Supplementary material to

A Comprehensive Analysis of the Italian School System Using Harmonized Open Data via the SchoolDataIT R Package

Leonardo Cefalo[1], Alessio Pollice[1], Paolo Maranzano[2]

[1]Department of Economics and Finance, Università degli Studi di Bari Aldo Moro, Bari, Italy.
[2]Dipartimento di Scienze Statistiche, Università degli studi di Milano Bicocca, Milan, Italy.

1 R chunks for maps

Here we provide the R code for the maps shown in section 4 and for the relevant input data. To save computational time, we download the shapefiles directly:

```
# Shapefile for provinces
Prov22_shp <- Get_Shapefile(2022, level = "NUTS-3")
# Shapefile for municipalities
Mun22_shp <- Get_Shapefile(2022, level = "LAU")
```

Section 4.2

This is the code for the maps in section 4.2. Firstly we define the input dataset; for brevity data editing and harmonisation is left up to the mapping function. If the `plot = ggplot` command is skipped, it is possible to visualise interactively all the values of the aggregated dataset.

```
Registry23 <- Get_Registry(2023)
input_DB23_MIUR <- Get_DB_MIUR(Year = 2023,
                               input_Registry = Registry23)

# Left map
```

```
Map_School_Buildings(data = input_DB23_MIUR,
                     field = "Urban_public_transport",
                     input_shp = Prov22_shp, level = "NUTS-3",
                     order = "High",
                     plot = "ggplot",
                     InnerAreas = FALSE)
# Right map
Map_School_Buildings(data = input_DB23_MIUR,
                     field = "Urban_public_transport",
                     input_shp = Mun22_shp, level = "LAU",
                     order = "High", region_code = 16,
                     plot = "ggplot",
                     InnerAreas = FALSE)
```

Section 4.3

This is the code for the maps in section 4.3. Since we are interested in plotting classroom size in high schools, we only filter out extreme values in high schools. Specifically, we remove all schools where any high school grade (option `filter_by_grade = TRUE`) has an average classroom size outside the interval of $[10, 40]$ students. For primary and middle schools we keep all observations by maintaining the default boundaries.

```
# Input
input_nstud23 <- Get_nstud(Year = 2023)
nteachers23 <- Get_nteachers_prov(Year = 2023 )

# Output 1
nstud23 <- Group_nstud(input_nstud23,
                       input_Registry = Registry23,
                       Year = 2023,
                       LB_nstud_byclass_grade = c(1,1,10),
                       UB_nstud_byclass_grade = c(99,99,40),
                       filter_by_grade = TRUE,
                       InnerAreas = TRUE,
                       ord_InnerAreas = TRUE)

# Output 2
teachers4stud23 <- Group_teachers4stud(
  input_nteachers = nteachers23,
  input_nstud_aggr = nstud23)


# Left map
Map_DB(data = nstud23$Province_data,
       field = "Students_per_class_10",
       input_shp = Prov22_shp, level = "NUTS-3",
```

```
       plot = "ggplot",
       main = "Students per class",
       col_rev = TRUE)

# Right map
Map_DB(data = teachers4stud23,
       field = "Teachers_per_student",
       order = "High",
       input_shp = Prov22_shp, level = "NUTS-3",
       plot = "ggplot",
       main = "Teachers per student")
```

In doing so, we get informed that:

```
Filtered out 1426 schools with either less than 1 (primary), 1 (middle), 10 (high)
 or more than 99 (primary), 99 (middle), 40 (high) students per class in any grade
```

Section 4.5

This is the code for the maps in section 4.5;.

```
input_Invalsi <- Get_Invalsi_IS(multiple_out = TRUE)
Prov23 <- Get_Shapefile(2023, level = "NUTS-3")
# Left map
Map_Invalsi(input_Invalsi$Province_data, Year = 2024,
            input_shp = Prov23_shp, level = "NUTS-3",
            subj_toplot = "ITA", grade = 13,
            plot = "ggplot",
            main = "Italian score, 13th grade, 2024")

# Right map
Map_Invalsi(input_Invalsi$Province_data, Year = 2024,
            input_shp = Prov23_shp, level = "NUTS-3",
            subj_toplot = "ERE", grade = 13,
            plot = "ggplot",
            main = "English reading score, 13th grade, 2024")
```

2 R code for the Usage Example

In the following we provide the R code for the usage example of this package described in section 5. We assume to be interested in gathering as much information as we can for the last school year whose data are completely available, i.e. 2023/24.

