Time dependent covariates in Lexis objects

SDCC
March 2022
http://bendixcarstensen.com/Epi
Version 5

Compiled Tuesday 25th April, 2023, 07:42
from:

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

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`

. . . provides the ability to amend a `Lexis` object with clinical measurements taken at different times, and propagate the values as LOCF (Last Observation Carried Forward) to all subsequent records. 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.

`addCov.Lexis` will also propagate missing values supplied as measurements. Therefore, if you want to have LOCF *across* supplied times of measurement you must explicitly apply `tidyr::fill` to the resulting `Lexis` object, after a suitable `group_by`.

1.2 `addDrug.Lexis`

As opposed to this, `addDrug.Lexis` will first use drug information at each date of recorded drug purchase, and subsequently `compute` cumulative exposure measures at the times in the resulting `Lexis` object. This is essentially by linear interpolation, so 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 frame 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 as `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 : chr  "A" "B" "C"
 $ 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 [1:2] "per" "age"
 - attr(*, "time.since")= chr [1:2] "" ""
 - attr(*, "breaks")=List of 2
  ..$ per: NULL
  ..$ age: NULL
> (Lx <- Lcoh[,1:6])
 lex.id     per   age lex.dur lex.Cst lex.Xst
      A 1965.59 13.06   31.90   Alive    Dead
      B 1972.69 18.44   22.70   Alive   Alive
      C 1991.97  4.54    6.58   Alive    Dead
```

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 the `_join` functions from `dplyr` 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)
> 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 [1:2] "per" "age"
 - attr(*, "time.since")= chr [1:2] "" ""
 - attr(*, "breaks")=List of 2
  ..$ per: NULL
  ..$ age: NULL
```

```
> Lx
 lex.id     per   age lex.dur lex.Cst lex.Xst
      A 1965.59 13.06   31.90   Alive    Dead
      B 1972.69 18.44   22.70   Alive   Alive
      C 1991.97  4.54    6.58   Alive    Dead
```

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 = cal.yr(c("1977-3-17",
+                                      "1973-7-29",
+                                      "1996-3-1",
+                                      "1990-7-14",
+                                      "1989-1-31")),
+                        bp = c(120, 140, 160, 157, 145),
+                      chol = c(NA, 5, 8, 9, 6),
+                      xnam = c("X2", "X1", "X1", "X2", "X0"),
+          stringsAsFactors = FALSE)
> str(clin)
'data.frame':        5 obs. of  5 variables:
 $ lex.id: chr  "A" "A" "C" "B" ...
 $ per   : 'cal.yr' num  1977 1974 1996 1991 1989
 $ bp    : num  120 140 160 157 145
 $ chol  : num  NA 5 8 9 6
 $ xnam  : chr  "X2" "X1" "X1" "X2" ...
> clin
  lex.id      per  bp chol xnam
1      A 1977.206 120   NA   X2
2      A 1973.573 140    5   X1
3      C 1996.163 160    8   X1
4      B 1990.531 157    9   X2
5      C 1989.083 145    6   X0
```

Note that we have chosen a measurement for person `C` from 1989—before the person's entry to the study, and have an `NA` for `chol` for person `A`.

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))
 lex.id     per   age  tfc lex.dur lex.Cst lex.Xst exnam  bp chol xnam
      A 1965.59 13.06   NA    7.98   Alive   Alive  <NA>  NA   NA <NA>
      A 1973.57 21.04 0.00    3.63   Alive   Alive   ex1 140    5   X1
      A 1977.21 24.67 0.00   20.28   Alive    Dead   ex2 120   NA   X2
      B 1972.69 18.44   NA   17.85   Alive   Alive  <NA>  NA   NA <NA>
      B 1990.53 36.28 0.00    4.86   Alive   Alive   ex1 157    9   X2
      C 1991.97  4.54 2.89    4.19   Alive   Alive   ex1 145    6   X0
      C 1996.16  8.73 0.00    2.40   Alive    Dead   ex2 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, while the variable xnam is just carried over as any other.

If we explicitly give the name of the variable holding the examination names we do not get a constructed exnam. We can also define the name of the (incomplete) timescale to hold the time since measurement, in this case as tfCl:

```
> (Dx <- addCov.Lexis(Lx, clin, exnam = "xnam", tfc = "tfCl"))
 lex.id     per   age tfCl lex.dur lex.Cst lex.Xst xnam  bp chol
      A 1965.59 13.06   NA    7.98   Alive   Alive <NA>  NA   NA
      A 1973.57 21.04 0.00    3.63   Alive   Alive   X1 140    5
      A 1977.21 24.67 0.00   20.28   Alive    Dead   X2 120   NA
      B 1972.69 18.44   NA   17.85   Alive   Alive <NA>  NA   NA
      B 1990.53 36.28 0.00    4.86   Alive   Alive   X2 157    9
      C 1991.97  4.54 2.89    4.19   Alive   Alive   X0 145    6
      C 1996.16  8.73 0.00    2.40   Alive    Dead   X1 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
  ""   ""  "X"
```

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   tfc lex.dur lex.Cst lex.Xst xnam  bp chol
      A 1965.59 13.06    NA    1.94   Alive   Alive <NA>  NA   NA
      A 1967.53 15.00    NA    5.00   Alive   Alive <NA>  NA   NA
      A 1972.53 20.00    NA    1.04   Alive   Alive <NA>  NA   NA
      A 1973.57 21.04  0.00    3.63   Alive   Alive   X1 140    5
      A 1977.21 24.67  0.00    0.33   Alive   Alive   X2 120   NA
      A 1977.53 25.00  0.33    5.00   Alive   Alive   X2 120   NA
      A 1982.53 30.00  5.33    5.00   Alive   Alive   X2 120   NA
      A 1987.53 35.00 10.33    5.00   Alive   Alive   X2 120   NA
      A 1992.53 40.00 15.33    4.95   Alive    Dead   X2 120   NA
      B 1972.69 18.44    NA    1.56   Alive   Alive <NA>  NA   NA
      B 1974.25 20.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1979.25 25.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1984.25 30.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1989.25 35.00    NA    1.28   Alive   Alive <NA>  NA   NA
      B 1990.53 36.28  0.00    3.72   Alive   Alive   X2 157    9
      B 1994.25 40.00  3.72    1.14   Alive   Alive   X2 157    9
      C 1991.97  4.54  2.89    0.46   Alive   Alive   X0 145    6
      C 1992.44  5.00  3.35    3.73   Alive   Alive   X0 145    6
      C 1996.16  8.73  0.00    1.27   Alive   Alive   X1 160    8
      C 1997.44 10.00  1.27    1.12   Alive    Dead   X1 160    8
```

```
> #
> # split AFTER add
> La <- splitLexis(addCov.Lexis(Lx,
+                               clin,
+                              exnam = "xnam" ),
+                    time.scale = "age",
+                        breaks = seq(0, 80, 5))
> La
```

```
 lex.id     per   age   tfc lex.dur lex.Cst lex.Xst xnam  bp chol
      A 1965.59 13.06    NA    1.94   Alive   Alive <NA>  NA   NA
      A 1967.53 15.00    NA    5.00   Alive   Alive <NA>  NA   NA
      A 1972.53 20.00    NA    1.04   Alive   Alive <NA>  NA   NA
      A 1973.57 21.04  0.00    3.63   Alive   Alive   X1 140    5
      A 1977.21 24.67  0.00    0.33   Alive   Alive   X2 120   NA
      A 1977.53 25.00  0.33    5.00   Alive   Alive   X2 120   NA
      A 1982.53 30.00  5.33    5.00   Alive   Alive   X2 120   NA
      A 1987.53 35.00 10.33    5.00   Alive   Alive   X2 120   NA
      A 1992.53 40.00 15.33    4.95   Alive    Dead   X2 120   NA
      B 1972.69 18.44    NA    1.56   Alive   Alive <NA>  NA   NA
      B 1974.25 20.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1979.25 25.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1984.25 30.00    NA    5.00   Alive   Alive <NA>  NA   NA
      B 1989.25 35.00    NA    1.28   Alive   Alive <NA>  NA   NA
      B 1990.53 36.28  0.00    3.72   Alive   Alive   X2 157    9
      B 1994.25 40.00  3.72    1.14   Alive   Alive   X2 157    9
      C 1991.97  4.54  2.89    0.46   Alive   Alive   X0 145    6
      C 1992.44  5.00  3.35    3.73   Alive   Alive   X0 145    6
      C 1996.16  8.73  0.00    1.27   Alive   Alive   X1 160    8
      C 1997.44 10.00  1.27    1.12   Alive    Dead   X1 160    8
```

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   NA TRUE TRUE TRUE TRUE TRUE TRUE   NA   NA   NA   NA   NA TRUE TRUE TRUE
[18] TRUE TRUE TRUE
```

```
> La$age == Lb$age
```

```
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[18] TRUE TRUE TRUE
```

```
> La$per == Lb$per
```

```
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[18] TRUE TRUE TRUE
```

The same goes for `splitMulti`:

```
> if (require(popEpi, quietly=TRUE)) {
+     ## 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
+     Ma$tfc == Mb$tfc
+ }
```

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.

2.4 Filling the `NA`s

As mentioned in the beginning, clinical measurements given as `NA` in the `clin` data frame are carried forward. If you want to have these replaced by 'older' clinical measurements you can do that explicitly by `dplyr::fill` with a construction like:

```
> if (require(tidyr, quietly=TRUE)) {
+     cov <- c("bp", "chol")
+     Lx <- La
+     Lx <- group_by(Lx, lex.id) %>%
+         fill(all_of(cov)) %>%
+         ungroup()
+     class(Lx)
+ }
[1] "tbl_df"     "tbl"        "data.frame"
```

We see that the `Lexis` attributes are lost by using the `group_by` function, so we fish out the covariates from the `tibble` and stick them back into the `Lexis` object:

```
> if (require(tidyr, quietly=TRUE)) {
+     Lx <- La
+     Lx[,cov] <- as.data.frame( group_by(Lx, lex.id)
+                               %>% fill(all_of(cov)))[,cov]
+     class(Lx)
+     La
+     Lx
+ }
 lex.id     per   age   tfc lex.dur lex.Cst lex.Xst xnam  bp chol
      A 1965.59 13.06    NA    1.94   Alive   Alive <NA>  NA   NA
      A 1967.53 15.00    NA    5.00   Alive   Alive <NA>  NA   NA
      A 1972.53 20.00    NA    1.04   Alive   Alive <NA>  NA   NA
      A 1973.57 21.04  0.00    3.63   Alive   Alive   X1 140    5
      A 1977.21 24.67  0.00    0.33   Alive   Alive   X2 120    5
      A 1977.53 25.00  0.33    5.00   Alive   Alive   X2 120    5
      A 1982.53 30.00  5.33    5.00   Alive   Alive   X2 120    5
      A 1987.53 35.00 10.33    5.00   Alive   Alive   X2 120    5
```

