# Cutting at multiple events: `mcutLexis` Multiple microvascular complications

SDCC, Melbourne
February 2017
http://bendixcarstensen.com/SDC/lbh
Draft version 2

Compiled Saturday 18th February, 2017, 14:54
from: `/home/bendix/sdc/coll/lbh/r/mcut.tex`

Bendix Carstensen Steno Diabetes Center, Gentofte, Denmark
& Department of Biostatistics, University of Copenhagen
<bcar0029@regionh.dk> <b@bxc.dk>
http://BendixCarstensen.com

# Contents

# Chapter 1

# Multiple splits

When you have multiple events of interest you would like handy way of setting up a cut Lexis objects where the sojourn time is subdivided according to the history of events.

The following is mostly R-code leading up to the definition of `mcutLexis`.

## 1.1 Microvascular complications data

```
> options( keep.source=TRUE, width=90 )
> library( Epi )
> sessionInfo()
```

```
R version 3.3.2 (2016-10-31)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 14.04.5 LTS

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C                LC_TIME=en_DK.UTF-8
 [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8     LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                   LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] utils     datasets  graphics  grDevices stats     methods   base

other attached packages:
[1] Epi_2.9

loaded via a namespace (and not attached):
 [1] cmprsk_2.2-7      MASS_7.3-45       Matrix_1.2-6      plyr_1.8.4
 [5] parallel_3.3.2    survival_2.40-1   etm_0.6-2         Rcpp_0.12.5
 [9] splines_3.3.2     grid_3.3.2        numDeriv_2014.2-1 lattice_0.20-33
```

```
> load( "../data/ndm.Rda" )
> lls()
```

```
  name mode class      dim           size(Kb)
1 ndm  list data.frame 4033 13          383.3
```

```
> head( ndm )
```

```
  newid    sex    doBth     doDM    doEnt    doRet   doNeph    doNeu      doX    doDth
1     1   Male 1962.249 1975.495 2002.181 2002.181 2002.181 2002.181 2011.218 2011.218
2     2   Male 1950.914 2007.183 2009.858       NA       NA       NA 2013.745       NA
3     3 Female 1945.324 2000.497 2002.728       NA       NA 2004.571 2008.500       NA
```

```
4     5 Female 1951.845 1981.496 2001.157 2001.157 2001.157 2004.954 2011.999 2011.999
5     7 Female 1954.115 2007.155 2009.472       NA       NA       NA 2012.579       NA
6     8   Male 1961.732 1963.495 2002.865 2002.865 2005.450 2010.038 2013.088 2013.088
       doR      doK      doN
1 2002.962 2001.938 2002.622
2       NA       NA       NA
3       NA       NA 2004.970
4 2000.210 2001.337 2005.726
5       NA       NA       NA
6 2002.164 2005.933 2010.415
```

We make a Lexis object from entry to exit:

```
> L0 <- Lexis( entry = list( age = doEnt-doBth,
+                            per = doEnt,
+                            dur = doEnt-doDM ),
+              exit = list( per = doX ),
+       exit.status = factor( !is.na(doDth) & abs(doDth-doX) < 1/1000,
+                             labels=c("None","Dead") ),
+              data = ndm )
NOTE: entry.status has been set to "None" for all.
```

Note that there are deaths occurring *after* `doX`, hence we need to only use deaths occurring at `doX`. Further note that by default the factor levels from a logical refer to (`FALSE`,`TRUE`) in that order, and since `Lexis` by default assigns all entry states (unless specified by `entry.status`) to the *first* level of the status factor the specification must be as "Death" and not as "Survival".

In this example we use "None" to indicate that persons start without complications.

We want to use the dates of complications occurrence, `doK`, `doN` and `doR`, to cut the follow-up in epochs with different complications status.

