Journal of Statistical Software

?? ??, Volume ??, Issue ??. http://www.jstatsoft.org/

Bayesian Functional Data Analysis using WinBUGS

Ciprian M. Crainiceanu and A. Jeffrey Goldsmith

Abstract

We provide user friendly software for Bayesian analysis of Functional Data Models using WinBUGS 1.4. The excellent properties of Bayesian analysis in this context are due to: 1) dimensionality reduction, which leads to low dimensional projection bases; 2) the mixed model representation of functional models, which provides a modular approach to model extension; and 3) the orthogonality of the principal component bases, which contributes to excellent chain convergence and mixing properties. Our paper provides one more, essential, reason for using Bayesian analysis for Functional models: the existence of software.

Keywords: MCMC, semiparametric regression.

1. Introduction

Functional data analysis (FDA) is an area of research concerned with the statistical analysis of functions. It is closely related to multivariate data analysis because functions are highly multivariate objects. The main distinction between FDA and multivariate analysis is the intrinsic ordering of observations; for example in time for time series or space for images. FDA is under intense methodological development: Chiou, Müller, and Wang (2003); James (2002); James, Hastie, and Sugar (2001); Müller and Stadtmüller (2005); Ramsay and Silverman (2006); Wang, Carroll, and Lin (1998); Yao, Müller, and Wang (2005); Yao and Lee (2006) are just a few examples of fundamental contributions in this area. Two comprehensive monographs of FDA with applications to curve and image analysis are Ramsay and Silverman (2005, 2006).

FDA was extended to multilevel functional data; see, for example, Baladandayuthapani, Mallick, Hong, Lupton, Turner, and Carroll (2008); Di, Crainiceanu, Caffo, and Punjabi (2009); Guo (2002); Morris, Vanucci, Brown, and Carroll (2003); Morris and Carroll (2006); Crainiceanu, Staicu, and Di (2009b); Staicu, Crainiceanu, and Carroll (2009). In this paper we will use the Di *et al.* (2009) approach to modeling multilevel functional data. This approach uses functional principal component bases to reduce data dimensionality and accelerate the associated algorithms, which is especially useful in moderate and large data sets.

A close inspection of various types of functional models reveals that they can all be viewed as mixed effects models James *et al.* (2001); James (2002); Morris and Carroll (2006); Di *et al.* (2009). Inferential methods for functional models range from using frequentist BLUP methods Yao *et al.* (2005); Yao and Lee (2006) to expectation maximization algorithms James *et al.* (2001); James (2002) to Bayesian simulations Morris *et al.* (2003); Morris and Carroll (2006); Crainiceanu *et al.* (2009b). However, Bayesian posterior inference has not become one of the standard inferential tools for Functional Data. Crainiceanu *et al.* (2009b) speculated that

> *Possible reasons for the current lack of Bayesian methodology in functional analysis could be: 1) the connection between functional models and joint mixed effects models was not known; and 2) the Bayesian inferential tools were perceived as unnecessarily complex and hard to implement.*

In this paper we will further clarify the connection between functional and mixed effects models. We will also disprove 2) by providing a simple combination of R and WinBUGS programs that make Bayesian simulations relatively painless. Arguing the necessity for Bayesian methods in general exceeds the scope of our paper; we defer instead to some excellent monographs describing the methodological and computational research conducted over the last 10-20 years. See, for example, Carlin and Louis (2000); Congdon (2003); Gelman, Carlin, Stern, and Rubin (2003); Gilks, Richardson, and Spiegelhalter (1996) and the citations therein for a good overview. Instead, we will argue that the excellent properties of Bayesian analysis in the context of Functional Data Analysis are due to: 1) dimensionality reduction, which leads to low dimensional projection bases; 2) the mixed model representation of functional models, which provides a modular approach to model extension; and 3) the orthogonality of the principal component bases, which contributes to excellent chain convergence and mixing properties.

The paper describes methods and software for a variety of functional models. Section 2 is focused on the Functional Principal Component Analysis of functional data observed at one visit. Section 3 is concerned with functional regression, that is regression where either the covariate or the outcome is a function. Section 4 is focused on the Multilevel Functional Principal Component Analysis of data observed at multiple visits. Relevant features of the WinBUGS 1.4 programs are described throughout the paper, the entire WinBUGS 1.4 programs are provided in the Appendices. R and WinBUGS 1.4 programs can be downloaded at

http://www.biostat.jhsph.edu/~ccrainic/webpage/software/Bayes_FDA.zip

2. Single-level modeling of functional data

Consider, for illustration, data from the Sleep Heart Health Study (SHHS). SHHS is a multi-center study of sleep disrupted breathing, hypertension and cardiovascular disease Quan, Howard, Kiley, Nieto, and O'Connor (1997). Here we focus on modeling a particular characteristic of the spectrum of the Electroencephalograph (EEG) data, the proportion of δ-power. For more details on the definition and interpretation of δ-power see, for example, Crainiceanu, Caffo, and Punjabi (2009a); Di *et al.* (2009); Crainiceanu *et al.* (2009b). In our context, it is sufficient to know that percent δ-power is a summary measure of the spectral representation of the EEG signal; in this paper percent we use percent δ-power calculated in 30-second intervals. Figure 1 displays the sleep EEG proportion of δ-power in each of 30-second interval for 3 subjects at the baseline visit. The x axis represents time in hours since sleep onset and the y axis represents the estimated proportion of δ-power. Observations are shown in adjacent 30-second intervals with missing observations indicating wake periods. For interpretation, δ-power corresponds to low-frequency neuronal firing and a higher value of an observation corresponds to lower frequency average neuron firing in a particular interval. Data are shown from sleep onset, when neuronal firing tends to contain higher frequencies, which corresponds to lower percent δ-power. As subjects go through deeper stages of sleep, neuronal firing slows down and contains more lower frequencies, which corresponds to higher values of percent δ-power. Please note the increase in average percent δ-power in the first 30-35 minutes of sleep. As time from sleep onset passes, the subject-specific percent δ-power functions become more and more de-synchronized; this results in a flattening of the population average, even though

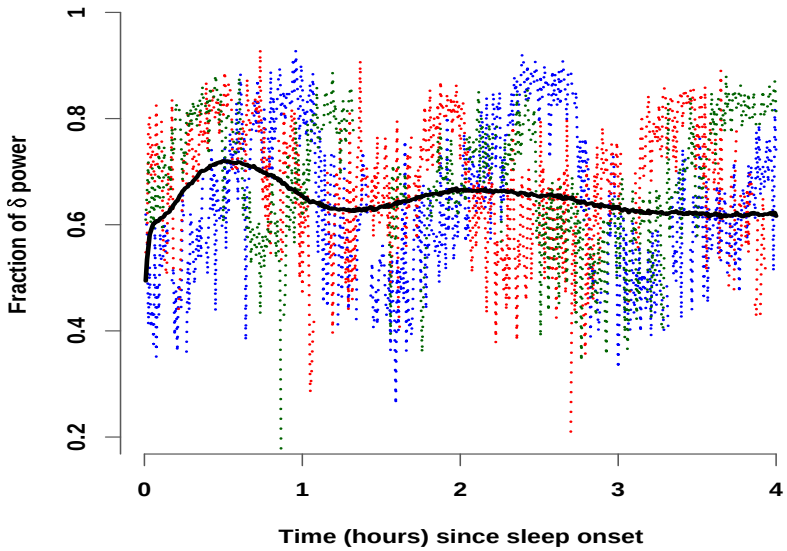

Figure 1: Percent δ-power since sleep onset (time=0). Data shown are for the first 4 hours of sleep of 3 subjects (dotted lines) at the baseline visit. The average over $3,040$ subjects is the black solid line.

the variability of the subject-specific curves around the population curve does not change. The SHHS contains more than $3,000$ subjects with baseline and visit 2 sleep EEG data.

In this Section we describe Bayesian methods for the analysis of a sample of curves observed at one visit. The basic idea is to decompose each subject specific curve into a population average, a subject-specific deviation from the population average and measurement error. The subject-specific deviations are modeled by projecting them on a small number of eigenvectors of the covariance matrix of the sample of curves. Using the mixed model formulation of the underlying model we obtain the joint posterior distribution of all parameters given the data. The parameter space includes all subject-specific functions and their individual scores. Methods in this section can be applied to sparse or dense functional data.

2.1. Functional Principal Component Analysis

We focus on the first hour of sleep EEG data for 500 subjects. Denote the observed EEG fraction of δ-power by $W_i(t)$, for subject $i = 1, \ldots, I = 500$, at the 30-second interval $t = 1, \ldots, T = 120$. In this data set 14% of observations are missing because wake periods are

removed. Wake periods are contiguous, have random lengths and appear at random times. Let $X_i(t)$ be the true EEG fraction of δ-power and assume that $W_i(t)$ is the functional proxy for $X_i(t)$ and that they are related via the following functional measurement error model

$$W_i(t) = \mu(t) + X_i(t) + \epsilon_i(t) \tag{1}$$

where $\epsilon_i(t)$ is a white noise process and $X_i(t)$ is a realization of a mean-zero stochastic process with covariance operator $K_X(t,s) = \text{cov}\{X_i(t), X_i(s)\}$. Using methods method of moments (MoM) and smoothing Di *et al.* (2009), we obtain a smooth estimator $\widehat{K}_X(t,s)$ and its corresponding eigenfunctions, $\psi_k(\cdot)$, $k = 1, \ldots, T$. Table 1 provides the first 10 eigenvalues and associated levels of variance explained for $\widehat{K}_X(t,s)$ indicating that more than 95% of the observed functional variability is associated with the first 6 eigenfunctions. Figure 2 displays the first two eigenfunctions (left panels) and the deformations from the population mean in the positive and negative directions of the eigenfunctions.

Staicu *et al.* (2009) and Crainiceanu *et al.* (2009b) point out that choosing the number of eigenfunctions corresponds to step-wise testing for zero variance components. They propose using a Restricted Likelihood Ratio Test (RLRT) for this zero variance. The null distribution can be easily approximated using methods introduced by Greven, Crainiceanu, Küchenhoff, and Peters (2008) based on the null distribution derived in Crainiceanu and Ruppert (2004) and Crainiceanu, Ruppert, Claeskens, and Wand (2005).

	Eigenvalues									
	1	2	3	4	5	6	7	8	9	10
Var	0.99	0.31	0.18	0.10	0.06	0.04	0.03	0.02	0.02	0.01
% Var	56.4	17.8	10.3	5.4	3.3	2.2	1.4	1.2	1.0	0.6
sum % Var	56.4	74.2	84.5	90.0	93.3	95.5	96.9	98.1	99.1	99.8

Table 1: Estimated eigenvalues for the percent δ power using data for the first hour of sleep during the baseline visit of 500 subjects from the SHHS.

We will retain the first $K = 10$ eigenfunctions and consider the case when $\mu(t) = 0$. This can be achieved by replacing $W_i(t)$ by $W_i(t) - \hat{\mu}(t)$, where $\hat{\mu}(t) = \sum_{i=1}^{I} W_i(t)/I$. The functional model (1) becomes

$$\begin{cases} W_i(t) & = & \sum_{k=1}^{K} \xi_{ik}\psi_k(t) + \epsilon_i(t); \\ \xi_{ik} & \sim & \text{N}(0, \lambda_k); \quad \epsilon_i(t) \sim \text{N}(0, \sigma_\epsilon^2), \end{cases} \tag{2}$$

which is a mixed effects model. We now show how to implement this model in WinBUGS 1.4.

To completely specify the Bayesian model, one needs to provide prior distributions for all model parameters. We used independent Gamma$(10^{-3}, 10^{-3})$ priors for $\sigma_\epsilon^2, \sigma_k^2$ for $k = 1, \ldots, K = 10$. The parameterization of the Gamma(a, b) distribution is chosen so that its mean is $a/b = 1$ and its variance is $a/b^2 = 10^3$. These choices are reasonable as the MoM estimators of σ_ϵ^2 and $\lambda_{K=10}$ were $6.7 * 10^{-3}$ and $1.16 * 10^{-2}$, respectively (see, for example, (Crainiceanu, Ruppert, Carroll, Adarsh, and Goodner 2007) for a thorough discussion on Gamma priors). In Bayesian models, the estimates of the variance components are known to be sensitive to the prior specification; see Gelman (2006). Alternative to gamma priors are discussed by, for example, Gelman (2006) and Natarajan and Kass (2000). These have the advantage of requiring less care in the choice of the hyperparameters. However, we find that with reasonable care, the conjugate gamma priors can be used in practice. Nonetheless, exploration of other prior families for functional data analysis would be well worthwhile, though beyond the scope of this paper.

