Identifying Inter-subject Difficulties in Norwegian GPA Data

Tony C. A. Tan

Centre for Educational Measurement, University of Oslo

Continuous Draft

Prof Rolf V. Olsen & Dr Astrid M. J. Sandsør

Høst 2022

Abstract

[Follow publisher's instructions.]

Keywords: grade point average, Norway, item response theory, assessment fairness, group invariance

Identifying Inter-subject Difficulties in Norwegian GPA Data

The grade point average (GPA, *skolepoeng* in Norwegian) plays a key role in Norway's educational assessment process. From Year 8 onwards, Norwegian high schoolers receive formal grades from both their teachers (*standpunktkarakter*) and year-end exams (Ræder et al., 2020), which are used for high-stake decisions such as graduate certifications (Year 10) and university admissions (VG3). Since different subjects are treated *equally* in its calculation (Directorate for Higher Education and Skills, 2021), GPA implicitly assumes that grades across different specialities are *equivalent* indicators of students' preparedness for the next stage of education — an assumption that remains untested and questioned by descriptive statistics (Directorate of Education, 2022).

Concerns for the comparability of subject difficulties are further deepened by prior studies from education systems similar to that of Norway's. He et al. (2018) in the UK and Korobko et al. (2008) in the Netherlands both reported persistent disagreements among subject difficulties, which may lead to differential treatment of students with different specialisations. Besides fairness concerns, the lack of difficulty comparability also leads to a lack of construct validity (Messick, 1989) in GPA, as the construct-irrelevant variance related to subject characteristics, in addition to candidates' competencies, have been included in the GPA calculation. Understanding the presence, directions, and magnitudes of inter-subject difficulties therefore becomes a key issue for assessing the validity of GPA. By analysing Norway's education record data, this study aims to test GPA's validity as a measurement scale in mapping candidate competence into numeric scores, as well as the fairness consequences subsequent to its use in high-stake situations. More specifically, this study will address the following research questions:

RQ1: Do Norwegian Year 10 subjects differ in their difficulty levels?

RQ2: Do subject difficulties differ by source such as between teachers and external examiners?

RQ3: Do subject difficulties differ across achievement levels?

RQ4: Do subject difficulties differ across demographic attributes such as socioeconomic status, gender and immigration background?

Conceptual Framework

The Norwegian Education and Assessment System

The Norwegian education system is organised into three levels: primary school (Year 1–7) where formal grading is not practised, lower secondary school (Year 8–10) and upper secondary school (Year 11–13). During the first ten years of schooling (*grunnskole*), students follow centralised national curricula with largely compulsory subjects plus a few electives. Upon successful completion of Year 10, students may choose between vocational and academic tracks for their upper secondary schools. The former is a two-year program (*fagskole*) that prepares students for employment in a specific field, whereas the latter is a three-year program (*videregående opplæring*, VG1–3) that prepares students for university studies. In preparation for the merit-based tertiary entry requirements, academic track students carefully choose their VG subjects in order to maximise their final GPA scores. This study focuses on Year 10 GPA data where the impact of subject choice is minimal.

The GPA aims to provide a sum-score measure of a student's overall competency. For *grunnskole* graduation purposes, the GPA is calculated as a weighted average of students' grades from all Year 10 subjects. The weights are determined by the number of hours spent on each subject, with the exception of Norwegian and English, which are weighted equally. Both teacher-assigned grades and exam grades are included in the GPA calculation, with each subject ranging from 1 (low competency) to 6 (outstanding). While every subject a student undertakes receives a teacher-assigned grade, Year 10 students are randomly assigned into participating in *one* of the three written exams, Norwegian, English, and mathematics, hence receives only one written exam grade. Similarly, students are randomly assigned into participating in *one* of the many oral exam options, including Norwegian, English, mathematics, and electives. A candidate's GPA is then computed by averaging the grades they have obtained, multiplying by 10, and rounding to two decimal places.

