Biosciences_Project_RMarkdown_Code_Version1

Haya Deeb

2024-01-23

#Impot the libraries and the Dataset

```
#import the used library and the csv Data File

library(dplyr)
```

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

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

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

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

```
library(ggplot2)
library(tidyr)
library(scales)
library(MASS)
```

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

##
## Attaching package: 'MASS'

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

```
data <- read.csv ("Biosciences_Publications_Dataset_2024.csv")
head(data)
```

```
##
## 1                    ArdA proteins from different mobile genetic elements can bind to the EcoKI Type
## 2                                          Bacteria repelling poly(methylmethacrylate-co-dimethylacryl
## 3                                Exosomes secreted by nematode parasites transfer small RNAs to mammal
## 4                                                    Parasite-Derived MicroRNAs in Host Serum As
## 5 Characterisation of Translation Elongation Factor eEF1B Subunit Expression in Mammalian Cells and
## 6                                    The translation elongation factor eEF1A1 couples transcription to
##                                          DOI  ResGrp Type
## 1 https://doi.org/10.1016/j.bbapap.2013.12.008 Blakely InfD
## 2           https://doi.org/10.1039/C4TB01129E Blakely InfD
## 3   https://www.nature.com/articles/ncomms6488    Buck InfD
## 4 https://doi.org/10.1371/journal.pntd.0002701    Buck InfD
## 5                 10.1371/journal.pone.0114117  Abbott  NCD
## 6                         10.7554/eLife.03164  Abbott  NCD
##                                       Institution
## 1         Edinburgh Infectious Diseases Organisation
## 2         Edinburgh Infectious Diseases Organisation
## 3         Edinburgh Infectious Diseases Organisation
## 4         Edinburgh Infectious Diseases Organisation
## 5 IGC (Centre for Genomic and Experimental Medicine)
## 6 IGC (Centre for Genomic and Experimental Medicine)
##                                                                                Journal
## 1 Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics (ScienceDirect)
## 2  Journal of Materials Chemistry B            (The Royal Society of Chemistry)
## 3                              Nature Communications                    (Nature)
## 4        PLOS Neglected Tropical Diseases      (Public Library of Science PLOS)
## 5                                                                      PLoS ONE
## 6                                                                         eLife
##   Year AnalysisPgrm CodeArchived DAS CorresAuthor Preprints Complete Reuse
## 1 2014            1           NA  NA            0         0        2     3
## 2 2014            1           NA  NA            0         0        2     2
## 3 2014            1            0  NA            1         0        2     4
## 4 2014            1            1  NA            1         0        4     4
## 5 2014            1           NA   1            0         0        1     1
## 6 2014            1           NA  NA            0         0        1     1
##   Access Licence Image Genomics Rescored
## 1      2       4     0        0        0
## 2      2       4     0       NA        0
## 3      4       4    NA        1        0
## 4      4       4    NA        1        0
## 5      1       1     0       NA        0
## 6      1       1     0       NA        0
```

#1 - Descriptive Statistics

#How many papers in each type

```
type_frequency <- table(data$Type)
print(type_frequency)
```

```
##
```

```
## InfD  NCD
##   82  111
```

#frequency and percentage of paper in each year

```
total_counts_by_type <- data %>%
  group_by(Type) %>%
  summarise(Total = n(), .groups = "drop")

# Calculate frequency and percentage for each Type and Year
summarized_data <- data %>%
  group_by(Year, Type) %>%
  summarise(count = n(), .groups = "drop") %>%
  left_join(total_counts_by_type, by = "Type") %>%
  mutate(Percentage = (count / Total) * 100)

print(summarized_data)
```

```
## # A tibble: 18 x 5
##     Year Type  count Total Percentage
##    <int> <chr> <int> <int>      <dbl>
##  1  2014 InfD     13    82      15.9
##  2  2014 NCD       7   111       6.31
##  3  2015 InfD      6    82       7.32
##  4  2015 NCD       9   111       8.11
##  5  2016 InfD      6    82       7.32
##  6  2016 NCD       7   111       6.31
##  7  2017 InfD      9    82      11.0
##  8  2017 NCD      17   111      15.3
##  9  2018 InfD     10    82      12.2
## 10  2018 NCD       8   111       7.21
## 11  2019 InfD     12    82      14.6
## 12  2019 NCD      15   111      13.5
## 13  2020 InfD     11    82      13.4
## 14  2020 NCD      11   111       9.91
## 15  2021 InfD      7    82       8.54
## 16  2021 NCD      18   111      16.2
## 17  2022 InfD      8    82       9.76
## 18  2022 NCD      19   111      17.1
```

Calculate the total number of papers for each year

```
yearly_totals <- data %>%
  group_by(Year) %>%
  summarise(Total = n(), .groups = "drop")

# Calculate the grand total of all papers
grand_total <- sum(yearly_totals$Total)

# Add a column for the percentage of each year's total relative to the grand total
```

```r
yearly_totals <- yearly_totals %>%
  mutate(Percentage = (Total / grand_total) * 100)

# Print the yearly totals and percentages
print(yearly_totals)
```

```
## # A tibble: 9 x 3
##    Year Total Percentage
##   <int> <int>      <dbl>
## 1  2014    20      10.4
## 2  2015    15       7.77
## 3  2016    13       6.74
## 4  2017    26      13.5
## 5  2018    18       9.33
## 6  2019    27      14.0
## 7  2020    22      11.4
## 8  2021    25      13.0
## 9  2022    27      14.0
```

#to categorize papers based on the year of publication compared to FAIR principles implementation 2016 (overall)

```r
data <- data %>%
  mutate(Period = ifelse(Year <= 2016, "Before or in 2016", "After 2016"))

# Calculate the total number of papers for each period
period_totals <- data %>%
  group_by(Period) %>%
  summarise(Total = n(), .groups = "drop")

# Calculate the grand total of all papers
grand_total <- sum(period_totals$Total)

# Add a column for the percentage of each period's total relative to the grand total
period_totals <- period_totals %>%
  mutate(Percentage = (Total / grand_total) * 100)

# Print the period totals and percentages
print(period_totals)
```

```
## # A tibble: 2 x 3
##   Period            Total Percentage
##   <chr>             <int>      <dbl>
## 1 After 2016          145       75.1
## 2 Before or in 2016    48       24.9
```

#to categorize papers based on the year of publication compared to FAIR principles implementation 2016 (depend on type)

```r
#Frequency and Percantage of paper before and After FAIR principles (2016)
total_counts_by_type <- data %>%
  group_by(Type) %>%
```

```r
  summarise(Total = n(), .groups = "drop")

# Categorize, calculate frequency and percentage for each Type based on year
summarized_data <- data %>%
  mutate(Category = ifelse(Year <= 2016, "On/Before 2016", "After 2020")) %>%
  group_by(Category, Type) %>%
  summarise(count = n(), .groups = "drop") %>%
  left_join(total_counts_by_type, by = "Type") %>%
  mutate(Percentage = (count / Total) * 100)

print(summarized_data)
```

```
## # A tibble: 4 x 5
##   Category       Type  count Total Percentage
##   <chr>          <chr> <int> <int>      <dbl>
## 1 After 2020     InfD     57    82       69.5
## 2 After 2020     NCD      88   111       79.3
## 3 On/Before 2016 InfD     25    82       30.5
## 4 On/Before 2016 NCD      23   111       20.7
```

#to categorise papers based on the year of publication compared to COVID 19 - 2020 (overall)

```r
data <- data %>%
  mutate(Period = ifelse(Year <= 2020, "Before or in 2020", "After 2020"))

# Calculate the total number of papers for each period
period_totals <- data %>%
  group_by(Period) %>%
  summarise(Total = n(), .groups = "drop")

# Calculate the grand total of all papers
grand_total <- sum(period_totals$Total)

# Add a column for the percentage of each period's total relative to the grand total
period_totals <- period_totals %>%
  mutate(Percentage = (Total / grand_total) * 100)

# Print the period totals and percentages
print(period_totals)
```

```
## # A tibble: 2 x 3
##   Period            Total Percentage
##   <chr>             <int>      <dbl>
## 1 After 2020           52       26.9
## 2 Before or in 2020   141       73.1
```

#to categorise papers based on the year of publication compared to COVID 19 - 2020 (NCD vs InfD)

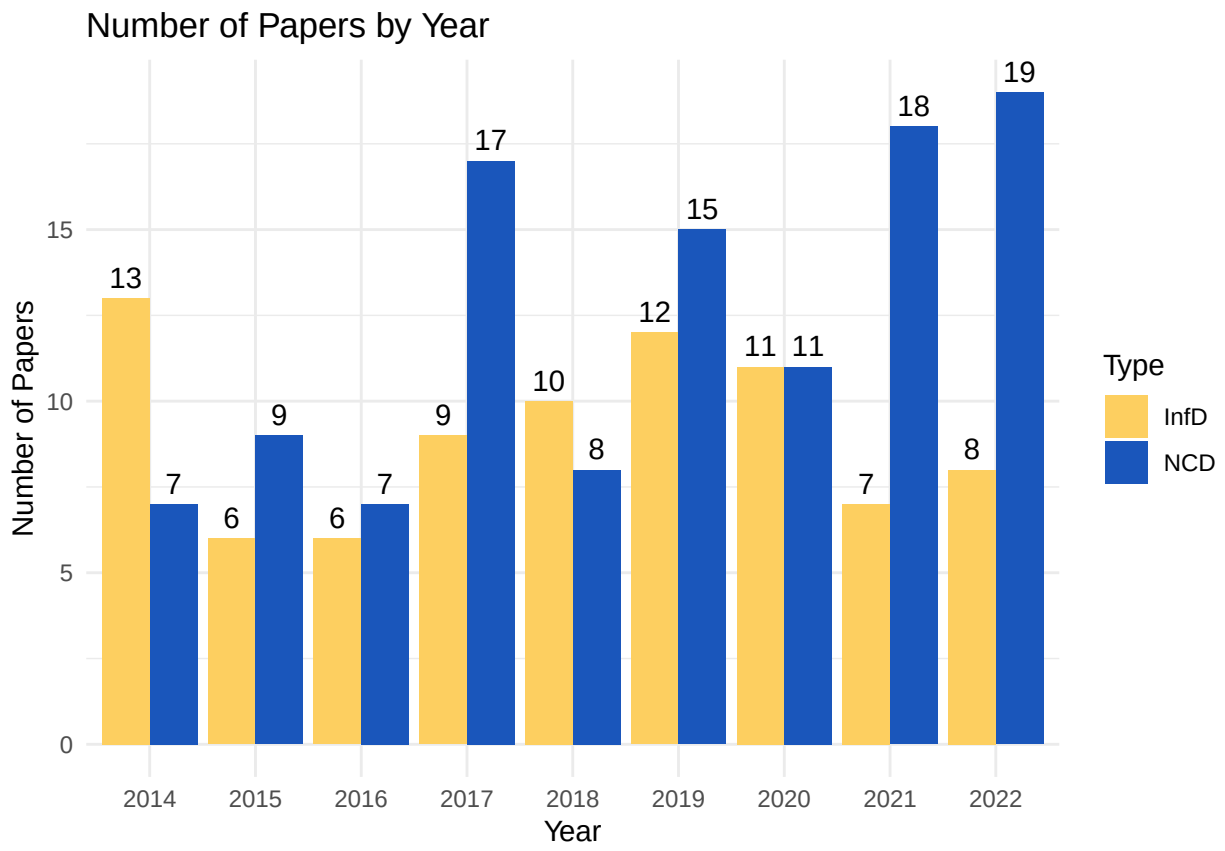
```r
total_counts_by_type <- data %>%
  group_by(Type) %>%
  summarise(Total = n(), .groups = "drop")
```

```r
# Categorize, calculate frequency and percentage for each Type based on year
summarized_data <- data %>%
  mutate(Category = ifelse(Year <= 2020, "On/Before 2020", "After 2020")) %>%
  group_by(Category, Type) %>%
  summarise(count = n(), .groups = "drop") %>%
  left_join(total_counts_by_type, by = "Type") %>%
  mutate(Percentage = (count / Total) * 100)

print(summarized_data)
```

```
## # A tibble: 4 x 5
##   Category       Type  count Total Percentage
##   <chr>          <chr> <int> <int>      <dbl>
## 1 After 2020     InfD     15    82       18.3
## 2 After 2020     NCD      37   111       33.3
## 3 On/Before 2020 InfD     67    82       81.7
## 4 On/Before 2020 NCD      74   111       66.7
```

#Plot the distribution of the included papers over the years

```r
aggregated_data <- data %>%
  group_by(Year, Type) %>%
  summarise(count = n(), .groups = "drop")

# Create the clustered bar chart with numbers on top of the bars and specified colors
year <- ggplot(aggregated_data, aes(x = as.factor(Year), y = count, fill = Type)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = count), position = position_dodge(width = 0.9), vjust = -0.5) +
  scale_fill_manual(values = c("#FDCF60", "#1A56BB")) +
  labs(x = "Year", y = "Number of Papers", title = "Number of Papers by Year") +
  theme_minimal()

print(year)
```




```
ggsave("year.png", year, width = 10, height = 8, dpi = 300, bg="white")
```

Frequency and Percentage of the papers that share data (Data Sharing= Data Completeness score >1) Overall

```
data <- data %>%
  mutate(Period = ifelse(Complete > 1, "Completeness = 1", "Completeness > 1"))

# Calculate the total number of papers for each period
period_totals <- data %>%
  group_by(Period) %>%
  summarise(Total = n(), .groups = "drop")

# Calculate the grand total of all papers
grand_total <- sum(period_totals$Total)

# Add a column for the percentage of each period's total relative to the grand total
period_totals <- period_totals %>%
  mutate(Percentage = (Total / grand_total) * 100)

# Print the period totals and percentages
print(period_totals)
```

```
## # A tibble: 2 x 3
##   Period           Total Percentage
##   <chr>            <int>      <dbl>
## 1 Completeness = 1   151       78.2
## 2 Completeness > 1    42       21.8
```

Frequency and Percentage of the papers that share data (Data Sharing= Data Completeness score >1) by Type

```
total_counts_by_type <- data %>%
  group_by(Type) %>%
  summarise(Total = n(), .groups = "drop")

# Categorize based on Completeness, calculate frequency and percentage for each Type
summarized_data <- data %>%
  filter(Complete == 1 | Complete > 1) %>%
  mutate(Category = ifelse(Complete == 1, "Completeness = 1", "Completeness > 1")) %>%
  group_by(Category, Type) %>%
  summarise(count = n(), .groups = "drop") %>%
  left_join(total_counts_by_type, by = "Type") %>%
  mutate(Percentage = (count / Total) * 100)

print(summarized_data)
```

```
## # A tibble: 4 x 5
##   Category         Type  count Total Percentage
##   <chr>            <chr> <int> <int>      <dbl>
## 1 Completeness = 1 InfD     19    82       23.2
## 2 Completeness = 1 NCD      23   111       20.7
## 3 Completeness > 1 InfD     63    82       76.8
## 4 Completeness > 1 NCD      88   111       79.3
```

#Chi square test to assess if there is any difference in data share between the two types (InfD vs NCD)

```
# Create a contingency table
table_completeness <- table(data$Type, data$Complete > 1)

# Perform the Chi-square test
result <- chisq.test(table_completeness)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_completeness
## X-squared = 0.053503, df = 1, p-value = 0.8171
```

```r
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##           FALSE     TRUE
##   InfD 17.84456 64.15544
##   NCD  24.15544 86.84456
```

```r
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

#Frequemcy and percentage of preprints

```r
# Calculate frequency and percentage for each Preprints category, across all types
preprints_summary <- data %>%
  group_by(Preprints) %>%
  summarise(count = n(), .groups = "drop")

# Calculate the overall total
overall_total <- sum(preprints_summary$count)

# Add a column for the percentage of each Preprints category relative to the overall total
preprints_summary <- preprints_summary %>%
  mutate(percentage = (count / overall_total) * 100)

# Print the summary
print(preprints_summary)
```

```
## # A tibble: 2 x 3
##   Preprints count percentage
##       <int> <int>      <dbl>
## 1         0   149       77.2
## 2         1    44       22.8
```

```r
# Frequency and Percentage of the papers with preprints
data %>%
  group_by(Type, Preprints) %>%
  summarise(count = n(), .groups = "drop") %>%
  group_by(Type) %>%
  mutate(total = sum(count),
         percentage = (count/total)*100) %>%
  ungroup() %>%
  arrange(Type, Preprints)
```

```
## # A tibble: 4 x 5
##   Type  Preprints count total percentage
##   <chr>     <int> <int> <int>      <dbl>
## 1 InfD          0    65    82       79.3
## 2 InfD          1    17    82       20.7
## 3 NCD           0    84   111       75.7
## 4 NCD           1    27   111       24.3
```

#Chi square test to assess if there is any difference in preprint between the two types (InfD, NCD)

```
# Create a contingency table
table_preprint <- table(data$Type, data$Preprints)

# Perform the Chi-square test
result <- chisq.test(table_preprint)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_preprint
## X-squared = 0.17184, df = 1, p-value = 0.6785
```

```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##              0       1
##   InfD 63.3057 18.6943
##   NCD  85.6943 25.3057
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

Calculate frequency and percentage for DAS

```
DAS_summary <- data %>%
  group_by(DAS) %>%
  summarise(count = n(), .groups = "drop")
```

```r
# Calculate the overall total
overall_total <- sum(DAS_summary$count)

# Add a column for the percentage of each DAS category relative to the overall total
DAS_summary <- DAS_summary %>%
  mutate(percentage = (count / overall_total) * 100)

print(DAS_summary)
```

```
## # A tibble: 3 x 3
##     DAS count percentage
##   <int> <int>      <dbl>
## 1     0    27       14.0
## 2     1    75       38.9
## 3    NA    91       47.2
```

```r
data$DAS <- as.factor(data$DAS)

# Frequency and Percentage of the papers with DAS in the publications
data %>%
  group_by(Type, DAS) %>%
  summarise(count = n(), .groups = "drop") %>%
  group_by(Type) %>%
  mutate(total = sum(count),
         percentage = (count/total)*100) %>%
  ungroup() %>%
  arrange(Type, DAS)
```

```
## # A tibble: 6 x 5
##   Type  DAS   count total percentage
##   <chr> <fct> <int> <int>      <dbl>
## 1 InfD  0         8    82       9.76
## 2 InfD  1        35    82      42.7
## 3 InfD  <NA>     39    82      47.6
## 4 NCD   0        19   111      17.1
## 5 NCD   1        40   111      36.0
## 6 NCD   <NA>     52   111      46.8
```

#Chi square test to assess if there is any difference in DAS between the two types (InfD, NCD)

```r
# Create a contingency table
table_das <- table(data$Type, data$DAS)

# Perform the Chi-square test
result <- chisq.test(table_das)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
```

```
## data:  table_das
## X-squared = 1.7161, df = 1, p-value = 0.1902
```

```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##                0        1
##   InfD 11.38235 31.61765
##   NCD  15.61765 43.38235
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

Calculate frequency and percentage for Code Sharing

```
CodeArchived_summary <- data %>%
  group_by(CodeArchived) %>%
  summarise(count = n(), .groups = "drop")

# Calculate the overall total
overall_total <- sum(CodeArchived_summary$count)

# Add a column for the percentage of each DAS category relative to the overall total
CodeArchived_summary <- CodeArchived_summary %>%
  mutate(percentage = (count / overall_total) * 100)

print(CodeArchived_summary)
```

```
## # A tibble: 3 x 3
##   CodeArchived count percentage
##          <int> <int>      <dbl>
## 1            0    21       10.9
## 2            1    48       24.9
## 3           NA   124       64.2
```

```
# Frequency and Percentage of the papers that shared the code used
data %>%
  group_by(Type, CodeArchived) %>%
  summarise(count = n(), .groups = "drop") %>%
  group_by(Type) %>%
  mutate(total = sum(count),
```

```
        percentage = (count/total)*100) %>%
  ungroup() %>%

  arrange(Type, CodeArchived)
```

```
## # A tibble: 6 x 5
##   Type  CodeArchived count total percentage
##   <chr>        <int> <int> <int>      <dbl>
## 1 InfD             0     8    82       9.76
## 2 InfD             1    14    82      17.1
## 3 InfD            NA    60    82      73.2
## 4 NCD              0    13   111      11.7
## 5 NCD              1    34   111      30.6
## 6 NCD             NA    64   111      57.7
```

#Chi square test to assess if there is any difference in Code Sharing between the two types (InfD, NCD)

```
# Create a contingency table
table_code <- table(data$Type, data$CodeArchived)

# Perform the Chi-square test
result <- chisq.test(table_code)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table_code
## X-squared = 0.20392, df = 1, p-value = 0.6516
```

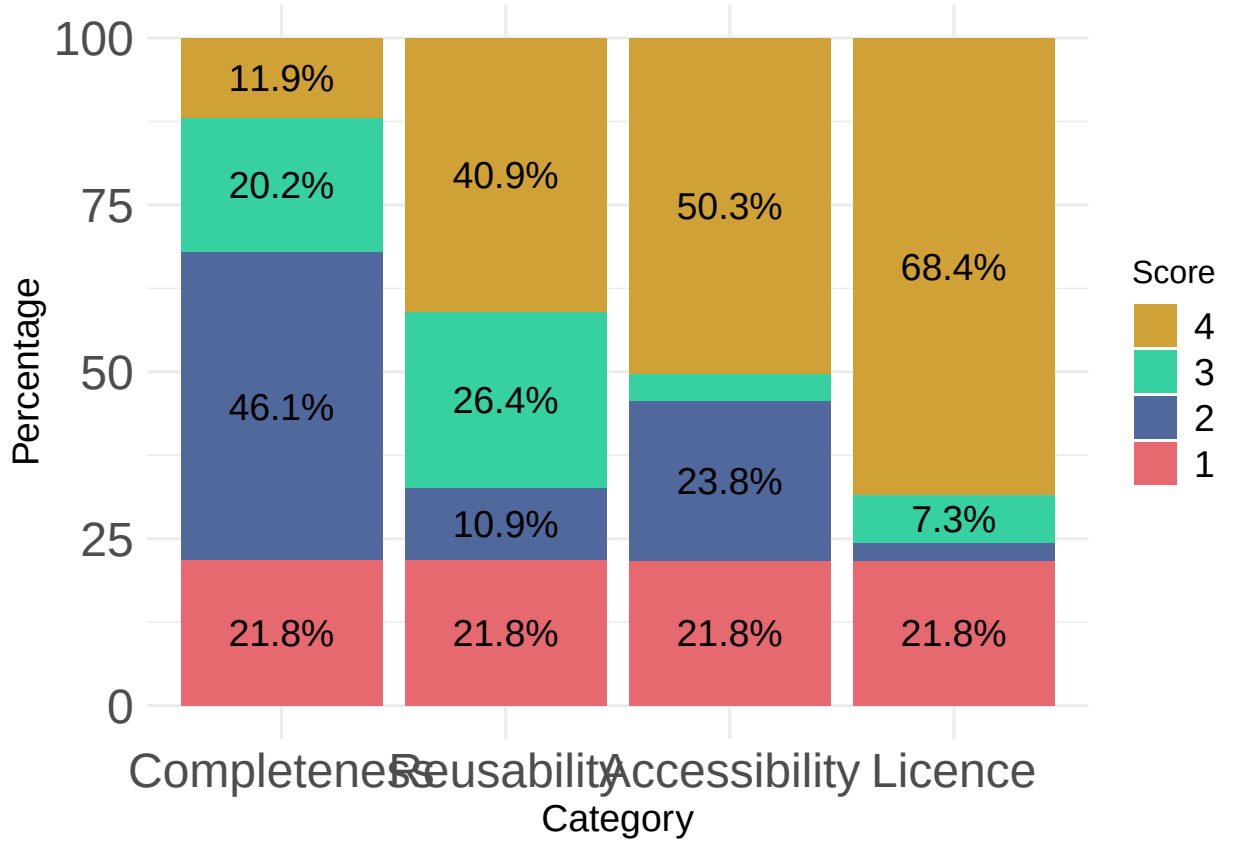
```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##                0        1
##   InfD  6.695652 15.30435
##   NCD  14.304348 32.69565
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

#Plot the 4 scoring criteria for all the papers

```r
# Convert all relevant columns to factors first to avoid the error
data <- data %>%
  mutate(across(c(Complete, Reuse, Access, Licence), as.character), .groups = "drop")  # Converts to ch

# Reshape data to long format
long_data <- data %>%
  pivot_longer(cols = c(Complete, Reuse, Access, Licence), names_to = "Category", values_to = "Value") %
  mutate(Value = as.character(Value))  # Ensure that Value is a character

# Rename and reorder categories
long_data$Category <- recode(long_data$Category,
                             "Complete" = "Completeness",
                             "Reuse" = "Reusability",
                             "Access" = "Accessibility",
                             "Licence" = "Licence")
long_data$Category <- factor(long_data$Category, levels = c("Completeness", "Reusability", "Accessibili

long_data <- long_data %>%
  mutate(Value = factor(Value, levels = c("4", "3", "2", "1")))

# Calculate counts and percentages
long_data <- long_data %>%
  group_by(Category, Value) %>%
  summarise(Count = n(), .groups = "drop") %>%
  ungroup() %>%
  group_by(Category) %>%
  mutate(Total = sum(Count),
         Percentage = Count / Total * 100) %>%
  ungroup() %>%
  arrange(Category, desc(Value)) %>%
  group_by(Category) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), ""),
         CumPercentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

# Plot
plot_all <- ggplot(long_data, aes(x = Category, y = Percentage, fill = Value)) +
  geom_bar(stat = "identity", position = "stack") +
  scale_fill_manual(values = c("#D0A136", "#36D0A1","#50689B","#E5696F" )) +
  labs(y = "Percentage", fill = "Score") +

  geom_text(aes(label = Label, y = CumPercentage), size = 5) +
  theme_minimal() +
   theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 18),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  ) +
  ylim(0, 100)

# Show the plot
print(plot_all)
```

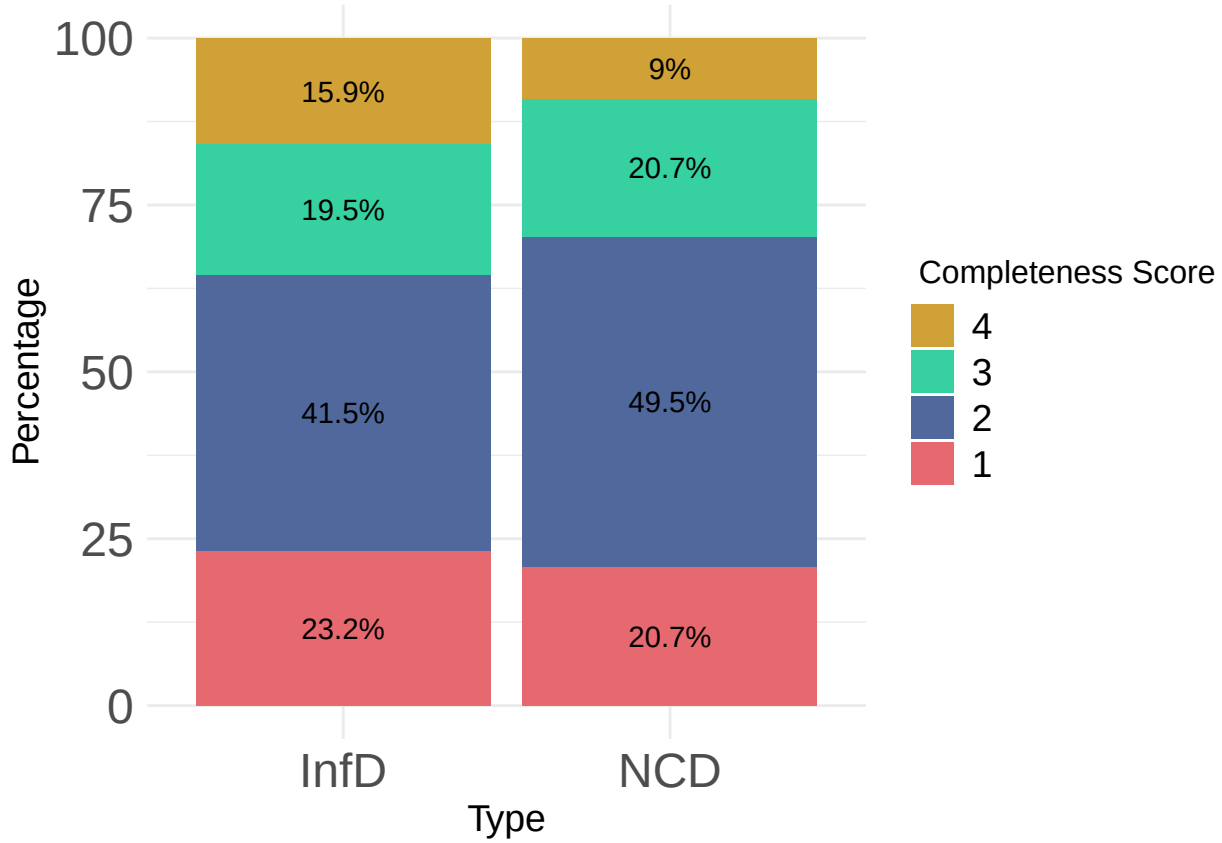



```
# Save the plot
ggsave("plotall.png", plot_all, width = 10, height = 8, dpi = 300, bg="white")
```

#Cluster bar chart fro the completeness criteria by the Type of the study

```
# Summarize data
summarized_data <- data %>%
  group_by(Type, Complete) %>%
  summarise(Count = n(), .groups = "drop") %>%
  left_join(data %>% group_by(Type) %>% summarise(Total = n(), .groups = "drop"), by = "Type") %>%
  mutate(Percentage = Count/Total * 100)

summarized_data <- summarized_data %>%
  mutate(Complete = factor(Complete, levels = c("4", "3", "2", "1")))

# For cumulative percentages
summarized_data <- summarized_data %>%
  arrange(Type, desc(Complete)) %>%
  group_by(Type) %>%
  mutate(CumPercentage = cumsum(Percentage))

# Create the stacked bar chart for Completeness with colors and annotations
gcomp <- ggplot(summarized_data, aes(x = Type, y = Percentage, fill = as.factor(Complete))) +
  geom_bar(stat = "identity", position = "stack") +
  scale_fill_manual(values = c("#D0A136", "#36D0A1","#50689B","#E5696F")) +
```

```
  labs(x = "Type", y = "Percentage", fill = " Completeness Score") +
  theme_minimal() +
  geom_text(aes(label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), ""),
                y = ifelse(Complete == 1, Percentage / 2, CumPercentage - (0.5 * Percentage))),
            color = "Black", size = 4) +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 18),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  ) +
  ylim(0, 100)


print(gcomp)
```




```
ggsave("gcomp.png", gcomp, width = 10, height = 8, dpi = 300)
```

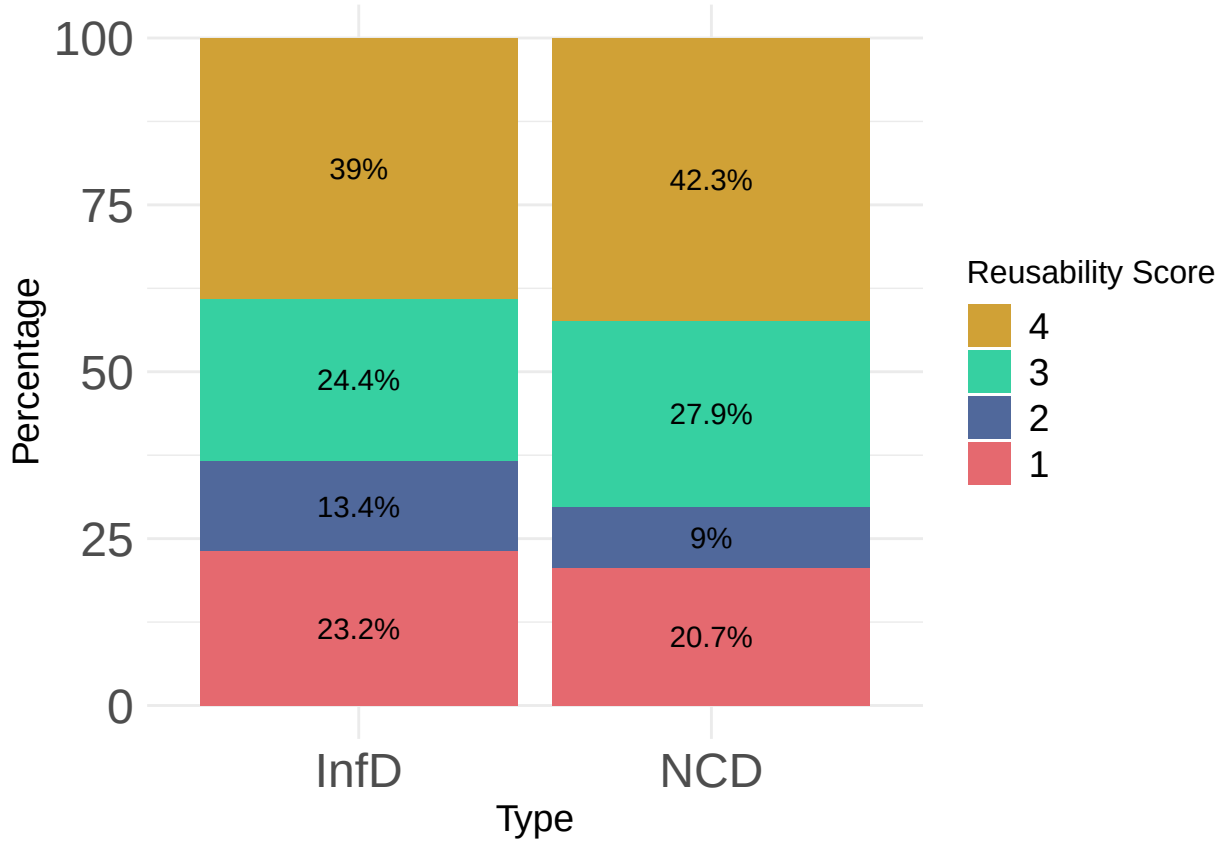
##Cluster bar chart fro the Reuse criteria by the Type of the study

```
summarized_data_reuse <- data %>%
  group_by(Type, Reuse) %>%
  summarise(Count = n(), .groups = "drop") %>%
  left_join(data %>% group_by(Type) %>% summarise(Total = n(), .groups = "drop"), by = "Type") %>%
```

```r
  mutate(Percentage = Count/Total * 100)

summarized_data_reuse <- summarized_data_reuse %>%
  mutate(Reuse = factor(Reuse, levels = c("4", "3", "2", "1")))


summarized_data_reuse <- summarized_data_reuse %>%
  arrange(Type, desc(Reuse)) %>%
  group_by(Type) %>%
  mutate(CumPercentage = cumsum(Percentage))
print(summarized_data_reuse)
```

```
## # A tibble: 8 x 6
## # Groups:   Type [2]
##   Type  Reuse Count Total Percentage CumPercentage
##   <chr> <fct> <int> <int>      <dbl>         <dbl>
## 1 InfD  1        19    82      23.2           23.2
## 2 InfD  2        11    82      13.4           36.6
## 3 InfD  3        20    82      24.4           61.0
## 4 InfD  4        32    82      39.0          100
## 5 NCD   1        23   111      20.7           20.7
## 6 NCD   2        10   111       9.01          29.7
## 7 NCD   3        31   111      27.9           57.7
## 8 NCD   4        47   111      42.3          100
```

```r
colors <- c( "#D0A136", "#36D0A1","#50689B","#E5696F")

# Visualization
greuse <- ggplot(summarized_data_reuse, aes(x = Type, y = Percentage, fill = as.factor(Reuse))) +
  geom_bar(stat = "identity", position = "stack") +
  scale_fill_manual(values = colors) +
  labs(x = "Type", y = "Percentage", fill = "Reusability Score") +
  theme_minimal() +
  geom_text(aes(label = paste0(round(Percentage, 1), "%"),
                y = CumPercentage - (0.5 * Percentage)),
            color = "Black", size = 4) +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 18),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  ) +
  ylim(0, 100)

print(greuse)
```




```
ggsave("greuse.png", greuse, width = 10, height = 8, dpi = 300)
```

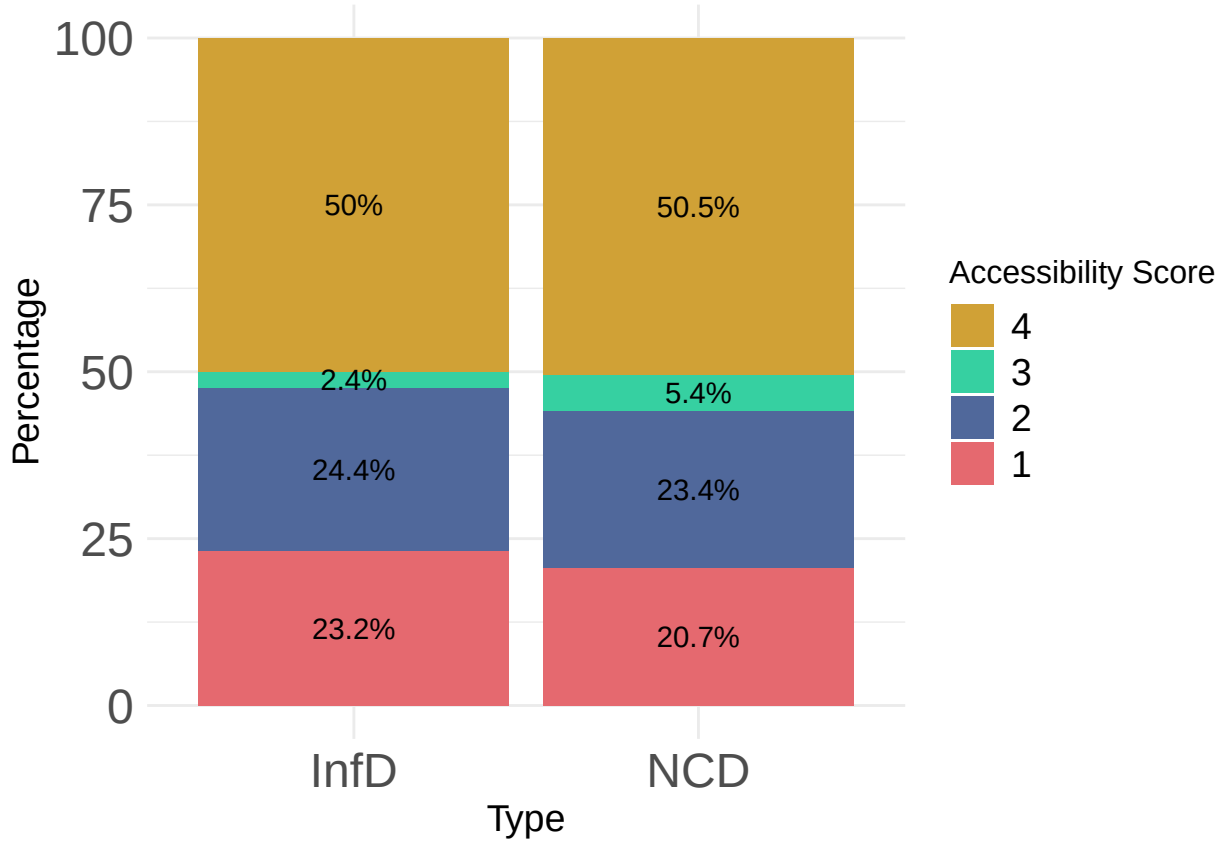
#bar chart for the Accessibility criteria by the Type of the study

```
summarized_data_Access <- data %>%
  group_by(Type, Access) %>%
  summarise(Count = n(), .groups = "drop") %>%
  left_join(data %>% group_by(Type) %>% summarise(Total = n(), .groups = "drop"), by = "Type") %>%
  mutate(Percentage = Count/Total * 100)

summarized_data_Access <- summarized_data_Access %>%
  mutate(Access = factor(Access, levels = c("4", "3", "2", "1")))

summarized_data_Access <- summarized_data_Access %>%
  arrange(Type, desc(Access)) %>%
  group_by(Type) %>%
  mutate(CumPercentage = cumsum(Percentage))
print(summarized_data_Access)
```

```
## # A tibble: 8 x 6
## # Groups:   Type [2]
##   Type  Access Count Total Percentage CumPercentage
##   <chr> <fct>  <int> <int>      <dbl>         <dbl>
## 1 InfD  1         19    82      23.2           23.2
## 2 InfD  2         20    82      24.4           47.6
```

```
## 3 InfD  3          2    82       2.44          50
## 4 InfD  4         41    82      50            100
## 5 NCD   1         23   111      20.7           20.7
## 6 NCD   2         26   111      23.4           44.1
## 7 NCD   3          6   111       5.41          49.5
## 8 NCD   4         56   111      50.5          100
```

```
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Visualization
gaccess <- ggplot(summarized_data_Access, aes(x = Type, y = Percentage, fill = as.factor(Access))) +
  geom_bar(stat = "identity", position = "stack") +
  scale_fill_manual(values = colors) +
  labs(x = "Type", y = "Percentage", fill = "Accessibility Score") +
  theme_minimal() +
  geom_text(aes(label = paste0(round(Percentage, 1), "%"),
                y = CumPercentage - (0.5 * Percentage)),
            color = "Black", size = 4) +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 18),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  ) +
  ylim(0, 100)

print(gaccess)
```




```
ggsave("gaccess.png", gaccess, width = 10, height = 8, dpi = 300)
```

