ST695\_SPD\_Analysis

**Instantiate Libraries**

library(lmerTest) #For REML Analysis

Loading required package: lme4

Loading required package: Matrix

Attaching package: 'lmerTest'

The following object is masked from 'package:lme4':  
  
 lmer

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

library(afex) #For ANOVA Analysis

\*\*\*\*\*\*\*\*\*\*\*\*  
Welcome to afex. For support visit: http://afex.singmann.science/

- Functions for ANOVAs: aov\_car(), aov\_ez(), and aov\_4()  
- Methods for calculating p-values with mixed(): 'S', 'KR', 'LRT', and 'PB'  
- 'afex\_aov' and 'mixed' objects can be passed to emmeans() for follow-up tests  
- Get and set global package options with: afex\_options()  
- Set sum-to-zero contrasts globally: set\_sum\_contrasts()  
- For example analyses see: browseVignettes("afex")  
\*\*\*\*\*\*\*\*\*\*\*\*

Attaching package: 'afex'

The following object is masked from 'package:lme4':  
  
 lmer

library(FrF2) #For QQ Plots

Loading required package: DoE.base

Loading required package: grid

Loading required package: conf.design

Attaching package: 'conf.design'

The following object is masked from 'package:lme4':  
  
 factorize

Registered S3 method overwritten by 'DoE.base':  
 method from   
 factorize.factor conf.design

Attaching package: 'DoE.base'

The following objects are masked from 'package:stats':  
  
 aov, lm

The following object is masked from 'package:graphics':  
  
 plot.design

The following object is masked from 'package:base':  
  
 lengths

library(BsMD) #For Lenth Plots

Attaching package: 'BsMD'

The following object is masked from 'package:FrF2':  
  
 DanielPlot

**Import SP PLA Data Set**

Ys\_data <- read.csv("Data/YsData.csv")  
Ys\_data <- Ys\_data[,7:13]  
Ys\_data$WP=as.factor(Ys\_data$WP)  
cat("SP PLA Data Set\n")

SP PLA Data Set

print(Ys\_data)

T P S D R WP Y\_S  
1 -1 -1 -1 -1 -1 1 445.5  
2 -1 -1 -1 -1 1 1 437.1  
3 -1 -1 -1 1 -1 1 198.0  
4 -1 -1 -1 1 1 1 151.8  
5 -1 -1 1 -1 -1 2 147.0  
6 -1 -1 1 -1 1 2 88.5  
7 -1 -1 1 1 -1 2 90.0  
8 -1 -1 1 1 1 2 74.2  
9 -1 1 -1 -1 -1 3 890.0  
10 -1 1 -1 -1 1 3 864.9  
11 -1 1 -1 1 -1 3 317.2  
12 -1 1 -1 1 1 3 249.3  
13 -1 1 1 -1 -1 4 299.2  
14 -1 1 1 -1 1 4 292.9  
15 -1 1 1 1 -1 4 164.2  
16 -1 1 1 1 1 4 125.5  
17 1 -1 -1 -1 -1 5 434.2  
18 1 -1 -1 1 -1 5 231.0  
19 1 -1 -1 1 1 5 113.4  
20 1 -1 1 -1 -1 6 144.9  
21 1 -1 1 -1 1 6 67.5  
22 1 -1 1 1 -1 6 73.3  
23 1 -1 1 1 1 6 55.8  
24 1 1 -1 -1 -1 7 882.6  
25 1 1 -1 -1 1 7 891.3  
26 1 1 -1 1 -1 7 334.5  
27 1 1 -1 1 1 7 207.9  
28 1 1 1 -1 -1 8 277.2  
29 1 1 1 -1 1 8 274.8  
30 1 1 1 1 -1 8 168.6  
31 1 1 1 1 1 8 120.3

**Method 1 (Ordinary Least Squares)**

WP\_factors <- colnames(Ys\_data[1:3])  
SP\_factors <- colnames(Ys\_data[4:5])  
linear\_terms <- c(WP\_factors, SP\_factors)  
# Model with all main effects and 2-way interactions  
formula\_1 <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2"))  
Method\_1 <- summary(lm(formula\_1, data = Ys\_data))  
Method\_1

Call:  
lm.default(formula = formula\_1, data = Ys\_data)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-62.225 -26.034 -0.719 27.928 58.100   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 299.0906 9.0640 32.998 2.03e-15 \*\*\*  
T -3.1156 9.0640 -0.344 0.735815   
P 98.4344 9.0640 10.860 1.67e-08 \*\*\*  
S -145.0969 9.0640 -16.008 7.72e-11 \*\*\*  
D -131.9031 9.0640 -14.552 2.97e-10 \*\*\*  
R -19.4969 9.0640 -2.151 0.048181 \*   
T:P 0.2406 9.0640 0.027 0.979171   
T:S -3.0781 9.0640 -0.340 0.738867   
T:D -0.9719 9.0640 -0.107 0.916033   
T:R -2.8156 9.0640 -0.311 0.760350   
P:S -37.0906 9.0640 -4.092 0.000962 \*\*\*  
P:D -54.6844 9.0640 -6.033 2.29e-05 \*\*\*  
P:R 0.3344 9.0640 0.037 0.971059   
S:D 86.8969 9.0640 9.587 8.68e-08 \*\*\*  
S:R 2.9406 9.0640 0.324 0.750093   
D:R -10.4156 9.0640 -1.149 0.268500   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 49.74 on 15 degrees of freedom  
Multiple R-squared: 0.981, Adjusted R-squared: 0.9619   
F-statistic: 51.53 on 15 and 15 DF, p-value: 3.718e-10

Residual Standard Error: 49.74. Error variance estimate: 49.74^2 = 2474.8

The error variance (mean squared error, MSE) is 2474.8 in your model. This value is the estimate of the residual variance from your standard linear model.

