Standard Methods for Point Estimation of Indicators on Social Exclusion and Poverty using the R Package laeken

Matthias Templ[1], Andreas Alfons[2]

Abstract This vignette demonstrates the use of the R package **laeken** for standard point estimation of indicators on social exclusion and poverty according to the definitions by Eurostat. The package contains synthetically generated data for the European Union Statistics on Income and Living Conditions (EU-SILC), which is used in the code examples throughout the paper. Furthermore, the basic object-oriented design of the package is discussed. Even though the paper is focused on showing the functionality of package **laeken**, it also provides a brief mathematical description of the implemented indicators.

1 Introduction

The *European Union Statistics on Income and Living Conditions* (EU-SILC) is a panel survey conducted in EU member states and other European countries, and serves as basis for measuring risk-of-poverty and social cohesion in Europe. A short overview of the 11 most important indicators on social exclusion and poverty according to Eurostat (2004) is given in the following.

Primary indicators

1. At-risk-of-poverty rate (after social transfers)
   a. At-risk-of-poverty rate by age and gender
   b. At-risk-of-poverty rate by most frequent activity status and gender
   c. At-risk-of-poverty rate by household type
   d. At-risk-of-poverty rate by accommodation tenure status
   e. At-risk-of-poverty rate by work intensity of the household
   f. At-risk-of-poverty threshold (illustrative values)
2. Inequality of income distribution: S80/S20 income quintile share ratio
3. At-persistent-risk-of-poverty rate by age and gender (60% median)
4. Relative median at-risk-of-poverty gap, by age and gender

[1] Zurich University of Applied Sciences
E-mail: matthias.templ@zhaw.ch

[2] Erasmus School of Economics, Erasmus University Rotterdam
E-mail: alfons@ese.eur.nl

Secondary indicators

5. Dispersion around the at-risk-of-poverty threshold
6. At-risk-of-poverty rate anchored at a moment in time
7. At-risk-of-poverty rate before social transfers by age and gender
8. Inequality of income distribution: Gini coefficient
9. At-persistent-risk-of-poverty rate, by age and gender (50% median)

Other indicators

10. Mean equivalized disposable income
11. The gender pay gap

Note that especially the Gini coefficient is very well studied due to its importance in many fields of research.

The add-on package **laeken** (Alfons et al. 2013) aims is to bring functionality for the estimation of indicators on social exclusion and poverty to the statistical environment R (R Development Core Team 2013). In the examples in this vignette, standard estimates for the most important indicators are computed according to the Eurostat definitions (Eurostat 2004, 2009). More sophisticated methods that are less influenced by outliers are described in vignette `laeken-pareto` (Alfons et al. 2011b), while the basic framework for variance estimation is discussed in vignette `laeken-variance` (Templ and Alfons 2011). Those documents can be viewed from within R with the following commands:

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

Morover, a general introduction to package **laeken** is published as Alfons and Templ (2013).

The example data set of package **laeken**, which is called `eusilc` and consists of 14 827 observations from 6 000 households, is used throughout the paper. It was synthetically generated from Austrian EU-SILC survey data from 2006 using the data simulation methodology proposed by Alfons et al. (2011a) and implemented in the R package **simPopulation** (Alfons and Kraft 2012). The first three observations of the synthetic data set `eusilc` are printed below.

```
R> library("laeken")
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
```

Only a few of the large number of variables in the original survey are included in the example data set. The variable names are rather cryptic codes, but these are the standardized names used by the statistical agencies. Furthermore, the variables `hsize` (household size), `age`, `eqSS` (equivalized household size) and `eqIncome` (equivalized disposable income) are not included in the standardized format of EU-SILC data, but have been derived from other variables for convenience. Moreover, some very sparse income components were not included in the the generation of this synthetic data set. Thus the equivalized household income is computed from the available income components.

For the remainder of the paper, the variable `eqIncome` (equivalized disposable income) is of main interest. Other variables are in some cases used to break down the data in order to evaluate the indicators on the resulting subsets.

