Detailed statistical analysis

2025-08-28

```
data <- read.table(file ="data.txt", header=TRUE)
dim(data)
```

```
## [1] 48  5
```

```
names(data)
```

```
## [1] "numDefects" "prebake"    "flux"       "cooling"    "temp"
```

```
data$prebake <- factor(data$prebake)
data$flux <- factor(data$flux)
data$cooling <- factor(data$cooling)
data$temp <- factor(data$temp)
str(data)
```

```
## 'data.frame':    48 obs. of  5 variables:
##  $ numDefects: int  13 4 20 42 14 10 36 5 29 10 ...
##  $ prebake   : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 2 2 ...
##  $ flux      : Factor w/ 2 levels "1","2": 1 1 1 1 2 2 2 2 1 1 ...
##  $ cooling   : Factor w/ 2 levels "1","2": 1 2 1 2 2 1 2 1 2 1 ...
##  $ temp      : Factor w/ 2 levels "1","2": 1 2 2 1 2 1 1 2 1 2 ...
```

The data is taken from Condra, Lloyd, Reliability Improvement with Design of Experiment. CRC Press, 2001. Full wavesolder data has 48 observations, each of which has the number of defects and seven predictor variables.

We are predicting numDefects, which represents a count of events. Since we don't know the total number of things defects come from, a binomial regression doesn't really make sense here. Because the number of defects are integers, poisson model should be better than gamma model.

First I'll plot the data to understand it.

Since the link function for poisson are log(mean), mean, and sqrt(mean), and the parameters only take 2 values, the gradient of the lines are going to stay the same after applying any of the link function to the mean. For simplicity of code, mean vs data is plotted.

```
par(mfrow = c(3, 2), mar = c(4.5, 4.5, 2, 1), oma = c(0, 0, 3, 0))

# Plot 1: Prebake vs. Flux
with(data, interaction.plot(x.factor = prebake, trace.factor = flux,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Prebake"))

# Plot 2: Prebake vs. Cooling
with(data, interaction.plot(x.factor = prebake, trace.factor = cooling,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Prebake"))

# Plot 3: Prebake vs. Temp
```

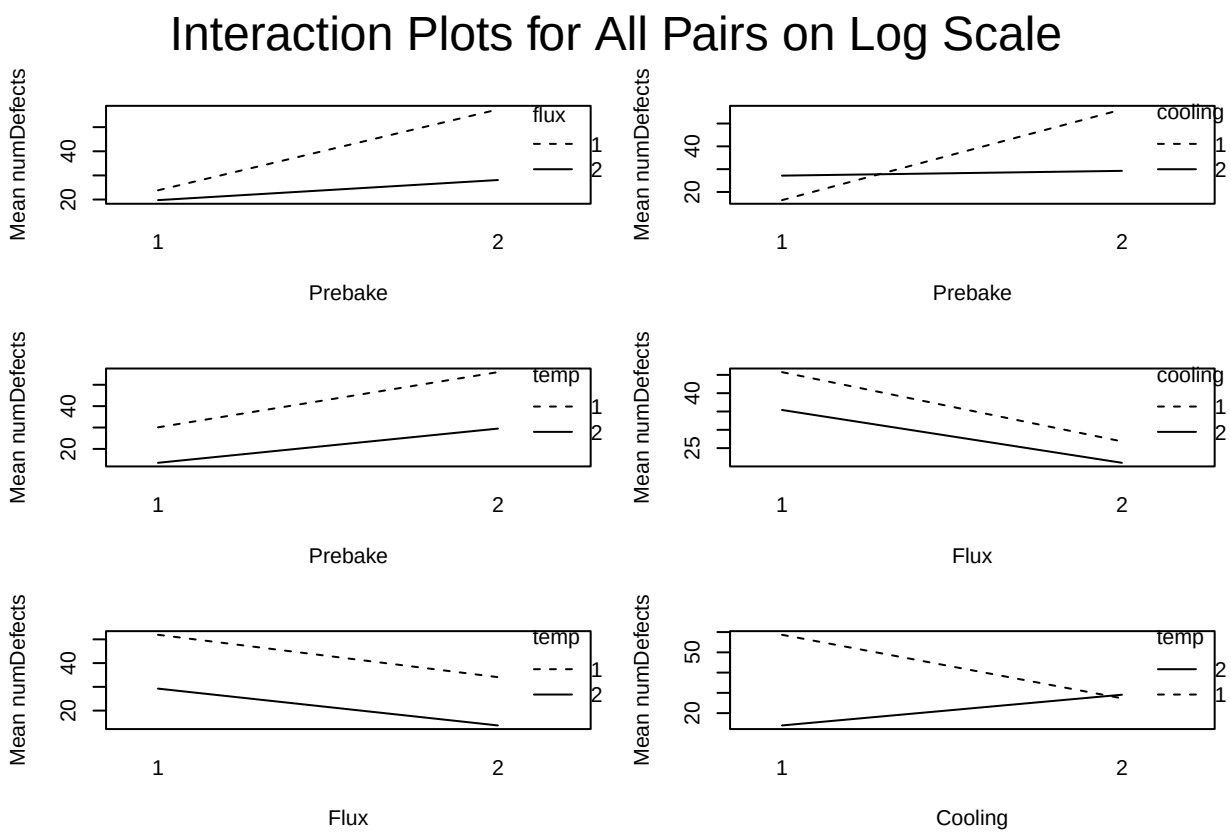
```r
with(data, interaction.plot(x.factor = prebake, trace.factor = temp,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Prebake"))

# Plot 4: Flux vs. Cooling
with(data, interaction.plot(x.factor = flux, trace.factor = cooling,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Flux"))

# Plot 5: Flux vs. Temp
with(data, interaction.plot(x.factor = flux, trace.factor = temp,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Flux"))

# Plot 6: Cooling vs. Temp
with(data, interaction.plot(x.factor = cooling, trace.factor = temp,
                            response = data$numDefects,
                            ylab = "Mean numDefects", xlab = "Cooling"))

# Add an overall title to the entire set of plots
mtext("Interaction Plots for All Pairs on Log Scale", outer = TRUE, cex = 1.5)
```



