Chapter 8 (Edition 8): 8.1, 8.10, 8.12, 8.14, 8.51

Jeremy Ling & Emmanuel Mejia

May 10, 2018

```
# loading libraries
library(gplots)
```

```
## Warning: package 'gplots' was built under R version 3.4.4
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(car)
```

8.1

Suppose that in the chemical process development experiment described in Problem 6.7, it was only possible to run a one-half fraction of the 2^4 design. Construct the design and perform the statistical analysis, using the data from replicate I.

```
#chemical data from 6.7
rep1 = c(90,74,81,83,77,81,88,73,98,72,87,85,99,79,87,80)
#rep2 = c(93,78,85,80,78,80,82,70,95,76,83,86,90,75,84,80)
A <- rep(x = c("-", "+"), times = 8)
B <- rep(x = c("-", "+"), each = 2, times = 4)
C <- rep(x = c("-", "+"), each = 4, times = 2)
D <- rep(x = c("-", "+"), each = 8)
#data
chemical = data.frame(A,B,C,D,rep1)

coded=function(x) #a function to code variable x
{
  ifelse(x=="+", 1, -1)
}
for (j in 1:4)
  chemical[, j]=as.numeric(coded(chemical[, j]))

fraction.chem=with(chemical, chemical[A * B * C * D == 1,])

#linear model
chem.lm = lm(rep1 ~ A*B*C*D, fraction.chem); summary(chem.lm)
```

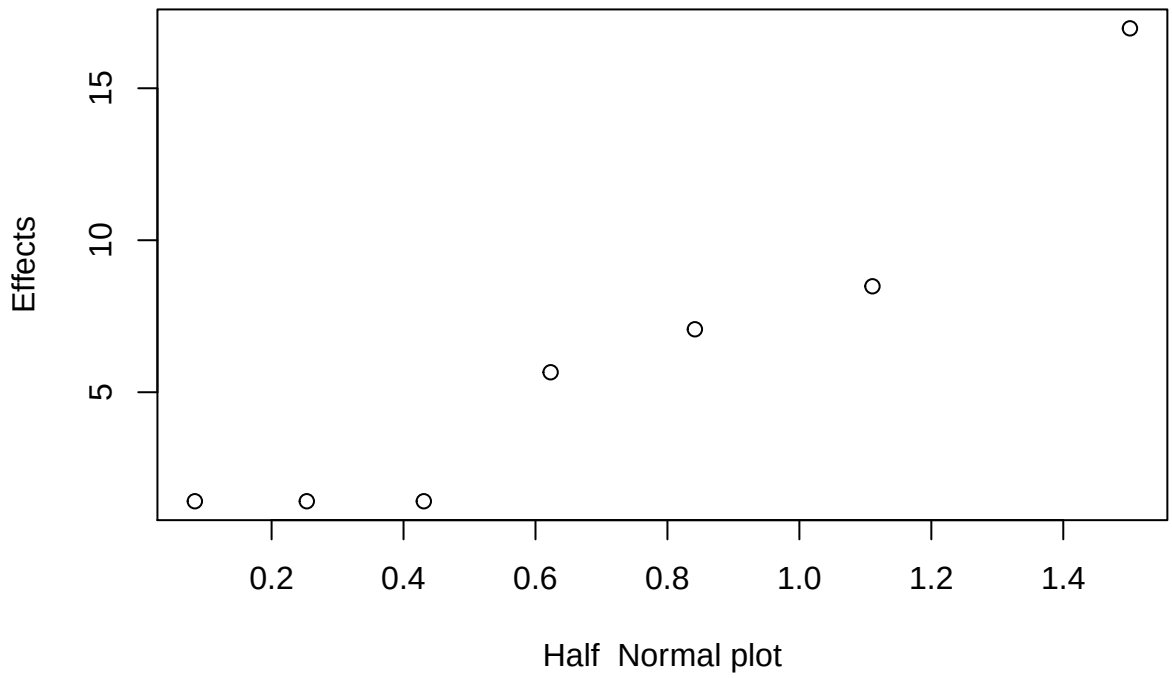
```
##
## Call:
## lm(formula = rep1 ~ A * B * C * D, data = fraction.chem)
##
## Residuals:
## ALL 8 residuals are 0: no residual degrees of freedom!
```

```
##
## Coefficients: (8 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     85.0         NA      NA       NA
## A               -6.0         NA      NA       NA
## B               -0.5         NA      NA       NA
## C                2.0         NA      NA       NA
## D               -0.5         NA      NA       NA
## A:B              3.0         NA      NA       NA
## A:C             -0.5         NA      NA       NA
## B:C             -2.5         NA      NA       NA
## A:D               NA         NA      NA       NA
## B:D               NA         NA      NA       NA
## C:D               NA         NA      NA       NA
## A:B:C             NA         NA      NA       NA
## A:B:D             NA         NA      NA       NA
## A:C:D             NA         NA      NA       NA
## B:C:D             NA         NA      NA       NA
## A:B:C:D           NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 7 and 0 DF,  p-value: NA
```

```
#alias
alias(chem.lm)
```

```
## Model :
## rep1 ~ A * B * C * D
##
## Complete :
##         (Intercept) A B C D A:B A:C B:C
## A:D     0           0 0 0 0 0   0   1
## B:D     0           0 0 0 0 0   1   0
## C:D     0           0 0 0 0 1   0   0
## A:B:C   0           0 0 0 1 0   0   0
## A:B:D   0           0 0 1 0 0   0   0
## A:C:D   0           0 1 0 0 0   0   0
## B:C:D   0           1 0 0 0 0   0   0
## A:B:C:D 1           0 0 0 0 0   0   0
```

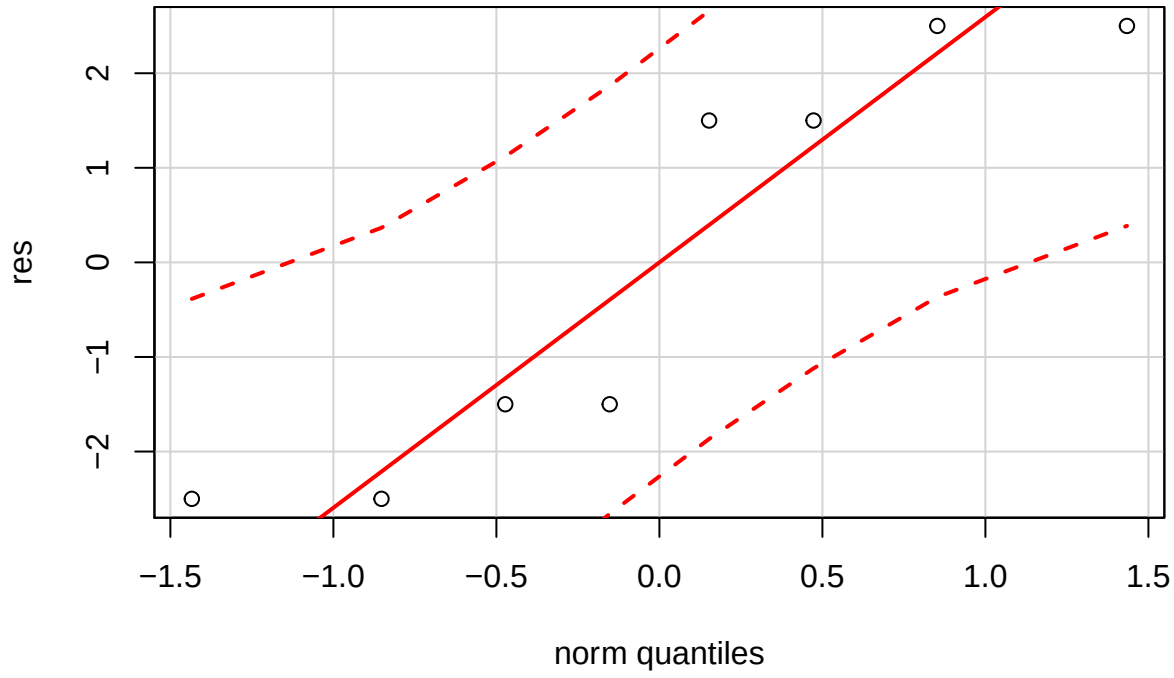
```
#normal probability plot
qqnorm(aov(rep1 ~ A * B * C * D, fraction.chem), label = TRUE)#A, C, AB, BC
```



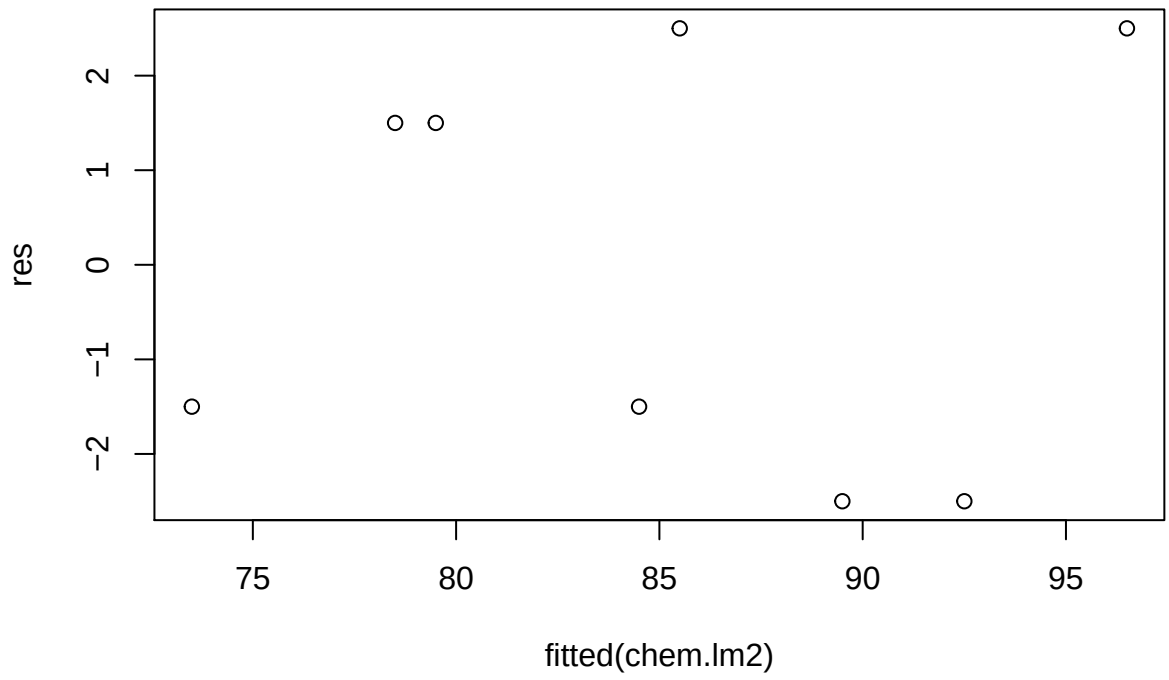

```
#refine model
chem.lm2 = lm(rep1 ~ A*B + A*D, fraction.chem); summary(chem.lm2)
```

```
##
## Call:
## lm(formula = rep1 ~ A * B + A * D, data = fraction.chem)
##
## Residuals:
##    1    4    6    7   10   11   13   16
## -2.5 -1.5  1.5  2.5 -1.5 -2.5  2.5  1.5
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   85.000      1.458  58.310 0.000294 ***
## A             -6.000      1.458  -4.116 0.054268 .
## B             -0.500      1.458  -0.343 0.764298
## D             -0.500      1.458  -0.343 0.764298
## A:B            3.000      1.458   2.058 0.175837
## A:D           -2.500      1.458  -1.715 0.228483
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.123 on 2 degrees of freedom
## Multiple R-squared:  0.9241, Adjusted R-squared:  0.7344
## F-statistic: 4.871 on 5 and 2 DF,  p-value: 0.1791
```

```r
#Residual Analysis
res = fraction.chem$rep1 - fitted(chem.lm2)
qqPlot(res)
```




```r
plot(fitted(chem.lm2), res)
```



We perform our analysis, check our aliases, and do half normal probability. We see that interaction effects AB + AD and their main effects have the largest effect on the response variable, we quickly check our residuals and everything seems good. Our model is good.

