The R Package emdi for Estimating and Mapping Regionally Disaggregated Indicators

Ann-Kristin Kreutzmann
Freie Universität Berlin

Sören Pannier
Freie Universität Berlin

Natalia Rojas-Perilla
Freie Universität Berlin

Timo Schmid
Freie Universität Berlin

Matthias Templ
Zurich University of Applied Sciences

Nikos Tzavidis
University of Southampton

Abstract

The R package **emdi** enables the estimation of regionally disaggregated indicators using small area estimation methods and includes tools for processing, assessing, and presenting the results. The mean of the target variable, the quantiles of its distribution, the Head Count Ratio, the Poverty Gap, the Gini coefficient, the Quintile Share Ratio, and customized indicators are estimated using direct and model-based estimation with the Empirical Best Predictor (EBP) (Molina and Rao 2010). The user is assisted by automatic estimation of data-driven transformation parameters. Parametric and semi-parametric, wild bootstrap for mean squared error estimation are implemented with the latter offering protection against possible misspecification of the error distribution. Tools for (a) customized parallel computing, (b) model diagnostic analyses, (c) creating high quality maps and (d) exporting the results to Excel™ and OpenDocument Spreadsheets are included. The functionality of the package is illustrated with example data sets for estimating the Gini coefficient and median income for districts in Austria.

Keywords: official statistics, survey statistics, parallel computing, small area estimation, visualization.

1. Introduction

In recent years an increased number of policy decisions has been based on statistical information derived from indicators estimated at disaggregated geographical levels using small area estimation methods. Clearly, the more detailed the information provided by official statistics estimates, the better the basis for targeted policies and evaluating intervention programs. The United Nations suggest further disaggregation of statistical indicators for monitoring the Sustainable Development Goals (SDGs). National Statistical Institutes (NSIs) and other organizations across the world have also recognized the potential of producing small area statistics and their use for informing policy decisions. Examples of NSIs with well-developed programs in the production of small area statistics include the US Bureau of Census, the UK Office for National Statistics (ONS) and the Statistical Office of Italy (ISTAT). Although the term domain is more general as it may include non-geographic dimensions, the term small area estimation (SAE) is the established one. We shall follow the custom in this paper and use the terms area/geography and domain/aggregation interchangeably.

Without loss of generality in this paper we will assume that the primary data sources used to estimate statistical indicators are national socio-economic household sample surveys. Sample surveys are designed to provide estimates with acceptable precision at national and possibly sub-national levels but usually have insufficient sizes to allow for precise estimation at lower geographical levels. Therefore, direct estimation that relies only on the use of survey data can be unreliable or even not possible for domains that are not represented in the sample. In the absence of financial resources for boosting the sample size of surveys, using model-based methodologies can help to obtain reliable estimates for the target domains.

Model-based SAE methods (Pfeffermann 2013; Rao and Molina 2015) work by using statistical models to link survey data, that are only available for a part of the target population, with administrative or census data that are available for the entire population. Despite the wide range of SAE methods that have been proposed by academic researchers, these are so far applied only by a fairly small number of NSIs or other practitioners. This gap between theoretical advances and applications may have several reasons one of which is the lack of suitable, user friendly statistical software. More precisely, software needs not only to be available but it also needs to simplify the application of the methods for the user. The R (R Core Team 2018) package **emdi** (Kreutzmann *et al.* 2018) aims to improve the user experience by simplifying the estimation of small area indicators and corresponding precision estimates. Furthermore, the user benefits from support that extends beyond estimation in particular, evaluating, processing, and presenting the results.

Traditionally model-based SAE methods have been employed for estimating simple, linear indicators for example, means and totals using for example, mixed (random) effects models and empirical best linear unbiased predictors (EBLUPs). Several software packages exist. In R, the package **JoSAE** (Breidenbach 2015) includes functions for EBLUPs using unit-level models. Functions in the package **hbsae** (Boonstra 2012) enable the use of unit- and area-level models and can be estimated either by using restricted maximum likelihood (REML) or hierarchical Bayes methods. The package **BayesSAE** (Shi and with contributions from Peng Zhang 2013) also allows for Bayesian methods. The **rsae** package by Schoch (2012) and package **saeRobust** by Warnholz (2016) provide functions for outlier robust small area estimation using unit- or area-level models. Gaussian area-level multinomial mixed-effects models for SAE can be done with the **mme** package (Lopez-Vizcaino *et al.* 2014). In addition, resources in R are available for Bayesian SAE from the BIAS (Bayesian methods for combining multiple Individual and Aggregate data Sources) project (Gómez-Rubio *et al.* 2010) and from the package **SAE2** (Gómez-Rubio *et al.* 2008) that provides likelihood-based methods. In Stata, functions `xtmixed` and `gllamm` support the estimation of linear mixed models, which is a popular basis for model-based SAE. EBLUPs can be derived using these functions (West *et al.* 2007). Similarly, `PROC MIXED` and `PROC IML` can be used for fitting unit- and area-level models in SAS as shown in Mukhopadhyay and McDowell (2011). Furthermore, several SAS macros for SAE are provided by the EURAREA (Enhancing Small Area Estimation Techniques to meet European Needs) project (EURAREA Consortium 2004).

More recently widespread application of SAE methods involves the estimation of poverty and inequality indicators and distribution functions (The World Bank 2007). In this case the use of methodologies for estimating means and totals is no longer appropriate since such indicators are complex, non-linear functions of the data. As an example, we refer to the Foster-Greer-

Thorbecke indicators (Foster *et al.* 1984), the Gini coefficient (Gini 1912) and the quantiles of the income distribution. Popular SAE approaches for estimating complex indicators include the Empirical Best Predictor (EBP) (Molina and Rao 2010), the World Bank method (Elbers *et al.* 2003) and the M-Quantile method (Chambers and Tzavidis 2006; Tzavidis *et al.* 2010). Although in this paper we focus exclusively on software for implementing the EBP method (Molina and Rao 2010), a future version of the package will include the M-Quantile and World Bank methods. The World Bank provides a free software for using the World Bank method called PovMap (The World Bank Group 2013). However, this focuses exclusively on poverty mapping. Creating a more general open-source software can help to accelerate the uptake of modern model-based methods. Currently, the best known package that also includes the EBP method is the R package **sae** (Molina and Marhuenda 2015). Although the **sae** package implements a range of small area methods, it lacks the necessary functionality for supporting the user beyond estimation for example, for performing model diagnostic analyses, visualising, and exporting the results for further processing. In contrast, **emdi** supports the user by providing more options and greater flexibility. In particular, package **emdi** offers the following attractive features that distinguish it from the **sae** package and other R packages for SAE:

- The estimation functions return by default estimates for a set of predefined indicators, including the mean, the quantiles of the distribution of the response variable and poverty and inequality indicators. Additionally, self-defined indicators or indicators available from other packages can be included.

- The user can select the type of data transformation to be used in **emdi**. Data-driven transformation parameters are estimated automatically.

- In contrast to other packages that include only a parametric bootstrap for mean squared error (MSE) estimation, package **emdi** includes two bootstrap methods, a parametric bootstrap and a semi-parametric wild bootstrap (see Appendix A) for MSE estimation. Both incorporate the uncertainty due to the estimation of the transformation parameter. The use of wild bootstrap (Thai *et al.* 2013; Flachaire 2005) protects the user against departures from the distributional assumptions of the nested error linear regression model. This offers additional protection against possible misspecification of the model assumptions.

- Customized parallel computing is offered for reducing the computational time associated with the use of bootstrap.

- Package **emdi** provides predefined functions for diagnostic analyses of the model assumptions. A mapping tool for plotting the estimated indicators enables the creation of high quality visualization. The output summarizing the most relevant results can be exported to Excel™ and to OpenDocument Spreadsheets for presentation and reporting purposes.

The remainder of this paper is structured as follows. Section 2 gives information about the estimation methods that are included in the package. In Section 3 we present the data sets that we used for illustrating the use of the **emdi** package. Section 4 describes the core functionality of the package. Examples demonstrate the use of the methods for computing, assessing and presenting the estimates. Section 5 shows how users can extend the set of indicators to be estimated by including customized options and describes the parallelization features of the package. Finally, Section 6 discusses future potential extensions.

2. Statistical methodology

In order to obtain regionally disaggregated indicators, package **emdi** includes direct estimation and currently model-based estimation using the EBP approach by Molina and Rao (2010). The predefined indicators returned by the estimation functions in **emdi** include the mean and quantiles Q_q (10%, 25%, 50%, 75%, 90%) of the target variable as well as non-linear indicators of the target variable. A widely used family of indicators measuring income deprivation and inequality is the Foster-Greer-Thorbecke (FGT) one (Foster *et al.* 1984). Package **emdi** includes the FGT measures of Head Count Ratio (HCR) and Poverty Gap (PG). In order to compute the HCR and PG indicators one must use a threshold z, also known as poverty line. This line is a minimum level of income deemed adequate for living in a particular country and can be defined in terms of absolute or relative poverty. For instance, the international absolute poverty line is currently set to \$1.90 per day by the World Bank (The World Bank 2017). Relative poverty means a low income relative to others in a particular country - for instance, below a percentage of the median income in that country. Another family of indicators of interest is the so-called Laeken indicators, endorsed by the European Council in Laeken, Belgium (Council of the European Union 2001). Two examples of Laeken indicators that are well-known for measuring inequality are the Gini coefficient (Gini 1912) and the Income Quintile Share Ratio (QSR) (Eurostat 2004). These two inequality indicators are also available in **emdi**. Therefore, in total **emdi** includes ten predefined indicators I_i - summarized in Table 1 - that are estimated at domain level i using a) direct estimation introduced in Section 2.1 and b) model-based estimation via the EBP method introduced in Section 2.2. In the following sections the notation denotes by U a finite population of size N, partitioned into D domains $U_1, U_2, \ldots, U_D$ of sizes $N_1, \ldots, N_D$, where $i = 1, \ldots, D$ refers to an ith domain and $j = 1, \ldots, N_i$ to the jth household/individual. From this population a random sample of size n is drawn. This leads to $n_1, \ldots, n_D$ observations in each domain. If n_i is equal to 0 the domain is not in the sample. The target variable is denoted by $\mathbf{y}$.

2.1. Direct estimation

Direct estimation relies on the use of sample data only. The definition of direct (point and variance) estimators in **emdi** follows Alfons and Templ (2013). The mean and the quantiles help to describe the level and the distribution of a target variable. Especially for target variables with a skewed distribution, quantiles can be more appropriate summary statistics than the mean, since these are robust to extreme values. Direct estimators of the mean and the quantiles are defined as follows,

$$\widehat{\mathrm{Mean}}_i = \frac{\sum_{j=1}^{n_i} w_{ij} y_{ij}}{\sum_{j=1}^{n_i} w_{ij}},$$

$$\widehat{\mathrm{Q}}_{i,q} = \begin{cases} \frac{1}{2}\left(y_{ik} + y_{ik+1}\right) & \text{if} \sum_{j=1}^{k} w_{ij} = q \sum_{j=1}^{n_i} w_{ij}; \\ y_{ik+1} & \text{if} \sum_{j=1}^{k} w_{ij} \leq q \sum_{j=1}^{n_i} w_{ij} \leq \sum_{j=1}^{k+1} w_{ij}, \end{cases}$$

where w denotes the sample weights and $q \in (0,1)$ defines the corresponding quantile.