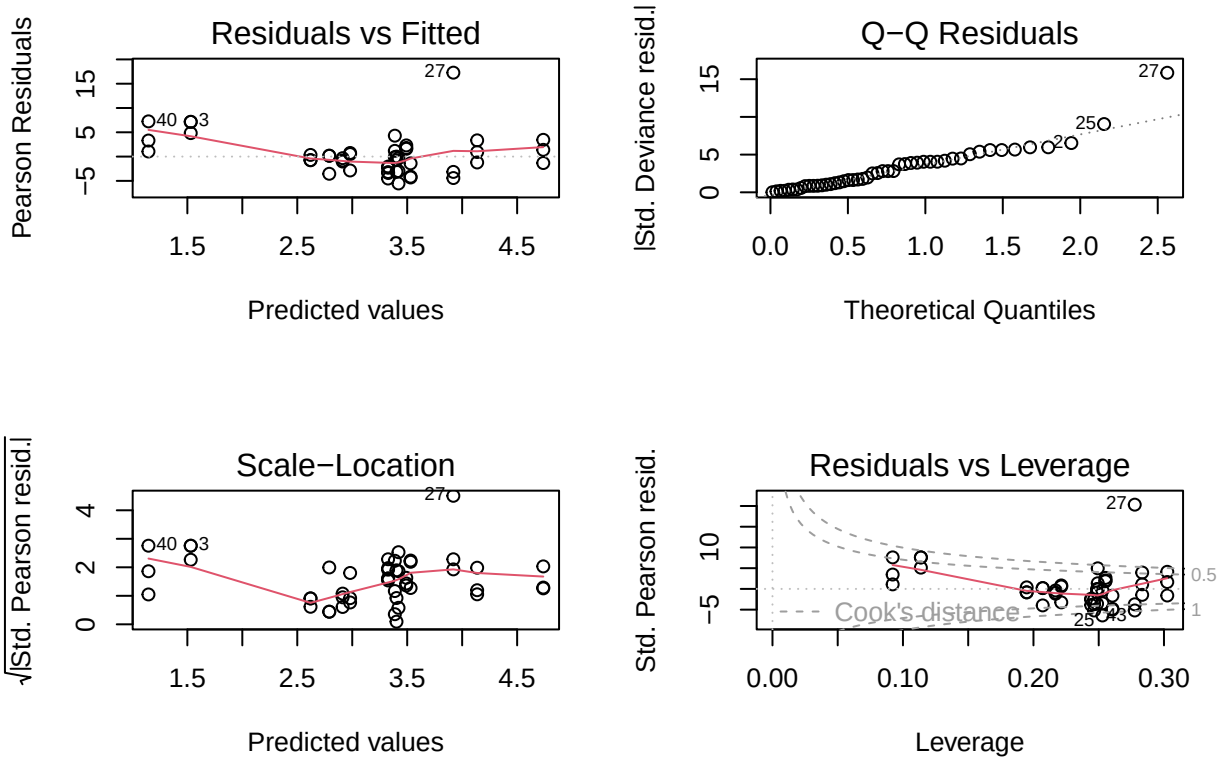
From the plot it looks like there's no interaction between temp&flux, and very little interaction between prebake&temp and flux&cooling. But strong evidence for interaction between other parameters are present.