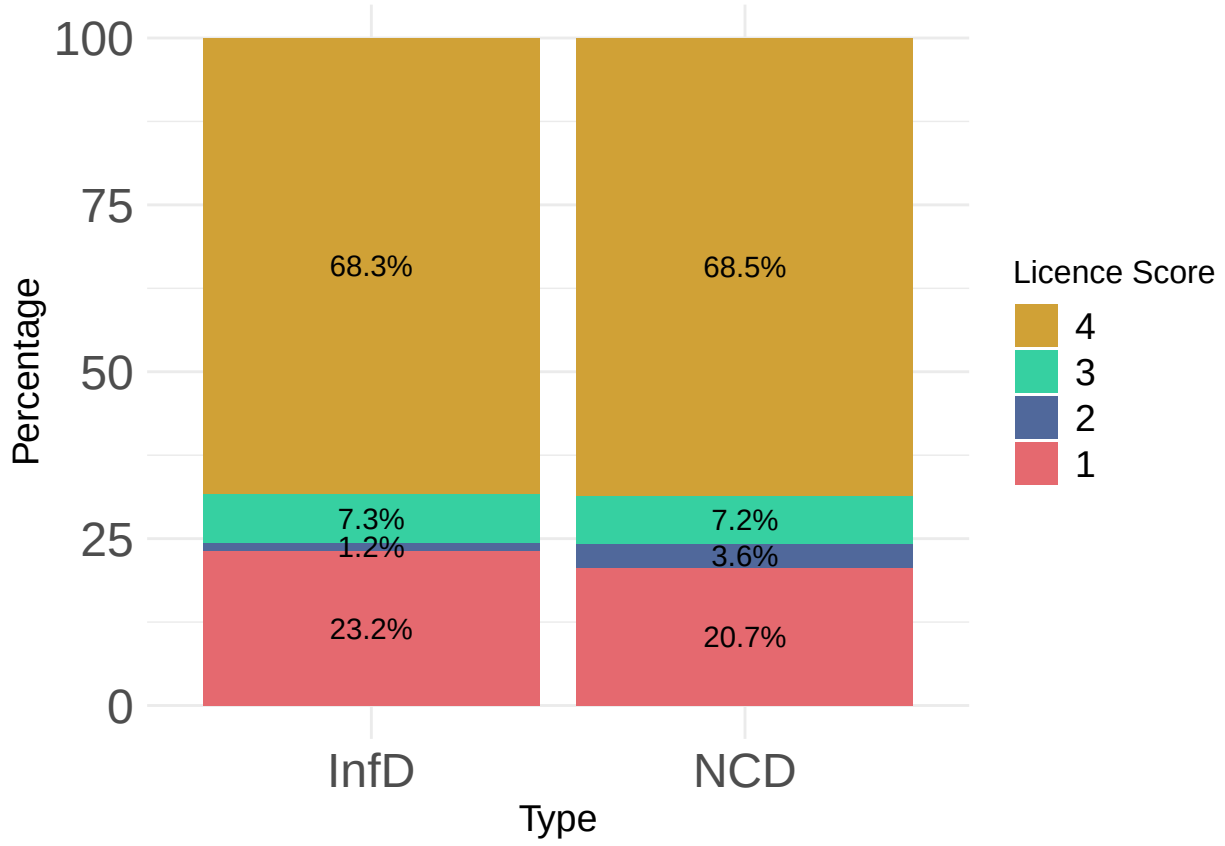
#bar chart for the Licence criteria by the Type of the study

```
summarized_data_Licence <- data %>%
  group_by(Type, Licence) %>%
  summarise(Count = n(), .groups = "drop") %>%
  left_join(data %>% group_by(Type) %>% summarise(Total = n(), .groups = "drop"), by = "Type") %>%
  mutate(Percentage = Count/Total * 100)

summarized_data_Licence <- summarized_data_Licence %>%
  mutate(Licence = factor(Licence, levels = c("4", "3", "2", "1")))

summarized_data_Licence <- summarized_data_Licence %>%
  arrange(Type, desc(Licence)) %>%
  group_by(Type) %>%
  mutate(CumPercentage = cumsum(Percentage))
print(summarized_data_Licence)
```

```
## # A tibble: 8 x 6
## # Groups:   Type [2]
##   Type  Licence Count Total Percentage CumPercentage
##   <chr> <fct>   <int> <int>      <dbl>         <dbl>
## 1 InfD  1          19    82      23.2           23.2
## 2 InfD  2           1    82       1.22          24.4
```

```
## 3 InfD  3           6    82       7.32          31.7
## 4 InfD  4          56    82      68.3           100
## 5 NCD   1          23   111      20.7           20.7
## 6 NCD   2           4   111       3.60          24.3
## 7 NCD   3           8   111       7.21          31.5
## 8 NCD   4          76   111      68.5           100
```

```
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Visualization
glicence <- ggplot(summarized_data_Licence, aes(x = Type, y = Percentage, fill = as.factor(Licence))) +
  geom_bar(stat = "identity", position = "stack") +
  scale_fill_manual(values = colors) +
  labs(x = "Type", y = "Percentage", fill = "Licence Score") +
  theme_minimal() +
  geom_text(aes(label = paste0(round(Percentage, 1), "%"),
                y = CumPercentage - (0.5 * Percentage)),
            color = "Black", size = 4) +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 18),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  ) +
  ylim(0, 100)

print(glicence)
```




```
ggsave("glicence.png", glicence, width = 10, height = 8, dpi = 300)
```

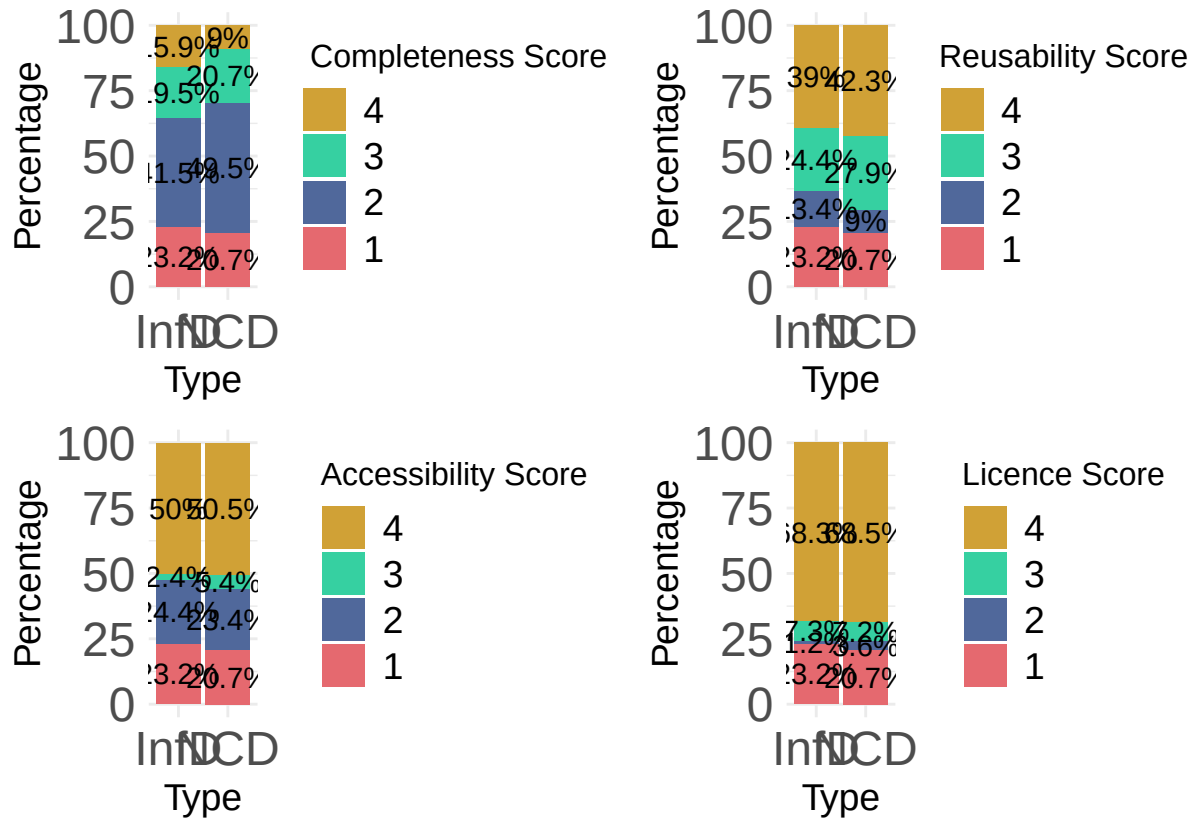
```
library(patchwork)
```

```
## Warning: package 'patchwork' was built under R version 4.2.2
```

```
##
## Attaching package: 'patchwork'
```

```
## The following object is masked from 'package:MASS':
##
##     area
```

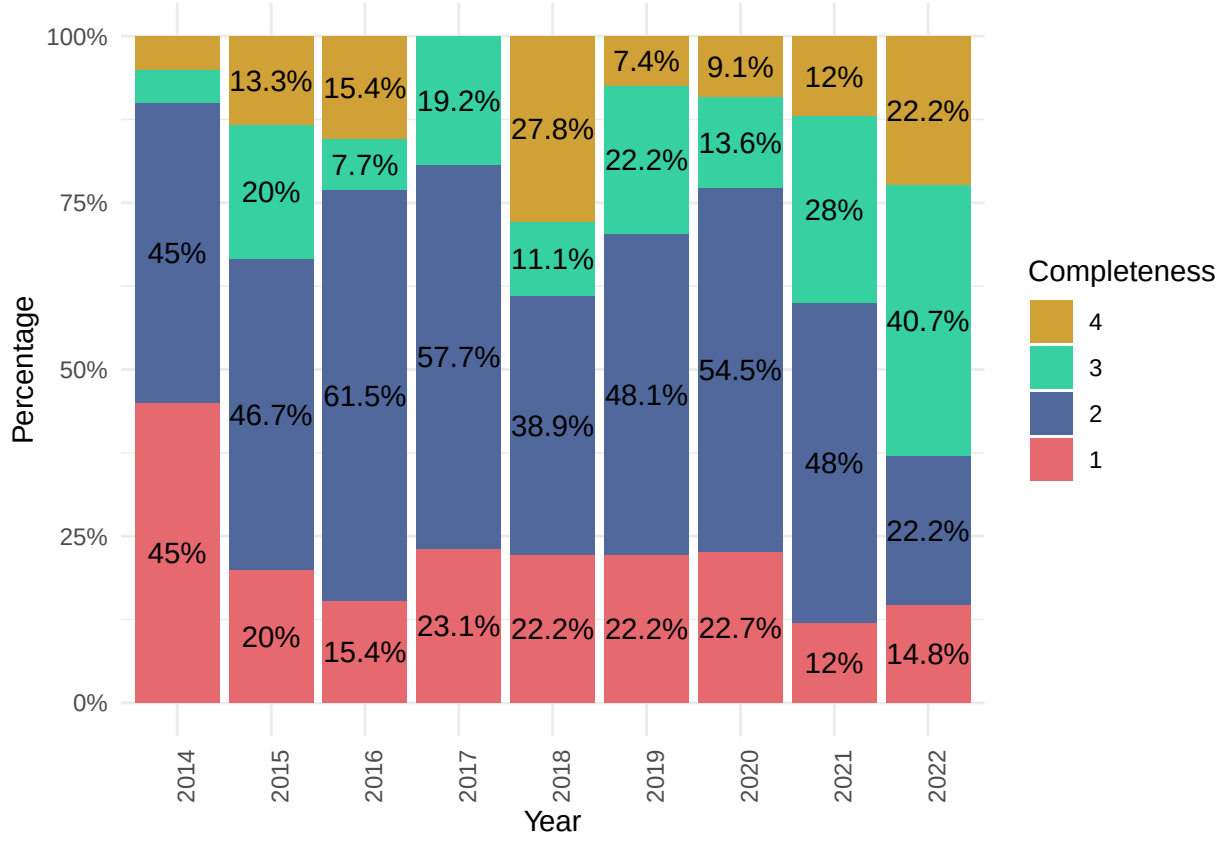
```
combinedplot <- gcomp + greuse + gaccess + glicence +
  plot_layout(
    ncol = 2, heights = c(10, 10), widths = c(10, 10)
  )
print(combinedplot)
```




```
ggsave("combinedplot.png", combinedplot, width = 15, height = 10, units = "in")
```

```
# Calculate the frequency and percentage for each 'Complete' score within each 'Year'
long_data <- data %>%
  count(Year, Complete) %>%
  group_by(Year) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Complete = factor(Complete, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
Compyear <- ggplot(long_data, aes(x = as.factor(Year), y = n, fill = as.factor(Complete))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
```

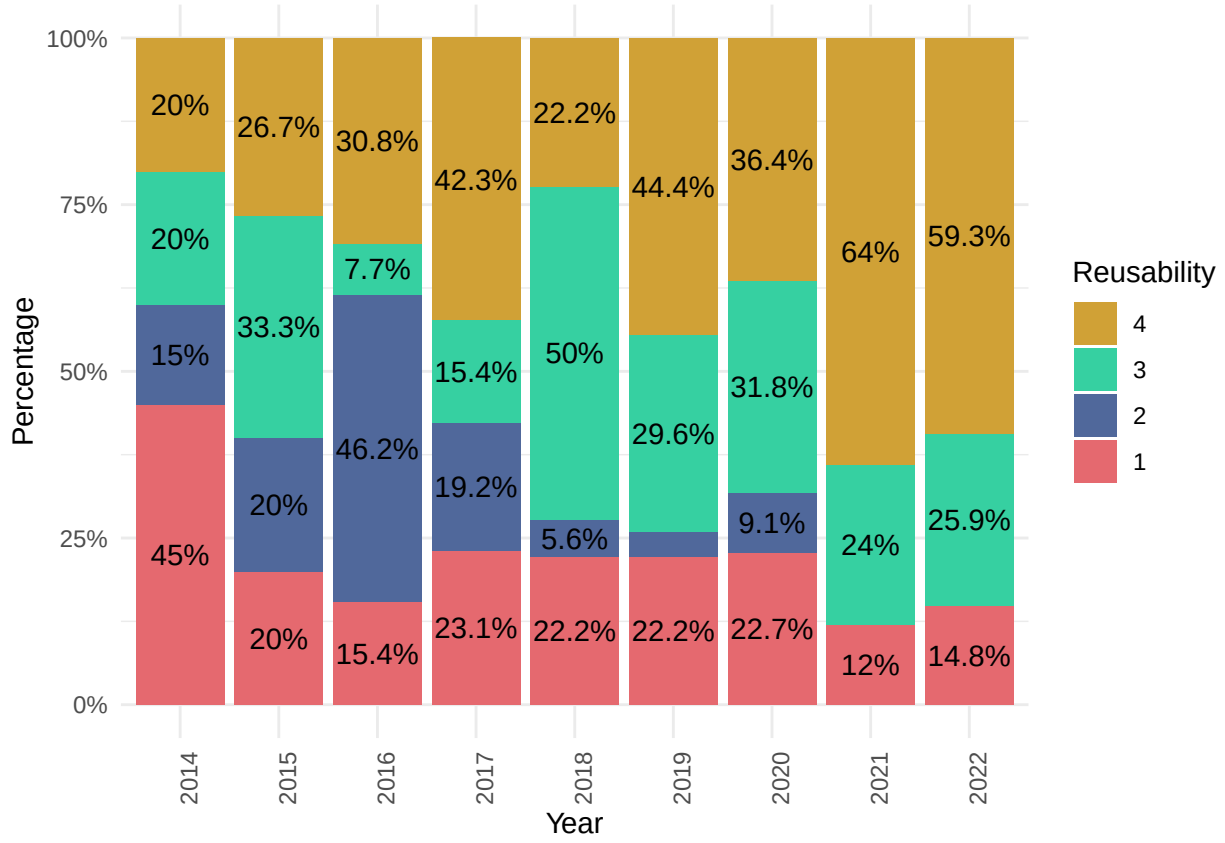
```
  ) +
  scale_fill_manual(values = colors) +
  labs(x = "Year",
       y = "Percentage",
       fill = "Completeness") +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 20),  # Increase axis text
    legend.title = element_text(size = 18), # Increase legend title
    legend.text = element_text(size = 20)  # Increase legend text
  ) +
  theme_minimal() +
  scale_x_discrete(name = "Year", labels = unique(long_data$Year)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(Compyear)
```




```
ggsave("Compyear.png", Compyear, width = 15, height = 10, units = "in", bg = "white")
```

```
# Calculate the frequency and percentage for each 'Reuse' score within each 'Year'
long_data <- data %>%
  count(Year, Reuse) %>%
  group_by(Year) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
```

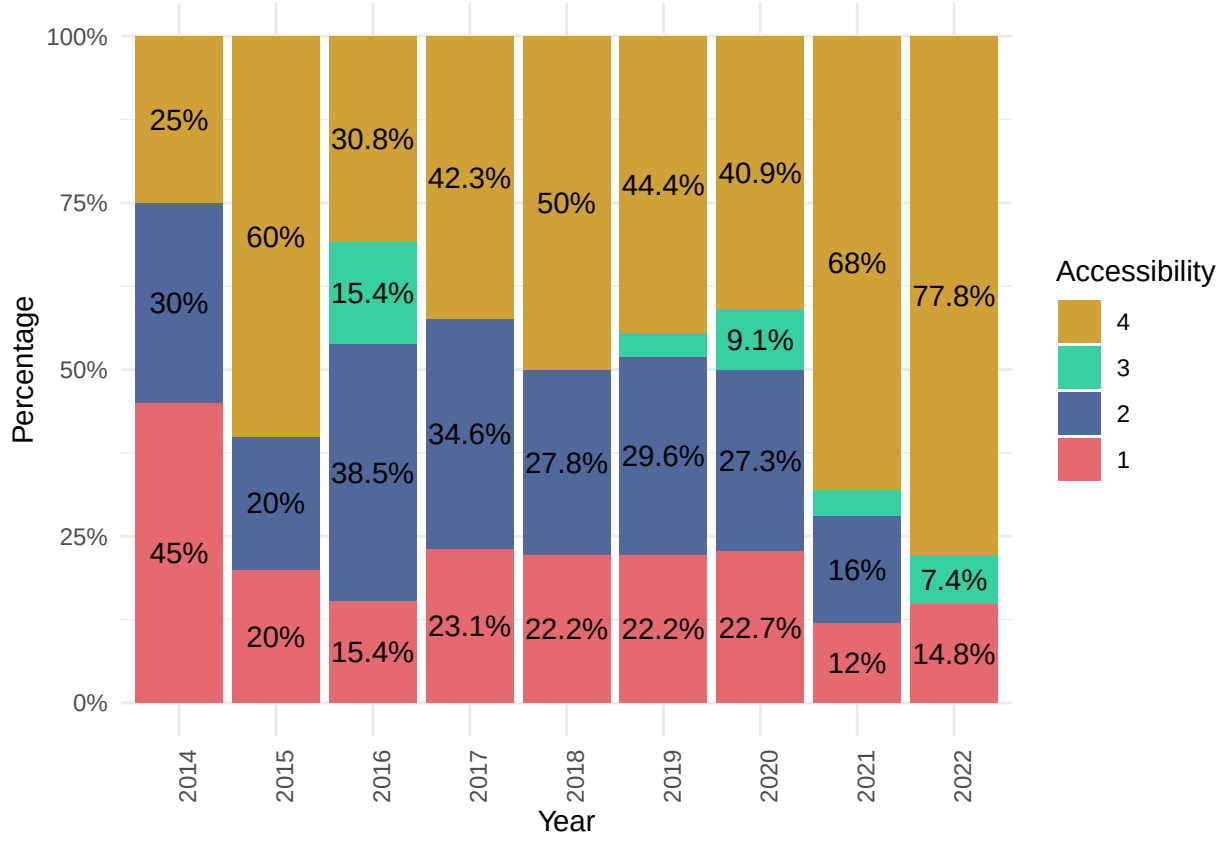
```r
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Reuse = factor(Reuse, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
reuseyear <- ggplot(long_data, aes(x = as.factor(Year), y = n, fill = as.factor(Reuse))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(x = "Year",
       y = "Percentage",
       fill = "Reusability") +
    theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 20),  # Increase axis text
    legend.title = element_text(size = 18), # Increase legend title
    legend.text = element_text(size = 20)  # Increase legend text
  ) +
  theme_minimal() +
  scale_x_discrete(name = "Year", labels = unique(long_data$Year)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(reuseyear)
```




```
ggsave("reuseyear.png", reuseyear, width = 15, height = 10, units = "in", bg = "white")
```

```
# Calculate the frequency and percentage for each 'Access' score within each 'Year'
long_data <- data %>%
  count(Year, Access) %>%
  group_by(Year) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Access = factor(Access, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
accessyear <- ggplot(long_data, aes(x = as.factor(Year), y = n, fill = as.factor(Access))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
```

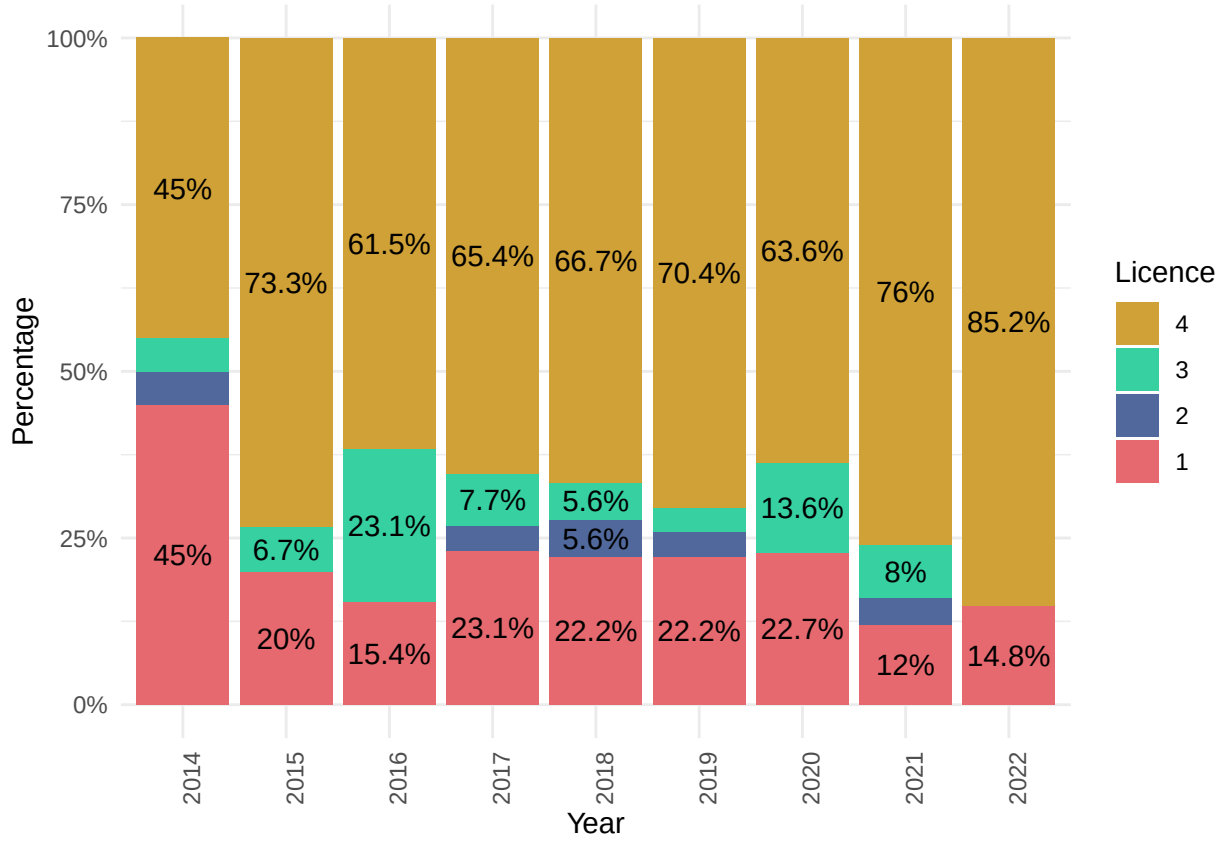
```
  ) +
  scale_fill_manual(values = colors) +
  labs(x = "Year",
       y = "Percentage",
       fill = "Accessibility") +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 20),  # Increase axis text
    legend.title = element_text(size = 18), # Increase legend title
    legend.text = element_text(size = 20)  # Increase legend text
  ) +
    theme_minimal() +
  scale_x_discrete(name = "Year", labels = unique(long_data$Year)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(accessyear)
```




```
ggsave("accessyear.png", accessyear, width = 15, height = 10, units = "in", bg = "white")
```

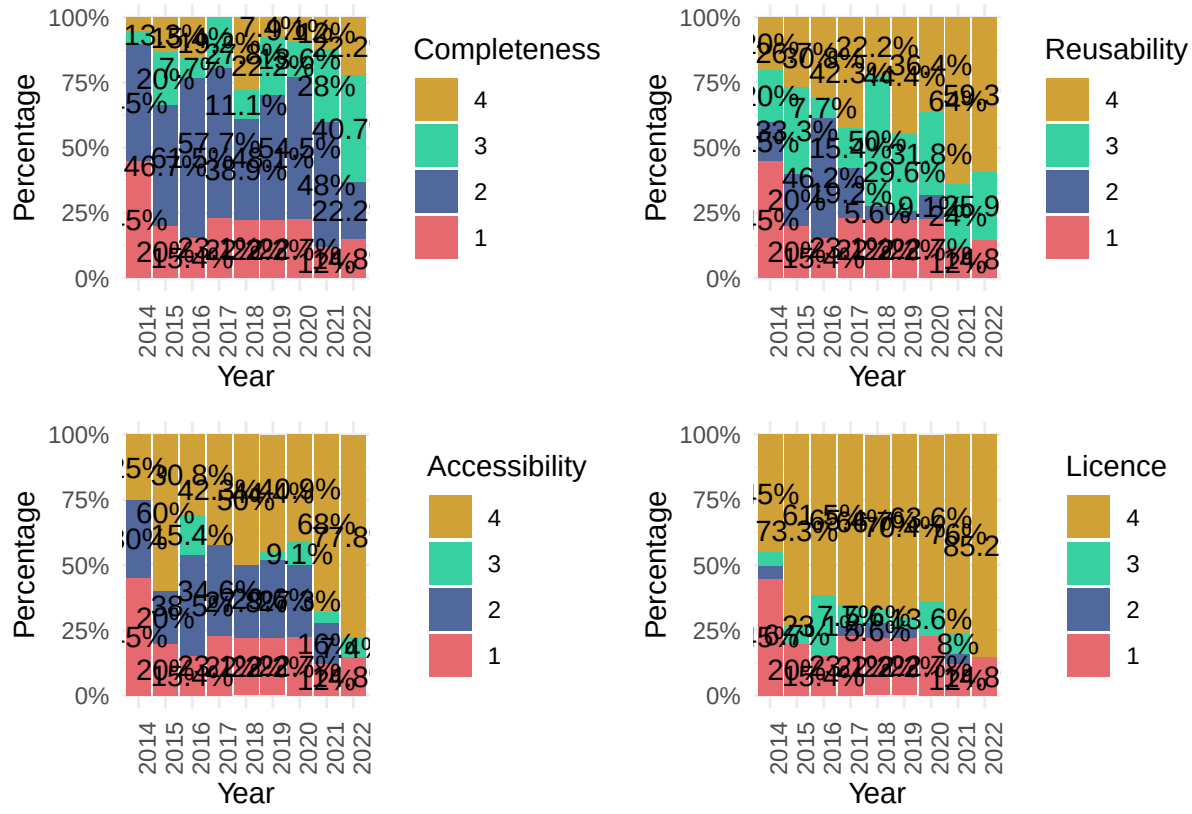
```
# Calculate the frequency and percentage for each 'Licence' score within each 'Year'
long_data <- data %>%
  count(Year, Licence) %>%
  group_by(Year) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
```

```
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Licence = factor(Licence, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
licenceyear <- ggplot(long_data, aes(x = as.factor(Year), y = n, fill = as.factor(Licence))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(x = "Year",
       y = "Percentage",
       fill = "Licence") +
  theme(
    axis.title = element_text(size = 14), # Increase axis titles
    axis.text = element_text(size = 20),  # Increase axis text
    legend.title = element_text(size = 18), # Increase legend title
    legend.text = element_text(size = 20)  # Increase legend text
  ) +
  theme_minimal() +
  scale_x_discrete(name = "Year", labels = unique(long_data$Year)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(licenceyear)
```




```
ggsave("licenceyear.png", licenceyear, width = 15, height = 10, units = "in", bg = "white")
```

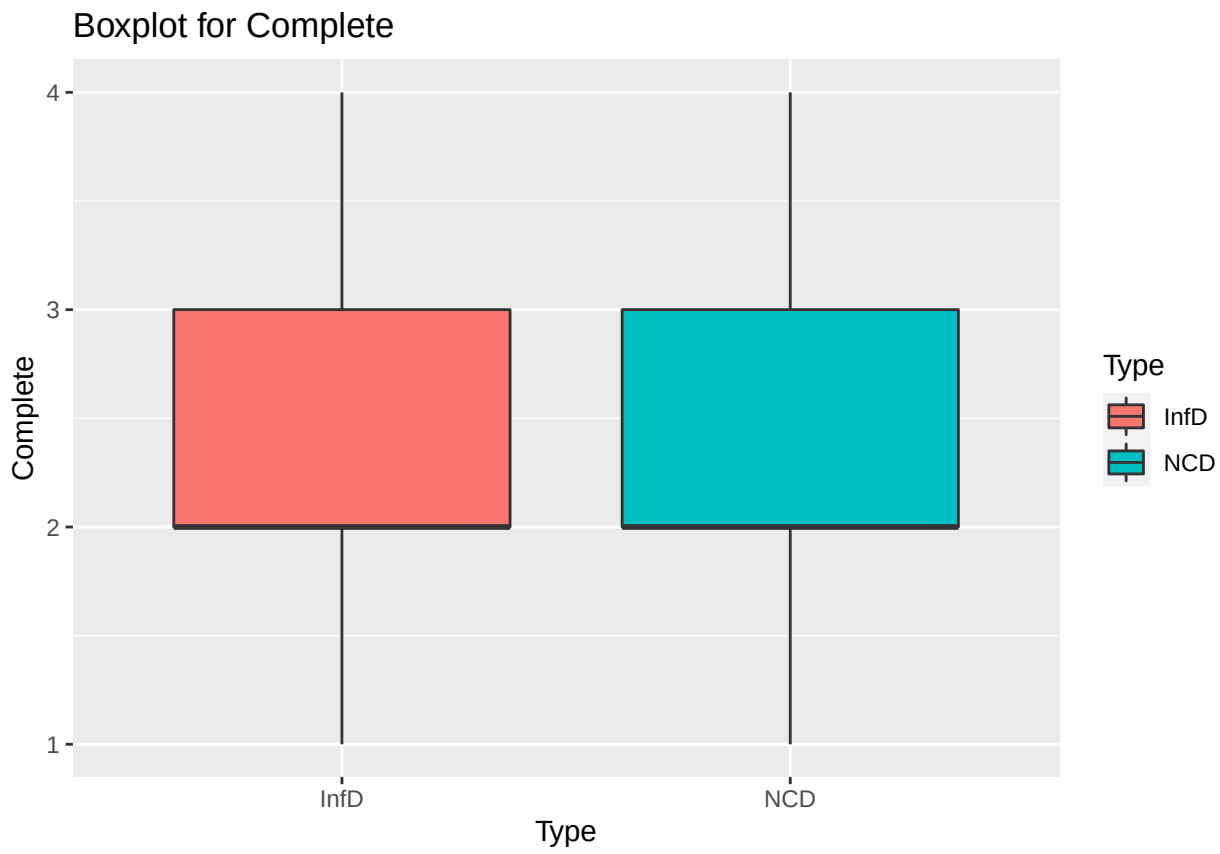
```
plotyear <- Compyear + reuseyear + accessyear + licenceyear +
  plot_layout(
    ncol = 2, heights = c(10, 10), widths = c(10, 10)
  )
print(plotyear)
```



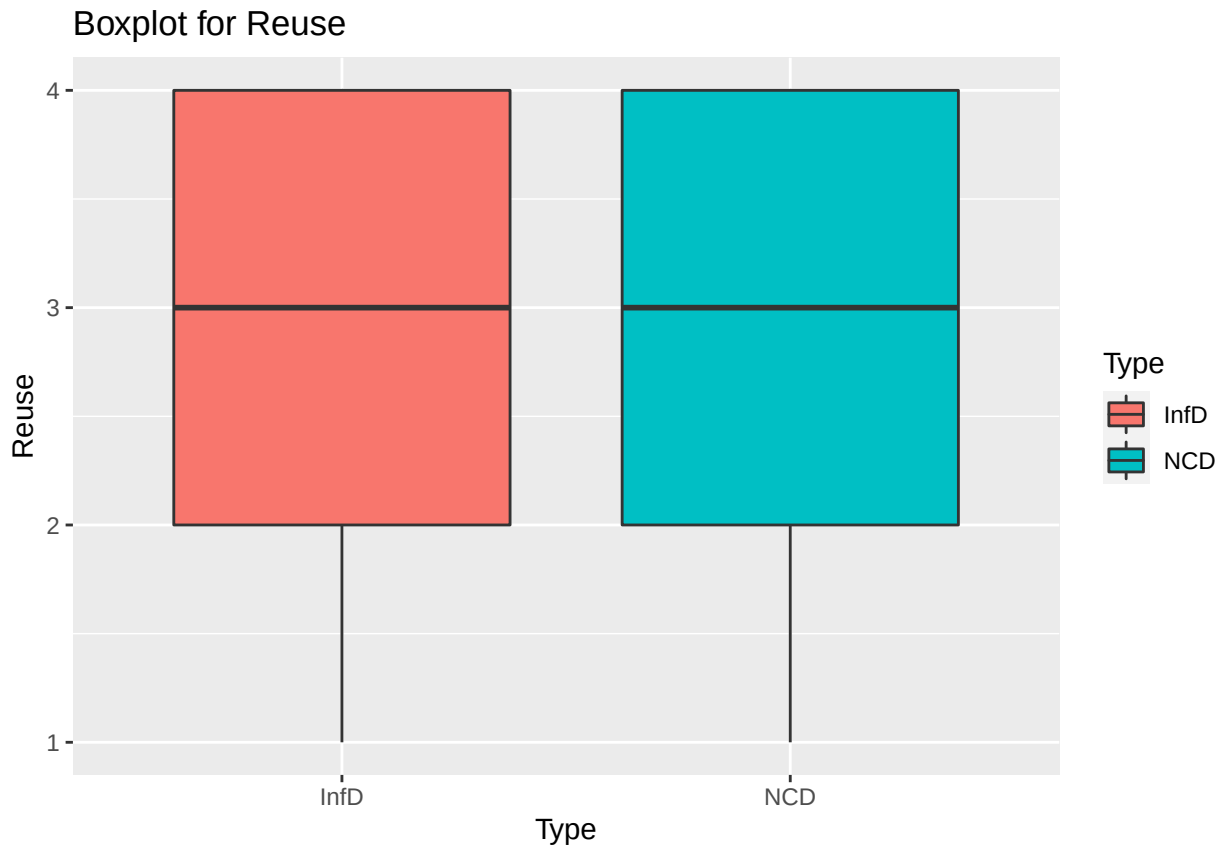

```
ggsave("plotyear.png", plotyear, width = 15, height = 10, units = "in")
```

#study the significant dfference between two type of the reseearch for each scoring criteria

