Follow-up data with the `Epi` package

Spring 2009.

Michael Hills	Retired Highgate, London
Martyn Plummer	International Agency for Research on Cancer, Lyon `plummer@iarc.fr`

Bendix Carstensen	Steno Diabetes Center, Gentofte, Denmark
	& Department of Biostatistics, University of Copenhagen
	`bxc@steno.dk`
	`www.pubhealth.ku.dk/~bxc`

Contents

1 Follow-up data in the `Epi` package

In the `Epi`-package, follow-up data is represented by adding some extra variables to a dataframe. Such a dataframe is called a `Lexis` object. The tools for handling follow-up data then use the structure of this for special plots, tabulations etc.

Follow-up data basically consists of a time of entry, a time of exit and an indication of the status at exit (normally either "alive" or "dead"). Implicitly is also assumed a status *during* the follow-up (usually "alive").
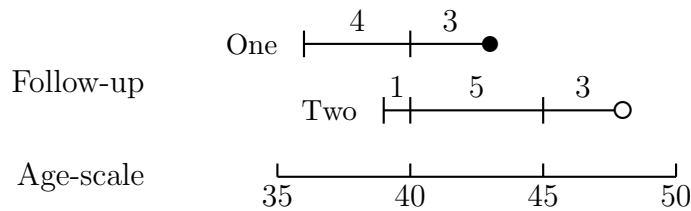

Figure 1: *Follow-up of two persons*

2 Timescales

A timescale is a variable that varies deterministically *within* each person during follow-up, *e.g.*:

- Age

- Calendar time

- Time since treatment

- Time since relapse

All timescales advance at the same pace, so the time followed is the same on all timescales. Therefore, it suffices to use only the entry point on each of the time scale, for example:

- Age at entry.

- Date of entry.

- Time since treatment (*at* treatment this is 0).

- Time since relapse (*at* relapse this is 0)..

In the `Epi` package, follow-up in a cohort is represented in a `Lexis` object. A `Lexis` object is a dataframe with a bit of extra structure representing the follow-up. For the `nickel` data we would construct a `Lexis` object by:

```
> data( nickel )
> nicL <- Lexis( entry = list( per=agein+dob,
+                              age=agein,
+                              tfh=agein-age1st ),
+                exit = list( age=ageout ),
+         exit.status = ( icd %in% c(162,163) )*1,
+                data = nickel )
```

