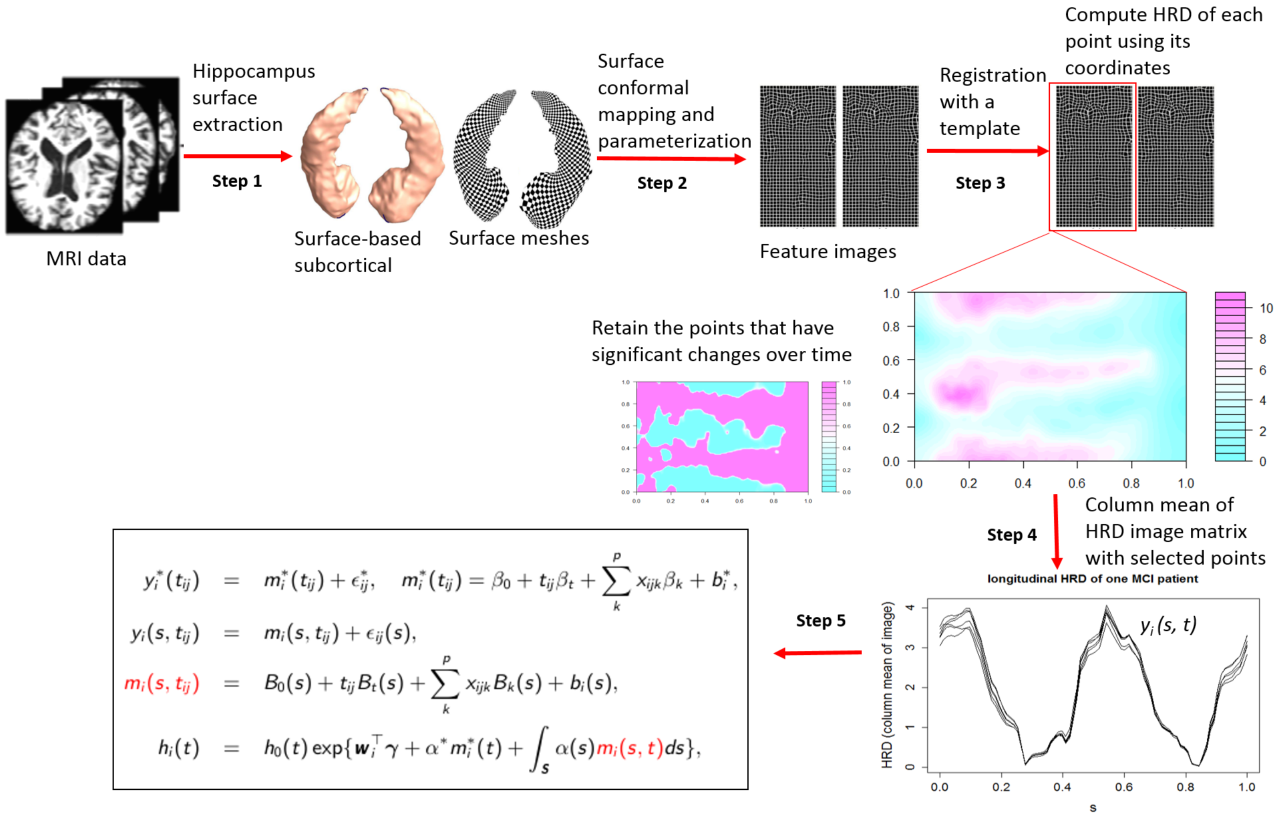
Web-based Supporting Materials for "Bayesian Functional Joint Models for Multivariate Longitudinal and Time-to-Event Data"

1 Hippocampus Image Processing

For image processing, we adopt a surface fluid registration package [1] which has been used in various studies [2–5]. The original imaging data, 3D brain MRI scans, are downloaded from `www.loni.ucla.edu/ADNI`. The first step is the extraction of the left and right hippocampi surfaces from original MRI scans using FIRST [6], an integrated surface analysis tool developed as part of the FSL library [7]. The surface is the outer layer of the brain region and has an inherent 2D structure. Then the surface is modeled as a mesh of triangles for each side of the hippocampi. Each triangle is know as a face. The place where the corners of the triangles meet is called a vertex. The coordinates (i.e., the X, Y, and Z) at each vertex are determined from the MRI during the extraction process. In the second step, each surface is conformally mapped to a 2D rectangle plane using holomorphic 1-forms. This process is similar to unfolding a paper bag. In the third step, a feature image of the surface is computed from this conformal representation and registered to a chosen template image via inverse consistent surface fluid registration. The images of all patients and all visits are registered to the same template. Using conformal mapping, the surface registration problem is essentially converted into an image registration problem. Detailed image processing and registration procedure can be found in Shi *et al* [1].

