optimal_difficulty_lmer_analysis

Erva

2024-04-26

Packages

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
## Warning: package 'progress' was built under R version 4.2.3
```

```
## Loading required package: viridisLite
```

```
## Warning: package 'viridisLite' was built under R version 4.2.3
```

```
## Warning: package 'ggforce' was built under R version 4.2.3
```

```
## Warning: package 'see' was built under R version 4.2.3
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
##
## Attaching package: 'gridExtra'
```

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

```
## 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
```

```
## Warning: package 'sjPlot' was built under R version 4.2.3
```

```
## Install package "strengejacke" from GitHub (`devtools::install_github("strengejacke/streng
ejacke")`) to load all sj-packages at once!
```

```
## Warning: package 'tibble' was built under R version 4.2.3
```

```
## corrplot 0.92 loaded
```

```
## Warning: package 'lmtest' was built under R version 4.2.3
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:data.table':
##
##     yearmon, yearqtr
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'parameters'
```

```
## The following objects are masked from 'package:moments':
##
##     kurtosis, skewness
```

```
##
## Attaching package: 'purrr'
```

```
## The following object is masked from 'package:data.table':
##
##     transpose
```

read file

```
file= "average_quest_dif_100.csv"
# read the file
average_quest_dif <- read.csv(file)

  name_file <- substr(file, 18, nchar(file))
```

rename variables to make them transparent

```
# rename the column
average_quest_dif <- average_quest_dif %>%
  rename(mean_relative_difficulty = mean_skill_difficulty_difference)

# rename the column
average_quest_dif <- average_quest_dif %>%
  rename(student_ability = ability_2020_2021)

# rename the column
average_quest_dif <- average_quest_dif %>%
  rename(relative_difficulty_slope = slope)
```

mean- center interaction variables (only the n_question_in_spec_training)

```
# Calculate the mean of n_question_in_spec_training
mean_n_question <- mean(average_quest_dif$n_question_in_spec_training, na.rm = TRUE)

# Center the variable n_question_in_spec_training by subtracting its mean from each observati
on
average_quest_dif$n_question_in_spec_training_c <- average_quest_dif$n_question_in_spec_train
ing - mean_n_question
```

correaltions

correlation matrix

```r
# Selecting the columns
for_table <- average_quest_dif[c('mean_relative_difficulty', 'student_ability', 'prop_correct
_ecn', 'mean_student_average_ability', 'n_question_in_spec_training', 'mean_difficulty','mean
_elo_ExpectedScore','n_question_in_spec_training_c')]


# Calculating means and standard deviations
means <- colMeans(for_table, na.rm = TRUE)
sds <- apply(for_table, 2, sd, na.rm = TRUE)

# Correlation matrix
correlations <- cor(for_table, use = 'pairwise.complete.obs')

# Create a formatted table with means, SDs, and correlations
formatted_table <- data.frame(M = means, SD = sds, correlations)

# Function to determine significance asterisks
significance_asterisks <- function(p_value) {
  if (p_value < 0.001) {
    return('***')
  } else if (p_value < 0.01) {
    return('**')
  } else if (p_value < 0.05) {
    return('*')
  } else {
    return('')
  }
}

# Populate the correlations with significance testing
for (col in colnames(for_table)) {
  for (row in colnames(for_table)) {
    if (col == row) {
      # Diagonals are not displayed in the table
      formatted_table[row, col] <- '—'
    } else {
      # Calculate the p-value
      p_value <- Hmisc::rcorr(as.matrix(for_table[, c(row, col)]))$P[1, 2]
      # Format with two decimal places and add significance asterisks
      corr <- correlations[row, col]
      formatted_table[row, col] <- paste0(format(corr, digits = 2), significance_asterisks(p_
value))
    }
  }
}

# You can save this table to a CSV or Excel file, or print it out
write.csv(formatted_table, file = 'formatted_correlation_table_with_significance.csv', row.na
mes = TRUE)

formatted_table
```

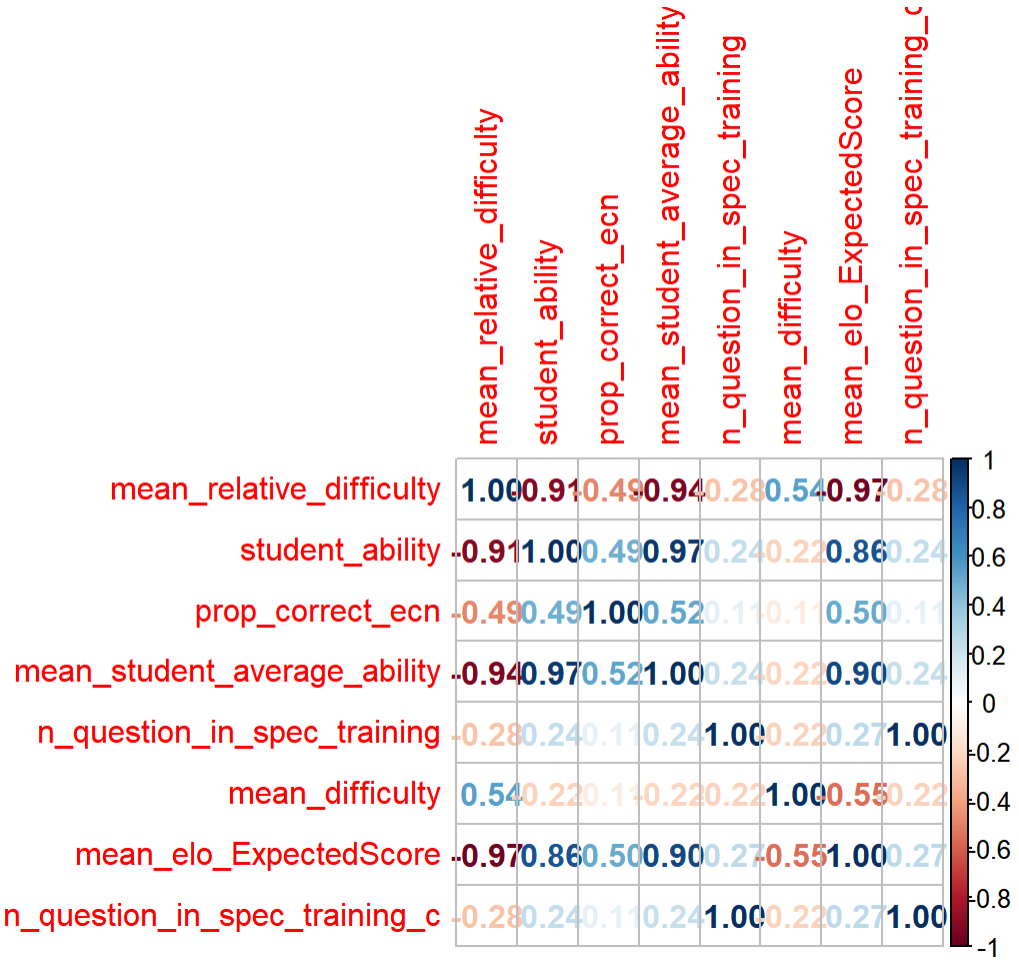
```
##                                        M           SD
## mean_relative_difficulty      -9.386030e-02   0.42899225
## student_ability                6.525809e-02   0.41442195
## prop_correct_ecn               4.519341e-01   0.13004373
## mean_student_average_ability   5.815021e-02   0.36843310
## n_question_in_spec_training    2.807386e+02 236.05363943
## mean_difficulty               -3.571009e-02   0.15409376
## mean_elo_ExpectedScore         4.953906e-01   0.08005331
## n_question_in_spec_training_c -4.343700e-15 236.05363943
##                               mean_relative_difficulty student_ability
## mean_relative_difficulty                             —        -0.91***
## student_ability                                -0.91***               —
## prop_correct_ecn                               -0.49***         0.49***
## mean_student_average_ability                   -0.94***         0.97***
## n_question_in_spec_training                    -0.28***         0.24***
## mean_difficulty                                 0.54***        -0.22***
## mean_elo_ExpectedScore                         -0.97***         0.86***
## n_question_in_spec_training_c                  -0.28***         0.24***
##                               prop_correct_ecn mean_student_average_ability
## mean_relative_difficulty             -0.49***                      -0.94***
## student_ability                       0.49***                       0.97***
## prop_correct_ecn                            —                       0.52***
## mean_student_average_ability          0.52***                             —
## n_question_in_spec_training           0.11***                       0.24***
## mean_difficulty                      -0.11***                      -0.22***
## mean_elo_ExpectedScore                 0.5***                        0.9***
## n_question_in_spec_training_c         0.11***                       0.24***
##                               n_question_in_spec_training mean_difficulty
## mean_relative_difficulty                       -0.28***         0.54***
## student_ability                                 0.24***        -0.22***
## prop_correct_ecn                                0.11***        -0.11***
## mean_student_average_ability                    0.24***        -0.22***
## n_question_in_spec_training                           —        -0.22***
## mean_difficulty                                -0.22***               —
## mean_elo_ExpectedScore                          0.27***        -0.55***
## n_question_in_spec_training_c                      1***        -0.22***
##                               mean_elo_ExpectedScore
## mean_relative_difficulty                  -0.97***
## student_ability                            0.86***
## prop_correct_ecn                             0.5***
## mean_student_average_ability               0.9***
## n_question_in_spec_training               0.27***
## mean_difficulty                           -0.55***
## mean_elo_ExpectedScore                           —
## n_question_in_spec_training_c             0.27***
##                               n_question_in_spec_training_c
## mean_relative_difficulty                         -0.28***
## student_ability                                   0.24***
## prop_correct_ecn                                  0.11***
## mean_student_average_ability                      0.24***
```