# 1.2 Splitting at complications events

## 1.2.1 Naive approach

The following naive approach by just cutting at each complications date obviously does not work to get us what we want — namely getting all possible sequences of events as states, even is we use the argument `split`:

```
> LK <- cutLexis( L0, cut = L0$doK,
+                     time = "per",
+                new.state = "K",
+                      pre = "None",
+                    split = TRUE )
> summary( LK )
Transitions:
     To
From   None    K Dead Dead(K)  Records:  Events: Risk time:  Persons:
  None 1960 1422   75       0      3457     1497   16341.96      3457
  K       0 1735    0     252      1987      252   12218.69      1987
  Sum  1960 3157   75     252      5444     1749   28560.65      4022
> LKN <- cutLexis( LK, cut = LK$doN,
+                      time = "per",
+                 new.state = "N",
```

```
+                     pre = c("None","K"),
+                   split = TRUE )
> summary( LKN )
```

```
Transitions:
     To
From   None    K    N Dead Dead(K) Dead(K)(N) Dead(N)  Records:  Events: Risk time:
  None 1457  833  681   24       0          0       0      2995     1538   11439.07
  K       0  761  477    0      30          0       0      1268      507    5339.78
  N       0    0 2066    0       0        222      51      2339      273   11781.81
  Sum  1457 1594 3224   24      30        222      51      6602     2318   28560.65

Transitions:
     To
From    Persons:
  None      2995
  K         1268
  N         1750
  Sum       4022
```

```
> LN <- cutLexis( L0, cut = L0$doN,
+                     time = "per",
+                new.state = "N",
+                      pre = "None",
+                    split = TRUE )
> summary( LN )
```

```
Transitions:
     To
From   None    N Dead Dead(N)  Records:  Events: Risk time:  Persons:
  None 2218 1158   54       0      3430     1212   16778.84      3430
  N       0 1477    0     273      1750      273   11781.81      1750
  Sum  2218 2635   54     273      5180     1485   28560.65      4022
```

```
> LNK <- cutLexis( LN, cut = LN$doK,
+                      time = "per",
+                 new.state = "K",
+                       pre = c("None","N"),
+                     split = TRUE )
> summary( LNK )
```

```
Transitions:
     To
From   None    N    K Dead Dead(N) Dead(N)(K) Dead(K)  Records:  Events: Risk time:
  None 1457  681  833   24       0          0       0      2995     1538   11439.07
  N       0  503  589    0      51          0       0      1143      640    4902.89
  K       0    0 2212    0       0        222      30      2464      252   12218.69
  Sum  1457 1184 3634   24      51        222      30      6602     2430   28560.65

Transitions:
     To
From    Persons:
  None      2995
  N         1143
  K         1987
  Sum       4022
```

```
> summary( Relevel(LNK,c(1,3,2,4,7,6,5)) )
```

```
Transitions:
     To
From   None    K    N Dead Dead(K) Dead(N)(K) Dead(N)  Records:  Events: Risk time:
  None 1457  833  681   24       0          0       0      2995     1538   11439.07
```

```
  K       0 2212    0    0      30        222       0      2464      252   12218.69
  N       0  589  503    0       0          0      51      1143      640    4902.89
  Sum  1457 3634 1184   24      30        222      51      6602     2430   28560.65

Transitions:
     To
From    Persons:
  None      2995
  K         1987
  N         1143
  Sum       4022
> with( LO, table( doN<doK, exclude=NULL ) )
FALSE  TRUE  <NA>
  574   691  2757
> ii <- LO$lex.id[LO$doN<LO$doK]
> jj <- LO$lex.id[LO$doN>LO$doK]
> ( idx <- c( ii[!is.na(ii)][1],
+             jj[!is.na(jj)][1] ) )
[1] 12  1
> subset( LO, lex.id %in% idx )[,c(2,4:7,12,17:20)]
        per  lex.dur lex.Cst lex.Xst lex.id    doEnt    doDth      doR      doK      doN
1  2002.181 9.037645    None    Dead      1 2002.181 2011.218 2002.962 2001.938 2002.622
12 2003.484 9.514031    None    None     12 2003.484       NA 2003.543 2004.223 2004.012
> subset( LK, lex.id %in% idx )[,c(2,4:7,12,17:20)]
          per   lex.dur lex.Cst lex.Xst lex.id    doEnt    doDth      doR      doK
4023 2002.181 9.0376454       K Dead(K)      1 2002.181 2011.218 2002.962 2001.938
12   2003.484 0.7388465    None       K     12 2003.484       NA 2003.543 2004.223
4034 2004.223 8.7751850       K       K     12 2003.484       NA 2003.543 2004.223
          doN
4023 2002.622
12   2004.012
4034 2004.012
> subset( LN, lex.id %in% idx )[,c(2,4:7,12,17:20)]
          per   lex.dur lex.Cst lex.Xst lex.id    doEnt    doDth      doR      doK
1    2002.181 0.4410307    None       N      1 2002.181 2011.218 2002.962 2001.938
4023 2002.622 8.5966147       N Dead(N)      1 2002.181 2011.218 2002.962 2001.938
12   2003.484 0.5280155    None       N     12 2003.484       NA 2003.543 2004.223
4034 2004.012 8.9860160       N       N     12 2003.484       NA 2003.543 2004.223
          doN
1    2002.622
4023 2002.622
12   2004.012
4034 2004.012
> subset( LKN, lex.id %in% idx )[,c(2,4:7,12,17:20)]
          per   lex.dur lex.Cst    lex.Xst lex.id    doEnt    doDth      doR      doK
1    2002.181 0.4410307       K          N      1 2002.181 2011.218 2002.962 2001.938
5445 2002.622 8.5966147       N Dead(K)(N)      1 2002.181 2011.218 2002.962 2001.938
16   2003.484 0.5280155    None          N     12 2003.484       NA 2003.543 2004.223
5460 2004.012 0.2108309       N          N     12 2003.484       NA 2003.543 2004.223
5461 2004.223 8.7751850       N          N     12 2003.484       NA 2003.543 2004.223
          doN
1    2002.622
5445 2002.622
16   2004.012
5460 2004.012
5461 2004.012
```

```
> subset( LNK, lex.id %in% idx )[,c(2,4:7,12,17:20)]
          per   lex.dur lex.Cst    lex.Xst lex.id    doEnt    doDth      doR      doK
5181 2002.181 0.4410307       K          K      1 2002.181 2011.218 2002.962 2001.938
5182 2002.622 8.5966147       K Dead(N)(K)      1 2002.181 2011.218 2002.962 2001.938
18   2003.484 0.5280155    None          N     12 2003.484       NA 2003.543 2004.223
19   2004.012 0.2108309       N          K     12 2003.484       NA 2003.543 2004.223
5199 2004.223 8.7751850       K          K     12 2003.484       NA 2003.543 2004.223
          doN
5181 2002.622
5182 2002.622
18   2004.012
19   2004.012
5199 2004.012
```

