Accelerated and Interpretable Oblique Random Survival Forests

Byron C. Jaeger, Sawyer Welden, Kristin Lenoir, Jaime L. Speiser, Matthew W. Segar, Ambarish Pandey, and Nicholas M. Pajewski

ABSTRACT
The oblique random survival forest (RSF) is an ensemble supervised learning method for right-censored outcomes. Trees in the oblique RSF are grown using linear combinations of predictors, whereas in the standard RSF, a single predictor is used. Oblique RSF ensembles have high prediction accuracy, but assessing many linear combinations of predictors induces high computational overhead. In addition, few methods have been developed for estimation of variable importance (VI) with oblique RSFs. We introduce a method to increase computational efficiency of the oblique RSF and a method to estimate VI with the oblique RSF. Our computational approach uses Newton-Raphson scoring in each non-leaf node, We estimate VI by negating each coefficient used for a given predictor in linear combinations, and then computing the reduction in out-of-bag accuracy. In benchmarking experiments, we find our implementation of the oblique RSF is hundreds of times faster, with equivalent prediction accuracy, compared to existing software for oblique RSFs. We find in simulation studies that "negation VI" discriminates between relevant and irrelevant numeric predictors more accurately than permutation VI, Shapley VI, and a technique to measure VI using analysis of variance. All oblique RSF methods in the current study are available in the aorsf R package, and additional supplemental materials are available online.

1. Introduction
Risk prediction may reduce the burden of disease by guiding strategies for prevention and treatment in a wide range of domains (Moons et al. 2012). The random survival forest (RSF; Hothorn, Hornik, and Zeileis 2006; Ishwaran and Kogalur 2010) is a supervised learning algorithm that has been used frequently for risk prediction (Wang and Li 2017). The RSF is a large set of de-correlated and randomized decision trees, with each tree contributing to the ensemble’s prediction function. Notable characteristics of the RSF include uniform convergence of its ensemble survival prediction function to the true survival function, first shown by Ishwaran and Kogalur (2010) and later by Cui et al. (2017) under more general conditions. However, Cui et al. (2017) noted that the RSF is at a disadvantage when predictors are correlated and some are not relevant to the censored outcome, which is a strong possibility when large clinical and “omic” databases are leveraged for risk prediction.

A potential approach to improve the RSF when predictors are correlated and some are not relevant to the censored outcome is to use oblique trees instead of axis based trees. Axis based trees split data using a single predictor, creating decision boundaries that are perpendicular or parallel to axes of the predictor space (see Breiman et al. 2017, chap. 2). Oblique trees split data using a linear combination of predictors, creating decision boundaries that are neither parallel nor perpendicular to axes of their contributing predictors (see Breiman et al. 2017, chap. 5). Oblique trees may create more adequate partitions of a predictor space compared to axis-based trees, as demonstrated in Figure 1. Menze et al. (2011) examined prediction accuracy of RFs in the presence of correlated predictors and found that oblique RFs had substantially higher prediction accuracy compared to axis-based RFs. Similarly, Jaeger et al. (2019) found that growing RSFs with oblique rather than axis-based trees reduced the RSF’s concordance error, with improvements ranging from 2.5% to 24.9% depending on the data analyzed.

Conceptually, oblique trees are similar to methods that use covariate rotation or extension to generate linear combinations of predictors prior to growing axis-based trees (Zhou et al. 2016; Wang and Zhou 2017). The difference is that oblique trees generate linear combinations of predictors within each non-leaf node, using only the data and predictors associated with that node, instead of generating the linear combinations prior to growing the tree. This “node-specific” approach to creating linear combinations of predictors leads to greater diversity in oblique tree ensembles, which may lead to greater prediction accuracy (Breiman 2001).
Figure 1. Decision boundaries from an axis-based (panel A) and oblique (panel B) decision tree used to classify penguin species based on bill depth and bill length. The decision boundary from the oblique tree is better able to capture the geometry of this data, leading to fewer misclassified penguins.

Despite the potential for higher accuracy, oblique trees have at least two notable drawbacks compared to axis-based trees. First, finding a locally optimal oblique decision rule may require exponentially more computation than an axis-based rule. If \( p \) predictors are potentially used to split \( n \) observations, up to \( O(n^p) \) oblique splits can be assessed versus \( O(n \cdot p) \) axis-based splits (Heath, Kasif, and Salzberg 1993; Murthy, Kasif, and Salzberg 1994). Second, although variable importance (VI) is one of the most widely used strategies to interpret RFs (Ishwaran and Lu 2019), few studies have investigated VI for oblique RFs (see Menze et al. 2011, sec. 5), and fewer have investigated VI specifically for the oblique RSF. Without general methodology to estimate VI, interpretation of oblique RFs is challenging.

The aim of this article is to introduce methodology that improves the computational efficiency and interpretation of oblique RSFs. Section 2 reviews prior work, introduces our method to reduce the computational cost of oblique RSFs (i.e., accelerate them), and introduces "negation VI", a method to estimate VI with oblique RSFs that does not require permutation of data. We describe benchmarking experiments and simulation studies to evaluate these methods in Section 3, and present results in Section 4. In Section 5, we summarize results from the current study, connecting our findings to prior work and outlining potential future research topics. All oblique RSF methods introduced in the current study are available in the aorsf R package (Jaeger et al. 2022).

2. Methods and Materials

Sections 2.1 and 2.2 briefly summarize prior studies that have developed methods related to the oblique RSF and VI, respectively. Section 2.3 describes our approach to reduce computational overhead of the oblique RSF and Section 2.4 introduces negation VI, a novel technique to estimate VI in oblique RFs.

2.1. Axis-based and Oblique Random Forests

After Breiman (2001) introduced the axis-based and oblique RF, numerous methods were developed to grow oblique RFs for classification or regression tasks (Menze et al. 2011; Zhang and Suganthan 2014; Rainforth and Wood 2015; Zhu, Zeng, and Kosorok 2015; Poona, Van Niekerk, and Ismail 2016; Qiu et al. 2017; Tomita et al. 2020; Katuwala et al. 2020). However, oblique splitting approaches for classification or regression may not generalize to censored outcomes (e.g., see Zhu 2013, sec. 4.5.1), and most research involving the RSF has focused on forests with axis-based trees (Wang and Li 2017).

