# Tree-Based Ensembles for Multi-Output Regression: Comparing multivariate approaches with separate univariate ones

Lena Schmid[a,*], Alexander Gerharz[a], Andreas Groll[a], Markus Pauly[a]

[a]*Department of Statistics, TU Dortmund University, 44227 Dortmund, Germany*

**Abstract**

Tree-based ensembles such as the Random Forest are modern classics among statistical learning methods. In particular, they are used for predicting univariate responses. In case of multiple outputs the question arises whether it is better to separately fit univariate models or directly follow a multivariate approach. For the latter, several possibilities exist that are, e.g. based on modified splitting or stopping rules for multi-output regression. These methods are compared in extensive simulations and a real data example to help in answering the primary question when to use multivariate ensemble techniques instead of univariate ones.

*Keywords:* Machine Learning, Multi-Output Regression, Multivariate Trees

## 1. Introduction

Multivariate data occur in a variety of disciplines, for example in biomedical research, the social sciences, or econometrics. Data are said to be multivariate if the response not only consists of one variable, but of $d \geq 2$ output variables, say $\mathbf{Y} \in \mathbb{R}^d$. Then, we are often interested in finding a functional relationship between the output $\mathbf{Y}$ and some feature variables $\mathbf{X} \in \mathbb{R}^p$, i.e. we want to perform a multivariate (also called multi-output) regression analysis. Unlike univariate multiple regression (with $d = 1$), which also includes multiple features $\mathbf{X} \in \mathbb{R}^p$, multivariate regression wants to specify the relationship of several outcome variables with $\mathbf{X}$ simultaneously. The hope of such multivariate analyses is, that the consideration of possible dependencies between the outcomes may lead to procedures with better power (in case of inference) or accuracy (in case of prediction) compared to separate univariate analyses. While the need for the development and use of valid and distributional robust or nonparametric multivariate methods has been recognized and addressed in inferential statistic (Dobler et al., 2020; Friedrich et al., 2019; Konietschke et al., 2015; Smaga, 2017; Vallejo and Ato, 2012; Zimmermann et al., 2020; Mavridis and Salanti, 2013), there do not exist exhausting studies that exploit the potential of multivariate regression methods for prediction.

Focussing on tree-based ensemble methods as the Random Forest, it is the aim of this paper to close this gap. In particular, we want to answer our research-motivating question:

*When should a holistic multivariate regression approach be preferred over separate univariate predictions?*

*Corresponding Author
*Email address:* lena.schmid@tu-dortmund.de (Lena Schmid)

The answer to this question is not clear as of now. In fact, univariate tree-based ensembles as the Random Forest (Breiman, 2001) or Extra Trees (Geurts et al., 2006) have been shown to be good predictive tools in various applications (Pauly, 2012; Gerke et al., 2018; Schauberger and Groll, 2018; Groll et al., 2019; Huang et al., 2020) and are applied far more often than multivariate regression approaches. One reason for this could be that there are still only a few multivariate extensions, such as (De'ath, 2002; Larsen and Speckman, 2004; D'Ambrosio et al., 2017; Zhang, 1998; Zhang and Ye, 2008), where the impurity measure used in the tree construction was multivariatly extended. Zhang (1998) developed an impurity function based on a generalized entropy criterion to handle multiple binary outputs, this work was extended by Zhang and Ye (2008) to ordinal outputs by transforming the outputs to binary-valued indicator functions. Siciliano and Mola (2000) used the weighted sum of Gini index reduction to construct trees for multiple binary outputs.

One of the first approaches for multi-output regression trees was proposed by De'ath (2002), which extended CART to multi-output regression by using the sum of squared errors over the multivariate outputs as the impurity function of a node (De'ath (2002) also developed more general forms of distance-based impurity functions). This method is implemented in the R-package `MVPART` (De'ath, 2012) and was used by Segal and Xiao (2011) to develop multivariate Random Forests. They have also adapted the method for an application to identify yeast transcriptional regulation networks Xiao and Segal (2009). Larsen and Speckman (2004) suggested a multivariate regression tree that uses the Mahalanobis distance as node impurity function where the covariance matrix is estimated from the whole data set. The Mahalanobis distance is also used in the R-packages `MultivariateRanfomForest` and `IntegratedMRF` (Rahman et al., 2017) to construct multivariate Random Forests. Furthermore, Rahman et al. (2017) have observed in several drug response predictions that multivariate Random Forests provide higher accuracy than Random Forests when outputs are highly correlated. The Eart Mower and Mallows distances were used as impurity functions by D'Ambrosio et al. (2017) to develop regression trees for multivalued numerical outputs. Cariou (2006) extended regression trees to interval data by defining a new impurity measure which is based on the Hausdorff distance and applied these trees to electricity load profiling. For functional data an extension of multivariate regression trees was presented by Yu and Lambert (1999). Kocev et al. (2009) have adapted the method to model a compound index of vegetation condition and compared it to single regression trees. For their setting, they have observed that multivariate models have a smaller size, are faster to learn while there is no statistically significant difference in their predictive power.

There are also several approaches that extended the GUIDE algorithm (Loh, 2002) to multivariate and longitudinal outputs (Loh and Zheng, 2013; Hsiao and Shih, 2007). Lee (2005) developed a method that can be used with multivariate outputs of any type. Here, GEE techniques are used to find the splits. Multivariate trees for a mixture of categorical and continuous outputs were presented by Dine et al. (2009), where the splits are derived from a likelihood-based approach for a general location model. For Extra Trees, there is also an extension for multi-task learning (Simm et al., 2014), where the split criterion has been adapted to handle multiple tasks. The method is implemented in the R-package `extraTrees` (Simm et al., 2014). Kocev and Ceci (2015) proposed an extension of Extra Trees for multioutput regression based on the predicitve clustering trees framework.

To answer the central research question, we compare the predictive accuracy of separate univariate analyses with a simultaneous multivariate analysis by means of exhaustive simulations and in an illustrative data analysis.

This work is structured as follows: Section 2 presents the univariate and multivariate ensemble methods. More precisely, univariate and multivariate Random Forest and Extra Trees algorithms are presented. In addition, the multi-task Extra Trees algorithm Simm et al. (2014) is described in more detail. The simulation design and framework are then presented in Section 3, while Section 4 summarizes the main simulation results. In Section 5, illustrative real-world data examples are shown before the manuscript concludes with a discussion of our findings and an outlook for future research (Section 6).

## 2. Methods

In this section, we explain the univariate and multivariate tree-based methods under investigation. We thereby distinguish between three different approaches: (i) univariate ensembles such as Random Forest and Extra Trees, (ii) their multivariate counterpart and (iii) multi-task Extra Trees.

### *2.1. Univariate Tree Ensemble Learner*

To explain the univariate methods, assume that one has access to a training data set $\mathcal{D}_n := \{(\mathbf{x}_i^\top, y_i)^\top \in \mathbb{R}^{p+1} : i = 1, ..., n\}$ consisting of realisations of random vectors with real valued metric outcome $Y$ and $p$-dimensional metric feature vector $\mathbf{X}$.

*Random Forest.* A Random Forest is a univariate machine learning method based on building ensembles of decision trees. It was developed to address predictive shortcomings of traditional Classification and Regression Trees (CARTs) (Breiman et al., 2017). Random Forests consist of a large number of weak decision tree learners, which are grown in parallel to reduce the bias and variance of the model at the same time (Breiman, 2001). For training a Random Forest, $N$ bootstrap samples are drawn from the training data set. Each bootstrap sample is then used to grow an unpruned tree. Instead of using all available features in this step, only a small and fixed number of randomly sampled $m_{try}$ features are selected as split candidates. In the regression case, Breiman (2001) proposed to use $m_{try} = \lfloor p/3 \rfloor$, which is still the default choice in many software implementations. A split is chosen by the CART-split criterion for regression, i.e. by minimizing the sum of squared errors in both child nodes. Instead of the CART-split criterion, many other distances such as the least absolute deviations of the mean (L1-norm) can also be used. These steps are then repeated until $B$ such trees are grown, and new data is predicted by taking the mean of all $B$ tree predictions. The important hyperparameters for the Random Forest are:

- $B$ as the number of grown trees. Note that this parameter is usually not tuned since it is known that more trees are better.
- The cardinality of the sample of features at every node is $m_{try}$.
- The number of observations in each bootstrap sample for each grown tree is $N$, which is usually the number of observations in the training data set.
- The minimum number of observations that each terminal node should contain (stopping criteria).

*Extra Trees.* Extra Trees (also called Extremely Randomized Trees) is an ensemble method developed by Geurts et al. (2006) that aims at obtaining trees that are more decorrelated than in the Random Forest approach. Similar to Random Forest, Extra Trees consist of $B$ trees, which are also grown in parallel. However, in contrast to Random Forest, the whole training data set is used for constructing a tree. For the determination of the next split a random sample of features of size $m_{try}$ is selected. Instead of computing the locally optimal cut-point for each feature in the random sample based on the CART-split criterion, random cut-points are selected. Then, of all the randomly generated splits, the split that optimizes the split criterion is chosen to split the node. The idea of this extra level of randomness is to improve the process of decorrelation leading to smaller variance and a potentially better predictive accuracy in some situations. The important hyperparameters for the Extra Trees are:

- $B$ as the number of grown trees.
- The cardinality of the sample of features at every node is $m_{try}$.
- The number of random split values at each node.
- The minimum number of observations that each terminal node should contain (stopping criteria).

### *2.2. Multivariate Tree Ensemble Learner*

For the multivariate case, we similarly assume that one has access to a training data set $\mathcal{D}_n := \{(\mathbf{x}_i^\top, \mathbf{y}_i^\top)^\top \in \mathbb{R}^{p+d} : i = 1, ..., n\}$ which now consists of realisations of random vectors with $d$-dimensional metric outcome $\mathbf{Y}$ and $p$-dimensional $\mathbf{X}$ feature vector. To obtain the multivariate versions of Random Forests and Extra Trees, we need to modify the splitting criterion. Here, a natural extension of the CART splitting criteria to multiple metric outputs is to work with the multivariate L2-distance to measure the impurity of a node (Segal and Xiao, 2011). Hence, the impurity function $i$ at node $t$ is given by

$$i_{L2}(t) = \sum_{\mathbf{y}_j \in t} (\mathbf{y}_j - \overline{\mathbf{y}}(t))^\top (\mathbf{y}_j - \overline{\mathbf{y}}(t)),$$

where $\overline{\mathbf{y}}(t)$ is the sample mean of the output vector at node $t$, see (De'ath, 2002) for details. Note that there are many other ways to adapt the splitting criterion, for example the multivariate L1-norm can be used as impurity function at node $t$, which is defined by

$$i_{L1}(t) = \sum_{j:\mathbf{y}_j \in t} \sum_{i=1}^{d} \left| y_j^i - \tilde{y}(t)^i \right|,$$

where $\mathbf{y}_j = (y_j^1, \ldots, y_j^d)$ and $\tilde{\mathbf{y}}(t) = (\tilde{y}(t)^1, \ldots \tilde{y}(t)^d)$ is the median of the output vector at node $t$. Thus, to construct a multivariate Random Forest, we simply proceed as in the univariate case and only change the splitting criterion to the multivariate split extension. The same holds for a multivariate Extra Tree, where for each split, the multivariate impurity function is used to select the optimal split among all randomly chosen splits.

*2.3. Multi-Task Extra Trees*

Another extension of the Extra Trees algorithm was proposed by Simm et al. (2014) in order to handle multi-task learning. For multi-task learning, suppose we have given $T$ supervised learning tasks and all data for the tasks come from the same space $\mathbf{X} \times Y$, where $\mathbf{X} \subset \mathbb{R}^p$ and $Y \subset \mathbb{R}$. Moreover, for each task $t$ we have access to a training data set

$$\mathcal{D}_{t,n} := \left\{ (\mathbf{x}_{i,t}^\top, y_{i,t})^\top \in \mathbb{R}^{p+1} : i = 1, \ldots, n \right\},$$

which is sampled from a distribution $P_t$ on $\mathbf{X} \times Y$. We assume that the $P_t$ are different for each task, but related. Simm et al. (2014) modified the split-criterion for Extra Trees in such a way that samples can now also be additionally split according to their tasks. These new splits will then create two child nodes and each node contains samples corresponding to separated task subsets. More precisely, when optimizing the split criterion for each node, not only the random feature splits are considered but also a random task split, which is determined as follows: In the regression case, for each task $t$, the task feature $f_t$ are computed by

$$f_t = \frac{\sum_{v \in I_t} y_v + \alpha \frac{1}{|I|} \sum_{w \in I} y_w}{|I_t| + \alpha},$$

where $I$ denotes the set of sample indexes at the current node, $I_t$ the set of sample indexes of the task $t$ at the current node and $\alpha$ is a regularization parameter weighting the influence of the tasks with default value 1. Then, the task cut point is randomly selected from the intervall $(\min_t f_t, \max_t f_t)$. This modification is implemented in the R-package `extraTrees` (Simm et al., 2014).

To predict multivariate outputs with the multi-task Extra Tree algorithm, the data has to be transformed as follows: since every component of the multivariate output is considered as a task, the training sample of a multivariate regression problem

$$\left\{ (\mathbf{x}_i^\top, \mathbf{y}_i^\top)^\top \in \mathbb{R}^{p+d} : i = 1, \ldots, n \right\},$$

is transformed to

$$\left\{ (\mathbf{x}_i^\top, y_i^j) \in \mathbb{R}^{p+1} : i = 1, \ldots, n,\ j = 1, \ldots, d \right\},$$

where $\mathbf{y}_i = (y_i^1, \ldots, y_i^d)^\top$. The multi-task implementation also requires the following task vector

$$\mathbf{task} = (1, \ldots, 1, \ldots, d, \ldots, d)^\top \in \mathbb{R}^{nd},$$

indicating to which component/task the output value $y_i^j$ belongs. We note, however, that this multivariate extension is only feasible as long as the outputs are consumerate, i.e. measured on the same scale.

*2.4. Some Theoretical Preliminaries*

After introducing the models, we might already have some ideas regarding the question when to use a multivariate approach instead of a univariate one. Consider again the setting from Subsection 2.2, where

the data generating process (DGP) for the outcomes $\mathbf{Y} = (Y_1, \ldots, Y_d)^\top \in \mathbb{R}^d$ may be given by

$$\mathbf{Y} = \mathbf{f}(\mathbf{X}) + \boldsymbol{\varepsilon}.$$

Here, $\mathbf{X} \in \mathbb{R}^p$ is the feature generating vector, $\mathbf{f} = (f_1, \ldots, f_p)^\top$ some unknown functions and $\boldsymbol{\varepsilon}$ denotes the multivariate error variable. Heuristically, we would expect multivariate approaches to show advantages when there are moderate to strong dependencies between the outcomes $Y_1, \ldots Y_p$, which cannot be adequately explained by the covariates alone. This could, e.g., be the case if there are just a few covariates or a few predictive covariates. This effect should be particularly pronounced for similar functions $f_i$. Otherwise, univariate methods should have advantages, because they allow for a more flexible adjustment to the respective functions $f_i$, e.g. by different hyperparameter settings. This effect should be particularly pronounced in the case of low dependencies. These initial thoughts motivate the consideration of simulation settings with (a) weak and strong dependencies and (b) equal and different univariate DGPs, i.e., $f_i$'s. We elaborate more on this in the next section.

## 3. Simulation Set-up

In our simulation study, we compare the methods described in Section 2 with respect to their (i) predictive power and (ii) runtime. Thereby, the multivariate and univariate L1 and L2 distances are used as impurity functions in the construction process. Moreover, for the univariate approaches, we build separate models for each component of the output vector. All simulations were conducted in the statistical computing software `R` (R Core Team, 2021). For the Multi-task Extra Trees approach we use the `extraTrees` package (Simm et al., 2014). For all other approaches, we implemented our own tree construction algorithm to allow a fair runtime comparison among them. All code required to reproduce the simulations can be found in Schmid and Gerharz (2022). The concrete simulation settings are described below.

We consider a 3-dimensional output vector $\mathbf{Y}$ together with 10 real-valued features $X_1, \ldots, X_{10}$ for which we specifiy different distributions, dependencies and underlying models, described in the following.

*Feature Dependencies and Distributions.* Three different dependence structures among the features $X_1, \ldots, X_5$ are considered as summarized in Table 1. These are similar to those used in Loh (2002, Table 2). In particular, in the first setting (Independent) we consider completely independent features, where $X_1 = Z \sim N(0, 1)$ is standard normally distributed, $X_2 = W \sim exp(1)$ is standard exponentially distributed, $X_3 = T \sim t_2$ is $t$-distributed with $df = 2$ degrees of freedom, while $X_4 = C_4$ and $X_5 = C_8$ are uniformly distributed on $[0, 4]$ and $[0, 8]$, respectively. This covers symmetric as well as skewed and heavy-tailed distributions. In the other two settings, we modeled a weak dependence between the first three features and a strong dependence between the first two, respectively, see the last two coloumns of Table 1. In addition, $X_6, \ldots, X_{10}$ are independent uniformly distributed random variables on the unit interval and independent of the first five features.