### 1.2.2 Restriction to a known ordering of events

But let us now do it in reverse order, where we split on the latest occurring event first:

```
> S3 <- subset( L0, doK < doN & doN < doR )
> summary( S3 )
Transitions:
     To
From   None Dead  Records:  Events: Risk time:  Persons:
  None  104   24       128       24    1052.38       128

> LR <- cutLexis( S3, cut = S3$doR,
+                     time = "per",
+                new.state = "R",
+                      pre = "None",
+                    split = TRUE )
> summary( LR )
Transitions:
     To
From   None   R Dead Dead(R)  Records:  Events: Risk time:  Persons:
  None    4 113    1       0       118      114     293.31       118
  R       0 100    0      23       123       23     759.06       123
  Sum     4 213    1      23       241      137    1052.38       128
> LNR <- cutLexis( LR, cut = LR$doN,
+                      time = "per",
+                 new.state = "N",
+                       pre = "None",
+                     split = TRUE )
> summary( LNR )
Transitions:
     To
From   None  N R Dead Dead(R) R(N) Dead(R)(N) Dead(N)  Records:  Events: Risk time:
  None    2 79 0    0       0    0          0       0        81       79     114.83
  N       0  2 0    0       0  213         23       1       239      237     937.55
  Sum     2 81 0    0       0  213         23       1       320      316    1052.38

Transitions:
     To
From    Persons:
  None        81
  N          126
  Sum        128
```

```
> LKNR <- cutLexis( LNR, cut = LNR$doK,
+                       time = "per",
+                  new.state = "K",
+                        pre = "None",
+                      split = TRUE )
> summary( LKNR )
Transitions:
     To
From   None  K N R Dead Dead(R) R(N) Dead(R)(N) Dead(N) N(K) R(N)(K) Dead(R)(N)(K)
  None    0 36 0 0    0       0    0          0       0    0       0             0
  K       0  2 0 0    0       0    0          0       0   81     213            23
  Sum     0 38 0 0    0       0    0          0       0   81     213            23

Transitions:
     To
From   Dead(N)(K)  Records:  Events: Risk time:  Persons:
  None          0        36       36      33.38        36
  K             1       320      318    1019.00       128
  Sum           1       356      354    1052.38       128
> table(tt <- table(LKNR$lex.id) )
 1  2  3  4
12 37 46 33
> idx <- names( tt[tt==4][1:2] )
> subset( LKNR, lex.id %in% idx )[,c(2,4:7,12,17,19,20,18)]
         per    lex.dur lex.Cst lex.Xst lex.id    doEnt    doDth      doK      doN
4   2002.627 0.03727905    None       K     11 2002.627 2008.045 2002.664 2003.585
324 2002.664 0.92086102       K    N(K)     11 2002.627 2008.045 2002.664 2003.585
325 2003.585 0.86771901       K R(N)(K)     11 2002.627 2008.045 2002.664 2003.585
326 2004.453 2.31650917       K R(N)(K)     11 2002.627 2008.045 2002.664 2003.585
12  2003.054 5.71250333    None       K     68 2003.054       NA 2008.767 2010.886
332 2008.767 2.11932055       K    N(K)     68 2003.054       NA 2008.767 2010.886
333 2010.886 1.59406825       K R(N)(K)     68 2003.054       NA 2008.767 2010.886
334 2012.480 0.51798193       K R(N)(K)     68 2003.054       NA 2008.767 2010.886
         doR
4   2004.453
324 2004.453
325 2004.453
326 2004.453
12  2012.480
332 2012.480
333 2012.480
334 2012.480
```

This is — sort of — ok if we make sure states appear in a well defined order, but the `split=` argument does not work properly; it does not update the `lex.Cst`. A clear bug in `cutLexis`.

### 1.2.3 Splitting explicitly for a known sequence of events

But here is a workaround, using hand-updating of states:

```
> LK <- cutLexis( S3, cut = S3$doK,
+                     time = "per",
+                new.state = "K",
+                      pre = "None" )
> summary( LK )
```

```
Transitions:
     To
From   None   K Dead  Records:  Events: Risk time:  Persons:
  None    0  36    0        36       36      33.38        36
  K       0 104   24       128       24    1019.00       128
  Sum     0 140   24       164       60    1052.38       128
> LKN <- cutLexis( LK, cut = LK$doN,
+                     time = "per",
+                new.state = "K-N",
+                      pre = c("None","K") )
> summary( LKN )
Transitions:
     To
From   None  K K-N Dead  Records:  Events: Risk time:  Persons:
  None    0 36   0    0        36       36      33.38        36
  K       0  2  79    0        81       79      81.45        81
  K-N     0  0 102   24       126       24     937.55       126
  Sum     0 38 181   24       243      139    1052.38       128
> LKNR <- cutLexis( LKN, cut = LKN$doR,
+                     time = "per",
+                new.state = "K-N-R",
+                      pre = c("None","K","K-N") )
> summary( LKNR )
Transitions:
     To
From    None  K K-N K-N-R Dead  Records:  Events: Risk time:  Persons:
  None     0 36   0     0    0        36       36      33.38        36
  K        0  2  79     0    0        81       79      81.45        81
  K-N      0  0   2   113    1       116      114     178.49       116
  K-N-R    0  0   0   100   23       123       23     759.06       123
  Sum      0 38  81   213   24       356      252    1052.38       128
> table(tt <- table(LKNR$lex.id) )
 1  2  3  4
12 37 46 33
> idx <- names( tt[tt==4][1:2] )
> subset( LKNR, lex.id %in% idx )[,c(2,4:7,12,17,19,20,18)]
         per    lex.dur lex.Cst lex.Xst lex.id    doEnt    doDth      doK      doN
3   2002.627 0.03727905    None       K     11 2002.627 2008.045 2002.664 2003.585
4   2002.664 0.92086102       K     K-N     11 2002.627 2008.045 2002.664 2003.585
5   2003.585 0.86771901     K-N   K-N-R     11 2002.627 2008.045 2002.664 2003.585
248 2004.453 2.31650917   K-N-R   K-N-R     11 2002.627 2008.045 2002.664 2003.585
9   2003.054 5.71250333    None       K     68 2003.054       NA 2008.767 2010.886
10  2008.767 2.11932055       K     K-N     68 2003.054       NA 2008.767 2010.886
11  2010.886 1.59406825     K-N   K-N-R     68 2003.054       NA 2008.767 2010.886
254 2012.480 0.51798193   K-N-R   K-N-R     68 2003.054       NA 2008.767 2010.886
         doR
3   2004.453
4   2004.453
5   2004.453
248 2004.453
9   2012.480
10  2012.480
11  2012.480
254 2012.480
```

This apparently works so we just need to hammer it out it in general terms, that is split the original dataset by the actually occurring sequences of events in the dataset — note that we must expand the number of precursor states as we go along too.

