R-Package surveysd

Johannes Gussenbauer, Alexander Kowarik, Matthias Till

2018-01-25

Motivation

- EU-SILC and at risk of social exclusion (`arose`)
- Qualitatively high well-being indicators at national or NUTS1
- Lower NUTS-Levels usually yield poor estimates

Methods

- Small area estimation
- administrative data to impute variable of interest
- pooled data and bootstrap techniques

surveysd

- R-package for variance estimation on regional levels
- Uses multiple (consecutive) waves of EU-SILC
- Variance estimation via bootstrap techniques

Methodology

- Let $\mathbf{X}_{(\mathbf{h},\mathbf{j})}$ be pooled data over $j = 1, ..., y$ years containing a sample size of $n_1, ..., n_y$
- Each year l contains a sample size n_l
- Household ID h is unique throughout the pooled data

Methodology

- Generate B bootstrap replicates without replacement and considering stratification/clustering [preston et al]
  - $f^i_{(h,j)}$ for household h in year j, $i = 1, ..., B$
- Bootstrap replicates are assigned for each household
  - stay constant until the household drops out of the sample
  - $f^i_{(h,k(h))} = f^i_{(h,j)}\ \forall j \in 1, ..., y,\ i = 1, ..., B$
  - with $k(h)$ as the year that household h first comes into SILC

Methodology

- Calculate replicate weights $b^i_{(h,j)}$ by multiplying with original weight $w^0_{(h,j)}$
  - $b^i_{(h,j)} = f^i_{(h,j)} w^0_{(h,j)}$
- Calibrate with iterative proportional updating
  - define margins of sampling design per year
- Calibrated weights are euqal in each household

Methodology

- Estimate Variance of point estimate $\theta(\mathbf{X}_j, \mathbf{w}_j^0)$ with $\mathbf{X}_j$ as observations- and $\mathbf{w}_j^0$ as weight-vector.
  - $sd(\theta) = \sqrt{\frac{1}{B-1}\sum_{i=1}^{B}(\theta(\mathbf{X}^{(j)}, \mathbf{b}^{(i,j)}) - \overline{\theta})^2}$
- Using $\overline{\theta} := \frac{1}{B}\sum_{i=1}^{B}\theta(\mathbf{X}^{(j)}, \mathbf{b}^{(i,j)})$ as the sample mean and $\mathbf{b}_j^i$ as the i-th vector of bootstrap weights for the year j.

Methodology

- Use data from consecutive years of SILC for regional estimates
- Apply filter to consecutive years using equal filter weights
  - poverty stays quite stable through out consecutive years
- Gain in precision for variance estimation via pooled data

Methodology

$$sd(\theta) = \sqrt{\frac{1}{B-1}\sum_{i=1}^{B}(\theta^{(3)}(\mathbf{X}^{(y)}, \mathbf{b}^{(i,y)})) - \overline{\theta^{(3)}})}$$

with

$$\theta^{(3)}(\mathbf{X}^{(y)}, \mathbf{b}^{(i,y)}) = \frac{1}{3}(\theta(\mathbf{X}^{(y-1)}, \mathbf{b}^{(i,y-1)}) + \theta(\mathbf{X}^{(y)}, \mathbf{b}^{(i,y)}) + \theta(\mathbf{X}^{(y+1)}, \mathbf{b}^{(i,y+1)}))$$

and

$$\overline{\theta^{(3)}} = \frac{1}{B}\sum_{i=1}^{B}\theta^{(3)}(\mathbf{X}^{(y)}, \mathbf{b}^{(i,y)}) \quad .$$

Using package surveysd

- Not yet on CRAN but on git https://github.com/statistikat/surveysd
- Contains of 3 major functions
  - `draw.bootstrap`
  - `recalib`
  - `calc.stError`

Using package surveysd

- Data must contain the following variables
    - Household Identifier
    - Sampling weights
    - Column specifing year of sample drawn
    - variables of interest
    - Columns by which sample was stratified
- Each row represents 1 Individual