8.10

An article by J. J. Pignatiello Jr. and J. S. Ramberg in the Journal of Quality Technology (Vol. 17, 1985, pp. 198-206) describes the use of a replicated fractional factorial to investigate the effect of five factors on the free height of leaf springs used in an automotive application. The factors are A = furnace temperature, B = heating time, C = transfer time, D = hold down time, and E = quench oil temperature. The data are shown in Table P8.1

(a) Write out the alias structure for this design. What is the resolution of this design?

We design a half factorial design with resolution V and generator ABCDE. The alias structure for this design is shown below:

$[A] \rightarrow A + BCDE$ $[B] \rightarrow B + ACDE$ $[C] \rightarrow C + ABDE$ $[D] \rightarrow D + ABCE$ $[E] \rightarrow E + ABCD$ $[AB] \rightarrow AB + CDE$ $[AC] \rightarrow AC + BDE$ $[AD] \rightarrow AD + BCE$ $[AE] \rightarrow AE + BCD$ $[BC] \rightarrow BC + ADE$ $[BD] \rightarrow BD + ACE$ $[BE] \rightarrow BE + ACD$ $[CD] \rightarrow CD + ABE$ $[CE] \rightarrow CE + ABD$ $[DE] \rightarrow DE + ABC$

```
# declaring data
A <- rep(x = c("-", "+"), times = 8)
B <- rep(x = c("-", "+"), each = 2, times = 4)
C <- rep(x = c("-", "+"), each = 4, times = 2)
D <- c("-", "+", "+", "-", "+", "-", "-", "+", "-", "+", "+", "-", "+", "-", "-", "+")
E <- rep(x = c("-", "+"), each = 8)
```

```r
FH1 <- c(7.78, 8.15, 7.5, 7.59, 7.54, 7.69, 7.56, 7.56, 7.5, 7.88, 7.5, 7.63, 7.32, 7.56, 7.18, 7.81)
FH2 <- c(7.78, 8.18, 7.56, 7.56, 8, 8.09, 7.52, 7.81, 7.25, 7.88, 7.56, 7.75, 7.44, 7.69, 7.18, 7.5)
FH3 <- c(7.81, 7.88, 7.5, 7.75, 7.88, 8.06, 7.44, 7.69, 7.12, 7.44, 7.5, 7.56, 7.44, 7.62, 7.25, 7.59)

# creating table
A <- c(A, A, A)
B <- c(B, B, B)
C <- c(C, C, C)
D <- c(D, D, D)
E <- c(E, E, E)
FH <- as.numeric(c(FH1, FH2, FH3))

spring <- data.frame(cbind(A, B, C, D, E, FH))

# defining coded
coded=function(x)
{
  ifelse(x=="+", 1, -1)
}

# decoding data
for (j in 1:5)
  spring[, j]=as.numeric(coded(spring[, j]))

# defining fraction
#fraction <- with(spring, spring[A * B * C * D * E == 1,])

# linear regression
summary(lm(as.numeric(FH) ~ A * B * C * D * E, spring))
```

```
##
## Call:
## lm(formula = as.numeric(FH) ~ A * B * C * D * E, data = spring)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.000 -1.000  0.000  2.000  5.333
##
## Coefficients: (16 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.52083    0.47324  22.231  < 2e-16 ***
## A            2.89583    0.47324   6.119 7.69e-07 ***
## B           -1.85417    0.47324  -3.918 0.000441 ***
## C           -0.52083    0.47324  -1.101 0.279297
## D            0.64583    0.47324   1.365 0.181868
## E           -2.56250    0.47324  -5.415 5.94e-06 ***
## A:B         -0.06250    0.47324  -0.132 0.895758
## A:C          0.02083    0.47324   0.044 0.965160
## B:C          0.02083    0.47324   0.044 0.965160
## A:D               NA         NA      NA       NA
## B:D               NA         NA      NA       NA
## C:D               NA         NA      NA       NA
## A:E          0.72917    0.47324   1.541 0.133200
## B:E          1.81250    0.47324   3.830 0.000563 ***
```

```
## C:E         -0.27083    0.47324  -0.572 0.571123
## D:E          0.22917    0.47324   0.484 0.631508
## A:B:C             NA         NA      NA       NA
## A:B:D             NA         NA      NA       NA
## A:C:D             NA         NA      NA       NA
## B:C:D             NA         NA      NA       NA
## A:B:E       -0.31250    0.47324  -0.660 0.513762
## A:C:E        0.02083    0.47324   0.044 0.965160
## B:C:E       -0.64583    0.47324  -1.365 0.181868
## A:D:E             NA         NA      NA       NA
## B:D:E             NA         NA      NA       NA
## C:D:E             NA         NA      NA       NA
## A:B:C:D           NA         NA      NA       NA
## A:B:C:E           NA         NA      NA       NA
## A:B:D:E           NA         NA      NA       NA
## A:C:D:E           NA         NA      NA       NA
## B:C:D:E           NA         NA      NA       NA
## A:B:C:D:E         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.279 on 32 degrees of freedom
## Multiple R-squared:  0.7666, Adjusted R-squared:  0.6572
## F-statistic: 7.008 on 15 and 32 DF,  p-value: 2.127e-06
```

```
# alias structure
alias(lm(as.numeric(FH) ~ A * B * C * D * E, spring))
```

```
## Model :
## as.numeric(FH) ~ A * B * C * D * E
##
## Complete :
##           (Intercept) A B C D E A:B A:C B:C A:E B:E C:E D:E A:B:E A:C:E
## A:D       0           0 0 0 0 0 0   0   1   0   0   0   0   0     0
## B:D       0           0 0 0 0 0 0   1   0   0   0   0   0   0     0
## C:D       0           0 0 0 0 0 1   0   0   0   0   0   0   0     0
## A:B:C     0           0 0 0 1 0 0   0   0   0   0   0   0   0     0
## A:B:D     0           0 0 1 0 0 0   0   0   0   0   0   0   0     0
## A:C:D     0           0 1 0 0 0 0   0   0   0   0   0   0   0     0
## B:C:D     0           1 0 0 0 0 0   0   0   0   0   0   0   0     0
## A:D:E     0           0 0 0 0 0 0   0   0   0   0   0   0   0     0
## B:D:E     0           0 0 0 0 0 0   0   0   0   0   0   0   0     1
## C:D:E     0           0 0 0 0 0 0   0   0   0   0   0   0   1     0
## A:B:C:D   1           0 0 0 0 0 0   0   0   0   0   0   0   0     0
## A:B:C:E   0           0 0 0 0 0 0   0   0   0   0   0   1   0     0
## A:B:D:E   0           0 0 0 0 0 0   0   0   0   0   1   0   0     0
## A:C:D:E   0           0 0 0 0 0 0   0   0   0   1   0   0   0     0
## B:C:D:E   0           0 0 0 0 0 0   0   0   1   0   0   0   0     0
## A:B:C:D:E 0           0 0 0 0 1 0   0   0   0   0   0   0   0     0
##           B:C:E
## A:D       0
## B:D       0
## C:D       0
## A:B:C     0
## A:B:D     0
```

```
## A:C:D     0
## B:C:D     0
## A:D:E     1
## B:D:E     0
## C:D:E     0
## A:B:C:D   0
## A:B:C:E   0
## A:B:D:E   0
## A:C:D:E   0
## B:C:D:E   0
## A:B:C:D:E 0
```

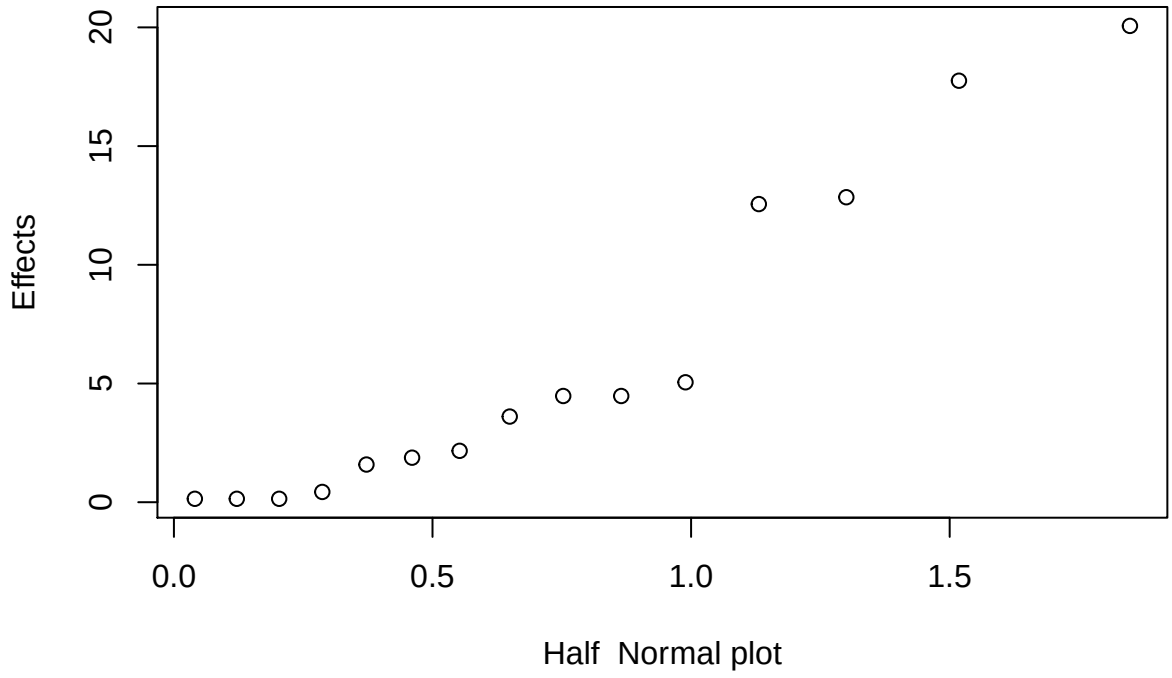
(b) Analyze the data. What factors influence the mean free height?

```
# linear regression
spring.lm <- lm(as.numeric(FH) ~ A * B * C * D * E, spring)
summary(spring.lm)
```

```
##
## Call:
## lm(formula = as.numeric(FH) ~ A * B * C * D * E, data = spring)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.000 -1.000  0.000  2.000  5.333
##
## Coefficients: (16 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.52083    0.47324  22.231  < 2e-16 ***
## A            2.89583    0.47324   6.119 7.69e-07 ***
## B           -1.85417    0.47324  -3.918 0.000441 ***
## C           -0.52083    0.47324  -1.101 0.279297
## D            0.64583    0.47324   1.365 0.181868
## E           -2.56250    0.47324  -5.415 5.94e-06 ***
## A:B         -0.06250    0.47324  -0.132 0.895758
## A:C          0.02083    0.47324   0.044 0.965160
## B:C          0.02083    0.47324   0.044 0.965160
## A:D               NA         NA      NA       NA
## B:D               NA         NA      NA       NA
## C:D               NA         NA      NA       NA
## A:E          0.72917    0.47324   1.541 0.133200
## B:E          1.81250    0.47324   3.830 0.000563 ***
## C:E         -0.27083    0.47324  -0.572 0.571123
## D:E          0.22917    0.47324   0.484 0.631508
## A:B:C             NA         NA      NA       NA
## A:B:D             NA         NA      NA       NA
## A:C:D             NA         NA      NA       NA
## B:C:D             NA         NA      NA       NA
## A:B:E       -0.31250    0.47324  -0.660 0.513762
## A:C:E        0.02083    0.47324   0.044 0.965160
## B:C:E       -0.64583    0.47324  -1.365 0.181868
## A:D:E             NA         NA      NA       NA
## B:D:E             NA         NA      NA       NA
## C:D:E             NA         NA      NA       NA
## A:B:C:D           NA         NA      NA       NA
```

```
## A:B:C:E           NA         NA      NA       NA
## A:B:D:E           NA         NA      NA       NA
## A:C:D:E           NA         NA      NA       NA
## B:C:D:E           NA         NA      NA       NA
## A:B:C:D:E         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.279 on 32 degrees of freedom
## Multiple R-squared:  0.7666, Adjusted R-squared:  0.6572
## F-statistic: 7.008 on 15 and 32 DF,  p-value: 2.127e-06
```

```
# half normal probability plot
qqnorm(aov(as.numeric(FH) ~ A * B * C * D * E, spring), label = TRUE)
```



