Journal of Data Science 24 (2), 455–475
April 2026
DOI: 10.6339/26-JDS1222
Computing in Data Science

Data-Driven Model Structure Diagrams for Hierarchical Linear Mixed Models

Greta M. Linse[1,*], Mark C. Greenwood[1], and Ronald K. June[2]

[1]*Department of Mathematical Sciences, Montana State University, Bozeman, Montana, United States of America*
[2]*Mechanical & Industrial Engineering Department, Montana State University, Bozeman, Montana, United States of America*

Abstract

Hierarchical linear mixed models are commonly used in many scientific fields. However, without a strong statistical background, it can be hard to understand the relationships between the random effect variables and the inferences that can be made when a model has nested random effects. Visualizing relationships makes it easier for the practitioner to understand what relationships the model is capable of estimating and testing. We present an R package *modeldiagramR* that seamlessly creates a visualization of the model based on the data and the model object created when fitting a linear mixed model using either *lme4* or *nlme*.

Keywords *data visualization; diagnostic plots; model assumption assessment; model visualization; multivariate high dimensional data; random effects*

1 Introduction

Across many scientific fields, complex data structures which include grouped and repeated measurements are the norm rather than the exception. Visualization of the data and statistical models are critical to understanding the relationships between the variables, cementing the information and conclusions that can be drawn. Simultaneously visualizing variables from these complex scenarios is challenging. Often the focus of the data visualization is on the primary variables of interest, the experimentally manipulated variables, or those involved in any hypothesis tests. This choice can mean that other influential or structural variables are not included, and as a result the representation of the relationships between the variables is incomplete. As the saying goes "a picture is worth a thousand words", and that is true even more so for complex data collection or statistical modeling scenarios. Solving this multivariate data visualization challenge can improve the communication of these statistical models for non-statisticians and statisticians alike.

Variables in a multivariate data set that are of direct interest, such as pertaining to an experiment or those being tested in hypotheses are typically called "fixed effect variables" or "fixed effects" for short. Linear models taught in an intermediate or an upper level college statistics course generally focus on one or more fixed effects, with or without interactions if multiple fixed effects are included in the model, such as in Ramsey and Schafer (2013). The variables that define the grouping structure, or to account for random selection of groups from a larger pool of possible groups, are called "random effect variables" or "random effects." Linear

*Corresponding author Email: greta.linse@montana.edu.

Received August 11, 2025; Accepted February 21, 2026

mixed effect models are models which include a mixture of fixed and random effect variables. These models provide a flexible statistical modeling framework for a variety of complex data collection scenarios and are described in detail in the foundational *Mixed-Effects Models in S and S-PLUS* text by Pinheiro and Bates (2000). When a random effect is included in a model, there is a very simple hierarchical structure imposed on the data with the levels of the random effect at the top of the hierarchy and the observations within each group level at the bottom. When there is more than one random effect in a model the hierarchy becomes more complex, possibly with a tree-like structure. Gelman and Hill (2021, p. 245) provide another perspective on modeling hierarchical data structures, calling random effects "the varying coefficients in a multilevel model." Linear mixed effect models in R are most commonly fit using either the *nlme* or the *lme4* packages which use different formula notation for these two types of variables, but can incorporate multiple fixed and random effects (Pinheiro et al., 2024; Bates et al., 2015).

A solution to visualize multivariate data typically involves bivariate graphs that plot the response against one or more fixed effect variables at a time, such as box, violin, or scatter plots. Two simultaneous multivariate data visualization techniques include table plots (Tennekes et al., 2021) and alluvial plots (Koneswarakantha, 2023). These have their own limitations, namely it is hard to visualize the different roles and contributions that the random effects and their imposed hierarchy have with both the fixed effect and the response variable. As the complexity of the hierarchical structure grows, the visualization challenge also increases. Zavez and Harel (2024) created an R package to draw the model structure in a hierarchical diagram in the *LearnVizLMM* package. Their solution is to use the model formula and other information provided directly by the user to construct the diagram. The resulting diagram is informative but can be further improved on by using more of the information available from the data and the statistical model, while requiring less input from the user.

In communicating the study design and model used, a good model visualization that includes all variables, can be as important as visualization of the data themselves or exploring the estimated coefficients from the model summary. When discussing linear mixed effect models in a classroom or with clients in a statistical consulting setting, we would draw model diagrams by hand. Alternatively, we would manually create them a node and an edge at a time with *DiagrammeR* (Iannone and Roy, 2024). The manual approach is useful, but is a tedious artistic or programming exercise and, in the case of hand-drawn diagrams, not of publication quality. With the advent of large language models (LLMs) and generative AI (GenAI), much of the programming or drawing could be outsourced to an LLM. However, using an LLM to create a model diagram does not remove the need to double check the diagram for accuracy, and replicating diagramming functions would require sharing data that may not be available to publicly share. With a dedicated package that has been tested and vetted by researchers and students, that draws the diagram with minimal input, there is greater trust that the diagram is operating as intended without needing to create and vet the code. We have developed such a package in R *modeldiagramR* with the function `model_diagram()` that will take a hierarchical mixed effects model and automatically draw a model diagram for both balanced and unbalanced designs. Informed by the data and the model, the diagram function will place both the random and fixed effects in nodes and draw connections between the random effects. Optional arguments allow for control of colors, font sizes, layout, diagram size, and exporting to a file.

Linear mixed effect models are described in greater detail below, along with the alternative multivariate and model diagram visualization approaches. Then three examples of our model diagram are presented, building in complexity. First visualizing a model on data from a simple balanced matched pairs experiment, next from an experiment conducted on dolphins in an

unbalanced observational scenario, and lastly on a complex simulated balanced multi-site split-split-plot design.

2 Methods

2.1 Linear Mixed Effects Models

Linear mixed effects models are models that include both fixed and random effects with an identity link and Gaussian (normal) response. These models are often used to account for repeated measurements on an observational unit or to account for random selection of groups from a larger pool of possible groups. For example, Zuur et al. (2009) use a linear mixed model to compare three different staining methods for identifying the age of different cetacean species from dolphins that were stranded on beaches in two locations, so there are repeated measurements for the same animal. On a more basic level, the simplest hierarchical model is one that has two measurements for each subject, otherwise known as a matched pairs design when the treatment is applied at the observation level. Luetkemeier et al. (2017) collected data on the sweat rate from ten subjects from a location on the subject's body with a tattoo and a matched (similar) location on the same subject without a tattoo. These data could be used to assess if the presence of a tattoo had a different sweat rate after accounting for the variability between the individual subjects. Lastly, in experimental designs such as split-plot or split-split-plot designs, the random effects correspond to block, whole-plot, split-plot factors. As an example of this, we simulated data based on the experimental design in the documentation of the `split_split_plot()` function from Murillo and Gezan (2024). Although not presented here, for another example of a linear mixed model, Gelman and Hill (2021, pp. 252–283) use hierarchical models to account for variability of log-radon levels in houses between counties within the United States, as radon is a gas found in regions with certain geologic features that would be expected to have amounts that are more similar on a local level than on a state-wide or country-wide level.

In all of these cases, the linear mixed effects model consists of a response variable, an intercept, a treatment or other fixed effects, group variability, and observational level variability as random error,

$$\text{Response} = \text{Intercept} + \text{Treatment} + \text{Group variability} + \text{Observational error}.$$

If we first start with the case of one fixed and one random effect, we could represent the linear mixed effect model for the j^{th} response in the i^{th} group, Y_{ij}, as:

$$Y_{ij} = \beta_0 + \beta_1 X_{ij} + \text{Group}_i + \varepsilon_{ij},$$

where:

β_0 is the intercept,

β_1 is the coefficient for the fixed effect X,

$\text{Group}_i \overset{iid}{\sim} N(0, \sigma^2_{\text{Group}})$,

$\varepsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$,

Group_i and ε_{ij} are independent,

$i = 1, \ldots, I$ levels of the group/random effect,

$j = 1, \ldots, r_i$ observations for each level of the group/random effect, and

$n = \Sigma r_i$ is the total number of observations.