Model with interaction between every parameter.

```r
model1 = glm(numDefects ~ .^2, family = poisson, data = data)
summary(model1)
```

```
##
## Call:
## glm(formula = numDefects ~ .^2, family = poisson, data = data)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        3.40257    0.09099  37.395  < 2e-16 ***
## prebake2           1.33522    0.09850  13.555  < 2e-16 ***
## flux2             -0.07552    0.11253  -0.671   0.5022
## cooling2           0.08918    0.11330   0.787   0.4312
## temp2             -1.87164    0.14963 -12.508  < 2e-16 ***
## prebake2:flux2    -0.52572    0.11601  -4.532 5.85e-06 ***
## prebake2:cooling2 -1.40598    0.12406 -11.333  < 2e-16 ***
## prebake2:temp2     0.66473    0.13189   5.040 4.66e-07 ***
## flux2:cooling2    -0.02873    0.11955  -0.240   0.8101
## flux2:temp2       -0.30910    0.12274  -2.518   0.0118 *
## cooling2:temp2     1.70585    0.12465  13.685  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1450.52  on 47  degrees of freedom
## Residual deviance:  626.93  on 37  degrees of freedom
## AIC: 879.67
##
## Number of Fisher Scoring iterations: 5
```

```r
par(mfrow = c(2,2))
plot(model1)
```



