9

Exposed to risk

Course Notes

Syllabus objectives

4.4 Estimate transition intensities dependent on age (exact or census)

- 4.4.3 Specify the data needed for the exact calculation of a central exposed to risk (waiting time) depending on age and sex.
- 4.4.4 Calculate a central exposed to risk given the data in 4.4.3.
- 4.4.5 Explain how to obtain estimates of transition probabilities.
- 4.4.8 Develop census formulae given age at birthday where the age may be classified as next, last, or nearest relative to the birthday as appropriate, and the deaths and census data may use different definitions of age.

0 Introduction

0.1 Contents

There is no material in the Core Reading for this chapter that relates specifically to R. However, in theory, any syllabus objective in Subject CS2 can be tested in the CS2B exam. For example, you could be asked to carry out calculations of the exposed to risk using either the exact method or the census method.

We have provided an exam-style worked example below that covers some types of exercises relating to this topic that could be examined in the practical exam.

0.2 Summary sheets

For the topics in this chapter you will need to be familiar with the basic R commands and functions. We have summarised these in the *R Refresher for CS2 B* document, which is available for you to download from the main menu.

0.3 Data requirements

If you wish to follow our workings for the worked example, you will need to download the following file into your working directory:

- FuneralData.csv

Worked example

Ex 1

Exam style

A life insurer that has been selling single-premium funeral plans to people aged over 50 since 1 January 2010 is analysing its recent mortality experience over the 5-year period from 1 January 2013 to 31 December 2017.

The company has asked you to do an independent check of its calculations of the estimated mortality rate at age 70 last birthday using both the exact method and the census method.

You have been provided with a data file called FuneralData.csv in the following format with entries separated by commas:

LIFE	BIRTH	ENTRY	DEATH
1	1944.089	2014.183	2017.358
2	1946.778	2017.050	2017.683
3	1946.931	2012.344	NA
4	1953.486	2012.322	2016.317
…	…	…	…

Dates are stored in decimal format so that 2018.500, for example, corresponds to 1 July 2018. ENTRY is the date the policy came into force. NA indicates a life that did not die during the investigation period.

(i) (a) Import the data from the file into R, checking that it has the format you are expecting.

(b) Determine the number of records provided. [6]

(ii) (a) Write down the three dates whose maximum determines the date at which exposure at age 70 last birthday begins.

(b) Write down the three dates whose minimum determines the date at which exposure at age 70 last birthday ends.

(c) Hence calculate the exact central exposed to risk at age 70 last birthday for this data set.

(d) Calculate the number of lives who died at age 70 last birthday during the investigation period.

(e) Hence calculate an estimate of the force of mortality at age 70 last birthday, stating the age to which your estimate relates. [30]

(iii) (a) Determine the number of lives aged 70 last birthday whose policies were in force at the start of the investigation period.

(b) Determine the corresponding figures for 1 January in years 2014, 2015, 2016, 2017 and 2018.

(c) Hence estimate the central exposed to risk for age 70 last birthday for the investigation period using a census approach.

(d) Hence calculate an estimate of the force of mortality at age 70 last birthday using the census method. [20]

(iv) Comment on your answers in (ii)(e) and (iii)(d). [4]

[Total 60]

Solution

Ex 1 (i)(a) ***Import the data***

I'm using the c:\Temp folder for my files. So I need to set the working directory (folder):

```
setwd("c:\\Temp")
```

Note that you need to put the folder name in double quotes and (if you're working in Windows) you need to use two backslashes to specify a subfolder.

We can now read in the data file and check that the first few records of the data look as expected:

```
data<-read.table("FuneralData.csv",sep=",",header=T)
head(data)
```

```
  LIFE    BIRTH    ENTRY    DEATH
1    1 1944.089 2014.183 2017.358
2    2 1946.778 2017.050 2017.683
3    3 1946.931 2012.344       NA
4    4 1953.486 2012.322 2016.317
5    5 1967.156 2017.192       NA
6    6 1957.681 2017.025       NA
```

(i)(b) ***Number of records***

There are several ways to find out how many records there are. For example, we could find the life with the highest number:

```
max(data$LIFE)
```

```
[1] 1000
```

