Journal of Statistics and Computer Science
Vol. 4, Issue 2, 2025, pp. 69-80
https://doi.org/10.47509/JSCS.2025.v04i02.01

Extension of Linear Mixed Models to Modern Predictive Methods: Mixed-Effects Ridge Regression and Regression Trees

Yahia S. El-Horbaty[1,2], Meirah R.O. Alhefeiti[1], Ali S. Safi[1], Raghad Badawi[1], Shahd B. Estaitia[1], Leen M. Alzaghal[1]

[1] Department of Statistics and Business Analytics. College of Business and Economics, United Arab Emirates University, UAE
[2] Department of Mathematics, Insurance, and Applied Statistics. Faculty of Commerce and Business Administration, Helwan University, Egypt

Abstract
In modern data analysis, classical statistical methods and machine learning tools are increasingly integrated to produce improved predictive models. This article provides an overview of linear mixed models (LMMs), including their notation, assumptions, and restricted maximum likelihood estimation. We further review advanced extensions—such as mixed-effects ridge regression and mixed-effects regression trees (MERT) for clustered data, emphasizing their notation and current applications. A real-data application assesses and compares the predictive capabilities of the methods under consideration.

Keywords. Machine Learning, Variance Components, Random Effects, Mean Squared Error
Mathematics Subject Classification: 62J07, 62J05.

1. Introduction

Linear models for cluster correlated data, also known as linear mixed models (LMMs), are indispensable tools in modern statistical analysis for addressing data structures characterized by non-independence, hierarchical clustering, longitudinal measurements, and multilevel designs (Goldstein, 2011; Raudenbush and Bryk, 2002; Jiang, 2017; Fitzmaurice et al., 2012). The versatility in LMMs allows integration of both fixed and random effects, establishing nuanced modeling of complex variance sources within a diversity of disciplines such as medicine, psychology, ecology, and economics (West et al., 2022; Pinheiro and Bates, 2000).

The growing availability of data with nested/grouped observations has spurred advances in LMMs theory and computation. Estimation techniques such as restricted maximum likelihood (REML) and extensions including mixed-effects ridge regression and tree-based models are now receiving increasing attention in various theoretical and applied studies. This article systematically presents a basic review of some foundational and advanced topics in LMMs. Those topics have been scattered with no much focus on highlighting their potential combined presence in various studies in the literature. We basically focus on the problem of nonlinear fixed effects and the multicollinearity (rank deficiency) problem in the design matrix for the fixed effects.

[1] Correspondent Author: yahia_mohamed@uaeu.ac.ae and yahia_mohamed@commerce.helwan.edu.eg

Article History
Received : 11 July 2025; Revised : 13 August 2025; Accepted : 19 August 2025; Published : 30 December 2025

To cite this paper
Yahia S. El-Horbaty, Meirah R.O. Alhefeiti & et al. (2025). Extension of Linear Mixed Models to Modern Predictive Methods: Mixed-Effects Ridge Regression and Regression Trees. *Journal of Statistics and Computer Science*. 4(2), 69-80.

Various studies have addressed each problem separately. For the multicollinearity problem under LMMs, Ozkal and Can (2017) and earlier, Elliot et al. (2011) have considered the issue. In parallel with these developments, other studies have focused on quantifying multicollinearity severity (Yu et al., 2015). These studies distinguish two scenarios. In the first, variance components are fixed at their estimated values while estimating the ridge parameters (i.e., fixed effects and shrinkage terms). In the second, variance components are simultaneously estimated with the ridge parameters. It is noted, however, that the effect of variance components estimation on the accuracy of ridge estimates has not yet been assessed using empirical studies.

Recently, attention has turned toward assessing the predictive capabilities of machine learning tools as alternatives to the stringent assumptions of classical likelihood estimation theory—particularly when the functional forms of covariates in working LMMs are unknown (i.e., potential nonlinearity). As we focus on recently developed tree-based machine learning methods in a regression context, we refer to the studies by Hajjem et al. (2011) and Hu et al. (2023). In these studies, assumptions such as the linear fixed-effects term (also known as linear population-average terms) are relaxed. Further, the prediction capabilities of tree-based regression models may also relax the necessity of correctly specifying the distribution of the random effects under the working model.

