Chapter 4

Detecting Treatment-Subgroup Interactions in Clustered Datasets: Combining Model-Based Recursive Partitioning and Random-Effects Estimation

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. Several tree-based algorithms have been developed for the detection of such treatment-subgroup interactions. In many instances, however, datasets may have a clustered structure, where observations are clustered within, for example, research centers or persons. In the current paper we propose a new algorithm, lmertree, that allows for detection of treatment-subgroup interactions, as well as estimation of cluster-specific random effects. The algorithm uses model-based recursive partitioning (MOB) to detect treatment-subgroup interactions, and a linear mixed-effects model for estimation of random-effects parameters. In a simulation study, we evaluate the performance of lmertree and compare it with that of MOB trees without random-effects estimation. In datasets without treatment subgroup interaction, lmertree was found to have a much lower Type I error rate than MOB without random effects (4 and 33%, respectively). Furthermore, in datasets with treatment-subgroup interactions, lmertree recovered the true treatment subgroups much more often than MOB without random effects (90% and 61% of the datasets, respectively). Also, lmertree predicted treatment outcome differences more accurate than MOB without random effects (average accuracy of .94 and .88, respectively). We illustrate the application of lmertree on a patient-level dataset of a meta-analysis on the effects of psycho- and pharmacotherapy for depression. We conclude that lmertree is a promising algorithm for the detection of treatment-subgroup interactions in clustered datasets, and discuss directions for future research.
4 Detecting Treatment-Subgroup Interactions in Clustered Datasets

4.1 Introduction

In medicine-efficacy research, the one-size-fits-all paradigm is slowly losing ground, and personalized medicine is becoming increasingly important. Personalized medicine presents us with the challenge of finding 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). In most cases, 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 a-priori. This may hamper identification of which treatments work best for whom, especially when there are no a-priori hypothesis about treatment-subgroup interactions. As noted by Kraemer, Frank en Kupfer (2006), there is a need for methods that generate, instead of test, hypotheses and that are specifically directed at the detection of treatment interactions.

Tree-based methods are such hypothesis-generating methods, as they can automatically detect subgroups which differ in terms of the expected outcomes for one or more treatments. Due to their flexibility, tree-based methods are preeminently suited to detect treatment-subgroup interactions: they can handle many potential predictor variables at once, and can automatically detect (higher order) interactions between predictor variables. Several promising tree-based algorithms and software packages have been developed to assist in the detection of treatment-subgroup interactions (Dusseldorp & Van Mechelen, 2014; Dusseldorp & Meulman, 2004; X. Su, Tsai, Wang, Nickerson & Li, 2009; Foster, Taylor & Ruberg, 2011; Lipkovich, Dmitrienko, Denne & Enas, 2011; Zeileis et al., 2008; see Doove et al., 2014 for an overview). Among these algorithms, model-based recursive partitioning (MOB; Zeileis et al., 2008) may be the most flexible method for detecting treatment-subgroup interactions, as it of-
Detectors a very generic data-analytic framework for detecting partitions in a dataset, with different model parameter estimates. The recursive partitioning in MOB can be based on a broad class of parametric models that can be fitted using M-type estimators (Zeileis et al., 2008), the most well-known example being the generalized linear model. Earlier, MOB based on a linear model has been successfully applied by Driessen et al. (2014) in the detection of subgroups with differential treatment outcomes for two different psychotherapies.

Single randomized clinical trials may often be underpowered for the detection of treatment-subgroup interactions. Therefore, meta-analysis of individual-level patient data (IPD), in which datasets from several RCTs are pooled, is becoming increasingly popular (Koopman, Van der Heijden, Glasziou, Grobbee & Rovers, 2007). In such analyses, the clustered structure of the dataset should be taken into account by including study-specific effects in the model, prompting the need for modeling both fixed and random effects (e.g., Friedenreich, 1993; DerSimonian & Laird, 1986; Higgins, Whitehead, Turner, Omar & Thompson, 2001). Likewise, analyses of datasets from clinical trials conducted in multiple research centers, and longitudinal datasets may also require estimation of random effects.

However, none of the aforementioned tree-based algorithms allow for estimation of random effects. Ignoring the clustered structure of datasets may lead to biased inference, due to underestimated standard errors (e.g., Bryk & Raudenbush, 1992; Hox, 1998; Van den Noortgate, Opdenakker & Onghena, 2005). More importantly, when the interest is in subgroup detection, ignoring random effects may result in the detection of spurious subgroups.

In the current paper, we present lmertree, a tree-based algorithm for treatment-subgroup interaction detection, which takes the clustered nature of datasets into account. The algorithm combines MOB with the estimation of random effects, thus allowing for the detection of treatment-
subgroup interactions, as well as accounting for variation between clusters.

In what follows, we will first discuss existing frameworks for estimating treatment effects: the linear fixed-effects model, model-based recursive partitioning, and the linear mixed-effects model. Then, we describe the lmertree algorithm, which combines model-based recursive partitioning and mixed-effects models. In the method and results section, we present a simulation study evaluating the comparative accuracy of lmertree, and provide an illustration by application of lmertree to an existing dataset on the outcomes of treatments for depression.

4.2 General modeling framework

4.2.1 Linear fixed-effects model

In a clinical trial, where the outcomes of two treatments are compared, an overall linear regression model may be used to estimate treatment effects. Let $N$ be the total number of observations in a dataset. The following linear regression model may be used to estimate the effects of the treatments:

$$y = X\beta + \epsilon$$  \hspace{1cm} (4.1)

Where $y$ is an $N \times 1$ column vector of outcome variable values, $X$ is an $N \times 2$ matrix of predictor variable values (the first column being a vector of 1s, and the second column a dummy indicator for treatment type); $\beta$ is a $2 \times 1$ column vector of regression coefficients (the first element of $\beta$ represents the intercept: the mean value of the outcome variable for treatment 1; and the second element of $\beta$ represents the mean difference

---

1 In the Appendix, an overview of the notation used in the current chapter is provided
in outcome variable $Y$ for treatments 1 and 2). Further, $\epsilon$ is an $N \times 1$ column vector of residuals, which is assumed to be normally distributed with mean 0 and variance $\sigma_\epsilon^2$, and independent of all other variables in the model.

