Time dependent covariates in `Lexis` objects

SDCC
April 2021
http://bendixcarstensen.com/Epi
Version 2

Compiled Wednesday 14th April, 2021, 14:04
from: `/home/bendix/stat/R/lib.src/Epi/pkg/vignettes/addLexis.tex`

Bendix Carstensen Steno Diabetes Center Copenhagen, Gentofte, Denmark
& Department of Biostatistics, University of Copenhagen
b@bxc.dk
http://BendixCarstensen.com

Contents

Chapter 1

Overview and rationale

This note describes the functions `addCov.Lexis` and `addDrug.Lexis` designed to add values of clinical measurements and drug exposures to time-split `Lexis` objects. If time-dependent variables are binary, such as for example "occurrence of CVD diagnosis" it may be relevant to define a new state as, say, `CVD`. But the purposes of the two functions here are to append quantitative variables that in principle can take any (positive) value. Adding new states at intermediate events is the business of `cutLexis`.

Both functions are so-called `S3` methods for `Lexis` objects, so in code you can omit the "`.Lexis`". Note that neither `splitLexis` or `splitMulti` are `S3` methods—there is no "`.`" in the names.

1.1 addCov.Lexis

The rationale behind `addCov.Lexis` is to provide the ability to amend a `Lexis` object with clinical measurements taken at different times, and propagate the values as LOCF (Last Observation Carried Forward). This means that time-splitting of a `Lexis` object *after* adding clinical measurements will be meaningful, because both `splitLexis` and `splitMulti` will carry variables forward to the split records. The follow-up in the resulting `Lexis` object will be cut at dates of clinical measurement.

1.2 addDrug.Lexis

As opposed to this, the rationale behind `addDrug.Lexis` is to add drug information at each date of drug purchase, *and* subsequently compute cumulative exposure measures at the times in the resulting `Lexis` object. Therefore, it will not be meaningful to further split an object resulting from `addDrug.Lexis`—LOCF is not meaningful for continuously time-varying covariates such as cumulative exposure.

If persons present with very frequent drug purchases, the intervals may become very small and the sheer number of records may present an impediment to analysis. Therefore the function `coarse.Lexis` is provided to collapse adjacent follow-up records.

Chapter 2

addCov.Lexis

2.1 Rationale

The function has arisen out of a need to attach values measured at clinical visits to a Lexis object representing follow-up for events constituting a multistate model. Hence the data with measurements at clinical visits will be called `clin` for mnemonic reasons.

2.2 Example

For illustration we devise a small bogus cohort of 3 people, where we convert the character dates into numerical variables (fractional years) using `cal.yr`. Note that we are using a character variable ad `id`:

```
> xcoh <- structure(list(id = c("A", "B", "C"),
+                       birth = c("1952-07-14", "1954-04-01", "1987-06-10"),
+                       entry = c("1965-08-04", "1972-09-08", "1991-12-23"),
+                        exit = c("1997-06-27", "1995-05-23", "1998-07-24"),
+                        fail = c(1, 0, 1) ),
+                      .Names = c("id", "birth", "entry", "exit", "fail"),
+                   row.names = c("1", "2", "3"),
+                       class = "data.frame" )
> xcoh$dob <- cal.yr(xcoh$birth)
> xcoh$doe <- cal.yr(xcoh$entry)
> xcoh$dox <- cal.yr(xcoh$exit )
> xcoh
  id      birth      entry       exit fail      dob      doe      dox
1  A 1952-07-14 1965-08-04 1997-06-27    1 1952.533 1965.589 1997.485
2  B 1954-04-01 1972-09-08 1995-05-23    0 1954.246 1972.686 1995.388
3  C 1987-06-10 1991-12-23 1998-07-24    1 1987.437 1991.974 1998.559
```

2.2.1 A `Lexis` object

Define this as a `Lexis` object with timescales calendar time (`per`, period) and age (`age`):

```
> Lcoh <- Lexis(entry = list(per = doe),
+                exit = list(per = dox,
+                            age = dox - dob),
+                  id = id,
```

```
+         exit.status = factor(fail, 0:1, c("Alive","Dead")),
+                data = xcoh)
NOTE: entry.status has been set to "Alive" for all.
> str(Lcoh)
Classes 'Lexis' and 'data.frame':       3 obs. of  14 variables:
 $ per    : 'cal.yr' num  1966 1973 1992
 $ age    : 'cal.yr' num  13.06 18.44 4.54
 $ lex.dur: 'cal.yr' num  31.9 22.7 6.58
 $ lex.Cst: Factor w/ 2 levels "Alive","Dead": 1 1 1
 $ lex.Xst: Factor w/ 2 levels "Alive","Dead": 2 1 2
 $ lex.id : Factor w/ 3 levels "A","B","C": 1 2 3
 $ id     : chr  "A" "B" "C"
 $ birth  : chr  "1952-07-14" "1954-04-01" "1987-06-10"
 $ entry  : chr  "1965-08-04" "1972-09-08" "1991-12-23"
 $ exit   : chr  "1997-06-27" "1995-05-23" "1998-07-24"
 $ fail   : num  1 0 1
 $ dob    : 'cal.yr' num  1953 1954 1987
 $ doe    : 'cal.yr' num  1966 1973 1992
 $ dox    : 'cal.yr' num  1997 1995 1999
 - attr(*, "time.scales")= chr  "per" "age"
 - attr(*, "time.since")= chr  "" ""
 - attr(*, "breaks")=List of 2
  ..$ per: NULL
  ..$ age: NULL
> (Lx <- Lcoh[,1:6])
       per       age   lex.dur lex.Cst lex.Xst lex.id
1 1965.589 13.056810 31.895962   Alive    Dead      A
2 1972.686 18.439425 22.702259   Alive   Alive      B
3 1991.974  4.536619  6.584531   Alive    Dead      C
```

Factor or character `lex.id`?

Note that when the `id` argument to `Lexis` is a character variable then the `lex.id` will be a factor. Which, if each person has a lot of records may save time, but if you subset may be a waste of space. Moreover merging (*i.e.* joining in the language of `tidyverse`) may present problems with different levels. `merge` from the `base` R, will coerce to factor with union of levels as levels, where as `join` will coerce to character.

