Brier scores and the redistribute-to-the-right algorithm

Terry Therneau

March 2023

1 Kaplan-Meier

The `rttright` function in the package illustrates something called the redistribute to the right algorithm. It was introduced by Efron as another way to compute the Kaplan-Meier estimate, but more so as an alternate way to understand the Kaplan-Meier. The function is much slower and less capable than `survfit`, but it has been useful as part of the test suite. Its primary purpose in the package, however, is for illustration.

Consider a set of subjects followed forward in time, each with an initial case weight of 1. Whenever someone is censored, evenly distribute the subject's current case weight to all those who are still at risk, but not censored. The basic idea is rather like someone distributing their assets, equally, to all the remaining relatives before exiting the stage.

Table 1 shows a hypothetical example with 10 subjects whose data is 1, 2, 2+, 3, 4+, 4+, 5, 5+, 8, and 9; the numbers are the follow-up times for each subject and the '+' indicates censoring. All start with a weight of 1. At time 2+ subject c transfers their weight evenly to the 7 subjects with longer survival (d–j), leaving each of them with a weight of $1 + 1/7 = 8/7$. For tied times, censors are assumed to occur after deaths, so subject b does not participate in the redistribution. At time 4+ subjects e and f each transfer a weight of 8/7 evenly to the remaining 4 (g–j), leaving each of those with a new weight of $8/7 + (1/4)(8/7) + (1/4)(8/7) = 12/7$. Finally, at time 5+ subject h transfers their weight of 12/7 equally to i and j, leaving each of those with a weight of $12/7 + (1/2)(12/7) = 18/7$. After each transfer, the total sum of the weights is unchanged.

If we form a (weighted) empricial CDF of the final data, it will have jumps of 1/10 at times 1 and 2, 8/70 at time 3, 12/70 at time 5, and 18/70 at times 8 and 9, which turns out to be exactly the step sizes in the Kaplan-Meir estimate. (In practice, the `rttright` routine first renorms the weights so as to sum to 1, thus the final weights will not have to be divided by 10 when creating the empirical CDF. However, the table was easier to illustrate initial weights of 1 rather than 1/10.) As a reminder, the KM is defined as the seqential product of the fraction who were at risk but did not have an event, at each event time. That is

```
> km <- rbind(time= c(0, 1, 2, 3, 5, 8, 9),
              km = cumprod(c(1, 9/10, 8/9, 6/7, 3/4, 1/2, 0/1)))
> km
     [,1] [,2] [,3]      [,4]      [,5]      [,6] [,7]
time    0  1.0  2.0 3.0000000 5.0000000 8.0000000    9
km      1  0.9  0.8 0.6857143 0.5142857 0.2571429    0
```

		Weight at time			
Id	Survival	0	2+	4+	5+
a	1	1	1	1	1
b	2	1	1	1	1
c	2+	1	0	0	0
d	3	1	8/7	8/7	8/7
e	4+	1	8/7	0	0
f	4+	1	8/7	0	0
g	5	1	8/7	12/7	12/7
h	5+	1	8/7	12/7	0
i	8	1	8/7	12/7	18/7
j	9	1	8/7	12/7	18/7
Total		10	10	10	10

Table 1: Hand calculation of the new RTTR weights, after each censoring time.

Another way to compute the RTTR weights is by using the inverse probability of censoring. Let $G(t)$ be the KM calculation where censoring is considered as the event. This has the values shown below.

```
> G <- cumprod(c(1, 7/8, 4/6, 2/3))
> names(G) <- c(0, "2+", "4+", "5+")
> G
        0        2+        4+        5+ 
1.0000000 0.8750000 0.5833333 0.3888889 
> 7/G
 0 2+ 4+ 5+ 
 7  8 12 18 
```

Because censoring happens after death, at time 2+ the product term is 7/8, 7 uncensored out of 8 at risk for censoring, whereas the term was 8/9 in the KM for death. (The fairly common practice of using `survfit(Surv(time, 1-status)  1)` does not give correct values for G, if there are any tied death/censoring pairs). $1/G$ has values of 1/10, 8/7, 12/7 and 18/7. That is, assigning as weight of $1/G(t)$ to an event at time t and 0 to censored values exactly reproduces the RTTR weights. The same care that about tied times used to compute G also applies to its use. Formally $G(t)$ is defined as a left continuous function: the death at time 2 is assigned a weight based on $G(2-\epsilon) = 1$, rather than $G(2+\epsilon) = 7/8$. Again, this is an unfortunately common coding error. We have used the `rttright` function as a code check for our $1/G$ methods as well as others.

