101B - HW6

```
suppressPackageStartupMessages(library(FrF2))
```

```
## Warning: package 'FrF2' was built under R version 4.1.2
```

```
## Warning: package 'DoE.base' was built under R version 4.1.2
```

```
set.seed(123)

A <- rep(c(-1,1),8)
B <- rep(c(-1,-1,1,1),4)
C <- rep(c(rep(-1,4),rep(1,4)),2)
D <- c(-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1)
E <- c(rep(-1,8),rep(1,8))

height <- c(7.78,8.15,7.50,7.59,7.54,7.69,7.56,7.56,7.50,7.88,7.50,7.63,7.32,7.56,7.18,7.81,
            7.78,8.18,7.56,7.56,8.00,8.09,7.52,7.81,7.25,7.88,7.56,7.75,7.44,7.69,7.18,7.50,
            7.81,7.88,7.50,7.75,7.88,8.06,7.44,7.69,7.12,7.44,7.50,7.56,7.44,7.62,7.25,7.59)


#height1 <- c(7.78,8.15,7.50,7.59,7.54,7.69,7.56,7.56,7.50,7.88,7.50,7.63,7.32,7.56,7.18,7.81)
#height2 <- c(7.78,8.18,7.56,7.56,8.00,8.09,7.52,7.81,7.25,7.88,7.56,7.75,7.44,7.69,7.18,7.50)
#height3 <- c(7.81,7.88,7.50,7.75,7.88,8.06,7.44,7.69,7.12,7.44,7.50,7.56,7.44,7.62,7.25,7.59)

#height <- list(height1, height2,height3)
#height <- rowMeans(as.data.frame(height))

data <- data.frame("A" = A, "B" = B, "C" = C, "D" = D, "E" = E, "Y" = height)

model <- lm(Y~A*B*C*D*E, data)
model
```

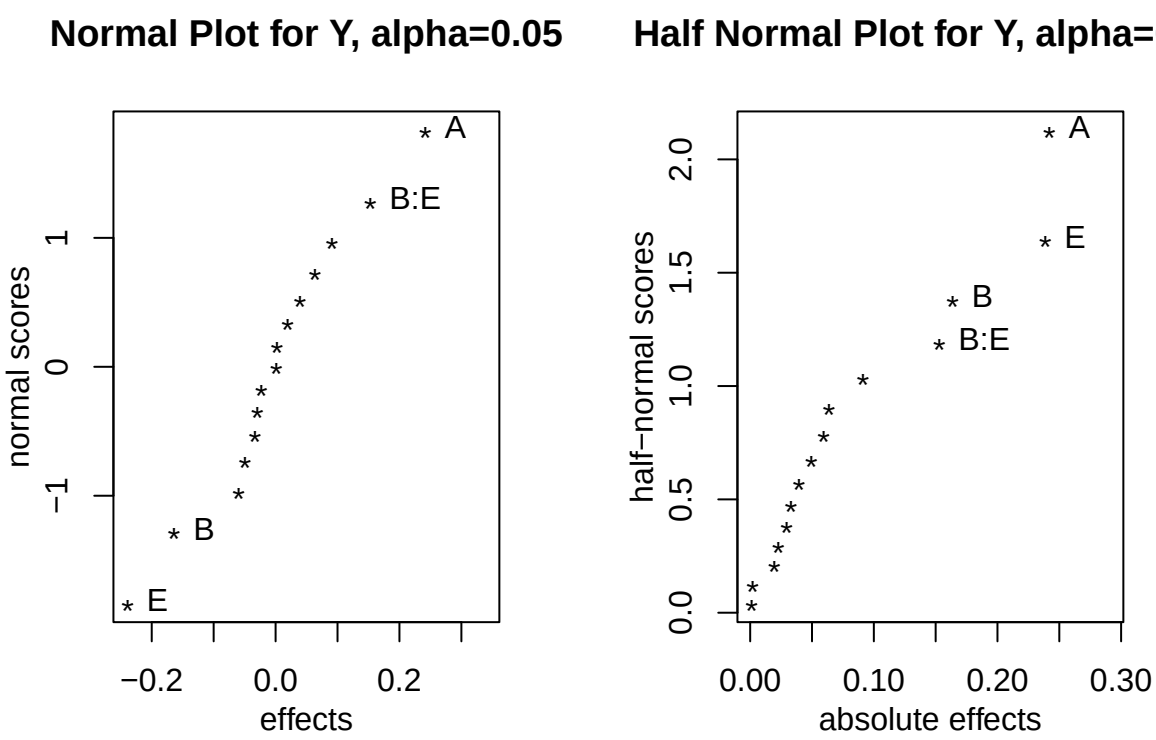
```
##
## Call:
## lm.default(formula = Y ~ A * B * C * D * E, data = data)
##
## Coefficients:
## (Intercept)            A            B            C            D            E
##    7.625625     0.121042    -0.081875    -0.024792     