In Figure 4.1, a graphical representation of the model in Equation 4.1 is provided, based on simulated data. In Figure 4.1, the boxplots show the distribution of the outcome variable (posttreatment depression score) among 150 participants, who were randomly assigned to treatment 1 and treatment 2, and the red dots and line represent the regression coefficients $\beta$. Figure 4.1 suggests little overall difference between the outcomes of treatment 1 and treatment 2, as the slope of the regression line is nearly zero. We shall see that this does not necessarily mean that posttreatment depression score and treatment type are unrelated, as the effect of treatment may be moderated by other variables. Conditional on other variables (i.e., subgroup indicators), the relationship between $X$ and $y$ may vary in strength and/or direction (c.f., Simpson’s paradox; Simpson, 1951). The challenge in detecting treatment-subgroup interactions is to find the variables that should be conditioned upon.

### 4.2.2 Model-based recursive partitioning

The rationale behind MOB is that a global model for all observations, like that in Equation 4.1, 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 a better model in each cell of the partition (Zeileis et al., 2008). This is reminiscent of the classification and regression tree (CART) algorithm of Breiman et al. (1984), which splits the dataset into subsets, of which the distributions of the outcome variable are most different. Whereas CART trees have constant fits in the terminal nodes, MOB trees have parametric models with one or more predictor variables in their terminal nodes.
To find partitions and better-fitting local models, the MOB algorithm tests for parameter instability: differences in parameter estimates across partitions of the dataset, which are defined by one or more additional covariates. To this end, the MOB algorithm cycles iteratively through the following steps (Zeileis et al., 2008): (1) fit the parametric model to the data set, (2) test for parameter instability over a set of partitioning variables, (3) if there is some overall parameter instability, split the data set with respect to the variable associated with the highest instability, (4) repeat the procedure in each of the resulting subsamples.

More specifically, in step (2), to test for parameter instability, so-called scores are computed, using the score function. The expected value of the scores over all observations in a data set is zero, by definition. Under the null hypothesis of parameter stability, the scores do not systematically deviate from the expected value of zero, when the observations are ordered by the values of a potential partitioning variable $U_k$. To statistically test whether there are systematic deviations of the scores from zero with re-
spectrum to variable $U_k$, the class of generalized M-fluctuation tests is used in MOB (Zeileis, 2005; Zeileis & Hornik, 2007).

If the null hypothesis of parameter stability in step (2) can be rejected, that is, if at least one of the partitioning variables $U_k$ has a p-value for the M-fluctuation test below the pre-specified significance level $\alpha$, the data set is partitioned into two subsets in step (3). In step (3), a binary partition is created using $U_{k*}$, the variable with the minimal p-value in step (2). The split point for $U_{k*}$ is selected, that minimizes the sum of the residual sums of squares in both partitions (Zeileis et al., 2008). In step (4), steps (1) through (3) are repeated in each partition, until the null hypothesis of parameter stability can no longer be rejected.

Due to the binary recursive nature of MOB, the resulting partitions can be represented as a tree, with parametric models in the terminal nodes. In the linear-model case, this is called a linear model tree (lmtree): a tree with a different fixed-effects linear regression model $y = X\beta + \epsilon$ in all $j = 1, ..., t$ terminal nodes. As a result, the model in Equation 4.1 is estimated separately for each terminal node, yielding a $\beta$ that is a $2t \times 1$ column vector $[\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T$, where each $\beta_j$ relates to the subsample of observations in terminal node $j$. $X$ becomes an $N \times 2t$ design matrix, with the $N_j \times 2$ matrices of predictor variable values in terminal node $j$ on the diagonal blocks. Note that when the null hypothesis of parameter stability can not be rejected in the first iteration, no partitions are created by MOB, and the resulting tree has a single node with the dimensions of $X$ remaining $N \times 2$ and the dimensions of $\beta$ remaining $2 \times 1$.

Figure 4.2 provides an example of a graphical representation of an lmtree, based on the same simulated data of Figure 4.1. By using several additional covariates (i.e., anxiety questionnaire score, duration of depressive symptoms at baseline, age), MOB separated the observations into four partitions, each with a different estimate for $\beta$ (i.e., different posttreatment depression means for treatment 1 and/or 2). Figure 4.2 shows a subgroup with low duration and low anxiety, for which treatment 1 is...
much more beneficial (i.e., lower posttreatment depression scores); a sub-group of patients with low duration and anxiety $\leq 12$, for whom treatment 1 is somewhat more beneficial; a subgroup of patients with low duration and anxiety $> 12$, for whom treatment 2 is slightly more beneficial; and a subgroup of patients with duration of depressive symptoms $> 6$ months, for whom treatment 2 is much more beneficial. Age did not have a main or moderating effect on treatment outcomes, and therefore does not appear as a splitting variable in the tree.

Figure 4.2: Example of tree representation of model-based recursive partitions, based on the same data as Figure 4.1. Three additional covariates (anxiety questionnaire score, duration of depressive symptoms at baseline in months and age) were used as potential splitting variables.
4.2.3 Linear mixed-effects model

When the dataset consists of observations from multiple clusters, the fixed-effects model in Equation 4.1 can be extended to include cluster-specific, or random, effects. With the inclusion of random effects, the model becomes a linear mixed-effects model:

\[ y = X{\beta} + Z{b} + \epsilon \]  

(4.2)

where \( Z \) is the \( N \times n \) (block diagonal) design matrix of random-effects predictor variables, with \( n \) being the number of clusters. \( Z \) has \( N_i \times 1 \) vectors of ones on the diagonal blocks, with \( N_i \) being the number of observations in cluster \( i \); \( b \) is the \( n \times 1 \) column vector \([b_1, ..., b_i, ..., b_n]^T\) of random-effects coefficients, and it is assumed that that \( b_i \) is normally distributed with mean zero and (scalar) variance \( \sigma^2_{b_i} \) (note that in the current paper, for simplicity, we incorporate random intercepts only, but random slopes can easily be included in the model). It is assumed that \( b_i \) and \( \epsilon \) are each independent from all other variables in the model. The parameters of the linear mixed-effects model can be estimated with, for example, maximum likelihood (ML) and restricted ML (REML), as described in (Laird & Ware, 1982; Bryk & Raudenbush, 1992).

