Estimation of Social Exclusion Indicators From Complex Surveys: The R Package laeken

Andreas Alfons
KU Leuven

Matthias Templ
Vienna University of Technology

Abstract

Units sampled from finite populations typically come with different inclusion probabilities. Together with additional preprocessing steps of the raw data, this yields unequal sampling weights of the observations. Whenever indicators are estimated from such complex samples, the corresponding sampling weights have to be taken into account. In addition, many indicators suffer from a strong influence of outliers, which are a common problem in real-world data. The R package **laeken** is an object-oriented toolkit for the estimation of indicators from complex survey samples via standard or robust methods. In particular the most widely used social exclusion and poverty indicators are implemented in the package. A general calibrated bootstrap method to estimate the variance of indicators for common survey designs is included as well. Furthermore, the package contains synthetically generated close-to-reality data for the European Union Statistics on Income and Living Conditions (EU-SILC) and the Structure of Earnings Survey (SES), which are used in the code examples throughout the paper. Even though the paper is focused on showing the functionality of package **laeken**, it also provides a brief mathematical description of the implemented indicator methodology.

Keywords:˜indicators, robust estimation, sample weights, survey methodology, R.

1. Introduction

Estimation of indicators is one of the main tasks in survey statistics. They are usually estimated from complex surveys with many thousands of observations, conducted in a harmonized manner over many countries.

Indicators are designed to reflect major developments in society, for example with respect to poverty, social cohesion or gender inequality, in order to quantify and monitor progress towards policy objectives. Moreover, by implementing a monitoring system across countries via a harmonized set of indicators, different policies can be compared based on quantitative information regarding their impact on society. Thus statistical indicators are an important source of information on which policy makers can base their decisions.

Nevertheless, for policy decisions to be effective, the underlying quantitative information from the indicators needs to be reliable. Not only should the variability of the indicators be kept in mind, but also the impact of data collection and preprocessing needs to be considered. Indicators are typically based on complex surveys, in which units are drawn from finite populations, most often with unequal inclusion probabilities. Hence the observations in the sample represent different numbers of units in the population, giving them unequal sample weights.

In addition, those initial weights are often modified by preprocessing steps such as calibration for nonresponse. Therefore, sample weights always need to be taken into account in the estimation of indicators from survey samples, otherwise the estimates may be biased.

The focus of this paper is on socioeconomic indicators on poverty, social cohesion and gender differences. In economic data, extreme outliers are a common problem. Such outliers can have a disproportionally large influence on the estimates of indicators and may completely distort them. If indicators are corrupted by outliers, wrong decisions could be drawn by policy makers. Robust estimators that give reliable estimates even in the presence of extreme outliers are therefore necessary.

We introduce the add-on package **laeken** (Alfons, Holzer, and Templ 2012) for the open source statistical computing environment R (R Development Core Team 2012). It provides functionality for standard and robust estimation of indicators on social exclusion and poverty from complex survey samples. The aim of the paper is to present the most important functionality of the package. A more complete overview of the available functionality is given in three additional package vignettes on specialized topics:

`laeken-standard`**:** standard estimation of the indicators

`laeken-pareto`**:** robust estimation of the indicators via Pareto tail modeling

`laeken-variance`**:** variance estimation for the indicators

These vignettes can be viewed from within R with the following commands:

```
R> vignette("laeken-standard")
R> vignette("laeken-pareto")
R> vignette("laeken-variance")
```

Even though official statistical agencies usually rely on commercial software, R has gained some traction in the survey statistics community over the years. Many add-on packages for survey methodology are now available. For instance, an extensive collection of methods for the analysis of survey samples is implemented in package **survey** (Lumley 2004, 2012). The accompanying book by Lumley (2010) also serves as an excellent introduction to survey statistics with R. Other examples for more specialized functionality are package **sampling** (Tillé and Matei 2012) for finite population sampling, and package **EVER** (Zardetto 2012) for variance estimation based on efficient resampling. For the common problem of nonresponse, package **VIM** (Templ, Alfons, Kowarik, and Prantner 2012b) allows to explore the structure of missing data via visualization techniques (see Templ, Alfons, and Filzmoser 2012a), and to impute the missing values via advanced imputation methods (e.g., Templ, Kowarik, and Filzmoser 2011). Even a general framework for simulation studies in survey statistics is available through package **simFrame** (Alfons, Templ, and Filzmoser 2010; Alfons 2012).

The rest of the paper is organized as follows. Section˜2 introduces the data sets that are used in the examples throughout the paper. In Section˜3, the most widely used indicators on social exclusion and poverty are briefly described. The basic design of the package and its core functionality are then presented in Section˜4. More advanced topics such as robust estimation and variance estimation via bootstrap techniques are discussed in Sections˜5 and˜6, respectively. The final Section˜7 concludes.

2. Data sets

Package **laeken** contains example data sets for two well-known surveys: the *European Union Statistics on Income and Living Conditions* (EU-SILC) and the *Structure of Earnings Survey* (SES). Since original data from those surveys are confidential, the example data sets are simulated using the methodology described in Alfons, Kraft, Templ, and Filzmoser (2011) and implemented in the R package **simPopulation** (Alfons and Kraft 2012). In any case, these data sets are used in the code examples throughout the paper.

2.1. European Union Statistics on Income and Living Conditions (EU-SILC)

EU-SILC is an annual household survey conducted in EU member states and other European countries. Samples consist of about 450 variables containing information on demographics, income and living conditions (see Eurostat 2004b). Most notably, EU-SILC serves as data basis for measuring risk-of-poverty and social cohesion in Europe. A subset of the indicators computed from EU-SILC is presented in Section˜3.2.

The EU-SILC example data set in **laeken** is called `eusilc` and contains 14 827 observations from 6 000 households and 28 variables. Thus only the most important variables from the survey are available. The data are synthetically generated from Austrian EU-SILC survey data from 2006. Most of the variable names are rather cryptic codes, but these are the standardized names used by the statistical agencies. A description of all the variables is given in the R help page of the data set. To give an overview of what the data look like, the first three observations of `eusilc` are printed below.

```
R> data("eusilc")
R> head(eusilc, 3)
```

```
  db030 hsize db040 rb030 age  rb090 pl030 pb220a   py010n py050n
1     1     3 Tyrol   101  34 female     2     AT  9756.25      0
2     1     3 Tyrol   102  39   male     1  Other 12471.60      0
3     1     3 Tyrol   103   2   male  <NA>   <NA>       NA     NA
  py090n py100n py110n py120n py130n py140n hy040n  hy050n hy070n
1      0      0      0      0      0      0 4273.9 2428.11      0
2      0      0      0      0      0      0 4273.9 2428.11      0
3     NA     NA     NA     NA     NA     NA 4273.9 2428.11      0
  hy080n hy090n hy110n hy130n hy145n eqSS eqIncome    db090    rb050
1      0  33.39      0      0      0  1.8 16090.69 504.5696 504.5696
2      0  33.39      0      0      0  1.8 16090.69 504.5696 504.5696
3      0  33.39      0      0      0  1.8 16090.69 504.5696 504.5696
```

For this paper, the variable `eqIncome` (equivalized disposable income) is of main interest. Other variables are in some cases used to break down the data into different demographics in order to estimate the indicators on those subsets.

2.2. Structure of Earnings Survey (SES)

The Structure of Earnings Survey (SES) (Eurostat 2006) is an enterprise survey that aims at providing harmonized data on earnings for almost all European countries. SES data not only contain information on the enterprise level, but also on the individual employment level from a large sample of employees. Similar surveys are conducted all over the world. In general such linked employer-employee data are used to identify the determinants/differentials of earnings, but also some indicators are directly derived from the hourly earnings data included in the survey. The most important indicator on the basis of SES data is the gender pay gap, which is described in Section˜3.3.

