Paper Helicopter Experiment

Nayeemuddin Mohammed

2026-05-09

Table of contents


```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

```
library(knitr)
library(kableExtra)
```

```
Attaching package: 'kableExtra'
```

```
The following object is masked from 'package:dplyr':

    group_rows
```

```r
library(ggplot2)
library(plotly)
```

```
Attaching package: 'plotly'

The following object is masked from 'package:ggplot2':

    last_plot

The following object is masked from 'package:stats':

    filter

The following object is masked from 'package:graphics':

    layout
```

```r
library(corrplot)
```

```
corrplot 0.95 loaded
```

```r
library(car)
```

```
Loading required package: carData

Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode

The following object is masked from 'package:purrr':

    some
```

```r
library(broom)
library(GGally)
library(patchwork)
library(tinytex)
```

```r
# Set theme for plots
theme_set(theme_minimal() +
          theme(plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
                plot.subtitle = element_text(hjust = 0.5, size = 12)))
```

```r
helicopter_data <- data.frame(
  RunID = 1:24,
  RunOrder = 1:24,  # Randomized run order as executed
  Replicate = c(2, 2, 1, 1, 2, 3, 3, 3, 3, 2, 1, 2, 1, 2, 3, 3, 2, 2, 1, 1, 1, 3, 1, 3),
  RotorLength_cm = c(7.5, 8.5, 8.5, 7.5, 8.5, 8.5, 7.5, 7.5, 8.5, 7.5, 7.5, 7.5, 7.5, 8.5, 8.5
  RotorWidth_cm = c(3.5, 5, 5, 3.5, 3.5, 5, 3.5, 5, 3.5, 5, 5, 5, 3.5, 3.5, 3.5, 3.5, 5, 3.5, 3
  PaperClip = c(2, 0, 0, 0, 0, 2, 0, 2, 2, 0, 2, 2, 2, 2, 0, 2, 2, 0, 2, 0, 0, 0, 2, 0),
  Time_s = c(3.03, 3.42, 3.75, 3.4, 4.12, 3.07, 3.31, 2.32, 3.4, 2.62, 2.98, 2.45, 3.03, 3.25,
)

# Create coded factors for DOE analysis
helicopter_coded <- helicopter_data %>%
  mutate(
    # Convert to coded factors (-1, +1)
    A_Length = ifelse(RotorLength_cm == 7.5, -1, 1),
    B_Width = ifelse(RotorWidth_cm == 3.5, -1, 1),
    C_Clip = ifelse(PaperClip == 0, -1, 1),

    # Create factor labels for visualization
    Length_Factor = factor(ifelse(A_Length == -1, "Short (7.5cm)", "Long (8.5cm)")),
    Width_Factor = factor(ifelse(B_Width == -1, "Narrow (3.5cm)", "Wide (5.0cm)")),
    Clip_Factor = factor(ifelse(C_Clip == -1, "No Clip (0)", "With Clip (2)")),

    # Create treatment combination labels
    Treatment = paste0(
      ifelse(A_Length == 1, "a", ""),
      ifelse(B_Width == 1, "b", ""),
      ifelse(C_Clip == 1, "c", "")
    ),
    Treatment = ifelse(Treatment == "", "(1)", Treatment)
  )

# Display first few rows
helicopter_coded %>%
  select(RunID, RotorLength_cm, RotorWidth_cm, PaperClip, Time_s, A_Length, B_Width, C_Clip, T
  head(10) %>%
  kable(caption = "First 10 rows of experimental data with coded factors") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 1: First 10 rows of experimental data with coded factors

RunID	RotorLength_cm	RotorWidth_cm	PaperClip	Time_s	A_Length	B_Width	C_Clip	Treat
1	7.5	3.5	2	3.03	-1	-1	1	c
2	8.5	5.0	0	3.42	1	1	-1	ab
3	8.5	5.0	0	3.75	1	1	-1	ab
4	7.5	3.5	0	3.40	-1	-1	-1	(1)
5	8.5	3.5	0	4.12	1	-1	-1	a
6	8.5	5.0	2	3.07	1	1	1	abc
7	7.5	3.5	0	3.31	-1	-1	-1	(1)
8	7.5	5.0	2	2.32	-1	1	1	bc
9	8.5	3.5	2	3.40	1	-1	1	ac
10	7.5	5.0	0	2.62	-1	1	-1	b

```
design_summary <- helicopter_coded %>%
  group_by(A_Length, B_Width, C_Clip, Treatment) %>%
  summarise(
    n_runs = n(),
    mean_time = round(mean(Time_s), 3),
    sd_time = round(sd(Time_s), 3),
    .groups = 'drop'
  ) %>%
  arrange(A_Length, B_Width, C_Clip)

design_summary %>%
  kable(
    caption = "2³ Factorial Design: Treatment Combinations and Summary Statistics",
    col.names = c("Length (A)", "Width (B)", "Clip (C)", "Treatment", "Replicates", "Mean Time
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 2: 2^3 Factorial Design: Treatment Combinations and Summary Statistics

Length (A)	Width (B)	Clip (C)	Treatment	Replicates	Mean Time (s)	SD Time (s)
-1	-1	-1	(1)	3	3.407	0.100
-1	-1	1	c	3	3.007	0.040
-1	1	-1	b	3	2.513	0.110
-1	1	1	bc	3	2.583	0.350
1	-1	-1	a	3	4.147	0.031
1	-1	1	ac	3	3.367	0.104
1	1	-1	ab	3	3.610	0.171
1	1	1	abc	3	3.110	0.069

```
cat("\n**Design Characteristics:**\n")
```

Design Characteristics:

```r
cat("- Factors: 3 (Length, Width, Paper Clips)\n")
```

- Factors: 3 (Length, Width, Paper Clips)

```r
cat("- Levels per factor: 2\n")
```

- Levels per factor: 2

```r
cat("- Design: 2³ Full Factorial\n")
```

- Design: 2^3 Full Factorial

```r
cat("- Total treatment combinations: 8\n")
```

- Total treatment combinations: 8

```r
cat("- Total runs:", nrow(helicopter_coded), "\n")
```

- Total runs: 24

```r
cat("- Replication: 3 replicates per treatment combination\n")
```

- Replication: 3 replicates per treatment combination

```r
cat("- Randomization: Runs executed in randomized order\n")
```

- Randomization: Runs executed in randomized order

```r
cat("- Response: Flight time from release to ground contact (seconds)\n")
```

- Response: Flight time from release to ground contact (seconds)

```r
overall_summary <- helicopter_coded %>%
  summarise(
    n = n(),
    mean_time = round(mean(Time_s), 3),
    median_time = round(median(Time_s), 3),
    sd_time = round(sd(Time_s), 3),
    min_time = round(min(Time_s), 3),
    max_time = round(max(Time_s), 3),
    range_time = round(max(Time_s) - min(Time_s), 3)
  )

overall_summary %>%
  kable(caption = "Overall Summary Statistics for Flight Time") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 3: Overall Summary Statistics for Flight Time

n	mean_time	median_time	sd_time	min_time	max_time	range_time
24	3.218	3.22	0.53	2.32	4.18	1.86

```r
# Factor-level summaries
factor_summaries <- list(
  Length = helicopter_coded %>%
    group_by(Length_Factor) %>%
    summarise(n = n(), mean_time = round(mean(Time_s), 3), sd_time = round(sd(Time_s), 3)),

  Width = helicopter_coded %>%
    group_by(Width_Factor) %>%
    summarise(n = n(), mean_time = round(mean(Time_s), 3), sd_time = round(sd(Time_s), 3)),

  Clip = helicopter_coded %>%
    group_by(Clip_Factor) %>%
    summarise(n = n(), mean_time = round(mean(Time_s), 3), sd_time = round(sd(Time_s), 3))
)

# Display factor summaries
cat("### Factor Level Summaries\n\n")
```

```
### Factor Level Summaries
```

```r
for(factor_name in names(factor_summaries)) {
  cat("**", factor_name, "Effect:**\n")
  print(kable(factor_summaries[[factor_name]],
              caption = paste(factor_name, "levels summary")) %>%
        kable_styling(bootstrap_options = c("striped", "hover", "condensed")))
```

```r
  cat("\n")
}
```

```
** Length Effect:**

\begin{longtable}[t]{lrrr}
\caption{\label{tab:unnamed-chunk-4}Length levels summary}\\
\toprule
Length\_Factor & n & mean\_time & sd\_time\\
\midrule
Long (8.5cm) & 12 & 3.558 & 0.410\\
Short (7.5cm) & 12 & 2.878 & 0.409\\
\bottomrule
\end{longtable}

** Width Effect:**

\begin{longtable}[t]{lrrr}
\caption{\label{tab:unnamed-chunk-4}Width levels summary}\\
\toprule
Width\_Factor & n & mean\_time & sd\_time\\
\midrule
Narrow (3.5cm) & 12 & 3.482 & 0.438\\
Wide (5.0cm) & 12 & 2.954 & 0.495\\
\bottomrule
\end{longtable}

** Clip Effect:**

\begin{longtable}[t]{lrrr}
\caption{\label{tab:unnamed-chunk-4}Clip levels summary}\\
\toprule
Clip\_Factor & n & mean\_time & sd\_time\\
\midrule
No Clip (0) & 12 & 3.419 & 0.623\\
With Clip (2) & 12 & 3.017 & 0.335\\
\bottomrule
\end{longtable}
```