0.045625    -0.119375
##         A:B          A:C          B:C          A:D          B:D          C:D
##   -0.014792     0.000625    -0.011458           NA           NA           NA
##         A:E          B:E          C:E          D:E        A:B:C        A:B:D
##    0.031875     0.076458    -0.016458     0.019792           NA           NA
##       A:C:D        B:C:D        A:B:E        A:C:E        B:C:E        A:D:E
##          NA           NA     0.001042     0.009792    -0.029792           NA
```

```
##       B:D:E       C:D:E     A:B:C:D     A:B:C:E     A:B:D:E     A:C:D:E
##          NA          NA          NA          NA          NA          NA
##     B:C:D:E   A:B:C:D:E
##          NA          NA
```

```
FrF2::aliases(model)
```

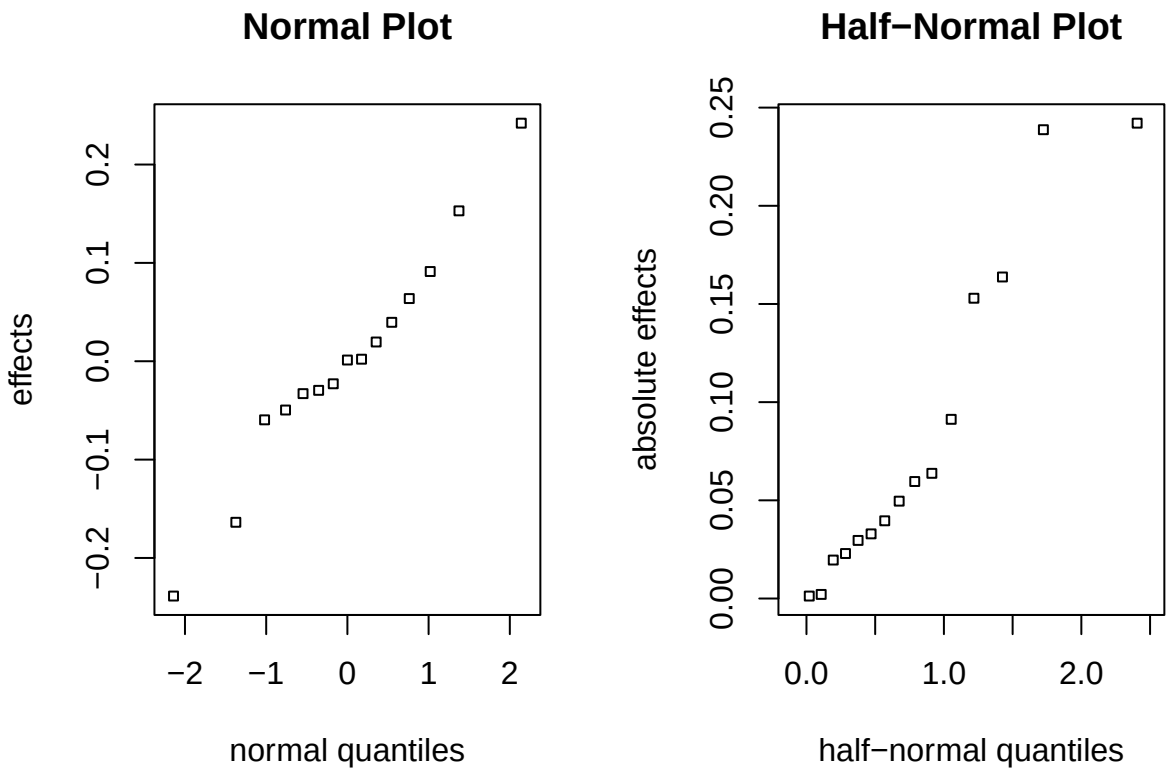
```
##
##  A = B:C:D
##  B = A:C:D
##  C = A:B:D
##  D = A:B:C
##  E = A:B:C:D:E
##  A:B = C:D
##  A:C = B:D
##  B:C = A:D
##  A:E = B:C:D:E
##  B:E = A:C:D:E
##  C:E = A:B:D:E
##  D:E = A:B:C:E
##  A:B:E = C:D:E
##  A:C:E = B:D:E
##  B:C:E = A:D:E
```

```
par(mfrow = c(1,2))
DanielPlot(model,half = FALSE)
DanielPlot(model, half= TRUE)
```




```
normalplot <- function(y, label=F, n=length(y), fac.names=NULL, xlim=c(-2.2, 2.2), main="Normal Plot",
{ # label the most singificant n effects
m <- length(y)
x <- seq(0.5/m, 1.0-0.5/m, by=1/m)
x <- qnorm(x)
y <- sort(y)
qqplot(x, y,
xlab="normal quantiles", ylab="effects",
xlim=xlim, main=main, ...)