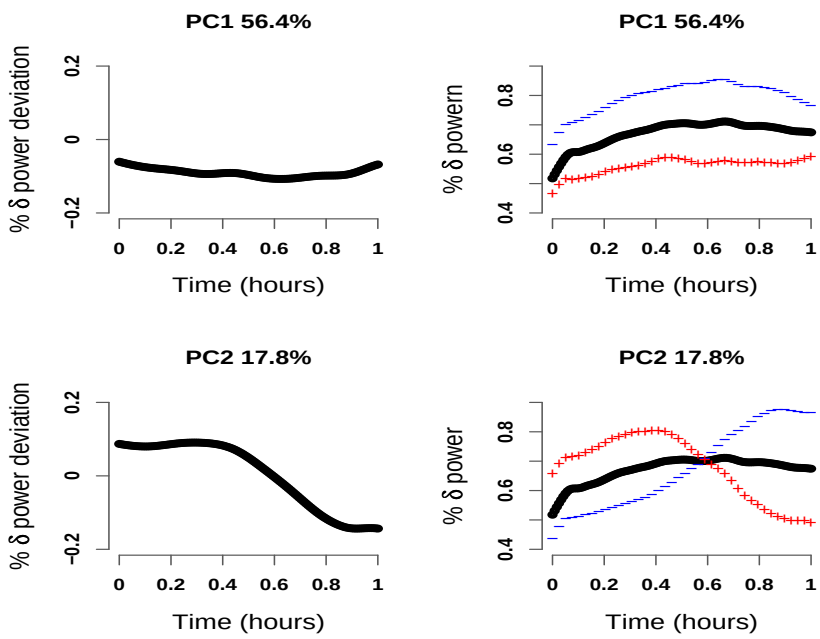

Figure 2: Left panels: First and second eigenvectors, $\psi_1(t)$ and $\psi_2(t)$, for the first hour of sleep percent δ power data on 500 subjects. Right panels: The positive ("+" sign) and negative ("-" sign) deformation from the population mean in the direction of the corresponding eigenfunction.

2.2. WinBUGS program for the single-level exposure model

We now describe the WinBUGS 1.4 program that follows closely the description of the Bayesian Functional Principal Component Analysis (FPCA) model (2). We provide the entire program in the Appendix A1. While the program was designed for the SHHS data, it can be used for other FPCA with only minor adjustments. Many features of this program will be repeated in the other examples in this paper and changes will be described, as needed.

Model (2) describes the core components (likelihood and shrinkage assumptions) of FPCA and is specified in WinBUGS 1.4 as follows

```
for (i in 1:N_subj)
    {for (t in 1:N_obs)
        {W[i,t]~dnorm(X[i,t],tau_eps)
         X[i,t]<-xi[i,1]*psi[t,1]+xi[i,2]*psi[t,2]+xi[i,3]*psi[t,3]+
                 xi[i,4]*psi[t,4]+xi[i,5]*psi[t,5]+xi[i,6]*psi[t,6]+
                 xi[i,7]*psi[t,7]+xi[i,8]*psi[t,8]+xi[i,9]*psi[t,9]+
                 xi[i,10]*psi[t,10]}

     for (k in 1:dim.space)
        {xi[i,k]~dnorm(0,tau_lambda[k])}
    }
```

This part of the program describes a double loop over the subjects, `for (i in 1:N_subj)`, and over the number of observations within subjects, `for (t in 1:N_obs)`. The number of subjects, `N_subj`, is a constant in the program and is equal to 500. The number of observations within subject, `N_obs`, is also a constant and is equal to 120, which corresponds to the number of 30-second intervals in one hour. Note that, in general, the number of observations for each subject is smaller than 120 due to missing observations. However, missing observations are treated as random and are estimated like all the other unknowns in the model.

The first statement specifies that the $W_i(t)$, the observed percent EEG δ-power, has a normal distribution with mean $X_i(t)$, the true percent EEG δ-power, and precision $\tau_\epsilon = \sigma_\epsilon^{-2}$. The second statement provides the structure of the conditional mean function, $X_i(t)$. Here `psi[t,k]` denotes the kth eigenfunction evaluated at time t, $\psi_k(t)$. All eigenfunctions are

obtained from the diagonalization procedure of the smooth estimator $\widehat{K}_X(t,s)$ of $K_X(t,s)$ and are treated as data. Also, `xi[i,k]` corresponds to ξ_{ik} in model (2), which is the score of the subject i-specific function on the kth eigenfunction, $\psi_k(t)$. Thus, `xi[,]` is a $I \times K$ dimensional matrix of random parameters, whose joint posterior distribution is the main target of inference. The third statement specifies that ξ_{ik} for $i = 1, \ldots, I$, the scores of subjects on principal component k, have a normal distribution with mean 0 and component k-specific precision $\tau_{k,\lambda} = \lambda_k^{-1}$. The matrices `W[,]` and `psi[,]` are $I \times T$ and $T \times K$ dimensional, respectively, are obtained outside WinBUGS 1.4 and are entered as data. The software accompanying this paper contains an auxiliary R program that calculates these matrices and uses the **R2WinBUGS** package to call WinBUGS 1.4 from R. The formulae for `X[i,t]` could be shortened using the inner product function `inprod`. However, depending on the application, computation time can be 5 times longer when `inprod` is used.

The model continues with the prior specifications for the precision parameters, $\tau_k = \sigma_k^{-2}$. These parameters were already estimated as the eigenvalues of the method of moments estimator $\widehat{K}_X(t,s)$ and could be embedded as constants. Here we estimate them again by using uninformative gamma priors.

```
for (k in 1:dim.space)
    {tau_lambda[k]~dgamma(1.0E-3,1.0E-3)
     lambda[k]<-1/ll[k]}
```

2.3. Results

We obtained $1,500$ simulations, discarded the first 500 as burn-in and used the remainder $1,000$ for inference. For $I = 500$ subjects and $T = 120$ grid points for each function the total computation time was 4.8 minutes (Dual Core Processor 3GHz, 8Mb RAM PC). This number of simulations was enough for our purposes because convergence and mixing of the chains was excellent. Indeed, Figure 3 displays the un-thinned histories for 4 chains corresponding to two variance components and two subject-specific deviations from the population mean. The independence-like behavior of the chains is due to the orthogonality of the functional basis, $\psi_k(\cdot)$.

Chain properties, such as convergence and mixing are crucial in Bayesian analysis based on posterior simulations. Indeed, if the chain does not converge or converges very slowly to the

target distribution then inferences based on these chains may be unreliable. Poor mixing may be due to a variety of factors, all of them undesirable: inadequate model parameterization, unidentifiable or nearly unidentifiable models, poor performance of simulation algorithms, wrong implementation, etc. Poor mixing is one of the most haunting problems of modern Bayesian computational problems, with hundreds of papers dedicated to improving mixing behavior. In practice, we found that convergence of chains is best illustrated by running multiple chains from over-dispersed initial values with respect to the target distribution and visually inspect when chains have converged. A more formal approach would be to monitor the Gelman and Rubin (1992) statistic Gelman and Rubin. In our case convergence is basically instantaneous and simply removing the first 500 simulations as burn-in is more than enough. Mixing is typically assessed by visual inspection, by calculating the autocorrelation function or the Monte Carlo error. Visual inspection of chains indicated that our chains are very close to independence sampling.

The orthogonality of the principal component basis, the data reduction and the mixed model representation of functional models make Bayesian analysis particularly appealing for functional data analysis.

Bayesian posterior simulations provide the joint posterior distribution of all parameters given the data. Note that the subject-specific functional random effects, $X_i(t) = \sum_{k=1}^{K} \xi_{ik}\psi_k(t)$, are explicit functions of the model parameters, ξ_{ik}; thus, one can obtain the joint posterior distribution of all $X_i(t)$, for every t. Figure 4 displays the data (black dots) for 4 subjects together with the posterior means (solid black line) and 95% point-wise credible intervals (shaded areas) of the subject-specific mean, $\mu(t) + X_i(t)$. Constructing credible or confidence intervals is considered to be technically difficult but important; using the Bayesian methods in this paper this can be done relatively easily for data sets with a large number of subjects. We are not aware of any other published result that allows this.

3. Functional Regression Models

In this section we describe Bayesian analysis of three types of functional regression. First, we introduce the classical functional regression model where an outcome is regressed on a functional predictor. This is done by adding a regression model to the functional models introduced in Section 2, where the outcome is regressed on the functional scores and other

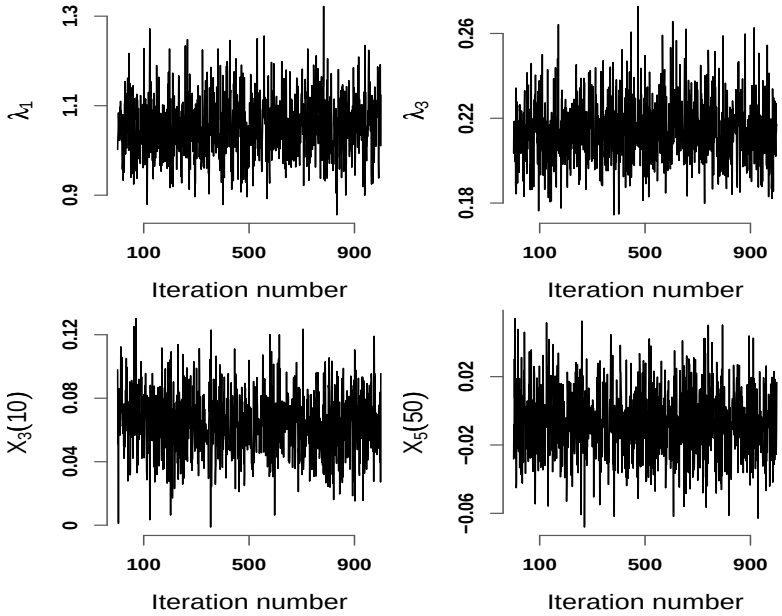

Figure 3: $1,000$ un-thinned draws from the posterior distributions for two variance components, λ_1 and λ_3, and two subject-specific deviations, $X_3(\cdot)$ and $X_5(\cdot)$, evaluated at two different time points, $t = 10$ and $t = 50$, respectively.

covariates. The regression and the functional model are fitted jointly, which correctly accounts for the uncertainty of functional estimators. Second, we introduce the Bayesian penalized B-splines model to estimate the functional coefficient. This is an alternative to the classical functional regression model that uses a large number of principal components to estimate the subject-specific functions and a spline penalty to control the amount of smoothing of the functional parameter. This method avoids the typical discussions about choosing the right number of principal components. Third, we introduce the case when functional scores are regressed on other covariates. This is done by adding a regression model to the functional models introduced in Section 2, where the outcome is one of the functional scores and the regressors are other covariates. Models are fitted jointly to correctly incorporate the variability of the unknowns.