Hothorn, Hornik, and Zeileis (2006) developed an axis-based RSF in their framework for unbiased recursive partitioning, more commonly referred to as the conditional inference forest (CIF). Zhou et al. (2016) developed a rotation forest based on the CIF and Wang and Zhou (2017) develop a method for extending the predictor space of the CIF. Ishwaran et al. (2008) developed an axis-based RF with strict adherence to the rules for growing trees proposed in Breiman (2001). Jaeger et al. (2019) developed the oblique RSF following the bootstrapping approach described in Breiman’s original RF and incorporating early stopping rules from the CIF.

Fast algorithms to fit axis-based RSFs are available in the randomForestSRC R package (Ishwaran and Kogalur 2019) and the ranger (Wright and Ziegler 2017) R package. randomForestSRC provides a unified interface to grow RFs in a wide range of analyses, and ranger is designed to grow RFs efficiently using high dimensional data. Fast algorithms to fit the CIF are provided by the party R package (Hothorn et al. 2019), which provides a computational toolbox for recursive partitioning using conditional inference trees. Jaeger et al. (2019) developed the obliqueRF package and found it was approximately 30 times slower than party and nearly 200 times slower than randomForestSRC. Few studies have developed software with fast algorithms for oblique RSFs that have comparable speed compared to algorithms for axis-based RSFs.
2.2. Variable Importance

Several techniques to estimate VI have been developed since Breiman (2001) introduced permutation VI, which is defined for each predictor as the difference in a RFs estimated prediction error before versus after the predictor’s values are randomly permuted. Strobl et al. (2007) identified bias in permutation VI driven by categorical predictors and bootstrap sampling, and proposed a permutation VI measure based on unbiased recursive partitioning (Hothorn, Hornik, and Zeileis 2006). Menze et al. (2011) introduced an approach to estimate VI for oblique RFs that computes an analysis of variance (ANOVA) table in non-leaf nodes to obtain p-values for each predictor contributing to the node. The ANOVA VI\(^1\) is then defined for each predictor as the proportion of times a p-value associated with the predictor is \(≤ 0.01\) while growing a forest. Lundberg and Lee (2017) introduced a method to estimate VI using SHapley Additive exPlanation (SHAP) values, which estimates the contribution of a predictor to a model’s prediction for a given observation. SHAP VI is computed for each predictor by taking the mean absolute value of SHAP values for that predictor across all observations in a given set. With the exception of Menze et al. (2011), few studies have evaluated estimation of VI using oblique RFs, and fewer have examined VI specifically for the oblique RSF.

2.3. The Accelerated Oblique Random Survival Forest

Consider the usual framework for right-censored time-to-event outcomes with training data

\[ D_{\text{train}} = \{ (T_i, \delta_i, x_i) \}_{i=1}^{N_{\text{train}}}. \]

Here, \( T_i \) is the event time if \( \delta_i = 1 \) or the censoring time if \( \delta_i = 0 \), and \( x_i \) is a vector of predictors values. Assuming there are no ties, let \( t_1 < \cdots < t_m \) denote the \( m \) unique event times in \( D_{\text{train}} \).

To accelerate the oblique RSF, we propose to identify linear combinations of predictor variables in non-leaf nodes by applying Newton Raphson scoring to the partial likelihood function of the Cox regression model:

\[
L(\beta) = \prod_{i=1}^{m} \exp\left( \sum_{j \in R_i} x_j^T \beta \right)^{-\delta_i},
\]

where \( R_i \) is the set of indices, \( j \), with \( T_j \geq t_i \) (i.e., those still at risk at time \( t_i \)) and \( j(i) \) is the index of the observation for which an event occurred at time \( t_i \). Newton Raphson scoring is an exceptionally fast estimation procedure, and the \texttt{survival} package (Therneau 2022b) includes documentation that outlines how to efficiently program it (Therneau 2022a). As described in Therneau and Grambsch (2000), a vector of estimated regression coefficients, \( \hat{\beta} \), is updated in each step of the procedure:

\[
\hat{\beta}^{k+1} = \hat{\beta}^k + U(\hat{\beta} = \hat{\beta}^k) I^{-1}(\hat{\beta} = \hat{\beta}^k),
\]

where \( U(\hat{\beta}) \) is the score vector and \( I^{-1}(\hat{\beta}) \) is the inverse of the observed information matrix. After obtaining \( \hat{\beta} \), a linear combination of variables, \( \eta_i \), is obtained by computing \( \eta = x^T \hat{\beta} \).

For statistical inference, it is recommended to continue updating \( \hat{\beta} \) by completing additional iterations of Newton Raphson scoring until a convergence threshold is met. However, since an estimate of \( \hat{\beta} \) is created by the first iteration of Newton Raphson scoring, only one iteration of Newton Raphson scoring is needed to identify a valid linear combination of predictors. In Section 4.1, we formally test whether growing oblique survival trees using one iteration of Newton Raphson scoring provides equivalent prediction accuracy compared to trees where iterations are completed until a convergence threshold is met.

Algorithm 1 presents our approach to fitting an oblique survival tree in the accelerated oblique RSF using default values from the \texttt{aorsf} \( \texttt{R} \) package. Several steps are taken to reduce computational overhead. First, memory is conserved by conducting bootstrap resampling via random integer-valued weights, rather than using a bootstrapped copy of the original data. Memory conservation also takes place in terminal nodes, where we restrict estimation of the survival and cumulative hazard function to event times that occur among observations in the node. Second, early stopping is applied to the tree-growing procedure if a statistical criterion is not met. In our case, the criterion is based on the magnitude of a log-rank test statistic corresponding to splitting the data at a current node. Third, instead of greedy recursive partitioning, we use “good enough” partitioning. More specifically, instead of computing a log-rank test statistic for several different linear combinations of variables and proceeding with the highest scoring option, we identify an optimal cut-point for one linear combination of variables and assess whether using this combination will create a split that passes the criterion for splitting a node. If it does not pass the criterion, then another linear combination will be tested, with the maximum number of attempts set by the parameter \texttt{n_retry}. Often a “good-enough” split can be found in just one attempt when the training set is large, which gives the accelerated oblique RSF a computational advantage in larger training sets compared to greedy partitioning.

