Memorial Sloan-Kettering Cancer Center
Memorial Sloan-Kettering Cancer Center, Dept. of Epidemiology
& Biostatistics Working Paper Series

Year 2011 *Paper* 23

Building a Nomogram for Survey-Weighted Cox Models Using R

Marinela Capanu* Mithat Gonen†

*Memorial Sloan-Kettering Cancer Center, capanum@mskcc.org

†Memorial Sloan-Kettering Cancer Center, gonenm@mskcc.org

http://biostats.bepress.com/mskccbiostat/paper23

Building a Nomogram for Survey-Weighted Cox Models Using R

Marinela Capanu and Mithat Gonen

Abstract

Nomograms have become a very useful tool among clinicians as they provide individualized predictions based on the characteristics of the patient. For complex design survey data with survival outcome, Binder (1992) proposed methods for fitting survey-weighted Cox models, but to the best of our knowledge there is no available software to build a nomogram based on such models. This paper introduces R software to accomplish this goal and illustrates its use on a gastric cancer dataset. Validation and calibration routines are also included.

Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. http://www.jstatsoft.org/

Building a Nomogram for Survey-Weighted Cox Models Using R

Marinela Capanu **Mithat Gönen**

Abstract

Nomograms have become a very useful tool among clinicians as they provide individualized predictions based on the characteristics of the patient. For complex design survey data with survival outcome, Binder (1992) proposed methods for fitting survey-weighted Cox models, but to the best of our knowledge there is no available software to build a nomogram based on such models. This paper introduces R software to accomplish this goal and illustrates its use on a gastric cancer dataset. Validation and calibration routines are also included.

Keywords: nomogram, Cox regression, sampling design, weights.

1. Introduction

A nomogram is a graphical representation of a mathematical model involving several predictors to predict a particular endpoint based on traditional statistical methods such as Cox proportional hazards model for survival data or logistic regression for binary outcome (Kattan 2003a; Iasonos *et al.* 2008; Shariat *et al.* 2008, among others). Nomograms have been widely used for cancer prognosis, primarily because they are designed to provide estimates of the probability of an event, such as death or recurrence, which are tailored to the profile of individual patients. For survival data, the underlying model on which the nomogram is based on is typically the Cox proportional hazards model which models the relationship between a set of covariates and the hazard function of a particular failure time. The model parameters are estimated using partial likelihood and most statistical packages implement the Cox model making it a very attractive tool for survival data.

For survey data with a complex sampling design, fitting the standard partial likelihood method which ignores the design of the survey can lead to seriously misleading results (Lin 2000). Binder (1992) proposed a method for fitting the Cox proportional hazards model that takes into account the complex design of the survey sample. He derives weighted estimators for

the Cox regression coefficients and their estimates of variance. Although there is available software for building nomograms based on the standard Cox model (see Harrell's **Design** package Harrell 2001), to the best of our knowledge there are no available tools to build a nomogram in the context of survey-weighted Cox models. This article introduces R functions that address this problem. Section 2 describes the Cox model and the survey-weighted Cox models while Section 3 summarizes the use of nomograms and introduces a procedure to build a nomogram for survey-weighted Cox models. Section 4 illustrates the method using a gastric cancer dataset. Section 5 concludes with a discussion. The Appendix includes generic functions to accomplish this task.

2. Cox proportional hazards model

This section provides a brief overview of the Cox proportional hazard in the context of non-survey and survey survival data.

2.1. Non-survey survival data

The Cox (1972) proportional hazards model assumes that the hazard function of the failure time T satisfies the relationship

$$h(t|X) = h_0(t) \exp^{\beta' X(t)}, \tag{1}$$

where X is a vector of observable, possibly time dependent covariates, $h_0(\cdot)$ is an unspecified baseline hazard function, and β is the vector-valued unknown regression parameter representing the log hazard ratio. Using similar notation to that of Binder (1992) and Lin (2000), denote C the censoring time and let $\widetilde{T} = \min(T, C)$, $\Delta = I(T \leq C)$ and $Y(t) = I(\widetilde{T} \leq t)$, where $I(\cdot)$ is the indicator function. If $\{\widetilde{T}_i, \Delta_i, X_i(\cdot)\}, i = 1, \ldots, N$ is a random sample from the joint distribution of $\{\widetilde{T}, \Delta, X(\cdot)\}$, then β can be estimated by determining B to maximize the partial likelihood function so that

$$\sum_{i=1}^{N} \Delta_i \{X_i(\widetilde{T}_i) - \frac{S^{(1)}(\beta, \widetilde{T}_i)}{S^{(0)}(\beta, \widetilde{T}_i)}\}, \tag{2}$$

where

$$\begin{aligned} S^{(0)}(\beta, t) &= \frac{1}{N} \sum_{i=1}^{N} Y_i(t) \exp^{\beta' X_i(t)}, \\ S^{(1)}(\beta, t) &= \frac{1}{N} \sum_{i=1}^{N} Y_i(t) \exp^{\beta' X_i(t)} X_i(t). \end{aligned}$$

The Cox regression model can be implemented in most statistical software packages. The functions `coxph` in package **survival** or `cph` in **Design** can be used to fit Cox models in R, although only the latter can be used to build a nomogram.