It is important to note that EU-SILC data are in practice conducted through complex sampling designs with different inclusion probabilities for the observations in the population, which results in different weights for the observations in the sample. Furthermore, calibration is typically performed for non-response adjustment of these initial design weights. Therefore, the sample weights have to be considered for all estimates, otherwise biased results are obtained.

The rest of the paper is organized as follows. Section 2 briefly illustrates the basic object-oriented design of the package. The calculation of the equivalized household size and the equivalized disposable income is then described in Section 3. Afterwards, Section 4 introduces the Eurostat definitions of the weighted median and weighted quantiles, which are required for the estimation of some of the indicators. In Section 5, a mathematical description of the most important indicators on social exclusion and poverty is given and their estimation with package **laeken** is demonstrated. Section 6 discusses a useful subsetting method, and Section 7 concludes.

2 Basic design of the package

The implementation of the package follows an object-oriented design using S3 classes (Chambers and Hastie 1992). Its aim is to provide functionality for point and variance estimation of Laeken indicators with a single command, even for different years and domains. Currently, the following indicators are available in the R package **laeken**:

- *At-risk-of-poverty rate*: function `arpr()`
- *Quintile share ratio*: function `qsr()`
- *Relative median at-risk-of-poverty gap*: function `rmpg()`
- *Dispersion around the at-risk-of-poverty threshold*: also function `arpr()`
- *Gini coefficient*: function `gini()`

Note that the implementation strictly follows the Eurostat definitions (Eurostat 2004, 2009).

2.1 Class structure

In this section, the class structure of package **laeken** is briefly discussed. Section 2.1.1 describes the basic class `"indicator"`, while the different subclasses for the specific indicators are listed in Section 2.1.2.

2.1.1 Class `"indicator"`

The basic class `"indicator"` acts as the superclass for all classes in the package corresponding to specific indicators. It consists of the following components:

`value`: A numeric vector containing the point estimate(s).

`valueByStratum`: A `data.frame` containing the point estimates by domain.

`varMethod`: A character string specifying the type of variance estimation used.

`var`: A numeric vector containing the variance estimate(s).

`varByStratum`: A `data.frame` containing the variance estimates by domain.

`ci`: A numeric vector or matrix containing the confidence interval(s).

`ciByStratum`: A `data.frame` containing the confidence intervals by domain.

`alpha`: The confidence level is given by 1−`alpha`.

`years`: A numeric vector containing the different years of the survey.

`strata`: A character vector containing the different strata of the breakdown.

These list components are inherited by each indicator in the package. One of the most important features of **laeken** is that indicators can be evaluated for different years and domains. The latter of which can be regions (e.g., NUTS2), but also any other breakdown given by a categorical variable (see the examples in Section 5).

In any case, the advantage of the object-oriented implementation is the possibility of sharing code among the indicators. To give an example, the following methods for the basic class `"indicator"` are implemented in the package:

```
R> methods(class="indicator")

[1] bootVar print   subset
see '?methods' for accessing help and source code
```

The `print()` and `subset()` methods are called by their respective generic functions if an object inheriting from class `"indicator"` is supplied. While the `print()` method defines the output of objects inheriting from class `"indicator"` shown on the R console, the `subset()` method allows to extract subsets of an object inheriting from class `"indicator"` and is discussed in detail in Section 6. Furthermore, the function `is.indicator()` is available to test whether an object is of class `"indicator"`.

2.1.2 Additional classes

For the specific indicators on social exclusion and poverty, 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(s).

- Class `"qsr"` with no additional components.

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

  `threshold`: The at-risk-of-poverty threshold(s).

- Class `"gini"` with no additional components.

All these classes are subclasses of the basic class `"indicator"` and therefore inherit all its components and methods. In addition, functions to test whether an object is a member of one of these subclasses are implemented. Similarly to `is.indicator()`, these are called `is.foo()`, where `foo` is the name of the respective class (e.g., `is.arpr()`).

