Ch-03 R Codes

Ping-Yang Chen

2024-03-15

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

Chapter 3

One-way ANOVA

Read the csv file `3_PlasmaEtching.csv` in R. Make sure that in the `data.frame` the variable `Power` is a factor. If not sure, apply `as.factor()` function to set the property of the variable `Power` after reading the dataset.

```
df1 <- read.csv(file.path("data", "3_PlasmaEtching.csv"))
df1$Power <- as.factor(df1$Power)
```

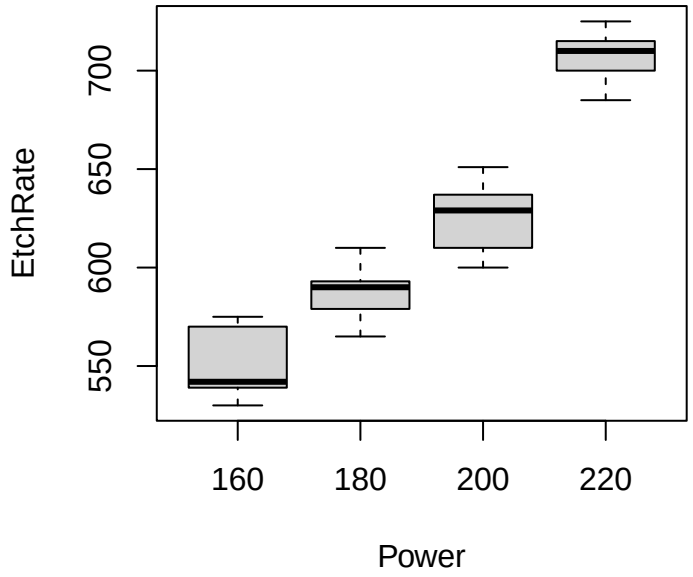
To compute descriptive statistics of the data in each subgroup of a dataset in R, we use `tapply()`.

```
tapply(df1$EtchRate, df1$Power, summary)
```

```
## $`160`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   530.0   539.0   542.0   551.2   570.0   575.0
##
## $`180`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   565.0   579.0   590.0   587.4   593.0   610.0
##
## $`200`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   600.0   610.0   629.0   625.4   637.0   651.0
##
## $`220`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     685     700     710     707     715     725
```

Alternatively, boxplots provide a quick and direct means of observing the differences among the responses of the four treatments (groups or levels of a factor).

```r
# Draw the grouped boxplot
boxplot(EtchRate ~ Power, data = df1)
```



