STOR 590 HW7 Solution

Taebin Kim

3/31/2020

Page 233 Exercise 2

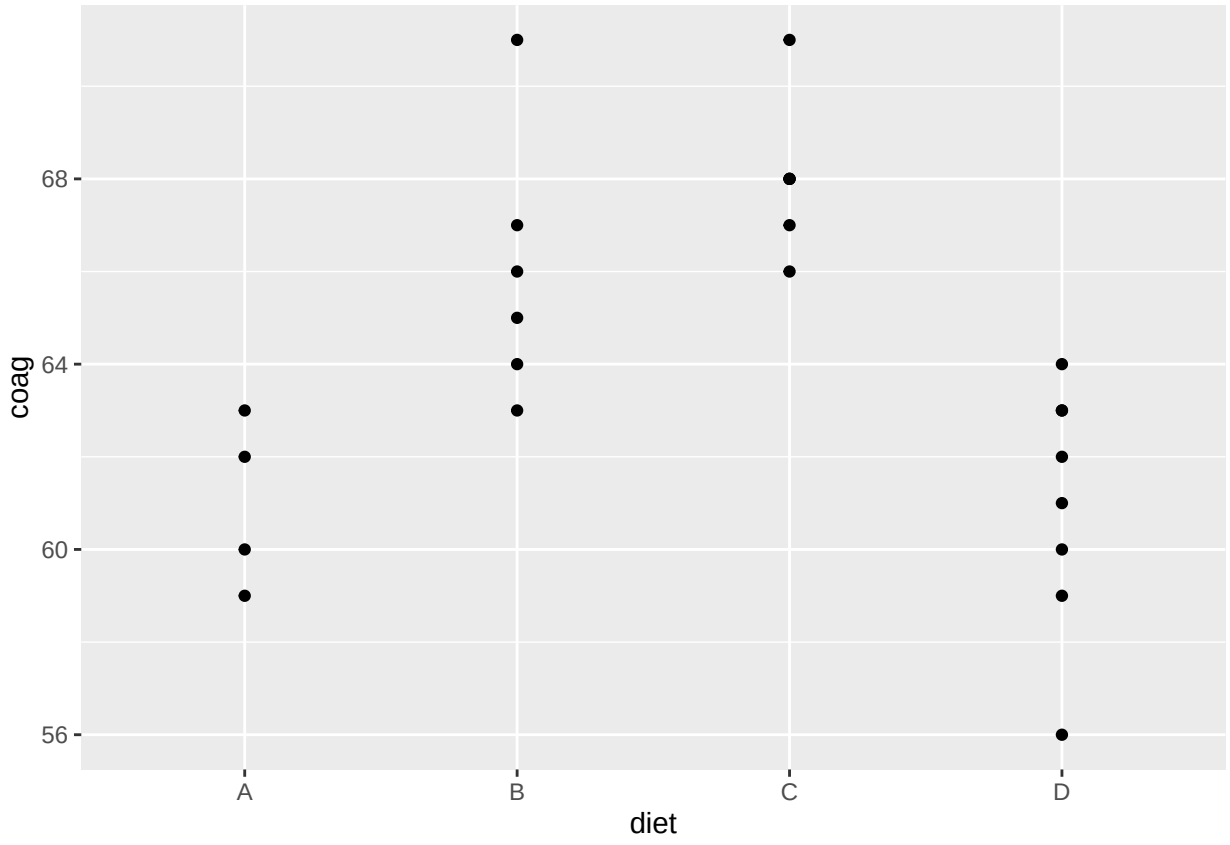
Part (a)

We plot the data.

```
library(faraway)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
data(coagulation)
ggplot(coagulation, aes(diet, coag)) + geom_point()
```



We can observe that diet A and D leads to less blood coagulation times than other diets.

Part (b)

We fit a fixed effects model and construct a prediction together with a 95% prediction interval for the response of a new animal assigned to diet D.

```
lmod <- aov(coag ~ diet, coagulation)
summary(lmod)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## diet         3    228    76.0   13.57 4.66e-05 ***
## Residuals   20    112     5.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
predict(lmod, newdata=data.frame(diet="D"), interval = "predict")
```

```
##   fit      lwr      upr
## 1  61 55.76427 66.23573
```

We can see that the predicted value of coag is 61 and the prediction interval is [55.76427, 66.23573].