if(is.null(fac.names)) fac.names <- names(y)
else fac.names <- rev( c(fac.names, rep("", length(y)-length(fac.names)) ) )
ord=order(abs(y))
if(label) for(i in ord[(m-n+1):m]) text(x[i]+.35,y[i], fac.names[i],cex=0.9)
}


halfnormalplot <- function(y, label=F, n=length(y), fac.names=NULL, xlim=c(-.1, 2.5), main="Half-Normal
{ # label the most singificant n effects
m <- length(y)
x <- seq(0.5+0.25/m, 1.0-0.25/m, by=0.5/m)
x <- qnorm(x)
y <- sort(abs(y))
qqplot(x, y,
xlab="half-normal quantiles", ylab="absolute effects",
xlim=xlim, main=main, ...)
if(is.null(fac.names)) fac.names <- names(y)
```

```
else fac.names <- rev( c(fac.names, rep("", length(y)-length(fac.names)) ) )
if(label) for(i in (m-n+1):m) text(x[i]+.2,y[i], fac.names[i],cex=0.9)
}

effects <- 2*coef(model)[-1]
par(mfrow = c(1,2))

normalplot(effects, label = T, type = "p", pch = 22, cex = 0.7)
halfnormalplot(effects, label = T, type = "p", pch = 22, cex = 0.7)
```




```
data$Y
```

```
##  [1] 7.78 8.15 7.50 7.59 7.54 7.69 7.56 7.56 7.50 7.88 7.50 7.63 7.32 7.56 7.18
## [16] 7.81 7.78 8.18 7.56 7.56 8.00 8.09 7.52 7.81 7.25 7.88 7.56 7.75 7.44 7.69
## [31] 7.18 7.50 7.81 7.88 7.50 7.75 7.88 8.06 7.44 7.69 7.12 7.44 7.50 7.56 7.44
## [46] 7.62 7.25 7.59
```

```
l <- data$Y
sd(l)
```

```
## [1] 0.248004
```

```
#No, this is not the best possible design. This is because there are 16 runs for 5 factors and by setti
