SDS 4392 HW4

Washington University, SP 2025

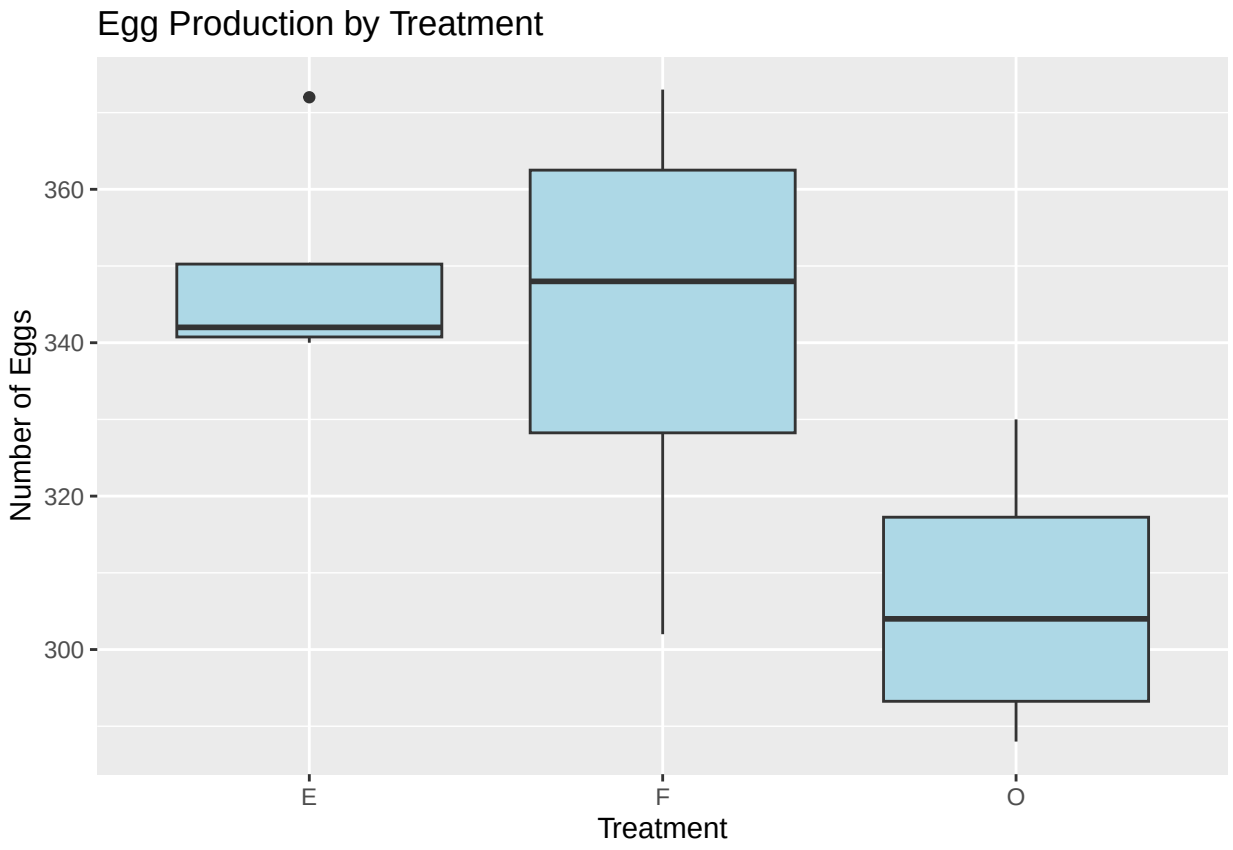
Aidan Kardan

SDS 4392 HW 4

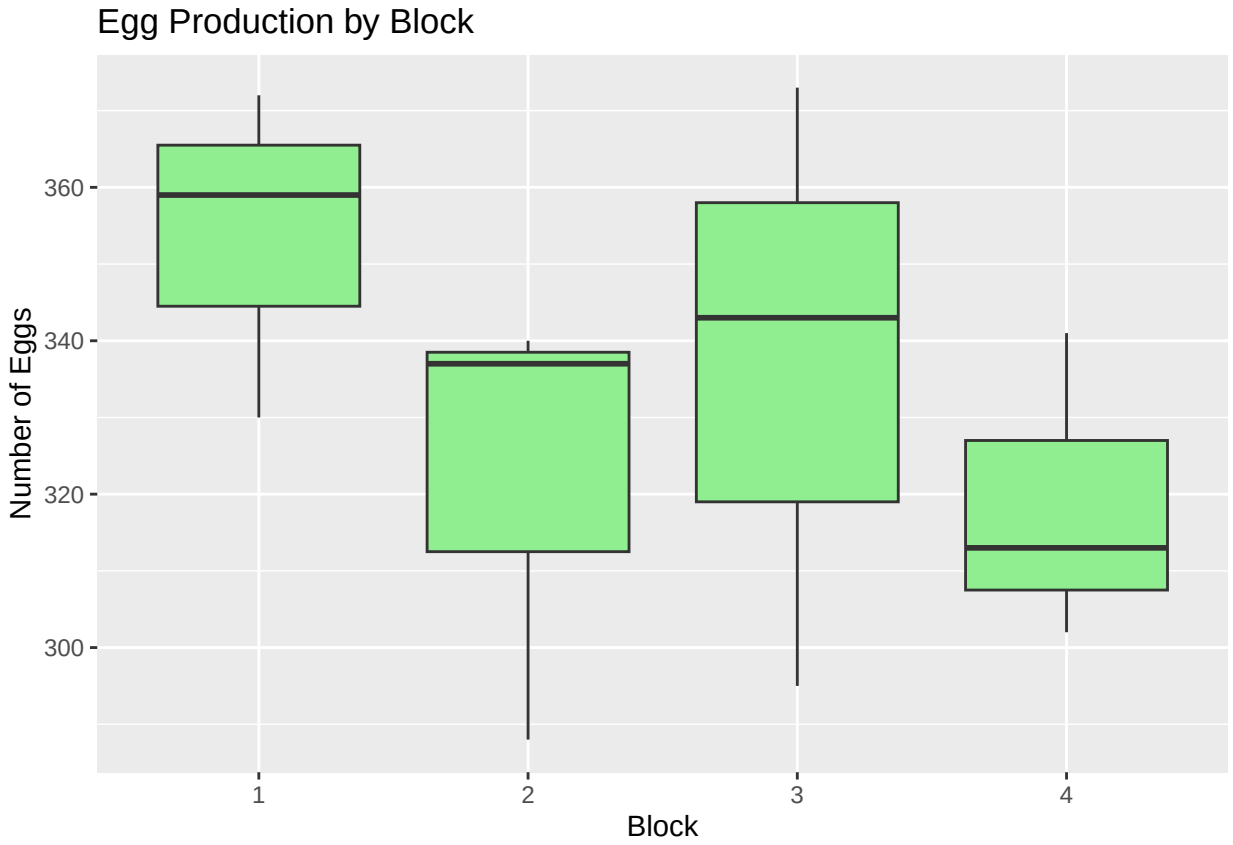
1. The `eggprod` dataset concerns an experiment where six pullets were placed into each of 12 pens. Four blocks were formed from groups of three pens based on location. Three treatments were applied. The bumber of eggs produced was recorded.

a) Make suitable plots of the data and comment.

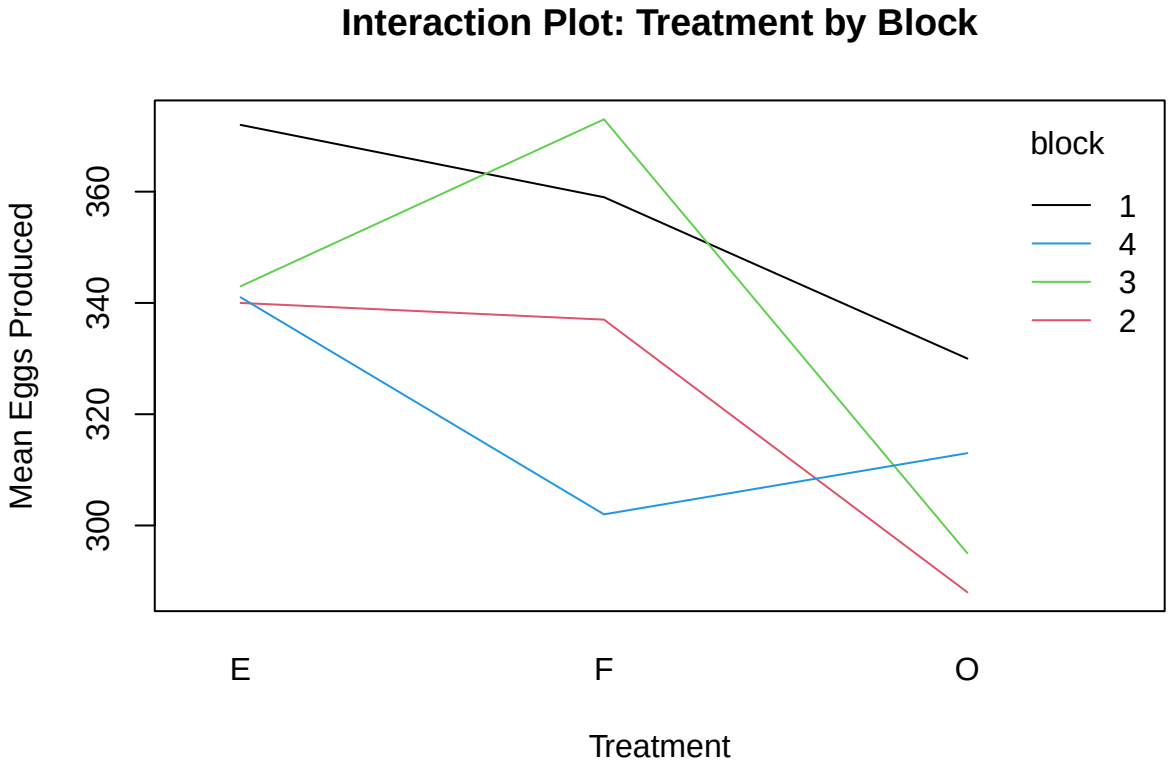
```
library(faraway)
library(ggplot2)
df1<- faraway::eggprod

# Boxplot by treatment
ggplot(df1, aes(x = factor(treat), y = eggs)) +
  geom_boxplot(fill = "lightblue") +
  labs(x = "Treatment", y = "Number of Eggs", title = "Egg Production by Treatment")
```



```r
# Boxplot by block
ggplot(df1, aes(x = factor(block), y = eggs)) +
  geom_boxplot(fill = "lightgreen") +
  labs(x = "Block", y = "Number of Eggs", title = "Egg Production by Block")
```




```r
# Interaction plot: mean eggs by treatment and block
with(df1, interaction.plot(treat, block, eggs,
                           col = 1:4, lty = 1,
                           main = "Interaction Plot: Treatment by Block",
                           xlab = "Treatment", ylab = "Mean Eggs Produced"))
```



Treatment Groups E and F have higher averages than Treatment Group O. Treatment Group E has an outlier (the highest number of eggs reported). Considering the spread/variation within and across Treatment groups shows that the variance within treatments groups is quite large, and the variance across treatment groups appears similar for Treatment Groups F and O, while the variance across Group E is quite tight apart from the outlier. Block 1,2 and 3 have higher averages than Block 4. The variance within blocks shows that block 1 is the most consistent with its egg production, while block 3 has the most variation. Block 2 has numerous observations near 340 and then gets dumped below 300 but the change is not as severe as Block 3. Block 4 has almost as much variation as Block 2 with a much lower average.