We begin our review with an introduction to the notation and assumptions underpinning LMM formulation, emphasizing matrix notation for clarity and generalizability. Subsequently, we summarize estimation procedures via REML to frame how parameters—including fixed effects and variance components—are obtained in practice. Building on this framework, we review recently proposed ridge regression adaptations for LMMs that address multicollinearity and improve estimator stability, referencing Ozkale and Can (2017), who applied their proposal to kidney failure data. Finally, we discuss MERT, proposed by Hajjem et al. (2011), a recent predictive methodology that integrates tree-based partitioning with random effects modeling, highlighting its utility for hierarchical or clustered data where nonlinearity and interactions are prevalent in the systematic component of LMMs.

The rest of this article is organized as follows: Section 2 presents LMMs; Section 3 reviews the necessary steps to obtain ridge regression estimators/predictors; Section 4 presents the modern integration of LMMs and regression trees (i.e., MERT); and Section 5 provides a real-data application followed by directions for future research.

2. Linear Mixed Models

Linear mixed models extend the classical linear regression framework by incorporating random effects to capture variability attributable to hierarchical or clustered data structures. Formally, the LMM for an $N \times 1$ observation vector y is expressed as

$$y = X\beta + Zu + \varepsilon,$$

where y denotes an $N \times 1$ observation vector, X is the $N \times p$ design matrix for fixed effects, β is the $p \times 1$ vector of fixed-effect coefficients, Z is the $N \times q$ design matrix for random effects, $u \sim N(0, G)$ is the $q \times 1$ vector of random effects, $\varepsilon \sim N(0, R)$ is the $N \times 1$ vector of residual errors. The random effects u and residuals ε are mutually independent. The covariance structure of the response vector y is thus

$$\mathrm{Var}(y) = V = ZGZ^T + R.$$

Common assumptions include normality of u and ε, zero means, and known covariance structures G and R up to parameters to be estimated. The matrices G and R are positive semi-definite and typically parameterized by variance components reflecting between-group and within-group variability, respectively.

This model accommodates both balanced and unbalanced designs and can represent nested, crossed, or partially crossed random effects by appropriate construction of Z and G. It unifies regression and analysis-of-variance concepts, offering interpretable fixed effects under mixed sources of variability. REML estimation is a widely adopted approach for fitting LMMs, offering less biased estimates of variance components compared to standard maximum likelihood (ML) methods. The central advantage of REML lies in its accounting for the loss of degrees of freedom incurred by estimating fixed effect parameters, which leads to more accurate inference on random effects covariance structures.

Let θ denote the vector of variance components parameterizing the covariance matrices $G(\theta)$ and $R(\theta)$, and β be the fixed effects parameter vector. The marginal distribution of y is

$$y \sim N(X\beta, V(\theta)),$$

with $V(\theta) = ZG(\theta)Z^{\top} + R(\theta)$.

REML maximizes the likelihood of a set of linear combinations of y that eliminate the fixed effects β, effectively basing inference solely on the variance parameters. Formally, the restricted log-likelihood can be expressed as

$$\ell(\theta) = -\frac{1}{2}\left(n\log|V(\theta)| + \log|X^{\top}V(\theta)^{-1}X| + (y - X\hat{\beta})^{\top}V(\theta)^{-1}(y - X\hat{\beta})\right) + \text{const.}$$

where

$$\hat{\beta} = (X^{\top}V(\theta)^{-1}X)^{-1}X^{\top}V(\theta)^{-1}y.$$

denotes the generalized least squares estimator of β given the variance components. The key estimation steps for REML are