Using package surveysd

```
library(data.table)
library(surveysd)
library(ggplot2)
dat <- fread("/mnt/obdatenaustausch/NETSILC3/udb_short_new.csv")
dat[,RB050:=gsub(",","\\.",RB050)]
dat[,RB050:=as.numeric(RB050)]

dat_es <- dat[RB020=="ES"]
dat_es[,.(RB010,RB030,DB040,arose,hsize,HX040,db050)]
```

```
##         RB010    RB030 DB040 arose hsize HX040 db050
##      1:  2013     10001  ES62     0     1     1  1407
##      2:  2014     10001  ES62     0     1     1  1407
##      3:  2015     10001  ES62     0     1     1  1407
##      4:  2016     10001  ES62     1     1     1  1407
##      5:  2016     40001  ES51     0     2     2   907
##     ---
## 310730:  2016 999950003  ES61     1     4     5   106
## 310731:  2015 999950004  ES61     1     4     5   106
## 310732:  2016 999950004  ES61     1     4     5   106
## 310733:  2015 999950005  ES61     1     4     5   106
## 310734:  2016 999950005  ES61     1     4     5   106
```

```
dat_es[agex==100,agex:=80]
```

Drawing bootstrap replicates

```
# draw 20 boostrap replicates with strata
dat_boot_1 <- draw.bootstrap(dat=copy(dat_es),REP=20,hid="db030",
                             weights="RB050",strata="db050",
                             year="RB010")
```

```
colnames(dat_boot_1)
```

```
##  [1] "RB010"   "RB020"   "RB030"   "db030"   "hid_new" "DB040"   "db050"
##  [8] "DB060"   "DB100"   "RB050"   "HX080"   "HX090"   "RB080"   "RB090"
## [15] "HX040"   "arose"   "agex"    "hsize"   "w1"      "w2"      "w3"
## [22] "w4"      "w5"      "w6"      "w7"      "w8"      "w9"      "w10"
## [29] "w11"     "w12"     "w13"     "w14"     "w15"     "w16"     "w17"
## [36] "w18"     "w19"     "w20"
```

Drawing bootstrap replicates

```r
# define stratified 1-Stage cluster sample
dat_boot <- draw.bootstrap(dat=copy(dat_es),REP=20,hid="db030",
                           weights="RB050",strata="db050",cluster="DB060",
                           year="RB010")
```

```
## Number of Clusters at each level are not specified
## Design is sampled with replacement
```

```r
# define Number of clusters in in each strata
# as well as number of households in each cluster
dat_es[,fpc1:=length(unique(DB060))/.05,by=list(db050,RB010)] # 5% of Clusters are drawn in each strata
dat_es[,fpc2:=sum(RB050[!duplicated(db030)]),by=list(DB060,db050,RB010)]

dat_boot <- draw.bootstrap(dat=copy(dat_es),REP=20,hid="db030",
                           weights="RB050",strata="db050",cluster="DB060",
                           year="RB010",totals=c("fpc1","fpc2"),boot.names=NULL)
```

Calibrating bootstrap replicates

```
dat_boot_calib_1 <- recalib(dat=copy(dat_boot_1),hid="db030",
                            weights="RB050",year="RB010",b.rep=paste0("w",1:20),
                            conP.var=c("RB090"),conH.var = c("DB040"))
```

```
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  2  steps
## Convergence reached in  2  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
## Convergence reached in  2  steps
## Convergence reached in  3  steps
```

Calibrating bootstrap replicates

```
dat_boot_calib <- recalib(dat=copy(dat_boot_1),hid="db030",
                  weights="RB050",year="RB010",b.rep=paste0("w",1:20),
                  conP.var=c("RB090","agex"),conH.var = c("DB040","DB100"))
```

```
## Convergence reached in  4  steps
## Convergence reached in  6  steps
## Convergence reached in  3  steps
## Convergence reached in  5  steps
## Convergence reached in  4  steps
## Convergence reached in  5  steps
## Convergence reached in  6  steps
## Convergence reached in  4  steps
## Convergence reached in  5  steps
## Convergence reached in  5  steps
## Convergence reached in  5  steps
## Convergence reached in  5  steps
## Convergence reached in  6  steps
## Convergence reached in  3  steps
## Convergence reached in  4  steps
## Convergence reached in  4  steps
## Convergence reached in  3  steps
## Convergence reached in  5  steps
## Convergence reached in  3  steps
## Convergence reached in  3  steps
```