2.2. Survey-weighted Cox models

In the context of survey data, the sample is drawn from a finite population via a complex survey design. The partial likelihood function in Equation 2 is no longer suitable for estimating

β as ignoring the survey design can result in misleading inferences. Binder (1992) developed a procedure for fitting proportional hazards models from survey data. Specifically, if we assume that a sample of size n is drawn from a survey population of size N using a complex design and denote the sampling weights by w_i scaled so that $\sum w_i = 1$, then Binder's procedure estimates β from the estimating equation

$$\sum_{i=1}^{n} w_i \Delta_i \{X_i(\widetilde{T}_i) - \frac{\hat{S}^{(1)}(\beta, \widetilde{T}_i)}{\hat{S}^{(0)}(\beta, \widetilde{T}_i)}\}, \tag{3}$$

where

$$\begin{aligned} \hat{S}^{(0)}(\beta, t) &= \frac{1}{n} \sum_{i=1}^{N} w_i Y_i(t) \exp^{\beta' X_i(t)}, \\ \hat{S}^{(1)}(\beta, t) &= \frac{1}{n} \sum_{i=1}^{N} w_i Y_i(t) \exp^{\beta' X_i(t)} X_i(t). \end{aligned}$$

Binder (1992) also derived a design-based variance for the weighted estimator $\hat{B}$ by assuming $\{\widetilde{T}_i, \Delta_i, X_i(\cdot)\}, i = 1, \ldots, N$ as fixed. This procedure can be implemented using the R package **survey**.

3. Nomograms

Nomograms have become a very popular tool among clinicians. A nice step by step guide for building, interpreting, and using nomograms to estimate cancer prognosis or other outcomes can be found in Iasonos *et al.* (2008). Briefly, nomograms create a simple graphical representation of a statistical predictive model mapping the predicted probability of a clinical event on a scale from 0 to 100. A clinician can then obtain the predicted probability of the event for a patient by accumulating the total points corresponding to the specific configuration of covariates for that patient. Nomograms have been shown to have high accuracy and discriminating ability for predicting outcomes in patients with cancer (Kattan 2003a,b; Shariat *et al.* 2006; Sternberg 2006; Chun *et al.* 2007; Shariat *et al.* 2008, among others). A nomogram's performance is usually evaluated in terms of discriminative ability and calibration. Discrimination refers to the ability to distinguish high-risk patients from low-risk patients and is commonly quantified via a concordance index which measures the level of concordance between the order of predicted probabilities and the order of the events of interest. One such index is Harrell's c index which, for survival data, is defined as the proportion of all pairs of subjects whose survival time can be ordered such that the subject with the higher predicted survival is the one who survived longer (see Harrell 2001, page 493). Calibration refers to whether the predicted probabilities agree with the observed probabilities and are usually assessed using calibration plots. To prevent over-fitting, validation methods such as cross-validation, bootstrap validation, or external validation are employed. This ensures that the nomogram will perform well when it is used in a new patient cohort.

3.1. Cox regression based nomograms

With independent sampling and time to event outcome, Cox proportional hazards model is the typical statistical model used to construct the nomogram. This can be easily done in R using the commands

```
R> library("Design"}
R> phmodel=cph(Surv(time, event)~ formula(predictors), x=TRUE, y=TRUE, surv=T,
se.fit=T, time.inc=24)
```

to fit the Cox model and store the 2 year survival and standard error, followed by a call to the function that builds the nomogram

```
R> nom=nomogram(phmodel, fun=surv2y, fun.at=ss, lmgp=0, lp=T),
```

where

```
R> surv=Survival(phmodel),
```

```
R> surv2y=function(x)surv(2*12,lp=x)
```

contains the 2 year predicted survival probabilities and

```
R> ss=c(0.05,0.2,0.4,0.6,0.7,0.8,0.9,0.95,0.99)
```

indicates the probabilities to be listed on horizontal axis of the graph (Harrell 2001).
Validation can then be achieved using, for example, bootstrap

```
R> validate.lrm(nomogram, B=150, dxy=TRUE)
```

and calibration graphs are obtained by calling the function `calibrate`:

```
R> graph=calibrate(nom, B=200, u=24, m=50)
R> plot(graph,main="Calibration for 2 Year Outcome")}.
```

3.2. Nomograms for survey-weighted Cox models

Building the nomogram

In the setting of complex design survey data to the best of our knowledge there is no software available for building a nomogram. In this section we present the key steps in building a nomogram for survey-weighted Cox models. These steps will be further detailed in the next section when the implementation is illustrated on a real dataset. General R functions that perform these steps altogether are provided in the Appendix. Note that the R package **survey** is needed for fitting the survey-weighted Cox model.

Step 1. Specify the complex survey design:
With survey data, before one can fit the survey-weighted Cox model and then build the nomogram, one has to first specify the sampling design of the survey using the `svydesign` function of the **survey** design. Without going into details, for example, in calling `dstr=svydesign(id~1, strata, prob, fpc, data)`, `id~1` indicates no clusters present, `strata` specifies the different strata, `prob` supplies the sampling probabilities, while `fpc` is specified as the total population

size in each stratum or as the fraction of the total population that has been sampled. A particular specification is presented in Section 4 for the gastric cancer dataset.

Note that this section is not intended to provide a comprehensive review of how to specify complex design surveys, but merely as an example to illustrate the methodology for building a nomogram in the setting of survey data. For more detailed information on survey design specification, the reader should consult the documentation of the **survey** package.

Step 2. Fit the survey-weighted Cox model:

```
R> library("survey")
R> svy.cox.fit=svycoxph(Surv(time, event)~ formula(predictors), x=TRUE,
design=dstr)
```

This is similar to fitting a regular Cox model except that now the survey design is accounted for via the design option.

Step 3. Obtain the linear predictors from the survey-weighted Cox model fitted above:

```
R> pred_lp_cox=predict(svy.cox.fit)
```

Step 4. Since there is no link between the `svycoxph` function in the **survey** package and the `nomogram` function in the **Design** package, we have to create this link using the function `ols` in the **Design** package. This approximates the model by fitting ordinary least squares to regress the linear predictors on the same predictors used to fit the survey-weighted Cox mode (for more details see Harrell 2001, Section 14.10). Note that the argument `sigma=1` is included to prevent numerical problems resulting from mean squared error of zero:

```
R> f=ols(pred_lp_cox~ formula(predictors), sigma=1,x=TRUE, y=TRUE)}
```

Step 5. Build the nomogram:

```
R> surveyNomogram=nomogram(f, fun=surv2y, funlabel=c("Prob of 2 year OS"),
fun.at=ss3, lmgp=0,lp=T)
R> mtext("2 year Overal Survival nomogram").
```

Details on the computation of the 2 year survival predicted probabilities are provided in the next section.