The factors that influence mean free height are A, B, E, BE. We can create a reduced model using this information.

```
# new linear regression
spring.lm2 <- lm(as.numeric(FH) ~ A + B * E, spring)
summary(spring.lm2)
```

```
##
## Call:
## lm(formula = as.numeric(FH) ~ A + B * E, data = spring)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -6.6458 -2.0417  0.2292  2.1146  6.1042
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.5208     0.4583  22.955  < 2e-16 ***
## A             2.8958     0.4583   6.318 1.26e-07 ***
## B            -1.8542     0.4583  -4.046 0.000213 ***
## E            -2.5625     0.4583  -5.591 1.43e-06 ***
## B:E           1.8125     0.4583   3.955 0.000282 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.175 on 43 degrees of freedom
## Multiple R-squared:  0.7059, Adjusted R-squared:  0.6785
## F-statistic:  25.8 on 4 and 43 DF,  p-value: 6.066e-11
```

(c) Calculate the range and standard deviation of the free height for each run. Is there any indication that any of these factors affects variability in the free height?

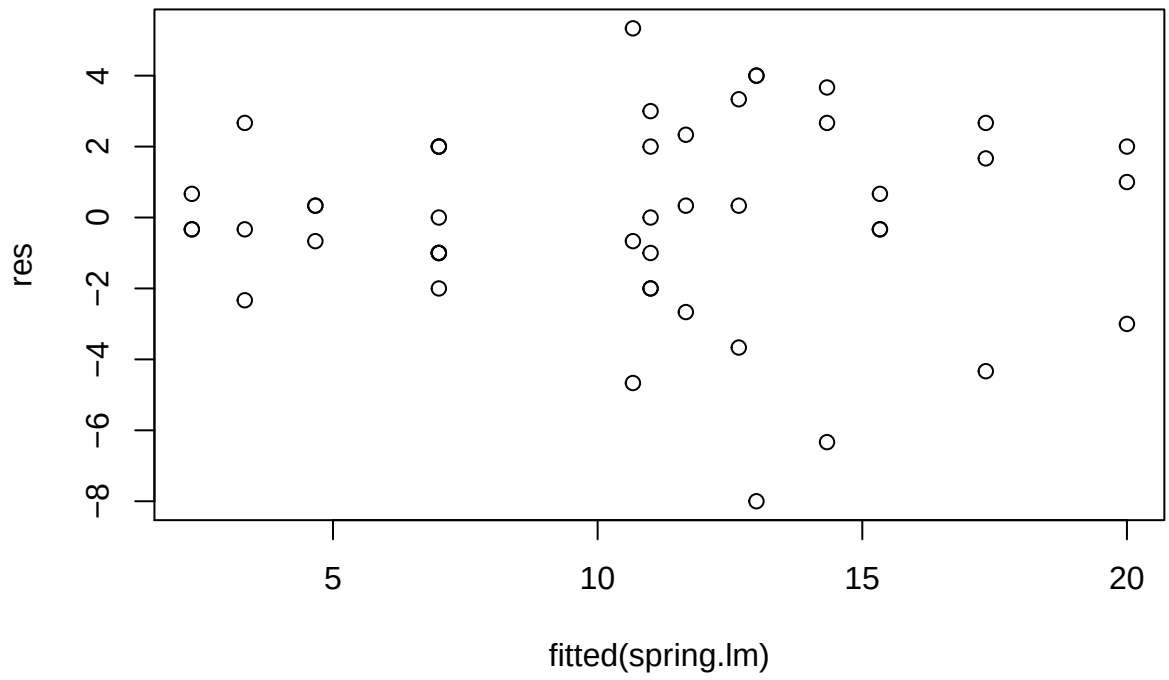
```
# min max and sd
data.frame("Free Height 1" = c(range(FH1), sd(FH1)),
           "Free Height 2" = c(range(FH2), sd(FH2)),
           "Free Height 3" = c(range(FH3), sd(FH3)),
           row.names = c("min", "max", "sd"))
```

```
##     Free.Height.1 Free.Height.2 Free.Height.3
## min     7.1800000     7.1800000     7.1200000
## max     8.1500000     8.1800000     8.0600000
## sd      0.2248398     0.2810746     0.2442122
```

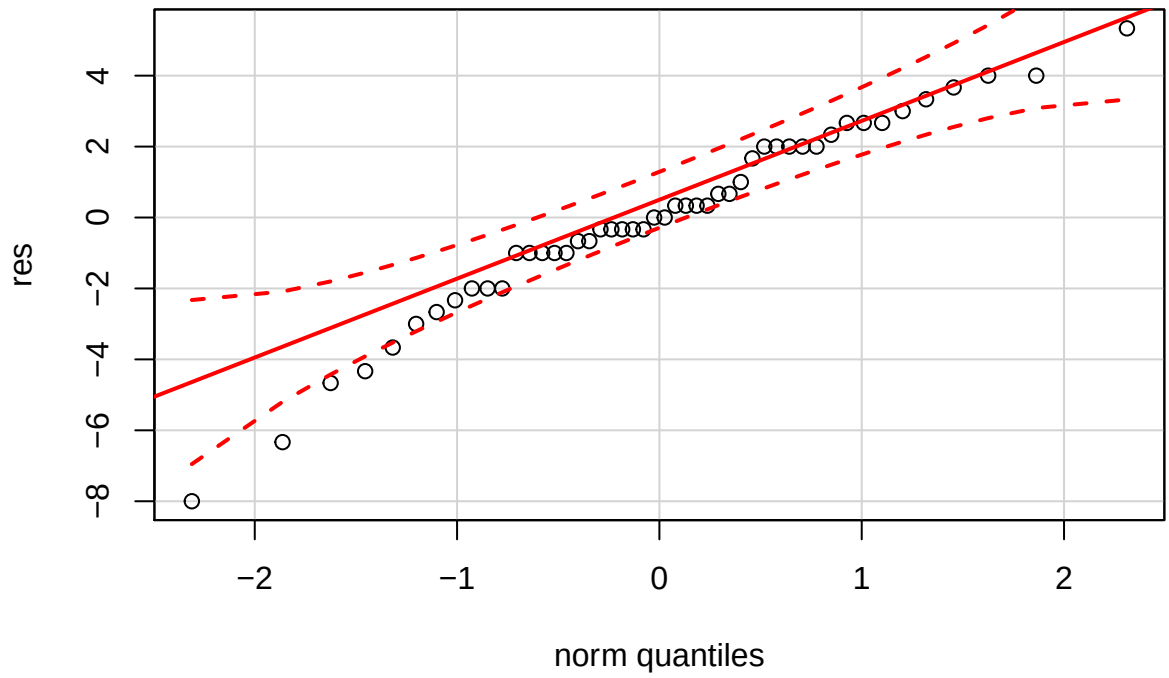
With similar values between all replicates, there's no obvious sign that range or standard devaition of the free height for each run affects variability in free height.

(d) Analyze the residuals from this experiment, and comment on your findings.

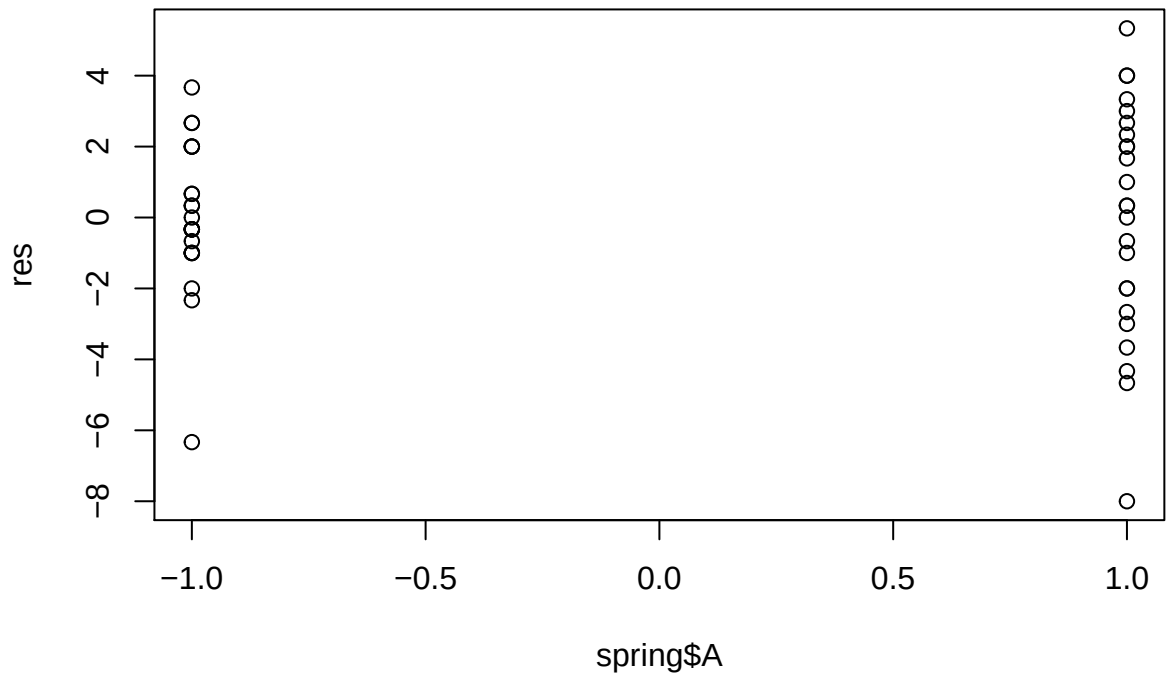
```
# plotting residuals
res <- as.numeric(spring$FH) - fitted(spring.lm)
plot(fitted(spring.lm), res)
```



