Web-Appendix for Longitudinal Scalar-on-Functions Regression with Application to Tractography Data

Jan Gertheiss[*†] Jeff Goldsmith[‡]
Ciprian Crainiceanu[‡] & Sonja Greven[†]

January 19, 2012

Illustration of LFPCA

For illustration of LFPCA, the estimated mean function $\hat{\eta}(s,T)$ when modeling fractional anisotropy along the corticospinal tract is shown in Figure 1. It is seen that variation of $\hat{\eta}(s,T)$ over time T is very low. Figures 2 and 3 show the first three estimated functional principal components $\{\hat{\phi}_k^0, \hat{\phi}_k^1\}$, $k = 1,2,3$, and the first two curves $\hat{\phi}_r^U$, $r = 1,2$. The estimated variance components $\hat{\lambda}_k$ and $\hat{\nu}_r$, $k = 1,\ldots,15$, $r = 1,\ldots,8$, are found in Figure 4. These 15 + 8 components are needed to explain 90% of the variance in the data. Apparently, for both within-subject and between-subject variation, the largest part of variation is explained by variation in the general level of fractional anisotropy in distinct regions along the corticospinal tract. The B-process varies most strongly roughly between measurement points 25 and 50 (see Figure 2, top left); and for the U-process the interesting region is roughly the first half of the tract (see Figure 3, left).

Results of Simulation Studies

Results for scenarios 1b, 1c and 2, that are not shown in the main paper, are found in Figure 5. For a description of simulation scenarios and the definition of abbreviations, see the main paper (Gertheiss et al., 2012).

Additional Comparisons between Methods

- *Connections between LPFR and LFPCR using Scores*: In some cases, results from a regression model using LFPCA scores can be used for estimating the

[*]To whom correspondence should be addressed: `jan.gertheiss@stat.uni-muenchen.de`
[†]Department of Statistics, Ludwig-Maximilians-Universität Munich, Germany.
[‡]Department of Biostatistics, Johns Hopkins University, Baltimore, USA.

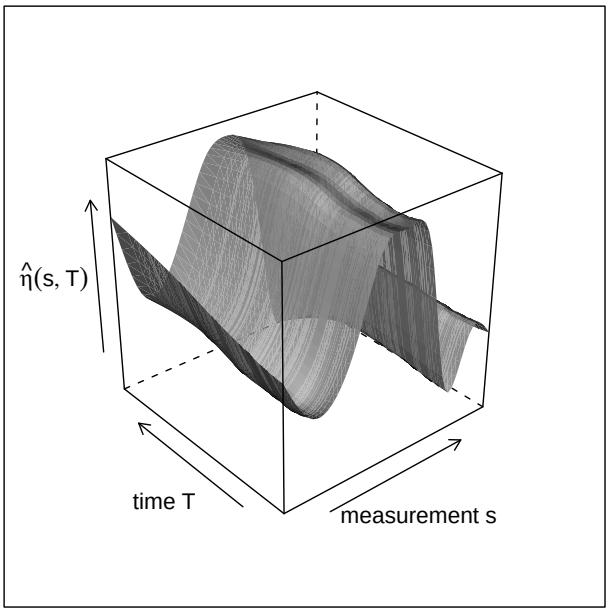

Figure 1: Estimated mean function $\hat{\eta}(d,T)$ when modeling fractional anisotropy along the corticospinal tract by LFPCA.

LPFR model. Since

$$\zeta_{ijr} = \int_{\mathcal{D}} U_{ij}(s)\phi_r^U(s)\,ds$$

and

$$U_{ij}(s) = W_{ij}(s) - \eta(s,T_{ij}) - B_{i,0}(s) - T_{ij}B_{i,1}(s) - \varepsilon_{ij}(s),$$

we have

$$\begin{aligned}\zeta_{ijr} &= \int_{\mathcal{D}} W_{ij}(s)\phi_r^U(s)\,ds - \int_{\mathcal{D}} B_{i,0}(s)\phi_r^U(s)\,ds \\ &- \int_{\mathcal{D}} \eta(s,T_{ij})\phi_r^U(s)\,ds - \int_{\mathcal{D}} T_{ij}B_{i,1}(s)\phi_r^U(s)\,ds - \int_{\mathcal{D}} \varepsilon_{ij}(s)\phi_r^U(s)\,ds.\end{aligned}$$

If variability over time of $\hat{\eta}(s,T)$ is very low (see Figure 1) and also the influence of $B_{i,1}(s)$ can be neglected (see, e.g., $\hat{\phi}_k^1$ in Figure 2), we have

$$\zeta_{ijr} \approx \int_{\mathcal{D}} W_{ij}(s)\phi_r^U(s)\,ds + \tilde{b}_{i,r} + c_r - \int_{\mathcal{D}} \varepsilon_{ij}(s)\phi_r^U(s)\,ds,$$

with constant c_r and "random intercept"