```
## n_question_in_spec_training                          1***
## mean_difficulty                                   -0.22***
## mean_elo_ExpectedScore                             0.27***
## n_question_in_spec_training_c                           –
```

```r
# Calculating the correlation matrix
correlation_matrix <- cor(for_table, use = "pairwise.complete.obs")

# Print the correlation matrix
print(correlation_matrix)
```

```
##                               mean_relative_difficulty student_ability
## mean_relative_difficulty                     1.0000000      -0.9110400
## student_ability                             -0.9110400       1.0000000
## prop_correct_ecn                            -0.4886090       0.4866384
## mean_student_average_ability                -0.9364858       0.9683204
## n_question_in_spec_training                 -0.2833501       0.2434495
## mean_difficulty                              0.5448624      -0.2210849
## mean_elo_ExpectedScore                      -0.9745705       0.8611216
## n_question_in_spec_training_c               -0.2833501       0.2434495
##                               prop_correct_ecn mean_student_average_ability
## mean_relative_difficulty           -0.4886090                   -0.9364858
## student_ability                     0.4866384                    0.9683204
## prop_correct_ecn                    1.0000000                    0.5247728
## mean_student_average_ability        0.5247728                    1.0000000
## n_question_in_spec_training         0.1086672                    0.2367446
## mean_difficulty                    -0.1055578                   -0.2161803
## mean_elo_ExpectedScore              0.5005724                    0.9026946
## n_question_in_spec_training_c       0.1086672                    0.2367446
##                               n_question_in_spec_training mean_difficulty
## mean_relative_difficulty                       -0.2833501       0.5448624
## student_ability                                 0.2434495      -0.2210849
## prop_correct_ecn                                0.1086672      -0.1055578
## mean_student_average_ability                    0.2367446      -0.2161803
## n_question_in_spec_training                     1.0000000      -0.2227895
## mean_difficulty                                -0.2227895       1.0000000
## mean_elo_ExpectedScore                          0.2691582      -0.5548609
## n_question_in_spec_training_c                   1.0000000      -0.2227895
##                               mean_elo_ExpectedScore
## mean_relative_difficulty                  -0.9745705
## student_ability                            0.8611216
## prop_correct_ecn                           0.5005724
## mean_student_average_ability               0.9026946
## n_question_in_spec_training                0.2691582
## mean_difficulty                           -0.5548609
## mean_elo_ExpectedScore                     1.0000000
## n_question_in_spec_training_c              0.2691582
##                               n_question_in_spec_training_c
## mean_relative_difficulty                        -0.2833501
## student_ability                                  0.2434495
## prop_correct_ecn                                 0.1086672
## mean_student_average_ability                     0.2367446
## n_question_in_spec_training                      1.0000000
## mean_difficulty                                 -0.2227895
## mean_elo_ExpectedScore                           0.2691582
## n_question_in_spec_training_c                    1.0000000
```

```r
# visiulazi
# Compute correlation matrix
correlation_matrix <- cor(for_table)

# Visualize the correlation matrix
library(corrplot)
corrplot(correlation_matrix, method = "number")
```



Full Model & Colinearity check : specialty as radnom intercept and slope

```
model_formula <- prop_correct_ecn ~ I(mean_relative_difficulty^2) + mean_relative_difficulty
+
  n_question_in_spec_training_c+
  student_ability+
  student_ability:I(mean_relative_difficulty^2)+
  student_ability:mean_relative_difficulty+
  student_ability:n_question_in_spec_training_c+
  (1 | student)+
  (1 + mean_relative_difficulty + I(mean_relative_difficulty^2) | specialty)

# Fit the linear mixed-effects model
model_fit <- lmer(model_formula, data = average_quest_dif, control = lmerControl(optimizer =
"bobyqa"))

# Display the model summary with the name of the file
cat(paste("Model summary for", name_file, ":\n"))
```

```
## Model summary for _100.csv :
```

```
print(summary(model_fit))
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: model_formula
##    Data: average_quest_dif
## Control: lmerControl(optimizer = "bobyqa")
##
## REML criterion at convergence: -96571.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.6931 -0.5915 -0.0035  0.5900  7.3607
##
## Random effects:
##  Groups    Name                          Variance  Std.Dev. Corr
##  student   (Intercept)                   3.741e-03 0.061164
##  specialty (Intercept)                   5.696e-03 0.075474
##            mean_relative_difficulty      3.336e-04 0.018266 -0.47
##            I(mean_relative_difficulty^2) 6.338e-05 0.007961 -0.88  0.30
##  Residual                                5.491e-03 0.074103
## Number of obs: 45436, groups:  student, 6451; specialty, 13
##
## Fixed effects:
##                                              Estimate Std. Error         df
## (Intercept)                                  4.575e-01  2.096e-02  1.204e+01
## I(mean_relative_difficulty^2)               -2.677e-02  5.687e-03  2.348e+02
## mean_relative_difficulty                    -4.894e-02  5.787e-03  1.892e+01
## n_question_in_spec_training_c                2.529e-05  2.642e-06  4.452e+04
## student_ability                              4.671e-02  2.630e-03  4.211e+04
## I(mean_relative_difficulty^2):student_ability -8.077e-03  2.008e-03  6.642e+02
## mean_relative_difficulty:student_ability     -2.037e-02  5.589e-03  3.191e+04
## n_question_in_spec_training_c:student_ability -6.998e-06  4.836e-06  2.567e+04
##                                             t value Pr(>|t|)
## (Intercept)                                  21.833 4.72e-11 ***
## I(mean_relative_difficulty^2)                -4.707 4.30e-06 ***
## mean_relative_difficulty                     -8.457 7.50e-08 ***
## n_question_in_spec_training_c                 9.573  < 2e-16 ***
## student_ability                              17.762  < 2e-16 ***
## I(mean_relative_difficulty^2):student_ability  -4.022 6.43e-05 ***
## mean_relative_difficulty:student_ability      -3.645 0.000268 ***
## n_question_in_spec_training_c:student_ability  -1.447 0.147854
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) I(m__^2) mn_rl_ n_q____ stdnt_ I(__^2): mn__:_
## I(mn_rl_^2) -0.347
## mn_rltv_dff -0.406  0.145
## n_qstn_n___  0.011  0.013    0.064
## studnt_blty  0.002  0.033    0.361 -0.031
## I(mn__^2):_  0.006 -0.003    0.105  0.040  -0.157
## mn_rltv_d:_  0.000  0.835    0.052  0.000   0.037  0.137
```

```
## n_qst____:_ -0.004 -0.020   -0.040 -0.487   0.059 -0.050    0.077
```

```
# Create a table of fixed effects
tabel_model <- tab_model(model_fit, show.se = TRUE, title = "Table: Regression Analysis Resul
ts")

# Print the table
tabel_model
```

Table: Regression Analysis Results

	prop_correct_ecn			
Predictors	*Estimates*	*std. Error*	*CI*	*p*
(Intercept)	0.46	0.02	0.42 – 0.50	**<0.001**
mean relative difficulty^2	-0.03	0.01	-0.04 – -0.02	**<0.001**
mean relative difficulty	-0.05	0.01	-0.06 – -0.04	**<0.001**
n question in spec training c	0.00	0.00	0.00 – 0.00	**<0.001**
student ability	0.05	0.00	0.04 – 0.05	**<0.001**
mean relative difficulty^2 × student ability	-0.01	0.00	-0.01 – -0.00	**<0.001**
mean relative difficulty × student ability	-0.02	0.01	-0.03 – -0.01	**<0.001**
n question in spec training c × student ability	-0.00	0.00	-0.00 – 0.00	0.148

Random Effects

σ^2	0.01
$\tau_{00\ student}$	0.00
$\tau_{00\ specialty}$	0.01
$\tau_{11\ specialty.mean\_relative\_difficulty}$	0.00
$\tau_{11\ specialty.I(mean\_relative\_difficulty^2)}$	0.00
$\rho_{01\ specialty.mean\_relative\_difficulty}$	-0.47
$\rho_{01\ specialty.I(mean\_relative\_difficulty^2)}$	-0.88
ICC	0.63

$N_{student}$	6451
$N_{specialty}$	13
Observations	45436
Marginal R^2 / Conditional R^2	0.092 / 0.666

```
## CHECK COLINEARITY
library(performance)
check_collinearity(model_fit)
```

```
## # Check for Multicollinearity
##
## Low Correlation
##
##                                          Term  VIF   VIF 95% CI Increased SE
##                  I(mean_relative_difficulty^2) 3.80 [3.74, 3.86]         1.95
##                       mean_relative_difficulty 1.27 [1.25, 1.28]         1.13
##                  n_question_in_spec_training_c 1.32 [1.31, 1.34]         1.15
##                                 student_ability 1.23 [1.22, 1.24]         1.11
##  I(mean_relative_difficulty^2):student_ability 1.17 [1.15, 1.18]         1.08
##       mean_relative_difficulty:student_ability 3.83 [3.77, 3.89]         1.96
##  n_question_in_spec_training_c:student_ability 1.37 [1.35, 1.39]         1.17
##  Tolerance Tolerance 95% CI
##       0.26     [0.26, 0.27]
##       0.79     [0.78, 0.80]
##       0.76     [0.75, 0.76]
##       0.81     [0.80, 0.82]
##       0.86     [0.85, 0.87]
##       0.26     [0.26, 0.27]
##       0.73     [0.72, 0.74]
```

