WSP setup and cleaning code

Lizzie Jones[1]

02/05/2021

Data cleaning!

About this rMarkdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com). To generate the document of all content, click the **Knit** button.

This rMarkdown document will be periodically updated and uploaded to the OneDrive folder and pushed to the WSP GitHub code repository. The primary format of this document is HTML, but this can be easily changed by changing the output (e.g. PDF, GitHub) using the 'output' section at the top of the document. The possible output formats are listed here: https://rmarkdown.rstudio.com/lesson-9.html (https://rmarkdown.rstudio.com/lesson-9.html).

Data cleaning intro

First I have conducted some data cleaning to identify any respondents or data points that need to be removed and explain why. First I converted the 'TimeTaken' column to a total number of seconds (SecsTaken) for easier to more easily investigate means and quantiles.

I initially focussed on the fastest 5% of respondents across both surveys as they are most likely to have straightlined through the survey. I visually inspected the data, then used the 'careless' package to find evidence of straightlining 'even-odd' consistencies, and intra-individual response variability (IRV), across the whole survey and within the multiple choice questions (particularly questions 4, 5, 13, 15, 16, 17, 22, 23, 24).

Finally I went through the full dataset manually and checked for any respondents that were definitely not taking the questionnaire seriously (e.g. open answers such as "jkjkjkjk"). I removed the entire row for respondents that were also straightlining, but for those who appreared to take the close questions seriously and put junk answers for the open questions, I removed the open answers only.

```
## Cleaning full dataset to prevent having to do code for all samples
all_data$Age_group_match <- all_data$Age_group # Create new column with matching age-group formats
all_data <- all_data  %>%
  dplyr::mutate(Age_group_match = recode(Age_group_match, "c('65-74', '75 and over')='65+'"))
summary(all_data$Age_group_match)
```

```
##                18-24                25-34                35-44
##                  260                  510                  585
##                45-54                55-64                  65+
##                  700                  774                  719
## Prefer not to answer
##                   12
```

```
# Formatting date and time columns
all_data$SecsTaken <- as.numeric(lubridate::seconds(all_data$TimeTaken)) # Create numeric column of time taken (s
econds)
all_data$StartDate <- as.Date(all_data$StartDate, format = "%d/%m/%Y")
all_data$CompletionDate <- as.Date(all_data$CompletionDate, format = "%d/%m/%Y")
```

```
##### Data cleaning using the 'careless' package

# Overall straightlining (whole survey)
all_straightline <- longstring(all_data, avg = FALSE) # Identifies the longest string of identical consecutive re
sponses for each observation
summary(all_straightline) # Mean number of consecutive attitude answers = 14, max = 14
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00   11.00   11.00   11.22   13.00   14.00
```

```
all_possible_st <- which(grepl(14, all_straightline)) # 127 rows with 14 consecutive answers (possible candidates
for removal)


### Checking straightlining for all Likert style questions with over 3 columns
# Checking the attitudes to WS columns (Q12, 13 and 14)
ncol(all_attitude_colnames) # Max possible number of consecutive answers is 10
```

```
## [1] 10
```

```
attitudes_straight <- longstring(all_attitude_colnames, avg = FALSE) # Identifies the longest string of identical
consecutive responses for each observation
summary(attitudes_straight) # Mean number of consecutive attitude answers = 3
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.000   2.995   4.000  10.000
```

```
attitude_possible_st <-which(grepl(10, attitudes_straight)) # Find rows with 10 consecutive answers (possible can
didates for removal)

# Checking the NCI columns
ncol(Q19_NCI_colnames) # Max possible number of consecutive answers is 6
```

```
## [1] 6
```

```
nci_straight <- longstring(Q19_NCI_colnames, avg = FALSE)
summary(nci_straight) # Mean number of consecutive attitude answers = 3
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   5.000   4.611   6.000   6.000
```

```
summary(which(grepl(6, nci_straight))) # Find rows with 6 consecutive answers (~1700 gave max consecutive for NCI
across both surveys, which makes sense especially for proactive sample, as sample will have a high interest and c
onnection to nature)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0   831.5  1624.0  1672.1  2424.5  3560.0
```

```
# Checking the ProCoBS columns
ncol(Q21_ProCoBS_colnames) # Max possible number of consecutive answers is 4
```

```
## [1] 4
```

```
ProCoBS_straight <- longstring(Q21_ProCoBS_colnames, avg = FALSE)
summary(ProCoBS_straight) # Mean number of consecutive attitude answers = 3
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   1.827   2.000   4.000
```

```
summary(which(grepl(4, ProCoBS_straight))) # 245 rows with 10 consecutive answers (possible candidates for remova
l, but only 4 questions so unintentional straightlining would be likely for this question)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       9    1251    2459    2105    2935    3553
```

```
# Comparing overall straightlining row numbers to attitude row numbers
both_straightline_rownames <- intersect(all_possible_st,attitude_possible_st) # rows in both
rows_straightlined <- all_data[c(2678, 2713, 2723, 2738, 2772, 3121, 3209, 3287, 3307, 3455, 3503), ] # Create df
to view
summary(rows_straightlined$SecsTaken) # Most took survey quickly and have skipped the open questions
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    21.0   106.0   153.0   196.4   242.0   491.0
```

```
# Removing straightlined participants
ID_straightlined <- c(2678, 2713, 2723, 2738, 2772, 3121, 3209, 3287, 3307, 3455, 3503)
## Create new dataset for further analysis and remove rows with straightlining etc.
data_clean <- all_data[!all_data$UniqueID_all %in% ID_straightlined,]

# Removing non-serious (and often also straightlined through most questions) participants
ID_notserious <- c(2607, 2630, 3297, 3285, 3340, 3441, 3439, 3474)
## Create new dataset for further analysis and remove rows with straightlining etc.
data_clean <- data_clean[!data_clean$UniqueID_all %in% ID_notserious,]
```

```
### Explore average time taken to complete questionnaire and check for straightlining
quantile(data_clean$SecsTaken, 0.1) # Fastest 10% of all respondents = completion in 188.9 seconds/ about 3 mins
```

```
## 10%
## 191
```

```
quantile(data_clean$SecsTaken, 0.05) # Fastest 5% of all respondents = completion in 117.95 seconds/ about 2 mins
```

```
##  5%
## 121
```

```
quantile(data_clean$SecsTaken, 0.025) # Fastest 2.5% of all respondents = completion in 70.975 seconds/ about 1.2
mins
```

```
## 2.5%
## 71.5
```

```
fastest_10 <- subset(data_clean, SecsTaken < 191) # Sample of fastest 10% of all respondents
fastest_5 <- subset(data_clean, SecsTaken < 121) # Sample of fastest 5% of all respondents
fastest_2.5 <- subset(data_clean, SecsTaken < 72) # Sample of fastest 2.5% of all respondents
summary(fastest_5$SurveyType) # 96% of respondents in fastest 5% are from the NatRep sample
```

```
##    NatRep Proactive
##       170         7
```

```
summary(fastest_2.5$SurveyType) # 100% of respondents in fastest 2.5% are from the NatRep sample
```

```
##    NatRep Proactive
##        89         0
```