Thus the most reasonable strategy thus seems to keep `lex.id` as a character variable.

```
> Lx$lex.id <- as.character(Lx$lex.id)
> Lx
       per       age   lex.dur lex.Cst lex.Xst lex.id
1 1965.589 13.056810 31.895962   Alive    Dead      A
2 1972.686 18.439425 22.702259   Alive   Alive      B
3 1991.974  4.536619  6.584531   Alive    Dead      C
> str(Lx)
Classes 'Lexis' and 'data.frame':       3 obs. of  6 variables:
 $ per    : 'cal.yr' num  1966 1973 1992
 $ age    : 'cal.yr' num  13.06 18.44 4.54
 $ lex.dur: 'cal.yr' num  31.9 22.7 6.58
 $ lex.Cst: Factor w/ 2 levels "Alive","Dead": 1 1 1
 $ lex.Xst: Factor w/ 2 levels "Alive","Dead": 2 1 2
```

```
 $ lex.id : chr  "A" "B" "C"
 - attr(*, "time.scales")= chr  "per" "age"
 - attr(*, "time.since")= chr  "" ""
 - attr(*, "breaks")=List of 2
  ..$ per: NULL
  ..$ age: NULL
```

Clinical measurements

Then we generate data frame with clinical examination data, that is date of examination in `per`, some bogus clinical measurements and also names of the examination rounds:

```
> clin <- data.frame(lex.id = c("A", "A", "C", "B", "C"),
+                        per = c(1977.3, 1971.7, 1996.2, 1990.6, 1989.2),
+                         bp = c(120, 140, 160, 157, 145),
+                       chol = c(5, 7, 8, 9, 6),
+                       xnam = c("X2", "X1", "X1", "X2", "X0"),
+          stringsAsFactors = FALSE)
> clin
  lex.id    per  bp chol xnam
1      A 1977.3 120    5   X2
2      A 1971.7 140    7   X1
3      C 1996.2 160    8   X1
4      B 1990.6 157    9   X2
5      C 1989.2 145    6   X0
```

```
> str(clin)
```

```
'data.frame':        5 obs. of  5 variables:
 $ lex.id: chr  "A" "A" "C" "B" ...
 $ per   : num  1977 1972 1996 1991 1989
 $ bp    : num  120 140 160 157 145
 $ chol  : num  5 7 8 9 6
 $ xnam  : chr  "X2" "X1" "X1" "X2" ...
```

Note that we have chosen a measurement for person `C` from 1989—before the person's entry to the study.

2.2.2 Adding clinical data

There is a slightly different behaviour according to whether the variable with the name of the examination is given or not, and whether the name of the (incomplete) time scale is given or not:

```
> (Cx <- addCov.Lexis(Lx, clin))
       per       age   lex.dur lex.id lex.Cst lex.Xst exnam     tfc  bp chol xnam
1 1965.589 13.056810  6.110678      A   Alive   Alive  <NA>      NA  NA   NA <NA>
2 1971.700 19.167488  5.600000      A   Alive   Alive   ex1 0.00000 140    7   X1
3 1977.300 24.767488 20.185284      A   Alive    Dead   ex2 0.00000 120    5   X2
4 1972.686 18.439425 17.914168      B   Alive   Alive  <NA>      NA  NA   NA <NA>
5 1990.600 36.353593  4.788090      B   Alive   Alive   ex1 0.00000 157    9   X2
6 1991.974  4.536619  4.226010      C   Alive   Alive   ex1 2.77399 145    6   X0
7 1996.200  8.762628  2.358522      C   Alive    Dead   ex2 0.00000 160    8   X1
```

Note that the clinical measurement preceding the entry of person C is included, and that the `tfc` (time from clinical measurement) is correctly rendered, we a non-zero value at date of entry.

We also see that a variable `exnam` is constructed with consecutive numbering of examinations within each person.

If we explicitly give the name of the variable holding the examination names we get this in the results. We can also determine the name of the (incomplete) timescale holding the time since measurement.

```
> (Dx <- addCov.Lexis(Lx, clin, exnam = "xnam", tfc = "TfCl"))
      per       age   lex.dur lex.id lex.Cst lex.Xst xnam    TfCl  bp chol
1 1965.589 13.056810  6.110678      A   Alive   Alive <NA>      NA  NA   NA
2 1971.700 19.167488  5.600000      A   Alive   Alive   X1 0.00000 140    7
3 1977.300 24.767488 20.185284      A   Alive    Dead   X2 0.00000 120    5
4 1972.686 18.439425 17.914168      B   Alive   Alive <NA>      NA  NA   NA
5 1990.600 36.353593  4.788090      B   Alive   Alive   X2 0.00000 157    9
6 1991.974  4.536619  4.226010      C   Alive   Alive   X0 2.77399 145    6
7 1996.200  8.762628  2.358522      C   Alive    Dead   X1 0.00000 160    8
> summary(Dx, t=T)
Transitions:
     To
From    Alive Dead  Records:  Events: Risk time:  Persons:
  Alive     5    2         7        2      61.18         3

Timescales:
 per  age TfCl
  ""   ""   ""
```

2.3 Exchanging split and add

As noted in the beginning of this not `addCov.Lexis` uses LOCF, and so it is commutative with `splitLexis`:

```
> # split BEFORE add
> Lb <- addCov.Lexis(splitLexis(Lx,
+                       time.scale = "age",
+                           breaks = seq(0, 80, 5)),
+                    clin,
+                    exnam = "xnam" )
> Lb
   lex.id      per       age   lex.dur lex.Cst lex.Xst xnam       tfc  bp chol
1       A 1965.589 13.056810 1.9431896   Alive   Alive <NA>        NA  NA   NA
2       A 1967.533 15.000000 4.1674880   Alive   Alive <NA>        NA  NA   NA
3       A 1971.700 19.167488 0.8325120   Alive   Alive   X1  0.000000 140    7
4       A 1972.533 20.000000 4.7674880   Alive   Alive   X1  0.832512 140    7
5       A 1977.300 24.767488 0.2325120   Alive   Alive   X2  0.000000 120    5
6       A 1977.533 25.000000 5.0000000   Alive   Alive   X2  0.232512 120    5
7       A 1982.533 30.000000 5.0000000   Alive   Alive   X2  5.232512 120    5
8       A 1987.533 35.000000 5.0000000   Alive   Alive   X2 10.232512 120    5
9       A 1992.533 40.000000 4.9527721   Alive    Dead   X2 15.232512 120    5
10      B 1972.686 18.439425 1.5605749   Alive   Alive <NA>        NA  NA   NA
11      B 1974.246 20.000000 5.0000000   Alive   Alive <NA>        NA  NA   NA
```

```
12      B 1979.246 25.000000 5.0000000   Alive   Alive <NA>        NA  NA   NA
13      B 1984.246 30.000000 5.0000000   Alive   Alive <NA>        NA  NA   NA
14      B 1989.246 35.000000 1.3535934   Alive   Alive <NA>        NA  NA   NA
15      B 1990.600 36.353593 3.6464066   Alive   Alive   X2  0.000000 157    9
16      B 1994.246 40.000000 1.1416838   Alive   Alive   X2  3.646407 157    9
17      C 1991.974  4.536619 0.4633812   Alive   Alive   X0  2.773990 145    6
18      C 1992.437  5.000000 3.7626283   Alive   Alive   X0  3.237372 145    6
19      C 1996.200  8.762628 1.2373717   Alive   Alive   X1  0.000000 160    8
20      C 1997.437 10.000000 1.1211499   Alive    Dead   X1  1.237372 160    8
> #
> # split AFTER add
> La <- splitLexis(addCov.Lexis(Lx,
+                               clin,
+                              exnam = "xnam" ),
+                  time.scale = "age",
+                      breaks = seq(0, 80, 5))
```

