Risk Score Imputation tutorial (Hsu 2009)

Nikolas S. Burkoff[**], Paul Metcalfe[*], Jonathan Bartlett[*] and David Ruau[*]

[*]AstraZeneca, B&I, Advanced Analytics Centre, UK
[**]Tessella, 26 The Quadrant, Abingdon Science Park, Abingdon, OX14 3YS, UK

July 23, 2020

1 Introduction

In this vignette we use the `InformativeCensoring` R library to perform the multiple imputation (MI) method of Chiu-Hsieh Hsu and Jeremy Taylor [1], which in this package is called 'risk-score imputation'. The purpose of the imputation method is two fold. First, to attempt to remove bias in standard analyses when it is thought that censoring may be informative. Second, to improve efficiency by imputing events times for individuals who were censored. The first part of this vignette describes the method and the second part shows the package in use.

2 Theory

In this section we describe the risk score imputation method and we refer readers to [1] for further details. Consider a two arm time to event data set where subject i has event time T_i and potential censoring time C_i. For each subject we observe a time $X_i = \min(T_i, C_i)$ and event indicator Δ_i which $= 1$ if the subject was observed to have had an event at X_i and $= 0$ otherwise. The independent censoring assumption states that T_i and C_i are independent, and when this assumption is violated, standard methods for inference are in general invalidated.

The risk score imputation approach creates multiple imputations of event times for those subjects whose event times were censored. The imputation procedure utilizes subject level covariates, which may be time-varying, and relaxes the assumption of independent censoring to an assumption that the event time and censoring time are conditionally independent given the covariates. This means that covariates which are known or believed to be related both to the hazard of failure and the hazard of censoring should be included in the imputation process. Including covariates which are only related to the hazard of failure is also recommended, as this is expected to increase the efficiency of the resulting inferences.

The method works by generating m imputed event times $Y_i^m \geq X_i$ and event indicators Δ_i^m. The data $\{Y_i^m, \Delta_i^m\}$ is imputed by creating a risk set of *similar* subjects to subject i and then using a procedure called Kaplan-Meier imputation (KMI). The creation of risk sets and the KMI procedure are described below.

Once the M data sets have been imputed, standard time to event statistical analyses (for example the log rank test) can be applied to each data set and the results combined (as described below) to produce point estimates of model parameters or to perform a hypothesis test (e.g. of equality of survivor functions).

2.1 Calculation of Risk Set

For each censored subject i we first calculate a risk set $R(i^+, NN)$ which contains the *nearest* NN subjects to subject i (in the same treatment group) with event/censoring time $> X_i$, where NN is a user specified number. When calculating risk sets the separate treatment groups are considered independently. If only $n \leq NN$ subjects in the same arm have event/censoring time $> X_i$ then all n subjects are part of the risk set and if $n = 0$ no imputation can be made and $Y_i^m = X_i$ and $\Delta_i^m = \Delta_i$.

In order to find the *nearest* subjects to subject i we use proportional hazards models to reduce the auxiliary variables of subjects into a pair of risk scores. In the case of only time independent covariates, for each treatment group independently, we fit a pair of Cox proportional hazard models, one for event times $\lambda_f(t)\exp(\beta_f \bar{\mathbf{V}}_f)$ (where $\bar{\mathbf{V}}_f$ is the vector of auxiliary variables used in the Cox model) and one for censored times $\lambda_c(t)\exp(\beta_c \bar{\mathbf{V}}_c)$.

The risk scores for each subject are linear combinations of the auxiliary variables, specifically $\hat{RS}_f = \hat{\beta}_f \bar{\mathbf{V}}_\mathbf{f}$ and $\hat{RS}_c = \hat{\beta}_c \bar{\mathbf{V}}_\mathbf{c}$. These risk scores are centred and scaled by subtracting the mean and dividing by their standard deviation to give normalized scores ($\hat{RS}^*_f$ and $\hat{RS}^*_c$). If either model cannot be fitted as all subjects considered have an event (or all are censored) then $\hat{RS}^*_f$ (or $\hat{RS}^*_c$) are set to zero for all subjects.