Methods

Population

This study retains the entire cohort of Norway's Year 10 students graduating in 2019 as its targeted population. Students' GPA (*grunnskolepoeng*), teacher-assigned grades (*standpunktkarakter*), as well as written (*SKR*) and oral (*MUN*) exam grades were extracted from the national register. This data source is unique because it is the *population*, rather than

samples, that forms our unit of analysis. Academic attainment records were then re-shaped into the format that each candidate occupies one row and each subject is represented by one column. This process led to a preliminary data set of 64,918 students and 200 subjects. Next, 4, 300 students without valid GPA records were excluded from subsequent analyses, representing a loss rate of 6.62%. Seventeen subjects were retained based on these inclusion criteria:

Teacher-assigned Grades (12 subjects)

Under the Norwegian education system, Year 10 students shall complete 13 compulsory subjects as well as electives. This study included all compulsory subjects except for Sidemål.[1] We applied equal treatment to courses instructed in Norwegian and in Sami language by merging these records.[2] Twelve teacher-assigned grades were included for our analysis: written Norwegian (NORW), oral Norwegian (NORO), written English (ENGW), oral English (ENGO), mathematics (MATH), natural sciences (NATS), social sciences (SOCS), religion (RELI), music (MUSI), arts and handcraft (HAND), food and health (FOOD), and physical education (PHED).

Written Exam Grades (3 subjects)

Norway uses a lottery draw to randomly assign Year 10 students to participate in *one* of the following three written exams: Norwegian (E-NORW), English (E-ENGW) and mathematics (E-MATH). This "planned missingness" implies that although numeral in quantity, the unobserved exam grades can be safely modelled under the missing-completely-at-random (MCAR) assumption (Little & Rubin, 2019).[3] Rasch models have a major advantage for handling missing values thanks to the sufficient overlap across subjects in the score matrix (He et al., 2018).

---

[1] The Norwegian language has two official written forms: Bokmål and Nynorsk, with the former being more prevalent in the media. Students growing up in one written form must enroll the other as their Sidemål, unless Norwegian is not their native language. Nynorsk users tend to have easier time in Sidemål due to existing exposure to Bokmål. Bokmål users, on the other hand, find Nynorsk more challenging while fulfilling Sidemål. Since Sidemål contains two sub-cohorts with distinct difficulty profiles, we opt not to include this subject in our analyses.

[2] For example, NAT0010 Naturfag 10. årstrinn and NAT0020 Naturfag, samisk plan, 10. årstrinn were merged into one subject Natural Sciences. If academic results were available from both instruction languages, we retained the higher grades during merging.

[3] Even if the lottery is less than perfectly random, Rasch models are still valid under the weaker assumption of missing-at-random (MAR), hence "ignorable" (Molenaar, 1995), as long as one is satisfied that missing propensities are unrelated to item or person parameters, a reasonable assumption in the Norwegian Year 10 context.

Oral Exam Grades (2 subjects)

Year 10's oral exams consist of the same three subjects as in written exams, plus a wide selection such as natural and social sciences, with students being randomly assigned into *one* oral exam by lottery. In order to better match teacher-assigned grades, only Oral Norwegian (E-NORO) and Oral English (E-ENGO) were included in this study. Since students are spread thinly across many oral exam subjects, E-NORO and E-ENGO appeared more sparse than their teacher-assigned counterparts, leading to larger confidence intervals in subsequent analyses.

References

Directorate for Higher Education and Skills. (2021). *Slik beregner du poeng* [How to calculate your GPA]. Direktoratet for høyere utdanning og kompetanse. https://utdanning.no/tema/soknad_og_opptak/slik_beregner_du_poeng

Directorate of Education. (2022). *Karakterstatistikk for videregående skole* [Grade statistics for upper secondary school]. Utdanningsdirektoratet. https://www.udir.no/tall-og-forskning/statistikk/statistikk-videregaende-skole/karakterer-vgs/

He, Q., Stockford, I., & Meadows, M. (2018). Inter-subject comparability of examination standards in GCSE and GCE in England. *Oxford Review of Education*, *44*(4), 494–513. https://doi.org/10.1080/03054985.2018.1430562

Korobko, O. B., Glas, C. A. W., Bosker, R. J., & Luyten, J. W. (2008). Comparing the difficulty of examination subjects with item response theory. *Journal of Educational Measurement*, *45*(2), 139–157. https://doi.org/10.1111/j.1745-3984.2007.00057.x

Little, R. J. A., & Rubin, D. B. (2019). *Statistical analysis with missing data* (3rd ed.). Wiley. https://doi.org/10.1002/9781119482260

Messick, S. (1989). Validity. In R. L. Linn (Ed.), *Educational measurement* (3rd ed., pp. 10–103). American Councile on Education; Macmillan.

