Detecting Treatment-Subgroup Interactions in Clustered Data with Generalized Linear Mixed-Effects Model Trees

M. Fokkema[1], N. Smits[2], A. Zeileis[3], T. Hothorn[4], H. Kelderman[5]
[1]Universiteit Leiden, [2]Universiteit van Amsterdam, [3]Universität Innsbruck, [4]Universität Zürich, [5]Universiteit Leiden and Vrije Universiteit, Amsterdam

Abstract

Identification of subgroups of patients for which treatment A is more effective than treatment B, and vice versa, is of key importance to the development of personalized medicine. Tree-based algorithms are helpful tools for the detection of such interactions, but none of the available algorithms allow for taking into account clustered or nested dataset structures, which are particularly common in psychological research. Therefore, we propose the generalized linear mixed-effects model tree (GLMM tree) algorithm, which allows for the detection of treatment-subgroup interactions, while accounting for the clustered structure of a dataset. The algorithm uses model-based recursive partitioning to detect treatment-subgroup interactions, and a GLMM to estimate the random-effects parameters. In a simulation study, GLMM trees show higher accuracy in recovering treatment-subgroup interactions, higher predictive accuracy, and lower Type-II error rates than GLM-based recursive partitioning and mixed-effects regression trees. Also, GLMM tree show somewhat higher predictive accuracy than linear mixed-effects models with pre-specified interaction effect, on average. We illustrate the application of GLMM trees on an individual patient-level data meta-analysis on treatments for depression. We conclude that GLMM trees are a promising exploratory tool for the detection of treatment-subgroup interactions in clustered datasets. An implementation is available in the R package `glmertree` and a short tutorial demonstrating how to fit GLMM trees in practice is provided in the accompanying vignette.

The authors would like to thank Prof. Pim Cuijpers, Prof. Jeanne Miranda, Dr. Boadie Dunlop, Prof. Rob DeRubeis, Prof. Zindel Segal, Dr. Sona Dimidjian, Prof. Steve Hollon and Erica Weitz for granting access to the dataset for the application. The work for this paper was partially done while MF, AZ and TH were visiting the Institute for Mathematical Sciences, National University of Singapore in 2014. The visit was supported by the Institute.

Keywords: model-based recursive partitioning, treatment-subgroup interactions, random effects, generalized linear mixed-effects model, classification and regression trees

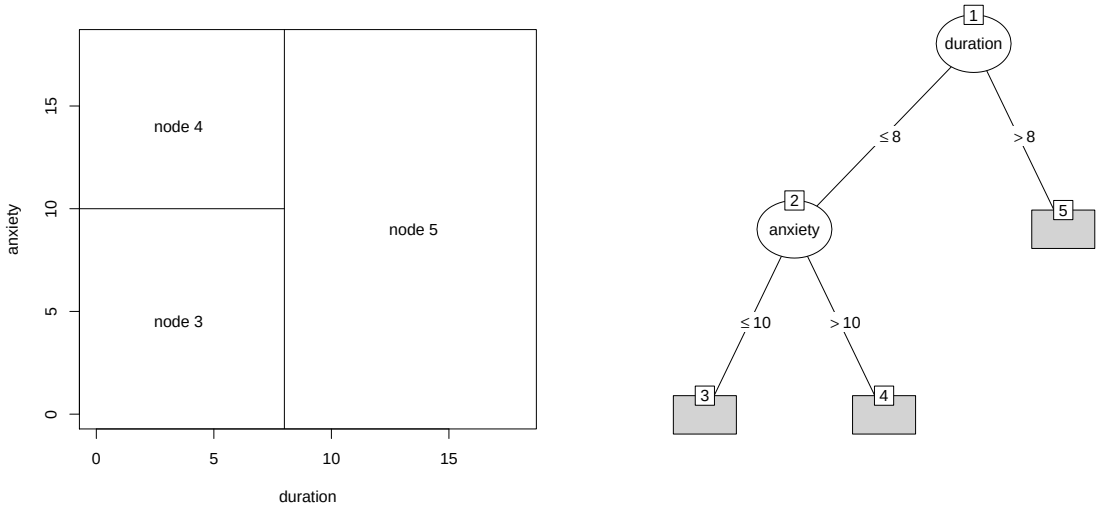
Introduction

In research on the efficacy of treatments for somatic and psychological disorders, the one-size-fits-all paradigm is slowly losing ground, and personalized or stratified medicine is becoming increasingly important. Stratified medicine presents the challenge of discovering which patients respond best to which treatments. This can be referred to as the detection of treatment-subgroup interactions (e.g., Doove, Dusseldorp, Van Deun, & Van Mechelen, 2014). Often, treatment-subgroup interactions are studied using linear models, such as factorial analysis of variance techniques, in which potential moderators have to be specified a-priori, have to be checked one at a time, and continuous moderator variables have to be discretized. This may hamper identification of which treatment works best for whom, especially when there are no a-priori hypotheses about treatment-subgroup interactions. As noted by Kraemer, Frank, and Kupfer (2006), there is a need for methods that generate instead of test such hypotheses.

Tree-based methods are such hypothesis-generating methods. Tree-based methods, also known as recursive partitioning methods, recursively split the space spanned by the predictor variables into rectangular regions, containing observations that are increasingly similar with respect to the outcome. Several tree-based methods take the mean or majority class of a single dependent variable as the outcome, one of the earliest and most well-know examples being the classification and regression tree (CART) approach of Breiman, Friedman, Olshen, and Stone (1984). Other tree-based methods take the estimated parameters of a more complex model, of which the RECPAM approach of Ciampi (1991) is the earliest example.

Due to the recursive nature of the splitting, the rectangular regions of the partition can be graphically depicted as nodes in a decision tree, as shown in the artificial example in Figure 1. The partition in Figure 1 is rather simple, based on the values of two predictor variables: duration and anxiety. The resulting tree has a depth of two, as the longest path travels along two splits. Each of the splits in the tree is defined by a splitting variable and value. The first split in the tree separates the observations into two subgroups, based on the splitting variable duration and a splitting value of 8, yielding two rectangular regions, represented by node 2 and node 3. As the observations in node 2 are not further split, node 2 is a terminal node. Node 3 is an inner node, as the observations in this node are further split into nodes 4 and 5, based on the anxiety variable.

If the partition in Figure 1 would be used for prediction of a new observation, the new observation would be assigned to one of the terminal nodes according to its values



(a) Rectangular areas

(b) Decision tree

Figure 1. : Example of a recursive partition. In the left panel, the partition is depicted as a set of rectangular areas. In the right panel, the same partition is depicted as a decision tree.

on the splitting variables. The prediction is then determined by the distribution of the training observations within the terminal node. For example, the prediction may be the node-specific mean of a single continuous variable. In the current paper, we focus on trees where the terminal nodes consist of a linear (LM) or generalized linear model (GLM), in which case the predicted value for a new observation is determined by the node-specific parameter estimates of the (G)LM, while also adjusting for random effects.

Tree-based methods are particularly useful for exploratory purposes, because they can handle many potential predictor variables at once and can automatically detect (higher order) interactions between predictor variables (Strobl, Malley, & Tutz, 2009). As such, they are preeminently suited to the detection of treatment-subgroup interactions. Several tree-based algorithms for the detection of treatment-subgroup interactions have already been developed (Dusseldorp, Doove, & Van Mechelen, 2016; Dusseldorp & Meulman, 2004; Su, Tsai, Wang, Nickerson, & Li, 2009; Foster, Taylor, & Ruberg, 2011; Lipkovich, Dmitrienko, Denne, & Enas, 2011; Zeileis, Hothorn, & Hornik, 2008; Seibold, Zeileis, & Hothorn, 2016; see Doove et al., 2014 for an overview). Also, Zhang, Tsiatis, Laber, and Davidian (2012) and Zhang, Tsiatis, Davidian, Zhang, and Laber (2012) have developed a flexible classification-based approach, allowing users to select from a range of statistical methods, including trees.

In many instances, researchers may want to detect treatment-subgroup interactions

in clustered or nested datasets. For example, in individual-level patient data meta-analyses, where datasets of multiple clinical trials on the same treatments are pooled. In such analyses, the nested or clustered structure of the dataset should be taken into account by including study-specific random effects in the model, prompting the need for a mixed-effects model (e.g., Cooper & Patall, 2009; Higgins, Whitehead, Turner, Omar, & Thompson, 2001). In linear models, ignoring the clustered structure may lead, for example, to biased inference due to underestimated standard errors (e.g., Bryk & Raudenbush, 1992). For tree-based methods, ignoring the clustered structure has been found to result in the detection of spurious subgroups and inaccurate predictor variable selection (e.g., Sela & Simonoff, 2012; Martin, 2015). However, none of the purely tree-based methods for treatment-subgroup interaction detection allow for taking into account the clustered structure of a dataset. Therefore, in the current paper, we present a tree-based algorithm which can be used for the detection of interactions and non-linearities in GLMM type models: generalized linear mixed-effects model trees, or GLMM trees.

The GLMM tree algorithm builds on model-based recursive partitioning (MOB, Zeileis et al., 2008), which offers a flexible framework for subgroup detection. For example, GLM-based MOB has been applied to detect treatment-subgroup interactions for the treatment of depression (Driessen et al., 2016) and amyotrophic lateral sclerosis (Seibold et al., 2016). In contrast to other purely tree-based methods (e.g., Zeileis et al., 2008; Su et al., 2009; Dusseldorp et al., 2016), GLMM tree allows for taking into account the clustered structure of datasets. In contrast to previously suggested regression trees with random effects (e.g., Hajjem, Bellavance, & Larocque, 2011; Sela & Simonoff, 2012), GLMM trees allow for treatment effect estimation, with continuous as well as non-continuous response variables.

The remainder of this paper is structured into four sections: In the first section, we introduce the GLMM tree algorithm using an artificial motivating dataset with treatment-subgroup interactions. The second section is a short tutorial on how to fit GLMM trees using the R package `glmertree` *(still needs to be adapted: vignette)*. In the third section, README we compare the performance of GLMM trees with that of three other methods: MOB trees without random effects, mixed-effects regression trees (MERTs) and linear mixed-effects models with pre-specified interactions. In the fourth section, we apply a GLMM tree to an existing dataset of a patient-level meta-analysis on the effects of psycho- and pharmacotherapy for depression. In the fith and last section we summarize the results and discuss limitations and directions for future research.