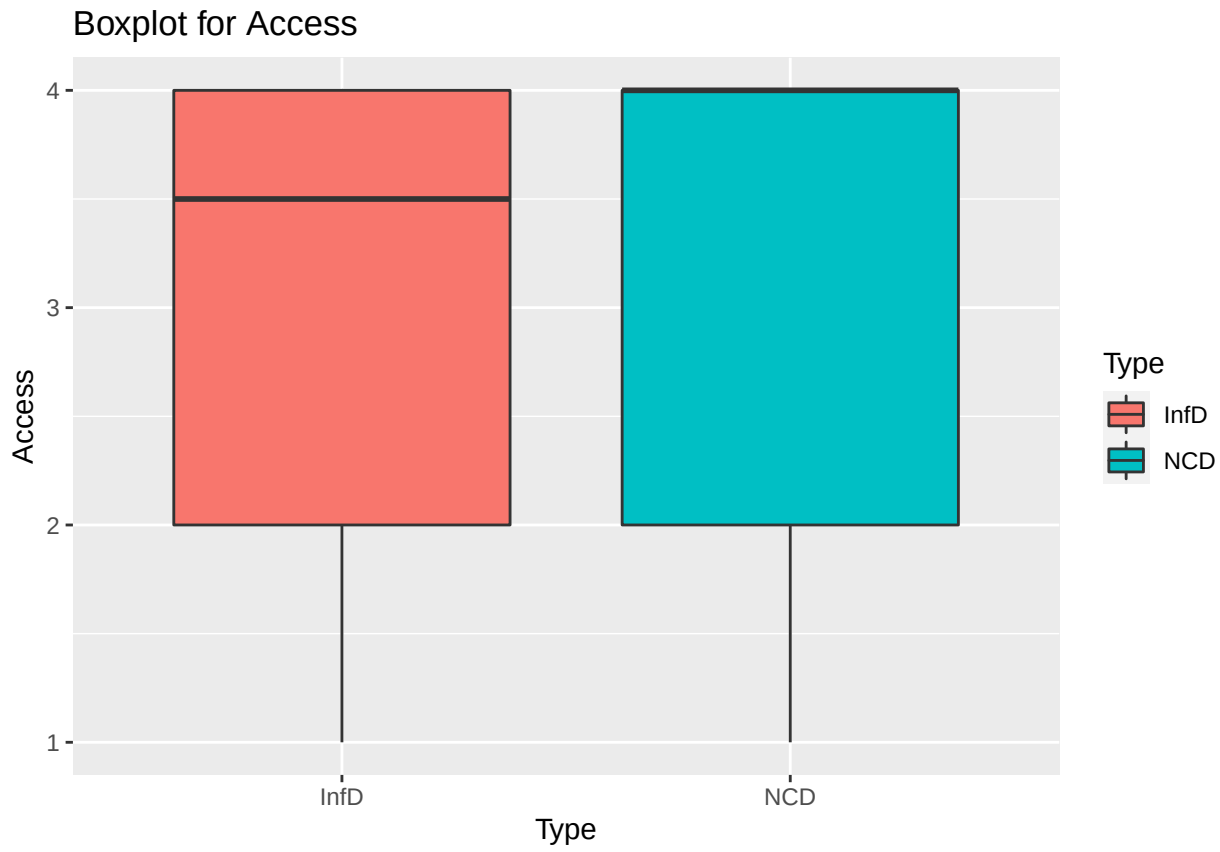
```
# Convert scoring criteria to numeric (if not already)
data <- data %>%
  mutate(across(c(Complete, Reuse, Access, Licence), ~as.numeric(as.character(.))))

# Assumption checks for each criterion
for (criterion in c("Complete", "Reuse", "Access", "Licence")) {
  # Create and display a box plot to check the distribution shape and spread
  print(ggplot(data, aes_string(x = "Type", y = criterion, fill = "Type")) +
    geom_boxplot() +
    ggtitle(paste("Boxplot for", criterion)))

  # Print the median for a basic comparison
  cat(paste("Median of", criterion, "by Type:\n"))
  print(aggregate(. ~ Type, data[c("Type", criterion)], median))

  cat("\n")
}
```



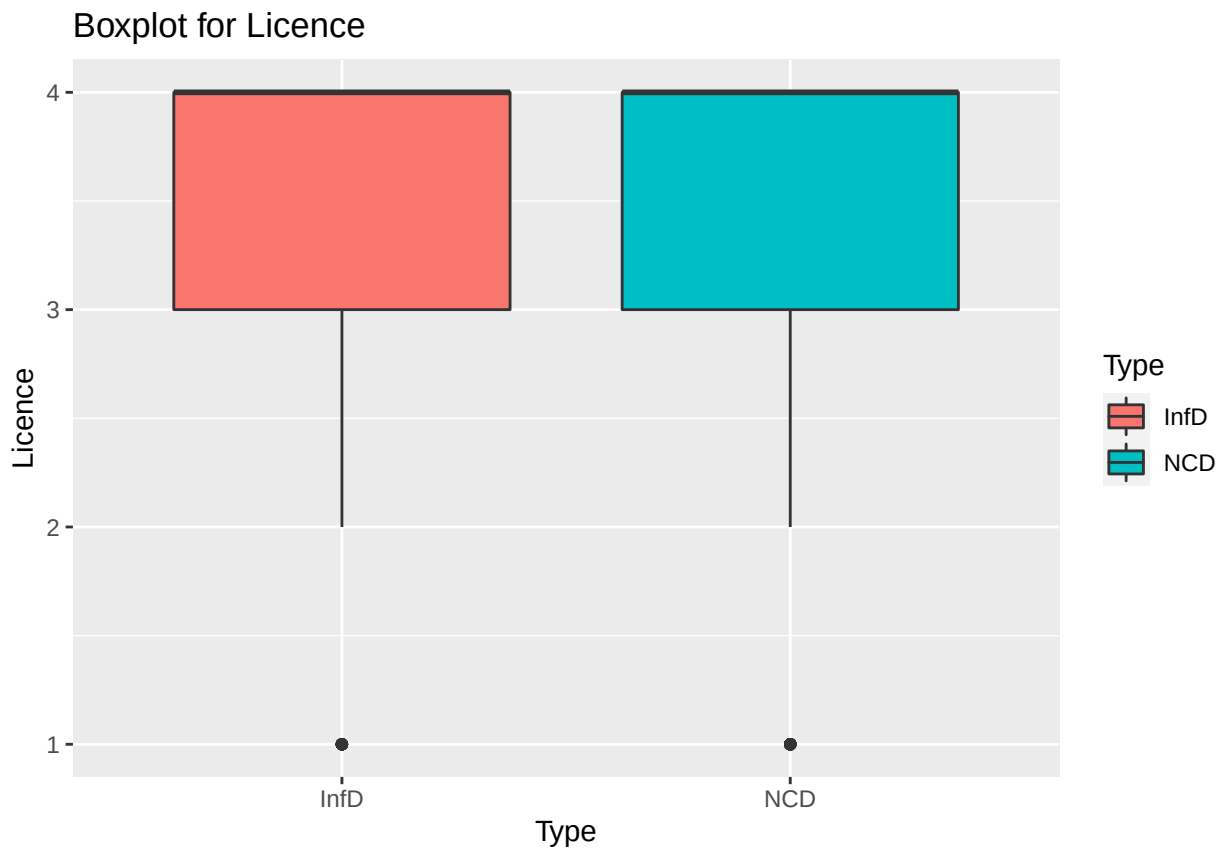

```
## Median of Complete by Type:
##   Type Complete
## 1 InfD        2
## 2  NCD        2
```




```
## Median of Reuse by Type:
##   Type Reuse
## 1 InfD     3
## 2  NCD     3
```




```
## Median of Access by Type:
##   Type Access
## 1 InfD    3.5
## 2  NCD    4.0
```




```
## Median of Licence by Type:
##   Type Licence
## 1 InfD       4
## 2  NCD       4
```

```r
# Function to perform Mann-Whitney U test
perform_mann_whitney <- function(data, criterion) {
  test_result <- wilcox.test(
    reformulate("Type", response = criterion),
    data = data,
    exact = FALSE
  )
  list(criterion = criterion, p.value = test_result$p.value)
}

# Perform the test for each criterion
mw_results <- lapply(c("Complete", "Reuse", "Access", "Licence"), function(criterion) {
  perform_mann_whitney(data, criterion)
})

# Convert the list of results to a dataframe
mw_results_df <- do.call(rbind, mw_results)

print(mw_results_df)
```

```
##      criterion  p.value
```

```
## [1,] "Complete" 0.5901688
## [2,] "Reuse"    0.4855373
## [3,] "Access"   0.7664346
## [4,] "Licence"  0.9162592
```

```
#The assumptions of the test were met:
### Ordinal Data Check: The scoring criteria are ordinal variables
### Similar Distribution Shapes: For each criterion, it creates a box plot to visually inspect the dist
### Independence of Observations:the variables are independence observations.
```

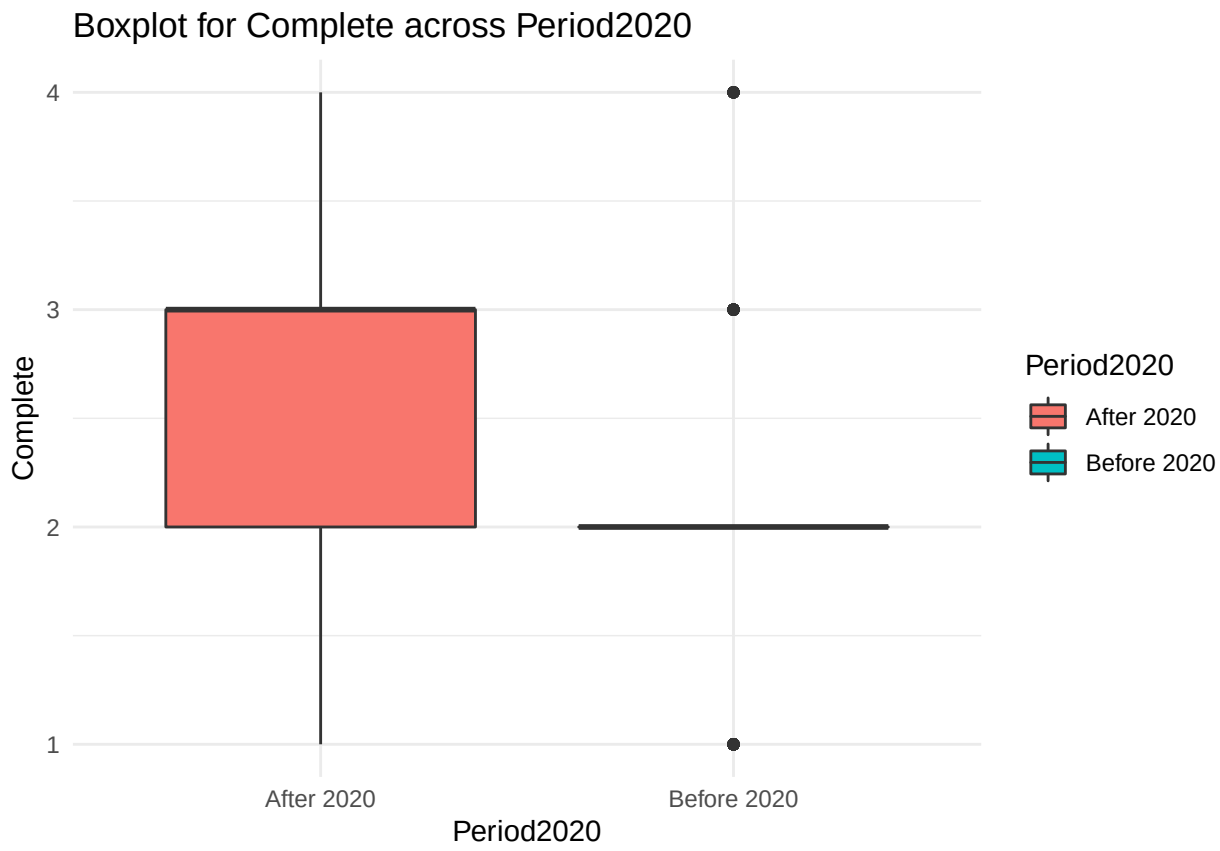
#Creat variables for FAIR principles in 2016 and COVID-19 in 2020

```
data <- data %>%
  mutate(Period2020 = ifelse(Year <= 2020, "Before 2020", "After 2020")) %>%
  mutate(Period2016 = ifelse(Year <= 2016, "Before 2016", "After 2016"))#
```

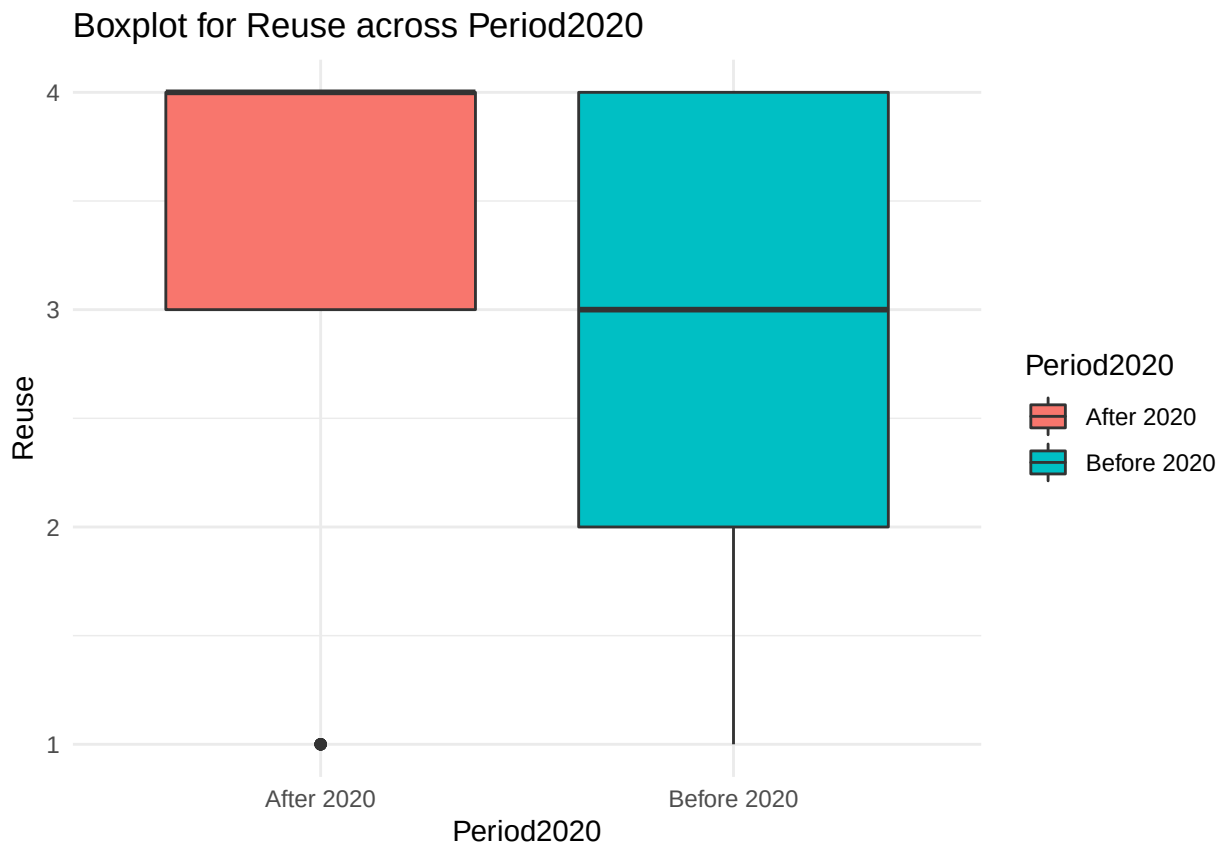
#Study the significant difference before and afterCovid-19 2020 using Median and Mann Whitney U test

```
# Ensure the scoring criteria are numeric
data <- data %>%
  mutate(across(c(Complete, Reuse, Access, Licence), ~as.numeric(as.character(.))))

# Function to generate box plots for distribution checks
generate_boxplot <- function(data, score_var, period_var) {
  ggplot(data, aes_string(x = period_var, y = score_var, fill = period_var)) +
    geom_boxplot() +
    labs(title = paste("Boxplot for", score_var, "across", period_var),
         x = period_var,
         y = score_var) +
    theme_minimal()
}

# Generate and display box plots for each score with Period2020
lapply(c("Complete", "Reuse", "Access", "Licence"), function(score_var) {
  generate_boxplot(data, score_var, "Period2020")
})
```

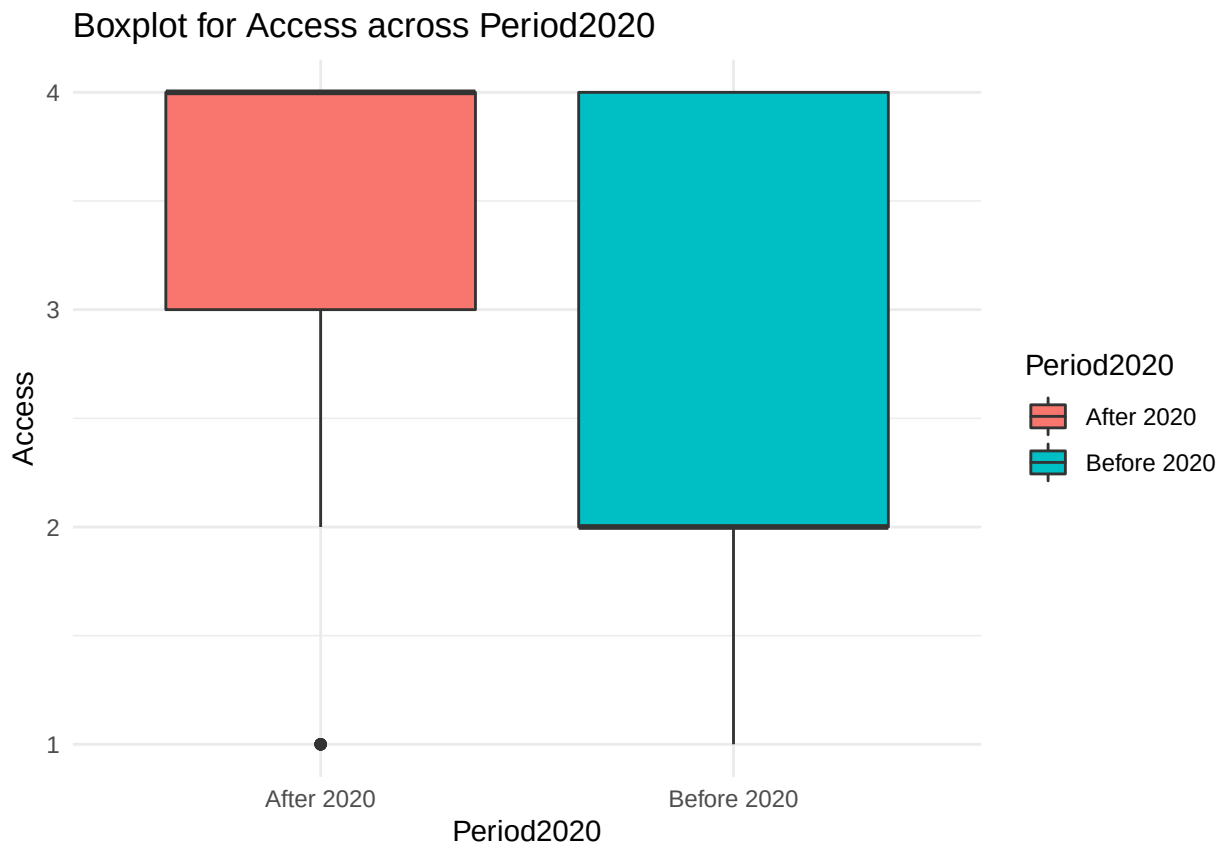
```
## [[1]]
```


Boxplot for Complete across Period2020
Complete
4
3
2
1
After 2020
Before 2020
Period2020
Period2020
After 2020
Before 2020

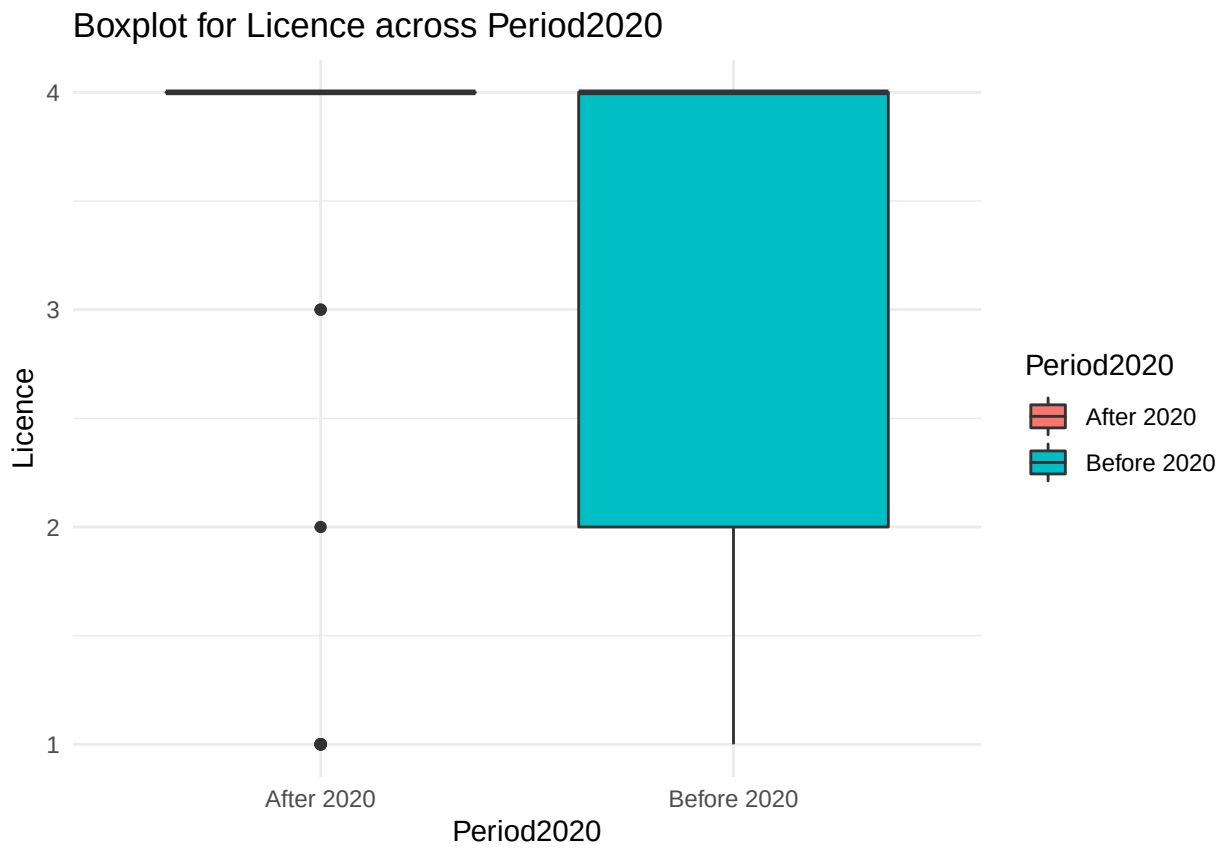
```
##
## [[2]]
```


Boxplot for Reuse across Period2020
Reuse
4
3
2
1
After 2020
Before 2020
Period2020
Period2020
After 2020
Before 2020

```
##
## [[3]]
```


Boxplot for Access across Period2020
Access
4
3
2
1
After 2020
Before 2020
Period2020
Period2020
After 2020
Before 2020

```
##
## [[4]]
```




```
# Function to perform the Mann-Whitney test and calculate medians
perform_analysis <- function(data, score_var, period_var) {
  test_result <- wilcox.test(reformulate(period_var, score_var), data = data, exact = FALSE)

  medians <- data %>%
    group_by(!!sym(period_var)) %>%
    summarise(median = median(!!sym(score_var), na.rm = TRUE), .groups = 'drop')

  list(median = medians, p.value = test_result$p.value)
}

# Analysis for each score for 2020
results_complete_2020 <- perform_analysis(data, "Complete", "Period2020")
results_reuse_2020 <- perform_analysis(data, "Reuse", "Period2020")
results_access_2020 <- perform_analysis(data, "Access", "Period2020")
results_licence_2020 <- perform_analysis(data, "Licence", "Period2020")

# Print results
print(results_complete_2020)
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       3
## 2 Before 2020      2
```

```
##
## $p.value
## [1] 0.00147346
```

```
print(results_reuse_2020)
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      3
##
## $p.value
## [1] 0.0002688369
```

```
print(results_access_2020)
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      2
##
## $p.value
## [1] 0.0002703917
```

```
print(results_licence_2020)
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      4
##
## $p.value
## [1] 0.02901313
```

```
#The assumptions of the test were met:
### Ordinal Data Check: The scoring criteria are ordinal variables
### Similar Distribution Shapes: For each criterion, it creates a box plot to visually inspect the distri
### Independence of Observations:the variables are independence observations.
```

#Study the significant difference before and afterCovid-19 2020 using Median and Mann Whitney U test for the NCD Papers

```
data_NCD <- data %>%
  filter(Type == "NCD")
```

```r
perform_analysis <- function(data_NCD, score_var, period_var) {
  test_result <- wilcox.test(reformulate(period_var, score_var), data = data_NCD, exact = FALSE)

  medians <- data_NCD %>%
    group_by(!!sym(period_var)) %>%
    summarise(median = median(!!sym(score_var), na.rm = TRUE), .groups = 'drop')

  list(median = medians, p.value = test_result$p.value)
}

# Analysis for each score for 2020
results_complete_2020_NCD <- perform_analysis(data_NCD, "Complete", "Period2020")
results_reuse_2020_NCD <- perform_analysis(data_NCD, "Reuse", "Period2020")
results_access_2020_NCD <- perform_analysis(data_NCD, "Access", "Period2020")
results_licence_2020_NCD <- perform_analysis(data_NCD, "Licence", "Period2020")

# Print results
results_complete_2020_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       2
## 2 Before 2020      2
##
## $p.value
## [1] 0.004012932
```

```r
results_reuse_2020_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      3
##
## $p.value
## [1] 0.001191915
```

```r
results_access_2020_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      2
##
## $p.value
## [1] 0.0007444155
```

```
results_licence_2020_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      4
##
## $p.value
## [1] 0.0153904
```

#Study the significant difference before and afterCovid-19 2020 using Median and Mann Whitney U test for the InfD Papers

```
data_InfD <- data %>%
  filter(Type == "InfD")


perform_analysis <- function(data_NCD, score_var, period_var) {
  test_result <- wilcox.test(reformulate(period_var, score_var), data = data_InfD, exact = FALSE)

  medians <- data_InfD %>%
    group_by(!!sym(period_var)) %>%
    summarise(median = median(!!sym(score_var), na.rm = TRUE), .groups = 'drop')

  list(median = medians, p.value = test_result$p.value)
}

# Analysis for each score for 2020
results_complete_2020_InfD <- perform_analysis(data_InfD, "Complete", "Period2020")
results_reuse_2020_InfD <- perform_analysis(data_InfD, "Reuse", "Period2020")
results_access_2020_InfD <- perform_analysis(data_InfD, "Access", "Period2020")
results_licence_2020_InfD <- perform_analysis(data_InfD, "Licence", "Period2020")

# Print results
results_complete_2020_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       3
## 2 Before 2020      2
##
## $p.value
## [1] 0.07239354
```

```
results_reuse_2020_InfD
```

```
## $median
## # A tibble: 2 x 2
```

```
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      3
##
## $p.value
## [1] 0.1105719
```

```
results_access_2020_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      2
##
## $p.value
## [1] 0.1227605
```

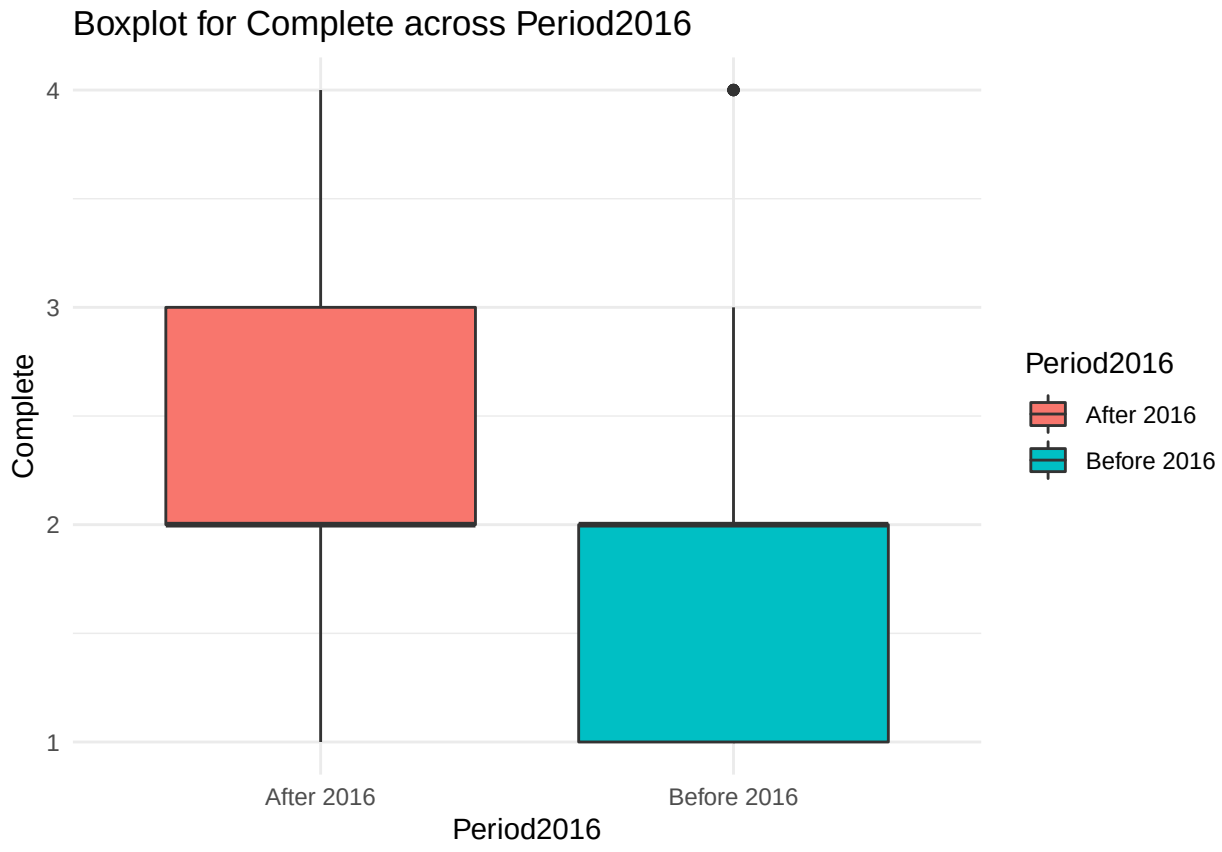
```
results_licence_2020_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2020  median
##   <chr>        <dbl>
## 1 After 2020       4
## 2 Before 2020      4
##
## $p.value
## [1] 0.6975206
```

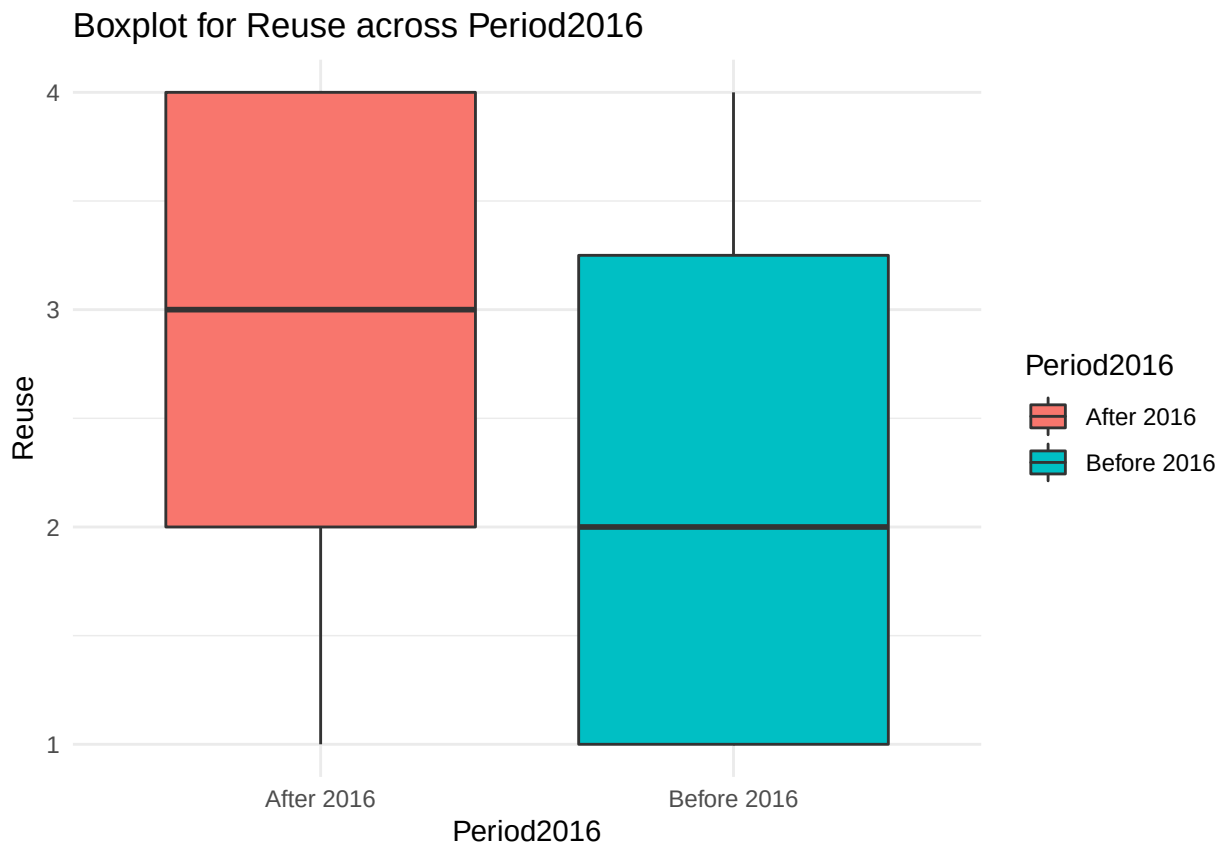
#Study the significant difference before and after Fair 2016 using Median and Mann Whitney U for all the papers

```
data <- data %>%
  mutate(Period2016 = ifelse(Year <= 2016, "Before 2016", "After 2016"))

# Function to perform the Mann-Whitney test, calculate medians, and generate box plots
perform_analysis <- function(data, score_var, period_var) {
  test_result <- wilcox.test(reformulate(period_var, score_var), data = data, exact = FALSE)

  medians <- data %>%
    group_by(!!sym(period_var)) %>%
    summarise(median = median(!!sym(score_var), na.rm = TRUE), .groups = 'drop')

  # Generate a box plot
  box_plot <- ggplot(data, aes_string(x = period_var, y = score_var, fill = period_var)) +
    geom_boxplot() +
    labs(title = paste("Boxplot for", score_var, "across", period_var),
         x = period_var,
         y = score_var) +
```

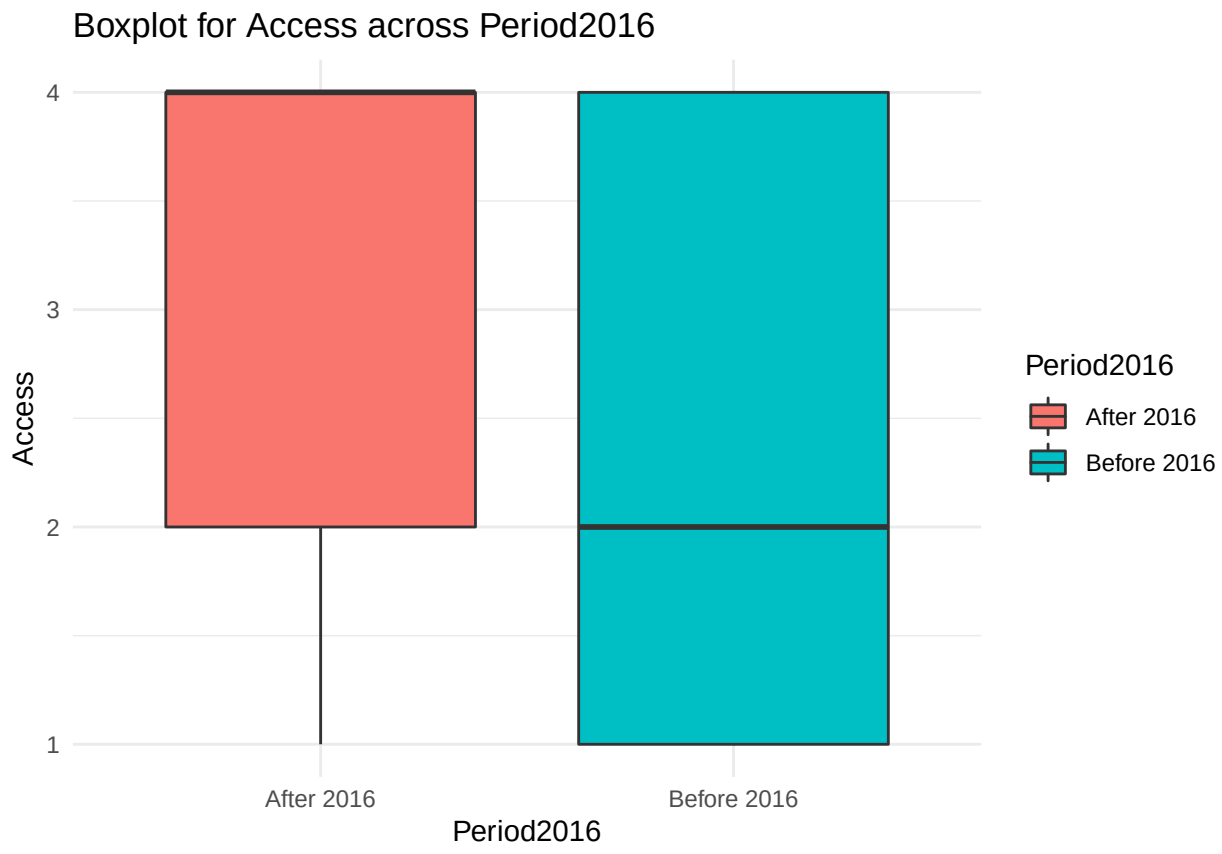
```
    theme_minimal()

  # Print the box plot
  print(box_plot)

  list(median = medians, p.value = test_result$p.value, plot = box_plot)
}

# Analysis for each score for 2016
results_complete_2016 <- perform_analysis(data, "Complete", "Period2016")
```



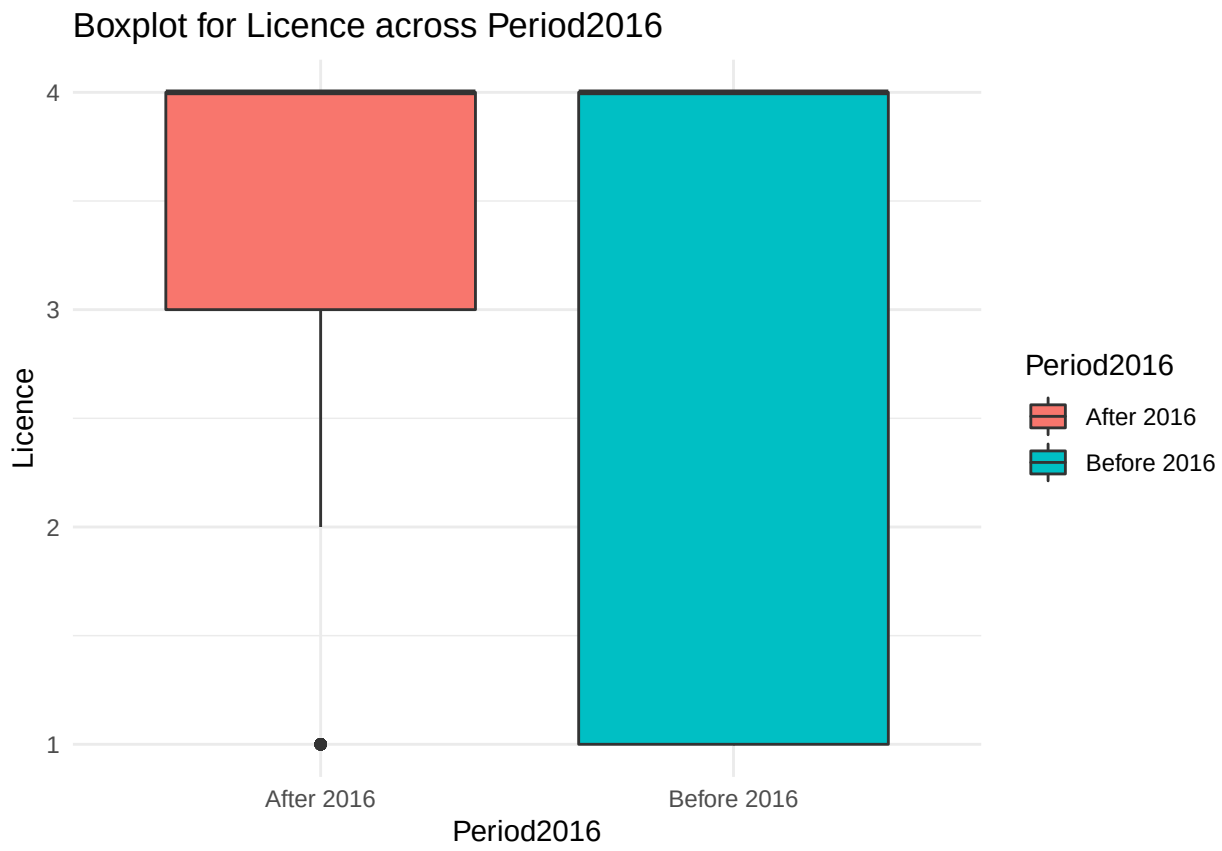

```
results_reuse_2016 <- perform_analysis(data, "Reuse", "Period2016")
```




```
results_access_2016 <- perform_analysis(data, "Access", "Period2016")
```