$$\tilde{b}_{i,r} = -\int_{\mathcal{D}} B_{i,0}(s)\phi_r^U(s)\,ds.$$

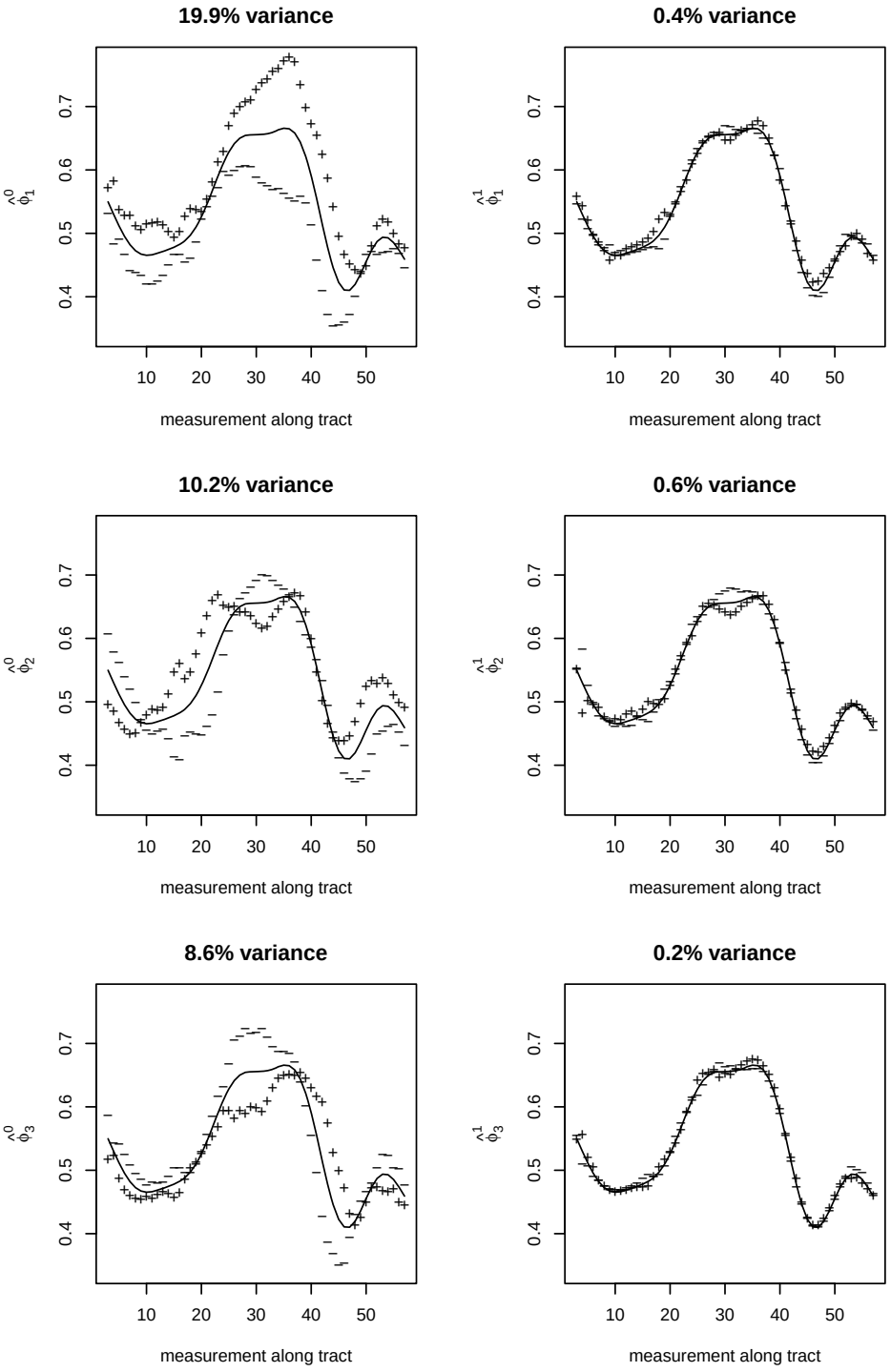

Figure 2: Estimated functional principal components $\{\hat{\phi}_k^0, \hat{\phi}_k^1\}$, $k = 1, 2, 3$. Depicted are the overall time-constant mean (solid line) $\pm 2\sqrt{\hat{\lambda}_k}$ times $\hat{\phi}_k^0$ or $\hat{\phi}_k^1$. Percentages give the proportion of overall variance that is explained by the respective component.

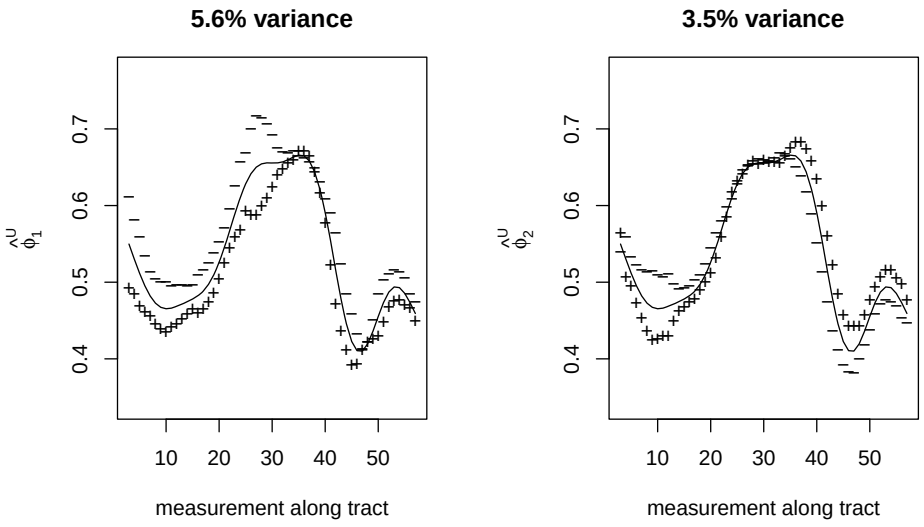

Figure 3: The first two functional principal components $\hat{\phi}_r^U$, $r = 1, 2$. Depicted are the overall time-constant mean (solid line) $\pm 2\sqrt{\hat{\nu}_r}$ times $\hat{\phi}_r^U$. Percentages give the proportion of overall variance that is explained by the respective component.

