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

Fit models

Matthew P Hamilton[1,*]

19 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."

1 About this code

1.1 Purpose

This program is used to fit choice models to a dataset from a Discrete Choice Experiment study that is currently being written up. Future versions of this program will include details of the parent study.

This program fits models that explore:

- how choice attributes affect the responses of our sample;
- the heterogeneity of these impacts within our sample;
- how individual respondent characteristics interacted with choice attributes; and
- whether we can identify distinct sub-groups based on preference profile.

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.

```r
library(ready4)
library(mychoice)
```

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.

```r
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.

```r
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.

```r
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 Fit choice models

We first create objects to log store all the models that we will fit and to log key modelling milestones.

```
mdls_ls <- list()
```

```
mdlng_log_ls <- list(session_ls = session_ls)
```

4.1 Model how Social Anxiety app attributes shape the choice behaviour of young people

In the first set of models we fit, we are only interested in exploring the role of the attributes of alternatives on participant choices and ignore respondent heterogeneity. We therefore begin by fitting a basic multinomial logit model. We create two versions of this model - one fitted using the gmnl package and the other using the mlogit package. The produce almost identical results, but each class of output has different distinctive methods defined for it, which means that each type has ease of use advantages over the other depending on the purpose to which the model will be applied.

```
mdls_ls <- add_choice_mdls(mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           records_ls = records_ls,
                           purpose_chr = "attributes")
```

4.2 Model the heterogeneity of young people's preferences for Social Anxiety app attributes

The next set of models to estimate are those that include random parameters to explore how preferences vary due to respondent heterogeneity.

We fit a mixed logit model that includes random parameters. At this stage we are looking at overall heterogeneity for each parameter and are not exploring the role of specific individual characteristics. Again we create two versions of this model - one with the gmnl package and the other with the mlogit package. We also fit a Generalized Multinomial Logit Model using the gmnl package.

```
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                            dce_design_ls = dce_design_ls,
                            mdl_params_ls = mdl_params_ls,
                            records_ls = records_ls,
                            purpose_chr = "heterogeneity")
```

```
## Estimating MIXL model
## Estimating GMNL model
```

4.3 Model innteractions between respondent characteristics and Social Anxiety app attributes

We next want to investigate the role of individual respondent characteristics on the coefficients for choice attributes.

Our first step is to explore which characteristics are influential for each choice feature. We do this by sequentially estimating models that interact all respondent characteristics with a specified choice feature. We then identify which respondent characteristics are significant below a specified threshold (in this case 0.05). We include a specified maximum number (in our case 3) of characteristics that were most frequently found to be significant predictors for two or more choice features in models that includes all choice features. For the final models, we excluded the characteristics that relate to respondents' participation in the survey as these are not something that could be meaningfully used in real world applications of the choice model.

```r
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           records_ls = records_ls,
                           exclude_chr = c("pilot participant","incomplete choice tasks", "time taken"),
                           max_concepts_1L_int = 3L,
                           min_threshold_1L_int = 2L,
                           purpose_chr = "interactions",
                           significant_at_1L_dbl = 0.05)
```

```
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
## Estimating MIXL model
```

4.4 Model Social Anxiety app preference based sub-groups of young people

We are also interested if it is possible to identify subgroups that are distinguishable based on the nature of their preferences. To do this we estimate Latent Class models.

4.4.1 Basic Latent Class Models

The first model we estimate seeks to identify two different preference based classes. The second adds respondent characteristic predictors, to identify what characteristics predict membership of which class.

```
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           include_int = 1:2,
                           records_ls = records_ls,
                           purpose_chr = "classes",
                           nbr_of_clss_1L_int = 2L)
```

```
## Estimating LC model
## Estimating LC model
```

4.4.2 Mixed-Mixed Multinomial Logit Model

The next group of models to be estimate add random parameters to the latent classes. We fit models with and without correlated random parameters.

```
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           include_int = 3,
                           records_ls = records_ls,
                           purpose_chr = "classes",
                           nbr_of_clss_1L_int = 2L,
                           iterlim = 500)
```

```
## Estimating MM-MNL model
```

```
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           include_int = 4,
                           records_ls = records_ls,
                           purpose_chr = "classes",
                           nbr_of_clss_1L_int = 2L,
                           iterlim = 500,
                           method = "bfgsr") # Method that produces fewest NANs
```

```
## Estimating MM-MNL model
```

4.5 Model Willingness to Pay of young people for distinct attributes of a Social Anxiety App

Our final group of models are designed to facilitate estimates of Willingness To Pay (WTP) values for Social Anxiety App attributes.

4.5.1 Scaled Multinomial Logit Model

We begin by estimating a Scaled Multinomial Logit Model model that we can use later to generate WTP estimates directly from WTP space.

```
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           include_int = 1,
                           records_ls = records_ls,
```

```r
                           purpose_chr = "wtp",
                           iterlim = 500,
                           method = "bhhh")
```

```
## Estimating SMNL model
```

4.5.2 Generalised Multinomial Logit Model

We next generate an GMNL model with fixed Cost parameters that allows WTP to vary across individuals. We fit models with and without correlated random parameters.

```r
mdls_ls <- add_choice_mdls(mdls_ls = mdls_ls,
                           dce_design_ls = dce_design_ls,
                           mdl_params_ls = mdl_params_ls,
                           include_int = 2:3,
                           records_ls = records_ls,
                           purpose_chr = "wtp")
```

```
## Estimating GMNL model
## Estimating GMNL model
```

5 Write output files

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

```r
paths_ls <- write_choice_mdlng_ws(paths_ls) # Not giving prompt.
```

```r
write_obj_with_prompt(mdlng_log_ls,
                      obj_nm_1L_chr = "mdlng_log_ls",
                      outp_dir_1L_chr = paths_ls$Replication,
                      consent_1L_chr = "Y")
```

```r
write_obj_with_prompt(mdls_ls,
                      obj_nm_1L_chr = "mdls_ls",
                      outp_dir_1L_chr = paths_ls$Models,
                      consent_1L_chr = "Y")
```