1. For a fixed θ, estimate fixed effects via $\hat{\beta} = (X^{\top}V(\theta)^{-1}X)^{-1}X^{\top}V(\theta)^{-1}y$
2. Maximize the restricted log-likelihood $\ell_R(\theta)$ with respect to the variance components θ, often achieved using numerical optimization routines.
3. Substitute the estimated variance components $\hat{\theta}$ back to obtain the best linear unbiased predictors (BLUPs) of random effects:

$$\hat{u} = G(\hat{\theta})Z^{\top}V(\hat{\theta})^{-1}(y - X\hat{\beta}).$$

These procedures yield efficient and nearly unbiased estimates for both fixed and random effects in the LMM framework. REML accounts for the degrees of freedom lost due to fixed-effects estimation and is thus preferable, especially in small or moderate sample sizes with complex random-effects structures. The matrix formulation elegantly captures the entire estimation process, making it suitable for computational implementation and theoretical investigations.

3. Mixed-Effects Ridge Regression

Addressing issues of multicollinearity and enhancing estimator stability in linear mixed models, ridge regression has been adapted within the mixed-effects modeling framework to provide penalized estimation solutions. Classic ridge regression adds a penalty proportional to the squared magnitude of coefficients to shrink estimates towards zero, thus reducing variance at the cost of introducing some bias (Hoerl and Kennard, 1970). This principle is extended in mixed-effects settings to regularize fixed and random effects simultaneously, mitigating overfitting and improving predictive accuracy especially in small sample or high-dimensional contexts.

Following Ozkale and Can (2017), consider the LMM

$$y = X\beta + Zu + \varepsilon$$

with the usual assumptions on $u \sim N(0, G)$ and $\varepsilon \sim N(0, R)$. The mixed-effects ridge estimator of fixed effects β is formulated by penalizing the weighted least squares objective function augmented by a ridge penalty term:

$$\hat{\beta}_{\text{ridge}}(\lambda) = \arg\min[(y - X\beta)^\top V^{-1}(y - X\beta) + \lambda\beta^\top\beta],$$

where $V = ZGZ^T + R$ is the covariance of y and $\lambda \geq 0$ is the ridge biasing parameter controlling the degree of shrinkage. Solving the optimization yields the ridge penalized estimator:

$$\hat{\beta}_{\text{ridge}} = \left(X^\top V^{-1} X + \lambda I_p\right)^{-1} X^\top V^{-1} y,$$

with I_p denoting a $p \times p$ identity matrix. The addition of λI_p stabilizes the inversion in cases of multicollinearity or near-singularity in the design matrix X. The random effects estimates are adapted accordingly, obtained as best linear unbiased predictors (BLUPs) using variance components and penalized fixed effect estimates:

$$\hat{u} = GZ^\top V^{-1}(y - X\hat{\beta}_{\text{ridge}}).$$

Selection of the optimal λ value can be performed by cross-validation, generalized cross validation, or information criteria such as AIC or BIC adapted to penalized contexts. This tuning governs the bias-variance trade-off in fixed-effects estimation. The approach retains the interpretability and flexible modeling advantages of LMMs while guarding against estimation instability due to correlated predictors or limited sample sizes—making it highly suitable for complex biomedical and longitudinal data analyses. Two biasing parameter selection methods are suggested below.

Harmonic Mean Estimator ($\hat{k}_h$): The authors extend the approach of Hoerl and Kennard (1970) to the linear mixed model. After transforming the model to a canonical form, the SMSE is minimized by choosing penalty components $k_i = 1/\gamma_i^2$, where γ_i are components of β in transformed space. Since β is unknown, an estimator substitutes β with $\tilde{\beta}$ (ridge or unbiased estimator), giving: $\hat{k}_h = \frac{p}{\tilde{\beta}^\top \tilde{\beta}}$, where p is the number of fixed effects. This estimator provides a biasing parameter balancing shrinkage and bias to minimize prediction error.

Generalized Cross Validation (GCV) Method (k_{GCV}): The ridge estimator produces fitted values $\tilde{y}$ expressible as $\tilde{y} = Sy$, where S is a smoothing matrix depending on k. The GCV criterion is defined as: $k_{GCV} = \frac{\frac{1}{n}\|y-\tilde{y}\|^2}{\left(1-\frac{\mathrm{trace}(S)}{n}\right)^2}$

where n is the sample size and k_{GCV} denotes the value minimizing the GCV criterion, which is obtained numerically as no closed-form solution exists.