GLMM tree algorithm

Artificial motivating dataset

We will use an artificial motivating dataset with treatment-subgroup interactions to introduce the GLMM tree algorithm. The R code used to generate the example dataset is provided in the Appendix *(I would put this into the appendix of the vignette instead.).* README

The dataset consists of a set of observations on $N = 150$ patients, who were randomly assigned to one of two treatment alternatives (Treatment 1 or Treatment 2). The treatment outcome is represented by the variable depression, quantifying post-treatment depressive symptomatology. The potential moderator variables are duration, age and anxiety. Duration reflects the number of months the patient has been suffering from depression prior to treatment, age reflects patients' age in years at the start of treatment and anxiety reflects patients' total scores on an anxiety inventory administered before treatment. Finally, each patient was part of one of ten clusters, each having a different value for the random intercept, uncorrelated with the partitioning variables.

The outcome variable was generated such that there are three subgroups with differential treatment effectiveness, corresponding to the terminal nodes in Figure 1: For the first subgroup of patients (node 3) with short duration (≤ 8) months of depressive symptoms prior to treatment and low anxiety scores (≤ 10), Treatment 1 leads to lower post-treatment depression than in Treatment 2. For the second subgroup of patients (node 4) with short duration but high anxiety scores (> 10) post-treatment depression is about equal in both treatment conditions. For the third subgroup of patients (node 5) with long duration (> 8 months) Treatment 2 leads to lower post-treatment depression than Treatment 1. Thus, duration and anxiety are true partitioning or moderator variables, whereas age is not. Anticipating the final results of our analyses, the treatment-subgroup interactions are depicted in Figure 4, which shows the GLMM tree that accurately recovered the treatment-subgroup interactions. *(I've incorporated a characterization of the groups in terms of duration and anxiety here. But probably a better description of these variables is needed earlier. Possibly in a table with some summary statistics, e.g., mean/sd/range. Finally, the description in the manual Rd pages of the package should also be improved.)* README

Model-based recursive partitioning

The rationale behind MOB is that a single global GLM (or other parametric model) may not describe the data well, and when additional covariates are available it may be possible to partition the dataset with respect to these covariates, and find better-fitting models in each cell of the partition. For example, to assess the effect of treatment, we may first fit a global GLM where the treatment indicator has the same effect/coefficient on the outcome for all observations. Subsequently, the data may be partitioned recursively with respect to

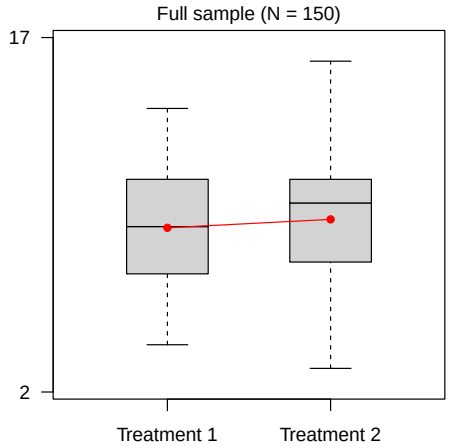


Figure 2. : Example of a global GLM for treatment outcomes, based on the artificial motivating dataset ($N = 150$). The y-axis represent treatment outcome (post-treatment depression). The dot for Treatment 1 represents the first and the slope of the regression line represents the second element of β.

other covariates, leading to separate models with different treatment effects/coefficients in each subsample. *(Expanded on the treatment GLM MOB explanation to give more context and motivation.)* README

More formally, in a single global GLM the expectation μ_i of outcome y_i given the treatment regressor x_i is modeled through a linear predictor and suitable link function:

$$E[y_i|x_i] = \mu_i, \quad (1)$$

$$g(\mu_i) = x_i^\top \beta, \quad (2)$$

where $x_i^\top \beta$ is the linear predictor for observation i and g is the link function. β is a vector of fixed-effects regression coefficients. For simplicity, in the current paper we focus on two treatment groups and no further covariates in the GLM, so that in our illustrations x_i and β both have length 2. For the continuous response variable in the motivating data set, we employ the identity link function and assume a normal distribution for the error (denoted by $\epsilon_i = y_i - \mu_i$) with mean zero and variance σ_ϵ^2. Thus, the first element of β then corresponds to the mean of the linear predictor in the first treatment group and the second element corresponds to the mean difference in the linear predictor between the first and second treatment groups. However, the model can easily accommodate additional treatment conditions and covariates, as well as binary or count/Poisson outcome variables.

Obviously, such a simple, global GLM will not fit the data well, especially in the

presence of moderators. For expository purposes, however, we take it as a starting point to illustrate MOB. The global GLM fitted to the motivating example dataset is depicted in Figure 2. As the boxplots show, there is little difference between the global effects of the two treatments and there is considerable residual variance.

The MOB algorithm can be used to partition the dataset using additional covariates and find better-fitting local models. To this end, the MOB algorithm tests for parameter stability over a set of auxiliary covariates, also called *partitioning variables*. When the partitioning is based on a GLM, instabilities are differences in $\hat{\beta}$ across partitions of the dataset, which are defined by one or more auxiliary covariates not included in the linear predictor. To find these partitions, the MOB algorithm cycles iteratively through the following steps (Zeileis et al., 2008): (1) fit the parametric model to the dataset, (2) statistically test for parameter instability over a set of partitioning variables, (3) if there is some overall parameter instability, split the dataset with respect to the variable associated with the highest instability, (4) repeat the procedure in each of the resulting subgroups.

In step (2) a test statistic quantifying parameter instability is calculated for every potential partitioning variable. As the distribution of these test statistics under the null hypothesis of parameter stability is known, a p-value for every partitioning variable can be calculated. Note that a more in-depth discussion of the parameter stability tests is beyond the scope of this paper, but can be found in Zeileis and Hornik (2007) and Zeileis et al. (2008).

If at least one of the partitioning variables yields a p-value below the pre-specified significance level α, the dataset is partitioned into two subsets in step (3). This partition is created using U_{k^*} *(the U variables have not been introduced, no symbol has been used for the partitioning variables so far)*, the partitioning variable with the minimal p-value in step (2). The split point for U_{k^*} is selected by taking the value that minimizes the sum of the values of the objective function in both partitions. In other words, for every possible split point, the objective function is minimized separately in the two resulting subsets of the data; the split point yielding the minimum sum of the objective functions is selected. In step (4), steps (1) through (3) are repeated in each partition, until the null hypothesis of parameter stability can no longer be rejected (or the subsets become too small). README

The partition resulting from application of MOB can be depicted as a decision tree. If the partitioning is based on a GLM, the result is a GLM tree, with a local fixed-effects regression model in every j-th ($j = 1, \ldots, J$) terminal node or subgroup:

$$g(\mu_{ij}) = x_i^\top \beta_j \tag{3}$$

To illustrate, we fitted a GLM tree on the artificial motivating dataset. In addition to the treatment indicator and treatment outcome used to fit the earlier GLM, we specified the anxiety, duration and age variables as potential partitioning variables. Figure 3 shows

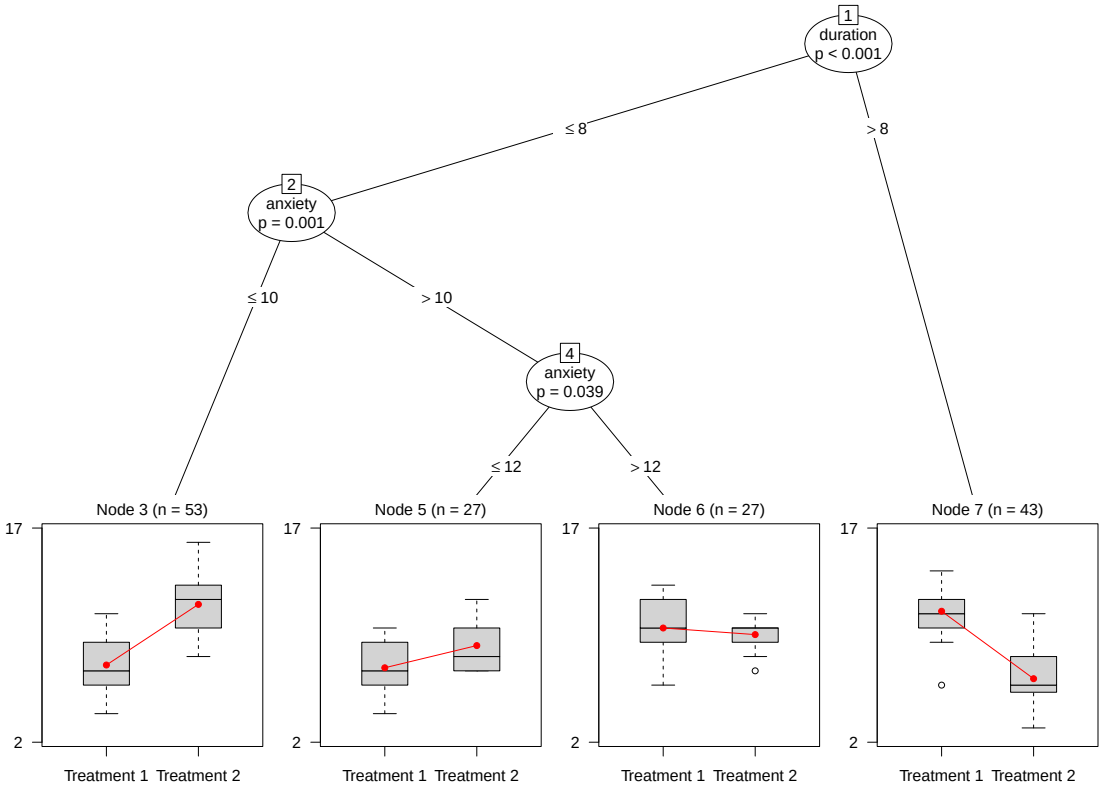


Figure 3. : GLM tree grown on the artificial motivating dataset. Three additional covariates (pre-treatment anxiety, duration and age) were used as potential splitting variables. The y-axes in the terminal nodes represent the treatment outcome (post-treatment depression severity).

the resulting GLM tree. MOB partitioned the observations into four subgroups, each with a different estimate β_j. Age was correctly not identified as a partitioning variable and the left- and rightmost nodes are in accordance with the true treatment-subgroup interactions described above. However, the two nodes in the middle have an unnecessary split and thus do not represent true subgroups, possibly due to the dependence of observations within clusters not being taken into account.

