The Univariate Fay-Herriot (UFH) model

Sylvia Harmening

Introduction

In this section, we will present a whole estimation procedure of the univariate area-level model introduced by Fay and Herriot (1979) in R. As with the disclaimer in the preceding section, this practical manual is not intended to serve as a theoretical introduction to area-level models. Instead, it offers a set of straightforward and practical R scripts, accompanied by clear explanations, to demonstrate how these tasks can be carried out in R. For a theoretical foundation please refer to Fay and Herriot (1979) and Rao and Molina (2015). In addition to theoretical information, the vignette "A framework for producing small area estimates based on area-level models in R" of the R package `emdi` (Harmening et al. 2023) provides further code examples for the FH model.

In this chapter, we will describe how to run the univariate Fay-Herriot (FH) using simulated income data from Spain. The estimation procedure is explained step by step.

Step 1: Data preparation. Compute the direct estimates and their corresponding variances on the area-level and eventually perform variance smoothing. Aggregate the available auxiliary variables to the same area level and combine both input data.

Step 2: Model selection. Select the aggregated auxiliary variables at the area level for the FH model using a stepwise selection based on information criteria like the Akaike, Bayesian or Kullback information criteria.

Step 3: Model estimation of FH point estimates and their mean squared error (MSE) estimates as uncertainty measure. Eventually apply a transformation.

Step 4: Assessment of the estimated model. Check the FH model assumptions, including linearity, normality of predicted area effects and standardized model residuals. When violations of the model assumptions are detected, the application of a transformation might help. Repeat Step 3 including a transformation and check again the model assumptions.

Step 5: Comparison of the FH results with the direct estimates.

Step 6: Benchmark the FH point estimates for consistency with higher results.

Step 7: Preparation of the results. Create tables and maps of results.

Step 8: Saving the results. One option is to export the results to known formats like Excel or OpenDocument Spreadsheets.

We will show below the use of the `fh()` function of the R package `emdi` (Harmening et al. 2023) which computes the EBLUPs and their MSE estimates of the standard FH model and several extensions of it, among others it allows for the application of transformations. Because the poverty rate is a ratio, it might be helpful to apply the arcsin transformation to guarantee that the results lie between 0 and 1. The function call is:

```
fh(fixed, vardir, combined_data, domains = NULL, method = "reml", interval =
NULL,   k = 1.345, mult_constant = 1, transformation = "no", backtransformation
= NULL,   eff_smpsize = NULL, correlation = "no", corMatrix = NULL, Ci =
NULL, tol = 1e-04,   maxit = 100, MSE = FALSE, mse_type = "analytical", B
= c(50, 0), seed = 123)
```

Data and preparation

Load the dataset

Usually, SAE combines multiple data sources: a survey data set and a census or administrative/register dataset. For the estimation of area-level models, we need area-level aggregates of the same area-level (e.g. NUTS3) of both datasets. The target variable (typically household welfare/income for poverty mapping) is available in the survey but not in the census data.

In this example, we use a synthetic data set adapted from R package `sae` called `incomedata`. The original data contains information for $n = 17,119$ fictitious individuals residing across $D = 52$ Spanish provinces. The variables include the name of the province of residence (`provlab`), province code (`prov`), as well as several correlates of income.

For this exercise, we use a random 10% sample of the `incomedata` to estimate the poverty rates. For the univariate case, we use the income variable from 2012.

The FH estimation process relies on 3 types of data:

(1) The data containing the outcome variable from which direct estimates will be computed. This is a usually a survey dataset for each year of data including outcome variable (such as income or welfare aggregates), weights, cluster identifiers (i.e. if available, psu or enumeration areas) at the unit level i.e. individual or household. In this example, we call this: `survey_dt`.

(2) A dataset containing for the right hand side (RHS) variables i.e. indicator estimates representative at the level of the target area. This is often obtained from administrative data sources or geospatial data such as remotely sensed high resolution data. The final RHS dataset ought to include an area id and variables of interest. In this example, we call this `rhs_dt`.

(3) Finally, a shapefile which spatially links each target area to their boundaries on a map. This will contain the area id as well as the geometry object which when visualized will show the shape of the areas in which poverty rates will be estimated. In this example, we call this `shp_dt`.

```
### lets read in our 3 aforementioned datasets

survey_dt <-
  bd_clean |>
  pin_read("pov_direct")


rhs_dt <-
  bd_clean |>
  pin_read("sae_data")


shp_dt <-
  bd_clean |>
  pin_read("geometries")
```

Below is what our datasets look like. These are generally the variables

```
survey_dt |>
  glimpse()
```

```
Rows: 5,148
Columns: 7
$ provlab <fct> Alava, Alava, Alava, Alava, Alava, Alava, Alava, Alava, Alava,~
$ prov    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
$ weight  <dbl> 19471.566, 19471.566, 19471.566, 7424.759, 23577.245, 7424.759~
$ ea_id   <int> 102, 102, 102, 103, 103, 103, 103, 103, 103, 104, 104, 104, 10~
$ year    <int> 2012, 2013, 2014, 2012, 2012, 2013, 2013, 2014, 2014, 2012, 20~
$ income  <dbl> 27920.737, 28508.978, 31512.153, 26125.271, 27404.099, 27500.6~
$ povline <dbl> 6477.484, 6515.865, 6515.865, 6477.484, 6477.484, 6515.865, 65~
```

