.43937046
## 3654_104    3.656796  -5.19536482
## 3654_105    3.720361  -6.19404064
## 3654_106    3.465344  -1.87001532
## 3654_107    4.099547 -13.00502995
## 3654_109    3.913942 -10.44648058
## 3654_110    3.818471  -7.11586406
## 3654_111    3.975365 -11.20459072
## 3654_112    3.854226  -7.56220539
## 3654_113    3.806421  -7.71573029
## 3654_114    3.904536  -9.71360755
## 3654_115    3.991236 -11.07739349
## 3654_116    3.819177  -8.02155530
## 3654_117    4.317853 -17.82388405
## 3654_118    3.914545  -9.91885797
## 3654_119    3.983790 -11.10051034
## 3654_120    3.945747 -11.01117722
## 3656_101    3.280403  -3.56586752
## 3656_102    3.297416  -4.27679893
## 3656_103    3.350325  -5.06884130
## 3656_104    3.370968  -5.60734528
## 3656_105    3.365435  -4.07190484
## 3656_106    3.313194  -3.45169617
## 3656_107    3.673484  -8.94744005
## 3656_108    3.422366  -4.96572083
## 3656_109    3.278438  -4.23227798
## 3656_110    3.531110  -4.84881875
## 3656_111    3.195581  -2.27715561
## 3656_112    3.323071  -3.41407352
## 3656_113    3.440933  -5.38646737
## 3656_114    3.487884  -6.02411785
## 3656_115    3.531326  -6.71521531
## 3656_116    3.434581  -5.70844589
## 3656_117    3.268099  -3.84080574
## 3656_118    3.306716  -3.53823672
## 3656_119    3.452799  -5.15108966
## 3661_101    3.668327  -6.66589823
## 3661_102    3.595788  -5.18400509
## 3661_103    3.749597  -8.08236423
## 3661_104    3.191195   2.01831149
## 3661_105    3.541633  -4.69207759
## 3661_106    3.651321  -6.53094687
## 3661_107    3.682300  -7.11888259
## 3661_108    3.743032  -8.93787403
## 3661_109    3.681956  -8.59798811
## 3661_110    3.458519  -3.68707705
## 3661_111    3.607566  -7.70361785
## 3661_112    3.462440  -3.43685561
## 3661_113    3.542485  -5.55287608
## 3661_114    3.379250  -2.24169449
## 3661_115    3.470682  -3.55491147
## 3661_116    3.819819 -10.79470200
## 3661_117    3.463933  -3.70656574
## 3661_118    3.561303  -5.33252189
## 3661_119    3.786790  -8.62134934
## 3661_120    3.816756  -9.27349397
## 3662_101    3.537742  -4.38686957
## 3662_102    3.637222  -5.70194015
## 3662_103    3.785344  -8.71536395
## 3662_104    3.673250  -6.77896166
## 3662_105    3.666948  -6.60281941
## 3662_107    3.618575  -5.92474732
## 3662_108    3.669810  -6.83892233
## 3662_109    3.787798  -8.47576906
## 3662_110    3.509912  -3.87975479
## 3662_111    3.579801  -4.81589143
## 3662_112    3.708259  -7.09082952
## 3662_113    4.018796 -12.56767103
## 3662_114    3.730046  -7.66666094
## 3662_115    3.490314  -3.61965311
## 3662_116    3.677517  -7.19550904
## 3662_117    3.398929  -1.80452011
## 3662_118    3.997587 -12.68106986
## 3662_119    4.468855 -22.30886822
## 3662_120    3.703560  -7.66702684
## 3662_121    4.258488 -16.85909606
## 3663_101    3.693452  -6.65515350
## 3663_102    3.967179 -11.36743622
## 3663_103    3.755743  -7.59206415
## 3663_104    3.570809  -4.52098979
## 3663_105    3.511527  -3.90269186
## 3663_106    3.503201  -3.56362905
## 3663_107    3.596166  -5.37323684
## 3663_108    3.491873  -3.19830618
## 3663_109    3.509289  -3.41809919
## 3663_110    3.385529  -1.91680430
## 3663_111    3.645423  -6.41174408
## 3663_112    3.340153  -1.46451882
## 3663_113    3.728377  -8.19819570
## 3663_114    3.795330  -9.84313476
## 3663_115    3.360617  -3.10415611
## 3663_116    3.669700  -8.51709837
## 3663_117    3.180533   0.37373379
## 3663_118    3.522450  -6.51859531
## 3663_119    3.399894  -6.04559839
## 3663_120    3.348541  -4.84626149
## 3664_101    3.130750  -2.03053442
## 3664_102    3.721407 -11.18583472
## 3664_103    2.947561   1.46103980
## 3664_104    3.538533  -8.06904237
## 3664_105    3.080943  -0.65453610
## 3664_106    3.399588  -6.44554501
## 3664_107    3.253409  -4.08536626
## 3664_108    3.263134  -4.17261066
## 3664_109    3.205725  -2.63083236
## 3664_110    3.545388  -8.46390766
## 3664_111    3.578459  -9.04270778
## 3664_112    3.373937  -5.58962355
## 3664_113    3.443971  -6.98503148
## 3664_114    3.174037  -1.90755699
## 3664_115    3.367585  -5.12361234
## 3664_116    3.534853  -8.41811017
## 3664_117    3.399891  -6.30670106
## 3664_118    3.535270  -8.31730358
## 3664_119    3.412538  -6.25775187
## 3664_120    3.258112  -3.87718709
## 3664_121    3.202599  -3.14461980
## 3664_122    3.288672  -4.31774864
## 3664_123    2.969839   0.67948820
## 3664_124    3.116912  -1.57953059
## 3665_201    3.063588   0.01848347
## 3665_202    3.202442  -2.07792898
## 3665_203    3.161364  -1.93237818
## 3665_204    3.368155  -4.77109801
## 3665_205    3.293130  -3.58115776
## 3665_206    3.216089  -2.79894938
## 3665_207    3.795376 -11.39320237
## 3665_208    3.158975  -1.49039969
## 3665_209    3.141741  -1.39974566
## 3665_210    3.249327  -3.72421853
## 3665_211    3.349137  -5.39949827
## 3665_212    3.356742  -4.32687144
## 3665_213    3.394083  -5.20578549
## 3665_214    3.232374  -2.51389382
## 3665_215    3.563236  -6.86355299
## 3665_216    3.281534  -2.68501465
## 3666_101    3.100569   3.05553767
## 3666_102    3.407073  -2.23644664
## 3666_103    3.547554  -4.82265531
## 3666_104    3.542821  -5.10143976
## 3666_105    3.401033  -3.17715117
## 3666_106    3.394580  -2.77715494
## 3666_107    3.455664  -3.59964798
## 3666_108    3.610475  -5.82023745
## 3666_109    3.677274  -6.46356605
## 3666_110    3.493255  -3.77128091
## 3666_111    3.492024  -3.91075410
## 3666_112    3.538887  -5.19166823
## 3666_113    3.680672  -7.30045662
## 3666_114    3.530532  -4.85119023
## 3666_115    3.708619  -8.02925840
##
## Degrees of freedom: 1398 total; 1092 residual
## Residual standard error: 0.02898948
```