0.1 Data Visualization

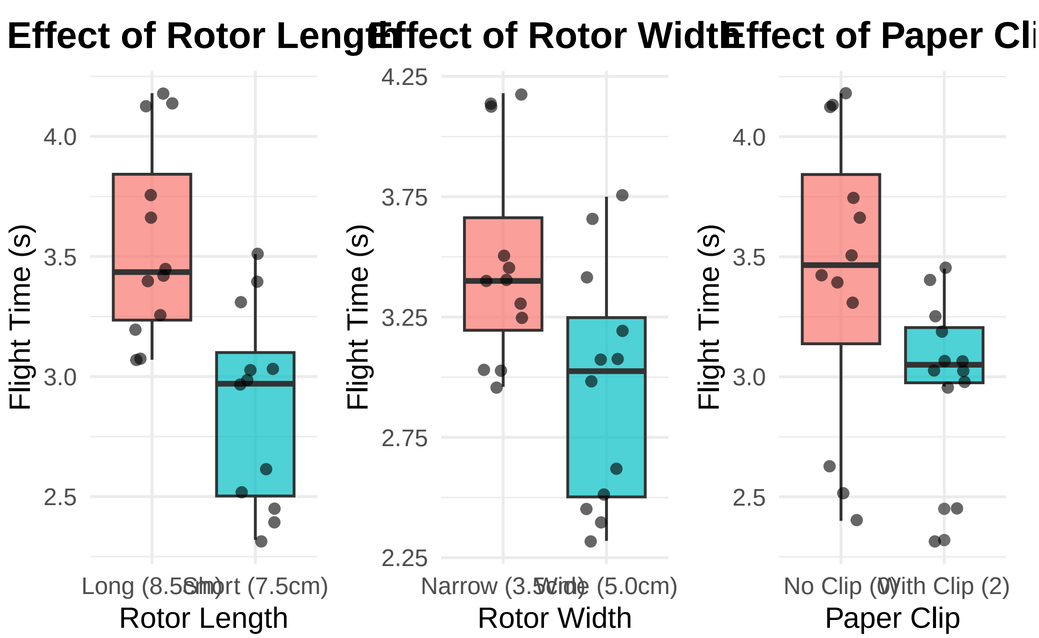
```r
p1 <- helicopter_coded %>%
  ggplot(aes(x = Length_Factor, y = Time_s)) +
  geom_boxplot(aes(fill = Length_Factor), alpha = 0.7) +
  geom_jitter(width = 0.2, alpha = 0.6) +
```

```r
  labs(title = "Effect of Rotor Length", x = "Rotor Length", y = "Flight Time (s)") +
  theme(legend.position = "none")

p2 <- helicopter_coded %>%
  ggplot(aes(x = Width_Factor, y = Time_s)) +
  geom_boxplot(aes(fill = Width_Factor), alpha = 0.7) +
  geom_jitter(width = 0.2, alpha = 0.6) +
  labs(title = "Effect of Rotor Width", x = "Rotor Width", y = "Flight Time (s)") +
  theme(legend.position = "none")

p3 <- helicopter_coded %>%
  ggplot(aes(x = Clip_Factor, y = Time_s)) +
  geom_boxplot(aes(fill = Clip_Factor), alpha = 0.7) +
  geom_jitter(width = 0.2, alpha = 0.6) +
  labs(title = "Effect of Paper Clip", x = "Paper Clip", y = "Flight Time (s)") +
  theme(legend.position = "none")

# Combine plots
library(patchwork)
p1 + p2 + p3
```




```r
# Interaction plots
library(patchwork)
int1 <- helicopter_coded %>%
  group_by(Length_Factor, Width_Factor) %>%
  summarise(mean_time = mean(Time_s), .groups = 'drop') %>%
  ggplot(aes(x = Length_Factor, y = mean_time, color = Width_Factor, group = Width_Factor)) +
```

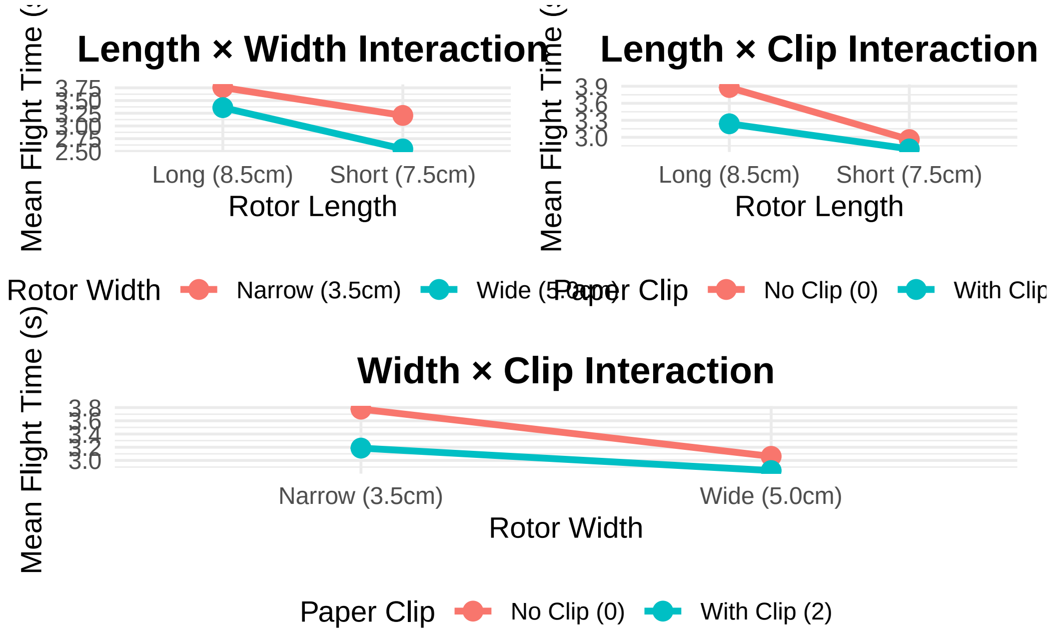
```r
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  labs(title = "Length × Width Interaction",
       x = "Rotor Length", y = "Mean Flight Time (s)",
       color = "Rotor Width") +
  theme(legend.position = "bottom")
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```

```r
int2 <- helicopter_coded %>%
  group_by(Length_Factor, Clip_Factor) %>%
  summarise(mean_time = mean(Time_s), .groups = 'drop') %>%
  ggplot(aes(x = Length_Factor, y = mean_time, color = Clip_Factor, group = Clip_Factor)) +
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  labs(title = "Length × Clip Interaction",
       x = "Rotor Length", y = "Mean Flight Time (s)",
       color = "Paper Clip") +
  theme(legend.position = "bottom")

int3 <- helicopter_coded %>%
  group_by(Width_Factor, Clip_Factor) %>%
  summarise(mean_time = mean(Time_s), .groups = 'drop') %>%
  ggplot(aes(x = Width_Factor, y = mean_time, color = Clip_Factor, group = Clip_Factor)) +
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  labs(title = "Width × Clip Interaction",
       x = "Rotor Width", y = "Mean Flight Time (s)",
       color = "Paper Clip") +
  theme(legend.position = "bottom")

(int1| int2) / int3
```




```
# Calculate effects using the standard DOE approach (Yates algorithm)
# Get average response for each treatment combination
treatment_means <- helicopter_coded %>%
  group_by(A_Length, B_Width, C_Clip, Treatment) %>%
  summarise(mean_response = mean(Time_s), .groups = 'drop') %>%
  arrange(A_Length, B_Width, C_Clip)