`survey_dt` contains our target area identifiers (`provlab`, `prov`), the weight variable `weight`, the cluster id i.e. enumeration area or psu, `ea_id`, the year, `year`, the income variable `income` and the poverty line `poverty`. In this example, `survey_dt` of class `data.frame` is at the level of the individual. It could also be at the level of the household if poverty lines are evaluated at that level.

```r
rhs_dt |>
  glimpse()
```

```
Rows: 156
Columns: 39
Groups: prov [52]
$ prov          <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ~
$ provlab       <fct> Alava, Albacete, Alicante, Almeria, Avila, Badajoz, Bal~
$ gen           <dbl> 1.542687, 1.293462, 1.552217, 1.456691, 1.240551, 1.243~
$ age2          <dbl> 0.12000039, 0.09008770, 0.19188855, 0.08809684, 0.00000~
$ age3          <dbl> 0.1568951, 0.5603895, 0.4129128, 0.4755861, 0.7215736, ~
$ age4          <dbl> 0.37497939, 0.08696019, 0.14055936, 0.11591406, 0.00000~
$ age5          <dbl> 0.26323453, 0.09024103, 0.09659673, 0.07212035, 0.27842~
$ educ1         <dbl> 0.26323453, 0.34052375, 0.17766183, 0.23213632, 0.41639~
$ educ2         <dbl> 0.00000000, 0.24339173, 0.54735236, 0.27227693, 0.58360~
$ educ3         <dbl> 0.42790749, 0.00000000, 0.11066865, 0.24730411, 0.00000~
$ nat1          <dbl> 0.9151094, 0.9648872, 0.8811369, 0.9492913, 1.0000000, ~
$ labor1        <dbl> 0.3658591, 0.3161702, 0.5388899, 0.5170435, 0.3430585, ~
$ labor2        <dbl> 0.00000000, 0.00000000, 0.02956489, 0.00000000, 0.00000~
$ labor3        <dbl> 0.32528288, 0.26774526, 0.26722804, 0.23467384, 0.65694~
$ abs           <dbl> 0.84933922, -0.97816135, 0.98863150, 0.31350535, -1.419~
$ ntl           <dbl> 0.2091905, 1.1589425, -0.4210840, 1.5472493, -2.1440414~
$ aec           <dbl> 1.16518757, -0.55990121, -0.66408247, -0.70557145, 4.99~
$ schyrs        <dbl> 1.70993214, -0.96018920, -0.18004563, 0.22504016, -1.64~
$ mkt           <dbl> 2.5782541, -1.0468569, -0.2044435, -0.9811182, -1.08118~
$ age2_X_gen    <dbl> 0.24000077, 0.13709455, 0.33728510, 0.14177034, 0.00000~
$ age3_X_gen    <dbl> 0.2189435, 0.7247252, 0.6071718, 0.6081940, 0.9621249, ~
$ age4_X_gen    <dbl> 0.53114607, 0.08696019, 0.19271194, 0.13736688, 0.00000~
$ age5_X_gen    <dbl> 0.46770599, 0.12535380, 0.16562463, 0.11584671, 0.27842~
$ educ1_X_gen   <dbl> 0.46770599, 0.46803355, 0.29892028, 0.30800487, 0.41639~
$ educ2_X_gen   <dbl> 0.00000000, 0.31533038, 0.87554713, 0.36826488, 0.82416~
$ educ3_X_gen   <dbl> 0.64612252, 0.00000000, 0.12205150, 0.32690817, 0.00000~
$ nat1_X_gen    <dbl> 1.457796, 1.223237, 1.369321, 1.405982, 1.240551, 1.243~
$ labor1_X_gen  <dbl> 0.5220258, 0.3881089, 0.7454287, 0.5966476, 0.3430585, ~
$ labor2_X_gen  <dbl> 0.00000000, 0.00000000, 0.04420301, 0.00000000, 0.00000~
$ labor3_X_gen  <dbl> 0.5918027, 0.3952551, 0.5068872, 0.4065303, 0.8974928, ~
$ age2_X_educ3  <dbl> 0.00000000, 0.00000000, 0.01138285, 0.00000000, 0.00000~
$ age3_X_educ3  <dbl> 0.15689512, 0.00000000, 0.03971356, 0.22585130, 0.00000~
$ age4_X_educ3  <dbl> 0.27101237, 0.00000000, 0.05957224, 0.02145281, 0.00000~
$ age5_X_educ3  <dbl> 0.00000000, 0.00000000, 0.00000000, 0.00000000, 0.00000~
$ nat1_X_educ3  <dbl> 0.42790749, 0.00000000, 0.07095509, 0.19659543, 0.00000~
$ labor1_X_educ3 <dbl> 0.36585913, 0.00000000, 0.11066865, 0.24730411, 0.00000~
```

Data Input Checklist for the Univariate Fay-Herriot Model
Datasets, levels, and Required variables

Dataset Name	Unit of Observation	Required Variables
`survey_dt`	Individual (or Household)	`target area identifiers`, `weights`, `cluster identifier`, `income/welfare varia`
`rhs_dt`	Target Area (e.g. Province)	`target area identifiers`, `covariates` (e.g. `gen`, `educ1`, `schyrs`, etc.)
`shp_dt`	Target Area (Spatial)	`target area identifiers`, `geometries` (e.g. `geometry` column from `sf`)

```
$ labor2_X_educ3 <dbl> 0.00000000, 0.00000000, 0.00000000, 0.00000000, 0.00000~
$ labor3_X_educ3 <dbl> 0.06204835, 0.00000000, 0.00000000, 0.00000000, 0.00000~
$ year           <int> 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2~
```

‘`rhs_dt` is an object of class `data.frame` created at the level of the target area. It should contain the same target area identifiers as in `survey_dt` and the year variable `year`.

```
shp_dt |>
  glimpse()
```

```
Rows: 52
Columns: 3
$ prov     <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
$ provlab  <fct> Alava, Albacete, Alicante, Almeria, Avila, Badajoz, Baleares,~
$ geometry <MULTIPOLYGON [°]> MULTIPOLYGON (((-2.858067 4..., MULTIPOLYGON (((~
```

`shp_dt` is an object of class `sf`, `data.frame` created at the level of the target area. This is the shapefile for the area of interest for which the poverty map will be estimated. It should contain the same target area ID found in `survey_dt` as well as `rhs_dt`.

A quick summary table on the data needs for the UFH model:

Direct estimation

We will use the direct Horvitz-Thompons estimators that use the survey weights in `weight` variable. We calculate the sample sizes for each provinces and compute the direct estimates and their variances. We use the `direct` function of the `emdi` package here. Other options are e.g. the `direct` function of package `sae` (Molina and Marhuenda 2015) or the `svyby` command of package `survey` (Lumley 2024). Then, we create a dataframe containing the direct estimate, the standard errors, the variances, the coefficient of variation and the design effects, that are needed for the arcsin transformation. The design effect is the ratio of the variance considering the sampling design to the variance estimated under simple random sampling. For detailled information about the arcsin transformation please refer to Casas-Cordero, Encina, and Lahiri

(2016) and Schmid et al. (2017). In some areas with a very small sample size, it may occur, that the individual data only consists of zeros and ones, resulting in a direct estimate of zero or one and a direct variance of zero. We set those areas to out-of-sample and for the final estimation results only the synthetic part of the FH model is used.

```
### a little bit of housekeeping to ensure ease of access
area_vars <- c("prov", "provlab") ### both variables are at the same level. If the levels va

cluster_var <- "ea_id"

weight_var <- "weight"

year_var <- "year"

outcome_var <- "income"

povline_var <- "povline"

candidate_vars <- colnames(rhs_dt)[!colnames(rhs_dt) %in% c(area_vars, year_var)]

## For the univariate model, we only use a single year (here 2012)
singleyear <- "2012"

survey_dt <- survey_dt |>
  filter(year == singleyear)

rhs_dt <- rhs_dt |>
  filter(year == singleyear)


## calculate sample size for each province
sampsize_dt <-
survey_dt |>
  group_by(!!!syms(area_vars[1])) |>
  summarize(N = n())

## computation of direct estimates and their variances (the poverty line is already included
## creating the poverty indicator
survey_dt <-
  survey_dt |>
  mutate(pov_indicator = ifelse(income < povline, 1, 0))
```

```r
### creating a survey object
design_obj <- survey::svydesign(ids = eval(expr(~!!sym(cluster_var))),
                                weights = eval(expr(~!!sym(weight_var))),
                                data = survey_dt)

var_dt <- survey::svyby(~pov_indicator, by=eval(expr(~!!sym(area_vars[1]))), design = design_


direct_dt <-
  var_dt |>
  rename(direct_povrate = "pov_indicator") |>
  rename(SD = "se") |>
  mutate(vardir = SD^2) |>
  mutate(CV = SD / direct_povrate) |>
  merge(sampsize_dt,
        by = area_vars[[1]]) |>
  mutate(var_SRS = direct_povrate * (1 - direct_povrate) / N) |>
  mutate(deff = vardir / var_SRS) |>
  mutate(n_eff = N/deff)

## set zero variance to OOS
direct_dt <- direct_dt[complete.cases(direct_dt), ]
```

