ReGenesees: an Advanced R System for Calibration, Estimation and Sampling Errors Assessment in Complex Sample Surveys

Diego Zardetto
Istat – Methodology Service
What is ReGenesees?

- **ReGenesees acronym:**
  - R Evolved Generalised Software for Sampling Estimates and Errors in Surveys

- **Scope:**
  - Design-Based and Model-Assisted analysis of complex sample surveys

- **Programming Language:**
  - Entirely developed in interpreted R code (~260 functions, ~20,000 lines of code)

- **System Architecture:**
  - 2 integrated R packages:
    - Package **ReGenesees**: implements the statistical kernel
    - Package **ReGenesees.GUI**: implements the presentation layer
Main statistical functions (1/2)

- **Complex Sampling Designs**
  - Multistage, stratified, clustered, sampling designs
  - Unequally weighted sampling, with or without replacement
  - “Mixed” sampling designs (i.e. with both self-representing and non-self-representing strata)

- **Calibration**
  - Global and partitioned (for factorizable calibration models)
  - Unit-level and cluster-level adjustment
  - Homoscedastic and heteroscedastic models

- **Basic Estimators**
  - Horvitz-Thompson
  - Calibration Estimators

- **Variance Estimation**
  - Multistage formulation (but Ultimate Cluster approximation provided too)
  - Collapse strata technique for handling lonely PSUs
  - Taylor-linearization for non-linear “smooth” estimators
Main statistical functions (2/2)

- Estimates and Sampling Errors (standard error, variance, coefficient of variation, confidence interval, design effect) for:
  - Totals
  - Means
  - Absolute and relative frequency distributions (marginal, conditional and joint)
  - Ratios between totals
  - Multiple regression coefficients
  - Quantiles (variance estimation via the Woodruff method)

- Estimates and Sampling Errors for Complex Estimators
  - Handles arbitrary differentiable functions of Horvitz-Thompson or Calibration estimators
  - Complex Estimators can be freely defined by the user
  - Automated Taylor-linearization
  - Design covariance and correlation between Complex Estimators

- Estimates and Sampling Errors for Subpopulations (Domains)
ReGenesees: sample GUI screenshots
Advantages from a user perspective

- Many NSIs developed their own calibration & estimation platforms (mostly based on SAS), e.g.:
  - VPLX (US Census Bureau), GES (Statistics Canada), CLAN (Statistics Sweden), CALMAR & POULPE (INSEE), BASCULA (CBS), g-CALIB (Statistics Belgium), GENESEES (Istat), …

- As compared to these platforms, ReGenesees:
  - Automatically generates the auxiliary variables to calibrate on
  - Assists and drives the user in preparing the corresponding known population totals
  - Whenever the sampling frame is available (e.g. for sbs surveys) automatically computes, arranges and formats such population totals
  - Provides estimates and sampling errors for a wider range of built-in estimators
  - Allows users to define their own Complex Estimators
  - Automatically linearizes non-linear estimators, so that the estimation of their variance comes at no cost at all to the user
ReGenesees: a paradigm shift

- User interaction takes place at a higher level of abstraction
  - ReGenesees leverages R ability to process *symbolic* information

- ReGenesees users no longer need to preprocess survey data relying on *ad hoc* programs...

- ...they only have to feed the system with:
  - the survey data as they are
  - symbolic *metadata* that describe
    - the adopted *sampling design*
    - the chosen *calibration model*
    - the desired *estimators*

- The system does all the rest!

+ dramatic workload reduction
+ better usability
+ increased robustness against possible errors
+ full reproducibility
1. Persistently bind survey data \( (sbs) \) to sampling design metadata:

\[
sbsdes <- \text{e.svydesign}(\text{data}=sbs, \text{ids}=\sim \text{id}, \\
\text{strata}=\sim \text{strata}, \text{weights}=\sim \text{weight}, \text{fpc}=\sim \text{fpc})
\]

2. Build a template dataset to store known population totals:

\[
\text{pop} <- \text{pop.template}(\text{data}=sbsdes, \\
\text{calmodel} = \sim ((\text{emp.num} + \text{ent}) : (\text{nace2} + \text{emp.cl:nace.macro})):\text{region}-1)
\]

3. Compute the requested totals from the universe \( (sbs.frame) \) and safely fill the template:

\[
\text{pop} <- \text{fill.template}(\text{universe}=sbs.frame, \text{template}=\text{pop})
\]

4. Perform the calibration task:

\[
sbscal <- \text{e.calibrate}(\text{design}=sbsdes, \text{df.population}=\text{pop}, \\
\text{calfun} = \text{“linear”}, \text{bounds} = \text{c}(0.01,3))
\]

A Calibration example

462 auxiliary variables

Frame level

Sample level

No need of working out the 462 auxiliary variables!
Taylor linearization is a well established approximate tool for estimating the variance of Complex Estimators