```
## AIC and BIC - Akaike Information Criterion (AIC) and the Bayesian Information Criterion (B
IC)
aic_value <- AIC(model_fit)
bic_value <- BIC(model_fit)
cat("AIC =", aic_value, "\n")
```

```
## AIC = -96539.61
```

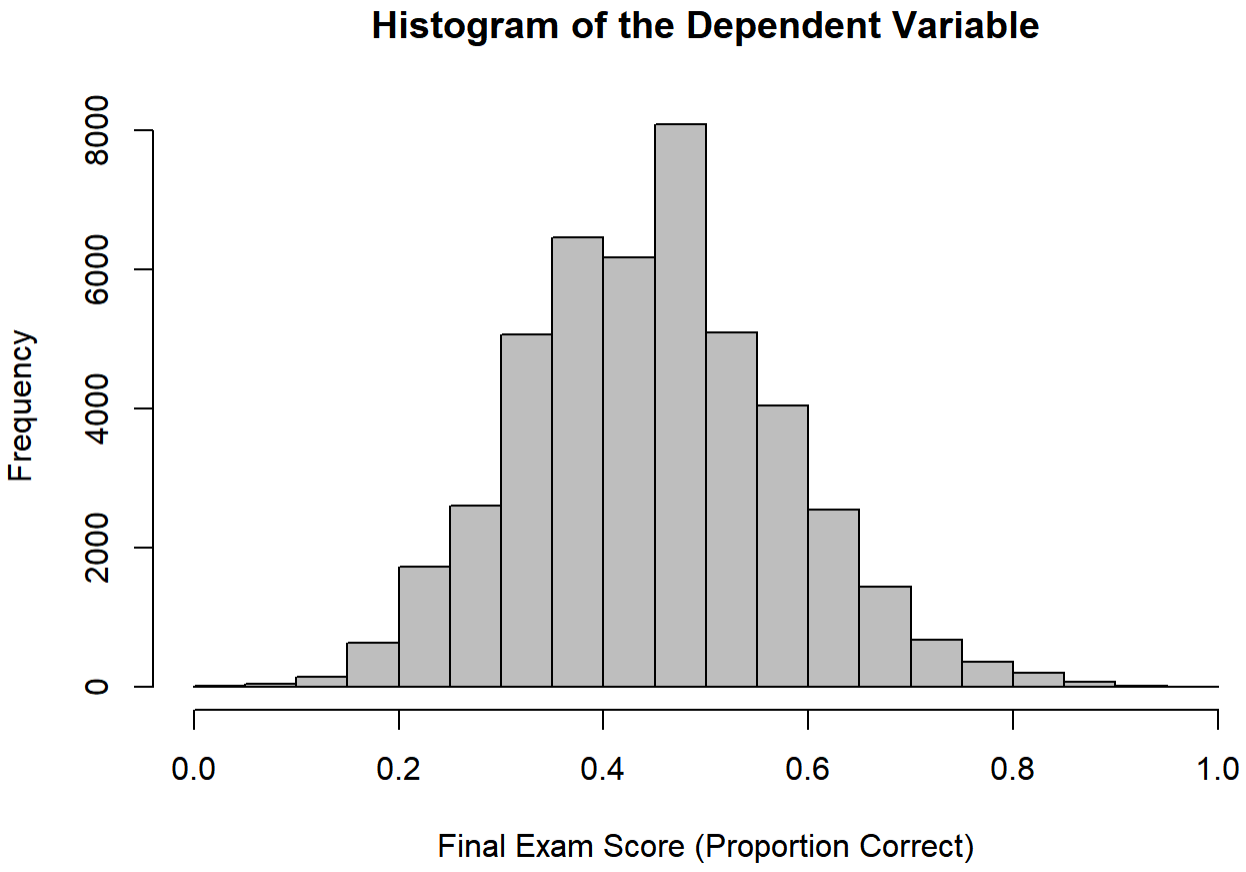
```
cat("BIC =", bic_value, "\n")
```

```
## BIC = -96400.03
```

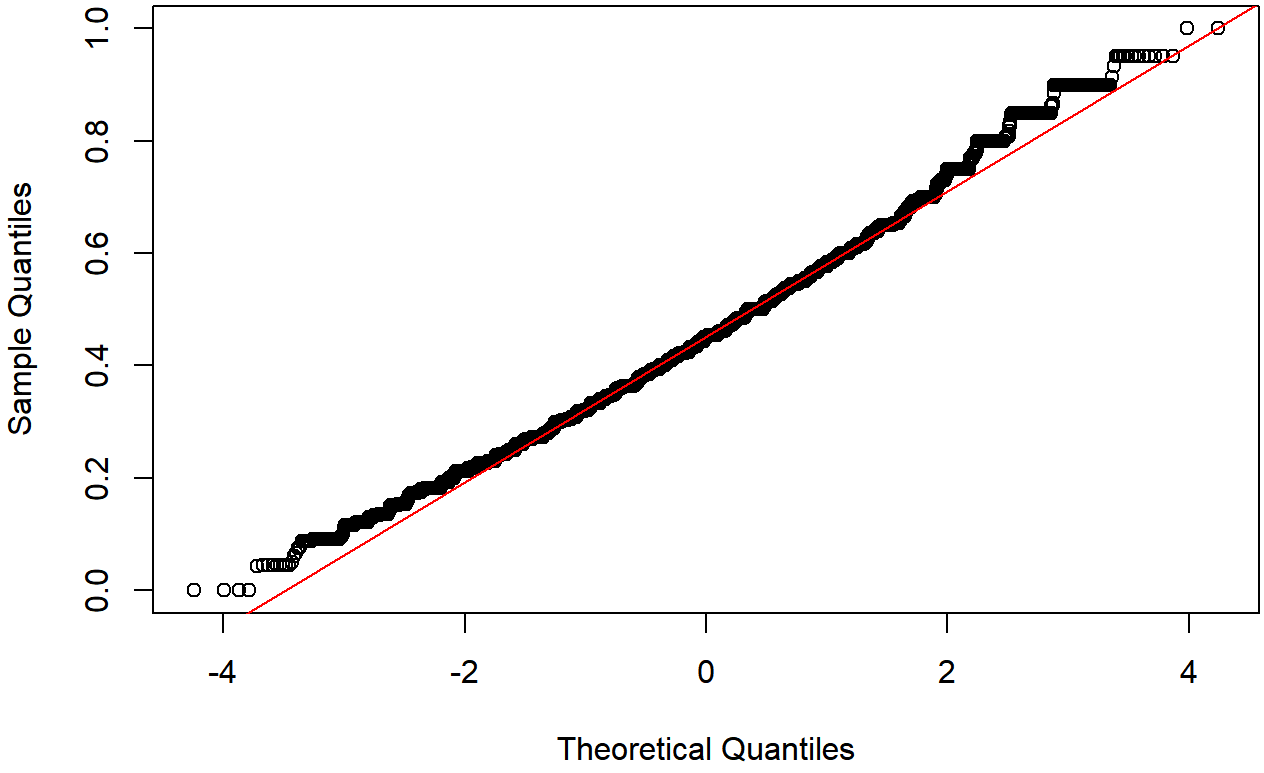
Assumption checks

for DV

```r
# Histogram of the DV
hist(average_quest_dif$prop_correct_ecn,
     main = "Histogram of the Dependent Variable",
     xlab = "Final Exam Score (Proportion Correct)",
     breaks = 30, col = "gray")
```




```r
# Q-Q plot of the DV
qqnorm(average_quest_dif$prop_correct_ecn, main = "Q-Q Plot of the Dependent Variable")
qqline(average_quest_dif$prop_correct_ecn, col = "red")
```



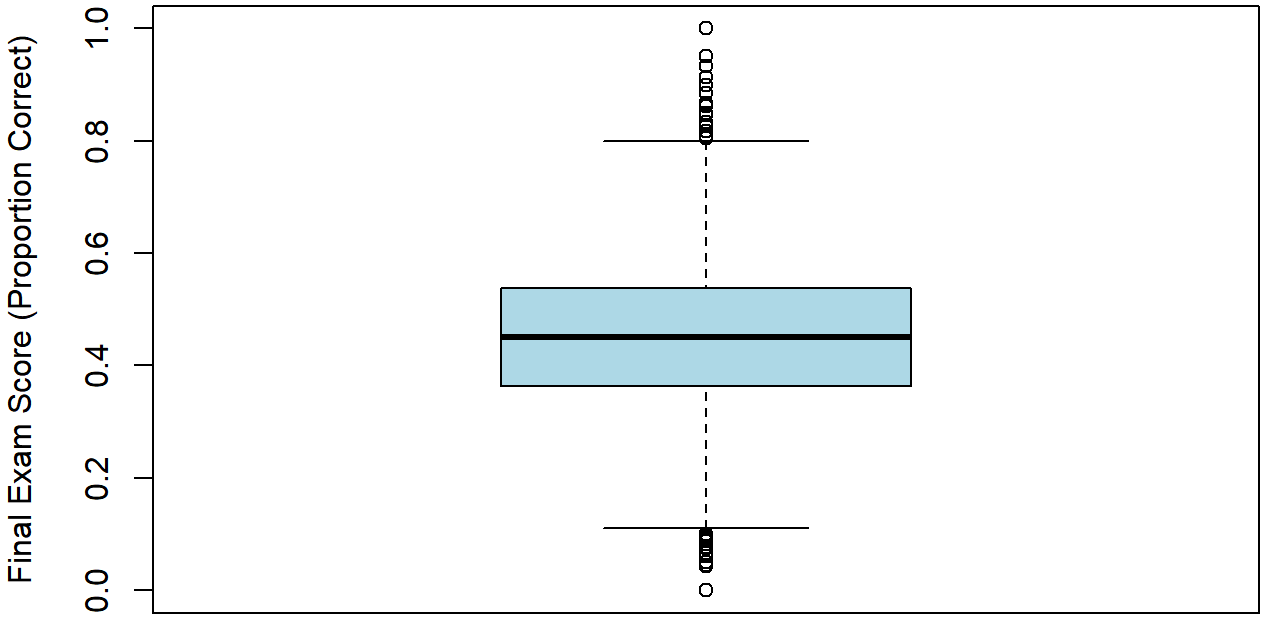

