<https://helda.helsinki.fi>

Calibrating aquatic microfossil proxies with regression-tree ensembles : Cross-validation with modern chironomid and diatom data

Salonen, J Sakari

2016-07

Salonen , J S , Verster , A J , Engels , S , Soininen , J , Trachsel , M & Luoto , M 2016 , ' Calibrating aquatic microfossil proxies with regression-tree ensembles : Cross-validation with modern chironomid and diatom data ' , Holocene , vol. 26 , no. 7 , pp. 1040-1048 . <https://doi.org/10.1177/0959683616632881>

<http://hdl.handle.net/10138/173956>

<https://doi.org/10.1177/0959683616632881>

acceptedVersion

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.

This reprint may differ from the original in pagination and typographic detail.

Please cite the original version.

Calibrating aquatic microfossil proxies with regression-tree ensembles: cross-validation with modern chironomid and diatom data

J Sakari Salonen^{1,*}, Adrian J Verster², Stefan Engels³, Janne Soininen¹, Mathias Trachsel⁴, Miska Luoto¹

¹ *Department of Geosciences and Geography, PO Box 64, 00014 University of Helsinki, Finland*

² *Department of Molecular Genetics, Terrence Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Toronto, Ontario M5S 3E1, Canada*

³ *Institute for Biodiversity and Ecosystem Dynamics (IBED), University of Amsterdam, Science Park 904, 1090 GE, Amsterdam, The Netherlands*

⁴ *Department of Biology, University of Bergen, PO Box 7803, 5020 Bergen, Norway*

* *Corresponding author. E-mail: sakari.salonen@helsinki.fi. Phone: +358-29-4151577.*

Abstract

We examine the ability of four different regression-tree ensemble techniques (bagging, random forest, rotation forest, boosted tree) in calibration of aquatic microfossil proxies. The methods are tested with six chironomid and diatom datasets, using a variety of cross-validation schemes. We find random forest, rotation forest and the boosted tree to have a similar performance, while bagging performs less well and in several cases has trouble producing continuous predictions. In comparison with commonly-used parametric transfer-function approaches (PLS, WA, WA-PLS), we find that in some cases tree-ensemble methods outperform the best-performing transfer-function technique, especially with large datasets characterized by complex taxon responses and abundant noise. However, parametric transfer functions

remain competitive with datasets characterised by low number of samples or linear taxon responses. We present an implementation of the rotation forest algorithm in R.

Keywords: bagging, boosted regression tree, partial least squares, random forest, rotation forest, weighted averaging

Introduction

Quantitative environmental reconstructions prepared from aquatic microfossils have a multitude of applications, including environmental monitoring, palaeoclimate research, and the study of sensitivity, resilience, and human impact in ecosystems. Starting in the 1970s, numerous quantitative approaches have emerged to infer palaeoenvironmental parameters from microfossil assemblages (Juggins and Birks, 2012). Key techniques in palaeolimnological reconstructions have been regression-based multivariate transfer functions such as weighted averaging (WA; Birks et al., 1990a) or weighted averaging-partial least squares (WA-PLS; ter Braak and Juggins, 1993), based on parametric estimation of taxon responses (e.g., linear or unimodal) using modern calibration data. Past environmental values are then predicted based on the modelled responses for the taxa found in the fossil assemblages. Such parametric transfer functions have been complemented by methods based on assemblage matching (the modern-analogue technique (MAT)) and less frequently by other approaches such as ones based on Bayesian statistics or artificial neural networks (Juggins and Birks, 2012).

While parametric transfer functions remain a cornerstone of palaeoenvironmental reconstruction, in other fields of quantitative ecology, such as species-distribution modelling, parametric modelling tools (e.g., generalized linear modelling) have been noted as having considerable limitations in modelling species–

climate relationships. Consequently, ecologists have increasingly turned to so-called machine-learning (ML) algorithms, which explore the data while making minimal assumptions about the species–environment relationships (Elith et al., 2006; Zimmermann et al., 2010). Among the emerging ML-based modelling techniques are methods based on creating ensembles of regression tree models. In recent years, there has been growing interest in applying such regression-tree ensembles in fossil proxy calibration and palaeoenvironmental reconstruction (Goring et al., 2010; Salonen et al., 2012, 2014; Simpson and Birks, 2012; Juggins et al., 2015). There is clear potential in such transfer of methodology, as while the typical aim in palaeoenvironmental reconstruction (i.e., quantifying past environmental change) differs from that in species-distribution modelling (predicting range shifts under climatic change), the two applications are similar with respect to the statistical modelling, as both are underpinned by models of taxon response over environmental gradients.

Based on the general properties of regression trees, several potential advantages have been identified for tree ensembles as proxy-calibration tools. These include (A) their ability to model complex (e.g., bimodal) taxon responses, (B) their ability to deal with calibration datasets showing a diverse combination of response shapes, (C) their ability to model interactions between predictors, (D) their inherent selectivity in only using taxa showing an important response along the reconstructed gradient, and (E) their sensitivity to rare indicator taxa (Salonen et al., 2014; Juggins et al., 2015). While these expected strengths are acknowledged, the empirical evidence about the performance of tree-ensemble methods in calibrating microfossil proxies is only starting to emerge. A general challenge in this work is the difficulty of evaluating the reliability of any proxy calibration (Juggins and Birks, 2012). Commonly-used cross-validation (CV) schemes (e.g., leave-one-out (LOO)) can be susceptible to showing inflated performance in the presence of spatial auto-correlation (Telford and Birks, 2009) or correlated, ecologically significant secondary gradients (Juggins, 2013a), while the sensitivity to such confounding

factors may vary between methods (Telford and Birks, 2009; Salonen et al., 2014; Juggins et al., 2015), considerably complicating between-method comparisons. Meanwhile, the assessment of predictions prepared for down-core (fossil) data is often challenging due to limited prior information on the past environmental conditions (but see Telford and Birks, 2011). Goring et al. (2010) applied a single tree-ensemble method, the random forest (Breiman, 2001), in split-sampling tests with modern pollen samples. They found random forest to outperform WA/WA-PLS in predicting summer growing degree days but not in predicting summer precipitation, while MAT performed best with both variables. However, Goring et al. (2010) note the challenges in assessing the relative performance between methods due to spatial autocorrelation. Salonen et al. (2014) tested another tree-ensemble method, the boosted regression tree (BRT; De'ath, 2007) in pollen–climate calibration with inconclusive results, as BRTs failed to consistently outperform WA and MAT in robust *h*-block CV. Two studies have tested BRTs using *k*-fold CV with modern diatom datasets, and found BRTs to not improve on parametric methods including WA and maximum likelihood calibration of response curves (MLRC) (Simpson and Birks, 2012; Juggins et al., 2015). However, when predicting for simulated fossil datasets, Juggins et al. (2015) found BRTs to often outperform WA and MLRC, especially in situations where some taxa mainly respond to a non-reconstructed secondary variable and independent past shifts occur in that secondary variable. In summary, while increasing evidence exists about the usefulness of tree ensembles from other applications (Simpson and Birks, 2012), the evidence on their ability to improve on classical parametric methods in calibration of fossil proxies remains limited and inconclusive.

Here we test the ability of several tree-ensemble approaches in prediction of environmental parameter values based on aquatic microfossil assemblages. The performance of tree ensembles is compared against commonly-used parametric transfer-function techniques. We use six datasets, representing two key palaeolimnological proxies (diatoms and chironomids), several environmental response variables, and

three different continents (Europe, North America, and Africa). The predictive ability with each dataset is assessed by applying a variety of cross-validation schemes, where the selection of the most suitable CV scheme depends on the spatial structure of the individual dataset.

Materials and methods

Datasets

We test the predictive ability of different calibration approaches using six datasets, each consisting of samples of modern diatom or chironomid assemblages obtained from aquatic environments (mostly surface sediment samples from lakes and ponds) and observations on the modern values of an environmental variable that is considered to be an important environmental driver of species distribution patterns (see references in Table 1 for more information on proxy–environment relationships). Details and references for the used datasets are presented in Table 1. The datasets differ considerably in terms of number of samples (69–622), number of predictor variables, i.e., the number of diatom or chironomid taxa (78–652), and complexity of taxon response. All variables are continuous: taxon abundance percentages for all predictors and measured environmental values for the response variables.

[Table 1]

Two of these are lake diatom–water pH datasets, originating from the Surface Waters Acidification Programme of the United Kingdom (**Dia–pH_UK**) and the European Diatom Database Initiative (**Dia–pH_EU**). The latter dataset is considerably larger in terms of geographic span and number of samples. A third diatom dataset (**Dia–logEC**) is sampled from Ugandan crater lakes, with water log electrical

conductivity (logEC) as the response variable. The fourth dataset (**Chi-Tjul**) consists of chironomid samples from North American lakes, with mean July air temperature (T_{jul}) as response variable. The final two datasets (**Chi-depth**, **Dia-colour**) share the key feature that they are spatially clustered, with each cluster alone sampling a significant portion of the environmental gradient. This allows a leave-one-cluster-out CV (LOCO CV; Wenger and Olden, 2012) to be performed with these datasets, in effect constituting a k -fold external CV and providing a robust estimate of predictive ability for new data. **Chi-depth** consists of chironomid assemblages sampled along depth gradients from seven lakes in southeast Massachusetts, with 5–25 samples from each lake and 98 samples in total (Figure 1A). We use water depth as the response variable with this dataset. **Dia-colour** consists of benthic diatom assemblages, sampled from stone surfaces in Finnish streams. We note that this is not a traditional palaeolimnological calibration dataset as the species samples come from stream epilithon and not lake surface sediment. While this dataset may not be directly applicable to palaeolimnological reconstructions, we use it here as the spatial structure of the data enables the LOCO CV to be performed, and the dataset can thus serve as a useful test case of the ability of different methods to model environment–diatom relations. The sampling locations form eight spatial clusters, with 13–59 sampling sites in each cluster and 225 samples in total (Figure 1B). We use water colour (mgPt/L) as the response variable with the **Dia-colour** dataset (e.g., Battarbee et al., 1997; Steinberg, 2003; Soininen et al., 2004). Water colour was chosen from a set of environmental factors identified in earlier work as important ecological drivers in this dataset, also including growing-degree days, conductivity and total phosphorus (e.g., Soininen et al., 2004; Pajunen et al., 2015). With water colour the individual clusters span overlapping segments of the gradient, a requirement for implementing the LOCO CV. Conductivity and growing-degree days were rejected for not meeting this requirement, while total phosphorus was rejected for showing very poor performance in preliminary CV runs. Some samples from the high ends of the gradients were removed from **Dia-colour** (two samples with water colour > 350 mgPt/L) and **Chi-depth** (six samples with water depth > 9.3 m), as

these gradient segments were only covered by a single sample cluster, making prediction of these values in LOCO CV impossible. Outliers identified by the dataset authors were removed from **Dia-logEC** (seven samples; for details see Mills and Ryves, 2012) and **Chi-Tjul** (one sample; cf. Fortin et al., 2015).

