Supplementary Materials for
Modeling Multilevel Nonlinear Treatment-by-Covariate Interactions in Cluster Randomized Controlled Trials using a Generalized Additive Mixed Model

Sun-Joo Cho, Vanderbilt University
Kristopher J. Preacher, Vanderbilt University
Haley E. Yaremych, Vanderbilt University
Matthew Naveiras, Vanderbilt University
Douglas Fuchs, Vanderbilt University
Lynn S. Fuchs, Vanderbilt University

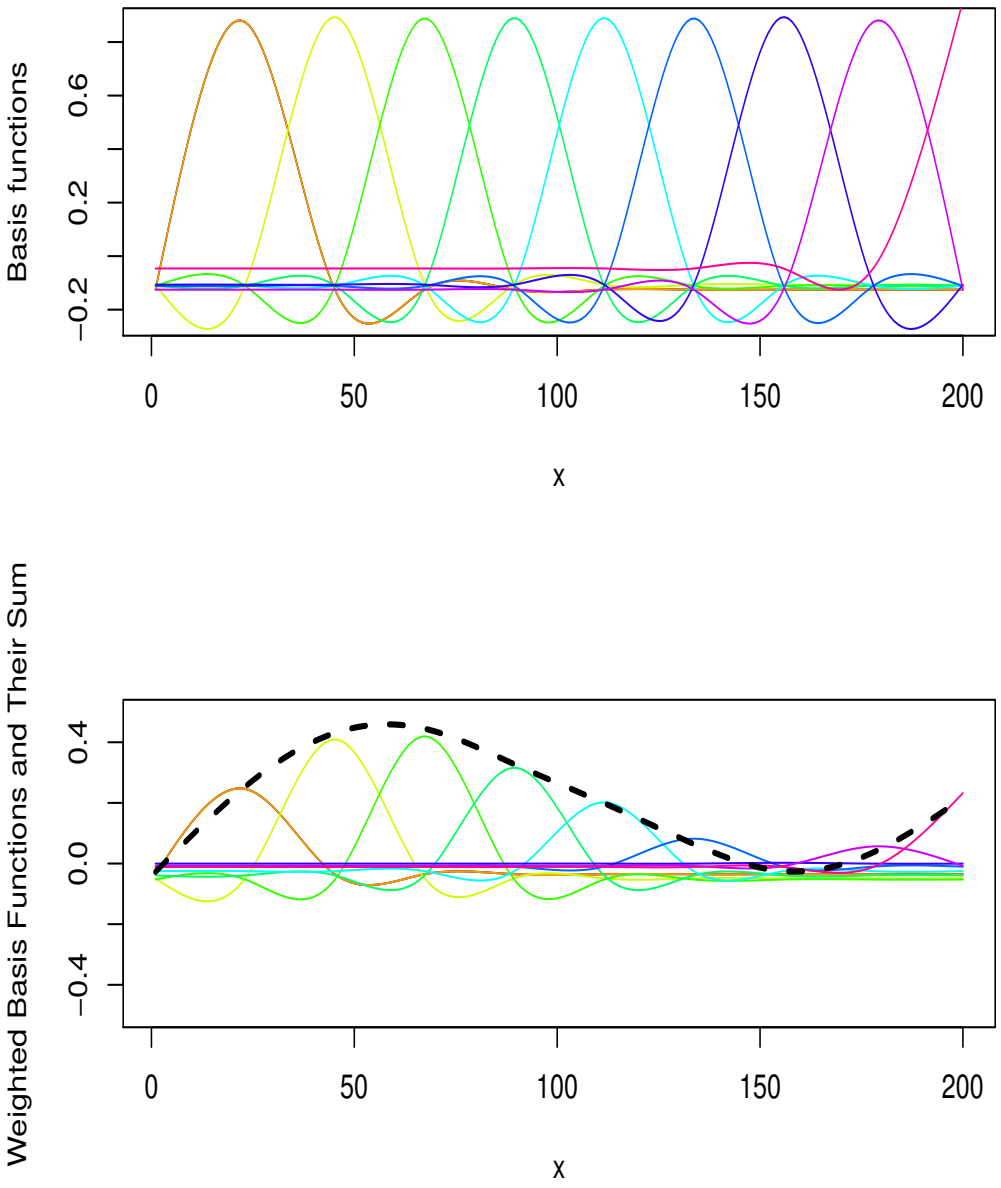
December 29, 2021

The authors' accepted version of the article currently in press
in *British Journal of Mathematical and Statistical Psychology*

Appendix S1.

Illustration of the CRS for a Smooth Function

To illustrate the CRS for a smooth function of $z_j = 0$ or $z_j = 1$, a data set was generated under a normal model with one smooth function of a continuous MOD (x). The figure below (top) illustrates the CRS basis function ($b_k(x)$) with 10 knots evenly distributed through the range of the covariate (x). The figure below (top) presents 9 CRS basis expansions ($10 - 1$ basis functions due to a model identification constraint [each smooth functions must sum to zero over the observed covariate values, $\sum_v f_h(x_{hv}) = 0$]) spread evenly through the range of the covariate. In addition, the figure below shows the same CRS basis functions weighted by the estimated coefficients ($\widehat{\boldsymbol{\delta}} = [0.281, 0.458, 0.472, 0.355, 0.227, 0.092, 0.003, 0.064, 0.244]'$) and the resulting smooth line (dotted line in the figure) created by summing up the weighted basis function at the different values of x (i.e., $\sum_{k=1}^{9} \widehat{\delta}_k b_k(x)$).



Basis functions of the CRS with knots spread evenly through the range of a covariate (top) and the scaled CRS functions weighted by the estimated basis coefficients $\widehat{\boldsymbol{\delta}}$ with the resulting line (dotted line) (bottom).

Appendix S2.

Illustration for the Reformulation of a Smooth Function as a Random Effect

Below, key derivations to reformulate smooth functions as fixed and random effects (Wood, 2004; 2006; 2017, p. 239) are illustrated. The `mgcv` package was used for illustration and its session information is as follows:

```
library(mgcv)
sessionInfo(package = "mgcv")
> sessionInfo(package = "mgcv")
R version 3.6.0 (2019-04-26)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 18363)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
character(0)

other attached packages:
[1] mgcv_1.8-35
```

```
loaded via a namespace (and not attached):
 [1] compiler_3.6.0  minqa_1.2.4     MASS_7.3-51.5   Matrix_1.3-2
 [5] graphics_3.6.0  utils_3.6.0     grDevices_3.6.0 Rcpp_1.0.6
 [9] stats_3.6.0     datasets_3.6.0  splines_3.6.0   nlme_3.1-152
[13] grid_3.6.0      methods_3.6.0   nloptr_1.2.2.2  boot_1.3-27
[17] lme4_1.1-26     base_3.6.0      statmod_1.4.35  lattice_0.20-38
```

For illustration, a data set was generated under a simple normal model with a smooth function rather than the GAMM specification to show the matrices in a concise manner. The simulated data set was generated under a linear normal model having a smooth function $y_v = f_1(x_v) + r_v = exp(2x_v) + r_v, r_v \sim N(0,1), v = 1, \ldots, 100$. The model was generated as follows:

```
set.seed(3412)
x <- runif(100, 0, 1); f1 <- exp(2 * x) #mean(f1)=2.917364 (intercept)
y <- f1 + rnorm(100,0,1)
data <- data.frame(x,y)
```

The model matrix $\mathbf{X}_1$ and the penalty matrix $\mathbf{S}_1$ for the smooth function $f_1(x_v)$ can be extracted using a prediction/construction wrapper function for a smooth function `smoothCon` in the `mgcv` package:

```
smooth.spec.object <- interpret.gam(y~s(x,k=10))$smooth.spec[[1]]
SM <- smoothCon(smooth.spec.object,data=data,knots=NULL,absorb.cons=TRUE)[[1]]
X <- SM$X #model matrix; S <- SM$S[[1]] #penalty matrix
```

In this example, the model matrix $\mathbf{X}_1$ is a 100×9 matrix with 100 observations and 10 basis functions ($K = 10$) and the penalty matrix $\mathbf{S}_1$ is a 9×9 matrix. Because of the identification constraint of the smooth function ($\sum_v f_1(x_{1v}) = 0$), the model matrix $\mathbf{X}_1$ is 100×9 instead of 100×10 and the penalty matrix S is 9×9 instead of 10×10. The eigendecompostion of $\mathbf{S}_1$ can be calculated using the `eigen` function:

```
E <- eigen(S) #penalty matrix
U <- E$vectors #eigenvector matrix
D <- E$values #eigenvalue matrix
nonzeros.D <- which(D > sqrt(.Machine$double.eps)) #nonzero eigenvalue matrix
#Machine$double.eps returns the smallest positive floating-point number x
#such that 1 + x != 1.
U.F <- U[,-nonzeros.D] #design matrix for fixed effects
U.R <- U[,nonzeros.D]  #design matrix for random effects
```

The `nonzeros.D` matrix is for obtaining the eigenvector matix corresponding to positive eigenvalues of $\mathbf{S}_1$. Using `nonzeros.D`, $\mathbf{U}_F$ and $\mathbf{U}_R$ can be easily calculated. With these matrices, $\mathbf{X}_s$ and $\mathbf{Z}_s$ can be calculated as follows:

```
X.s <- X %*% U.F #design matrix for fixed effects
Z.s <- X %*% U.R %*% diag(1/sqrt(D[nonzeros.D]))
#design matrix of random effects
```

In this example, $\mathbf{X}_s$ is a 100×1 matrix and $\mathbf{Z}_s$ is a 100×8 matrix.

Output of the `gamm` function using the generated data set is as follows:

```
> gamm.fit$lme
Linear mixed-effects model fit by maximum likelihood
  Data: strip.offset(mf)
  Log-likelihood: -147.5444
  Fixed: y.0 ~ X - 1
X(Intercept)     Xs(x)Fx1
    2.902320     1.593412


Random effects:
 Formula: ~Xr - 1 | g
```

```
 Structure: pdIdnot
                Xr1          Xr2          Xr3          Xr4          Xr5          Xr6
StdDev: 8.52342e-05 8.52342e-05 8.52342e-05 8.52342e-05 8.52342e-05 8.52342e-05
                Xr7          Xr8 Residual
StdDev: 8.52342e-05 8.52342e-05 1.058132

Number of Observations: 100
Number of Groups: 1
```

The 100×1 matrix $\mathbf{X}_s$ is `Xs(x)Fx1` in the output and the 100×8 matrix $\mathbf{Z}_s$ is $[\texttt{Xr1}, \texttt{Xr2}, \texttt{Xr3}, \texttt{Xr4}, \texttt{Xr5}, \texttt{Xr6}, \texttt{Xr7}, \texttt{Xr8}]'$ in the output. The smooth parameter estimate $\widehat{\lambda}_1$ is calculated as $1/(8.52342e-05)^2 = 1.38E+08$, where $8.52342e-05$ is the `StdDev` under `Random effects` in the output.