4.2.4 Combining model-based recursive partitioning and random-effects estimation

As noted earlier, the linear (mixed-effects) model is not well suited for the detection of treatment-subgroup interactions, whereas the MOB algorithm is, but does not allow for estimation of random effects. Therefore, we propose an extension of the MOB algorithm, which does allow for estimation of random effects. For this extension, we replace the fixed-effects part of the mixed-effects model by an lmtree, so \( \beta \) becomes the \( 2t \times 2 \) column vector \([\beta_1^T, ..., \beta_j^T, ..., \beta_i^T]^T\), and \( X \) becomes the \( N \times 2t \)
design matrix, with the $N_j \times 2$ matrices of predictor variable values in terminal node $j$ on the diagonal blocks. The random-effects coefficients $b_i$ in Equation 4.2 are estimated globally, that is, the random-effects model is invariant over the terminal nodes of the lmtree. This approach resembles that of Hajjem, Bellavance en Larocque (2011) and Sela en Simonoff (2012), who added random-effects estimation to CART trees with constant fits, instead of linear models, in the terminal nodes.

To estimate $\beta$ and $b$ for this model, we take an iterative approach, alternating between assuming random effects $b$ known, allowing for estimation of $\beta = [\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T$; and assuming $\beta$ known, allowing for estimation of the random-effects vector $b$. As the algorithm involves estimation of local linear fixed effects, as well as cluster-specific random effects, we will refer to it as linear mixed-effects regression tree, or lmertree. In Figure 4.3, a schematic representation of the lmertree algorithm is presented.

**Imertree** algorithm

Step 0: Initialize by setting $b = 0$ and $\beta = 0$.

Step 1: Given the current estimate of $b$, estimate $\beta = [\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T$.

Step 2: Given the current estimate of $\beta = [\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T$, estimate $b$.

Step 3: Repeat steps 1 and 2 until convergence.

Figure 4.3: Description of the lmertree algorithm

The lmertree algorithm initializes by setting $b$ to zero, since both the random effects are initially unknown. In every iteration, the lmtree (i.e., partitions $j$ in $X$, and the associated fixed-effects coefficient vectors $\beta_j$) and random-effects coefficients $b$ are re-estimated. The lmtree is estimated, given the estimated value of $b$ from the last iteration; in turn, $b$ is estimated, given the estimated value of $\beta$ from the current iteration. Iterations are continued until convergence, which is monitored by computing the log-likelihood criterion of the mixed-effects model in Equation 4.2.
In what follows, we present a simulation study in which we assess the performance of lmertree in recovering treatment-subgroup interactions, and predicting differences between the outcomes of two treatments. Furthermore, we will compare the performance of lmertree with that of lmtree without random-effects estimation. We generated a large number of datasets with and without treatment-subgroup interactions, varying seven parameters: total sample size, number of clusters, variance of the random coefficients, number of potential partitioning variables, correlations between potential partitioning variables, correlations between random coefficients and potential partitioning variables, and the magnitude of the differences in treatment outcomes.

For lmertree, we expect the accuracy of recovered trees and predictions to improve with increasing sample size and magnitude of the difference in treatment outcomes. For lmtree, we have the same expectation, when random effects are absent; that is, when the variance of the random coefficients is zero, we expect lmtree and lmertree to perform equally well. When random effects are present, we expect lmertree to perform better than lmtree, in terms of accuracy of recovered trees and predictions, especially when the variance of the random coefficients and the correlations between the random coefficients and potential partitioning variables increases.

4.3 Simulation study: method

4.3.1 Software

R (R Development Core Team, 2010) was used for generation and analysis of all datasets. Two additional R packages were used: partykit (a toolkit for recursive partitioning; Hothorn & Zeileis, 2014; Zeileis et al., 2008) for the estimation of linear models trees, and lme4 (linear mixed-effects models; Bates, Maechler & Bolker, 2012) for the estimation of
random-effects coefficients. For all functions, default settings were used, with exception of the maximum tree depth, which was set to four (i.e., eight terminal nodes) for all trees. This means that the random-effects coefficients were estimated with REML.

**Tree estimation**

For estimating lm trees, the `lmmtree` function from `partykit` was used. The `lmmtree` function builds a linear model tree: a model-based recursive partition based on least squares regression (Equation 4.1).

For estimating lmertrees, we implemented the algorithm as represented in Figure 4.3 in a function that estimates $\beta = [\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T$ with the `lmmtree` function, and estimates $b$ with the `lmer` function from `lme4`.

More specifically, in each iteration, in step (2), the predictions of the lm tree from step (1) are included as an offset (i.e., a variable to be included in the model, with a-priori coefficient of one) in estimating the random coefficients. In each iteration $>1$, in step (1), the predictions of the mixed-effects model from step (2) of the last iteration are included as an offset in estimating the lm tree. Convergence is monitored using the log-likelihood value of the linear mixed-effects model estimated in step (2): when the difference in log-likelihoods of two consecutive iterations is less than .001, lmertree has converged.

**4.3.2 Simulation design**

Datasets with treatment-subgroup interactions

For generating datasets with treatment-subgroup interactions, we used a treatment-subgroup interaction design from Dusseldorp en Van Mechelen (2014), which is also depicted in Figure 4.4. Figure 4.4 shows four subgroups, of which three groups have differential treatment effects, as
shows by their values of $\beta_{j1}$ and $d$. The four subgroups are characterized by their values on the partitioning variables $U_2$, and $U_1$ or $U_5$. In other words, $U_1$, $U_2$ and $U_5$ are true partitioning variables, whereas the other potential partitioning variables in $U$ (e.g., $U_3$, $U_4$) are noise variables.

Datasets without treatment-subgroup interactions

For generating datasets without treatment-subgroup interactions, we used a design in which there is only a main effect of therapy in the population. In the data-generating model for the datasets with a main treatment effect only, the mean value of the outcome variable was 30.0 in all datasets, and $\beta$ and $d$ have the same value for all observations, e.g., when $d = 1.0$, $\beta = (27.5, 5.0)$.

Parameters of the data-generating process

In generating the datasets, we varied seven parameters of the data-generating process:
1. Three levels for the total number of observations: \( N = 200, N = 500, N = 1000 \).

2. Two levels for the number of potential partitioning covariates \( U_1 \) through \( U_r \): \( r = 5, r = 15 \) (where only \( U_1, U_2 \) and \( U_5 \) are true partitioning variables and the other \( U_k \)s are not).

3. Two levels of intercorrelations between the covariates \( U_1 \) through \( U_r \): \( \rho_{U_k, U_{k'}} = 0, \rho_{U_k, U_{k'}} = .3 \).

4. Three levels for the number of clusters: \( n = 5, n = 10, n = 25 \).

5. Three levels for the population standard deviation of the normal distribution from which the cluster specific intercepts are drawn: \( \sigma_{b_i} = 0, \sigma_{b_i} = 5, \sigma_{b_i} = 10 \).

