Complete study program to reproduce all steps from data ingest through to results dissemination for a study to map mental health measures to AQoL-6D health utility

Matthew P Hamilton[†,1,*] Caroline X Gao[†,1,2,3]

[†] These authors contributed equally to this work.

[1] Orygen, Parkville, Australia
[2] Centre for Youth Mental Health; University of Melbourne, Parkville, Australia
[3] School of Public Health and Preventive Medicine, Monash University, Clayton, Australia

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

1 About this code

This R code is the complete study program for all steps from data ingest, analysis and reporting for the study summarised here: https://doi.org/10.1101/2021.07.07.21260129.

When using this code it is important to note:

1) Some of the steps in this program involve interactivity - they generate a prompt that a user must respond to before proceeding. Therefore, **this code should be run step by step** (i.e run one chunk at a time and do not try to run the program by knitting the R Markdown version of this code). Although it would be possible to add work-arounds to the interactivity issue, running the program by knitting the RMD version is still not recommended as it will prevent the documents generated by this program from rendering properly.

2) The code in this program is highly abstracted which means that its detailed workings are not exposed. However, as all code is open source you are able to scrutinise the underlying code if you wish. The websites, manuals and repositories of the libraries used to run this program (those loaded by calls to the `library` function below) are a good starting point if you wish to do so.

2 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/ready4show)
# devtools::install_github("ready4-dev/ready4use)
# devtools::install_github("ready4-dev/youthvars)
# devtools::install_github("ready4-dev/scorz)
# devtools::install_github("ready4-dev/specific")
# devtools::install_github("ready4-dev/TTU")
```

Next load the libraries required to run this program.

```
library(ready4)
library(ready4show)
library(ready4use)
library(youthvars)
library(scorz)
library(specific)
library(TTU)
```

3 Ingest, transform, label and score data

3.1 Ingest data

To start, we need to specify the path to our input dataset. The setting below (empty) will allow you to run the program without requiring access to the original dataset (instead it will use a synthetic dataset that though entirely fake, closely resembles the study dataset). If you have access to the source dataset and wish to use it, you can specify the path to that data here.

```
path_to_data_1L_chr <- NA_character_
```

Based on the value specified above, we can now ingest either the real study dataset or the synthetic (fake) data.

```
if(!is.na(path_to_data_1L_chr)){
  ds_tb <- readRDS(path_to_data_1L_chr)
}else{
  ds_tb <- Ready4useRepos(dv_nm_1L_chr = "fakes",
                          dv_ds_nm_1L_chr = "https://doi.org/10.7910/DVN/HJXYKQ",
                          dv_server_1L_chr = "dataverse.harvard.edu") %>%
  ingest(fls_to_ingest_chr = c("ymh_clinical_tb"),
         metadata_1L_lgl = F)
}
```


3.2 Transform data

We perform any required pre-analysis dataset transformations (variable renaming, re-grouping data).

```
ds_tb <- ds_tb %>%
  transform_raw_ds_for_analysis()
```

3.3 Label data

We ingest a `Ready4useRecord` object containing a data dictionary for our dataset and some supplementary information about the candidate predictors that we will be exploring in our analysis.

```
A <- Ready4useRepos(dv_nm_1L_chr = "TTU",
                    dv_ds_nm_1L_chr = "https://doi.org/10.7910/DVN/DKDIBO",
                    dv_server_1L_chr = "dataverse.harvard.edu") %>%
  ingest(fls_to_ingest_chr = c("dictionary_r3","predictors_r3"))
```

We can now pair our dataset with its dictionary as a `Ready4useDyad`.

```
B <- Ready4useDyad(ds_tb = ds_tb,
                   dictionary_r3 = procureSlot(A,
                                               "b_Ready4useIngest",
                                               use_procure_mthd_1L_lgl = T,
                                               fl_nm_1L_chr = "dictionary_r3"))
```

We add the labels from the dictionary to the dataset.

```
B <- B %>%
  renew(type_1L_chr = "label")
```

We add some metadata about the structural properties of our dataset (variables for the unique record identifier and data collection round), converting our dataset into a `YouthvarsSeries` object.

```
B <- YouthvarsSeries(a_Ready4useDyad = B,
                     id_var_nm_1L_chr = "fkClientID",
                     timepoint_var_nm_1L_chr = "round",
                     timepoint_vals_chr = levels(procureSlot(B,
                                                             "ds_tb")$round))
```


3.4 Score health utility

We label the dataset as appropriate for deriving adolescent AQoL-6D scores by converting it to a `ScorzAqol6Adol` object. We then calculate these scores.