### 1.2.4 Generalizing

For more generality we need an order function that gives the ordering but excludes NAs:

```
> NAorder <-
+ function( x )
+     {
+     oo <- order( x, na.last=T )
+     oo[is.na(x)] <- NA
+     paste( oo[!is.na(oo)], collapse="-" )
+     }
```

We can use this to extract the sequences of the dates in `L0`, and define the names of the states

```
> wh <- 18:20
> ( stnam <- gsub( "do", "", names(L0)[wh] ) )
[1] "R" "K" "N"

> whseq <- apply( L0[,wh], 1, NAorder )
> table( whseq )

whseq
          1   1-2 1-2-3 1-3-2     2   2-1 2-1-3 2-3-1   3-1 3-1-2   3-2 3-2-1
  833   858   530   234   240   123   309   157   128    55   243   172   140

> unique( whseq )

 [1] "2-3-1" ""      "2"     "1-2-3" "2-1-3" "1"     "1-3-2" "1-2"   "2-1"   "3-1"
[11] "3-1-2" "3-2-1" "3-2"
```

Once we have these unique sequences, we can run through them, for each select the relevant set of persons, and cut their follow-up time according to the order in which the events (complications) occur.

## 1.3 The `mcutLexis` function

These tricks are all implemented in the `mcutLexis` function: which has the following help page:

---

`mcutLexis` *Cut follow-up at multiple event dates and keep track of order of events*

---

### Description

A generalization of `cutLexis` to the case where different events may occur in any order.

### Usage

```
mcutLexis( L0, timescale = 1, wh,
              new.states = NULL,
        precursor.states = NULL,
              seq.states = TRUE,
              new.scales = NULL,
            ties.resolve = FALSE )
```

## Arguments

| | |
|---|---|
| L0 | A Lexis object. |
| timescale | Which time scale do the variables in L0[,wh] refer to. Can be character or integer. |
| wh | Which variables contain the event dates. Character or integer vector |
| new.states | Names of the events forming new states. If NULL equal to the variable names from wh. |
| precursor.states | Which states are precursor states. See cutLexis for definition of precursor states. |
| seq.states | Should the sequence of events be kept track of? That is, should A-B be considered different from B-A. If FALSE, the state with combined preceding events A and B will be called A+B. |
| new.scales | Should we construct new time scales indicating the time since each of the event occurrences. |
| ties.resolve | Should tied event times be resolved by adding random noise to tied event dates. If FALSE the function will not accept that two events occur at the same time for a person (ties). If TRUE a random quantity in the range c(-1,1)/100 will be added to all event times in all records with at least one tie. If numeric a random quantity in the range c(-1,1)*ties.resolve will be added to all event times in all records with at least one tie. |

## Value

A Lexis object with extra states created by occurrence of a number of intermediate events.

## Author(s)

Bendix Carstensen, http://BendixCarstensen.com

## See Also

cutLexis, Lexis, splitLexis

## Examples

```
# A dataframe of times
set.seed(563248)
dd <- data.frame( id = 1:10,
                  doN = round(runif(10,-30, 0),1),
                  doE = round(runif(10,  0,20),1),
                  doX = round(runif(10, 50,60),1),
                  doD = round(runif(10, 50,60),1),
                  # these are the event times
                  doA = c(NA,20,NA,27,35, NA,52, 5,43,80),
                  doB = c(25,NA,37,40,NA, NA,15,23,36,61) )
Lx <- Lexis( entry = list(time=doE,
                          age=doE-doN),
              exit = list(time=pmin(doX,doD)),
       exit.status = factor(doD<doX,labels=c("OK","D")),
              data = dd )
summary( Lx )
L2 <- mcutLexis( Lx, "time", wh=c("doA","doB"),
                 new.states = c("A","B"),
```

```
              precursor.states = "OK",
                    seq.states = TRUE,
                    new.scales = c("tfA","tfB") )
   summary( L2 )
   L2
   boxes( L2, boxpos=list(x=c(10,50,50,90,50,90),
                          y=c(50,90,50,90,10,10)),
              show.R=FALSE, show.BE=TRUE )

   L3 <- mcutLexis( Lx, "time", wh=c("doA","doB"),
                    new.states = c("A","B"),
              precursor.states = "OK",
                    seq.states = FALSE,
                    new.scales = c("tfA","tfB") )
   summary( L3 )
   boxes( L3, boxpos=list(x=c(10,50,50,90,50),
                          y=c(50,90,50,50,10)),
              show.R=FALSE, show.BE=TRUE )
```

