Live births in Germany: density-on-scalar regression

Eva-Maria Maier *

Wirtschaftswissenschaftliche Fakultät,
Humboldt-Universität zu Berlin,
Unter den Linden 6, D-10099 Berlin, Germany.

This vignette illustrates how to use **FDboost**, which was designed for functional regression (Brockhaus et al., 2015), for density-on-scalar regression. Despite being a special case of function-on-scalar regression (at least for densities defined on a nontrivial interval with respect to the Lebesgue measure, which we refer to as *continuous case*), it has to be treated differently due to the special properties of probability density functions, namely nonnegativity and integration to one. Our vignette is based on the approach by Maier et al. (2021).

1 Load and plot data

We use the data set `birthDistribution` from the package **FDboost**, containing densities of live births in Germany over the months per year (1950-2019) and sex (male and female), resulting in 140 densities. It is a list with the following elements:

- `birth_densities`: A 140 x 12 matrix containing the birth densities in its rows. The first 70 rows correspond to male newborns, the second 70 rows to female ones. Within both of these, the years are ordered increasingly (1950-2019).
- `birth_densities_clr`: A 140 x 12 matrix containing the clr transformed densities in its rows. Same structure as `birth_densities`.
- `sex`: A factor vector of length 140 with levels `"m"` (male) and `"f"` (female), corresponding to the sex of the newborns for the rows of `birth_densities` and `birth_densities_clr`. The first 70 elements are `"m"`, the second 70 `"f"`.
- `year`: A vector of length 140 containing the integers $1950, \ldots, 2019, 1950, \ldots, 2019$, corresponding to the years for the rows of `birth_densities` and `birth_densities_clr`.
- `month`: A vector containing the integers from 1 to 12, corresponding to the months for the columns of `birth_densities` and `birth_densities_clr` (domain $\mathcal{T}$ of the (clr-)densities).

This list already is in the format needed to pass it to `FDboost`. Note that to compensate for the different lengths of the months, the average number of births per day for each month (by sex and year) was used to compute the birth shares from the absolute birth counts. The 12 shares corresponding to one year and sex form one density in the Bayes Hilbert space $B^2(\delta) = B^2(\mathcal{T}, \mathcal{A}, \delta)$, where $\mathcal{T} = \{1, \ldots, 12\}$ corresponds to the set of the 12 months, $\mathcal{A} := \mathcal{P}(\mathcal{T})$ corresponds to the power set of $\mathcal{T}$, and the reference measure $\delta := \sum_{t=1}^{12} \delta_t$ corresponds to the sum of dirac measures at $t \in \mathcal{T}$. Thus, our analysis is an example for the discrete case and the integral of a density is simply the sum of all 12 share values. We indicate how to proceed in the continuous case, whenever it is distinct from the discrete one over the course of this vignette. We denote the density contained in the i-th row of `birth_densities` with $f_i = f_{sex_i, year_i}$, where sex_i and $year_i$ denote the i-th elements of `sex` and `year`, respectively, $i = 1, \ldots, 140$. We load the package and the data and plot the densities:

*E-mail: eva-maria.maier@hu-berlin.de

```r
# load FDboost package
library(FDboost)
# load birth_densities
data("birthDistribution", package = "FDboost")

# function to plot a matrix or vector containing functions in B^2(delta) or L^2_0(delta);
# Is used for densities, effects, predictions (also clr transformed)
plot_function <- function(plot_matrix, ...) {
  funplot(1:12, plot_matrix, xlab = "month", xaxp = c(1, 12, 11), pch = 20, ...)
  abline( h = 0, col = "grey", lwd = 0.5)
}

# function to create two plots (by sex) from a matrix containing densities or predictions
# (also clr transformed) for males in first half of rows and females in second half
plot_birth_densities <- function(birth_matrix, ylim = range(birth_matrix), ...) {
  par(mfrow = c(1, 2))
  for (k in 1:2) {
    n_obs <- nrow(birth_matrix) / 2
    obs <- 1:n_obs + (k - 1) * n_obs
    plot_function(birth_matrix[obs, ], main = c("Male", "Female")[k],
                  ylim = ylim, col = rainbow(n_obs, start = 0.5, end = 1),
                  lty = c(1, 2, 4, 5), ...)
  }
}

# Plot densities
plot_birth_densities(birthDistribution$birth_densities, ylab = "densities")
```