3 Calculation of the equivalized disposable income

For each person, the 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 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 (Eurostat 2004, 2009).

In practice, the equivalized disposable income needs to be computed from the income components included in EU-SILC for the estimation of the indicators on social exclusion and poverty. Therefore, this section outlines how to perform this step with package **laeken**, even though the variable `eqIncome` containing the equivalized disposable income is already available in the example data set `eusilc`. Note that not all variables that are required for an exact computation of the equivalized income are included in the synthetic example data. However, the functions of the package can be applied in exactly the same manner to real EU-SILC data.

First, the equivalized household size according to the modified OECD scale needs to be computed. This can be done with the function `eqSS()`, which requires the household ID and the age of the individuals as arguments. In the example data, household ID and age are stored in the variables `db030` and `age`, respectively. It should be noted that the variable `age` is not in the standardized format of EU-SILC data and needs to be calculated from the data beforehand. Nevertheless, these computations are very simple and are therefore not shown here (for details, see Eurostat 2009). The following two lines of code calculate the equivalized household size, add it to the data set, and print the first eight observations of the variables involved.

```
R> eusilc$eqSS <- eqSS("db030", "age", data=eusilc)
R> head(eusilc[,c("db030", "age", "eqSS")], 8)

  db030 age eqSS
1     1  34  1.8
2     1  39  1.8
3     1   2  1.8
4     2  38  2.1
5     2  43  2.1
6     2  11  2.1
7     2   9  2.1
8     3  26  1.0
```

Then the equivalized disposable income can be computed with the function `eqInc()`. It requires the following information to be supplied: the household ID, the household income components to be added and subtracted, respectively, the personal income components to be added and subtracted, respectively, as well as the equivalized household size. With the following commands, the equivalized disposable income is calculated and added to the data set, after which the first eight observations of the important variables in this context are printed.

```
R> hplus <- c("hy040n", "hy050n", "hy070n", "hy080n", "hy090n", "hy110n")
R> hminus <- c("hy130n", "hy145n")
R> pplus <- c("py010n", "py050n", "py090n", "py100n",
+      "py110n", "py120n", "py130n", "py140n")
R> eusilc$eqIncome <- eqInc("db030", hplus, hminus,
+      pplus, character(), "eqSS", data=eusilc)
R> head(eusilc[,c("db030", "eqSS", "eqIncome")], 8)

  db030 eqSS eqIncome
1     1  1.8 16090.69
2     1  1.8 16090.69
3     1  1.8 16090.69
4     2  2.1 27076.24
5     2  2.1 27076.24
6     2  2.1 27076.24
7     2  2.1 27076.24
8     3  1.0 19659.53
```

Note that the net income is considered in this example, therefore no personal income component needs to be subtracted (see Eurostat 2004, 2009). This is reflected in the call to `eqInc()` by the use of an empty character vector `character()` for the corresponding argument.

4 Weighted median and quantile estimation

Some of the indicators on social exclusion and poverty require the estimation of the median income or other quantiles of the income distribution. Hence functions that strictly follow the definitions according to Eurostat (2004, 2009) are implemented in package **laeken**. They are used internally for the estimation of the respective indicators, but can also be called by the user directly.

In the analysis of income distributions, the median income is typically of higher interest than the arithmetic mean. This is because income distributions commonly are strongly right-skewed with a heavy tail of *representative outliers* (correctly measured units that are not unique to the population) and *nonrepresentative outliers* (either measurement errors or correct observations that can be considered unique in the population). Therefore, the center of the distribution is more reliably estimated by a weighted median than by a weighted mean, as the latter is highly influenced by extreme values.

