juliacon

Survey.jl - An Efficient Framework for Analysing Complex Surveys

Ayush Patnaik[1], Nadia Enhaili[2], Siddhant Chaudhary[3], and Shikhar Mishra[1]

[1]XKDR Forum
[2]Simon Fraser University
[3]Chennai Mathematical Institute

ABSTRACT

Estimating variances in survey data analysis is challenging due to the complex nature of survey designs. It is typically done through resampling methods like bootstrapping, which can be computationally intensive. The Survey.jl package leverages Julia to provide an efficient framework for these resampling techniques, facilitating faster survey data analysis.

Keywords

Julia, Survey, Statistics, Sampling

1. Introduction

The growing volume of survey datasets necessitates more efficient analysis methods, particularly for variance estimation in complex survey designs. Computationally demanding resampling techniques, such as bootstrapping and jackknife, are required when dealing with stratification, clustering, and unequal weights.

Many software packages exist for survey analysis[1]. Notable examples include the R survey package, SAS/STAT, SPSS Complex Samples, Stata, and SUDAAN. The R survey package by Thomas Lumley[3] is widely recognized for its comprehensive capabilities and open-source availability. However, it is limited by R's computational efficiency, especially for large-scale data. Survey.jl leverages Julia to offer a faster resampling framework for variance estimation and survey data analysis.

2. Survey design

A `SurveyDesign` object can be created to incorporate the sampling design. This object requires the following arguments: `data::DataFrame`, which is the survey data in the form of a `DataFrame`; `clusters::Symbol`, specifying the column name containing the clusters; `strata::Symbol`, specifying the column name containing the strata; `weights::Symbol`, indicating the column name containing the sampling weights; and `popsize::Symbol`, indicating the column name containing the population size.[2]

For example, consider the NHANES dataset, which includes clustering and stratification. The following example demonstrates how to create a `SurveyDesign` object for this dataset:

```
julia> nhanes = load_data("nhanes");
# CSV dataframe included with the package

julia> design = SurveyDesign(nhanes;
                    clusters=:SDMVPSU,
                    strata=:SDMVSTRA,
                    weights=:WTMEC2YR);
```

Consider another example, a cluster sample based on the Academic Performance Index for all California schools based on standardised testing of students. There is no stratification in this example.

```
julia> apiclus1 = load_data("apiclus1");
# CSV dataframe included with the package

julia> design = SurveyDesign(apiclus1;
            clusters=:dnum, weights=:pw);
```

3. Estimation

Survey.jl provides a range of estimators for survey data analysis. These include univariate statistics such as mean, median, total, and quantiles, as well as multivariate statistics such as regressions and ratios. For example, to estimate the mean of the `:api99` column in the `design` SurveyDesign:

```
julia> mean(:api99, design)
1x1 DataFrame
    Row | mean
        | Float64
--------|--------
      1 | 606.978
```

This command estimates the mean of column `:api99`.
For multivariate statistics such as regressions[3]:

[1]A comprehensive list is provided by [1]
[2]Internaly, there is a single constructor for all types of surveys. Every survey is assumed to be a complex survey. If there is no stratification, we assume that everything is part of one stratum. If there is no clustering, we assume each member is a cluster.
[3]Regressions are performed using GLM.jl. Instead of passing a DataFrame, a survey design is passed to the function, maintaining a familiar interface.

```
julia> glm(@formula(y ~ x),
           my_design, Normal(), IdentityLink());
```

4. Replicate weights

The standard error of an estimator measures the average amount of variability or uncertainty in the estimated value. Standard errors are often provided alongside point estimates in various statistical packages.
To estimate standard errors for complex survey designs, Survey.jl uses replicate weights, generated through resampling techniques such as bootstrap and jackknife. Each replicate sample represents a plausible variation of the original sample, allowing for the estimation of variability as if the sampling were repeated multiple times. The estimate is calculated for each replicate, and then the standard error is computed from the distribution of these estimates.