It seems that Treatment Group E and block 1 are the best combination for producing the most eggs in the long-run, but the plots clearly show Treatment Group E is the best regardless of block chosen.

b) Fit a fixed effects model for the number of eggs produced with the treatments and blocks as predictors. Determine the significance of the two predictors and perform a basic diagnostic check.

```
mod_fixed <- lm(eggs ~ treat + block, data = df1)
summary(mod_fixed)
```

```
##
## Call:
## lm(formula = eggs ~ treat + block, data = df1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -26.667  -8.125   2.083   5.521  26.000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   369.92      13.91  26.596 1.87e-07 ***
## treatF         -6.25      13.91  -0.449   0.6690
## treatO        -42.50      13.91  -3.056   0.0224 *
```

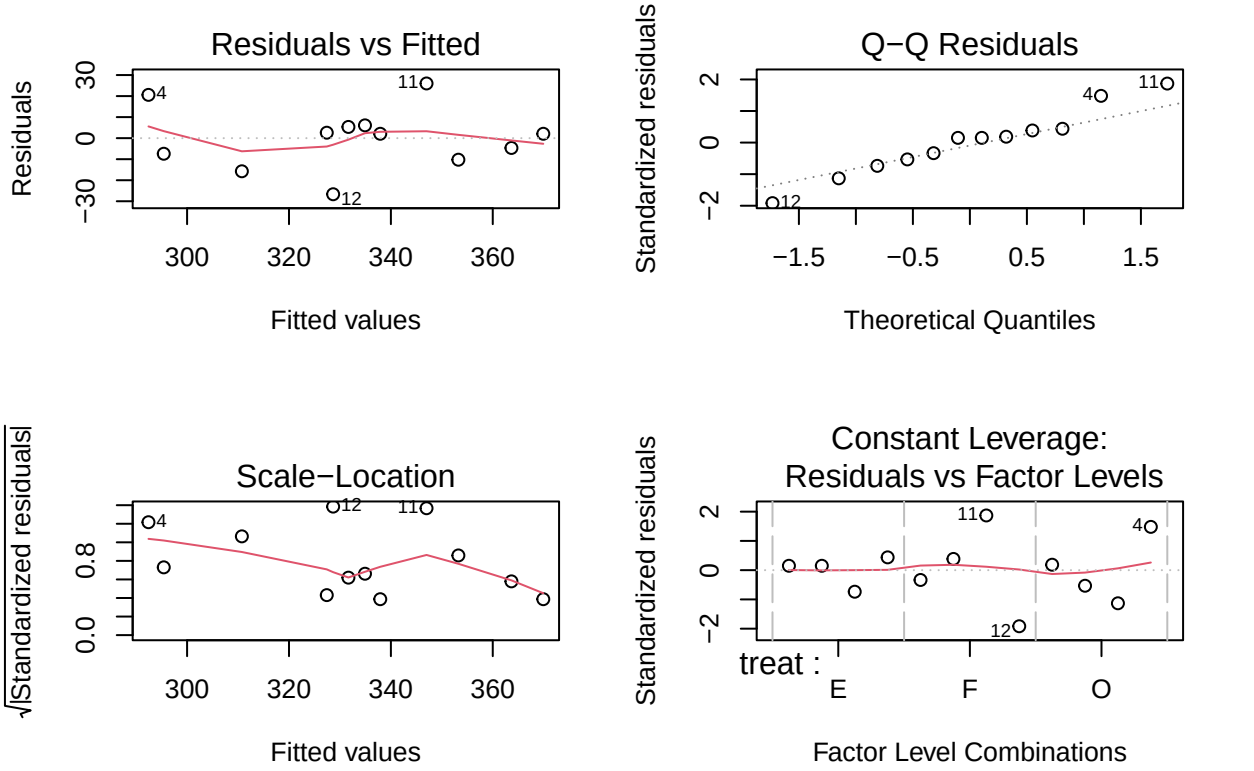
```
## block2        -32.00      16.06  -1.992   0.0934 .
## block3        -16.67      16.06  -1.038   0.3394
## block4        -35.00      16.06  -2.179   0.0721 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.67 on 6 degrees of freedom
## Multiple R-squared:  0.7381, Adjusted R-squared:  0.5199
## F-statistic: 3.382 on 5 and 6 DF,  p-value: 0.08504
```

The predictor that signifies Treatment Group O is statistically significant at the 0.05 level. The model fit is not bad with a decent R-squared (0.7381), but the F-test shows that the overall model is not statistically significant.

Treatment O was considered statistically significant which makes sense since the plots showed a clear disparity, much lower average egg production for Treatment Group O compared to the other groups. None of the blocks were statistically significant at the 0.05 level, meaning none of the blocks appeared to have a profound effect.

The intercept, which contains Group E and Block 1 is the most statistically significant result, which establishes the baseline level. All predictor coefficients are negative meaning they would contribute to lower egg production if they were statistically significant. Overall, the results suggest that differences in Treatment, particularly for Treatment O are relevant but further investigation is needed to rank block performance.

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



The residuals vs fitted values shows good dispersion with no obvious pattern. The QQ-residuals do not seem to exhibit a clear S-shape signifying violation of the normality assumption. Scale-Location shows relatively horizontal line with points evenly spread, indicating variance of residuals is constant. There are no influential observations based on Residuals vs Leverage plot.

c) Fit a model for the number of eggs produced with the treatments as fixed effects and the blocks as

random effects. Which treatment is best in terms of maximizing production according to the model?

```
library(lme4)
```

```
## Loading required package: Matrix
```

```
mod_mixed <- lmer(eggs ~ treat + (1 | block), data = df1)
summary(mod_mixed)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: eggs ~ treat + (1 | block)
##    Data: df1
##
## REML criterion at convergence: 85.4
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.71233 -0.47453 -0.02845  0.64196  1.42942
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  block    (Intercept) 129.9    11.40
##  Residual             386.9    19.67
## Number of obs: 12, groups:  block, 4
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   349.00      11.37  30.702
## treatF         -6.25      13.91  -0.449
## treatO        -42.50      13.91  -3.056
##
## Correlation of Fixed Effects:
##        (Intr) treatF
## treatF -0.612
## treatO -0.612  0.500
```

According to the mixed effects model, the best treatment is the baseline treatment, which is Treatment E. Treatment F does not differ significantly from Treatment E. Treatment O does significantly worse, with a notable decrease in egg production compared to Treatment E.

The best treatment for maximizing egg production is Treatment E.

d) Use the Kenward-Roger approximation for an F-test to check ofr differences between the treatments. How does the results compare to the fixed effects result?

```
library(lmerTest)
```

```
##
## Attaching package: 'lmerTest'
```

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

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

```r
mod_mixed_kr <- lmer(eggs ~ treat + (1 | block), data = df1)
anova(mod_mixed_kr, ddf = "Kenward-Roger")
```

```
## Type III Analysis of Variance Table with Kenward-Roger's method
##       Sum Sq Mean Sq NumDF DenDF F value  Pr(>F)
## treat 4212.5  2106.2     2     6  5.4437 0.04485 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The Kenward-Roger F-test for the treatment effect yields an F value of 5.4437 with a p-value of 0.04485, indicating that there are significant differences among the treatments at the 5% level. This result suggests that, when accounting for the random block effects, the treatment factor significantly influences egg production. Compared to the fixed effects analysis—which identified a significant difference for one treatment (Treatment O) but did not show overall significance as clearly—the Kenward-Roger adjusted test supports the conclusion that treatments differ in their effect. This confirms that the mixed-effects model (with block as a random effect) provides a more sensitive test for treatment differences in this experiment.

e) Perform the same test but using a bootstrap method. How do the results compare?

```r
set.seed(123)  # For reproducibility
boot_fun <- function(mod) {
  fixef(mod)
}
boot_mod <- bootMer(mod_mixed, FUN = boot_fun, nsim = 1000)

# Summarize the bootstrap estimates for each treatment coefficient
boot_mod
```

```
##
## PARAMETRIC BOOTSTRAP
##
##
## Call:
## bootMer(x = mod_mixed, FUN = boot_fun, nsim = 1000)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1*   349.00  0.08328915    11.05306
## t2*    -6.25 -0.18058820    14.24923
## t3*   -42.50  0.13948987    13.73866
```