[Figure 1]

Calibration methods

We test four different algorithms for building regression-tree ensembles: bagging (Breiman, 1996), random forest (RaF; Breiman, 2001), rotation forest (RoF; Rodríguez et al., 2006), and BRT (De'ath, 2007). All four approaches build on the simple regression tree (De'ath and Fabricius, 2000), a commonly-used and versatile data analysis tool. However, a weakness of the single tree model is poor predictive ability for new data, a drawback of the simple model structure. The ensemble methods used here are based on improving the predictive ability by combining a large number of trees, while still benefiting of many of the intrinsic advantages of a tree-based model (Simpson and Birks, 2012).

Bagging, RaF, and RoF are closely related methods which all generate a set number of trees and calculate the prediction based on the mean of the predictions from the individual trees (or a majority vote for a categorical response). The three methods differ in that they use increasingly complex ways to introduce diversity into the ensemble. In bagging, each tree is built with a bootstrap sample of the training data. The bootstrap is also used in RaF, but additionally, only a random subset of the predictor set is provided when determining each tree split. In RoF, the individual trees are likewise built from bootstrap samples. However, in RoF the training data predictor set for each tree is then divided into K random subsets, and principal component analysis (PCA) is applied separately on each subset. This creates a random rotation

of the predictor space for each tree and adds a second layer of stochasticity into RoF model building. In effect, these introduced stochasticities force the individual learners (trees) to view the data in slightly different contexts (e.g., using a subset of samples or predictors) which helps the learners to produce diverse and complementary "rules of thumb" which then enable the combined ensemble to predict accurately. In the ensemble building in bagging and RaF/RoF, new trees are calculated independently, without considering the predictive power of the previous trees. Our final tree-ensemble method, the BRT, differs significantly in that the ensemble is built sequentially, with each added tree aiming to explain the residuals of the previously fitted ensemble. The BRT can thus be likened to an additive regression model in which the individual terms are regression trees (e.g., Kuncheva and Rodríguez, 2007; Elith et al., 2008; Simpson and Birks, 2012). RaF and BRT have seen increasing use in palaeoecology (for a review cf. Simpson and Birks, 2012; for more recent examples cf. Salonen et al., 2012, 2014; Veloz et al., 2012; Zhao et al., 2012; Felde et al., 2014; Juggins et al., 2015; Li et al., 2015). Bagging or RoF have not, to our knowledge, been previously used in palaeoecology or palaeolimnology.

We also use three commonly-used transfer functions to assess the performance of these tree-ensemble techniques against parametric methods: PLS, WA, and WA-PLS. The three techniques differ in the shape of assumed response which is linear in PLS and unimodal in WA/WA-PLS (Juggins and Birks, 2012). To guide the choice between PLS and WA/WA-PLS we fit HOF models to the percentage values of each taxon (Huisman et al., 1993; implemented using the R package EHOF (version 1.5.4; Jansen and Oksanen, 2013)). In HOF, increasingly complex parametric response models are fitted to determine the most complex model to give a statistically significant improvement in fit. In addition, we calculate for each dataset the lengths of first axes in detrended correspondence analysis (DCA), another commonly-used method to choose between linear and unimodal modelling (Lepš and Šmilauer, 2003). The DCA first axis lengths are shortest with the two chironomid datasets (Table 2). With these datasets the predominant

determined response, among taxa showing any significant response (i.e., the final fitted model type is other than I) is sigmoidal (Table 2; HOF model type II or III). A short DCA axis length (1.90) is found with **Chi–depth**, and we thus choose PLS with this dataset, as done in the original work of Engels et al. (2012). The authors of **Chi–Tjul** consider their dataset a borderline case between linear and unimodal modelling (Fortin et al., 2015), and here we use both PLS and WA/WA-PLS with this dataset. With the remaining datasets, the DCA axis lengths are longer while the predominant response shape becomes unimodal (Table 1; model type IV or V) and thus WA/WA-PLS is chosen with these datasets.

[Table 2]

Cross-validation schemes

A range of CV approaches are used with the six datasets, with each dataset using a CV type best suited for the spatial structure in the data. CV performance is assessed mainly based on the root-mean-square error of prediction (*RMSEP*), however the maximum bias of each model is also considered. With the two spatially clustered datasets, we perform LOCO CV, with the models in each iteration trained with samples from all but one lake (with **Chi–depth**) or sample cluster (with **Dia–colour**). These models are used to predict the environmental values for a test set consisting of the left-out lake or sample cluster. LOCO CV provides a strong test of model transferability (i.e. generality), particularly since the clusters differ considerably in environmental character in terms of secondary gradients, e.g. with different clusters of **Dia–colour** (Figure 1) sampling different sections of the latitudinal gradient in growing degree days (Wenger and Olden, 2012; le Roux et al., 2013). With the remaining datasets, LOCO CV is not possible, and with these datasets we perform a conventional ten-fold CV. The ten-fold CV is repeated ten times to stabilize the results across CVs using different random divisions of samples to ten groups.

A limitation of ten-fold CV is that it may give an over-optimistic view of predictive ability in the presence of spatial autocorrelation in the calibration data. A k -fold CV is marginally more robust to spatial autocorrelation compared to the commonly-used LOO CV, as pseudoreplicate samples have a small chance ($1/k$) of being removed from the training data, however no explicit attempt is made to remove pseudoreplication (see also Trachsel and Telford, 2015). Of the four datasets here using ten-fold CV, three describe the response of diatom communities to water chemistry parameters (**Dia-pH_UK**, **Dia-pH_EU**, **Dia-logEC**). Such datasets are typically little affected by spatial autocorrelation, as the water chemistries of lakes are individualistic even over small distances (Telford and Birks, 2009; Trachsel and Telford, 2015), and ten-fold CV is thus likely to give a robust estimate of predictive ability with these datasets. The situation may however be different for the final dataset (**Chi-Tjul**), as the environmental drivers include a continental-scale climatic gradient. We thus supplement the ten-fold CV for **Chi-Tjul** with an h -block CV, a CV approach robust to spatial autocorrelation (Telford and Birks, 2009). In h -block CV, observations within h km of the test observation are omitted during the CV. A challenge with h -block CV is the selection of a correct value for h : if h is too small, the CV performance may remain inflated by spatial autocorrelation in nuisance variables, while if h is too large, predictive ability is hampered due to lack of data (Trachsel and Telford, 2015). Here we run h -block CV at a range of h from 100 to 1000 km at 100 km increments, to examine the behaviour of the calibration models as samples from an increasingly large neighbourhood are omitted. As a guide to assessing the results, we also estimate the optimal h following Telford and Birks (2009), who suggest estimating h as the range of a circular variogram fitted to the residuals of a WA model in LOO CV. Trachsel and Telford (2015) suggest a complementary approach to estimate the correct h . This involves creating simulated variables with the same autocorrelation structure as the variable of interest, running h -block CV for these simulated

variables using a range of h , and using the h for which the cross-validated coefficient of determination (r^2) is closest to the r^2 between the simulated variables and the variable of interest. However, this approach cannot be implemented for our ensemble-learner methods, due to the prohibitive calculation requirements of running h -block CVs for a range of simulated variables.

Implementation

All analyses were performed in R (version 3.0.3; R Core Team, 2014). R code for implementing different cross-validation approaches and palaeoreconstructions with these calibration methods is included in Supplementary Data.

PLS, WA, and WA-PLS calibrations were prepared using the package RIOJA (version 0.8-7; Juggins, 2013b) with square-root transformed species data. The methods were run in several variations, using WA with inverse, classical and monotonic deshrinking, and each of these with and without tolerance down-weighting, while PLS and WA-PLS were run using one, two and three-component models. Results for the best-performing PLS or WA/WA-PLS model are shown with each dataset. The van der Voet (1994) t -test was used to choose between PLS and WA-PLS models with different numbers of components.

Bagging was implemented using the package RPART (version 4.1-8; Therneau et al., 2014) to prepare the individual (unpruned) tree models; for the bootstrap aggregation routine see Supplementary Data. RaF models were prepared using RANDOMFOREST (version 4.6-10; Liaw and Wiener, 2002) and BRTs with GBM (version 2.1; Ridgeway, 2013). The ensemble size used with bagging, RaF, and RoF was mostly 500 trees, however with some larger datasets a 100-tree ensemble was used with RoF due to computer memory requirements. BRT parameters were determined by running CVs with a grid of combinations for

learning rate (LR), tree complexity (TC), bagging fraction (BF), and maximum number of trees (MT).

The RoF algorithm has not been available for R, and we publish an implementation here (Supplementary Data). In addition to ensemble size (L), RoF requires a second parameter, the number of groups K in which the predictors are passed through the PCA. Kuncheva and Rodríguez (2007) tested RoF with 32 datasets, and did not find a pattern for best K and were thus unable to recommend a value. We tested the effect of K by running CVs for each dataset with a range of K values, with the K producing the lowest mean $RMSEP$ chosen for the final analysis. We further explored the effect of ensemble size by running CVs for each dataset using a range of L .