Molenaar, I. W. (1995). Estimation of item parameters. In G. H. Fischer & I. W. Molenaar (Eds.), *Rasch models: Foundations, recent developments, and applications* (pp. 39–51). Springer-Verlag. https://doi.org/10.1007/978-1-4612-4230-7_3

Ræder, H. G., Olsen, R. V., & Blömeke, S. (2020). Large-scale assessments in the Norwegian context. In H. Harju-Luukkainen, N. McElvany, & J. Stang (Eds.), *Monitoring student achievement in the 21st Century: European policy perspectives and assessment strategies* (pp. 195–206). Springer. https://doi.org/10.1007/978-3-030-38969-7_16

Analysis Code, Additional Tables and Figures

Register Data Re-format

```r
# Only keep 2019 data
# STP (Teacher assigned marks)
teacher_mk <- gpa[which(gpa$AVGDATO == 201906), c(1:4, 7)]
# Save the total number of students
n_student <- dim(teacher_mk)[1] # Should be 1,073,204 obs
if (interactive()) {n_student}

# Inspect unusual marks in the "STP" column
if (interactive()) {
    table(unlist(teacher_mk$STP))
}
# These marks are not usable:
#   '' empty [n = 20,042],
#   7 [n = 33],
#   D [n = 58,182],
#   F [n = 37,273],
#   GK [n = 55],
#   IM [n = 2],
#   IV [n = 12,576].

# Recode un-usable STP into NA
teacher_mk$STP <- car::recode(teacher_mk$STP, "
    c('', '7', 'D', 'F', 'GK', 'IM', 'IV') = NA
")



# Part 1: Re-shape teacher-assigned marks columns: one subject per column

# How many subjects there are? (Answer: 200 different subjects in total)
# How many times each subject name appeared (with or without valid score)?
subject_frequency <- sort(table(unlist(teacher_mk$FAGKODE)), decreasing = T)
if (interactive()) {subject_frequency}
# Save subject list
subject_list <- as.character(data.frame(subject_frequency)[, 1])
# Save total number of subjects
n_subject <- length(subject_list)
if (interactive()) {n_subject} # Should be 200 subjects in total

# Create a placeholder spreadsheet
stp_spreadsheet <- data.frame(matrix(NA, nrow = n_student, ncol = n_subject))
colnames(stp_spreadsheet) <- subject_list

# Stitch STP and this empty placeholder spreadsheet together
teacher_reshape <- cbind(teacher_mk, stp_spreadsheet)
if (interactive()) {names(teacher_reshape)}

# Set up a progress bar
n_iter <- dim(teacher_reshape)[2] # Set the progress bar's end point
pb <- progress::progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

for (j in 6:n_iter) { # 200 cycles
    # Insert progress bar here
    progress::pb$tick() # Update progress bar

    # Create a placeholder list
    temp <- rep(names(teacher_reshape)[j], n_student)
    # Test whether subject names match
```

```r
    equal_test <- temp == teacher_reshape[, 4]
    # Turn FALSE/TRUE to 0/1
    equal_test <- equal_test + 0

    # If subject name matches, copy-paste teacher-assign marks
    # into the temp_subject column
    temp_subject <- equal_test * teacher_reshape[, 5]
    # Turn off list property (in order to recode)
    temp_subject <- as.numeric(unlist(temp_subject))
    # Recode 0 to NA
    teacher_reshape[, j] <- car::recode(temp_subject, "0 = NA")
}
cat("\n") # Start a new line once progress bar is full

# Remove subject name and STP columns
teacher_reshaped <- teacher_reshape[, -c(4, 5)]
# Inspect the newly shaped data set
if (interactive()) {head(teacher_reshaped, 20)}

# Save to external file.
if (Sys.info()["sysname"] == "Windows") {
    data.table::fwrite(teacher_reshaped,
        "M:/p1708-tctan/Documents/teacher0.csv",
        row.names = F
    )
} else {
    data.table::fwrite(teacher_reshaped,