Calculate Estimates using bootstrap replicates

- Calculate variance or distribution of point estimate $\theta(\mathbf{X}, \mathbf{w})$ using bootstrap and original weights
- Predefined point estimates
  - `weightedRatio` - `weightedRatioNat`
  - `weightedSum`
  - `popSize` - `sampSize`

Calculate Estimates using bootstrap replicates

- Point estimates are applied on specified variables using the original and bootstrap weights
- Estimates are calcualted per `year`, but additional subgroups can be defined $\rightarrow$ regional estimates

Calculate Estimates using bootstrap replicates

- Differences of point estimates between years and rolling means over consecutive years can also be applied
- Differences for rolling means is also supported
  - $\theta^{(2014-2016)}(\mathbf{X},\mathbf{b}) - \theta^{(2008-2010)}(\mathbf{X},\mathbf{b})$

Calculate Estimates using bootstrap replicates

```
erg <- calc.stError(dat=copy(dat_boot_calib_1),fun="weightedRatio",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=NULL,year.diff=NULL,year.mean=NULL)
erg
```

```
## Calculated point estimates for variable(s)
##
##  HX080
##
## using function weightedRatio
##
## Results hold 9 point estimates for 9 years
```

```
erg$Estimates
```

```
##    RB010 val_HX080 stE_HX080
## 1:  2008  19.83106 0.6665232
## 2:  2009  20.36623 0.5915289
## 3:  2010  20.72276 0.5618657
## 4:  2011  20.64952 0.4714167
## 5:  2012  20.83131 0.4894204
## 6:  2013  20.37943 0.6754405
## 7:  2014  22.22518 0.5520584
## 8:  2015  22.13294 0.4333467
## 9:  2016  22.34648 0.4857942
```

Calculate Estimates using bootstrap replicates

```r
# Estimate mean over 3 consecutive years (-> default)
erg <- calc.stError(dat=copy(dat_boot_calib_1),fun="weightedRatio",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=NULL,year.diff=NULL,year.mean=3)

erg
```

```
## Calculated point estimates for variable(s)
##
##  HX080
##
## using function weightedRatio
##
## Results hold 16 point estimates for 9 years
```

Calculate Estimates using bootstrap replicates

```
erg$Estimates
```

```
##              RB010 val_HX080 stE_HX080
##  1:           2008  19.83106 0.6665232
##  2: 2008_2009_2010  20.30668 0.5062914
##  3:           2009  20.36623 0.5915289
##  4: 2009_2010_2011  20.57950 0.4346648
##  5:           2010  20.72276 0.5618657
##  6: 2010_2011_2012  20.73453 0.3705039
##  7:           2011  20.64952 0.4714167
##  8: 2011_2012_2013  20.62009 0.4076710
##  9:           2012  20.83131 0.4894204
## 10: 2012_2013_2014  21.14531 0.4912443
## 11:           2013  20.37943 0.6754405
## 12: 2013_2014_2015  21.57918 0.4384756
## 13:           2014  22.22518 0.5520584
## 14: 2014_2015_2016  22.23487 0.2955162
## 15:           2015  22.13294 0.4333467
## 16:           2016  22.34648 0.4857942
```

Calculate Estimates using bootstrap replicates

```r
# define subgroups
# Estimates per DB040 AND DB100 are also calculated
erg <- calc.stError(dat=copy(dat_boot_calib_1),fun="weightedRatio",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=list("DB100",c("agex","DB100")),year.diff=NULL)

erg
```

```
## Calculated point estimates for variable(s)
##
##  HX080
##
## using function weightedRatio
##
## Results hold 304 point estimates for 9 years in 19 subgroups
##
## Estimted standard error exceeds 10 % of the the point estimate in 29 cases
```

