Tarea 3

Daniel

16 de noviembre de 2018

Ejercicio 1: Ordenar los datos

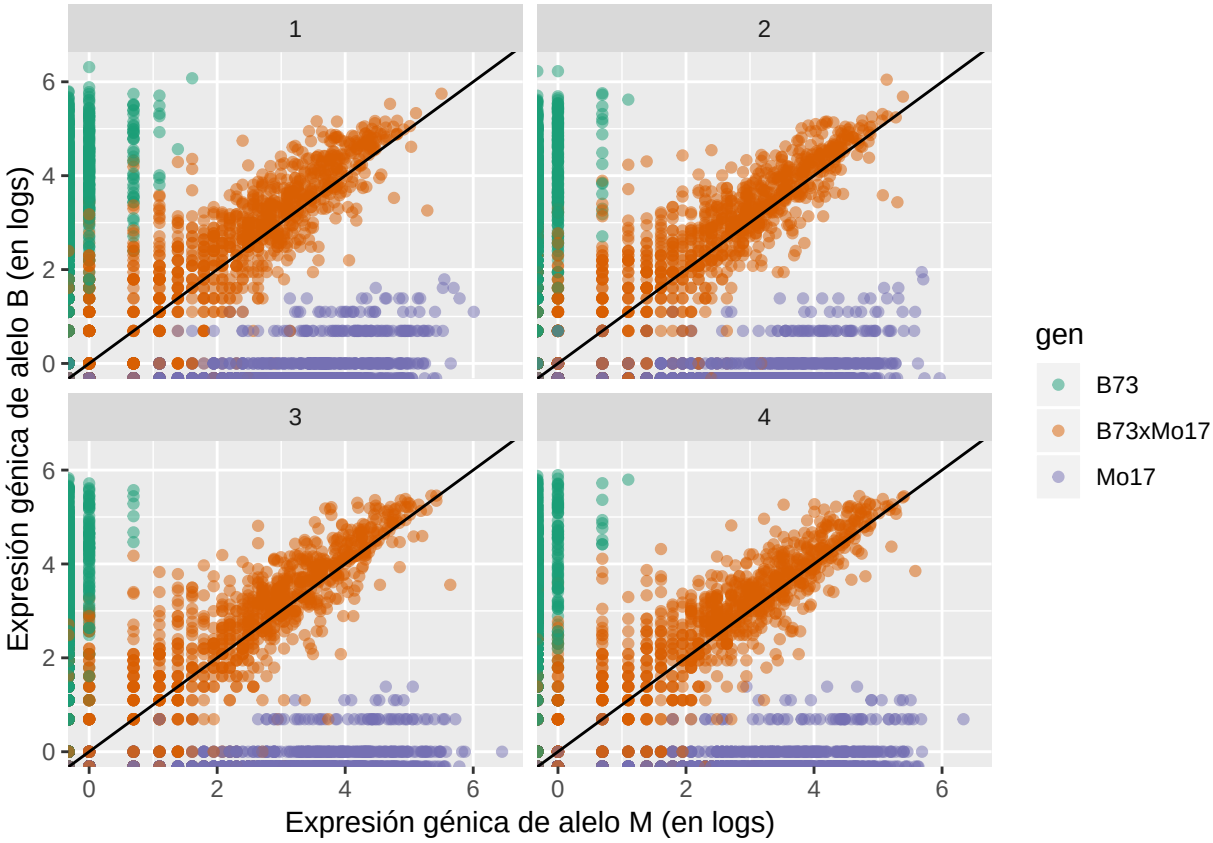
a)

```
dd <- read.table('datos_rna.txt', header = T)
dd.tidy <- NULL
dd.tidy <- dd %>%
  gather(rep, n, -GeneID) %>%
  separate(2, into = c('Rep', 'gen', 'cond')) %>%
  spread(cond,n) %>% mutate(Rep = recode(Rep, 'REP1' = 1, 'REP2' = 2, 'REP3' = 3, 'REP4' = 4) ) %>%
  mutate(gen = recode(gen, B = 'B73',  M = 'Mo17', BM = 'B73xMo17', MB= 'Mo17xB73') )

dd.tidy %>% filter(GeneID == 'AC155377.1_FG001')
```

```
##              GeneID Rep      gen   b   m total
## 1  AC155377.1_FG001   1      B73 140   0  1758
## 2  AC155377.1_FG001   1 B73xMo17 106  67  2057
## 3  AC155377.1_FG001   1     Mo17   2 138  2014
## 4  AC155377.1_FG001   1 Mo17xB73 128  99  2521
## 5  AC155377.1_FG001   2      B73 179   0  2050
## 6  AC155377.1_FG001   2 B73xMo17 134  84  2454
## 7  AC155377.1_FG001   2     Mo17   1 150  2205
## 8  AC155377.1_FG001   2 Mo17xB73 119  70  2001
## 9  AC155377.1_FG001   3      B73  34   0   173
## 10 AC155377.1_FG001   3 B73xMo17  22   8   166
## 11 AC155377.1_FG001   3     Mo17   1  35   201
## 12 AC155377.1_FG001   3 Mo17xB73  19  17   186
## 13 AC155377.1_FG001   4      B73  30   0   170
## 14 AC155377.1_FG001   4 B73xMo17  24  11   138
## 15 AC155377.1_FG001   4     Mo17   1  38   258
## 16 AC155377.1_FG001   4 Mo17xB73  27  22   213
```

b)

```
dd.tidyg <- dd.tidy %>% filter(gen != 'Mo17xB73')

ggplot(dd.tidyg, aes(x= log(m), y= log(b), colour=gen)) + geom_point(alpha=.5) + facet_wrap( ~ Rep) + g
```



c) Las cuatro repeticiones muestran que las plantas tienen en su mayoría el aleo correspondiente al gen orginal, y las mixtas parecen tener una proporción similar de cada aleo.