Bootstrap validation

For each bootstrap sample (constructed by sampling with replacement from the original data) we fit the survey-weighted Cox model following the steps outlined above. We calculate the Harrell's c index based on the normalized linear predictors from the model fitted on the boostrap data and obtain the bias by subtracting the c index for the observed data.

Calibration

Once the predicted survival probabilities at 2 years are sorted, they can be grouped into a specific number of groups (usually 4 or 5 groups) and then the median of the 2 year predicted

survival probabilities computed for each of the groups. The calibration graph plots these median estimates versus the 2 year survey weighted Kaplan-Meier estimate (obtained using the `svykm` function in the **survey** package) in each of the groups. Points close to the diagonal line indicate good calibration.

4. Application to gastric cancer

We have implemented this methodology to build a nomogram that predicts 2 year survival for patients with metastatic gastric cancer (Power, Capanu, Kelsen, and Shah 2011). This study comprises all patients with metastatic gastric/gastroesophageal junction(GEJ) adenocarcinoma who received chemotherapy at Memorial Sloan-Kettering Cancer Center from January 1999 to July 2007. The majority of patients with metastatic gastric cancer die within one year of diagnosis, and fewer than 15 per cent survive for 2 or more years. The goal of this study was to better characterize these patients with exceptional survival. As obtaining the necessary information for all of these patients would require going through their medical records which would have been too time consuming, a random sample was drawn instead. To maximize the population of interest, amongst the cohort of patients meeting eligibility criteria (total of 985 patients), all patients surviving 2 years or longer (total of 132 patients, denoted as $group \geq 24$) were included for detailed analysis and approximately 30 per cent of patients surviving less than 2 years were randomly selected (among the remaining 853 patients) for a total of 253 patients (denoted as $group < 24$). All patients had at least 2 year follow-up. To account for the sampling design we have employed survey-weighted Cox regression model as described in Section 2.2. Inverse probability weights were used. The final regression model underlying the nomogram was chosen based on the clinical and statistical significance of the predictors in univariate survey-weighted Cox models. More details on the statistical analysis are provided in Power *et al.* (2011). The key steps in the implementation of this analysis in the R follow the outline from Section 3.2 and are provided below. We note that most of the tasks below are automated via the R functions provided in the Appendix, but we present their manual implementation here to facilitate understanding.

Building the nomogram

First we load the R packages needed: **Design**, **survival**, and **survey**.

```
R> library("Design")
R> library("survival")
R> library("survey")
```

Then we read the data and declare as factors the categorical variables (code not shown)

```
R> nom_weights=read.table("nom_weightsJSS.txt", header = TRUE)
R> attach(nom_weights)
```

and exclude the observations with missing values

```
R> noNA=na.omit(subset(nom_weightsJSS,select=c(survival, surv_cens,group,
inv_weight, ssize,ECOG,Alb,Hb,Age,Differentiation,
Gt_1_m1site,lymph_only,liver_only)))
R> attach(noNA)
```

Alternatively one can impute the missing values, but we will not consider this here. Next step specifies the complex survey design which in this example is a stratified independent sampling design, as the sampling was conducted stratified based on whether the patient survived longer than 24 months or not.

```
R> dstr2=svydesign(id=~1, strata=~group, prob=~inv_weight, fpc=~ssize,
data=noNA)
```

The sampling probabilities specified by `prob` are equal to 1 for the patients in the group of long term survivors ($group \geq 24$) and are all equal to 253/853 for those that survived less than 2 years. The `fpc` option specifies the total population that has been sampled in each stratum and is equal to 132 in $group \geq 24$ and is equal to 853 for patients in $group < 24$. To fit the survey-weighted Cox model containing the age of the patients, the ECOG status, the Albumin and Hemoglobin levels, the tumor differentiation, the presence of more than one metastatic sites, the presence or absence of metastasis only in the liver, the presence or absence of metastasis only in the lymph nodes and accounting for the stratified independent sampling design we use

```
R> svy.cox.fit=svycoxph(Surv(survival,surv_cens) ~ ECOG+liver_only+Alb+Hb+Age+
Differentiation+Gt_1_m1site+lymph_only,x=TRUE,design=dstr2)
```

The estimated model parameters long with their significance level are listed below

```
                      coef exp(coef) se(coef)      z       p
ECOG2                0.96065     2.613  0.08713 11.025 0.0e+00
liver_only1          0.32708     1.387  0.11100  2.947 3.2e-03
Alb                 -0.31835     0.727  0.16756 -1.900 5.7e-02
Hb                  -0.19525     0.823  0.04376 -4.462 8.1e-06
Age                  0.00706     1.007  0.00339  2.083 3.7e-02
Differentiation2-3 -0.45802     0.633  0.10940 -4.187 2.8e-05
Gt_1_m1site1         0.05540     1.057  0.10111  0.548 5.8e-01
lymph_only1         -0.25212     0.777  0.14431 -1.747 8.1e-02
```

and the corresponding linear predictors from the model are obtained as

```
R> pred_lp_cox=predict(svy.cox.fit)
```

We also store the predicted survival for each patient as

```
R> pred_survey_cox=predict(svy.cox.fit,type="curve",newdata=noNA)
```

