Measuring the Algorithmic Convergence of Randomized Ensembles: The Regression Setting*

Miles E. Lopes†, Suofei Wu†, and Thomas C. M. Lee†

Abstract. When randomized ensemble methods such as bagging and random forests are implemented, a basic question arises: Is the ensemble large enough? In particular, the practitioner desires a rigorous guarantee that a given ensemble will perform nearly as well as an ideal infinite ensemble (trained on the same data). The purpose of the current paper is to develop a bootstrap method for solving this problem in the context of regression — which complements our companion paper in the context of classification (Lopes, *Ann. Statist.*, 2019, 47(2), 1088–1112). In contrast to the classification setting, the current paper shows that theoretical guarantees for the proposed bootstrap can be established under much weaker assumptions. In addition, we illustrate the flexibility of the method by showing how it can be adapted to measure algorithmic convergence for variable selection. Lastly, we provide numerical results demonstrating that the method works well in a range of situations.

Key words. bagging, bootstrap, randomized algorithms, random forests

AMS subject classifications. 62F40, 65B05, 68W20, 60G25

1. Introduction.

Ensemble methods are a fundamental approach to prediction, based on the principle that accuracy can be enhanced by aggregating a diverse collection of prediction functions. Two of the most widely used methods in this class are *random forests* and *bagging*, which rely on randomization as a general way to diversify an ensemble [10, 11]. For these types of randomized ensembles, it is generally understood that the predictive accuracy improves and eventually stabilizes as the ensemble size becomes large. Likewise, in the theoretical analysis of randomized ensembles, it is common to focus on the ideal case of an infinite ensemble [14, 28, 6, 50, 5, 55]. However, in practice, the user does not know the true relationship between accuracy and ensemble size. As a result, it is difficult to know if a given ensemble is large enough so that its accuracy will nearly match the ideal level of an infinite ensemble.

Beyond these statistical considerations, the relationship between accuracy and ensemble size is important for computational reasons. Indeed, as an ensemble becomes larger, more resources are needed to train it, to store it in memory, and to make new predictions on unlabeled points — especially when large volumes of data are involved. Consequently, if it were possible for the user to know the true relationship between accuracy and ensemble size, it would be possible to do "just enough" computation to achieve a desired degree of convergence. Similarly, this would also make it possible to ensure that the amount of computation is *adaptive* to the unique data the user has at hand.

The purpose of the current paper is develop a solution to the problem of measuring algorithmic convergence for random forests, bagging, and related methods in the context of

*Submitted to the editors June 19, 2020

Funding: The first author was supported in part by NSF grants DMS 1613218 and 1915786, and the third author was supported in part by NSF grants DMS 1811405, 1811661, and 1916125.

†Department of Statistics, University of California, Davis (melopes@ucdavis.edu, swu@ucdavis.edu, tcmlee@ucdavis.edu).

regression. More specifically, we offer a bootstrap method for estimating how far the prediction error of a finite ensemble is from the ideal prediction error of an infinite ensemble (trained on the same data). To put this into perspective for the setting of regression, it is worth noting that a theoretically justified method for solving this problem has not previously been available. In this way, our work fills a significant gap in the literature by providing users with a more rigorous alternative to informal rules that are used in practice for selecting ensemble size. Furthermore, our approach is of broader conceptual interest, because it indicates new possibilities for applying bootstrap methods to randomized algorithms outside the scope of classical statistical inference (see subsection 1.3 for additional details).

In the remainder of the introduction, we give a precise description of the problem formulation in subsection 1.1, followed by a summary of related work and contributions in subsection 1.2 and subsection 1.3.

1.1. Background and setup. To fix some basic notation for the regression setting, let $\mathcal{D} = \{(X_j, Y_j)\}_{j=1}^n$ denote a set of training data in a space $\mathcal{X} \times \mathbb{R}$, where each Y_j is the scalar response variable associated to X_j, and the space $\mathcal{X}$ is arbitrary. In addition, for each $i = 1, \dots, t$, we write $T_i : \mathcal{X} \to \mathbb{R}$ to refer to the ith regression function in an ensemble of size t trained on $\mathcal{D}$.

Randomized regression ensembles. For the purpose of understanding our setup, it is helpful to quickly review the methods of bagging and random forests. The method of bagging works by generating random sets $\mathcal{D}_1^*, \dots, \mathcal{D}_t^*$, each of size n, by sampling with replacement from $\mathcal{D}$. Next, a standard "base" regression algorithm is used to train a regression function T_i on $\mathcal{D}_i^*$ for each $i = 1, \dots, t$. For instance, it is especially common to apply a decision tree algorithm like CART [12] to each set $\mathcal{D}_i^*$. In turn, future predictions are made by using the averaged regression function, which is defined for each $x \in \mathcal{X}$ by

$$\bar{T}_t(x) = \frac{1}{t} \sum_{i=1}^{t} T_i(x). \tag{1.1}$$

Much like bagging, the method of random forests uses sampling with replacement to generate the same type of random sets $\mathcal{D}_1^*, \dots, \mathcal{D}_t^*$. However, random forests adds an additional source of randomness. Namely, if the space $\mathcal{X}$ is (say) p-dimensional, and CART is the base regression algorithm, then random forests uses randomly chosen subsets of the p features when "split points" are selected for the CART regression trees. Apart from this distinction, random forests also uses the average (1.1) when making final predictions. A more detailed description may be found in [22].

In order to unify the methods of bagging and random forests within a common theoretical framework, our analysis will consider a more general class of randomized ensembles. This class consists of regression functions $T_1, \dots, T_t$ that can be represented in the abstract form

$$T_i(x) = \varphi(x; \mathcal{D}, \xi_i), \tag{1.2}$$

where $\xi_1, \dots, \xi_t$ are i.i.d. "randomizing parameters" generated independently of $\mathcal{D}$, and φ is a deterministic function that does not depend on n or t. In particular, the representation (1.2) implies that the random functions $T_1, \dots, T_t$ are conditionally i.i.d., given $\mathcal{D}$. To see why bagging is representable in this form, note that ξ_i can be viewed as a random vector that specifies which points in $\mathcal{D}$ are randomly sampled into $\mathcal{D}_i^*$. Similarly, in the case of random forests, each ξ_i encodes the points in $\mathcal{D}_i^*$, as well as randomly chosen sets of features used for training T_i. More generally, the representation (1.2) is relevant to other types of randomized ensembles, such as those based on random rotations [9], random projections [15], or posterior sampling [46, 17].

Algorithmic convergence. In our analysis of algorithmic convergence, we will focus on quantifying how the mean-squared error (MSE) of an ensemble behaves as the ensemble size t becomes large. To define this measure of error in more precise terms, let $\boldsymbol{\xi}_t := (\xi_1, \dots, \xi_t)$ denote the randomizing parameters of the ensemble, and let $\nu = \mathcal{L}(X, Y)$ denote the joint distribution of a test point $(X, Y) \in \mathcal{X} \times \mathbb{R}$, which is drawn independently of $\mathcal{D}$ and $\boldsymbol{\xi}_t$. Accordingly, we define

$$\text{(1.3)} \qquad \textsc{mse}_t \;=\; \int_{\mathcal{X}\times\mathbb{R}} \big(y - \bar{T}_t(x)\big)^2 d\nu(x,y) \;=\; \mathbb{E}\Big[(Y - \bar{T}_t(X))^2 \,\Big|\, \boldsymbol{\xi}_t, \mathcal{D}\Big],$$

where the expectation on the right is only over the test point (X, Y). In this definition, it is important to notice that $\textsc{mse}_t$ is a random variable that depends on both $\boldsymbol{\xi}_t$ and $\mathcal{D}$. However, due to the fact that the *algorithmic* fluctuations of $\textsc{mse}_t$ arise only from $\boldsymbol{\xi}_t$, we will view the set $\mathcal{D}$ as a fixed input to the training algorithm, and likewise, our analysis will always be conditional on $\mathcal{D}$. Indeed, the conditioning on $\mathcal{D}$ is motivated by the fact that the user would like to assess convergence for the particular set $\mathcal{D}$ that they actually have, and this viewpoint has been adopted in several other analyses of algorithmic convergence for randomized ensembles [46, 35, 53, 15, 36].

As a conceptual illustration, Figure 1 shows what algorithmic convergence looks like when random forests is applied to a fixed training set $\mathcal{D}$. In detail, the left panel displays values of the convergence gap $\textsc{mse}_t - \text{mse}_\infty$ as decision trees are added during a single run of random forests, from $t = 1$ up to $t = 2{,}000$, where mse_∞ denotes the limit of $\textsc{mse}_t$ as $t \to \infty$. If this entire process is repeated by running random forests many more times on the same set $\mathcal{D}$, then the result is a large collection of overlapping sample paths, as shown in the right panel of Figure 1. (Note also that none of these sample paths are observable in practice, and that the figure is given only for illustration.)

From a practical standpoint, the user would like to know the size of the convergence gap $\textsc{mse}_t - \text{mse}_\infty$ as a function of t. For this purpose, it is useful to consider the $(1-\alpha)$-quantile of the random variable $\textsc{mse}_t - \text{mse}_\infty$, which is defined for any $\alpha \in (0,1)$ by

$$q_{1-\alpha}(t) \;=\; \inf\Big\{q \in \mathbb{R} \,\Big|\, \mathbb{P}\big(\textsc{mse}_t - \text{mse}_\infty \le q \,\big|\, \mathcal{D}\big) \;\ge\; 1-\alpha\Big\}.$$

In other words, the value $q_{1-\alpha}(t)$ is the *tightest possible* upper bound on the gap that holds with probability at least $1 - \alpha$, conditionally on the set $\mathcal{D}$. This interpretation of $q_{1-\alpha}(t)$ can also be understood from the right panel of Figure 1, where we have plotted $q_{1-\alpha}(t)$ in gray, with $\alpha = 1/10$.

The problem to be solved. Although it is clear that the quantile $q_{1-\alpha}(t)$ represents a precise measure of algorithmic convergence, this function is unknown in practice. This leads to the problem of estimating $q_{1-\alpha}(t)$, which we propose to solve.

