mpitbR: A toolbox for calculating multidimensional poverty indices in R

by Ignacio Girela

Abstract Multidimensional poverty measurement is a vital tool for assessing the deprivations and well-being of people across different dimensions, such as health, education, and living standards. However, existing packages for multidimensional poverty measurement and estimation have some limitations, such as ignoring the complex survey design of microdata or user-friendliness. In this paper, we present **mpitbR**, a package for calculating multidimensional poverty indices based on the Alkire-Foster measurement approach, which accounts for the survey design, and offer various options and features for users. This package is the R version of the Stata `mpitb` package, which reproduces the workflow of the global Multidimensional Poverty Index developed by the United Nations Development Programme and Oxford University. The package provides functions for estimating the Alkire-Foster measures, as well as for analyzing poverty changes over time. The usage of the main functions and features of the **mpitbR** package are described and illustrated with an application over a synthetic data that has a typical household survey design.

1 Introduction

The first goal of the 2030 Agenda for Sustainable Development ambitiously postulates *Ending poverty in all its forms everywhere*. This call to global action recognizes poverty as not merely a lack of income but as a complex phenomenon with multiple dimensions that affect individuals' well-being.

To capture these diverse aspects of poverty, the last two decades have witnessed the development of various methodologies for multidimensional poverty measurement. Notably, the *dual-cut-off-counting* approach, proposed by Alkire and Foster (2011) (AF), has gained widely adoption in both academic research and policy-making spheres. The global Multidimensional Poverty Index (MPI), published annually by the United Nations Development Programme (UNDP) and the Oxford Poverty and Human Development Initiative (OPHI), exemplifies the relevance of the AF method (UNDP and OPHI 2022). Furthermore, numerous countries have built their national MPIs based on the AF framework to monitor and evaluate their poverty reduction strategies.

The AF method is lauded for its simplicity, comprehensiveness and flexibility for measuring multidimensional poverty. An index derived from the AF method can be disaggregated into several partial indices, known as AF measures, which are invaluable for policy analysis and design (Alkire 2021). Notwithstanding, calculating these measures in practice is far from straightforward. Poverty measurement typically depends on household survey data, and overlooking the complex survey design can result in biased estimates, leading to erroneous statistical inference, such as when comparing poverty levels across subgroups or changes over time.

Therefore, there is a need for a user-friendly and reliable tool that can estimate the AF measures and their associated standard errors, confidence intervals, and p-values, taking into account the survey design. To address this need, OPHI developed a Stata package called *mpitb*, which provides an integrated framework that mirrors the estimation process of the global MPI to researchers, analysts, and practitioners (Suppa 2023). This package offers a set of subcommands for estimating the key quantities of interest and storing them in a standardized format for further analysis. The main objective of this paper is to present a package that adapts the mpitb framework for R users. This package, named after **mpitbR**, enables users to estimate the AF measures and their standard errors and confidence intervals, using methods from the **survey** package (Lumley 2004). In addition, with the view of consistency, the **mpitbR** package provides functions with outputs that are equivalent to the Stata package subcommands.

The main contribution of this paper is to introduce the **mpitbR** package as a novel and useful tool for multidimensional poverty measurement and analysis in R. Although there exists some R packages available that compute AF measures, such as **MPI** and **mpindex**, these packages do not account for the complex survey design. They assume the data was obtained through a simple random sampling, which is rarely the case of household surveys. The paper focuses on demonstrating the functionality and features of the package through a series of examples and applications using a household survey data structure. It will be shown how the **mpitbR** package is consistent with the global MPI workflow and that it can handle different types of data and survey designs.

The remainder of this paper is organized as follows. Section 2 reviews the AF method and the MPI calculation. Section 3 describes the **mpitbR** package and its main functions. Section 4 illustrates the usage of the package with some examples and applications. Section 6 concludes with a summary and provides some suggestions for further extensions of the package.

2 Measuring multidimensional poverty: the Alkire-Foster Method

Alkire and Foster (2011) proposed a flexible approach to measure multidimensional poverty that can be tailored to different contexts and policy purposes. The flexibility of this method mainly derived from the so-called "dual cutoff counting approach" for identifying the poor and the possibility to build the MPI by aggregating different partial measures. Building an MPI based on the AF approach can be summarized in the following steps (for a detailed description, see Alkire et al. 2015):

1. Determine a set of dimensions of poverty $\mathcal{D}$ that are considered relevant for human development in a specific context (e.g., the global MPI chooses dimensions of health, education, and living standards, but other dimensions can be chosen depending on the context and goals).
2. Select d indicators that represents deprivations in each dimension (e.g., child mortality and malnutrition are the two indicators that represents health dimension in the global MPI).