```
##
## 307 message(s): boundary (singular) fit: see help('isSingular')
```

The parametric bootstrap analysis estimates are very similar to those obtained using the Kenward-Roger approximation. The bootstrap results confirm that the baseline treatment (t1) is robustly estimated around 349 eggs. The effect of Treatment F (t2) remains small and non-significant. The negative effect for Treatment O (t3) is consistent (with an estimate around –42.5), reinforcing its statistical significance.

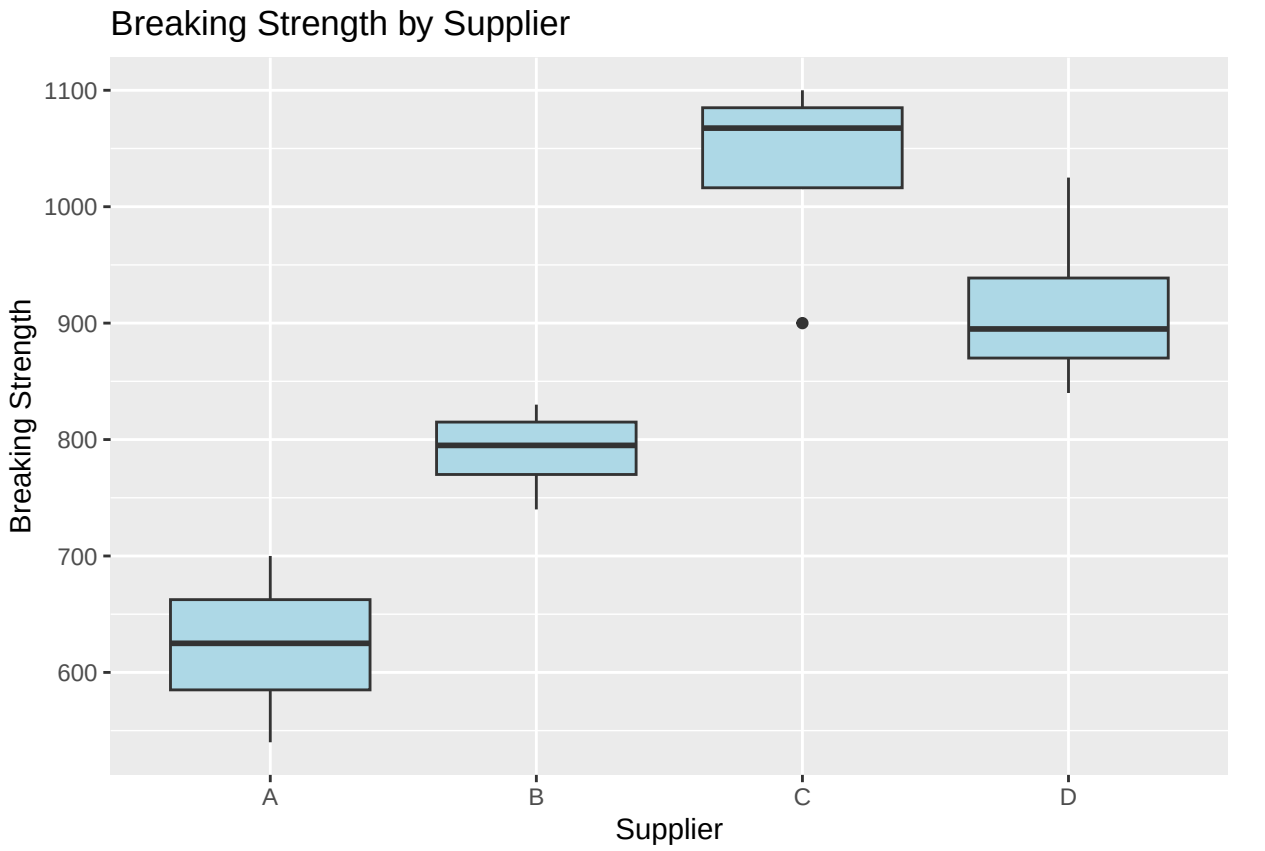
There is some variability as indicated by the bootstrap standard errors, and although a few singular fits were encountered (a known issue in small datasets), the overall conclusions remain the same. In summary, the bootstrap results are consistent with the Kenward-Roger F-test, supporting the conclusion that there are significant differences between treatments, primarily driven by the effect of Treatment O.

2. An experiment was conducted to select the supplier of raw materials for production of a component. The breaking strength of the component was the objective of interest. Four suppliers were considered.

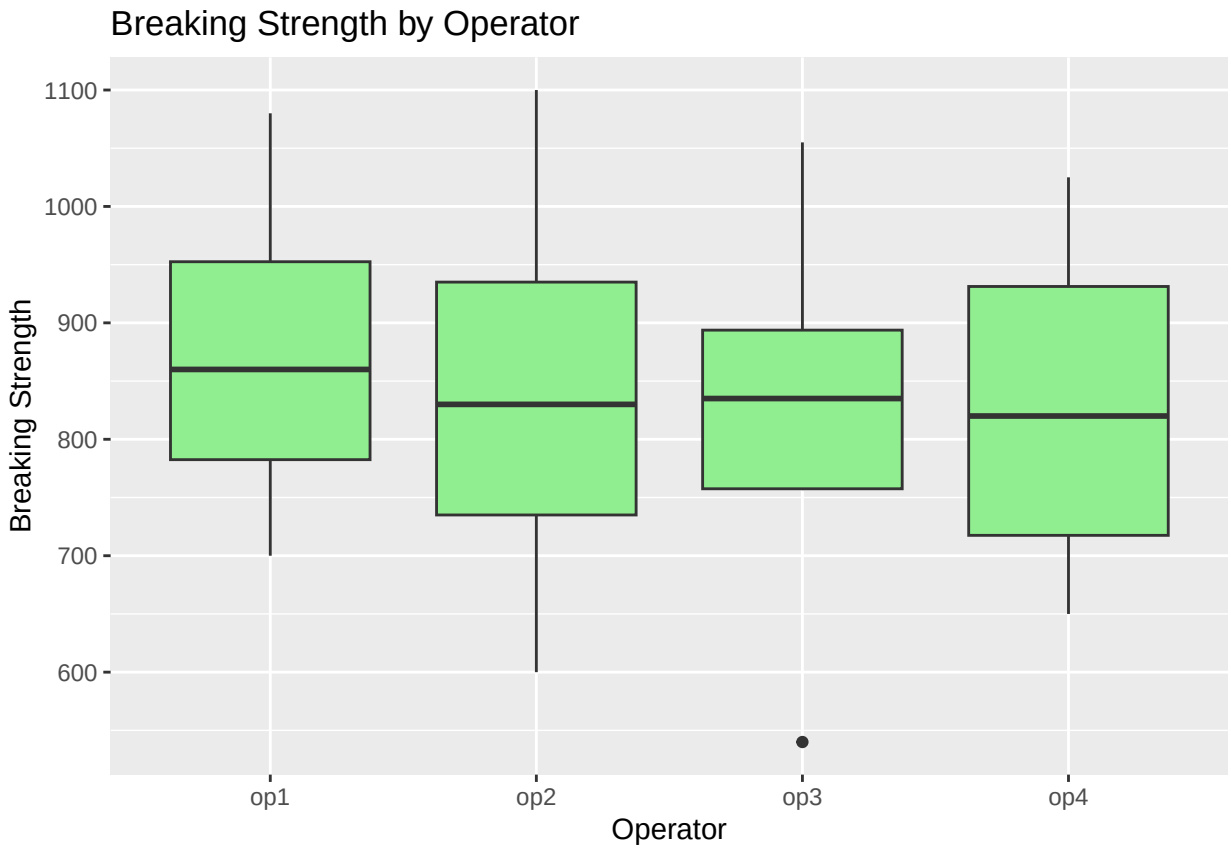
The four operators can only produce one component each per day. A latin square design is used and the data is presented in breaking.

(a) Plot the data and interpret.

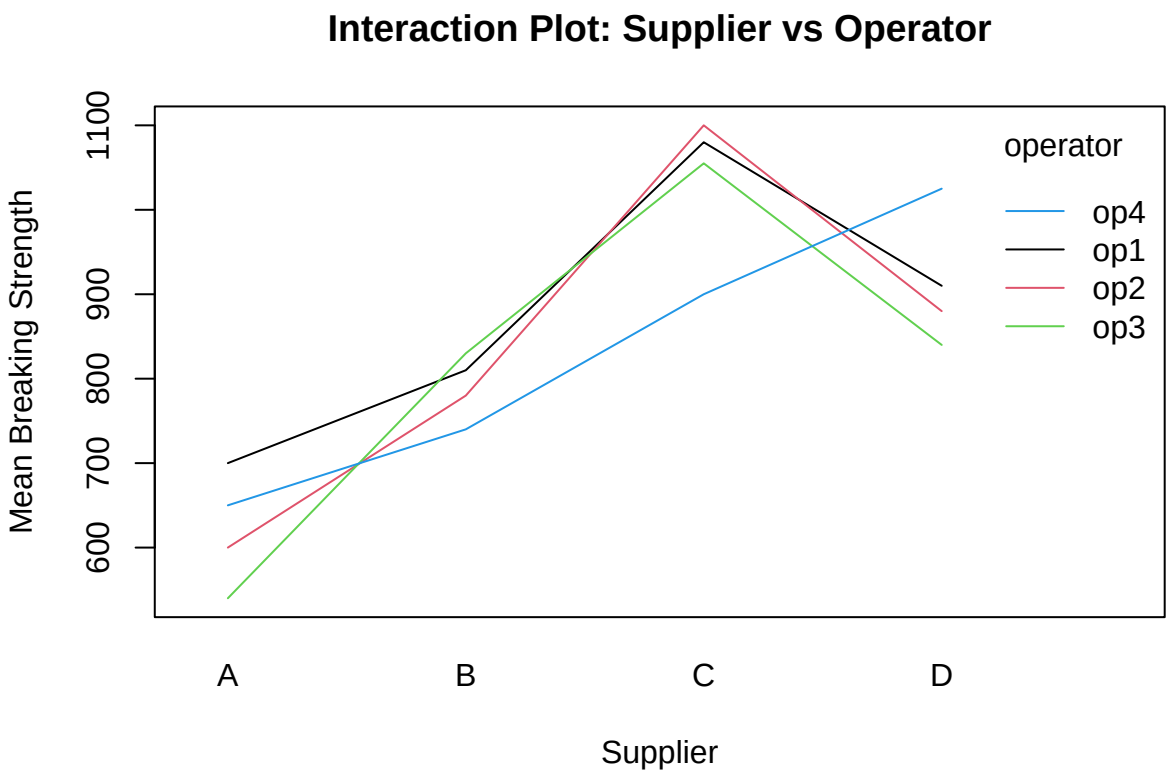
```
library(faraway)
library(ggplot2)
df2 <- faraway::breaking

# Boxplot by supplier
ggplot(df2, aes(x = factor(supplier), y = y)) +
  geom_boxplot(fill = "lightblue") +
  labs(x = "Supplier", y = "Breaking Strength", title = "Breaking Strength by Supplier")
```




```
# Boxplot by operator (if desired)
ggplot(df2, aes(x = factor(operator), y = y)) +
  geom_boxplot(fill = "lightgreen") +
  labs(x = "Operator", y = "Breaking Strength", title = "Breaking Strength by Operator")
```




```
# Interaction plot for supplier vs. operator (or day)
with(df2, interaction.plot(supplier, operator, y,
                           col = 1:4, lty = 1,
                           main = "Interaction Plot: Supplier vs Operator",
                           xlab = "Supplier", ylab = "Mean Breaking Strength"))
```



There is an outlier in Supplier C. The variance within Suppliers is the tightest for Supplier B, with Supplier A close behind. Supplier C has higher dispersion of breaking strength than A or B. Supplier D has the highest dispersion across the suppliers. Supplier A has the lowest average breaking strength. Supplier B has the second lowest, Supplier C has the highest, Supplier D has the second highest. The breaking strength by operator truly shows that operator should be treated as a random effect because the variance within each operator and across operators are almost identical, with almost identical average breaking strengths as well.

(b) Fit a fixed effects model for the main effects. Determine which factors are significant.

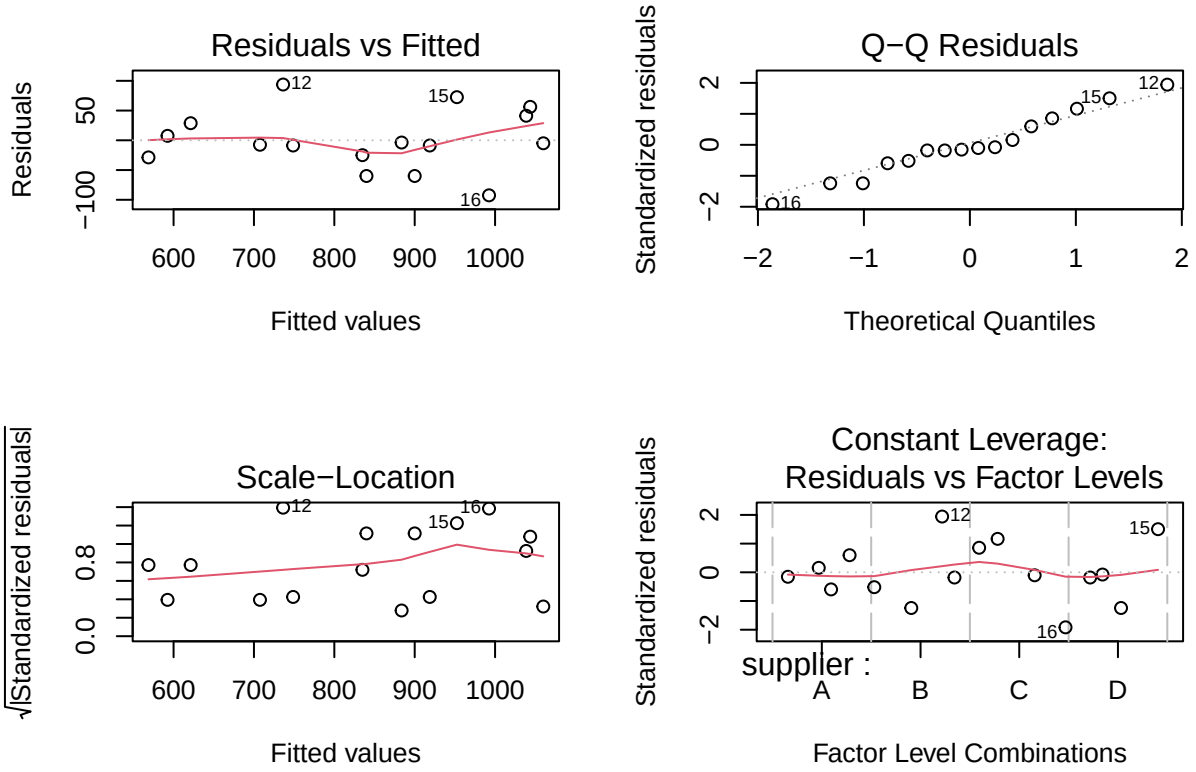
```
mod2_fixed <- lm(y ~ supplier + operator + day, data = df2)
summary(mod2_fixed)
```

```
##
## Call:
## lm(formula = y ~ supplier + operator + day, data = df2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -92.50 -25.94  -6.25  31.88  93.75
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   667.50      62.29  10.716  3.9e-05 ***
## supplierB     167.50      55.72   3.006 0.023812 *
## supplierC     411.25      55.72   7.381 0.000317 ***
## supplierD     291.25      55.72   5.227 0.001962 **
## operatorop2   -35.00      55.72  -0.628 0.553020
## operatorop3   -58.75      55.72  -1.054 0.332266
## operatorop4   -46.25      55.72  -0.830 0.438247
## dayday2       -40.00      55.72  -0.718 0.499782
## dayday3        40.00      55.72   0.718 0.499782
```

```
## dayday4       -40.00      55.72  -0.718 0.499782
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 78.79 on 6 degrees of freedom
## Multiple R-squared:  0.9141, Adjusted R-squared:  0.7853
## F-statistic: 7.094 on 9 and 6 DF,  p-value: 0.01348
```

Supplier is the only significant variable as we expected. Supplier C is the most statistically significant while Supplier B and Supplier D are also highly statistically significant. Supplier C has the most profound effect on the breaking strength as well, it increases breaking strength the most amongst suppliers, based on this regression. The operator or day are not at all significant as expected.