Part (c)

We fit a random effects model using REML. Then we predict the blood coagulation time for a new animal assigned to diet D along with a 95% prediction interval.

```
library(lme4)
```

```
## Loading required package: Matrix
```

```
mmod <- lmer(coag ~ 1 + (1|diet), coagulation)
summary(mmod)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: coag ~ 1 + (1 | diet)
##    Data: coagulation
##
## REML criterion at convergence: 115.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.18491 -0.59921  0.09332  0.54078  2.17508
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  diet     (Intercept) 11.692   3.419
##  Residual              5.599   2.366
## Number of obs: 24, groups:  diet, 4
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)    64.01       1.78   35.96
```

```r
group.sd <- as.data.frame(VarCorr(mmod))$sdcor[1]
resid.sd <- as.data.frame(VarCorr(mmod))$sdcor[2]
predict(mmod, newdata=data.frame(diet="D"))
```

```
##        1
## 61.17017
```

```r
pv <- numeric(1000)

set.seed(590)
for(i in 1:1000){
  y <- unlist(simulate(mmod))
  bmod <- refit(mmod, y)
  pv[i] <- predict(bmod, newdata=data.frame(diet="D"))
  + rnorm(n=1,sd=resid.sd)
}
```

```
## boundary (singular) fit: see ?isSingular
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.0103843
## (tol = 0.002, component 1)
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00232904
## (tol = 0.002, component 1)
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
quantile(pv, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 57.33823 71.43798
```

The predicted value of the blood coagulation time for a new animal assigned to diet D is 61.17017 and the prediction interval is [57.33823, 71.43798].

Part (d)

We predict the blood coagulation time for a new animal given a new diet along with a 95% prediction interval.

```
predict(mmod, re.form=~0)[1]
```

```
##        1
## 64.01266
```

```
pv <- numeric(1000)
set.seed(590)
for(i in 1:1000){
  y <- unlist(simulate(mmod))
  bmod <- refit(mmod, y)
  pv[i] <- predict(bmod, re.form=~0)[1]
  + rnorm(n=1,sd=group.sd)
  + rnorm(n=1,sd=resid.sd)
}
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00593833
## (tol = 0.002, component 1)
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00571489
## (tol = 0.002, component 1)

## boundary (singular) fit: see ?isSingular

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00359389
## (tol = 0.002, component 1)

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00204675
## (tol = 0.002, component 1)

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
quantile(pv, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 60.63008 67.45680
```

The predicted value is 64.01266 and the 95% prediction interval is [60.63008, 67.45680].

Part (e)

We predict the blood coagulation time for the first animal in the dataset given a new diet along with a 95% prediction interval.

```
predict(mmod, re.form=~0)[1]
```

```
##        1
## 64.01266
```

```
pv <- numeric(1000)
set.seed(590)
for(i in 1:1000){
  y <- unlist(simulate(mmod))
  bmod <- refit(mmod, y)
  pv[i] <- predict(bmod, re.form=~0)[1]
  + rnorm(n=1,sd=group.sd)
}
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.0103843
## (tol = 0.002, component 1)

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00232904
## (tol = 0.002, component 1)

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