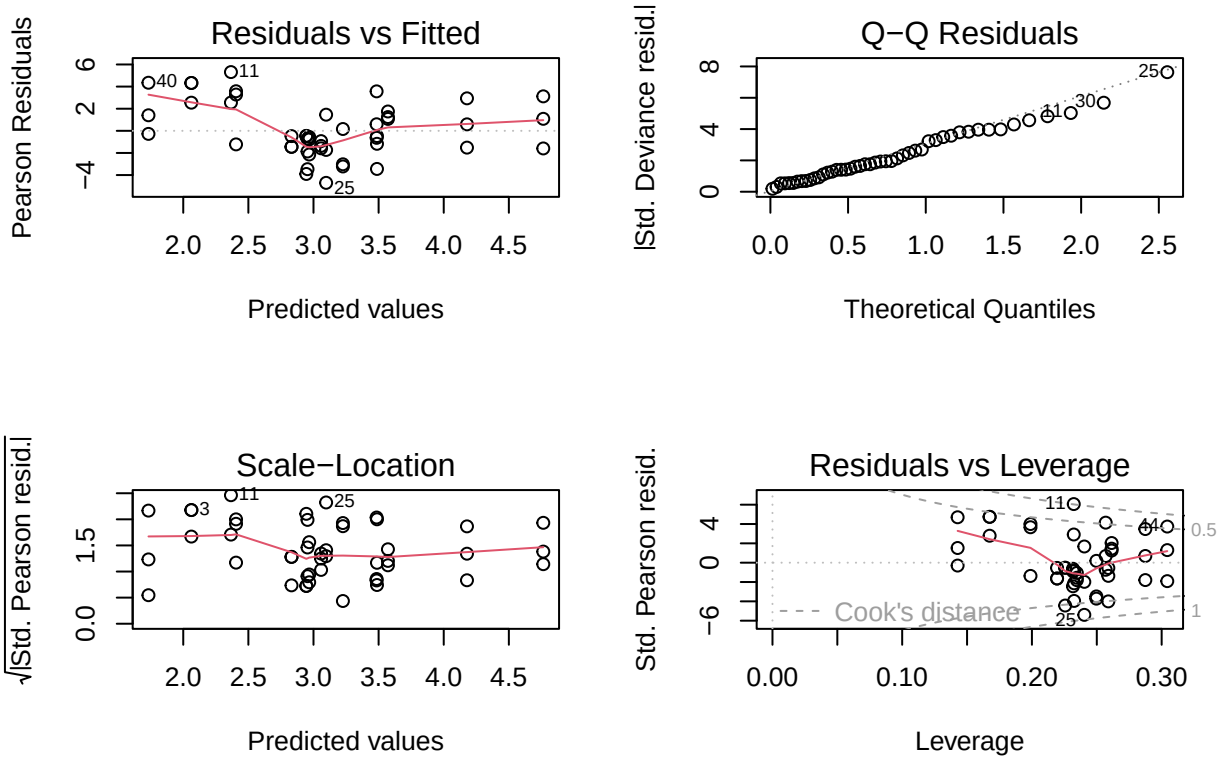
From all the diagnostic plots, point 27 is a very big outlier. It has a very big cook's distance, skewing the result. Get rid of this data point.

```
data2 = data[-27,]
model1 = glm(numDefects ~ .^2, family = poisson, data = data2)
summary(model1)
```

```
##
## Call:
## glm(formula = numDefects ~ .^2, family = poisson, data = data2)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         3.48763    0.08897  39.201  < 2e-16 ***
## prebake2            1.27668    0.09709  13.150  < 2e-16 ***
## flux2              -0.52051    0.12273  -4.241 2.23e-05 ***
## cooling2            0.08401    0.11157   0.753 0.451468
## temp2              -1.42521    0.14675  -9.712  < 2e-16 ***
## prebake2:flux2     -0.06522    0.12486  -0.522 0.601393
## prebake2:cooling2 -1.75122    0.12582 -13.918  < 2e-16 ***
## prebake2:temp2     -0.11317    0.14214  -0.796 0.425926
## flux2:cooling2      0.43362    0.12634   3.432 0.000599 ***
## flux2:temp2         0.19109    0.12856   1.486 0.137195
## cooling2:temp2      0.80771    0.14261   5.664 1.48e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1137.32  on 46  degrees of freedom
## Residual deviance:  288.05  on 36  degrees of freedom
## AIC: 533.8
##
## Number of Fisher Scoring iterations: 5
```

```
par(mfrow = c(2,2))
plot(model1)
```




```
df1 = nrow(data) - 11
pearson_chisq1 <- sum(residuals(model1, type="pearson")^2)
pearson_chisq1/df1
```

```
## [1] 7.453388
```

The diagnostic plots are a lot better than before. Residuals are pretty evenly scattered around, points lie roughly on the qq plot, and there're no points with huge cook's distance. However, $7.4533882 > 1$. Pearson's chi squared statistics is an estimator for phi ($= 1$ for poisson model). Since it's much bigger than 1 here that means our data has a greater variance than the mdoel expects, meaning there's overdispersion at play here. Deal with overdispersion with quasilikelihood method.

```
model2 = glm(numDefects ~ .^2, family = quasipoisson, data = data2)
summary(model2)
```

```
##
## Call:
```

```
## glm(formula = numDefects ~ .^2, family = quasipoisson, data = data2)
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        3.48763    0.24624  14.163 2.73e-16 ***
## prebake2           1.27668    0.26871   4.751 3.21e-05 ***
## flux2             -0.52051    0.33970  -1.532  0.13420
## cooling2           0.08401    0.30881   0.272  0.78714
## temp2             -1.42521    0.40617  -3.509  0.00123 **
## prebake2:flux2    -0.06522    0.34557  -0.189  0.85135
## prebake2:cooling2 -1.75122    0.34824  -5.029 1.38e-05 ***
## prebake2:temp2    -0.11317    0.39340  -0.288  0.77525
## flux2:cooling2     0.43362    0.34968   1.240  0.22298
## flux2:temp2        0.19109    0.35583   0.537  0.59456
## cooling2:temp2     0.80771    0.39472   2.046  0.04808 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 7.660433)
##
##     Null deviance: 1137.32  on 46  degrees of freedom
## Residual deviance:  288.05  on 36  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Comparing model2 and model1, we can see that many of the parameters are no longer significant.

Now drop parameters that are not significant. Update until everything is significant from drop1. A model with less parameter is desired, since it will have more degrees of freedom, less likely to overfit and easier to explain.