2.4. Negation Variable Importance

This Section introduces negation VI, which is similar to permutation VI in that it measures how much a model’s prediction error increases when a variable’s role in the model is destabilized. Specifically, negation VI measures the increase in an oblique RF’s prediction error after flipping the sign of all coefficients linked to a variable (i.e., negating them). As negating a coefficient effectively flips decision boundaries around the corresponding predictor’s axis, scaling numeric predictors to have a mean of zero and standard deviation of one is recommended.\(^2\) For the current study, we use Harrell’s concordance (C)-statistic (Harrell et al. 1982) to measure change in prediction error when computing negation VI.

To demonstrate negation VI, consider a classification task where the goal is prediction of penguin species (chinstrap, gentoo, or adelie) based on bill length and bill depth (Horst, Hill, and Gorman 2020). Scaling these predictors to be centered at 0, we

\(^1\)Menze et al. (2011) name their method “oblique RF VI,” but we use the name ‘ANOVA VI’ in this article to avoid confusing Menze’s approach with other approaches to estimate VI for oblique RFs.

\(^2\)The \texttt{aorsf} package automatically scales numeric inputs to a mean of zero and standard deviation of one.
Algorithm 1 Accelerated oblique random survival tree using default parameters.

Require: Training data $D_{\text{train}} = \{(T_i, d_i, x_i)\}_{i=1}^{N_{\text{train}}}$, mtry = $\sqrt{\text{ncol}(x_{\text{train}})}$, n_split = 5, n_retry = 3, and split_min_stat = 3.841459

1: $T \leftarrow \emptyset$
2: $w \leftarrow \text{sample(from = \{0, \ldots, 10\}, n = \text{nrow}(x_{\text{train}}))}$, replace = T)
3: $D_{\text{in-bag}} \leftarrow \text{subset}(D_{\text{train}}, \text{rows} = \text{which}(w > 0))$
4: $w \leftarrow \text{subset}(w, \text{which}(w > 0))$
5: $\text{node_assignments} \leftarrow \text{rep}(1, \text{times} = \text{nrow}(x_{\text{in-bag}}))$
6: $\text{nodes_to_split} \leftarrow \{1\}$
7: while $\text{nodes_to_split} \neq \emptyset$ do
8:   for node $\in \text{nodes_to_split}$
9:       $n_{\text{try}} \leftarrow 1$
10:      $\text{node_rows} \leftarrow \text{which}(\text{node_assignments} \equiv \text{node})$
11:      $\text{node_cols} \leftarrow \text{sample(from = \{1, \ldots, \text{ncol}(x)\}, \text{size} = \text{mtry}, \text{replace} = \text{F})}$
12:      $D_{\text{node}} \leftarrow \text{subset}(D_{\text{in-bag}}, \text{rows} = \text{node_rows}, \text{columns} = \text{node_cols})$
13:      $\beta \leftarrow \text{newt_raph}(D_{\text{node}}, \text{weights} = \text{subset}(w, \text{node_rows}), \text{max_iter} = 1)$
14:      $\eta \leftarrow x_{\text{node}}^T \beta$
15:      $C \leftarrow \text{sample(from = \text{unique}(\eta), \text{size} = \text{n_split}, \text{replace} = \text{F})}$
16:      $c \leftarrow \text{argmax}_{c \in C} \left\{ \log\text{rank_stat}(\eta, c^{*}) \right\}$
17:      if log_rank_stat($\eta, c$) $\geq$ split_min_stat then
18:         $\hat{T} \leftarrow \text{add_node}(T, \text{name} = \text{node}, \beta = \beta, \text{cutpoint} = c)$
19:         COMMENT Right node logic omitted for brevity (identical to left node logic)
20:         $\text{node_left_name} \leftarrow \text{max(\text{node_assignments})} + 1$
21:         $\text{node_left_rows} \leftarrow \text{subset}(\text{node_rows}, \text{which}(\eta \leq c))$
22:         $\text{subset}(\text{node_assignments}, \text{node_left_rows}) \leftarrow \text{node_left_name}$
23:         if is_splittable(subset(\text{node_assignments, node_left_rows})) then
24:             nodes_to_split $\leftarrow$ nodes_to_split $\cup$ node_left_name
25:     else
26:         $\hat{T} \leftarrow \text{add_leaf}(\hat{T}, \text{data} = \text{subset}(D_{\text{node}}, \text{rows} = \text{node_left_rows}))$
27:     end if
28:     else if $n_{\text{try}} \leq n_{\text{retry}}$ then
29:         $n_{\text{try}} \leftarrow n_{\text{try}} + 1$
30:     go to 11
31:     else
32:         $\hat{T} \leftarrow \text{add_leaf}(\hat{T}, \text{data} = D_{\text{node}})$
33:     end if
34:     if nodes_to_split $\not= \emptyset$ then
35:         nodes_to_split $\leftarrow$ nodes_to_split $\setminus$ \{node\}
36:     end for
37: end while
38: return $\hat{T}$

find oblique decision boundaries defined by linear combinations of bill length and bill depth correctly classify most of the data (Figure 2, Panel A). Permuting the values of bill length leads to several misclassified observations, suggesting that bill length is an important predictor (Figure 2, Panel B). However, inspecting the permuted data shows that a number of observations moved to a region in the predictor space where there were previously no observations. Moving data to unobserved or perhaps unobservable regions of the predictor space may cause extrapolation error, which Hooker, Mentch, and Zhou (2021) identified as a cause of bias in permutation importance. Negating the coefficients for bill length in the linear combinations that define our decision boundaries causes the boundaries to flip across bill length’s axis (Figure 2, Panel C). This leads to several misclassified observations, suggesting that bill length is an important predictor without distorting the joint distribution of bill length and bill depth.