We see that the results are identical bar the sequence of variables and attributes.

We can more explicitly verify that the resulting data frames are the same:

```
> La$tfc == Lb$tfc
 [1]    NA    NA TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE    NA    NA    NA    NA    NA TRUE TRUE TRUE T
[19] TRUE TRUE
> La$age == Lb$age
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE T
[19] TRUE TRUE
> La$per == Lb$per
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE T
[19] TRUE TRUE
```

The same goes for `splitMulti`:

```
> # split BEFORE add
> Mb <- addCov.Lexis(splitMulti(Lx, age = seq(0, 80, 5)),
+                    clin,
+                    exnam = "xnam" )
> #
> # split AFTER add
> Ma <- splitMulti(addCov.Lexis(Lx,
+                               clin,
+                              exnam = "xnam" ),
+                  age = seq(0, 80, 5))
> La$tfc == Mb$tfc
 [1]    NA    NA TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE    NA    NA    NA    NA    NA TRUE TRUE TRUE T
[19] TRUE TRUE
> Ma$tfc == Mb$tfc
 [1]    NA    NA TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE    NA    NA    NA    NA    NA TRUE TRUE TRUE T
[19] TRUE TRUE
```

In summary, because both `addCov.Lexis` and `splitLexis`/`splitMulti` use LOCF for covariates the order of splitting and adding does not matter.

This is certainly not the case with addDrug.Lexis as we shall see.

Chapter 3

addDrug.Lexis

The general purpose of the function is to amend a `Lexis` object with drug exposure data. The data base with information on a specific drug is assumed to be a data frame with one entry per drug purchase, containing the date and the amount purchased and optionally the prescribed dosage (that is how much is supposed to be taken per time). We assume that we have such a data base for each drug of interest, of course also including an id variable that matches the `Lex.id` variable in the `Lexis` object.

For each type of drug the function derives 4 variables:

`ex` : logical, is the person currently `ex`posed
`tf` : numeric, `t`ime since `f`irst purchase
`ct` : numeric, `c`umulative `t`ime on the drug
`cd` : numeric, `c`umulative `d`ose of the drug

These names are pre- or suf-fixed by the drug name, so that exposures to different drugs can be distinguished; see the examples.

The resulting `Lexis` object has extra records corresponding to cuts at each drug purchase and at each expiry date of a purchase.

Specifically, for each purchase the coverage period is derived (different methods for this are available), and if the end of this (the expiry date) is earlier than the next purchase of the person, the person is considered off the drug from the expiry date, and a cut in the follow-up is generated with `ex` set to `FALSE`.

3.1 The help example

The following is a slight modification of the code from the example section of the help page for `addDrug.Lexis`

First we generate follow-up of 2 persons, and split the follow-up in intervals of length 0.6 years along the calendar time scale, `per`:

```
> fu <- data.frame(doe = c(2006, 2008),
+                  dox = c(2015, 2018),
+                  dob = c(1950, 1951),
+                  xst = factor(c("A","D")))
> Lx <- Lexis(entry = list(per = doe,
+                          age = doe- dob),
+             exit = list(per = dox),
```

```
+       exit.status = xst,
+              data = fu)
NOTE: entry.status has been set to "A" for all.
> Lx <- subset(Lx, select = -c(doe, dob, dox, xst))
> Sx <- splitLexis(Lx, "per", breaks = seq(1990, 2020, 0.6))
> summary(Sx)
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 32 1        33        1         19         2
```

Then we generate drug purchases for these two persons, one data frame for each of the drugs `F` and `G`. Note that we generate `lex.id`$\in (1, 2)$ referring to the values of `lex.id` in the lexis object `Sx`.

```
> set.seed(1952)
> rf <- data.frame(per = c(2005 + runif(12, 0, 10)),
+                  amt = sample(2:4, 12, replace = TRUE),
+               lex.id = sample(1:2, 12, replace = TRUE)) %>%
+       arrange(lex.id, per)
> rg <- data.frame(per = c(2009 + runif(10, 0, 10)),
+                  amt = sample(round(2:4/3,1), 10, replace = TRUE),
+               lex.id = sample(1:2, 10, replace = TRUE)) %>%
+       arrange(lex.id, per)
```

Strictly speaking we do not need to sort the drug purchase data frames, but it makes it easier to grasp the structure.

The way purchase data is supplied to the function is in a `list` where each element is a data frame of purchase records for one type of drug. The list must be named, the names will be used as prefixes of the generated exposure variables. We can show the resulting data:

```
> pdat <- list(F = rf, G = rg)
> pdat
$F
        per amt lex.id
1  2013.964   4      1
2  2014.251   4      1
3  2014.509   3      1
4  2014.990   2      1
5  2005.311   2      2
6  2007.595   3      2
7  2011.710   4      2
8  2011.812   3      2
9  2012.865   2      2
10 2013.331   2      2
11 2013.417   2      2
12 2013.932   2      2

$G
        per amt lex.id
1  2009.630 1.3      1
2  2012.987 1.3      1
3  2013.018 1.3      1
4  2016.954 0.7      1
```

```
5  2017.924 1.0      1
6  2009.089 1.3      2
7  2011.599 0.7      2
8  2014.566 1.0      2
9  2016.912 0.7      2
10 2017.004 1.0      2
```

```
> Lx
```

```
   per age lex.dur lex.Cst lex.Xst lex.id
1 2006  56       9       A       A      1
2 2008  57      10       A       D      2
```

Note that we have generated data so that there are drug purchases of drug `F` that is *before* start of follow-up for person 2.

