MS data from KR

Study Circle
December 2024
http://bendixcarstensen.com/PMM/KRex
Version 2

Compiled Monday 30th December, 2024, 11:20
from: C:\Bendix\teach\StudyCircle\MState\R/KRex.tex

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

Chapter 1

Reading and showing example data from KR

This initial chapter sets up data with relevant categorical variables as factors and follow-up converted to `Lexis` objects. Some sections address the same modeling issues as the book. The book is throughout referred to as "KR" (Kragh and Ravn—as the cover only understandable by Danish literates).

. . . now input from `pbc3.tex`

1.1 pbc3

```
> pbc3 <- read.csv("https://multi-state-book.github.io/companion/data/pbc3.csv",
+                  header = TRUE)
> pbc3 <- mutate(pbc3,
+                time = days / 365.25,
+              status = factor(status,
+                              levels = 0:2,
+                              labels = c("Alive","Trans","Dead")),
+               treat = factor(tment,
+                              levels = 0:1,
+                              labels = c("Placebo", "CyA")))
> str(pbc3)
'data.frame':        349 obs. of  12 variables:
 $ id    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ unit  : int  5 4 3 2 3 1 4 3 4 4 ...
 $ days  : int  1168 405 1735 241 754 1593 235 1332 1163 1423 ...
 $ status: Factor w/ 3 levels "Alive","Trans",..: 2 1 1 1 1 1 3 1 1 2 ...
 $ tment : int  1 1 0 1 1 0 1 1 1 0 ...
 $ sex   : int  0 1 1 0 0 0 1 0 0 0 ...
 $ age   : int  58 54 54 57 64 63 66 42 39 43 ...
 $ bili  : num  52 14.8 7 95.5 12 ...
 $ alb   : num  36 38 42.7 36 42.2 ...
 $ stage : int  4 2 2 4 3 2 4 2 NA NA ...
 $ time  : num  3.2 1.11 4.75 0.66 2.06 ...
 $ treat : Factor w/ 2 levels "Placebo","CyA": 2 2 1 2 2 1 2 2 2 1 ...
> save(pbc3, file = "../data/pbc3.Rda")
```

1.1.1 Lexis object

```
> Lx <- Lexis(exit = list(tfr = time),
+      exit.status = status,
+             data = pbc3)

NOTE: entry.status has been set to "Alive" for all.
NOTE: entry is assumed to be 0 on the tfr timescale.

> summary(Lx, by = "treat")

$Placebo

Transitions:
     To
From    Alive Trans Dead  Records:  Events: Risk time:  Persons:
  Alive   127    15   31       173       46     446.74       173

$CyA

Transitions:
     To
From    Alive Trans Dead  Records:  Events: Risk time:  Persons:
  Alive   132    14   30       176       44     453.98       176


> boxes(Lx, boxpos = TRUE, show.BE = TRUE, scale.R = 100)
```

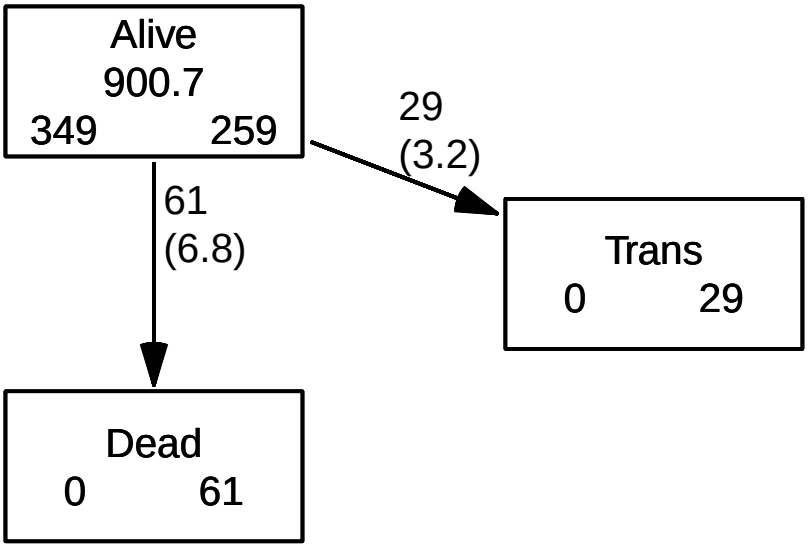

Figure 1.1: *Transitions in the* `pbc3` *trial* `../graph/pbc3-box3`

```
> c0 <- coxphLexis(Lx, tfr ~ treat)
```

```
NOTE:
Multiple transitions *from* state '  ' - are you sure?
The analysis requested is effectively merging outcome states.
You may want analyses using a *stacked* dataset - see ?stack.Lexis
survival::coxph analysis of Lexis object Lx:
Rates for transitions:
Alive->Trans
Alive->Dead
Baseline timescale: tfr
> ci.lin(c0)
            Estimate    StdErr          z         P       2.5%     97.5%
treatCyA -0.05873834 0.2109201 -0.2784862 0.7806392 -0.4721342 0.3546575
```

The parametric Poisson likelihood:

```
> Sx <- splitLexis(Lx, seq(0, 6, 0.1))
> summary(Lx)
Transitions:
     To
From    Alive Trans Dead  Records:  Events: Risk time:  Persons:
  Alive   259    29   61       349       90     900.71       349
> summary(Sx)
Transitions:
     To
From    Alive Trans Dead  Records:  Events: Risk time:  Persons:
  Alive  9088    29   61      9178       90     900.71       349
> tk <- c(0,1,3,5) # knots for spline
> p0 <- glmLexis(Sx, ~ Ns(tfr, knots = tk) + treat)
NOTE:
Multiple transitions *from* state ' Alive ' - are you sure?
The analysis requested is effectively merging outcome states.
You may want analyses using a *stacked* dataset - see ?stack.Lexis
stats::glm Poisson analysis of Lexis object Sx with log link:
Rates for transitions:
Alive->Trans
Alive->Dead
> ci.exp(p0)
                      exp(Est.)       2.5%      97.5%
(Intercept)          0.07604004 0.03927368  0.1472255
Ns(tfr, knots = tk)1 1.87660482 0.74182404  4.7472790
Ns(tfr, knots = tk)2 2.12378226 0.39897248 11.3051684
Ns(tfr, knots = tk)3 1.53250737 0.54487930  4.3102736
treatCyA             0.94331331 0.62394711  1.4261465
```

1.1.2 Mortality and survival

The mortality function can be derived from the `glmLexis` using `ci.pred` The survival function can be derived from the `glmLexis` using `ci.surv`

```
> nd <- data.frame(tfr = seq(0, 6, .2),
+                  treat = "CyA")
> nx <- mutate(nd, treat = "Placebo")
> Rc <- ci.pred(p0, nd)
> Rp <- ci.pred(p0, nx)
> Sp <- ci.surv(p0, nd)
```

```
NOTE: interval length chosen from  as tfr[2] - tfr[1]
> par(mfrow = c(1,2))
> matshade(nd$tfr, Rc * 100, lwd = 2, log = "y", plot = TRUE,
+          xlab = "Time from randomizations (years)",
+          ylab = "Event rate in CyA / placebo per 100PY")
> matshade(nd$tfr, Rp * 100, lwd = 2, col = "blue")
> matshade(nd$tfr, Sp, lwd = 2, ylim = 0:1, yaxs = "i", plot = TRUE,
+          xlab = "Time from randomizations (years)",
+          ylab = "Survival")
> lines(survfit(c0, data.frame(treat = "CyA")), col = "red")
> lines(survfit(c0, data.frame(treat = "CyA")), col = "red",
+       lwd = 2, conf.int = FALSE)
```

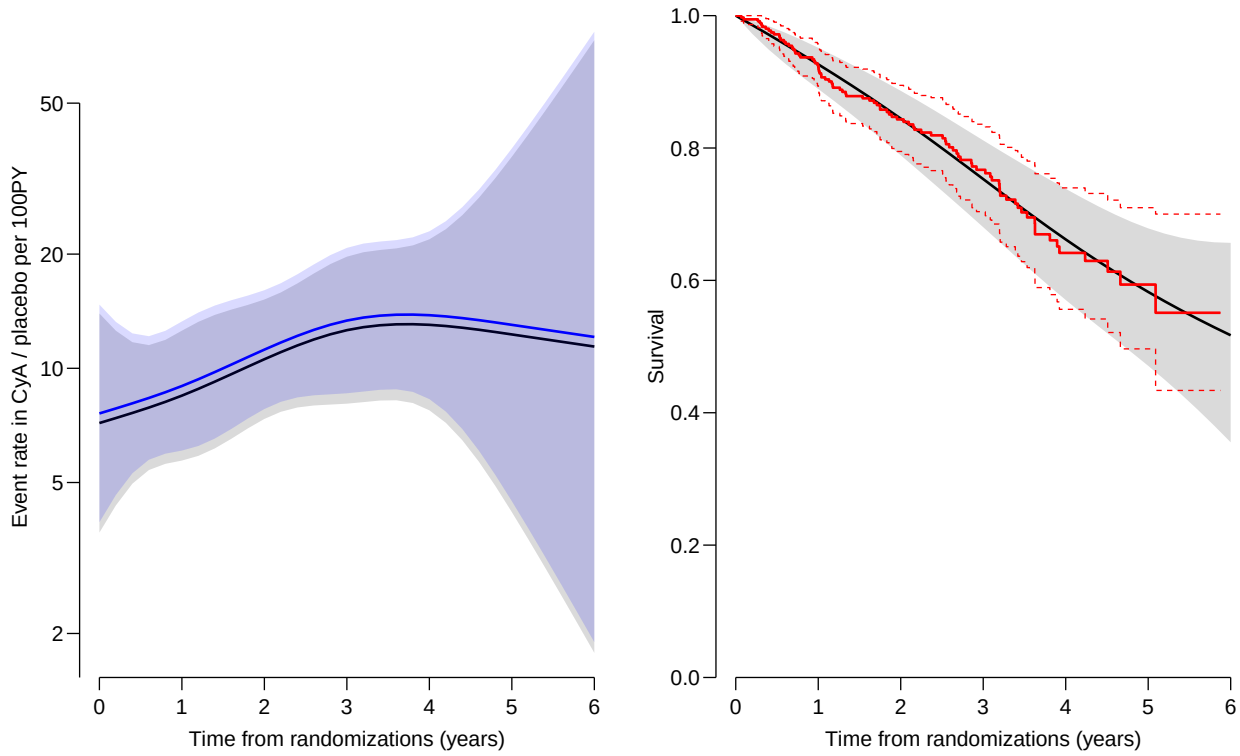

Figure 1.2: *Total event rates in the* `CyA` *(black) and* `Placebo` *(blue) arm of the PBC3 trial and survival function in the* `CyA` *arm. The red curve in the survival plot is the Breslow-estimator for the* `CyA` *arm from the Cox model. The shaded areas / broken lines indicate 95% confidence intervals.* `../graph/pbc3-mort`

