m9

Will Tirone

Q1)

a)

Here just copying the script posted on Sakai for Methods 9.

```
#set up a true mean value for cases that get no treatments at all
trt0 = 10

#set up true values for the main effects for 4 treatments
trta = 2
trtb = -1
trtc = 4
trtd = 3

#set up true values for the effects for two-way interactions
trtab = 4
trtac = -1
trtad = 2
trtbc = 6
trtbd = -3
trtcd = 2

#these are for use in Problem 3
#trtbc = 0
#trtbd = 0
#trtcd = 0


#we will set all three-way interactions and the four-way interaction
trtabc = 0
```

```r
trtabd = 0
trtacd = 0
trtbcd = 0
trtabcd = 0

true_values = c(trt0, trta, trtb, trtc, trtd, trtab, trtac, trtad,
                trtbc, trtbd, trtcd)

true_three = c(true_values, 0, 0, 0, 0)

x0 = c(0, 0, 0, 0)
xa = c(1, 0, 0, 0)
xb = c(0, 1, 0, 0)
xc = c(0, 0, 1, 0)
xd = c(0, 0, 0, 1)
xab = c(1, 1, 0, 0)
xac = c(1, 0, 1, 0)
xad = c(1, 0, 0, 1)
xbc = c(0, 1, 1, 0)
xbd = c(0, 1, 0, 1)
xcd = c(0, 0, 1, 1)
xabc = c(1, 1, 1, 0)
xabd = c(1, 1, 0, 1)
xacd = c(1, 0, 1, 1)
xbcd = c(0, 1, 1, 1)
xabcd = c(1, 1, 1, 1)

## with this information, you can make the datasets needed
#to answer the questions about the full factorial design

n = 4
sigma = 0.2
y0 = rnorm(n, trt0, sigma)
ya = rnorm(n, trt0 + trta, sigma)
yb = rnorm(n, trt0 + trtb, sigma)
yc = rnorm(n, trt0 + trtc, sigma)
yd = rnorm(n, trt0 + trtd, sigma)
yab = rnorm(n, trt0 + trta + trtb + trtab, sigma)
yac = rnorm(n, trt0 + trta + trtc + trtac, sigma)
yad = rnorm(n, trt0 + trta + trtd + trtad, sigma)
ybc = rnorm(n, trt0 + trtb + trtc + trtbc, sigma)
```

```
ybd = rnorm(n, trt0 + trtb + trtd + trtbd, sigma)
ycd = rnorm(n, trt0 + trtc + trtd + trtcd, sigma)
yabc = rnorm(n, trt0 + trta + trtb + trtc + trtab + trtbc + trtac + trtabc,
             sigma)
yabd = rnorm(n, trt0 + trta + trtb + trtd + trtab + trtad + trtbd + trtabd,
             sigma)
yacd = rnorm(n, trt0 + trta + trtc + trtd + trtac + trtad + trtcd + trtacd,
             sigma)
ybcd = rnorm(n, trt0 + trtb + trtc + trtd + trtbc + trtbd + trtcd + trtbcd,
             sigma)
yabcd = rnorm(n, trt0 + trta + trtb + trtc + trtd + trtab + trtac + trtad +
                trtbc + trtbd + trtcd + trtabc + trtabd + trtacd + trtbcd +
                trtabcd, sigma)

#make a matrix with 4 rows per treatment group in order of treatment group
thedata = rbind(x0, x0, x0, x0)
thedata = rbind(thedata, xa, xa, xa, xa)
thedata = rbind(thedata, xb, xb, xb, xb)
thedata = rbind(thedata, xc, xc, xc, xc)
thedata = rbind(thedata, xd, xd, xd, xd)
thedata = rbind(thedata, xab, xab, xab, xab)
thedata = rbind(thedata, xac, xac, xac, xac)
thedata = rbind(thedata, xad, xad, xad, xad)
thedata = rbind(thedata, xbc, xbc, xbc, xbc)
thedata = rbind(thedata, xbd, xbd, xbd, xbd)
thedata = rbind(thedata, xcd, xcd, xcd, xcd)
thedata = rbind(thedata, xabc, xabc, xabc, xabc)
thedata = rbind(thedata, xabd, xabd, xabd, xabd)
thedata = rbind(thedata, xacd, xacd, xacd, xacd)
thedata = rbind(thedata, xbcd, xbcd, xbcd, xbcd)
thedata = rbind(thedata, xabcd, xabcd, xabcd, xabcd)

#now add the y column, which is in the same order as thedata
y = c(y0, ya, yb, yc, yd, yab, yac, yad, ybc, ybd, ycd, yabc, yabd, yacd,
      ybcd, yabcd)
thedata = cbind(thedata, y)
thedata = data.frame(thedata)
names(thedata)[1:4] = c("A", "B", "C", "D")
```