The distance between subjects j and k is then given by

$$d(j,k) = \sqrt{w_f(\hat{RS}^*_f(j) - \hat{RS}^*_f(k))^2 + w_c(\hat{RS}^*_c(j) - \hat{RS}^*_c(k))^2}$$

where w_c is a user specified weighting between 0 and 1 and $w_f = 1 - w_c$.

The NN subjects with smallest $d(i,.)$ with event/censoring time $> X_i$ form the risk set for subject i[1].

For data sets with time dependent covariates, following [1], 'for every censored observation these two time-independent proportional hazard models are fitted to the data of those at risk at the censoring time using the currently available auxiliary variables as fixed covariates'. I include subjects who leave the study at time > the censored observation and normalize only these scores. There can be problems with convergence when trying to fit a Cox model to a small number of subjects, therefore a 'minimum subjects' option is included and the simplified example below describes its function:

Suppose the times of leaving the trial are given by 0.5, 1, 2.5, 2.5, 5, 10 and 20 then for the subject censored at time 10, a Cox model would have been fitted to only the sample 20. If the minimum subjects parameter is set to 4 then the subjects with time >= 2.5 will be included in the Cox model fit (with time dependent variables at their values at time 2.5).

2.2 Kaplan-Meier Imputation

In order to impute $\{Y^m_i, \Delta^m_i\}$ we take the given risk set $R(i^+, NN)$ and use KMI. We draw the Kaplan-Meier estimator of the subjects in the risk set. We then sample $U \sim [0,1]$ and take the time at which the KM estimator equals U (see the Figure below for further details) as the imputed event time.

In certain cases we wish to impute the event time for all subjects; however, in other cases we are interested in imputing what the data would look like on a given calendar date, the data cut off (DCO) date. For example, if a subject was recruited 10 days after the study started and was withdrawn 20 days later, we could impute that the subject and an event after 115 days on the trial. However, if the cutoff date was 100 days after the trial start date then this event would not have been observed.

For each subject a `DCO.time`, D_i is required. In the example above $D_i = 90$ as after 90 days on the study the subject would have been censored at the DCO date. In general, if the imputed time $Y^m_i \geq D_i$ then $Y^M_i = D_i$ and $\Delta^m_i = 0$, i.e. the subject is censored at the DCO date[2].

2.3 Bootstrapped data to fit the models

The addition of a bootstrapping step ensures that the multiple imputations are proper, such that Rubin's rules provide a valid estimate of variance [1]. Specifically, for each imputed data set, proportional hazard models are fitted to a bootstrapped data set (keeping the same treatment group allocation ratio) and risk scores are calculated. The risk set for subject i from the original data set is then given by the nearest neighbours to i in the bootstrapped data set using the calculated risk scores from the newly fitted models. For each subject i, I calculate the raw risk score for subject i using the fitted models[3]. I then take this score and the normalized scores from the (bootstrapped) data set used to fit the model to calculate a normalized score for subject i (using the same mean and variance normalization factors). These scores can then be used for risk set identification for subject i.

[1] If there are ties, so that two subjects are exactly the same distance from subject i and including them both would increase the size of the risk set to $> NN$ then both are included.

[2] Note this was not mentioned in [1], however, by setting $D_i =$ `Inf`, the original method can be reproduced– see below

[3] In the time dependent case, subject i uses the values of its time dependent covariates at its censoring time