```
quantile(pv, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 60.80694 67.75822
```

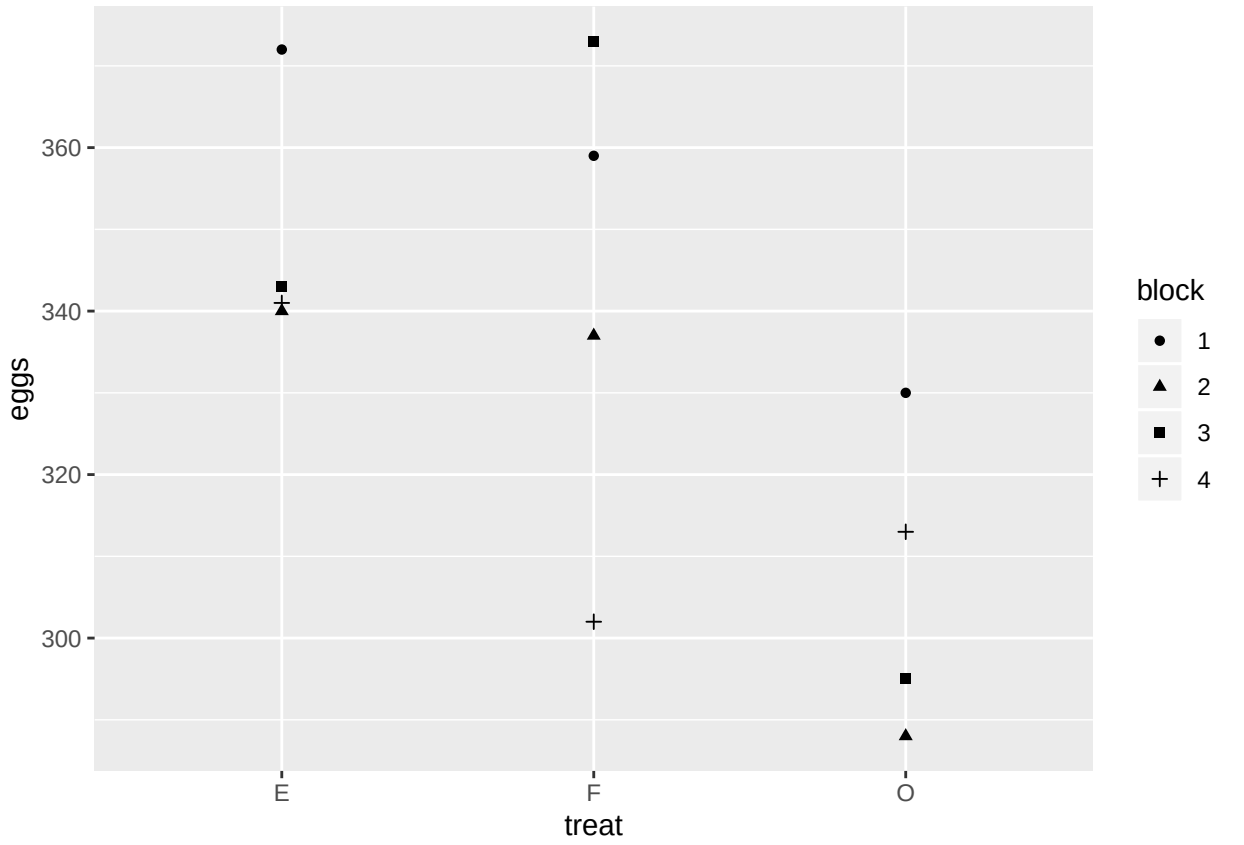
The predicted value is 64.01266 and the predicted interval is [60.80694, 67.75822].

Page 233 Exercise 3

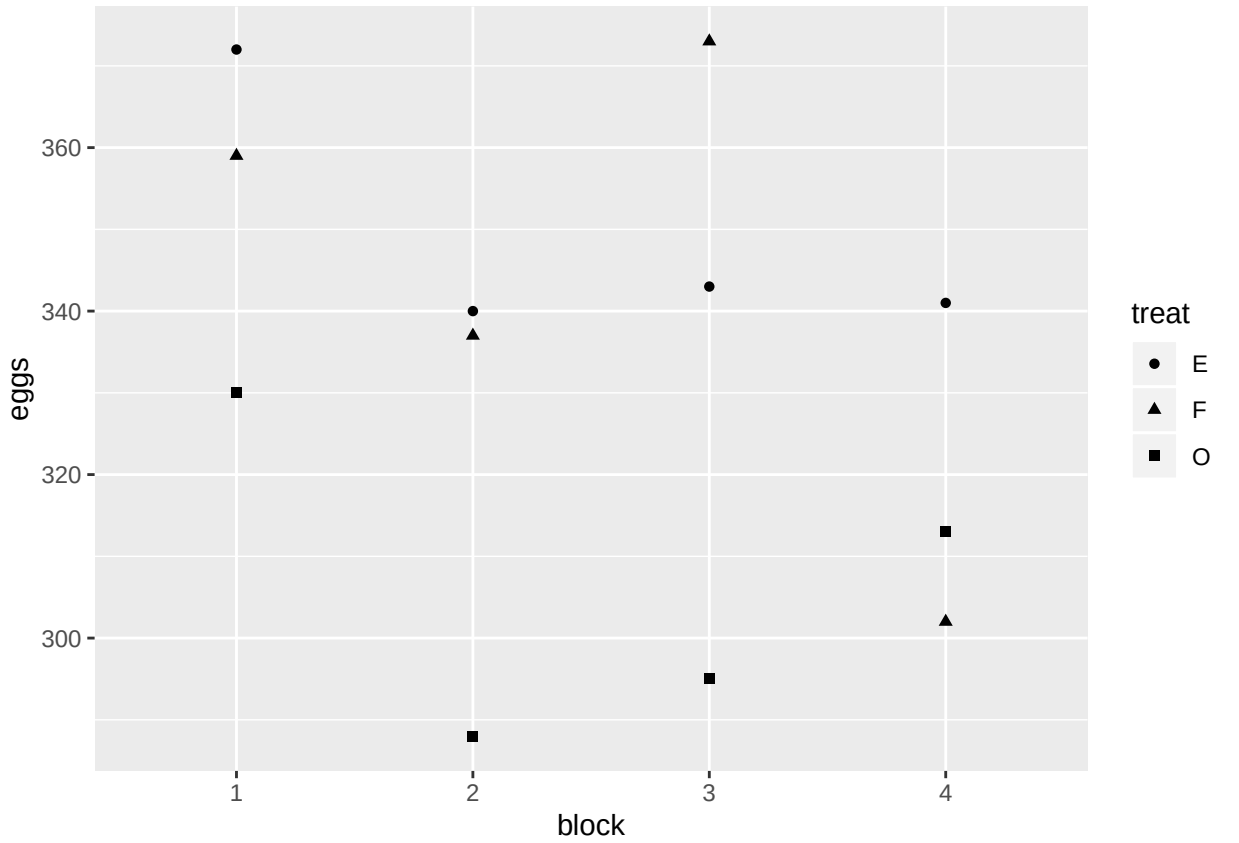
Part (a)

We make suitable plots of the `eggprod` data.