The SES example data set in **laeken** is called `ses` and contains information on 27 variables and 15691 employees from 500 places of work. It is a subset of synthetic data that are simulated from real Austrian SES 2006 data. The first three observations are shown below.

```
R> data("ses")
R> head(ses, 3)
```

```
    location    NACE1  size economicFinanc payAgreement IDunit  sex
112      AT3 C-Mining E1000              B            B  81461 male
111      AT3 C-Mining E1000              B            B  81461 male
114      AT3 C-Mining E1000              B            B  81461 male
        age     education occupation            contract fullPart
112 (39,49] ISCED 3 and 4         34 indefinite duration       FT
111 (39,49] ISCED 3 and 4         34 indefinite duration       FT
114 (29,39] ISCED 3 and 4         31 indefinite duration       FT
    lengthService weeks hoursPaid overtimeHours shareNormalHours
112       (19,29] 52.14  160.2820      0.000000              100
111       (19,29] 52.14  144.5616      0.000000              100
114       (19,29] 52.14  174.4551      8.940864              100
     holiday   notPaid earningsOvertime paymentsShiftWork earningsMonth
112 21.89396  9688.986      1181.125474                 0      3913.502
111 28.20815 10390.651       919.354409                 0      5151.819
114 30.26150 21916.674         1.656994                 0      4915.543
     earnings earningsHour weightsEmployers weightsEmployee weights
112  81431.45     31.16641                1               1       1
111 131128.02     17.20338                1               1       1
114 143250.38     33.01964                1               1       1
```

In this paper, the SES data is used to illustrate the estimation of the gender pay gap. Hence the most important variables for our purposes are `earningsHour`, `sex` and `education`. For a description of all the variables in the data set, the reader is referred to its R help page.

3. Indicators

This section gives a brief description of the most widely used indicators to measure poverty, social cohesion and gender differences. Unless otherwise stated, the presented definitions strictly follow Eurostat (2004a, 2009), where more details can be found. In addition to the mathematical definitions of the indicators, quick examples for computing them with package

laeken are provided. A detailed discussion on the respective functions is given later on in Section˜4.

3.1. Weighted median and quantile estimation

Nearly all of the indicators considered in the paper require the estimation of the median income or other quantiles of the income distribution. Note that in the analysis of income distributions, the median income is of higher interest than the arithmetic mean, since income distributions typically are strongly right-skewed.

In mathematical terms, quantiles are defined as $q_p := F^{-1}(p)$, where F is the distribution function on the population level and $0 \leq p \leq 1$. The median as an important special case is given by $p = 0.5$. For the following definitions, let n be the number of observations in the sample, let $\boldsymbol{x} := (x_1, \ldots, x_n)'$ denote the income with $x_1 \leq \ldots \leq x_n$, and let $\boldsymbol{w} := (w_i, \ldots, w_n)'$ be the corresponding sample weights. Weighted quantiles for the estimation of the population values are then given by

$$\hat{q}_p = \hat{q}_p(\boldsymbol{x}, \boldsymbol{w}) := \begin{cases} \frac{1}{2}(x_j + x_{j+1}), & \text{if } \sum_{i=1}^{j} w_i = p \sum_{i=1}^{n} w_i, \\ x_{j+1}, & \text{if } \sum_{i=1}^{j} w_i < p \sum_{i=1}^{n} w_i < \sum_{i=1}^{j+1} w_i. \end{cases} \tag{1}$$

3.2. Indicators on social exclusion and poverty

The indicators described in this subsection are estimated from EU-SILC data based on household income rather than personal income. For each person, this *equivalized disposable income* is defined as the total household disposable income divided by the equivalized household size. It follows that each person in the same household receives the same equivalized disposable income. The total disposable income of a household is thereby calculated by adding together the personal income received by all of the household members plus the income received at the household level. The equivalized household size is defined according to the modified OECD scale, which gives a weight of 1.0 to the first adult, 0.5 to other household members aged 14 or over, and 0.3 to household members aged less than 14.

For the definitions of the following indicators, let $\boldsymbol{x} := (x_1, \ldots, x_n)'$ be the equivalized disposable income with $x_1 \leq \ldots \leq x_n$ and let $\boldsymbol{w} := (w_i, \ldots, w_n)'$ be the corresponding sample weights, where n denotes the number of observations. Furthermore, define the following index sets for a certain threshold t:

$$I_{<t} := \{i \in \{1, \ldots, n\} : x_i < t\}, \tag{2}$$

$$I_{\leq t} := \{i \in \{1, \ldots, n\} : x_i \leq t\}, \tag{3}$$

$$I_{>t} := \{i \in \{1, \ldots, n\} : x_i > t\}. \tag{4}$$

At-risk-at-poverty rate

In order to define the *at-risk-of-poverty rate* (ARPR), the *at-risk-of-poverty threshold* (ARPT) needs to be introduced first, which is set at 60% of the national median equivalized disposable income. Then the at-risk-at-poverty rate is defined as the proportion of persons with an equivalized disposable income below the at-risk-at-poverty threshold. In a more mathematical

notation, the at-risk-at-poverty rate is defined as

$$ARPR := P(x < 0.6 \cdot q_{0.5}) \cdot 100, \tag{5}$$

where $q_{0.5} := F^{-1}(0.5)$ denotes the population median (50% quantile) and F is the distribution function of the equivalized income on the population level.

For the estimation of the at-risk-at-poverty rate from a sample, first the at-risk-at-poverty threshold is estimated by

$$\widehat{ARPT} = 0.6 \cdot \hat{q}_{0.5}, \tag{6}$$

where $\hat{q}_{0.5}$ is the weighted median as defined in Equation˜(1). Then the at-risk-at-poverty rate can be estimated by

$$\widehat{ARPR} := \frac{\sum_{i \in I_{<\widehat{ARPT}}} w_i}{\sum_{i=1}^{n} w_i} \cdot 100, \tag{7}$$

where $I_{<\widehat{ARPT}}$ is an index set of persons with an equivalized disposable income below the estimated at-risk-of-poverty threshold as defined in Equation˜(2).

In package **laeken**, the function `arpr()` is implemented to estimate the at-risk-at-poverty rate.

```
R> arpr("eqIncome", weights = "rb050", data = eusilc)

Value:
[1] 14.44422

Threshold:
[1] 10859.24
```

Note that the at-risk-of-poverty threshold is computed internally by `arpr()`. If necessary, it can also be computed by the user through function `arpt()`.

In addition, a highly related indicator is the *dispersion around the at-risk-of-poverty threshold*, which is defined as the proportion of persons with an equivalized disposable income below 40%, 50% and 70% of the national weighted median equivalized disposable income. For the estimation of this indicator with function `arpr()`, the proportion of the median equivalized income to be used can easily be adjusted via the argument `p`.

```
R> arpr("eqIncome", weights = "rb050", p = 0.4, data = eusilc)

Value:
[1] 4.766885

Threshold:
[1] 7239.491

R> arpr("eqIncome", weights = "rb050", p = 0.5, data = eusilc)

Value:
[1] 7.988134

Threshold:
[1] 9049.363
```

```
R> arpr("eqIncome", weights = "rb050", p = 0.7, data = eusilc)

Value:
[1] 21.85638

Threshold:
[1] 12669.11
```

Quintile share ratio

The income *quintile share ratio* (QSR) is defined as the ratio of the sum of the equivalized disposable income received by the 20% of the population with the highest equivalized disposable income to that received by the 20% of the population with the lowest equivalized disposable income.

For a given sample, let $\hat{q}_{0.2}$ and $\hat{q}_{0.8}$ denote the weighted 20% and 80% quantiles, respectively, as defined in Equation˜(1). Using index sets $I_{\leq \hat{q}_{0.2}}$ and $I_{>\hat{q}_{0.8}}$ as defined in Equations˜(3) and˜(4), respectively, the quintile share ratio is estimated by

$$\widehat{QSR} := \frac{\sum_{i \in I_{>\hat{q}_{0.8}}} w_i x_i}{\sum_{i \in I_{\leq \hat{q}_{0.2}}} w_i x_i}. \tag{8}$$

To estimate the quintile share ratio, the function `qsr()` is available.

```
R> qsr("eqIncome", weights = "rb050", data = eusilc)

Value:
[1] 3.971415
```

Relative median at-risk-of-poverty gap

The *relative median at-risk-of-poverty gap* (RMPG) is given by the difference between the median equivalized disposable income of persons below the at-risk-of-poverty threshold and the at-risk of poverty threshold itself, expressed as a percentage of the at-risk-of-poverty threshold.

For the estimation of the relative median at-risk-of-poverty gap from a sample, let $\widehat{ARPT}$ be the estimated at-risk-of-poverty threshold according to Equation˜(6), and let $I_{<\widehat{ARPT}}$ be an index set of persons with an equivalized disposable income below the estimated at-risk-of-poverty threshold as defined in Equation˜(2). Using this index set, define $\boldsymbol{x}_{<\widehat{ARPT}} := (x_i)_{i \in I_{<\widehat{ARPT}}}$ and $\boldsymbol{w}_{<\widehat{ARPT}} := (w_i)_{i \in I_{<\widehat{ARPT}}}$. Furthermore, let $\hat{q}_{0.5}(\boldsymbol{x}_{<\widehat{ARPT}}, \boldsymbol{w}_{<\widehat{ARPT}})$ be the corresponding weighted median according to the definition in Equation˜(1). Then the relative median at-risk-of-poverty gap is estimated by

$$\widehat{RMPG} = \frac{\widehat{ARPT} - \hat{q}_{0.5}(\boldsymbol{x}_{<\widehat{ARPT}}, \boldsymbol{w}_{<\widehat{ARPT}})}{\widehat{ARPT}} \cdot 100. \tag{9}$$

The relative median at-risk-of-poverty gap is implemented in the function `rmpg()`.

```
R> rmpg("eqIncome", weights = "rb050", data = eusilc)

Value:
[1] 18.9286

Threshold:
[1] 10859.24
```

Gini coefficient

The *Gini coefficient* is defined as the relationship of cumulative shares of the population arranged according to the level of equivalized disposable income, to the cumulative share of the equivalized total disposable income received by them.