        "/tsd/p1708/home/p1708-tctan/Documents/teacher0.csv",
        row.names = F
    )
}
# Should be 239,329 KB in size



# Part 2: Re-shape rows: one student per row

# How many (unique) students there are? (Answer: 64,918 unique students)
# How many times each student ID appeared (with or without valid score)?
student_frequency <- data.frame(sort(
    table(unlist(teacher_reshaped$w21_4952_lopenr_person)),
    decreasing = T
))
# Display the top 20 students who took the most number of subjects
head(student_frequency, 20)
# Display the bottom 20 students who took the least number of subjects
tail(student_frequency, 20)
# Save student list
student_list <- as.character(student_frequency[, 1])
# Save total number of unique students
(n_unique_student <- length(student_list)) # 64,918 unique students

# Set up a placeholder spreadsheet
teacher_reshaped_final <- matrix(
    nrow = n_unique_student, ncol = dim(teacher_reshaped)[2]
)
colnames(teacher_reshaped_final) <- names(teacher_reshaped)
teacher_reshaped_final <- data.frame(teacher_reshaped_final)

# Prepare multi-core processing
if (Sys.info()["sysname"] == "Windows") { # Windows can only use single core
    n_cores <- 1
} else { # Both Linux and Mac can implement multicore
    n_cores <- parallel::detectCores() # Count the total number of CPU cores
    n_cores <- n_cores - 1 # Reserve one core for system admin
}

# Set up a progress bar
n_iter <- n_unique_student  # Set the progress bar's end point
pb <- progress::progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
```

```
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

for(i in 1:n_iter) {
    # Insert progress bar here
    progress::pb$tick()

    # Pull out lines that share the same Student ID
    student_temp <- teacher_reshaped[which(
        teacher_reshaped[, 1] == student_list[i]
    ), ]
    # Collapse multiple lines into one line
    student_temp_teacher <- parallel::mclapply(student_temp[, -c(1:3)],
    function(x) max(x, na.rm = T), mc.cores = n_cores)
    # In cases where, same person, same subject, but multiple marks,
    # take the maximum, because I do not know which score was given first.
    # When I asked R to compute max from a column containing NA only,
    # R produced -Inf and a warning.
    # Safe to ignore these warnings and turn -Inf to NA.
    # Recode 0 and -Inf to NA
    student_temp_teacher <- car::recode(student_temp_teacher, "
        c('0', '-Inf') = NA
    ")
    # Stitch admin variables to student_temp_teacher (need transpose)
    teacher_reshaped_final[i, ] <- data.frame(cbind(
        student_temp[1, c(1:3)], t(student_temp_teacher)
    ))
}
cat("\n") # Start a new line once progress bar is full

# Save the standard Student ID list for subsequent work
if (Sys.info()["sysname"] == "Windows") {
    write.table(teacher_reshaped_final[, 1],
        "M:/p1708-tctan/Documents/student_id.csv",
        row.names = F, col.names = c("student_id")
    )
} else {
    write.table(teacher_reshaped_final[, 1],
        "/tsd/p1708/home/p1708-tctan/Documents/student_id.csv",
        row.names = F, col.names = "student_id"
    )
}
# Should be 888 KB in size

# Save teacher-assigned marks
if (Sys.info()["sysname"] == "Windows") {
    data.table::fwrite(teacher_reshaped_final,
        "M:/p1708-tctan/Documents/teacher1.csv",
        row.names = F
    )
} else {
    data.table::fwrite(teacher_reshaped_final,
        "/tsd/p1708/home/p1708-tctan/Documents/teacher1.csv",
        row.names = F
    )
}
# Should be 15,345 KB in size
```