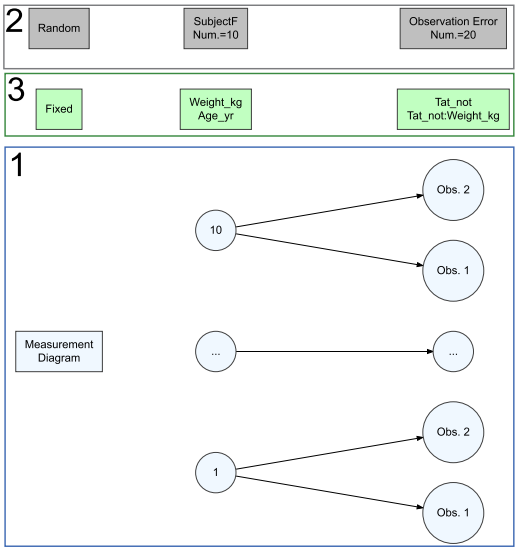


Figure 1: Model diagram for the `combinedtattoo` data set annotated where (1) the nodes in the blue outlined box with light blue fill indicates the "measurement diagram", (2) the nodes in the gray outlined box with gray fill indicate the random effects, and (3) the nodes in the green outlined box with green fill indicate the fixed effects.

Usually we have more than one random effect with multiple fixed effects, including one or more treatments, with additional covariates that need to be accounted for in the model. Names for these types of models can include matched pairs designs, repeated measures analysis of variance (r(M)ANOVA) or longitudinal analysis of variance (ANOVA), split-plot (or more complicated split-split-plot, etc.) designs, hierarchical or multilevel models, or clustered observational studies. This is not an inclusive list, and while there are some differences in analysis methods or requirements for these models, they can be thought of generally as mixed effects models. As the complexity of the model increases, the importance of visualizing the model also increases in order to ensure that it is correctly specified.

Random effects drive the structure of the model, but the fixed effects are usually the reason for fitting the model, and it is just as important to understand their role in the model. The level that the fixed effects vary at determines the inferences that can be made for the results from those components in the model. For example, in a model with repeated measurements on subjects, if fixed effects are age and treatment, age might be at the subject level (if all measurements occur in the same day or age is measured at the first time, then age is the same for all responses for each subject) and treatment might be at the observation level (if it varies across the repeated measures). Our model visualization tool can be expanded to many, if not all, generalized linear mixed effects models with a nested hierarchical structure. However, for the sake of simplicity, the methods and examples below will focus on linear and not generalized linear mixed effects models.

2.2 Model Diagram Structure

There are three parts to our model diagram, (1) the "measurement diagram" (Section 2.2.1), (2) the random effects (Section 2.2.2), and (3) the fixed effects (Section 2.2.3) that combine to fully explore all the aspects of the model as shown in Figure 1. The model and data set used for the annotated diagram are discussed in Section 4.2.

2.2.1 Measurement Diagram

The "measurement diagram" is a directed graph showing the grouping structure and hierarchical relationships between the random effects. In Figure 1 these are the nodes in the box labeled "1" with a light blue fill color. When oriented vertically, this directed graph moves from left "top level" to right "observational level." In the horizontal orientation, the directed graph moves from the top to the bottom where the bottom nodes represent the "observational level." Depending on the scientific field, the levels may be numbered from the bottom to top or top to bottom, so we avoid numbering the levels. The diagram is drawn by using the group structure object provided by the model output from *nlme*'s `lme` function. The group structure object explicitly lists the order of the nesting structure and creates unique labels of levels of nested random effects, so if unique labels are not provided this will not affect the diagram. In order to visualize structures with either large number of levels for each random effect group, a large number of levels in the nesting structure, or both, only the first and last group at each level is drawn in the diagram, and all of the other middle levels (if there are three or more levels for a particular random effect) are represented with ellipses ("..."). This aids in faster rendering times and results in a diagram that is smaller and more publication friendly.

2.2.2 Random Effects

The second part of the diagram is a set of boxes at the top (in a row for vertical orientation) or right hand side (in a column for horizontal orientation) listing the random effect variables that correspond to the group levels in the measurement diagram. In Figure 1 these are the nodes in the box labeled "2" with a light gray fill color. Along with the variable name the number of levels for the random effect are reported. This is based on the group size information provided in the model output, and not the number of subjects or sites. This helps with understanding the number of observations available at each level of the study or design. The "top" level (level on the left of the diagram) is the random effect that all other random effects are nested within. The "bottom" level is labeled "Observation Error" and indicates how many total observations there are according to the model. In between, the random effect labels for nested variables are written in plain English (*i.e.*, "Day in Subject") to make the diagram accessible to a wide audience.

By extracting and illustrating the random effects based on the levels in the model, the hierarchical order of the random effects is made explicit for the user. Combined with the measurement diagram, this highlights potential errors in the nesting order in multilevel models that the researcher or student has imposed. The hierarchical tree and random effect alignment helps to understand the model implications for implied correlation of observations at different levels where connected lines mean observations are correlated and the closer the connection, the more correlated they are. It also helps to identify the potential for understanding and calculating various intra-class correlations (see Zuur et al. (2009), for example) that relate to observations connected at different levels of the diagram.

2.2.3 Fixed Effects

The third part of the diagram is a row of boxes containing the fixed effects, which is placed between the row of random effect labels and the measurement diagram. In Figure 1 these are the nodes in the box labeled "3" with a green fill color. If a level does not have any associated fixed effects the box remains empty. If the amount of variation for a fixed effect variable is consistent with the number of levels for the random effect at that level, the fixed effect, or

interactions between fixed effects, is listed at that level. While model diagrams were designed with mixed model visualization in mind, they can work without any fixed effects in the model such as for random effect only models.

The placement of fixed effects in the diagram implies the level of variation that it can explain. For example, in a longitudinal study with repeated measures on subjects, demographic variables that do not vary over time only can explain subject-to-subject differences in the response but are unable to explain variation over time. If the predictor values are repeated for a fixed effect, we can determine if the value is constant across a random effects level for a particular node. The denominator degrees of freedom for the related F-test reflect that the fixed effect occurs at that level and not at a lower level of the hierarchy of the design, although the results depend on the method used, such as the *nlme* rules, Kenward-Roger, or Satterthwaite (Kuznetsova et al., 2017). By labeling the level of fixed effects, researchers can consider the interpretation of the fixed effect more carefully and also consider issues such as available degrees of freedom at each level of the study design. This diagramming process can be particularly useful for quantitative predictors to explore potential aggregating or centering at different levels of a design (such as those discussed in Shönbrodt (2014)).

2.3 Compatibility

Because we use R (R Core Team, 2024) for fitting mixed models, the `model_diagramR()` function has been developed based on `lme` objects from the *nlme* package (Pinheiro et al., 2024), which are relatively more explicit about the hierarchy of the terms than `merMod` objects from *lme4* (Bates et al., 2015). Currently, any model that can be translated into an `lme` object can directly interface with the current version of the diagramming function; future work will focus on generating diagrams based on the `merMod` objects from *lme4* and related packages that cannot be coerced into `lme` objects. As accurate representation of the response variable and its relationship to the fixed and random effects is not a requirement, for generalized linear mixed models, this relationship may be simplified by converting binary or count responses to quantitative responses for linear mixed models without loss of information about the model structure.