Here is what the results look like:

```r
head(direct_dt)
```

```
  prov direct_povrate         SD      vardir        CV  N     var_SRS      deff
1    1     0.30885799 0.13648649 0.018628563 0.4419070 10 0.021346473 0.8726764
2    2     0.05339445 0.04756853 0.002262765 0.8908891 17 0.002973146 0.7610677
3    3     0.19412452 0.07386414 0.005455911 0.3804988 54 0.002897041 1.8832705
4    4     0.42100526 0.14325415 0.020521753 0.3402669 20 0.012187992 1.6837682
6    6     0.15381464 0.08031553 0.006450585 0.5221579 49 0.002656239 2.4284658
7    7     0.13697936 0.05803163 0.003367670 0.4236523 63 0.001876445 1.7947077
     n_eff
1 11.45900
2 22.33704
3 28.67352
4 11.87812
6 20.17735
7 35.10321
```

Variance smoothing

A quick inspection of the preceding results will show some provinces contain low sample sizes which sometimes result in extreme value poverty rates and hence 0 variance. To avoid this, we will show you how to apply the variance smoothing method suggested by You and Hidiroglou (2023). Please see the code and Roxygen comments below explaining the use of the `varsmoothie_king()` function which computes smoothed variances. In case, the arcsin transformation will be applied, the variance smoothing described here is not necessary, since the arcsin transformation works variance stabilizing itself. When applying the arcsin transformation, the direct variances are automatically set to 1/(4*effective sampling size) when using the `fh` function of package `emdi`. The effective sample size equals the sample size of each area divided by the design effect. If the variance stabilizing effect is not enough, the design effect of a higher area level could also be used here (in this example the regions `ac`).

The goal now is to use the `varsmoothie_king()` function to add an additional column of smoothed variances into our `direct_dt` dataframe. Required inputs: a vector of unique domains, the raw variances estimated from sample data and the sample size for each domain.

```
var_smooth <- varsmoothie_king(domain = direct_dt[[area_vars[1]]],
                               direct_var = direct_dt$vardir,
                               sampsize = direct_dt$N)

direct_dt <- var_smooth |> merge(direct_dt, by.x = "Domain",
        by.y = area_vars[1])

# Replace the variances that are zero with their smoothed counterparts
direct_dt <-
  direct_dt |>
  mutate(across(
    starts_with("v_"),
    ~ if_else(abs(.x) <= 1e-4, get(paste0("vsv", str_remove(cur_column(), "^v"))), .x),
    .names = "{.col}"
  ))
```

Thus far, we have careful set up the types of data we require for the FH model. We only need to combine the dataframe containing the direct estimates and their variances with the auxiliary variables.

```
fh_dt <- merge(direct_dt, rhs_dt,
    by.x = "Domain", by.y = area_vars[[1]],
    all = TRUE)
```

Model selection

Model preparation

FH does not run if there is any missing value in the auxiliary variables, and therefore, any variable with missing value should be removed in advance.

```r
rowsNAcovariates <- rowSums(sapply(fh_dt[,..candidate_vars], is.na))
fh_dt <- fh_dt[rowsNAcovariates == 0, ]
```

Check multicollinearity

With the help of the `step()` function of package **emdi**, we perform a variable selection based on the chosen variable selection criterion and directly get the model with fewer variables. The function `step_wrapper()` implemented below is a wrapper to the `emdi::step()` function and performs all the perfunctory cleaning necessary to use `step()`. This includes dropping columns that are entirely missing (`NA`) and keep only complete cases/observations (for the model selection only the in-sample domains are used) and remove perfectly or near collinear variables and combinations.

We apply the function to select the variables. Required inputs: data set, character vector containing the set of auxiliary variables, name of y variable, a correlation threshold between 0 and 1, name of information criterion and name of direct variance variable. In case, a transformation should be applied, "arcsin" as transformation and name of variable that contains the effective sample size.

```r
fh_step <- step_wrapper_fh(dt = fh_dt,
                           xvars = candidate_vars,
                           y = "direct_povrate",
                           cor_thresh = 0.7,
                           criteria = "BIC",
                           vardir = "vardir",
                           transformation = "arcsin",
                           eff_smpsize = "n_eff")
```

```r
# Resulting model formula
print(fh_step$fixed)
```

```
direct_povrate ~ educ2 + ntl + schyrs
<environment: 0x000001874c004b38>
```

Model estimation of FH point and their MSE estimates.

In this example, we use the function `fh` to calculate the FH estimates. Because we want to estimate a ratio, we need to apply the arcsin transformation to guarantee that the results lie between 0 and 1. For that, we choose "arcsin" as `transformation`, and a bias-corrected `backtransformation` ("bc"). Additionally, the effective sample size, which equals the sample size of each area divided by the design effect, is needed for the arcsin transformation. We set the `MSE` estimation to `TRUE`, the `mse_type` to "boot" (necessary for the type of transformation) and determine the number of bootstrap iterations. For practical applications, values larger than 200 are recommended. In case, no transformation is desired, the `transformation` argument must be set to "no" and the inputs `backtransformation` and `eff_smpsize` are no longer needed.

```r
set.seed(123)
fh_model <- fh(fixed = formula(fh_step$fixed),
               vardir = "vardir",
               combined_data = fh_dt,
               domains = "Domain",
               method = "ml",
               transformation = "arcsin",
               backtransformation = "bc",
               eff_smpsize = "n_eff",
               MSE = TRUE,
               mse_type = "boot", B = c(200, 0))

## In case, no transformation is desired, the call would like this:
# fh_model <- fh(
#   fixed = Direct ~ age2 + age3 + age4 + age5 + educ1 + ntl + schyrs, #formula(fh_step$fixe
#   vardir = "vardir", combined_data = comb_Data, domains = "Domain",
#   method = "ml", MSE = TRUE)
```

Assessment of the estimated model.

With the help of the `summary` method of `emdi`, we gain detailed insights into the data and model components. It includes information on the estimation methods used, the number of domains, the log-likelihood, and information criteria as proposed by Marhuenda, Morales, and Camen Pardo (2014). It also reports the adjusted R^2 from a standard linear model and the adjusted R^2 specific to FH models, as introduced by Lahiri and Suntornchost (2015). It also offers diagnostic measures to assess model assumptions regarding the standardized realized residuals and random effects. These include skewness and kurtosis (based on the `moments` package by Komsta and Novomestky (2015)), as well as Shapiro-Wilk test statistics and corresponding p-values to evaluate the normality of both error components.

```r
summary(fh_model)
```