The `entry` argument is a *named* list with the entry points on each of the timescales we want to use. It defines the names of the timescales and the entry points. The `exit` argument gives the exit time on *one* of the timescales, so the name of the element in this list must match one of the neames of the `entry` list. This is sufficient, because the follow-up time on all time scales is the same, in this case `ageout - agein`. Now take a look at the result:

```
> str( nickel )

'data.frame':        679 obs. of  7 variables:
 $ id      : num  3 4 6 8 9 10 15 16 17 18 ...
 $ icd     : num  0 162 163 527 150 163 334 160 420 12 ...
 $ exposure: num  5 5 10 9 0 2 0 0.5 0 0 ...
 $ dob     : num  1889 1886 1881 1886 1880 ...
 $ age1st  : num  17.5 23.2 25.2 24.7 30 ...
 $ agein   : num  45.2 48.3 53 47.9 54.7 ...
 $ ageout  : num  93 63.3 54.2 69.7 76.8 ...

> str( nicL )

Classes 'Lexis' and 'data.frame':        679 obs. of  14 variables:
 $ per     : num  1934 1934 1934 1934 1934 ...
 $ age     : num  45.2 48.3 53 47.9 54.7 ...
 $ tfh     : num  27.7 25.1 27.7 23.2 24.8 ...
 $ lex.dur : num  47.75 15 1.17 21.77 22.1 ...
 $ lex.Cst : num  0 0 0 0 0 0 0 0 0 0 ...
 $ lex.Xst : num  0 1 1 0 0 1 0 0 0 0 ...
 $ lex.id  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ id      : num  3 4 6 8 9 10 15 16 17 18 ...
 $ icd     : num  0 162 163 527 150 163 334 160 420 12 ...
 $ exposure: num  5 5 10 9 0 2 0 0.5 0 0 ...
 $ dob     : num  1889 1886 1881 1886 1880 ...
 $ age1st  : num  17.5 23.2 25.2 24.7 30 ...
 $ agein   : num  45.2 48.3 53 47.9 54.7 ...
 $ ageout  : num  93 63.3 54.2 69.7 76.8 ...
 - attr(*, "time.scales")= chr  "per" "age" "tfh"
 - attr(*, "breaks")=List of 3
  ..$ per: NULL
  ..$ age: NULL
  ..$ tfh: NULL
```

```
> head( nicL )

        per      age       tfh lex.dur lex.Cst lex.Xst lex.id id icd exposure
1 1934.246 45.2273 27.7465 47.7535       0       0      1  3   0        5
2 1934.246 48.2684 25.0820 15.0028       0       1      2  4 162        5
3 1934.246 52.9917 27.7465  1.1727       0       1      3  6 163       10
4 1934.246 47.9067 23.1861 21.7727       0       0      4  8 527        9
5 1934.246 54.7465 24.7890 22.0977       0       0      5  9 150        0
6 1934.246 44.3314 23.0437 18.2099       0       1      6 10 163        2
        dob   age1st    agein   ageout
1 1889.019 17.4808 45.2273 92.9808
2 1885.978 23.1864 48.2684 63.2712
3 1881.255 25.2452 52.9917 54.1644
4 1886.340 24.7206 47.9067 69.6794
5 1879.500 29.9575 54.7465 76.8442
6 1889.915 21.2877 44.3314 62.5413
```

The `Lexis` object `nicL` has a variable for each timescale which is the entry point on this timescale. The follow-up time is in the variable `lex.dur` (**dur**ation).

There is a `summary` function for `Lexis` objects that list the numer of transitions and records as well as the total follow-up time:

```
> summary( nicL )

Transitions:
     To
From   0   1  Records:  Events:  Risk time:  Persons:
   0 542 137       679      137    15348.06       679


Rates:
     To
From 0    1 Total
   0 0 0.01  0.01
```

We defined the exit status to be death from lung cancer (ICD7 162,163), i.e. this variable is 1 if follow-up ended with a death from this cause. If follow-up ended alive or by death from another cause, the exit status is coded 0, i.e. as a censoring.

Note that the exit status is in the variable `lex.Xst` (e**X**it **st**atus. The variable `lex.Cst` is the state where the follow-up takes place (**C**urrent **st**atus), in this case 0 (alive).