It is not clear from the text where the hazard rates shown in figure 2.4 come from—they do not exist in smooth continuous form in the Cox model, and no hint is in the code on book website (figure 2.4 is not mentioned). The left part of figure 1.2 is what the lower part of figure 2.4 claims to be (note the absence of quantitative scales on the y-axis).

1.1.3 Models with covariates

We can now expand the model to include covariates, we fit (not quite) the same model in three guises: Cox, Poisson with natural splines based on the value of `tfr` in each 0.1-year intervals, and Poisson for 2-year intervals. We compare to the results in KR:

```
> c1 <- coxphLexis(Sx, tfr ~ treat + alb + I(bili / 100))
NOTE:
Multiple transitions *from* state '  ' - are you sure?
The analysis requested is effectively merging outcome states.
You may want analyses using a *stacked* dataset - see ?stack.Lexis
survival::coxph analysis of Lexis object Sx:
Rates for transitions:
Alive->Trans
Alive->Dead
Baseline timescale: tfr
```

```
> g1 <-   glmLexis(Sx, ~ 0 + cut(tfr, breaks = 0:3*2, right = FALSE)
+                          + treat + alb + I(bili / 100))
```

```
NOTE:
Multiple transitions *from* state ' Alive ' - are you sure?
The analysis requested is effectively merging outcome states.
You may want analyses using a *stacked* dataset - see ?stack.Lexis
stats::glm Poisson analysis of Lexis object Sx with log link:
Rates for transitions:
Alive->Trans
Alive->Dead
```

```
> p1 <-   glmLexis(Sx, ~ Ns(tfr, knots = tk)
+                          + treat + alb + I(bili / 100))
```

```
NOTE:
Multiple transitions *from* state ' Alive ' - are you sure?
The analysis requested is effectively merging outcome states.
You may want analyses using a *stacked* dataset - see ?stack.Lexis
stats::glm Poisson analysis of Lexis object Sx with log link:
Rates for transitions:
Alive->Trans
Alive->Dead
```

```
> round(ci.lin(c1)[,1:2], 3)
```

```
            Estimate StdErr
treatCyA      -0.497  0.226
alb           -0.116  0.021
I(bili/100)    0.895  0.098
```

```
> round(ci.lin(p1)[,1:2], 3)
```

```
                     Estimate StdErr
(Intercept)             0.818  0.867
Ns(tfr, knots = tk)1    1.503  0.498
Ns(tfr, knots = tk)2    2.182  0.922
Ns(tfr, knots = tk)3    0.923  0.563
treatCyA               -0.508  0.226
alb                    -0.116  0.021
I(bili/100)             0.905  0.098
```

```
> round(ci.lin(g1)[,1:2], 3)
```

```
                                                Estimate StdErr
cut(tfr, breaks = 0:3 * 2, right = FALSE)[0,2)    1.288  0.806
cut(tfr, breaks = 0:3 * 2, right = FALSE)[2,4)    2.146  0.833
cut(tfr, breaks = 0:3 * 2, right = FALSE)[4,6)    1.608  1.012
treatCyA                                         -0.475  0.224
alb                                              -0.112  0.021
I(bili/100)                                       0.846  0.094
```

We see that we get the same results is shown in KR tables 2.4 (Cox model `c1`) and 2.5 (Poisson model `g1`.

We can inspect the regression parameters (exponentiated) and the difference / ratio between the pairs of models:

```
> wh <- c("tre", "alb", "bil")
> round(cbind(ci.exp(c1, subset = wh),
+             ci.exp(p1, subset = wh),
+             ci.exp(g1, subset = wh)), 3)
             exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5%
treatCyA         0.609 0.391 0.947     0.601 0.386 0.936     0.622 0.401 0.965
alb              0.891 0.854 0.929     0.891 0.854 0.929     0.894 0.857 0.932
I(bili/100)      2.447 2.019 2.965     2.472 2.041 2.993     2.330 1.938 2.801

> # differences of log-HRs and ratios of HRs:
> round(cbind(ci.exp(c1, subset = wh, Exp = FALSE) -
+             ci.exp(p1, subset = wh, Exp = FALSE),
+             ci.exp(p1, subset = wh, Exp = FALSE) -
+             ci.exp(g1, subset = wh, Exp = FALSE)), 3)
             Estimate   2.5%  97.5% Estimate   2.5%  97.5%
treatCyA        0.012  0.012  0.012   -0.033 -0.036 -0.030
alb             0.000  0.001  0.000   -0.004 -0.004 -0.003
I(bili/100)    -0.010 -0.011 -0.009    0.059  0.052  0.066

> round(cbind(ci.exp(c1, subset = wh) /
+             ci.exp(p1, subset = wh),
+             ci.exp(p1, subset = wh) /
+             ci.exp(g1, subset = wh)), 3)
             exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5%
treatCyA         1.012 1.012 1.012     0.967 0.964 0.971
alb              1.000 1.001 1.000     0.996 0.996 0.997
I(bili/100)      0.990 0.989 0.991     1.061 1.053 1.068
```

The Cox model and the parametric Poisson model are closer than the parametric Poisson model and the piecewise Poisson model. The piecewise Poisson model is oversimplified and does not address the question of how mortality looks.

. . . now input from `bissau.tex`

1.2 bissau

```
> bissau <- read.csv("https://multi-state-book.github.io/companion/data/bissau.csv",
+                    header = TRUE)
> str(bissau)
'data.frame':        5274 obs. of  8 variables:
 $ id     : int  1 2 3 4 5 6 7 8 9 10 ...
 $ cluster: int  214 115 117 117 119 119 121 121 121 121 ...
 $ fuptime: int  65 161 166 166 161 161 166 166 166 166 ...
 $ dead   : int  1 0 0 0 0 0 0 0 0 0 ...
 $ bcg    : int  1 1 0 1 1 1 1 1 1 1 ...
 $ dtp    : int  1 2 0 0 0 0 2 1 2 2 ...
 $ sex    : int  NA 1 1 1 1 1 1 1 1 1 ...
 $ age    : int  182 125 69 96 131 26 129 90 119 146 ...
> save(bissau, file = "../data/bissau.Rda")
```

. . . now input from `testis.tex`

1.3 testis

```
> testis <- read.csv("https://multi-state-book.github.io/companion/data/testis.csv",
+                    header = TRUE)
> str(testis)
'data.frame':        237 obs. of  8 variables:
 $ age      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ pyrs     : num  25096.8 1859 64.2 21779.2 4972.1 ...
 $ cases    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ semi     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ nonsemi  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ parity   : int  1 2 3 1 2 3 4 1 2 3 ...
 $ cohort   : int  1950 1950 1950 1950 1950 1950 1950 1958 1958 1958 ...
 $ motherage: int  12 12 12 20 20 20 20 12 12 12 ...
> save(testis, file = "../data/testis.Rda")
```

. . . now input from `prova.tex`

1.4 prova

```
> prova <- read.csv("https://multi-state-book.github.io/companion/data/prova.csv",
+                   header = TRUE)
> prova <- mutate(prova, beta = factor(beta, labels = c("noBeta", "Beta")),
+                        scle = factor(scle, labels = c("noScle", "Scle")),
+                        # scale times to years
+                        timedeath = timedeath / 365.25,
+                        timebleed = timebleed / 365.25)
> str(prova)
'data.frame':        286 obs. of  12 variables:
 $ id       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ timedeath: num  3.743 0.183 3.751 0.945 0.895 ...
 $ death    : int  0 1 0 0 0 0 0 1 1 0 ...
 $ timebleed: num  NA NA NA NA NA ...
 $ bleed    : int  0 0 0 0 0 0 0 1 0 0 ...
 $ beta     : Factor w/ 2 levels "noBeta","Beta": 1 2 2 1 1 1 2 2 2 1 ...
 $ scle     : Factor w/ 2 levels "noScle","Scle": 1 2 2 2 1 2 2 1 2 2 ...
 $ sex      : int  1 1 1 1 0 1 1 0 1 1 ...
 $ age      : int  37 58 38 33 47 62 51 47 67 50 ...
 $ bili     : int  21 57 15 15 30 23 82 71 27 85 ...
 $ coag     : int  NA 50 81 95 66 73 34 44 76 67 ...
 $ varsize  : int  2 3 1 2 1 1 2 1 1 1 ...
> save(prova, file = "../data/prova.Rda")
> with(prova, table(beta, scle, exclude = NULL))
        scle
beta     noScle Scle
  noBeta     72   73
  Beta       68   73
```

