DIME Analytics
Peer Code Review Checklist: Stata

Reviewer Details

Reviewer Name: Coder Name:

The following checklist outlines best practices for writing and reviewing Stata code. Items/Sections marked with an asterisk * are strongly recommended to ensure reproducibility.

Main Do-file Setup*

- [ ] Sets core configurations (`version`, `matsize`, `varabbrev`) directly or via a wrapper (e.g., `ieboilstart`).
- [ ] Script runs from start to end after changing directory paths in one place only.
- [ ] Uses only **relative paths** (no `C:/...`).
- [ ] Uses forward slashes in file paths for OS compatibility.
- [ ] Installs required packages or includes an `ado` folder with dependencies.
- [ ] Sets a **random seed** for reproducibility.
- [ ] The main do-file runs all code files (using `run` or `do` to files) without any need to manually run files in a certain order.

Data Management

- [ ] Dataset includes a **unique ID** and is sorted.
- [ ] The same unique ID is used consistently across datasets that share the same unit of observation.
- [ ] *Duplicate resolution is stable (does not use `duplicates drop, force`).
- [ ] Does not include PII or sensitive information.
- [ ] All variables are clearly labeled.
- [ ] Value labels are consistent (e.g., avoiding cases where `varA: 1 = yes, 0 = no` but `varB: 1 = yes, 2 = no`).
- [ ] Extended missing values are used where applicable (e.g., `.d` for Do not know, `.r` for Refuse to answer, etc.).

*Sorting is consistently and uniquely enforced using `sort` or `gsort` before commands that depend on it.

Avoids saving intermediate datasets unless needed for later use (uses `tempfile` when appropriate).

*Saves final dataset only once, avoids repeated overwriting.

Follows tidy data principles: one row per observation, one column per variable, and one unit of observation per `.dta` file (e.g., avoid wide-format household member data in a household-level file).

Avoids interactive commands (`edit`, `browse`).

Data Types & Variables

String variables are only used when necessary (e.g., proper nouns or alphanumeric IDs).

Converts categorical strings into labeled numeric variables (e.g., using `encode`).

Date variables are stored in proper date formats (e.g., `%td`, `%tm`).

Merge Checks

*No `m:m` merges used.

Mismatches or dropped observations are explained using `tab _merge` and `assert` checks.

If any observations are dropped, a clear justification is provided in the code.

Append Checks

Variables being appended are of the same type and structure.

Avoids `append, force`.

Any new variables introduced in appended datasets are properly handled.

Ensures that the resulting dataset remains uniquely identifiable, either with the original ID or a new combination of variables after the append.

Code Readability & Style

Uses proper indentation inside loops or programs.

Uses white space and line breaks (`///`) for long lines.

Uses descriptive index names in loops/globals.

Uses `${}` syntax for global macros.

Comments clearly explain steps and analysis decisions.

Each section is clearly marked (e.g., `*** SECTION: Construct Outcomes`).

Avoids hardcoding values (uses macros).

Avoids copy-pasting blocks; uses loops or programs where repetitive code appears.

Variable Construction

Each variable's logic aligns with the codebook or documentation.

Transformations (log, winsorize, unit-standardization, etc.) are justified and explained.

Categorical variables are properly labeled and encoded.

Data transformations are verified with `assert` or summary checks.

Collapse / Group-wise Calculations

*Data is sorted uniquely before using `by: egen` or `by: gen`.

Aggregations (e.g., using `collapse`, `egen`, or group-level calculations) are correct and clearly documented.

Missing values are handled appropriately during `collapse` and `egen`.

Output & Logging

*Outputs are not copied manually to external files. Instead, they are exported using commands like `esttab`, `outreg`, `asdoc`, `graph export`, among others.

Output files are clearly named and saved in dedicated folders.

Log files are started with `log using` and closed with `log close`.

*Tables are saved in plain text formats (e.g., `.csv`, `.txt`, `.tex`) to ensure compatibility with Git and facilitate version control.

*Export commands include the `replace` option to prevent errors if output files already exist.

Reproducibility & Documentation*

Code runs reproducibly from a fresh Stata session.

README documents required Stata version and packages.

`ieboilstart` or equivalent ensures version stability.

Folder and file structure is documented.

README specifies the main do-file and highlights which line(s) to update to run the code.