We can then expand the time-split `Lexis` object, `Sx` with the drug information. Note that we not only add 8 variables (4 from each drug), we also add records representing the purchase dates and possible expiry dates.

Here is a small utility to compactify the printing of data frames

```
> prdf <-
+ function(df)
+ {
+ whn <- sapply(df, is.numeric)
+ df[, whn] <- round(df[, whn], 2)
+ print(df, row.names=FALSE)
+ }
```

```
> summary(Sx) ; names(Sx)
```

```
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 32 1        33        1         19         2
```

```
[1] "lex.id"  "per"     "age"     "lex.dur" "lex.Cst" "lex.Xst"
```

```
> ex1 <- addDrug.Lexis(Sx, pdat, method = "ext") # default
```

```
NOTE: timescale taken as 'per'
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
```

```
> summary(ex1) ; names(ex1)
```

```
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 58 1        59        1         19         2
```

```
 [1] "lex.id"  "per"     "age"     "lex.dur" "lex.Cst" "lex.Xst" "F.ex"    "F.tf"    "F.ct"
[10] "F.cd"    "G.ex"    "G.tf"    "G.ct"    "G.cd"
```

```
> prdf(ex1)
```

```
 lex.id     per   age lex.dur lex.Cst lex.Xst  F.ex F.tf F.ct  F.cd  G.ex G.tf G.ct G.cd
      1 2006.00 56.00    0.20       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2006.20 56.20    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2006.80 56.80    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2007.40 57.40    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2008.00 58.00    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2008.60 58.60    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2009.20 59.20    0.43       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
```

```
1 2009.63 59.63    0.17       A       A FALSE 0.00 0.00  0.00  TRUE 0.00 0.00 0.00
1 2009.80 59.80    0.60       A       A FALSE 0.00 0.00  0.00  TRUE 0.17 0.17 0.07
1 2010.40 60.40    0.60       A       A FALSE 0.00 0.00  0.00  TRUE 0.77 0.77 0.30
1 2011.00 61.00    0.60       A       A FALSE 0.00 0.00  0.00  TRUE 1.37 1.37 0.53
1 2011.60 61.60    0.60       A       A FALSE 0.00 0.00  0.00  TRUE 1.97 1.97 0.76
1 2012.20 62.20    0.60       A       A FALSE 0.00 0.00  0.00  TRUE 2.57 2.57 1.00
1 2012.80 62.80    0.19       A       A FALSE 0.00 0.00  0.00  TRUE 3.17 3.17 1.23
1 2012.99 62.99    0.03       A       A FALSE 0.00 0.00  0.00  TRUE 3.36 3.36 1.30
1 2013.02 63.02    0.03       A       A FALSE 0.00 0.00  0.00  TRUE 3.39 3.39 2.60
1 2013.05 63.05    0.35       A       A FALSE 0.00 0.00  0.00 FALSE 3.42 3.42 3.90
1 2013.40 63.40    0.56       A       A FALSE 0.00 0.00  0.00 FALSE 3.77 3.42 3.90
1 2013.96 63.96    0.04       A       A  TRUE 0.00 0.00  0.00 FALSE 4.33 3.42 3.90
1 2014.00 64.00    0.25       A       A  TRUE 0.04 0.04  0.50 FALSE 4.37 3.42 3.90
1 2014.25 64.25    0.26       A       A  TRUE 0.29 0.29  4.00 FALSE 4.62 3.42 3.90
1 2014.51 64.51    0.09       A       A  TRUE 0.54 0.54  8.00 FALSE 4.88 3.42 3.90
1 2014.60 64.60    0.10       A       A  TRUE 0.64 0.64  9.41 FALSE 4.97 3.42 3.90
1 2014.70 64.70    0.29       A       A FALSE 0.74 0.74 11.00 FALSE 5.07 3.42 3.90
1 2014.99 64.99    0.01       A       A  TRUE 1.03 0.74 11.00 FALSE 5.36 3.42 3.90
2 2008.00 57.00    0.60       A       A  TRUE 0.00 0.40  0.35 FALSE 0.00 0.00 0.00
2 2008.60 57.60    0.49       A       A  TRUE 0.60 1.00  0.88 FALSE 0.00 0.00 0.00
2 2009.09 58.09    0.11       A       A  TRUE 1.09 1.49  1.31  TRUE 0.00 0.00 0.00
2 2009.20 58.20    0.60       A       A  TRUE 1.20 1.60  1.40  TRUE 0.11 0.11 0.06
2 2009.80 58.80    0.60       A       A  TRUE 1.80 2.20  1.93  TRUE 0.71 0.71 0.37
2 2010.40 59.40    0.60       A       A  TRUE 2.40 2.80  2.46  TRUE 1.31 1.31 0.68
2 2011.00 60.00    0.02       A       A  TRUE 3.00 3.40  2.98  TRUE 1.91 1.91 0.99
2 2011.02 60.02    0.58       A       A FALSE 3.02 3.43  3.00  TRUE 1.93 1.93 1.00
2 2011.60 60.60    0.00       A       A FALSE 3.60 3.43  3.00  TRUE 2.51 2.51 1.30
2 2011.60 60.60    0.11       A       A FALSE 3.60 3.43  3.00  TRUE 2.51 2.51 1.30
2 2011.71 60.71    0.10       A       A  TRUE 3.71 3.43  3.00  TRUE 2.62 2.62 1.36
2 2011.81 60.81    0.08       A       A  TRUE 3.81 3.53  7.00  TRUE 2.72 2.72 1.41
2 2011.89 60.89    0.31       A       A FALSE 3.89 3.61 10.00  TRUE 2.80 2.80 1.45
2 2012.20 61.20    0.60       A       A FALSE 4.20 3.61 10.00  TRUE 3.11 3.11 1.61
2 2012.80 61.80    0.07       A       A FALSE 4.80 3.61 10.00  TRUE 3.71 3.71 1.92
2 2012.87 61.87    0.09       A       A  TRUE 4.87 3.61 10.00  TRUE 3.78 3.78 1.96
2 2012.95 61.95    0.38       A       A  TRUE 4.95 3.69 10.37 FALSE 3.86 3.86 2.00
2 2013.33 62.33    0.07       A       A  TRUE 5.33 4.07 12.00 FALSE 4.24 3.86 2.00
2 2013.40 62.40    0.02       A       A  TRUE 5.40 4.14 13.61 FALSE 4.31 3.86 2.00
2 2013.42 62.42    0.09       A       A  TRUE 5.42 4.16 14.00 FALSE 4.33 3.86 2.00
2 2013.50 62.50    0.43       A       A FALSE 5.50 4.24 16.00 FALSE 4.41 3.86 2.00
2 2013.93 62.93    0.07       A       A  TRUE 5.93 4.24 16.00 FALSE 4.84 3.86 2.00
2 2014.00 63.00    0.45       A       A  TRUE 6.00 4.31 16.27 FALSE 4.91 3.86 2.00
2 2014.45 63.45    0.12       A       A FALSE 6.45 4.76 18.00 FALSE 5.36 3.86 2.00
2 2014.57 63.57    0.03       A       A FALSE 6.57 4.76 18.00  TRUE 5.48 3.86 2.00
2 2014.60 63.60    0.60       A       A FALSE 6.60 4.76 18.00  TRUE 5.51 3.90 2.01
2 2015.20 64.20    0.60       A       A FALSE 7.20 4.76 18.00  TRUE 6.11 4.50 2.27
2 2015.80 64.80    0.60       A       A FALSE 7.80 4.76 18.00  TRUE 6.71 5.10 2.53
2 2016.40 65.40    0.51       A       A FALSE 8.40 4.76 18.00  TRUE 7.31 5.70 2.78
2 2016.91 65.91    0.09       A       A FALSE 8.91 4.76 18.00  TRUE 7.82 6.21 3.00
2 2017.00 66.00    0.00       A       A FALSE 9.00 4.76 18.00  TRUE 7.91 6.30 3.67
2 2017.00 66.00    0.13       A       A FALSE 9.00 4.76 18.00  TRUE 7.91 6.30 3.70
2 2017.14 66.14    0.46       A       A FALSE 9.14 4.76 18.00 FALSE 8.05 6.43 4.70
2 2017.60 66.60    0.40       A       D FALSE 9.60 4.76 18.00 FALSE 8.51 6.43 4.70
```

```
> ex2 <- addDrug.Lexis(Sx, pdat, method = "ext", grace = 0.5)
NOTE: timescale taken as 'per'
Values of grace has been recycled across 2 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