```
summary(reg_nlme_talh_par)
```

```
## Call:
##   Model: LN_HT ~ INV_DAP | TALHAO_PAR
##    Data: dados2
##
## Coefficients:
##    (Intercept)
##          Estimate Std. Error  t value      Pr(>|t|)
## 3654_101 3.803258 0.17660429 21.53548  5.139966e-86
## 3654_102 4.174023 0.22642275 18.43465  2.789865e-66
## 3654_103 3.851391 0.13881241 27.74529 1.147956e-128
## 3654_104 3.656796 0.16822956 21.73694  2.416078e-87
## 3654_105 3.720361 0.14009773 26.55547 2.793223e-120
## 3654_106 3.465344 0.11944710 29.01154 1.168706e-137
## 3654_107 4.099547 0.16890553 24.27124 2.023660e-104
## 3654_109 3.913942 0.10505628 37.25567 1.020511e-196
## 3654_110 3.818471 0.11314350 33.74892 1.312900e-171
## 3654_111 3.975365 0.11832894 33.59588 1.648856e-170
## 3654_112 3.854226 0.11557773 33.34748 1.002458e-168
## 3654_113 3.806421 0.12796068 29.74680 6.637924e-143
## 3654_114 3.904536 0.26792246 14.57338  4.296743e-44
## 3654_115 3.991236 0.25992425 15.35538  2.497673e-48
## 3654_116 3.819177 0.16035832 23.81652 2.606762e-101
## 3654_117 4.317853 0.11028062 39.15333 3.355367e-210
## 3654_118 3.914545 0.10209546 38.34201 1.897877e-204
## 3654_119 3.983790 0.08913454 44.69413 7.291690e-249
## 3654_120 3.945747 0.16237517 24.30019 1.281117e-104
## 3656_101 3.280403 0.11703823 28.02847 1.133052e-130
## 3656_102 3.297416 0.09594512 34.36772 4.743917e-176
## 3656_103 3.350325 0.11774669 28.45367 1.088203e-133
## 3656_104 3.370968 0.31156351 10.81952  5.479569e-26
## 3656_105 3.365435 0.20223967 16.64083  1.367198e-55
## 3656_106 3.313194 0.08677634 38.18084 2.649865e-203
## 3656_107 3.673484 0.08880707 41.36477 8.979454e-226
## 3656_108 3.422366 0.16414574 20.84956  1.562866e-81
## 3656_109 3.278438 0.10219845 32.07914 1.287524e-159
## 3656_110 3.531110 0.08123617 43.46721 2.078762e-240
## 3656_111 3.195581 0.10550594 30.28816 8.933983e-147
## 3656_112 3.323071 0.14981547 22.18109  2.746872e-90
## 3656_113 3.440933 0.04000020 86.02290  0.000000e+00
## 3656_114 3.487884 0.11461524 30.43124 8.451403e-148
## 3656_115 3.531326 0.12346035 28.60291 9.461075e-135
## 3656_116 3.434581 0.11503945 29.85568 1.106745e-143
## 3656_117 3.268099 0.06916731 47.24919 2.999235e-266
## 3656_118 3.306716 0.26964055 12.26342  1.749226e-32
## 3656_119 3.452799 0.09531038 36.22690 2.264328e-189
## 3661_101 3.668327 0.11803897 31.07725 1.988215e-152
## 3661_102 3.595788 0.16913447 21.25994  3.303568e-84
## 3661_103 3.749597 0.21074626 17.79200  2.210852e-62
## 3661_104 3.191195 0.30135051 10.58964  5.178673e-25
## 3661_105 3.541633 0.11962799 29.60539 6.792963e-142
## 3661_106 3.651321 0.17185379 21.24667  4.034488e-84
## 3661_107 3.682300 0.20331250 18.11153  2.601062e-64
## 3661_108 3.743032 0.14577522 25.67674 3.887038e-114
## 3661_109 3.681956 0.15584780 23.62533 5.226445e-100
## 3661_110 3.458519 0.17332717 19.95371  8.998513e-76
## 3661_111 3.607566 0.21697353 16.62676  1.648935e-55
## 3661_112 3.462440 0.13547907 25.55701 2.647725e-113
## 3661_113 3.542485 0.10263653 34.51486 4.172130e-177
## 3661_114 3.379250 0.05067354 66.68668  0.000000e+00
## 3661_115 3.470682 0.32161971 10.79126  7.237207e-26
## 3661_116 3.819819 0.16095828 23.73173 9.862158e-101
## 3661_117 3.463933 0.13308488 26.02800 1.378899e-116
## 3661_118 3.561303 0.14961149 23.80367 3.189255e-101
## 3661_119 3.786790 0.13361081 28.34195 6.764868e-133
## 3661_120 3.816756 0.12378460 30.83385 1.104509e-150
## 3662_101 3.537742 0.21001816 16.84493  8.926713e-57
## 3662_102 3.637222 0.10669527 34.08982 4.685625e-174
## 3662_103 3.785344 0.19439373 19.47256  9.998037e-73
## 3662_104 3.673250 0.10610360 34.61946 7.411273e-178
## 3662_105 3.666948 0.11353926 32.29674 