We now set up a `Lexis` object that gives the possibility of both tabular abd graphical overview of data:

```
> Lx <- Lexis(exit = list(tfe = timedeath),
+             exit.status = factor(death, labels = c("noDis", "Dead")),
+             data = prova)
NOTE: entry.status has been set to "noDis" for all.
NOTE: entry is assumed to be 0 on the tfe timescale.
> Lx <- cutLexis(Lx, cut = Lx$timebleed,
+              new.state = "Bleed",
+            split.state = TRUE)
> Lx <- Relevel(Lx, c(1,2,4,3))
> summary.Lexis(Lx)
Transitions:
     To
From    noDis Bleed Dead(Bleed) Dead  Records:  Events: Risk time:  Persons:
  noDis   190    50           0   46       286       96     500.71       286
  Bleed     0    21          29    0        50       29      55.00        50
  Sum     190    71          29   46       336      125     555.71       286
```

```
> # summaries ordered to match table 1.2 in KR
> summary.Lexis(Lx, by = c("beta","scle"))[c(3,2,4,1)]
$noBeta.Scle

Transitions:
     To
From    noDis Bleed Dead(Bleed) Dead  Records:  Events: Risk time:  Persons:
  noDis    47    13           0   13        73       26     118.70        73
  Bleed     0     8           5    0        13        5      17.56        13
  Sum      47    21           5   13        86       31     136.27        73

$Beta.noScle

Transitions:
     To
From    noDis Bleed Dead(Bleed) Dead  Records:  Events: Risk time:  Persons:
  noDis    51    12           0    5        68       17     127.39        68
  Bleed     0     6           6    0        12        6      21.15        12
  Sum      51    18           6    5        80       23     148.55        68

$Beta.Scle

Transitions:
     To
From    noDis Bleed Dead(Bleed) Dead  Records:  Events: Risk time:  Persons:
  noDis    41    12           0   20        73       32     117.14        73
  Bleed     0     2          10    0        12       10       5.12        12
  Sum      41    14          10   20        85       42     122.25        73

$noBeta.noScle

Transitions:
     To
From    noDis Bleed Dead(Bleed) Dead  Records:  Events: Risk time:  Persons:
  noDis    51    13           0    8        72       21     137.48        72
  Bleed     0     5           8    0        13        8      11.17        13
  Sum      51    18           8    8        85       29     148.64        72
```

We see that the number of bleeds and deaths is in accordance with table, except for the dropout which are not in the data; there is no indication of the dropouts; we can only infer that 211 (=190 + 21) were alive at the end of FU, be that end of study or drop out. If drop outs were to be included we would need information on whether the total follow up (`timedeath`) ended in trial completion, drop out or death.

```
> boxes(Lx, boxpos = TRUE,
+           show.BE = TRUE,
+           scale.R = 100)
> legendbox(20, 95)


> par( mfrow = c(2,2))
> for (b in levels(Lx$beta))
+ for (s in levels(Lx$scle))
+ {
+ sL <- subset(Lx, beta == b & scle == s)
+ boxes(sL, boxpos = TRUE,
+           show.BE = TRUE,
```

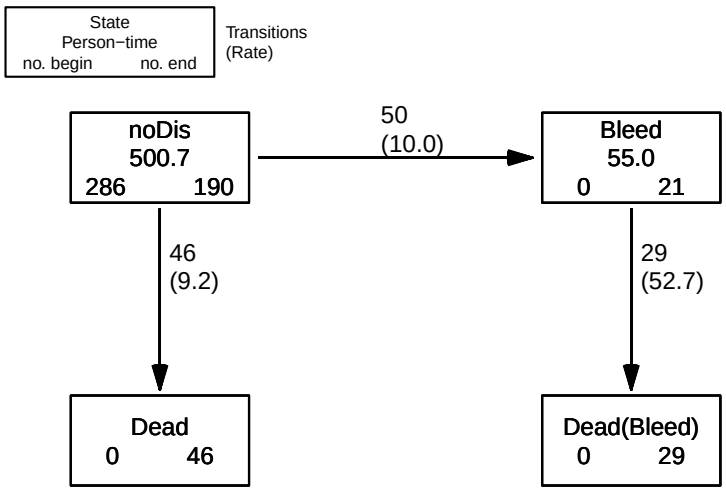

Figure 1.3: *The follow-up in the PROVA study with deaths and bleeding events. The drop-outs are not in the data. Risk time is in years, rates per 100 years.* ../graph/prova-provaboxes

```
+           scale.R = 100)
+ text(50, 95, paste0(b, ", ", s), cex = 2)
+ }
```

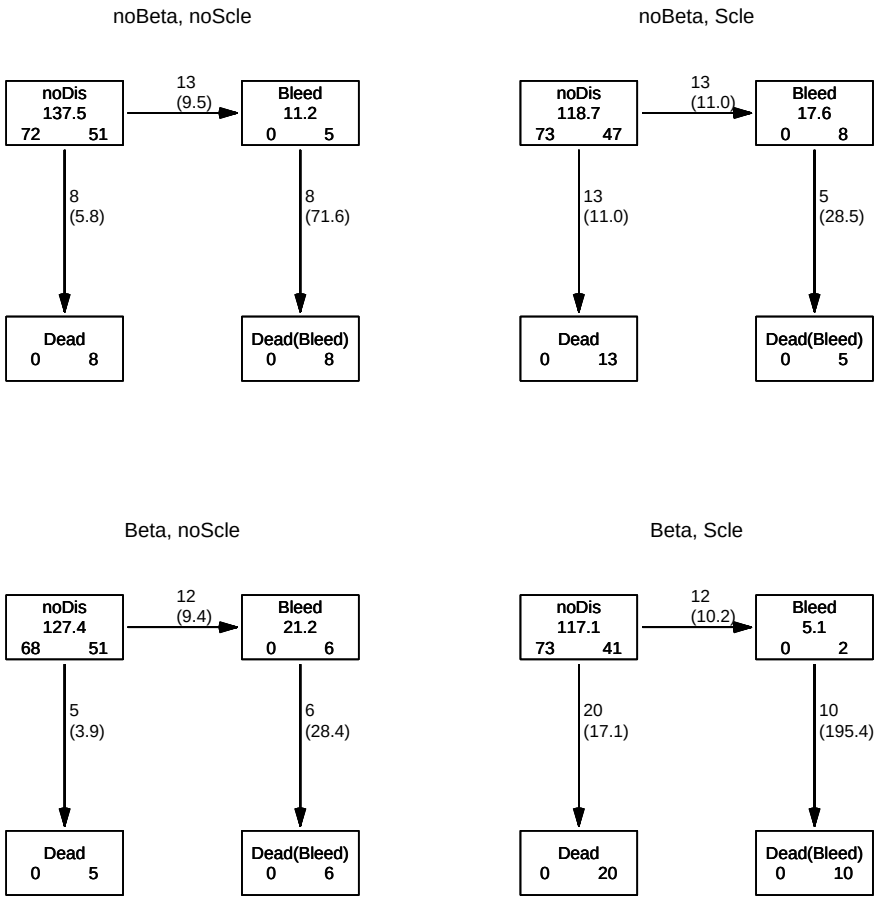

Figure 1.4: *The follow-up in the PROVA study with deaths and bleeding events, subdivided by randomization group. The drop-outs are not in the data. Risk time is in years, rates per 100 years.* `../graph/prova-provaboxes4`

. . . now input from affective.tex

1.5 affective

```
> affective <- read.csv("https://multi-state-book.github.io/companion/data/affective.csv",
+                   header = TRUE)
> str(affective)
'data.frame':        1287 obs. of  10 variables:
 $ id     : int  1 1 1 1 1 1 2 2 2 2 ...
 $ episode: int  1 1 2 2 3 3 1 1 2 2 ...
 $ state  : int  1 0 1 0 1 0 1 0 1 0 ...
 $ start  : num  0 1 148 150 160 162 0 4 8 14 ...
 $ stop   : num  1 148 150 160 162 243 4 8 14 53 ...
 $ status : int  0 1 0 1 0 2 0 1 0 1 ...
 $ bip    : int  1 1 1 1 1 1 0 0 0 0 ...
 $ sex    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ age    : int  65 65 65 65 65 65 72 72 72 72 ...
 $ year   : int  62 62 62 62 62 62 63 63 63 63 ...
> save(affective, file = "../data/affective.Rda")
```

. . . now input from `bmt.tex`

1.6 bmt

```
> bmt <- read.csv("https://multi-state-book.github.io/companion/data/bmt.csv",
+                 header = TRUE)
> str(bmt)
'data.frame':        2009 obs. of  14 variables:
 $ id         : int  1 2 3 4 5 6 7 8 9 10 ...
 $ team       : int  224 248 218 5 86 36 244 22 230 140 ...
 $ timedeath  : num  13.75 103.32 44.7 3.65 3.52 ...
 $ death      : int  1 0 0 1 1 0 0 0 1 1 ...
 $ timerel    : num  6.94 NA NA NA NA NA NA NA NA NA ...
 $ rel        : int  1 0 0 0 0 0 0 0 0 0 ...
 $ timegvhd   : num  NA NA NA 0.43 NA NA 0.3 NA NA 0.6 ...
 $ gvhd       : int  0 0 0 1 0 0 1 0 0 1 ...
 $ timeanc500: num  0.66 0.53 0.6 0.36 0.7 0.4 0.6 0.53 0.4 0.6 ...
 $ anc500     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ sex        : int  1 0 1 0 0 1 1 0 1 1 ...
 $ age        : num  9.51 25.86 49.18 53.62 45.02 ...
 $ all        : int  0 0 0 0 0 0 1 0 0 1 ...
 $ bmonly     : int  1 1 1 0 1 0 0 1 0 0 ...
> save(bmt, file = "../data/bmt.Rda")
```