*Models.* Various relationships between the output and some of the features are considered as given in Table 2. The first six types of dependent models are designed similarly to those in Loh (2002, Table 7). Here, for each type there exists one model (high), where all outputs are produced by the same DGP and have a high

| Features | Independent (ind) | Weakly dependent (wd) | Strongly dependent (sd) |
|---|---|---|---|
| $X_1$ | $Z$ | $Z+W+T$ | $W+0.1Z$ |
| $X_2$ | $W$ | $W$ | $W$ |
| $X_3$ | $T$ | $T$ | $T$ |
| $X_4$ | $C_4$ | $C_2$ | $C_2$ |
| $X_5$ | $C_8$ | $C_8$ | $C_8$ |

Table 1: Distributions of $X_1, \ldots, X_5$ used in the simulation studies. $C_m$ denotes a random variable that is uniformly distributed on $[0, m]$, $T \sim t_2$, $W \sim \exp(1)$, $Z \sim \mathcal{N}(0,1)$ and $C_m, T, W$ and $Z$ are mutually independent.

correlation. In another model (low), the features used in the DGP to compute the outcomes are different for each of the three outcomes. In these low models, the correlation between the outcomes will be much lower. In addition, we also introduce a linear model and three types of MGAMs (multivariate generalized additive models). The linear model is the sum of the first features and is presented as one model with high and another one with low correlated outcomes. The first MGAM model (MGAM1) considers the same additive structure for all the outcomes. Again we introduce one model with high and another model with low correlated outcomes. The other two types of MGAMs use different DGPs for each of the outcomes. We thus do not distinguish between low and highly correlated outcomes for these two models.

| | |
|---|---|
| $\text{Jump}_{high}$ | $\mathbf{Y} = (U + 0.7\mathbb{1}(X_3 > 1))\mathbf{1} + \varepsilon,$ |
| $\text{Jump}_{low}$ | $\mathbf{Y} = (U + 0.7\mathbb{1}(X_1 > 1),\ U + 0.7\mathbb{1}(X_2 > 1),\ U + 0.7\mathbb{1}(X_3 > 1))^\top + \varepsilon,$ |
| $\text{Quadratic}_{high}$ | $\mathbf{Y} = 0.8X_2^2\mathbf{1} + \varepsilon,$ |
| $\text{Quadratic}_{low}$ | $\mathbf{Y} = 0.8(X_1^2,\ X_2^2,\ X_3^2)^\top + \varepsilon,$ |
| $\text{Cubic}_{high}$ | $\mathbf{Y} = 0.02X_2^3\mathbf{1} + \varepsilon,$ |
| $\text{Cubic}_{low}$ | $\mathbf{Y} = 0.02(X_1^3,\ X_2^3,\ X_3^3)^\top + \varepsilon,$ |
| $\text{Additive}_{high}$ | $\mathbf{Y} = 0.7\mathbb{1}(X_3 > 1)\mathbf{1} + 0.125\sum_{i=1}^{8} i\mathbb{1}(i-1 \le X_5 < i)\mathbf{1} + \varepsilon,$ |
| $\text{Additive}_{low}$ | $\mathbf{Y} = 0.7(\mathbb{1}(X_1 > 1),\ \mathbb{1}(X_2 > 1),\ \mathbb{1}(X_3 > 1))^\top + 0.125\sum_{i=1}^{8} i\mathbb{1}(i-1 \le X_5 < i)\mathbf{1} + \varepsilon,$ |
| $\text{Cross}_{high}$ | $\mathbf{Y} = 0.5\text{sgn}(X_3 - 1)X_2\mathbf{1} + \varepsilon,$ |
| $\text{Cross}_{low}$ | $\mathbf{Y} = 0.5(\text{sgn}(X_1 - 1)X_3,\ \text{sgn}(X_2 - 1)X_1,\ \text{sgn}(X_3 - 1)X_2)^\top + \varepsilon,$ |
| Random $\text{jump}_{high}$ | $\mathbf{Y} = \text{sgn}(X_3 - 1)U\mathbf{1} + \varepsilon,$ |
| Random $\text{jump}_{low}$ | $\mathbf{Y} = U(\text{sgn}(X_1 - 1),\ \text{sgn}(X_2 - 1),\ \text{sgn}(X_3 - 1))^\top + \varepsilon,$ |
| $\text{Linear}_{high}$ | $\mathbf{Y} = \sum_{i=1}^{5} X_i\mathbf{1} + \varepsilon,$ |
| $\text{Linear}_{low}$ | $\mathbf{Y} = (\sum_{i=1}^{3} X_i,\ \sum_{i=1}^{5} X_i,\ \sum_{i=1}^{10} X_i)^\top + \varepsilon,$ |
| $\text{MGAM1}_{high}$ | $\mathbf{Y} = (X_2^2 + \log(X_4) + \cos(X_5))\mathbf{1} + \varepsilon,$ |
| $\text{MGAM1}_{low}$ | $\mathbf{Y} = (X_2^2 + \log(X_4) + \cos(X_5),\ X_4^2 + \log(X_5) + \cos(X_2),\ X_5^2 + \log(X_2) + \cos(X_4))^\top + \varepsilon,$ |
| MGAM2 | $\mathbf{Y} = (0.1\sin(X_1), 0.5\log(X_2), U)^\top + \varepsilon,$ |
| MGAM3 | $\mathbf{Y} = (0.1\sin(X_1), 0.5\log(X_2), U)^\top + \sum_{i=4}^{10} X_i\mathbf{1} + \varepsilon.$ |

Table 2: Different dependent models between the output and some of the features. $U$ is a uniform random variable on $[0, 1]$, $\mathbf{1} = (1, 1, 1)^\top$. Random jump in the following *rjump*

Thereby the error $\varepsilon \sim \mathcal{N}(0, \Sigma_{\ell\rho})$ is generated from a multivariate normal distribution with covariance matrix

$$\Sigma_{\ell\rho} = \begin{pmatrix} 1 & \rho & \rho^\ell \\ \rho & 1 & \rho \\ \rho^\ell & \rho & 1 \end{pmatrix}.$$

Here, we distinguish between independence ($\rho = 0$) and moderate respectively strong correlations $\rho \in$

$\{0.5, 0.9\}$ of adjacent components. The correlation between the first and last component is either equal to $\rho$ ($\ell = 1$) or smaller ($\ell = 2$) corresponding to a compound symmetry and autoregressive covariance structure, respectively. For each setting, we generated samples of size $n$ from the respective model with $n \in \{100, 200, 500\}$. In total, this results in 18 (DGPs) $\times 5$ (errors) $\times 3$ (feature dependencies) $\times 3$ (sample sizes) $= 810$ different simulation settings for each of the 9 ensemble approaches.

*Choice of Ensemble Parameters.* In order not to have to discuss the different possibilities for hyperparameter tuning, we use the default values recommended in the literature (Breiman, 2001; Wright and Ziegler, 2017; Hastie et al., 2001). This has the additional advantage of a reduced runtime. Thus, each ensemble learner consists of 500 trees, the inner bootstrap sample is equal to $m_{try} = 3 = \lfloor \frac{10}{3} \rfloor$, the number of sample points $N$ in the bagging step is equal to the number of sample size $n$. Each terminal node should at least contain five observations. Following the default values of the multi-task Extra Tree implementation (Simm et al., 2014), the number of random cuts is set to one.

*Performance Measures.* As the three components of the output are computed on the same scale, we use the overall MSE for all three output components together to evaluate the predictive power. Additionally, we also consider the runtime of the algorithms. For each setting, we repeated the runtime measurement and the MSE computation $1,000$ times.

## 4. Results

In this section, we describe the results of the simulation study. In particular, we present the overall MSE of the different construction algorithms under various simulation configurations. The runtime analysis can be found in the Appendix.

### *4.1. Predictive Power*

For ease of presentation, we aggregated the overall MSE with respect to the $1,000$ replications. Note that the simulation results of the methods in the setup of Cubic Low and Quadratic Low relationship are not presented in the graphics. This is because of their relatively poor performance regarding the predictive power. The methods resulted into average MSE values being greater than $100,000$ (see Figure A.6 and A.7 of the Appendix).

The average MSE for most of the relationships between output vector and features separated by the sample size and the used methods are shown in Figure 1. Here, each boxplot represents 5 (errors) $\times 3$ (dependencies) $= 15$ different average MSE values. More detailed boxplots without averaging but w.r.t. the simulation runs are given in the Appendix (see Figures A.8 - A.23). These do not alter the following findings. Generally, we observe that the performance of the methods depends on the relationship between output and features. Except in the MGAM2, either the Random Forest approaches or the multi-task Extra Trees outperformed the other approaches regarding the predictive power. In the MGAM2 setup, the univariate approaches exhibit the smallest MSE values, especially the univariate Extra Trees. Similar results can be observed for MGAM3, where the respective differences between univariate and multivariate approaches are smaller.

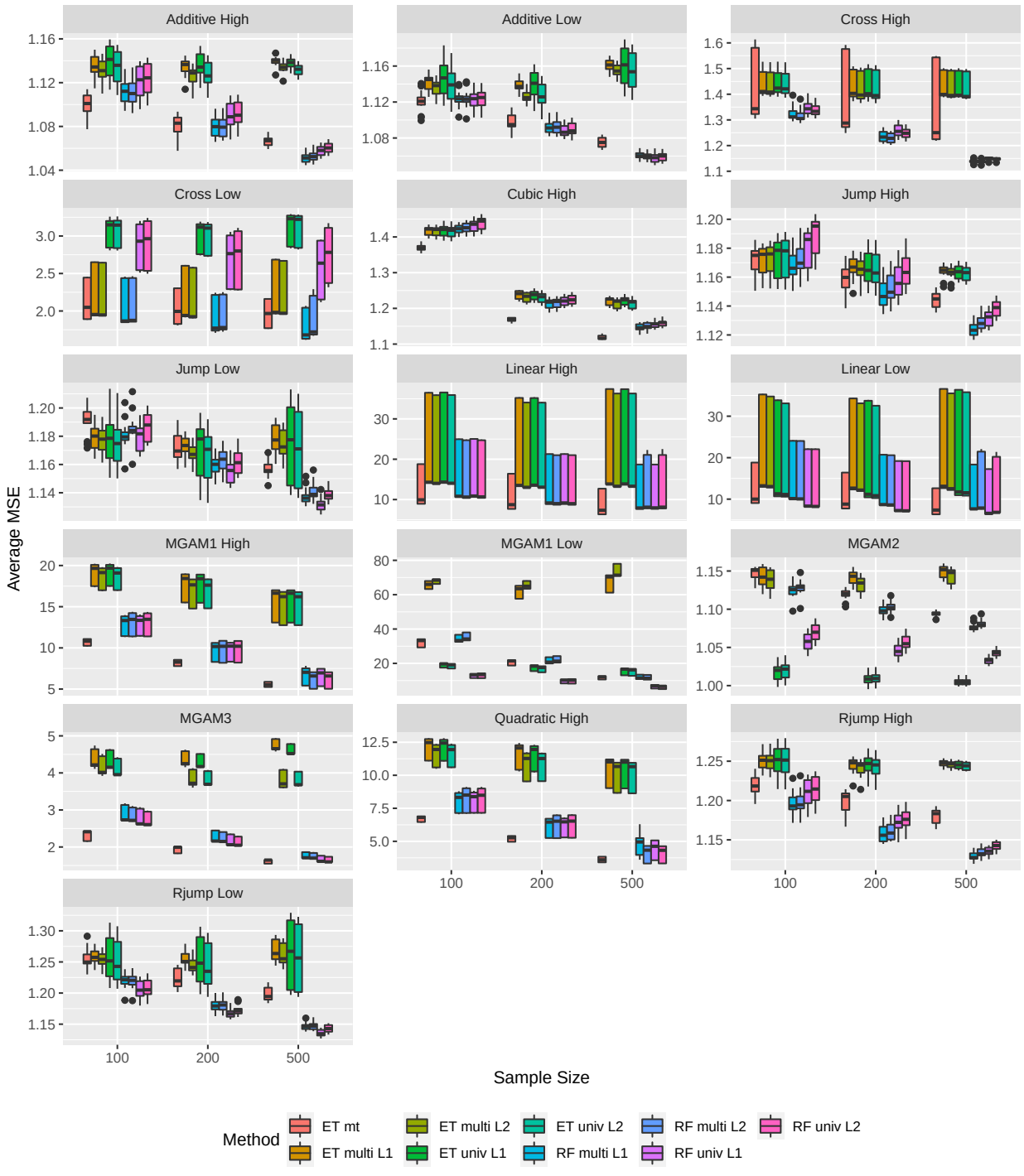


Figure 1: Simulation results on the average MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) separated by the relation between output and feature and sample sizes.

In all high setups the Random Forest approaches either performed similarly (linear high, MGAM1 high and quadratic high) or the multivariate outperformed the univariate approach (additive high, cross high, cubic high, jump high, rjump high). For the Extra Trees, on the other hand, no major differences in the performance of the univariate and multivariate approaches are visible for the high setups. However, we observed that the variation in MSE values is smaller for the multivariate approaches. When focusing on the low setups, the performance of the univariate approaches improves. Here, they outperform the multivariate approaches in the linear low, MGAM1 low and rjump low setups or performed similarly well (additive low, jump low). Only in the cross low case, the MSE of the Random Forest methods was better for the multivariate approaches. However, the differences between the univariate and multivariate methods are very small except for MGAM1 low and cross low. In general, the average MSE increases for the low setups compared to the high setups.

As the sample size increases, the average MSEs for both Random Forest approaches decreases in all setups. However, the improvement in predictive power becomes slightly smaller with increasing sample size. The same observation can be made for the multi-task Extra Trees. This is different to the multivariate and univariate Extra Trees: While the average MSEs slighlty decrease when the sample size is doubled from 100 to 200, the changes in average MSEs are either marginal or the average MSEs increase slightly when comparing sample sizes 200 and 500. Comparing the influence of the different impurity measures L1 and L2, we find that the predictive power is almost equal for most cases. Thus, no clear recommendation can be made.

Figures 2 and 3 summarize the predicition results for all methods separated by the relationship between output vector and features and the dependency structure of the features. When the features are weakly dependent, all methods perform poorly in the linear low and high setups. Overall, the average MSEs are more than twice as large as for the other two feature dependence structures except for the multi-task Extra Trees. Here, the difference to the other feature dependence structures is slightly smaller. In the rjump low setup, we observed that the univariate Extra Trees also perform poorly in the weak dependent structure setting.

Different performances of the univariate and multivariate approaches are observed in the cross-low setting. While the multivariate approaches have the largest average MSE values when the features are weakly dependent, the univariate approaches perform best in this setting. In the corresponding high setup, the performances of the random forest or Extra Trees approaches are similar, with the Extra Trees approaches, especially the multi-task approach, showing larger average MSEs in the strong dependent feature setting. For the quadratic high, MGAM1 high and low setup, it can be noted that the average MSE values decrease with increasing feature dependence structure. Moreover, the variations of the MSE values are smaller for the univariate approaches compared to the multivariate approaches in the MGAM1 low setup.

Except for these settings, the feature dependency structure had only a slight effect on the predictive power. In the MGAM2 setup, the average MSEs of the multivariate approaches decrease slightly with increasing feature dependence, but the average MSEs of the univariate methods increase. In all other setups, the multivariate and univariate approaches behave similarly with increasing feature dependence.

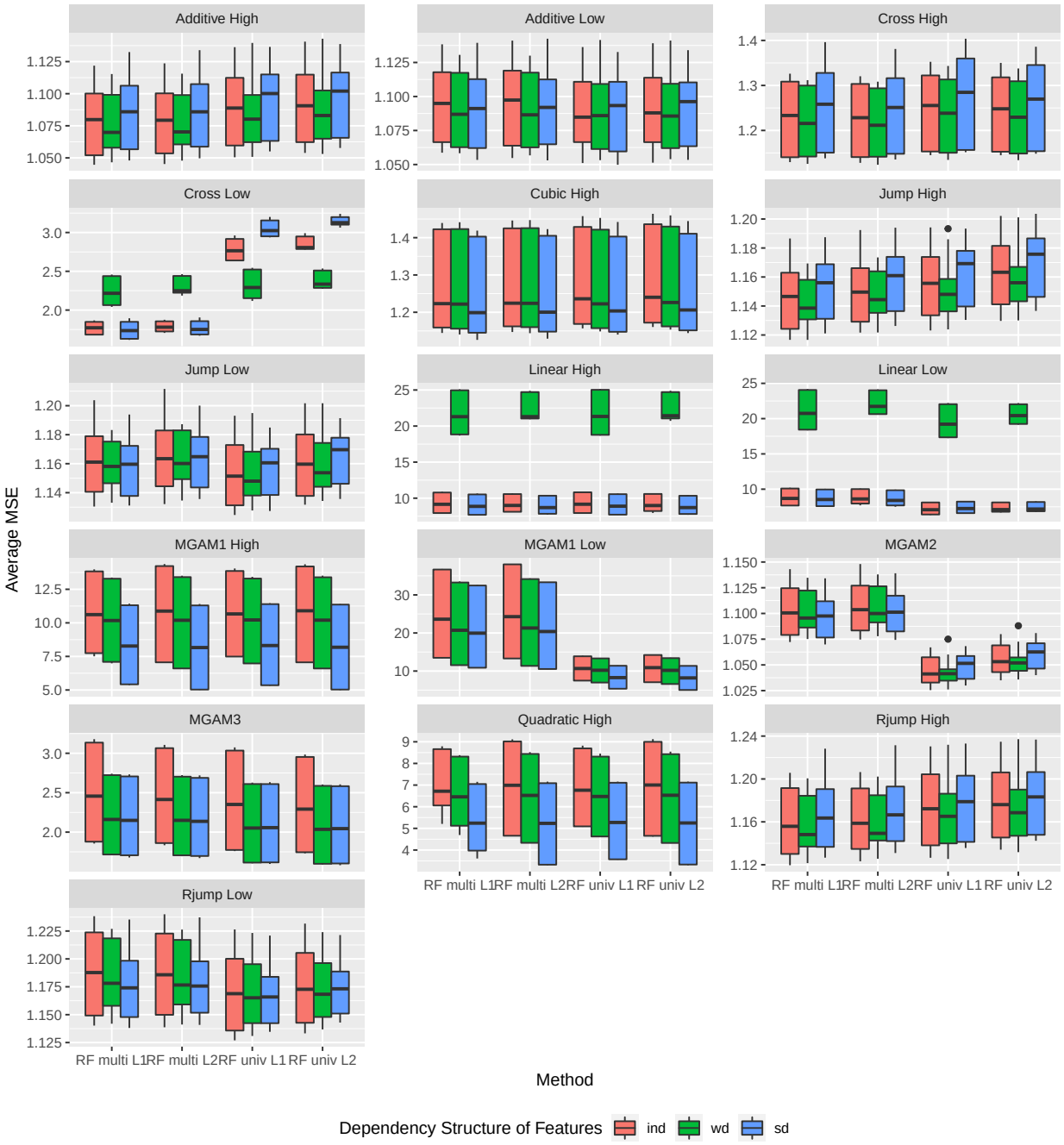


Figure 2: Simulation results on the average MSE for dependency structure of the features separated by the relation between output and feature and all Random Forest approaches multivariate (RF multi L1 and L2) and univariate (RF univ L1 and L2). Each boxplot represents 5(errors)×3(sample sizes)=15 different average MSE values.

Also no difference in the used impurity measures can be observed.