6. Three levels for the intercorrelations between \( b_i \) and one of the \( U_k \)s: \( b_i \) and \( U_k \) uncorrelated, \( b_i \) correlated with a true partitioning covariate (i.e., \( U_2, U_1, \) or \( U_5 \), introducing a correlation between the correlated \( U_k \) and \( b_i \) of about 0.42), \( b_i \) correlated with a non-partitioning covariate (i.e., \( U_3 \) or \( U_4 \), introducing a correlation between the correlated \( U_k \) and \( b_i \) of about 0.42).

7. Two different levels for the unstandardized mean difference in treatment outcomes, in subgroups with differential effects for treatment 1 (\( X_1 = 0 \)) and treatment 2 (\( X_1 = 1 \)). The levels for mean differences in subgroups with differential treatment effect were \( |\beta_1| = 2.5 \) (corresponding to a medium effect size, Cohen’s \( d = 0.5 \); Cohen, 1992) and \( |\beta_1| = 5.0 \) (corresponding to a large effect size; Cohen’s \( d = 1.0 \)).

We expected the last three facets of the simulation design to have the strongest effect, and we expect the first four facets to have little effect on the difference in performance between lmtree and lmertree.

For each cell, 50 datasets with treatment-subgroup interactions were generated, resulting in \( 50 \times 3 \times 2 \times 2 \times 3 \times 3 \times 2 = 32,400 \) training datasets.
For the datasets with a main treatment effect only, the sixth facet of the data-generating process had only two levels (\( b_i \) correlated with one of the \( U \) variables, and \( b_i \) not correlated with any of the \( U \) variables). Therefore, 
\[ 50 \times 3 \times 2 \times 2 \times 3 \times 3 \times 2 \times 2 = 21,600 \]
datasets without treatment-subgroup interactions were generated.

**Variable distributions**

Following a design similar to that of Dusseldorp en Van Mechelen (2014), all covariates \( U_1 \) through \( U_{15} \) were drawn from a multivariate normal distribution with means \( \mu_{U_1}, \mu_{U_2}, \mu_{U_4}, \) and \( \mu_{U_5} \) fixed at 10, 30, -40 and 70, respectively. The means for all other covariates (i.e., \( \mu_{U_3}, \) and \( \mu_{U_6} \) through \( \mu_{U_{15}} \)) were drawn from a discrete uniform distribution of natural numbers on the interval \([-70, 70]\). All covariates \( U_1 \) through \( U_{15} \) have the same standard deviation: \( \sigma_{U_k} = 10 \). Correlations between the variables in \( U \) vary according to the third facet of the simulation design described above.

To generate the random error term \( \epsilon \), for every observation we drew a value from a normal distribution with \( \mu_\epsilon = 0 \) and \( \sigma_\epsilon = 5 \).

To generate the cluster-specific intercepts \( b_i \), we partitioned the sample into 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 we drew a single value \( b_i \) from a normal distribution with \( \mu_{b_i} = 0 \) and the value of \( \sigma_{b_i} \) given by the fifth facet of the simulation design. When \( b_i \) was correlated with one of the potential partitioning variables \( U \), the partitioning or non-partitioning covariate correlated with \( b_i \) was randomly selected.

To generate the node-specific fixed-effects, we partitioned the sample according to the terminal nodes of the tree in Figure 4.3. In combination with the seventh facet of the simulation design, this determines the values of \( \beta_j \). For every observation, we generated a binomial variable (with
4 Detecting Treatment-Subgroup Interactions in Clustered Datasets

$p = .5$) as an indicator for treatment type (i.e., the second column of $X$).

Finally, outcome variable $Y$ was calculated according to the model in Equation 4.2.

### 4.3.3 Evaluation of performance

#### Tree size

For every dataset, we calculated the tree size of lmtrees and lmertrees, and compared them with the true tree size. For datasets without treatment subgroup interactions, and a main treatment effect only, this allowed us to assess the Type I error: the probability that the algorithm erroneously partitions the dataset.

To detect predictors of tree size for lmtree and lmertree, an lmtree was used. The outcome variable in this tree is the number of nodes of the generated trees. The predictor variable for the linear model is algorithm type (lmtree or lmertree), and the potential partitioning variables are the parameters of the data-generating process, which is described in the Simulation design subsection. To ease interpretation, maximum depth of the lmtree was set to four.

#### Recovery of partitioning variables and values

Recovery of partitioning variables and values was assessed for datasets with treatment-subgroup interactions, as the datasets with main treatment effects only do not have true splitting variables and values. To assess the accuracy with which partitioning variables and values were recovered, we calculated the proportion of accurately recovered lmtrees and lmertrees.
An accurately recovered tree was defined as a tree with three splits, in which the first split involved variable $U_2$ and a value of $30 \pm 5$, the second split involved variable $U_1$ and a value of $17 \pm 5$, and the third split involved variable $U_5$ and a value of $63 \pm 5$. This amounts to an allowance around the true splitting values of .5 times the standard deviation of the partitioning variable.

In addition, means and variances of the recovered partitioning variables were calculated. These were calculated and assessed separately for trees that were accurately, and trees that were not accurately recovered.

**Predictive accuracy**

We evaluated the predictive accuracy of lmtree and lmertree by calculating correlations between the predicted and true treatment-effect differences ($\beta_{j1}$, Figure 4.4) of test observations. This correlation can only be assessed for datasets with treatment-subgroup interactions, as the true predicted treatment differences are a constant in datasets without treatment-subgroup interactions.

Evaluation using the same data for training and testing a model results in overly optimistic estimates of predictive accuracy (Hastie et al., 2009). Therefore, the lmtrees and lmertrees were used for prediction of new observations in test datasets. Observations in these test datasets were drawn from the same populations as the training datasets. That is, the simulation design described above was used for both the training and test datasets. The cluster-specific intercepts $b_i$ were randomly generated for training as well as testing datasets, and therefore the observations in the test datasets were from 'new' clusters. Consequently, the random effects were not used for prediction, and only the linear model tree from the last iteration of the algorithm was used for prediction with lmertree.

For every dataset, two correlation coefficients were calculated, representing the linear association between the true and predicted treatment dif-
ferences: one for lmertree, and one for lmtree. To detect predictors of predictive accuracy of both algorithms, an lmtree was built, with maximum tree depth set to four. The outcome variable in this tree is the correlation between the true and predicted treatment differences. The predictor variable for the linear model is algorithm type (lmtree or lmertree), and the potential partitioning variables are the parameters of the data-generating process.