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 equivalized disposable income with $x_1 \leq \ldots \leq x_n$, and let $\boldsymbol{w} := (w_i, \ldots, w_n)'$ be the corresponding personal sample weights. Weighted quantiles for the estimation of the population values according to Eurostat (2004, 2009) are then given by

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

This definition of weighted quantiles is available in **laeken** through the function `weightedQuantile()`. The following command computes the weighed 20% quantile, the weighted median, and the weighted 80% quantile. In the context of social exclusion indicators, these are of most importance.

```
R> weightedQuantile(eusilc$eqIncome, eusilc$rb050,
+      probs = c(0.2, 0.5, 0.8))

[1] 12212.60 18098.73 25997.65
```

For the important special case of the weighted median, the function `weightedMedian()` is available for convenience.

```
R> weightedMedian(eusilc$eqIncome, eusilc$rb050)

[1] 18098.73
```

In addition, the functions `incMedian()` and `incQuintile()` are more tailored towards application in the case of indicators on social exclusion and poverty and provide a similar interface as the functions for the indicators (see Section 5). In particular, they allow to supply an additional variable to be used as tie-breakers for sorting, and to compute the weighted median and income quintiles, respectively, for several years of the survey. With the following lines of code, the median income as well as the 1st and 4th income quintile (i.e., the weighted 20% and 80% quantiles) are estimated.

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

[1] 18098.73

R> incQuintile("eqIncome", weights = "rb050", k = c(1, 4), data = eusilc)

       1        4
12212.60 25997.65
```

5 Indicators on social exclusion and poverty

In this section, the most important indicators on social exclusion and poverty are described in detail. Furthermore, the functionality of package **laeken** to estimate these indicators is demonstrated.

It should be noted that all functions for the implemented indicators provide a very similar interface. Most importantly, it is possible to compute estimates for several years of the survey and different subdomains with a single command. Furthermore, the functions allow to supply an additional variable to be used as tie-breakers for sorting. However, not all of the implemented functionality is shown in this vignette. For a complete description of the functions and their arguments, the reader is referred to the corresponding R help pages.

In addition, only point estimation of the indicators on social exclusion and poverty is illustrated here, statistical significance of these estimates is not discussed. The functionality for variance estimation of the indicators is described in the package vignette `laeken-variance` (Templ and Alfons 2011).

For the following definitions of the estimators according to Eurostat (2004, 2009), 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 personal sample weights, where n denotes the number of observations. Furthermore, define the following index sets for a certain threshold t:

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

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

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

5.1 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 (Eurostat 2004, 2009). 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, the sample weights need to be taken into account. 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 functions `arpt()` and `arpr()` are implemented for the estimation of the at-risk-of-poverty threshold and the at-risk-of-poverty rate. Whenever sample weights are available in the data, they should be supplied as the `weights` argument. Even though `arpt()` is called internally by `arpr()`, it can also be called by the user directly.

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

[1] 10859.24

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

```
Value:
[1] 14.44422

Threshold:
[1] 10859.24
```

It is also possible to use these functions for the estimation of the indicator *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. The proportion of the median equivalized income to be used can thereby 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
```

In order to compute estimates for different subdomains, a breakdown variable simply needs to be supplied as the `breakdown` argument. Note that in this case the same overall at-risk-of-poverty threshold is used for all subdomains (see Eurostat 2004, 2009). The following command computes the overall estimate, as well as estimates for all NUTS2 regions.

```
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
```

However, any kind of breakdown can be supplied, e.g., the breakdowns defined by Eurostat (2004, 2009). 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 for 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 results are even more heterogeneous than for the breakdown into NUTS2 regions.

5.2 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 (Eurostat 2004, 2009).

For the estimation of the quintile share ratio from a 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

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

With package **laeken**, the quintile share ratio can be estimated using the function `qsr()`. As for the at-risk-of-poverty rate, sample weights can be supplied via the `weights` argument.