Results and discussion

Model parameterization

The best-performing calibration method parameterizations for each dataset are shown in Table 3. Among parametric methods, a one-component PLS model was chosen for **Chi-depth**. With two of the datasets selected for unimodal modelling (**Dia-pH_UK**, **Dia-pH_EU**), two-component WA-PLS showed sufficient reduction in $RMSEP$ and was chosen over WA, while the remaining datasets use different variations of WA. In CVs with different BRT parameterizations, we found lowest $RMSEP$ using a low LR and allowing the ensembles to grow large ($MT = 10,000$ or $20,000$ trees). With TC , some initial improvement in predictive ability was found with all datasets going from $TC = 1$ to larger values, with $RMSEP$ falling with increasing TC until stabilizing at a TC ranging from 2 to 12, depending on dataset. This supports the suggestion of Schonlau (2005) to use high rather than low TC when in doubt. We found occasionally considerable differences in $RMSEP$ with $BF = 0.75$ compared to $BF = 0.5$, and thus we

recommend testing the effect of *BF*.

[Table 3]

The effect of RoF parameters on cross-validated *RMSEP* is plotted in Figure 2. All datasets show an improvement in predictions going from $K = 2$ to larger values (Figure 2, upper panels), while increasing K starts to eventually hurt performance with all datasets. However, the K giving best CV performance varies with dataset, and no clear relationship is seen between best K and either the number of predictors or the number of samples. Like Kuncheva and Rodríguez (2007) we thus cannot recommend a single value to use. However, we note that the sensitivity of *RMSEP* to K is not great and for most datasets used here any K between 5 and 10 would allow close to optimal performance. While we here mostly used ensembles of 500 trees, the plots of *RMSEP* as function of L (Figure 2, lower panels) show that with these datasets *RMSEP* tends to stabilize at a lower L of ca. 50.

[Figure 2]

Model performance

CV results for the calibration models are presented in Figure 3. With one dataset (**Chi-depth**), the parametric PLS transfer function ($RMSEP = 1.61$ m) improves on all tree-ensemble approaches ($RMSEP$ between 1.68 m (RoF) and 1.83 m (BRT)). With the **Dia-pH_UK**, **Dia-pH_EU**, and **Dia-logEC** datasets, the parametric transfer function used (WA or WA-PLS) has near-identical performance with the best performing tree-ensemble approach (either BRT or RaF). With **Chi-Tjul** and **Dia-colour** datasets, all tree-ensemble methods improve on the parametric transfer function used (WA). With **Dia-colour**, the

difference between the best tree-ensemble method and WA is quite small ($RMSEP = 52.5$ mgPt/L with BRT and 57.7 mgPt/L with WA), while a larger difference is found with **Chi-Tjul** ($RMSEP = 1.65$ °C with RaF and 1.92 °C with WA). With **Dia-colour**, all methods show a relatively poor performance, possibly owing to the relatively weak signal of water colour in the diatom data, as evidenced by the proportion of non-informative taxa in HOF modelling (Table 2). However, for all methods the cross-validated $RMSEP$ (52.5 – 57.7 mgPt/L) is smaller than the standard deviation of water colour in the calibration data (65.4 mgPt/L). The diatom–water colour models especially struggle to predict for the samples with high observed water colour values of ≥ 200 mgPt/L, with all predicted values falling well below the observations, while for the low end of the water colour gradient (observed values < 200 mgPt/L), especially the best models produce a satisfactory pattern around the 1:1 line in the observed–predicted plots.

[Figure 3]

Results of the complementary h -block CVs for **Chi-Tjul** are shown in Figure 4. For all calibration methods, $RMSEP$ increases between $h = 0$ km and $h = 200$ km, where $RMSEP$ remains constant until $h = 500$ km. With h larger than 500 km, $RMSEP$ for tree-ensemble techniques increases again, whereas $RMSEP$ of WA with monotonic deshrinking (which is conceptually related to two-component WAPLS) only starts increasing with h larger than 800 km. The results of h -block cross-validation were as expected for datasets affected by spatial autocorrelation. The increase of $RMSEP$ at short distances of h is indicative of performance estimates affected by spatial autocorrelation. Removing spatially close samples, pseudoreplicates that are not fully independent of the test sample are removed from the training set. As lack of independence increases transfer function performance (lower $RMSEP$), removal of spatially close samples leads to increased $RMSEP$. The increase of $RMSEP$ with large values of h is caused by the

reduction of data and thereby information available (Trachsel and Telford, 2015). Between the two distances of h with increasing $RMSEP$, i.e. at h between 200 and 500 km, $RMSEP$ is largely unaffected by h . This region is expected to show unbiased performance estimates, avoiding use of pseudoreplicates but still having enough information to make meaningful inference. The value of h estimated with the method proposed by Telford and Birks (2009) was of 475–525 km (Fig 4) and thereby at the upper end of the lengths of h where $RMSEP$ was not affected by the removal distance. Trachsel and Telford (2015) found that estimates of h obtained using a circular variogram were usually larger than estimates of h obtained using other methods. Hence h of 475–525 km is certainly conservative. In conclusion, the finding based on ten-fold CV of RaF showing lowest $RMSEP$ for **Chi–Tjul** is not changed using h -block CV. However, the performance gap between WA and best tree-ensemble methods narrows somewhat in h -block vs. ten-fold CV. This is expected, as tree ensemble methods predict based on a subset of training samples residing in a single terminal tree node, and thus they benefit more of individual pseudoreplicate samples compared to WA which predicts using taxon responses estimated based on the entire training data.

[Figure 4]

Discussion

This is the first time that different tree-ensemble techniques are compared in fossil proxy calibration, and based on these findings, we can make some generic recommendations about the use of these methods. Among the tree-ensemble approaches tested here, bagging constitutes a baseline algorithm which the other methods build upon. In our results, we find bagging to have the worst performance among tree-ensemble methods for all but one dataset (**Chi–depth**). A noteworthy feature in the model predictions with bagging are the gaps in the distributions of predictions seen with a number of the datasets, e.g., at ca.

6 with **UK_Dia-pH**, at 2.9–3.4 with **Dia-logEC**, and at ca. 7 and 10 °C with **Chi-Tjul** (Figure 3). Thus with these datasets, tree building from bootstrap samples does not appear fully sufficient to ensure diverse predictions, and especially with the smallest dataset used (**Dia-logEC**) the predictions remain clearly semi-discrete. By comparison, these gaps in predictions disappear or become less conspicuous with RaF and RoF (Figure 3) which add further sources of stochasticity into the model building. In view of these results, we cannot recommend the use of bagging in palaeolimnological proxy calibration. By comparison, we find little to choose between RaF, RoF, and BRT, as they show near-identical performance with most datasets, and each of them has best performance among tree-ensemble methods with at least one dataset. Kuncheva and Rodríguez (2007) tested a number of tree-based algorithms, including bagging, RaF, and RoF (but not BRT) in classification, i.e. with a categorical response. They ran ten-fold CVs using 32 datasets featuring diverse sample sizes and varying combinations of categorical and continuous predictors. In these classification tests, bagging outperformed RaF with some datasets and had only slightly inferior overall performance. However, Kuncheva and Rodríguez (2007) found RoF to improve considerably on both bagging and RaF.

A distinguishing feature between the tree-ensemble methods are the max. bias figures, which are lowest for BRT with all but one dataset (**Chi-depth**). For most models (28 out of 30), the max. bias value comes from a gradient-end segment, i.e. either the first or the last of the ten segments used for max. bias calculation. Looking at the predicted–observed plots (Figure 3), the tendency to overestimate for low observed values and underestimate for high observed values is clearly present in all tree-ensemble models. Bias towards dataset mean at gradient ends is an inevitable feature of tree-ensemble predictors, as with gradient-end samples the trees become increasingly likely predict using samples located towards the centre of the gradient. Based on the max. bias figures, BRT appears somewhat more robust to this effect, and this is particularly well seen in the gradient-end predictions with **Dia-pH_EU** where BRT

considerably improves on the other tree-ensemble approaches. However, we find parametric transfer functions which, unlike tree ensembles, have some ability to extrapolate beyond the calibration data (Birks et al., 2010), to have the lowest max. bias with all datasets.

We note three methodological features which set tree ensembles apart from transfer-function approaches such as WA/PLS/WA-PLS, and which should be considered in palaeolimnological applications. *First*, none of the tree-ensemble methods yield exactly the same results with repeated runs (unlike e.g. PLS/WA/WA-PLS), due to the random selection of samples and possibly also predictors during tree building. This causes some instability in the predictions, and also in CV metrics like *RMSEP*, even in CV schemes using a predetermined division of samples into test and training sets (LOO, *h*-block, LOCO) instead of a randomized grouping (*k*-fold). However, in ten repeated runs of the LOCO CV using **Chi-depth**, we found relatively small ranges of variation in *RMSEP* with bagging (1.73–1.75 m), RaF (1.77–1.83 m), RoF (1.68–1.70 m), and BRT (1.80–1.85 m), supporting our conclusions about the relative performance of the methods. With **Dia-colour** the relative variation in *RMSEP* was smaller than with **Chi-depth** with all tree-ensemble methods, so it appears that the individual trees became more stable with the larger dataset. *Second*, tree ensembles are not affected by commonly-used species-data transformations, as the individual tree splits are determined by the rank order for each predictor, and this order is preserved in monotonic transformations such as the square-root transformation (De'ath and Fabricius, 2000). *Third*, all predictor variables need not be continuous (e.g., taxon percentages), but any mixture of continuous, binary and categorical predictor variables can be employed (Elith et al., 2008). This could be useful in incorporating non-continuous supplementary variables (e.g., observations of botanical macroremains) in environmental reconstructions (see Salonen et al. (2014) for further discussion). Likewise, the response (environmental) variable can be binary or categorical instead of continuous (De'ath and Fabricius, 2000).