We include in the DB the Invalsi census survey at the municipality level, the counts of students, and the centroids of municipalities as defined on January 1st 2023 (this latter object being useful for the coordinates of municipalities). The object `School2mun24 is needed to map schools to municipalities; if not provided as an input to Group_nstud it would be downloaded`

automatically but not saved in the global environment. Option Year = 2023 is required since this object also includes the registry of school buildings which is currently (April 3rd 2025) not available for school year 2023/24. Teachers counts and the school buildings DB are not included since they are not available; ultra-broadband activation is not of interest.

```
Registry24 <- Get_Registry(2024)
School2mun24 <- Get_School2mun(2023,
                               input_Registry = Registry24)
input_nstud24 <- Get_nstud(2024)
nstud24_byclass <- Util_nstud_wide(input_nstud24,
                                   LB_nstud_byclass = c(1, 10, 1),
                                   UB_nstud_byclass = c(99, 40, 99),
                                   filter_by_grade = TRUE)
input_Invalsi_mun <- Get_Invalsi_IS(level ="LAU")
Mun23_ctr <- Get_Shapefile(2023, centroids = TRUE)
```

We join input data with the `SET_DB` function. Student counts are defined at the level of schools or school buildings, while Invalsi scores are already aggregated across municipalities. The common aggregation level we choose is clearly LAU (municipalities, default option). Please notice we remove all middle schools with average classroom size of less than 10 or more than 40 students.

```
DB24 <- Set_DB(Year = 2024,
               ord_InnerAreas = TRUE,
               BroadBand = FALSE,
               input_Registry = Registry24,
               input_School2mun = School2mun23,
               SchoolBuildings = FALSE,
               input_Invalsi_IS = input_Invalsi$Municipality_data,
               input_nstud = nstud24_byclass,
               input_nteachers = nteachers23)
Y
Y
```

Each row of the final dataset is a combination of municipality and school order, for a total of 13,456 rows (Italian municipalities at September 1st 2023 were 7,901 school orders are Primary, Middle and High school). Not all inputs have the same data availability. As a consequence, for each of them some row entries may be missing. By default, the system leaves the choice to keep or discard missing observations to the user. The most critical dataset, to this aim, is the Invalsi census survey; more than half of the rows are indeed missing.

```
8603  units in  Registry   are missing in  Invalsi_IS
Do you want to clean NA values?
     - To remove NAs, please press 'Y'
     - To abort the operation and cancel the join, please press 'A'
     - To keep the NAs, press any other key
```

```
 (please, do not use quotes in the prompt)

  > Y
NAs deleted
```

We choose to remove the municipalities with missing Invalsi records by typing `Y` (Yes) in the prompt. The same option will be triggered for students counts; we remove the NAs in the same way; the final database has 4.704 rows. To shut down the interactive choice, the user can set either `Set_DB(... , NA_autoRM = FALSE)` to automatically retain all missing entries or `Set_DB(... , NA_autoRM = TRUE)` to delete all them.

The plot of the two variables whose association is of interest, i.e. Invalsi scores in Mathematics and class size at the last year of middle school, is in Section 5.1 of the paper:

```
Map_DB(DB24, input_shp = Prov23_shp, field = "M_Mathematics_8",
       order = "Middle",level = "NUTS-3", plot = "ggplot",
       main = "Invalsi scores, grade 8")
Map_DB(DB24, input_shp = Prov23_shp,field = "Students_per_class_8",
       order = "Middle",level = "NUTS-3", plot = "ggplot",
       main = "Class size, grade 8")
```

Next, we arrange the data object for the regression model. To make the model interpretable, please notice we scale the class size to have mean = 0 and variance = 1.

```
library(magrittr)

DB24_mid <- DB24 %>%
  dplyr::filter(.data$Classes_grade_8 >0) %>%
  #Save storage space: remove irrelevant school orders
  dplyr::select_if(!grepl("_\\d+", names(.)) |
                     grepl("_8$|_08$", names(.))) %>%
  dplyr::mutate(nstud_8_std =
                  as.vector(scale(.data$Students_per_class_8))) %>%
  dplyr::filter(!is.na(.data$M_Mathematics_8))

# Add centroids
DB24_mid_ctr <- Mun23_ctr %>% dplyr::transmute(Municipality_code = .data$PRO_COM_T) %>%
  dplyr::left_join(DB24_mid) %>%
  dplyr::filter(!is.na(.data$Province_code))

# Add coordinates
DB24_mid_ctr$lat <- scale(sf::st_coordinates(DB24_mid_ctr))[,2]
DB24_mid_ctr$long <- scale(sf::st_coordinates(DB24_mid_ctr))[,1]
```

Spatial regression is carried out in the `R-INLA` framework (https://www.r-inla.org/). To run the model, we define the mesh and index all its nodes. In this case, the mesh has more nodes than the observation points. The object `A.mat` is the projection

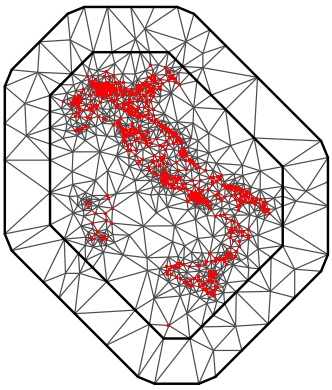

Fig. 1 Mesh employed to fit this model.

matrix A, such that $u = Az$, where u is the latent Matérn field and z is the Gaussian Markov random field taking values on the mesh (function `INLA::inla.spde.make.A`):

```
library(INLA)
coords <- sf::st_coordinates(DB24_mid_ctr)
distances <- sf::st_distance(DB24_mid_ctr, DB24_mid_ctr)
mesh <- inla.mesh.2d(loc = coords,
                     max.edge = c(max(distances)/5, max(distances)/2) )
spde <- inla.spde2.pcmatern(mesh = mesh, alpha = 2,
                            prior.range = c(2e+5, 0.01),
                            prior.sigma = c(5, 0.05))
A.mat <- inla.spde.make.A(mesh=mesh, loc= coords)
idx <- inla.spde.make.index(name = "idx", n.spde = spde$n.spde)

plot(mesh, cex = 1)
points( sf::st_coordinates(DB24_mid_ctr), pch = ".", cex = 2, col="red")
```

In figure 1 we display the mesh and the nodes corresponding to locations at which z is defined.