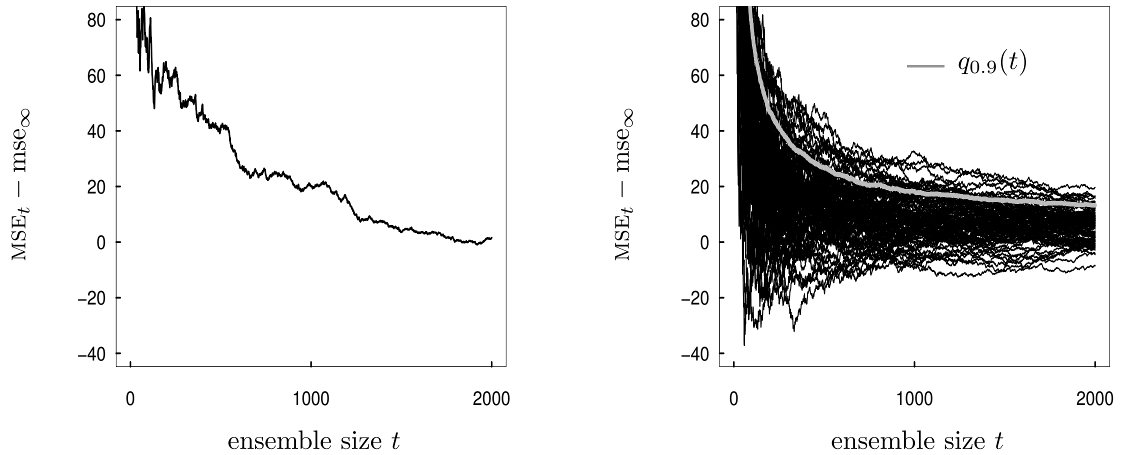

Figure 1: Left Panel: A sample path of $\text{MSE}_t - \text{mse}_\infty$ over a single run of random forests on the 'Housing data' described in section 5. Right Panel: Many sample paths of $\text{MSE}_t - \text{mse}_\infty$, with the 90% quantile $q_{.90}(t)$ overlaid in gray. (The curves in these panels are not observable to the user.)

Beyond the fact that $q_{1-\alpha}(t)$ is unknown, it is also important to keep in mind that estimating $q_{1-\alpha}(t)$ involves some additional constraints. First, the user would like to be able to assess convergence from the output a *single* run of the ensemble method. However, at first sight, it is not obvious that the output of a single run provides enough information to successfully estimate $q_{1-\alpha}(t)$. Second, the method for estimating $q_{1-\alpha}(t)$ should be computationally inexpensive, so that the cost of checking convergence is manageable in comparison to the cost of training the ensemble itself. Accordingly, we will show that the proposed method is able to handle both of these constraints in section 2 and section 4 respectively.

1.2. Related work. The general problem of measuring the algorithmic convergence of randomized ensembles has attracted sustained interest over the past two decades. For instance, there have been numerous empirical studies of algorithmic convergence for both classification and regression (e.g. [31, 2, 52, 48, 49]).

With regard to the theoretical analysis of convergence, we will now review the existing results for classification and regression separately. In the setting of classification, much of the literature has studied convergence in terms of the misclassification probability for majority voting, denoted ERR_t (a counterpart of MSE_t), which is viewed as a random variable that depends on $\boldsymbol{\xi}_t$ and $\mathcal{D}$. For this measure of error, the convergence of $\mathbb{E}[\text{ERR}_t|\mathcal{D}]$ and $\text{var}(\text{ERR}_t|\mathcal{D})$ as $t \to \infty$ has been analyzed in the papers [46, 35, 15], which have developed asymptotic formulas for $\mathbb{E}[\text{ERR}_t|\mathcal{D}]$, as well as bounds for $\text{var}(\text{ERR}_t|\mathcal{D})$. Related results for a different measure of error can also be found in [29]. More recently, our companion paper [36] has developed a bootstrap method for measuring the convergence of ERR_t, which is able to circumvent some of the limitations of formula-based results.

In the setting of regression, algorithmic convergence results on MSE_t are scarce in comparison to those for ERR_t. Instead, much more attention in the regression literature has focused on how the size of t influences the variance of point predictions $\bar{T}_t(x)$, with $x \in \mathcal{X}$ held fixed, e.g. [56, 1, 62, 43, 53]. To the best of our knowledge, the only paper that has analyzed algorithmic convergence in terms of a prediction error measure is [53], which considers the risk $r_t := \mathbb{E}[(\bar{T}_t(X) - \mu(X))^2]$, where $\mu(x) := \mathbb{E}[Y|X = x]$ is the true regression function, and the expectation in the definition of r_t is over all of the objects $(X, \mathcal{D}, \boldsymbol{\xi}_t)$. In particular, the paper [53] develops an elegant theoretical bound on the gap between r_t and the risk of an infinite ensemble, denoted r_∞. Under the assumption of a Gaussian regression model with $\mathcal{X} = [0,1]^p$, this bound has the form

$$r_t - r_\infty \;\leq\; \tfrac{8}{t}\Big(\|\mu\|_\infty^2 + \sigma^2(1 + 4\log(n))\Big), \tag{1.4}$$

where $\sigma^2 = \text{var}(Y)$, and $\|\mu\|_\infty = \sup_{x\in\mathcal{X}} |\mu(x)|$. However, due to the fact that the parameters σ and $\|\mu\|_\infty$ are unknown, and that $\|\mu\|_\infty$ is inherently conservative, this bound does not lend itself to a practical method for measuring convergence, and is primarily of theoretical interest.

1.3. Contributions.

Methodology. From a methodological standpoint, the approach taken here differs in several ways from previous works in the regression setting. Most notably, our work looks at algorithmic convergence in terms of an error measure that is conditional on $\mathcal{D}$. (For instance, this differs from the analysis of r_t mentioned above, which averages over $\mathcal{D}$.) In more concrete terms, we will provide a quantile estimate $\hat{q}_{1-\alpha}(t)$, such that the bound

$$\text{MSE}_t - \text{mse}_\infty \;\leq\; \hat{q}_{1-\alpha}(t)$$

holds with a probability that is nearly $1-\alpha$ or larger, conditionally on $\mathcal{D}$. This conditioning is especially important from the viewpoint of the user, who is typically interested in algorithmic convergence with respect to the *actual dataset at hand*. Another distinct feature of our method is that it provides the user with a *direct numerical estimate of convergence*, whereas formula-based results are more likely to depend on specialized models, involve conservative constants, or depend on unknown parameters, such as in the bound (1.4).

In addition, the scope of the proposed method goes beyond MSE_t, and in subsection 2.3 we will show how the bootstrap method is flexible enough that it can also be applied to variable selection. In this context, the ensemble provides a ranking of variables according to an "importance measure", and this ranking typically stabilizes as $t \to \infty$. However, the notion of convergence is somewhat subtle, because the importance measure for some variables may converge more slowly than for others — *which can distort the overall ranking of variables.* As far as we know, this issue has not be addressed in the literature, and the method proposed in subsection 2.3 provides a way to check that convergence has been achieved uniformly variables, so that they can be compared fairly.

Theory. From a theoretical standpoint, the most important aspect of our work is that it establishes consistency guarantees for the proposed methods under very mild assumptions. To place our assumptions into context, it should be emphasized that most analyses of randomized ensembles deal with specialized types of prediction functions $T_1, \dots, T_t$ that are simpler than the ones used in practice, e.g. [34, 1, 6, 5, 55, 53, 54, 36]. By contrast, our current results for regression only rely on (1.2) and basic moment assumptions (to be detailed in section 3). In particular, the crucial ingredient that enables us to handle general types of prediction functions in our main result (Theorem 3.1) is a version of Rosenthal's inequality due to Talagrand [61], which is applicable to sums of independent Banach-valued random variables. Moreover, this allows our analysis to be fully *non-asymptotic*.

To make a more direct comparison with the main theoretical result in our previous paper in the classification setting [36], there are three points to highlight. First, the previous analysis requires that the classifier functions, say $Q_1, \dots, Q_t$, have a particular form, which is not generally satisfied by the decision tree classifiers in random forests — whereas our current theory is applicable to *actual* random forests. Second, if we let $\omega(x) = \mathbb{E}[Q_1(x)|\mathcal{D}]$ for any fixed $x \in \mathcal{X}$, then the previous analysis assumes that the distribution $\mathcal{L}(\omega(X)|\mathcal{D})$ has a continuously differentiable density function, while the current analysis involves no analogue of this condition. Third, the previous result on bootstrap consistency is stated in terms of a distributional limit, and does not provide a rate of convergence. Instead, our current result avoids the reliance on such a limit, and gives a more quantitative description of coverage probability.

Links between inference and computation. Traditionally, bootstrap methods have been viewed by statisticians as a way to use computation in the service of inference. For this reason, it should be emphasized that our work looks at bootstrap methods from a *reciprocal perspective*, since we aim to use inference in the service of computation (viz. using a quantile estimate to measure algorithmic convergence).

More generally, this way of looking at bootstrap methods has the potential to be applied to the convergence analysis of other randomized algorithms. For instance, in the growing field of randomized numerical linear algebra (or "matrix sketching"), it turns out that convergence can often be framed in terms of the quantiles of certain error variables. Some specific examples include randomized algorithms for matrix multiplication, least-squares, and singular value decomposition [39, 40, 37], and we refer to the recent survey [42, pp.14-18] for related discussion of the potential of bootstrap methods in this context. In addition, several variants of bootstrap methods have attracted interest as a way to assess the quality of solutions obtained from stochastic gradient descent (SGD) algorithms [21, 32, 60, 20]. Likewise, given the rising use of randomized algorithms in data science, it seems that considerable opportunity remains for developing bootstrap methods along these lines.

Outline. The remainder of the paper is organized as follows. The proposed methods are described in section 2, and our theoretical results on bootstrap consistency are presented in section 3. Next, computational cost is assessed in section 4, and numerical experiments are given in section 5. Finally, all proofs are given in the supplementary material.

2. Methodology.

Below, we present our core method for measuring algorithmic convergence with respect to MSE_t in subsection 2.1. Later on, we show how this approach can be extended to measuring convergence with respect to variable importance in subsection 2.3.

2.1. Measuring convergence with respect to mean-squared error.

The intuition for the proposed method is based on two main considerations. First, the definition of MSE_t in equation (1.3) shows that it can be interpreted as a functional of $\bar{T}_t$. Namely, if we let $f : \mathcal{X} \to \mathbb{R}$ denote a generic function, then we define the functional ψ according to

$$\psi(f) = \int_{\mathcal{X}\times\mathbb{R}} (y - f(x))^2 d\nu(x, y), \tag{2.1}$$

and it follows that MSE_t can be written as

$$\text{MSE}_t = \psi(\bar{T}_t). \tag{2.2}$$

Second, it is a general principle that bootstrap methods are well-suited to approximating distributions derived from smooth functionals of sample averages — which is precisely what the representation (2.2) entails.