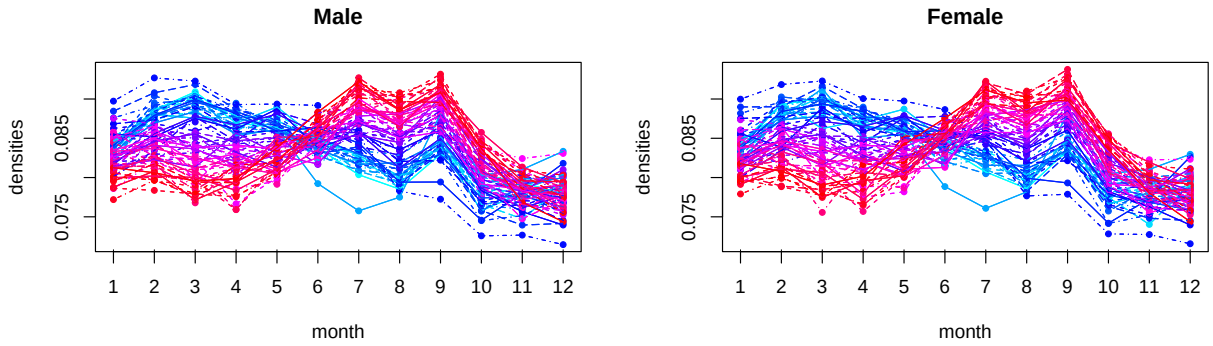

Figure 1: Densities of births in Germany per year and sex in $B^2(\delta)$. Years are coded by different colors and line types, see Figure 2.

```r
# legend (also for later plots)
year_col <- rainbow(70, start = 0.5, end = 1)
year_lty <- c(1, 2, 4, 5)
par(mar = c(0, 0, 0, 0) + 0.1)
plot(NULL, xaxt = "n", yaxt = "n", bty = "n", ylab = "", xlab = "", xlim = 0:1, ylim = 0:1)
legend("top", xpd = TRUE, legend = 1950:2019, lty = year_lty, ncol = 10, bty = "n",
       text.col = year_col, col = year_col, cex = 0.7)
```

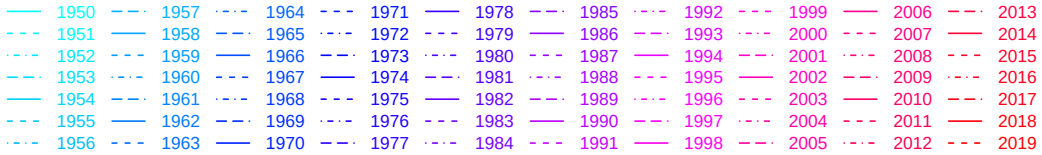

Figure 2: Coding of the years.

Overall, the range of the density values is quite small (from 0.071 to 0.094). While there is hardly a visible difference between the two sexes, we see a trend over the years: In the early months of the years the density values tend to decrease, in the later ones it is vice versa.

2 Model equation and clr transformation

We consider the model

$$f_i = \beta_0 \oplus I(sex_i = sex) \odot \beta_{sex} \oplus g(year_i) \oplus \varepsilon_i, \qquad i = 1, \ldots, 140, \tag{1}$$

with a group-specific intercept β_{sex} for $sex \in \{\text{male, female}\}$, a flexible effect $g(year)$ for $year \in [1950, 2019]$, and functional error terms $\varepsilon_i \in B^2(\delta)$ with $\mathbb{E}(\varepsilon_i) = 0$ the additive neutral element of $B^2(\delta)$, corresponding to a constant density. Equivalently, we can consider the centered log-ratio (clr) transformed model

$$\operatorname{clr}[f_i] = \operatorname{clr}[\beta_0] + I(sex_i = sex) \cdot \operatorname{clr}[\beta_{sex}] + \operatorname{clr}[g(year_i)] + \operatorname{clr}[\varepsilon_i] \qquad i = 1, \ldots, 140, \tag{2}$$

for estimation, which is part of $L_0^2(\delta) = L_0^2(\mathcal{T}, \mathcal{A}, \delta) = \left\{ f \in L^2(\delta) \mid \int_{\mathcal{T}} f \, \mathrm{d}\delta = 0 \right\}$, a closed subspace of $L^2(\delta) = L^2(\mathcal{T}, \mathcal{A}, \delta)$. **FDboost** was desiged for functions in $L^2(\mathbb{R}, \mathfrak{B}, \lambda)$, where $\mathfrak{B}$ denotes the Borel σ-algebra and λ the Legesgue measure. However, with some unfamiliar specifications, **FDboost** can be used to estimate model (2). Thus, our first step towards estimation is to apply the clr transformation on our densities, which is given by

$$\operatorname{clr}[f] := \log f - \frac{1}{\delta(\mathcal{T})} \int_{\mathcal{T}} \log f \, \mathrm{d}\delta = \log f - \frac{1}{12} \sum_{t=1}^{12} \log f(t). \tag{3}$$