Mathematically speaking, the Gini coefficient is estimated from a sample by

$$\widehat{Gini} := 100 \left[\frac{2 \sum_{i=1}^{n} \left(w_i x_i \sum_{j=1}^{i} w_j \right) - \sum_{i=1}^{n} {w_i}^2 x_i}{\left(\sum_{i=1}^{n} w_i \right) \sum_{i=1}^{n} \left(w_i x_i \right)} - 1 \right]. \tag{10}$$

For estimating the Gini coefficient, the function `gini()` can be used.

```
R> gini("eqIncome", weights = "rb050", data = eusilc)

Value:
[1] 26.48962
```

3.3. The Gender Pay Gap

Probably the most important indicator derived from the SES data is the *gender pay gap* (GPG). The calculation of the gender pay gap is based on each person's hourly earnings, which are given by the gross monthly earnings from employment divided by the number of hours usually worked per week in employment during 4.33 weeks. The gender pay gap in unadjusted form is then defined as the difference between average gross earnings of male paid employees and of female paid employees divided by the earnings of male paid employees (Eurostat 2004a). Further discussion on the gender pay gap in Europe can be found in, e.g., Beblot, Beniger, Heinze, and Laisney (2003).

For the following definitions, let $\boldsymbol{x} := (x_1, \ldots, x_n)'$ be the hourly earnings with $x_1 \leq \ldots \leq x_n$, where n is the number of observations. As in the previous subsections, $\boldsymbol{w} := (w_i, \ldots, w_n)'$ denotes the corresponding sample weights. Then define the index set

$$\begin{aligned} I_M := \{ i \in \{1, \ldots, n\} : & \text{ worked as least 1 hour per week} \wedge \\ & (16 \leq \text{age} \leq 65) \wedge \text{person is male} \}, \end{aligned}$$

and define I_F analogously as the index set which differs from I_M in the fact that it includes females instead of males. With these index sets, the gender pay gap in unadjusted form is estimated by

$$GPG_{(mean)} = \left(\frac{\sum_{i \in I_M} w_i x_i}{\sum_{i \in I_M} w_i} - \frac{\sum_{i \in I_F} w_i x_i}{\sum_{i \in I_F w_i}} \right) \Bigg/ \frac{\sum_{i \in I_M} w_i x_i}{\sum_{i \in I_M} w_i}. \tag{11}$$

The function `gpg()` is implemented in **laeken** to estimate the gender pay gap.

```
R> gpg("earningsHour", gender = "sex", weigths = "weights",
+      data = ses)

Value:
[1] 0.2517759
```

While Eurostat (2004a) proposes the weighted mean as a measure for the average in the definition of the gender pay gap, the U.S. Census Bureau uses the weighted median as a robust alternative (see, e.g., Weinberg 2007). In this case, the estimate of the gender pay gap in unadjusted form changes to

$$GPG_{(med)} = \frac{\hat{q}_{0.5}(\boldsymbol{x}_{I_M}) - \hat{q}_{0.5}(\boldsymbol{x}_{I_F})}{\hat{q}_{0.5}(\boldsymbol{x}_{I_M})}, \tag{12}$$

where $\boldsymbol{x}_{I_M} := (x_i)_{i \in I_M}$ and $\boldsymbol{x}_{I_F} := (x_i)_{i \in I_F}$.

It should be noted that even though Eurostat proposes to estimate the gender pay gap via weighted means, Statistics Austria for example uses the variant based on weighted medians as well.

In function `gpg()`, using the weighted median rather than the weighted mean can be specified via the `method` argument.

```
R> gpg("earningsHour", gender = "sex", weigths = "weights",
+      data = ses, method = "median")

Value:
[1] 0.229818
```

4. Basic design and core functionality

This section discusses the basic design of package **laeken** and its core functions for the estimation of indicators. First, Section˜4.1 describes the functions for estimating the indicators and the class structure of the returned objects. Sections˜4.2 and˜4.3 then show how to estimate indicators for different subdomains and extract subsets of the information from the resulting objects.

4.1. Indicators and class structure

Small examples for computing the social exclusion and poverty indicators with package **laeken** were already shown in Section˜3. These functions are now discussed in detail. As a reminder, the following indicators are implemented in the package:

`arpr()` for the at-risk-of-poverty rate, as well as the dispersion around the at-risk-of-poverty threshold

`qsr()` for the quintile share ratio

`rmpg()` for the relative median at-risk-of-poverty gap

`gini()` for the gini coefficient

`gpg()` for the gender pay gap

All these functions have a very similar interface and allow to compute point and variance estimates with a single command, even for different subdomains of the data. Most importantly, the user can supply character strings specifying the household income via the first argument and the sample weights via the `weights` argument. The data are then taken from the data frame passed as the `data` argument.

```
R> gini("eqIncome", weights = "rb050", data = eusilc)

Value:
[1] 26.48962
```

Alternatively, the user can supply the data directly as vectors:

```
R> gini(eusilc$eqIncome, weights = eusilc$rb050)

Value:
[1] 26.48962
```

For a full list of arguments, the reader is referred to the R help page of the corresponding function.

At this point, it is important to note that the implementation of package **laeken** follows an object-oriented design using S3 classes (Chambers and Hastie 1992). Thus each of the above functions returns an object of a certain class for the respective indicator. All those classes thereby inherit from the class `"indicator"`.

Among other information, the basic class `"indicator"` contains the following components:

`value:` the point estimate

`valueByStratum:` a data frame containing the point estimates for each domain

`var:` the variance estimate

`varByStratum:` a data frame containing the variance estimates for each domain

`ci:` the confidence interval

`ciByStratum:` a data frame containing the confidence intervals for each domain

All indicators inherit the components of class `"indicator"`, as well as the methods that are defined for this basic class, which has the advantage that code can be shared among the set of indicators. However, each indicator also has its own class such that methods unique to the indicator can be defined. Following a common convention for S3 classes, the classes for the indicators have the same names as the functions for computing them. Hence the following classes are implemented in package **laeken**:

- Class `"arpr"` with the following additional components:

  `p`: the percentage of the weighted median used for the at-risk-of-poverty threshold

  `threshold`: the at-risk-of-poverty threshold

- Class `"qsr"` with no additional components

- Class `"rmpg"` with the following additional components:

  `threshold`: the at-risk-of-poverty threshold

- Class `"gini"` with no additional components

- Class `"gpg"` with no additional components

Furthermore, functions to test whether an object is a member of the basic class or one of the subclasses are available. The function to test for the basic class is called is called `is.indicator()`. Similarly, the functions to test for the subclasses are called `is.foo()`, where foo is the name of the corresponding class (e.g., `is.arpr()`).

```
R> a <- arpr("eqIncome", weights = "rb050", data = eusilc)
R> is.arpr(a)

[1] TRUE

R> is.indicator(a)

[1] TRUE

R> class(a)

[1] "arpr"      "indicator"
```

4.2. Estimating the indicators in subdomains

One of the most important features of **laeken** is that indicators can easily be evaluated for different subdomains. These can be regions, but also any other breakdown given by a categorical variable, for instance age categories or gender. All the user needs to do is to specify such a categorical variable via the `breakdown` argument. Note that for the at-risk-of-poverty rate and relative median at-risk-of-poverty gap, the same overall at-risk-of-poverty threshold is used for all subdomains (see Eurostat 2004a, 2009).