It is possible to get a visualization of the follow-up along the timescales chosen by using the `plot` method for `Lexis` objects. `nicL` is an object of *class* `Lexis`, so using the function `plot()` on it means that **R** will look for the function `plot.Lexis` and use this function.

```
> plot( nicL )
```
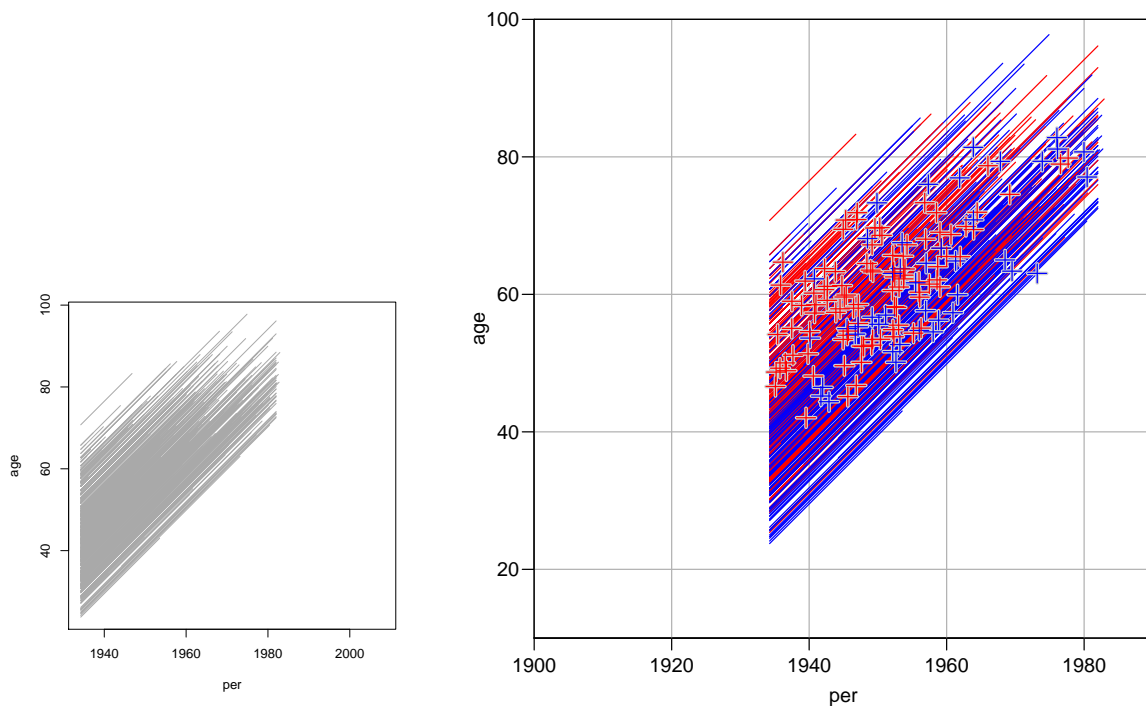
Figure 2: *Lexis diagram of the* `nickel` *dataset, left panel the default version, the right one with bells and whistles. The red lines are for persons with exposure* > 0, *so it is pretty evident that the oldest ones are the exposed part of the cohort.*

The function allows a lot of control over the output, and a `points.Lexis` function allows plotting of the endpoints of follow-up:

```
> par( mar=c(3,3,1,1), mgp=c(3,1,0)/1.6 )
> plot( nicL, 1:2, lwd=1, col=c("blue","red")[(nicL$exp>0)+1],
+        grid=TRUE, lty.grid=1, col.grid=gray(0.7),
+        xlim=1900+c(0,90), xaxs="i",
+        ylim=  10+c(0,90), yaxs="i", las=1 )
> points( nicL, 1:2, pch=c(NA,3)[nicL$lex.Xst+1],
+          col="lightgray", lwd=3, cex=1.5 )
> points( nicL, 1:2, pch=c(NA,3)[nicL$lex.Xst+1],
+          col=c("blue","red")[(nicL$exp>0)+1], lwd=1, cex=1.5 )
```

The results of these two plotting commands are in figure 2.

3   Splitting the follow-up time along a timescale

The follow-up time in a cohort can be subdivided by for example current age. This is achieved by the `splitLexis` (note that it is *not* called `split.Lexis`). This requires that the timescale and the breakpoints on this timescale are supplied. Try:

```
> nicS1 <- splitLexis( nicL, "age", breaks=seq(0,100,10) )
> summary( nicL )
```

```
Transitions:
     To
From   0   1  Records:  Events: Risk time:  Persons:
   0 542 137       679      137   15348.06       679


Rates:
     To
From 0     1 Total
   0 0 0.01  0.01


> summary( nicS1 )

Transitions:
     To
From    0   1  Records:  Events: Risk time:  Persons:
   0 2073 137      2210      137   15348.06       679


Rates:
     To
From 0     1 Total
   0 0 0.01  0.01
```

So we see that the number of events and the amount of follow-up is the same in the two
datasets; only the number of records differ.