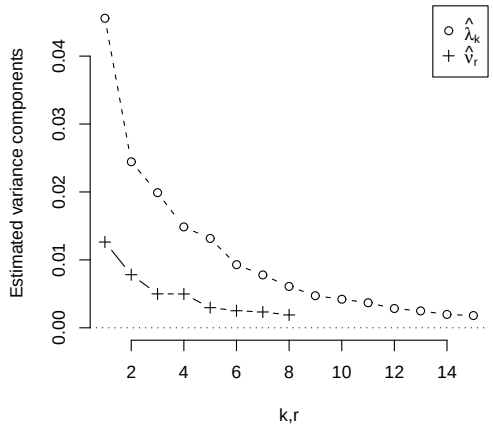

Figure 4: Estimated variance components $\hat{\lambda}_k$ and $\hat{\nu}_r$, $k = 1, \ldots, 15$, $r = 1, \ldots, 8$.

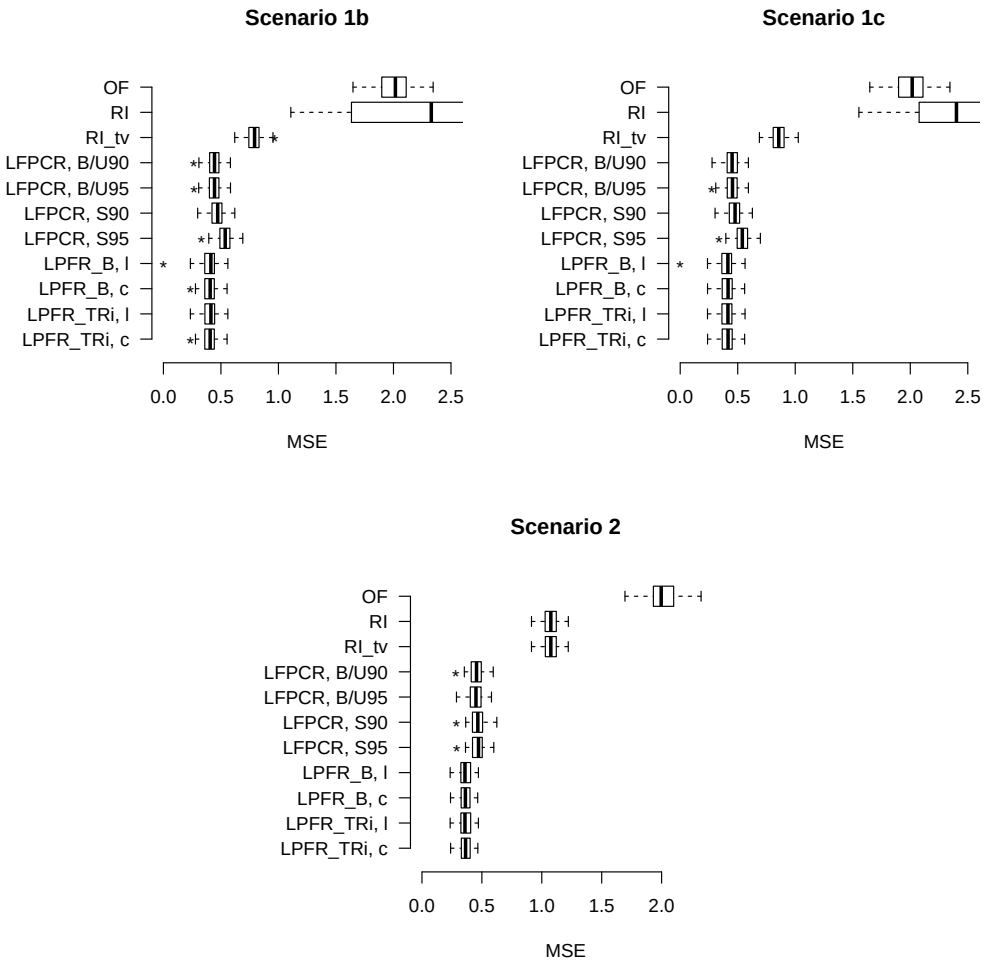

Figure 5: Results of simulation scenarios 1b, 1c, and 2 in terms of the (observed) mean squared error (MSE) $\frac{1}{n}\sum_{i,j}(\mu_{ij} - \hat{\mu}_{ij})^2$. Abbreviations are defined in the main paper (Gertheiss et al., 2012). Boxes for RI are not completely shown because a number of error values were too extreme.