To make a more detailed connection between these general ideas and the problem of estimating $q_{1-\alpha}(t)$, recall that we aim to approximate the distribution of the gap $\text{MSE}_t - \text{mse}_\infty$, rather than just MSE_t itself. Fortunately, the limiting value mse_∞ can be linked with ψ through the function ϑ defined by

$$\vartheta(x) = \mathbb{E}[\bar{T}_t(x)|\mathcal{D}], \tag{2.3}$$

where the expectation is only over the algorithmic randomness in $\bar{T}_t$ (i.e. over the random vector $\boldsymbol{\xi}_t$). More specifically, when the functions $T_1, \dots, T_t$ satisfy the representation (1.2), the law of large numbers implies $\text{mse}_\infty = \psi(\vartheta)$ under basic integrability assumptions, which leads to the relation

$$\text{MSE}_t - \text{mse}_\infty = \psi(\bar{T}_t) - \psi(\vartheta). \tag{2.4}$$

This relation is the technical foundation for the proposed method, since it suggests that in order to mimic the fluctuations of $\text{MSE}_t - \text{mse}_\infty$, we can develop a bootstrap method by viewing the functions $T_1, \dots, T_t$ as "observations", and viewing $\bar{T}_t$ as an estimator of ϑ. In other words, if we sample t functions $T_1^*, \dots, T_t^*$ with replacement from $T_1, \dots, T_t$, then we can formally define a bootstrap sample of $\text{MSE}_t - \text{mse}_\infty$ according to

$$\text{MSE}_t^* - \text{MSE}_t = \psi(\bar{T}_t^*) - \psi(\bar{T}_t), \tag{2.5}$$

where $\bar{T}_t^* := \frac{1}{t}\sum_{i=1}^t T_i^*$. In turn, after generating a collection of such bootstrap samples, we can use their empirical $(1-\alpha)$-quantile as an estimate of $q_{1-\alpha}(t)$. However, as a technical point, it should be noted that (2.5) is a "theoretical" bootstrap sample of $\text{MSE}_t - \text{mse}_\infty$, because the functional ψ depends on the unknown distribution of the test point $\mathcal{L}(X, Y)$. Nevertheless, the same reasoning can still be applied by replacing ψ with an estimate $\hat{\psi}$, which will be explained in detail later in this subsection. Altogether, the method is summarized by the following algorithm.

2.2. Using hold-out or out-of-bag samples. To complete our discussion of Algorithm 2.1, it remains to clarify how the functional ψ can be estimated from either hold-out samples, or so-called "out-of-bag" (OOB) samples. With regard to the first case, suppose a set of m labeled

Algorithm 2.1 Bootstrap method for estimating $q_{1-\alpha}(t)$

For $b = 1, \dots, B$ **:**

- Sample t functions $T_1^*, \dots, T_t^*$ with replacement from $T_1, \dots, T_t$.
- Compute the bootstrap sample $z_{t,b} := \hat{\psi}(\bar{T}_t^*) - \hat{\psi}(\bar{T}_t)$.

Return: the empirical $(1-\alpha)$-quantile of $z_{t,1}, \dots, z_{t,B}$ to estimate $q_{1-\alpha}(t)$.

samples $\tilde{\mathcal{D}} = \{(\tilde{X}_1, \tilde{Y}_1), \dots, (\tilde{X}_m, \tilde{Y}_m)\}$ has been held out from the training set $\mathcal{D}$. Using this set, the estimate $\hat{\psi}(\bar{T}_t)$ in Algorithm 2.1 can be easily obtained as

$$\hat{\psi}(\bar{T}_t) \;=\; \frac{1}{m}\sum_{j=1}^{m}(\tilde{Y}_j - \bar{T}_t(\tilde{X}_j))^2. \tag{2.6}$$

Analogously, we may also obtain $\hat{\psi}(\bar{T}_t^*)$ by using $\bar{T}_t^*$ instead of $\bar{T}_t$ in the formula above.

If the regression functions $T_1, \dots, T_t$ are trained via bagging or random forests, it is possible to avoid the use of a hold-out set by taking advantage of OOB samples, which are a unique attribute of these methods. To define the notion of an OOB sample, recall that these methods train each function T_i using a random set $\mathcal{D}_i^*$ obtained from $\mathcal{D}$ by sampling with replacement. Due to this sampling mechanism, it follows that each set $\mathcal{D}_i^*$ is likely to exclude approximately $(1-\frac{1}{n})^n \approx 37\%$ of the training points in $\mathcal{D}$. So, as a matter of terminology, if a particular training point X_j does not appear in $\mathcal{D}_i^*$, we say that X_j is "out-of-bag" for the function T_i. Also, we write $\textsc{oob}(X_j) \subset \{1, \dots, t\}$ to denote the index set corresponding to the functions for which X_j is OOB.

From a statistical point of view, OOB samples are important because they serve as "effective" hold-out points. (That is, if X_j is OOB for T_i, then the function T_i "never touched" the point X_j during the training process.) Hence, it is natural to consider the following alternative estimate of ψ based on OOB samples,

$$\hat{\psi}_{\text{O}}(\bar{T}_t) \;=\; \frac{1}{n}\sum_{j=1}^{n}(Y_j - \bar{T}_{t,\text{O}}(X_j))^2, \tag{2.7}$$

where we define $\bar{T}_{t,\text{O}}(X_j)$ to be the average over the functions for which X_j is OOB,

$$\bar{T}_{t,\text{O}}(X_j) = \tfrac{1}{|\textsc{oob}(X_j)|}\sum_{i\in\textsc{oob}(X_j)} T_i(X_j),$$

and $|\cdot|$ refers to the cardinality of a set. Similarly, the quantity $\hat{\psi}_{\text{O}}(\bar{T}_t^*)$ may be defined in terms of a corresponding average with the functions $T_1^*, \dots, T_t^*$. Lastly, in the case when $\textsc{oob}(X_j)$ is empty, we arbitrarily define $\bar{T}_{t,\text{O}}(X_j) = Y_j$, but this occurs very rarely. In fact, it can be checked that for a given point X_j, the set $\textsc{oob}(X_j)$ is empty with probability approximately equal to $(0.63)^t$.

2.3. Measuring convergence with respect to variable importance. In addition to their broad application in prediction problems, randomized ensembles have been very popular for the task of variable selection, e.g. [18, 58, 30, 23, 41, 24, 27]. Although a variety of procedures have been proposed for variable selection in this context, they are generally based on a common approach of ranking the variables according to a measure of averaged variable importance (VI). Under this approach, the averaged VI assigned to each variable typically converges to a limiting value as the ensemble becomes large. However, in practice, the user does not know how this convergence depends on the ensemble size — much like we have seen already for MSE_t.

Uniform convergence across variables. Before moving on to the details of our extended method, it is worth mentioning an extra subtlety of measuring algorithmic convergence for VI. Specifically, we must keep in mind that because variable selection is based on ranking, it is important that algorithmic convergence is reached uniformly across variables. In other words, if the VI for some variables converges more slowly than for others, then the ranking of variables will be distorted by purely algorithmic effects. Motivated by this issue, our extended method will provide a way to ensure that algorithmic convergence is achieved in a uniform sense. As an illustration of this point, Figure 2 shows how uniform convergence of VI across several variables can differ considerably from the convergence of VI for a single variable.

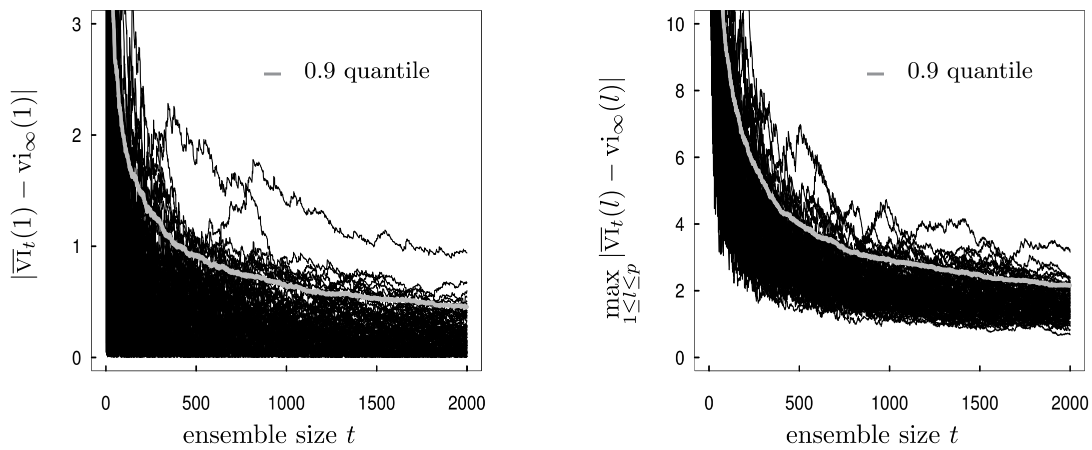

Figure 2: Left Panel: 1,000 sample paths of $|\overline{\mathrm{VI}}_t(1) - \mathrm{vi}_\infty(1)|$, with the true 0.9 quantile curve in gray. Right Panel: 1,000 sample paths of the variable $\max_{1\leq l\leq p} |\mathrm{VI}_t(l) - \mathrm{vi}_\infty(l)|$, with the true 0.9 quantile curve in gray. Both panels were obtained from the Music dataset described in section 5.

Setup for variable importance. To describe algorithmic convergence for VI in detail, let $T_1, \dots, T_t$ be a randomized ensemble that satisfies the representation (1.2), and consider a situation where the space $\mathcal{X}$ is p-dimensional. Also, suppose that for each function T_i, we have a rule for computing an associated value, say $\mathrm{VI}_i(l)$, to each variable $l \in \{1, \dots, p\}$. (Note that since T_i is a random function, it follows that $\mathrm{VI}_i(l)$ is random as well.) Likewise, the vector of such values associated with T_i is denoted $\mathrm{VI}_i = (\mathrm{VI}_i(1), \dots, \mathrm{VI}_i(p))$, and the average over $i = 1, \dots, t$ is denoted as

$$\overline{\text{VI}}_t = \frac{1}{t}\sum_{i=1}^{t} \text{VI}_i. \tag{2.8}$$

Hence, by comparing the entries of this vector, i.e. $(\overline{\text{VI}}_t(1), \dots, \overline{\text{VI}}_t(p))$, the user is able to rank the variables, and this is commonly done using a built-in option from the standard random forests software package [33].