Each registered feature image is a 150 × 100 image matrix corresponding to either left or right hippocampal surfaces with 15,000 vertices. Knowledge of the coordinates of the vertices allows us to compute radial distance, which measures the distance from the medial core to each surface vertex and represents the hippocampal thickness. We adopt radial distance as our target measurement. The image in Step 3 of Figure 1 displays the hippocampus radial distance being coded in colors. To reduce the noise in the data that may affect prediction, we only retain the vertices that have significant changes over time. We conduct a vertex-by-vertex analysis to identify the vertices. Specifically, for each vertex, we fit a linear mixed effect model that regress longitudinal HRD on time. We keep about 5000 vertices that have a significant time effect, i.e., the adjusted p-values are less than 0.05 after multiple comparison adjustment using Bonferroni method. In the final step, we filter the insignificant points in each column of the image matrix, and then calculate the column mean of the image matrix with the remaining points. In this way, the corresponding radial distances of the vertices on the left or right hippocampal at visit time t are aggregate to form a 1×100-dimensional functional variable denoted by $y_i(s,t)$. Figure 1 displays the data processing procedure along with an illustration of a functional joint model.



$$
\begin{aligned}
y_i^*(t_{ij}) &= m_i^*(t_{ij}) + \epsilon_{ij}^*, \quad m_i^*(t_{ij}) = \beta_0 + t_{ij}\beta_t + \sum_k^p x_{ijk}\beta_k + b_i^*,\\
y_i(s, t_{ij}) &= m_i(s, t_{ij}) + \epsilon_{ij}(s),\\
m_i(s, t_{ij}) &= B_0(s) + t_{ij}B_t(s) + \sum_k^p x_{ijk}B_k(s) + b_i(s),\\
h_i(t) &= h_0(t)\exp\{\mathbf{w}_i^\top\gamma + \alpha^* m_i^*(t) + \int_S \alpha(s) m_i(s, t) ds\},
\end{aligned}
$$

Figure 1: Hippocampus image processing procedure along with an illustration of a multivariate functional joint model. The solid black curves in Step 4 figure illustrate the six longitudinal HRD measurements (column mean of HRD image matrix) of one MCI patients.

2 Additional Simulation Study

	Bias	AMSE	SE	SD	CP
$\beta_0 = 4$	<0.001	0.003	0.052	0.053	0.920
$\beta_t = 1$	−0.003	<0.001	0.022	0.021	0.950
$\beta_1 = 0.5$	<0.001	<0.001	0.004	0.003	0.980
$\gamma_1 = 0.76$	0.018	0.017	0.116	0.130	0.920
$\alpha^* = 0.5$	0.033	0.005	0.059	0.067	0.900
$\alpha = 0.3$	−0.019	0.063	0.231	0.252	0.910

Table 1: Parameter estimation in simulation study based on 100 datasets and $\widehat{K}_\phi = 3$.