The influence of the different errors on the prediction power can be seen in Figure 4 and 5. Generally,

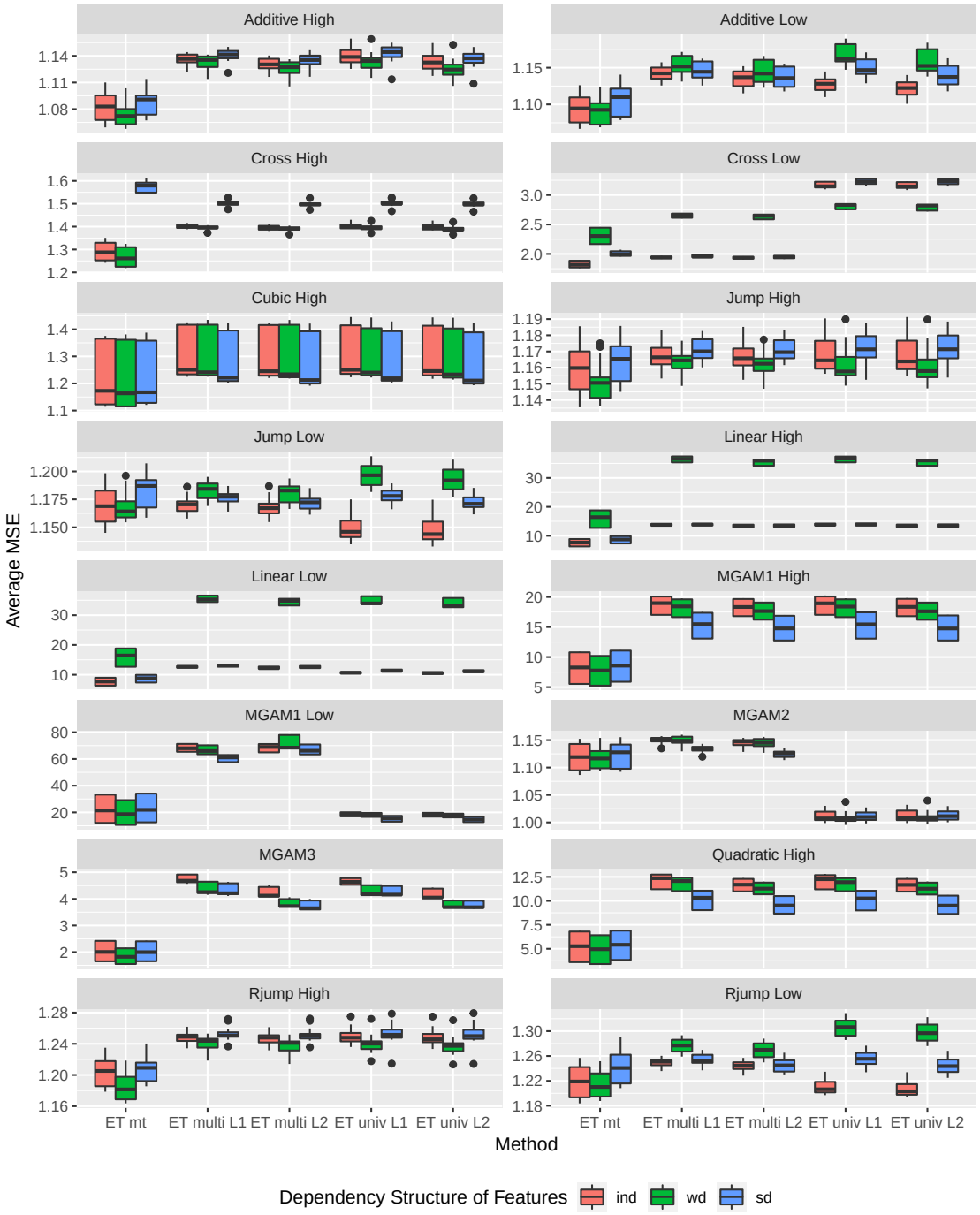


Figure 3: Simulation results on the average MSE for dependency structure of the features separated by the relation between output and feature and all Extra Trees approaches multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2) and univariate Extra Trees (ET univ L1 and L2). Each boxplot represents 5(errors)×3(sample sizes)=15 different average MSE values.

the different errors had little impact on the predictive power. For the linear high and low, MGAM1 high and low, MGAM3 and quadratic high setup, different errors did not lead to average MSE differences. In

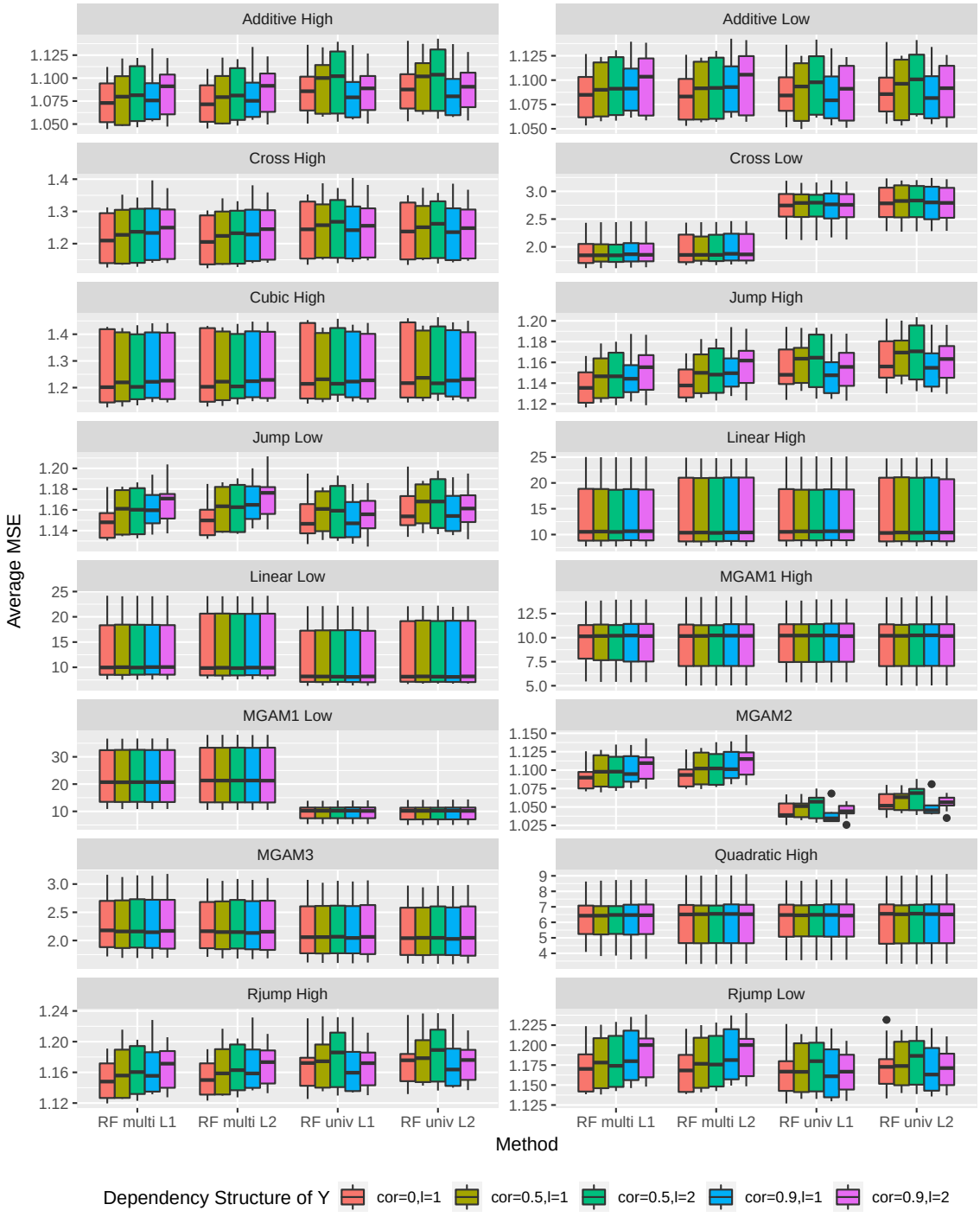


Figure 4: Simulation results on the average MSE for dependency structure of the outputs separated by the relation between output and feature and for all Random Forest approaches multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2). Each boxplot represent 3(feature dependencies)×3(sample sizes)=9 different average MSE values.

addition, the Extra Tree approaches are also similar in the cross low setup. In all other setups, the average MSEs of the multivariate Random Forest increase with an increasing dependency structure in the outputs.

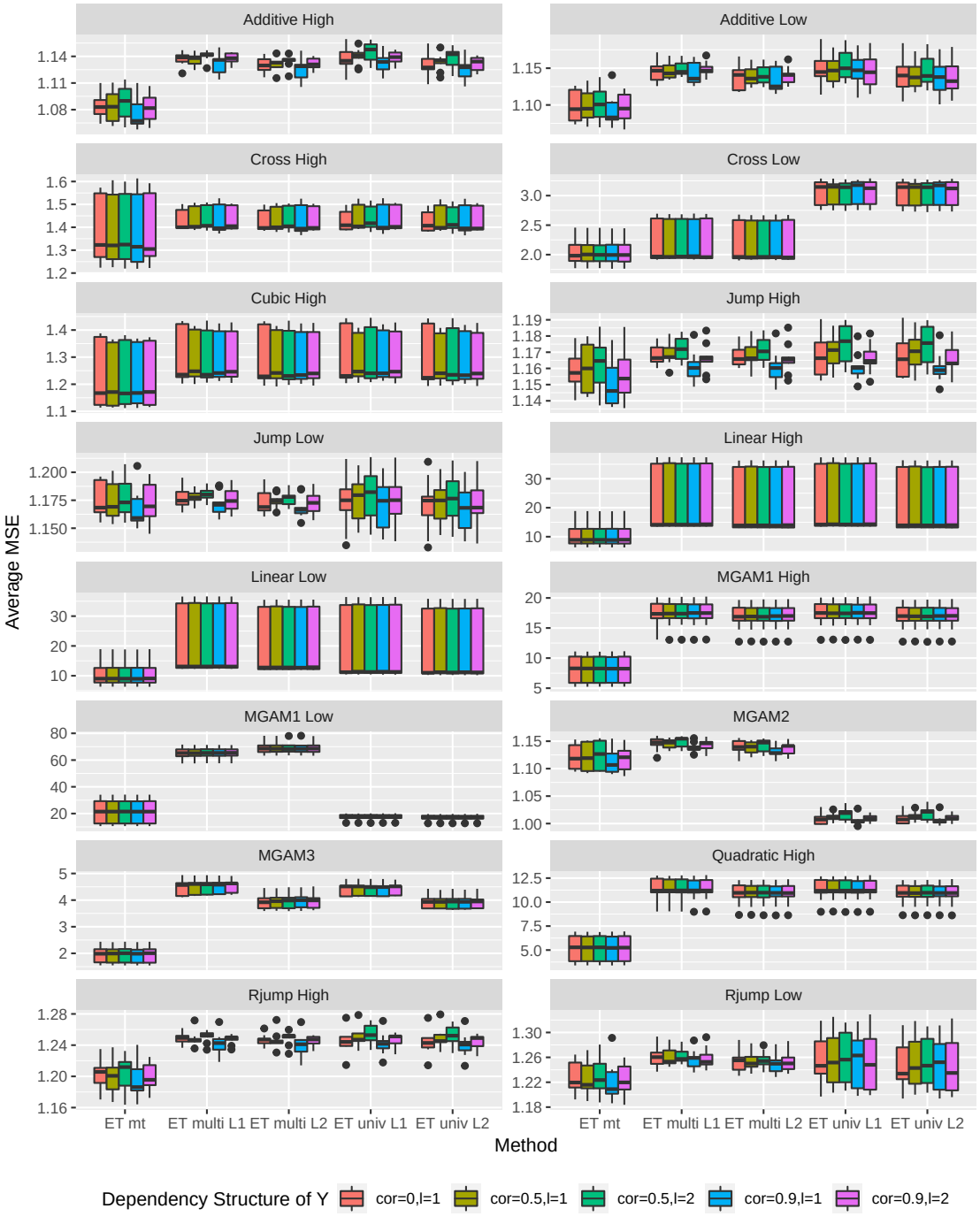


Figure 5: Simulation results on the average MSE for dependency structure of the outputs separated by the relation between output and feature and for all Extra Tree methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (rf_multi) and univariate Random Forest (rf_univ). Each boxplot represent 3(feature dependencies)×3(sample sizes)=9 different average MSE values.

## 5. Illustrative Data Examples

In the following experiment, the investgated methods are compared on five multi-output regression data sets. Each data set has a different complexity, see Table 3. It reports the name, source, number of samples, number of features and dimension of the output vector for each data set. A detailed description of the data sets can be found in the cited papers.

| Data Set | Source | $n$ | $p$ | $d$ |
|---|---|---|---|---|
| Concrete | (Yeh, 2007; Dua and Graff, 2017) | 103 | 8 | 3 |
| Jura | (Goovaerts et al., 1997) | 359 | 7 | 4 |
| Hspider | (Smeenk-Enserink and Aart, 1974) | 28 | 6 | 12 |
| Stock | (Liu and Yeh, 2015; Dua and Graff, 2017) | 63 | 6 | 6 |
| Enb | (Tsanas and Xifara, 2012; Dua and Graff, 2017) | 768 | 7 | 2 |

Table 3: Name, source, number of samples ($n$), number of features ($p$) and dimension of the output vector of each ($d$) used data set.

For the analyses, the output vectors were standardized to zero mean and unit variance. To compare the overall MSE of the approaches, we applied 5-fold cross-validation and repeated it 100 times. The results are reported in Table 4. We have also performed a runtime analysis (see Table B.6 in the Appendix), which matches the corresponding analyses in the simulation study.

| Method | Concrete | Jura | Hspider | Stock | Enb |
|---|---|---|---|---|---|
| RF univ L1 | 0.538 | 0.339 | 0.669 | 0.533 | 0.035 |
| RF univ L2 | 0.524 | 0.334 | 0.678 | 0.512 | 0.032 |
| RF multi L1 | 0.569 | 0.367 | 0.675 | 0.557 | 0.034 |
| RF multi L2 | 0.560 | 0.359 | 0.673 | 0.550 | 0.031 |
| ET univ L1 | 0.744 | 0.665 | 0.811 | 0.812 | 0.124 |
| ET univ L2 | 0.708 | 0.618 | 0.779 | 0.791 | 0.092 |
| ET multi L1 | 0.721 | 0.576 | 0.772 | 0.766 | 0.109 |
| ET multi L2 | 0.603 | 0.533 | 0.771 | 0.765 | 0.109 |
| ET mt | 1.092 | unavailable | 1.082 | 1.300 | unavailable |

Table 4: Average MSE of the methods presented in Section 2 for all data sets.

As the multi-task Extra Trees require numeric features, the overall MSE could not be computed for the jura and enb data sets. In general, the RF approaches outperformed the Extra Trees methods across all data sets. When comparing the univariate and multivariate approaches, all RF approaches show almost similar results with a slight advantage (at the second or third decimal position) for the univariate RF. Only for enb the multivariate approach performs slightly better. For the Extra Trees, the observation is reserved, i.e. multivariate Extra Trees lead to slightly lower MSEs than the univariate ones across all data sets, except again on enb with the L2 impurity measure. The difference in performance when comparing the L1 and L2 distances (for the split) shows a slight advantage for L2. Here, the only exception is the univariate RF in the hspider data set, where it is the other way around. However, as before, the differences between the two impurity functions used are negligible. There, the only exception is the multivariate Extra Trees algorithm on the concrete data set, where the results differ at the first decimal. A more detailed presentation of the results can be found in Tables B.7-B.11 in the Appendix, where for each data set the MSE for each

component of the output vector is given. We also exemplified a more detailed comparison on the concrete data set. As the conclusions are similar, the results are only given in the Appendix.

## 6. Conclusion and Outlook

The main purpose of this simulation study was to compare the predictive accuracy of univariate Random Forests and Extra Tree algorithms with multivariate approaches in case of multivariate outputs.

In most of the simulation settings it was clearly shown that either the Random Forest or the Extra Tree approaches have yielded better performances than the other methods. However, comparing the multivariate approaches with their univariate counterparts, advantages for the univariate approaches in terms of performance (average MSE) can be observed in most simulation settings, where the DGPs of the components of the output are different. In all other simulation settings the multivariate approaches have shown at least similar or even better performances than the univariate approaches. Especially, when comparing univariate and multivariate Random Forests, the multivariate approaches were substantially better for some of the considered settings. Moreover, for all methods, the different dependency structures within the features did have an impact on the methods' performance. However, it was virtually the same for both, the univariate approaches and the multivariate approaches. Another interesting finding was that correlation within the outputs only showed substantial impacts on the performance of the multivariate Random Forests in some simulation settings, while for all other approaches this had little to no impact at all.

While the differences of the univariate and multivariate approaches' predictive performance were small to moderate, a huge difference in the runtime could be noted. Here, the multi-task Extra Tree method was the fastest approach. However, it was the only method that was not specially implemented for this simulation study. For a fair runtime comparison the other approaches were implemented in a comparable way and it was shown that the multivariate approaches have a huge runtime advantage over the univariate approaches.

Last but not least, illustrative data analyses showed that the multivariate approaches can improve the performance when considering multivariate outputs, particularly for the Extra Trees. However, for the Random Forests the multivariate counterpart could only improve the performance in the enb data set.

As only regression problems with numeric outputs were considered, future simulation studies should investigate whether the same potential for improvement can also be found for multivariate classification problems. Also, mixed problems with numeric and categorical outputs at the same time have to be investigated. Moreover, as only relatively small sample sizes were investigated, the behavior of these approaches in big data settings with larger sample sizes should also be further researched. In addition, multivariate tree-based ensembles should be compared with other univariate and multivariate regression methods in the future. As the runtime of our tree construction algorithm is quite high for some settings, we should further explore how to make the construction algorithm more efficient.

**Acknowledgement**

The authors gratefully acknowledge the computing time provided on the Linux HPC cluster at Technical University Dortmund (LiDO3), partially funded in the course of the Large-Scale Equipment Initiative by the German Research Foundation (DFG) as project 271512359. The HPC LiDO3 from TU Dortmund is a heterogeneous computing cluster with 30 TB RAM and 8,160 CPU cores spread over 366 nodes.
Declaration of interest: none.

## References

Breiman, L., 2001. Random Forests. Machine Learning 45, 5 – 32. doi:https://doi.org/10.1023/A:1010933404324.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J., 2017. Classification and Regression Rrees. Routledge.

Cariou, V., 2006. Extension of multivariate regression trees to interval data. application to electricity load profiling. Computational Statistics 21, 325–341.

De'ath, G., 2002. Multivariate Regression Trees: A New Technique for Modeling Species-Environment Relationships. Ecology 83, 1105–1117.

De'ath, G., 2012. MVPART: Multivariate Partitioning. R package version 1.6-0.

Dine, A., Larocque, D., Bellavance, F., 2009. Multivariate trees for mixed outcomes. Computational Statistics & Data Analysis 53, 3795–3804. URL: https://www.sciencedirect.com/science/article/pii/S0167947309001376, doi:https://doi.org/10.1016/j.csda.2009.04.003.

Dobler, D., Friedrich, S., Pauly, M., 2020. Nonparametric MANOVA in meaningful effects. Annals of the Institute of Statistical Mathematics 72, 997–1022.

Dua, D., Graff, C., 2017. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

D'Ambrosio, A., Aria, M., Iorio, C., Siciliano, R., 2017. Regression trees for multivalued numerical response variables. Expert Systems with Applications 69, 21–28. URL: https://www.sciencedirect.com/science/article/pii/S0957417416305528, doi:https://doi.org/10.1016/j.eswa.2016.10.021.

Friedrich, S., Konietschke, F., Pauly, M., 2019. Resampling-based analysis of multivariate data and repeated measures designs with the R package MANOVA.RM. R J. 11, 380.

Gerke, J., Koenig, A.M., Conrad, D., Doyen-Waldecker, C., Pauly, M., Gündel, H., Wilker, S., Kolassa, I.T., 2018. Childhood maltreatment as risk factor for lifetime depression: The role of different types of experiences and sensitive periods. Mental Health & Prevention 10, 56–65.

Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. Machine Learning 63, 3–42. URL: https://doi.org/10.1007/s10994-006-6226-1.

Goovaerts, P., et al., 1997. Geostatistics for natural resources evaluation. Oxford University Press on Demand.

Groll, A., Ley, C., Schauberger, G., Van Eetvelde, H., 2019. A hybrid random forest to predict soccer matches in international tournaments. Journal of Quantitative Analysis in Sports 15, 271–287.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning. Springer Series in Statistics, Springer New York Inc.

Hsiao, W.C., Shih, Y.S., 2007. Splitting variable selection for multivariate regression trees. Statistics & Probability Letters 77, 265–271.

Huang, H., Pouls, M., Meyer, A., Pauly, M., 2020. Travel time prediction using tree-based ensembles, in: International Conference on Computational Logistics, Springer. pp. 412–427.