4.4 Simulation study: results

4.4.1 Tree structure recovery

Tree size in datasets with a main treatment effect only

In Table 4.1, tree sizes for lmtee and lmertree are presented for datasets with a main treatment effect only. Overall, smaller trees were created by lmertree, than by lmtree. The true tree size for datasets with a main treatment effect only was 1; the average size of the lmertrees was 1.09 (SD=0.44) and of the lmtrees 2.02 (SD=1.68). The Type I error, or the probability that the algorithm erroneously partitioned the dataset, was very small for lmertree (.04) and much larger for lmtree (.33).

In Figure 4.5, a linear model tree depicting the relationship between the various data-generating parameters and tree size is presented. The terminal nodes of Figure 4.5 show, that when the variance of the random effects is 0, both lmtree and lmertree tend to built trees of the right size (node 2 in Figure 4.5). However, when the variance of the random effects is greater than 0, lmtree tends to erroneously build trees of size > 1, whereas lmertree tends to correctly build trees of size 1 (nodes 5, 6, 8 and 9 in Figure 4.5). In addition, the size of lmtrees tends to be larger with a larger sample size (i.e., N = 1000; nodes 5 and 6 in Figure 4.5). Thus, lmtrees may erroneously pick up cluster effects as covariate effects.
Table 4.1: Tree size distributions for lmtree and lmertree for datasets with a main treatment effect only.

<table>
<thead>
<tr>
<th>tree size</th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>lmertree</td>
<td>20625</td>
<td>932</td>
<td>43</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>21,600</td>
</tr>
<tr>
<td></td>
<td>(.96)</td>
<td>(.04)</td>
<td>(&lt;.01)</td>
<td>(.00)</td>
<td>(.00)</td>
<td>(.00)</td>
<td>(1.00)</td>
</tr>
<tr>
<td>lmtree</td>
<td>14501</td>
<td>4202</td>
<td>2013</td>
<td>802</td>
<td>79</td>
<td>3</td>
<td>21,600</td>
</tr>
<tr>
<td></td>
<td>(.67)</td>
<td>(.20)</td>
<td>(.09)</td>
<td>(.04)</td>
<td>(&lt;.01)</td>
<td>(&lt;.01)</td>
<td>(1.00)</td>
</tr>
</tbody>
</table>

*Note.* Bracketed values are proportions. Tree sizes are expressed as the total number of nodes in the tree. A tree with a total of \(k\) nodes has \((k+1)/2\) terminal nodes; the true tree size in datasets with a main treatment effect only is 1.

**Tree size in datasets with treatment-subgroup interactions**

For datasets with treatment-subgroup interactions, lmertree also created smaller trees than lmtree did. For these datasets, the true tree size was 7 (4 terminal nodes and 3 inner nodes; Figure 4.4). The size of the trees resulting from application of lmtree and lmertree are presented in Table 4.2. The average size of lmertrees was 7.16 (SD=0.62), and the average size of lmtrees was 8.13 (SD=2.05). The estimated probability that the algorithm erroneously did not partition the dataset was 0, for both lmtree and lmertree. However, as Table 4.2 shows, 91% of lmertrees matched the true tree size, whereas only 63% of lmtrees matched the true tree size.

In Figure 4.6, a linear model tree is presented, depicting the relationship between the various data-generating parameters and tree size for datasets with treatment-subgroup interactions. The terminal nodes of Figure 4.6 indicate that in many situations lmtree and lmertree build about equally sized trees, with two notable exceptions: nodes 7 and 14.
Figure 4.5: Linear model tree of tree sizes for lmertree and Imertree for datasets with a main treatment effect only. The y-axes of the boxplots represent the total number of nodes in a tree (a tree with a total of \( k \) nodes has \( (k + 1)/2 \) terminal nodes). \( N \) = total sample size; \( \text{sigmabi} = \sigma_{b_i} \). Circles represent outliers (values below \( Q_1 - 1.5 \times \text{IQR} \) or above \( Q_3 + 1.5 \times \text{IQR} \), IQR = interquartile range).

Node 14 in Figure 4.6 represents a large majority (59.26%) of the simulated datasets with treatment-subgroup interactions, and shows that Imertree recovered the true tree size in nearly all of these datasets, whereas lmertree provided trees with more than 7 nodes in the majority of these datasets. The treesize distributions in node 14 indicate that the increased power provided by a larger sample size (\( N = 500 \) or \( N = 1000 \)) results in spurious splits by lmertree, when \( \sigma_{b_i} \) is non-zero and when \( b_i \) is correlated to one of the variables in \( U \). Because Imertree can more adequately deal with the additional variance in \( Y \) caused by non-zero values of \( \sigma_{b_i} \), the
Figure 4.6: Linear model tree of tree sizes for lmtry and lmmtree with treatment-subgroup interactions. The y-axes of the boxplots represent the total number of nodes in a tree (a tree with a total of $k$ nodes has $(k + 1)/2$ terminal nodes). $N =$ total sample size; $\text{cor}_{Ui} =$ correlation between $b_i$ and one of the $U$ variables; $\text{sig}_{mabi} = \sigma_{b_i}$; $np =$ number of potential partitioning variables; circles represent outliers (values below $Q_1 - 1.5 \times IQR$ or above $Q_3 + 1.5 \times IQR$, $IQR =$ interquartile range).
Table 4.2: Tree size distributions for lmtree and lmertree for datasets with treatment-subgroup interactions.

<table>
<thead>
<tr>
<th>Tree size</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>13</th>
<th>15</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>lmtree</td>
<td>2</td>
<td>281</td>
<td>2938</td>
<td>2675</td>
<td>83</td>
<td>1</td>
<td>0</td>
<td>32,400</td>
</tr>
<tr>
<td></td>
<td>(&lt;.01)</td>
<td>(&lt;.01)</td>
<td>(.91)</td>
<td>(.08)</td>
<td>(&lt;.01)</td>
<td>(&lt;.01)</td>
<td>(.00)</td>
<td>(1.00)</td>
</tr>
<tr>
<td>lmertree</td>
<td>139</td>
<td>955</td>
<td>20530</td>
<td>4565</td>
<td>4164</td>
<td>1619</td>
<td>428</td>
<td>32,400</td>
</tr>
<tr>
<td></td>
<td>(&lt;.01)</td>
<td>(.03)</td>
<td>(.63)</td>
<td>(.14)</td>
<td>(.13)</td>
<td>(.05)</td>
<td>(.01)</td>
<td>(1.00)</td>
</tr>
</tbody>
</table>

*Note.* Bracketed values are proportions. Tree sizes are expressed as the total number of nodes in the tree. A tree with a total of \( k \) nodes has \( (k + 1)/2 \) terminal nodes; the true tree size in datasets with treatment-subgroup interactions was 7.