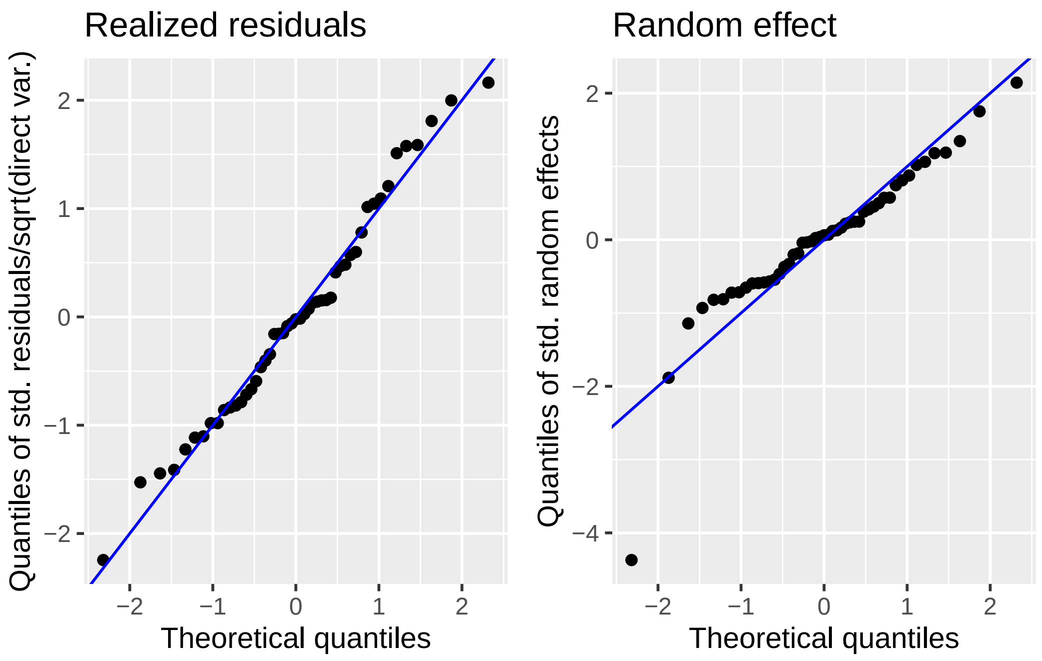
```
Call:
 fh(fixed = formula(fh_step$fixed), vardir = "vardir", combined_data = fh_dt,
    domains = "Domain", method = "ml", transformation = "arcsin",
    backtransformation = "bc", eff_smpsize = "n_eff", MSE = TRUE,
    mse_type = "boot", B = c(200, 0))

Out-of-sample domains:  3
In-sample domains:  49

Variance and MSE estimation:
Variance estimation method:  ml
Estimated variance component(s):  0.001516632
MSE method:  bootstrap

Coefficients:
            coefficients std.error t.value   p.value
(Intercept)     0.574213  0.043456 13.2135 < 2.2e-16 ***
educ2          -0.312942  0.109379 -2.8611  0.004222 **
ntl            -0.036281  0.015525 -2.3369  0.019446 *
schyrs          0.097804  0.015779  6.1982 5.712e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Explanatory measures:
   loglike       AIC       BIC     AdjR2     FH_R2
1 43.24663 -76.49325 -67.03415 0.4438399 0.7338144

Residual diagnostics:
                         Skewness Kurtosis Shapiro_W    Shapiro_p
Standardized_Residuals  0.2379309 2.587363 0.9817356 0.6397095022
Random_effects         -1.4751927 9.023352 0.8850521 0.0001849586

Transformation:
 Transformation Back_transformation
         arcsin                  bc
```

We can see, that 49 domains are in-sample domains. The 3 out-of-sample domains belong to the domains with 0 direct and variance estimates that we set to `NA` in the beginning. The variance of the random effects equals 0.001516632. All of the included auxiliary variables are significant on a 0.05 significance level and their explanatory power is large with an adjusted R^2 (for FH models) of around 0.74. The results of the Shapiro-Wilk-test indicate that normality is not rejected for the standardized residuals.

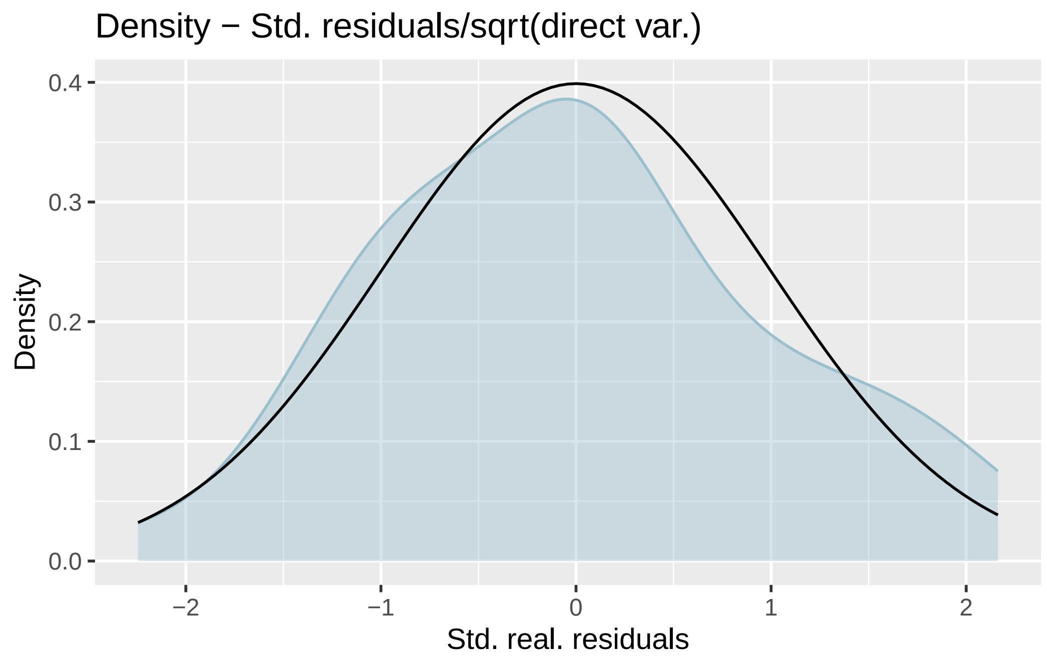
Diagnostic plots

We produce normal quantile-quantile (Q-Q) plots of the standardized realized residuals and random effects and plots of the kernel densities of the distribution of both error terms by the `plot` method of `emdi`.

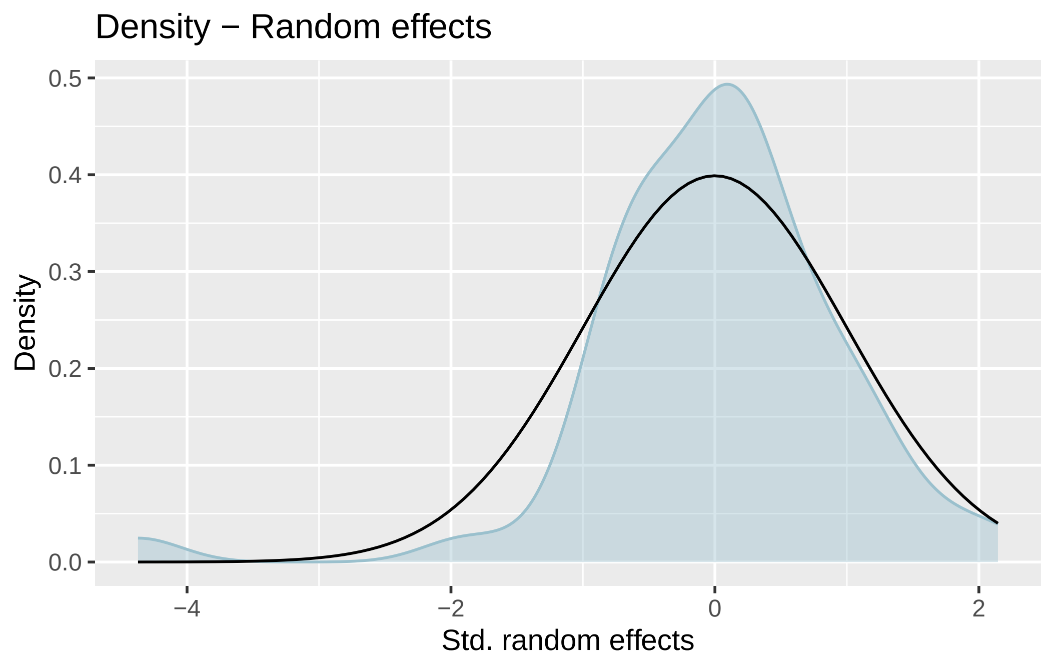
```
plot(fh_model)
```



```
Press [enter] to continue
```



```
Press [enter] to continue
```