Kocev, D., Ceci, M., 2015. Ensembles of extremely randomized trees for multi-target regression, in: Discovery Science, Springer International Publishing. pp. 86–100.

Kocev, D., Džeroski, S., White, M.D., Newell, G.R., Griffioen, P., 2009. Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. Ecological Modelling 220, 1159–1168.

Konietschke, F., Bathke, A.C., Harrar, S.W., Pauly, M., 2015. Parametric and nonparametric bootstrap methods for general MANOVA. Journal of Multivariate Analysis 140, 291–301.

Larsen, D.R., Speckman, P.L., 2004. Multivariate regression trees for analysis of abundance data. Biometrics 60, 543–549.

Lee, S.K., 2005. On generalized multivariate decision tree by using GEE. Computational Statistics & Data Analysis 49, 1105–1119.

Liu, Y.C., Yeh, I.C., 2015. Using mixture design and neural networks to build stock selection decision support systems. Neural Computing and Applications 28, 521–535.

Loh, W.Y., 2002. Regression trees with unbiased variable selection and interaction detection. Statistica Sinica 12, 361–386.

Loh, W.Y., Zheng, W., 2013. Regression trees for longitudinal and multiresponse data. The Annals of Applied Statistics 7, 495 – 522. doi:10.1214/12-AOAS596.

Mavridis, D., Salanti, G., 2013. A practical introduction to multivariate meta-analysis. Statistical Methods in Medical Research 22, 133–158. doi:10.1177/0962280211432219.

Pauly, O., 2012. Random forests for medical applications. Ph.D. thesis. Technische Universität München.

R Core Team, 2021. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Rahman, R., Otridge, J., Pal, R., 2017. IntegratedMRF: random forest-based framework for integrating prediction from different data types. Bioinformatics 33, 1407–1410. doi:10.1093/bioinformatics/btw765.

Schauberger, G., Groll, A., 2018. Predicting matches in international football tournaments with random forests. Statistical Modelling 18, 460–482.

Schmid, L., Gerharz, A., 2022. Multivariate trees. https://github.com/LenaSchmid/Multivariate-Trees-Code.

Segal, M., Xiao, Y., 2011. Multivariate random forests. Wiley interdisciplinary reviews: Data mining and knowledge discovery 1, 80–87.

Siciliano, R., Mola, F., 2000. Multivariate data analysis and modeling through classification and regression trees. Computational Statistics & Data Analysis 32, 285–301. URL: https://www.sciencedirect.com/science/article/pii/S0167947399000821, doi:https://doi.org/10.1016/S0167-9473(99)00082-1.

Simm, J., De Abril, I.M., Sugiyama, M., 2014. Tree-based ensemble multi-task learning method for classification and regression. IEICE TRANSACTIONS on Information and Systems 97, 1677–1681.

Smaga, Ł., 2017. Bootstrap methods for multivariate hypothesis testing. Communications in Statistics-Simulation and Computation 46, 7654–7667.

Smeenk-Enserink, N., Aart, P.V.D., 1974. Correlations between distributions of hunting spiders (lycosidae, ctenidae) and environmental characteristics in a dune area. Netherlands Journal of Zoology 25, 1–45.

Tsanas, A., Xifara, A., 2012. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. Energy and Buildings 49, 560–567.

Vallejo, G., Ato, M., 2012. Robust tests for multivariate factorial designs under heteroscedasticity. Behavior Research Methods 44, 471–489.

Wright, M.N., Ziegler, A., 2017. ranger: A fast implementation of random forests for high dimensional data in C++ and R. Journal of Statistical Software 77, 1–17. doi:10.18637/jss.v077.i01.

Xiao, Y., Segal, M., 2009. Identification of yeast transcriptional regulation networks using multivariate random forests. PLoS Comput Biol , e1000414.

Yeh, I.C., 2007. Modeling slump flow of concrete using second-order regressions and artificial neural networks. Cement and Concrete Composites 29, 474–480. doi:10.1016/0003-4916(63)90068-X.

Yu, Y., Lambert, D., 1999. Fitting trees to functional data, with an application to time-of-day patterns. Journal of Computational and graphical Statistics 8, 749–762.

Zhang, H., 1998. Classification Trees for Multiple Binary Responses. Journal of the American Statistical Association 93, 180–193.

Zhang, H., Ye, Y., 2008. A tree-based method for modeling a multivariate ordinal response. Statistics and its Interface 1, 169–178. doi:10.4310/sii.2008.v1.n1.a14.

Zimmermann, G., Pauly, M., Bathke, A.C., 2020. Multivariate analysis of covariance with potentially singular covariance matrices and non-normal responses. Journal of Multivariate Analysis 177, 104594.

## Appendix A. Additional Simulation Results

### *Appendix A.1. Setting Quadratic Low*

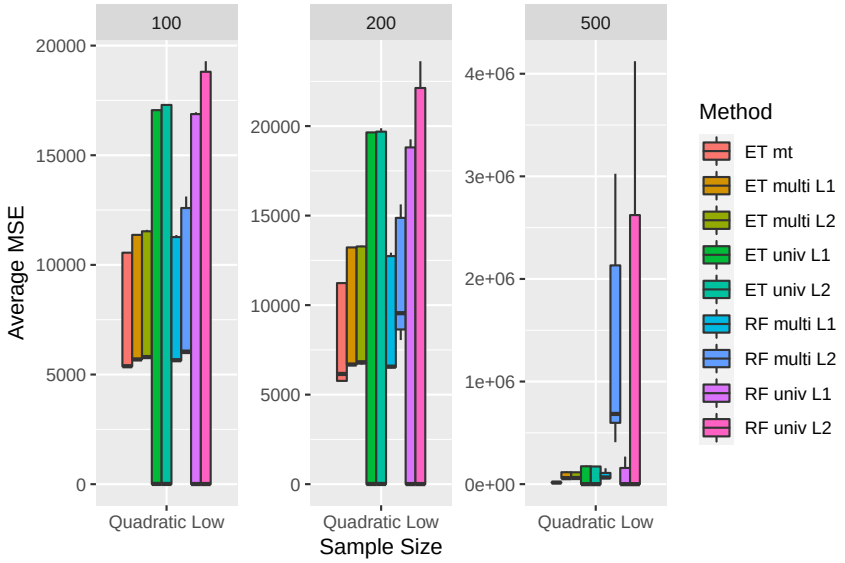


Figure A.6: Simulation results on the average MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Quadratic Low setting separated by sample sizes.

### *Appendix A.2. Setting Cubic Low*

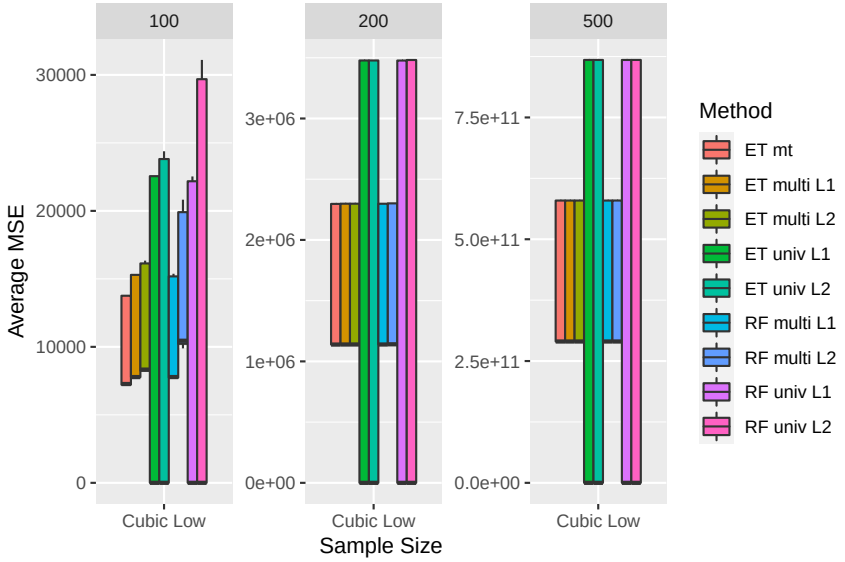


Figure A.7: Simulation results on the average MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Cubic Low setting separated by sample sizes.

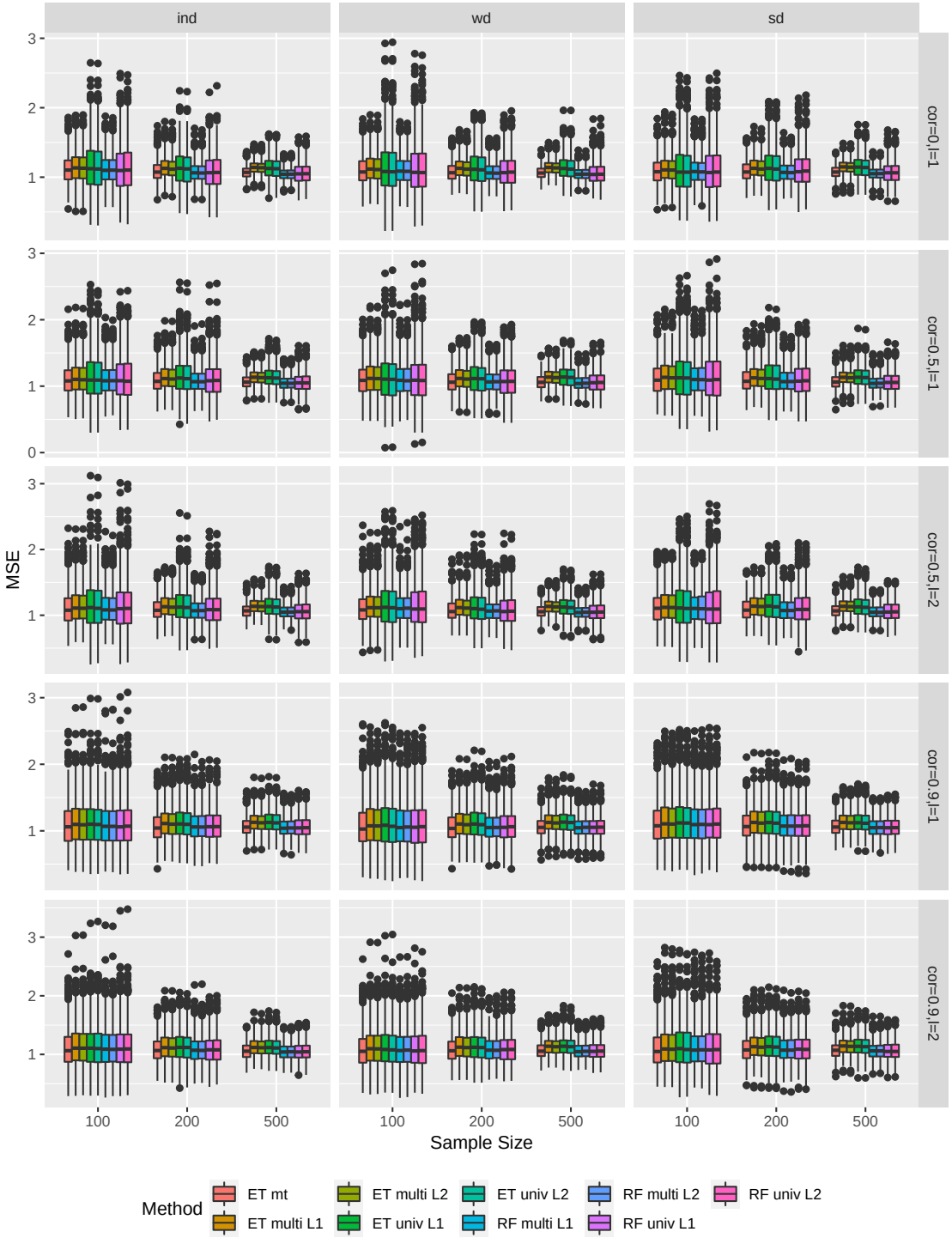


Figure A.8: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Additive High setting. Each boxplots contains 1.000 MSE values.

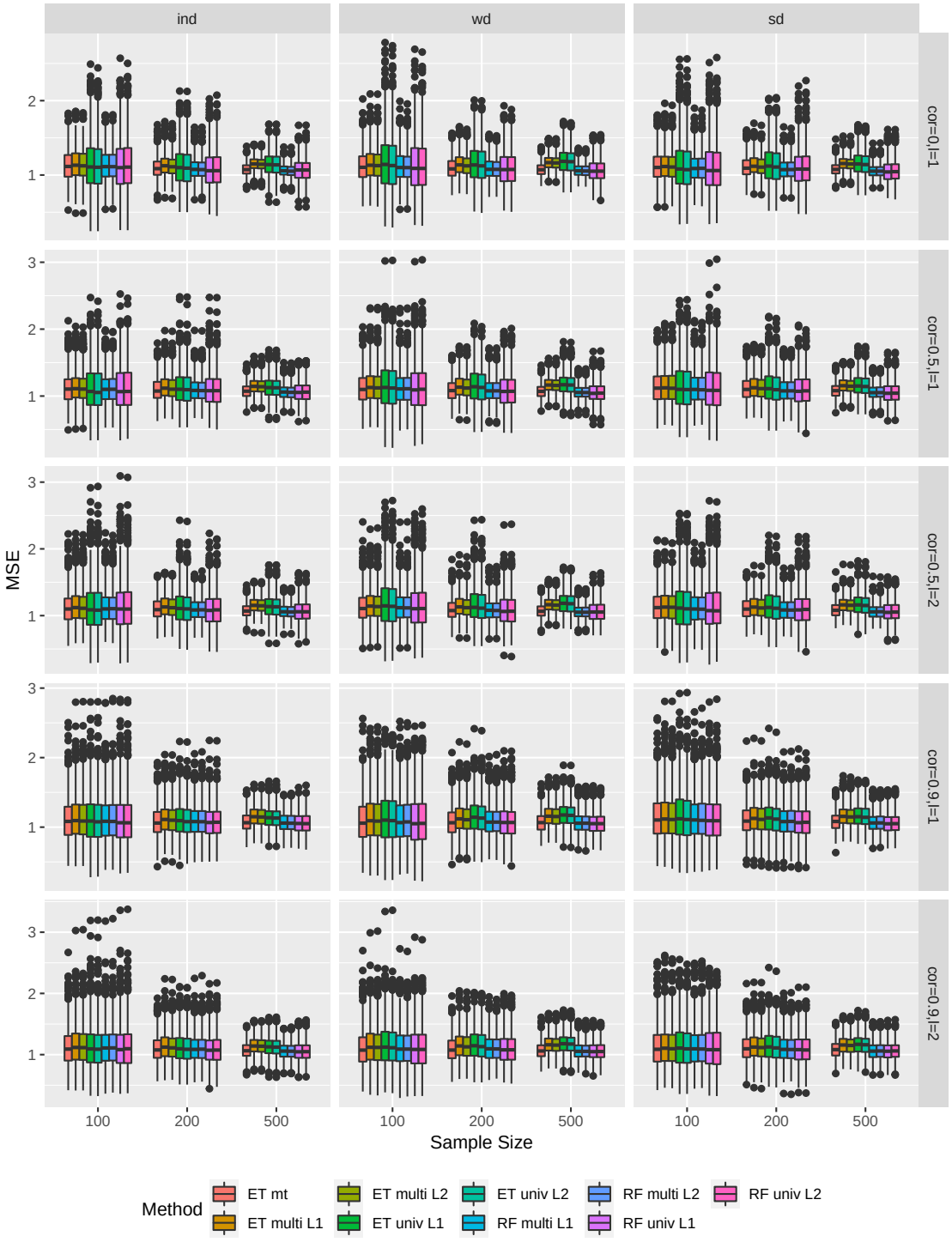


Figure A.9: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Additive Low setting. Each boxplots contains 1.000 MSE values.

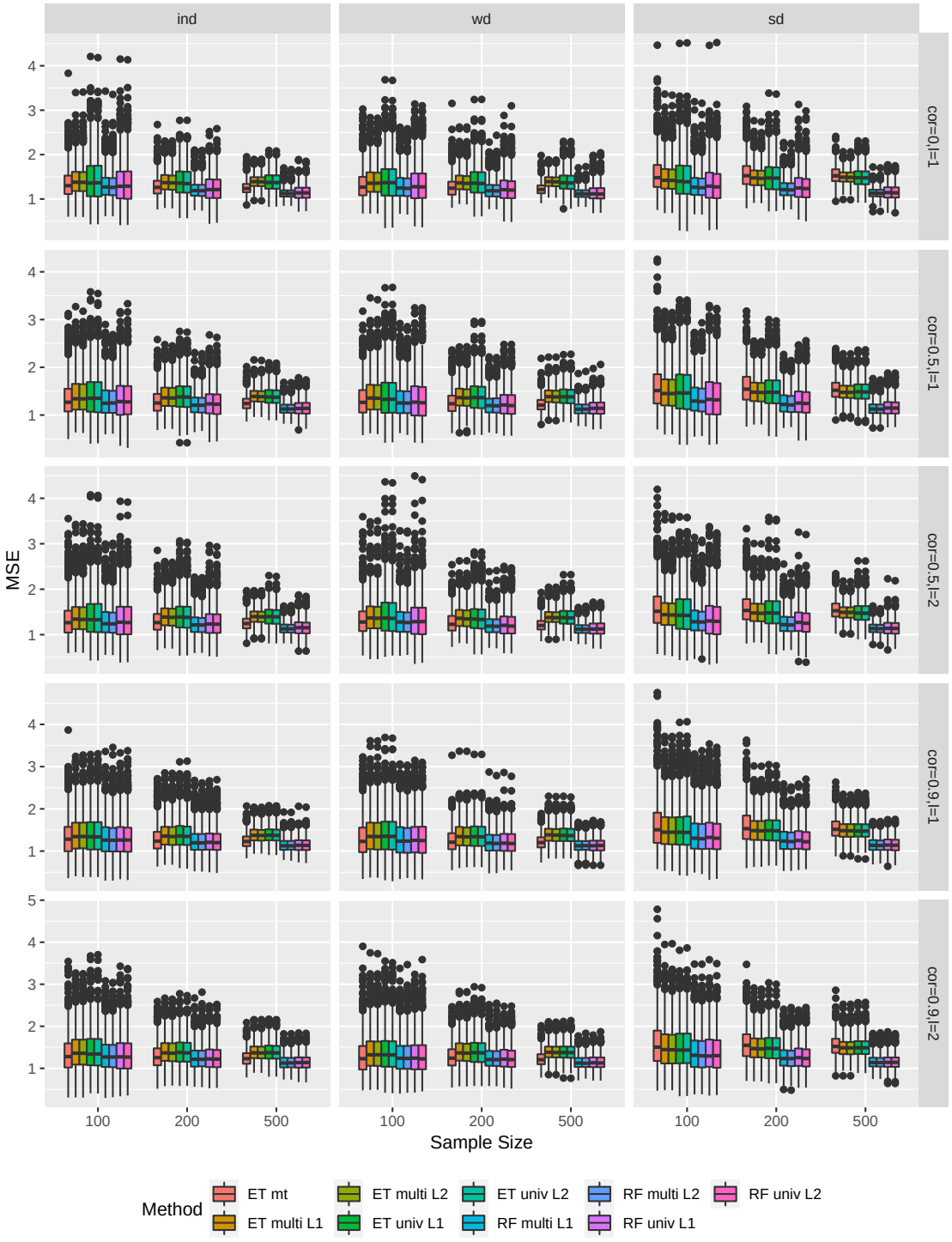


Figure A.10: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Cross High setting. Each boxplots contains 1.000 MSE values.