3.1. Classical Functional Regression Model

A particularly useful class of models that describe associations between non-gaussian outcomes and functional data is the class of generalized functional linear models (GFLM) Müller

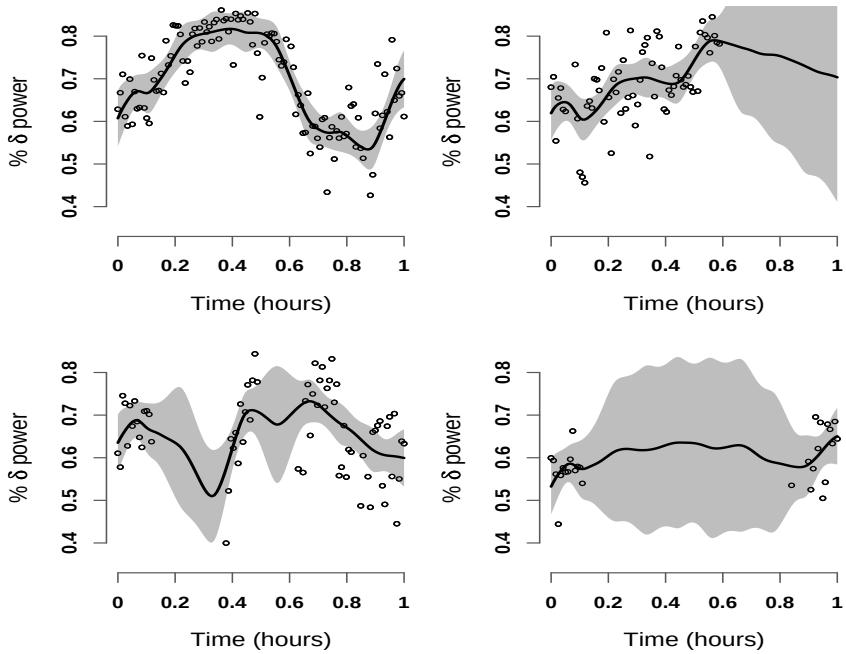

Figure 4: Estimated subject-specific means with 95% point-wise credible intervals for the EEG normalized δ power in the first hour of sleep of 4 subjects from the SHHS.

and Stadtmüller (2005); Cardot, Ferraty, and Sarda (1999, 2003); Reiss and Ogden (2007); Ramsay and Silverman (2006, 2005). The observed data for the ith subject in a GFLM is $[Y_i, \boldsymbol{Z}_i, \{W_i(t_{im}), t_{im} \in [0,1]\}]$, where Y_i is the continuous or discrete outcome, $\boldsymbol{Z}_i$ is a vector of covariates, and $W_i(t_{im})$ is a random curve in $L_2[0,1]$ observed at time t_{im}, which is the mth observation, $j = 1, \ldots, M_i$, for the ith subject, $i = 1, \ldots, n$. We assume that $W_i(t)$ is a proxy observation of the true underlying functional signal $X_i(t)$ and that $W_i(t) = \mu(t) + X_i(t) + \epsilon_i(t)$, where $\mu(t)$ is the population average and $\epsilon_i(t)$ is a mean zero white noise process with variance σ_ϵ^2. We also assume that the distribution of Y_i is in the exponential family with linear predictor ϑ_i and dispersion parameter a, denoted here by $\mathrm{EF}(\vartheta_i, a)$. The linear predictor is assumed to have the following form

$$\vartheta_i = \int_0^1 X_i(t)\beta(t)dt + \boldsymbol{Z}_i^t\boldsymbol{\gamma}, \tag{3}$$

where $\beta(\cdot) \in L_2[0,1]$ is a functional parameter and the main target of inference. Note that if $\{\psi_k(\cdot), k \geq 1\}$ is an orthonormal basis in $L_2[0,1]$ then both $X_i(\cdot)$ and $\beta(\cdot)$ have unique representations $X_i(t) = \sum_{k\geq 1} \xi_{ik}\psi_k(t)$, $\beta(t) = \sum_{k\geq 1} \beta_k\psi_k(t)$ and equation (3) can be rewritten as $\vartheta_i = \sum_{k\geq 1} \xi_{ik}\beta_k + \boldsymbol{Z}_i^t\boldsymbol{\gamma}$. Thus, conditional on the eigenfunctions $\psi_k(t)$, $k = 1, \ldots, K$, and on

the number of eigenfunctions, K, the standard functional regression model can be re-written as the following mixed effects model

$$\begin{cases} Y_i & \sim & \mathrm{EF}(\vartheta_i, a); \\ \vartheta_i & = & \sum_{k\geq 1} \xi_{ik}\beta_k + \boldsymbol{Z}_i^t\boldsymbol{\gamma}; \\ W_i(t) & = & \sum_{k=1}^{K} \xi_{ik}\psi_k(t) + \epsilon_i(t); \\ \xi_{ik} & \sim & \mathrm{N}(0, \lambda_k); \;\; \epsilon_i(t) \sim \mathrm{N}(0, \sigma_\epsilon^2). \end{cases} \tag{4}$$

The first line of model (4) describes the distribution of the outcome, where $\mathrm{EF}(\vartheta_i, a)$ is an exponential family distribution with linear predictor ϑ_i and dispersion parameter a. The second line describes the structure of the linear predictor which contains the functional scores, ξ_{ik}, and other covariates, $\boldsymbol{Z}_i$, as regressors. The scores ξ_{ik} are not directly observable, but can be estimated from the functional exposure model described in the third line and the distributional assumptions in the fourth line. The difference between model (4) and a standard generalized linear mixed model (GLMM) is that in a GLMM the random effects are random parameters for known regressors. In model (4) the random effects are the regressors in the linear predictor ϑ_i.

Changes to the WinBUGS 1.4 program for the classical functional regression model. We provide the entire program in the Appendix A2. For illustration, consider the case where a binary outcome is regressed on the functional scores on the first three eigenfunctions. That is, the functional predictor has the form

$$\vartheta_i = \mu + \xi_{i1}\beta_1 + \xi_{i2}\beta_2 + \xi_{i3}\beta_3. \tag{5}$$

This additional regression level is included in the WinBUGS 1.4 program by simply adding the following code describing the logistic model for the outcome to the WinBUGS 1.4 code described in Section 2.2

```
for (i in 1:N_subj)
    {Y[i]~dbern(pY[i])
    logit(pY[i])<-mu+beta[1]*xi[i,1]+beta[2]*xi[i,2]+beta[3]*xi[i,3]}
```

One also needs to specify the prior distributions for $\mu, \beta_1, \beta_2, \beta_3$. This is done as

```
for (j in 1:3){beta[j]~dnorm(0,1.0E-2)}


mu~dnorm(0,1.0E-2)
```

Extending this regression model to include more scores or other covariates is straightforward by simply adding terms to `logit(pY[i])`.

Results. We applied the functional regression model for regressing hypertension status, a binary variable, on the scores of the first three principal components. We used model the functional model (4) with the particular form of the linear predictor given in (5). We used the first hour of normalized EEG δ-power for 500 subjects as functional predictors. Table 2 provides the posterior mean and standard deviation of the parameters μ, β_1, β_2 and β_3.

	μ	β_1	β_2	β_3
mean	-0.25	0.14	-0.16	0.30
St. Dev.	0.09	0.09	0.16	0.21

Table 2: Estimated functional effects for sleep EEG normalized δ-power on Hypertension. Results shown for 500 subjects and first hour of sleep.

3.2. Functional Regression Using Penalized B-Splines

An important approach to GFLMs uses a penalized spline expansion for the coefficient function O'Sullivan (1986); Cardot *et al.* (2003); Cardot and Sarda (2005); Ruppert, Carroll, and Wand (2003); Wood (2006). As in Section 3.1, assume that for the ith subject we observe $[Y_i, Z_i, \{W_i(t_{im}), t_{im} \in [0,1]\}]$ where $W_i(t)$ is a measured-with-error proxy for the true subject-specific function $X_i(t)$, that $Y_i \sim \mathrm{EF}(\vartheta_i, a)$, and that the linear predictor ϑ_i has the form given in (3). Rather than expressing $\beta(\cdot)$ in terms of the eigenfunctions $\psi_k(t)$, we use a cubic B-spline basis $\{\phi_l(t) : 1 \le l \le L\}$ with equally spaced knots, so that $\beta(t) = \sum_{l=1}^{L} \beta_l \phi_l(t)$; other spline bases could be used, but the parameters of the B-spline have good mixing properties in a Bayesian posterior simulation context. Given the truncation lag K and the identifiability constraint $L \le K$, the linear predictor has the form

$$\begin{aligned} \vartheta_i &= \int_0^1 \{\textstyle\sum_{k=1}^{K} \xi_{ik}\psi_k(t)\}\{\sum_{l=1}^{L} \beta_l \phi_l(t)\}dt + Z_i\gamma \\ &= \xi_i \boldsymbol{J}\boldsymbol{\beta}^T + \boldsymbol{Z}_i\gamma, \end{aligned}$$

where $\xi_i = [\xi_{i1}, ..., \xi_{iK}]^T$, $\boldsymbol{\beta} = [\beta_1, ..., \beta_L]^T$ and $\boldsymbol{J}$ is a $K \times L$ dimensional matrix with the (k,l)th entry equal to $\int_0^1 \psi_k(t)\phi_l(t)dt$. These integrals are computed by numeric integration and are fixed for the purpose of this paper. WinBUGS 1.4 loads the matrix $\boldsymbol{J}$ as data.

In this paper we use B-splines, which are a flexible basis commonly used in smoothing applications; Figure 5 shows a cubic B-spline basis with 10 functions and 6 equally spaced

knots. Smoothness on $\beta(t)$ is typically induced by assuming a random walk prior on the β_l, $l = 1, \ldots, L$ parameters Brezger, Kneib, and Lang (2005); Lang and Brezger (2004). We use a first order random prior

$$\beta_{l+1} \sim N(\beta_l, \sigma_\beta^2), l = 1, \ldots, L-1,$$

where β_1 is treated as a fixed unknown parameter. A second order random walk prior is

$$\beta_{l+2} \sim N(2\beta_{l+1} - \beta_l, \sigma_\beta^2), l = 1, \ldots, L-2,$$

where β_1 and β_2 are treated as a fixed unknown parameter. Both priors are easy to implement in WinBUGS 1.4 and we choose to use the first order random walk prior.

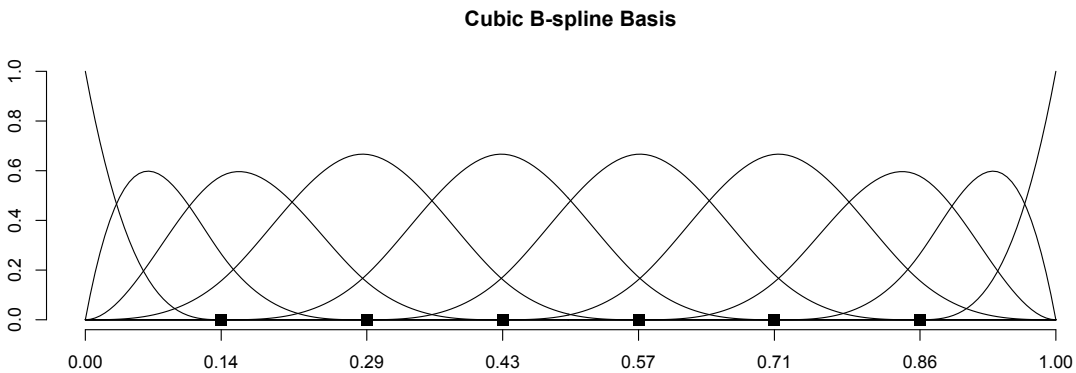

Figure 5: Cubic B-spline basis with 10 functions and 6 equally spaced knots, with knots marked by solid squares.