Up to this point, we have not specified a particular rule for computing the values $\text{VI}_i(l)$, but several choices are available. For instance, two of the standard choices are based on the notions of "node impurity" (for regression trees) or "random permutations" (for general regression functions). However, from an abstract point of view, our proposed method does not depend on the underlying details of these rules, and so we refer to the book [22, §15.3.2] for additional background. Indeed, our proposed method is applicable to any VI rule, provided that the random vectors $\text{VI}_1, \dots, \text{VI}_t$ are conditionally i.i.d. given $\mathcal{D}$ — and this is satisfied by both of the standard rules when $T_1, \dots, T_t$ follow the representation (1.2). Also, it should be mentioned that a considerable literature has investigated limitations and improvements of the standard VI rules, e.g. [59, 51, 58, 47, 45], and the study of variable importance in this context continues to be an open direction of research.

When the vectors $\text{VI}_1, \dots, \text{VI}_t$ are conditionally i.i.d. given $\mathcal{D}$, the vector $\overline{\text{VI}}_t$ will typically converge to a limit, say $\text{vi}_\infty \in \mathbb{R}^p$, as $t \to \infty$ and $\mathcal{D}$ is held fixed. In order to measure this convergence uniformly across $l \in \{1, \dots, p\}$, we will focus on the (unobserved) random variable defined by

$$\varepsilon_t := \max_{1\le l\le p} |\overline{\text{VI}}_t(l) - \text{vi}_\infty(l)|, \tag{2.9}$$

and our goal will be to estimate its $(1-\alpha)$-quantile, denoted as

$$\mathsf{q}_{1-\alpha}(t) := \inf\Big\{q \in [0,\infty) \,\Big|\, \mathbb{P}\big(\varepsilon_t \le q \,\big|\mathcal{D}\big) \ge 1-\alpha\Big\}. \tag{2.10}$$

The bootstrap method for variable importance. By analogy with our method for estimating the quantiles of $\text{MSE}_t - \text{mse}_\infty$, we propose to construct bootstrap samples of ε_t by resampling the vectors $\text{VI}_1, \dots, \text{VI}_t$, and then estimating $\mathsf{q}_{1-\alpha}(t)$ with the empirical $(1-\alpha)$-quantile. In algorithmic form, the procedure works as follows.

Algorithm 2.2 Bootstrap method for estimating $\mathsf{q}_{1-\alpha}(t)$

For $b = 1, \dots, B$ **:**

- Sample t vectors $(\text{VI}_1^*, \dots, \text{VI}_t^*)$ with replacement from $(\text{VI}_1, \dots, \text{VI}_t)$, and let $\overline{\text{VI}}_t^* = \frac{1}{t}\sum_{i=1}^{t} \text{VI}_i^*$.
- Compute the bootstrap sample

$$\varepsilon_{t,b}^* := \max_{1\le l\le p} |\overline{\text{VI}}_t^*(l) - \overline{\text{VI}}_t(l)|. \tag{2.11}$$

Return: the empirical $(1-\alpha)$-quantile of $\varepsilon_{t,1}^*, \dots, \varepsilon_{t,B}^*$ to estimate $\mathsf{q}_{1-\alpha}(t)$.

Numerical results illustrating the performance of this algorithm, as well as Algorithm 2.1, are given section 5. Also, in Appendix F of the supplementary material, we show how Algorithm 2.2 can be adapted to the situation where convergence is measured in terms of the relative error variable $\max_{1\leq l\leq p}|\overline{\text{VI}}_t(l) - \text{vi}_\infty(l)|/|\text{vi}_\infty|$. Numerical results for the case of relative error are provided there as well.

3. Main result.

In this section, we develop the main theoretical result of the paper (Theorem 3.1), which quantifies the coverage probability of the bootstrap estimate $\hat{q}_{1-\alpha}(t)$ for $q_{1-\alpha}(t)$. Namely, we will show that for a fixed set $\mathcal{D}$, the inequality

$$\text{MSE}_t - \text{mse}_\infty \;\leq\; \hat{q}_{1-\alpha}(t) \tag{3.1}$$

holds with a probability that is not much less than $1-\alpha$. Later on, we will also show that a corresponding result holds for estimating the quantile $\mathsf{q}_{1-\alpha}(t)$ in the context of variable importance (cf. subsection 3.1).

To establish the main result, we will rely on a common type of simplification, which is to exclude sources of error beyond the resampling process itself. More specifically, we will focus on bootstrap samples of the form $\text{MSE}_t^* - \text{MSE}_t$, defined in equation (2.5), since these are not affected by the extraneous error from estimating the functional ψ. (In other words, these samples are different from those of the form $\hat{\psi}(\bar{T}_t^*) - \hat{\psi}(\bar{T}_t)$ and $\hat{\psi}_{\text{O}}(\bar{T}_t^*) - \hat{\psi}_{\text{O}}(\bar{T}_t)$ described in subsection 2.2.) Meanwhile, even with such a simplification, the proof of the result is still quite involved. Also, this same choice was used in our previous analysis of the classification setting for the same reasons [36], but apart from this detail, the analysis in the current paper is entirely different.

With regard to the ensemble, it will only be assumed to satisfy the representation (1.2) and a basic moment condition in Theorem 3.1. From the standpoint of existing theory for randomized ensembles, these assumptions are very mild — because the representation (1.2) is always satisfied by bagging and random forests. By contrast, it is much more common in the theoretical literature to work with ensembles that are simpler than the ones used in practice, and similarly, our previous work in the classification setting relied on a specialized type of ensemble. Finally, it is notable that our result is fully *non-asymptotic*, whereas much existing work on the convergence of randomized ensembles has taken an asymptotic approach that does not always provide explicit rates of convergence.

Notation. If g and h are real-valued functions on $\mathcal{X}\times\mathbb{R}$, we denote their inner product with respect to the test point distribution $\nu = \mathcal{L}(X,Y)$ as

$$\langle g, h\rangle = \int_{\mathcal{X}\times\mathbb{R}} g(x,y)\, h(x,y)\, d\nu(x,y),$$

and accordingly, we write $\|g\|_{L_2} = \sqrt{\langle g, g\rangle}$. In addition, recall the function $\vartheta(x) = \mathbb{E}[T_1(x)|\mathcal{D}]$ from equation (2.3), and define the random variable

$$\zeta = 2\,\langle \vartheta - y, T_1 - \vartheta\rangle,$$

where $\vartheta - y$ is understood as the function that sends (x,y) to $\vartheta(x) - y$. When the random variable ζ is conditioned on $\mathcal{D}$, we denote its standard deviation by

$$\sigma(\mathcal{D}) = \sqrt{\text{var}(\zeta|\mathcal{D})},$$

and the finiteness of this quantity will follow from assumption **A2** below. Also, all expressions involving $1/\sigma(\mathcal{D})$ will be understood as ∞ in the exceptional case when $\sigma(\mathcal{D}) = 0$. Lastly, for each positive integer k, we define the moment parameter

$$\gamma_k(\mathcal{D}) = \big(\mathbb{E}\big[\|T_1 - y\|_{L_2}^{2k} \big| \mathcal{D}\big]\big)^{1/k}. \tag{3.2}$$

To interpret the role of this parameter, note that the random variable MSE_t can be written as $\|\frac{1}{t}\sum_{i=1}^t (T_i - y)\|_{L_2}^2$. Hence, the fluctuations of MSE_t are determined by the tail behavior of the summands $T_i - y$, and the parameter $\gamma_k(\mathcal{D})$ describes the tails of the summands through their moments.

Assumptions. With the above notation in place, the two assumptions for our main result may be stated as follows.

A1. The ensemble $T_1, \ldots, T_t$ can be represented in the form (1.2).

A2. There is at least one integer $k \geq 2$ such that $\gamma_{3k}(\mathcal{D}) < \infty$.

Regarding the finiteness of $\gamma_{3k}(\mathcal{D})$ in **A2**, it is noteworthy that this condition is satisfied for any k whenever the regression functions $T_1, \ldots, T_t$ are trained by the standard method of CART and the test label distribution has moments of all orders. This is because the regression trees trained by CART have a range that is determined by the training labels $Y_1, \ldots, Y_n$. In particular, if we define $M(\mathcal{D}) = \max_{1 \leq i \leq n} |Y_i|$, then every tree T_i satisfies $\sup_{x \in \mathcal{X}} |T_i(x)| \leq M(\mathcal{D})$. The same reasoning also applies beyond CART to any other method whose predictions are obtained as local averages of training labels.

We now state the main result of the paper.

Theorem 3.1. *Suppose that* **A1** *and* **A2** *hold. In addition, fix any small constant* $\alpha \in (0, 1)$ *and let* $k \geq 2$ *be as in* **A2**. *Lastly, let* $\hat{q}_{1-\alpha}(t)$ *denote the empirical* $(1-\alpha)$*-quantile of* B *bootstrap samples of the form* (2.5)*. and define the quantity*

$$\delta(\mathcal{D}) \ = \ \frac{k^2}{\sqrt{t}}\Big(\frac{\gamma_{3k}(\mathcal{D})}{\sigma(\mathcal{D})}\Big)^3 \ + \ e^{-k/2} \ + \ \sqrt{\frac{\log(B)}{B}}. \tag{3.3}$$

Then, there is an absolute constant $c_0 > 0$ *such that* $\hat{q}_{1-\alpha}(t)$ *satisfies*

$$\mathbb{P}\Big(\text{MSE}_t - \text{mse}_\infty \leq \hat{q}_{1-\alpha}(t) \,\Big|\, \mathcal{D}\Big) \ \geq \ 1 - \alpha - c_0\,\delta(\mathcal{D}). \tag{3.4}$$

Remarks. In essence, the result shows that $\hat{q}_{1-\alpha}(t)$ bounds the unknown convergence gap $\text{MSE}_t - \text{mse}_\infty$ with a probability that is not much less than the ideal value of $1-\alpha$. To comment on some further aspects of the result, note that the inequality (3.4) has the desirable property of being *scale-invariant* with respect to the labels $Y_1, \ldots, Y_n$ and the functions $T_1, \ldots, T_t$. More precisely, if we were to change the units of the labels and functions by a common scale factor, it can be checked that both sides of (3.4) would remain unchanged.