**Method 2A (Restricted Maximum Likelihood using Satterwaithe)**

# Model with all main effects, 2-way interactions, and random WP effect  
formula\_2A <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2 + (1|WP)"))  
Method\_2A <- summary(lmer(formula\_2A, data = Ys\_data))

boundary (singular) fit: see help('isSingular')

Method\_2A

Linear mixed model fit by REML. t-tests use Satterthwaite's method [  
lmerModLmerTest]  
Formula: formula\_2A  
 Data: Ys\_data  
  
REML criterion at convergence: 214.5  
  
Scaled residuals:   
 Min 1Q Median 3Q Max   
-1.25093 -0.52338 -0.01445 0.56145 1.16801   
  
Random effects:  
 Groups Name Variance Std.Dev.  
 WP (Intercept) 0 0.00   
 Residual 2474 49.74   
Number of obs: 31, groups: WP, 8  
  
Fixed effects:  
 Estimate Std. Error df t value Pr(>|t|)   
(Intercept) 299.0906 9.0640 15.0000 32.998 2.03e-15 \*\*\*  
T -3.1156 9.0640 15.0000 -0.344 0.735815   
P 98.4344 9.0640 15.0000 10.860 1.67e-08 \*\*\*  
S -145.0969 9.0640 15.0000 -16.008 7.72e-11 \*\*\*  
D -131.9031 9.0640 15.0000 -14.552 2.97e-10 \*\*\*  
R -19.4969 9.0640 15.0000 -2.151 0.048181 \*   
T:P 0.2406 9.0640 15.0000 0.027 0.979171   
T:S -3.0781 9.0640 15.0000 -0.340 0.738867   
T:D -0.9719 9.0640 15.0000 -0.107 0.916033   
T:R -2.8156 9.0640 15.0000 -0.311 0.760350   
P:S -37.0906 9.0640 15.0000 -4.092 0.000962 \*\*\*  
P:D -54.6844 9.0640 15.0000 -6.033 2.29e-05 \*\*\*  
P:R 0.3344 9.0640 15.0000 0.037 0.971059   
S:D 86.8969 9.0640 15.0000 9.587 8.68e-08 \*\*\*  
S:R 2.9406 9.0640 15.0000 0.324 0.750093   
D:R -10.4156 9.0640 15.0000 -1.149 0.268500   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Correlation matrix not shown by default, as p = 16 > 12.  
Use print(x, correlation=TRUE) or  
 vcov(x) if you need it

optimizer (nloptwrap) convergence code: 0 (OK)  
boundary (singular) fit: see help('isSingular')

**Method 2B (Restricted Maximum Likelihood using Kenward Roger with 2-Way Interactions)**

# Model with all main effects, 2-way interactions, and random WP effect  
formula\_2B <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2+ (1|WP)"))  
# Fit model with Kenward-Roger method  
model\_2B <- lmer(formula\_2B, data = Ys\_data, REML = TRUE)

boundary (singular) fit: see help('isSingular')

Method\_2B <- summary(model\_2B, ddf = "Kenward-Roger")  
Method\_2B

Linear mixed model fit by REML. t-tests use Kenward-Roger's method [  
lmerModLmerTest]  
Formula: formula\_2B  
 Data: Ys\_data  
  
REML criterion at convergence: 214.5  
  
Scaled residuals:   
 Min 1Q Median 3Q Max   
-1.25093 -0.52338 -0.01445 0.56145 1.16801   
  
Random effects:  
 Groups Name Variance Std.Dev.  
 WP (Intercept) 0 0.00   
 Residual 2474 49.74   
Number of obs: 31, groups: WP, 8  
  
Fixed effects:  
 Estimate Std. Error df t value Pr(>|t|)   
(Intercept) 299.0906 9.1398 0.9845 32.724 0.0205 \*   
T -3.1156 9.1398 0.9845 -0.341 0.7915   
P 98.4344 9.1398 0.9845 10.770 0.0610 .   
S -145.0969 9.1398 0.9845 -15.875 0.0417 \*   
D -131.9031 9.1398 14.1074 -14.432 7.66e-10 \*\*\*  
R -19.4969 9.1398 14.1074 -2.133 0.0509 .   
T:P 0.2406 9.1398 0.9845 0.026 0.9833   
T:S -3.0781 9.1398 0.9845 -0.337 0.7939   
T:D -0.9719 9.1398 14.1074 -0.106 0.9168   
T:R -2.8156 9.1398 14.1074 -0.308 0.7625   
P:S -37.0906 9.1398 0.9845 -4.058 0.1568   
P:D -54.6844 9.1398 14.1074 -5.983 3.25e-05 \*\*\*  
P:R 0.3344 9.1398 14.1074 0.037 0.9713   
S:D 86.8969 9.1398 14.1074 9.507 1.63e-07 \*\*\*  
S:R 2.9406 9.1398 14.1074 0.322 0.7524   
D:R -10.4156 9.1398 14.1074 -1.140 0.2734   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Correlation matrix not shown by default, as p = 16 > 12.  
Use print(x, correlation=TRUE) or  
 vcov(x) if you need it

optimizer (nloptwrap) convergence code: 0 (OK)  
boundary (singular) fit: see help('isSingular')

**Method 2C (Restricted Maximum Likelihood using Kenward Roger with 3-Way Interactions)**

# Model with all main effects, 3-way interactions without T:P:S, and random WP effect  
formula\_2C <- Y\_S ~ (T + P + S + D + R)^3 - T:P:S + (1|WP)  
# Fit model with Kenward-Roger method  
model\_2C <- lmer(formula\_2C, data = Ys\_data, REML = TRUE)

boundary (singular) fit: see help('isSingular')

Method\_2C <- summary(model\_2C, ddf = "Kenward-Roger")  
Method\_2C

Linear mixed model fit by REML. t-tests use Kenward-Roger's method [  
lmerModLmerTest]  
Formula: formula\_2C  
 Data: Ys\_data  
  
REML criterion at convergence: 137.1  
  
Scaled residuals:   
 Min 1Q Median 3Q Max   
-0.9591 -0.2555 -0.1015 0.4184 0.7610   
  
Random effects:  
 Groups Name Variance Std.Dev.  
 WP (Intercept) 0 0.00   
 Residual 336 18.33   
Number of obs: 31, groups: WP, 8  
  