```

```
> summary(ex2)
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 56 1        57        1         19         2
> dos <- addDrug.Lexis(Sx, pdat, method = "dos", dpt = 6)
NOTE: timescale taken as 'per'
NOTE: end of exposure based on purchase and dosage (dpt).
> summary(dos)
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 61 1        62        1         19         2
> prdf(dos)
 lex.id     per   age lex.dur lex.Cst lex.Xst  F.ex F.tf F.ct  F.cd  G.ex G.tf G.ct G.cd
      1 2006.00 56.00    0.20       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2006.20 56.20    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2006.80 56.80    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2007.40 57.40    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2008.00 58.00    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2008.60 58.60    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2009.20 59.20    0.43       A       A FALSE 0.00 0.00  0.00 FALSE 0.00 0.00 0.00
      1 2009.63 59.63    0.17       A       A FALSE 0.00 0.00  0.00  TRUE 0.00 0.00 0.00
      1 2009.80 59.80    0.05       A       A FALSE 0.00 0.00  0.00  TRUE 0.17 0.17 1.02
      1 2009.85 59.85    0.55       A       A FALSE 0.00 0.00  0.00 FALSE 0.22 0.22 1.30
      1 2010.40 60.40    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 0.77 0.22 1.30
      1 2011.00 61.00    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 1.37 0.22 1.30
      1 2011.60 61.60    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 1.97 0.22 1.30
      1 2012.20 62.20    0.60       A       A FALSE 0.00 0.00  0.00 FALSE 2.57 0.22 1.30
      1 2012.80 62.80    0.19       A       A FALSE 0.00 0.00  0.00 FALSE 3.17 0.22 1.30
      1 2012.99 62.99    0.03       A       A FALSE 0.00 0.00  0.00  TRUE 3.36 0.22 1.30
      1 2013.02 63.02    0.22       A       A FALSE 0.00 0.00  0.00  TRUE 3.39 0.25 2.60
      1 2013.23 63.23    0.17       A       A FALSE 0.00 0.00  0.00 FALSE 3.60 0.46 3.90
      1 2013.40 63.40    0.56       A       A FALSE 0.00 0.00  0.00 FALSE 3.77 0.46 3.90
      1 2013.96 63.96    0.04       A       A  TRUE 0.00 0.00  0.00 FALSE 4.33 0.46 3.90
      1 2014.00 64.00    0.25       A       A  TRUE 0.04 0.04  0.50 FALSE 4.37 0.46 3.90
      1 2014.25 64.25    0.26       A       A  TRUE 0.29 0.29  4.00 FALSE 4.62 0.46 3.90
      1 2014.51 64.51    0.09       A       A  TRUE 0.54 0.54  8.00 FALSE 4.88 0.46 3.90
      1 2014.60 64.60    0.39       A       A  TRUE 0.64 0.64  8.57 FALSE 4.97 0.46 3.90
      1 2014.99 64.99    0.01       A       A  TRUE 1.03 1.03 11.00 FALSE 5.36 0.46 3.90
      2 2008.00 57.00    0.10       A       A  TRUE 0.00 0.40  2.43 FALSE 0.00 0.00 0.00
      2 2008.10 57.10    0.50       A       A FALSE 0.10 0.50  3.00 FALSE 0.00 0.00 0.00
      2 2008.60 57.60    0.49       A       A FALSE 0.60 0.50  3.00 FALSE 0.00 0.00 0.00
      2 2009.09 58.09    0.11       A       A FALSE 1.09 0.50  3.00  TRUE 0.00 0.00 0.00
      2 2009.20 58.20    0.11       A       A FALSE 1.20 0.50  3.00  TRUE 0.11 0.11 0.67
      2 2009.31 58.31    0.49       A       A FALSE 1.31 0.50  3.00 FALSE 0.22 0.22 1.30
      2 2009.80 58.80    0.60       A       A FALSE 1.80 0.50  3.00 FALSE 0.71 0.22 1.30
      2 2010.40 59.40    0.60       A       A FALSE 2.40 0.50  3.00 FALSE 1.31 0.22 1.30
      2 2011.00 60.00    0.60       A       A FALSE 3.00 0.50  3.00 FALSE 1.91 0.22 1.30
      2 2011.60 60.60    0.00       A       A FALSE 3.60 0.50  3.00  TRUE 2.51 0.22 1.30
      2 2011.60 60.60    0.11       A       A FALSE 3.60 0.50  3.00  TRUE 2.51 0.22 1.30
      2 2011.71 60.71    0.01       A       A  TRUE 3.71 0.50  3.00  TRUE 2.62 0.33 1.97
      2 2011.72 60.72    0.10       A       A  TRUE 3.72 0.51  3.22 FALSE 2.63 0.33 2.00
      2 2011.81 60.81    0.39       A       A  TRUE 3.81 0.60  7.00 FALSE 2.72 0.33 2.00
      2 2012.20 61.20    0.11       A       A  TRUE 4.20 0.99  9.33 FALSE 3.11 0.33 2.00
```

```
      2 2012.31 61.31    0.49       A       A FALSE 4.31 1.10 10.00 FALSE 3.22 0.33 2.00
      2 2012.80 61.80    0.07       A       A FALSE 4.80 1.10 10.00 FALSE 3.71 0.33 2.00
      2 2012.87 61.87    0.33       A       A  TRUE 4.87 1.10 10.00 FALSE 3.78 0.33 2.00
      2 2013.20 62.20    0.13       A       A FALSE 5.20 1.44 12.00 FALSE 4.11 0.33 2.00
      2 2013.33 62.33    0.07       A       A  TRUE 5.33 1.44 12.00 FALSE 4.24 0.33 2.00
      2 2013.40 62.40    0.02       A       A  TRUE 5.40 1.50 13.61 FALSE 4.31 0.33 2.00
      2 2013.42 62.42    0.33       A       A  TRUE 5.42 1.52 14.00 FALSE 4.33 0.33 2.00
      2 2013.75 62.75    0.18       A       A FALSE 5.75 1.85 16.00 FALSE 4.66 0.33 2.00
      2 2013.93 62.93    0.07       A       A  TRUE 5.93 1.85 16.00 FALSE 4.84 0.33 2.00
      2 2014.00 63.00    0.27       A       A  TRUE 6.00 1.92 16.41 FALSE 4.91 0.33 2.00
      2 2014.27 63.27    0.30       A       A FALSE 6.27 2.19 18.00 FALSE 5.18 0.33 2.00
      2 2014.57 63.57    0.03       A       A FALSE 6.57 2.19 18.00  TRUE 5.48 0.33 2.00
      2 2014.60 63.60    0.13       A       A FALSE 6.60 2.19 18.00  TRUE 5.51 0.37 2.20
      2 2014.73 63.73    0.47       A       A FALSE 6.73 2.19 18.00 FALSE 5.64 0.50 3.00
      2 2015.20 64.20    0.60       A       A FALSE 7.20 2.19 18.00 FALSE 6.11 0.50 3.00
      2 2015.80 64.80    0.60       A       A FALSE 7.80 2.19 18.00 FALSE 6.71 0.50 3.00
      2 2016.40 65.40    0.51       A       A FALSE 8.40 2.19 18.00 FALSE 7.31 0.50 3.00
      2 2016.91 65.91    0.09       A       A FALSE 8.91 2.19 18.00  TRUE 7.82 0.50 3.00
      2 2017.00 66.00    0.00       A       A FALSE 9.00 2.19 18.00  TRUE 7.91 0.59 3.67
      2 2017.00 66.00    0.17       A       A FALSE 9.00 2.19 18.00  TRUE 7.91 0.59 3.70
      2 2017.17 66.17    0.43       A       A FALSE 9.17 2.19 18.00 FALSE 8.08 0.76 4.70
      2 2017.60 66.60    0.40       A       D FALSE 9.60 2.19 18.00 FALSE 8.51 0.76 4.70
> fix <- addDrug.Lexis(Sx, pdat, method = "fix", maxt = 1)
NOTE: timescale taken as 'per'
Values of maxt has been recycled across 2 drugs
NOTE: end of exposure based on fixed coverage time of 1 .
> summary(fix)
Transitions:
     To
From  A D  Records:  Events: Risk time:  Persons:
   A 58 1        59        1         19         2
```