```
drop1(model2, test = "F")
```

```
## Single term deletions
##
## Model:
## numDefects ~ (prebake + flux + cooling + temp)^2
##                 Df Deviance F value    Pr(>F)
## <none>               288.05
## prebake:flux     1   288.32  0.0340   0.85464
## prebake:cooling  1   497.61 26.1914 1.048e-05 ***
## prebake:temp     1   288.68  0.0790   0.78029
## flux:cooling     1   299.86  1.4762   0.23228
## flux:temp        1   290.25  0.2753   0.60300
## cooling:temp     1   320.03  3.9967   0.05318 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model3 = update(model2, . ~ . - prebake:flux)
drop1(model3,test = "F")
```

```
## Single term deletions
##
## Model:
## numDefects ~ prebake + flux + cooling + temp + prebake:cooling +
```

```
##     prebake:temp + flux:cooling + flux:temp + cooling:temp
##                 Df Deviance F value    Pr(>F)
## <none>               288.32
## prebake:cooling  1   502.65 27.5049 6.612e-06 ***
## prebake:temp     1   289.02  0.0892   0.76681
## flux:cooling     1   304.31  2.0518   0.16042
## flux:temp        1   290.60  0.2920   0.59219
## cooling:temp     1   320.06  4.0733   0.05086 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model3 = update(model3, . ~ . - prebake:temp)
drop1(model3,test = "F")
```

```
## Single term deletions
##
## Model:
## numDefects ~ prebake + flux + cooling + temp + prebake:cooling +
##     flux:cooling + flux:temp + cooling:temp
##                 Df Deviance F value    Pr(>F)
## <none>               289.02
## prebake:cooling  1   513.45 29.5088 3.419e-06 ***
## flux:cooling     1   304.88  2.0854   0.15691
## flux:temp        1   291.22  0.2895   0.59369
## cooling:temp     1   333.27  5.8190   0.02079 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model3 = update(model3, . ~ . - flux:temp)
drop1(model3,test = "F")
```

```
## Single term deletions
##
## Model:
## numDefects ~ prebake + flux + cooling + temp + prebake:cooling +
##     flux:cooling + cooling:temp
##                 Df Deviance F value    Pr(>F)
## <none>               291.22
## prebake:cooling  1   514.67 29.9254 2.814e-06 ***
## flux:cooling     1   309.76  2.4828   0.12317
## cooling:temp     1   339.19  6.4243   0.01538 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model3 = update(model3, . ~ . - flux:cooling)
drop1(model3,test = "F")
```

```
## Single term deletions
##
## Model:
## numDefects ~ prebake + flux + cooling + temp + prebake:cooling +
##     cooling:temp
##                 Df Deviance F value    Pr(>F)
## <none>               309.76
## flux             1   351.33  5.3684   0.02571 *
## prebake:cooling  1   530.57 28.5143 3.987e-06 ***
```

```
## cooling:temp     1   358.86  6.3408   0.01590 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model3)
```

