Gustave: An R *package* for variance estimation in surveys

Nicolas Paliod, Martin Chevalier,
Lionel Delta, Thomas Deroyon

New Techniques and Technologies for Statistics 2019

13 March 2019

Insee
Mesurer pour comprendre

Why computing variance estimations of a survey indicator ?

- Variance estimation is important for the data analyst :
  - Enables to get a confidence interval
  - Enables to comment variations' significance
- Variance estimation is a way to measure survey indicators' quality
  - Included in the quality reports sent to Eurostat
  - Required in different surveys by the new European framework regulation for household surveys (*Integrated European social statistics*) under negotiation
- Variance estimation is an important step in the production process of a survey

Goals

What Gustave is supposed to do :

- Facilitates interactions between methodologists and data analysts
- Offers a computational framework to methodologists and data analysts with existing functionalities (linearization functions, domains...)
- Simplifies analytical variance estimation

What Gustave does not do :

- Does not provide a ready for use variance estimation function to methodologists
- Does not offer a computational framework for variance estimation with bootstrap

Plan

A toolkit

- Variance estimation for domains included in Gustave
- Some linearization functions included in Gustave (*ratio*, *diff_of_ratio*, *mean*)
- Some variance functions already coded (Sen Yates Grundy variance estimator *varSYG*, Deville-Tille variance estimator *varDT*)
- A way to take into account calibration included in the package (*res_cal*)

The use of Gustave for the methodologist

What the methodologist produces :

- A variance function for a total

What the methodologist does not need to code :

- Formatting
- Estimation for domains
- Classic linearization functions

Example (1/2)

- First step : Data computation in order to prepare all variables which are necessary for the variance estimation (for instance, non-response units' probability)

- Second step : Coding of the variance function
  Example with a two-stage sample scheme with primary units' sampling before a dwellings' sampling, a reweighting for nonresponse adjustment and a calibration on simulated data of Labour Force Survey

```
varEec <- function(y, up, log, ind){

  variance <- list()

  # Etape 0 : Agrégation par logement
  y <- sum_by(y, by = ind$idlog)

  # Etape 1 : Prise en compte du calage
  y <- add_zero(y, log$id[log$cal])
  y <- res_cal(y, precalc = log$res_cal_precalc)
```

Example (2/2)

```
  (...)

  # Etape 2 : Prise en compte de la non-réponse
  variance[["nr"]] <- colSums(
    (1/log$pilog[log$cal]^2 - log$qlog[log$cal]) *
      (1 - log$pinr[log$cal]) * (y/log$pinr[log$cal])^2
  )
  y <- add_zero(y / log$pinr[log$cal], log$id)

  # Etape 3 : Sélection des logements dans les up
  variance[["log"]] <- varDT(
    y, w = 1/(log$piup^2) - log$qup,
    precalc = log$varDT_precalc
  )

  # Etape 4 : Sélection des up
  y <- sum_by(y, by = log$idup, w = 1/log$pilog_up)
  y <- add_zero(y, up$id)
  variance[["up"]] <- varDT(y, precalc = up$precalc)

  colSums(do.call(rbind, variance))

}
```

A key functionality : the variance *wrapper* (1/2)

The variance wrapper adds functionalities (linearization functions, estimation for domains, ...) to the variance function for totals coded by the methodologist

Last part of the example :

```r
# Création du wrapper de variance avec define_variance_wrapper()
precisionEec <- define_variance_wrapper(
  variance_function = varEec,
  technical_data = list(up = up, log = log, ind = ind),
  reference_id = technical_data$ind$id,
  reference_weight = technical_data$ind$w,
  default_id = quote(paste0(ident, noi))
)

# Utilisation du wrapper de variance (données du T4 2014)
precisionEec(z, acteu %in% 2)
```

```
##                       call     est   variance      std       cv   lower
## 1 total(y = acteu %in% 2) 3001046 2158830156 46463.21 1.548234 2909980
##     upper
## 1 3092112
```

