Multilevel Models for Big Data

Approaches for handling very large datasets

Ömercan Mısırlıoğlu, Yordan Saputra

Department of Statistics
TU Dortmund University

Final Presentation TBA, January (?) 2026

Table of Contents

Introduction – Multilevel Models

What are multilevel models?

- Hierarchical Structure
- Residual Components
- Variance Partitioning

Why use multilevel models?

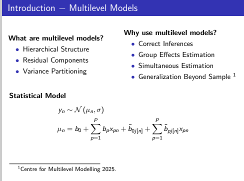
- Correct Inferences
- Group Effects Estimation
- Simultaneous Estimation
- Generalization Beyond Sample [1]

Statistical Model

$$y_n \sim \mathcal{N}(\mu_n, \sigma)$$

$$\mu_n = b_0 + \sum_{p=1}^{P} b_p x_{pn} + \tilde{b}_{0j[n]} + \sum_{p=1}^{P} \tilde{b}_{pj[n]} x_{pn}$$

[1] Centre for Multilevel Modelling 2025.

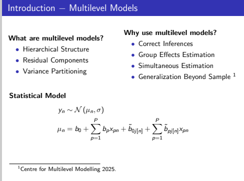


What are multilevel models?

- Hierarchical Structure: Observational data often feature individuals nested within higher-level groups, such as schools, workplaces, or geographical areas.
- Residual Components: Multilevel models account for these hierarchies by incorporating residual components at every level of the data structure.
- Variance Partitioning: These models divide residual variance into between-group and within-group components to capture unobserved factors influencing outcomes.

Why use multilevel models?

- Correct Inferences: Traditional methods assume independent observations, which are often false. Additionally, ignoring hierarchical structures can lead to underestimated standard errors and overstated statistical significance.
- Group Effects Estimation: Directly quantify between-group variation and identify outlying groups.
- Simultaneous Estimation: Unlike fixed effects models, the separation of observed and unobserved group characteristics is possible, allowing for simultaneous estimation of group-level and individual-level effects.
- Generalization Beyond Sample: Unlike fixed effects models which only describe sampled groups, multilevel models treat groups as random samples from a population, enabling generalizations to unobserved groups.



Statistical Model

$$y_n \sim \mathcal{N}(\mu_n, \sigma)$$
$$\mu_n = b_0 + b_1 x_{1n} + \ldots + b_p x_{pn} + \tilde{b}_{0j[n]} + \tilde{b}_{1j[n]} x_{1n} + \ldots + \tilde{b}_{pj[n]} x_{pn}$$

where:

- y_n: dependent variable for observation n
- x_{pn}: predictor variable p for observation n
- b_p: overall slope (or intercept for $p = 0$) for predictor p across all groups
- $\tilde{b}_{pj[n]}$: random effect of predictor p for group j that observation n belongs to
- σ: residual standard deviation (assumed constant across observations)

Introduction — Issues with Large Datasets

As hierarchical data scales (N groups $\times$ n individuals), massive datasets create the following issues:

- High number of groups
- Large group sizes[2]
- Design matrix construction[3]

[2] Clark 2019; Speelman, Heylen, and Geeraerts 2018.
[3] Wood, Goude, and Shaw 2015.

Introduction – Issues with Large Datasets

As hierarchical data scales (N groups $\times$ n individuals), massive datasets create the following issues:

- High number of groups
- Large group sizes[2]
- Design matrix construction[3]

[2] Clark 2019; Speelman, Heylen, and Geeraerts 2018.
[3] Wood, Goude, and Shaw 2015.

Issues with Large Datasets

- Large number of groups: Computational bottleneck from numerical integration over random effects for each group at every optimization step, leading to high computational costs
- Large group sizes: High-dimensional multivariate distributions create numerical issues with large covariance matrix inversion, even in linear models
- construction of the full design matrix X leads to high computational costs. In GAM, the estimator $\hat{\beta} = (X^T X + \sum \lambda_j S_j)^{-1} X^T y$ is hard to compute when X is large.
- it's even worse when we add another level to the hierarchy, e.g. students nested within classes nested within schools, which is common in educational research.