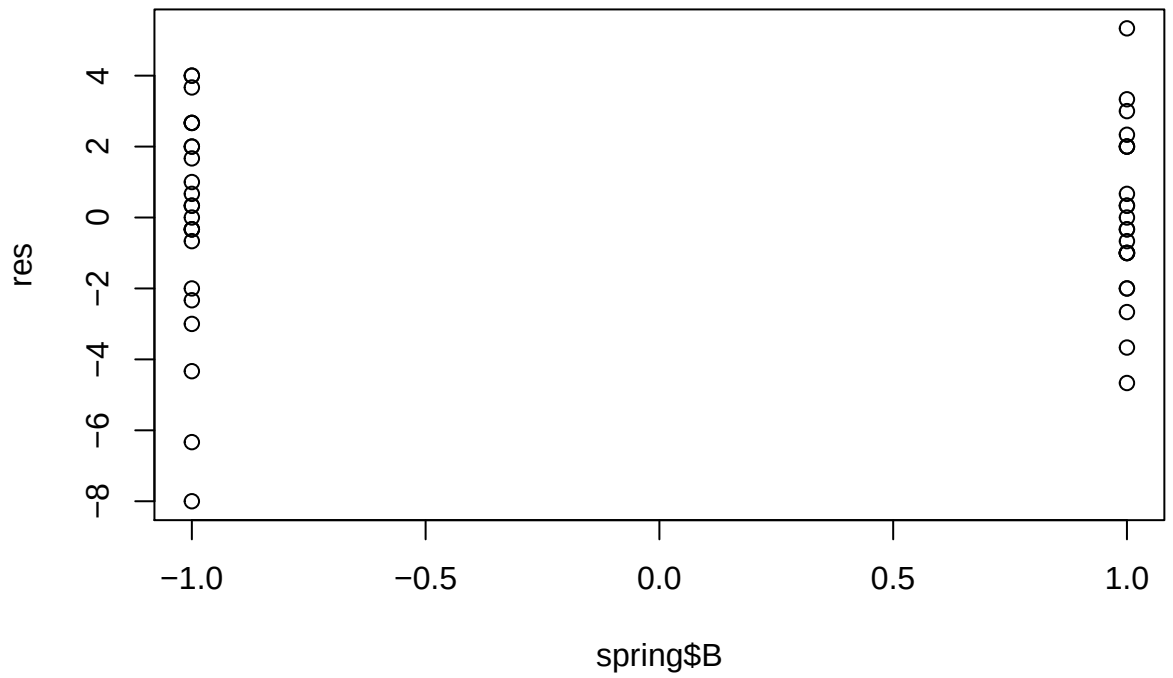

```
qqPlot(res)
```



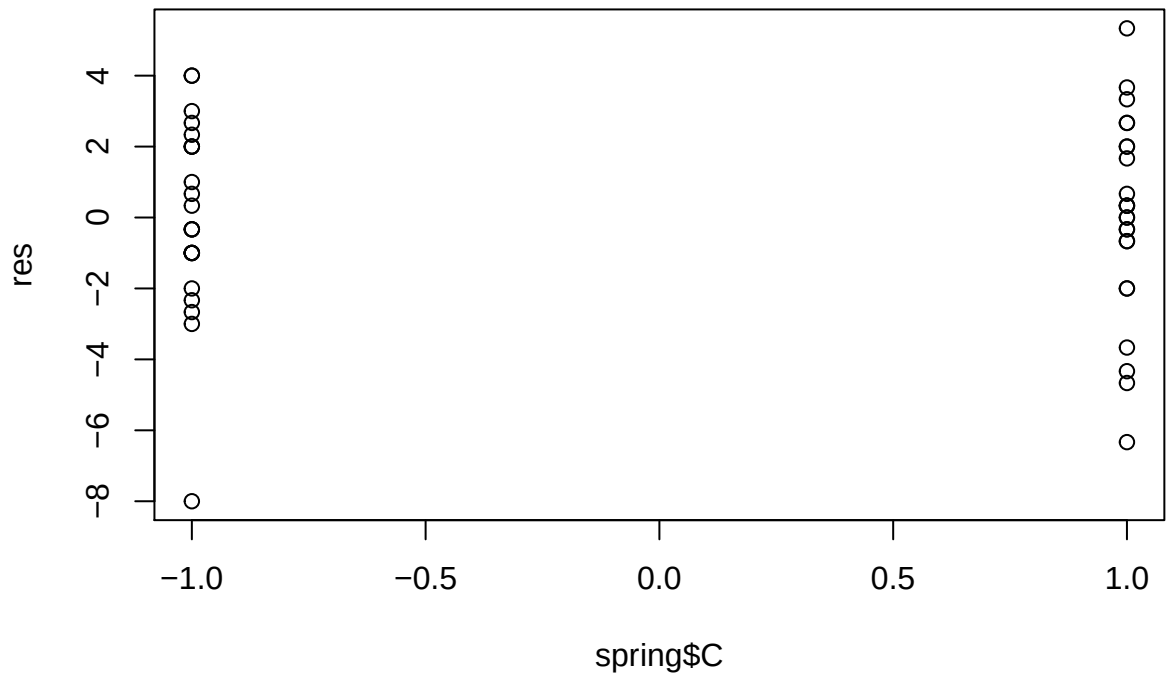

```
plot(spring$A, res)
```



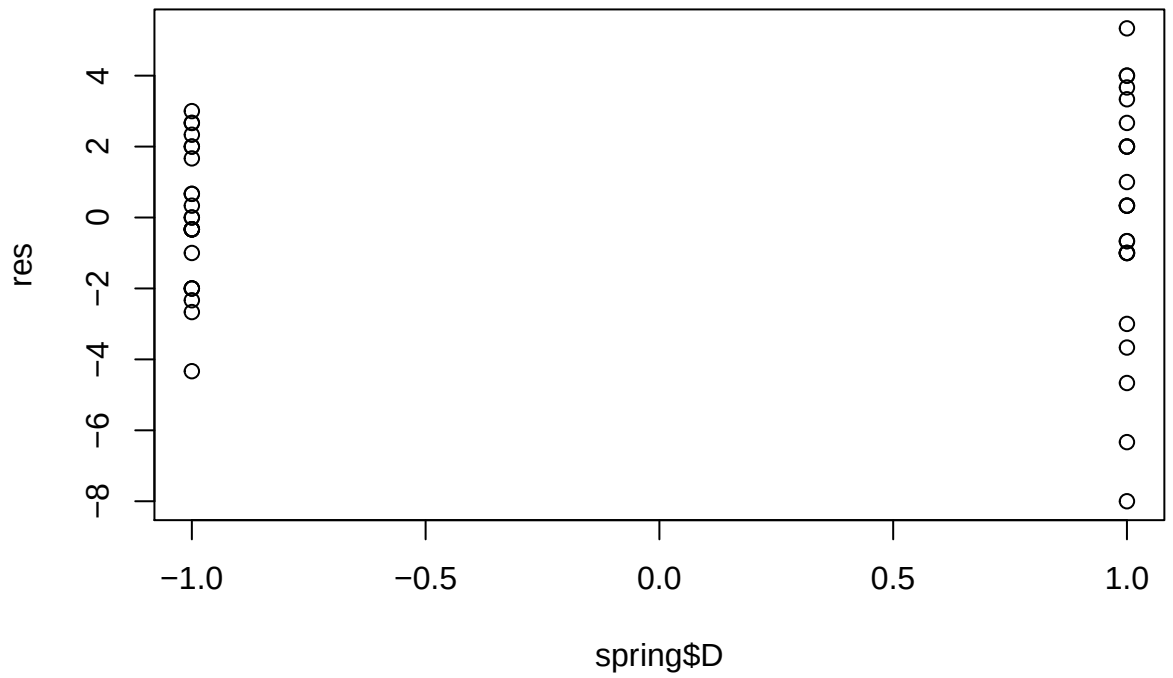

```
plot(spring$B, res)
```



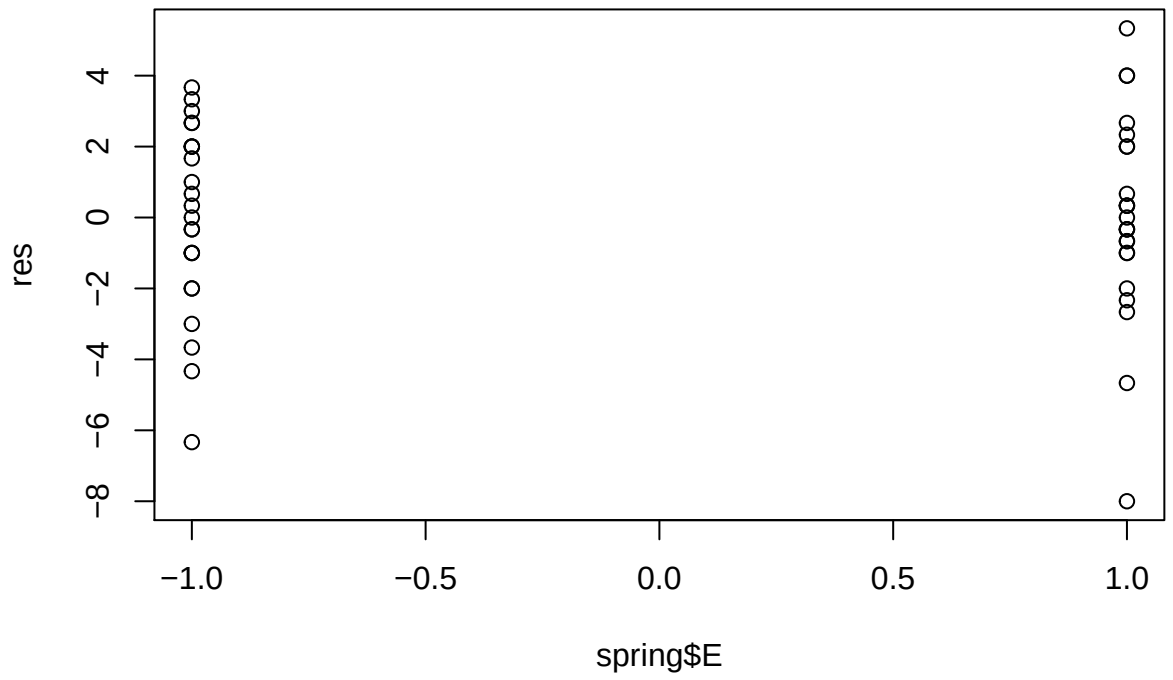

```
plot(spring$C, res)
```



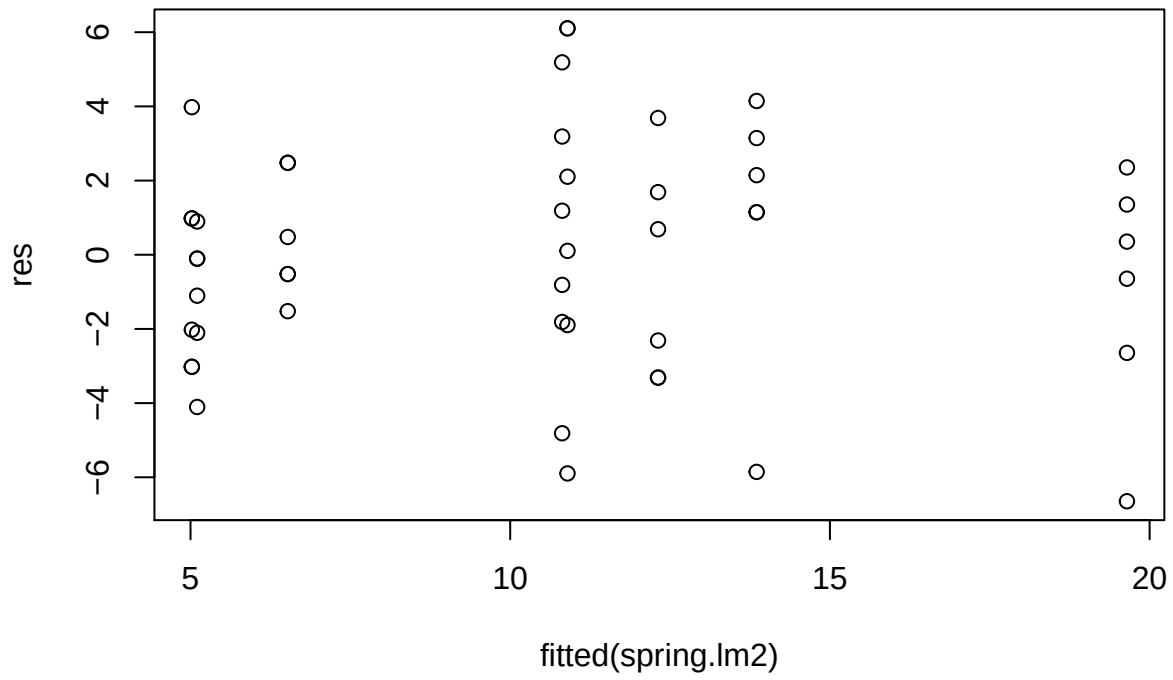

```
plot(spring$D, res)
```



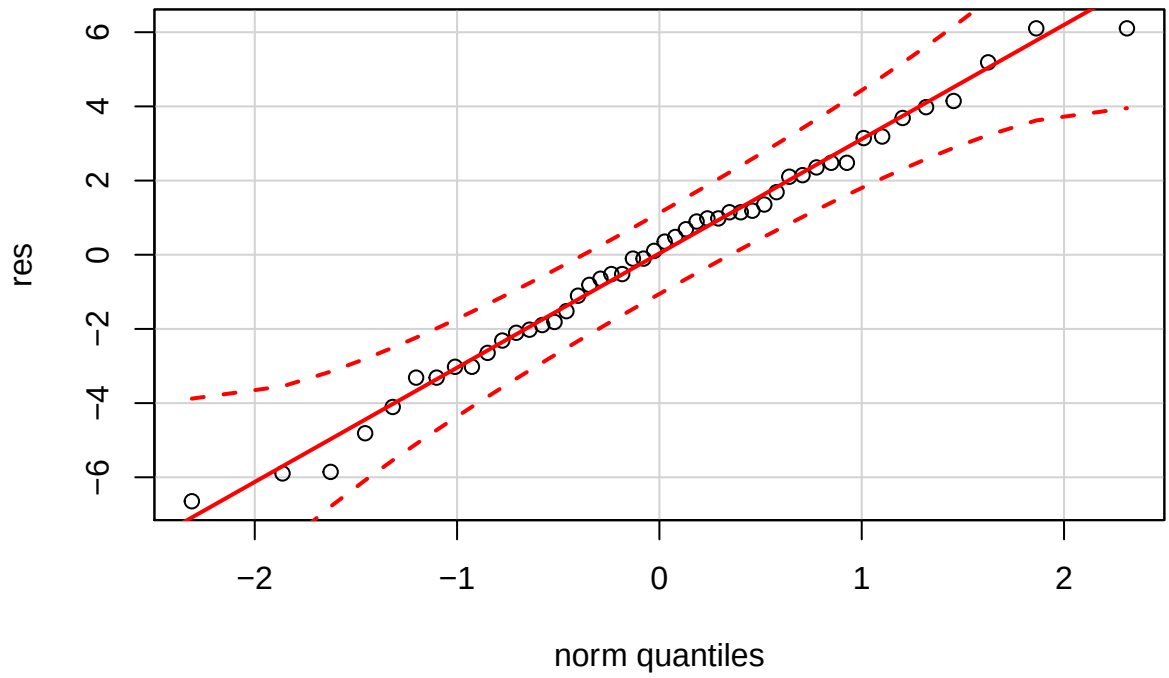

```
plot(spring$E, res)
```



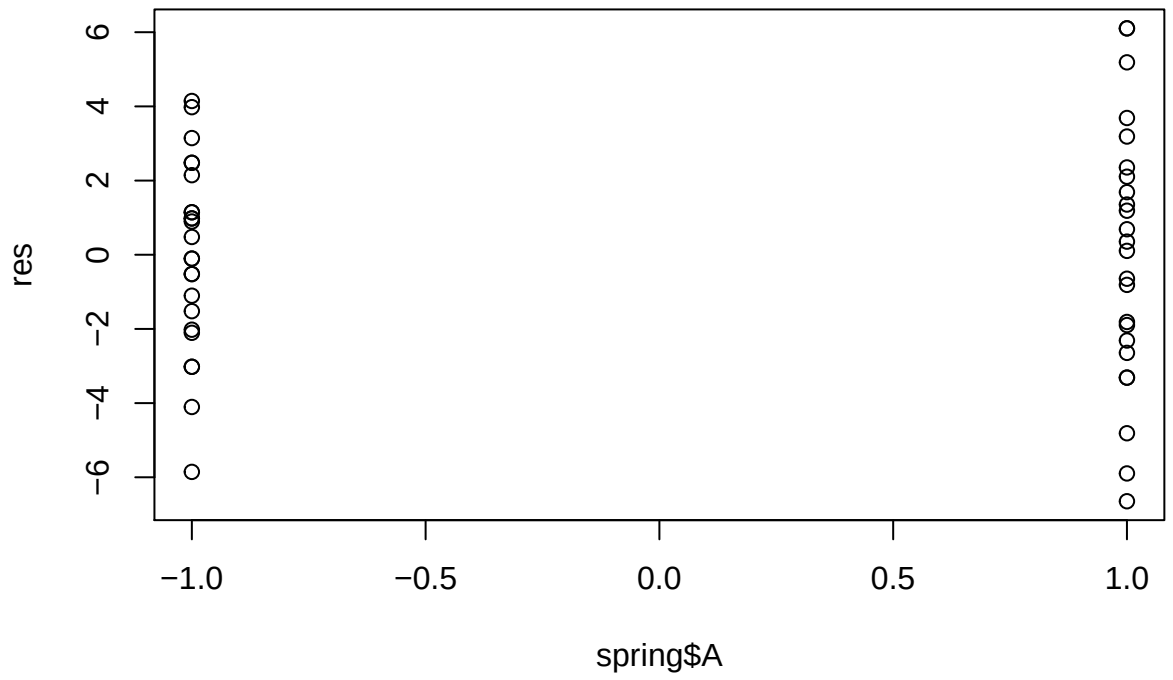

```
# plotting new model residuals
res <- as.numeric(spring$FH) - fitted(spring.lm2)
