Ch-07_08 R Codes

Ping-Yang Chen

2024-05-26

Textbook: Montgomery, D. C. (2012). *Design and analysis of experiments*, 8th Edition. John Wiley & Sons.

Online handouts: https://github.com/PingYangChen/ANOVA_Course_R_Code

7.21

By the defining contrast, to confound eight blocks with $ABCD$, ACE and $ABEF$, let

$$\begin{aligned} L_1 &= x_1 + x_2 + x_3 + x_4 \\ L_2 &= x_1 + x_3 + x_5 \\ L_3 &= x_1 + x_2 + x_5 + x_6 \end{aligned}$$

```r
designMat <- data.frame(
    A = rep(0:1, 32),
    B = rep(rep(0:1, each = 2), 16),
    C = rep(rep(0:1, each = 4), 8),
    D = rep(rep(0:1, each = 8), 4),
    E = rep(rep(0:1, each = 16), 2),
    F = rep(0:1, each = 32)
)
#print(head(designMat, 6))

letterMat <- sapply(1:ncol(designMat), function(j) {
  ifelse(designMat[,j] == 1, letters[j], "")
})
#print(head(letterMat, 6))

effectNames <- sapply(1:nrow(letterMat), function(i) {
  ifelse(all(letterMat[i,] == ""), "(1) ", paste0(letterMat[i,], collapse = ""))
})
rownames(designMat) <- effectNames
print(head(designMat, 6))
```

```
##      A B C D E F
## (1)  0 0 0 0 0 0
## a    1 0 0 0 0 0
## b    0 1 0 0 0 0
## ab   1 1 0 0 0 0
## c    0 0 1 0 0 0
## ac   1 0 1 0 0 0
```

Compute the linear combinations $L_1(i)$, $L_2(i)$ and $L_3(i)$, and take (mod 2) for each of them, $i = 1, 2, ..., 64$.

```r
attach(designMat)
assignBlock <- data.frame(
    L1 = (A + B + C + D) %% 2,
    L2 = (A + C + E) %% 2,
    L3 = (A + B + E + F) %% 2
)
detach(designMat)
```

Get the block IDs for each run.

```r
blockId <- as.matrix(assignBlock) %*% c(2^2, 2, 1) + 1
```

Present the runs in each block.

```r
result <- matrix("", 8, 8)
for (i in 1:8) {
  result[,i] <- effectNames[which(blockId == i)]
}
colnames(result) <- sprintf("Block %d", 1:8)
print(data.frame(result))
```

```
##   Block.1 Block.2 Block.3 Block.4 Block.5 Block.6 Block.7 Block.8
## 1    (1)       ac      ab      bc     abc       b       c       a
## 2    abcd      bd      cd      ad       d     acd     abd     bcd
## 3     bce     abe     ace       e      ae      ce      be    abce
## 4     ade     cde     bde   abcde    bcde    abde    acde      de
## 5     acf       f     bcf     abf      bf    abcf      af      cf
## 6     bdf   abcdf     adf     cdf    acdf      df    bcdf    abdf
## 7    abef    bcef      ef    acef     cef     aef   abcef     bef
## 8    cdef    adef  abcdef    bdef   abdef   bcdef     def   acdef
```

The other effects confounded with blocks:

$$
\begin{gathered}
(ABCD)(ACE) = A^2BC^2DE = BDE \\
(ABCD)(ABEF) = A^2B^2CDEF = CDEF \\
(ACE)(ABEF) = A^2BCE^2F = BCF \\
(ABCD)(BCF) = AB^2C^2DF = ADF
\end{gathered}
$$

8.11

2^{5-2} fractional factorial design with defining relation

$$I = ACE \text{ and } I = BDE$$

1. generate the 2^3 full factorial design

```
lvl <- c(-1, 1)
FF3 <- data.frame(
    A = rep(lvl, 4),
    B = rep(rep(lvl, each = 2), 2),
    C = rep(lvl, each = 4)
)
print(FF3)
```

```
##    A  B  C
## 1 -1 -1 -1
## 2  1 -1 -1
## 3 -1  1 -1
## 4  1  1 -1
## 5 -1 -1  1
## 6  1 -1  1
## 7 -1  1  1
## 8  1  1  1
```

2. add two columns D and E to form the 2^{5-2} by the defining relation

$$\begin{aligned} I = ACE &\implies E = AC \\ I = BDE &\implies D = BE \implies D = BAC \end{aligned}$$

```
attach(FF3)
augmentFrF <- data.frame(
    D = A * B * C,
    E = A * C
)
detach(FF3)

FrF5_2 <- cbind(FF3, augmentFrF)
# Get letters of each effect
letterMat <- sapply(1:ncol(FrF5_2), function(j) {
  ifelse(FrF5_2[,j] == 1, letters[j], "")
})
# Combine letters
effectNames <- sapply(1:nrow(letterMat), function(i) {
  ifelse(all(letterMat[i,] == ""), "(1) ", paste0(letterMat[i,], collapse = ""))
})
rownames(FrF5_2) <- effectNames
print(FrF5_2)
```

```
##        A  B  C  D  E
## e     -1 -1 -1 -1  1
## ad     1 -1 -1  1 -1
## bde   -1  1 -1  1  1
## ab     1  1 -1 -1 -1
## cd     -1 -1  1  1 -1
## ace     1 -1  1 -1  1
## bc     -1  1  1 -1 -1
## abcde   1  1  1  1  1
```