3. Assign weights to each dimension and indicator, reflecting their relative importance where w_j is represents the weight of the j-th indicator for $j = 1, \ldots, d$. In practice, indicators in each dimension are weighted equally such that $\sum_{j=1}^{d} w_j = 1$.
4. Set the indicators deprivation cutoffs, which define the minimum level of achievement required to be considered non-deprived in each indicator (e.g., the global MPI uses cutoffs of having at least five years of schooling, having access to electricity, etc., but different thresholds can be set reflecting desired standards).
5. Apply the deprivations cutoff vector to each of the n observations (individuals or households) and build the $n \times d$ deprivation matrix $\mathbf{g^0}$. Each element $\left[\mathbf{g^0}\right]_{ij}$ of this matrix is a binary variable. If $\left[\mathbf{g^0}\right]_{ij} = 1$, the i-th observation is deprived in indicator j, and the opposite if $\left[\mathbf{g^0}\right]_{ij} = 0$.
6. Build the weighted deprivation matrix $\bar{\mathbf{g}}^{\mathbf{0}}$ assigning the corresponding weight to each indicator (i.e., $\bar{\mathbf{g}}^{\mathbf{0}} = \mathbf{g^0} \times \mathbf{diag}(w_1, \ldots, w_d)$) and calculate the deprivations score for each observation c_i, which is the weighted sum of the deprivations $c_i = \sum_{j=1}^{d} w_j d_{ij}$.
7. Identify who is poor by setting a unique poverty cutoff k meaning the minimum proportion of weighted deprivations a household needs to experience to be considered multidimensional poor. This cutoff is compared with the deprivations score. Therefore, if $c_i \geq k$, the person is multidimensional poor (e.g., the global MPI uses a cutoff k of 33.3% or 1/3 which means that a person is poor if it is deprived in one-third or more of the weighted indicators).
8. Censor data of the non-poor and get the so-called censored (weighted) deprivation matrix ($\bar{\mathbf{g}}^{\mathbf{0}}(k)$), and censored deprivations scores ($c(k)$), where $c_i(k) = c_i$ if $c_i \geq k$, and $c_i(k) = 0$ otherwise.
9. Compute the MPI (M_0) by taking the mean of the censored deprivation score ($c(k)$):

$$M_0 = \frac{1}{n} \sum_{i=1}^{n} c_i(k)$$

As mentioned above, the MPI can be re-expressed as function of other partial measures which we referred as AF measures. For instance,

- From the latter expression, by multiplying and dividing by the number of people identified as poor (q)

$$M_0 = \frac{q}{n} \times \frac{1}{q} \sum_{i=1}^{n} c_i(k) = H \times A$$

we obtained the intensity (H) and intensity (A) of poverty. The former is the proportion of multidimensional poor people while the latter are the average weighted deprivations suffered by the poor.

- From the censored deprivation matrix $\mathbf{g^0}(k)$, if we take the mean of each column, we obtain the censored indicators headcount ratios $h_j(k)$ which mean the proportion of people deprived in indicator *j and* are multidimensional poor for $j, \ldots, d$. From these measures we can also arrive to the MPI:

$$M_0 = \sum_{j=1}^{d} w_j h_j(k)$$

Naturally, we can obtain the uncensored indicators headcount ratios h_j before censoring the non-poor from data $\mathbf{g^0}$.

- Also we can decompose not only the MPI but also every partial measure (H, A, h_j, $h_j(k)$) by different population subgroups (age, regions, etc.).

$$M_0 = \sum_{l=1}^{L} \phi^l M_0^l$$

where ϕ^l is the population share of the l-th subgroup for $l = 1, \ldots, L$.

Finally, the absolute $w_j h_j(k)$ and proportional $w_j h_j(k)/M_0$ contribution of each indicator to poverty are usually reported.