Fixed effects:  
 Estimate Std. Error df t value Pr(>|t|)   
(Intercept) 297.5107 3.6330 0.9213 81.890 0.010852 \*   
T -4.6955 3.6330 0.9213 -1.292 0.432538   
P 100.0143 3.6330 0.9213 27.529 0.029619 \*   
S -143.5170 3.6330 0.9213 -39.503 0.021240 \*   
D -130.3232 3.6330 5.2033 -35.872 1.96e-07 \*\*\*  
R -21.0768 3.6330 5.2033 -5.801 0.001877 \*\*   
T:P 1.8205 3.6330 0.9213 0.501 0.709835   
T:S -1.4982 3.6330 0.9213 -0.412 0.755503   
T:D 0.6080 3.6330 5.2033 0.167 0.873396   
T:R -4.3955 3.6330 5.2033 -1.210 0.278407   
P:S -38.6705 3.6330 0.9213 -10.644 0.070936 .   
P:D -56.2643 3.6330 5.2033 -15.487 1.49e-05 \*\*\*  
P:R 1.9143 3.6330 5.2033 0.527 0.619943   
S:D 85.3170 3.6330 5.2033 23.484 1.76e-06 \*\*\*  
S:R 4.5205 3.6330 5.2033 1.244 0.266509   
D:R -8.8357 3.6330 5.2033 -2.432 0.057291 .   
T:P:D -0.8455 3.6330 5.2033 -0.233 0.824847   
T:P:R 2.4830 3.6330 5.2033 0.683 0.523561   
T:S:D 1.0982 3.6330 5.2033 0.302 0.774147   
T:S:R 2.7518 3.6330 5.2033 0.757 0.481666   
T:D:R -4.4420 3.6330 5.2033 -1.223 0.273934   
P:S:D 30.5830 3.6330 5.2033 8.418 0.000319 \*\*\*  
P:S:R 2.6795 3.6330 5.2033 0.738 0.492702   
P:D:R -7.1893 3.6330 5.2033 -1.979 0.102495   
S:D:R 10.3545 3.6330 5.2033 2.850 0.034231 \*   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Correlation matrix not shown by default, as p = 25 > 12.  
Use print(x, correlation=TRUE) or  
 vcov(x) if you need it

optimizer (nloptwrap) convergence code: 0 (OK)  
boundary (singular) fit: see help('isSingular')

**Method 4A ANOVA using Base R**

# Model with all main effects, 2-way interactions,  
#and split-plot error structure  
formula\_4A <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2 + Error(WP)"))  
  
Method\_4A <- summary(aov(formula\_4A, data = Ys\_data))  
Method\_4A

Error: WP  
 Df Sum Sq Mean Sq  
T 1 2251 2251  
P 1 356982 356982  
S 1 613959 613959  
D 1 23273 23273  
T:P 1 2808 2808  
T:S 1 9591 9591  
P:S 1 23020 23020  
  
Error: Within  
 Df Sum Sq Mean Sq F value Pr(>F)   
D 1 527529 527529 199.115 1.14e-09 \*\*\*  
R 1 13710 13710 5.175 0.0392 \*   
T:D 1 16 16 0.006 0.9388   
T:R 1 685 685 0.259 0.6190   
P:D 1 107948 107948 40.745 1.70e-05 \*\*\*  
P:R 1 927 927 0.350 0.5635   
S:D 1 226165 226165 85.366 2.47e-07 \*\*\*  
S:R 1 377 377 0.142 0.7116   
D:R 1 3290 3290 1.242 0.2839   
Residuals 14 37091 2649   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

**Method 4B ANOVA using Letsinger**

# Proposed ANOVA method: saturated model with all 2-way + T:P:S interaction  
  
library(readr)  
  
data <- read\_csv("Data/PLA\_Data.csv")

Rows: 31 Columns: 7  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
dbl (7): T, P, S, D, R, WP, Denier  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# Saturated model: all 2-way interactions + T:P:S  
saturated\_formula <- "Denier ~ (T + P + S + D + R)^2 + T:P:S"  
saturated\_model <- lm(saturated\_formula, data = data)  
  
# Reduced model: T + P + S + D + R + P:S + P:D + S:D  
reduced\_formula <- "Denier ~ T + P + S + D + R + P:S + P:D + S:D"  
reduced\_model <- lm(reduced\_formula, data = data)  
  
# Calculate sums of squares and degrees of freedom  
SS\_saturated <- sum(residuals(saturated\_model)^2)  
SS\_reduced <- sum(residuals(reduced\_model)^2)  
df\_saturated <- saturated\_model$df.residual  
df\_reduced <- reduced\_model$df.residual  
  
MSsp <- SS\_saturated / df\_saturated  
MSdif <- (SS\_reduced - SS\_saturated) / (df\_reduced - df\_saturated)  
m <- 4  
sigma2\_wp <- (MSdif - MSsp) / m  
sigma2\_sp <- MSsp  
  
cat("=== Proposed Method Results (Saturated model: 2-way + T:P:S) ===\n")

=== Proposed Method Results (Saturated model: 2-way + T:P:S) ===

cat("Whole-plot error variance (σ²wp): ", round(sigma2\_wp, 6), "\n")

Whole-plot error variance (σ²wp): -528.8143

cat("Sub-plot error variance (σ²sp): ", round(sigma2\_sp, 6), "\n")

Sub-plot error variance (σ²sp): 2649.362

cat("Total error variance: ", round(sigma2\_wp + sigma2\_sp, 6), "\n")

Total error variance: 2120.548

if(!is.na(sigma2\_wp) && sigma2\_wp >= 0) {  
 prop\_wp <- sigma2\_wp / (sigma2\_wp + sigma2\_sp) \* 100  
 prop\_sp <- sigma2\_sp / (sigma2\_wp + sigma2\_sp) \* 100  
 cat("Whole-plot error proportion: ", round(prop\_wp, 1), "%\n")  
 cat("Sub-plot error proportion: ", round(prop\_sp, 1), "%\n")  
} else {  
 cat("\*\*\* WARNING: Negative whole-plot variance estimate! \*\*\*\n")  
 if(!is.na(sigma2\_wp)) {  
 prop\_wp <- sigma2\_wp / (sigma2\_wp + sigma2\_sp) \* 100  
 prop\_sp <- sigma2\_sp / (sigma2\_wp + sigma2\_sp) \* 100  
 cat("Whole-plot error proportion: ", round(prop\_wp, 1), "%\n")  
 cat("Sub-plot error proportion: ", round(prop\_sp, 1), "%\n")  
 }  
}