Table of Contents

The split-sample approach – Pseudo Likelihood

- Consider the log-likelihood function $\ell(\boldsymbol{\theta}) = \sum_i \ell(\boldsymbol{y_i}|\boldsymbol{\theta})$ where $\boldsymbol{y_i}$ is the vector of all observations in group i
- Replaces the log-likelihood contribution $\ell(\boldsymbol{y_i}|\boldsymbol{\theta})$ by a weighted sum of log-likelihood contributions for sub-vectors $\boldsymbol{Y_i}^{(s)}$
- More specifically, the pseudo-log-likelihood function:

$$p\ell(\psi) = \sum_i \sum_s \delta_s \, \ell(\boldsymbol{y_i}^{(s)}|\psi)$$

  is maximized instead with respect to ψ, which is not necessarily identical to $\boldsymbol{\theta}$
- Although $\hat{\psi}$ is not the MLE estimate, it still has similar properties such as consistency and asymptotic normality[4]

[4] Clark 2019.

The split-sample approach – Pseudo Likelihood

- Consider the log-likelihood function $\ell(\boldsymbol{\theta}) = \sum_i \ell(\boldsymbol{y_i}|\boldsymbol{\theta})$ where $\boldsymbol{y_i}$ is the vector of all observations in group i
- Replaces the log-likelihood contribution $\ell(\boldsymbol{y_i}|\boldsymbol{\theta})$ by a weighted sum of log-likelihood contributions for sub-vectors $\boldsymbol{Y_i}^{(s)}$
- More specifically, the pseudo-log-likelihood function:

$$p\ell(\psi) = \sum_i \sum_s \delta_s \, \ell(\boldsymbol{y_i}^{(s)}|\psi)$$

  is maximized instead with respect to ψ, which is not necessarily identical to $\boldsymbol{\theta}$
- Although $\hat{\psi}$ is not the MLE estimate, it still has similar properties such as consistency and asymptotic normality[4]

[4] Clark 2019.

- Now, how do we split $\boldsymbol{y_i}$ into sub-vectors $\boldsymbol{Y_i}^{(s)}$? There are different ways to do this, and we will discuss some of them in the next slides.
- Going back to the slide on "issues with large datasets", it's obvious that we can split the data in two (technically three) different ways: either we can split the data by groups, or we can split the data by observations within groups. The first one is more suitable when we have a large number of groups, while the second one is more suitable when we have a large number of observations within groups.

The split-sample approach — Graphical representation

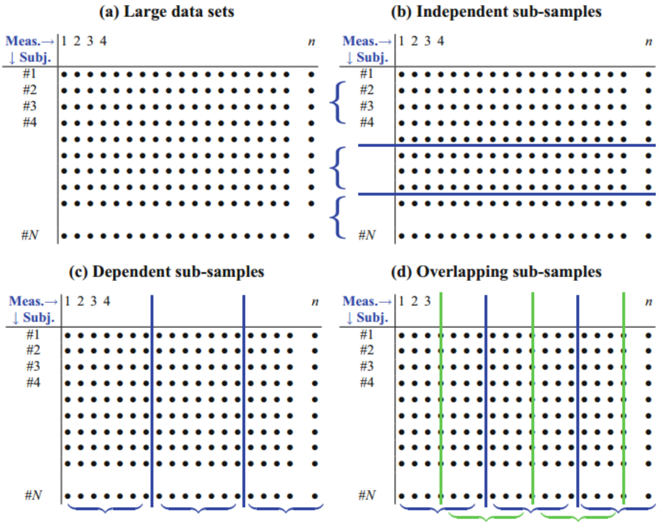


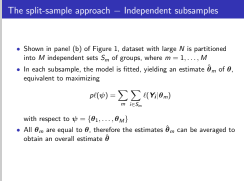
Figure 1: Graphical representation of different ways to split large samples

The split-sample approach – Independent subsamples

- Shown in panel (b) of Figure 1, dataset with large N is partitioned into M independent sets S_m of groups, where $m = 1, \dots, M$
- In each subsample, the model is fitted, yielding an estimate $\hat{\boldsymbol{\theta}}_m$ of $\boldsymbol{\theta}$, equivalent to maximizing

$$p\ell(\boldsymbol{\psi}) = \sum_{m} \sum_{i \in S_m} \ell(\boldsymbol{Y_i} | \boldsymbol{\theta}_m)$$

  with respect to $\boldsymbol{\psi} = \{\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_M\}$
- All $\boldsymbol{\theta}_m$ are equal to $\boldsymbol{\theta}$, therefore the estimates $\hat{\boldsymbol{\theta}}_m$ can be averaged to obtain an overall estimate $\hat{\boldsymbol{\theta}}$



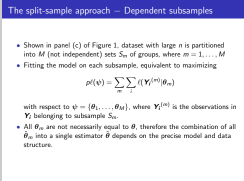
- $\boldsymbol{\theta}_m$ are all equal to $\boldsymbol{\theta}$ because the subsamples are independent
- Mention parallelization here, since we can fit the model on each subsample in parallel, which can significantly reduce the computational time.