4. Mixed-Effects Regression Tree

Regression tree fitting under the Classification and Regression Trees (CART) framework is a non-parametric method for predicting a continuous outcome variable by partitioning the predictor space into distinct, non-overlapping regions, each associated with a predicted value. The fitting process begins with the entire data as the root node and recursively splits it into two subsets based on the values of predictor variables. Splits are chosen to minimize the residual sum of squares (RSS) within the child nodes. Formally, for splitting based on the variable x_j at split point s, the goal is to find

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right],$$

where $R_1(j,s) = \{x \mid x_j \leq s\}$ and $R_2(j,s) = \{x \mid x_j > s\}$ are the resulting regions after the split, and the constants c_1 and c_2 are the mean response values in each region. This recursive binary partitioning proceeds until a stopping criterion is met, typically a minimum number of observations in a node or a maximum tree depth, after which the average response in each terminal node becomes the predicted value.

The key mathematical concept behind CART is thus the greedy search for splits that yield maximal reduction in node impurity measured by the residual sum of squares. Splitting is done per variable and potential split point to find the optimal (j,s) that minimizes the total RSS after the split. Tree complexity control is achieved by pruning, which balances fit and model simplicity by minimizing a cost complexity criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|,$$

where $|T|$ is the number of terminal nodes, R_m are the regions defined by the tree T, $\hat{y}_{R_m}$ is the mean response in region R_m, and $\alpha \geq 0$ is a complexity parameter controlling pruning strength. This criterion helps prevent overfitting by penalizing excessively large trees, leading to a more generalizable regression tree model. Thus, CART combines recursive binary splitting to minimize within-node variance and pruning to optimize bias-variance tradeoff in the final fitted tree.

This algorithmic approach enables flexible nonlinear regression and interaction detection without requiring parametric assumptions about the data distribution or functional form. The mathematical formulation centers on partitioning the predictor space via greedy minimization of residual sums of squares, complemented by cost-complexity pruning to determine optimal model size.

The emergence of MERT represents a significant advancement in modeling clustered or hierarchical data with complex nonlinearities and interactions that traditional linear mixed models cannot capture effectively. Hajjem et al. (2011) introduced this methodology, extending standard regression trees to clustered continuous response data by integrating random-effects components within the tree structure to account for intra-cluster correlation.

The MERT model preserves the essential structure of a linear mixed model for cluster i with observations $y_i = (y_{i1}, \ldots, y_{im})^\top$:

$$y_i = \mu(X_i) + Z_i u_i + \varepsilon_i,$$

where $\mu(X_i)$ represents mean responses predicted by the regression tree function of covariates X_i, Z_i is random-effects design matrix for cluster i, $u_i \sim N(0, D)$ are cluster-specific random effects capturing correlated variability, and $\varepsilon_i \sim N(0, \sigma^2 I_{m_i})$ are residual errors. Unlike traditional LMMs, which model fixed effects linearly, MERT estimates the fixed part via adaptive tree partitioning, accounting for predictor interactions and nonlinearities—while simultaneously accommodating within-cluster correlation.

The estimation algorithm for MERT involves iteratively alternating between fitting the tree structure to residuals adjusted for random effects and updating random effect predictions given the current tree. This is akin to an Expectation-Maximization (EM) approach:

Table 1: Estimation Algorithm for MERT (Hajjem et al., 2014)

Step	Description
1	Initialize iteration $r = 0$, variance components $D^{(0)}$, residual variance $\sigma^{2(0)}$.
2	Fit regression tree to pseudo-responses $y_i - Z_i u_i^{(r)}$
3	Update random effects: $u_i^{(r+1)} = D^{(r)} Z_i^{\top} V_i^{(r)-1}(y_i - \hat{\mu}(X_i))$.
4	Re-estimate variance components $D^{(r+1)}$, $\sigma^{2(r+1)}$ from residuals.
5	Check convergence; if not converged, increment r and repeat from Step 2.