```
par(mfrow = c(2, 2))
plot(mod2_fixed)
```



Residuals vs Fitted does not seem to exhibit a clear pattern. QQ-residuals are not S-shaped so we do not need to worry about normality assumption being violated. Scale-location actually does not seem to show heteroskedasticity. There do not seem to be any influential points based on the Leverage plot.

(c) Fit a mixed effects model with operators and days as random effects but the suppliers as fixed effects. Why is this a natural choice of fixed and random effects? Which supplier results in the highest breaking point? What is the nature of the variation between operators and days?

```
library(lme4)
mod2_mixed <- lmer(y ~ supplier + (1 | operator) + (1 | day), data = df2)
```

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

```
summary(mod2_mixed)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
```

```
## Formula: y ~ supplier + (1 | operator) + (1 | day)
##    Data: df2
##
## REML criterion at convergence: 142.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.8299 -0.4718  0.1027  0.6518  1.4691
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  operator (Intercept)    0.0    0.00
##  day      (Intercept)  219.1   14.80
##  Residual             4990.3   70.64
## Number of obs: 16, groups:  operator, 4; day, 4
##
## Fixed effects:
##             Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)   622.50      36.09  11.94  17.250 8.39e-10 ***
## supplierB     167.50      49.95   9.00   3.353  0.00848 **
## supplierC     411.25      49.95   9.00   8.233 1.76e-05 ***
## supplierD     291.25      49.95   9.00   5.831  0.00025 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) spplrB spplrC
## supplierB -0.692
## supplierC -0.692  0.500
## supplierD -0.692  0.500  0.500
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

This is a natural choice of fixed and random effects because we saw that operator and day had no statistically significant impact on the response; however, they are key ingredients in the process, so we do not want to exclude them but simply treat them as random effects. The model shows that operator-to-operator variability is essentially zero while there is moderate variability between days.

According to the output, Supplier C produces the highest breaking point (622.5 + 411.25) based on this regression, indicating that Supplier C is best in terms of maximizing the breaking strength.

(d) Test the operator and days effects.

```
# Test operator effect
mod2_noOperator <- lmer(y ~ supplier + (1 | day), data = df2, REML = FALSE)
mod2_full <- lmer(y ~ supplier + (1 | operator) + (1 | day), data = df2, REML = FALSE)
```

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

```
anova(mod2_full, mod2_noOperator)
```

```
## Data: df2
## Models:
## mod2_noOperator: y ~ supplier + (1 | day)
## mod2_full: y ~ supplier + (1 | operator) + (1 | day)
##                 npar    AIC    BIC  logLik deviance Chisq Df Pr(>Chisq)
## mod2_noOperator    6 189.69 194.33 -88.847   177.69
```

```
## mod2_full          7 191.69 197.10 -88.847   177.69     0  1          1
```

The ANOVA comparing the full model (with both operator and day effects) and the reduced model (with only the day effect) shows a Chi-square of 0 with 1 degree of freedom (p = 1). This means that dropping the operator random effect has no impact on the model fit, indicating that operator variability is negligible.

```
# Test day effect
mod2_noDay <- lmer(y ~ factor(supplier) + (1 | operator), data = df2, REML = FALSE)
```

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

```
anova(mod2_full, mod2_noDay)
```

```
## Data: df2
## Models:
## mod2_noDay: y ~ factor(supplier) + (1 | operator)
## mod2_full: y ~ supplier + (1 | operator) + (1 | day)
##            npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## mod2_noDay    6 189.74 194.37 -88.867   177.74
## mod2_full     7 191.69 197.10 -88.847   177.69 0.0403  1     0.8409
```

The ANOVA comparing the full model to the one without the day effect yields a Chi-square value of 0.0403 with 1 degree of freedom (p = 0.8409), indicating that dropping the day effect does not significantly worsen the model fit. This means that the day-to-day variation in breaking strength is not statistically significant in this data set.

(e) Test the significance of the supplier effect.

```
mod2_noSupplier <- lmer(y ~ 1 + (1 | operator) + (1 | day), data = df2, REML = FALSE)
```

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

```
anova(mod2_full, mod2_noSupplier)
```

```
## Data: df2
## Models:
## mod2_noSupplier: y ~ 1 + (1 | operator) + (1 | day)
## mod2_full: y ~ supplier + (1 | operator) + (1 | day)
##                 npar    AIC    BIC   logLik deviance Chisq Df Pr(>Chisq)
## mod2_noSupplier    4 216.72 219.81 -104.362   208.72
## mod2_full          7 191.69 197.10  -88.847   177.69 31.03  3  8.377e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

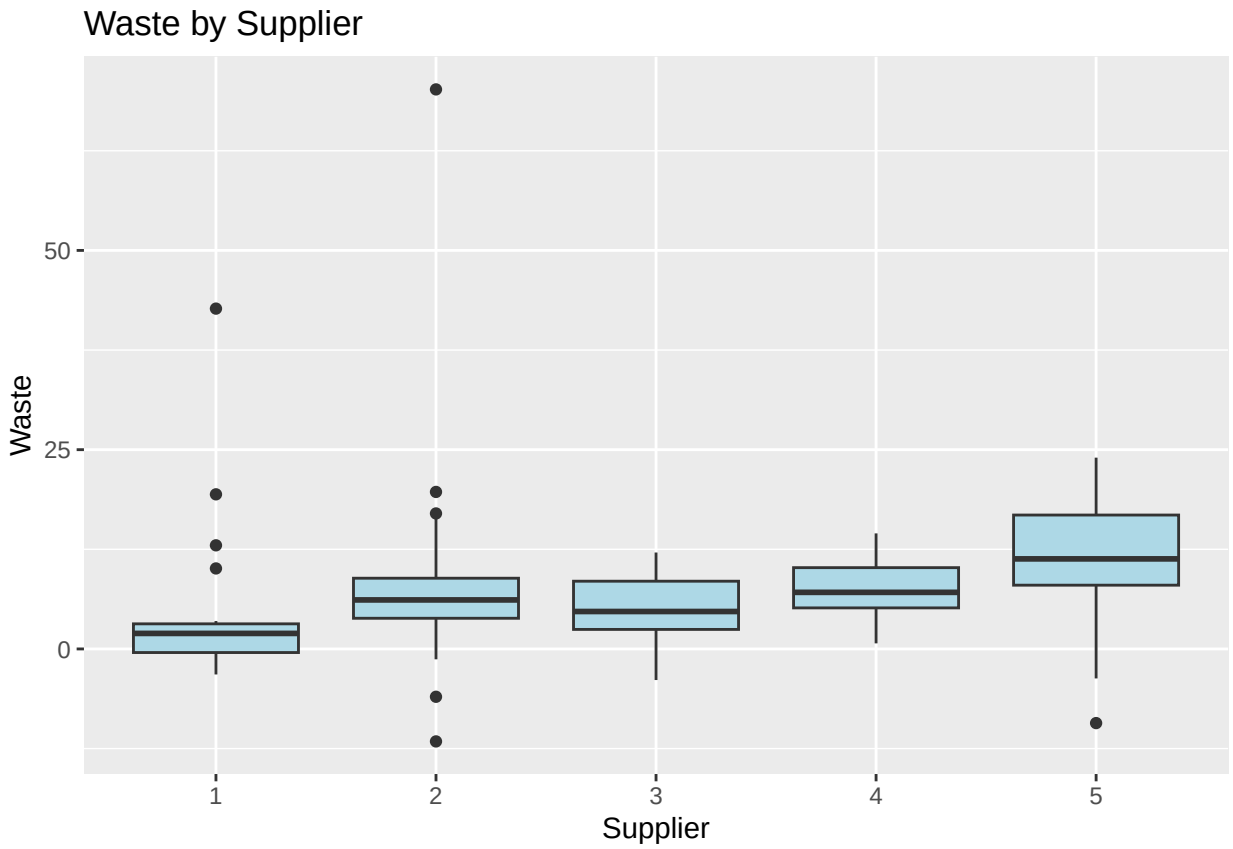
The ANOVA comparing the full model (which includes the supplier effect) to a model without any supplier effect yields a Chi-square value of 31.03 with 3 degrees of freedom and a very small p-value. This highly significant result indicates that the supplier effect is a crucial predictor of the breaking strength. The differences among suppliers are statistically significant and substantially improve model fit.

3. The `denim` dataset concerns the amount of waste in material cutting for a jeans manufacturer due to five suppliers.

a) Plot the data and comment. Fit a one-way ANOVA model using `INLA` with the default prior. Comment on the fit.

```
library(faraway)
df3 <- faraway::denim
# Plot the data: boxplot of waste by supplier
ggplot(df3, aes(x = factor(supplier), y = waste)) +
```

```
  geom_boxplot(fill = "lightblue") +
  labs(x = "Supplier", y = "Waste", title = "Waste by Supplier")
```



There seem to be a lot of outliers in this dataset. Values are dispersed across negative axis as well, so need to consider negative waste as well. The variance within each supplier is quite tight actually, but this is due to outliers not being picked up. The average waste across Suppliers is quite similar, with Supplier 1 noticeably having the lowest average waste and Supplier 5 with the highest average waste. Supplier 5 also has the highest within supplier variance, while Supplier 3 and 4, with no outliers, seem to be the Suppliers with the most consistent waste.

```
library(INLA)
```

```
## This is INLA_24.12.11 built 2024-12-11 20:06:08 UTC.
##  - See www.r-inla.org/contact-us for how to get help.
##  - List available models/likelihoods/etc with inla.list.models()
##  - Use inla.doc(<NAME>) to access documentation
```

```
# Define model formula with supplier as a random effect using an "iid" model:
formula_denim_random <- waste ~ 1 + f(supplier, model = "iid")

# Fit the model with INLA (default priors)
inla_fit_default <- inla(formula_denim_random,
                         family = "gaussian",
                         data = df3,
                         control.compute=list(dic=TRUE))
# Display summary for the default model
summary(inla_fit_default)
```

```
## Time used:
##     Pre = 2.51, Running = 0.953, Post = 0.0425, Total = 3.5
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 6.977 1.012      4.988    6.977      8.966 6.977   0
##
## Random effects:
##   Name     Model
##     supplier IID model
##
## Model hyperparameters:
##                                           mean       sd 0.025quant 0.5quant
## Precision for the Gaussian observations 1.0e-02 2.00e-03      0.008     0.01
## Precision for supplier                  2.2e+04 2.41e+04   1481.810 14489.91
##                                         0.975quant    mode
## Precision for the Gaussian observations   1.40e-02    0.01
## Precision for supplier                    8.59e+04 4047.01
##
## Deviance Information Criterion (DIC) ...............: 707.77
## Deviance Information Criterion (DIC, saturated) ....: 99.12
## Effective number of parameters .....................: 1.86
##
## Marginal log-Likelihood:  -366.67
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

The model fit appears acceptable. Although the DIC is 707.77—which is higher than the saturated model's value of 99.15 (as expected when comparing to a saturated model)—the effective number of parameters is only 1.86, indicating that the model achieves a good balance between simplicity and explanation of the data.