The smooth function, $f_1(x_v) = exp(2x_v) + r_v$, is constructed based on estimates of $\widehat{\gamma} = 2.902320$ (`X(Intercept)` result in the output), $\widehat{\gamma}_s = 1.593412$ (`Xs(x)Fx1` result in the output), and $\widehat{\lambda}_1 = 1.593412$:

$$\tilde{f}_1(x_v) = \widehat{\gamma} + \tilde{\mu}_s = 2.902320 + \mathbf{X}_s 1.593412 + \mathbf{Z}_s \tilde{\mathbf{u}}_s, \tag{1}$$

where $\mathbf{u}_s \sim MN(\mathbf{0}, (\widehat{\lambda}_1 \tilde{\mathbf{S}}_1)^{-1})$ with $\tilde{\mathbf{S}}_1 = \mathbf{U}_R^T \mathbf{S}_1 \mathbf{U}_R$. Equation 1 can be calculated using `R` as follows:

```
#extacting fixed effects
fixed <- as.matrix(coef(gamm.fit$lme))
gamma <- fixed[1] #2.902320
slope.Xs <- fixed[2] #1.593412

#smooth function reconstruction
Stilde <- t(U.R) %*% S %*% U.R
Sigma <- solve(8.52342e-05* Stilde)
```

```
U.s <- mvrnorm(100,mu=rep(0, 8),Sigma)
f1.fixed <-  X.s * slope.Xs   #100 X 1 matrix
f1.random <- Z.s %*% t(U.s) #100 X 8 matrix
f1.fitted <-  gamma + f1.fixed + rowSums(f1.random)  #100 X 1 matrix
```

In this example, the predicted smooth function $\tilde{f}_1(x_v)$ is close to the true smooth function $f_1(x_v)$ ($Corr(\tilde{f}_1(x_v), f_1(x_v)) = 0.9645481$).

Here, one can see that there are two methods to construct a smooth function $f_h(x)$. As illustrated above, the first method is to obtain $\tilde{f}_h(x)$ using $\widehat{\gamma}$, $\widehat{\gamma}_s$, and $\widehat{\lambda}$, as random effects in GLMM. The second method is to estimate the basis coefficients ($\boldsymbol{\delta}_h$) directly along with other model parameters of GAMM. When the first method is used as in this study, the basis coefficients ($\boldsymbol{\delta}_1$) can be calculated as follows, based on the predicted outcome $\tilde{y}_r = \tilde{f}_1(x_v) + \tilde{r}_v$ and known basis functions $\mathbf{b}_1(x_1)$:

```
SM <- smoothCon(smooth.spec.object,data=data,knots=NULL,absorb.cons=TRUE)[[1]]
X <- SM$X #basis functions
predicted.y <- data.frame(fitted(gamm.fit$lme))
delta <- coef(lm(predicted.y[,1]~X-1))
delta
```

In this example, 9 basis coefficients were calculated as
$\widehat{\boldsymbol{\delta}}_1 = [-2.672771e{-}09, 3.768338e{-}09, -1.009072e{-}09, -4.907397e{-}09, 1.789757e{-}10, -5.078812e{-}09, -2.014012e-09, -2.390540e-08, 1.593412e+00]'$.

Appendix S3.