Alternatively, we could use the **summary** or **tail** function:

```
summary(data)
```

```
      LIFE            BIRTH          ENTRY          DEATH
 Min.   :   1.0   Min.   :1936   Min.   :2010   Min.   :2011
 1st Qu.: 250.8   1st Qu.:1950   1st Qu.:2012   1st Qu.:2015
 Median : 500.5   Median :1958   Median :2014   Median :2017
 Mean   : 500.5   Mean   :1956   Mean   :2014   Mean   :2016
 3rd Qu.: 750.2   3rd Qu.:1963   3rd Qu.:2016   3rd Qu.:2018
 Max.   :1000.0   Max.   :1968   Max.   :2018   Max.   :2018
                                                NA's   :847
```

```
tail(data)
```

```
     LIFE    BIRTH    ENTRY    DEATH
995   995 1948.925 2016.833       NA
996   996 1965.858 2016.186       NA
997   997 1946.011 2014.536 2017.172
998   998 1962.789 2012.858       NA
999   999 1960.792 2012.297       NA
1000 1000 1948.917 2014.667       NA
```

These all confirm that there are 1,000 records (excluding the header record).

(ii)(a) ***Start date***

The start date for the exposed to risk at age 70 last birthday is the latest of:

- the life's 70th birthday
- the date this life's policy started (the entry date)
- the start of the investigation period.

(ii)(b) ***End date***

As these are single-premium policies (with the premium paid up-front), policies can only be terminated by death. So the end date for the exposed to risk at age 70 last birthday is the earliest of:

- the life's 71st birthday
- the date of death (if applicable)
- the end of the investigation period.

(ii)(c) ***Exact central exposed to risk***

We will present two possible approaches that could be used here.

Method 1 (working with vectors)

With this method we create extra vectors (columns) of dates and do the calculations in parallel.

Start date

To find the 70th birthdays we need to add 70 years to BIRTH:

```
date70=data$BIRTH+70
head(date70)
```

```
[1] 2014.089 2016.778 2016.931 2023.486 2037.156 2027.681
```

We also need to use the start date for the investigation:

```
date0=2013.000
```

We can then apply the 'parallel maximum' function **pmax** to find the latest of the three dates for each life:

```
startdate=pmax(date70,data$ENTRY,date0)
head(startdate)
```

```
[1] 2014.183 2017.050 2016.931 2023.486 2037.156 2027.681
```

End date

To find the 71st birthdays we need to add 71 years to date1:

```
date71=data$BIRTH+71
```

Alternatively, we could add 1 year to `date70`, which we calculated above:

```
date71=date70+1
```

Most of the death dates appear as NA. To make the calculations work correctly for the end dates, we need to replace these with the end date of the investigation period:

```
date99=2017.999 #end date of the investigation

date4=data$DEATH
date4[is.na(date4)]=date99
```

The `is.na` function returns the value `TRUE` if the input equals NA and `FALSE` otherwise.

We can check that this has worked correctly:

```
head(date4)
```

```
[1] 2017.358 2017.683 2017.999 2016.317 2017.999 2017.999
```

We can then apply the 'parallel minimum' function **pmin** to find the earliest of the three dates for each life:

```
enddate=pmin(date71,date4,date99)
head(enddate)
```

```
[1] 2015.089 2017.683 2017.931 2016.317 2017.999 2017.999
```

Exposed to risk

We can see that, for lives who were not exposed to risk at age 70 during the investigation period, the theoretical start date we have calculated may actually be *after* the end date (which would result in an incorrect negative value for the exposed to risk). We can remedy this by replacing the end date with the start date (or *vice versa*) in these cases.

```
enddate2=pmax(enddate,startdate)
```

We can now calculate the exposed to risk by subtracting the start date from the end date:

```
temp=enddate2-startdate
head(temp)
```

```
[1] 0.906 0.633 1.000 0.000 0.000 0.000
```

We now need to sum this over all the lives:

```
(etr70=sum(temp))
```

```
[1] 70.444
```

So the central exposed to risk at age 70 last birthday is 70.444 years.

Method 2 (using a function)

Alternatively, we can set up a function that calculates the exposed to risk for each individual and then apply this function to the whole data table.