Error $\varepsilon_{ij}(s)$ is just white noise. So if in the score-based LFPCR model besides X_{ijl} only ζ_{ij1} is used as predictor, we have

$$E(Y_{ij}|b_i, \tilde{b}_{i,1}, X_{ij1}, \ldots, X_{ij1}, \zeta_{ij1}) \approx h(\tilde{\alpha} + \tilde{b}_i + \sum_{l=1}^{p} X_{ijl}\beta_l + \int_{\mathcal{D}} \delta_1 W_{ij}(s)\phi_1^U(s)\, ds),$$

with $\tilde{\alpha} = \alpha + \delta_1 c_1$ and random intercept $\tilde{b}_i = b_i + \delta_1 \tilde{b}_{i,1}$ (with unknown distribution). That means, $\hat{\delta}_1 \hat{\phi}_1^U(s)$ can also be seen as an estimate of coefficient function $\gamma(s)$ in a functional linear model with random intercept. Note, however, only in special cases as above a regression model using scores from LFPCA as predictors can be transformed into a (generalized) functional linear model with scalar random effects as assumed when LPFR is applied.

- *Comparing LFPCR to Simple Mixed Models*: We consider the LFPCR model based on the B- and U-processes, and in particular

$$\begin{aligned} \int_{\mathcal{D}} \gamma_B(s) B_i(s, T_{ij}) ds &= \int_{\mathcal{D}} \gamma_B(s)(B_{i,0}(s) + T_{ij} B_{i,1}) ds \\ &= \int_{\mathcal{D}} \gamma_B(s) B_{i,0}(s) ds \; + \; T_{ij} \int_{\mathcal{D}} \gamma_B(s) B_{i,1}(s) ds \\ &= \mathfrak{b}_{i,0} + T_{ij} \mathfrak{b}_{i,1}. \end{aligned}$$

  If the process $U_{ij}(s)$ corresponds to functional measurement error and is not relevant for response Y_{ij}, we have $\gamma_U(s) \approx 0$ in the LFPCR model. Then, the LFPCR model can also be seen as a simple linear mixed model with random intercept $b_i + \mathfrak{b}_{i,0}$ and random slope $\mathfrak{b}_{i,1}$ with a modified distribution. This explains why in simulation scenario 5 (see the main paper) the simple random intercept model with time-varying α performs relatively well (the relatively small changes in $B_i(s, T)$ over time T cause that a random slope $\mathfrak{b}_{i,1}$ is not necessary here). However, if such a simple model is fit directly, substantial insight into dependencies between response and predictors will be lost.

Additional Information on the Analysis of Tractography Data

First, we show results for our regression model which have not been shown in the main paper (Gertheiss et al., 2012). As mentioned there, we consider the fractional anisotropy along the corpus callosum, the corticospinal tract that is contralateral to the dominant hand, and the optic radiations tract as potential functional predictors for the *peg9* score for the dominant hand. In addition to these functional predictors, we consider scalar covariates sex and age, and a dummy variable indicating whether it is the patient's first visit or not. We fit a generalized functional linear model with random intercept. Given the covariates and the random intercept, *peg9* scores are assumed to follow a Gamma distribution with log-link. Resulting estimates of coefficients curves are given in Figure 6. The solid black line is obtained when a penalized B-spline basis is used for the coefficient functions and deviations from a constant line are penalized. The red dashed line refers to the same penalty but using a truncated power spline basis with knots at each observation point. The blue dashed/dotted line is the `refund` (Crainiceanu

and Reiss, 2011) implementation `lpfr()` where deviations from a linear function are penalized. In the latter two cases missing observations are imputed as described in Gertheiss et al. (2012). In the first case (solid black), curves with missing values are omitted. The shaded region corresponds to 90% pointwise confidence intervals as provided by `mgcv` (Wood, 2006, 2011). Apparently there is only a dependence between measures along the corticospinal tract and *peg9*.

When using fractional anisotropy and magnetization transfer ratio along the corticospinal tract as functional predictors for *peg9*, predicted scores for LFPCR and a generalized functional linear model with random intercept are very similar, as mentioned in the main paper. Figure 8 shows corresponding scatterplots of fitted *peg9* scores, as well as plots of predictions with respect to the linear predictor for the used Gamma model with log-link.

In addition to MS patients, 49 healthy volunteers were also scanned, and we investigate differences between MS patients and controls. Since the case status does not change from visit to visit, we focus on the person-specific deviation from the overall trend $\eta(s,T)$. Since both the person-specific (functional) level $B_{i,0}$ and the person-specific trend $B_{i,1}$ may be interesting, we consider these two processes separately as predictors in a functional logistic regression model (with cases being coded as 1). Since these processes and class membership are observed only once for each person, observations are independent, and we do not include random effects in the linear predictor. However, we still correct for age and sex. LFPCA B_0- and B_1-processes of fractional anisotropy (FA) and magnetization transfer ratio (MTR) of the corpus callosum (CCA), the corticospinal tract (CST) and the optic radiations tract (OPR) are used as functional predictors. That means that a logit model with 12 functional and 2 scalar covariates is fit (using `mgcv`). Figure 8 shows the resulting estimated coefficient functions. When fitting these curves, we penalized deviations from a constant, and tuning parameters were determined by REML. Interpretation of the results, however, is difficult. Some curves (such as coefficient functions for B_0 of MTR/CST or FA/OPR) indicate that lower values of FA or MTR are associated with cases, but others do not. In addition, (90% pointwise) confidence intervals (as provided by `mgcv`) indicate that uncertainty is rather high. Apparently the relevant functional predictors are the B_0-processes of FA/CCA, MTR/CCA and MTR/CST, as well as the B_1-processes of FA/CST, FA/OPR and MTR/OPR.