— and the definition of the function is:

```
> mcutLexis
function( L0, # A Lexis object
       timescale = 1,    # the time scale referred to by L0[,wh]
              wh,        # indices/names of columns holding dates of state entries (events)
      new.states = NULL, # Names of the event types (states)
precursor.states = NULL,
      seq.states = TRUE, # Should state names reflect ordering of events
      new.scales = NULL, # Time-scales referring to time since
    ties.resolve = FALSE # Are tied event times accepted?
        )
{
### we rely on referring to the timescale and event time variables by name
if( is.numeric(timescale) ) timescale <- timeScales(L0)[timescale]
if( is.numeric(wh) ) wh <- names(L0)[wh]

### don't be silly
if( length(wh)==1 )
  stop( "mcutLexis not needed for one type of event - use cutLexis\n" )

### states
if( is.null(new.states) )
  {
  new.states <- wh
  cat( "NOTE: Name of new states set to\n", new.states )
  }
if( length(wh) != length(new.states) )
  stop( "wh and new.states must have same length, but lengths are",
        "wh:", length(wh), "and new.states:", length(new.states), "\n" )

### timescales
# either all or none
if( is.logical(new.scales) )
  if( any( new.scales ) )
  {
  new.scales <- paste( "tf", new.states, sep="" )
```

```
  cat( "NOTE: new.scales set to: ", new.scales, "\n" )
  }
if( is.character(new.scales) & length(new.scales) != length(wh) )
  {
  new.scales <- paste( "tf", new.states, sep="" )
  warning( "new.scales not of same length as wh. Set to: ",
           new.scales, "\n" )
  }
if( is.character(new.scales) & length(intersect(new.states,timeScales(L0))) )
  stop( "Names of new time scales must be different from names of timescales:\n",
        timeScales(L0) )

### Tied transition times untied
has.ties <- any( wh.tied <- apply( L0[,wh], 1,
                           function(x) any(diff(sort(x[!is.na(x)]))==0) ) )
if( has.ties & is.logical(ties.resolve) & !ties.resolve )
  stop( "Tied event times not allowed with ties.resolve=FALSE:\n",
        "there were", length(wh.tied), "records with tied event times.")
if( has.ties & is.logical(ties.resolve) & ties.resolve ) ties.resolve = 1/100
if( has.ties & is.numeric(ties.resolve) )
  {
  rnd <- L0[wh.tied,wh]*0
  rnd[,] <- runif(rnd,-1,1) * ties.resolve
  L0[wh.tied,wh] <- L0[wh.tied,wh] + rnd
  cat( "NOTE: ", length(wh.tied),
       "records with tied events times resolved.\n",
       "Results only reproducible if the seed for the random number generator is set.")
  }
# End of checks

# The object to return initiated as NULL
Lcut <- NULL

# Utility function returning sequences of ocurrences as paste of numbers
NAorder <-
function( x )
    {
    oo <- order( x, na.last=T )
    oo[is.na(x)] <-NA
    paste( oo[!is.na(oo)], collapse="-" )
    }

# where do the different sequences of events actually occur in data
L0$whseq <- apply( L0[,wh], 1, NAorder )

# Loop through the actually occurring orders of event occurrences
for( sq in unique(L0$whseq) )
{
# Persons with none of the events occurring transferred to result
if( sq=="" ) Lcut <- rbind( Lcut, L0[L0$whseq=="",] )
else {

# Extract the subset of persons with a given sequence of events
Ltmp <- L0[L0$whseq==sq,]

# The numerical sequence of states (refer to the elements of wh)
ost <- as.numeric( strsplit( sq, "-" )[[1]] )
nxst <- ""
prst <- precursor.states
```

```
for( cs in ost )
   {
 nxst <- ifelse( cs==ost[1],  new.states[cs],
                 paste( nxst, new.states[cs], sep="-" ) )
 Ltmp <- cutLexis( Ltmp, cut = Ltmp[,wh[cs]],
                   timescale = timescale,
                   new.state = nxst,
            precursor.states = prst )
 # include the created state among the precursor states for next cut
 prst <- c(prst,nxst)
   } # end of for loop through events in this sequence (cs)

# Attach it to the end of the Lexis object
Lcut <- rbind( Lcut, Ltmp )
     } # end of the else clause

} # end of for loop through sequences (sq)

# Do we want the sequences or just the unordered set of previous events
if( !seq.states )
  {
   lvl <- levels( Lcut )
  ulvl <- sapply( lapply( strsplit(lvl,"-"),
                          sort ),
                  paste,
                  collapse="+" )
  levels( Lcut$lex.Cst ) <-
  levels( Lcut$lex.Xst ) <- ulvl
  }

# Did we ask for timescales as time since events?
if( !is.null(new.scales) )
  {
  # insert columns for the new time scales
  Lcut <- Lcut[,c(rep("whseq",length(new.scales)),names(Lcut))]
  names( Lcut )[1:length(new.scales)] <- new.scales
  for( i in 1:length(wh) )
     Lcut[,i] <- ifelse( Lcut[,timescale] - Lcut[,wh[i]] < 0,
                         NA,
                         Lcut[,timescale] - Lcut[,wh[i]] )
  # set attributes
  attr( Lcut, "time.scales" ) <- c( attr( Lcut, "time.scales" ), new.scales )
  attr( Lcut, "time.since"  ) <- c( attr( Lcut, "time.since"  ), new.states )
  }

# return the cut object without the auxilary variable
rmcol <- grep( "whseq", names(Lcut) )
Lcut[,-rmcol]
  }
```

### 1.3.1 Illustration

So now we have `mcutLexis` to cut the initial Lexis object, showing how the arguments `wh`, `new.states` and NEW.STATES are parallel.

```
> Lx <- mcutLexis( L0,
+           timescale = "per",
```

```
+                  wh = c("doR","doK","doN"),
+           new.states = c(  "R",   "K",   "N"),
+           new.scales = c( "tR", "tK", "tN"),
+     precursor.states = "None",
+           seq.states = FALSE )
```

We should reorder the levels in a sensible way:

```
> levels( Lx )[c(1,2,6,9,7,3,8,4,5)]
[1] "None"  "K"     "R"     "N"     "K+R"   "K+N"   "N+R"   "K+N+R" "Dead"
> Lx <- Relevel( Lx, c(1,2,6,9,7,3,8,4,5) )
> summary( L0 )
Transitions:
     To
From   None Dead  Records:  Events: Risk time:  Persons:
  None 3695  327      4022      327   28560.65      4022
> summary( Lx, t=T )
Transitions:
     To
From    None   K    R   N  K+R K+N N+R K+N+R Dead  Records:  Events: Risk time:  Persons:
  None  1246 259  798 214    0   0   0     0   27      2544      1298    8370.09      2544
  K        0  72    0   0  306  79   0     0   11       468       396     875.07       468
  R        0   0  797   0  415   0 181     0   30      1423       626    6423.13      1423
  N        0   0    0 196    0  90 168     0   28       482       286    1738.08       482
  K+R      0   0    0   0  489   0   0   351   23       863       374    3481.50       863
  K+N      0   0    0   0    0   5   0   233    3       241       236     378.52       241
  N+R      0   0    0   0    0   0  16   440    5       461       445    1208.78       461
  K+N+R    0   0    0   0    0   0   0   874  200      1074       200    6085.49      1074
  Sum   1246 331 1595 410 1210 174 365  1898  327      7556      3861   28560.65      4022

Timescales:
  time.scale time.since
1        age
2        per
3        dur
4         tR          R
5         tK          K
6         tN          N
```