. . . now input from holter.tex

```
> options(width = 90)
> par(mar = c(3,3,1,1),
+     mgp = c(3,1,0)/1.6,
+     las = 1,
+     bty = "n",
+    lend = "butt")
```

1.7 holter

First the paraphernalia:

```
     R   Epi   popEpi
 4.4.2  2.59   0.4.12
```

Then we read the data:

```
> holter <- read.csv("https://multi-state-book.github.io/companion/data/cphholter.csv",
+                    header = TRUE)
> str(holter)
'data.frame':        678 obs. of  17 variables:
 $ id        : int  1 2 3 4 5 6 7 8 9 10 ...
 $ timedeath : int  5396 5392 5373 5436 5352 5367 4184 5261 5288 3347 ...
 $ death     : int  0 0 0 0 0 0 1 0 0 1 ...
 $ timeafib  : int  NA NA NA NA NA NA NA NA 4282 NA ...
 $ afib      : int  0 0 0 0 0 0 0 0 1 0 ...
 $ timestroke: int  NA NA NA NA NA NA NA NA NA NA ...
 $ stroke    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ sex       : int  0 0 1 1 0 1 1 0 1 0 ...
 $ age       : int  70 65 60 60 75 55 70 65 55 75 ...
 $ smoker    : int  1 0 1 0 0 1 0 0 1 0 ...
 $ esvea     : int  0 0 0 0 1 0 0 0 0 1 ...
 $ chol      : num  5 5.4 6.8 6.5 6.5 4.4 7.1 6.1 5.1 6 ...
 $ diabet    : int  0 0 0 0 0 0 1 0 0 1 ...
 $ bmi       : num  31.2 26.8 23.1 22.9 23.3 ...
 $ aspirin   : int  1 0 0 0 1 0 1 0 0 0 ...
 $ probnp    : num  4.65 5.98 9.08 0.77 12.9 ...
 $ sbp       : int  160 190 155 120 140 155 170 190 180 180 ...
```

Convert times to years instead of days:

```
> (ts <- fgrep("time", names(holter)))
[1] "timedeath"  "timeafib"   "timestroke"
> holter[, ts] <- holter[, ts] / 365.25
```

A slightly more compact overview:

```
> with(holter, ftable(                afib, stroke, death, esvea))
```

```
                  esvea   0   1
afib stroke death
0    0      0           320  34
            1           158  32
     1      0            17   1
            1            25  14
1    0      0            29   8
            1            20   4
     1      0             7   1
            1             3   5
```

```
> with(holter, ftable(addmargins(table(afib, stroke, death, esvea))))
```

```
                  esvea   0   1 Sum
afib stroke death
0    0      0           320  34 354
            1           158  32 190
            Sum         478  66 544
     1      0            17   1  18
            1            25  14  39
            Sum          42  15  57
     Sum    0           337  35 372
            1           183  46 229
            Sum         520  81 601
1    0      0            29   8  37
            1            20   4  24
            Sum          49  12  61
     1      0             7   1   8
            1             3   5   8
            Sum          10   6  16
     Sum    0            36   9  45
            1            23   9  32
            Sum          59  18  77
Sum  0      0           349  42 391
            1           178  36 214
            Sum         527  78 605
     1      0            24   2  26
            1            28  19  47
            Sum          52  21  73
     Sum    0           373  44 417
            1           206  55 261
            Sum         579  99 678
```

We have a few ties of `Afib` and `Str` — we could arbitrarily assume that `Afib` precedes `Str` in such cases—the three cases where the two inequalities both are `FALSE` is where the two ties are equal:

```
> with(holter, ftable(death, esvea,
+                     "s>a" = timestroke > timeafib,
+                     "s<a" = timestroke < timeafib,
+                     exclude = NULL))
                  s<a FALSE TRUE  NA
death esvea s>a
0     0     FALSE         1    4   0
            TRUE          2    0   0
            NA            0    0 366
      1     FALSE         1    0   0
            TRUE          0    0   0
```

```
            NA            0    0  43
1     0     FALSE         1    1   0
            TRUE          1    0   0
            NA            0    0 203
      1     FALSE         0    2   0
            TRUE          3    0   0
            NA            0    0  50
```

```
> with(holter, ftable("s>a" = timestroke > timeafib,
+                     "s<a" = timestroke < timeafib,
+                     exclude = NULL))
```

```
      s<a FALSE TRUE  NA
s>a
FALSE         3    7   0
TRUE          6    0   0
NA            0    0 662
```

So we see that there are 16 $(3 + 7 + 6)$persons with both a `timeafib` and a `timestroke` and 3 of these have identical dates. For deaths occurring at time of stroke, the date of stroke will just be ignored and the date only counted as a date of death:

```
> (subset(holter, timeafib   == timestroke |
+                 timeafib   == timedeath  |
+                 timestroke == timedeath)[,c("id", ts)] -> sb)
```

```
     id timedeath  timeafib timestroke
12   12 14.576318 11.934292  11.934292
242 242  3.474333        NA   3.474333
267 267  5.431896        NA   5.431896
533 533  6.335387  6.198494   6.198494
550 550 14.715948  8.930869   8.930869
```

1.7.1 A `Lexis` data frame

We can now set up a `Lexis` data frame:

```
> Lho <- Lexis(exit = list(time = timedeath),
+       exit.status = factor(death, labels = c("Alive", "Dead")),
+               id = id,
+             data = holter)
```

```
NOTE: entry.status has been set to "Alive" for all.
NOTE: entry is assumed to be 0 on the time timescale.
```

```
> summary(Lho, t = T)
```

```
Transitions:
     To
From    Alive Dead  Records:  Events: Risk time:  Persons:
  Alive   417  261       678      261    8305.11       678

Timescales:
time
  ""
```

```
> range(Lho$lex.dur)
```

```
[1]  0.04654346 15.18685832
```

1.7.2 Cutting follow-up at intermediate events

Once we have set up the total follow-up we subdivide ("cut") at the times of the intermediate events using `mcutLexis`:

```
> set.seed(1952)
> Mhl <- mcutLexis(Lho,
+                  wh = c('timeafib', 'timestroke'),
+           new.states = c('Afib', 'Str'),
+           seq.states = TRUE,
+         ties.resolve = TRUE)
NOTE: Precursor states set to Alive
NOTE: 3 records with tied events times resolved (adding 0.01 random uniform),
      so results are only reproducible if the random number seed was set.
> Mhl <- Relevel(Mhl, c("Alive", "Afib", "Str", "Afib-Str", "Str-Afib", "Dead"))
> summary(Mhl)
Transitions:
     To
From       Alive Afib Str Afib-Str Str-Afib Dead  Records:  Events: Risk time:  Persons:
  Alive      354   68  64        0        0  192       678      324    7748.02       678
  Afib         0   37   0        7        0   24        68       31     247.17        68
  Str          0    0  18        0        9   37        64       46     284.54        64
  Afib-Str     0    0   0        2        0    5         7        5      10.07         7
  Str-Afib     0    0   0        0        6    3         9        3      15.31         9
  Sum        354  105  82        9       15  261       826      409    8305.11       678
```

Paths traveled

We can derive the path for each person using `paths.Lexis`—an S3 method (only from `Epi 2.59`):

```
> cbind(table(pl <- paths.Lexis(Mhl)))
                            [,1]
Alive                        354
Alive->Afib                   37
Alive->Afib->Afib-Str          2
Alive->Afib->Afib-Str->Dead    5
Alive->Afib->Dead             24
Alive->Dead                  192
Alive->Str                    18
Alive->Str->Dead              37
Alive->Str->Str-Afib           6
Alive->Str->Str-Afib->Dead     3
```

but if we want it subdivided by `esvea` state we must merge the paths to the `holter` data frame. This requires the paths as a data frame:

```
> dfp <- data.frame(id = as.numeric(names(pl)), path = pl)
> dfp <- left_join(dfp, holter[,c("id", "esvea")])
> str(dfp)
'data.frame':        678 obs. of  3 variables:
 $ id   : num  1 2 3 4 5 6 7 8 9 10 ...
 $ path : Factor w/ 10 levels "Alive","Alive->Afib",..: 1 1 1 1 1 1 6 1 2 6 ...
 $ esvea: int  0 0 0 0 1 0 0 0 0 1 ...
```

```
> with(dfp, addmargins(table(path, esvea)))
```

```
                              esvea
path                            0   1 Sum
  Alive                       320  34 354
  Alive->Afib                  29   8  37
  Alive->Afib->Afib-Str         2   0   2
  Alive->Afib->Afib-Str->Dead   2   3   5
  Alive->Afib->Dead            20   4  24
  Alive->Dead                 160  32 192
  Alive->Str                   17   1  18
  Alive->Str->Dead             23  14  37
  Alive->Str->Str-Afib          5   1   6
  Alive->Str->Str-Afib->Dead    1   2   3
  Sum                         579  99 678
```

This is clearly not an exact reproduction of the table 1.5 in KR, owing to the arbitrary sequencing of `Afib` and `Str` where noted as coincident in time.