# Extract the 8 treatment combination means in standard order
y <- treatment_means$mean_response
names(y) <- treatment_means$Treatment

# Calculate effects using contrast coefficients (standard DOE method)
n_reps <- 3  # number of replicates per treatment combination

# Main effects (contrast: high level - low level / 4)
A_effect <- (sum(y[c("a", "ab", "ac", "abc")]) - sum(y[c("(1)", "b", "c", "bc")])) / 4
B_effect <- (sum(y[c("b", "ab", "bc", "abc")]) - sum(y[c("(1)", "a", "c", "ac")])) / 4
C_effect <- (sum(y[c("c", "ac", "bc", "abc")]) - sum(y[c("(1)", "a", "b", "ab")])) / 4

# Two-factor interactions
AB_effect <- (sum(y[c("ab", "abc")]) + sum(y[c("(1)", "c")]) - sum(y[c("a", "ac")]) - sum(y[c(
AC_effect <- (sum(y[c("ac", "abc")]) + sum(y[c("(1)", "b")]) - sum(y[c("a", "ab")]) - sum(y[c(
BC_effect <- (sum(y[c("bc", "abc")]) + sum(y[c("(1)", "a")]) - sum(y[c("b", "ab")]) - sum(y[c(

# Three-factor interaction
ABC_effect <- (sum(y[c("abc", "(1)", "ab", "c")]) - sum(y[c("a", "b", "ac", "bc")])) / 4

# Calculate standard errors for effects (assuming equal variance)
```

```
mse <- sum((helicopter_coded$Time_s - ave(helicopter_coded$Time_s, helicopter_coded$Treatment)
se_effect <- sqrt(mse / (4 * n_reps))  # Standard error for effects

# Create effects summary with statistical testing
effects_summary <- tibble(
  Effect = c("A (Length)", "B (Width)", "C (Clip)", "AB", "AC", "BC", "ABC"),
  Estimate = round(c(A_effect, B_effect, C_effect, AB_effect, AC_effect, BC_effect, ABC_effect
  SE = round(rep(se_effect, 7), 4),
  t_stat = round(c(A_effect, B_effect, C_effect, AB_effect, AC_effect, BC_effect, ABC_effect) 
  Abs_Effect = round(abs(c(A_effect, B_effect, C_effect, AB_effect, AC_effect, BC_effect, ABC_
) %>%
  arrange(desc(Abs_Effect))

effects_summary %>%
  kable(caption = "Factorial Effects Summary (Ranked by Absolute Magnitude)") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 4: Factorial Effects Summary (Ranked by Absolute Magnitude)

Effect	Estimate	SE	t_stat	Abs_Effect
A (Length)	0.6808	0.0447	15.236	0.6808
B (Width)	-0.5275	0.0447	-11.804	0.5275
C (Clip)	-0.4025	0.0447	-9.007	0.4025
AC	-0.2375	0.0447	-5.315	0.2375
BC	0.1875	0.0447	4.196	0.1875
AB	0.1308	0.0447	2.928	0.1308
ABC	0.1308	0.0447	2.928	0.1308

```
# Display treatment combination means in standard order
treatment_table <- treatment_means %>%
  select(Treatment, A_Length, B_Width, C_Clip, mean_response) %>%
  mutate(
    Length_Level = ifelse(A_Length == -1, "Short (7.5)", "Long (8.5)"),
    Width_Level = ifelse(B_Width == -1, "Narrow (3.5)", "Wide (5.0)"),
    Clip_Level = ifelse(C_Clip == -1, "No Clip (0)", "With Clip (2)")
  ) %>%
  select(Treatment, Length_Level, Width_Level, Clip_Level, mean_response) %>%
  arrange(mean_response)

treatment_table %>%
  kable(caption = "Treatment Combination Means (Ordered by Flight Time)",
        col.names = c("Treatment", "Length", "Width", "Clips", "Mean Time (s)")) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 5: Treatment Combination Means (Ordered by Flight Time)

Treatment	Length	Width	Clips	Mean Time (s)
b	Short (7.5)	Wide (5.0)	No Clip (0)	2.513333
bc	Short (7.5)	Wide (5.0)	With Clip (2)	2.583333
c	Short (7.5)	Narrow (3.5)	With Clip (2)	3.006667
abc	Long (8.5)	Wide (5.0)	With Clip (2)	3.110000
ac	Long (8.5)	Narrow (3.5)	With Clip (2)	3.366667
(1)	Short (7.5)	Narrow (3.5)	No Clip (0)	3.406667
ab	Long (8.5)	Wide (5.0)	No Clip (0)	3.610000
a	Long (8.5)	Narrow (3.5)	No Clip (0)	4.146667

0.2 Anova Analysis

```r
# Fit full factorial model
model_full <- lm(Time_s ~ A_Length * B_Width * C_Clip, data = helicopter_coded)

# ANOVA table
anova_table <- anova(model_full)
print(anova_table)
```

```
Analysis of Variance Table

Response: Time_s
                        Df  Sum Sq Mean Sq  F value    Pr(>F)    
A_Length                 1 2.78120 2.78120 116.0649 9.650e-09 ***
B_Width                  1 1.66954 1.66954  69.6729 3.183e-07 ***
C_Clip                   1 0.97204 0.97204  40.5649 9.306e-06 ***
A_Length:B_Width         1 0.10270 0.10270   4.2860  0.054967 .  
A_Length:C_Clip          1 0.33844 0.33844  14.1236  0.001718 ** 
B_Width:C_Clip           1 0.21094 0.21094   8.8028  0.009084 ** 
A_Length:B_Width:C_Clip  1 0.01354 0.01354   0.5649  0.463188    
Residuals               16 0.38340 0.02396                       
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Model summary
model_summary <- summary(model_full)
print(model_summary)
```