In a first example, the overall estimate for the at-risk-of-poverty rate is computed together with more regional estimates.

```
R> arpr("eqIncome", weights = "rb050", breakdown = "db040",
+       data = eusilc)
```

```
Value:
[1] 14.44422

Value by domain:
        stratum    value
1    Burgenland 19.53984
2     Carinthia 13.08627
3 Lower Austria 13.84362
4      Salzburg 13.78734
5        Styria 14.37464
6         Tyrol 15.30819
7 Upper Austria 10.88977
8        Vienna 17.23468
9    Vorarlberg 16.53731

Threshold:
[1] 10859.24
```

With the following lines of code, a breakdown variable with all possible combinations of age categories and gender is defined and added to the data set, before it is used to compute estimates in the corresponding domains.

```
R> ageCat <- cut(eusilc$age, c(-1, 16, 25, 50, 65, Inf), right = FALSE)
R> eusilc$breakdown <- paste(ageCat, eusilc$rb090, sep = ":")
R> arpr("eqIncome", weights = "rb050", breakdown = "breakdown",
+      data = eusilc)

Value:
[1] 14.44422

Value by domain:
           stratum     value
1   [-1,16):female 18.948125
2     [-1,16):male 17.973597
3   [16,25):female 16.703016
4     [16,25):male 16.156673
5   [25,50):female 15.220300
6     [25,50):male  9.638359
7   [50,65):female 12.941125
8     [50,65):male  8.221154
9  [65,Inf):female 21.252184
10   [65,Inf):male 12.046903

Threshold:
[1] 10859.24
```

Clearly, the latter results are even more heterogeneous than the former with the breakdown into regions.

Now we compute the gender pay gap with breakdown according to education. First, we use the weighted mean as an estimate for the average hourly earnings.

```
R> gpg("earningsHour", gender = "sex", weigths = "weights",
+      breakdown = "education", data = ses)

Value:
[1] 0.2517759

Value by domain:
        stratum      value
1 ISCED 0 and 1 0.02347578
2       ISCED 2 0.21265286
3 ISCED 3 and 4 0.22974069
4      ISCED 5A 0.23323499
5      ISCED 5B 0.18445275
```

Note that the weighted mean is heavily influenced by skewness and outliers. In order to more accurately reflect the average in such skewed distributions, the weighted median should be used instead (even though that is not the standard definition according to Eurostat 2004a).

```
R> gpg("earningsHour", gender = "sex", weigths = "weights",
+      breakdown = "education", data = ses, method = "median")

Value:
[1] 0.229818

Value by domain:
        stratum      value
1 ISCED 0 and 1 0.08695672
2       ISCED 2 0.18361501
3 ISCED 3 and 4 0.20080888
4      ISCED 5A 0.25119196
5      ISCED 5B 0.19350093
```

The differences in the both the overall estimates and the estimates for different education levels indicate that the results based on the arithmetic means are possibly distorted due to skewness and outliers. To further investigate the significance of those differences, the respective variances and confidence intervals should be estimated with the methods presented in Section˜6.

4.3. Extracting information using the `subset()` method

If estimates of an indicator have been computed for several subdomains, it may sometimes be desired to extract the results for some domains of particular interest. For this purpose, let us revisit the first example from the previous subsection, where we compute the at-risk-of-poverty rate for regional subdomains.

```
R> a <- arpr("eqIncome", weights = "rb050", breakdown = "db040",
+      data = eusilc)
```

In package **laeken**, extracting a subset from such an indicator can be done with the corresponding `subset()` method. For example, the following command extracts the estimates of the at-risk-of-poverty rate for the regions Lower Austria and Vienna from the object computed above.

```
R> subset(a, strata = c("Lower Austria", "Vienna"))

Value:
[1] 14.44422

Value by domain:
        stratum    value
3 Lower Austria 13.84362
8        Vienna 17.23468

Threshold:
[1] 10859.24
```

It is thereby worth pointing out that not every indicator needs its own `subset()` method due to inheritance from the basic class `"indicator"`.

5. Robust estimation

From a robustness point of view, the standard estimators for many of the social exclusion indicators are problematic when outliers are present in the data. In particular the income inequality indicators quintile share ratio (QSR) and Gini coefficient suffer from a lack of robustness. Point estimates can be highly influenced and variance estimates can be inflated, as demonstrated in a practical application by Alfons, Templ, and Filzmoser (2013).

In economic data, the distributions of variables such as income typically have heavy tails, as well as even more extreme outliers deviating from the rest of the tail. Following Chambers (1986), the observations in the heavy tails can be seen as *representative* and the extreme values as *nonrepresentative* outliers. As the term suggests, representative outliers carry relevant information regarding the population distribution and need to be included in the estimation of quantities of interest. Nonrepresentative outliers, on the other hand, have to be excluded from estimation or downweighted, since they are either incorrectly recorded or can be considered unique in the population in some sense. Note that in the latter case, nonrepresentative outliers very well belong to the true population distribution. Cowell and Flachaire (2007) hence coined the term *high-leverage* observations for such data points.

As a remedy to this problem, heavy tails can be modeled by a Pareto distribution (e.g. Kleiber and Kotz 2003), which combined with robust parameter estimation allows to identify extreme outliers. The idea behind Pareto tail modeling in the context of survey samples is that the upper tail of the population values follow a Pareto distribution. Even though the Pareto distribution is well studied in the literature, sample weights from finite population

sampling are typically not considered in the estimation of its parameters. Therefore, Alfons *et˜al.* (2013) recently adapted promising methods to incorporate sample weights into the estimation process. These methods are reviewed in the remainder of this section and their computation with package **laeken** is demonstrated.

5.1. Pareto distribution

The *Pareto distribution* is defined in terms of its cumulative distribution function

$$F_\theta(x) = 1 - \left(\frac{x}{x_0}\right)^{-\theta}, \qquad x \geq x_0, \tag{13}$$

where $x_0 > 0$ is the scale parameter and $\theta > 0$ is the shape parameter (Kleiber and Kotz 2003). Furthermore, its density function is given by

$$f_\theta(x) = \frac{\theta x_0^\theta}{x^{\theta+1}}, \qquad x \geq x_0. \tag{14}$$

Clearly, the Pareto distribution is a highly right-skewed distribution with a heavy tail.

In Pareto tail modeling, the cumulative distribution function on the whole range of x is then modeled as

$$F(x) = \begin{cases} G(x), & \text{if } x \leq x_0, \\ G(x_0) + (1 - G(x_0))F_\theta(x), & \text{if } x > x_0, \end{cases} \tag{15}$$

where G is an unknown distribution function (Dupuis and Victoria-Feser 2006). For a given survey sample, let $\boldsymbol{x} = (x_1, \ldots, x_n)'$ be the observed values of the variable of interest with $x_1 \leq \ldots \leq x_n$ and $\boldsymbol{w} := (w_i, \ldots, w_n)'$ the corresponding sample weights, where n denotes the total number of observations. In addition, let k denote the number of observations to be used for tail modeling. Note that the estimation of x_0 and k directly correspond with each other. If k is fixed, the threshold is estimated by $\hat{x}_0 = x_{n-k}$. If in turn an estimate $\hat{x}_0$ is obtained, k is given by the number of observations that are larger than $\hat{x}_0$.

In this section, we focus on the EU-SILC example data, where the equivalized disposable income is the main variable of interest. To illustrate the robustness of the presented methods, we replace the equivalized disposable income of the household with the highest income with a large outlier. Note that the resulting income vector is stored in a new variable.

```
R> hID <- eusilc$db030[which.max(eusilc$eqIncome)]
R> eqIncomeOut <- eusilc$eqIncome
R> eqIncomeOut[eusilc$db030 == hID] <- 10000000
```

Moreover, since the equivalized disposable income is a form of household income, the Pareto distribution needs to be modeled on the household level rather than the personal level. Thus we create a data set that only contains the equivalized disposable income with the outlier and the sample weights on the household level.

```
R> keep <- !duplicated(eusilc$db030)
R> eusilcH <- data.frame(eqIncome=eqIncomeOut, db090=eusilc$db090)[keep,]
```

5.2. Pareto quantile plot and finding the threshold

```
R> paretoQPlot(eusilcH$eqIncome, w = eusilcH$db090)
```

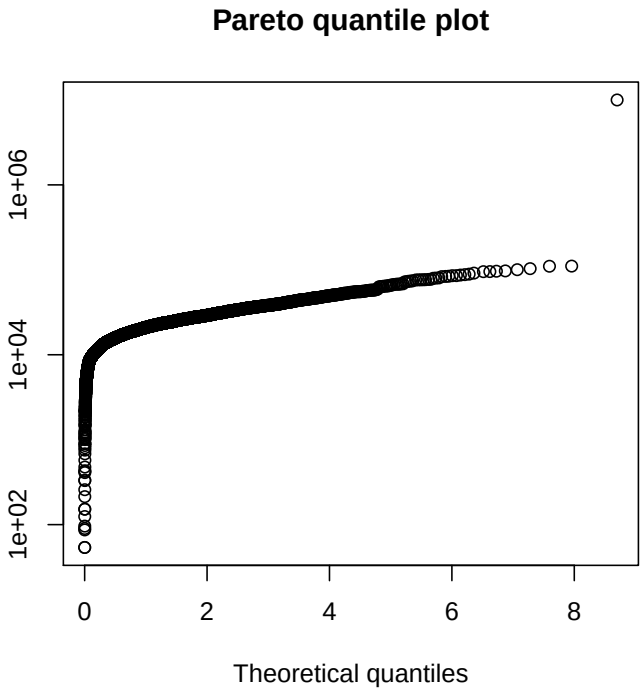

Figure 1: Pareto quantile plot for the EU-SILC example data on the household level with the largest observation replaced by an outlier.