R Code Used for the Empirical Study

```
library(nlme) #for the "gamm" function
library(mgcv) #for the "gamm" function
library(mgcViz) #for figures of smooth functions
library(lme4) #for the "lmer" function
library(ggplot2) #for figures
library(robumeta) #for calculating cluster means
library(fastDummies) #for creating dummy variables of school ids
library(car) #for Q-Q plots
library(itsadug)  #for the find_difference function
library(psych) #for calculating quantiles


####################
##Data Management##
####################

#491 students in PALS fluency student data
data <- read.table("C:\\data.txt",header=T)

#variables in use
#schlnum tchrnum  PAMOD prePAMOD treatmnt

#dummy variables of school ids
dummyschs <- dummy_cols(data$schlnum,remove_first_dummy = TRUE)

names(dummyschs)[1] <- "schid"
names(dummyschs)[2] <- "schid2"
names(dummyschs)[3] <- "schid3"
names(dummyschs)[4] <- "schid4"
names(dummyschs)[5] <- "schid5"
names(dummyschs)[6] <- "schid6"
names(dummyschs)[7] <- "schid7"
names(dummyschs)[8] <- "schid8"


#level-1 prePA
x1 <- group.center(data$prePAMOD, data$tchrnum)

#level-2 prePA
x2 <- data$prePAMOD - x1

#TRT z_j: PALS-Only (2) and PALS+Fluency groups (3)=1 and a control group (1)=0
attach(data)
data$z[treatmnt==1] <- 0
data$z[treatmnt==2] <- 1
data$z[treatmnt==3] <- 1

#numetric dummy variable for TRT
data$z1 <- as.numeric(data$z)-1
data1 <- cbind(data,dummyschs,dummyrace,x1,x2)
data1$tchrnum <- as.factor(data1$tchrnum)
data1$schlnum <- as.factor(data1$schlnum)
data1$z <- as.factor(data1$z)


##########
##Step 1##
##########

#fixed school effects
model0.fixed <- gamm(PAMOD ~ 1 + schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8,
random=list(tchrnum=~1), data=data1, method="ML")
model0.fixed$lme
```

```
VarCorr(model0.fixed$lme)


##########
##Step 2##
##########

#Figure 2: scatter plots of postPA vs. prePA
p.x1 <- ggplot(data=data1,
aes(x = x1, y = PAMOD, shape=z, color=z)) +
geom_point(aes(colour = factor(z))) +
geom_smooth(aes(linetype="Smooth"), method="loess", se=T, fullrange=F) +
geom_smooth(aes(linetype="Linear"),method="lm", se=F, fullrange=F) +
scale_linetype_manual(name="Smoother Type:",values=c("Smooth"="solid", "Linear"="longdash")) +
xlab("Class-Mean Centered Pre-Treatment Phonological Awareness") +
ylab("Post-Treatment Phonological Awareness") +
theme_bw() +
theme(legend.direction = "horizontal", legend.position = "bottom", legend.key = element_blank(),
        legend.background = element_rect(fill = "white", colour = "gray30")) +
guides(fill = guide_legend(keywidth = 1, keyheight = 1), linetype=guide_legend(keywidth = 3, keyheight = 1),
         colour=guide_legend(keywidth = 3, keyheight = 1))

p.x1 + scale_color_brewer(palette="Dark2")
# Use grey scale
p.x1 + scale_color_grey()



p.x2 <- ggplot(data=data1,
aes(x = x2, y = PAMOD, shape=z, color=z)) +
geom_point(aes(colour = factor(z))) +
geom_smooth(aes(linetype="Smooth"), method="loess", se=T, fullrange=F) +
geom_smooth(aes(linetype="Linear"),method="lm", se=F, fullrange=F) +
scale_linetype_manual(name="Smoother Type:",values=c("Smooth"="solid", "Linear"="longdash")) +
xlab("Class-Means of Pre-Treatment Phonological Awareness") +
ylab("Post-Treatment Phonological Awareness") +
theme_bw() +
theme(legend.direction = "horizontal", legend.position = "bottom", legend.key = element_blank(),
        legend.background = element_rect(fill = "white", colour = "gray30")) +
guides(fill = guide_legend(keywidth = 1, keyheight = 1), linetype=guide_legend(keywidth = 3, keyheight = 1),
         colour=guide_legend(keywidth = 3, keyheight = 1))

p.x2 + scale_color_brewer(palette="Dark2")
# Use grey scale
p.x2 + scale_color_grey()


##model comparisons
#random intercept model
model1 <- gamm(PAMOD ~ 1 + z +  schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8 +
s(x1,by=z,bs="cr",k=10) + s(x2,by=z,bs="cr",k=10), random=list(tchrnum=~1), data=data1)

gam.check(model1$gam)
summary(model1$gam)
summary(model1$lme)
plot(model1$gam, page=1)
vis.gam(model1$gam)
VarCorr(model1$lme)

#basis coefficients
gamma.crs <- as.matrix(coef(model1$gam))
gamma.crs

#K selection using the corrected AIC for model1
model1.12 <- gamm(PAMOD ~ 1 + z +  schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8 +
s(x1,by=z,bs="cr",k=12) + s(x2,by=z,bs="cr",k=12), random=list(tchrnum=~1), data=data1)
summary(model1.12$lme)
logLik.gam(model1.12$gam)
```

```
model1.14 <- gamm(PAMOD ~ 1 + z +  schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8 +
s(x1,by=z,bs="cr",k=14) + s(x2,by=z,bs="cr",k=14), random=list(tchrnum=~1), data=data1)
summary(model1.14$lme)
logLik.gam(model1.14$gam)


model1.16 <- gamm(PAMOD ~ 1 + z +  schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8 +
s(x1,by=z,bs="cr",k=16) + s(x2,by=z,bs="cr",k=16), random=list(tchrnum=~1), data=data1)
summary(model1.16$lme)
logLik.gam(model1.16$gam)



#random intercept and slope model
model2 <- gamm(PAMOD ~ 1 + z +  schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8 +
s(x1,by=z,bs="cr",k=10) + s(x2,by=z,bs="cr",k=10), random=list(tchrnum=~1+x1), data=data1)

gam.check(model2$gam)
summary(model2$gam)
summary(model2$lme)
plot(model2$gam, page=1)
vis.gam(model2$gam)
VarCorr(model2$lme)

#AIC and BIC calculations
anova(model1$lme,model2$lme)

#root mean squared error as a model-data fit index (RMSEI)
#random intercept GAMM
model1.std.con <- data.frame(std.con=resid(model1$lme,type="p",level=1))
model1.mean <- mean(model1.std.con[,1]^2)
model1.rmse <- sqrt(model1.mean)

#random-intercept-and-slope GAMM
model2.std.con <- data.frame(std.con=resid(model2$lme,type="p",level=1))
model2.mean <- mean(model2.std.con[,1]^2)
model2.rmse <- sqrt(model2.mean)


##########
##Step 3##
##########

#model1=a selected model

#Figure 3(a)
quantile(data1$x2,probs = c(0.1,0.25,0.5,0.75,0.9))

x2.1 <- -0.5376471
x2.2 <- -0.2214286
x2.3 <- -0.0287500
x2.4 <- 0.3942857
x2.5 <- 0.5806667

gamm.gamma00 <- 0.3841851
gamm.gamma02 <- 0.4386390

newdata.z0 <- data.frame(z=c(0,0,0,0,0),schid1=c(0,0,0,0,0),schid2=c(0,0,0,0,0),schid3=c(0,0,0,0,0),
schid4=c(0,0,0,0,0),schid5=c(0,0,0,0,0),schid6=c(0,0,0,0,0),schid7=c(0,0,0,0,0),schid8=c(0,0,0,0,0),
x1=c(0,0,0,0,0),x2=c(x2.1,x2.2,x2.3,x2.4,x2.5))

newdata.z1 <- data.frame(z=c(1,1,1,1,1),schid1=c(0,0,0,0,0),schid2=c(0,0,0,0,0),schid3=c(0,0,0,0,0),
schid4=c(0,0,0,0,0),schid5=c(0,0,0,0,0),schid6=c(0,0,0,0,0),schid7=c(0,0,0,0,0),schid8=c(0,0,0,0,0),
x1=c(0,0,0,0,0),x2=c(x2.1,x2.2,x2.3,x2.4,x2.5))

predict.z0 <- predict(model1$gam,newdata.z0,type="iterms",se=TRUE)
predict.z1 <- predict(model1$gam,newdata.z1,type="iterms",se=TRUE)
```

```
predict.z0$fit[,11]
predict.z1$fit[,12]

gamm.1.y.z0 <- gamm.gamma00 + predict.z0$fit[1,11]
gamm.1.y.z1 <- gamm.gamma00 + gamm.gamma02 + predict.z1$fit[1,12]

gamm.2.y.z0 <- gamm.gamma00 + predict.z0$fit[2,11]
gamm.2.y.z1 <- gamm.gamma00 + gamm.gamma02 + predict.z1$fit[2,12]

gamm.3.y.z0 <- gamm.gamma00 + predict.z0$fit[3,11]
gamm.3.y.z1 <- gamm.gamma00 + gamm.gamma02 + predict.z1$fit[3,12]

gamm.4.y.z0 <- gamm.gamma00 + predict.z0$fit[4,11]
gamm.4.y.z1 <- gamm.gamma00 + gamm.gamma02 + predict.z1$fit[4,12]

gamm.5.y.z0 <- gamm.gamma00 + predict.z0$fit[5,11]
gamm.5.y.z1 <- gamm.gamma00 + gamm.gamma02 + predict.z1$fit[5,12]


z <- rbind("Control","Treatment","Control","Treatment","Control","Treatment","Control","Treatment","Control","Treatment")
x2 <- rbind("0.1","0.1","0.25","0.25","0.5","0.5","0.75","0.75","0.9","0.9")
gamm.y <- rbind(gamm.1.y.z0,gamm.1.y.z1,gamm.2.y.z0,gamm.2.y.z1,gamm.3.y.z0,gamm.3.y.z1,
gamm.4.y.z0,gamm.4.y.z1,gamm.5.y.z0,gamm.5.y.z1)
gamm.y <- round(gamm.y, digits = 3)
gamm.y.z <- data.frame(cbind(z,x2,gamm.y))
names(gamm.y.z) <- c("z","x.j","gamm.y")
gamm.y.z$gamm.y <- as.numeric(gamm.y)
gamm.y.z


g.y.z <- ggplot(gamm.y.z, aes(x=z, y=gamm.y, group=x.j, color=x.j)) +
         geom_line(aes(linetype=x.j), size = 1.5) +
         geom_point(aes(shape=x.j), size = 4) +
   ylab("Class-Means of Post-Treatment Phonological Awareness") +
         scale_y_continuous(breaks=seq(-0.8, 1.2, 0.2),limits=c(-0.8, 1.2))

g.y.z + scale_color_brewer(palette="Dark2")
# Use grey scale
g.y.z + scale_color_grey()
g.y.z + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),panel.background = element_blank(),
axis.line = element_line(colour = "black"))


#Figure 3(b)
vcov <- as.matrix(vcov(model1$gam))
vcov

coef <- coef(model1$gam)
coef

#level2 simple effect=gamma01 + x2.diff vs. x2
x2.trt.mean <- x2.dat$y + coef[2]
x2.trt.se <- x2.se + sqrt(vcov[2,2])

x2.trt.data <- data.frame(x2.trt.mean,x2.trt.se,x2.x)

gamm.fit.x2 <- ggplot(x2.trt.data, aes(x2.x, y=x2.trt.mean)) + geom_line() +
      geom_ribbon(aes(ymin = x2.trt.mean - 1.96*x2.trt.se, ymax = x2.trt.mean + 1.96*x2.trt.se),
      linetype="dashed",alpha=0.2,fill="white",colour="black") +
      geom_vline(xintercept = c(-0.4945455,0.5680808), color="red", linetype = 3, size=1.5) +
      ylim(-3,3) +
      geom_hline(yintercept = 0,linetype="dashed") +
xlab("Class-Means of Pre-Treatment Phonological Awareness") +
ylab("Treatment Effect on Post-Treatment Phonological Awareness") +
theme_bw() +
theme(legend.direction = "horizontal", legend.position = "bottom", legend.key = element_blank(),
      legend.background = element_rect(fill = "white", colour = "gray30")) +
guides(fill = guide_legend(keywidth = 1, keyheight = 1), linetype=guide_legend(keywidth = 3, keyheight = 1),
```

```
      colour=guide_legend(keywidth = 3, keyheight = 1))

gamm.fit.x2 + scale_color_brewer(palette="Dark2")
# Use grey scale
gamm.fit.x2 + scale_color_grey()
gamm.fit.x2 + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),panel.background = element_blank(),
axis.line = element_line(colour = "black"))

find_difference(x2.trt.mean,x2.trt.se,x2.x)



#########
## MLM ##
#########

#linear effect comparisons: random intercept model
model1.lin <- lme(PAMOD ~ 1 + x1*z + x2*z + schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8,
random=list(tchrnum=~1), data=data1, method="ML")
summary(model1.lin)

#linear effect comparisons: random intercept-slope model
model2.lin <- lme(PAMOD ~ 1 + x1*z + x2*z + schid2 + schid3 + schid4 + schid5 + schid6 + schid7 + schid8,
random=list(tchrnum=~1+x1), data=data1, method="ML")
summary(model2.lin)


# Figure 3(c)
quantile(data1$x2,probs = c(0.1,0.25,0.5,0.75,0.9))

x2.1 <- -0.5376471
x2.2 <- -0.2214286
x2.3 <- -0.0287500
x2.4 <- 0.3942857
x2.5 <- 0.5806667


y.z0.1 <- gamma00 + gamma01*x2.1
y.z1.1 <- gamma00 + gamma01*x2.1 + gamma02 + gamma03*x2.1

y.z0.2 <- gamma00 + gamma01*x2.2
y.z1.2 <- gamma00 + gamma01*x2.2 + gamma02 + gamma03*x2.2

y.z0.3 <- gamma00 + gamma01*x2.3
y.z1.3 <- gamma00 + gamma01*x2.3 + gamma02 + gamma03*x2.3

y.z0.4 <- gamma00 + gamma01*x2.4
y.z1.4 <- gamma00 + gamma01*x2.4 + gamma02 + gamma03*x2.4


y.z0.5 <- gamma00 + gamma01*x2.5
y.z1.5 <- gamma00 + gamma01*x2.5 + gamma02 + gamma03*x2.5


#z <- rbind(0,1,0,1,0,1)
z <- rbind("Control","Treatment","Control","Treatment","Control","Treatment","Control","Treatment","Control","Treatment")
x2 <- rbind("0.1","0.1","0.25","0.25","0.5","0.5","0.75","0.75","0.9","0.9")
y <- rbind(y.z0.1,y.z1.1,y.z0.2,y.z1.2,y.z0.3,y.z1.3,y.z0.4,y.z1.4,y.z0.5,y.z1.5)
y <- round(y, digits = 3)

mlm.y.z <- data.frame(cbind(z,x2,y))
names(mlm.y.z) <- c("z","x.j","y")
mlm.y.z$y <- as.numeric(y)
mlm.y.z


y.z <- ggplot(mlm.y.z, aes(x=z, y=y, group=x.j, color=x.j)) +
       geom_line(aes(linetype=x.j), size = 1.5) +
```

```
       geom_point(aes(shape=x.j), size = 4) +
       ylab("Class-Means of Post-Treatment Phonological Awareness") +
       scale_y_continuous(breaks=seq(-0.8, 1.2, 0.2),limits=c(-0.8, 1.2))

y.z + scale_color_brewer(palette="Dark2")
# Use grey scale
y.z + scale_color_grey()
y.z + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),panel.background = element_blank(),
axis.line = element_line(colour = "black"))


#Figure 3(d)
gamma00 <- 0.3180692
gamma10 <- 0.6929973
gamma01 <- 0.5366350
gamma02 <- 0.4634864
gamma03 <- -0.1834441
gamma11 <- -0.2118012

#var-cov estimates of fixed effects
mlm.vcov <- vcov(model1.lin)
mlm.vcov

var.gamma00 <- mlm.vcov[1,1]
var.gamma10 <- mlm.vcov[2,2]
var.gamma01 <- mlm.vcov[4,4]
var.gamma02 <- mlm.vcov[3,3]
var.gamma03 <- mlm.vcov[13,13]
var.gamma11 <- mlm.vcov[12,12]
cov.gamma10.11 <- mlm.vcov[2,12]
cov.gamma02.01 <- mlm.vcov[3,4]
cov.gamma02.03 <- mlm.vcov[3,13]
cov.gamma01.03 <- mlm.vcov[4,13]

x1 <- data1$x1
x2 <- data1$x2
z1 <- data1$z1

#estimate
data1$omega.B <- gamma02+gamma03*x2

#SE
data1$SE.B <- sqrt(var.gamma02^2+var.gamma03*x2^2+2*x2*cov.gamma02.03)


#Figure 3(d)
mlm.fit.x2 <- ggplot(data1, aes(x=x2, y=omega.B)) + geom_line() +
      geom_ribbon(aes(ymin = omega.B - 1.96*SE.B, ymax = omega.B + 1.96*SE.B),
      linetype="dashed",alpha=0.2,fill="white",colour="black") +
      geom_vline(xintercept = c(-1.690,0.800), linetype="dotted", color = "red",  linetype = 3, size=1.5) +
      ylim(-3,3) +
      geom_hline(yintercept = 0,linetype="dashed") +
xlab("Class-Means of Pre-Treatment Phonological Awareness") +
ylab("Treatment Effect on Post-Treatment Phonological Awareness") +
theme_bw() +
theme(legend.direction = "horizontal", legend.position = "bottom", legend.key = element_blank(),
      legend.background = element_rect(fill = "white", colour = "gray30")) +
guides(fill = guide_legend(keywidth = 1, keyheight = 1), linetype=guide_legend(keywidth = 3, keyheight = 1),
      colour=guide_legend(keywidth = 3, keyheight = 1))

mlm.fit.x2 + scale_color_brewer(palette="Dark2")
# Use grey scale
mlm.fit.x2 + scale_color_grey()
mlm.fit.x2 + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),panel.background = element_blank(),
axis.line = element_line(colour = "black"))

find_difference(data1$omega.B,data1$SE.B,data1$x2)
```

