Score TDS Qualtrics Data

John Flournoy

2018-01-25

Contents

This presents both a walk-through for how you could go about adding a new questionnaire to be scored, and also should provide up-to-date descriptives and scored scales every time you recompile this Rmd document.

It would be nice if this were something completely automated. Unfortunately, the nature of data that is still under collection, and the fact that there are a lot of moving pieces means that something totally automated is likely to break pretty easily (at least, something automated coded by me). So this document will take a highly modular approach so hopefully if something goes wrong, you can see more exactly where that happens. I'll try to explain every step pretty verbosely too. You should probably be reading this document in R Studio. Make sure you're up to date with upgrades.

Setting options

If you're reading the compiled HTML, you won't see the options below, because we've writtent `warning=F,echo=F,message=F,error=F` in the head of the chunk. You can change this by replacing the "F"s with "T"s.

Install the `scorequaltrics` package

We need to have the scorequaltrics package installed. I wrote this so that I can maintain helpful functions related to scoring.

```
#this chunk won't evaluate. If you need to
#install the package, run this by hand.
devtools::install_github('jflournoy/qualtrics')
```

Accessing qualtrics data

You also need to have a token in a YAML formatted file for accessing qualtrics via the API. It's formatted like:

```
user: username
token: apitoken
```

Once we've loaded this, we can get a list of questionnaires.

```
library(scorequaltrics)
```

```
## Warning: replacing previous import 'data.table::first' by 'dplyr::first'
## when loading 'scorequaltrics'
```

```
## Warning: replacing previous import 'data.table::between' by
## 'dplyr::between' when loading 'scorequaltrics'
```

```
## Warning: replacing previous import 'data.table::last' by 'dplyr::last' when
## loading 'scorequaltrics'
```

```
## Warning: replacing previous import 'ggplot2::alpha' by 'psych::alpha' when
## loading 'scorequaltrics'
```

```
## Warning: replacing previous import 'ggplot2::%+%' by 'psych::%+%' when
## loading 'scorequaltrics'
```

```
library(ggplot2)
library(dplyr)
library(tidyr)

credentials <- scorequaltrics::creds_from_file(cred_file_location)

rawSurveys <- scorequaltrics::get_surveys(credentials)
rawSurveysTDS <- filter(rawSurveys, grepl('.*(TDS1|TDS2|TDS3).*',SurveyName))

knitr::kable(arrange(select(rawSurveysTDS, SurveyName), SurveyName))
```

SurveyName
QSupport TDS2 Session 3 - Parent - Copy - Copy
TDS1 AND TDS3 PSQI
TDS1 CBCL - Post
TDS1 CBCL - Pre
TDS1 PDS (Session2) - Post
TDS1 Saliva Questions - Post
TDS1 Saliva Questions - Pre
TDS1 Session 1 & 2 Makeups
TDS1 Session 1 - Post
TDS1 Session 1 - Pre
TDS1 Session 2 A - Post
TDS1 Session 2 A - Pre
TDS1 Session 2 B - Post
TDS1 Session 2 B - Pre
TDS1 Session 2 B - SHORTENED_A
TDS1 Session 2 B - SHORTENED_B
TDS1, Session 3 - Child
TDS1 Session 3 Child *A*
TDS1 Session 3 Child *B*
TDS1, Session 3 - Parent
TDS1, Session 3 - Parent part 2
TDS1 Spinning Wheel Game - New
TDS1 Spinning Wheel Game - Post

SurveyName
TDS2 CBCL
TDS2 - PDS (Session 2)
TDS2 Saliva Questions
TDS2 Session 1
TDS2 Session 2 A
TDS2 Session 2 B
TDS2 Session 2 B - SHORTENED_A
TDS2 Session 2 B - SHORTENED_B
TDS2 Session 3 - Child
TDS2 Session 3 - KSRQ & SAQ
TDS2 Session 3 - Parent
TDS2 Session 3 - Parent - Copy
TDS2 Spinning Wheel Game

We have a lot of different questionnaires from different samples and different sessions. For simplicity, and to aid in diagnosing any problems, we can proceed through each sample and each wave of data collection. *Note*, however, that if we ensured that naming conventions were consistent across all questionnaires and rubrics, and if we had accurate session dates attached, we could score everything in one fell swoop.

Cleaning and scoring data

As an example, I'll walk through step-by-step how to clean and score data using the TDS2, Wave 1 sample.

This is the first sample collected, so we can begin here. I'll demonstrate in this how to do a single massive data scoring. Following that, I'll demonstrate how you can get more information about scales that have been constructed in a psychometric tradition, and that therefore are easy to evaluate using standard reliability metrics.

First, we download the data for the surveys we want.

```
tds2_wave1_surveys <- rawSurveysTDS %>%
    filter(grepl('TDS2 (Session [12]|CBCL|- PDS)', SurveyName))

print(tds2_wave1_surveys$SurveyName)
```

```
## [1] "TDS2 - PDS (Session 2)"         "TDS2 CBCL"
## [3] "TDS2 Session 2 B"               "TDS2 Session 2 B - SHORTENED_B"
## [5] "TDS2 Session 2 A"               "TDS2 Session 1"
## [7] "TDS2 Session 2 B - SHORTENED_A"
```

```
tds2_wave1_long <- scorequaltrics::get_survey_data(tds2_wave1_surveys,
                                                   credentials,
                                                   pid_col = pid_column_name)

dim(tds2_wave1_long)
```

```
## [1] 86051     4
```

```
names(tds2_wave1_long)
```

```
## [1] "SID"         "item"        "value"       "survey_name"
```

The resulting data frame should have a lot of rows (the first part of the output of `dim`) and 4 columns.

```
## It looks like all is in order here. Note that the PID column is named "SID".
```