```
B <- ScorzAqol6Adol(a_YouthvarsProfile = B) %>%
  renew()
```

4 Describe and analyse data

4.1 Specify modelling parameters

We create a `SpecificModels` object.

```
C <- SpecificConverter(a_ScorzProfile = B) %>%
  metamorphose()
```

We add the required parameters (allowable range for utility scores, the variable names of candidate predictors to be explored in the primary analysis, candidate covariates, variables to use when preparing descriptive statistics, the data collection date-stamp variable and metadata on candidate predictors).

```
C <- renewSlot(C,
               "b_SpecificParameters@depnt_var_min_max_dbl",
                  c(0.03,1)) %>%
  renewSlot("b_SpecificParameters@candidate_predrs_chr",
               c("BADS","GAD7", "K6", "OASIS", "PHQ9", "SCARED")) %>%
  renewSlot("b_SpecificParameters@candidate_covars_chr",
               c("d_sex_birth_s", "d_age",  "d_sexual_ori_s", "d_studying_working",
                 "c_p_diag_s", "c_clinical_staging_s", "SOFAS")) %>%
  renewSlot("b_SpecificParameters@descv_var_nms_chr",
                c("d_age","Gender","d_relation_s",
                  "d_sexual_ori_s", "Region", "d_studying_working",
                  "c_p_diag_s", "c_clinical_staging_s","SOFAS")) %>%
  renewSlot("b_SpecificParameters@msrmnt_date_var_nm_1L_chr",
                  "d_interview_date") %>%
    renewSlot("b_SpecificParameters@predictors_lup",
               procureSlot(A,
                           "b_Ready4useIngest",
                           use_procure_mthd_1L_lgl = T,
                           fl_nm_1L_chr = "predictors_r3"))
```

We confirm that our parameters and dataset are internally consistent. This step will also rename dataset variables that use certain incompatible naming conventions.

```
C <- ratify(C)
```

4.2 Create local workspace

We add details of the directory where data generated by our analysis will be written. Note, if using real data **this must be a secure location** as copies of the dataset will be saved in this directory.

```
C <- renewSlot(C,
                  "paths_chr",
                  "Data") %>%
  renewSlot("b_SpecificParameters@fake_1L_lgl",
               is.na(path_to_data_1L_chr))
```

Create sub-directories of the nominated output data directory.

```
C <- author(C,
               what_1L_chr = "workspace")
```

4.3 Generate descriptive statistics

We now generate tables and charts that describe our dataset. These are saved in a sub-directory of our output data directory.

```
C <- author(C,
            what_1L_chr = "descriptives",
            digits_1L_int = 3L)
```

4.4 Undertake exploratory modelling using baseline dataset

4.4.1 Identify preferred model types

We next compare the performance of different model types. This step saves model objects and plots to a sub-directory of our output directory and outputs a `SpecificPredictors` object.

```
D <- investigate(C,
                 depnt_var_max_val_1L_dbl = 0.99,
                 session_ls = sessionInfo())
```

After inspecting the output of the previous command, we can now specify the preferred model types to use from this point onwards.

```
D <- renew(D,
           new_val_xx = c("GLM_GSN_LOG", "OLS_CLL"),
           type_1L_chr = "results",
           what_1L_chr = "prefd_mdls")
```

4.4.2 Identify preferred predictors and covariates

Next we assess multiple versions of our preferred model type - one single predictor model for each of our candidate predictors and the same models with candidate covariates added. The output is a `SpecificFixed` object, plus a number of model/plot objects saved locally.

```
E <- investigate(D)
```

4.4.3 Assess performance of final model specification

After reviewing the output of the previous step, we specify the covariates we wish to add to the models.

```
E <- renew(E,
           new_val_xx = "SOFAS",
           type_1L_chr = "results",
           what_1L_chr = "prefd_covars")
```

We now assess the multivariate models. The output is a `SpecificMixed` object and more locally saved model/plot objects.

```
F1 <- investigate(E)
```

4.5 Undertake longitudinal modelling

We next reformulate the models we finalised in the previous step so that they are suitable for modelling longitudinal change.

4.5.1 Primary analysis

For our primary analysis, we use the longitudinal formulation of the models we previously selected. As with previous steps the output of this analysis is stored in the `F1` object and in files written to the local output data directory.