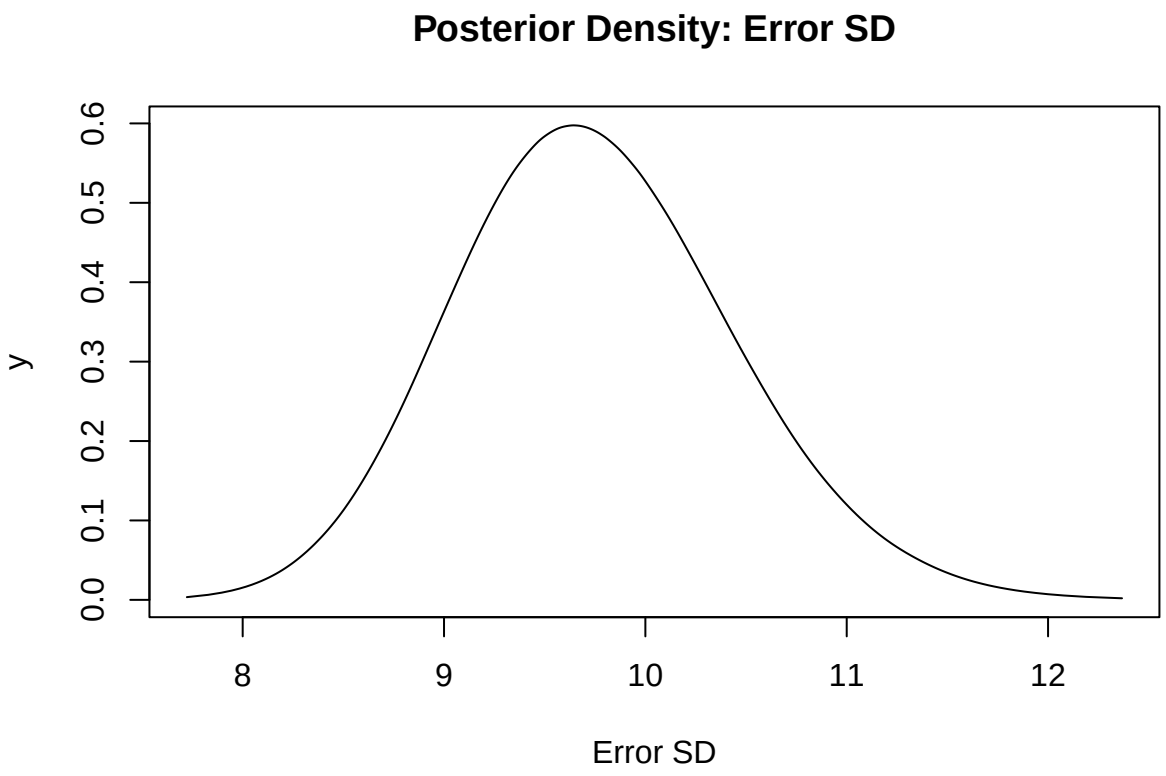
The fixed effect (intercept approx. 6.98) is estimated with good precision, and both the precision for the Gaussian observations (approx. 0.01) and for the supplier random effect are well quantified. Additionally, the marginal log-likelihood of –366.66 and the posterior summaries for the linear predictor and fitted values support that the model is capturing the essential structure in the data without unnecessary complexity. Overall, these diagnostics demonstrate that the model fits the data adequately.

b) Refit the model but with more informative prioris. Make a density plot of the error and supplier SD posterior densities.

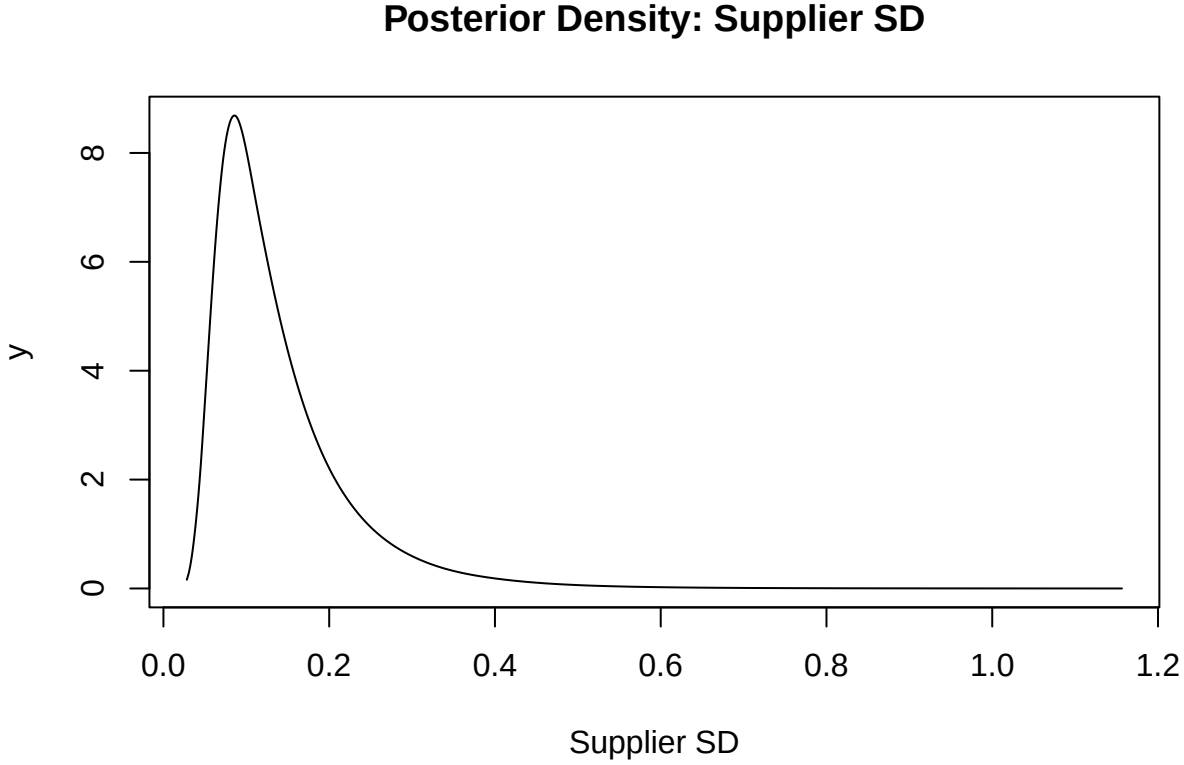
```r
# Specify informative gamma (loggamma) priors for error precision and supplier precision
hyper_error <- list(prec = list(prior = "loggamma", param = c(2, 0.01)))
hyper_supplier <- list(prec = list(prior = "loggamma", param = c(1, 0.01)))

# Update model formula to incorporate the informative prior for supplier's precision
formula_denim_random_info <- waste ~ 1 + f(supplier, model = "iid",
                                           hyper = list(prec = hyper_supplier$prec))

# Fit the model using INLA with the informative prior for the family (error term)
inla_fit_info <- inla(formula_denim_random_info,
                      family = "gaussian",
                      data = df3,
                      control.family = list(hyper = hyper_error),
                      control.compute = list(dic = TRUE))
# Display summary for the informative prior model
summary(inla_fit_info)
```

```
## Time used:
##     Pre = 2.29, Running = 0.82, Post = 0.0261, Total = 3.13
## Fixed effects:
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 6.978 1.006      5.002    6.978      8.955 6.978   0
##
## Random effects:
##   Name     Model
##     supplier IID model
##
## Model hyperparameters:
##                                           mean      sd 0.025quant 0.5quant
## Precision for the Gaussian observations   0.011   0.002      0.008    0.011
## Precision for supplier                  110.056 120.447      7.391   72.327
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.014  0.01
## Precision for supplier                     429.864 20.18
##
## Deviance Information Criterion (DIC) ...............: 707.77
## Deviance Information Criterion (DIC, saturated) ....: 101.13
## Effective number of parameters .....................: 1.88
##
## Marginal log-Likelihood:  -370.53
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

```r
 # In INLA, the hyperpar list usually contains:
#   [1] = error precision
#   [2] = supplier (random effect) precision
marg_error <- inla_fit_info$marginals.hyperpar[[1]]
marg_supplier <- inla_fit_info$marginals.hyperpar[[2]]

# Transform the precisions to standard deviations: SD = 1/sqrt(tau)
marg_sd_error <- inla.tmarginal(function(tau) 1/sqrt(tau), marg_error)
marg_sd_supplier <- inla.tmarginal(function(tau) 1/sqrt(tau), marg_supplier)

# Plot density for error standard deviation
plot(marg_sd_error, type = "l", main = "Posterior Density: Error SD", xlab = "Error SD")
```




```
# Plot density for supplier standard deviation
plot(marg_sd_supplier, type = "l", main = "Posterior Density: Supplier SD", xlab = "Supplier SD")
```



The Posterior Density for the Error SD appears to be normally distributed which is a good sign. The posterior density for the Supplier SD is much more compact, with values concentrated from 0 to 0.2, meaning that the posterior density for the Supplier SD was estimated with pretty good accuracy.

c) Calculate summaries of the posteriors from the model fit.

Posterior Summary for Supplier SD

```
# Summarize posterior for supplier SD
supplier_sd_summary <- inla.zmarginal(marg_sd_supplier)
```

```
## Mean            0.139649
## Stdev           0.0851613
## Quantile  0.025 0.0482468
## Quantile  0.25  0.0835097
## Quantile  0.5   0.116112
## Quantile  0.75  0.169026
## Quantile  0.975 0.364809
```

```
print(supplier_sd_summary)
```

```
## $mean
## [1] 0.1396491
##
## $sd
## [1] 0.08516126
##
## $quant0.025
## [1] 0.04824677
##
## $quant0.25
## [1] 0.08350966
##
## $quant0.5
## [1] 0.1161122
##
## $quant0.75
## [1] 0.1690256
##
## $quant0.975
## [1] 0.364809
```

Posterior Summary for Error SD

```
# Summarize posterior for error SD
error_sd_summary <- inla.zmarginal(marg_sd_error)
```

```
## Mean            9.75016
## Stdev           0.694605
## Quantile  0.025 8.47393
## Quantile  0.25  9.26437
## Quantile  0.5   9.71674
## Quantile  0.75  10.201
## Quantile  0.975 11.2006
```

```
print(error_sd_summary)
```

```
## $mean
## [1] 9.750162
##
## $sd
## [1] 0.6946052
```

```
##
## $quant0.025
## [1] 8.473929
##
## $quant0.25
## [1] 9.264373
##
## $quant0.5
## [1] 9.716745
##
## $quant0.75
## [1] 10.20104
##
## $quant0.975
## [1] 11.20062
```

Fixed Effects Summary

```
# Summarize fixed effects (only intercept in random model)
fixed_summary <- summary(inla_fit_info)$fixed
print(fixed_summary)
```

```
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 6.978 1.006      5.002    6.978      8.955 6.978   0
```

Hyperparameters Effects Summary

```
# Summarize hyperparameters
hyper_summary <- summary(inla_fit_info)$hyperpar
print(hyper_summary)
```

```
##                                           mean      sd 0.025quant 0.5quant
## Precision for the Gaussian observations   0.011   0.002      0.008    0.011
## Precision for supplier                  110.056 120.447      7.391   72.327
##                                         0.975quant   mode
## Precision for the Gaussian observations      0.014  0.010
## Precision for supplier                     429.864 20.178
```

d) Report 95% credible intervals for the SDs using the summary output. Compute the posterio modes for the error and supplier SDs and compare these to the posterior means.