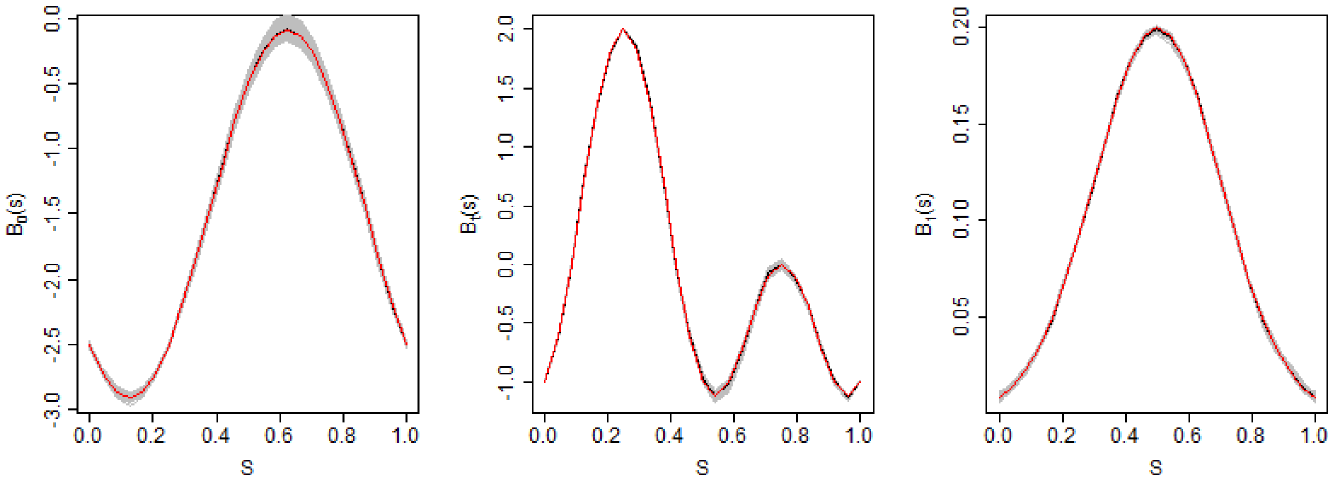


Figure 2: The estimates of the coefficient functions in the simulation study based on 100 runs and $\widehat{K}_\phi = 3$. The red solid lines are mean estimated curves and the grey solid lines are 100 estimated curves based on each individual dataset.