Retain Students with Valid GPA and Sufficient Teacher-assigned Grades

```
# Full data set: N = 64,918

# Drop students without valid GPAs
teacher_gpa <- teacher[!is.na(teacher$GRUNNSKOLEPOENG), ]
if (interactive()) {dim(teacher_gpa)} # 60,618 students remain

# Compute data loss rate (n = 4,300, % = 6.62)
```

```r
if (interactive()) {
    dim(teacher)[1] - dim(teacher_gpa)[1]
    round((dim(teacher)[1] - dim(teacher_gpa)[1]) / dim(teacher)[1] * 100, 2)
}

# Sort columns by number of valid entries
teacher_sorted <- teacher_gpa[ , order(colSums(is.na(teacher_gpa)))]

# Keep the order of admin variables
teacher_export <- cbind(teacher_gpa[, c(1:8)], teacher_sorted[, -c(1:8)])

# Save the subject list
n_valid <- dim(teacher_export)[1] - colSums(is.na(teacher_export[,-c(1:8)]))
r_valid <- round(n_valid / dim(teacher_export)[1] * 100, 2)
teacher_valid <- cbind(n_valid, r_valid)

# Preserve subject list to an external file
data.table::fwrite(data.frame(teacher_valid), "subject_list.csv", row.names = T)

# Save data set containing anyone with valid GPA
data.table::fwrite(teacher_export, "./Rolf/stp_valid_gpa.csv", row.names = F)

# Save "the 12 subjects" including both Norwegian and Sami as instruction lang
teacher_export_subj_12 <- teacher_export[, c(
    # English x 2
        # ENGW:
    'ENG0012',  # 1 English (written)
        # ENGO:
    'ENG0013',  # 2 English  (oral)
    # HAND: Handcraft
    'KHV0010',  # 3 Handcraft
    'KHV0020',  # 4 Duoji (Sami handcraft)
    # PHED: Physical Education
    'KRO0020',  # 5 P.E.
    # MATH: Mathematics
    'MAT0010',  # 6 Mathematics
    # FOOD: Food and Health
    'MHE0010',  # 7 Food and Health
    'MHE0020',  # 8 Food and Health  (instructed in Sami)
    # MUSI: Music
    'MUS0010',  # 9 Music
    'MUS0020',  # 10 Music  (instructed in Sami)
    # NATS: Natural Sciences
    'NAT0010',  # 11 Natural Sciences
    'NAT0020',  # 12 Natural Sciences  (instructed in Sami)
    # Norwegian x 2
        # NORW:
    'NOR0214',  # 13 Norwegian  (written)
    'NOR0041',  # 14 Norwegian  (written, native language Sami)
        # NORO:
    'NOR0216',  # 15 Norwegian  (oral)
    'NOR0042',  # 16 Norwegian  (oral, native language Sami)
    # RELI: Religion
    'RLE0030',  # 17 Religion
    'RLE0040',  # 18 Religion  (instructed in Sami)
    # SOCS: Social Sciences
    'SAF0010',  # 19 Social Sciences
    'SAF0020'   # 20 Social Sciences (instructed in Sami)
)]
# Add admin variables to "the 12 subjects" list
teacher_export_12 <- cbind(teacher_export[, c(1:8)], teacher_export_subj_12)
# Save "the 12 subjects"
data.table::fwrite(teacher_export_12, "./Rolf/stp_12.csv", row.names = F)



# Merge Norwegian- and Sami-instructed marks

# Create a placeholder matrix
subj_12 <- data.frame(matrix(NA, nrow= dim(teacher_export_12)[1], ncol = 12))
names(subj_12) <- c(
    "ENGW", "ENGO", # 1, 2
    "HAND",         # 3
```

```
    "PHED",          # 4
    "MATH",          # 5
    "FOOD",          # 6
    "MUSI",          # 7
    "NATS",          # 8
    "NORW", "NORO", # 9, 10
    "RELI",          # 11
    "SOCS"           # 12
)

# Copy-paste subjects that do not need merges
subj_12[, 1] <- teacher_export_subj_12[, 1] # ENGW: English (written)
subj_12[, 2] <- teacher_export_subj_12[, 2] # ENGO: English (oral)
subj_12[, 4] <- teacher_export_subj_12[, 5] # PHED: Physical education
subj_12[, 5] <- teacher_export_subj_12[, 6] # MATH: Mathematics


# Set up a progress bar
library(progress)

n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge HAND
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 3] <- max(
        teacher_export_subj_12[i, 3],
        teacher_export_subj_12[i, 4],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge FOOD
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 6] <- max(
        teacher_export_subj_12[i, 7],
        teacher_export_subj_12[i, 8],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
```

```
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge MUSI
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 7] <- max(
        teacher_export_subj_12[i, 9],