Calculate Estimates using bootstrap replicates

```r
erg$cvHigh
```

```
##      RB010 DB100 agex HX080
##   1:  2008     1    0 FALSE
##   2:  2008     1   20 FALSE
##   3:  2008     1   40 FALSE
##   4:  2008     1   60 FALSE
##   5:  2008     1   80  TRUE
##  ---
## 300:  2016     3   40 FALSE
## 301:  2016     3   60 FALSE
## 302:  2016     3   80 FALSE
## 303:  2016     3   NA FALSE
## 304:  2016    NA   NA FALSE
```

Calculate Estimates using bootstrap replicates

```
# Add estimation of differences between the years 2016 and 2008
erg <- calc.stError(dat=copy(dat_boot_calib_1),fun="weightedRatio",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=list("DB100",c("agex","DB100")),
                    year.diff=c("2016-2008"))
```

```
## Can not calcualte differences between years 2016 and 2008 over 3 years.
```

```
erg
```

```
## Calculated point estimates for variable(s)
##
##  HX080
##
## using function weightedRatio
##
## Results hold 323 point estimates for 9 years in 19 subgroups
##
## Estimted standard error exceeds 10 % of the the point estimate in 40 cases
```

Calculate Estimates using bootstrap replicates

```
# Calculate not only standard Error
# but also .025 and .975 percentile
erg <- calc.stError(dat=copy(dat_boot_calib_1),fun="weightedRatio",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=list("DB100",c("agex","DB100")),
                    year.diff=c("2016-2008"),p=c(.025,.975))
```

```
## Can not calcualte differences between years 2016 and 2008 over 3 years.
```

```
erg$Estimates
```

```
##          RB010 DB100 agex  val_HX080 stE_HX080 p0.025_HX080 p0.975_HX080
##   1:      2008     1    0  25.171215 1.6216158   22.4147958    27.776771
##   2:      2008     1   20  14.587201 1.0712621   13.1224750    16.688469
##   3:      2008     1   40  14.084083 0.9294461   13.1391893    15.750170
##   4:      2008     1   60  15.825289 1.0585613   13.4570976    17.237957
##   5:      2008     1   80  22.647548 2.2770126   18.7320050    26.299361
##  ---
## 319: 2016-2008     3   40   9.901689 1.8317388    6.6112064    12.886983
## 320: 2016-2008     3   60  -9.457798 1.8286126  -12.3303206    -5.903782
## 321: 2016-2008     3   80 -17.454288 2.8370711  -22.0215811   -11.842494
## 322: 2016-2008     3   NA   4.708759 1.3404008    2.8822519     6.994277
## 323: 2016-2008    NA   NA   2.515426 0.9037091    0.8507179     3.760437
```

Calculate Estimates using bootstrap replicates

```r
# user-defined function
# take the gini - index
library(laeken)
# simulate income
dat_income <- unique(dat_boot_calib_1,by="db030")
dat_income[,income:=exp(rnorm(.N,mean=10,sd=0.8))-1]

# gini() returns list - calc.stError needs function that returns double or integer
help_gini <- function(x,w){
  return(gini(x,w)$value)
}

erg <- calc.stError(dat=copy(dat_income),fun="help_gini",
                    weights="RB050",year="RB010",b.weights=paste0("w",1:20),
                    var="HX080",cross_var=list(c("DB040","DB100")),
                    year.diff=c("2014-2008"),p=c(.025,.975))
```

```
## For grouping by RB010~DB040~DB100:
## Sample size lower 20 for 34 groups
```

Calculate Estimates using bootstrap replicates

```r
erg
```

```
## Calculated point estimates for variable(s)
##
##  HX080
##
## using function help_gini from .GlobalEnv
##
## Results hold 1044 point estimates for 9 years in 58 subgroups
##
## 28 subgroups contained less than 20 observations
##
## Point estimates are NAs in 152 cases due to either
##  no observations or only NAs for the variable(s) in the corresponding subgroups.
##
## Estimted standard error exceeds 10 % of the the point estimate in 189 cases
```

```r
head(erg$smallGroups)
```

```
##    RB010 DB040 DB100  N
## 1:  2010  ES30     3 17
## 2:  2010  ES41     2 11
## 3:  2011  ES13     2  8
## 4:  2011  ES21     3 16
## 5:  2011  ES30     2  9
## 6:  2011  ES41     2 15
```