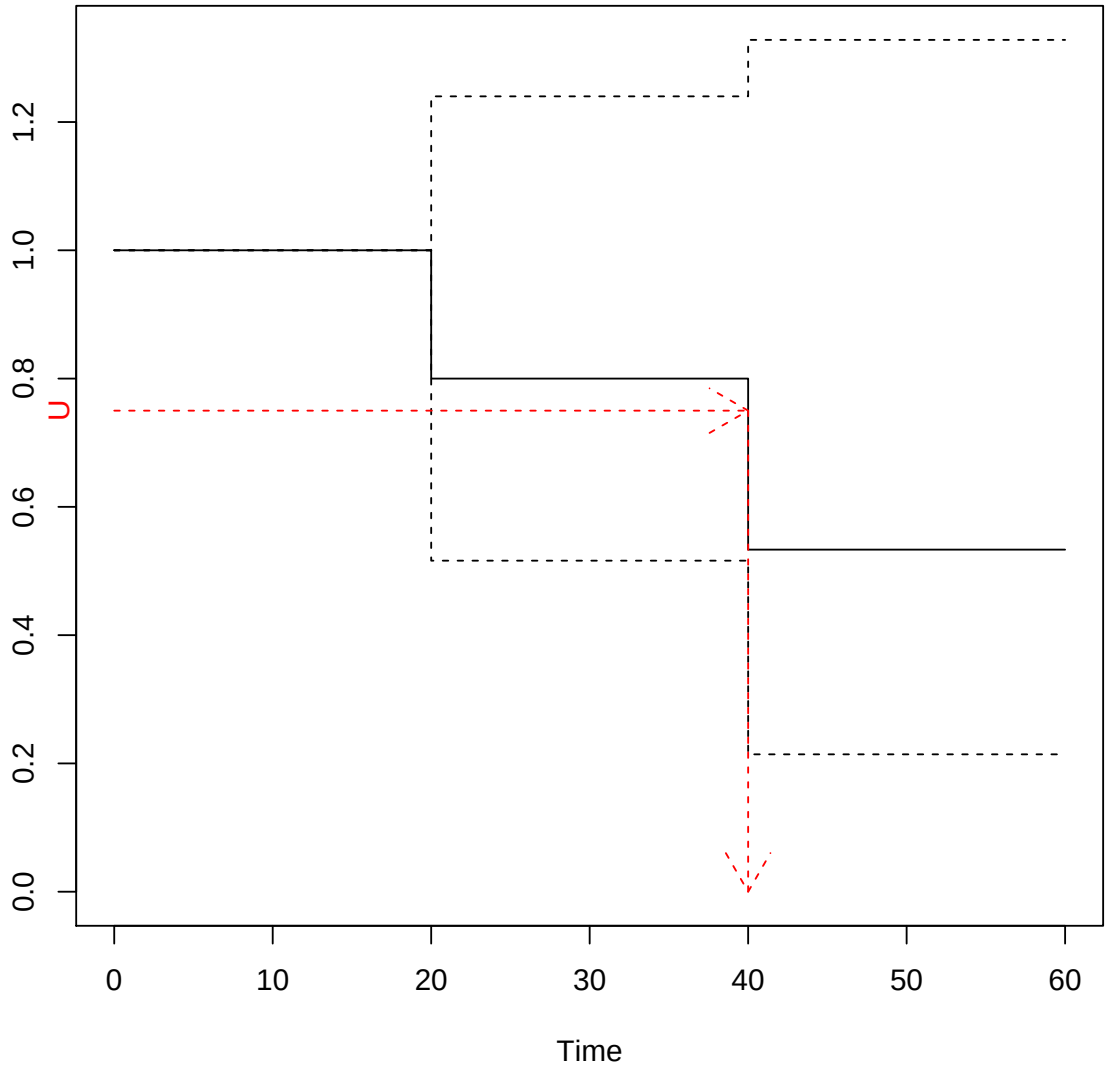

Figure 1: Kaplan-Meier Imputation. In this example, for the KM curve shown, we sampled $U = 0.75$ and this implies an imputed event time of 40. If U is less than any value on the KM curve (e.g. if $U = 0.1$ in this example) then the subject's imputed time is the last censored time of the risk set (in this example time 60) and the subject is censored rather than having an event at this time.

2.4 Test Statistics

Given M imputed data sets, we can perform time to event statistical analyses on each data set. The results can be combined to give a single p-value estimate in two distinct ways:

- **meth1:** Each data set produces a single point estimate for the null hypothesis ($\theta = \theta_0$) and these can be combined to obtain a single point estimate $\bar{\theta}$ with associated variance $V_1 = U_1 + (1+M^{-1})B_1$ where B_1 is the sample variance of the M point estimates and U_1 is the average of the M variance estimates. The test statistic $D = (\bar{\theta} - \theta_0)' V_1^{-1} (\bar{\theta} - \theta_0)$ has a F_{1,v_1} distribution with $v_1 = 4 + (t-4)(1 + (1 - 2t^{-1})/r)^2$ where $t = M - 1$ and $r = (1 + M^{-1})B_1 U_1^{-1}$. Specifically, the p-value is given by the R code `(1-pf(D,1,v1))`. **Note, the degrees of freedom are given by [2] rather than [1] (which used** $4 + (t-4)(1 + (1 - 2t^{-1})/r)$**).**
- **meth2:** Each data set produces a (normal) test statistic $Z_1, Z_2, \ldots, Z_m$ and these can be averaged to give an overall test statistic $\bar{Z}$ with variance $V_2 = 1 + (1 + M^{-1})B_2$ where B_2 is the sample variance of the Z_i. A t-test statistic with v_2 degrees of freedom can be used to with the statistic $s = \bar{Z}/\sqrt{V_2}$ where $v_2 = [1 + (M/(M+1))/B_2]^2 (M-1)$. Specifically, the p-value is given by the R code `2*(1-pt(abs(s),v2))`.