Negation VI is an extension of “anti VI”, a VI technique for axis-based trees which became the default VI method for randomForestSRC in version 2.14.0. Anti VI reverses the direction of all decision nodes that use a specific variable, and then reassesses the ensemble prediction error. So, if an axis-based decision rule were defined as $x > 5 \Rightarrow$ send data to right node, the decision rule would become $x < 5 \Rightarrow$ send data to left node when computing the importance of a predictor $x$.

Negation VI has several strengths. First, it generalizes to any oblique RF (i.e., not just RSFs) using any valid error function, making it both general and flexible. Second, negation is nonrandom, making it reproducible without setting a random seed and making it slightly faster than permutation importance.

3The acrsf package enables customized functions to be applied in lieu of the default C-statistic.
Fourth, since negation VI does not permute variables, the analyst need not worry about impossible combinations of predictors that may occur when one predictor is randomly permuted, such as having a negative status for type 2 diabetes and having Hemoglobin A1c level $\geq 6.5\%$ (a value indicative of type 2 diabetes) as a result of randomly permuting the values of Hemoglobin A1c. However, in scenarios where decision boundaries have symmetry around the origin of the predictor space (e.g., all positive cases lie in a circle centered at the origin, with negative cases sitting outside the circle), negation importance will be less effective than permutation.

3. Numeric Experiments

Sections 3.1 and 3.2 present the design of numerical experiments examining the accelerated oblique RSF and negation VI, respectively. Section 3.3 summarizes our approach to evaluating computational efficiency of learning algorithms, with a focus on the accelerated oblique RSF and other RSF implementations. Section 3.4 provides details on computation and code.

3.1. Benchmark of Prediction Accuracy

The aim of this numeric experiment is to evaluate the prediction accuracy of the accelerated oblique RSF compared to its predecessor (the oblique RSF from the obliqueRSF R package) and to several other machine learning algorithms. Inferences drawn from this experiment include equivalence and inferiority tests based on Bayesian linear mixed models.

3.1.1. Learners

We consider four classes of learners: RSFs (both axis-based and oblique), boosting ensembles, regression models, and neural networks. Specific learners from each class are summarized in Table 1. To facilitate fair comparisons, tuning parameters were harmonized within each class. For example, for RSF learners, we set the minimum node size (a parameter shared by all RSF learners) as 10. Additionally, for RSF learners, the number of randomly selected predictors was the square root of the total number of predictors rounded to the nearest integer, and the number of trees in the ensemble was 500 (a common default value for the number of trees). For boosting, regression, and neural network learners, nested cross-validation was applied to tune relevant model parameters. Nested cross-validation includes an “inner” cross-validation loop to evaluate different specifications of tuning parameters and an “outer” loop that evaluates the tuned model, providing an unbiased estimate of the underlying model and its tuning procedure. Tuning for boosting models included identifying the number of steps to complete. The maximum number of steps was 5000, the learning rate was fixed at 0.01, and early stopping was applied if there was no improvement in cross-validated negative log-likelihood for 25 steps. For regression models, tuning was used to identify the magnitude of penalization. For neural networks, the number of layers (i.e., length) and number of nodes in the layers (i.e., width) was tuned, while dropout rate was fixed at 10%, batch size was fixed at 32 observations, and the rectified linear unit activation function was applied. In addition, neural networks completed a maximum of 500 epochs, with possible early stopping based on prediction accuracy in a validation set comprising 25% of its training data.

3.1.2. Evaluation of Prediction Accuracy

Our primary metric for evaluating the accuracy of predicted risk is the integrated and scaled Brier score (Graf et al. 1999), a proper scoring rule that combines discrimination and calibration in one value and improves interpretability by adjusting for a benchmark model (Kattan and Gerds 2018). Consider a testing dataset:

$$D_{\text{test}} = \{(T_i, \delta_i, x_i)\}_{i=1}^{N_{\text{test}}}.$$ 

Let $\hat{S}(t \mid x_i)$ be the predicted probability of survival up to a given prediction time of $t > 0$. For observation $i$ in $D_{\text{test}}$, let $\hat{S}(t \mid x_i)$ be the predicted probability of survival up to a given prediction time of $t > 0$. Define

$$\hat{B}(t) = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \left( \hat{S}(t \mid x_i)^2 \cdot I(T_i \leq t, \delta_i = 1) \cdot \hat{G}(T_i)^{-1} \right)$$

$$+ \left[ 1 - \hat{S}(t \mid x_i) \right]^2 \cdot I(T_i > t) \cdot \hat{G}(t)^{-1}$$
where $\hat{G}(t)$ is the Kaplan-Meier estimate of the censoring distribution. As $\hat{BS}(t)$ is time dependent, integration over time provides a summary measure of performance over a range of plausible prediction times. The integrated $\hat{BS}(t)$ is defined as

$$\hat{BS}(t_1,t_2) = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} \hat{BS}(t) dt.$$  (2)

In our results, $t_1$ and $t_2$ are the 25th and 75th percentile of event times, respectively. $\hat{BS}(t_1,t_2)$, a sum of squared prediction errors, can be scaled to produce a measure of explained residual variation (i.e., an $R^2$ statistic) by computing

$$R^2 = 1 - \frac{\hat{BS}(t_1,t_2)}{\hat{BS}_0(t_1,t_2)}$$  (3)

where $\hat{BS}_0(t_1,t_2)$ is the integrated Brier score when a Kaplan-Meier estimate for survival based on the training data is used as the survival prediction function $\hat{S}(t)$. We refer to this $R^2$ statistic as the index of prediction accuracy (IPA) (Kattan and Gerds 2018).

Our secondary metric for evaluating predicted risk is the time-dependent concordance (C)-statistic. We compute the first time-dependent C-statistic proposed by Blanche, Dartigues, and Jaccmin-Gadda (2013, eq. 3), which is interpreted as the probability that a risk prediction model will assign higher risk to a case (i.e., an observation with $T \leq t$ and $\delta = 1$) versus a non-case (i.e., an observation with $T > t$). Similar to the IPA, observations with $T \leq t$ and $\delta = 0$ only contribute to inverse probability of censoring weights for the time-dependent C-statistic.