```
# Calculate the posterior mean for the error SD from its transformed marginal density
mean_error_sd <- inla.emarginal(function(x) x, marg_sd_error)

# Calculate the posterior mean for the supplier SD
mean_supplier_sd <- inla.emarginal(function(x) x, marg_sd_supplier)

# Calculate 95% credible intervals for the error SD
cred_error <- inla.qmarginal(c(0.025, 0.975), marg_sd_error)

# Calculate 95% credible intervals for the supplier SD
cred_supplier <- inla.qmarginal(c(0.025, 0.975), marg_sd_supplier)

# Print the summaries for Error SD
cat("Posterior Summary for Error SD:\n",
    "Mean =", mean_error_sd, "\n95% Credible Interval =", cred_error, "\n\n")
```

```
## Posterior Summary for Error SD:
```

```
##  Mean = 9.750162
## 95% Credible Interval = 8.473929 11.20062
```

```
# Print the summaries for Supplier SD
cat("Posterior Summary for Supplier SD:\n",
    "Mean =", mean_supplier_sd, "\n95% Credible Interval =", cred_supplier, "\n")
```

```
## Posterior Summary for Supplier SD:
##  Mean = 0.1396491
## 95% Credible Interval = 0.04824677 0.364809
```

```
# Compute the posterior mode for the error SD
mode_error_sd <- inla.mmarginal(marg_sd_error)

# Compute the posterior mode for the supplier SD
mode_supplier_sd <- inla.mmarginal(marg_sd_supplier)

# Print the posterior modes
cat("Posterior Mode for Error SD =", mode_error_sd, "\n")
```

```
## Posterior Mode for Error SD = 9.64308
```

```
cat("Posterior Mode for Supplier SD =", mode_supplier_sd, "\n")
```

```
## Posterior Mode for Supplier SD = 0.08567067
```

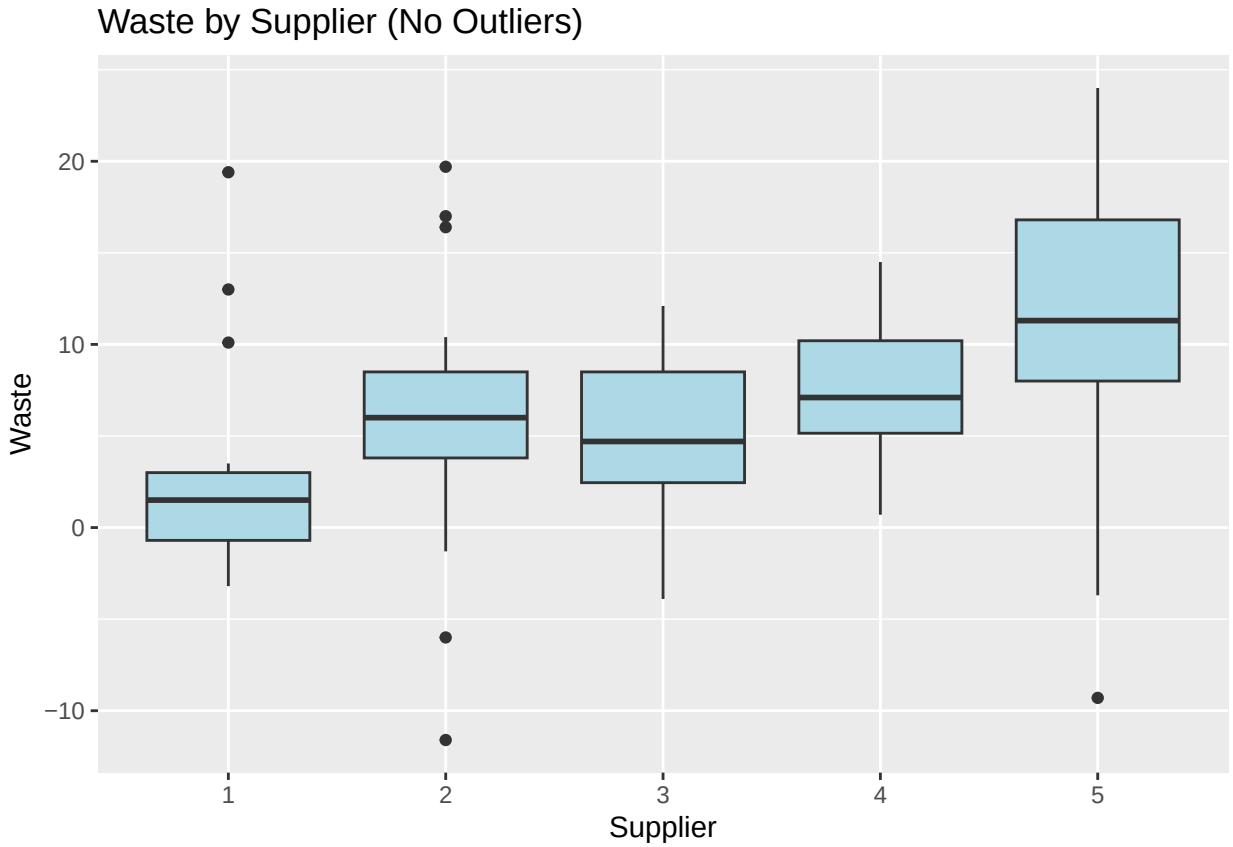
The posterior modes for both the error and supplier SDs lie within their respective 95% credible intervals, indicating that the estimation procedure is robust and produces results as expected.

e) Remove the two outliers from the data and repeat the analysis. Comment on the any interesting differences.

```
# Remove the two observations with the highest waste values
df3_sorted <- df3[order(df3$waste), ]
df3_no_outliers <- df3_sorted[1:(nrow(df3_sorted) - 2), ]

# Report the new number of observations
cat("Number of observations after removing outliers:", nrow(df3_no_outliers), "\n")
```

```
## Number of observations after removing outliers: 93
```

```
library(ggplot2)
# Create a boxplot for the outlier-free dataset
ggplot(df3_no_outliers, aes(x = factor(supplier), y = waste)) +
  geom_boxplot(fill = "lightblue") +
  labs(x = "Supplier", y = "Waste", title = "Waste by Supplier (No Outliers)")
```



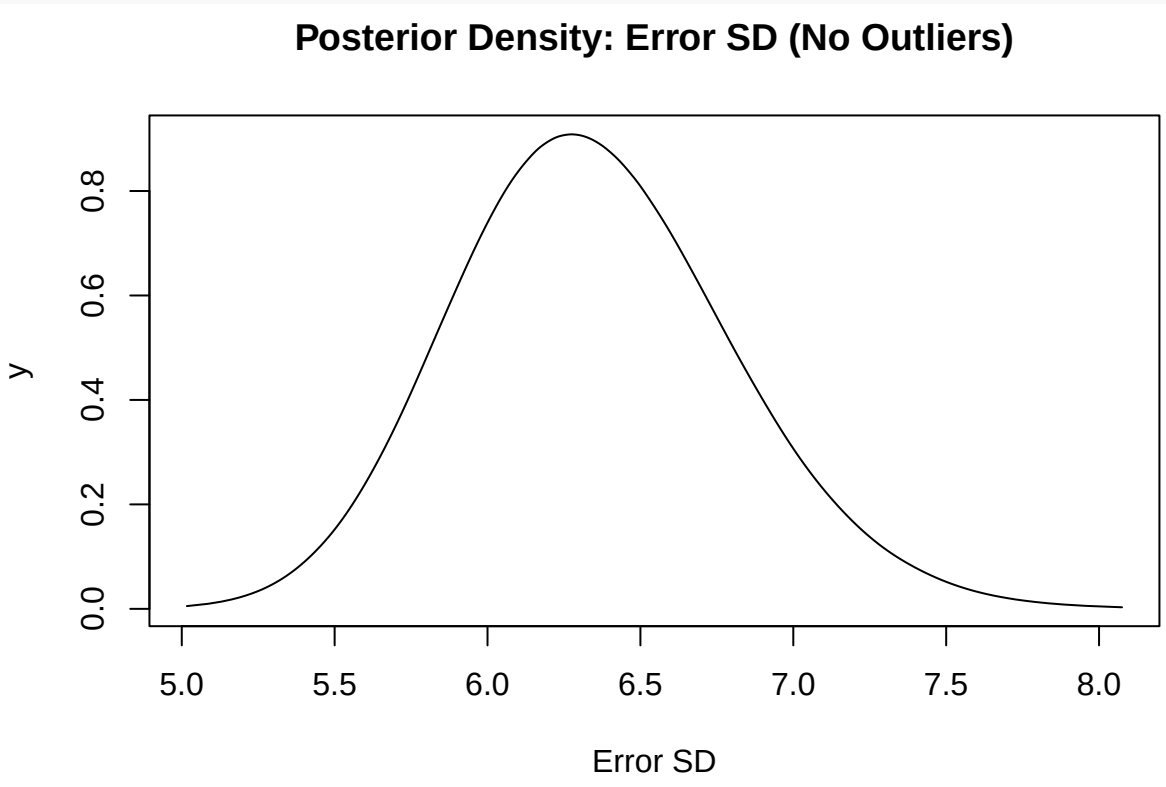
The box plot distribution looks much better than before. The variance within suppliers and across suppliers is much more pronounced. It is more clear how the data is dispersed based on this graph after removing the two outliers. It is quite clear that Supplier 5 has an alarmingly large dispersion across its Waste levels over time after removing outliers. Supplier 3 and 4 still show a compact waste distribution, unaffected since neither had any outliers. The removal of 2 outliers did not have a pronounced effect on Supplier 1 or Supplier 2's box plot because their outliers were not effectively removed.

```
# Refit the model with informative priors using the modified dataset
inla_fit_info_no <- inla(formula_denim_random_info,
                         family = "gaussian",
                         data = df3_no_outliers,
                         control.family = list(hyper = hyper_error),
                         control.compute = list(dic = TRUE))

# Print summary of the model without outliers
print(summary(inla_fit_info_no))
```

```
## Time used:
##     Pre = 2.25, Running = 0.76, Post = 0.0252, Total = 3.04
## Fixed effects:
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 5.924 0.674      4.603    5.923      7.251 5.923   0
##
## Random effects:
##   Name     Model
##     supplier IID model
##
## Model hyperparameters:
```

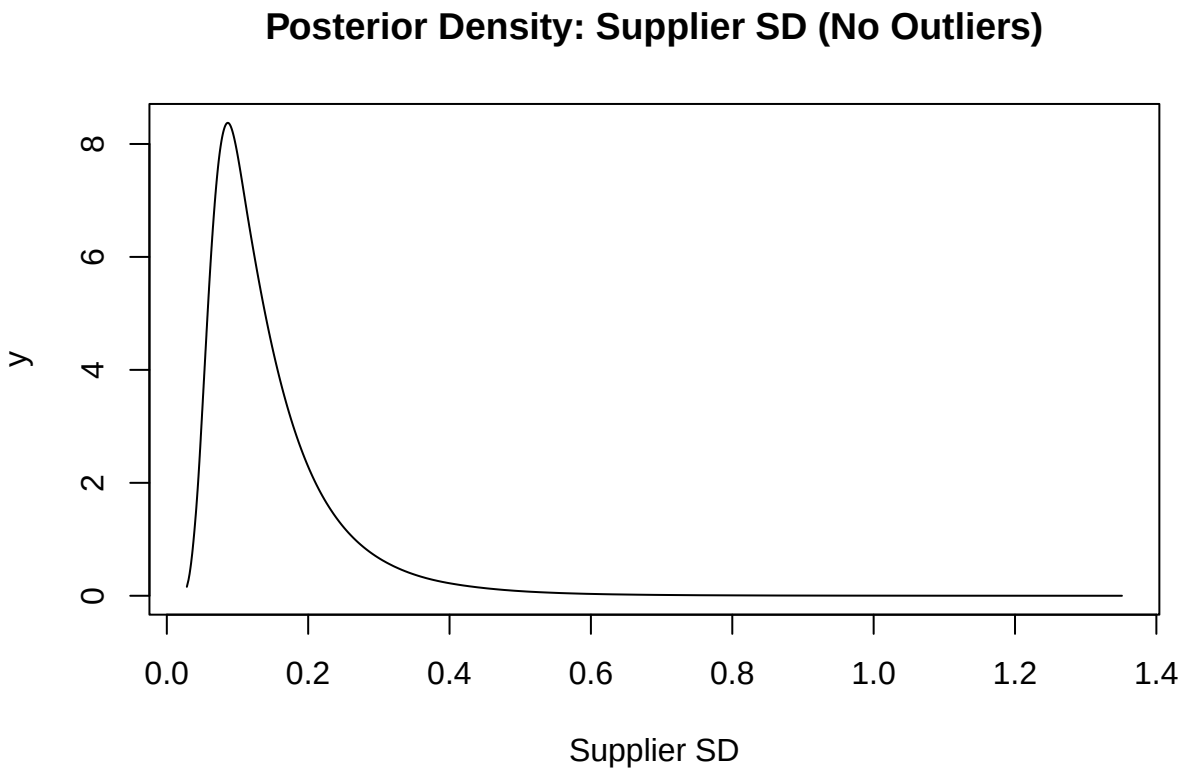
```
##                                           mean      sd 0.025quant 0.5quant
## Precision for the Gaussian observations   0.025   0.004      0.019    0.025
## Precision for supplier                  106.380 119.029      6.448   68.656
##                                         0.975quant   mode
## Precision for the Gaussian observations      0.033  0.025
## Precision for supplier                     422.259 17.252
##
## Deviance Information Criterion (DIC) ...............: 612.77
## Deviance Information Criterion (DIC, saturated) ....: 99.30
## Effective number of parameters .....................: 2.07
##
## Marginal log-Likelihood:  -321.91
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

```r
# Extract the hyperparameter marginals for the outlier-free model
marg_error_no    <- inla_fit_info_no$marginals.hyperpar[[1]]
marg_supplier_no <- inla_fit_info_no$marginals.hyperpar[[2]]

# Transform these to standard deviations
marg_sd_error_no <- inla.tmarginal(function(tau) 1/sqrt(tau), marg_error_no)
marg_sd_supplier_no <- inla.tmarginal(function(tau) 1/sqrt(tau), marg_supplier_no)

# Plot the posterior density of error SD (No Outliers)
plot(marg_sd_error_no, type = "l",
     main = "Posterior Density: Error SD (No Outliers)", xlab = "Error SD")
```




```r
# Plot the posterior density of supplier SD (No Outliers)
plot(marg_sd_supplier_no, type = "l",
```

```
     main = "Posterior Density: Supplier SD (No Outliers)", xlab = "Supplier SD")
```



After removing the outliers, the posterior density of the error standard deviation shifts considerably—it is now centered around 6 instead of 10. This change suggests that the outliers were inflating the error variability estimate.