To see how records are split for each individual, it is useful to list the results for a few
individuals:

```
> round( subset( nicS1, id %in% 8:10 ), 2 )

   lex.id      per   age   tfh lex.dur lex.Cst lex.Xst id icd exposure      dob
11      4 1934.25 47.91 23.19    2.09       0       0  8 527        9 1886.34
12      4 1936.34 50.00 25.28   10.00       0       0  8 527        9 1886.34
13      4 1946.34 60.00 35.28    9.68       0       0  8 527        9 1886.34
14      5 1934.25 54.75 24.79    5.25       0       0  9 150        0 1879.50
15      5 1939.50 60.00 30.04   10.00       0       0  9 150        0 1879.50
16      5 1949.50 70.00 40.04    6.84       0       0  9 150        0 1879.50
17      6 1934.25 44.33 23.04    5.67       0       0 10 163        2 1889.91
18      6 1939.91 50.00 28.71   10.00       0       0 10 163        2 1889.91
19      6 1949.91 60.00 38.71    2.54       0       1 10 163        2 1889.91
   age1st agein ageout
11  24.72 47.91  69.68
12  24.72 47.91  69.68
13  24.72 47.91  69.68
14  29.96 54.75  76.84
15  29.96 54.75  76.84
16  29.96 54.75  76.84
```

```
17   21.29 44.33   62.54
18   21.29 44.33   62.54
19   21.29 44.33   62.54
```

The resulting object, `nicS1`, is again a `Lexis` object, and so follow-up may be split further along another timescale. Try this and list the results for individuals 8, 9 and 10 again:

```
> nicS2 <- splitLexis( nicS1, "tfh", breaks=c(0,1,5,10,20,30,100) )
> round( subset( nicS2, id %in% 8:10 ), 2 )
```

```
    lex.id      per   age   tfh lex.dur lex.Cst lex.Xst id  icd exposure      dob
13       4 1934.25 47.91 23.19    2.09       0       0  8  527        9 1886.34
14       4 1936.34 50.00 25.28    4.72       0       0  8  527        9 1886.34
15       4 1941.06 54.72 30.00    5.28       0       0  8  527        9 1886.34
16       4 1946.34 60.00 35.28    9.68       0       0  8  527        9 1886.34
17       5 1934.25 54.75 24.79    5.21       0       0  9  150        0 1879.50
18       5 1939.46 59.96 30.00    0.04       0       0  9  150        0 1879.50
19       5 1939.50 60.00 30.04   10.00       0       0  9  150        0 1879.50
20       5 1949.50 70.00 40.04    6.84       0       0  9  150        0 1879.50
21       6 1934.25 44.33 23.04    5.67       0       0 10  163        2 1889.91
22       6 1939.91 50.00 28.71    1.29       0       0 10  163        2 1889.91
23       6 1941.20 51.29 30.00    8.71       0       0 10  163        2 1889.91
24       6 1949.91 60.00 38.71    2.54       0       1 10  163        2 1889.91
   age1st agein ageout
13  24.72 47.91  69.68
14  24.72 47.91  69.68
15  24.72 47.91  69.68
16  24.72 47.91  69.68
17  29.96 54.75  76.84
18  29.96 54.75  76.84
19  29.96 54.75  76.84
20  29.96 54.75  76.84
21  21.29 44.33  62.54
22  21.29 44.33  62.54
23  21.29 44.33  62.54
24  21.29 44.33  62.54
```

If we want to model the effect of these timescales we will for each interval use either the value of the left endpoint in each interval or the middle. There is a function `timeBand` which returns these. Try:

```
> timeBand( nicS2, "age", "middle" )[1:20]
```

```
 [1] 45 45 55 65 75 85 95 45 55 55 65 55 45 55 55 65 55 55 65 75
```

```
> # For nice printing and column labelling use the data.frame() function:
> data.frame( nicS2[,c("id","lex.id","per","age","tfh","lex.dur")],
+             mid.age=timeBand( nicS2, "age", "middle" ),
+             mid.tfh=timeBand( nicS2, "tfh", "middle" ) )[1:20,]
```