The size of lmertrees seems not to be affected by the level of \( \sigma_{b_i} \).

Node 7 in Figure 4.6 indicates that with low sample size \( (N = 200) \), when \( b_i \) is not correlated to one of the \( U \) variables, and the variance of \( b_i \) is large, lmtree tends to grow smaller trees. Because lmtree cannot account for the variance in \( Y \) due to cluster-specific effects, lmtree has difficulty to detect partitions when \( \sigma_{b_i} \) is large and sample size is low. At the same time, the upper whisker of the boxplot of lmtree in node 7 reaches a value of 9, indicating that in many cases, lmtree may still detect spurious splits in small datasets, when \( \sigma_{b_i} \) is large. The number of nodes in lmertrees, however, seems not to be affected by low sample size and the level of \( \sigma_{b_i} \).

**Recovery of partitioning variables and values**

Overall, in recovering partitioning variables and values, lmertree performed better than lmtree. For the first split, lmertree always selected the right variable \( (U_2) \); lmtree selected a wrong variable \( (U_1) \) only once.
The true splitting value for \( U_2 \) was 30 (Figure 4.4), and the mean splitting value selected for the first split, involving \( U_2 \), was 29.94 for both lmer-tree and lmtree. However, lmtree showed somewhat higher variability in the recovered splitting value, than lmertree (SD = .154 and SD = .127, respectively).

Table 4.3: Summary statistics for splitting variables and values for accurately recovered trees

<table>
<thead>
<tr>
<th>variable</th>
<th>true value</th>
<th>#</th>
<th>M</th>
<th>SD</th>
<th>#</th>
<th>M</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_1 )</td>
<td>17</td>
<td>29,220</td>
<td>16.59</td>
<td>0.78</td>
<td>19,905</td>
<td>16.55</td>
<td>0.85</td>
</tr>
<tr>
<td>( U_2 )</td>
<td>30</td>
<td>29,220</td>
<td>29.94</td>
<td>0.13</td>
<td>19,905</td>
<td>29.94</td>
<td>0.15</td>
</tr>
<tr>
<td>( U_5 )</td>
<td>63</td>
<td>29,220</td>
<td>63.13</td>
<td>0.79</td>
<td>19,905</td>
<td>63.17</td>
<td>0.86</td>
</tr>
</tbody>
</table>

*Note.* # refers to the number of splits involving a variable; M and SD refer to the mean and standard deviation of the recovered splitting value for a variable.

In Table 4.3, summary statistics for the partitioning values in accurately recovered trees are presented. Accurately recovered trees were defined as trees with three splits, with the first split involving \( U_2 \) with values 30 ± 5, the second split involving \( U_1 \) with values 17 ± 5 and the third split involving \( U_5 \) with values 63 ± 5 (Figure 4.4). As Table 4.3 shows, lmertree accurately recovered the tree in 29,220 out of 32,400 (90.19%) datasets, which is much more often than lmtree, which accurately recovered the tree in 19,905 out of 32,400 (61.43%) datasets. For both algorithms, the recovered splitting values in accurately recovered trees were quite close to the true values. However, the splitting values recovered by lmtree deviated somewhat more from the true values and showed somewhat higher variances, than the splitting values recovered by lmertree (Table 4.3).
Table 4.4: Summary statistics for splitting variables and values in trees that were not accurately recovered

<table>
<thead>
<tr>
<th>var.</th>
<th>value</th>
<th>#</th>
<th>M</th>
<th>SD</th>
<th>#</th>
<th>M</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_1$</td>
<td>17</td>
<td>3,391</td>
<td>15.85</td>
<td>3.40</td>
<td>14,708</td>
<td>14.99</td>
<td>4.48</td>
</tr>
<tr>
<td>$U_2$</td>
<td>30</td>
<td>3,586</td>
<td>29.92</td>
<td>2.76</td>
<td>14,352</td>
<td>29.94</td>
<td>3.42</td>
</tr>
<tr>
<td>$U_3$-$U_4$</td>
<td>-</td>
<td>856</td>
<td>-</td>
<td>-</td>
<td>11,265</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$U_5$</td>
<td>63</td>
<td>3,416</td>
<td>63.94</td>
<td>3.50</td>
<td>14,621</td>
<td>64.81</td>
<td>5.64</td>
</tr>
<tr>
<td>$U_6$-$U_{15}$</td>
<td>-</td>
<td>850</td>
<td>-</td>
<td>-</td>
<td>768</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Note. # refers to the number of splits involving a variable; M and SD refer to the mean and standard deviation of the recovered splitting value for a variable. The total number of trees that were not accurately recovered was 3,180 for lmertree, and 12,495 for lmtree. Because the total number of nodes in inaccurate trees can range from 3 to 15, the number of times a variable appears in trees may exceed the number of trees. True partitioning variables are $U_1$, $U_2$ and $U_5$; other potential partitioning variables do not have true partitioning values.

In Table 4.4, summary statistics for the partitioning variables and values appearing in trees that were not accurately recovered are presented. Variables that were not true partitioning variables appeared much more often in inaccurate lmtrees (in 12,033 out of 55,714 splits), than in inaccurate lmertrees (in 1,706 out of 12,099 splits). Also, the recovered partitioning values for the true partitioning variables showed higher variance for lmtree than for lmertree, which was also found for the accurately recovered trees.
4.4.2 Predictive accuracy on test datasets

Overall, treatment differences as predicted by lmerTree were closer to the true differences than those predicted by lmtree. The average correlation between the true and predicted treatment differences over all 32,400 datasets was .88 (SD=0.19) for the trees generated by lmtree, and .94 (SD=0.11) for the trees generated by lmerTree.

The linear model tree depicting the relationship between the various data-generating parameters and predictive accuracy of lmtree and lmerTree is presented in Figure 4.7, showing that lmerTree outperforms lmtree, in general. The dependent variable in this tree is the correlation between the true treatment differences, and the predictions of lmtree and lmerTree.