3.526473e-161
## 3662_107 3.618575 0.23280305 15.54351  2.277240e-49
## 3662_108 3.669810 0.11743879 31.24871 1.172296e-153
## 3662_109 3.787798 0.12338804 30.69826 1.034333e-149
## 3662_110 3.509912 0.19399124 18.09315  3.362210e-64
## 3662_111 3.579801 0.19378576 18.47298  1.624334e-66
## 3662_112 3.708259 0.17768849 20.86944  1.161087e-81
## 3662_113 4.018796 0.23707742 16.95141  2.133701e-57
## 3662_114 3.730046 0.34535313 10.80067  6.597134e-26
## 3662_115 3.490314 0.15405151 22.65680  1.823129e-93
## 3662_116 3.677517 0.21497796 17.10649  2.628999e-58
## 3662_117 3.398929 0.13056862 26.03175 1.298215e-116
## 3662_118 3.997587 0.17497111 22.84713  9.606114e-95
## 3662_119 4.468855 0.14621066 30.56450 9.392098e-149
## 3662_120 3.703560 0.19484391 19.00783  8.086147e-70
## 3662_121 4.258488 0.21365359 19.93174  1.241545e-75
## 3663_101 3.693452 0.19610447 18.83410  9.679119e-69
## 3663_102 3.967179 0.10426886 38.04759 2.346249e-202
## 3663_103 3.755743 0.18439897 20.36748  2.030319e-78
## 3663_104 3.570809 0.11361899 31.42792 6.074509e-155
## 3663_105 3.511527 0.21114288 16.63105  1.557400e-55
## 3663_106 3.503201 0.21146216 16.56656  3.672215e-55
## 3663_107 3.596166 0.16238205 22.14633  4.678742e-90
## 3663_108 3.491873 0.22863529 15.27268  7.116107e-48
## 3663_109 3.509289 0.22678192 15.47429  5.508596e-49
## 3663_110 3.385529 0.09645123 35.10094 2.612637e-181
## 3663_111 3.645423 0.28103372 12.97148  6.853175e-36
## 3663_112 3.340153 0.16075534 20.77786  4.560106e-81
## 3663_113 3.728377 0.15764863 23.64992 3.555856e-100
## 3663_114 3.795330 0.11789452 32.19259 1.973387e-160
## 3663_115 3.360617 0.12204197 27.53657 3.435483e-127
## 3663_116 3.669700 0.08135495 45.10728 1.073577e-251
## 3663_117 3.180533 0.20904929 15.21427  1.487381e-47
## 3663_118 3.522450 0.09713424 36.26373 1.234718e-189
## 3663_119 3.399894 0.10881397 31.24501 1.246080e-153
## 3663_120 3.348541 0.13074021 25.61218 1.094110e-113
## 3664_101 3.130750 0.16254211 19.26116  2.122690e-71
## 3664_102 3.721407 0.22646594 16.43253  2.170364e-54
## 3664_103 2.947561 0.23992259 12.28547  1.376903e-32
## 3664_104 3.538533 0.15469227 22.87466  6.270931e-95
## 3664_105 3.080943 0.13527519 22.77538  2.916683e-94
## 3664_106 3.399588 0.23522529 14.45248  1.883502e-43
## 3664_107 3.253409 0.15790816 20.60317  6.155496e-80
## 3664_108 3.263134 0.09117877 35.78831 3.111337e-186
## 3664_109 3.205725 0.21497321 14.91221  6.540907e-46
## 3664_110 3.545388 0.13617706 26.03513 1.229374e-116
## 3664_111 3.578459 0.18778713 19.05593  4.058560e-70
## 3664_112 3.373937 0.21379662 15.78106  1.078041e-50
## 3664_113 3.443971 0.20285877 16.97719  1.507636e-57
## 3664_114 3.174037 0.22870939 13.87804  1.883042e-40
## 3664_115 3.367585 0.29285965 11.49897  5.752217e-29
## 3664_116 3.534853 0.19087847 18.51887  8.496213e-67
## 3664_117 3.399891 0.14037927 24.21933 4.593018e-104
## 3664_118 3.535270 0.10334576 34.20818 6.625147e-175
## 3664_119 3.412538 0.25034163 13.63153  3.439631e-39
## 3664_120 3.258112 0.14544098 22.40161  9.305862e-92
## 3664_121 3.202599 0.09619541 33.29264 2.482777e-168
## 3664_122 3.288672 0.15307981 21.48338  1.131154e-85
## 3664_123 2.969839 0.15375991 19.31478  9.794778e-72
## 3664_124 3.116912 0.20494298 15.20868  1.596004e-47
## 3665_201 3.063588 0.13184453 23.23637  2.274225e-97
## 3665_202 3.202442 0.07790453 41.10726 5.724933e-224
## 3665_203 3.161364 0.10308040 30.66891 1.678344e-149
## 3665_204 3.368155 0.21985821 15.31967  3.927077e-48
## 3665_205 3.293130 0.10913724 30.17421 5.839334e-146
## 3665_206 3.216089 0.09927047 32.39724 6.693980e-162
## 3665_207 3.795376 0.21112656 17.97678  1.702377e-63
## 3665_208 3.158975 