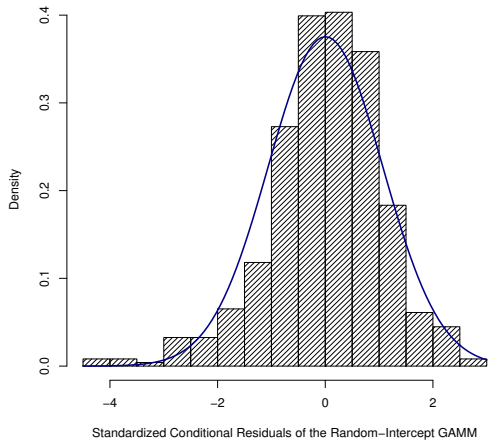
Appendix S4.

Diagnostic Plots of the Selected Random-Intercept GAMM

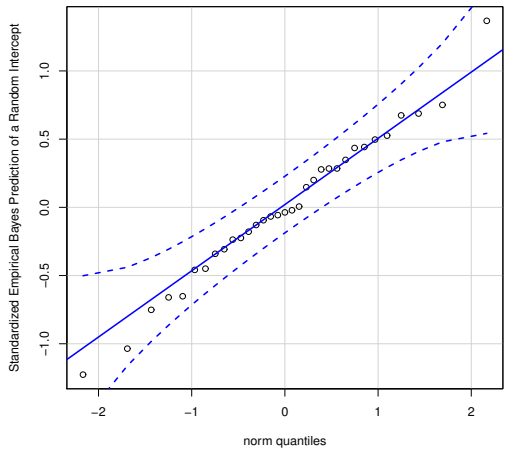
To evaluate model-data fit, residuals of the selected GAMM were calculated. Residuals can be specified at each level of the model (e.g., Hilden-Minton, 1995). For the individual level (level 1), standardized conditional residuals were calculated. The standardized conditional residuals are discrepancies between the observed and fitted values and they indicate how much the observed values deviate from the predicted regression line for a cluster j, by conditioning on the predicted random effects $\tilde{\mathbf{u}}$: $\frac{\mathbf{y}_j - E(\mathbf{y}_j|\tilde{\mathbf{u}})}{Var(\mathbf{y}_j|\tilde{\mathbf{u}})}$. If a selected model is correct, the standardized conditional residuals should approximately have mean zero and variance one. Random effects are considered higher-level (> 1) residuals (e.g., Longford, 1993, pp. 60–61). In this study, standardized empirical Bayes (EB) prediction is used to calculate the higher-level residuals.

Figures below present diagnostic plots of the selected random-intercept GAMM. The mean and variance of standardized conditional residuals were 0 and 1.129. A majority of the standardized conditional residuals (level-1 residuals) of the model are close to 0 (1st quartile= -0.598, 3rd quartile= 0.661, range= $[-4.327, 2.711]$). A histogram of the standardized conditional residuals overlaid with a normal curve (Figure (a)) suggests that the deviance from normality is not large and larger residuals (5 individuals having residuals outside ± 3) were mainly found at the lower end. In addition, a Q-Q plot of standardized EB predictions of the random intercept (Figure (b)) indicates that normality holds for the random intercept. Furthermore, there are no systematic patterns in a plot of standardized conditional residuals vs. the fitted values of the random-intercept GAMM (Figure (c)), which indicates that the assumption of constant variance σ^2 holds. Figure (d) shows the prediction of the random-intercept GAMM for each class. The prediction lines in Figure (d) are not strictly linear because of the smooth functions in GAMM.

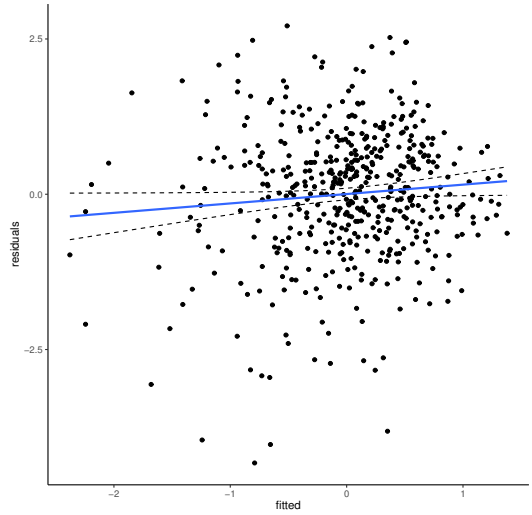
(a) Normal Curve over Histogram of Standardized Conditional Residuals



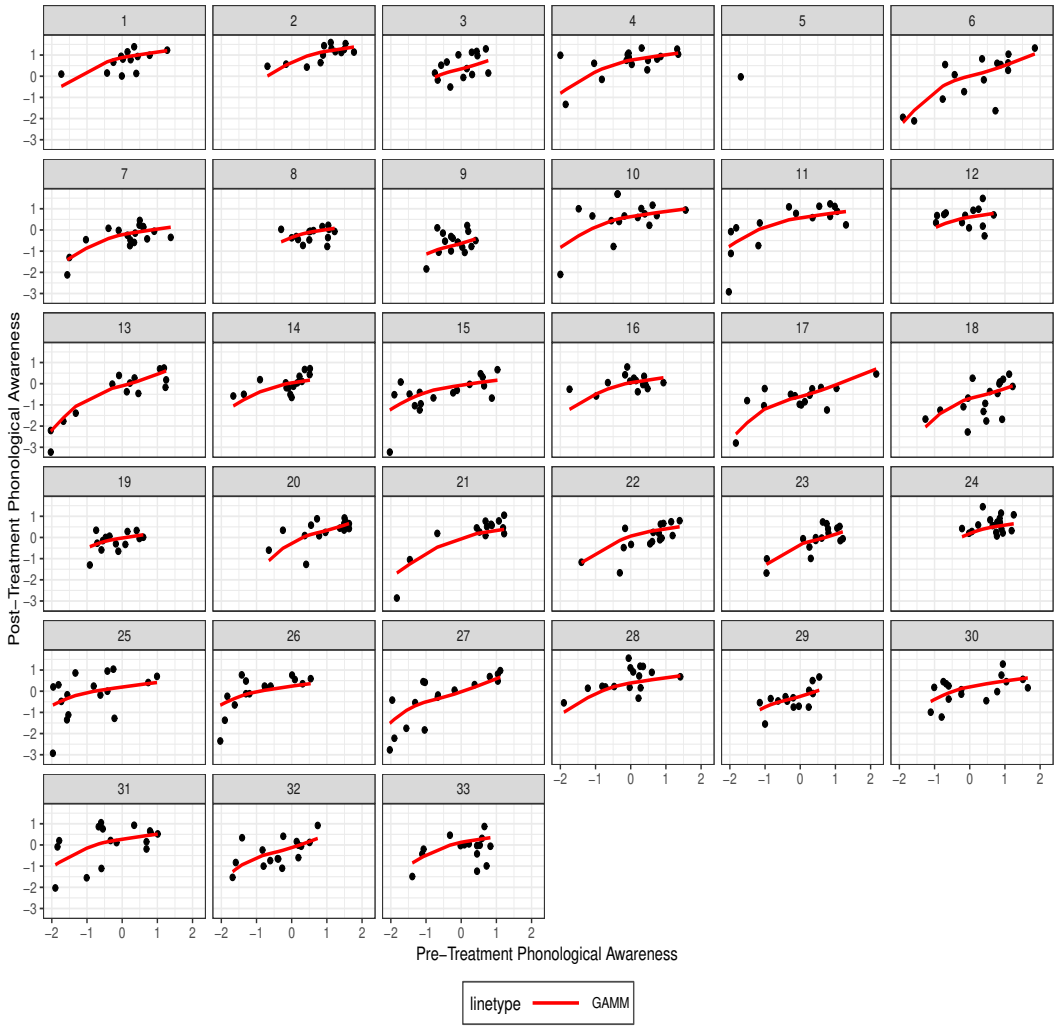
(b) QQ-plot of Standardized EB Predictions of a Random Intercept



(c) Standardized Conditional Residuals vs. Fitted (Predicted) Values



(d) Predictions for Each Class



Appendix S5.