We refer the reader to [1] for further details.

3 Using the package

We first load the package and set the seed for reproducibilty:

```
library(InformativeCensoring)
set.seed(421234)
```

4 Time Independent Covariates

In this Section we apply the method to time to event data with only time independent covariates.

4.1 Data

We use a simulated data set inspired by the simulation procedure given in [1]. We first load the data:

```
data(ScoreInd)

head(ScoreInd)
```

```
##   Id arm Z1        Z2 Z3        Z4 Z5 event      time to.impute  DCO.time
## 1  1   0  1 0.7956354  0 0.5071207  1     1 0.8723524     FALSE 1.4981742
## 2  2   0  0 0.4482940  0 0.2975735  1     0 0.6155495      TRUE 0.9791961
## 3  3   0  0 0.3153451  1 0.1770421  1     0 1.1929294      TRUE 1.2161512
## 4  4   0  1 0.4998216  1 0.3376044  0     1 2.0661984     FALSE 2.7655619
## 5  5   0  0 0.6134157  1 0.5640466  1     1 0.7885539     FALSE 1.0267005
## 6  6   0  1 0.7215678  1 0.7771897  0     1 2.0486443     FALSE 2.6222115
```

The data set contains the following columns:

- `Id`: Subject Id
- `arm`: Treatment group (0=control, 1=active)
- `Z1-Z5`: Time independent covariates; Z1, Z3 and Z5 are binary, Z2 and Z4 are real valued

- `event`: Event indicator (0=censored, 1=had event), Δ_i.
- `time`: The time the subject was censored/had event (in years), X_i.
- `to.impute`: Should the subject's time to event be imputed – if subject had event this column is ignored
- `DCO.time`: The time the subject would have been censored if they were still on the trial at the data cutoff time, D_i, if an event time is imputed after DCO.time then the subject will be censored at DCO.time

We ensure that the treatment group flag is a factor and that the control group is the first level

```
ScoreInd$arm <- factor(ScoreInd$arm)

levels(ScoreInd$arm)

## [1] "0" "1"
```

The risk score imputation procedure needs to know which columns of the data frame represent subjects' event indicator, time on study, Id, treatment arm, DCO time and whether times are to be imputed. The `col.headings` function is used to setup this information:

```
col.control <- col.headings(has.event="event", time="time",Id="Id",arm="arm",
                            DCO.time="DCO.time", to.impute="to.impute")
```

If administrative censoring is being taken into account then an additional argument is required to the `col.headings` function (see later in the vignette for further details).

4.2 Imputed Dataset

We use the `ScoreImpute` function to generate the imputed data sets:

```
imputed.data.sets <- ScoreImpute(data=ScoreInd,event.model=~Z1+Z2+Z3+Z4+Z5,
                                 col.control=col.control, m=5,
                                 bootstrap.strata=ScoreInd$arm,
                                 NN.control=NN.options(NN=5,w.censoring = 0.2))
```

The `ScoreImpute` function uses the following arguments:

- `data` The data frame to be used for the imputation
- `event.model`: The right hand side of the formula for the Cox model fit for the model used to calculate the time to event scores ($\hat{RS}_f$). The terms `cluster` and `tt` cannot be used in the model formula.
- `censor.model`: The right hand side of the formula for the Cox model fit for the model used to calculate the time to censoring scores ($\hat{RS}_c$). If this argument is not used then the `event.model` argument is used instead.
- `col.control`: Key column names of the data set, see Section 4.1 for further details
- `m` The number of data sets to impute (must be > 4)
- `bootstrap.strata` When performing the bootstrap procedure to generate the data sets for the model fits, the strata argument for the bootstrap procedure (see `help(boot)` for further details).
- `NN.control`: The options used by the risk score imputation method when calculating the risk set for each subject. The `NN.options` function should be used with the following two options:
  - `NN`: The size of the risk set.