The first step in any practical analysis should be to explore the data with visualization techniques. For our purpose, the *Pareto quantile plot* is a powerful tool to check whether the Pareto model is appropriate. The plot was introduced by Beirlant, Vynckier, and Teugels (1996b) for the case without sample weights, and adapted to take sample weights into account by Alfons *et˜al.* (2013).

The idea behind the Pareto quantile plot is that under the Pareto model, there exists a linear relationship between the logarithms of the observed values and the quantiles of the standard exponential distribution. For survey samples, the observed values are therefore plotted against the quantities

$$-\log\left(1-\frac{\sum_{j=1}^{i} w_j}{\sum_{j=1}^{n} w_j}\frac{n}{n+1}\right), \qquad i=1,\ldots,n. \tag{16}$$

When all sample weights are equal, the correction factor $n/(n+1)$ ensures that (16) reduces to the theoretical quantiles taken on the n inner gridpoints from $n+1$ equally sized subsets of the interval $[0,1]$ (see Alfons *et˜al.* 2013, for details).

In package **laeken**, the Pareto quantile plot is implemented in the function `paretoQPlot()`. Figure˜1 shows the resulting plot for the EU-SILC example data on the household level. Since the tail of the data forms almost a straight line, the Pareto tail model is suitable for the data

at hand.

Moreover, Figure˜1 illustrates the two main advantages that make the Pareto quantile plot so powerful. First, nonrepresentative outliers (i.e., extremely large observations that deviate from the Pareto model) are clearly visible. In our example, the outlier that we introduced into the data set is located far away from the rest of the data in the top right corner of the plot. Second, the leftmost point of a fitted line in the tail of the data can be used as an estimate of the threshold x_0 in the Pareto model, i.e., the scale parameter of fitted Pareto distribution. The slope of the fitted line is then in turn an estimate of $1/\theta$, the reciprocal of the shape parameter. A disadvantage of this graphical method to determine the parameters of the fitted Pareto distribution is of course that it is not very exact.

Nevertheless, the function `paretoQPlot()` allows the user to select the threshold in the Pareto model interactively by clicking on a data point. Information on the selected threshold is thereby printed on the R console. This process can be repeated until the user terminates the interactive session, typically by a secondary mouse click. Then the selected threshold is returned as an object of class `"paretoScale"`, which consists of the component `x0` for the threshold (scale parameter) and the component `k` for the number of observations in the tail (i.e., larger than the threshold).

Van Kerm's rule of thumb

For EU-SILC data, Van˜Kerm (2007) developed a formula for the threshold x_0 in the Pareto model that has more of a rule-of-thumb nature. It is given by

$$\hat{x}_0 := \min(\max(2.5\bar{x}, \hat{q}_{0.98}), \hat{q}_{0.97}), \tag{17}$$

where $\bar{x}$ is the weighted mean, and $\hat{q}_{0.98}$ and $\hat{q}_{0.97}$ are weighted quantiles as defined in Equation˜(1). It is important to note that this formula is designed specifically for the equivalized disposable income in EU-SILC data and can withstand a small number of nonrepresentative outliers.

In **laeken**, the function `paretoScale()` provides functionality for estimating the threshold via Van˜Kerm's formula. Its argument `w` can be used to supply sample weights.

```
R> ts <- paretoScale(eusilcH$eqIncome, w = eusilcH$db090)
R> ts

Threshold: 48459.43
Number of observations in the tail: 119
```

The estimated threshold is again returned as an object of class `"paretoScale"`.

Other methods for finding the threshold

Many procedures for finding the threshold in the Pareto model have been introduced in the literature. For instance, Beirlant, Vynckier, and Teugels (1996a,b) developed an analytical procedure for finding the optimal number of observations in the tail for the maximum likelihood estimator of the shape parameter by minimizing the asymptotic mean squared error (AMSE). This procedure is available in **laeken** through function `minAMSE()`, but is not further

discussed here since it is not robust. Dupuis and Victoria-Feser (2006), on the other hand, proposed a robust prediction error criterion for choosing the optimal number of observations in the tail and the shape parameter simultaneously. Nevertheless, our implementation of this robust criterion is unstable and is therefore not included in **laeken**.

5.3. Estimation of the shape parameter

Once the threshold for the Pareto model is determined, the shape parameter θ can be estimated via the *points over threshold* method, i.e., by fitting the distribution to the k data points that are larger than the threshold. Since our aim is to identify extreme outliers that deviate from the Pareto model, the shape parameter needs to be estimated in a robust way. Two promising estimators are discussed in the following. Both were adjusted for sample weights by Alfons *et˜al.* (2013).

Integrated squared error estimator

The integrated squared error (ISE) criterion was first introduced by Terrell (1990) as a more robust alternative to maximum likelihood estimation. Vandewalle, Beirlant, Christmann, and Hubert (2007) proposed to use this criterion in the context of Pareto tail modeling, but they do not consider sample weights. However, the Pareto distribution is modeled in terms of the *relative excesses*

$$y_i := \frac{x_{n-k+i}}{x_{n-k}}, \qquad i = 1, \ldots, k. \tag{18}$$

Now the density function of the Pareto distribution for the relative excesses is approximated by

$$f_\theta(y) = \theta y^{-(1+\theta)}. \tag{19}$$

With this model density, the integrated squared error criterion can be written as

$$\hat{\theta} = \arg\min_\theta \left[\int f_\theta^2(y) dy - 2\mathbb{E}(f_\theta(Y)) \right], \tag{20}$$

see Vandewalle *et˜al.* (2007). For survey samples, Alfons *et˜al.* (2013) propose to use the weighted mean as an estimator of $\mathbb{E}(f_\theta(Y))$ to obtain the *weighted integrated squared error* (wISE) estimator:

$$\hat{\theta}_{\text{wISE}} = \arg\min_\theta \left[\int f_\theta^2(y) dy - \frac{2}{\sum_{i=1}^k w_{n-k+i}} \sum_{i=1}^k w_{n-k+i} f_\theta(y_i) \right]. \tag{21}$$

The wISE estimator can be computed using the function `thetaISE()`. The arguments `k` and `x0` are available to supply either the number of observations in the tail or the threshold, and sample weights can be supplied via the argument `w`.

```
R> thetaISE(eusilcH$eqIncome, k = ts$k, w = eusilcH$db090)

[1] 3.993801

R> thetaISE(eusilcH$eqIncome, x0 = ts$x0, w = eusilcH$db090)
```

```
[1] 3.993801
```

Partial density component estimator

Following the observation by Scott (2004) that f_θ in the ISE criterion does not need to be a real density, Vandewalle *et˜al.* (2007) proposed to minimize the ISE criterion based on an incomplete density mixture model uf_θ instead. Alfons *et˜al.* (2013) generalized their estimator to take sample weights into account, yielding the *weighted partial density component* (wPDC) estimator

$$\hat{\theta}_{\text{wPDC}} = \arg\min_\theta \left[u^2 \int f_\theta^2(y)dy - \frac{2u}{\sum_{i=1}^k w_{n-k+i}} \sum_{i=1}^k w_{n-k+i} f_\theta(y_i) \right] \tag{22}$$

with

$$\hat{u} = \frac{1}{\sum_{i=1}^k w_{n-k+i}} \sum_{i=1}^k w_{n-k+i} f_{\hat{\theta}}(y_i) \Bigg/ \int f_{\hat{\theta}}^2(y)dy. \tag{23}$$

Based on extensive simulation studies, Alfons *et˜al.* (2013) conclude that the wPDC estimator is favorable over the wISE estimator due to better robustness properties.