```

```
si <- read.csv("Problem8.37 dataset.csv")
si
```

```
##    Order  A  B  C  D  E  F  G  H  PVM NPU
## 1      4 -1 -1 -1 -1 -1 -1 -1  1 1.00   5
## 2     13  1 -1 -1 -1 -1  1  1  1 1.04  13
## 3      6 -1  1 -1 -1 -1  1  1 -1 1.02  16
## 4      3  1  1 -1 -1 -1 -1 -1 -1 0.99  12
## 5     19 -1 -1  1 -1 -1  1 -1 -1 1.02  15
## 6     25  1 -1  1 -1 -1 -1  1 -1 1.01   9
## 7     21 -1  1  1 -1 -1 -1  1  1 1.01  12
## 8     14  1  1  1 -1 -1  1 -1  1 1.03  17
## 9     10 -1 -1 -1  1 -1 -1  1 -1 1.04  21
## 10    22  1 -1 -1  1 -1  1 -1 -1 1.14  20
## 11     1 -1  1 -1  1 -1  1 -1  1 1.20  25
## 12     2  1  1 -1  1 -1 -1  1  1 1.13  21
## 13    30 -1 -1  1  1 -1  1  1  1 1.14  25
## 14     8  1 -1  1  1 -1 -1 -1  1 1.07  13
## 15     9 -1  1  1  1 -1 -1 -1 -1 1.06  20
## 16    20  1  1  1  1 -1  1  1 -1 1.13  26
## 17    17 -1 -1 -1 -1  1 -1 -1 -1 1.02  10
## 18    18  1 -1 -1 -1  1  1  1 -1 1.10  13
## 19     5 -1  1 -1 -1  1  1  1  1 1.09  17
## 20    26  1  1 -1 -1  1 -1 -1  1 0.96  13
## 21    31 -1 -1  1 -1  1  1 -1  1 1.02  14
## 22    11  1 -1  1 -1  1 -1  1  1 1.07  11
## 23    29 -1  1  1 -1  1 -1  1 -1 0.98  10
## 24    23  1  1  1 -1  1  1 -1 -1 0.95  14
## 25    32 -1 -1 -1  1  1 -1  1  1 1.10  28
## 26     7  1 -1 -1  1  1  1 -1  1 1.12  24
## 27    15 -1  1 -1  1  1  1 -1 -1 1.19  22
## 28    27  1  1 -1  1  1 -1  1 -1 1.13  15
## 29    12 -1 -1  1  1  1  1  1 -1 1.20  21
## 30    28  1 -1  1  1  1 -1 -1 -1 1.07  19
## 31    24 -1  1  1  1  1 -1 -1  1 1.12  21
## 32    16  1  1  1  1  1  1  1  1 1.21  27
```