Including random effects

For datasets containing observations from multiple clusters (e.g., trials or research centers), application of a mixed-effects model would be more appropriate. The GLM in Equation 2 is then extended to include cluster-specific, or random effects:

$$g(\mu_i) = x_i^\top \beta + z_i^\top b \tag{4}$$

For a random-intercept only model, z_i is a unit vector of length M, of which the m-th element takes a value of 1, and all other elements take a value of 0; m ($m = 1, \ldots, M$) denotes the cluster which observation i is part of. Further, b is a random vector of length M, each m-th element corresponding to the random intercept for cluster m. For simplicity, we employ a cluster-specific intercept only, but further random effects can easily be included in z_i. Furthermore, within the GLMM it is assumed that b is normally distributed, with mean zero and variance σ_b^2. Also, the errors ϵ are assumed to follow a normal distribution, with constant variance across clusters. The parameters of the GLMM can be estimated with, for example, maximum likelihood (ML) and restricted ML (REML).

Although the random-effects part of the GLMM in Equation 4 accounts for the nested structure of the dataset, the global fixed-effects part $x_i^\top \beta$ may not describe the data well. Therefore, we propose the GLMM tree model, in which the fixed-effects part may be partitioned as in Equation 3 while still adjusting for random effects:

$$g(\mu_i) = x_i^\top \beta_j + z_i^\top b \tag{5}$$

To estimate the parameters of this model, we take an approach similar to that of the mixed-effects regression tree (MERT) approach of Hajjem et al. (2011) and Sela and Simonoff (2012). In the MERT approach, the fixed-effects part of a GLMM is replaced by a CART tree with constant fits in the nodes, and the random-effects parameters are estimated as usual. To estimate a MERT, an iterative approach is taken, alternating between (1) assuming random effects known, allowing for estimation of the CART tree, and (2) assuming the CART tree known, allowing for estimation of the random-effects parameters.

For estimating GLMM trees, we take this approach two steps further: (1) Instead of a CART tree with constant fits to estimate the fixed-effects part of the GLMM, we use a GLM tree. This allows not only for detection of differences in intercepts across terminal nodes, but also for detection of differences in slopes such as treatment effects. (2) By using generalized linear (mixed) models, the response may also be a binary or count variable instead of a continuous variable. The GLMM tree algorithm takes the following steps to estimate the model in Equation 5:

Step 0: Initialize by setting r and all values $\hat{b}_{(r)}$ to 0.

Step 1: Set $r = r + 1$. Estimate a GLM tree using $z_i^\top \hat{b}_{(r-1)}$ as an offset.

Step 2: Fit the mixed-effects model $g(\mu_i) = x_i^\top \beta_j + z_i^\top b$ with subgroups $j(r)$ from the GLM tree estimated in Step 1. Extract posterior predictions $\hat{b}_{(r)}$ from the estimated model.

Step 3: Repeat Steps 1 and 2 until convergence.

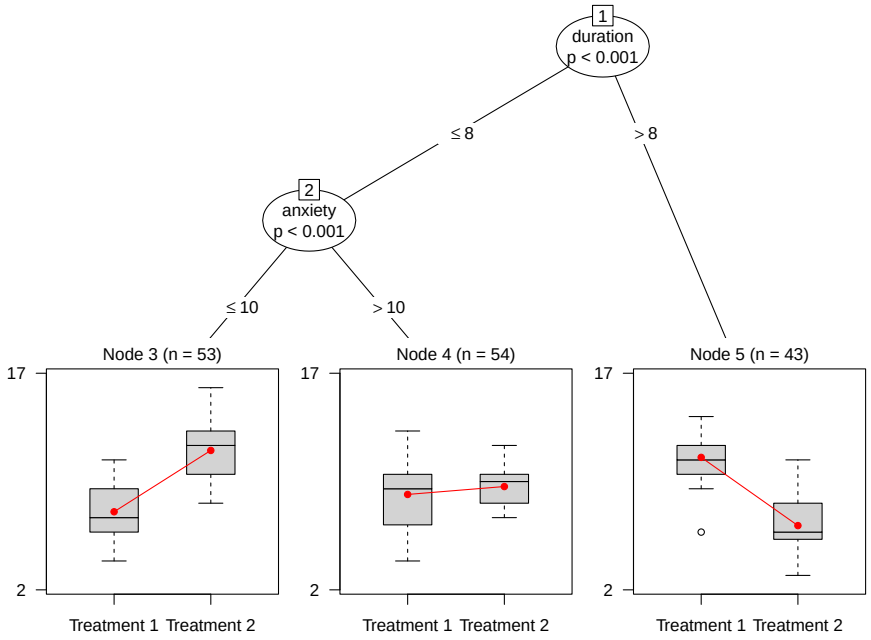


Figure 4. : GLMM tree of the motivating example dataset. Three covariates (anxiety questionnaire score, duration of depressive symptoms at baseline in months) and age were used as potential splitting variables, and the clustering structure was taken into account by estimating random intercepts.

The algorithm initializes by setting b to 0, since the random effects are initially unknown. In every iteration, the GLM tree is re-estimated in step (1) and the fixed- and random-effects parameters are re-estimated in step (2). Note that the random-effects part of the model is not partitioned, but estimated globally. Only the fixed effects are estimated locally, within the cells of the partition. Convergence of the algorithm is monitored by computing the log-likelihood criterion of the mixed-effects model in Equation 5. Typically, this converges if the tree does not change from one iteration to the next.

In Figure 4, the result of applying the GLMM tree algorithm to the motivating dataset is presented. In addition to the treatment indicator, treatment outcome and the potential partitioning variables, the GLMM tree has also included a random intercept term with respect to the cluster indicator. As a result, the dependence between observations is taken into accoutn, the true treatment subgroups have been recovered and the spurious split involving the anxiety variable no longer appears in the tree.

Tutorial

README

(This tutorial should be turned into a vignette and then the R output, graphics, etc. should be shown as well.)

We implemented the GLMM tree algorithm in the R package `glmertree` (version 0.1-0; Fokkema & Zeileis, 2016; available from R-Forge). The package makes use of the `partykit` package (Hothorn & Zeileis, 2015) and `lme4` package (Bates, Mächler, Bolker, & Walker, 2015) to estimate trees and mixed-effects models, respectively. The latest version of the `glmertree` package can be downloaded, installed and loaded as follows:

```
install.packages("glmertree", repos="http://R-Forge.R-project.org")
library(glmertree)
```

Package documentation and examples can be accessed as follows:

```
?glmertree
```

The main functions in the `glmertree` package are `lmertree()`, for continuous outcome variables, and `glmertree()`, for binary or count outcome variables. Both functions require the user to specify at least two arguments: `formula` and `data`. The data we will be using is the motivating example dataset described in the previous section. This dataset is included in the `partykit` package and can be loaded as follows:

```
data("a_m_data", package = "glmertree")
summary(a_m_data)
```

The model formula to be specified consists of a left- and right hand side. The left hand side of the model formula (preceding the tilde symbol) specifies the outcome variable. The right hand side consists of three parts, separated by vertical bars: The first part specifies the predictor variable(s) of the (generalized) linear model, the second part specifies the random effects and the third part specifies the potential partitioning variables:

```
exampleglmmtree <- lmertree(depression ~ treatment | cluster | age + duration +
                            anxiety, data = a_m_data)
```

Note that in the example above, the partitioning variables are continuous, but (ordered) categorical partitioning variables may also be specified. Also, we specified only a single variable in the random-effects part, resulting in estimation of a random intercept with respect to `cluster`. More complex random effects can also be specified: for example, specifying the random-effects part as `... | (1 + age | cluster) | ...` would yield a model with a random intercept as well as a random slope for age with respect to cluster. The brackets are necessary to protect the vertical bars in the random effects formulation.

Now, we can use the `plot` method to recreate the tree depicted in Figure 4:

```
plot(exampleglmmtree)
```

In every inner node of the tree, the splitting variable and corresponding p-value from the parameter stability test is reported. To control for multiple testing, the p-values are Bonferroni corrected, by default. This can be turned off by adding `bonferroni = FALSE` to the function call, yielding a less conservative criterion for the parameter stability tests, but note that this will increase the likelihood of overfitting. The significance level α equals .05 by default, but a different value, say for example .01, can be specified by including `alpha = .01` in the function call.

The predictions of the random effects can be plotted by adding `plotranef = TRUE` to the function call: *(As already indicated by e-mail, plotranef is an awkward name and should be improved, e.g., to which.)* README

```
plot(exampleglmmtree, plotranef = TRUE)
```

To obtain numerical results, `print`, `coef` and `ranef` methods can be used:

```
print(exampleglmmtree)
coef(exampleglmmtree)
ranef(exampleglmmtree)
```

To obtain predicted values, the `predict` method can be used:

```
predict(exampleglmmtree, newdata = a_m_data[1:10,])
```

When `newdata` is not specified, predictions for the training observations are returned, by default. Random effects can be excluded from the predictions by adding `re.form = NA`. This is useful, for example, when `newdata` is specified, but the new observations do not have a cluster indicator or are from new clusters:

```
predict(exampleglmmtree, newdata = a_m_data[1:10, -4], re.form = NA)
```

Residuals of the fitted GLMM tree can be obtained with the `residuals` method. This can be useful for assessing potential misspecification of the model (e.g., heteroscedasticity):

```
resids <- residuals(exampleglmmtree)
preds <- predict(exampleglmmtree)
plot(a_m_data$cluster, resids)
scatter.smooth(preds, resids)
```

The first plot did not indicate substantial variation in error variances across levels of the random effects. The second plot of fitted values against residuals also did not reveal a pattern indicating model misspecification.

Simulation-based evaluation

To assess the performance of GLMM trees, we carried out three simulation studies: In Study I we assessed and compared the accuracy of GLMM trees, linear-model based MOB and mixed-effects regression trees (MERTs) in datasets with treatment-subgroup interactions. In Study II, we assessed and compared the Type-I error of GLMM trees and linear-model based MOB in datasets without treatment-subgroup interactions. In Study III, we assessed and compared the performance of GLMM trees and linear mixed-effects models (LMMs) with pre-specified interactions in datasets with piecewise and continuous interactions. As the outcome variable was continuous in all simulated dataset, the GLMM tree algorithm and the trees resulting from its application will be referred to as LMM tree(s).