- Clear mathematical framework

Non-linear functions of Horvitz-Thompson estimators of Totals

\[ \hat{\theta} = f(\hat{Y}_1, \ldots, \hat{Y}_m) \quad (1) \]

\[ \hat{\theta} \approx \theta + \sum_{j=1}^{m} \frac{\partial f}{\partial \hat{Y}_j} \bigg|_{\hat{Y}} (\hat{Y}_j - Y_j) \approx \hat{\theta}_{\text{lin}} \quad (2) \]

\[ \hat{\theta}_{\text{lin}} \approx \sum_{k \in s} d_k \hat{z}_k + \text{const} \quad (3) \]

\[ \hat{z}_k = \sum_{j=1}^{m} \frac{\partial f}{\partial \hat{Y}_j} \bigg|_{\hat{Y}} y_{jk} \quad (4) \]

\[ \hat{V}(\hat{\theta}) \approx \hat{V}(\sum_{k \in s} d_k \hat{z}_k) \quad (5) \]

“The Golden Rule”

Non-linear functions of Calibration estimators of Totals

\[ \hat{\theta} = f(\hat{Y}^{\text{CAL}}_1, \ldots, \hat{Y}^{\text{CAL}}_m) \quad (6) \]

\[ \hat{z}_k = \sum_{j=1}^{m} \frac{\partial f}{\partial \hat{Y}_j} \bigg|_{\hat{Y}^{\text{CAL}}} g_k \hat{\epsilon}_{jk} \quad (7) \]

\[ g_k = w_k / d_k \quad (8) \]

\[ \hat{\epsilon}_{jk} = y_{jk} - x_k \cdot \hat{\beta}_j \quad (9) \]
Traditional software issues with the Taylor approach:
- First order Taylor series expansion does depend on the functional form of the non-linear estimator
- Most of the traditional computing environments (e.g. SAS) are unable to perform symbolic differentiation
- Programs have to be developed separately for each Complex Estimator

R overcomes these limits:
- R can compute symbolic derivatives!
- R can manipulate user-defined mathematical expressions!

Thus ReGenesees:
- Provides a universal linearization function
- Allows its users to define their own Complex Estimators, i.e. statistics which are not built-in

ReGenesees function svystatL
Handling Complex Estimators

- ReGenesees has a simple syntax for specifying arbitrary Complex Estimators through their functional form
  - It exploits R methods for manipulating `expression` objects
  - The estimator of the total of a survey variable is simply represented by the name of the variable itself
  - The convenience name `ones` identifies an artificial variable whose value is 1 for each sampling unit; useful for:
    - Estimating the population size
    - Defining estimators of means

- Some elementary examples:
  - $\hat{Y}$ maps to: `expression(y)`
  - $\hat{R} = \frac{\hat{Y}}{\hat{X}}$ maps to: `expression(y/x)`
  - $\hat{N}$ maps to: `expression(ones)`
  - $\hat{\mu}_y = \frac{\hat{Y}}{\hat{N}}$ maps to: `expression(y/ones)`
Complex Estimators: example 1

Geometric mean \( \hat{G}_y = e^{\left(\frac{\hat{T}_{\log(y)}}{\hat{N}}\right)} \)

1. Add a new computed variable (i.e. the log) to the survey design:
   ```r
   sbsdes <- des.addvars(sbsdes, log.emp.num=log(emp.num))
   ```

2. Estimate the geometric mean, its standard error and confidence interval:
   ```r
   G <- svystatL(sbsdes, expression(exp(log.emp.num/ones)), conf.int=T)
   ```

3. Print on screen the obtained results:
   ```r
   Geometric mean
   
   \( \hat{G}_y \) = 20.5156
   SE = 0.0608
   CI.l(95%) = 20.3965
   CI.u(95%) = 20.6347
   ```

4. Compare with the “true value” (=computed from the population list):
   ```r
   exp(mean(log(sbs.frame$emp.num)))
   ```
   ```r
   [1] 20.5362
   ```
Complex Estimators: example 2

Standard deviation of a variable

\[
\hat{S}_y = \sqrt{\frac{\hat{N}}{\hat{N} - 1} \left[ \hat{\mu}_y^2 - (\hat{\mu}_y)^2 \right]}
\]

1. Add the squared variable to the survey design (va is for ‘value added’):
   ```
sbsdes <- des.addvars(sbsdes, va2=va^2)
   
   S
   ```

2. Compute the estimate, its standard error and confidence interval:
   ```
   S <- svystatL(sbsdes, expression(sqrt((ones/(ones-1))*(va2/ones) - (va/ones)^2))), conf.int=T)
   ```

3. Print on screen the obtained results:
   ```
   S
   ```

4. Compare with the “true value” (=computed from the population list):
   ```
   sd(sbs.frame$va)
   ```

[1] 9211.96
Ongoing work and future extensions

- Current ReGenesees version is 1.3, next release is scheduled for mid 2013
  - ReGenesees 1.0 was published in December 2011

- Migrating Istat procedures towards ReGenesees
  - ReGenesees has already been used successfully in production by 8 Istat large-scale surveys
    - 5 structural business surveys + 3 in the social-demographic domain
  - Gradually, all Istat surveys are expected to migrate from SAS to ReGenesees

- Further enhancements of the system
  - The Generalized Variance Functions method is currently under development
  - We are currently studying the feasibility of integrating inside ReGenesees the R package EVER
    - Provides the extended delete-a-group jackknife method for variance estimation
    - Would enable ReGenesees to handle non-analytic estimators (e.g. at-risk-of-poverty rate and other Laeken indicators)