Checking the fastest 5% of responses

Here I have checked the responses of the fastest 5% of the dataframe (after straightlined responses had been removed). I compare the mean values of the numeric/score columns between the full cleaned dataset and the fastest 5%, checked for overall straightlining again and then manually checked the dataset for any irregularities.

I have then created a 'final' dataset for further data checking, stats and analysis called 'final_data'.

```
### Checking the fastest 5% of respondents
# Checking the fastest 5% of the cleaned dataset for straightlining across whole survey
long_fastest_5 <- longstring(fastest_5, avg = FALSE) # Identifies the longest string of identical consecutive res
ponses for each respondent
evenodd_fastest_5 <- evenodd(fastest_5, rep(5,10)) # Calculates the even-odd consistency score

# Checking the fastest 5% for straightlining within each set of mutliple choice questions
# Q5 diet
# summary(data_clean$Q5_overallscore_diet)
# summary(fastest_5$Q5_overallscore_diet) ### Not a significant difference in Q5 diet score between all_data, fas
test 5% and 2.5% samples
# # Q6 habitat
# summary(data_clean$Q6_habitat_overallscore)
# summary(fastest_5$Q6_habitat_overallscore)
# # Overall knowledge score
# summary(data_clean$KnowledgeScore)
# summary(fastest_5$KnowledgeScore)
# # NCI
# summary(data_clean$NCI)
# summary(fastest_5$NCI)
# # Pro-cons behaviours
# summary(data_clean$ProCoBS)
# summary(fastest_5$ProCoBS)
# # Bird Interest Score
# summary(data_clean$BirdInterestScore)
# summary(fastest_5$BirdInterestScore)

### Full cleaned dataset
# Calculates the even-odd consistency score
careless_all <- evenodd(data_clean, rep(5,10))
# Calculates the intra-individual response variability (IRV)
irv_total <- irv(data_clean)

### Fastest 5%
# Calculates the even-odd consistency score
careless_fast <- evenodd(fastest_5, rep(5,10))
# Calculates the intra-individual response variability (IRV)
irv_fast <- irv(fastest_5)


# Writing the fastest 5% subset of the cleaned dataframe as a dataframe for visual inspection in Excel
# write.csv(fastest_5, "WSP_fastest5.csv")

# Manually check the data
manualcheckID_to_remove <- c(3321, 2643, 566, 916) # Removed as comments suggested not taking the survey seriousl
y (e.g. "lololol")
## Create new dataset for further analysis and remove rows with straightlining etc.
data_clean <- data_clean[!data_clean$UniqueID_all %in% manualcheckID_to_remove,]
```