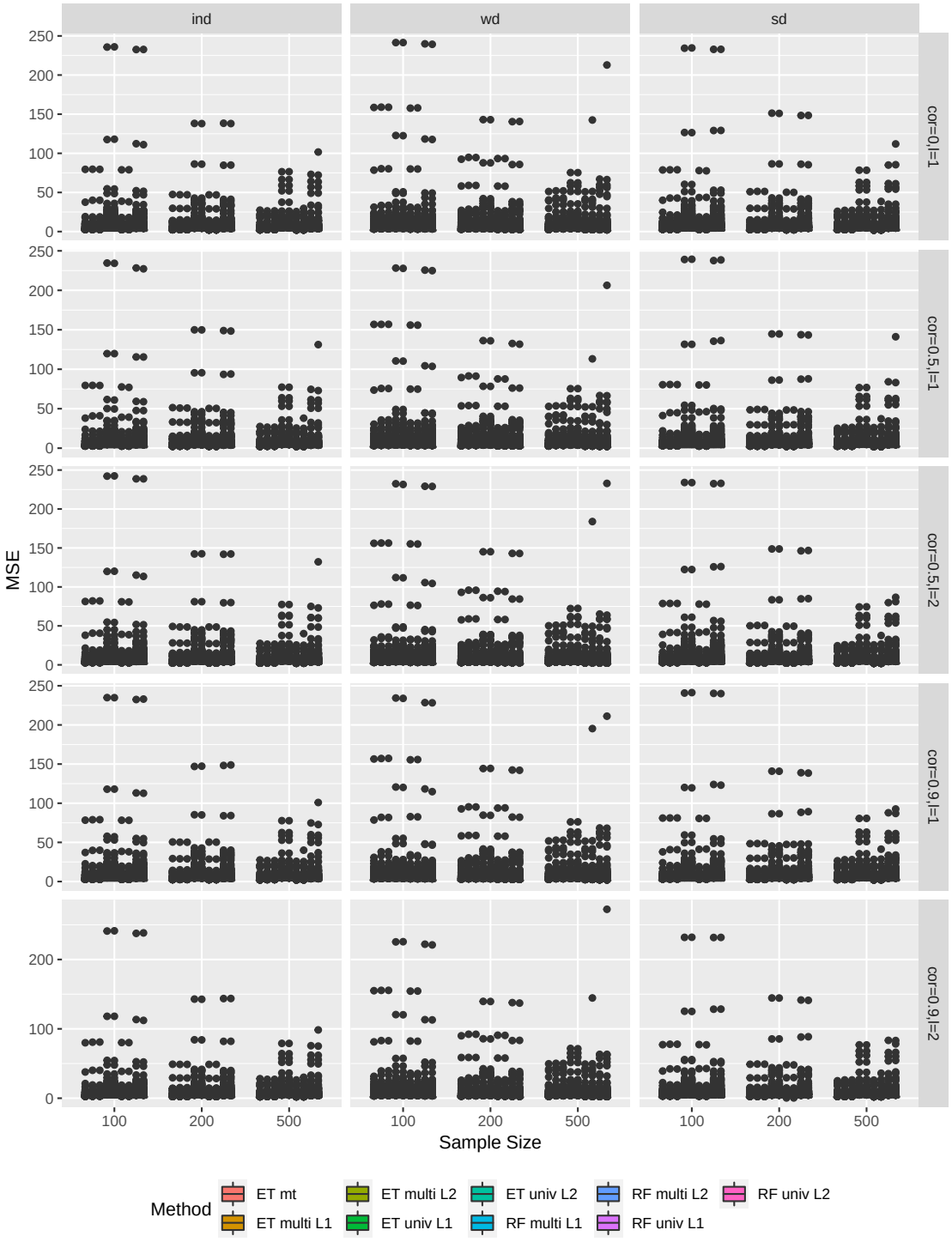


Figure A.11: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Cross Low setting. Each boxplots contains 1.000 MSE values.

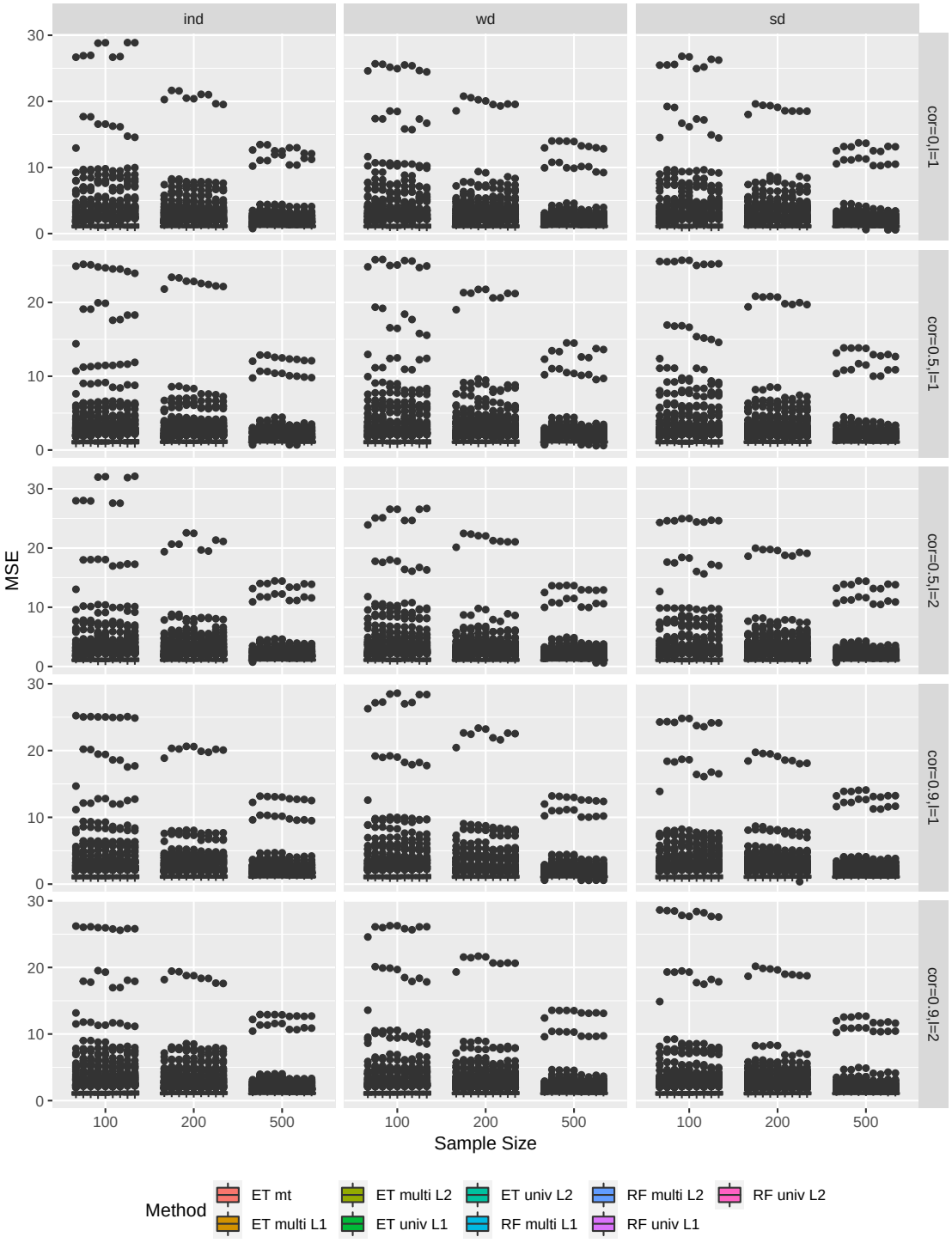


Figure A.12: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Cubic High setting. Each boxplots contains 1.000 MSE values.

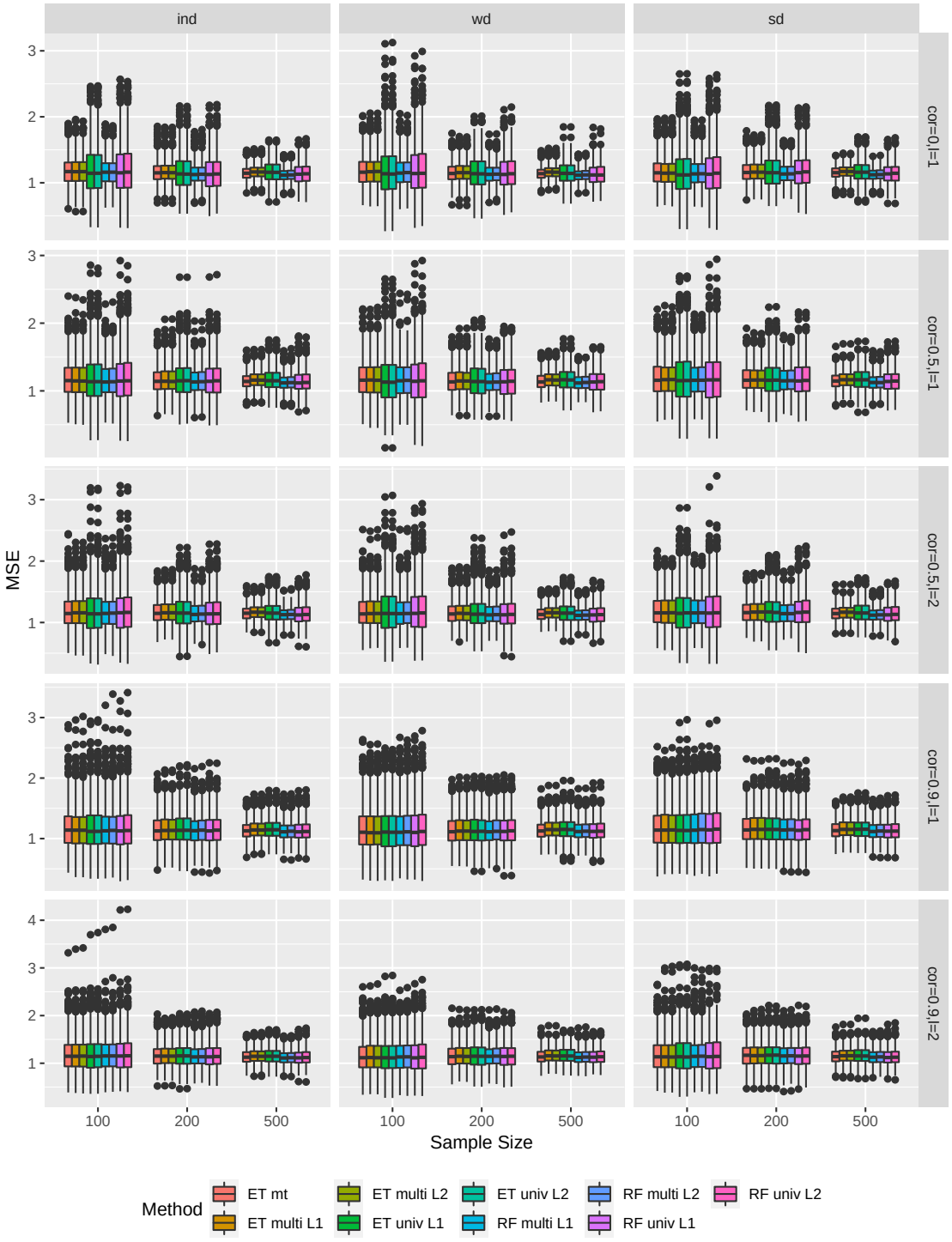


Figure A.13: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Jump High setting. Each boxplots contains 1.000 MSE values.

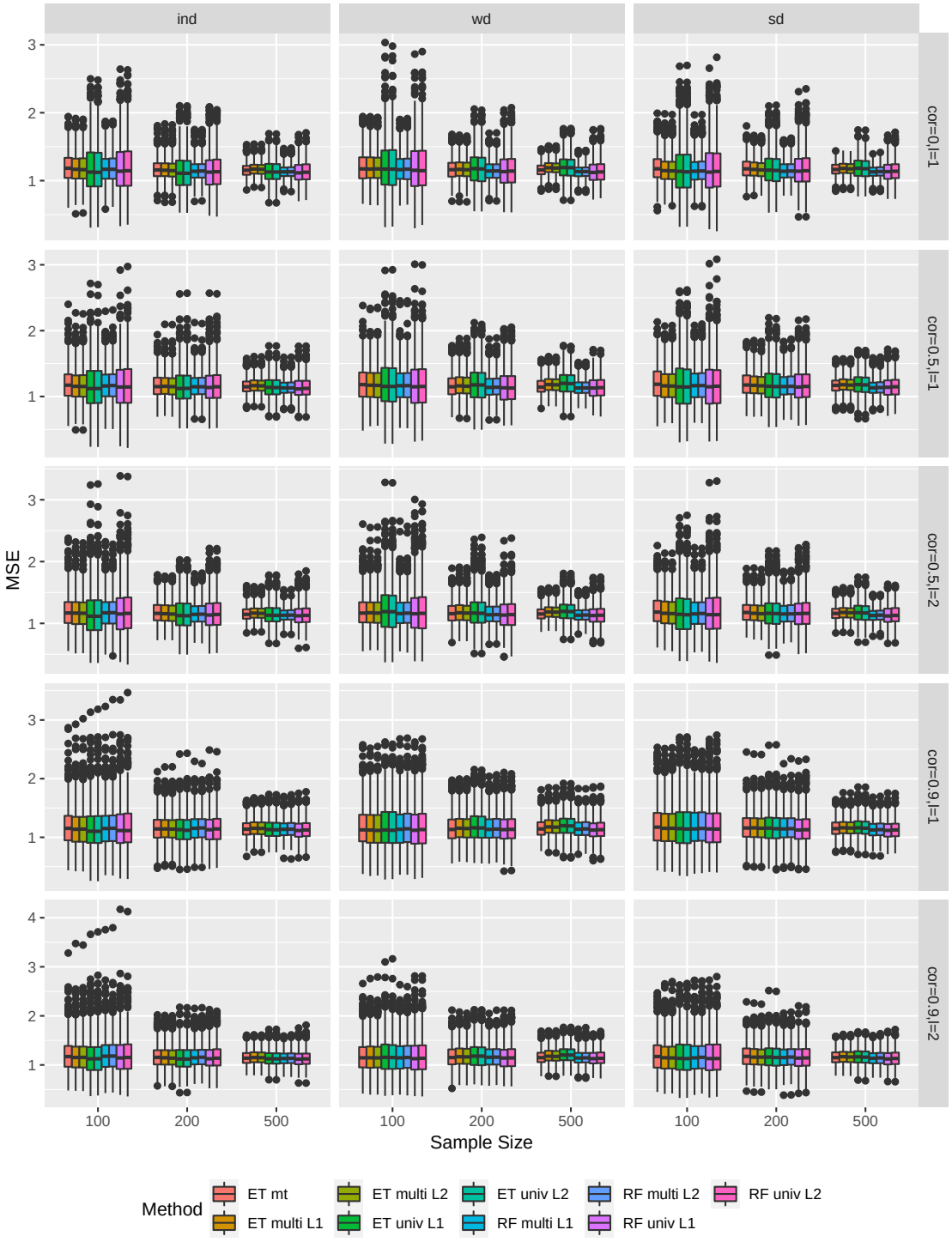


Figure A.14: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Jump Low setting. Each boxplots contains 1.000 MSE values.

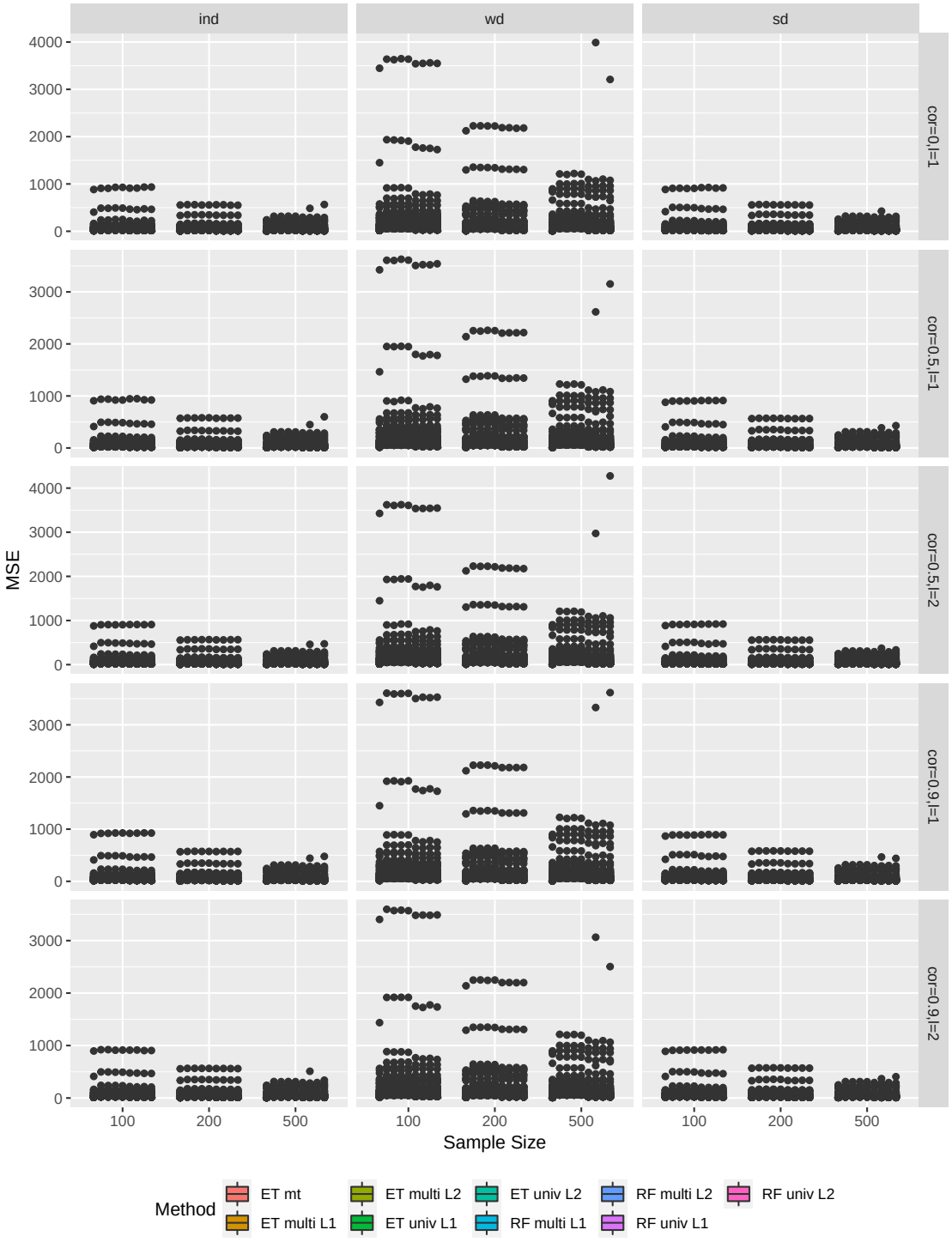


Figure A.15: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Linear High setting. Each boxplots contains 1.000 MSE values.