Example R Code

This section contains some example R code that can be used to carry out the proposed longitudinal functional principal components regression (LFPCR) and longitudinal penalized functional regression (LPFR). At first, we need some packages

```
> library(mgcv)
> library(refund)
> library(orthopolynom)
```

For reproducibility, we set

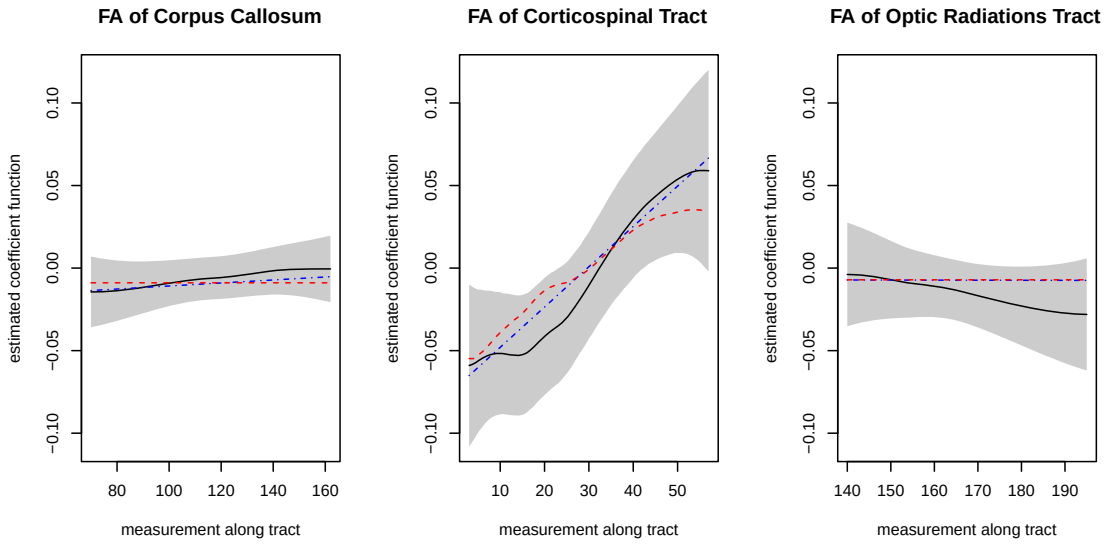

Figure 6: Estimated coefficient functions when a generalized functional linear model with random intercept is fit to data with functional predictors fractional anisotropy (FA) of the corpus callosum, corticospinal tract and optic radiations tract (and scalar predictors sex, age, and indicating if visit > 1).

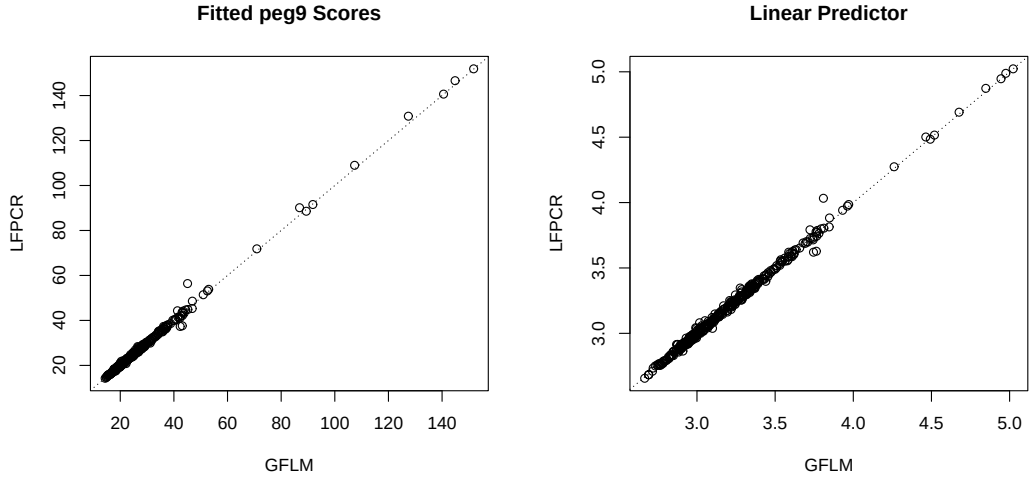

Figure 7: Predictions of *peg9* using LFPCR versus predictions of a generalized functional linear model (GFLM) with random intercept. Considered are fitted *peg9* scores (left) as well as predictions with respect to the linear predictor (right) for the used Gamma model with log link.

