Codebook for Replication Data for: *Crossing the Boundaries: An Implementation of Two Methods for Projecting Data across Boundary Changes*

Max Goplerud*

September 30, 2015

This provides an overview of the replication materials for *Crossing the Boundaries: An Implementation of Two Methods for Projecting Data across Boundary Changes*. Detailed information on the sources of the electoral data used is provided in Appendix A.[1] An initial point notes that much of the analysis reported here was done using Harvard's Remote Computing Environment and its CONDOR system that allows 'batch processing' of embarassingly parallel tasks. This radically increased the speed at which the results could be calculated though, in theory and using the code provided, all results could be calculated using other means. Parts of the code thus require some modification to run on other systems. I attempt to sketch possible modifications in this codebook. This mostly affects the simulations and Appendix C concerning the timing of the interpolations. One final note is that the directory structure of the data as stored when performing the analysis is reflected in the code; thus, to use the code as written, one should put each of the country-specific empirical folders into the main working directory. The code will call them using the relative paths. If these are re-arranged, it may require modifying parts of the code where the data is loaded.

The data uploaded consists of a number of compressed files (tar.gz),[2] which should ideally all be unpacked into the same main directory:

`replication_code.tar.gz` contains the key `R` code to replicate the main results and the interpolation function. It also contains aggregate versions of the data needed to replicate the main results. Finally, it contains copies of the three supporting documents (Appendices A, B, and C).

`australia.tar.gz, canada.tar.gz, germany.tar.gz`, `new_zealand.tar.gz`, `us.tar.gz` contain all of the data needed to replicate the empirical results for each country. All data used in these files should be cited as noted in Appendix A. Further, as noted in Appendix A, the British data has a restricted license and thus cannot be uploaded to the Dataverse. Appendix A provides further details on how to access the data, but please contact me with any specific questions.

`condor.tar.gz` contains all of the results from the simulations as well as the timings reported in Appendix C. It further contains the CONDOR specific code and submit files.

The remainder of this codebook outlines the various chunks of `R` code included in the replication data.

- `interpolation_programme.R` — This file contains the functions needed to run the interpolation algorithm. The arguments of the function are outlined below. Using 'source' on this file provides a number of auxiliary functions, also outlined below, that shouldn't need to be called directly. Work is on-going to turn this into a proper R package available through CRAN.

  Required Arguments of `interpolation`:

*Codebook Version 1: 30 Sept 2015. Contact Details: goplerud [at] g.harvard.edu

[1] More detailed code on how the electoral data was cleaned and merged into the relevant shapefiles is available upon request. The shapefiles and data to replicate the analysis exactly are included in the Dataverse.

[2] Windows users can use 7-zip to open these files.

1. `old.shp` - This is the shapefile containing the old constituencies (boundaries) and the data associated with those boundaries
2. `old.name` - This is the name of a variable in `old.shp` that uniquely indexes the old boundaries.
3. `new.shp` - This is the shapefile containing the new constituencies (boundaries).
4. `new.name` - This is the name of a variable in `new.shp` that uniquely indexes the new boundaries.
5. `pop.shp` - This is the shapefile that contains the population data to be used in the Dasymetric Interpolation (DI) method.

   If such population data does not exist, this should be set to `NULL`. In this case, a 'fake' set of population data is created from the old boundaries where each unit is assigned the same population of '1'. In this case, the output of the function will still include data that nominally comes from the DI method, however, it should be identical to the data from the AW method.
6. `pop.name` - This is the name of the variable in `pop.shp` that contains the population data. If `pop.shp` is set to `NULL`, this should also be set to `NULL`.
7. `raster.size` - This defines the size in pixels of one side of the square raster whose overall dimensions are `raster.size` by `raster.size`.

Optional Arguments of `interpolation`. Default value listed.

1. `prime.numbers` - 10^6 - This is the number that is fed into the Sieve of Eratosthenes to generate prime numbers. It generates approximately 78,500 prime numbers to index the relevant shapefiles. The function will through an error if there are not enough prime numbers generated. One can increase this parameter but I would suspect a better solution in that case is to break the shapefiles into smaller pieces and interpolate them separately at lower resolutions.
2. `pfx.di` - `di` - This is the prefix that is appended to all variables created using the Dasymetric Interpolation ('DI') method, i.e. the method incorporating population data.
3. `pfx.aw` - `aw` - This is the prefix that is appended to all variables created using the Areal Weighting ('AW') method, i.e. the method using only the physical overlap of the constituencies.
4. `manual.extent` - `NULL` - If one wishes to specify the extent of the raster manually, rather than relying on the extents of the shapefiles in question, the dimensions can be set here.
5. `verbose` - 1 - Can be set to 0, 1, or 2 and displays increasing levels of information about which stage of the interpolation programme is underway.
6. `use.gdal` - `TRUE` - An option stating whether the interpolation programme should use the `gdal` package or rely on the default `raster` package. As Appendix C shows, `gdal` is much faster and thus this is selected by default. If `gdal` is not installed properly, the function will throw an error.
7. `old.data` - `NULL` - The data to be interpolated from the old shapefile to the new shapefile. The default option means that all variables in the old shapefile are interpolated. Specifically, this means that all numeric and integer variables are interpolated whilst all factor or character variables are returned unmodified in the interpolation output. One can pass a data.frame to `old.data` to only interpolate those variables. NB: The row order of this data frame must exactly match the row order of the shapefile to avoid incorrect interpolations. The easiest way to use this option would be to subset the shapefile's data.frame directly, i.e. `old.shp@data[...]`.

8. `temp.dir` - `NULL` - If `use.gdal` is set to `TRUE`, this option determines where the temporary rasters and shapefiles are saved. They are all removed at the end of the process but this allows for a greater degree of control. The default option (`NULL`) simply saves them into the default temporary directory in `R`.
9. `repair.pop` - 99 - This is the percentage threshold at which the repair procedure is applied. If fewer than `repair.pop` percent of wards are rasterised, then the function attempts to 'repair' the interpolation by assigning the population in the unrasterised wards to the closest rasterised ward. Ideally, one will set `raster.size` sufficiently high such that this method does not need to be used.

Auxiliary Functions:

1. `sieve.of.e` - This function implements a Sieve of Eratosthenes to calculate prime numbers.[3] It generates all prime numbers less than n.
2. `weights.calculation` - This function takes the interpolation weights and produces the interpolated variables, i.e. the weighted average based on the interpolation weights. It is called from inside the main function but could be used separately if one wished to take the existing interpolation weights and apply them to new variables that were not included in the original data frame.

- `pa_simulations_condor.R` — This file contains the code need to replicate the simulations. These files first define some relevant functions for the simulations (the gerrymandering algorithm, for example); to use this outside a CONDOR system, one would need to modify the code to remove the `c.process` variable and manually σ (`sd.range`) and r_{prop} (`vario.range.prop`). If they were set to a vector of values, the programme would run `reps` simulations for each combination.[4] This produces data frames with various pieces of information for each simulation; each combination of σ and r_{prop} generates ten observations as there are five levels of gerrymandering g implemented and two methods for each combination. Some columns in this data frame give other error statistics, e.g. the IQR of the error, that are not reported in the main text.

- `pa_simulations_analysis.R` — This file combines the simulations from the CONDOR batches and produces the relevant figures and tables. It also produces the preliminary figures and tables to describe the simulation process as well as the figures in Appendix B. Its file structure assumes there are 25 files located in the sub-directory `condor/output/`, each labelled with CONDOR-specific file names. Once these are loaded into `R`, the analysis begins. For convenience, I also include a file (`simulations_data.RData` in `replication_data.tar.gz`) that has already merged these together. This could be loaded directly to more easily replicate the results.

- `pa_empirical_analysis.R` — This file contains the code needed to replicate the empirical analysis, i.e. Table 1, in the paper. The relevant shapefiles and data are included in subfolders labelled by country. The code as written assumes a directory structure such that each subfolder with country data is is a sub-directory of the main working directory.

  This is a very long code file and is broken up into sections. The preliminary material loads the relevant packages. The first section (Auxiliary Functions) defines many of the functions used when formatting the interpolation results and thus should be run in all cases. There are then six country sections (Australia, Canada, Germany, Great Britain, New Zealand, and the US) in which the results for each country and its elections are calculated. Each section

[3] The code is a slightly modified of the one found here: `http://rosettacode.org/wiki/Sieve_of_Eratosthenes`

[4] I ran distributed the tasks on CONDOR such that each combination of parameters σ and r_{prop} was run 10 times for 75 iterations. To exactly replicate the results in the paper, one must exactly repeat this specification and ensure the random seeds are set appropriately. I would imagine that running 750 iterations using a different process of distributing the tasks (and a different random seed) should return almost identical results. The specific seed used can be inferred by the final digit in the file names (0-9).

can be run separately.[5] The final section combines and formats the preceding six sections and produces the tables and figures located in the main paper and in Appendix A.

Because compiling all of the results takes a fair amount of time, I have also provide a file (`empirical_results.RData`) with all of the country-specific interpolations and a second file (`interpolation_results.RData`) with the raw interpolation output (both located in `replication_data.tar.gz`) that can be loaded.

- `pa_condor_timings.R` — This file contains the code needed to replicate the timing analysis in Appendix 3. It is set up using the CONDOR system but effectively does the following; it loads an `RData` file containing the relevant Canadian shapefiles. It then produces one interpolation for some given raster size and some given `use.gdal` option and then stores the time and relevant interpolation data in a separate file. These are loaded and pulled together in `pa_simulation_analysis.R`

[5] NB: The Canadian simulations may fail to run because of `R`'s unwillingness to allocation vectors of certain sizes. If this occurs, I would suggest trying to interpolate Canada 'in parts', e.g. by a single province at a time as I did in the United States.