The split-sample approach – Dependent subsamples

- Shown in panel (c) of Figure 1, dataset with large n is partitioned into M (not independent) sets S_m of groups, where $m = 1, \ldots, M$
- Fitting the model on each subsample, equivalent to maximizing

$$p\ell(\psi) = \sum_m \sum_i \ell(\boldsymbol{Y_i}^{(m)} | \boldsymbol{\theta}_m)$$

  with respect to $\psi = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M\}$, where $\boldsymbol{Y_i}^{(m)}$ is the observations in $\boldsymbol{Y_i}$ belonging to subsample S_m.
- All $\boldsymbol{\theta}_m$ are not necessarily equal to $\boldsymbol{\theta}$, therefore the combination of all $\hat{\boldsymbol{\theta}}_m$ into a single estimator $\hat{\boldsymbol{\theta}}$ depends on the precise model and data structure.



- $\boldsymbol{\theta}_m$ are not necessarily equal to $\boldsymbol{\theta}$ because the subsamples are not independent, and there may be some correlation between the observations in different subsamples.
- (GPT warning, dont trust 100%) The combination of all $\hat{\boldsymbol{\theta}}_m$ into a single estimator $\hat{\boldsymbol{\theta}}$ can be done using various methods, such as meta-analysis techniques, or by fitting a model to the estimates $\hat{\boldsymbol{\theta}}_m$ themselves.

The split-sample approach — Overlapping subsamples

- Shown in panel (d) of Figure 1, longitudinal dataset with very large n is partitioned similarly to dependent subsamples, but association between longitudinal observations is accounted for by letting the subsamples overlap
- Denoting the parameters in pair $\{\boldsymbol{Y_i}^{(p)}, \boldsymbol{Y_i}^{(q)}\}$ by $\boldsymbol{\theta}_{p,q}$, fitting the models on all pairs is equivalent to maximizing

$$p\ell(\psi) = \sum_{p<q} \sum_{i} \ell(\boldsymbol{Y_i}^{(p)}, \boldsymbol{Y_i}^{(q)} | \boldsymbol{\theta}_{p,q})$$

  with respect to $\boldsymbol{\psi} = \{\boldsymbol{\theta}_{1,2}, \boldsymbol{\theta}_{1,3}, \dots, \boldsymbol{\theta}_{Q-1,Q}\}$, where $\boldsymbol{Y_i}^{(p)}$ and $\boldsymbol{Y_i}^{(q)}$ are the observations in $\boldsymbol{Y_i}$ belonging to subsamples S_p and S_q, respectively.
- Similarly, the combination of all $\hat{\boldsymbol{\theta}}_{p,q}$ into a single estimator $\hat{\boldsymbol{\theta}}$ depends on the precise model and data structure.

The split-sample approach – Overlapping subsamples

- Shown in panel (d) of Figure 1, longitudinal dataset with very large n is partitioned similarly to dependent subsamples, but association between longitudinal observations is accounted for by letting the subsamples overlap
- Denoting the parameters in pair $\{\boldsymbol{Y_i}^{(p)}, \boldsymbol{Y_i}^{(q)}\}$ by $\boldsymbol{\theta}_{p,q}$, fitting the models on all pairs is equivalent to maximizing

$$p\ell(\psi) = \sum_{p<q} \sum_{i} \ell(\boldsymbol{Y_i}^{(p)}, \boldsymbol{Y_i}^{(q)} | \boldsymbol{\theta}_{p,q})$$

  with respect to $\psi = \{\boldsymbol{\theta}_{1,2}, \boldsymbol{\theta}_{1,3}, \dots, \boldsymbol{\theta}_{Q-1,Q}\}$, where $\boldsymbol{Y_i}^{(p)}$ and $\boldsymbol{Y_i}^{(q)}$ are the observations in $\boldsymbol{Y_i}$ belonging to subsamples S_p and S_q, respectively.
- Similarly, the combination of all $\hat{\boldsymbol{\theta}}_{p,q}$ into a single estimator $\hat{\boldsymbol{\theta}}$ depends on the precise model and data structure.