3.2 A more realistic example with timing

3.2.1 Follow-up data: `DMlate`

As example data we use the `DMlate` example data from the `Epi` package:

```
> data(DMlate) ; str(DMlate)
'data.frame':        10000 obs. of  7 variables:
 $ sex  : Factor w/ 2 levels "M","F": 2 1 2 2 1 2 1 1 2 1 ...
 $ dobth: num  1940 1939 1918 1965 1933 ...
 $ dodm : num  1999 2003 2005 2009 2009 ...
 $ dodth: num  NA NA NA NA NA ...
 $ dooad: num  NA 2007 NA NA NA ...
 $ doins: num  NA NA NA NA NA NA NA NA NA NA ...
 $ dox  : num  2010 2010 2010 2010 2010 ...
> Lx <- Lexis(entry = list(per = dodm,
+                          age = dodm - dobth,
+                          tfd = 0),
+             exit = list(per = dox),
+       exit.status = factor(!is.na(dodth),
+                            labels = c("DM", "Dead")),
+             data = DMlate)
```

```
NOTE: entry.status has been set to "DM" for all.
NOTE: Dropping  4  rows with duration of follow up < tol
> summary(Lx)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 7497 2499      9996     2499   54273.27      9996
```

We split the data along the age-scale (omitting the variables we shall not need):

```
> library(popEpi)
> Sx <- splitMulti(Lx[,1:7], age = 0:120)
> summary(Sx)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 61627 2499     64126     2499   54273.27      9996
```