	id	lex.id	per	age	tfh	lex.dur	mid.age	mid.tfh
1	3	1	1934.246	45.2273	27.7465	2.2535	45	25
2	3	1	1936.500	47.4808	30.0000	2.5192	45	65
3	3	1	1939.019	50.0000	32.5192	10.0000	55	65
4	3	1	1949.019	60.0000	42.5192	10.0000	65	65
5	3	1	1959.019	70.0000	52.5192	10.0000	75	65
6	3	1	1969.019	80.0000	62.5192	10.0000	85	65
7	3	1	1979.019	90.0000	72.5192	2.9808	95	65
8	4	2	1934.246	48.2684	25.0820	1.7316	45	25
9	4	2	1935.978	50.0000	26.8136	3.1864	55	25
10	4	2	1939.164	53.1864	30.0000	6.8136	55	65
11	4	2	1945.978	60.0000	36.8136	3.2712	65	65
12	6	3	1934.246	52.9917	27.7465	1.1727	55	25
13	8	4	1934.246	47.9067	23.1861	2.0933	45	25
14	8	4	1936.340	50.0000	25.2794	4.7206	55	25
15	8	4	1941.060	54.7206	30.0000	5.2794	55	65
16	8	4	1946.340	60.0000	35.2794	9.6794	65	65
17	9	5	1934.246	54.7465	24.7890	5.2110	55	25
18	9	5	1939.457	59.9575	30.0000	0.0425	55	65
19	9	5	1939.500	60.0000	30.0425	10.0000	65	65
20	9	5	1949.500	70.0000	40.0425	6.8442	75	65

Note that these are the midpoints of the intervals defined by `breaks=`, *not* the
midpoints of the actual follow-up intervals. This is because the variable to be used in
modelling must be independent of the consoring and mortality pattern — it should only
depend on the chosen grouping of the timescale.

4   Splitting time at a specific date

If we have a recording of the date of a specific event as for example recovery or relapse,
we may classify follow-up time as being before of after this intermediate event. This is
achieved with the function `cutLexis`, which takes three arguments: the time point, the
timescale, and the value of the (new) state following the date.

   Now we define the age for the nickel vorkers where the cumulative exposure exceeds
50 exposure years:

```
> subset( nicL, id %in% 8:10 )
```

	per	age	tfh	lex.dur	lex.Cst	lex.Xst	lex.id	id	icd	exposure
4	1934.246	47.9067	23.1861	21.7727	0	0	4	8	527	9

```
5 1934.246 54.7465 24.7890 22.0977          0          0      5  9 150          0
6 1934.246 44.3314 23.0437 18.2099          0          1      6 10 163          2
        dob   age1st    agein   ageout
4 1886.340 24.7206 47.9067 69.6794
5 1879.500 29.9575 54.7465 76.8442
6 1889.915 21.2877 44.3314 62.5413


> agehi <- nicL$age1st + 50 / nicL$exposure
> nicC <- cutLexis( data=nicL, cut=agehi, timescale="age",
+                   new.state=2, precursor.states=0 )
> subset( nicC, id %in% 8:10 )


          per      age      tfh lex.dur lex.Cst lex.Xst lex.id id icd exposure
5     1934.246 54.7465 24.7890 22.0977       0       0      5  9 150          0
6     1934.246 44.3314 23.0437  1.9563       0       2      6 10 163          2
4100  1934.246 47.9067 23.1861 21.7727       2       2      4  8 527          9
680   1936.203 46.2877 25.0000 16.2536       2       1      6 10 163          2
          dob   age1st    agein   ageout
5     1879.500 29.9575 54.7465 76.8442
6     1889.915 21.2877 44.3314 62.5413
4100  1886.340 24.7206 47.9067 69.6794
680   1889.915 21.2877 44.3314 62.5413
```