Measurement	**Indicator I_i**	**Expression**	**Range**
Location	Mean_i	$\frac{\sum_{j=1}^{N_i} y_{ij}}{N_i}$	$\mathbb{R}$
	$\text{Q}_{i,q}$	$F_i^{-1}(q) = \inf\{y_i \in \mathbb{R} : F_i(y_i) \geq q\}$	$\mathbb{R}$
Poverty	HCR_i	$\frac{1}{N_i}\sum_{j=1}^{N_i} \mathbf{I}(y_{ij} \leq z)$	$[0,1]$
	PG_i	$\frac{1}{N_i}\sum_{j=1}^{N_i} \left(\frac{z-y_{ij}}{z}\right)\mathbf{I}(y_{ij} \leq z)$	$[0,1]$
Inequality	Gini_i	$\frac{2\sum_{j=1}^{N_i} j y_{ij}}{N_i \sum_{j=1}^{N_i} y_{ij}} - \frac{(N_i+1)}{N_i}$	$[0,1]$
	QSR_i	$\frac{\sum_{j=1}^{N_i} \mathbf{I}(y_{ij} > Q_{i,0.8}) y_{ij}}{\sum_{j=1}^{N_i} \mathbf{I}(y_{ij} \leq Q_{i,0.2}) y_{ij}}$	$\mathbb{R}$

Table 1: List of predefined population indicators in **emdi**. Note that $F_i(y_i)$ denotes the empirical distribution function of the population in domain i and quantiles are generally defined for $q \in (0,1)$. The predefined quantiles in **emdi** are $q \in (0.1, 0.25, 0.5, 0.75, 0.9)$.

FGT measures HCR and PG are estimated by package **emdi** as follows,

$$\widehat{\text{HCR}}_i = \frac{1}{\sum_{j=1}^{n_i} w_{ij}} \sum_{j=1}^{n_i} w_{ij}\mathbf{I}(y_{ij} \leq z),$$

$$\widehat{\text{PG}}_i = \frac{1}{\sum_{j=1}^{n_i} w_{ij}} \sum_{j=1}^{n_i} w_{ij}\left(\frac{z - y_{ij}}{z}\right)\mathbf{I}(y_{ij} \leq z),$$

where the indicator function $\mathbf{I}(\cdot)$ equals 1 if the target variable y_{ij} is below the threshold z and 0 otherwise. As already mentioned, for the computation of the HCR and PG indicators one must use a threshold z, also known as the poverty line. Package **laeken** (Alfons and Templ 2013) uses relative poverty lines defined as 60% of median equivalized disposable income, which corresponds to the EU definition for poverty lines and thus in this case the HCR is the At-risk-of-poverty rate. In contrast, package **emdi** allows both for absolute and relative poverty lines and the user is free to set the poverty line. Therefore, the threshold can be given as an argument in **emdi** or, alternatively, the user can define an arbitrary function depending on the target variable and sampling weights. As a default, a relative threshold defined as 60% of the median of the target variable is used. The HCR describes the proportion of the population below the poverty line and the PG further takes into account how far, on average, this proportion falls below the threshold. Both indicators are between 0 and 1.
The two inequality indicators Gini and QSR are estimated in **emdi** by

$$\widehat{\text{Gini}}_i = \left[\frac{2\sum_{j=1}^{n_i}\left(w_{ij}y_{ij}\sum_{k=1}^{j} w_{ik}\right) - \sum_{j=1}^{n_i} w_{ij}^2 y_{ij}}{\sum_{j=1}^{n_i} w_{ij}\sum_{j=1}^{n_i} w_{ij}y_{ij}} - 1\right],$$

$$\widehat{\text{QSR}}_i = \frac{\sum_{j=1}^{n_i} \mathbf{I}(y_{ij} > Q_{i,0.8}) w_{ij} y_{ij}}{\sum_{j=1}^{n_i} \mathbf{I}(y_{ij} \leq Q_{i,0.2}) w_{ij} y_{ij}},$$

where $\mathbf{I}(\cdot)$ is an indicator function that equals 1 if the target variable is above the weighted 80% quantile or below the 20% quantile and 0 otherwise. The Gini coefficient is between 0 and 1, and the higher the value, the higher the inequality is. The extreme values of 0 and 1 indicate perfect equality and inequality, respectively. QSR is typically used when the target variable is income and in this case it is defined as the ratio of total income of the 20% richest households to the 20% poorest households. The higher the value of QSR, the higher the inequality is.

While variance estimation in package **laeken** (Alfons and Templ 2013) is only available for the poverty and inequality indicators, package **emdi** also provides a non-parametric bootstrap method (Alfons and Templ 2013) for estimating the variance of estimates of the mean and the quantiles. The variance is, on the one hand, an important measure for measuring the precision of estimates. On the other hand, it is also important to compute the coefficient of variation (CV) which is one measure for showing the extent of the variability of the estimate. The CV is used, for instance, by NSIs for quantifying the uncertainty associated with the estimates and is defined as follows,

$$CV = \frac{\sqrt{\widehat{MSE}(\hat{I}_i)}}{\hat{I}_i},$$

where $\hat{I}_i$ is an estimate of an indicator I_i for domain i and $\widehat{MSE}(\hat{I}_i)$ is the corresponding mean squared error.

2.2. Model-based estimation

The implementation of the EBP method in package **emdi** is based on the theory described in Molina and Rao (2010) and Rao and Molina (2015). The underlying model is a unit-level mixed model also known in SAE literature as the nested error linear regression model (Battese *et al.* 1988). In its current implementation the EBP method is based on a two-level nested error linear regression model that includes a random area/domain-specific effect and a unit (household or individual)-level error term.

In addition to the notation above, here we assume that $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_p)^\top$ is the design matrix, containing p explanatory variables. The nested error linear regression model is defined by

$$T(y_{ij}) = \mathbf{x}_{ij}^\top \boldsymbol{\beta} + u_i + e_{ij}, \quad j = 1, \dots, n_i, \quad i = 1, \dots, D, \quad u_i \overset{iid}{\sim} N(0, \sigma_u^2), \quad e_{ij} \overset{iid}{\sim} N(0, \sigma_e^2), \quad (1)$$

where T denotes a transformation of the target variable $\mathbf{y}$, $\mathbf{x}_{ij}$ is a vector of unit-level auxiliary variables of dimension $(p+1) \times 1$, $\boldsymbol{\beta}$ is the $(p+1) \times 1$ vector of regression coefficients and u_i and e_{ij} denote the random area and unit-level error terms. The EBP approach works by using at least two data sources, namely a sample data set used to fit the nested error linear regression model and a population (e.g., census or administrative) data set used for predicting - under the model - synthetic values of the outcome for the entire population. Both data sources must share identically defined covariates but the target variable is only available in the sample data set.

Use of data transformations

Under model (1) we assume that the model error terms follow a Gaussian distribution. However, in certain applications - as is the case when analyzing economic variables - this assumption may be unrealistic. Package **emdi** includes the option of using a one-to-one transformation $T(y_{ij})$ of the target variable $\mathbf{y}$ aiming to make the Gaussian assumptions more plausible. A logarithmic-type transformation is very often used in practice (Elbers *et al.* 2003; Molina and Rao 2010). However, this is not necessarily the optimal transformation, for example, when the model error terms do not follow exactly a log-normal distribution. In addition to a logarithmic transformation, package **emdi** allows the use of a data-driven Box-Cox transformation. The Box-Cox transformation is denoted by

$$T(y_{ij}) = \begin{cases} \frac{(y_{ij}+s)^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0; \\ \log(y_{ij}+s) & \text{if } \lambda = 0, \end{cases} \tag{2}$$

where λ is an unknown transformation parameter and s denotes the shift parameter, which is a constant and chosen automatically such that $y_{ij} + s > 0$. A general algorithm for estimating the transformation parameter λ is the residual maximum likelihood (REML), which is described in detail in Rojas-Perilla *et al.* (2017). One advantage of using the Box-Cox transformation is that it includes the logarithmic and no transformation as cases for specific values of λ. Package **emdi** currently includes the following options: no transformation, logarithmic transformation and Box-Cox transformation.
The EBP method is implemented using the following algorithm:

1. For a given transformation obtain $T(y_{ij})$. If the user selects the Box-Cox transformation, the transformation parameter λ is automatically estimated by the **emdi** package.

2. Use the sample data to fit the nested error linear regression model and estimate $\boldsymbol{\theta}$ denoted by $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\beta}}, \hat{\sigma}_u^2, \hat{\sigma}_e^2)$. The variance components are estimated by REML using the function `lme` from the package **nlme** (Pinheiro *et al.* 2018). Also compute $\hat{\gamma}_i = \frac{\hat{\sigma}_u^2}{\hat{\sigma}_u^2 + \frac{\hat{\sigma}_e^2}{n_i}}$.

3. For $l = 1, \ldots, L$:

   (a) For in-sample domains (domains that are part of the sample data set), generate a synthetic population of the target variable by $T(y_{ij}^{*(l)}) = \mathbf{x}_{ij}^{\top}\hat{\boldsymbol{\beta}} + \hat{u}_i + v_i^* + e_{ij}^*$, with $v_i^* \overset{iid}{\sim} N(0, \hat{\sigma}_u^2(1-\hat{\gamma}_i))$, $e_{ij}^* \overset{iid}{\sim} N(0, \hat{\sigma}_e^2)$ and $\hat{u}_i$, the conditional expectation of u_i given y_i.
   For out-of-sample domains (domains with no data in the sample) the conditional expectation of u_i cannot be computed, hence for these domains generate a synthetic population by using $T(y_{ij}^{*(l)}) = \mathbf{x}_{ij}^{\top}\hat{\boldsymbol{\beta}} + v_i^* + e_{ij}^*$, with $v_i^* \overset{iid}{\sim} N(0, \hat{\sigma}_u^2)$, $e_{ij}^* \overset{iid}{\sim} N(0, \hat{\sigma}_e^2)$. For additional details we refer to Molina and Rao (2010).

   (b) Back-transform to the original scale $\mathbf{y}_i^{(l)} = T^{-1}(\mathbf{y}_i^{*(l)})$ and calculate the target indicator $I_i^{(l)}(\mathbf{y}_i^{(l)})$ in each domain. Note that $I_i^{(l)}$ is used here as a generic notation for any indicator of interest.

4. Compute the final estimates by taking the mean over the L Monte Carlo simulations in each domain, $\hat{I}_i^{EBP} = 1/L \sum_{l=1}^{L} I_i^{(l)}(\mathbf{y}_i^{(l)})$.

The **emdi** package fits the nested error linear regression model by using the **nlme** package and currently does not permit the use of an alternative package for example **lme4** (Bates *et al.* 2015). The reason for this choice is that in future developments of **emdi** we plan to allow for more complex covariance structures for the unit-level error term and the random effect for example, allowing for spatially correlated errors (Pratesi and Salvati 2009; Schmid *et al.* 2016). To the best of our knowledge, the **nlme** package offers sufficient flexibility for incorporating such models.