Checking the final dataframes for incompatible answers

Questions with multiple choice columns (e.g., Diet, habitat and nesting habitat - see the full questions below). For these questions respondents had the oppertunity to provide multiple answers (e.g., Diet = fish/ carrion) and able to also select don't know. For these columns I am using the following code to inspect the data and remove any 'don't know' answers where a respondent has also selected any other choice/answer.

Questions: Q5) What do white stork's typically eat? select all that apply Q6) What are white stork's preferred feeding habitat? select all that apply Q7) Where do white stork's typically nest? select all that apply

```
# colnames(final_data)

### DIET
# Create a sum of all Q5_diet columns (not Don't know, rawscore, overallscore)
final_data <- final_data %>%
  mutate(Q5_diet_sum = rowSums(select(., starts_with("Q5") & ends_with("diet"))))
# Where the summed columns >0 set 'don't know' column to 0
final_data$Q5l_diet_Don.tKnow <- ifelse(final_data$Q5_diet_sum > 0, final_data$Q5l_diet_Don.tKnow, 0)
# View all diet columns to check data
Q5_alldiet_colnames <- select(final_data, starts_with("Q5"))
# Q5_alldiet_colnames


### PREFERRED HABITAT
# Create a sum of all Q6_habitat columns (not Don't know, rawscore, overallscore)
final_data <- final_data %>%
  mutate(Q6_habitat_sum = rowSums(select(., starts_with("Q6") & ends_with("habitat"))))
# Where the summed columns >0 set 'don't know' column to 0
final_data$Q6f_habitat_Don.tKnow <- ifelse(final_data$Q6_habitat_sum > 0, final_data$Q6f_habitat_Don.tKnow, 0)
# View all habitat columns to check data
Q6_allhabitat_colnames <- select(final_data, starts_with("Q6"))
# Q6_allhabitat_colnames


### NESTING HABITAT
# Create a sum of all Q7_nesting columns (not Don't know, rawscore, overallscore)
final_data <- final_data %>%
  mutate(Q7_nesting_sum = rowSums(select(., ends_with("nesting"))))
# Where the summed columns >0 set 'don't know' column to 0
final_data$Q7f_nesting_Don.tKnow <- ifelse(final_data$Q7_nesting_sum > 0, final_data$Q7f_nesting_Don.tKnow, 0)
# View all habitat columns to check data
Q7_allnesting_colnames <- select(final_data, starts_with("Q7"))
# Q7_allnesting_colnames

# View the overall score columns
overall_score_colnames <- select(final_data, ends_with("overallscore")) # Can use for stacked bar charts in close
d question analysis
# overall_score_colnames

# View and save the FINAL cleaned dataframe as a CSV file for use in other r scripts
# write.csv(final_data, "WSP_R_cleaned_dataset3.csv")
```

Cronbach’s alpha

Now we have a cleaned dataset I have gone through the grouped columns are numeric scores of Likert or multiple choice questions, including: AttitudeScore, NCI, EnvConcern.score, ProCoBS and BirdInterestScore.