(The `precursor.states=` argument is explained below). Note that individual 6 has had his follow-up split at age 25 where 50 exposure-years were attained. This could also have been achieved in the split dataset `nicS2` instead of `nicL`, try:

```
> subset( nicS2, id %in% 8:10 )

   lex.id      per      age      tfh lex.dur lex.Cst lex.Xst id icd exposure
13      4 1934.246 47.9067 23.1861  2.0933       0       0  8 527          9
14      4 1936.340 50.0000 25.2794  4.7206       0       0  8 527          9
15      4 1941.060 54.7206 30.0000  5.2794       0       0  8 527          9
16      4 1946.340 60.0000 35.2794  9.6794       0       0  8 527          9
17      5 1934.246 54.7465 24.7890  5.2110       0       0  9 150          0
18      5 1939.457 59.9575 30.0000  0.0425       0       0  9 150          0
19      5 1939.500 60.0000 30.0425 10.0000       0       0  9 150          0
20      5 1949.500 70.0000 40.0425  6.8442       0       0  9 150          0
21      6 1934.246 44.3314 23.0437  5.6686       0       0 10 163          2
22      6 1939.915 50.0000 28.7123  1.2877       0       0 10 163          2
23      6 1941.203 51.2877 30.0000  8.7123       0       0 10 163          2
24      6 1949.915 60.0000 38.7123  2.5413       0       1 10 163          2
        dob   age1st    agein   ageout
13 1886.340 24.7206 47.9067 69.6794
14 1886.340 24.7206 47.9067 69.6794
15 1886.340 24.7206 47.9067 69.6794
```

```
16 1886.340 24.7206 47.9067 69.6794
17 1879.500 29.9575 54.7465 76.8442
18 1879.500 29.9575 54.7465 76.8442
19 1879.500 29.9575 54.7465 76.8442
20 1879.500 29.9575 54.7465 76.8442
21 1889.915 21.2877 44.3314 62.5413
22 1889.915 21.2877 44.3314 62.5413
23 1889.915 21.2877 44.3314 62.5413
24 1889.915 21.2877 44.3314 62.5413


> agehi <- nicS2$age1st + 50 / nicS2$exposure
> nicS2C <- cutLexis( data=nicS2, cut=agehi, timescale="age",
+                     new.state=2, precursor.states=0 )
> subset( nicS2C, id %in% 8:10 )


      lex.id        per       age       tfh lex.dur lex.Cst lex.Xst id icd exposure
17         5 1934.246 54.7465 24.7890  5.2110       0       0  9 150        0
18         5 1939.457 59.9575 30.0000  0.0425       0       0  9 150        0
19         5 1939.500 60.0000 30.0425 10.0000       0       0  9 150        0
20         5 1949.500 70.0000 40.0425  6.8442       0       0  9 150        0
21         6 1934.246 44.3314 23.0437  1.9563       0       2 10 163        2
3142       4 1934.246 47.9067 23.1861  2.0933       2       2  8 527        9
3143       4 1936.340 50.0000 25.2794  4.7206       2       2  8 527        9
3144       4 1941.060 54.7206 30.0000  5.2794       2       2  8 527        9
3145       4 1946.340 60.0000 35.2794  9.6794       2       2  8 527        9
3150       6 1936.203 46.2877 25.0000  3.7123       2       2 10 163        2
3151       6 1939.915 50.0000 28.7123  1.2877       2       2 10 163        2
3152       6 1941.203 51.2877 30.0000  8.7123       2       2 10 163        2
3153       6 1949.915 60.0000 38.7123  2.5413       2       1 10 163        2
          dob age1st   agein  ageout
17   1879.500 29.9575 54.7465 76.8442
18   1879.500 29.9575 54.7465 76.8442
19   1879.500 29.9575 54.7465 76.8442
20   1879.500 29.9575 54.7465 76.8442
21   1889.915 21.2877 44.3314 62.5413
3142 1886.340 24.7206 47.9067 69.6794
3143 1886.340 24.7206 47.9067 69.6794
3144 1886.340 24.7206 47.9067 69.6794
3145 1886.340 24.7206 47.9067 69.6794
3150 1889.915 21.2877 44.3314 62.5413
3151 1889.915 21.2877 44.3314 62.5413
3152 1889.915 21.2877 44.3314 62.5413
3153 1889.915 21.2877 44.3314 62.5413
```

Note that follow-up subsequent to the event is classified as being in state 2, but that the final transition to state 1 (death from lung cancer) is preserved. This is the point of the

`precursor.states=` argument. It names the states (in this case 0, "Alive") that will be over-witten by `new.state` (in this case state 2, "High exposure"). Clearly, state 1 ("Dead") should not be updated even if it is after the time where the persons moves to state 2. In other words, only state 0 is a precursor to state 2, state 1 is always subsequent to state 2.