- `w.censoring` The weighting (w_c) to be applied to the censoring score ($\hat{RS}_c$) when calculating the distance between subjects. The weighting, w_f applied to the event score ($\hat{RS}_f$) is given by `1-w.censoring`.
- `min.subjects` Only used for the time dependent case: the minimum number of subjects to be included when fitting the Cox models, see `help(NN.options)` for default value.

- `time.dep`: Additional data to perform time dependent score imputation method, see Section 5 for details.
- `parallel`, `ncpus`, `cl`: parallelism options, see `help(gammaImpute)` and the `parallel` package vignette for further details – note when using `parallel="multicore"` or `parallel="snow"` it is necessary to set the random number generator to type L'Ecuyer-CMRG using the command `RNGkind("L'Ecuyer-CMRG")` in order to ensure proper random number stream behaviour. A warning is output if this is not the case, see `parallel` package vignette for further details.

Any additional arguments are passed to the Cox model fit function (`survival::coxph`). Note, the `subset` and `na.action` arguments cannot be used (`na.fail` is used)

Cox model convergence issues: There may be issues with convergence of the various Cox models, especially in the time dependent case and those with not many data points with lots of covariates. If this occurs a warning `Warning in fitter(X, Y, strats, offset, init, control, weights = weights, : Ran out of iterations and did not converge` is output. It is possible to use ridge regression by including a `ridge` term in the model formula, for example `event.model=` $\sim$ `Z1+Z2+Z3+Z4+ridge(Z5,theta=1)`; see `help(ridge)` for further details.[4]

Accessing individual imputed data sets: We use the `ExtractSingle` function to extract out a single imputed data set. The `index` argument is an integer between 1 and m allowing the user to specify which imputed data set is to be extracted:

```
#for the third data set
imputed.data.set <- ExtractSingle(imputed.data.sets,index=3)
```

We can view the imputed data. Note the two new columns, `impute.time` and `impute.event`:

```
head(imputed.data.set$data)

##   Id arm Z1        Z2 Z3        Z4 Z5 event      time to.impute  DCO.time
## 1  1   0  1 0.7956354  0 0.5071207  1     1 0.8723524     FALSE 1.4981742
## 2  2   0  0 0.4482940  0 0.2975735  1     0 0.6155495      TRUE 0.9791961
## 3  3   0  0 0.3153451  1 0.1770421  1     0 1.1929294      TRUE 1.2161512
## 4  4   0  1 0.4998216  1 0.3376044  0     1 2.0661984     FALSE 2.7655619
## 5  5   0  0 0.6134157  1 0.5640466  1     1 0.7885539     FALSE 1.0267005
## 6  6   0  1 0.7215678  1 0.7771897  0     1 2.0486443     FALSE 2.6222115
##   impute.time impute.event
## 1   0.8723524            1
## 2   0.6690834            1
## 3   1.2161512            0
## 4   2.0661984            1
## 5   0.7885539            1
## 6   2.0486443            1
```

[4]The formula and data get passed into `coxph` and then `predict(coxph.model,type="lp")` and `predict(coxph.model,type="lp",newdata=...)` are used. For complex formulae it may be worth checking directly that these functions perform as you expect before using them inside the Score Imputation procedure.

4.3 Model Fit the Imputed Data

Given the imputed data sets we use the `ImputeStat` function to fit a model to each data set and we again use the `ExtractSingle` to view the individual fit:

```
logrank.fits <- ImputeStat(imputed.data.sets,method="logrank",
                           formula=~arm+strata(Z1,Z3))

third.fit <- ExtractSingle(logrank.fits,index=3) #view log rank fit for third data set
print(third.fit)

## Method used: logrank (estimator for O-E)
## Point Estimate: -30.36566
## Variance estimate: 64.3452
## Z stastistic: -3.785512
## Use x$model to view the model fit
```

The `method` argument must be one of 'logrank', 'Wilcoxon'[5] or 'Cox'.