We want to plot the transitions too, including the number of persons starting and ending in each state:

```
> # pdf("bb.pdf",width=15, height=10)
> boxes( Lx, boxpos=list(x=  c(10,36,36,36,64,64,64,92    ,50),
+                        y=c(c(50,80,50,20,80,50,20,50)+15, 5) ),
+            show.BE=TRUE, scale.R=100,
+            col.arr=c("black","gray")[c(1,1,1,2,
+                                        1,1,2,
+                                        1,1,2,
+                                        1,1,2,
+                                          1,2,
+                                          1,2,
+                                          1,2,
+                                            2)])
> par( new=TRUE )
> boxes( Lx, boxpos=list(x=  c(10,36,36,36,64,64,64,92    , 5),
+                        y=c(c(50,80,50,20,80,50,20,50)+15,50) ),
+            show.BE=TRUE, scale.R=100, subset=1:8, col.bg="white" )
> # dev.off()
```

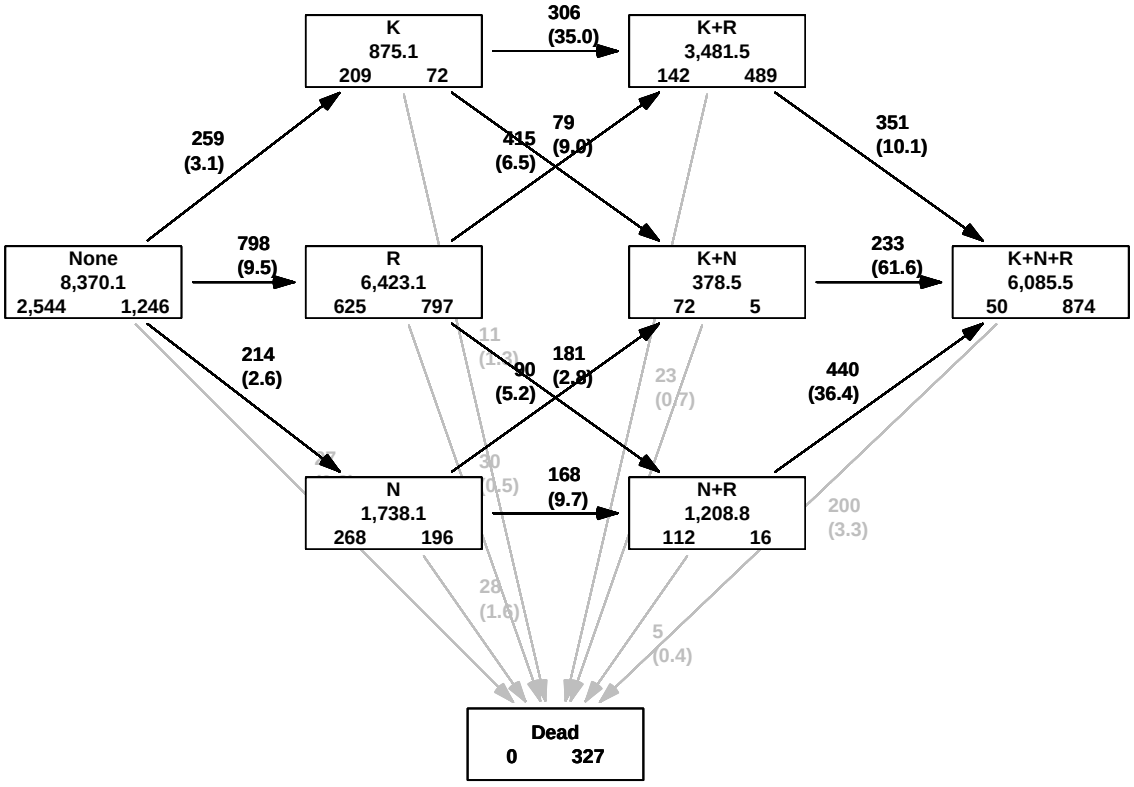


Figure 1.1: *Transitions between complications states. For clarity, the Dead state should really be split in two one for persons with 0 or 1 complications and one with 2 or 3.*