Thus, the functional regression model using penalized B-splines has the mixed model representation

$$\begin{cases} Y_i & \sim & EF(\vartheta_i, a); \\ \vartheta_i & = & \xi_i \boldsymbol{J}\boldsymbol{\beta}^T + \boldsymbol{Z}_i\gamma; \\ W_i(t) & = & \sum_{k=1}^K \xi_{ik}\psi_k(t) + \epsilon_i(t); \\ \xi_{ik} & \sim & N(0, \lambda_k);\ \epsilon_i(t) \sim N(0, \sigma_\epsilon^2); \\ \beta_{l+1} & \sim & N(\beta_l, \sigma_\beta^2), l = 1, \ldots, L-1. \end{cases} \tag{6}$$

Changes to the WinBUGS 1.4 program for the penalized B-spline functional regression. We provide the entire program in the Appendix A3. We describe the changes in the WinBUGS 1.4 program to account for the different parameterization of ϑ_i. Here, we choose $K = L = 10$; straightforward changes to the code below allow for larger values of both. First, we use the following code to define the functional predictor for each subject:

```
logit(pY[i])<-mu[i]
     mu[i]<-mp+eta[i]
     eta[i]<-xi[i,1]*gamma[1]+xi[i,2]*gamma[2]+xi[i,3]*gamma[3]+
          xi[i,4]*gamma[4]+xi[i,5]*gamma[5]+xi[i,6]*gamma[6]+
          xi[i,7]*gamma[7]+xi[i,8]*gamma[8]+xi[i,9]*gamma[9]+
          xi[i,10]*gamma[10]
```

Here, gamma is the $L \times 1$ vector $\mathbf{J}\beta^{\mathbf{T}}$. This vector is updated once per iteration, and is computed outside the loop over subjects in the following code:

```
for(l in 1:L){
     gamma[l]<-J[l,1]*beta[1]+J[l,2]*beta[2]+J[l,3]*beta[3]+
          J[l,4]*beta[4]+J[l,5]*beta[5]+J[l,6]*beta[6]+
          J[l,7]*beta[7]+J[l,8]*beta[8]+J[l,9]*beta[9]+
          J[l,10]*beta[10]
}
```

The first order random walk prior is specified as

```
for (l in 2:L)
     {beta[l]~dnorm(beta[l-1],taubeta)}
      beta[1]~dnorm(0,1.0E-6)
      taubeta~dgamma(1.0E-3,1.0E-3)
```

Results. Figure 6 shows the estimate of the functional coefficient $\beta(t)$ in the linear predictor (3) using the penalized B-spline approach (blue solid line); for reference, we also plot the estimate from the classical functional regression model presented in Section 3.1 (red solid line). In the Bayesian context obtaining credible intervals for the functional parameter is straightforward by simply monitoring $\boldsymbol{\beta}$. Figure 6 displays the pointwise 95% credible interval for the estimated coefficient function, indicating that in this example there is not much evidence of statistical significance. Also, the classical and the penalized spline regression provide similar results, but this need not be the case in general Goldsmith, Feder, and Crainiceanu (2010).

3.3. Regression with functional scores as outcomes

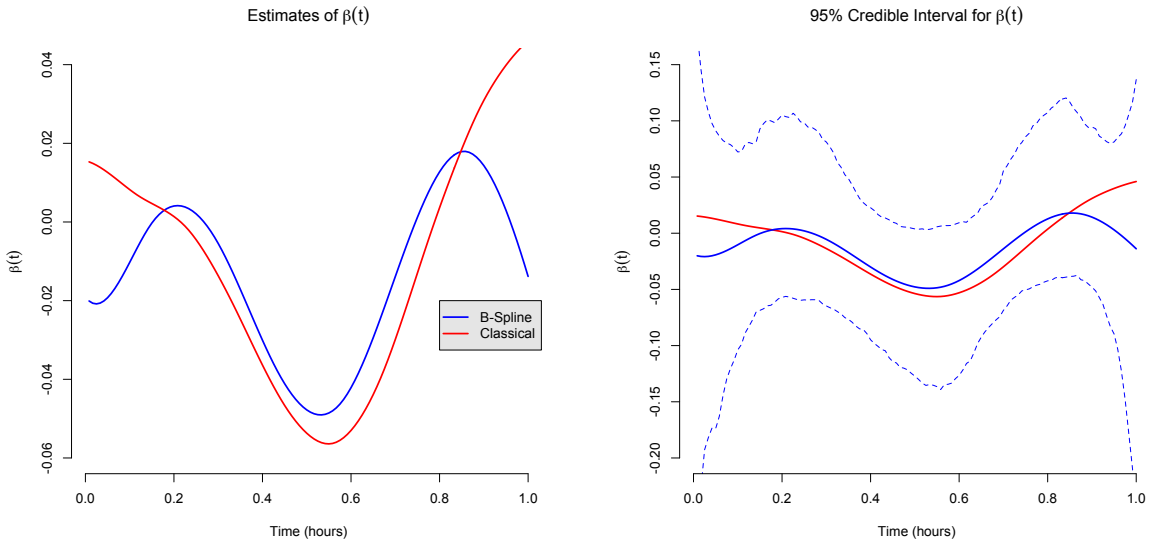

Figure 6: The left panel shows two estimates of the functional coefficient $\beta(t)$, one from the Classical Functional Regression Model (red solid line) and the other from the Penalized B-Spline approach (blue solid line). The right panel also shows the estimated functions and the pointwise 95% credible interval for $\beta(t)$ using the Penalized B-spline approach.

In this section we focus on models that regress functional scores on other covariates. This type of models are important to identify and quantify predictors of observed principal directions of functional variability. For simplicity, we focus on predictors of subject-specific scores, ξ_{i,k_0}, on the k_0th eigenfunction, $\phi_{k_0}(t)$. We regress ξ_{ik_0} on the covariate vector $\boldsymbol{Z}_i$, which could include the scores on other eigenfunctions. The full model is

$$\begin{cases} \xi_{ik_0} & \sim & \mathrm{N}(\boldsymbol{Z}_i^t\boldsymbol{\gamma}, \lambda_{k_0}); \\ W_i(t) & = & \sum_{k=1}^{K} \xi_{ik}\psi_k(t) + \epsilon_i(t); \\ \xi_{ik} & \sim & \mathrm{N}(0, \lambda_k)\ k \neq k_0; \\ \epsilon_i(t) & \sim & \mathrm{N}(0, \sigma_\epsilon^2). \end{cases} \tag{7}$$

The main difference between this model and model (2) is that the scores ξ_{ik_0} are shrunk towards $\boldsymbol{Z}_i^t\boldsymbol{\gamma}$ instead of 0.

Changes to the WinBUGS 1.4 program for the classical functional regression model. We provide the entire program in the Appendix A4. For illustration, consider the case where a the functional scores on the first eigenfunction is regressed on age and body mass index (BMI). That is,

$$\boldsymbol{Z}_i^t\boldsymbol{\gamma} = \mu + \mathrm{age}_i\gamma_1 + \mathrm{BMI}_i\gamma_2.$$

This additional level is included in the WinBUGS 1.4 program by simply adding the following

code describing the logistic model for the outcome to the WinBUGS 1.4 code described in Section 2.2

```
for (i in 1:N_subj)
    {xi[i,1]~dnorm(m_xi[i],ll[1])
     m_xi[i]<-mu+gamma[1]*age[i]+gamma[2]*BMI[i]
```

In this case there were 1 missing age and 41 missing BMI values. We added an imputation model for age and BMI by adding the following code

```
for (i in 1:N_subj)
    {age[i]~dnorm(mu_X[1],tau[1])
     BMI[i]~dnorm(mu_X[2],tau[2])}
```

This is the WinBUGS 1.4 representation of the simple Gaussian imputation priors $\text{age}_i \sim N(\mu_{X,1}, \tau_1)$ and $\text{BMI}_i \sim N(\mu_{X,2}, \tau_2)$. More complex imputation priors could be set up, but this exceeds the scope of this paper.

One also needs to specify the prior distributions for $\mu, \gamma_1, \gamma_2, \mu_{X,1}, \mu_{X,2}, \tau_1, \tau_2$. This is done as

```
mu~dnorm(0,1.0E-2)
for (l in 1:2)
    {gamma[l]~dnorm(0,1.0E-2)
     mu_X[l]~dnorm(m_prior[l],1.0E-3)
     tau[l]~dgamma(1.0E-3,1.0E-3)}
```

Here `m_prior[l]` is a 2-dimensional vector containing the sample means of the observed age and BMI values, respectively.

Results. We used the first hour of normalized EEG δ-power for 500 subjects as functional predictors and we regressed the first principal component scores on age and BMI. Table 3 provides the posterior mean and standard deviation of the parameters γ_1 and γ_2. Results indicate that higher age and BMI are positively associated with higher scores on the first principal component. A closer look at Figure 2 shows that the first principal component is negative and is, roughly, a vertical shift. We conclude that higher age and BMI are significantly associated with lower EEG percent δ-power in the first hour of sleep.

	γ_1 (age)	γ_2 (BMI)
mean	0.016	0.021
St. Dev.	0.005	0.007

Table 3: Estimated effects of age and BMI on the first principal component scores. Results shown for 500 subjects and first hour of sleep.

4. Multilevel modeling of functional data

The number of data sets where functional data is observed at multiple visits or within clusters is increasing. For example, the SHHS contains EEG data at two visits for thousands of subjects.

In this Section we describe Bayesian methods for the analysis of a sample of curves observed at multiple visits. The basic idea is to decompose each curve into a population average, a visit-specific deviation from the population average, a subject-specific deviation from the visit-specific mean, a subject/visit specific deviation from the subject-specific mean and measurement error. The subject-specific and the subject/visit specific deviations are modeled by projecting them on the eigenvectors of the covariance matrices of the subject- and subject/visit-specific processes, respectively. Using the mixed model formulation of the underlying model we obtain the joint posterior distribution of all parameters given the data. The parameter space includes all subject-specific and subject/visit-specific functions and their individual scores. Methods in this section can be applied to sparse or dense multilevel functional data.

4.1. Multilevel Functional Principal Component Analysis

For illustration consider the sleep EEG percent δ-power for the first hour of sleep of 500 subjects who have two visits, roughly 5 years apart. Denote the observed sleep EEG fraction of δ-power by $W_{ij}(t)$, for subject $i = 1, \ldots, I = 500$ at visit $j = 1, J = 2$, at the 30-second interval $t = 1, \ldots, T = 120$. In this data set 18% of observations are missing because wake periods are removed.

We assume that $W_{ij}(t)$ is a proxy observation of the true underlying subject-specific functional signal $X_i(t)$, and that

$$W_{ij}(t) = \mu(t) + \eta_j(t) + X_i(t) + U_{ij}(t) + \epsilon_{ij}(t).$$

Here $\mu(t)$ is the overall mean function, $\eta_j(t)$ is the visit j specific shift from the overall mean function, $X_i(t)$ is the subject i specific deviation from the visit specific mean function, and $U_{ij}(t)$ is the residual subject/visit specific deviation from the subject specific mean. To ensure identifiability we assume that $X_i(t)$, $U_{ij}(t)$, and $\epsilon_{ij}(t)$ are uncorrelated, that $\sum_j \eta_j(t) = 0$ and that $\epsilon_{ij}(t)$ is a white noise process with variance σ_ϵ^2. Given the large sample size of the SHHS data, we can assume that $\mu(t)$ and $\eta_j(t)$ are estimated with negligible error by $\overline{W}_{\cdot\cdot}(t)$ and $\overline{W}_{\cdot j}(t) - \overline{W}_{\cdot\cdot}$, respectively. Here $\overline{W}_{\cdot\cdot}(t)$ is the average over all subjects, i, and visits, j, of $W_{ij}(t)$ and $\overline{W}_{\cdot j}(t)$ is the average over all subjects, i, of observation at visit j of $W_{ij}(t)$. We can assume that these estimates have been subtracted from $W_{ij}(t)$, so that $W_{ij}(t) = X_i(t) + U_{ij}(t) + \epsilon_{ij}(t)$.

Eigenvalues and eigenvectors in MPFPCA. We use MFPCA Di *et al.* (2009) to obtain the parsimonious bases that capture most of the functional variability of the space spanned by $X_i(t)$ and $U_{ij}(t)$, respectively. MFPCA is based on the spectral decomposition of the within- and between-visit functional variability covariance operators. We summarize here the main components of this methodology. Denote by $K_T^W(s,t) = \text{cov}\{\, W_{ij}(s), W_{ij}(t)\,\}$ and $K_B^W(s,t) = \text{cov}\{\, W_{ij}(s), W_{ik}(t)\,\}$ for $j \neq k$ the total and between covariance operator corresponding to the observed process, $W_{ij}(\cdot)$, respectively. Denote by $K^X(t,s) = \text{cov}\{X_i(t), X_i(s)\}$ the covariance operator of the $X_i(\cdot)$ process and by $K^U(t,s) = \text{cov}\{\, U_{ij}(s), U_{ij}(t)\,\}$ the total covariance operator of the $U_{ij}(\cdot)$ process. By definition, $K_B^U(s,t) = \text{cov}\{\, U_{ij}(s), U_{ik}(t)\,\} = 0$ for $j \neq k$. Moreover, $K_B^W(s,t) = K^X(s,t)$ and $K_T^W(s,t) = K^X(s,t) + K^U(s,t) + \sigma_\epsilon^2 \delta_{ts}$, where δ_{ts} is equal to 1 when $t = s$ and 0 otherwise. Thus, $K^X(s,t)$ can be estimated using a method of moments estimator of $K_B^W(s,t)$, say $\widehat{K}_B^W(s,t)$. For $t \neq s$ a method of moment estimator of $K_T^W(s,t) - K_B^W(s,t)$, say $\widehat{K}^U(s,t)$, can be used to estimate $K^U(s,t)$. To estimate $\widehat{K}^U(t,t)$ one predicts $K^U(t,t)$ using a bivariate thin-plate spline smoother of $\widehat{K}^U(s,t)$ for $s \neq t$. This method was proposed for single-level FPCA Yao and Lee (2006) and shown to work well in the MFPCA context Di *et al.* (2009).