```
A 1992.53 40.00 15.33    4.95   Alive    Dead   X2 120    5
B 1972.69 18.44    NA    1.56   Alive   Alive <NA>  NA   NA
B 1974.25 20.00    NA    5.00   Alive   Alive <NA>  NA   NA
B 1979.25 25.00    NA    5.00   Alive   Alive <NA>  NA   NA
B 1984.25 30.00    NA    5.00   Alive   Alive <NA>  NA   NA
B 1989.25 35.00    NA    1.28   Alive   Alive <NA>  NA   NA
B 1990.53 36.28  0.00    3.72   Alive   Alive   X2 157    9
B 1994.25 40.00  3.72    1.14   Alive   Alive   X2 157    9
C 1991.97  4.54  2.89    0.46   Alive   Alive   X0 145    6
C 1992.44  5.00  3.35    3.73   Alive   Alive   X0 145    6
C 1996.16  8.73  0.00    1.27   Alive   Alive   X1 160    8
C 1997.44 10.00  1.27    1.12   Alive    Dead   X1 160    8
```

The slightly convoluted code where the covariate columns are explicitly selected, owes to the fact that the `dplyr` functions will strip the data frames of the `Lexis` attributes. So we needed to use `fill` to just generate the covariates and not touch the `Lexis` object itself.

This should of course be built into `addCov.Lexis` as a separate argument, but is not yet.

Note that the `tfc`, time from clinical measurement is now not a valid variable any more; the 5 in `chol` is measured in 1971.5 but `tfc` is reset to 0 at 1977.3, despite only `bp` but not `chol` is measured at that time. If you want that remedied you will have to use `addCov.Lexis` twice, one with a `clin` data frame with only `bp` and another with a data frame with only `chol`, each generating a differently named time from clinical measurement.

This is a problem that comes from the structure of the supplied *data* not from the program features; in the example we had basically measurements of different clinical variables at different times, and so necessarily also a need for different times since last measurement.

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 (or prescription), 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, which also includes an id variable, `lex.id`, 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. 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
```

```
> str(Sx)
```

```
Classes ‘Lexis’ and 'data.frame':       33 obs. of  6 variables:
 $ lex.id : int  1 1 1 1 1 1 1 1 1 1 ...
 $ per    : num  2006 2006 2007 2007 2008 ...
 $ age    : num  56 56.2 56.8 57.4 58 ...
 $ lex.dur: num  0.2 0.6 0.6 0.6 0.6 ...
 $ lex.Cst: Factor w/ 2 levels "A","D": 1 1 1 1 1 1 1 1 1 1 ...
 $ lex.Xst: Factor w/ 2 levels "A","D": 1 1 1 1 1 1 1 1 1 1 ...
 - attr(*, "breaks")=List of 2
  ..$ per: num [1:51] 1990 1991 1991 1992 1992 ...
  ..$ age: NULL
 - attr(*, "time.scales")= chr [1:2] "per" "age"
 - attr(*, "time.since")= chr [1:2] "" ""
```

Note that as opposed to the previous example, the time scales are not of class `cal.yr`, they are just numerical.

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)
```

We do not need to sort the drug purchase data frames (it is done internally by `addDrug.Lexis`), 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
```