The function `aov()` fits the ANOVA model. For one-way ANOVA, the command is as follows. Then, we call `summary()` to examine the ANOVA table.

```r
fit <- aov(EtchRate ~ Power, data = df1)
summary(fit)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Power        3  66871   22290    66.8 2.88e-09 ***
## Residuals   16   5339     334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model Adequacy Checking

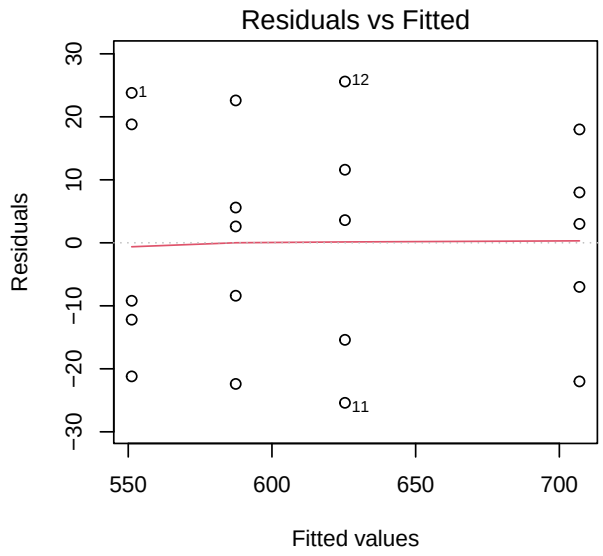
The adequacy of an ANOVA model can be studied from residual plots. The basic approach is to use the `plot()` function with the fitted ANOVA model object as its input argument. Since there are four residual plots, we can use `par(mfrow = c(2, 2))` before the `plot()` function to view all of them simultaneously.

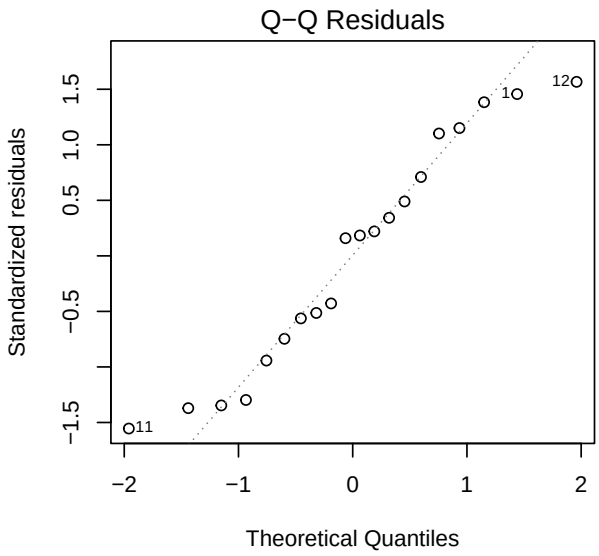
The first (upper left) plot is the residual plot against the fitted values. This plot is used to check the consistency of the variance with changes in the fitted value. A lack of any visually obvious pattern in the dots on the plot is desired.

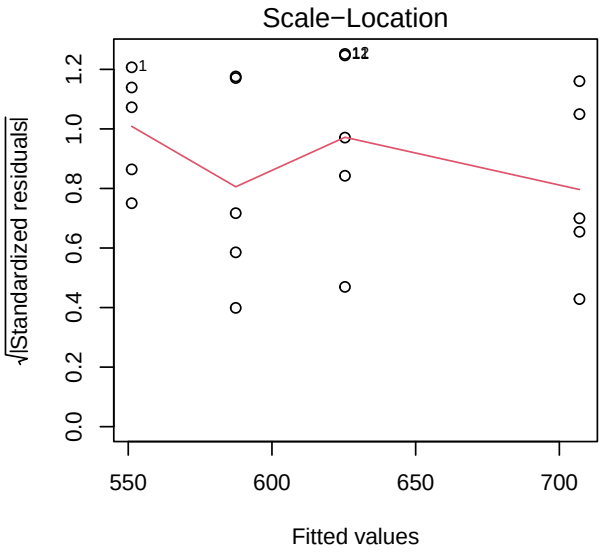
The second (upper right) plot is the residuals' Normal Quantile-Quantile (QQ) plot. Ideally, the dots form a straight line.

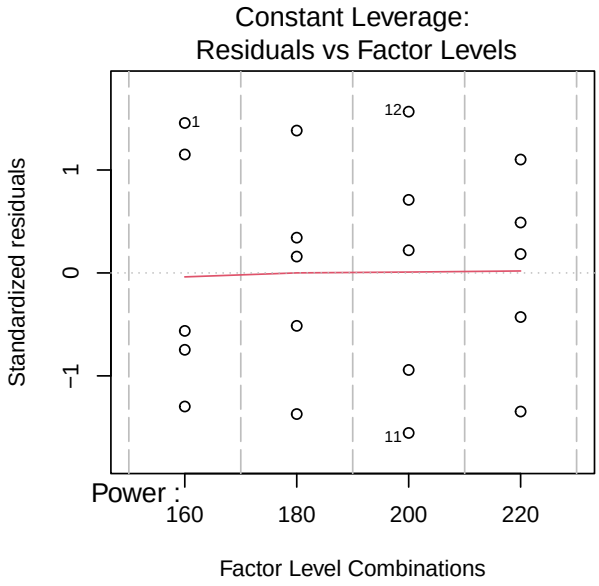
The remaining two plots at the bottom are standardized residuals against the fitted values and standardized residuals against the factor levels, respectively. They are also used to check the consistency of the variance.