Examples of Generated Functions and Estimation Code of the Simulation Study

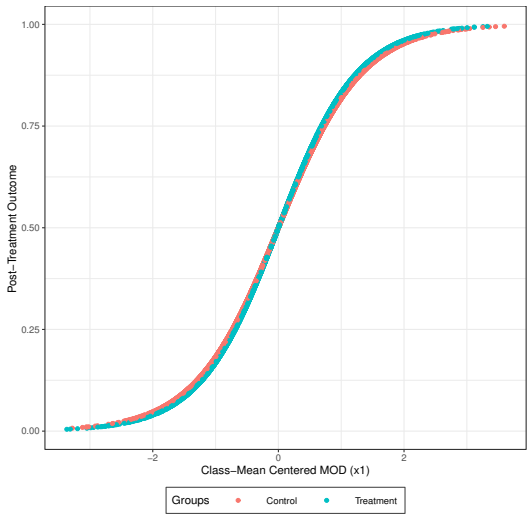
MLM with a Logistic Function as a Data-Generating Model

• An Example of a Generated Logistic Function for TRT × MOD

The following is an example of a generated logistic function for the condition with number of clusters=200, cluster size=30, and $ICC = 0.30$. Parameters were set as follows: $\gamma_{00} = 0$, $\gamma_{10} = 1.5$, $\gamma_{01} = 7.2$, $\gamma_{02} = 1.7$, $\gamma_{03} = 5.1$, and $\gamma_{11} = 0.1$. These parameters were chosen to have no differences in logistic functions between control and treatment groups at level 1 and to have large differences in logistic functions between control and treatment groups at level 2 as expected in a C-RCT:

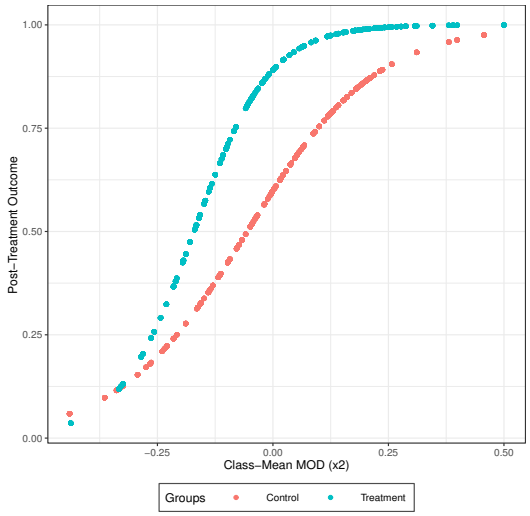
- Level-1 Logistic Function

$$\frac{1}{1 + exp[-\{(\gamma_{10} + \gamma_{11} z_j)(x_{ij} - x_{.j})\}]} \tag{2}$$



- Level-2 Logistic Function

$$\frac{1}{1 + exp[-\{(\gamma_{00} + \gamma_{02} z_j) + (\gamma_{01} + \gamma_{03} z_j) x_{.j}\}]} \tag{3}$$



- Parameter estimation of MLM-Logistic using the `nlme` package in `R`

```
library(nlme)
data <- read.table(file="C:/data1.txt",header=T,fill=T)
data$cluster <- as.factor(data$cluster)
model <- nlme(
    y ~ A + 1/(1 +  exp(-(gamma00 + gamma10*x1 + gamma01*x2 + gamma02*z + gamma03*x2*z + gamma11*x1*z ))),
    #A is assigned for the random intercept and random slope.
    data = data,
    fixed = gamma00+gamma10+gamma01+gamma02+gamma03+gamma11 ~ 1,
    random = list(cluster = A ~ x1),
    start = c(gamma00=0,gamma10=1,gamma01=7,gamma02=2,gamma03=5,gamma11=0),
    method = "ML"
)
```

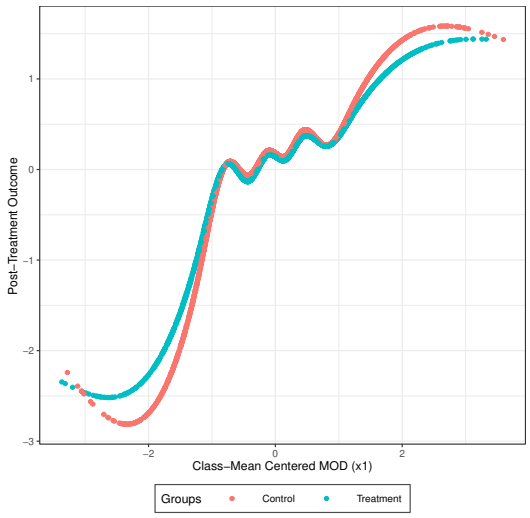
GAMM as a Data-Generating Model

- An Example of Generated Smooth Functions

The following is an example of generating smooth functions for the condition with number of clusters=200, cluster size=30, and $ICC = 0.30$. Nine basis functions ($\mathbf{b}_h$) for CRS with $K = 10$ were extracted using the `interpret.gam` and `smoothCon` functions in the `mgcv` package.

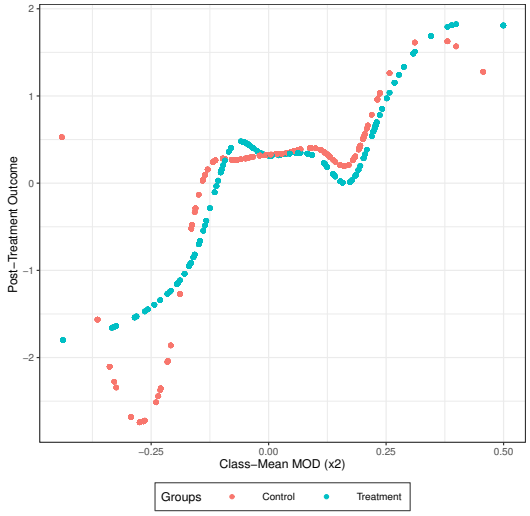
- Level-1 Smooth Functions

The true basis coefficients were set as $[-0.017, 0.238, 0.680, 0.760, 0.667, 1.217, 0.376, 1.840, 1.499]'$ for $f_2(x_{ij} - x_{.j})(z_j = 0)$ and $[0.337, 0.238, 0.680, 0.760, 0.667, 1.217, 0.376, 1.840, 1.499]'$ for $f_2(x_{ij} - x_{.j})(z_j = 1)$. Based on CRS basis functions with $K = 10 - 1$ and the true basis coefficients, the generated level-1 smooth functions are as follows:



- Level-2 Smooth Functions

The true basis coefficients were set as $[-2.290, -0.143, 0.111, 0.169, 0.192, 0.171, 0.172, 0.641, 0.977]'$ for $f_1(x_{.j})(z_j = 0)$ and $[-0.450, -0.333, 0.963, 0.855, 0.863, 0.847, 0.173, 1.536, 1.901]'$ for $f_1(x_{.j})(z_j = 1)$. Based on CRS basis functions with $K = 10 - 1$ and the true basis coefficients, the generated level-2 smooth functions are as follows:



Appendix S6.

Results of the Simulation Study

MLM with a Logistic Function as a Data-Generating Model

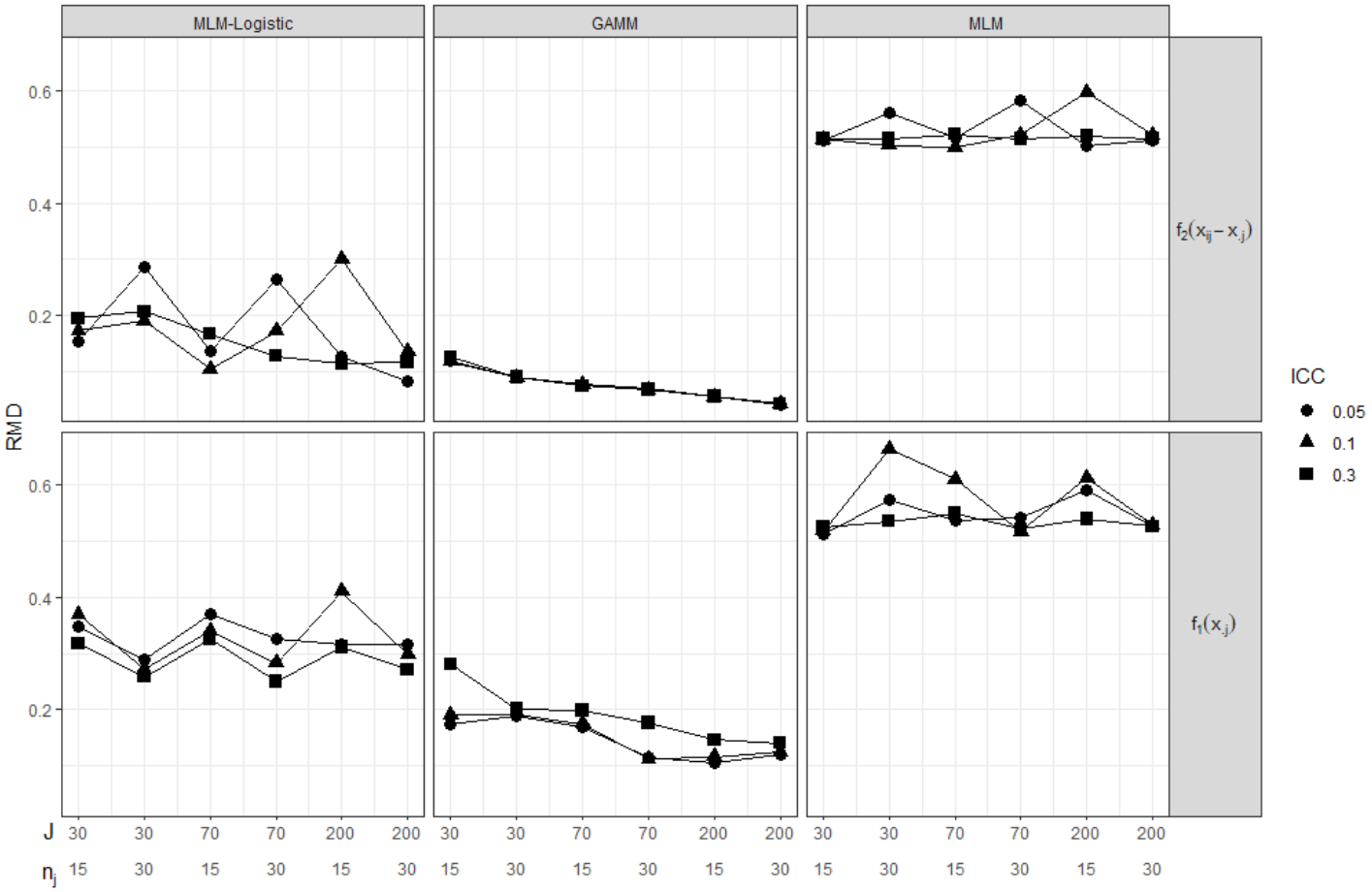


Figure S6.1 RMD in MLM-Logistic, GAMM, and MLM at Level 1 (top) and at Level 2 (bottom).