0.20099891 15.71638  2.480822e-50
## 3665_209 3.141741 0.19271583 16.30245  1.207035e-53
## 3665_210 3.249327 0.16306465 19.92662  1.338353e-75
## 3665_211 3.349137 0.11288823 29.66772 2.437347e-142
## 3665_212 3.356742 0.15110333 22.21488  1.636755e-90
## 3665_213 3.394083 0.14874531 22.81808  1.506066e-94
## 3665_214 3.232374 0.13584931 23.79382 3.722769e-101
## 3665_215 3.563236 0.26637085 13.37697  6.647932e-38
## 3665_216 3.281534 0.22258395 14.74290  5.336258e-45
## 3666_101 3.100569 0.11509371 26.93952 5.584985e-123
## 3666_102 3.407073 0.31433790 10.83889  4.527050e-26
## 3666_103 3.547554 0.07483512 47.40493 2.675243e-267
## 3666_104 3.542821 0.11340806 31.23959 1.362736e-153
## 3666_105 3.401033 0.14123630 24.08045 4.103903e-103
## 3666_106 3.394580 0.14875882 22.81935  1.476798e-94
## 3666_107 3.455664 0.16010232 21.58409  2.460218e-86
## 3666_108 3.610475 0.09438063 38.25441 7.952944e-204
## 3666_109 3.677274 0.12566711 29.26202 1.915881e-139
## 3666_110 3.493255 0.15140994 23.07150  2.956717e-96
## 3666_111 3.492024 0.14418027 24.21985 4.555389e-104
## 3666_112 3.538887 0.12651533 27.97200 2.847518e-130
## 3666_113 3.680672 0.06588093 55.86854 1.877449e-322
## 3666_114 3.530532 0.09261529 38.12040 7.125503e-203
## 3666_115 3.708619 0.15227147 24.35531 5.361141e-105
##    INV_DAP
##              Estimate Std. Error      t value     Pr(>|t|)
## 3654_101  -7.26954436  3.0444976 -2.387764828 1.712042e-02
## 3654_102 -13.75046406  4.0747938 -3.374517733 7.654368e-04
## 3654_103  -8.43937046  2.4653975 -3.423127727 6.420083e-04
## 3654_104  -5.19536482  2.8226169 -1.840619873 6.594847e-02
## 3654_105  -6.19404064  2.4307451 -2.548206607 1.096386e-02
## 3654_106  -1.87001532  2.0915404 -0.894085188 3.714733e-01
## 3654_107 -13.00502995  2.8913281 -4.497943295 7.595020e-06
## 3654_109 -10.44648058  1.7979414 -5.810245421 8.177418e-09
## 3654_110  -7.11586406  1.9294113 -3.688101117 2.370566e-04
## 3654_111 -11.20459072  1.8995146 -5.898659829 4.882426e-09
## 3654_112  -7.56220539  1.9636132 -3.851168457 1.243896e-04
## 3654_113  -7.71573029  2.1894606 -3.524032504 4.426261e-04
## 3654_114  -9.71360755  4.6192343 -2.102860967 3.570632e-02
## 3654_115 -11.07739349  4.6418610 -2.386412172 1.718320e-02
## 3654_116  -8.02155530  2.7522868 -2.914505617 3.635202e-03
## 3654_117 -17.82388405  1.9492317 -9.144055926 2.862466e-19
## 3654_118  -9.91885797  1.8216907 -5.444864017 6.401515e-08
## 3654_119 -11.10051034  1.4971150 -7.414601091 2.441712e-13
## 3654_120 -11.01117722  2.8216191 -3.902432142 1.010574e-04
## 3656_101  -3.56586752  1.9829814 -1.798235446 7.241581e-02
## 3656_102  -4.27679893  1.5148291 -2.823288070 4.839738e-03
## 3656_103  -5.06884130  1.9117052 -2.651476442 8.130113e-03
## 3656_104  -5.60734528  5.6061638 -1.000210755 3.174301e-01
## 3656_105  -4.07190484  3.3676033 -1.209140292 2.268707e-01
## 3656_106  -3.45169617  1.3537614 -2.549708016 1.091705e-02
## 3656_107  -8.94744005  1.3804550 -6.481514964 1.371913e-10
## 3656_108  -4.96572083  2.6992177 -1.839688867 6.608523e-02
## 3656_109  -4.23227798  1.5741547 -2.688603512 7.284379e-03
## 3656_110  -4.84881875  1.5010670 -3.230248051 1.273729e-03
## 3656_111  -2.27715561  1.5164855 -1.501600606 1.334894e-01
## 3656_112  -3.41407352  2.4649439 -1.385051225 1.663197e-01
## 3656_113  -5.38646737  0.5677600 -9.487226301 1.432622e-20
## 3656_114  -6.02411785  1.7768820 -3.390274499 7.231973e-04
## 3656_115  -6.71521531  1.9313025 -3.477039583 5.269287e-04
## 3656_116  -5.70844589  1.8076006 -3.158023955 1.631988e-03
## 3656_117  -3.84080574  1.0536843 -3.645119980 2.798482e-04
## 3656_118  -3.53823672  4.1757466 -0.847330319 