```
F1 <- investigate_SpecificMixed(F1)
```

4.5.2 Secondary analyses

For our secondary analyses, we specify alternative combinations of predictors and covariates.

```
scndry_anlys_params_ls <- make_scndry_anlys_params(candidate_predrs_chr = c("SOFAS"),
                                                   candidate_covar_nms_chr = c("d_sex_birth_s",
                                                                               "d_age",
                                                                               "d_sexual_ori_s",
                                                                               "d_studying_working"),
                                                   prefd_covars_chr = NA_character_) %>%
  make_scndry_anlys_params(candidate_predrs_chr = c("SCARED","OASIS","GAD7"),
                           candidate_covar_nms_chr = c("PHQ9", "SOFAS",
                                                       "d_sex_birth_s",
                                                       "d_age",
                                                       "d_sexual_ori_s",
                                                       "d_studying_working"),
                           prefd_covars_chr = "PHQ9")
F1 <- investigate_SpecificMixed(F1,
                                scndry_anlys_params_ls = scndry_anlys_params_ls)
```

5 Report and disseminate findings

5.1 Create shareable models

The model objects created and saved in our working directory by the preceding steps are not suitable for public dissemination. They are both too large in file size and, more importantly, include copies of our source dataset. We can overcome these limitations by creating shareable versions of the models.

Two types of shareable version are created - copies of the original model objects in which fake data overwrites the original source data and tables of model coefficients.

```
F1 <- authorData(F1)
```

5.2 Specify study reporting metadata

We create a `SpecificSynopsis` object that contains the fields necessary to render and share reports.

```
G <- metamorphose(F1)
```

We add metadata relevant to the reports that we will be generating to these fields.

```
G <- G %>%
  renewSlot("authors_r3",
            {
              if(is.na(path_to_data_1L_chr)){
                ready4show::authors_tb
                }else{
                  read.csv("CSVs/Authors.csv") %>%
                    tibble::as_tibble() %>%
                    ready4show_authors()
                  }
                }
            ) %>%
        renewSlot("institutes_r3",
                                {
              if(is.na(path_to_data_1L_chr)){
                ready4show::institutes_tb
                }else{
                  read.csv("CSVs/Institutes.csv") %>%
                    tibble::as_tibble() %>%
                    ready4show_institutes()
                  }
                }) %>%
  renewSlot("digits_int", c(3L,3L)) %>%
  renewSlot("outp_formats_chr", c("PDF","PDF")) %>%
  renewSlot("title_1L_chr",
            ifelse(is.na(path_to_data_1L_chr),
                   "A hypothetical study using fake data",
                   "Mapping psychological distress, depression and anxiety measures to AQoL-6D utility using data from a sample of young p
```

```
            ) %>%
  renewSlot("correspondences_r3",
            old_nms_chr = c("PHQ9", "GAD7"),
            new_nms_chr = c("PHQ-9", "GAD-7"))
```

Add details of the repository to which we will write copies of (non-confidential) study outputs. The below logic uses the same repository that stores our input data and only adds these details if real data is being used. You can ammend this section as appropriate (for example, if you wish to use a different repository [necessary if you don't have write permissions to `A`]).

```
if(!is.na(path_to_data_1L_chr)){
    G <- G %>%
      renewSlot("e_Ready4useRepos",
                procureSlot(A,
                        "a_Ready4usePointer@b_Ready4useRepos"))
}
```

5.3 Describe and share models

We now create a `TTUReports` object, that we can use to efficiently retrieve and apply programs that can summarise our study results.

```
H <- TTUReports(a_SpecificSynopsis = G)
```

5.3.1 Author model catalogues

We download a program for generating a catalogue of models and use it to summarising the models created under each study analysis (one primary and two secondary). The catalogues are saved locally.

```
author(H,
       download_tmpl_1L_lgl = T,
       what_1L_chr = "Catalogue")
```

5.3.2 Share model catalogue

We share the catalogues that we created, uploading a copy to our study online repository. To run this step you will need write permissions to the online repository.

```
shareSlot(H,
          "a_SpecificSynopsis",
          type_1L_chr = "Report",
          what_1L_chr = "Catalogue")
```

5.3.3 Share models

We share tables of coefficients and other meta-data about the models we have created by posting them to the online repository. The object we create and share is designed to be used in conjunction with the `youthu` package to make it easier to make predictions with these models using new data. Again, you will need write permissions to the online repository.

