HEC MONTRÉAL
École affiliée à l'Université de Montréal

Essays on covariate-based canonical correlation estimation, prediction intervals, and covariance regression with random forests

par
Cansu Alakus

Thèse présentée en vue de l'obtention du grade de Ph. D. en administration
(spécialisation Science des données)

Décembre 2022

HEC MONTRÉAL

École affiliée à l'Université de Montréal

Cette thèse intitulée :

Essays on covariate-based canonical correlation estimation, prediction intervals, and covariance regression with random forests

Présentée par :

Cansu Alakus

a été évaluée par un jury composé des personnes suivantes :

Marc Fredette
HEC Montréal
Président-rapporteur

Denis Larocque
HEC Montréal
Directeur de recherche

Aurélie Labbe
HEC Montréal
Codirectrice de recherche

Juliana Schulz
HEC Montréal
Membre du jury

Ruoqing Zhu
University of Illinois Urbana-Champaign
Examinateur externe

Guy Paré
HEC Montréal
Représentant du directeur de HEC Montréal

Résumé

Les forêts aléatoires peuvent être utilisées pour former l'ensemble des plus proches voisins pour l'observation que nous voulons prédire. Nous utilisons cette caractéristique des forêts aléatoires pour proposer de nouvelles méthodes pour différents problèmes dans les trois chapitres de cette thèse. Dans le premier chapitre, nous nous concentrons sur l'analyse de corrélation canonique, qui est une méthode statistique pour étudier la relation entre deux ensembles de variables, et en particulier, sur son extension pour prendre en considération l'effet de covariables liées au sujet sur la corrélation. Nous proposons une nouvelle méthode appelée Random Forest with Canonical Correlation Analysis (RFCCA) pour estimer la corrélation canonique conditionnelle entre deux ensembles de données multivariées étant donné des covariables liées au sujet. Après avoir développé les arbres individuels avec une règle de division spécialement conçue pour trouver des sous-groupes d'observations avec des corrélations canoniques distinctes, pour une nouvelle observation, nous formons l'ensemble des plus proches voisins de la forêt et estimons la corrélation canonique à l'aide des observations de cet ensemble. Dans le deuxième chapitre, nous intégrons l'idée d'utiliser les observations du plus proche voisin pour estimer la distribution de l'erreur de prédiction conditionnelle pour une variable réponse continue à la méthode de la forêt boostée en une étape qui produit des prédictions ponctuelles avec correction du biais. Nous proposons une nouvelle méthode pour construire pour construire les intervalles nommée Prediction Intervals with Boosted Forests (PIBF). Après avoir construit une forêt boostée, pour une nouvelle observation, nous obtenons la prédiction ponctuelle avec correction du biais et construisons l'intervalle de prédiction pour

cette prédiction ponctuelle en utilisant l'ensemble des voisins les plus proches. Dans le troisième chapitre, nous nous concentrons sur l'estimation de la matrice de covariance conditionnelle pour un vecteur de réponse multivariée compte tenu de l'ensemble de covariables et sur l'analyse de la variation de ces matrices de covariance conditionnelle par rapport aux covariables. Nous proposons une nouvelle méthode appelée Covariance Regression with Random Forests (CovRegRF) pour estimer la matrice de covariance conditionnelle d'une réponse multivariée à l'aide de forêts aléatoires. Les arbres de la forêt sont construits avec une règle de division spécialement conçue pour trouver des sous-groupes d'observations avec des matrices de covariance distinctes. Ensuite, pour une nouvelle observation, nous trouvons les observations des voisins les plus proches à l'aide de la forêt et estimons la matrice de covariance à l'aide de ces observations. Les méthodes proposées présentées dans cette thèse sont évaluées par des études de simulation, et leurs performances surpassent généralement celles des méthodes concurrentes. Les trois méthodes sont implémentées dans des packages R librement disponibles sur CRAN et peuvent être très utiles dans la pratique.

Mots-clés

Forêt aléatoire, analyse de corrélation canonique, intervalle de prédiction, forêt boostée, régression de covariance, règle de division, test d'hypothèse, importance des variables.

Méthodes de recherche

Exploration de données, intégration de données, analyse multivariée.

Abstract

Random forests can be used to form the set of nearest neighbours for the observation we want to predict. We utilise this feature of random forests to propose new methods for different problems in the three chapters of this thesis. In the first chapter, we focus on canonical correlation analysis, which is a statistical method to investigate the relationship between two sets of variables, and in particular, on extending it to take into consideration a subject-related covariate effect on the correlation. We propose a new method called Random Forest with Canonical Correlation Analysis (RFCCA) to estimate the conditional canonical correlation between two multivariate data sets given subject-related covariates. After growing the individual trees with a splitting rule specifically designed to find subgroups of observations with distinct canonical correlations, for a new observation, we form the set of nearest neighbours from the forest and estimate the canonical correlation using the observations in this set. In the second chapter, we integrate the idea of using the nearest neighbour observations for estimating the conditional prediction error distribution for a continuous response variable to the one-step boosted forest method which produces bias-corrected point predictions. We propose a new method to build Prediction Intervals with Boosted Forests (PIBF). After training a boosted forest, for a new observation, we predict the bias-corrected point prediction and build the prediction interval for that point prediction using the set of nearest neighbours. In the third chapter, we focus on estimating the conditional covariance matrix for a multivariate response vector given the set of covariates and analyzing how these conditional covariance matrices vary with respect to the covariates. We propose a new method called Covariance Regression with

Random Forests (CovRegRF) to estimate the conditional covariance matrix of a multivariate response using random forests. The trees in the forest are built with a splitting rule specifically designed to find subgroups of observations with distinct covariance matrices. Then, for a new observation, we find the nearest neighbour observations using the forest and estimate the covariance matrix using those observations. The proposed methods presented in this thesis are evaluated through simulation studies, and they generally outperform their current competing methods. All three methods are implemented in freely available R packages on CRAN and can be very useful in practice.

Keywords

Random forest, canonical correlation analysis, prediction interval, boosted forest, covariance regression, splitting rule, significance test, variable importance.

Research methods

Data mining, data integration, multivariate analysis.

Contents

List of Tables

List of Figures

Acknowledgements

Pursuing a doctoral degree is a long and hard journey that is full of challenges and surprises. My journey has been made significantly easier by the support, guidance and love I received from my supervisors, friends and family along the way.

First and foremost, I would like to express my deepest gratitude to my research supervisors Denis Larocque and Aurélie Labbe, for their continuous support, guidance and encouragement. You were always there at the critical points of my research, and you were always very kind and positive. I have felt very lucky throughout my studies to work with you. I will miss our weekly meetings and discussions very much!

I would also like to thank my other co-authors, especially Sarah Lippé for sharing her knowledge and contributing to this work. I am also grateful to my committee members, Ahlem Hajjem, Juliana Schulz, Marc Fredette and Ruoqing Zhu, for the time they spent reviewing my thesis and for their very helpful comments and suggestions. I want to thank the HEC Montreal Ph.D. Program, especially Nathalie Bilodeau for her overall support and guidance through the technical difficulties involved in Ph.D. life.

I want to thank my friends, Berk, Burak, Elçin and Sena. My favourite memories from my time in Montreal involve our times together. I would like to express my special thanks to my parents and my sister, for their unconditional love and support throughout my endless studies.

To finish, a special word for my best friend and my joy, Barış. A million thanks to you for sharing this journey with me and always being there for me.

General Introduction

Random forest (Breiman, 2001) is a widely used ensemble method for statistical learning. Random forests were introduced as a way to get predictions by averaging the predictions from many decision trees and are typically studied as a useful method for non-parametric conditional mean estimation, *i.e.* $E[Y|X = x] = \mu(x)$ where Y is a real-valued response variable and X is a vector of predictors. Besides this traditional view, random forests can be seen as a way to find nearest neighbour observations that are close to the one we want to predict. This modern view of random forests is utilised to propose novel methods for several problems in the following three chapters of this thesis. In all three chapters, after building the forest with the training data, for a new observation, we find the set of nearest neighbours and use this set to obtain the final estimation.

The first and third chapters share the concept of estimating a conditional measure given a set of covariates using a random forest. For the first chapter, this conditional measure is the canonical correlation, while for the third, it is the covariance matrix. In both chapters, individual trees in the forest are built with a splitting rule specially designed to maximize the difference between the measure of interest of child nodes which enables to find subgroups of observations with distinct canonical correlations or covariance matrices. Different from those two chapters, the second chapter investigates building prediction intervals with random forests.

The first chapter focuses on multi-view data integration using random forests. Multi-view data denotes many kinds of data that provide information about a subject from multiple sources. Developing integrative statistical and machine learning models for better

use of information from multi-view data is a challenging and important step to deepening our understanding of biological systems. The relationship between two multivariate data sets can be investigated with canonical correlation analysis (CCA) which is a two-view data integration tool. CCA assumes that the correlation between the two sets of variables is constant for all subjects. However, in practice, the correlation between the two sets can sometimes depend on a third set of covariates, often subject-related ones such as age or gender. In this case, applying CCA to the whole population is not optimal and the subject-related covariates should be accounted for in the analyses. We propose a new method called Random Forest with Canonical Correlation Analysis (RFCCA) to estimate the conditional canonical correlations between two sets of variables given subject-related covariates. The individual decision trees in the forest are built with a splitting rule specially designed to partition the data to find subgroups of subjects with distinct canonical correlations. Once the random forest is built, for a new observation, the set of nearest neighbours is formed, and the canonical correlation is computed using these nearest neighbours. The proposed method assumes that the subject-related covariates are indeed relevant to estimating the conditional canonical correlations of subjects which may not always be true. Therefore, we propose a significance test to detect the global effect of the subject-related covariates on the canonical correlations. The performance of the proposed method and the global significance test is evaluated through simulation studies which show accurate canonical correlation estimations and well-controlled Type-1 error. We also propose a way to compute variable importance measures of the subject-related covariates. We demonstrate an application of the significance test and proposed method with EEG data.

In the second chapter, we investigate building prediction intervals with random forests. In predictive modeling, the main goal is to provide a point prediction for a new observation. However, a point prediction does not include information about its reliability. In the decision-making context, assessing and quantifying the uncertainty in the prediction is also important, and can be achieved with a prediction interval. In regression analysis, which is a form of predictive modeling, random forests provide point predictions for a

new observation, i.e. conditional mean of the response given the covariates. Random forests reduce over-fitting by combining many decision trees. However, final predictions can be still biased because each tree in the forest is built under the same random process and captures the same part of the response signal, so weaker parts of the response signal may be left untargeted. Ghosal and Hooker (2021) propose a bias correction method for regression forests called *one-step boosted forest*. The main idea of this method is to predict the bias of the prediction, which is the out-of-bag residuals of the point predictions obtained with the first random forest, using a second random forest. The final prediction for a new observation is then the sum of the predictions from two random forests. We propose a new method to build Prediction Intervals with Boosted Forests (PIBF). In the proposed method, besides using the second random forest for bias correction, we use it to form the set of nearest neighbours for a new observation. Then, we build the prediction interval for a new observation by estimating the conditional prediction error distribution from that neighbourhood. We compare the performance of the proposed method to ten existing methods for building prediction intervals with random forests through an extensive simulation study and real data analyses that show it globally outperforms competing methods.

In the third chapter, we study estimating the conditional covariance matrix of a multivariate response vector given a set of covariates with random forests. For most of the existing multivariate regression analysis methods, the main goal is to estimate the conditional mean of the response vector based on its covariates. Usually, these methods assume a constant covariance matrix for the response vector for all subjects. However, this assumption may not be valid for some data sets in practice. Therefore, estimating the conditional covariances amongst multiple response variables given a set of covariates is also important. We propose a new method called Covariance Regression with Random Forests (CovRegRF) to estimate the conditional covariance matrix of a multivariate response using random forests and to analyze how these conditional covariance matrices vary with respect to the covariates. Each tree in the forest is built with a splitting rule specially designed to find subgroups of subjects with distinct covariance matrices. Sim-

ilar to what we do in the first chapter, for a new observation, we form the set of nearest neighbour observations using the forest and we estimate the conditional covariance matrix using these nearest neighbours. Like RFCCA, the proposed method assumes that the set of covariates is important to identify the subgroups of subjects with similar covariance matrices, but some or all covariates may not be relevant. Hence, we propose a significance test to evaluate the effect of a subset of covariates on the covariance matrix estimates while controlling for the other covariates. We perform a simulation study to evaluate the performance of the proposed method and the significance test. Like in the first chapter, we propose a way to compute the variable importance of covariates. We show an application of the significance test and proposed method with a thyroid disease data set.

Chapter 1

Conditional canonical correlation estimation based on covariates with random forests

Abstract

Investigating the relationships between two sets of variables helps to understand their interactions and can be done with canonical correlation analysis (CCA). However, the correlation between the two sets can sometimes depend on a third set of covariates, often subject-related ones such as age, gender, or other clinical measures. In this case, applying CCA to the whole population is not optimal and methods to estimate conditional CCA, given the covariates, can be useful. We propose a new method called Random Forest with Canonical Correlation Analysis (RFCCA) to estimate the conditional canonical correlations between two sets of variables given subject-related covariates. The individual trees in the forest are built with a splitting rule specifically designed to partition the data to maximize the canonical correlation heterogeneity between child nodes. We also propose a significance test to detect the global effect of the covariates on the relationship between two sets of variables. The performance of the proposed method and the global significance

test is evaluated through simulation studies that show it provides accurate canonical correlation estimations and well-controlled Type-1 error. We also show an application of the proposed method with EEG data.

1.1 Introduction

Data from multiple sources, called multi-view data, refers to many types of data that include complementary information from different aspects to characterize a subject. For example, in biomedical studies, the collection of data may include subject-related covariates (e.g. age, gender, medical history), DNA sequencing, transcriptomics (e.g. mRNA, microRNA, RNA sequencing) and proteomics for a single subject (Cancer Genome Atlas Network, 2012; ENCODE Project Consortium, 2012). As another example, in functional neuroimaging, we may have subject-related covariates (e.g. age, gender, intracranial volume (ICV)), brain imaging data, and cognitive measurements for subjects (Fratello et al., 2017). Integration of multiple feature sets and investigating the relationships between them may help to understand their interactions and obtain more meaningful interpretations. Studying the integration of multiple feature sets requires statistical and machine-learning tools which include methods for dimension reduction, clustering, classification, and association studies for multi-view data integration (see Sun (2013); Meng et al. (2016); Min et al. (2017); Li et al. (2018) for comprehensive reviews).

Canonical correlation analysis (CCA), firstly introduced in Hotelling (1936), is a multivariate statistical method that analyzes the relationship between two multivariate data sets, X and Y. CCA searches for linear combinations of each of the two data sets, Xa and Yb, having maximum correlation. In CCA, the components Xa and Yb are called canonical variates and their correlations are the canonical correlations. CCA is a two-view data integration tool. It was later generalized to data with more than two views (Kettenring, 1971). There are some extensions of CCA for under-determined data sets through regularized CCA (Vinod, 1976; Cruz-Cano and Lee, 2014), for sparse data sets through sparse-CCA (Witten et al., 2009; Hardoon and Shawe-Taylor, 2011) and for nonlinear

relationships through generalized CCA (Melzer et al., 2001), deep CCA (Andrew et al., 2013), kernel CCA (KCCA) (Akaho, 2001) and non-parametric CCA (NCCA) (Michaeli et al., 2016). Although very flexible, all these methods suppose that the relationship between the two sets of variables is constant for all subjects, which is not always the case in practice. For example, hundred of gene-environment studies have shown that gene effects on diseases are modulated by environmental factors (Caspi and Moffitt, 2006; Hunter, 2005; Ma et al., 2011). In the field of neuroscience, age and gender are known to interact with brain-behaviour correlations (Davis et al., 2008; Li et al., 2010) and this should be accounted for in the analyses. In this paper, we focus on CCA and specifically on extending it to account for a subject-related covariate effect on the correlation.

There are several ways to account for subject-related covariates in multi-view data integration. They can be treated as one of the views (Hanna et al., 2010; Li and Jung, 2017; Moser et al., 2018; Mihalik et al., 2020) or they can be used to identify subgroups in the data while analyzing the relationships between other views. Recently, Choi et al. (2020) proposed a recursive partitioning approach, namely correlation tree, to identify homogeneous correlated subgroups within data. In their work, the data consists of a set of subject-related covariates Z (e.g. age, gender, education) and two univariate continuous variables X and Y which are assumed to be correlated. The correlation tree method grows a decision tree with covariates to identify subgroups of subjects with different correlations between the two univariate variables. A simple illustration of this approach is shown in Figure 1.1 with a single split of the decision tree (see Appendix A for more examples). In this example, the overall correlation between X and Y is $\rho = 0.329$. However, this hides the fact that the subgroup with $Z_1 > 0.011$ has a much higher correlation of $\rho_R = 0.741$ while the other subgroup $Z_1 \leq 0.011$ has almost no correlation ($\rho_L = 0.018$). Such a situation can be modeled in practice with a regression model including an interaction effect between X and Z, but the situation is not as straightforward when both Y and X are multivariate and when the interaction pattern with Z is complex enough to not be captured efficiently by a single tree. Therefore, the goal of this paper is to propose a novel way to estimate the canonical correlations between two sets of multivariate variables X and Y,

depending on Z, using a random forests framework.

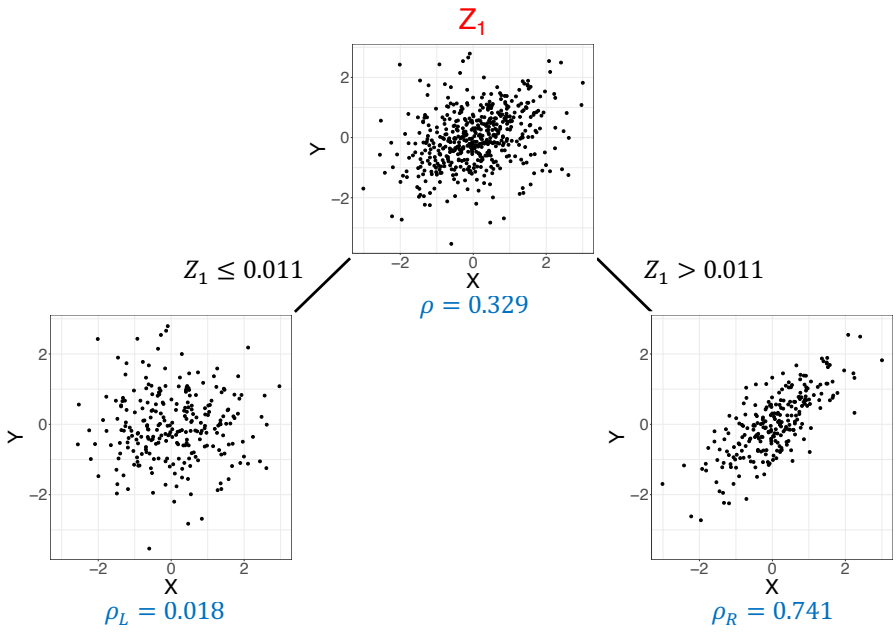

Figure 1.1: Example of a dataset (X,Y) where the correlation between X and Y depends on a covariate Z_1. This can be captured by a single split of the decision tree.

Random forest is an ensemble method which contains many decision trees. It is a powerful prediction method due to its ability to limit over-fitting. Moreover, random forests can be seen as a way to find nearest neighbor observations that are close to the one we want to predict (Hothorn et al., 2004; Lin and Jeon, 2006; Moradian et al., 2017, 2019; Roy and Larocque, 2020; Tabib and Larocque, 2020). Each tree in the proposed random forest framework is built with a new splitting rule designed to produce child nodes with maximum difference in the canonical correlation between X and Y. For a new observation with subject-related covariate values z^*, the proposed random forest provides a set of similar observations from the training data set that will be used to compute a canonical correlation estimate given z^*. Moreover, we propose a significance test to detect the global effect of Z on the relationship between X and Y.

This paper is organized as follows. In Section 1.2, we describe the proposed method, global significance test and variable importance measure. In Section 1.3, simulation study results for accuracy evaluation and global effect of covariates are presented to show the

performance of the method. A real data example is provided in Section 1.4, followed by concluding remarks in Section 1.5.

1.2 Proposed method

In this section, we describe the proposed method in detail.

1.2.1 Canonical correlation analysis (CCA)

Canonical correlation analysis (CCA), firstly introduced in Hotelling (1936), seeks vectors of $a \in R^p$ and $b \in R^q$, given two mean centered multivariate data sets $X \in R^{n \times p}$ and $Y \in R^{n \times q}$, such that Xa and Yb are maximally linearly correlated. We can formulate the problem as finding the coefficients (a^*, b^*) such that:

$$(a^*, b^*) = \underset{a,b}{\operatorname{argmax}}\, corr(Xa, Yb), \tag{1.1}$$

where

$$corr(Xa, Yb) = \frac{a^T \Sigma_{XY} b}{\sqrt{a^T \Sigma_{XX} a}\sqrt{b^T \Sigma_{YY} b}},$$

where Σ_{XX} and Σ_{YY} are the covariance matrices of X and Y, respectively, and Σ_{XY} is the cross-covariance matrix. Since rescaling a and b does not affect $corr(Xa, Yb)$, we can add the constraints $a^T \Sigma_{XX} a = 1$, $b^T \Sigma_{YY} b = 1$ to the maximization problem (1.1). There are several ways to solve the CCA problem, such as solving standard or generalized eigenvalue problems (Hotelling, 1936; Hardoon et al., 2004; Bach and Jordan, 2002), using alternating least squares regression (Branco et al., 2005; Wilms and Croux, 2015) and using singular value decomposition (Healy, 1957; Ewerbring and Luk, 1989). See Appendix B for the details.

1.2.2 Tree growing process

We use an unsupervised random forest based on the set of covariates Z to find subgroups of observations with similar canonical correlations between X and Y. This random forest

consists of many unsupervised decision trees with a specialized splitting criterion. The tree growing process follows the CART approach (Breiman et al., 1984). The basic idea of tree growing with CART is to select the best split at each parent node among all possible splits to obtain the purest child nodes. Assume we want to split a parent node with n_P observations into two child nodes, namely left and right nodes. To split a node, all possible splits are evaluated with the selected splitting criterion. A split value for each split point is calculated by using the observations in the parent node. Then, the best split is selected among all possible splits. The CART algorithm evaluates all possible splits to decide the split variable and split point. In random forests, instead of evaluating all possible splits, the best split search is limited to a randomly selected subset of covariates. Splitting continues until all nodes become terminal nodes.

Since the goal is to find subgroups of subjects with distinct canonical correlations, we propose a splitting rule that will seek to increase the canonical correlation heterogeneity as fast as possible (Athey et al., 2019; Moradian et al., 2017; Tabib and Larocque, 2020). Define ρ_L and ρ_R as the canonical correlation estimations of the left and right nodes, respectively. The proposed splitting criterion is

$$\sqrt{n_L n_R} * |\rho_L - \rho_R|, \tag{1.2}$$

where n_L and n_R are the left and right node sizes, respectively. The best split among all possible splits is the one that maximizes (1.2). Choi et al. (2020) propose a similar criterion without the $\sqrt{n_L n_R}$ term among one of three possible splitting criteria for their correlation tree method.

1.2.3 Random forest and estimation of canonical correlation for new observations

The previous section describes the splitting criterion and tree growing process for a single tree. The final canonical correlations are estimated with a random forest. Random forest (Breiman, 2001) is a data-driven weight generator. For example, for a continuous

outcome, we can represent random forest predictions for a new observation as a weighted average of the true responses such as $\hat{o}^{new} = \sum_{i=1}^{N} \hat{w}_i(c^{new})o_i$, where $\hat{w}_i$ are the predicted weights from the random forest, c^{new} are the covariates of the new observation, and o_i are the observed outcomes (Hothorn et al., 2004; Lin and Jeon, 2006).

A slightly different representation of these weights was presented in Moradian et al. (2017) and later used in Moradian et al. (2019), Roy and Larocque (2020) and Tabib and Larocque (2020). For a new observation, we form a set of observations which includes the training observations that are in the same terminal nodes as the new observation. Roy and Larocque (2020) called this set of observations the Bag of Observations for Prediction (BOP). We can define BOP for a new observation c^{new} as

$$BOP(c^{new}) = \bigcup_{b=1}^{B} S_b^{new},$$

where S_b^{new} is the set of training observations that are in the same node as c^{new} in the b^{th} tree. Any desired measure can be obtained by using the constructed BOP. In this paper, we use the BOP idea to estimate the canonical correlations for the new observations. Once we train the random forest, we can estimate the correlation for any new observation. In our problem, for a new observation with covariates z^{new}, we firstly form $BOP(z^{new})$. Then, we apply canonical correlation analysis for X and Y with the observations in $BOP(z^{new})$, to compute the canonical correlation estimation $\hat{\rho}(z^{new})$.

We can estimate CCA components only if the sample size, n, is larger than $(p+q)$. In fact, if $n < (p+q)$, the first $(p+q-n)$ canonical correlations will be exactly one and uninformative (Pezeshki et al., 2004). When $n > (p+q)$, although we can estimate the CCA correlations, overfitting can still be a problem. As the sample size in proportion to $(p+q)$ increases, the likelihood of overfitting decreases. During the tree building process, the number of observations in the nodes are getting smaller as we move down in the tree. When the sample size of a node in proportion to a fixed total number of X and Y variables is close to one, we are more likely to overfit. Therefore, we need to control the minimum sample size in the nodes and we do it using the `nodesize` feature of random forests.

1.2.4 Global significance test

For each tree grown in the forest, the proposed method uses the covariate space to identify groups of observations with similar canonical correlations. By doing so, we might be tempted to assume that the set of covariates is indeed relevant to distinguish between canonical correlations. However, this might not be the case and we propose a hypothesis test to evaluate the global effect of the covariates on the canonical correlation. The unconditional canonical correlation between X and Y can be found by computing CCA using the whole sample. If there is a global effect of Z on such correlations, the estimated conditional canonical correlations with the proposed method should be significantly different from the unconditional canonical correlation. We perform the following statistical significance test for the null hypothesis

$$H_0 : \rho(X,Y|Z) = \rho_{CCA}(X,Y), \tag{1.3}$$

where $\rho(X,Y|Z)$ is the conditional canonical correlation between X and Y given Z, and $\rho_{CCA}(X,Y)$ is the unconditional CCA correlation in the population.

Before describing the global significance test, we will describe how to estimate canonical correlations for the training data using out-of-bag (OOB) observations. We train a random forest with B trees using the training observations. Each tree $b = \{1,...,B\}$ is built with the selected random bootstrap sample, i.e. inbag observations (IB^b), that contains approximately 63% distinct observations from the original sample. The remaining training observations are the OOB observations for that tree, namely OOB^b, and they are not used for building the b^{th} tree. After training a random forest with B trees, we have (IB^b,OOB^b) sets for each tree. The estimation of canonical correlation with OOB observations is described in Algorithm 1.1 for a training set with a sample of size n. Basically, for a given training observation, the canonical correlation is estimated with the BOP, but using only the trees for which that observation is OOB.

The proposed global significance test is described in Algorithm 1.2. Firstly, we apply CCA to all X and Y to compute the unconditional canonical correlation, which is the root node correlation, say ρ_{root}. Then, we apply the proposed method for X,Y,Z and estimate

Algorithm 1.1 Estimation of canonical correlation for a training observation z_i with OOB observations

1: **for** i=1,...,n **do**
2: **for** b=1,...,B **do**
3: **if** $z_i \in OOB^b$ **then**
4: Find the terminal node of z_i at tree b, say d
5: $BOP_{oob}(z_i) = BOP_{oob}(z_i) \cup IB_d^b(z_i)$ (where $IB_d^b(z_i)$ is the inbag observations that are in the same terminal node d as z_i)
6: **end if**
7: **end for**
8: Apply CCA for X and Y with the observations in $BOP_{oob}(z_i)$ to find the estimated canonical correlation $\hat{\rho}(z_i)$
9: **end for**

the canonical correlations for each training observation, $\hat{\rho}(z_i)$, by using OOB observations as described in Algorithm 1.1. Finally, we compute the global test statistic with

$$T = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\rho}(z_i) - \rho_{root}\right)^2. \tag{1.4}$$

The global test statistic is the mean squared difference between the unconditional canonical correlation between X and Y, and the estimated canonical correlations with the proposed method. It measures how far the estimated canonical correlations are spread out from the unconditional canonical correlation between X and Y. The larger T is, the more evidence against H_0 we have. We perform a permutation test under the null hypothesis (1.3) by randomly permuting rows of Z. For each permuted Z, we compute the global test statistic (1.4) and estimate a p-value with

$$p = \frac{1}{R}\sum_{r=1}^{R} I(T_r' > T), \tag{1.5}$$

where T_r' is the test statistic for the r^{th} permuted Z and R is the total number of permutations. If the p-value is less than the pre-specified significance level α, we reject the null hypothesis (1.3).

Algorithm 1.2 Global permutation test for covariates' effects

Compute CCA for X and Y in the root node, say ρ_{root}
Train RF with X, Y, Z
Compute estimated canonical correlations with Algorithm 1.1, say $\hat{\rho}(z_i)$
Compute test statistic with $T = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\rho}(z_i) - \rho_{root}\right)^2$
for $r = 1 : R$ **do**
    Permute rows of Z, say Z_r
    Train RF with X, Y, Z_r
    Compute estimated canonical correlations with Algorithm 1.1, say $\hat{\rho}'_r(z_i)$
    Compute test statistic with $T'_r = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\rho}'_r(z_i) - \rho_{root}\right)^2$
end for
Approximate the permutation p-value with (1.5)
Reject the null hypothesis when $p < \alpha$. Otherwise, do not reject the null hypothesis.

1.2.5 Variable importance

Random forests use OOB samples to construct variable importance (VIMP) measures by evaluating the average change in prediction accuracy (see Appendix C for the details). However, we do not have a true response variable since the problem is unsupervised by nature. Therefore, we use the predicted values to compute any VIMP measures we want from existing packages. The idea is that we are measuring the importance of the variables by using a regression forest to reproduce the canonical correlation estimations we obtained from our method. Hence, the VIMP measures reflect the predictive power of the variables on the estimated canonical correlations. Therefore, higher value of VIMP measure implies higher importance for the estimation of canonical correlations. We propose a two-step process to estimate VIMP measures. Firstly, we build a random forest with the proposed splitting criterion for X, Y, Z and compute the estimated canonical correlations, $\hat{\rho}(z_i)$, as described in Algorithm 1.1. Then, we use the $\hat{\rho}(z_i)$ estimations as a continuous response variable for the original covariates (Z) and we train a regression random forest. Finally, we use the VIMP measures from this random forest.

1.2.6 Implementation

We utilised the custom splitting feature of the `randomForestSRC` package (Ishwaran and Kogalur, 2020) to implement our splitting criterion in the tree building process. We have developed an R package called `RFCCA`. The package is available on CRAN, https://CRAN.R-project.org/package=RFCCA.

1.3 Simulations

Evaluating the performance of the proposed method with a real data set is not possible since the true relationships between X and Y with varying Z are typically unknown. Hence, we perform a simulation study with a known data generation process (DGP) to show the performance of our method. We first describe the DGP used in the simulation study. Next, we construct scenarios to validate the proposed global significance test. Since we know the true correlations, we can evaluate the accuracy of the correlation estimations. We can also evaluate the estimated importance ranking of the covariates since we know the set of covariates that are effectively related to the relationship between X and Y, and the ones that are redundant.

1.3.1 DGP

Assume we want to generate a data set with $X \in R^{n\times p}$, $Y \in R^{n\times q}$ and $Z \in R^{n\times r}$ where n is the sample size. We firstly generate the covariates Z according to a standard multivariate normal distribution with an equicorrelated covariance matrix, that is from $N(0,\Sigma_Z)$ where $\Sigma_Z = (1-\rho_z)\mathbf{I}_r + \rho_z \mathbf{J}_r$, $\mathbf{I}_r$ and $\mathbf{J}_r$ are $r \times r$ identity matrix and matrix of ones, respectively. Then, the true correlation between Xa and Yb for each observation i, $\rho(z_i)$, is generated with the following logit model

$$\rho(z_i) = \frac{1}{1+\exp(-(\beta_0 + \sum_{l=1}^{r} \beta_l z_{il} + z_{i1}^2))},$$

where β_0 is the intercept parameter and β_l are the weights for the Z variables. In the simulations, we set all β_l to be $1/r$.

Next, we generate the X and Y coefficients for CCA, a and b in (1.1) respectively, with the following two linear equations:

$$a_{ij} = \max\{0, (1 - s_x \times \rho(z_i) \times j)\} \;\; \forall j = \{1, 2, ..., p\},$$
$$b_{ik} = \max\{0, (1 - s_y \times \rho(z_i) \times k)\} \;\; \forall k = \{1, 2, ..., q\},$$

where s_x and s_y are some slope parameters to be defined. Note that according to these equations, the X and Y variables have a descending order of relative importance. Also, we may set some coefficients to be 0 with an appropriate choice of s_x and s_y, in order to have redundant covariates that are not directly related to the canonical correlations.

Finally, for each observation i, we generate the X and Y variables with a multivariate normal distribution $N(0, \Sigma_i)$, where $\Sigma_i = \begin{pmatrix} \Sigma_X & \Sigma^i_{XY} \\ \Sigma^i_{YX} & \Sigma_Y \end{pmatrix}$, $\Sigma_X = (1 - \rho_x)\mathbf{I}_p + \rho_x \mathbf{J}_p$, $\Sigma_Y = (1 - \rho_y)\mathbf{I}_q + \rho_y \mathbf{J}_q$ and $\Sigma^i_{XY} = \rho(z_i)\Sigma_X a_i b_i^T \Sigma_Y$. See Appendix D for examples of sample distributions with different parameter settings.

1.3.2 Simulation design

Evaluation of the power of the global significance test

In order to evaluate the effect of Z, we consider four scenarios where two of them are under the null hypothesis (1.3) and the other two are under the alternative hypothesis. For all scenarios, we generate X and Y with two levels of CCA correlation defined to be low (0.3) and high (0.6). For the first scenario (case 1 under H_0), these levels represent the population correlation. For the other three scenarios, they represent the mean sample correlation. We generate the data sets for these scenarios as follows:

1. H_0 (case 1): We generate 5 X, 5 Y with a constant population canonical correlation and 10 Z variables which are all independent and following a standard normal distribution. In this case, the correlation between X and Y is independent of Z and we are therefore under the null hypothesis.

2. H_0 (case 2): We first generate 5 X, 5 Y and 5 Z with the proposed DGP. Then, we replace the Z set with 10 independent Z variables generated with a standard normal distribution. In this case, the correlation between X and Y varies with some of the Z variables but those Z variables are not available in the training set. Hence, although the correlation between X and Y is a function of covariates, these covariates are not part of the training set. Therefore, we are again under the null hypothesis.

3. H_1 (without noise): We generate 5 X, 5 Y and 5 Z with the proposed DGP, and the covariates are available in the training set. In this case, the correlation between X and Y varies with all Z variables.

4. H_1 (with noise): We generate 5 X, 5 Y and 5 Z with the proposed DGP and we add 5 independent Z variables to the covariates' training set. In this case, the correlation between X and Y varies with some of the Z variables.

Each of the scenarios above are repeated under the low and high correlation settings, which leads to a total of 8 scenarios to investigate. We also investigate how the performance of the proposed method varies with the training sample sizes, and we use $n_{train} = \{200, 300, 500, 1000, 1500\}$. The number of permutations in the permutation test is set to 500 and each scenario is repeated 500 times. Type-1 error is estimated as the proportion of rejection in the scenarios simulated under H_0. Similarly, we estimate power as the proportion of rejection in the scenarios simulated under H_1.

For each replication of the simulation, we obtain an estimated p-value from the permutation test. If the p-value is less than the significance level $\alpha = 0.05$, we reject the null hypothesis. The proportion of rejection is then calculated over the 500 replications.

Accuracy evaluation

We perform a simulation study to evaluate the accuracy of the method for estimating the canonical correlation. Table 1.1 presents the DGP parameter settings for these simulations. We generate Z with $(r, r^{noise}) = \{(1,5), (5,5), (10,5)\}$ where r and r^{noise} are

Parameters	Low CCA correlation	High CCA correlation
(p,q)	(1,1), (5,5), (10,10)	(1,1), (5,5), (10,10)
(r,r^{noise})	(1,5), (5,5), (10,5)	(1,5), (5,5), (10,5)
(ρ_x,ρ_y,ρ_z)	(0.3, 0.3, 0.1)	(0.3, 0.3, 0.1)
(β_0,β_l)	(-2,$\frac{1}{r}$)	(-0.3,$\frac{1}{r}$)
(s_x,s_y)	(0.7, 0.4)	(0.4, 0.3)

Table 1.1: DGP parameter settings for accuracy evaluation simulations. We consider two levels of average CCA correlation: low (0.3) and high (0.6).

the number of important and noise Z variables, respectively. We generate X and Y with $(p,q) = \{(1,1),(5,5),(10,10)\}$ where p and q are the number of X and Y variables, respectively. We use training sample sizes of $n_{train} = \{100,200,300,500,1000,5000\}$. Also, we consider six values for parameter `nodesize` that controls the size of the trees: `nodesize` = $\{2\times(p+q),3\times(p+q),4\times(p+q),6\times(p+q),8\times(p+q),10\times(p+q)\}$. Overall, these combinations produce 648 settings (2 mean CCA correlation levels $\times$ 3 Z dimensionality $\times$ 3 X and Y dimensionality $\times$ 6 training sample sizes $\times$ 6 `nodesize` levels). Each setting is repeated 100 times for a total of 64,800 runs. In each run, we generate an independent test set of new observations with $n_{test} = 1000$.

We evaluate the performance with the mean absolute errors (MAE), given by

$$MAE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} |\hat{\rho}(z_i) - \rho(z_i)|, \tag{1.6}$$

where $\hat{\rho}(z_i)$ is the estimated canonical correlation and $\rho(z_i)$ is the true correlation for the i^{th} test observation with covariates z_i. Smaller values of the MAE show better performance. We use ordinary CCA, without covariates, as a simple benchmark method. In this case, we let ρ_{train} be the training sample estimated canonical correlation with CCA. This value is used as the correlation estimation for all new observations from the test set.