The function `thetaPDC()` is implemented in package **laeken** to compute the wPDC estimator. As before, it is necessary to supply either the number of observations in the tail via the argument `k`, or the threshold via the argument `x0`. Sample weights can be supplied using the argument `w`.

```
R> thetaPDC(eusilcH$eqIncome, k = ts$k, w = eusilcH$db090)
```

```
[1] 4.132596
```

```
R> thetaPDC(eusilcH$eqIncome, x0 = ts$x0, w = eusilcH$db090)
```

```
[1] 4.132596
```

Other estimators for the shape parameter

Many other estimators for the shape parameter are implemented in package **laeken**, e.g., the maximum likelihood estimator (Hill 1975) or the more robust weighted maximum likelihood estimator (Dupuis and Morgenthaler 2002). However, those estimators are either not robust or have not (yet) been adapted for sample weights and are therefore not further discussed in this paper.

5.4. Robust estimation of the indicators via Pareto tail modeling

The basic idea for robust estimation of the indicators is to first detect nonrepresentative outliers based on the Pareto model. Afterwards their influence on the indicators is reduced by either downweighting the outliers and recalibrating the remaining observations, or by replacing the outlying values with values from the fitted distribution. The main advantage of this general approach is that it can be applied to any indicator.

With the fitted Pareto distribution $F_{\hat{\theta}}$, nonrepresentative outliers can now be detected as observations being larger than a certain $F_{\hat{\theta}}^{-1}(1-\alpha)$ quantile. From extensive simulation studies (Hulliger *et˜al.* 2011; Alfons *et˜al.* 2013), $\alpha = 0.005$ or $\alpha = 0.01$ are seem suitable choices for this tuning parameter. Then the following approaches are implemented in **laeken** to reduce the influence of the outliers:

Calibration of nonrepresentative outliers (CN) As nonrepresentative outliers are considered to be somewhat unique to the population data, the sample weights of the corresponding observations are set to 1. The weights of the remaining observations are adjusted accordingly by calibration (see, e.g., Deville, Särndal, and Sautory 1993).

Replacement of nonrepresentative outliers (RN) The outliers are replaced by values drawn from the fitted distribution $F_{\hat{\theta}}$, thereby preserving the order of the original values.

Shrinkage of nonrepresentative outliers (SN) The outliers are shrunken to the theoretical quantile $F_{\hat{\theta}}^{-1}(1-\alpha)$ used for outlier detection.

A more mathematical formulation and further details on the CN and RN approaches can be found in Alfons *et˜al.* (2013), who advocate the CN approach in combination with the wPDC estimator for fitting the Pareto distribution.

For a practical analysis with package **laeken**, let us first revisit the estimation of the shape parameter. Rather than applying a function such as `thetaPDC()` directly as in the previous subsection, the function `paretoTail()` should be used to fit the Pareto distribution to the upper tail of the data. It returns an object of class `"paretoTail"`, which contains all necessary information for further analysis with one of the approaches described above.

```
R> fit <- paretoTail(eqIncomeOut, k = ts$k, w = eusilc$db090,
+      groups = eusilc$db030)
```

Note that the household IDs are supplied via the argument `groups` such that the Pareto distribution is fitted on the household level rather than the individual level. By default, the wPDC is used to estimate the shape parameter, but other estimators can be specified via the `method` argument. In addition, the tuning parameter α for outlier detection can be supplied as argument `alpha`.

Moreover, the `plot()` method for `"paretoTail"` objects produces a Pareto quantile plot (see Section˜5.2) with additional diagnostic information. Figure˜2 contains the resulting plot for the object computed above. The lower horizontal dotted line corresponds to the estimated threshold $\hat{x}_0$, whereas the slope of the solid grey line is given by the reciprocal of the estimated shape parameter $\hat{\theta}$. Furthermore, the upper horizontal dotted line represents the theoretical quantile used for outlier detection. In this example, the threshold seems somewhat too high. Nevertheless, the estimate of the shape parameter is accurate and the cutoff point for outlier detection is appropriate, resulting in correct identification of the outlier that we added to the data set.

For downweighting nonrepresentative outliers, the function `reweightOut()` is available. It returns a vector of the recalibrated weights. In the command below, we use regional information as auxiliary variables for calibration. The function `calibVars()` thereby transforms a factor into a matrix of binary variables. The returned recalibrated weights are then simply used to estimate the Gini coefficient with function `gini()`.

```
R> plot(fit)
```

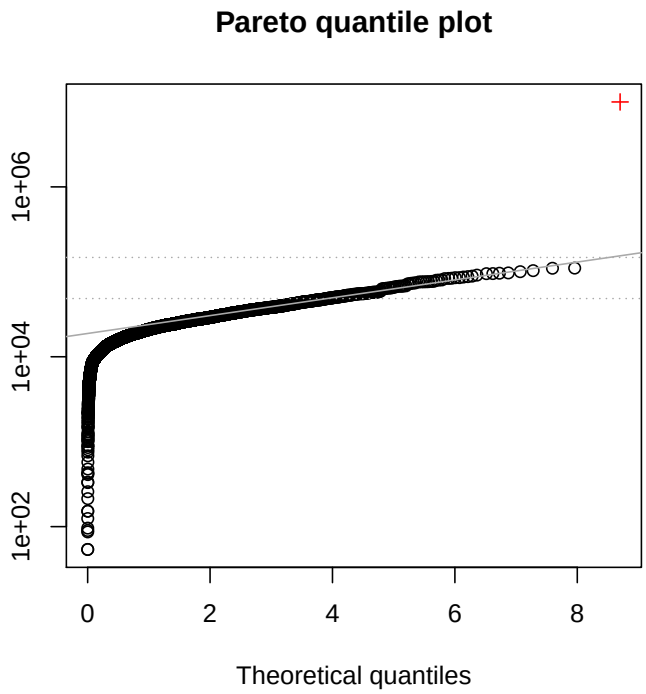

Figure 2: Pareto quantile plot for the EU-SILC example data with additional diagnostic information on the fitted distribution and any detected outliers.

```
R> w <- reweightOut(fit, calibVars(eusilc$db040))
R> gini(eqIncomeOut, w)

Value:
[1] 26.45973
```

To replace the nonrepresentative outliers with values drawn from the fitted distribution, the function `replaceOut()` is implemented. For reproducible results, the seed of the random number generator is set beforehand. The returned income vector is then supplied to `gini()` to estimate the Gini coefficient.

```
R> set.seed(123)
R> eqIncomeRN <- replaceOut(fit)
R> gini(eqIncomeRN, weights = eusilc$rb050)

Value:
[1] 26.4645
```

Similarly, the function `shrinkOut()` can be used to shrink the nonrepresentative outliers to the theoretical quantile used for outlier detection.

```
R> eqIncomeSN <- shrinkOut(fit)
R> gini(eqIncomeSN, weights = eusilc$rb050)

Value:
[1] 26.48831
```

All three robust estimates are very close to the original value before the outlying household had been introduced (see Section˜3.2). For comparison, we compute the standard estimate of Gini coefficient with the income vector including the outlying household.

```
R> gini(eqIncomeOut, weights = eusilc$rb050)

Value:
[1] 29.24333
```

Clearly, the standard estimate shows an unreasonably large influence of only one outlying household, illustrating the need for the robust methods.

6. Variance estimation

When computing point estimates of indicators from samples, variance estimates and confidence intervals should be computed as well in order to account for variability due to sampling. Additional variability coming from, e.g., editing or imputation may need to be considered as well, but this is not further discussed in this paper.

The most common indicators for measuring poverty, social cohesion and gender inequality are nonlinear, nonsmooth estimators. Measuring the accuracy of such estimators via standard errors and confidence intervals requires to apply appropriate variance estimation methods such as linearization or resampling. Resampling methods can thereby be applied in a more general manner since their computation is the same for any estimator. Linearization formulas, on the other hand, need to be derived for each estimator individually.

Concerning resampling methods, it is well known that bootstrap methods in general provide better estimates for nonsmooth estimators than other other techniques such as jackknifing or balanced repeated replication (e.g., Bruch, Münnich, and Zins 2011). Nevertheless, the bootstrap may result in biased estimates when the sample weights are not properly considered. Keep in mind that resampling methods should mimic the true sampling design in each replication. Strictly speaking, the bootstrap is therefore suitable for stratified cluster sampling designs, which are commonly used in EU-SILC and many other surveys. From a practical point of view, however, the naive bootstrap is quite fast to compute and provides reasonable estimates whenever there is not much variation in the sample weights, which is for example typically the case for EU-SILC data. If there is larger variation among the sample weights, a calibrated bootstrap should be applied.

Naive and calibrated bootstrap methods are implemented in package **laeken** and are described in Sections˜6.1 and˜6.2, respectively. As discussed above, it should be noted that very complex sampling designs may require more complex variance estimation procedures. In R, further variance estimation techniques for complex survey samples are available in other packages.