As described in Step 4 of Section 3.2, we approximate the model using the `ols` function.

```
R> f=ols(pred_lp_cox~ ECOG+liver_only+Alb+Hb+Age+
Differentiation+Gt_1_m1site+lymph_only, sigma=1,x=TRUE, y=TRUE,data=noNA)
```

In preparation for building the nomogram we define

```
R> dd=datadist(ECOG, Alb, Hb, Age, Differentiation, Gt_1_m1site,
liver_only, lymph_only, data=noNA)
R> options(datadist="dd")
R> ss3=c(0.05,0.2,0.4,0.6,0.7,0.8,0.9,0.95,0.99)
```

as well as define the baseline survival and the survival function to be used in building the nomogram

```
R> twoyears=pred_survey_cox[[1]]$time[which(pred_survey_cox[[1]]$time>24)[1]-1]
R> baseline=exp(log(pred_survey_cox[[1]]$surv[names(twoyears)])/
exp(svy.cox.fit$linear.predictors[1]))
R> surv2y=function(x) baseline[[1]]^exp(x).
```

Note that in the definition of `surv2y`, the index "[[1]]", could have been replaced by any number from 1 to `length(pred_survey_cox)` as `surv2y` is the same for any patient.

Finally the nomogram for 2 year survival is built as

```
R> nom=nomogram(f, fun=surv2y, funlabel=c("Prob of 2 year OS"),fun.at=ss3,lmgp=0,
lp=T, vnames="labels")
R> mtext("2 year Overal Survival nomogram")
```

and displayed in Figure 4.0.4.

Validation of the nomogram

Harrell's c-index on the original data is obtained after normalizing the linear predictors from the survey-weighted Cox model on which the nomogram was built on

```
lp_normalized=svy.cox.fit$x %*% as.matrix(svy.cox.fit$coefficients)
-mean(svy.cox.fit$x %*% as.matrix(svy.cox.fit$coefficients))
cindex.orig=1-rcorr.cens(lp_normalized,Surv(survival,surv_cens))[[1]]
```

and equals

```
R> cindex.orig
[1] 0.7575879
```

We perform bootstrap validation using 200 bootstrap datasets constructed by sampling with replacement from the original data but in such a way to maintain the same ratio of long term survivors relative to the number of patients surviving for less than 2 years (stratified bootstrap). More precisely, among the long term survivors, we sampled with replacement as many long term survivors as in the observed data (132), and similarly we sampled with replacement 253 patients of the total 253 patients surviving less than 2 years; the two groups together formed the bootstrap sample. We then repeated this process 200 times.

```
R> bootit=200

R> for(i in 1:bootit){
```

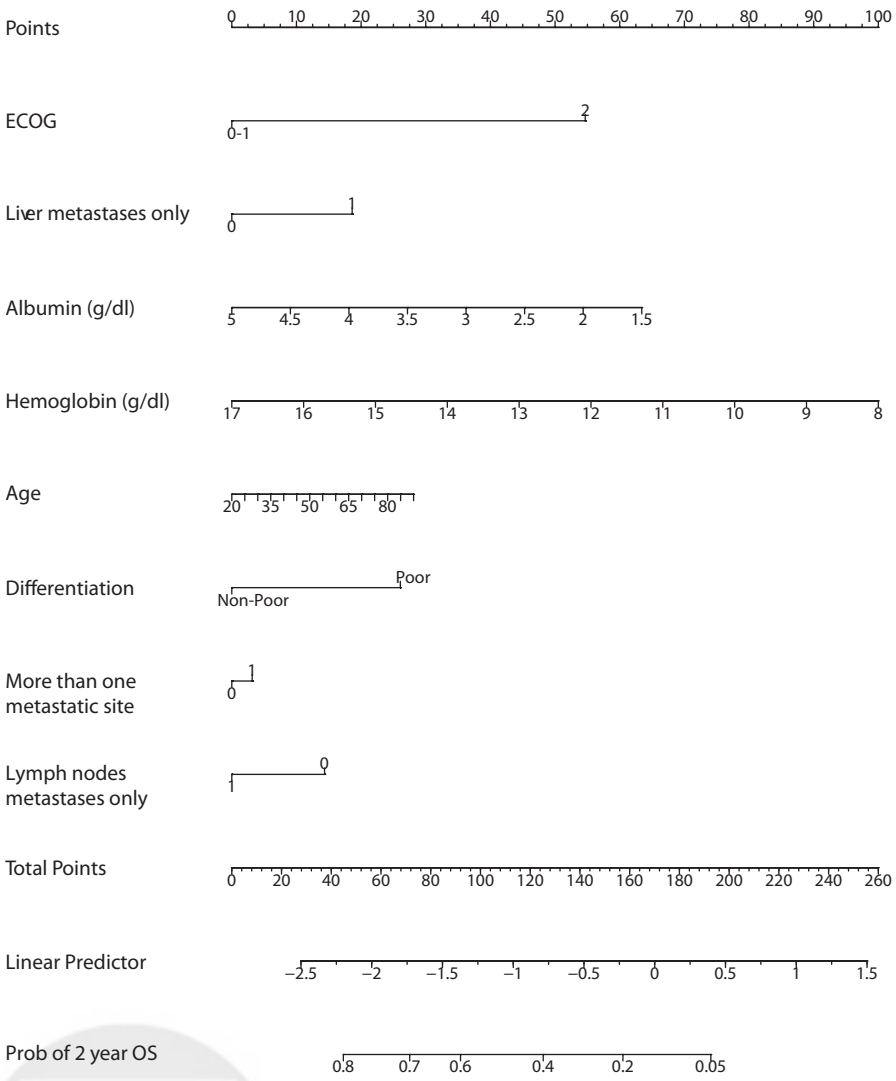

Figure 1: Clinical nomogram for metastatic gastric cancer patients treated with systemic chemotherapy estimating the probability of surviving for 2 years. Variables with the greatest discriminatory value are those with the widest point range in the nomogram. For example, an ECOG performance status of 2 (55 points) and baseline albumin of 2-3 g/dl (55-35 points) provide substantial discrimination for 2-year survival probability versus the alternative of an ECOG performance of 0-1 (0 points) and a normal serum albumin of 4 g/dl or higher (<20 points).

```
R> case=noNA[group=="long",]
R> control=noNA[group=="<24",]

R> bootindex.case=sample(1:nrow(case),replace=T)
R> boot.case.data=case[bootindex.case,]

R> bootindex.control=sample(1:nrow(control),replace=T)
R> boot.control.data=control[bootindex.control,]

R> boot.data=rbind(boot.case.data,boot.control.data)
```