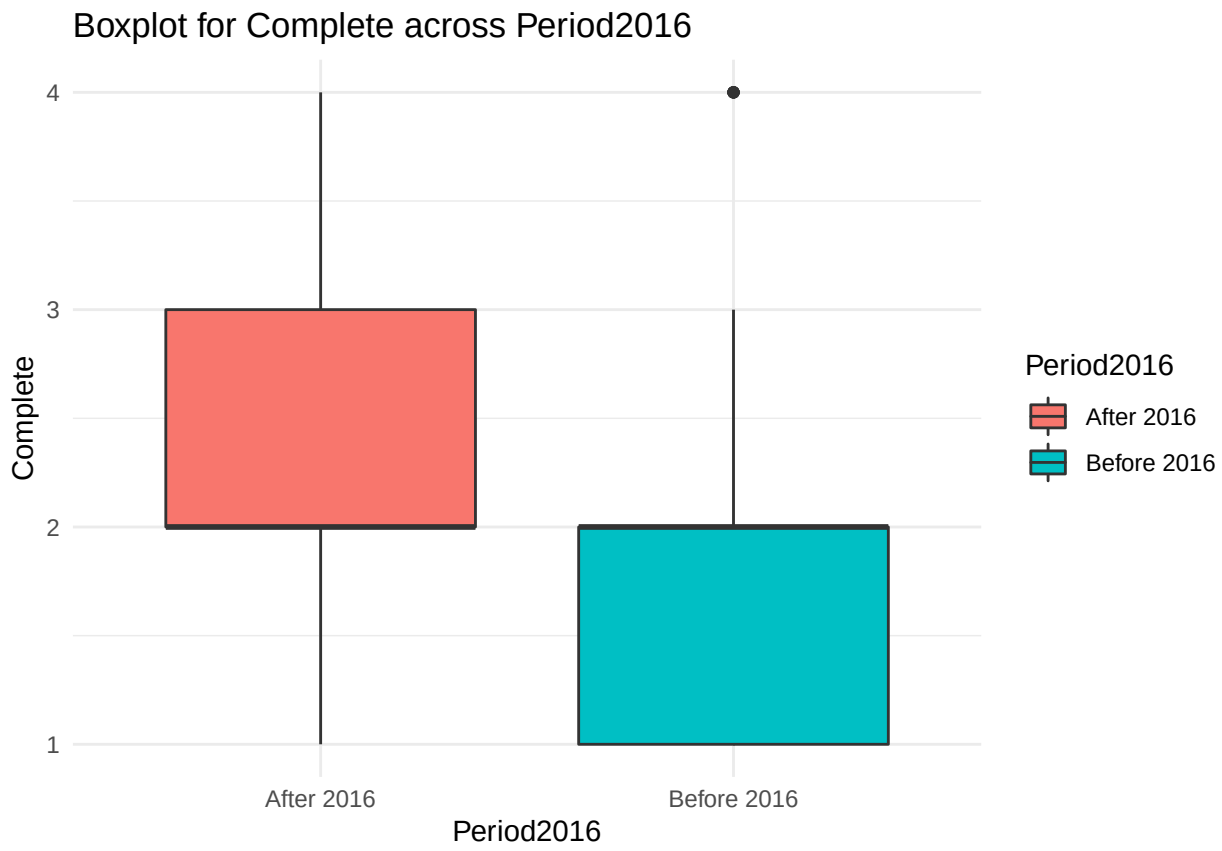



```
results_licence_2016 <- perform_analysis(data, "Licence", "Period2016")
```



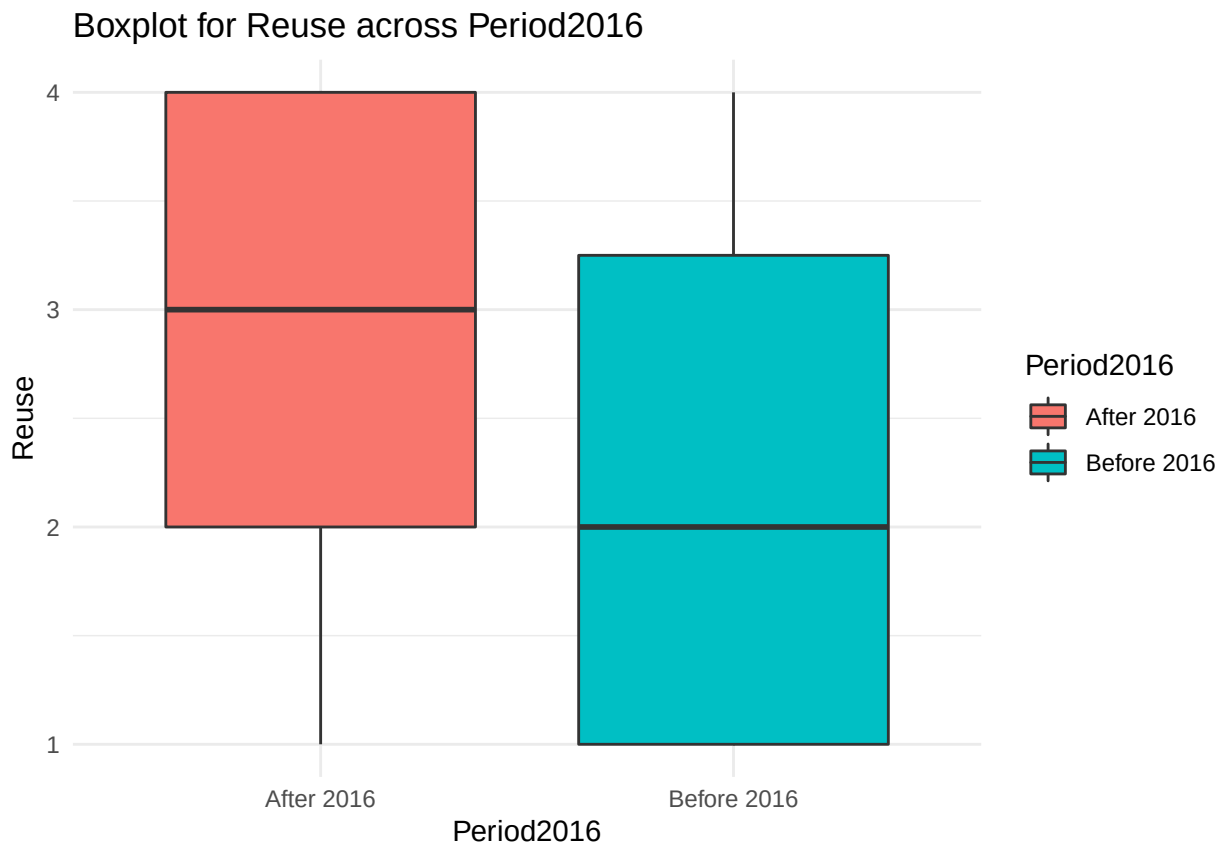

```
# Print results
results_complete_2016
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       2
## 2 Before 2016      2
##
## $p.value
## [1] 0.05617637
##
## $plot
```



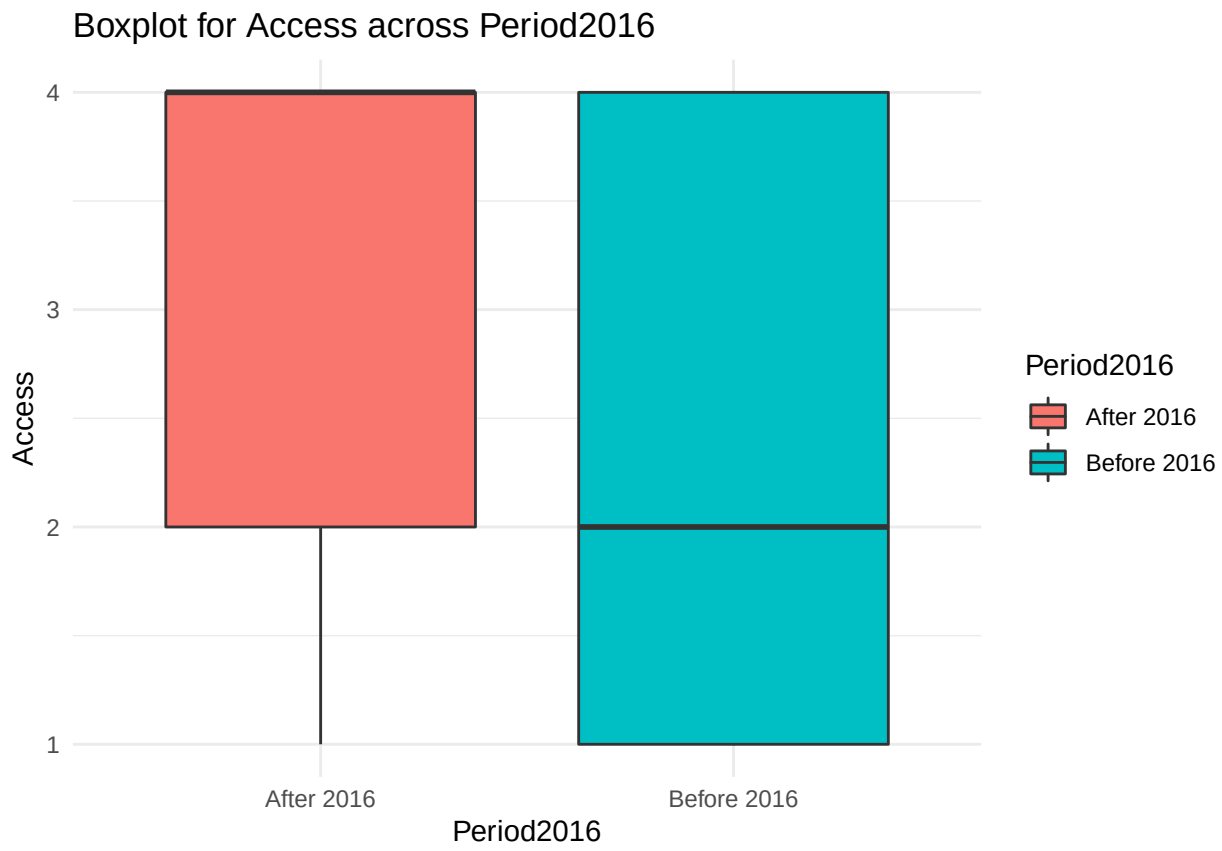

```
results_reuse_2016
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       3
## 2 Before 2016      2
##
## $p.value
## [1] 0.002078694
##
## $plot
```



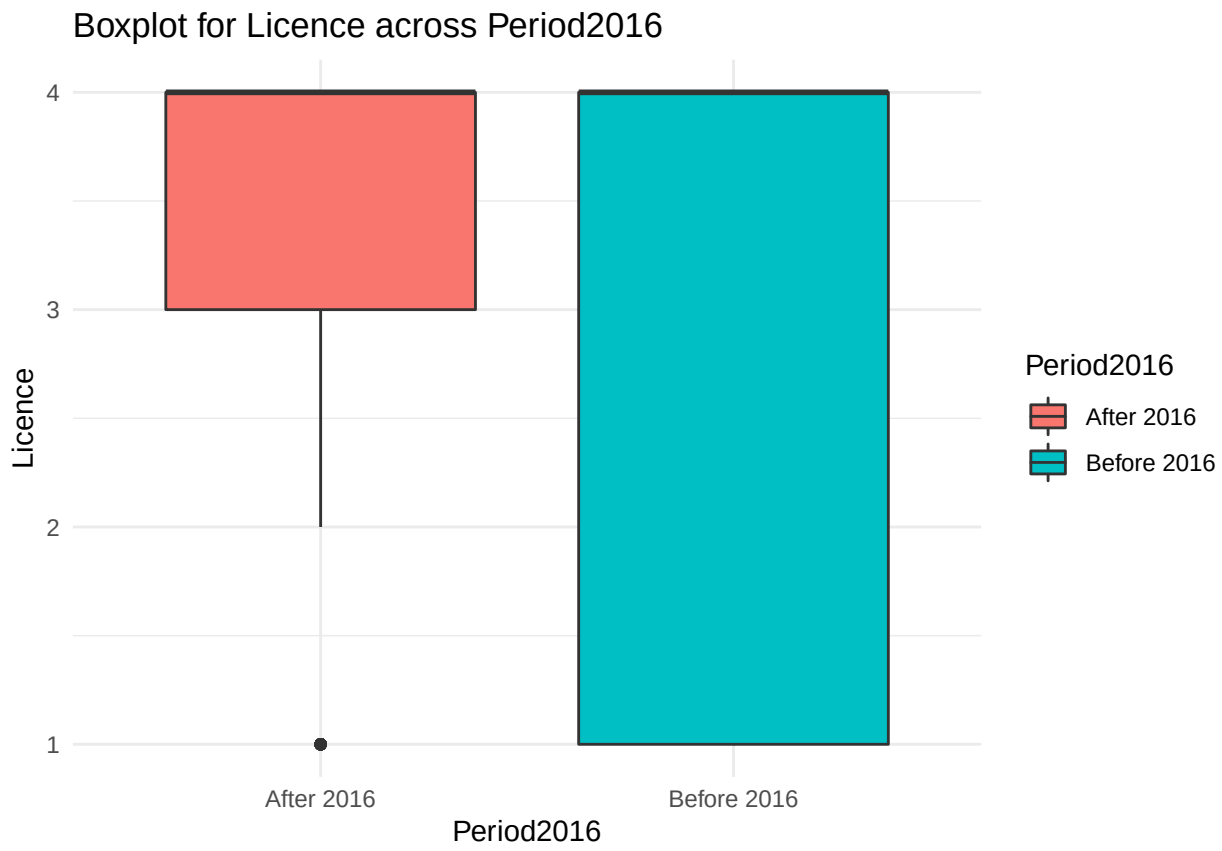

```
results_access_2016
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      2
##
## $p.value
## [1] 0.03765665
##
## $plot
```




```
results_licence_2016
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      4
##
## $p.value
## [1] 0.08882603
##
## $plot
```



#Study the significant difference before and after Fair 2016 using Median and Mann Whitney U test for the NCD Papers

```
perform_analysis <- function(data_NCD, score_var, period_var) {
  test_result <- wilcox.test(reformulate(period_var, score_var), data = data_NCD, exact = FALSE)

  medians <- data_NCD %>%
    group_by(!!sym(period_var)) %>%
    summarise(median = median(!!sym(score_var), na.rm = TRUE), .groups = 'drop')

  list(median = medians, p.value = test_result$p.value)
}
# Analysis for each score for 2016
results_complete_2016_NCD <- perform_analysis(data_NCD, "Complete", "Period2016")
results_reuse_2016_NCD <- perform_analysis(data_NCD, "Reuse", "Period2016")
results_access_2016_NCD <- perform_analysis(data_NCD, "Access", "Period2016")
results_licence_2016_NCD <- perform_analysis(data_NCD, "Licence", "Period2016")

# Print results
results_complete_2016_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       2
```

```
## 2 Before 2016      2
##
## $p.value
## [1] 0.09162029
```

```
results_reuse_2016_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       3
## 2 Before 2016      3
##
## $p.value
## [1] 0.03941278
```

```
results_access_2016_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      2
##
## $p.value
## [1] 0.1542596
```

```
results_licence_2016_NCD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      4
##
## $p.value
## [1] 0.15619
```

#Study the significant difference before and after Fair 2016 using Median and Mann Whitney U test for the InfD Papers

```
# Analysis for each score for 2016
results_complete_2016_InfD <- perform_analysis(data_InfD, "Complete", "Period2016")
results_reuse_2016_InfD <- perform_analysis(data_InfD, "Reuse", "Period2016")
results_access_2016_InfD <- perform_analysis(data_InfD, "Access", "Period2016")
results_licence_2016_InfD <- perform_analysis(data_InfD, "Licence", "Period2016")

# Print results
results_complete_2016_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       2
## 2 Before 2016      2
##
## $p.value
## [1] 0.2801454
```

```
results_reuse_2016_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       3
## 2 Before 2016      2
##
## $p.value
## [1] 0.02973854
```

```
results_access_2016_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      2
##
## $p.value
## [1] 0.147289
```

```
results_licence_2016_InfD
```

```
## $median
## # A tibble: 2 x 2
##   Period2016  median
##   <chr>        <dbl>
## 1 After 2016       4
## 2 Before 2016      4
##
## $p.value
## [1] 0.3305261
```

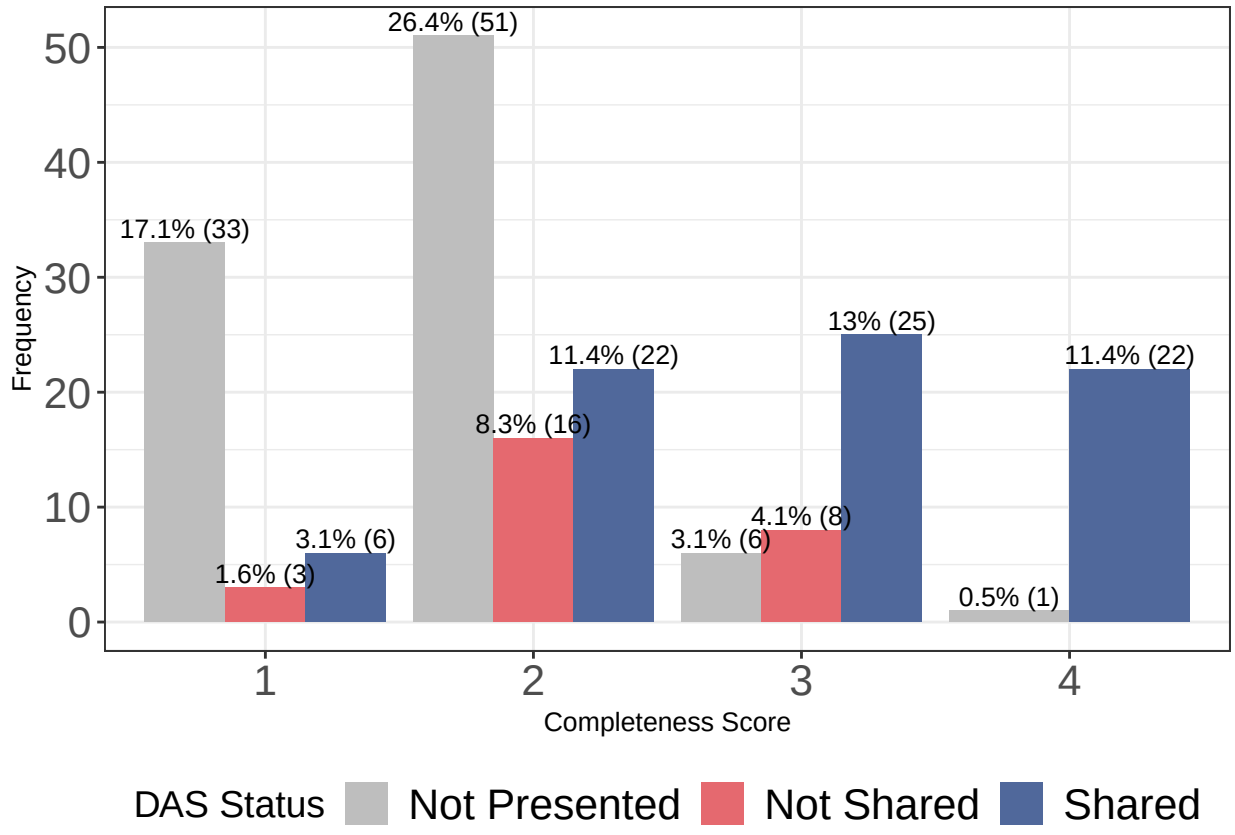
#plot the distribution of DAS for each completeness score

```
# Transform the DAS variable
data <- data %>%
  mutate(NewDAS = case_when(
    is.na(DAS) ~ "Not Presented",
```

```
    DAS == 1 ~ "Shared",
    DAS == 0 ~ "Not Shared",
    TRUE ~ as.character(DAS)))  # This line is just a fallback to handle unexpected values

class(data$NewDAS)
```

```
## [1] "character"
```

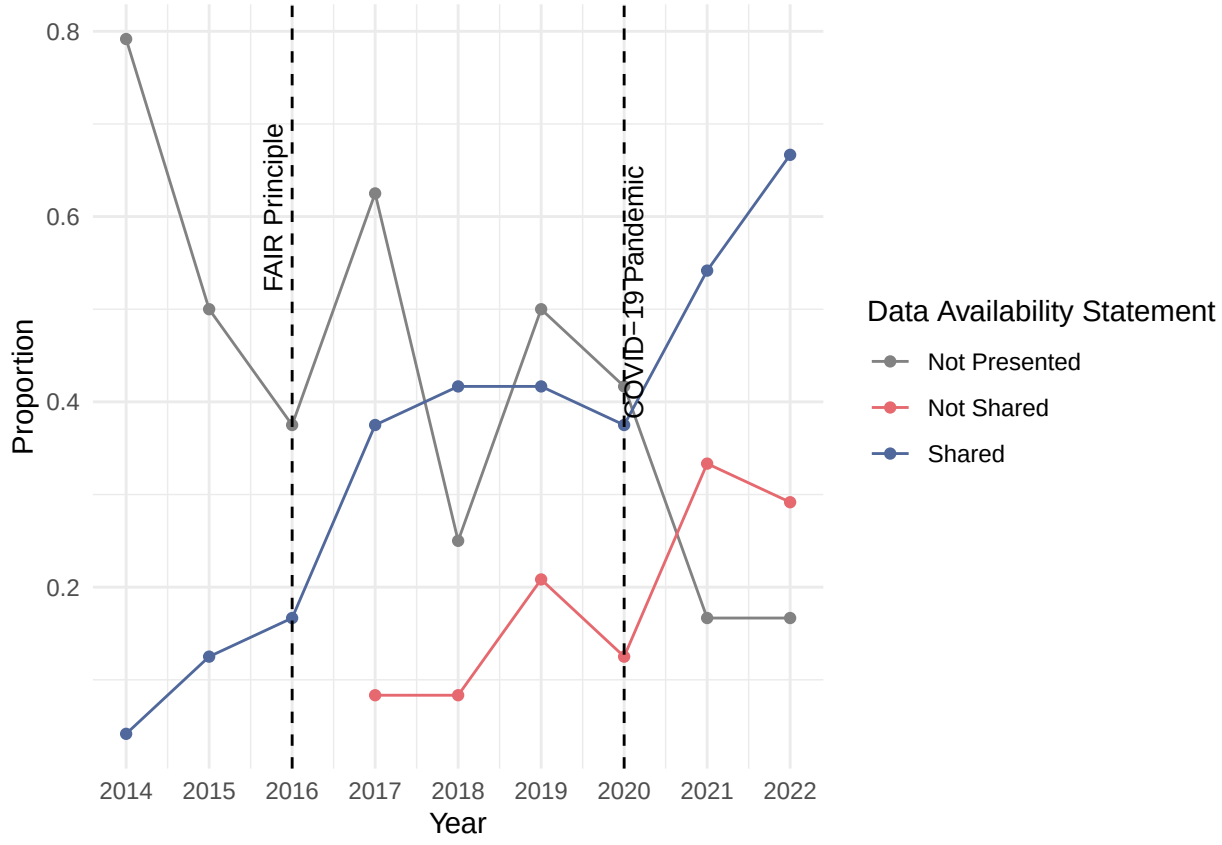
```
# Convert the new DAS variable to a factor for plotting
data$NewDAS <- factor(data$NewDAS, levels = c("Not Presented", "Not Shared", "Shared"))

# Calculate frequency and percentage for each combination of 'Complete' score and 'NewDAS' status
das_frequency <- data %>%
  group_by(Complete, NewDAS) %>%
  summarise(Frequency = n(), .groups = 'drop') %>%
  mutate(Percentage = (Frequency / sum(Frequency)) * 100)

# Define specific colors for the new DAS values
colors <- c("Not Presented" = "gray", "Not Shared" = "#E5696F", "Shared" = "#50689B")

# Create the plot
das_plot <- ggplot(das_frequency, aes(x = as.factor(Complete), y = Frequency, fill = NewDAS)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = paste0(round(Percentage, 1), "% (", Frequency, ")")),
            position = position_dodge(width = 0.9), vjust = -0.25, size = 3.5) +
  scale_fill_manual(values = colors, name = "DAS Status") +
  theme_bw() +  # Use theme_bw for a white background
   theme(
    axis.title = element_text(size = 10), # Increase axis titles
    axis.text = element_text(size = 16),  # Increase axis text
    legend.title = element_text(size = 14), # Increase legend title
    legend.text = element_text(size = 16)  # Increase legend text
  ) +
  labs(x = "Completeness Score", y = "Frequency") +
  theme(legend.position = "bottom")

# Print and save the plot
print(das_plot)
```




```
ggsave("new_das_distribution_plot.png", das_plot, width = 8, height = 6,  bg = "white")
```

```
# Calculate the total count per year
yearly_totals <- data %>%
  group_by(Year) %>%
  summarise(Total = n(), .groups = 'drop')

# Join the totals back to the original data and calculate proportions
proportion_data <- data %>%
  left_join(yearly_totals, by = "Year") %>%
  group_by(Year, NewDAS) %>%
  summarise(Count = n(), .groups = 'drop') %>%
  mutate(Proportion = Count / n())

# Get all unique years for the x-axis
all_years <- sort(unique(proportion_data$Year))

colors <- c("Not Presented" = "#828282", "Not Shared" = "#E5696F", "Shared" = "#50689B")

# Plot with every year on the x-axis
DASyear <- ggplot(proportion_data, aes(x = Year, y = Proportion, color = NewDAS, group = NewDAS)) +
  geom_line() +
  geom_point() +
  geom_vline(xintercept = 2016, linetype = "dashed", color = "black") +
  geom_vline(xintercept = 2020, linetype = "dashed", color = "black") +
  annotate("text", x = 2016, y = max(proportion_data$Proportion), label = "FAIR Principle", angle = 90,
```

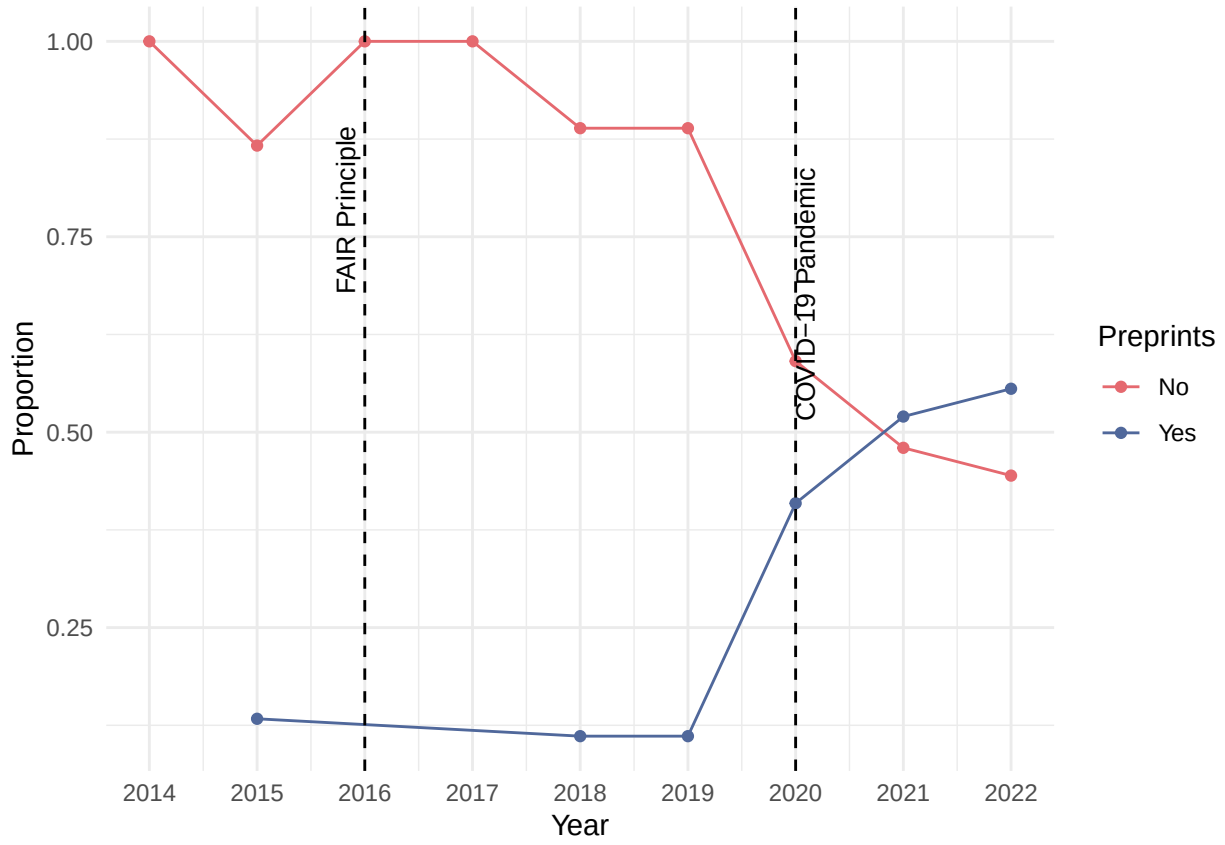
```
  annotate("text", x = 2020, y = max(proportion_data$Proportion), label = "COVID-19 Pandemic", angle = 9
  scale_x_continuous(breaks = all_years) +
  labs(x = "Year", y = "Proportion", color = "Data Availability Statement") +
  theme(
    axis.title = element_text(size = 10), # Increase axis titles
    axis.text = element_text(size = 16),  # Increase axis text
    legend.title = element_text(size = 14), # Increase legend title
    legend.text = element_text(size = 16)  # Increase legend text
  ) +
  theme_minimal() +
  scale_color_manual(values = colors)

print(DASyear)
```




```
ggsave("DASyear.png", DASyear, width = 8, height = 6,  bg = "white")
```

```
 #Convert the Preprint numeric values to factor levels 'Yes' and 'No'
data$Preprints <- factor(ifelse(data$Preprints == 1, "Yes", "No"))

# Calculate the total count per year and proportion for Preprint
proportion_data <- data %>%
  group_by(Year) %>%
  count(Preprints) %>%
  mutate(Total = sum(n),
         Proportion = n / Total)
```

```r
# Get all unique years for the x-axis
all_years <- sort(unique(proportion_data$Year))

# Define colors for 'Yes' and 'No'
colors <- c("No" = "#E5696F", "Yes" = "#50689B")

# Plot with every year on the x-axis for Preprints
PreprintYear <- ggplot(proportion_data, aes(x = Year, y = Proportion, color = Preprints, group = Prepri
  geom_line() +
  geom_point() +
  geom_vline(xintercept = 2016, linetype = "dashed", color = "black") +
  geom_vline(xintercept = 2020, linetype = "dashed", color = "black") +
  annotate("text", x = 2016, y = max(proportion_data$Proportion), label = "FAIR Principle", angle = 90,
  annotate("text", x = 2020, y = max(proportion_data$Proportion), label = "COVID-19 Pandemic", angle = 9
  scale_x_continuous(breaks = all_years) +
  labs(x = "Year", y = "Proportion", color = "Preprints") +
  theme(
    axis.title = element_text(size = 10), # Increase axis titles
    axis.text = element_text(size = 16),  # Increase axis text
    legend.title = element_text(size = 14), # Increase legend title
    legend.text = element_text(size = 16)  # Increase legend text
  ) +
  theme_minimal() +
  scale_color_manual(values = colors)

print(PreprintYear)
```




```
# Save the plot for Preprint
ggsave("PreprintYear.png", PreprintYear, width = 8, height = 6, bg = "white")
```

#study if FAIR implementation and Covid19 have an effect on DAS and Preprint # 1- Fair implementataion

```
total_by_period <- data %>%
  group_by(Period2016) %>%
  summarise(Total = n(), .groups = "drop")

# Join this total with the DAS and Preprints summaries and calculate the percentage
Preprints_summary <- data %>%
  group_by(Period2016, Preprints) %>%
  summarise(Frequency = n(), .groups = "drop") %>%
  left_join(total_by_period, by = "Period2016") %>%
  mutate(Percentage = (Frequency / Total) * 100)

# Print the Preprints summary table
print(Preprints_summary)
```

```
## # A tibble: 4 x 5
##   Period2016  Preprints Frequency Total Percentage
##   <chr>       <fct>         <int> <int>      <dbl>
## 1 After 2016  No              103   145       71.0
## 2 After 2016  Yes              42   145       29.0
## 3 Before 2016 No               46    48       95.8
## 4 Before 2016 Yes               2    48        4.17
```

```r
Preprints_table <- table(data$Preprints, data$Period2016)

# Perform the Chi-square test
result <- chisq.test(Preprints_table)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  Preprints_table
## X-squared = 11.231, df = 1, p-value = 0.0008044
```

```r
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##       After 2016 Before 2016
##   No   111.94301    37.05699
##   Yes   33.05699    10.94301
```

```r
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

```r
total_by_period <- data %>%
  group_by(Period2016) %>%
  summarise(Total = n(), .groups = "drop")

# Join this total with the NewDAS and Preprints summaries and calculate the percentage
das_summary <- data %>%
  group_by(Period2016, NewDAS)%>%
  summarise(Frequency = n(), .groups = "drop") %>%
  left_join(total_by_period, by = "Period2016") %>%
  mutate(Percentage = (Frequency / Total) * 100)

# Print the DAS summary table
print(das_summary)
```

```
## # A tibble: 5 x 5
##   Period2016  NewDAS         Frequency Total Percentage
##   <chr>       <fct>              <int> <int>      <dbl>
## 1 After 2016  Not Presented         51   145       35.2
## 2 After 2016  Not Shared           27   145       18.6
```

```
## 3 After 2016  Shared               67   145       46.2
## 4 Before 2016 Not Presented        40    48       83.3
## 5 Before 2016 Shared                8    48       16.7
```

```
das_table <- table(data$NewDAS, data$Period2016)

# Perform the Chi-square test
result <- chisq.test(das_table)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test
##
## data:  das_table
## X-squared = 34.776, df = 2, p-value = 2.809e-08
```

```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##                 After 2016 Before 2016
##   Not Presented   68.36788   22.632124
##   Not Shared      20.28497    6.715026
##   Shared          56.34715   18.652850
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal).
```

#2- Covid 19 # the impact of COVID-19 on Preprints

```
total_by_period <- data %>%
  group_by(Period2020)%>%
  summarise(Total = n(), .groups = "drop")

# Join this total with the DAS and Preprints summaries and calculate the percentage
Preprints_summary <- data %>%
  group_by(Period2020, Preprints) %>%
  summarise(Frequency = n(), .groups = "drop") %>%
  left_join(total_by_period, by = "Period2020") %>%
  mutate(Percentage = (Frequency / Total) * 100)

# Print the Preprints summary table
print(Preprints_summary)
```

```
## # A tibble: 4 x 5
##   Period2020  Preprints Frequency Total Percentage
##   <chr>       <fct>         <int> <int>      <dbl>
## 1 After 2020  No               24    52       46.2
## 2 After 2020  Yes              28    52       53.8
## 3 Before 2020 No              125   141       88.7
## 4 Before 2020 Yes              16   141       11.3
```

```
Preprints_table <- table(data$Preprints, data$Period2020)

# Perform the Chi-square test
result <- chisq.test(Preprints_table)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  Preprints_table
## X-squared = 36.607, df = 1, p-value = 1.445e-09
```

```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##       After 2020 Before 2020
##   No    40.14508   108.85492
##   Yes   11.85492    32.14508
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal)
```

the impact of COVID-19 on DAS

```
total_by_period <- data %>%
  group_by(Period2020) %>%
  summarise(Total = n(), .groups = "drop")

# Join this total with the NewDAS and Preprints summaries and calculate the percentage
das_summary <- data %>%
  group_by(Period2020, NewDAS)%>%
```

```
  summarise(Frequency = n(), .groups = "drop") %>%
  left_join(total_by_period, by = "Period2020") %>%
  mutate(Percentage = (Frequency / Total) * 100)

# Print the DAS summary table
print(das_summary)
```

```
## # A tibble: 6 x 5
##   Period2020  NewDAS         Frequency Total Percentage
##   <chr>       <fct>              <int> <int>      <dbl>
## 1 After 2020  Not Presented          8    52      15.4
## 2 After 2020  Not Shared            15    52      28.8
## 3 After 2020  Shared                29    52      55.8
## 4 Before 2020 Not Presented         83   141      58.9
## 5 Before 2020 Not Shared            12   141       8.51
## 6 Before 2020 Shared                46   141      32.6
```

```
das_table <- table(data$NewDAS, data$Period2016)

# Perform the Chi-square test
result <- chisq.test(das_table)

#Print the chi-square results
print(result)
```

```
##
##  Pearson's Chi-squared test
##
## data:  das_table
## X-squared = 34.776, df = 2, p-value = 2.809e-08
```

```
# Get the expected values
expected_values <- result$expected

# Print the expected values
print(expected_values) ##in order to check the assumption of chi square test: Each cell in the continge
```

```
##
##                 After 2016 Before 2016
##   Not Presented   68.36788   22.632124
##   Not Shared      20.28497    6.715026
##   Shared          56.34715   18.652850
```

```
#The assumptions of that test were met:

#### Sample Size: Each cell in the contingency table has an expected count of 5 or more.
#### Independence: The observations are  independent of each other. This means that the selection of on
### Random Sampling: The data is a random sample.
### Categorical Data: Both variables should be categorical (either nominal or ordinal)
```

#2 - Ordinal regression models (Scoring Criteria and year)

```r
library(ordinal) # ordinal logistic regression: cumulative link mixed models, clm function is in this p
```

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

```
##
## Attaching package: 'ordinal'
```

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

```r
library(VGAM) # more ordinal regression
```

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

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```
##
## Attaching package: 'VGAM'
```

```
## The following objects are masked from 'package:ordinal':
##
##     dgumbel, dlgamma, pgumbel, plgamma, qgumbel, rgumbel, wine
```

```r
library(dplyr) # Data manipulation
library(chisq.posthoc.test) # If needed
```

```
## Warning: package 'chisq.posthoc.test' was built under R version 4.2.3
```

```r
library(gmodels) # For SPSS style chi-sq/ contingency tables
```