```
set.seed(123)
my.design <- FrF2(nruns = 32, nfactors = 8, randomize = FALSE)

des_inf <- design.info(my.design)
des_inf$aliased
```

```
## $legend
## [1] "A=A" "B=B" "C=C" "D=D" "E=E" "F=F" "G=G" "H=H"
##
## $main
## character(0)
##
## $fi2
## [1] "AB=CF=DG" "AC=BF"    "AD=BG"    "AF=BC"    "AG=BD"    "CD=FG"    "CG=DF"
```

```
design.info(my.design)$aliased$fi2
```

```
## [1] "AB=CF=DG" "AC=BF"    "AD=BG"    "AF=BC"    "AG=BD"    "CD=FG"    "CG=DF"
```

```
model_P <- lm(PVM ~ A+B+C+D+F+E+G+H+A:B+A:C+A:D+A:F+A:G+C:D+C:G, si)

summary(model_P)
```

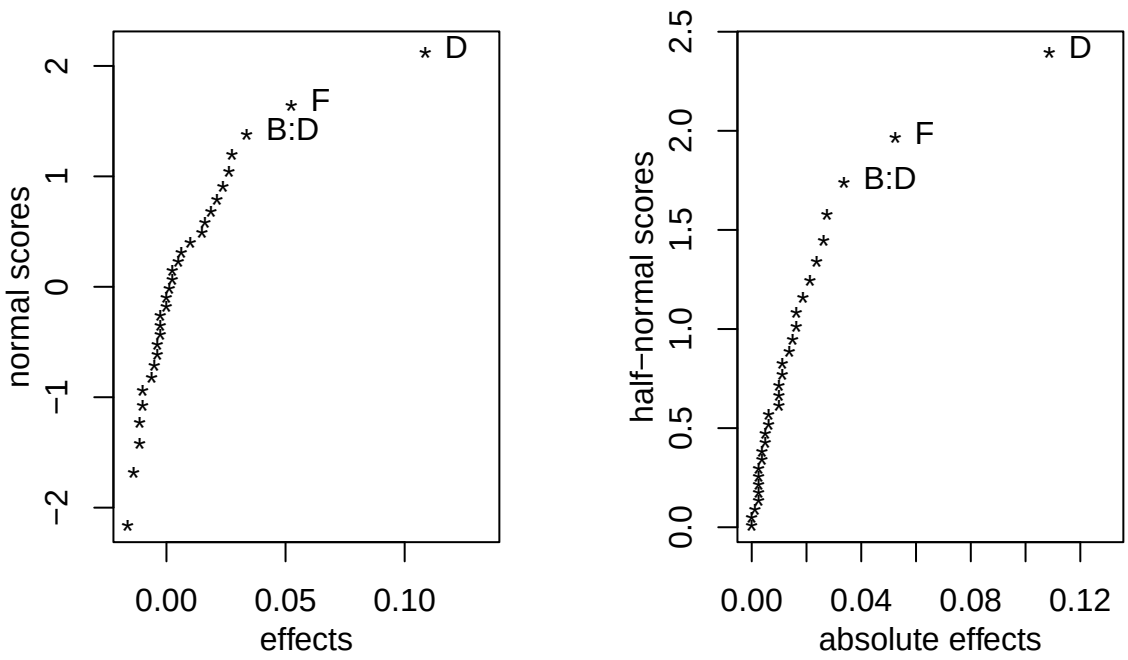
```
##
## Call:
## lm.default(formula = PVM ~ A + B + C + D + F + E + G + H + A:B +
##     A:C + A:D + A:F + A:G + C:D + C:G, data = si)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04063 -0.01969  0.00375  0.01469  0.04938
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.073750   0.005482 195.864  < 2e-16 ***
## A           -0.001875   0.005482  -0.342 0.736787
## B            0.001250   0.005482   0.228 0.822524
## C           -0.005625   0.005482  -1.026 0.320126
## D            0.054375   0.005482   9.919 3.08e-08 ***
## F            0.026250   0.005482   4.788 0.000201 ***
## E            0.009375   