```
Call:
lm(formula = Time_s ~ A_Length * B_Width * C_Clip, data = helicopter_coded)
```

```
Residuals:
     Min       1Q   Median       3Q      Max 
-0.26333 -0.05917  0.00000  0.05750  0.39667 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)              3.21792    0.03160 101.839  < 2e-16 ***
A_Length                 0.34042    0.03160  10.773 9.65e-09 ***
B_Width                 -0.26375    0.03160  -8.347 3.18e-07 ***
C_Clip                  -0.20125    0.03160  -6.369 9.31e-06 ***
A_Length:B_Width         0.06542    0.03160   2.070  0.05497 .  
A_Length:C_Clip         -0.11875    0.03160  -3.758  0.00172 ** 
B_Width:C_Clip           0.09375    0.03160   2.967  0.00908 ** 
A_Length:B_Width:C_Clip -0.02375    0.03160  -0.752  0.46319    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1548 on 16 degrees of freedom
Multiple R-squared:  0.9408,	Adjusted R-squared:  0.9148 
F-statistic:  36.3 on 7 and 16 DF,  p-value: 1.177e-08
```

```r
# Create a tidy ANOVA table
anova_tidy <- tidy(anova_table) %>%
  mutate(
    term = case_when(
      term == "A_Length" ~ "A: Rotor Length",
      term == "B_Width" ~ "B: Rotor Width",
      term == "C_Clip" ~ "C: Paper Clip",
      term == "A_Length:B_Width" ~ "AB: Length × Width",
      term == "A_Length:C_Clip" ~ "AC: Length × Clip",
      term == "B_Width:C_Clip" ~ "BC: Width × Clip",
      term == "A_Length:B_Width:C_Clip" ~ "ABC: Length × Width × Clip",
      term == "Residuals" ~ "Error",
      TRUE ~ term
    ),
    across(where(is.numeric), round, 4)
  ) %>%
  select(Source = term, DF = df, SS = sumsq, MS = meansq, `F-value` = statistic, `p-value` = p
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `across(where(is.numeric), round, 4)`.
Caused by warning:
! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
Supply arguments directly to `.fns` through an anonymous function instead.

  # Previously
```

```
across(a:b, mean, na.rm = TRUE)

# Now
across(a:b, \(x) mean(x, na.rm = TRUE))
```

```r
anova_tidy %>%
  kable(caption = "ANOVA Table for 2³ Factorial Design") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

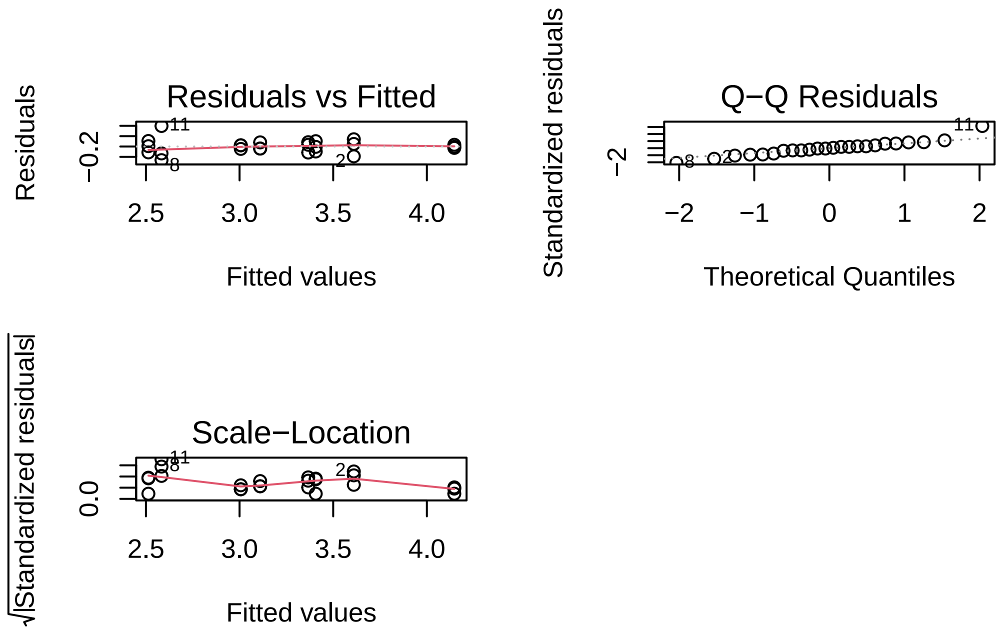
Table 6: ANOVA Table for 2^3 Factorial Design

Source	DF	SS	MS	F-value	p-value
A: Rotor Length	1	2.7812	2.7812	116.0649	0.0000
B: Rotor Width	1	1.6695	1.6695	69.6729	0.0000
C: Paper Clip	1	0.9720	0.9720	40.5649	0.0000
AB: Length × Width	1	0.1027	0.1027	4.2860	0.0550
AC: Length × Clip	1	0.3384	0.3384	14.1236	0.0017
BC: Width × Clip	1	0.2109	0.2109	8.8028	0.0091
ABC: Length × Width × Clip	1	0.0135	0.0135	0.5649	0.4632
Error	16	0.3834	0.0240	NA	NA