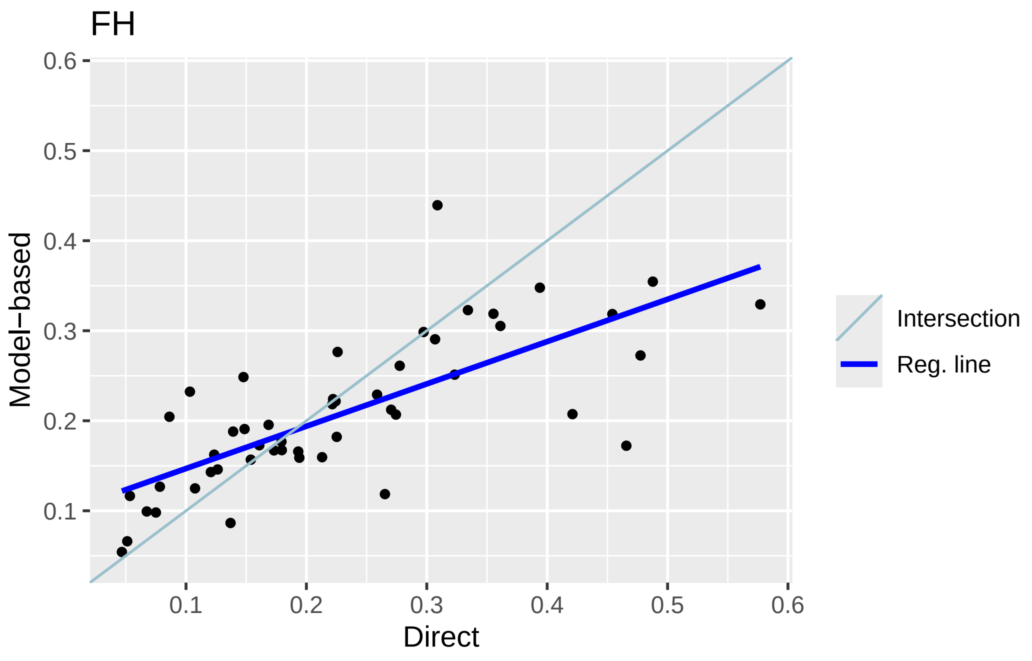
The plots show slight deviations of the distributions from normality. The normality assumption is not required for the computation of the FH estimates, but at the obtained MSE estimates we have to look with some care when the normality assumption does not hold.

Comparison of the FH results with the direct estimates.

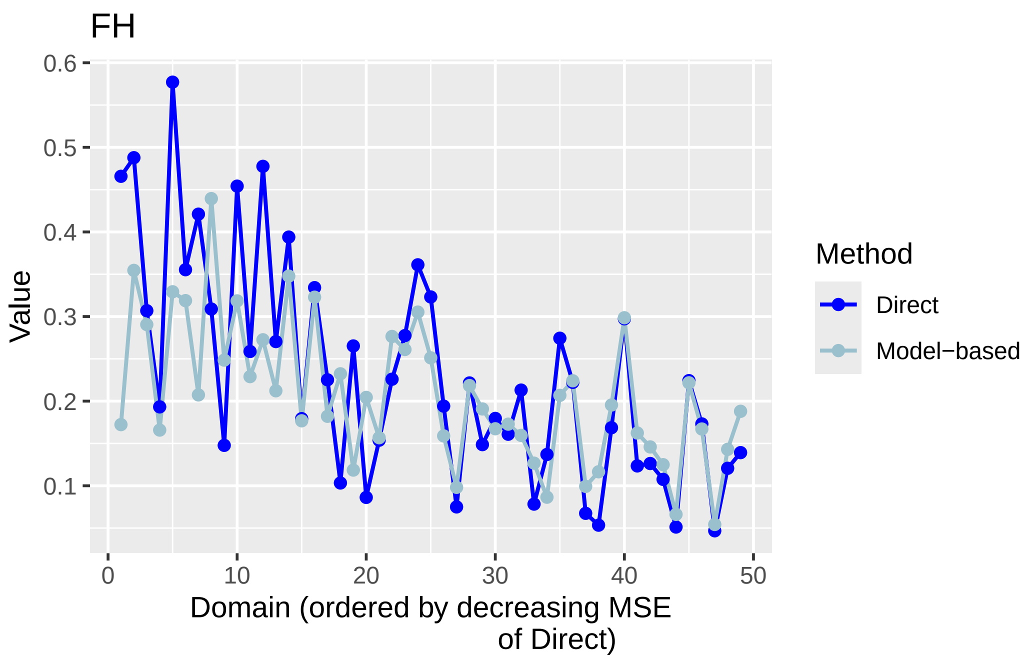
The FH estimates are expected to align closely with the direct estimates in domains with small direct MSEs and/or large sample sizes. Moreover, incorporating auxiliary information should enhance the precision of the direct estimates. We produce a scatter plot proposed by Brown et al. (2001) and a line plot. The fitted regression and the identity line of the scatter plot should not differ too much. The FH estimates should track the direct estimates within the line plot especially for domains with a large sample size/small MSE of the direct estimator. Furthermore, we compare the MSE and CV estimates for the direct and FH estimators using boxplots and ordered scatter plots (by setting the input arguments `MSE` and `CV` to `TRUE`).

Additionally, we compute a correlation coefficient of the direct estimates and the estimates of the regression-synthetic part of the FH model (Chandra, Salvati, and Chambers 2015) and a goodness of fit diagnostic (Brown et al. 2001).

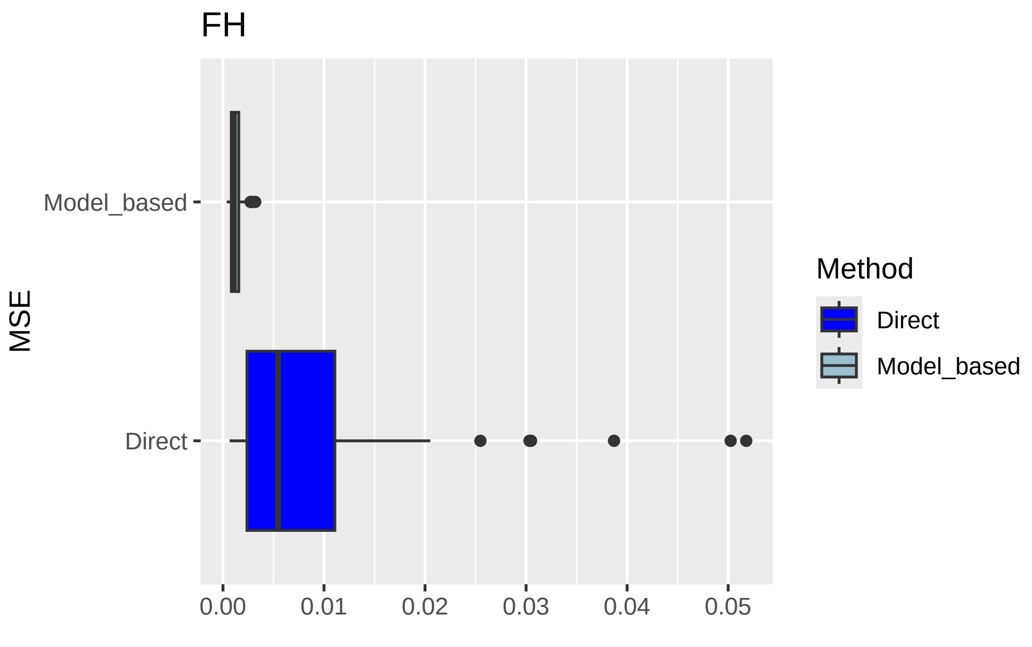
```
compare_plot(fh_model, MSE = TRUE, CV = TRUE)
```