Measuring the uncertainty of the EBP estimates is done by using bootstrap methods. Here the uncertainty is quantified by the MSE. Package **emdi** includes two bootstrap schemes. One is parametric bootstrap under model (1) following Molina and Rao (2010), which additionally includes the uncertainty due to the estimation of the transformation parameter (Rojas-Perilla *et al.* 2017). Using an appropriate transformation often mitigates the departures from normality. However, even after transformations, departures from normality may still exist in particular for the unit-level error term. For this reason, **emdi** also includes a variation of semi-parametric wild bootstrap (Flachaire 2005; Thai *et al.* 2013; Rojas-Perilla *et al.* 2017) to protect against departures from the model assumptions. The semi-parametric wild bootstrap is presented in detail in Appendix A. A simulation study comparing the performance of both MSE estimators is presented in Rojas-Perilla *et al.* (2017). Since the bootstrap schemes presented here are computationally intensive, **emdi** includes an option for parallelization that is described in detail in Section 5.2.

3. Data sets

The main idea of SAE is to combine multiple data sources. Typically, one data set is obtained from a survey on unit-level and the other one from census or administrative/register data. The target variable is available in the survey but not in the census data. The administrative data contains explanatory variables that are potentially correlated with the target variable and hence they can be used to assist the estimation. Depending on the model type and the indicator of interest, census information is needed at the unit-level, i.e., information is available for every unit in each domain, or it is required at the area-level which means that aggregated data for each domain is given. If the user is interest in estimating non-linear functions of the target variable (like indicators discussed in Section 2), then access to unit-level census data is needed. As the EBP approach in package **emdi** is suitable for estimating non-linear indicators, one population data set (`eusilcA_pop`) and one survey data set (`eusilcA_smp`) are provided at the household level such that the method can be illustrated. The two data sets are based on the use of `eusilcP` from the package **simFrame** (Alfons *et al.* 2010). This data set is a simulated close-to-reality version of the European Union Statistics on Income and Living Conditions (EU-SILC) in Austria from 2006. Austria is a federal republic in Central Europe made up of nine states and 94 districts (79 districts headed by commissions and 15 statutory cities) with a total population of about 8.8 million in 2018. The original EU-SILC data is obtained from an annual household survey that is nowadays conducted in all EU member states and six other European countries and enables the analysis of income, socio-demographic factors and living conditions.

For practical reasons, we need to modify the `eusilcP` data set used in package **simFrame**. Due to confidentiality constraints the lowest geographical level in this data set includes the nine states and identifiers for lower regional levels, like the 94 districts, are not included. However, in the context of SAE the interest is on lower geographical levels like districts or municipalities. Therefore, we assigned households to Austrian districts for illustrating the methodology better. The modified synthetic population is called `eusilcA_pop`. The assignment is based on two criteria available from external sources: a) the population sizes at state and district level and b) the income level in each district. From the last register-based census in 2011 the population sizes in each district and in each state are known and publicly available (Statistik Austria 2013). We defined the district population sizes in relation to the state population sizes in the `eusilcA_pop` data set such that their population ratios mimic the *true* ratios in Austria. Furthermore, the Austrian Chamber of Commerce published a ranking of the districts within the states based on the net per capita income (Wirtschaftskammer Österreich 2017). Based on this ranking we assigned households to districts such that the ordering of the districts within states is maintained. One drawback of the population data set is the small number of households in some districts. For instance, the number of households is only 5 in Rust (Stadt). This is, however, partly due to the fact that it is also in reality a really small district with only 1896 inhabitants (Statistik Austria 2013). Although the `eusilcA_pop` data set in **emdi** mimics some real characteristics in Austria, it is a synthetic population data set for demonstrating the functionality of the package and conclusions about the levels of inequality and poverty in the Austrian districts observed from this data are not official estimates. The full documented code for the assignment of the households to the districts is available from the authors' `GitHub` folder (`https://github.com/SoerenPannier/districtAssignment.git`).

The target variable in the example is the equivalized household income (`eqIncome`), which is defined as the total household disposable income divided by the equivalized household size determined by the modified OECD scale (Hagenaars *et al.* 1994). Thus, the indicators in our illustration describe the distribution of income, poverty and inequality similarly to the analysis in Alfons and Templ (2013). The remaining variables in `eusilcA_pop` are variables that identify the regional levels (`state` and `district`) and auxiliary variables that can be used for modeling income. These explanatory variables are, among others, gender (`gender`), the equivalized household size (`eqsize`) as well as financial resources like the employees cash (`cash`) or unemployment benefits (`unempl_ben`). Table 2 gives an overview of possible model covariates.

The sample data set `eusilcA_smp` is a household sample from the `eusilcA_pop` population that includes 1945 observations. The sample is drawn by stratified random sampling where the districts define the strata. For the 75% largest districts (in terms of number of households) 10% of the households were selected and the maximum number of sampled households is equal to 200 in any given district. Consequently, the 25% smallest districts do not have any observation in the sample. Summaries of state and district-specific sample sizes are given below.

```
R> data("eusilcA_smp")
R> table(eusilcA_smp$state)

Burgenland     Carinthia Lower Austria      Salzburg        Styria
        31           162           387           163           337
Tyrol Upper Austria        Vienna    Vorarlberg
  173           392           200           100

R> summary(as.numeric(table(eusilcA_smp$district)))

 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
14.00   17.00   22.50   27.79   29.00  200.00
```

Variable	Meaning	Scale level
Target variable		
`eqIncome`	The equivalized household income.	Numeric
Domain identifiers		
`state`	Austrian states.	Factor
`district`	Austrian districts.	Factor
Explanatory variables		
`eqsize`	The equivalized household size according to the modified OECD scale.	Numeric
`gender`	The person's gender (levels: `female` and `male`).	Factor
`cash`	Employee cash or near cash income.	Numeric
`self_empl`	Cash benefits or losses from self-employment (net).	Numeric
`unempl_ben`	Unemployment benefits (net).	Numeric
`age_ben`	Old-age benefits (net).	Numeric
`surv_ben`	Survivor's benefits (net).	Numeric
`sick_ben`	Sickness benefits (net).	Numeric
`dis_ben`	Disability benefits (net).	Numeric
`rent`	Income from rental of a property or land (net).	Numeric
`fam_allow`	Family/children related allowances (net).	Numeric
`house_allow`	Housing allowances (net).	Numeric
`cap_inv`	Interest, dividends, profit from capital investments in unincorporated business (net).	Numeric
`tax_adj`	Repayments/receipts for tax adjustment (net).	Numeric
Design variable		
weight	Sampling weight.	Numeric

Table 2: Variables of the two data sets in package **emdi**. Note that the population data set does not contain a variable for the sampling weights.

District-specific sample sizes (in contrast to state-specific) are quite small with 25% of districts having no sample data at all. Hence, the use of SAE methods may be useful in this case. In Section 4 we discuss the estimation of regional indicators based on these data sets in detail. In addition to SAE methods, package **emdi** provides a function called `map_plot` that produces maps of the estimated indicators. In order to demonstrate the use of the function `map_plot` package **emdi** contains a shape file for the 94 Austrian districts which is downloaded from the SynerGIS website (Bundesamt für Eich- und Vermessungswesen 2017). This shape file is saved in `.rda` format and the object `shape_austria_dis` is a `SpatialPolygonsDataFrame`. For more information about this class we refer to Bivand *et al.* (2013).

4. Basic design and core functionality

Section 2 presented the statistical methodology that uses either direct estimation or the model-based EBP approach. In package **emdi** direct and model-based estimation are implemented with functions `direct` and `ebp`, respectively. A key benefit offered by **emdi** is the flexibility for producing, assessing, presenting and exploring the estimates. This is achieved by using the following commands:

1. Estimate domain indicators including MSE estimation: `direct` and `ebp`
2. Get summary statistics and model diagnostics: `summary` and `plot`
3. Extract and compare the indicators of interest: `estimators` and `compare`
4. Visualize the estimated indicators: `map_plot`
5. Export the results to Excel™: `write.excel`

The package **emdi** uses the S3 object system (Chambers and Hastie 1992). All objects created in the package **emdi** by an estimation function (`direct` and `ebp`) share the class `emdi`. Objects of class `emdi` comprise ten components, which are presented in Table 3. Some of these components are specific only to one of the estimation methods, such that they are `NULL` for the other one. These components are indicated in the second column of Table 3. Depending on the estimation method, the `emdi` object is also of class `direct` or `model`. Thus, the commands can be tailored to the estimation method, e.g., model diagnostics (provided by the command `plot`) are only suitable when a model-based approach is used. In what follows the estimation functions are presented and **emdi** functionalities are illustrated.

4.1. Estimation of domain indicators

As far as possible, the two estimation functions (`direct` and `ebp`) have the same structure and variable names, which helps to simplify their use. For function `direct`, the user has to specify three arguments (see Table 4), that include the target variable, the sample data set, and the variable name that defines the domain identifier in the sample data. For the remaining arguments suitable defaults are defined. The EBP approach is implemented in **emdi**, using function `ebp`. As shown in Table 5, the user has to specify five arguments that include the structure of the fixed effects of the nested error linear regression model, the two data sets (population and sample), and the variable names that define the domain identifiers in each data set. For the remaining arguments suitable defaults are defined. Following Molina and Rao (2010), the number of Monte Carlo iterations L and the number of bootstrap populations B are set to 50 by default. In practice, we recommend using larger values for example, $L \geq 200$ and $B \geq 200$. The choice of a transformation is simplified since the user only has to choose the type of transformation. The shift parameter s and the optimal transformation parameter λ in the case of using the Box-Cox transformation are automatically estimated. This distinguishes **emdi** from package **sae** (Molina and Marhuenda 2015) where the user has to select the transformation parameters manually. Since the Box-Cox transformation includes the no transformation and logarithmic transformation as special cases, this family of transformations is chosen as the default option.