plot(fitted(spring.lm2), res)
```



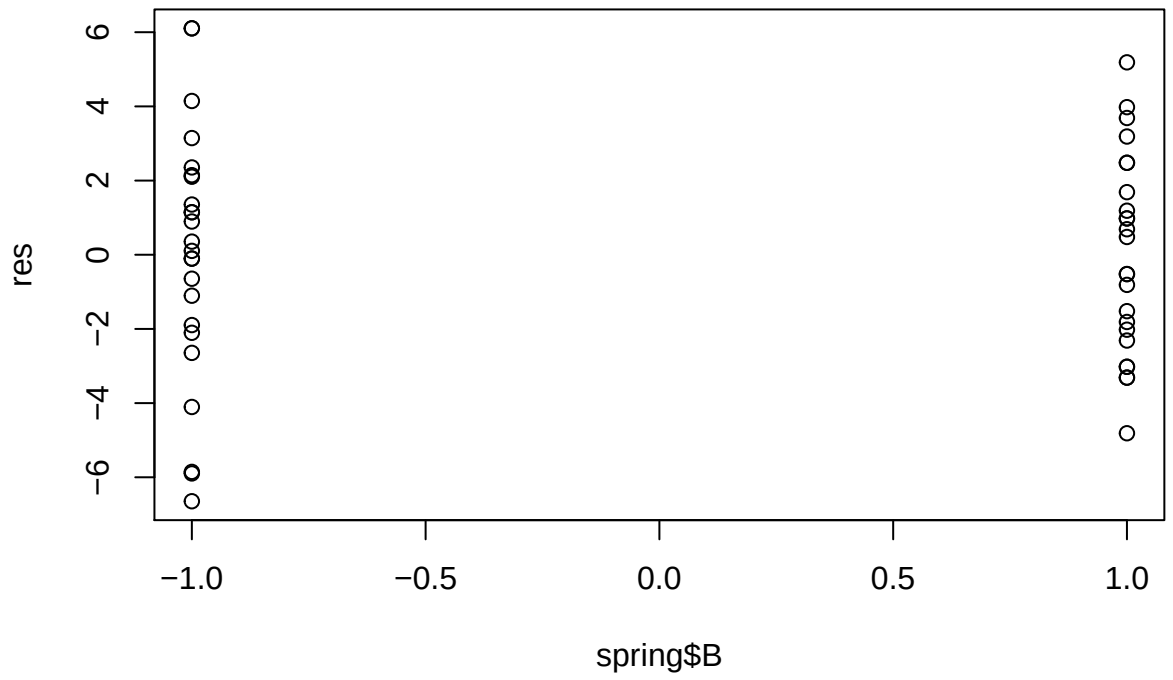

```
qqPlot(res)
```



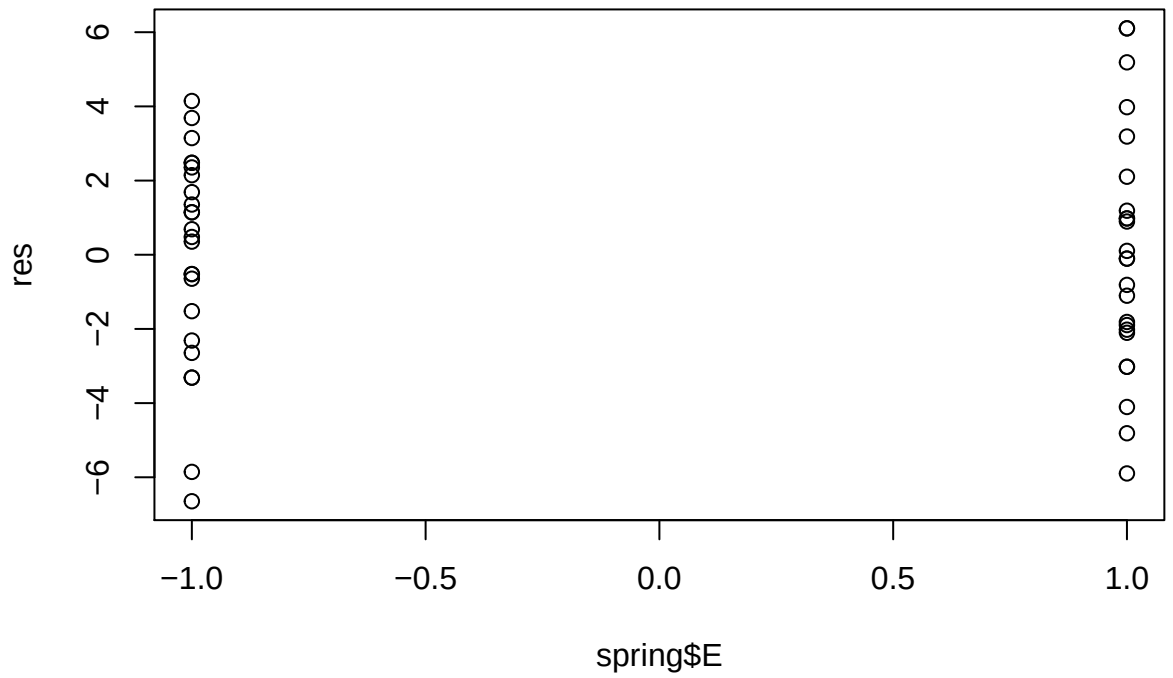

```
plot(spring$A, res)
```




```
plot(spring$B, res)
```




```
plot(spring$E, res)
```



Both models are normally distributed and have residuals who's variances are homogenous. The models are appropriate.

(e) Is this the best possible design for five factors in 16 runs? Specifically, can you find a fractional design for five factors in 16 runs with a higher resolution than this one?