Based on the 0.7 threshold, all groups have an acceptable Cronbach’s alpha score.

```
### Reminding myself of the column names again!
# colnames(final_data)
# nrow(final_data)

# Using Cronbach's alpha on the score columns using the psych package (alpha::psych)
# Question 13 attitudes
final_data %>%
  select(., starts_with("Q13") &  ends_with('score')) %>%
  psych::alpha(title = "Attitudes")
```

```
##
## Reliability analysis  Attitudes
## Call: psych::alpha(x = ., title = "Attitudes")
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean   sd median_r
##       0.82      0.82    0.81      0.47 4.5 0.0048    4 0.74     0.46
##
##  lower alpha upper     95% confidence boundaries
## 0.81 0.82 0.83
##
##  Reliability if an item is dropped:
##                       raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r
## Q13.1_agreement_score      0.80      0.79    0.76      0.49 3.8   0.0056 0.026
## Q13.2_agreement_score      0.82      0.82    0.79      0.53 4.5   0.0047 0.014
## Q13.3_agreement_score      0.75      0.76    0.73      0.44 3.1   0.0068 0.013
## Q13.4_agreement_score      0.78      0.79    0.76      0.48 3.7   0.0059 0.012
## Q13.5_agreement_score      0.75      0.75    0.72      0.43 3.0   0.0069 0.013
##                       med.r
## Q13.1_agreement_score  0.50
## Q13.2_agreement_score  0.54
## Q13.3_agreement_score  0.43
## Q13.4_agreement_score  0.46
## Q13.5_agreement_score  0.41
##
##  Item statistics
##                         n raw.r std.r r.cor r.drop mean   sd
## Q13.1_agreement_score 3491  0.72  0.74  0.64   0.57  4.5 0.82
## Q13.2_agreement_score 3416  0.68  0.67  0.54   0.47  3.9 0.96
## Q13.3_agreement_score 3389  0.82  0.82  0.77   0.70  3.8 1.04
## Q13.4_agreement_score 2910  0.76  0.75  0.67   0.61  3.7 1.05
## Q13.5_agreement_score 3251  0.83  0.83  0.79   0.72  4.0 1.00
##
## Non missing response frequency for each item
##                            1    2    3    4    5 miss
## Q13.1_agreement_score 0.02 0.01 0.07 0.25 0.65 0.01
## Q13.2_agreement_score 0.02 0.04 0.28 0.34 0.32 0.03
## Q13.3_agreement_score 0.04 0.06 0.26 0.38 0.26 0.04
## Q13.4_agreement_score 0.04 0.09 0.28 0.36 0.24 0.18
## Q13.5_agreement_score 0.03 0.05 0.18 0.41 0.33 0.08
```

```
# Question 14 attitudes
final_data %>%
  select(., starts_with("Q14") &  ends_with('score')) %>%
  psych::alpha(title = "Attitudes")
```

```
##
## Reliability analysis  Attitudes
## Call: psych::alpha(x = ., title = "Attitudes")
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean   sd median_r
##       0.77      0.77    0.76      0.41 3.4 0.0063  4.2 0.66      0.4
##
##  lower alpha upper     95% confidence boundaries
## 0.75 0.77 0.78
##
##  Reliability if an item is dropped:
##                       raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r
## Q14.1_agreement_score      0.73      0.74    0.70      0.42 2.9   0.0077 0.0218
## Q14.2_agreement_score      0.72      0.73    0.69      0.41 2.7   0.0077 0.0093
## Q14.3_agreement_score      0.69      0.70    0.65      0.37 2.4   0.0085 0.0078
## Q14.4_agreement_score      0.77      0.77    0.73      0.46 3.4   0.0062 0.0117
## Q14.5_agreement_score      0.70      0.71    0.68      0.38 2.5   0.0083 0.0163
##                       med.r
## Q14.1_agreement_score  0.39
## Q14.2_agreement_score  0.40
## Q14.3_agreement_score  0.37
## Q14.4_agreement_score  0.44
## Q14.5_agreement_score  0.35
##
##  Item statistics
##                         n raw.r std.r r.cor r.drop mean   sd
## Q14.1_agreement_score 3421  0.70  0.71  0.60   0.53  4.4 