1.3.3 Results

Global significance test

Figure 1.2 illustrates the estimated Type-1 error for different training sample sizes for both H_0 case 1 and 2. In the first and second columns, we have results for the low and

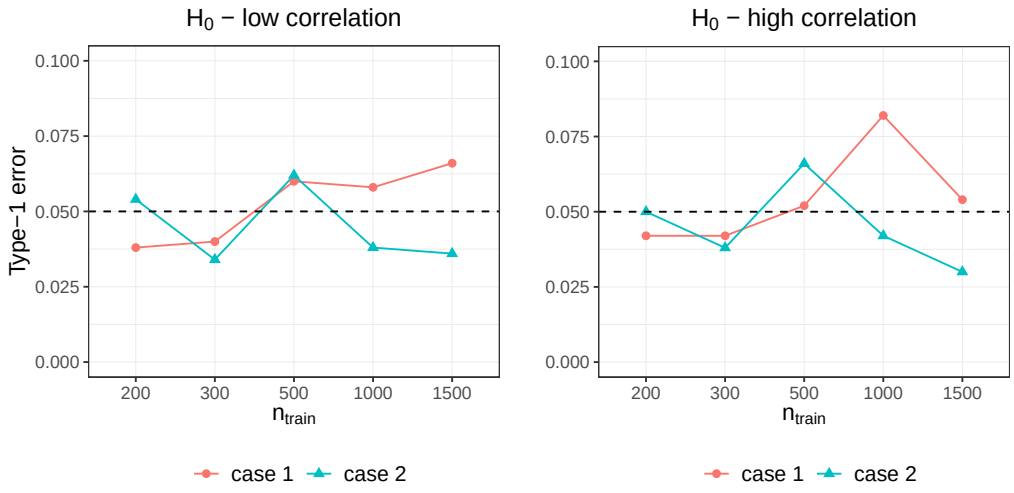

Figure 1.2: Global significance test results for H_0 cases. In case 1 the correlation between X and Y is not varying with Z and in case 2 the correlation between X and Y is varying with some of the Z variables but those Z variables are not used in the training set. Left and right plots are for low and high correlated data sets, respectively. Dashed line represents the significance level of $\alpha = 0.05$.

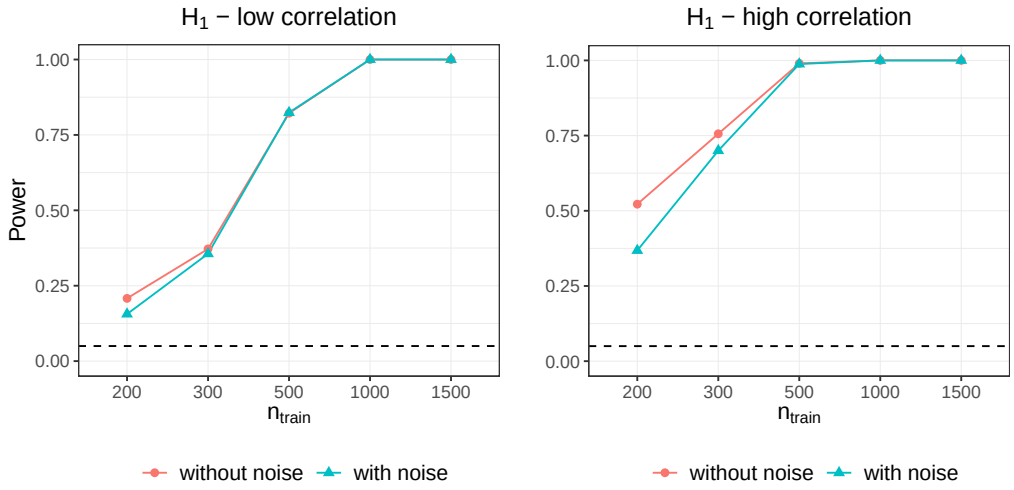

Figure 1.3: Global significance test results for H_1 scenarios. In without noise case, we have only important covariates for the varying correlation between X and Y and in with noise case we have additional noise covariates. Left and right plots are for low and high correlated data sets, respectively. Dashed line represents the significance level of $\alpha = 0.05$.

high correlated data sets, respectively. In H_0 scenarios, we expect the Type-1 error to be close to the significance level ($\alpha = 0.05$). As can be seen from plots, the Type-1 error is well controlled in both cases. Figure 1.3 illustrates the power of the test. In all scenarios and as expected, the power is increasing with the sample size. However, adding noise covariates slightly decreases the power when the sample size is small. As the sample size increases, the power is not affected by the presence of noise covariates.

`nodesize` selection

In this section, we compare the performance of the proposed method with six levels of `nodesize`, $\{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$, for low and high correlated data settings and $n_{train} = \{100, 200, 300, 500, 1000, 5000\}$. Figures 1.4 and 1.5 present the average *MAE* over the 100 repetitions with each n_{train} for the low and high correlation settings, respectively. See Appendix E for the *MAE* results with each `nodesize` for the low and high correlation settings. As can be seen from the results, in some scenarios, *MAE* increases as `nodesize` increases (*e.g.*, high correlation setting with $p = 10, q = 10, r = 5$ and $n_{train} = 1000$) whereas in some scenarios *MAE* decreases as `nodesize` increases (*e.g.*, low correlation setting with $p = 5, q = 5, r = 5$ and $n_{train} = 1000$). Moreover, there are some cases where *MAE* decreases first and then increases as `nodesize` increases (*e.g.*, low correlation setting with $p = 5, q = 5, r = 5$ and $n_{train} = 300$). In such situations, normally we do a hyperparameter tuning with cross-validation or by dividing the data set into train and validation sets. However, in our case, we cannot tune the `nodesize` parameter because we do not have a target. As can be seen from the `nodesize` comparison results, although it is not optimal, the proposed method is almost always better than the benchmark method.

The best accuracy is achieved with different levels of `nodesize` parameter for each scenario and it is hard to select the global best level of `nodesize` for all scenarios from those figures. Hence, we provide a global view of the performance for `nodesize` parameter in Figure 1.6. To be able to compare the accuracy of the proposed method with different levels of `nodesize` parameter across different scenarios, we use the percentage

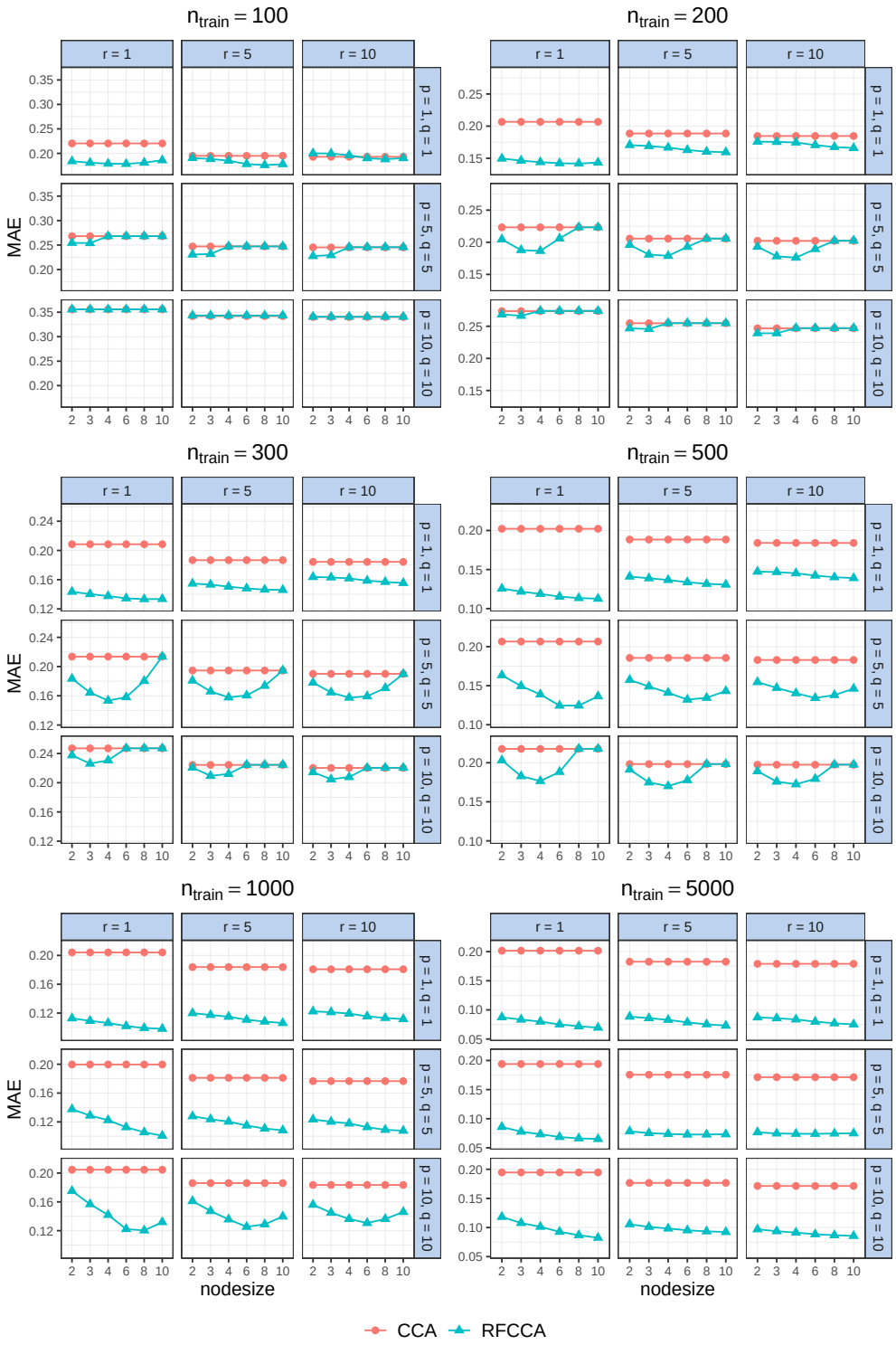

Figure 1.4: Accuracy evaluation results for low correlated data sets. $r^{noise} = 5$ in all settings. The values in the x-axis correspond to the levels of `nodesize` parameter. CCA is the benchmark method. Smaller values of *MAE* are better.

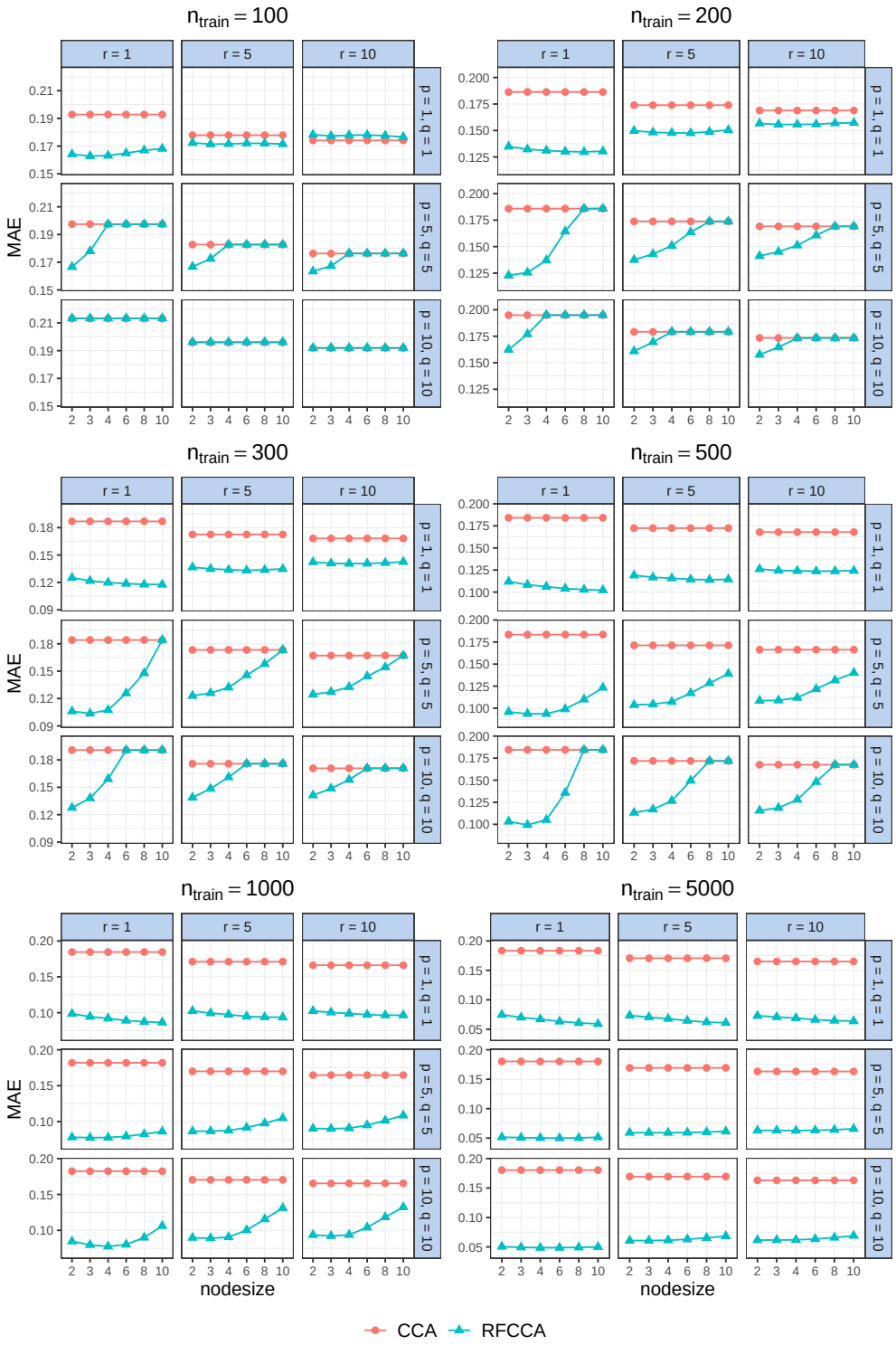

Figure 1.5: Accuracy evaluation results for high correlated data sets. $r^{noise} = 5$ in all settings. The values in the x-axis correspond to the levels of `nodesize` parameter. CCA is the benchmark method. Smaller values of *MAE* are better.

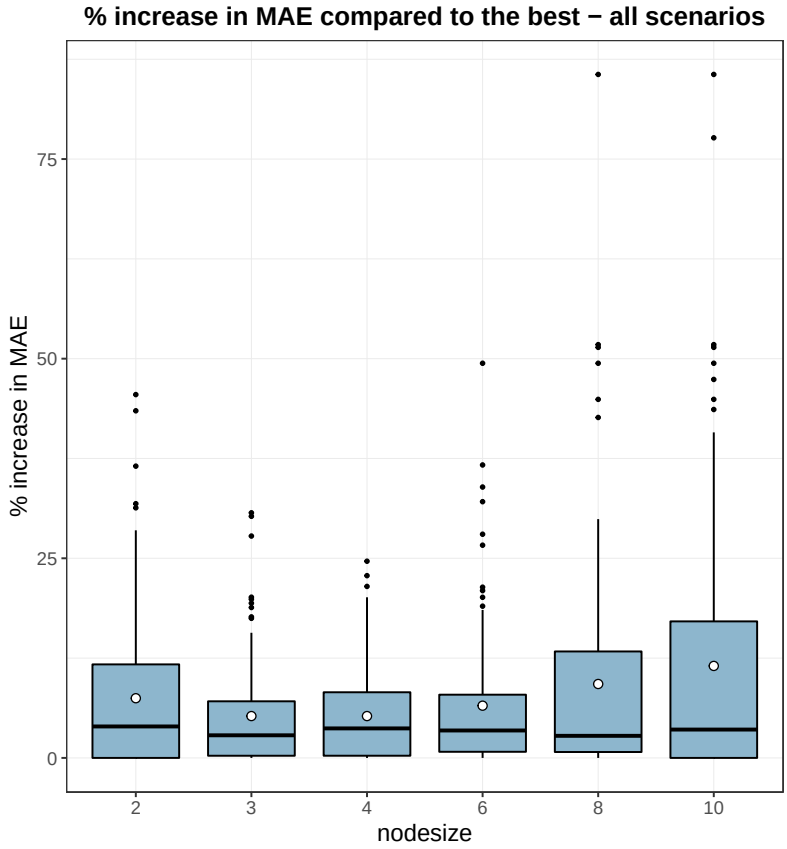

Figure 1.6: Distributions of the percentage increase in MAE of each level of `nodesize` parameter with respect to best performing `nodesize` value for a given scenario for all 108 scenarios in the simulation study.

increase in *MAE* with respect to the best performer `nodesize` value for a given scenario. This way, we can aggregate the results across all scenarios. There are 108 scenarios for each level of `nodesize` parameter in this simulation study (2 mean CCA correlation levels $\times$ 3 Z dimensionality $\times$ 3 X and Y dimensionality $\times$ 6 training sample sizes). For a given scenario, we have the *MAE* on the test set for each `nodesize` level. Let MAE_n be the *MAE* of `nodesize` n, where $n = \{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$, for this scenario. The percentage increase in *MAE* of `nodesize` n with respect to the best performer for this scenario is computed as

$$100 \times \frac{MAE_n - min_n\{MAE_n\}}{min_n\{MAE_n\}}$$

where $min_n\{MAE_n\}$ is the smallest *MAE* for this scenario. Hence, the smaller values indi-

cate better accuracy. Figure 1.6 shows the distributions of this measure for the `nodesize` parameter. The results show that, the mean and median of $3 \times (p+q)$, $4 \times (p+q)$ and $6 \times (p+q)$ are very similar and among them $3 \times (p+q)$ has a smaller median and interquartile range. Overall, selecting `nodesize` as $3 \times (p+q)$ leads to globally smaller or very similar *MAE* results. Hence, we evaluate the accuracy of the proposed method by setting `nodesize` $= 3 \times (p+q)$ for all scenarios.

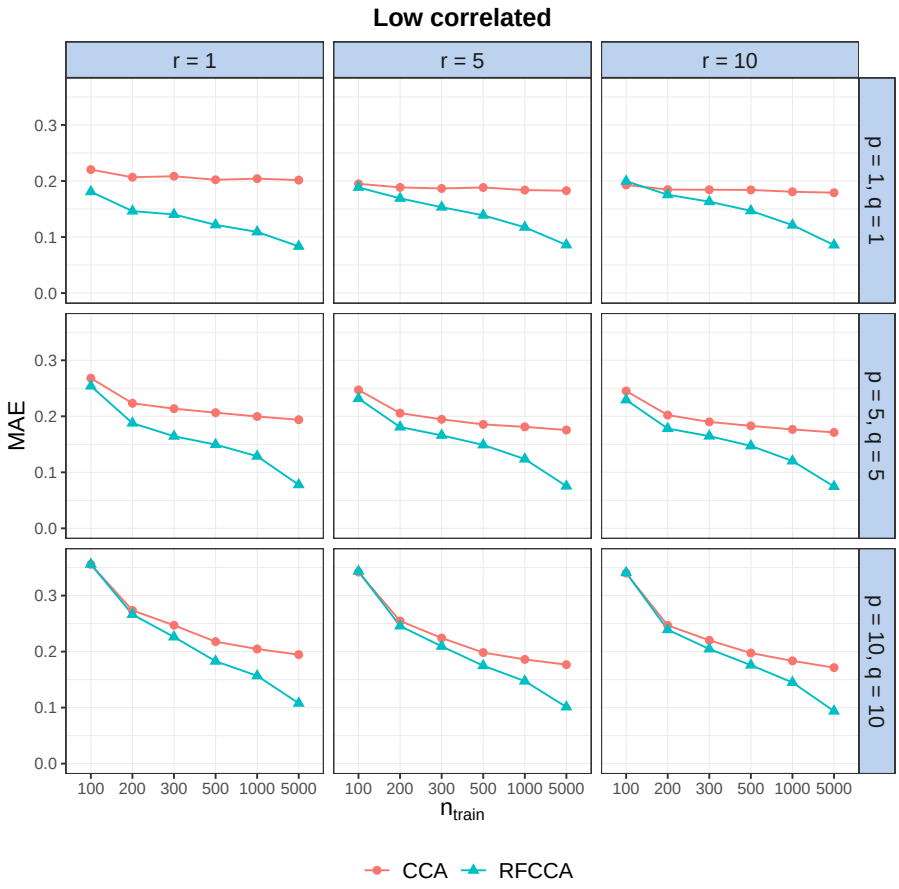

Figure 1.7: Accuracy evaluation results for low correlated data sets when `nodesize` $= 3 \times (p+q)$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

Accuracy evaluation

We provide a summary of the results in Figures 1.7 and 1.8 which present the average *MAE* over the 100 repetitions when `nodesize` $= 3 \times (p+q)$ for the low and high correla-

tion settings, respectively. The plots illustrate the change in *MAE* with increasing training sample size for different r and (p,q) settings. As can be seen from both figures, the proposed method and the benchmark have a similar performance, with a slight advantage for the proposed method in some cases when $n = 100$. When the sample size increases, the MAE of both methods decrease but markedly faster for the proposed method. Hence, in the settings considered, a small sample size of 100 is not sufficient for the proposed method to improve over the ordinary CCA. But when the sample size increases, the proposed method successfully exploits the covariates to provide more accurate estimations of the canonical correlation, the relative gain being more important in the high correlation settings.

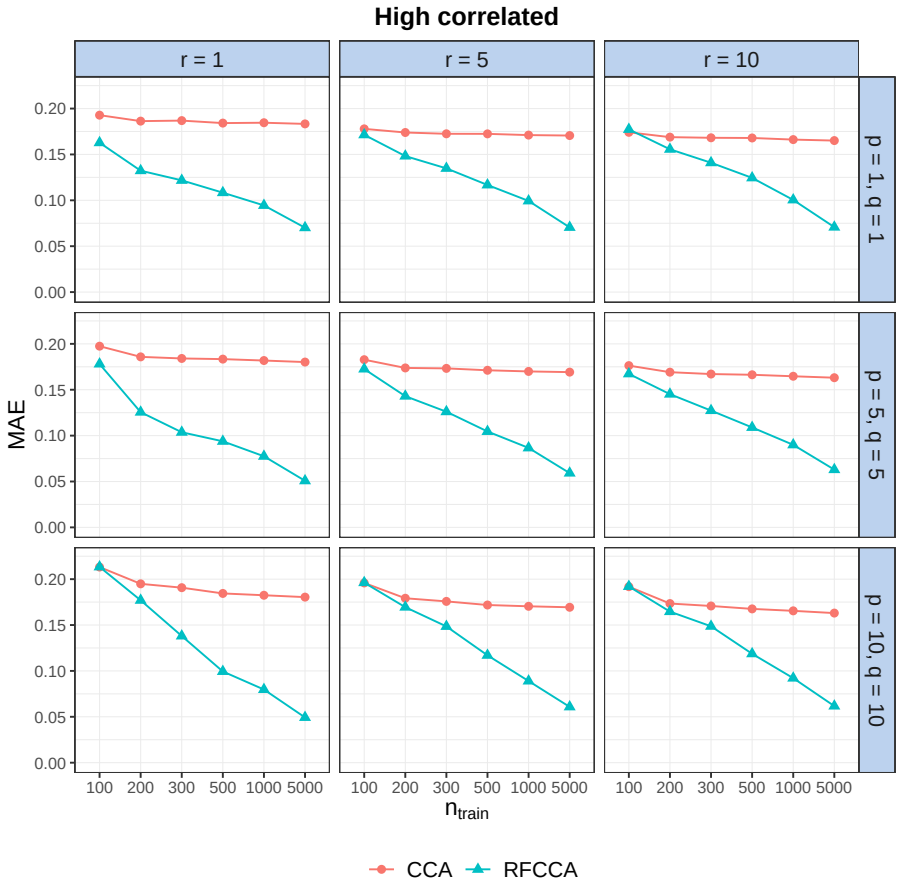

Figure 1.8: Accuracy evaluation results for high correlated data sets when `nodesize` $= 3\times(p+q)$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

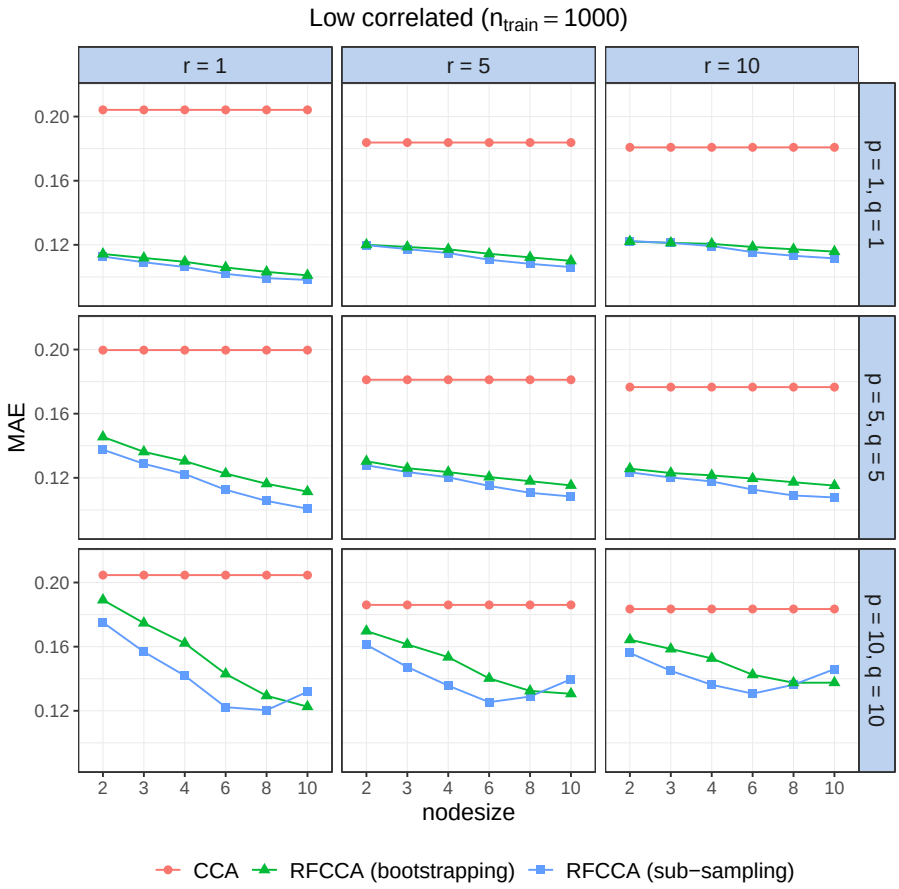

Figure 1.9: Accuracy evaluation results for low correlated data sets when $n_{train} = 1000$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

In `randomForestSRC`, the default sampling for random forest training is sampling without replacement (sub-sampling), unlike the original random forest algorithm that uses bootstrapping. We investigate the effect of sampling on the performance of the proposed method on the scenarios with $n_{train} = 1000$. Also, we analyze the effect of sampling on the selection of the `nodesize` parameter. Figures 1.9 and 1.10 present the average *MAE* over the 100 repetitions when $n_{train} = 1000$ for the low and high correlation settings, respectively. The values in the x-axis correspond the values of the `nodesize` parameter which are $\{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$. We can both compare the effect of sampling method and `nodesize` parameter on the accuracy with those plots. CCA is used as the benchmark method. In most of the settings, there is

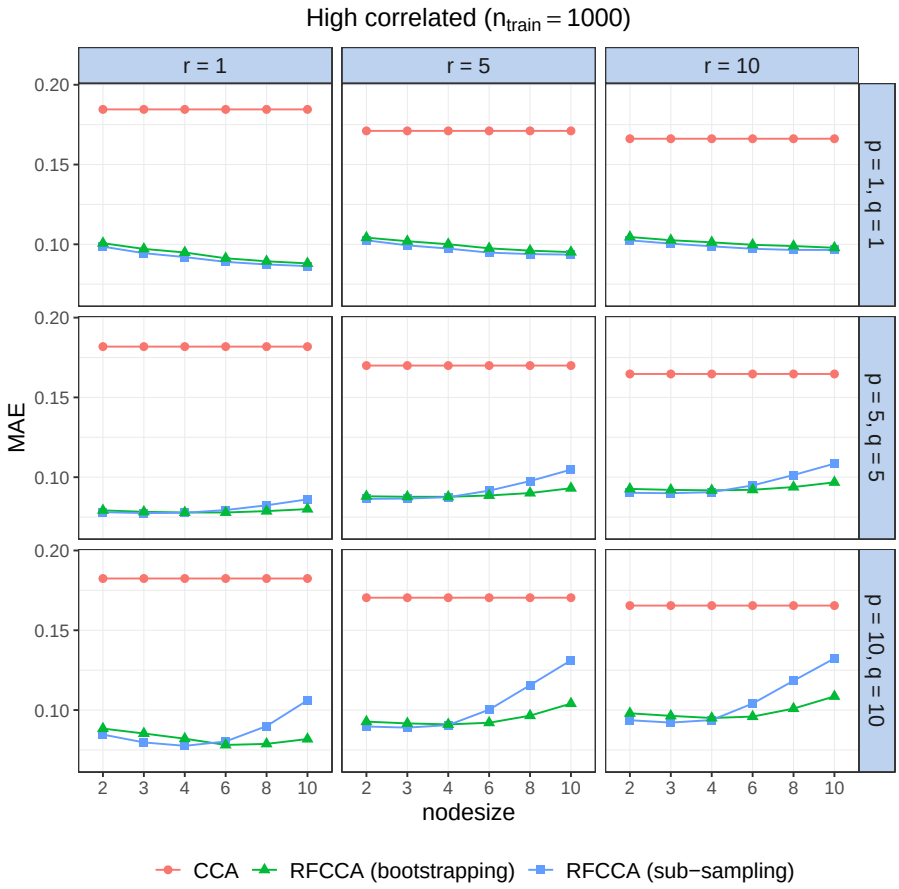

Figure 1.10: Accuracy evaluation results for high correlated data sets when $n_{train} = 1000$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

no significant difference in performance between sub-sampling and bootstrapping. However, in some cases (*e.g.* low correlated data sets with $p = 10, q = 10$), sub-sampling has slightly better accuracy than bootstrapping. There are also some cases, for instance in high correlated data sets with $p = 10, q = 10$, sub-sampling shows better performance for the smaller `nodesize` values whereas bootstrapping has smaller *MAE* for the larger `nodesize` values. However, in those cases the best accuracy is still obtained with sub-sampling and smaller `nodesize` values. Hence, we use sub-sampling in our simulations. Overall, although the optimal value for the `nodesize` parameter may change with the selected sampling method, the accuracy of the proposed method with both sampling methods have a very similar pattern for different levels of `nodesize` parameter.

Variable importance

For the variable importance, we evaluate if the estimated VIMP measures tend to rank the important variables first. In all scenarios for performance evaluation, we include noise covariates. Figures 1.11 and 1.12 present the average rank, from the estimated VIMP measures, for the important variables group and noise variables group, for both low and high correlated data sets, respectively. The most important variable (the one with the highest VIMP measure) has rank 1. As ranks increases, variable importance decreases. In almost all settings, the important variables have smaller average ranks than noise variables. Only in a few settings when $n_{train} = 100$, we have close average ranks for the important and noise variables.

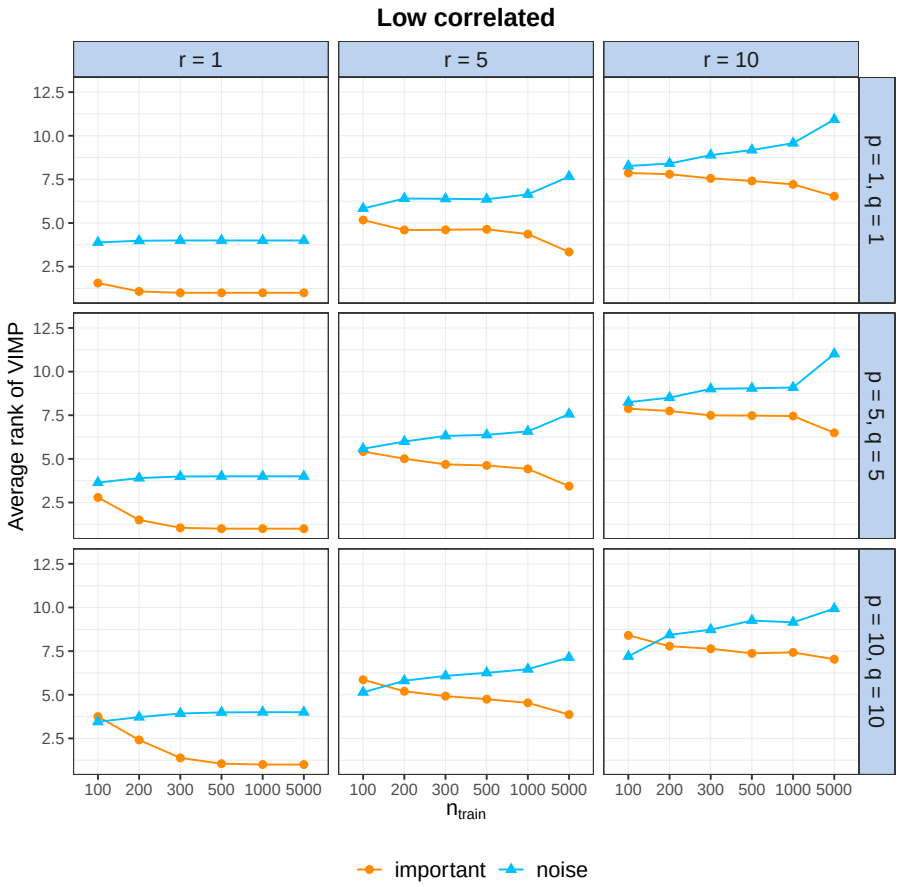

Figure 1.11: Average ranks from estimated VIMP measures for low correlated data sets when `nodesize` = $3 \times (p+q)$. Smaller values of rank indicate a more important variable (the most important variable has rank 1).

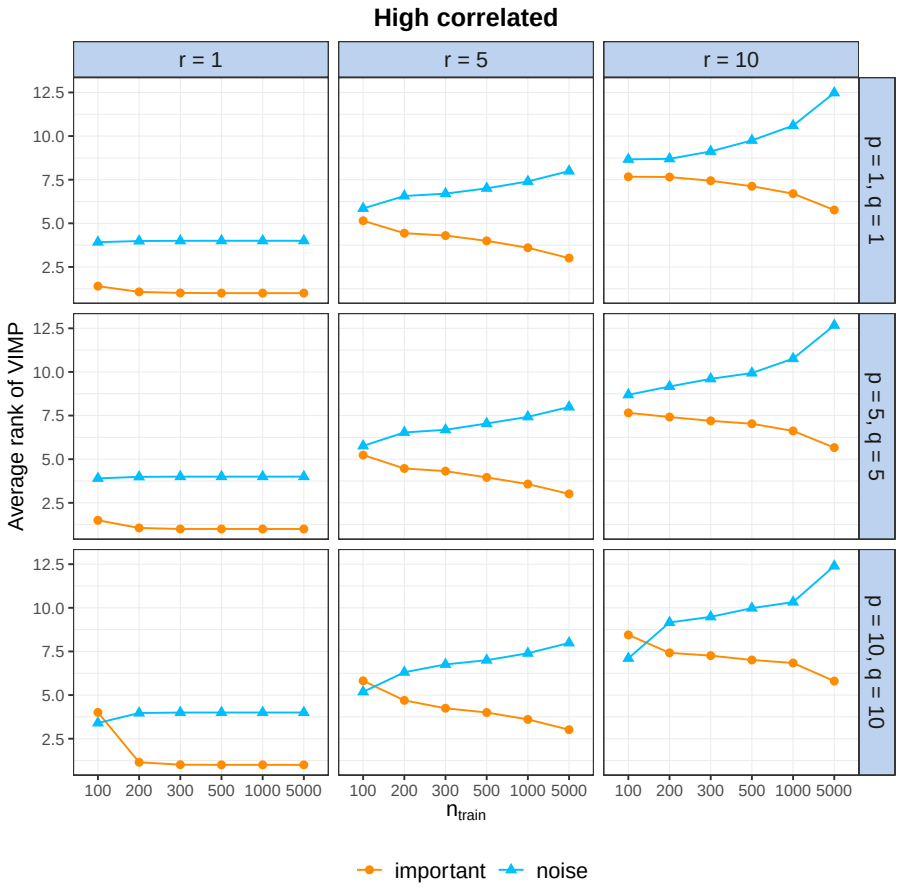

Figure 1.12: Average ranks from estimated VIMP measures for high correlated data sets when `nodesize` = $3 \times (p+q)$. Smaller values of rank indicate a more important variable (the most important variable has rank 1).

1.4 Real data example

Electroencephalogram (EEG) measures neuronal activity. Electrodes are distributed on the scalp to record ongoing electrical fields coming from assembles of pyramidal neurons situated in the cortex. The signal is composed of continuous variation in rhythms that can be spectrally decomposed over time through time and frequency analyses. Oscillations at different frequency bands have been found to be interdependent (Samiee and Baillet, 2017). Only a handful of studies have assessed the effect of age on cross-frequency interdependencies. Its relationship with the intellectual level is still unknown.

In this study, 241 participants between 3 and 64 years old (113 male/128 female) and with performance IQ (pIQ) within [56, 129] and verbal IQ (vIQ) within [50, 127] were submitted to broadband noises of 50ms at 1Hz while the EEG signal was recorded using the 128 electrode EGI system (auditory evoked potentials, AEPs) (see Lippé et al. (2009) for paradigm details). 99 participants presented a copy number variation (CNV). The data collection and preprocessing steps are described in Appendix F. After applying the time-frequency (TF) and inter-trial coherence (ITC) analyses, we have two variables of interest, power, and phase-locking value (PLV) for each window. The time and frequency windows of interest are selected to assess low and high-frequency dependencies. In particular, the three windows for PLV are in theta waves with 3–5 Hz (100–300 ms), 3–10 Hz (4–400 ms), and 3–10 Hz (100–300 ms), and the window for power is in gamma wave with 30–50 Hz (50–150 ms).

When recording auditory evoked potential, the electrodes capture information coming from the auditory cortex in the signal at the scalp level over the mid-frontal region (Albrecht et al., 2000). We want to analyze the association between the PLV variables in theta waves (X, $p = 3$) and power in gamma wave (Y, $q = 1$) in the mid-frontal (MF) region which is composed of Fz-Fcz in addition to four surrounding electrodes. We apply the proposed method with the subject-related covariates age, sex, pIQ, vIQ (Z, $r = 4$) to investigate the correlation between PLV and power for the sample with $n = 241$. We first perform the global significance test to evaluate the global effect of the covariates. Using 500 permutations, the estimated p-value with (1.5) is 0.004 and we reject the null hypothesis (1.3), indicating that the canonical correlation varies significantly with the covariates. Next, we apply the proposed method to the data and obtain the canonical correlation estimations. Figure 1.13 presents the VIMP showing that age is the most important variable followed by pIQ, vIQ and sex. Then, we use the Boruta approach (Kursa and Rudnicki, 2010), which is a permutation test based variable selection algorithm, to evaluate the statistical significance of variable importances. The main idea of this method is to compare the variable importance of the original variables with those of randomly permuted copies using statistical testing and several runs of random forests. All four covariates are selected

as important variables within the significance level $\alpha = 0.01$.