```
data(eggprod)
ggplot(eggprod, aes(treat, eggs, shape = block)) + geom_point()
```




```
ggplot(eggprod, aes(block, eggs, shape = treat)) + geom_point()
```



We can observe that the number of eggs produced is large for treatment E and small for treatment O, while it is evenly distributed across different blocks.

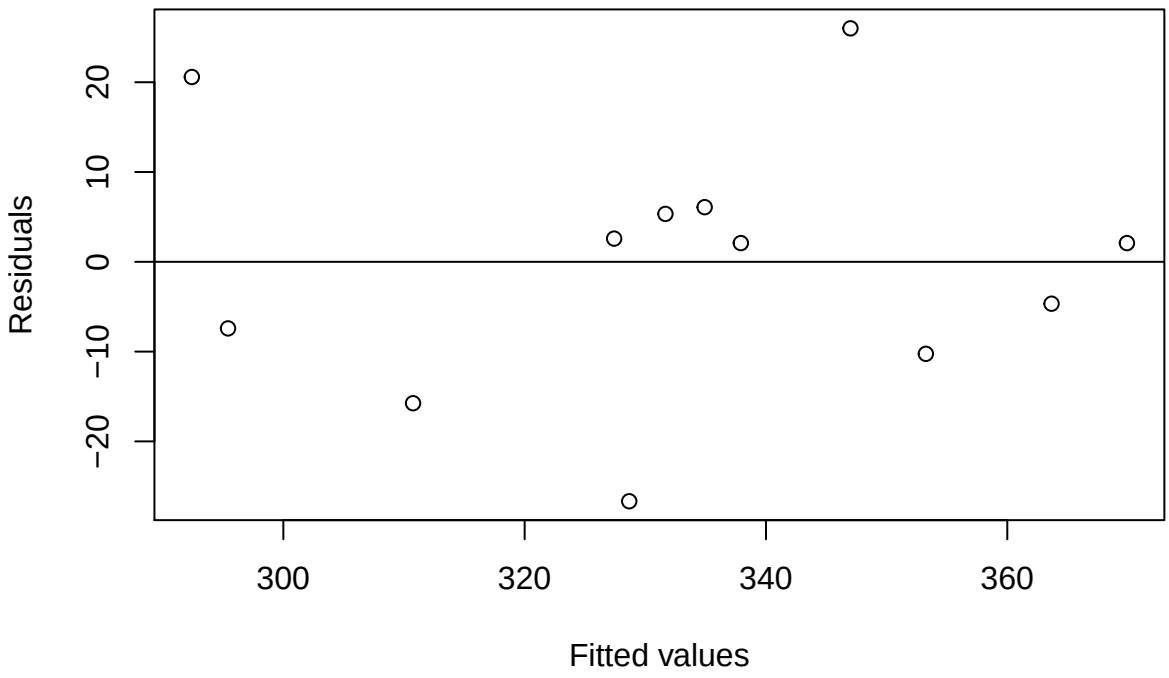
Part (b)

We fit a fixed effects model for the number of eggs produced with the treatments and blocks as predictors. We also determine the significance of the two predictors and perform a basic diagnostic check.