We could expand the "Dead" state according to whether (0 or 1) or (2 or 3) complications have occurred, to make the plot more legible in terms of the mortality rates.

```
> Lxx <- transform( Lx, lex.Xst = ifelse( lex.Xst=="Dead",
+                                         paste( lex.Cst, as.character(lex.Xst) ),
+                                         as.character(lex.Xst) ) )
> table( Lxx$lex.Xst, Lx$lex.Xst )
             None    K    R    N  K+R  K+N  N+R K+N+R Dead
  K             0  331    0    0    0    0    0     0    0
  K Dead        0    0    0    0    0    0    0     0   11
  K+N           0    0    0    0    0  174    0     0    0
  K+N Dead      0    0    0    0    0    0    0     0    3
  K+N+R         0    0    0    0    0    0    0  1898    0
  K+N+R Dead    0    0    0    0    0    0    0     0  200
  K+R           0    0    0    0 1210    0    0     0    0
  K+R Dead      0    0    0    0    0    0    0     0   23
  N             0    0    0  410    0    0    0     0    0
  N Dead        0    0    0    0    0    0    0     0   28
  None       1246    0    0    0    0    0    0     0    0
  None Dead     0    0    0    0    0    0    0     0   27
  N+R           0    0    0    0    0    0  365     0    0
  N+R Dead      0    0    0    0    0    0    0     0    5
  R             0    0 1595    0    0    0    0     0    0
  R Dead        0    0    0    0    0    0    0     0   30

> print( table( Lxx$lex.Xst, nchar(Lxx$lex.Xst) ), zero="." )
```

```
              1    3    4    5    6    8    9   10
  K         331    .    .    .    .    .    .    .
  K Dead      .    .    .    .   11    .    .    .
  K+N         .  174    .    .    .    .    .    .
  K+N Dead    .    .    .    .    .    3    .    .
  K+N+R       .    .    . 1898    .    .    .    .
  K+N+R Dead  .    .    .    .    .    .    .  200
  K+R         . 1210    .    .    .    .    .    .
  K+R Dead    .    .    .    .    .   23    .    .
  N         410    .    .    .    .    .    .    .
  N Dead      .    .    .    .   28    .    .    .
  None        .    . 1246    .    .    .    .    .
  None Dead   .    .    .    .    .    .   27    .
  N+R         .  365    .    .    .    .    .    .
  N+R Dead    .    .    .    .    .    5    .    .
  R        1595    .    .    .    .    .    .    .
  R Dead      .    .    .    .   30    .    .    .
> Lxx$lex.Xst <- ifelse( nchar(Lxx$lex.Xst)%in%c(6, 9), "Dead(0-1)", Lxx$lex.Xst )
> Lxx$lex.Xst <- ifelse( nchar(Lxx$lex.Xst)%in%c(8,10), "Dead(2-3)", Lxx$lex.Xst )
> Lxx <- Relevel( Lxx )
> summary( Lxx )
Transitions:
     To
From    None   K    R   N  K+R K+N N+R K+N+R Dead(0-1) Dead(2-3)  Records:  Events:
  None  1246 259  798 214    0   0   0     0        27          0      2544     1298
  K        0  72    0   0  306  79   0     0        11          0       468      396
  R        0   0  797   0  415   0 181     0        30          0      1423      626
  N        0   0    0 196    0  90 168     0        28          0       482      286
  K+R      0   0    0   0  489   0   0   351         0         23       863      374
  K+N      0   0    0   0    0   5   0   233         0          3       241      236
  N+R      0   0    0   0    0   0  16   440         0          5       461      445
  K+N+R    0   0    0   0    0   0   0   874         0        200      1074      200
  Sum   1246 331 1595 410 1210 174 365  1898        96        231      7556     3861

Transitions:
     To
From    Risk time:  Persons:
  None     8370.09      2544
  K         875.07       468
  R        6423.13      1423
  N        1738.08       482
  K+R      3481.50       863
  K+N       378.52       241
  N+R      1208.78       461
  K+N+R    6085.49      1074
  Sum     28560.65      4022
> # pdf("bbx.pdf",width=15, height=10)
> zz <-
+ boxes( Lxx, boxpos=list(x=  seq( 6,94,,4)[c(1,2,2,2,3,3,3,4,1,4)],
+                          y=c(seq(15,95,,3)[c(2,3:1,3:1,2)],5,5) ),
+            show.BE=TRUE, scale.R=100,
+            col.bg=rep(c("white","gray"),c(8,2)),
+            col.arr=c("black","gray")[c(1,1,1,2,
+                                        1,1,2,
+                                        1,1,2,
+                                        1,1,2,
+                                          1,2,
+                                          1,2,
```

```
+                                                  1,2,
+                                                    2)])
> par( new=TRUE )
> boxes( Lxx, boxpos=list(x=seq(6,94,,4)[c(1,2,2,2,3,3,3,4,1,4)],
+                         y=c(seq(15,95,,3)[c(2,3:1,3:1,2)],5,5) ),
+            show.BE=TRUE, scale.R=100, subset=1:8, col.bg="white" )
> zz

$Boxes
         xx yy       wd       ht font lwd col.txt col.border col.bg
1   6.00000 55 15.21772 9.309135    2   2   black      black  white
2  35.33333 95 15.21772 9.309135    2   2   black      black  white
3  35.33333 55 15.21772 9.309135    2   2   black      black  white
4  35.33333 15 15.21772 9.309135    2   2   black      black  white
5  64.66667 95 15.21772 9.309135    2   2   black      black  white
6  64.66667 55 15.21772 9.309135    2   2   black      black  white
7  64.66667 15 15.21772 9.309135    2   2   black      black  white
8  94.00000 55 15.21772 9.309135    2   2   black      black  white
9   6.00000  5 15.21772 9.309135    2   2   black      black   gray
10 94.00000  5 15.21772 9.309135    2   2   black      black   gray

$State.names
 [1] "None\n8,370.1\n2,544          1,246" "K\n875.1\n209          72"
 [3] "R\n6,423.1\n625          797"        "N\n1,738.1\n268          196"
 [5] "K+R\n3,481.5\n142          489"      "K+N\n378.5\n72          5"
 [7] "N+R\n1,208.8\n112          16"       "K+N+R\n6,085.5\n50          874"
 [9] "Dead(0-1)\n0          96"            "Dead(2-3)\n0          231"

$Tmat

           None   K   R   N K+R K+N N+R K+N+R Dead(0-1) Dead(2-3)
  None       NA 259 798 214  NA  NA  NA    NA        27        NA
  K          NA  NA  NA  NA 306  79  NA    NA        11        NA
  R          NA  NA  NA  NA 415  NA 181    NA        30        NA
  N          NA  NA  NA  NA  NA  90 168    NA        28        NA
  K+R        NA  NA  NA  NA  NA  NA  NA   351        NA        23
  K+N        NA  NA  NA  NA  NA  NA  NA   233        NA         3
  N+R        NA  NA  NA  NA  NA  NA  NA   440        NA         5
  K+N+R      NA  NA  NA  NA  NA  NA  NA    NA        NA       200
  Dead(0-1)  NA  NA  NA  NA  NA  NA  NA    NA        NA        NA
  Dead(2-3)  NA  NA  NA  NA  NA  NA  NA    NA        NA        NA

$Arrows
   lwd.arr col.arr pos.arr col.txt.arr font.arr offset.arr
1        2   black    0.45       black        2          2
2        2   black    0.45       black        2          2
3        2   black    0.45       black        2          2
4        2    gray    0.45        gray        2          2
5        2   black    0.45       black        2          2
6        2   black    0.45       black        2          2
7        2    gray    0.45        gray        2          2
8        2   black    0.45       black        2          2
9        2   black    0.45       black        2          2
10       2    gray    0.45        gray        2          2
11       2   black    0.45       black        2          2
12       2   black    0.45       black        2          2
13       2    gray    0.45        gray        2          2
14       2   black    0.45       black        2          2
15       2    gray    0.45        gray        2          2
```

```
16       2   black    0.45       black        2          2
17       2    gray    0.45        gray        2          2
18       2   black    0.45       black        2          2
19       2    gray    0.45        gray        2          2
20       2    gray    0.45        gray        2          2

$Arrowtext
 [1] "259\n(3.1)"   "798\n(9.5)"   "214\n(2.6)"   "27\n(0.3)"    "306\n(35.0)" "79\n(9.0)"
 [7] "11\n(1.3)"    "415\n(6.5)"   "181\n(2.8)"   "30\n(0.5)"    "90\n(5.2)"   "168\n(9.7)"
[13] "28\n(1.6)"    "351\n(10.1)" "23\n(0.7)"    "233\n(61.6)" "3\n(0.8)"     "440\n(36.4)"
[19] "5\n(0.4)"     "200\n(3.3)"

attr(,"class")
[1] "MS"
> # dev.off()
```