0.3 Model Diagnostics

```r
# Create diagnostic plots
par(mfrow = c(2, 2))
plot(model_full)
```

```
hat values (leverages) are all = 0.3333333
 and there are no factor predictors; no plot no. 5
```




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

# Additional diagnostic tests
cat("**Shapiro-Wilk Test for Normality:**\n")
```

```
**Shapiro-Wilk Test for Normality:**
```

```r
shapiro_test <- shapiro.test(residuals(model_full))
print(shapiro_test)
```

```

    Shapiro-Wilk normality test

data:  residuals(model_full)
W = 0.93265, p-value = 0.1117
```

```r
cat("\n**Levene's Test for Homogeneity of Variance:**\n")
```

```

**Levene's Test for Homogeneity of Variance:**
```

```r
levene_test <- leveneTest(Time_s ~ interaction(Length_Factor, Width_Factor, Clip_Factor),
                          data = helicopter_coded)
print(levene_test)
```

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  7  0.9019 0.5286
      16
```

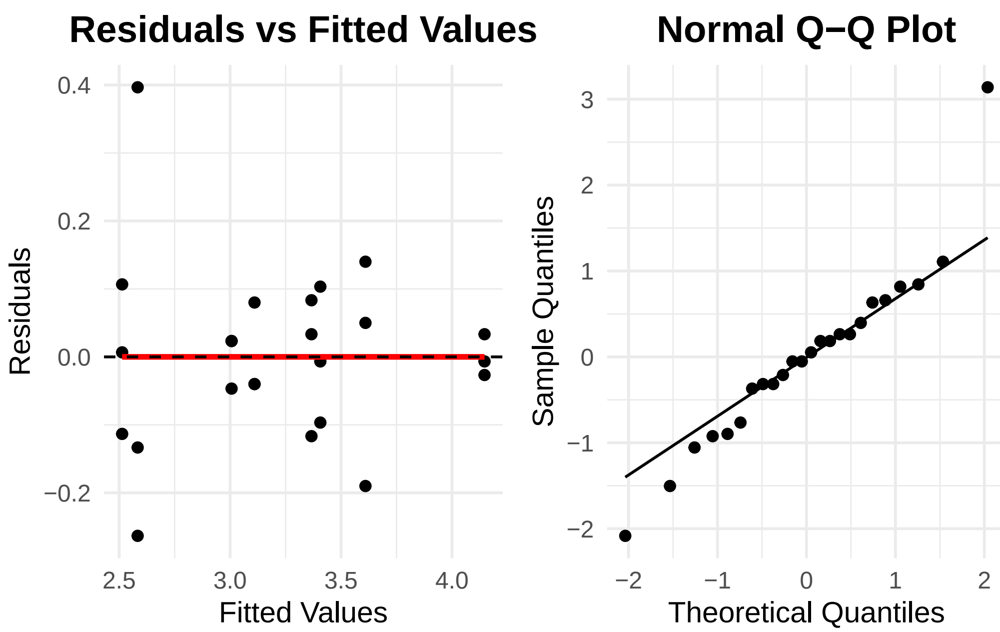
```r
# Residual analysis
residual_data <- helicopter_coded %>%
  mutate(
    fitted = fitted(model_full),
    residuals = residuals(model_full),
    std_residuals = rstandard(model_full)
  )

# Plot residuals vs fitted
res_plot <- residual_data %>%
  ggplot(aes(x = fitted, y = residuals)) +
  geom_point() +
  geom_smooth(method = "loess", se = FALSE, color = "red") +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(title = "Residuals vs Fitted Values",
       x = "Fitted Values", y = "Residuals")

# Normal Q-Q plot
qq_plot <- residual_data %>%
  ggplot(aes(sample = std_residuals)) +
  stat_qq() +
  stat_qq_line() +
  labs(title = "Normal Q-Q Plot",
       x = "Theoretical Quantiles", y = "Sample Quantiles")

res_plot + qq_plot
```

```
`geom_smooth()` using formula = 'y ~ x'
```



1 Results and Interpretation

1.1 Significant Effects

```
# Extract significant effects (p < 0.05)
significant_effects <- anova_tidy %>%
  filter(!is.na(`p-value`), `p-value` < 0.05) %>%
  arrange(`p-value`)

significant_effects %>%
  kable(caption = "Significant Effects (p < 0.05)") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 7: Significant Effects ($p < 0.05$)

Source	DF	SS	MS	F-value	p-value
A: Rotor Length	1	2.7812	2.7812	116.0649	0.0000
B: Rotor Width	1	1.6695	1.6695	69.6729	0.0000
C: Paper Clip	1	0.9720	0.9720	40.5649	0.0000
AC: Length × Clip	1	0.3384	0.3384	14.1236	0.0017
BC: Width × Clip	1	0.2109	0.2109	8.8028	0.0091

```r
# Effect interpretation
cat("### Effect Interpretations:\n\n")
```

Effect Interpretations:

```r
if (abs(A_effect) > 0.1) {
  cat("**Rotor Length (A):**",
      ifelse(A_effect > 0, "Longer rotors increase", "Longer rotors decrease"),
      "flight time by", round(abs(A_effect), 3), "seconds on average.\n")
}
```

Rotor Length (A): Longer rotors increase flight time by 0.681 seconds on average.

```r
if (abs(B_effect) > 0.1) {
  cat("**Rotor Width (B):**",
      ifelse(B_effect > 0, "Wider rotors increase", "Wider rotors decrease"),
      "flight time by", round(abs(B_effect), 3), "seconds on average.\n")
}
```

Rotor Width (B): Wider rotors decrease flight time by 0.527 seconds on average.