The connection between inverse probability of censoring (IPC) weights and the RTTR is connected to survey sampling ideas. At any chosen time point s the RTTR for those still alive at s will be larger than one, each subject represents both themselves and some fraction of those who have been censored. The survey sampling version of the same idea gives a weight of 1/Pr(not censored) to each of these subjects, which turns out to be the same.

The addition of case weights add a small wrinkle to the algorithm: when the weight for censored subject is redistributed, it now done proportional to the case weights. Practially, it only means that the code has keep the original case weights as a separte vector from the ongoing, redistributed weight for each observation. For the IPC approach, new weights are $w_i/G(t_i+)$ where w is the original case weight.

2 Competing risks

For the case of competing risks all the above arguments translate directly. Assume for instance that there were three causes of death A= cardiac and B=cancer and C= other, and that the deaths at times 1, 3 and 9 were of type A, times 2 and 5 of type B, and the death at time 8 of type C. After application of the RTTR or IPC process we have new weights, as before. The cardiac death curve will start at 0 and have jumps of size 1/10, 12/70 and 18/70 at time 1, 5 and 9, respectively, the cancer death curve jumps of size 8/70 and 12/70 at times 3 and 5, and the other death curve a jump of size 18/70 at time 8. This matches the result of an Aalen-Johansen curve, i.e., the result of `survfit` when the status variable has been replaced with a factor variable with levels of censor, cardiac death, cancer death, and other death.

An incorrect, though common error for competing risks data is to estimate one of the failure curves using an ordinary Kaplan-Meier, with status variable 1 for the endpoint of interest and 0 for censoring *and* the other endpoints. The RTTR approach illustrates the folly of this: the KM of time to cardiac corresponds to an RTTR calculation which will redistribute the weight for subject b at $t = 2$, the point of their cancer death. This implicity assumes that subject b is still at risk for a cardiac death sometime in the future and leads, not surprisingly, to an overestimate of cumulative cardiac death risk. A correct RTTR only redistrubutes actual censorings.

3 Brier score

A well known goodness of fit estimate for binomial data is the Brier score, which is essentially an ordinary correlation coefficient using the 0/1 response as y and the prediction $\hat{p}$ from a fitted model. Recall the usual defintion

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y - \overline{y})^2}$$

To adapt R^2 for survival data involves 3 steps. The first two are straightforard: decide on a particular time point τ, then use P(death time $\leq tau$) as the response y; and decide on the intercept value $\overline{y}$ to be used as reference. For the second of these the usual choice is $1 - KM(\tau)$, the estimated overall probability of death by time τ based on the Kaplan-Meier. Below we will use the Rotterdam data set, contains long term death and recurrence status for a cohort of breast cancer patients; there is a median follow-up of 9.3 years and a median recurrence free survival of 6.9 years.

```
> # recurrent free survival (earlier of death or progression)
> #  see help(rotterdam) for explanation of ignore variable
> ignore <- with(rotterdam, recur ==0 & death==1 & rtime < dtime)
> rfs <- with(rotterdam, ifelse(recur==1 | ignore, recur, death))
> rfstime <- with(rotterdam, ifelse(recur==1 | ignore, rtime, dtime))/365.25
> rsurv <- survfit(Surv(rfstime, rfs) ~1, rotterdam)
> rsurv
Call: survfit(formula = Surv(rfstime, rfs) ~ 1, data = rotterdam)

        n events median 0.95LCL 0.95UCL
[1,] 2982   1670    6.9    6.17    7.56
> ybar <- 1- summary(rsurv, time=4)$surv
> rfit <- coxph(Surv(rfstime, rfs) ~ pspline(age) + meno + size + pmin(nodes,12),
                rotterdam)
> psurv <- survfit(rfit, newdata= rotterdam)
> dim(psurv)
data
2982
> yhat <- 1- summary(psurv, time=4)$surv
```