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

```r
library(emmeans)
```

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

```r
library(sure)
```

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

```r
data$Complete <- factor(data$Complete, ordered = TRUE)
data$Reuse <- factor(data$Reuse, ordered = TRUE)
data$Access <- factor(data$Access, ordered = TRUE)
data$Licence <- factor(data$Licence, ordered = TRUE)

levels(data$Complete)
```

```
## [1] "1" "2" "3" "4"
```

```r
levels(data$Reuse)
```

```
## [1] "1" "2" "3" "4"
```

```r
levels(data$Access)
```

```
## [1] "1" "2" "3" "4"
```

```r
levels(data$Licence)
```

```
## [1] "1" "2" "3" "4"
```

#Ordinal regression model for the Completeness by year

```r
m1a <- clm(Complete ~ Year, data = data)
```

```
## Warning in x$code == 0L || action == "silent": 'length(x) = 2 > 1' in coercion
## to 'logical(1)'
```

```
## Warning: (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```r
summary_m1a <- summary(m1a)


# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                 OR      LowerCI       UpperCI      p_value
## 1|2  7.098397e+156 3.817148e+64 1.320023e+249 0.0008626939
## 2|3  5.984716e+157 2.942469e+65 1.217237e+250 0.0008077966
## 3|4  2.197013e+158 1.041525e+66 4.634421e+250 0.0007751331
## Year  1.196741e+00 1.077146e+00  1.329615e+00 0.0008275139
```

```r
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Complete ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -238.52 485.05
## Year
```

```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

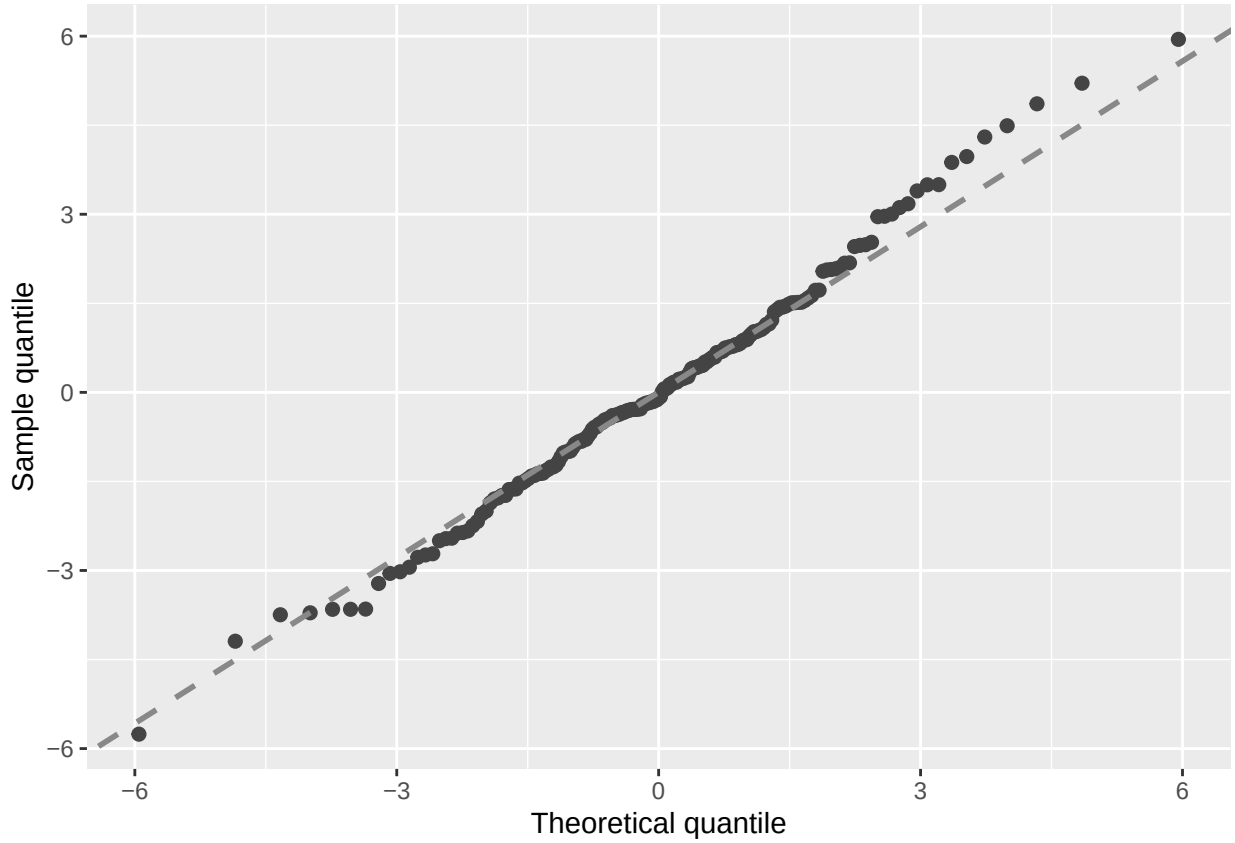
```
## Warning: (-2) Model failed to converge: degenerate Hessian with 1 negative eigenvalues
## In addition: iteration limit reached
```

```
## Tests of scale effects
##
## formula: Complete ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -238.52 485.05
## Year
```

```
convergence(m1a) # This is another way to assess the model
```

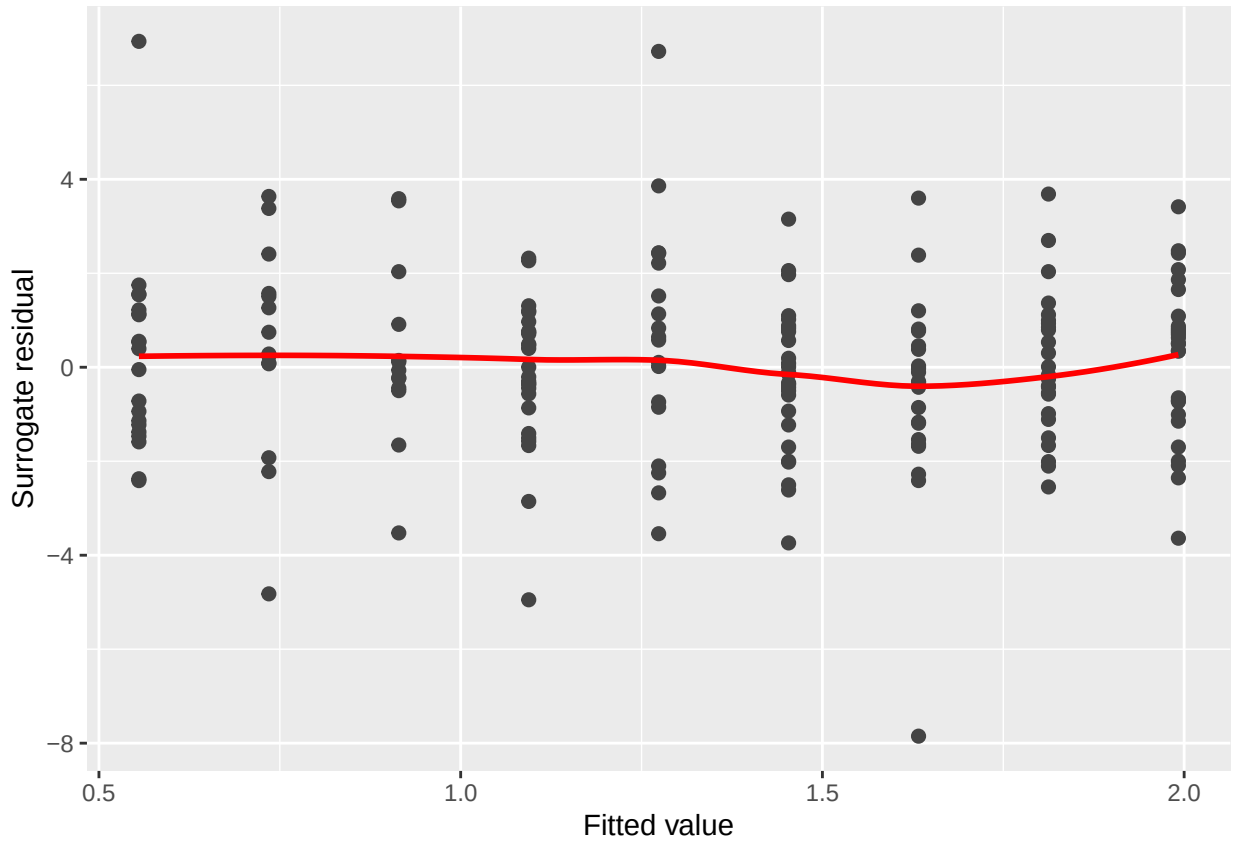
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -238.52 5(0)  9.97e-08 8.1e+12 <1e-10
##
##      Estimate   Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2  361.1631 108.39704  4.74e-10 -3.95e-09       8      11
## 2|3  363.2951 108.44275 -4.15e-10 -3.97e-09       8      11
## 3|4  364.5955 108.46135 -1.03e-11 -3.97e-09       8      11
## Year   0.1796   0.05372 -9.97e-08 -1.96e-12      11      11
##
## Eigen values of Hessian:
## 2.299e+08 7.161e+01 2.950e+01 2.836e-05
##
## Convergence message from clm:
## (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## (3) Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



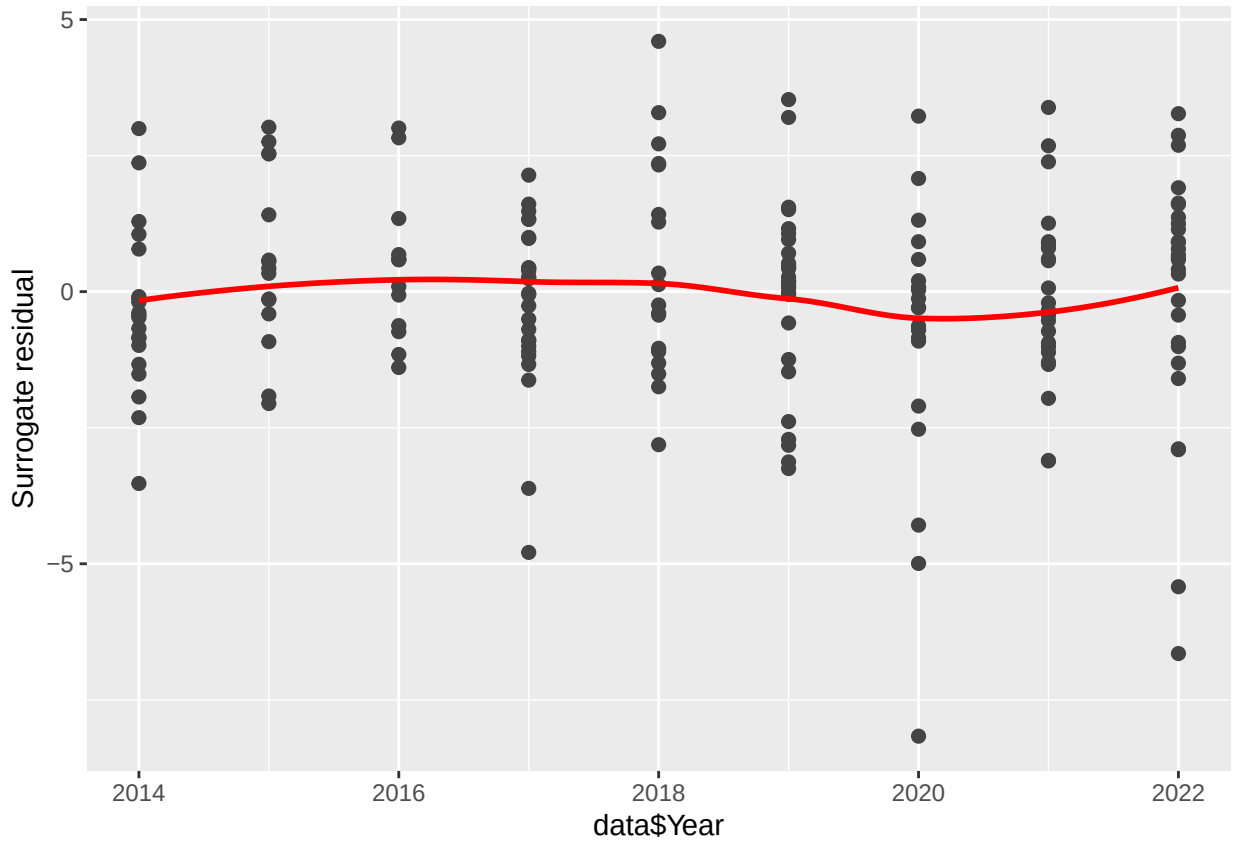

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
autoplot.clm(m1a, what = c("covariate"), x = data$Year) # scale test indicated no variance issues. This
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
# N.B. - Interpreting residual plots is largely subjective!

###plotting the model
# Predict probabilities for each level of 'Complete'
new_data <- data.frame(Year = sort(unique(data$Year)))
pred_probs <- predict(m1a, newdata = new_data, type = "prob")

# Add the 'Year' column to the predicted probabilities dataframe
pred_probs_df <- cbind(new_data, as.data.frame(pred_probs))

# Convert the predicted probabilities to a long format for ggplot
library(reshape2)
```

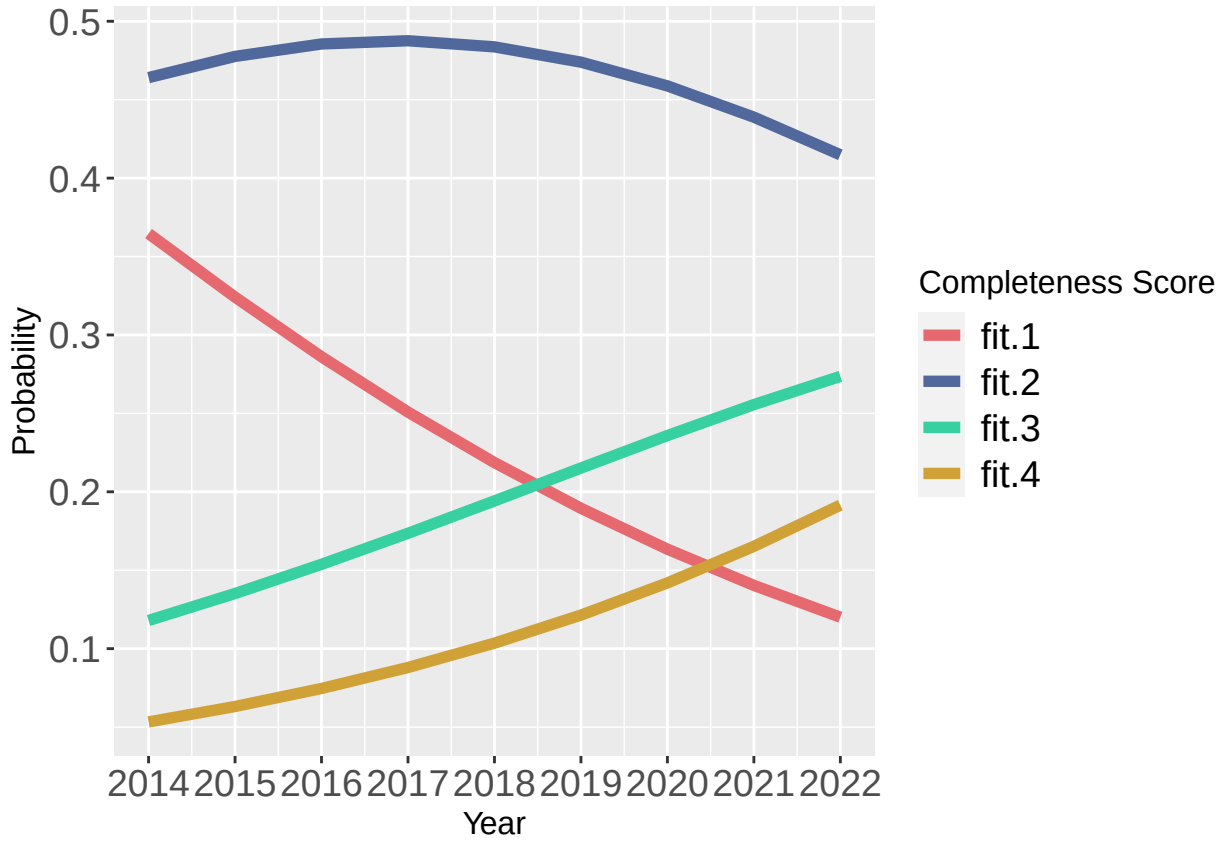
```
## Warning: package 'reshape2' was built under R version 4.2.3
```

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

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
pred_probs_long <- melt(pred_probs_df, id.vars = 'Year', variable.name = 'CompleteLevel', value.name =
```

```r
# Set the colors
my_colors <- c("#E5696F", "#50689B", "#36D0A1", "#D0A136")

# Modify ggplot command
og1 <- ggplot(pred_probs_long, aes(x = Year, y = PredictedProbability, group = CompleteLevel, color = C
  geom_line(size = 2) +  # Set size of the lines to make them thicker
  scale_color_manual(values = my_colors) +
  scale_x_continuous(breaks = unique(pred_probs_long$Year)) +  # Show all years on x-axis
  labs(x = "Year", y = "Probability", color = "Completeness Score") +
  theme(
    axis.title = element_text(size = 12), # Increase axis titles
    axis.text = element_text(size = 14),  # Increase axis text
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  )
print(og1)
```




```r
ggsave("og1.png", og1, width = 15, height = 10, units = "in",  bg = "white")
```

#Ordinal regression model for the Reusibility by year

```r
m1a <- clm(Reuse ~ Year, data = data)
```

```
## Warning in x$code == 0L || action == "silent": 'length(x) = 2 > 1' in coercion
## to 'logical(1)'
```

```
## Warning: (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
summary_m1a <- summary(m1a)


# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                  OR      LowerCI       UpperCI      p_value
## 1|2  2.349088e+189 9.057314e+96 6.092548e+281 5.910538e-05
## 2|3  4.254302e+189 1.602643e+97 1.129328e+282 5.785682e-05
## 3|4  1.399145e+190 4.955124e+97 3.950672e+282 5.549741e-05
## Year  1.241997e+00 1.117686e+00  1.380133e+00 5.629614e-05
```

```
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Reuse ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -240.66 489.32
## Year
```

```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

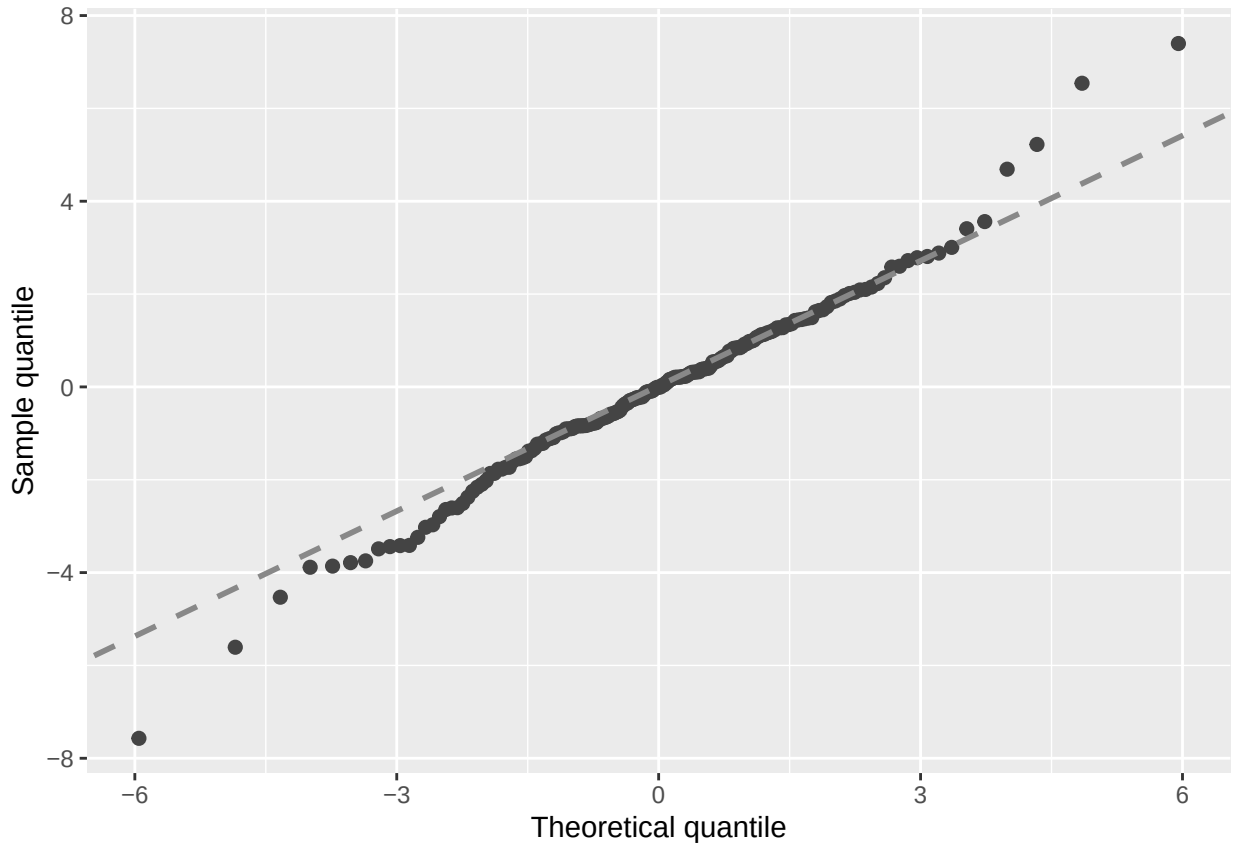
```
## Warning: (-1) Model failed to converge with max|grad| = 451.581 (tol = 1e-06)
## In addition: iteration limit reached
```

```
## Tests of scale effects
##
## formula: Reuse ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -240.66 489.32
## Year
```

```
convergence(m1a) # This is another way to assess the model
```

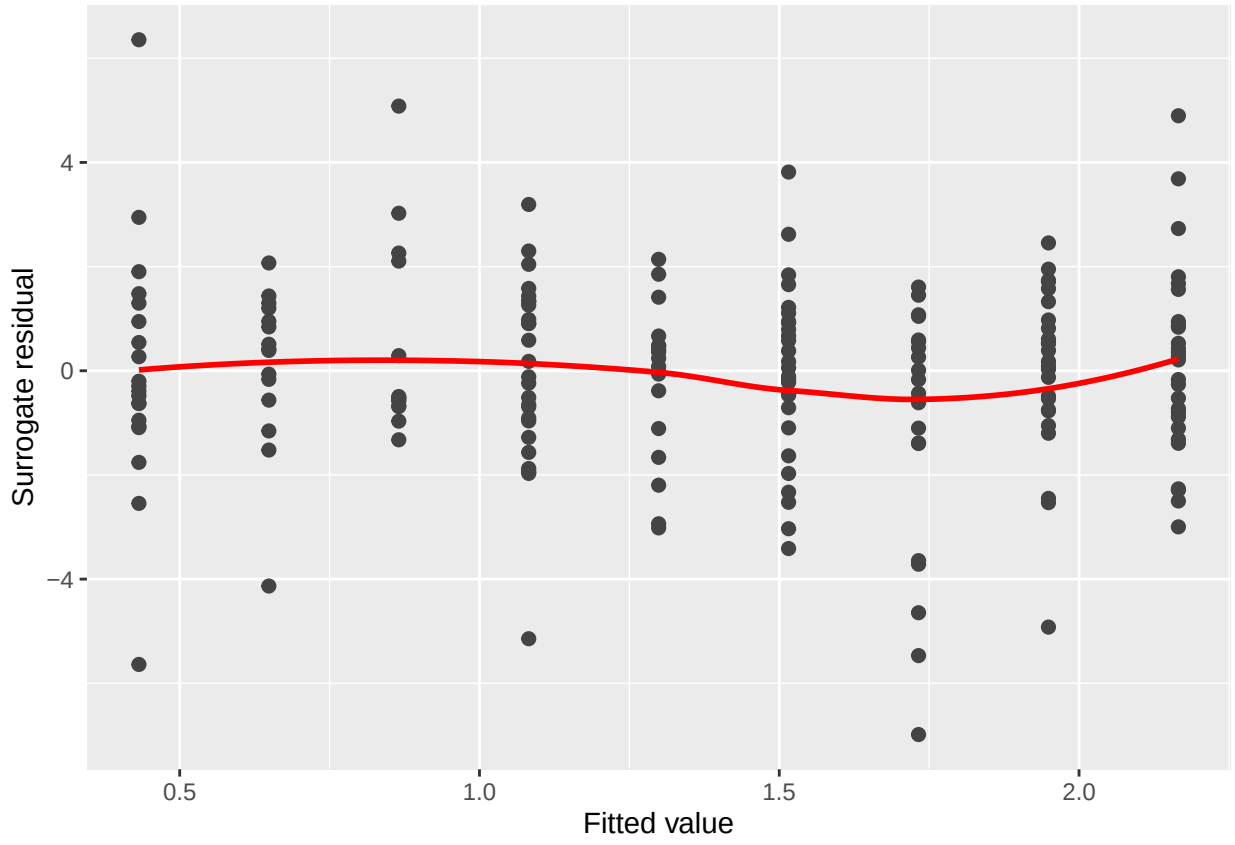
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -240.66 7(0)  1.79e-09 8.2e+12 <1e-10
##
##      Estimate   Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2  436.0426 108.56677  2.06e-12 -2.72e-11      10      13
## 2|3  436.6365 108.57863 -2.78e-12 -2.73e-11      10      13
## 3|4  437.8270 108.61013  1.60e-12 -2.73e-11      10      13
## Year   0.2167   0.05381 -1.79e-09 -1.35e-14      13      13
##
## Eigen values of Hessian:
## 2.310e+08 1.550e+02 5.908e+01 2.827e-05
##
## Convergence message from clm:
## (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## (3) Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearity
```



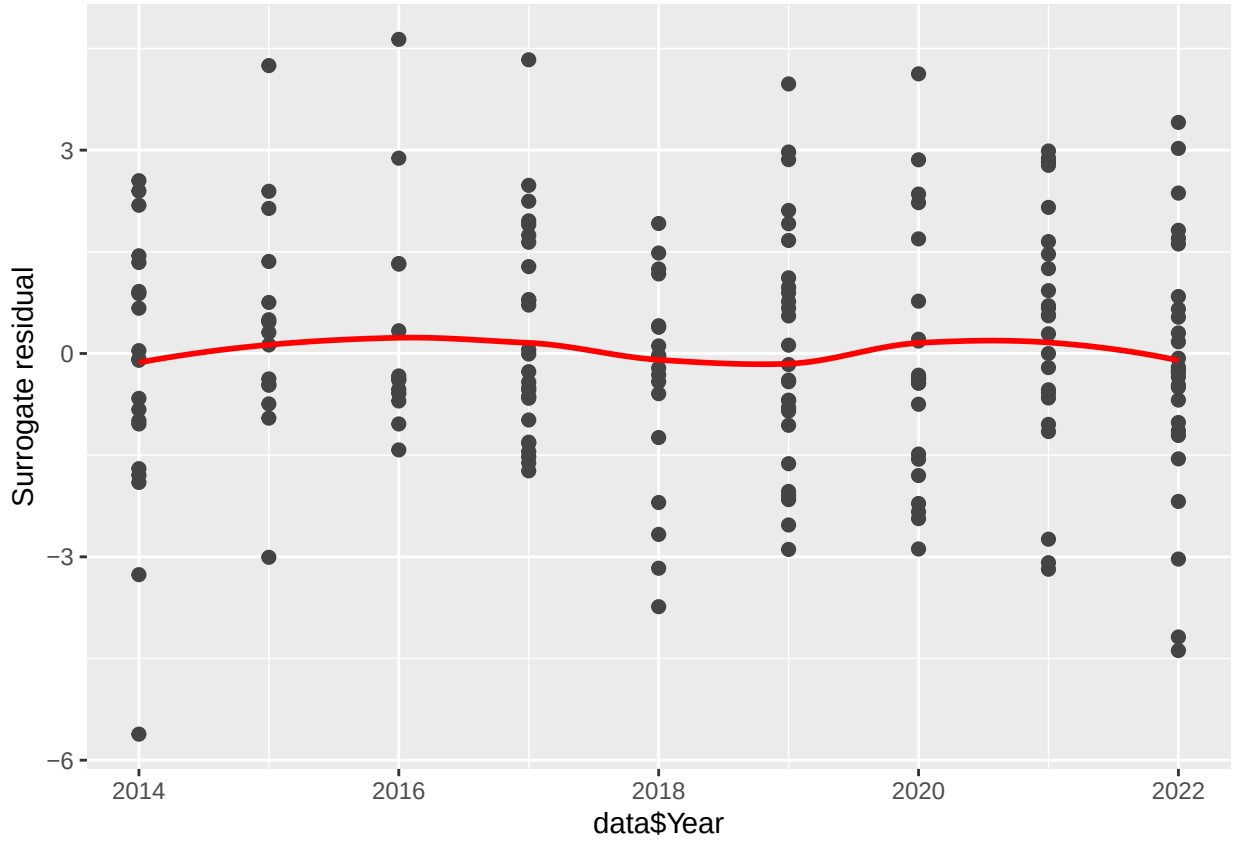
```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



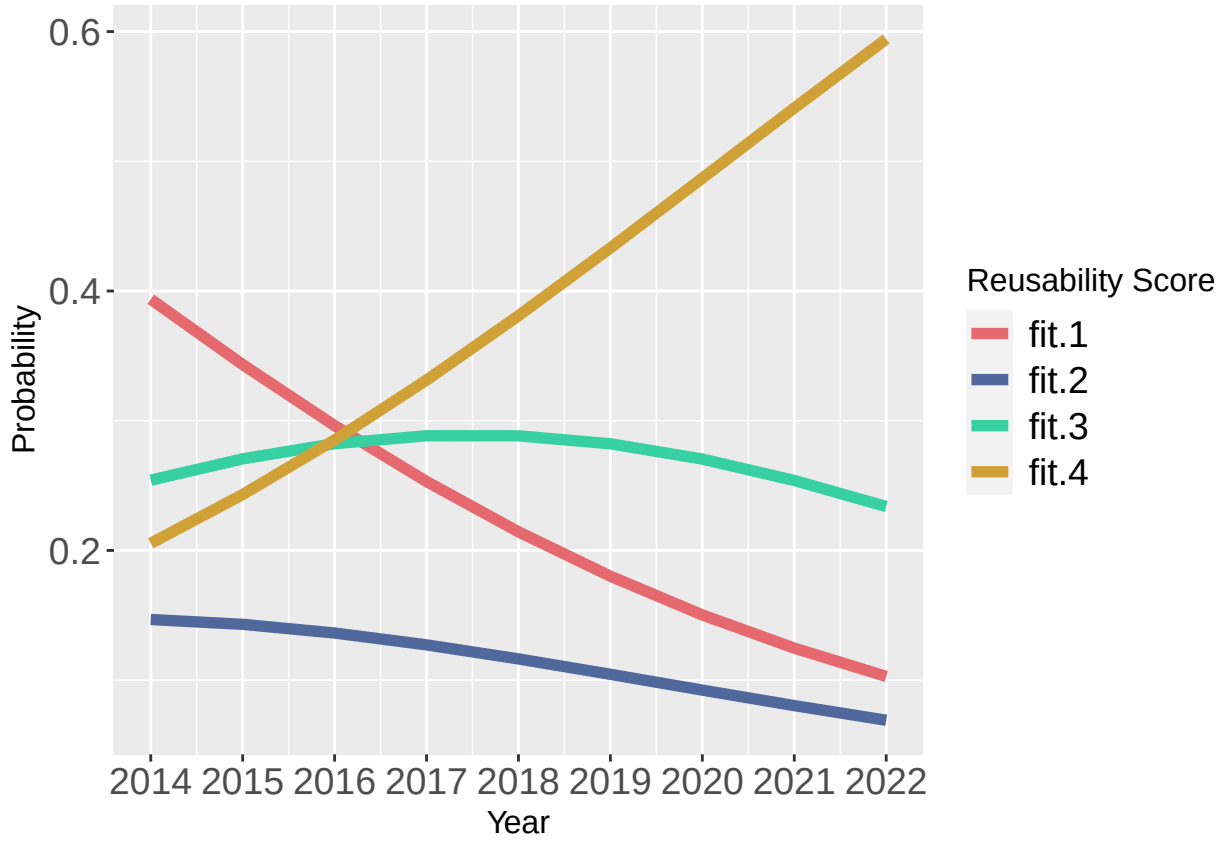

```
autoplot.clm(m1a, what = c("covariate"), x = data$Year) # scale test indicated no variance issues. This
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```r
# N.B. - Interpreting residual plots is largely subjective!

###plotting the model
# Predict probabilities for each level of 'Complete'
new_data <- data.frame(Year = sort(unique(data$Year)))
pred_probs <- predict(m1a, newdata = new_data, type = "prob")

# Add the 'Year' column to the predicted probabilities dataframe
pred_probs_df <- cbind(new_data, as.data.frame(pred_probs))

# Convert the predicted probabilities to a long format for ggplot
library(reshape2)
pred_probs_long <- melt(pred_probs_df, id.vars = 'Year', variable.name = 'Reuselevel', value.name = 'Pr

# Set the colors
my_colors <- c("#E5696F", "#50689B", "#36D0A1", "#D0A136")

# Modify ggplot command
og2 <- ggplot(pred_probs_long, aes(x = Year, y = PredictedProbability, group = Reuselevel, color = Reus
  geom_line(size = 2) +  # Set size of the lines to make them thicker
  scale_color_manual(values = my_colors) +  # Use colors
  scale_x_continuous(breaks = unique(pred_probs_long$Year)) +  # Show all years on x-axis
  labs(x = "Year", y = "Probability", color = "Reusability Score") +
  theme(
    axis.title = element_text(size = 12), # Increase axis titles
    axis.text = element_text(size = 14),  # Increase axis text
```

```
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  )
print(og2)
```




```
ggsave("og2.png", og2, width = 15, height = 10, units = "in",  bg = "white")
```

```
m1a <- clm(Access ~ Year, data = data)
```

```
## Warning in x$code == 0L || action == "silent": 'length(x) = 2 > 1' in coercion
## to 'logical(1)'
```

```
## Warning: (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
summary_m1a <- summary(m1a)


# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors
```

```r
# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                 OR      LowerCI       UpperCI      p_value
## 1|2  3.320521e+169 6.180429e+74 1.783996e+264 0.0004524204
## 2|3  1.065638e+170 1.898440e+75 5.981670e+264 0.0004360872
## 3|4  1.273957e+170 2.252216e+75 7.206093e+264 0.0004336612
## Year  1.214171e+00 1.089767e+00  1.352777e+00 0.0004336952
```

```r
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Access ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -215.84 439.68
## Year
```

```r
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

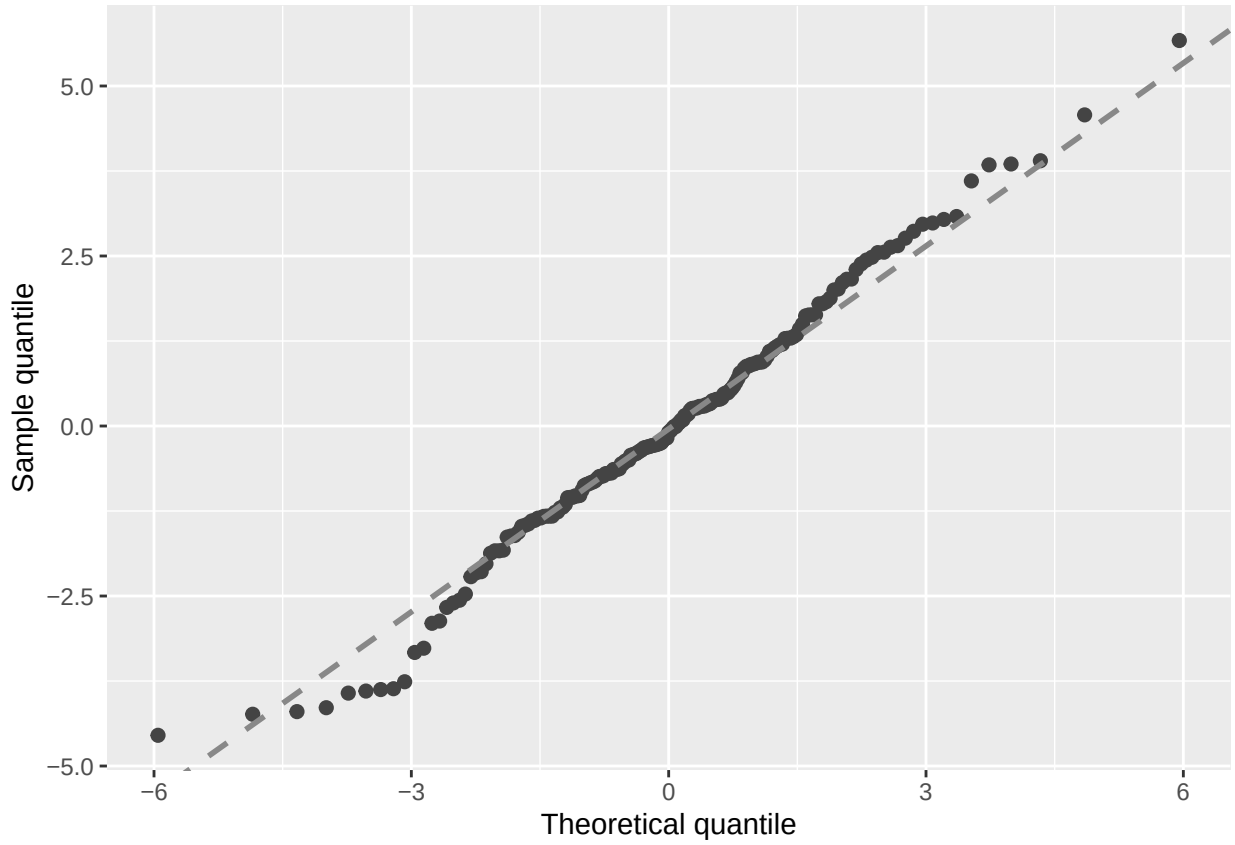
```
## Warning: (-2) Model failed to converge: degenerate Hessian with 1 negative eigenvalues
## In addition: iteration limit reached

## Tests of scale effects
##
## formula: Access ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -215.84 439.68
## Year
```