Ejercicio 2: Rcpp y benchmark

a)

```
c <- dd.tidy %>% filter(gen == 'B73xMo17') %>%  filter(GeneID == 'AC155377.1_FG001')

compara <- function(x, y) {
  m <- length(x)
  n <- length(y)
  # calculo el estadistico de la prueba
  sp <- sqrt(((m-1)*sd(x)^2 + (n-1)*sd(y)^2) / (m+n-2))
  tstat <- (mean(x) - mean(y)) / (sp*sqrt(1/m + 1/n))
  # calculo el p-valor
  2*(1 - pt( abs(tstat), df = n+m-2) )
}

compara(c$b,c$m)
```

```
## [1] 0.433179
```

```r
t.test(c$b, c$m)
```

```
##
##  Welch Two Sample t-test
##
## data:  c$b and c$m
## t = 0.83985, df = 5.2765, p-value = 0.4374
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -58.38153 116.38153
## sample estimates:
## mean of x mean of y
##      71.5      42.5
```

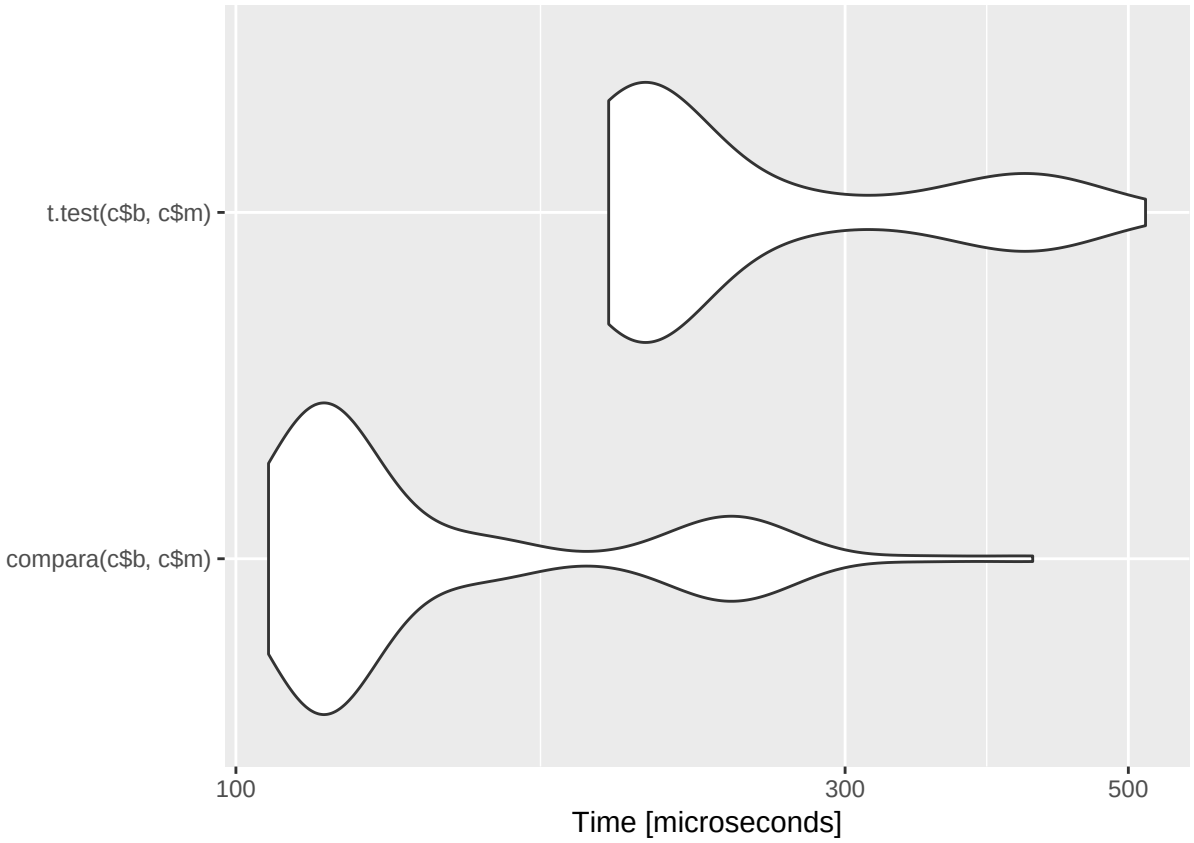
-cppFunction('double comparaC(NumericVector x, NumericVector y) { int m = x.size(); int z = y.size(); int sp = sqrt(((m-1)*pow(sd(x), 2.0) + (z-1)*pow(sd(y),2.0)) / (m+z-2)); int tstat = (mean(x) - mean(y)) / (sp*sqrt(1/m + 1/z)); double n = z+m-2; double a = abs(tstat); double pvalue = 2*(1 - pt(a,n)); return pvalue; }')

comparaC(cb, cm)

b)

```r
mb <- microbenchmark(
  compara(c$b,c$m),
  t.test(c$b, c$m)
)


autoplot(mb)
```

```
## Coordinate system already present. Adding new coordinate system, which will replace the existing one
```



c) Compara va ser más eficiente que la función t-test.

d) La baja cantidad de observaciones.