Calculate Estimates using bootstrap replicates

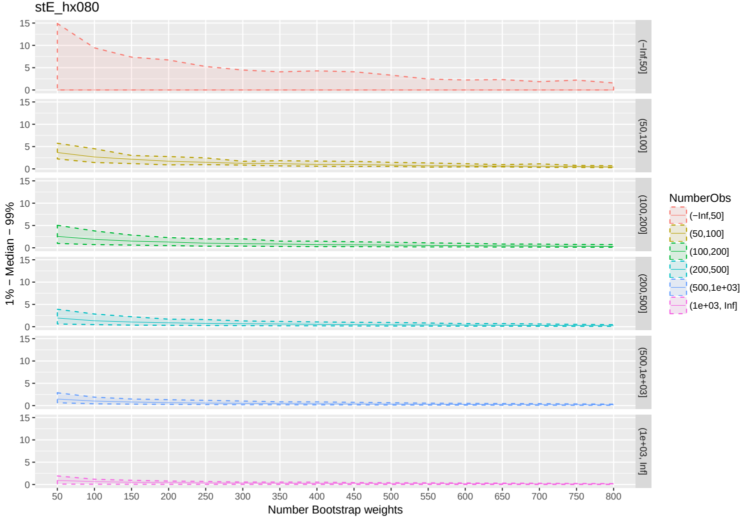
```
erg$cvHigh[HX080==TRUE]
```

```
##                RB010 DB040 DB100 HX080
##   1:            2008  ES22     2  TRUE
##   2:            2008  ES30     2  TRUE
##   3:            2008  ES63     1  TRUE
##   4: 2008_2009_2010  ES53     2  TRUE
##   5:            2009  ES12     3  TRUE
##  ---
## 185:            2016  ES62     2  TRUE
## 186:            2016  ES63     1  TRUE
## 187:            2016  ES64     1  TRUE
## 188:            2016  ES70     1  TRUE
## 189:            2016  ES70     3  TRUE
```

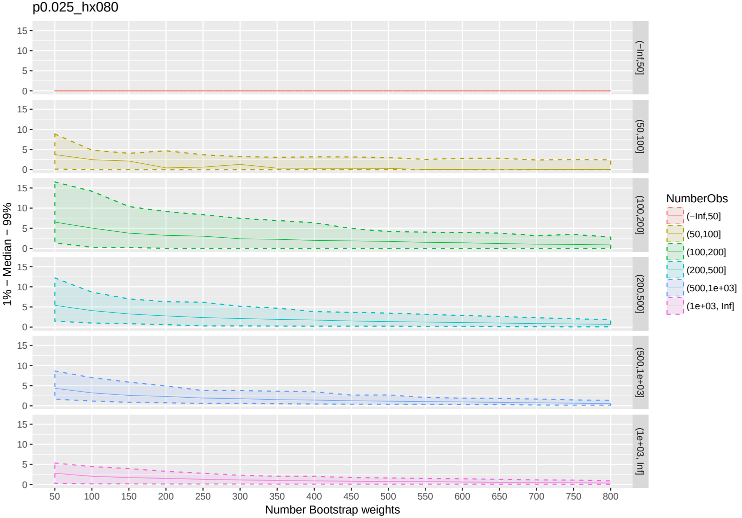
Suggested Number of bootstrap replicates

- Current Population Survey (https://cps.ipums.org/cps/) suggest 150 replicates
    - use Jackknife resampling technique
- Depending on size of subgroup different number of bootstrap replicates might be usefull
- In general we suggest 250 bootstrap replicates