- without the pairwise fitting, we have to fit the model on the entire dataset, which is computationally infeasible when n is very large. By fitting the model on pairs of subsamples, we can reduce the computational burden while still accounting for the association between longitudinal observations.
- not gonna go into details here since this is more suitable for longitudinal data, which is not the focus of our presentation, but the idea is similar to dependent subsamples
- similarly, $\boldsymbol{\psi} = \{\boldsymbol{\theta}_{1,2}, \boldsymbol{\theta}_{1,3}, \dots, \boldsymbol{\theta}_{Q-1,Q}\} = \{\boldsymbol{\theta}_{p,q} : p < q\}$
- what if both n and N are large? there is no mention of this in the literature, should we mention this?

Handling Large Datasets in Multilevel / Mixed-like Models

Problem

- Large datasets make model fitting slow
- Main bottleneck: matrix crossproducts $X^T WX$
- Cost grows with sample size $O(np^2)$

Main Idea (Paper)

- Many covariate values repeat or are very similar
- Discretize / compress covariate space
- Compute crossproducts using unique / discretized values
- Reuse computations instead of recalculating per observation

Why This Helps Large Multilevel Data

- Repeated measurements per group
- Repeated random-effect design patterns
- Much faster computation while using full dataset

Practical Implementation

- R: `mgcv::bam(..., discrete=TRUE)`

Table of Contents

R Packages — `lme4`

- `lme4` is an R package for fitting linear and generalized linear mixed-effects (multilevel) models using 'Eigen' C++ library and S4 classes.
- It's computationally efficient, enabling it to handle very large sample sizes for simpler mixed models and to process hundreds of thousands of observations with random effects on a typical laptop.
- Modeling functions:
  - `lmer()`: fits linear multilevel model using restricted maximum likelihood (REML) or maximum likelihood estimation.
  - `glmer()`: fits generalized linear multilevel model, accommodating non-normal response distributions.
- Reference: `https://cran.r-project.org/web/packages/lme4/lme4.pdf`

R Packages – `mgcv`

- `mgcv` is an R package for fitting generalized additive models (GAMs) and generalized additive mixed models (GAMMs) using penalized regression splines.
- Modeling functions:
  - `gam()`: fits generalized additive multilevel models using penalized regression splines with smooth terms that can incorporate multilevel structure through random effect splines.
  - `bam()`: a computationally efficient version of `gam()` optimized for very large datasets.
- Reference:
  `https://cran.r-project.org/web/packages/mgcv/mgcv.pdf`

R Packages – Why use `bam()`?

- The underlying model between `gam()` function and `lme4` is the same, with differences only in the way parameters are estimated.
- `bam()` employs parallelized computation on model matrix subsets and optional data discretization to extract minimal necessary information, enabling efficient estimation of large multilevel models.
- With large enough datasets, this discretization has negligible impact on parameter estimates (differing only at high decimal precision), but leads to dramatic speed improvements.

R Packages – When to use `bam()`?

- In general, `lme4` is preferred for most multilevel datasets due to its straightforward syntax and robust estimation methods.
- `bam()` is particularly useful when:
  - The model structure is complex and computationally demanding for `lme4`
  - Smooth (nonlinear) terms are required
  - The dataset is very large and memory limitations arise
  - Parallel computing resources can be effectively utilized

Table of Contents

Summary

- Multilevel models are powerful tools for analyzing hierarchical data structures, allowing for accurate inferences and simultaneous estimation of group and individual effects.
- However, large multilevel datasets pose significant computational challenges, including memory constraints and slow estimation times.
- The split-sample approach offers a practical solution by partitioning data into manageable subsamples, enabling efficient model fitting while retaining key statistical properties.
- R packages like `lme4` and `mgcv` provide robust tools for fitting multilevel models, with `bam()` in `mgcv` being particularly suited for very large datasets due to its computational efficiency.
- Choosing the right approach and tools depends on the specific characteristics of the dataset and the research questions at hand.

Table of Contents

References I

- Centre for Multilevel Modelling (2025). *What are multilevel models and why should I use them?* University of Bristol. URL: https://www.bristol.ac.uk/cmm/learning/multilevel-models/what-why.html.
- Clark, Michael (2019). *Mixed Models for Big Data*. URL: https://m-clark.github.io/posts/2019-10-20-big-mixed-models/ (visited on 02/17/2025).
- Speelman, Dirk, Kris Heylen, and Dirk Geeraerts (2018). *Mixed-Effects Regression Models in Linguistics*. Springer International Publishing, pp. 11–28.
- Wood, Simon, Yannig Goude, and Simon Shaw (2015). "Generalized Additive Models for Large Data Sets". In: *Journal of the Royal Statistical Society Series C-Applied Statistics* 64, pp. 139–155. DOI: 10.1111/rssc.12068.