For the above fit, say we would like to evaluate the Brier score at $tau = 4$ years. An overall probability of death is easily extracted from the KM `rsurv`. For $\hat{p}$ we first obtain all 2982 predicted survival curves based on the fitted model, one for each subject, and then read off the prediction at 4 years for each subject. The third ingredient of R^2 is the 0/1 alive/dead status y_i of each subject at 4 years, and raises an immediate difficulty: what value should be used for subject 33, who is censored at 2.8 years? We simply do not know what their alive/dead status will be at 4 years. There are 62 such subjects. One very bad idea is to simply leave them out of the calculation. A better one is to employ the RTTR algorithm, stopping at 4 years. The result is a weight vector which is 0 for the 62 early censorings, each one's weight has been appropriately redistributed distributed over those with a longer follow-up.

```
> wt4 <- rttright(Surv(rfstime, rfs) ~ 1, times =4, rotterdam)
> table(wt4 ==0)
FALSE  TRUE
 2920    62
> brier1 <- sum(wt4 * (rfs- yhat)^2)/ sum(wt4)
> brier0 <- sum(wt4 * (rfs- ybar)^2) / sum(wt4)
> r2 <- 1- (brier1/brier0)
> temp <- c(numerator= brier1, denominator = brier0, rsquared = r2)
> round(temp,3)
  numerator denominator    rsquared
      0.252       0.282       0.107
```

Formally, the value `brier1` above is the Brier score. The result is far more useful in the r-squared form, however, since the normalizing value (denominator) can have a large variation across cutoff times. Figure 1 shows results for a Rotterdam data for a range of cutoff times. When the overall survival is above .8 the two components of the score are small. The R-squared value has also been called the "index of precision" [?]. This computation for multiple cutoff times is more efficiently done by the `brier` function.

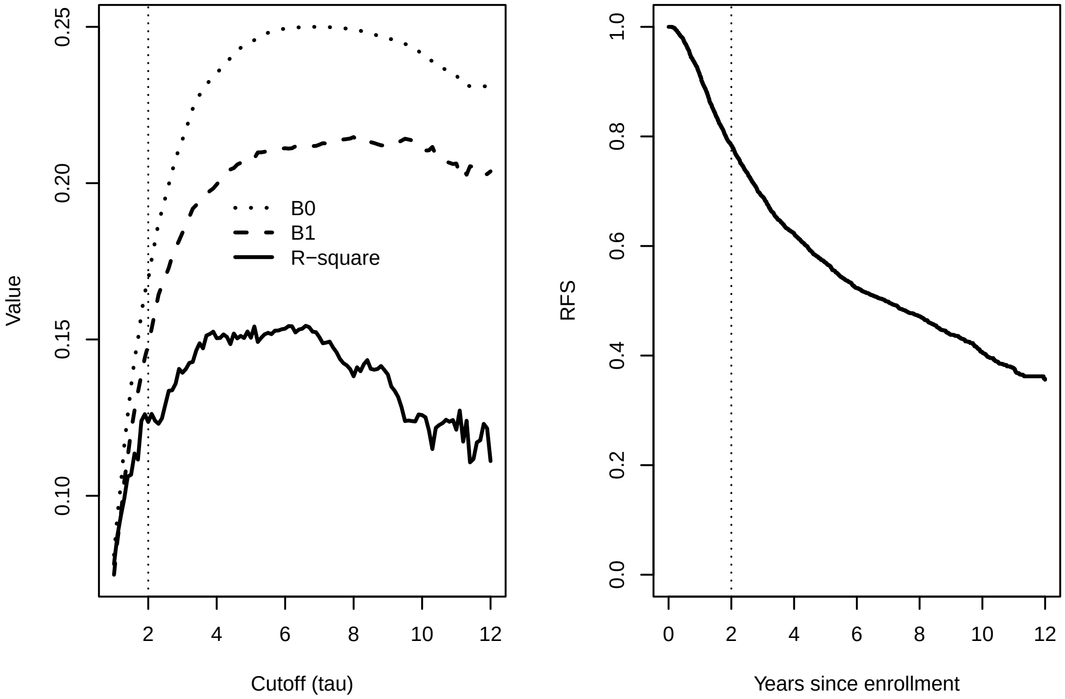

Figure 1: Components of the Brier score at .1 year increments from 1 to 12 years. B0 = the denominator, the mean squared error using the overall KM for prediction. B1 = the numerator, the mean squared error using the model's predictions of survival.

```
> cutoff <- seq(1, 12, by=.1)
> bfit <- brier(rfit, cutoff)
> names(bfit)
[1] "rsquared" "brier"    "times"
```