```
# Skewness and Kurtosis
skewness_value <- skewness(average_quest_dif$prop_correct_ecn)
kurtosis_value <- kurtosis(average_quest_dif$prop_correct_ecn)
print(paste("Skewness:", skewness_value))
```

```
## [1] "Skewness: 0.279933632684906"  "Skewness: 0.0114907110788987"
```

```
print(paste("Kurtosis:", kurtosis_value))
```

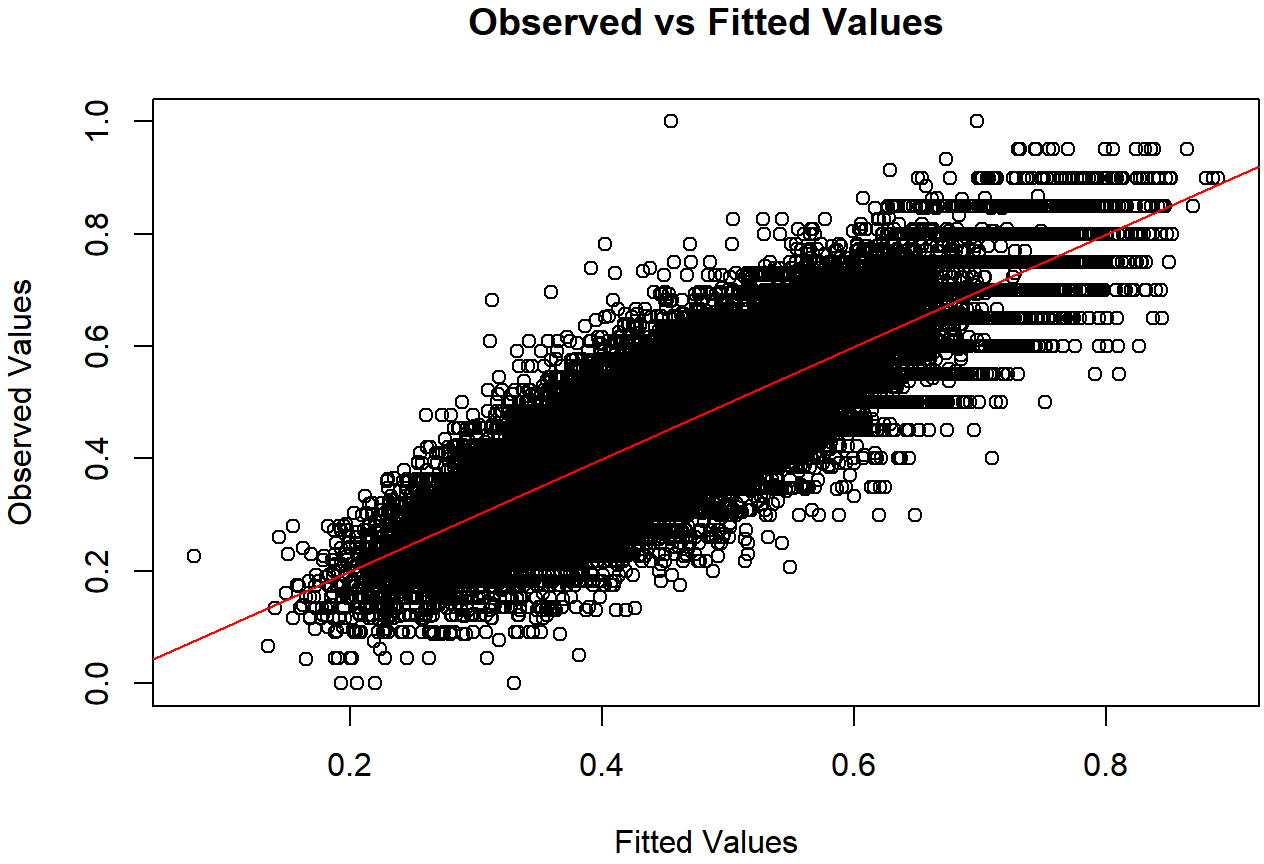
```
## [1] "Kurtosis: 0.170395181520379"  "Kurtosis: 0.0229791462283684"
```