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

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. `addDrug.Lexis` not only adds 8 *variables* (4 from each drug), it also adds *records* representing cuts at the purchase dates and possible expiry dates.

```
> 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 64 1        65        1         19         2

 [1] "lex.id"  "per"     "age"     "lex.dur" "lex.Cst" "lex.Xst" "F.ex"    "F.tf"
 [9] "F.ct"    "F.cd"    "G.ex"    "G.tf"    "G.ct"    "G.cd"
> 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
      1 2006.00 56.00    0.20       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.20 56.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.80 56.80    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
```

```
   1 2007.40 57.40    0.60        A        A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
   1 2008.00 58.00    0.60        A        A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
   1 2008.60 58.60    0.60        A        A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
   1 2009.20 59.20    0.43        A        A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
   1 2009.63 59.63    0.17        A        A FALSE 0.000 0.000  0.000  TRUE 0.000 0.000
   1 2009.80 59.80    0.60        A        A FALSE 0.000 0.000  0.000  TRUE 0.170 0.170
   1 2010.40 60.40    0.60        A        A FALSE 0.000 0.000  0.000  TRUE 0.770 0.770
   1 2011.00 61.00    0.60        A        A FALSE 0.000 0.000  0.000  TRUE 1.370 1.370
   1 2011.60 61.60    0.60        A        A FALSE 0.000 0.000  0.000  TRUE 1.970 1.970
   1 2012.20 62.20    0.60        A        A FALSE 0.000 0.000  0.000  TRUE 2.570 2.570
   1 2012.80 62.80    0.19        A        A FALSE 0.000 0.000  0.000  TRUE 3.170 3.170
   1 2012.99 62.99    0.03        A        A FALSE 0.000 0.000  0.000  TRUE 3.357 3.357
   1 2013.02 63.02    0.03        A        A FALSE 0.000 0.000  0.000  TRUE 3.388 3.388
   1 2013.05 63.05    0.35        A        A FALSE 0.000 0.000  0.000 FALSE 3.420 3.420
   1 2013.40 63.40    0.56        A        A FALSE 0.000 0.000  0.000 FALSE 3.770 3.420
   1 2013.96 63.96    0.00        A        A  TRUE 0.000 0.000  0.000 FALSE 4.334 3.420
   1 2013.96 63.96    0.04        A        A  TRUE 0.000 0.000  0.000 FALSE 4.334 3.420
   1 2014.00 64.00    0.25        A        A  TRUE 0.036 0.036  0.497 FALSE 4.370 3.420
   1 2014.25 64.25    0.00        A        A  TRUE 0.286 0.286  4.000 FALSE 4.621 3.420
   1 2014.25 64.25    0.26        A        A  TRUE 0.286 0.286  4.000 FALSE 4.621 3.420
   1 2014.51 64.51    0.00        A        A  TRUE 0.545 0.545  8.000 FALSE 4.879 3.420
   1 2014.51 64.51    0.09        A        A  TRUE 0.545 0.545  8.000 FALSE 4.879 3.420
   1 2014.60 64.60    0.10        A        A  TRUE 0.636 0.636  9.406 FALSE 4.970 3.420
   1 2014.70 64.70    0.00        A        A FALSE 0.739 0.739 11.000 FALSE 5.073 3.420
   1 2014.70 64.70    0.29        A        A FALSE 0.739 0.739 11.000 FALSE 5.073 3.420
   1 2014.99 64.99    0.01        A        A  TRUE 1.025 0.739 11.000 FALSE 5.360 3.420
   2 2008.00 57.00    0.60        A        A  TRUE 0.000 0.405  0.354 FALSE 0.000 0.000
   2 2008.60 57.60    0.49        A        A  TRUE 0.600 1.005  0.879 FALSE 0.000 0.000
   2 2009.09 58.09    0.11        A        A  TRUE 1.089 1.494  1.308  TRUE 0.000 0.000
   2 2009.20 58.20    0.60        A        A  TRUE 1.200 1.605  1.405  TRUE 0.111 0.111
   2 2009.80 58.80    0.60        A        A  TRUE 1.800 2.205  1.930  TRUE 0.711 0.711
   2 2010.40 59.40    0.60        A        A  TRUE 2.400 2.805  2.455  TRUE 1.311 1.311
   2 2011.00 60.00    0.02        A        A  TRUE 3.000 3.405  2.981  TRUE 1.911 1.911
   2 2011.02 60.02    0.00        A        A FALSE 3.022 3.427  3.000  TRUE 1.933 1.933
   2 2011.02 60.02    0.58        A        A FALSE 3.022 3.427  3.000  TRUE 1.933 1.933
   2 2011.60 60.60    0.00        A        A FALSE 3.599 3.427  3.000  TRUE 2.510 2.510
   2 2011.60 60.60    0.11        A        A FALSE 3.600 3.427  3.000  TRUE 2.511 2.511
   2 2011.71 60.71    0.00        A        A  TRUE 3.710 3.427  3.000  TRUE 2.621 2.621
   2 2011.71 60.71    0.10        A        A  TRUE 3.710 3.427  3.000  TRUE 2.621 2.621
   2 2011.81 60.81    0.08        A        A  TRUE 3.812 3.529  7.000  TRUE 2.723 2.723
   2 2011.89 60.89    0.31        A        A FALSE 3.889 3.605 10.000  TRUE 2.800 2.800
   2 2012.20 61.20    0.60        A        A FALSE 4.200 3.605 10.000  TRUE 3.111 3.111
   2 2012.80 61.80    0.07        A        A FALSE 4.800 3.605 10.000  TRUE 3.711 3.711
   2 2012.87 61.87    0.09        A        A  TRUE 4.865 3.605 10.000  TRUE 3.776 3.776
   2 2012.95 61.95    0.38        A        A  TRUE 4.951 3.691 10.368 FALSE 3.862 3.862
   2 2013.33 62.33    0.07        A        A  TRUE 5.331 4.071 12.000 FALSE 4.242 3.862
   2 2013.40 62.40    0.02        A        A  TRUE 5.400 4.140 13.606 FALSE 4.311 3.862
   2 2013.42 62.42    0.09        A        A  TRUE 5.417 4.157 14.000 FALSE 4.328 3.862
   2 2013.50 62.50    0.43        A        A FALSE 5.503 4.243 16.000 FALSE 4.414 3.862
   2 2013.93 62.93    0.07        A        A  TRUE 5.932 4.243 16.000 FALSE 4.843 3.862
   2 2014.00 63.00    0.45        A        A  TRUE 6.000 4.311 16.265 FALSE 4.911 3.862
   2 2014.45 63.45    0.12        A        A FALSE 6.447 4.758 18.000 FALSE 5.357 3.862
   2 2014.57 63.57    0.03        A        A FALSE 6.566 4.758 18.000  TRUE 5.477 3.862
   2 2014.60 63.60    0.60        A        A FALSE 6.600 4.758 18.000  TRUE 5.511 3.895
   2 2015.20 64.20    0.60        A        A FALSE 7.200 4.758 18.000  TRUE 6.111 4.495
   2 2015.80 64.80    0.60        A        A FALSE 7.800 4.758 18.000  TRUE 6.711 5.095
   2 2016.40 65.40    0.51        A        A FALSE 8.400 4.758 18.000  TRUE 7.311 5.695
```

```
     2 2016.91 65.91    0.09       A       A FALSE 8.912 4.758 18.000  TRUE 7.822 6.207
     2 2017.00 66.00    0.00       A       A FALSE 9.000 4.758 18.000  TRUE 7.911 6.295
     2 2017.00 66.00    0.13       A       A FALSE 9.004 4.758 18.000  TRUE 7.914 6.299
     2 2017.14 66.14    0.46       A       A FALSE 9.135 4.758 18.000 FALSE 8.046 6.430
     2 2017.60 66.60    0.40       A       D FALSE 9.600 4.758 18.000 FALSE 8.511 6.430
 G.cd
0.000
0.000
0.000
0.000
0.000