General simulation design

In all simulation studies, the following data-generating parameters were varied:

1. Sample size: $N = 200$, $N = 500$, $N = 1000$.
2. Number of potential partitioning covariates U_1 through U_K: $K = 5$ and $K = 15$.
3. Intercorrelation between the potential partitioning covariates U_1 through U_K: $\rho_{U_k,U_{k'}} = 0.0$, $\rho_{U_k,U_{k'}} = 0.3$.
4. Number of clusters: $M = 5$, $M = 10$, $M = 25$.
5. Population standard deviation (SD) of the normal distribution from which the cluster specific intercepts were drawn: $\sigma_b = 0$, $\sigma_b = 5$, $\sigma_b = 10$.
6. Intercorrelation between b and one of the U_k variables: b and all U_k covariates uncorrelated, b correlated with one of the U_k covariates ($r = .42$).

Following the approach of Dusseldorp and Van Mechelen (2014), all partitioning covariates U_1 through U_K were drawn from a multivariate normal distribution with means $\mu_{U1} = 10$, $\mu_{U2} = 30$, $\mu_{U4} = -40$, and $\mu_{U5} = 70$. Means for other potential partitioning covariates were drawn from a discrete uniform distribution on the interval $[-70, 70]$. All covariates U_1 through U_{15} had the same standard deviation: $\sigma_{Uk} = 10$.

To generate the cluster-specific intercepts, we partitioned the sample into M equally-sized clusters, conditional on one of the variables U_1 through U_5, producing the correlations in the sixth facet of the simulation design. For each cluster, a single value b_m was drawn from a normal distribution with mean 0 and the value of σ_b given by the fifth facet of the simulation design. If b was correlated with one of the potential partitioning variables, the correlated variable was randomly selected.

For every observation, we generated a binomial variable (with probability .5) as an indicator for treatment type. Random errors ϵ were drawn from a normal distribution with $\mu_\epsilon = 0$ and $\sigma_\epsilon = 5$. The value of the outcome variable y_i was calculated as the sum of the random intercept, (node-specific) fixed effects and the random error term.

Due to the large number of cells in the simulation design, the most important predictors of accuracy were determined by means of ANOVAs and/or GLMs. The most important predictors of accuracy where then assessed through graphical displays. The ANOVAs and GLMs included main effects of algorithm type and the parameters of the data-generating process, as well as first-order interactions between algorithm type and each of the data-generating parameters.

Software

R (R Core Team, 2016) was used for data generation and analyses. The `partykit` package (version 1.0-2; Hothorn & Zeileis, 2015) was employed for estimating LM trees, using the `lmtree` function. For estimation of LMM trees the `lmertree` function of the `glmertree` package (version 0.1-0; Fokkema & Zeileis, 2016; available from R-Forge) was used. The significance level α for the parameter instability tests was set to .05 for all trees, with a Bonferroni correction applied for multiple testing. The minimum number of observations per node in trees was set to 20 and maximum tree depth was set to three, thus limiting the number of terminal nodes to eight in every tree.

The `REEMtree` package (Sela & Simonoff, 2011) was employed for estimating MERTs, using default settings. For estimating LMMs the `lmer` function function from the `lme4` package (version 1.1-7; Bates et al., 2015) was employed, using restricted maximum likelihood (REML) estimation. The `lmerTest` package (version 2.0-32; Kuznetsova, Brockhoff, & Christensen, 2016) was used to assess statistical significance of fixed-effects predictors in LMMs in Study III. The `lmerTest` package calculates effective degrees of freedom and p-values based on Satterthwaite approximations.

Study I: Performance of LMM tree, LM tree and MERT in datasets with treatment-subgroup interactions

Method

Treatment-subgroup interaction design. For generating datasets with treatment-subgroup interactions, we used a design from Dusseldorp and Van Mechelen (2014) which is depicted in Figure 5. Figure 5 shows four subgroups, characterized by values of the partitioning variables U_2, and U_1 or U_5. Two of the subgroups have mean differences in treatment outcome, indicated by a non-zero value of β_{j1}, and two subgroups do not have mean differences in treatment outcome, indicated by a β_{j1} value of 0.

In this simulation design, some of the potential partitioning covariates are true partitioning covariates, the others are noise variable. Therefore, an extra level was added to the sixth facet of the *General simulation design*:

6. Intercorrelation between b and one of the U_k variables: b and all U_k covariates

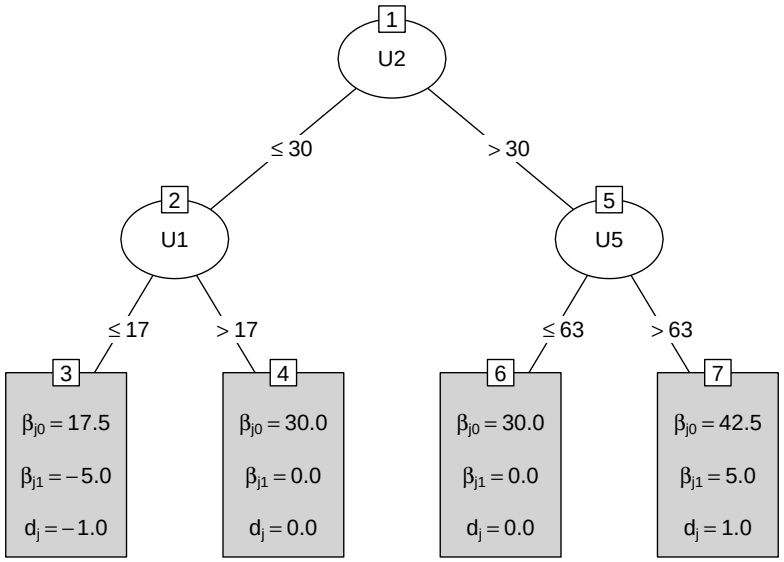


Figure 5. : Data-generating model for treatment-subgroup interactions. Parameter d_j denotes the node-specific standardized mean difference between the outcomes of Treatment 1 and 2 (i.e., $\beta_{j1}/\sigma_\epsilon$).

uncorrelated, b correlated with one of the true partitioning covariates (U_1, U_2 or U_5), b correlated with one of the noise variables (U_3 or U_4).

An additional facet was added to assess the effect of the magnitude of treatment-effect differences:

7. Two levels for the mean difference in treatment outcomes: The absolute value of the treatment-effect difference was varied to be $|\beta_{j1}| = 2.5$ (corresponding to a medium effect size, Cohen's $d = 0.5$; Cohen, 1992) and $|\beta_{j1}| = 5.0$ (corresponding to a large effect size; Cohen's $d = 1.0$).

For each cell of the design, 50 datasets were generated. In every dataset, the outcome variable was calculated as $y_i = x_i^\top \beta_j + z_i^\top b_m + \epsilon_i$.

Assessment of performance. Performance of the algorithms was assessed by means of tree size, tree accuracy and predictive accuracy. An accurately recovered tree was defined as a tree with (1) seven nodes in total, (2) the first split involving variable U_2 with a value of 30 ± 5, (3) the next split on the left involving variable U_1 with a value of 17 ± 5, and (4) the next split on the right involving variable U_5 with a value of 63 ± 5. The allowance of ± 5 equals plus or minus half the population SD of the partitioning variable (σ_{U_k}).

For MERT, the number of nodes and tree accuracy was not assessed, as the treatment-subgroup interaction design in Figure 5 corresponds to a large number of regression tree

structures, that would all be different but also correct. Therefore, performance of MERT was only assessed in terms of predictive accuracy.

Predictive accuracy of each method was assessed by calculating the correlation between true and predicted treatment-effect differences. To prevent overly optimistic estimates of predictive accuracy (Hastie, Tibshirani, & Friedman, 2009), predictive accuracy was assessed using test datasets. Test datasets were generated from the same population as training datasets, but test observations were not drawn from the same clusters as the training observations, but from 'new' clusters.

For MERT, predicted treatment-effect differences were obtained by fitting two MERTs on the training data: one using observations in the first treatment condition and one using observations in the second treatment condition. Predictions of treatment-effect differences for test observations were obtained by dropping test observations down both trees and taking the difference between the two predicted values. This approach was taken as it yielded higher predictive accuracy than the alternative of fitting a single MERT using treatment as a potential partitioning variable.

Results

Tree size. The average size of LMM trees was 7.15 nodes (SD = 0.61), whereas the average size of LM trees was 8.15 nodes (SD = 2.05), indicating that LM trees tend to involve more spurious splits than LMM trees. The effects of the most important predictors of tree size are depicted in Figure 6. The average size of LMM trees was close to the true tree size in all conditions. In the absence of random effects, this was also the case for LM trees. In the presence of random effects that are correlated to a (potential) partitioning variable, LM trees start to create spurious splits, especially with larger σ_b values. In the presence of random effects that are uncorrelated to the other variables in the model, LM trees lack power to detect treatment-subgroup interactions if sample size is small (i.e., $N = 200$). With larger sample sizes, LM trees showed about the true tree size, on average.

Accuracy of recovered trees. The estimated probability that a dataset was erroneously not partitioned (Type-II error) was 0 for both algorithms. For the first split, LMM trees selected the true partitioning variable (U_2) in all datasets, and LM trees in all but one datasets. The mean splitting value of the first split was 29.94 for LM as well as LMM trees, which is very close to the true splitting value of 30 (Figure 5).