In contrast, the posterior density for the supplier standard deviation remains largely unchanged by the removal of the outliers, indicating robust estimation across models.

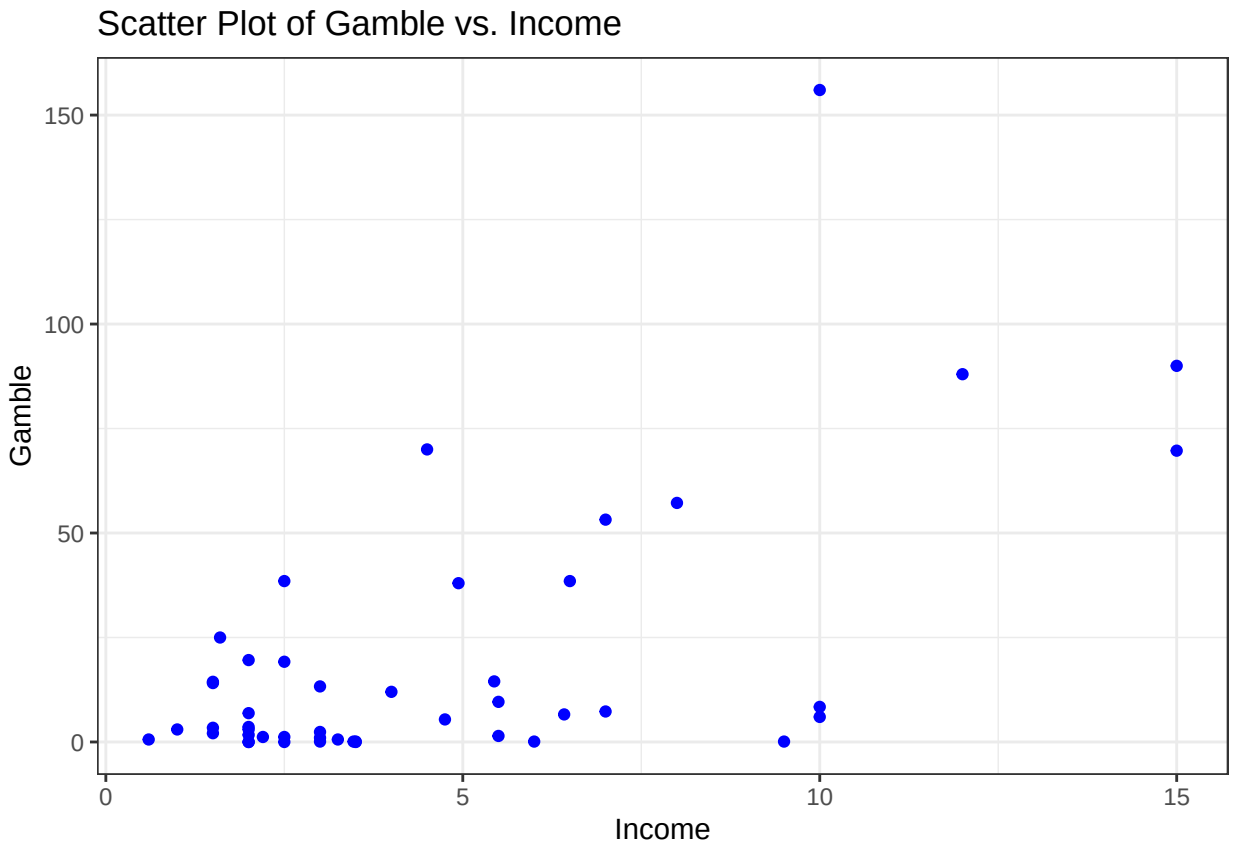
Although the summary statistics (mean, standard deviation, and mode) for both parameters differ between models, the overall shape of the supplier SD's posterior density is consistent while the error SD's density shows a marked decrease.

This demonstrates that outliers predominantly affect the error component rather than the variability between suppliers.

4. The dataset `teengamb` concerns a study of teenage gambling in Britain. Take the variables gamble as the response and income as the predictor.

a) Make a plot of the data.

```
library(faraway)
df4 <- faraway::teengamb
# Scatter plot of gamble vs. income
ggplot(df4, aes(x = income, y = gamble)) +
  geom_point(color = "blue") +
  labs(x = "Income", y = "Gamble", title = "Scatter Plot of Gamble vs. Income") +
  theme_bw()
```



b) Fit a curve to the data using kernel smoothing with a cross-validated choice of bandwidth. Display the fit on the data. Does the fit look linear?

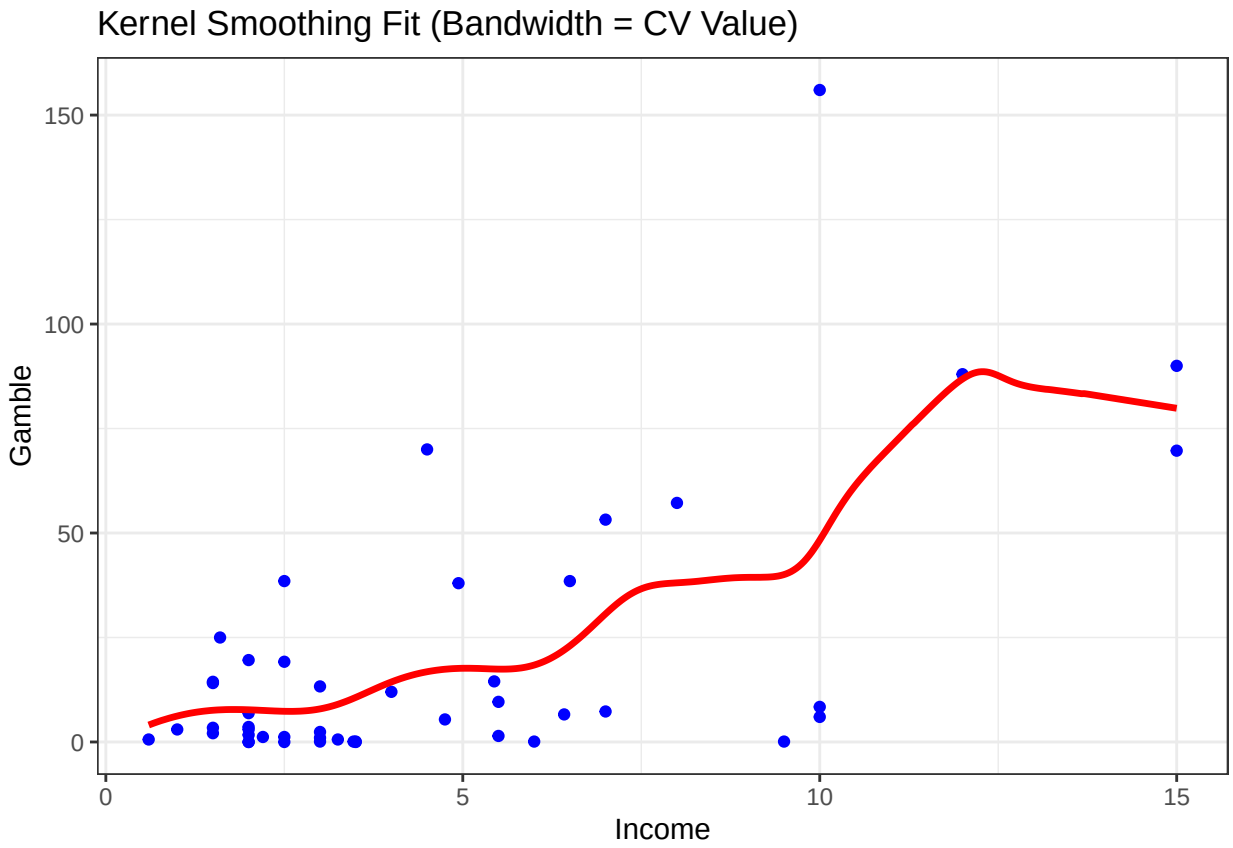
```
# Load KernSmooth for kernel smoothing functions
library(KernSmooth)
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
# Determine a cross-validated bandwidth for income
h_cv <- dpik(df4$income)
cat("Cross-validated bandwidth =", h_cv, "\n")
```

```
## Cross-validated bandwidth = 0.9245437
```

```
# Compute kernel smoothing fit using locpoly
kernel_fit <- locpoly(x = df4$income, y = df4$gamble, degree = 1, bandwidth = h_cv)

# Plot the data and add the kernel smoothing fit
ggplot(df4, aes(x = income, y = gamble)) +
  geom_point(color = "blue") +
  geom_line(data = as.data.frame(kernel_fit), aes(x = x, y = y), color = "red", linewidth = 1.2) +
  labs(x = "Income", y = "Gamble", title = "Kernel Smoothing Fit (Bandwidth = CV Value)") +
  theme_bw()
```



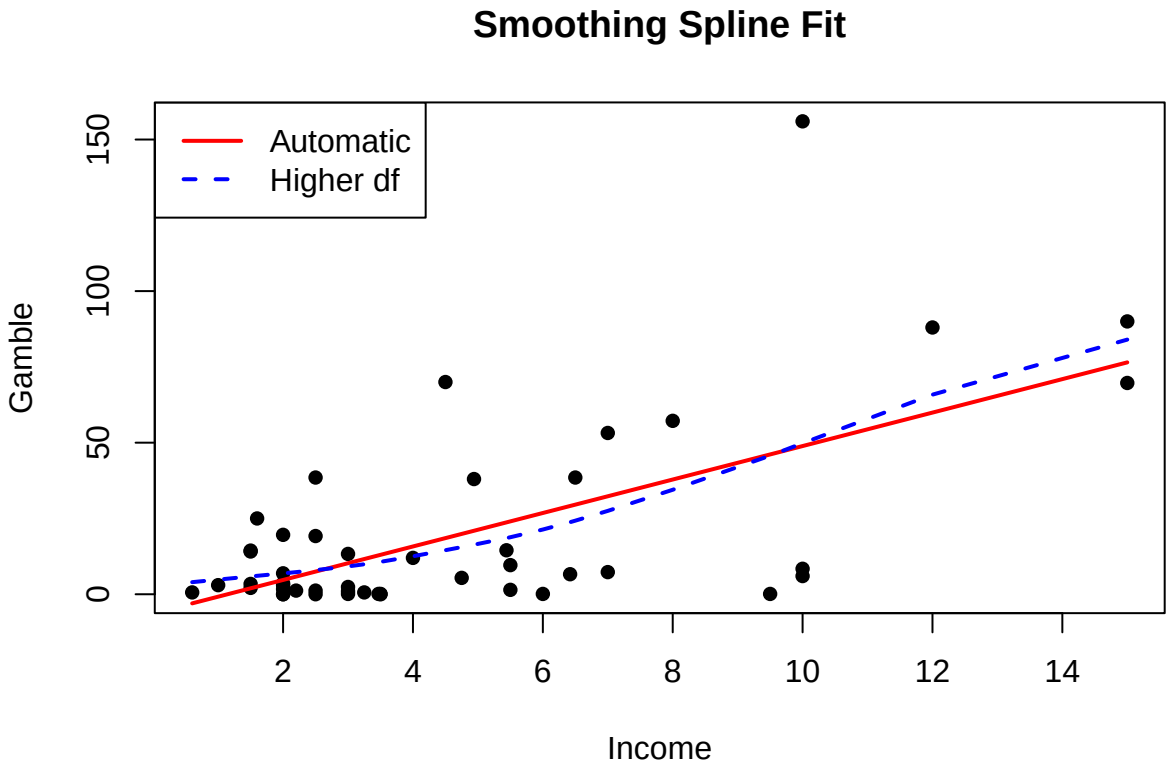
The fit is somewhat linear, but it is clearly locally adaptive, adjusting itself dramatically after an income above 10.