```
shareSlot(H,
          "a_SpecificSynopsis",
          type_1L_chr = "Models",
          what_1L_chr = "ingredients")
```

5.4 Author manuscript

We add some content about the manuscript we wish to author.

```
G <- procureSlot(H,
                 "a_SpecificSynopsis") %>%
  renewSlot("background_1L_chr",
            "BACKGROUND TEXT GOES HERE") %>%
  renewSlot("coi_1L_chr", "None declared") %>%
  renewSlot("conclusion_1L_chr",
            "CONCLUSION TEXT GOES HERE") %>%
  renewSlot("ethics_1L_chr",
            paste0("The study was reviewed and granted approval by ",
                   ifelse(is.na(path_to_data_1L_chr),
                          "no-one.",
                          "the University of Melbourne's Human Research Ethics Committee and the local Human Ethics and Advisory Group (16
                   )) %>%
  renewSlot("funding_1L_chr",
            paste0("The study was funded by ",
                   ifelse(is.na(path_to_data_1L_chr),
                          "no-one.",
                          "the National Health and Medical Research Council (NHMRC, APP1076940), Orygen and headspace."))) %>%
  renewSlot("interval_chr",
            "three months") %>%
  renewSlot("keywords_chr",
            c("anxiety", "AQoL","depression",
              "psychological distress", "QALYs", "utility mapping")) %>%
  renewSlot("sample_desc_1L_chr",
            ifelse(is.na(path_to_data_1L_chr),
```

```
                    "The study sample is fake data.",
                    "")
            )
H <- renewSlot(H,
               "a_SpecificSynopsis",
               G)
```

We add arguments to help us create an abstract for the the manuscript.

```
H <- H %>%
  renewSlot("a_SpecificSynopsis@abstract_args_ls",
            manufactureSlot(H,
                            "a_SpecificSynopsis",
                            what_1L_chr = "abstract_args_ls"))
```

We create a summary of results that can be interpreted by the program that authors the manuscript.

```
H <- enhanceSlot(H,
                 "a_SpecificSynopsis",
                 depnt_var_nms_chr = c("AQoL-6D",
                                       "Adolescent AQoL Six Dimension"),
                 with_1L_chr = "results_ls")
```

We create and save the plots that will be used in the manuscript.

```
composite_1_plt <- depictSlot(H,
                              "a_SpecificSynopsis",
                              depnt_var_desc_1L_chr = "AQoL-6D",
                              timepoint_old_nms_chr = procureSlot(H,
                                                                  "a_SpecificSynopsis@d_YouthvarsProfile@timepoint_vals_chr"),
                              timepoint_new_nms_chr = c("Baseline","Follow-up"),
                              what_1L_chr = "composite_mdl",
                              write_1L_lgl = T)
```

```
composite_2_plt <- depictSlot(H,
                              slot_nm_1L_chr = "a_SpecificSynopsis",
                              what_1L_chr = "composite_utl",
                              write_1L_lgl = T)
```

We download a program for generating a template manuscript and run it to author a first draft of the manuscript.

```
author(H,
       download_tmpl_1L_lgl = T,
       what_1L_chr = "Manuscript_Auto")
```

We can now customise the template program so that it can generate the manuscript we plan on submitting. It is recommended that the updated manuscript generation program is saved in a new sub-directory, for example "Manuscript_Submission". Once we have done this (a step undertaken manually, not by this algorithm) we can the following command to author the submission manuscript.

```
if(paste0(H@a_SpecificSynopsis@a_Ready4showPaths@outp_data_dir_1L_chr,
          "/",
          H@a_SpecificSynopsis@a_Ready4showPaths@mkdn_data_dir_1L_chr,
          "/Manuscript_Submission") %>%
   dir.exists()){
  author(H %>%
           renewSlot("a_SpecificSynopsis@tables_in_body_lgl",
                     F) %>%
           renewSlot("a_SpecificSynopsis@figures_in_body_lgl",
                     F),
         what_1L_chr = "Manuscript_Submission")
}
```


6 Tidy workspace

The preceding steps saved multiple objects (mostly R model objects) that have embedded within them copies of the source dataset. We can now purge all such copies from our output data directory.

```
author(C,
       type_1L_chr = "purge_write")
```