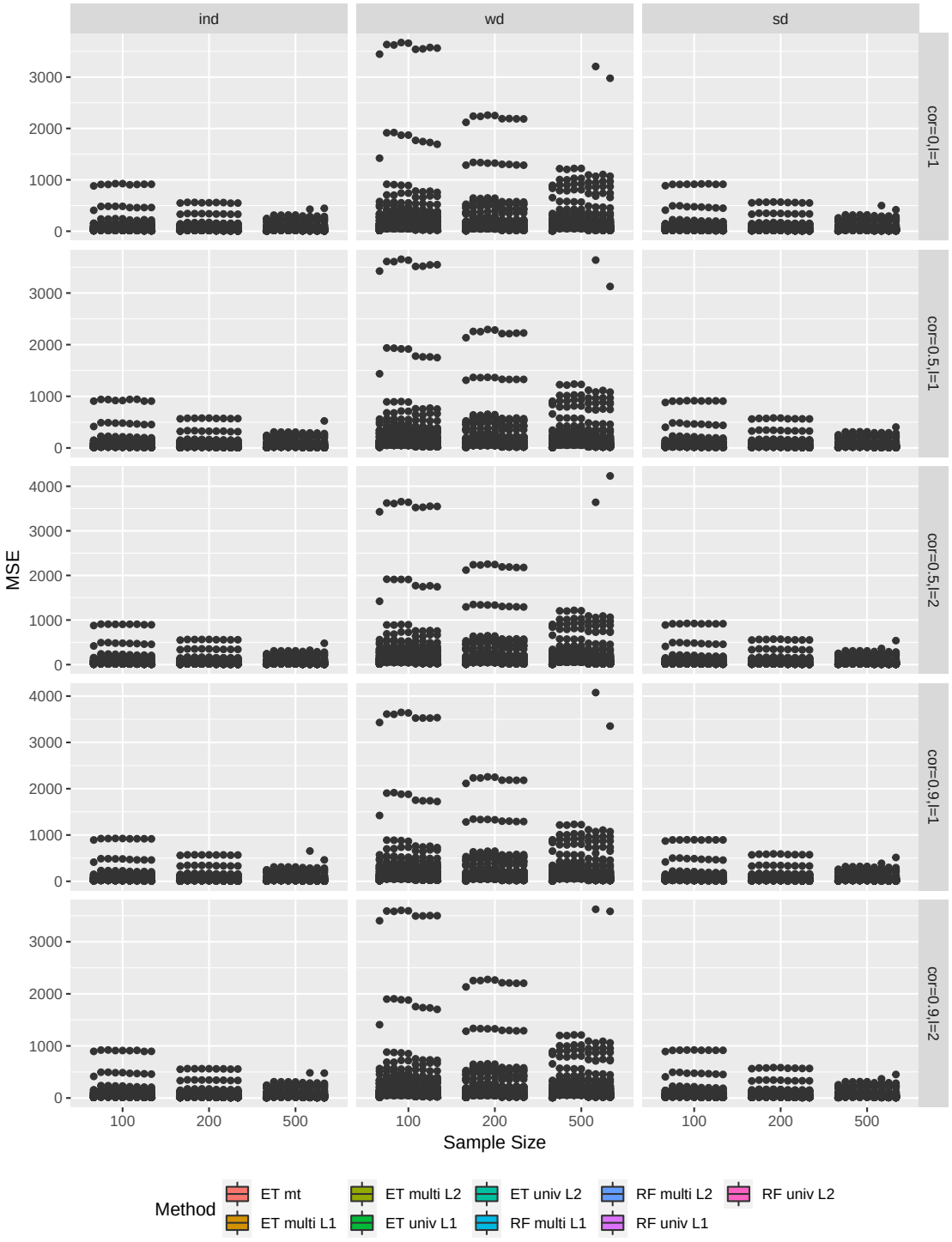


Figure A.16: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Linear Low setting. Each boxplots contains 1.000 MSE values.

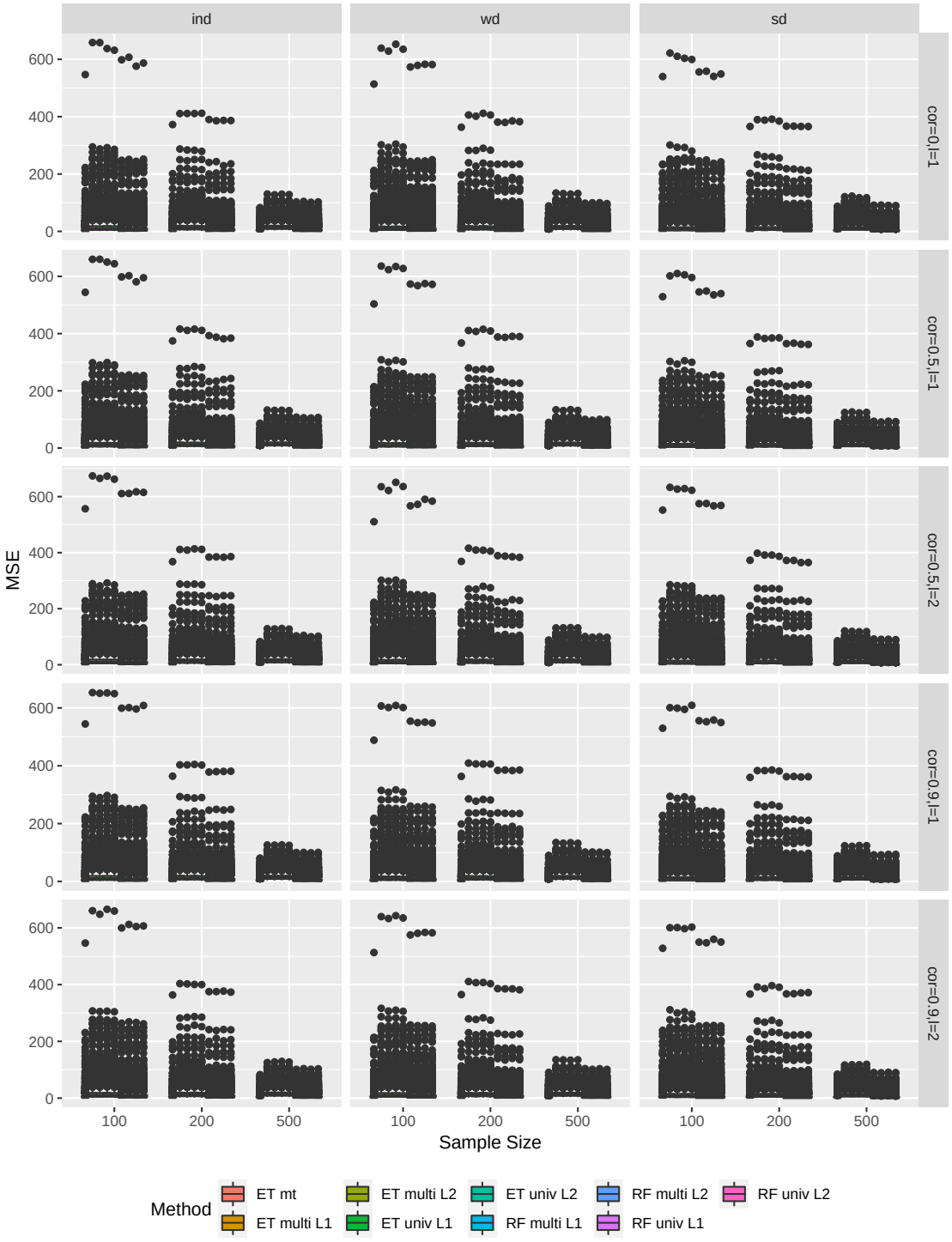


Figure A.17: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the MGAM1 High setting. Each boxplots contains 1.000 MSE values.

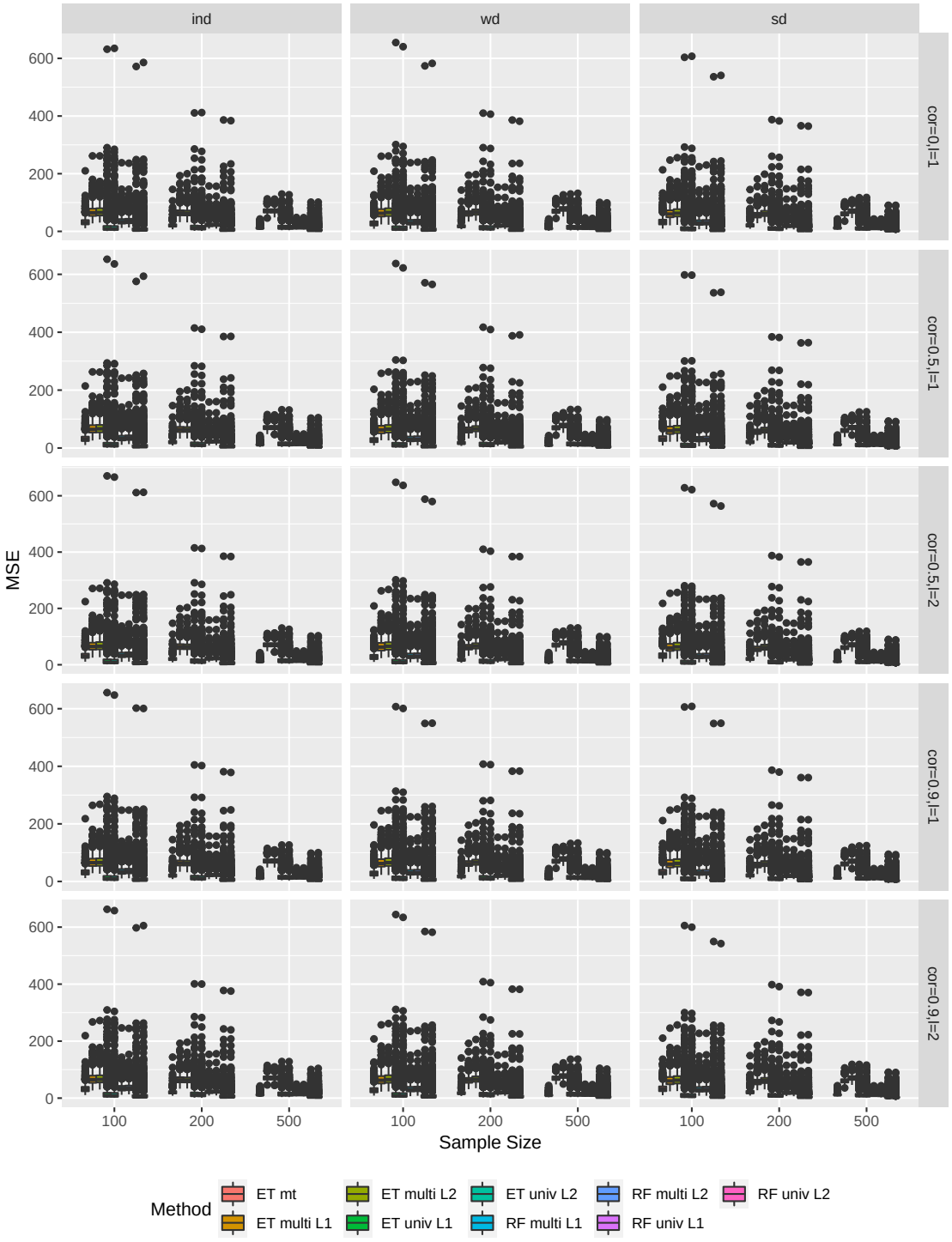


Figure A.18: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the MGAM1 Low setting. Each boxplots contains 1.000 MSE values.

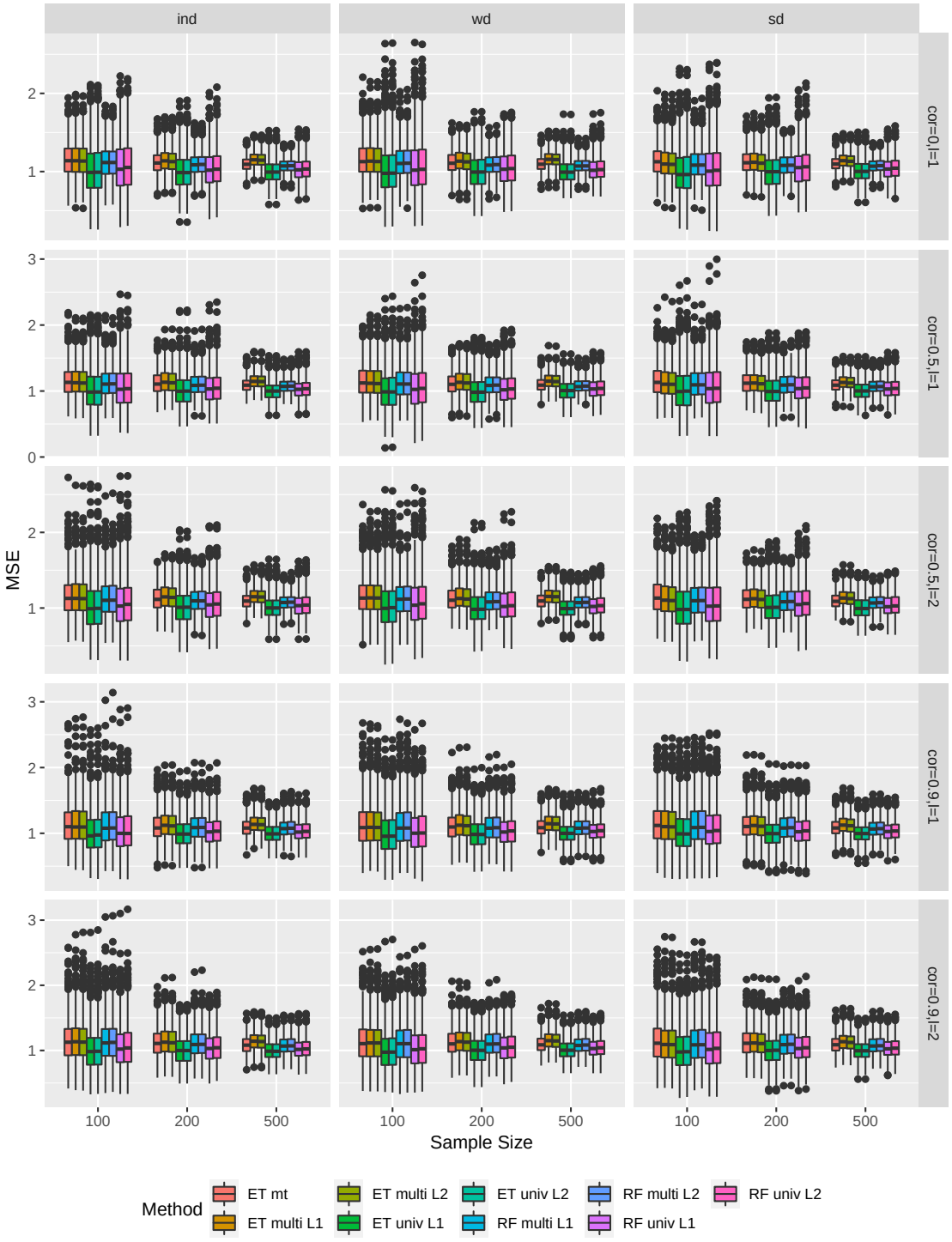


Figure A.19: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the MGAM2 setting. Each boxplots contains 1.000 MSE values.

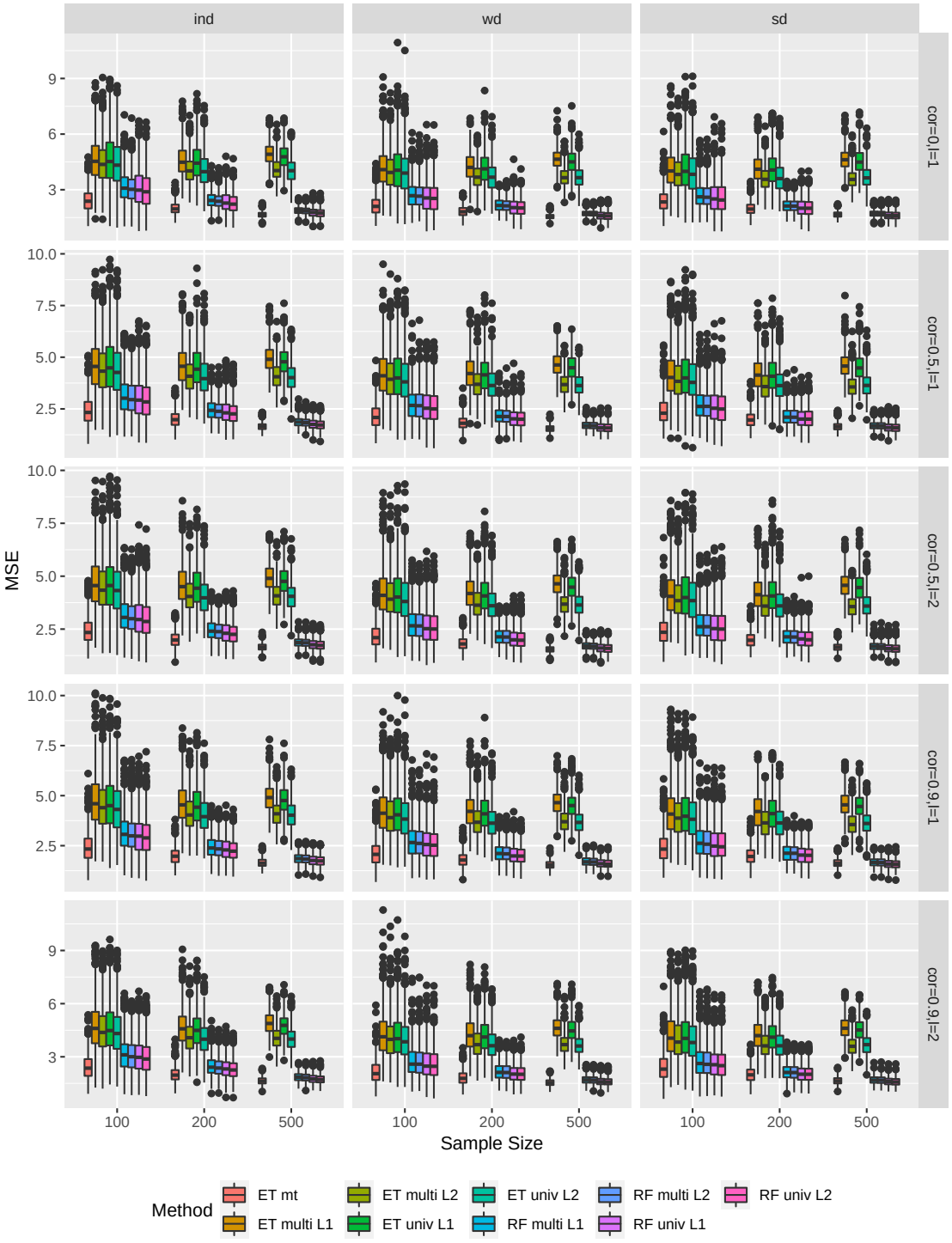


Figure A.20: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the MGAM3 setting. Each boxplots contains 1.000 MSE values.