For each of the bootstrap datasets the stratified independent sampling design is specified

```
R> dstr.boot=svydesign(id=~1, strata=~group, prob=~inv_weight, fpc=~ssize,
data=boot.data),
```

and the survey-weighted Cox model is fitted to the bootstrap data

```
R> boot.fit=svycoxph(Surv(survival,surv_cens) ~ ECOG+liver_only+Alb+Hb+Age+
Differentiation+Gt_1_m1site+lymph_only, x=TRUE,design=dstr.boot).
```

After normalizing the linear predictors from the survey-weighted Cox model fitted on the boostrap data (lp.boot) and evaluated on the original data (lp.test)

```
R> lp.boot=boot.fit$x%*%as.matrix(boot.fit$coefficients)-
mean(boot.fit$x%*%as.matrix(boot.fit$coefficients))
R> lp.test=svy.cox.fit$x%*%as.matrix(boot.fit$coefficients)-
mean(svy.cox.fit$x%*%as.matrix(boot.fit$coefficients))
```

the Harrell's c-index is computed for the bootstrap sample as well as on the original data

```
R> cindex.train=1-rcorr.cens(lp.boot,Surv(boot.data$survival,
boot.data$surv_cens))[[1]]
R> cindex.test=1-rcorr.cens(lp_=.test,Surv(noNA$survival,noNA$surv_cens))[[1]]
```

and the difference between the two indices is the optimism of it. After repeating this process 200 times, the final optimism estimate is estimated by the average of the 200 corresponding differences.

```
R> bias=rep(1,bootit)
R> bias[i]=abs(cindex.train-cindex.test)
}
```

An unbiased measure of the concordance probability is then obtained by subtracting the optimism estimate from the concordance probability of the original data.

```
R> print(mean(bias))
[1] 0.0106067
R> print(paste("Adjusted C-index=",(cindex.orig-mean(bias)),sep=" "),digit=5)
[1] "Adjusted C-index= 0.74698"
```

Calibration of the nomogram

We first split the data into 5 groups (note that the number of groups is chosen by the user)

```
R> grouped_pred_2years=cut(pred_2years,quantile(pred_2years,seq(0,1,0.2)),
include.lowest=T,labels=1:5)
```

and then compute the medians of the 2 year predicted survival probabilities for each of the 5 groups

```
R> median_pred_2years_1=median(pred_2years[grouped_pred_2years==1])
R> median_pred_2years_2=median(pred_2years[grouped_pred_2years==2])
R> median_pred_2years_3=median(pred_2years[grouped_pred_2years==3])
R> median_pred_2years_4=median(pred_2years[grouped_pred_2years==4])
R> median_pred_2years_5=median(pred_2years[grouped_pred_2years==5])
R> median_pred_2years=cbind(median_pred_2years_1,median_pred_2years_2,
median_pred_2years_3,median_pred_2years_4,median_pred_2years_5).
```

A simplified code to compute the medians is provided below

```
R> median_pred_2years=as.vector(by(pred_2years,grouped_pred_2years,median))
```

Survey-weighted Kaplan-Meier 2 year survival probabilities are estimated in each of the 5 groups

```
R> km_1=svykm(Surv(survival,surv_cens)~1, design=subset(dstr2,sorted==1),
se=TRUE)
R> km_2=svykm(Surv(survival,surv_cens)~1, design=subset(dstr2,sorted==2),
se=TRUE)
R> km_3=svykm(Surv(survival,surv_cens)~1, design=subset(dstr2,sorted==3),
se=TRUE)
R> km_4=svykm(Surv(survival,surv_cens)~1, design=subset(dstr2,sorted==4),
se=TRUE)
R> km_5=svykm(Surv(survival,surv_cens)~1, design=subset(dstr2,sorted==5),
se=TRUE)


R> km1_2years=km_1[[2]][which(km_1[[1]]>24)[1]-1]
R> km2_2years=km_2[[2]][which(km_2[[1]]>24)[1]-1]
R> km3_2years=km_3[[2]][which(km_3[[1]]>24)[1]-1]
R> km4_2years=km_4[[2]][which(km_4[[1]]>24)[1]-1]
R> km5_2years=km_5[[2]][which(km_5[[1]]>24)[1]-1]
R> km_observed_2years=cbind(km1_2years,km2_2years,
km3_2years,km4_2years,km5_2years)
```

along with their corresponding variances on the log scale

```
R> varlog1_2years=km_1[[3]][which(km_1[[1]]>24)[1]-1]
R> varlog2_2years=km_2[[3]][which(km_2[[1]]>24)[1]-1]
R> varlog3_2years=km_3[[3]][which(km_3[[1]]>24)[1]-1]
R> varlog4_2years=km_4[[3]][which(km_4[[1]]>24)[1]-1]
R> varlog5_2years=km_5[[3]][which(km_5[[1]]>24)[1]-1]
```

