Step by Step Instructions

David J Dorer Dorer Community Service Foundation

Background work

1. Decide on your research question. Think about what tables and graphs you would like for a report.
2. Research PUMS variables that could help you create your tables and graphs.
3. Create derived PUMS factor (categorical) variables `help(BuiltinVariables)` to see examples. To see the associated R code, type the function name, e.g. `Education5.v`
4. Research ACS tables (Detail/Subject/DataProfile) that can be used as targets/marginals for adjustment of the variable based model. data.census.gov is a good place to start. Generally the more marginal tables the better.
5. Once you have marginal tables you may need to add variables in step 3. The variables in the marginal tables need to have corresponding PUMS derived variable to be able to fit the marginals. For example, if you want to use the Age x Race x Sex marginal you need Age.v Sex.v and Race.v as a PUMS variable. The marginal tables and the PUMS variables must have the same names and categories (factor labels).
6. You will also have "carry along" PUMS variables that don't correspond to any marginal tables. This will allow you to create multi-way tabulations for census tracts that don't correspond to any "B", "S" or "DP" tables
7. Select a unique model name, for example, MyStateI. Create a list to define the model. See Creating-Models, `vignette("CreatingModels")`
8. Make sure to include needed parameters in the model definition. For example the nages parameter will be needed for the Age.v PUMS variable and the AgeRaceSex.m marginal. You can also use your own "custom" parameters in your variable and marginal table functions as long as you have the parameter and value in your model parameters list.
9. Run 'PAT.test.model(model="MyStateI",state=,puma=) on your model to check for problems. See vignette CreatingModels.
10. Once step 9 is working, run `PAT.synth.data(<arguments>)` See `help(PAT.synth.data)` Start small. Select a PUMA and and a county or a single tract to see that everything is working.

`PAT.synth.data` will allow you to do all tracts within a state but be prepared to wait a long time, perhaps several days. There are functions that will divide up the work so that you can run several "batch" jobs at the same time. To run multiple jobs requires a lot of memory (depending on how many PUMS variables you have) and a processor with many "cores." The checkpoint file will be useful if you need to restart a run where you left off.

10. After you have created tract synthetic data, zip and merge the individual tract files `help(PAT.merge.data)`
11. Learn how to use the R package "survey" to create tables from your synthetic datasets with weights.
12. You can also aggregate the tracts "up" to geographies such as school districts or some counties that don't correspond to any PUMA, or collection of PUMAs.

Start of session

1. Set census key ‘PAT.census.key(“”)
2. Set root folder ‘PAT.root(“”);
3. Optionally set/review:
    a. ACS vintage and period for marginal tables: `PAT.vintage("<vintage>")` `PAT.period("<period>")`
    b. ACS vintage and period for PUMS variables: `PAT.pums.vintage("<vintage>")` `PAT.pums.period("<period>")`
4. Run `PAT.synth.example()` to test your setup. Watch for errors.
5. Review Getting Started to see other setup steps `vignette("GettingStarted")`

Some Technical Details

One issue with IPF is that a seed/model/PUMS table zero stays zero during the model fitting process. This may not be what you want. For example the PUM/PUMA table may have a sampling zero. This happens when the PUMA sample is too small to show a non-zero value even though there are individuals with that combination of characteristics. You can have a PUMA zero when there are marginal/target table non-zero values for that PUMS table cell. One way to handle this situation is to add a small amount to the seed/starting table/model using the `add` parameter. This allows a non-zero value for the zero PUMS cell to be fit.

The downside of this approach is that by adding a small amount to each cell in the seed table, every cell in the fit table will be non-zero. This creates a data set with as many rows as non-zero fit table cells. With a many variables the seed table can be several gigabytes. One antidote to large synthetic dataset is to drop synthetic data records small weights, hence the `minwt` argument to `PAT.synth.data()`. You can also drop records at the merge step `PAT.merge.data()`.

To put these considerations into effect a trial-by-error method may be required. If you drop records by using a minwt greater than zero you should check the results by computing synthetic data cross-tabulations that correspond to your marginal tables. Using 0.0001 for the `add` argument and 0.01 for the `minwt` argument to the `PAT.synth.data` function seems to be a reasonable compromise between synthetic dataset size and accuracy of the synthetic data tabulations when compared to the marginal tables used to fit the model. You might try several geographies to test the fit. See `vignette("SmallAreaEstimation")` “Internal and External Validity.”

v1.5 16 Dec 2023 18:59