We first set up a function that operates on x, the vector of four data values in an individual record.

```
etrfn<-function(x) {
d0=2013.000 #start of investigation;
d1=x[[2]] #date of birth;
d70=d1+70 #70th birthday;
d2=x[[3]] #date of entry;
startdate=max(d70,d2,d0);
d99=2017.999 #end of investigation;
d71=d1+71 #71st birthday;
d3=x[[4]] #date of death;
d4=ifelse(is.na(d3),d99,d3) #Replace NA's with end of investigation
date;
enddate=min(d71,d4,d99);
enddate2=max(enddate,startdate) #Check enddate is not before
startdate;
enddate2-startdate }

etr70=sum(apply(data,1,etrfn))
etr70
```

```
[1] 70.444
```

(ii)(d) ***Number of deaths at age 70 last birthday***

We can extract just the records where we have a date of death for the policyholder:

```
deaths=data$DEATH[!is.na(data$DEATH)]
```

The ! used here is the logical operator for 'not'.

We can examine the first 20 (say):

```
deaths[1:20]
```

```
 [1] 2017.358 2017.683 2016.317 2018.275 2017.158 2018.008 2017.528
2018.183
 [9] 2015.117 2016.692 2017.611 2018.017 2015.744 2017.533 2014.069
2018.406
[17] 2017.953 2012.842 2016.719 2013.639
```

We can see from this list that it includes some lives who died before the investigation period started (*eg* 2012.275) or after it ended (*eg* 2018.008). In order to ensure correspondence with the exposed to risk, we need to exclude these individuals from the death count. So we will need to add some extra restrictions to the subset we have extracted.

```
deaths=data$DEATH[!is.na(data$DEATH)
&data$DEATH>2013.000&data$DEATH<2018.000]
```

The & used here is the logical operator for 'and'.

```
deaths[1:20]
```

```
 [1] 2017.358 2017.683 2016.317 2017.158 2017.528 2015.117 2016.692
2017.611
 [9] 2015.744 2017.533 2014.069 2017.953 2016.719 2013.639 2015.625
2015.731
[17] 2017.950 2017.511 2014.081 2016.717
```

To calculate the ages of these individuals at death, we will also need to extract the corresponding dates of birth:

```
births=data$BIRTH[!is.na(data$DEATH)
&data$DEATH>2013.000&data$DEATH<2018.000]
births[1:20]
```

```
 [1] 1944.089 1946.778 1953.486 1943.169 1959.597 1965.047 1962.533
1949.594
 [9] 1940.144 1958.733 1942.933 1937.214 1947.367 1952.181 1961.314
1941.694
[17] 1944.356 1958.967 1958.989 1964.056
```

We can then calculate the exact age at death for these lives:

```
ages=deaths-births
ages[1:20]
```

```
 [1] 73.269 70.905 62.831 73.989 57.931 50.070 54.159 68.017 75.600
58.800
[11] 71.136 80.739 69.352 61.458 54.311 74.037 73.594 58.544 55.092
52.661
```

We can use the `floor` function to truncate these values to find their ages last birthday:

```
(agelast=sapply(ages,floor))
```

```
 [1] 73 70 62 73 57 50 54 68 75 58 71 80 69 61 54 74 73 58 55 52 52 68 76
76 71
 [26] 69 56 74 56 71 68 80 55 75 77 55 66 53 57 52 50 54 61 52 67 54 72
73 56 52
 [51] 51 59 64 54 79 55 62 59 63 52 72 72 53 51 50 58 64 52 66 57 53 75
70 57 73
 [76] 55 62 66 77 72 64 56 70 54 55 74 63 58 72 68 69 54 55 68 59 71 51
53 55 68
[101] 62 57 66 71 61 51 75 66 64 53 64 53 71 75 58 63 71
```

Finally, we can count how many policyholders died at age 70 last birthday:

```
sum(agelast==70)
```

```
[1] 3
```

Note that to test whether two values are equal in R we need to use “double equals”.

Just to check, we can find these three individuals and look at their full details:

```
lives=data$LIFE[!is.na(data$DEATH)
&data$DEATH>2013.000&data$DEATH<2018.000]
lives[agelast==70]
```

```
[1]   2 588 676
```

```
data[lives[agelast==70],]
```



```
    LIFE    BIRTH    ENTRY    DEATH
2      2 1946.778 2017.050 2017.683
588  588 1944.186 2014.383 2014.575
676  676 1945.086 2014.619 2015.989
```

(ii)(e) ***Estimated force of mortality (exact method)***

We can then calculate an estimate of the force of mortality for age 70 last birthday by dividing the number of deaths by the exposed to risk:

```
      (mu70_exact=3/etr70)
```

```
[1] 0.04258702
```

So the estimated force of mortality for this rate interval is 0.04259.