And printing all the data:

```
thedata |> kable()
```

	A	B	C	D	y
x0	0	0	0	0	10.259263
x0.1	0	0	0	0	9.940897
x0.2	0	0	0	0	10.017658
x0.3	0	0	0	0	10.203165
xa	1	0	0	0	11.944762
xa.1	1	0	0	0	12.350000
xa.2	1	0	0	0	12.286880
xa.3	1	0	0	0	11.709432
xb	0	1	0	0	9.154557
xb.1	0	1	0	0	8.887689
xb.2	0	1	0	0	9.179199
xb.3	0	1	0	0	8.882747
xc	0	0	1	0	13.551902
xc.1	0	0	1	0	14.017316
xc.2	0	0	1	0	14.081149
xc.3	0	0	1	0	14.256050
xd	0	0	0	1	12.995930
xd.1	0	0	0	1	12.953389
xd.2	0	0	0	1	13.196978
xd.3	0	0	0	1	13.129387
xab	1	1	0	0	15.373295
xab.1	1	1	0	0	15.142900
xab.2	1	1	0	0	15.192717
xab.3	1	1	0	0	15.382183
xac	1	0	1	0	14.981626
xac.1	1	0	1	0	15.017623
xac.2	1	0	1	0	14.881186
xac.3	1	0	1	0	14.899980
xad	1	0	0	1	16.652175
xad.1	1	0	0	1	17.177277
xad.2	1	0	0	1	16.851713
xad.3	1	0	0	1	16.736433
xbc	0	1	1	0	19.133619
xbc.1	0	1	1	0	19.068356
xbc.2	0	1	1	0	18.726274
xbc.3	0	1	1	0	18.740028
xbd	0	1	0	1	9.009708
xbd.1	0	1	0	1	9.222824
xbd.2	0	1	0	1	8.683258
xbd.3	0	1	0	1	9.055300
xcd	0	0	1	1	19.298030
xcd.1	0	0	1	1	19.073507
xcd.2	0	0	1	1	19.293920
xcd.3	0	0	1	1	19.031915
xabc	1	1	1	0	24.132857
xabc.1	1	1	1	0	23.904715
xabc.2	1	1	1	0	23.815693
xabc.3	1	1	1	0	24.099111
xabd	1	1	0	1	16.956414
xabd.1	1	1	0	1	16.933446
xabd.2	1	1	0	1	17.086319

b)

Yes! The values for the main and interaction effects are all fairly close to the true values in the table below.

```
mod_b = lm(y ~ A*B + A*C + A*D + B*C + B*D + C*D, data=thedata)

output_b = data.frame(
  estimate = coef(mod_b),
  true = true_values
)

output_b |> kable()
```

	estimate	true
(Intercept)	10.105926	10
A	1.978736	2
B	-1.065266	-1
C	3.887663	4
D	2.951568	3
A:B	4.226194	4
A:C	-1.056779	-1
A:D	1.817023	2
B:C	5.956470	6
B:D	-3.003320	-3
C:D	2.222948	2

c)

The interaction effect between B and C can be thought of as the "extra" effect that combining the two treatments has that each of the individual treatments does not. Since we have a positive effect here, we conclude that the two treatments do more together than they do individually.