        teacher_export_subj_12[i, 10],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge NATS
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 8] <- max(
        teacher_export_subj_12[i, 11],
        teacher_export_subj_12[i, 12],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge NORW
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 9] <- max(
        teacher_export_subj_12[i, 13],
        teacher_export_subj_12[i, 14],
        na.rm = T
    )
```

```r
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge NORO
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 10] <- max(
        teacher_export_subj_12[i, 15],
        teacher_export_subj_12[i, 16],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge RELI
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12[i, 11] <- max(
        teacher_export_subj_12[i, 17],
        teacher_export_subj_12[i, 18],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Reset progress bar
n_iter <- dim(teacher_export_12)[1] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)

# Merge SOCS
for (i in 1:n_iter) {
    # Insert progress bar here
    pb$tick() # Update progress bar
```

```r
    subj_12[i, 12] <- max(
        teacher_export_subj_12[i, 19],
        teacher_export_subj_12[i, 20],
        na.rm = T
    )
}
cat("\n") # Start a new line once progress bar is full

# Turn -Inf to NA column-by-column
# Create a placeholder matrix
subj_12_clean <- matrix(NA, nrow = dim(subj_12)[1], ncol = dim(subj_12)[2])
n_iter <- dim(subj_12_clean)[2] # Set the progress bar's end point
pb <- progress_bar$new( # Refresh progress bar's internal definition
    format = "(:spin) [:bar] :percent [Elapsed time: :elapsedfull || Estimated time remaining: :
    ↪ eta]",
    total = n_iter,
    complete = "=",
    incomplete = "-",
    current = ">",
    clear = F,
    width = 100
)
for (j in 1:n_iter) { # 12 cycles
    # Insert progress bar here
    pb$tick() # Update progress bar

    subj_12_clean[, j] <- car::recode(subj_12[, j], "
        '-Inf' = NA
    ")
}
subj_12_clean <- data.frame(subj_12_clean)
names(subj_12_clean) <- names(subj_12)

# Re-order subjects
subj_12_clean <- subj_12_clean[, c(9,10,1,2,5,8,12,4,7,6,3,11)]
# New order:
#   NORW: Norwegian (written)
#   NORO: Norwegian (oral)
#   ENGW: English (wrItten)
#   ENGO: English (oral)
#   MATH: Mathematics
#   NATS: Natural Sciences
#   SOCS: Social Sciences
#   PHED: Physical Education
#   MUSI: Music
#   FOOD: Food and Health
#   HAND: Arts and Handcraft
#   RELI: Religion

# Count the number of missings for each student
missing_12 <- rowSums(is.na(subj_12_clean)) # Total number of missings
missing_7 <- rowSums(is.na(subj_12_clean[, c(1:7)])) # 7 major subjects
missing_5 <- rowSums(is.na(subj_12_clean[, c(8:12)])) # 5 minor subjects

# Stitch admin, missing counts and marks together
teacher_final <- cbind(teacher_export_12[, c(1:8)], # Admin variables
    missing_12, missing_7, missing_5,               # Missing counts
    subj_12_clean                                   # Teacher-assigned marks
)

# Save teacher_final
data.table::fwrite(teacher_final, "./Rolf/60618.csv", row.names = F)

# Keep students with 4 or more of the 7-major subjects

major_4_plus <- teacher_final[which(teacher_final$missing_7 < 4), ]
if (interactive()) {dim(major_4_plus)} # 59,517 students remain

# Compute data loss rate (n = 1,101, % = 1.82)
if (interactive()) {
    dim(teacher_final)[1] - dim(major_4_plus)[1]
    round((dim(teacher_final)[1] - dim(major_4_plus)[1]) / dim(teacher_final)[1] * 100, 2)
}
```

```r

# Save major_4_plus to an external file
data.table::fwrite(major_4_plus[, -c(9:11)], "./Rolf/59517.csv", row.names = F)

# Keep students with 3 or more of the 5-minor subjects

minor_3_plus <- major_4_plus[which(major_4_plus$missing_5 < 3), ]
dim(minor_3_plus) # 57,730 students remain

# Compute data loss rate (n = 1,787, % = 3)
if (interactive()) {
    dim(major_4_plus)[1] - dim(minor_3_plus)[1]
    round((dim(major_4_plus)[1] - dim(minor_3_plus)[1]) / dim(major_4_plus)[1] * 100, 2)
}

# Save minor_3_plus to an external file
data.table::fwrite(minor_3_plus[, -c(9:11)], "./Rolf/57730.csv", row.names = F)
```