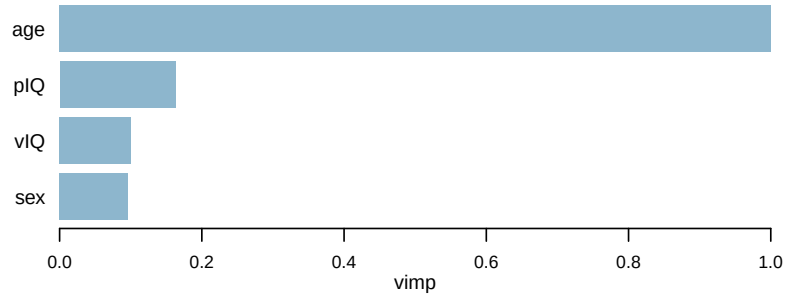

Figure 1.13: Variable importance measures computed with the proposed method.

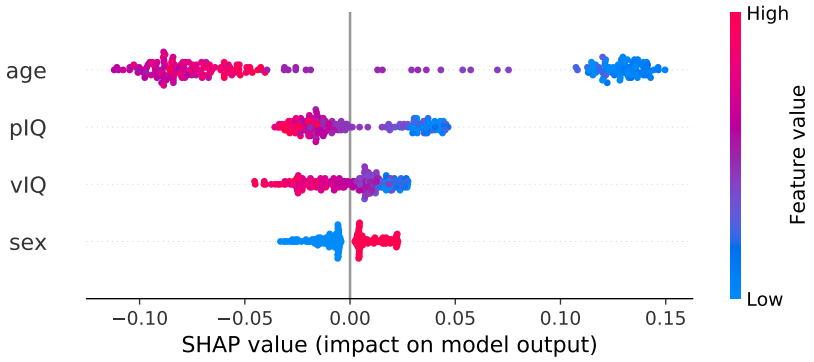

Figure 1.14: SHAP summary plot.

We also use SHAP values (Lundberg and Lee, 2017) to gain additional insights. SHAP values are the contributions of each variable to the difference between the actual prediction and the expected model prediction. The sum of the contributions for each variable (SHAP values) is equal to the prediction. SHAP values show how much each variable contributes, either positively or negatively, to the individual predictions. Since the problem is unsupervised, as in the VIMP computation, we use the predicted correlations to compute the SHAP values, using Lundberg et al. (2020). Figure 1.14 presents the summary plot. The covariates are ordered in the y-axis of the plot according to their global importance showing that we obtain the same ranking as with the VIMP and thus that age is again the most important variable. The insights from this exploratory analysis fall into the expectations. Age, or brain maturation, is accompanied by important neurofunctional modifications that are reflected in the strength of the theta phase coherence and gamma

power co-variations. The demonstration of a positive correlation and a strong contribution to the model in children and adolescents (3 to 20 years of age) is concordant with current brain development literature (Cho et al., 2015). Intellectual quotient, the hallmark of cognitive abilities, is found to be the second most contributive variable to the theta-gamma co-variation. Current literature focused on the theta-gamma coupling links with performances on specific cognitive tasks (Alekseichuk et al., 2016), and showed positive correlations. This result points to the relevance of theta-gamma co-variation in the context of abnormal neurodevelopment (Port et al., 2019).

Figure 1.15 shows the main effect of age on the predictions. We can see how age's attributed importance changes as its value varies. The attributed importance is on the *y*-axis. The points are colored according to the predicted correlation. We can interpret this plot as the impact of age on correlation is positive and high for subjects younger than 20. Then it drops sharply reaching 0 (no impact) around 25 (note that we do not have observations around 20 and the results should be interpreted cautiously in that age range). It then continues to decrease in the negative direction, meaning the impact increases until the beginning of the 30's where it stabilizes with a slight increase afterwards (i.e. a decreasing impact).

Left plot in Figure 1.16 presents the interaction effect between sex and pIQ. Similarly, right plot in Figure 1.16 presents the interaction effect between sex and vIQ. We see that the impact increases as we move away from the average IQ. The impact of the interaction on the theta phase coherence strength and gamma power co-variation is positive for high IQ females and negative for low IQ females. The opposite is observed in males.

As a neuroscientific conclusion, PLV in theta waves and power in gamma waves are statistically coupled and this coupling varies according to age and IQ. This confirms that the co-variation in these frequency bands relate to cognitive development. The results suggest that the co-variation between theta and gamma and intellectual capacities is non-linear and that it follows a distinct pattern in males and females. Overall, these results indicate the importance of considering sex, intellectual capacities, and age in the study of brain signal dynamics.

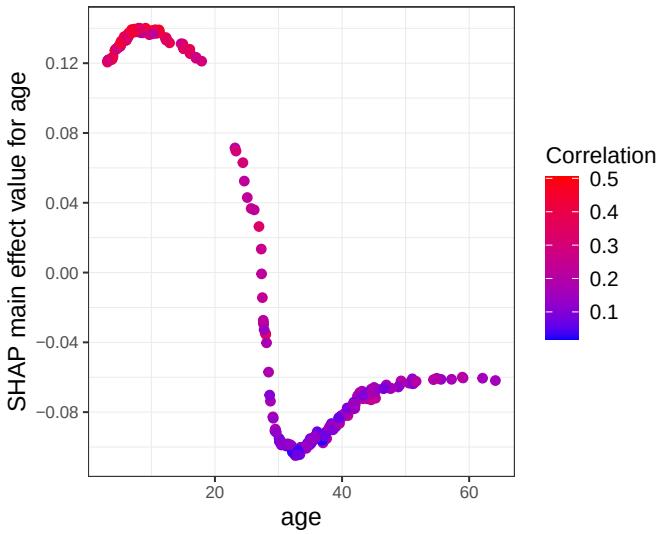

Figure 1.15: SHAP main effect values for age.

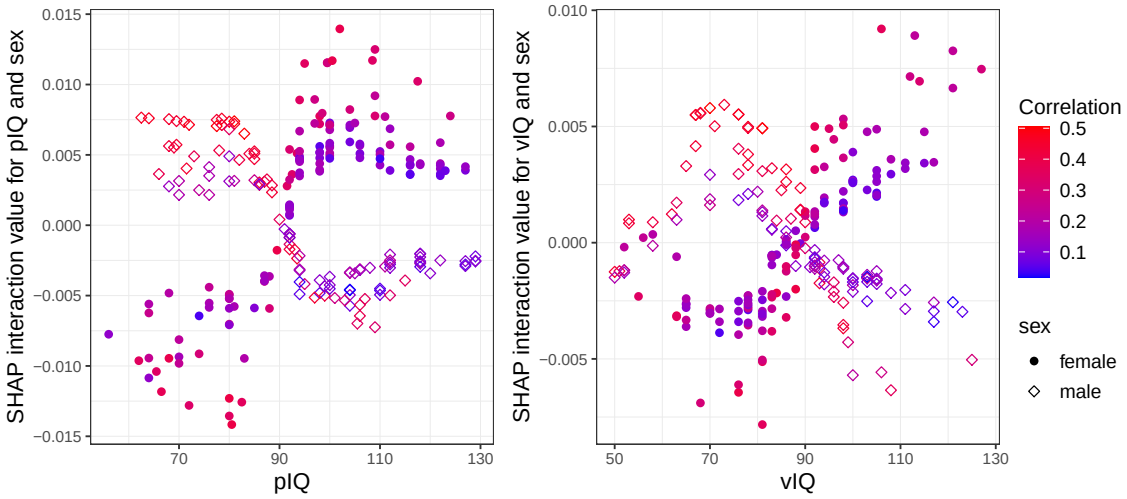

Figure 1.16: (Left) SHAP interaction values for sex and pIQ. (Right) SHAP interaction values for sex and vIQ.

1.5 Conclusion

In this paper, we study and propose a novel random forest method to estimate the canonical correlations between two sets of variables depending on a set of subject-related covariates. The trees of the forest are built with a new splitting rule designed to form child nodes with maximum difference in the canonical correlation between the two multivariate data sets. Random forest is used to build Bag of Observations for Prediction (BOP) which can be used to compute any desired measure. We use the BOP to estimate the correlations with canonical correlation analysis for the new observations. The proposed method is flexible to various extensions. One of them is to use CCA variants such as sparse CCA (Witten et al., 2009) and regularized CCA (Vinod, 1976; Leurgans et al., 1993) for final canonical correlation estimation for a new observation using the constructed BOP. Another one is to build trees with alternative splitting rules such as $n_L\rho_L^2 + n_R\rho_R^2$ where ρ_L and ρ_R are left and right canonical correlation estimations and n_L and n_R are left and right node sizes, respectively. It would be interesting to investigate these extensions as a future work.

We also propose a global significance test to evaluate the global effect of the subject-related covariates and a way to compute variable importance measures. It would also be interesting to study the statistical significance of variable importance measures. The proposed method is based on the CART approach. Other tree-growing paradigms could be used, like the one that separates the variable and split point selections; the conditional inference framework (Hothorn et al., 2006) being one popular method. In principle, the proposed permutation test for covariates' effects could be used to select the split variable at a node, analogous to the conditional inference framework. More precisely, assume we are at a node and want to decide whether to split it or not and with which covariate. Using only the observations in the node;

1. For one covariate at a time, apply the permutation test described in Section 1.2.4. However, permute only the rows of the given covariate instead of permuting rows of covariate set (Z). Obtain a p-value for that covariate.

2. Repeat Step 1 for all covariates to obtain one p-value per covariate.

3. If none of the covariates are significant (after applying a multiple testing correction if deemed appropriate), then do not split the node.

4. Otherwise, select the covariate with the smallest p-value as the split variable. Find the best split using the proposed split criterion.

However, a limitation of the proposed method is the computational time. Computing CCA for $X \in R^{n \times p}$ and $Y \in R^{n \times q}$ has a time complexity $O\big(n(p^2+q^2)\big)$ where $n > p+q$. For each node split in each tree of the forest, we compute CCA for left and right nodes which brings a lot of CCA computations. Therefore, testing the statistical significance of variable importance of covariates at each node with a permutation test has a great computational cost.

For simulations and real data analysis, we used the default parameter settings for `randomForestSRC` except the number of trees and the `nodesize` argument. We train random forests with 200 trees and `nodesize`$= 3 \times (p+q)$. The simulation study results showed that using different levels of `nodesize` could change the performance of the proposed method. In such situations, normally, we do a hyperparameter tuning to select the optimal level of the parameter. However, in our case, it is not straightforward to tune the `nodesize` parameter because we do not have an observed target. It would be interesting to find such a way to tune `nodesize` parameter as a future work. Moreover, by default, only 10 random splits are considered at each candidate splitting variable to increase the speed. Evaluating all possible splits could also improve the performance.

The proposed method can be used in other bioinformatics studies. For example, in gene-environment interaction studies (Caspi and Moffitt, 2006; Hunter, 2005; Ma et al., 2011), the covariates (Z) would be the environment variables, and the two multivariate data sets (X and Y) would correspond to brain imaging and genomic variables. In another application, the proposed method would allow us to investigate how gene expression (Z) can modulate the correlation between genomic (X) and brain imaging data (Y). In all

these examples, the proposed algorithm can capture nonlinear interactions, which is new compared to existing approaches.

References

Akaho, S. (2001). A kernel method for canonical correlation analysis. In *Proceedings of the International Meeting of the Psychometric Society (IMPS2001)*. Springer-Verlag.

Albrecht, R., Suchodoletz, W., and Uwer, R. (2000). The development of auditory evoked dipole source activity from childhood to adulthood. *Clinical Neurophysiology*, 111(12):2268–2276.

Alekseichuk, I., Turi, Z., de Lara, G. A., Antal, A., and Paulus, W. (2016). Spatial working memory in humans depends on theta and high gamma synchronization in the prefrontal cortex. *Current Biology*, 26(12):1513–1521.

Andrew, G., Arora, R., Bilmes, J., and Livescu, K. (2013). Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, page III–1247–III–1255. JMLR.org.

Athey, S., Tibshirani, J., Wager, S., et al. (2019). Generalized random forests. *The Annals of Statistics*, 47(2):1148–1178.

Bach, F. R. and Jordan, M. I. (2002). Kernel independent component analysis. *Journal of machine learning research*, 3(Jul):1–48.

Branco, J. A., Croux, C., Filzmoser, P., and Oliveira, M. R. (2005). Robust canonical correlations: A comparative study. *Computational Statistics*, 20(2):203–229.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press, Boca Raton.

Cancer Genome Atlas Network (2012). Comprehensive molecular portraits of human breast tumours. *Nature*, 490(7418):61.

Caspi, A. and Moffitt, T. E. (2006). Gene–environment interactions in psychiatry: joining forces with neuroscience. *Nature Reviews Neuroscience*, 7(7):583–590.

Cho, R. Y., Walker, C. P., Polizzotto, N. R., Wozny, T. A., Fissell, C., Chen, C.-M. A., and Lewis, D. A. (2015). Development of sensory gamma oscillations and cross-frequency coupling from childhood to early adulthood. *Cerebral cortex*, 25(6):1509–1518.

Choi, D., Li, L., Liu, H., and Zeng, L. (2020). A recursive partitioning approach for subgroup identification in brain–behaviour correlation analysis. *Pattern Analysis and Applications*, 23(1):161–177.

Cruz-Cano, R. and Lee, M.-L. T. (2014). Fast regularized canonical correlation analysis. *Computational Statistics & Data Analysis*, 70:88–100.

Davis, S. W., Dennis, N. A., Daselaar, S. M., Fleck, M. S., and Cabeza, R. (2008). Que pasa? the posterior–anterior shift in aging. *Cerebral cortex*, 18(5):1201–1209.

ENCODE Project Consortium (2012). An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57–74.

Ewerbring, L. M. and Luk, F. T. (1989). Canonical correlations and generalized svd: applications and new algorithms. *Journal of computational and applied mathematics*, 27(1-2):37–52.

Fratello, M., Caiazzo, G., Trojsi, F., Russo, A., Tedeschi, G., Tagliaferri, R., and Esposito, F. (2017). Multi-view ensemble classification of brain connectivity images for neurodegeneration type discrimination. *Neuroinformatics*, 15(2):199–213.

Hanna, F., Molfenter, S. M., Cliffe, R. E., Chau, T., and Steele, C. M. (2010). Anthropometric and demographic correlates of dual-axis swallowing accelerometry signal characteristics: a canonical correlation analysis. *Dysphagia*, 25(2):94–103.

Hardoon, D. R. and Shawe-Taylor, J. (2011). Sparse canonical correlation analysis. *Machine Learning*, 83(3):331–353.

Hardoon, D. R., Szedmak, S., and Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664.

Healy, M. (1957). A rotation method for computing canonical correlations. *Mathematics of Computation*, 11(58):83–86.

Hotelling, H. (1936). Relations between two sets of variates. *Biometrika*, 28(3/4):321–377.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, 15(3):651–674.

Hothorn, T., Lausen, B., Benner, A., and Radespiel-Tröger, M. (2004). Bagging survival trees. *Statistics in medicine*, 23(1):77–91.

Hunter, D. J. (2005). Gene–environment interactions in human diseases. *Nature Reviews Genetics*, 6(4):287–298.

Ishwaran, H. and Kogalur, U. (2020). *Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 2.9.3.

Kettenring, J. R. (1971). Canonical analysis of several sets of variables. *Biometrika*, 58(3):433–451.

Kursa, M. B. and Rudnicki, W. R. (2010). Feature selection with the Boruta package. *Journal of Statistical Software*, 36(11):1–13.

Leurgans, S. E., Moyeed, R. A., and Silverman, B. W. (1993). Canonical correlation analysis when the data are curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 55(3):725–740.

Li, G. and Jung, S. (2017). Incorporating covariates into integrated factor analysis of multi-view data. *Biometrics*, 73(4):1433–1442.

Li, T., Luo, Q., and Gong, H. (2010). Gender-specific hemodynamics in prefrontal cortex during a verbal working memory task by near-infrared spectroscopy. *Behavioural brain research*, 209(1):148–153.

Li, Y., Wu, F.-X., and Ngom, A. (2018). A review on machine learning principles for multi-view biological data integration. *Briefings in bioinformatics*, 19(2):325–340.

Lin, Y. and Jeon, Y. (2006). Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101(474):578–590.

Lippé, S., Martinez-Montes, E., Arcand, C., and Lassonde, M. (2009). Electrophysiological study of auditory development. *Neuroscience*, 164(3):1108–1118.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. (2020). From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, 2(1):2522–5839.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Ma, S., Yang, L., Romero, R., and Cui, Y. (2011). Varying coefficient model for gene–environment interaction: a non-linear look. *Bioinformatics*, 27(15):2119–2126.

Melzer, T., Reiter, M., and Bischof, H. (2001). Nonlinear feature extraction using generalized canonical correlation analysis. In *International Conference on Artificial Neural Networks*, pages 353–360. Springer.

Meng, C., Zeleznik, O. A., Thallinger, G. G., Kuster, B., Gholami, A. M., and Culhane, A. C. (2016). Dimension reduction techniques for the integrative analysis of multi-omics data. *Briefings in bioinformatics*, 17(4):628–641.

Michaeli, T., Wang, W., and Livescu, K. (2016). Nonparametric canonical correlation analysis. In *International Conference on Machine Learning*, pages 1967–1976.

Mihalik, A., Ferreira, F. S., Moutoussis, M., Ziegler, G., Adams, R. A., Rosa, M. J., Prabhu, G., de Oliveira, L., Pereira, M., Bullmore, E. T., et al. (2020). Multiple holdouts with stability: Improving the generalizability of machine learning analyses of brain–behavior relationships. *Biological psychiatry*, 87(4):368–376.

Min, S., Lee, B., and Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869.

Moradian, H., Larocque, D., and Bellavance, F. (2017). L1 splitting rules in survival forests. *Lifetime data analysis*, 23(4):671.

Moradian, H., Larocque, D., and Bellavance, F. (2019). Survival forests for data with dependent censoring. *Statistical methods in medical research*, 28(2):445–461.

Moser, D. A., Doucet, G. E., Lee, W. H., Rasgon, A., Krinsky, H., Leibu, E., Ing, A., Schumann, G., Rasgon, N., and Frangou, S. (2018). Multivariate associations among behavioral, clinical, and multimodal imaging phenotypes in patients with psychosis. *JAMA psychiatry*, 75(4):386–395.

Pezeshki, A., Scharf, L. L., Azimi-Sadjadi, M. R., and Lundberg, M. (2004). Empirical canonical correlation analysis in subspaces. In *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*, volume 1, pages 994–997. IEEE.

Port, R. G., Dipiero, M. A., Ku, M., Liu, S., Blaskey, L., Kuschner, E. S., Edgar, J. C., Roberts, T. P., and Berman, J. I. (2019). Children with autism spectrum disorder

demonstrate regionally specific altered resting-state phase–amplitude coupling. *Brain Connectivity*, 9(5):425–436.

Roy, M.-H. and Larocque, D. (2020). Prediction intervals with random forests. *Statistical Methods in Medical Research*, 29(1):205–229.

Samiee, S. and Baillet, S. (2017). Time-resolved phase-amplitude coupling in neural oscillations. *NeuroImage*, 159:270–279.

Sun, S. (2013). A survey of multi-view machine learning. *Neural computing and applications*, 23(7-8):2031–2038.

Tabib, S. and Larocque, D. (2020). Non-parametric individual treatment effect estimation for survival data with random forests. *Bioinformatics*, 36(2):629–636.

Vinod, H. D. (1976). Canonical ridge and econometrics of joint production. *Journal of econometrics*, 4(2):147–166.

Wilms, I. and Croux, C. (2015). Sparse canonical correlation analysis from a predictive point of view. *Biometrical Journal*, 57(5):834–851.

Witten, D. M., Tibshirani, R., and Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534.

Chapter 2

RFpredInterval: An R Package for Prediction Intervals with Random Forests and Boosted Forests

Abstract

Like many predictive models, random forests provide point predictions for new observations. Besides the point prediction, it is important to quantify the uncertainty in the prediction. Prediction intervals provide information about the reliability of the point predictions. We have developed a comprehensive R package, `RFpredInterval`, that integrates 16 methods to build prediction intervals with random forests and boosted forests. The set of methods implemented in the package includes a new method to build prediction intervals with boosted forests (PIBF) and 15 method variations to produce prediction intervals with random forests, as proposed by Roy and Larocque (2020). We perform an extensive simulation study and apply real data analyses to compare the performance of the proposed method to ten existing methods for building prediction intervals with random forests. The results show that the proposed method is very competitive and, globally, outperforms competing methods.

2.1 Introduction

Predictive modelling is the general concept of building a model that describes how a group of covariates can be used to predict a response variable. The objective is to predict the unknown responses of observations given their covariates. For example, predictive models could be used to predict the sale price of houses given house characteristics (De Cock, 2011). In its simplest form, a predictive model aims to provide a point prediction for a new observation. However, a point prediction does not contain information about its precision that can tell us how close to the true response we can expect the prediction to be, which is often important in decision-making context. Hence, although the point prediction is often the main goal of predictive analysis, assessing its reliability is equally important, and this can be achieved with a prediction interval (PI). A PI contains a set of likely values for the true response with an associated level of confidence, usually, 90% or 95%. Given that shorter PIs are more informative, developing predictive models that can produce shorter PIs along with the point predictions is crucial in assessing and quantifying the prediction error. In real-world applications, knowing the prediction error alongside the point prediction increases the practical value of the prediction.

Regression analysis is a form of predictive modelling technique that examines the relationship between a response variable and a group of covariates. In this paper, we consider a general regression model

$$Y = g(X) + \varepsilon, \tag{2.1}$$

where Y is a univariate continuous response variable, X is a p-dimensional vector of predictors, and ε is an error term. We assume $g(.)$ is an unknown smooth function $\Re^p \to \Re$ and $E[Y|X = x] = g(X)$. A confidence interval of the prediction is a range likely to contain the location of the response variable's true population mean. However, a prediction interval for a new observation is wider than its corresponding confidence interval and provides a range likely to contain this new observation's response value.

In the past decade, random forests have increased in popularity and provide an efficient way to generate point predictions for model (2.1). A random forest is an ensemble

method composed of many decision trees, which can be described with a simple algorithm (Breiman, 2001). For each tree $b = \{1, ..., B\}$, a bootstrap sample of observations is drawn and a fully grown tree is built such that a set of predictors is randomly selected at each node and the best split is selected among all possible splits with those predictors only. The random forest prediction for a new observation is the average of the B trees

$$\hat{y}_{new} = \frac{1}{B} \sum_{b=1}^{B} \hat{y}_{new}^{b},$$

where $\hat{y}_{new}^{b}$ is the tree prediction for the new observation in the bth tree, i.e. the average of observations in the terminal node corresponding to the new observation. Besides this traditional description, the modern view also considers random forests as data-driven weight generators (Hothorn et al., 2004; Lin and Jeon, 2006; Moradian et al., 2017, 2019; Athey et al., 2019; Roy and Larocque, 2020; Tabib and Larocque, 2020; Alakuş et al., 2021).

Although random forests limit over-fitting by combining many trees, which reduces the variance of the estimator, final predictions can be biased (Mentch and Hooker, 2016; Wager and Athey, 2018). Since each tree is built under the same random process, all trees focus on the same part of the response signal, usually the strongest. Therefore, some parts of the response signal may be left untargeted, which could result in biased point predictions. Wager and Athey (2018) provide bounds for the extent of the bias of random forests under some assumptions about the tree growing process. Following their work, Ghosal and Hooker (2021) proposed a bias correction method in a regression framework called *one-step boosted forest*, which is introduced in Breiman (2001) and Zhang and Lu (2012). The main idea of the proposed method is to sum the predictions of two random forests, where the first is a regression forest fitted on the target data set and the second is fitted on the out-of-bag residuals of the former. Empirical studies show that this method provides a significant bias reduction when compared to a simple random forest.

The current paper proposes an R package providing, among other features, an extension of the one-step boosted forest method described above (Ghosal and Hooker, 2021). The literature on prediction intervals for random forests consists mostly of recent studies. The first method is the Quantile Regression Forests (QRF) method proposed by Mein-

shausen (2006). The aim of QRF is to estimate conditional quantiles of the response variable, instead of conditional means, using an estimated cumulative distribution function obtained with the nearest neighbour forest weights introduced by Hothorn et al. (2004). Prediction intervals can be built directly from the estimated conditional quantiles. The method is implemented in the CRAN package `quantregForest` (Meinshausen, 2017).

In a more recent study, Athey et al. (2019) proposed Generalized Random Forests (GRF), a very general framework to estimate any quantity, such as conditional means, quantiles or average partial effects, identified by local moment equations. Trees are grown with splitting rules designed to maximize heterogeneity with respect to the quantity of interest. Quantile regression forest is one of the applications of GRF. Similar to the QRF, the GRF method uses the neighbourhood information from different trees to compute a weighted set of neighbours for each test point. Unlike QRF, which grows trees with the least-squares criterion, GRF uses a splitting rule designed to capture heterogeneity in conditional quantiles. An implementation of quantile regression forest with GRF is available in the function `quantile_forest` of the CRAN package `grf` (Tibshirani et al., 2021).

Vovk et al. (2005, 2009) introduced a general distribution-free conformal prediction interval framework. Any predictive model, including random forests, can be used within the proposed methodology. The idea is to use an augmented data set that includes the new observation to be predicted to fit the model, and apply a set of hypothesis tests to provide an error bound around the point prediction for the new observation. Although this method does not require any distribution assumptions, it is computationally intensive. Lei et al. (2018) proposed a variant of this method, called Split Conformal (SC) prediction, which splits the data into two subsets, one to fit the model, and one to compute the quantiles of the residual distribution. We note that, while the original full conformal prediction interval framework produces shorter intervals, SC is computationally more efficient. The R package `conformalInference` (Tibshirani, 2019), available on GitHub, implements this method.

Roy and Larocque (2020) proposed 20 distinct variations of methods to improve the

performance of prediction intervals with random forests. These approaches differ according to 1) the method used to build the forest and 2) the method used to build the prediction interval. Four methods can be used to build the forest: three from the classification and regression tree (CART) paradigm (Breiman and Breiman, 1984) and the transformation forest method (TRF) proposed by Hothorn and Zeileis (2021). Within the CART paradigm, in addition to the default least-squares (LS) splitting criterion, two alternative splitting criteria, L_1 and shortest prediction interval (SPI), are considered. Prediction intervals are built using the Bag of Observations for Prediction (BOP), which is the set of nearest neighbour observations previously used in Moradian et al. (2017, 2019). In addition to the type of forest chosen, there are also five methods to build prediction intervals: the classical method (LM), the quantile method (Quant), the shortest prediction interval (SPI), the highest density region (HDR), and the contiguous HDR (CHDR). LM is computed based on an intercept-only linear model using the BOP as the sample, and produces a symmetric PI around the point prediction. The quantile method, similar to the QRF method, is based on the quantiles of the BOP. SPI corresponds to the shortest interval among the intervals that contain at least $(1-\alpha)\,100\%$ of the observations in the BOP. As an alternative to SPI, HDR is the smallest region in the BOP, with the desired coverage $(1-\alpha)$. Note that HDR is not necessarily a single interval. If the distribution is multimodal, it can be formed by multiple intervals. Finally, CHDR is a way to obtain a single prediction interval from HDR intervals by building an interval with the minimum and maximum bounds of the HDR intervals.

Zhang et al. (2020) proposed a forest-based prediction interval method, called Out-of-Bag (OOB) prediction intervals, to estimate prediction intervals using the empirical quantiles of the out-of-bag prediction errors. The method assumes that OOB prediction errors are identically distributed and that their distribution can be well approximated by the out-of-bag prediction errors obtained from all training observations. The resulting prediction intervals have the same width for all test observations. The method is implemented in the CRAN package `rfinterval` (Zhang, 2019).

Lu and Hardin (2021) proposed a very interesting and useful method to estimate the

conditional prediction error distribution of a random forest. The main idea of the proposed method is to use a random forest to compute the out-of-bag residuals for the training observations and to form a set of out-of-bag neighbours for each test point. Then, the conditional prediction error distribution for each test point is determined with the out-of-bag residuals in the neighbourhood. Estimating the prediction error distribution enables the estimation of conditional quantiles, conditional biases and conditional mean squared prediction errors. The prediction interval for a test point x, defined by $\widehat{PI}_\alpha(x)$, is formed by adding the $\alpha/2$ and $1-\alpha/2$ quantiles of the conditional prediction error distribution to the random forest point prediction. The estimators are implemented in the CRAN package `forestError` (Lu and Hardin, 2020).

Note that the conformal inference, OOB approach of Zhang et al. (2020) and the $\widehat{PI}_\alpha(x)$ method of Lu and Hardin (2021) all use the prediction errors to build the prediction intervals. Instead of using the training responses directly to estimate quantiles, using prediction errors provides a better predictive power. However, unlike conformal inference and the OOB approach, the $\widehat{PI}_\alpha(x)$ method uses the nearest neighbour observations to estimate the prediction error distribution. This idea is very similar to the BOP idea (Roy and Larocque, 2020), but instead of using in-bag observations, Lu and Hardin (2021) use out-of-bag observations to form the neighbourhoods. This approach allows the local information for the test observations to be extracted.

In this paper, we introduce the R package `RFpredInterval` (Alakus et al., 2022), which is the novel implementation of 16 methods to build prediction intervals with random forests and boosted forests. The set of methods implemented in the package includes a new method to build prediction intervals with boosted forests and 15 method variations (three splitting rules with the CART paradigm which are LS, L_1 and SPI, and five methods to build prediction intervals which are LM, SPI, Quant, HDR and CHDR) proposed by Roy and Larocque (2020). These 15 methods had been thoroughly investigated before through simulation studies and with real data sets in Roy and Larocque (2020). However, they are not easily available to use. One of the main contributions of our package is the implementation of these competitive methods and the ability for users to compare various

prediction interval methods within the same package. The other main contribution of our paper is a new method to build prediction intervals. Contrary to the 15 methods proposed by Roy and Larocque (2020), the newly introduced method was not tested before. That is why in the paper we placed a greater emphasis on investigating the new method through extensive simulation studies and with real data. For performance comparison purposes, we compared the new method to 10 existing methods which include:

- 5 of the 15 implemented method variations of Roy and Larocque (2020); see the Competing methods subsection for details.

- 5 other competing methods from the literature: Quantile Regression Forests (QRF), Generalized Random Forests (GRF), Split Conformal prediction method (SC), Out-of-Bag (OOB) prediction intervals method and $\widehat{PI}_{\alpha}(x)$ method.

The new proposed method to build Prediction Intervals with Boosted Forests is called PIBF. This approach integrates the idea of using the nearest neighbour out-of-bag observations to estimate the conditional prediction error distribution presented in Lu and Hardin (2021) to the one-step boosted forest proposed by Ghosal and Hooker (2021). We will show in this paper that PIBF significantly improves the performance of prediction intervals with random forests when compared with 10 existing methods using a variety of simulated and real benchmark data sets.

The rest of the paper is organized as follows. In the next section, we describe the algorithm implemented in PIBF. We then present the details of the package and provide a practical and reproducible example. We also perform a simulation study to compare the performance of our proposed method to existing competing methods, and we investigate the performance of the proposed method with real data sets. Lastly, we conclude with a discussion of the results.

2.2 Method and implementation

The proposed method is based on the one-step boosted forest method proposed by Ghosal and Hooker (2021). It consists in fitting two regression random forests: the first is fitted to get point predictions and out-of-bag (OOB) residuals using the given data set, whereas the second is fitted to predict those residuals using the original covariates. As empirical studies demonstrate, the one-step boosted forest provides point predictions with reduced bias compared to the simple random forest. Ghosal and Hooker (2021) use subsampling for their theoretical investigations, even though random forests were originally described with bootstrap samples and obtain notable performance improvements. They have also investigated the effect of using bootstrapping with the one-step boosted forest on the bias estimations. From the results presented in their appendix, the use of bootstrapping yields the best performance and reduces the bias the most in exchange for an increase in their proposed variance estimator, which is defined under asymptotic normality. In this paper, we use bootstrapping for the one-step boosted forest method following the better performance results on bias reduction. The final prediction for a new observation, x_{new}, is the sum of the predictions from the two random forests

$$\hat{y}^*_{new} = \hat{y}_{new} + \hat{\varepsilon}_{new}, \tag{2.2}$$

where $\hat{y}_{new}$ is the point prediction obtained from the first random forest and $\hat{\varepsilon}_{new}$ is the bias estimation from the second forest.

Besides bias correction, we use the second random forest as a way to construct a prediction interval by finding the nearest neighbour observations that are close to the one we want to predict. The idea of finding the nearest neighbour observations, a concept very similar to the 'nearest neighbour forest weights' (Hothorn et al., 2004; Lin and Jeon, 2006), was introduced in Moradian et al. (2017) and later used in Moradian et al. (2019), Roy and Larocque (2020), Tabib and Larocque (2020) and Alakuş et al. (2021). For a new observation, the set of in-bag training observations that are in the same terminal nodes as the new observation forms the set of nearest neighbour observations. Roy and Larocque (2020) called this set of observations the Bag of Observations for Prediction (BOP). We

can define the BOP for a new observation x_{new} as

$$BOP\left(x_{new}\right) = \bigcup_{b=1}^{B} I_b\left(x_{new}\right), \tag{2.3}$$

where $I_b\left(x_{new}\right)$ is the set of in-bag training observations, *i.e.*, observations in the bootstrap sample that are in the same terminal node as x_{new} in the b^{th} tree. $I_b\left(.\right)$ consists of the training observations that are in the bootstrap sample of the b^{th} tree.

Instead of forming the set of nearest neighbour observations with the in-bag training observations, we can use the out-of-bag observations which are not in the bootstrap sample, as used in Lu and Hardin (2021). We can define the out-of-bag equivalent of the BOP for a new observation x_{new} (2.3) as

$$BOP^{*}\left(x_{new}\right) = \bigcup_{b=1}^{B} O_b\left(x_{new}\right), \tag{2.4}$$

where $O_b\left(x_{new}\right)$ is the set of out-of-bag observations that are in the same terminal node as x_{new} in the b^{th} tree. $O_b\left(.\right)$ consists of the training observations that are not in the bootstrap sample of the b^{th} tree.

Out-of-bag observations are not used in the tree growing process. Thus, for the trees where the training observations are out-of-bag, they are like the unobserved test observations for those trees. The only difference is that, for a new observation, we use all the trees in the forest whereas for an out-of-bag observation we have only a subset of the forest trees. By using the out-of-bag equivalent of the BOP for a new observation, we can make use of the analogy between the out-of-bag observations and test observations. The out-of-bag neighbours of a new observation represent the new observation better than the in-bag neighbours.

Any desired measure can be obtained by using the constructed BOPs. In this paper, we use the BOP idea to build a prediction interval for a test observation. For a new observation with covariates x_{new}, we firstly form $BOP^{*}\left(x_{new}\right)$ (2.4) using the out-of-bag neighbours. Then, as proposed by Lu and Hardin (2021), we estimate the conditional prediction error distribution, $\hat{F}\left(x_{new}\right)$, but now with the bias-corrected out-of-bag residuals of the observations in $BOP^{*}\left(x_{new}\right)$. Lastly, we build a prediction interval for the new

observation as

$$PI\left(x_{new}\right)=\left[\hat{y}_{new}^{*}+SPI_{\alpha}^{l}\left(\hat{F}\left(x_{new}\right)\right),\hat{y}_{new}^{*}+SPI_{\alpha}^{u}\left(\hat{F}\left(x_{new}\right)\right)\right], \tag{2.5}$$

where $\hat{y}_{new}^{*}$ is the bias-corrected prediction, $SPI_{\alpha}^{l}\left(\hat{F}\left(x_{new}\right)\right)$ and $SPI_{\alpha}^{u}\left(\hat{F}\left(x_{new}\right)\right)$ are the lower and upper bounds of the $SPI_{\alpha}\left(\hat{F}\left(x_{new}\right)\right)$, which is the shortest interval formed by the observations in $BOP^{*}\left(x_{new}\right)$ that contains at least $(1-\alpha)\,100\%$ of the observations. By using the bias-corrected residuals to form prediction error distribution and picking the shortest interval among the qualified intervals, we can expect narrower prediction intervals.

We can summarize the steps of the proposed method as follows:

1. Train the first regression RF with covariates X to predict the response variable Y, and get the OOB predictions $\hat{Y}_{oob}$
2. Compute the OOB residuals as $\hat{\varepsilon}_{oob}=Y-\hat{Y}_{oob}$
3. Train the second regression RF with covariates X to predict the OOB residuals $\hat{\varepsilon}_{oob}$, and get the OOB predictions for residuals $\hat{\hat{\varepsilon}}_{oob}$
4. Update the OOB predictions as $\hat{Y}_{oob}^{*}=\hat{Y}_{oob}+\hat{\hat{\varepsilon}}_{oob}$
5. Compute the updated OOB residuals after bias-correction as $\hat{\varepsilon}_{oob}^{*}=Y-\hat{Y}_{oob}^{*}$
6. For a new observation x_{new}, get the point predictions $\hat{y}_{new}$ from the first RF, and get the predicted residuals $\hat{\varepsilon}_{new}$ from the second RF, then the final prediction for the new observation is

   $$\hat{y}_{new}^{*}=\hat{y}_{new}+\hat{\varepsilon}_{new}$$

   where $\hat{\varepsilon}_{new}$ is the estimated bias.
7. Form a BOP for x_{new} with the OOB neighbours using the second RF, $BOP^{*}\left(x_{new}\right)$ (2.4) and estimate the conditional prediction error distribution for x_{new} as

   $$\hat{F}\left(x_{new}\right)=\left\{\hat{\varepsilon}_{oob,i}^{*}|i\in BOP^{*}\left(x_{new}\right)\right\}$$

8. Build a PI for x_{new} as $PI\left(x_{new}\right)=\left[\hat{y}_{new}^{*}+SPI_{\alpha}^{l}\left(\hat{F}\left(x_{new}\right)\right),\hat{y}_{new}^{*}+SPI_{\alpha}^{u}\left(\hat{F}\left(x_{new}\right)\right)\right]$

2.2.1 Calibration

The principal goal of any prediction interval method is to ensure the desired coverage level. In order to attain the desired coverage level $(1-\alpha)$, we may need a calibration procedure. The goal of the calibration is to find the value of α_w, called the working level in Roy and Larocque (2020), such that the coverage level of the PIs for the training observations is closest to the desired coverage level. Roy and Larocque (2020) presented a calibration procedure that uses the BOPs that are built using only the trees where the training observation x_i is OOB. The idea is to find the value of α_w using the OOB-BOPs. In this paper, we call this procedure OOB calibration.

