Allocating Treatments in a CRD

Imagine 4 treatments and 16 units total. In a completely randomized design (CRD), we want to allocate 4 units to each of the 4 treatments at random:

```
trts = sample(rep(1:4, each=4)) # "rep"eats 1,2,3,4, 4 times each: 1,1,1,1,2,2,...,4
trts # "sample": like drawing from a hat, without replacement
```

```
 [1] 2 2 3 3 1 3 3 4 4 1 1 2 4 2 1 4
```

now we can arrange these allocations of treatments to units in a 4x4 grid:

```
crd = matrix(trts, 4, 4)
```

```
     [,1] [,2] [,3] [,4]
[1,]    2    1    4    4
[2,]    2    3    1    2
[3,]    3    3    1    1
[4,]    3    4    2    4
```

Here is a “fair” arrangement of the treatments: each treatment appears only once in each row and column:

```
library(magic) # the function "rlatin" is in package "magic"
fair = rlatin(4)
```

```
     [,1] [,2] [,3] [,4]
[1,]    3    4    2    1
[2,]    2    1    3    4
[3,]    1    3    4    2
[4,]    4    2    1    3
```

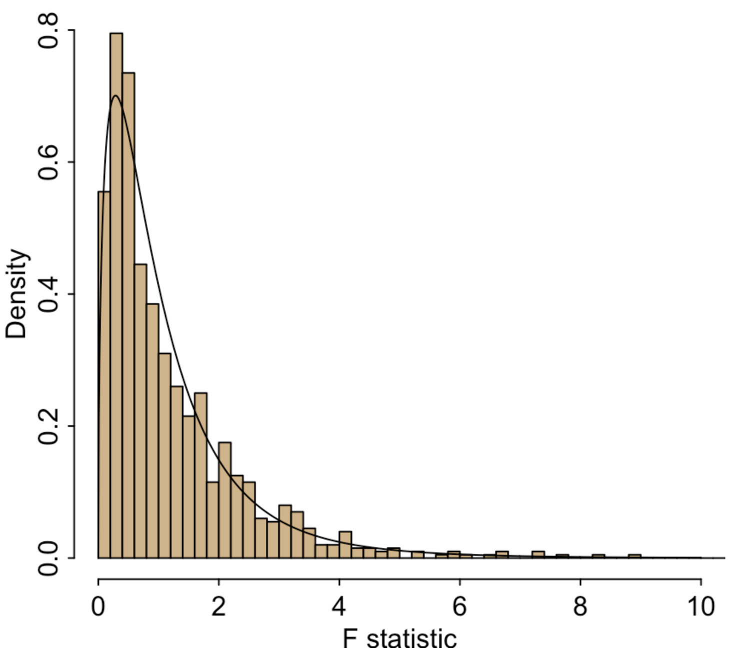
Now suppose we have some data, with a hidden trend. In a corn field, this trend might be caused by heterogeneity, in the *absence* of any treatment: the plots on the left might be well watered, and the plots on the right might be too dry (uphill):

