Production of social statistics... goes social!

Jacopo Grazzini and Pierre Lamarche
Eurostat, European Commission

November 17, 2016

Abstract

The scope of this paper is to present a framework adopted in practice for the integration of software applications into a statistical production chain. The focus is the actual implementation of a high-level collaborative platform aiming not only at producing social statistics, but also at further fostering experimentation and analysis in that field. The need for transparency fosters an open, reusable, verifiable, reproducible and collaborative implementation of (statistical) processes, especially for data-driven policies. In this paper, we tackle the practical implementation framework of such processes - based on reasonable mix of *bottom-up* and *top-down* designs, as well as case studies. As a conclusion, we sketch the possible way forward and a future action plan.

1 Introduction

The integration of software applications into the statistical production chain is a key question in a modern environment where standards and software may swiftly be outdated, thereby necessitating a thorough reflection in terms of transparency and documentation of the processes. We would like to exhibit and promote in this paper the implementation of a high-level collaborative platform aiming not only at producing social statistics, but also at further fostering experimentation and analysis in that field. In doing so, we strongly support the (obvious) claim of [?] that "the modernisation and industrialisation of official statistical production needs a unified combination of statistics and computer science in its very principles".
Motivated by the consensus that processes - in particular statistical processes [?] - for data-driven policy should be transparent [?], we require software development and deployment in a statistical organisation to be open, reusable verifiable, reproducible, and collaborative. Beyond just devising guidelines and best practices, we show how the platform is implemented for the production of social statistics. For that purpose, we adopt a reasonable mix of bottom-up (from low-level scope to high-level vision) and top-down (from black-box process models to traceable functional modules) designs, so as to "think global, [and] act local" [?]. In building the parts while planning the whole, we provide with a flexible and agile approach to immediate needs and current legacy issues, as well as long-term problems and potential future requirements in statistical production [?].

2 Motivation

3 Adopted design principles

4 Practical implementation framework and current status

5 Case studies

5.1 Quantile computation

5.1.1 The unweighted case

When one is interested in describing the distribution of some variable of interest (that will be denoted X), quantiles are most of the times the first indicators that come to mind. Quantiles are defined according to p, a variable between 0 and 1 (or alternatively between 0 and 100), as following:

$$Q(p) = F^{-1}(p) = inf\{x/F(x) \geq p\}, 0 < p < 1 \quad (1)$$

Since F is referring to the cumulative distribution function (CDF), such a definition entails the uniqueness of the solution: there is one and only one x associated to a given p. However, in computational terms, as we can only compute an **estimation** of the quantiles, there are many ways for computing quantiles: [?] list 9 different methods that are not meant to provide with the same result. We leave aside other methods, such as the estimation of L-moments or quantile mixture [?] [?][1], as they rely on strong assumptions over the parametric shape of the distribution. [?] summarizes the 9 possible methods as follows, for $(X_{(1)}, ..., X_{(n)})$ the n sorted observations of the variable X, $i \in \{1, ..., 9\}$, $0 \leq \gamma \leq 1$ and $m \in \mathbb{R}$:

$$\hat{Q}_i(p) = (1-\gamma)X_{(j)} + \gamma X_{(j+1)} \tag{2}$$

$$\frac{j-m}{n} \leq p < \frac{j-m+1}{n} \tag{3}$$

The values taken by γ is function of i, $j = \lfloor pn + m \rfloor$ and $g = pn + m - j$. The different possibilities in SAS® are mapped with their counterpart in **R** in Table **??**. As shown by [?], the different definitions may be classified into two main categories, reflecting assumptions on the shape of the inverse CDF:

- discountinuous functions (definitions 1, 2 and 3)
- piecewise linear continuous functions (definitions 4 and higher)

Each quantile definition presents advantages and drawbacks, especially in terms of assumptions and desirable properties. From this respect, according to Table **??**, SAS® is giving more prominence to discontinuous functions (thereby relying on less demanding assumptions), whereas **R** offers the possibility to the users to compute any type of estimation. [?] show that definition 5 is the more appropriate definition of the estimation for quantiles with respect to a list of desirable properties for quantiles; however they rather advocate for using the definition 8, following several papers that all conclude to the superiority of this definition when looking for a median-unbiased estimator. As a matter of fact, such definitions (both 5 and 8) are not implemented in SAS®, leaving the user willing to use these definitions with no alternative than implementing himself the computation (or switching to **R**).
As a consequence, the usual statistical software **R** or SAS® provide users with procedures enabling various methods for computing quantiles. As a matter of fact, in **R** the command to compute an estimation of the median

```
> quantile(x, probs = 0.5, type = 1)
```

refers to the method 1 described in [?], according to [?]. The counterpart in SAS® may be obtained thanks to the following command:

```
PROC UNIVARIATE DATA = mydata PCTLDEF = 3 ;
VAR x ;
RUN ;
```

[1]For an implementation of such methods in **R**, the interested reader may refer to [?].