c) Fit a curve using smoothing splines with the automatically chosen amount of smoothing. Display the fit and report the effective degrees of freedom. Dis- play a fit with somewhat larger degrees of freedom. Was the automatic choice satisfactory?

```
# Fit a smoothing spline with the automatically chosen smoothing parameter
ss_fit <- smooth.spline(x = df4$income, y = df4$gamble)
cat("Effective degrees of freedom (automatic):", ss_fit$df, "\n")
```

```
## Effective degrees of freedom (automatic): 2.000119
```

```
# Plot the data and add the smoothing spline fit
plot(df4$income, df4$gamble, pch = 16, main = "Smoothing Spline Fit", xlab = "Income", ylab = "Gamble")
lines(ss_fit, col = "red", lwd = 2)

# Fit a smoothing spline with increased degrees of freedom (e.g., add 2 df)
ss_fit2 <- smooth.spline(x = df4$income, y = df4$gamble, df = ss_fit$df + 2)
lines(ss_fit2, col = "blue", lwd = 2, lty = 2)
legend("topleft", legend = c("Automatic", "Higher df"),
       col = c("red", "blue"), lwd = 2, lty = c(1,2))
```

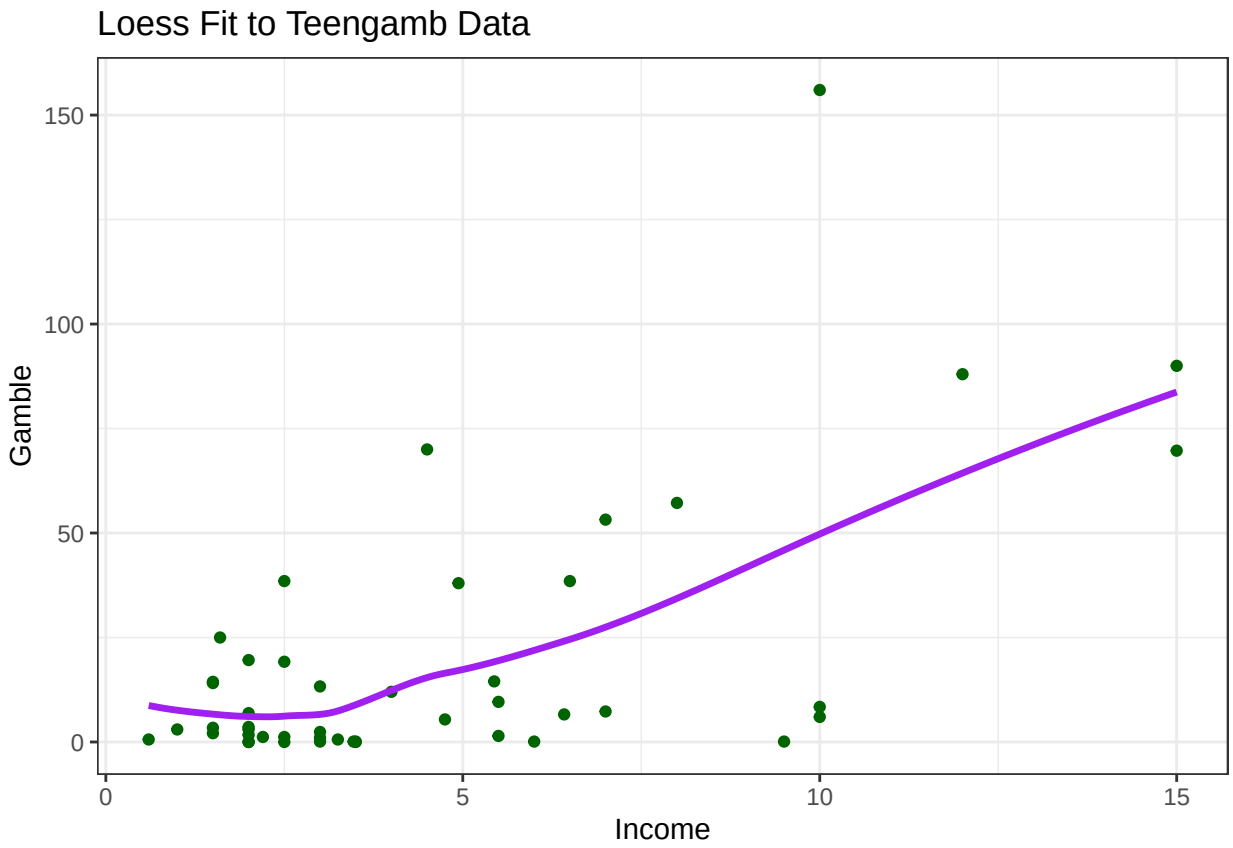


The automatic fit was not bad, it is definitely satisfactory, but the higher df fit was better as expected. However, one must consider if the additional precision is valuable or if it causes the model to be more prone to over-fitting.

d) Use loess to fit a curve to the data with the default amount of smoothing. Display the fit.

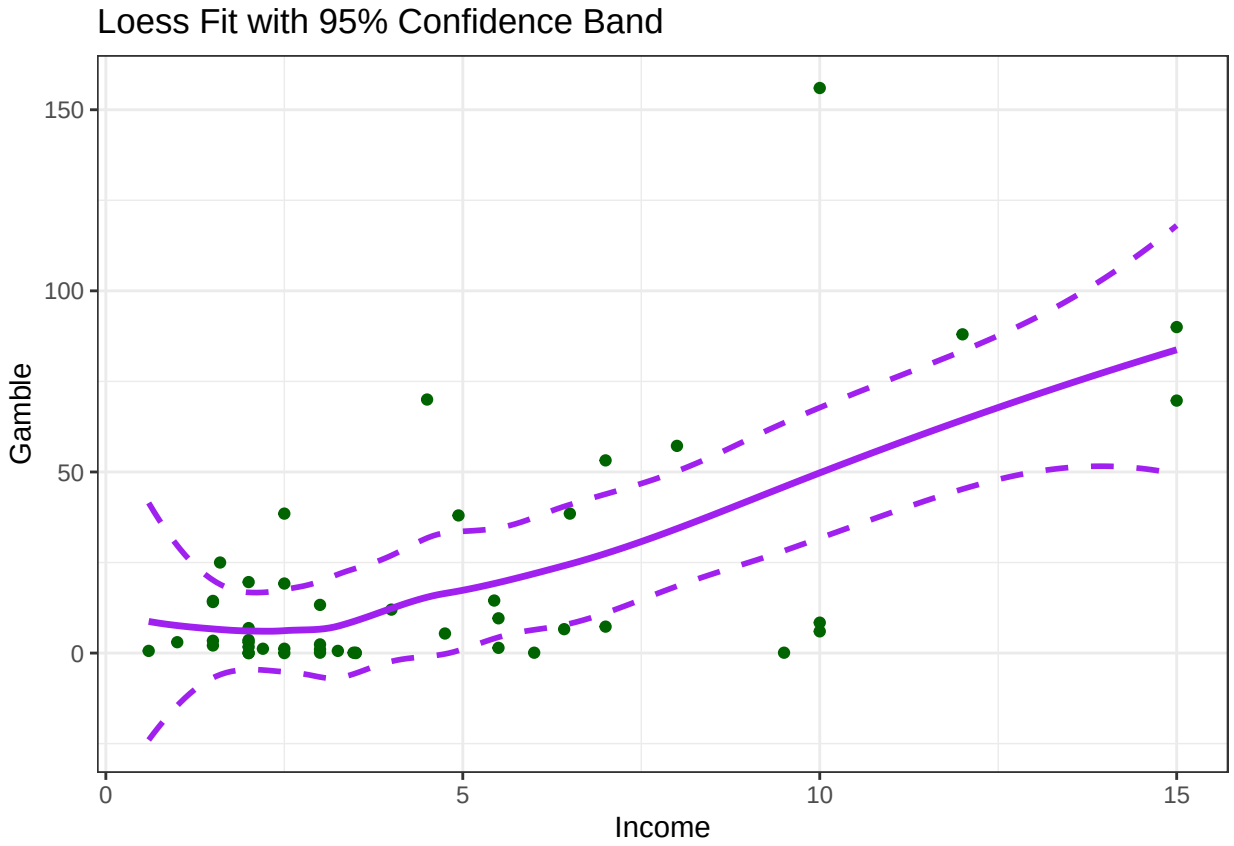
```
# Fit a loess model using the default smoothing parameters
loess_fit <- loess(gamble ~ income, data = df4)

# Create a fine grid for the predictor
income_grid <- seq(min(df4$income), max(df4$income), length.out = 200)

# Obtain loess predictions on the grid
loess_pred <- predict(loess_fit, newdata = data.frame(income = income_grid))

# Prepare a data frame for plotting the loess fit
df_loess <- data.frame(income = income_grid, gamble_fit = loess_pred)

# Plot the scatter and add the loess curve
ggplot(df4, aes(x = income, y = gamble)) +
  geom_point(color = "darkgreen") +
  geom_line(data = df_loess, aes(x = income, y = gamble_fit), color = "purple", size = 1.2) +
  labs(x = "Income", y = "Gamble", title = "Loess Fit to Teengamb Data") +
  theme_bw()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



e) Produce and show a plot with a 95% confidence band for the fit. Is a linear fit plausible?

```r
# Get predictions along with standard errors using predict(loess, se = TRUE)
loess_pred <- predict(loess_fit, newdata = data.frame(income = income_grid), se = TRUE)

# Calculate the upper and lower 95% confidence bands (using approx. 1.96*SE)
upper <- loess_pred$fit + 1.96 * loess_pred$se.fit
lower <- loess_pred$fit - 1.96 * loess_pred$se.fit

# Create a data frame with the income grid, fitted values, and confidence bands
df_band <- data.frame(income = income_grid, fit = loess_pred$fit, upper = upper, lower = lower)

# Plot the data, loess fit, and confidence band using ggplot2
ggplot(df4, aes(x = income, y = gamble)) +
  geom_point(color = "darkgreen") +
  geom_line(data = df_band, aes(x = income, y = fit), color = "purple", size = 1.2) +
  geom_line(data = df_band, aes(x = income, y = upper), color = "purple", linetype = "dashed", size = 1
  geom_line(data = df_band, aes(x = income, y = lower), color = "purple", linetype = "dashed", size = 1
  labs(x = "Income", y = "Gamble", title = "Loess Fit with 95% Confidence Band") +
  theme_bw()
```



Yes, that's definitely plausible. The Loess curve appears quite linear overall for this data. It's smarter than a regular regression because it adapts to the local density of points—capturing the concentration at lower income levels and adjusting as income increases and data become sparser, which makes it locally adaptive.