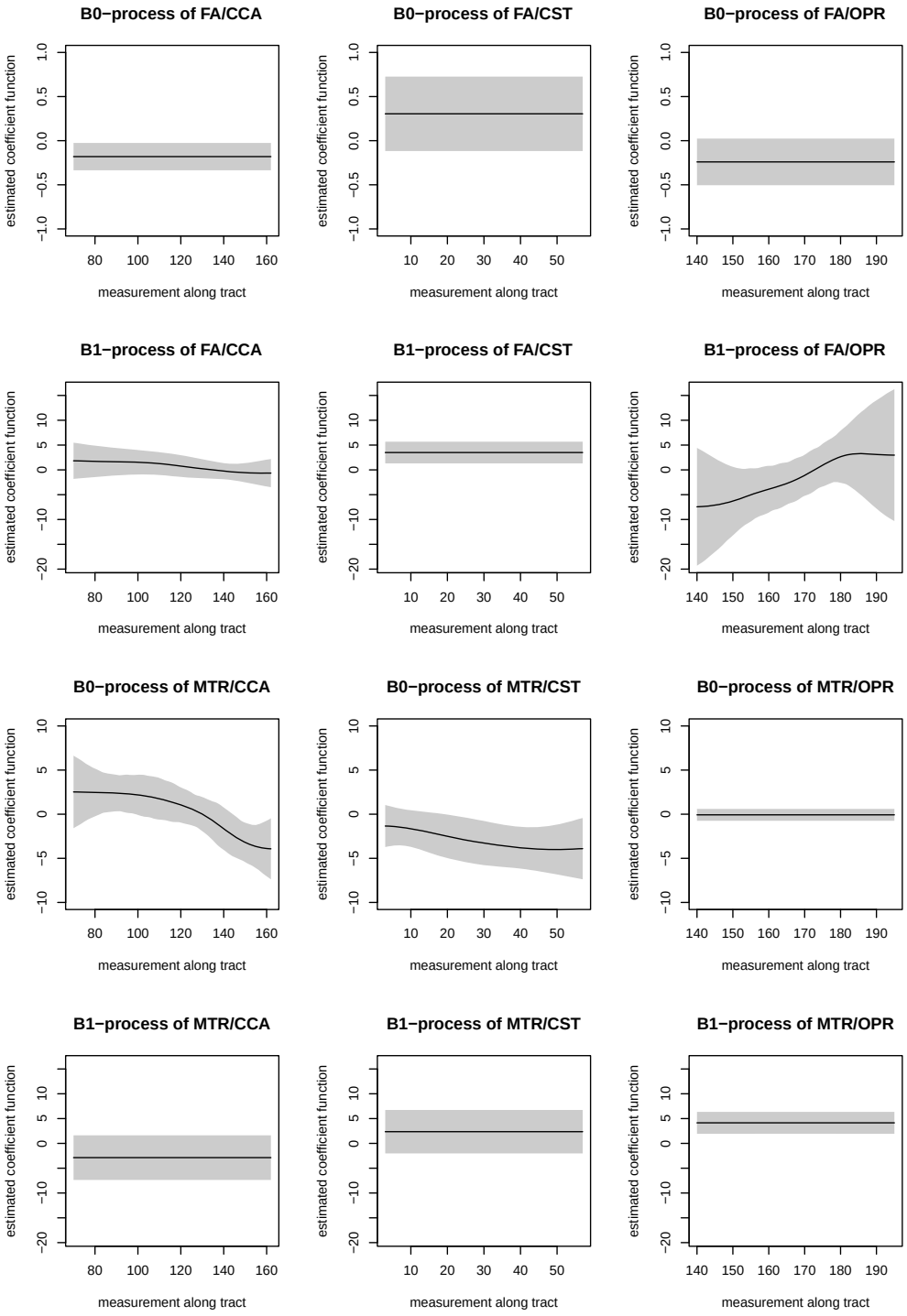

Figure 8: Estimated coefficient functions when a functional logistic regression model is fit to data with LFPCA B_0- and B_1-processes of fractional anisotropy (FA) and magnetization transfer ratio (MTR) of the corpus callosum (CCA), the corticospinal tract (CST) and the optic radiations tract (OPR) being used as functional predictors (and scalar predictors sex, age).

```
> set.seed(123)
```

For illustration, we write a function that uses the functional random intercept/slope model and the Karhunen-Loève expansion (assume normal ζ/ξ) to generate functional data. Arguments are:

```
     subj    vector of subjects (sorted)
     tps     grid on which functions are observed
     varx    measurement error variance
     eta     mean function eta(tps, times) with arguments tps and times
     phiU    eigenfunctions phiU(tps, k) with arguments tps and k
     phi0    eigenfunctions phi0(tps, k) with arguments tps and k
     phi1    eigenfunctions phi1(tps, k) with arguments tps and k
     times   vector of visit times
     nu      variances of the zeta scores
     lambda  variances of the xi scores
     ...     additional arguments to phiU, phi0, phi1
```

```
> simFREnor <- function(subj, tps, varx, eta, phiU, phi0, phi1 = NULL,
+     times, nu, lambda, ...) {
+     nI <- length(unique(subj))
+     n <- length(subj)
+     Jvec <- table(subj)
+     slope <- length(phi1) > 0
+     nU <- length(nu)
+     nX <- length(lambda)
+     etam <- t(outer(tps, times, eta))
+     phiUm <- matrix(NA, nU, length(tps))
+     phi0m <- matrix(NA, nX, length(tps))
+     if (slope)
+         phi1m <- phi0m
+     for (k in 1:nU) {
+         phiUm[k, ] <- phiU(tps, k, ...)
+     }
+     for (k in 1:nX) {
+         phi0m[k, ] <- phi0(tps, k, ...)
+         if (slope)
+             phi1m[k, ] <- phi1(tps, k, ...)
+     }
+     zeta <- matrix(rnorm(nU * n), n, nU) %*% diag(sqrt(nu))
+     xi <- matrix(rnorm(nX * nI), nI, nX) %*% diag(sqrt(lambda))
+     U <- zeta %*% phiUm
+     X0 <- xi[rep(1:nI, Jvec), ] %*% phi0m
+     if (slope)
+         X1 <- xi[rep(1:nI, Jvec), ] %*% phi1m
+     else X1 <- NULL
```

```
+     eps <- matrix(rnorm(length(tps) * n, 0, sqrt(varx)), n, length(tps))
+     Y <- etam + X0 + ifelse(slope, X1 * matrix(rep(times, length(tps)),
+         n, length(tps)), 0) + U
+     Yobs <- Y + eps
+     return(list(funx = Y, funcs = Yobs, zeta = zeta, xi = xi,
+         U = U, X0 = X0, X1 = X1))
+ }
```