```r
convergence(m1a) # This is another way to assess the model
```

```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -215.84 6(2)  4.89e-09 8.0e+12 <1e-10
##
##      Estimate   Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2  390.3370 111.28792  2.50e-13 -2.16e-10       9      12
## 2|3  391.5030 111.31027 -5.25e-12 -2.16e-10       9      12
```

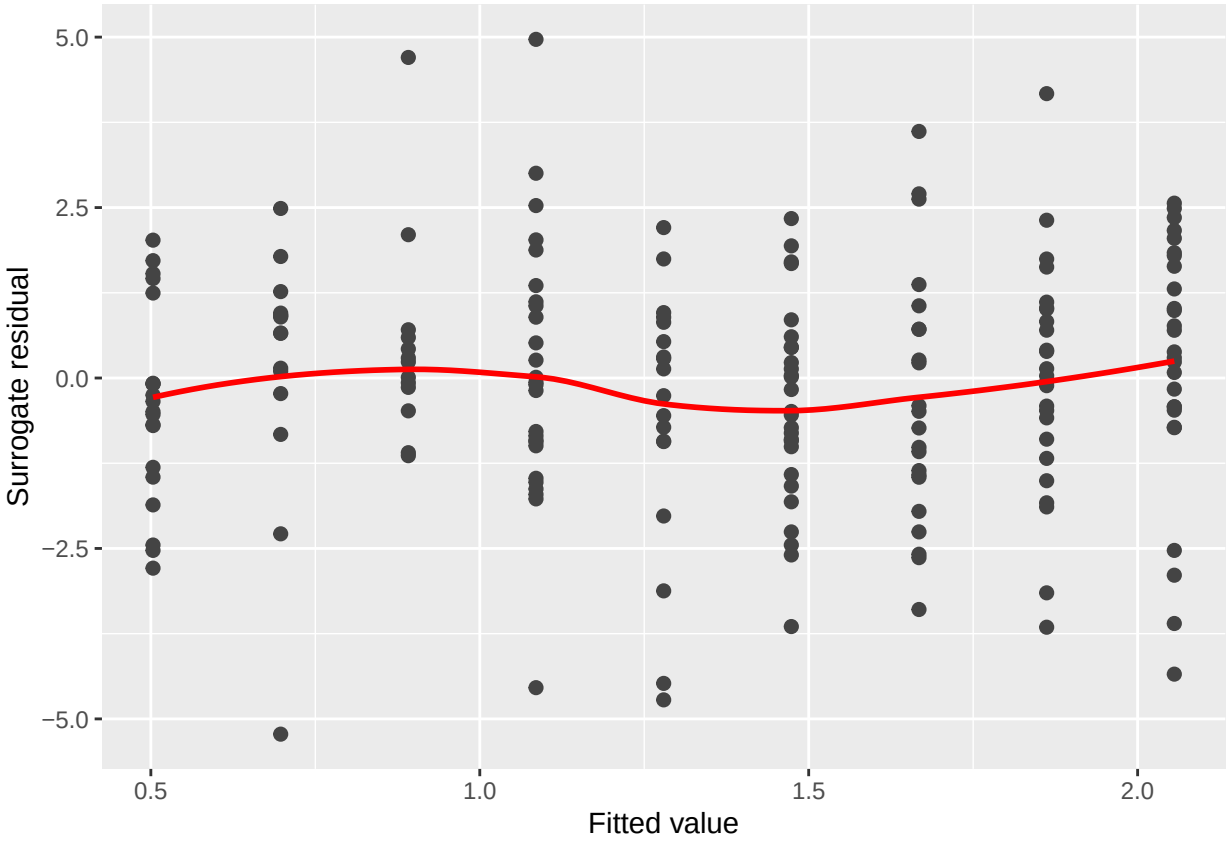
```
## 3|4  391.6816 111.31418  7.41e-12 -2.16e-10       9      12
## Year   0.1941   0.05515 -4.89e-09 -1.07e-13      12      12
##
## Eigen values of Hessian:
## 2.159e+08 5.342e+02 6.066e+01 2.691e-05
##
## Convergence message from clm:
## (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## (3) Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearity
```



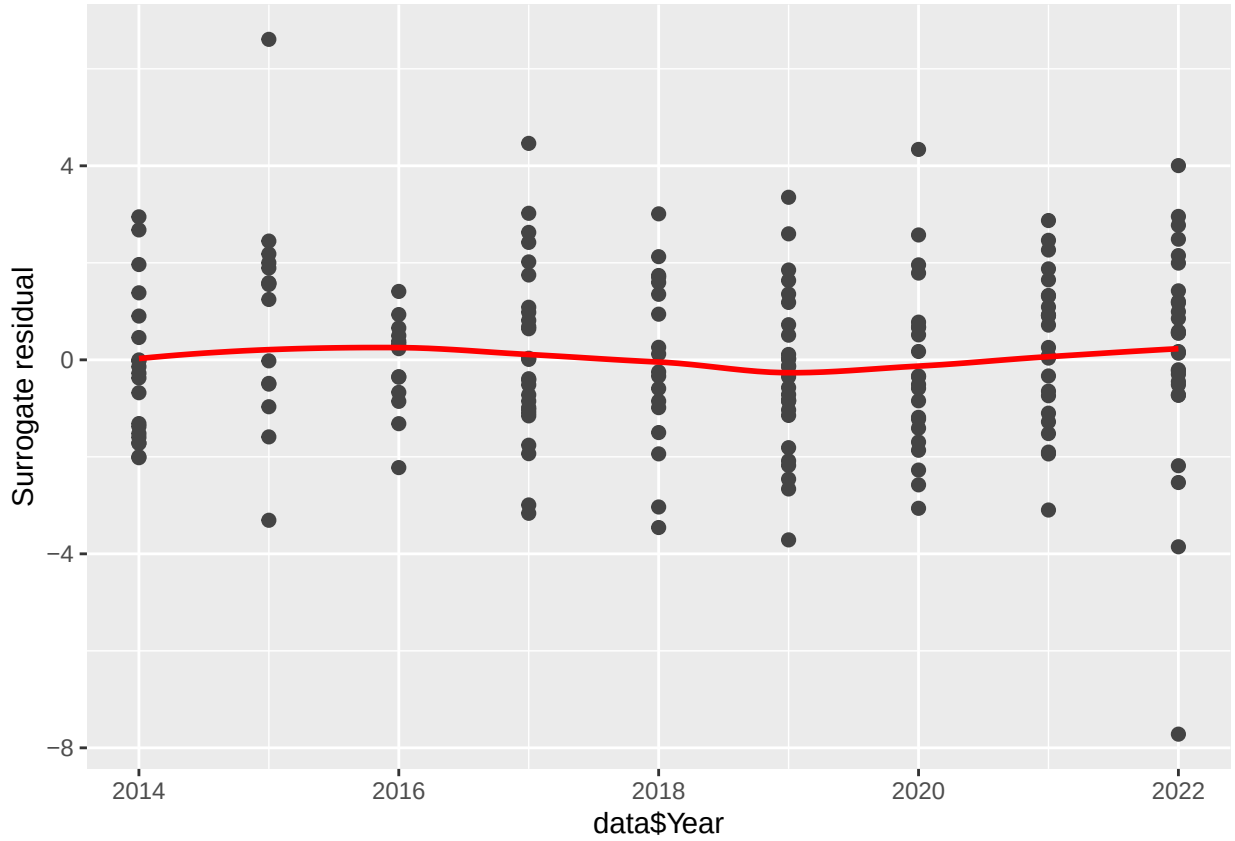

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



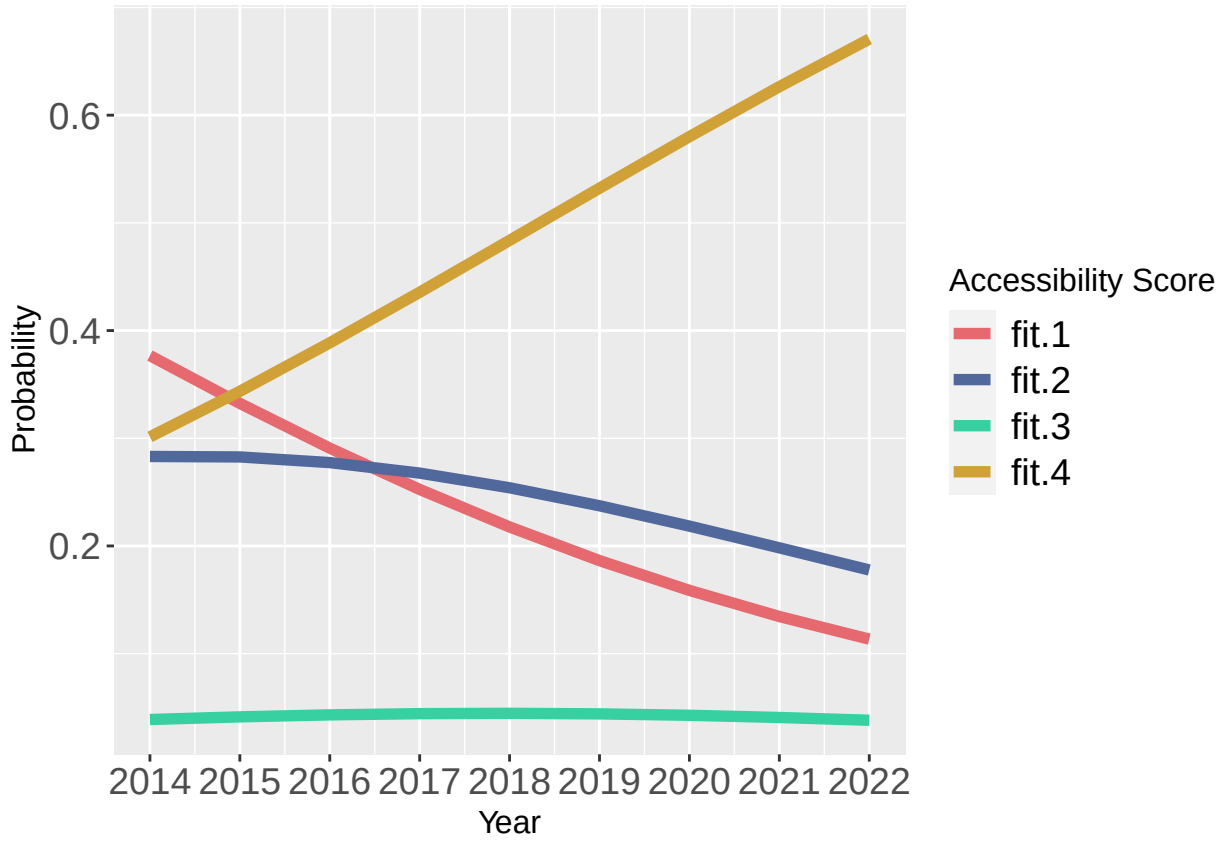

```
autoplot.clm(m1a, what = c("covariate"), x = data$Year) # scale test indicated no variance issues. This
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```r
# N.B. - Interpreting residual plots is largely subjective!

###plotting the model
# Predict probabilities for each level of 'Complete'
new_data <- data.frame(Year = sort(unique(data$Year)))
pred_probs <- predict(m1a, newdata = new_data, type = "prob")

# Add the 'Year' column to the predicted probabilities dataframe
pred_probs_df <- cbind(new_data, as.data.frame(pred_probs))

# Convert the predicted probabilities to a long format for ggplot
library(reshape2)
pred_probs_long <- melt(pred_probs_df, id.vars = 'Year', variable.name = 'AccessLevel', value.name = 'Pr

# Set the colors
my_colors <- c("#E5696F", "#50689B", "#36D0A1", "#D0A136")

# Modify ggplot command
og3 <- ggplot(pred_probs_long, aes(x = Year, y = PredictedProbability, group = AccessLevel, color = Acc
  geom_line(size = 2) +  # Set size of the lines to make them thicker
  scale_color_manual(values = my_colors) +  # Use colors
  scale_x_continuous(breaks = unique(pred_probs_long$Year)) +  # Show all years on x-axis
  labs(x = "Year", y = "Probability", color = "Accessibility Score") +
  theme(
    axis.title = element_text(size = 12), # Increase axis titles
    axis.text = element_text(size = 14),  # Increase axis text
```

```
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  )
print(og3)
```




```
ggsave("og3.png", og3, width = 15, height = 10, units = "in",  bg = "white")
```

```
m1a <- clm(Licence ~ Year, data = data)
```

```
## Warning in x$code == 0L || action == "silent": 'length(x) = 2 > 1' in coercion
## to 'logical(1)'
```

```
## Warning: (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
summary_m1a <- summary(m1a)


# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors
```

```r
# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                 OR      LowerCI       UpperCI    p_value
## 1|2  2.256630e+132 9.929195e+27 5.128692e+236 0.01292492
## 2|3  2.621908e+132 1.149732e+28 5.979135e+236 0.01288181
## 3|4  3.805940e+132 1.653113e+28 8.762366e+236 0.01277571
## Year  1.163746e+00 1.033102e+00  1.310911e+00 0.01255954
```

```r
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Warning in x$code == 0L || action == "silent": 'length(x) = 2 > 1' in coercion
## to 'logical(1)'
```

```
## Warning in !inherits(nfit, "try-error") && nfit$convergence$code >= 0:
## 'length(x) = 2 > 1' in coercion to 'logical(1)'
```

```
## Tests of nominal effects
##
## formula: Licence ~ Year
##        Df  logLik    AIC      LRT Pr(>Chi)
## <none>    -166.02 340.03
## Year    2 -165.98 343.97 0.062447   0.9693
```

```r
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

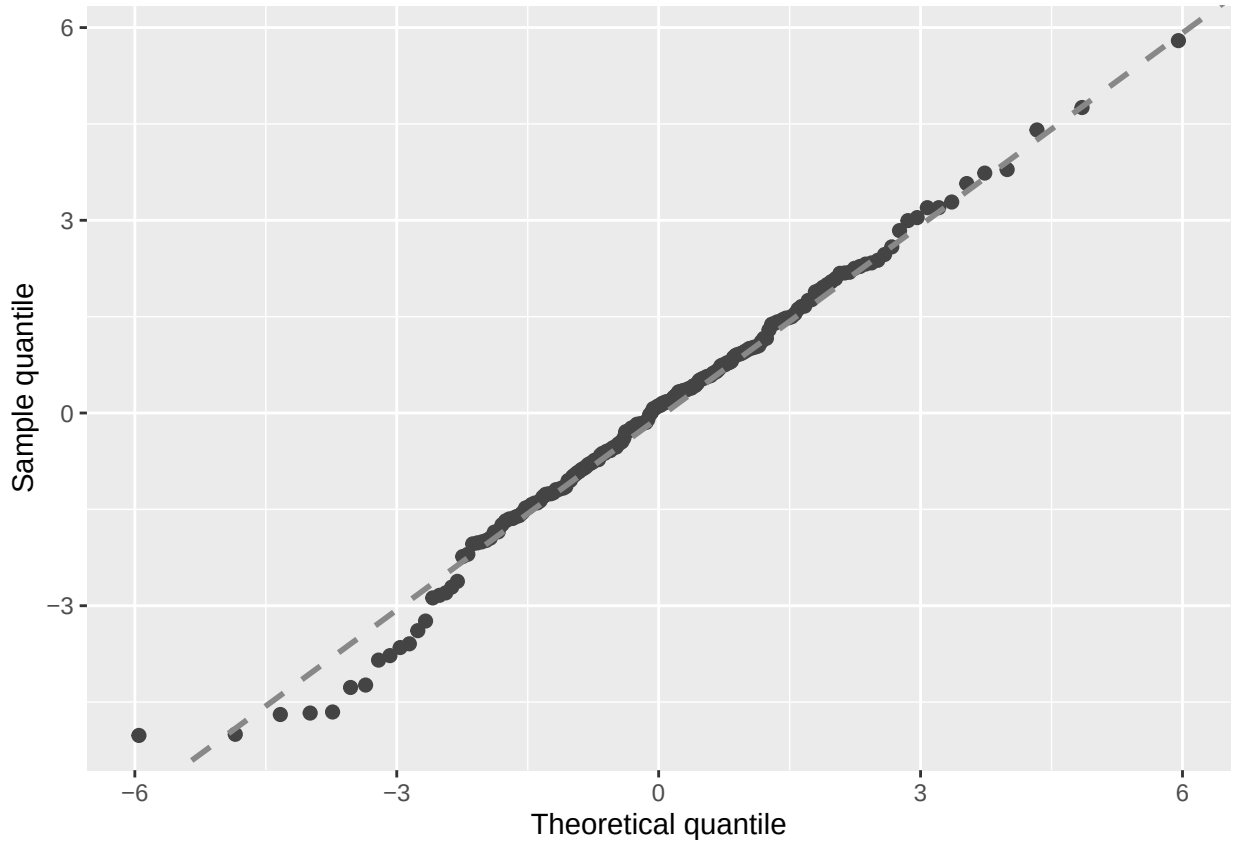
```
## Warning: (-1) Model failed to converge with max|grad| = 41.6269 (tol = 1e-06)
## In addition: iteration limit reached
```

```
## Tests of scale effects
##
## formula: Licence ~ Year
##        Df  logLik    AIC LRT Pr(>Chi)
## <none>    -166.02 340.03
## Year
```

```r
convergence(m1a) # This is another way to assess the model
```

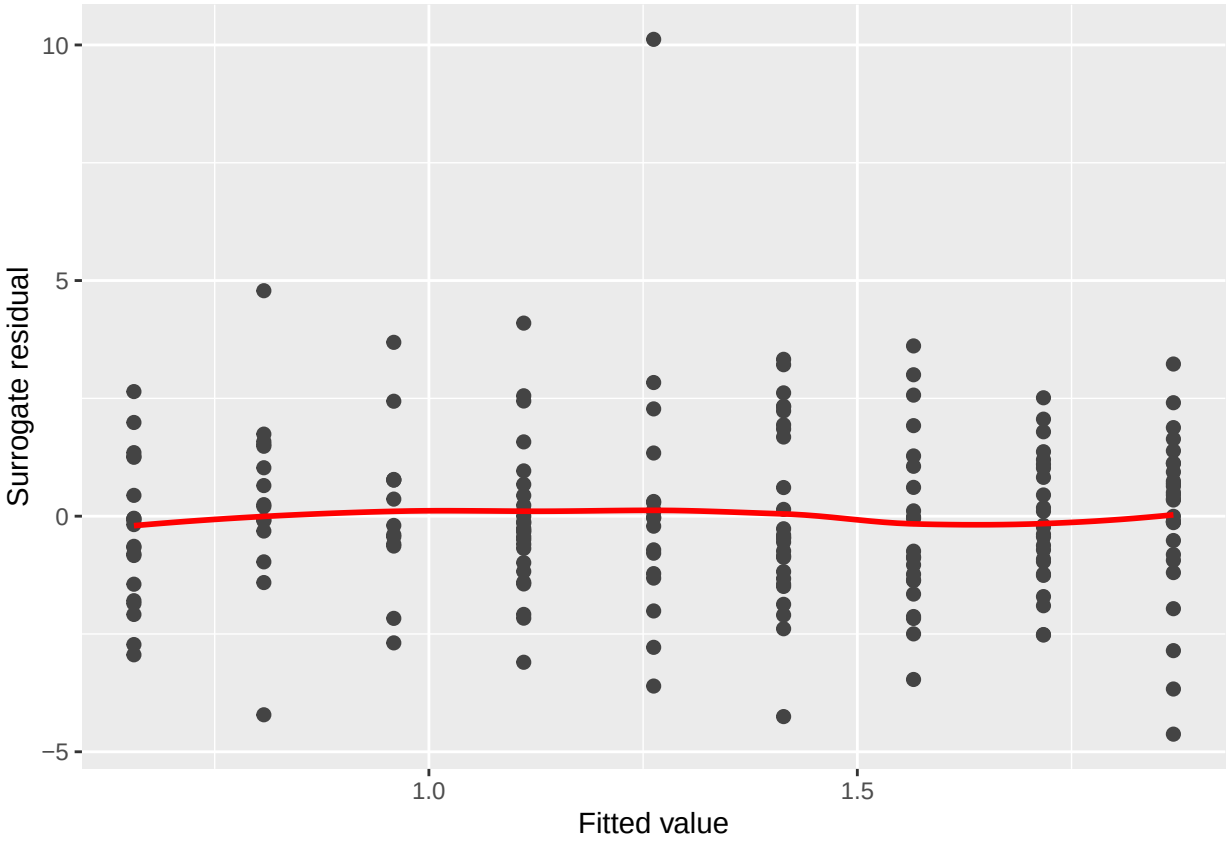
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -166.02 6(2)  1.68e-08 7.7e+12 <1e-10
##
##      Estimate   Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2  304.7551 122.59685  1.30e-09 -3.27e-10       9      12
## 2|3  304.9051 122.59858 -1.30e-09 -3.33e-10       9      12
## 3|4  305.2778 122.60345  7.54e-12 -3.32e-10       9      12
## Year   0.1516   0.06075 -1.68e-08 -1.64e-13      12      12
##
## Eigen values of Hessian:
## 1.709e+08 5.184e+02 1.427e+02 2.219e-05
##
## Convergence message from clm:
## (2) Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## (3) Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



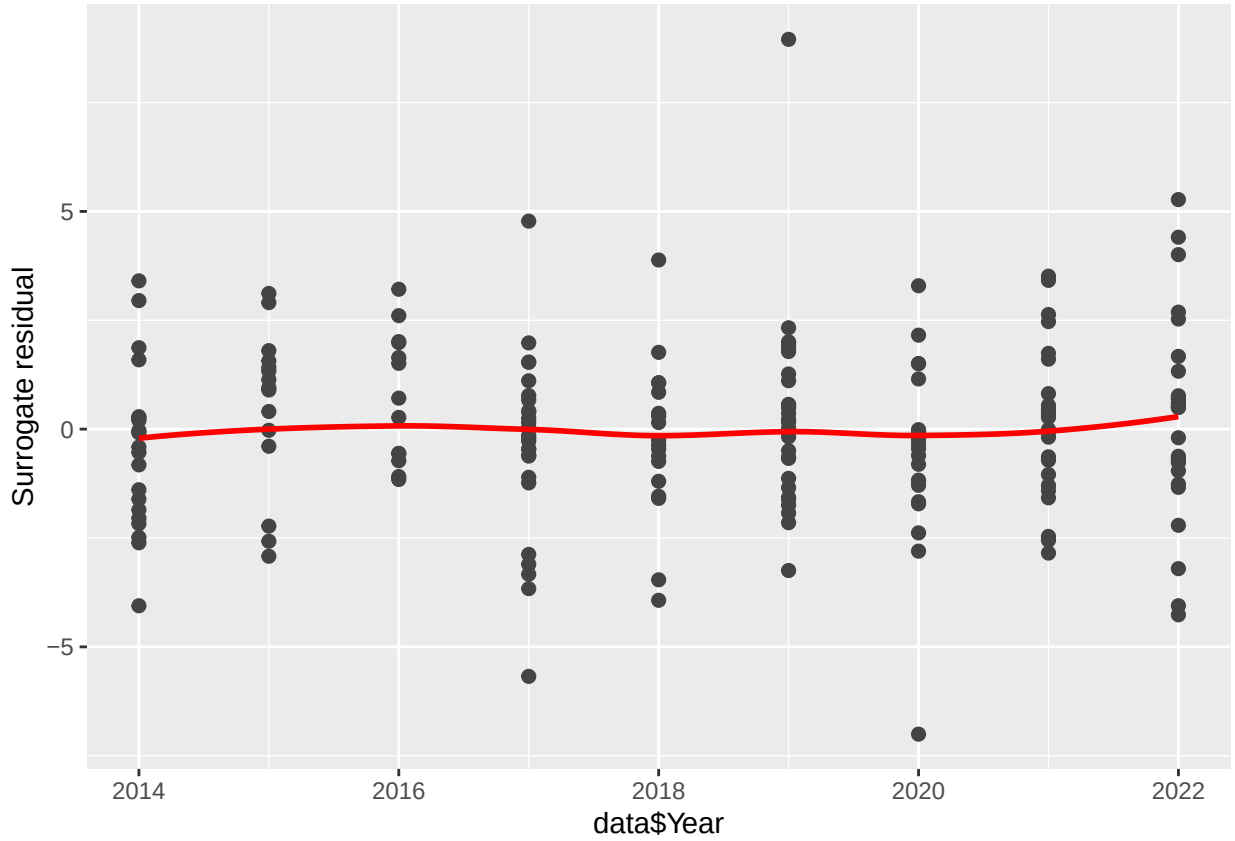

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



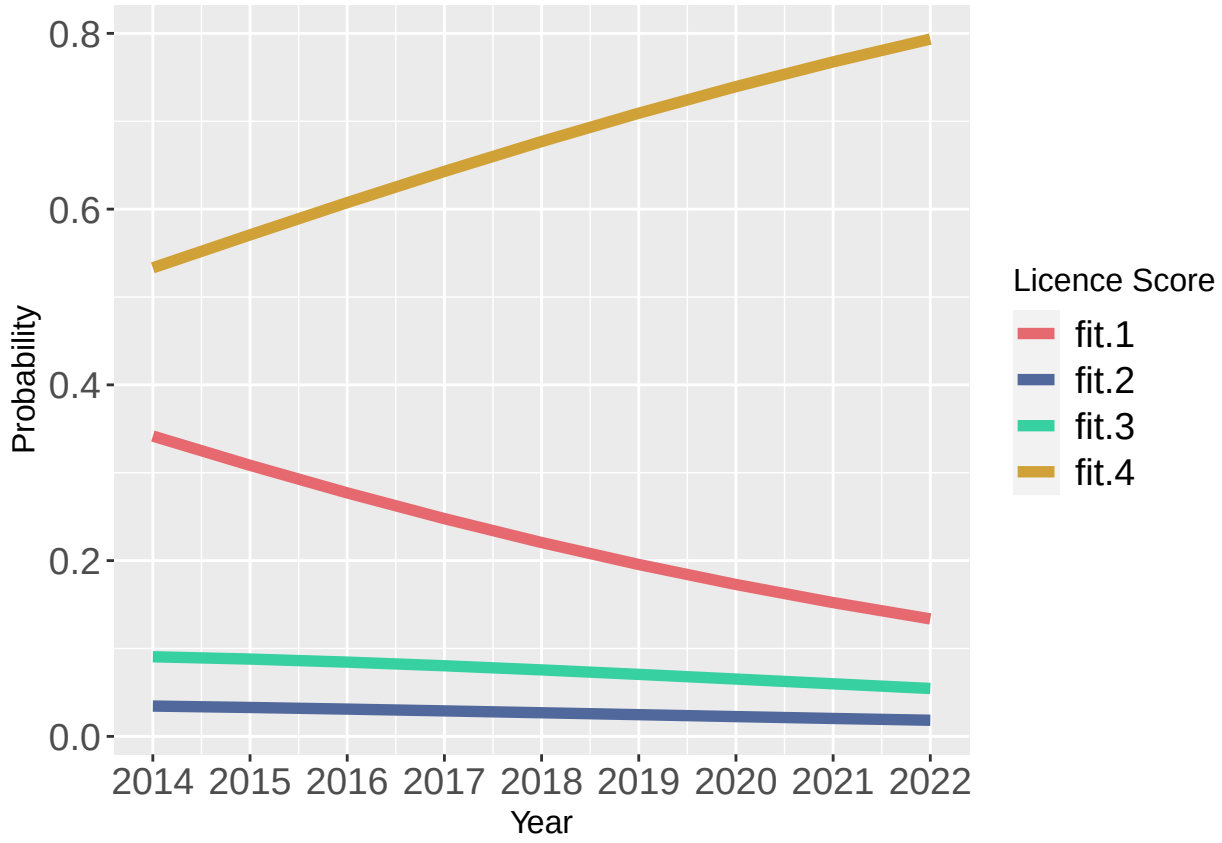

```
autoplot.clm(m1a, what = c("covariate"), x = data$Year) # scale test indicated no variance issues. This
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



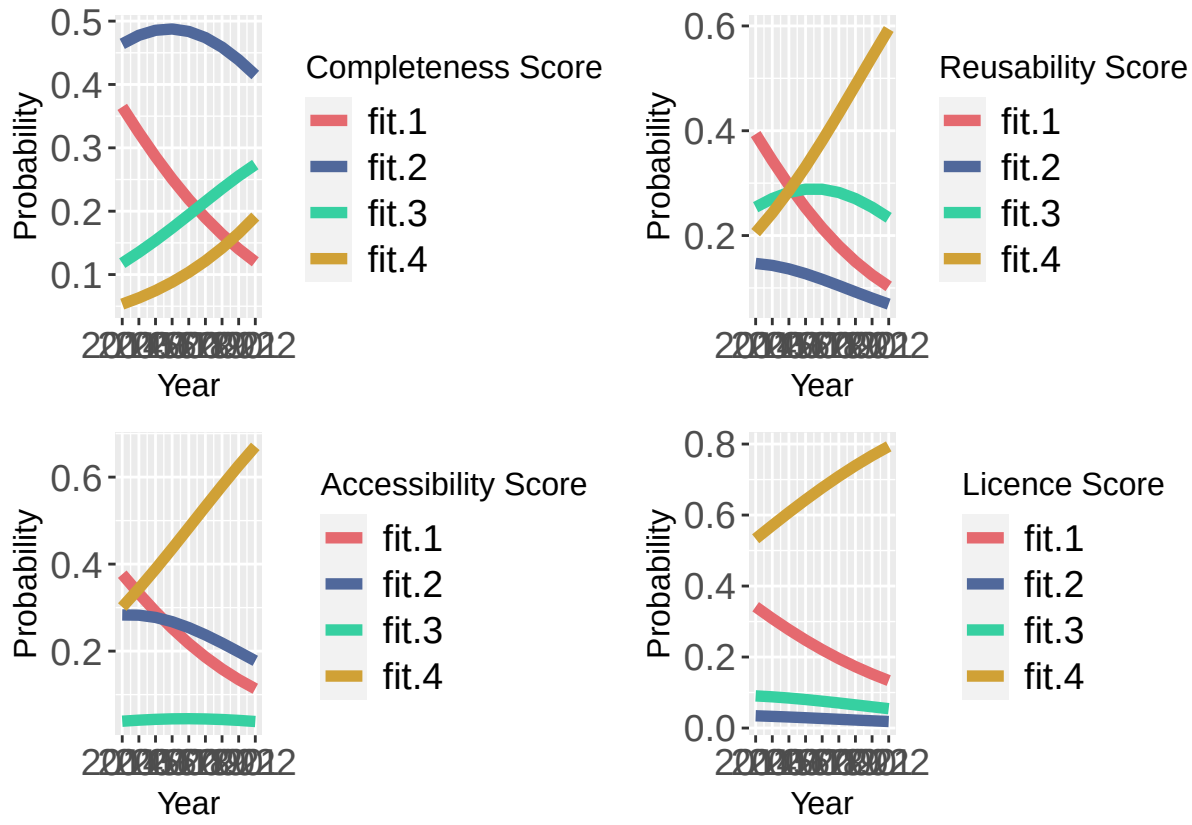

```r
# N.B. - Interpreting residual plots is largely subjective!

###plotting the model
# Predict probabilities for each level of 'Complete'
new_data <- data.frame(Year = sort(unique(data$Year)))
pred_probs <- predict(m1a, newdata = new_data, type = "prob")

# Add the 'Year' column to the predicted probabilities dataframe
pred_probs_df <- cbind(new_data, as.data.frame(pred_probs))

# Convert the predicted probabilities to a long format for ggplot
library(reshape2)
pred_probs_long <- melt(pred_probs_df, id.vars = 'Year', variable.name = 'Licencelevel', value.name = '

# Set the colors
my_colors <- c("#E5696F", "#50689B", "#36D0A1", "#D0A136")

# Modify ggplot command
og4 <- ggplot(pred_probs_long, aes(x = Year, y = PredictedProbability, group = Licencelevel, color = Li
  geom_line(size = 2) +  # Set size of the lines to make them thicker
  scale_color_manual(values = my_colors) +  # Use colors
  scale_x_continuous(breaks = unique(pred_probs_long$Year)) +  # Show all years on x-axis
  labs(x = "Year", y = "Probability", color = "Licence Score") +
  theme(
    axis.title = element_text(size = 12), # Increase axis titles
    axis.text = element_text(size = 14),  # Increase axis text
```

```
    legend.title = element_text(size = 12), # Increase legend title
    legend.text = element_text(size = 14)  # Increase legend text
  )
print(og4)
```




```
ggsave("og4.png", og4, width = 15, height = 10, units = "in",  bg = "white")
```

```
ordinalplot <- og1 + og2 + og3 + og4 +
  plot_layout(
    ncol = 2, heights = c(10, 10), widths = c(10, 10)
  )
print(ordinalplot)
```




```
ggsave("ordinalplot.png", ordinalplot, width = 15, height = 10, units = "in", bg="white")
```

####3

#Ordinal Regression models depending on the sharing projects

```
m1a <- clm(Complete ~ NewDAS + Preprints, data = data)
summary_m1a <- summary(m1a)

# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors

# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                           OR   LowerCI     UpperCI      p_value
## 1|2               0.6532438  0.432356   0.9869818 4.315240e-02
## 2|3              10.0537125  5.590398  18.0804911 1.287859e-14
## 3|4              52.1239514 24.658787 110.1800485 0.000000e+00
## NewDASNot Shared  3.2602625  1.392494   7.6332896 6.472506e-03
## NewDASShared     13.1606441  6.295668  27.5113854 7.357448e-12
## PreprintsYes      2.2372515  1.138340   4.3970105 1.949855e-02
```

```
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Complete ~ NewDAS + Preprints
##           Df  logLik    AIC    LRT Pr(>Chi)
## <none>       -207.28 426.57
## NewDAS     4 -202.41 424.82 9.7501  0.04485 *
## Preprints  2 -207.14 430.28 0.2913  0.86448
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

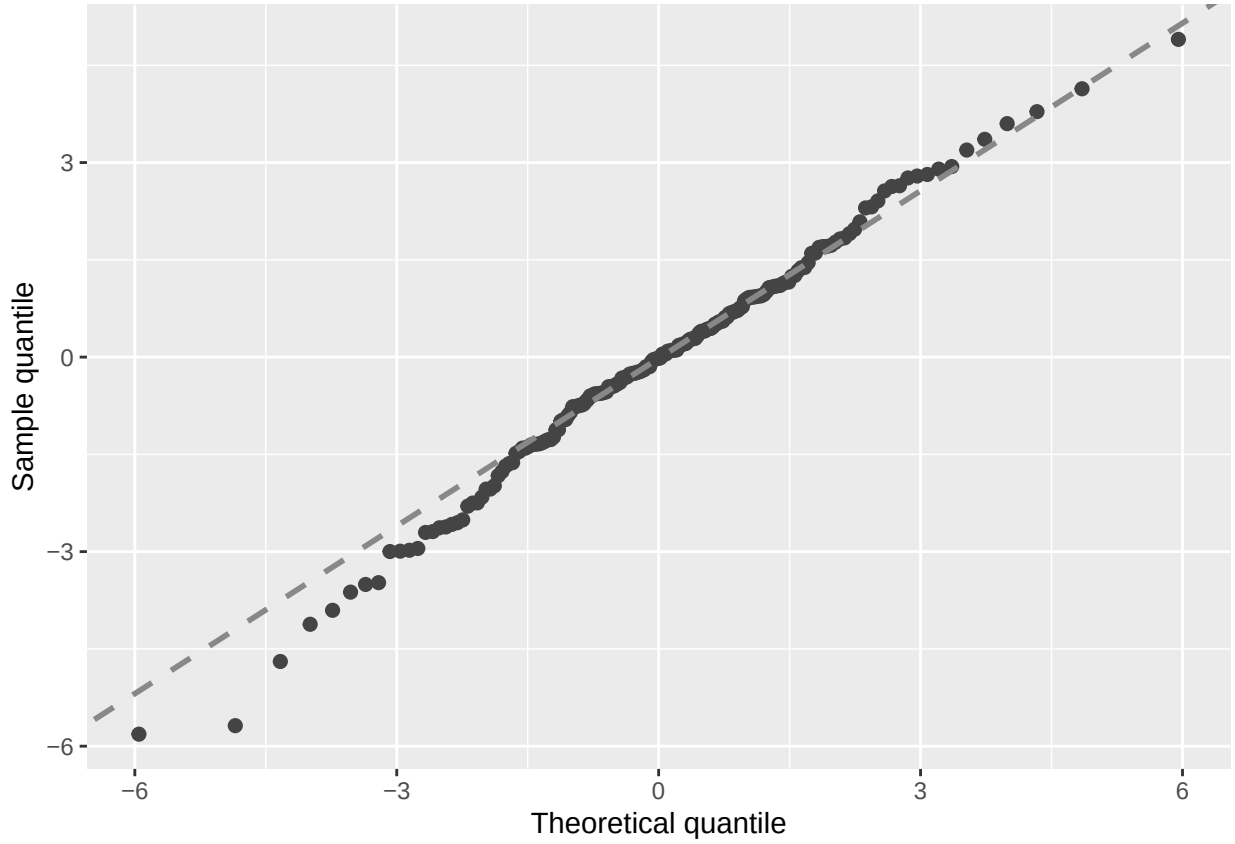
```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