\*\*\* WARNING: Negative whole-plot variance estimate! \*\*\*  
Whole-plot error proportion: -24.9 %  
Sub-plot error proportion: 124.9 %

cat("\n=== Reduced Model Fixed Effects ===\n")

=== Reduced Model Fixed Effects ===

print(summary(reduced\_model)$coefficients)

Estimate Std. Error t value Pr(>|t|)  
(Intercept) 298.736957 7.830079 38.1524813 1.349787e-21  
T -3.469293 7.830079 -0.4430726 6.620395e-01  
P 98.788043 7.830079 12.6164805 1.510049e-11  
S -144.743207 7.830079 -18.4855350 6.862177e-15  
D -131.549457 7.830079 -16.8005266 4.916830e-14  
R -19.850543 7.830079 -2.5351650 1.885784e-02  
P:S -37.444293 7.830079 -4.7821091 8.941094e-05  
P:D -55.038043 7.830079 -7.0290531 4.715092e-07  
S:D 86.543207 7.830079 11.0526602 1.896102e-10

# Continue with any further analysis or reporting as needed

**Method 5A Additional Replicates**

# Function to simulate whole-plot and sub-plot replicates  
WP\_SP\_replicates <- function(data, response = "Denier", r = 3, sd = 5, seed = 123) {  
 set.seed(seed)  
 # Repeat each row r times  
 data\_rep <- data[rep(1:nrow(data), each = r), ]  
 # Add replicate identifier  
 data\_rep$RPL <- rep(1:r, times = nrow(data))  
 # Add random noise to the response variable  
 data\_rep[[response]] <- data\_rep[[response]] + rnorm(nrow(data\_rep), mean = 0, sd = sd)  
 rownames(data\_rep) <- NULL  
 return(data\_rep)  
}  
  
# Function to calculate sub-plot variance (σ²\_sp)  
calculate\_subplot\_variance <- function(data, response, wp\_col, sp\_col) {  
 # Split data by whole-plot column  
 wp\_groups <- split(data, data[[wp\_col]])  
   
 # Calculate variance within each whole-plot group  
 sp\_variances <- sapply(wp\_groups, function(group) {  
 sum((group[[response]] - mean(group[[response]]))^2) / (nrow(group) - 1)  
 })  
   
 # Return the average sub-plot variance  
 sigma2\_sp <- mean(sp\_variances)  
 return(sigma2\_sp)  
}  
  
# Function to calculate total variance (σ²\_tot)  
calculate\_total\_variance <- function(data, response, replicate\_col) {  
 # Split data by replicate column  
 replicate\_groups <- split(data, data[[replicate\_col]])  
   
 # Calculate mean for each replicate group  
 replicate\_means <- sapply(replicate\_groups, function(group) mean(group[[response]]))  
   
 # Calculate variance of replicate means  
 sigma2\_tot <- var(replicate\_means)  
 return(sigma2\_tot)  
}  
  
# Function to calculate whole-plot variance (σ²\_wp)  
calculate\_wholeplot\_variance <- function(sigma2\_tot, sigma2\_sp) {  
 sigma2\_wp <- sigma2\_tot - sigma2\_sp  
 return(sigma2\_wp)  
}  
  
# Main analysis function for Method 5A  
method5A\_analysis <- function(data, response = "Denier", wp\_col = "WP", sp\_col = "RPL", replicate\_col = "RPL") {  
 # Calculate sub-plot variance  
 sigma2\_sp <- calculate\_subplot\_variance(data, response, wp\_col, sp\_col)  
   
 # Calculate total variance  
 sigma2\_tot <- calculate\_total\_variance(data, response, replicate\_col)  
   
 # Calculate whole-plot variance  
 sigma2\_wp <- calculate\_wholeplot\_variance(sigma2\_tot, sigma2\_sp)  
   
 # Return results  
 return(list(  
 sigma2\_sp = sigma2\_sp,  
 sigma2\_tot = sigma2\_tot,  
 sigma2\_wp = sigma2\_wp  
 ))  
}  
  
# Example usage  
  
  
# Simulate replicates  
replicated\_data <- WP\_SP\_replicates(data = Ys\_data, response = "Denier", r = 3, sd = 5, seed = 123)  
  
# Perform Method 5A analysis  
results <- method5A\_analysis(replicated\_data, response = "Denier", wp\_col = "WP", sp\_col = "RPL", replicate\_col = "RPL")  
  
# Print results  
print(results)

**Method 5B Hierarchical Design**

# Function to simulate hierarchical replicates for Method 5B  
simulate\_hierarchical\_data <- function(data, response = "Denier", wp\_col = "WP",   
 num\_wp\_replicates = 3, num\_sp\_replicates = 2,   
 sd\_wp = 10, sd\_sp = 5, seed = 123) {  
 set.seed(seed)  
   
 # Create whole-plot replicates  
 wp\_replicates <- data[rep(1:nrow(data), each = num\_wp\_replicates), ]  
 wp\_replicates$WP\_Rep <- rep(1:num\_wp\_replicates, times = nrow(data))  
   
 # Add random noise for whole-plot variability  
 wp\_replicates[[response]] <- wp\_replicates[[response]] +   
 rnorm(nrow(wp\_replicates), mean = 0, sd = sd\_wp)  
   
 # Create sub-plot replicates within each whole-plot replicate  
 sp\_replicates <- wp\_replicates[rep(1:nrow(wp\_replicates), each = num\_sp\_replicates), ]  
 sp\_replicates$SP\_Rep <- rep(1:num\_sp\_replicates, times = nrow(wp\_replicates))  
   
 # Add random noise for sub-plot variability  
 sp\_replicates[[response]] <- sp\_replicates[[response]] +   
 rnorm(nrow(sp\_replicates), mean = 0, sd = sd\_sp)  
   
 # Reset row names and return the simulated dataset  
 rownames(sp\_replicates) <- NULL  
 return(sp\_replicates)  
}  
  
# Example usage  
  
# Simulate hierarchical data  
simulated\_data <- simulate\_hierarchical\_data(  
 data = Ys\_data,   
 response = "Denier",   
 wp\_col = "WP",   
 num\_wp\_replicates = 3, # Number of whole-plot replicates  
 num\_sp\_replicates = 2, # Number of sub-plot replicates  
 sd\_wp = 10, # Standard deviation for whole-plot noise  
 sd\_sp = 5, # Standard deviation for sub-plot noise  
 seed = 123 # Random seed for reproducibility  
)  
  