A striking feature in our results is the inability of tree ensembles to compete with a simple, one-component PLS transfer function with **Chi–depth**. We suggest three possible contributors to this. *First*, many of the most abundant chironomid taxa show a strong, linear or otherwise monotonic responses to water depth (Engels and Cwynar, 2011; Table 1), and such relative absence of nonlinearities or interactions would diminish many of the intrinsic advantages of tree-based modelling while linearity (as in PLS) may prove an adequate approximation (De’ath and Fabricius, 2000; Schonlau, 2005). *Second*, this dataset is one of the smallest used here, and especially with one cluster omitted in LOCO CV the data may become sparse enough to disadvantage tree-based methods, which base each prediction on a small subset of samples residing in a terminal tree node, and thus require densely sampled calibration data to populate the gradient evenly with terminal nodes for accurate prediction (Hjort and Marmion, 2008; Wisz et al., 2008). By comparison, a method such as PLS or WA/WA-PLS which estimate taxon responses using the entire training set, may prove more robust to gaps in gradient sampling. This is also a possible explanation for the superior performance of WA compared to tree-ensemble methods with large values of h (> 500 km) in h -block CV (Figure 4) for the **Chi–Tjul** dataset, where the performance of all methods starts to weaken due to poor sampling of the gradient, but WA suffers considerably less, and starts to outperform all tree-ensemble approaches by $h = 700$ km. *Third*, the water-depth signal appears to be strong in the chironomid data, as a comparatively small proportion of taxa shows no significant response in HOF modelling. By contrast, we find all tree-ensemble methods to improve on WA with **Dia–colour**, the dataset with the largest proportion of non-informative taxa (Table 1). A high number of non-informative taxa might favour the tree-based models which are inherently selective in identifying and employing the best indicator taxa while ignoring taxa which have no indicator value (Juggins et al., 2015).

Conclusion

In practice, a proxy-calibration method must often be chosen without access to fully independent test sets, and assessing the relative strengths of different methods can be challenging (Juggins and Birks, 2012; Trachsel and Telford, 2015). Thus, it would be useful to identify guidelines for recognizing calibration scenarios (e.g., dataset size, distribution of response shapes, or the relative influence of the reconstructed variable) which might favour a specific method family, such as parametric transfer functions or a tree-based approach. Here we have shown that tree-ensemble methods can in some cases outperform a generally robust transfer-function technique, especially with larger datasets characterized by complex and diverse taxon responses and abundant noise. However, parametric transfer functions often show near-equal performance or even improve on tree ensembles, particularly with smaller datasets and data characterized by strong, linear species responses. Considering this pattern, it seems prudent to expand the testing of tree-ensemble approaches to even larger and more complex calibration datasets, such as the continental-scale pollen–climate datasets commonly used in palaeoclimate studies. We encourage researchers to further test these methods using a range of new proxy datasets representing different taxa, environments, and spatio-temporal scales, to explore the potential and limitations of these methods, and to identify scenarios and applications where they may prove especially useful.

Acknowledgements

SE is grateful to Les C Cwynar for support in producing the Massachusetts chironomid data. We thank Andrew Scott Medeiros, Keely Mills and Jan Weckström for their help in acquiring data for this study.

Funding

This work was funded by the Academy of Finland program on long-term environmental changes (project no. 278692). JSS acknowledges additional funding from the Finnish Cultural Foundation. AJV is supported by a Canadian Institute for Health Research (CIHR) grant (MOP-97939) and MT by the Norwegian Research Council FriMedBio project palaeoDrivers (213607).

References

- Battarbee RW, Flower FJ, Juggins S et al. (1997) The relationship between diatoms and surface water quality in the Høylandet area of Nord-Trøndelag, Norway. *Hydrobiologia* 348: 69–80.
- Birks HJB, Line JM, Juggins S et al. (1990a) Diatoms and pH reconstruction. *Philosophical Transactions of the Royal Society of London B* 327: 263–278.
- Birks HJB, Juggins S, Line JM (1990b) Lake surface-water chemistry reconstructions from palaeolimnological data. In: Mason BJ (ed) *The Surface Waters Acidification Programme*. Cambridge: Cambridge University Press, pp. 301–313.
- Birks HJB, Heiri O, Seppä H et al. (2010) Strengths and weaknesses of quantitative climate reconstructions based on late-Quaternary biological proxies. *The Open Ecology Journal* 3: 68–110.
- Breiman L (1996) Bagging predictors. *Machine Learning* 24: 123–140.
- Breiman L (2001) Random forests. *Machine Learning* 45: 5–32.

- De'ath G (2007) Boosted trees for ecological modeling and prediction. *Ecology* 88: 243–251.
- De'ath G and Fabricius KE (2000) Classification and regression trees: a powerful yet simple technique for ecological data analysis. *Ecology* 81: 3178–3192.
- Elith J, Graham CH and Anderson RP (2006) Novel methods improve prediction of species' distributions from occurrence data. *Ecography* 29: 129–151.
- Elith J, Leathwick JR and Hastie T (2008) A working guide to boosted regression trees. *Journal of Animal Ecology* 77: 802–813.
- Engels S and Cwynar LC (2011) Changes in fossil chironomid remains along a depth gradient: evidence for common faunal thresholds within lakes. *Hydrobiologia* 665: 15–38.
- Engels S, Cwynar LC, Rees ABH et al. (2012) Chironomid-based water depth reconstructions: an independent evaluation of site-specific and local inference models. *Journal of Paleolimnology* 48: 693–709.
- Felde VA, Bjune AE, Grytnes J-A et al. (2014) A comparison of novel and traditional numerical methods for the analysis of modern pollen assemblages from major vegetation–landform types. *Review of Palaeobotany and Palynology* 210: 22–36.
- Fortin M-C, Medeiros AS, Gajewski K, et al. (2015) Chironomid-environment relations in northern North

America. *Journal of Paleolimnology* 54: 223–237.

Goring S, Lacourse T, Pellatt MG et al. (2010) Are pollen-based climate models improved by combining surface samples from soil and lacustrine substrates. *Review of Palaeobotany and Palynology* 162: 203–212.

Hjort JK and Marmion M (2008) Effects of sample size on the accuracy of geomorphological models. *Geomorphology* 102: 341–350.

Huisman J, Olff H and Fresco LFM (1993) A hierarchical set of models for species response analysis. *Journal of Vegetation Science* 4: 37–46.

Jansen F and Oksanen J (2013) How to model species responses along ecological gradients – Huisman-Olff-Fresco models revisited. *Journal of Vegetation Science* 24: 1108–1117.

Juggins S (2013a). Quantitative reconstructions in palaeolimnology: new paradigm or sick science? *Quaternary Science Reviews* 64: 20–32.

Juggins S (2013b) rioja: Analysis of Quaternary science data. <http://cran.r849project.org/web/packages/rioja/index.html> [18 December 2013]

Juggins S and Birks HJB (2012) Quantitative environmental reconstructions from biological data. In: Birks HJB, Lotter AF, Juggins S et al. (eds) *Tracking environmental change using lake sediments, volume 5: data handling and numerical techniques*. Dordrecht: Springer, pp. 431–494.

Juggins S, Simpson GL and Telford RJ (2015) Taxon selection using statistical learning techniques to improve transfer function prediction. *The Holocene* 25: 130–136.

Kuncheva LI and Rodríguez JJ (2007) An experimental study on rotation forest ensembles. In: *Multiple classifier systems*. Berlin, Heidelberg: Springer, pp. 459–468.

Lepš J and Šmilauer P (2003) *Multivariate analysis of ecological data using CANOCO*. Cambridge: Cambridge University Press, 269 pp.

le Roux PC, Lenoir J, Pelissier L et al. (2013) Horizontal, but not vertical, biotic interactions affect fine-scale plant distribution patterns in a low energy system. *Ecology* 94: 671–682.

Li J, Xu Q, Zheng Z et al. (2015) Assessing the importance of climate variables for the spatial distribution of modern pollen data in China. *Quaternary Research* 83: 289–297.

Liaw A and Wiener M (2002) Classification and regression by randomForest. *R News* 2(3), 18–22.

Mills K and Ryves DB (2012) Diatom-based models for inferring past water chemistry in western Ugandan crater lakes. *Journal of Paleolimnology* 48: 383–399.

Pajunen V, Luoto M and Soininen J (2015) Climate is an important driver for stream diatom distributions. *Global Ecology and Biogeography*, in press. DOI: 10.1111/geb.12399

- R Core Team (2014) *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. <http://www.R-project.org/>
- Ridgeway G (2013) *gbm: generalized boosted regression models. R package version 2.1*. <http://CRAN.R-project.org/package=gbm> [27 January 2015]
- Rodríguez JJ, Kunecheva LI and Alonso CJ (2006) Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28: 1619–1630.
- Salonen JS, Seppä H, Luoto M et al. (2012) A North European pollen–climate calibration set: analysing the climatic responses of a biological proxy using novel regression tree methods. *Quaternary Science Reviews* 45: 95–110.
- Salonen JS, Luoto M, Alenius T et al. (2014) Reconstructing palaeoclimatic variables from fossil pollen using boosted regression trees: Comparison and synthesis with other quantitative reconstruction methods. *Quaternary Science Reviews* 88: 69–81.
- Schonlau M (2005) Boosted regression (boosting): an introductory tutorial and a Stata plugin. *The Stata Journal* 5: 330–354.
- Simpson G and Birks H (2012) Statistical learning in palaeolimnology. In: Birks HJB, Lotter AF, Juggins S et al. (eds) *Tracking environmental change using lake sediments: data handling and numerical techniques*. Dordrecht: Springer, pp. 249–327.

Soininen J, Paavola R and Muotka T (2004) Benthic diatom communities in boreal streams: community structure in relation to environmental and spatial gradients. *Ecography* 27: 330–342.

Steinberg C (2003) *Ecology of humic substances in freshwaters: determinants from geochemistry to ecological niches*. Berlin: Springer Science and Business Media.