3.969967e-01
## 3656_119  -5.15108966  1.4765795 -3.488528477 5.050350e-04
## 3661_101  -6.66589823  2.0014698 -3.330501609 8.958941e-04
## 3661_102  -5.18400509  3.0141163 -1.719908802 8.573236e-02
## 3661_103  -8.08236423  3.5712937 -2.263147464 2.382229e-02
## 3661_104   2.01831149  5.4461304  0.370595514 7.110107e-01
## 3661_105  -4.69207759  2.0178247 -2.325314813 2.023764e-02
## 3661_106  -6.53094687  3.1012748 -2.105891075 3.544163e-02
## 3661_107  -7.11888259  3.5301601 -2.016589153 4.398283e-02
## 3661_108  -8.93787403  2.5301144 -3.532596807 4.286909e-04
## 3661_109  -8.59798811  2.6680005 -3.222633578 1.307742e-03
## 3661_110  -3.68707705  3.0097302 -1.225052365 2.208197e-01
## 3661_111  -7.70361785  3.8310961 -2.010812979 4.459065e-02
## 3661_112  -3.43685561  2.2574003 -1.522483868 1.281774e-01
## 3661_113  -5.55287608  1.8618858 -2.982393493 2.923529e-03
## 3661_114  -2.24169449  0.8019701 -2.795234333 5.277071e-03
## 3661_115  -3.55491147  5.7640139 -0.616742350 5.375332e-01
## 3661_116 -10.79470200  2.8733996 -3.756770162 1.812149e-04
## 3661_117  -3.70656574  2.2619793 -1.638638219 1.015767e-01
## 3661_118  -5.33252189  2.5056838 -2.128170295 3.354641e-02
## 3661_119  -8.62134934  2.3040607 -3.741806541 1.922089e-04
## 3661_120  -9.27349397  2.1309370 -4.351838734 1.476569e-05
## 3662_101  -4.38686957  3.8152821 -1.149815257 2.504717e-01
## 3662_102  -5.70194015  1.8895173 -3.017670234 2.606304e-03
## 3662_103  -8.71536395  3.4602593 -2.518702540 1.192079e-02
## 3662_104  -6.77896166  1.8057672 -3.754061739 1.831599e-04
## 3662_105  -6.60281941  1.8674198 -3.535798073 4.235882e-04
## 3662_107  -5.92474732  4.0102512 -1.477400537 1.398566e-01
## 3662_108  -6.83892233  1.9849565 -3.445376461 5.919373e-04
## 3662_109  -8.47576906  2.0492409 -4.136052976 3.804400e-05
## 3662_110  -3.87975479  3.1696640 -1.224027140 2.212060e-01
## 3662_111  -4.81589143  3.2616954 -1.476499439 1.400982e-01
## 3662_112  -7.09082952  2.9812220 -2.378497670 1.755455e-02
## 3662_113 -12.56767103  4.0126330 -3.132026065 1.782217e-03
## 3662_114  -7.66666094  5.8222039 -1.316797047 1.881829e-01
## 3662_115  -3.61965311  2.5629561 -1.412296185 1.581477e-01
## 3662_116  -7.19550904  3.7935977 -1.896750687 5.812453e-02
## 3662_117  -1.80452011  2.1981459 -0.820928278 4.118663e-01
## 3662_118 -12.68106986  3.0185391 -4.201061943 2.873357e-05
## 3662_119 -22.30886822  2.4467503 -9.117754350 3.587360e-19
## 3662_120  -7.66702684  3.2649848 -2.348258066 1.903908e-02
## 3662_121 -16.85909606  3.5563380 -4.740577574 2.412625e-06
## 3663_101  -6.65515350  3.4347981 -1.937567580 5.293358e-02
## 3663_102 -11.36743622  1.7243226 -6.592407036 6.723320e-11
## 3663_103  -7.59206415  2.9887383 -2.540223830 1.121578e-02
## 3663_104  -4.52098979  1.8814853 -2.402883408 1.643241e-02
## 3663_105  -3.90269186  3.7826379 -1.031738147 3.024232e-01
## 3663_106  -3.56362905  3.7299861 -0.955400098 3.395869e-01
## 3663_107  -5.37323684  2.8836739 -1.863330232 6.268410e-02
## 3663_108  -3.19830618  4.1292491 -0.774549093 4.387738e-01
## 3663_109  -3.41809919  3.9790974 -0.859013692 3.905215e-01
## 3663_110  -1.91680430  1.6545307 -1.158518398 2.469058e-01
## 3663_111  -6.41174408  4.9727397 -1.289378575 1.975395e-01
## 3663_112  -1.46451882  2.7528164 -0.532007436 5.948291e-01
## 3663_113  -8.19819570  2.6262858 -3.121593134 1.846006e-03
## 3663_114  -9.84313476  1.9126344 -5.146375427 3.147309e-07
## 3663_115  -3.10415611  1.9881874 -1.561299568 1.187428e-01
## 3663_116  -8.51709837  1.4823286 -5.745756069 1.186018e-08
## 3663_117   0.37373379  3.8987494  0.095859915 9.236494e-01
## 3663_118  -6.51859531  1.8035015 -3.614410744 