A key functionality : the variance *wrapper* (2/2)

Variance estimation of unemployment rate :

```r
precisionEec(z, ratio(acteu %in% 2, acteu %in% c(1, 2)))
```

```
##                                                    call
## 1 ratio(num = acteu %in% 2, denom = acteu %in% c(1, 2))
##         est   variance         std       cv     lower
## 1 0.1044647 2.5918e-06 0.001609907 1.541101 0.1013094
```

Variance estimation of unemployment rate for people over 50 :

```r
precisionEec(z,
  ratio(acteu %in% 2, acteu %in% c(1, 2)),
  where = age >= 50
)
```

```
##          est     variance         std       cv
## 1 0.07047492 5.402617e-06 0.002324353 3.298128
```

Variance estimation of unemployment rate by region :

```r
precisionEec(z,
  ratio(acteu %in% 2, acteu %in% c(1, 2)),
  by = reg
)
```

```
##   by       est     variance         std       cv
## 1 11 0.1003089 1.538408e-05 0.003922254 3.910175
## 2 21 0.1130015 1.068723e-04 0.010337904 9.148463
## 3 22 0.1220682 9.565600e-05 0.009780388 8.012235
```

An expandable *package*

define_variance_wrapper() can handle any variance function in input :

- as many parameters as necessary for the variance function
- use of other *packages* to code the variance function (use of require() in the variance function)

New linearization functions with define_statistic_wrapper() :

```r
# Définition du coefficient de gini à partir du package vardpoor
gini <- define_statistic_wrapper(
  statistic_function = function(y, weight){
    require(vardpoor)
    result <- lingini(Y = y, weight = weight)
    list(point = result$value$Gini, lin = result$lin$lin_gini)
  },
  arg_type = list(data = "y", weight = "weight", param = NULL)
)

# Utilisation pour calculer la précision dans l'enquête SRCV en 2014
precisionSrcv(r, gini(HX090))
```

qvar() : a ready for use variance estimation function

- "What Gustave does not do : to provide a ready for use variance estimation function to methodologists"... but for one specific case
- A ready for use variance estimation function for the following case :
  - Stratified simple random sampling
  - Nonresponse adjustment by reweighting with response homogeneity groups
  - Calibration
- Useful for business surveys
- qvar() built as a variance estimation function, which is integrated in a wrapper

Dissemination at Insee

Examples of use at Insee :

- Labour Force Survey
- Statistics on Income and Living Conditions
- Some business surveys

Used for surveys with different issues :

- French master sample (household surveys, formula from Breidt, Chauvet, 2011 and Gros, Moussallam, 2015)
- Multiple stages for indicators on individuals (Cadre de vie et sécurité)
- Weight sharing (SILC)

Plan

Constraints for dissemination

- Warning concerning the dissemination of the variance function produced in Gustave :

  The variance *wrapper* is a fully self-sufficient function : answers to the survey and information describing the sample scheme are given in input in the technical_data parameter and are kept in the function environment (it's a *closure*)

- Be careful to disseminate the variance function only to people who have the right to access your data and are aware of the rules on confidentiality

Dissemination, unit testing and seamless integration

- *package* available on Cran, last version : August 2018
- Open source code on several development platforms, in particular github.com
- possibility for users to suggest upgrades (to rectify bugs, to answer to specific requests, *pull requests*)
- More than 180 unitary tests in Gustave to test Gustave's functionalities
- Tests which are automatically computed at each *package's* new version : seamless integration

Conclusion

Insee's Statistical Method Department developed an R *package* to simplify variance estimation :

- a solution that simplify methodologists' work
- production of variance estimation functions which are simple to use
- a documented *package*, with a vignette in progress

A *package* used for Insee's household surveys :

- to produce variance estimation joined to quality reports
- to check that surveys satisfy to IESS's precision constraints