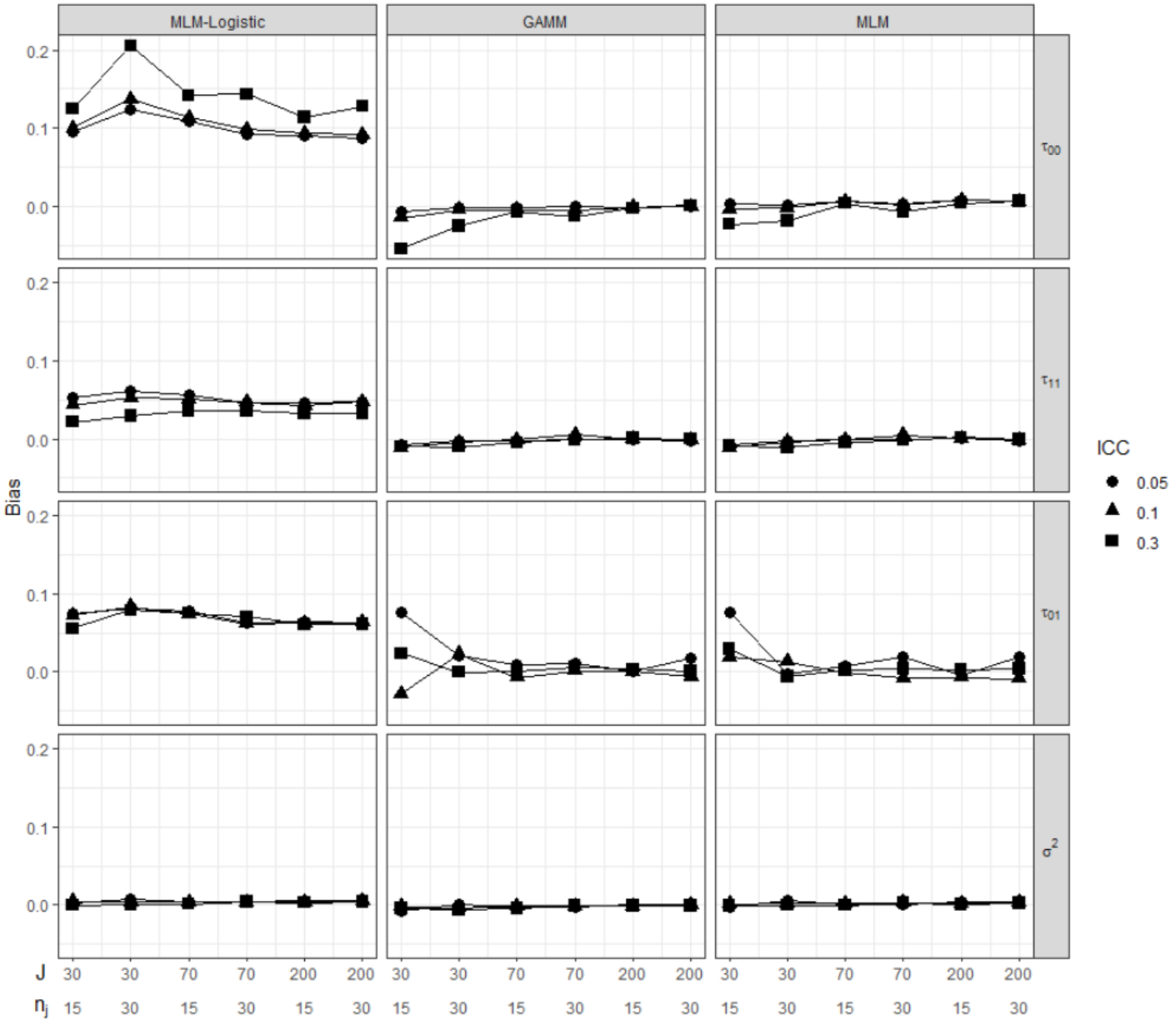


Figure S6.2 Bias of parameter estimates in MLM-Logistic, GAMM, and MLM.

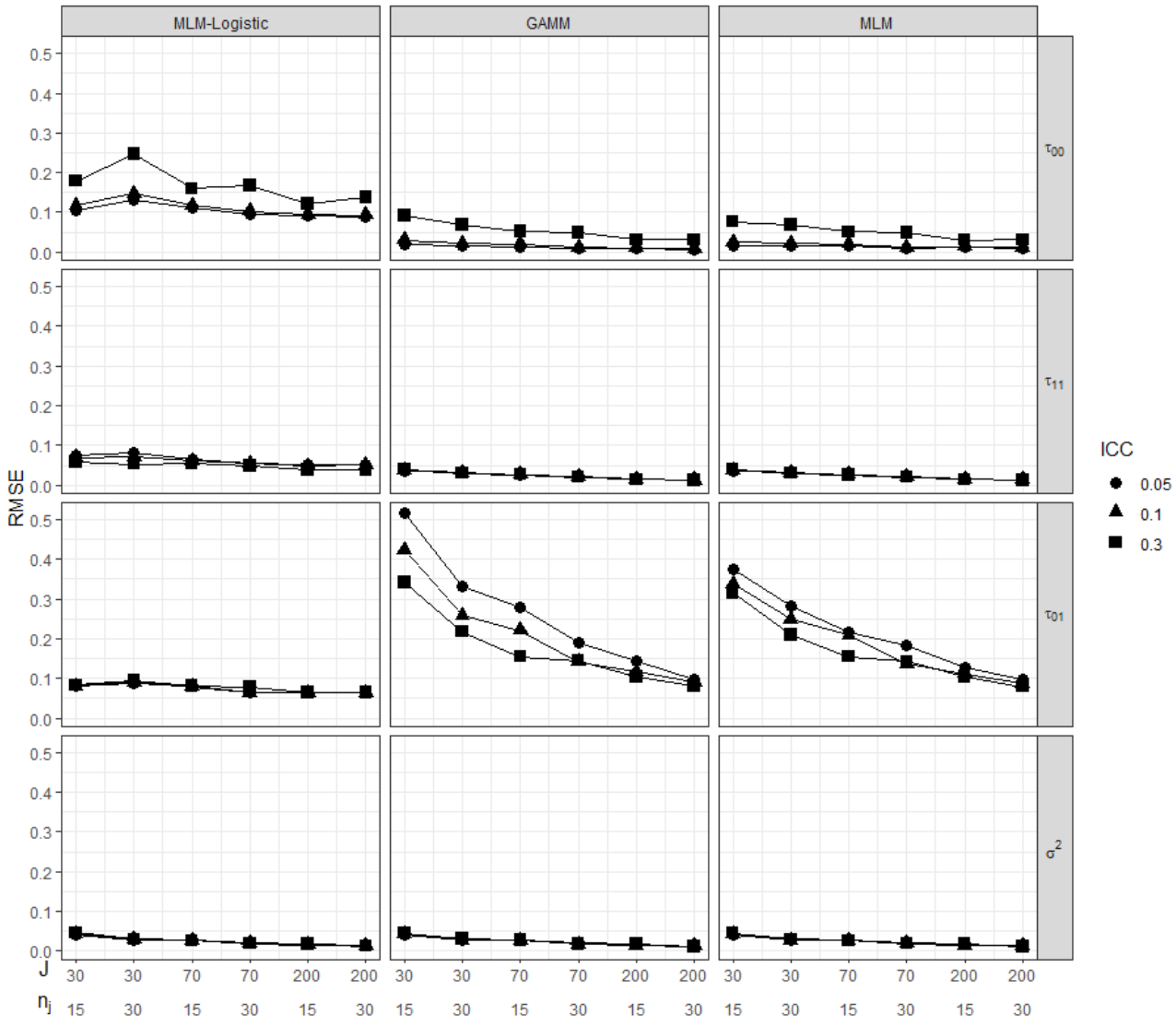


Figure S6.3 RMSE of parameter estimates in MLM-Logistic, GAMM, and MLM.

GAMM as a Data-Generating Model

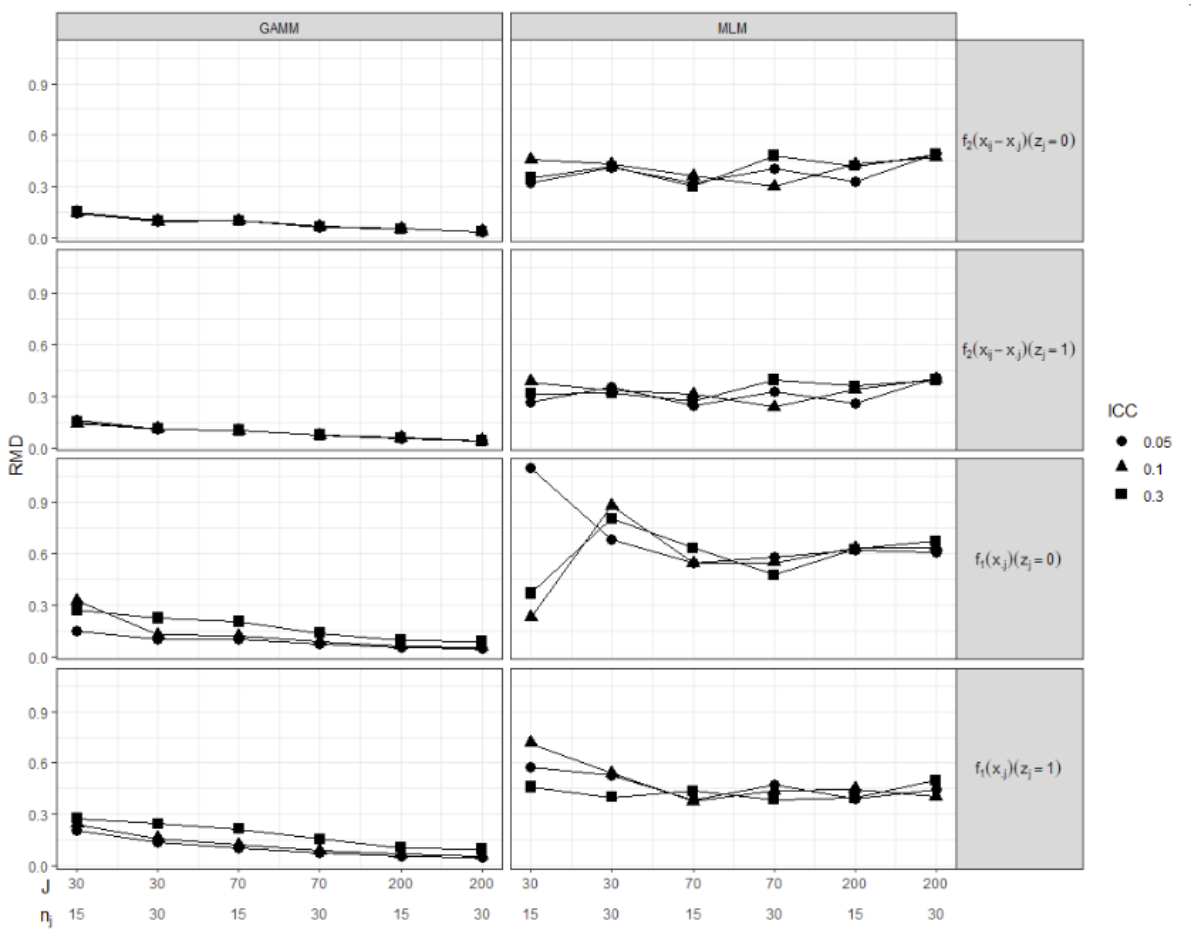


Figure S6.4 RMD of Smooth Functions in GAMM and MLM.