As an example, say we're measuring levels of impairment in individuals who take certain drugs. Say treatment `B == Alcohol` and `C == Benzodiazepines`. An individual would feel some kind of impairment from Alcohol, and some kind of impairment from Benzodiazepines, but combining them has a severe (and can be fatal) effect that neither of the treatments alone have.

d)

Comparison

Visually inspecting the output table, it looks like this model does much worse than the first model. In particular, the three-way interaction effects are very far from the truth.

```
mod_d = lm(y ~ A*B + A*C + A*D + B*C + B*D + C*D +
                    A*B*C + A*B*D + A*C*D + B*C*D,
                    data=thedata)

output_d = data.frame(
  estimate = coef(mod_d),
  true = true_three
)

output_d |> kable()
```

	estimate	true
(Intercept)	10.0885345	10
A	2.0009454	2
B	-1.0457749	-1
C	3.9047810	4
D	2.9970978	3
A:B	4.2123579	4
A:C	-1.0658684	-1
A:D	1.7511106	2
B:C	5.9528174	6
B:D	-3.0637961	-3
C:D	2.1672185	2
A:B:C	-0.0429874	0
A:B:D	0.0706593	0
A:C:D	0.0611663	0
B:C:D	0.0502929	0

Nested F-test

No. In the F-test, the null hypothesis we're suggesting is that the three-way interactions would be non-zero. Since we have a p-value = 0.1486, we conclude that including them does not add to the model.

```r
anova(mod_b, mod_d)
```

```
Analysis of Variance Table

Model 1: y ~ A * B + A * C + A * D + B * C + B * D + C * D
Model 2: y ~ A * B + A * C + A * D + B * C + B * D + C * D + A * B * C +
    A * B * D + A * C * D + B * C * D
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     53 2.0297
2     49 2.0166  4  0.013111 0.0796 0.9882
```

Q2)

a)

```r
fractional = thedata[c(1:4, 21:44, 61:64), ]
fractional |> kable()
```

	A	B	C	D	y
x0	0	0	0	0	10.259263
x0.1	0	0	0	0	9.940897
x0.2	0	0	0	0	10.017658
x0.3	0	0	0	0	10.203165
xab	1	1	0	0	15.373295
xab.1	1	1	0	0	15.142900
xab.2	1	1	0	0	15.192717
xab.3	1	1	0	0	15.382183
xac	1	0	1	0	14.981626
xac.1	1	0	1	0	15.017623
xac.2	1	0	1	0	14.881186
xac.3	1	0	1	0	14.899980
xad	1	0	0	1	16.652175
xad.1	1	0	0	1	17.177277
xad.2	1	0	0	1	16.851713
xad.3	1	0	0	1	16.736433
xbc	0	1	1	0	19.133619
xbc.1	0	1	1	0	19.068356
xbc.2	0	1	1	0	18.726274
xbc.3	0	1	1	0	18.740028
xbd	0	1	0	1	9.009708
xbd.1	0	1	0	1	9.222824
xbd.2	0	1	0	1	8.683258
xbd.3	0	1	0	1	9.055300
xcd	0	0	1	1	19.298030
xcd.1	0	0	1	1	19.073507
xcd.2	0	0	1	1	19.293920
xcd.3	0	0	1	1	19.031915
xabcd	1	1	1	1	27.775531
xabcd.1	1	1	1	1	28.184955
xabcd.2	1	1	1	1	28.280645
xabcd.3	1	1	1	1	27.979933

b)

Yes, based on the summary below, 3 of the interaction effects are NA values, indicating R couldn't fit them.

```r
mod_b_2 = lm(y ~ A*B + A*C + A*D + B*C + B*D + C*D, data=fractional)

summary(mod_b_2)
```

```
Call:
lm(formula = y ~ A * B + A * C + A * D + B * C + B * D + C *
    D, data = fractional)

Residuals:
     Min       1Q   Median       3Q      Max
-0.30951 -0.12094  0.00712  0.12060  0.32288

Coefficients: (3 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.10525    0.09055 111.596  < 2e-16 ***
A           -0.59674    0.12806  -4.660 9.86e-05 ***
B           -0.68487    0.09055  -7.563 8.40e-08 ***
C            9.49670    0.09055 104.876  < 2e-16 ***
D           -0.42760    0.09055  -4.722 8.42e-05 ***
A:B          6.44914    0.12806  50.360  < 2e-16 ***
A:C         -4.06010    0.12806 -31.705  < 2e-16 ***
A:D          7.77349    0.12806  60.702  < 2e-16 ***
B:C               NA         NA      NA       NA
B:D               NA         NA      NA       NA
C:D               NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1811 on 24 degrees of freedom
Multiple R-squared:  0.9992,    Adjusted R-squared:  0.999
F-statistic:  4317 on 7 and 24 DF,  p-value: < 2.2e-16
```