In the logrank and Wilcoxon cases the point estimate is for $O - E$ (observed - expected) and the test statistic $Z = \frac{O-E}{\sqrt{V}}$ which is standard normal distribution (Z^2 is the standard χ^2 test statistic). In the Cox case the point estimate is for the log of the hazard ratio.

When fitting models a formula can be included (only the right hand side of the formula is needed). In the example below we fit a Cox model with arm and the 5 covariates:

```
Cox.fits <- ImputeStat(imputed.data.sets,method="Cox",
                       formula=~arm+Z1+Z2+Z3+Z4+Z5)

ExtractSingle(Cox.fits,index=3)$model

## Call:
## model.function(formula = formula, data = object$data, model = TRUE)
##
##          coef exp(coef) se(coef)       z       p
## arm1 -0.7427    0.4758   0.1321  -5.621 1.9e-08
## Z1   -1.8802    0.1526   0.1532 -12.273 < 2e-16
## Z2    0.2562    1.2920   0.2132   1.202   0.229
## Z3   -1.9431    0.1433   0.1523 -12.754 < 2e-16
## Z4    2.0023    7.4061   0.2291   8.741 < 2e-16
## Z5    2.0668    7.8997   0.1526  13.547 < 2e-16
##
## Likelihood ratio test=376.2  on 6 df, p=< 2.2e-16
## n= 400, number of events= 290
```

There are a few rules regarding acceptable formulae: the first term must be the treatment group indicator and there can be no interactions between the treatment group and the other covariates. For the Wilcoxon and logrank tests only `strata` terms can be included alongside the treatment group. For the Cox model: stratified Cox models can be used (e.g. `~arm+strata(Z1,Z3)`), though the `cluster` and `tt` terms cannot be used. Finally if no formula argument is included then a model with only treatment group as a covariate is used. Other arguments to `ImputeStat` are passed into the model fitting function.

The `ImputeStat` function can be parallelized using the same `parallel`, `ncpus` and `cl` arguments as described above for the `ScoreImpute` function.

We can view a matrix of the test statistics:

[5]The Wilcoxon uses the Peto & Peto modification of the Gehan-Wilcoxon test (i.e. `survival::survdiff` with `rho=1`)

```
Cox.fits$statistics

##        estimate        var         Z
## [1,] -0.7286290 0.01709469 -5.572830
## [2,] -0.6503759 0.01684734 -5.010705
## [3,] -0.7427209 0.01745734 -5.621298
## [4,] -0.7402563 0.01717471 -5.648556
## [5,] -0.7681086 0.01746141 -5.812768
## attr(,"class")
## [1] "ScoreStatSet"
```

Each row of the matrix is the point estimate, its variance and the test statistic (`=estimate/sqrt(var)`) from a single imputed data set.

4.4 Calculating Summary Statistics

Summarizing the results averages the test statistics following the two methods described above.

```
final.answer <- summary(Cox.fits)

print(final.answer)

## Summary statistics of Imputed Data Sets
## Method 1 (averaging of point estimates):
## Estimator: -0.7260181
## Var of Estimator: 0.01960166
## Test statistic: 26.8907
## Distribution: F
## with 1, 4 degrees of freedom
## giving a p-value of 0.006580808
##
## Method 2 (averaging of test statistics):
## Test statistic: -5.246857
## Distribution: t
## with 393.4239 degrees of freedom
## giving a p-value of 2.534479e-07

#can access individual elements of the summary
cat("log HR estimate:", final.answer$meth1$estimate)

## log HR estimate: -0.7260181
```

Finally we can view the confidence interval on the log hazard ratio:

```
confint(final.answer, level=0.95)

##       2.5%      97.5%
## -1.0016693 -0.4503669
```

The confidence interval is estimated using Rubin's rules [3] to estimate the standard error and number of degrees of freedom of for the t-distribution.

4.5 Administrative Censoring

By default, both subjects who are administratively and non-administratively censored are deemed to have 'the event of censoring' when calculating the cox regression model to calculate $\hat{RS}_c$. It may be beneficial to

state that only non-administratively censored subjects have 'the event of censoring' when fitting this model. This is possible by defining a censor type column in the data frame containing the values 0, 1 and 2 where 0 = has event, 1=non-administratively censored and 2=administratively censored. Only subjects with a 1 in this column will be considered as having 'the event' of censoring.

