Censored Likelihood Multiple Imputation in R

Jonathan Boss

May 20, 2019

Installing the `clmi` package

Installation of the `clmi` package requires the `install_github` function in the `devtools` package. If you have not downloaded the `devtools` package then run `install.packages("devtools")`.

```
library(devtools)

install_github("bossjona/Single-Pollutant-Multiple-LODs")
```

Load the `clmi` package and an example dataset

For convenience we have included a example dataset called `toy.data`, which can be loaded by running `data("toy-example")`. Let's look at the first 10 entries of the example dataset.

```
library(clmi)

data("toy-example")

head(toy.data, n = 10)
```

```
##       id case_cntrl      poll smoking gender batch1
## 1  13707          1  3.588607       0      1      0
## 2  18641          1        NA       0      0      0
## 3  27407          1  2.619124       1      0      0
## 4  45462          1  7.203193       0      1      1
## 5  50357          1  7.336160       1      1      1
## 6  59168          1        NA       0      0      0
## 7  61477          1  5.136974       0      1      0
## 8  76585          1 11.794483       1      1      0
## 9  80681          1  1.280289       0      0      1
## 10 84391          1  5.480510       1      1      0
```

The `id` column gives the study ID and is unimportant for the purposes of this example. The `case_cntrl` takes values 0 or 1, where 1 indicates that the subject has the disease of interest and 0 indicates that the subject is a healthy control. The column `poll` is the environmental exposure of interest, where `NA` indicates that the concentration is below the limit of detection (LOD). The variables `smoking` and `gender` are covariates that we are going to include in the imputation model. Lastly, `batch1` takes two values; 1 if the subject's biosample was assayed in batch 1 and 0 if the subject's biosample was assayed in batch 2. Moreover, the LOD for batch 1 is 0.8 and the LOD for batch 2 is 0.65.

Implementing Censored Likelihood Multiple Imputation

The function that performs censored likelihood multiple imputation is the `clmi` function. For details see `help(clmi)`.

```
clmi.out <- clmi(df = toy.data, contaminant = "poll", batch = "batch1",
                 outcome = "case_cntrl", contam.covars = c("smoking", "gender"),
```

```
                  lod.info = data.frame(batch.info = c("1","0"), lod = c(0.8, 0.65)),
                  n.imps = 20, seed = 12345, t.function = function(x) log(x))
```

Some important things to notice:

- The `contaminant`, `batch`, `outcome`, and `contam.covars` arguments must correspond to the variable names in `toy.data`.
- The `lod.info` argument is where all of the LOD information goes. The first column must be named `batch.info` and contains all levels of the `batch1` variable as character strings. The second column must be named `lod` and contains the LODs corresponding to each batch. Remember that all of the values below the LOD must be entered as `NA` so that the function recognizes them as missing.
- The `t.function` argument specifies whether you want to return an imputed concentrations on the untransformed scale (default) or on a transformed scale. It is often the case that contaminant concentrations are log-transformed to handle right skewed exposure data while still maintaining interpretable effect estimates. To impute on the log-scale specify, `t.function = function(x) log(x)`. As another example, if you want to impute on the square root scale then specify, `t.function = function(x) sqrt(x)`.

The imputed datasets can be extracted as a list using `$imputed.dfs`:

```
extract.imputed.dfs <- clmi.out$imputed.dfs
```

Fit and pool outcomes models

The `pool.clmi` function takes the output generated by the `clmi` function, fits outcome models on each of the imputed datasets, and pools inference across outcome models. For details see `help(pool.clmi)`.

```
results <- pool.clmi(clmi.out = clmi.out, regression.type = "logistic",
                     outcome.covars = NULL)
```

A few things that deserve further comment:

- There are currently two options for the `regression.type` argument. If you have binary outcome data (as in the current example) use `regression.type = "logistic"` so that the outcome models fit on the imputed datasets are logistic regression models. If you have continuous outcome data use `regression.type = "linear"` so that the outcome models fit on the imputed datasets are linear regression models.
- The `outcome.covars` argument is a vector of strings containing the variable names corresponding to the precision variables that you want to include as adjustment covariates in the outcome models. Note that the strings must correspond to variable names in your dataset. Variables included in the `contam.covars` when running the `clmi` function are automatically adjusted for. Therefore, `outcome.covars` should only contain additional adjustment covariates that were not included in the imputation model. In this case `outcome.covars = NULL` means that we only want an single-pollutant outcome model that adjusts for the variables in `contam.covars`, i.e. `smoking` and `gender`.

To display the pooled results use `$output`:

```
results$output
```

```
##                              est         se       df     p.values
## (Intercept)             0.36326915 0.07548992 91.98337 5.826613e-06
## smoking                -0.08165125 0.11788361 93.48825 4.902506e-01
## gender                  0.18971427 0.11566557 91.80907 1.043863e-01
## poll_transform_imputed  0.09303489 0.04861608 89.73839 5.885053e-02
##                            LCL.95    UCL.95
```

```
## (Intercept)             0.213339286 0.5131990
## smoking                -0.315728622 0.1524261
## gender                 -0.040013900 0.4194424
## poll_transform_imputed -0.003553288 0.1896231
```

If you want to look at the individual regressions fit on each imputed dataset use `$regression.summaries`

```
results$regression.summaries
```

Contact information

If you would like to report a bug in the code, ask questions, or send requests/suggestions e-mail Jonathan Boss at `bossjona@umich.edu`.