3.147520e-04
## 3663_119  -6.04559839  2.0060928 -3.013618564 2.641065e-03
## 3663_120  -4.84626149  2.2281187 -2.175046404 2.984065e-02
## 3664_101  -2.03053442  2.4893997 -0.815672301 4.148655e-01
## 3664_102 -11.18583472  3.5417569 -3.158272838 1.630608e-03
## 3664_103   1.46103980  3.7416531  0.390479763 6.962580e-01
## 3664_104  -8.06904237  2.3613009 -3.417202125 6.559941e-04
## 3664_105  -0.65453610  2.1386524 -0.306050723 7.596244e-01
## 3664_106  -6.44554501  3.7899523 -1.700692915 8.928538e-02
## 3664_107  -4.08536626  2.4615040 -1.659703253 9.726129e-02
## 3664_108  -4.17261066  1.4754266 -2.828070637 4.768546e-03
## 3664_109  -2.63083236  3.5690507 -0.737123842 4.612054e-01
## 3664_110  -8.46390766  2.3128083 -3.659580252 2.647019e-04
## 3664_111  -9.04270778  3.0898810 -2.926555320 3.498371e-03
## 3664_112  -5.58962355  3.6212086 -1.543579552 1.229799e-01
## 3664_113  -6.98503148  3.3732669 -2.070702271 3.862117e-02
## 3664_114  -1.90755699  3.9158743 -0.487134371 6.262609e-01
## 3664_115  -5.12361234  4.8885736 -1.048079207 2.948340e-01
## 3664_116  -8.41811017  3.3098532 -2.543348537 1.111657e-02
## 3664_117  -6.30670106  2.2327438 -2.824641573 4.819493e-03
## 3664_118  -8.31730358  1.6488274 -5.044374941 5.325226e-07
## 3664_119  -6.25775187  4.2056268 -1.487947510 1.370534e-01
## 3664_120  -3.87718709  2.3907273 -1.621760526 1.051433e-01
## 3664_121  -3.14461980  1.5226482 -2.065230724 3.913675e-02
## 3664_122  -4.31774864  2.3876635 -1.808357302 7.082590e-02
## 3664_123   0.67948820  2.4531962  0.276980785 7.818473e-01
## 3664_124  -1.57953059  3.3959471 -0.465122265 6.419366e-01
## 3665_201   0.01848347  2.0530011  0.009003148 9.928183e-01
## 3665_202  -2.07792898  1.3175402 -1.577127553 1.150558e-01
## 3665_203  -1.93237818  1.6649379 -1.160630797 2.460457e-01
## 3665_204  -4.77109801  3.4476584 -1.383866229 1.666822e-01
## 3665_205  -3.58115776  1.6352845 -2.189929554 2.874031e-02
## 3665_206  -2.79894938  1.5470535 -1.809213001 7.069282e-02
## 3665_207 -11.39320237  3.5054320 -3.250156437 1.188626e-03
## 3665_208  -1.49039969  3.3048028 -0.450979920 6.520936e-01
## 3665_209  -1.39974566  3.0406093 -0.460350380 6.453564e-01
## 3665_210  -3.72421853  2.4420879 -1.525014104 1.275451e-01
## 3665_211  -5.39949827  1.8481782 -2.921524700 3.554915e-03
## 3665_212  -4.32687144  2.3891969 -1.811015020 7.041322e-02
## 3665_213  -5.20578549  2.2395550 -2.324473134 2.028284e-02
## 3665_214  -2.51389382  2.1381449 -1.175735971 2.399567e-01
## 3665_215  -6.86355299  4.3478622 -1.578604061 1.147165e-01
## 3665_216  -2.68501465  3.4941601 -0.768429203 4.423984e-01
## 3666_101   3.05553767  2.0094982  1.520547596 1.286629e-01
## 3666_102  -2.23644664  5.5473486 -0.403155957 6.869124e-01
## 3666_103  -4.82265531  1.2024659 -4.010638014 6.467463e-05
## 3666_104  -5.10143976  1.8870323 -2.703419350 6.969567e-03
## 3666_105  -3.17715117  2.3837906 -1.332814723 1.828706e-01
## 3666_106  -2.77715494  2.4784533 -1.120519382 2.627389e-01
## 3666_107  -3.59964798  2.5515614 -1.410762856 1.585994e-01
## 3666_108  -5.82023745  1.5463142 -3.763942353 1.761578e-04
## 3666_109  -6.46356605  2.0795972 -3.108085634 1.931713e-03
## 3666_110  -3.77128091  2.6894668 -1.402241088 1.611275e-01
## 3666_111  -3.91075410  2.4882921 -1.571662013 1.163186e-01
## 3666_112  -5.19166823  2.1631746 -2.400022734 1.656070e-02
## 3666_113  -7.30045662  1.0993508 -6.640697932 4.911905e-11
## 3666_114  -4.85119023  1.5256615 -3.179729121 1.515599e-03
## 3666_115  -8.02925840  2.5032982 -3.207471781 1.377982e-03
##
## Residual standard error: 0.02898948 on 1092 degrees of freedom
```

Criação da tabela de coeficientes