Another important aspect of Theorem 3.1 deals with the dependence of $\delta(\mathcal{D})$ on the value of k, and it is of interest to develop a bound on $\delta(\mathcal{D})$ that simplifies this dependence. To do

This manuscript is for review purposes only.

this, we can look at a basic situation where the regression functions are trained by CART and the test label variable is bounded. In addition, we may consider the particular choice

$$k = \lceil \log(t) - 4\log\log(t) \rceil, \tag{3.5}$$

which leads to the following bounds,

$$e^{-k/2} \le \frac{\log(t)^2}{\sqrt{t}} \quad \text{and} \quad \frac{k^2}{\sqrt{t}} \le \frac{c_1 \log(t)^2}{\sqrt{t}},$$

for some absolute constant $c_1 > 0$ and all $t \ge 2$. In turn, it follows that there is a number $c(\mathcal{D}) > 0$ not depending on t, k, or B, such that

$$\delta(\mathcal{D}) \le \tfrac{c(\mathcal{D})\log(t)^2}{\sqrt{t}} + \sqrt{\tfrac{\log(B)}{B}}, \tag{3.6}$$

which provides a considerable simplification. Hence, under the conditions just mentioned, and with $\mathcal{D}$ held fixed, the quantity $\delta(\mathcal{D})$ converges to 0 at *nearly parametric rates* with respect to t and B.

3.1. Bootstrap consistency in the context of variable importance.

Having developed our main result as a consistency guarantee for Algorithm 2.1 in the context of mean-squared error, we now aim to establish a corresponding result for Algorithm 2.2 in the context of variable importance, which is given as Theorem 3.2 below.

Setting and assumptions. In order to formulate this result, we will proceed along the lines of the setup described in subsection 2.3. Recall that for each random function T_i with $i \in \{1, \dots, t\}$, there is an associated random vector $\textsc{vi}_i = (\textsc{vi}_i(1), \dots, \textsc{vi}_i(p)) \in \mathbb{R}^p$, where $\textsc{vi}_i(l)$ refers to the importance assigned to the variable l by the function T_i, and the sample average is denoted $\overline{\textsc{vi}}_t = \frac{1}{t}\sum_{i=1}^t \textsc{vi}_i$. The only two conditions required of $\textsc{vi}_1, \dots, \textsc{vi}_t$ are as follows.

A3. The random vectors $\textsc{vi}_1, \dots, \textsc{vi}_t \in \mathbb{R}^p$ are conditionally i.i.d. given $\mathcal{D}$.

A4. There are positive numbers $b(\mathcal{D})$ and $b'(\mathcal{D})$ such that the following inequalities hold almost surely for all $l \in \{1, \dots, p\}$,

$$b(\mathcal{D}) \le \sqrt{\operatorname{var}(\textsc{vi}_1(l)|\mathcal{D})} \quad \text{and} \quad \textsc{vi}_1(l) \le b'(\mathcal{D}). \tag{3.7}$$

Perhaps the most important point to emphasize about **A3** is that it is automatically satisfied by two of the standard variable importance measures used within random forests, namely the "node impurity" measure and the "random permutations" measure [33]. More generally, as long as each vector $\textsc{vi}_i$ can be computed as a function of T_i, and as long as T_i can be represented in the abstract form (1.2), then **A3** will hold. With regard to the first inequality in **A4**, this is simply a non-degeneracy condition, which rules out situations where $\textsc{vi}_i(l)$ has no algorithmic fluctuations. Meanwhile, the second inequality in **A4** is always satisfied by the two standard variable importance measures in random forests when each T_i is trained via CART. Lastly, the condition **A4** ensures that $\overline{\textsc{vi}}_t$ has a limit as $t \to \infty$ with $\mathcal{D}$ held fixed, which is given by $\mathrm{vi}_\infty = \mathbb{E}[\textsc{vi}_1|\mathcal{D}]$.

The gist of Theorem 3.2 below is that the output $\hat{\mathrm{q}}_{1-\alpha}(t)$ of Algorithm 2.2 has reliable coverage probability when it is used as an upper bound on $\max_{1\le l\le p} |\overline{\textsc{vi}}_t(l) - \mathrm{vi}_\infty(l)|$.

Theorem 3.2. *Suppose that* **A3** *and* **A4** *hold, and fix any small constants* $\alpha, \eta \in (0,1)$. *In addition, let* $\hat{\mathsf{q}}_{1-\alpha}(t)$ *denote the empirical* $(1-\alpha)$*-quantile of* B *bootstrap samples of the form* (2.11)*, and define the quantity*

$$\tilde{\delta} = \sqrt{\tfrac{\log(2pt)^3}{t}} \; + \; \sqrt{\tfrac{\log(B)}{B}}. \tag{3.8}$$

Then, there is a number $\tilde{c}(\mathcal{D}) > 0$ *depending only on the triple* $(\eta, b(\mathcal{D}), b'(\mathcal{D}))$ *such that* $\hat{\mathsf{q}}_{1-\alpha}(t)$ *satisfies*

$$\mathbb{P}\bigg(\max_{1\leq l\leq p} |\overline{\mathrm{VI}}_t(l) - \mathrm{vi}_\infty(l)| \leq \hat{\mathsf{q}}_{1-\alpha}(t) + \eta \,\bigg|\, \mathcal{D}\bigg) \;\geq\; 1 - \alpha - \tilde{c}(\mathcal{D})\,\tilde{\delta}. \tag{3.9}$$

Remarks. Just like Theorem 3.1 given earlier, this result quantifies coverage probability in a non-asymptotic manner. On the other hand, one small point of contrast with Theorem 3.1 is the constant $\eta \in (0,1)$ in the present result, which serves only as a theoretical expedient, and can be fixed at an *arbitrarily small* value. Concerning the proof, it leverages recent advances on bootstrap methods for "max statistics" [16]. Furthermore, under some extra structural assumptions on the covariance matrix of VI_1, it is possible to replace the error term $\log(2pt)^{3/2}t^{-1/2}$ in equation (3.8) with a dimension-free term of the form $t^{-1/2+\epsilon_0}$, for an arbitrarily small constant $\epsilon_0 > 0$ [38].

4. Computation and speedups.

In order for the proposed method to be a practical a tool for checking algorithmic convergence, its computational cost should be manageable in comparison to training the ensemble itself. Below, in subsection 4.1, we offer a quantitative comparison, showing that under simple conditions, Algorithm 2.1 and Algorithm 2.2 are not a bottleneck in relation to training t regression functions with CART. Additionally, we show in subsection 4.2 how an extrapolation technique from our previous work on classification can be improved in our current setting with a *bias correction rule.*

4.1. Cost comparison.

Since the CART method is based on a greedy iterative algorithm, the exact computational cost of training a regression tree is difficult to describe analytically. Due to this difficulty, the authors of CART studied its cost in the simplified situation where each node of a regression tree is split into exactly 2 child nodes (except for the leaves). To be more precise, suppose $\mathcal{X} \subset \mathbb{R}^p$, and let $d \geq 2$ denote the "depth" of the tree, so that there are 2^d leaves. In addition, suppose that when the algorithm splits a given node, it searches over $\lceil p/3 \rceil$ candidate variables that are randomly chosen from $\{1, \ldots, p\}$, which is the default rule when CART is used by random forests [33]. Based on these assumptions, the analysis in the book [12, p.166] shows that the number of operations involved in training t such trees is at least of order $\Omega(t \cdot p \cdot d \cdot n)$.[1]

The cost of Algorithm 2.1. To determine the cost of Algorithm 2.1, it is important to clarify that when bagging and random forests are used in practice, the prediction error of the ensemble is typically estimated automatically using either hold-out or OOB samples. As a result, the predicted values of each tree on these samples can be regarded as being pre-computed by the

[1] We use $\Omega(\cdot)$ and $\mathcal{O}(\cdot)$ in the conventional way, so that they respectively refer to lower and upper bounds that hold up to constants [26, §9.2].

ensemble method. Once these values are available, the subsequent cost of Algorithm 2.1 is simple to measure. Specifically, in the case of hold-out samples, equation (2.6) shows that the cost to obtain $\hat{\psi}(\bar{T}_t) - \hat{\psi}(\bar{T}_t^*)$ for each bootstrap sample is $\mathcal{O}(t \cdot m)$, which leads to an overall cost that is $\mathcal{O}(B \cdot t \cdot m)$. Similarly, for the case of OOB samples, the overall cost is $\mathcal{O}(B \cdot t \cdot n)$. Altogether, this leads to the conclusion that the cost of Algorithm 2.1 does not exceed that of training the ensemble if the number of bootstrap samples satisfies the very mild condition

$$B = \mathcal{O}(p \cdot d), \tag{4.1}$$

and this applies to either the hold-out or OOB cases, provided $m = \mathcal{O}(n)$. Moreover, our discussion in subsection 4.2 will show that the condition (4.1) can be relaxed even further via extrapolation.

Beyond the fact that Algorithm 2.1 compares well with the cost of training an ensemble, there are several other favorable aspects to mention. First, the algorithm only relies on predicted labels for its input, and it never needs to access any points in the space $\mathcal{X}$. In particular, this means that the cost of the algorithm is independent of the dimension of $\mathcal{X}$. Second, the bootstrap samples in Algorithm 2.1 are simple to compute in parallel, which means that the runtime of the algorithm can essentially be reduced by a factor of B.

The cost of Algorithm 2.2. Many of the previous considerations for Algorithm 2.1 also apply to Algorithm 2.2, but it turns out that the cost of Algorithm 2.2 can be much less when n is large. Because each bootstrap sample in Algorithm 2.2 requires forming an average of t vectors in $\mathbb{R}^p$, it is straightforward to check that the overall cost is $\mathcal{O}(B \cdot t \cdot p)$, where we view the vectors $\text{VI}_1, \dots, \text{VI}_t$ as being pre-computed by the ensemble method. Consequently, the cost is independent of n, and the algorithm is thus highly scalable. Furthermore, under the setup of our earlier cost comparison with CART, the cost of Algorithm 2.2 does not exceed the cost of training the ensemble if

$$B = \mathcal{O}(n \cdot d),$$

which allows for plenty of bootstrap samples in practice. Better still, our numerical experiments show that just a few dozen bootstrap samples can be sufficient when n is on the order of 10^4, indicating that Algorithm 2.2 is quite inexpensive in comparison to training.