# View the first few rows of the simulated dataset  
head(simulated\_data)  
  
  
# Function to perform Method 5B analysis  
method5B\_analysis <- function(data, response = "Denier", wp\_col = "WP", wp\_rep\_col = "WP\_Rep", sp\_rep\_col = "SP\_Rep") {  
 # Step 1: Calculate the total mean  
 grand\_mean <- mean(data[[response]])  
   
 # Step 2: Calculate the whole-plot mean squares (MS\_wp)  
 wp\_means <- aggregate(data[[response]], by = list(data[[wp\_col]], data[[wp\_rep\_col]]), FUN = mean)  
 colnames(wp\_means) <- c(wp\_col, wp\_rep\_col, "WP\_Mean")  
 wp\_ms <- sum((wp\_means$WP\_Mean - grand\_mean)^2) / (nrow(wp\_means) - 1)  
   
 # Step 3: Calculate the sub-plot mean squares (MS\_sp)  
 sp\_means <- aggregate(data[[response]], by = list(data[[sp\_rep\_col]]), FUN = mean)  
 colnames(sp\_means) <- c(sp\_rep\_col, "SP\_Mean")  
 sp\_ms <- sum((sp\_means$SP\_Mean - grand\_mean)^2) / (nrow(sp\_means) - 1)  
   
 # Step 4: Calculate the residual mean squares (MS\_res)  
 residuals <- data[[response]] - ave(data[[response]], data[[wp\_col]], data[[wp\_rep\_col]], data[[sp\_rep\_col]], FUN = mean)  
 ms\_res <- sum(residuals^2) / (nrow(data) - nrow(wp\_means) - nrow(sp\_means) + 1)  
   
 # Step 5: Estimate variances  
 sigma2\_wp <- (wp\_ms - sp\_ms) / (length(unique(data[[wp\_rep\_col]])) \* length(unique(data[[sp\_rep\_col]])))  
 sigma2\_sp <- (sp\_ms - ms\_res) / length(unique(data[[sp\_rep\_col]]))  
 sigma2\_res <- ms\_res  
   
 # Return results  
 return(list(  
 sigma2\_wp = sigma2\_wp,  
 sigma2\_sp = sigma2\_sp,  
 sigma2\_res = sigma2\_res  
 ))  
}  
  
# Example usage  
  
# Perform Method 5B analysis  
results <- method5B\_analysis(simulated\_data, response = "Denier", wp\_col = "WP", wp\_rep\_col = "WP\_Rep", sp\_rep\_col = "SP\_Rep")  
  
# Print results  
print(results)

**Method 6 QQ Plot Method for S Model**

# Model with all main effects and 2-way interactions  
formula\_6S <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2"))  
model\_6S <- lm(formula\_6S, data = Ys\_data)   
effects <- coef(model\_6S)[-1] #Remove intercept  
# Plot 1 - QQ Plot of SP Effects  
SP\_effects\_names <- c(  
 "D", # SP main effect  
 "R", # SP main effect  
 "D:R", # SP interaction  
 "T:D", # SP x WP  
 "T:R", # SP x WP  
 "P:D", # SP x WP  
 "P:R", # SP x WP  
 "S:D", # SP x WP  
 "S:R" # SP x WP  
)  
SP\_effects <- effects[SP\_effects\_names]  
#Plot 2 - QQ Plot of WP Effects  
WP\_effects\_names <- c(  
 "T", # WP main effect  
 "S", # WP main effect  
 "P", # WP main effect  
 "T:S", # WP interaction  
 "T:P", # WP interaction  
 "P:S" # WP interaction  
)  
WP\_effects <- effects[WP\_effects\_names]

**Method 7 Lenth Method for S Model**

# Model with all main effects and 2-way interactions  
formula\_7S <- as.formula(paste("Y\_S ~ (",   
 paste(linear\_terms, collapse = " + "),")^2"))  
model\_7S <- lm(formula\_7S, data = Ys\_data) # Use lm()  
effects <- coef(model\_7S)[-1] # Remove intercept  
# Plot 1 - Lenth Plot of SP Effects  
SP\_effects\_names <- c(  
 "D", # SP main effect  
 "R", # SP main effect  
 "D:R", # SP interaction  
 "T:D", # SP x WP  
 "T:R", # SP x WP  
 "P:D", # SP x WP  
 "P:R", # SP x WP  
 "S:D", # SP x WP  
 "S:R" # SP x WP  
)  
SP\_effects <- effects[SP\_effects\_names]  
#Plot 2 - Lenth Plot of WP Effects  
WP\_effects\_names <- c(  
 "T", # WP main effect  
 "S", # WP main effect  
 "P", # WP main effect  
 "T:S", # WP interaction  
 "T:P", # WP interaction  
 "P:S" # WP interaction  
)  
WP\_effects <- effects[WP\_effects\_names]

\*\*2 x 2 Plot of QQ and Lenth Plots for S Model

pdf("2x2\_plot.pdf", width = 8, height = 8)  
par(mfrow = c(2, 2))  
  
# plot1 - WP QQ Plot  
halfnormal(  
 WP\_effects,  
 main = "Half-Normal Plot for S Model :\nWP Factors + 2-way Interactions",  
 labs = WP\_effects\_names,  
 alpha = 1  
)

simulated critical values not available for all requests, used conservative ones

# plot2 - WP Lenth Plot  
LenthPlot(WP\_effects, main = "Lenth Plot for S Model: WP Effects")

alpha PSE ME SME   
 0.050000 4.645312 19.987167 50.026631

# plot3 - SP QQ Plot  
halfnormal(  
 SP\_effects,  
 main = "Half-Normal Plot for S Model :\nSP Factors + 2-way Interactions +\n2-way Interactions with WP Factors",  
 labs = SP\_effects\_names,  
 alpha = 1  
)

simulated critical values not available for all requests, used conservative ones

# plot4 - SP Lenth Plot  
LenthPlot(SP\_effects, main = "Lenth Plot for S Model: SP Effects")

alpha PSE ME SME   
 0.050000 4.317187 13.739217 30.773902

dev.off()

png   
 2

**Import WP Factor PLA Data**

Yw\_data <- read.csv("Data/YwData.csv")  
Yw\_data <- Yw\_data[,14:17]  
cat("WP PLA Data Set\n")