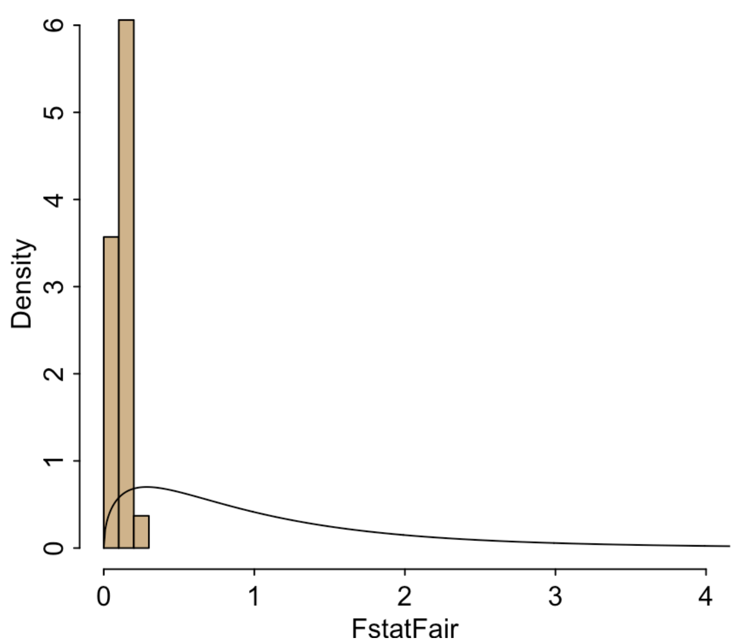
```
dat = c(12,14,15,17,  7,6,10,12,  5,4,8,7,  4,3,2,1)
matrix(dat, 4,4) # to view in 4 rows, 4 columns
```

```
     [,1] [,2] [,3] [,4]
[1,]   12    7    5    4
[2,]   14    6    4    3
[3,]   15   10    8    2
[4,]   17   12    7    1
```

Suppose there is no treatment effect. Then we are really just randomly assigning the labels “1”, “2”, “3”, and “4” to the plots. From each (random) assignment, we can calculate the F test statistic. It should look like what it does when the null hypothesis is true: an F distribution.

```
n_simulations = 1000
FstatRandom = rep(NA, n_simulations) # 1000 missing values for now
FstatFair = rep(NA, n_simulations)
for (i in 1:n_simulations){
  trts = sample(rep(1:4, each=4))
  fit  = lm(dat ~ as.factor(trts))
  FstatRandom[i] = anova(fit)$F[1] # i-th F stat: extracted from ANOVA table
  trts = c(rlatin(4)) # do it over, but fair allocation
  fit  = lm(dat ~ as.factor(trts))
  FstatFair[i] = anova(fit)$F[1]
}
layout(matrix(1:2,1,2))
hist(FstatRandom, probability=TRUE, breaks=40, ylim=c(0,.8), xlim=c(0,10),
     col="tan", main="", xlab="F statistic")
xx = seq(0, 10, 0.01)
lines(xx, df(xx, 3, 12)) # df = density of F distribution
hist(FstatFair, probability=TRUE, breaks=2, xlim=c(0, 4), col="tan", main="")
lines(xx, df(xx, 3, 12))
```





above left: histogram of 1000 F-statistics based on 1000 random allocations of treatment labels in a CRD with k=4 treatments and n=4 units per treatments. The smooth line corresponds to an $F_{3,12}$ distribution.

above right: the same but with “fair” allocations, using latin squares. histogram of 1000 F-statistics based on 1000 “fair” allocations of treatment labels in a CRD with k=4 treatments and n=4 units per treatments. The smooth line corresponds to an $F_{3,12}$ distribution.

latin square analysis: to follow the design

if we assign treatments with a latin square ("fair" assignment), we should analyze the resulting data accordingly: with a row factor, a column factor, and the treatment factor, like in this example (note that the column effect is detected as significant).

```
plotrow = factor(rep(1:4, 4))      # the row number: 1,2,3,4, 1,2,3,4, etc.
plotcol = factor(rep(1:4, each=4)) #  column number: 1,1,1,1, 2,2,2,2, etc.
trts = c(rlatin(4)) # one random fair allocation of treatments
fit  = lm(dat ~ plotrow + plotcol + as.factor(trts))
drop1(fit, test="F")
```

```
Single term deletions

Model:
dat ~ plotrow + plotcol + as.factor(trts)
                Df Sum of Sq    RSS    AIC F value    Pr(>F)    
<none>                        20.38 23.868                      
plotrow          3    18.688  39.06 28.281  1.8344 0.2414210    
plotcol          3   308.188 328.56 62.354 30.2515 0.0005094 ***
as.factor(trts)  3    11.687  32.06 25.122  1.1472 0.4034870    
```