```r
par(mfrow = c(2, 2))
plot(fit)
```










```
par(mfrow = c(1, 1))
```

Post-ANOVA Comparison of Means

The estimate of the overall mean μ and the Power's treatment effects τ_1 to τ_4 are

$$\hat{\mu} = \frac{1}{an}\sum_{i=1}^{a}\sum_{j=1}^{n} y_{ij} = \bar{y}_{..};$$

$$\hat{\tau}_1 = \frac{1}{n}\sum_{j=1}^{n} y_{1j} - \hat{\mu} = \bar{y}_{1.} - \bar{y}_{..}; \hat{\tau}_2 = \frac{1}{n}\sum_{j=1}^{n} y_{2j} - \hat{\mu} = \bar{y}_{2.} - \bar{y}_{..};$$

$$\hat{\tau}_3 = \frac{1}{n}\sum_{j=1}^{n} y_{3j} - \hat{\mu} = \bar{y}_{3.} - \bar{y}_{..}; \hat{\tau}_4 = \frac{1}{n}\sum_{j=1}^{n} y_{4j} - \hat{\mu} = \bar{y}_{4.} - \bar{y}_{..}.$$

The R codes are as follows.

```
mean(df1$EtchRate) # Overall
mean(df1$EtchRate[df1$Power == 160]) - mean(df1$EtchRate) # tau_1
mean(df1$EtchRate[df1$Power == 180]) - mean(df1$EtchRate) # tau_2
mean(df1$EtchRate[df1$Power == 200]) - mean(df1$EtchRate) # tau_3
mean(df1$EtchRate[df1$Power == 220]) - mean(df1$EtchRate) # tau_4
```

Following an ANOVA in which we have rejected the null hypothesis of equal treatment means, we wish to test all pairwise mean comparisons:

$$H_0: \ \mu_i = \mu_j$$
$$H_1: \ \mu_i \neq \mu_j$$

for all $i \neq j$. Here, we introduce three approaches.

Pairwise t-tests

The straightforward approach to test for all pairs of the hypotheses is to conduct the Pairwise t-tests simultaneously. The following codes give the results under Bonferroni adjustment on the p-value.

```
pairwise.t.test(df1$EtchRate, df1$Power, p.adjust = "bonferroni")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df1$EtchRate and df1$Power
##
##     160     180     200
## 180 0.038   -       -
## 200 5.1e-05 0.028   -
## 220 2.2e-09 1.0e-07 1.6e-05
##
## P value adjustment method: bonferroni
```

Tukey's Test

Tukey's procedure makes use of the distribution of the studentized range statistic

$$q = \frac{\bar{y}_{max} - \bar{y}_{min}}{\sqrt{MS_E/n}}$$

where $\bar{y}_{max}$ and $\bar{y}_{min}$ are the largest and smallest sample means respectively, out of a group of p sample means. For equal sample sizes, Tukey's test declares two means significantly different if the absolute value of their sample differences exceeds

$$T_\alpha = q_\alpha(a, f)\sqrt{\frac{MS_E}{n}}$$

where $q_\alpha(a, f)$ is the upper α percentage points of q and f is the number of degrees of freedom associated with the MS_E. For more insights on the distribution of q, please refer to the textbook. Tukey's method is performed by the function `TukeyHSD()`.

```
TukeyHSD(fit)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = EtchRate ~ Power, data = df1)
##
## $Power
##          diff        lwr       upr     p adj
## 180-160  36.2   3.145624  69.25438 0.0294279
## 200-160  74.2  41.145624 107.25438 0.0000455
## 220-160 155.8 122.745624 188.85438 0.0000000
## 200-180  38.0   4.945624  71.05438 0.0215995
## 220-180 119.6  86.545624 152.65438 0.0000001
## 220-200  81.6  48.545624 114.65438 0.0000146
```

Fisher's LSD Method

The R package **agricolae** provides the function `LSD.test()` to perform Fisher's LSD test. Adjustment for the P-value is necessary. Typically, we set `p.adj = "bonferroni"` for the Bonferroni method.