0.000
0.000
0.000
0.066
0.298
0.531
0.763
0.995
1.228
1.300
2.600
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
3.900
0.000
0.000
0.000
0.057
0.368
0.679
0.990
1.001
1.001
1.300
1.300
1.358
1.358
1.410
1.450
1.611
1.922
1.956
2.000
2.000
2.000
2.000
```

```
 2.000
 2.000
 2.000
 2.000
 2.000
 2.014
 2.270
 2.526
 2.782
 3.000
 3.673
 3.700
 4.700
 4.700
```

```
> 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 62 1        63        1         19         2
```

```
> ex2
```

```
 lex.id     per   age lex.dur lex.Cst lex.Xst  F.ex  F.tf  F.ct   F.cd  G.ex  G.tf  G.ct
      1 2006.00 56.00    0.20       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.20 56.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.80 56.80    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2007.40 57.40    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2008.00 58.00    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2008.60 58.60    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2009.20 59.20    0.43       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2009.63 59.63    0.17       A       A FALSE 0.000 0.000  0.000  TRUE 0.000 0.000
      1 2009.80 59.80    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 0.170 0.170
      1 2010.40 60.40    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 0.770 0.770
      1 2011.00 61.00    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 1.370 1.370
      1 2011.60 61.60    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 1.970 1.970
      1 2012.20 62.20    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 2.570 2.570
      1 2012.80 62.80    0.19       A       A FALSE 0.000 0.000  0.000  TRUE 3.170 3.170
      1 2012.99 62.99    0.03       A       A FALSE 0.000 0.000  0.000  TRUE 3.357 3.357
      1 2013.02 63.02    0.38       A       A FALSE 0.000 0.000  0.000  TRUE 3.388 3.388
      1 2013.40 63.40    0.15       A       A FALSE 0.000 0.000  0.000  TRUE 3.770 3.770
      1 2013.55 63.55    0.41       A       A FALSE 0.000 0.000  0.000 FALSE 3.920 3.920
      1 2013.96 63.96    0.00       A       A  TRUE 0.000 0.000  0.000 FALSE 4.334 3.920
      1 2013.96 63.96    0.04       A       A  TRUE 0.000 0.000  0.000 FALSE 4.334 3.920
      1 2014.00 64.00    0.25       A       A  TRUE 0.036 0.036  0.497 FALSE 4.370 3.920
      1 2014.25 64.25    0.00       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 3.920
      1 2014.25 64.25    0.26       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 3.920
      1 2014.51 64.51    0.00       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 3.920
      1 2014.51 64.51    0.09       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 3.920
      1 2014.60 64.60    0.39       A       A  TRUE 0.636 0.636  8.567 FALSE 4.970 3.920
      1 2014.99 64.99    0.00       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 3.920
      1 2014.99 64.99    0.01       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 3.920
      2 2008.00 57.00    0.60       A       A  TRUE 0.000 0.405  0.309 FALSE 0.000 0.000
      2 2008.60 57.60    0.49        A       A  TRUE 0.600 1.005  0.768 FALSE 0.000 0.000
      2 2009.09 58.09    0.11        A       A  TRUE 1.089 1.494  1.141  TRUE 0.000 0.000
      2 2009.20 58.20    0.60        A       A  TRUE 1.200 1.605  1.226  TRUE 0.111 0.111
      2 2009.80 58.80    0.60        A       A  TRUE 1.800 2.205  1.684  TRUE 0.711 0.711
      2 2010.40 59.40    0.60        A       A  TRUE 2.400 2.805  2.143  TRUE 1.311 1.311
      2 2011.00 60.00    0.52        A       A  TRUE 3.000 3.405  2.601  TRUE 1.911 1.911
      2 2011.52 60.52    0.00        A       A FALSE 3.522 3.927  3.000  TRUE 2.433 2.433
      2 2011.52 60.52    0.08        A       A FALSE 3.522 3.927  3.000  TRUE 2.433 2.433
      2 2011.60 60.60    0.00        A       A FALSE 3.599 3.927  3.000  TRUE 2.510 2.510
      2 2011.60 60.60    0.11        A       A FALSE 3.600 3.927  3.000  TRUE 2.511 2.511
      2 2011.71 60.71    0.00        A       A  TRUE 3.710 3.927  3.000  TRUE 2.621 2.621
      2 2011.71 60.71    0.10        A       A  TRUE 3.710 3.927  3.000  TRUE 2.621 2.621
      2 2011.81 60.81    0.39        A       A  TRUE 3.812 4.029  7.000  TRUE 2.723 2.723
      2 2012.20 61.20    0.19        A       A  TRUE 4.200 4.416  9.017  TRUE 3.111 3.111
      2 2012.39 61.39    0.41        A       A FALSE 4.389 4.605 10.000  TRUE 3.300 3.300
      2 2012.80 61.80    0.07        A       A FALSE 4.800 4.605 10.000  TRUE 3.711 3.711
      2 2012.87 61.87    0.47        A       A  TRUE 4.865 4.605 10.000  TRUE 3.776 3.776
      2 2013.33 62.33    0.07        A       A  TRUE 5.331 5.071 12.000  TRUE 4.242 4.242
      2 2013.40 62.40    0.02        A       A  TRUE 5.400 5.140 13.606  TRUE 4.311 4.311
      2 2013.42 62.42    0.03        A       A  TRUE 5.417 5.157 14.000  TRUE 4.328 4.328
      2 2013.45 62.45    0.48        A       A  TRUE 5.451 5.191 14.132 FALSE 4.362 4.362
      2 2013.93 62.93    0.07        A       A  TRUE 5.932 5.672 16.000 FALSE 4.843 4.362
      2 2014.00 63.00    0.57        A       A  TRUE 6.000 5.740 16.134 FALSE 4.911 4.362
      2 2014.57 63.57    0.03        A       A  TRUE 6.566 6.306 17.251  TRUE 5.477 4.362
      2 2014.60 63.60    0.35        A       A  TRUE 6.600 6.340 17.317  TRUE 5.511 4.395
      2 2014.95 63.95    0.25        A       A FALSE 6.947 6.687 18.000  TRUE 5.857 4.742
      2 2015.20 64.20    0.60        A       A FALSE 7.200 6.687 18.000  TRUE 6.111 4.995
      2 2015.80 64.80    0.60        A       A FALSE 7.800 6.687 18.000  TRUE 6.711 5.595
      2 2016.40 65.40    0.51        A       A FALSE 8.400 6.687 18.000  TRUE 7.311 6.195
      2 2016.91 65.91    0.09        A       A FALSE 8.912 6.687 18.000  TRUE 7.822 6.707
      2 2017.00 66.00    0.00        A       A FALSE 9.000 6.687 18.000  TRUE 7.911 6.795
      2 2017.00 66.00    0.60        A       A FALSE 9.004 6.687 18.000  TRUE 7.914 6.799
      2 2017.60 66.60    0.04        A       A FALSE 9.600 6.687 18.000  TRUE 8.511 7.395
      2 2017.64 66.64    0.36        A       D FALSE 9.635 6.687 18.000 FALSE 8.546 7.430
  G.cd
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.066
 0.298
 0.531
 0.763
 0.995
 1.228
 1.300
 2.600
 3.534
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 0.000
