A Appendix: simulation code and model files

A.1 R code to simulate data

```
################################################################
############ function to simulate survival time ###############
################################################################
# survival function is given by: S(t)= exp(- exp(B) * (exp(A*t) - 1) ) / A)
sim_Ti = function(n=250, alpha1, alpha2, delta=c(1,1), gamma=c(1,1)){
  # random effects
  Sigma=matrix(c(0.09, 0.09*0.16, 0.09*0.16, 0.09), nrow=2, byrow=T)
  u=rmvnorm(n,c(0,0),Sigma)
  H = matrix(rnorm(2*n),n,2)
  W = matrix(rnorm(2*n),n,2)

  # define the survival function
  surv = function(t) {
    if(alpha1==0 & alpha2==0) {res=t*exp(W%*%gamma)}
    else{
      res = ((exp(alpha2*u[,1]+alpha2*t*u[,2]+alpha1*delta[1]*H[,1]+alpha1*
    delta[2]*H[,2]*t+W%*%gamma)-exp(alpha1*H[,1]*delta[1]+W%*%gamma+alpha2*u
    [,1]))/(alpha2*u[,2]+alpha1*H[,2]*delta[2]))
    }

    return(exp(-res))
  }

  rnd =runif(n)
  Ti = rep(Inf,n)
  # w = which(surv(1e8)-rnd < 0)
  w = which(1-surv(101)-rnd > 0)
  Ti[w] = sapply(w,function(j) uniroot(function(x) 1-surv(x)[j]-rnd[j],lower
    =0,upper=101)$root)
  Ti[Ti==0] = min(Ti[Ti>0])/2
  Ci = 5*rbeta(n,4,1)
  Delta = as.numeric(Ti < Ci)
  Ti2 = pmin(Ti, Ci)

  list(Ti=Ti2, event=Delta, H=H, W=W, U=u)
}

################################################################
###### function to simulate longitudinal data #################
################################################################
sim_longitudinal_data = function(survival_data=surdata, n=250, time=c(0, 0.25,
     0.5, 0.75, 1, 3), tau, sigma=1, beta=c(1,1), delta=c(1,1)){
  # survival_data - data simulated from survival model
  # n - # of subjects
  # time - time points of observations
  # tau - quantile
  # sigma - scale parameter
```

```
44
45  # random generation of ALD(0,sigma,p)
46  rald = function(n, location=0, scale, p){
47    u = rnorm(n)
48    z = rexp(n)
49    v = scale*z
50    theta = (1-2*p)/(p*(1-p))
51    tau = sqrt(2/(p*(1-p)))
52    sample = theta * z + tau * sqrt(z) * u
53  }
54
55  time = time # at most # = length(time) observations per patient
56  y = matrix(NA, nrow=n, ncol=length(time)) # wide format
57  Ti = survival_data$Ti
58  U = survival_data$U # random effects
59  H = survival_data$H
60  X = cbind(1, rnorm(n))
61  count = sapply(Ti, function(x) sum(x > time)) # number of observations after
      drop-outs
62
63  for (i in 1:n){
64    for (j in 1:count[i]){
65      location = beta %*% X[i, ] + delta %*% c(H[i,1], H[i,2]*time[j]) + U[i,]
      %*% c(1, time[j])
66      y[i,j] = location + rald(1, scale=sigma, p=tau)
67    }
68  }
69
70  list(y = y, X = X, J=count)
71 }
72
73 ###################################################################
74 ###### function to simulate multiple joint data sets ##########
75 ###################################################################
76 sim_multiple_data = function(N, sur_fun=sim_Ti, longi_fun=sim_longitudinal_
     data, alpha, tau){
77  # N - number of data sets to generate
78  # sur_fun - function to simulate survival data
79  # longi_fun - function to sumualte longitudinal data
80  # alpha - association mechanism for JM
81  # tau - quantile
82
83  outdata = vector(mode='list', N)
84  for (i in 1:N){
85    sur_data = sur_fun(alpha1=alpha[1], alpha2=alpha[2])
86    longi_data = longi_fun(sur_data, tau=tau)
87    outdata[[i]] = list(survival_data=sur_data, longitudinal_data=longi_data)
88  }
89  outdata
90 }
```

A.2 JAGS model file

```
model{
  zero[1] <- 0
  zero[2] <- 0
  k1 <- (1-2*qt)/(qt*(1-qt))
  k2 <- 2/(qt*(1-qt))

  for (i in 1:I){
  # prior for random effects
  u[i, 1:2] ~ dmnorm(zero[], precision[,])

  # longitudinal process, BQR mixed model using ALD representation
    for (j in 1:J[i]){
      er[i,j] ~ dexp(sigma)
      mu[i,j] <- u[i,1] + u[i,2]*t[j] + inprod(X[i,], beta[]) + delta[1]*H[i
    ,1] + delta[2]*H[i,2]*t[j] + k1*er[i,j]
      prec[i,j] <- sigma/(k2*er[i,j])
      y[i,j] ~ dnorm(mu[i,j], prec[i,j])
    } #end of j loop

  # survival process, baseline hazard is set to 1
  A[i] <- alpha2*u[i,2] + alpha1*delta[2]*H[i,2]
  B[i] <- alpha1*delta[1]*H[i,1] + alpha2*u[i,1] + inprod(gamma, W[i,])
  S[i] <- exp(- c*exp(B[i])*(pow(exp(A[i]), Ti[i])-1)/A[i])
  h[i] <- c*exp(inprod(gamma, W[i,]) + alpha1*(delta[1]*H[i,1] + delta[2]*H[i
    ,2]*Ti[i]) + alpha2*(u[i,1] + u[i,2]*Ti[i]))
  L[i] <- pow(h[i], event[i])*S[i]/1.0E+08

  # zero trick
  phi[i] <- -log(L[i])
  zeros[i] ~ dpois(phi[i])

  }#end of i loop

  precision[1:2,1:2] <- inverse(Sigma[,])
    Sigma[1,1] <- 1
    Sigma[1,2] <- rho*sig1
    Sigma[2,1] <- Sigma[1,2]
    Sigma[2,2] <- sig1*sig1

  # priors for other parameters
  alpha1 ~ dnorm(0, 0.001)
  alpha2 ~ dnorm(0, 0.001)
  beta[1] ~ dnorm(0, 0.001)
  beta[2] ~ dnorm(0, 0.001)
  delta[1] ~ dnorm(0, 0.001)
  delta[2] ~ dnorm(0, 0.001)
  gamma[1] ~ dnorm(0, 0.001)
  gamma[2] ~ dnorm(0, 0.001)
  sigma ~ dgamma(0.001, 0.001)
```

```
48   rho ~ dunif(-1, 1)
49   sig1 ~  dgamma(0.01, 0.01)
50   c ~ dunif(0.01, 10)
51 }
```