Both the IPA and time-dependent C-statistic generally take values between 0 and 1. To avoid presenting an excessive amount of leading zeroes in our tables, figures, and text, we scale both the IPA and time-dependent C-statistic by 100. For example, we present a value of 25 if the IPA is 0.25, 87 if the time-dependent C-statistic is 0.87, and present 10.2 if the difference between two IPA values is 0.102

### 3.1.3. Datasets

We used a collection of 21 datasets containing a total of 35 risk prediction tasks (tasks per dataset ranged from one to four). Participant-level data from the GUIDE-IT and SPRINT clinical trials and the ARIC, MESA, and JHS community cohort studies was obtained from the National Institute of Health Biologic Specimen and Data Repository Coordinating Center (BioLINCC). Designs and protocols for these studies have been made available (ARIC Investigators 1989; Bild et al. 2002; Taylor Jr et al. 2005; SPRINT Research Group 2015; Felker et al. 2017). All other datasets were publicly available and obtained through R packages (see Appendix A.1 in Supplementary Materials). Across all prediction tasks, the number of observations ranged from 137 to 17,549 (median: 1,384), the number of predictors ranged from 7 to 1,692 (median: 41), and the percentage of censored observations ranged from 5.26 to 97.7 (median: 78.1) (Table A1 in supplementary materials).

### 3.1.4. Monte Carlo Cross-validation

For each risk prediction task, we completed 25 runs of Monte Carlo cross-validation. In each run, we used a random sample containing 50% of the available data for training and the remaining 50% for testing each of the learners described in Section 3.1.1. Then, for each learner, we computed the IPA and time-dependent C-statistic. If any learner failed to obtain predictions on any particular split of data, the results for that split were omitted from downstream analyses for all learners.

---

4For example, when the prediction task was to predict risk of death in the ACTG 320 clinical trial (26 events total), some splits did not leave enough events in the training data to fit complex learners such as neural networks.
3.1.5. Statistical Analysis
After collecting data from 25 replications of Monte Carlo cross validation for the 14 learners in all 35 risk prediction tasks, we analyzed the resulting 12,250 observations of IPA and, separately, time-dependent C-statistic, using a Bayesian linear mixed model. Our approach follows the ideas described by Benavoli et al. (2017) and Kuhn and Wickham (2020), who developed guidelines on making statistical comparisons between learners using Bayesian models. Specifically, we fit two models:

\[
IPA = \gamma_0 + \gamma \cdot \text{learner} + (1 | \text{data/run})
\]

and

\[
C\text{-stat} = \gamma_0 + \gamma \cdot \text{learner} + (1 | \text{data/run}).
\]

Random intercepts for specific splits of data (i.e., run in the model formula) were nested within datasets. The intercept, \(\gamma_0\), was the expected value of the outcome using \(\text{aorsf-fast}\), making the coefficients in \(\gamma\) the expected differences between \(\text{aorsf-fast}\) and other learners. Default priors from \texttt{rstanarm} were applied for model fitting (Goodrich et al. 2022).

Hypothesis testing. For both the IPA and time-dependent C-statistic, we conducted equivalence and inferiority tests based on a 1 point region of practical equivalence. More specifically, we concluded that two learners had practically equivalent IPA or time-dependent C-statistic if there was a 95% or higher posterior probability that the absolute difference in the relevant metric was less than 1. We concluded that one learner was weakly superior when there was \(\geq 95\%\) posterior probability that the absolute difference in the relevant metric was nonzero, and concluded superiority when there was \(\geq 95\%\) posterior probability that the absolute difference in the relevant metric was 1 or more.

3.2. Benchmark of Variable Importance
The aim of this experiment is to evaluate negation VI and similar VI methods based on how well they can discriminate between relevant and irrelevant variables, where relevance is defined by having a relationship with the simulated outcome. We consider methods that are intrinsic to the oblique RF (e.g., ANOVA VI), those that are intrinsic to the RF (e.g., permutation VI), and those that are model-agnostic (e.g., SHAP VI). VI methods with unavailable or still developing software were not included.\(^5\)

3.2.1. Variable Importance Techniques
We compute permutation VI for axis based RSFs using the \texttt{randomForestSRC} package. We compute ANOVA VI, negation VI, and permutation VI for oblique RSFs using the \texttt{aorsf} package. For ANOVA VI, we applied a \(p\)-value threshold of 0.01, following the threshold recommended by Menze et al. (2011). We compute SHAP VI for boosted tree models using the \texttt{xgboost} package (Chen et al. 2022), which incorporates the tree SHAP approach proposed by Lundberg, Erion, and Lee (2018).

\(^5\)Although the \texttt{party} package implements the approach to VI developed by Strobl et al. (2007), the developers of the \texttt{party} package note that the implementation of this approach for survival outcomes is “extremely slow and experimental” as of version 1.3.10. Therefore, it is not incorporated in the current simulation study.

3.2.2. Variable Types
We considered five classes of predictor variables, with each class characterized by its variables’ relationship to a right-censored outcome on the log-hazard scale. Specifically,

- irrelevant variables had no relationship with the outcome.
- main effect variables had a linear relationship to the outcome on the log-hazard scale.
- nonlinear effect variables had a nonlinear relationship to the outcome. A normally distributed variable \(x\) was generated with a linear relationship to the outcome on the log-hazard scale, then \(\tilde{x} = \sin(a \cdot \pi \cdot x)\) was retained for modeling. The constant \(a\) varied uniformly from 0.125 to 0.25.
- combination effect variables were formed by linear combinations of three other variables. While their combination was linearly related to the outcome on the log-hazard scale, each of the three variables contributing to the combination had no relation to the outcome.
- interaction effect variables were related to the outcome by multiplicative interaction with one other variable, which could have been a main effect, nonlinear effect, or combination effect variable.