 0.000
 0.000
 0.057
 0.368
 0.679
 0.990
 1.260
 1.260
 1.300
 1.300
 1.342
 1.342
 1.381
 1.527
 1.598
 1.754
 1.779
 1.955
 1.981
 1.987
 2.000
 2.000
 2.000
 2.000
 2.014
 2.162
 2.270
 2.526
 2.782
 3.000
 3.673
 3.700
 4.644
 4.700
```

```
> 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 66 1        67        1         19         2
```

```
> 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
      1 2006.00 56.00    0.20       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.20 56.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.80 56.80    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2007.40 57.40    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
```

```
1 2008.00 58.00    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
1 2008.60 58.60    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
1 2009.20 59.20    0.43       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
1 2009.63 59.63    0.17       A       A FALSE 0.000 0.000  0.000  TRUE 0.000 0.000
1 2009.80 59.80    0.05       A       A FALSE 0.000 0.000  0.000  TRUE 0.170 0.170
1 2009.85 59.85    0.55       A       A FALSE 0.000 0.000  0.000 FALSE 0.217 0.217
1 2010.40 60.40    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.770 0.217
1 2011.00 61.00    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 1.370 0.217
1 2011.60 61.60    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 1.970 0.217
1 2012.20 62.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 2.570 0.217
1 2012.80 62.80    0.19       A       A FALSE 0.000 0.000  0.000 FALSE 3.170 0.217
1 2012.99 62.99    0.03       A       A FALSE 0.000 0.000  0.000  TRUE 3.357 0.217
1 2013.02 63.02    0.22       A       A FALSE 0.000 0.000  0.000  TRUE 3.388 0.248
1 2013.23 63.23    0.17       A       A FALSE 0.000 0.000  0.000 FALSE 3.605 0.465
1 2013.40 63.40    0.56       A       A FALSE 0.000 0.000  0.000 FALSE 3.770 0.465
1 2013.96 63.96    0.00       A       A  TRUE 0.000 0.000  0.000 FALSE 4.334 0.465
1 2013.96 63.96    0.04       A       A  TRUE 0.000 0.000  0.000 FALSE 4.334 0.465
1 2014.00 64.00    0.25       A       A  TRUE 0.036 0.036  0.497 FALSE 4.370 0.465
1 2014.25 64.25    0.00       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 0.465
1 2014.25 64.25    0.26       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 0.465
1 2014.51 64.51    0.00       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 0.465
1 2014.51 64.51    0.09       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 0.465
1 2014.60 64.60    0.39       A       A  TRUE 0.636 0.636  8.567 FALSE 4.970 0.465
1 2014.99 64.99    0.00       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 0.465
1 2014.99 64.99    0.01       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 0.465
2 2008.00 57.00    0.10       A       A  TRUE 0.000 0.405  2.428 FALSE 0.000 0.000
2 2008.10 57.10    0.00       A       A FALSE 0.095 0.500  3.000 FALSE 0.000 0.000
2 2008.10 57.10    0.50       A       A FALSE 0.095 0.500  3.000 FALSE 0.000 0.000
2 2008.60 57.60    0.49       A       A FALSE 0.600 0.500  3.000 FALSE 0.000 0.000
2 2009.09 58.09    0.11       A       A FALSE 1.089 0.500  3.000  TRUE 0.000 0.000
2 2009.20 58.20    0.11       A       A FALSE 1.200 0.500  3.000  TRUE 0.111 0.111
2 2009.31 58.31    0.49       A       A FALSE 1.306 0.500  3.000 FALSE 0.217 0.217
2 2009.80 58.80    0.60       A       A FALSE 1.800 0.500  3.000 FALSE 0.711 0.217
2 2010.40 59.40    0.60       A       A FALSE 2.400 0.500  3.000 FALSE 1.311 0.217
2 2011.00 60.00    0.60       A       A FALSE 3.000 0.500  3.000 FALSE 1.911 0.217
2 2011.60 60.60    0.00       A       A FALSE 3.599 0.500  3.000  TRUE 2.510 0.217
2 2011.60 60.60    0.11       A       A FALSE 3.600 0.500  3.000  TRUE 2.511 0.217
2 2011.71 60.71    0.01       A       A  TRUE 3.710 0.500  3.000  TRUE 2.621 0.328
2 2011.72 60.72    0.10       A       A  TRUE 3.716 0.506  3.216 FALSE 2.627 0.333
2 2011.81 60.81    0.39       A       A  TRUE 3.812 0.602  7.000 FALSE 2.723 0.333
2 2012.20 61.20    0.11       A       A  TRUE 4.200 0.990  9.326 FALSE 3.111 0.333
2 2012.31 61.31    0.49       A       A FALSE 4.312 1.102 10.000 FALSE 3.223 0.333
2 2012.80 61.80    0.07       A       A FALSE 4.800 1.102 10.000 FALSE 3.711 0.333
2 2012.87 61.87    0.33       A       A  TRUE 4.865 1.102 10.000 FALSE 3.776 0.333
2 2013.20 62.20    0.13       A       A FALSE 5.199 1.435 12.000 FALSE 4.109 0.333
2 2013.33 62.33    0.07       A       A  TRUE 5.331 1.435 12.000 FALSE 4.242 0.333
2 2013.40 62.40    0.02       A       A  TRUE 5.400 1.504 13.606 FALSE 4.311 0.333
2 2013.42 62.42    0.33       A       A  TRUE 5.417 1.521 14.000 FALSE 4.328 0.333
2 2013.75 62.75    0.18       A       A FALSE 5.750 1.855 16.000 FALSE 4.661 0.333
2 2013.93 62.93    0.07       A       A  TRUE 5.932 1.855 16.000 FALSE 4.843 0.333
2 2014.00 63.00    0.27       A       A  TRUE 6.000 1.923 16.409 FALSE 4.911 0.333
2 2014.27 63.27    0.30       A       A FALSE 6.265 2.188 18.000 FALSE 5.176 0.333
2 2014.57 63.57    0.03       A       A FALSE 6.566 2.188 18.000  TRUE 5.477 0.333
2 2014.60 63.60    0.13       A       A FALSE 6.600 2.188 18.000  TRUE 5.511 0.367
2 2014.73 63.73    0.47       A       A FALSE 6.733 2.188 18.000 FALSE 5.644 0.500
2 2015.20 64.20    0.60       A       A FALSE 7.200 2.188 18.000 FALSE 6.111 0.500
2 2015.80 64.80    0.60       A       A FALSE 7.800 2.188 18.000 FALSE 6.711 0.500
```

```
     2 2016.40 65.40    0.51       A       A FALSE 8.400 2.188 18.000 FALSE 7.311 0.500
     2 2016.91 65.91    0.09       A       A FALSE 8.912 2.188 18.000  TRUE 7.822 0.500
     2 2017.00 66.00    0.00       A       A FALSE 9.000 2.188 18.000  TRUE 7.911 0.588
     2 2017.00 66.00    0.17       A       A FALSE 9.004 2.188 18.000  TRUE 7.914 0.592
     2 2017.17 66.17    0.43       A       A FALSE 9.170 2.188 18.000 FALSE 8.081 0.759
     2 2017.60 66.60    0.40       A       D FALSE 9.600 2.188 18.000 FALSE 8.511 0.759
  G.cd
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 1.020
 1.300
 1.300
 1.300
 1.300
 1.300
 1.300
 1.300
 2.600
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 0.000
 0.000
 0.000
 0.000
 0.000
 0.665
 1.300
 1.300
 1.300
 1.300
 1.300
 1.304
 1.967
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.201
 3.000
 3.000