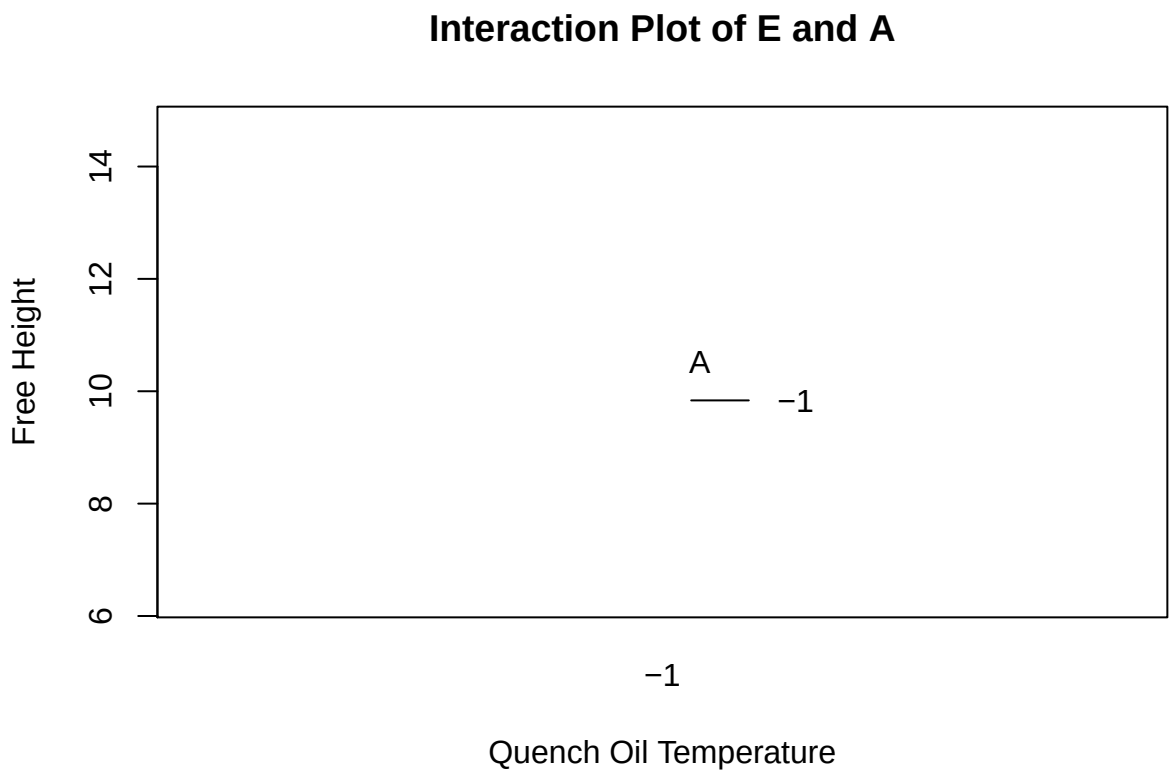
It shouldn't be possible to find a fractional design with a higher resolution that our current one, since it's already a 5 resolution design.

8.12

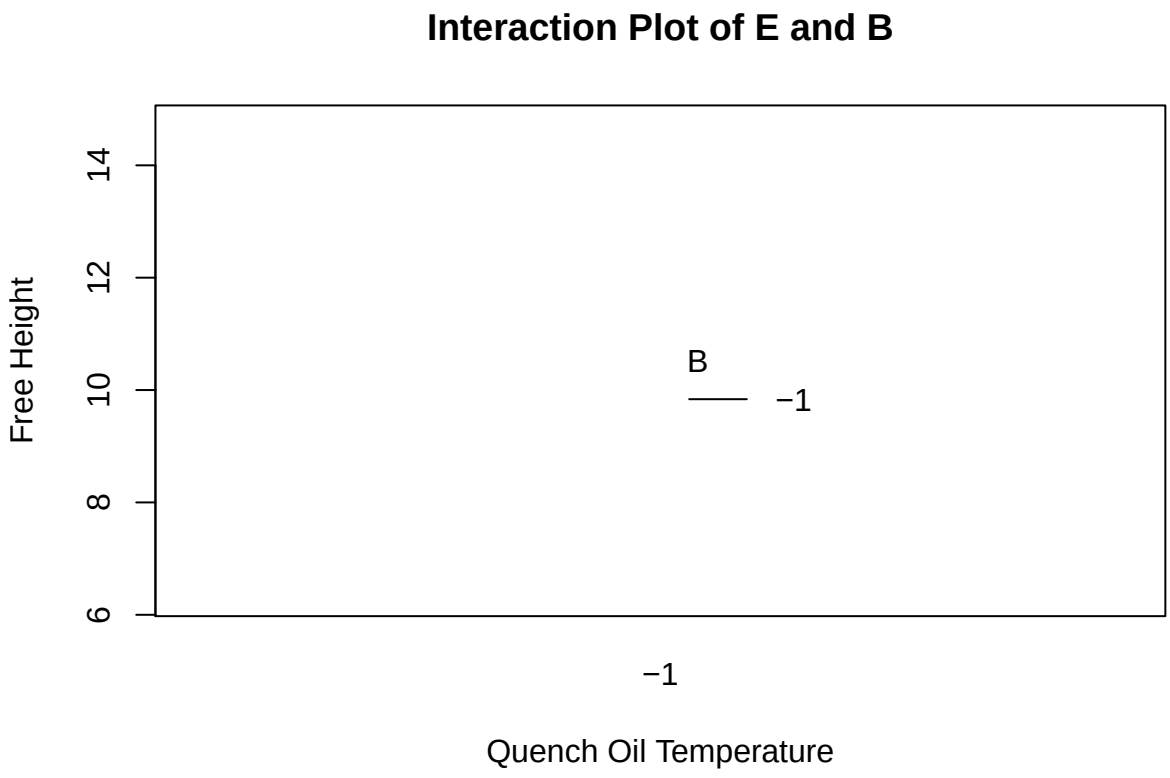
Consider the leaf spring experiment in Problem 8.7. Suppose that factor E (quench oil temperature) is very difficult to control during manufacturing. Where would you set factors A, B, C, and D to reduce variability in the free height as much as possible regardless of the quench oil temperature used? Note refer to 8.10 instead of 8.7

```
# declaring data
A <- rep(x = c("-", "+"), times = 8)
B <- rep(x = c("-", "+"), each = 2, times = 4)
C <- rep(x = c("-", "+"), each = 4, times = 2)
D <- c("-", "+", "+", "-", "+", "-", "-", "+", "-", "+", "+", "-", "+", "-", "-", "+")
E <- rep(x = c("-", "+"), each = 8)
FH1 <- c(7.78, 8.15, 7.5, 7.59, 7.54, 7.69, 7.56, 7.56, 7.5, 7.88, 7.5, 7.63, 7.32, 7.56, 7.18, 7.81)
FH2 <- c(7.78, 8.18, 7.56, 7.56, 8, 8.09, 7.52, 7.81, 7.25, 7.88, 7.56, 7.75, 7.44, 7.69, 7.18, 7.5)
FH3 <- c(7.81, 7.88, 7.5, 7.75, 7.88, 8.06, 7.44, 7.69, 7.12, 7.44, 7.5, 7.56, 7.44, 7.62, 7.25, 7.59)

# defining coded
```

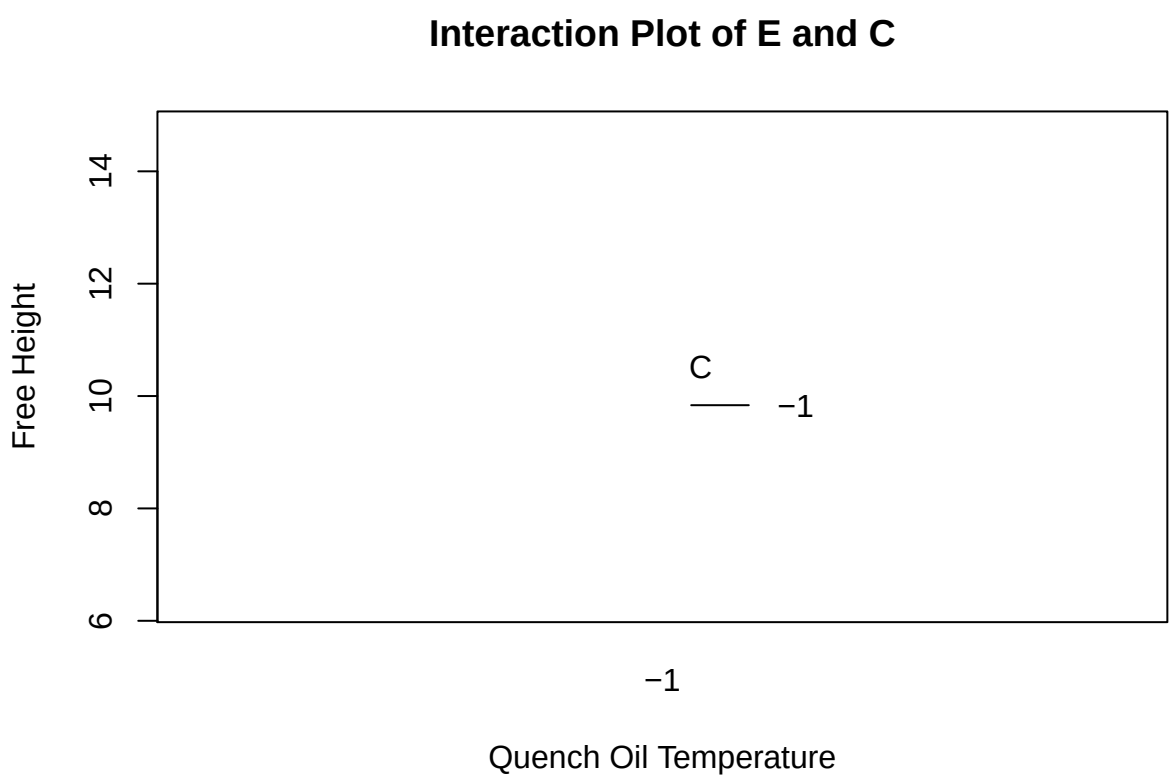
```r
coded=function(x)
{
  ifelse(x=="+", 1, -1)
}

# decoding data
for (j in 1:5)
  spring[, j]=as.numeric(coded(spring[, j]))


with(spring, interaction.plot(E, A, as.numeric(FH), xlab = "Quench Oil Temperature",
                              ylab = "Free Height", main = "Interaction Plot of E and A"))
```