Then the relevant data are arranged in the format required by `R-INLA`.

```
DB24_mid_stack <- inla.stack(
  data = list(M_Mathematics_8 = DB24_mid_ctr$M_Mathematics_8),
  A = list(A.mat, 1),
```

```
  effects = list(c(idx), list(
    Intercept = rep(1, nrow(DB24_mid_ctr)),
    nstud_8_std = DB24_mid_ctr$nstud_8_std,
    Inner_area = DB24_mid_ctr$Inner_area,
    lat = DB24_mid_ctr$lat,
    long = DB24_mid_ctr$long)))
```

And the regression model is run:

```
m8_spde <- inla(
  M_Mathematics_8 ~ -1 + Intercept +
    nstud_8_std + Inner_area + lat + long +
    f(idx, model = spde),
  data = inla.stack.data(DB24_mid_stack, spde = spde),
  family = "gaussian",
  control.fixed = list(prec = 0),
  control.predictor = list(A = inla.stack.A(DB24_mid_stack)),
  # Commands to make code reproducible
  control.compute = list(internal.opt = FALSE, waic = T),
  num.threads = 1, verbose = T)
```

Lastly, this is the code to plot the expected linear trend and the expected spatial latent effect. To the latter aim, we define a grid and map $\mathbb{E}[u|y]$ from the mesh (discrete domain) to the grid (continuous domain).

```
x_range <- range(coords[, 1])
y_range <- range(coords[, 2])
grid <- expand.grid(
  x = seq(x_range[1], x_range[2], length.out = 400),
  y = seq(y_range[1], y_range[2], length.out = 400)
)

A_grid <- inla.spde.make.A(mesh = mesh, loc = as.matrix(grid))

grid_sf <- data.frame(
  x = grid$x, y = grid$y,
  GF = as.vector(A_grid %*% m8_spde$summary.random$idx$mean),
  trend = as.vector(
    scale(grid) %*% m8_spde$summary.fixed$mean[c(5,4)])) %>%
  sf::st_as_sf(coords = c("x", "y"), crs = 32632)
intersected_sf <- sf::st_intersection(grid_sf, Prov23_shp)
intersected_sf$x <- sf::st_coordinates(intersected_sf)[,1]
intersected_sf$y <- sf::st_coordinates(intersected_sf)[,2]

# Linear spatial trend:
ggplot2::ggplot(
    intersected_sf, ggplot2::aes(x=x, y=y, fill = .data$trend)) +
    ggplot2::geom_tile() +
```

	mean	s.d.	$Q_{0.025}$	$Q_{0.5}$	$Q_{0.975}$
σ^2_ε	44.341	3.808	37.479	44.108	52.422
σ^2	29.079	6.760	17.923	28.353	44.353
r	106.499	27.255	64.741	102.330	171.094

Table 1 Summaries of the hyperparameters of the spatial regression model

```
    ggplot2::scale_fill_viridis_c()+
    ggplot2::ggtitle("Expected linear trend")+
    ggplot2::theme(
        axis.text.x = ggplot2::element_blank(),
        axis.text.y = ggplot2::element_blank(),
        panel.background =
            ggplot2::element_rect(fill = "white", color = NA),
        panel.border =
            ggplot2::element_rect(color = "black", fill = NA),
        panel.grid.major =
            ggplot2::element_line(color = "grey80"),
        panel.grid.minor =
            ggplot2::element_line(color = "grey80"),
    )
# Latent spatial effect:
ggplot2::ggplot(
    intersected_sf, ggplot2::aes(x=x, y=y, fill = .data$GF)) +
    ggplot2::geom_tile() +
    ggplot2::scale_fill_viridis_c()+
    ggplot2::ggtitle("Expected Gaussian field")+
    ggplot2::theme(
        axis.text.x = ggplot2::element_blank(),
        axis.text.y = ggplot2::element_blank(),
        panel.background =
            ggplot2::element_rect(fill = "white", color = NA),
        panel.border =
            ggplot2::element_rect(color = "black", fill = NA),
        panel.grid.major =
            ggplot2::element_line(color = "grey80"),
        panel.grid.minor =
            ggplot2::element_line(color = "grey80"),
    )
```

Lastly, we show in table 1 the summaries of hyperparameters, respectively the variance of the Gaussian error σ^2_ε, the global variance of the latent effect σ^2 and the range of the latent effect r, expressed in kilometers.

The estimated range has a posterior median of ≈ 102 km, a value in the order of the distance of municipalities within the same region. The global variance of the latent

effect is smaller than the error variance, denoting that most variability unexplained by the covariates is attributable to noise.