Suppose in our toy example, subjects 2 and 3 were administratively censored. First we set up the new column of the data frame:

```
ScoreInd$Ctype <- 1 - ScoreInd$event
ScoreInd$Ctype[ScoreInd$Id %in% c(2,3)] <- 2
```

Next we use the `censor.type` argument when creating the column control object:

```
col.control.a.censor <- col.headings(has.event="event",time="time",Id="Id",
                            arm="arm",DCO.time="DCO.time",
                            to.impute="to.impute",
                            censor.type="Ctype") #Note new arg
```

The rest of the imputation procedure is exactly as before:

```
with.a.censor <- ScoreImpute(data=ScoreInd,m=5,
                event.model="~Z1+Z2+Z3+Z4+Z5,
                censor.model="~Z1+Z3+Z5,
                bootstrap.strata=ScoreInd$arm,
                col.control=col.control.a.censor,
                NN.control=NN.options(NN=5,w.censoring = 0.2))
```

5 Time Dependent Covariates

It is possible to use this method with time dependent covariates. Specifically, for every censored observation, two time independent proportional hazard models are fitted to the data of those at risk at the censoring time using the currently available time dependent variables as fixed covariates [1]. The package can be used with time dependent variables. First we load a data set of time dependent covariates which can be used with the `ScoreInd` data set above.

```
data(ScoreTimeDep)

head(ScoreTimeDep)

##   Id start       end W1       W2
## 1  1   0.0 0.2000000  0 1.325036
## 2  1   0.2 0.4000000  0 3.044134
## 3  1   0.4 0.6000000  1 4.951916
## 4  1   0.6 0.8000000  1 6.993584
## 5  1   0.8 0.8723524  1 9.140960
## 6  2   0.0 0.2000000  0 2.477479
```

The data set has two time dependent covariates `W1` and `W2` and is in panel format; the value of the covariate in `(start,end]` for subject with the given `Id` is given in each row. In order to use this data set within the score imputation method we first use the `MakeTimeDepScore` function with a chosen data frame, giving three additional arguments, the column names of subject Id and the start and end points of the time interval for the panelling.

```
time.dep <- MakeTimeDepScore(ScoreTimeDep,Id="Id",
                             time.start="start",
                             time.end="end")

head(time.dep)

##   Id W1       W2 time.start  time.end
## 1  1  0 1.325036        0.0 0.2000000
## 2  1  0 3.044134        0.2 0.4000000
## 3  1  1 4.951916        0.4 0.6000000
## 4  1  1 6.993584        0.6 0.8000000
## 5  1  1 9.140960        0.8 0.8723524
## 6  2  0 2.477479        0.0 0.2000000
```

Using the `time.dep` argument to `ScoreImpute` we can impute data using the time dependent covariates (note the `min.subjects` argument used to control the minimum number of subjects used when fitting the Cox models) and the rest of the imputation procedure is exactly as for the time independent case:

```
imputed.data.with.td <- ScoreImpute(data=ScoreInd,
                             m=5, bootstrap.strata=ScoreInd$arm,
                             event.model="Z1+ridge(W2,theta=1), #Note the W2 and
                             censor.model="Z2+ridge(W2,theta=1),  #ridge here
                             col.control=col.control,
                             NN.control=NN.options(NN=12,w.censoring = 0.2,
                                                   min.subjects=35), #min.subjects argument
                             time.dep=time.dep) #key argument
```

Note if the `time.dep` argument is used separate models will be fitted for each censored observation as described in the introduction and it is possible the Cox model will fail to converge. Ridge regression (e.g `ridge(W2,theta=1)` in the example above) can be used when fitting the Cox model. See `help(ridge)` for further details.

References

[1] Chiu-Hsieh Hsu and Jeremy MG Taylor. Nonparametric comparison of two survival functions with dependent censoring via nonparametric multiple imputation. *Statistics in Medicine*, 28(3):462–475, 2009.

[2] Kim-Hung Li, Xiao-Li Meng, Trivellore E Raghunathan, and Donald B Rubin. Significance levels from repeated p-values with multiply-imputed data. *Statistica Sinica*, pages 65–92, 1991.

[3] Donald B Rubin. Multiple Imputation for Nonresponse in Surveys (Wiley Series in Probability and Statistics). 1987.