WP PLA Data Set

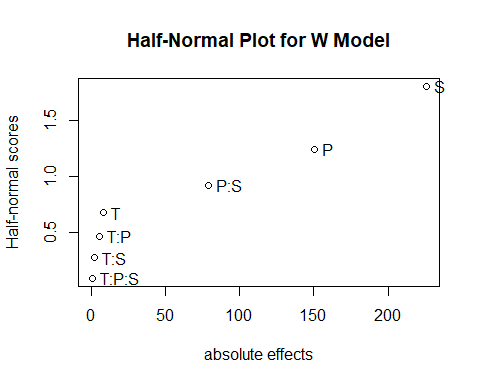
print(Yw\_data)

T P S Y\_W  
1 -1 -1 -1 445.5  
2 -1 -1 1 145.8  
3 -1 1 -1 914.7  
4 -1 1 1 302.1  
5 1 -1 -1 434.2  
6 1 -1 1 145.8  
7 1 1 -1 882.6  
8 1 1 1 277.2

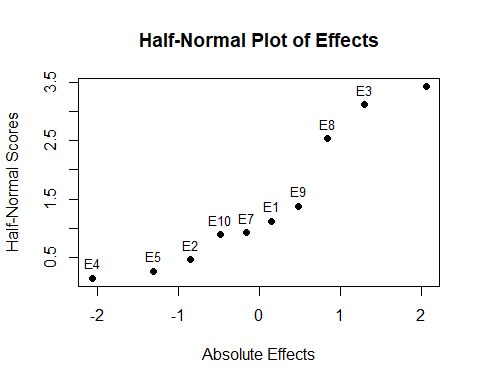
**Method 6 QQ Plot Method for W Model**

# Model for WP factors (main effects, 2- and 3-way interactions)  
formula\_6W <- as.formula(paste("Y\_W ~ (",  
 paste(WP\_factors, collapse = " + "),")^3"))  
model\_6W <- lm(formula\_6W, data = Yw\_data)  
effects\_6W <- coef(model\_6W)[-1]  
effect\_names\_6W <- names(coef(model\_6W))[-1]  
halfnormal(  
 effects\_6W,  
 main = "Half-Normal Plot for W Model",  
 labs = effect\_names\_6W,  
 alpha = 1  
)

simulated critical values not available for all requests, used conservative ones

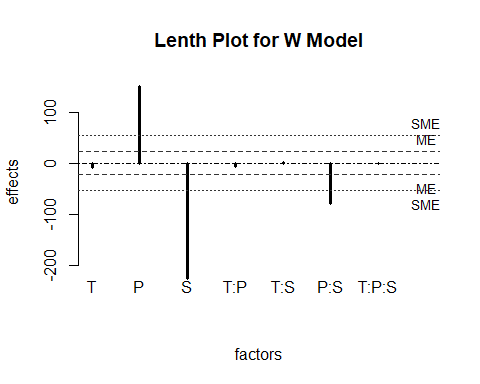


# Example: Half-Normal Plot with labels and reference line  
  
set.seed(123)  
effects <- rnorm(10, mean = 0, sd = 2) # 10 random effects  
effect\_labels <- paste0("E", 1:10)  
  
# Sort absolute effects and keep labels  
order\_idx <- order(abs(effects))  
abs\_effects <- abs(effects)[order\_idx]  
labels\_sorted <- effect\_labels[order\_idx]  
  
n <- length(effects)  
expected <- qnorm((1:n - 0.5) / n) \* sqrt(pi/2)  
  
plot(expected, abs\_effects,  
 main = "Half-Normal Plot of Effects",  
 xlab = "Absolute Effects",  
 ylab = "Half-Normal Scores",  
 pch = 19)  
  
# Add labels to points  
text(expected, abs\_effects, labels = labels\_sorted, pos = 3, cex = 0.8)



**Method 7 Lenth Method for W Model**

# Model for WP factors (main effects, 2- and 3-way interactions)  
formula\_7W <- as.formula(paste("Y\_W ~ (",  
 paste(WP\_factors, collapse = " + "),")^3"))  
model\_7W <- lm(formula\_7W, data = Yw\_data)  
effects\_7W <- coef(model\_7W)[-1] # Remove intercept  
LenthPlot(effects\_7W, main = "Lenth Plot for W Model")



alpha PSE ME SME   
 0.05000 6.01875 22.65532 54.21875

**Method 8A Sequential Split Plot Method with 2-way Interactions**

#-----------------------------------------------------------  
#S Model - SP Factors (WP main effects, SP main effects,   
#2-way interactions within WP and SP and 2-way interactions   
# with WP Factors)  
formula\_S <- as.formula(paste("Y\_S ~ (", paste(linear\_terms,   
 collapse = " + "),")^2"))  
#Create model matrix for S Model  
Y\_S <- Ys\_data$Y\_S  
X\_S <- model.matrix(formula\_S, data = Ys\_data)  
#-----------------------------------------------------------  
#W Model - WP factors (main WP effects, 2-way interactions)  
formula\_W <- as.formula(paste("Y\_W ~ (",   
 paste(WP\_factors, collapse = " + "),")^2"))  
  
#Create model matrix for W Model  
X\_W <- model.matrix(formula\_W, data = Yw\_data)  
# Replicate each row 4 times and drop row 20.  
X\_W <- X\_W[rep(1:nrow(X\_W), each = 4), ][-20, ]  
  
Y\_W <- rep(Yw\_data$Y\_W, each = 4)[-20]  
df\_W <- data.frame(Y\_W, X\_W)  
  
cat("Linear Model Fit for W Model (OLS)", "\n")

Linear Model Fit for W Model (OLS)

W\_Model <- summary(lm(formula\_W, data = Yw\_data))  
W\_Model

Call:  
lm.default(formula = formula\_W, data = Yw\_data)  
  