In this implementation, as explained in the SAS® documentation[2], there are 5 possible estimations (versus 9 in **R**). It is therefore not possible to fully map all possible methods from **R** into SAS®.

Definition	Values taken by m, γ	SAS® `PCTLDEF`	**R** `type`
Definition 1	$m = 0$ and $\gamma = \begin{cases} 1 \text{ if } g > 0 \\ 0 \text{ if } g = 0 \end{cases}$	3	1
Definition 2	$m = 0$ and $\gamma = \begin{cases} 1 \text{ if } g > 0 \\ 1/2 \text{ if } g = 0 \end{cases}$	5	2
Definition 3	$m = -\frac{1}{2}$ and $\gamma = \begin{cases} 0 \text{ if } g = 0 \text{ and } j \text{ is even} \\ 1 \text{ otherwise} \end{cases}$	2	3
Definition 4	$m = 0, \gamma = g$	1	4
Definition 5	$m = \frac{1}{2}, \gamma = g$	n.a.	5
Definition 6	$m = p, \gamma = g$	4	6
Definition 7	$m = 1 - p, \gamma = g$	n.a.	7
Definition 8	$m = \frac{1+p}{3}, \gamma = g$	n.a.	8
Definition 9	$m = \frac{2p+3}{8}, \gamma = g$	n.a.	9

Table 1: Mapping between SAS® and **R** unweighted quantile methods, according to definitions as in [?]

5.1.2 The weighted case

When facing weighted data (such as survey data), one has to account for the sampling design in order to properly compute the estimation of the quantiles. In the weighted case, an Horvitz-Thompson estimator [?] of the CDF F would be:

$$\hat{F}(x) = \frac{1}{\sum_{i \in s} \omega_i} \sum_{i \in s} \omega_i \mathbb{1}_{\{X_i \leq x\}} \tag{4}$$

This being said, all the difficulty is to define estimators of the inverse CDF. Following the framework in the unweighted casis, such estimators may be defined through a generic form, considering again $(X_{(1)}, ..., X_{(n)})$ the n sorted observations of X, but this time coming along with the set of weights $(\omega_{(1)}, ..., \omega_{(n)})$, $\hat{N} = \sum_{i \in s} \omega_i$:

$$\hat{Q}_i(p) = (1 - \gamma) X_{(j)} + \gamma X_{(j+1)} \tag{5}$$

$$\frac{\sum_{i=1}^{j} \omega_{(i)} - m\omega_j}{\hat{N}} \leq p < \frac{\sum_{i=1}^{j+1} \omega_{(i)} - m\omega_j}{\hat{N}} \tag{6}$$

[2]https://support.sas.com/documentation/cdl/en/procstat/63104/HTML/default/viewer.htm

Similarly, j and g may also be defined as $j = Argmax_{k\in\{1,\ldots,n\}}\{k/\sum_{i=1}^{k}\omega_{(i)} \leq p\hat{N}+m\omega_k\}$ and $g = p\hat{N} + m\omega_j - j$.
Hence, things get more complicated when facing survey data, or any source of data which entails the use of weighted statistics. Indeed, if the latter still applies for SAS®, the described command `quantile` in **R** does not embed an option for specifying a weighting variable, thereby shrinking dramatically the utility of such a command, as in practice it can be only applied to simple random sample (*i.e.* with equal probability of selection). One of the alternatives is to use the package `Hmisc` [?]. The quantile (in this case the median, in order to pursue the example above) may then be computed as follows, provided that the vector giving the weights is called `w`:

```
> library(Hmisc)
> wtd.quantile(x, weights = w, probs = 0.5, type = 'quantile')
```

There is another way for computing quantiles on survey data in **R**, using the `survey` package [?]. However, using such a package entails also to describe the sampling design of the survey, since the point estimate comes along with an estimation of the variance. The call of this estimation is done as follows:

```
> library(survey)
> svyquantile(x, design=designSurvey, quantile = 0.5, method = 'linear',
   ties = 'discrete')
```

It turns out, when comparing the **R** command `wtd.quantile` with the SAS® procedure `UNIVARIATE`, that the software go into opposite directions
One of the main drawbacks of **R** comes from its own strengths: as the packages downloadable on the `CRAN` website are developped in a decentralized manner, there is very few moves toward some significant harmonization/rationalization efforts for the implemented functions. Therefore, in the package `Hmisc`, if the option `type` remains the same across packages `base` and `Hmisc`, the naming of the various methods called through this option varies.

5.2 Sampling process

6 Future plans