2.4 Space Constraints

As the number of observations grows at more than a doubling rate in hierarchical study designs when there are more than two observations per higher level observation; the data sets grow quite large as the number of levels increase. Also, many hierarchical study designs leverage generally "large" data sets, at least at the observation level of the diagram. Because of this, it is not possible to draw out every observational unit node at every level of the tree-structure, in most cases. To facilitate a reasonable visual characterization of the study design, we focus on a conceptual depiction instead of a fully rendered version of the tree. This is accomplished by identifying the first and last level of each random effect group at each level of the hierarchy and at the observational level, with the rest of the levels collapsed using ellipses. This allows large data sets to be qualitatively explored without onerous computational rendering or space needs for the diagram. We also rotate the diagrams by default vertically to orient left-to-right for rendering of the diagram, although diagrams can be oriented horizontally for a top-to-bottom layout.

3 Alternative Approaches

3.1 Alluvial Diagrams

An alluvial diagram is a way to visualize multivariate data sets (particularly factor and categorical variables or binned quantitative variables) and the relationships between the variables. Thus, alluvial diagrams provide a data-based way to explore hierarchies of variables. It is a way to show the state transitions and potential level combinations across the data set. It is also a good way of visualizing a few variables from large data sets simultaneously. Another way to visualize multivariate categorical data was the `tableplot()` function from the *tabplot* package (Tennekes et al., 2021) which is no longer maintained. While `tableplot()` was a good way to visualize large multivariate data sets, it did not show the direct connections between the variables, which is a feature of an alluvial diagram. Both alluvial diagrams and table plots attempt to solve the problem of visualization of high-dimensional multivariate data sets as a multi-way contingency table, displaying only the unique combinations of each level within previous levels, but both methods have their own limitations.

While there are numerous packages that provide functions to create alluvial diagrams (*e.g.*, *easyalluvial*, *ggalluvial*, *ggsankeyfier*, and *alluvial*), we will focus on the diagram created by *easyalluvial* (Koneswarakantha, 2023) with assistance from *forcats* (Wickham, 2023a) to ensure that the level used for organizing the flows is sorted in an order of our choosing. To draw an alluvial diagram, all possible interactions or combinations between all levels for all the variables are identified, which are called "alluvia", and the number of observations (and proportions) for each of those interactions are calculated. That determines the width of each flow going between pairs of variables. The colors can either be set based on the left most variable or the right most variable. In interactive alluvial diagrams the flows can be reordered to make patterns in the flows between variables more visible.

The advantages to an alluvial diagram for the variables used to define random effects to visualizing model structure as compared to `model_diagram()` is that you can visualize all levels of each random effect and not just the first and last levels. If the grouping structure from an `lme` model object is not used, then particular care needs to be taken that the nested levels are named uniquely, otherwise the nested structure will appear crossed as observations from multiple higher order levels will flow into the same nested random effect. As all observed levels are drawn (with unique names) you can see the proportion of each flow that comes from the "top" level random effect down into the lowest (most inner) nested random effect.

On the other hand, the disadvantages include the fact that larger more complicated nesting structures can become harder to visualize and more overwhelming as the number of pieces of the diagram increases. While the "measurement diagram" has observational error as the bottom level, it would make a very tedious and hard to visualize alluvial diagram if the right most variable only had a single observation for each unique combination. Thus you would need to remove this level. Additionally, as all of the combinations would need to be uniquely named, the labels would overlap in the diagram and would need to be suppressed or selectively included. Also the fixed effects do not have a clear place in the diagram. For hierarchical structures with three or more nested random effects, groups may be indistinguishable. Alluvial diagrams can be colorful, but they can also be harder to explain. Lastly, for manuscripts that need to have black and white friendly graphics, alluvial diagrams would be almost impossible to make work as they heavily rely on different colors to identify flows between variables and color-blind friendly palettes have a limited number of distinguishable levels.

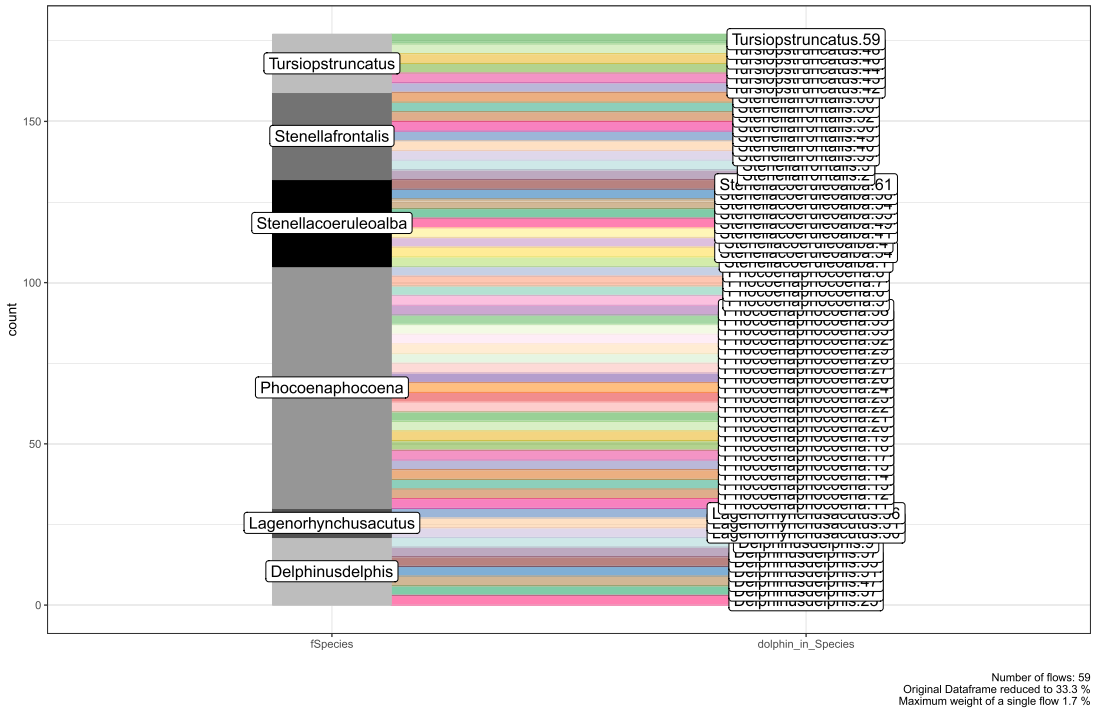


Figure 2: Alluvial diagram for the random effects of individual dolphins within the six species, not including the repeated measurements for each dolphin in this visualization for the `cetaceans` data set.

```
library(modeldiagramR) # For data set
library(forcats) # For sorting interaction level
library(easyalluvial) # For alluvial_wide() function
data("cetaceans")
cetaceans_data <- cetaceans %>%
  mutate(dolphin_in_Species = paste0(fSpecies, ".", fDolphinID),
         dolphin_in_Species = fct_relevel(dolphin_in_Species, sort)) %>%
  dplyr::select(fSpecies, dolphin_in_Species)
# Figure 2
alluvial_wide(cetaceans_data, fill_by="last_variable")
```

In Figure 2, we visualize the "Cetaceans" data from Zuur et al. (2009) in an alluvial diagram using the provided code – this example is discussed further and our approach is demonstrated in Section 4.3. In the alluvial diagram we can see the relative proportions of each species and that *Phocoenaphocoena* is the species with the most stranded dolphins in the data set. Only the top label for the individual dolphins is legible and we are not able to capture the three measurements that occur for each animal. Each alluvium on the left goes to one of the combinations on the right. Information is provided about the total number of flows (individual dolphins in this case) and the fact that the original "dataframe" is reduced to 33.3%, since the observational error (repeated measurements from each dolphin) is not included in the diagram and there were three observations per dolphin. So while this is useful, it only tells part of the story.