c)

We can only estimate one set of interactions here! The summary is reported below.

```r
mod_c_2 = lm(y ~ A*B + C*D, data = fractional)
summary(mod_c_2)
```

```
Call:
lm(formula = y ~ A * B + C * D, data = fractional)

Residuals:
   Min     1Q Median     3Q    Max
-3.268 -1.485 -0.078  1.554  3.281

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.1769     1.0559   8.691 3.62e-09 ***
A             1.2600     1.2193   1.033 0.310951
B            -0.6849     1.2193  -0.562 0.579128
C             7.4666     0.8622   8.660 3.87e-09 ***
D             3.4591     0.8622   4.012 0.000453 ***
A:B           6.4491     1.7243   3.740 0.000917 ***
C:D               NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.439 on 26 degrees of freedom
Multiple R-squared:  0.8441,    Adjusted R-squared:  0.8142
F-statistic: 28.16 on 5 and 26 DF,  p-value: 1.024e-09
```

d)

No, the interactions here do not equal the true values. This is because, for example, AB and CD are aliased, so we are assuming the CD interaction is negligible when fitting A:B. But, since we generated the data, we know that is not true.

```r
mod_d_2 = lm(y ~ A*B + A*C + A*D , data = fractional)
summary(mod_d_2)
```

```
Call:
lm(formula = y ~ A * B + A * C + A * D, data = fractional)

Residuals:
     Min       1Q   Median       3Q      Max
-0.30951 -0.12094  0.00712  0.12060  0.32288
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.10525    0.09055 111.596  < 2e-16 ***
A           -0.59674    0.12806  -4.660 9.86e-05 ***
B           -0.68487    0.09055  -7.563 8.40e-08 ***
C            9.49670    0.09055 104.876  < 2e-16 ***
D           -0.42760    0.09055  -4.722 8.42e-05 ***
A:B          6.44914    0.12806  50.360  < 2e-16 ***
A:C         -4.06010    0.12806 -31.705  < 2e-16 ***
A:D          7.77349    0.12806  60.702  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1811 on 24 degrees of freedom
Multiple R-squared:  0.9992,    Adjusted R-squared:  0.999
F-statistic:  4317 on 7 and 24 DF,  p-value: < 2.2e-16
```

```r
output_d_2 = data.frame(
  estimate = coef(mod_d_2),
  true = true_values[1:8]
)

output_d_2 |> kable()
```

	estimate	true
(Intercept)	10.1052457	10
A	-0.5967401	2
B	-0.6848735	-1
C	9.4966970	4
D	-0.4275997	3
A:B	6.4491420	4
A:C	-4.0600990	-1
A:D	7.7734936	2

Q3)

a)