```
tab_nlme_talh_par <- t(vapply(reg_nlme_talh_par, function(x) c(coef(x),
    summary(x)$adj.r.squared, summary(x)$sigma), c(b0 = 0,
    b1 = 0, Rsqr = 0, Std.Error = 0)))
tab_nlme_talh_par <- cbind(TALHAO_PAR = rownames(tab_nlme_talh_par),
    as.data.frame(tab_nlme_talh_par))

head(tab_nlme_talh_par)
```

```
##          TALHAO_PAR       b0         b1         Rsqr  Std.Error
## 3654_101   3654_101 3.803258  -7.269544  0.678026756 0.01744684
## 3654_102   3654_102 4.174023 -13.750464  0.690768894 0.02398376
## 3654_103   3654_103 3.851391  -8.439370  0.360676662 0.04460684
## 3654_104   3654_104 3.656796  -5.195365  0.172643512 0.03145268
## 3654_105   3654_105 3.720361  -6.194041  0.631096864 0.01824305
## 3654_106   3654_106 3.465344  -1.870015 -0.008799552 0.02687367
```

5.4) Comparar as tabelas geradas

Pode-se testar se as tabelas geradas são iguais Primeiro argumento Target, segundo argumento Current

```
all.equal(tab_nlme_talh_par, tab_rbase_talh_par)
```

```
## [1] "Attributes: < Component \"row.names\": Modes: character, numeric >"
## [2] "Attributes: < Component \"row.names\": target is character, current is numeric >"
```

Aqui recebos um comando TRUE, ou seja, as tabelas são idênticas Neste caso como a tabela gerada pelo dplyr possui uma coluna a mais, Deve-se especificar as colunas que deseja-se comparar:

```
all.equal(tab_nlme_talh_par[, c(2, 3, 4, 5)], tab_dplyr_talh_par[,
    c(3, 4, 5, 6)])
```

```
## [1] "Attributes: < Component \"class\": Lengths (1, 3) differ (string compare on first 1) >"
## [2] "Attributes: < Component \"class\": 1 string mismatch >"
## [3] "Attributes: < Component \"row.names\": Modes: character, numeric >"
## [4] "Attributes: < Component \"row.names\": target is character, current is numeric >"
```

Recebe-se o aviso de a classe dos nomes das colunas são diferentes porém não há nenhum outro aviso, ou seja, os dados gerados são idênticos.

6) Exportar tabelas de coeficientes

```
write.csv2(tab_rbase_talh, "tabelas/tab_rbase_talh.csv")
write.csv2(tab_rbase_talh_par, "tabelas/tab_rbase_talh_par.csv")

write.csv2(tab_dplyr_talh, "tabelas/tab_dplyr_talh.csv")
write.csv2(tab_dplyr_talh_par, "tabelas/tab_dplyr_talh_par.csv")

write.csv2(tab_nlme_talh, "tabelas/tab_nlme_talh.csv")
write.csv2(tab_nlme_talh_par, "tabelas/tab_nlme_talh_par.csv")
```

7) Junção dos dados originais e dados de regressão

7.1) Importar dados

```
# tab_reg <- read.csv(file.choose(), header = T)
```

7.2) R base

Caso os dados nao possuam a chave utilizada na regressão, deve-se adicioná-la:

```
dados_orig_mod <- dados_orig
dados_orig_mod$TALHAO_PAR <- paste(dados_orig_mod$TALHAO,
    dados_orig_mod$PARCELA, sep = "_")
```

Utiliza-se a função Merge, que se comporta de forma similar ao PROCV; Os primeiros dois argumentos informam os dados a serem assimilados; O terceiro argumento informa sera o fator de uniao. Utiliza-se o argumento all.x = TRUE, para garantir que os dados originais sejam preservados

```
tab_final_rbase_talh <- merge(dados_orig, tab_rbase_talh,
    by = "TALHAO", all.x = TRUE)
tab_final_rbase_talh_par <- merge(dados_orig_mod, tab_rbase_talh_par,
    by = "TALHAO_PAR", all.x = TRUE)
```

Substitui-se NAs por 0, caso desejado