Changes over time

Alkire, Roche, and Vaz (2017) proposed a four measures to assess pro-poor multidimensional poverty reduction between two periods of time using repeated cross-sectional data. The following measures are explained using the Adjusted Headcount Ratio, however, all the changes-over-time measures can be extend to the aforementioned partial measures.

Let t^1 and t^2 denote the initial and final period, respectively. Then, $M_0^{t^1}$ and $M_0^{t^2}$ are their corresponding Adjusted Headcounts Ratio. Note that for comparability purposes, these two poverty measures must have the same set of parameters (indicators, weights, deprivations and poverty cutoffs).

We define the *absolute rate of change* (Δ) as the difference in M_0 between the final t^2 and the initial period t^1:

$$ \Delta(M_0)= M_0^{t^2} - M_0^{t^1}$$$$

The *relative rate of change* (δ) is defined as the difference in M_0 as a percentage of the initial poverty level t^1, i.e.,

$$\delta(M_0) = \frac{M_0^{t^2} - M_0^{t^1}}{M_0^{t^1}} \times 100 \tag{1}$$

On the other hand, the annualized versions of these two measures are used in order to compare changes over time across countries with different periods of reference. The *annualized absolute rate of change* ($\bar{\Delta}$) is the absolute rate of change as defined above divided by the difference in the two time periods:

$$\bar{\Delta}(M_0) = \frac{M_0^{t^2} - M_0^{t^1}}{(t^2 - t^1)} \tag{2}$$

Finally, the *annualized relative rate of change* ($\bar{\delta}$) is defined as the compound rate of reduction in M_0 per year between the initial and the final periods, i.e.,

$$\bar{\delta}(M_0) = \left[\left(\frac{M_0^{t^2}}{M_0^{t^1}} \right)^{\frac{1}{t^2 - t^1}} - 1 \right] \times 100 \tag{3}$$

An explanation of the usage of the main functions for estimating AF measures and their changes over time using the **mpitbR** package is provided in the following section.

3 Overview of mpitbR package

This toolbox package consists mainly of two functions, `mpitb.set` and `mpitb.est`, which are the two key tools of the original Stata package. In addition, **mpitbR** includes functions from the build-in R package **stats** `coef`, `confint`, and `summary`. Last but not least, **mpitbR** mainly depends upon **survey** package as we will see below.

`mpitb.set` function

In this first function, users specify the relevant information for estimating the MPI. For instance, they provide the the data, the deprivation indicators for the MPI, and other optional auxiliary character arguments to label the project, such as its name and a brief description.

The usage and input arguments of function `mpitb.set` are summarized as follows:

```
mpitb.set(data, indicators, ...,
          name = "unnamed", desc = "desc.")
```

The first argument should be a dataset containing information about the household survey. To consider the survey design, users should specify previously the survey structure using `svydesign` function from **survey** package and pass a "survey.design2"-class object to `data` argument. If `data` is an object of the "data.frame" class, it is coerced to a "survey.design2"-class object assuming simple random sampling design.

The `indicators` argument contains the name of the deprivation indicators corresponding to the columns names in `data`. There are two different ways to pass this argument: a list or a character vector. If it is a list, the user will define the set of dimensions and their corresponding indicators. Each element of the list represents a dimension containing a character vector with the name of the indicators associated to that dimension. This way is useful for automatically calculating equal nested weights in the subsequent estimations. At most 10 dimensions are allowed. If `indicators` is a character vector, that is, they are not grouped by dimensions. In this case, it is advisable to pass the weights of each indicator manually with a numeric vector when estimating the AF measures. Otherwise, all indicators will weight equally.

It is highly important to mention that it is assumed that the names of the indicators in `data` are the columns of the deprivation matrix $\mathbf{g}^0$. In other words, the column names in `data` corresponding to `indicators` should be binary variables. Another caveat that is worth mentioning is that these columns should not contain any missing value. The R **survey** package supports missing values for calculating the point estimates but it would not be able to calculate the standard error and, therefore, the confidence intervals.

Finally, `name` and `desc` arguments are useful for identifying each setting while working in a multidimensional poverty measurement and analysis project. The former is the project name and serves as an ID of the setting. It is recommended to use short names (at most 10 characters are permitted) while the latter is a character containing a brief description.

The output of this function is a "mpitb_set"-class object with the survey.design data, the name of the indicators and the setting name and description as attributes.