4 Time dependent AUROC

A parallel use for the RTTR is to compute a measure refered to as the TD-ROC, though we think that a more descriptive label is dichotomized time ROC or DT-ROC. Essentially: dichotomize time at a chosen point τ, compute the concordance C between this binomial outcome variable and the prediction score of a model. Then since concordance and AUROC are identical for a binomial outcome, relabel the result. As in the Brier score above, the key issue is dealing with subjects who are censored before time τ.

An alternative to this which does not involve dichotomization or reweighting is the thresholded concordance. In many studies, we may not be interested in the accuracy of prediction beyond a particular time threshold. Examples might be a cancer trial where the drug effect for a subject, if any, is assumed to occur within the first 3 years, or a risk score that will be used to decide patient managment over a 2 year time horizon. In both cases, predictions at longer times would be irrelevant to the question at hand. A `ymax=3` argument to the concordance function treats any observed outcomes of over 3 years as tied, causing those pairs to be ignored in the calculation.

```
> rwt <- rttright(Surv(rfstime, rfs) ~1, rotterdam, times= cutoff)
> cstat <- matrix(0, length(cutoff), 4)
> for (i in 1:length(cutoff)) {
      temp1 <- concordance(rfit, ymax= cutoff[i])
      ycut  <- ifelse(rfstime > cutoff[i], 1, 0)
      temp2 <- concordance(ycut ~ rfit$linear.predictor, weight= rwt[,i],
                           reverse=TRUE)
      cstat[i,] <- c(temp1$concordance, temp2$concordance,
                     sqrt(temp1$var), sqrt(temp2$var))
  }
> dimnames(cstat) <- list(cutoff,
                          c("Threshold C", "Discrete time C", "sd1", "sd2"))
```

Figure 2 shows the two concordance estimates. The standard errors start out similar, at early times there are few deaths and so both estimates have limited information. By year 6 the threshold C nearly at the non-thresholded value and the standard error has stabilized. The dichotomized time DT-C values are always larger, and have a standard error that first falls and then grows, the second due to an increased number of large case weights that are assigned to only a few subjects. As a comparison, the R^2 estimate of .11 is about .55 when put on the same scale (R^2 goes from -1 to 1 and C from 0 to 1). It is not fair to say that one is better than another based on overall size since the three measures are estimating different targets. A simple yes/no outcome is easier to predict than the full rank vector, which is in turn easier to predict than a continous probability.

5 Covariate based RTTR

The RTTR algorithm serves as a good reminder that when a subject is censored, it is important that they do not differ in some systematic way from everyone else who is still under observation, otherwise the concept that the others fairly "represent the future" of the censored observation will not be valid. If there is something different about the censored subject, this will bias in the estimated survival probabilities. Therefore, when estimating separate curves for subgroups of patients (e.g., males and females or treatment groups), the RTTR may be applied separately for each subgroup of interest. The assumption is, e.g., that redistributing a censored male's weight to the other males would provide a cohort whose future is "more like" what would have been if said subject were not to be censored. It is a bit like the temptation to leave one's inheritance the the child who is most like you.

Of course, one cannot guess the future. Another strategy is attempt balance. For instance, assume that 2/3 of the males but only 1/3 of the females are censored in a particular study. It can be argued that weights should then be redistibuted preferentially to males, thus preserving