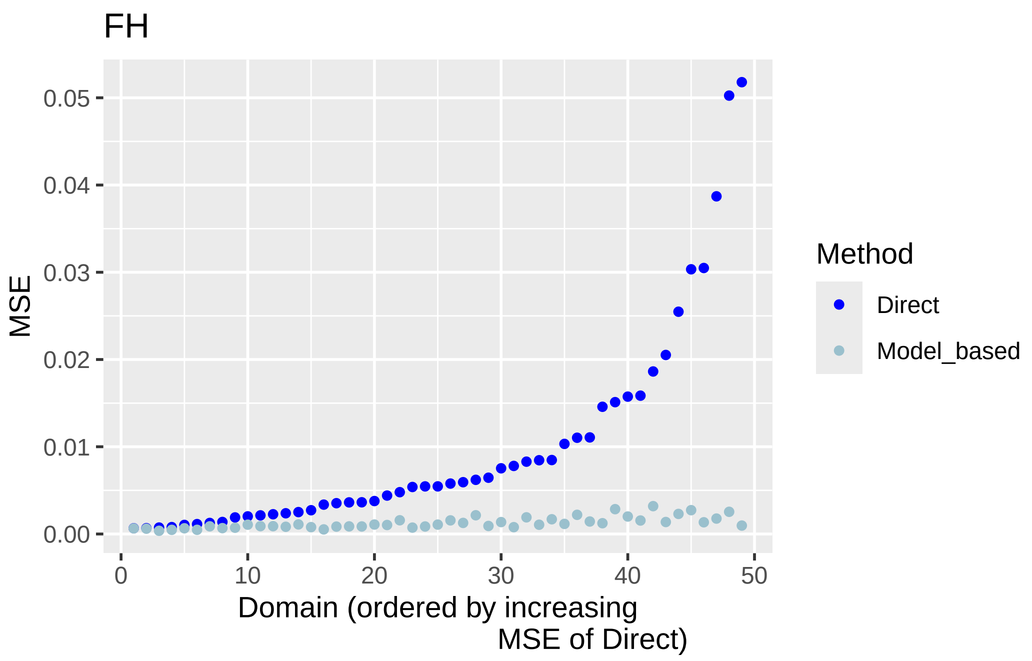
```
Press [enter] to continue
```



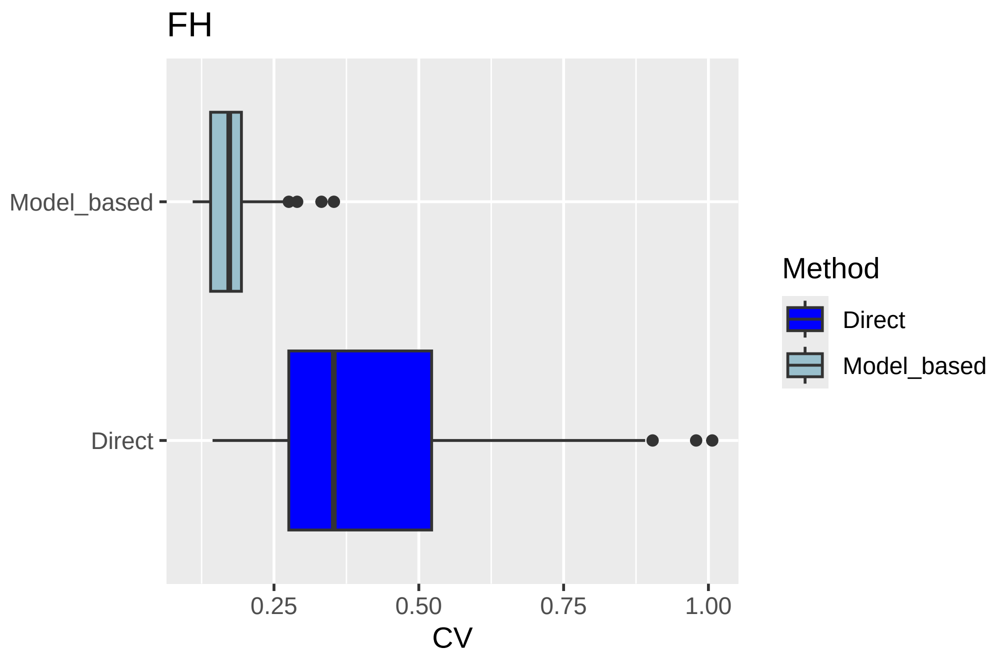
```
Press [enter] to continue
```



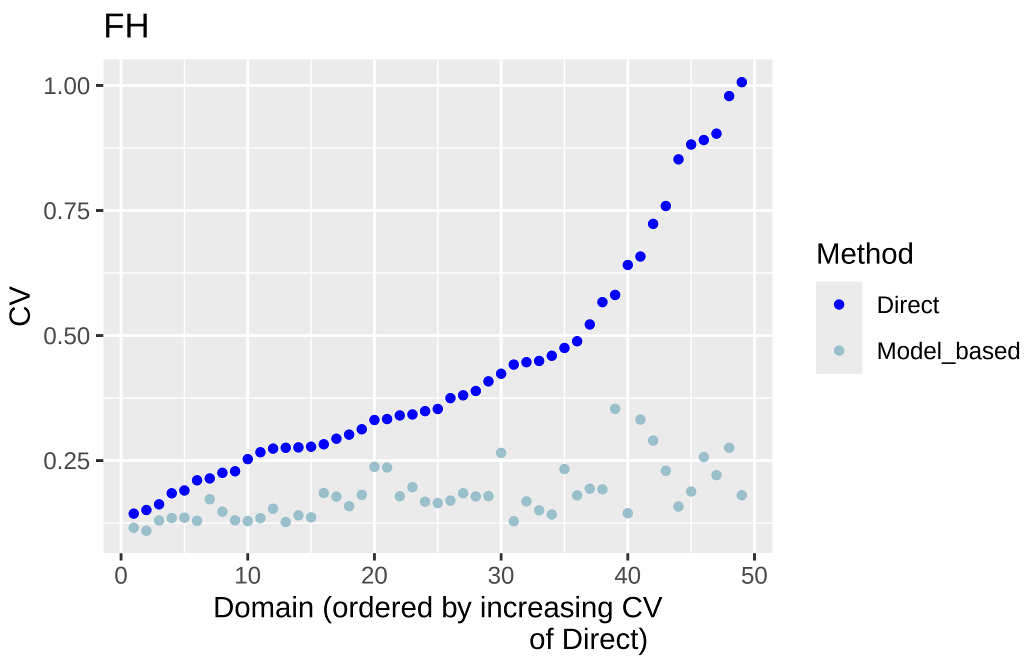
```
Press [enter] to continue
```



```
Press [enter] to continue
```



```
Press [enter] to continue
```



```r
compare(fh_model)
```

```
Brown test

Null hypothesis: EBLUP estimates do not differ significantly from the
      direct estimates

  W.value Df   p.value
 27.97119 49 0.9932077

Correlation between synthetic part and direct estimator:  0.66
```

The direct estimates are tracked by most of the FH estimates within the line plot. The precision of the direct estimates could be improved by the usage of the FH model in terms of MSEs and CVs. The null hypothesis of the Brown test is not rejected and the correlation coefficient indicates a positive correlation (0.66) between the direct and FH estimates.