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

Value:
[1] 3.970004
```

Computing estimates for different subdomains is again possible by specifying the `breakdown` argument. In the following example, estimates for each NUTS2 region are computed in addition to the overall estimate.

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

```
Value:
[1] 3.970004

Value by domain:
        stratum    value
1    Burgenland 5.008486
2     Carinthia 3.562404
3 Lower Austria 3.824539
4      Salzburg 3.768393
5        Styria 3.464305
6         Tyrol 3.586046
7 Upper Austria 3.668289
8        Vienna 4.654743
9    Vorarlberg 4.366511
```

Nevertheless, it should be noted that the quintile share ratio is highly influenced by outliers (see Hulliger and Schoch 2009, Alfons et al. 2010). Since the upper tail of income distributions virtually always contains nonrepresentative outliers, robust estimators of the quintile share ratio should preferably be used. Thus robust semi-parametric methods based on Pareto tail modeling are implemented in package **laeken** as well. Their application is discussed in vignette `laeken-pareto` (Alfons et al. 2011b).

5.3 Relative median at-risk-of-poverty gap (by age and gender)

The *relative median at-risk-of-poverty gap* (RMPG) is defined as 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 (Eurostat 2004, 2009).

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

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

In package **laeken**, the function `rmpg()` is implemented for the estimation of the relative median at-risk-of-poverty gap. If available in the data, sample weights should be supplied as the `weights` argument. Note that the function `arpt()` for the estimation of the at-risk-of-poverty threshold is called internally (cf. function `arpr()` for the at-risk-of-poverty rate in Section 5.1).

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

Value:
[1] 18.9286

Threshold:
[1] 10859.24
```

Estimates for different subdomains can be computed by making use of the `breakdown` argument. With the following command, the overall estimate and estimates for all NUTS2 regions are computed.

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

```
Value:
[1] 18.9286

Value by domain:
        stratum    value
1    Burgenland 12.32438
2     Carinthia 13.12787
3 Lower Austria 17.48023
4      Salzburg 28.89533
5        Styria 15.53486
6         Tyrol 19.58447
7 Upper Austria 19.47177
8        Vienna 23.35608
9    Vorarlberg 26.96706

Threshold:
[1] 10859.24
```

For the relative median at-risk-of-poverty gap, the breakdown by age and gender is of particular interest. In the following example, a breakdown variable with all possible combinations of age categories and gender is defined and added to the data set. Afterwards, estimates for the corresponding domains are computed.

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

Value:
[1] 18.9286

Value by domain:
            stratum    value
1    [-1,16):female 19.05696
2      [-1,16):male 19.05696
3    [16,25):female 32.93985
4      [16,25):male 23.70534
5    [25,50):female 20.78422
6      [25,50):male 18.19213
7    [50,65):female 21.34382
8      [50,65):male 18.92860
9   [65,Inf):female 14.48597
10    [65,Inf):male 15.34966

Threshold:
[1] 10859.24
```

5.4 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 (Eurostat 2004, 2009).

For the estimation of the Gini coefficient from a sample, the sample weights need to be taken into account. In mathematical terms, the Gini coefficient is estimated by

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

The function `gini()` is available in **laeken** to estimate the Gini coefficient. As for the other indicators, sample weights can be specified with the `weights` argument.

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

Value:
[1] 26.48962
```

Using the `breakdown` argument in the following command, estimates for the NUTS2 regions are computed in addition to the overall estimate.

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

Value:
[1] 26.48962

Value by domain:
        stratum    value
1    Burgenland 32.05489
2     Carinthia 25.49448
3 Lower Austria 25.93737
4      Salzburg 25.01652
5        Styria 23.71190
6         Tyrol 25.24881
7 Upper Austria 25.49202
8        Vienna 28.94944
9    Vorarlberg 28.74120
```

Since outliers have a strong influence on the Gini coefficient, robust estimators are preferred to the standard estimation described above (see Alfons et al. 2010). Vignette `laeken-pareto` (Alfons et al. 2011b) describes how to apply the robust semi-parametric methods implemented in package **laeken**.

6 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. In package **laeken**, this is implemented by taking advantage of the object-oriented design of the package. Each of the functions for the indicators described in Section 5 returns an object belonging to a class of the same name as the respective function, e.g., function `arpr()` returns an object of class `"arpr"`. All these classes thereby inherit from the basic class `"indicator"` (see Section 2).

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

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
```

```
R> is.arpr(a)
```

```
[1] TRUE
```

```
R> is.indicator(a)
```

```
[1] TRUE
```

```
R> class(a)
```