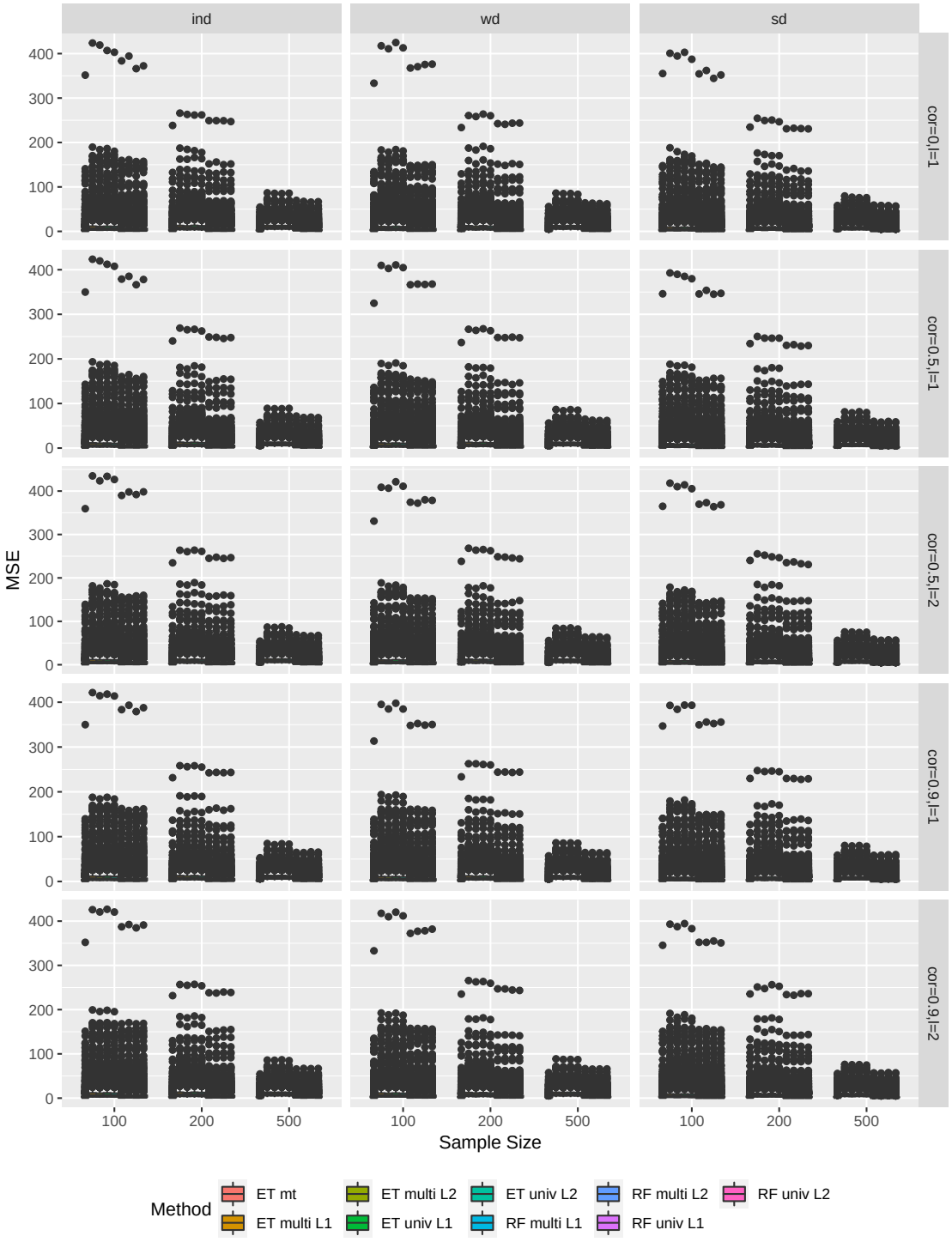


Figure A.21: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Quadradtic High setting. Each boxplots contains 1.000 MSE values.

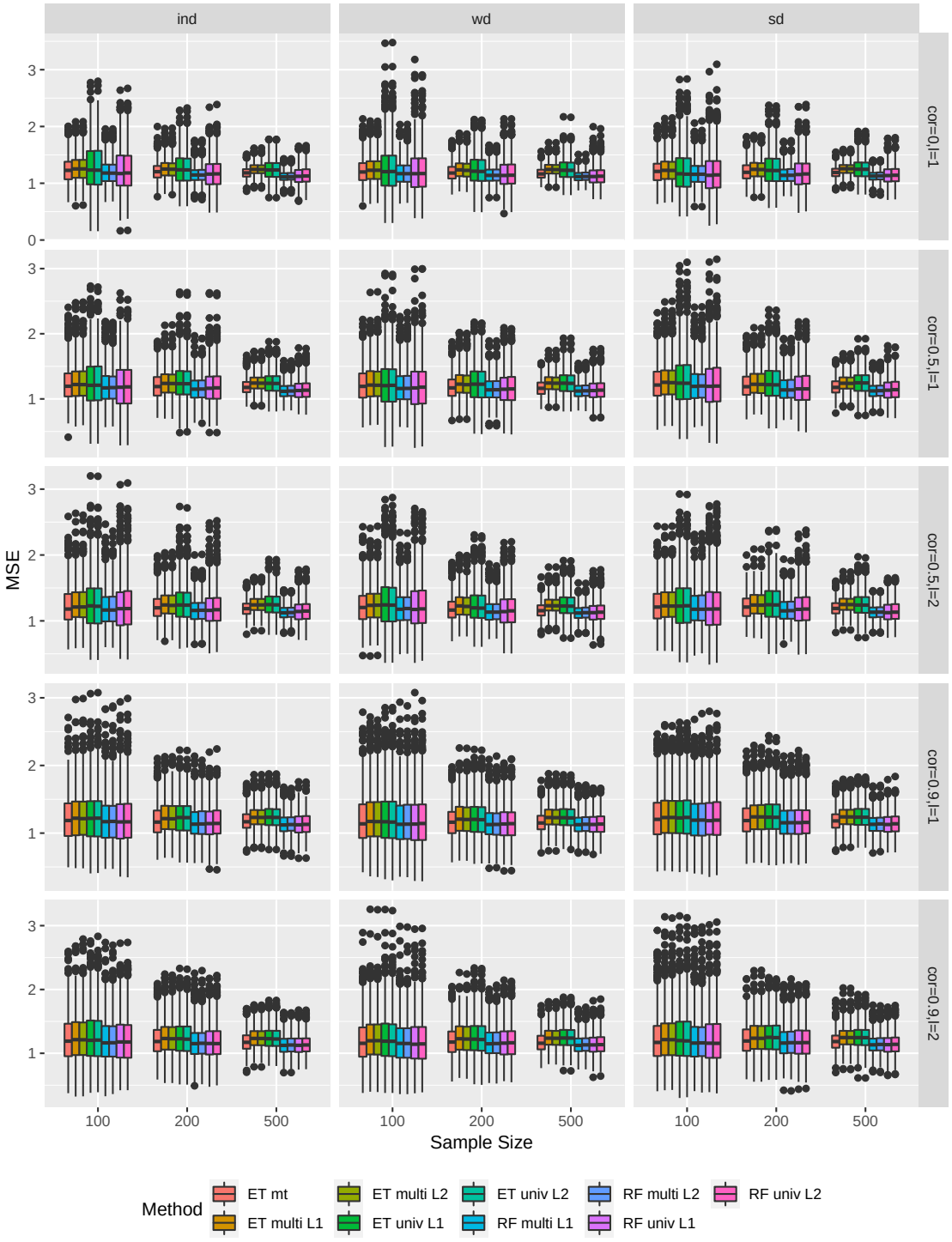


Figure A.22: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Rjump High setting. Each boxplots contains 1.000 MSE values.

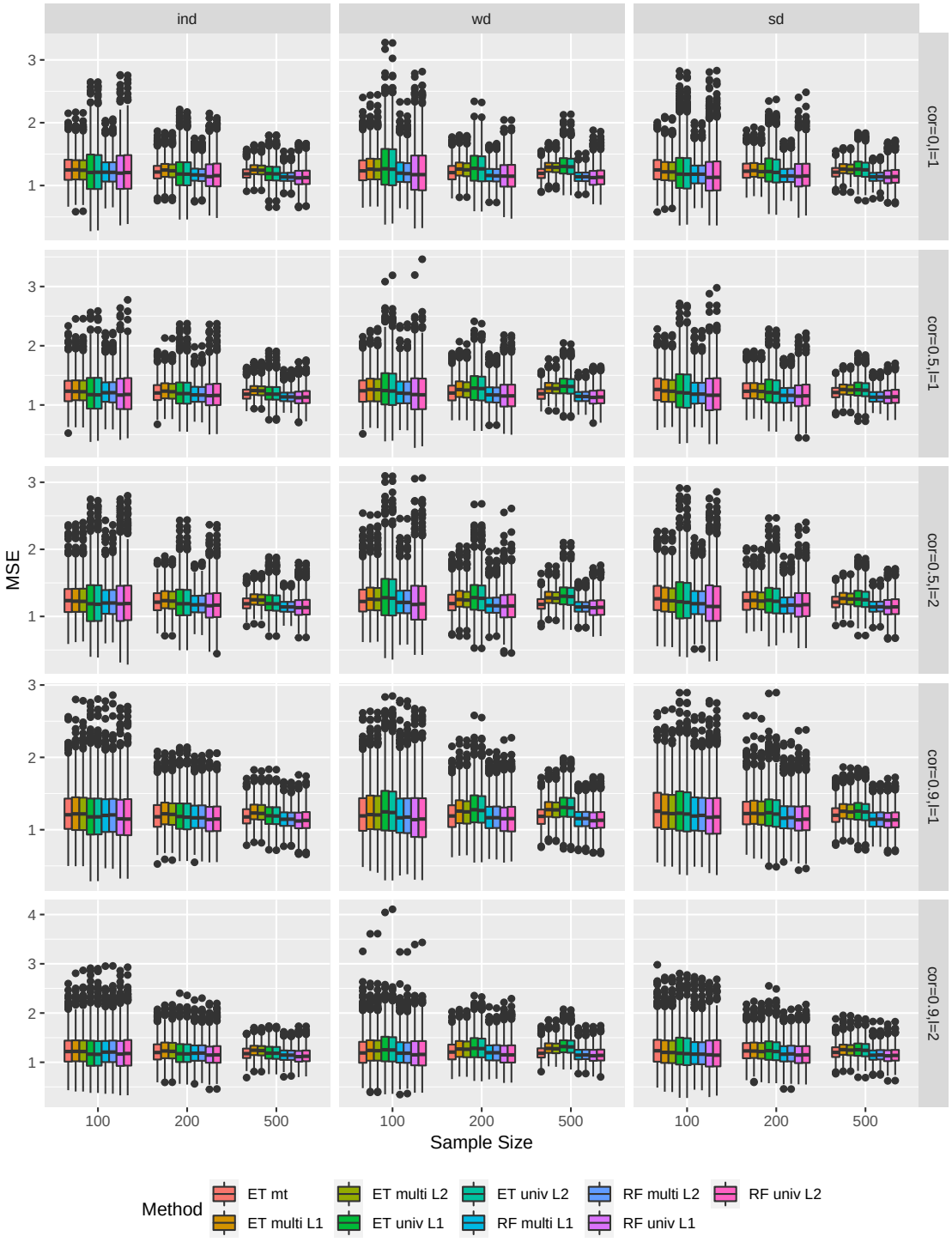


Figure A.23: Simulation results on the MSE for all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi L1 and L2), univariate Extra Trees (ET univ L1 and L2), multivariate Random Forest (RF multi L1 and L2) and univariate Random Forest (RF univ L1 and L2) in the Rjump Low setting. Each boxplots contains 1.000 MSE values.

*Appendix A.4. Runtime Analysis*

The results of the runtime analysis for the high settings and the MGAM settings are presented below. Since the run times of the approaches with L1 or L2 as the impurity measure are similar, the results with L2 as the impurity measure are shown. The runtimes for all different setups are given in Figures A.24 and A.25.

Studying the results in detail, we realize that the dependency structures of the features and the dependency structures of the output had little impact on the runtime. For ease of presentation, we therefore aggregated the runtime of the multiple simulation settings with regard to both dependency structures and the number of repetitions in Table A.5. It summarizes the mean runtime for each construction method separated by the sample sizes and the relationship between output and features. Since the multi-task method

| | $n = 100$ | | | | | $n = 200$ | | | | | $n = 500$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Setting | ET mt | ET multi | ET univ | RF multi | RF univ | ET mt | ET multi | ET univ | RF multi | RF univ | ET mt | ET multi | ET univ | RF multi | RF univ |
| Linear | 0.09 | 35.12 | 90.72 | 197.48 | 351.47 | 0.19 | 57.51 | 148.44 | 466.85 | 799.12 | 0.52 | 94.91 | 242.25 | 1464.91 | 2397.94 |
| Additive | 0.09 | 24.48 | 51.11 | 205.52 | 365.07 | 0.19 | 21.68 | 49.51 | 501.73 | 854.92 | 0.52 | 11.56 | 28.56 | 1511.66 | 2476.85 |
| Cross | 0.10 | 35.07 | 73.51 | 200.93 | 354.34 | 0.20 | 45.65 | 92.20 | 496.67 | 838.58 | 0.53 | 28.47 | 67.30 | 1574.91 | 2549.15 |
| Cubic | 0.10 | 34.59 | 70.13 | 202.80 | 354.38 | 0.20 | 43.17 | 83.76 | 512.42 | 852.50 | 0.53 | 21.75 | 53.67 | 1697.36 | 2709.18 |
| Jump 1 | 0.09 | 34.40 | 69.35 | 199.73 | 352.07 | 0.19 | 40.80 | 78.93 | 494.31 | 835.34 | 0.53 | 15.27 | 40.26 | 1603.17 | 2602.08 |
| MGAM1 | 0.09 | 31.47 | 79.42 | 207.01 | 361.96 | 0.20 | 42.71 | 106.84 | 515.78 | 862.85 | 0.53 | 40.20 | 99.86 | 1661.02 | 2639.98 |
| MGAM2 | 0.09 | 37.89 | 71.30 | 197.38 | 351.83 | 0.20 | 51.81 | 85.44 | 494.44 | 846.71 | 0.55 | 30.39 | 49.63 | 1541.62 | 2594.42 |
| MGAM3 | 0.09 | 35.77 | 86.17 | 185.40 | 332.39 | 0.19 | 62.06 | 145.50 | 447.06 | 773.65 | 0.51 | 117.51 | 264.01 | 1389.48 | 2302.53 |
| Quadratic | 0.09 | 31.98 | 76.54 | 204.49 | 356.83 | 0.19 | 42.24 | 98.86 | 509.71 | 851.53 | 0.52 | 33.67 | 79.25 | 1625.67 | 2595.54 |
| Rjump | 0.09 | 34.15 | 69.65 | 197.58 | 348.84 | 0.20 | 40.77 | 80.19 | 486.75 | 822.54 | 0.53 | 16.09 | 41.99 | 1572.21 | 2545.54 |

Table A.5: Average runtimes (in seconds) for the dependency models between the output and the features, aggregated over all dependency settings of the outputs and features and all repetitions for the methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi), univariate Extra Trees (ET univ), multivariate Random Forest (RF multi) and univariate Random Forest (RF univ).

was programmed runtime-efficiently in Java (Simm et al., 2014), unlike the other methods, this method is the fastest. Its average runtimes are faster than 0.6 seconds for all settings. In addition, the runtime behaves approximately linear to the sample size with about 0.1 seconds for $n = 100$ up to 0.6 seconds for $n = 500$.

Considering our implementations, we first note that the Extra Trees algorithms are faster than the Random Forest algorithms. The multivariate Random Forest requires on average (overall settings and sample sizes) 25 times as long as the multivariate Extra Trees. In the univariate case, the Random Forest takes on average 18 times as long as the Extra Trees algorithm. This time difference is probably due to the different choices of split values in the methods. Extra Trees randomly selects the split values, while Random Forest uses an exhaustive search to find these values. More important for us is a comparison between the multivariate and the univariate approaches: the results show that in both cases the multivariate approaches require less runtime than the univariate approaches. In fact, the multivariate Extra Trees show a decrease in runtime between 38% and 62% compared to the univariate approach. Using the multivariate Random Forest approach decreases runtime by 37% to 44% compared to the corresponding univariate method. For both Random Forest approaches, average runtimes increase similarly with increasing sample size. When sample size doubles from 100 to 200, runtimes for both approaches increase by 127-153%, while runtimes increase by 190-231% when sample size changes from 200 to 500. The different relations between output and features have a small effect on runtimes, as runtimes for fixed sample sizes (compared to the fastest setting MGAM3) increase by 6-22% for multivariate Random Forests and 3-17% for univariate Random Forests.

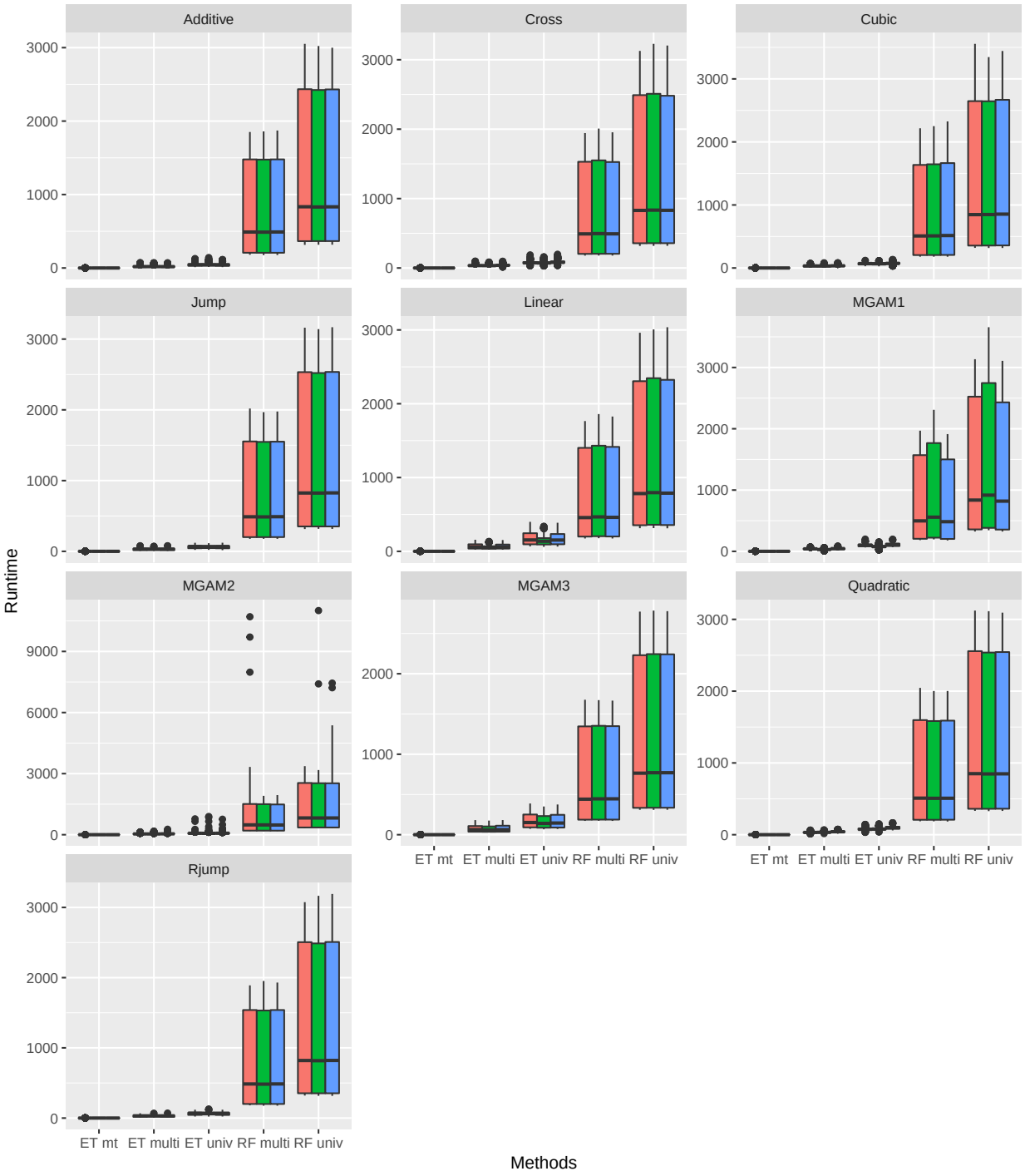


Figure A.24: Simulation results on the runtime for the feature dependence settings separated by the relation between output and feature and all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi), univariate Extra Trees (ET univ), multivariate Random Forest (RF multi) and univariate Random Forest (RF univ). The arranged boxplots in each method segment correspond to the following feature dependence structures: (from left to right) independent (red), weakly dependent (green), and strongly dependent (blue). Each boxplot contains 15,000 runtime measurements.