Further splits were more accurately recovered by LMM trees yielding 90.40% accuracy for the full partition comparted to only 61.44% for LM trees. The effects of the four most important predictors of tree accuracy are depicted in Figure 7. In the absence of random effects, LM and LMM tree were about equally accurate. In the presence of random effects, LM trees were much less accurate than LMM trees when random effects were correlated

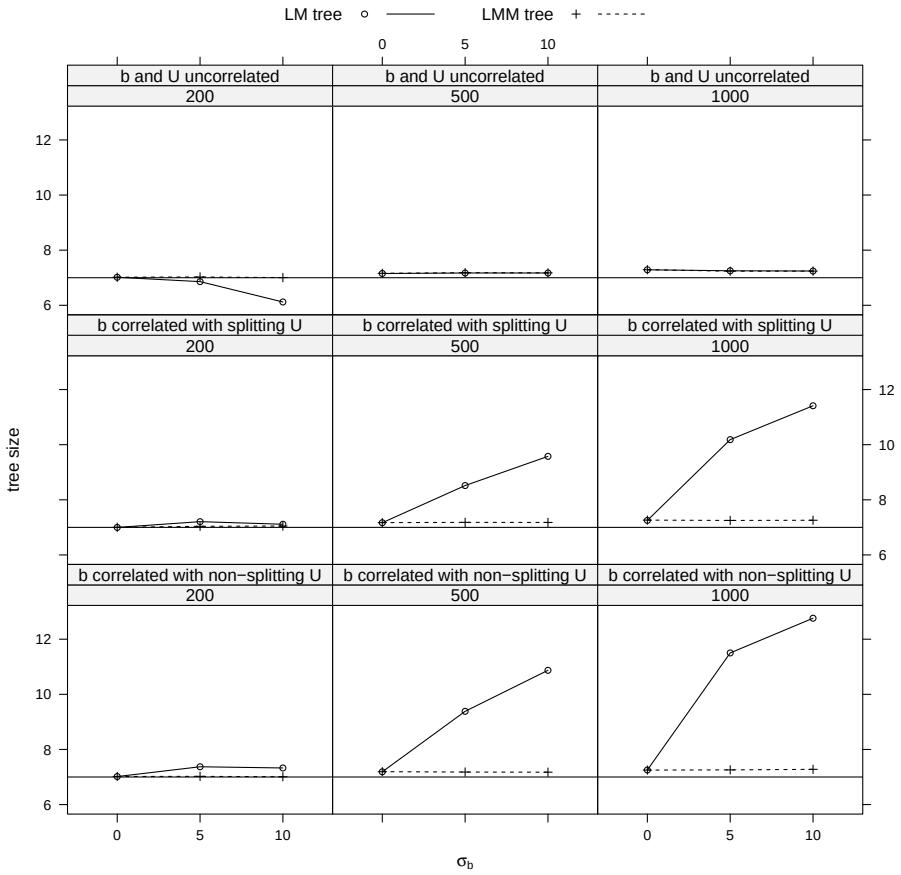


Figure 6. : Average tree size of LM and LMM trees. Tree size is defined as the total number of nodes in a tree. Rows represent the correlation between the random intercepts and one of the partitioning variables, columns represent sample size. The horizontal line in each graph indicates the 'true' tree size (the total number of nodes in the tree used for generating the interactions).

with a partitioning covariate. When random intercepts were not correlated with one of the U_k variables, LMM trees outperformed LM trees only when sample size was small (i.e., $N = 200$).

Predictive accuracy. The predicted treatment-effect differences of LMM tree show an average correlation of .93 (SD = .13) with the true differences. LM trees and MERT show lower accuracy, with an average correlations of .88 (SD = .19) and .75 (SD = .21), respectively. The most important predictors of predictive accuracy are depicted in Figure 8 *(Is it necessary to span the whole unit interval on the y-axis? Using 0.3 to 1.1 should be more than sufficient, I guess).* Performance of all three algorithms improves with increasing sample size and treatment-effect differences. Furthermore, LMM trees and MERT are not README

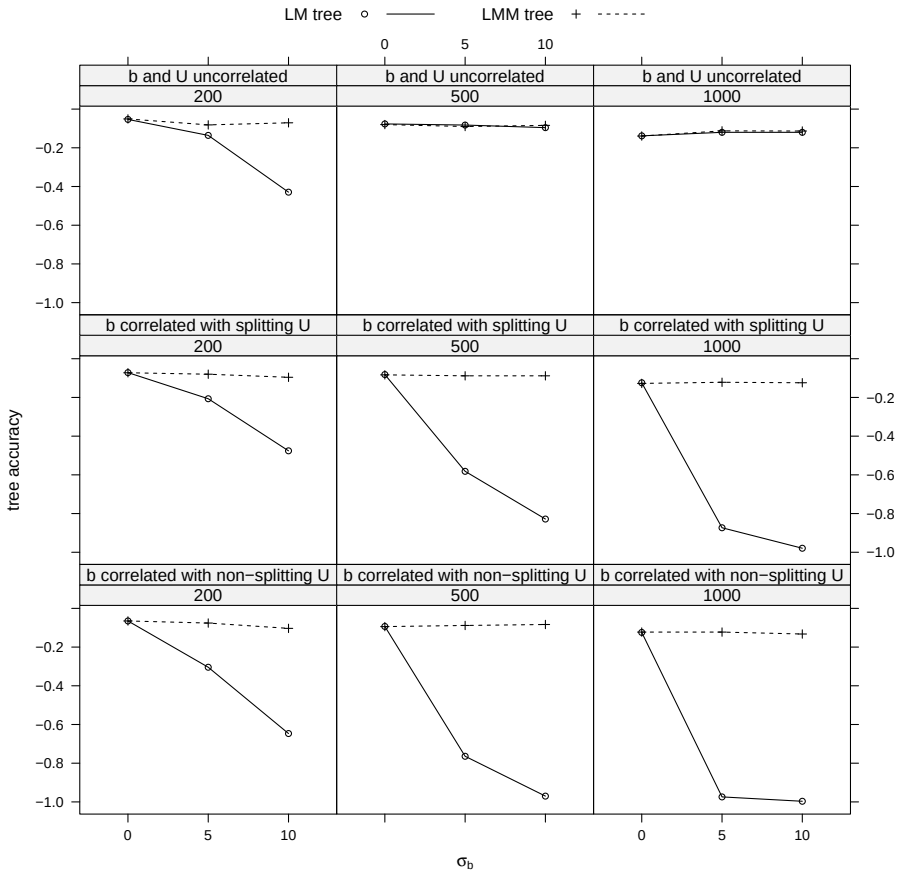


Figure 7. : Tree accuracy of LM and LMM tree. Tree accuracy is defined as the proportion of datasets in which the true tree was accurately recovered. Rows represent dependence between random effects (b) and one of the partitioning variables U_k, columns represent sample size.

much affected by the presence and magnitude of random effects in the data. LMM trees perform most accurately in most conditions and are never outperformed by the other methods. MERT performs least accurately in most conditions and never outperforms the other methods. The differences in accuracy become less pronounced with larger sample and effect sizes.

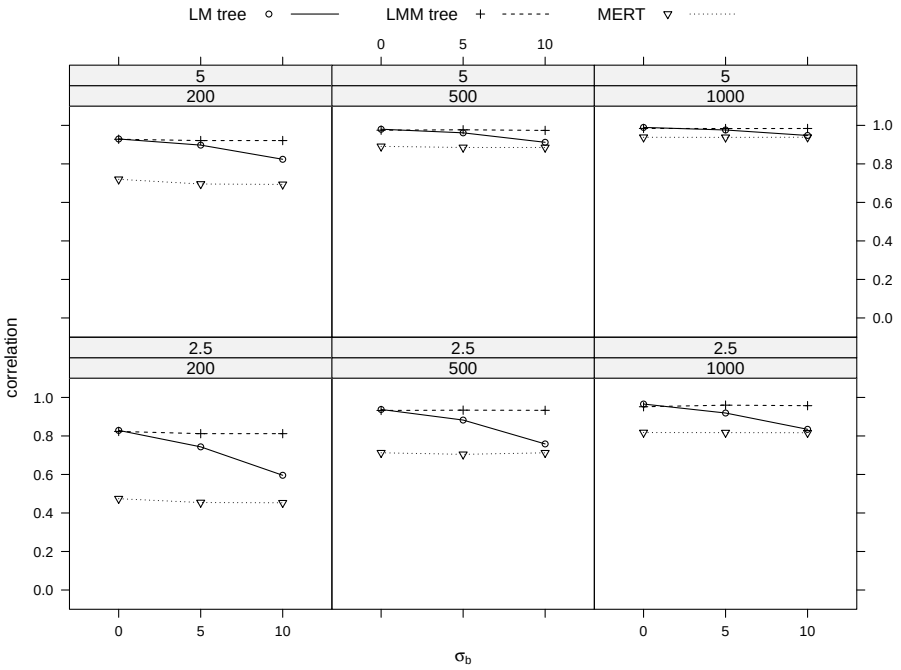


Figure 8. : Average predictive accuracy of LM and LMM trees. Predictive accuracy is defined as the correlation between the true and predicted treatment-effect differences. Rows represent absolute treatment-effect differences in subgroups with treatment-effect differences, columns represent sample size.

Study II: Type-I error of LM and LMM trees

Method

Design. In the second simulation study we assessed the Type-I error rate of LM and LMM tree. In the datasets in this study there was only a main effect of treatment in the population. Put differently, there was only a single global value of $\beta_j = \beta$ in every dataset. A Type-I error was defined as the proportion of datasets without treatment-subgroup interactions which were erroneously partitioned by the algorithm.

To assess the effect of the treatment-effect difference β, an additional facet was added to the *General simulation design*:

7. Two levels for β, the global mean difference in treatment outcomes: $\beta = 2.5$ (corresponding to a medium effect size, Cohen's $d = 0.5$) and $\beta = 5.0$ (corresponding to a large effect size; Cohen's $d = 1.0$).

For each cell in the simulation design, 50 datasets were generated. In every dataset, the outcome variable was calculated as $y_i = x_i^\top \beta + z_i^\top b_m + \epsilon_i$.

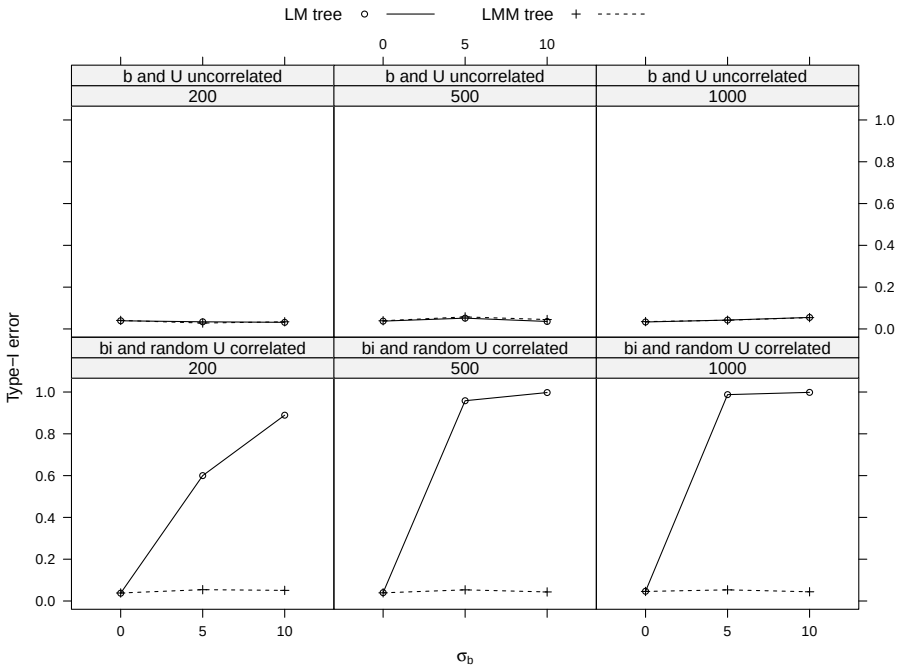


Figure 9. : Type-I error rate of LM and LMM trees. Rows represent dependence between random effects (b) and one of the partitioning variables U_k; columns represent sample size.