We also include a cross-validation-based calibration procedure with the proposed method to acquire the desired $(1-\alpha)$ coverage level. In this calibration, we apply k-fold cross-validation to form prediction intervals for the training observations. In each fold, we split the original training data set into training and testing sets. For the training set, we go through the steps 1-5 defined above. Then, for each observation in the testing set, we apply steps 6-8 and build a PI. After completing CV, we compute the coverage level with the constructed PIs and if the coverage is not within the acceptable coverage range, then we apply a grid search to find the α_w such that α_w is the closest to the target α among the set of α_w's. Once we find the α_w, we use this level to build the PI for the new observations.

2.2.2 The `RFpredInterval` package

In our package, we implement 16 methods that apply random forest training. Ten of these methods have specialized splitting rules in the random forest growing process. These methods are the ones with L_1 and shortest prediction interval (SPI) splitting rules proposed by Roy and Larocque (2020). To implement these methods, we have utilised the custom split feature of the `randomForestSRC` package (Ishwaran and Kogalur, 2021).

The `randomForestSRC` package allows users to define a custom splitting rule for the tree growing process. The user needs to define the customized splitting rule in the

`splitCustom.c` file with C-programming. After modifying the `splitCustom.c` file, all C source code files in the package's `src` folder must be recompiled. Finally, the package must be re-installed for the custom split rule to become available.

In our package development process, we froze the version of `randomForestSRC` to the latest one available at the time, which is version 2.11.0, to apply specialized splitting rules. After defining the L_1 and SPI splitting rules, all C files were re-compiled. Finally, all package files including our R files for prediction interval methods were re-built to make the package ready for the user installation.

The `RFpredInterval` package has two main R functions as below:

- `pibf()`: Constructs prediction intervals with the proposed method, PIBF.
- `rfpi()`: Constructs prediction intervals with 15 distinct variations proposed by Roy and Larocque (2020).

Table 2.1 presents the list of functions and methods implemented in `RFpredInterval`. For `pibf()`, `RFpredInterval` uses the CRAN package `ranger` (Wright et al., 2020) to fit the random forests. For `rfpi()`, `RFpredInterval` uses `randomForestSRC` package. For the least-squares splitting rule, both `randomForestSRC` and `ranger` packages are applicable.

Table 2.1: List of functions and methods with characteristics

Function	Method		Details
`pibf()`	PIBF		Builds prediction intervals with boosted forests. The `ranger` package is used to fit the random forests. Calibration options are `"cv"` and `"oob"`. Returns constructed PIs and bias-corrected point predictions for the test data.
`rfpi()`	*Split method*	*PI method*	
	LS	LM SPI Quant HDR CHDR	Splitting rule is the least-squares (LS) from the CART paradigm. `"ls"` is used for the `"split_method"` argument. A vector of characters `c("lm","spi","quant","hdr","chdr")` is used for the `"pi_method"` argument to apply all or a subset of the PI methods. `ranger` or `randomForestSRC` can be used to fit the random forest. Returns a list of constructed PIs for the selected PI methods and point predictions for the test data.
	L_1	LM SPI Quant HDR CHDR	Splitting rule is the L_1 from the CART paradigm (Roy and Larocque, 2020). `"l1"` is used for the `"split_method"` argument. A vector of characters `c("lm","spi","quant","hdr","chdr")` is used for the `"pi_method"` argument to apply all or a subset of the PI methods. Only `randomForestSRC` can be used to fit the random forest since the split rule is implemented with the custom split feature of that package. Returns a list of constructed PIs for the selected PI methods and point predictions for the test data.
	SPI	LM SPI Quant HDR CHDR	Splitting rule is the shortest PI (SPI) from the CART paradigm (Roy and Larocque, 2020). `"spi"` is used for the `"split_method"` argument. A vector of characters `c("lm","spi","quant","hdr","chdr")` is used for the `"pi_method"` argument to apply all or a subset of the PI methods. Only `randomForestSRC` can be used to fit the random forest since the split rule is implemented with the custom split feature of that package. Returns a list of constructed PIs for the selected PI methods and point predictions for the test data.
`piall()`	All methods		Builds prediction intervals with all of the implemented PI methods. The `ranger` package is used to fit the random forests for the PIBF and methods with LS split rule. `randomForestSRC` package is used for the methods with L_1 and SPI split rules. Default values are assigned to the function arguments of `pibf()` and `rfpi()`. Returns an object of class `"piall"` containing a list of constructed PIs with 16 methods, and point predictions obtained with the PIBF method, LS, L_1 and SPI split rules for the test data.
`plot()`			Plots the 16 constructed PIs obtained with `piall()` function for a test observation.
`print()`			Prints the summary output of `pibf()`, `rfpi()` and `piall()` functions.

LM: Classical method, **SPI**: Shortest PI, **Quant**: Quantiles, **HDR**: Highest density region, **CHDR**: Contiguous HDR

In this section, we illustrate the usage of the `RFpredInterval` package with the Ames Housing data set (De Cock, 2011). The data set was introduced as a modern alternative to the well-known Boston Housing data set. The data set contains many explanatory variables on the quality and quantity of physical attributes of houses in Ames, IA sold from 2006 to 2010. Most of the variables give information to a typical home buyer who would like to know about a house (*e.g.* number of bedrooms and bathrooms, square footage, heating type, lot size, etc.).

The `AmesHousing` (Kuhn, 2020) package contains the raw data and processed versions of the Ames Housing data set. The raw data contains 2930 observations and 82 variables, which include 23 nominal, 23 ordinal, 14 discrete, and 20 continuous variables, involved in assessing house values. The processed version of the data set has 2330 observations and 81 variables, including the target variable `Sale_Price` representing the value of houses in US dollars. The usual goal for this data set is to predict the sale price of each house given covariates.

We load the processed version of the Ames Housing data set from the `AmesHousing` package and prepare the data set that we will use for the analyses. The preprocessing steps are presented in Appendix G. This version of the data set contains 22 factors and 59 numeric variables, including 1 response variable `Sale_Price`, for 2929 observations. We split the data set into training and testing samples.

```
set.seed(3456)
n <- nrow(AmesHousing)
trainindex <- sample(1:n, size = round(0.7*n), replace = FALSE)
traindata <- AmesHousing[trainindex, ]
testdata <- AmesHousing[-trainindex, ]
```

We fit a random forest with 1000 trees using the training data and construct 95% prediction intervals for the observations in the testing data with the proposed method. We apply 5-fold cross-validation based calibration and set the acceptable coverage range to $[.945, .955]$. We can pass the list of random forest parameters for `ranger` package.

```
out <- pibf(formula = Sale_Price ~ .,
            traindata = traindata,
            testdata = testdata,
            alpha = 0.05,
            calibration = "cv",
            numfolds = 5,
            coverage_range = c(0.945, 0.955),
            params_ranger = list(num.trees = 1000),
            oob = TRUE)
```

We can then analyze the constructed PIs and bias-corrected random forest predictions for the testing data, as shown below. The PI output is a list containing lower and upper bounds. For example, we can print the point prediction and prediction interval for the tenth observation in the testing data.

```
out$pred_interval
out$test_pred
c(out$pred_interval$lower[10], out$test_pred[10],
  out$pred_interval$upper[10])
[1] 133.8629 160.2426 194.5804
```

We can also print the summary output. In the summary output, we can always see the mean PI length over the test data set. If calibration is applied, we can see the working level of α. If the test data set has true response information, as in our example, coverage and prediction errors for the test set are also printed. Moreover, since we have entered `oob = TRUE` in the function arguments, in the summary output we can see the mean PI length and coverage measures along with the prediction errors for the training set. The prediction intervals are built with the out-of-bag (OOB) predictions and prediction errors.

```
print(out)
```

```
>                       alpha_w: 0.050
>                Mean PI length: 73.081
>                      Coverage: 96.8%
>       MAE of test predictions: 12.773
>      RMSE of test predictions: 19.545
>
>      Mean PI length (OOB PIs): 74.823
>            Coverage (OOB PIs): 94.7%
>  MAE of OOB train predictions: 14.179
> RMSE of OOB train predictions: 23.875
```

Next, we construct 95% prediction intervals using the variations proposed by Roy and Larocque (2020). In the following example, the splitting is rule is set to L_1 and we want to apply LM, Quant and SPI methods for building prediction intervals. We apply OOB calibration and set the acceptable coverage range to $[.945, .955]$. We can pass the the list of random forest parameters for `randomForestSRC` package.

```
out2 <- rfpi(formula = Sale_Price ~ .,
             traindata = traindata,
             testdata = testdata,
             alpha = 0.05,
             calibration = TRUE,
             split_rule = "l1",
             pi_method = c("lm", "quant", "spi"),
             params_rfsrc = list(ntree = 1000),
             params_calib = list(range = c(0.945, 0.955)),
             oob = FALSE)
```

We can analyze the constructed PIs for the testing data as below. Each PI output is a list containing lower and upper bounds. For instance, we can print the point prediction

and LM prediction interval for the tenth observation in the testing data.

```
out2$lm_interval
out2$quant_interval
out2$spi_interval
c(out2$lm_interval$lower[10], out2$test_pred[10],
  out2$lm_interval$upper[10])
[1] 129.9474 154.2098 176.8429
```

We print the summary output. In the summary output, we can see the splitting rule selected in the first row. Since the test data set has true responses in our example, we can see the coverage information for the selected PI methods besides the mean PI length and α_w in the printed table. Below the table, we have the mean prediction errors for the test set.

```
print(out2)
```

```
>                    Split rule: L1
> ------------------------------------------------------------------
>                                  Mean PI length  Coverage  alpha_w
> Classical method (LM)                81.641        96.2%    0.140
> Shortest prediction interval (SPI)   81.953        96.1%    0.100
> Quantile method (Quant)              80.893        95.8%    0.120
> ------------------------------------------------------------------
>       MAE of test predictions: 14.605
>      RMSE of test predictions: 22.603
```

Although, with the `pibf()` and `rfpi()` functions, we have more flexibility to set the arguments for the methods, we can build prediction intervals with all 16 methods implemented in the package with the `piall()` function. We will build 95% prediction intervals for the test set.

```
out3 <- piall(formula = Sale_Price ~ .,
              traindata = traindata,
              testdata = testdata,
              alpha = 0.05,
              num.trees = 1000)
```

The output is a list of constructed prediction intervals with 16 methods and point predictions obtained with the PIBF method, LS, L_1, and SPI split rules. Hence, the output includes 16 prediction intervals and 4 point predictions which is a list of 20 items in total.

We print the summary output.

```
print(out3)
```

```
> ----------------------------------------
>               Mean PI length    Coverage
> PIBF              72.752          96.6%
> LS-LM             81.800          96.5%
> LS-SPI            82.662          95.4%
> LS-Quant          81.523          95.2%
> LS-HDR            81.733          96.0%
> LS-CHDR           83.025          96.1%
> L1-LM             81.649          96.1%
> L1-SPI            81.805          96.1%
> L1-Quant          80.836          95.3%
> L1-HDR            83.032          96.4%
> L1-CHDR           82.482          96.1%
> SPI-LM            81.578          96.0%
> SPI-SPI           81.960          96.2%
> SPI-Quant         81.036          95.4%
> SPI-HDR           84.404          96.6%
```

```
> SPI-CHDR          82.927           96.1%
> ----------------------------------------
>                     MAE             RMSE
> PIBF              12.774          19.428
> LS split          14.412          22.413
> L1 split          14.596          22.569
> SPI split         14.558          22.600
```

Lastly, we plot the constructed prediction intervals with all 16 methods, for the 15th observation in the test set.

```
plot(out3, test_id = 15)
```

Figure 2.1 presents the prediction intervals and point predictions for the test observation. The methods are ordered in the y-axis based on their resulting PI length. For each method, the red point presents the point prediction and blue lines show the constructed prediction interval(s) for the test observation. If the true response of the test observation is known, it is demonstrated with a dashed vertical line. Note that we may have multiple prediction intervals with the HDR PI method. As we can see from the figure, we may have four different point predictions for the same test observation. The PIBF method and the three splitting rules LS, L_1 and SPI can produce different point predictions. But all PI method variations for the same splitting rule have the same point prediction.

2.3 Simulation study

In this section, we compare the predictive performance of the prediction intervals constructed with our proposed method to the existing methods presented in the Introduction using a variety of simulated and real benchmark data sets.

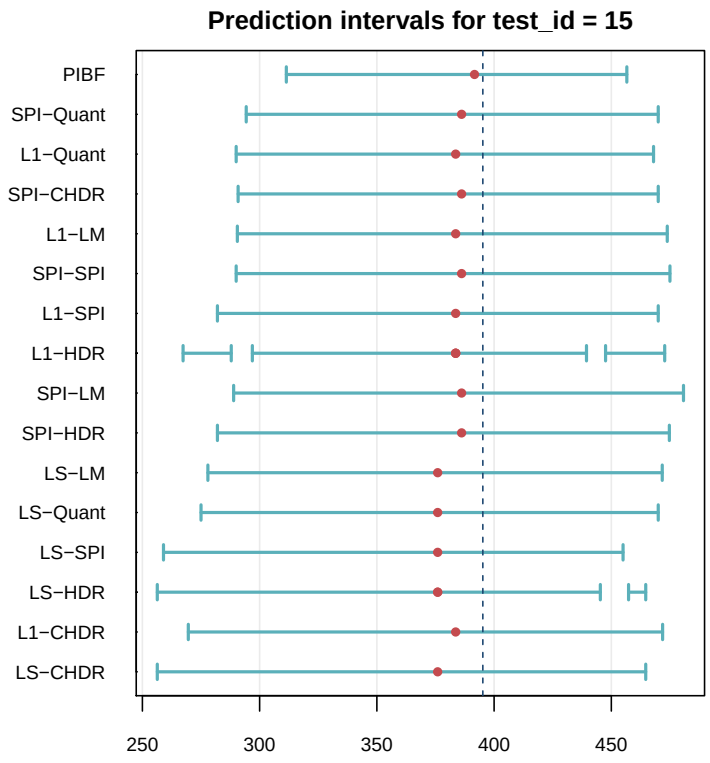

Figure 2.1: Prediction intervals for the 15th test observation in the test data. The x-axis represents the sale price of houses in thousands. For each method, red dots represent the point prediction and blue lines show the prediction interval(s). The vertical dashed line shows the true response value for the test observation. PIBF: Prediction intervals with boosted forests (the proposed method). The notation for the other 15 methods is *split rule-PI method*. Splitting rules are LS: Least-squares, L1: L_1, SPI: Shortest PI split rule. PI methods are LM: Classical method, Quant: Quantiles, SPI: Shortest PI, HDR: Highest density region, CHDR: Contiguous HDR.

2.3.1 Simulation design

We apply a simulation study based on seven simulated data sets from the literature. The first three of the data sets are Friedman's benchmark regression problems described in Jerome H. Friedman (1991) and Breiman (1996). We use the CRAN package `mlbench` (Leisch and Dimitriadou, 2021) to generate these data sets.

In Friedman Problem 1, the inputs are 10 independent variables uniformly distributed on the interval $[0,1]$. The first five covariates are used to generate the response:

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \varepsilon,$$

where ε is $N\left(0,\sigma^2\right)$ and the default standard deviation of ε is 1 which yields a signal-to-noise ratio (SNR) (*i.e.*, the ratio of the standard deviation of signal to the standard deviation of error) of 4.8:1.

In Friedman Problem 2, the response is generated as

$$y = \left(x_1^2 + \left(x_2 x_3 - \frac{1}{x_2 x_4}\right)^2\right)^{0.5} + \varepsilon,$$

where the inputs are four independent variables uniformly distributed over the ranges

$$\begin{aligned} 0 \leq & x_1 \leq 100 \\ 40\pi \leq & x_2 \leq 560\pi \\ 0 \leq & x_3 \leq 1 \\ 1 \leq & x_4 \leq 11 \end{aligned}$$

and ε is $N\left(0,\sigma^2\right)$. The default value of 125, which yields a SNR of 3:1, is used for the standard deviation of ε.

In Friedman Problem 3, the inputs are four independent variables uniformly distributed over the same ranges as Friedman Problem 2. The response is generated as

$$y = \arctan\left(\frac{x_2 x_3 - \frac{1}{x_2 x_4}}{x_1}\right) + \varepsilon,$$

where ε is $N\left(0,\sigma^2\right)$ and the default value of 0.01 for the standard deviation of ε is used, which yields a SNR of 3:1.

The fourth data set is the Peak Benchmark Problem which is also from the `mlbench` package. Let $r = 3u$ where u is uniform on $[0,1]$ and let x be uniformly distributed on the d-dimensional sphere of radius r. The response is $y = 25\exp\left(-0.5r^2\right)$. The default value of $d = 20$ dimensions is used.

The fifth one is a modification of Friedman Problem 1, which was used in Hothorn and Zeileis (2021) in their H2c setup. This data set was designed to have heteroscedasticity. The inputs are 10 independent variables uniformly distributed on the interval $[0,1]$.

The first five covariates are used in the mean function and the unscaled mean function is defined as

$$\mu = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5.$$

Then, the scaled mean function on the interval $[-1.5, 1.5]$ is

$$\mu^S = \frac{3(\mu - \mu_{min})}{\mu_{max} - \mu_{min}} - 1.5,$$

where μ_{min} and μ_{max} are the minimum and maximum values of μ over the sample. The last five covariates are used in the standard deviation function and the unscaled standard deviation function is

$$\sigma = 10\sin(\pi x_6 x_7) + 20(x_8 - 0.5)^2 + 10x_9 + 5x_{10}.$$

The standard deviation is scaled as

$$\sigma^S = \exp\left(\frac{3(\sigma - \sigma_{min})}{\sigma_{max} - \sigma_{min}} - 1.5\right),$$

where σ_{min} and σ_{max} are the minimum and maximum values of σ over the sample. The response is generated as a normal random variable with mean μ^S and standard deviation σ^S.

The last two data sets, which were used in Roy and Larocque (2020), have a tree-based response variable. The inputs are seven independent variables generated from the standard normal distribution. The response is generated with the seven covariates according to a tree model with a depth of three, with eight terminal nodes:

$$\begin{aligned} y =& u_1 I(x_1 < 0, x_2 < 0, x_4 < 0) \\ &+ u_2 I(x_1 < 0, x_2 < 0, x_4 \geq 0) \\ &+ u_3 I(x_1 < 0, x_2 \geq 0, x_5 < 0) \\ &+ u_4 I(x_1 < 0, x_2 \geq 0, x_5 \geq 0) \\ &+ u_5 I(x_1 \geq 0, x_3 < 0, x_6 < 0) \\ &+ u_6 I(x_1 \geq 0, x_3 < 0, x_6 \geq 0) \\ &+ u_7 I(x_1 \geq 0, x_3 \geq 0, x_7 < 0) \\ &+ u_8 I(x_1 \geq 0, x_3 \geq 0, x_7 \geq 0) + \varepsilon, \end{aligned}$$

where the terminal node means are $u = (5, 10, 15, 20, 25, 30, 35, 40)$ and I is the indicator function. The difference in the two data sets is the distribution of the error. In the first one, ε is generated from a standard normal distribution and in the other it is from an exponential distribution with mean 1. The signal-to-noise ratio is 11.5:1 for both data sets.

We use training sample sizes of $n_{train} = \{200, 500, 1000, 5000\}$, resulting in 28 scenarios. Each scenario is repeated 500 times. In each run, we generate an independent test set of new observations with $n_{test} = 1000$.

2.3.2 Competing methods

We compare our proposed prediction interval estimator with 10 competing methods which were presented in the Introduction. The first is the $\widehat{PI}_\alpha$ method. We fit the random forest with the `ranger` package and use the `forestError` package to build PIs. The second is the OOB method. The `rfinterval` package is used. The third is the split conformal method. The `conformalInference` package is used. The fourth is the QRF method and the `quantregForest` package is used. The fifth is the GRF method. The function `quantile_forest` in the `grf` package is used.

The last five are variations of Roy and Larocque (2020). To compare the performance of the method variations, a comprehensive simulation study and real data analyses were performed in Roy and Larocque (2020). One of the biggest conclusions from these comparison studies was that, among the three alternative splitting criteria within the CART paradigm, the impact of the choice of the splitting rule on the performance of the prediction intervals was moderate whereas the selection of the PI method had a much greater impact on the performance. Hence, in this simulation study, we set up the splitting rule to the least-squares (LS) and only compare the five PI methods: LM, Quant, SPI, HDR, and CHDR. For those methods, the `rfpi()` function of the `RFpredInterval` package is used. We fit the random forest with the `ranger` package.

2.3.3 Parameter settings

For the simulations, we use the following parameters. For all methods, we set the number of trees to 2000. Letting p be the number of covariates, then the number of covariates to randomly split at each node, *mtry*, is set to $\max\{\lfloor p/3 \rfloor, 1\}$ (except for the GRF method). For the GRF method, following Athey et al. (2019), *mtry* is set to $\min\left\{\lceil \sqrt{p}+20 \rceil, p\right\}$. Also, we use the honest splitting regime with the default fraction of 0.5 for the GRF method. The minimum node size parameter for all forests is set to 5. The desired coverage is set to 95% for all methods. For the proposed method, we perform the cross-validation-based calibration as the primary calibration procedure, but we also investigate the OOB calibration. For the method variations in Roy and Larocque (2020), we perform the OOB calibration procedure as they proposed. For both calibration procedures, the acceptable range of coverage is set to $[.945, .955]$. Calibration is not performed for the competing methods since no option for calibration is offered in their CRAN packages.

2.3.4 Performance with the simulated data sets

We can evaluate the performance of the competing methods with two measures: the mean coverage and the mean prediction interval length. Table 2.2 presents the average coverage rate of each method on the test set over 500 replications for a given simulated data set and sample size, with average mean prediction interval lengths shown in parentheses. The principal goal of any prediction interval method is to ensure the desired coverage level. In this simulation study, the desired coverage level is set to 95% for all methods. The left plot in Figure 2.2 shows the mean coverages over the 28 scenarios for all methods. Overall, all of the methods, except the QRF and GRF methods which tend to be conservative, provide a mean coverage close to the desired level. QRF and GRF methods have an average mean coverage of 0.975 and 0.974 over all scenarios, respectively. Although the $\widehat{PI}_{\alpha}$ method has an average of the mean coverages of 0.957, close to the desired level, its variability is large. Over 308 (11 methods $\times$ 28 scenarios) average coverage values, there is only one case where the mean coverage is below 0.94. It corresponds to the $\widehat{PI}_{\alpha}$ method in

Friedman Problem 2 with $n_{train} = 5000$.

Once the prediction intervals provide the desired coverage level, the next goal of any PI method is to provide the shortest PI length. Prior to carrying out a detailed comparison of interval lengths, we can globally compare the interval lengths over all scenarios with the percentage increase in mean PI length of a method with respect to the best method for a given run. For a given run, define ml_i as the mean PI length of method i and ml^* as the shortest mean PI length over the 10 competing methods. The percentage increase in PI length for method i is computed as $100 \times (ml_i - ml^*)/ml^*$. Smaller values for this measure indicate better performances. The right plot in Figure 2.2 presents the relative lengths of the methods across 14,000 runs (28 scenarios $\times$ 500 replications). The prediction intervals with the GRF method are the widest, followed by the QRF method. However, as we saw in the left plot in Figure 2.2, GRF and QRF produce conservative prediction intervals, so their PI lengths cannot be fairly compared to the other methods with a coverage closer to 0.95. Based on the global results, the proposed method, PIBF, performs the best. Following PIBF, $\widehat{PI}_\alpha$, OOB, LM, SPI, HDR and CHDR perform similarly well, with $\widehat{PI}_\alpha$ being slightly better. Among the variations of Roy and Larocque (2020), the PIs with quantiles produce longer prediction intervals than the other four variations.

Now, we investigate the performance of the methods separately for each scenario. Figures 2.3 to 2.9 present the mean PI length results of each method for each simulated data set. Each figure has four facets corresponding to the four levels of the training sample size. For all methods and data sets, the mean PI lengths and their variability decrease as the sample size increases (except the GRF method for the tree-based data sets). We see that for Friedman Problem 1, from Figure 2.3, for all sample sizes, PIBF consistently outperforms the 10 competing methods in terms of mean PI length while ensuring the desired coverage level (see Table 2.2 for the mean coverage results). QRF and GRF provide the widest prediction intervals for all sample sizes. However, as presented in Table 2.2, these methods heavily over-cover and are therefore not comparable with the other methods. While the $\widehat{PI}_\alpha$ method also slightly over-covers, it has shorter PIs than other methods.

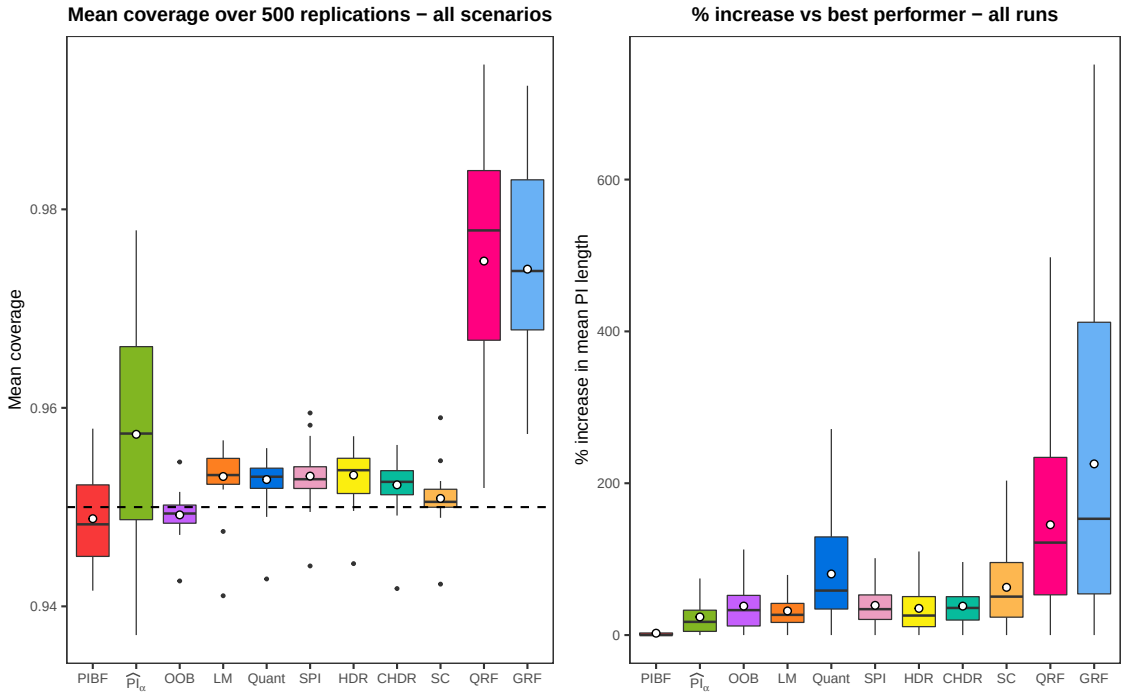

Figure 2.2: *(Left)* Boxplots for the mean coverage over all scenarios. All methods, except the QRF and GRF methods, are able to provide a mean coverage close to the desired coverage level of 0.95. Each white circle is the average of the mean coverages over 28 scenarios. *(Right)* Boxplots for the percentage increase in mean PI length of each method compared to the shortest PI length for a given run across 14,000 runs. The smallest is the percentage increase, the better is the method. Each white circle is the average of the relative lengths over 14,000 runs. Since the outlier values are distorting the scales, they are removed from the graph. PIBF: Prediction intervals with boosted forests (the proposed method), $\widehat{PI}_\alpha$: Conditional α-level prediction interval, OOB: Out-of-Bag approach, LM: Classical method, Quant: Quantiles, SPI: Shortest PI, HDR: Highest density region, CHDR: Contiguous HDR, SC: Split conformal, QRF: Quantile regression forest, GRF: Generalized random forest.

For Friedman Problem 2 (Figure 2.4), we see that the proposed method has the shortest mean PI length for the smallest sample size, and as the sample size increases $\widehat{PI}_\alpha$ provides shorter PIs. However, we should also take into account the mean coverages presented in Table 2.2. The proposed method has smaller coverage levels for $n_{train} = 200$ compared to $\widehat{PI}_\alpha$, but as the sample size increases the coverage levels decrease for the $\widehat{PI}_\alpha$ method (up to 0.937 for $n_{train} = 5000$) whereas PIBF keeps it around 0.945. For $n_{train} = 5000$, the OOB method builds shorter PIs while ensuring the desired coverage level. Again, QRF and GRF have the widest PIs for all sample sizes due to their conservative PIs.

The performance of PIBF and $\widehat{PI}_\alpha$ is very similar for Friedman Problem 3 (Figure 2.5). Both methods provide the shortest PIs with similar coverage levels and mean PI lengths. Results for the Peak Benchmark Problem presented in Figure 2.6 are very similar to those of Friedman Problem 1. For all sample sizes, PIBF consistently outperforms the 10 competing methods in terms of mean PI length. But this time, SPI method with LS splitting rule comes in second place.

For the H2c setup (Figure 2.7), which is the modification of Friedman Problem 1, we can see that all methods are comparable since QRF and GRF do not over-cover. In this setting, all methods also perform fairly well with respect to PI length. Overall, the LM prediction interval method with the LS splitting rule provides slightly shorter PIs.

For the tree-based data sets (figures 2.8 and 2.9), overall, it seems that the distribution of the error does not have a significant effect on the results. Again, QRF and GRF have conservative PIs. Unlike the other data sets, we see here that the mean PI lengths of the GRF method decrease very slowly as the sample size increases. For $n_{train} = 5000$, all methods (except QRF and GRF) perform similarly. For the smallest sample size, HDR PI building methods and the proposed method perform slightly better than other methods. As the sample size increases, PIBF produces the shortest prediction intervals.

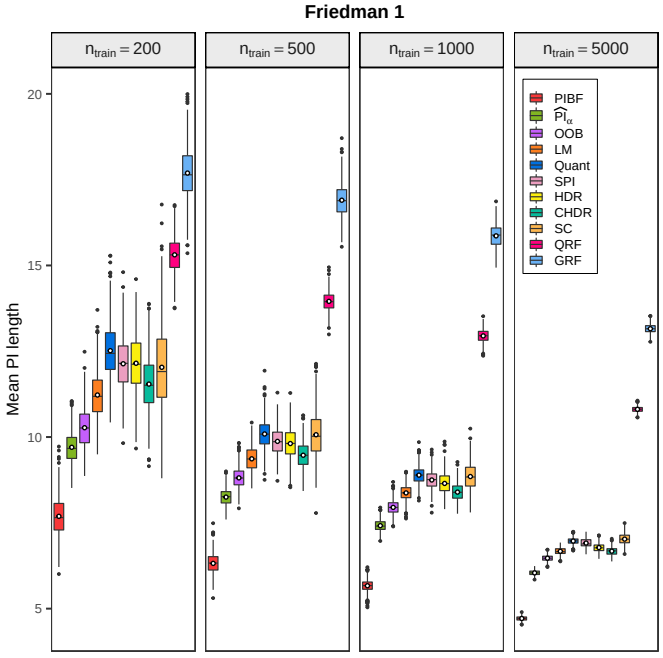

Figure 2.3: Distributions of the mean PI length over the test set across 500 replications for Friedman Problem 1.

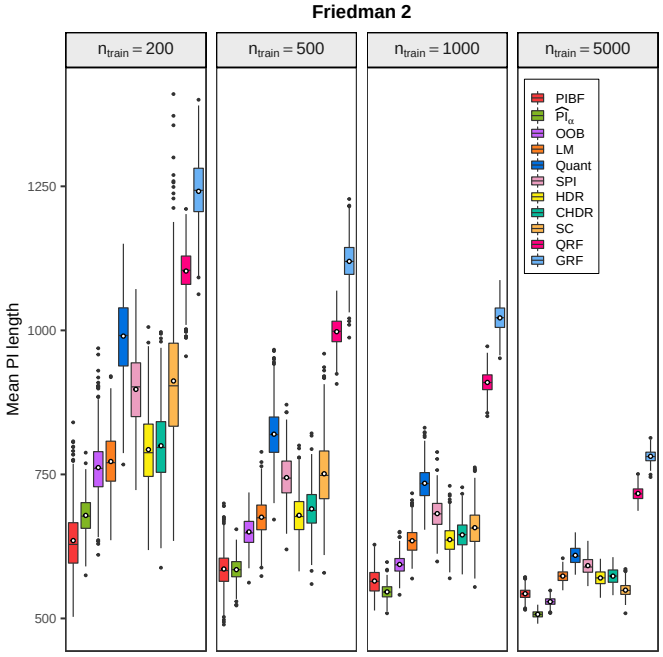

Figure 2.4: Distributions of the mean PI length over the test set across 500 replications for Friedman Problem 2.

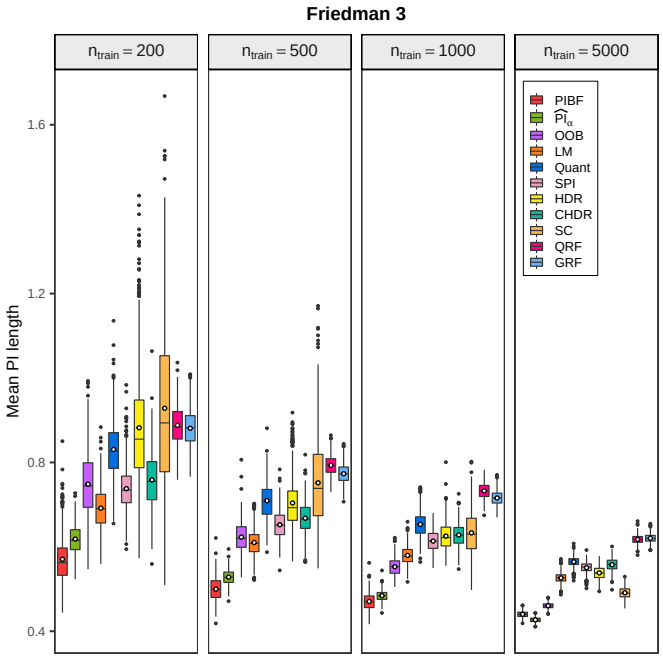

Figure 2.5: Distributions of the mean PI length over the test set across 500 replications for Friedman Problem 3.

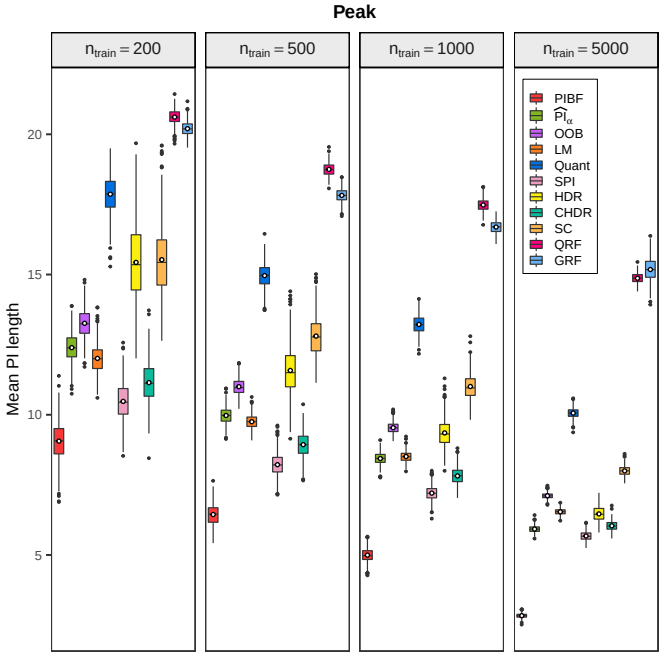

Figure 2.6: Distributions of the mean PI length over the test set across 500 replications for Peak Benchmark Problem.

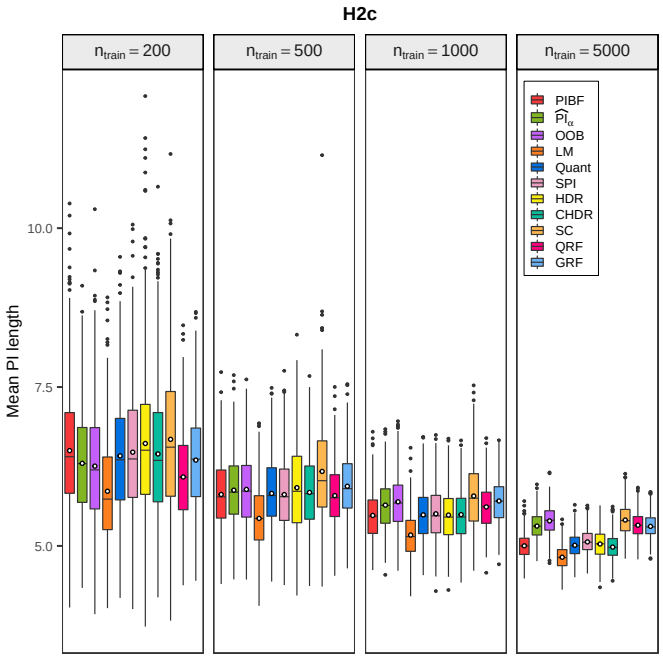

Figure 2.7: Distributions of the mean PI length over the test set across 500 replications for the H2c setup.

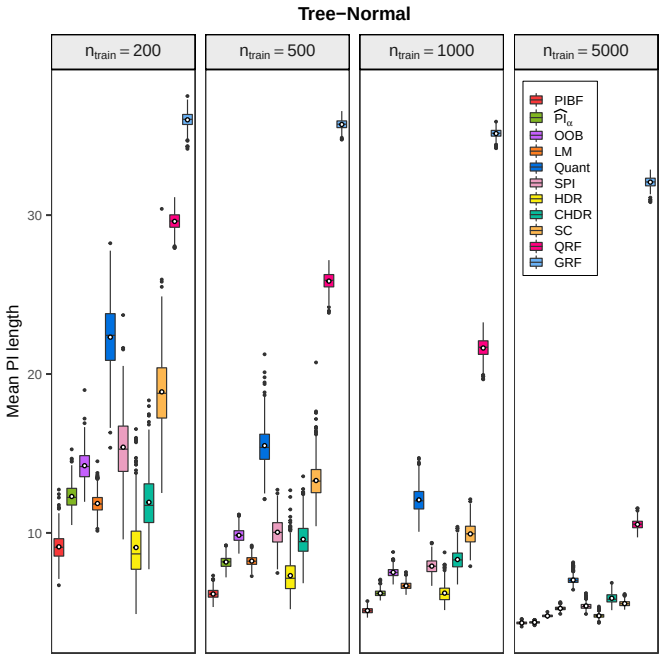

Figure 2.8: Distributions of the mean PI length over the test set across 500 replications for the tree-based problem with normally distributed error.