0.78
## Q14.2_agreement_score 3219  0.73  0.73  0.64   0.54  3.8 0.97
## Q14.3_agreement_score 3423  0.77  0.78  0.73   0.63  4.2 0.84
## Q14.4_agreement_score 3262  0.68  0.64  0.49   0.42  4.1 1.06
## Q14.5_agreement_score 3418  0.76  0.76  0.68   0.60  4.4 0.88
##
## Non missing response frequency for each item
##                            1    2    3    4    5 miss
## Q14.1_agreement_score 0.01 0.02 0.08 0.32 0.58 0.03
## Q14.2_agreement_score 0.02 0.05 0.27 0.38 0.27 0.09
## Q14.3_agreement_score 0.01 0.02 0.14 0.45 0.38 0.03
## Q14.4_agreement_score 0.04 0.05 0.12 0.32 0.47 0.08
## Q14.5_agreement_score 0.02 0.02 0.09 0.27 0.61 0.03
```

```
# Question 21 ProCoBS
final_data %>%
  select(., starts_with("Q21") & ends_with('score')) %>%
  psych::alpha(title = "ProCoBS")
```

```
##
## Reliability analysis  ProCoBS
## Call: psych::alpha(x = ., title = "ProCoBS")
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean  sd median_r
##       0.82      0.82    0.79      0.52 4.4 0.0047    4 1.4      0.5
##
##  lower alpha upper     95% confidence boundaries
## 0.81 0.82 0.83
##
##  Reliability if an item is dropped:
##             raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## Q21.1.score      0.84      0.84    0.79      0.64 5.4   0.0045 0.0094  0.61
## Q21.2.score      0.71      0.71    0.63      0.45 2.4   0.0082 0.0111  0.42
## Q21.3.score      0.77      0.76    0.73      0.52 3.2   0.0063 0.0428  0.44
## Q21.4.score      0.74      0.74    0.67      0.49 2.9   0.0071 0.0110  0.44
##
##  Item statistics
##                n raw.r std.r r.cor r.drop mean  sd
## Q21.1.score 3517  0.65  0.69  0.51   0.46  4.2 1.4
## Q21.2.score 3517  0.89  0.87  0.85   0.77  4.5 1.8
## Q21.3.score 3517  0.81  0.81  0.71   0.65  3.0 1.7
## Q21.4.score 3517  0.85  0.83  0.79   0.70  4.3 1.9
##
## Non missing response frequency for each item
##                1    2    3    4    5    6    7 miss
## Q21.1.score 0.04 0.08 0.17 0.30 0.23 0.13 0.05 0.01
## Q21.2.score 0.09 0.08 0.10 0.20 0.20 0.14 0.18 0.01
## Q21.3.score 0.25 0.22 0.15 0.21 0.09 0.05 0.04 0.01
## Q21.4.score 0.11 0.09 0.13 0.20 0.18 0.13 0.16 0.01
```

```
# Question 22 BirdInterestScore
final_data %>%
  select(., starts_with("Q23") & ends_with('Score')) %>%
  psych::alpha(title = "BirdInterestScore")
```

```
##
## Reliability analysis  BirdInterestScore
## Call: psych::alpha(x = ., title = "BirdInterestScore")
##
##   raw_alpha std.alpha G6(smc) average_r S/N    ase mean   sd median_r
##       0.87      0.87    0.82      0.69 6.6 0.0039  4.1 0.85     0.69
##
##  lower alpha upper     95% confidence boundaries
## 0.86 0.87 0.87
##
##  Reliability if an item is dropped:
##              raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## Q23.1..Score      0.77      0.77    0.63      0.63 3.4   0.0076    NA  0.63
## Q23.2.Score       0.85      0.85    0.74      0.74 5.7   0.0051    NA  0.74
## Q23.3.Score       0.81      0.82    0.69      0.69 4.4   0.0062    NA  0.69
##
##  Item statistics
##                 n raw.r std.r r.cor r.drop mean   sd
## Q23.1..Score 3539  0.90  0.91  0.85   0.79  4.2 0.90
## Q23.2.Score  3539  0.88  0.87  0.76   0.71  4.0 1.01
## Q23.3.Score  3539  0.89  0.89  0.81   0.74  4.1 0.96
##
## Non missing response frequency for each item
##                 1    2    3    4    5 miss
## Q23.1..Score 0.02 0.04 0.13 0.40 0.41    0
## Q23.2.Score  0.02 0.08 0.12 0.42 0.36    0
## Q23.3.Score  0.02 0.05 0.17 0.35 0.41    0
```

1. University of Brighton, l.jones4@brighton.ac.uk (mailto:l.jones4@brighton.ac.uk)↩