```
##
## Call:
## glm(formula = numDefects ~ prebake + flux + cooling + temp +
##     prebake:cooling + cooling:temp, family = quasipoisson, data = data2)
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3.4384     0.2091  16.448  < 2e-16 ***
## prebake2            1.2300     0.2212   5.561 1.95e-06 ***
## flux2              -0.3502     0.1494  -2.344   0.0241 *
## cooling2            0.2601     0.2675   0.972   0.3367
## temp2              -1.4388     0.2351  -6.121 3.19e-07 ***
## prebake2:cooling2  -1.7608     0.3375  -5.218 5.90e-06 ***
## cooling2:temp2      0.8914     0.3472   2.567   0.0141 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 7.458044)
##
##     Null deviance: 1137.32  on 46  degrees of freedom
## Residual deviance:  309.76  on 40  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Make sure dropping these parameters is not detremental to the model, compare the final model and full model with F test

```
anova(model3,model2, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: numDefects ~ prebake + flux + cooling + temp + prebake:cooling +
##     cooling:temp
## Model 2: numDefects ~ (prebake + flux + cooling + temp)^2
##   Resid. Df Resid. Dev Df Deviance      F Pr(>F)
## 1        40     309.76
## 2        36     288.05  4   21.709 0.7085 0.5915
```

The extra terms in anova is not significant enough to be put into the model

Take a look at the reduced model

```
summary(model3)
```

```
##
## Call:
## glm(formula = numDefects ~ prebake + flux + cooling + temp +
##     prebake:cooling + cooling:temp, family = quasipoisson, data = data2)
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)         3.4384     0.2091  16.448  < 2e-16 ***
## prebake2            1.2300     0.2212   5.561 1.95e-06 ***
## flux2              -0.3502     0.1494  -2.344   0.0241 *
## cooling2            0.2601     0.2675   0.972   0.3367
## temp2              -1.4388     0.2351  -6.121 3.19e-07 ***
## prebake2:cooling2  -1.7608     0.3375  -5.218 5.90e-06 ***
## cooling2:temp2      0.8914     0.3472   2.567   0.0141 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 7.458044)
##
##     Null deviance: 1137.32  on 46  degrees of freedom
## Residual deviance:  309.76  on 40  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Main effects

Prebake, flux, and temp were all found to have a significant main effect on the number of defects. The negative coefficients for flux2 (-0.35) and temp2 (-1.44) indicate that switching from level 1 to level 2 for these factors is associated with a decrease in defects. Conversely, the positive coefficient for prebake2 (1.23) suggests that using the second prebake setting is associated with an increase in defects.

The main effect for cooling was not statistically significant (p = 0.3367). This implies that, on its own, changing the cooling level does not have a simple, consistent impact. However, cooling is a critically important variable due to its involvement in two significant interactions.

interaction effects

Prebake:cooling (p = 0.0059): The coefficient for prebake2:cooling2 is -1.76. This significant negative interaction means that the effect of prebake depends on the cooling level. While prebake=2 is generally bad (increases defects), pairing it with cooling=2 counteracts this negative effect more strongly than would be expected from their individual effects. This suggests a specific combination of settings can mitigate a problematic factor.

Cooling:temp (p = 0.0141): The coefficient for cooling2:temp2 is 0.89. This significant positive interaction indicates that the combined effect of cooling=2 and temp=2 is less beneficial than the sum of their individual effects. While both settings are generally associated with fewer defects, their benefits are not fully additive when used together.

Conclusion

All parameters have an effect on the defects, and they're not independent.

Since 1.2300264+0.2600942-1.7607741=-0.2706535 (prebake2 + cooling2 + prebake2:cooling2), 0.8914352-1.4387845=-0.5473493 (temp2 + cooling2:temp2) and flux2 with -0.3502419. There's significant evidence supporting soldering with everything on the second level reduces the rate of defects arising.

If the business is unable to switch all process settings to level 2, the data provides a clear, risk-based path forward:

The most effective and lowest-risk strategy is to prioritize switching Flux and Temperature to Setting 2. The model shows that these two factors independently and significantly reduce the defect rate. Their lack of a strong interaction means their benefits are additive and can be implemented without complex process control. The Temp setting, in particular, offers the single largest improvement.

Conversely, leaving Prebake and Cooling at Setting 1 is the most prudent course of action in a constrained environment. While the prebake2:cooling2 combination does offer a minor net benefit, it introduces significant

operational risk. The model clearly shows that the main effects of prebake2 and cooling2 on their own can increase defects, meaning an error in implementing the combination could be highly detrimental. Therefore, the minor potential reward does not outweigh the substantial risk.