0.005482   1.710 0.106564
## G            0.013750   0.005482   2.508 0.023289 *
## H            0.008125   0.005482   1.482 0.157743
## A:B         -0.006875   0.005482  -1.254 0.227828
## A:C          0.001250   0.005482   0.228 0.822524
## A:D         -0.001250   0.005482  -0.228 0.822524
## A:F         -0.008125   0.005482  -1.482 0.157743
## A:G          0.016875   0.005482   3.078 0.007201 **
## C:D          0.002500   0.005482   0.456 0.654499
## C:G          0.011875   0.005482   2.166 0.045748 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03101 on 16 degrees of freedom
## Multiple R-squared:  0.9049, Adjusted R-squared:  0.8157
## F-statistic: 10.15 on 15 and 16 DF,  p-value: 1.68e-05
```

```
model_N <- lm(NPU ~ A+B+C+D+F+E+G+H+A:B+A:C+A:D+A:F+A:G+C:D+C:G, si)

summary(model_N)
```

```
##
## Call:
## lm.default(formula = NPU ~ A + B + C + D + F + E + G + H + A:B +
##     A:C + A:D + A:F + A:G + C:D + C:G, data = si)
##
```

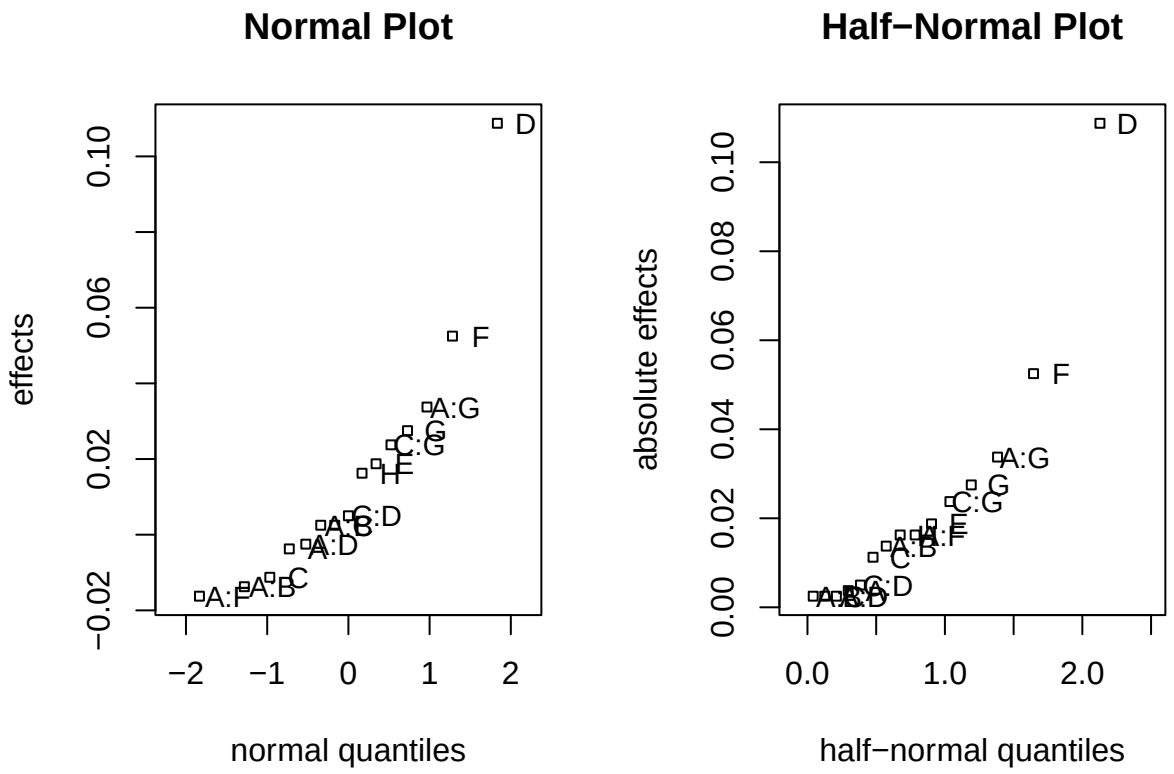
```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6875 -0.8437  0.2187  1.2031  4.2500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.15625    0.52663  32.577 4.68e-16 ***
## A           -0.46875    0.52663  -0.890 0.386608
## B            0.84375    0.52663   1.602 0.128677
## C           -0.03125    0.52663  -0.059 0.953417
## D            4.59375    0.52663   8.723 1.77e-07 ***
## F            2.15625    0.52663   4.094 0.000846 ***
## E            0.28125    0.52663   0.534 0.600648
## G            0.65625    0.52663   1.246 0.230658
## H            0.71875    0.52663   1.365 0.191203
## A:B          0.59375    0.52663   1.127 0.276180
## A:C          0.34375    0.52663   0.653 0.523199
## A:D         -0.65625    0.52663  -1.246 0.230658
## A:F          0.40625    0.52663   0.771 0.451708
## A:G         -0.46875    0.52663  -0.890 0.386608
## C:D         -0.21875    0.52663  -0.415 0.683385
## C:G         -0.15625    0.52663  -0.297 0.770516
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.979 on 16 degrees of freedom
## Multiple R-squared:  0.8676, Adjusted R-squared:  0.7434
## F-statistic: 6.988 on 15 and 16 DF,  p-value: 0.0001868
```

```
m <- lm(PVM ~ A*B*C*D*E*F*G*H, si)
par(mfrow=c(1,2))
DanielPlot(m, half = F)
DanielPlot(m, half = T)
```

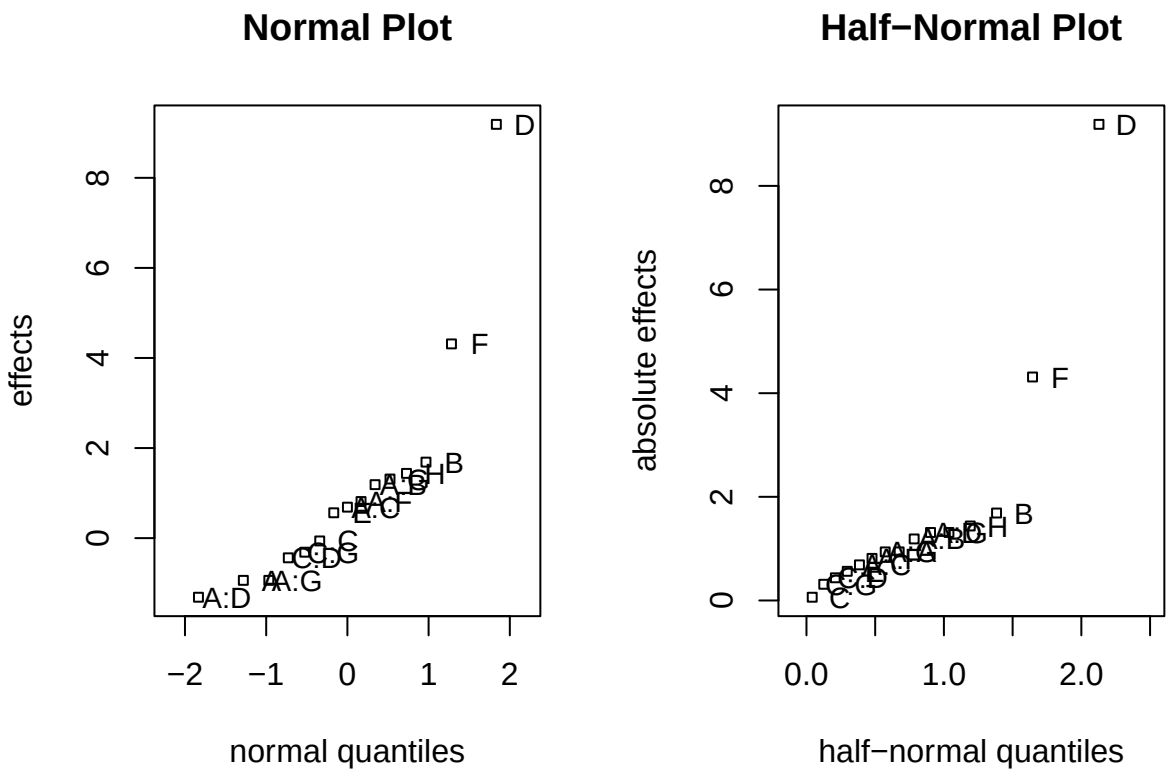



```
normalplot <- function(y, label=F, n=length(y), fac.names=NULL, xlim=c(-2.2, 2.2), main="Normal Plot",
{ # label the most singificant n effects
m <- length(y)
x <- seq(0.5/m, 1.0-0.5/m, by=1/m)
x <- qnorm(x)
y <- sort(y)
qqplot(x, y,
xlab="normal quantiles", ylab="effects",
xlim=xlim, main=main, ...)
if(is.null(fac.names)) fac.names <- names(y)
else fac.names <- rev( c(fac.names, rep("", length(y)-length(fac.names)) ) )
ord=order(abs(y))