Residuals:  
 1 2 3 4 5 6 7 8   
 0.5125 -0.5125 -0.5125 0.5125 -0.5125 0.5125 0.5125 -0.5125   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 443.4875 0.5125 865.341 0.000736 \*\*\*  
T -8.5375 0.5125 -16.659 0.038170 \*   
P 150.6625 0.5125 293.976 0.002166 \*\*   
S -225.7625 0.5125 -440.512 0.001445 \*\*   
T:P -5.7125 0.5125 -11.146 0.056962 .   
T:S 2.3125 0.5125 4.512 0.138845   
P:S -78.7375 0.5125 -153.634 0.004144 \*\*   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 1.45 on 1 degrees of freedom  
Multiple R-squared: 1, Adjusted R-squared: 1   
F-statistic: 5.075e+04 on 6 and 1 DF, p-value: 0.003398

#------------------------------------------------------------  
#S-W Model by Satterwaithe  
#Create model matrix for S-W Model  
X\_SW <- X\_S[, SP\_effects\_names]  
Y\_SW <- Y\_S - Y\_W  
df\_SW <- data.frame(Y\_SW, X\_SW, check.names = FALSE)  
 formula\_SW <- as.formula(  
 "Y\_SW ~ D + R + `D:R` + `T:D` + `T:R` + `P:D` + `P:R` + `S:D` + `S:R`")  
SW\_Model <-lm(formula\_SW, data = df\_SW)  
  
cat("\n")

cat("Linear Model Fit for SW1 Model (OLS)", "\n")

Linear Model Fit for SW1 Model (OLS)

SW\_Model\_summary <- summary(SW\_Model)  
SW\_Model\_summary

Call:  
lm.default(formula = formula\_SW, data = df\_SW)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-220.51 -68.83 37.78 86.55 152.59   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -145.2415 24.6531 -5.891 7.56e-06 \*\*\*  
D -131.0585 24.6531 -5.316 2.85e-05 \*\*\*  
R -20.3415 24.6531 -0.825 0.41858   
`D:R` -9.5710 24.6531 -0.388 0.70176   
`T:D` -0.1273 24.6531 -0.005 0.99593   
`T:R` -3.6602 24.6531 -0.148 0.88339   
`P:D` -55.5290 24.6531 -2.252 0.03512 \*   
`P:R` 1.1790 24.6531 0.048 0.96231   
`S:D` 86.0523 24.6531 3.491 0.00218 \*\*   
`S:R` 3.7852 24.6531 0.154 0.87944   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 136.4 on 21 degrees of freedom  
Multiple R-squared: 0.6929, Adjusted R-squared: 0.5613   
F-statistic: 5.265 on 9 and 21 DF, p-value: 0.0008274

#-------------------------------------------------------------

**SSPD Method Model Adequacy Test for Method 8A**

# Perform F-test using ANOVA  
# model1 is the reduced model (SW Model)  
# model 2 is the full model (S Model)  
  
# SW Model Comparison  
Y\_SW <- Y\_S - Y\_W  
  
#$Model 1 - Reduced Model  
df\_SW1 <- data.frame(Y\_SW, X\_S, check.names = FALSE)  
formula\_SW1 <- as.formula(  
 "Y\_SW ~ D + R + `D:R` + `T:D` + `T:R` + `P:D` + `P:R` + `S:D` + `S:R`")  
  
SW\_Model\_1 <-lm(formula\_SW1, data = df\_SW1)  
  
#Model 2 - Full Model  
df\_SW2 <- data.frame(Y\_SW, X\_S, check.names = FALSE)  
formula\_SW2 <- as.formula(paste("Y\_SW ~ (", paste(linear\_terms,   
 collapse = " + "),")^2"))  
SW\_Model\_2 <-lm(formula\_SW2, data = df\_SW2)  
  
# Summaries  
cat("\n")

cat("Linear Model Fit for SW1 Model (OLS)", "\n")

Linear Model Fit for SW1 Model (OLS)

SW\_Model\_1\_summary <- summary(SW\_Model\_1)  
SW\_Model\_1\_summary

Call:  
lm.default(formula = formula\_SW1, data = df\_SW1)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-220.51 -68.83 37.78 86.55 152.59   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -145.2415 24.6531 -5.891 7.56e-06 \*\*\*  
D -131.0585 24.6531 -5.316 2.85e-05 \*\*\*  
R -20.3415 24.6531 -0.825 0.41858   
`D:R` -9.5710 24.6531 -0.388 0.70176   
`T:D` -0.1273 24.6531 -0.005 0.99593   
`T:R` -3.6602 24.6531 -0.148 0.88339   
`P:D` -55.5290 24.6531 -2.252 0.03512 \*   
`P:R` 1.1790 24.6531 0.048 0.96231   
`S:D` 86.0523 24.6531 3.491 0.00218 \*\*   
`S:R` 3.7852 24.6531 0.154 0.87944   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 136.4 on 21 degrees of freedom  
Multiple R-squared: 0.6929, Adjusted R-squared: 0.5613   
F-statistic: 5.265 on 9 and 21 DF, p-value: 0.0008274

cat("\n")

cat("Linear Model Fit for SW2 Model (OLS)", "\n")

Linear Model Fit for SW2 Model (OLS)

SW\_Model\_2\_summary <- summary(SW\_Model\_2)  
SW\_Model\_2\_summary

Call:  
lm.default(formula = formula\_SW2, data = df\_SW2)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-62.737 -25.938 -1.039 27.896 57.588   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -144.4289 9.0683 -15.927 8.30e-11 \*\*\*  
T 5.3898 9.0683 0.594 0.561130   
P -52.1961 9.0683 -5.756 3.80e-05 \*\*\*  
S 80.6977 9.0683 8.899 2.27e-07 \*\*\*  
D -131.8711 9.0683 -14.542 3.01e-10 \*\*\*  
R -19.5289 9.0683 -2.154 0.047953 \*   
T:P 5.9852 9.0683 0.660 0.519256   
T:S -5.3586 9.0683 -0.591 0.563377   
T:D -0.9398 9.0683 -0.104 0.918828   
T:R -2.8477 9.0683 -0.314 0.757829   
P:S 41.6148 9.0683 4.589 0.000355 \*\*\*  
P:D -54.7164 9.0683 -6.034 2.29e-05 \*\*\*  
P:R 0.3664 9.0683 0.040 0.968303   
S:D 86.8648 9.0683 9.579 8.77e-08 \*\*\*  
S:R 2.9727 9.0683 0.328 0.747589   
D:R -10.3836 9.0683 -1.145 0.270134   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 49.77 on 15 degrees of freedom  
Multiple R-squared: 0.9708, Adjusted R-squared: 0.9416   
F-statistic: 33.24 on 15 and 15 DF, p-value: 8.674e-09

model\_adequacy <- anova(SW\_Model\_1, SW\_Model\_2)  
print(model\_adequacy)