Before doing any scoring, we should take care of all the complex response recoding that may be specified. So We'll load all the response recoding rubrics and apply those. It's important that you pass the full path of the file to the next function, so if you use `dir` to collect filenames as I do below, make sure you set `full.names = TRUE`.

```
dir(file.path(tds2_wave1_rubric_dir), pattern = '.*response_recoding.*.csv')
```

```
## [1] "PAL2_response_recoding.csv"
## [2] "SES_response_recoding.csv"
## [3] "YRBS_response_recoding_TDS2_session_2.csv"
```

```
#You should see a result below -- if not, the path is likely wrong.

tds2_wave1_recoding_rubrics <- data.frame(file = dir(file.path(tds2_wave1_rubric_dir),
                                                     pattern = '.*response_recoding.*.csv',
                                                     full.names = TRUE))

tds2_wave1_recoding_data_long <- scorequaltrics::get_rubrics(tds2_wave1_recoding_rubrics,
                                                             type = 'recoding')

tds2_wave1_long_recoded <- scorequaltrics::recode_responses(tds2_wave1_long,
                                                            tds2_wave1_recoding_data_long)
```

Now let's load in the scoring rubrics.

```
tds2_wave1_scoring_rubrics <- data.frame(file = dir(file.path(tds2_wave1_rubric_dir),
                                                    pattern = '.*scoring_rubric.*.csv',
                                                    full.names = TRUE))

tds2_wave1_scoring_data_long <- scorequaltrics::get_rubrics(tds2_wave1_scoring_rubrics,
                                                            type = 'scoring')

head(tds2_wave1_scoring_data_long[, -1])
```

```
## # A tibble: 6 x 9
##   data_file_name scale_name column_name transform reverse min   max
##   <chr>          <chr>      <chr>       <chr>     <chr>   <chr> <chr>
## 1 TDS2_Session_1 ACE        ACE_1       0         0       0     0
## 2 TDS2_Session_1 ACE        ACE_2       0         0       0     0
## 3 TDS2_Session_1 ACE        ACE_3       0         0       0     0
## 4 TDS2_Session_1 ACE        ACE_4       0         0       0     0
## 5 TDS2_Session_1 ACE        ACE_5       0         0       0     0
## 6 TDS2_Session_1 ACE        ACE_6       0         0       0     0
## # ... with 2 more variables: scored_scale <chr>, include <chr>
```

Cleaning

We can make sure we clean out duplicate responses which will help later with ensuring that scale scores are calculated from teh correct subset of items. This is a point at which, if there is something funky going on, you'll want to investigate it and make a decision. For example, if a participant has two conflicting answers to the same question for the same wave, it's likely that a small investigation should commence.

We can also ensure that we're only scoring data for participants with the correct ID numbers. The line in the middle of the first call, `filter(grepl('[1234]\\d\\d', SID))`, ensures we only keep people with ID's starting with "1".

Before we do that, we can ensure that we're only keeping the data in the scoring rubrics in the first place.

```
tds2_wave1_long_recoded_nodupes <- tds2_wave1_long_recoded %>%
    get_items_in_rubric(tds2_wave1_scoring_data_long) %>%
    filter(grepl('1\\d\\d', SID)) %>%
    scorequaltrics::clean_dupes(pid_col = 'SID')
```

If you get "NAs introduced by coercion" it probably means that one of the rubrics references a column that has text input that is not transformable into a number. For example, if the questionnaire asks for ethnicity and someone writes in "White" it is not possible to turn that into a score to be used in a scale calculation (but there's a rubric that thinks it can). We can check that by using the function `scorequaltrics::get_uncoercibles()`.

```
tds2_wave1_uncoer <- tds2_wave1_long_recoded %>%
    get_items_in_rubric(tds2_wave1_scoring_data_long) %>%
    filter(grepl('[1234]\\d\\d', SID)) %>%
    scorequaltrics::get_uncoercibles() %>%
    distinct(item, value)

head(tds2_wave1_uncoer, 10)
```

```
## Empty data.table (0 rows) of 2 cols: item,value
```

```
unique(tds2_wave1_uncoer$item)
```

```
## character(0)
```

Now we can look at what rubrics have those items, if any.

```
tds2_wave1_scoring_data_long %>%
    filter(column_name %in% unique(tds2_wave1_uncoer$item),
           include %in% c(1, "1", "sum", "prod")) %>%
    ungroup() %>%
    select(scale_name, scored_scale, column_name, include)
```

```
## # A tibble: 0 x 4
## # ... with 4 variables: scale_name <chr>, scored_scale <chr>,
## #   column_name <chr>, include <chr>
```

If the above two chunks didn't result in output, we're good!

```
#Check that dropped values weren't ambiguous
tds2_wave1_long_recoded_nodupes %>%
  filter(dropped) %>%
  group_by(SID, item) %>%
  summarize(noinfo = all(length(unlist(old.value)) < 1)) %>%
  ungroup() %>%
  summarize(n_with_info = sum(!noinfo))
```

```
## # A tibble: 1 x 1
##   n_with_info
##         <int>
## 1          14
```

```
tds2_wave1_long_recoded_nodupes %>%
    filter(dropped) %>%
    group_by(SID, item) %>%
    filter(!all(length(unlist(old.value)) < 1)) %>%
    mutate(old.value = paste(old.value, collaps = ' ')) %>%
    knitr::kable()
```

SID	item	value	survey_name	old.value	dropped
109	YRBS_10	NA	TDS2 Session 2 B	c(1, 0)	TRUE
124	PDS_F3	NA	TDS2 Session 1	c(4, 3)	TRUE
125	YRBS_10	NA	TDS2 Session 2 B	c(1, 0)	TRUE
159	YRBS_10	NA	TDS2 Session 2 B	c(1, 0)	TRUE
189	YRBS_10	NA	TDS2 Session 2 B	c(1, 0)	TRUE
190	YRBS_10	NA	TDS2 Session 2 B	c(1, 0)	TRUE
196	BIS_1	NA	TDS2 Session 2 A	c(3, 2)	TRUE
196	BIS_10	NA	TDS2 Session 2 A	c(4, 2)	TRUE
196	BIS_14	NA	TDS2 Session 2 A	c(2, 3)	TRUE
196	BIS_2	NA	TDS2 Session 2 A	c(3, 2)	TRUE
196	BIS_6	NA	TDS2 Session 2 A	c(2, 3)	TRUE
196	BIS_7	NA	TDS2 Session 2 A	c(4, 3)	TRUE
196	SSS_3	NA	TDS2 Session 2 A	c(1, 0)	TRUE
196	SSS_4	NA	TDS2 Session 2 A	c(1, 0)	TRUE

For now, if there are ambiguous entries, I'm going to ingore them. The value to be used for scoring is set to NA – we have to treat that data as missing since the responses are in conflict.