Once consistent estimators of $K^X(s,t)$ and $K^U(s,t)$ are available, the spectral decomposition and functional regression proceed as in the single-level case. More precisely, Mercer's theorem (see Indritz (1963), Chapter 4) provides the following convenient spectral decompositions $K^X(t,s) = \sum_{k=1}^\infty \lambda_k^{(1)} \psi_k^{(1)}(t) \psi_k^{(1)}(s)$, where $\lambda_1^{(1)} \geq \lambda_2^{(1)} \geq \ldots$ are the ordered eigenvalues and $\psi_k^{(1)}(\cdot)$ are the associated orthonormal eigenfunctions of $K^X(\cdot,\cdot)$ in the L^2 norm. Similarly, $K^U(t,s) = \sum_{l=1}^\infty \lambda_l^{(2)} \psi_l^{(2)}(t) \psi_l^{(2)}(s)$, where $\lambda_1^{(2)} \geq \lambda_2^{(2)} \geq \ldots$ are the ordered eigen-

values and $\psi_l^{(2)}(\cdot)$ are the associated orthonormal eigenfunctions of $K^U(\cdot,\cdot)$ in the L^2 norm. The Karhunen-Loève (KL) decomposition Karhunen (1947); Loève (1945) provides the following infinite decompositions $X_i(t) = \sum_{k=1}^{\infty} \xi_{ik}\psi_k^{(1)}(t)$ and $U_{ij}(t) = \sum_{l=1}^{\infty} \zeta_{ijl}\psi_l^{(2)}(t)$ where $\xi_{ik} = \int_0^1 X_i(t)\psi_k^{(1)}(t)dt$, $\zeta_{ijl} = \int_0^1 U_{ij}(t)\psi_l^{(2)}(t)dt$ are the principal component scores with $E(\xi_{ik}) = E(\zeta_{ijl}) = 0$, $\mathrm{Var}(\xi_{ik}) = \lambda_k^{(1)}$, $\mathrm{Var}(\zeta_{ijl}) = \lambda_l^{(2)}$. The zero-correlation assumption between the $X_i(\cdot)$ and $U_{ij}(\cdot)$ processes is ensured by the assumption that $\mathrm{cov}(\xi_i, \zeta_{ijl}) = 0$. These properties hold for every i, j, k, and l. For simplicity we will refer to $\psi_k^{(1)}(\cdot)$, $\psi_l^{(2)}(\cdot)$ and $\lambda_k^{(1)}$, $\lambda_l^{(2)}$ as the level 1 and 2 eigenfunctions and eigenvalues, respectively.

For the SHHS example the estimated eigenvalues of $\widehat{K}^X(\cdot,\cdot)$ and $\widehat{K}^U(\cdot,\cdot)$ are provided in Table 4. The Level 1 and 2 labels to refer to the X and U processes, respectively. Approximately 99% of the subject-specific process variability is explained by the first 3 principal components whereas as many as 12 components are necessary to explain the same percentage of variability for the subject-visit-specific variability. The within-subject between-visit correlation can be estimated as

$$\widehat{\rho}_W = \frac{\sum_{k\geq 1} \widehat{\lambda}_k^{(1)}}{\sum_{k\geq 1} \widehat{\lambda}_k^{(1)} + \sum_{l\geq 1} \widehat{\lambda}_l^{(2)}}.$$

In the SHHS example considered in this paper $\widehat{\rho}_W = 0.33$.

Figure 7 displays the first two eigenvectors for the Level 1 (top panels) and 2 (bottom panels) processes. The first eigenvalues at both levels capture mainly vertical shifts. The second eigenvalue at the first level is roughly centered around zero. Subjects who are positively loaded on this component will tend to have: 1) a lower sleep EEG δ-power proportion than the population average in the first 10 and last 10 minutes of the first hour of sleep; and 2) a higher sleep EEG δ-power proportion than the population average between minutes 20 and 40. This eigenfunction is not smooth, most likely because of the sample size and small proportion of variation explained. One could easily smooth it using any reasonable scatterplot smoother. Here we work directly with the eigenvalue shown in Figure 7. The second eigenvalue at the second level is also roughly centered around zero. Subject-visits that are positively loaded on this component will tend to have: 1) a higher sleep EEG δ-power proportion than the subject-specific average in the first 25 minutes of the first hour of sleep; and 2) a lower sleep EEG δ-power proportion than the subject-specific average between in the last 10 minutes of the first hour of sleep.

Estimating the scores in the MFPCA. Conditional on the eigenfunctions and truncation lags

Eigenvalues - level 1										
	1	2	3	4	5	6	7	8	9	10
Var	0.49	0.04	0.02							
% Var	88.6	6.8	3.9							
sum % Var	88.6	95.4	99.3							
Eigenvalues - level 2										
Var	0.42	0.27	0.14	0.09	0.06	0.04	0.02	0.02	0.02	0.01
% Var	37.7	24.1	12.3	7.9	5.1	3.3	2.0	1.7	1.4	1.3
sum % Var	37.7	61.9	74.2	82.2	87.2	90.6	92.6	94.3	95.7	97.0

Table 4: Estimated eigenvalues for the percent δ power using data for the first hour of sleep during the baseline visit of 500 subjects from the SHHS.

K and L, the model for observed functional data can be written as a linear mixed model. Indeed, by assuming a normal shrinkage distribution for scores and errors, the model can be rewritten as

$$\begin{cases} W_{ij}(t) & = & \sum_{k=1}^{K} \xi_{ik}\psi_k^{(1)}(t) + \sum_{l=1}^{L} \zeta_{ijl}\psi_l^{(2)}(t) + \epsilon_{ij}(t); \\ \xi_{ik} & \sim & \mathrm{N}\{0, \lambda_k^{(1)}\};\ \zeta_{ijl} \sim \mathrm{N}\{0, \lambda_l^{(2)}\};\ \epsilon_{ij}(t) \sim \mathrm{N}(0, \sigma_\epsilon^2). \end{cases} \tag{8}$$

This model, like the other models in this paper, are especially well suited for Bayesian inference; the implementation in WinBUGS 1.4 is simple, as we show in the following Section. The properties of the mixing chains remain outstanding for the same reasons: orthogonality of the principal component bases, data reduction and mixed model representation of functional models.

4.2. WinBUGS 1.4 program for the multilevel exposure model

We provide the entire program in the Appendix A5. The WinBUGS 1.4 program for the multilevel functional model has a similar structure with the one for the single level functional model presented in Section 2.2. Model (8) describes the core components (likelihood and shrinkage assumptions) of MFPCA and is specified in WinBUGS 1.4 as follows

```
for (i in 1:N_subj)
    {for (t in 1:N_obs)
        {W_1[i,t]~dnorm(m_1[i,t],taueps)
         W_2[i,t]~dnorm(m_2[i,t],taueps)
```

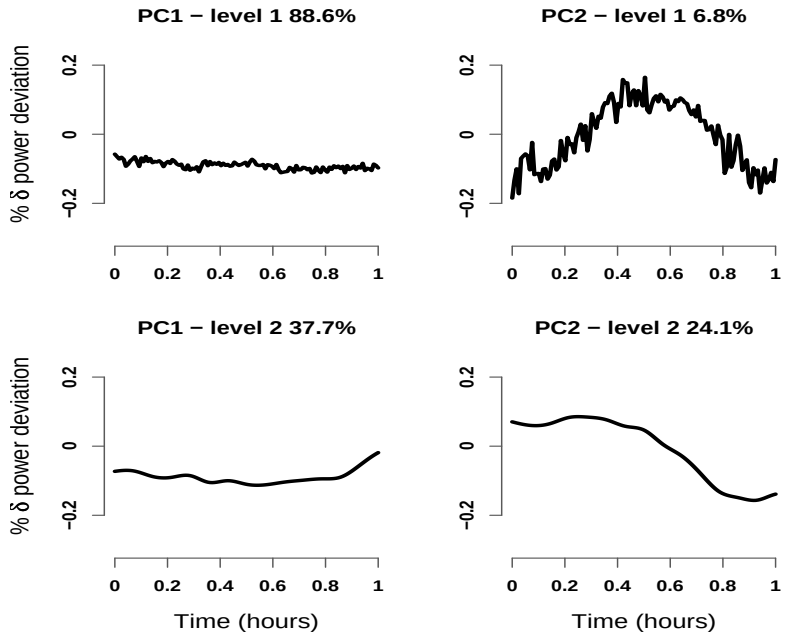

Figure 7: Top panels: First and second eigenvectors at level 1, $\psi_1^{(1)}(t)$ and $\psi_2^{(1)}(t)$. Bottom panels: First and second eigenvectors at level 1, $\psi_1^{(2)}(t)$ and $\psi_2^{(2)}(t)$.

```
m_1[i,t]<-X[i,t]+U_1[i,t]
m_2[i,t]<-X[i,t]+U_2[i,t]

X[i,t]<-xi[i,1]*psi_1[t,1]+xi[i,2]*psi_1[t,2]+xi[i,3]*psi_1[t,3]

U_1[i,t]<-zi[i,1,1]*psi_2[t,1]+zi[i,2,1]*psi_2[t,2]+zi[i,3,1]*psi_2[t,3]+
   zi[i,4,1]*psi_2[t,4]+zi[i,5,1]*psi_2[t,5]+zi[i,6,1]*psi_2[t,6]+
   zi[i,7,1]*psi_2[t,7]+zi[i,8,1]*psi_2[t,8]+zi[i,9,1]*psi_2[t,9]+
   zi[i,10,1]*psi_2[t,10]

U_2[i,t]<-zi[i,1,2]*psi_2[t,1]+zi[i,2,2]*psi_2[t,2]+zi[i,3,2]*psi_2[t,3]+
   zi[i,4,2]*psi_2[t,4]+zi[i,5,2]*psi_2[t,5]+zi[i,6,2]*psi_2[t,6]+
   zi[i,7,2]*psi_2[t,7]+zi[i,8,2]*psi_2[t,8]+zi[i,9,2]*psi_2[t,9]+
   zi[i,10,2]*psi_2[t,10]
}
```

```
for (k in 1:dim.space_b)
   {xi[i,k]~dnorm(0,ll_b[k])}

for (l in 1:dim.space_w)
   {zi[i,l,1]~dnorm(0,ll_w[l])
    zi[i,l,2]~dnorm(0,ll_w[l])}
   }#
```

This part of the program describes a double loop over the subjects, `for (i in 1:N_subj)`, and over the number of observations within subjects, `for (t in 1:N_obs)`. The number of subjects, `N_subj`, is a constant in the program and is equal to 500. The number of observations within subject, `N_obs`, is also a constant and is equal to 120, which corresponds to the number of 30-second intervals in one hour. As in the single-level case, missing observations are treated as random and are estimated like all the other unknowns in the model. Even though `N_obs` is a constant in the program, the number of observations per subject varies. Observations that are missing are simply included as `NA` in the program.

The first four statements specify that the functions $W_{ij}(t)$, the observed percent sleep EEG δ-power, have a normal distribution with mean $m_{ij}(t) = X_i(t) + U_{ij}(t)$, the true percent EEG δ-power, and precision $\tau_\epsilon = \sigma_\epsilon^{-2}$. Here `W_1[i,t]` and `W_2[i,t]` are the WinBUGS 1.4 representation of $W_{i1}(t)$ and $W_{i2}(t)$, respectively. Similarly, `U_1[i,t]` and `U_2[i,t]` are the WinBUGS 1.4 representation of $U_{i1}(t)$ and $U_{i2}(t)$, respectively. A more compact representation of these processes in WinBUGS 1.4 could be achieved using triple-indexing, which would be especially useful for more than two visits. However, our implementation works well and proved to be especially useful during debugging.