if(label) for(i in ord[(m-n+1):m]) text(x[i]+.35,y[i], fac.names[i],cex=0.9)
}


halfnormalplot <- function(y, label=F, n=length(y), fac.names=NULL, xlim=c(-.1, 2.5), main="Half-Normal
{ # label the most singificant n effects
m <- length(y)
x <- seq(0.5+0.25/m, 1.0-0.25/m, by=0.5/m)
x <- qnorm(x)
y <- sort(abs(y))
qqplot(x, y,
xlab="half-normal quantiles", ylab="absolute effects",
xlim=xlim, main=main, ...)
if(is.null(fac.names)) fac.names <- names(y)
```

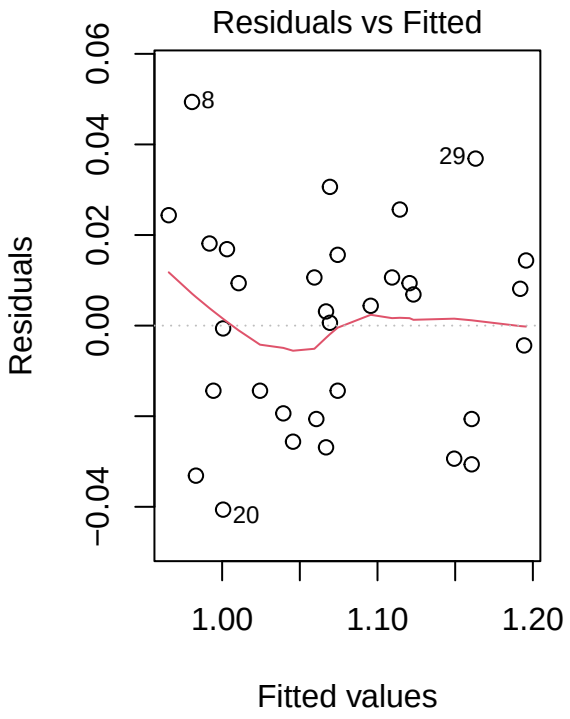
```
else fac.names <- rev( c(fac.names, rep("", length(y)-length(fac.names)) ) )
if(label) for(i in (m-n+1):m) text(x[i]+.2,y[i], fac.names[i],cex=0.9)
}

par(mfrow=c(1,2))
effects = 2*coef(model_P)[-1]
normalplot(effects,label=T,type="p",pch=22,cex=0.7)
halfnormalplot(effects,label=T,type="p",pch=22,cex=0.7)
```

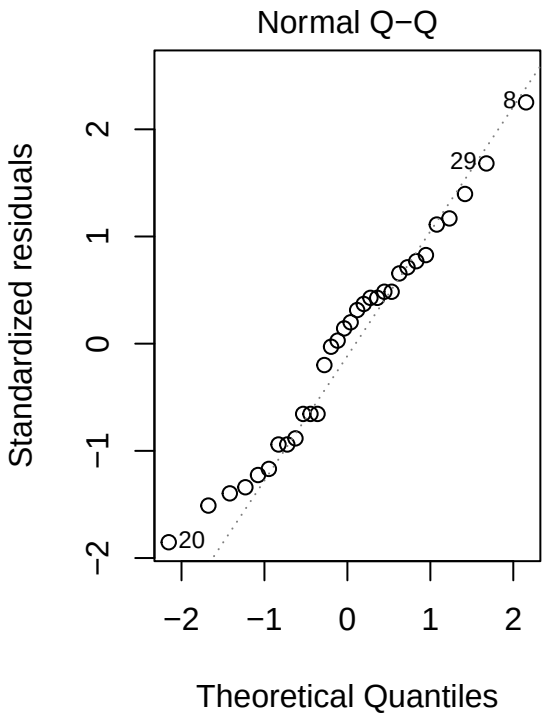



```
par(mfrow=c(1,2))
effects = 2*coef(model_N)[-1]
normalplot(effects,label=T,type="p",pch=22,cex=0.7)
halfnormalplot(effects,label=T,type="p",pch=22,cex=0.7)
```

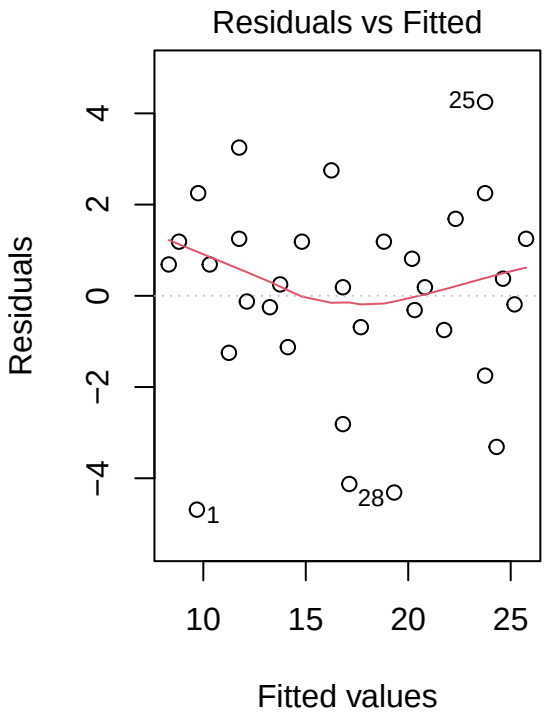


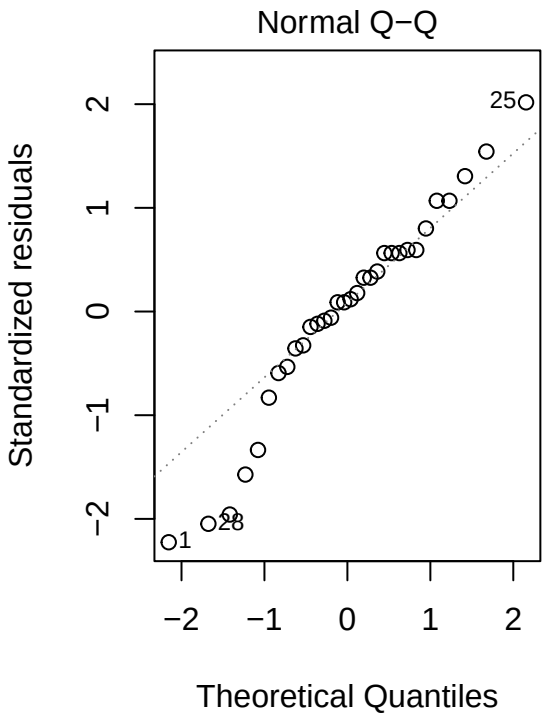

```
#Comparing both models NPU and VPM, we can conclude D and F are significant.


par(mfrow=c(1,2))
plot(model_P,1:2)
```






```
par(mfrow=c(1,2))
plot(model_N,1:2)
```






```
#By looking at the two models, we can conclude that the constant variation assumption is satisfied, an
```