Assessment of performance. To assess the Type-I error rates of LM and LMM trees, tree sizes were calculated and trees of size > 1 were classified as Type-I errors. The nominal Type-I error rate for both LM and LMM trees equals .05, corresponding to the pre-specified significance level α for the parameter instability tests.

Results

In datasets without treatment-subgroup interactions, average tree size was 1.09 (SD = 0.44) for LMM trees, and 2.02 (SD = 1.68) for LM trees. The average Type-I error rate was only .04 for LMM trees, and .33 for LM trees. Main predictors of type-I error are depicted in Figure 9, which shows that LMM trees have a Type-I error rate somewhat below the pre-specified α level in all conditions. The same goes for LM trees, when random effects are absent, or uncorrelated to one of the partitioning covariates. When the random intercept is correlated with one of the potential partitioning covariates, the type-I error rapidly increases for LM trees. With increasing sample size or random-effects variance, LM trees will yield a larger number of spurious splits.

Study III: Recovery of piecewise and continuous interactions by LMM trees and LMMs with pre-specified interactions

Method

Interaction design. The treatment-subgroup interactions in Study I (Figure 5) can be referred to as piecewise interactions, as their effect is a stepwise function of the moderator (partitioning) variables. Trees are pre-eminently suited for recovering such piecewise interactions, but may have difficulty when the true interactions are continuous functions of moderator variables (for example, $U_1 \cdot U_2$). At the same time, linear regression models with pre-specified interaction terms may perform well in recovering continuous interactions, but may have difficulty in recovering piecewise interactions. Therefore, in the third simulation study, we added a seventh facet to the *General simulation design* described above:

7. Three levels for interaction type: continuous, piecewise and combined piecewise-continuous interactions.

For datasets with purely piecewise interactions, the same partition as in Study I (Figure 5) was used. In other words, the outcome variable in this design was calculated as $y_i = x_i^\top \beta_j + z_i^\top b + \epsilon_i$, with the value of β_j depending on the values of U_2, U_1 and U_5.

For the datasets with both piecewise and continuous interactions, the partition as depicted in Figure 5 was also used. However, the fixed-effects part $x_i^\top \beta_j$ in each of the terminal nodes now comprised continuous main and (treatment) interaction effects of the partitioning variables. The corresponding node-specific parameters are presented in Table 1. The β_j values were chosen to yield the same treatment-subgroup means as in Figure 5. The interaction terms were created using centered U_k variables, calculated by subtracting their variable means.

In datasets with purely continuous interactions, β has a global value and no subscript, comprising only purely continuous main and interaction effects, as shown by the single β column in Table 1.

Furthermore, in this simulation study, the number of cells in the design was reduced by limiting the fourth facet of the data-generating design to a single level ($M = 25$ clusters), as Study I and II indicated no effects of the number of clusters. The fifth facet of the data-generating design was limited to two levels ($\sigma_b = 2.5$ and $\sigma_b = 7.5$). For every cell of the design, 50 datasets were generated.

LMMs with pre-specified interactions. LMMs were estimated by specifying main effects for all covariates U_k and the treatment indicator, first-order interactions between all pairs of covariates U_k, and second-order interactions between all pairs of covariates U_k and treatment. Continuous predictor variables were centered by subtracting the mean value, before calculating and including the interaction term in the LMM.

Table 1:: Fixed-effects terms in simulations with continuous and combined continuous and piecewise interaction designs.

Term	β_3	β_4	β_6	β_7	β
intercept	27	27	27	27	27
U_2	.1	.1	.1	.1	.1
$U_2 \cdot U_1$	$-.357$	0	0	0	$-.357$
$U_2 \cdot U_5$	0	0	0	.357	.357
$U_2 \cdot U_1 \cdot$ treatment	$-.151$	0	0	0	$-.151$
$U_2 \cdot U_5 \cdot$ treatment	0	0	0	.151	.151

Note: Subscripted β values refer to the terminal nodes in Figure 5 for the combined piecewise and continuous interaction design; β without subscript refers to the global coefficients in the continuous interaction design.

Assessment of performance. Predictive accuracy was assessed in terms of the correlation between the true and predicted treatment-effect differences in test datasets. As the full LMM models were likely to overfit, LMMs were refitted on the training data, using only the predictors with p-values $< .05$ in the original LMM. Predictions for test observations were obtained using the refitted LMMs.

Results

On average, LMM trees showed somewhat higher accuracy: the average correlation between true and predicted treatment-effect differences was .54 (SD = .40) for LMM trees and .51 (SD = .43) for LMMs. The effects of the most important predictors of predictive accuracy are depicted in Figure 10. As Figure 10 indicates, LMM trees show highest predictive accuracy in datasets with purely piecewise interactions, whereas LMMs show highest predictive accuracy in datasets with purely continuous interactions. LMMs perform poorly when interactions are not purely piecewise, and LMM trees perform poorly only when interactions are purely linear.

Performance of both methods improves with increasing sample size. Furthermore, performance of LMM trees is not affected by the number of covariates, whereas the predictive accuracy of LMMs deteriorates when the number of covariates increases, especially when the true interactions are not purely continuous. This indicates that LMM trees are especially useful for exploratory purposes, where there are many potential moderator variables.

In terms of interpretability, LMM trees will often provide simpler models: The LMMs included 12.30 significant terms on average, whereas LMM trees had 3.38 inner nodes on average, requiring only about 3-4 variables to be evaluated for making predictions.

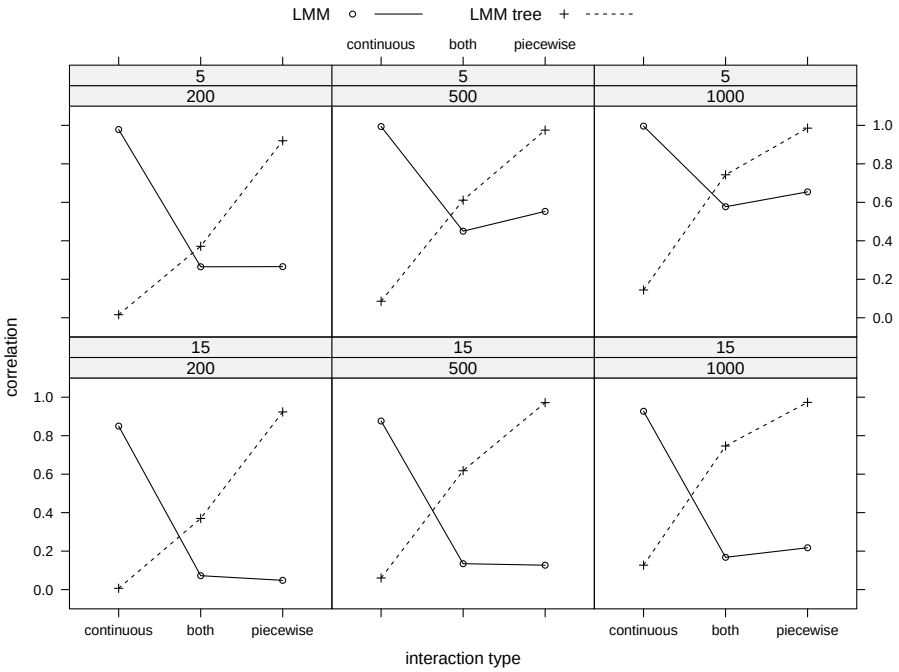


Figure 10. : Average predictive accuracy of LMMs and LMM trees. Predictive accuracy of trees is defined as the correlation between the true and predicted differences between Treatment 1 and 2. Columns represent sample size, rows represent the number of covariates.

Application: Individual patient-level meta-analysis on treatments for depression

Method

Dataset. To illustrate the use of GLMM trees in real application, we employ a dataset from an individual-patient data meta-analysis of Cuijpers et al. (2014). This meta-analysis was based on patient-level observations from 14 RCTs, comparing the effects of psychotherapy (cognitive behavioral therapy; CBT) and pharmacotherapy (PHA) in the treatment of depression. The study of Cuijpers et al. (2014) was aimed at establishing whether gender is a predictor or moderator of the outcomes of psychological and pharmacological treatments for depression. Treatment outcomes were assessed by means of the 17-item Hamilton Rating Scale for Depression (HAM-D; Hamilton, 1960). Cuijpers et al. (2014) found no indication that gender predicted or moderated treatment outcome.

In our analyses, post-treatment HAM-D score was the outcome variable, and potential partitioning variables were age, gender, level of education, presence of a comorbid anxiety disorder at baseline, and pre-treatment HAM-D score. The predictor variable in the linear model was treatment type (0 = CBT and 1 = PHA). An indicator for study was used as

the cluster indicator.

In RCTs, ANCOVAs are often employed, to linearly control post-treatment values on the outcome measure for pre-treatment values. Therefore, post-treatment HAM-D scores, controlled for the linear effects of pre-treatment HAM-D scores were taken as the outcome variable. All models were fitted using data of the 694 patients from 7 studies, for which complete data was available. Results of our analysis may therefore not be fully representative of the complete dataset of the meta-analysis by Cuijpers et al. (2014).

Models and comparisons. As the outcome variable is continuous, we employed an identity link and Gaussian response distribution. The resulting GLMM tree will therefore be refered to as an LMM tree. To compare the accuracy of the LMM tree, we also fitted an LM tree and an LMM with pre-specified interactions to the data. In the LMM, the outcome variable was regressed on a random intercept, main effects of treatment and the potential moderators (partitioning variables) and interactions between treatment and the potential moderators. As it is not known in advance how to interact the potential moderators, higher-order interactions were not included.

Effect size. To provide a standardized estimate of the treatment effect differences in the final nodes of the trees, we calculated node-specific Cohen's d values. Cohen's d was calculated by dividing the node-specific predicted treatment outcome difference by the node-specific pooled standard deviation.

Predictive accuracy and stability. Predictive accuracy of each method was assessed by calculating average correlation between observed and predicted HAM-D post-treatment scores, based on 50-fold cross validation.