Note if the intermediate event is to be used as a time-dependent variable in a Cox-model, then `lex.Cst` should be used as the time-dependent variable, and `lex.Xst==1` as the event.

5   Competing risks — multiple types of events

If we want to consider death from lung cancer and death from other causes as separate events we can code these as for example 1 and 2.

```
> data( nickel )
> nicL <- Lexis( entry = list( per=agein+dob,
+                               age=agein,
+                               tfh=agein-age1st ),
+                 exit = list( age=ageout ),
+          exit.status = ( icd > 0 ) + ( icd %in% c(162,163) ),
+                 data = nickel )
> summary( nicL )

Transitions:
     To
From  0   1    2  Records:  Events:  Risk time:  Persons:
   0 47 495 137       679       632    15348.06        679

Rates:
     To
From 0    1     2 Total
   0 0 0.03 0.01  0.04

> subset( nicL, id %in% 8:10 )

        per     age     tfh lex.dur lex.Cst lex.Xst lex.id id icd exposure
4 1934.246 47.9067 23.1861 21.7727       0       1      4  8 527        9
5 1934.246 54.7465 24.7890 22.0977       0       1      5  9 150        0
6 1934.246 44.3314 23.0437 18.2099       0       2      6 10 163        2
       dob   age1st   agein  ageout
4 1886.340 24.7206 47.9067 69.6794
5 1879.500 29.9575 54.7465 76.8442
6 1889.915 21.2877 44.3314 62.5413
```

If we want to label the states, we can enter the names of these in the `states` parameter, try for example:

```
> nicL <- Lexis( entry = list( per=agein+dob,
+                              age=agein,
+                              tfh=agein-age1st ),
+               exit = list( age=ageout ),
+         exit.status = ( icd > 0 ) + ( icd %in% c(162,163) ),
+                data = nickel,
+              states = c("Alive","D.oth","D.lung") )
> summary( nicL )
```

```
Transitions:
     To
From     Alive D.oth D.lung  Records:  Events: Risk time:  Persons:
  Alive     47   137    495       679      632   15348.06       679
```

```
Rates:
     To
From     Alive D.oth D.lung Total
  Alive      0  0.01   0.03  0.04
```

Note that the Lexis function automatically assumes that all persons enter in the first level (given in the states= argument)

When we cut at a date as in this case, the date where cumulative exposure exceeds 50 exposure-years, we get the follow-up *after* the date classified as being in the new state if the exit (lex.Xst) was to a state we defined as one of the precursor.states:

```
> nicL$agehi <- nicL$age1st + 50 / nicL$exposure
> nicC <- cutLexis( data = nicL,
+                    cut = nicL$agehi,
+              timescale = "age",
+              new.state = "HiExp",
+        precursor.states = "Alive" )
> subset( nicC, id %in% 8:10 )
```

```
           per      age      tfh lex.dur lex.Cst lex.Xst lex.id id icd exposure
5     1934.246 54.7465 24.7890 22.0977   Alive  D.lung      5  9 150        0
6     1934.246 44.3314 23.0437  1.9563   Alive   HiExp      6 10 163        2
4100  1934.246 47.9067 23.1861 21.7727   HiExp  D.lung      4  8 527        9
680   1936.203 46.2877 25.0000 16.2536   HiExp   D.oth      6 10 163        2
           dob  age1st   agein  ageout    agehi
5     1879.500 29.9575 54.7465 76.8442      Inf
6     1889.915 21.2877 44.3314 62.5413 46.28770
4100  1886.340 24.7206 47.9067 69.6794 30.27616
680   1889.915 21.2877 44.3314 62.5413 46.28770
```

```
> summary( nicC, scale=1000 )
```

```
Transitions:
     To
From     Alive HiExp D.oth D.lung  Records:  Events: Risk time:  Persons:
   Alive    39    83    65    279       466      427     10.77       466
   HiExp     0     8    72    216       296      288      4.58       296
   Sum      39    91   137    495       762      715     15.35       679


Rates (per 1000):
     To
From     Alive HiExp D.oth D.lung Total
   Alive     0   7.7  6.03  25.90 39.64
   HiExp     0   0.0 15.74  47.21 62.94
```