Here just copy-pasting from Q1) but with different interactions and printing the dataset.

```r
#set up a true mean value for cases that get no treatments at all
trt0 = 10

#set up true values for the main effects for 4 treatments
trta = 2
trtb = -1
trtc = 4
trtd = 3

#set up true values for the effects for two-way interactions
trtab = 4
trtac = -1
trtad = 2

#these are for use in Problem 3
trtbc = 0
trtbd = 0
trtcd = 0

#we will set all three-way interactions and the four-way interaction
trtabc = 0
trtabd = 0
trtacd = 0
trtbcd = 0
trtabcd = 0

true_values = c(trt0, trta, trtb, trtc, trtd, trtab, trtac, trtad,
                trtbc, trtbd, trtcd)

true_three = c(true_values, 0, 0, 0, 0)

x0 = c(0, 0, 0, 0)
xa = c(1, 0, 0, 0)
xb = c(0, 1, 0, 0)
xc = c(0, 0, 1, 0)
xd = c(0, 0, 0, 1)
xab = c(1, 1, 0, 0)
xac = c(1, 0, 1, 0)
xad = c(1, 0, 0, 1)
xbc = c(0, 1, 1, 0)
xbd = c(0, 1, 0, 1)
```

```
xcd = c(0, 0, 1, 1)
xabc = c(1, 1, 1, 0)
xabd = c(1, 1, 0, 1)
xacd = c(1, 0, 1, 1)
xbcd = c(0, 1, 1, 1)
xabcd = c(1, 1, 1, 1)

## with this information, you can make the datasets needed
#to answer the questions about the full factorial design

n = 4
sigma = 0.2
y0 = rnorm(n, trt0, sigma)
ya = rnorm(n, trt0 + trta, sigma)
yb = rnorm(n, trt0 + trtb, sigma)
yc = rnorm(n, trt0 + trtc, sigma)
yd = rnorm(n, trt0 + trtd, sigma)
yab = rnorm(n, trt0 + trta + trtb + trtab, sigma)
yac = rnorm(n, trt0 + trta + trtc + trtac, sigma)
yad = rnorm(n, trt0 + trta + trtd + trtad, sigma)
ybc = rnorm(n, trt0 + trtb + trtc + trtbc, sigma)
ybd = rnorm(n, trt0 + trtb + trtd + trtbd, sigma)
ycd = rnorm(n, trt0 + trtc + trtd + trtcd, sigma)
yabc = rnorm(n, trt0 + trta + trtb + trtc + trtab + trtbc + trtac + trtabc,
             sigma)
yabd = rnorm(n, trt0 + trta + trtb + trtd + trtab + trtad + trtbd + trtabd,
             sigma)
yacd = rnorm(n, trt0 + trta + trtc + trtd + trtac + trtad + trtcd + trtacd,
             sigma)
ybcd = rnorm(n, trt0 + trtb + trtc + trtd + trtbc + trtbd + trtcd + trtbcd,
             sigma)
yabcd = rnorm(n, trt0 + trta + trtb + trtc + trtd + trtab + trtac + trtad +
                trtbc + trtbd + trtcd + trtabc + trtabd + trtacd + trtbcd +
                trtabcd, sigma)

#make a matrix with 4 rows per treatment group in order of treatment group
thedata = rbind(x0, x0, x0, x0)
thedata = rbind(thedata, xa, xa, xa, xa)
thedata = rbind(thedata, xb, xb, xb, xb)
thedata = rbind(thedata, xc, xc, xc, xc)
thedata = rbind(thedata, xd, xd, xd, xd)
```

```
thedata = rbind(thedata, xab, xab, xab, xab)
thedata = rbind(thedata, xac, xac, xac, xac)
thedata = rbind(thedata, xad, xad, xad, xad)
thedata = rbind(thedata, xbc, xbc, xbc, xbc)
thedata = rbind(thedata, xbd, xbd, xbd, xbd)
thedata = rbind(thedata, xcd, xcd, xcd, xcd)
thedata = rbind(thedata, xabc, xabc, xabc, xabc)
thedata = rbind(thedata, xabd, xabd, xabd, xabd)
thedata = rbind(thedata, xacd, xacd, xacd, xacd)
thedata = rbind(thedata, xbcd, xbcd, xbcd, xbcd)
thedata = rbind(thedata, xabcd, xabcd, xabcd, xabcd)

#now add the y column, which is in the same order as thedata
y = c(y0, ya, yb, yc, yd, yab, yac, yad, ybc, ybd, ycd, yabc, yabd, yacd,
      ybcd, yabcd)
thedata = cbind(thedata, y)
thedata = data.frame(thedata)
names(thedata)[1:4] = c("A", "B", "C", "D")

fractional = thedata[c(1:4, 21:44, 61:64), ]
fractional |> kable()
```