3.2.2 Artificial prescription data

To explore how `addDrug.Lexis` works, we also need some drug exposure data, but these are unfortunately not available, so we must simulate three datasets:

```
> set.seed(1952)
> purA <-
+   ( data.frame(lex.id = rep(Lx$lex.id,
+                             round(runif(nrow(Lx), 0, 20))))
+ %>% left_join(Lx[,c("lex.id", "dodm", "dox")])
+ %>% mutate(per = dodm + runif(length(dodm), -0.1, 0.99) * (dox - dodm),
+             amt = sample(4:20*10, length(dodm), replace = TRUE),
+             dpt = amt * round(runif(length(dodm), 3, 7)))
+ %>% select(-dodm, -dox)
+ %>% arrange(lex.id, per)
+   )
> addmargins(table(table(purA$lex.id)))
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18
 504  529  474  480  489  445  485  476  494  489  518  487  501  525  522  524  505  524
  20  Sum
 254 9723
> str(purA)
'data.frame':        100736 obs. of  4 variables:
 $ lex.id: int  1 1 1 1 1 1 1 1 1 1 ...
 $ per   : num  1999 1999 1999 2000 2000 ...
 $ amt   : num  200 160 190 90 160 90 100 90 90 190 ...
 $ dpt   : num  1000 960 1330 360 640 630 500 450 630 950 ...
> purB <-
+   ( data.frame(lex.id = rep(Lx$lex.id,
+                             round(pmax(runif(nrow(Lx), -10, 15), 0))))
+ %>% left_join(Lx[,c("lex.id", "dodm", "dox")])
+ %>% mutate(per = dodm + runif(length(dodm), -0.1, 0.99) * (dox - dodm),
+             amt = sample(4:20*10, length(dodm), replace = TRUE),
+             dpt = amt * round(runif(length(dodm), 5, 9)))
+ %>% select(-dodm, -dox)
+ %>% arrange(lex.id, per)
+   ) -> purB
> addmargins(table(table(purB$lex.id)))
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15  Sum
 374  418  414  406  391  383  361  429  375  415  394  401  375  394  223 5753
```

```
> str(purB)
```

```
'data.frame':        44695 obs. of  4 variables:
 $ lex.id: int  1 1 1 1 1 1 1 1 1 1 ...
 $ per   : num  1998 1998 1999 1999 2000 ...
 $ amt   : num  100 190 110 140 120 90 140 70 150 180 ...
 $ dpt   : num  800 1520 770 980 960 810 1260 560 1050 1440 ...
```

```
> purC <-
+   ( data.frame(lex.id = rep(Lx$lex.id,
+                             round(pmax(runif(nrow(Lx), -5, 12), 0))))
+ %>% left_join(Lx[,c("lex.id", "dodm", "dox")])
+ %>% mutate(per = dodm + runif(length(dodm), -0.1, 0.99) * (dox - dodm),
+             amt = sample(4:20*10, length(dodm), replace = TRUE),
+             dpt = amt * round(runif(length(dodm), 5, 7)))
+ %>% select(-dodm, -dox)
+ %>% arrange(lex.id, per)
+   )
> addmargins(table(table(purB$lex.id)))
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15  Sum
 374  418  414  406  391  383  361  429  375  415  394  401  375  394  223 5753
```

```
> str(purC)
```

```
'data.frame':        42302 obs. of  4 variables:
 $ lex.id: int  1 1 1 1 1 5 5 5 5 5 ...
 $ per   : num  1998 2000 2003 2007 2008 ...
 $ amt   : num  90 40 200 100 100 160 60 90 170 60 ...
 $ dpt   : num  450 240 1200 600 700 1120 360 630 1190 360 ...
```

```
> head(purC)
```

```
  lex.id      per amt  dpt
1      1 1998.363  90  450
2      1 2000.059  40  240
3      1 2002.642 200 1200
4      1 2006.568 100  600
5      1 2007.684 100  700
6      5 2008.775 160 1120
```

Note that the time scale is in years, so the `dpt` must be in amount per year, so that `dpt/amt` is the approximate number of annual drug purchases.

We now have three artificial drug purchase datasets so we can see how `addDrug.Lexis` performs on larger datasets:

```
> save(Sx, purA, purB, purC, file = "~/stat/R/examples/data/SxpurABC.Rda")
```

3.2.3 Using `addDrug`

1000 and 500 persons

We start out with a small sample and a two month grace period to limit the number of gaps:

```
> Sx1 <- subset(Sx, lex.id < 1000)
> pur <- list(A = subset(purA, lex.id < 1000),
+             B = subset(purB, lex.id < 1000),
+             C = subset(purC, lex.id < 1000))
> system.time(ad1 <- addDrug.Lexis(Sx1, pur, tnam = "per", grace = 1/4))
Values of grace has been recycled across 3 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
 35.217   0.014  35.278
> summary(Sx1)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 6037  241      6278      241    5303.26       999
> summary(ad1)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 26885  241     27126      241    5303.26       999
```

We can then cut the number of persons in half:

```
> Sx2 <- subset(Sx, lex.id < 500)
> pur <- list(A = subset(purA, lex.id < 500),
+             B = subset(purB, lex.id < 500),
+             C = subset(purC, lex.id < 500))
> system.time(ad2 <- addDrug.Lexis(Sx2, pur, tnam = "per", grace = 1/6))
Values of grace has been recycled across 3 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
 15.832   0.016  15.846
> summary(Sx2)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 3023  118      3141      118    2653.65       499
> summary(ad2)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 14096  118     14214      118    2653.65       499
```

It looks like timing is broadly proportional to the number of persons.

Fewer prescription records

We can try to cut the number of purchases in half:

```
> pur <- list(A = subset(purA, lex.id < 500 & runif(nrow(purA)) < 0.5),
+             B = subset(purB, lex.id < 500 & runif(nrow(purB)) < 0.5),
+             C = subset(purC, lex.id < 500 & runif(nrow(purC)) < 0.5))
> sapply(pur, nrow)
```

```
   A    B    C
2557 1098 1063
> system.time(ad3 <- addDrug.Lexis(Sx2, pur, tnam = "per", grace = 1/6))
Values of grace has been recycled across 3 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
  9.086   0.000   9.091
> summary(Sx2)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 3023  118      3141      118    2653.65       499
> summary(ad3)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 8620  118      8738      118    2653.65       499
```

It appears that the number of purchases per person is a major determinant of the run time too; the timing is also largely proportional to the number of drug records.