`mpitb.est` function

Once declared the relevant information for the multidimensional poverty measurement and analysis project, in the `mpitb.est` function all the parameters and the desired measures to be calculated are specified.

```
mpitb.est(set, klist = NULL, weights = "equal",
          measures = c("M0", "H", "A"),
          indmeasures = c("hd", "hdk", "actb", "pctb"),
          indklist = NULL,  over = NULL,  ...,
          cotyear = NULL,  tvar = NULL,
          cotmeasures = c("M0", "H", "A", "hd", "hdk"),
          ann = FALSE,  cotklist = NULL,
          cotoptions = "total",
          noraw = FALSE,  nooverall = FALSE,
          level = 0.95,
          multicore = getOption("mpitb.multicore"),
          verbose = TRUE)
```

This function is a S3 method for "mpitb_set"-class objects declared in `set` argument. Users pass the vector of poverty cut-offs (k) in `klist` argument in percentage format, i.e., numbers between 1 and 100. Information about the weighting scheme is defined in `weights` argument. By default it calculates the nested equal weights for each indicator. It can be a numeric vector of values between 0 and 1 such that all sum up to 1. Its values should match the order of the indicators in `mpitb.set`.

Arguments `measures` and `indmeasures` control the AF measures to be estimated. The former includes the MPI (or `"M0"`) and its disaggregation by incidence (`"H"`) and intensity (`"A"`) of poverty. The latter comprehend the rest of the indicator-specific AF measures such as the uncensored and censored indicators headcount ratios (`"hd"`and `"hdk"`) and their absolute and proportional contribution to overall poverty (`"actb"` and `"pctb"`). It is possible to specify a poverty cut-offs vector for these indicators-specific measures in `indklist`. If this argument is `NULL`, it is equal to `klist`. By default, all these measures are calculated, however, users can define the desired measures to save time.

If any of these arguments is `NULL`, `mpitb.est()` skips these measures. This avoids calculating unnecessary estimations. For instance, if `measures = c("M0")` and `indmeasures = NULL`, only the MPI will be estimated.

To specify the population subgroups (e.g., living area, sex, etc.) and estimate the disaggregated measures by each level of the subgroup, the users should pass the column names of the population subgroups in the data using `over` argument. If `NULL`, the measures are estimated using all the observations.

Changes over time

To analyze how multidimensional poverty changes over time, a variable to index the time or survey round is needed. Argument `tvar` is a character with the column name that references the time period. If annualized measures are to be estimated, it is required the values of the years of each round. `cotyear` argument is a character with the column name that have information about the years (decimal digits allowed).

The **mpitbR** permits estimating the changes over time of each AF measure for every population subgroup. This is declared in `cotmeasures` argument. By default, it includes all AF measures but users can modify it to save time. As with the indicator-specific measures, it is possible to specify a poverty cut-offs vector for the changes over time in `cotklist`. If `NULL`, it is equal to `klist`. Here it is worth mentioning that the standard errors of changes-over-time measures are estimated using Delta method (detailed in `svycontrast()` function from **survey** package).

If there are more than two survey rounds, `cotoptions` is used whether to estimate the changes over the total period of observation or year-to-year changes. To do the former, `cotoptions = "total"` whereas for the latter case, `cotoptions = "insequence"`.

Other arguments

Some additional logical arguments are included to avoid unnecessary estimations. `ann` is a control argument. If `TRUE`, annualized measures are calculated. If `cotyear` is passed, `ann` is automatically set to `TRUE`. If `cotyear` is not `NULL` and `ann` is `FALSE`, only non-annualized measures are estimated. In the case that only annualized measure are under study, the user can switch `noraw` to `TRUE` to avoid estimating non-annualized changes.

On the other hand, the users specify the population subgroups of interest in `over`. If `nooverall = TRUE`, estimations overall data (e.g., national-level) are not calculated.

Finally, by default confidence intervals are estimated using a confidence level of 95%. Users can change this in `level` argument. The confidence intervals are estimated considering measures as proportions using `svyciprop()` function from **survey** package (it uses the "logit" method which consists of fitting a logistic regression model and computes a Wald-type interval on the log-odds scale, which is then transformed to the probability scale).

mpitbR includes the possibility to do parallel calculations over all the measures and poverty cut-offs with the logical argument `multicore`. The package uses Forking method for parallelization. Hence, this option is only available on Unix-like systems.