Figure 4.7 shows clear main effects of sample size $N$, magnitude of treatment differences $d$, and variance of random intercepts $\sigma_{b_i}$: when $N$ or $d$ increase, the accuracy of predictions of both lmtree and lmerTree increases. At the same time, the difference in performance between lmtree and lmerTree decreases, when $N$ and $d$ increase. The variance of the random intercepts influences the accuracy of lmtree and lmerTree in the reverse direction: When $\sigma_{b_i}$ increases, the accuracy of both lmtree and lmerTree decreases. However, the performance of lmtree is much stronger influenced by the magnitude of $\sigma_{b_i}$: When the variance of the random intercepts is large (i.e., $\sigma_{b_i} = 10$), lmtree is clearly outperformed by lmerTree.

As the boxplots in the terminal nodes of the tree in Figure 4.7 show, for some simulated datasets, correlations between predicted and true differences were clear outliers. Extreme outliers (i.e., correlations < 0) were more often found for lmtree (298 out of 32,400 datasets; 0.92%), than for lmerTree (38 out of 32,400 datasets; 0.12%). As Figure 4.7 shows, extreme outliers were observed almost exclusively when treatment differences were small. Obviously, in these cases, due to a low signal-to-noise ratio, selection of a wrong splitting variable is more likely, in rare
Figure 4.7: Linear model tree of correlations with true treatment differences for lmertree and lmtries. N = total sample size; treatdiff = $\beta_1$: the unstandardized mean difference in treatment outcomes in subgroups with differential effects for treatment 1 and treatment 2; sigmabi = $\sigma_{b_i}$; circles represent outliers (values below $Q_1 - 1.5 \times IQR$ or above $Q_3 + 1.5 \times IQR$; IQR = interquartile range).
cases even resulting in predictions that are negatively correlated to the true treatment differences.

In addition to the correlations between true and predicted treatment differences, we assessed potential bias in the predicted treatment differences. The mean true treatment difference was 0 (Figure 4.4). The mean treatment difference, averaged over all test datasets, as estimated by lmertree was $-0.029$ (SD=0.64), and as estimated by lmtree was $-0.026$ (SD=0.95). This was only a small deviation from the expected value of 0, when compared to the spread of the distribution of treatment differences: The standard deviation of the true treatment differences, averaged over all test datasets, was 3.34. The average standard deviation of the predicted treatment differences was 3.49 for lmertree, and 3.66 for lmtree. Thus, on average, the means of the predicted treatment differences of lmtree and lmertree deviate only slightly from their expected value of 0, indicating that predicted treatment differences are unbiased indicators of the true treatment differences.

It is also important to ensure whether the estimation of random effects does not negatively affect accuracy, when random effects are absent from the dataset. In datasets where $\sigma_{b_i} = 0$, the average correlation between the true and predicted treatment differences of lmtree and of lmertree were identical up to the third decimal: the average correlation was .9382 (SD=.10) for lmertree and .9385 for lmtree (SD=.10).

### 4.5 Application to real data

#### 4.5.1 Method

To illustrate the application and the potential differences in the results of lmtree and lmertree, we applied both algorithms to a dataset from a meta-analytic study of Cuijpers et al. (2014). This meta-analysis was based on
individual-patient data of 1,766 observations from 14 RCTs, comparing
the effects of psychotherapy (cognitive behavior therapy; CBT) and phar-
macotherapy in the treatment of depression. The study of Cuijpers et al.
(2014) was aimed at establishing whether gender is a predictor or mod-
erator 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 either predicted or
moderated treatment outcomes. Further details on the dataset are pro-
vided in Cuijpers et al. (2014).

In our analyses, posttreatment HAM-D score was the outcome variable,
and potential partitioning variables were age, gender, level of educa-
tion, presence of a comorbid anxiety disorder at baseline, and pretreat-
ment HAM-D score. The linear predictor was treatment type (0=CBT
and 1=pharmacotherapy). An indicator for study was used as the cluster
indicator.

In RCTs, treatment effects are often estimated, after controlling posttreat-
ment values on the outcome measure for the linear effect of pretreatment
values on the same measure. Therefore, we included the predictions of a
linear regression of HAM-D posttreatment on HAM-D pretreatment sco-
res, as an offset variable in all models. An offset variable is a linear pre-
dictor with an a-priori determined coefficient of one. Including the linear
regression predictions as an offset has the same effect as statistically con-
trolling for the linear effects of a of pretreatment scores, as is often done
in ANCOVA.

The lmtree function deals with missing data by listwise deletion. There-
fore, we build the trees using data of a subset of 694 patients, as complete
observations (i.e., observations with non-missing values for potential par-
titioning variables, and pre- and posttreatment HAM-D score) for these
patients were available. Results of our analysis may therefore not be rep-
resentative of the complete dataset of the meta-analysis by (Cuijpers et
Predictive accuracy of lmtree and lmertree was assessed by calculating the average correlations between observed and predicted HAM-D scores, based on 50-fold cross validation.

### 4.5.2 Results

We applied lmtree and lmertree to a dataset from the individual-patient data meta-analysis of Cuijpers et al. (2014). The outcome variable in the trees is posttreatment Hamilton Rating Scale for Depression (HAM-D) score ($M=8.41$, $SD=6.26$), controlled for the linear effects of the pretreatment HAM-D score. The resulting lmtree and lmertree are presented in Figure 4.8 and 4.9. Note that the lmtree is also the tree that is created in the first iteration of the lmertree algorithm.

The lmtree (Figure 4.8) 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. Terminal node 2 of Figure 4.8 indicates that for patients with a low level of education, antidepressant medication provides the greatest reduction in HAM-D scores. Terminal node 4 indicates that for patients with a higher level of education, and no comorbid anxiety disorder, the reduction in HAM-D scores is about the same for CBT and antidepressant mediation. Terminal node 5 indicates, that for patients with a higher level of education, and a comorbid anxiety disorder, the reduction in HAM-D scores is greatest for antidepressant medication.

By taking into account the study-specific intercepts, the final lmertree (Figure 4.9) indicates that the first split made by lmtree is a spurious split. The lmertree selected only presence of a comorbid anxiety disorder as a partitioning variable. The terminal nodes of Figure 4.9 show only a single treatment-subgroup interaction: for patients without a comorbid anxiety disorder, antidepressant medication provides the greatest reduction in HAM-D scores.
Figure 4.8: Linear model tree for prediction of posttreatment total scores on the Hamilton Rating Scale for Depression (HAM-D). The y-axes of the boxplots represent posttreatment HAM-D scores, and the x-axes represent treatment levels: cognitive behavior therapy (CBT) vs. pharmacotherapy (PHA).

Disorder, CBT and antidepressant medication provide more or less the same reduction in HAM-D scores, whereas for patients with a comorbid anxiety disorder, antidepressant medication provides a greater reduction in HAM-D scores than CBT. The estimated variance of the random intercept term was 2.12, with an estimated intraclass correlation coefficient of .054.