In any concrete application it is recommended to run the function on a fairly small sample of persons, say 1000 to get a feel for the run time. It may also be a good idea to run the function on chunks of the persons, to make sure that you do not lose all the processed data in a crash.

```
> prdf(ad1[1:6,])
 lex.id     per   age  tfd lex.dur lex.Cst lex.Xst  A.ex A.tf A.ct   A.cd B.ex B.tf B.ct
      1 1998.92 58.66 0.00    0.00      DM      DM FALSE 0.00 0.00   0.00 TRUE 0.00 0.11
      1 1998.92 58.66 0.00    0.01      DM      DM FALSE 0.00 0.00   0.00 TRUE 0.00 0.11
      1 1998.93 58.67 0.01    0.05      DM      DM  TRUE 0.00 0.00   0.00 TRUE 0.01 0.12
      1 1998.98 58.72 0.06    0.22      DM      DM  TRUE 0.05 0.05 200.00 TRUE 0.06 0.17
      1 1999.20 58.94 0.28    0.06      DM      DM  TRUE 0.27 0.27 360.00 TRUE 0.28 0.39
      1 1999.26 59.00 0.34    0.05      DM      DM  TRUE 0.33 0.33 381.36 TRUE 0.34 0.45
   B.cd C.ex C.tf C.ct  C.cd
 106.98 TRUE 0.00 0.55 29.43
 110.00 TRUE 0.00 0.56 29.59
 113.67 TRUE 0.01 0.57 30.13
 130.46 TRUE 0.06 0.61 32.61
 209.95 TRUE 0.28 0.84 44.35
 230.65 TRUE 0.34 0.89 47.41
```

Fewer prescription types

We can try to cut the number of drugs, to assess how this influence the run time:

```
> pur <- list(B = subset(purB, lex.id < 500 & runif(nrow(purB)) < 0.5),
+             C = subset(purC, lex.id < 500 & runif(nrow(purC)) < 0.5))
> sapply(pur, nrow)
   B    C
1148 1084
> system.time(ad4 <- addDrug.Lexis(Sx2, pur, tnam = "per", grace = 1/6))
```

```
Values of grace has been recycled across 2 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
  5.287   0.000   5.290
> summary(Sx2)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 3023  118      3141      118    2653.65       499
> summary(ad4)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 5660  118      5778      118    2653.65       499
```

3.2.4 Too many records —`coarse.Lexis`

If we look at the length of the intervals as given in `lex.dur` we see that some are are quite small:

```
> summary(ad1$lex.dur)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
0.0000046 0.0389576 0.1163127 0.1955046 0.2691996 1.0000000
```

Half are smaller then 0.11 years, 40 days. We could without much loss of precision in the analysis merge adjacent records that have total risk time less than 3 months.

The function `coarse.Lexis` will collapse adjacent records with short `lex.dur`. The collapsing will use the covariates from the first record, and so the entire follow-up from the two records will have the characteristics of the first. Therefore it is wise to choose first records with reasonably short `lex.dur`—the approximation of the records will then be better than if the first record was with a larger `lex.dur`. There are two values supplied to `coarse.Lexis`; the maximal length of the first record's `lex.dur` and the maximal length of the `lex.dur` in the resulting record. The larger these parameters are, the more the `Lexis` object is coarsened.

```
> summary(ad1)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 26885  241     27126      241    5303.26       999
> summary(adc <- coarse.Lexis(ad1, lim = c(1/6,1/2)))
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 14654  241     14895      241    5303.26       999
> summary(adc$lex.dur)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.000222 0.207401 0.303527 0.356043 0.440250 1.000000
```

This could cut the number of units for analysis substantially, in this case from about 27,000 to some 13,000.

Records to be kept

However, when we are dealing with drug exposure data we will be interested keeping the record that holds the start of a drug exposure. Some may argue that it does not matter much.

The records (*i.e.* beginnings of FU intervals) that should be kept must be given in logical vector in the argument `keep`:

```
> summary(Sx2)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 3023  118      3141      118    2653.65       499

> system.time(ad4 <- addDrug.Lexis(Sx2,
+                                  pur,
+                                 tnam = "per",
+                                grace = 1/6))

Values of grace has been recycled across 2 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
  4.706   0.000   4.710

> summary(ad4)

Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 5660  118      5778      118    2653.65       499

> #
> ad5 <- coarse.Lexis(ad4,
+                     lim = c(1/4, 1/2))
> summary(ad5)

Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 4550  118      4668      118    2653.65       499

> #
> ad4$keep <- with(ad4, (B.ex & B.ct == 0) |
+                       (C.ex & C.ct == 0))
> ad6 <- coarse.Lexis(ad4,
+                     lim = c(1/4, 1/2),
+                    keep = ad4$keep)
> summary(ad6)

Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 4656  118      4774      118    2653.65       499
```

We see the expected behaviour when we use `coarse.Lexis` we get fewer records, but identical follow-up. And the `keep` argument gives the possibility to keep selected records, or more precisely beginnings. `keep` prevents a record to be collapsed wit a previous one, but not with a subsequent one.

3.2.5 The entire example dataset

The entire amount of example data consist of some 10,000 persons and some 200,000 prescriptions:

```
> dim(Sx)
[1] 64126     7
> pur <- list(A = purA,
+             B = purB,
+             C = purC)
> sapply(pur, nrow)
     A      B      C
100736  44695  42302
> system.time(adx <- addDrug.Lexis(Sx, pur, tnam = "per", grace = 1/6))
Values of grace has been recycled across 3 drugs
NOTE: end of exposure based on differences in purchase times (per)
 and amount purchased (amt).
   user  system elapsed
547.739  16.644 564.595
> system.time(adc <- coarse.Lexis(adx, lim = c(1/6,1/2)))
   user  system elapsed
  0.491   0.000   0.491
> summary(Sx)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 61627 2499     64126     2499   54273.27      9996
> summary(adx)
Transitions:
     To
From     DM Dead  Records:  Events: Risk time:  Persons:
  DM 280231 2499    282730     2499   54273.27      9996
> summary(adc)
Transitions:
     To
From     DM Dead  Records:  Events: Risk time:  Persons:
  DM 150533 2499    153032     2499   54273.27      9996
```

We see hat the number of records is quite large because we have cut at all purchase dates and integer ages. For practical purposes we might therefor want to merge successive records with a total duration `lex.dur` less than some limit. More about that another time.