Example using Austrian districts:
For illustrating the functions of package **emdi** we estimate indicators using the data sets described in Section 3. The target variable is the equivalized income (`eqIncome`) and the regional level of interest are Austrian districts included in variable `district`. For direct estimation of the indicators the user has to specify these two arguments and the sample data set called `eusilcA_smp`. In addition, several other arguments are defined as shown below. We account

Position	**Name**	**Short description**	**Available** for `direct`	`model`
1	`ind`	Point estimates for indicators per domain	✓	✓
2	`MSE`	Variance/MSE estimates per domain	✓	✓
3	`transform_param`	Transformation and shift parameters		✓
4	`model`	Fitted linear mixed-effects model as `lme` object		✓
5	`framework`	List with 8 components describing the data	✓	✓
6	`transformation`	Type of transformation		✓
7	`method`	Estimation method for transformation parameter		✓
8	`fixed`	Formula of fixed effects used in the nested error linear regression model		✓
9	`call`	Image of the function call that produced the object	✓	✓
10	`successful_bootstraps`	A matrix with domains as rows, indicators as columns and the number of corresponding successful bootstraps	✓	

Table 3: Components of `emdi` objects. All explanations can be found in the documentation of the `emdi` object in the package.

for the sampling design by including the sampling weights in the estimation. Furthermore, we set the threshold argument to 60% of the median of equivalized income that - in this example - equals 10885.33 and we are also interested in obtaining the variance estimates of the indicators.

```
R> emdi_direct <- direct(y = "eqIncome", smp_data = eusilcA_smp,
+    smp_domains = "district", weights = "weight", threshold = 10885.33,
+    var = TRUE)
```

The R object `emdi_direct` is of classes `emdi` and `direct`.
An example of using the `ebp` method for computing point and MSE estimates for the predefined indicators and two custom indicators, namely the minimum and maximum equivalized income is provided below:

```
R> emdi_model <- ebp(fixed = eqIncome ~ gender + eqsize + cash + self_empl +
+    unempl_ben + age_ben + surv_ben + sick_ben + dis_ben + rent +
+    fam_allow + house_allow + cap_inv + tax_adj, pop_data = eusilcA_pop,
+    pop_domains = "district", smp_data = eusilcA_smp,
+    smp_domains = "district", threshold = 10885.33, MSE = TRUE,
+    custom_indicator = list(my_max = function(y, threshold){max(y)},
+                            my_min = function(y, threshold){min(y)}))
```

In contrast to the direct estimation, the user also has to choose the auxiliary variables to be included in the nested error linear regression model. The variables that are chosen to explain the equivalized income are demographics as gender and the equivalized household size but also financial benefits and allowances as for example cash income, unemployment benefits and capital investement. Furthermore, model-based estimation requires the use of both, population (`eusilcA_pop`) and sample (`eusilcA_smp`) data and the domain identifiers. For enabling the comparison between direct and model-based estimates of the indicators of interest we use the same threshold as in the direct estimation. MSE estimates are returned by setting the MSE argument to `TRUE`. The final R object `emdi_model` is of classes `emdi` and `model`. For this object we show in the following subsections the **emdi** functionalities.

4.2. Summary statistics and model diagnostics

R-users typically use a `summary` method for summarizing the results. For `emdi` objects the summary outputs differ depending on the two classes. The summary for objects obtained by direct estimation gives information about the number of domains in the sample, the total and domain-specific sample sizes. The summary for model-based objects is more extensive. In addition to information about the sample sizes, information about the population size and the number of out-of-sample domains is provided. Since model-based SAE relies on prediction under the model, including model diagnostics in **emdi** is important for users. A first measure to consider when evaluating the working model is the well known R^2. Nakagawa and Schielzeth (2013) provide a generalization of this measure for linear mixed models. A marginal R^2 and a conditional (a measure that accounts for the random effect) R^2 are implemented via function `r.squaredGLMM` in package **MuMIn** (Barton 2018). The `summary` method uses this function to calculate and present both measures. For the EBP and model-based SAE methods in general the validity of parametric assumptions is crucial. Therefore, **emdi** also

Arguments	**Short description**	**Default**
`y`	Target variable	
`smp_data`	Survey data	
`smp_domains`	Domain identifier	
`weights`	Sampling weights	No weights
`design`	Variable indicating strata	No design
`threshold`	Threshold for poverty indicators	60% of the median of the target variable
`var`	Variance estimation	No variance estimation
`boot_type`	Type of bootstrap: naive or calibrate	Naive
`B`	Number of bootstrap populations	50
`seed`	Seed for random number generator	123
`X_calib`	Calibration variables	None
`totals`	Population totals	None
`custom_indicator`	Customized indicators	None
`na.rm`	Deletion of observations with missing values	No deletion

Table 4: Input arguments for function `direct`. All explanations can also be found in the documentation of the `direct` function in the package.

outputs residual diagnostics. In particular, results include the skewness and kurtosis of both sets of residuals (random effects and unit-level) and the results from using the Shapiro-Wilk test for normality (test statistic and p-value). The intra-cluster correlation (ICC) coefficient is further used for assessing the remaining unobserved heterogeneity. Finally, the `summary` command gives information about the selected transformation. If the user opts for a Box-Cox transformation, the transformation parameter λ and the shift parameter s are reported.
In addition to the diagnostics provided by `summary`, **emdi** enables the use of graphical diagnostics (see Figure 1). The `plot` method outputs graphics of residual diagnostics. The first set of plots (Figure 1a) are Normal Quantile-Quantile (Q-Q) plots of Pearson unit-level residuals and standardized random effects. Figure 1b and 1c are kernel density plots of the distribution of the two sets of residuals contrasted against a standard normal distribution. Outliers can have a significant impact on the model fit and hence on prediction. Hence, a Cook's distance plot is also available (Figure 1d), in which the three largest values of the standardized residuals are identified alongside the case identification number for further investigation. Finally, if a Box-Cox transformation is used, a plot of the profile log-likelihood that shows the value of the transformation parameter for which the likelihood is maximized is also produced (Figure 1e). The user can customize the format of all plots. Method `plot` accepts the parameter `label` with the predefined values `blank` (deletes all labels) and `no_title` (axis labels are given, but no plot titles). In addition, a user-defined list that contains specific labels for each plot list can be given. Another parameter available is `color` which accepts a vector with two color specifications. The first color defines the lines in Figure 1a, 1d and 1e and the second one specifies the color of the shapes in Figure 1b and 1c. For the likelihood plot the range in which the likelihood should be computed can be specified by using the parameter `range`. The appearance of the plots benefits from the use of the **ggplot2** package (Wickham 2009). Hence, `plot` accepts a `gg_theme` argument that allows for all customization options of `theme` that is

Arguments	**Short description**	**Default**
`fixed`	Fixed effects formula of the nested error regression model	
`pop_data`	Census or administrative data	
`pop_domains`	Domain identifier for population data, `pop_data`	
`smp_data`	Survey data	
`smp_domains`	Domain identifier for sample data, `smp_data`	
`L`	Number of Monte Carlo iterations	50
`threshold`	Threshold for poverty indicators	60% of the median of the target variable
`transformation`	Type of transformation: no, log or Box-Cox	Box-Cox
`interval`	Interval for the estimation of the optimal transformation parameter	(-1,2)
`MSE`	Mean Squared Error (MSE) estimation	No MSE estimation
`B`	Number of bootstrap populations	50
`seed`	Seed for random number generator	123
`boot_type`	Type of bootstrap: parametric or wild	Parametric
`parallel_mode`	Mode of parallelization	Automatic
`cpus`	Number of kernels for parallelization	1
`custom_indicator`	Customized indicators	None
`na.rm`	Deletion of observations with missing values	No deletion

Table 5: Input arguments for function `ebp`. All explanations can also be found in the documentation of the `ebp` function in the package.

a tool for modifying non-data components of a plot.

Example using Austrian districts:
In order to check the diagnostics in our example we use the `summary` and the `plot` methods. The summary output of the object `emdi_model` is presented below.

```
R> summary(emdi_model)

Empirical Best Prediction

Call:
ebp(fixed = eqIncome ~ gender + eqsize + cash + self_empl + unempl_ben +
age_ben + surv_ben + sick_ben + dis_ben + rent + fam_allow +
house_allow + cap_inv + tax_adj, pop_data = eusilcA_pop,
pop_domains = "district", smp_data = eusilcA_smp, smp_domains = "district",
threshold = 10885.33, MSE = TRUE,
custom_indicator = list(my_max = function(y, threshold) {
        max(y)
}, my_min = function(y, threshold) {
        min(y)
}))

Out-of-sample domains:  24
In-sample domains:  70

Sample sizes:
Units in sample:  1945
Units in population:  25000
                   Min. 1st Qu. Median      Mean 3rd Qu. Max.
Sample_domains       14    17.0   22.5  27.78571   29.00  200
Population_domains    5   126.5  181.5 265.95745  265.75 5857

Explanatory measures:
Marginal_R2 Conditional_R2
  0.6325942       0.709266

Residual diagnostics:
               Skewness Kurtosis Shapiro_W    Shapiro_p
Error         0.7523871 9.646993 0.9619824 3.492626e-22
Random_effect 0.4655324 2.837176 0.9760574 1.995328e-01

ICC:  0.2086841

Transformation:
Transformation Method Optimal_lambda Shift_parameter
       box.cox   reml      0.6046901               0
```

This output helps to justify the use of a model-based approach for SAE in this specific example. On the one hand, 24 out of 94 districts are out-of-sample such that direct estimates cannot be produced for these districts. Furthermore, the sample sizes in the districts are rather small with a median of 22.5 households and vary between a minimum of 14 households and a maximum of 200 households. The explanatory power of the selected covariates is high with the conditional R^2, the measure that jointly considers the fixed and the random effect, of around 71%. The ICC of 20.9% further justifies the inclusion of a random effect. The normality tests show that normality is rejected for the unit-level error term but not for the random effect. The use of transformations helps to reduce the skewness of the distribution of the error terms. The optimal transformation parameter is 0.6 indicating that neither using the untransformed income or the logarithmic transformation of income would be appropriate for this data set. The plots in Figure 1 used for residual analyses of the object `emdi_model` can be produced as follows,

```
R> plot(emdi_model, label = "no_title", color = c("red3", "red4"))
```

The Q-Q plots and the densities of the two error terms confirm that normality seems to be reasonable for the random effect but not for the unit-level error term. Furthermore, the Cook's distance plot identifies possible outliers. The last plot shows the optimal transformation parameter, which is the maximum of the profile log-likelihood.