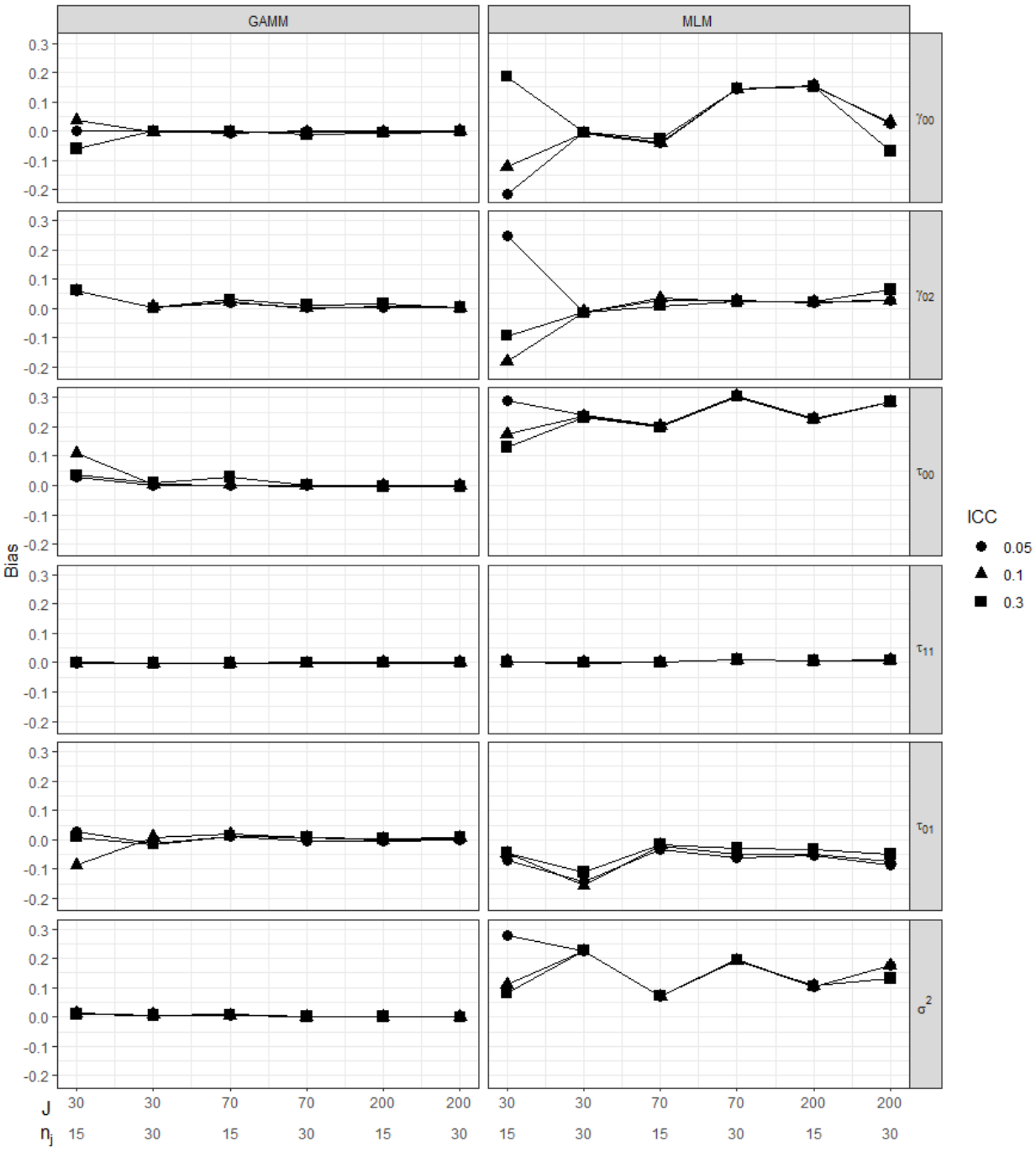


Figure S6.5 Bias of parameter estimates in GAMM and MLM.

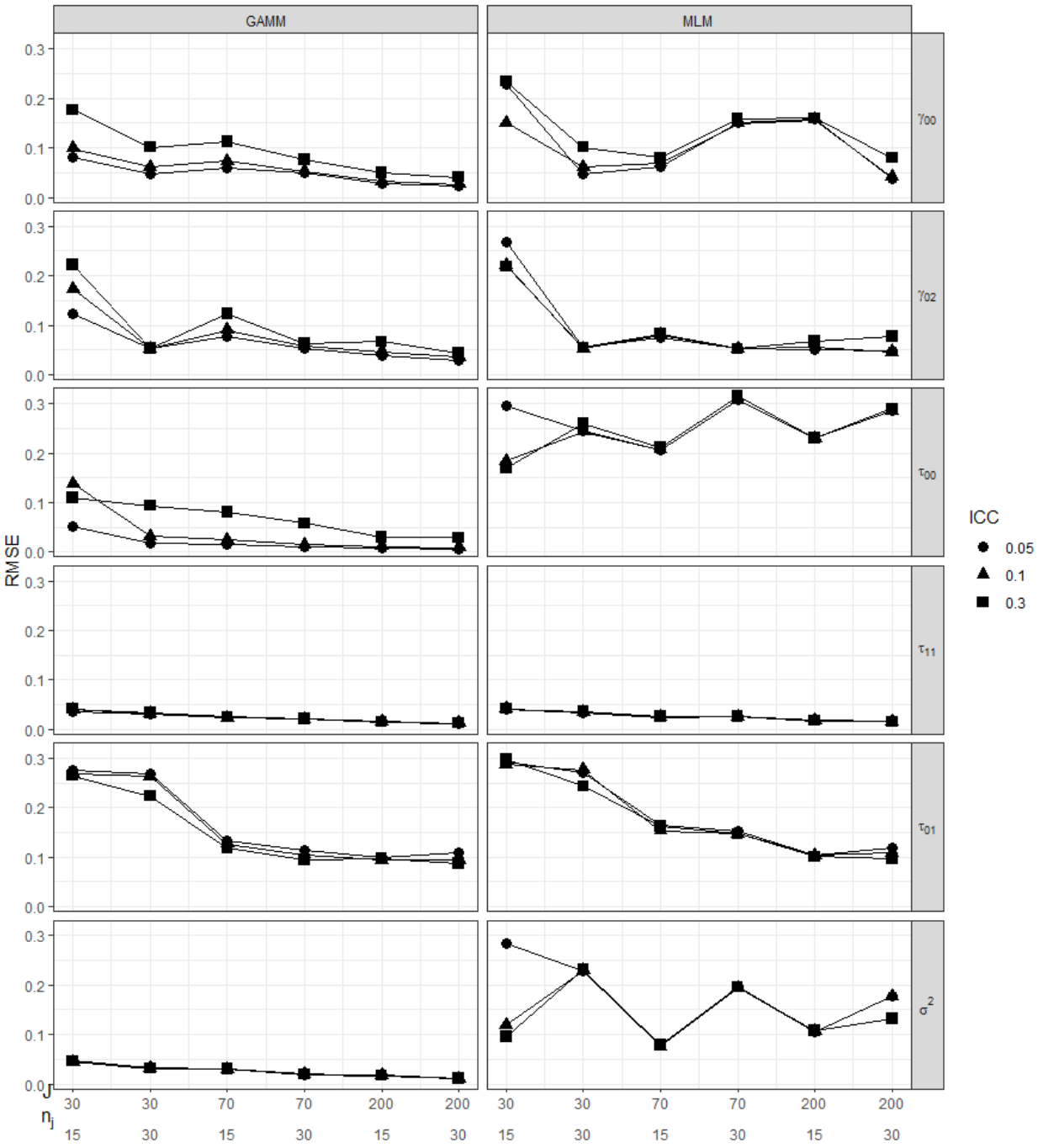


Figure S6.6 RMSE of parameter estimates in GAMM and MLM.

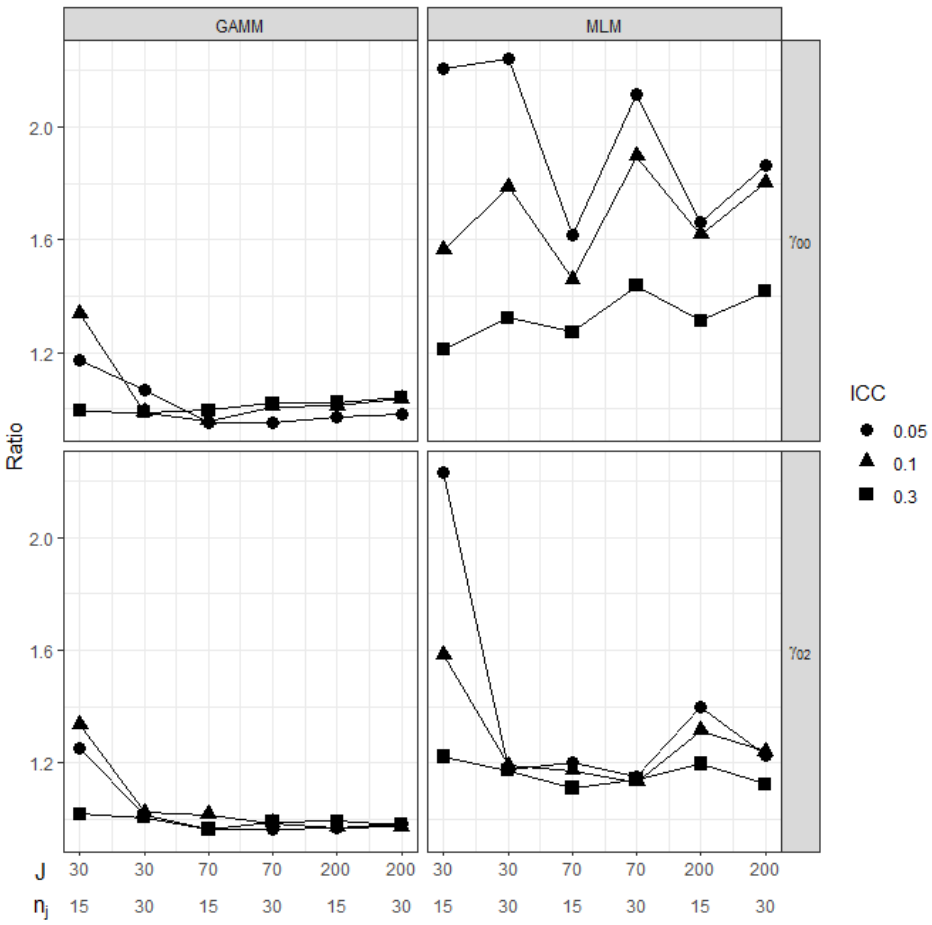


Figure S6.7 Ratio of M(SE) to SD for estimates of fixed effects in GAMM and MLM.