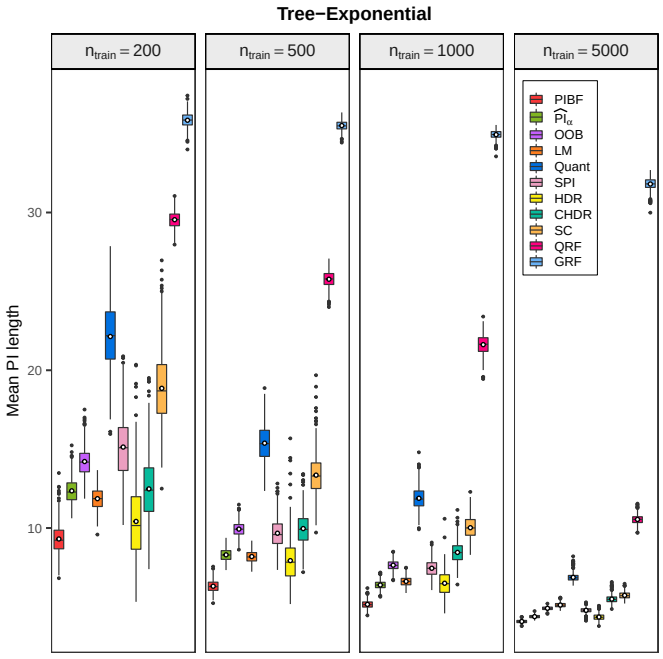

Figure 2.9: Distributions of the mean PI length over the test set across 500 replications for the tree-based problem with exponentially distributed error.

Table 2.2: Results of the simulation study. Average coverage rates of each method in each simulation, with average mean PI lengths shown in parentheses. The desired coverage level is 0.95. Shortest average mean PI lengths are emphasized with bold text. PIBF: Prediction intervals with boosted forests (the proposed method), $\widehat{PI}_\alpha$: Conditional α-level prediction interval, OOB: Out-of-Bag approach, LM: Classical method, Quant: Quantiles, SPI: Shortest PI, HDR: Highest density region, CHDR: Contiguous HDR, SC: Split conformal, QRF: Quantile regression forest, GRF: Generalized random forest.

n_{train}	Data	PIBF	$\widehat{PI}_\alpha$	OOB	LM	Quant	SPI	HDR	CHDR	SC	QRF	GRF
200	Friedman 1	0.952 (**7.69**)	0.959 (9.7)	0.949 (10.3)	0.955 (11.2)	0.956 (12.5)	0.956 (12.1)	0.955 (12.2)	0.956 (11.5)	0.952 (12)	0.978 (15.3)	0.968 (17.7)
	Friedman 2	0.942 (**635**)	0.951 (679)	0.947 (762)	0.956 (772)	0.955 (990)	0.956 (898)	0.955 (793)	0.954 (800)	0.951 (912)	0.969 (1103)	0.972 (1241)
	Friedman 3	0.944 (**0.57**)	0.948 (0.62)	0.949 (0.75)	0.956 (0.69)	0.953 (0.83)	0.952 (0.74)	0.951 (0.88)	0.954 (0.76)	0.953 (0.93)	0.961 (0.89)	0.968 (0.88)
	Peak	0.958 (**9.06**)	0.956 (12.4)	0.949 (13.3)	0.955 (12)	0.956 (17.9)	0.957 (10.5)	0.955 (15.4)	0.955 (11.1)	0.951 (15.5)	0.972 (20.6)	0.978 (20.2)
	H2c	0.954 (6.5)	0.954 (6.3)	0.955 (6.26)	0.955 (**5.86**)	0.956 (6.42)	0.957 (6.47)	0.956 (6.61)	0.955 (6.45)	0.959 (6.68)	0.952 (6.09)	0.957 (6.35)
	Tree-N	0.949 (9.12)	0.967 (12.3)	0.947 (14.2)	0.956 (11.9)	0.956 (22.3)	0.958 (15.4)	0.955 (**9.08**)	0.955 (11.9)	0.951 (18.9)	0.979 (29.6)	0.960 (36)
	Tree-exp	0.949 (**9.31**)	0.967 (12.4)	0.947 (14.2)	0.955 (11.9)	0.956 (22.1)	0.959 (15.1)	0.956 (10.4)	0.956 (12.5)	0.950 (18.9)	0.979 (29.5)	0.960 (35.8)
500	Friedman 1	0.952 (**6.32**)	0.962 (8.25)	0.948 (8.81)	0.953 (9.37)	0.952 (10.1)	0.953 (9.87)	0.952 (9.81)	0.953 (9.47)	0.951 (10.1)	0.986 (14)	0.978 (16.9)
	Friedman 2	0.945 (586)	0.946 (**585**)	0.948 (650)	0.952 (676)	0.952 (820)	0.952 (744)	0.951 (679)	0.951 (690)	0.952 (751)	0.975 (998)	0.979 (1120)
	Friedman 3	0.943 (**0.5**)	0.946 (0.53)	0.948 (0.62)	0.952 (0.61)	0.951 (0.71)	0.951 (0.65)	0.951 (0.7)	0.951 (0.67)	0.952 (0.75)	0.965 (0.79)	0.970 (0.77)
	Peak	0.957 (**6.44**)	0.967 (9.97)	0.951 (11)	0.953 (9.75)	0.954 (15)	0.954 (8.22)	0.956 (11.6)	0.953 (8.93)	0.952 (12.8)	0.980 (18.7)	0.984 (17.8)
	H2c	0.953 (5.81)	0.956 (5.88)	0.951 (5.89)	0.952 (**5.43**)	0.953 (5.83)	0.953 (5.81)	0.954 (5.92)	0.952 (5.84)	0.955 (6.17)	0.954 (5.79)	0.959 (5.94)
	Tree-N	0.948 (**6.16**)	0.969 (8.17)	0.949 (9.85)	0.952 (8.23)	0.953 (15.5)	0.953 (10)	0.954 (7.31)	0.952 (9.6)	0.952 (13.3)	0.984 (25.8)	0.969 (35.7)
	Tree-exp	0.947 (**6.31**)	0.966 (8.3)	0.949 (9.93)	0.952 (8.19)	0.953 (15.4)	0.954 (9.67)	0.954 (7.93)	0.953 (9.96)	0.953 (13.4)	0.984 (25.8)	0.970 (35.5)
1000	Friedman 1	0.953 (**5.67**)	0.964 (7.42)	0.949 (7.95)	0.954 (8.37)	0.954 (8.89)	0.954 (8.75)	0.954 (8.65)	0.953 (8.4)	0.950 (8.85)	0.990 (12.9)	0.985 (15.9)
	Friedman 2	0.945 (565)	0.942 (**546**)	0.950 (594)	0.953 (635)	0.953 (735)	0.953 (682)	0.952 (637)	0.953 (645)	0.951 (658)	0.977 (910)	0.983 (1022)
	Friedman 3	0.944 (**0.47**)	0.945 (0.48)	0.948 (0.55)	0.954 (0.58)	0.952 (0.65)	0.952 (0.61)	0.951 (0.63)	0.952 (0.63)	0.949 (0.63)	0.967 (0.73)	0.971 (0.72)
	Peak	0.957 (**5**)	0.972 (8.44)	0.950 (9.54)	0.952 (8.51)	0.953 (13.2)	0.953 (7.21)	0.956 (9.35)	0.952 (7.82)	0.951 (11)	0.983 (17.5)	0.986 (16.7)
	H2c	0.952 (5.48)	0.955 (5.64)	0.949 (5.69)	0.948 (**5.17**)	0.949 (5.49)	0.950 (5.51)	0.950 (5.49)	0.949 (5.49)	0.949 (5.78)	0.954 (5.61)	0.958 (5.71)
	Tree-N	0.946 (**5.11**)	0.965 (6.2)	0.950 (7.52)	0.954 (6.67)	0.953 (12.1)	0.954 (7.91)	0.954 (6.21)	0.953 (8.33)	0.950 (9.94)	0.985 (21.6)	0.976 (35.1)
	Tree-exp	0.946 (**5.16**)	0.964 (6.39)	0.949 (7.64)	0.954 (6.62)	0.953 (11.9)	0.954 (7.45)	0.955 (6.5)	0.953 (8.45)	0.950 (10)	0.985 (21.6)	0.975 (34.9)
5000	Friedman 1	0.950 (**4.71**)	0.967 (6.04)	0.950 (6.47)	0.953 (6.67)	0.953 (6.97)	0.953 (6.91)	0.954 (6.78)	0.953 (6.67)	0.950 (7.03)	0.995 (10.8)	0.992 (13.2)
	Friedman 2	0.945 (543)	0.937 (**507**)	0.950 (529)	0.952 (574)	0.951 (610)	0.951 (592)	0.951 (570)	0.951 (573)	0.950 (549)	0.975 (717)	0.982 (781)
	Friedman 3	0.945 (0.44)	0.941 (**0.43**)	0.950 (0.46)	0.953 (0.53)	0.952 (0.56)	0.952 (0.55)	0.951 (0.54)	0.952 (0.56)	0.950 (0.49)	0.966 (0.62)	0.971 (0.62)
	Peak	0.955 (**2.84**)	0.978 (5.92)	0.952 (7.11)	0.952 (6.54)	0.954 (10.1)	0.952 (5.68)	0.957 (6.46)	0.951 (6.04)	0.950 (8)	0.988 (14.9)	0.990 (15.2)
	H2c	0.949 (5)	0.953 (5.31)	0.943 (5.39)	0.941 (**4.82**)	0.943 (5.01)	0.944 (5.06)	0.944 (5.03)	0.942 (4.98)	0.942 (5.41)	0.953 (5.33)	0.958 (5.31)
	Tree-N	0.945 (**4.33**)	0.949 (4.37)	0.951 (4.77)	0.957 (5.25)	0.951 (7.03)	0.950 (5.4)	0.953 (4.78)	0.951 (5.89)	0.950 (5.55)	0.979 (10.5)	0.986 (32.1)
	Tree-exp	0.945 (**4.08**)	0.959 (4.39)	0.950 (4.91)	0.955 (5.12)	0.951 (6.87)	0.952 (4.78)	0.954 (4.36)	0.949 (5.49)	0.950 (5.73)	0.978 (10.5)	0.986 (31.8)

2.3.5 Effect of calibration on the performance of prediction intervals

In this section, we investigate the effect of the proposed calibration on performance of the prediction intervals. We apply the same simulation study using the seven simulated data sets. We compare the results of the proposed method without calibration, with OOB calibration and calibration with cross-validation. The desired coverage level is set to 95%. In Figure 2.10, the left plot presents the mean coverages over 28 scenarios for the three variations, and the right plot shows the percentage increase in mean PI length of each of the three calibration variants across 14,000 runs (28 scenarios $\times$ 500 replications). Although we obtain the shortest prediction intervals without calibration, the variability of the mean coverage level is larger and sometimes the coverage falls below 0.94. Looking at the left plot, we can say that the variability of the mean coverage level decreases with both calibration procedures. However, applying OOB calibration provides conservative PIs. The median of the mean coverage level is more than 0.96 and the PIs with the OOB calibration are the widest. Applying calibration with CV produces slightly longer PIs than those with no calibration, but these PIs have coverage levels closer to the desired level.

In the package, both calibration procedures are implemented for the PIBF method. Simulation study results show that, compared to the OOB calibration, calibration with CV produces shorter PIs while maintaining the desired coverage level. Therefore, the default calibration procedure is set to CV in the `pibf()` function. In terms of computational time (see tables 2.4 and 2.5), calibration with k-fold CV is slower than OOB calibration since it needs to fit two additional random forests for each fold.

2.3.6 Performance with real data sets

To further explore the performance of the prediction intervals built with the proposed method, we use 11 real data sets. Since two of the data sets have two response variables, we consider that we are analyzing 13 real data sets. Boston housing and Ames housing data sets are from the R packages `mlbench` and `AmesHousing`, respectively. The other

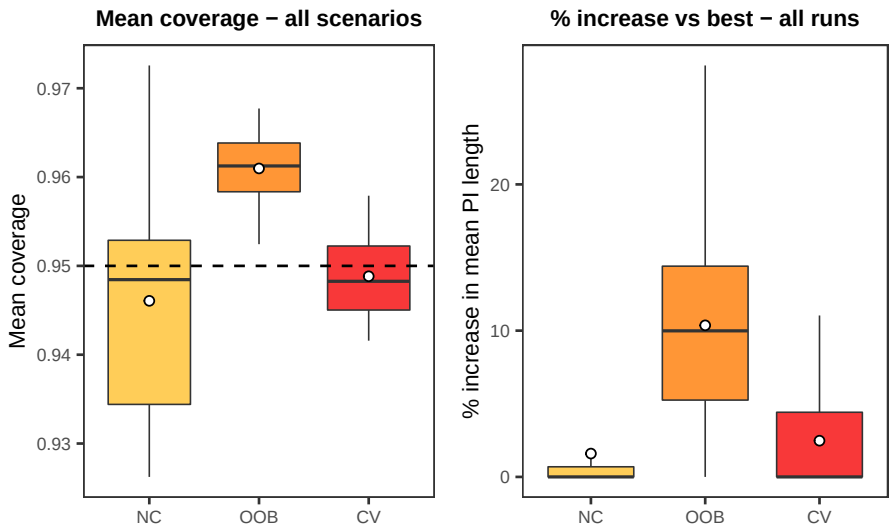

Figure 2.10: Global performance results of the proposed method for the simulated data sets with different calibration procedures. NC: No calibration, OOB: OOB calibration, CV: Calibration with cross-validation. *(Left)* Boxplots for the mean coverage over 500 replications across all scenarios. *(Right)* Boxplots for the percentage increase in mean PI length of each calibration procedure compared to the shortest PI length for a given run across 14,000 runs. Smaller values are better. Since outlier values are distorting the scales, they are removed.

data sets are obtained through the UCI Machine Learning Repository (Dua and Graff, 2017).

For each data set, we apply 100 times 10-fold cross-validation for each method. Hence, for each fold, the training and testing sets correspond to 90% and 10% of the whole data set, respectively. The desired coverage level is set to 95% for all methods. Table 2.3 presents the results of the real data analyses (n is the number of observations and p is the number of predictors) with the average coverage rate of each method over 100 repetitions, and mean prediction interval lengths averaged over 100 repetitions shown in parentheses. Figure 2.11 illustrates the global results of the analyses across datasets. The left plot in Figure 2.11 shows the mean coverages over the 13 real data sets for all methods. Similar to what we have seen with the simulated data sets, the QRF and GRF methods produce conservative prediction intervals, whereas the other methods provide a mean coverage close to the desired level. Again, the $\widehat{PI}_\alpha$ method maintains the target level on average with 0.959, but its variability is the highest among all methods. Across

all data sets, there are three cases where the mean coverage is below 0.94: the proposed method for Concrete slump, and the $\widehat{PI}_\alpha$ method for Auto MPG and Computer hardware.

The right plot in Figure 2.11 presents the relative lengths of methods across 13 real data sets. For each method, there are 13 points in the boxplot, and each point corresponds to the percentage increase in mean PI length compared to the best method for a single real data set. Again, the prediction intervals with GRF and QRF are the widest among eleven methods. Among the other nine methods, the proposed method performs the best, followed by $\widehat{PI}_\alpha$.

For each real data set, we analyze the performance of each method through the mean PI lengths presented in figures 2.12 to 2.14. For the Abalone data set, the HDR method produces the shortest prediction intervals, followed by the SPI and Quant methods. While the QRF method over-covers, its PIs are no wider than those of most of the other methods. The proposed method, PIBF, is distinctly the best prediction interval method yielding the shortest PI lengths for the Air quality data set with absolute and relative humidity response variables, Airfoil selfnoise, Ames housing, Boston housing, Concrete compression, Energy efficiency data set with cooling and heating load response variables, and Servo data sets. In the Auto MPG data set, $\widehat{PI}_\alpha$ has the shortest mean PI length but with a mean coverage of 0.929. Among the other methods, PIBF, OOB, LM, Quant and SPI methods show similarly good performances while maintaining the desired coverage level. For the Computer hardware data set, PIBF, $\widehat{PI}_\alpha$, and SPI methods perform better than the other methods. They have similar mean PI lengths. For the Concrete slump data set, the proposed method has the shortest mean PI length, but with a slightly smaller coverage of 0.939. This data set is the only one of the simulated and real data sets where the proposed method has a mean coverage below 0.94. After PIBF, $\widehat{PI}_\alpha$ and LM show a good performance with a mean coverage close to the target level. Overall, we can conclude that the proposed method shows better performance than the 10 competing methods for almost all of the real data sets.

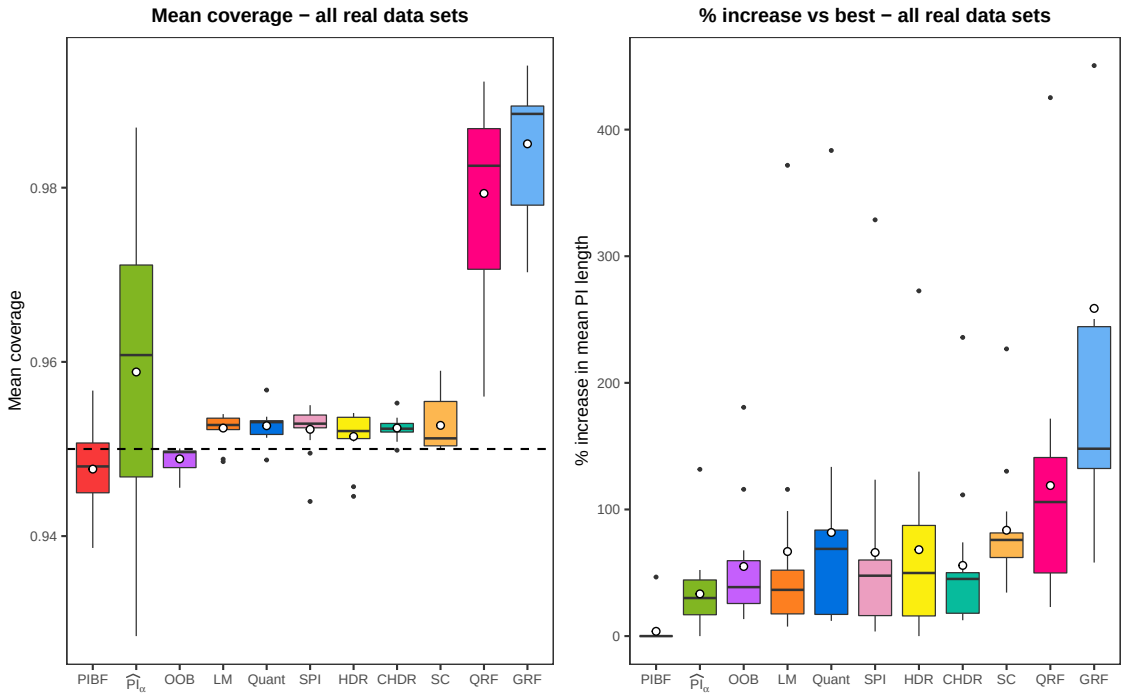

Figure 2.11: *(Left)* Boxplots for the mean coverage over all real data sets. All methods except the QRF and GRF methods are able to provide a mean coverage close to the desired coverage level of 0.95. Each white circle is the average of the mean coverages over 13 real data sets. *(Right)* Boxplots for the percentage increase in mean PI length of each method compared to the shortest PI length for a given real data set across 13 data sets. The smallest the percentage increase, the better the method. Each white circle is the average of the relative lengths over 13 real data sets. One of the outlier values for GRF with the percentage increase of 1244% is removed from the graph since it is distorting the scales. PIBF: Prediction intervals with boosted forests (the proposed method), $\widehat{PI}_\alpha$: Conditional α-level prediction interval, OOB: Out-of-Bag approach, LM: Classical method, Quant: Quantiles, SPI: Shortest PI, HDR: Highest density region, CHDR: Contiguous HDR, SC: Split conformal, QRF: Quantile regression forest, GRF: Generalized random forest.

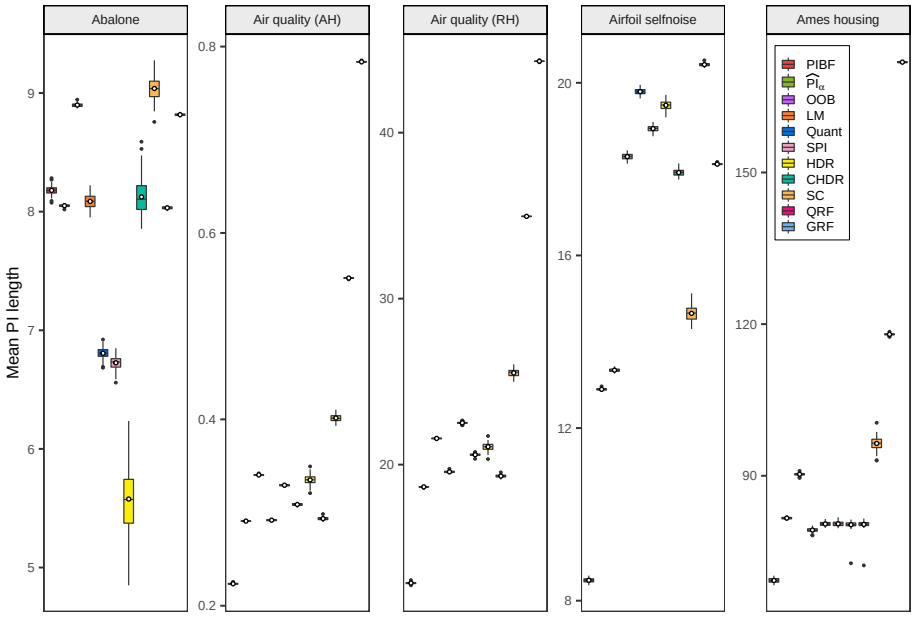

Figure 2.12: Distributions of the mean PI length across 100 repetitions for Abalone, Air quality with absolute and relative humidity, Airfoil selfnoise, and Ames housing data sets.

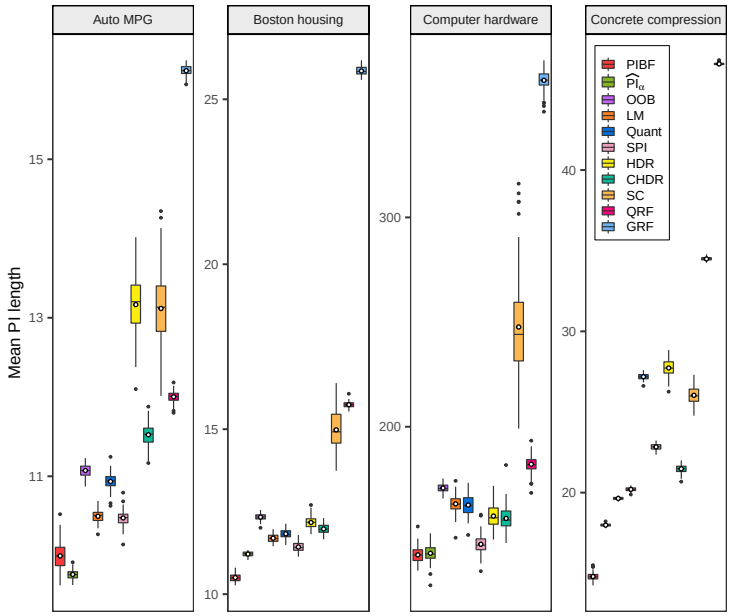

Figure 2.13: Distributions of the mean PI length across 100 repetitions for Auto MPG, Boston housing, Computer hardware, and Concrete compression data sets.

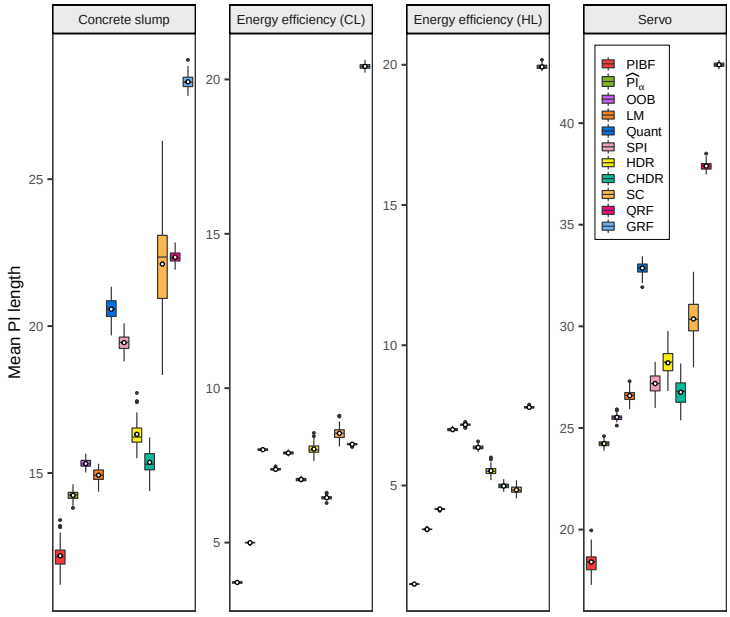

Figure 2.14: Distributions of the mean PI length across 100 repetitions for Concrete slump, Energy efficiency with cooling and heating load, and Servo data sets.

Table 2.3: Results of the real data analysis. Average coverage rates of each method for each real data set, with average mean PI lengths shown in parentheses. The desired coverage level is 0.95. Shortest average mean PI lengths are shown in bold. PIBF: Prediction intervals with boosted forests (the proposed method), $\widehat{PI}_\alpha$: Conditional α-level prediction interval, OOB: Out-of-Bag approach, LM: Classical method, Quant: Quantiles, SPI: Shortest PI, HDR: Highest density region, CHDR: Contiguous HDR, SC: Split conformal, QRF: Quantile regression forest, GRF: Generalized random forest.

Data	n	p	PIBF	$\widehat{PI}_\alpha$	OOB	LM	Quant	SPI	HDR	CHDR	SC	QRF	GRF
Abalone	4177	8	0.947 (8.18)	0.946 (8.05)	0.950 (8.9)	0.949 (8.09)	0.953 (6.81)	0.951 (6.73)	0.946 (**5.58**)	0.951 (8.12)	0.950 (9.04)	0.971 (8.03)	0.978 (8.82)
Air quality (AH)	6941	13	0.948 (**0.22**)	0.966 (0.29)	0.950 (0.34)	0.954 (0.29)	0.954 (0.33)	0.954 (0.31)	0.954 (0.34)	0.954 (0.29)	0.950 (0.4)	0.992 (0.55)	0.994 (0.78)
Air quality (RH)			0.949 (**12.9**)	0.971 (18.7)	0.950 (21.6)	0.953 (19.6)	0.953 (22.5)	0.954 (20.6)	0.954 (21.1)	0.953 (19.3)	0.950 (25.5)	0.989 (35)	0.989 (44.3)
Airfoil selfnoise	1503	5	0.946 (**8.47**)	0.972 (12.9)	0.950 (13.3)	0.952 (18.3)	0.952 (19.8)	0.953 (18.9)	0.952 (19.5)	0.953 (17.9)	0.951 (14.7)	0.987 (20.4)	0.988 (18.1)
Ames housing	2929	80	0.955 (**69.3**)	0.961 (81.6)	0.950 (90.3)	0.954 (79.3)	0.953 (80.5)	0.953 (80.5)	0.952 (80.3)	0.952 (80.4)	0.951 (96.4)	0.987 (118)	0.993 (172)
Auto MPG	392	7	0.945 (10)	0.929 (**9.76**)	0.947 (11.1)	0.953 (10.5)	0.953 (10.9)	0.953 (10.5)	0.954 (13.2)	0.952 (11.5)	0.955 (13.1)	0.967 (12)	0.977 (16.1)
Boston housing	506	13	0.942 (**10.5**)	0.948 (11.2)	0.949 (12.3)	0.954 (11.7)	0.953 (11.8)	0.953 (11.4)	0.952 (12.2)	0.952 (12)	0.951 (15)	0.982 (15.7)	0.990 (25.9)
Computer hardware	209	6	0.942 (**139**)	0.939 (140)	0.946 (171)	0.952 (163)	0.951 (163)	0.950 (144)	0.951 (157)	0.950 (156)	0.957 (248)	0.965 (182)	0.985 (365)
Concrete compression	1030	8	0.948 (**14.8**)	0.957 (18)	0.950 (19.6)	0.954 (20.2)	0.953 (27.2)	0.954 (22.8)	0.953 (27.7)	0.953 (21.5)	0.950 (26)	0.982 (34.5)	0.989 (46.6)
Concrete slump	103	7	0.939 (**12.2**)	0.947 (14.2)	0.948 (15.3)	0.949 (14.9)	0.949 (20.6)	0.944 (19.4)	0.945 (16.3)	0.951 (15.4)	0.958 (22.1)	0.956 (22.3)	0.970 (28.3)
Energy efficiency (CL)	768	8	0.953 (**3.71**)	0.977 (5)	0.950 (8.01)	0.952 (7.37)	0.952 (7.9)	0.952 (7.04)	0.951 (8.03)	0.953 (6.46)	0.951 (8.54)	0.985 (8.18)	0.986 (20.4)
Energy efficiency (HL)			0.951 (**1.48**)	0.987 (3.43)	0.950 (4.16)	0.953 (6.99)	0.952 (7.17)	0.954 (6.35)	0.951 (5.52)	0.952 (4.98)	0.952 (4.84)	0.988 (7.79)	0.989 (19.9)
Servo	167	4	0.957 (**18.4**)	0.966 (24.2)	0.948 (25.5)	0.953 (26.6)	0.957 (32.9)	0.955 (27.2)	0.954 (28.2)	0.955 (26.8)	0.959 (30.4)	0.983 (37.9)	0.977 (42.9)

2.3.7 Comparison of the computational times

All simulations and real data analyses were conducted in R version 3.6.0 on a Linux machine with Intel(R) Xeon(R) E5-2667 v3 @ 3.20GHz with 396 GB of memory. The average computational time of each method for the simulated and real data sets are presented in tables 2.4 and 2.5. For the proposed method (PIBF), the computational times for both calibration methods, cross-validation and OOB, are presented in the tables. We can see from the tables that, for most of the data sets, calibration with cross-validation has longer running times than OOB calibration, which is expected since with the k-fold cross-validation, we fit $2k$ more random forests than applying OOB calibration.

For the variations of Roy and Larocque (2020), since we can build prediction intervals with the five PI methods by only fitting a single random forest with a selected splitting rule, we present the total computational time for building the five variations under RFPI. To be clear, for a given splitting rule, the `rfpi()` function fits a random forest and then the set of PI methods requested by the user are applied to the output of the random forest. In our simulations, we choose to return all five PI methods for the selected splitting rule, *i.e.* when we measure the running time of the `rfpi()` function, we get the total running time of building five prediction intervals. Therefore, we should interpret the values for RFPI with care while comparing the computational times of the methods. Although not all PI methods have similar computational complexities, we can say that even the average time of building prediction intervals with one of these variations, assuming they have similar running times, is reasonable. Since, for the HDR-based PI methods, an optimal bandwidth has to be chosen, which is a time-consuming process, among the five PI methods, the slowest ones are the HDR and CHDR. From the remaining variations, the classical method, LM, is the fastest, followed by the Quant and SPI methods.

For both the simulations and real data analyses, the OOB and GRF methods have the smallest running times. For most of the methods, the increase in the sample size has a mild effect on the ratio of increase in running times. However, for the split conformal method with the simulated data sets, running times increase more than the proportional

Table 2.4: Average computational time (in seconds) of each method over 500 replications for each simulated data set. The values represent the average time to build prediction intervals for a test set with 1000 observations. PIBF-CV: The proposed method with the cross-validation calibration, PIBF-OOB: The proposed method with the OOB calibration, RFPI: The average total running time of the five PI methods, *i.e.* LM + Quant + SPI + HDR + CHDR.

n_{train}	Data	PIBF-CV	PIBF-OOB	$\widehat{PI}_\alpha$	OOB	RFPI	SC	QRF	GRF
200	Friedman 1	21.36	13.44	9.62	0.25	87.79	0.61	3.05	0.27
	Friedman 2	21.83	13.96	9.62	0.23	59.50	0.44	2.61	0.26
	Friedman 3	28.84	16.03	11.46	0.20	75.47	0.41	2.50	0.22
	Peak	18.76	10.95	11.91	0.28	73.15	0.89	4.05	0.43
	H2c	12.92	9.83	7.99	0.27	36.94	0.79	4.05	0.30
	Tree-N	22.75	21.17	12.21	0.23	100.02	0.50	2.81	0.26
	Tree-exp	16.28	15.62	15.65	0.23	87.49	0.53	2.85	0.25
500	Friedman 1	38.37	14.63	8.50	0.44	96.18	2.45	9.29	0.66
	Friedman 2	27.07	13.64	7.64	0.43	84.19	1.80	6.79	0.47
	Friedman 3	18.85	17.49	7.98	0.47	70.78	1.86	4.96	0.45
	Peak	29.16	18.14	9.72	0.70	212.23	2.29	7.49	1.26
	H2c	14.84	11.29	11.20	0.44	58.60	1.74	4.95	0.78
	Tree-N	17.80	20.31	7.70	0.51	183.92	2.21	4.28	0.51
	Tree-exp	18.05	19.05	7.74	0.51	170.24	1.72	4.45	0.49
1000	Friedman 1	32.92	18.07	12.11	0.64	181.30	3.35	9.54	1.50
	Friedman 2	23.07	15.94	10.25	0.45	135.64	2.14	6.26	0.90
	Friedman 3	23.24	16.07	7.37	0.44	96.76	2.27	6.40	0.69
	Peak	51.93	33.12	7.95	0.83	374.89	5.35	19.71	1.65
	H2c	26.81	12.56	8.07	0.59	80.38	3.68	13.11	0.87
	Tree-N	22.84	16.70	7.11	0.47	288.35	2.72	9.53	0.78
	Tree-exp	21.06	17.26	7.03	0.48	272.52	2.62	10.38	0.69
5000	Friedman 1	155.81	139.88	28.70	3.72	928.84	40.79	134.97	4.84
	Friedman 2	155.62	101.58	35.25	2.30	430.80	33.43	60.99	2.65
	Friedman 3	106.70	91.72	34.41	2.36	330.81	28.47	73.18	2.72
	Peak	287.67	245.17	22.76	6.84	1914.56	57.96	123.19	11.80
	H2c	283.53	107.57	24.12	4.31	271.18	42.69	79.19	5.01
	Tree-N	106.17	71.78	22.97	3.19	753.32	26.88	74.81	3.86
	Tree-exp	95.85	68.90	21.76	3.32	779.42	26.09	64.02	3.85

increase in sample sizes. Similarly, we can see that the QRF method is also affected from the training sample size as it rises to 5000.

2.4 Conclusion

In this paper, we have introduced an R package named `RFpredInterval`. This package implements 16 methods to build prediction intervals with random forests: a new method to build **P**rediction **I**ntervals with **B**oosted **F**orests (PIBF) and 15 different variations to produce prediction intervals with random forests proposed by Roy and Larocque (2020).

Table 2.5: Average computational time (in seconds) of each method over 100 times 10-fold cross-validation for each real data set. PIBF-CV: The proposed method with the cross-validation calibration, PIBF-OOB: The proposed method with the OOB calibration, RFPI: The average total running time of the five PI methods, *i.e.* LM + Quant + SPI + HDR + CHDR.

Data	n	p	PIBF-CV	PIBF-OOB	$\widehat{PI}_\alpha$	OOB	RFPI	SC	QRF	GRF
Abalone	4177	8	102.37	64.39	17.65	5.20	422.10	20.42	45.39	6.32
Air quality (AH)	6941	13	383.60	126.95	26.63	11.55	1256.56	45.45	126.36	16.07
Air quality (RH)			383.53	124.71	26.60	11.55	1111.16	46.13	127.02	15.95
Airfoil selfnoise	1503	5	256.92	16.61	3.57	0.31	271.35	1.80	3.65	0.65
Ames housing	2929	80	195.07	28.56	15.64	8.73	333.28	42.95	226.49	11.34
Auto MPG	392	7	11.02	7.08	1.82	0.35	47.84	0.58	1.73	0.40
Boston housing	506	13	14.77	8.89	2.32	0.49	63.20	1.31	3.63	0.70
Computer hardware	209	6	6.53	3.80	0.93	0.22	26.49	0.28	0.98	0.24
Concrete compression	1030	8	28.28	17.80	3.38	0.38	257.19	2.03	5.86	0.66
Concrete slump	103	7	9.52	2.10	0.65	0.14	26.60	0.13	0.62	0.11
Energy efficiency (CL)	768	8	85.97	15.93	2.51	0.34	341.73	0.94	2.39	0.52
Energy efficiency (HL)			99.95	20.31	2.47	0.34	380.97	0.92	2.42	0.42
Servo	167	4	12.05	3.99	0.72	0.15	53.14	0.12	0.65	0.15

PIBF provides bias-corrected point predictions obtained with the one-step boosted forest and prediction intervals by using the nearest neighbour out-of-bag observations to estimate the conditional prediction error distribution.

We performed an extensive simulation study with a variety of simulated data sets and applied real data analyses to investigate the performance of the proposed method. The performance was evaluated based on the coverage level and length of the prediction intervals. We compared the performance of the proposed method to 10 existing methods for building prediction intervals with random forests. The proposed method was able to maintain the desired coverage level with both the simulated and real data sets. In terms of the PI lengths, globally, the proposed method provided the shortest prediction intervals among all methods. The conclusions drawn from the analysis of real data sets were very similar to those with the simulated data sets. This provides evidence for the reliability of the proposed method. All results obtained indicate that the proposed method can be used with confidence for a variety of regression problems.

Note that the coverage rate of prediction intervals for new observations can have several interpretations. An interesting discussion about this issue is given in Mayr et al. (2012). In that paper, the authors presented two interpretation of coverage: sample cover-

age and conditional coverage. Sample coverage means that if we draw a new sample from the same population as the training sample and build PIs with a desired coverage level of $(1-\alpha)$, then the global coverage rate over this sample will be $(1-\alpha)$. The conditional coverage means that if we sample many new observations always having the same set of covariates and build PIs for them with a desired coverage level of $(1-\alpha)$, then about $(1-\alpha)\,100\%$ of these prediction intervals will contain the true value of the response. To hold a desired level of conditional coverage, the predictive method needs to provide the desired coverage level over the entire covariate space. On the other hand, sample coverage needs only maintain the desired coverage level over the new sample, on average. Therefore, if the conditional coverage holds, then the sample coverage also holds. In practice, predictive models are mostly evaluated with their global predictive performance. Hence, ensuring that the sample coverage level is achieved should be sufficient for most applications. The proposed calibration method with cross-validation is designed to ensure the sample coverage property. From the simulation study and real data analyses, we can see that the sample coverage is attained with the proposed calibration method.

References

Alakus, C., Larocque, D., and Labbe, A. (2022). *RFpredInterval: Prediction Intervals with Random Forests and Boosted Forests*. R package version 1.0.5.

Alakuş, C., Larocque, D., Jacquemont, S., Barlaam, F., Martin, C.-O., Agbogba, K., Lippé, S., and Labbe, A. (2021). Conditional canonical correlation estimation based on covariates with random forests. *Bioinformatics*, 37(17):2714–2721.