Visualizing transitions

We can then visualize the follow-up in two different ways; either keeping track of the *order* of occurrence of `Afib` and `Str`, or lumping the two states together, shown in the two figures 1.5 and 1.6:

```
> boxes(Mhl, boxpos = list(x = c(15, 85, 15, 85, 55, 50),
+                          y = c(85, 85, 15, 45, 15, 50)),
+           scale.R = 100,
+           show.BE = TRUE)
> legendbox(80, 20)
```

```
> Mho <- Relevel(Mhl, list(1, 2, 3, "Afib+Str" = 4:5, 6))
> boxes(Mho, boxpos = list(x = c(15, 85, 15, 85, 50),
+                          y = c(85, 85, 15, 15, 50)),
+           scale.R = 100,
+           show.BE = TRUE)
```

However, the graph in KR figure 1.7 lumps the states with occurrence of both `Afib` and `Str` to the state of the *last* occurring event (not clear where that assumption comes from):

```
> levels(Mhl)
```

```
[1] "Alive"    "Afib"     "Str"      "Afib-Str" "Str-Afib" "Dead"
```

```
> Mr <- Relevel(Mhl, list(1, Afib = c(2,5), Str = c(3,4), 6))
> summary(Mr)
```

```
Transitions:
     To
From    Alive Afib Str Dead  Records:  Events: Risk time:  Persons:
  Alive   354   68  64  192       678      324    7748.02       678
  Afib      0   43   7   27        77       34     262.49        77
  Str       0    9  20   42        71       51     294.61        71
  Sum     354  120  91  261       826      409    8305.11       678
```

We can now mimic the KR Figure 1.7:

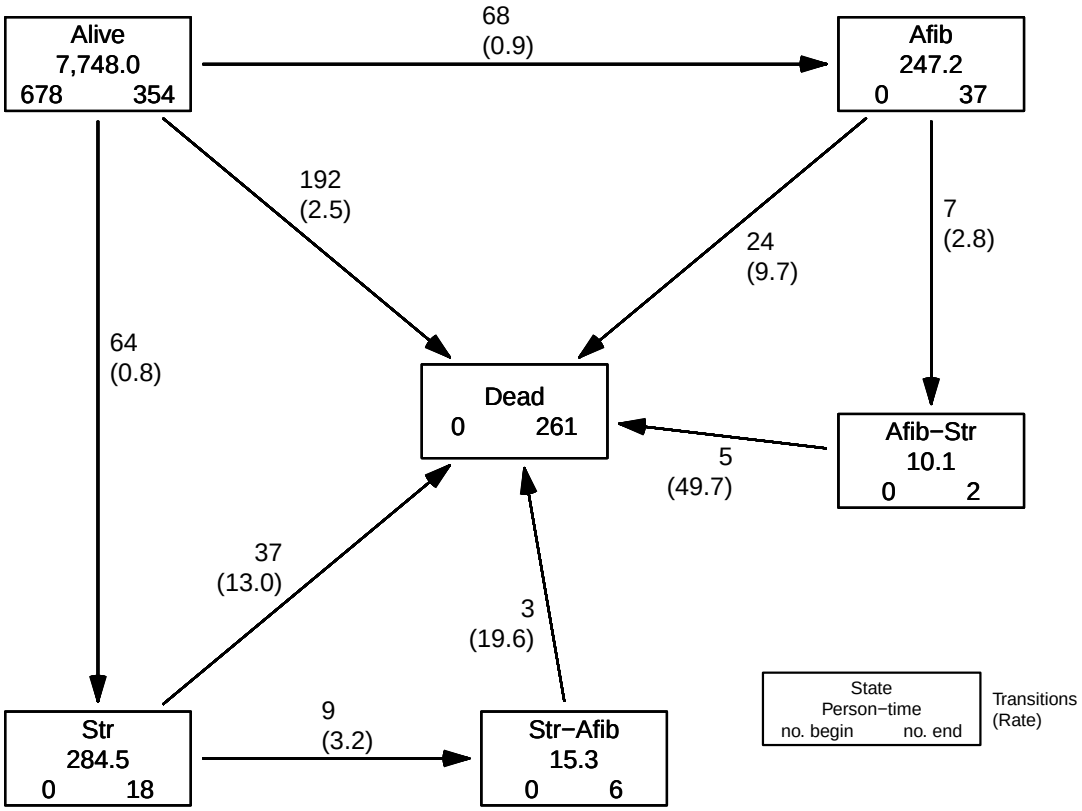

Figure 1.5: *The transitions in the multistate model, where the order of* `Afib` *and* `Str` *is represented (bar the inaccuracy in data with identical times of* `Afib` *and* `Str`*.../graph/holter-boxes6*

```
> boxes(Mr, boxpos = list(x = c(15, 85, 15, 85),
+                         y = c(85, 85, 15, 15)),
+           scale.R = 100,
+           show.BE = TRUE,
+           pos.arr = 0.25)
```

...but we can avoid crossing intensity arrows by placing `Dead` in the middle:

```
> boxes(Mr, boxpos = list(x = c(10, 90, 90, 60),
+                         y = c(50, 90, 10, 50)),
+           scale.R = 100,
+           show.BE = TRUE,
+           pos.arr = 0.35)
```

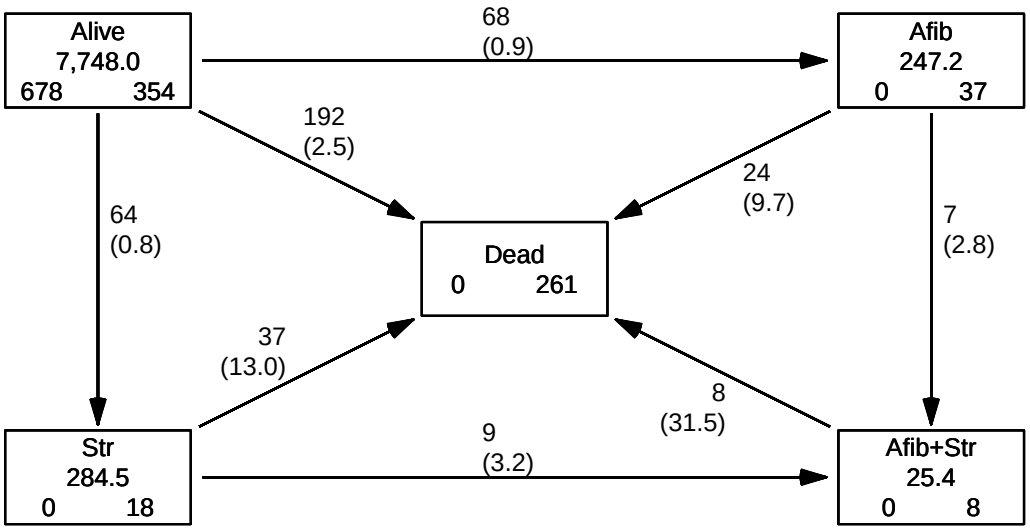

Figure 1.6: *The transitions in the multistate model, where the order of* `Afib` *and* `Str` *is ignored.* `../graph/holter-boxes5`

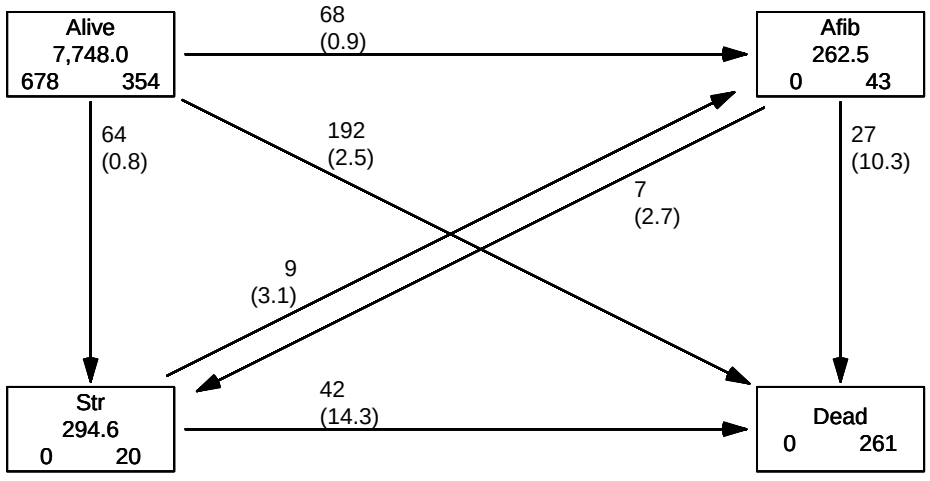

Figure 1.7: *The transitions in the multistate model, where the order of* `Afib` *and* `Str` *is mapped to the last state assumed, placed as figure 1.7.* `../graph/holter-boxes1-7`