```
lmod <- aov(eggs ~ treat + block, eggprod)
summary(lmod)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## treat        2   4212  2106.2   5.444 0.0449 *
## block        3   2330   776.8   2.008 0.2145
## Residuals    6   2322   386.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(residuals(lmod) ~ fitted(lmod), xlab = "Fitted values", ylab = "Residuals")
abline(h = 0)
```



The result shows that `treat` is the only significant predictor at 5% significance level. The diagnostic plot show no problem in the model.

Part (c)

We fit a model for the number of eggs produced with the treatments as fixed effects and the bloacks as random effects.

```
mmod <- lmer(eggs ~ treat + (1|block), eggprod)
sumary(mmod)
```

```
## Fixed Effects:
##             coef.est coef.se
## (Intercept) 349.00    11.37
## treatF       -6.25    13.91
## treatO      -42.50    13.91
##
## Random Effects:
##  Groups   Name        Std.Dev.
##  block    (Intercept) 11.40
##  Residual             19.67
## ---
## number of obs: 12, groups: block, 4
## AIC = 95.4, DIC = 124.4
## deviance = 104.9
```

We can see that `treatE` has the largest coefficient (=0), which implies that treatment E maximizes the estimated egg production. We are not sure if it is better than treatment F since the difference between the coefficient of `treatE` and `treatF`is smaller than the estimated standard error.

Part (d)

We use the Kenward-Roger approximation for an F-test to check for differences between the treatments.

```
library(pbkrtest)
```

```
## Warning: package 'pbkrtest' was built under R version 3.6.3
```

```
amod <- lmer(eggs ~ treat + (1|block), eggprod, REML = FALSE)
nmod <- lmer(eggs ~ 1 + (1|block), eggprod, REML = FALSE)
```

```
## boundary (singular) fit: see ?isSingular
```

```
KRmodcomp(amod, nmod)
```

```
## F-test with Kenward-Roger approximation; time: 0.11 sec
## large : eggs ~ treat + (1 | block)
## small : eggs ~ 1 + (1 | block)
##         stat    ndf    ddf F.scaling p.value
## Ftest 5.4438 2.0000 6.0000         1 0.04485 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value of the F-test is smaller than 0.05, we conclude that `treat` is significant. We can make the same conclusion with the fixed effects model.

Part (e)

We perform the same test using a bootstrap method.

```
set.seed(590)
pmod <- PBmodcomp(amod, nmod)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.00561394
## (tol = 0.002, component 1)
```

```
summary(pmod)
```

```
## Bootstrap test; time: 16.95 sec;samples: 1000; extremes: 68;
## large : eggs ~ treat + (1 | block)
## small : eggs ~ 1 + (1 | block)
##            stat     df   ddf p.value
## LRT      8.4245 2.0000       0.01481 *
```

```
## PBtest   8.4245              0.06893 .
## Gamma    8.4245              0.06470 .
## Bartlett 5.4376 2.0000       0.06595 .
## F        4.2123 2.0000 2.953 0.13649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value of PBtest is larger than 0.05, we conclude that `treat` is not significant. The result is different from (f). (This part might vary for different seeds.)

Part (f)

We test for significance of the blocks.

```
library(RLRsim)
```

```
## Warning: package 'RLRsim' was built under R version 3.6.3
```

```
set.seed(590)
exactRLRT(mmod)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0.51536, p-value = 0.2156
```

Since the p-value is larger than 0.05, we conclude that `block` is not significant. Note that the outcome is similar to the fixed effects model.