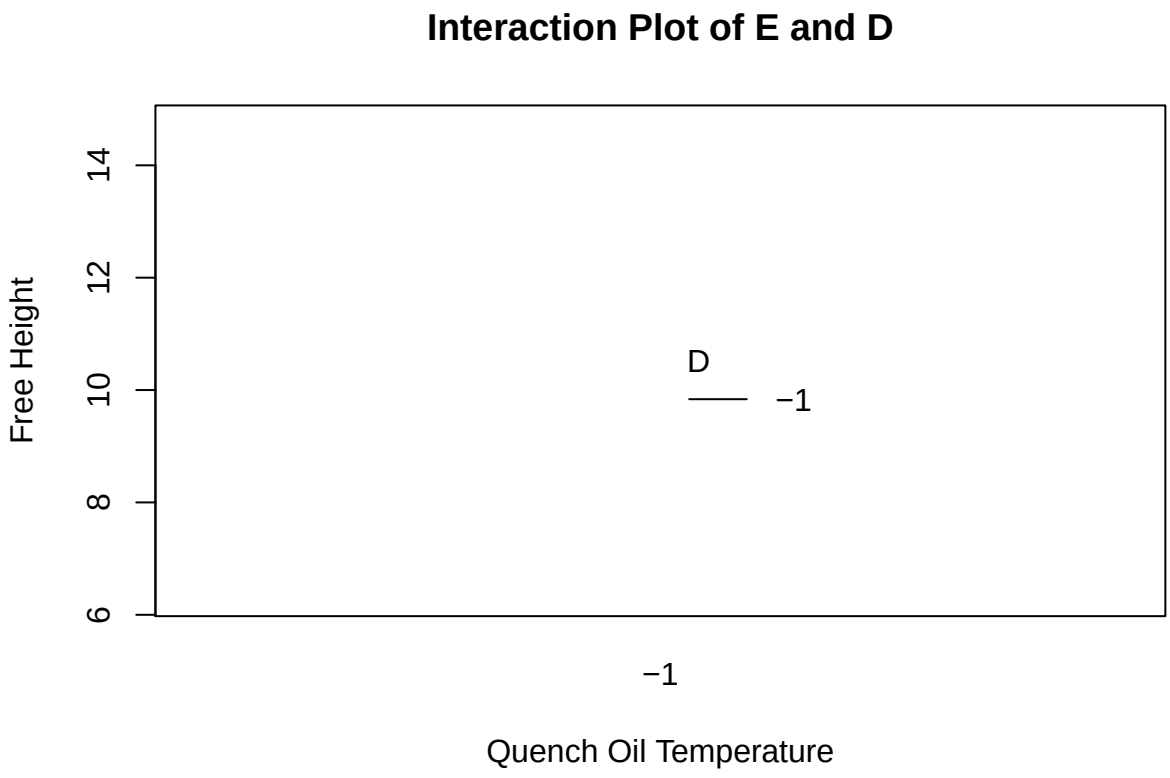

```r
with(spring, interaction.plot(E, B, as.numeric(FH), xlab = "Quench Oil Temperature",
                              ylab = "Free Height", main = "Interaction Plot of E and B"))
```




```
with(spring, interaction.plot(E, C, as.numeric(FH), xlab = "Quench Oil Temperature",
                              ylab = "Free Height", main = "Interaction Plot of E and C"))
```




```
with(spring, interaction.plot(E, D, as.numeric(FH), xlab = "Quench Oil Temperature",
                              ylab = "Free Height", main = "Interaction Plot of E and D"))
```




```
##mean
with(spring, tapply(as.numeric(FH),A,mean))
```

```
##       -1
## 10.52083
```

```
with(spring, tapply(as.numeric(FH),B,mean))
```

```
##       -1
## 10.52083
```

```
with(spring, tapply(as.numeric(FH),C,mean))
```

```
##       -1
## 10.52083
```

```
with(spring, tapply(as.numeric(FH),D,mean))
```

```
##       -1
## 10.52083
```

We use interaction plots to see what is the recommended setting for each factor. We take a look at our results and see that to reduce variability of Free Height we must have factor A high, factor B low, factor C low, and factor D high.

8.14

Consider the 2^5 design in Problem 6.24. Suppose that only a one-half fraction could be run. Furthermore, two days were required to take the 16 observations, and it was necessary to confound the 2^{5-1} design in two blocks. Construct the design and analyze the data.

```
yield = c(7,9,34,55,16,20,40,60,8,10,32,50,18,21,44,61,8,12,35,52,15,22,45,65,6,10,30,53,15,20,41,63)
A <- rep(x = c("-", "+"), times = 16)
B <- rep(x = c("-", "+"), each = 2, times = 8)
C <- rep(x = c("-", "+"), each = 4, times = 4)
D <- rep(x = c("-", "+"), each = 8, times = 2)
E <- rep(x = c("-", "+"), each = 16)

experimento = data.frame(A,B,C,D,E,yield)

coded=function(x) #a function to code variable x
{
  ifelse(x=="+", 1, -1)
}
for (j in 1:5)
  experimento[, j]=as.numeric(coded(experimento[, j]))

fraction.experi=with(experimento, experimento[A * B * C * D * E== 1,])

#linear model
experi.lm = lm(yield ~ A*B*C*D*E, fraction.experi); summary(experi.lm)
```

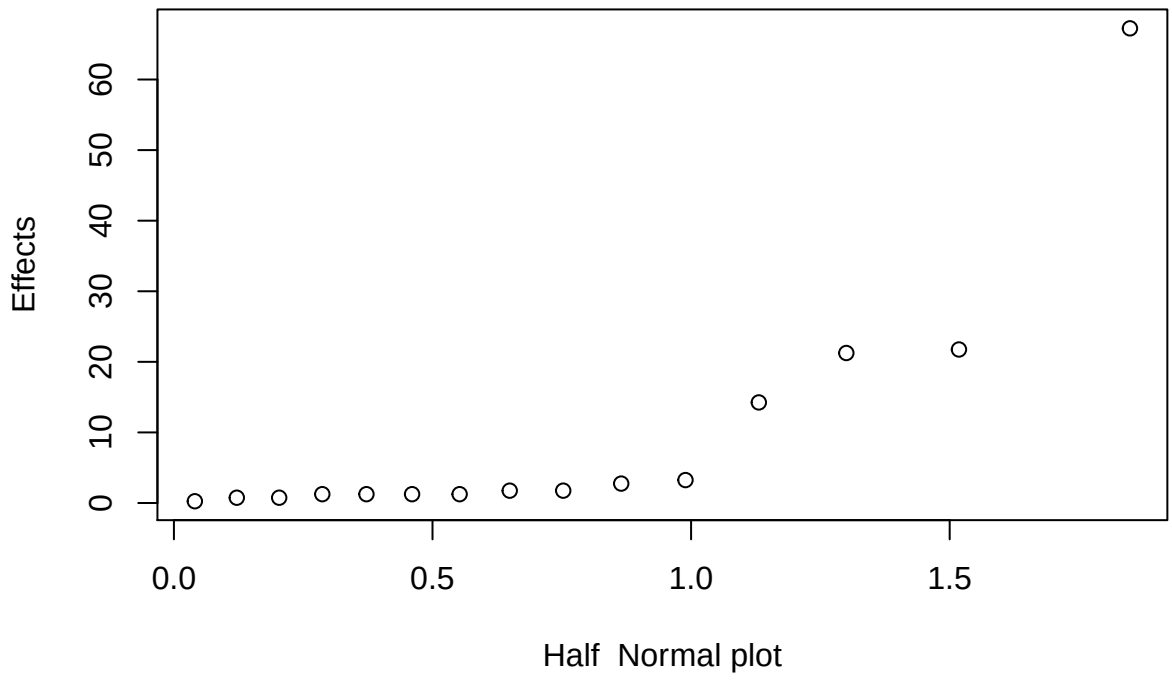
```
##
## Call:
## lm(formula = yield ~ A * B * C * D * E, data = fraction.experi)
##
## Residuals:
## ALL 16 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (16 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.4375         NA      NA       NA
## A             5.4375         NA      NA       NA
## B            16.8125         NA      NA       NA
## C             5.3125         NA      NA       NA
## D            -0.3125         NA      NA       NA
## E             0.1875         NA      NA       NA
## A:B           3.5625         NA      NA       NA
## A:C           0.3125         NA      NA       NA
## B:C           0.4375         NA      NA       NA
## A:D           0.4375         NA      NA       NA
## B:D          -0.1875         NA      NA       NA
## C:D           0.3125         NA      NA       NA
## A:E           0.6875         NA      NA       NA
## B:E           0.0625         NA      NA       NA
## C:E           0.3125         NA      NA       NA
## D:E          -0.8125         NA      NA       NA
## A:B:C             NA         NA      NA       NA
## A:B:D             NA         NA      NA       NA
## A:C:D             NA         NA      NA       NA
```

```
## B:C:D             NA         NA      NA       NA
## A:B:E             NA         NA      NA       NA
## A:C:E             NA         NA      NA       NA
## B:C:E             NA         NA      NA       NA
## A:D:E             NA         NA      NA       NA
## B:D:E             NA         NA      NA       NA
## C:D:E             NA         NA      NA       NA
## A:B:C:D           NA         NA      NA       NA
## A:B:C:E           NA         NA      NA       NA
## A:B:D:E           NA         NA      NA       NA
## A:C:D:E           NA         NA      NA       NA
## B:C:D:E           NA         NA      NA       NA
## A:B:C:D:E         NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 15 and 0 DF,  p-value: NA
```

```
#alias
alias(experi.lm)
```

```
## Model :
## yield ~ A * B * C * D * E
##
## Complete :
##           (Intercept) A B C D E A:B A:C B:C A:D B:D C:D A:E B:E C:E D:E
## A:B:C     0           0 0 0 0 0 0   0   0   0   0   0   0   0   0   1
## A:B:D     0           0 0 0 0 0 0   0   0   0   0   0   0   0   1   0
## A:C:D     0           0 0 0 0 0 0   0   0   0   0   0   0   1   0   0
## B:C:D     0           0 0 0 0 0 0   0   0   0   0   0   1   0   0   0
## A:B:E     0           0 0 0 0 0 0   0   0   0   0   1   0   0   0   0
## A:C:E     0           0 0 0 0 0 0   0   0   0   1   0   0   0   0   0
## B:C:E     0           0 0 0 0 0 0   0   0   1   0   0   0   0   0   0
## A:D:E     0           0 0 0 0 0 0   0   1   0   0   0   0   0   0   0
## B:D:E     0           0 0 0 0 0 0   1   0   0   0   0   0   0   0   0
## C:D:E     0           0 0 0 0 0 1   0   0   0   0   0   0   0   0   0
## A:B:C:D   0           0 0 0 0 1 0   0   0   0   0   0   0   0   0   0
## A:B:C:E   0           0 0 0 1 0 0   0   0   0   0   0   0   0   0   0
## A:B:D:E   0           0 0 1 0 0 0   0   0   0   0   0   0   0   0   0
## A:C:D:E   0           0 1 0 0 0 0   0   0   0   0   0   0   0   0   0
## B:C:D:E   0           1 0 0 0 0 0   0   0   0   0   0   0   0   0   0
## A:B:C:D:E 1           0 0 0 0 0 0   0   0   0   0   0   0   0   0   0
```

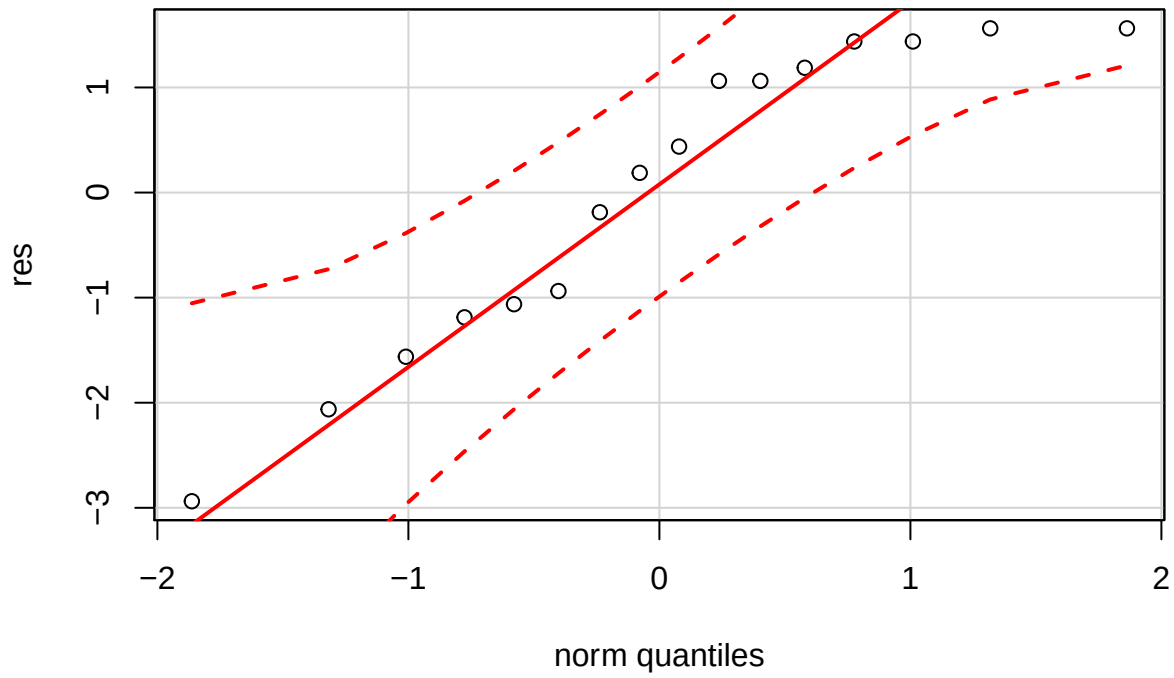
```
#normal probabiliy plot
qqnorm(aov(yield ~ A * B * C * D * E, fraction.experi), label = TRUE)# A,B,C, and AB
```