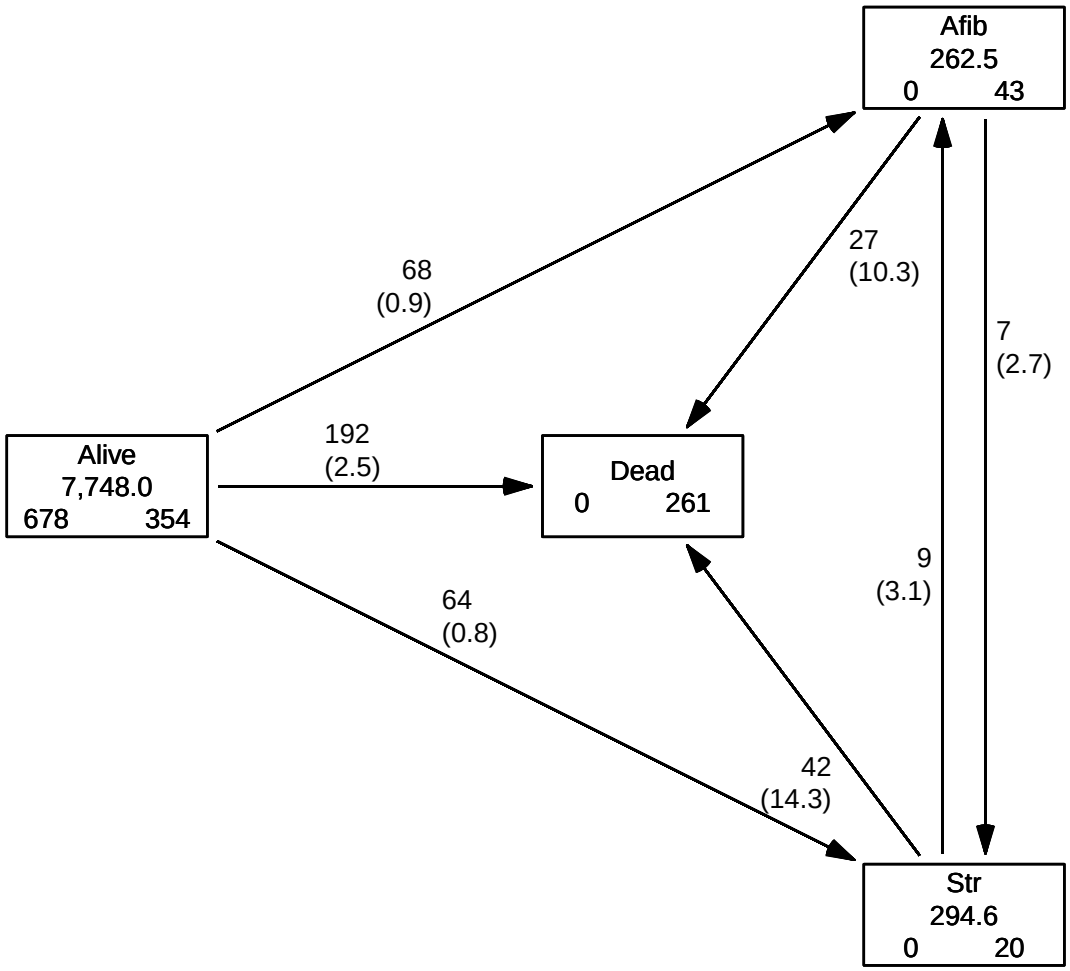

Figure 1.8: *The transitions in the multistate model, where the order of* `Afib` *and* `Str` *is mapped to the last state assumed, placed to avoid crossing transitions* `../graph/holter-boxes4`

1.7.3 Modeling of transition rates

We initially split the follow-up before drug inception in intervals of 1 month (that is 1/12 year), creating a `Lexis` object for a competing risks situation with three possible event types:

```
> Sho <- splitLexis(Mho, breaks = seq(0, 20, 1/12))
> summary(Sho)
Transitions:
     To
From       Alive Afib  Str Afib+Str Dead  Records:  Events: Risk time:  Persons:
  Alive    92994   68   64        0  192     93318      324    7748.02       678
  Afib         0 3007    0        7   24      3038       31     247.17        68
  Str          0    0 3428        9   37      3474       46     284.54        64
  Afib+Str     0    0    0      310    8       318        8      25.38        16
  Sum      92994 3075 3492      326  261    100148      409    8305.11       678
```

We can easily model the 4 mortality rates by a proportional hazards model, and test whether the HRs are all 1:

```
> ps <- glmLexis(Sho, ~ Ns(time, knots = c(0,2,6,12)) + lex.Cst)
stats::glm Poisson analysis of Lexis object Sho with log link:
Rates for transitions:
Alive->Dead
Afib->Dead
Str->Dead
Afib+Str->Dead
> p0 <- glmLexis(Sho, ~ Ns(time, knots = c(0,2,6,12)))
stats::glm Poisson analysis of Lexis object Sho with log link:
Rates for transitions:
Alive->Dead
Afib->Dead
Str->Dead
Afib+Str->Dead
> round(ci.exp(ps), 2)
                                  exp(Est.) 2.5% 97.5%
(Intercept)                            0.01 0.00  0.02
Ns(time, knots = c(0, 2, 6, 12))1      2.08 1.15  3.74
Ns(time, knots = c(0, 2, 6, 12))2      7.30 1.36 39.12
Ns(time, knots = c(0, 2, 6, 12))3      2.32 1.56  3.45
lex.CstAfib                            3.13 2.03  4.83
lex.CstStr                             4.45 3.11  6.36
lex.CstAfib+Str                        9.24 4.48 19.04
> anova(ps, p0, test = "Chisq")
Analysis of Deviance Table

Model 1: cbind(trt(Lx$lex.Cst, Lx$lex.Xst) %in% trnam, Lx$lex.dur) ~ Ns(time,
    knots = c(0, 2, 6, 12)) + lex.Cst
Model 2: cbind(trt(Lx$lex.Cst, Lx$lex.Xst) %in% trnam, Lx$lex.dur) ~ Ns(time,
    knots = c(0, 2, 6, 12))
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1    100141     3520.9
2    100144     3601.9 -3  -80.946 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can also fit the corresponding Cox-models and compare the estimated HRs:

```
> cs <- coxphLexis(Mho, time ~ lex.Cst)
survival::coxph analysis of Lexis object Mho:
Rates for transitions:
Alive->Dead
Afib->Dead
Str->Dead
Afib+Str->Dead
Baseline timescale: time
> c0 <- coxphLexis(Mho, time ~ 1)
survival::coxph analysis of Lexis object Mho:
Rates for transitions:
Alive->Dead
Afib->Dead
Str->Dead
Afib+Str->Dead
Baseline timescale: time
> round(cbind(ci.exp(ps, subset = 5:7),
+             ci.exp(cs, subset = 1:3)), 2)
               exp(Est.) 2.5% 97.5% exp(Est.) 2.5% 97.5%
lex.CstAfib         3.13 2.03  4.83      3.13 2.03  4.83
lex.CstStr          4.45 3.11  6.36      4.45 3.11  6.36
lex.CstAfib+Str     9.24 4.48 19.04      9.60 4.66 19.79
```

We can make a forest plot of the estimates comparing the Poisson and Cox estimates:

```
> plotEst(ci.exp(ps, subset = "lex"),
+         xlog = TRUE, xlim = c(1,20), grid = c(1,2,5,10,20),
+         xlab = "HR of death versus noDis")
> pointsEst(ci.exp(cs, subset = 1:3), y = 3:1 - 0.2, col = "gray")
> axis(side = 1, at = 1:2)
```

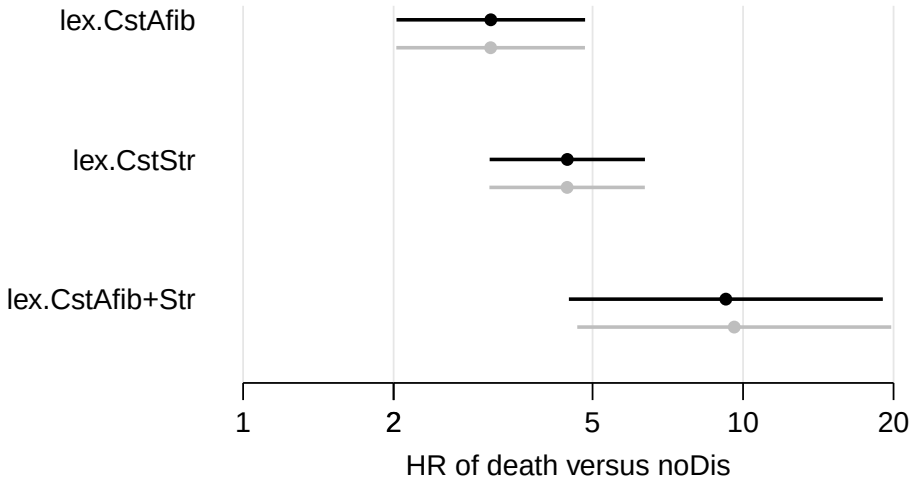

Figure 1.9: *Mortality HRs from a model with current state as categorical and time since entry as smooth term. Black estimates correspond to estimates from a Poisson model, gray from a Cox model.* `../graph/holter-forest`

We then plot the overall mortality rates from model `p0`:

```
> par(mfrow = c(1,2))
> nd <- data.frame(time = seq(0, 15, 0.1))
> matshade(nd$time, ci.pred(p0, nd) * 100,
+          lwd = 3, plot = TRUE, log = "y",
+          xlab = "Time since entry (years)",
+          ylab = "Mortality rate per 100 PY")
> matshade(nd$time, ci.surv(p0, nd), lwd = 3, plot = TRUE,
+          ylim = 0:1, yaxs = "i",
+          xlab = "Time since entry (years)",
+          ylab = "Survival probability")
NOTE: interval length chosen from  as time[2] - time[1]
```

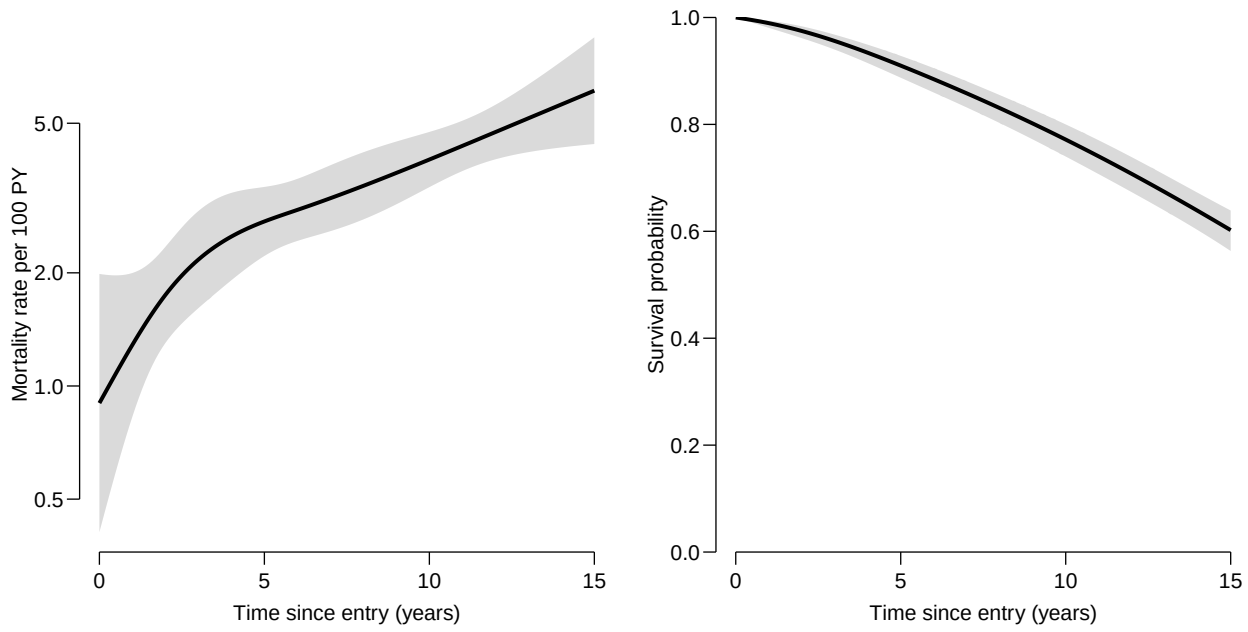

Figure 1.10: *The mortality rates from a model with only time since entry as smooth term.*
../graph/holter-m-rate

```
  Start time: 2024-12-30, 11:19:51
    End time: 2024-12-30, 11:19:55
Elapsed time: 0.06 minutes
```

. . . now input from `smSurv.tex`

1.8 A small survival example (table 1.6)

This is not quite what is given in the book's table 1.6; identical entry ages (4 in 0 and two in 6) have been offset a bit for the sake of the two-dimensional Lexis diagram:

```
> times <- c(5,6,7,8,9,12,13,15,16,20,22,23)
> age <- c(12,0,0.3,10,6,6.3,9,3,8,0.6,0.9,2)
> event <- c(1,0,1,1,0,0,1,1,1,0,0,1)
> id <- 1:12
> (simpledata <- as.data.frame(cbind(id, times, event, age)))
   id times event  age
1   1     5     1 12.0
2   2     6     0  0.0
3   3     7     1  0.3
4   4     8     1 10.0
5   5     9     0  6.0
6   6    12     0  6.3
7   7    13     1  9.0
8   8    15     1  3.0
9   9    16     1  8.0
10 10    20     0  0.6
11 11    22     0  0.9
12 12    23     1  2.0
```

The easiest way to get a proper plot is to set up the data as a `Lexis` object:

```
> Lx <- Lexis(entry = list(Tse = 0,
+                          Age = age),
+             exit = list(Tse = times),
+       exit.status = factor(event, labels = c("Alive","Dead")),
+             data = simpledata)
NOTE: entry.status has been set to "Alive" for all.
> Lx
 lex.id Tse  Age lex.dur lex.Cst lex.Xst id times event  age
      1   0 12.0       5   Alive    Dead  1     5     1 12.0
      2   0  0.0       6   Alive   Alive  2     6     0  0.0
      3   0  0.3       7   Alive    Dead  3     7     1  0.3
      4   0 10.0       8   Alive    Dead  4     8     1 10.0
      5   0  6.0       9   Alive   Alive  5     9     0  6.0
      6   0  6.3      12   Alive   Alive  6    12     0  6.3
      7   0  9.0      13   Alive    Dead  7    13     1  9.0
      8   0  3.0      15   Alive    Dead  8    15     1  3.0
      9   0  8.0      16   Alive    Dead  9    16     1  8.0
     10   0  0.6      20   Alive   Alive 10    20     0  0.6
     11   0  0.9      22   Alive   Alive 11    22     0  0.9
     12   0  2.0      23   Alive    Dead 12    23     1  2.0
> summary(Lx)
Transitions:
     To
From    Alive Dead  Records:  Events: Risk time:  Persons:
  Alive     5    7        12        7        156        12
```

First we plot the two one-dimensional Lexis diagrams:

```
> par(mfrow = c(1,2), yaxt = "n")
> # Tse as time
> plot(Lx, time.scale = 1, col = "black", lwd = 2)
> abline(v = seq(0,20,5), col = gray(0.9))
> points(Lx, pch = c(21,16), lwd = 2, bg = "white")
> # Age as time
> plot(Lx, time.scale = 2, col = "black", lwd = 2)
> abline(v = seq(0,25,5), col = gray(0.9))
> points(Lx, pch = c(21,16), lwd = 2, bg = "white")
```

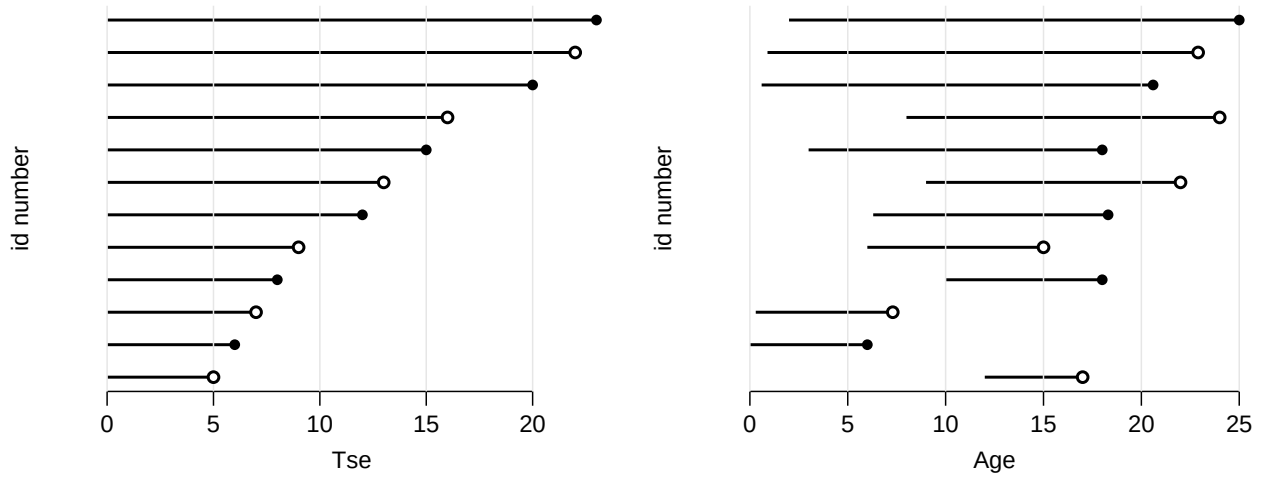

Figure 1.11: *One-dimensional Lexis diagrams of the (slightly modified)example data in table 1.6 in KR, corresponding to figure 1.8.* `../graph/smSurv-Lx1`

Then we can plot the Lexis diagram that should have been done in the first place:

```
> par(mfrow=c(1,1), mar = c(3,3,1,1), mgp = c(3,1,0)/1.6)
> plot(Lx, col = "black", lwd = 2,
+      xaxs = "i", yaxs = "i",
+      xlim = c(0,26), ylim = c(0,26))
> abline(v = 1:5 * 5, h = 1:5 * 5, col = gray(0.7))
> lines(Lx, lwd=2)
> points(Lx, pch = c(21,16), lwd = 2, bg = "white")
```

If we want to see the follow-up in 5-year classes we must split the FU along the two time scales, for example by using `splitMulti` from the `popEpi` package:

```
> Sx <- splitMulti(Lx, Age = 0:6 * 5, Tse = 0:6 * 5)
> par(mar = c(3,3,1,1), mgp = c(3,1,0)/1.6)
> plot(Sx, col = "black", lwd = 2,
+      xaxs = "i", yaxs = "i",
+      xlim = c(0,26), ylim = c(0,26))
> # A ring at the end of each person's FU (how wise is that)
> points(exit(Sx, by.id = TRUE), pch = 21, lwd = 1, bg = "white")
> # A blob at each event, overplotted
> points(Sx, pch = c(NA,16)[Sx$lex.Xst], lwd = 2, bg = "white")
> # annotate with the person-years
> PY.ann(Sx, cex = 0.6)
> box()
```

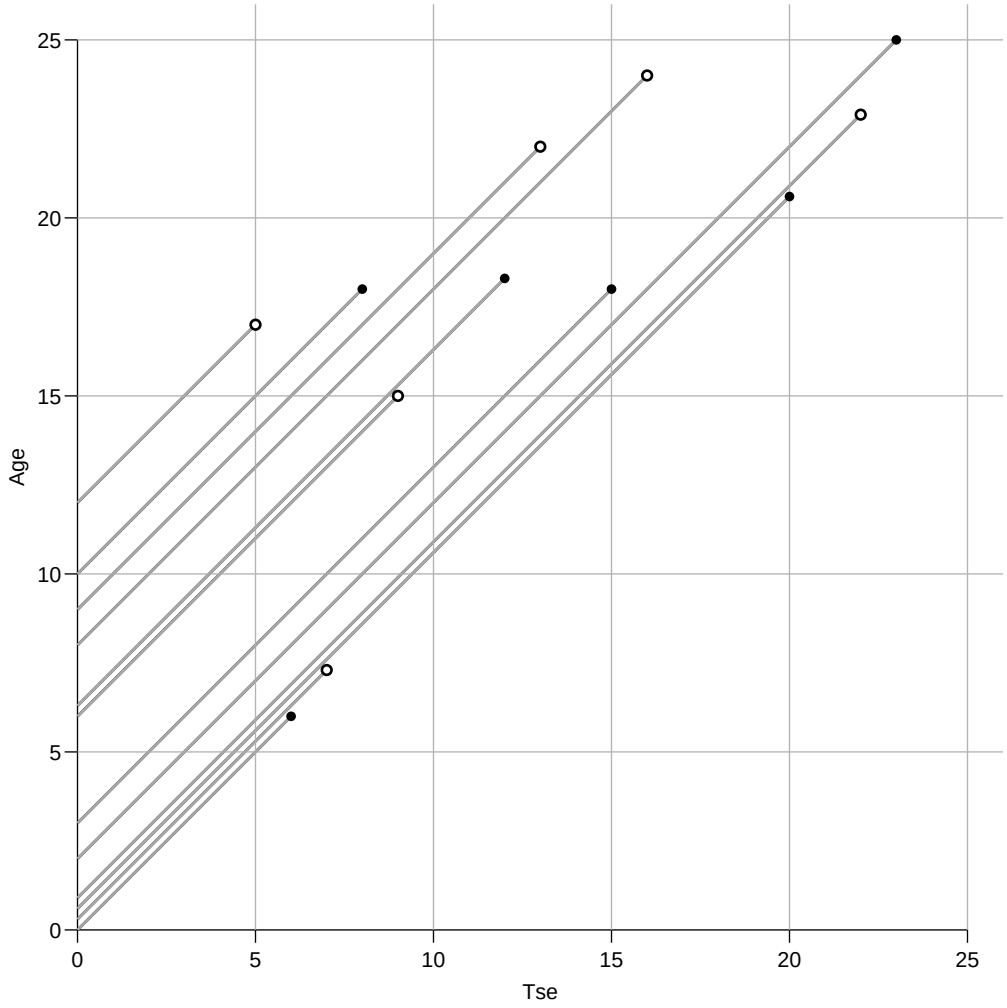

Figure 1.12: *Lexis diagram of the example data in table 1.6 in KR.* `../graph/smSurv-Lx2`

We can summarize the person-years in each age by time bin as defined in `breaks`:

```
> attr(Sx, "breaks")
$Tse
[1]  0  5 10 15 20 25 30

$Age
[1]  0  5 10 15 20 25 30
```

```
> print(ftable(
+ xtabs(cbind(D = lex.Xst == "Dead",
+              Y = lex.dur)
+       ~ T + A,
+       data = mutate(Sx,
+                     A = timeBand(Sx, "Age", "left"),
+                     T = timeBand(Sx, "Tse", "left")))[,1:5,],
+              col.vars = c(3,1)), zero.print = ".")
```

```
        D                          Y
   T    0    5   10   15   20    0    5   10   15   20
A
0       .    .    .    .    .   23.2  .    .    .    .
```

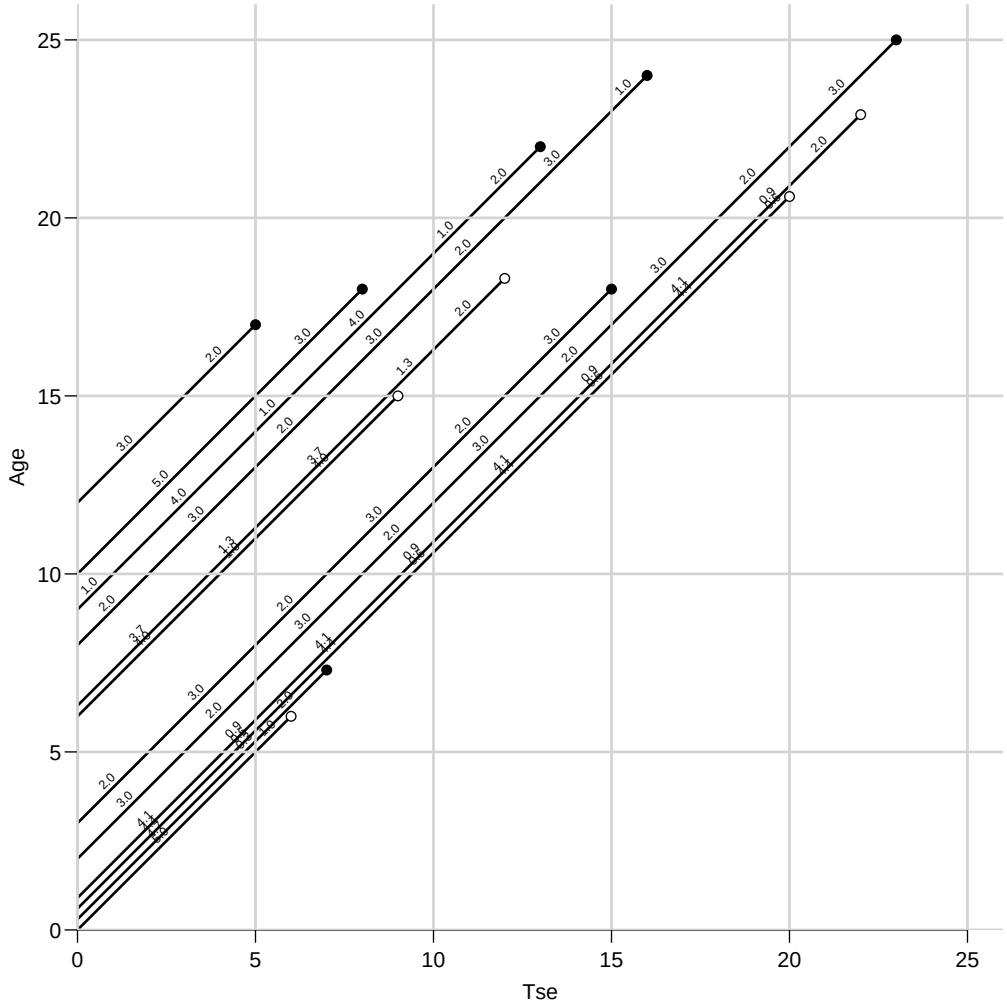

Figure 1.13: *Lexis diagram of the example data in table 1.6 in KR. Grid corresponds to splitting time in 5 year intervals on both time scales.* ../graph/smSurv-Sx2

```
5     .    1.0  .    .    .   17.5 16.5  .    .    .
10    .    .    .    .    .   17.3 17.2 13.5  .    .
15    1.0  1.0  1.0  .    .    2.0 11.3 11.5 11.5  .
20    .    .    1.0  1.0  1.0  .    .    5.0  4.5  5.0
```

We can derive the survival function as the probability of being in the state `Alive`, using the result of `AaJ.Lexis`

```
> aaj.A <- AaJ.Lexis(Sx, timeScale = "Age")
NOTE: Timescale is Age
> aaj.T <- AaJ.Lexis(Sx, timeScale = "Tse")
NOTE: Timescale is Tse
> par(mfrow = c(1,2))
> plot(aaj.T, lwd = 2, yaxs = "i", xlim = c(0, 26), xlab = "Time since entry",
+      noplot="Dead")
> plot(aaj.A, lwd = 2, yaxs = "i", xlim = c(0, 26), xlab = "Age",
+      noplot="Dead")
```

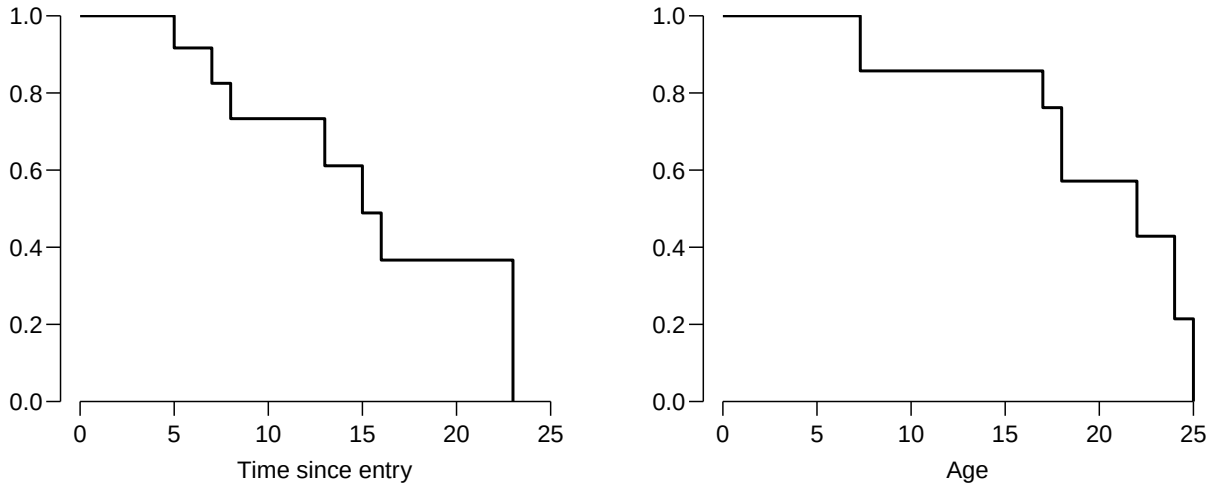

Figure 1.14: *Survival function since 0 on both time scales. Luckily both time scales have 0 as a meaningful origin—this may not always be the case for age.* `../graph/smSurv-aaj`