Subject Difficulty Analysis using GPCM

```r
# Load R package ‘mirt‘
suppressWarnings(suppressMessages(library(mirt)))

# Generalised partial credit model
gpcm <- mirt(difficulty[,c(9:20)], itemtype = "gpcm", SE = T)
coef(gpcm, printSE = T, IRTpars = T)
data.table::fwrite(coef(gpcm, printSE = T, IRTpars = T, as.data.frame = T),
    "./Rolf/parameter.csv",
    now.names = T
)

# Save subjects' codes and names
subj_code <- names(difficulty)[-c(1:8)]
subj_name <- c(
    "Written Norwegian",
    "Oral Norwegian",
    "Written English",
    "Oral English",
    "Mathematics",
    "Natural Sciences",
    "Social Sciences",
    "Physical Education",
    "Music",
    "Food and Health",
    "Arts and Handcraft",
    "Religion"
)

# Item characteristic curves
# Auto-print is off in loops, causing corrupted PDFs. Insert print().
for (i in 1:12) {
    pdf(file = paste0("./Rolf/trace/trace_", subj_code[i], ".pdf"))
    print(directlabels::direct.label(
        itemplot(gpcm, item = i, type = 'trace',
            xlim = c(-6.5,6.5),
            main = paste0(
                "Trace Plot for ", subj_code[i], " (", subj_name[i], ")"
            )
        ), 'top.points'
    ))
    dev.off()
}

# Expected scores
for (i in 1:12) {
    pdf(file = paste0("./Rolf/score/score_", subj_code[i], ".pdf"))
    print(itemplot(gpcm, item = i, type = 'score', CE = T,
            xlim = c(-6.5,6.5),
            main = paste0(
                "Expected Score for ", subj_code[i], " (", subj_name[i], ")"
            )
```

```
    ))
    dev.off()
}

# Information and standard errors
for (i in 1:12) {
    pdf(file = paste0("./Rolf/info/infoSE_", subj_code[i], ".pdf"))
    print(itemplot(gpcm, item = i, type = 'infoSE', CE = T,
            xlim = c(-6.5,6.5),
            main = paste0(
                "Information and SE for ", subj_code[i], " (", subj_name[i], ")"
            )
    ))
    dev.off()
}
```