```
# Box plot for the dependent variable
boxplot(average_quest_dif$prop_correct_ecn,
        main = "Box Plot of Dependent Variable",
        ylab = "Final Exam Score (Proportion Correct)",
        col = "lightblue")
```

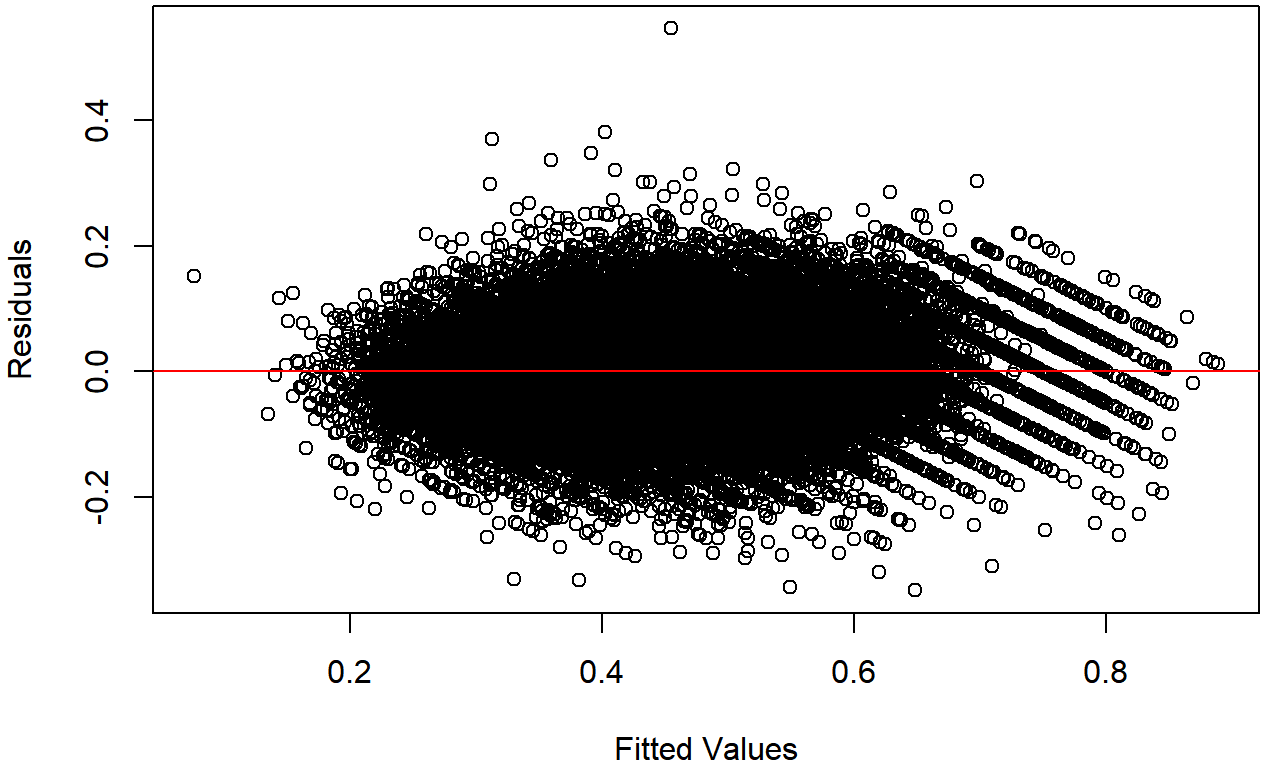


for residuals

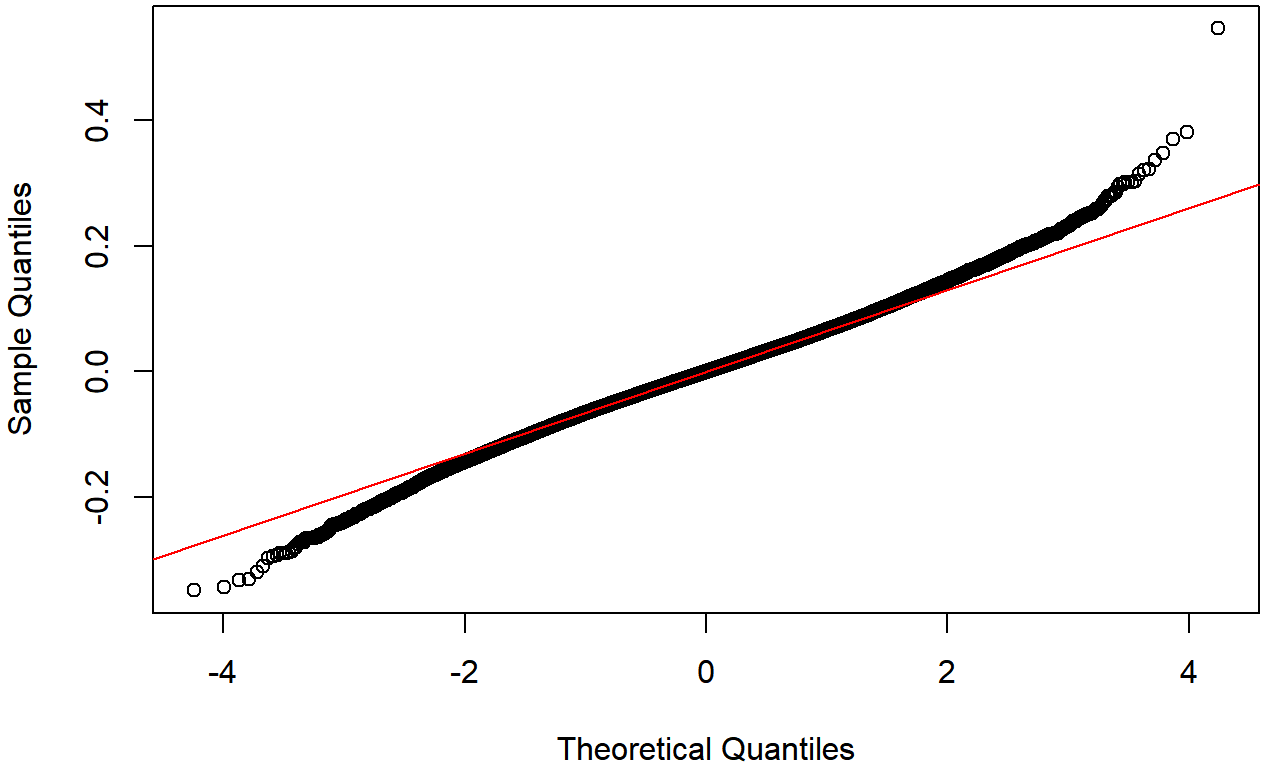
```
# linearity
# Plot observed vs fitted values
fitted_values <- fitted(model_fit)
plot(fitted_values, average_quest_dif$prop_correct_ecn,
     main = "Observed vs Fitted Values",
     xlab = "Fitted Values", ylab = "Observed Values")
abline(a = 0, b = 1, col = "red")  # Add a diagonal line
```



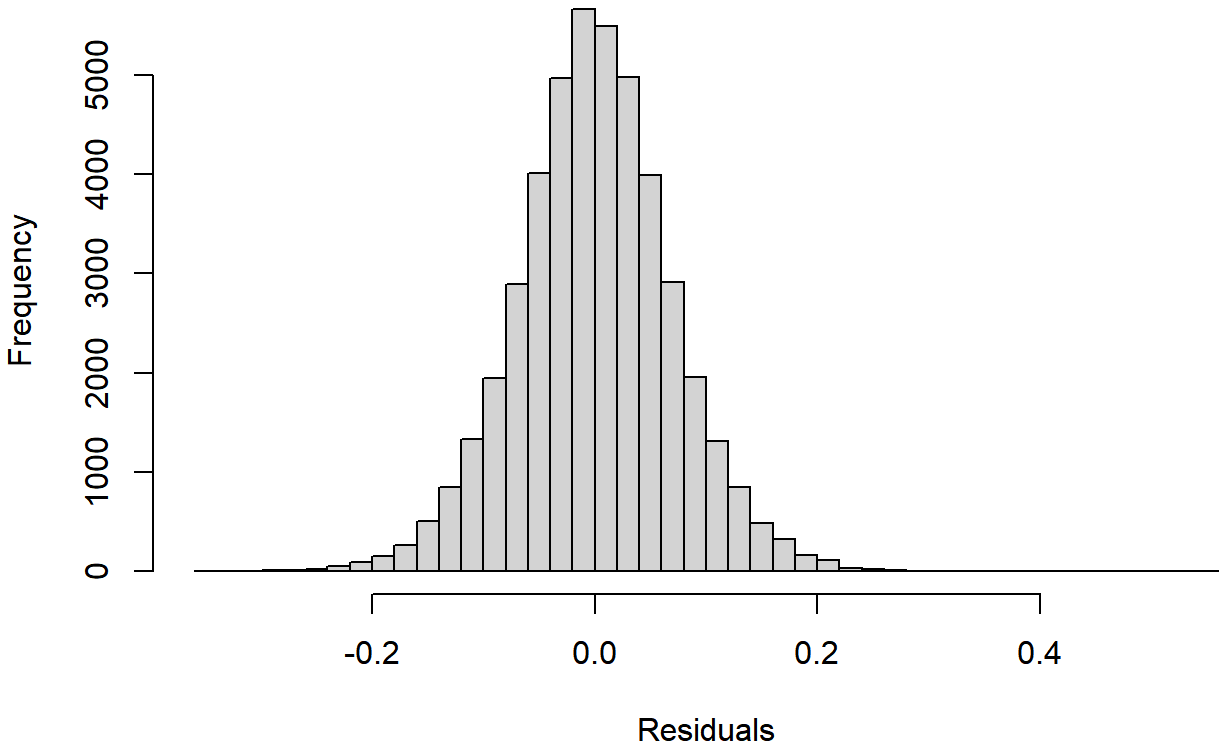

```
# Residuals vs Fitted Values
residuals <- resid(model_fit)
plot(fitted_values, residuals,
     main = "Residuals vs Fitted Values",
     xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")  # Add a horizontal line at 0
```




```
residuals <- resid(model_fit)

# Q-Q plot
qqnorm(residuals)
qqline(residuals, col = "red")
```




```
# Histogram
hist(residuals, breaks = 50, main = "Histogram of Residuals", xlab = "Residuals")
```



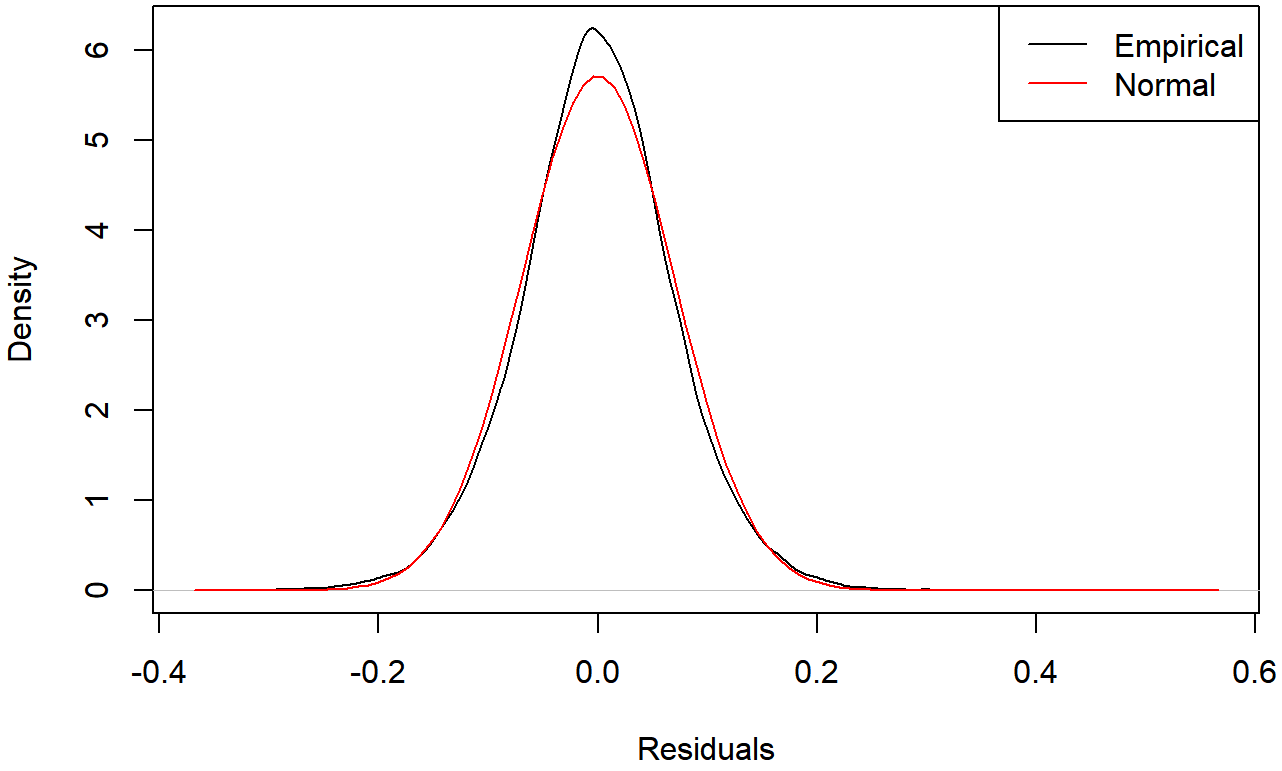