For instance, package **EVER** (Zardetto 2012) provides functionality for the delete-a-group jackknife. Other methods such as balanced repeated replication are implemented in package **survey** (Lumley 2004, 2012). The incorporation of those packages for additional variance estimation procedures is therefore considered for future work.

6.1. Naive bootstrap

Let τ denote a certain indicator of interest and let $\boldsymbol{X} := (\mathbf{x}_1, \ldots, \mathbf{x}_n)'$ be a survey sample with n observations. Then the *naive bootstrap* algorithm for estimating the variance and confidence interval of an estimate $\hat{\tau}(\boldsymbol{X})$ of the indicator can be summarized as follows:

1. Draw R independent bootstrap samples $\boldsymbol{X}_1^*, \ldots, \boldsymbol{X}_R^*$ from $\boldsymbol{X}$. For stratified sampling designs, resampling is performed within each stratum independently.

2. Compute the bootstrap replicate estimates $\hat{\tau}_r^* := \hat{\tau}(\boldsymbol{X}_r^*)$ for each bootstrap sample $\boldsymbol{X}_r^*$, $r = 1, \ldots, R$, taking the sample weights from the respective bootstrap samples into account.

3. Estimate the variance $V(\hat{\tau})$ by the variance of the R bootstrap replicate estimates:

$$\hat{V}(\hat{\tau}) := \frac{1}{R-1} \sum_{r=1}^{R} \left(\hat{\tau}_r^* - \frac{1}{R} \sum_{s=1}^{R} \hat{\tau}_s^* \right)^2 . \tag{24}$$

4. Estimate the confidence interval at confidence level $1-\alpha$ by one of the following methods (for details, see Davison and Hinkley 1997):

   Percentile method: $\left[\hat{\tau}^*_{((R+1)\frac{\alpha}{2})}, \hat{\tau}^*_{((R+1)(1-\frac{\alpha}{2}))} \right]$, as suggested by Efron and Tibshirani (1993)

   Normal approximation: $\hat{\tau} \pm z_{1-\frac{\alpha}{2}} \cdot \hat{V}(\hat{\tau})^{1/2}$ with $z_{1-\frac{\alpha}{2}} = \Phi^{-1}(1-\frac{\alpha}{2})$

   Basic bootstrap method: $\left[2\hat{\tau} - \hat{\tau}^*_{((R+1)(1-\frac{\alpha}{2}))}, 2\hat{\tau} - \hat{\tau}^*_{((R+1)\frac{\alpha}{2})} \right]$

   For the percentile and the basic bootstrap method, $\hat{\tau}^*_{(1)} \leq \ldots \leq \hat{\tau}^*_{(R)}$ denote the order statistics of the bootstrap replicate estimates.

With package **laeken**, variance estimates and confidence intervals can easily be included in the estimation of an indicator, it is only necessary to specify a few more arguments in the call to the function computing the indicator. The argument `var` is available to specify the type of variance estimation, although only the bootstrap is currently implemented. Furthermore, the significance level α for the confidence intervals can be supplied via the argument `alpha` (the default is to use `alpha=0.05` for 95% confidence intervals). Additional arguments are then passed to the underlying function for variance estimation.

```
R> arpr("eqIncome", weights = "rb050", design = "db040", data = eusilc,
+       var = "bootstrap", bootType = "naive", seed = 123)

Value:
[1] 14.44422
```

```
Variance:
[1] 0.0920564

Confidence interval:
   lower    upper
13.87663 15.19417

Threshold:
[1] 10859.24
```

For the bootstrap, the function `bootVar()` is called internally for variance and confidence interval estimation. Important arguments are `design` for specifying the strata in the sampling design, `R` for supplying the number of bootstrap replicates, `bootType` for specifying the type of bootstrap estimator, and `ciType` for specifying the type of confidence interval. For reproducibility, the seed of the random number generator can be set via the argument `seed`.

An important feature of package **laeken** is that indicators can be estimated for different subdomains with a single command, which still holds for variance and confidence interval estimation. Using the `breakdown` argument, the example below produces estimates for each gender in addition to the overall values.

```
R> arpr("eqIncome", weights = "rb050", breakdown = "rb090",
+       design = "db040", data = eusilc, var = "bootstrap",
+       bootType = "naive", seed = 123)

Value:
[1] 14.44422

Variance:
[1] 0.0920564

Confidence interval:
   lower    upper
13.87663 15.19417

Value by domain:
  stratum    value
1    male 12.02660
2  female 16.73351

Variance by domain:
  stratum       var
1    male 0.1359585
2  female 0.1837318

Confidence interval by domain:
  stratum    lower    upper
```

```
1    male 11.29858 12.75120
2  female 16.00919 17.72657

Threshold:
[1] 10859.24
```

6.2. Calibrated bootstrap

In the practice of survey sampling, the initial sample weights from the sampling design are often adjusted by calibration, for instance to account for non-response or to ensure that certain known population totals can be precisely estimated from the survey sample. To give a simplified example, if the population sizes for different age groups are known, the sample weights may be calibrated such that the sums of the sample weights for all observations within the respective age groups equal the known true values. However, drawing a bootstrap sample then has the effect that the sample weights in the bootstrap sample no longer sum up to the correct values. As a remedy, the sample weights of each bootstrap sample should be recalibrated. Detailed information on calibration methods can be found in, e.g., Deville and Särndal (1992); Deville *et˜al.* (1993).

The naive bootstrap does not include the recalibration of bootstrap samples and therefore is, strictly speaking, not suitable for many practical applications. Nevertheless, the naive bootstrap still works well in many situations even though a bias may be introduced. Hence it is frequently used in practice due to its faster computation compared with the calibrated version.

For better accuracy at a higher computational cost, the *calibrated bootstrap* algorithm is obtained by adding the following step between Steps˜1 and˜2 of the naive bootstrap algorithm from the previous section:

1b. Calibrate the sample weights for each bootstrap sample $\boldsymbol{X}_r^*$, $r = 1, \ldots, R$.

Using **laeken**, the function call for including variance and confidence intervals via the calibrated bootstrap is very similar to its counterpart for the naive bootstrap. A matrix of auxiliary calibration variables needs to be supplied via the argument `X`. The function `calibVars()` can thereby by used to transform a factor into a matrix of binary variables. In the examples below, information on region and gender is used for calibration. Furthermore, the argument `totals` can be used to supply the corresponding population totals. If the `totals` argument is omitted, the population totals are computed from the sample weights of the original sample. This follows the assumption that those weights are already calibrated on the supplied auxiliary variables.

```
R> aux <- cbind(calibVars(eusilc$db040), calibVars(eusilc$rb090))
R> arpr("eqIncome", weights = "rb050", design = "db040", data = eusilc,
+       var = "bootstrap", X = aux, seed = 123)

Value:
[1] 14.44422
```

```
Variance:
[1] 0.09077282

Confidence interval:
   lower    upper
13.87605 15.17314

Threshold:
[1] 10859.24

R> arpr("eqIncome", weights = "rb050", breakdown = "rb090",
+       design = "db040", data = eusilc, var = "bootstrap",
+       X = aux, seed = 123)

Value:
[1] 14.44422

Variance:
[1] 0.09077282

Confidence interval:
   lower    upper
13.87605 15.17314

Value by domain:
  stratum    value
1    male 12.02660
2  female 16.73351

Variance by domain:
  stratum       var
1    male 0.1345168
2  female 0.1810947

Confidence interval by domain:
  stratum    lower    upper
1    male 11.32096 12.73850
2  female 16.01951 17.73165

Threshold:
[1] 10859.24
```

7. Conclusions

In this paper, we demonstrate the use of the R package **laeken** for computing point and variance estimates of indicators from complex surveys. Various commonly used indicators on

social exclusion and poverty are thereby implemented. Their estimation is made easy with the package, as the corresponding functions allow to compute point and variance estimates with a single command, even for different subdomains of the data.

In addition, we illustrate with a simple example that some of the indicators are highly influenced by extreme outliers in the data (cf. Hulliger and Schoch 2009; Alfons *et˜al.* 2013). As a remedy, a general procedure for robust estimation of the indicators is implemented in **laeken**. The procedure is based on fitting a Pareto distribution to the upper tail of the data and has the advantage that it can be applied to any indicator. A diagnostic plot thereby allows to check whether the Pareto tail model is appropriate for the data at hand.

Acknowledgments

This work was partly funded by the European Union (represented by the European Commission) within the 7$^{\text{th}}$ framework programme for research (Theme˜8, Socio-Economic Sciences and Humanities, Project AMELI (Advanced Methodology for European Laeken Indicators), Grant Agreement No. 217322). Visit http://ameli.surveystatistics.net for more information on the project.