	A	B	C	D	y
x0	0	0	0	0	10.05432
x0.1	0	0	0	0	10.45101
x0.2	0	0	0	0	10.13461
x0.3	0	0	0	0	10.25792
xab	1	1	0	0	15.27589
xab.1	1	1	0	0	15.03127
xab.2	1	1	0	0	15.09762
xab.3	1	1	0	0	15.42749
xac	1	0	1	0	14.97415
xac.1	1	0	1	0	15.14585
xac.2	1	0	1	0	14.76271
xac.3	1	0	1	0	14.99227
xad	1	0	0	1	17.10951
xad.1	1	0	0	1	17.09733
xad.2	1	0	0	1	17.02391
xad.3	1	0	0	1	17.01811
xbc	0	1	1	0	13.21087
xbc.1	0	1	1	0	12.91088
xbc.2	0	1	1	0	13.06563
xbc.3	0	1	1	0	13.22648
xbd	0	1	0	1	11.88612
xbd.1	0	1	0	1	11.68837
xbd.2	0	1	0	1	12.04686
xbd.3	0	1	0	1	11.59067
xcd	0	0	1	1	16.92600
xcd.1	0	0	1	1	16.89804
xcd.2	0	0	1	1	17.24512
xcd.3	0	0	1	1	16.95810
xabcd	1	1	1	1	23.35939
xabcd.1	1	1	1	1	23.19097
xabcd.2	1	1	1	1	23.02707
xabcd.3	1	1	1	1	23.36365

b)

Yes, confirming that we can only fit 3 interactions here.

```
mod_b_3 = lm(y ~ A*B + A*C + A*D + B*C + B*D + C*D , data = fractional)
summary(mod_b_3)
```

```
Call:
lm(formula = y ~ A * B + A * C + A * D + B * C + B * D + C *
    D, data = fractional)

Residuals:
     Min       1Q   Median       3Q      Max
-0.21234 -0.10920 -0.01622  0.11131  0.24386

Coefficients: (3 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.22447    0.07981  128.11  < 2e-16 ***
A            1.77741    0.11287   15.75 3.75e-14 ***
B           -1.16241    0.07981  -14.56 2.06e-13 ***
C            4.04141    0.07981   50.64  < 2e-16 ***
D            2.74094    0.07981   34.34  < 2e-16 ***
A:B          4.36859    0.11287   38.71  < 2e-16 ***
A:C         -1.07454    0.11287   -9.52 1.27e-09 ***
A:D          2.31939    0.11287   20.55  < 2e-16 ***
B:C               NA         NA      NA       NA
B:D               NA         NA      NA       NA
C:D               NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1596 on 24 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9982
F-statistic:  2510 on 7 and 24 DF,  p-value: < 2.2e-16
```

c)

Yes, these estimates look much better compared to Question 2, since now we know that the true interaction effects for the aliased variables are 0.

```r
mod_c_3 = lm(y ~ A*B + A*C + A*D, data = fractional)
summary(mod_c_3)
```

```
Call:
lm(formula = y ~ A * B + A * C + A * D, data = fractional)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-0.21234 -0.10920 -0.01622  0.11131  0.24386

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.22447    0.07981  128.11  < 2e-16 ***
A            1.77741    0.11287   15.75 3.75e-14 ***
B           -1.16241    0.07981  -14.56 2.06e-13 ***
C            4.04141    0.07981   50.64  < 2e-16 ***
D            2.74094    0.07981   34.34  < 2e-16 ***
A:B          4.36859    0.11287   38.71  < 2e-16 ***
A:C         -1.07454    0.11287   -9.52 1.27e-09 ***
A:D          2.31939    0.11287   20.55  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1596 on 24 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9982
F-statistic:  2510 on 7 and 24 DF,  p-value: < 2.2e-16
```

```r
output_d_3 = data.frame(
  estimate = coef(mod_c_3),
  true = true_values[1:8]
)

output_d_3 |> kable()
```

	estimate	true
(Intercept)	10.224467	10
A	1.777411	2
B	-1.162405	-1
C	4.041405	4
D	2.740944	3
A:B	4.368595	4
A:C	-1.074539	-1
A:D	2.319392	2