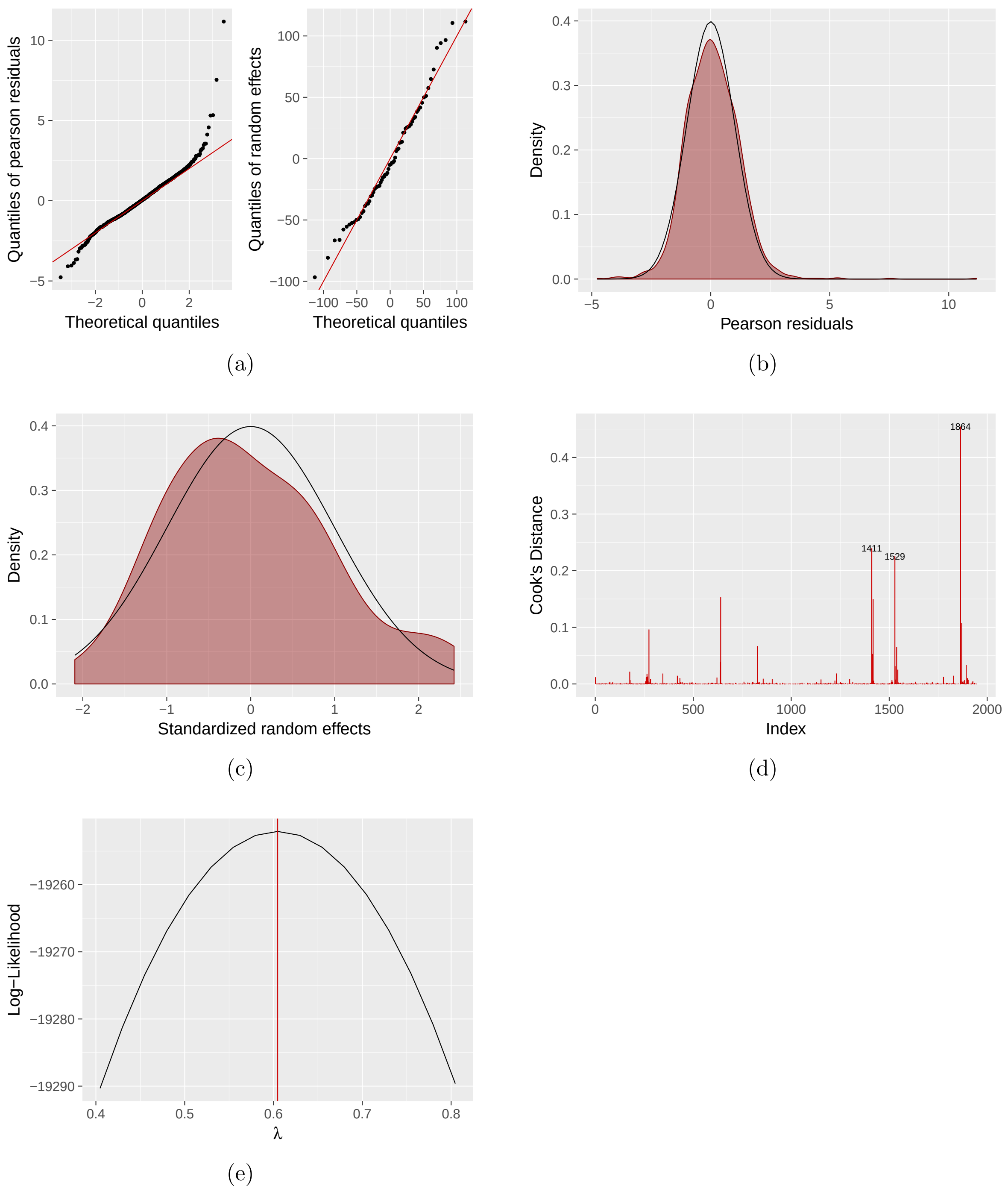


Figure 1: Graphics obtained by using `plot(emdi_model)`. (a) shows Normal Q-Q plots of the unit-level errors and the random effects. (b) and (c) show kernel density estimates of the distributions of standardized unit-level errors and standardized random effects compared to a standard normal distribution (black density). The Cook's distance plot is displayed in (d) whereby the index of outliers is labeled. The profile log-likelihood for the optimal parameter value of the Box-Cox transformation is shown in (e).

4.3. Selection and comparison of indicators

Package **emdi** returns a set of predefined and customized indicators. The ten predefined indicators are summarized in Table 1. However, the user may only be interested in some of these or only in individually defined (customized) indicators. A function called `estimators` helps the user to select the indicator or indicators of interest. This is done by using the `indicator` argument that takes a vector of indicator names as an argument, but in addition also accepts keywords defining predefined groups; for example, the keyword `custom` returns only user-defined indicators. In addition to variance and MSE estimates, NSIs often use the CV as an additional measure of the quality of the estimates. Estimated CVs as defined in Section 2 can be returned alongside MSE estimates.

It is often important to compare model-based and direct estimates. Direct estimates do not depend on the use of a model and hence the analyst should be interested in deriving model-based estimates that are close to direct estimates. Comparing model-based to direct estimates offers an internal validation procedure for checking whether the use of a model leads to unreasonable estimates. Package **emdi** provides a function called `compare_plot` that returns two plots, a scatter plot according to Brown *et al.* (2001) and a line plot. The scatter plot shows the direct and model-based point estimates, the fitted regression line, and the identity line. The closer the regression line is to the identity line, the closer the estimates are. The line plot is shown for domains ordered by the sample size. Thus, the user can see how the model-based estimates track the direct estimates across domains. In accordance with the function `estimators` the user can choose which indicators are compared by using the `indicator` argument. Similarly to the diagnostic plots, the user can modify the layout of the two plots. The label options are also `blank` (deletes all labels) and `no_title` (axis labels are given, but no plot titles). The color, the shape of the points and the type of the lines can be changed by using arguments `color`, `shape` and `line_type`, respectively.

Example using Austrian districts:

We illustrate how to estimate the median of equivalized income and the Gini coefficient and the corresponding CV estimates for the first 6 districts in Austria.

```
R> head(estimators(emdi_model, indicator = c("Gini", "Median"),
+    MSE = FALSE, CV = TRUE))
```

```
               Domain      Gini    Gini_CV   Median  Median_CV
1 Eisenstadt-Umgebung 0.2214688 0.09790984 25414.07 0.10381883
2  Eisenstadt (Stadt) 0.2872751 0.06110093 49274.84 0.07673551
3             Güssing 0.1906263 0.13046770 16718.13 0.12732081
4         Jennersdorf 0.2098103 0.15371048 12869.55 0.17815504
5         Mattersburg 0.2091353 0.10851693 20102.09 0.12764578
6     Neusiedl am See 0.1865026 0.05934130 18386.83 0.06346778
```

For these districts, the Gini coefficient and the median income are highest in Eisenstadt (Stadt). The lowest Gini is in Neusiedl am See and the lowest median in Jennersdorf. Furthermore, it can be noted that none of the CVs is above 20%. This threshold is used by the ONS in UK in order to decide if estimates can be reported.

The plots in Figure 2 are obtained by

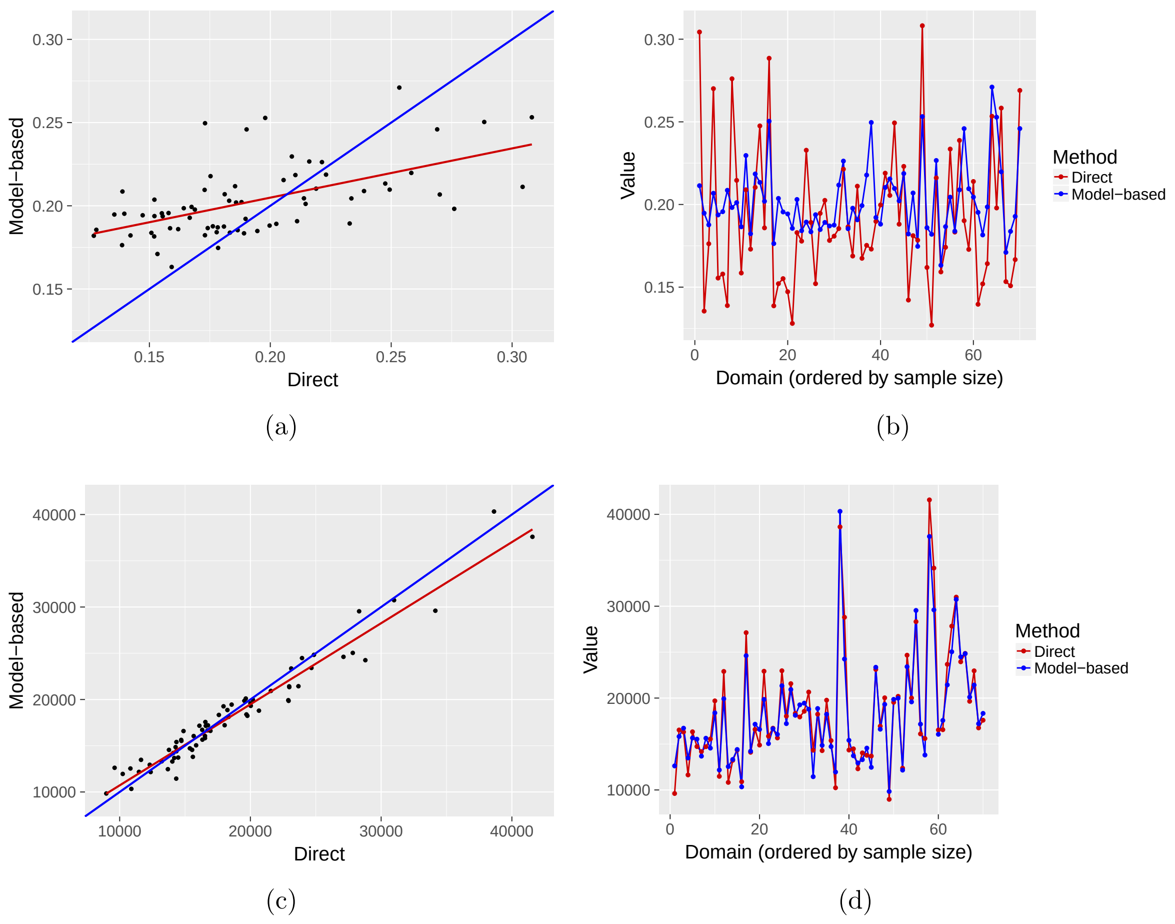


Figure 2: Graphics obtained by using `compare_plot(emdi_model)`. (a) and (c) show the scatter plots of the direct and model-based estimates for the Gini coefficient (top) and the median (bottom), respectively. (b) and (d) are line plots of the same estimates where the domains are ordered by increasing sample size.

```
R> compare_plot(emdi_direct, emdi_model, indicator = c("Gini", "Median"),
+    label = "no_title", color = c("red3", "blue"))
```

The scatter plots highlight that the disparity of the fitted regression line from the identity line is higher for the Gini coefficient than for the median. The model-based estimates do not track the direct estimates and show also a lower variability across the domains. In contrast, the direct and model-based estimates for the median are close to each other. Especially for large domains the difference is negligible.

4.4. Mapping of the estimates

In SAE maps are a natural way to present the estimates as they help describing the spatial distribution of issues like poverty and inequality. Creating maps can be demanding or laborious in practice. Package **emdi** includes function `map_plot` that simplifies the creation of maps. Given a spatial polygon provided by a shape file and a corresponding `emdi` object

pop_data_id	shape_id
ID of domain 1 in the emdi obj	ID of domain 1 in the shape file
ID of domain 2 in the emdi obj	ID of domain 2 in the shape file
ID of domain 3 in the emdi obj	ID of domain 3 in the shape file
⋮	⋮

Table 6: Example of a mapping table for argument `map_tap` in function `map_plot` in **emdi**.

`map_plot` produces maps of selected indicators and corresponding MSE and CV estimates. The parameters `MSE`, `CV` and `indicator` correspond to those in the `estimators` function. As Wickham (2009) points out the matching of domain identifiers in the statistical data to the corresponding identifiers in the spatial data (shape file) is challenging and general solutions are hard to obtain. The function `map_plot` in **emdi** allows for an argument `map_tab` when the identifiers do not match. The user must define a mapping table (cf. Table 6) for the argument `map_tab` in the form of a data frame that matches the domain variable in the population data set with the domain variable in the shape file. If the domain identifiers in both data sources match, this table is not required. The handling of the spatial shape files can be done using package **maptools** (Bivand and Lewin-Koh 2017) in combination with package **rgeos** (Bivand and Rundel 2017). Alternative approaches are provided by the packages **rgdal** (Bivand *et al.* 2018) and **sf** (Pebesma 2018). For general information on how to work with spatial data and shape files we refer the reader to Bivand *et al.* (2013).