We call the resulting clr transformed densities *clr-densities* in the following. The data set `birthDistribution` already contains the clr-densities. Whenever that's not the case, one can use the function `clr()` to compute the clr-densities, which we include here for the sake of completeness. Note that the choice of appropriate integration weights `w` for the corresponding Bayes Hilbert space is crucial to get a reasonable result. In our discrete case, equal weights `w = 1` are appropriate. In the continuous case, the choice of the weights depends on the grid on which the function was evaluated. The weight for each function value must correspond to the length of the subinterval it represents. E.g., for a function defined on $\mathcal{T} = [a, b]$ evaluated on a grid with equidistant distance d, where the boundary grid values are $a + \frac{d}{2}$ and $b - \frac{d}{2}$ (i.e., the grid points are centers of subintervals of size d), equal weights d should be chosen for `w`.

```
# The function clr() can be used to compute the clr-densities; Our reference measure delta
# corresponds to equal integration weights w = 1 for all density values
birth_densities_clr_test <- t(apply(birthDistribution$birth_densities, 1, clr, w = 1))
# Compare with clr-densities contained in data set
sum(birth_densities_clr_test != birthDistribution$birth_densities_clr)

## [1] 1

# Plot clr-densities
plot_birth_densities(birthDistribution$birth_densities_clr, ylab = "clr-densities")
```

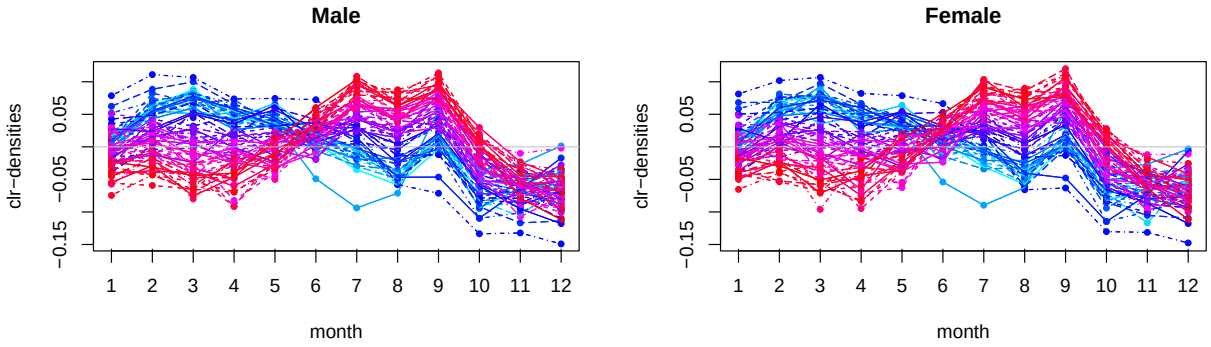

Figure 3: Clr transformed densities in $L_0^2(\delta)$. Years are coded by different colors and line types, see Figure 2.

Due to the small range of the density values in this example, their shape is very similar to the original densities, see Figure 1. In general, this is not the case.

3 Estimation

When fitting model (2) with the function `FDboost`, the specification of the `timeformula` needs some special attention. First, we must respect the integrate-to-zero constraint of $L_0^2(\delta)$. This is achieved by using the constrained base-learner `bbsc` in the `timeformula` (instead of the unconstrained `bbs` as usual), which transforms the basis such that it fulfills the sum-to-zero constraint. In our discrete case, this corresponds to the integrate-to-zero constraint directly. In the continuous case, the sum is proportional to the integral numerically, if the grid points where the function is evaluated are selected appropriately (e.g., centers of equal sized subintervals). Thus, using `bbsc` is suitable in this case, as well. Second, we must specify the B-spline basis in `bbsc` appropriately. The continuous case is straightforward, e.g., by using cubic B-splines. In our discrete case, a suitable (unconstrained) basis is $(\mathbb{1}_{\{1\}}, \ldots, \mathbb{1}_{\{12\}}) \in L^2(\delta)^{12}$, where $\mathbb{1}_A$ denotes the indicator function of $A \in \mathcal{A}$. This results in the identity matrix as design matrix. In `bbs()` (or `bbsc()`, which yields the corresponding constrained basis), this can be achieved using `degree = 1` with knots equal to $\mathcal{T}$.

```
model <- FDboost(birth_densities_clr ~ 1 + bolsc(sex, df = 1) +
                   bbsc(year, df = 1, differences = 1),
                 # use bbsc() in timeformula to ensure integrate-to-zero constraint
                 timeformula = ~bbsc(month, df = 4,
                                     # December is followed by January of subsequent year
                                     cyclic = TRUE,
                                     # knots = {1, ..., 12} with additional boundary knot
                                     # 0 (coinciding with 12) due to cyclic = TRUE
                                     knots = 1:11, boundary.knots = c(0, 12),
                                     # degree = 1 with these knots yields identity matrix
                                     # as design matrix
                                     degree = 1),
                 data = birthDistribution, offset = 0,
                 control = boost_control(mstop = 1000))
```