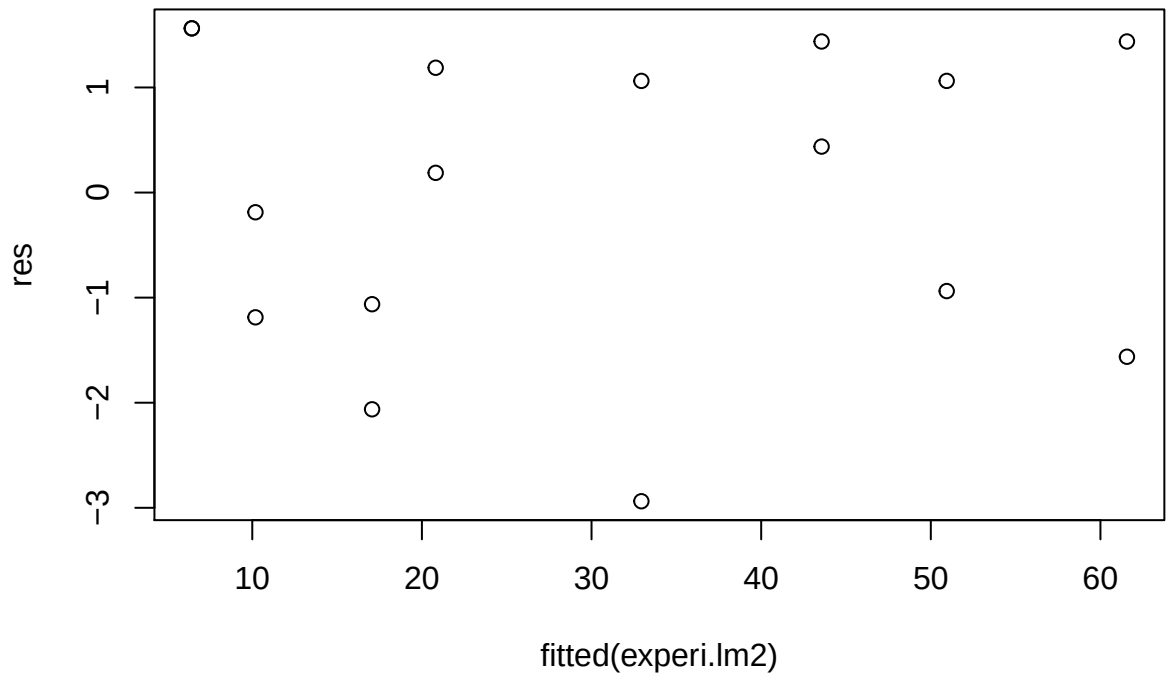

```
#new linear model
experi.lm2 = lm(yield ~ A + B + C + A*B, fraction.experi); summary(experi.lm2)
```

```
##
## Call:
## lm(formula = yield ~ A + B + C + A * B, data = fraction.experi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9375 -1.0938  0.3125  1.2500  1.5625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.4375     0.4243  71.733 4.80e-16 ***
## A             5.4375     0.4243  12.815 5.90e-08 ***
## B            16.8125     0.4243  39.623 3.21e-13 ***
## C             5.3125     0.4243  12.520 7.51e-08 ***
## A:B           3.5625     0.4243   8.396 4.11e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.697 on 11 degrees of freedom
## Multiple R-squared:  0.9944, Adjusted R-squared:  0.9924
## F-statistic: 490.4 on 4 and 11 DF,  p-value: 2.606e-12
```

```
res = fraction.experi$yield - fitted(experi.lm2)
qqPlot(res)
```




```
plot(fitted(experi.lm2), res)
```



We begin with our analysis by checking the null model and checking if the main effects are aliased with their interaction effects. We see the aliased chart has good results. We use half normal probability and see that the largest effects are A,B,C, and AB. We fit that into a new refined model, check our p-values, normality and p-values and our results look appropriate. We can state that our refined model is good.

8.51

A 16-run fractional factorial experiment in nine factors was conducted by Chrysler Motors Engineering and described in the article "Sheet Molded Compound Process Improvement," by P. I. Hsieh and D. E. Goodwin (Fourth Symposium on Taguchi Methods, American Supplier Institute, Dearborn, MI, 1986, pp. 13-21). The purpose was to reduce the number of defects in the finish of sheet-molded grill opening panels. The design, and the resulting number of defects, c, observed on each run, is shown in Table P8.14. This is a resolution III fraction with generators E = BD, F = BCD, G = AC, H = ACD, and J = AB.

(a) Find the defining relation and the alias relationships in this design.

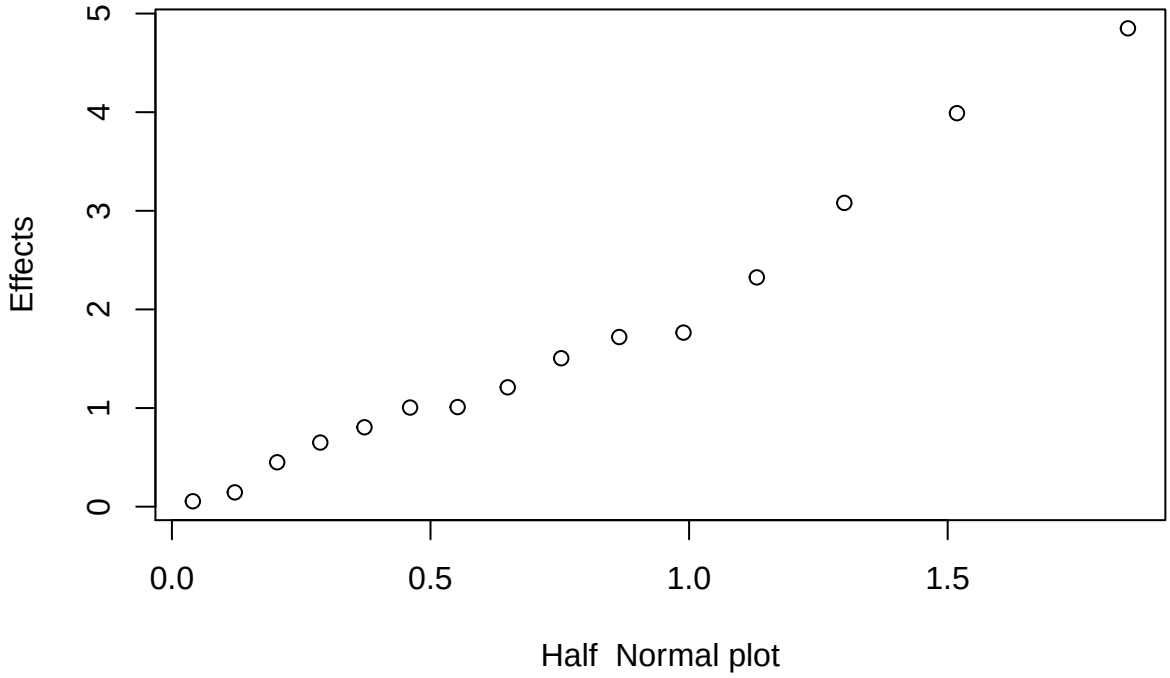
The generators we gather from the table are shown below: I = BDE = BCDF = ACG = ACDH = ABJ

From there, we can derive the following aliases: CEF = ABCDEG = ABCEH = ADEJ = ABDFG = ABFG = ACDFJ = DGH = BCGJ = BCDHJ

(b) Estimate the factor effects and use a normal probability plot to tentatively identify the important factors.

```
A <- rep(x = c("-", "+"), times = 8)
B <- rep(x = c("-", "+"), each = 2, times = 4)
C <- rep(x = c("-", "+"), each = 4, times = 2)
```

```r
D <- rep(x = c("-", "+"), each = 8)
E <- c("+","+","-","-","+","+","-","-","-","-","+","+","-","-","+","+")
F <- c("-","-","+","+","+","+","-","-","+","+","-","-","-","-","+","+")
G <- c("+","-","+","-","-","+","-","+","+","-","+","-","-","+","-","+")
H <- c("-","+","-","+","+","-","+","-","+","-","+","-","-","+","-","+")
J <- c("+","-","-","+","+","-","-","+","+","-","-","+","+","-","-","+")
FTMOD <- c(7.52,4.18,1.57,2.12,1.87,2.12,7.12,1.57,1.21,0.50,1.87,3.54,1.87,2.12,0.50,0.50)
#data
hardata = data.frame(A,B,C,D,E,F,G,H,J,FTMOD)

coded=function(x) #a function to code variable x
{
  ifelse(x=="+", 1, -1)
}
for (j in 1:9)
  hardata[, j]=as.numeric(coded(hardata[, j]))

fraction.hardata=with(hardata, hardata[A * B * C * D * E * F * G * H * J== 1,])

#linear regression
hardata.lm <- lm(FTMOD ~ A*B*C*D*E*F*G*H*J, fraction.hardata)

#alias
#alias(hardata.lm)


qqnorm(aov(FTMOD ~ A*B*C*D*E*F*G*H*J, fraction.hardata), label = TRUE)#F and D
```



(c) Fit an appropriate model using the factors identified in part (b) above.

```
#refined model
#hardata.lm2 <- lm(FTMOD ~ F, fraction.hardata); summary(hardata.lm2)
```

(d) Plot the residuals from this model versus the predicted number of defects. Also, prepare a normal probability plot of the residuals. Comment on the adequacy of these plots.

```
#residual analysis
#res <- fraction.hardata$FTMOD - fitted(hardata.lm2)
#qqPlot(res)
#plot(fitted(hardata.lm2, res))
```

(e) In part (d) you should have noticed an indication that the variance of the response is not constant. (Considering that the response is a count, you should have expected this.) The previous table also shows a transformation on c, the square root, that is a widely used variance stabilizing transformation for count data. (Refer to the discussion of variance stabilizing transformations in Chapter 3.) Repeat parts (a) through (d) using the transformed response and comment on your results. Specifically, are the residual plots improved?

(f) There is a modification to the square root transformation, proposed by Freeman and Tukey ("Transformations Related to the Angular and the Square Root," Annals of Mathematical Statistics, Vol. 21, 1950, pp. 607-611) that improves its performance. FandT's modification to the square root transformation $\frac{\sqrt{c}+\sqrt{c+1}}{2}$ is Rework parts (a) through (d) using this transformation and comment on the results. (For an interesting discussion and analysis of this experiment, refer to "Analysis of Factorial Experiments with Defects or Defectives as the Response," by S. Bisgaard and H. T. Fuller, Quality Engineering, Vol. 7, 1994-95, pp. 429-443.)