```r
if (abs(C_effect) > 0.1) {
  cat("**Paper Clip (C):**",
      ifelse(C_effect > 0, "Adding a paper clip increases", "Adding a paper clip decreases"),
      "flight time by", round(abs(C_effect), 3), "seconds on average.\n")
}
```

Paper Clip (C): Adding a paper clip decreases flight time by 0.402 seconds on average.

2 Optimal Settings

```r
# Find optimal settings for maximum and minimum flight time
optimal_max <- treatment_means %>%
  filter(mean_response == max(mean_response))

optimal_min <- treatment_means %>%
  filter(mean_response == min(mean_response))

cat("### Optimal Settings:\n\n")
```

Optimal Settings:

```r
cat("**For MAXIMUM flight time:**\n")
```

For MAXIMUM flight time:

```r
cat("- Treatment combination:", optimal_max$Treatment, "\n")
```

```
- Treatment combination: a
```

```r
cat("- Rotor Length:", ifelse(optimal_max$A_Length == 1, "8.5 cm (Long)", "7.5 cm (Short)"), "
```

```
- Rotor Length: 8.5 cm (Long)
```

```r
cat("- Rotor Width:", ifelse(optimal_max$B_Width == 1, "5.0 cm (Wide)", "3.5 cm (Narrow)"), "\
```

```
- Rotor Width: 3.5 cm (Narrow)
```

```r
cat("- Paper Clip:", ifelse(optimal_max$C_Clip == 1, "2 clips", "0 clips"), "\n")
```

```
- Paper Clip: 0 clips
```

```r
cat("- Average flight time:", round(optimal_max$mean_response, 3), "seconds\n\n")
```

```
- Average flight time: 4.147 seconds
```

```r
cat("**For MINIMUM flight time:**\n")
```

For MINIMUM flight time:

```r
cat("- Treatment combination:", optimal_min$Treatment, "\n")
```

```
- Treatment combination: b
```

```r
cat("- Rotor Length:", ifelse(optimal_min$A_Length == 1, "8.5 cm (Long)", "7.5 cm (Short)"), "
```

```
- Rotor Length: 7.5 cm (Short)
```

```r
cat("- Rotor Width:", ifelse(optimal_min$B_Width == 1, "5.0 cm (Wide)", "3.5 cm (Narrow)"), "\
```

```
- Rotor Width: 5.0 cm (Wide)
```

```r
cat("- Paper Clip:", ifelse(optimal_min$C_Clip == 1, "2 clips", "0 clips"), "\n")
```

```
- Paper Clip: 0 clips
```

```r
cat("- Average flight time:", round(optimal_min$mean_response, 3), "seconds\n")
```

```
- Average flight time: 2.513 seconds
```

3 Final Model

```r
# Create reduced model with only significant terms (if any)
significant_terms <- anova_tidy %>%
  filter(!is.na(`p-value`), `p-value` < 0.05, Source != "Error") %>%
  pull(Source)

if (length(significant_terms) > 0) {
  cat("### Reduced Model with Significant Terms Only:\n")

  # Build model formula for significant terms
  formula_terms <- character()

  if ("A: Rotor Length" %in% significant_terms) formula_terms <- c(formula_terms, "A_Length")
  if ("B: Rotor Width" %in% significant_terms) formula_terms <- c(formula_terms, "B_Width")
  if ("C: Paper Clip" %in% significant_terms) formula_terms <- c(formula_terms, "C_Clip")
  if ("AB: Length × Width" %in% significant_terms) formula_terms <- c(formula_terms, "A_Length
  if ("AC: Length × Clip" %in% significant_terms) formula_terms <- c(formula_terms, "A_Length:C
  if ("BC: Width × Clip" %in% significant_terms) formula_terms <- c(formula_terms, "B_Width:C_C
  if ("ABC: Length × Width × Clip" %in% significant_terms) formula_terms <- c(formula_terms, "A

  if (length(formula_terms) > 0) {
    formula_string <- paste("Time_s ~", paste(formula_terms, collapse = " + "))
    model_reduced <- lm(as.formula(formula_string), data = helicopter_coded)

    cat("**Reduced Model:**\n")
    print(summary(model_reduced))

    # Model comparison
    cat("\n**Model Comparison:**\n")
    comparison <- anova(model_reduced, model_full)
    print(comparison)
  }
} else {
  cat("### No statistically significant effects found at   = 0.05 level.\n")
```

```r
  cat("Consider using the grand mean as the best predictor.\n")
  cat("Grand mean flight time:", round(mean(helicopter_coded$Time_s), 3), "seconds\n")
}
```

```
### Reduced Model with Significant Terms Only:
**Reduced Model:**

Call:
lm(formula = as.formula(formula_string), data = helicopter_coded)

Residuals:
     Min       1Q   Median       3Q      Max
-0.30500 -0.09708 -0.00167  0.06938  0.35500

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      3.21792    0.03401  94.621  < 2e-16 ***
A_Length         0.34042    0.03401  10.010 8.80e-09 ***
B_Width         -0.26375    0.03401  -7.755 3.80e-07 ***
C_Clip          -0.20125    0.03401  -5.918 1.33e-05 ***
A_Length:C_Clip -0.11875    0.03401  -3.492   0.0026 **
B_Width:C_Clip   0.09375    0.03401   2.757   0.0130 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1666 on 18 degrees of freedom
Multiple R-squared:  0.9228,    Adjusted R-squared:  0.9014
F-statistic: 43.03 on 5 and 18 DF,  p-value: 2.157e-09


**Model Comparison:**
Analysis of Variance Table

Model 1: Time_s ~ A_Length + B_Width + C_Clip + A_Length:C_Clip + B_Width:C_Clip
Model 2: Time_s ~ A_Length * B_Width * C_Clip
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     18 0.49964
2     16 0.38340  2   0.11624 2.4255 0.1202
```

4 Conclusions