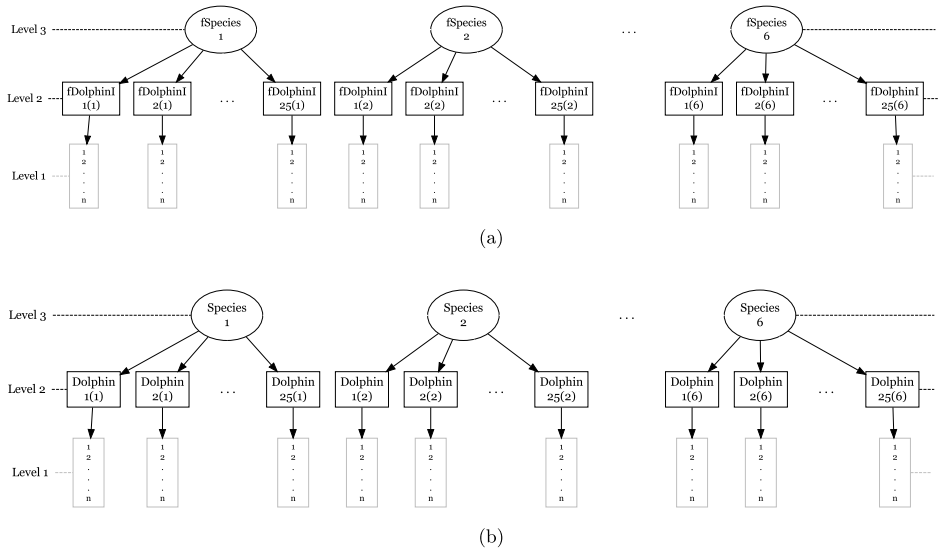


Figure 3: Alternate versions of the model diagram created by *LearnVizLMM* for the `cetaceans` data set, (a) the model diagram after specifying each of the two random effects not at the observation level and the maximum number of levels for each of the random effects; (b) the model formula string and the number of levels of the top "grouping factor".

3.2 *LearnVizLMM*

Another approach to visualizing a linear mixed effects model structure, which is more similar to our approach, is found in the *LearnVizLMM* package by Zavez and Harel (2024). This approach focuses on visualizing the model formula without needing data or a model object to draw the diagram. This makes this method particularly useful during the experimental design phase, before even simulated data are created. The `extract_structure()` function in this package takes either a model formula and the number of levels in the highest order random effect or each random effect with maximum group size information and creates a model diagram cartoon based on the structure and the provided number of groups (Figure 3a). As actual (or simulated) data are not included, this diagram does not contain information about the number of observations at each level unless it is explicitly provided (Figure 3b) and cannot place the fixed effects at the appropriate level in the model diagram, and so does not attempt to visualize the fixed effects.

```
library(LearnVizLMM) # For extract_structure() function
# Figure 3a
extract_structure(n_gf=2,
                  gf_nlevels = c(6,25),
                  gf_description = "nested",
                  gf_names = c("Species", "Dolphin"),
                  export_type = "text",
                  )
# Figure 3b
extract_structure(model = "lmer(Age ~ fSex * fStain * fLocation +
                  (1|fSpecies/fDolphinID)",
                  gf_nlevels = c(6,25),
                  export_type = "text")
```

This package can also create preliminary LaTeX code to include the regression model equation in manuscripts as illustrated for the `cetaceans` data using `extract_equation()` to generate Equation (1). The function does need modification to adjust the width of the text and to remove an extra term that is created for the random effects pieces of a `lmer` model formula string. Additionally, while you can specify the number of levels for each fixed effect, you cannot specify their names or the number of levels for the random effects. The fixed effect components do not include subscript notation that relates to subscripts on the response or random effects. However, for someone inexperienced in writing model equations in LaTeX this could be a useful place to start.

```
library(LearnVizLMM) # For extract_equation() function
extract_equation(model = "lmer(Age ~ fSex * fStain * fLocation +
                  (1|fSpecies/fDolphinID)",
                 cat_vars = c("fSex", "fStain", "fLocation"),
                 cat_vars_nlevels = c(2,3,2))
```

$$
\begin{aligned}
\text{Age}_{ijk} = {} & \beta_0 + \beta_1(\text{fSexB}) + \beta_2(\text{fStainB}) + \beta_3(\text{fStainC}) + \beta_4(\text{fLocationB}) \\
& + \beta_5(\text{fSexB*fStainB}) + \beta_6(\text{fSexB*fStainC}) + \beta_7(\text{fSexB*fLocationB}) \\
& + \beta_8(\text{fStainB*fLocationB}) + \beta_9(\text{fStainC*fLocationB}) \\
& + \beta_{10}(\text{fSexB*fStainB*fLocationB}) + \beta_{11}(\text{fSexB*fStainC*fLocationB}) \\
& + u_{0i} \\
& + v_{0j(i)} \\
& + \varepsilon_{ijk} \\
u_{0i} \sim {} & N\left(0, \tau^2_{u_0}\right), \text{ for fSpecies i} = 1, \ldots, \text{a} \\
v_{0j(i)} \sim {} & N\left(0, \tau^2_{v_0}\right), \text{ for fDolphinI j} = 1, \ldots, \text{b} \\
\epsilon_{ijk} \sim {} & N\left(0, \sigma^2\right), \text{ for Observation k} = 1, \ldots, \text{n}.
\end{aligned}
\tag{1}
$$

We feel that our function improves on the diagrams in Figure 3 with using more of the model information available as well as the data set. Our approach does require some data that a model could use, so if data have not been collected yet, data will need to be simulated. Our function does not yet provide LaTeX equation notation based on the chosen model (and data).

4 Examples

Three examples are presented below, for each example the data used are included in *modeldiagramR* so all results can be replicated.

4.1 Installing and Function Description

For *modeldiagramR*, R must be at least version 4.1, and it uses *base* and the following other R Core Team (2024) packages: *stats*, *utils*, and *methods*. Additional packages needed include *stringr* (Wickham, 2023b), *dplyr* (Wickham et al., 2023), *tidyr* (Wickham et al., 2024), *tidyselect* (Henry and Wickham, 2024), *magrittr* (Bache and Wickham, 2022), *tibble* (Müller and Wickham, 2023), *gtools* (Warnes et al., 2023), *forcats* (Wickham, 2023a), *nlme* (Pinheiro et al., 2024), *DiagrammeR* (Iannone and Roy, 2024), and *DiagrammeRsvg* (Iannone, 2016). To visualize model diagrams for objects created by *lme4* (Bates et al., 2015), that package will also need to be installed.

The *modeldiagramR* package can be installed from GitHub using the following code:

```
install.packages("remotes")
remotes::install_github("glinse-stat/modeldiagramR")
```

The only required argument for the `model_diagram()` function is either a "`merMod`" object with nested random effects created by using the `lmer` or `glmer` functions from the *lme4* package, or an "`lme`" object created by using the `lme` function from the *nlme* package. There are additional arguments that adjust the layout, diagram size, adjust orientation, colors, shift labels, return a `grViz` object, or export the diagram directly to a file. The required and optional arguments are listed in Table 1.

4.2 Matched Pairs Design

The matched pairs example presented in Luetkemeier et al. (2017), has a single random effect variable (subject) and with two observations per subject. The data from this study were obtained by digitizing the plotted data from the paper. As a result, the results from our version do not exactly replicate the original results. The digitized data are included in our package *modeldiagramR*, under the name `combinedtattoo`.