```
## Tests of scale effects
##
## formula: Complete ~ NewDAS + Preprints
##           Df  logLik    AIC    LRT Pr(>Chi)
## <none>       -207.28 426.57
## NewDAS     2 -203.91 423.83 6.7398  0.03439 *
## Preprints  1 -207.28 428.56 0.0031  0.95535
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
convergence(m1a) # This is another way to assess the model
```

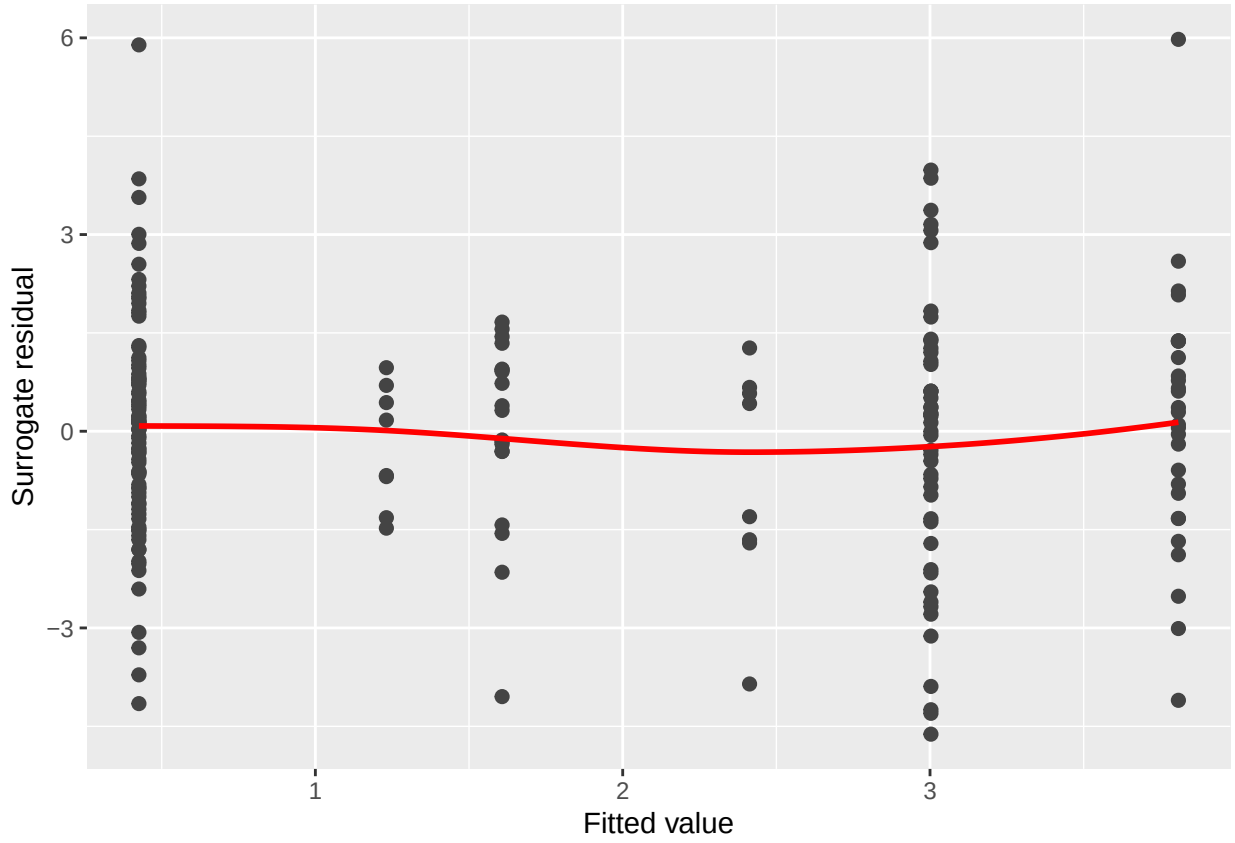
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -207.28 6(0)  3.61e-12 1.9e+01 <1e-10
##
##                   Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2                -0.4258  0.2106  3.09e-12  1.17e-14      13      13
## 2|3                 2.3079  0.2994 -3.61e-12 -2.81e-13      12      13
## 3|4                 3.9536  0.3819 -3.23e-14 -2.81e-13      12      13
## NewDASNot Shared    1.1818  0.4340 -3.21e-13 -2.26e-13      12      13
## NewDASShared        2.5772  0.3762 -3.15e-13 -2.72e-13      12      13
## PreprintsYes        0.8052  0.3447  2.07e-14 -2.27e-14      13      13
##
## Eigen values of Hessian:
## 48.919 34.721 28.580  9.736  6.586  2.565
##
## Convergence message from clm:
## (0) successful convergence
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



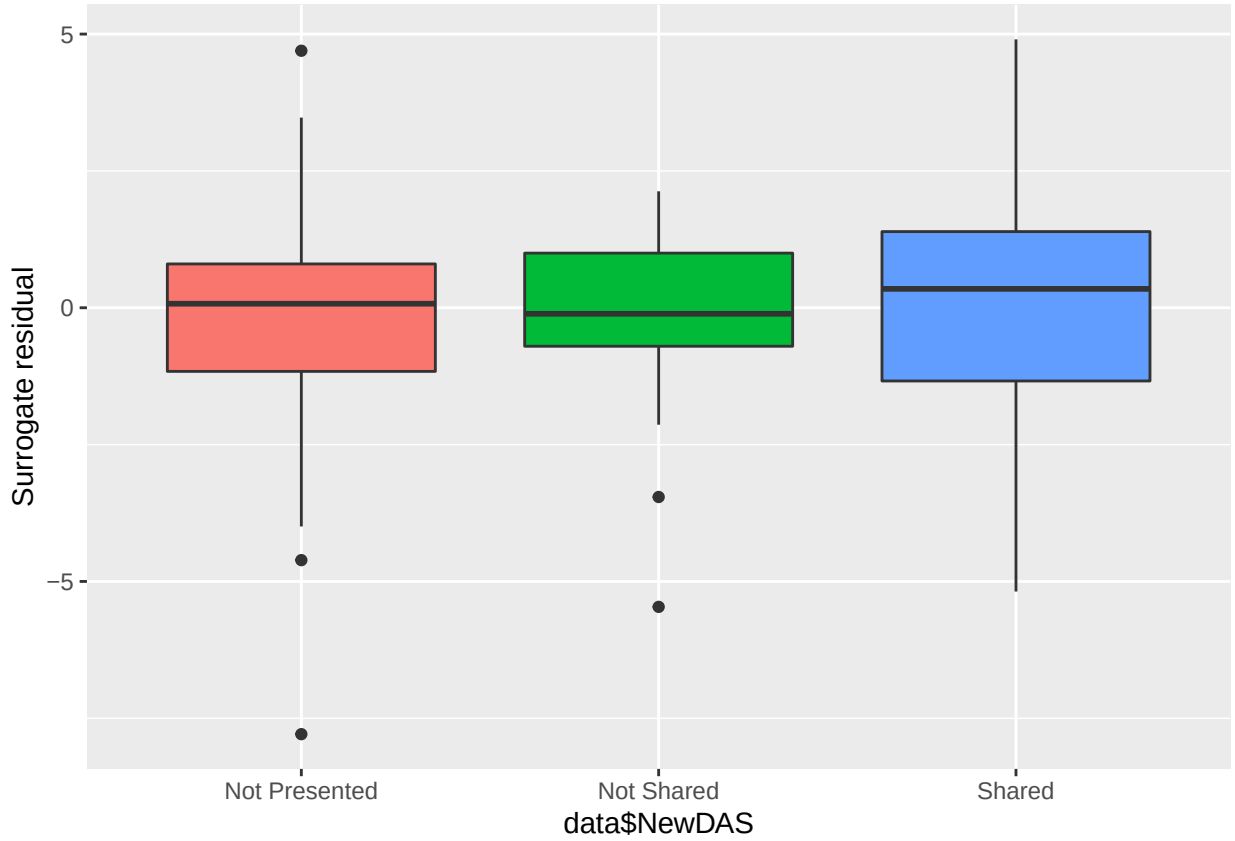

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



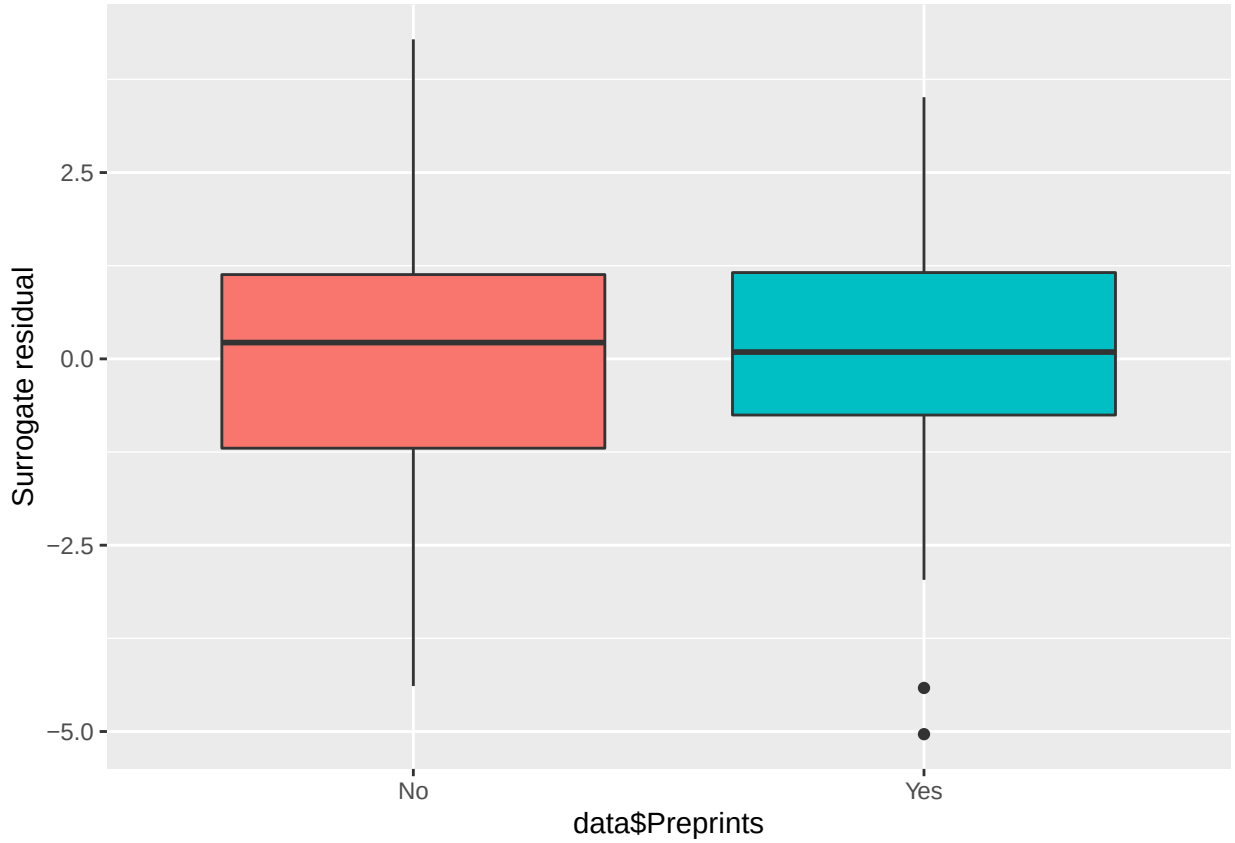

```
autoplot.clm(m1a, what = c("covariate"), x = data$NewDAS) # scale test indicated no variance issues. Th
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
autoplot.clm(m1a, what = c("covariate"), x = data$Preprints)
```

```
## Warning: ‘guides(<scale> = FALSE)‘ is deprecated. Please use ‘guides(<scale> =
## "none")‘ instead.
```




```
# N.B. - Interpreting residual plots is largely subjective!
```

```
m1a <- clm(Reuse ~ NewDAS + Preprints, data = data)
summary_m1a <- summary(m1a)

# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors

# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                       OR   LowerCI    UpperCI      p_value
## 1|2             0.5851276 0.3875115  0.8835203 1.080242e-02
```

```
## 2|3              1.1384211 0.7593092  1.7068181 5.303823e-01
## 3|4              4.4224283 2.7525267  7.1054250 7.981718e-10
## NewDASNot Shared 5.5189574 2.3571077 12.9221468 8.306053e-05
## NewDASShared     6.7456538 3.5058381 12.9794487 1.085746e-08
## PreprintsYes     1.0835906 0.5520896  2.1267718 8.154950e-01
```

```
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Reuse ~ NewDAS + Preprints
##           Df  logLik    AIC    LRT Pr(>Chi)
## <none>       -226.59 465.19
## NewDAS     4 -223.98 467.96 5.2220  0.26527
## Preprints  2 -222.23 460.46 8.7302  0.01271 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

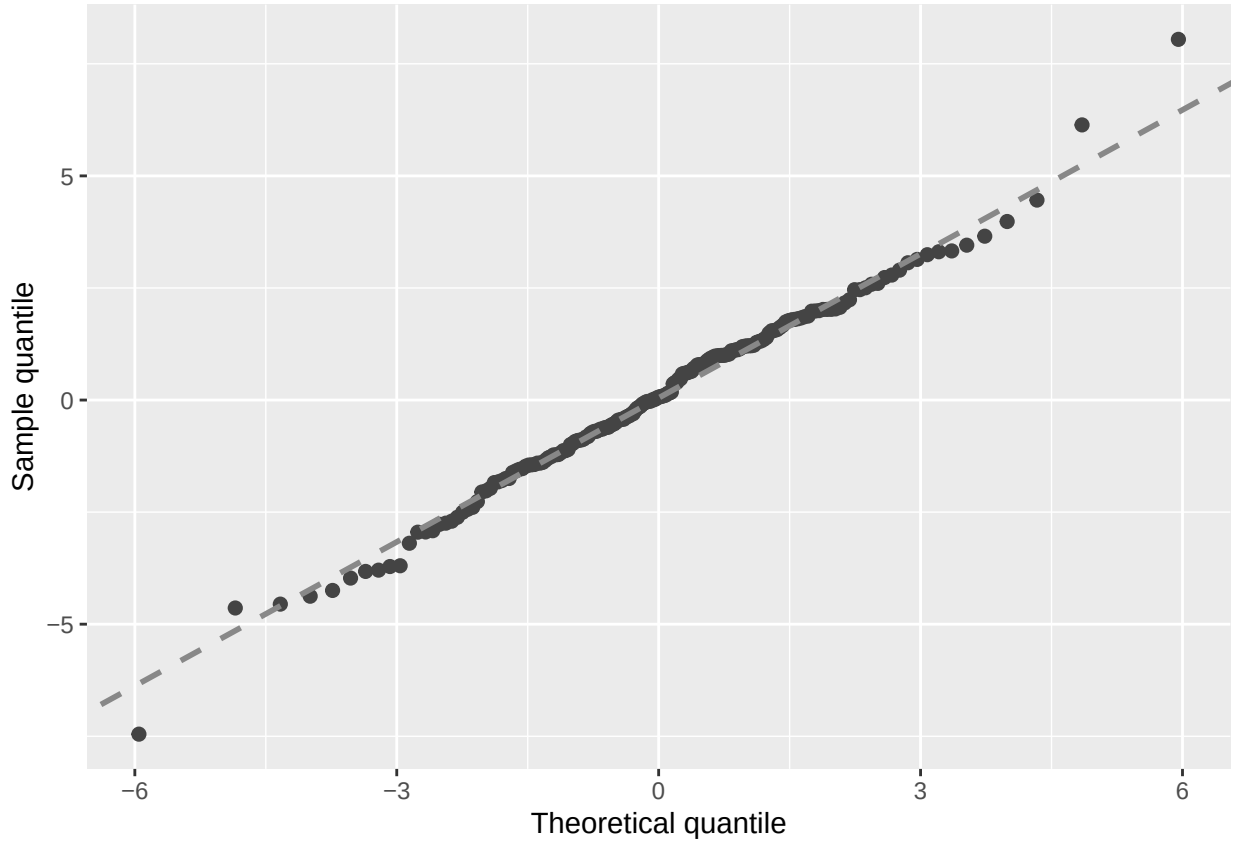
```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

```
## Tests of scale effects
##
## formula: Reuse ~ NewDAS + Preprints
##           Df  logLik    AIC     LRT Pr(>Chi)
## <none>       -226.59 465.19
## NewDAS     2 -226.39 468.78 0.41107   0.8142
## Preprints  1 -225.55 465.11 2.07927   0.1493
```

```
convergence(m1a) # This is another way to assess the model
```

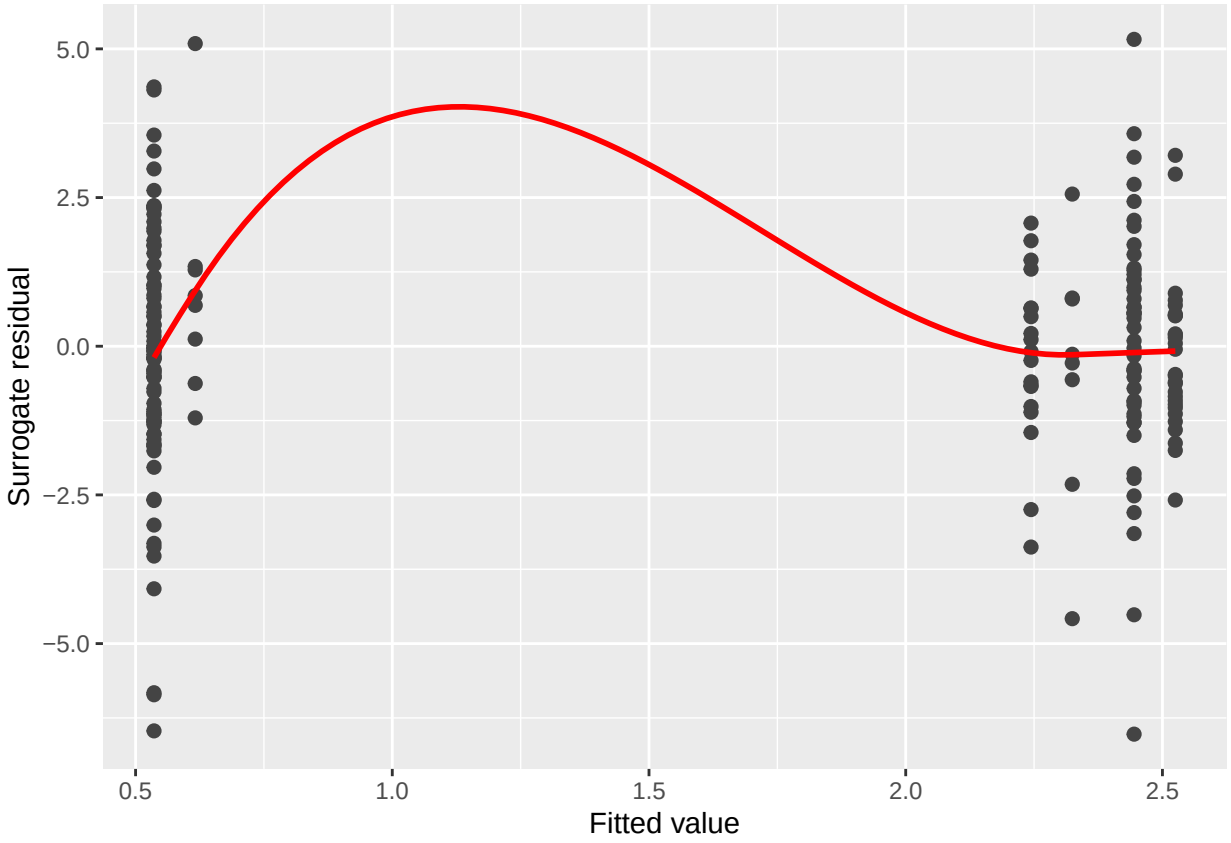
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -226.59 6(0)  1.80e-13 3.3e+01 <1e-10
##
##                  Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2              -0.53593  0.2102  1.80e-13  2.64e-15      14      14
## 2|3               0.12964  0.2066 -1.56e-13 -5.47e-16      14      14
## 3|4               1.48669  0.2419  6.55e-15 -3.82e-16      15      16
## NewDASNot Shared  1.70819  0.4341 -9.99e-16 -5.94e-16      14      15
## NewDASShared      1.90890  0.3339 -7.88e-15 -1.01e-15      14      15
## PreprintsYes      0.08028  0.3440  2.89e-15  7.09e-16      14      13
##
## Eigen values of Hessian:
## 123.454  60.436  27.013  10.229   7.437   3.706
##
## Convergence message from clm:
## (0) successful convergence
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



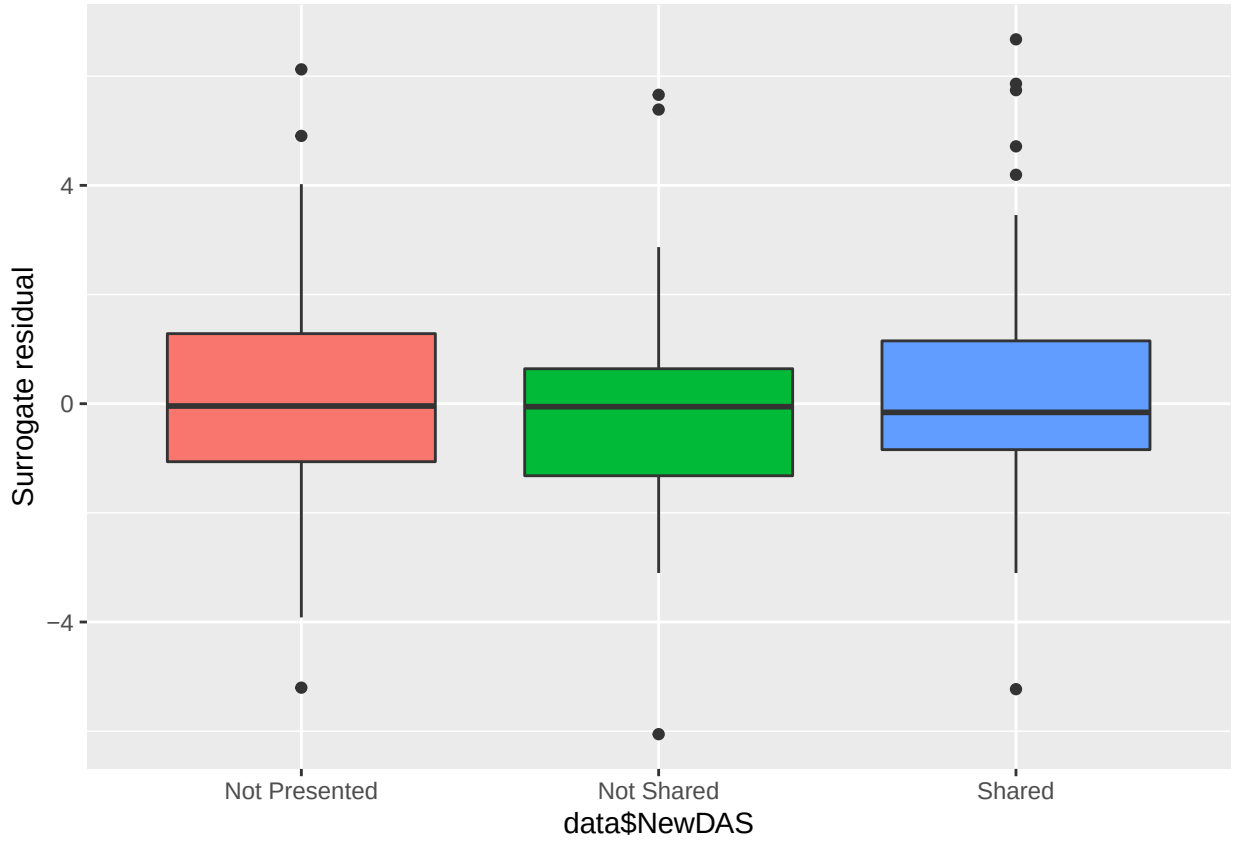

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



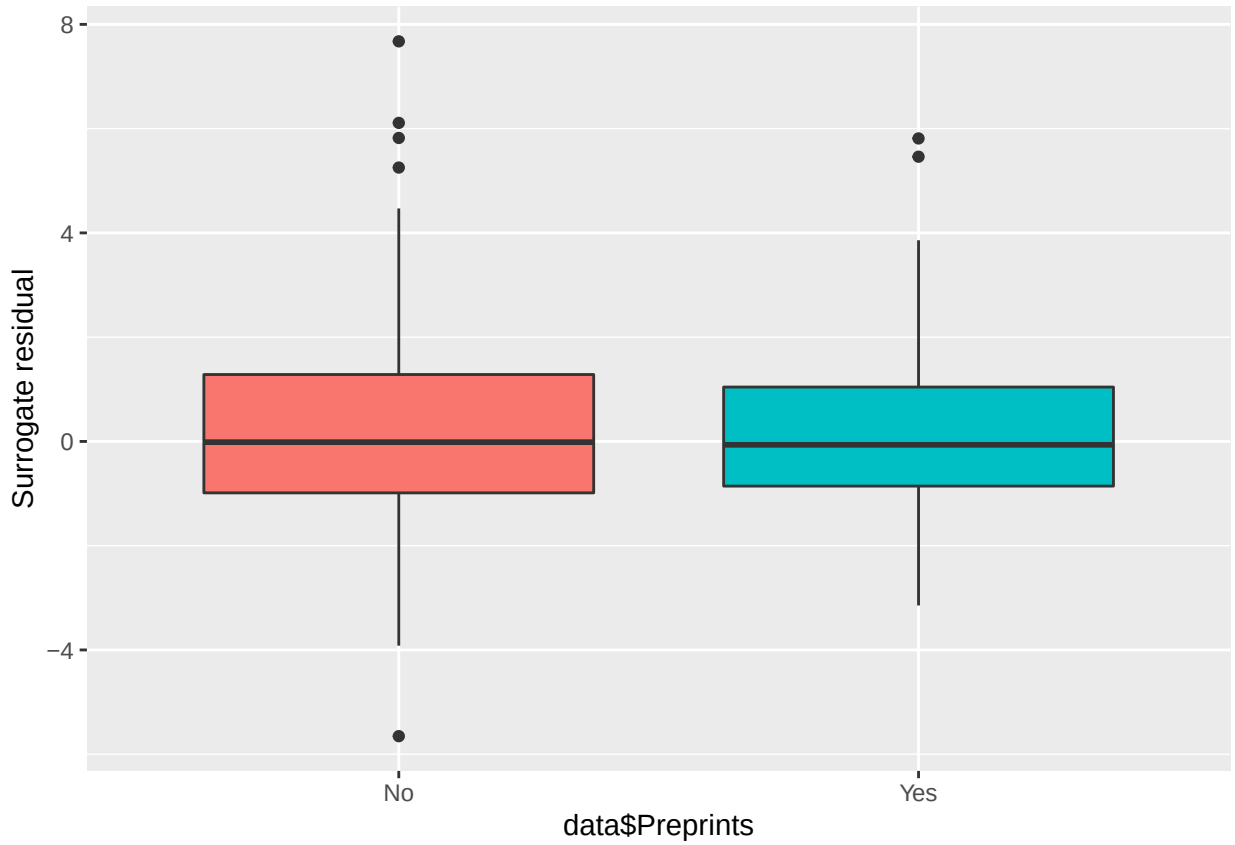

```
autoplot.clm(m1a, what = c("covariate"), x = data$NewDAS) # scale test indicated no variance issues. Th
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
autoplot.clm(m1a, what = c("covariate"), x = data$Preprints)
```

```
## Warning: ‘guides(<scale> = FALSE)‘ is deprecated. Please use ‘guides(<scale> =
## "none")‘ instead.
```




```
# N.B. - Interpreting residual plots is largely subjective!
```

```
m1a <- clm(Access ~ NewDAS + Preprints, data = data)
summary_m1a <- summary(m1a)

# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors

# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                       OR   LowerCI    UpperCI      p_value
## 1|2             0.6034796 0.3984970  0.9139029 1.707011e-02
```

```
## 2|3              2.3011558 1.4917473  3.5497420 1.642623e-04
## 3|4              2.8320314 1.8164013  4.4155452 4.349083e-06
## NewDASNot Shared 5.0164438 2.0473496 12.2913588 4.200776e-04
## NewDASShared     5.8615538 3.0241489 11.3611511 1.627745e-07
## PreprintsYes     1.6652166 0.7931505  3.4961165 1.777868e-01
```

```
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Access ~ NewDAS + Preprints
##           Df  logLik    AIC    LRT Pr(>Chi)
## <none>       -200.61 413.22
## NewDAS     4 -197.91 415.82 5.4031   0.2484
## Preprints  2 -199.89 415.78 1.4450   0.4855
```

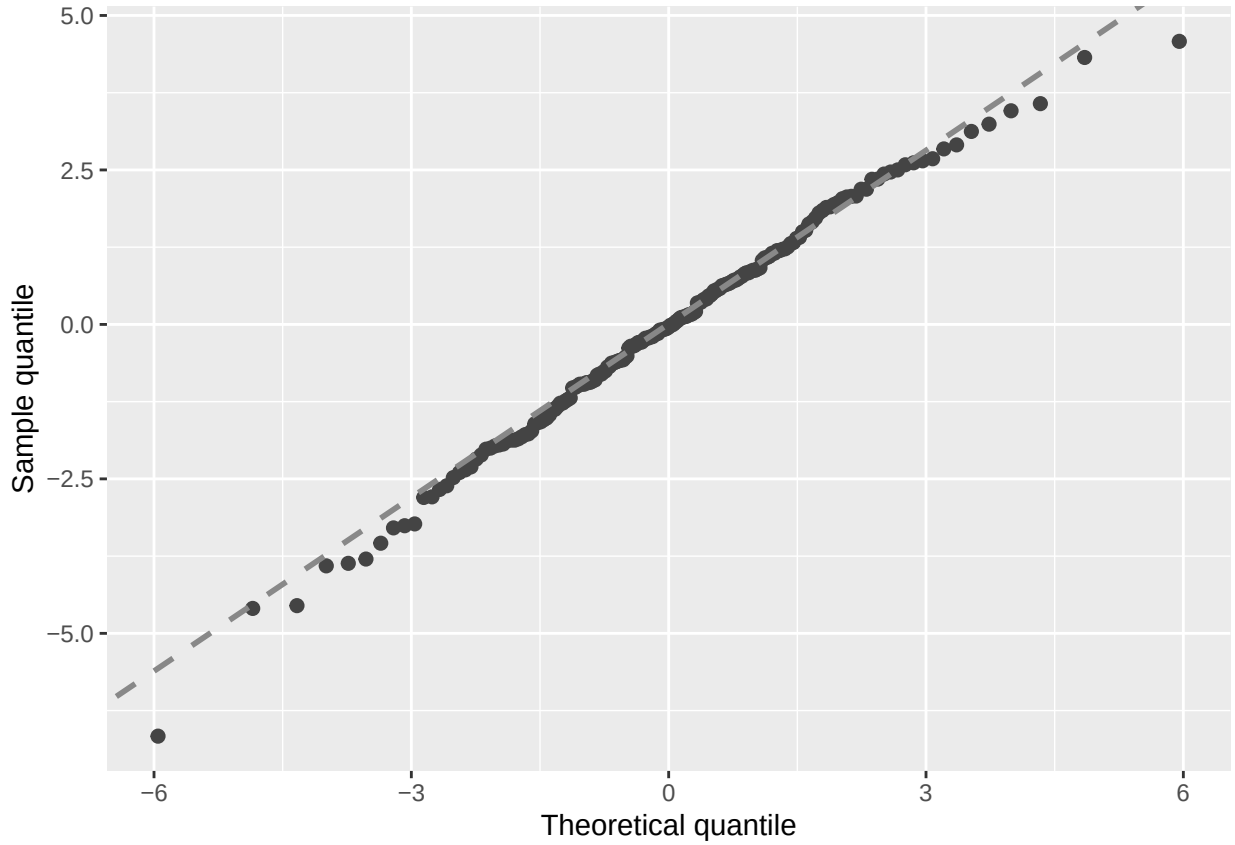
```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

```
## Tests of scale effects
##
## formula: Access ~ NewDAS + Preprints
##           Df  logLik    AIC     LRT Pr(>Chi)
## <none>       -200.61 413.22
## NewDAS     2 -200.39 416.78 0.44455   0.8007
## Preprints  1 -200.24 414.47 0.75425   0.3851
```

```
convergence(m1a) # This is another way to assess the model
```

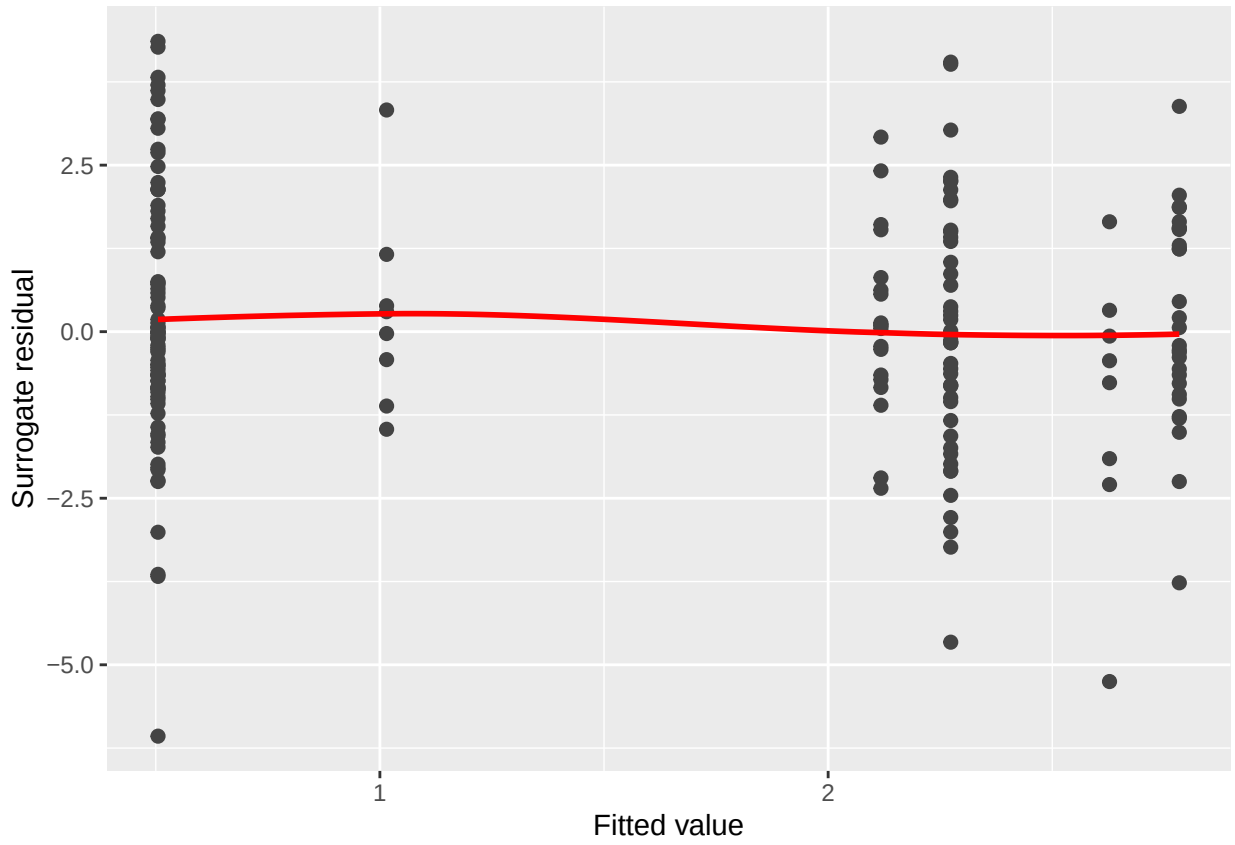
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -200.61 6(2)  2.14e-12 1.1e+02 <1e-10
##
##                  Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2               -0.5050  0.2117  5.49e-13 -1.28e-14      13      13
## 2|3                0.8334  0.2212  1.39e-13 -2.88e-14      13      13
## 3|4                1.0410  0.2266  1.61e-12 -2.95e-14      13      14
## NewDASNot Shared   1.6127  0.4572 -3.03e-13 -2.78e-14      13      14
## NewDASShared       1.7684  0.3376 -1.99e-12 -8.64e-14      12      13
## PreprintsYes       0.5100  0.3784 -2.14e-12 -2.06e-13      12      12
##
## Eigen values of Hessian:
## 398.229  50.296  27.709   8.866   6.325   3.547
##
## Convergence message from clm:
## (0) successful convergence
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



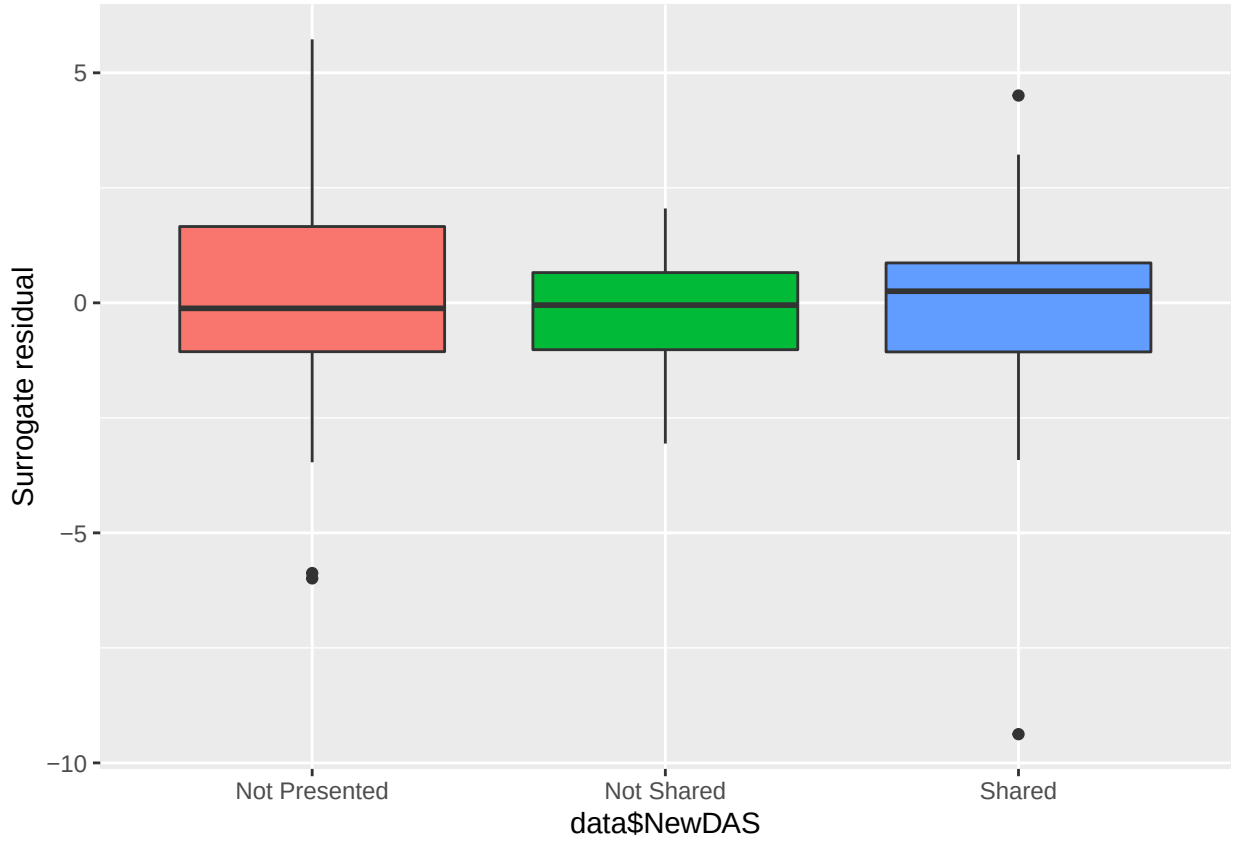

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



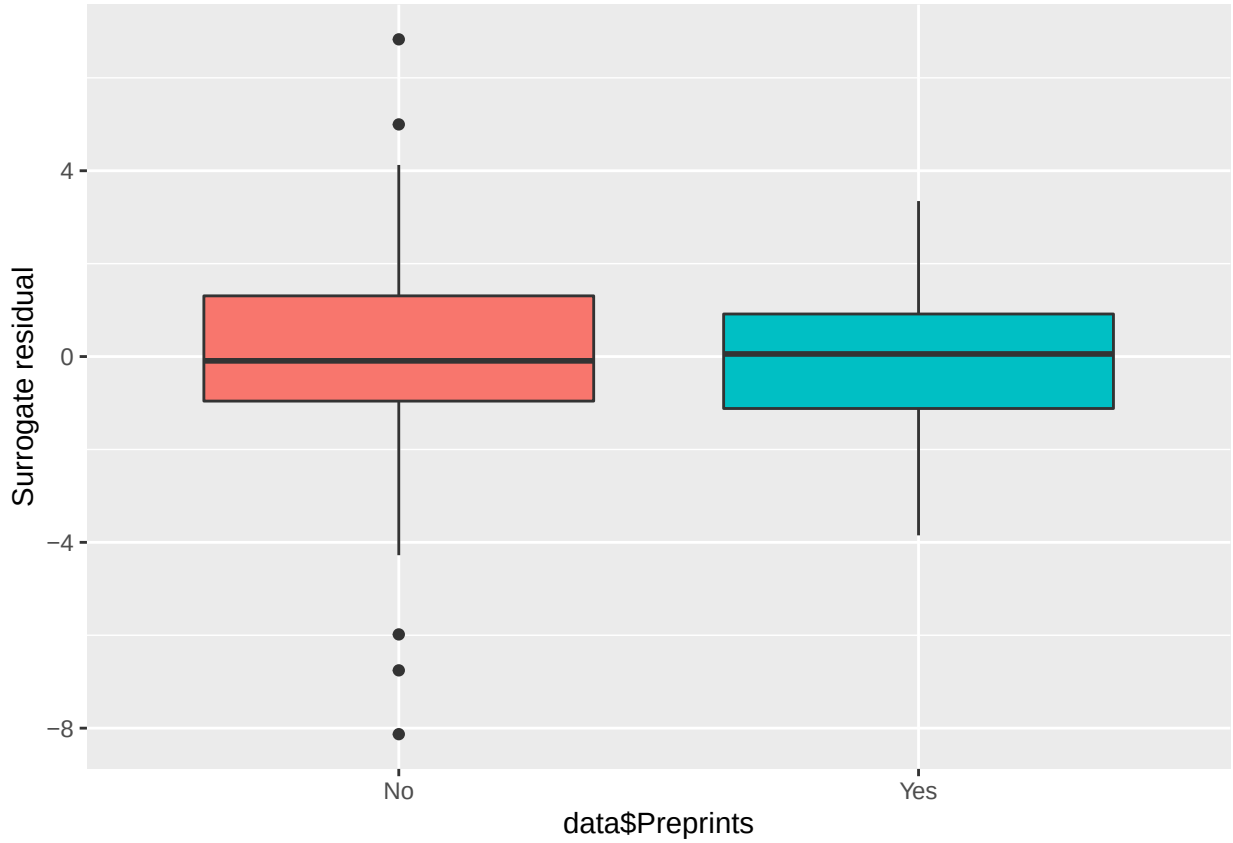

```
autoplot.clm(m1a, what = c("covariate"), x = data$NewDAS) # scale test indicated no variance issues. Th
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
autoplot.clm(m1a, what = c("covariate"), x = data$Preprints)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
# N.B. - Interpreting residual plots is largely subjective!
```

```
m1a <- clm(Licence ~ NewDAS + Preprints, data = data)
summary_m1a <- summary(m1a)

# Extract coefficients and standard errors
coefs <- summary_m1a$coefficients[, 1]  # Coefficients
se_coefs <- summary_m1a$coefficients[, 2]  # Standard errors

# Extract coefficients and standard errors
coefs <- summary(m1a)$coefficients
OR <- exp(coefs[,1])
lower_ci <- exp(coefs[,1] - 1.96 * coefs[,2])
upper_ci <- exp(coefs[,1] + 1.96 * coefs[,2])

# Calculate z-values and p-values
z_values <- coefs[,1] / coefs[,2]
p_values <- 2 * (1 - pnorm(abs(z_values)))

# Combine everything into a data frame for easy viewing
results <- data.frame(OR = OR, LowerCI = lower_ci, UpperCI = upper_ci, p_value = p_values)
print(results)
```

```
##                       OR   LowerCI    UpperCI      p_value
## 1|2             0.6287768 0.4134465  0.9562549 3.007374e-02
```

```
## 2|3              0.7442143 0.4924399  1.1247157 1.608644e-01
## 3|4              1.1535025 0.7648622  1.7396180 4.957229e-01
## NewDASNot Shared 5.3128274 1.6908919 16.6930451 4.246679e-03
## NewDASShared     6.3953648 2.8022381 14.5957227 1.045395e-05
## PreprintsYes     2.3785527 0.8260296  6.8490444 1.083171e-01
```

```
#Check the Assumptions
nominal_test(m1a) # This is a test of the proportional odds assumption, odds are proportional in this c
```

```
## Tests of nominal effects
##
## formula: Licence ~ NewDAS + Preprints
##           Df  logLik    AIC LRT Pr(>Chi)
## <none>       -149.58 311.16
## NewDAS
## Preprints
```