followed by estimation of the lower and upper 95 per cent confidence intervals assuming normal approximation for the log of the survival function

```
R> ll1_2years=exp(log(km1_2years)-1.96*sqrt(varlog1_2years))
R> ll2_2years=exp(log(km2_2years)-1.96*sqrt(varlog2_2years))
R> ll3_2years=exp(log(km3_2years)-1.96*sqrt(varlog3_2years))
R> ll4_2years=exp(log(km4_2years)-1.96*sqrt(varlog4_2years))
R> ll5_2years=exp(log(km5_2years)-1.96*sqrt(varlog5_2years))

R> ul1_2years=exp(log(km1_2years)+1.96*sqrt(varlog1_2years))
R> ul2_2years=exp(log(km2_2years)+1.96*sqrt(varlog2_2years))
R> ul3_2years=exp(log(km3_2years)+1.96*sqrt(varlog3_2years))
R> ul4_2years=exp(log(km4_2years)+1.96*sqrt(varlog4_2years))
R> ul5_2years=exp(log(km5_2years)+1.96*sqrt(varlog5_2years))
```

Finally, the calibration plot is obtained by plotting the medians of the 2 year predicted survival probabilities estimated by the model against the survey-weighted Kaplan-Meier 2 year survival probabilities. The graph can be viewed in Figure 2.

```
R> plot(median_pred_2years,km_observed_2years, xlim=c(0,0.75), ylim=c(0,0.75))
R> lines(x=rep(median_pred_2years_1,2),y=c(ll1_2years,ul1_2years))
R> lines(x=rep(median_pred_2years_2,2),y=c(ll2_2years,ul2_2years))
R> lines(x=rep(median_pred_2years_3,2),y=c(ll3_2years,ul3_2years))
R> lines(x=rep(median_pred_2years_4,2),y=c(ll4_2years,ul4_2years))
R> lines(x=rep(median_pred_2years_5,2),y=c(ll5_2years,ul5_2years))
R> abline(0, 1, lty= 2)
R> lines(median_pred_2years,km_observed_2years)
```

5. Discussion

Nomograms are widely used among clinicians as they provide estimates of the probability of an event, such as death or recurrence, tailored to the profile of individual patients, thus facilitating cancer prognosis. In the presence of complex design survey data with survival outcome, survey-weighted Cox models have been proposed but to the best of our knowledge there is no available software for building nomograms in this context. This article introduces R software to build, validate, and check the calibration of a nomogram for survey-weighted Cox models. The tool is illustrated step by step on a real dataset of gastric cancer and generic functions are also provided. It requires the user prespecify the complex survey design using existent functions in the **survey** package. For the validation of the nomogram, the user has to modify the generation of the bootstrap samples to fit the specific survey design. The software is easy to use and can be easily extended to binary endpoints.

Acknowledgements

The authors are grateful to Manish Shah and Derek Power for permission to use the gastric cancer data, and to Frank E. Harrell and Thomas Lumley for their helpful suggestions.