While the matched pairs design is straight-forward, it can be difficult to understand the information level of the fixed effect variables which in this case are condition (skin was tattooed or not) and covariates including weight and age. For this data set there were ten subjects, so only the first and last alpha-numerically will be included (subjects "1" and "10"). While not a requirement for a linear mixed effects model from the *lme4* package, the subject ID has been converted to a factor variable from a numeric variable as indicated with the "f" at the beginning of the variable name. For this example, since there are only two observations for each subject, both of those will be included in the model diagram ("Obs. 1" and "Obs. 2") (Figure 4). The fixed effects are allocated to either subject or observation level, clarifying their role in the model and suggesting notation for the model of:

$$\begin{aligned}\text{SweatRate}_{ij} =& \beta_0 + \beta_1 I_{\text{Tat\_not=not},ij} + \beta_2 \,\text{Weight\_kg}_i + \beta_3 \,\text{Age\_yr}_i + \\ & \beta_4 I_{\text{Tat\_not=not},ij} * \text{Weight\_kg}_i + \text{Subject}_i + \varepsilon_{ij},\end{aligned}$$

where:

$$\begin{aligned}\text{SweatRate}_{ij} &\text{ is the response for the } j^{\text{th}} \text{ measurement on the } i^{\text{th}} \text{ subject,}\\ \beta_0 &\text{ is the intercept,}\\ \beta_1, \ldots \beta_4 &\text{ are the coefficients for the fixed effects,}\\ I_{\text{Tat\_not=not},ij} &\text{ is 1 if the location of the } j^{\text{th}} \text{ measurement on the } i^{\text{th}}\\ &\text{subject was not tattooed and 0 if the location was tattooed,}\\ \text{Weight\_kg}_i &\text{ is the weight in kg for the } i^{\text{th}} \text{ subject,}\\ \text{Age\_yr}_i &\text{ is the age in years for the } i^{\text{th}} \text{ subject,}\\ \text{Subject}_i &\overset{iid}{\sim} N(0, \sigma^2_{\text{Subject}}),\\ \varepsilon_{ij} &\overset{iid}{\sim} N(0, \sigma^2),\\ &\text{Subject}_i \text{ and } \varepsilon_{ij} \text{ are independent,}\end{aligned}$$

Table 1: Arguments for the `modeldiagramR::model_diagram()` function.

Argument	Default	Description
modelObject[1]		Input model. Either an `lme` or `merMod` (including `glmerMod`) object with a nested random effects structure.
filePath	NULL	Path to a location to export the diagram.
fileType	`"PNG"`	File type to export the diagram.
width	800	Width of diagram in pixels.
height	1600	Height of diagram in pixels.
includeSizes	TRUE	Include group sizes in random effect labels.
includeLabels	TRUE	Include labels for the model diagram components.
orientation	`"vertical"`	Orientation of the diagram, either vertically or horizontally. Options are `"vertical"` or `"horizontal"`.
scaleFontSize	1	Proportional font size adjustment for model diagram component, fixed effect, and random effect labels. Multiplies these label font sizes by the specified amount.
shiftFixed	0	Additive x axis adjustment for fixed effect labels, only used when `orientation == "horizontal"`.
shiftRandom	0	Additive x axis adjustment for random effect labels, only used when `orientation == "horizontal"`.
nodeColors	`md_color( diagram = "gray25", random = "gray25", fixed = "gray25")`	Function specifying the colors (`md_color()`) for the outline of the nodes. Components can be specified individually (`diagram`, `random`, and `fixed`).
nodeFillColors	`md_fill( diagram = "aliceblue", random = "aliceblue", fixed = "darkseagreen1")`	Function specifying the colors (`md_fill()`) for the fill color of the nodes. Components can be specified individually (`diagram`, `random`, and `fixed`).
nodeFontColors	`md_fontColor( diagram = "black", random = "black", fixed = "black")`	Function specifying the colors (`md_fill()`) for the font color of the nodes. Components can be specified individually (`diagram`, `random`, and `fixed`).
exportObject	1	Object to return, `1` for a rendered DiagrammeR graph as an SVG document, `2` for a grViz editable object, and `3` both.

[1]Required, all other arguments are optional.

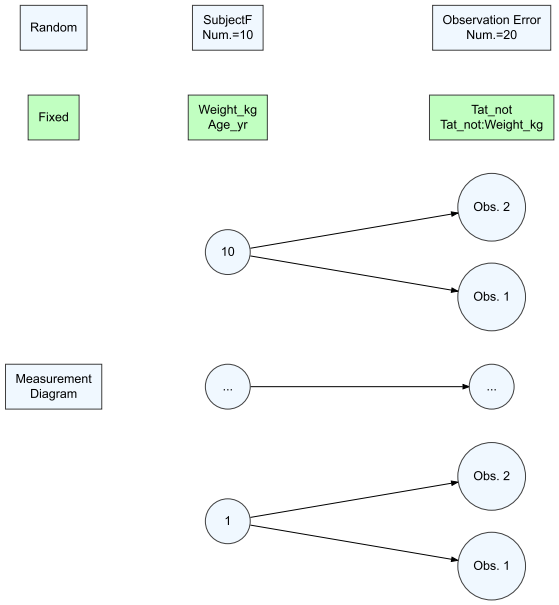


Figure 4: Model diagram for the matched pairs design for sweat rates with subject as the random effect, two observations per subject, and treatment, weight, and age as fixed effects. This diagram is the same diagram presented in Figure 1 without boxes around the model diagram components.

$$
\begin{aligned}
&i = 1, \ldots, 10 \text{ subjects,} \\
&j = 1, 2 \text{ observations for each subject, and} \\
&n = 20 \text{ is the total number of measurements/observations.}
\end{aligned}
$$

The diagram (Figure 4) allows the viewer to identify that weight and age were subject level, but tattoo presence/absence varies within subject, leading to the use of i or ij subscripts in the model. All model diagram default parameters are used in this example and only the `lmer` model object is provided to the `model_diagram()` function.

```
library(lme4) # For lmer() function
library(modeldiagramR) # For data and model_diagram() function
data("combinedtattoo")
combinedtattoo <- combinedtattoo %>%
  mutate(fSubject = factor(Subject))

lmer_tat <- lmer(SweatRate ~ Tat_not * Weight_kg + Age_yr +(1|fSubject),
                 data = combinedtattoo)
# Figure 4
model_diagram(lmer_tat)
```

4.3 Three-Level Hierarchical Nested Design

For demonstration purposes we use the "Cetaceans" data from Zuur et al. (2009, p. 460) as they include a cartoon representation of the hierarchical model structure that was the original

inspiration for visualizing the hierarchical structure of the random effect variables. The data set contains information on dolphins from six species that were stranded and died as a result. Researchers wanted to know the age of the dolphins at the time of their stranding, and if there were differences in the age based on the staining method that was used to determine the age. Multiple dolphins of each species were sampled and three tooth staining methods used to determine the age of each dolphin. As the strandings were naturally occurring events, the number of animals in each location and of each species were unbalanced, although the number of teeth stained for each animal are always the same. Fixed effects included the sex, location, and staining method. The model used in the text, which we reproduce, fits a three-way interaction with sex, location, and staining method, and the dolphin within species and species random effects. This complex model can be written as in Equation (2).

$$\begin{aligned}
\text{Age}_{ijk} = {} & \beta_0 + \beta_1 I_{\text{Sex=2},ij} + \beta_2 I_{\text{Stain=Mayer},ijk} + \\
& \beta_3 I_{\text{Stain=Toluidine},ijk} + \beta_4 I_{\text{Location=Spain},ij} + \\
& \beta_5 I_{\text{Sex=2},ij} * I_{\text{Stain=Mayer},ijk} + \beta_6 I_{\text{Sex=2},ij} * I_{\text{Stain=Toluidine},ijk} + \\
& \beta_7 I_{\text{Sex=2},ij} * I_{\text{Location=Spain},ij} + \beta_8 I_{\text{Stain=Mayer},ijk} * I_{\text{Location=Spain},ij} + \\
& \beta_9 I_{\text{Stain=Toluidine},ijk} * I_{\text{Location=Spain},ij} + \\
& \beta_{10} I_{\text{Sex=2},ij} * I_{\text{Stain=Mayer},ijk} * I_{\text{Location=Spain},ij} + \\
& \beta_{11} I_{\text{Sex=2},ij} * I_{\text{Stain=Toluidine},ijk} * I_{\text{Location=Spain},ij} + \\
& \text{Species}_i + \text{Dolphin}_{ij} + \varepsilon_{ijk},
\end{aligned} \tag{2}$$

where:

Age_{ijk} is the k^{th} response of the j^{th} dolphin of the i^{th} species,
β_0 is the baseline mean age for Sex = 1, Stain = Elrich, and stranding Location = Scotland,
$\beta_1, \ldots, \beta_{11}$ are the coefficients for the fixed effects,
$I_{\text{Sex=2},ij}$ is the indicator variable for Sex $= 2$, coding of sex is unknown,
$I_{\text{Stain=Mayer},ijk}$ is the indicator variable for Stain = Mayer,
$I_{\text{Stain=Toluidine},ijk}$ is the indicator variable for Stain = Toluidine,
$I_{\text{Location=Spain},ij}$ is the indicator variable for stranding Location = Spain,
$\text{Species}_i \overset{iid}{\sim} N(0, \sigma^2_{\text{Species}})$,
$\text{Dolphin}_{ij} \overset{iid}{\sim} N(0, \sigma^2_{\text{Dolphin in Species}})$,
$\varepsilon_{ijk} \overset{iid}{\sim} N(0, \sigma^2)$,
$\text{Species}_i, \text{Dolphin}_{ij}, \varepsilon_{ijk}$ all independent,
$i = 1, \ldots, 6$ species,
r_i is the number of dolphins in species i,
$j = 1, \ldots, r_i$ dolphins in the i^{th} species,
$k = 1, 2, 3$ staining method for dolphin ij, and
$n = 59 * 3 = 177$ total observations from 59 dolphins.

4.3.1 Incorrectly Specified Model

A student or a researcher unfamiliar with the formulas for random effects may flip the order of the terms for the random effects structure. While the output from a linear mixed effect model can be used to check model specification, to the inexperienced user this may be hard to identify. A model diagram based on the data can make this mistake clearly visible, allowing for the model formula to be corrected. Additionally, if there are redundant or aliased random effects, this approach can identify that issue.

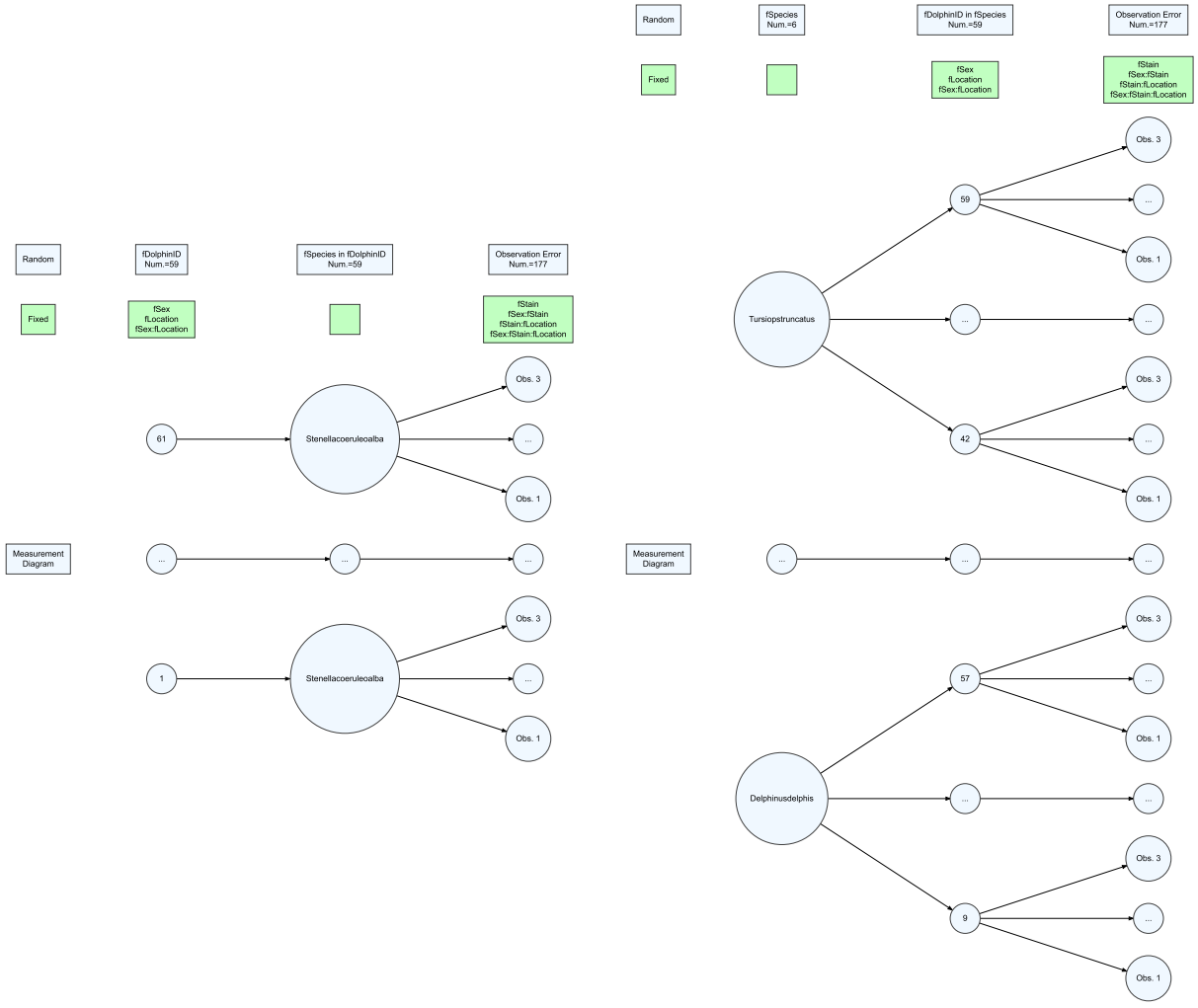
Each dolphin can only belong to one species, and for this example, we believe there are multiple dolphins from each of the six species. We also know that the fixed effects include the interaction between sex, location, and staining method, all two-way interactions, and each main effect, so we are confident that the fixed effects portion of the model should look like `fSex*fLocation*fStain` (R's short-hand for the three-way interaction and all lesser interactions and main effects). If we try to write the random effects the way we speak "dolphin nested in species" it might look like `Dolphin/Species`. As a random effect term that would be `1|Dolphin/Species` regardless of the modeling function. The code to fit that model and to examine the model diagram from our package is below, with only default parameters used for the `model_diagram()` function.

```
library(lme4) # For lmer() function
library(modeldiagramR) # For data and model_diagram() function
data("cetaceans")
lmer_cetaceans_d_slash_s <- lmer(Age ~ fSex * fLocation * fStain +
                                       (1|fDolphinID/fSpecies),
                                 data = cetaceans)
# Figure 5a
model_diagram(lmer_cetaceans_d_slash_s)
```

It should be readily apparent that the model is not specified correctly as in the diagram (Figure 5a) each dolphin only has one species, and based on the dolphin IDs selected, both of these animals have the same species, suggesting that dolphin is aliased with species. Since we know that there should be more than one dolphin per species, this indicates we have possibly specified our model incorrectly. One way of incorrectly specifying the model is having the random effects in the wrong order. However, in a situation with just a few animals, it is possible that the dolphin and species variables are truly aliased (one dolphin per species, one species per dolphin). In that case, only one random effect would be needed – either dolphin or species, but not both, and the diagram would also help to detect that issue. Visualizing the model object fit on the actual data in this case is vital to detecting the one-to-one relationship, otherwise an incorrect hierarchy could be imposed.