Scoring almost all at once

There are a few different options for scoring questionnaires. First, we can provide a rubric and data to `scorequaltrics::score_questionnaire(dataDF, rubricsDF, psych = TRUE)`, which will use the `psych` package to do the scoring. This has the advantage that you get back a lot of information about the measurement quality of the scale, but it only works for scales that follow certain psychometric principles (e.g., each item is rated on a continuous scale, and is an indicator of a latent construct). It won't work well for other kinds of data (like scales where you want to know the number of risky behaviors, for example).

The second option is to use `scorequaltrics::score_step_one_and_two(dataDF, rubricsDF)` which was created to take care of several special cases for the TDS project questionnaires. The RPI, and RSQ both require special handling because of their idiosyncratic questionnaire design.

```
tds2_wave1_scored <- scorequaltrics::score_step_one_and_two(tds2_wave1_long_recoded_nodupes,
                                                            tds2_wave1_scoring_data_long)

set.seed(9567)
tds2_wave1_scored %>% ungroup %>%
    dplyr::sample_n(size = 10) %>%
    select(-SID)
```

```
## # A tibble: 10 x 6
##    scale_name        scored_scale       score     n_items n_missing method
##    <chr>             <chr>              <chr>       <int>     <int> <chr>
##  1 CBCL              somatic_complaints 0.090909~      11         0 1
##  2 CARE-R Expected ~ risky_car          1               2         0 1
##  3 RSQ_part2         rsq_mean_anxious_~ 19.6           10         0 1
##  4 PAL-2             pal2_antisocial    1.090909~      11         0 1
##  5 CARE-R Willingne~ risky_sex_regular~ 1               5         0 1
##  6 SPSRQ-S           sensitivity_rewar~ 0.1            10         0 1
##  7 UPPS-P            pos_urgency        2.357142~      14         0 1
##  8 PEQ-R             prosocial_to_me    3.6             5         0 1
##  9 RPI_part2         rpi_mean           3.444444~       9         1 1
## 10 CARE-R Social     care_soc_not_risk~ 3               3         0 1
```

One thing missing still is the Pubertal Development Scale scored via the Shirtcliff method, so I'll calculate that now with the special function `scorequaltrics::score_pdss()`

```
tds2_wave1_scored_pdss <- scorequaltrics::score_pdss(tds2_wave1_long_recoded_nodupes,
                          gender_mix = pdss_gender_mix,
                          gendercode = pdss_gender_code)
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

That hopefully went off without a hitch. Now we can examine the scales we've scored, and along the way, see how we can get more information about these scales when we are able to use the `psych` package scoring function.

Scoring a single scale using `score_questionnaire`

It is possible, and maybe most convenient in many cases, to score a single questionnaire at a time, and write it to a CSV file. I'll demonstrate with the Barratt's Impulsiveness Scale Version 15. This scale has just a single rubric – no recoding rubric, and no part 2 scoring rubric.

All we have to do is load the rubric using `get_rubrics` and then score the questionnaire using `score_questionnaire`.

```
tds2_wave1_bis_scoring <- scorequaltrics::get_rubrics(
    rubric_filenames = data.frame(
        file = file.path(tds2_wave1_rubric_dir, 'BIS_scoring_rubric.csv')))

#using the data we've already cleaned and recoded -- note that BIS does not require recoding
tds2_wave1_bis_scored <- scorequaltrics::score_questionnaire(tds2_wave1_long_recoded_nodupes,
                                                             tds2_wave1_bis_scoring)

#display a random set of 10 rows to check out the data.
set.seed(4210)
sample_n(select(ungroup(tds2_wave1_bis_scored), -SID), size = 10)
```

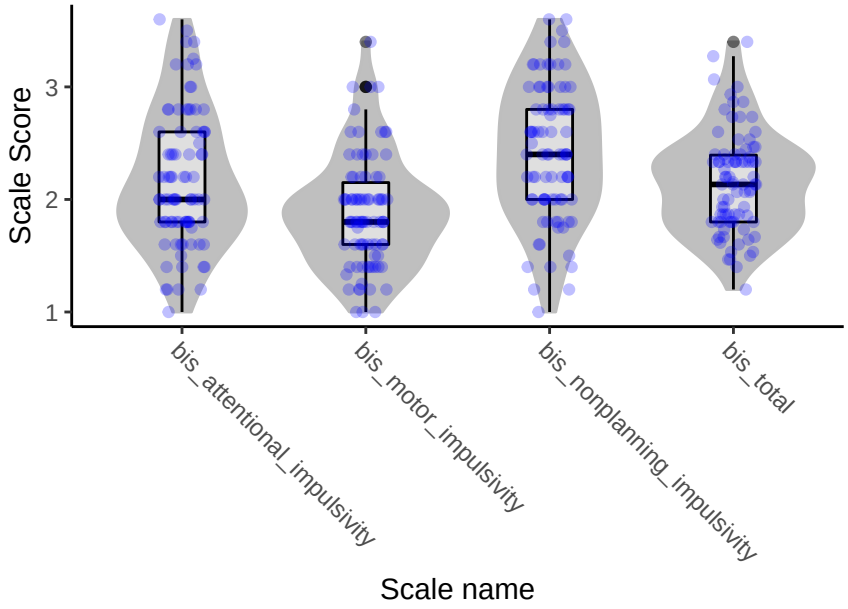
```
## # A tibble: 10 x 6
##    scale_name scored_scale                score   n_items n_missing method
##    <chr>      <chr>                       <chr>     <int>     <int> <chr>
##  1 BIS-15     bis_nonplanning_impulsivity 2.8           5         0 1
##  2 BIS-15     bis_nonplanning_impulsivity 2.4           5         0 1
##  3 BIS-15     bis_motor_impulsivity       1.6           5         0 1
##  4 BIS-15     bis_total                   1.6666~      15         0 1
##  5 BIS-15     bis_nonplanning_impulsivity 1.8           5         0 1
##  6 BIS-15     bis_total                   2.3333~      15         0 1
##  7 BIS-15     bis_motor_impulsivity       1.75          4         1 1
##  8 BIS-15     bis_motor_impulsivity       2             5         0 1
##  9 BIS-15     bis_total                   2.4666~      15         0 1
```