Athey, S., Tibshirani, J., and Wager, S. (2019). Generalized random forests. *The Annals of Statistics*, 47(2):1148–1178.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Breiman, L. and Breiman, L. (1984). *Classification and regression trees*. The Wadsworth & Brooks/Cole statistics/probability series. Wadsworth International Group, Belmont, Calif.

De Cock, D. (2011). Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. *Journal of Statistics Education*, 19(3).

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Ghosal, I. and Hooker, G. (2021). Boosting Random Forests to Reduce Bias; One-Step Boosted Forest and Its Variance Estimate. *Journal of Computational and Graphical Statistics*, 30(2):493–502.

Hothorn, T., Lausen, B., Benner, A., and Radespiel-Tröger, M. (2004). Bagging survival trees. *Statistics in Medicine*, 23(1):77–91.

Hothorn, T. and Zeileis, A. (2021). Predictive Distribution Modeling Using Transformation Forests. *Journal of Computational and Graphical Statistics*, pages 1–16.

Ishwaran, H. and Kogalur, U. B. (2021). *randomForestSRC: Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 2.11.0.

Jerome H. Friedman (1991). Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19(1):1–67.

Kuhn, M. (2020). *AmesHousing: The Ames Iowa Housing Data*. R package version 0.0.4.

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., and Wasserman, L. (2018). Distribution-Free Predictive Inference for Regression. *Journal of the American Statistical Association*, 113(523):1094–1111.

Leisch, F. and Dimitriadou, E. (2021). *mlbench: Machine Learning Benchmark Problems*. R package version 2.1-3.

Lin, Y. and Jeon, Y. (2006). Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association*, 101(474):578–590.

Lu, B. and Hardin, J. (2020). *forestError: A Unified Framework for Random Forest Prediction Error Estimation*. R package version 0.2.0.

Lu, B. and Hardin, J. (2021). A Unified Framework for Random Forest Prediction Error Estimation. *Journal of Machine Learning Research*, 22(8):1–41.

Mayr, A., Hothorn, T., and Fenske, N. (2012). Prediction intervals for future BMI values of individual children - a non-parametric approach by quantile boosting. *BMC Medical Research Methodology*, 12(1):6.

Meinshausen, N. (2006). Quantile Regression Forests. *Journal of Machine Learning Research*, 7(35):983–999.

Meinshausen, N. (2017). *quantregForest: Quantile Regression Forests*. R package version 1.3-7.

Mentch, L. and Hooker, G. (2016). Quantifying Uncertainty in Random Forests via Confidence Intervals and Hypothesis Tests. *Journal of Machine Learning Research*, 17(26):1–41.

Moradian, H., Larocque, D., and Bellavance, F. (2017). L1 splitting rules in survival forests. *Lifetime Data Analysis*, 23(4):671–691.

Moradian, H., Larocque, D., and Bellavance, F. (2019). Survival forests for data with dependent censoring. *Statistical Methods in Medical Research*, 28(2):445–461.

Roy, M.-H. and Larocque, D. (2020). Prediction intervals with random forests. *Statistical Methods in Medical Research*, 29(1):205–229.

Tabib, S. and Larocque, D. (2020). Non-parametric individual treatment effect estimation for survival data with random forests. *Bioinformatics*, 36(2):629–636.

Tibshirani, J., Athey, S., Sverdrup, E., and Wager, S. (2021). *grf: Generalized Random Forests*. R package version 2.0.2.

Tibshirani, R. (2019). *conformalInference: Tools for conformal inference in regression*. R package version 1.1.

Vovk, V., Gammerman, A., and Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer Science & Business Media.

Vovk, V., Nouretdinov, I., and Gammerman, A. (2009). On-line predictive linear regression. *The Annals of Statistics*, 37(3):1566–1590.

Wager, S. and Athey, S. (2018). Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

Wright, M. N., Wager, S., and Probst, P. (2020). *ranger: A Fast Implementation of Random Forests*. R package version 0.12.1.

Zhang, G. and Lu, Y. (2012). Bias-corrected random forests in regression. *Journal of Applied Statistics*, 39(1):151–160.

Zhang, H. (2019). *rfinterval: Predictive Inference for Random Forests*. R package version 1.0.0.

Zhang, H., Zimmerman, J., Nettleton, D., and Nordman, D. J. (2020). Random Forest Prediction Intervals. *The American Statistician*, 74(4):392–406.

Chapter 3

Covariance regression with random forests

Abstract

Capturing the conditional covariances or correlations among the elements of a multivariate response vector based on covariates is important to various fields including neuroscience, epidemiology and biomedicine. We propose a new method called Covariance Regression with Random Forests (CovRegRF) to estimate the covariance matrix of a multivariate response given a set of covariates, using a random forest framework. Random forest trees are built with a splitting rule specially designed to maximize the difference between the sample covariance matrix estimates of the child nodes. We also propose a significance test for the partial effect of a subset of covariates. We evaluate the performance of the proposed method and significance test through a simulation study which shows that the proposed method provides accurate covariance matrix estimates and that the Type-1 error is well controlled. We also demonstrate an application of the proposed method with a thyroid disease data set.

3.1 Introduction

Most existing multivariate regression analyses focus on estimating the conditional mean of the response variable given its covariates. For example, in traditional regression analysis, the expectation of the response variables is related to linear combinations of covariates. While estimating the conditional covariances or correlations among multiple responses based on covariates is also important, it is a less studied problem. For example, functional brain connectivity focuses on the exploration of the co-occurrence of brain activity in different brain regions, and this co-variability can be explained as a function of covariates (Seiler and Holmes, 2017). As another example, human biomarkers such as glucose, cholesterol, iron, albumin, and so on, are important for biomedical research and the covariance of these biomarkers is influenced by age (Le Goallec and Patel, 2019). In microbiome studies, the changes in the co-occurrence patterns among taxa with respect to the covariates have been studied (Levy and Borenstein, 2013; McGregor et al., 2020).

In general terms, let $\mathbf{Y}_{n\times q}$ be a matrix of q response variables measured on n observations, where $\mathbf{y}_i$ represents the ith row of $\mathbf{Y}$. Similarly, let $\mathbf{X}_{n\times p}$ be a matrix of p covariates available for all n observations, where $\mathbf{x}_i$ represents the ith row of $\mathbf{X}$. For an observation with covariates $\mathbf{x}_i$ and responses $\mathbf{y}_i$, the goal is to estimate the covariance of the response variables based on the covariates $\Sigma_{\mathbf{x}_i}$ and to analyze how this conditional covariance matrix varies with respect to the covariates. For this problem, Yin et al. (2010) use a kernel estimator to estimate the conditional covariance matrix for a single continuous covariate. However, it is not clear how to extend this approach to situations with multiple covariates. Hoff and Niu (2012) propose a linear covariance regression model

$$\mathbf{y}_i = (\mathbf{A} + \gamma_i \mathbf{B})\mathbf{x}_i + \varepsilon_i,$$

where the mean and covariance of the multivariate response is parameterized as functions of covariates. This model can also be interpreted as a special random-effects model where $\mathbf{A}_{q\times(p+1)}$ and $\mathbf{B}_{q\times(p+1)}$ characterize the fixed and random parts of the model, respectively. The scalar γ_i can be interpreted as an individual-level variability in addition to the random error ε_i. The rows of $\mathbf{B}$ indicate how much this additional variability affects $\mathbf{y}_i$. The vector

ε_i is of dimension $q \times 1$ and is assumed to be normally distributed. In this framework, they assume that $E[\gamma_i] = 0$, $E[\varepsilon_i] = 0$, $E[\gamma_i \varepsilon_i] = 0$, $Var[\gamma_i] = 1$, $Var[\varepsilon_i] = \Psi$, leading to the following covariance matrix

$$\Sigma_{\mathbf{x}_i} = \Psi + \mathbf{B}\mathbf{x}_i\mathbf{x}_i^T\mathbf{B}^T.$$

Niu and Hoff (2019) illustrate an application of this model with a four-dimensional health outcome. Fox and Dunson (2015) propose a Bayesian nonparametric model for covariance regression within a high-dimensional response context. Their approach relates the high-dimensional multivariate response set to a lower-dimensional subspace through covariate-dependent factor loadings obtained with a latent factor model. The conditional covariance matrix is a quadratic function of these factor loadings. The method is limited to data sets with smaller sample sizes. Franks (2021) proposes a parametric Bayesian model for high-dimensional responses. In this model, the conditional covariance matrices vary with continuous covariates. Zou et al. (2017) propose another covariance regression model where the covariance matrix is linked to the linear combination of similarity matrices of covariates.

In this study, we propose a nonparametric covariance regression method for estimating the covariance matrix of a multivariate response given a set of covariates, using a random forest framework. The above-mentioned methods are very useful in modelling covariance matrix but compared to them the proposed method offers higher flexibility in estimating the covariance matrix given the set of covariates. For example, with the proposed method, we can estimate the conditional covariance matrix for a set of covariates including multiple continuous and categorical variables, and the proposed method can be used to capture complex interaction patterns with the set of covariates. Moreover, the proposed method is nonparametric and needs less computational time compared to the parametric models, and can be applied to data sets with larger sample sizes.

Random forest (Breiman, 2001) is an ensemble tree-based algorithm involving many decision trees, and can also be seen as an adaptive nearest neighbour predictor (Hothorn et al., 2004; Lin and Jeon, 2006; Moradian et al., 2017, 2019; Roy and Larocque, 2020;

Tabib and Larocque, 2020; Alakuş et al., 2021). In the proposed random forest framework, we grow each tree with a splitting rule specially designed to maximize the difference in the sample covariance of $\mathbf{Y}$ between child nodes. For a new observation $\mathbf{y}^*$ with covariates $\mathbf{x}^*$, the proposed random forest finds the set of nearest neighbour observations among the out-of-bag (OOB) observations that are not used in the tree growing process. This set of nearest neighbour observations is then used to estimate the conditional covariance matrix of $\mathbf{y}^*$ given $\mathbf{x}^*$. In each tree built in the proposed random forest framework, the set of covariates is used to find subgroups of observations with similar conditional covariance matrices, assuming that they are related to conditional covariance matrices. We propose a hypothesis test to evaluate the effect of a subset of covariates on the estimated covariance matrices while controlling for the others. We investigate two particular cases, the global effect of the covariates and the partial effect of a single covariate.

This paper is organized as follows. In Section 3.2, we give the details of the proposed method, significance test and variable importance measure. The simulation study results for accuracy evaluation, global and partial effects of covariates, and variable importance are presented in Section 3.3. We provide a real data example in Section 3.4, and conclude with some remarks in Section 3.5.

3.2 Proposed method

Let $\Sigma_{\mathbf{x}_i}$ be the true conditional covariance matrix of $\mathbf{y}_i$ based on covariates $\mathbf{x}_i$, and $\Sigma_{\mathbf{X}}$ be the collection of all conditional covariance matrices for n observations. Similarly, let $\hat{\Sigma}_{\mathbf{x}_i}$ be the estimated conditional covariance matrix of $\mathbf{y}_i$ based on covariates $\mathbf{x}_i$, and $\hat{\Sigma}_{\mathbf{X}}$ be the collection of all estimated conditional covariance matrices for n observations. In this section, we describe the proposed method in detail.

3.2.1 Tree growing process and estimation of covariance matrices for new observations with random forests

We aim to train a random forest with the set of covariates $\mathbf{X}$ to find subgroups of observations with similar covariance matrices of $\mathbf{Y}$, based on many unsupervised decision trees built with a specialized splitting criterion. The tree growing process follows the CART approach (Breiman et al., 1984). The basic idea of the CART algorithm is to select the best split at each parent node among all possible splits, all evaluated with a selected splitting criterion, to obtain the purest child nodes. The algorithm evaluates all possible splits to determine the split variable and split point. Instead of considering all possible splits at each parent node, the best split search in random forests is confined to a randomly chosen subset of covariates that varies from node to node. The splitting process continues until all nodes are terminal.

Our goal is to obtain subgroups of observations with distinct covariance matrices. Hence, we propose a customized splitting rule that will seek to increase the difference in covariance matrices between two child nodes in the tree (Athey et al., 2019; Moradian et al., 2017; Tabib and Larocque, 2020; Alakuş et al., 2021). We define Σ^L as the sample covariance matrix estimate of the left node as follows:

$$\Sigma^L = \frac{1}{n_L - 1} \sum_{i \in t_L} (\mathbf{y}_i - \bar{\mathbf{Y}}_L)(\mathbf{y}_i - \bar{\mathbf{Y}}_L)^T,$$

where t_L is the set of indices of the observations in the left node, n_L is the left node size and $\bar{\mathbf{Y}}_L = \frac{1}{n_L} \sum_{i \in t_L} \mathbf{y}_i$. The sample covariance matrix estimate of the right node, Σ^R, is computed in the same way, where n_R is the right node size. The proposed splitting criterion is

$$\sqrt{n_L n_R} * d(\Sigma^L, \Sigma^R), \tag{3.1}$$

where $d(\Sigma^L, \Sigma^R)$ is the Euclidean distance between the upper triangular part of the two matrices and computed as follows:

$$d(A,B) = \sqrt{\sum_{i=1}^{q} \sum_{j=i}^{q} (\mathbf{A}_{ij} - \mathbf{B}_{ij})^2}, \tag{3.2}$$

where $\mathbf{A}_{q\times q}$ and $\mathbf{B}_{q\times q}$ are symmetric matrices. The best split among all possible splits is the one that maximizes (3.1).

The final covariance matrices are estimated based on the random forest. For a new observation, we use the nearest neighbour observations to estimate the final covariance matrix. The idea of finding the nearest neighbour observations, a concept very similar to the 'nearest neighbour forest weights' (Hothorn et al., 2004; Lin and Jeon, 2006), was introduced in Moradian et al. (2017) and later used in Moradian et al. (2019); Roy and Larocque (2020); Tabib and Larocque (2020); Alakuş et al. (2021). Roy and Larocque (2020) called this set of observations the Bag of Observations for Prediction (BOP).

For a new observation $\mathbf{x}^*$, we form the set of nearest neighbour observations with the out-of-bag (OOB) observations (Lu and Hardin, 2021; Alakuş et al., 2022). We can define the BOP_{oob} for a new observation as

$$BOP_{oob}(\mathbf{x}^*) = \bigcup_{b=1}^{B} O_b(\mathbf{x}^*),$$

where B is the number of trees and $O_b(\mathbf{x}^*)$ is the set of OOB observations in the same terminal node as $\mathbf{x}^*$ in the bth tree. Each tree is built with a selected random sub-sample, i.e. in-bag observations (I_b), which has about 63 percent distinct observations from the original sample. The remaining training observations, namely O_b, are OOB observations for that tree and are not used to build the bth tree.

BOP_{oob} is slightly different than the nearest neighbour sets in the previous papers who use in-bag observations to form BOP. Since the OOB observations are not used in the tree building process, for the trees where they are OOB, they act as new observations. Therefore, OOB observations represent a new observation better than in-bag observations. Using OOB observations for neighbourhood construction is similar to the idea of honesty in the context of forests. An honest double-sample tree splits the training subsample into two parts: one part for tree growing and another part for estimating the desired response (Wager and Athey, 2018). We use the nearest neighbour construction idea to estimate the covariance matrices for the new observations. Algorithm 3.1 describes how to estimate the covariance matrix with OOB observations for a new or training observation. After

training the random forest with the specialized splitting criterion, for a new observation $\mathbf{x}^*$, we form $BOP_{oob}(\mathbf{x}^*)$ and then we estimate the covariance matrix by computing the sample covariance matrix of the observations in $BOP_{oob}(\mathbf{x}^*)$.

3.2.2 `nodesize` tuning

The number of observations in the nodes decreases as we progress down the tree during the tree-building process. The `nodesize` parameter is the target average size for the terminal nodes. Lowering this parameter results in deeper trees, which means more splits until the terminal nodes. Tuning the `nodesize` parameter can potentially improve the prediction performance (Lin and Jeon, 2006).

In typical supervised problems where the target is the observed true response, random forests search for the optimal level of the `nodesize` parameter by using out-of-bag (OOB) prediction errors computed using the true responses and OOB predictions. The `nodesize` value with the smallest OOB error is chosen. However, in our problem, the target is the conditional covariance matrix which is unknown. Therefore, we propose a heuristic method for tuning the `nodesize` parameter. For `nodesize` tuning, we use the OOB covariance matrix estimates, as described in Algorithm 3.1.

The general idea of the `nodesize` tuning method is to find the `nodesize` level where the OOB covariance matrix predictions at two consecutive `nodesize` levels become similar. We first train separate random forests for a set of `nodesize` values (see the Parameter settings section in simulation study). Then, we compute the OOB covariance matrix estimates as described in Algorithm 3.1 for each random forest. Define $MAD(\mathbf{A},\mathbf{B}) = \frac{2}{q(q+1)}\sum_{i=1}^{q}\sum_{j=i}^{q}|\mathbf{A}_{ij}-\mathbf{B}_{ij}|$. Let $\hat{\Sigma}^{s}_{\mathbf{x}_i}$ be the estimated covariance matrix for observation i when `nodesize`$= s$. Let $s(1) < \ldots < s(M)$ be a set of increasing node sizes. For $j = \{1,\ldots M-1\}$, let

$$MAD_j = \frac{1}{n}\sum_{i=1}^{n} MAD\left(\hat{\Sigma}^{s(j)}_{\mathbf{x}_i}, \hat{\Sigma}^{s(j+1)}_{\mathbf{x}_i}\right).$$

Then we select $s(j)$ that corresponds to the value j for which MAD_j is the minimum

among $\{MAD_1, \ldots, MAD_M\}$. See Appendix H for the results of a `nodesize` tuning experiment.

When a node sample size n_d is smaller than the number of responses q, the sample covariance matrix becomes highly variable. In fact, if $n_d - 1 < q$, the estimate is singular and hence non-invertible. Therefore, the tuning set of `nodesize` levels should be larger than q. In fact, we need more than q distinct values, so we use sub-sampling instead of bootstrap resampling for tree building to guarantee distinctness, assuming the observations in the original sample are distinct.

Algorithm 3.1 Estimation of covariance matrix for a new or training observation

Input: A forest built with the proposed method

1: $BOP_{oob}(\mathbf{x}_i) = \emptyset$
2: **for** b=1,...,B **do**
3: **if** $\mathbf{x}_i$ is a new observation **or** ($\mathbf{x}_i$ is a training observation **and** $\mathbf{x}_i \in O_b$) **then**
4: Find the terminal node of $\mathbf{x}_i$ at tree b, say d
5: $BOP_{oob}(\mathbf{x}_i) = BOP_{oob}(\mathbf{x}_i) \cup O_b^d(\mathbf{x}_i)$ (where $O_b^d(\mathbf{x}_i)$ is the set of OOB observations in the same terminal node d as $\mathbf{x}_i$, excluding $\mathbf{x}_i$ itself when $\mathbf{x}_i$ is a training observation)
6: **end if**
7: **end for**
8: Compute sample covariance matrix with the observations in $BOP_{oob}(\mathbf{x}_i)$

3.2.3 Significance test

The proposed method uses covariates to find groups of observations with similar covariance matrices with the assumption that the set of covariates is important to distinguish between these covariance matrices. However, some (or all) covariates might not be relevant. In this paper, we propose a hypothesis test to evaluate the effect of a subset of covariates on the covariance matrix estimates, while controlling for the other covariates.

If a subset of covariates has an effect on the covariance matrix estimates obtained with the proposed method, then the conditional covariance matrix estimates given all covariates should be significantly different from the conditional covariance matrix estimates given the controlling set of covariates. We propose a hypothesis test to evaluate the effect of a

subset of covariates on the covariance matrix estimates for the null hypothesis

$$H_0 : \Sigma_{\mathbf{X}} = \Sigma_{\mathbf{X}^c}, \tag{3.3}$$

where $\Sigma_{\mathbf{X}}$ is the conditional covariance matrix of $\mathbf{Y}$ given all X variables, and $\Sigma_{\mathbf{X}^c}$ is the conditional covariance matrix of $\mathbf{Y}$ given only the set of controlling X variables. The proposed significance test is described in Algorithm 3.2. After computing the covariance matrix estimates for all covariates and control variables only, we compute the test statistic with

$$T = \frac{1}{n}\sum_{i=1}^{n} d\big(\hat{\Sigma}_{\mathbf{x}_i}, \hat{\Sigma}_{\mathbf{x}_i^c}\big), \tag{3.4}$$

where $d(.,.)$ is computed as (3.2). The test statistic specifies how much the covariance matrix estimates given all covariates differ from the estimates given only the controlling set of covariates. As T becomes larger, we have more evidence against H_0.

We conduct a permutation test under the null hypothesis (3.3) by randomly permuting rows of $\mathbf{X}$. Let R be the total number of permutations and T_r be the global test statistic (3.4) computed for the rth permuted $\mathbf{X}$. We estimate the test p-value with

$$p = \frac{1}{R}\sum_{r=1}^{R} I(T_r > T), \tag{3.5}$$

and we reject the null hypothesis (3.3) at a pre-specified level α if the p-value is less than α.

In the significance test described above, we need to apply the proposed method many times: for the original data with *(i)* all covariates and *(ii)* the set of control covariates, and at each permutation for the permuted data with *(iii)* all covariates and *(iv)* the set of control covariates. The proposed method applies a `nodesize` tuning as described in the previous section. Since tuning the `nodesize` parameter can be computationally demanding, we tune the `nodesize` for the original data with all covariates and with the set of control covariates only and use those tuned values for their corresponding permutation steps.

The proposed significance test has two particular cases of interest. The first is to evaluate the global effect of the covariates on the conditional covariance estimates. If $\mathbf{X}$

Algorithm 3.2 Permutation test for a subset of covariates effect

1: Train RF with $\mathbf{X}$ and $\mathbf{Y}$, estimate covariance matrices as described in Algorithm 3.1, say $\hat{\Sigma}_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
2: Train RF with $\mathbf{X}^c$ and $\mathbf{Y}$, and estimate covariance matrices as described in Algorithm 3.1, say $\hat{\Sigma}^c_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
3: Compute test statistic with $T = \frac{1}{n}\sum_{i=1}^{n} d\big(\hat{\Sigma}_{\mathbf{x}_i}, \hat{\Sigma}^c_{\mathbf{x}_i}\big)$, where $d(.,.)$ is computed as (3.2)
4: **for** $r = 1 : R$ **do**
5: Permute rows of $\mathbf{X}$, say $\mathbf{X}_r$
6: Train RF with $\mathbf{X}_r$ and $\mathbf{Y}$
7: Estimate covariance matrices as described in Algorithm 3.1, say $\hat{\Sigma}'_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
8: Train RF with $\mathbf{X}^c_r$ and $\mathbf{Y}$
9: Estimate covariance matrices as described in Algorithm 3.1, say $\hat{\Sigma}^{c'}_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
10: Compute test statistic with $T_r = \frac{1}{n}\sum_{i=1}^{n} d\big(\hat{\Sigma}'_{\mathbf{x}_i}, \hat{\Sigma}^{c'}_{\mathbf{x}_i}\big)$
11: **end for**
12: Approximate the permutation p-value with $p = \frac{1}{R}\sum_{r=1}^{R} I(T_r > T)$
13: Reject the null hypothesis when $p < \alpha$. Otherwise, do not reject the null hypothesis.

has a global effect on the covariance matrix estimates obtained with the proposed method, then the conditional estimates $\Sigma_{\mathbf{X}}$ should be significantly different from the unconditional covariance matrix estimate Σ_{root} which is computed as the sample covariance matrix of $\mathbf{Y}$. The null hypothesis (3.3) becomes

$$H_0 : \Sigma_{\mathbf{X}} = \Sigma_{root}. \tag{3.6}$$

See Appendix I for the details of the global significance test. The second case is to evaluate the effect of a single covariate when the other covariates are in the model. In that particular case, the null hypothesis (3.3) remains. The only difference between the global and partial significance tests is the number of forests we need to train. In the partial significance test, we need to train two random forests per sample, one for all covariates and one for the controlling variables, which makes a total $2R+2$ random forests. However, when we test for the global effect, we need to train only one random forest per sample (in total $R+1$ random forests) since we do not need to build a random forest for the root node.

3.2.4 Variable importance

For traditional regression tree problems, we can get the variable importance (VIMP) measures by computing the average change in prediction accuracy using the OOB samples. However, the covariance regression problem does not have an observed target. We can compute the VIMP measures by using the *fit-the-fit* approach which has been applied to enhance interpretability of the covariates on the response (Lee et al., 2020; Alakuş et al., 2021; Bargagli Stoffi et al., 2021; Spanbauer and Sparapani, 2021; Stoffi et al., 2021; Meid et al., 2022). In the univariate response case, we get the importance measures by fitting a regression forest to re-predict the predicted values. However, in covariance regression, we have a predicted covariance matrix for each observation and not a single value. Therefore, we use a multivariate splitting rule based on the Mahalanobis distance (Ishwaran et al., 2021) to re-predict the predicted covariance matrices. We begin by applying the proposed method using the original covariates and responses and estimate the covariance matrices as described in Algorithm 3.1. Next, we train a random forest with the original covariates and the vector of upper-triangular estimated covariance matrix elements as a multivariate response. VIMP measures are obtained from this random forest. Covariates with higher VIMP measures indicate higher importance for the estimation of covariance matrices.

3.2.5 Implementation

We have developed an R package called `CovRegRF`. We used the custom splitting feature of the `randomForestSRC` package (Ishwaran and Kogalur, 2022) to implement our specially designed splitting criterion in the tree building process. The package is available on CRAN, https://CRAN.R-project.org/package=CovRegRF.

3.3 Simulations

In this section, we perform a simulation study to demonstrate the performance of the proposed method, validate the proposed significance test with two particular cases—global

and partial significance tests—and evaluate the variable importance estimations of the covariates.

3.3.1 Data generating process

We carry out a simulation study using four Data Generating Processes (DGPs). The details of the DGPs are given in Appendix J. The first two DGPs are variations of the first simulated data set used in Hoff and Niu (2012). Both DGPs include one covariate and two response variables. The covariate x is generated uniformly on $[-1,1]$. In DGP1, the covariance matrix for the observation x_i is $\Sigma_{\mathbf{x_i}} = \Psi + \mathbf{B}\mathbf{x}_i\mathbf{x}_i^T\mathbf{B}^T$ where $\mathbf{x}_i^T = (1, x_i)^T$. DGP2 is similar to DGP1, except that we add a quadratic term to the covariance matrix equation such as $\Sigma_{\mathbf{x_i}} = \Psi + \mathbf{B}\dot{\mathbf{x}}_i\dot{\mathbf{x}}_i^T\mathbf{B}^T$ where $\dot{\mathbf{x}}_i^T = (1, (x_i + x_i^2))^T$.

In DGP3, the vector of covariates includes seven independent variables generated from the standard normal distribution. For the covariance structure, we use an AR(1) structure with heterogeneous variances. The correlations are generated with all seven covariates according to a tree model with a depth of three and eight terminal nodes. The variances are functions of the generated correlations. In DGP4, the covariance matrix has a compound symmetry structure with heterogeneous variances. Both variances and correlations are functions of covariates. The covariates are generated from the standard normal distribution. The correlations are generated with a logit model and the variances are functions of these generated correlations. The number of covariates and response variables varies depending on the simulation settings. For all DGPs, after generating $\Sigma_{\mathbf{x}_i}$, $\mathbf{y}_i$ is generated from a multivariate normal distribution $N(\mathbf{0}, \Sigma_{\mathbf{x}_i})$.

3.3.2 Simulation design

Accuracy evaluation

We perform a simulation study based on the four DGPs described above to evaluate the accuracy of the proposed method for estimating the covariance matrices. For DGP3 and DGP4, we consider five response variables. For each DGP, we use several values of

the training sample size $n_{train} = \{50, 100, 200, 500, 1000\}$, which generates a total of 20 settings (4 DGPs $\times$ 5 training sample sizes). We repeat each setting 100 times. In each run of the simulations, we generate an independent test set of new observations with $n_{test} = 1000$.

We evaluate the performance of the covariance matrix estimates using the mean absolute errors (MAE) computed for both the estimated correlations and standard deviations separately. For the estimated correlations, we compute the MAE between the upper triangular (off-diagonal) matrices of the true and estimated correlations over all observations as follows:

$$MAE^{cor}(\hat{\mathbf{C}}_{\mathbf{X}}, \mathbf{C}_{\mathbf{X}}) = \frac{2}{q(q-1)n_{test}} \sum_{i=1}^{n_{test}} \sum_{j=1}^{q} \sum_{k=j+1}^{q} |\hat{\rho}_{ijk} - \rho_{ijk}|,$$

where $\mathbf{C}_{\mathbf{X}}$ and $\hat{\mathbf{C}}_{\mathbf{X}}$ are the collection of all correlation matrices corresponding to $\Sigma_{\mathbf{X}}$ and $\hat{\Sigma}_{\mathbf{X}}$, respectively. The values ρ_{ijk} and $\hat{\rho}_{ijk}$ represent the correlations in row j and column k of $\mathbf{C}_{\mathbf{x}_i}$ and $\hat{\mathbf{C}}_{\mathbf{x}_i}$, respectively.

For the estimated standard deviations, we compute the normalized MAE between the true and estimated standard deviations over all observations as follows:

$$MAE^{sd}(\hat{\Sigma}_{\mathbf{X}}, \Sigma_{\mathbf{X}}) = \frac{1}{qn_{test}} \sum_{i=1}^{n_{test}} \sum_{j=1}^{q} \left| \frac{\hat{\sigma}_{ij} - \sigma_{ij}}{\sigma_{ij}} \right|.$$

The values σ_{ij}^2 and $\hat{\sigma}_{ij}^2$ represent the jth diagonal element of $\Sigma_{\mathbf{x}_i}$ and $\hat{\Sigma}_{\mathbf{x}_i}$, respectively.

Smaller values of MAE^{cor} and MAE^{sd} indicate better performance. We compare our proposed method with the original Gaussian-based covariance regression model `covreg` which was presented in the Introduction. This method is currently available in the `covreg` R package (Niu and Hoff, 2014). Moreover, as a simple benchmark method, we compute the sample covariance matrix without covariates, which is then used as the covariance matrix estimate for all new observations from the test set.

Variable importance

For the variable importance evaluation simulations, we use DGP3 and DGP4 in which we add five noise variables X to the covariates set. As above, we consider several values

for the training sample sizes $n_{train} = \{50, 100, 200, 500, 1000\}$, for a total of 10 scenarios studied. We examine whether the estimated VIMP measures tend to rank the important variables first. The variable with the highest VIMP measure has a rank of 1. For each scenario, we compute the average rank for the important variables group and for the noise variables group.

Evaluating the power of the global significance test

We studied four scenarios to evaluate the global effect of the covariates, two of which are under the null hypothesis (3.6) and the other two under the alternative hypothesis. We generate the data sets for these scenarios as follows:

1. H_0 (case 1): we generate 5 Y with a constant population covariance matrix and 10 X variables which are all independent following a standard normal distribution. In this case, the covariance of Y is independent of X and we are therefore under the null hypothesis.

2. H_0 (case 2): we first generate 7 X and 5 Y under DGP3. Then, we replace the $\mathbf{X}$ matrix with 10 independent X variables generated from a standard normal distribution. In this case, the covariance of $\mathbf{Y}$ varies with some of the X variables but those X variables are not available in the training set. Therefore, we are again under the null hypothesis.

3. H_1 (without noise): we generate 7 X and 5 Y under DGP3, and the covariates are available in the training set. In this case, the covariance of $\mathbf{Y}$ varies with all X variables.

4. H_1 (with noise): we generate 7 X and 5 Y under DGP3 and we add 3 independent X variables to the covariates' training set. In this case, the covariance of $\mathbf{Y}$ varies with some of the X variables but not all.

Evaluating the power of the partial significance test

We can consider three scenarios to evaluate the effect of a single covariate, where one is under the null hypothesis (3.3) and the other two under the alternative hypothesis. We generate the data sets for these scenarios as follows:

1. H_0: We first generate 2 X and 5 Y with DGP4 and we add 1 independent X variable to the covariates' training set. In this case, the covariance of $\mathbf{Y}$ varies only with the first two X variables. The control set of variables is $\{X_1, X_2\}$ and we evaluate the effect of the X_3 variable. Therefore, we are under the null hypothesis.

2. $H_1(weakest)$: We generate 3 X and 5 Y with DGP4. In this case, the covariance of $\mathbf{Y}$ varies with all X variables. The control set of variables is $\{X_1, X_2\}$ and we evaluate the effect of X_3, which has the weakest effect on the covariance matrix.

3. $H_1(strongest)$: We generate 3 X and 5 Y with DGP4. In this case, the covariance of $\mathbf{Y}$ again varies with all X variables. But now the control set of variables is $\{X_2, X_3\}$ and we evaluate the effect of X_1, which has the strongest effect on the covariance matrix.

For both the global and partial significance test simulations, we use training sample sizes of $n_{train} = \{50, 100, 200, 300, 500\}$. The number of permutations and the number of replications for each scenario are set to 500. We estimate the type-1 error as the proportion of rejection in the scenarios simulated under H_0 and the power as the proportion of rejection in the scenarios simulated under H_1. We estimate a p-value for each replication and we reject the null hypothesis if the p-value is less than the significance level $\alpha = 0.05$. Finally, we compute the proportion of rejection over 500 replications.

3.3.3 Parameter settings

For the simulations, we use the following parameters for the proposed method. We set the number of trees to 1000. Letting p be the number of covariates, then the number of

covariates to randomly split at each node, `mtry`, is set to $\lceil p/3 \rceil$. The number of random splits for splitting a covariate at each node, `nsplit`, is set to $\max\{n_{train}/50, 10\}$. We tune the `nodesize` parameter with the set of `nodesize`$= \{[sampsize \times (2^{-1}, 2^{-2}, 2^{-3}, \ldots)] > q\}$ where q is the number of responses and `sampsize`$= 0.632 n_{train}$. In each replication, `covreg` is run in four independent chains for 8000 iterations, with the first half taken as burn-in.

3.3.4 Results

Accuracy evaluation

Figures 3.1 and 3.2 present the accuracy results for 100 repetitions. For each method, we can see the change in MAE^{cor} and MAE^{sd} computed for 100 repetitions with an increasing training sample size. As demonstrated in Figure 3.1, for DGP1 and DGP2 when $n_{train} = 50$, the proposed method and `covreg` both have a similar performance with respect to the correlation estimation, with a slight advantage for `covreg`. For DGP1, `covreg` performs better for both the correlation and standard deviation compared to the proposed method as the sample size increases. This is expected since DGP1 is generated exactly under the `covreg` model. However, the proposed method still remains competitive. For DGP2, in which a quadratic term is added, the proposed method performs better for the correlation than `covreg` with increasing sample size. `covreg` shows better standard deviation estimation performance for smaller sample sizes, but after $n_{train} = 500$ the proposed method performs slightly better. As demonstrated in Figure 3.2, for DGP3, the proposed method shows a significantly smaller MAE^{cor} and MAE^{sd} than `covreg` for all sample sizes. Moreover, for the smaller sample sizes, the proposed method has considerably lower variance in MAE. For DGP4, both methods improve with increasing sample size, but the proposed method shows smaller or equal MAEs for both correlation and standard deviation estimations. For DGP3 and DGP4, these results are expected, since the proposed method can capture a nonlinear effect.

For the `nodesize` tuning, we compare the accuracy results for different levels of

`nodesize` along with the proposed tuning method. Figures H1 and H2 in Appendix H present the MAE results for all DGPs which show that the tuning method works well.

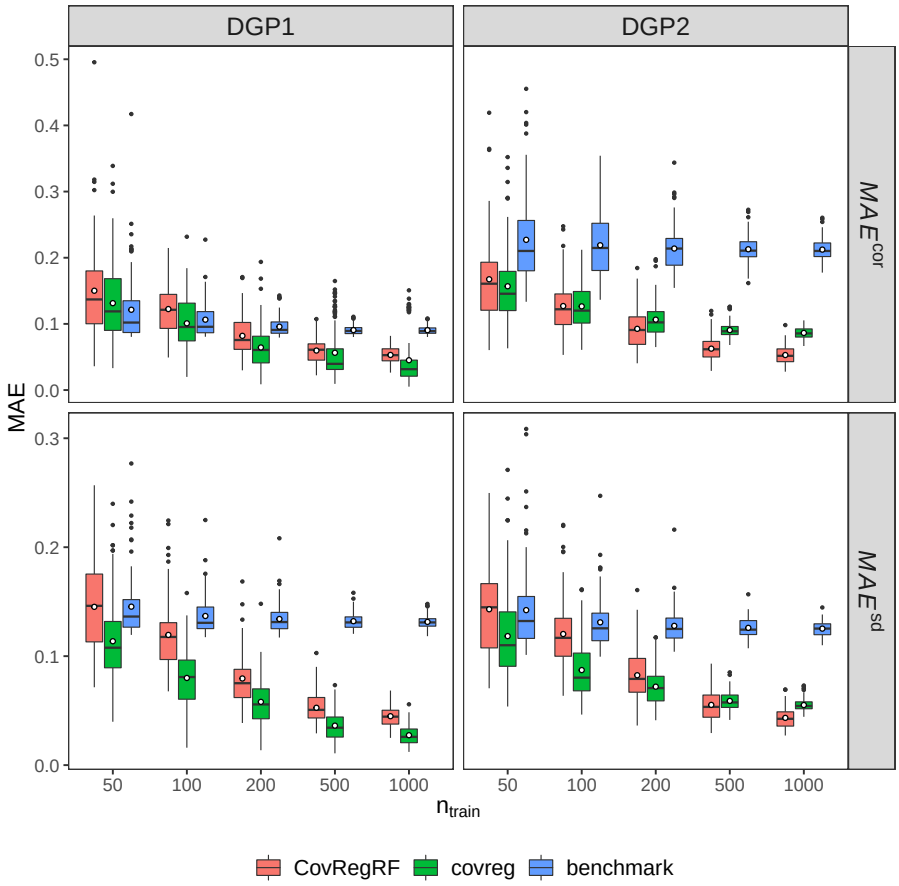

Figure 3.1: Accuracy evaluation results for DGP1 and DGP2. Smaller values of MAE^{cor} and MAE^{sd} are better.

Variable importance

Figure 3.3 presents the average ranks of the VIMP measures for both the important and noise sets of variables for DGP3 and DGP4. In all scenarios, the important variables have smaller average ranks than noise variables. As the sample size increases, the difference between the average ranks of important and noise variables increases, as expected.