```
if (!("agricolae" %in% rownames(installed.packages()))) {
  install.packages("agricolae")
}
library(agricolae)
out <- LSD.test(fit, "Power", p.adj = "bonferroni")
print(out)
```

```
## $statistics
##   MSerror Df   Mean       CV  t.value      MSD
##     333.7 16 617.75 2.957095 3.008334 34.75635
##
## $parameters
##         test  p.ajusted name.t ntr alpha
##   Fisher-LSD bonferroni  Power   4  0.05
##
## $means
##     EtchRate      std r       se      LCL      UCL Min Max Q25 Q50 Q75
## 160    551.2 20.01749 5 8.169455 533.8815 568.5185 530 575 539 542 570
## 180    587.4 16.74216 5 8.169455 570.0815 604.7185 565 610 579 590 593
## 200    625.4 20.52559 5 8.169455 608.0815 642.7185 600 651 610 629 637
## 220    707.0 15.24795 5 8.169455 689.6815 724.3185 685 725 700 710 715
##
## $comparison
```

```
## NULL
##
## $groups
##     EtchRate groups
## 220    707.0      a
## 200    625.4      b
## 180    587.4      c
## 160    551.2      d
##
## attr(,"class")
## [1] "group"
```

The most important parts of the outputs are shown below:

- `$means` displays the estimated mean of the etching rate at each level of power.
- `$groups` indicates the significance of the difference in the etching rate at each level of power. The column groups in `$groups` encodes the treatment levels with no significant difference in the etching rate by the same alphabet letter.

Simluate for Observing Robustness of ANOVA

The Error is Normally Distributed

Understand the meaning of the type I error.

The effect model is defined as

$$y_{ij} = \mu + \tau_i + \varepsilon_{ij}, i = 1, \ldots, a, j = 1, \ldots, n$$

where the error is assumed Normally distributed with mean 0 and variance σ^2.

$$\varepsilon_{ij} \sim N(0, \sigma^2)$$

The simulation is based on the Completely Randomized Design (CRD) of one factor with $a = 4$ treatment levels. The parameters of the true model are set as follows:

```
mu_t <- 0 # the true overall mean
tau_t <- c(-3, -1, 1, 3) # the true effects of the 4 levels
a <- length(tau_t)
n <- 5 # sample size of a treatment level
x <- as.factor(rep(1:a, each = n))
sig <- 2 # assumed standard deviation of the normal distribution
```

Let the significance level to be 0.05 and the simulation below creates 2000 independent scenarios with the data that under the null hypothesis $\tau_1 = ... = \tau_a = 0$. The aim is to count how many times, out of 2000 scenarios, that the F-tests reject H_0 when H_0 is known to be true.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Normal error
  # under Null hypothesis \tau_1 = ... = \tau_a = 0
  y <- mu_t + rnorm(n*a, 0, sig)
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
  indv_res[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test rejects H_0 when H_0 is true (type I error)
indv_signif <- sapply(1:n_sim, function(i) indv_res[[i]]$signif)
sim_t1e <- sum(indv_signif)/n_sim
print(sim_t1e)
```

```
## [1] 0.05
```

The simulation shows that there are 100 out of 2000 scenarios that, although H_0 is true, the F-test still reject H_0. Therefore, the simulated type I error is 0.05, which is closed to the claimed significance level 0.05.

The baseline of the type II error is also necessary so that we can observe type II errors when changing the error assumption. To simulate the type II error, the response is the sum of the true overall, the true effects and the random error.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res_2 <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Normal error
  # under Alternative hypothesis with true effect \tau = c(-3, -1, 1, 3)
  effects <- rep(tau_t, each = n)
  y <- mu_t + effects + rnorm(n*a, 0, sig)
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
  indv_res_2[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test does not rejects H_0 when H_1 is true (type II error)
indv_signif_2 <- sapply(1:n_sim, function(i) indv_res_2[[i]]$signif)
sim_t2e <- sum(1 - indv_signif_2)/n_sim
print(sim_t2e)
```