The fifth statement provides the structure of the subject-specific process, $X_i(t)$. Here `psi_1[t,k]` denotes the kth level 1 eigenfunction evaluated at time t, $\psi_k^{(1)}(t)$. In this case we only used the first three eigenfunctions because together they explain more than 99% of the functional variability at the subject-level. All eigenfunctions are obtained from the diagonalization procedure of the smooth estimator $\widehat{K}_X(t,s)$ of $K_X(t,s)$ and are treated as data; the first two eigenfunctions are displayed in the top panels of Figure 7. Also, `xi[i,k]` corresponds to ξ_{ik} in model (8), which is the score of the subject i-specific mean function, $X_i(t)$, on the kth

level 1 eigenfunction, $\psi_k^{(1)}(t)$. Thus, `xi[,]` is a $I \times K$ dimensional matrix of random parameters, whose joint posterior distribution is one of the targets of inference. In our example, $I \times K = 500 * 3 = 1,500$.

The sixth and seventh statements specify the structure of the subject-visit-specific deviations, $U_{i1}(t)$ and $U_{i2}(t)$, from the subject specific mean, $X_i(t)$. Here `psi_2[t,k]` denotes the kth level 2 eigenfunction evaluated at time t, $\psi_k^{(2)}(t)$. In this case we used the first ten eigenfunctions because together they explain more than 97% of the functional variability at the subject-visit level. All eigenfunctions are obtained from the diagonalization procedure of the smooth estimator $\widehat{K}_W(t,s)$ of $K_W(t,s)$ and are treated as data; the first two eigenfunctions are displayed in the bottom panels of Figure 7. Also, `zi[i,l,j]` corresponds to ζ_{ijl} in model (8), which is the score of the subject i, visit j deviation, $U_{ij}(t)$, from the subject-specific mean, $X_i(t)$, on the lth level 2 eigenfunction, $\psi_k^{(2)}(t)$. Thus, `zi[,,]` is a $I \times L \times J$ dimensional matrix of random parameters, whose joint posterior distribution is one of the targets of inference. In our example, $I \times L \times J = 500 * 10 * 2 = 10,000$.

The shrinkage assumptions $\xi_{ik} \sim \mathrm{N}\{0, \lambda_k^{(1)}\}$ in model (8) are specified as `xi[i,k]~dnorm(0,ll_b[k])`, where `ll_b[k]` are the precision parameters $\tau_k^{(1)} = 1/\lambda_k^{(1)}$ and are estimated from the data. Here the dimension of the level 1 space is $K = 3$, is denoted as `dim.space_b` and is loaded as data in the program.

The shrinkage assumptions $\zeta_{ijl} \sim \mathrm{N}\{0, \lambda_l^{(2)}\}$ in model (8) are specified as `zi[i,l,1]~dnorm(0,ll_w[l])` and `zi[i,l,1]~dnorm(0,ll_w[l])`, where `ll_w[k]` are the precision parameters $\tau_k^{(2)} = 1/\lambda_k^{(2)}$ and is a parameter that is estimated from the data. Here the dimension of the level 2 space is $L = 10$, is denoted as `dim.space_w` and is loaded as data in the program.

The parameters $\lambda_k^{(1)}$, $\lambda_k^{(2)}$ σ_ϵ^2 are jointly estimated with the other parameters of the model. Thus, we need to specify priors for the variance parameters as

```
for (i in 1:dim.space_b)
    {ll_b[i]~dgamma(1.0E-3,1.0E-3)
     lambda_b[i]<-1/ll_b[i]}


for (i in 1:dim.space_w)
    {ll_w[i]~dgamma(1.0E-3,1.0E-3)
     lambda_w[i]<-1/ll_w[i]}
```

```
taueps~dgamma(1.0E-3,1.0E-3)
sigma_sq_eps=1/taueps
```

An alternative would be to use the method of moment estimators of $\lambda_k^{(1)}$ and use this estimator as data. This is a robust approach that works well in practice, but is not used in this paper. The matrices `W_1[,]` and `W_2[,]` are $I \times T$, `psi_1[,]` is $T \times K$, and `psi_2[,]` is $T \times L$ dimensional, respectively; they are obtained outside WinBUGS 1.4 and are entered as data.

4.3. Results

We obtained $1,500$ simulations, discarded the first 500 as burn-in and used the remainder $1,000$ for inference. For $I = 500$ subjects, $J = 2$ visits and $T = 120$ grid points for each function the total computation time was 10.2 minutes (Dual Core Processor 3GHz, 8Mb RAM PC). Figure 8 displays the posterior mean and 95% pointwise credible intervals for the same 4 subjects from Figure 4 at the first visit, using the multilevel model (8) instead of the single level model (2). For the subject in the bottom left panel of Figure 4 we display the more detailed inference permitted by the multilevel analysis. Figure 9 displays the sleep EEG percent δ-power for this subject at visit 1 and 2 (top two panels) together with the posterior mean and 95% pointwise credible intervals. The bottom left panel in Figure 9 displays the posterior mean and pointwise 95% credible intervals for the subject specific deviation from the visit-specific mean, $X_i(t)$. The posterior mean of $X_i(t)$ is positive everywhere indicating that this subject tends to have a higher sleep EEG percent δ-power than the average. However, none of these differences is statistically significant as all credible intervals cross zero. The bottom right panel in Figure 9 displays the posterior means of the subject-visit-specific deviations, $U_{i1}(t)$ and $U_{i2}(t)$, from the subject-specific mean, $\mu(t) + \eta_j(t) + X_i(t)$. The black solid line corresponds to the random functional effect at the first visit, U_{i1}, and the gray solid line corresponds to the random functional effect at the first visit, U_{i2}. Pointwise credible intervals are available for these processes, as well, but were omitted in the plot to avoid cluttering.

5. Interface with and processing in R

We have used R to define the data, initial values and parameters and the R2winBUGS function

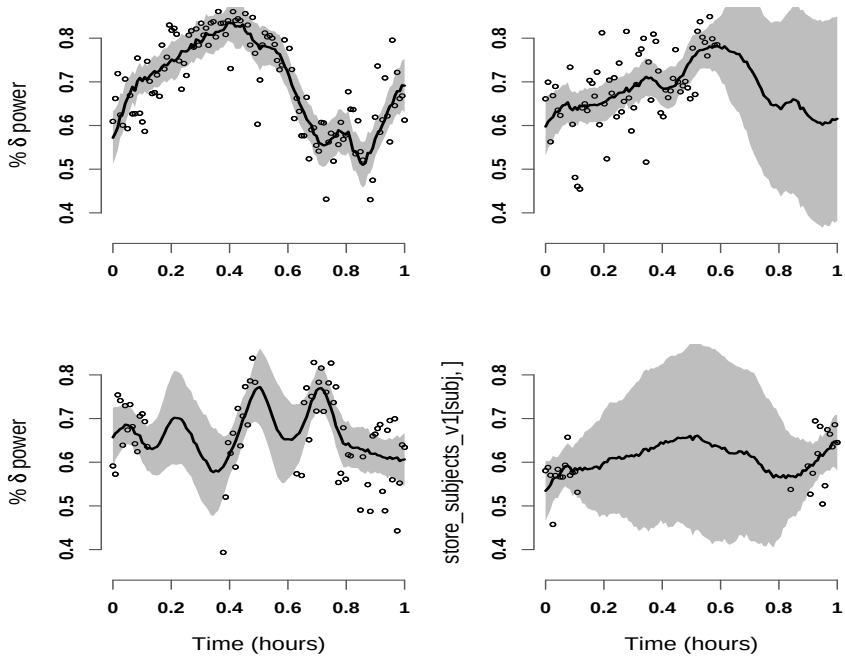

Figure 8: Estimated subject-specific means with 95% pointwise credible intervals for the EEG normalized δ-power in the first hour of sleep of the first visit for 4 subjects from the SHHS. The model used was the multilevel model (8).

Sturtz, Ligges, and Gelman (2005) to call and run the WinBUGS 1.4 part of the program. R was also used to do output checking and processing as well as plotting. Once the WinBUGS 1.4 code is written and debugged, the user can simply use the R interface to perform analyses or set up simulations. The software accompanying this paper contains the commented R interface. As this part is now becoming routine, we do not present here the details.

6. Discussion

This paper is a compilation of examples of Bayesian functional data analysis implemented in WinBUGS 1.4. This provides a transparent, easy to use, reproducible alternative to frequentist software such as FDA maintained by J. Ramsay, PACE maintained by H.G. Müller, and the R package nlmeODE maintained by C.W. Tornoe. There are at least three reasons for providing WinBUGS code for the Bayesian analysis of functional data. First, for regression models such as those introduced in Section 3, Bayesian analysis provides a natural platform for joint modeling. This can be especially useful when functional data are sparse or functional scores

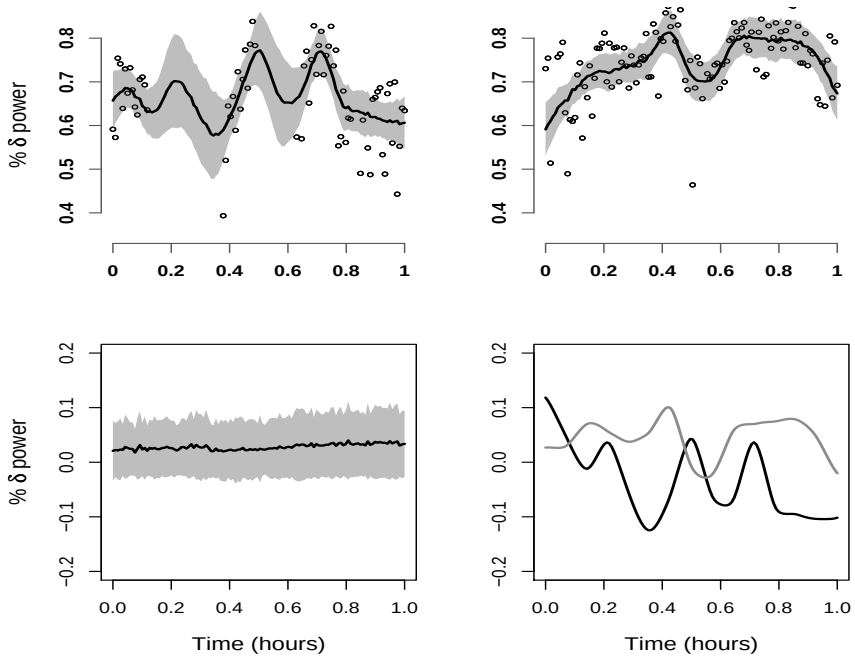

Figure 9: Top panels: First and second visit EEG normalized δ-power data for the subject shown in the bottom left panel of Figure 8. Also shown are the posterior means and 95% credible intervals using the multilevel model (8). Bottom left panel: Posterior mean and 95% pointwise credible interval for the subject-specific deviation process, $X_i(t)$. Bottom-right panel: Posterior means of the subject-visit-specific deviation processes, U_{i1} (black solid line) and U_{i2} (gray solid line).

are predicted with sizeable error. Second, these examples could provide the starting point for the implementation of more complex models for realistic data; our WinBUGS programs would require only minimal changes to include random effects, smooth uni- or multi-variate components, and missing or miss-measured data. Third, these programs provide an alternative platform that could be used to confirm results of frequentist software. While, at this time, our programs are the only ones that can handle smooth penalized regression (Section 3.2) and multilevel FPCA (Section 4), this will probably change with new versions of frequentist software.

Acknowledgments

Crainiceanu's research was supported by Award Number R01NS060910 from the National Institute Of Neurological Disorders And Stroke. The content is solely the responsibility of

the author and does not necessarily represent the ościal views of the National Institute Of Neurological Disorders And Stroke or the National Institutes of Health.

References

Baladandayuthapani V, Mallick BK, Hong MY, Lupton JR, Turner ND, Carroll R (2008). "Bayesian hierarchical spatially correlated functional data analysis with application to colon carcinogenesis." *Biometrics*, **64**, 64Ű73.