Example using Austrian districts:
The steps for obtaining a map of median income in Austrian districts and the corresponding CVs are outlined below. First, the shape file needs to be loaded.

```
R> load_shapeaustria()
```

Then, two maps are created (cf. Figure 3).

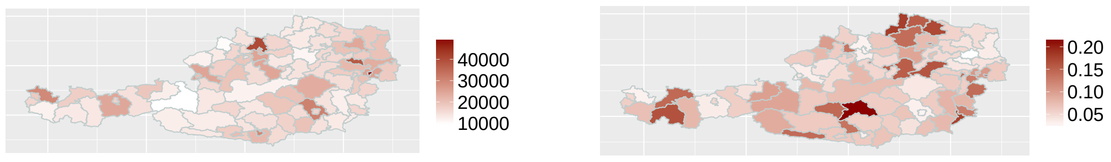
```
R> map_plot(emdi_model, MSE = FALSE, CV = TRUE, map_obj = shape_austria_dis,
+    indicator = "Median", map_dom_id = "PB")
```

As the domain identifiers in the data set and shape file already match, the argument `map_tab` is not required. For an example where the argument `map_tab` needs to be specified, we refer the reader to `help(map_plot)`.

The map of the median equivalized income in Figure 3 indicates differences across Austrian districts. The richest district appears to be Eisenstadt (Stadt) followed by Urfahr-Umgebung. Furthermore, throughout the country some districts have a relatively low median income like Zell am See and Schärding. The map of the CVs shows that most districts have a CV below 20%. The highest CVs occur in the out-of-sample domains.

4.5. Exporting the results

Exporting the results from R to other widely used software such as Excel™ is important for users. For doing so a large set of well established tools already exists. Nevertheless, exporting



(a) Median income. (b) CV of the median income estimates.

Figure 3: Maps of point estimates and CVs of the median income for 94 districts in Austria.

all model information, including the information contained in the summary output is not straightforward. Function `write.excel` creates a new Excel™ file that contains the summary output in the first sheet and the results from the selected estimators in the following sheet. Again the parameters `MSE`, `CV` and `indicator` correspond to those in the `estimators` function. The link with the Excel™ file format is done by using the package **openxlsx** (Walker 2017). This package does not require a Java™ installation, which offers an advantage over the use of the **xlsx** package (Dragulescu 2014) because Java™ may be seen as a potential security threat. Nevertheless, package **openxlsx** (Walker 2017) needs a zipping application available to R. Under Microsoft Windows™ this can be achieved by installing RTools while under macOS™ or Linux™ such an application is available by default. In addition to exporting the results to Excel™, **emdi** also provides an option to export output directly as OpenDocument Spreadsheets via the function `write.ods`.

Example using Austrian districts:
Excel™ outputs of model-based estimates for Austrian districts can be obtained by the following command.

```
R> write.excel(emdi_model, file = "excel_output.xlsx", indicator = "Median",
+     MSE = FALSE, CV = TRUE)
```

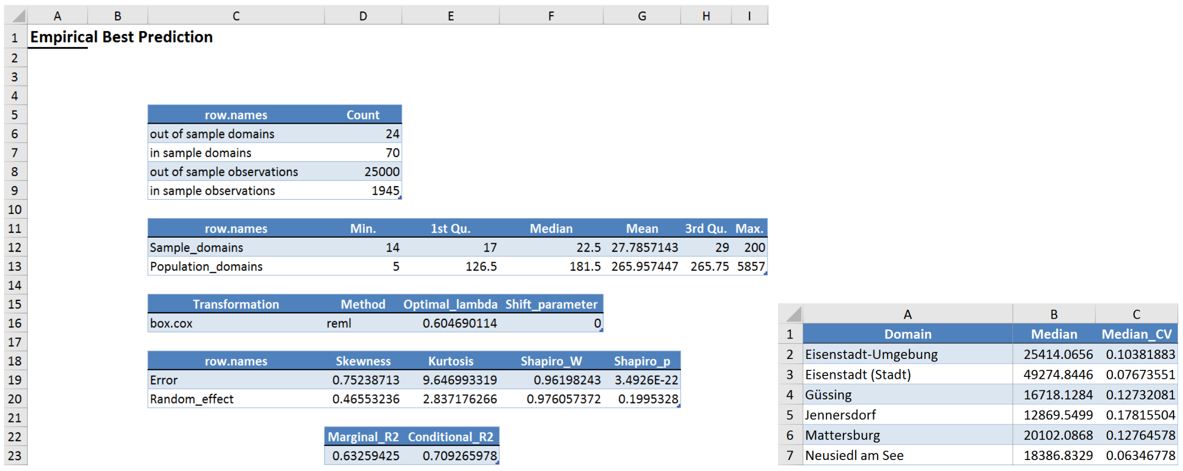
The output is presented in Figure 4 and shows that also the Excel™ user receives the same diagnostics from the summary and results for selected estimates. The summary output is described in detail in Section 4.2.

5. Additional features

In addition to those features that are essential for estimating regional indicators, package **emdi** offers to incorporate external indicators and increases the computational efficiency of the MSE estimation by parallel computing. In this section we show how users can bring indicators from other R packages into **emdi** and how parallel computing can help with reducing the computational burden.

5.1. Incorporating an external indicator

A feature we should pay attention to is the ease by which indicators of other R packages can be brought into **emdi**. This is demonstrated by using the Theil index from the R package



Empirical Best Prediction

row.names	Count
out of sample domains	24
in sample domains	70
out of sample observations	25000
in sample observations	1945

row.names	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Sample_domains	14	17	22.5	27.7857143	29	200
Population_domains	5	126.5	181.5	265.957447	265.75	5857

Transformation	Method	Optimal_lambda	Shift_parameter
box.cox	reml	0.604690114	0

row.names	Skewness	Kurtosis	Shapiro_W	Shapiro_p
Error	0.75238713	9.646993319	0.96198243	3.4926E-22
Random_effect	0.46553236	2.837176266	0.976057372	0.1995328

Marginal_R2	Conditional_R2
0.63259425	0.709265978

Domain	Median	Median_CV
Eisenstadt-Umgebung	25414.0656	0.10381883
Eisenstadt (Stadt)	49274.8446	0.07673551
Güssing	16718.1284	0.12732081
Jennersdorf	12869.5499	0.17815504
Mattersburg	20102.0868	0.12764578
Neusiedl am See	18386.8329	0.06346778

Figure 4: Export of the summary output and estimates to Excel™.

ineq (Zeileis 2014). The Theil index describes economic inequality and thus can be also used in the application with the data of this paper. It belongs to a family of generalized entropy inequality measures and can be expressed by

$$\text{Theil}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \frac{y_{ij}}{\bar{y}} log \left(\frac{y_{ij}}{\bar{y}} \right),$$

where $\bar{y} = \frac{1}{n_i} \sum_{j=1}^{n_i} y_{ij}$ (Cowell 2011). The Theil index takes values from 0 to ∞ with 0 indicating equality and higher values increasing inequality (World Bank Institute 2005). As the function `ineq` only requires a numeric vector of the target variable, it can be straightforwardly wrapped into a form usable within the `direct` or `ebp` functions. Using the function `direct` the Theil index can be estimated as follows.

First, the package **ineq** needs to be installed and loaded.

```
R> install.packages("ineq")
R> library("ineq")
```

Subsequently, the function `ineq` with `type = "Theil"` can be given to the argument `custom_indicator`. As the function `direct` needs the arguments `y`, `weights` and `threshold`, these arguments have to be also specified in the newly defined function.

```
R> my_theil <- function(y, weights, threshold) {
+     ineq(x = y, type = "Theil")
+     }
```

The argument `custom_indicator` needs to include a named list of self-defined indicators.

```
R> my_indicators <- list(theil = my_theil)
R> emdi_direct2 <- direct(y = "eqIncome", smp_data = eusilcA_smp,
+     smp_domains = "district", weights = "weight", var = TRUE,
+     custom_indicator = my_indicators)
```

As the Theil index is now part of the `emdi` object, all methods shown in Section 4 can be also used for this newly defined inequality indicator. For instance, by estimating a customized indicator via function `direct` a bootstrap variance estimator is used and the `subset` method can be applied in order to get results for certain districts.

```
R> select_theil <- estimators(emdi_direct2, indicator = "theil", CV = TRUE)
R> subset(select_theil, Domain == "Wien")

   Domain     theil  theil_CV
67   Wien 0.1202542 0.1108617
```

5.2. Parallelization

Bootstrapping the MSE can be very costly in terms of computation time and the possibilities of speeding up are limited when staying within R. Nevertheless, as the bootstrap procedures described in Section 2.2 and Appendix A consist of B independent iterations, they are suitable for efficient parallel computing. In this particular case, parallelization may be described as follows:

1. The user predefines how many parallel processes (`cpus`) and bootstrap iterations (`B`) should be used in function `ebp`.
2. The bootstrap iterations are equally distributed on the parallel processes.
3. In each process the differences between EBP point estimates and the pseudo true values $\widehat{\Delta I}_{i,b} = \hat{I}_{i,b}^{EBP} - I_{i,b}$ (compare e.g., Appendix A) are calculated. This is done on different central processing units (CPUs) at the same time (parallel computing).
4. The results $\widehat{\Delta I}_{i,b}$ from all processes are combined and the MSE is estimated by $\widehat{MSE}\left(\hat{I}_i^{EBP}\right) = B^{-1}\sum_{b=1}^{B}\left(\widehat{\Delta I}_{i,b}\right)^2$.

In R there are numerous ways and packages for implementing parallel computing. The most used package in this context is **parallel** (R Core Team 2018), which mainly builds on the work of packages **snow** (Tierney *et al.* 2016) and **multicore** (Urbanek 2009 - 2014). These packages follow two different approaches for parallelization. Package **snow** launches a new version of R on each core. Those versions communicate with the master process through the so-called “socket”. Therefore, we will proceed calling this way of parallelization the socket approach. The second approach is called “forking” and is the approach developed in the **multicore** package. Forking duplicates the entire current version of R and shifts it to a new core. Forking has one crucial advantage: all slave processes share the same memory with the master process for any object that is not modified. This feature makes it very fast. Its disadvantage is that it is not available on Microsoft Windows™ operating systems. The **parallel** package allows for both approaches but uses different functions. These functions are given an unified interface by the package **parallelMap** (Bischl and Lang 2015). This interface for parallelization is used in **emdi**. In the `ebp` function the parallelization approach defaults to socket if a Microsoft Windows™ OS is detected and to forking otherwise. The parallelization is activated by setting the `cpus` argument to an integer value larger than 1. In the example below the computation time is measured when the number of CPUs is set equal to 1 and to 2, respectively:

```
R> system.time(emdi_model1 <- ebp(fixed = eqIncome ~ gender + eqsize +
+    cash + self_empl + unempl_ben + age_ben + surv_ben + sick_ben +
+    dis_ben + rent + fam_allow + house_allow + cap_inv + tax_adj,
+    pop_data = eusilcA_pop, pop_domains = "district",
+    smp_data = eusilcA_smp, smp_domains = "district", threshold = 10885.33,
+    MSE = TRUE, seed = 100, cpus = 1))
```

```
  user      system     elapsed
155.86        0.09      157.36
```

```
R> system.time(emdi_model2 <- ebp(fixed = eqIncome ~ gender + eqsize +
+    cash + self_empl + unempl_ben + age_ben + surv_ben + sick_ben +
+    dis_ben + rent + fam_allow + house_allow + cap_inv + tax_adj,
+    pop_data = eusilcA_pop, pop_domains = "district",
+    smp_data = eusilcA_smp, smp_domains = "district", threshold = 10885.33,
+    MSE = TRUE, seed = 100, cpus = 2))
```

```
user      system     elapsed
3.62        0.45       89.45
```

The return value `elapsed` from function `system.time` informs the user about the real time that has passed from submitting the command until completion. Hence, the time comparison shows that two parallel processes reduce the time that is needed for the `ebp` function to run approximately by half. Please note that computation times are not replicable.