5. Application

The Wages dataset is a retrieved from the National Longitudinal Survey of Youth (NLSY) represents an unbalanced longitudinal dataset of individual workers' wages observed repeatedly over time. The *log-wage* serves as the response variable, while *exper* (duration of work experience), race (individual races: White, Black, and Hispanic), and *hgc* (highest grade completed) are covariates. Analysis of this dataset typically involves an unobserved individual random effect in additive form to the linear combination of explanatory variables (i.e., *exper*, *hgc*, and *race*), which represents the fixed-effects term of an LMM. The need for random effects has been tested using suitable tests, with the null hypothesis of no random effects rejected (El-Horbaty and Hanafy, 2020, 2024; El-Horbaty, 2022, 2024a, 2024b, 2025).

The suggested LMM for the Wages dataset is thus given by

$$log\_wage_{ij} = \beta_0 + \beta_1 exper_{ij} + \beta_2 hgc_{ij} + \beta_3 race_{ij} + u_i + \varepsilon_{ij},$$

where β_s denotes the model fixed effects for $s = 0,1,2,3$, u_i denotes the unobserved random effect of the i^{th} individual, and ε_{ij} denotes the residual errors. The subscripts j and i stand, respectively, for the j^{th} observation from the i^{th} individual where $i = 1, \ldots, 729$ and j ranges between 5 and 13. The overall sample size is $n = 6005$ observations.

The mixed effect regression-tree model is given by

$$log\_wage_{ij} = \mu(exper_{ij}, hgc_{ij}, race_{ij}) + u_i + \varepsilon_{ii},$$

We use 10-fold cross validation to assess the prediction performance between the statistical methods under review. The results reported the average root mean squared error (aRMSE). The outcomes of our assessments are reported in Table 2. To obtain the RMSE we used the formula

$$RMSE = \sqrt{\frac{\sum\left(y_{tst} - \hat{y}_{pred}\right)^2}{n_*}}$$

where y_{tst} denotes the response value from the testing set, $\hat{y}_{pred}$ denotes the predicted value of the response variable based on the statistical method using the training dataset, and $n_* = 0.3n$. Then, the average of the ten folds is calculate yielding the aRMSE for each method.

Table 2. *aRMSE* values for different methods on the Wages dataset via 10-fold cross validation

Method	LMM	Mixed Effects Ridge	Mixed Effects Regression Tree
aRMSE value	4.1325	4.1326	3.8349

The results from Table 2 indicate that the MERT method is superior, exhibiting better predictive performance (lower *aRMSE*) than the other methods. It is also noted that, methods based on linear models (LMM and mixed-effects ridge) perform nearly identically in terms of *aRMSE*. This yields two key insights: first, the superior prediction by the mixed-effects regression tree justifies a potentially nonlinear link function connecting the response (log-wage) to the covariates; second, multicollinearity effects are negligible, resulting in nearly identical *aRMSE* values for both LMM and mixed-effects ridge regression.

6. Conclusion

In this study, attention has been given to two modern predictive extensions of LMMs, highlighting their differences and success in capturing certain violations of standard LMM assumptions by comparing their predictive capabilities on a real dataset. The application reveals that nonlinearity is the main reason for the lower *aRMSE* achieved by the mixed-effects regression tree. For end users, an R code is provided in the Appendix for guidance. However, multicollinearity can also have serious effects, though it was not evident in this real-data application. For future work, extending assessments of predictive efficiency in the presence of multicollinearity under generalized LMMs presents an attractive research problem. Furthermore, extending MERT models to accommodate autoregressive errors offers another promising research area with substantial relevance to economic applications.

Acknowledgment

The authors are grateful to the reviewers and editors, whose comments improved the final version of this manuscript. This research is sponsored by the UAEU Summer Undergraduate Research Experience (SURE PLUS) Grant Application 2025 (Grant Code: G00005171).

References

El-Horbaty, Y. S. (2022). A note on covariance decomposition in linear models with nested-error structure: new and alternative derivations of the F-test. Journal of Statistical Theory and Practice, 16(4), 69.