```
## 10 BIS-15     bis_total                    1.4666~      15         0 1
```

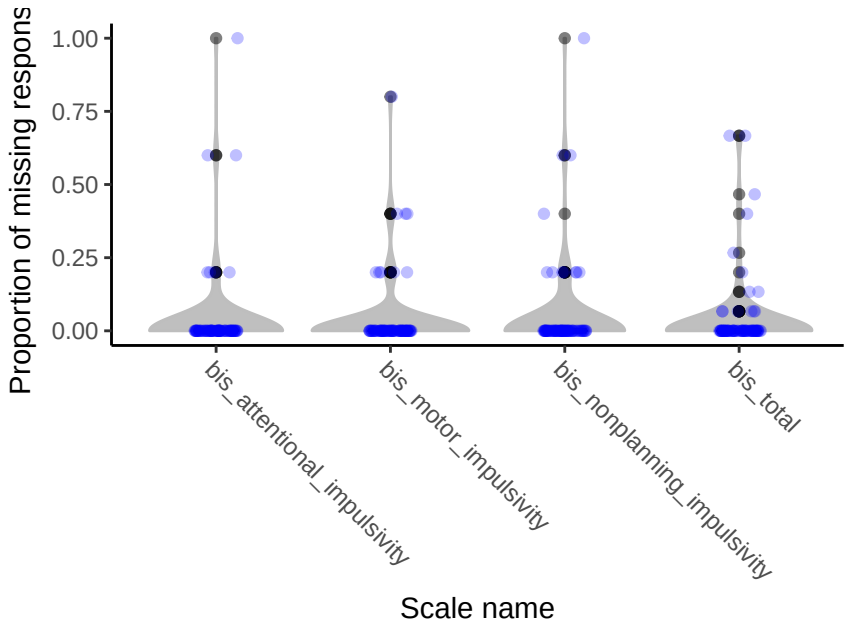
I can plot these data now using `plot_scored_scale`, which has options to select the scale name (not subscale) using regular expressions (defaults to all), or to plot number or proportion of items missing.

```
scorequaltrics::plot_scored_scale(tds2_wave1_bis_scored)
```

```
## Warning: Removed 2 rows containing non-finite values (stat_ydensity).
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
## Warning: Removed 2 rows containing missing values (geom_point).
```




```
scorequaltrics::plot_scored_scale(tds2_wave1_bis_scored, type = 'p_missing')
```



I can also widen this data frame for ease of use in regressions and exporting, using `widen_qualtrics_long`. Note that even with a single scored scale, you need to specify the scale name to be matched exactly. In this

case, we can see that `unique(tds2_wave1_bis_scored$scale_name)` gives us: BIS-15. Note that the return value of the `widen_qualtrics_long` function is a list that contains a data frame of scores at `...$scores` and of missing responses at `...$data_quality`.

```
tds2_wave1_bis_scored_wide <- scorequaltrics::widen_qualtrics_long(tds2_wave1_bis_scored,
                                                                   scale_names = 'BIS-15')

set.seed(322415)
sample_n(select(tds2_wave1_bis_scored_wide$scores, -SID), size = 3)
```

```
## # A tibble: 3 x 5
## # Groups:   scale_name [1]
##   scale_name bis_attentional_~ bis_motor_impul~ bis_nonplanning~ bis_total
##   <chr>      <chr>             <chr>            <chr>            <chr>
## 1 BIS-15     2                 2.4              2                2.133333~
## 2 BIS-15     2.8               2.4              2.6              2.6
## 3 BIS-15     2.8               1.8              2.4              2.333333~
```

```
sample_n(select(tds2_wave1_bis_scored_wide$data_quality, -SID), size = 3)
```

```
## # A tibble: 3 x 9
## # Groups:   scale_name [1]
##   scale_name bis_attentional_imp~ bis_attentional_impu~ bis_motor_impulsi~
##   <chr>                     <int>                 <int>             <int>
## 1 BIS-15                        5                     0                 5
## 2 BIS-15                        5                     0                 5
## 3 BIS-15                        5                     0                 5
## # ... with 5 more variables: bis_motor_impulsivity_n_missing <int>,
## #   bis_nonplanning_impulsivity_n_items <int>,
## #   bis_nonplanning_impulsivity_n_missing <int>, bis_total_n_items <int>,
## #   bis_total_n_missing <int>
```

We can easily write these tables to a file now:

```
write.csv(tds2_wave1_bis_scored_wide$scores, '/location/to/save/csv/file')
```

Scoring a single scale using `score_questionnaire` with `psych=TRUE`

There is a subset of scales that we can examine using tools from the psychometrics literature. That is, scales that consist of many items attempting to measure a theoretical latent construct can be examined using reliability measures to see how much consistency there is, from person to person, among these items that are ostensibly measuring the same thing. This doesn't make sense to look at in a scale like the Adverse Childhood Events or Youth Risk Behavior Survey because these are indices that are supposed to measure the number of certain events or behaviors that together are thought to convey risk. In other words, there is not theoretical basis for assuming that all adverse events in childhood are caused by the same latent construct. So even though it may be the case that certain adverse events tend to correlate, it is not necessary that they *do* correlate.

To investigate a scale using these tools, all we have to do is set it to be scored by the `psych` package, rather than the `scorequaltrics` backend. To do so, we merely set the option `psych=TRUE`. We can then apply the `print` function to get information about reliability, and the `pairs.panels` function to the object's `...$scores`.

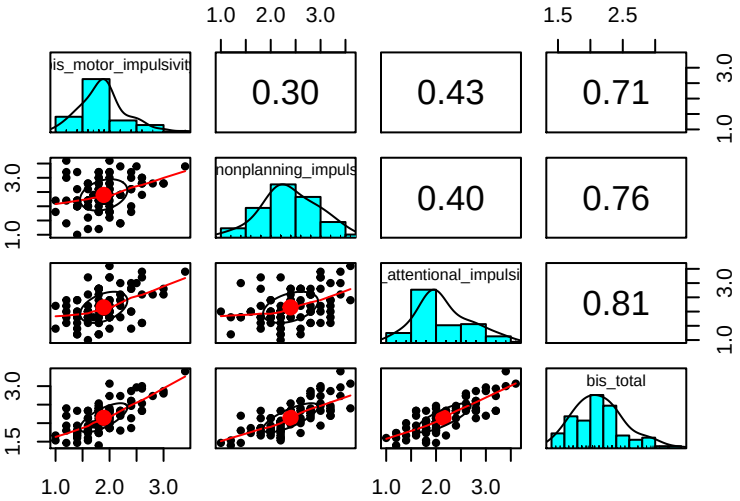
```
#using the rubric we've alread loaded:
tds2_wave1_bis_psych <- scorequaltrics::score_questionnaire(
    tds2_wave1_long_recoded_nodupes,
```