If the result of the model assessment is not satisfactory, the following should be checked again: Can the direct estimation including variance estimation be improved? Are there further auxiliary variables and/or must possible interaction effects be taken into account? Does a (different) transformation need to be used?

Benchmark the FH point estimates for consistency with higher results.

Benchmarking is based on the principle that aggregated FH estimates should sum up to the estimates at a higher regional level. For the benchmark function, a benchmark value and a vector containing the shares of the population size per area (N_d/N) is required. Please note, that only the FH estimates are benchmarked and not their MSE estimates. As benchmark types "raking", "ratio" and "MSE_adj" can be chosen. For further details about using the function, please refer to the `emdi` vignette and for general information about the benchmarking options to Datta et al. (2011).

```r
## compute the benchmark value (mean of poverty indicator for the whole country)
benchmark_value <- weighted.mean(survey_dt$pov_indicator, survey_dt[[weight_var]])

## compute the share of population size in the total population size (N_d/N) per area
data("sizeprov")
fh_dt <- fh_dt |>
  left_join(sizeprov |>
  mutate(ratio_n = Nd/sum(Nd)), by = c("Domain" = area_vars[1]))

fh_bench <- benchmark(fh_model,
                      benchmark = benchmark_value,
                      share = fh_dt$ratio_n,
                      type = "ratio",
                      overwrite = TRUE)
head(fh_bench$ind)
```

```
  Domain     Direct         FH   FH_Bench Out
1      1 0.30885799 0.43938750 0.46776070   0
2      2 0.05339445 0.11648959 0.12401184   0
3      3 0.19412452 0.15892475 0.16918723   0
4      4 0.42100526 0.20736148 0.22075173   0
5      5         NA 0.09212262 0.09807139   1
6      6 0.15381464 0.15675123 0.16687335   0
```

Preparation of the results.

Create one dataframe that contains the direct and FH estimation results including MSE and CV results.

```r
pov_fh <- as.data.frame(estimators(fh_model, MSE = TRUE, CV = TRUE))
head(pov_fh)
```

```
  Domain     Direct  Direct_MSE Direct_CV         FH        FH_MSE     FH_CV
1      1 0.30885799 0.018628563 0.4419070 0.43938750 0.0031930592 0.1286044
2      2 0.05339445 0.002262765 0.8908891 0.11648959 0.0008970256 0.2571078
3      3 0.19412452 0.005455911 0.3804988 0.15892475 0.0008615970 0.1846973
4      4 0.42100526 0.020521753 0.3402669 0.20736148 0.0013723833 0.1786527
5      5         NA          NA        NA 0.09212262 0.0011072566 0.3612084
6      6 0.15381464 0.006450585 0.5221579 0.15675123 0.0009226177 0.1937760
```

```
pov_fh <- pov_fh |>
  rename(!!area_vars[1] := "Domain") |>
  mutate(year = singleyear)

bd_out |>
  pin_write(x = pov_fh,
            name = "pov_fh",
            type = "rds")

write.csv(pov_fh, "data/clean-example/pov_fh.csv")
```

Poverty map

With the help of geographical maps, the results can be presented in a user-friendly way and differences among the areas can be detected more easily. For the map, a shape file is reqired. The domain identifiers in the results object (`fh_model`) need to match to the respective identifiers of the shape file. Therefore, we create a mapping table first and then produce the map by `emdi::map_plot`.

```
## Create a suitable mapping table
## Find the right order
domain_ord <- match(shp_dt[[area_vars[1]]], fh_model$ind$Domain)

## Create the mapping table based on the order obtained above
map_tab <- data.frame(pop_data_id = fh_model$ind$Domain[domain_ord],
                      shape_id = shp_dt[[area_vars[1]]])

## Create map
map_plot(object = fh_model, MSE = TRUE, map_obj = shp_dt,
 map_dom_id = area_vars[1], map_tab = map_tab)
```

```
Press [enter] to continue
Press [enter] to continue

Press [enter] to continue
```