3.2.3. Simulated Data
We initiated each set of simulated data with a random draw of size \(n\) from a \(p\)-dimensional multivariate normal distribution, yielding \(n\) observations of \(p\) predictors. Each of \(p\) predictor variables had a mean of zero, standard deviation of 1, and correlation with other predictor variables drawn at random between a lower and upper boundary. A time-to-event outcome with roughly 45% of observations censored was generated using the \texttt{simsurv} package (Brilleman 2018; Brillemann et al. 2020) from a Weibull distribution with scale parameter (i.e., \texttt{lambda}) equal to 0.1 and shape parameter (i.e., \texttt{gammas}) equal to 1.5. The full predictor matrix (i.e., including interactions, nonlinear mappings, and combinations) was used to generate the outcome. Interactions, nonlinear mappings, and combinations were dropped from the predictor matrix after the outcome was generated so that VI techniques could be evaluated based on their ability to detect these effects.

3.2.4. Parameter Specifications
Parameters that varied in the current simulation study included the number of observations (500, 1000, and 2500) and the absolute value of the maximum correlation between predictors (0.3, 0.15, and 0). Parameters that remain fixed throughout the study included the number of predictors in each class (15) and the effect size of each predictor (one standard deviation increase associated with a 64% increase in relative risk). Using this design for simulated data, the Heller explained relative risk (95% confidence interval) of our covariates was 88.5 (88.2, 88.7) (Heller 2012) with 2500 observations.

3.2.5. Evaluation of Variable Importance
We compared VI techniques based on their discrimination (i.e., C-statistic) between relevant and irrelevant variables. Specifically, we generated a binary outcome for each predictor variable
based on its relevance (i.e., the binary outcome is 1 if the variable is relevant, 0 otherwise). Treating VI as if it were a “prediction” for these binary outcomes yields a C-statistic which may be interpreted as the probability that the VI technique will rank a relevant variable higher than an irrelevant variable (Harrell et al. 1982).

### 3.3. Benchmark of Computational Efficiency

The aim of this numeric experiment is to evaluate the computational efficiency of the accelerated oblique RSF compared to its predecessor (the oblique RSF from the oblique RSF R package) and to several other machine learning algorithms.

#### 3.3.1. Evaluation of Computational Efficiency

For each learner discussed in Section 3.1.1 and for each of the 35 risk prediction tasks analyzed in Section 3.1, we tracked the amount of time required to fit a prediction model (including time used to tune parameters) and compute predicted risk.

**Figure 3.** Index of prediction accuracy in multiple risk prediction tasks. Text appears in tasks where the accelerated oblique random survival forest obtained the highest score, showing absolute and relative improvement over the second best learner. Since this figure is intended to compare aorsf-fast to learners that are not oblique random survival forests, aorsf-cph, aorsf-random, and obliqueRSF-net are not included.
We performed additional benchmarks on the time required to fit 500 trees using aorsf, randomForestSRC, and ranger. The learners that represented these R packages were aorsf-fast, rsf-standard, and rsf-extratrees, respectively. To allow for controlled comparisons of computational efficiency with varying dimensions of training data, we used the same process to simulate data as described in Section 3.2.3, varying the number of observations from 100 to 10,000 and the number of predictors from 10 to 1000. The minimum node size of trees in this experiment was dynamically set as the nearest integer to the number of observations in the training set divided by 10.

### 3.4. Computational Details

All analyses were conducted using R version 4.1.3 with version 0.0.4 of the aorsf (Jaeger et al. 2022) package. Analyses were coordinated with assistance from the targets package (Landau 2021). To standardize comparisons of computational efficiency, we used the same process to simulate data as described in Section 3.2.3, varying the number of observations from 100 to 10,000 and the number of predictors from 10 to 1000. The minimum node size of trees in this experiment was dynamically set as the nearest integer to the number of observations in the training set divided by 10.

**Figure 4.** Expected differences in index of prediction accuracy between the accelerated oblique random survival forest and other learning algorithms. A region of practical equivalence is shown by purple dotted lines, and a boundary of nonzero difference is shown by an orange dotted line at the origin.
efficiency, all learners and VI techniques used up to 4 processing units.

4. Results

In Sections 4.1, 4.2, and 4.3, we present results from the benchmark of prediction accuracy, the simulation study of VI, and the benchmark of computational efficiency, respectively.

4.1. Prediction Accuracy

A full summary of all results presented in this section is provided in Table A.2 in supplementary materials. In total, 871 out of 875 Monte Carlo cross-validation runs were completed. On run 13, 18, 24, and 25 for the ACTG 320 data, the nn-cox learner encountered an error during its fitting procedure.

Index of prediction accuracy. Compared to learners that were not oblique RSFs, aorsf-fast had the highest IPA in 17 out of 35 risk prediction tasks, with an overall mean IPA of 12.6 (Figure 3). Compared to the learner with the second highest mean IPA (cif-standard), aorsf-fast's mean was 1.33 points higher, a relative increase of 11.7%. The posterior probability of aorsf-fast and aorsf-cph having practically equivalent expected IPA was 0.99, and the posterior probability

![Figure 5. Time-dependent concordance statistic for the accelerated oblique random survival forest and other learning algorithms across multiple risk prediction tasks. Text appears in tasks where the accelerated oblique random survival forest obtained the highest concordance, showing the absolute and percent improvement over the second best learner. Since this figure is intended to compare aorsf-fast to learners that are not oblique random survival forests, aorsf-cph, aorsf-random, and obliqueRSF-net are not included.](image-url)
**Figure 6.** Expected differences in time-dependent concordance statistic between the accelerated oblique random survival forest and other learning algorithms. A region of practical equivalence is shown by purple dotted lines, and a boundary of nonzero difference is shown by an orange dotted line at the origin.