4.1 Bootstrapping

In the bootstrap method, each replicate r involves selecting a simple random sample of $n_h - 1$ primary sampling units (PSUs) with replacement from the n_h sample PSUs in stratum h. The adjusted weight $w_i'(r)$ for observation i in replicate r is calculated as:

$$w_i'(r) = w_i(r) \times \frac{n_h}{n_h - 1} \times m_h(r) \tag{1}$$

Where $w_i(r)$ denotes the initial weight for observation i within replicate r, n_h is the total number of observations in stratum h, and $m_h(r)$ is the number of PSUs in stratum h that are selected in replicate r [2].
`bootweights` can be used to generate `ReplicateDesign{BootstrapReplicates}` from a `SurveyDesign`.

```
julia> bdesign = bootweights(design; replicates =
1000)
```

The replicate design object facilitates variance estimation. When a function receives a `ReplicateDesign` rather than a `SurveyDesign`, it provides the standard error along with the point estimate. For example:

```
julia> mean(:api99, bdesign)
1x2 DataFrame
    Row | mean      SE
        | Float64   Float64
--------|-----------------
      1 | 606.978   24.7505
```

For each replicate r, $\hat{\theta}_r^*$ is the estimator of θ, calculated the same way as $\hat{\theta}$ but using weights $w_i'(r)$ instead of the original weights w_i. The variance of the estimator is given by:

$$\hat{V}_B(\hat{\theta}) = \frac{1}{R-1} \sum_{r=1}^{R} (\hat{\theta}_r^* - \hat{\theta})^2. \tag{2}$$

This approach of using multiple dispatch is applied to all estimators imported from other packages, ensuring consistency and ease of use.

4.2 Jackknife

In the jackknife method, each PSU is systematically omitted one at a time to create replicates. The adjusted weight $w_{i(hj)}$ for observation i when PSU j in stratum h is omitted is calculated as:

$$w_{i(hj)} = \begin{cases} w_i & i \notin h \\ 0 & i \in j_h \\ \dfrac{n_h}{n_h - 1} w_i & i \in h \text{ and } i \notin j_h \end{cases} \tag{3}$$

[2]
`jackknifeweights` can be used to generate `ReplicateDesign{JackknifeReplicates}` from a `SurveyDesign`.

```
    julia> my_jackknife_design =
jackknifeweights(my_design)
```

This object can be passed to estimators to obtain an estimate of variance alongside the point estimate.
$\hat{\theta}$ represents the estimator computed using the original weights, and $\hat{\theta_{(hj)}}$ represents the estimator computed from the replicate weights obtained when PSU j from cluster h is removed. The variance is estimated as:

$$\hat{V}_{\text{JK}}(\hat{\theta}) = \sum_{h=1}^{H} \frac{n_h - 1}{n_h} \sum_{j=1}^{n_h} (\hat{\theta}_{(hj)} - \hat{\theta})^2 \tag{4}$$

4.3 Extending variance estimation

Survey.jl currently supports variance estimation for the summary statistics functions provided by the package, but the framework can be extended to custom estimators. The `variance` function can be applied to `ReplicateDesign` objects to estimate the variance of an estimator function, such as `Survey.mean`.

5. Conclusions

Survey.jl provides a comprehensive framework for survey data analysis, leveraging Julia's computational efficiency. The package has been tested against R's survey package, and future development aims to port all features from R.

6. Acknowledgements

We gratefully acknowledge the financial support from JuliaLab at MIT for this project. Iulia Dumitru has been a key contributor through GSoC. Harsh Arora, Sayantika Dasgupta, and other volunteers have also contributed. We thank Prof. Rajeeva Karandikar, Ajay Shah, Susan Thomas, Sourish Das, and Mousum Dutta for their valuable inputs.

7. References

[1] Summary of Survey Analysis Software. https://www.hcp.med.harvard.edu/statistics/survey-soft/#Packages.

[2] Sharon L. Lohr. *Sampling Design and Analysis*. Cengage Learning, 2010.

[3] Thomas Lumley. Analysis of complex survey samples. *Journal of statistical software*, 9:1–19, 2004.