Now we define functions `phi0`, `phi1`, `phiU`:

```
> phi0 <- function(tps, k, tmin = min(tps), tmax = max(tps)) {
+     tpsc <- 2 * pi * (tps - tmin)/(tmax - tmin)
+     if (k%%2 == 1)
+         result <- sin(ceiling(k/2) * tpsc)
+     else result <- cos((k/2) * tpsc)
+     return(result/sqrt(tmax - tmin))
+ }
> phi1 <- function(tps, k, tmin = min(tps), tmax = max(tps)) {
+     tpsc <- 2 * pi * (tps - tmin)/(tmax - tmin)
+     if (k == 1)
+         result <- rep(1/sqrt(2), length(tpsc))
+     else if (k%%2 == 1)
+         result <- cos((ceiling(k/2) + 1) * tpsc)
+     else result <- sin((k/2 + 2) * tpsc)
+     return(result/sqrt(tmax - tmin))
+ }
> phiU <- function(tps, k, tmin = min(tps), tmax = max(tps)) {
+     tpsc <- 2 * (tps - tmin)/(tmax - tmin) - 1
+     result <- as.function(legendre.polynomials(k, normalized = TRUE)[[k]])(tpsc)
+     return(result/sqrt((tmax - tmin)/2))
+ }
```

We are going to create data for an unbalanced design with 100 subjects, each with four visits on average.

```
> nI <- 100
> nV <- 4
```

We have one functional predictor with 100 measurement points.

```
> tps <- seq(0.5, 99.5, by = 1)
```

Measurement error and score variances are:

```
> varx <- 0.01
> lambda <- 2 * 0.5^((1:6) - 1)
> nu <- 0.5 * 0.5^((1:4) - 1)
```

We generate visit times.

```
> myfun <- function(j) {
+     times <- 0
+     if (j > 1) {
+         for (jj in 2:j) {
+             times <- cbind(times, sum(times) + runif(1))
+         }
+     }
+     return(times - mean(times))
+ }
```

The assumed η-function is:

```
> eta <- function(tps, times) {
+     (times/4 - tps/max(tps))^2/2
+ }
```

The true coefficient function is assumed to be:

```
> btrue <- 10 * dgamma(tps, 5, 1/5)
```

And for variances of the random intercept and the error term we set:

```
> vare <- 2
> vary <- 2
```

So now, we generate subjects:

```
> ncl <- c(1:nI, sample(1:nI, (nV - 1) * nI, replace = T))
> Jvec <- as.numeric(sort(table(ncl), decreasing = T))
> n <- sum(Jvec)
> subj <- rep(1:nI, Jvec)
```

Visit times with uniform increments:

```
> times <- unlist(lapply(table(subj), FUN = myfun))
> times <- times/sqrt(var(times))
```

We generate functional predictors:

```
> funData <- simFREnor(subj, tps, varx, eta, phiU, phi0, phi1,
+     times, nu, lambda, tmin = 0, tmax = 100)
> funcs <- funData$funcs
> funx <- funData$funx
```

The random intercept and response values according to the functional linear model (as assumed by LPFR):

```
> ref <- rep(rnorm(nI, 0, sqrt(vare)), Jvec)
> mutrue <- ref + funx %*% btrue
> y <- mutrue + rnorm(n, 0, sqrt(vary))
```

Now we start with the analysis. First we apply the LPFR function from the `refund` package.

```
> lpfrResult <- lpfr(Y = y, funcs = funcs, subj = subj)
```

Plot the result and compare it to the true function, or check the MSE:

```
> plot(tps, lpfrResult$BetaHat[[1]], type = "l")
> lines(tps, btrue, col = 2)
> mean((mutrue - lpfrResult$fitted.vals)^2)

[1] 0.5191954
```