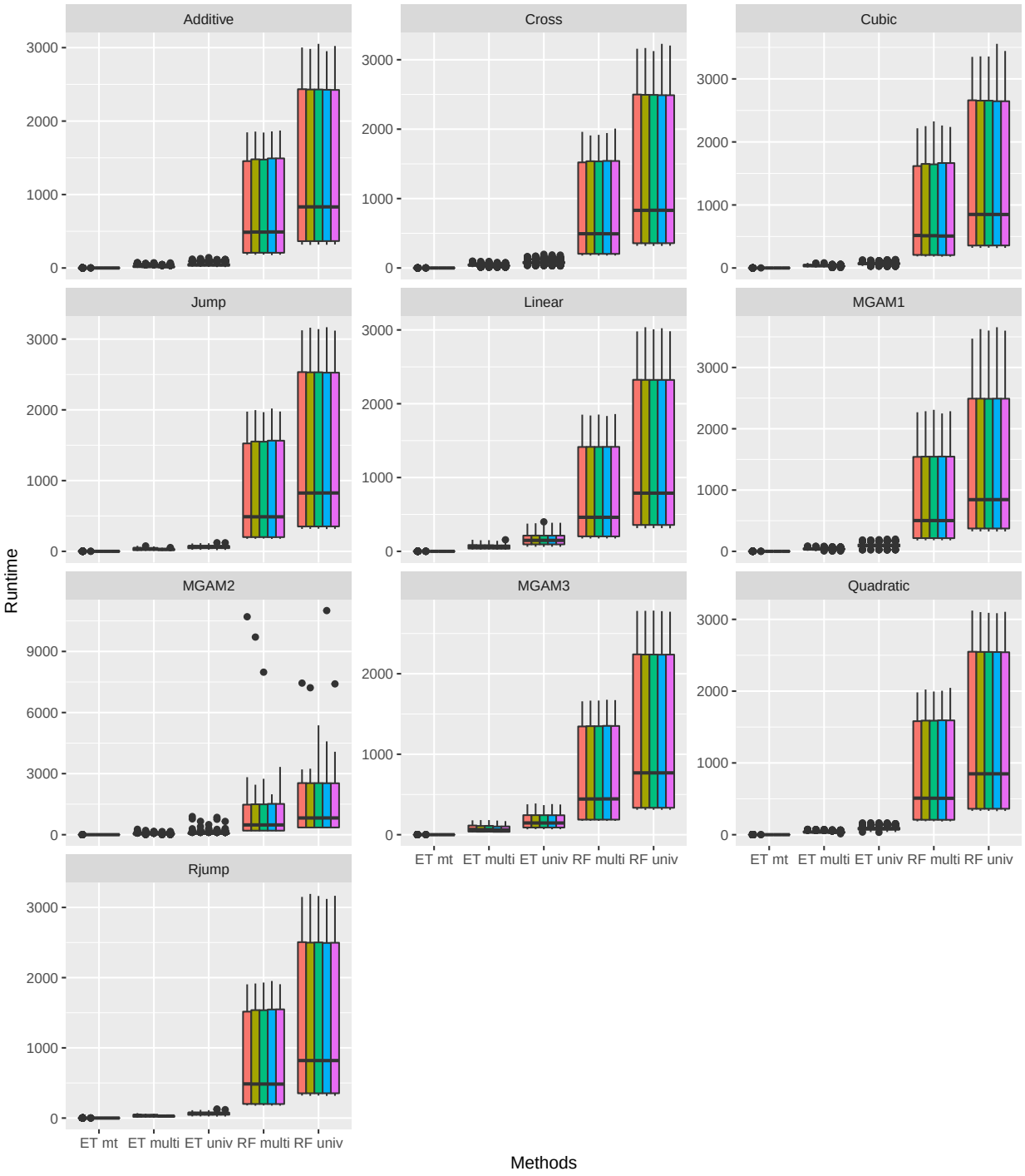


Figure A.25: Simulation results on the runtime for the output dependence settings ated by the relation between output and feature and all methods multi-task Extra Trees (ET mt), multivariate Extra Trees (ET multi), univariate Extra Trees (ET univ), multivariate Random Forest (RF multi) and univariate Random Forest (RF univ). The arranged boxplots in each method segment correspond to the following dependence structures: (from left to right) cor=0 and $\ell = 1$, cor=0.5 and $\ell = 1$, cor=0.5 and $\ell = 2$, cor=0.9 and $\ell = 1$ and cor=0.9 and $\ell = 2$. Each boxplot contains 9,000 runtime measurements.

A different observation can be made for the multivariate and univariate Extra Trees. When the sample sizes are increased, their runtime behaviour varies from setting to setting, see Figure A.26. For the settings

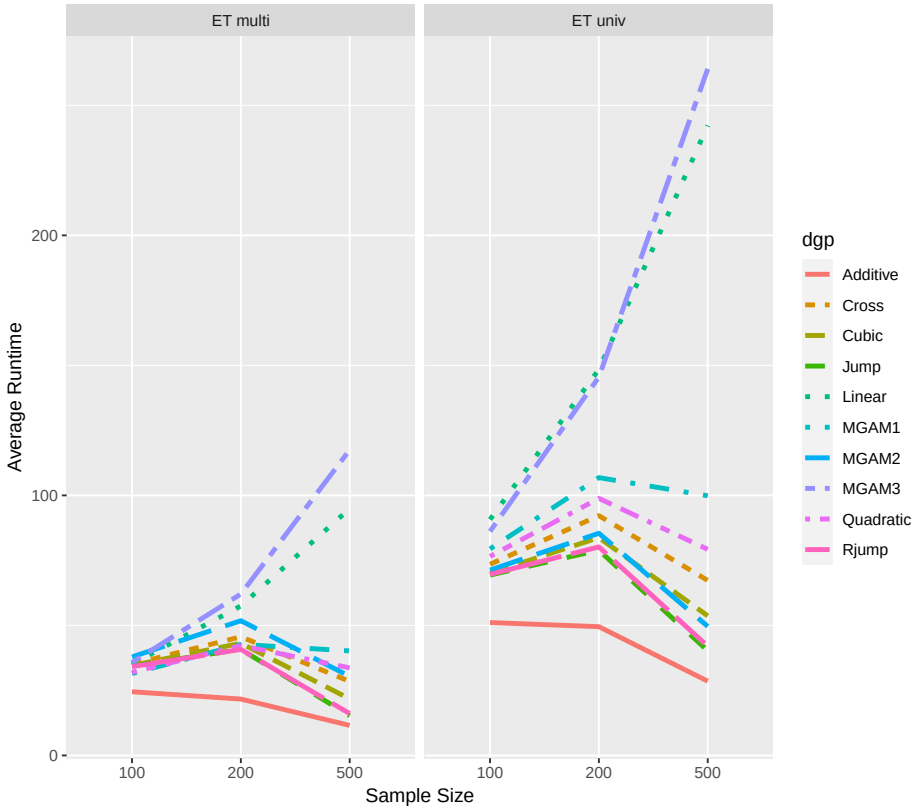


Figure A.26: Average runtimes for multivariate (left) and univariate Extra Trees (right) aggregated over all dependency settings of the outputs and features and all repetitions.

linear and MGAM3, the runtimes increase for increasing sample sizes. Note that these are also the settings with the largest runtimes for a fixed sample size. However, the setting additive has the fastest runtimes and decreases slightly with increasing sample size. The runtimes of the other settings increase by 13.81-36.75% when the sample size is doubled from 100 to 200, but decrease by 5.87-62.56% when increasing the sample size from 200 to 500.

## Appendix B. Illustrative Data Example Analysis

### *Appendix B.1. Runtime*

The results of the runtime analysis on all five data sets presented in Section 5 are reported in Table A.7. For each setting we repeated the runtime measurment 100 times.

The multi-task method is the fastest. Its average runtimes are faster than 0.23 seconds for all data sets. In addition, we obtain that the dimension of the output vector has an influence of the runtime, since for example the average runtimes in the stock data sets is almost twice as high as in the concrete data

| | univariate | | | | multivariate | | | | multi task |
|---|---|---|---|---|---|---|---|---|---|
| Data Set | RF univ L1 | RF univ L2 | ET univ L1 | ET univ L2 | RF multi L1 | RF multi L2 | ET multi L1 | ET multi L2 | ET mt |
| Concrete | 313.26 | 251.17 | 52.85 | 53.24 | 199.58 | 134.61 | 28.26 | 26.64 | 0.07 |
| Jura | 1641.15 | 1326.31 | 96.38 | 122.22 | 922.9 | 624.21 | 70.52 | 73.22 | 0.23 |
| Hspider | 128.44 | 129.84 | 54.928 | 67.32 | 12.87 | 11.92 | 15.65 | 10.99 | 0.07 |
| Stock | 279.70 | 262.90 | 67.15 | 74.36 | 106.93 | 77.28 | 29.81 | 23.94 | 0.13 |
| Enb | 489.50 | 475.55 | 94.02 | 141.97 | 311.26 | 275.58 | 55.90 | 76.86 | 0.12 |

Table B.6: Average runtimes of the construction methods presented in Section 2 across all data sets based on 100 replicates.

set, although the sample size in the concrete data set is about twice as large as in the stock. However, the dimension of the output vector in the stock data set is twice as large as that of the output vector in concrete.

As in the simulation study, we first note that the Extra Trees algorithms are faster than the Random Forest algorithms except on the hspider data set. Here, the multivariate Random Forest with L2 as impurity measure is on average one second faster than the Extra Tree counterpart.

Comparing the multivariate and the univariate approaches, we obtain as in the simulation study that the multivariate approaches require less runtime than the univariate approaches across all data sets. In fact, using the multivariate Random Forest approach in the hspider data set decreases runtime by about 90% compared to the corresponding univariate method. The multivariate Extra Trees show a decrease in runtime between 27% and 84% compared to the univariate approaches.While the Random Forest approaches with L2 as impurity measure are on average faster than the Random Forest approaches with L1 as impurity measure (except in the hspider data set the univariate approach), a different observation can made for the multivariate and univariate Extra Trees. Here, except in the hspider and stock data set the multivariate approaches, the method with L1 as impurity measure reduces the runtime.

*Appendix B.2. Real Data Performance*

| | Univariate | | | Multivariate | | |
|---|---|---|---|---|---|---|
| Method | 1st component | 2nd component | 3rd component | 1st component | 2nd component | 3rd component |
| RF L1 | 0.659 | 0.582 | 0.374 | 0.65 | 0.594 | 0.462 |
| RF L2 | 0.65 | 0.579 | 0.344 | 0.645 | 0.59 | 0.444 |
| ET L1 | 0.834 | 0.759 | 0.639 | 0.79 | 0.722 | 0.602 |
| ET L2 | 0.807 | 0.723 | 0.596 | 0.774 | 0.703 | 0.573 |

Table B.7: Average MSE for each component of the output vector of the construction methods presented in Section 2 using concrete data set based on 5-fold cross-validation and 100 replicates.

| | Univariate | | | | Multivariate | | | |
|---|---|---|---|---|---|---|---|---|
| Method | 1st com | 2nd com | 3rd com | 4th com | 1st com | 2nd com | 3rd comt | 4th com |
| RF L1 | 0.478 | 0.337 | 0.335 | 0.206 | 0.231 | 0.481 | 0.377 | 0.377 |
| RF L2 | 0.471 | 0.332 | 0.327 | 0.204 | 0.236 | 0.477 | 0.374 | 0.351 |
| ET L1 | 0.717 | 0.685 | 0.709 | 0.548 | 0.459 | 0.66 | 0.557 | 0.628 |
| ET L2 | 0.716 | 0.603 | 0.666 | 0.485 | 0.408 | 0.625 | 0.515 | 0.583 |

Table B.8: Average MSE for each component (com) of the output vector of the construction methods presented in Section 2 using jura data set based on 5-fold cross-validation and 100 replicates.

| | Univariate | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | 1st com | 2nd com | 3rd com | 4th com | 5th com | 6th com | 7th comt | 8th com | 9th com | 10th com | 11th com | 12th com |
| RF L1 | 0.547 | 0.865 | 0.362 | 0.935 | 0.808 | 0.622 | 0.925 | 0.739 | 0.724 | 0.439 | 0.329 | 0.735 |
| RF L2 | 0.539 | 0.868 | 0.348 | 0.902 | 0.775 | 0.596 | 0.927 | 0.745 | 0.723 | 0.377 | 0.615 | 0.72 |
| ET L1 | 0.648 | 0.928 | 0.519 | 0.996 | 0.934 | 0.847 | 0.884 | 0.8056 | 0.882 | 0.815 | 0.601 | 0.872 |
| ET L2 | 0.641 | 0.915 | 0.485 | 0.963 | 0.842 | 0.767 | 0.886 | 0.784 | 0.829 | 0.613 | 0.767 | 0.855 |
| | Multivariate | | | | | | | | | | | |
| Method | 1st com | 2nd com | 3rd com | 4th com | 5th com | 6th com | 7th comt | 8th com | 9th com | 10th com | 11th com | 12th com |
| RF L1 | 0.63 | 0.855 | 0.429 | 0.927 | 0.81 | 0.593 | 0.887 | 0.774 | 0.716 | 0.398 | 0.345 | 0.732 |
| RF L2 | 0.611 | 0.858 | 0.441 | 0.922 | 0.816 | 0.588 | 0893 | 0.761 | 0.712 | 0.404 | 0.348 | 0.721 |
| ET L1 | 0.671 | 0.91 | 0.505 | 0.963 | 0.849 | 0.781 | 0.89 | 0.783 | 0.834 | 0.633 | 0.591 | 0.859 |
| ET L2 | 0.67 | 0.908 | 0.509 | 0.96 | 0.851 | 0.775 | 0.892 | 0.785 | 0.830 | 0.629 | 0.585 | 0.854 |

Table B.9: Average MSE for each component (com) of the output vector of the construction methods presented in Section 2 using hspider data set based on 5-fold cross-validation and 100 replicates.

| | Univariate | | | | | | Multivariate | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | 1st com | 2nd com | 3rd com | 4th com | 5th com | 6th com | 1st comt | 2nd com | 3rd com | 4th com | 5th com | 6th com |
| RF L1 | 0.444 | 0.454 | 0.601 | 0.563 | 0.707 | 0.427 | 0.462 | 0.477 | 0.657 | 0.593 | 0.721 | 0.432 |
| RF L2 | 0.415 | 0.424 | 0.576 | 0.549 | 0.684 | 0.424 | 0.459 | 0.47 | 0.638 | 0.593 | 0.711 | 0.429 |
| ET L1 | 0.768 | 0.783 | 0.839 | 0.823 | 0.904 | 0.754 | 0.712 | 0.726 | 0.806 | 0.772 | 0.85 | 0.728 |
| ET L2 | 0.732 | 0.759 | 0.833 | 0.802 | 0.867 | 0.756 | 0.713 | 0.725 | 0.803 | 0.776 | 0.845 | 0.728 |

Table B.10: Average MSE for each component (com) of the output vector of the construction methods presented in Section 2 using stock data set based on 5-fold cross-validation and 100 replicates.

| | Univariate | | Multivariate | |
|---|---|---|---|---|
| Method | 1st component | 2nd component | 1st component | 2nd component |
| RF L1 | 0.023 | 0.047 | 0.024 | 0.044 |
| RF L2 | 0.02 | 0.043 | 0.02 | 0.042 |
| ET L1 | 0.119 | 0.13 | 0.122 | 0.123 |
| ET L2 | 0.085 | 0.1 | 0.089 | 0.094 |

Table B.11: Average MSE for each component of the output vector of the construction methods presented in Section 2 using enb data set based on 5-fold cross-validation and 100 replicates.

*Appendix B.3. Additional Experiments on the Concrete Data Set*

To compare the predictive accuracy of our approaches on the concrete data set to MVPART (De'ath, 2012), univariate and multivariate GUIDE (Loh and Zheng, 2013), we follow Loh and Zheng (2013) and apply leave-one-out cross-validation to estimate the sum of MSEs of the trees, where the sum is over the three output variables. In contrast to Loh and Zheng (2013), we do not prune the trees in our approaches. The results are shown in Table B.12. Note that we include the results of Loh and Zheng (2013) in the table

| Methods | RF univ L1 | RF univ L2 | RF multi L1 | RF multi L2 | MVPART | GUIDE uni |
|---|---|---|---|---|---|---|
| Sum of MSE | 1.517 | 1.495 | 1.598 | 1.581 | 2.096 | 1.957 |
| Methods | ET univ L1 | ET univ L2 | ET multi L1 | ET multi L2 | ET mt | GUIDE multi |
| Sum of MSE | 2.167 | 2.177 | 2.048 | 1.969 | 3.251 | 2.097 |

Table B.12: Sum of average MSEs of methods considered in Section 2 and Loh and Zheng (2013) using concrete data set based on leave-one-out crossvalidation.

without performing their experiments ourselves. While the multivariate Extra Trees outperformed their univariate method by decreasing the sum of MSE by 5.810% to 10.564%, the univariate RF and GUIDE

slightly decreased the sum of MSEs by 5.33% to 7.15%s compared with their multivariate approaches. In general, the Random Forest approaches have the smallest sum of MSEs ranging between from 1.495 to 1.598.