4.2. Further reduction of cost by extrapolation. The basic idea of extrapolation is to check algorithmic convergence for a small "initial" ensemble, say of size t_0, and then use this information to "look ahead" and predict convergence for a larger ensemble of size $t > t_0$. This general technique has a long history in the development of resampling methods and numerical algorithms, and further background can be found in [8, 3, 4, 7, 13, 57] among others. In the remainder of this section, we first summarize how extrapolation was previously developed in our companion paper [36], and then explain how that approach can be improved in the present context with a bias correction rule for OOB samples.

A basic version of extrapolation. At a technical level, our use of extrapolation is based on the central limit theorem, which suggests that the fluctuations of $\text{MSE}_t - \text{mse}_\infty$ should scale like $1/\sqrt{t}$ as a function of t. As a result, we expect that the quantile $q_{1-\alpha}(t)$ should behave like

$$q_{1-\alpha}(t) \approx \tfrac{\kappa}{\sqrt{t}},$$

for some quantity κ that may depend on all problem parameters except t.

To take advantage of this heuristic scaling property, suppose that we train an initial ensemble of size t_0, and run Algorithm 2.1 to obtain an estimate $\hat{q}_{1-\alpha}(t_0)$. We can then extract an estimate of κ by defining

$$\hat{\kappa} = \sqrt{t_0}\,\hat{q}_{1-\alpha}(t_0).$$

Next, we can rapidly estimate $q_{1-\alpha}(t)$ for all subsequent $t \geq t_0$ by defining the extrapolated estimate

$$\hat{q}^{\,\text{ext}}_{1-\alpha}(t) \;=\; \tfrac{\hat{\kappa}}{\sqrt{t}} \;=\; \tfrac{\sqrt{t_0}\hat{q}_{1-\alpha}(t_0)}{\sqrt{t}}. \tag{4.2}$$

In particular, there are two crucial benefits of this estimate: (1) It is much faster to apply Algorithm 2.1 to a small initial ensemble of size t_0 than to a large one of size t. (2) If we would like MSE_t to be within some tolerance $\epsilon > 0$ of the limit mse_∞, then we can use the condition

$$\hat{q}^{\,\text{ext}}_{1-\alpha}(t) \leq \epsilon$$

to *dynamically predict* how large t must be chosen to reach that tolerance, namely $t \geq (\sqrt{t_0}\hat{q}_{1-\alpha}(t_0)/\epsilon)^2$.

Bias-corrected extrapolation. If the initial estimate $\hat{q}_{1-\alpha}(t_0)$ is obtained by implementing Algorithm 2.1 with OOB samples, it turns out to be a biased estimate of $q_{1-\alpha}(t_0)$. Fortunately however, it is possible to correct for this bias in a simple way, as we now explain.

To understand the source of the bias, consider a particular training point X_j and note that for an initial ensemble of size t_0, the expected number of functions for which X_j is OOB is given by

$$\tau_n(t_0) = (1 - 1/n)^n \cdot t_0. \tag{4.3}$$

In other words, this means that when an ensemble of size t_0 makes a prediction on an OOB point, the "effective" size of the ensemble is $\tau_n(t_0)$, rather than t_0. As a result, if we implement Algorithm 2.1 using OOB samples with an initial ensemble of size t_0, then the output $\hat{q}_{1-\alpha}(t_0)$ should really be viewed as an estimate of $q_{1-\alpha}(\tau_n(t_0))$, rather than $q_{1-\alpha}(t_0)$.

Based on this reasoning, we can adjust our previous definition of the estimate $\hat{q}^{\,\text{ext}}_{1-\alpha}(t)$ in (4.2) by using

$$\hat{q}^{\,\text{ext,O}}_{1-\alpha}(t) = \tfrac{\sqrt{\tau_n(t_0)}\hat{q}_{1-\alpha}(t_0)}{\sqrt{t}} \qquad \text{for} \qquad t \geq \tau_n(t_0). \tag{4.4}$$

Later on, in section 5 we will demonstrate that this simple adjustment works well in practice.

Remark. As a clarification, it should be noted that the definition (4.4) is only to be used when Algorithm 2.1 is implemented with OOB samples, and the basic rule (4.2) should be used in the case of hold-out samples. Also, the basic rule (4.2) can be easily adapted to extrapolate the estimate produced by Algorithm 2.2, and so we omit the details in the interest of brevity.

5. Numerical results.

We now demonstrate the bootstrap's numerical accuracy in the tasks of measuring algorithmic convergence with respect to both mean-squared error and variable importance. Overall, our results show that the extrapolated OOB estimate is accurate at predicting the effect of increasing t. In fact, the results show that extrapolation succeeds at predicting what will happen when t is increased by a factor of 4 beyond t_0, and possibly much farther.

This manuscript is for review purposes only.

5.1. Organization of experiments.

Data preparation. Our experiments were based on several natural datasets that were each randomly partitioned in the following way. Letting $\mathcal{F}$ denote the full set of observation pairs $(X_1, Y_1), (X_2, Y_2), \dots$ for a given dataset, we evenly split $\mathcal{F}$ into a disjoint union $\mathcal{F} = \mathcal{D} \sqcup \mathcal{T}$, where $\mathcal{D}$ was used as a training set, and $\mathcal{T}$ was used as a "ground truth set" to approximate the true quantile curves $q_{1-\alpha}(t)$ and $\mathtt{q}_{1-\alpha}(t)$.

Since Algorithm 2.1 relies on a hold-out set, we also used a relatively small subset $\mathcal{H} \subset \mathcal{T}$ for that purpose. Specifically, the hold-out set $\mathcal{H}$ was chosen so that its cardinality satisfied $|\mathcal{H}|/(|\mathcal{H}| + |\mathcal{D}|) = 1/6$, up to rounding error. This reflects a practical situation where the user can only afford to allocate 1/6 of the available data for the hold-out set. In other words, the idea is to think of the user as only having access to $\mathcal{D} \sqcup \mathcal{H}$, while the set $\mathcal{T}$ is used externally to determine $q_{1-\alpha}(t)$ and $\mathtt{q}_{1-\alpha}(t)$.

Each of the full datasets are briefly summarized below.

- *Housing*: This dataset originates from 1990 California census and is available as part of the online supplement to the book [25]. The observations correspond to different housing districts, and for each one, there are 9 features for predicting the median home price in that district. ($|\mathcal{F}| = 20{,}640$, $|\mathcal{D}| = |\mathcal{T}| = 10{,}320$, $|\mathcal{H}| = 4{,}128$).
- *Protein*: This is dataset was collected from the fifth through ninth series of CASP experiments [44], and is available at the UCI repository [19] under the title *Physicochemical Properties of Protein Tertiary Structure Data Set.* The observations correspond to artificially generated conformations of proteins (known as decoys) that are described by 9 biophysical features. Each decoy can be thought of as a perturbation of an associated "target" protein, and the features are used to predict how far the decoy is from its target. ($|\mathcal{F}| = 45{,}730$, $|\mathcal{D}| = |\mathcal{T}| = 22{,}865$, $|\mathcal{H}| = 4{,}573$)
- *Music*: This dataset consists of audio recordings (observations) described by 68 features that are used to predict the geographic latitude of the recording, as described in [64]. The dataset is available at the UCI repository [19] under the title *Geographical Origin of Music Data Set.* ($|\mathcal{F}| = 1{,}059$, $|\mathcal{D}| = |\mathcal{T}| = 530$, $|\mathcal{H}| = 106$)
- *Diamond*: This dataset arises from a collection of diamonds, each described by 9 features that are used to predict the diamond's price. The dataset was obtained as a downsampled version of `diamonds` in the package `ggplot2` [63]. ($|\mathcal{F}| = 10{,}000$, $|\mathcal{D}| = |\mathcal{T}| = 5{,}000$, $|\mathcal{H}| = 1{,}000$)

Computing the true quantile curves $q_{1-\alpha}(t)$ and $\mathtt{q}_{1-\alpha}(t)$. Once a full dataset $\mathcal{F}$ was partitioned as above, we ran the random forests algorithm 1,000 times on the associated set $\mathcal{D}$, using the `R` package `randomForest` [33]. The overall process was a serious computational undertaking, because 2,000 regression trees were trained during every run, and hence a total of 2×10^6 trees were trained on each dataset.

During each run, as the ensemble size increased from $t = 1$ up to $t = 2{,}000$, the corresponding true values of $\textsc{mse}_t$ were approximated with the ensemble's error rate on $\mathcal{T}$. Also, the true value of mse_∞ was approximated with the average of the 1,000 approximate values of $\textsc{mse}_{2{,}000}$. In this way, the collection of runs produced 1,000 approximate sample paths of

$\text{MSE}_t - \text{mse}_\infty$, similar to those illustrated in the right panel of Figure 1. Finally, the quantile curve $q_{.90}(t)$ was approximated by using the empirical 90% quantile of the sample paths at each $t \in \{1, \dots, 2{,}000\}$.

To handle the setting of variable importance, essentially the same steps were used. Specifically, we measured variable importance in terms of node impurity to compute $\overline{\text{VI}}_t \in \mathbb{R}^p$ at every value $t \in \{1, \dots, 2{,}000\}$, for each of the 1,000 runs mentioned above. In addition, we approximated $\text{vi}_\infty \in \mathbb{R}^p$ with the average of the 1,000 realizations of $\overline{\text{VI}}_{2{,}000}$. Altogether, these computations provided us with 1,000 approximate sample paths of $\varepsilon_t = \max_{1 \leq l \leq p} |\overline{\text{VI}}_t(l) - \text{vi}_\infty(l))|$, and then we used the empirical 90% quantile at each $t \in \{1, \dots, 2{,}000\}$ as an approximation to $\mathsf{q}_{.90}(t)$.

Applying the bootstrap algorithms with extrapolation. For each of the described 1,000 runs of random forests, we applied the extrapolated versions of Algorithm 2.1 and Algorithm 2.2 at the initial ensemble size of $t_0 = 500$, using a choice of $B \in \{25, 50, 100\}$ bootstrap samples. (The extrapolation was carried out to a final ensemble size of $t = 2{,}000$.) Also, for Algorithm 2.1, we implemented both the hold-out and OOB versions, including the bias correction for the OOB samples described in equation (4.4). Hence, this provided us with 1,000 realizations of each type of estimate, allowing for an assessment of their variability.