```
skewness_value <- skewness(residuals)
kurtosis_value <- kurtosis(residuals)

print(paste("Skewness:", skewness_value))
```

```
## [1] "Skewness: 0.00672416259567654" "Skewness: 0.0114907110788987"
```

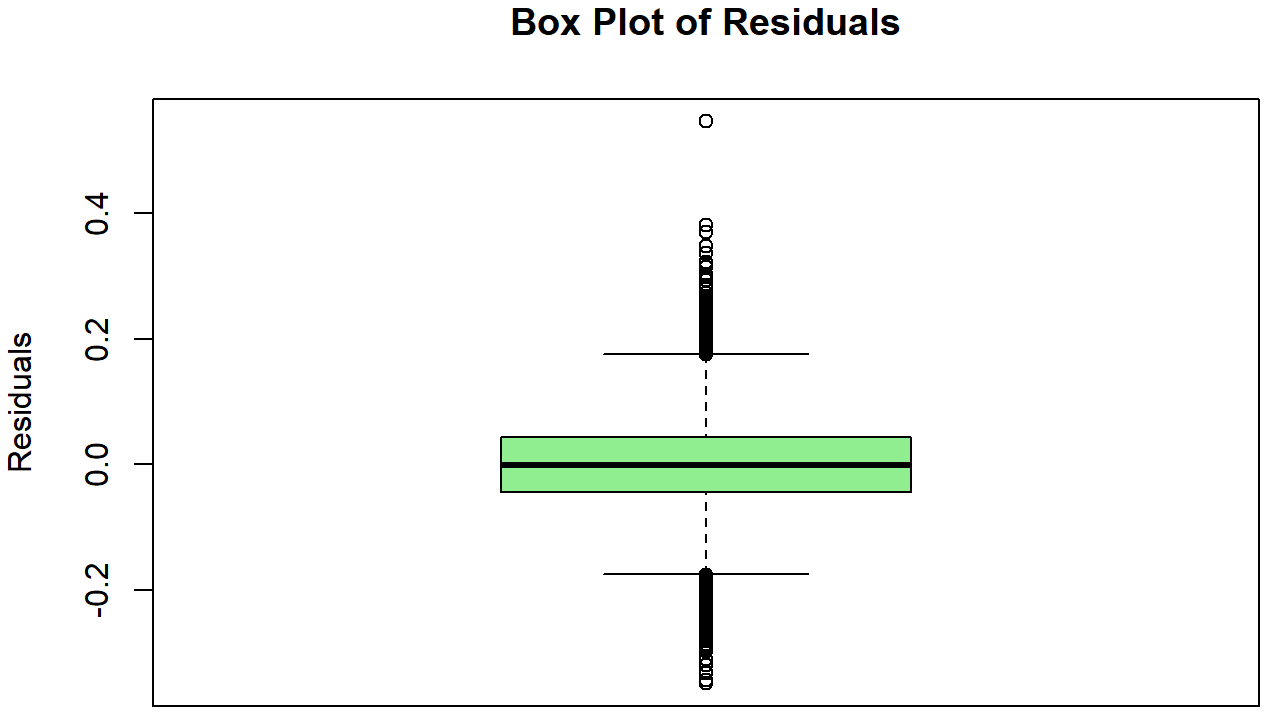
```
print(paste("Kurtosis:", kurtosis_value))
```

```
## [1] "Kurtosis: 0.729985099728653"  "Kurtosis: 0.0229791462283684"
```

```
plot(density(residuals), main = "Density Plot of Residuals", xlab = "Residuals")
curve(dnorm(x, mean = mean(residuals), sd = sd(residuals)), add = TRUE, col = "red")
legend("topright", legend = c("Empirical", "Normal"), col = c("black", "red"), lty = 1)
```




```
# Box plot for residuals
residuals <- resid(model_fit)  # Calculate residuals from the fitted model
boxplot(residuals,
        main = "Box Plot of Residuals",
        ylab = "Residuals",
        col = "lightgreen")
```



p value for random effect

```
library(lme4)
full_model <- lmer(prop_correct_ecn ~ I(mean_relative_difficulty^2) + mean_relative_difficult
y +
  n_question_in_spec_training_c +
  student_ability +
  student_ability:I(mean_relative_difficulty^2) +
  student_ability:mean_relative_difficulty +
  student_ability:n_question_in_spec_training_c +
  (1 | student) +
  (1 + mean_relative_difficulty + I(mean_relative_difficulty^2) | specialty),
  data = average_quest_dif, control = lmerControl(optimizer = "bobyqa"), REML = FALSE)


reduced_model <- lmer(prop_correct_ecn ~ I(mean_relative_difficulty^2) + mean_relative_diffic
ulty +
  n_question_in_spec_training_c +
  student_ability +
  student_ability:I(mean_relative_difficulty^2) +
  student_ability:mean_relative_difficulty +
  student_ability:n_question_in_spec_training_c +
  (1 | student) +
  (1 + I(mean_relative_difficulty^2) | specialty),
  data = average_quest_dif, control = lmerControl(optimizer = "bobyqa"), REML = FALSE)

library(lmtest)
anova_result <- anova(reduced_model, full_model)
print(anova_result)
```

```
## Data: average_quest_dif
## Models:
## reduced_model: prop_correct_ecn ~ I(mean_relative_difficulty^2) + mean_relative_difficulty
+ n_question_in_spec_training_c + student_ability + student_ability:I(mean_relative_difficult
y^2) + student_ability:mean_relative_difficulty + student_ability:n_question_in_spec_training
_c + (1 | student) + (1 + I(mean_relative_difficulty^2) | specialty)
## full_model: prop_correct_ecn ~ I(mean_relative_difficulty^2) + mean_relative_difficulty +
n_question_in_spec_training_c + student_ability + student_ability:I(mean_relative_difficulty^
2) + student_ability:mean_relative_difficulty + student_ability:n_question_in_spec_training_c
+ (1 | student) + (1 + mean_relative_difficulty + I(mean_relative_difficulty^2) | specialty)
##               npar    AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## reduced_model   13 -96307 -96194  48167   -96333
## full_model      16 -96641 -96501  48336   -96673 339.64  3  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

##have table with exact numbers

```r
# Assume we already have the summary data frame
fixed_effects <- summary(model_fit)$coefficients

# Check if p-values are below .Machine$double.eps and adjust display
fixed_effects[,"Pr(>|t|)"] <- ifelse(fixed_effects[,"Pr(>|t|)"] < .Machine$double.eps,
                                     "< 1e-16",  # You can choose the threshold
                                     format.pval(fixed_effects[,"Pr(>|t|)"], digits = 10))

# Print the modified fixed effects table
print(fixed_effects)
```

```
##                                                 Estimate
## (Intercept)                                    "0.457535620564066"
## I(mean_relative_difficulty^2)                  "-0.0267699245486933"
## mean_relative_difficulty                       "-0.0489430168494265"
## n_question_in_spec_training_c                  "2.52919720419422e-05"
## student_ability                                "0.0467072002168218"
## I(mean_relative_difficulty^2):student_ability "-0.00807715168270789"
## mean_relative_difficulty:student_ability       "-0.0203701972624671"
## n_question_in_spec_training_c:student_ability "-6.99842162894966e-06"
##                                                 Std. Error
## (Intercept)                                    "0.0209566294401127"
## I(mean_relative_difficulty^2)                  "0.00568710977246767"
## mean_relative_difficulty                       "0.00578718044392728"
## n_question_in_spec_training_c                  "2.64209171749322e-06"
## student_ability                                "0.00262958167672603"
## I(mean_relative_difficulty^2):student_ability "0.00200826237771116"
## mean_relative_difficulty:student_ability       "0.00558870513437058"
## n_question_in_spec_training_c:student_ability "4.83585183050647e-06"
##                                                 df
## (Intercept)                                    "12.0415704438775"
## I(mean_relative_difficulty^2)                  "234.753897025449"
## mean_relative_difficulty                       "18.9228005444686"
## n_question_in_spec_training_c                  "44516.2658754383"
## student_ability                                "42107.8805820956"
## I(mean_relative_difficulty^2):student_ability "664.172213911189"
## mean_relative_difficulty:student_ability       "31905.9330499032"
## n_question_in_spec_training_c:student_ability "25672.6823292776"
##                                                 t value
## (Intercept)                                    "21.832500396667"
## I(mean_relative_difficulty^2)                  "-4.7071228831016"
## mean_relative_difficulty                       "-8.45714373754916"
## n_question_in_spec_training_c                  "9.57270781876521"
## student_ability                                "17.7622169450826"
## I(mean_relative_difficulty^2):student_ability "-4.02196036352258"
## mean_relative_difficulty:student_ability       "-3.64488674437129"
## n_question_in_spec_training_c:student_ability "-1.44719521487421"
##                                                 Pr(>|t|)
## (Intercept)                                    "4.716804929e-11"
## I(mean_relative_difficulty^2)                  "4.295765072e-06"
## mean_relative_difficulty                       "7.502799914e-08"
## n_question_in_spec_training_c                  "< 1e-16"
## student_ability                                "< 1e-16"
## I(mean_relative_difficulty^2):student_ability "6.434354644e-05"
## mean_relative_difficulty:student_ability       "0.0002679341625"
## n_question_in_spec_training_c:student_ability "0.1478544664251"
```