The results of recursive partitioning techniques are known to be potentially unstable, in the sense that small changes in the dataset may substantially alter the variables or values selected for partitioning. Therefore, following Philipp, Zeileis, and Strobl (2016), subsampling is used to assess the stability of the selected splitting variables and values. More precisely, variable selection frequencies of the trees are computed from 500 subsamples, each comprising 90% of the full dataset.

Results

The trees and effect sizes resulting from application of LMM and LM trees are presented in Figure 11 and Figure 12, respectively. The LM tree (Figure 12) selected level of education as the first partitioning variable, and presence of a comorbid anxiety disorder as a second partitioning variable, for observations with a higher level of education. By taking into account study-specific intercepts, the LMM tree (Figure 11) indicates that the first split in the LM tree may be spurious. The LMM tree selected presence of a comorbid anxiety

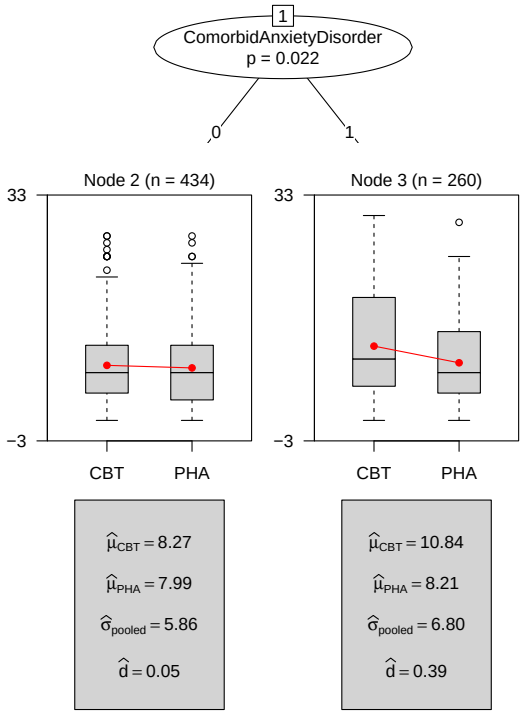


Figure 11. : LMM tree for prediction of treatment outcomes. Upper terminal nodes: y-axes represent post-treatment HAM-D scores, x-axes represent treatment levels (cognitive behavior therapy, CBT vs. pharmacotherapy, PHA). Lower terminal nodes: Subgroup-specific descriptive statistics.

disorder as the only partitioning variable. The terminal nodes of Figure 11 show only a single treatment-subgroup interaction: for patients without a comorbid anxiety disorder, CBT and PHA provide more or less the same reduction in HAM-D scores (Cohen's $d = 0.05$). For patients with a comorbid anxiety disorder, PHA provides a greater reduction in HAM-D scores (Cohen's $d = 0.39$). The estimated intraclass correlation coefficient for the GLMM tree was .05.

The LMM with pre-specified treatment interactions yielded three significant predictors of treatment outcome: like in the GLMM tree, an effect of the presence of a comorbid anxiety disorder was found (main effect: $b = 2.29$, $p = .002$; interaction with treatment: $b = -2.10$, $p = .028$). Also, the GLMM indicated an interaction between treatment and age ($b = .10$, $p = .018$).

Assessment of predictive accuracy by means of 50-fold cross validation indicated better predictive accuracy for the LMM tree than for the LM tree and the LMM. The correlation between true and predicted post-treatment HAM-D total scores averaged over 50 folds was .272 ($var = .067$) for LMM tree, .233 ($var = .064$) for the LMM with pre-specified

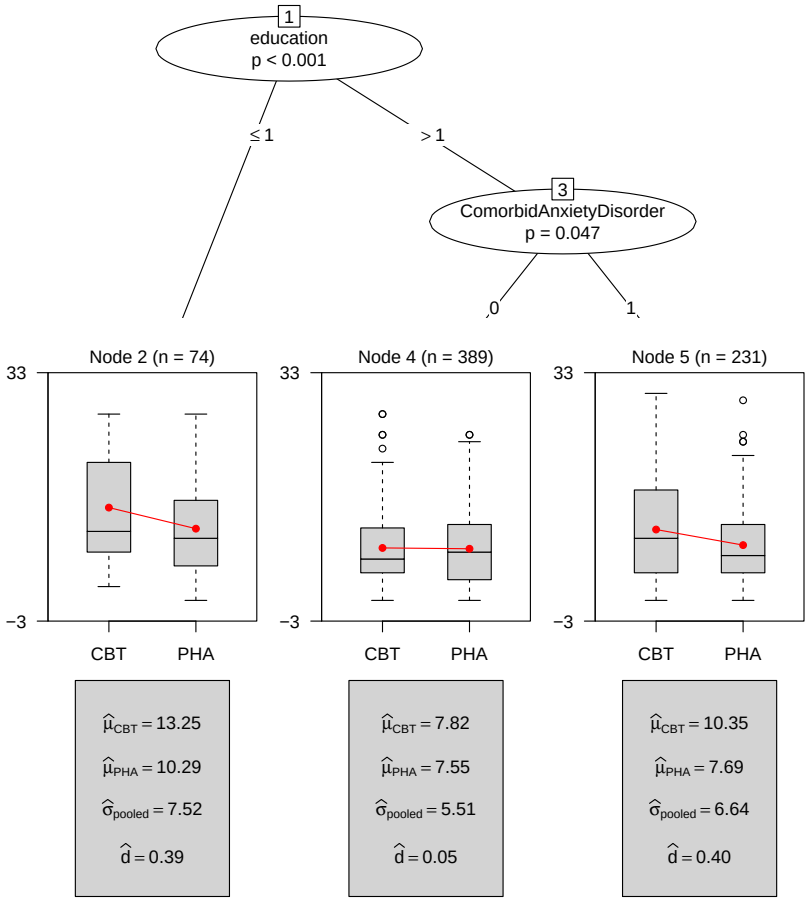


Figure 12. : LM tree for prediction of treatment outcomes. Upper terminal nodes: y-axes represent post-treatment HAM-D scores, x-axes represent treatment levels (cognitive behavior therapy, CBT vs. pharmacotherapy, PHA). Lower terminal nodes: Subgroup-specific descriptive statistics.

interactions and .190 (var = .084) for the LM tree. *(SD rather than var is used in the manuscript up to this point.)* README

Table 2 presents statistics on the variables selected for partitioning in subsamples of the dataset. Presence of a comorbid anxiety disorder was selected for partitioning in the majority of LMM trees grown on subsamples of the dataset, while the other variables were selected in at most 4% of the subsamples. As the comorbid anxiety disorder variable involved only a single splitting value, further assessment of the stability of splitting values was not necessary. *(Practitioners might still worry that the CAD variable is not selected more often – only in a bit more than 50% of the cases. Maybe this would be better in bootstrap sampling rather than resampling because the sample size is not reduzed?)* README

Table 2:: Variable selection statistics

	Selection frequency	
Variable	LM tree	LMM tree
Education	.956	.014
ComorbidAnxietyDisorder	.398	.528
HRSDt0	.034	.002
Age	.000	.022
Gender	.002	.004

Note. Frequencies are calculated over 500 random subsamples of the complete dataset. Frequencies do not add up to 1, as trees may involve multiple or no splits.

Discussion

Summary

(This summary is quite extensive. Personally, I would prefer a more condensed summary of 'highlights'.) README

We presented the GLMM tree algorithm, which allows for estimation of a GLM-based recursive partition, as well as estimation of random-effects parameters. We hypothesized GLMM trees to be well suited for the detection of treatment-subgroup interactions in clustered datasets and confirmed this by our simulation studies.

GLMM trees accurately recovered the subgroups in 90% of simulated datasets with treatment-subgroup interactions. In contrast, GLM trees accurately recovered treatment-subgroup interactions in only 61% of these datasets. In terms of predictive accuracy, GLMM trees outperformed GLM trees as well as MERT, with an average correlation between true and predicted treatment-effect differences of .94 for GLMM trees, .88 for GLM trees and .75 for MERT. The Type-I error rate of GLMM trees very closely approximated the α level used for testing parameter stability: GLMM trees erroneously detect subgroups in 4% of datasets without treatment-subgroup interactions, whereas GLM trees erroneously detect subgroups in 33% of those datasets.

The better performance of GLMM trees was mostly observed when random effects in the datasets were sizeable, and random intercepts were correlated with potential partitioning variables. In those datasets, GLM trees are likely to detect spurious splits and subgroups. At the same time, GLM trees showed less power to detect the true subgroups in the presence of random effects. As expected, the accuracy of MERT was not affected much by the presence of random effects, but only approached the accuracy of GLMM trees in datasets with the largest sample and effect sizes. GLMM trees especially outperformed the other methods

when effect size was small (i.e., Cohen's $d = .5$) and sample size was small (i.e., 200). Such effect and sample sizes are quite common in multi-center clinical trials, and GLMM trees may therefore provide a helpful tool for subgroup detection in those instances.

When random effects were absent from the simulated datasets, GLM and GLMM trees yielded very similar predictive accuracy. This finding is of practical importance, as it indicates that application of GLMM trees will not 'hurt': GLM trees and GLMM trees are expected to perform equally well in the absence of random effects, while GLMM trees will likely outperform GLM trees in the presence of random effects.

Compared to linear mixed-effects models with pre-specified interactions, GLMM trees provided somewhat better accuracy, on average. As expected, GLMM trees performed poorly in datasets with purely continuous interactions, but much better than GLMMs when interactions were at least partly piecewise. We found a clear advantage of GLMM trees in the presence of larger numbers of potential moderator variables, indicating that GLMM trees are much better suited than GLMMs for exploratory analyses. Also, GLMM trees may be easier to interpret: The number of terms in a GLMM increases quadratically with the number of potential moderator variables, yielding complex models. The trees in our simulations were limited to a maximum depth of three, requiring evaluation of at most 3 variables to make a prediction or decision in practice.

In the Application, we obtained similar findings: the GLMM tree yielded higher predictive accuracy, while using a smaller number of variables for prediction than the GLM tree and a GLMM with pre-specified interactions. In addition, the GLMM trees obtained over repeated subsamples of the training data proved to be relatively stable.