Other functions

The output of `mpitb.est()` function mirrors the original Stata package. It is a two-elements list where each one is a `data.frame` containing all the estimations with the same format as the so-called Stata package. Then, users can apply functions such as `coef`, `confint`, and `summary`.

Since the output data may contains multiple AF measures, by cut-offs and levels of the subgroups and possibly by different time periods. Therefore, for user-friendliness purposes, it is required to filter data. For instance, all these methods are only available for one AF measure.

`summary` function performs a t-test over the estimates inheriting the confidence level in `level` argument of `mpitb.est()` function. The `summary` function is particularly relevant for the changes-over-time estimates where the user can infer if there have been or not a pro-poor multidimensional poverty reduction between two periods of time. In the case of changes over time, `summary` is available for one AF measure, one change-over-time measure and one poverty cut-off.

The next section provides examples of how to implement these functions in practice using a household survey.

4 Applications

This section includes illustrative examples for i) a traditional exercise of cross-sectional estimations for a single country, ii) how to use options of the pacakge functions to save time and avoid unnecessary estimations, iii) using alternative weighting schemes, and, if data permits, iv) study changes over time.

With the view of comparability, all the examples use the `syn_cdta` which is a synthetic data used in the Stata package examples (Suppa 2023).

Estimate AF measures for a single country in a single year

First, we load the **survey** and **mpitbR** packages.

```
library(survey)
library(mpitbR)
```

Table 1 prints the first few rows of the data set variables.

A synthetic household survey data

d_nutr

d_cm

d_satt

d_educ

d_elct

d_sani

d_wtr

d_hsg

d_ckfl

d_asst

area

region

stratum

psu

weight

year

t

0

0

1

0

0

0

0

1

0

0

1

4

401

401000

1

2010

1

0

0

0
0
1
0
0
1
1
1
1
1
104
104003
1
2010
1
0
0
0
0
1
0
0
0
0
0
1
20
2002
2002005
1
2010
1
0
0
0
0
0
0
0
1
0
0
1
20
2004
2004002
1
2010
1
0

Table 1: A synthetic household survey data

d_nutr	d_cm	d_satt	d_educ	d_elct	d_sani	d_wtr	d_hsg	d_ckfl	d_asst	area	region	stratum	psu	weight	yea
0	0	1	0	0	0	0	1	0	0	1	4	401	401000	1	201
0	0	0	0	1	0	0	1	1	1	1	1	104	104003	1	201
0	0	0	0	1	0	0	0	0	0	1	20	2002	2002005	1	201
0	0	0	0	0	0	0	1	0	0	1	20	2004	2004002	1	201
0	0	1	0	1	0	0	1	0	0	1	18	1805	1805000	1	201
0	0	1	0	1	1	0	1	0	1	0	18	1803	1803001	1	201

0

1

0

1

0

0

1

0

0

1

18

1805

1805000

1

2010

1

0

0

1

0

1

1

0

1

0

1

0

18

1803

1803001

1

2010

1

The first ten columns in our data set contains the ten global MPI indicators as binary variables. Following the order of the columns, these names stand for: Nutrition, Child Mortality, School Attendance, Education, Electricity, Sanitation, Water, Housing, Cooking Fuel, and Assets (see UNDP and OPHI 2022, for more details). In addition, there are two columns of population subgroups to decompose the MPI:`area` and `region`.

The data set consists of two surveys. The variables `year` and `t` provide information about the year and the number of each survey round. The survey design is defined by the variables that contains the primary sampling unit, the weights and the strata of each observation (`psu`, `weights` and `stratum`, respectively)

Prior to any calculations, it is necessary to define the design of our survey data. This is highly important because poverty indices are estimated using household surveys and accounting for the complex survey design ensures valid statistical inferences, accurate variance estimation, and representative estimates.