| Learner            | Time-dependent C-statistic | Equivalence | Difference < 0 | Difference < -1 |
|--------------------|----------------------------|-------------|----------------|-----------------|
| aorsf-fast         |                            |             |                |                 |
| obliqueRSF-net     |                            | >.999       | 0.39           | 0.00            |
| aorsf-cph          |                            | >.999       | 0.60           | 0.00            |
| cif-standard       |                            | 0.85        | 0.99           | 0.15            |
| xgboost-aft        |                            | 0.55        | >.999          | 0.45            |
| ranger-extratrees  |                            | 0.46        | >.999          | 0.54            |
| cif-extension      |                            | 0.40        | >.999          | 0.60            |
| glmnet-cox         |                            | 0.06        | >.999          | 0.94            |
| rsf-standard       |                            | 0.02        | >.999          | 0.98            |
| cif-rotate         |                            | 0.01        | >.999          | 0.99            |
| xgboost-cox        |                            | 0.00        | >.999          | >.999           |
| aorsf-random       |                            | 0.00        | >.999          | >.999           |

The three techniques that used “aorsf” to estimate VI were ranked first (aorsf-negate; $C = 75.9$), second (aorsf-anova; $C = 73.9$), and third (aorsf-permute; $C = 73.2$) in overall mean C-statistic across all of the simulation scenarios,
with aorsf-negate obtaining the highest C-statistic in 26 out of 36 VI tasks (Figure 7). Among the four relevant variable classes, aorsf-negate had the highest mean C-statistic for main effects, combination effects, and nonlinear effects, with the greatest advantage of using aorsf-negate occurring among nonlinear and combination variables. Full results from the experiment are provided in Table A.3 in supplementary materials. Computationally, ANOVA VI was faster than negation and permutation VI, with a median time of 2.88 sec versus 20.4 and 21.8 sec, respectively.

4.3. Computational Efficiency
In the analysis of 35 risk prediction tasks, aorsf-fast was the second fastest learner overall, with a median time to develop a risk prediction model and compute predictions...
about 337 milliseconds longer than glmnet-cox (Figure 8). Comparing median computing times, aorsf-fast was 1291.6 times faster than its predecessor, obliqueRSF-net. In addition, aorsf-fast was 19.8, 1.90, and 3.18 faster than axis based forests grown using the party, ranger, and randomForestSRC packages, respectively.

In the analysis of time to fit 500 trees using simulated data, the ranger package exhibited the fastest computation times overall (Figure 9). aorsf was the second fastest when the number of predictors was 10 or 100, and randomForestSRC had similar computation time versus aorsf when 1000 predictors were present.

Figure 8. Distribution of time taken to fit a prediction model and compute predicted risk. The median time, in seconds, is printed and annotated for each learner by a vertical line.
5. Discussion

In this article, we have developed two contributions to the oblique RSF: (a) the accelerated oblique RSF (i.e., aorsf-fast) and (b) negation VI. Our technique to accelerate the oblique RSF reduces the number of operations required to find linear combinations of inputs using a single iteration of Newton Raphson scoring, while our VI technique directly engages with coefficients in linear combinations of inputs to measure importance of individual variables. In numeric experiments, we found that aorsf-fast is approximately 1291.6 times faster than its predecessor, obliqueRSF-net, with a practically equivalent C-statistic. We also found that estimating VI using negation instead of ANOVA increased the C-statistic for ranking a relevant variable higher than an irrelevant variable by 2.05, a relative increase of 2.78%.

To understand potential differences in computational efficiency, we reviewed code in the aorsf, randomForestSRC, and ranger packages. We found differences in how survival outcome data are saved in leaf nodes. For each leaf node, aorsf stores data with one row per unique event time using training data that are stored in the leaf, whereas randomForestSRC and ranger store survival outcomes at a fixed grid of event times in each leaf. By default, ranger creates a grid that includes all event times in the training data. While aorsf and randomForestSRC store data more efficiently, the grid strategy can cause higher computing time and memory usage when the grid of event times is large and a large number of leaf nodes are included in each tree, which can occur when minimum node size is small relative to the size of the training data. We kept minimum node size fixed in our benchmark of computational efficiency using real data, and dynamically increased minimum node size based on the size of the training set when we benchmarked computational efficiency using simulated data. Because of this decision, the randomForestSRC and ranger packages ran slower than aorsf in our benchmark of real data but not in the benchmark of simulated data.

5.1. Implications of Our Results

Accurate risk prediction models have the potential to improve healthcare by directing timely interventions to patients who are most likely to benefit. However, prediction models that cannot scale adequately to large databases or cannot be interpreted and explained will struggle to gain acceptance in clinical practice (Moss et al. 2022). The current study advances the oblique RSF, an accurate risk prediction model, toward being accurate, scalable, and interpretable. The improved computational efficiency of the accelerated oblique RSF increases the feasibility of applying oblique RSFs in a wide range of prediction tasks. Faster model evaluation and re-fitting also improve diagnosis and resolution of model-based issues (e.g., model calibration deteriorates over time). The introduction of negation VI also advances interpretability. VI is intrinsically linked to model fairness, as it can be used to identify when protected characteristics such as race, religion, and sexuality are inadvertently used (either directly or through correlates of these characteristics) by a prediction model. Since negation VI engages with the coefficients used in linear combinations of variables, a major component of oblique RSFs, it may be more capable of diagnosing unfairness in oblique RSFs compared to permutation importance and model-agnostic VI techniques.

5.2. Limitations and Next Steps

While the current study advances the oblique RSF toward being scalable and interpretable, there remain several limitations that can be targeted in future studies. The validity of the Cox partial likelihood function depends on an assumption of non-informative censoring, therefore the oblique RSF could be improved by incorporating methods to account for informative censoring prior to identifying a linear combination of variables (He et al. 2016; Cui et al. 2017). The accelerated oblique RSF...
does not account for competing risks, and biased estimation of incidence may occur when competing risks are ignored. Thus, allowing the oblique RSF to account for competing risks is a high priority for future studies. In addition, the current study only considered data without missing values, only evaluated oblique RSFs that applied the log-rank statistic for node splitting, and only considered negation VI estimates based on Harrell's C-statistic. Few studies have developed strategies to deal with missing data while growing oblique survival trees. Prior studies have found that log-rank tests can be misinformative when survival curves cross (Li et al. 2015), and that Harrell's C-statistic is dependent on the censoring distribution of the outcome (Uno et al. 2011). Thus, a second item is to expand the range of options available to users of the aorsf package, enabling them to apply strategies for imputation of missing values and use a broad range of statistical criteria while growing oblique survival trees. Last, Cui et al. (2017) found that estimating an inverse-probability weighted hazard function at each non-leaf node of a survival tree allows the RSF to converge asymptotically to the true survival function when some variables contribute both to the risk of the event and the risk of censoring, a scenario that is very likely in the analysis of medical data. The accelerated oblique RSF could incorporate this splitting technique by using Newton Raphson scoring to fit a model for the censoring distribution after which a weighted model could be fit to the failure distribution. This final item has the highest priority, as Cui et al. (2017) showed it is a requisite condition for consistency of axis-based survival trees in fairly general settings.