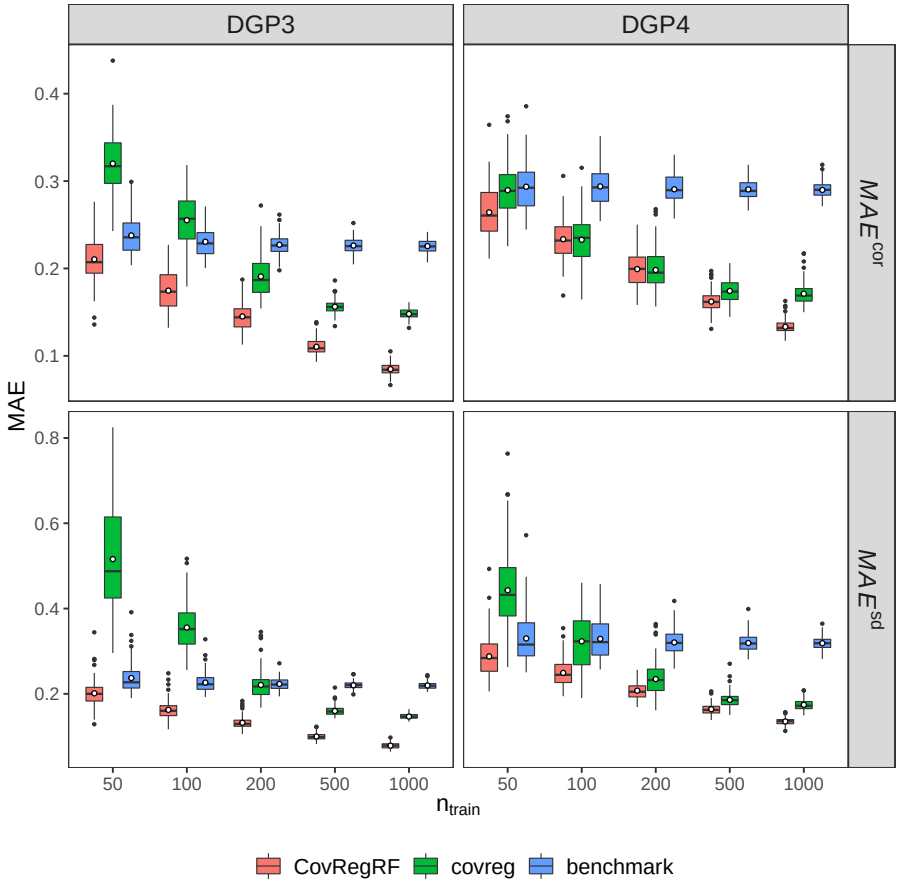

Figure 3.2: Accuracy evaluation results for DGP3 and DGP4. Smaller values of MAE^{cor} and MAE^{sd} are better.

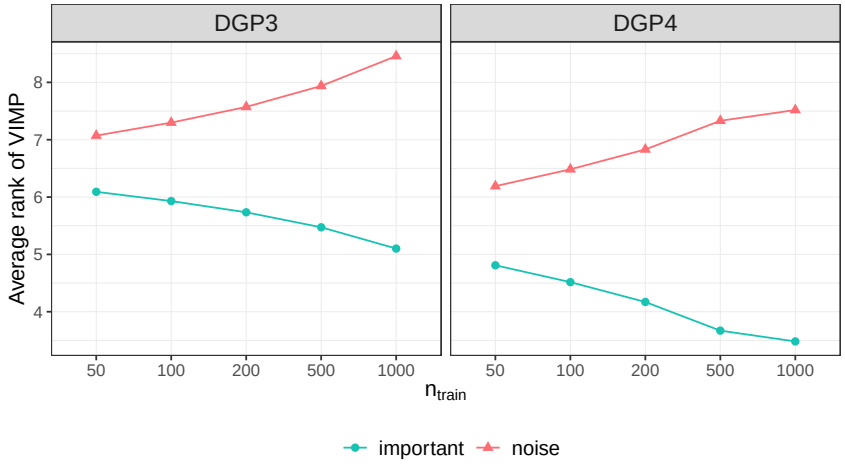

Figure 3.3: Average ranks from estimated VIMP measures for DGP3 and DGP4. Smaller rank values indicate a more important variable (the most important variable has rank 1).

Global significance test

The left plot in Figure 3.4 presents the estimated type-1 error and power for different training sample sizes for the two H_0 scenarios and two H_1 scenarios, respectively. We expect the type-1 error to be close to the significance level ($\alpha = 0.05$) and we can see that it is well controlled in both cases studied. In both H_1 scenarios, the power increases with the sample size. When the sample size is small, adding noise covariates slightly decreases the power, but this effect disappears as the sample size increases.

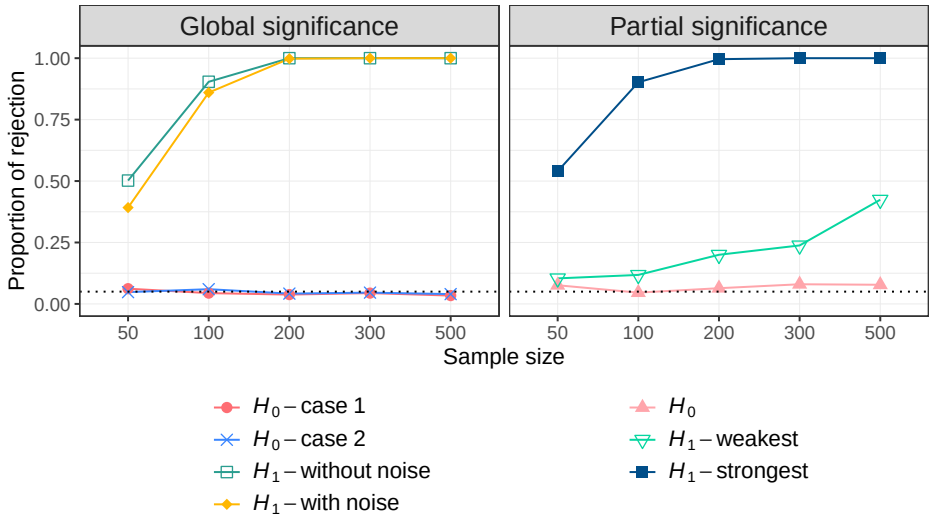

Figure 3.4: Significance test results. The left and right plots present the results for global and partial significance tests, respectively. The proportion of rejection corresponds to the type-1 error for H_0 scenarios, and power for H_1 scenarios. The dotted line represents the significance level of $\alpha = 0.05$.

Partial significance test

The right plot in Figure 3.4 presents the estimated type-1 error and power for different training sample sizes for the H_0 scenario and two H_1 scenarios, respectively. As can be seen from the H_0 line, the type-1 error is close to the significance level ($\alpha = 0.05$). In both H_1 scenarios, the power increases with the sample size as expected. However, the power is much smaller when one tests the weakest covariate compared to the strongest covariate.

3.4 Real data example

Thyroid hormone, the collective name for two hormones, is widely known for regulating several body processes, including growth and metabolism (Yen, 2001; Shahid et al., 2022). The main hormones produced by the thyroid gland are triiodothyronine (T3) and thyroxine (T4). The synthesis and secretion of these hormones are primarily regulated by thyroid stimulating hormone (TSH), which is produced by the pituitary gland. Primary hypothyroidism is a condition that occurs when the thyroid gland is underactive and the thyroid hormone produced is insufficient to meet the body's requirements, which leads to an increase of TSH. Contrarily, when the thyroid gland produces levels of thyroid hormones that are too high, leading to decreased levels of TSH, the resulting condition is hyperthyroidism.

Serum levels of the thyroid hormones and TSH are used to evaluate subjects' thyroid function status and to identify subjects with a thyroid dysfunction. Therefore, establishing reference intervals for these hormones is critical in the diagnosis of thyroid dysfunction. However, reference ranges are affected by age and sex (Kapelari et al., 2008; Aggarwal and Razvi, 2013; Biondi, 2013; Strich et al., 2017; Park et al., 2018). Furthermore, there is a relationship between TSH and thyroid hormone, and the effects of age and sex on this relationship have not been well described (Hadlow et al., 2013; Lee et al., 2020). Serum levels of these hormones are also affected by the subject's diagnosis, i.e. hormone levels would be within the reference ranges for normal subjects and out of range for subjects with thyroid dysfunction. The conditional mean of these hormones based on the covariates is studied in the literature, but to our knowledge, no study has yet explicitly investigated the effect of covariates on the conditional covariance matrix of these hormones. Hence, our contribution is to study the effect of age, sex and diagnosis on the covariance matrix of the thyroid hormones and TSH.

In this study, we investigate the thyroid disease data set from the UCI machine learning repository (Dua and Graff, 2017). This data set originally included 9172 subjects and 30 variables including age, sex, hormone levels and diagnosis. Following the exclusion

criteria applied in Hadlow et al. (2013) and Strich et al. (2017), we exclude pregnant women, subjects who have euthyroid sick syndrome (ESS), goitre, hypopituitarism or tumour, subjects who use antithyroid medication, thyroxine or lithium, who receive I131 treatment, or who have had thyroid surgery. The subjects have different diagnoses including hypothyroidism and hyperthyroidism, as well as normal subjects. Since the sample size of hyperthyroidism subjects is small, we exclude them from the analysis. We also exclude the very young and very old subjects, since there are only a few subjects on the extremes. The remaining data set consists of 324 hypothyroidism and 2951 normal subjects ($n = 3275$) between 20 and 80 years of age (2021 females/1254 males). We want to estimate the covariance matrix of four thyroid-related hormones—TSH, T3, TT4 (total T4) and FTI (free thyroxine index/free T4)—based on covariates and investigate how the relationship between these hormones varies with the covariates. We apply the proposed method with the covariates age, sex and diagnosis to estimate the covariance matrix of the four hormones. We first perform the significance test with 500 permutations to evaluate the global effect of the three covariates. The estimated p-value with (3.5) is 0 and we reject the null hypothesis (3.6), which indicates that the conditional covariance matrices vary significantly with the set of covariates. Next, we apply the proposed method and obtain the covariance matrix estimates. We analyze the correlations between hormones as a function of covariates, and as shown in Figure 3.5, age seems not to have much effect on the estimated correlations. We also compute the variable importance measures, and age (0.001) is found to be the least important variable where diagnosis (1.000) is the most important variable, followed by sex (0.011). Therefore, we apply the significance test to evaluate the effect of age on covariance matrices while controlling for sex and diagnosis. Using 500 permutations, the estimated p-value with (3.5) is 0.42 and we fail to reject the null hypothesis (3.3), indicating that we have insufficient evidence to prove that age has an effect on the estimated covariance matrices while sex and diagnosis are in the model. Although the mean levels of TSH and thyroid hormones differ with age (Kapelari et al., 2008; Aggarwal and Razvi, 2013; Biondi, 2013; Park et al., 2018), the correlation between these hormones may not be affected by aging. Similarly, we apply the signifi-

cance test for diagnosis and sex while controlling for the remaining two covariates, and the estimated p-values for both tests are 0, which indicates that both diagnosis and sex, taken individually, have an effect on the covariance matrix of the four hormones. We compare the estimated correlations using the proposed method to the sample correlations computed using the whole sample, which are represented with the black dashed lines in Figure 3.5. For example, the sample correlation between TSH and T3 over all samples is -0.28 which is not close to the estimated correlation of either hypothyroidism or normal subjects. Furthermore, the estimated variances of the four hormones as a function of age, sex and diagnosis are presented in Figure 3.6. We can see that the variances also differ with covariates. For a mean regression analysis for any of these hormones, assuming a constant variance could yield misleading results.

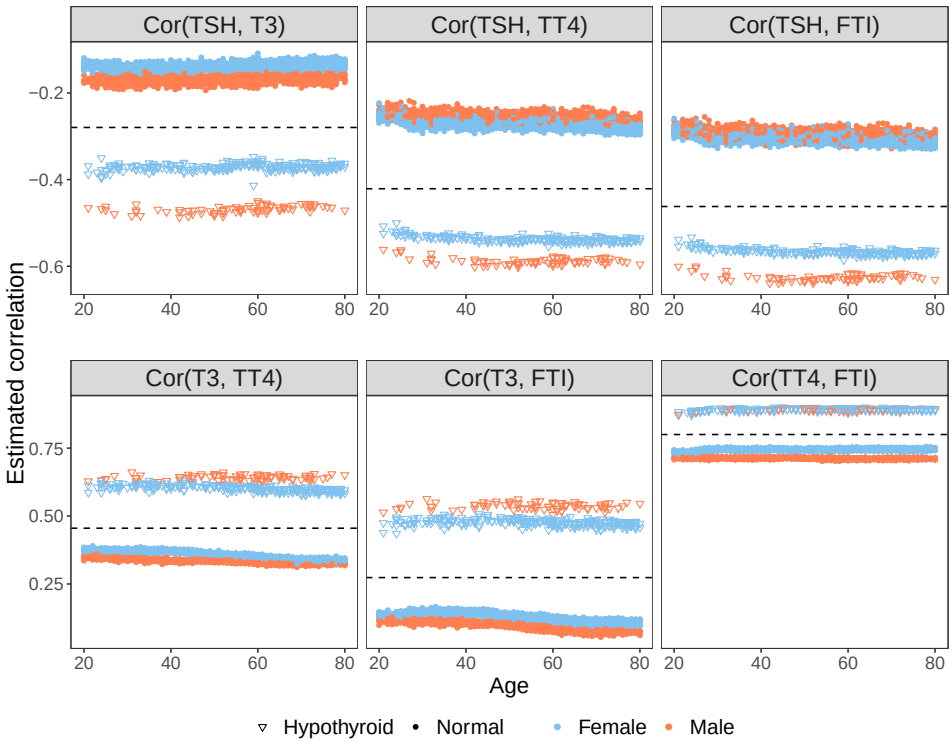

Figure 3.5: Estimated correlations between the four hormones as a function of age, sex and diagnosis. Dashed lines represent the sample correlations computed using the whole sample.

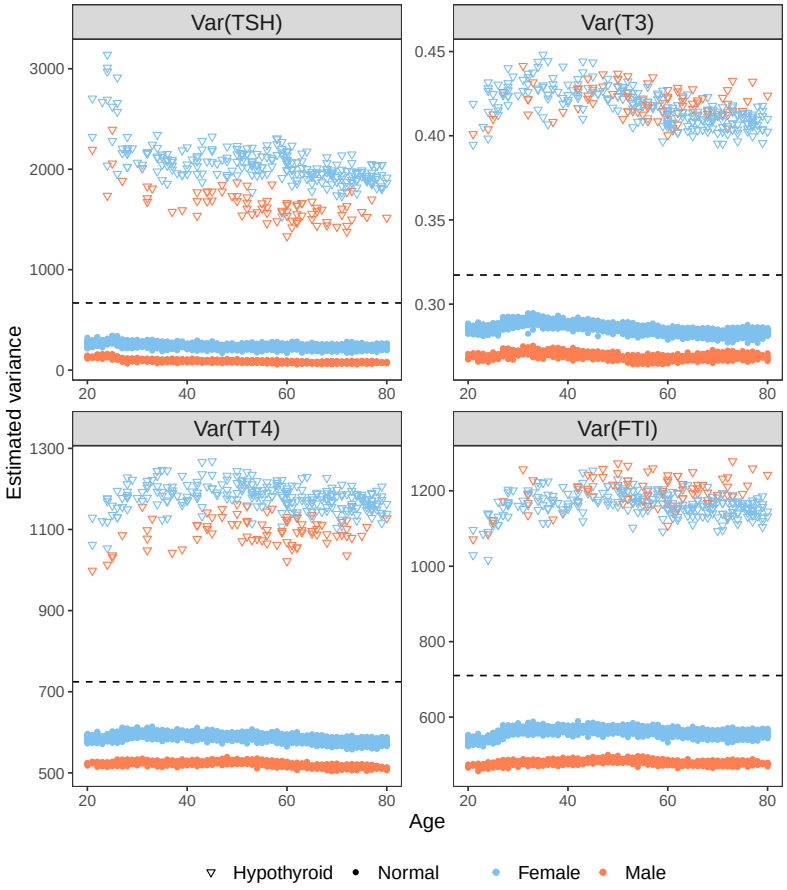

Figure 3.6: Estimated variances for the four hormones as a function of age, sex and diagnosis. Dashed lines represent the sample variances computed using the whole sample.

3.5 Conclusion

In this study, we propose a nonparametric covariance regression method, using a random forest framework, for estimating the covariance matrix of a multivariate response given a set of covariates. Random forest trees are built with a new splitting rule designed to maximize the distance between the sample covariance matrix estimates of the child nodes. For a new observation, the random forest provides the set of nearest neighbour out-of-bag (OOB) observations which is used to estimate the conditional covariance matrix for that observation. We perform a simulation study to test the performance of the proposed method and compare it to the original Gaussian-based covariance regression model `covreg`. The average computational times of both methods for the simulations are

presented in Appendix K. We can see from the table that the proposed method is significantly faster than `covreg`. For the real data analysis, the computational time was 200.14 seconds. Furthermore, we propose a significance test to evaluate the effect of a subset of covariates while the other covariates are in the model. We investigate two particular cases: the global effect of covariates and the effect of a single covariate. We also propose a way to compute variable importance measures.

The proposed method can be extended in a variety of ways. One of them is to compute the weighted Euclidean distance between covariance matrices of the child nodes as $d(\mathbf{A},\mathbf{B}) = \sqrt{\sum_{i=1}^{q}\sum_{j=i}^{q} w_{ij}(\mathbf{A}_{ij} - \mathbf{B}_{ij})^2}$ for the splitting rule. The weights are like the measures of importance for the elements of the covariance matrices. Another one is that for the final covariance matrix estimation for a new observation, we can use sparse or robust covariance matrix estimations (Rousseeuw and Driessen, 1999; Bien and Tibshirani, 2011) using the nearest neighbour observations. Similarly, it is theoretically possible to use the sparse or robust covariance matrix estimations instead of the sample covariance matrix for the tree building process. However, the computational time could be a limiting factor. The proposed method can be applied to larger $\mathbf{X}$ dimensions. The computational time increases linearly with `mtry` which is the number of covariates to randomly split at each node. It can also be adapted to larger $\mathbf{Y}$ dimensions, but the computational time could be a limitation for very large $\mathbf{Y}$ dimensions. Computing the sample covariance matrix has a time complexity $\mathcal{O}(nq^2)$ for q response variables and we compute covariance matrix for each node split in each tree of the forest which necessitates many covariance matrix computations.

References

Aggarwal, N. and Razvi, S. (2013). Thyroid and aging or the aging thyroid? An evidence-based analysis of the literature. *Journal of thyroid research*, 2013.

Alakuş, C., Larocque, D., Jacquemont, S., Barlaam, F., Martin, C.-O., Agbogba, K.,

Lippé, S., and Labbe, A. (2021). Conditional canonical correlation estimation based on covariates with random forests. *Bioinformatics*, 37(17):2714–2721.

Alakuş, C., Larocque, D., and Labbe, A. (2022). The R Journal: RFpredInterval: An R Package for Prediction Intervals with Random Forests and Boosted Forests. *The R Journal*, 14(1):300–320.

Athey, S., Tibshirani, J., and Wager, S. (2019). Generalized random forests. *The Annals of Statistics*, 47(2):1148–1178.

Bargagli Stoffi, F. J., Gnecco, G., and De Witte, K. (2021). Heterogeneous causal effects with imperfect compliance: a Bayesian machine learning approach. *Annals Of Applied Statistics*.

Bien, J. and Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.

Biondi, B. (2013). The Normal TSH Reference Range: What Has Changed in the Last Decade? *The Journal of Clinical Endocrinology & Metabolism*, 98(9):3584–3587.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press, Boca Raton.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Fox, E. B. and Dunson, D. B. (2015). Bayesian nonparametric covariance regression. *The Journal of Machine Learning Research*, 16(1):2501–2542.

Franks, A. M. (2021). Reducing subspace models for large-scale covariance regression. *Biometrics*.

Hadlow, N. C., Rothacker, K. M., Wardrop, R., Brown, S. J., Lim, E. M., and Walsh, J. P. (2013). The Relationship Between TSH and Free T4 in a Large Population Is Complex

and Nonlinear and Differs by Age and Sex. *The Journal of Clinical Endocrinology & Metabolism*, 98(7):2936–2943.

Hoff, P. D. and Niu, X. (2012). A covariance regression model. *Statistica Sinica*, 22(2):729–753.

Hothorn, T., Lausen, B., Benner, A., and Radespiel-Tröger, M. (2004). Bagging survival trees. *Statistics in medicine*, 23(1):77–91.

Ishwaran, H. and Kogalur, U. (2022). *Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 3.1.0.

Ishwaran, H., Tang, F., Lu, M., and Kogalur, U. B. (2021). randomForestSRC: Multivariate splitting rule vignette.

Kapelari, K., Kirchlechner, C., Högler, W., Schweitzer, K., Virgolini, I., and Moncayo, R. (2008). Pediatric reference intervals for thyroid hormone levels from birth to adulthood: a retrospective study. *BMC Endocrine Disorders*, 8(1):15.

Le Goallec, A. and Patel, C. J. (2019). Age-dependent co-dependency structure of biomarkers in the general population of the United States. *Aging*, 11(5):1404–1426.

Lee, K., Bargagli-Stoffi, F. J., and Dominici, F. (2020). Causal rule ensemble: Interpretable inference of heterogeneous treatment effects. *arXiv preprint arXiv:2009.09036*.

Levy, R. and Borenstein, E. (2013). Metabolic modeling of species interaction in the human microbiome elucidates community-level assembly rules. *Proceedings of the National Academy of Sciences*, 110(31):12804–12809.

Lin, Y. and Jeon, Y. (2006). Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association*, 101(474):578–590.

Lu, B. and Hardin, J. (2021). A Unified Framework for Random Forest Prediction Error Estimation. *Journal of Machine Learning Research*, 22(8):1–41.

McGregor, K., Labbe, A., and Greenwood, C. M. T. (2020). MDiNE: a model to estimate differential co-occurrence networks in microbiome studies. *Bioinformatics*, 36(6):1840–1847.

Meid, A. D., Gerharz, A., and Groll, A. (2022). Machine learning for tumor growth inhibition: Interpretable predictive models for transparency and reproducibility. *CPT: Pharmacometrics & Systems Pharmacology*, 11(3):257.

Moradian, H., Larocque, D., and Bellavance, F. (2017). L1 splitting rules in survival forests. *Lifetime data analysis*, 23(4):671.

Moradian, H., Larocque, D., and Bellavance, F. (2019). Survival forests for data with dependent censoring. *Statistical methods in medical research*, 28(2):445–461.

Niu, X. and Hoff, P. (2014). *covreg: A simultaneous regression model for the mean and covariance*. R package version 1.0.

Niu, X. and Hoff, P. D. (2019). Joint mean and covariance modeling of multiple health outcome measures. *The annals of applied statistics*, 13(1):321–339.

Park, S. Y., Kim, H. I., Oh, H.-K., Kim, T. H., Jang, H. W., Chung, J. H., Shin, M.-H., and Kim, S. W. (2018). Age-and gender-specific reference intervals of TSH and free T4 in an iodine-replete area: data from Korean National Health and Nutrition Examination Survey IV (2013–2015). *PLoS One*, 13(2):e0190738.

Rousseeuw, P. J. and Driessen, K. V. (1999). A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics*, 41(3):212–223.

Roy, M.-H. and Larocque, D. (2020). Prediction intervals with random forests. *Statistical Methods in Medical Research*, 29(1):205–229.

Seiler, C. and Holmes, S. (2017). Multivariate Heteroscedasticity Models for Functional Brain Connectivity. *Frontiers in Neuroscience*, 11.

Shahid, M. A., Ashraf, M. A., and Sharma, S. (2022). Physiology, thyroid hormone. In *StatPearls [Internet]*. StatPearls Publishing.

Spanbauer, C. and Sparapani, R. (2021). Nonparametric machine learning for precision medicine with longitudinal clinical trials and Bayesian additive regression trees with mixed models. *Statistics in Medicine*, 40(11):2665–2691.

Stoffi, F. J. B., De Beckker, K., Maldonado, J. E., and De Witte, K. (2021). Assessing Sensitivity of Machine Learning Predictions. A Novel Toolbox with an Application to Financial Literacy. *arXiv preprint arXiv:2102.04382*.

Strich, D., Karavani, G., Edri, S., Chay, C., and Gillis, D. (2017). FT3 is higher in males than in females and decreases over the lifespan. *Endocrine Practice*, 23(7):803–807.

Tabib, S. and Larocque, D. (2020). Non-parametric individual treatment effect estimation for survival data with random forests. *Bioinformatics*, 36(2):629–636.

Wager, S. and Athey, S. (2018). Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *Journal of the American Statistical Association*, 113(523):1228–1242. Publisher: Taylor & Francis.

Yen, P. M. (2001). Physiological and molecular basis of thyroid hormone action. *Physiological reviews*, 81(3):1097–1142.

Yin, J., Geng, Z., Li, R., and Wang, H. (2010). Nonparametric covariance model. *Statistica Sinica*, 20:469.

Zou, T., Lan, W., Wang, H., and Tsai, C.-L. (2017). Covariance regression analysis. *Journal of the American Statistical Association*, 112(517):266–281.

General Conclusion

This thesis proposes a new method in each chapter, using the random forest framework,

1. Random Forest with Canonical Correlation Analysis (RFCCA) estimates the conditional canonical correlation between two multivariate data sets given the subject-related covariates,

2. Prediction Intervals with Boosted Forests (PIBF) builds a prediction interval for the bias-corrected point prediction,

3. Covariance Regression with Random Forests (CovRegRF) estimates the conditional covariance matrix of a multivariate response vector based on covariates.

The first and third chapters are similar in terms of the idea of estimating a conditional measure, which is the canonical correlation for the first chapter and the covariance matrix for the third, given a set of covariates using a random forest. Along the same line, the proposed splitting criteria in these two chapters target maximizing the difference in child nodes in order to identify the subgroups of subjects with distinct canonical correlations or covariance matrices. For a new observation, a set of nearest neighbours are formed from the forest and the conditional measure of interest is computed using these neighbours. Indeed, estimating the conditional canonical correlation between two univariate data sets based on a set of covariates in the first chapter is analogous to estimating the conditional correlation matrix of a two-dimensional response vector given a set of covariates in the third chapter. Moreover, we propose a significance test to evaluate the effect of the covariates on the estimated canonical correlations or covariance matrices and a way to compute

variable importance measures. Both methods are evaluated through simulation studies that show they provide accurate canonical correlation or covariance matrix estimations. In both chapters, an application of the proposed method is demonstrated.

In the second chapter, the proposed method provides bias-corrected point predictions obtained with the one-step boosted forest and prediction intervals built from the conditional prediction error distribution which is estimated using the set of nearest neighbour observations. We perform a simulation study and apply real data analyses to compare the performance of the proposed method to ten competing methods. The results show that the proposed method, globally, provides the shortest prediction intervals among all methods.

The proposed methods presented in this thesis are implemented in freely available R packages on CRAN and can be very useful in practice.

The proposed methods in the first and third chapters can be extended in a variety of ways. One of them is that for the final canonical correlation estimation for a new observation in the first chapter, we can use sparse CCA (Witten et al., 2009) or regularized CCA (Leurgans et al., 1993; Vinod, 1976) using the nearest neighbour observations. Similarly, for the final covariance matrix estimation for a new observation in the third chapter, we can use sparse or robust covariance matrix estimations (Rousseeuw and Driessen, 1999; Bien and Tibshirani, 2011). We can also use these sparse or robust variants instead of CCA or sample covariance matrix for the tree building process. However, the computational time could be a limiting factor for both methods. Other extensions would be to use alternative splitting rules to build the trees, such as the weighted sum of canonical correlations in the child nodes or the weighted euclidean distance between covariance matrices of the child nodes. Extending the proposed method in the second chapter to time-series data or spatial data would be interesting for future research.

Bibliography

Bach, F. R. and Jordan, M. I. (2002). Kernel independent component analysis. *Journal of machine learning research*, 3(Jul):1–48.

Bien, J. and Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.

Björck, Å. and Golub, G. H. (1973). Numerical methods for computing angles between linear subspaces. *Mathematics of computation*, 27(123):579–594.

Branco, J. A., Croux, C., Filzmoser, P., and Oliveira, M. R. (2005). Robust canonical correlations: A comparative study. *Computational Statistics*, 20(2):203–229.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Delorme, A. and Makeig, S. (2004). Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21.

Delorme, A., Sejnowski, T., and Makeig, S. (2007). Enhanced detection of artifacts in eeg data using higher-order statistics and independent component analysis. *Neuroimage*, 34(4):1443–1449.

Do Q, L. (2012). Numerically efficient methods for solving least squares problems.

Ewerbring, L. M. et al. (1990). Canonical correlations and generalized svd: applications and new algorithms. In *Advances in Parallel Computing*, volume 1, pages 37–52. Elsevier.

Ghosal, I. and Hooker, G. (2021). Boosting Random Forests to Reduce Bias; One-Step Boosted Forest and Its Variance Estimate. *Journal of Computational and Graphical Statistics*, 30(2):493–502.

Golub, G. H. (1969). Matrix decompositions and statistical calculations. In *Statistical computation*, pages 365–397. Elsevier.

Hardoon, D. R., Szedmak, S., and Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664.

Healy, M. (1957). A rotation method for computing canonical correlations. *Mathematics of Computation*, 11(58):83–86.

Herrmann, C. S., Grigutsch, M., and Busch, N. A. (2005). Eeg oscillations and wavelet analysis. *Event-related potentials: A methods handbook*, pages 229–259.

Hotelling, H. (1936). Relations between two sets of variates. *Biometrika*, 28(3/4):321–377.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, 15(3):651–674.

Ishwaran, H. (2007). Variable importance in binary regression trees and forests. *Electronic Journal of Statistics*, 1:519–537.

Ishwaran, H. and Kogalur, U. (2020). *Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 2.9.3.

Lachaux, J.-P., Rodriguez, E., Martinerie, J., and Varela, F. J. (1999). Measuring phase synchrony in brain signals. *Human brain mapping*, 8(4):194–208.

Leurgans, S. E., Moyeed, R. A., and Silverman, B. W. (1993). Canonical correlation analysis when the data are curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 55(3):725–740.

Makeig, S., Debener, S., Onton, J., and Delorme, A. (2004). Mining event-related brain dynamics. *Trends in cognitive sciences*, 8(5):204–210.

Rousseeuw, P. J. and Driessen, K. V. (1999). A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics*, 41(3):212–223.

Tallon-Baudry, C. and Bertrand, O. (1999). Oscillatory gamma activity in humans and its role in object representation. *Trends in cognitive sciences*, 3(4):151–162.

Tucker, D. M. (1993). Spatial sampling of head electrical fields: the geodesic sensor net. *Electroencephalography and clinical neurophysiology*, 87(3):154–163.

Vinod, H. D. (1976). Canonical ridge and econometrics of joint production. *Journal of econometrics*, 4(2):147–166.

Wilms, I. and Croux, C. (2015). Sparse canonical correlation analysis from a predictive point of view. *Biometrical Journal*, 57(5):834–851.

Witten, D. M., Tibshirani, R., and Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534.

Appendix A – Motivating examples

Example with univariate X and Y

We generated a data set with $Z = (Z_1, \ldots, Z_{10}) \in R^{n \times 10}$, $X \in R^{n \times 1}$ and $Y \in R^{n \times 1}$ where $n = 500$, $Z_i \sim N(0,1)\ \forall i = \{1,2,...,10\}$ and $(X,Y) \sim MVN(0,\Sigma)$ where $\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$. The correlation between X and Y is a function of Z_1 given by

$$\rho = \begin{cases} 0 & Z_1 \leq 0, \\ 0.8 & Z_1 > 0, \end{cases}$$

where ρ is the population correlation between X and Y. The sample correlation between X and Y, as a function of Z_1, is

$$r(X,Y) = \begin{cases} 0.017 & Z_1 \leq 0, \\ 0.737 & Z_1 > 0, \end{cases}$$

whereas the sample correlation between all X and Y is 0.329. Our aim is to identify the two groups of observations having the most different correlation. After applying the proposed method with a single tree and a single split, we have two groups of observations. Figure A1 illustrates the single split of the method. The observations are grouped according to their Z_1 values, the first group has $Z_1 \leq 0.011$ and the second group has $Z_1 > 0.011$. The correlation between X and Y in those two groups are respectively 0.018 and 0.741. Hence, the proposed method was able to identify the two groups.

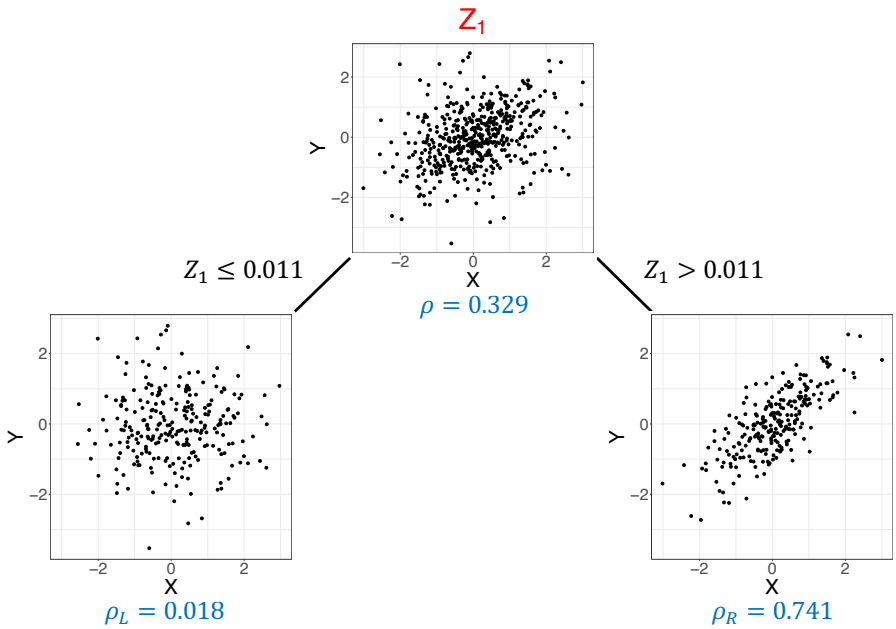

Figure A1: Illustration for the example, single split of a single decision tree.

Example with multivariate X and Y

As a second example, now with multivariate X and Y, we have a data set with $Z \in R^{n\times 10}$, $X \in R^{n\times 2}$ and $Y \in R^{n\times 2}$ where $n = 500$, $Z_i \sim N(0,1)\ \forall i = \{1,2,...,10\}$. The canonical correlation between X and Y is a function of Z_1

$$\rho_{CCA} = \begin{cases} 0.2 & Z_1 \leq 0, \\ 0.8 & Z_1 > 0, \end{cases}$$

where ρ_{CCA} is the population CCA correlation between X and Y. The sample CCA correlations between X and Y as a function of Z_1 is

$$r_{CCA}(X,Y) = \begin{cases} 0.237 & Z_1 \leq 0, \\ 0.785 & Z_1 > 0, \end{cases}$$

whereas the sample CCA correlation between all X and Y is 0.535. Again, after applying the proposed method with a single tree and single split, we have two groups of observations. The observations are grouped according to their Z_1 values, the first group has $Z_1 \leq 0.014$ and the second group has $Z_1 > 0.014$. The CCA correlation between X and Y

in those two groups are respectively 0.237 and 0.785. Thus, again, the proposed method was able to identify the two groups.

Appendix B – Computing canonical correlations

Canonical correlation analysis, firstly introduced in Hotelling (1936), seeks vectors of $a \in R^p$ and $b \in R^q$ for two multivariate data sets $X \in R^{n \times p}$ and $Y \in R^{n \times q}$ such that Xa and Yb are maximally linearly correlated. We can formulate the problem

$$(a^*, b^*) = \underset{a,b}{\operatorname{argmax}}\, corr(Xa, Yb), \tag{7}$$

where

$$corr(Xa, Yb) = \frac{a^T \Sigma_{XY} b}{\sqrt{a^T \Sigma_{XX} a}\sqrt{b^T \Sigma_{YY} b}},$$

and where Σ_{XX} and Σ_{YY} are the covariance matrices of X and Y, respectively, and Σ_{XY} is the cross-covariance matrix. The choice of rescaling of a and b does not affect the $corr(Xa, Yb)$, so we can add the constraints $a^T \Sigma_{XX} a = 1$, $b^T \Sigma_{YY} b = 1$ to the maximization problem (7).

In Hotelling (1936), the CCA solution is based on two similar-looking equations

$$(\Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY} - \rho^2 \Sigma_{YY}) y = 0, \tag{8}$$

$$(\Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX} - \rho^2 \Sigma_{XX}) x = 0, \tag{9}$$

where Σ_{YX} is the transpose of Σ_{XY}. We can find canonical correlations and (a^*, b^*) by solving two standard eigenvalue problems. Basically, eigenvalues of (8) and (9) are the same and equal to the squared canonical correlations.

In addition to solving standard eigenvalue problem, there are some alternative ways to solve CCA. We can solve the generalized eigenvalue problem (Hardoon et al., 2004; Bach

and Jordan, 2002). We can find the solution of CCA by alternating least squares regression (Branco et al., 2005; Wilms and Croux, 2015). Alternatively, we can use singular value decomposition (SVD) to find the canonical correlations (Healy, 1957; Ewerbring et al., 1990). In the SVD method, we firstly find the singular value decompositions of X and Y as

$$X = U_X D_X V_X^T,$$
$$Y = U_Y D_Y V_Y^T,$$

where the columns of $U_X(U_Y)$ and the columns of $V_X(V_Y)$ are called the left and right singular vectors of $X(Y)$, respectively. Then, we can find the canonical correlations by finding the singular values of $U_X^T U_Y$. Overall, this method requires three singular value decompositions.

Another way to compute canonical correlations is to find the QR decomposition of X and Y and then apply SVD (Golub, 1969; Björck and Golub, 1973). The QR decomposition of X and Y are

$$X = Q_X R_X,$$
$$Y = Q_Y R_Y,$$

where $Q_X^T Q_X = I_p$, $Q_Y^T Q_Y = I_q$, and R_X and R_Y are upper triangular matrices. Following this, we can express Σ_X and Σ_Y as

$$\begin{aligned}
\Sigma_X &= X^T X \\
&= R_X^T Q_X^T Q_X R_X \\
&= R_X^T R_X, \\
\Sigma_Y &= Y^T Y \\
&= R_Y^T Q_Y^T Q_Y R_Y \\
&= R_Y^T R_Y,
\end{aligned}$$

where $\Sigma_X = R_X^T R_X$ and $\Sigma_Y = R_Y^T R_Y$ are the Cholesky decompositions. The singular values of $Q_X^T Q_Y$, which are derived from

$$\begin{aligned}(R_X^T)^{-1}\Sigma_{XY}R_Y^{-1} &= (R_X^T)^{-1}R_X^T Q_X^T Q_Y R_Y R_Y^{-1} \\ &= Q_X^T Q_Y,\end{aligned}$$

are the canonical correlations. This method requires two QR decompositions and one singular value decomposition. Since QR decomposition for a matrix requires less computational work than computing SVD (Do Q, 2012), we utilise QR decomposition to compute canonical correlations in the implementation of our methods.