```
    tds2_wave1_bis_scoring,
    psych = T)

print(tds2_wave1_bis_psych)
```

```
## Call: scoreItems(keys = key_list, items = dataDF_w)
##
## (Unstandardized) Alpha:
##       bis_motor_impulsivity bis_nonplanning_impulsivity
## alpha                   0.7                        0.71
##       bis_attentional_impulsivity bis_total
## alpha                        0.73      0.81
##
## Standard errors of unstandardized Alpha:
##       bis_motor_impulsivity bis_nonplanning_impulsivity
## ASE                   0.076                       0.075
##       bis_attentional_impulsivity bis_total
## ASE                         0.073     0.038
##
## Average item correlation:
##           bis_motor_impulsivity bis_nonplanning_impulsivity
## average.r                  0.32                       0.33
##           bis_attentional_impulsivity bis_total
## average.r                        0.35      0.22
##
##  Guttman 6* reliability:
##          bis_motor_impulsivity bis_nonplanning_impulsivity
## Lambda.6                  0.76                       0.77
##          bis_attentional_impulsivity bis_total
## Lambda.6                        0.78      0.87
##
## Signal/Noise based upon av.r :
##              bis_motor_impulsivity bis_nonplanning_impulsivity
## Signal/Noise                   2.4                        2.4
##              bis_attentional_impulsivity bis_total
## Signal/Noise                          2.6       4.2
##
## Scale intercorrelations corrected for attenuation
##  raw correlations below the diagonal, alpha on the diagonal
##  corrected correlations above the diagonal:
##                            bis_motor_impulsivity
## bis_motor_impulsivity                       0.70
## bis_nonplanning_impulsivity                 0.30
## bis_attentional_impulsivity                 0.43
## bis_total                                   0.71
##                            bis_nonplanning_impulsivity
## bis_motor_impulsivity                             0.42
## bis_nonplanning_impulsivity                       0.71
## bis_attentional_impulsivity                       0.40
## bis_total                                         0.76
##                            bis_attentional_impulsivity bis_total
## bis_motor_impulsivity                             0.61      0.95
## bis_nonplanning_impulsivity                       0.55      1.01
## bis_attentional_impulsivity                       0.73      1.06
```

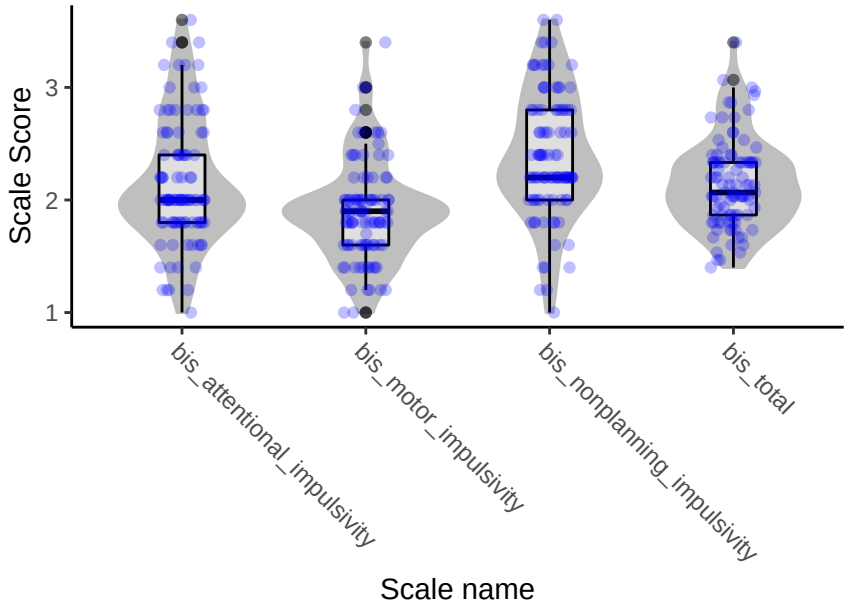
```
## bis_total                                          0.81      0.81
##
##  In order to see the item by scale loadings and frequency counts of the data
##  print with the short option = FALSE
```