3 Stan code

```
// Stan code for 'Bayesian Functional Joint Models for Multivariate Longitudinal and Time-to-
    Event Data'

data {
        int<lower=0> I;              // number of subjects
        int<lower=0> obs_long;       // total number of observations
        int subj_long[obs_long];     // subject index
        int<lower=0> D;              // grid length
        int<lower=0> p_fun_long;     // number of fixed effects in functional longitudinal sub-
            model, includes intercept and time
        int<lower=0> p_scalar_long;  // number of fixed effects in scale longitudinal sub-model,
            includes intercept and time
        int<lower=0> p_surv;         // number of fixed effects in survival sub-model
        int Kt;                      // number of spline basis functions for B-spline of
            functional coefficients
        int Ka;                      // number of spline basis functions for associate function \
            alpha(s); Ka=1 in the simulation.
        int Kp;                      // number of fPC basis functions for random intercept
            function
        vector[D] Y_fun[obs_long];   // functional outcome matrix
        real Y_scalar[obs_long];     // scalar outcome matrix
        vector[p_fun_longa] X[obs_long];           // fixed effect design matrix in functional
            longitudinal sub-model
        vector[(p_fun_long-1)] X_base[I];          // fixed effect design matrix in functional
            longitudinal sub-model, with only time-independent covariate
        vector[p_scalar_longa] X_scalar[obs_long];  // fixed effect design matrix in scalar
            longitudinal sub-model
        vector[(p_scalar_long-1)] X_scalar_base[I];  // fixed effect design matrix in scalar
            longitudinal sub-model,with only time-independent covariate
        matrix[D,Kt] BS;             // B-spline evaluation matrix
        matrix[D,Ka] BSa;            // Spline evaluation matrix for associate function \alpha(s);
             BSa = [1,...,1] in the simulation.
        matrix[Ka, Kt] M_mat;        // Integrate t(BSa)*BS ds
        vector[p_surv] W[I];         // covariate in survival submodel
        real<lower=0> time[I];       // survival time
        int<lower=0> event[I];       // censor indicator
        cov_matrix[Kt] PenMat;       // prior precision matrix for spline effects (penalty matrix)
        vector[Kp+1] zero;           // design matrix of zero
        real by;                     //1/D

        // apply truncated power series spline (optional)
        // int<lower=0> p_VR;        // number of knots for truncated power series spline
        // vector[p_VR] VR[obs_long];  // design matrix of truncated power series spline on one
            covariate
```

```
}

transformed data {
    vector[Kt] mu_beta;              // prior mean for spline effects
       for (k in 1:Kt) {
              mu_beta[k] <- 0;
       }
}

parameters {
       matrix[p_fun_long,Kt] beta;  // matrix of fixed effect spline coefficients in functional
           sub-model
       vector<lower=-10, upper=10>[p_scalar_long] beta_scale;  // matrix of fixed effect
           coefficients in scalar sub-model
       vector[Ka] alpha;            // association function, constant value in simulation
       real alpha_scalar;           // association parameter
       matrix[Kp,Kt] beta_psi;      // matrix of fPC spline coefficients
       vector[Kp+1] U[I];           // matrix of random effect and fPC loadings (fPC scores)
       vector<lower=0>[Kp] lambda;  //fPC eigenvalue
       real<lower=0> var_b;         //variance of random effect in scalar sub-model
    real<lower=-1, upper=1> rho;
       vector[p_surv] gamma;        // vector of coefficients in survival
       vector<lower=0,upper=100>[p_fun_long] beta_sig; // tuning variance
       vector<lower=0,upper=100>[Kp] psi_sig;      // tuning variance
       real<lower=0,upper=10> tau_alpha;           // precision parameter for association function
       real<lower=0,upper=10> tau_alpha_scalar;    //precision parameter for association parameter
       real<lower=0,upper=10> tau_e;  // precision parameter for error term in functional sub-
           model
       real<lower=0,upper=10> tau_e_scalar;        // precision parameter for error term in scalar
            sub-model
       real logscale;               // baseline log hazard

       // // apply truncated power series spline (optional)
       // vector<lower=-20, upper=20>[p_VR] zeta;         // spline coefficients of truncated
           power series spline
       // real<lower=0> var_spline_zeta;
}

transformed parameters {
       vector<lower=0.01>[p_fun_long] beta_tau2;  // tuning parameter
       real<lower=0> sigma_alpha;
       real<lower=0> sigma_alpha_scalar;
       vector<lower=0.01>[Kp] psi_tau2;    // tuning parameter
       vector[D] Y_fun_est[obs_long];      //estimated function outcome
       real Y_scalar_est[obs_long];        //estimated scalar outcome
       real eta_surv[I];                   //estimated survival part
       real<lower=0> sigma_b;
       vector<lower=0>[Kp] sigma_lambda;
```

```
real<lower=0> sigma_e;
real<lower=0> sigma_e_scale;
cov_matrix[Kp+1] Sigma_U;           // covariate matrix of

// // apply truncated power series spline (optional)
// real<lower=0> sigma_spline_zeta;
// sigma_spline_zeta <- pow(var_spline_zeta, 0.5);

// construct longitudinal sub-models
for (ob in 1:obs_long) {
       Y_fun_est[ob] <- (BS * beta') * X[ob] +  // fixed effects
                             (BS * beta_psi') * U[subj_long[ob], 2:(Kp+1)];    // subject-
                                 level PC effects
       Y_scalar_est[ob] <- beta_scale'*X_scalar[ob] + U[subj_long[ob], 1];

       // or apply truncated power series spline (optional)
       // Y_scalar_est[ob] <- beta_scale'*X_scalar[ob] + zeta'*VR[ob] + U[subj_long[ob],
           1];
}

// construct survival sub-models
for (i in 1:I){
        eta_surv[i] <- gamma'*W[i] + alpha'*BSa'*((BS * beta[1:(p_fun_long-1)]') * X_base[
            i]+(BS * beta_psi') *U[i, 2:(Kp+1)])*by + alpha_scalar*(beta_scale[1:(
            p_scalar_long-1)]'*X_scalar_base[i] + U[i, 1]);
}

// variance
for(p in 1:p_fun_long) {
       beta_tau2[p] <- pow(beta_sig[p], -1);
}
sigma_alpha <- pow(tau_alpha, -0.5);
sigma_alpha_scalar <- pow(tau_alpha_scalar, -0.5);
for(k in 1:Kp) {
       psi_tau2[k] <- pow(psi_sig[k], -1);
}
sigma_e <- pow(tau_e, -0.5);
sigma_e_scale <- pow(tau_e_scale, -0.5);
sigma_b <- pow(var_b, 0.5);
for(kcur in 1:Kp) {
       sigma_lambda[kcur] <- pow(lambda[kcur], 0.5);
}

// construct covariance matrix for random effects
for(i in 1:(Kp+1)){
       for(j in 1:(Kp+1)){
              Sigma_U[i,j] <- 0;
       }
```

```
        }
        Sigma_U[1,1] <- var_b;
        for(i in 2:(Kp+1)){
                Sigma_U[i,i] <- lambda[i-1];
        }
        Sigma_U[1,2] <- rho*sigma_lambda[1]*sigma_b;
        Sigma_U[2,1] <- Sigma_U[1,2];
}

model {

        real h[I];
        real S[I];
        real LL[I];

        U ~ multi_normal(zero, Sigma_U);

        //////////////////////////////////////
        // Prior distributions
        //////////////////////////////////////

        // Prior for variance components controlling smoothness in beta
        for (p in 1:p_fun_longa) {
                beta_sig[p] ~ inv_gamma(.001,.001);
        }

        // Prior for variance components controlling smoothness in psi
        for (k in 1:Kp) {
                psi_sig[k] ~ inv_gamma(.001,.001);
        }

        // Prior for variance components of association parameters
        tau_alpha ~ gamma(.001,.001);
        tau_alpha_scalar ~ gamma(.001,.001);


        // Prior for spline coefficients for beta
        for (p in 1:p_fun_longa) {
                (beta[p])’ ~ multi_normal_prec(mu_beta, beta_tau2[p] * PenMat);
        }

        // Prior for association parameters
        alpha~ normal(0, sigma_alpha);
        alpha_scalar ~ normal(0,sigma_alpha_scalar);

        // Prior for spline coefficients for psi
        for (k in 1:Kp) {
                (beta_psi[k])’ ~ multi_normal_prec(mu_beta, psi_tau2[k] * PenMat);
```

```
       }

       ////////////////////////////////////
       // Outcome likelihood
       ////////////////////////////////////
       for (ob in 1:obs_long) {
              for(d in 1:D){
                     // functional longitudinal sub-model
                     Y[ob, d] ~ normal(Y_fun_est[ob, d], sigma_e);
              }
       }

       // scalar longitudinal sub-model
       Y_scalar ~ normal(Y_scalar_est, sigma_e_scale);

       for(i in 1:I){
              h[i] <- exp(logscale + eta_surv[i] + alpha'*M_mat*beta[p_fun_long]'*time[i]+
                  alpha_scalar*beta_scale[p_scalar_long]*time[i]);
              S[i] <- exp(-exp(logscale + eta_surv[i])*(exp(alpha'*M_mat*beta[p_fun_long]'*time[i
                  ] + alpha_scalar*beta_scale[p_scalar_long]*time[i])-1)/(alpha'*M_mat*beta[
                  p_fun_long]'+alpha_scalar*beta_scale[p_scalar_long]));
              LL[i] <- log(pow(h[i],event[i])*S[i]);
  }

   increment_log_prob(LL);

       // Prior for other parameters
       tau_e ~ gamma(0.1,0.1);
       tau_e_scale ~ gamma(0.1,0.1);
       logscale ~ normal(0, 10);
       beta_scale ~ normal(0,10);
       gamma ~ normal(0, 10);
       lambda ~ inv_gamma(0.01, 0.01);
       var_b ~ inv_gamma(0.01, 0.01);
       rho ~ uniform(-1, 1);

       // // apply truncated power series spline (optional)
       // // first order random walk prior is specified as
       // var_spline_zeta ~ inv_gamma(0.01, 0.01);
       // zeta[1] ~ normal(0, sd_spline_zeta);
       // for(i in 2:p_VR){
       //     zeta[i] ~ normal(zeta[i-1], sd_spline_zeta);
       // }
}
```