Model Fit

```
## PLOT

  # Create a sequence of mean_relative_difficulty values for plotting
  x_values <- seq(
    min(average_quest_dif$mean_relative_difficulty ),
    max(average_quest_dif$mean_relative_difficulty ),
    length.out = 20
  )

  x_values_slope <- seq(
    min(average_quest_dif$relative_difficulty_slope ),
    max(average_quest_dif$relative_difficulty_slope ),
    length.out = 20
  )

   x_values_ability <- seq(
    min(average_quest_dif$student_ability ),
    max(average_quest_dif$student_ability ),
    length.out = 20
  )

   x_values_nb_question<-seq(
    min(average_quest_dif$n_question_in_spec_training_c ),
    max(average_quest_dif$n_question_in_spec_training_c ),
    length.out = 10
  )

  # List of unique specialties
  specialties <- unique(average_quest_dif$specialty)


  # Create a data frame to store the values for prediction
prediction_data <- expand.grid(
  mean_relative_difficulty = x_values,
  specialty = unique(average_quest_dif$specialty),
  student_ability = x_values_ability,
  n_question_in_spec_training_c=mean(average_quest_dif$n_question_in_spec_training_c ),
  relative_difficulty_slope=mean(average_quest_dif$relative_difficulty_slope )
)

  # Predict values using the model, including random effects for 'specialty'
  predicted_values <- predict(model_fit, newdata = prediction_data, re.form = ~   (1 + mean_r
elative_difficulty + I(mean_relative_difficulty^2) | specialty))

  prediction_data$learning_slope = predicted_values

  # Convert student to factor
  average_quest_dif <- average_quest_dif %>%
    mutate(student = factor(student))

  library(ggplot2)
```

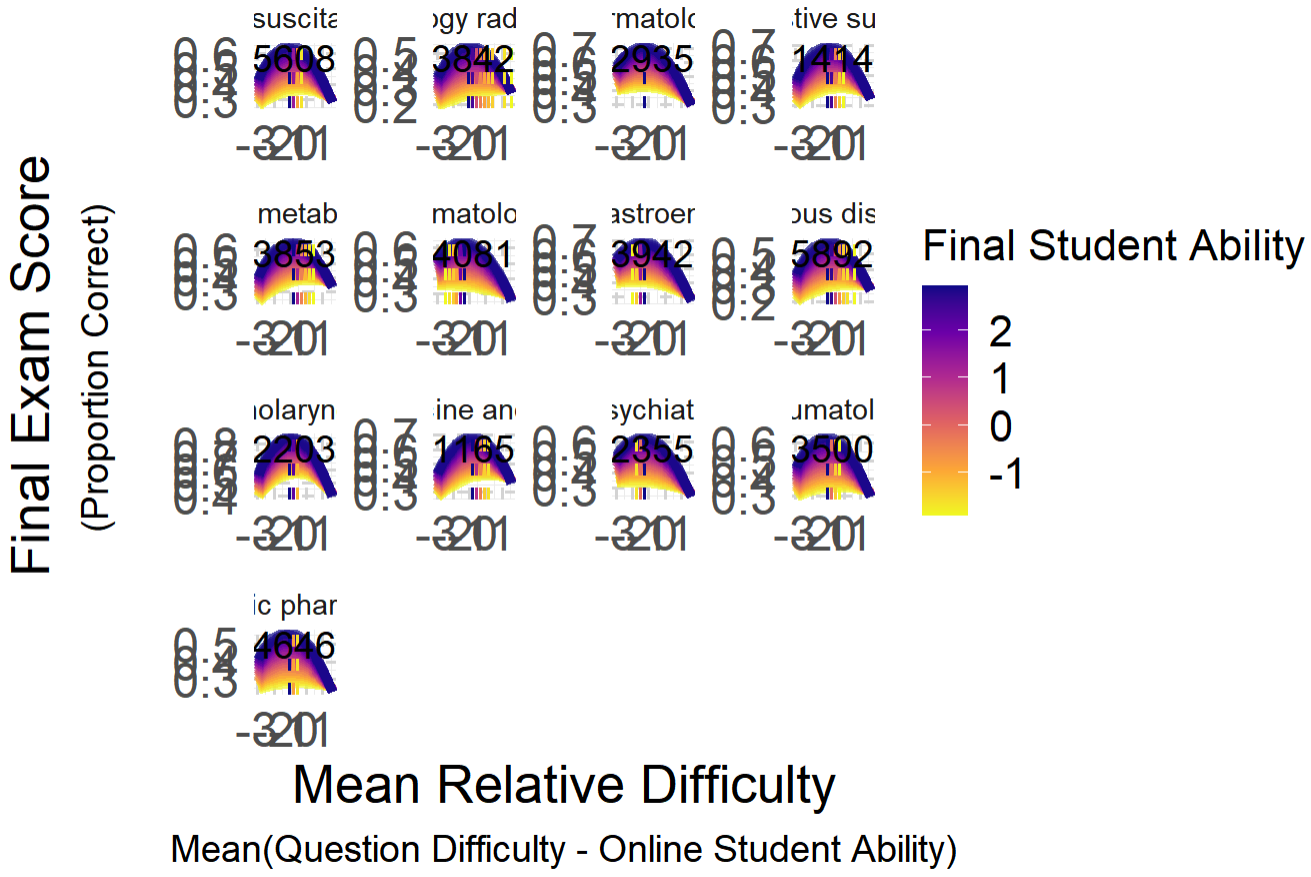
```
  #
  # # find the mean of mean relative diffciulty
  # mean_mean_relative_diffciulty= prediction_data %>%
  # group_by(specialty) %>%
  #    summarise(mean_mean = mean(mean_relative_difficulty))

# Find the mean_relative_difficulty values that maximize the predicted prop_correct_ecn for e
ach specialty and student_ability
max_x_values <- prediction_data %>%
  group_by(specialty, student_ability) %>%
  summarise(max_mean_difficulty = mean_relative_difficulty[which.max(learning_slope)])
```

```
## `summarise()` has grouped output by 'specialty'. You can override using the
## `.groups` argument.
```

```
  # Calculate the number of points in each facet for each specialty
facet_counts <- average_quest_dif %>%
  group_by(specialty) %>%
  summarise(n = n())

p <- ggplot(prediction_data, aes(x = mean_relative_difficulty, y = learning_slope)) +
  geom_line(aes(group = interaction(specialty, student_ability), color = student_ability),
            size = 2, alpha = 0.7) +
  labs(
    x = expression(atop("Mean Relative Difficulty", atop("Mean(Question Difficulty - Online S
tudent Ability)", ""))),
    y = expression(atop("Final Exam Score", atop("(Proportion Correct)", "")))
  ) +
  theme_minimal(base_size = 12) +
  theme(
    axis.text = element_text(size = 18),
    axis.title = element_text(size = 20),
    plot.title = element_text(size = 16, face = "bold"),
    plot.subtitle = element_text(size = 16, margin = margin(b = 10)),
    plot.caption = element_text(size = 14, margin = margin(t = 10)),
    legend.title = element_text(size = 16),
    legend.text = element_text(size = 16),
    strip.text = element_text(size = 11),
    panel.grid.major = element_line(color = "lightgray", linetype = "dashed")
  ) +
  facet_wrap(~gsub("_", " ", specialty), scales = "free", ncol = 4) +  # 4 columns
  scale_color_viridis_c(option = "C", direction = -1) +
  geom_vline(
    data = max_x_values,
    aes(xintercept = max_mean_difficulty, color = student_ability),
    size = 0.7,
    linetype = "dashed"
  ) +
  geom_text(
    data = facet_counts,
    aes(label = paste("n =", n), x = Inf, y = Inf),
    hjust = 1,
    vjust = 1,
    size = 5,
    show.legend = FALSE
  ) +
  labs(color = "Final Student Ability", linetype = "Optimal")
```

```
## 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.
```

```
# Increase the width for 4 subplots per row
ggsave(
   "C:/Users/Ghislaine/Desktop/optimal_difficulty_1/code/data/sides/elo_bins/modelfit.png",
  plot = p,
  width = 19, height = 10, units = "in", dpi = 300
)

print(p)
```



