Week 7: Latin Squares & 2^2 Factorial (Lab)

PSTAT122: Design and Analysis of Experiments

true

Contents

```
knitr::opts_chunk$set(error =  FALSE)
```

1 Overview

This section teaches Latin-square blocking and a simple 2^2 factorial analysis in R. By the end you should be able to:

- create and analyze a Latin square in R using `aov()`,
- run diagnostics on the fitted model,
- interpret ANOVA for blocked designs, and
- simulate a 2^2 factorial and estimate (via simulation) its power.

2 Latin Square

2.1 What is a Latin square?

A Latin square of order p places p treatments so each treatment appears once per row and once per column.

— it blocks two nuisance factors (rows, columns). Use it when there are two systematic nuisance sources (e.g., batches and operators).

Note that a Latin Square is an incomplete design: not all treatment combinations are observed (unlike a full factorial).

An assumption that we make when using a Latin square design is that the three factors (treatments, and two nuisance factors) do not interact. If this assumption is violated, the Latin Square design error term will be inflated.

Situations where you should use a Latin Square are where you have a single treatment factor and you have two blocking or nuisance factors to consider, which can have the same number of levels as the treatment factor.

2.2 Simulated 4X4 Latin square

We simulate a 4×4 Latin square with treatments A–D, where rows = batches and columns = operators.

```
# Build a standard 4x4 Latin square (standard square)

latin4 <- matrix(c("A","B","C","D",
"B","C","D","A",
"C","D","A","B",
```

```
"D","A","B","C"),
nrow=4, byrow=TRUE)

# Create a data frame with one observation per cell

p <- 4
treats <- as.vector(latin4)
rows <- rep(1:p, each=p)       # row index
cols <- rep(1:p, times=p)      # column index

# Simulate a response: baseline + treatment effects + random error

set.seed(20251103)
mu <- 10
t_effects <- c(A=0, B=1.2, C=-0.5, D=0.3)   # true treatment effects
y <- numeric(p*p)
for(i in 1:(p*p)){
trt <- treats[i]

# small row and column effects

row_eff <- c(0, 0.6, -0.4, 0.2)[rows[i]]
col_eff <- c(0, -0.3, 0.5, -0.2)[cols[i]]
y[i] <- mu + t_effects[trt] + row_eff + col_eff + rnorm(1, 0, 1.5)
}

latin_df <- data.frame(y = y,
    trt = factor(treats),
    row = factor(rows),
    col = factor(cols)
)

head(latin_df)
```

```
##           y trt row col
## 1 10.749978   A   1   1
## 2 12.812171   B   1   2
## 3  9.183537   C   1   3
## 4 10.192350   D   1   4
## 5 12.001114   B   2   1
## 6  9.775260   C   2   2
```

We have a balanced Latin-square layout with one observation per treatment-row-column combination.

2.3 Fit Latin square ANOVA (simulated data)

We fit the standard Latin-square model $y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_k + \epsilon_{ijk}$ where:

- α_i = treatment effect
- β_j = row (batch) effect
- γ_k = column (operator) effect

```
model_sim <- aov(y ~ trt + row + col, data=latin_df)
summary(model_sim)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
```