```
pairs.panels(tds2_wave1_bis_psych$scores)
```



If we want to apply the `scorequaltrics` plot function, we need to transform the data to long-form using the `longen_psych_wide` function.

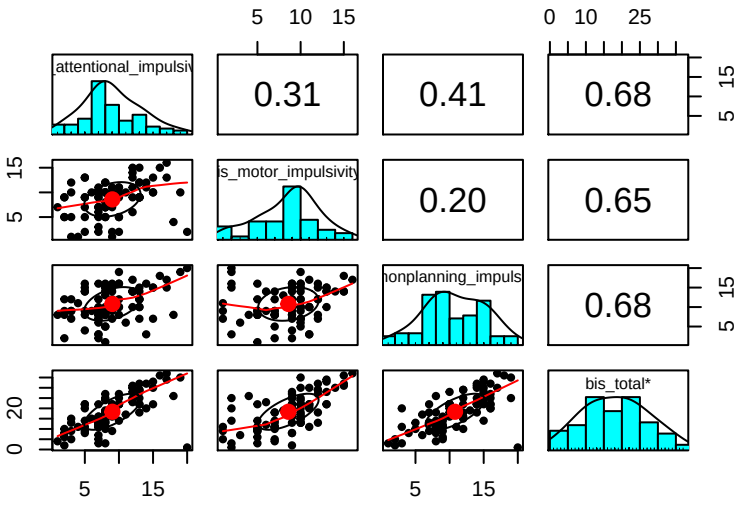
```
tds2_wave1_bis_score_long <- scorequaltrics::longen_psych_wide(tds2_wave1_bis_psych$scores)

scorequaltrics::plot_scored_scale(tds2_wave1_bis_score_long)
```



Finally, we can also use the `pairs.plot` function with data scored using the `scorequaltrics` backend using the `widen_qualtrics_long` function again. This time I'll take the BIS-15 data from the main scored data set.

```
tds2_wave1_bis15_wide <- scorequaltrics::widen_qualtrics_long(tds2_wave1_scored, scale_names = 'BIS-15'
```

```
library(psych)
#remove the first two columns that contain the ID and scale name
pairs.panels(tds2_wave1_bis15_wide$scores[, -(1:2)])
```



Putting it all together

In the next section, I will simply walk through the scoring and display of descriptives for the data we have using the tools reviewed above.

TDS 2, Wave 1 Scores

Available scales:

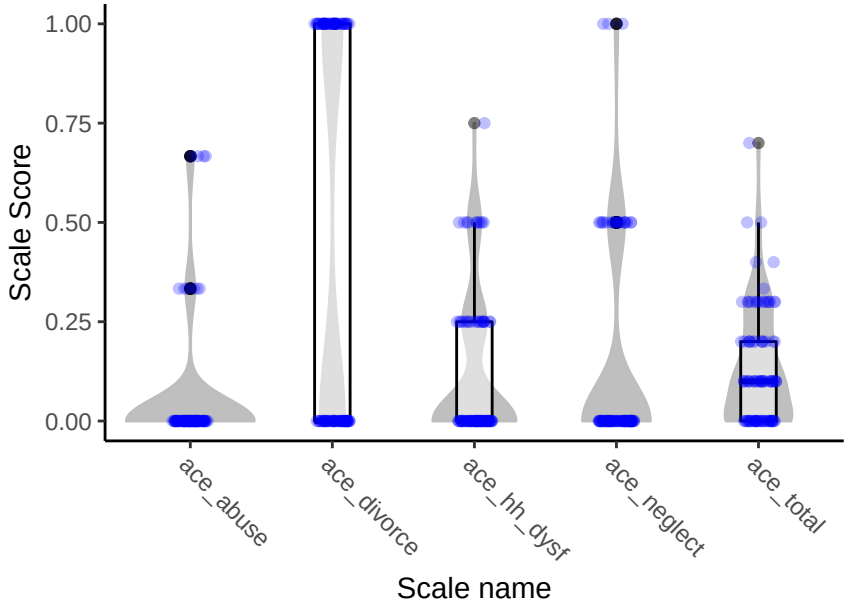
```
tds2_wave1_scored %>%
    ungroup() %>%
    distinct(scale_name) %>%
    knitr::kable()
```

scale_name
ACE
BFNE
BIS-15
Brief SCARED
BSSS
CARE-R Expected Involvement
CARE-R Social
CARE-R Willingness to Engage
CBCL
CES-DC
MSSSS
NTS
PAL-2
PDS
PEQ-R
SES
SPSRQ-S

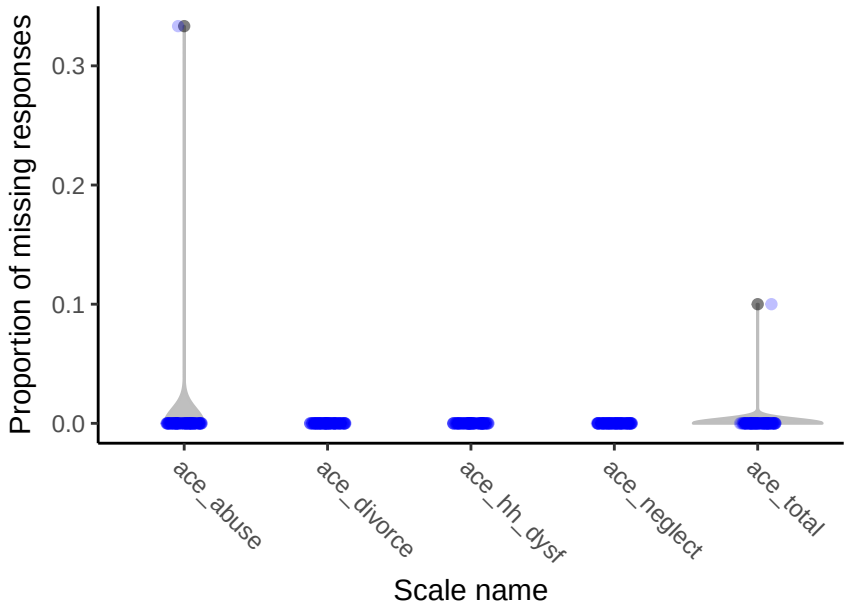
scale_name
UPPS-P
YRBS
RPI_part2
RSQ_part2

ACE