El-Horbaty, Y. S. (2024a). A Monte Carlo permutation procedure for testing variance components in generalized linear regression models. *Computational Statistics*, 39(5), 2605-2621.

El-Horbaty, Y. S. (2024b). A new matrix-based formulation for computing the variance components F-test in linear models with crossed random effects. Quality & Quantity, 58(3), 3001-3020.

El-Horbaty, Y. S. (2025). Testing subsets of crossed random effects in generalized linear mixed models: a distribution-free approach. Quality & Quantity, 1-19.

El-Horbaty, Y. S., Hanafy, E. M. (2020). On testing variance components using permutation tests. *Journal of Applied Probability and Statistics*, 15(1), 111-124.

El-Horbaty, Y. S., Hanafy, E. M. (2024). A Monte Carlo permutation procedure for testing variance components using robust estimation methods. *Statistical Papers*, 65(1), 335-356.

Eliot, M., Ferguson, J., Reilly, M. P., & Foulkes, A. S. (2011). Ridge regression for longitudinal biomarker data. *The International Journal of Biostatistics*, 7(1). doi. 0000102202155746791353.

Fitzmaurice, G. M., Laird, N. M., & Ware, J. H. (2012). Applied longitudinal analysis. John Wiley & Sons.

Goldstein, H. Multilevel statistical models. John Wiley & Sons. (2011).

Hajjem, A., Bellavance, F., & Larocque, D. (2011). Mixed effects regression trees for clustered data. *Statistics & probability letters*, 81(4), 451-459.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55-67.

Hu, S., Wang, Y. G., Drovandi, C., & Cao, T. (2023). Predictions of machine learning with mixed-effects in analyzing longitudinal data under model misspecification. *Statistical Methods & Applications*, 32(2), 681-711.

Jiang, J. (2017). Asymptotic analysis of mixed effects models: theory, applications, and open problems. Chapman and Hall/CRC.

Özkale, M. R., & Can, F. (2017). An evaluation of ridge estimator in linear mixed models: an example from kidney failure data. *Journal of Applied Statistics*, 44(12), 2251-2269.

Pinheiro, J. C., & Bates, D. M. (2000). Mixed-effects models in S and S-PLUS. New York, NY: Springer New York.

Raudenbush, S. W., & Bryk, A. S. (2002). Hierarchical linear models: Applications and data analysis methods (Vol. 1). sage.

West, B. T., Welch, K. B., & Galecki, A. T. (2022). Linear mixed models: a practical guide using statistical software. Chapman and Hall/CRC.

Yu, H., Jiang, S., and Land, K.C. (2015). Multicollinearity in Hierarchical Linear Models. Social Science Research. 53: 118-136. doi.org/10.1016/j.ssresearch.2015.04.008.