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

To extract a subset of results from such an object, a `subset()` method for the class `"indicator"` is implemented in **laeken**. The method `subset.indicator()` is hidden from the user and is called internally by the generic function `subset()` whenever an object of class `"indicator"` is supplied. In the following example, the estimates of the at-risk-of-poverty rate for the regions Lower Austria and Vienna are extracted 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
```

7 Conclusions

This vignette demonstrates the use of package **laeken** for point estimation of the European Union indicators on social exclusion and poverty. Since the description of the indicators in Eurostat (2004, 2009) is weak from a mathematical point of view, a more precise notation is given in this paper. Currently, the most important indicators are implemented in **laeken**. Their estimation is made easy with the package, as it is even possible to compute estimates for several years and different subdomains with a single command.

Concerning the inequality indicators quintile share ratio and Gini coefficient, it is clearly visible from their definitions that the standard estimators are highly influenced by outliers (see also Hulliger and Schoch 2009, Alfons et al. 2010). Therefore, robust semi-parametric methods are implemented in **laeken** as well. These are described in vignette `laeken-pareto` (Alfons et al. 2011b), while variance and confidence interval estimation for the indicators on social exclusion and poverty with package **laeken** is treated in vignette `laeken-variance` (Templ and Alfons 2011).

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

A. Alfons and S. Kraft. ***simPopulation**: Simulation of Synthetic Populations for Surveys Based on Sample Data*, 2012. URL https://CRAN.R-project.org/package=simPopulation. R package version 0.4.0.

A. Alfons and M. Templ. Estimation of social exclusion indicators from complex surveys: The R package **laeken**. *Journal of Statistical Software*, 54(15):1–25, 2013. doi: 10.18637/jss.v054.i15.

A. Alfons, M. Templ, P. Filzmoser, and J. Holzer. A comparison of robust methods for Pareto tail modeling in the case of Laeken indicators. In C. Borgelt, G. González-Rodríguez, W. Trutschnig, M.A. Lubiano, M.A. Gil, P. Grzegorzewski, and O. Hryniewicz, editors, *Combining Soft Computing and Statistical Methods in Data Analysis*, volume 77 of *Advances in Intelligent and Soft Computing*, pages 17–24. Springer-Verlag, Heidelberg, 2010. ISBN 978-3-642-14745-6.

A. Alfons, S. Kraft, M. Templ, and P. Filzmoser. Simulation of close-to-reality population data for household surveys with application to EU-SILC. *Statistical Methods & Applications*, 20(3): 383–407, 2011a.

A. Alfons, M. Templ, P. Filzmoser, and J. Holzer. Robust pareto tail modeling for the estimation of social inclusion indicators using the R package **laeken**. Research Report CS-2011-2, Department of Statistics and Probability Theory, Vienna University of Technology, 2011b.

A. Alfons, J. Holzer, and M. Templ. ***laeken**: Estimation of Indicators on Social Exclusion and Poverty*, 2013. URL https://CRAN.R-project.org/package=laeken. R package version 0.4.5.

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

Eurostat. 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, 2004.

Eurostat. 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, 2009.

B. Hulliger and T. Schoch. Robustification of the quintile share ratio. New Techniques and Technologies for Statistics, Brussels, 2009.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013. URL https://www.R-project.org. ISBN 3-900051-07-0.

M. Templ and A. Alfons. Variance estimation of social inclusion indicators using the R package **laeken**. Research Report CS-2011-3, Department of Statistics and Probability Theory, Vienna University of Technology, 2011.