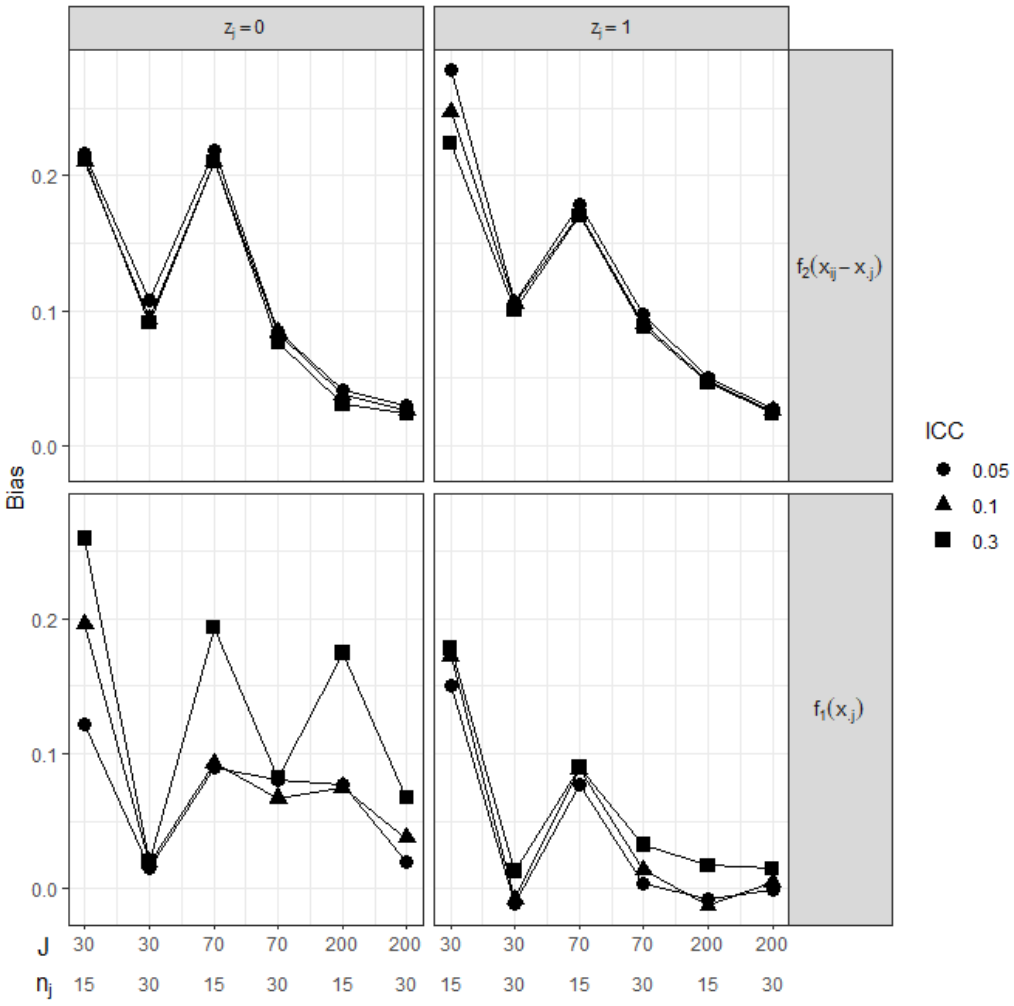


Figure S6.8 Average bias for estimates of basis coefficients in GAMM.

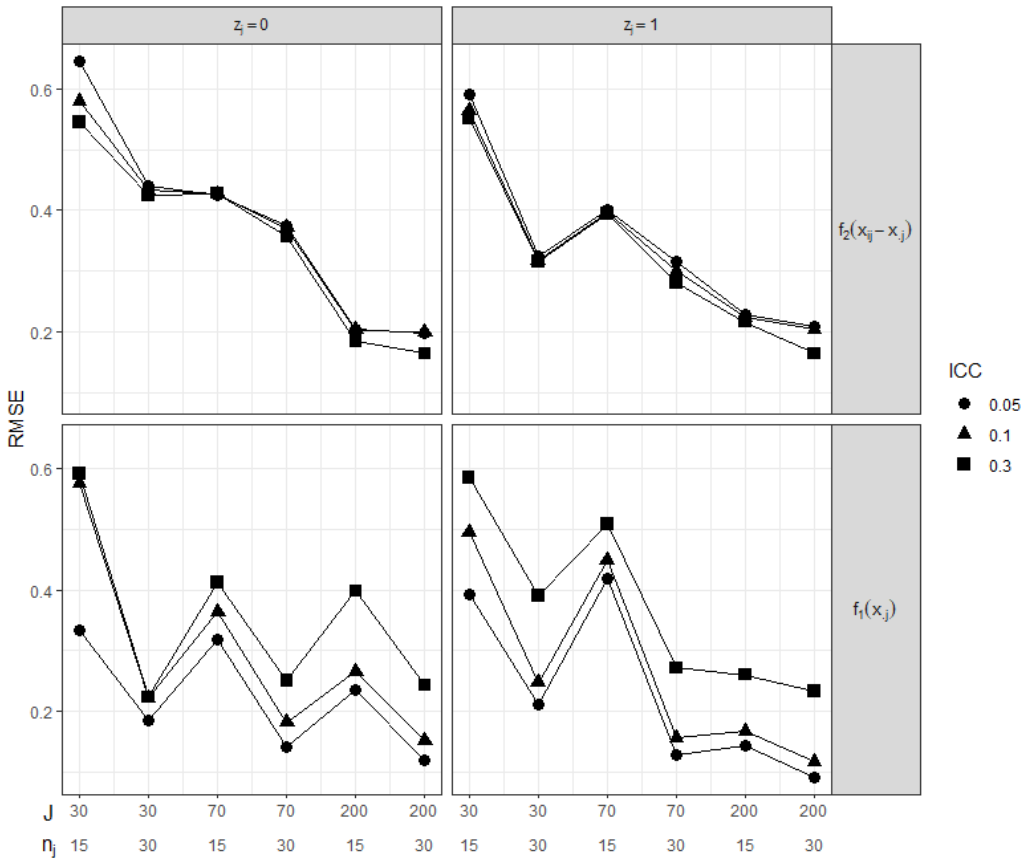


Figure S6.9 Average RMSE for estimates of basis coefficients in GAMM.

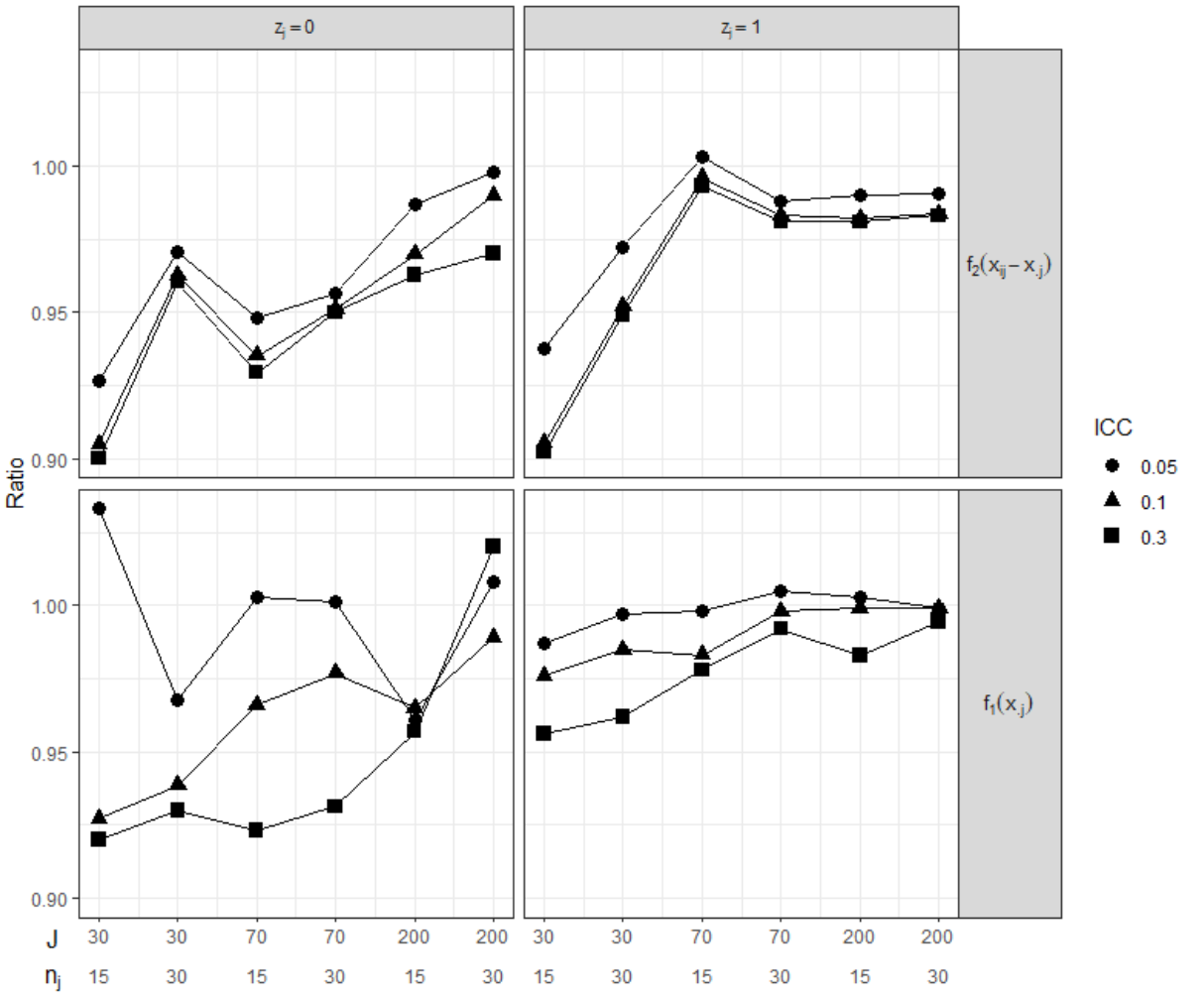


Figure S6.10 Average ratio of M(SE) to SD (bottom) for estimates of basis coefficients in GAMM.