Complete defining relation is

$$I = ACE = BDE = ABCD$$

All aliases are

$$\begin{aligned} A &= CE = BCDAB = CD \\ B &= DE = ACDAD = BC \\ C &= AE = ABDAC = BD \\ D &= BE = ABC \\ E &= AC \end{aligned}$$

Add column of the response variable.

```
y <- numeric(8)
y[effectNames == "e"]     <- 23.2
y[effectNames == "ad"]    <- 16.9
y[effectNames == "cd"]    <- 23.8
y[effectNames == "bde"]   <- 16.8
y[effectNames == "ab"]    <- 15.5
y[effectNames == "bc"]    <- 16.2
y[effectNames == "ace"]   <- 23.4
y[effectNames == "abcde"] <- 18.1
frfData <- cbind(FrF5_2, y = y)
```

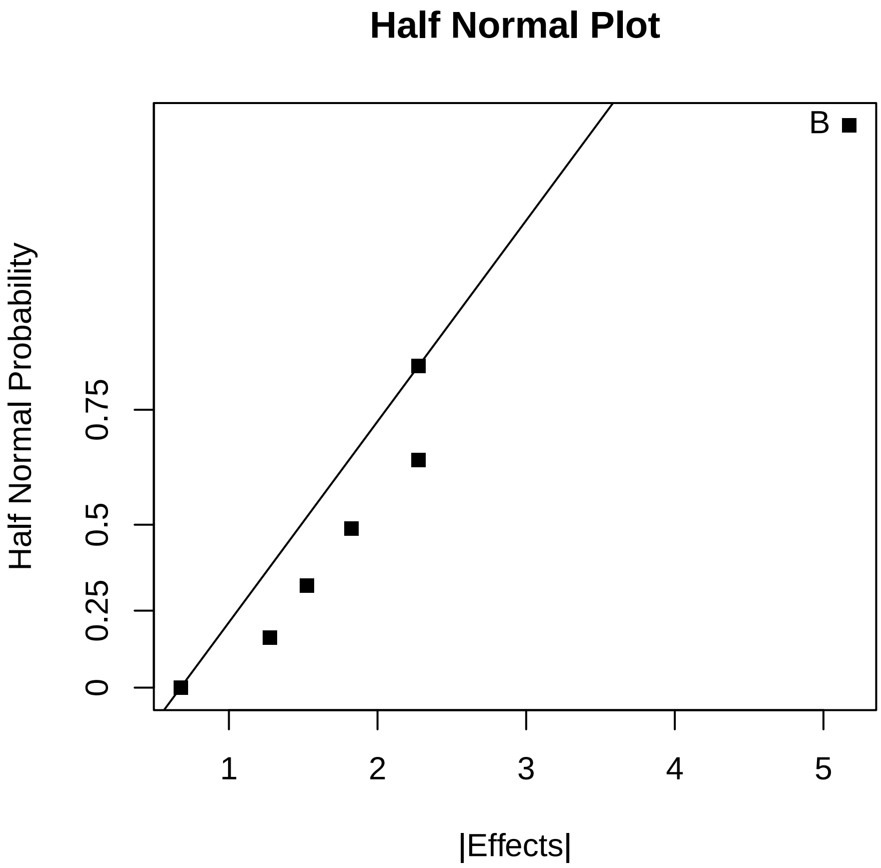
The estimation of factor effects are:

```
# Compute the model matrix of all effect terms without intercept
mmat5 <- model.matrix( ~ A+B+C+D+E+A:B+A:D - 1, data = frfData)
# Calculate the effect sizes using the +/- signs of the model matrix
eff5 <- numeric(ncol(mmat5))
for (i in 1:ncol(mmat5)) {
  eff5[i] <- 2*mean(frfData$y*mmat5[,i])
}
names(eff5) <- colnames(mmat5)
```

```
##   Factor Est.Effect
## 1      A     -1.525
## 2      B     -5.175
## 3      C      2.275
## 4      D     -0.675
## 5      E      2.275
## 6    A:B      1.825
## 7    A:D     -1.275
```

```r
# Half Normal Plot
halfqqnorm <- function(input, tol = 0.5) {
    y <- sort(abs(input))
    nq <- qnorm(seq(0.5, 0.99, length = length(y)))
    plot(y, nq, yaxt = "n", pch = 15,
         xlab = "|Effects|", ylab = "Half Normal Probability")
    title("Half Normal Plot")
    # choose anchor point to draw a straight line
    s <- min(which(diff(y)/diff(range(y)) > 1/(length(y)-1)))
    abline(a = -y[1]*(nq[s]-nq[1])/(y[s]-y[1]), b = (nq[s]-nq[1])/(y[s]-y[1]))
    axis(2, at = qnorm(seq(0.5, 0.9999, length = 5)),
         labels = round(seq(0, 1, length = 5),2))
    loc <- sqrt((nq - (y - y[1])*(nq[s]-nq[1])/(y[s]-y[1]))^2) > tol
    if (is.null(names(y))) {
        text(y[loc], nq[loc], order(abs(input))[loc], pos = 2)
    } else {
        text(y[loc], nq[loc], names(abs(input))[order(abs(input))[loc]], pos = 2)
    }
}
```

```r
halfqqnorm(eff5)
```



```r
fit2 <- aov(y ~ B, data = frfData)
summary(fit2)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## B            1  53.56   53.56   8.883 0.0246 *
## Residuals    6  36.18    6.03
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```