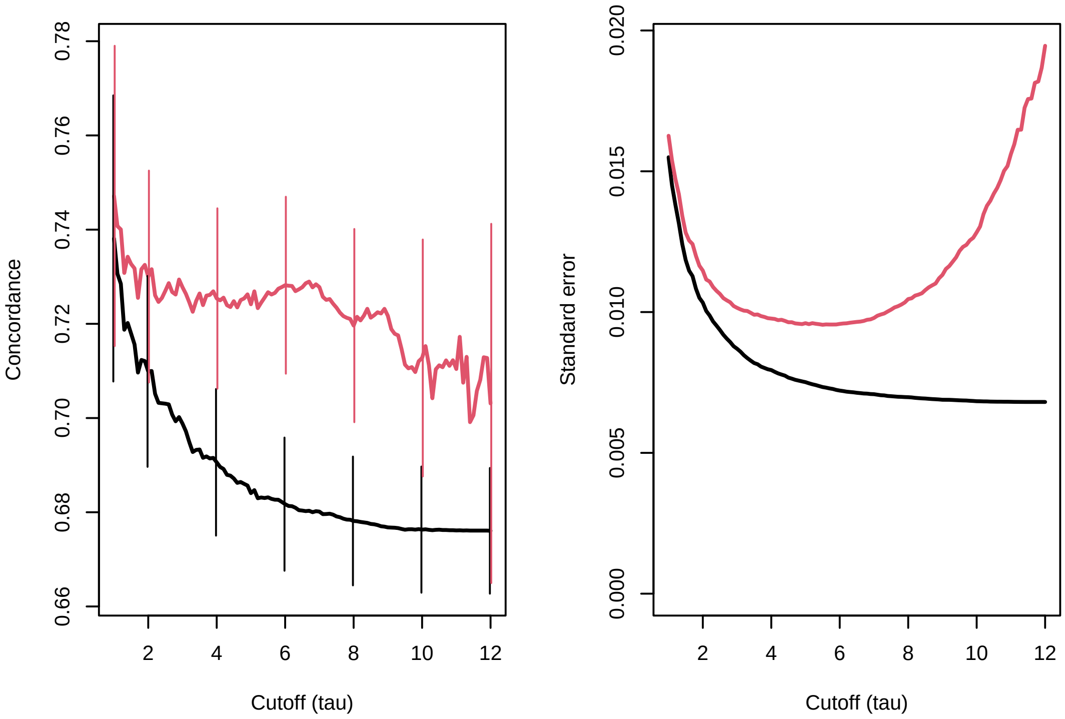

Figure 2: Left panel: Time threshold (black) and dichotomized time (red) concordance estimates for the fitted Rotterdam model, along with 95% confidence intervals. Right panel: comparative standard errors.

the overall balance of the study. There are a few ways to go about this. If the data set is large, then a separate logisitic regression model can be fit at each censoring time, with P(censored) as the target response. RTTR weights are reassigned proporional to the fitted probability, or equivalently using 1/P(uncensored). This highlights a close connection between the RTTR and marginal structural models. For smaller data sets one can smooth over time by using an overall Cox model of censoring, and then using per-time point prediction. A Cox model with select time-dependent coefficients would be an intermediate approach.

6 Delayed entry

The discussion so far has assumed that any subject is represented by a proper data set, i.e., no warning or error messages from the `survcheck` function, and we will continue to assume that. We have also assumed that all subjects start at the same time, in the same state; this assumption can be relaxed.

(This code is not yet implemented.)

- Let $p(t)$ be the probability vector at time t, one element for each state.
- At the start we almost always have $p(0) = (1, 0, 0, \ldots)$, everyone in the same starting state; the starting stated is placed first in the set of states by convention.
- Let $t0$ be the start time, which can be different than 0. Then $p(t0)$ is the emprical distribution of the states among subjects at $t0$.
- When a subject is censored (really censored) their weight is distributed among all remaining subjects in the same state, proportional to case weights. We keep two vectors: working weights, which are subject to redistribution, and the case weights, which stay constant.
- Whenever a subject enters a new state, either delayed entry to the study or a transition from another state, there is a pooling and redistribution. All the working weights in the new state are added up, and then re-apportioned among those at risk for further transitions, proportional to the case weights. It is sort of like a gang of socialist thieves: when a new member joins the crew any resources he/she brought with them, plus all resources currently extant, are pooled and re-divided: everyone is equal.

If we do this, then at any given time the weighted distribution of states is exactly the Aalen-Johansen estimate. For tied times the order of operations is first any moves to a new state, then censorings, then delayed entry. If someone is censored, with no one else to give the weight to, it isn't clear quite what to do.

Geskus has another way to do this, which is very clever. Let $G(t)$ be the censoring distribution, as before. Let $H(t)$ be the KM for entry, but computed in reverse time. Then a working weight of $cw_i/[G(s-)H(s-)]$ can be used at time s, where w is the case weight. The normalization constant c is use to make the working weights sum to 1 at the start of the proces.