DIME Analytics
Peer Code Review - Sampling and Randomization
v1.0

Reviewer:

Coder:

NOTE: Make sure that fill out this checklist **ONLY IF** your partner's submission includes **sampling and/or randomization** tasks.

Sampling and Randomization tasks

This checklist lists important factors to consider while reviewing your code review partner's **sampling/randomization** code. Please fill this checklist, and submit it as an attachment when you submit this detailed form.

Note: We are piloting this checklist as part of improving the code review process. The questions in this list are therefore **not required**, and we welcome your feedback on the points listed below.

Sampling:

The code clearly identifies the population eligible for sampling. (This should be in the form of an original dataset that is fixed, unless sampling is continuous or ongoing due to research design; in that case, this should be clearly described)

The population dataset is uniquely identified.

The population dataset contains all variables used in the sampling process, such as clusters and strata.

Advanced check: Clusters, strata, and individual-level IDs do not contain unmasked information about the identity of the clusters, strata, or individuals.

Advanced check: The original dataset should be checks for stability when loaded, using some combination of commands like `datasignature`, file hashing, `assert`, and/or `isid, sort` (or an equivalent in other coding languages.)

The method and rationale for random processes is described in documentation and understandable in code, including:

The overall sampling strategy is clearly documented, such as any characteristics or levels of observation that affect probabilities of inclusion/exclusion from sample.

Handling of strata and clusters are implemented as described in the research design, for example, using the procedures described by `randtreat` in Stata (or equivalent).

Handling of unequal strata and cluster sizes is clearly specified, appropriate, and justified.

Probability weights are calculated and stored for each unit, when required.

Random processes-generating samples are implemented in a reproducible script, including:

Version settings are explicitly set according to software requirements (in Stata, the `version` or `ieboilstart` commands)

Seed is set using a unique, random seed generated from an external source (such as random.org, for each random process that is intended to be independent.

Population dataset is sorted uniquely before each independent random process.

The script outputs a resulting dataset of results of the random processes for each potential sampling group (or treatment arm)

If the sampling result is intended to be "finalized" for field use, there is a logic switch to ensure that final data cannot be overwritten by new data. For example:

```
if ‘replace_results‘ == 1 save ‘results‘ , replace
else di as err "Results not saved: To replace results, toggle logic switch.
```

The output dataset is created with corresponding codebooks, value labels, and variable labels describing all results of the random processes.

The final sampling results are stable and reproducible

Randomization:

The project uses a script to randomize.

If not, the reason is fully documented.

The code sets the version of Stata before randomizing.

No new observations are created in the randomization section.

The data is sorted before randomizing.

If yes, the data is sorted on a variable that is uniquely identifying.

The code uses `set seed` before the sorting.

If yes, the seed is set using a unique, random seed generated from an external source (such as random.org, for each random process that is intended to be independent.

The code creates a categorical variable to identify the treatment and control groups.

The randomization code is reproducible and consistent across multiple runs. For more information, you can refer to this resource on iedorep.