To determine the optimal stopping iteration we perform a 10-fold bootstrap. This is rather time-consuming (especially in the continuous case, when the response densities are evaluated at many grid-values) and preferably should be executed parallelized on multiple cores. In order to avoid long compilation times for the vignette, the following code is commented out, but it should be possible to obtain the same stopping iteration within a few minutes.

```
# set.seed(1708)
# folds <- applyFolds(model, folds = cv(rep(1, model$ydim[1]), type = "bootstrap", B = 10))
# ms <- mstop(folds) # = 999
ms <- 999
model <- model[ms]
```

Our final object `model` contains the fit of model (2), i.e., on clr-level.

```
# Plotting 'model' yields the clr-transformed effects
par(mfrow = c(1, 3))
plot(model, n1 = 12, n2 = 12)
```

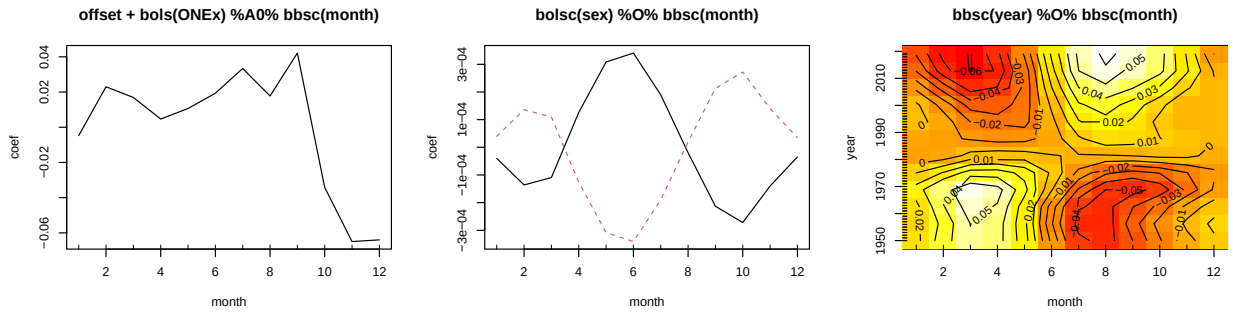

Figure 4: Estimated effects in $L_0^2(\delta)$. Years are coded by different colors and line types, see Figure 2.

Since model (2) is equivalent to model (1) via the clr transformation, we have to use the inverse clr transformation to get densities of interest (like estimated effects or predictions) for model (1) in the Bayes Hilbert space, after extracting them from `model`. The inverse clr transformation is given by

$$\mathrm{clr}^{-1}(\tilde{f}) := \frac{\exp \tilde{f}}{\int_{\mathcal{T}} \exp \tilde{f} \, \mathrm{d}\delta} = \frac{\exp \tilde{f}}{\sum_{t=1}^{12} \exp \tilde{f}(t)}.$$

for $\tilde{f} \in L_0^2(\delta)$ and can be computed using `clr(..., inverse = TRUE)`, again specifying appropriate integration weights `w`. Note that in contrast to Maier et al. (2021), the definition above includes normalization to obtain the probability density function (which is the representative of the equivalence class of proportional functions in $B^2(\delta)$).

```
# Get estimated clr transformed effects; we use predict(), which returns a matrix of the
# same dimension as the response (140 x 12), i.e., we have to extract the respective rows;
# Alternatively, one could use coef(), but has to specify n1 = 12, n2 = 12 to get the den-
# sities at 1, ..., 12, which also only yields the year effect on a grid of 12 years

# all rows contain intercept
intercept_clr <- predict(model, which = 1)[1, ]
# first 70 rows contain effect for sex = male, second 70 rows for sex = female
sex_clr <- predict(model, which = 2)[c(1, 71), ]
# first 70 rows contain effect for years from 1950 to 2019, second 70 rows are repetition
year_clr <- predict(model, which = 3)[1:70, ]

sex_col <- c("blue", "red")
par(mfrow = c(1, 3), mar = c(5, 5, 4, 2) + 0.1)

# Retransform to Bayes Hilbert space using clr(..., inverse = TRUE); Our reference measure
# delta corresponds to equal integration weights w = 1 for all function values
intercept <- clr(intercept_clr, w = 1, inverse = TRUE)
```

```
sex <- t(apply(sex_clr, 1, clr, w = 1, inverse = TRUE))
year <- t(apply(year_clr, 1, clr, w = 1, inverse = TRUE))

# Plot retransformed effects
plot_function(intercept, main = "Intercept", ylab = expression(hat(beta)[0]),
              id = rep(1, 12)) # id is passed to funplot since intercept is a vector
plot_function(sex, main = "Effect of sex", col = sex_col,
              ylab = expression(hat(beta)["sex"]))
legend("topleft", legend = c("sex = male", "sex = female"), text.col = sex_col, bty = "n")
plot_function(year, main = "Effect of year", col = year_col,
              ylab = expression(hat(g)("year")), lty = year_lty)
```