Appendix C – Variable importance computation

The quantification of variable importance (VIMP) is important to assess relative importance of covariates. In the random forest framework, the mostly used VIMP idea proposed by Breiman (2001) is based on the increase in the prediction error when the link between the covariates and the response is broken by permuting out-of-bag (OOB) observations. In a regression setting with a set of covariates X and for a continuous response variable Y, VIMP for X_j can be computed as

$$VIMP(X_j) = \frac{1}{B}\sum_{b=1}^{B}\left(MSE(OOB_j^b) - MSE(OOB^b)\right),$$

where OOB^b is the OOB sample of the b^{th} tree of the forest, OOB_j^b is the OOB sample of the b^{th} tree where the j^{th} covariate is randomly permuted and MSE stands for the mean squared error. The average over B trees gives the variable importance measure for X_j. Larger VIMP shows greater importance.

`randomForestSRC` uses the same idea of breaking the link between covariates and the response but with another way of permuting X_j as proposed in Ishwaran (2007). Instead of permuting OOB samples at each tree, during the tree growing process, observations in the parent node are assigned to child nodes at random or consistently to the other child node when the split variable is X_j. VIMP for X_j can be computed as

$$VIMP(X_j) = \frac{1}{B}\sum_{b=1}^{B}\left(MSE(\tilde{t}^b, OOB^b) - MSE(t^b, OOB^b)\right), \tag{10}$$

where t^b is the original b^{th} tree of the forest, $\tilde{t}^b$ is the permuted b^{th} tree, and OOB^b is the OOB sample of the b^{th} tree.

Appendix D – Examples of sample distributions with DGP

Figures D1 and D2 show some examples for sample distributions with different parameter settings. See Section 1.3.1 for the explanation of DGP. Figures D1 and D2 correspond to the low and high correlated data settings with $n_{train} = 1000$ in the simulations for accuracy evaluation (Section 1.3.2), respectively. The left plot in the figures is the histogram of the generated sample. In the low correlated data setting (Figure D1), the mean and median correlations of the sample are 0.29 and 0.20, respectively. In the high correlated setting (Figure D2), the mean and median correlations of the sample are 0.61 and 0.57, respectively. The right plot in the figures is the ordered bar chart for the average of the coefficients within the X and Y sets over the sample. The selection of parameters s_x and s_y affects the coefficients of variables. In both low and high correlated settings, $s_x > s_y$ which result in faster decrease in generated X coefficients (a) compared to Y coefficients (b).

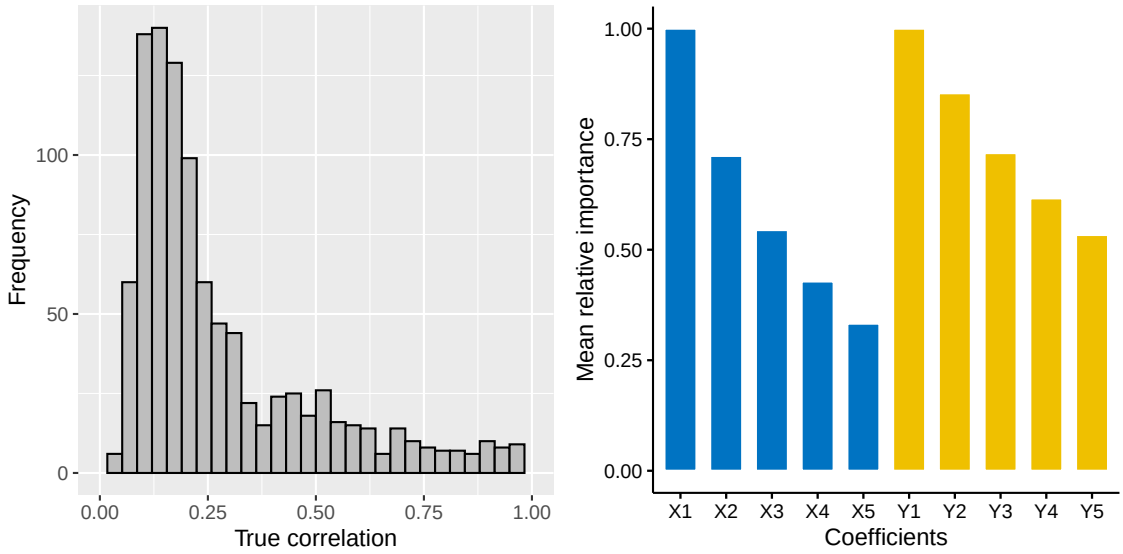

Figure D1: In this example, we have $5X, 5Y, 5Z$ variables and the sample size is 1000. The DGP parameters are $\beta_0 = -2$, $\rho_x = \rho_y = 0.3$, $\rho_z = 0.1$, $s_x = 0.7$, $s_y = 0.4$. The mean and median correlations are 0.29 and 0.20, respectively. This setting corresponds to the low correlated data set with $n_{train} = 1000$. *(left)* The sample distribution of correlations. *(right)* The ordered bar chart for the average of the coefficients within the X and Y sets over the sample.

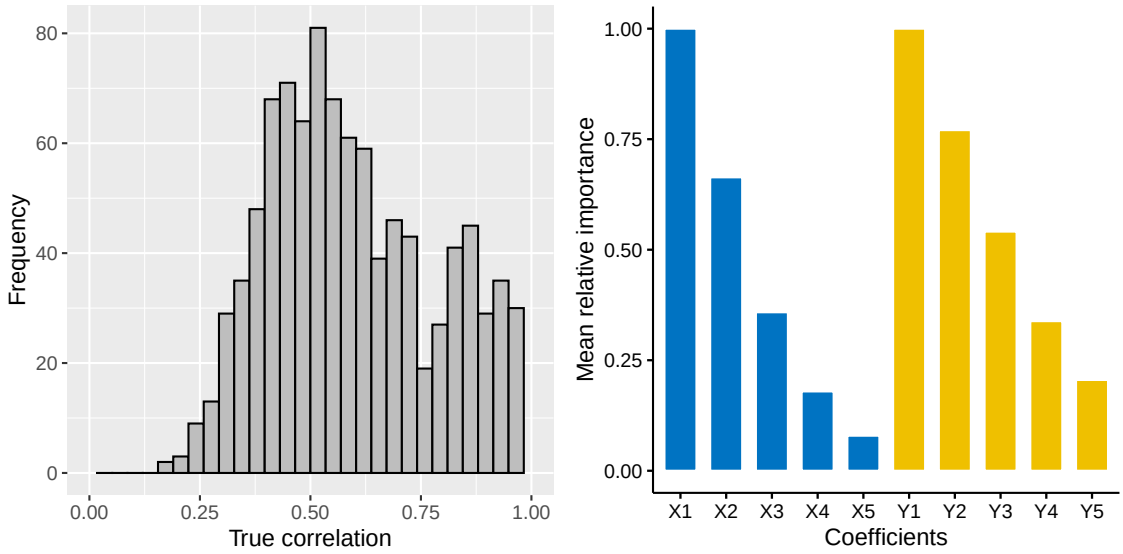

Figure D2: In this example, we have $5X, 5Y, 5Z$ variables and the sample size is 1000. The DGP parameters are $\beta_0 = -0.3$, $\rho_x = \rho_y = 0.3$, $\rho_z = 0.1$, $s_x = 0.4$, $s_y = 0.3$. The mean and median correlations are 0.61 and 0.57, respectively. This setting corresponds to the high correlated data set with $n_{train} = 1000$. *(left)* The sample distribution of correlations. *(right)* The ordered bar chart for the average of the coefficients within the X and Y sets over the sample.

Appendix E – Performance results for different values of parameter `nodesize`

Figures E1 and E2 present the average *MAE* over the 100 repetitions when `nodesize` = $\{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$ for the low and high correlation settings, respectively. In fact, top right plot in Figure E1, which shows the results of `nodesize` = $3\times(p+q)$, is the Figure 1.7 and is represented here to be able to compare with the results of other `nodesize` values. Similarly, top right plot in Figure E2 is the Figure 1.8. As can be seen in Figures E1 and E2, for the larger `nodesize` values *MAE* of the proposed method and the benchmark method are the same for scenarios with smaller n_{train}. For example, when `nodesize` = $6\times(p+q)$, *MAE* of the proposed method and the benchmark method are the same for $p = 10, q = 10$ when $n_{train} = \{100, 200, 300\}$. For this setting, `nodesize` $= 6\times(10+10) = 120$ which results in no splits for $n_{train} = \{100, 200\}$. For $n_{train} = 300$, a single split can occur in trees with this `nodesize`. However, the *BOP* of a new observation may include all of the training observations due to randomness. Each tree of the forest is a stump and there is a high chance that the union of the training observations that are in the same terminal nodes as the new observation is equal to the set of training observations. Estimating canonical correlation with this *BOP* is the same as computing CCA for all X and Y. When the sample size is small, increasing the `nodesize` may cause underfitting.

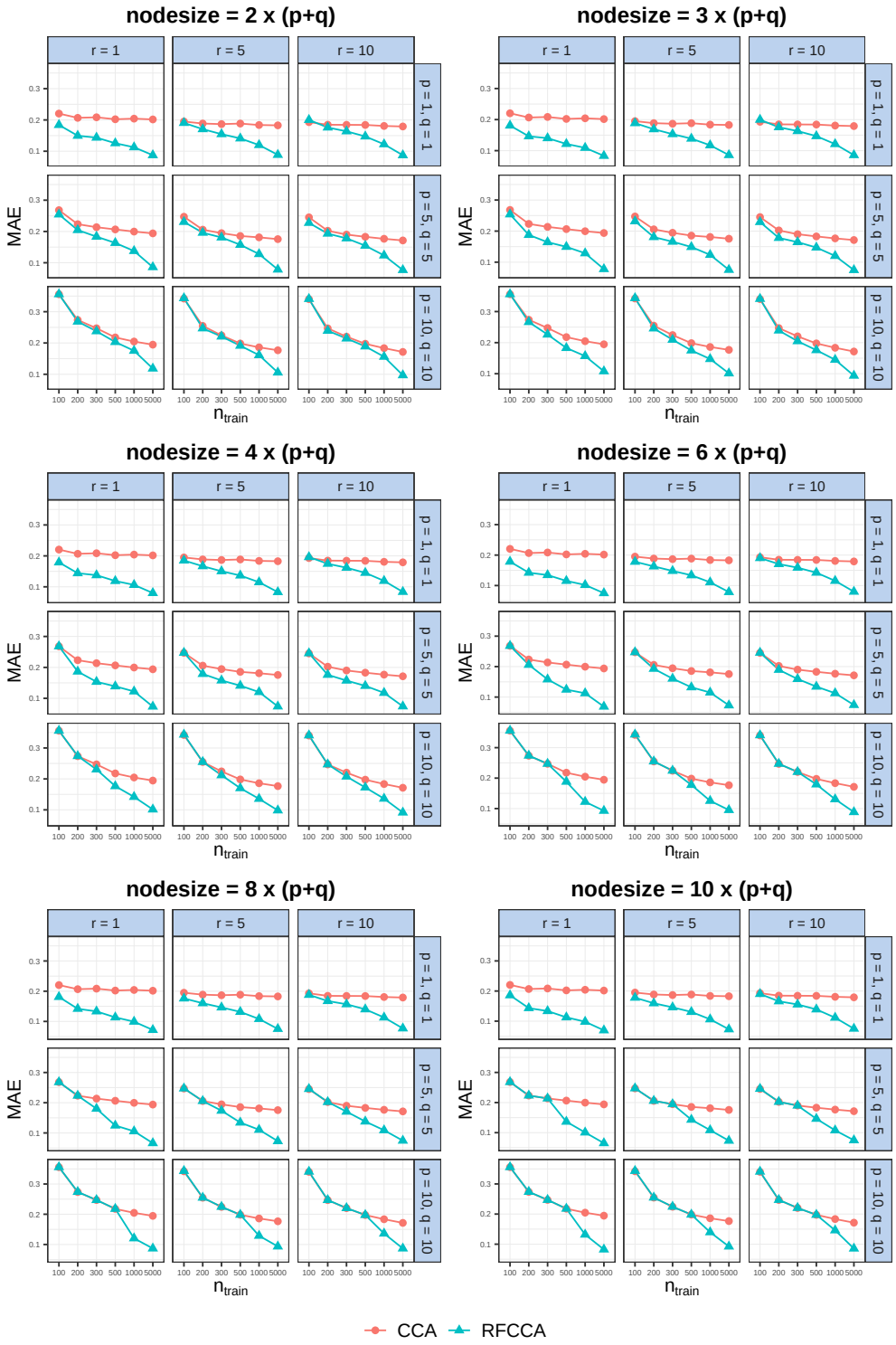

Figure E1: Accuracy evaluation results for low correlated data sets for six values of parameter `nodesize` = $\{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

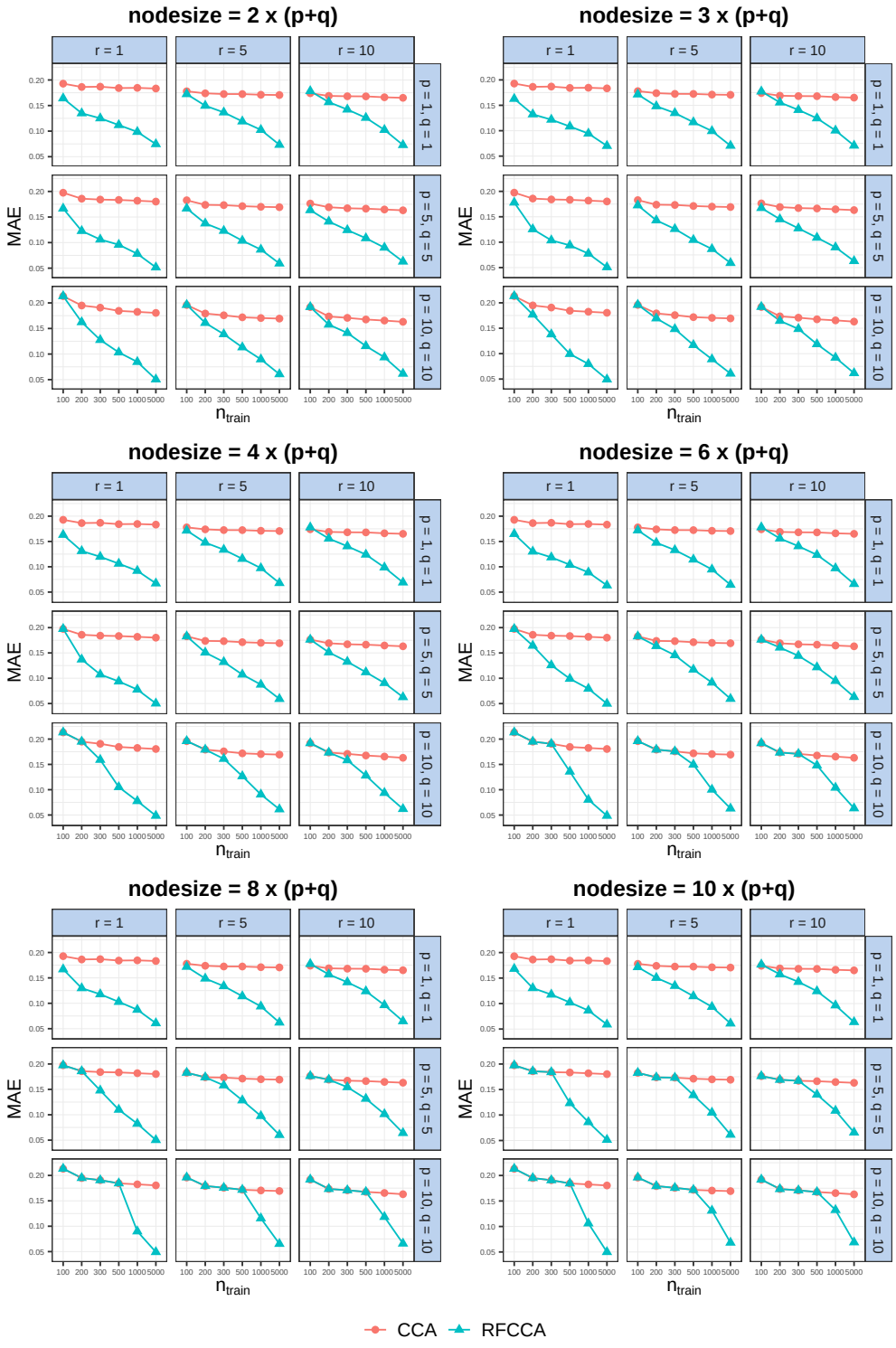

Figure E2: Accuracy evaluation results for high correlated data sets for six values of parameter `nodesize` = $\{2\times(p+q), 3\times(p+q), 4\times(p+q), 6\times(p+q), 8\times(p+q), 10\times(p+q)\}$. $r^{noise} = 5$ in all settings. CCA is the benchmark method. Smaller values of MAE are better.

Appendix F – EEG data

In this section, the data collection process and preprocessing steps are explained.

Neuropsychological scale

To evaluate the intellectual disabilities, performance IQ (pIQ) and verbal IQ (vIQ), which are the scores for total IQ, were obtained using the Mullen, the WPPSI-IV, the WISC-V, and the WAIS-IV batteries depending on the age of the participant. The administration was adapted for the clinical participants by starting at the first item rather than the starting point for their chronological age, frequent breaks were proposed, and participants were motivated through many creative incentives if necessary (songs, games, conversations about their interests, etc.).

EEG auditory task

The auditory task was prepared using E-prime 2.0 software (Psychology Software Tools Inc., Pittsburgh, PA, USA) on a screen placed at a viewing distance of 60 cm. Sounds, which are presented binaurally and simultaneously, were delivered through two speakers located laterally at 30 cm from the participants' ears. The auditory stimulus consisted of 24 dB/octave white noise burst. Each stimulus lasted 50 ms with an inter-stimulus interval varying between 1200 and 1400 ms to avoid a process of habituation. The task was composed of 150 trials. The total task duration was around 4 minutes. To assure maximal collaboration, a movie without sound and subtitles was presented. The participant was told to focus his/her attention on the movie and not to give attention to the auditory stimuli.

EEG recordings

The subject was placed in an electrically shielded room in the Sainte Justine's Hospital. The continuous EEG was recorded with a high-density EEG system containing 128 electrodes placed according to the extended 10 - 20 system (Electrical Geodesics System Inc., Eugene, OR, USA). Signals were acquired and processed by a G4 Macintosh computer using NetStation EEG Software (Version 2.0). Before recording, impedances were verified and were kept below 40 kΩ (Tucker, 1993). EEG data were amplified, band-pass-filtered 0.1–4000 Hz, and sampled at 1000 Hz with a vertex reference.

Preprocessing for the analysis

Off-line signal processing and ERP analyses were performed using the EEGLAB toolbox via custom Matlab scripts (Delorme and Makeig, 2004). EEG acquired data was subjected to the following pre-processing steps. First, EEG signals were digitally filtered with a high-band pass filter (0.5 Hz) and a 60 Hz notch filter. Twenty-eight electrodes placed on the neck and the face and containing muscular artifacts have been removed to avoid contamination of average reference. Moreover, a voltage threshold method (2–200 μV) was applied to exclude channels with artifacts. Data were off-line re-referenced to the mean of the EEG selected electrodes (100 channels). Independent component analysis (ICA) as implemented in the EEGLAB toolbox (with default parameters) was used to remove ocular artifacts. By removing or minimizing the effects of overlapping components, ICA enables a detailed examination of the separate dynamics of electrical brain activity as well as artifacts to remove them (Delorme et al., 2007). Ocular and cardiac ICA components (range across subjects: 1–3 components) were identified by visual inspection and deleted from the global signal. Continuous EEG was segmented into epochs covering a time window from -200 ms to 800 ms relative to the onset of the tone. Also, as the artifact ICA components could be deemed unsatisfactory, the segmented EEG recordings were visually inspected by a well-trained experimenter, and trials presenting with residual artifacts were rejected.

For time-frequency (TF) and inter-trial coherence (ITC) analyses, segments were exported to MATLAB (version R20174b) (The MathWorks Inc., Natik, MA, USA) after artifact rejection. TF and ITC analyses were processed using the EEGLAB toolbox (v.13.6.5b) (La Jolla, CA, USA). TF analysis allows us to explore different frequency bands in terms of their power and temporal distributions (Herrmann et al., 2005). We used complex Morlet's wavelet transformation (Tallon-Baudry and Bertrand, 1999) to provide power maps in time and frequency domains. The simplified mathematical expression for applying this specific wavelet convolution on our EEG signal is as follows:

$$M(t,f) = \int_t W\Big(\frac{t-a}{b}, f\Big) S(t)dt,$$

where $M(t,f)$ is a matrix of complex values (vectors) for a given time (t) and frequency (f), S is the signal as a function of time (t) and W corresponds to Morlet's wavelet which is a complex exponential (Fourier) with a Gaussian envelope that undergoes a series of translations (a) and dilations (b) dependently of the frequency (f). The event-related spectral perturbation (ERSP) computation uses the complex values (amplitude and phase) given by Morlet's wavelet transform as shown in the following formula calculating the power spectrum for each time and frequency point:

$$P(t,f) = 10\log_{10}|M(t,f)|^2,$$

where $P(t,f)$ denotes TF power in terms of decibels (dB). Final TF maps were computed as follows:

$$TF = \frac{1}{N}\sum_{n=1}^{N} P(t,f),$$

where N is the total number of trials. The range of frequency investigated was from 3 to 100 Hz. ITC, analogous to phase-locking value (PLV), allows the assessment of the strength of phase coherence across trials in temporal and spectral domains (Makeig et al., 2004). The ITC computation uses only the phase of the complex values given by Morlet's wavelet transform. ITC was computed as in Lachaux et al. (1999) to extract PLV. ITC measures phase coupling across trials at all latencies and frequencies and is defined by:

$$ITC = \frac{1}{N}\Big|\sum_{n=1}^{N} exp(j\theta(f,t,n))\Big|,$$

where θ represents the phase for a given frequency (f), time point (t), and trial (n). The obtained values are always defined between 0 and 1. Phase-locking values close to 1 indicate strong inter-trial phase-locking, thus representing evoked activity while scores closer to 0 indicate a high inter-trial phase variability, thus representing induced activity (Lachaux et al., 1999).

Appendix G – Data preprocessing

In this section, we present the steps to prepare Ames Housing data set for the analysis. We use the processed version of the data set from the `AmesHousing` package.

```
library("AmesHousing")
AmesHousing <- make_ordinal_ames()

## Data preprocessing
# remove observations with missing values
AmesHousing <- AmesHousing[complete.cases(AmesHousing), ]

# convert the response variable in thousands
AmesHousing$Sale_Price <- AmesHousing$Sale_Price/1000

# convert the ordered factors to numeric to preserve the ordering of
# the factors
ord_vars <- vapply(AmesHousing, is.ordered, logical(1))
nam_ord <- names(ord_vars)[ord_vars]
AmesHousing[, nam_ord] <- data.frame(lapply(AmesHousing[, nam_ord],
                                            as.numeric))

# group together levels with less than 30 observations,
# we use the combineLevels() function from "rockchalk" package
```

```
# for this step
library("rockchalk")
fac_vars <- vapply(AmesHousing, is.factor, logical(1))
AmesHousing[, fac_vars] <- data.frame(
  lapply(AmesHousing[, fac_vars],
         function(x, nmin) combineLevels(x,
                           levs = names(table(x))[table(x)<nmin],
                           newLabel=c("combinedLevels")),
         nmin=30)
)
```

Appendix H – `nodesize` tuning

Figures H1 and H2 present the accuracy results for different levels of `nodesize` along with the proposed method which applies a `nodesize` tuning as described in Section 3.2.2 of the main paper. In the figures, the red boxplots illustrate the MAE results for the proposed `nodesize` tuning heuristic, and the remaining boxplots show the accuracy obtained when we set the `nodesize` to a specific value. For the set of `nodesize` levels to be searched in the proposed method, we have `nodesize`$= \{[(2^{-1}, 2^{-2}, 2^{-3}, \ldots)s] > q\}$ where q is the number of outcomes and s is the sub-sample size computed as $s = 0.632n_{train}$. As can be seen from the results in Figure H1, as `nodesize` decreases, first MAE^{cor} and MAE^{sd} decrease and after a point increase for both DGP1 and DGP2. Since we have more levels of `nodesize` in the larger sample scenarios, it is easier to see this behaviour. For these two DGPs, smaller `nodesize` values do not mean better performance. As can be seen from Figure H2, for DGP3 and DGP4, contrary to results of DGP1 and DGP2, MAE^{cor} and MAE^{sd} decrease as the `nodesize` increases. Hence, the best performing `nodesize` is mostly the smallest. Overall, when we compare the accuracy of the proposed `nodesize` tuning heuristic to the individual results of different `nodesize` values, we can see that it mostly performs well, especially for the larger sample sizes.

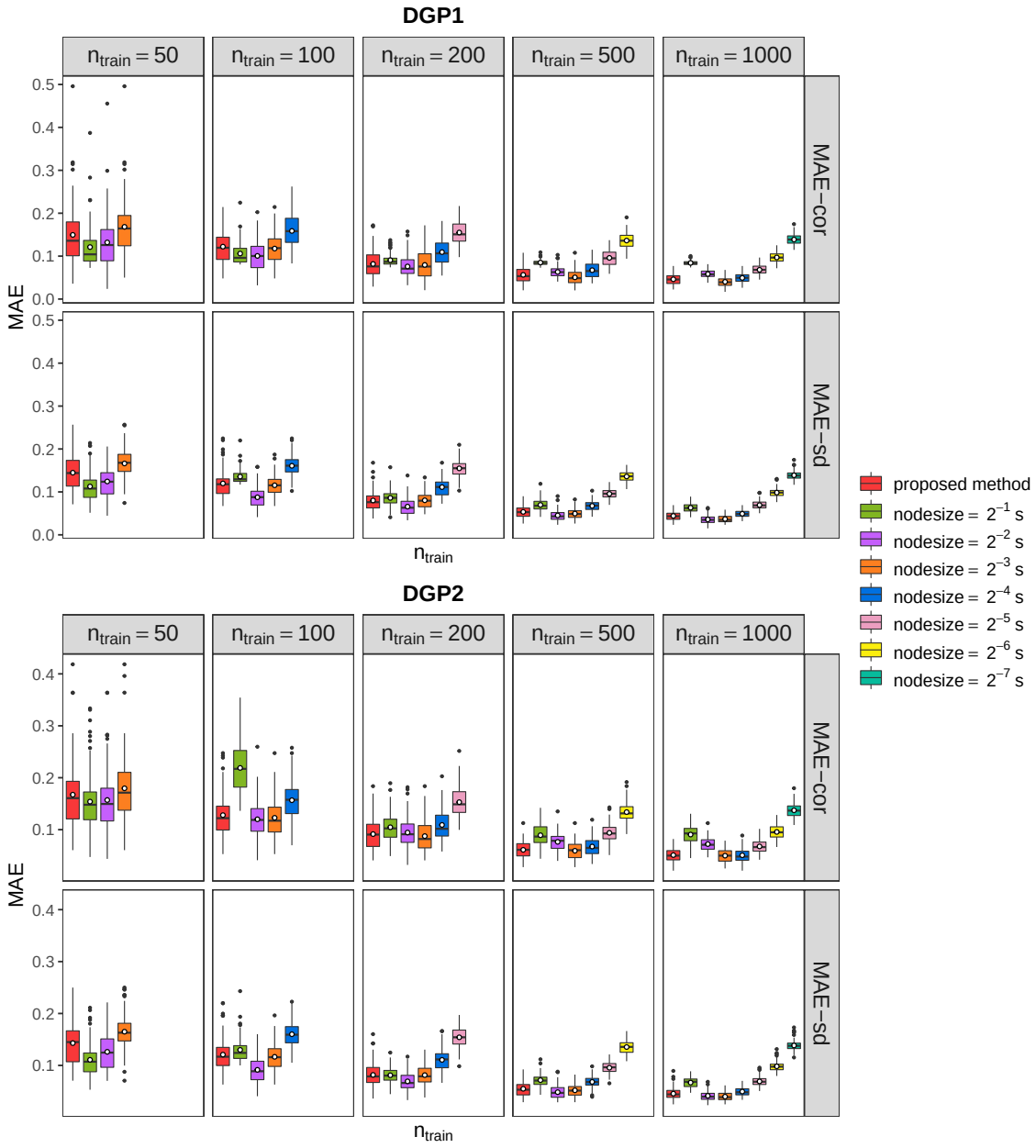

Figure H1: MAE results for different `nodesize` values for DGP1 and DGP2. Smaller values of MAE^{cor} and MAE^{sd} are better. s is the sub-sample size, *i.e.* $s = .632n_{train}$. Red boxplots illustrate the accuracy for the proposed `nodesize` tuning, and the rest of the boxplots show the accuracy obtained when we set the `nodesize` to a specific value.

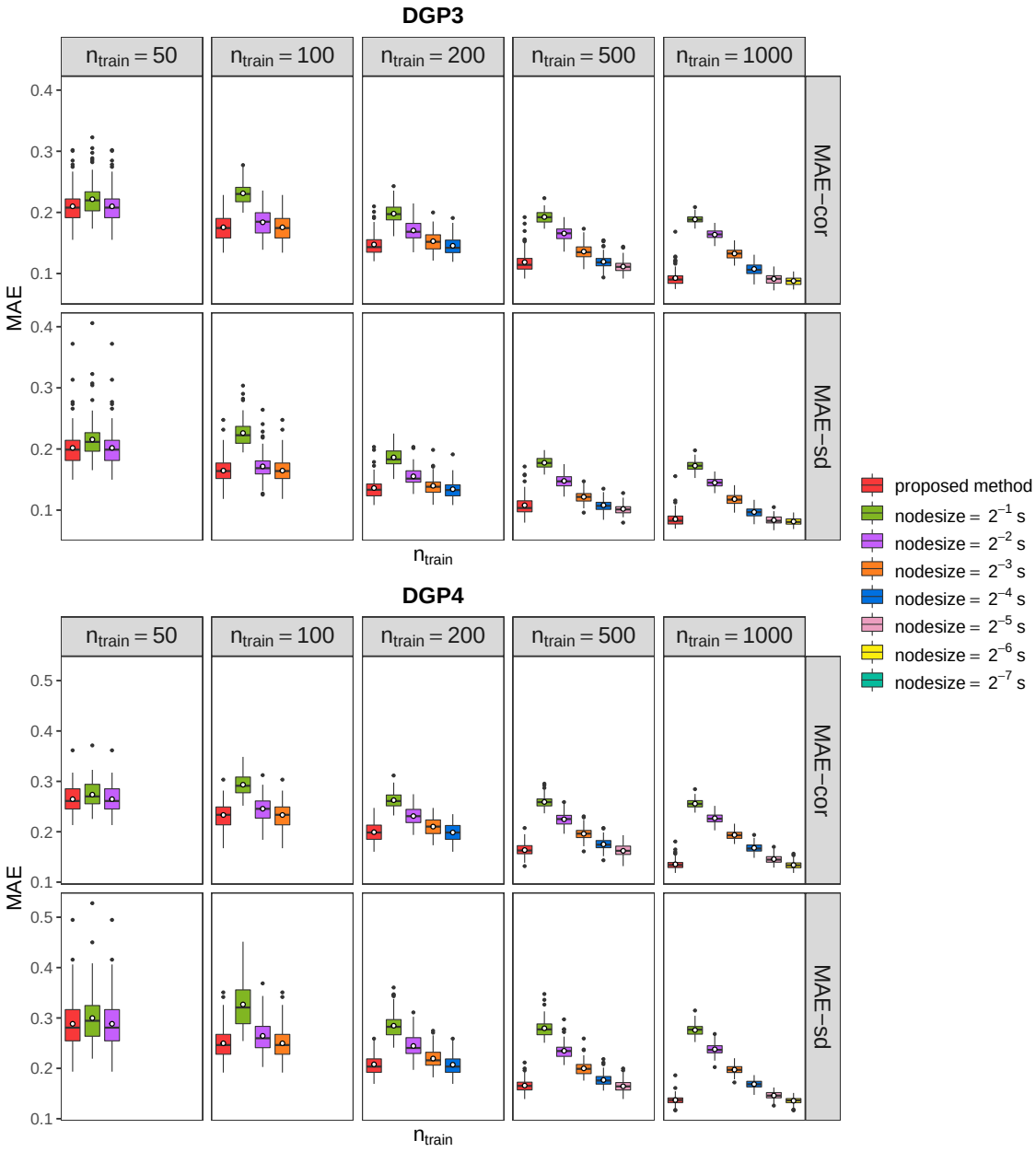

Figure H2: MAE results for different `nodesize` values for DGP3 and DGP4. Smaller values of MAE^{cor} and MAE^{sd} are better. s is the sub-sample size, *i.e.* $s = .632n_{train}$. Red boxplots illustrate the accuracy for the proposed `nodesize` tuning, and the rest of the boxplots show the accuracy obtained when we set the `nodesize` to a specific value.

Appendix I – Global significance test

The proposed global significance test is described in Algorithm I1. After computing the unconditional and conditional covariance matrices, Σ_{root} and $\Sigma_{\mathbf{x}_i}$, respectively, we compute the global test statistic with

$$T = \frac{1}{n}\sum_{i=1}^{n} d\left(\hat{\Sigma}_{\mathbf{x}_i}, \Sigma_{root}\right), \tag{11}$$

where $d(.,.)$ is computed as (3.2) in the main paper.

Algorithm I1 Global permutation test for covariates' effects

1: Compute sample covariance matrix of $\mathbf{Y}$ in the root node, say Σ_{root}
2: Train a RF with $\mathbf{X}$ and $\mathbf{Y}$
3: Estimate covariance matrices as described in Algorithm 3.1 of the main paper, say $\hat{\Sigma}_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
4: Compute test statistic T as in (11)
5: **for** $r = 1 : R$ **do**
6: Permute rows of $\mathbf{X}$ to obtain $\mathbf{X}_r$
7: Train a RF with $\mathbf{X}_r$ and $\mathbf{Y}$
8: Estimate covariance matrices as described in Algorithm 3.1 of the main paper, say $\hat{\Sigma}'_{\mathbf{x}_i}\ \forall i = \{1,\ldots,n\}$
9: Compute test statistic with $T'_r = \frac{1}{n}\sum_{i=1}^{n} d\left(\hat{\Sigma}'_{\mathbf{x}_i}, \Sigma_{root}\right)$
10: **end for**
11: Approximate the permutation p-value with $p = \frac{1}{R}\sum_{r=1}^{R} I(T'_r > T)$
12: Reject the null hypothesis at level α when $p < \alpha$. Otherwise, do not reject the null hypothesis.

Appendix J – Data generating process

In DGP1, the covariance matrix for the observation x_i is

$$\Sigma_{\mathbf{x_i}} = \Psi + \mathbf{B}\mathbf{x}_i\mathbf{x}_i^T\mathbf{B}^T,$$

where $\mathbf{x}_i^T = (1, x_i)^T$, $\mathbf{B}_0 = [(1,-1)^T, (1,1)^T]$, $\mathbf{B} = \frac{w}{w+1}\mathbf{B}_0$, $\Psi_0 = \mathbf{B}_0[(1,0)^T, (0,1/3)^T]\mathbf{B}_0^T$, $\Psi = \frac{1}{w+1}\Psi_0$ and $w = 1$.

In DGP3, the correlations are generated with all seven covariates according to a tree model with a depth of three and eight terminal nodes:

$$\begin{aligned}
\rho(\mathbf{x}_i) = {} & u_1 I(x_{i1} < 0, x_{i2} < 0, x_{i4} < 0) \\
& + u_2 I(x_{i1} < 0, x_{i2} < 0, x_{i4} \geq 0) \\
& + u_3 I(x_{i1} < 0, x_{i2} \geq 0, x_{i5} < 0) \\
& + u_4 I(x_{i1} < 0, x_{i2} \geq 0, x_{i5} \geq 0) \\
& + u_5 I(x_{i1} \geq 0, x_{i3} < 0, x_{i6} < 0) \\
& + u_6 I(x_{i1} \geq 0, x_{i3} < 0, x_{i6} \geq 0) \\
& + u_7 I(x_{i1} \geq 0, x_{i3} \geq 0, x_{i7} < 0) \\
& + u_8 I(x_{i1} \geq 0, x_{i3} \geq 0, x_{i7} \geq 0),
\end{aligned}$$

where the terminal node values are $u = (0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)$ and I is the indicator function. The variances are functions of ρ and computed as $Var(y_j|\mathbf{x}_i) = (1 + \rho(\mathbf{x}_i))^j$, $j = \{1, \ldots, q\}$.

In DGP4, for an observation $\mathbf{x}_i$, we can generate the correlation with the logit model,

$$\rho(\mathbf{x}_i) = \frac{1}{1 + \exp\left(-(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + x_{i1}^2)\right)},$$

where β_0 is the intercept parameter fixed to $\beta_0 = -1$ and β_j are the weights for the covariates, fixed to $(1, 1-\frac{1}{p}, 1-\frac{2}{p}, \ldots, 1-\frac{(p-1)}{p})$. For an observation $\mathbf{x}_i$, the variance of each response is generated as $Var(y_j|\mathbf{x}_i) = (1+\rho(\mathbf{x}_i))^j$, $j = \{1, \ldots, q\}$.

Appendix K – Comparison of computational times

All simulations were run in R version 3.6.0 on a Linux machine with Intel(R) Xeon(R) E5-2667 v3 @ 3.20GHz with 396 GB of memory. The average computational time of each method for the four DGPs is presented in Table K1. For both methods, the time for a setting consists of the time for training and the time for prediction for a new data set. We can see that the proposed method is significantly faster than `covreg`.

Table K1: Average computational time (in seconds) of both methods over 100 replications for each simulated data set.

n_{train}	DGP	`CovRegRF`	`covreg`
50	DGP1	2.89	148.23
	DGP2	2.85	149.12
	DGP3	2.60	304.43
	DGP4	2.46	248.27
100	DGP1	4.01	151.55
	DGP2	4.01	151.97
	DGP3	3.74	283.25
	DGP4	3.44	247.77
200	DGP1	6.46	228.57
	DGP2	6.67	229.55
	DGP3	6.48	428.89
	DGP4	5.82	495.07
500	DGP1	15.07	383.16
	DGP2	14.77	384.52
	DGP3	15.38	593.00
	DGP4	13.49	744.41
1000	DGP1	52.64	771.69
	DGP2	52.34	739.52
	DGP3	62.96	984.28
	DGP4	53.95	1318.28