References

[1] Shi J, Thompson PM, Gutman B, Wang Y, Initiative ADN, et al. Surface fluid registration of conformal representation: Application to detect disease burden and genetic influence on hippocampus. NeuroImage. 2013;78:111–134.

[2] Colom R, Stein JL, Rajagopalan P, Martínez K, Hermel D, Wang Y, et al. Hippocampal structure and human cognition: Key role of spatial processing and evidence supporting the efficiency hypothesis in females. Intelligence. 2013;41(2):129–140.

[3] Luders E, Thompson PM, Kurth F, Hong JY, Phillips OR, Wang Y, et al. Global and regional alterations of hippocampal anatomy in long-term meditation practitioners. Human Brain Mapping. 2013;34(12):3369–3375.

[4] Monje M, Thomason ME, Rigolo L, Wang Y, Waber DP, Sallan SE, et al. Functional and structural differences in the hippocampus associated with memory deficits in adult survivors of acute lymphoblastic leukemia. Pediatric Blood & Cancer. 2013;60(2):293–300.

[5] Shi J, Wang Y, Ceschin R, An X, Lao Y, Vanderbilt D, et al. A multivariate surface-based analysis of the putamen in premature newborns: regional differences within the ventral striatum. PLOS ONE. 2013;8(7):e66736.

[6] Patenaude B, Smith SM, Kennedy DN, Jenkinson M. A Bayesian model of shape and appearance for subcortical brain segmentation. NeuroImage. 2011;56(3):907–922.

[7] Jenkinson M, Beckmann CF, Behrens TE, Woolrich MW, Smith SM. FSL. NeuroImage. 2012;62(2):782–790.