Brezger A, Kneib T, Lang S (2005). "BayesX: Analyzing Bayesian Structured Additive Regression Models." *Journal of Statistical Software*, **14**.

Cardot H, Ferraty F, Sarda P (1999). "Functional linear model." *Statistics & Probability Letters*, **45**, 11–22.

Cardot H, Ferraty F, Sarda P (2003). "Spline estimators for the functional linear model." *Statistica Sinica*, **13**, 571–591.

Cardot H, Sarda P (2005). "Estimation in Generalized Linear Model for Functional Data via Penalized Likelihood." *Journal of Multivariate Analysis*.

Carlin B, Louis T (2000). *Bayes and Empirical Bayes Methods for Data Analysis, Second Edition*. Chapman & Hall/CRC.

Chiou JM, Müller HG, Wang JL (2003). "Functional quasi-likelihood regression models with smooth random effects." *Journal of the Royal Statistical Society, Series B*, **65**, 405–423.

Congdon P (2003). *Applied Bayesian Modelling*. Wiley.

Crainiceanu C, Caffo B, Punjabi N (2009a). "Nonparametric signal extraction and measurement error in the analysis of electroencephalographic activity during sleep." *Journal of the American Statistical Association, to appear*.

Crainiceanu C, Ruppert D (2004). "Likelihood ratio tests in linear mixed models with one variance component." *Journal of the Royal Statistical Society, Series B*, **66**.

Crainiceanu C, Ruppert D, Carroll R, Adarsh J, Goodner B (2007). "Spatially adaptive Penalized splines with heteroscedastic errors." *Journal of Computational and Graphical Statistics*, **16(2)**.

Crainiceanu C, Ruppert D, Claeskens G, Wand M (2005). "Exact Likelihood Ratio Tests for Penalized Splines." *Biometrika*, **92(1)**.

Crainiceanu C, Staicu A, Di C (2009b). "Generalized Multilevel Functional Regression." *Journal of the American Statistical Association, to appear.*

Di C, Crainiceanu C, Caffo B, Punjabi N (2009). "Multilevel Functional Principal Component Analysis." *Annals of Applied Statistics, online access 2008*, **3(1)**, 458–488.

Gelman A (2006). "Prior distributions for variance parameters in hierarchical models." *Bayesian Analysis*, **1**(3), 515–533.

Gelman A, Carlin J, Stern H, Rubin D (2003). *Bayesian Data Analysis, Second Edition.* Chapman & Hall/CRC.

Gelman A, Rubin D (????). "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Science*, pp. 457–472.

Gilks W, Richardson S, Spiegelhalter D (1996). *Markov Chain Monte Carlo in Practice.* Chapman & Hall/CRC.

Goldsmith A, Feder J, Crainiceanu C (2010). "Penalized Functional Regression." *manuscript.*

Greven S, Crainiceanu C, Küchenhoff H, Peters A (2008). "Restricted Likelihood Ratio Testing for Zero Variance Components in Linear Mixed Models." *Journal of Computational and Graphical Statistics*, **17**(4), 870–891.

Guo W (2002). "Functional mixed effects models." *Biometrics*, **58**, 121–128.

Indritz J (1963). *Methods in analysis.* Macmillan & Colier-Macmillan.

James G (2002). "Generalized Linear Models with Functional Predictors." *Journal of the Royal Statistical Society, Series B*, **64**, 411–432.

James G, Hastie T, Sugar C (2001). "Principal component models for sparse functional data." *Biometrika*, **87**, 587–602.

Karhunen K (1947). *Über lineare Methoden in der Wahrscheinlichkeitsrechnung.* Suomalainen Tiedeakatemia.

Lang S, Brezger A (2004). "Bayesian P-Splines." *Journal of Computational and Graphical Statistics*, **13**, 183–212.

Loève M (1945). "Functions aleatoire de second ordre." *Comptes Rendus Acad. Sci*, **220**.

Morris J, Carroll R (2006). "Wavelet-based functional mixed models." *Journal of the Royal Statistical Society, B*, **68**, 179Ű–199.

Morris J, Vanucci M, Brown P, Carroll R (2003). "Wavelet-Based Nonparametric Modeling of Hierarchical Functions in Colon Carcinogenesis." *Journal of the American Statistical Association*, **98**, 573–583.

Müller HG, Stadtmüller U (2005). "Generalized Functional Linear Models." *The Annals of Statististics*, **33**(2), 774–805.

Natarajan R, Kass R (2000). "Reference Bayesian methods for generalized linear mixed models." *Journal of the American Statistical Association*, **95**, 227–237.

O'Sullivan F (1986). "A statistical perspective on ill-posed inverse problems (with discussion)." *Statistical Science*, (1), 505Ű–527.

Quan S, Howard B, Kiley CIJ, Nieto F, O'Connor G (1997). "The Sleep Heart Health Study: design, rationale, and methods." *Sleep*, **20**, 1077–85.

Ramsay J, Silverman B (2005). *Applied Functional Data Analysis.* Springer-Verlag, New York.

Ramsay J, Silverman B (2006). *Functional Data Analysis.* Springer-Verlag, New York.

Reiss P, Ogden R (2007). "Functional Principal Component Regression and Functional Partial Least Squares." *Journal of the American Statistical Asscoation*, **102**, 984–996.

Ruppert D, Carroll R, Wand M (2003). *Semiparametric Regression.* Cambridge University Press.

Staicu AM, Crainiceanu C, Carroll R (2009). "Fast Methods for Spatially Correlated Multilevel Functional Data." *manuscript.*

Sturtz S, Ligges U, Gelman A (2005). "R2WinBUGS: A Package for Running WinBUGS from R." *Journal of Statistical Software*, **12**(3), 1–16. URL http://www.jstatsoft.org.

Wang N, Carroll R, Lin X (1998). "Efficient semiparametric marginal estimation for longitudinal/clustered data." *Journal of the American Statistical Association*, **100**, 147–157.

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

Yao F, Lee T (2006). "Penalized spline models for functional principal component analysis." *Journal of the Royal Statistical Society. Series B*, **68**, 3–25.

Yao F, Müller HG, Wang JL (2005). "Functional linear regression analysis for longitudinal data." *The Annals of Statistics*, **33**, 2873–2903.

Appendix A1. WinBUGS 1.4 code for Functional Principal Component Analysis with 10 eigenfunctions (Section 2.1).

```
model
{#Start model
    for (i in 1:N_subj)
    {#Begin loop over subjects

    for (t in 1:N_obs)
        {#Begin loop over observations within subjects
        W[i,t]~dnorm(m[i,t],taueps)
        m[i,t]<-xi[i,1]*E[t,1]+xi[i,2]*E[t,2]+xi[i,3]*E[t,3]+
            xi[i,4]*E[t,4]+xi[i,5]*E[t,5]+xi[i,6]*E[t,6]+
            xi[i,7]*E[t,7]+xi[i,8]*E[t,8]+xi[i,9]*E[t,9]+
            xi[i,10]*E[t,10]
        }#End loop over observations within subjects

    for (k in 1:dim.space)
        {#Begin loop over the eigenfunctions
        xi[i,k]~dnorm(0,ll[k])
        }#End loop over the eigenfunctions

    }#End loop over subjects

for (k in 1:dim.space)
    {ll[k]~dgamma(1.0E-3,1.0E-3)
     lambda[k]<-1/ll[k]}

taueps~dgamma(1.0E-3,1.0E-3)
}#End model
```

1. `N_subj` is the number of subjects, in this case equal to 500, and is loaded as data

2. `N_obs` is the number of observations within subjects, and is loaded as data. The program is set up to allow for different numbers of observations per subject with missing data being loaded as `NA`

3. `W[i,t]` is the observed realization of the subject i specific function at the time t. The matrix `W[,]` is `N_subj`×`N_obs`, is loaded as data and may contain missing observations.

4. `m[i,t]` is the smooth mean of `W[i,t]` and is an unknown whose joint distribution is simulated.

5. `taueps` is the precision of the error process due to imperfect observation of `W[i,t]` around its smooth mean `m[i,t]`. This is a parameter of the model that is estimated.

6. `xi[i,k]` is the score of the ith subject on the kth eigenfunction, `E[,k]`. The matrix `xi[,]` is `N_subj`×K dimensional and contains parameters whose posterior joint distribution is simulated. Here `K` is equal to 10,and is hard coded into the program.

7. `E[t,k]` is the kth eigenfunction evaluated at the time t. The matrix `E[,]` is N_obs×`K` dimensional, does not contain any missing observation and is loaded as data.

8. `dim.space` is the dimension of the eigenspace used for projection and is loaded as data. In this case `dim.space` is equal to 10

9. `ll[k]` are the precisions of the scores `xi[i,k]`, are unknown parameters that are estimated from the data.

10. `lambda[k]` are the variances of the scores `xi[i,k]` and are equal to 1/`ll[k]`

11. All precision priors are Gamma priors with mean 1 and variance 1000

Appendix A2. WinBUGS 1.4 code for Functional Regression Analysis with 10 eigenfunctions (Section 3.1).

```
model {#Start model
    for (i in 1:N_subj)
    {#Begin loop over subjects

    Y[i]~dbern(pY[i])
    logit(pY[i])<-mp+beta[1]*xi[i,1]+beta[2]*xi[i,2]+beta[3]*xi[i,3]

    for (t in 1:N_obs)
    {#Begin loop over observations within subjects

        W[i,t]~dnorm(m[i,t],taueps)
        m[i,t]<-xi[i,1]*E[t,1]+xi[i,2]*E[t,2]+xi[i,3]*E[t,3]+
            xi[i,4]*E[t,4]+xi[i,5]*E[t,5]+xi[i,6]*E[t,6]+
            xi[i,7]*E[t,7]+xi[i,8]*E[t,8]+xi[i,9]*E[t,9]+
            xi[i,10]*E[t,10]
    }#End loop over observations within subjects

    for (j in 1:dim.space)
        {#Begin loop over the eigenfunctions
        xi[i,j]~dnorm(0,ll[j])
        }#End loop over the eigenfunctions
    }#

for (i in 1:dim.space)
    {ll[i]~dgamma(1.0E-3,1.0E-3)
     lambda[i]<-1/ll[i]}
for (l in 1:3) {beta[l]~dnorm(0,1.0E-2)}
mp~dnorm(0,1.0E-2)
taueps~dgamma(1.0E-3,1.0E-3) }#End model
```

The definition of most variables is the same as the definition for WinBUGS 1.4 code for Functional Principal Component Analysis. We describe only the new variables here.

1. `Y[i]` is the Bernoulli outcome for subject i. The vector `Y[]` has length `N_subj` and is loaded as data. In the example in the paper the outcome is hypertension status YES/NO.

2. `pY[i]` is the probability that subject i is hypertensive. The vector `pY[]` has length `N_subj` and is a vector of parameters that are estimated in the model.

3. `logit(pY[i])` is the logit transformation of the probabilities `pY[i]`. The probit transformation could also be used.

4. `beta[]` is a vector of regression parameters of length 3 relating the `logit(pY[i])` to the scores `xi[,]` of subject-specific functions on eigenfunctions, `E[,]`. The length 3 is not set in stone and other lengths can be accommodated when the outcome is regressed on a different set of eigenfunctions.

5. `mp` is the intercept parameter of the regression of `Y[]` on the scores `xi[,]`. This is a parameter that will be estimated in the model.

6. The priors for each `beta[]` parameter and `mp` are mutually independent normals with mean 0 and variance 100 (recall that WinBUGS 1.4 specifies the precision, 0.01, instead of the variance, 100).