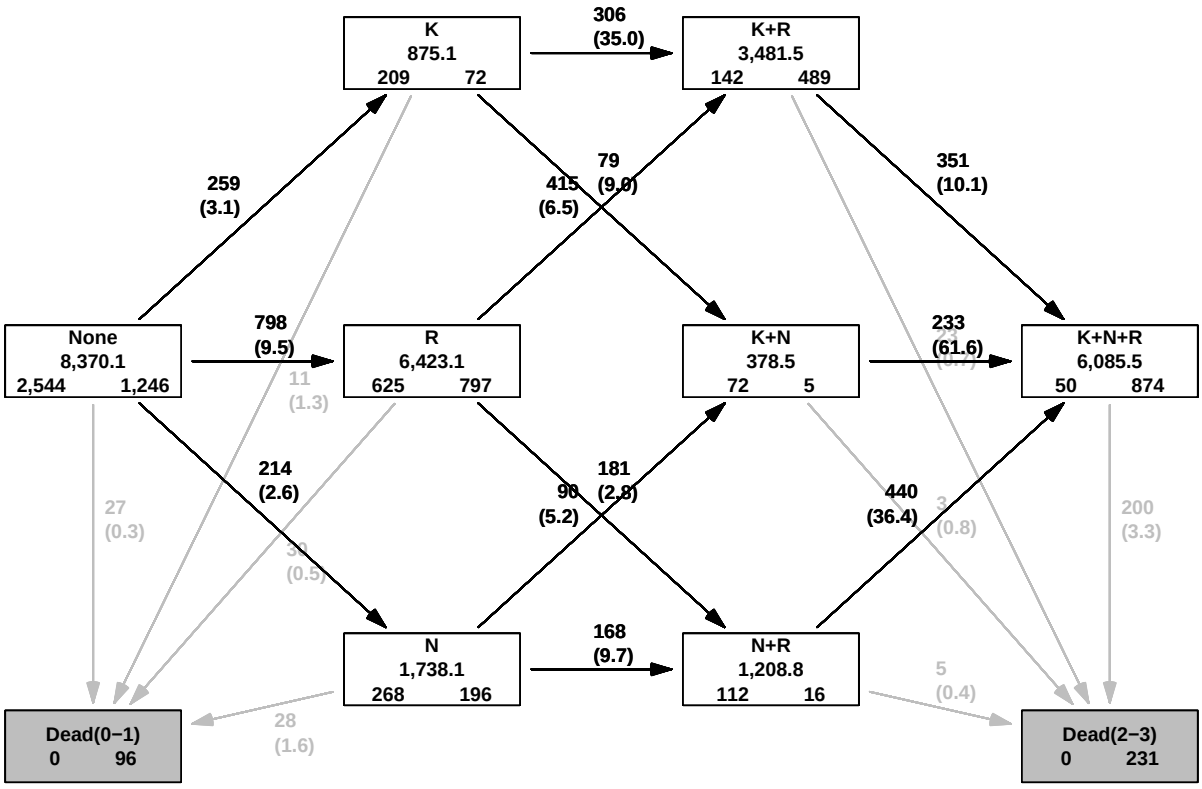


Figure 1.2: *Transitions between complications states. For clarity, the Dead state has been split.*

# Chapter 2

# Exercises

1. Try and see what happens if you run the code with the argument `seq.states=TRUE`
2. Modify the Lexis object so that there are 4 death states, dead with 0, 1, 2 or 3 complications. *Hint:* For the records with `lex.Xst=="Dead"` rename `lex.Xst` according to the corresponding value of `lex.Cst`.
3. Modify the plot to make it more legible w.r.t. mortality rates.
4. Put colors on arrows and rates, and move them along the arrows so that they can be read. *Hint:* Put the results of `boxes` in an object (which will be of class `MS`), print it, modify it and use the modified object as input to `boxes.MS`.
5. Split time using `splitLexis` in say 3-month intervals.
6. Fit relevant models for the rates, using `lex.Cst` as time-dependent covariate. Remember the outcome specification must also include `lex.Cst!=lex.Xst`, and not only specification of a range of values for `lex.Xst` — why?
7. Use `grep` to make an elegant construct of the outcome value for `lex.Xst` in analyses.
8. Put it all in a pdf document that will look the same wherever it is printed.