Despite the advantages in terms of computation time, parallelization comes with a major drawback. The reproducibility of results that depends on random number generations is non trivial. The usual `set.seed()` command that is used in R to ensure reproducibility is not sufficient due to the different R sessions used in parallel computing. In the socket approach, the function `clusterSetRNGStream()` from the **parallel** package is used to provide reproducible random number streams to each process that are far apart from each other. Therefore, all processes would produce different but reproducible random numbers. When using the forking approach, reproducibility can be more easily achieved by simply using a different random number generator. In the `ebp` function, `set.seed(seed, kind = "L'Ecuyer")` is used to set the random number generation to L'Ecuyer (L'Ecuyer *et al.* 2002) which is based on L'Ecuyer (1999). The multiple substreams of random numbers are created by the **rstream** package (Leydold 2017) in both approaches. Please note that results obtained from parallel computation are only reproducible if the same number of processes and the same parallelization approach are used. The reproducibility is demonstrated below by reproducing the results with `cpus` equal to 2.

```
R> emdi_model22 <- ebp(fixed = eqIncome ~ gender + eqsize +
+    cash + self_empl + unempl_ben + age_ben + surv_ben + sick_ben +
+    dis_ben + rent + fam_allow + house_allow + cap_inv + tax_adj,
+    pop_data = eusilcA_pop, pop_domains = "district",
+    smp_data = eusilcA_smp, smp_domains = "district", threshold = 10885.33,
+    MSE = TRUE, seed = 100, cpus = 2)
```

```
R> all.equal(emdi_model2, emdi_model22)

[1] TRUE
```

6. Conclusion and future developments

In this paper we show how the **emdi** package can simplify the application of SAE methods. This package is, to the best of our knowledge, the first R SAE package that supports the user beyond estimation in the production of complex, non-linear indicators. Another important feature is that data-driven transformation parameters are estimated automatically. Estimating the uncertainty of small area estimates is achieved by using both parametric bootstrap and semi-parametric wild bootstrap. The additional uncertainty due to the estimation of the transformation parameter is also captured in MSE estimation. Customized parallel computing is included for reducing the computational time. The complexity in applying SAE methods is considerably reduced, useful diagnostic tools are incorporated and the user is also supported by the availability of tools for presenting, visualizing and further processing the results. For instance, the model summary and results can be exported to Excel™ and to OpenDocument Spreadsheets. Since **emdi** makes the application of SAE methods in R almost as simple as fitting a linear or a generalized linear regression model, it also has the potential to close the gap between theoretical advances in SAE and their application by practitioners.

Additional features will be integrated in future versions of the package. Firstly, the implementation of alternative SAE methods will increase the usage of the package. For example, the World Bank (Elbers *et al.* 2003) and M-Quantile (Chambers and Tzavidis 2006; Tzavidis *et al.* 2010) methods complement the EBP approach (Molina and Rao 2010) for estimating disaggregated complex, non-linear indicators. Secondly, including additional evaluation and diagnostic tools for comparing direct and model-based estimates will assist the user with deciding which estimation method should be preferred. Thirdly, currently **emdi** includes only some possible types of transformations and one estimation method for the transformation parameter, namely REML. Future versions of the package will include a wider range of transformations (e.g., log shift and dual power transformations) and alternative estimation methods (minimization of the skewness or measures of symmetry) for the transformation parameter.

Acknowledgments

Rojas-Perilla, Schmid and Tzavidis gratefully acknowledge support by grant ES/N011619/1 - Innovations in Small Area Estimation Methodologies from the UK Economic and Social Research Council. The work of Kreutzmann and Schmid has been also supported by the German Research Foundation within the project QUESSAMI (SCHM 3113/2-1). The numerical results are not official estimates and are only produced for illustrating the methods.

References

Alfons A, Templ M (2013). "Estimation of Social Exclusion Indicators from Complex Surveys: The R Package **laeken**." *Journal of Statistical Software*, **54**(15), 1–25. URL http://www.jstatsoft.org/v54/i15/.

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/article/view/v037i03.

Barton K (2018). ***MuMIn**: Multi-Model Inference.* R package version 1.40.4, URL https://CRAN.R-project.org/package=MuMIn.

Bates D, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using **lme4**." *Journal of Statistical Software*, **67**(1), 1 – 48.

Battese G, Harter R, Fuller W (1988). "An Error-Components Model for Prediction of County Crop Areas Suing Survey and Satellite Data." *Journal of the American Statistical Association*, **83**(401), 28–36.

Bischl B, Lang M (2015). ***parallelMap**: Unified Interface to Parallelization Back-Ends.* R package version 1.3, URL https://CRAN.R-project.org/package=parallelMap.

Bivand R, Keitt T, Rowlingson B (2018). ***rgdal**: Bindings for the 'Geospatial' Data Abstraction Library.* R package version 1.2-18, URL https://CRAN.R-project.org/package=rgdal.

Bivand R, Lewin-Koh N (2017). ***maptools**: Tools for Reading and Handling Spatial Objects.* R package version 0.9-2, URL https://CRAN.R-project.org/package=maptools.

Bivand R, Pebesma E, Gómez-Rubio V (2013). *Applied Spatial Data Analysis with R.* Springer-Verlag New York.

Bivand R, Rundel C (2017). ***rgeos**: Interface to Geometry Engine - Open Source ('GEOS').* R package version 0.3-26, URL https://CRAN.R-project.org/package=rgeos.

Boonstra H (2012). ***hbsae**: Hierarchical Bayesian Small Area Estimation.* R package version 1.0, URL https://CRAN.R-project.org/package=hbsae.

Breidenbach J (2015). ***JoSAE**: Functions for Some Unit-Level Small Area Estimators and Their Variances.* R package version 0.2.3, URL https://CRAN.R-project.org/package=JoSAE.

Brown G, Chambers R, Heady P, Heasman D (2001). "Evaluation of Small Area Estimation Methods - An Application to Unemployment Estimates from the UK LFS." *Proceedings of Statistics Canada Symposium.*

Bundesamt für Eich- und Vermessungswesen (2017). "Verwaltungsgrenzen (VGD) - 1:250.000 Bezirksgrenzen, Daten vom 01.04.2017 von SynerGIS." [accessed: 07.02.2018], URL http://data-synergis.opendata.arcgis.com/datasets/bb4acc011100469185d2e59fa4cae5fc_0.

Chambers J, Hastie T (1992). *Statistical Models in S*. Wadsworth & Brooks/Cole computer science series. Wadsworth & Brooks/Cole Advanced Books & Software.

Chambers R, Tzavidis N (2006). "M-Quantile Models for Small Area Estimation." *Biometrika*, **93**(2), 255–268.

Council of the European Union (2001). "Report on Indicators in the Field of Poverty and Social Exclusions." *Report*, European Union.

Cowell F (2011). *Measuring Inequality*. Oxford University Press.

Dragulescu AA (2014). ***xlsx**: Read, Write, Format Excel™ 2007 and Excel™ 97/2000/XP/2003 Files*. R package version 0.5.7, URL https://CRAN.R-project.org/package=xlsx.

Elbers C, Lanjouw J, Lanjouw P (2003). "Micro-Level Estimation of Poverty and Inequality." *Econometrica*, **71**(1), 355–364.

EURAREA Consortium (2004). *Enhancing Small Area Estimation Techniques to meet European Needs*. Project Reference Volume, Deliverable 7.1.4.

Eurostat (2004). "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.

Feng X, He X, Hu J (2011). "Wild Bootstrap for Quantile Regression." *Biometrika*, **98**(4), 995–999.

Flachaire E (2005). "Bootstrapping Heteroskedastic Regression Models: Wild Bootstrap vs. Pairs Bootstrap." *Computational Statistics & Data Analysis*, **49**(2), 361–376.

Foster J, Greer J, Thorbecke E (1984). "A Class of Decomposable Poverty Measures." *Econometrica*, **52**(3), 761–766.

Gini C (1912). *Variabilità e Mutabilità. Contributo allo Studio delle Distribuzioni e delle Relazioni Statistiche*. P. Cuppini, Bologna.

Gómez-Rubio V, Best N, Richardson S, Li G, Clarke P (2010). "Bayesian Statistics for Small Area Estimation." Technical Report [accessed: 27.02.2018], URL http://eprints.ncrm.ac.uk/1686/1/BayesianSAE.pdf.

Gómez-Rubio V, Salvati N, *et al.* (2008). ***SAE2**: Small Area Estimation with R*. R package version 0.09, URL https://github.com/becarioprecario/SAE2.

Hagenaars A, de Vos K, Zaidi M (1994). *Poverty Statistics in the Late 1980s: Research Based on Mirco-data*. Office for the Official Publications of the European Communities.

Kreutzmann AK, Pannier S, Rojas-Perilla N, Schmid T, Templ M, Tzavidis N (2018). ***emdi**: Estimating and Mapping Disaggregated Indicators*. R package version 1.1.2, URL https://CRAN.R-project.org/package=emdi.

L'Ecuyer P (1999). "Good Parameters and Implementations for Combined Multiple Recursive Random Number Generators." *Operations Research*, **47**(1), 159–164.

L'Ecuyer P, Simard R, Chen EJ, Kelton WD (2002). "An Object-Oriented Random-Number Package with Many Long Streams and Substreams." *Operations Research*, **50**(6), 1073–1075.

Leydold J (2017). ***rstream**: Streams of Random Numbers.* R package version 1.3.5, URL `https://CRAN.R-project.org/package=rstream`.

Lopez-Vizcaino E, Lombardia M, Morales D (2014). ***mme**: Multinomial Mixed Effects Models.* R package version 0.1-5, URL `https://CRAN.R-project.org/package=mme`.

Molina I, Marhuenda Y (2015). "**sae**: An R Package for Small Area Estimation." *The R Journal*, **7**(1), 81–98. URL `https://journal.r-project.org/archive/2015/RJ-2015-007/index.html`.

Molina I, Rao J (2010). "Small Area Estimation of Poverty Indicators." *The Canadian Journal of Statistics*, **38**(3), 369–385.

Mukhopadhyay PK, McDowell A (2011). "Small Area Estimation for Survey Data Analysis Using SAS Software." *Paper 336-2011*, SAS Institute Inc.