We restrict the analysis for a single year. In this case, the first round of the survey (i.e., `t == 1`). All the information of the survey structure of our data is declared using `svydesign` function from **survey** package. Our data contains some missing values in our indicators. Hence, we pass our data to the `svydesign` function omitting them.

```
# Subset data
syn_cdta1 <- subset(syn_cdta, t == 1)
# Drop NA in indicators columns
syn_cdta1 <- na.omit(syn_cdta1)
# Define survey design
svydata1 <- svydesign(id=~psu, weights = ~weight, strata = ~stratum, data = syn_cdta1)
```

To specify the MPI estimation settings, we use the `mpitb.set` function. Apart from the survey data, another required argument is `indicators`, in which we select the indicators columns with the possibility to organize them in a set of dimensions. This can be defined first in a list and the passed to the function as follows.

```
indicators <- list(hl = c("d_nutr","d_cm"),
                   ed = c("d_satt","d_educ"),
                   ls = c("d_elct","d_sani","d_wtr","d_hsg","d_ckfl","d_asst"))

set.trial01 <- mpitb.set(svydata1, indicators = indicators,
                         name = "trial01", desc = "pref. spec")
```

Note that the indicators are grouped in three dimensions (`hl`, `ed`, and `ls`) that stand for Health, Education and Living Standards (the three dimensions included in the global MPI estimations). Finally, we defined a short name (`"trial01"`) and description (`"pref. spec"`) to our project.

All the information about the specification of the multidimensional poverty measurement is stored in the `set.trial01` object, which is of the class "mpitb_set" and then passed to the `mpitb.est` S3 method in order to compute the indices.

For this first round assume we want to estimate all the AF measures. These are all the aggregate measures (M_0, H, and A) and the indicator-specific measures (h_d, $h_d(k)$, $actb$, and $pctb$). Also, we prefer a equal-nested weighting scheme (equal weights for all dimensions and equal indicator weights within dimensions) and for each measure we set three poverty cut-offs: 20%, 33%, and 50%. Finally, we also want to calculated the disaggregated measure by the different subnational regions and living areas (urban and rural), accounting for the complex design of the survey.

All this information is specified in the `mpitb.est` function as follows:

```
est.trial01 <- mpitb.est(set = set.trial01, klist = c(20, 33, 50),
                         weights = "equal",
                 measures = c("M0","H","A"),
                 indmeasures = c("hd", "hdk", "actb", "pctb"),
                 over = c("area","region"))
```

The "mpitb_set"-class object is passed in `set` argument. The poverty cut-offs in `klist` and the population subgroups in `over`. The equal-nested weighting scheme is specified in `weights` argument passing the character `"equal"`. The aggregate measures and the indicator-specific measure in `measures` and `indmeasures` respectively. However, by default all these measures and equal-nested weights are calculated. Hence, this arguments can easily be omitted in this case.

Users can verified the specification of their project with the `mpitb.est` messages. It reports 1) the function call to verified correct arguments assignment, 2) the dimensions with their assigned indicators and their corresponding weights, 3) which measures are being estimated, 4) the parameters of the estimation (number of poverty cut-offs and subgroups), and 5) other features such as the confidence level and if parallel estimation have been used. All these messages can be suppressed using `verbose = FALSE` in `mpitb.est` function.

The `mpitb.est` function returns a two-element list ("mpitb_est"-class), where each element is a data frame containing all the estimates. The first ('lframe') include all the cross-sectional estimates for each level of analysis, whereas the second ('cotframe') all the changes over time measures for each level of analysis. In this first example, 'cotframe' is `NULL`.

For this instance, the `lframe` first rows are as depicted:

```
head(est.trial01$lframe)
```

Each row represent an estimate from each measure and population subgroup. In this example, we have 2507 estimates. These data frames resemble the output of the so-called Stata package. The first columns include most important features such as the point estimate ("b") with the corresponding standard error ("se") that account for the complex survey design and the lower and upper confidence limits ("ll" and "ul") calculated with a given confidence level. Each row represent an estimate of a "measure" for a specific "indicator", if applied, given a cut-off "k" for each subgroup level of analysis, which can be tracked in the columns of "loa" and "subg".

Since the elements of the list are data frames, users can easily subset the 'lframe' or 'cotframe' elements to examine point estimates or confidence intervals of certain subgroups and measures of interest. For instance, assume we want to see the incidence H confidence intervals for the "area" subgroups for the cut-off 33%, the line code goes as follows:

```
confint(subset(est.trial01$lframe, measure == "H" & loa == "area" & k == 20))
```

Finally, summary calculates a t-test statistic for each point estimate, with the corresponding p-value and significance level to infer if it is statistically different to zero. Such analysis is more practical when it comes to make inference of any change-over-time measure. An example of this will be provided below.

mpitb.est arguments to avoid unnecessary estimations

As it is possible to observe from the previous example, the amount of estimates can soar for a small number of parameters (poverty cut-offs, population subgroups, etc.). Hence, in order to avoid unnecessary estimations and save time, it is important to determine which are the measures to be prioritized.