Stevenson AC, Juggins S, Birks HJB, et al. (1991) *The Surface Waters Acidification Project palaeolimnology programme: modern diatom/lake-water chemistry data-set*. ENSIS, London.

Telford RJ and Birks HJB (2009) Evaluation of transfer functions in spatially structured environments. *Quaternary Science Reviews* 28: 1309–1316.

Telford RJ and Birks HJB (2011) A novel method for assessing the statistical significance of quantitative reconstructions inferred from biotic assemblages. *Quaternary Science Reviews* 30: 1272–1278.

ter Braak CJF and Juggins S (1993) Weighted averaging partial least squares regression (WA-PLS): An improved method for reconstructing environmental variables from species assemblages. *Hydrobiologia* 269/270: 485–502.

Therneau T, Atkinson B and Ripley B (2014) *Recursive partitioning and regression trees*. <http://cran.r-project.org/web/packages/rpart/index.html> [12 January 2015]

Trachsel M and Telford RJ (2015) Technical Note: Estimating unbiased transfer-function performances in spatially structured environments. *Climate of the Past Discussions* 11: 4729–4749.

van der Voet H (1994) Comparing the predictive accuracy of models using a simple randomization test. *Chemometrics and Intelligent Laboratory Systems* 25: 313–323.

Veloz SD, Williams JW, Blois JL et al. (2012) No-analog climates and shifting realized niches during the late quaternary: implications for 21st-century predictions by species distribution models. *Global Change Biology* 18: 1698–1713.

Wenger SI and Olden JD (2012) Assessing transferability of ecological models: an underappreciated aspect of statistical validation. *Methods in Ecology and Evolution* 3: 260–267.

Wisn MS, Hijmans RJ, Li J et al. (2008) Effects of sample size on the performance of species distribution models. *Diversity and Distributions* 14: 763–773.

Zhao Y, Liu H, Li F et al. (2012) Application and limitations of the *Artemisia*/Chenopodiaceae pollen ratio in arid and semi-arid China. *The Holocene* 22: 1385–1392.

Zimmerman NE, Edwards TC, Graham CH et al. (2010) New trends in species distribution modelling. *Ecography* 33: 985–989.

Table 1. Details and references for the datasets used.

Code	Location	Predictor	Response	<i>n</i> samples	<i>n</i> predictors	Gradient span	Cross- validation type	References
Dia-pH_UK	United Kingdom	Diatoms	Water pH	167	277	4.3–7.3	ten-fold (ten repeats)	Surface Waters Acidification Programme (Birks et al., 1990; Stevenson et al., 1991)
Dia-pH_EU	Europe	Diatoms	Water pH	622	652	4.3–8.4	ten-fold (ten repeats)	European Diatom Database Initiative (http://craticula.ncl.ac.uk/Eddi/jsp/index.jsp)
Dia-logEC	Uganda	Diatoms	Water log electrical conductivity	69	265	1.8–4.2	ten-fold (ten repeats)	Mills and Ryves, 2012
Chi-Tjul	North America	Chironomids	Mean July air temperature (°C)	434	78	2.0–16.3	ten-fold (ten repeats), <i>h</i> -block	Fortin et al., 2015
Chi-depth	Massachusetts	Chironomids	Water depth (m)	98	82	0.3–9.3	Leave-one-cluster-out	Engels and Cwynar, 2011
Dia-colour	Finland	Diatoms	Water colour (mgPt/L)	225	214	5–350	Leave-one-cluster-out	Soininen et al., 2004; Pajunen et al., 2015

Table 2. Lengths of the first axes in detrended correspondence analysis (DCA1) in standard deviation units and the distribution of best-fitting HOF model types (Huisman et al., 1993) for the species data in each dataset (Table 1). All taxa occurring in at least 10 samples are included in the HOF modelling. The datasets are shown ordered according to increasing DCA1 length.

Dataset	DCA1 length	HOF type I (no significant response)	HOF type II (sigmoidal)	HOF type III (sigmoidal, with plateau)	HOF type IV (unimodal)	HOF type V (unimodal, skewed)
Chi-depth	1.9	5 (9.4 %)	21 (39.6 %)	6 (11.3 %)	13 (24.5 %)	8 (15.1 %)
Chi-Tjul	3.2	4 (6.8 %)	13 (22.0 %)	24 (40.7 %)	1 (1.7 %)	17 (28.8 %)
Dia-colour	3.4	46 (27.7 %)	34 (20.5 %)	7 (4.2 %)	31 (18.7 %)	48 (28.9 %)
Dia-pH_UK	4.3	45 (23.6 %)	62 (32.5 %)	9 (4.7 %)	45 (23.6 %)	30 (15.7 %)
Dia-pH_EU	4.4	66 (18.2 %)	80 (22.0 %)	26 (7.2 %)	76 (20.9 %)	115 (31.7 %)
Dia-logEC	5.6	7 (13.7 %)	5 (9.8 %)	7 (13.7 %)	11 (21.6 %)	21 (41.2 %)

Table 3. Parameterization of calibration methods for each dataset (Table 1). Abbreviations: WA = weighted averaging, PLS = partial least squares, TC = tree complexity, LR = learning rate, BF = bagging fraction, MT = maximum number of trees. For explanation of the rotation forest K parameter, see text.

Dataset	Parametric method	Bagging	Random forest	Rotation forest	Boosted regression tree
Dia-pH_UK	WA-PLS (2 components)	500 trees	500 trees	500 trees, $K = 4$	$TC = 4$, $LR = 0.0005$, $BF = 0.75$, $MT = 20,000$
Dia-pH_EU	WA-PLS (2 components)	500 trees	500 trees	100 trees, $K = 11$	$TC = 12$, $LR = 0.001$, $BF = 0.5$, $MT = 20,000$
Dia-logEC	WA (inverse deshrinking)	500 trees	500 trees	500 trees, $K = 3$	$TC = 2$, $LR = 0.001$, $BF = 0.75$, $MT = 20,000$
Chi-Tjul	WA (monotonic deshrinking, tolerance downweighting)	500 trees	500 trees	100 trees, $K = 7$	$TC = 5$, $LR = 0.001$, $BF = 0.5$, $MT = 20,000$
Chi-depth	PLS (1 component)	500 trees	500 trees	500 trees, $K = 5$	$TC = 4$, $LR = 0.0005$, $BF = 0.75$, $MT = 10,000$
Dia-colour	WA (inverse deshrinking)	500 trees	500 trees	500 trees, $K = 25$	$TC = 4$, $LR = 0.001$, $BF = 0.75$, $MT = 10,000$

Figure 1. Maps of datasets **Chi-depth (A)** and **Dia-colour (B)** (see Table 1 for details). Different symbols indicate spatial cluster membership. Numbers of samples in each spatial cluster are indicated.

Figure 2. Effect of rotation forest parameters on predictive performance. The upper panels show the effect of K (the number of groups in which the predictor set is passed through principal component analysis) on cross-validated root-mean-square error of prediction ($RMSEP$) while using ensembles of ten trees. The K producing the lowest $RMSEP$ is indicated with a vertical line. The lower panels show the effect of L (ensemble size) on $RMSEP$, while using the optimal K . Results are shown for six datasets (see Table 1 for details). All plots are based on running ten repeats of leave-one-cluster-out (for **Chi-depth** and **Dia-colour**) or ten repeats of ten-fold (for other datasets) cross-validations with each K or L tested, with mean $RMSEP$ reported for each K or L .

Figure 3. Model performance in cross-validation. The models are shown for each dataset ordered according to increasing root-mean-square error of prediction ($RMSEP$). Results are shown for six datasets (see Table 1 for details). For each dataset, results are shown using a parametric transfer function (one of partial least squares (PLS), weighted averaging (WA), or weighted averaging-partial least squares (WA-PLS)), bagging, random forest (RaF), rotation forest (RoF), and a boosted regression tree (BRT). Each plot shows the sample-specific predictions vs. the observed environmental values. Indicated within each plot are cross-validation performance statistics including the $RMSEP$ and the maximum bias. The cross-validation type used is leave-one-cluster-out for **Chi-depth** and **Dia-colour**, while the predictions and performance statistics for the other datasets are the means from ten runs of ten-fold cross-validation. The sample symbols for **Chi-depth** and **Dia-colour** indicate spatial sample cluster membership (see Figure 1).

Figure 4. Cross-validated root-mean-square error of prediction (*RMSEP*) for the dataset **Chi–Tjul** (see Table 1 for details). Results are shown with h -block cross-validations (Telford and Birks, 2009) using a range of values between 100 and 1000 km for h , while the results for $h = 0$ km are from a ten-fold cross-validation repeated ten times. The h value suggested by the test of Telford and Birks (2009) for an unbiased estimate of predictive ability (see text for details) is shown highlighted.