5.3. Conclusion

Oblique RSFs have exceptional prediction accuracy and this study has shown how they can be fit with computational efficiency that rivals their axis-based counterparts. We have also introduced a general and flexible method to estimate VI with oblique RFs, and demonstrated its effectiveness for numeric, correlated predictors.

Supplementary Materials

Supplemental materials for the current analysis are available online.

Code The code used to generate results in the current analysis are available online at https://github.com/bcjaeger/aorsf-bench and provided as supplemental material (aorsf_bench.zip).

aorsf The R package used to fit oblique RSFs in the current analysis is available online at https://github.com/ropensci/aorsf and version 0.0.7.9000 is provided as supplemental material (aorsf_package.zip).

Appendix Additional details on data used in the current analysis as well as additional details for numeric experiments (aorsf_appendix.pdf).

Disclosure Statement

No potential conflict of interest was reported by the authors.

Funding

The authors gratefully acknowledge the Center for Biomedical Informatics, Wake Forest University School of Medicine for supporting this research. The project was also supported by the National Center for Advancing Translational Sciences (NCATS), National Institutes of Health through Grant Award Number UL1TR001420. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH. Dr. Pajewski was supported by grant number P30AG021332 from the National Institutes of Health, while Dr. Spieser was supported by K25AG068253.

ORCID

Kristin Lenoir http://orcid.org/0000-0003-1834-7398
Nicholas M. Pajewski http://orcid.org/0000-0002-4447-6196

References

ARD Investigators (1989), “The Atherosclerosis Risk in Communities (ARIC) Study: Design and Objectives,” American Journal of Epidemiology, 129, 687–702. [197]

Benavoli, A., Corani, G., Demšar, J., and Zaffalon, M. (2017), “Time for a Change: A Tutorial for Comparing Multiple Classifiers through Bayesian Analysis,” The Journal of Machine Learning Research, 18, 2653–2688. [198]

Bild, D. E., Bluemke, D. A., Burke, G. L., Detrano, R., Diez Roux, A. V., Folsom, A. R., Greenland, P., Jacobs Jr, D. R., Kronmal, R., Liu, K., et al. (2002), “Multi-Ethnic Study of Atherosclerosis: Objectives and Design,” American Journal of Epidemiology, 156, 871–881. [197]

Blanche, P., Dartigues, J.-F., and Jacqmin-Gadda, H. (2013), “Estimating and Comparing Time-Dependent Areas Under Receiver Operating Characteristic Curves for Censored Event Times with Competing Risks,” Statistics in Medicine, 32, 5381–5397. [197]

Breiman, L. (2001), “Random Forests,” Machine Learning, 45, 5–32. [192,193,194]

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (2017), Classification and Regression Trees, New York: Routledge. [192]

Brilleman, S. (2018), simsurv: Simulate Survival Data. R package version 0.2.2, available at https://CRAN.R-project.org/package=simsurv. [198]

Brilleman, S. L., Wolfe, R., Moreno-Betancur, M., and Crowther, M. J. (2020), “Simulating Survival Data using the simsurv R package,” Journal of Statistical Software, 97, 1–27. [198]

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y., and Yuan, J. (2022), xgboost: Extreme Gradient Boosting, R package version 1.5.2.1. [198]

Cui, Y., Zhu, R., Zhou, M., and Kosorok, M. (2017), “Consistency of Survival Tree and Forest Models: Splitting Bias and Correction,” arXiv preprint arXiv:1707.09631. [192,205,206]

Felker, G. M., Anstrom, K. J., Adams, K. E., Ezekowitz, J. A., Fiuzat, M., Houston-Miller, N., Januzzi, J. L., Mark, D. B., Piña, I. L., Passmore, G., et al. (2017), “Effect of Natriuretic Peptide–Guided Therapy on Hospitalization or Cardiovascular Mortality in High-Risk Patients with Heart Failure and Reduced Ejection Fraction: A Randomized Clinical Trial,” JAMA, 318, 713–720. [197]

Goodrich, B., Gabry, J., Ali, I., and Brilleman, S. (2022), rstanarm: Bayesian Applied Regression Modeling via Stan, R package version 2.21.3. [198]

Graf, E., Schmoor, C., Sauerbrei, W., and Schumacher, M. (1999), “Assessment and Comparison of Prognostic Classification Schemes for Survival Data,” Statistics in Medicine, 18, 2529–2545. [196]

Harrell, F. E., Califf, R. M., Pryor, D. B., Lee, K. L., and Rosati, R. A. (1982), “Evaluating the Yield of Medical Tests,” JAMA, 247, 2543–2546. [194,199]

He, P., Eriksson, F., Scheike, T. H., and Zhang, M.-J. (2016), “A Proportional Hazards Regression Model for the Subdistribution with Covariates-Adjusted Censoring Weight for Competing Risks Data,” Scandinavian Journal of Statistics, 43, 103–122. [205]

Heath, D., Kasif, S., and Salzberg, S. (1993), “Induction of Oblique Decision Trees,” in IJCAI (Vol. 1993), pp. 1002–1007, Citeseer. [193]

Heller, G. (2012), “A Measure of Explained Risk in the Proportional Hazards Model,” Biostatistics, 13, 315–325. [198]

Hooker, G., Mentch, L., and Zhou, S. (2021), “Unrestricted Permutation Forces Extrapolation: Variable Importance Requires at Least One More...