Analysis of Variance Table  
  
Model 1: Y\_SW ~ D + R + `D:R` + `T:D` + `T:R` + `P:D` + `P:R` + `S:D` +   
 `S:R`  
Model 2: Y\_SW ~ (T + P + S + D + R)^2  
 Res.Df RSS Df Sum of Sq F Pr(>F)   
1 21 390667   
2 15 37151 6 353517 23.789 7.349e-07 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

**Method 8B Sequential Split Plot Method with Three-Way Interactions**

SP\_effects\_names <- c(  
 "D",  
 "R",  
 "D:R",  
 "T:D",   
 "P:D",  
 "S:D",  
 "T:R",  
 "P:R",  
 "S:R",  
 "T:D:R",  
 "P:D:R",  
 "S:D:R",  
 "T:P:D",  
 "T:S:D",  
 "P:S:D",  
 "T:P:R",  
 "T:S:R",  
 "P:S:R")  
#-----------------------------------------------------------  
#S Model - SP Factors (WP main effects, SP main effects,   
#and all 2-way and 3-way interactions (10 three-way interactions)  
formula\_S <- as.formula(paste("Y\_S ~ (", paste(linear\_terms,   
 collapse = " + "),")^3"))  
#Create model matrix for S Model  
Y\_S <- Ys\_data$Y\_S  
X\_S <- model.matrix(formula\_S, data = Ys\_data)  
#-----------------------------------------------------------  
#W Model - WP factors (main WP effects, 2-way interactions)  
formula\_W <- as.formula(paste("Y\_W ~ (",   
 paste(WP\_factors, collapse = " + "),")^2"))  
  
#Create model matrix for W Model  
X\_W <- model.matrix(formula\_W, data = Yw\_data)  
# Replicate each row 4 times and drop row 20.  
X\_W <- X\_W[rep(1:nrow(X\_W), each = 4), ][-20, ]  
  
Y\_W <- rep(Yw\_data$Y\_W, each = 4)[-20]  
df\_W <- data.frame(Y\_W, X\_W)  
  
cat("Linear Model Fit for W Model (OLS)", "\n")

Linear Model Fit for W Model (OLS)

W\_Model <- summary(lm(formula\_W, data = Yw\_data))  
W\_Model

Call:  
lm.default(formula = formula\_W, data = Yw\_data)  
  
Residuals:  
 1 2 3 4 5 6 7 8   
 0.5125 -0.5125 -0.5125 0.5125 -0.5125 0.5125 0.5125 -0.5125   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 443.4875 0.5125 865.341 0.000736 \*\*\*  
T -8.5375 0.5125 -16.659 0.038170 \*   
P 150.6625 0.5125 293.976 0.002166 \*\*   
S -225.7625 0.5125 -440.512 0.001445 \*\*   
T:P -5.7125 0.5125 -11.146 0.056962 .   
T:S 2.3125 0.5125 4.512 0.138845   
P:S -78.7375 0.5125 -153.634 0.004144 \*\*   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 1.45 on 1 degrees of freedom  
Multiple R-squared: 1, Adjusted R-squared: 1   
F-statistic: 5.075e+04 on 6 and 1 DF, p-value: 0.003398

#------------------------------------------------------------  
  
  
#S-W Model by Satterwaithe  
#Create model matrix for S-W Model  
  
X\_SW <- X\_S[, SP\_effects\_names]  
Y\_SW <- Y\_S - Y\_W  
df\_SW <- data.frame(Y\_SW, X\_SW, check.names = FALSE)  
   
formula\_SW <- as.formula(  
 "Y\_SW ~ D + R +  
 `D:R` + `T:D` + `P:D` + `S:D` + `T:R` + `P:R` + `S:R` +  
 `T:D:R` + `P:D:R` + `S:D:R` + `T:P:D` + `T:S:D` + `P:S:D` + `T:P:R` + `T:S:R` + `P:S:R`")  
  
SW\_Model <-lm(formula\_SW, data = df\_SW)  
  
cat("\n")

cat("Linear Model Fit for SW1 Model (OLS)", "\n")

Linear Model Fit for SW1 Model (OLS)

SW\_Model\_summary <- summary(SW\_Model)  
SW\_Model\_summary

Call:  
lm.default(formula = formula\_SW, data = df\_SW)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-205.91 -74.01 57.06 82.39 110.28   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -146.6769 31.5293 -4.652 0.000558 \*\*\*  
D -129.6231 31.5293 -4.111 0.001443 \*\*   
R -21.7769 31.5293 -0.691 0.502903   
`D:R` -8.1356 31.5293 -0.258 0.800754   
`T:D` 1.3082 31.5293 0.041 0.967587   
`P:D` -56.9644 31.5293 -1.807 0.095929 .   
`S:D` 84.6168 31.5293 2.684 0.019898 \*   
`T:R` -5.0957 31.5293 -0.162 0.874297   
`P:R` 2.6144 31.5293 0.083 0.935282   
`S:R` 5.2207 31.5293 0.166 0.871243   
`T:D:R` -3.7418 31.5293 -0.119 0.907494   
`P:D:R` -7.8894 31.5293 -0.250 0.806646   
`S:D:R` 9.6543 31.5293 0.306 0.764699   
`T:P:D` -1.5457 31.5293 -0.049 0.961707   
`T:S:D` 0.3981 31.5293 0.013 0.990134   
`P:S:D` 31.2832 31.5293 0.992 0.340689   
`T:P:R` 3.1832 31.5293 0.101 0.921250   
`T:S:R` 3.4519 31.5293 0.109 0.914629   
`P:S:R` 1.9793 31.5293 0.063 0.950977   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 171.9 on 12 degrees of freedom  
Multiple R-squared: 0.7214, Adjusted R-squared: 0.3034   
F-statistic: 1.726 on 18 and 12 DF, p-value: 0.1688

#-------------------------------------------------------------

```