Appendix. R code

```
# Wages Dataset Example:
# =====================
# --- Read Data ----
Wages     =     as.data.frame(read.csv(file     =     "D:/Y-Research     UAEU/(2)     SURE+2025/(4)     RealData
Application/v1.WAGES.SURE+.csv",header=T))
Y = Wages$resp ; x1 = Wages$exper ; x2 = Wages$hgc ; x3 = as.factor(Wages$race) ; id = Wages$Gr.ID
m = 729 # this dataset has 729 clusters (so m = 729)
p = 4 ; n = length(Y)
cluster_sizes <- table(id)  # Calculate cluster sizes for each unique cluster
ni <- as.numeric(cluster_sizes) # Convert cluster_sizes to numeric vector as needed

X = cbind(1,x1,x2,x3)
Zst = matrix(1,n,1) # stands for stacked vectors of ones corresponding to the random effects


Z = matrix(1,ni[1],1) ; for(i in 2:m) {Z = adiag(Z,matrix(1,ni[i],1))}

df.n = data.frame(id, Y, x1, x2, x3, Zst) %>%
  mutate(original_row = row_number())
```

```
# --- k-fold CV ---
k.fold = 10
sRMSE.lmm = 0 ; sRMSE.mRidge = 0 ;   sRMSE = 0  # necessary counters
#set.seed(123) # fixing all random selections

for (i in 1:k.fold) {  print (i) # Changed to 5 iterations for consistency

# --- Train-Test Split (80% train, 20% test within each cluster) ---
train_data <- df.n %>%
  group_by(id) %>%
  sample_frac(0.8) %>%
  ungroup()
test_data <- anti_join(df.n, train_data, by = setdiff(colnames(df.n), "original_row"))

train_data <- as.data.frame(train_data[order(train_data$original_row), ])
test_data <- test_data[order(test_data$original_row), ]
# --- Extract original row indices ---
train_indices <- train_data$original_row
test_indices <- test_data$original_row

# --- Apply the splits to X , Y , Zst ---
Z.tra <- Z[train_indices, ]
Z.tst <- Z[test_indices, ]
Zst.tra <- Zst[train_indices, ]
Zst.tst <- Zst[test_indices, ]
Y.tra <- Y[train_indices]
Y.tst <- Y[test_indices]
X.tra <- X[train_indices, ]
X.tst <- X[test_indices, ]

# --- apply lmm & Ridge prediction ---

# --- LMMs ---

# ----------   Start LMM model fit followed by extracting eFE & pRE -----

lmm.REML = lmer(Y ~ 1 + x1 + x2 + x3  + (1 | id), data = train_data, REML=TRUE ,
                control = lmerControl(optimizer = "bobyqa", restart_edge = TRUE, boundary.tol = 1e-2 , optCtrl =
list(maxfun = 5e5) )  )


    eFE.lmm             = as.vector(fixef(lmm.REML))             #extract estimated FEs from model "lmm.REML"
    pRE.lmm             = as.vector(t(ranef(lmm.REML)$id))             #To present the predicted random effects
coeffecient(b^)


    # --- Predict on Test Set ---
    X_design <- model.matrix(~ x1 + x2 + x3, data = test_data)
    #Y_pred.lmm <- (X_design %*% eFE.lmm) + (Z.tst %*% pRE.lmm)
    Y_pred.lmm <- (X_design %*% eFE.lmm)
    # --- Evaluate Prediction Performance ---
    RMSE.lmm <- sqrt(mean((Y.tst - Y_pred.lmm)^2))
    #cat("Prediction RMSE on Test Set:", round(RMSE, 3), "\n")
    sRMSE.lmm = sRMSE.lmm + RMSE.lmm

# ---   Start mixed-Ridge model fit followed by calculating beta.Ridge & u.Ridge  -----
    Iden=diag(1,p)
    k=p/sum(eFE.lmm^2)  #OR#  k=p/t(eFE.lmm)%*%eFE.lmm

    sigma2.epsilon.lmm  = (sigma(lmm.REML))^2        # Extract σ and find σ^2
    vc                 = as.data.frame(VarCorr(lmm.REML))  # VarCorr() provides the estimated variance components
"vc" & their St.Dev
    sigma2.ua.lmm        = vc$vcov[1]                  #estract estimated sigma2.ua
    eFE.lmm.CovMat      = vcov(lmm.REML)              #extract Var(β^)= σ2(X'V^-1X)^-1