```r
cat("## Key Findings:\n\n")
```

```
## Key Findings:
```

```r
cat("1. **Design Summary:** This 2³ factorial experiment successfully evaluated the effects of
```

```
1. **Design Summary:** This 2³ factorial experiment successfully evaluated the effects of roto
```

```r
cat("2. **Effect Magnitudes:** The largest effects in order of importance are:\n")
```

```
2. **Effect Magnitudes:** The largest effects in order of importance are:
```

```r
for (i in 1:min(3, nrow(effects_summary))) {
  cat("   -", effects_summary$Effect[i], ":", sprintf("%+.3f", effects_summary$Estimate[i]), "s
}
```

```
   - A (Length) : +0.681 seconds
   - B (Width) : -0.527 seconds
   - C (Clip) : -0.403 seconds
```

```r
cat("\n")
```

```r
cat("3. **Physical Interpretation:**\n")
```

```
3. **Physical Interpretation:**
```

```r
if (abs(A_effect) > 0.1) {
  cat("   - Rotor Length:", ifelse(A_effect > 0, "Longer rotors increase flight time", "Shorte
      "by", round(abs(A_effect), 3), "seconds on average\n")
}
```

```
   - Rotor Length: Longer rotors increase flight time by 0.681 seconds on average
```

```r
if (abs(B_effect) > 0.1) {
  cat("   - Rotor Width:", ifelse(B_effect > 0, "Wider rotors increase flight time", "Narrower
      "by", round(abs(B_effect), 3), "seconds on average\n")
}
```

```
   - Rotor Width: Narrower rotors increase flight time by 0.527 seconds on average
```

```r
if (abs(C_effect) > 0.1) {
  cat("   - Paper Clips:", ifelse(C_effect > 0, "Adding clips increases flight time", "Removin
      "by", round(abs(C_effect), 3), "seconds on average\n")
}
```

```
   - Paper Clips: Removing clips increases flight time by 0.402 seconds on average
```

```r
cat("\n")
```

```r
cat("4. **Statistical Significance:** ")
```

```
4. **Statistical Significance:**
```

```r
if (nrow(significant_effects) > 0) {
  cat("The following effects were statistically significant (p < 0.05):\n")
  for (effect in significant_effects$Source) {
    cat("   -", effect, "\n")
  }
} else {
  cat("No effects were statistically significant at the   = 0.05 level.\n")
}
```

```
The following effects were statistically significant (p < 0.05):
   - A: Rotor Length
   - B: Rotor Width
   - C: Paper Clip
   - AC: Length × Clip
   - BC: Width × Clip
```

```r
cat("\n")
```

```r
cat("5. **Optimal Conditions for Maximum Flight Time:**\n")
```

```
5. **Optimal Conditions for Maximum Flight Time:**
```

```r
cat("   - Configuration: Treatment", optimal_max$Treatment, "\n")
```

```
   - Configuration: Treatment a
```

```r
cat("   - Rotor Length:", ifelse(optimal_max$A_Length == 1, "8.5 cm (Long)", "7.5 cm (Short)")
```

```
   - Rotor Length: 8.5 cm (Long)
```

```r
cat("   - Rotor Width:", ifelse(optimal_max$B_Width == 1, "5.0 cm (Wide)", "3.5 cm (Narrow)"),
```

```
   - Rotor Width: 3.5 cm (Narrow)
```

```r
cat("   - Paper Clips:", ifelse(optimal_max$C_Clip == 1, "2 clips", "0 clips"), "\n")
```

```
   - Paper Clips: 0 clips
```

```r
cat("   - Predicted flight time:", round(optimal_max$mean_response, 3), "seconds\n\n")
```

```
   - Predicted flight time: 4.147 seconds
```

```r
cat("6. **Model Adequacy Assessment:**\n")
```

```
6. **Model Adequacy Assessment:**
```

```r
cat("   - R² =", sprintf("%.3f", summary(model_full)$r.squared), "(explains", sprintf("%.1f%%"
```

```
   - R² = 0.941 (explains 94.1% of variation)
```

```r
cat("   - Adjusted R² =", sprintf("%.3f", summary(model_full)$adj.r.squared), "\n")
```

```
   - Adjusted R² = 0.915
```

```r
cat("   - Residual standard error =", sprintf("%.3f", summary(model_full)$sigma), "seconds\n")
```

```
   - Residual standard error = 0.155 seconds
```

```r
if (shapiro_test$p.value > 0.05) {
  cat("   - Normality assumption: Satisfied (Shapiro-Wilk p =", sprintf("%.3f", shapiro_test$p
} else {
  cat("   - Normality assumption: Some concern (Shapiro-Wilk p =", sprintf("%.3f", shapiro_tes
}
```

```
   - Normality assumption: Satisfied (Shapiro-Wilk p = 0.112 )
```

```r
cat("\n**Practical Recommendations:**\n")
```

```

**Practical Recommendations:**
```

```r
cat("- This classic DOE demonstrates fundamental principles of factorial experimentation\n")
```

```
- This classic DOE demonstrates fundamental principles of factorial experimentation
```

```r
cat("- The design efficiently evaluates all factor combinations with proper replication\n")
```

```
- The design efficiently evaluates all factor combinations with proper replication
```

```r
cat("- Results provide clear guidance for optimizing paper helicopter performance\n")
```

```
- Results provide clear guidance for optimizing paper helicopter performance
```

```r
cat("- Methodology is suitable for educational purposes and real process improvement\n")
```

```
- Methodology is suitable for educational purposes and real process improvement
```