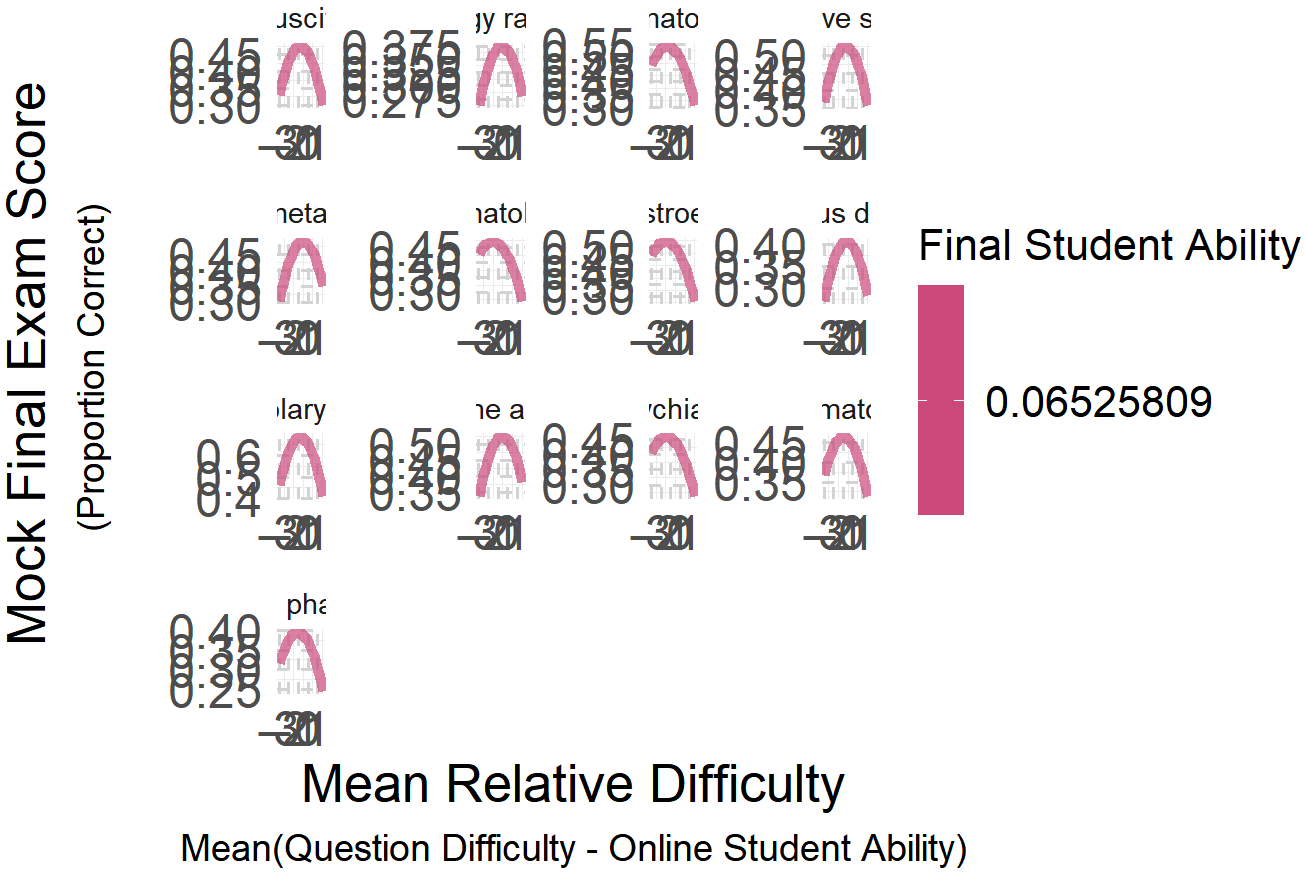
model fit for only students with mean student ability

```
## PLOT

  # Create a sequence of mean_relative_difficulty values for plotting
  x_values <- seq(
    min(average_quest_dif$mean_relative_difficulty ),
    max(average_quest_dif$mean_relative_difficulty ),
    length.out = 20
  )

  x_values_slope <- seq(
    min(average_quest_dif$relative_difficulty_slope ),
    max(average_quest_dif$relative_difficulty_slope ),
    length.out = 20
  )

   x_values_ability <- seq(
    min(average_quest_dif$student_ability ),
    max(average_quest_dif$student_ability ),
    length.out = 20
  )

   x_values_nb_question<-seq(
    min(average_quest_dif$n_question_in_spec_training_c ),
    max(average_quest_dif$n_question_in_spec_training_c ),
    length.out = 10
  )

  # List of unique specialties
  specialties <- unique(average_quest_dif$specialty)


  # Create a data frame to store the values for prediction
prediction_data <- expand.grid(
  mean_relative_difficulty = x_values,
  specialty = unique(average_quest_dif$specialty),
  student_ability = mean(average_quest_dif$student_ability ),
  n_question_in_spec_training_c=mean(average_quest_dif$n_question_in_spec_training_c ),
  relative_difficulty_slope=mean(average_quest_dif$relative_difficulty_slope )
)

  # Predict values using the model, including random effects for 'specialty'
  predicted_values <- predict(model_fit, newdata = prediction_data, re.form = ~   (1 + mean_r
elative_difficulty + I(mean_relative_difficulty^2) | specialty))

  prediction_data$learning_slope = predicted_values

  # Convert student to factor
  average_quest_dif <- average_quest_dif %>%
    mutate(student = factor(student))

  library(ggplot2)
```

```
  #
  # # find the mean of mean relative diffciulty
  # mean_mean_relative_diffciulty= prediction_data %>%
  # group_by(specialty) %>%
  #    summarise(mean_mean = mean(mean_relative_difficulty))

# Find the mean_relative_difficulty values that maximize the predicted prop_correct_ecn for e
ach specialty and student_ability
max_x_values <- prediction_data %>%
  group_by(specialty, student_ability) %>%
  summarise(max_mean_difficulty = mean_relative_difficulty[which.max(learning_slope)])
```

```
## `summarise()` has grouped output by 'specialty'. You can override using the
## `.groups` argument.
```

```r
  # Calculate the number of points in each facet for each specialty
facet_counts <- average_quest_dif %>%
  group_by(specialty) %>%
  summarise(n = n())

p <- ggplot(prediction_data, aes(x = mean_relative_difficulty, y = learning_slope)) +
  geom_line(aes(group = interaction(specialty, student_ability), color = student_ability),
            size = 2, alpha = 0.7) +
  labs(
    x = expression(atop("Mean Relative Difficulty", atop("Mean(Question Difficulty - Online S
tudent Ability)", ""))),
    y = expression(atop("Mock Final Exam Score", atop("(Proportion Correct)", "")))
  ) +
  theme_minimal(base_size = 12) +
  theme(
    axis.text = element_text(size = 18),
    axis.title = element_text(size = 20),
    plot.title = element_text(size = 16, face = "bold"),
    plot.subtitle = element_text(size = 16, margin = margin(b = 10)),
    plot.caption = element_text(size = 14, margin = margin(t = 10)),
    legend.title = element_text(size = 16),
    legend.text = element_text(size = 16),
    strip.text = element_text(size = 11),
    panel.grid.major = element_line(color = "lightgray", linetype = "dashed")
  ) +
  facet_wrap(~gsub("_", " ", specialty), scales = "free", ncol = 4) +  # 4 columns
  scale_color_viridis_c(option = "C", direction = -1) +
  # geom_vline(
  #   data = max_x_values,
  #   aes(xintercept = max_mean_difficulty, color = student_ability),
  #   size = 1,
  #   linetype = "dashed"
  # ) +
  labs(color = "Final Student Ability", linetype = "Optimal")
  #+geom_text(data = max_x_values, aes(x = max_mean_difficulty-1.5, y = 0.3, label = round(ma
x_mean_difficulty, 2)), vjust = -1, size = 7.5, color = "darkgreen", hjust = -0.1)


# Increase the width for 4 subplots per row
ggsave(
   "C:/Users/Ghislaine/Desktop/optimal_difficulty_1/code/data/sides/elo_bins/modelfit.png",
  plot = p,
  width = 19, height = 10, units = "in", dpi = 300
)

print(p)
```


Mock Final Exam Score
(Proportion Correct)
Mean Relative Difficulty
Mean(Question Difficulty - Online Student Ability)
Final Student Ability
0.06525809

```r
library(lme4)
library(dplyr)

# Assuming model_fit is already fitted

# Calculate mean values for student_ability and n_question_in_spec_training_c
mean_student_ability <- mean(average_quest_dif$student_ability, na.rm = TRUE)
mean_n_questions <- mean(average_quest_dif$n_question_in_spec_training_c, na.rm = TRUE)

# Define a sequence of mean_relative_difficulty values to evaluate
mrd_seq <- seq(from = min(average_quest_dif$mean_relative_difficulty, na.rm = TRUE),
               to = max(average_quest_dif$mean_relative_difficulty, na.rm = TRUE), length.out
= 100)

# Function to calculate predicted prop_correct_ecn for a given specialty
predict_specialty <- function(specialty) {
  # Create a data frame for predictions
  pred_data <- expand.grid(
    mean_relative_difficulty = mrd_seq,
    student_ability = mean_student_ability,
    n_question_in_spec_training_c = mean_n_questions,
    specialty = specialty
  )

  # Predict prop_correct_ecn using the model
  pred_data$prop_correct_ecn <- predict(model_fit, newdata = pred_data,  re.form = ~ (1 + mea
n_relative_difficulty + I(mean_relative_difficulty^2) | specialty))

  # Find the mean_relative_difficulty that maximizes prop_correct_ecn
  max_difficulty <- pred_data$mean_relative_difficulty[which.max(pred_data$prop_correct_ecn)]
  return(max_difficulty)
}

# Apply function to each specialty
specialties <- unique(average_quest_dif$specialty)
optimal_difficulties <- sapply(specialties, predict_specialty)

# View results
names(optimal_difficulties) <- specialties
optimal_difficulties
```

```
##                 cancerology_radiotherapy
##                                -0.5785345
##       endocrinology_metabolism_nutrition
##                                -0.5322532
##                                hematology
##                                -1.5504431
##                   hepatogastroenterology
##                                -1.5041618
##                      infectious_diseases
##                                -0.6248159
##                              rheumatology
##                                -0.9487854
## anesthesiology_resuscitation_emergencies
##                                -0.8099413
##                               dermatology
##                                -1.3190363
##                         digestive_surgery
##                                -0.7173786
##                       otorhinolaryngology
##                                -0.8099413
##     physical_medicine_and_rehabilitation
##                                -0.5322532
##                                psychiatry
##                                -1.4115990
##                 therapeutic_pharmacology
##                                -1.0413482
```

```
# Calculate mean and standard deviation of optimal mean_relative_difficulties
mean_difficulty <- mean(optimal_difficulties)
sd_difficulty <- sd(optimal_difficulties)

# Print results
cat("Mean of optimal mean_relative_difficulty: ", mean_difficulty, "\n")
```

```
## Mean of optimal mean_relative_difficulty:  -0.9523455
```

```
cat("Standard deviation of optimal mean_relative_difficulty: ", sd_difficulty, "\n")
```

```
## Standard deviation of optimal mean_relative_difficulty:  0.3776313
```