Appendix A3. WinBUGS 1.4 code for Penalized Functional Regression Using B-Splines (Section 3.2).

```
model
{#Start model
for (i in 1:N_subj)
     {#Begin loop over subjects
     Y[i]~dbern(pY[i])
     logit(pY[i])<-mp+eta[i]
     eta[i]<-xi[i,1]*gamma[1]+xi[i,2]*gamma[2]+xi[i,3]*gamma[3]+
          xi[i,4]*gamma[4]+xi[i,5]*gamma[5]+xi[i,6]*gamma[6]+
          xi[i,7]*gamma[7]+xi[i,8]*gamma[8]+xi[i,9]*gamma[9]+
          xi[i,10]*gamma[10]

          for (t in 1:N_obs)
          {#Begin loop over observations within subjects
          W[i,t]~dnorm(m[i,t],taueps)
               m[i,t]<-xi[i,1]*E[t,1]+xi[i,2]*E[t,2]+xi[i,3]*E[t,3]+
                    xi[i,4]*E[t,4]+xi[i,5]*E[t,5]+xi[i,6]*E[t,6]+
                    xi[i,7]*E[t,7]+xi[i,8]*E[t,8]+xi[i,9]*E[t,9]+
                    xi[i,10]*E[t,10]
               }# End loop over observations within subjects
     for (j in 1:dim.space)
          {#Begin loop over eigenfunctions
               xi[i,j]~dnorm(0,ll[j])
               }#End loop over eigenfunctions
     }#End loop over subjects

for(l in 1:L){#Begin loop rows of J
     gamma[l]<-J[l,1]*beta[1]+J[l,2]*beta[2]+J[l,3]*beta[3]+
          J[l,4]*beta[4]+J[l,5]*beta[5]+J[l,6]*beta[6]+
          J[l,7]*beta[7]+J[l,8]*beta[8]+J[l,9]*beta[9]+
```

```
          J[l,10]*beta[10]
}#End loop over rows of J

for (i in 1:dim.space)
     {ll[i]~dgamma(1.0E-3,1.0E-3)
      lambda[i]<-1/ll[i]}
for (i in 2:L)
     {beta[l]~dnorm(beta[l-1],taubeta)}
      beta[1]~dnorm(0,1.0E-6)
      taubeta~dgamma(1.0E-3,1.0E-3)

mp~dnorm(0,1.0E-2)
taueps~dgamma(1.0E-3,1.0E-3)}#End model
```

The definition of most variables is the same as the definition for WinBUGS 1.4 code for Functional Regression Analysis with 10 eigenfunctions. We describe only the new variables here.

1. `eta[i]` is the subject-specific component of the linear predictor. The vector `eta[]` has length `N_subj` and is the product of the parameters `xi[i,]` and `b[]`, which are estimated in the model, and the matrix `Jm[,]`, which is fixed.

2. `gamma[]` is the vector containing the current product of the fixed matrix `Jm[,]` and the parameter vector `b[]`, and has length `L` (here L is equal to 10). It is updated outside the loop over subjects.

3. `beta[i]` is the coefficient of the ith B-spline basis function used to express the functional regressor. `beta[]`is the vector of regression parameters and has length `L`. Other values of `L` are possible with straightforward changes to the code presented here.

4. `J[,]` is the `K x L` matrix containing the product of the eigenfunctions and the B-spline basis functions (here $K = L = 10$). It is loaded as data.

5. `taubeta` is the precision parameter for `beta[]`. It has the prior distribution `dgamma(1.0E-3,1.0E-3)` (as do other precision parameters).

Appendix A4. WinBUGS 1.4 code for regression with functional scores as outcomes (Section 3.3).

```
model {#Start model

    for (i in 1:N_subj)
    {#Begin loop over subjects

    xi[i,1]~dnorm(m_xi[i],ll[1])
    m_xi[i]<-mu+gamma[1]*age[i]+gamma[2]*BMI[i]
    age[i]~dnorm(mu_X[1],tau[1])
    BMI[i]~dnorm(mu_X[2],tau[2])

    for (t in 1:N_obs)
        {#Begin loop over observations within subjects
        W[i,t]~dnorm(m[i,t],taueps)
        m[i,t]<-xi[i,1]*E[t,1]+xi[i,2]*E[t,2]+xi[i,3]*E[t,3]+
            xi[i,4]*E[t,4]+xi[i,5]*E[t,5]+xi[i,6]*E[t,6]+
            xi[i,7]*E[t,7]+xi[i,8]*E[t,8]+xi[i,9]*E[t,9]+
            xi[i,10]*E[t,10]
        }#End loop over observations within subjects

    for (k in 2:dim.space)
        {#Begin loop over scores that are not regressed
        xi[i,k]~dnorm(0,ll[k])
        }#End loop over scores that are not regressed
    }#End loop over subjects


for (k in 1:dim.space)
    {ll[k]~dgamma(1.0E-3,1.0E-3)
     lambda[k]<-1/ll[k]}
```

```
for (l in 1:2)
    {#Begin loop over covariates
    gamma[l]~dnorm(0,1.0E-2)
    mu_X[l]~dnorm(m_prior[l],1.0E-3)
    tau[l]~dgamma(1.0E-3,1.0E-3)
    }#End loop over covariates

mu~dnorm(0,1.0E-2)
taueps~dgamma(1.0E-3,1.0E-3) }#End model
```

The definition of most variables is the same as the definition for WinBUGS 1.4 code for Functional Principal Component Analysis and Functional Regression Analysis. We describe only the new variables here.

1. `m_xi[i]` is the mean of the of the subject-specific score `xi[i,1]` on the first eigenfunction. This is different from the classical Functional Regression Analysis where `xi[i,1]` is shrunk towards zero. The vector of parameters `m_xi[]` has length `N_subj` and is estimated in the model.

2. `gamma[]` is a vector of regression parameters of length 2 relating `m_xi[i]` to the covariates `age[i]` and `BMI[i]`. The length 2 is not set in stone and other lengths can be accommodated when the outcome is regressed on a different set of eigenfunctions.

3. `mu` is the intercept parameter of the regression of `xi[i,1]` on the covariates `age[i]` and `BMI[i]`. This is a parameter that will be estimated in the model.

4. `age[]` and `BMI[]` are `N_subj` vectors containing the age and BMI covariates. They are loaded as data in the program.

5. `mu_X[1]` and `mu_X[2]` are the means of the covariates `age[]` and `BMI[]`. These are two parameters that are estimated from the data and are used because some age and BMI observations are missing.

6. `tau[1]` and `tau[2]` are the precisions of the covariates `age[]` and `BMI[]`. These are

two parameters that are estimated from the data and are used because some age and BMI observations are missing.

7. The priors for each `gamma[]` parameter are mutually independent normals with mean 0 and variance 100.

8. The priors for each of `mu_X[l]` parameter are mutually independent normals centered at the empirical mean of the observed covariates `m_prior[l]` and variance 1000.

9. `m_prior[l]` are the empirical means of the observed covariates and are loaded as data in the program.

10. The priors for each of `tau[l]` parameter are mutually independent gammas with mean 1 and variance 1000.

Appendix A5. WinBUGS 1.4 code for multilevel functional modeling (Section 4).

```
model {#Start model
    for (i in 1:N_subj)
    {#Begin loop over subjects

    for (t in 1:N_obs)
        {#Begin loop over observations within subjects

         #Define the distribution of visit 1 and 2 observations
         W_1[i,t]~dnorm(m_1[i,t],taueps)
         W_2[i,t]~dnorm(m_2[i,t],taueps)

         #Write the mean as the subject-specific mean plus
         #the subject-visit-specific deviation
         m_1[i,t]<-X[i,t]+U_1[i,t]
         m_2[i,t]<-X[i,t]+U_2[i,t]

         #Define the subject-specific mean structure
         X[i,t]<-xi[i,1]*psi_1[t,1]+xi[i,2]*psi_1[t,2]+xi[i,3]*psi_1[t,3]

         #Define the visit 1-specific deviation from the mean
         U_1[i,t]<-zi[i,1,1]*psi_2[t,1]+zi[i,2,1]*psi_2[t,2]+zi[i,3,1]*psi_2[t,3]+
            zi[i,4,1]*psi_2[t,4]+zi[i,5,1]*psi_2[t,5]+zi[i,6,1]*psi_2[t,6]+
            zi[i,7,1]*psi_2[t,7]+zi[i,8,1]*psi_2[t,8]+zi[i,9,1]*psi_2[t,9]+
            zi[i,10,1]*psi_2[t,10]

         #Define the visit 2-specific deviation from the mean
         U_2[i,t]<-zi[i,1,2]*psi_2[t,1]+zi[i,2,2]*psi_2[t,2]+zi[i,3,2]*psi_2[t,3]+
            zi[i,4,2]*psi_2[t,4]+zi[i,5,2]*psi_2[t,5]+zi[i,6,2]*psi_2[t,6]+
            zi[i,7,2]*psi_2[t,7]+zi[i,8,2]*psi_2[t,8]+zi[i,9,2]*psi_2[t,9]+
```

```
            zi[i,10,2]*psi_2[t,10]
         }

      for (k in 1:dim.space_b)
         {xi[i,k]~dnorm(0,ll_b[k])}

      for (l in 1:dim.space_w)
         {zi[i,l,1]~dnorm(0,ll_w[l])
          zi[i,l,2]~dnorm(0,ll_w[l])}
         }#

for (k in 1:dim.space_b)
    {ll_b[k]~dgamma(1.0E-3,1.0E-3)
     lambda_b[k]<-1/ll_b[k]}

for (l in 1:dim.space_w)
    {ll_w[l]~dgamma(1.0E-3,1.0E-3)
     lambda_w[l]<-1/ll_w[l]}

taueps~dgamma(1.0E-3,1.0E-3)
sigma_sq_eps<-1/taueps
}#End model
```

The definition of most variables is the same as the definition for WinBUGS 1.4 code for Functional Principal Component Analysis and Functional Regression Analysis. The structure is very similar to the one for Functional Principal Component Analysis with the exception that there are two levels of functional regression.

1. `W_1[i,t]` and `W_2[i,t]` are the functional observations at visit 1 and 2, respectively for subject i at time t. Both matrices `W_1[,]` and `W_2[,]` are `N_subj`×`N_obs`, are loaded as data and may contain missing observations.

2. `m_1[i,t]` and `m_2[i,t]` are the smooth means of `W_1[i,t]` and `W_2[i,t]`, respectively, are unknown and their joint distribution is simulated.

3. `X[i,t]` is the subject-specific mean process. `X[,]` is a `N_subj`×`N_obs` dimensional matrix of parameters that are estimated from the model.

4. `U_1[i,t]` and `U_2[i,t]` are the subject/visit specific deviations from the subject-specific mean. `U_1[,]` and `U_2[,]` are `N_subj`×`N_obs` dimensional matrices of parameters that are estimated from the model.

5. `psi_1[t,k]` and `psi_2[t,l]` are the level 1 and 2 eigenfunctions evaluated at the time t. The matrices `psi_1[,]` and `psi_2[,]` are `N_obs`×`3` and `N_obs`×`10` dimensional, respectively, do not contain any missing observation and are loaded as data.

6. `xi[i,k]` are the scores for the subject i on the kth level 1 eigenfunction `psi_1[t,k]`. `xi[,]` is a `N_subj`×`3` dimensional array of parameters that are estimated from the model.

7. `zi[i,l,1]` and `zi[i,l,2]` are the scores for the subject i on the kth level 2 eigenfunction `psi_2[t,k]`. `zi[,,]` is an `N_subj`×`10`×`2` dimensional array of parameters that are estimated from the model.

8. `ll_b[k]` are the precisions for the distribution of the scores `xi[i,k]`. `ll_b[]` is a vector of parameters of size 3 that are estimated from the model.

8. `ll_w[l]` are the precisions for the distribution of the scores `zi[i,l,1]` and `zi[i,l,2]`, as the level 2 scores are assumed exchangeable. `ll_w[]` is a vector of parameters of size 10 that are estimated from the model.

Affiliation:

Ciprian Crainiceanu

Department of Biostatistics

Johns Hopkins University

615 N. Wolfe St. E3636

Baltimore, MD 21205, United States of America

E-mail: ccrainic@jhsph.edu

URL: http://www.biostat.jhsph.edu/~ccrainic/

A. Jeffrey Goldsmith

Department of Biostatistics

Johns Hopkins University

615 N. Wolfe St. E3037

Baltimore, MD 21205, United States of America

E-mail: jgoldsmi@jhsph.edu

Journal of Statistical Software http://www.jstatsoft.org/

published by the American Statistical Association http://www.amstat.org/

Volume ??, Issue ?? *Submitted:* 2009-07-29

?? ?? *Accepted:* ??