5.2. Numerical results for mean-squared error.

Organization of the plots. The hold-out and OOB estimates for $q_{.90}(t)$ are illustrated in Figures 3 through 6. For each choice of $B \in \{25, 50, 100\}$, the colored curves represent the averages of the estimates over the 1,000 runs described previously, and the error bars display the fluctuations of the estimates over repeated runs — corresponding to the 10th and 90th percentiles of the estimates. (For the values of t between the endpoints, we omit the error bars for clarity. Also, the error bars should *not* be interpreted as confidence intervals for $q_{.90}(t)$, and they are only intended to illustrate the variability of the estimates.)

With regard to computation, another point to mention is that the estimates were only computed for the initial ensemble size $t_0 = 500$, and the rest of the estimated curves were obtained essentially *for free* by extrapolation. Lastly, as a clarification, it should be noted that the OOB curves are shifted to the left of the hold-out curves because of the bias correction rule (4.4) for OOB samples.

Remarks on performance. The main point to take away from the plots is that the OOB estimate performs well overall, and can be noticeably more accurate than the hold-out estimate (cf. Figures 3, 5, and 6). Furthermore, the OOB estimate has an extra advantage because it does not require the user to hold out any data. For these reasons, we recommend the OOB estimate in practice.

Concerning the number of bootstrap samples, we see the expected pattern that larger values of B reduce the fluctuations of the estimates. Nevertheless, even at $B = 25$, the fluctuations are well-behaved. So, for practical purposes, this indicates that the speedup from a small choice of B may outweigh a relatively minor reduction in variance.

Another conclusion to draw from the plots is that the bias correction plays a significant role in the extrapolation of the OOB estimate. To see this, note that if the bias correction were not used, this would be equivalent to shifting the blue curves so that they start at the same point as the green curves, which would clearly lead to a loss in accuracy. Also, it is remarkable that the extrapolated OOB estimate continues to be accurate at a final ensemble size of $t = 2{,}000$ that is 4 times larger than the initial ensemble size $t_0 = 500$. Hence, this provides the user with a very inexpensive way to predict how quickly the ensemble will converge.

To explain the inferior performance of the hold-out estimate, recall that it uses the small set $\mathcal{H}$ in order to estimate MSE_t. As a result of the small size of $\mathcal{H}$, the estimate of MSE_t has high variability, which inflates the upper extremes and ultimately leads to a larger estimate of $q_{.90}(t)$. On the other hand, the OOB estimate is able to take advantage of the OOB samples in the much larger set $\mathcal{D}$, which reduces this detrimental effect.

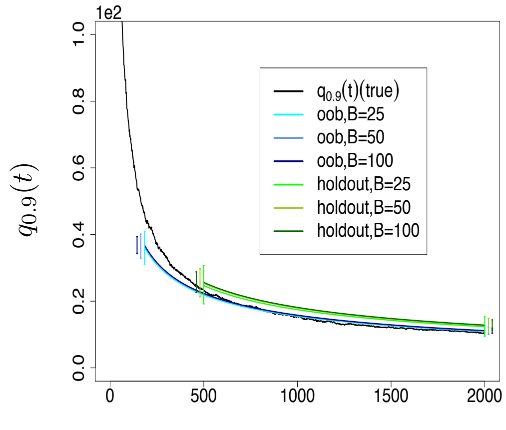

Figure 3: Housing Data

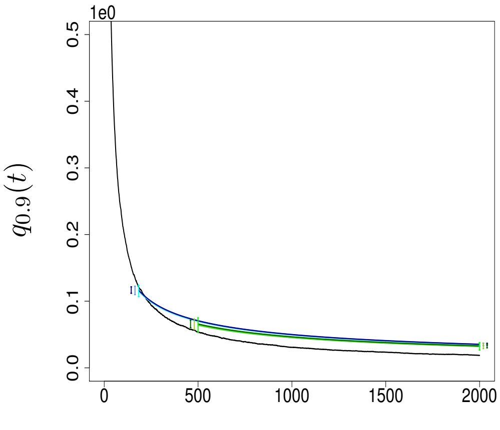

Figure 4: Protein Data

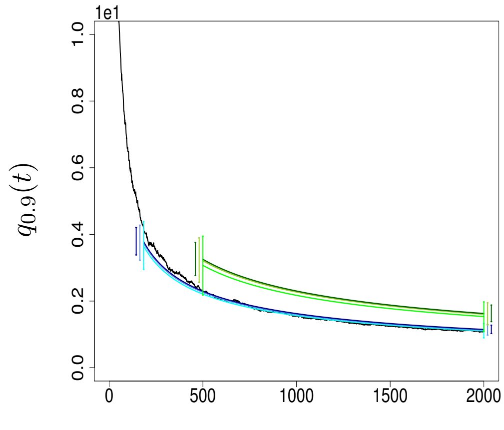

Figure 5: Music Data

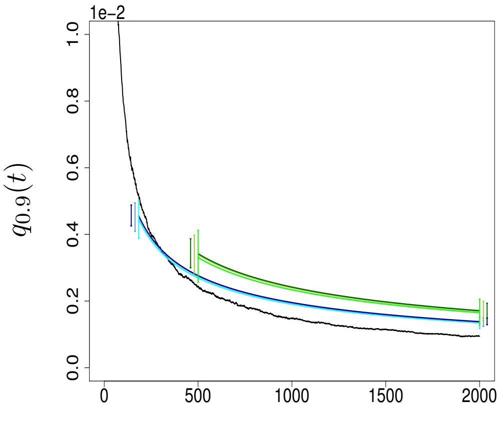

Figure 6: Diamond Data

5.3. Numerical results for variable importance. The results in the setting of variable importance are simpler to describe, since there is only one type of estimate for $\mathsf{q}_{.90}(t)$. Figures 7 through 10 display the average of the 1,000 realizations of the estimate using a blue curve (corresponding to $B = 50$), and as before, the error bars at the endpoints represent the 10th and 90th percentiles. Also, the extrapolation procedure was performed using an initial ensemble size of $t_0 = 500$, as in the previous subsection. From the four plots, it is clear that the extrapolated estimate displays excellent overall performance, with its bias and variance both being very small.

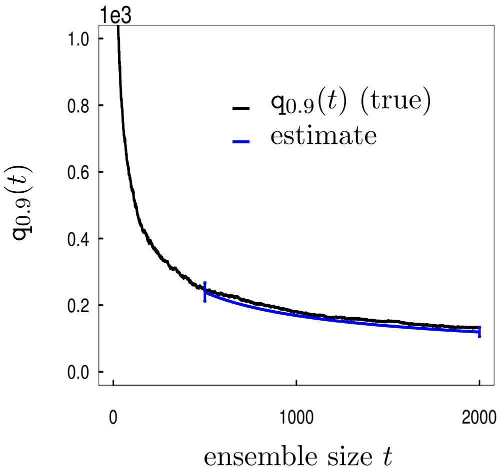

Figure 7: Housing Data

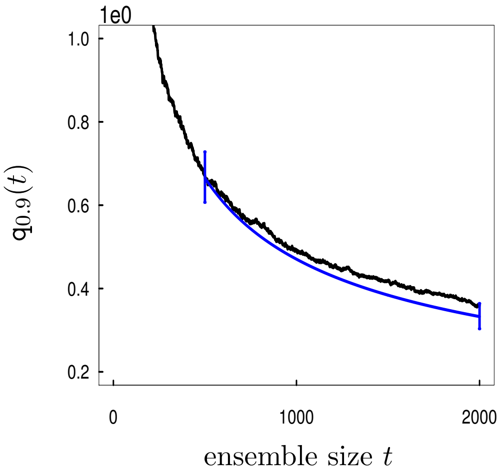

Figure 8: Protein Data

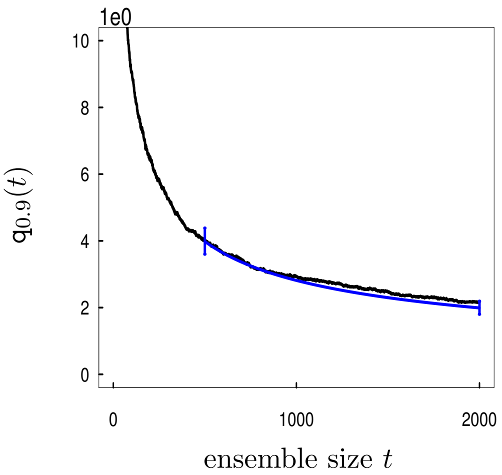

Figure 9: Music Data

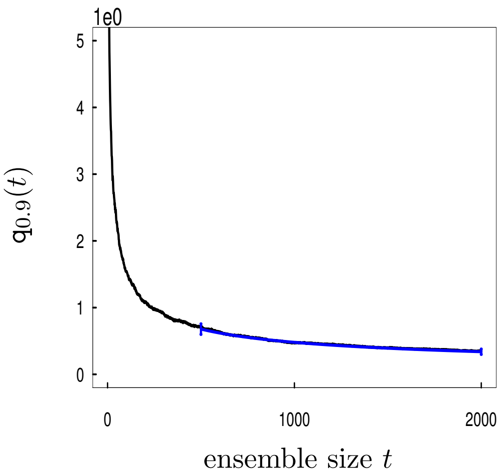

Figure 10: Diamond Data

6. Conclusion. In this paper, we have developed a bootstrap method that allows users to measure the algorithmic convergence of regression ensembles with a level of precision that has not previously been available. In particular, the method provides users with a systematic way to determine when the ensemble is large enough so that it will perform nearly as well as an ideal infinite ensemble — with respect to either mean-squared error or variable importance. With regard to theory, our approach is supported by guarantees in Theorems 3.1 and 3.2 that quantify the coverage probabilities of the quantile estimates produced by Algorithms 2.1 and 2.2. Computationally, the method incurs only modest cost in comparison to training the ensemble itself, and furthermore, the method naturally lends itself to speedups via parallel computing and extrapolation. Lastly, we have shown empirically that the method has encouraging finite-sample performance in a range of situations.

Acknowledgments. The authors would like to thank the reviewers and editors for their valuable feedback, which significantly improved the paper.