For example, although the deprivation scores, c_i and $c_i(k)$, are real-valued functions, for fixed values of weights and k , they will assume a finite number of values. Hence, it is advisable to be aware of which values of k are included to avoid estimating measures that will yield the same results. Furthermore, this is specially important if we incorporate different population subgroups into the analysis since adding subgroups in the main source of the increasing number of estimations.

In this package, it is possible to control which AF measures we want to estimate and to specify a separate list of poverty cut-off values for the aggregate measures and the indicator-specific measures with klist and indklist arguments.

In the following lines of code, we run the same last example with few changes. We rule out from the analysis the H and A measures and the contribution measures of each indicator. On the other hand, we specify a different poverty cut-off for the indicator specific measures and we avoid estimating national level estimates setting nooverall argument to TRUE.

```
mpitb.est(set = set.trial01, klist = c(20, 33, 50),
          weights = "equal", measures = c("M0"),
          indmeasures = c("hd", "hdk"), indklist = c(33),
          over = c("area","region"), nooverall = TRUE)
```

The number of estimates have been reduced drastically in comparison to the former example. In the previous estimations there were a total of 2507 estimates whereas in this case 946 estimates.

Specify alternative weighting schemes

In the previous examples we have assumed equal-nested weights across dimensions and indicators. In other words, equal weights for all dimensions and equal indicator weights within dimensions. We have specified this by passing the indicators grouped in a list to mpitb.set function and setting weights = "equal", which is the default value in mpitb.est function.

There is an alternative way to do this, which is passing a numeric vector to weights argument with the values of the weights corresponding to each indicator exemplified here below.

```
mpitb.est(set = set.trial01, klist = c(33),
          measures = c("M0"), indmeasures = NULL,
          weights = c(1/6, 1/6, 1/6, 1/6,
                      1/18, 1/18, 1/18, 1/18, 1/18, 1/18))
```

Now assume we assign a weight of 50% to Health dimension and 25% to the rest of dimension. If we take equal weights in each indicator within each dimension, the vector of weights will be specify as follows:

```
mpitb.est(set = set.trial01, klist = c(33),
          measures = c("M0"), indmeasures = NULL,
          weights = c(1/4, 1/4, 1/8, 1/8,
                      1/24, 1/24, 1/24, 1/24, 1/24, 1/24))
```

Another convenient feature of this workflow is that users can easily evaluate the effects of adding or dropping some indicators, merging two indicators, and comparing different deprivation thresholds and alternative selections of indicators. Such analysis, jointly with specifying alternative weighting schemes, is often used during the construction process of an MPI. The following lines of code show the same first example framework but dropping electricity indicator from the analysis.

```
indicators <- list(hl = c("d_nutr","d_cm"),
                   ed = c("d_satt","d_educ"),
                   ls = c("d_sani","d_wtr","d_hsg","d_ckfl","d_asst"))

set.trial02 <- mpitb.set(syn_cdta, indicators = indicators,
                 name = "trial02", desc = "w/o electricity")

est.trial02 <- mpitb.est(set = set.trial02, klist = c(20), weights = "equal",
                 measures = c("M0","H","A"),
                 indmeasures = c("hd", "hdk", "actb", "pctb"),
                 over = c("area","region"))
```

Estimate AF measures for a single country in several years

In order to analyse changes over time, we need a sequence of survey rounds for the same population. This workflow assumes that the microdata from the different survey are appended with a column that serves as an identifier. For instance, in `syn_cdta` data the variable `t` index the two available survey rounds (1 and 2).