    V.lmm                         =  (  sigma2.epsilon.lmm*diag(1,round(nrow(Z.tra)),round(nrow(Z.tra)))  )  +  (
sigma2.ua.lmm*Z.tra%*%t(Z.tra)  )  #V= (σ^2)*I + (σ^2)*ZZ'

    solve.V.lmm         = solve(V.lmm)
    Xt.invV.X            = (t(X.tra)%*%solve.V.lmm%*%X.tra)
    eFE.mRidge           = (solve(Xt.invV.X + k*Iden))%*% t(X.tra)%*%solve.V.lmm%*%Y.tra
    pRE.mRidge           = (sigma2.ua.lmm*t(Z.tra))%*%solve.V.lmm%*%(Y.tra-X.tra%*%eFE.mRidge)

    # --- Predict on Test Set ---
    #Y_pred.mRidge <- (X.tst %*% eFE.mRidge) + (Z.tst %*% pRE.mRidge)
    Y_pred.mRidge <- (X.tst %*% eFE.mRidge)
```

```
    # --- Evaluate Prediction Performance ---
    RMSE.mRidge <- sqrt(mean((Y.tst - Y_pred.mRidge)^2))
    #cat("Prediction RMSE on Test Set:", round(RMSE, 3), "\n")
    sRMSE.mRidge = sRMSE.mRidge + RMSE.mRidge


# --- here the beginning of EM-CART phase ---

    # --- Initial Random Forest on training data ---
    rf_model <- randomForest(Y ~  x1 + x2 + x3 , data = train_data, ntree = 500)
    Y_hat_rf <- predict(rf_model, newdata = train_data)
    res.tra <- Y.tra - Y_hat_rf

    # --- EM Algorithm ---
    Omega  <- diag(rep(1,1))                   # initial value for Omega matrix
    s2e <- 1                                     # initial value for residual-variance
    z.res  <- as.vector(res.tra)               # estimated residuals
    uid  <- unique(train_data$id)
    niter = 50

    for (k in 1:niter) { #print(k)

      iO.A    <- solve(Omega)                         # inerse of Omega matrix
      T.A     <- R.A <- C.A <- 0
      mu.A    <- u.A <- NULL

      for (i in uid ) {

        row.i    <- which(train_data$id==i)
        Xi.A     <- X.tra[row.i,]
        Ai.A     <- Zst.tra[row.i]
        AAi.A    <- t(Ai.A)%*%Ai.A
        zi.res   <- z.res[row.i]
        Gammai.A <- solve(AAi.A/s2e + iO.A)            # define Gamma fun.
        mui.A <- (Gammai.A%*%t(Ai.A)%*%zi.res)/s2e         # define Mu fun. where mui = E[eta_i | y_i ; theta]
        mu.A  <- c(mu.A, mui.A)
        u.A   <- c(u.A, Ai.A%*%mui.A)                      # ui = ( Ai %*% E[eta_i | y_i ; theta] )
        Si.A  <- Gammai.A + mui.A%*%t(mui.A)               # Si = E[eta_i%*%t(eta_i)  | y_i ; theta]
        R.A   <- R.A + Si.A                              # R : the final numerator of the Omega matrix
        T.A   <- T.A + sum(diag(Si.A%*%AAi.A))             # trace( t(Ai)%*%Ai %*% E[eta_i%*%t(eta_i)  | y_i ;
theta] )
        C.A   <- C.A + t(mui.A)%*%t(Ai.A)%*%zi.res
      }

      # Update RF

      res.tra_adj <- Y.tra - Z.tra %*% mu.A
      train_data$Y <- res.tra_adj
      rf_model <- randomForest(Y ~  x1 + x2 + x3 , data = train_data, ntree = 500)
      Y_hat_rf <- predict(rf_model, newdata = train_data)
      res.tra <- Y.tra - Y_hat_rf

      z.res <- as.vector(res.tra)
      S2e <- (sum(z.res^2) -2*C.A[1]  + T.A )/length(z.res)
      Omega <- as.matrix(R.A/m)

    } # ending EM algorithm

    # --- Predict on Test Set ---
    fixed_pred <- predict(rf_model, newdata = test_data)
    random_pred <- Z.tst %*% mu.A
    Y_pred <- fixed_pred + random_pred

    # --- Evaluate Prediction Performance ---
    RMSE <- sqrt(mean((Y.tst - Y_pred)^2))
    #cat("Prediction RMSE on Test Set:", round(RMSE, 3), "\n")
    sRMSE = sRMSE + RMSE

    # here EM-CART phase has ended

} # terminate the k-fold loop

aRMSE.lmm = sRMSE.lmm / k.fold

aRMSE.mRidge = sRMSE.mRidge / k.fold
```

```
aRMSE = sRMSE / k.fold

aRMSE.lmm
aRMSE.mRidge
aRMSE
```