REFERENCES

[1] S. Arlot and R. Genuer, *Analysis of purely random forests bias*, arXiv:1407.3939, (2014).
[2] J. Basilico, M. Munson, T. Kolda, K. Dixon, and W. Kegelmeyer, *Comet: A recipe for learning and using large ensembles on massive data*, in Data Mining (ICDM), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 41–50.
[3] P. Bertail et al., *Second-order properties of an extrapolated bootstrap without replacement under weak assumptions*, Bernoulli, 3 (1997), pp. 149–179.
[4] P. Bertail and D. N. Politis, *Extrapolation of subsampling distribution estimators: the iid and strong mixing cases*, Canadian Journal of Statistics, 29 (2001), pp. 667–680.
[5] G. Biau, *Analysis of a random forests model*, Journal of Machine Learning Research, 13 (2012), pp. 1063–1095.
[6] G. Biau, L. Devroye, and G. Lugosi, *Consistency of random forests and other averaging classifiers*, Journal of Machine Learning Research, 9 (2008), pp. 2015–2033.
[7] P. J. Bickel and A. Sakov, *Extrapolation and the bootstrap*, Sankhyā: The Indian Journal of Statistics, Series A, (2002), pp. 640–652.
[8] P. J. Bickel and J. A. Yahav, *Richardson extrapolation and the bootstrap*, Journal of the American Statistical Association, 83 (1988), pp. 387–393.
[9] R. Blaser and P. Fryzlewicz, *Random rotation ensembles*, The Journal of Machine Learning Research, 17 (2016), pp. 126–151.
[10] L. Breiman, *Bagging predictors*, Machine Learning, 24 (1996), pp. 123–140.
[11] L. Breiman, *Random forests*, Machine Learning, 45 (2001), pp. 5–32.
[12] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, CRC press, 1984.
[13] C. Brezinski and M. R. Zaglia, *Extrapolation Methods: theory and practice*, Elsevier, 2013.
[14] P. Bühlmann and B. Yu, *Analyzing bagging*, Annals of Statistics, 30 (2002), pp. 927–961.
[15] T. I. Cannings and R. J. Samworth, *Random projection ensemble classification (with discussion)*, Journal of the Royal Statistical Society Series B, (2017).
[16] V. Chernozhukov, D. Chetverikov, K. Kato, and Y. Koike, *Improved central limit theorem and bootstrap approximations in high dimensions*, `arxiv:1912.10529`, (2019).
[17] H. A. Chipman, E. I. George, R. E. McCulloch, et al., *Bart: Bayesian additive regression trees*, The Annals of Applied Statistics, 4 (2010), pp. 266–298.
[18] R. Díaz-Uriarte and S. A. De Andres, *Gene selection and classification of microarray data using random forest*, BMC bioinformatics, 7 (2006), p. 3.
[19] D. Dua and C. Graff, *UCI machine learning repository*, 2017, http://archive.ics.uci.edu/ml.
[20] Y. Fang, *Scalable statistical inference for averaged implicit stochastic gradient descent*, Scandinavian Journal of Statistics, 46 (2019), pp. 987–1002.
[21] Y. Fang, J. Xu, and L. Yang, *Online bootstrap confidence intervals for the stochastic gradient descent estimator*, The Journal of Machine Learning Research, 19 (2018), pp. 3053–3073.

[22] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning*, Springer, 2001.
[23] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, *Variable selection using random forests*, Pattern Recognition Letters, 31 (2010), pp. 2225–2236.
[24] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, *Vsurf: an R package for variable selection using random forests*, The R Journal, 7 (2015), pp. 19–33.
[25] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow*, O'Reilly Media, 2017, https://github.com/ageron/handson-ml/tree/master/datasets.
[26] R. L. Graham, D. E. Knuth, O. Patashnik, and S. Liu, *Concrete Mathematics: A Foundation for Computer Science*, Addison & Wesley, 1994.
[27] B. Gregorutti, B. Michel, and P. Saint-Pierre, *Correlation and variable importance in random forests*, Statistics and Computing, 27 (2017), pp. 659–678.
[28] P. Hall and R. J. Samworth, *Properties of bagged nearest neighbour classifiers*, Journal of the Royal Statistical Society: Series B, 67 (2005), pp. 363–379.
[29] D. Hernández-Lobato, G. Martínez-Muñoz, and A. Suárez, *How large should ensembles of classifiers be?*, Pattern Recognition, 46 (2013), pp. 1323–1336.
[30] H. Ishwaran, *Variable importance in binary regression trees and forests*, Electronic Journal of Statistics, 1 (2007), pp. 519–537.
[31] P. Latinne, O. Debeir, and C. Decaestecker, *Limiting the number of trees in random forests*, in Multiple Classifier Systems, Springer, 2001.
[32] T. Li, L. Liu, A. Kyrillidis, and C. Caramanis, *Statistical inference using SGD*, in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[33] A. Liaw and M. Wiener, *Classification and regression by randomForest*, R News, 2 (2002), pp. 18–22, https://cran.r-project.org/web/packages/randomForest/randomForest.pdf.
[34] Y. Lin and Y. Jeon, *Random forests and adaptive nearest neighbors*, Journal of the American Statistical Association, 101 (2006), pp. 578–590.
[35] M. E. Lopes, *Estimating a sharp convergence bound for randomized ensembles*, arXiv:1303.0727, (2016).
[36] M. E. Lopes, *Estimating the algorithmic variance of randomized ensembles via the bootstrap*, The Annals of Statistics, 47 (2019), pp. 1088–1112.
[37] M. E. Lopes, N. B. Erichson, and M. W. Mahoney, *Error estimation for sketched SVD via the bootstrap*, in International Conference on Machine Learning (to appear), 2020.
[38] M. E. Lopes, Z. Lin, and H.-G. Mueller, *Bootstrapping max statistics in high dimensions: Near-parametric rates under weak variance decay and application to functional and multinomial data*, The Annals of Statistics, 48 (2020), pp. 1214–1229.
[39] M. E. Lopes, S. Wang, and M. W. Mahoney, *Error estimation for randomized least-squares algorithms via the bootstrap*, in International Conference on Machine Learning, 2018, pp. 3223–3232.
[40] M. E. Lopes, S. Wang, and M. W. Mahoney, *A bootstrap method for error estimation in randomized matrix multiplication.*, Journal of Machine Learning Research, 20 (2019), pp. 1–40.
[41] G. Louppe, L. Wehenkel, A. Sutera, and P. Geurts, *Understanding variable importances in forests of randomized trees*, in Advances in Neural Information Processing Systems, 2013, pp. 431–439.
[42] P.-G. Martinsson and J. Tropp, *Randomized numerical linear algebra: Foundations & algorithms*, Acta Numerica (to appear), `arxiv:2002.01387`, (2020).
[43] L. Mentch and G. Hooker, *Quantifying uncertainty in random forests via confidence intervals and hypothesis tests*, Journal of Machine Learning Research, 17 (2016), pp. 1–41.
[44] J. Moult, K. Fidelis, A. Kryshtafovych, and A. Tramontano, *Critical assessment of methods of protein structure prediction (CASP) — round IX*, Proteins: Structure, Function, and Bioinformatics, 79 (2011), pp. 1–5.
[45] S. Nembrini, I. R. König, and M. N. Wright, *The revival of the Gini importance?*, Bioinformatics, 34 (2018), pp. 3711–3718.
[46] A. Y. Ng and M. I. Jordan, *Convergence rates of the voting Gibbs classifier, with application to Bayesian feature selection*, in International Conference on Machine Learning, 2001, pp. 377–384.
[47] K. K. Nicodemus, J. D. Malley, C. Strobl, and A. Ziegler, *The behaviour of random forest permutation-based variable importance measures under predictor correlation*, BMC bioinformatics, 11 (2010), p. 110.
[48] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, *How many trees in a random forest?*, in Machine

Learning and Data Mining in Pattern Recognition, Springer, 2012, pp. 154–168.

[49] P. Probst and A.-L. Boulesteix, *To tune or not to tune the number of trees in random forest*, Journal of Machine Learning Research, 18 (2018), pp. 1–18.

[50] R. J. Samworth, *Optimal weighted nearest neighbour classifiers*, The Annals of Statistics, 40 (2012), pp. 2733–2763.

[51] M. Sandri and P. Zuccolotto, *A bias correction algorithm for the Gini variable importance measure in classification trees*, Journal of Computational and Graphical Statistics, 17 (2008), pp. 611–628.

[52] A. Schwing, C. Zach, Y. Zheng, and M. Pollefeys, *Adaptive random forest – How many "experts" to ask before making a decision?*, in Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 1377–1384.

[53] E. Scornet, *On the asymptotics of random forests*, Journal of Multivariate Analysis, 146 (2016), pp. 72–83.

[54] E. Scornet, *Random forests and kernel methods*, IEEE Transactions on Information Theory, 62 (2016), pp. 1485–1500.

[55] E. Scornet, G. Biau, and J.-P. Vert, *Consistency of random forests*, The Annals of Statistics, 43 (2015), pp. 1716–1741.

[56] J. Sexton and P. Laake, *Standard errors for bagged and random forest estimators*, Computational Statistics & Data Analysis, 53 (2009), pp. 801–811.

[57] A. Sidi, *Practical Extrapolation Methods: Theory and Applications*, Cambridge University Press, 2003.

[58] C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis, *Conditional variable importance for random forests*, BMC bioinformatics, 9 (2008), p. 307.

[59] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, *Bias in random forest variable importance measures: Illustrations, sources and a solution*, BMC bioinformatics, 8 (2007), p. 25.

[60] W. J. Su and Y. Zhu, *Uncertainty quantification for online learning and stochastic approximation via hierarchical incremental gradient descent*, arXiv:1802.04876, (2018).

[61] M. Talagrand, *Isoperimetry and integrability of the sum of independent Banach-space valued random variables*, The Annals of Probability, (1989), pp. 1546–1570.

[62] S. Wager, T. Hastie, and B. Efron, *Confidence intervals for random forests: the jackknife and the infinitesimal jackknife.*, Journal of Machine Learning Research, 15 (2014), pp. 1625–1651.

[63] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*, Springer, 2016.

[64] F. Zhou, Q. Claire, and R. D. King, *Predicting the geographical origin of music*, in 2014 IEEE International Conference on Data Mining, IEEE, 2014, pp. 1115–1120.

This manuscript is for review purposes only.