In our first example, we subset the data selecting the rows corresponding to the first year round `t == 1`. Now we will use the entire data set to specify our survey design with the difference that we have to pass which is the column that indexes the survey rounds in the `tvar` argument of `mpitb.est` function.

```
# Drop NA in indicators columns
syn_cdta <- na.omit(syn_cdta)
# Define survey design
svydata <- svydesign(id=~psu, weights = ~weight, strata = ~stratum, data = syn_cdta)

indicators <- list(hl = c("d_nutr","d_cm"),
                   ed = c("d_satt","d_educ"),
                   ls = c("d_elct","d_sani","d_wtr","d_hsg","d_ckfl","d_asst"))

set.trialCOT <- mpitb.set(svydata, indicators = indicators, name = "trialCOT", desc = "Estimate changes over time

est.trialCOT <- mpitb.est(set = set.trialCOT, klist = c(1, 33, 50),
                 indmeasures = NULL,
                 over = c("region"), tvar = "t", cotyear = "year")
```

5 Summary

We have presented the main functions for estimating multidimensional poverty indices (MPI) with **mpitbR** package which aims to provide a thorough but user-friendly framework for both academics and practitioners of multidimensional poverty measurement.

Because the toolbox has been developed in the context of the global MPI, it is also tailored to its needs, whether in terms of the underlying data, the quantities produced out of the box, or the related forms of analysis.

Multidimensional poverty measurement and analysis is, however, an active field of research, where new measures, analyses, and other methodological innovations are still proposed and discussed. mpitb may already be useful for such endeavors and take some load off of researchers working these topics.

The very nature of mpitb as a toolbox seeks to allow for further features, novel analyses, and additional tools being added in the future. Likewise, adding support for novel complementary measures within the AF framework, for example, for the analysis of inequality among the poor (Alkire and Foster 2019) seems natural. Extensions along these lines may be implemented directly into mpitb est. Other types of analyses, however, may require one or more tools on their own, such as a panel-data-based analysis within the AF framework (for example, Alkire et al. [2017a], Suppa [2018]). Standalone tools may also be needed for the analysis of pairwise robust comparisons, which examines country orderings in terms of their poverty indices (Alkire and Santos 2014; Alkire et al. 2022a), or the recently proposed modeling framework for computing projections of multidimensional poverty (Alkire et al. Forthcoming). Aside from the implementation of genuine methodological innovations, one may also consider convenience tools, which, for instance, help to compare different measures using specific tabulations or visualizations during the trial stage. Future developments, however, depend on many factors, including user needs, further progress in research, and available resources.

6 Acknowledgements

I would like to express my sincere thanks both to Rodrigo García Arancibia and José Vargas, my mentors, for their in valuable guidance, feedback, and encouragement throughout the development of this package. I also wish to acknowledge Nicolai Suppa for his generous support and enthusiasm in adapting his Stata package to R users. Without his constructive suggestions this package would not have been possible. Finally, I would like to thank every instructor from the OPHI Summer School 2022, for their thorough lectures on the Alkire-Foster method for multidimensional poverty measurement and the provision of the Stata source code which mainly motivated this project.

References

Alkire, Sabina. 2021. "Multidimensional Poverty Measures as Policy Tools." In, edited by & R. Lepenies V. Beck H. Hahn, 197–214. Philosophy and Poverty 2. Springer.

Alkire, Sabina, and J. E. Foster. 2011. "Counting and Multidimensional Poverty Measurement." *Journal of Public Economics* 95 (7): 476–87. https://doi.org/10.1016/j.jpubeco.2010.11.006.

Alkire, Sabina, James Foster, Suman Seth, Maria Emma Santos, José Manuel Roche, and Paola Ballon. 2015. *Multidimensional Poverty Measurement and Analysis*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199689491.001.0001.

Alkire, Sabina, José Manuel Roche, and Ana Vaz. 2017. "Changes over Time in Multidimensional Poverty: Methodology and Results for 34 Countries." *World Development* 94: 232–49. https://doi.org/https://doi.org/10.1016/j.worlddev.2017.01.011.

Lumley, Thomas. 2004. "Analysis of Complex Survey Samples." *Journal of Statistical Software* 9 (8): 1–19. https://doi.org/10.18637/jss.v009.i08.

Suppa, Nicolai. 2023. "mpitb: A Toolbox for Multidimensional Poverty Indices." *The Stata Journal* 23 (3): 625–57. https://doi.org/10.1177/1536867X231195286.

UNDP, and OPHI. 2022. "2022 Global Multidimensional Poverty Index (MPI)." *UNDP (United Nations Development Programme)*.

Ignacio Girela
CONICET - Universidad Nacional de Córdoba
Facultad de Ciencias Económicas
Córdoba, Argentina
https://www.eco.unc.edu.ar/
ORCiD: 0000-0003-3297-3854
ignacio.girela@unc.edu.ar