References

Alfons A (2012). ***simFrame**: Simulation Framework*. R package version˜0.5.0, URL http://CRAN.R-project.org/package=simFrame.

Alfons A, Holzer J, Templ M (2012). ***laeken**: Estimation of Indicators on Social Exclusion and Poverty*. R package version 0.4.0, URL http://CRAN.R-project.org/package=laeken.

Alfons A, Kraft S (2012). ***simPopulation**: Simulation of Synthetic Populations for Surveys based on Sample Data*. R package version˜0.4.0, URL http://CRAN.R-project.org/package=simPopulation.

Alfons A, Kraft S, Templ M, Filzmoser P (2011). "Simulation of Close-to-Reality Population Data for Household Surveys With Application to EU-SILC." *Statistical Methods & Applications*, **20**(3), 383–407.

Alfons A, Templ M, Filzmoser P (2010). "An Object-Oriented Framework for Statistical Simulation: The R package **simFrame**." *Journal of Statistical Software*, **37**(3), 1–36. URL http://www.jstatsoft.org/v37/i03.

Alfons A, Templ M, Filzmoser P (2013). "Robust Estimation of Economic Indicators from Survey Samples based on Pareto Tail Modeling." *Journal of the Royal Statistical Society, Series C*. In press.

Beblot M, Beniger D, Heinze A, Laisney F (2003). "Methodological Issues Related to the Analysis of Gender Gaps in Employment, Earnings and Career Progression." *Technical report*, Employment and Social Affairs DG, European Commission.

Beirlant J, Vynckier P, Teugels J (1996a). "Excess Functions and Estimation of the Extreme-Value Index." *Bernoulli*, **2**(4), 293–318.

Beirlant J, Vynckier P, Teugels J (1996b). "Tail Index Estimation, Pareto Quantile Plots, and Regression Diagnostics." *Journal of the American Statistical Association*, **31**(436), 1659–1667.

Bruch C, Münnich R, Zins S (2011). "Variance Estimation for Complex Surveys." *Deliverable D3.1*, AMELI Project. URL http://ameli.surveystatistics.net.

Chambers J, Hastie T (1992). *Statistical Models in S*. Chapman & Hall, London, UK. ISBN 9780412830402.

Chambers R (1986). "Outlier Robust Finite Population Estimation." *Journal of the American Statistical Association*, **81**(396), 1063–1069.

Cowell F, Flachaire E (2007). "Income Distribution and Inequality Measurement: The Problem of Extreme Values." *Journal of Econometrics*, **141**(2), 1044–1072.

Davison A, Hinkley D (1997). *Bootstrap Methods and their Applications*. Cambridge University Press. ISBN 0 521 57471 4.

Deville JC, Särndal CE (1992). "Calibration Estimators in Survey Sampling." *Journal of the American Statistical Association*, **87**(418), 376–382.

Deville JC, Särndal CE, Sautory O (1993). "Generalized Raking Procedures in Survey Sampling." *Journal of the American Statistical Association*, **88**(423), 1013–1020.

Dupuis D, Morgenthaler S (2002). "Robust Weighted Likelihood Estimators with an Application to Bivariate Extreme Value Problems." *The Canadian Journal of Statistics*, **30**(1), 17–36.

Dupuis D, Victoria-Feser MP (2006). "A Robust Prediction Error Criterion for Pareto Modelling of Upper Tails." *The Canadian Journal of Statistics*, **34**(4), 639–658.

Efron B, Tibshirani R (1993). *An Introduction to the Bootstrap*. Chapman & Hall, New York, New York. ISBN 0-412-04231-2.

Eurostat (2004a). "Common Cross-Sectional EU Indicators based on EU-SILC; the Gender Pay Gap." *EU-SILC 131-rev/04*, Unit D-2: Living conditions and social protection, Directorate D: Single Market, Employment and Social statistics, Eurostat, Luxembourg.

Eurostat (2004b). "Description of Target Variables: Cross-sectional and Longitudinal." *EU-SILC 065/04*, Unit E-2: Living conditions, Directorate E: Social and regional statistics and geographical information system, Eurostat, Luxembourg.

Eurostat (2006). "Structure of Earnings Survey 2006: Eurostat's arrangements for implementing the Council Regulation 530/1999, the Commission Regulations 1916/2000 and 1738/2005." *Technical report*, Unit F-2: Labour markes statistics, Directorate F: Social statistics, Eurostat.

Eurostat (2009). "Algorithms to Compute Social Inclusion Indicators based on EU-SILC and Adopted under the Open Method of Coordination (OMC)." *Doc. LC-ILC/39/09/EN-rev.1*, Unit F-3: Living conditions and social protection, Directorate F: Social and information society statistics, Eurostat, Luxembourg.

Hill B (1975). "A Simple General Approach to Inference about the Tail of a Distribution." *The Annals of Statistics*, **3**(5), 1163–1174.

Hulliger B, Alfons A, Bruch C, Filzmoser P, Graf M, Kolb JP, Lehtonen R, Lussmann D, Meraner A, Münnich R, Nedyalkova D, Schoch T, Templ M, Valaste M, Veijanen A, Zins S (2011). "Report on the Simulation Results." *Deliverable D7.1*, AMELI Project. URL http://ameli.surveystatistics.net.

Hulliger B, Schoch T (2009). "Robustification of the Quintile Share Ratio." New Techniques and Technologies for Statistics, Brussels.

Kleiber C, Kotz S (2003). *Statistical Size Distributions in Economics and Actuarial Sciences*. John Wiley & Sons, Hoboken, New Jersey. ISBN 0-471-15064-9.

Lumley T (2004). "Analysis of Complex Survey Samples." *Journal of Statistical Software*, **9**(1), 1–19. URL http://www.jstatsoft.org/v09/i08.

Lumley T (2010). *Complex Surveys: A Guide to Analysis using R*. John Wiley & Sons, Hoboken, New Jersey. ISBN 978-0-470-28430-8.

Lumley T (2012). ***survey**: Analysis of Complex Survey Samples*. R package version 3.28-2, URL http://CRAN.R-project.org/package=survey.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Scott D (2004). "Partial Mixture Estimation and Outlier Detection in Data and Regression." In M˜Hubert, G˜Pison, A˜Struyf, S˜Van˜Aelst (eds.), *Theory and Applications of Recent Robust Methods*, pp. 297–306. Birkhäuser, Basel. ISBN 3-7643-7060-2.

Templ M, Alfons A, Filzmoser P (2012a). "Exploring Incomplete Data using Visualization Techniques." *Advances in Data Analyis and Classification*, **6**(1), 29–47.

Templ M, Alfons A, Kowarik A, Prantner B (2012b). ***VIM**: Visualization and Imputation of Missing Values*. R package version˜3.0.2, URL http://CRAN.R-project.org/package=VIM.

Templ M, Kowarik A, Filzmoser P (2011). "Iterative Stepwise Regression Imputation using Standard and Robust Methods." *Computational Statistics & Data Analysis*, **55**(10), 2793–2806.

Terrell G (1990). "Linear Density Estimates." In *Proceedings of the Statistical Computing Section*, pp. 297–302. American Statistical Association.

Tillé Y, Matei A (2012). ***sampling**: Survey Sampling*. R package version 2.5, URL http://CRAN.R-project.org/package=sampling.

Van˜Kerm P (2007). "Extreme Incomes and the Estimation of Poverty and Inequality Indicators from EU-SILC." *IRISS Working Paper Series 2007-01*, CEPS/INSTEAD.

Vandewalle B, Beirlant J, Christmann A, Hubert M (2007). "A Robust Estimator for the Tail Index of Pareto-type Distributions." *Computational Statistics & Data Analysis*, **51**(12), 6252–6268.

Weinberg D (2007). "Earnings by Gender: Evidence from Census 2000." *Monthly Labor Review Online*, **130**(July/August), 26–34.

Zardetto D (2012). ***EVER**: Estimation of Variance by Efficient Replication*. R package version 1.2, URL http://CRAN.R-project.org/package=EVER.

Affiliation:

Andreas Alfons
Faculty of Business and Economics
KU Leuven
3000 Leuven, Belgium
E-mail: andreas.alfons@kuleuven.be
URL: http://www.econ.kuleuven.be/andreas.alfons/public/

Matthias Templ
Department of Statistics and Probability Theory
Vienna University of Technology
1040 Vienna, Austria
E-mail: matthias@data-analysis.at
URL: http://www.data-analysis.at