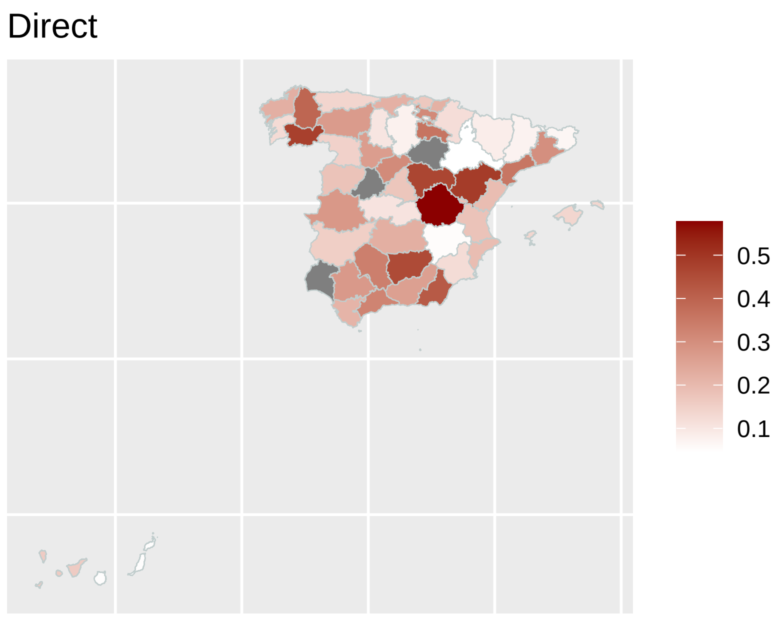


Figure 1

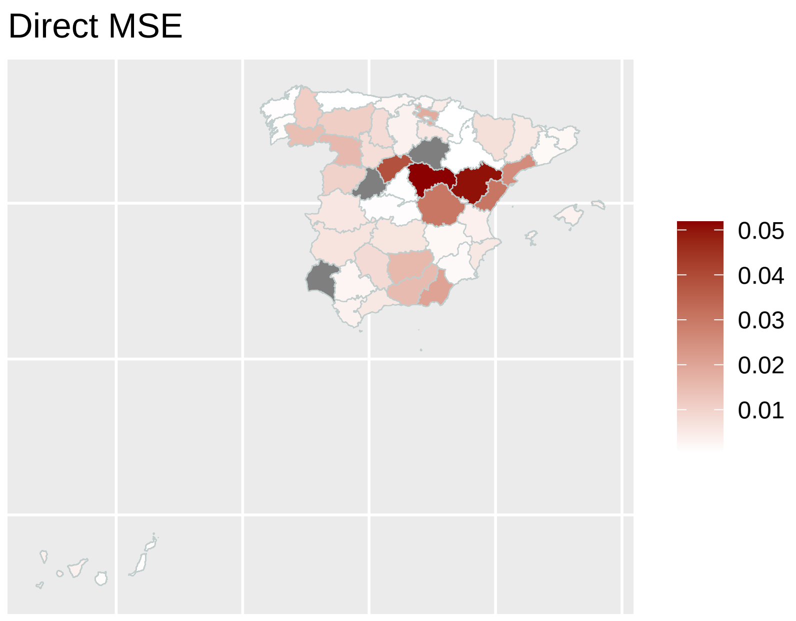


Figure 2

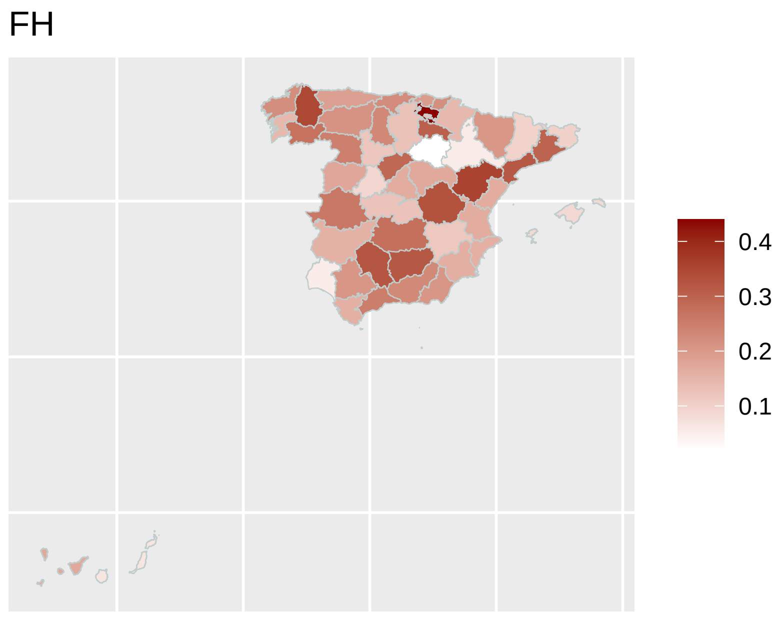


Figure 3

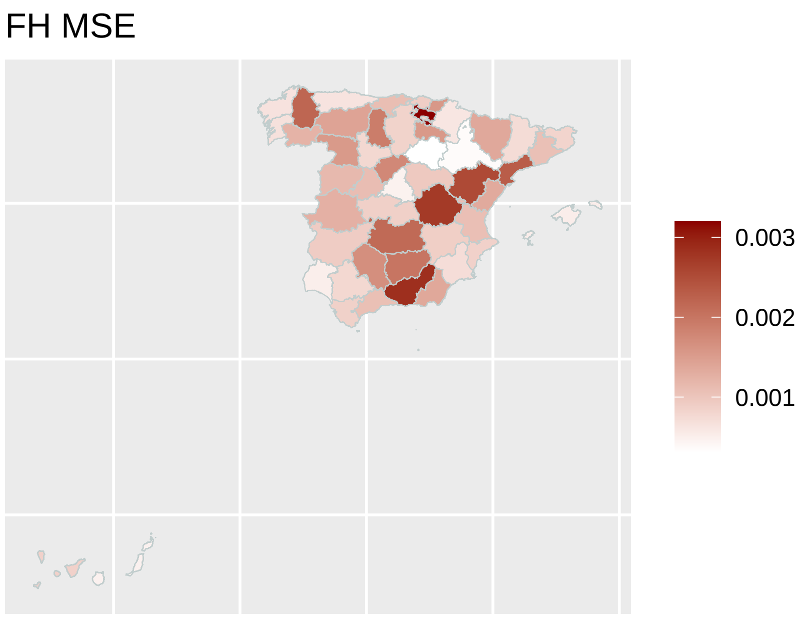


Figure 4

Saving the results.

Either by using `save.image("fh_estimation.RData")` or export of the model output and estimation results to Excel or OpenDocument spreadsheet (ODS) `write.excel(fh_model, file = "fh_model_output.xlsx", MSE = TRUE, CV = TRUE`.

References

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

Casas-Cordero, C., J. Encina, and P. Lahiri. 2016. "Poverty Mapping for the Chilean Comunas." In *Analysis of Poverty by Small Area Estimation*, by M. Pratesi, 379–403. John Wiley & Sons. https://doi.org/10.1002/9781118814963.ch20.

Chandra, H., N. Salvati, and R. Chambers. 2015. "A Spatially Nonstationary Fay-Herriot Model for Small Area Estimation." *Journal of the Survey Statistics and Methodology* 3 (2): 109–35. https://doi.org/10.1093/jssam/smu026.

Datta, G. S., M. Ghosh, R. Steorts, and J. Maples. 2011. "Bayesian Benchmarking with Applications to Small Area Estimation." *TEST* 20 (3): 574–88. https://doi.org/10.1007/s11749-010-0218-y.

Fay, Robert E., and Roger A. Herriot. 1979. "Estimates of Income for Small Places: An Application of James-Stein Procedures to Census Data." *Journal of the American Statistical Association* 74: 269–77.

Harmening, Sylvia, Ann-Kristin Kreutzmann, Sören Schmidt, Nicola Salvati, and Timo Schmid. 2023. "A Framework for Producing Small Area Estimates Based on Area-Level Models in r." *The R Journal* 15 (1): 316–41. https://doi.org/10.32614/RJ-2023-039.

Komsta, Lukasz, and Frederick Novomestky. 2015. *Moments: Moments, Cumulants, Skewness, Kurtosis and Related Tests*. https://CRAN.R-project.org/package=moments.

Lahiri, P., and J. Suntornchost. 2015. "Variable Selection for Linear Mixed Models with Applications in Small Area Estimation." *The Indian Journal of Statistics* 77-B (2): 312–20. https://www.jstor.org/stable/43694416.

Lumley, Thomas. 2024. "Survey: Analysis of Complex Survey Samples."

Marhuenda, Y., D. Morales, and M. del Camen Pardo. 2014. "Information Criteria for Fay-Herriot Model Selection." *Computational Statistics and Data Analysis* 70: 268–80. https://doi.org/10.1016/j.csda.2013.09.016.

Molina, Isabel, and Yolanda Marhuenda. 2015. "sae: An R Package for Small Area Estimation." *The R Journal* 7 (1): 81–98. https://journal.r-project.org/archive/2015/RJ-2015-007/RJ-2015-007.pdf.

Rao, J. N. K., and Isabel Molina. 2015. *Small Area Estimation*. John Wiley; Sons, Inc, Hoboken, NJ, USA.

Schmid, T., F. Bruckschen, N. Salvati, and T. Zbiranski. 2017. "Constructing Sociodemographic Indicators for National Statistical Institutes Using Mobile Phone Data: Estimating Literacy Rates in Senegal." *Journal of the Royal Statistical Society A* 180 (4): 1163–90. https://doi.org/10.1111/rssa.12305.

You, Yong, and Mike Hidiroglou. 2023. "Application of Sampling Variance Smoothing Methods for Small Area Proportion Estimation." *Journal of Official Statistics* 39 (4): 571–90.