```
## [1] 0.024
```

The simulation shows that there are 48 out of 2000 scenarios that, when H_1 is true, the F-test fails to reject H_0. Therefore, the simulated type II error is 0.024.

The Error is not Normally Distributed

When the assumption of Normally distributed error is broken, the probability of F-test making mistakes may no longer to be easily controlled at the pre-specified significance level 0.05. Let's replace the Normal distribution of the simulation setup by, for example, Exponential distribution and see the resulting type I error.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res_exp <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Exponentially distributed error
  # under Null hypothesis \tau_1 = ... = \tau_a = 0
  y <- mu_t + rexp(n*a, 0.5)
```

```
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
  indv_res_exp[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test rejects H_0 when H_0 is true (type I error)
indv_signif_exp <- sapply(1:n_sim, function(i) indv_res_exp[[i]]$signif)
sim_t1e_exp <- sum(indv_signif_exp)/n_sim
print(sim_t1e_exp)
```

```
## [1] 0.034
```

The simulation shows that there are 68 out of 2000 scenarios that the F-test still reject H_0 when H_0 is true. That is, the simulated type I error is 0.034 when assume Exponentially distributed error, which is less than the claimed significance level 0.05.

Let's check the type II error of the F-test when the population of the random error is not Normal distribution.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res_exp_2 <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Exponentially distributed error
  # under Alternative hypothesis with true effect \tau = c(-3, -1, 1, 3)
  effects <- rep(tau_t, each = n)
  y <- mu_t + effects + rexp(n*a, 0.5)
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
  indv_res_exp_2[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test does not rejects H_0 when H_1 is true (type II error)
indv_signif_exp_2 <- sapply(1:n_sim, function(i) indv_res_exp_2[[i]]$signif)
sim_t2e_exp <- sum(1 - indv_signif_exp_2)/n_sim
print(sim_t2e_exp)
```

```
## [1] 0.0545
```

The simulated type II error is 0.0545 when assume Exponentially distributed error, which is more than twice as the type II error in the case of Normal assumption.

The Error is Normally Distributed but Variance Is Inconsistent

When the assumption of constant variance is broken, the probability of F-test making mistakes may no longer to be easily controlled at the pre-specified significance level 0.05. To observe the consequence of the violation of the constant variance assumption, let's varying the variance of the random error by treatment levels and see the resulting type I error.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res_ncv <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Normal error but vary the variance across treatment levels
  # under Null hypothesis \tau_1 = ... = \tau_a = 0
  err_multi <- rep(1:a, each = n)
  y <- mu_t + rnorm(n*a)*err_multi
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
  indv_res_ncv[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test rejects H_0 when H_0 is true (type I error)
indv_signif_ncv <- sapply(1:n_sim, function(i) indv_res_ncv[[i]]$signif)
sim_t1e_ncv <- sum(indv_signif_ncv)/n_sim
print(sim_t1e_ncv)
```

```
## [1] 0.0685
```

The simulation shows that there are 137 out of 2000 scenarios that the F-test still reject H_0 when H_0 is true. That is, the simulated type I error is 0.0685 for Normally distributed error with inconsistent variance, which is larger to the claimed significance level 0.05.

```
alpha <- 0.05 # significance level
n_sim <- 2000
indv_res_ncv_2 <- vector("list", length = n_sim)
# Simulate n_sim times with different data randomly drawn from the fixed population
for (i in 1:n_sim) {
  set.seed(i) # set seed for reproducity
  # Generate response with Normal error but vary the variance across treatment levels
  # under Alternative hypothesis with true effect \tau = c(-3, -1, 1, 3)
  effects <- rep(tau_t, each = n)
  err_multi <- rep(1:a, each = n)
  y <- mu_t + effects + rnorm(n*a)*err_multi
  # compute ANOVA table and save the p-value
  fit <- aov(y ~ x)
  fit_sum <- summary(fit)
  pval <- fit_sum[[1]][1, 5]
```

```r
  indv_res_ncv_2[[i]] <- list(
    "y" = y,
    "fit" = fit,
    "signif" = pval < alpha
  )
}
# Count the times that the F-test does not rejects H_0 when H_1 is true (type II error)
indv_signif_ncv_2 <- sapply(1:n_sim, function(i) indv_res_ncv_2[[i]]$signif)
sim_t2e_ncv <- sum(1 - indv_signif_ncv_2)/n_sim
print(sim_t2e_ncv)
```

```
## [1] 0.232
```

The simulated type II error is 0.232 when the variance is not constant, which much more than the type II error in the case of Normal assumption.