An R Markdown program to analyse responses to a Discrete Choice Experiment (DCE) survey exploring the online help seeking preferences of socially anxious young people

Part 2: Describe Data

Matthew P Hamilton[1,*]

27 October 2022

[1] Orygen, Parkville, Australia

[*] Correspondence: Matthew P Hamilton <matthew.hamilton@orygen.org.au>

Suggested citation: "Matthew Hamilton (2022). dce_sa_analysis: An R Markdown program to analyse responses to a Discrete Choice Experiment (DCE) survey exploring online help seeking in socially anxious young people. Zenodo. https://doi.org/10.5281/zenodo.7223286. Version 0.0.9.1"

1 About this code

1.1 Purpose

This program generates descriptive statistics and plots from survey response data for a Discrete Choice Experiment study that is currently being written up. Future versions of this program will include details of the parent study.

1.2 Status

This code is the same version as was used to analyse study data. The only items that have been modified are those that remove references to the local directory structure on which study data was stored.

1.3 Use

This code can be run either by "knitting" the parent RMD file or by manually executing each code chunk. The code as is currently optimised for being knit in one step, which means that in a number of instance the value "Y" has been supplied to the `consent_1L_chr` argument of program functions. Supplying this value overrides the default behaviour of functions that write files which is to prompt users for their active consent prior to to write files and directories to their machine. If running this code interactively, we recommend omitting the `consent_1L_chr` argument as this will provide you with greater transparency about what files and directories are being written to your machine.

2 Prepare workspace

2.1 Install and load required libraries

If you do not already have the required libraries to run this program installed, you can do so by un-commenting and running the following lines.

```
# devtools::install_github("ready4-dev/ready4")
# devtools::install_github("ready4-dev/mychoice") # add lwgeom to imports
```

Next we load the libraries required to run this program.

```
library(ready4)
library(mychoice)
library(ready4use)
```

2.2 Specify data directories

We begin by specifying where our input data can be located and where we wish to write our outputs to. You must supply these details or the rest of this code will not work.

```
paths_ls <- list(input_data_dir_1L_chr = "PROVIDE DETAILS HERE",
                 output_data_dir_1L_chr = "PROVIDE DETAILS HERE",
                 raw_data_fl_nms_chr = "PROVIDE DETAILS HERE")
```

2.3 Reproducibility

We now set a seed to aid reproducibility.

```
set.seed(1001)
```

Having set the seed, it is now likely that if you run the syntax described in this document on your own installation of R you will get identical results to those reported in this document. However, if you do not, it may be that you have a different version of R, or of some of the packages that we used to produce this analysis. We therefore save a record of the software that we have on the machine used for this analysis so this can be made available for comparisons.

```
session_ls <- sessionInfo()
```

3 Create custom functions

We now create a number of functions that we will use in subsequent parts of this program.

3.1 Ingest additional data

4 Describe data

We begin by creating an object to store the descriptive statistics we generate.

```
descriptives_ls <- list()
```

We create a summary table of descriptive statistics.

```
descriptives_ls$summary_tb <- records_ls$ds_tb %>%
  make_smry_tb(var_metadata_tb = mdl_params_ls$candidate_predrs_tb, # Investigate alternative call formulations
               digits_1L_int = 2L)
```

We compare selected descriptive statistics from our sample with expected population values.

```
descriptives_ls$prpn_cmprsns_ls <- add_age_and_area_cmprsns(list(),
                                            ds_tb = records_ls$ds_tb,
                                            consent_1L_chr = "Y",
                                            write_to_1L_chr = paths_ls$Replication) %>%
  add_cut_pnts_cmprsn(cmprsn_nm_1L_chr = "SEIFA_cmprsn_tb",
                      ds_tb = records_ls$ds_tb,
                      grouping_var_nms_chr = c("der_SEIFA_Quartile", "SEIFA Disadvantage Quartile"),
                      expected_dbl = 25,
                      is_pc_1L_lgl = T) %>%
  add_cut_pnts_cmprsn(cmprsn_nm_1L_chr = "urban_cmprsn_tb",
                      ds_tb = records_ls$ds_tb,
                      grouping_var_nms_chr = c("der_urban", "Resides in urban area"),
                      expected_dbl = (make_sos_lup(write_to_1L_chr = paths_ls$Replication) %>% dplyr::pull(share_dbl))[1:2] * 100,
                      is_pc_1L_lgl = T)
```

We generate summary tables and plots of choice responses by attribute.

```
descriptives_ls$choice_smrys_ls <- make_choice_smrys(dce_design_ls,
                                                     mdl_params_ls = mdl_params_ls,
                                                     records_ls = records_ls,
                                                     text_size_1L_int = 7,
                                                     wrapping_1L_int = 20)
```

5 Write output files

We now save our output. If running this code interactively, we recommend omitting the `consent_1L_chr` argument.