 3.000
 3.000
 3.000
 3.673
 3.700
 4.700
 4.700
```

```
> 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 63 1        64        1         19         2
```

```
> fix
```

```
 lex.id     per   age lex.dur lex.Cst lex.Xst  F.ex  F.tf  F.ct   F.cd  G.ex  G.tf  G.ct
      1 2006.00 56.00    0.20       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.20 56.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2006.80 56.80    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2007.40 57.40    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2008.00 58.00    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2008.60 58.60    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2009.20 59.20    0.43       A       A FALSE 0.000 0.000  0.000 FALSE 0.000 0.000
      1 2009.63 59.63    0.17       A       A FALSE 0.000 0.000  0.000  TRUE 0.000 0.000
      1 2009.80 59.80    0.60       A       A FALSE 0.000 0.000  0.000  TRUE 0.170 0.170
      1 2010.40 60.40    0.23       A       A FALSE 0.000 0.000  0.000  TRUE 0.770 0.770
      1 2010.63 60.63    0.37       A       A FALSE 0.000 0.000  0.000 FALSE 1.000 1.000
      1 2011.00 61.00    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 1.370 1.000
      1 2011.60 61.60    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 1.970 1.000
      1 2012.20 62.20    0.60       A       A FALSE 0.000 0.000  0.000 FALSE 2.570 1.000
      1 2012.80 62.80    0.19       A       A FALSE 0.000 0.000  0.000 FALSE 3.170 1.000
      1 2012.99 62.99    0.03       A       A FALSE 0.000 0.000  0.000  TRUE 3.357 1.000
      1 2013.02 63.02    0.38       A       A FALSE 0.000 0.000  0.000  TRUE 3.388 1.031
      1 2013.40 63.40    0.56       A       A FALSE 0.000 0.000  0.000  TRUE 3.770 1.413
      1 2013.96 63.96    0.00       A       A  TRUE 0.000 0.000  0.000  TRUE 4.334 1.978
      1 2013.96 63.96    0.04       A       A  TRUE 0.000 0.000  0.000  TRUE 4.334 1.978
      1 2014.00 64.00    0.02       A       A  TRUE 0.036 0.036  0.497  TRUE 4.370 2.013
      1 2014.02 64.02    0.23       A       A  TRUE 0.054 0.054  0.752 FALSE 4.388 2.031
      1 2014.25 64.25    0.00       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 2.031
      1 2014.25 64.25    0.26       A       A  TRUE 0.286 0.286  4.000 FALSE 4.621 2.031
      1 2014.51 64.51    0.00       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 2.031
      1 2014.51 64.51    0.09       A       A  TRUE 0.545 0.545  8.000 FALSE 4.879 2.031
      1 2014.60 64.60    0.39       A       A  TRUE 0.636 0.636  8.567 FALSE 4.970 2.031
      1 2014.99 64.99    0.00       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 2.031
      1 2014.99 64.99    0.01       A       A  TRUE 1.025 1.025 11.000 FALSE 5.360 2.031
      2 2008.00 57.00    0.60       A       A  TRUE 0.000 0.405  1.214 FALSE 0.000 0.000
      2 2008.60 57.60    0.00       A       A FALSE 0.595 1.000  3.000 FALSE 0.000 0.000
      2 2008.60 57.60    0.00       A       A FALSE 0.595 1.000  3.000 FALSE 0.000 0.000
      2 2008.60 57.60    0.49       A       A FALSE 0.600 1.000  3.000 FALSE 0.000 0.000
      2 2009.09 58.09    0.11       A       A FALSE 1.089 1.000  3.000  TRUE 0.000 0.000
      2 2009.20 58.20    0.60       A       A FALSE 1.200 1.000  3.000  TRUE 0.111 0.111
      2 2009.80 58.80    0.29       A       A FALSE 1.800 1.000  3.000  TRUE 0.711 0.711
      2 2010.09 59.09    0.31       A       A FALSE 2.089 1.000  3.000 FALSE 1.000 1.000
      2 2010.40 59.40    0.60       A       A FALSE 2.400 1.000  3.000 FALSE 1.311 1.000
      2 2011.00 60.00    0.60       A       A FALSE 3.000 1.000  3.000 FALSE 1.911 1.000
      2 2011.60 60.60    0.00       A       A FALSE 3.599 1.000  3.000  TRUE 2.510 1.000
      2 2011.60 60.60    0.11       A       A FALSE 3.600 1.000  3.000  TRUE 2.511 1.001
      2 2011.71 60.71    0.10       A       A  TRUE 3.710 1.000  3.000  TRUE 2.621 1.111
      2 2011.81 60.81    0.39       A       A  TRUE 3.812 1.102  7.000  TRUE 2.723 1.213
      2 2012.20 61.20    0.40       A       A  TRUE 4.200 1.490  8.163  TRUE 3.111 1.601
      2 2012.60 61.60    0.20       A       A  TRUE 4.599 1.889  9.361 FALSE 3.510 2.000
      2 2012.80 61.80    0.01       A       A  TRUE 4.800 2.090  9.963 FALSE 3.711 2.000
      2 2012.81 61.81    0.05       A       A FALSE 4.812 2.102 10.000 FALSE 3.723 2.000
      2 2012.87 61.87    0.47       A       A  TRUE 4.865 2.102 10.000 FALSE 3.776 2.000
      2 2013.33 62.33    0.07       A       A  TRUE 5.331 2.568 12.000 FALSE 4.242 2.000
      2 2013.40 62.40    0.02       A       A  TRUE 5.400 2.637 13.606 FALSE 4.311 2.000
      2 2013.42 62.42    0.51       A       A  TRUE 5.417 2.654 14.000 FALSE 4.328 2.000
      2 2013.93 62.93    0.07       A       A  TRUE 5.932 3.168 16.000 FALSE 4.843 2.000
      2 2014.00 63.00    0.57       A       A  TRUE 6.000 3.237 16.136 FALSE 4.911 2.000
      2 2014.57 63.57    0.03       A       A  TRUE 6.566 3.803 17.269  TRUE 5.477 2.000
      2 2014.60 63.60    0.33       A       A  TRUE 6.600 3.837 17.336  TRUE 5.511 2.034
      2 2014.93 63.93    0.27       A       A FALSE 6.932 4.168 18.000  TRUE 5.843 2.365
      2 2015.20 64.20    0.37       A       A FALSE 7.200 4.168 18.000  TRUE 6.111 2.634
      2 2015.57 64.57    0.23       A       A FALSE 7.566 4.168 18.000 FALSE 6.477 3.000
      2 2015.80 64.80    0.60       A       A FALSE 7.800 4.168 18.000 FALSE 6.711 3.000
      2 2016.40 65.40    0.51       A       A FALSE 8.400 4.168 18.000 FALSE 7.311 3.000
      2 2016.91 65.91    0.09       A       A FALSE 8.912 4.168 18.000  TRUE 7.822 3.000
      2 2017.00 66.00    0.00       A       A FALSE 9.000 4.168 18.000  TRUE 7.911 3.088
      2 2017.00 66.00    0.60       A       A FALSE 9.004 4.168 18.000  TRUE 7.914 3.092
      2 2017.60 66.60    0.40       A       D FALSE 9.600 4.168 18.000  TRUE 8.511 3.688
  G.cd
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.000
 0.221
 1.001
 1.300
 1.300
 1.300
 1.300
 1.300
 1.300
 2.600
 3.096
 3.830
 3.830
 3.876
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 3.900
 0.000
 0.000
 0.000
 0.000
 0.000
 0.144
 0.924
 1.300
 1.300
 1.300
 1.300
 1.301
 1.378
 1.449
 1.721
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.000
 2.034
 2.365
 2.634
 3.000
 3.000
 3.000
 3.000
 3.673
 3.700
 4.299
```