IRT Analysis Output

Table 1

Partial Credit Model (PCM) Parameter Estimates

Subject Code	Subject Name	b_1	b_2	b_3	b_4	b_5
NORW	Written Norwegian	−5.953 (0.065)	−3.062 (0.021)	−0.805 (0.015)	1.205 (0.015)	3.605 (0.024)
NORO	Oral Norwegian	−6.313 (0.087)	−3.724 (0.026)	−1.633 (0.016)	0.242 (0.014)	2.622 (0.018)
ENGW	Written English	−5.189 (0.049)	−2.967 (0.021)	−1.017 (0.015)	0.957 (0.015)	3.138 (0.021)
ENGO	Oral English	−5.655 (0.067)	−3.618 (0.026)	−1.788 (0.016)	0.310 (0.014)	2.618 (0.018)
MATH	Mathematics	−4.934 (0.037)	−1.856 (0.017)	−0.213 (0.015)	1.111 (0.016)	3.028 (0.021)
NATS	Natural Sciences	−5.724 (0.059)	−3.085 (0.022)	−1.226 (0.016)	0.345 (0.014)	2.433 (0.018)
SOCS	Social Sciences	−6.065 (0.074)	−3.519 (0.025)	−1.638 (0.016)	0.071 (0.014)	2.333 (0.017)
RELI	Religion and Ethics	−5.822 (0.066)	−3.374 (0.024)	−1.583 (0.016)	0.152 (0.014)	2.325 (0.017)
MUSI	Music	−6.305 (0.104)	−4.288 (0.034)	−2.471 (0.018)	−0.198 (0.014)	2.557 (0.017)
HAND	Arts and Handcraft	−6.447 (0.117)	−4.599 (0.037)	−2.522 (0.018)	−0.143 (0.014)	2.687 (0.018)
FOOD	Food and Health	−7.768 (0.274)	−5.455 (0.050)	−2.791 (0.019)	−0.336 (0.014)	2.485 (0.017)
PHED	Physical Education	−6.221 (0.114)	−4.607 (0.041)	−3.035 (0.021)	−0.730 (0.014)	2.265 (0.016)

Note. A partial credit model (PCM) computes the difficulty (b) parameters for each subject. Standard errors are enclosed in parenthesis below point estimates. All estimates are significant at .001 level.

Table 2

Generalised Partial Credit Model (GPCM) Parameter Estimates

Subject Code	Subject Name	a	b_1	b_2	b_3	b_4	b_5
NORW	Written Norwegian	3.021 (0.025)	−2.882 (0.026)	−1.535 (0.010)	−0.403 (0.006)	0.627 (0.007)	1.789 (0.011)
NORO	Oral Norwegiani	3.346 (0.028)	−3.024 (0.031)	−1.845 (0.011)	−0.817 (0.007)	0.154 (0.006)	1.300 (0.008)
ENGW	Written English	1.790 (0.015)	−2.875 (0.029)	−1.638 (0.013)	−0.559 (0.009)	0.534 (0.008)	1.741 (0.013)
ENGO	Oral English	1.689 (0.014)	−3.185 (0.041)	−2.024 (0.016)	−1.001 (0.010)	0.175 (0.008)	1.473 (0.011)
MATH	Mathematics	1.715 (0.014)	−2.773 (0.024)	−1.027 (0.010)	−0.113 (0.009)	0619 (0.009)	1.694 (0.013)
NATS	Natural Sciences	2.656 (0.022)	−2.877 (0.026)	−1.602 (0.010)	−0.640 (0.007)	0.206 (0.007)	1.266 (0.009)
SOCS	Social Sciences	3.397 (0.028)	−2.920 (0.027)	−1.766 (0.010)	−0.826 (0.007)	0.074 (0.006)	1.170 (0.008)
RELI	Religion and Ethics	3.154 (0.026)	−2.850 (0.025)	−1.715 (0.011)	−0.800 (0.007)	0.108 (0.006)	1.181 (0.008)
MUSI	Music	1.331 (0.011)	−3.832 (0.077)	−2.558 (0.026)	−1.493 (0.014)	−0.140 (0.009)	1.559 (0.013)
HAND	Arts and Handcraft	1.138 (0.010)	−4.129 (0.101)	−2.924 (0.032)	−1.616 (0.016)	−0.124 (0.010)	1.751 (0.016)
FOOD	Food and Health	1.429 (0.012)	−4.565 (0.173)	−3.263 (0.037)	−1.644 (0.014)	−0.208 (0.009)	1.479 (0.012)
PHED	Physical Education	0.804 (0.008)	−4.483 (0.138)	−3.264 (0.050)	−2.295 (0.026)	−0.658 (0.015)	1.750 (0.019)

Note. A generalised partial credit model (GPCM) computes the discrimination (a) and difficulty (b) parameters for each subject. Standard errors are enclosed in parenthesis below point estimates. All estimates are significant at .001 level.