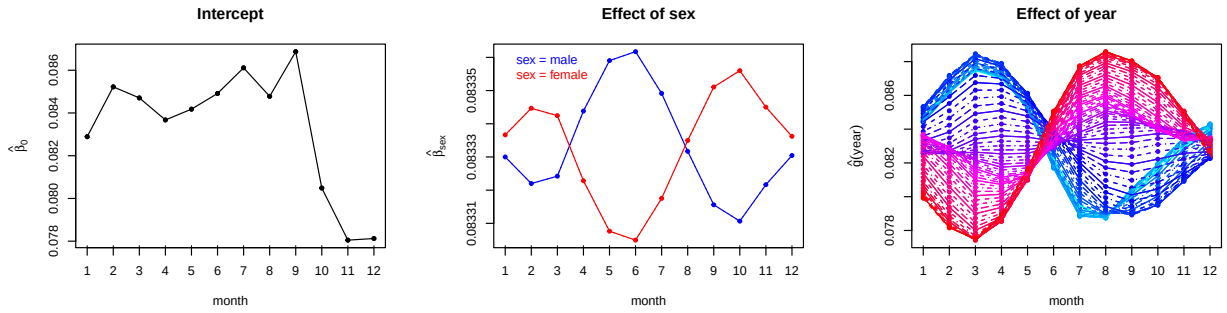

Figure 5: Estimated effects in $B^2(\delta)$. Years are coded by different colors and line types, see Figure 2.

While all effects get selected by the algorithm, the effects of sex are very small. We plot the predictions using the same range as in Figure 1 for better comparison:

```
predictions_clr <- predict(model)
predictions <- t(apply(predictions_clr, 1, clr, inverse = TRUE))
plot_birth_densities(predictions, ylim = range(birthDistribution$birth_densities),
                     ylab = "predictions")
```

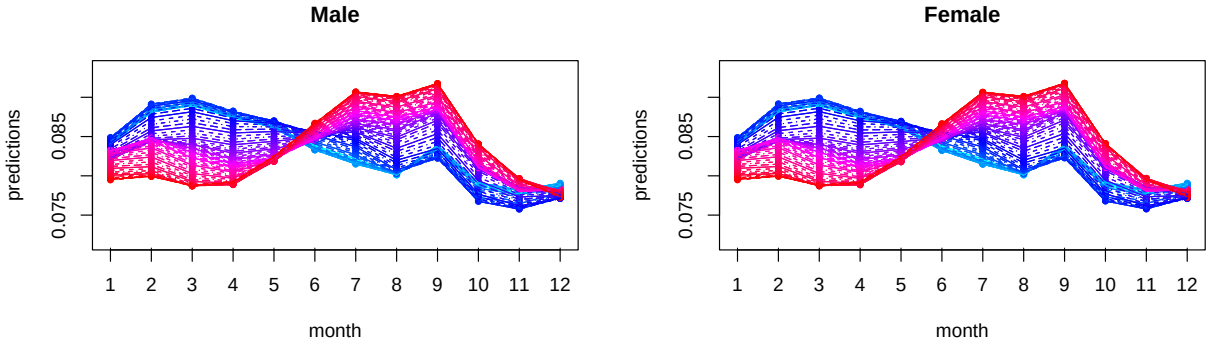

Figure 6: Predicted densities in $B^2(\delta)$. Years are coded by different colors and line types, see Figure 2.

References

Brockhaus, S., Scheipl, F., Hothorn, T., and Greven, S. (2015). The functional linear array model. *Statistical Modelling* 15(3), 279–300.

Maier, E.-M., Stöcker, A., Fitzenberger, B., Greven, S. (2021). Additive Density-on-Scalar Regression in Bayes Hilbert Spaces with an Application to Gender Economics. *arXiv preprint arXiv:2110.11771.*