We are estimating a single value for the force of mortality for a rate interval in which the exact ages of the lives vary from 70 to 71. So our estimate applies to age 70.5.

(iii)(a) ***Number of lives aged 70 last birthday at the start of the investigation period***

For this calculation, it is probably easiest to write a function that we can apply to each record in the data set that will return the value 1 if the life was aged 70 last birthday on that date and their policy was in force at that time, and 0 otherwise.

```
      count70<-function(x) {
      d1=x[[2]] #Date of birth;
      d2=x[[3]] #Date of entry;
      d3=x[[4]] #Date of death;
      age=floor(census-d1) #Age last birthday;
      present=(d2<=census)&((d3>census)|is.na(d3)) #Present on census
      date?;
      ifelse((age==70)&present,1,0) }
```

We can then set the census date to 1 January 2013 and apply the function.

```
      census=2013.000;sum(apply(data,1,count70))
```

```
[1] 12
```

So the number of policies in force at age 70 last birthday on 1 Jan 2013 was 12.

(iii)(b) ***Number of lives aged 70 on the other census dates***

Repeating this for the other census dates gives:

```
      census=2014.000;sum(apply(data,1,count70))
```

```
[1] 9
```

```
      census=2015.000;sum(apply(data,1,count70))
```

```
[1] 18
```

```
      census=2016.000;sum(apply(data,1,count70))
```

```
[1] 14
```

```
      census=2017.000;sum(apply(data,1,count70))
[1] 14
      census=2018.000;sum(apply(data,1,count70))
[1] 19
```

So the numbers of policies in force at age 70 last birthday at the other census dates were:

9, 18, 14, 14 and 19

(iii)(c) ***Estimated central exposed to risk (census method)***

We can then calculate the approximate central exposed to risk using the trapezium rule:

$$E_{70}^{c} = \tfrac{1}{2}(12+9)+\tfrac{1}{2}(9+18)+\tfrac{1}{2}(18+14)+\tfrac{1}{2}(14+14)+\tfrac{1}{2}(14+19)$$
$$= 10.5+13.5+16+14+16.5$$
$$= 70.5$$

As this is a simple calculation, you don't need to use R for this!

(iii)(d) ***Estimated force of mortality (census method)***

We can then calculate an estimate of the force of mortality for age 70 last birthday by dividing the number of deaths by the exposed to risk:

```
      (mu70_census=3/70.5)
[1] 0.04255319
```

So the estimated force of mortality calculated using the census method is 0.04255.

(iv) ***Comment***

The census method assumes that the number of in-force policies varies linearly over each calendar year. Even though there were some quite big changes in these numbers from year to year in this example, the census estimate of the exposed to risk (70.5) was quite close to the exact value (70.444). As a result, the two estimates of the force of mortality were quite similar.

Out of interest, I did a bit more programming to produce a graph showing how the numbers of lives aged 70 last birthday in this data set actually varied during the investigation period, as compared with the linear assumption made in the census method.

To produce the graph on the next page I used the following R code, which is explained in the notes below:

```
times=seq(2013,2018,by=1/360)

graph=NULL;for(t in times)
{census=t;graph=c(graph,sum(apply(data,1,count70)))}

plot(times,graph,type="l",lwd=3,ylim=c(0,20),las=1,
xlab="Date",ylab="Number of people",
main="Number of lives aged 70 last birthday")

lines(seq(2013,2018),c(12,9,18,14,14,19),col="red",lty=2,lwd=2)
```

Notes:

- `times` is a vector consisting of a list of time points corresponding to the individual days during the observation period.
- `graph` is a vector of the number of people aged 70 last birthday at each of these time points. This is initialised to an empty vector (`NULL`), then the count for each date is tagged on.
- `plot` then draws a graph of these numbers. The option `lwd=3` specifies a line width 3 times thicker than normal and `las=1` specifies that the labels on the y-axis should not be rotated.
- `lines` then overlays the dashed (`lty=2`) lines to indicate the linear assumption made in the census method.