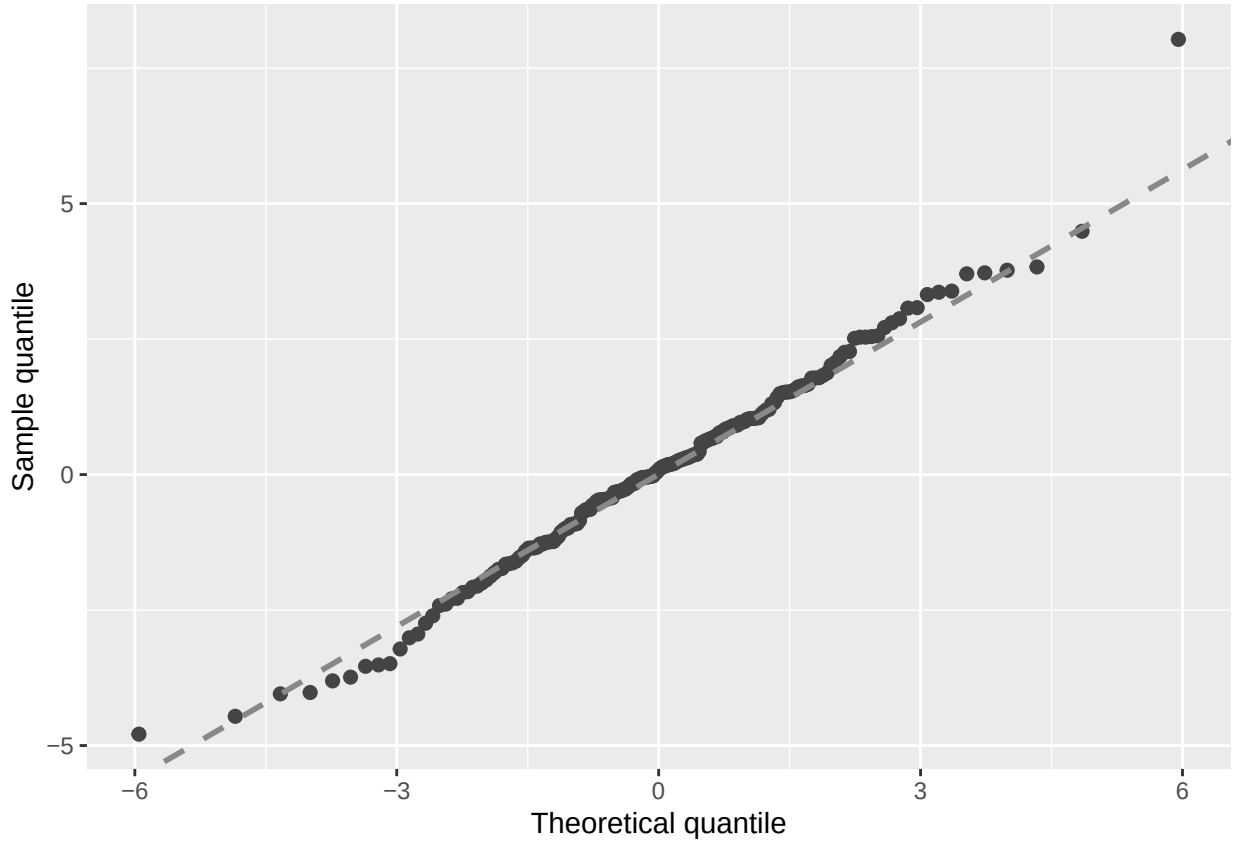
```
scale_test(m1a) # scale test checks for equal variance/ scale across year, assumptions are not violated
```

```
## Tests of scale effects
##
## formula: Licence ~ NewDAS + Preprints
##           Df  logLik    AIC     LRT Pr(>Chi)
## <none>       -149.58 311.16
## NewDAS     2 -148.95 313.91 1.25533   0.5338
## Preprints  1 -149.10 312.21 0.95565   0.3283
```

```
convergence(m1a) # This is another way to assess the model
```

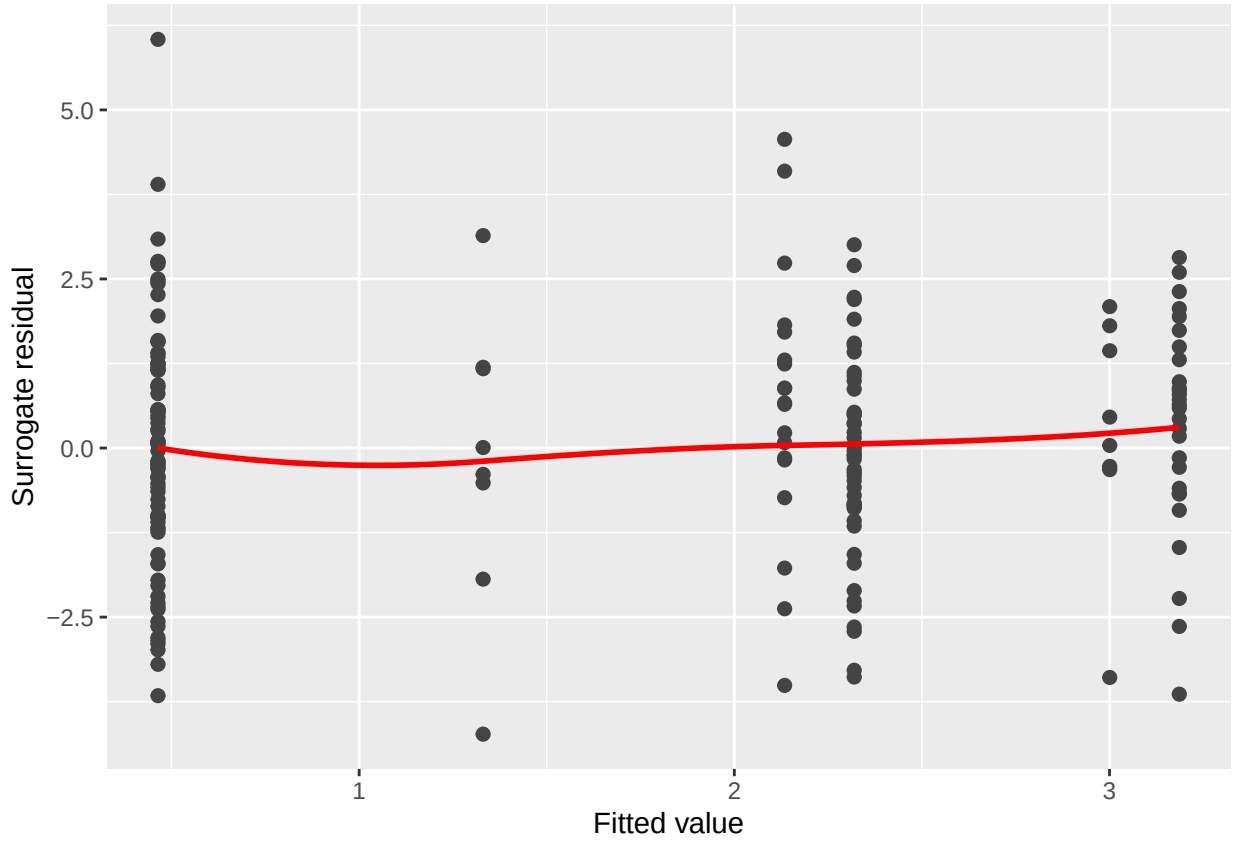
```
##  nobs logLik  niter max.grad cond.H  logLik.Error
##  193  -149.58 7(2)  5.84e-12 1.6e+02 <1e-10
##
##                  Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
## 1|2               -0.4640  0.2139  4.50e-13 -4.50e-14      13      13
## 2|3               -0.2954  0.2107  1.58e-13 -4.58e-14      13      13
## 3|4                0.1428  0.2096  5.71e-12 -5.06e-14      12      12
## NewDASNot Shared   1.6701  0.5841 -7.88e-13 -5.52e-14      12      13
## NewDASShared       1.8556  0.4210 -4.57e-12 -3.21e-13      12      13
## PreprintsYes       0.8665  0.5396 -5.84e-12 -1.33e-12      11      11
##
## Eigen values of Hessian:
## 406.815 113.328  15.099   5.486   3.398   2.567
##
## Convergence message from clm:
## (0) successful convergence
## In addition: Absolute and relative convergence criteria were met
```

```
######## Graphically validate proportional odds using the sure package #######

autoplot.clm(m1a, what = c("qq")) # most of the points fall along the line, so no violation of linearit
```



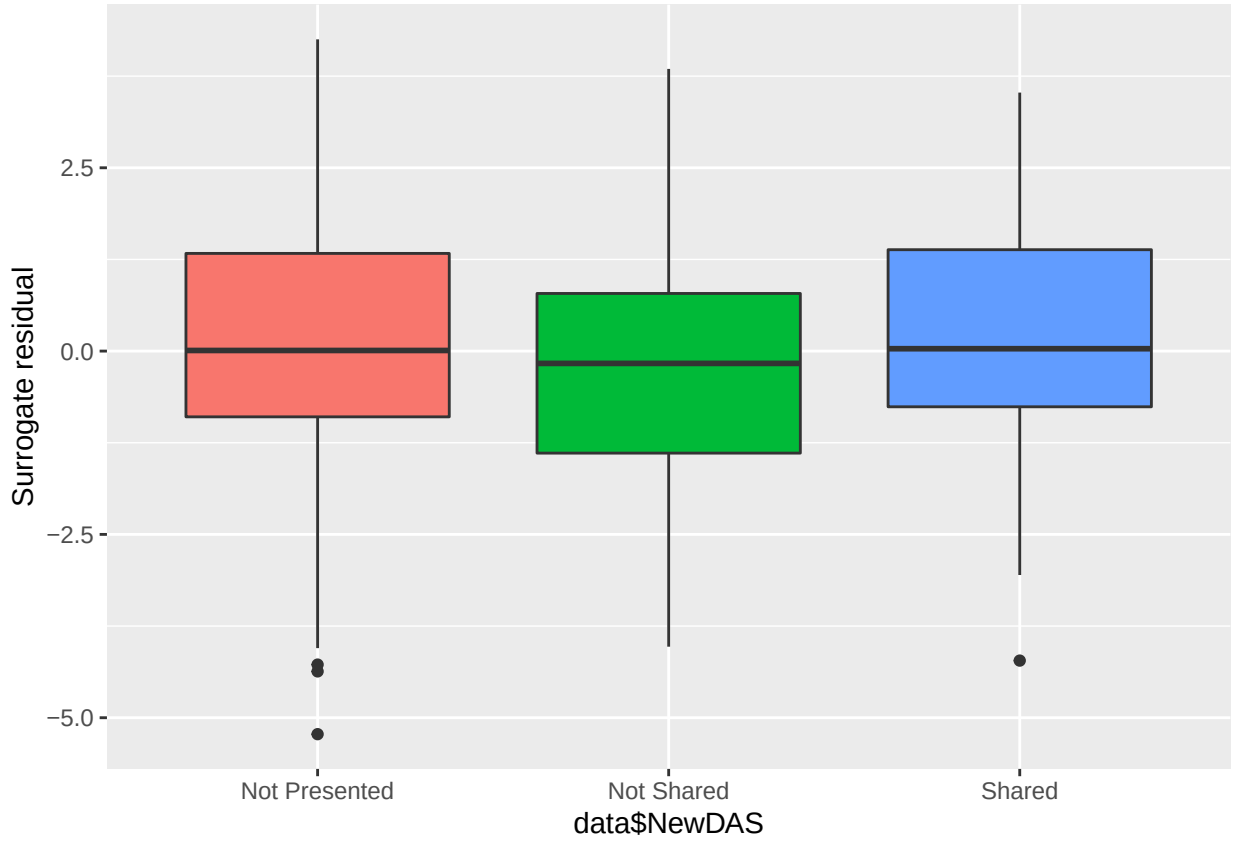

```
autoplot.clm(m1a, what = c("fitted")) # the residuals do not show a clear pattern or trend
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



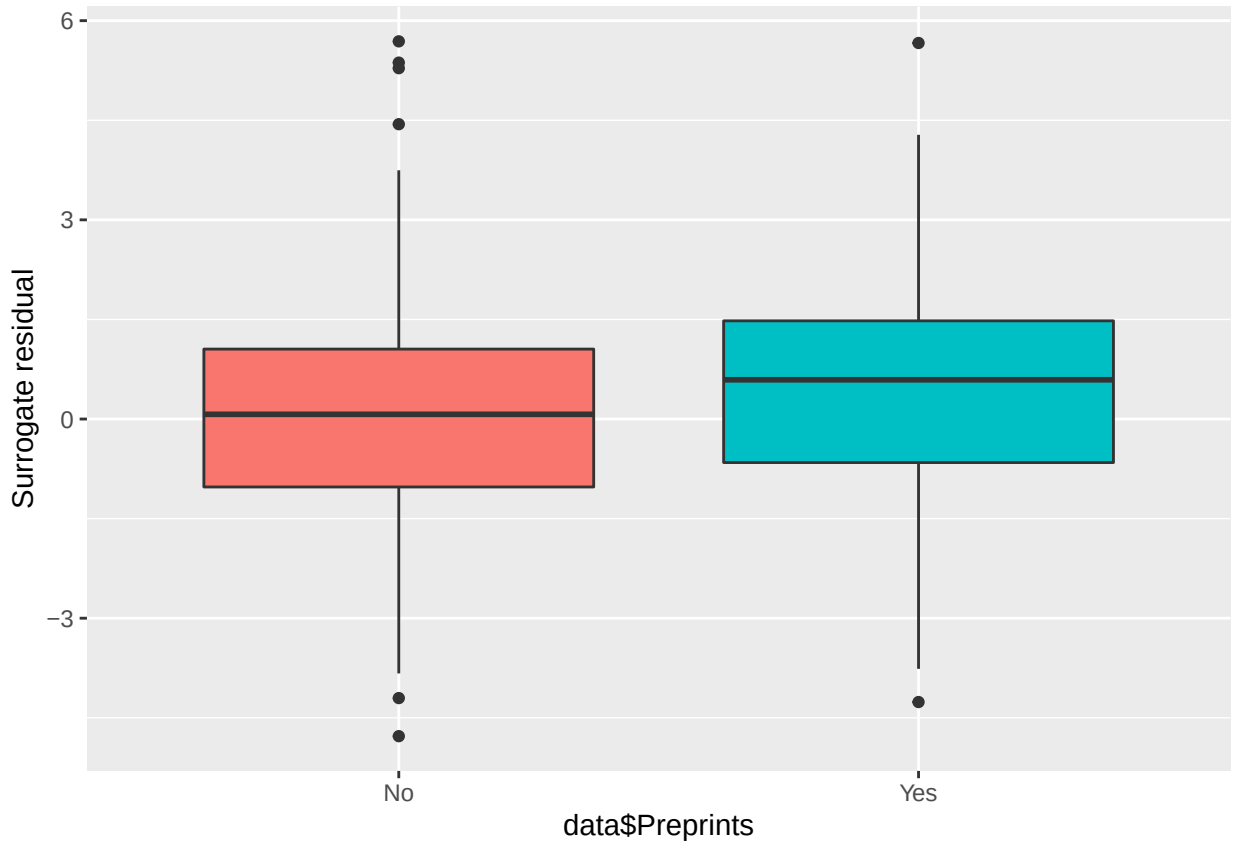

```
autoplot.clm(m1a, what = c("covariate"), x = data$NewDAS) # scale test indicated no variance issues. Th
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
autoplot.clm(m1a, what = c("covariate"), x = data$Preprints)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

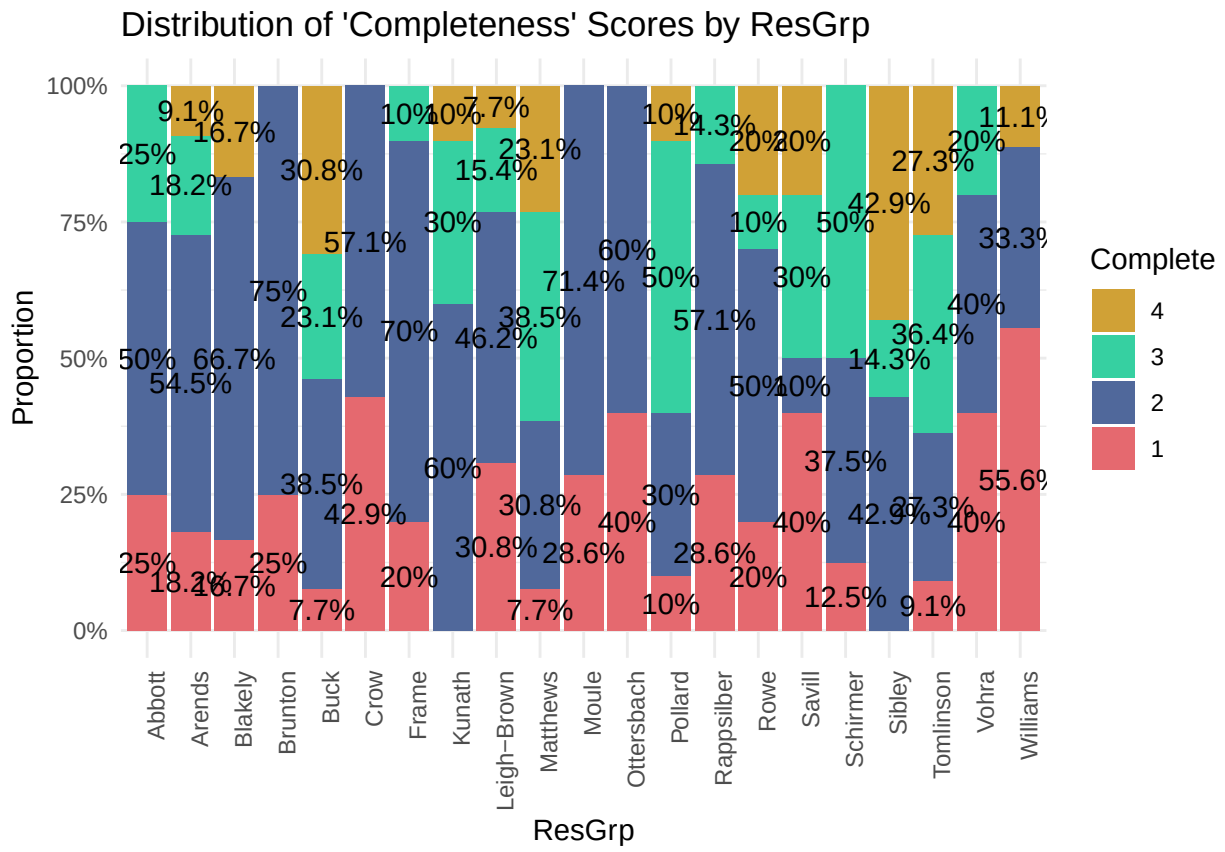



```
# N.B. - Interpreting residual plots is largely subjective!
```

#Research Group figures (Internal)

```
# Calculate the frequency and percentage for each 'Complete' score among the research Group
long_data <- data %>%
  count(ResGrp, Complete) %>%
  group_by(ResGrp) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Complete = factor(Complete, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
internal1 <- ggplot(long_data, aes(x = as.factor(ResGrp), y = n, fill = as.factor(Complete))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
```

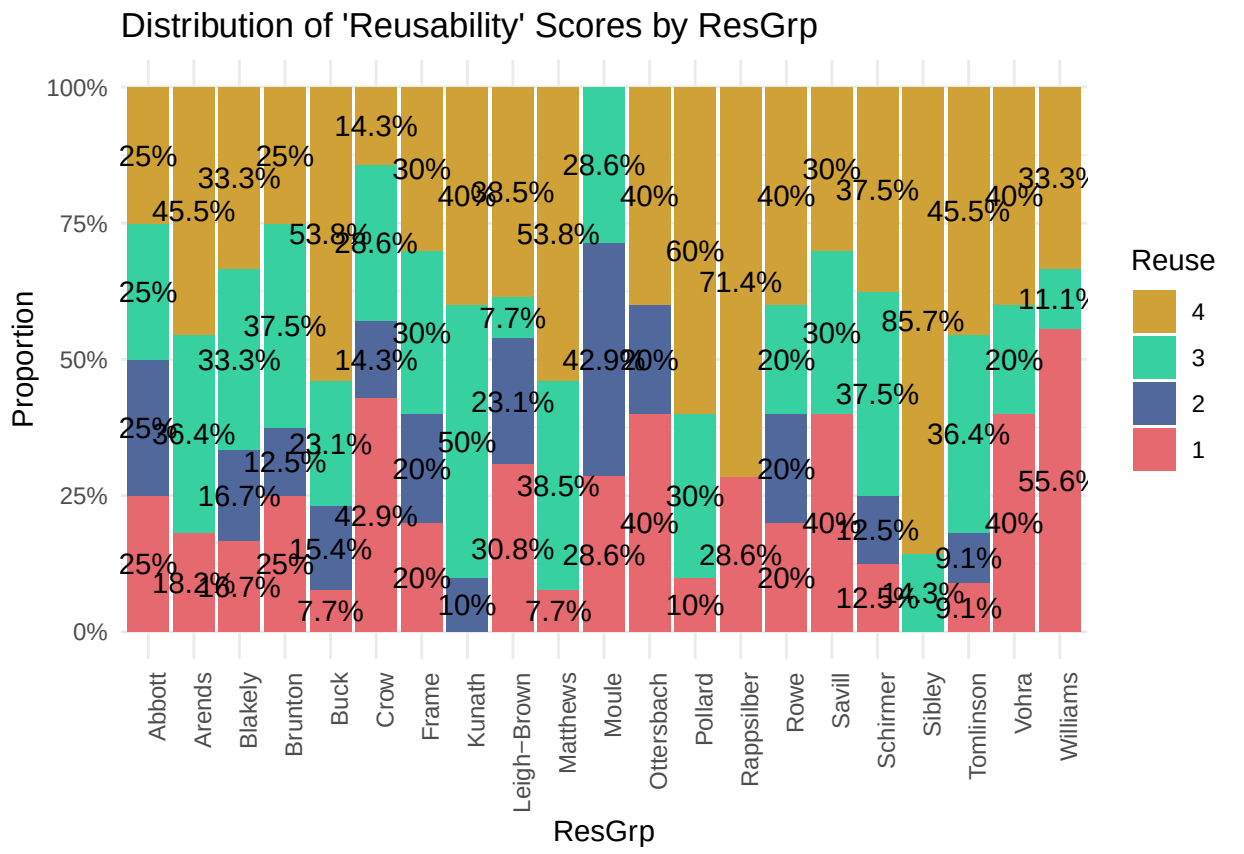
```
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(title = "Distribution of 'Completeness' Scores by ResGrp",
       x = "ResGrp",
       y = "Proportion",
       fill = "Complete") +
  theme_minimal() +
  scale_x_discrete(name = "ResGrp", labels = unique(long_data$ResGrp)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(internal1)
```




```
ggsave("internal1.png", internal1, width = 15, height = 10, units = "in", bg= "white")
```

```
# Calculate the frequency and percentage for each 'Reuse' score among the research Group
long_data <- data %>%
  count(ResGrp, Reuse) %>%
  group_by(ResGrp) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
```

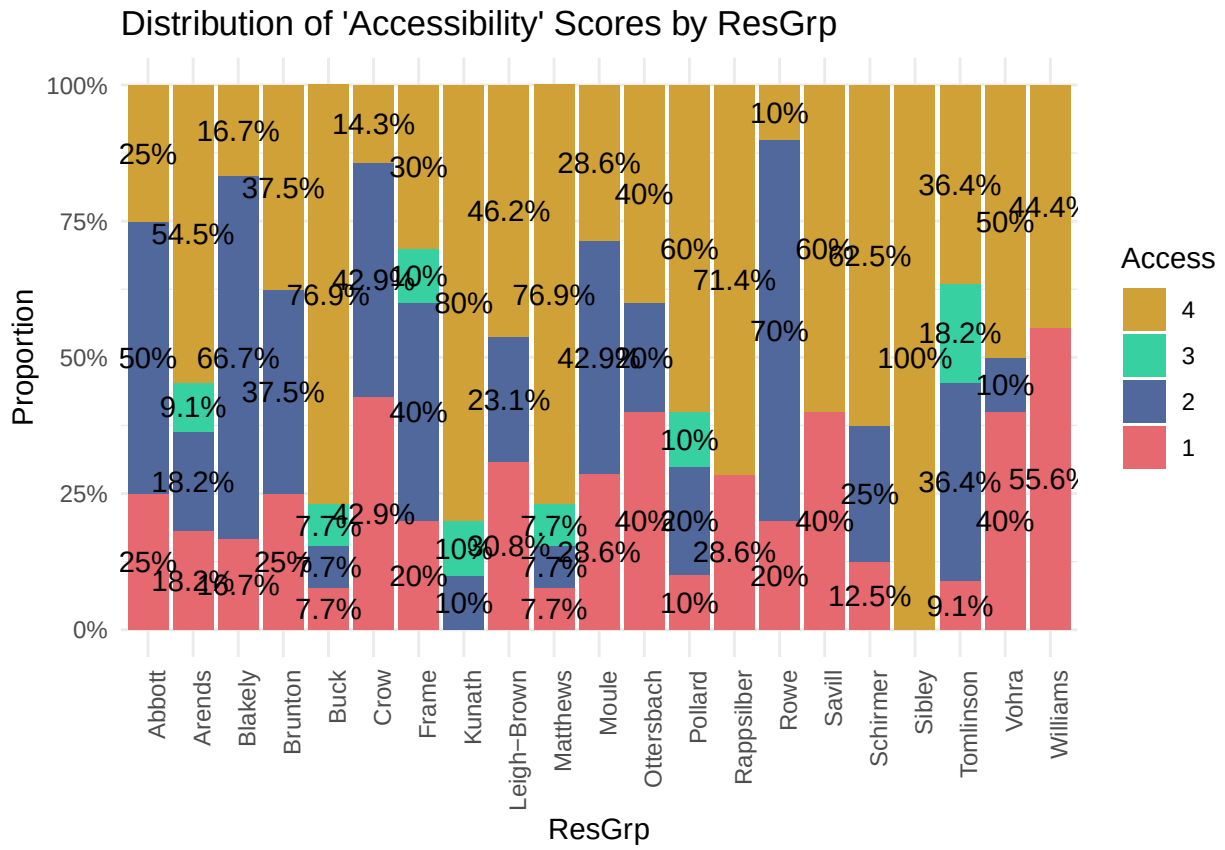
```r
  ungroup()

long_data <- long_data %>%
  mutate(Reuse = factor(Reuse, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
internal2 <- ggplot(long_data, aes(x = as.factor(ResGrp), y = n, fill = as.factor(Reuse))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(title = "Distribution of 'Reusability' Scores by ResGrp",
       x = "ResGrp",
       y = "Proportion",
       fill = "Reuse") +
  theme_minimal() +
  scale_x_discrete(name = "ResGrp", labels = unique(long_data$ResGrp)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(internal2)
```




```
ggsave("internal2.png", internal2, width = 15, height = 10, units = "in", bg= "white")
```

```
# Calculate the frequency and percentage for each 'Access' score among the research Group
long_data <- data %>%
  count(ResGrp, Access) %>%
  group_by(ResGrp) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

long_data <- long_data %>%
  mutate(Access = factor(Access, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
internal3 <- ggplot(long_data, aes(x = as.factor(ResGrp), y = n, fill = as.factor(Access))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
```

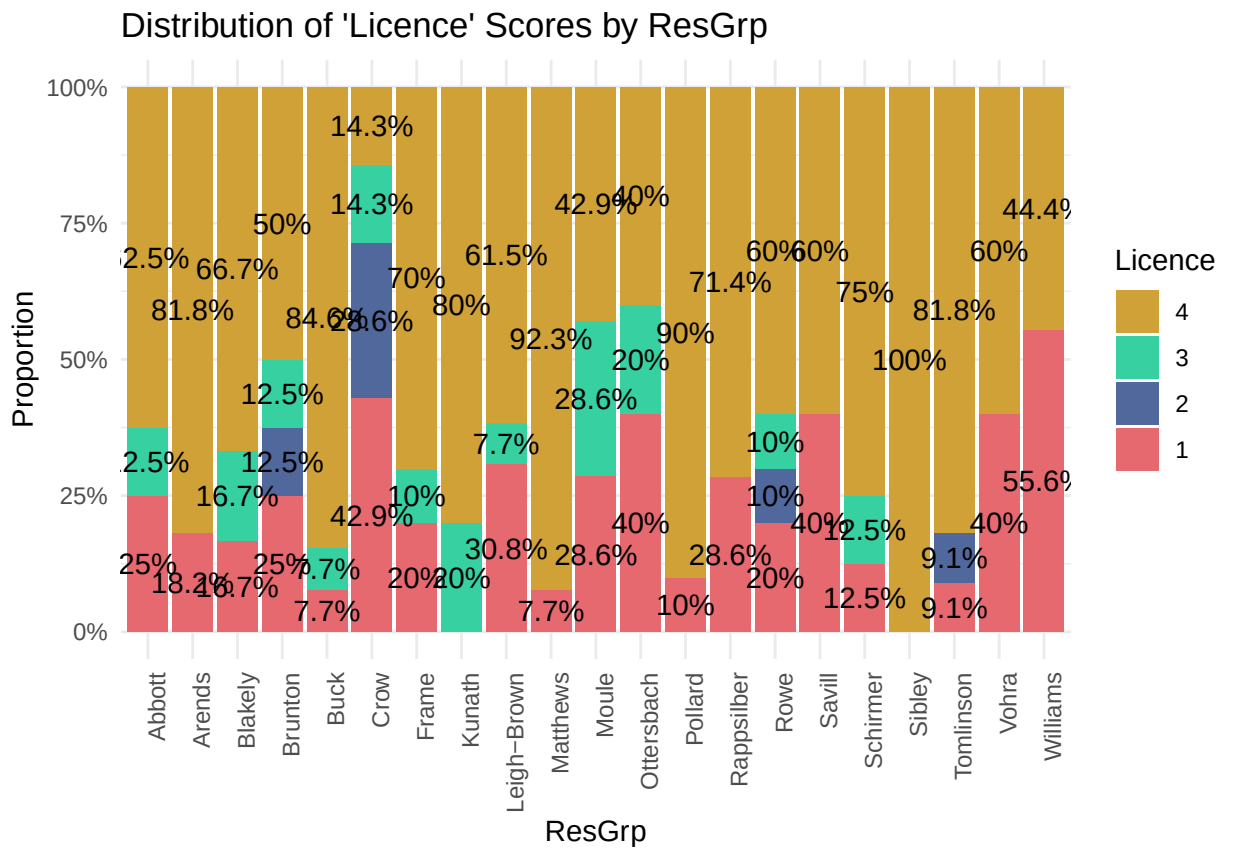
```
  ) +
  scale_fill_manual(values = colors) +
  labs(title = "Distribution of 'Accessibility' Scores by ResGrp",
       x = "ResGrp",
       y = "Proportion",
       fill = "Access") +
  theme_minimal() +
  scale_x_discrete(name = "ResGrp", labels = unique(long_data$ResGrp)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(internal3)
```




```
ggsave("internal3.png", internal3, width = 15, height = 10, units = "in", bg= "white")
```

```
# Calculate the frequency and percentage for each 'Licence' score among the research Group
long_data <- data %>%
  count(ResGrp, Licence) %>%
  group_by(ResGrp) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()
```

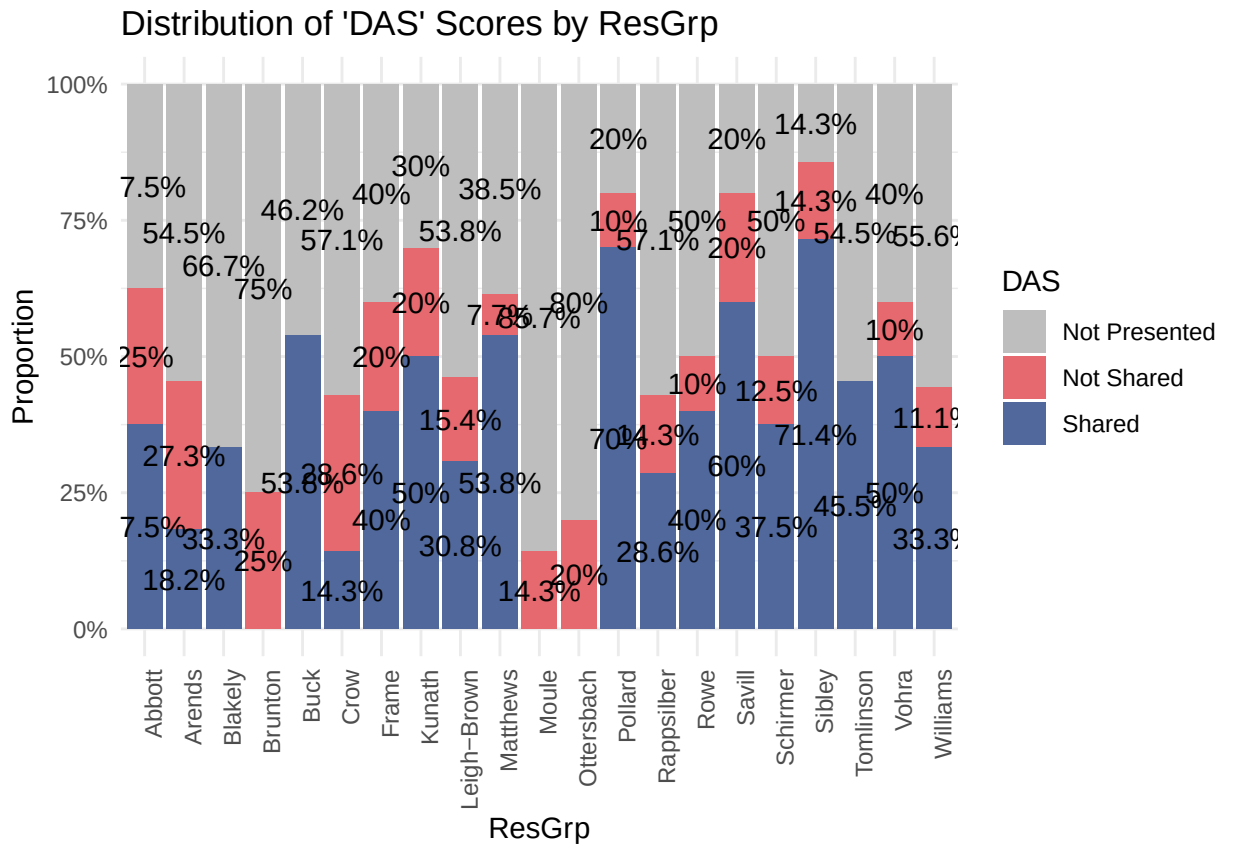
```r
long_data <- long_data %>%
  mutate(Licence = factor(Licence, levels = c("4", "3", "2", "1")))

# Custom colors
colors <- c("#D0A136", "#36D0A1","#50689B","#E5696F")

# Creating the stacked bar chart with percentage labels
internal4 <- ggplot(long_data, aes(x = as.factor(ResGrp), y = n, fill = as.factor(Licence))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(title = "Distribution of 'Licence' Scores by ResGrp",
       x = "ResGrp",
       y = "Proportion",
       fill = "Licence") +
  theme_minimal() +
  scale_x_discrete(name = "ResGrp", labels = unique(long_data$ResGrp)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(internal4)
```




```
ggsave("internal4.png", internal4, width = 15, height = 10, units = "in", bg= "white")
```

```
# Calculate the frequency and percentage for each 'DAS' score among the research Group
long_data <- data %>%
  count(ResGrp, NewDAS) %>%
  group_by(ResGrp) %>%
  mutate(Percentage = n / sum(n) * 100) %>%
  mutate(Label = ifelse(Percentage > 5, paste0(round(Percentage, 1), "%"), "")) %>%
  mutate(Cumulative_Percentage = cumsum(Percentage) - (0.5 * Percentage)) %>%
  ungroup()

# Custom colors
colors <- c("Not Presented" = "gray", "Not Shared" = "#E5696F", "Shared" = "#50689B")

# Creating the stacked bar chart with percentage labels
DASinternal <- ggplot(long_data, aes(x = as.factor(ResGrp), y = n, fill = as.factor(NewDAS))) +
  geom_bar(stat = "identity", position = "fill") +  # Scale the bar heights to proportions
  scale_y_continuous(labels = percent_format()) +  # Convert the y-axis to percentage
  geom_text(
    aes(label = Label, y = Percentage),
    size = 4,
    color = "black",
    position = position_fill(vjust = 0.5)
  ) +
  scale_fill_manual(values = colors) +
  labs(title = "Distribution of 'DAS' Scores by ResGrp",
```

```
       x = "ResGrp",
       y = "Proportion",
       fill = "DAS") +
  theme_minimal() +
  scale_x_discrete(name = "ResGrp", labels = unique(long_data$ResGrp)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Adjust text angle for readability

# Print the plot
print(DASinternal)
```




```
ggsave("DASinternal.png", DASinternal, width = 15, height = 10, units = "in", bg= "white")
```