Limitations and future research

(I would emphasize early on that GLMM trees rely on the properties of their building blocks, namely GLMMs and MOBs. The effect of misspecifications are studied to some degree for both methods. For GLMM one could cite papers discussing this. For MOB, or trees more broadly, there are also some references. One of these could be the 'To Split or to Mix?' paper by Frick, Strobl, and Zeileis.) README

Recursive partitioning methods were originally developed as a non-parametric tool for classification and regression, assuming the mechanism that generated the data unknown (e.g., Breiman, 2001). However, GLMM trees are obviously a parametric tool, as they fit fixed-effects linear models in the nodes of the tree and a global model for the random effects, in turn introducing several distributional assumptions about the random effects and errors. Misspecification of these distributions will likely have a negative effect on the accuracy of the estimated GLMM tree.

Furthermore, misspecification of partitioning and fixed- and random-effects variables will also reduce accuracy of the resulting GLMM tree. If relevant variables are omitted,

or incorrectly specified, GLMM tree can only approximate the true subgroups using the specified variables. Our simulations indicate that LM trees detect spurious subgroups as a result of misspecifying (that is, not including) the random effects. Reduced accuracy and spurious splits can also be expected to occur when relevant random effects are not included when specifying the GLMM tree model. Furthermore, as the random effects are estimated globally, misspecification of the random effects can have a strong impact on the resulting GLMM tree model.

Another source of misspecification is the inclusion of irrelevant variables. Although our simulations indicate that the performance of GLMM tree was not negatively affected by increasing the number of noise variables specified for partitioning from 2 to 12, the power to detect subgroups may be reduced with much larger numbers of noise variables. Including irrelevant variables in the random or fixed effects may also negatively affect accuracy of GLMM tree, but we have not assessed this in our study.

Users can reduce the risk of misspecification when fitting a GLMM tree in two ways. Firstly, by carefully specifying the predictors of the GLM, the partitioning variables and the random effects. Secondly, by inspecting residuals and assessing stability of the resulting model. In the Tutorial we have shown how residuals can be plotted to assess potential misspecification. In the Application we have shown how the `stablelearner` package can be used to assess tree stability. However, more research on the effects that various types of misspecification will have on the performance of GLMM tree is required.

As GLMM tree fits more complex models than non-parametric tree-based methods, like CART for example, larger sample sizes are needed to fit the model. How much larger likely depends on the complexity of the specified model: Our simulations show that with fixed and random-effects specifications with only a single predictor, a sample size of 200 is sufficient to detect subgroups with moderate differences in treatment effect. More complex fixed- and random-effects specifications will require larger sample sizes; how these affect the performance of GLMM tree requires further research.

In the Introduction we mentioned several existing tree-based methods for treatment-subgroup interaction detection. These methods have different objectives and there is not yet an agreed-upon single best method. In a simulation study, Sies and Van Mechelen (2016) found the method of Zhang, Tsiatis, Davidian, et al. (2012) to perform best, followed by MOB. However, the method of Zhang et al. performed worst under some conditions of the simulation study in terms of the Type I error rate. Further research comparing tree-based methods for treatment-subgroup interaction detection is needed, especially for clustered datasets, as our simulations and comparisons only focused on GLMM tree and GLM-based MOB.

Conclusion

Our results indicate that GLMM tree provides accurate recovery of treatment-subgroup interactions and prediction of treatment effects, both in the presence and absence of random effects and interactions. Therefore, GLMM tree is a promising algorithm for the detection of treatment-subgroup interactions in clustered datasets, for example in multi-center trials or individual-level patient data meta-analyses.

References

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015). *Fitting linear mixed-effects models using lme4* (No. 1).

Breiman, L. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, *16*(3), 199–231.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. New York: Wadsworth.

Bryk, A. S., & Raudenbush, S. W. (1992). *Hierarchical linear models: Applications and data analysis methods*. Newbury Park, CA: Sage.

Ciampi, A. (1991). Generalized regression trees. *Computational Statistics & Data Analysis*, *12*(1), 57–78.

Cohen, J. (1992). A power primer. *Psychological Bulletin*, *112*(1), 155–159.

Cooper, H., & Patall, E. A. (2009). The relative benefits of meta-analysis conducted with individual participant data versus aggregated data. *Psychological Methods*, *14*(2), 165.

Cuijpers, P., Weitz, E., Twisk, J., Kuehner, C., Cristea, I., David, D., et al. (2014). Gender as predictor and moderator of outcome in cognitive behavior therapy and pharmacotherapy for adult depression: An "individual-patients data" meta-analysis. *Depression and Anxiety*, *31*(11), 941–951.

Doove, L. L., Dusseldorp, E., Van Deun, K., & Van Mechelen, I. (2014). A comparison of five recursive partitioning methods to find person subgroups involved in meaningful treatment-subgroup interactions. *Advances in Data Analysis and Classification*, *8*, 403–425.

Driessen, E., Smits, N., Dekker, J., Peen, J., Don, F. J., Kool, S., et al. (2016). Differential efficacy of cognitive behavioral therapy and psychodynamic therapy for major depression: A study of prescriptive factors. *Psychological Medicine*, *46*(4), 731–744.

Dusseldorp, E., Doove, L., & Van Mechelen, I. (2016). Quint: An R package for the identification of subgroups of clients who differ in which treatment alternative is best for them. *Behavior Research Methods*, *48*, 650.

Dusseldorp, E., & Meulman, J. J. (2004). The regression trunk approach to discover treatment covariate interaction. *Psychometrika*, *69*(3), 355–374.

Dusseldorp, E., & Van Mechelen, I. (2014). Qualitative interaction trees: A tool to identify qualitative treatment-subgroup interactions. *Statistics in Medicine*, *33*(2), 219–237.

Fokkema, M., & Zeileis, A. (2016). *glmertree: Generalized linear mixed model trees.* (R package version 0.1-1)

Foster, J. C., Taylor, J. M. G., & Ruberg, S. J. (2011). Subgroup identification from randomized

clinical trial data. *Statistics in Medicine*, *30*(24), 2867–2880.

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

Hamilton, M.(1960). A rating scale for depression. *Journal of Neurology, Neurosurgery and Psychiatry*, *23*(1), 56.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning* (2nd ed.). New York: Springer.

Higgins, J., Whitehead, A., Turner, R. M., Omar, R. Z., & Thompson, S. G.(2001). Meta-analysis of continuous outcome data from individual patients. *Statistics in Medicine*, *20*(15), 2219–2241.

Hothorn, T., & Zeileis, A.(2015, December). partykit: A modular toolkit for recursive partytioning in R. *Journal of Machine Learning Research*, *16*, 3905–3909.

Kraemer, H. C., Frank, E., & Kupfer, D. J. (2006). Moderators of treatment outcomes: Clinical, research, and policy importance. *Journal of the American Medical Association*, *296*(10), 1286–1289.

Kuznetsova, A., Brockhoff, P., & Christensen, R. (2016). *lmertest: Tests in linear mixed effects models.* (R package version 2.0-32)

Lipkovich, I., Dmitrienko, A., Denne, J., & Enas, G.(2011). Subgroup identification based on differential effect search – A recursive partitioning method for establishing response to treatment in patient subpopulations. *Statistics in Medicine*, *30*(21), 2601–2621.

Martin, D. (2015). *Efficiently exploring multilevel data with recursive partitioning.* Unpublished doctoral dissertation, University of Virginia.

Philipp, M., Zeileis, A., & Strobl, C.(2016). A toolkit for stability assessment of tree-based learners. In A. Colubi, A. Blanco, & C. Gatu (Eds.), *Proceedings of COMPSTAT 2016 – 22nd international conference on computational statistics* (pp. 315–325). Oviedo: The International Statistical Institute/International Association for Statistical Computing.

R Core Team.(2016). *R: A language and environment for statistical computing.* Vienna, Austria.

Seibold, H., Zeileis, A., & Hothorn, T. (2016). Model-based recursive partitioning for subgroup analyses. *International Journal of Biostatistics*, *12*(1), 45–63.

Sela, R. J., & Simonoff, J. S. (2011). *Reemtree: Regression trees with random effects.* (R package version 0.90.3)

Sela, R. J., & Simonoff, J. S. (2012). RE-EM trees: A data mining approach for longitudinal and clustered data. *Machine Learning*, *86*(2), 169–207.

Sies, A., & Van Mechelen, I. (2016). *Comparing four methods for estimating tree-based treatment regimes.* (Submitted)

Strobl, C., Malley, J., & Tutz, G. (2009). An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological Methods*, *14*(4), 323.

Su, X., Tsai, C.-L., Wang, H., Nickerson, D. M., & Li, B. (2009). Subgroup analysis via recursive partitioning. *The Journal of Machine Learning Research*, *10*, 141–158.

Zeileis, A., & Hornik, K.(2007). Generalized M-fluctuation tests for parameter instability. *Statistica Neerlandica*, *61*(4), 488–508.

Zeileis, A., Hothorn, T., & Hornik, K. (2008). Model-based recursive partitioning. *Journal of*

Computational and Graphical Statistics, *17*(2), 492–514.

Zhang, B., Tsiatis, A. A., Davidian, M., Zhang, M., & Laber, E. (2012). Estimating optimal treatment regimes from a classification perspective. *Stat*, *1*(1), 103–114.

Zhang, B., Tsiatis, A. A., Laber, E. B., & Davidian, M. (2012). A robust method for estimating optimal treatment regimes. *Biometrics*, *68*(4), 1010–1018.

Appendix

R code for generating artificial motivating dataset

```
set.seed(123)
treatment <- rbinom(n = 150, size = 1, prob = .5)
duration <- round(rnorm(150, mean = 7, sd = 3))
anxiety <- round(rnorm(150, mean = 10, sd = 3))
age <- round(rnorm(150, mean = 45, sd = 10))
error <- rnorm(150, 0, 2)
cluster <- error + rnorm(150, 0, 6)
rand_int <- sort(rep(rnorm(10, 0, 1), each = 15))
rand_int[order(cluster)] <- rand_int
error <- error - rand_int
cluster[order(cluster)] <- rep(1:10, each = 15)
node3t1 <- ifelse(duration <= 8 & anxiety <= 10 & treatment == 0, -2, 0)
node3t2 <- ifelse(duration <= 8 & anxiety <= 10 & treatment == 1, 2, 0)
node5t1 <- ifelse(duration > 8 & treatment == 0, 2.5, 0)
node5t2 <- ifelse(duration > 8 & treatment == 1, -2.5, 0)
depression <- round(9 + node3t1 + node3t2 + node5t1 + node5t2 + .4*treatment +
                    error + rand_int)
treatment <- factor(treatment, labels = c("Treatment 1", "Treatment 2"))
a_m_data <- data.frame(depression, treatment, cluster, age, anxiety, duration)
```