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):

```
> Sx <- splitLexis(Lx[,1:7], time.scale="age", breaks = 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 simulate three datasets representing three types of drugs:

```
> 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
  19   20  Sum
 498  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:

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
 21.470   0.008  21.479
```

```
> 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 28720  241     28961      241    5303.26       999
```

We 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
 11.053   0.004  11.060
```

```
> 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 15413  118     15531      118    2653.65       499
```

. . . 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
  5.117   0.008   5.126
```

```
> 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 9646  118      9764      118    2653.65       499
```

It also appears that the number of purchases per person is also a determinant of the run time too; the timing is 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.

Fewer prescription types

Finally we try to cut the number of drugs, to assess how this influences the run time:

```
> pur <- list(B = subset(purB, lex.id < 500),
+             C = subset(purC, lex.id < 500))
> sapply(pur, nrow)
   B    C
2197 2170
```

```
> 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
  4.074   0.000   4.075
```

```
> 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 8976  118      9094      118    2653.65       499
```

We see that the number of drugs also influence the run time proportionally.

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. 
0.00000 0.02848 0.10277 0.18312 0.25620 1.00000 
```

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

The function `coarse.Lexis` will collapse records with short `lex.dur` with the subsequent record. 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 will be better than if the first record was with a larger `lex.dur`. Therefore 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 combined 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 28720  241     28961      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 14670  241     14911      241    5303.26       999
```

```
> summary(adc$lex.dur)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.2079  0.3048  0.3557  0.4406  1.0000 
```

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

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, though.

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
  4.248   0.000   4.248
> summary(ad4)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 8976  118      9094      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 5460  118      5578      118    2653.65       499
```

We can identify the first date of exposure to drug `B`, say, by the exposure (`B.ex`) being true and the cumulative time on the drug (`B.ct`) being 0:

```
> 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 5578  118      5696      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 with a previous one, but not with a subsequent one.