Figure 1

Figure 2

Figure 3

Figure 4

```
#####
#
# SUPPLEMENTARY DATA
#
# Salonen JS, Verster AJ, Engels S, et al. (xxx) Calibrating aquatic
# microfossil proxies with regression-tree ensembles: cross-validation
# with modern chironomid and diatom data. The Holocene (submitted).
#
# ---
#
# This file includes code for:
# - Running cross-validations (CVs)-- leave-one-out, leave-one-group-out, or
# h-block -- with a range of transfer function and tree-ensemble methods.
# - Predicting for new species data (e.g. "palaeo-reconstruction"), using
# the same methods.
# - Testing the effect of rotation forest K and L parameters on RMSEP
# in leave-one-group-out CVs.
#
# If you use this code in published work, please cite the original paper.
#
# NOTE ON CALCULATION TIMES: the CV loops can take a long time to run (from
# hours to days), especially with large datasets. All the CV loops have been
# rigged with code which estimates and reports the remaining calculation
# time after each iteration. If the calculation time looks unacceptable,
# numerous avenues exist for quickening the run, e.g.,
# - using a lower no. of folds in cross-validation
# - using smaller ensemble sizes (bagging, RaF, RoF)
# - using a suboptimal maximum allowed ensemble size with BRT
#####

library (rioja)
library (randomForest)
library (rpart)
library (gbm)
library (caret)

source ("RotationForest.R")

# A. Helper function for calculating CV statistics.
#
# Calculates some CV statistics (correlation, RMSEP, max. bias) based on
# vectors on observations and predictions. The results are returned as a
# list. The correlation method used ("pearson", "spearman", or "kendall")
# is given as a parameter (default = "pearson").

cv.stat.calc <- function (obs, pred, cor.method="pearson") {

  stats <- list()

  stats$Cor <- cor(x=obs, y=pred, method=cor.method)
  stats$RMSEP <- mean((pred-obs)^2)^(1/2)

  biases <- numeric() # Maximum bias calculation
  residuals <- pred-obs
  bias.interval <- (max(obs) - min(obs))/10
  for (i in 1:10) {
    interval.start <- min(obs)+(i-1)*bias.interval
    interval.end <- interval.start + bias.interval
    if (i != 10) {
      indices <- which (obs >= interval.start & obs < interval.end)
    } else {
      indices <- which (obs >= interval.start & obs <= interval.end)
    }
    biases[i] <- mean (residuals[indices])
  }
  stats$Max.bias <- max(abs(biases))
}
```

```

stats
}

# B. Function for testing effect of rotation forest K parameter.
#
# Runs a desired number of leave-one-group-out CVs with each K, calculates RMSEP.
#
# Parameters:
# - pred: a data frame of predictors
# - resp: a data frame with a response column (first column is used)
# - k.min, k.max: smallest and largest K to test (defaults to 2 and 20, respectively)
# - n.trees: size of ensemble to use (default=10)
# - cv.folds: how many CV folds (not used if "grouping" is provided; default=10)
# - cv.repeat: how many repeats of the CV with each K (default=5)
# - grouping: an integer vector indicating a predefined grouping of the samples
#               for the CV (if provided will override cv.folds)
# - no.plot: set to TRUE to suppress plotting (default=FALSE)
#
# The function plots (mean) RMSEP as function of K, and returns a list with the
# following elements:
# - k.best: the K value producing the lowest RMSEP
# - k.test.rmsep: a matrix of RMSEP values from the individual runs
# - k.test.pred: a three-dimensional array of CV predictions (for further analyses)

rof.k.test <- function (pred, resp, k.min=2, k.max=20, n.trees=10, cv.folds=10,
cv.repeat=5, grouping=NA, no.plot=FALSE) {
  require (caret)
  run.k.fold <- length(grouping)==1
  k.test.rmsep <- matrix (nrow=cv.repeat, ncol=k.max-k.min+1, dimnames=list(paste
("iter", 1:cv.repeat, sep=""), paste ("k", k.min:k.max, sep="")))
  k.test.pred <- array (dim=c(nrow(pred), k.max-k.min+1, cv.repeat),
dimnames=list(paste ("sample", 1:nrow(pred), sep=""), paste ("k", k.min:k.max,
sep=""), paste ("iter", 1:cv.repeat, sep="")))
  start.time <- Sys.time()
  for (i in 1:cv.repeat) { # Repeats n times the CV for each K
    if (run.k.fold) {
      grouping=numeric()
      folds <- createFolds (1:nrow(pred), k=cv.folds)
      for (j in 1:cv.folds) grouping[folds[[j]]] <- j
    }
    for (j in k.min:k.max) { # Runs a single CV for each K
      cv.pred <- rep (NA, nrow(pred))
      for (k in 1:max(grouping)) { # Runs a single CV
        new.rows <- grouping==k
        pred.iter <- pred[!new.rows,]
        resp.iter <- resp[!new.rows,1,drop=FALSE]
        new.iter <- pred[new.rows,]
        zeros <- colSums(pred.iter)==0
        pred.iter <- pred.iter[,!zeros]
        new.iter <- new.iter[,!zeros]
        rof.cv.model <- RotationForest(Df.X=pred.iter, Df.Y=resp.iter, kuse=j,
juse=n.trees)
        rof.preds <- predict.RotationForest (rof.cv.model, Df.X=new.iter)
        cv.pred[new.rows] <- rof.preds
      }
      iter.rmsep <- mean((cv.pred-resp[,1])^2)^(1/2)
      k.test.rmsep[i,j-k.min+1] <- iter.rmsep
      k.test.pred[j-k.min+1,i] <- cv.pred
      iter.done <- (i-1)*(k.max-k.min+1)+j-(k.min-1)
      iter.todo <- cv.repeat*(k.max-k.min+1) - iter.done
      time.remain <- iter.todo*difftime(Sys.time(), start.time,
units="mins")/iter.done
      cat (iter.done, "/", iter.done+iter.todo, " done (K=",j,",
RMSEP=",iter.rmsep,")", estimated time remaining: ", round(time.remain, 1), "
minutes.\n", sep="")
    }
  }
}

```

```

    }
  }
  k.best <-
(k.min:k.max)[which(colMeans(k.test.rmsep)==min(colMeans(k.test.rmsep)))]
  if (!no.plot) {
    plot (x=k.min:k.max, y=colMeans(k.test.rmsep), type="l", col="red", xlab="K",
ylab="Mean RMSEP", las=1)
    abline (v=k.best, col="green")
  }
  results <- list (k.best=k.best, k.test.rmsep=k.test.rmsep,
k.test.pred=k.test.pred)
  results
}

# C. A function for testing the effect of ensemble size (L) with rotation forest.
#
# Runs a desired number of leave-one-group-out CV with each L, calculates RMSEP.
#
# Parameters:
# - pred: a data frame of predictors
# - resp: a data frame with a response column (first column is used)
# - k.use: rotation forest K parameter to use (default=5)
# - l.values: vector of L values to test (defaults to a set of 27 values between
#           1 and 500)
# - cv.folds: how many CV folds (not used if "grouping" is provided; default=10)
# - cv.repeat: how many repeats of the CV with each K (default=5)
# - grouping: an integer vector indicating a predefined grouping of the samples
#             for the CV (if provided will override cv.folds)
# - no.plot: set to TRUE to suppress plotting (default=FALSE)
#
# The function plots (mean) RMSEP as function of L, and returns a list with the
# following elements:
# - l.test.rmsep: a matrix of RMSEP values from the individual CV runs
# - l.test.pred: a three-dimensional array of CV predictions (for further analyses)

rof.l.test <- function (pred, resp, k.use=5, l.values=c(1:10, seq(20,100,10),
seq(150, 500, 50)), cv.folds=10, cv.repeat=5, grouping=NA, no.plot=FALSE) {
  require (caret)
  run.k.fold <- length(grouping)==1
  l.test.rmsep <- matrix (nrow=cv.repeat, ncol=length(l.values), dimnames=list(paste
("Iter", 1:cv.repeat, sep=""), paste ("L", l.values, sep="")))
  l.test.pred <- array (dim=c(nrow(pred), length(l.values), cv.repeat),
dimnames=list(paste ("sample", 1:nrow(pred), sep=""), paste ("L", l.values, sep=""),
paste ("iter", 1:cv.repeat, sep="")))
  trees.total <- sum(l.values) * cv.repeat * ifelse(run.k.fold, cv.folds,
max(grouping))
  start.time <- Sys.time()
  for (i in 1:cv.repeat) { # Repeats n times the cross-validations for each L
    if (run.k.fold) {
      grouping=numeric()
      folds <- createFolds (1:nrow(pred), k=cv.folds)
      for (j in 1:cv.folds) grouping[folds[[j]]] <- j
    }
    for (j in 1:length(l.values)) { # Runs a single cross-validation for each L
      cv.pred <- rep (NA, nrow(spe))
      for (k in 1:max(grouping)) { # Runs a single cross-validation
        new.rows <- grouping==k
        spe.iter <- spe[!new.rows,]
        env.iter <- env[!new.rows,1,drop=FALSE]
        new.iter <- spe[new.rows,]
        zeros <- colSums(spe.iter)==0
        spe.iter <- spe.iter[,!zeros]
        new.iter <- new.iter[,!zeros]
        rof.cv.model <- RotationForest(Df.X=spe.iter, Df.Y=env.iter, kuse=k.use,
juse=l.values[j])
        rof.preds <- predict.RotationForest (rof.cv.model, Df.X=new.iter)

```

```

        cv.pred[new.rows] <- rof.preds
    }
    iter.rmsep <- mean((cv.pred-resp[,1])^2)^(1/2)
    l.test.rmsep[i,j] <- iter.rmsep
    l.test.pred[,j,i] <- cv.pred
    trees.done <- ((i-1)*sum(l.values)+sum(l.values[1:j])) * ifelse(run.k.fold,
cv.folds, max(grouping))
    trees.todo <- trees.total-trees.done
    iter.done <- (i-1)*(length(l.values))+j
    iter.todo <- cv.repeat*length(l.values) - iter.done
    time.remain <- trees.todo*difftime(Sys.time(), start.time,
units="mins")/trees.done
    cat (iter.done, "/", iter.done+iter.todo, " done (L=",l.values[j],"
RMSEP=",iter.rmsep,"), estimated time remaining: ", round(time.remain, 1), "
minutes.\n", sep="")
    }
}
if (!no.plot)
    plot (x=l.values, y=colMeans(l.test.rmsep, na.rm=TRUE),
xlim=c(0,max(l.values)), type="l", col="red", xlab="L", ylab="Mean RMSEP", las=1)
    results <- list (l.test.rmsep=l.test.rmsep, l.test.pred=l.test.pred)
    results
}

# D. Read data from Excel (via clipboard).

spe <- read.table ("clipboard", header=TRUE)      # Calibration species data columns
                                                # (with column names).

env <- read.table ("clipboard", header=TRUE)      # Calibration env. data column
                                                # (may include several columns, but
                                                # the first column is used; must
                                                # include column name).

spe.new <- read.table ("clipboard", header=TRUE)  # Possible new species data to
                                                # predict for (e.g., palaeo-
                                                # reconstruction). Note that
                                                # the set of species column names
                                                # must PRECISELY match with
                                                # the calibration data (even if a
                                                # given column is full of zeroes
                                                # for the new data).

dst <- read.table ("clipboard", header=TRUE)      # Calibration data distance matrix
                                                # (only needed for h-block CV)

grp <- read.table ("clipboard", header=TRUE)      # Grouping of the calibration
                                                # samples. This should be a single
                                                # integer column. This is only needed
                                                # for running leave-one-group-out CV
                                                # with a pre-defined grouping. If not
                                                # provided, a random grouping is used
                                                # (i.e., k-fold CV).

# E. Create sample grouping.
#
# This is needed if leave-one-group-out (k-fold) CV is run using the below
# code, and if no pre-defined grouping (dataframe "grp") was provided above.

cv.folds <- 10  # set desired k
folds <- createFolds (1:nrow(spe), k=cv.folds)
grp <- numeric()
for (i in 1:cv.folds) grp[folds[[i]]] <- i
grp <- data.frame (grp)

# F. Set parameters for the CV (and reconstruction) runs.