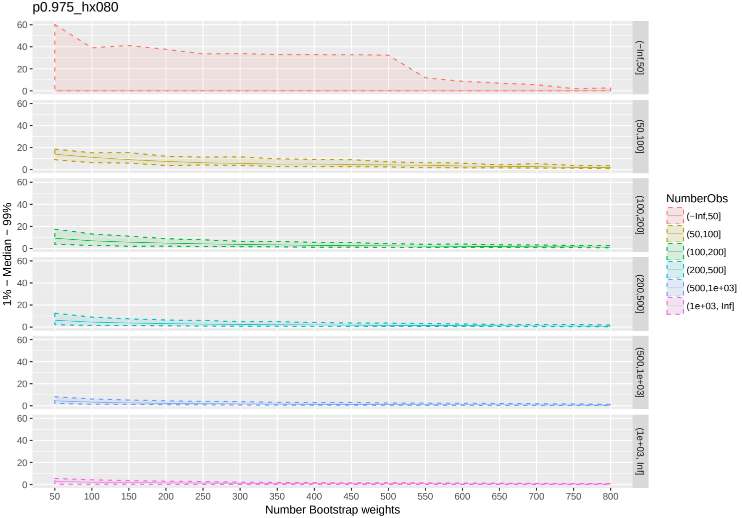
Suggested Number of bootstrap replicates



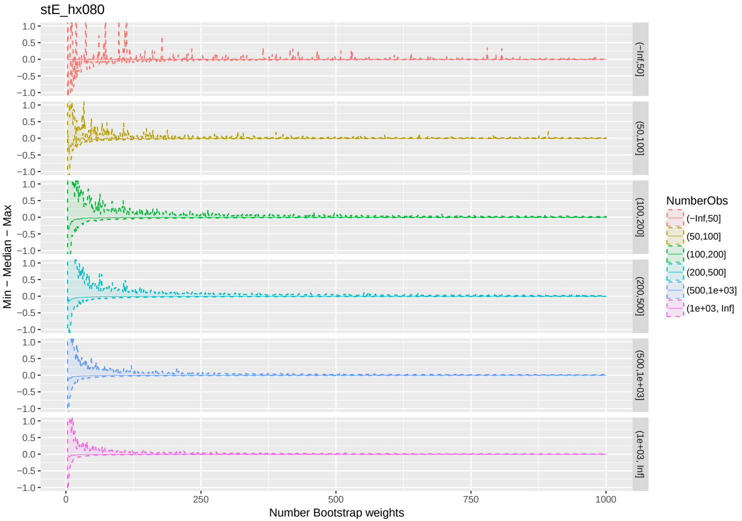
Suggested Number of bootstrap replicates



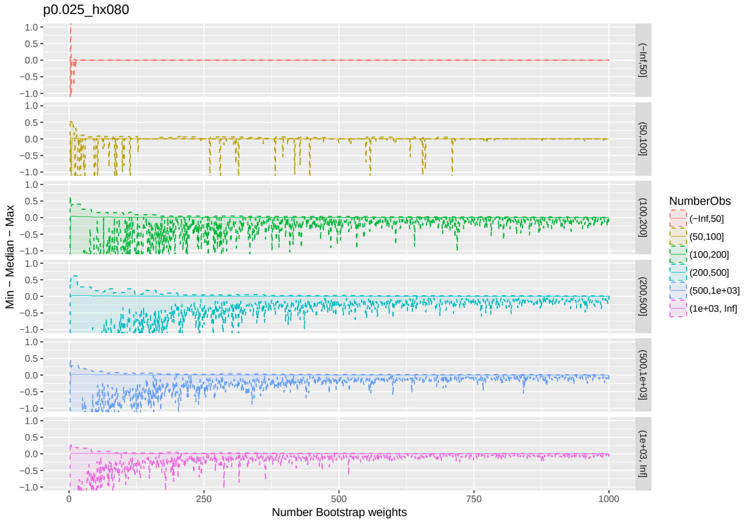
Suggested Number of bootstrap replicates



Suggested Number of bootstrap replicates



Suggested Number of bootstrap replicates



Suggested Number of bootstrap replicates