```
tab_final_rbase_talh[is.na(tab_final_rbase_talh)] <- 0
tab_final_rbase_talh_par[is.na(tab_final_rbase_talh_par)] <- 0

head(tab_final_rbase_talh)
```

```
##   TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD        b0        b1
## 1   3654     101        301 51.1 28.4    Normal 30.45 0.7599015 -9.160042
## 2   3654     101        303 50.8  0.0    Normal 30.45 0.7599015 -9.160042
## 3   3654     101        304 53.2  0.0    Normal 30.45 0.7599015 -9.160042
## 4   3654     101        503 50.0  0.0    Normal 30.45 0.7599015 -9.160042
## 5   3654     101        302 51.3 29.0    Normal 30.45 0.7599015 -9.160042
## 6   3654     101        401 54.0  0.0    Normal 30.45 0.7599015 -9.160042
##          b2      Rsqr Std.Error
## 1 0.9190132 0.6494627 0.0363929
## 2 0.9190132 0.6494627 0.0363929
## 3 0.9190132 0.6494627 0.0363929
## 4 0.9190132 0.6494627 0.0363929
## 5 0.9190132 0.6494627 0.0363929
## 6 0.9190132 0.6494627 0.0363929
```

```
head(tab_final_rbase_talh_par)
```

```
##   TALHAO_PAR TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD       b0
## 1   3654_101   3654     101        204 53.2 28.8    Normal 30.45 3.803258
## 2   3654_101   3654     101        405 59.0 30.2 Dominante 30.45 3.803258
## 3   3654_101   3654     101        101 51.0 29.2    Normal 30.45 3.803258
## 4   3654_101   3654     101        201 59.3 30.7 Dominante 30.45 3.803258
## 5   3654_101   3654     101        203 54.0 28.7    Normal 30.45 3.803258
## 6   3654_101   3654     101        202 56.6 30.6    Normal 30.45 3.803258
##          b1      Rsqr  Std.Error
## 1 -7.269544 0.6780268 0.01744684
## 2 -7.269544 0.6780268 0.01744684
## 3 -7.269544 0.6780268 0.01744684
## 4 -7.269544 0.6780268 0.01744684
## 5 -7.269544 0.6780268 0.01744684
## 6 -7.269544 0.6780268 0.01744684
```

7.3) dplyr

Com o pacote dplyr nao e necessário unir as chaves: caso elas nao existam, devem ser adicionadas com a função mutate() ou cbind() separadamente, e entao, Utiliza-se full join (ou left join) para garantir que os dados originais nao sejam alterados

```
tab_final_dplyr_talh <- dados_orig %>% full_join(tab_dplyr_talh)
```

```
## Joining, by = "TALHAO"
```

```
tab_final_dplyr_talh_par <- dados_orig %>% full_join(tab_dplyr_talh_par,
    by = c("TALHAO", "PARCELA"))
```

Substituir NAs por 0, caso desejado

```
tab_final_dplyr_talh[is.na(tab_final_dplyr_talh)] <- 0
tab_final_dplyr_talh_par[is.na(tab_final_dplyr_talh_par)] <- 0

head(tab_final_dplyr_talh)
```

```
##   TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD        b0        b1
## 1   3654     101        301 51.1 28.4    Normal 30.45 0.7599015 -9.160042
## 2   3654     101        303 50.8  0.0    Normal 30.45 0.7599015 -9.160042
## 3   3654     101        304 53.2  0.0    Normal 30.45 0.7599015 -9.160042
## 4   3654     101        503 50.0  0.0    Normal 30.45 0.7599015 -9.160042
## 5   3654     101        302 51.3 29.0    Normal 30.45 0.7599015 -9.160042
## 6   3654     101        401 54.0  0.0    Normal 30.45 0.7599015 -9.160042
##          b2      Rsqr Std.Error
## 1 0.9190132 0.6494627 0.0363929
## 2 0.9190132 0.6494627 0.0363929
## 3 0.9190132 0.6494627 0.0363929
## 4 0.9190132 0.6494627 0.0363929
## 5 0.9190132 0.6494627 0.0363929
## 6 0.9190132 0.6494627 0.0363929
```

```
head(tab_final_dplyr_talh_par)
```

```
##   TALHAO PARCELA COD_ARVORE  CAP   HT CATEGORIA    HD       b0        b1
## 1   3654     101        301 51.1 28.4    Normal 30.45 3.803258 -7.269544
## 2   3654     101        303 50.8  0.0    Normal 30.45 3.803258 -7.269544
## 3   3654     101        304 53.2  0.0    Normal 30.45 3.803258 -7.269544
## 4   3654     101        503 50.0  0.0    Normal 30.45 3.803258 -7.269544
## 5   3654     101        302 51.3 29.0    Normal 30.45 3.803258 -7.269544
## 6   3654     101        401 54.0  0.0    Normal 30.45 3.803258 -7.269544
##        Rsqr  Std.Error
## 1 0.6780268 0.01744684
## 2 0.6780268 0.01744684
## 3 0.6780268 0.01744684
## 4 0.6780268 0.01744684
## 5 0.6780268 0.01744684
## 6 0.6780268 0.01744684
```

Pode-se verificar se as tabelas finais são iguais:

```
all.equal(tab_final_dplyr_talh, tab_final_rbase_talh)
```

```
## [1] TRUE
```

Recebe-se a resposta TRUE, ou seja, as tabelas são idênticas

8) Exportar tabelas finais

```
write.csv(tab_final_rbase_talh, "tabelas/tab_final_rbase_talh.csv")
write.csv(tab_final_rbase_talh_par, "tabelas/tab_final_rbase_talh_par.csv")
```

```r
write.csv(tab_final_dplyr_talh, "tabelas/tab_final_dplyr_talh.csv")
write.csv(tab_final_dplyr_talh_par, "tabelas/tab_final_dplyr_talh_par.csv")
```