```



```

cv.type <- "lgo"           # Choose CV type. One of "loo" (leave-one-out),
                           # "lgo" (leave-one-group-out, e.g. k-fold), or "h.block".
h.value <- 100             # Value of h for h-block CV.

run.tf <- TRUE             # Run a parametric transfer function (WA, WA-PLS, or PLS)?
run.bagging <- TRUE        # Run bagging?
run.raf <- TRUE            # Run random forest?
run.rof <- TRUE            # Run rotation forest?
run.brt <- TRUE            # Run BRT?

para.method <- "WA"        # Which parametric method to use: "WA", "WA-PLS", or "PLS"
pls.n.comp <- 2            # How many components in the WA-PLS or PLS model
wa.type <- "WA.m"          # Which WA variation: one of "WA.inv", "WA.cla", "WA.m"
                           # "WA.inv.tol", "WA.cla.tol", or "WA.m.tol"

bagging.n.trees <- 500     # Ensemble size in bagging
raf.n.trees <- 500         # Ensemble size in random forest
rof.n.trees <- 500         # Ensemble size in rotation forest
rof.k <- 5                 # K parameter in rotation forest

brt.int.depth <- 2         # Parameters for BRT. For a cross-validation
brt.lrate <- 0.001         # approach for estimating good values, see
brt.max.trees <- 3000      # Elith et al. (2008).
brt.bag.fraction <- 0.5    #
brt.n.cores <- 1           # Number of CPU cores to use in BRT model building.

cv.cor.method <- "pearson" # Which correlation coefficient to report.
                           # One of "pearson", "spearman", and "kendall"

# G. Run CV.

cv.pred <- matrix(nrow=nrow(spe), ncol=5, dimnames=list(paste("Cal", 1:nrow(spe),
sep=""), c("tf", "bagging", "raf", "rof", "brt"))) # CV predictions matrix
cv.stat <- matrix(nrow=3, ncol=5, dimnames=list(c("Cor", "RMSEP", "Max.bias"),
c("tf", "bagging", "raf", "rof", "brt"))) # CV statistics matrix
start.time <- Sys.time()
env.param <- colnames(env)[1]
n.iter <- ifelse(cv.type=="lgo", max(grp), nrow(spe))

for (i in 1:n.iter) {

  if (cv.type == "lgo") {      # Determine test set rows
    new.rows <- grp[,1] == i
  } else
    new.rows <- 1:nrow(spe) == i

  if (cv.type == "h.block") { # Determine cal. test rows
    cal.rows <- !new.rows & dst[,i] > h.value
  } else
    cal.rows <- !new.rows

  spe.iter <- spe[cal.rows,]
  env.iter <- env[cal.rows,env.param,drop=FALSE]
  new.iter <- spe[new.rows,]

  zeros <- colSums(spe.iter)==0
  spe.iter <- spe.iter[,!zeros]
  new.iter <- new.iter[,!zeros]

  # WA/WA-PLS/PLS

  if (run.tf) {

    if (para.method != "WA") { # Using WA-PLS or PLS

```

```

    pls.cv.model <- WAPLS(sqrt(spe.iter), env.iter[,env.param],
iswappls=para.method=="WA-PLS")
    pls.pred <- predict(pls.cv.model, sqrt(new.iter))
    cv.pred[new.rows, "tf"] <- pls.pred$fit[,pls.n.comp]
  }
  else { # using WA
    wa.cv.model <- WA(sqrt(spe.iter), env.iter[,env.param], mono=TRUE, toldW=TRUE)
    wa.pred <- predict(wa.cv.model, sqrt(new.iter))
    cv.pred[new.rows, "tf"] <- wa.pred$fit[,wa.type]
  }
}

# Bagged tree

if (run.bagging) {

bagging.pred <- matrix(nrow=nrow(new.iter), ncol=bagging.n.trees)
for (j in 1:bagging.n.trees) { # Run n iterations with bootstrapped
  # calibration data
  cal.samples.iter <- sample (1:nrow(spe.iter), nrow(spe.iter), replace=TRUE)
  spe.iter.bagging <- spe.iter[cal.samples.iter,]
  env.iter.bagging <- env.iter[cal.samples.iter,env.param,drop=FALSE]
  zeros <- colSums(spe.iter.bagging)==0
  spe.iter.bagging <- spe.iter.bagging[,!zeros]
  new.iter.bagging <- new.iter[,!zeros]
  bagging.formula.iter <- as.formula(paste (env.param,
paste(colnames(spe.iter.bagging), collapse=" + "), sep=" ~ "))
  bagging.model.iter <- rpart (bagging.formula.iter, data=cbind(env.iter.bagging,
spe.iter.bagging))
  bagging.pred[,j] <- predict (bagging.model.iter, new.iter.bagging)
}
cv.pred[new.rows,"bagging"] <- rowMeans (bagging.pred) # Calculate mean
# predictions
}

# Random forest

if (run.raf) {

raf.cv.model <- randomForest (x=spe.iter, y=env.iter[,env.param],
ntree=raf.n.trees)
raf.pred <- predict (raf.cv.model, new.iter)
cv.pred[new.rows,"raf"] <- raf.pred

}

# Rotation forest

if (run.rof) {

rof.cv.model <- RotationForest(Df.X=spe.iter, Df.Y=env.iter, kuse=rof.k,
juse=rof.n.trees)
rof.pred <- predict.RotationForest (rof.cv.model, Df.X=new.iter)
cv.pred[new.rows,"rof"] <- rof.pred

}

# Boosted tree

if (run.brt) {

brt.formula <- as.formula(paste (env.param, "~", paste(colnames(spe.iter),
collapse=" + ")))

```

```

brt.cv.model <- gbm(brt.formula, distribution="gaussian", n.trees=brt.max.trees,
shrinkage=brt.lrate, interaction.depth=brt.int.depth, bag.fraction=brt.bag.fraction,
cv.folds=4, verbose=FALSE, data=cbind(env.iter, spe.iter), n.cores=brt.n.cores)
brt.ntrees <- gbm.perf(brt.cv.model, method="cv", plot.it=FALSE)
brt.pred <- predict(brt.cv.model, new.iter, brt.ntrees, type="response")
cv.pred[new.rows,"brt"] <- brt.pred

}

iter.done <- i
iter.todo <- n.iter-i
time.remain <- iter.todo*difftime(Sys.time(), start.time, units="mins")/iter.done
cat (cv.type, " CV iteration ", iter.done, "/", iter.done+iter.todo, " done,
estimated time remaining: ", round(time.remain, 1), " minutes.\n", sep="")
cat ("Test set n=", sum(new.rows), ", Training set n=", sum(cal.rows), " (omitted
samples: ", paste (which (cal.rows==FALSE), collapse=" "), ")\n", sep="")

}

# H. Calculate CV statistics.

for (i in 1:ncol(cv.pred)) {
  cv.statistics <- cv.stat.calc(env[,env.param], cv.pred[,i],
cor.method=cv.cor.method)
  cv.stat["Cor",i] <- cv.statistics$Cor
  cv.stat["RMSEP",i] <- cv.statistics$RMSEP
  cv.stat["Max.bias",i] <- cv.statistics$Max.bias
}

# I. Save CV results (via clipboard to Excel).

write.table (cv.pred, file="clipboard", sep="\t")
write.table (cv.stat, file="clipboard", sep="\t")

# J. Predict for new data (e.g., "palaeo-reconstruction"). Note that this is
# essentially a simplified version of the above CV loop: here the models are
# trained only once, using the entire calibration data, while the test set
# is replaced by the fossil data (spe.new).

new.pred <- matrix(nrow=nrow(spe.new), ncol=5, dimnames=list(paste("New",
1:nrow(spe.new), sep=""), c("tf", "bagging", "raf", "rof", "brt"))) # Set up result
matrix

if (run.tf) {

  if (para.method != "WA") { # Using WA-PLS or PLS
    pls.pred.model <- WAPLS(sqrt(spe), env[,env.param], iswappls=para.method=="WA-
PLS")
    pls.pred <- predict(pls.pred.model, sqrt(spe.new))
    new.pred[, "tf"] <- pls.pred$fit[,pls.n.comp]
  }
  else { # using WA
    wa.pred.model <- WA(sqrt(spe), env[,env.param], mono=TRUE, toldW=TRUE)
    wa.pred <- predict(wa.pred.model, sqrt(spe.new))
    new.pred[, "tf"] <- wa.pred$fit[,wa.type]
  }
}

}

# Bagged tree

if (run.bagging) {

  bagging.pred <- matrix(nrow=nrow(spe.new), ncol=bagging.n.trees)
  for (j in 1:bagging.n.trees) { # Run n iterations with bootstrapped
    # calibration data