4.3.2 Correctly Specified Model

The correctly specified hierarchical nesting structure is repeated measurements for each dolphin within species (`1|Species/Dolphin`) which we can visualize using our model structure diagram in Figure 5b.



(a) (b)

Figure 5: The model diagram for the `cetaceans` data set with (a) the model specified incorrectly and (b) the model specified correctly.

```
library(lme4) # For lmer() function
library(modeldiagramR) # For data and model_diagram() function
data("cetaceans")
lmer_cetaceans_s_slash_d <- lmer(Age ~ fSex * fLocation * fStain +
                                   (1|fSpecies/fDolphinID),
                                 data = cetaceans)
# Figure 5b
model_diagram(lmer_cetaceans_s_slash_d)
```

It can be verified that the fixed effects are correctly placed at the level of the structure that `lme` uses to determine the degrees of freedom. In both diagrams (Figures 5a and 5b), sex, location and the interaction between sex and location are specified at the dolphin level, where stain and all other interactions including the three-way interaction between sex, location, and

stain, are at the observation or random error level. The information from the correctly specified diagram can be used to confirm the subscripts and notation used in writing out the model in Equation 2.

4.4 Simulated Multi-Site Split-Split-Plot Design

The model diagram function is also very useful with simulated data created during the experimental design phase, especially in designs with hierarchically defined observational units like split-plot or split-split-plot. It can help to visualize the design, make sure all desired aspects of the model are estimable (particularly at the whole or first split-plot levels), and ensure the correct model is fit.

To simulate this data set, the experimental design for a multi-site split-split-plot without replication was generated using the *FielDHub* package (Murillo and Gezan, 2024) and was based on the second example from the help file for `split_split_plot()` with modifications including reducing the number of split-split-plots (varieties of beans) from ten to four, and removing the replication. This design works when replication is included, including replications just adds another level to the hierarchical diagram that was not deemed necessary here.

A researcher might design an experiment for three locations (sites A, B, and C) where in each location, the field is split into two whole-plots with a treatment applied to each (irrigation or no irrigation), five split-plot treatments (four fungicides and one control), with four varieties of beans in split-split-plots in each location. There might be some prior information about the effect of the treatments and treatment combinations on the yield of beans, or that information could be randomly generated. In this simulation study, information about the main effects and the interaction adjustments for the irrigation and fungicide are specified, the other interaction adjustments are randomly drawn from various ranges of integer values centered around zero. The random effects for the location, whole-plot, split-plot, and split-split-plots are randomly and independently drawn from normal distributions with mean zero. The standard deviations can either be from known values or chosen randomly.

The regression model with informative subscripts on the indicator variables (additive main effects only) can be written as:

$$\begin{aligned} Y_{hijkm} =& \beta_0 + \beta_1 I_{\text{Irrigation=Yes},hi} + \beta_2 I_{\text{Fung.=1},hij} + \beta_3 I_{\text{Fung.=2},hij} + \beta_4 I_{\text{Fung.=3},hij} + \\ & \beta_5 I_{\text{Fung.=4},hij} + \beta_6 I_{\text{Bean=2},hijk} + \beta_7 I_{\text{Bean=3},hijk} + \beta_8 I_{\text{Bean=4},hijk} + \\ & \text{Location}_h + \text{Whole\_Plot}_{hi} + \text{Split\_Plot}_{hij} + \varepsilon_{hijk}, \end{aligned}$$

where:

Y_{hijk} is the response for the k$^{\text{th}}$ split-split plot, within the j$^{\text{th}}$ split plot, within the i$^{\text{th}}$ whole plot, within the h$^{\text{th}}$ location, and

β_0 is the baseline mean for treatment combination of no Irrigation, control fungicide, and bean variety 1,

$\beta_1, \ldots, \beta_8$ are coefficients for the respective fixed effects,

$I_{\text{Irrigation=Yes},hi}$ is the indicator for the Irrigation treatment applied to whole plots,

$I_{\text{Fung.=1},hij}, \ldots, I_{\text{Fung.=4},hij}$ are indicators for type of Fungicide applied to split-plots,

$I_{\text{Bean=2},hijk}, \ldots, I_{\text{Bean=4},hijk}$ are indicators for variety of bean planted in the split-split-plots,

$\text{Location}_h \sim N(0, \sigma^2_{\text{Location}})$,

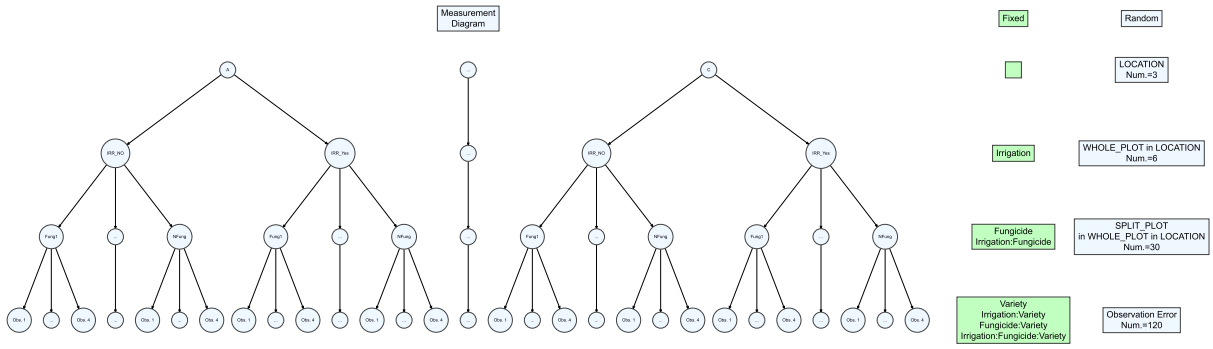


Figure 6: Model diagram for the simulated split-split-plot design with one measurement per split-split-plot within each location.

$$
\begin{aligned}
\text{Whole\_Plot}_{hi} &\sim N(0, \sigma^2_{\text{WP}}),\\
\text{Split\_Plot}_{hij} &\sim N(0, \sigma^2_{\text{SP}}),\\
\varepsilon_{hijk} &\sim N(0, \sigma^2),\\
&\quad \text{Location}_h, \text{Whole\_Plot}_{hi}, \text{Split\_Plot}_{hij}, \text{ and } \varepsilon_{hikj} \text{ independent},\\
h &= 1, 2, 3 \text{ levels of the Location random effect},\\
i &= 1, 2 \text{ levels of the Whole\_Plot random effect},\\
j &= 1, \ldots, 5 \text{ levels of Split\_Plot random effect},\\
k &= 1, \ldots, 4 \text{ observations (levels of Split\_Split\_Plot}\\
&\quad \text{random effect), and}\\
n &= h * i * j * k = 120 \text{ total observations.}
\end{aligned}
$$

```
library(lme4) # For lmer() function
library(modeldiagramR) # For data and model_diagram() function
data("SSPD3")
lmer_ssp <- lmer(RESP ~ Irrigation*Fungicide*Variety +
                         (1|LOCATION/WHOLE_PLOT/SPLIT_PLOT),
                 data=SSPD3)
# Figure 6
model_diagram(lmer_ssp, # model from lmer
              width = 800, # width of image should be 800 pixels
              height = 400, # height of image should be 400 pixels
              orientation = "horizontal", # orient diagram horizontally
              shiftFixed = 2, # shift column of fixed effect labels right 2 units
              shiftRandom = 5, # shift column of random effect labels right 5
              scaleFontSize = 2) # scale text in boxes 2 times bigger
```

While the yield response is simulated for the split-split-plot treatment, we can visualize the model structure of this complicated design and the information available in the treatments applied to each whole-plot and smaller units. This sort of exposition of the model structure is invaluable for understanding the model and explaining it to various audiences, both in instructional and research venues. For this diagram in Figure 6, optional parameters are used to orient

the diagram horizontally, nudge the columns of label boxes to prevent overlapping, increase the size of the font for the text in the label boxes to account for the scaling of the large diagram, and specification of the dimensions of the image are provided. This allows the user to select the best orientation for the diagram without an overwhelming number of parameters that would need adjustment.