```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^ACE$',
                                  type = 'score')
```

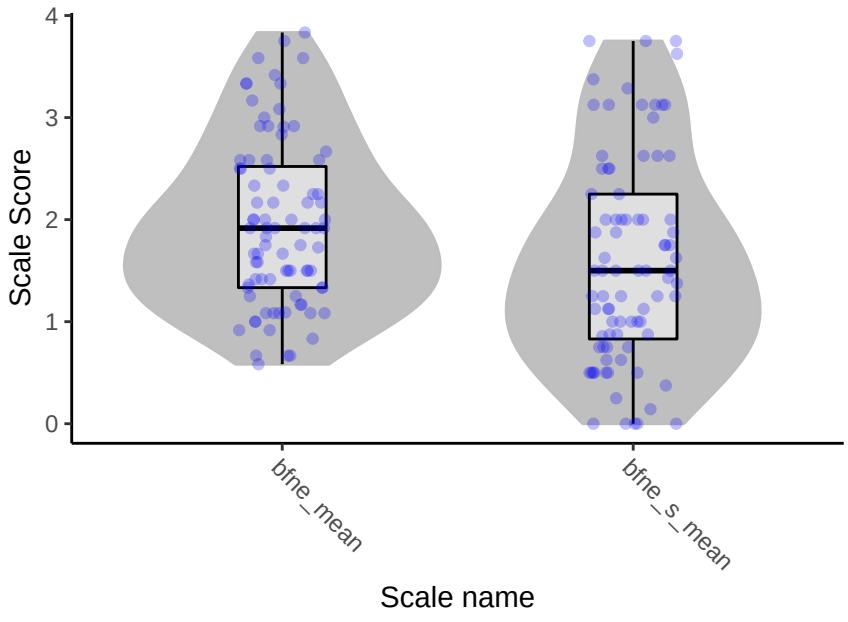



```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^ACE$',
                                  type = 'p_missing')
```

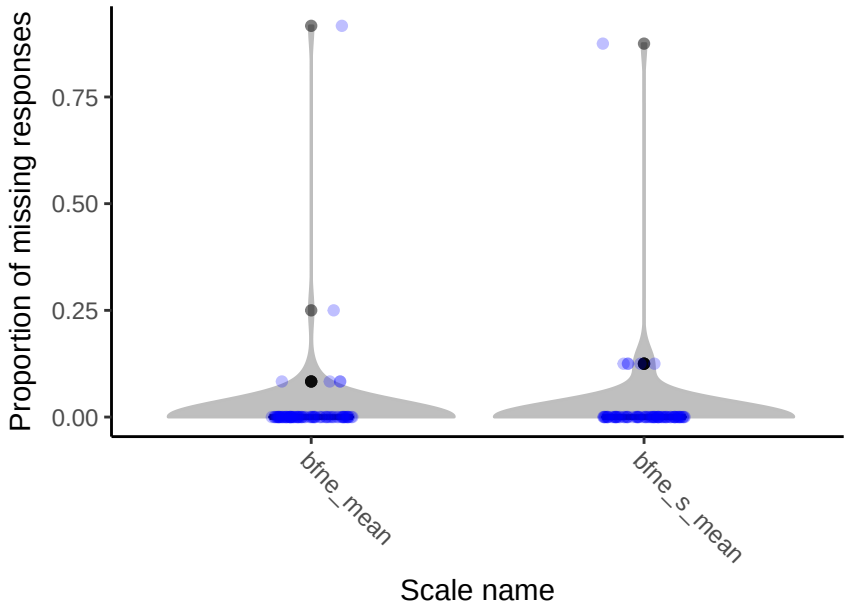


BFNE

```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^BFNE$',
                                  type = 'score')
```




```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^BFNE$',
                                  type = 'p_missing')
```

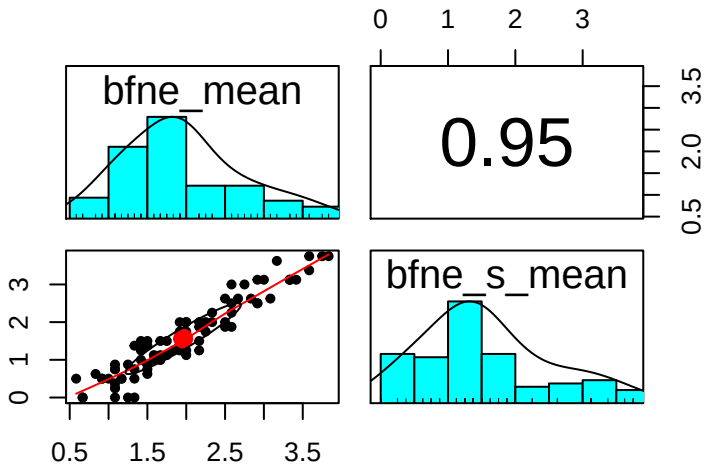



```
BFNERubric <- tds2_wave1_scoring_data_long %>%
    filter(scale_name == 'BFNE')

tds2_wave1_bfne_psych <- scorequaltrics::score_questionnaire(tds2_wave1_long_recoded_nodupes,
                                    BFNERubric,
                                    psych = T)
print(tds2_wave1_bfne_psych)
```

```
## Call: scoreItems(keys = key_list, items = dataDF_w)
##
## (Unstandardized) Alpha:
##       bfne_mean bfne_s_mean
## alpha      0.89        0.93
##
## Standard errors of unstandardized Alpha:
##       bfne_mean bfne_s_mean
## ASE       0.029       0.029
##
## Average item correlation:
##           bfne_mean bfne_s_mean
## average.r       0.4        0.64
##
##  Guttman 6* reliability:
##          bfne_mean bfne_s_mean
## Lambda.6      0.92        0.94
##
## Signal/Noise based upon av.r :
##              bfne_mean bfne_s_mean
## Signal/Noise       7.9          14
##
## Scale intercorrelations corrected for attenuation
##  raw correlations below the diagonal, alpha on the diagonal
##  corrected correlations above the diagonal:
##             bfne_mean bfne_s_mean
```

