The Journal of Open Source Software

aorsf: An R package for supervised learning using the oblique random survival forest

Byron C. Jaeger[1], **Sawyer Welden**[1], **Kristin Lenoir**[1], **and Nicholas M Pajewski**[1]

1 Wake Forest University School of Medicine

DOI:

Software

- Review
- Repository
- Archive

Submitted:
Published:

Summary

The random survival forest (RSF) is a supervised learning method for right-censored time-to-event data that combines predictions from a large set of survival decision trees (Ishwaran, Kogalur, Blackstone, & Lauer, 2008). Similar to random forests for classification and regression (Breiman, 2001), trees in the RSF are de-correlated by using random subsets of training data to grow each tree and considering a random subset of predictor variables at each non-terminal node. Decision trees in the RSF may be axis based or oblique (Jaeger et al., 2019). Axis-based trees split non-terminal nodes using individual predictor variables, whereas oblique trees use a linear combination of variables. Although the oblique RSF proposed by Jaeger et al. (2019) generally outperformed the axis-based RSF in prediction tasks, the time to fit an oblique RSF was about 200 times that of the axis based RSF, making it difficult to use the oblique RSF in applied settings. `aorsf` is an R package optimized for supervised learning with oblique RSFs. Extensions of `aorsf`'s core features are supported by allowing users to supply their own function to identify a linear combination of inputs when growing oblique survival trees. The target audience includes both **analysts** aiming to develop an accurate risk prediction model (e.g., see Segar et al. (2021)) and **researchers** who want to conduct experiments comparing different techniques for identifying linear combinations of predictor variables (e.g., see Katuwal, Suganthan, & Zhang (2020)). Key features of `aorsf` include computational efficiency compared to existing software, extensive unit and integration testing that ensure cross-platform consistency and reproducibility, and user-friendly documentation paired with an application programming interface that facilitates proper usage of the core algorithms. Future development of `aorsf` will focus on methods to interpret oblique RSFs, including methods for variable importance compatible with oblique decision trees.

Existing software

The `obliqueRF` and `RLT` R packages support oblique classification and regression random forests but not oblique RSFs. The `ranger` and `randomForestSRC` packages support axis based RSFs but not oblique RSFs. The `obliqueRSF` R package fits oblique RSFs, but its computational inefficiency is a barrier in applied settings. `aorsf` is a re-write of the `obliqueRSF` package, with a focus on computational efficiency and flexibility, allowing users to grow oblique RSFs using Newton Raphson scoring (the default), penalized Cox regression (the default method for `obliqueRSF`), or a user-defined function to identify linear combinations of variables.

Newton Raphson scoring

The default routine for creating linear combinations of predictor variables in `aorsf` applies Newton Raphson scoring to the partial likelihood function of the Cox regression model. `aorsf` uses the same approach as the `survival` package to complete this estimation procedure efficiently. Full details on the steps involved have been made available by Therneau (2022). Briefly, a vector of estimated regression coefficients, $\hat{\beta}$, is updated in each step of the procedure based on its first derivative, $U(\hat{\beta})$, and second derivative, $H(\hat{\beta})$:

$$\hat{\beta}^{k+1} = \hat{\beta}^{k} + U(\hat{\beta} = \hat{\beta}^{k})\, H^{-1}(\hat{\beta} = \hat{\beta}^{k})$$

While it is standard practice in statistical modeling to iterate until a convergence threshold is met, the default approach in `aorsf` only completes one iteration. The rationale for this approach is based on three points. First, while completing more iterations reduces bias in the regression coefficients, it generally amounts to little or no gain in prediction accuracy for the oblique RSF due to the bias-variance trade-off. Second, computing U and H requires computation and exponentiation of the vector $X\hat{\beta}$, where X is the matrix of predictor values, but these steps can be skipped on the first iteration if an initial value of $\hat{\beta} = 0$ is chosen, allowing for a reduction in required computation. Third, using only one iteration with a starting value of 0 for $\hat{\beta}$ ensures numerical stability by avoiding exponentiation of large numbers, which can occur in later iterations depending on the scale of variables in X.

Computational efficiency

The increased efficiency of `aorsf` versus `obliqueRSF` results from improved memory management and Newton Raphson scoring. The benchmark below shows `aorsf` is about 3 times faster than `obliqueRSF` when both packages use penalized regression and about 400 times faster when `aorsf` uses Newton Raphson scoring, suggesting that the increased efficiency of `aorsf` is largely attributable to its use of Newton Raphson scoring.

```r
library(randomForestSRC)
library(microbenchmark)
library(obliqueRSF)
library(parallelly)
library(tidyverse)
library(aorsf)
library(knitr)
library(kableExtra)

# limit parallel computation to 4 cores
options(mc.cores = 4L)
# options(microbenchmark.unit="relative")

data_bench <- pbc_orsf
data_bench$id <- NULL

bm <- microbenchmark(

 obliqueRSF = ORSF(data = data_bench,
                   verbose = FALSE,
                   compute_oob_predictions = FALSE,
                   ntree = 1),
```

	Time to fit	
R package	milliseconds	ratio
obliqueRSF	138.2179	11.793034
aorsf_net	37.0067	3.157487
randomForestSRC	17.3158	1.477420
aorsf	11.7203	1.000000

```
 aorsf_net = orsf(data_train = data_bench,
                  formula = time + status ~ .,
                  control = orsf_control_net(),
                  oobag_pred = FALSE,
                  n_tree = 1),

 randomForestSRC = rfsrc(Surv(time, status) ~ .,
                         data = data_bench,
                         samptype = 'swr',
                         perf.type = 'none',
                         ntree = 1),

 aorsf = orsf(data_train = data_bench,
              formula = time + status ~ .,
              oobag_pred = FALSE,
              n_tree = 1),

 times = 1

)

summary(bm) |>
 as_tibble() |>
 select(expr, mean) |>
 mutate(mean_relative = mean / mean[expr == 'aorsf']) |>
 kable(col.names = c("R package", "milliseconds", "ratio")) |>
 add_header_above(header = c(" ", "Time to fit" = 2)) |>
 kable_styling()
```

Acknowledgements

The development of this software was supported by the Center for Biomedical Informatics at Wake Forest University School of Medicine.

References

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

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests. *The annals of applied statistics*, *2*(3), 841–860.

Jaeger, B. C., Long, D. L., Long, D. M., Sims, M., Szychowski, J. M., Min, Y.-I., Mcclure, L. A., et al. (2019). Oblique random survival forests. *The Annals of Applied Statistics*, *13*(3), 1847–1883.

Katuwal, R., Suganthan, P. N., & Zhang, L. (2020). Heterogeneous oblique random forest. *Pattern Recognition*, *99*, 107078.

Segar, M. W., Jaeger, B. C., Patel, K. V., Nambi, V., Ndumele, C. E., Correa, A., Butler, J., et al. (2021). Development and validation of machine learning–based race-specific models to predict 10-year risk of heart failure: A multicohort analysis. *Circulation*, *143*(24), 2370–2383.

Therneau, T. (2022, April). Survival package source code documentation. Retrieved from https://github.com/therneau/survival/blob/5440691d44abea537b08aeb60153a31654d66a9b/noweb