Nakagawa S, Schielzeth H (2013). "A General and Simple Method for Obtaining R^2 from Generalized Linear Mixed-Effects Models." *Methods in Ecology and Evolution*, **4**(2), 133–142.

Pebesma E (2018). ***sf**: Simple Features for R.* R package version 0.6-0, URL `https://CRAN.R-project.org/package=sf`.

Pfeffermann D (2013). "New Important Developments in Small Area Estimation." *Statistical Science*, **28**(1), 40–68.

Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2018). ***nlme**: Linear and Nonlinear Mixed Effects Models.* R package version 3.1-131.1, URL `https://cran.r-project.org/web/packages/nlme/index.html`.

Pratesi M, Salvati N (2009). "Small Area Estimation in the Presence of Correlated Random Area Effects." *Journal of Official Statistics*, **25 (1)**, 37–53.

Rao JNK, Molina I (2015). *Small Area Estimation.* John Wiley & Sons.

R Core Team (2018). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Rojas-Perilla N, Pannier S, Schmid T, Tzavidis N (2017). "Data-Driven Transformations in Small Area Estimation." Discussion Paper 30/2017, School of Business and Economics, Freie Universiät Berlin.

Schmid T, Tzavidis N, Münnich R, Chambers R (2016). "Outlier Robust Small Area Estimation under Spatial Correlation." *Scandinavian Journal of Statistics*, **43**, 806–826.

Schoch T (2012). "Robust Unit-Level Small Area Estimation: A Fast Algorithm for Large Datasets." *Austrian Journal of Statistics*, **41**(4), 243–265.

Shi C, with contributions from Peng Zhang (2013). ***BayesSAE**: Bayesian Analysis of Small Area Estimation.* R package version 1.0-1, URL https://CRAN.R-project.org/package=BayesSAE.

Statistik Austria (2013). *Registerbasierte Statistiken Demographie (RS).* Schnellbericht 10.7.

Thai HT, Mentré F, Holford NH, Veyrat-Follet C, Comets E (2013). "A Comparison of Bootstrap Approaches for Estimating Uncertainty of Parameters in Linear Mixed-Effects Models." *Pharmaceutical Statistics*, **12**(3), 129–140.

The World Bank (2007). "More Than a Pretty Picture: Using Poverty Maps to Design Better Policies and Interventions." *Report*, The International Bank for Reconstruction and Development - The World Bank.

The World Bank (2017). "Measuring Poverty." [accessed: 27.04.2017], URL http://www.worldbank.org/en/topic/measuringpoverty.

The World Bank Group (2013). "Software for Poverty Mapping." [accessed: 11.02.2016], URL http://go.worldbank.org/QG9L6V7P20.

Tierney L, Rossini AJ, Li N, Sevcikova H (2016). ***snow**: Simple Network of Workstations.* R package version 0.4-2, URL https://CRAN.R-project.org/package=snow.

Tzavidis N, Marchetti S, Chambers R (2010). "Robust Estimation of Small Area Means and Quantiles." *Australian and New Zealand Journal of Statistics*, **52**(2), 167–186.

Urbanek S (2009 - 2014). ***multicore**: Parallel Processing of R Code on Machines with Multiple Cores or CPUs.* URL https://cran.r-project.org/package=multicore.

Ushey K, McPherson J, Cheng J, Atkins A, Allaire J (2018). ***packrat**: A Dependency Management System for Projects and their R Package Dependencies.* R package version 0.4.9-1, URL https://CRAN.R-project.org/package=packrat.

Walker A (2017). ***openxlsx**: Read, Write and Edit XLSX Files.* R package version 4.0.17, URL https://CRAN.R-project.org/package=openxlsx.

Warnholz S (2016). ***saeRobust**: Robust Small Area Estimation.* R package version 0.1.0, URL https://CRAN.R-project.org/package=saeRobust.

West B, Welch K, Galecki A (2007). *Linear Mixed Models: A Practical Guide Using Statistical Software.* Taylor & Francis Group, LLC.

Wickham H (2009). ***ggplot2**: Elegant Graphics for Data Analysis.* Springer-Verlag, New York.

Wirtschaftskammer Österreich (2017). "Wirtschaftsdaten auf Bezirksebene." [accessed: 07.02.2018], URL https://www.wko.at/service/zahlen-daten-fakten/wirtschaftsdaten-bezirksebene.html.

World Bank Institute (2005). *Introduction to Poverty Analysis.*

Zeileis A (2014). ***ineq**: Measuring Inequality, Concentration, and Poverty.* R package version 0.2-13, URL https://CRAN.R-project.org/package=ineq.

A. Semi-parametric wild bootstrap

The semi-parametric wild bootstrap is implemented as follows,

1. Fit model 1 (using an appropriate transformation for $\mathbf{y}$) to obtain estimates $\hat{\boldsymbol{\beta}}, \hat{\sigma}_u^2, \hat{\sigma}_e^2, \hat{\lambda}$.
2. Calculate the sample residuals by $\hat{e}_{ij} = y_{ij} - \mathbf{x}_{ij}^\top \hat{\boldsymbol{\beta}} - \hat{u}_i$.
3. Scale and center these residuals using $\hat{\sigma}_e$. The scaled and centered residuals are denoted by $\hat{\epsilon}_{ij}$.
4. For $b = 1, ..., B$
   - (a) Generate $u_i^{(b)} \overset{iid}{\sim} N(0, \hat{\sigma}_u^2)$.
   - (b) Calculate the linear predictor $\eta_{ij}^{(b)}$ by $\eta_{ij}^{(b)} = \mathbf{x}_{ij}^\top \hat{\boldsymbol{\beta}} + u_i^{(b)}$.
   - (c) Match $\eta_{ij}^{(b)}$ with the set of estimated linear predictors $\{\hat{\eta}_k | k \in n\}$ from the sample by using nn
     $$\min_{k \in n} \left| \eta_{ij}^{(b)} - \hat{\eta}_k \right|$$
     and define $\tilde{k}$ as the corresponding index.
   - (d) Generate weights w from a distribution satisfying the conditions in Feng *et al.* (2011) where w is a simple two-point mass distribution with probabilities 0.5 at $w = 1$ and $w = -1$, respectively.
   - (e) Calculate the bootstrap population as $T(y_{ij}^{(b)}) = \mathbf{x}_{ij}^\top \hat{\boldsymbol{\beta}} + u_i^{(b)} + w_{\tilde{k}} |\hat{\epsilon}_{\tilde{k}}^{(b)}|$.
   - (f) Back-transform $T(y_{ij}^{(b)})$ to the original scale and compute the bootstrap population value $I_{i,b}$.
   - (g) Select the bootstrap sample and use the EBP method as described above.
   - (h) Obtain $\hat{I}_{i,b}^{EBP}$.
5. $\widehat{MSE}_{Wild} \left(\hat{I}_i^{EBP} \right) = B^{-1} \sum_{b=1}^{B} \left(\hat{I}_{i,b}^{EBP} - I_{i,b} \right)^2$.

A simulation study assessing the performance of the semi-parametric wild bootstrap is presented in Rojas-Perilla *et al.* (2017).

B. Reproducibility

The results presented in this paper were obtained under R version 3.4.4 on a 64-bit platform under Microsoft Windows 7™. The installed packages are listed in Table 7. A snapshot of the corresponding repository was created with the package **packrat** (Ushey *et al.* 2018) and is available from the authors' `GitHub` folder (`https://github.com/SoerenPannier/emdi.git`). To make use of this repository Git must be installed. The authors recommend the following workflow:

- Use the new project functionality from RStudio.

- Choose checkout from version control and select Git.
- Enter the repository URL: `https://github.com/SoerenPannier/emdi.git`.
- Wait until **packrat** finishes the initialization process.
- Restart RStudio.
- Enter the R command `packrat::restore()`.
- After the package installation has finished all packages are installed as documented in Table 7.

Affiliation:

Timo Schmid
Institute for Statistics and Econometrics
School of Business & Economics
Freie Universität Berlin
14195 Berlin, Germany
E-mail: `timo.schmid@fu-berlin.de`
URL: `http://www.wiwiss.fu-berlin.de/fachbereich/vwl/Schmid/Team/index.html`

Package	Version	Package	Version	Package	Version
assertthat	0.2.0	mgcv	1.8-23	stringi	1.1.7
backports	1.1.2	mime	0.5	stringr	1.3.0
BBmisc	1.11	minqa	1.2.4	testthat	2.0.0
BH	1.66.0-1	moments	0.14	tibble	1.4.2
boot	1.3-20	MuMIn	1.40.4	utf8	1.1.3
brew	1.0-6	munsell	0.4.3	viridisLite	0.3.0
cellranger	1.1.0	nlme	3.1-131.1	whisker	0.3-2
checkmate	1.8.5	nloptr	1.0.4	withr	2.1.2
cli	1.0.0	openssl	1.0.1	xml2	1.2.0
colorspace	1.3-2	openxlsx	4.0.17	base	3.4.4
commonmark	1.4	packrat	0.4.9-1	boot	1.3-20
crayon	1.3.4	parallelMap	1.3	class	7.3-14
curl	3.1	pillar	1.2.1	cluster	2.0.6
desc	1.1.1	pkgconfig	2.0.1	codetools	0.2-15
devtools	1.13.5	plyr	1.8.4	compiler	3.4.4
dichromat	2.0-0	praise	1.0.0	datasets	3.4.4
digest	0.6.15	R.cache	0.13.0	foreign	0.8-69
emdi	1.1.2	R.methodsS3	1.7.1	graphics	3.4.4
foreign	0.8-69	R.oo	1.21.0	grDevices	3.4.4
ggplot2	2.2.1	R.rsp	0.42.0	grid	3.4.4
git2r	0.21.0	R.utils	2.6.0	KernSmooth	2.23-15
glue	1.2.0	R6	2.2.2	lattice	0.20-35
gridExtra	2.3	RColorBrewer	1.1-2	MASS	7.3-49
gtable	0.2.0	Rcpp	0.12.16	Matrix	1.2-12
HLMdiag	0.3.1	RcppArmadillo	0.8.400.0.0	methods	3.4.4
hms	0.4.2	RcppEigen	0.3.3.4.0	mgcv	1.8-23
httr	1.3.1	readODS	1.6.4	nlme	3.1-131.1
ineq	0.2-13	readr	1.1.1	nnet	7.3-12
jsonlite	1.5	rematch	1.0.1	parallel	3.4.4
labeling	0.3	reshape2	1.4.3	rpart	4.1-13
laeken	0.4.6	rgeos	0.3-26	spatial	7.3-11
lattice	0.20-35	rlang	0.2.0	splines	3.4.4
lazyeval	0.2.1	RLRsim	3.1-3	stats	3.4.4
lme4	1.1-15	roxygen2	6.0.1	stats4	3.4.4
magrittr	1.5	rprojroot	1.3-2	survival	2.41-3
maptools	0.9-2	rstudioapi	0.7	tcltk	3.4.4
MASS	7.3-49	scales	0.5.0	tools	3.4.4
Matrix	1.2-12	simFrame	0.5.3	utils	3.4.4
memoise	1.1.0	sp	1.2-7		

Table 7: Packages installed while producing the results presented in this paper.