Alternatively, function `gamm()` from `mgcv` can be used. For specifying the integral in functional linear model, the smooth term has to be chosen as `s(X,by=L)`, where the matrix `X` contains in its rows a grid of s-values $s_1, \ldots, s_m$, and the matrix `L` contains the functional predictor curves $W_{ij}(s)$ evaluated at $s_1, \ldots, s_m$, and multiplied by weights $\omega_1, \ldots, \omega_m$ for numerical integration, such that $\int_{\mathcal{D}} W_{ij}(s)\gamma(s)\,ds \approx \sum_{r=1}^{m} \omega_r W_{ij}(s_r)\gamma(s_r)$. In the simplest case with equidistant s_r and $s_r - s_{r-1} = 1$, we may use $\int_{\mathcal{D}} W_{ij}(s)\gamma(s)\,ds \approx \sum_{r=1}^{m} W_{ij}(s_r)\gamma(s_r)$. Coefficient function $\gamma(s)$ is typically fit as a spline. When `gamm()` is used, various bases can be chosen, such as cubic B-splines, and deviations from a constant line, a linear function or higher degree polynomials can be penalized. In addition, different types of random effects b_i can be included, including but not limited to random intercepts.

So for fitting the model, we need the matrix of measurement points.

```
> Tps <- matrix(tps, nrow(funcs), ncol(funcs), byrow = T)
```

Since points are equidistant here, we can easily fit the model. Concerning the estimation of the coefficient function, we use a P-spline approach and choose to penalize deviations from linearity (as with `refund`), i.e. `m = 2`, and from a constant line, i.e. `m = 1`.

```
> subjf <- as.factor(subj)
> gammResult1 <- gamm(y ~ s(Tps, by = funcs, bs = "ps", m = 1),
+     random = list(subjf = ~1), method = "REML")
> gammResult2 <- gamm(y ~ s(Tps, by = funcs, bs = "ps", m = 2),
+     random = list(subjf = ~1), method = "REML")
```

Again, plot the results and compare them to the true function:

```
> par(mfrow = c(1, 2))
> plot(gammResult1$gam)
> lines(tps, btrue, col = 2)
> plot(gammResult2$gam)
> lines(tps, btrue, col = 2)
```

Alternatively, we may also check the MSEs:

```
> mean((as.numeric(mutrue) - predict(gammResult1$lme))^2)
```

```
[1] 0.5198476
```

```
> mean((as.numeric(mutrue) - predict(gammResult2$lme))^2)
```

```
[1] 0.5192673
```

For longitudinal functional principal components regression (LFPCR) we first need the LFPCA R-function:

```
> source("http://www.statistik.lmu.de/institut/ag/fda/software/LFPCA.r")
```

Then we can carry out longitudinal functional principal components analysis, here with 90% variance explained.

```
> lfpca90 <- LFPCA(Y = funcs, subject = subj, Time = times, L = 0.9)
```

For the LFPCR approach that is based on scores, we first need to construct the matrix of new regressors.

```
> zetas <- lfpca90$zeta
> xis <- matrix(NA, length(subj), ncol(lfpca90$xi))
> for (i in unique(subj)) {
+     xis[subj == i, ] <- matrix(lfpca90$xi[unique(subj) == i,
+         ], sum(subj == i), ncol(lfpca90$xi), byrow = T)
+ }
```

Now we can fit a mixed model, for example a random intercept model:

```
> subjf <- as.factor(subj)
> scoresResult <- gamm(y ~ s(times, bs = "ps", m = 1) + xis + zetas,
+     random = list(subjf = ~1), method = "REML")
```

And check the MSE:

```
> mean((as.numeric(mutrue) - predict(scoresResult$lme))^2)
```

```
[1] 0.5865393
```

For the approach based on B- and U-processes, we need the corresponding matrices of (functional) covariates:

```
> B0 <- lfpca90$xi %*% t(lfpca90$phi.0)
> B1 <- lfpca90$xi %*% t(lfpca90$phi.1)
> B0x <- matrix(NA, length(subj), ncol(B0))
> for (i in unique(subj)) {
+     B0x[subj == i, ] <- matrix(B0[unique(subj) == i, ], sum(subj ==
+         i), ncol(B0), byrow = T)
+ }
> B1x <- matrix(NA, length(subj), ncol(B1))
> for (i in unique(subj)) {
```

```
+     B1x[subj == i, ] <- matrix(B1[unique(subj) == i, ], sum(subj ==
+         i), ncol(B1), byrow = T)
+ }
> Ux <- lfpca90$zeta %*% t(lfpca90$phi.U)
> Tps <- matrix(tps, nrow(Ux), ncol(Ux), byrow = T)
```

Then a mixed model can be fit by using `mgcv`:

```
> BUResult <- gamm(y ~ s(times, bs = "ps", m = 1) + s(Tps, by = (B0x +
+     B1x * times), bs = "ps", m = 1) + s(Tps, by = Ux, bs = "ps",
+     m = 1), random = list(subjf = ~1), method = "REML")
```

And the MSE is:

```
> mean((as.numeric(mutrue) - predict(BUResult$lme))^2)

[1] 0.5699787
```

References

Crainiceanu, C. M. and P. Reiss (2011). *refund: Regression with Functional Data*. R package version 0.1-5.

Gertheiss, J., J. Goldsmith, C. Crainiceanu, and S. Greven (2012). Longitudinal scalar-on-functions regression with application to tractography data. *preprint*.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. London: Chapman & Hall.

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society B 73*, 3–36.