Assessment of predictive accuracy by means of 50-fold cross validation showed that lmertree had higher predictive accuracy than lmtree. The correlation between true and predicted posttreatment HAM-D total scores, averaged over the 50 folds, was .39 (SD=.20) for lmertree, and .31 (SD=.24) for lmtree. This indicates that lmertree not only provided higher predictive accuracy, on average, but also had lower variability of
Figure 4.9: Linear mixed-effects regression tree for prediction of posttreatment total scores on the Hamilton Rating Scale for Depression (HAM-D). The y-axes of the boxplots represent posttreatment HAM-D scores, and the x-axes represent treatment levels (cognitive behavior therapy (CBT) vs. pharmacotherapy (PHA)).

predictive accuracy than lmtree.

4.6 Discussion

The results of our simulation study show that lmertree performed very well in recovering treatment-subgroup interactions. The lmertree algorithm was found to have a Type I error rate of 4%, whereas trees without random-effects estimation were found to have a Type I error rate of 33%. Further, the lmertree algorithm recovered the true interactions in 90% of the datasets with treatment-subgroup interactions, whereas trees without random effects recovered the true treatment-subgroup interactions in only 61% of those datasets. Also, lmertree predicted treatment outcome differences with an accuracy of .94, whereas trees without random effects
showed an accuracy of .88, on average.

The better performance of lmertree was mostly observed when random effects in the datasets were sizable, and random intercepts were correlated with potential partitioning variables. In these instances, the random effects gave rise to spurious subgroup detection (spurious splits), in the trees resulting from application of lmtree, both in datasets with and without treatment-subgroup interactions.

As expected, when random effects were absent from the simulated datasets, lmtree and lmertree performed equally well. This finding indicates that lmertree can be applied, whenever cluster-specific random effects are expected: In the absence of random effects, lmtree and lmertree are expected to perform equally well, and in the presence of random effects, lmertree will outperform lmtree. This seems especially to be the case for large sample sizes ($N > 500$), as the increased power will likely lead lmtree to create spurious splits in the presence of random effects.

Not surprisingly, accuracy of predicted treatment of both algorithms deteriorated when sample size was low (i.e., $N = 200$). Sample size influenced performance of lmtree and lmertree similarly, suggesting that the larger number of estimated parameters for lmertree did adversely influence accuracy with low sample sizes. However, our simulation results do warrant caution for the detection of treatment-subgroup interactions or treatment moderators in small datasets (e.g., single RCTs), irrespective of the algorithm used.

Application of lmtree and lmertree to the dataset of the individual-patient data meta-analysis of Cuijpers et al. (2014), showed that lmertree provided better predictive accuracy, and that ignoring the clustered structure of the dataset by application of lmtree may result in erroneous conclusions about treatment-subgroup interactions. When random effects were ignored, level of education was found to be involved in treatment interaction, in addition to the presence of a comorbid anxiety disorder.
However, application of lmertree indicated that level of education is not a moderator of treatment effect, and only the presence of a comorbid anxiety disorder is.

Although these findings are encouraging for the use of lmertree in the detection of treatment-subgroup interactions in datasets with clustered structures, some limitations of our study and challenges for future research should be noted.

As noted in the Introduction, simulations in the current study were confined to random-intercept models. Although lmertree allows for the estimation of random slopes as well, we did not include random slopes in our simulation study. In addition, random slope estimation would raise the question of whether random treatment effects should be estimated in the model, as the treatment effects are already estimated with local linear fixed-effects models.

Our simulations also did not include models with multiple fixed-effects predictor variables in $X$. Multiple fixed-effects predictor variables can be easily included in lmtree and lmertree, but it should be noted that the parameters corresponding to these variables will then be included in tests for parameter instability as well, which may be undesirable. For example, in RCTs, ANCOVA will often be used to control for the linear effects of pretreatment values on the treatment outcome variable. Whether or not such parameters should be included in parameter stability tests, or should be allowed to vary over partitions, should be decided by the researcher.

One challenge for further research is the development of parameter stability tests for random-effects parameters. In the current study, random-effects parameters were estimated globally, using all observations in the dataset, and fixed-effects parameters were estimated locally, using the observations in a single node. This would allow, for example, for estimation of random treatment effects, instead of fixed treatment effects.

A second challenge is the development of more adequate ways to deal
with missing data in treatment-subgroup interaction detection. lmtree, like all tree-based algorithms for treatment-subgroup interaction detection, handles missing data by listwise deletion. However, missing data commonly occurs in clinical trails, and listwise deletion is obviously not the preferred method for dealing with missing data (e.g., Wood, White & Thompson, 2004).

In conclusion, lmtree provided highly accurate recovery of treatment-subgroup interactions and predictions of treatments differences, in the presence and absence of cluster-specific random effects. Therefore, lmtree is a promising algorithm for the detection of treatment-subgroup interactions in datasets with a clustered structure, like for example in multi-center trials, individual-level patient data meta-analyses, and longitudinal studies.
4.7 Appendix: Notation

1, ..., i, ..., n denotes cluster number
1, ..., j, ..., t denotes terminal node number in a tree
1, ..., k, ..., r denotes partitioning variable number
\( \beta \) \( 2t \times 1 \) column vector \( [\beta_1^T, ..., \beta_j^T, ..., \beta_t^T]^T \) of fixed-effects coefficients
\( \beta_j \) \( 2 \times 1 \) column vector of fixed-effects coefficients in terminal node \( j \)
\( b \) \( n \times 1 \) column vector \( [b_1, ..., b_i, ..., b_n]^T \) of random intercepts
\( b_i \) random intercept in cluster \( i \)
\( d \) effect size; standardized mean difference in outcome \( Y \) between treatments 1 and 2
\( \epsilon \) \( N \times 1 \) column vector of residuals
\( N \) total number of observations
\( N_i \) number of observations in cluster \( i \)
\( N_j \) number of observations in terminal node \( j \)
\( \sigma_{b_i} \) square root of variance of \( b_i \)
\( \sigma_\epsilon \) square root of the variance of \( \epsilon \)
\( U \) \( N \times r \) matrix of (potential) partitioning variables
\( X \) \( N \times 2t \) (block diagonal) design matrix of fixed-effects predictor variable values
\( y \) \( N \times 1 \) column vector of response variable values
\( Z \) \( N \times n \) (block diagonal) design matrix of random-effects predictor variable values