```

```

cal.samples.iter <- sample (1:nrow(spe), nrow(spe), replace=TRUE)
spe.iter.bagging <- spe[cal.samples.iter,]
env.iter.bagging <- env[cal.samples.iter,env.param,drop=FALSE]
zeros <- colSums(spe.iter.bagging)==0
spe.iter.bagging <- spe.iter.bagging[,!zeros]
new.iter.bagging <- spe.new[,!zeros]
bagging.formula.iter <- as.formula(paste (env.param,
paste(colnames(spe.iter.bagging), collapse=" + "), sep=" ~ "))
bagging.model.iter <- rpart (bagging.formula.iter, data=cbind(env.iter.bagging,
spe.iter.bagging))
bagging.pred[,j] <- predict (bagging.model.iter, new.iter.bagging)
}
new.pred[, "bagging"] <- rowMeans (bagging.pred)      # Calculate mean
                                                       # predictions
}

# Random forest

if (run.raf) {

  raf.pred.model <- randomForest (x=spe, y=env[,env.param], ntree=raf.n.trees)
  raf.pred <- predict (raf.pred.model, spe.new)
  new.pred[, "raf"] <- raf.pred

}

# Rotation forest

if (run.rof) {

  rof.pred.model <- RotationForest(Df.X=spe, Df.Y=env, kuse=rof.k, juse=rof.n.trees)
  rof.pred <- predict.RotationForest (rof.pred.model, Df.X=spe.new)
  new.pred[, "rof"] <- rof.pred

}

# Boosted tree

if (run.brt) {

  brt.formula <- as.formula(paste (env.param, "~", paste(colnames(spe), collapse=" +
  ")))
  brt.pred.model <- gbm(brt.formula, distribution="gaussian", n.trees=brt.max.trees,
shrinkage=brt.lrate, interaction.depth=brt.int.depth, bag.fraction=brt.bag.fraction,
cv.folds=4, verbose=FALSE, data=cbind(env, spe), n.cores=brt.n.cores)
  brt.ntrees <- gbm.perf(brt.pred.model, method="cv", plot.it=FALSE)
  brt.pred <- predict(brt.pred.model, spe.new, brt.ntrees, type="response")
  new.pred[, "brt"] <- brt.pred

}

# K. Save reconstructions (via clipboard to Excel).

write.table (new.pred, file="clipboard", sep="\t")

```

```
#####
#                                     SUPPLEMENTARY DATA                                     #
#                                                                                             #
# Salonen JS, Verster AJ, Engels S, et al. (xxx) Calibrating aquatic                       #
# microfossil proxies with regression-tree ensembles: cross-validation                   #
# with modern chironomid and diatom data. The Holocene (submitted).                     #
#                                                                                             #
#                                     ---                                                         #
#                                                                                             #
# This file includes code for:                                                             #
# - creating a rotation forest model (function RotationForest)                         #
# - predicting with a rotation forest model (function predict.RotationForest)          #
#                                                                                             #
# If you use this code in published work, please cite the original paper.               #
#####

RotationForest <- function(Df.X,Df.Y,kuse = NULL,Muse = NULL,juse = 10,rows.use.frac
= 0.75,verbose = FALSE,...) {
  # Trains a rotation forest
  #
  # Args:
  # Df.X: a data frame of X independent variables
  # Df.Y: a data frame of Y response values
  # kuse: The number of groups of independent variables
  #       # Muse: The number of independent variables to use in each rotation
  #       # NOTE: Specify only ONE of kuse and Muse since they are directly related
to one another
  # juse: The number of trees to train in the ensemble
  # verbose: A logical for if you want to print the progress
  # ...: Objects to pass onto the rpart function
  #
  # Returns:
  # An object of class RotationForest

  RotForest <- list()
  class(RotForest) <- "RotationForest"
  #Check if Df.Y is a dataframe, if so, convert to vector
  if (!is.null(dim(Df.Y))) {
    if (ncol(Df.Y) != 1)
      stop("You should only have 1 column in Df.Y")
    Df.Y <- Df.Y[,1]
  }

  if (is.factor(Df.Y))
    RotForest$type <- "Classification"
  else
    RotForest$type <- "Regression"
  if ((RotForest$type == "Regression") && (length(unique(Df.Y)) <= 5)) {
    warning("The response has five or fewer unique values. Are you sure you
want to do regression?")
  }
  if ((RotForest$type == "Classification") && (length(unique(Df.Y)) < 2))
    stop("Need at least two classes to do classification.")
  if (is.null(kuse) && is.null(Muse)) {
    Muse <- 3
    print(sprintf("Didn't specify either kuse or Muse, setting to default of
Muse = %i",Muse))
  }
  if (sum(c(is.null(kuse),is.null(Muse))) != 1)
    stop("Must specify one and only one of kuse and Muse")

  #Test for NAs and convert to numeric
  if (typeof(Df.X) != "double")
    Df.X <- data.matrix(Df.X)
  if (sum(is.na(Df.X)))
    stop("You have NA in your data")
}
```

```

RotForest$X <- Df.X
RotForest$Y <- Df.Y

#The relationship between k and M
if (is.null(Muse))
  Muse <- ceiling(ncol(RotForest$X) / kuse)
if (is.null(kuse))
  kuse <- ceiling(ncol(RotForest$X) / Muse)

RotForest$k <- kuse
RotForest$M <- Muse
RotForest$j <- juse

fits <- list()
rots <- list()
for (i in 1:juse) {
  model.current <- BuildOneModel(RotForest, rows.use.frac = rows.use.frac, ...)

  fits[[i]] <- model.current[[1]]
  rots[[i]] <- model.current[[2]]
  if (verbose == TRUE) {
    print(sprintf("Currently completed %i out of %i models", i, j.use))
  }
}
RotForest$models <- fits
RotForest$rotations <- rots
return(RotForest)
}

predict.RotationForest <- function(RotForest, Df.X = NULL, prob = FALSE) {
  # predict() for an object of class RotationForest
  #
  # Args:
  # RotForest: An object of class RotationForest
  # Df.X: a data frame of X independent variables
  # prob: A logical indicating if you want the probabilities of being in each class,
  rather than the default predictions
  #
  # Returns:
  # A vector or predictions or a table of the probabilities of the different classes

  if (is.null(Df.X))
    Df.X <- RotForest$X

  #Creates the predictions and puts them into a list
  prediction.values <- list()
  for (i in 1:length(RotForest$models)) {
    model.current <- RotForest$models[[i]]
    data.current <- as.matrix(Df.X) %*% RotForest$rotations[[i]]
    data.current <- as.data.frame(data.current)
    colnames(data.current) <- paste0("X", 1:ncol(data.current))
    prediction.values[[i]] <- predict(model.current, data.current)
  }

  #Calculates the probability of each class by averaging across the different trees
  if (RotForest$type == "Classification") {
    results <- matrix(ncol = ncol(prediction.values[[1]]), nrow = nrow(Df.X))
    colnames(results) <- colnames(prediction.values[[1]])
    for (i in 1:nrow(Df.X)) {
      results[i,] <- apply(do.call(rbind, lapply(prediction.values, function(x)
x[i,])), 2, mean)
    }
    if (prob == TRUE)
      return(results)
    else
      return(apply(results, 1, function(x) names(which(x == max(x)))))
  }
}

```

```

    }
    else if (RotForest$type == "Regression") {
      return(apply(do.call(rbind,lapply(prediction.values,function(x)
x)),2,mean))
    }
    else
      stop("Type of RotForest object must be Classification or Regression")
  }
}

```

```

BuildOneModel <- function(RotForest,rows.use.frac = 0.75,...) {
  # Builds a single decision tree using the methodology from Rodriguez et al. 2006
  #
  # Args:
  # RotForest - object of class RotationForest
  # rows.use.frac: The fraction of data points to use in each model of the ensemble
  # ...: Objects to pass onto the rpart function
  #
  # Returns:
  # A list containing the rpart object and the rotation matrix R

  require('rpart')

  M <- RotForest$M
  k <- RotForest$k
  j <- RotForest$j
  Df.X <- RotForest$X
  Df.Y <- RotForest$Y

  R <- matrix(nrow = ncol(Df.X),ncol = ncol(Df.X), data = 0)
  R.order <- R

  Order <- data.frame(1:ncol(Df.X),sample(sort(rep(1:k, times = M))[1:ncol(Df.X)],
size = ncol(Df.X), replace = FALSE))
  colnames(Order) <- c("V1","V2")

  if (max(Order$V2) != k) {
    warning(sprintf("Impossible to create %i groups, using %i
instead",k,max(Order$V2)))
    k <- max(Order$V2)
    RotForest$k <- max(Order$V2)
  }

  for (i in 1:k) {
    #Replace must be TRUE for actual bootstrapping
    rows.use <- sample(1:nrow(Df.X),size = round(rows.use.frac * nrow(Df.X)),replace
= TRUE)
    cols.use <- subset(Order,V2 == i)$V1

    if (length(rows.use) != M) {
      rows.use.frac.new <- M / nrow(Df.X)
      rows.use <- sample(1:nrow(Df.X),size = M,replace = TRUE)
      warning(sprintf("If you only sample %f of the rows you have less rows than
columns, sampling %s instead", rows.use.frac, rows.use.frac.new))
    }

    #For keeping track of the position inside the rotation matrix R
    start <- (i - 1) * M + 1
    if (i != k) {
      end <- i * M
    }
    else {
      end <- ncol(Df.X)
    }
  }
}

```

```

Df.X.sub <- Df.X[rows.use,cols.use]
Df.X.sub.rotation <- prcomp(Df.X.sub)$rotation
R[start:end,start:end] <- Df.X.sub.rotation
#Now change the position of the columns to match that of Df.X
R.order[start:end,cols.use] <- R[start:end,start:end]
}

#rotate data onto the matrix
Df.X.rotate <- as.matrix(Df.X) %*% R.order
Df.rotate.full <- data.frame(Df.Y,Df.X.rotate)
colnames(Df.rotate.full)[1] <- "class"

fit <- rpart(class ~ ., data = Df.rotate.full,...)

return.list <- list()
return.list[[1]] <- fit
return.list[[2]] <- R.order

return(return.list)
}

```