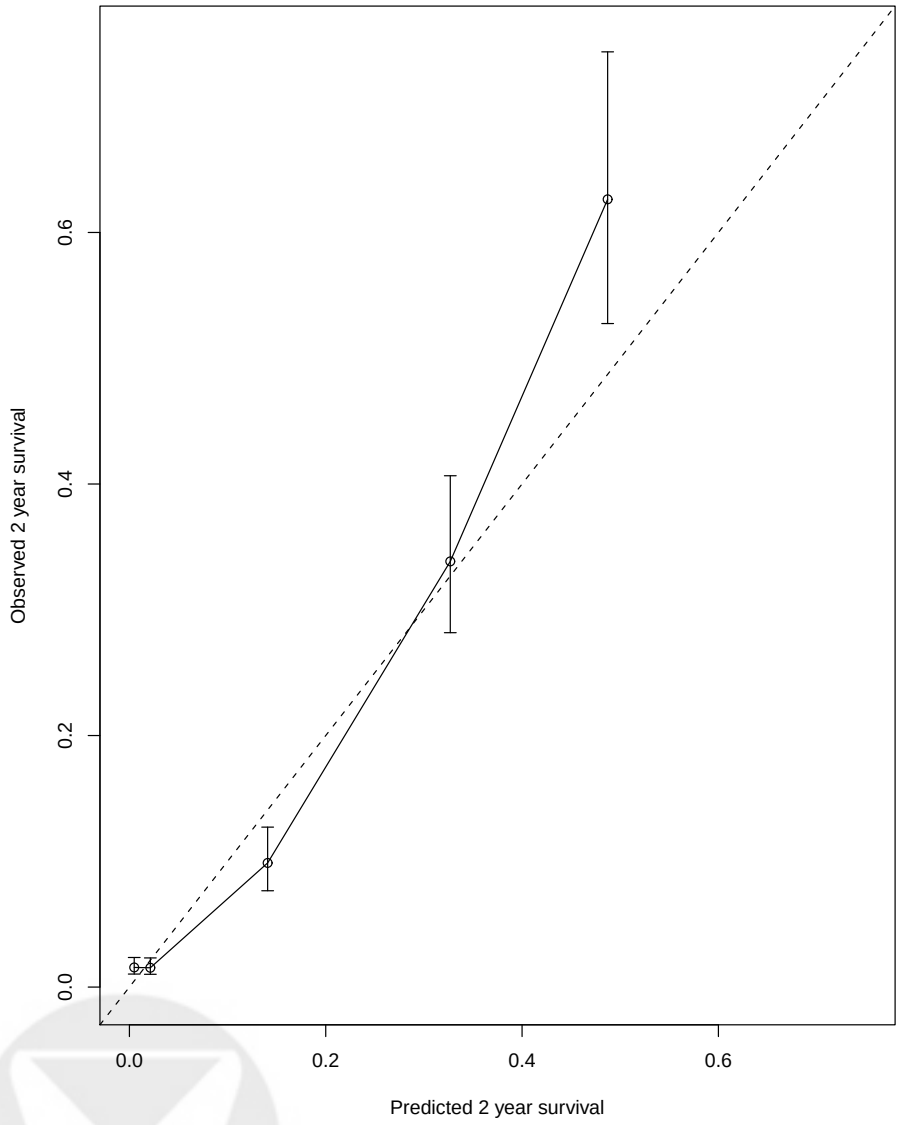

Figure 2: Calibration curve for 2 year survival. X-axis shows the nomogram predicted probability, while the Y-axis is actual 2-year survival as estimated by the Kaplan-Meier method. The dotted line represents an ideal agreement between actual and predicted probabilities of 2-year survival. The solid line represents our nomogram and the vertical bars represent 95 per cent CIs. Dots correspond to apparent predictive accuracy.

Appendix

We present general functions (available at mskcc.org/marinelacapanu), that can automatically perform the steps outlined in Section 3 and 4 to create the nomogram, validate it using bootstrap as well as produce the calibration plots. Note that the user needs to provide as input a survey design (as described in Step 1 of Section 3.2). The user also needs to ensure the dataset does not include any missing values and this can be achieved using the `na.omit` command in R. Another requirement is to run the `datadist` option to store the distribution summaries for all potential variables and insure adequate plotting ranges.

```
R> dd= datadist(variables, data)
R> options (datadist="dd")
```

The following libraries are required to call the functions:

```
library("survival")
library("survey")
library("Design")
```

Function to build the nomogram

The function `svycox.nomogram` which builds the nomogram is invoked as
`svycox.nomogram=function(.design,.model,.data,pred.at,fun.lab)`
and its arguments are described below:

- `.design` represents a survey design object;
- `.model` indicates a Cox model specification;
- `.data` contains the data on which the model is to be fit (can not contain NAs);
- `pred.at` specifies the time point at which the nomogram prediction axis will be drawn, while
- `fun.lab` designate the label of the prediction axis.

The function generates a plot and also returns a nomogram object which should be saved as it is required for the subsequent validation and calibration functions. The body of the function `svycox.nomogram` is provided below.

```
svycox.nomogram=function(.design,.model,.data,pred.at,fun.lab){
design.call=.design$call
svy.cox.fit=svycoxph(.model,x=TRUE,design=.design)
pred.lp.cox=predict(svy.cox.fit)
pred.survey.cox=predict(svy.cox.fit,type="curve",newdata=.data)
.rhs=.model[[3]]
f.form=paste("pred.lp.cox~",paste(all.vars(.model)[-(1:2)],collapse="+"))
.f=ols(as.formula(f.form),sigma=1,x=TRUE,y=TRUE,data=noNA)
```

```
.ss3=c(0.05,0.2,0.4,0.6,0.7,0.8,0.9,0.95,0.99)
.ss3.label=100*.ss3
time.at=pred.survey.cox[[1]]$time[which(pred.survey.cox[[1]]$time>pred.at)[1]-1]
.baseline=exp(log(pred.survey.cox[[1]]$surv[names(time.at)])/
exp(svy.cox.fit$linear.predictors[1]))

.tempfun=function(x) .baseline[[1]]^exp(x)

.nom=nomogram(.f, fun=.tempfun, funlabel=fun.lab,fun.at=.ss3,lp=T,
              vnames="labels")
return(list(nomog=.nom,design=.design,svy.cox=svy.cox.fit,
            preds=pred.survey.cox,pred.at=pred.at))
}
```

To build the nomogram for the gastric cancer data as described in Section 4 one would call this function as below

```
mynom=svycox.nomogram(.design=dstr2,
                  .model=Surv(survival,surv_cens)~ECOG+liver_only+Alb+Hb+Age+
                 Differentiation+Gt_1_m1site+lymph_only,
                  .data=noNA,
                  pred.at=24,
                  fun.lab="Prob of 2 Yr OS")
```

Function to validate the nomogram

The function `validate.svycox` validates the nomogram using bootstrap and uses as arguments `.boot.index` which includes a matrix of bootstrap sample indicators with the number of rows the same as the number of rows in the data and the number of columns being the number of bootstrap samples; `.nom` contains a nomogram object returned from `svycox.nomogram`, and `.data` is the dataset on which the validation will take place. The function prints the estimated optimism and returns the vector of optimism values for each bootstrap values so the user can summarize it with the measure of choice. The function `validate.svycox` is included below:

```
validate.svycox=function(.boot.index,.nom,.data){

internal.validate.func=function(.boot.vec,.nom2,.data2){
boot.data=.data2[.boot.vec,]
design.boot=svydesign(id=~1,
    strata=as.formula(paste("~",names(.nom2$design$strata))),
        prob=as.formula(paste("~",names(.nom2$design$prob))),
        fpc=as.formula(paste("~",colnames(.nom2$design$fpc$popsize))),
        data=boot.data)
boot.fit=svycoxph(formula(.nom2$svy.cox), x=TRUE,design=design.boot)
lp.boot=boot.fit$x%*%as.matrix(boot.fit$coefficients)-
                  mean(boot.fit$x%*%as.matrix(boot.fit$coefficients))
```

```
lp.test=(.nom2$svy.cox)$x%*%as.matrix(boot.fit$coefficients)-
                mean((.nom2$svy.cox)$x%*%as.matrix(boot.fit$coefficients))
cindex.train=1-rcorr.cens(lp.boot,
             Surv(boot.data$survival,boot.data$surv_cens))[[1]]
cindex.test=1-rcorr.cens(lp.test,Surv(.data2$survival,.data2$surv_cens))[[1]]
return(cindex.train-cindex.test)
}

val.res=apply(boot.index,2,internal.validate.func,.nom2=.nom,.data2=.data)
print(mean(val.res))
return(val.res)

}
```