```
## bfne_mean        0.89        1.04
## bfne_s_mean      0.95        0.93
##
##  In order to see the item by scale loadings and frequency counts of the data
##  print with the short option = FALSE
```

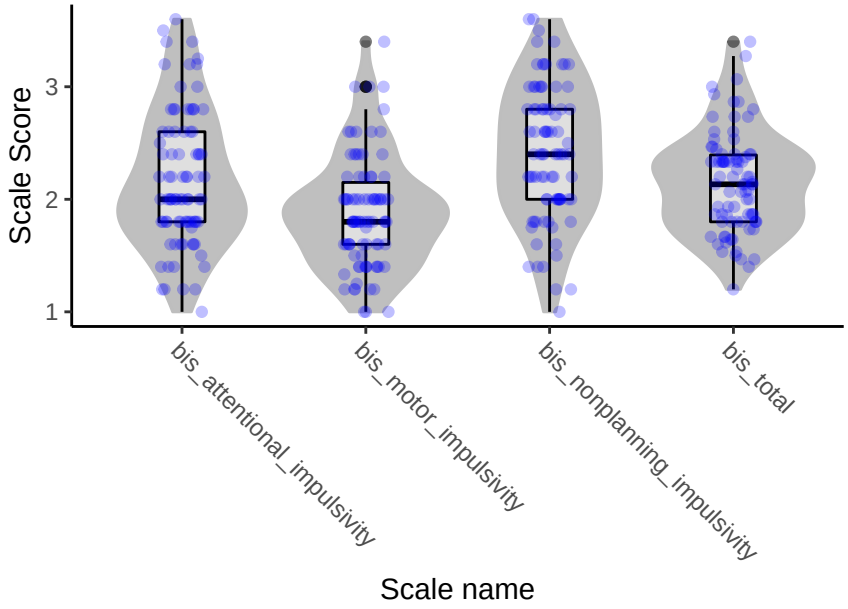
```
pairs.panels(tds2_wave1_bfne_psych$scores)
```



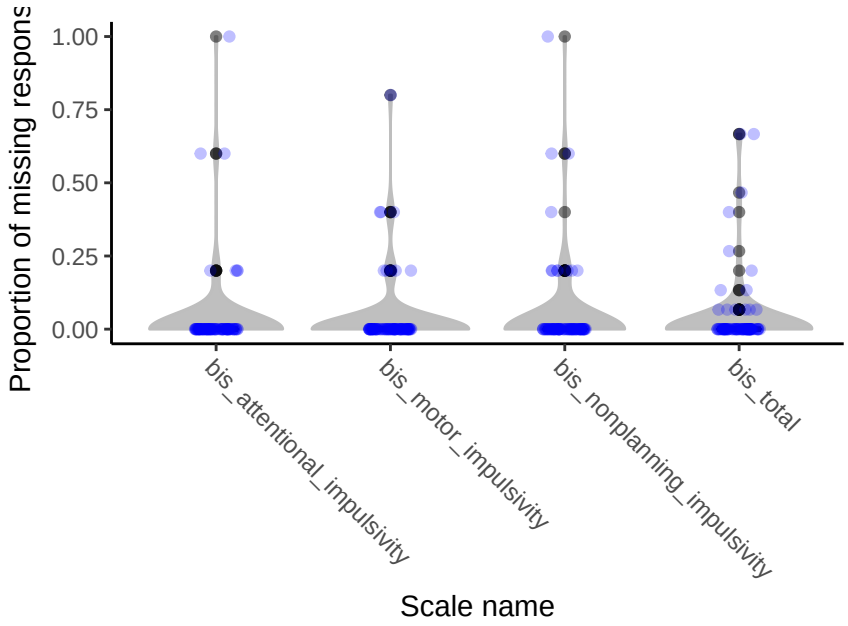
BIS-15

```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^BIS-15$',
                                  type = 'score')
```

```
## Warning: Removed 2 rows containing non-finite values (stat_ydensity).
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
## Warning: Removed 2 rows containing missing values (geom_point).
```



```
scorequaltrics::plot_scored_scale(tds2_wave1_scored,
                                  scale_regx = '^BIS-15$',
                                  type = 'p_missing')
```




```
BISRubric <- tds2_wave1_scoring_data_long %>%
    filter(scale_name == 'BIS-15')

tds2_wave1_bis_psych <- scorequaltrics::score_questionnaire(tds2_wave1_long_recoded_nodupes,
                                  BISRubric,
                                  psych = T)
print(tds2_wave1_bis_psych)
```

```
## Call: scoreItems(keys = key_list, items = dataDF_w)
##
## (Unstandardized) Alpha:
##       bis_motor_impulsivity bis_nonplanning_impulsivity
## alpha                   0.7                        0.71
##       bis_attentional_impulsivity bis_total
## alpha                        0.73      0.81
##
## Standard errors of unstandardized Alpha:
##       bis_motor_impulsivity bis_nonplanning_impulsivity
## ASE                   0.076                       0.075
##       bis_attentional_impulsivity bis_total
## ASE                          0.073     0.038
##
## Average item correlation:
##           bis_motor_impulsivity bis_nonplanning_impulsivity
## average.r                  0.32                       0.33
##           bis_attentional_impulsivity bis_total
## average.r                        0.35      0.22
##
##  Guttman 6* reliability:
##          bis_motor_impulsivity bis_nonplanning_impulsivity
```

```
## Lambda.6                   0.76                          0.77
##          bis_attentional_impulsivity bis_total
## Lambda.6                        0.78      0.87
##
## Signal/Noise based upon av.r :
##              bis_motor_impulsivity bis_nonplanning_impulsivity
## Signal/Noise                   2.4                         2.4
##              bis_attentional_impulsivity bis_total
## Signal/Noise                          2.6       4.2
##
## Scale intercorrelations corrected for attenuation
##  raw correlations below the diagonal, alpha on the diagonal
##  corrected correlations above the diagonal:
##                              bis_motor_impulsivity
## bis_motor_impulsivity                         0.70
## bis_nonplanning_impulsivity                   0.30
## bis_attentional_impulsivity                   0.43
## bis_total                                     0.71
##                              bis_nonplanning_impulsivity
## bis_motor_impulsivity                                0.42
## bis_nonplanning_impulsivity                          0.71
## bis_attentional_impulsivity                          0.40
## bis_total                                            0.76
##                              bis_attentional_impulsivity bis_total
## bis_motor_impulsivity                                0.61      0.95
## bis_nonplanning_impulsivity                          0.55      1.01
## bis_attentional_impulsivity                          0.73      1.06
## bis_total                                            0.81      0.81
##
##  In order to see the item by scale loadings and frequency counts of the data
##  print with the short option = FALSE
```

```
pairs.panels(tds2_wave1_bis_psych$scores)
```