Note that the persons-years is the same, but that the number of events has changed. This is because events are now defined as any transition from alive, including the transitions to `HiExp`.

Also note that (so far) it is necessary to specify the variable with the cutpoints in full, using only `cut=agehi` would give an error.

Subdivision of existing states

It may be of interest to subdivide the states following the intermediate event according to wheter the event has occurred or not. That is done by the argument `split.states=TRUE`.

Moreover, it will also often be of interest to introduce a new timescale indicating the time since intermediate event. This can be done by the argument `new.scale=TRUE`, alternatively `new.scale="tfevent"`, as illustrated here:

```
> nicC <- cutLexis( data = nicL,
+                    cut = nicL$agehi,
+               timescale = "age",
+               new.state = "Hi",
+            split.states=TRUE, new.scale=TRUE,
+         precursor.states = "Alive" )
> subset( nicC, id %in% 8:10 )


           per      age      tfh   Hi.dur lex.dur lex.Cst    lex.Xst lex.id id icd
4100  1934.246 47.9067 23.1861 17.63054 21.7727      Hi D.lung(Hi)      4  8 527
5     1934.246 54.7465 24.7890       NA 22.0977   Alive     D.lung      5  9 150
6     1934.246 44.3314 23.0437       NA  1.9563   Alive         Hi      6 10 163
680   1936.203 46.2877 25.0000  0.00000 16.2536      Hi   D.oth(Hi)     6 10 163
      exposure      dob  age1st   agein  ageout    agehi
4100         9 1886.340 24.7206 47.9067 69.6794 30.27616
5            0 1879.500 29.9575 54.7465 76.8442      Inf
6            2 1889.915 21.2877 44.3314 62.5413 46.28770
680          2 1889.915 21.2877 44.3314 62.5413 46.28770
```

```
> summary( nicC, scale=1000 )

Transitions:
     To
From    Alive Hi D.oth D.lung D.oth(Hi) D.lung(Hi)  Records:  Events:
  Alive    39 83    65    279         0          0       466      427
  Hi        0  8     0      0        72        216       296      288
  Sum      39 91    65    279        72        216       762      715

Transitions:
     To
From    Risk time:  Persons:
  Alive      10.77       466
  Hi          4.58       296
  Sum        15.35       679

Rates (per 1000):
     To
From    Alive  Hi D.oth D.lung D.oth(Hi) D.lung(Hi) Total
  Alive     0 7.7  6.03   25.9      0.00       0.00 39.64
  Hi        0 0.0  0.00    0.0     15.74      47.21 62.94
```

6 Multiple events of the same type (recurrent events)

Sometimes more events of the same type are recorded for each person and one would then like to count these and put follow-up time in states accordingly. Essentially, each set of cutpoints represents progressions from one state to the next. Therefore the states should be numbered, and the numbering of states subsequently occupied be increased accordingly.

This is a behaviour different from the one outlined above, and it is achieved by the argument `count=TRUE` to `cutLexis`. When `count` is set to `TRUE`, the value of the arguments `new.state` and `precursor.states` are ignored. Actually, when using the argument `count=TRUE`, the function `countLexis` is called, so an alternative is to use this directly.