5 Discussion

With only the output of a linear mixed model, or generalized linear mixed model, we can draw a diagram representing the information about the model structure informed by the data. Typically the focus on model output is on the estimated slopes and p-values with some discussion of variance estimates and model diagnostics. We propose that visualizing the model structure is an important model diagnostic step that should be included with all mixed model analyses. In addition, including these diagrams in manuscripts would assist in describing the random and fixed effects, making the model structure more accessible to readers less versed in hierarchical mixed effects models. It can also play an important role in helping students and researchers when learning about mixed models.

Future extensions of the `model_diagram()` function will explore work with non-nested random effect notation (*i.e.*, "`(1|RandomEffect1) + (1|RandomEffect2)`") in `mer` models to detect nested or non-nested random effects, draw diagrams for all nested random effects regardless of notation, visualize models with five or more hierarchical levels, and add compatibility with other mixed model functions in R (for example, *glmmTMB* (Brooks et al., 2017) or `aov` model objects). Eventually, we would like to be able to detect and visualize non-nested models which may require expanding the diagram into a third dimension and further explorations into how to best align fixed effects in studies that are not hierarchical.

Just as there are many ways for people to write code, there are alternative specifications for hierarchical random effects, and numerous ways of writing model equations. We have chosen a regression model notation, similar to that of the output from *LearnVizLMM*'s `extract_equation()` function with indicator functions for the categorical groups, number of levels for each group, and with informative names for the random effects, but with informative subscripts similar to Zuur et al. (2009). Rather than replicating this code, instead we recommend using *LearnVizLMM* for their equation generating function as a starting point for writing out models.

As noted previously but not demonstrated in any of the examples shown, the model diagram works for generalized linear mixed models fit using the *lme4* package (`glmerMod` objects). However, for generalized models with a non-Gaussian link function, the lowest level of the model diagram is no longer the "Observation Error". In this case it is really the observation level since there is no random error in most GLMMs. While we make some simplifying assumptions to transform most nested GLMMs from *lme4* into a `lme` model object for visualization, we do address the lack of random error by adjusting the label for the lowest level of the diagram to say "Observation Level" instead of "Observation Error".

Model diagrams are an important way to communicate the model being tested that goes beyond just an ANOVA table, a point estimate, p-value, or even a confidence interval. Many model visualizations focus on the fixed effects as the inference is typically regarding the treatments or the predictors, but it is just as important to know and understand the structure of the random effects. This is needed to be able to understand the level of inference that can be made as well as the correlations between the observations. While our function and R package

still have room for improvement, we hope that it is a visualization tool that can be useful for anyone fitting mixed models on their data.

Supplementary Material

The supplementary material is an R script file that contains all the code used to create the figures and model diagrams from this manuscript.

Acknowledgments

We would like to acknowledge students in upper level statistics courses at Montana State University that have provided feedback on the model diagrams as they were developed and the 2025 Symposium on Data Science and Statistics for the opportunity to present this material. Additionally, we would like to thank the reviewers for their time and thoughtful feedback, and The Writing Center at Montana State University for their assistance with the revisions.

Funding

This work was supported by the National Institute of Arthritis and Musculoskeletal and Skin Diseases of the National Institutes of Health under Award Number 1R01AR081489 and the National Institute of General Medical Sciences of the National Institutes of Health under Award Number P20GM103474. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

Bache SM, Wickham H (2022). magrittr: A Forward-Pipe Operator for R. R package version 2.0.3.

Bates D, Mächler M, Bolker B, Walker S (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1): 1–48. https://doi.org/10.18637/jss.v067.i01

Brooks ME, Kristensen K, van Benthem KJ, Magnusson A, Berg CW, ... Bolker BM (2017). glmmTMB balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. *The R Journal*, 9(2): 378–400. https://doi.org/10.32614/RJ-2017-066

Gelman A, Hill J (2021). *Data Analysis Using Regression and Multilevel/Hierarchical Models Analytical Methods for Social Research*. Cambridge Univ. Press, Cambridge, 23rd printing edition.

Henry L, Wickham H (2024). tidyselect: Select from a Set of Strings. R package version 1.2.1.

Iannone R (2016). *DiagrammeRsvg: Export DiagrammeR Graphviz Graphs as SVG*. R package version 0.1.

Iannone R, Roy O (2024). DiagrammeR: Graph/Network Visualization. R package version 1.0.11.

Koneswarakantha B (2023). easyalluvial: Generate Alluvial Plots with a Single Line of Code. R package version 0.3.2.

Kuznetsova A, Brockhoff PB, Christensen RHB (2017). lmerTest package: Tests in linear mixed effects models. *Journal of Statistical Software*, 82(13): 1–26. https://doi.org/10.18637/jss.v082.i13

Luetkemeier MJ, Hanisko JM, Aho KM (2017). Skin tattoos alter sweat rate and Na+ concentration. *Medicine & Science in Sports & Exercise*, 49(7): 1432–1436. https://doi.org/10.1249/MSS.0000000000001244

Murillo D, Gezan S (2024). FielDHub: A Shiny App for Design of Experiments in Life Sciences. R package version 1.4.2.

Müller K, Wickham H (2023). tibble: Simple Data Frames. R package version 3.2.1.

Pinheiro J, Bates D, R Core Team (2024). nlme: Linear and Nonlinear Mixed Effects Models. R package version 3.1-166.

Pinheiro JC, Bates DM (2000). *Mixed-Effects Models in S and S-PLUS*. Springer, New York.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ramsey FL, Schafer DW (2013). *The Statistical Sleuth: A Course in Methods of Data Analysis*. Brooks/Cole, Cengage Learning, Boston, 3rd edition. OCLC: 794592462.

Shönbrodt F (2014). Mixed Models in R. Ludwig-Maximilians-Universität München. https://www.personality-project.org/r/tutorials/summerschool.14/MLM_Schoenbrodt.pdf.

Tennekes M, Jonge ED, Daas PJH (2021). Visualizing and Inspecting Large Datasets with Tableplots. *Journal of Data Science* 11(1): 43–58. Publisher: School of Statistics, Renmin University of China.

Warnes GR, Bolker B, Lumley T, Magnusson A, Venables B, ..., Moeller S (2023). gtools: Various R Programming Tools. R package version 3.9.5.

Wickham H (2023a). forcats: Tools for Working with Categorical Variables (Factors). R package version 1.0.0.

Wickham H (2023b). stringr: Simple, Consistent Wrappers for Common String Operations. R package version 1.5.1.

Wickham H, François R, Henry L, Müller K, Vaughan D (2023). dplyr: A Grammar of Data Manipulation. R package version 1.1.4.

Wickham H, Vaughan D, Girlich M (2024). tidyr: Tidy Messy Data. R package version 1.3.1.

Zavez K, Harel O (2024). LearnVizLMM: Learning and Communicating Linear Mixed Models Without Data. R package version 1.0.0.

Zuur AF, Ieno EN, Walker N, Saveliev AA, Smith GM (2009). *Mixed Effects Models and Extensions in Ecology with R. Statistics for Biology and Health*. Springer, New York.