Note that generating the bootstrap sample is design dependent and thus we did not make it part of the function `validate.svycox`. The user has to generate the bootstrap samples consistent with the design used. For example, to validate the nomogram for the gastric cancer data, we first use stratified bootstrap to generate the bootstrap data for the matched case-control design

```
bootit=200

cases=which(noNA$group=="long")
controls=which(noNA$group=="<24")

boot.index=matrix(NA,nrow(noNA),bootit)
for(i in 1:bootit){
boot.index[,i]=c(sample(cases,replace=T),sample(controls,replace=T))
}
```

and then call the `validate.svycox` function to validate the nomogram.

```
myval=validate.svycox(boot.index,mynom,noNA)
```

Function to check calibration

The function `calibrate.svycox` uses the arguments `.nom` which stores a nomogram object from `svycox.nomogram`, `.timept` indicating the time point at which calibration will take place (it defaults to the time value of the prediction axis in the nomogram), and `.ngroup` specifying the number of groups to be formed for validation purposes. The function returns a calibration plot.

```
calibrate.svycox=function(.nom,.timept=.nom$pred.at,.ngroup=5){
.loc=max(which(.nom$preds[[1]]$time<=.timept))
pred.timept=rep(NA,.nom$svy.cox$n)
for(i in 1:length(pred.timept)) pred.timept[i]=.nom$preds[[i]]$surv[.loc]
pred.timept.grp=cut(pred.timept,
```

```
          quantile(pred.timept,seq(0,1,1/.ngroup)),include.lowest=T, labels=1:.ngroup)
.predicted=tapply(pred.timept,pred.timept.grp,median)
.observed=matrix(NA,nrow=.ngroup,ncol=3)
colnames(.observed)=c("Observed","Lower 95%","Upper 95%")
for(i in 1:.ngroup){
  .km1=svykm(as.formula(paste(names(.nom$svy.cox$model)[1],"~","1")),
                        design=subset(.nom$design,pred.timept.grp==i),
                  se=TRUE)
  .km1.timept=.km1[[2]][which(.km1[[1]]>.timept)[1]-1]
  .varlog1.timept=.km1[[3]][which(.km1[[1]]>.timept)[1]-1]
  .ll1.timept=exp(log(.km1.timept)-1.96*sqrt(.varlog1.timept))
  .ul1.timept=exp(log(.km1.timept)+1.96*sqrt(.varlog1.timept))
  .observed[i,]=c(.km1.timept,.ll1.timept,.ul1.timept)
  }
plot(.predicted,.observed[,1],xlim=0:1,ylim=0:1,
     xlab="Predicted",ylab="Observed",pch=16)
arrows(x0=.predicted,y0=.observed[,2],y1=.observed[,3],
      angle=90,code=3,length=0.1,lwd=2)
abline(0,1,lty=2)
return(cbind(Predicted=.predicted,.observed))
}
```

To obtain the calibration plot in Figure 2 one can use the syntax

```
calibrate.svycox(mynom)
```

References

Binder DA (1992). "Fitting Cox's Proportional Hazards Models from Survey Data." *Biometrika*, **79**, 139–147.

Chun FK, Karakiewicz PI, Briganti A, et al (2007). "A Critical Appraisal of Logistic Regression-Based Nomograms, Artificial Neural Networks, Classification and Regression-Tree Models, Look-up Tables and Risk-Group Stratification Models for Prostate Cancer." *BJU Int*, **99**, 794–800.

Harrell FE (2001). *Regression Modelling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer-Verlag.

Iasonos A, Schrag D, Raj GV, Panageas KS (2008). "How to Build and Interpret a Nomogram for Cancer Prognosis." *Journal of Clinical Oncology*, **26**, 1364–1370.

Kattan MW (2003a). "Comparison of Cox Regression with Other Methods for Determining Prediction Models and Nomograms." *Journal of Urology*, **170**, S6–9.

Kattan MW (2003b). "Nomograms Are Superior to Staging and Risk Grouping Systems for Identifying High-Risk Patients: Preoperative Application in Prostate Cancer." *Curr Opin Urol*, **13**, 111–116.

Lin DY (2000). "On Fitting Cox's Proportional Hazards Models to Survey Data." *Biometrika*, **87**, 37–47.

Power DG, Capanu M, Kelsen DP, Shah MA (2011). "Development of a Nomogram to Predict 2-Year Survival With Metastatic Gastric Cancer." Under review.

Shariat SF, Karakiewicz PI, Palapattu GS, et al (2006). "Nomograms Provide Improved Accuracy for Predicting Survival after Radical Cystectomy." *Clinical Cancer Research*, **12**, 6663–6676.

Shariat SF, Karakiewicz PI, Suardi N, Kattan MW (2008). "Comparison of Nomograms with Other Methods for Predicting Outcomes in Prostate Cancer: A Critical Analysis of the Literature." *Clinical Cancer Research*, **14**, 4400–4407.

Sternberg CN (2006). "Are Nomograms Better than Currently Available Stage Groupings for Bladder Cancer." *Journal of Clinical Oncology*, **24**, 3819–3820.

Affiliation:

Marinela Capanu
Department of Epidemiology and Biostatistics
Memorial Sloan-Kettering Cancer Center
E 307, 63rd St, 3rd Fl, NY, 10021
E-mail: capanum@mskcc.org

Mithat Gönen
Department of Epidemiology and Biostatistics
Memorial Sloan-Kettering Cancer Center
E 307, 63rd St, 3rd Fl, NY, 10021
E-mail: gonenm@mskcc.org

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd