Occurrence rates, cumulative risks, competing risks, state probabilities with multiple states and time scales in modern, groundbreaking, cutting edge, frontier, state of the art Register Research with R and Epi::

Computer practicals

SDCstats
October 2021
http://bendixcarstensen.com/AdvCoh/courses/SDC-2021
Version 1

Compiled Wednesday 15th September, 2021, 18:05
from: `/home/bendix/teach/AdvCoh/courses/SDCA.2021/pracs/pracs.tex`

Bendix Carstensen	Steno Diabetes Center Copenhagen, Gentofte, Denmark & Department of Biostatistics, University of Copenhagen b@bxc.dk http://BendixCarstensen.com
Lars Jorge Diaz	Steno Diabetes Center Copenhagen, Gentofte, Denmark
Adam Hulman	Steno Diabetes Center Midt, Skejby, Denmark

Contents

0.0 Preface

This course draws on the content of the book "Epidemiology with R" [1], (`http://bendixcarstensen.com/EwR`), but in particular on the draft of my new book (which by no means is sure ever to appear as a book) "Practical multistate modeling with R and `Epi:Lexis`". The former is available through Oxford University Press, the latter as a draft (updated at unpredictable times) as `http://bendixcarstensen.com/MSbook.pdf`.

- The **target audience** is the group of statisticians and epidemiologists working in or with the 5 SDCentres.
- The **prerequisites** are
  1. a basic knowledge of R,
  2. a working installation of `Epi_2.44`
  3. a working installation of `popEpi_0.4.8`
  4. some epidemiological practice
- The main groundbreaking etc. **feature** of the course is that you are supposed to turn on your brain before you start coding.
- The **format** of the course will be short lectures closely aligned with the topics in the exercises. The exercises will be run in chunks between the short lectures.

Exercises are given including most of the solutions. You can get the exercise code chunks from the course website `http://bendixcarstensen.com/AdvCoh/courses/SDC-2021`

Chapter 1

Practicals

1.1 Survival and rates: lung

- lung data from survival package
- KM estimator
- Cox model with effect of sex and age (at entry)
- Lexis object, simple
- Timesplit by splitLexis / splitMulti
- glm with Ns - same result as Cox
- Baseline hazard using predict, survival using ci.surv
- Survival functions from smooth and KM models compared

1.1.1 Paraphernalia

It is advisable to load all packages needed at the start:

```
> library(survival)
> library(Epi)
> library(popEpi)
> # popEpi::splitMulti returns a data.frame rather than a data.table
> options("popEpi.datatable" = FALSE)
```

1.1.2 Data

1. Load the lung data from the survival package, and convert sex to a factor (*always* do that with categorical variables). Also we rescale time from days to months:

```
> data(lung)
> lung$sex <- factor(lung$sex, labels = c("M", "W"))
> lung$time <- lung$time / (365.25/12)
> head(lung)

  inst      time status age sex ph.ecog ph.karno pat.karno meal.cal wt.loss
1    3 10.053388      2  74   M       1       90       100     1175      NA
2    3 14.948665      2  68   M       0       90        90     1225      15
3    3 33.182752      1  56   M       0       90        90       NA      15
4    5  6.899384      2  57   M       1       90        60     1150      11
5    1 29.010267      2  60   M       0      100        90       NA       0
6   12 33.577002      1  74   M       1       50        80      513       0
```

2. Use `survfit` to construct the Kaplan-Meier estimator of overall survival:

```
> ?Surv
> ?survfit
> km <- survfit(Surv(time, status == 2) ~ 1, data = lung)
> km

Call: survfit(formula = Surv(time, status == 2) ~ 1, data = lung)

      n  events  median 0.95LCL 0.95UCL
 228.00  165.00   10.18    9.36   11.93

> # summary(km) # very long output
```

The standard print method just prints the number of events and the median survival, while the `summary` prints the entire survival function estimate.

We can plot the survival curve—this is the default plot for a `survfit` object:

```
> plot(km)
```

What is the median survival? What does it mean?

3. Explore if survival patterns between men and women are different:

```
> kms <- survfit(Surv(time, status == 2) ~ sex, data = lung)
> kms

Call: survfit(formula = Surv(time, status == 2) ~ sex, data = lung)

        n events median 0.95LCL 0.95UCL
sex=M 138    112   8.87    6.97    10.2
sex=W  90     53  14.00   11.43    18.1
```

We can plot the two resulting survival curves with confidence limits:

```
>  plot(kms)
>  plot(kms, col = c("blue", "red"), lwd = 1, conf.int = TRUE)
> lines(kms, col = c("blue", "red"), lwd = 3)
```

We see that men have worse survival than women, but they are also a bit older (`age` is age at diagnosis of lung cancer):

```
> with(lung, tapply(age, sex, mean))

       M        W
63.34058 61.07778
```

Formally there is a significant difference in survival between men and women

```
> ?survdiff
> survdiff(Surv(time, status==2) ~ sex, data = lung)

Call:
survdiff(formula = Surv(time, status == 2) ~ sex, data = lung)

        N Observed Expected (O-E)^2/E (O-E)^2/V
sex=M 138      112     91.6      4.55      10.3
sex=W  90       53     73.4      5.68      10.3

 Chisq= 10.3  on 1 degrees of freedom, p= 0.001
```

What is the null hypothesis tested here?

4. Now explore how sex and age (at diagnosis) influence the mortality—note that we are now addressing the mortality rate and not the survival in a Cox-model:

```
> c0 <- coxph(Surv(time, status == 2) ~ sex      , data = lung)
> c1 <- coxph(Surv(time, status == 2) ~ sex + age, data = lung)
> summary(c1)

Call:
coxph(formula = Surv(time, status == 2) ~ sex + age, data = lung)

  n= 228, number of events= 165

          coef exp(coef)  se(coef)      z Pr(>|z|)
sexW -0.513219  0.598566  0.167458 -3.065  0.00218
age   0.017045  1.017191  0.009223  1.848  0.06459

     exp(coef) exp(-coef) lower .95 upper .95
sexW    0.5986     1.6707    0.4311    0.8311
age     1.0172     0.9831    0.9990    1.0357

Concordance= 0.603  (se = 0.025 )
Likelihood ratio test= 14.12  on 2 df,   p=9e-04
Wald test            = 13.47  on 2 df,   p=0.001
Score (logrank) test = 13.72  on 2 df,   p=0.001

> ci.exp(c0)

     exp(Est.)      2.5%     97.5%
sexW 0.5880028 0.4237178 0.8159848
```

```
> ci.exp(c1)

     exp(Est.)      2.5%     97.5%
sexW  0.598566 0.4310936 0.8310985
age   1.017191 0.9989686 1.0357467
```

We see that there is not much confounding by age; the W/M mortality RR (hazard ratio is another word for this) is slightly below 0.6 whether age is included or not.

The age effect is formally non-significant, the estimate corresponds to a mortality RR of 1.7% per year of age at diagnosis.

What is the mortality RR for a 10 year age difference?

5. We can check if the assumption of proportional hazards holds, `cox.zph` provides a test, and the plot method shows the Schoenfeld residuals and a smooth of them; interpretable as an estimate of the interaction effect; that is how the W/M (log) rate-ratio depends on time:

```
> ?cox.zph
> cox.zph(c0)

       chisq df     p
sex     2.86  1 0.091
GLOBAL  2.86  1 0.091

> (z1 <- cox.zph(c1))

       chisq df    p
sex    2.608  1 0.11
age    0.209  1 0.65
GLOBAL 2.771  2 0.25

> par(mfrow = c(1, 2)) ; plot(z1)
```

6. But we do not know how the mortality *per se* looks as a function of time (since diagnosis). That function is not available from the Cox-model or from the `survfit` object. To that end we must provide a model for the effect of time on mortality; the simplest is of course to assume that it is constant or a simple linear function of time.

If we assume the mortality is constant over time, it is so that the likelihood for the model is equivalent to a Poisson likelihood, which can be fitted using the `poisreg` family from the `Epi` package:

```
> ?poisreg
> p1 <- glm(cbind(status == 2, time) ~ sex + age,
+            family = poisreg,
+              data = lung)
> ci.exp(p1)

             exp(Est.)       2.5%     97.5%
(Intercept) 0.03255152 0.01029228 0.1029511
sexW        0.61820515 0.44555636 0.8577537
age         1.01574132 0.99777446 1.0340317
```

```
> ci.exp(c1)

     exp(Est.)      2.5%     97.5%
sexW  0.598566 0.4310936 0.8310985
age   1.017191 0.9989686 1.0357467
```

We see that the estimates of sex and age effects are quite close between the Poisson and the Cox models, but also that the Poisson model has an intercept term, the estimate of the (assumed) constant underlying mortality. Since we entered the risk time part of the response (second argument in the `cbind`) in units of months (remember we rescaled in the beginning?), the `(Intercept)` is a rate per 1 person-month.

What age and sex does the `(Intercept)` refer to?

The syntax for `poisreg` is a bit different from that for `poisson`, which would be:

```
> px <- glm(status == 2  ~ sex + age,
+           offset = log(time),
+           family = poisson,
+             data = lung)
> px <- glm(status == 2  ~ sex + age + offset(log(time)),
+           family = poisson,
+             data = lung)
> ci.exp(px)
```

This is the reason that papers use the description ". . . we fitted a Poisson model with log person years as offset". The drawback of the `poisson` approach is that you need the `time` (person-years) variable in the prediction frame, that is not the case for `poisreg`

1.1.3 Lexis object

If we want to see how mortality varies by age we must split the follow-up of each person in small intervals of say, 30 days. This is most easily done using a `Lexis` object. That is basically just taking the `lung` dataset and adding a few features that defines times and states. The point is that it makes life a lot easier when things get more complex than just simple survival.

7. First make a `Lexis` object:

```
> ?Lexis
> Ll <- Lexis(exit = list(tfl = time),
+             exit.status = factor(status,
+                                  levels = 1:2,
+                                  labels = c("Alive","Dead")),
+             data = lung)

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

```
> head(L1)

  tfl   lex.dur lex.Cst lex.Xst lex.id inst      time status age sex ph.ecog ph.karno
1   0 10.053388   Alive    Dead      1    3 10.053388      2  74   M       1       90
2   0 14.948665   Alive    Dead      2    3 14.948665      2  68   M       0       90
3   0 33.182752   Alive   Alive      3    3 33.182752      1  56   M       0       90
4   0  6.899384   Alive    Dead      4    5  6.899384      2  57   M       1       90
5   0 29.010267   Alive    Dead      5    1 29.010267      2  60   M       0      100
6   0 33.577002   Alive   Alive      6   12 33.577002      1  74   M       1       50
  pat.karno meal.cal wt.loss
1       100     1175      NA
2        90     1225      15
3        90       NA      15
4        60     1150      11
5        90       NA       0
6        80      513       0
```

We see that 5 variables have been added to the dataset:

`tfl`: time from lung cancer *at the time of entry*, therefore it is 0 for all persons; the entry time is 0 from the entry time.

`lex.dur`: the *length* of time a person is in state `lex.Cst`, here measured in months, because `time` is.

`lex.Cst`: Current `state`, the state in which the `lex.dur` time is spent.

`lex.Xst`: eXit `state`, the state to which the person moves after the `lex.dur` time in `lex.Cst`.

`lex.id`: a numerical id of each record in the dataset (normally this wll be a person id).

This seems a bit of an overkill for keeping track of time and death for the lung cancer patients, but the point is that this generalizes to multistate data too.

It also gives a handy overview of the follow-up:

```
> summary(L1)

Transitions:
     To
From    Alive Dead  Records:  Events: Risk time:  Persons:
  Alive    63  165       228      165    2286.42       228
```

What is the average follow-up time for persons?

For a graphical representation, try:

```
> ?boxes
> boxes(L1, boxpos = TRUE)
```

Explain the numbers in the resulting graph. Redo the graph with risk time counted in years.

8. We can make the Cox-analysis using the Lexis-specific variables by:

```
> ?Surv
> cl <- coxph(Surv(tfl,
+                  tfl + lex.dur,
+                  lex.Xst == "Dead") ~ sex + age,
+             data = Ll)
```

but even simpler, by using the Lexis features:

```
> ?coxph.Lexis
> cL <- coxph.Lexis(Ll, tfl ~ sex + age)

survival::coxph analysis of Lexis object Ll:
Rates for the transition Alive->Dead
Baseline timescale: tfl

> ci.exp(cL)

     exp(Est.)      2.5%     97.5%
sexW  0.598566 0.4310936 0.8310985
age   1.017191 0.9989686 1.0357467

> ci.exp(cl)

     exp(Est.)      2.5%     97.5%
sexW  0.598566 0.4310936 0.8310985
age   1.017191 0.9989686 1.0357467
```

9. And we can make the Poisson-analysis by:

```
> pc <- glm(cbind(lex.Xst == "Dead", lex.dur) ~ sex + age,
+           family = poisreg,
+             data = Ll)
```

or even simpler, by using the Lexis features:

```
> pL <- glm.Lexis(Ll, ~ sex + age)

stats::glm Poisson analysis of Lexis object Ll with log link:
Rates for the transition: Alive->Dead

> ci.exp(pL)

             exp(Est.)       2.5%     97.5%
(Intercept) 0.03255152 0.01029228 0.1029511
sexW        0.61820515 0.44555636 0.8577537
age         1.01574132 0.99777446 1.0340317

> ci.exp(pc)

             exp(Est.)       2.5%     97.5%
(Intercept) 0.03255152 0.01029228 0.1029511
sexW        0.61820515 0.44555636 0.8577537
age         1.01574132 0.99777446 1.0340317
```

Remember that the Poisson-model fitted is a very brutal approximation to the Cox-model; it assumes that the baseline hazard is constant, whereas the Cox-model allows the baseline hazard to vary arbitrarily by time.

1.1.4 Splitting time

If we want a more detailed version of the baseline hazard we split follow-up time in small intervals, assume that the hazard is constant in each small interval, and assume the the *size* of the hazard varies smoothly with time, `tfl`:

10. We can subdivide the follow-up in small intervals by `survival:::survSplit`, `Epi:::splitLexis` or `popEpi:::splitMulti` (and possibly many more). The `splitMulti` is by far the easiest to use (and fastest as well). Recall we rescaled time to months, so we split in 1 month intervals:

    ```
    > S1 <- splitMulti(L1, tfl = 0:36)
    ```

    This will split the follow-up along the time-scale `tfl` at times 0, 1, . . . , 36 months; we see that the follow-up time is the same, but there are now about 10 times as many records:

    ```
    > summary(L1)

    Transitions:
         To
    From    Alive Dead  Records:  Events: Risk time:  Persons:
      Alive    63  165       228       165    2286.42       228

    > summary(S1)

    Transitions:
         To
    From    Alive Dead  Records:  Events: Risk time:  Persons:
      Alive  2234  165      2399       165    2286.42       228
    ```

    We can see how the follow up for person, 10 say, is in the original and the split dataset:

    ```
    > wh <- names(L1)[1:10] # names of variables in some order
    > subset(L1, lex.id == 10)[,wh]

       tfl  lex.dur lex.Cst lex.Xst lex.id inst     time status age sex
    10   0 5.453799   Alive    Dead     10    7 5.453799      2  61   M

    > subset(S1, lex.id == 10)[,wh]

         tfl   lex.dur lex.Cst lex.Xst lex.id inst     time status age sex
    163    0 1.0000000   Alive   Alive     10    7 5.453799      2  61   M
    164    1 1.0000000   Alive   Alive     10    7 5.453799      2  61   M
    165    2 1.0000000   Alive   Alive     10    7 5.453799      2  61   M
    166    3 1.0000000   Alive   Alive     10    7 5.453799      2  61   M
    167    4 1.0000000   Alive   Alive     10    7 5.453799      2  61   M
    168    5 0.4537988   Alive    Dead     10    7 5.453799      2  61   M
    ```

    In `S1` each record now represents a small interval of follow-up for a person, so each person has many records. The main thing to note here is `tfl`, which represents the time from lung cancer at the beginning of each interval, and `lex.dur` representing the risk time ("person-years", in months though).

11. We can now include a smooth effect of `tfl` in the Poisson-model allowing the baseline hazard to vary by time. That is done by natural splines, `Ns`:

```
> ps <- glm(cbind(lex.Xst == "Dead", lex.dur)
+            ~ Ns(tfl, knots = seq(0, 36, 12)) + sex + age,
+            family = poisreg,
+              data = Sl)
> ci.exp(ps)

                                 exp(Est.)        2.5%       97.5%
(Intercept)                      0.0189837 0.005700814  0.06321569
Ns(tfl, knots = seq(0, 36, 12))1 2.4038681 0.809442081  7.13896863
Ns(tfl, knots = seq(0, 36, 12))2 4.1500822 0.436273089 39.47798357
Ns(tfl, knots = seq(0, 36, 12))3 0.8398973 0.043928614 16.05849662
sexW                             0.5987171 0.431232662  0.83124998
age                              1.0165872 0.998377104  1.03512945
```

or even simpler:

```
> ?glm.Lexis
> ps <- glm.Lexis(Sl, ~ Ns(tfl, knots = seq(0, 36, 12)) + sex + age)

stats::glm Poisson analysis of Lexis object Sl with log link:
Rates for the transition: Alive->Dead

> ci.exp(ps)

                                 exp(Est.)        2.5%       97.5%
(Intercept)                      0.0189837 0.005700814  0.06321569
Ns(tfl, knots = seq(0, 36, 12))1 2.4038681 0.809442081  7.13896863
Ns(tfl, knots = seq(0, 36, 12))2 4.1500822 0.436273089 39.47798357
Ns(tfl, knots = seq(0, 36, 12))3 0.8398973 0.043928614 16.05849662
sexW                             0.5987171 0.431232662  0.83124998
age                              1.0165872 0.998377104  1.03512945
```

12. Compare these to the regression estimates from the Cox-model and from the model with constant baseline:

```
> round(cbind(ci.exp(cl),
+              ci.exp(ps, subset = c("sex","age")),
+              ci.exp(pc, subset = c("sex","age"))), 3)

     exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5%
sexW     0.599 0.431 0.831     0.599 0.431 0.831     0.618 0.446 0.858
age      1.017 0.999 1.036     1.017 0.998 1.035     1.016 0.998 1.034
```

We see that the smooth parametric Poisson model and the Cox model produce virtually the same estimates, whereas the Poisson model with constant hazard produce slightly different ones.

The same again:

```
> round(cbind(ci.exp(cl),ci.exp(ps,subset=c("sex","age")),ci.exp(pc,
+ subset=c("sex","age"))), 3)
```

```
     exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5% exp(Est.)  2.5% 97.5%
sexW     0.599 0.431 0.831     0.599 0.431 0.831     0.618 0.446 0.858
age      1.017 0.999 1.036     1.017 0.998 1.035     1.016 0.998 1.034
```

What is wrong with that, it gives the same result?

13. We now have a parametric model for the baseline hazard which means that we can show how the estimates baseline hazard for a 60-year old woman, by supplying a prediction frame, i.e. a data frame where each row represents a set of covariate values where we want the predicted mortality:

```
> prf <- data.frame(tfl = seq(0, 30, 0.2),
+                   sex = "W",
+                   age = 60)
```

We can overplot with the predicted rates from the model where mortality rates are constant, the only change is the model (`pc` instead of `ps`):

```
> matshade(prf$tfl, ci.pred(ps, prf),
+          plot = TRUE, log = "y", lwd = 3)
> matshade(prf$tfl, ci.pred(pc, prf), lty = 3, lwd = 3)
```

What we see from the plot is that mortality rates are increasing during the first 1.5 years after lung cancer and then leveling off.

Put some sensible axis labels on the plot, and rescale the rates to rates per 1 person-year.

14. We can transform the hazard function, $\lambda(t)$, to a survival function, $S(t)$ using the relationship $S(t) = \exp(-\int_0^t \lambda(u)\, du)$. This is implemented in the `ci.surv` function, which takes the model and a prediction data frame as arguments; the prediction data frame must correspond to a sequence of equidistant time points, so we can use `prf` for this purpose:

```
> matshade(prf$tfl, ci.surv(ps, prf, intl = 0.2),
+          plot = TRUE, ylim = 0:1, lwd = 3)
```

We can expand this by overlaying the survival function from the model with constant hazard (also known as ”exponential(y distributed) survival”) and the KM-estimator

```
> matshade(prf$tfl, ci.surv(ps, prf, intl = 0.2),
+          plot = TRUE, ylim = 0:1, lwd = 3)
> lines(prf$tfl, ci.surv(pc, prf, intl = 0.2)[,1])
> lines(survfit(c1, newdata = data.frame(sex = "W", age = 60)),
+       lwd = 2, lty = 1)
```

We see that the survival function from the constant hazard model is quite a bit off, but also a good correspondence between the Cox-model based survival and the survival from the parametric hazard function.

We can bring the plots together in one graph:

```
> par(mfrow = c(1,2))
> # hazard scale
> matshade(prf$tfl, ci.pred(ps, prf),
+          plot = TRUE, log = "y", lwd = 3)
> matshade(prf$tfl, ci.pred(pc, prf), lty = 3, lwd = 3)
> # survival
> matshade(prf$tfl, ci.surv(ps, prf, intl = 0.2),
+          plot = TRUE, ylim = 0:1, lwd = 3)
> matshade(prf$tfl, ci.surv(pc, prf, intl = 0.2),
+          lty = 3, alpha = 0, lwd = 3)
> lines(survfit(c1, newdata = data.frame(sex = "W", age = 60)),
+       col = "forestgreen", lwd = 3)
```

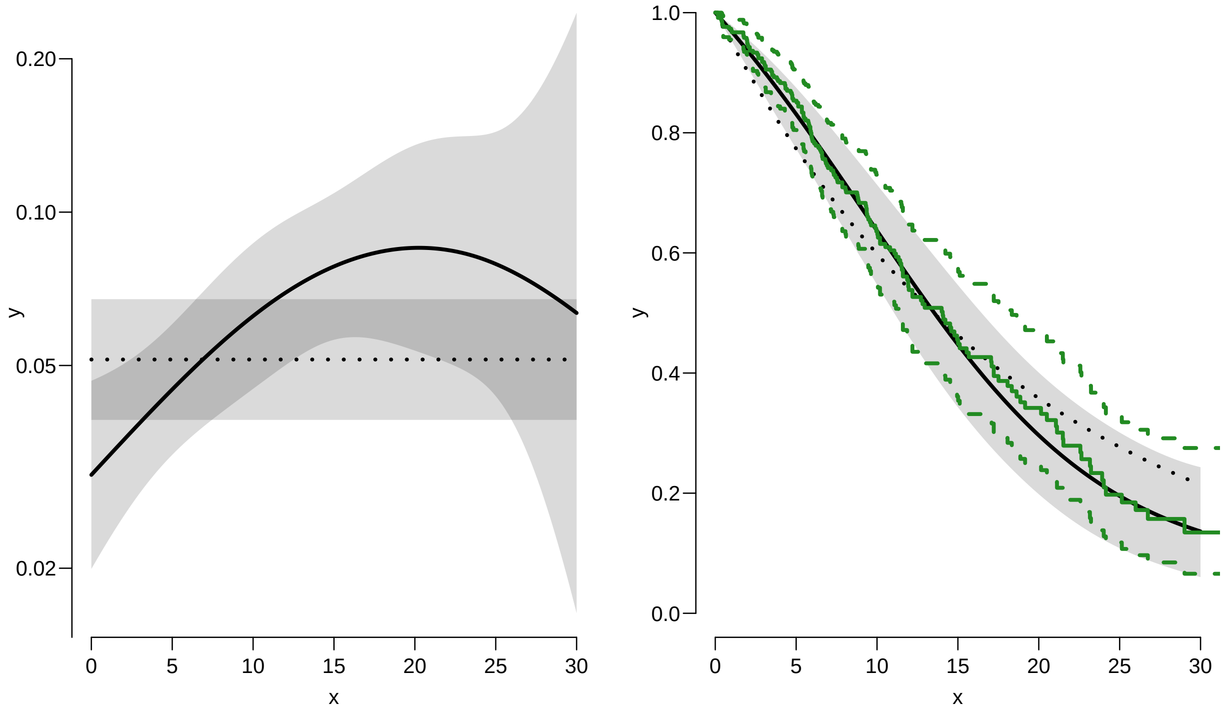

Figure 1.1: *Hazards (left) and survival (right) for 60 year old women. The left hand plot is unavailable from the Cox model.*

`../graph/surv-ratesurv`

15. We have compared the predicted a survival curve from a Poisson model with time since lung cancer, age and sex to that from a Cox-model with age and sex and time since lung cancer as underlying time scale.

We now go back to the Kaplan-Meier estimator and compare that to the corresponding Poisson-model, which is one with time (`tfl`) as the only covariate:

```
> par(mfrow=c(1,2))
> pk <- glm(cbind(lex.Xst == "Dead",
+                lex.dur) ~ Ns(tfl, knots = seq(0, 36, 12)),
+           family = poisreg,
+             data = S1)
> # hazard
> matshade(prf$tfl, ci.pred(pk, prf),
+          plot = TRUE, log = "y", lwd = 3, ylim = c(0.01,1))
> # survival from smooth model
> matshade(prf$tfl, ci.surv(pk, prf, intl = 0.2) ,
+          plot = TRUE, lwd = 3, ylim = 0:1)
> # K-M estimator
> lines(km, lwd = 2)
```

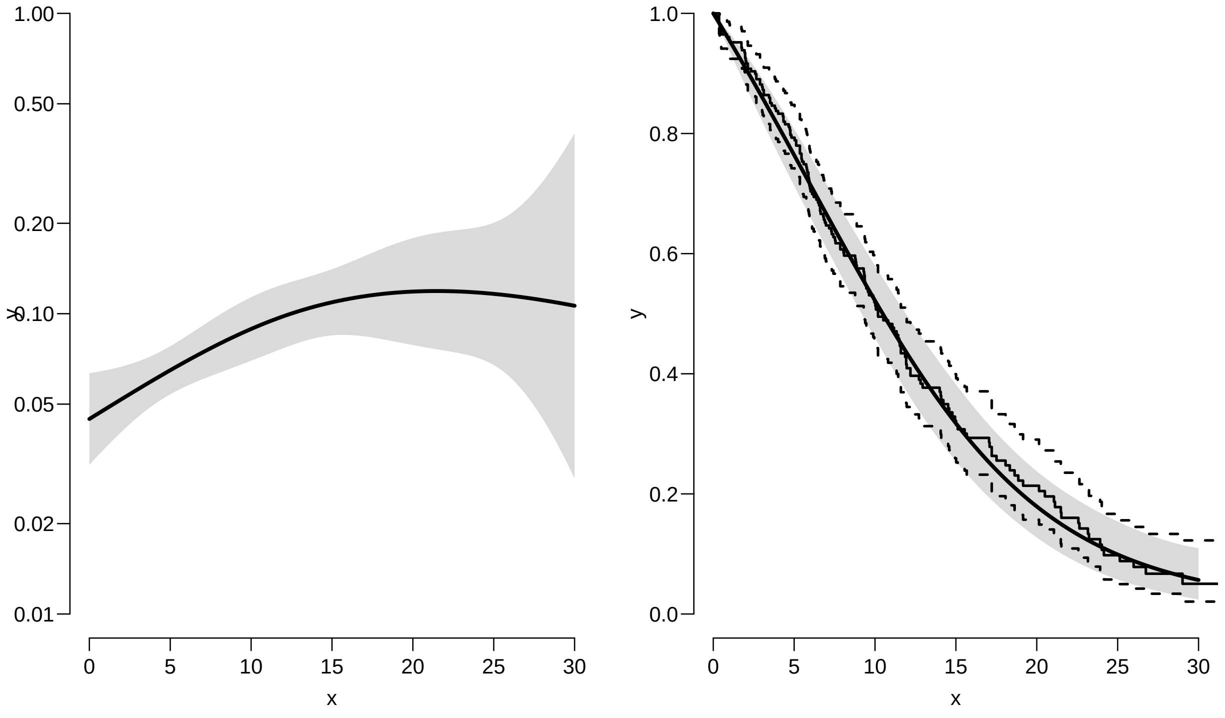

Figure 1.2: *Baseline hazard (left), and corresponding survival function from parametric model and Kaplan-Meier estimator.*

`../graph/surv-parkm`

16. We can explore how the tightness of the knots in the smooth model influence the underlying hazard and the resulting survival function:

```
> zz <-
+ function(dk)
+ {
+ par(mfrow=c(1,2))
+ kn <- seq(0, 36, dk)
+ pk <- glm(cbind(lex.Xst == "Dead",
+                  lex.dur) ~ Ns(tfl, knots = kn),
+            family = poisreg,
+              data = Sl)
+ matshade(prf$tfl, ci.pred(pk, prf),
+           plot = TRUE, log = "y", lwd = 3, ylim = c(0.01,1))
+ rug(kn, lwd=3)
+
+ matshade(prf$tfl, ci.surv(pk, prf, intl = 0.2) ,
+           plot = TRUE, lwd = 3, ylim = 0:1)
+ lines(km, lwd = 2)
+ }
```

```
> zz(12)
```

```
> zz(2)
```

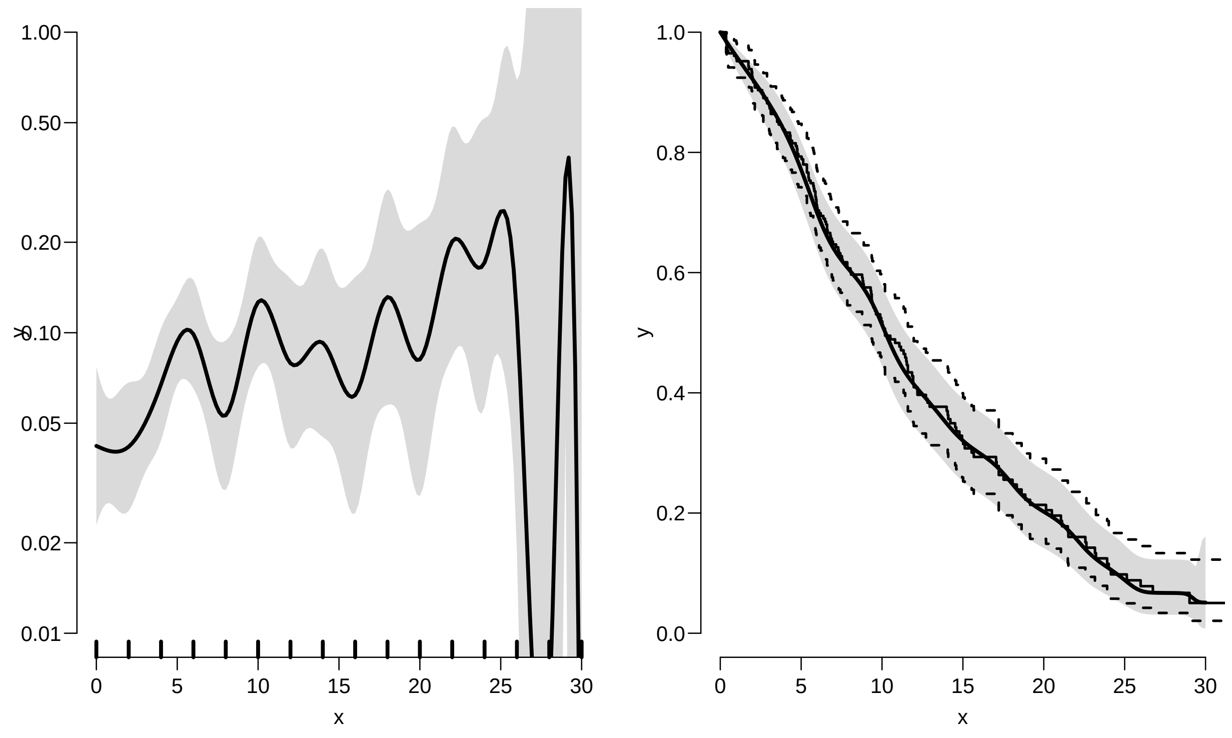

Figure 1.3: *Hazard (left) and survival (right) comparing a parametric model with knots every 2 months and the Kaplan-Meier estimator.*

`../graph/surv-knots2`

You will see that the more knots you include, the closer the parametric estimate gets to the Kaplan-Meier estimator. But also that the estimated underlying hazard becomes increasingly silly.

The ultimate silliness is of course achieved when we arrive at the Kaplan-Meier estimator, so the absence of the underlying hazard is most convenient.

1.2 Competing risks: `DMlate`

1. Competing risks: `dodth`, `doins`, `dooad`
2. `Lexis` object to `dodth`, `dox`
3. `mcutLexis` for the two competing risks
4. Aalen-Johansen estimator via `survfit`
5. Parametric rates
6. Cumulative risks from parametric rates
7. Cumulative risks fro `survival::survfit` - Aalen-Johansen estimator.

1.2.1 Paraphernalia

It is advisable to load all packages needed at the start:

```
> library(survival)
> library(Epi)
> library(popEpi)
> # popEpi::splitMulti returns a data.frame rather than a data.table
> options("popEpi.datatable" = FALSE)
> library(tidyverse)
```

1.2.2 Data

This exercise follows quite closely the section on competing risks in “Epidemiology with R”, pp. 207 and 210 ff. With the major exception that we will use the function `ci.Crisk`, which was not available in the *Epi* package when the book was written.

We shall use the `DMlate` dataset which is a random sample of Danish diabetes patients, with dates of birth, diabetes, OAD start, insulin start and death.

We want to look at the event “start of OAD”, which occurs at `dooad`, while taking death as competing event into account. This means that we want to address the question of the probability of starting OAD, while taking death into account. Essentially estimating the probability of being in each of the states `DM`, `OAD` and `Dead`, where `OAD` means “started OAD and either alive or dead after this” and `Dead` means “dead without starting OAD”.

1. Load the `DMlate` data from the `Epi` package, and for ease of calculation restrict to a random sample of 2000 persons:

   ```
   > data(DMlate)
   > # str(DMlate)
   > set.seed(1952)
   > DMlate <- DMlate[sample(1:nrow(DMlate), 2000),]
   > str(DMlate)
   ```

```
'data.frame':        2000 obs. of  7 variables:
 $ sex  : Factor w/ 2 levels "M","F": 2 1 2 1 1 1 1 1 1 1 ...
 $ dobth: num  1964 1944 1957 1952 1952 ...
 $ dodm : num  2003 2006 2008 2007 2003 ...
 $ dodth: num  NA NA NA NA NA NA NA NA NA NA ...
 $ dooad: num  NA 2006 NA 2007 2006 ...
 $ doins: num  NA NA NA 2008 NA ...
 $ dox  : num  2010 2010 2010 2010 2010 ...

> head(DMlate)

       sex    dobth     dodm dodth    dooad    doins      dox
70126    F 1963.591 2003.481    NA       NA       NA 2009.997
235221   M 1944.127 2005.644    NA 2005.778       NA 2009.997
230872   F 1956.790 2007.886    NA       NA       NA 2009.997
138167   M 1952.355 2006.969    NA 2006.969 2008.026 2009.997
406109   M 1952.240 2003.361    NA 2005.852       NA 2009.997
72438    M 1978.758 2001.948    NA       NA 2001.967 2009.997
```

2. Define a `Lexis` object with the total follow up for each person:

```
> Ldm <- 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  1  rows with duration of follow up < tol

> summary(Ldm)

Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 1521  478      1999      478   10742.34      1999
```

Then subdivide the follow-up at the date of OAD, using `dooad`:

```
> Cdm <- cutLexis(Ldm,
+                cut = Ldm$dooad,
+          timescale = "per",
+          new.state = "OAD")
> summary(Cdm)

Transitions:
     To
From   DM  OAD Dead  Records:  Events: Risk time:  Persons:
  DM  685  634  226      1545      860    5414.29      1545
  OAD   0  836  252      1088      252    5328.05      1088
  Sum 685 1470  478      2633     1112   10742.34      1999
```

In this context we are not interested in what goes on after OAD so we only keep follow-up in state `DM` (note that we must use `subset` because `filter` does not have a method for `Lexis` objects):

```
> Adm <- subset(Cdm, lex.Cst == "DM")
> summary(Adm)
```

```
Transitions:
     To
From  DM OAD Dead  Records:  Events: Risk time:  Persons:
  DM 685 634  226      1545      860    5414.29      1545
```

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

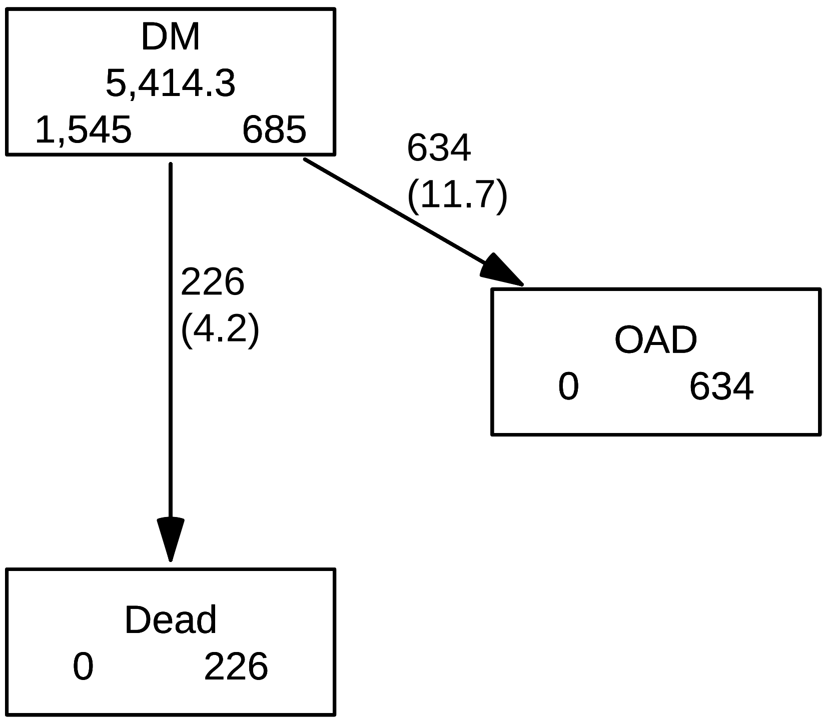

Figure 1.4: *Competing risks set-up for events* `OAD` *and* `Dead`.
`../graph/cmpr-boxCR`

As shown in figure 1.4 we now have a traditional competing risks set-up, with some 1500 DM patients starting without OAD, and where the quantity of interest is the probability of starting drug treatment, and the `OAD` state here means "having been on pharmaceutical treatment, disregarding subsequent death". The other event considered is `Dead` which here means "dead without initiating pharmaceutical treatment".

3. We can compute the (correct) counterpart of the survival function for this competing risks setup. The survival function gives the probability of being alive, and the complement is the probability of being dead, so the probabilities of being in each of the Alive/Dead states.

   `survfit` can do the corresponding calculation for the three states in the figure; the requirements are: 1) the third argument to the `Surv` function is a factor and 2) an `id` argument is given, pointing to an id variable that links together records belonging to the same person. The latter is superfluous in this case because there is only one record for each person, but even it is required by the function

```
> levels(Adm$lex.Xst)
```

```
[1] "DM"   "OAD"  "Dead"
```

```
> m3 <- survfit(Surv(tfd,
+                    tfd + lex.dur,
+                    lex.Xst) ~ 1,
+               data = Adm,
+                 id = lex.id)
> names(m3)
```

```
 [1] "n"            "time"         "n.risk"       "n.event"      "n.censor"     "pstate"
 [7] "p0"           "cumhaz"       "std.err"      "sp0"          "logse"        "transition
[13] "conf.int"     "conf.type"    "lower"        "upper"        "conf.type"    "conf.int"
[19] "states"       "type"         "call"
```

```
> m3$states
```

```
[1] "(s0)" "OAD"  "Dead"
```

```
> head(cbind(time = m3$time, m3$pstate))
```

```
            time
[1,] 0.002737851 0.9987055 0.001294498 0.0000000000
[2,] 0.005475702 0.9928803 0.006472492 0.0006472492
[3,] 0.008213552 0.9889968 0.009061489 0.0019417476
[4,] 0.010951403 0.9877023 0.009708738 0.0025889968
[5,] 0.013689254 0.9838188 0.013592233 0.0025889968
[6,] 0.016427105 0.9805825 0.016828479 0.0025889968
```

   Because `lex.Xst` is a factor, `survfit` will compute the Aalen-Johansen estimator of being in a given state and place the probabilities in the matrix `m3$pstate`; the times these refer to are in the vector `m3$time`. These are measured in years since diabetes, because `tfd` is in units of years,

   Explore the object `m3`; start by using `names(m3)`.

4. The `m3$pstate` contains the Aalen-Johansen probabilities of being in the `Alive`, having left to the `OAD`, resp. `Dead` state.

   Plot the three curves in the same graph (use for example `matplot`). Add the confidence limits.

5. These three curves have sum 1, so basically this is a way of distributing the probabilities across states at each time. It is therefore natural to stack the probabilities, which can be done by `stackedCIF`:

```
> par( mfrow=c(1,2) )
> matplot(m3$time, m3$pstate,
+         type="s", lty=1, lwd=4,
+         col=c("ForestGreen","red","black"),
+         xlim=c(0,15), xaxs="i",
+         ylim=c(0,1), yaxs="i" )
> stackedCIF(m3, lwd=3, xlim=c(0,15), xaxs="i", yaxs="i" )
> text( rep(12,3), c(0.9,0.3,0.6), levels(Cdm) )
> box()
```

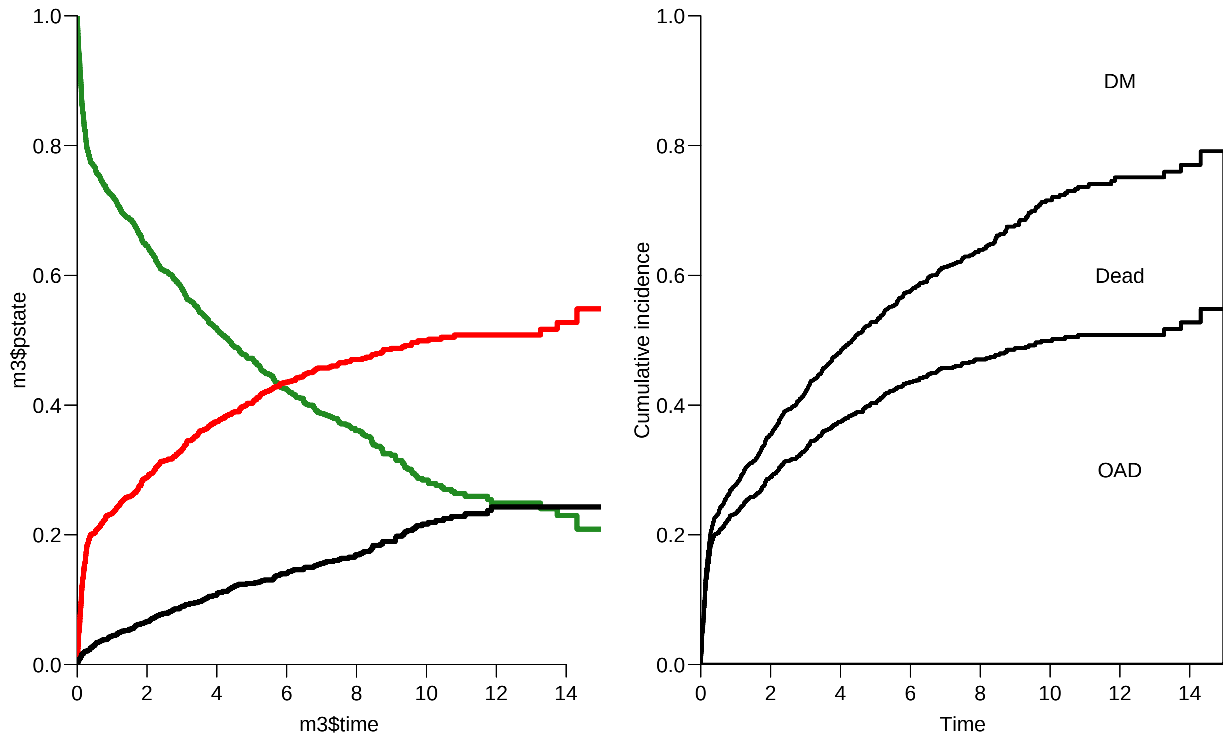

Figure 1.5: *Separate state probabilities (left) and stacked state probabilities (right). In the left panel,* `Alive` *is green,* `OAD` *is red and* `Dead` *is black.* `../graph/cmpr-surv2`

6. What do you get if you replace "~ 1" by "~ sex" in the call to `survfit`?

1.2.3 What not to do

A very common error is to use a *partial* outcome such as OAD, when there is a competing type of event, in this case Dead. If that is ignored and a traditional survival analysis is made *as if* OAD were the only possible event, we will have a substantial *over*estimate of the cumulative probability of going on drug. Here is an illustration of this erroneous approach:

```
> m2 <- survfit( Surv(tfd,
+                     tfd + lex.dur,
+                     lex.Xst == "OAD" ) ~ 1, data = Adm)
> M2 <- survfit( Surv(tfd,
+                     tfd + lex.dur,
+                     lex.Xst == "Dead") ~ 1, data = Adm)
> par( mfrow=c(1,2) )
> mat2pol(m3$pstate, c(2,3,1), x = m3$time,
+         col = c("red", "black", "transparent"),
+         xlim=c(0,15), xaxs="i",
+         yaxs = "i", xlab = "time since DM", ylab = "" )
>   lines(m2$time, 1 - m2$surv, lwd = 3, col = "red" )
> mat2pol(m3$pstate, c(3,2,1), x = m3$time, yaxs = "i",
+         col = c("black","red","transparent"),
+         xlim=c(0,15), xaxs="i",
+         yaxs = "i", xlab = "time since DM", ylab = "" )
>   lines(M2$time, 1 - M2$surv, lwd = 3, col = "black" )
```

The first two statements calculate the survival as if only OAD, respectively Dead were the only way of exiting the state Alive. The mat2pol (matrix to polygon) takes the columns of state probabilities from the survfit object m3 that contains the correctly modeled probabilities and plot them as coloured areas stacked; the second argument to mat2pol is the order in which they should be stacked. The lines plot the wrongly computed cumulative risks (from m2 and M2) — in order to find these we fish out the surv component from the survfit objects.

1.3 Modeling cause specific rates

There is nothing wrong with modeling the cause-specific event-rates, the problem lies in how you transform them into probabilities. The relevant model for a competing risks situation normally consists of separate models for each of the cause-specific rates. Not for technical or statistical reasons, but for substantial reasons; it is unlikely that rates of different types of event (OAD initiation and death, say) depend on time in the same way.

7. Now model the two sets of rates by parametric models; this must be based on a time-split data set — choose whether you want to use the gam or the glm approach:

```
> Sdm <- splitMulti(Adm, tfd = seq(0,20,0.1) )
> summary(Adm)
```

```
Transitions:
     To
```

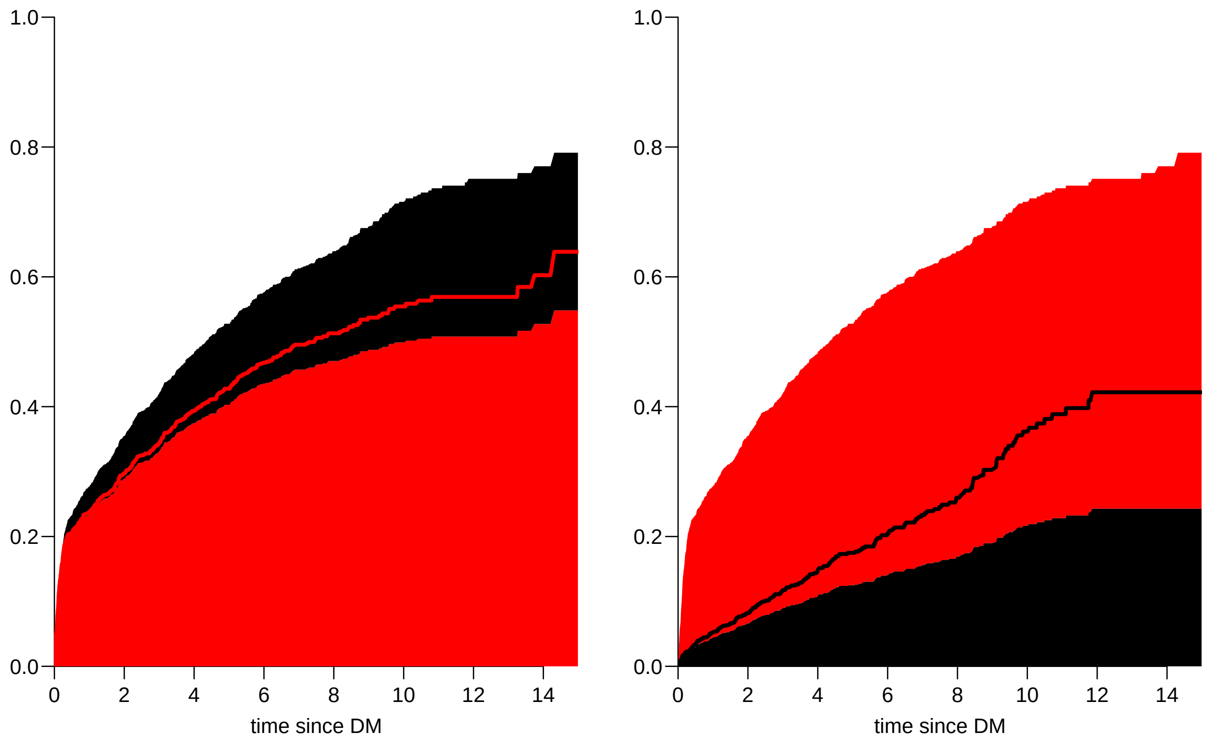

Figure 1.6: *Stacked state probabilities* `Alive` *is white,* `OAD` *is red and* `Dead` *is black. The red line in the left panel is the wrong (but often computed) "cumulative risk" of* `OAD`*, and the black line in the right panel is the wrong (but often computed) "cumulative risk" of* `Death`*. The black and the red areas in the two plots represent the correctly computed probabilities; they have the same size in both panels, only they are stacked differently.* `../graph/cmpr-surv3`

```
From  DM OAD Dead  Records:  Events: Risk time:  Persons:
  DM 685 634  226      1545      860    5414.29      1545

> summary(Sdm)

Transitions:
     To
From    DM OAD Dead  Records:  Events: Risk time:  Persons:
  DM 54064 634  226     54924      860    5414.29      1545

> gla <- gam.Lexis(Sdm, ~ s(tfd, k = 5), from = "DM", to = "OAD" )

mgcv::gam Poisson analysis of Lexis object Sdm with log link:
Rates for the transition: DM->OAD

> gma <- gam.Lexis(Sdm, ~ s(tfd, k = 5), from = "DM", to = "Dead" )

mgcv::gam Poisson analysis of Lexis object Sdm with log link:
Rates for the transition: DM->Dead
```

8. As an alternative to the gam model that uses penalized splines, we can use natural splines in a non-penalized model using glm. The glm requires a set of pre-specified knots for the time variable, where the specification should be (partially) guided by the location on the times of the events:

```
> round(cbind(
+ with(subset(Sdm, lex.Xst == "OAD" ), quantile(tfd + lex.dur, 0:10/10)),
+ with(subset(Sdm, lex.Xst == "Dead"), quantile(tfd + lex.dur, 0:10/10))),
+ 3)
```

```
       [,1]   [,2]
0%    0.003  0.005
10%   0.038  0.129
20%   0.095  0.507
30%   0.142  1.083
40%   0.239  1.730
50%   0.534  2.552
60%   1.268  3.584
70%   2.199  4.490
80%   3.373  6.196
90%   5.213  8.471
100% 14.311 11.858
```

We see that the OAD occur earlier than Dead, so we choose the knots a bit earlier:

```
> okn <- c(0,0.5,3,6)
> dkn <- c(0,2.0,5,9)
> gll <- glm.Lexis(Sdm, ~ Ns(tfd, knots = okn), from = "DM", to = "OAD" )
```

```
stats::glm Poisson analysis of Lexis object Sdm with log link:
Rates for the transition: DM->OAD
```

```
> gml <- glm.Lexis(Sdm, ~ Ns(tfd, knots = dkn), from = "DM", to = "Dead")
```

```
stats::glm Poisson analysis of Lexis object Sdm with log link:
Rates for the transition: DM->Dead
```

9. With models for the two rates out of the DM state we can derive the estimated rates from the two models for rates by time by using a prediction frame, nd:

```
> int <- 0.01
> nd <- data.frame(tfd = seq(0, 15, int))
> lama <- ci.pred(gla, nd)
> mrta <- ci.pred(gma, nd)
> laml <- ci.pred(gll, nd)
> mrtl <- ci.pred(gml, nd)
```

Now plot the estimated rates, in this case the gam models with dotted and glm models with full lines; mortality with black and OAD rates with red:

```
> matshade(nd$tfd,
+          cbind(lama, mrta, laml, mrtl) * 100,
+          plot = TRUE,
+          log = "y", ylim = c(2, 20),
+          col = rep(c("red","black"), 2),
+          lty = rep(c("21","solid"), each = 2), lwd = 3, lend = "butt")
```

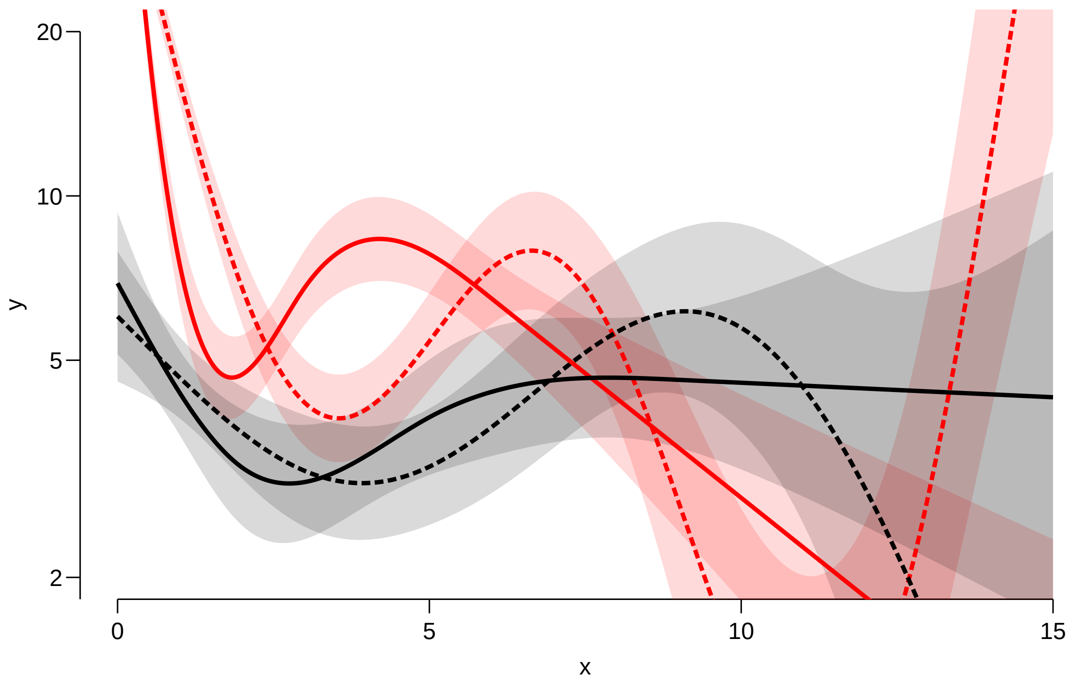

Figure 1.7: *Mortality rates (black) and OAD-rate (red), from* **`glm`** *model with natural splines (full lines) and* **`gam`** *models with penalized splines (dotted lines).*

`../graph/cmpr-OAD-mort`

1.3.1 Integrals with R

Based on these parametric models we can estimate the cumulative risks of being in each of the states, but also the expected time time spent in each state. The theory of these involves calculation of integrals of the rate functions. Integrals looks scary to many people, but they are really just areas under curves.

The key is to understand how a curve is represented in R. A curve representing the function μ is just a set of a vector ts and a vector $y = \mu(t)$s. When we have a model such as `gml` above that estimates the mortality as a function of time (`tfd`), we can get a representation of this by first choosing the timepoints, say from 0 to 15 years in steps of 0.01 year ($\approx$ 4 days), and put this in a dataframe with the variable name from the model::

```
> t <- seq(0, 15, 0.01)
> prfrm <- data.frame(tfd = t)
> mu <- ci.pred(gml, prfrm)[,1]
> head(cbind(t, mu))
```

```
     t         mu
1 0.00 0.06919036
2 0.01 0.06885302
3 0.02 0.06851733
4 0.03 0.06818330
5 0.04 0.06785093
6 0.05 0.06752022
```

This is a representaion of the points $(t, \mu(t))$; if we want the integral of mu over the interval $[0, 5]$, say, $M(5) = \int_0^5 \mu(s)\,\mathrm{d}s$, we just need the area under the curve. Each t represents an endpoint of an interval, what we want in order to compute the area under the curve is the *width* of each interval, `diff(t)`, multiplied by the average of the function values at the ends of each interval. (This goes under the name of the ”trapezoidal formula”). So we need a small function to compute midpoints between successive values in a vector:

```
> mid <- function(x) x[-1] - diff(x) / 2
> mid(c(1:5,7,10))
[1] 1.5 2.5 3.5 4.5 6.0 8.5
```

Note that `mid(x)` is a vector that is 1 shorter than the vector `x`, just as `diff(x)` is.

So if we want the integral over the period 0 to 5 years, we want the sum over the first 500 intervals, corresponding to teh first 501 interval endpoints:

```
> sum(diff(t[1:501]) * mid(mu[1:501]))
[1] 0.1896222
```

So now we have computed $\int_0^5 \mu(s)\,\mathrm{d}(s)$.

In pratice we will want the integral `function` of μ, so for every t we want $M(t) = \int_0^t \mu(s)\,\mathrm{d}(s)$. This is easily accomplished by the function `cumsum`:

```
> Mu <- c(0, cumsum(diff(t) * mid(mu)))
> head(cbind(t, Mu))
     t           Mu
  0.00 0.0000000000
2 0.01 0.0006902169
3 0.02 0.0013770686
4 0.03 0.0020605718
5 0.04 0.0027407429
6 0.05 0.0034175987
```

Note the first value which is the integral from 0 to 0, so by definition 0.

1.3.2 Cumulative risks

Here is the theory where we need integration: The cumulative risk of `OAD` at time t is:

$$R_{\mathtt{OAD}} = \int_0^t \lambda(u)S(u)\,\mathrm{d}u = \int_0^t \lambda(u)\exp\Big(-\int_0^u \lambda(s) + \mu(s)\,\mathrm{d}s\Big)\,\mathrm{d}u$$

where λ is the rate of `OAD` (`lam`), and μ the mortality rate (`mrt`). A similar formula is obtained for the cumulative risk of `Dead` (that is “dead without OAD”), by exchanging λ and μ.

The prectical calculation of these quantities are on pages 214–5 of “Epidemiology with R”.

10. This means that if we have estimates of λ and μ as functions of time, we can derive the cumulative risks. In practice this will be by numerical integration; compute the rates at closely spaced intervals and evaluate the integrals as sums. This is easy, but what is not so easy is to come up with confidence intervals for the cumulative risks.

    Confidence intervals are most conveniently produced by simulation ("parametric bootstrap" as some say):

    (a) generate a random vector from the multivariate normal distribution with mean equal to the parameters of the model, and variance-covariance equal to the estimated variance-covariance of the parameter estimates (the Hessian as it is called).
    (b) use this to generate a simulated set of rates evaluated a closely spaced times
    (c) use these in numerical integration to derive state probabilities at these times
    (d) repeat 1000 times, say, to obtain 1000 sets of state probabilities
    (e) use these to derive confidence intervals for the state probabilities as the 2.5 and 97.5 percentiles of the state probabilities at each time

    This machinery is implemented in the function `ci.Crisk`

```
> cR <- ci.Crisk(mods = list(OAD = gll,
+                            Dead = gml),
+                       nd = nd)
```

```
NOTE: Times are assumed to be in the column tfd at equal distances of 0.01
```

```
> str(cR)
```

```
List of 4
 $ Crisk: num [1:1501, 1:3, 1:3] 1 0.991 0.983 0.975 0.968 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:1501] "0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:3] "Surv" "OAD" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:1501, 1:2, 1:3] 0 0.000692 0.001374 0.002048 0.002713 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:1501] "0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:2] "Dead" "Dead+OAD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:1501, 1:3, 1:3] 0 0.00996 0.01983 0.02963 0.03934 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:1501] "0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:3] "Surv" "OAD" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:1501] 0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 ...
 - attr(*, "int")= num 0.01
```

There are 4 components of the results, the three first are simply arrays with 2 or 3 functions of time with confidence intervals.

So now plot the cumulative risks of being in each of the states (the `Crisk` component):

```
> matshade(cR$time, cbind(cR$Crisk[,1,],
+                         cR$Crisk[,2,],
+                         cR$Crisk[,3,]), plot = TRUE,
+          lwd = 2, col = c("limegreen","red","black"))
```

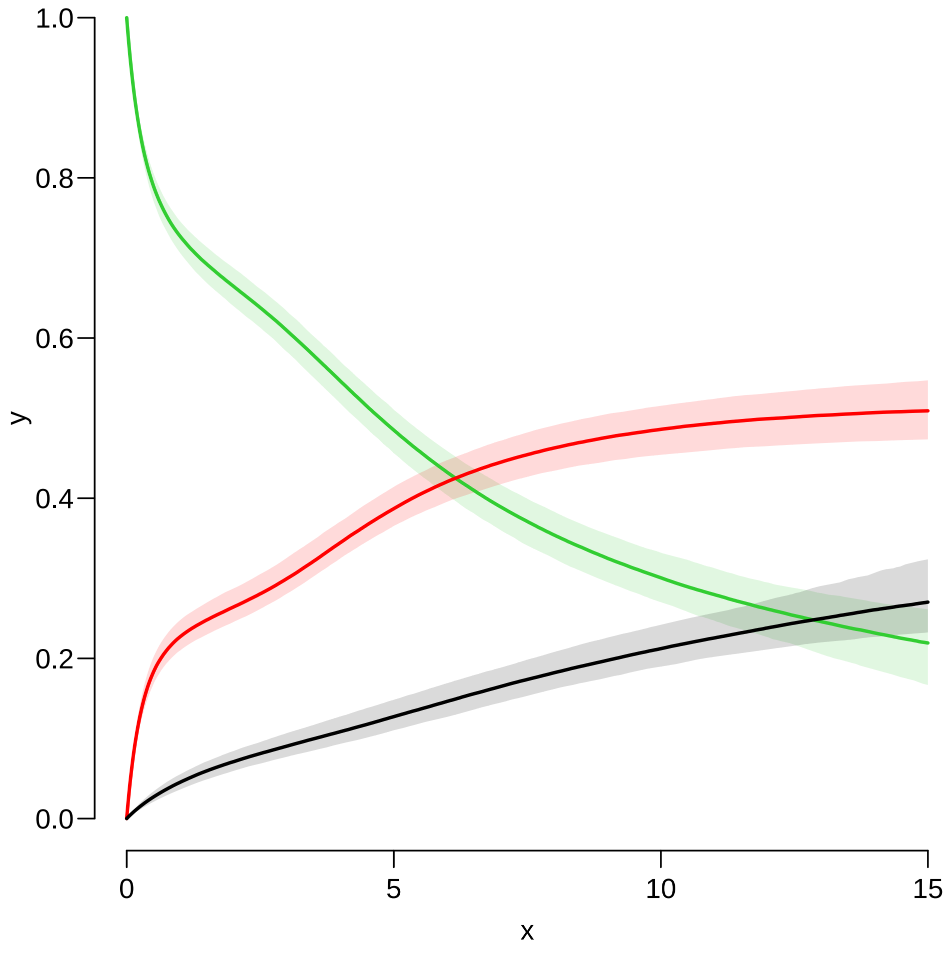

Figure 1.8: *Cumulative risks of being in each of the states* `DM` *(green),* `OAD` *(red) and* `Dead`
`../graph/cmpr-crisk`

11. Plot the stacked probabilities (`matrix` 2 `polygons`):

```
> mat2pol(cR$Crisk[,3:1,1],
+          col = c("forestgreen","red","black")[3:1])
```

The component `Srisk` has the confidence limits of the stacked probabilities, add these to the plot, for example by semi-transparent shades or dotted lines,

If you are really entrepreneurial, devise a function that will take the `Srisk` component of `cR` and produce a stacked plot with shaded confidence limits; here is the stacked plot:

```
> matshade(cR$time, cbind(cR$Srisk[,1,],
+                         cR$Srisk[,2,]), plot = TRUE,
+          lwd = 2, col = c("black","red"),
+          ylim = 0:1, yaxs = "i")
```

Note the `yaxs = "i"`...

You may want to look at `adjustcolor` or `rgb` to see how to make semi-transparent colours.

12. It is not only the cumulative risks of being in different states that my be of interest, the *integrals* — area under the cumulative risk curves are of interest too. The cumulative risks are probabilities, so dimensionless, which means that integrals of these along the time-axis will have dimension time; they will represent the expected time spent in each of the states.

    The areas between the lines (up to say 10 years) are `expected sojourn times`, that is:

    - expected years alive without OAD
    - expected years lost to death without OAD
    - expected years after OAD, including years dead after OAD

    Not all of these are of direct relevance; actually only the first may be so. They are available (with simulation-based confidence intervals) in the component of `cR`, `Stime` (Sojourn `time`).

    A relevant quantity would be the expected time alive without OAD during the first 5, 10 and 15 years:

```
> str(cR$Stime)

 num [1:1501, 1:3, 1:3] 0 0.00996 0.01983 0.02963 0.03934 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfd   : chr [1:1501] "0" "0.01" "0.02" "0.03" ...
  ..$ cause: chr [1:3] "Surv" "OAD" "Dead"
  ..$       : chr [1:3] "50%" "2.5%" "97.5%"

> round(cR$Stime[c("5","10","15"),"Surv",], 1)

tfd  50% 2.5% 97.5%
  5  3.2  3.1   3.3
  10 5.1  4.9   5.3
  15 6.4  6.0   6.8
```

13. We can also compute the expected fraction of the first 5, 10, 15 years alive:

```
> (mY <- matrix(rep(1:3 * 5, 3), 3, 3))

     [,1] [,2] [,3]
[1,]    5    5    5
[2,]   10   10   10
[3,]   15   15   15
```

```
> round(100 * cR$Stime[c("5","10","15"),"Surv",] / mY, 1)
```

```
tfd   50% 2.5% 97.5%
  5  64.7 62.5  66.8
  10 51.3 49.1  53.4
  15 42.7 40.3  45.0
```

This can also be shown as a function of time; how large a fraction of the first t time can a person expect to be alive, for t ranging from 0 to 15 years:

```
> matshade(cR$time, cR$Stime[,"Surv",] /
+                   cbind(cR$time, cR$time, cR$time) * 100,
+          plot=TRUE,
+          ylim = 0:1*100, yaxs = "i", xaxs = "i")
```

Amend the plot with proper axis labels.

1.4 Multistate models: steno2

1. `Lexis` object for `steno2`
2. `rcut` using `st2alb`
3. `boxes` to get an overview - revise data to avoid `norm`→`mac`
4. Mortality rates: 3 initial states, 2 outcomes
5. `addCov` using `st2clin`
6. Mortality by `chol` and `bp`
7. Transitions between micro vascular complications states, one model, use `stack`
8. State probabilities for different *baseline* values of sex and age.
9. Limitations in using clinical measurements as time-dependent variables without a model for the clinical variables
10. State probabilities only using time since entry; comparison to Aalen-Johansen approach from `survival`
11. . . . comparison of smooth modeling to Cox-models, using the `mstate` machinery.

1.4.1 Paraphernalia

First we load the relevant packages and set options:

```
> library(Epi)
> library(popEpi)
> # popEpi::splitMulti returns a data.frame rather than a data.table
> options("popEpi.datatable" = FALSE)
> library(tidyverse)
```

1.4.2 Lexis object for steno2

1. Bring in the `steno2` dataset, and convert dates to `cal.yr` to get a natural unit of time (years—365.25 days, that is). Because of the way data were anonymized, the `doEnd` is not perfectly aligned to `doDth`, which we remedy on the fly by resetting `doEnd` if a `doDth` is known.

   ```
   > data(steno2)
   > steno2 <- transform(cal.yr(steno2),
   +                     doEnd = ifelse(!is.na(doDth),
   +                                    doDth,
   +                                    doEnd))
   > str(steno2)
   ```

```
'data.frame':       160 obs. of  14 variables:
 $ id      : num  1 2 3 4 5 6 7 8 9 10 ...
 $ allo    : Factor w/ 2 levels "Int","Conv": 1 1 2 2 2 2 2 1 1 1 ...
 $ sex     : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 1 2 2 2 ...
 $ baseCVD : num  0 0 0 0 0 1 0 0 0 0 ...
 $ deathCVD: num  0 0 0 0 1 0 0 0 1 0 ...
 $ doBth   : 'cal.yr' num  1932 1947 1943 1945 1936 ...
 $ doDM    : 'cal.yr' num  1991 1982 1983 1977 1986 ...
 $ doBase  : 'cal.yr' num  1993 1993 1993 1993 1993 ...
 $ doCVD1  : 'cal.yr' num  2014 2009 2002 1995 1994 ...
 $ doCVD2  : 'cal.yr' num  NA 2009 NA 1997 1995 ...
 $ doCVD3  : 'cal.yr' num  NA 2010 NA 2003 1998 ...
 $ doESRD  : 'cal.yr' num  NaN NaN NaN NaN 1998 ...
 $ doEnd   : num  2015 2015 2002 2003 1998 ...
 $ doDth   : 'cal.yr' num  NA NA 2002 2003 1998 ...
```

2. Start by setting up a `Lexis` data frame for the entire observation time for each person; from entry (`doBase`, date of baseline) to exit, `doEnd`. Note that we call the initial state `Mic`(roalbuminuria), because all patients in the Steno2 study had this status—it was one of the inclusion criteria:

```
> L2 <- Lexis(entry = list(per = doBase,
+                          age = doBase - doBth,
+                          tfi = 0),
+              exit = list(per = doEnd),
+       exit.status = factor(deathCVD + !is.na(doDth),
+                            labels=c("Mic","D(oth)","D(CVD)")),
+                id = id,
+              data = steno2)

NOTE: entry.status has been set to "Mic" for all.

> summary(L2, t = TRUE)

Transitions:
     To
From  Mic D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic  67     55     38       160       93    2420.91       160

Timescales:
per age tfi
 ""  ""  ""

> boxes(L2, boxpos = TRUE, show.BE = TRUE)
```

How many deaths are there in the cohort?

How many person-years?

3. In this set-up we can study the CVD and the non-CVD mortality rates, a classical competing risks problem, but we want in particular to see how the mortality rates depend on albuminuria status.

In order to allocate follow-up (person-time and events) to *current* albuminuria status we need to know when the persons change status; this is recorded in the data frame `st2alb`.

We will cut the follow-up at possibly several times per person, so will use the function `rcutLexis` (`r`ecurrent `cut`s), which requires a data frame of transitions with columns `lex.id`, `cut` and `new.state` — see `?rcutLexis`.

We change the scale of the date of transition to year by `cal.yr` (to align with the `per` variable in `L2`), rename the id variable to `lex.id` and the date variable `doTr` to `cut`

```
> data(st2alb)
> cut2 <- rename(cal.yr(st2alb),
+                lex.id = id,
+                   cut = doTr,
+             new.state = state)
> str(cut2)
```

```
'data.frame':        563 obs. of  3 variables:
 $ lex.id   : num  1 1 1 1 1 2 2 2 2 2 ...
 $ cut      : 'cal.yr' num  1993 1995 2000 2002 2007 ...
 $ new.state: Factor w/ 3 levels "Norm","Mic","Mac": 2 1 2 1 2 1 2 3 2 2 ...
```

How many persons are in the `cut2` data frame?

```
> with(cut2, addmargins(table(table(lex.id))))
```

```
  1   2   3   4   5 Sum
  4  25  40  46  41 156
```

Explain the entries in this table.

4. Now cut at intermediate transition times (note that `rcutLexis` assumes that values in the `cut` column refer to the `per` timescale by default since it is the first of the time scales):

```
> L3 <- rcutLexis(L2, cut2)
> summary(L3)
```

```
Transitions:
     To
From   Mic Norm Mac D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic  299   72  65     27     13       476      177    1383.56       160
  Norm  31   90   5     14      7       147       57     608.75        69
  Mac   20    3  44     14     18        99       55     428.60        64
  Sum  350  165 114     55     38       722      289    2420.91       160
```

```
> boxes(L3, boxpos = TRUE, cex = 0.8)
```

Note that there are transitions both ways between all three of `Norm`, `Mic` and `Mac`, which is a bit illogical, since we have a natural ordering of states: `Norm` < `Mic` < `Mac`

5. In order to remedy this anomaly we find all transitions `Norm` → `Mac` and provide a transition `Norm` → `Mic` in between. And of course similarly for transitions `Mac` → `Norm`.

   The relevant "jump" transitions are easily found:

```
> (jump <-
+ subset(L3, (lex.Cst == "Norm" & lex.Xst == "Mac") |
+              (lex.Xst == "Norm" & lex.Cst == "Mac"))[,
+         c("lex.id", "per", "lex.dur","lex.Cst", "lex.Xst")])

    lex.id      per    lex.dur lex.Cst lex.Xst
291     70 1999.487  2.6748802     Mac    Norm
353     86 2001.759 12.8158795    Norm     Mac
506    130 2000.910  1.8781656     Mac    Norm
511    131 1997.756  4.2354552    Norm     Mac
525    136 1997.214  0.4709103     Mac    Norm
526    136 1997.685  4.2436687    Norm     Mac
654    171 1996.390  5.3388090    Norm     Mac
676    175 2004.585  9.8836413    Norm     Mac
```

   What we need to do for these "jumps" is to provide an extra transition to `Mic` at a time during the stay in either `Norm` or `Mac`, i.e. between `per` and `per` + `lex.dur` in these records; we choose a random time in the middle 80% between the dates:

```
> set.seed(1952)
> xcut <- select(transform(jump,
+                              cut = per + lex.dur * runif(per, 0.1, 0.9),
+                         new.state = "Mic"),
+                   c(lex.id, cut, new.state))
> xcut

    lex.id      cut new.state
291     70 2001.789       Mic
353     86 2012.232       Mic
506    130 2001.488       Mic
511    131 2001.032       Mic
525    136 1997.610       Mic
526    136 2000.780       Mic
654    171 1997.057       Mic
676    175 2013.472       Mic
```

   How many extra records will be used for cutting follow-up?

6. Now make extra cuts at these dates using `rcutLexis` with `xcut` on the `L3` object:

```
> L4 <- rcutLexis(L3, xcut)
> summary(L4)

Transitions:
     To
From   Mic Norm Mac D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic  312   72  65     30     14       493      181    1437.39       160
  Norm  35   90   0     13      6       144       54     581.83        66
  Mac   22    0  41     12     18        93       52     401.70        60
  Sum  369  162 106     55     38       730      287    2420.91       160
```

We see that there are no transitions directly between `Norm` and `Mac` in `L4`, so we can make an intelligible plot of the transitions:

```
> boxes(L4, boxpos = list(x = c(20,20,20,80,80),
+                         y = c(50,80,20,75,25)),
+           show.BE = "nz",
+           scale.R = 100,
+           cex = 0.8)
```

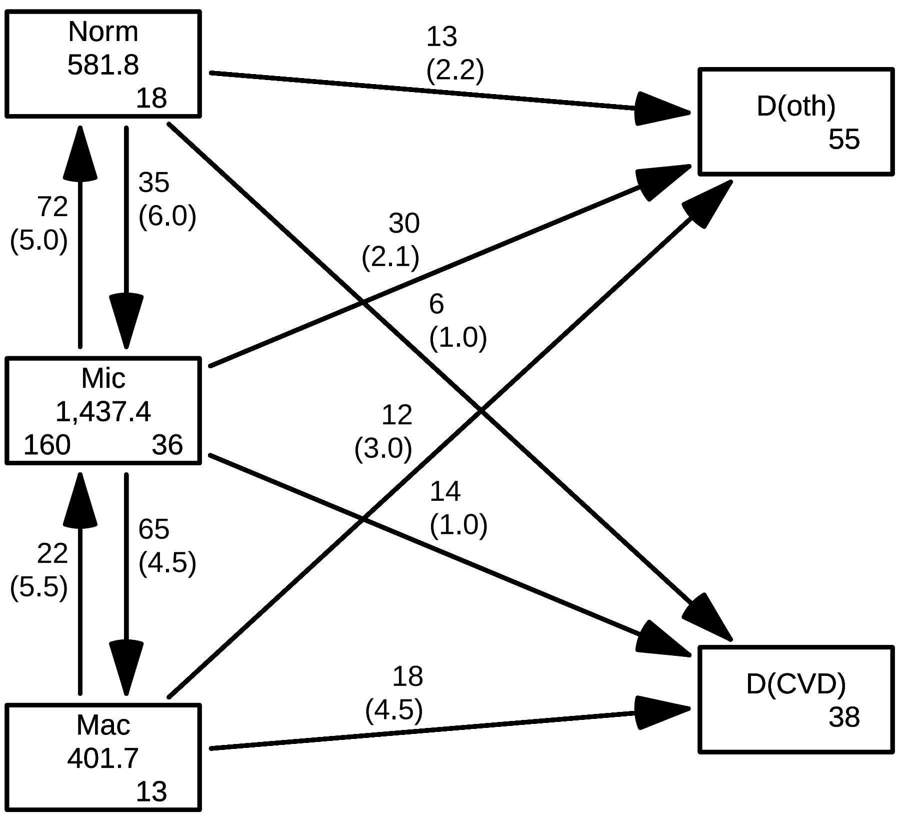

Figure 1.9: *Transitions between states in the Steno2 study.*

`../graph/ms-b4`

Describe the overall effect of albuminuria on the two mortality rates.

With this multistate model (well, there is no model yet) set up we can look at mortality rates and see how they depend on the current albuminuria state, or look at the transition rates between the different albuminuria states and assess how these depend on various other covariates.

1.4.3 Mortality rates: 3 initial states, 2 outcomes, multiple time scales

7. First we look at how the overall mortality depends on albuminuria status. We will model the mortality rates with parametric functions, so we need to split the dataset along some time scale; we will use 3 month intervals (they should be sufficiently small to accommodate an assumption of constant rates in the interval):

```
> S4 <- splitMulti(L4, tfi = seq(0, 25, 1/4))
> summary(L4)
```

```
Transitions:
     To
From   Mic Norm Mac D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic  312   72  65     30     14       493      181    1437.39       160
  Norm  35   90   0     13      6       144       54     581.83        66
  Mac   22    0  41     12     18        93       52     401.70        60
  Sum  369  162 106     55     38       730      287    2420.91       160
```

```
> summary(S4)
```

```
Transitions:
     To
From    Mic Norm  Mac D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic  5986   72   65     30     14      6167      181    1437.39       160
  Norm   35 2418    0     13      6      2472       54     581.83        66
  Mac    22    0 1644     12     18      1696       52     401.70        60
  Sum  6043 2490 1709     55     38     10335      287    2420.91       160
```

We can then model the overall mortality rates as functions of age and duration (time since entry) using the defaults for `glm.Lexis` (this function call will trigger a warning):

```
> ma <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                       Ns(age, knots = seq(60, 75, 5)) +
+                       lex.Cst)
```

```
stats::glm Poisson analysis of Lexis object S4 with log link:
Rates for transitions: Mic->D(oth), Norm->D(oth), Mac->D(oth), Mic->D(CVD), Norm->D(CV
```

The warning here just tells you that you are modeling the occurrence of any type of death, so assuming that CVD and non-CVD death rates are identical, and assuming the mortality rates are proportional between states of albuminuria.

The `glm.Lexis` is just a convenience wrapper for:

```
> ma <- glm(cbind(lex.Xst %in% c("D(oth)", "D(CVD)") & lex.Cst != lex.Xst,
+                 lex.dur)
+             ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+               Ns(age, knots = seq(60, 75, 5)) +
+               lex.Cst,
+             family = poisreg,
+             data = subset(S4, lex.Cst %in% c("Norm","Mic","Mac")))
> # which gives the same as:
> ma <- glm((lex.Xst %in% c("D(oth)", "D(CVD)") & lex.Cst != lex.Xst)
+             ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+               Ns(age, knots = seq(60, 75, 5)) +
+               lex.Cst,
+             offset = log(lex.dur),
+             family = poisson,
+             data = subset(S4, lex.Cst %in% c("Norm","Mic","Mac")))
```

—note the difference between `poisreg` and `poisson` syntax.

The parameters are (exponentiated, so on the rate-scale):

```
> round(ci.exp(ma), 2)

                                  exp(Est.) 2.5%   97.5%
(Intercept)                            0.00 0.00    0.02
Ns(tfi, knots = seq(0, 20, 5))1        6.30 1.21   32.72
Ns(tfi, knots = seq(0, 20, 5))2        4.11 0.95   17.67
Ns(tfi, knots = seq(0, 20, 5))3       42.02 0.93 1904.98
Ns(tfi, knots = seq(0, 20, 5))4        0.50 0.16    1.53
Ns(age, knots = seq(60, 75, 5))1       2.06 0.99    4.32
Ns(age, knots = seq(60, 75, 5))2       4.59 2.46    8.57
Ns(age, knots = seq(60, 75, 5))3       3.91 2.11    7.25
lex.CstNorm                            1.04 0.61    1.79
lex.CstMac                             1.77 1.10    2.85
```

We see there is a higher mortality in the `Mac` state but no discernible difference between the `Mic` and the `Norm` states. It can be formally tested whether the three states carry the same mortality using a Wald test:

```
> Wald(ma, subset = "lex.Cst")

    Chisq      d.f.         P
6.1107777 2.0000000 0.0471044
```

So the mortality from the three states is not the same, but it is also quite clear that the mortality from state `Mac` is higher than the two other (surprise, surprise).

8. Now do the same analysis for the two causes of death separately, using the `to` argument to `glm.Lexis`:

```
> mo <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                       Ns(age, knots = seq(60, 75, 5)) +
+                       lex.Cst,
+                 to = "D(oth)")
```

```
stats::glm Poisson analysis of Lexis object S4 with log link:
Rates for transitions: Mic->D(oth), Norm->D(oth), Mac->D(oth)

> round(ci.exp(mo), 3)

                                 exp(Est.)  2.5%        97.5%
(Intercept)                          0.000 0.000 6.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1    111.549 2.334 5.331112e+03
Ns(tfi, knots = seq(0, 20, 5))2     29.787 1.287 6.892080e+02
Ns(tfi, knots = seq(0, 20, 5))3  22802.029 3.309 1.571478e+08
Ns(tfi, knots = seq(0, 20, 5))4      1.768 0.304 1.027600e+01
Ns(age, knots = seq(60, 75, 5))1     2.854 1.063 7.662000e+00
Ns(age, knots = seq(60, 75, 5))2     4.163 1.926 8.998000e+00
Ns(age, knots = seq(60, 75, 5))3     5.569 2.402 1.291500e+01
lex.CstNorm                          1.023 0.531 1.970000e+00
lex.CstMac                           0.999 0.505 1.977000e+00

> mC <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                        Ns(age, knots = seq(60, 75, 5)) +
+                        lex.Cst,
+                  to = "D(CVD)")

stats::glm Poisson analysis of Lexis object S4 with log link:
Rates for transitions: Mic->D(CVD), Norm->D(CVD), Mac->D(CVD)

> round(ci.exp(mC), 3)

                                 exp(Est.)  2.5%   97.5%
(Intercept)                          0.004 0.001   0.026
Ns(tfi, knots = seq(0, 20, 5))1      1.237 0.176   8.671
Ns(tfi, knots = seq(0, 20, 5))2      1.988 0.302  13.105
Ns(tfi, knots = seq(0, 20, 5))3      1.368 0.018 105.801
Ns(tfi, knots = seq(0, 20, 5))4      0.143 0.019   1.098
Ns(age, knots = seq(60, 75, 5))1     1.488 0.475   4.658
Ns(age, knots = seq(60, 75, 5))2     5.922 1.944  18.039
Ns(age, knots = seq(60, 75, 5))3     2.565 1.007   6.534
lex.CstNorm                          1.078 0.411   2.827
lex.CstMac                           3.520 1.722   7.194
```

What is the conclusion w.r.t. the effect of albuminuria state on the two mortality rates?

Can you make a formal test of a relevant hypothesis?

```
> Wald(mo, subset = "Cst")

      Chisq        d.f.           P
0.005164312 2.000000000 0.997421175

> Wald(mC, subset = "Cst")

       Chisq         d.f.            P
1.384601e+01 2.000000e+00 9.848646e-04
```

9. We can show how mortality rates look for persons currently in state `Mic` entering the study at ages 60, 65 and 70, as a function of current age. We need a prediction data frame, with values for all variables in the model:

```
> expand.grid(tfi = c(NA, seq(0, 20, 5)),
+             ain = c(60, 65, 70))[-1,]

   tfi ain
2    0  60
3    5  60
4   10  60
5   15  60
6   20  60
7   NA  65
8    0  65
9    5  65
10  10  65
11  15  65
12  20  65
13  NA  70
14   0  70
15   5  70
16  10  70
17  15  70
18  20  70

> prf <- transform(expand.grid(tfi = c(NA, seq(0, 20, 0.5)),
+                              ain = c(60, 65, 70))[-1,],
+                  age = ain + tfi,
+              lex.Cst = "Mic")
> head(prf)

  tfi ain  age lex.Cst
2 0.0  60 60.0     Mic
3 0.5  60 60.5     Mic
4 1.0  60 61.0     Mic
5 1.5  60 61.5     Mic
6 2.0  60 62.0     Mic
7 2.5  60 62.5     Mic

> matshade(prf$age, cbind(ci.pred(mo, prf),
+                         ci.pred(mC, prf)) * 100,
+          lty = c("22","solid"), lend = "butt", lwd = 3, col = 1:2,
+          log = "y", ylim = c(0.01,50), plot = TRUE)
```

The rates of death from other causes is very small at the beginning and increases steeply over the first 5 years of follow-up, while the CVD mortality is pretty stable with a foreseeable increase by age.

Give a proper description of the curves.

10. We can show the impact of albuminuria state on the mortality rates in a 3-panel layout:

```
> par(mfrow=c(1,3))
> for(st in c("Norm","Mic","Mac"))
+    {
+ matshade(prf$age, cbind(ci.pred(mo, transform(prf, lex.Cst = st)),
+                        ci.pred(mC, transform(prf, lex.Cst = st))) * 100,
+          lty = c("22","solid"), lend = "butt", lwd = 3, col = 1:2,
+          log = "y", ylim = c(0.1,50), plot = TRUE)
+ text(60, 50, st, adj = 0)
+    }
```

How are the curves in the three panels related?

Describe the effect of albuminuria status on the two types of mortality.

How can you see this from the model parameters?

1.4.4 State probabilities for different *baseline* values of sex and age.

11. We would like to see how the probability of being in each of the states look as a function of time since entry, and we will in particular be interested in how this depends on `allo`, the allocation to intensified or standard treatment.

    Thus we will need models for 1) all cause mortality rates and 2) transition rates between albuminuria states.

    In this analysis we will collapse the two causes of death to one; this is done by `Relevel`, that also allows re-sequencing of states (see `?Relevel.Lexis` and `?Relevel`):

```
> summary(S4)

Transitions:
     To
From     Mic Norm  Mac D(oth) D(CVD)  Records:  Events: Risk time:  Persons:
  Mic   5986   72   65     30     14      6167      181    1437.39       160
  Norm    35 2418    0     13      6      2472       54     581.83        66
  Mac     22    0 1644     12     18      1696       52     401.70        60
  Sum   6043 2490 1709     55     38     10335      287    2420.91       160

> S5 <- Relevel(S4, list(2, 1, 3, Dead = 4:5))
> summary(S5)

Transitions:
     To
From    Norm  Mic  Mac Dead  Records:  Events: Risk time:  Persons:
  Norm 2418   35    0   19      2472       54     581.83        66
  Mic    72 5986   65   44      6167      181    1437.39       160
  Mac     0   22 1644   30      1696       52     401.70        60
  Sum   2490 6043 1709   93     10335      287    2420.91       160

> boxes(S5, boxpos = TRUE)
> par(mfrow=c(1,2))
> for (al in levels(S5$allo))
```

```
+     {
+     boxes(subset(S5, allo == al),
+           boxpos = list(x = c(15,15,15,85),
+                         y = c(15,50,85,50)),
+           cex = 0.8, show.BE = 'Nz',
+           scale.R = 100)
+     text(85, 90, al, adj = 1)
+     }
```

Describe how mortality depends on albuminuria status and intervention group.

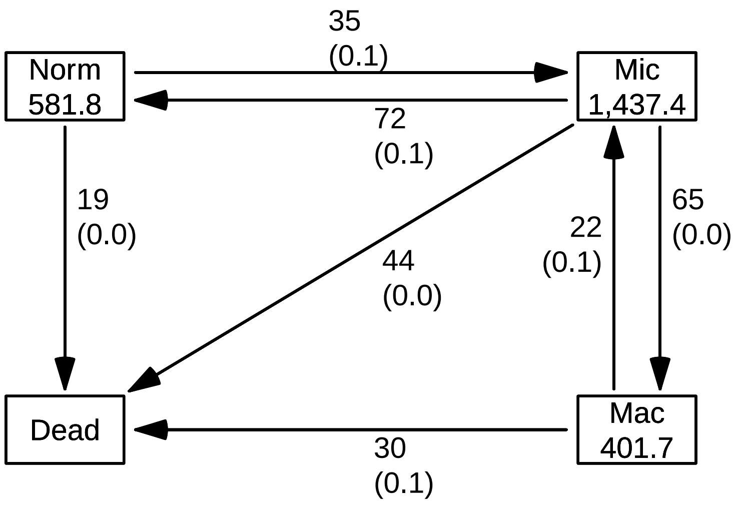

Figure 1.10: *Transitions between states after collapsing the two causes of death.*

`../graph/ms-b5`

12. Now model the overall mortality using a proportional hazards model, but allowing different mortality between the two allocation groups, and the three albuminuria groups:

```
> m0 <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                       Ns(age, knots = seq(60, 75, 5)) +
+                       lex.Cst * allo)

stats::glm Poisson analysis of Lexis object S5 with log link:
Rates for transitions: Norm->Dead, Mic->Dead, Mac->Dead

> round(ci.exp(m0), 3)
```

```
                                 exp(Est.)  2.5%    97.5%
(Intercept)                          0.002 0.000    0.015
Ns(tfi, knots = seq(0, 20, 5))1      6.215 1.196   32.302
Ns(tfi, knots = seq(0, 20, 5))2      4.514 1.053   19.354
Ns(tfi, knots = seq(0, 20, 5))3     45.137 0.994 2048.866
Ns(tfi, knots = seq(0, 20, 5))4      0.542 0.177    1.661
Ns(age, knots = seq(60, 75, 5))1     2.156 1.035    4.489
Ns(age, knots = seq(60, 75, 5))2     4.725 2.536    8.806
Ns(age, knots = seq(60, 75, 5))3     4.095 2.219    7.557
lex.CstMic                           0.883 0.396    1.970
lex.CstMac                           1.468 0.586    3.675
alloConv                             1.605 0.644    4.001
lex.CstMic:alloConv                  1.133 0.377    3.402
lex.CstMac:alloConv                  1.155 0.346    3.851
```

We would however like to see the allocation effect separately for each albuminuria state; this is done by the "/" operator in the model formula:

```
> mi <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                       Ns(age, knots = seq(60, 75, 5)) +
+                       lex.Cst / allo)
```

```
stats::glm Poisson analysis of Lexis object S5 with log link:
Rates for transitions: Norm->Dead, Mic->Dead, Mac->Dead
```

```
> round(ci.exp(mi), 3)
```

```
                                 exp(Est.)  2.5%    97.5%
(Intercept)                          0.002 0.000    0.015
Ns(tfi, knots = seq(0, 20, 5))1      6.215 1.196   32.302
Ns(tfi, knots = seq(0, 20, 5))2      4.514 1.053   19.354
Ns(tfi, knots = seq(0, 20, 5))3     45.137 0.994 2048.866
Ns(tfi, knots = seq(0, 20, 5))4      0.542 0.177    1.661
Ns(age, knots = seq(60, 75, 5))1     2.156 1.035    4.489
Ns(age, knots = seq(60, 75, 5))2     4.725 2.536    8.806
Ns(age, knots = seq(60, 75, 5))3     4.095 2.219    7.557
lex.CstMic                           0.883 0.396    1.970
lex.CstMac                           1.468 0.586    3.675
lex.CstNorm:alloConv                 1.605 0.644    4.001
lex.CstMic:alloConv                  1.819 0.994    3.330
lex.CstMac:alloConv                  1.854 0.858    4.003
```

```
> c(deviance(m0), deviance(mi))
```

```
[1] 969.5975 969.5975
```

The use of the deviance gives a good indication that the models fitted actually *is* the same model, just differently parametrized.

What is the meaning of the parameters?

If you want to *test* for interaction use the formulation `m0`, and see if the two interaction parameters are 0:

```
> ci.exp(m0, subset = ":")

                     exp(Est.)      2.5%    97.5%
lex.CstMic:alloConv   1.133265 0.3774755 3.402314
lex.CstMac:alloConv   1.154598 0.3461396 3.851325

>   Wald(m0, subset = ":")

     Chisq       d.f.          P
0.06357982 2.00000000 0.96871008
```

So there is no indication of interaction, that means that we can safely assume that the allocation effect on mortality is the same for all three groups of albuminuria.

13. For a complete description of transitions we need model for the transitions between albuminuria states; we will use different models for deterioration and improvement:

```
> det <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                        Ns(age, knots = seq(60, 75, 5)) +
+                        lex.Cst / allo,
+                  from = c("Norm","Mic"),
+                    to = c("Mic","Mac"))

stats::glm Poisson analysis of Lexis object S5 with log link:
Rates for transitions: Norm->Mic, Mic->Mac

> imp <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                        Ns(age, knots = seq(60, 75, 5)) +
+                        lex.Cst / allo,
+                    to = c("Norm","Mic"),
+                  from = c("Mic","Mac"))

stats::glm Poisson analysis of Lexis object S5 with log link:
Rates for transitions: Mic->Norm, Mac->Mic

> round(  ci.exp(det, subset="al"), 1)

                      exp(Est.) 2.5% 97.5%
lex.CstNorm:alloConv        0.5  0.2   1.1
lex.CstMic:alloConv         1.9  1.2   3.2

> round(  ci.exp(imp, subset="al"), 1)

                     exp(Est.) 2.5% 97.5%
lex.CstMic:alloConv        0.5  0.3   0.9
lex.CstMac:alloConv        1.3  0.5   3.2

> round(1/ci.exp(imp, subset="al"), 1)[,c(1,3,2)]

                     exp(Est.) 97.5% 2.5%
lex.CstMic:alloConv        1.9   1.2  3.1
lex.CstMac:alloConv        0.8   0.3  1.9
```

What do the parameters in the model represent?

Why that inverted version of the parameters in the `imp` model?

14. We now have statistical models for all transitions, one common model for the three mortality rates, and two models for transitions between albuminuria states.

    We can therefore assess the probability of being in each of the states at a given time after entry to the study, separately for the the two intervention groups. However these depend on the age at entry to the study (because current age (`age`) and time since entry, (`tfi`) are both in the model), so this can be approached in (at least) two different ways:

    (a) Use a population with the same age-distribution as the entire study population

    (b) Evaluate the probabilities for a prespecified range of ages at entry.

    The state probabilities are not trivial to compute, essentially they can only be computed by simulation[1].

    What is needed for this is a data frame of persons indicating their initial status. `simLexis` will then simulate their individual trajectories through states (what transition takes place when) and produce a simulated cohort of persons in the form of a `Lexis` object. The initial data frame should be a Lexis object, but the values of `lex.Xst` and `lex.dur` need not be given, since these will be simulated.

    First construct a cohort with the same covariates as the entire study for each of the allocation groups:

```
> ini <- L2[,c("per", "age", "tfi")]
> ini <- rbind(transform(ini, lex.Cst = "Mic", allo = "Int"),
+              transform(ini, lex.Cst = "Mic", allo = "Conv"))
> str(ini)
```

```
Classes ‘Lexis’ and 'data.frame':        320 obs. of  5 variables:
 $ per    : 'cal.yr' num  1993 1993 1993 1993 1993 ...
 $ age    : 'cal.yr' num  61.1 46.6 49.9 48.5 57.3 ...
 $ tfi    : num  0 0 0 0 0 0 0 0 0 0 ...
 $ lex.Cst: Factor w/ 1 level "Mic": 1 1 1 1 1 1 1 1 1 1 ...
 $ allo   : Factor w/ 2 levels "Int","Conv": 1 1 1 1 1 1 1 1 1 1 ...
 - attr(*, "breaks")=List of 3
  ..$ per: NULL
  ..$ age: NULL
  ..$ tfi: NULL
 - attr(*, "time.scales")= chr  "per" "age" "tfi"
 - attr(*, "time.since")= chr  "" "" ""
```

    This will be the initial values in the cohort we follow through states.

    We also need a specification of what transitions are modeled, since the simulated transitions will be using predictions from these models. This is specified in a list of lists (remember what a list is??)

[1] A detailed description of the use of `simLexis` is available in the vignette in the `Epi` package, also available as http://bendixcarstensen.com/Epi/simLexis.pdf

```
> Tr <- list(Norm = list(Mic = det,
+                        Dead = mi),
+             Mic = list(Mac = det,
+                        Norm = imp,
+                        Dead = mi),
+             Mac = list(Mic = imp,
+                        Dead = mi))
> lapply(Tr, names)

$Norm
[1] "Mic"  "Dead"

$Mic
[1] "Mac"  "Norm" "Dead"

$Mac
[1] "Mic"  "Dead"
```

For example, the object `Tr$Norm$Dead` is a model for the transition rate `Norm` → `Dead`; we see that there are 7 models in the specification of `Tr`, corresponding to each of the 7 transitions in the diagram in figure 1.10.

15. First we simulate transitions from a large cohort that looks like the study population, say 10 copies of each persons in the original data set (see `?simLexis`):

```
> set.seed(1952)
> system.time(
+ Sorg <- simLexis(Tr = Tr,   # models for each transition
+                 init = ini, # cohort of straters
+                    N = 10,  # how many copies of each
+              t.range = 20,  # how long should we simulate before censoring
+                n.int = 200))# how many intervals for evaluating rates

   user  system elapsed
 20.440   6.544  19.635

> summary(Sorg, t = T)

Transitions:
     To
From    Mic Norm  Mac Dead  Records:  Events: Risk time:  Persons:
  Mic   661 1468 1282  856      4267     3606   26941.05      3200
  Norm  706  370    0  392      1468     1098   11808.25      1347
  Mac   361    0  293  628      1282      989    7570.29      1164
  Sum  1728 1838 1575 1876      7017     5693   46319.59      3200

Timescales:
per age tfi
 ""  ""  ""

> subset(Sorg, lex.id %in% 29:32)
```

```
   lex.id      per      age       tfi   lex.dur lex.Cst lex.Xst allo     cens
75     29 1993.373 49.94387  0.000000  3.793669     Mic     Mac  Int 2013.373
76     29 1997.167 53.73754  3.793669 13.478129     Mac     Mic  Int 2013.373
77     29 2010.645 67.21567 17.271798  2.728202     Mic     Mic  Int 2013.373
78     30 1993.373 49.94387  0.000000 19.821493     Mic    Dead  Int 2013.373
79     31 1993.337 48.50376  0.000000  4.423084     Mic    Norm  Int 2013.337
80     31 1997.761 52.92685  4.423084 15.576916    Norm    Norm  Int 2013.337
81     32 1993.337 48.50376  0.000000  8.402331     Mic    Dead  Int 2013.337
```

```
> addmargins(table(table(Sorg$lex.id)))

   1    2    3    4    5    6    7    8  Sum 
 869 1355  588  304   54   23    6    1 3200 
```

Describe in words how the simulated data look, and what each record represents.

We can now just count how many of the original 3200 persons are in each of the states at each time; this is done by the function `nState`:

```
> system.time(
+ Nst <- nState(Sorg,
+                   at = seq(0, 20, 0.1),
+                 from = 0,
+           time.scale = "tfi"))

   user  system elapsed 
  2.640   0.007   2.646 

> str(Nst)

 'table' int [1:201, 1:4] 3200 3130 3063 3019 2964 2905 2860 2817 2779 2736 ...
 - attr(*, "dimnames")=List of 2
  ..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
  ..$ State: chr [1:4] "Mic" "Norm" "Mac" "Dead"

> head(Nst)

     State
when   Mic Norm  Mac Dead
  0   3200    0    0    0
  0.1 3130   47   22    1
  0.2 3063  101   35    1
  0.3 3019  135   44    2
  0.4 2964  171   62    3
  0.5 2905  214   77    4
```

This is not necessarily a relevant summary; we would be interested in seeing how things look in each of the allocation groups, `Int` and `Conv`. There is no guaranteed order of the columns in the `Nst` object, so we explicitly reorder the columns:

```
> Nint <- nState(subset(Sorg, allo == "Int"),
+                   at = seq(0, 20, 0.1),
+                 from = 0,
+           time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")]
> Nconv<- nState(subset(Sorg, allo == "Conv"),
+                   at = seq(0, 20, 0.1),
+                 from = 0,
+           time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")]
> head(Nint)

     State
when  Norm  Mic  Mac Dead
  0      0 1600    0    0
  0.1   29 1564    7    0
  0.2   69 1517   14    0
  0.3   90 1496   14    0
  0.4  112 1470   18    0
  0.5  139 1437   24    0

> head(Nconv)

     State
when  Norm  Mic  Mac Dead
  0      0 1600    0    0
  0.1   18 1566   15    1
  0.2   32 1546   21    1
  0.3   45 1523   30    2
  0.4   59 1494   44    3
  0.5   75 1468   53    4
```

16. If we want the cumulated state probabilities we can derive these by `pState`, that yields a matrix with the cumulative state probabilities. This has class `pState`, an object for which there is plot method:

```
> Pint  <- pState(Nint )
> Pconv <- pState(Nconv)
>  str(Pint)

 'pState' num [1:201, 1:4] 0 0.0181 0.0431 0.0563 0.07 ...
 - attr(*, "dimnames")=List of 2
  ..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
  ..$ State: chr [1:4] "Norm" "Mic" "Mac" "Dead"

> head(Pint)

     State
when      Norm      Mic Mac Dead
  0   0.000000 1.000000   1    1
  0.1 0.018125 0.995625   1    1
  0.2 0.043125 0.991250   1    1
  0.3 0.056250 0.991250   1    1
  0.4 0.070000 0.988750   1    1
  0.5 0.086875 0.985000   1    1
```

Describe the structure of `Pst`.

There is a standard plotting method for a `pState` object, in order

```
> par(mfrow = c(1,2), mar=c(3,3,2,2))
> plot(Pint, col = c("forestgreen", "orange", "red", gray(0.4)),
+           xlim = c(0,20))
> lines(as.numeric(rownames(Pint)), Pint[,"Mac"], lwd = 4)
> text(rownames(Pint)[100],
+      Pint[100,] - diff(c(0,Pint[100,]))/2,
+      colnames(Pint),
+      col = "white")
> plot(Pconv, col = c("forestgreen", "orange", "red", gray(0.4)),
+           xlim = c(20,0))
> lines(as.numeric(rownames(Pconv)), Pconv[,"Mac"], lwd = 4)
> text(rownames(Pconv)[100],
+      Pconv[100,] - diff(c(0,Pconv[100,]))/2,
+      colnames(Pconv),
+      col = "white")
> mtext(c("Int","Conv"), side = 3, at = c(1,3)/4, outer = TRUE, line = -2)
```

Redo the plot with proper labeling of axes, including units where needed.

17. The plot 1.11 is however of limited interest, the probabilities here are really "the probability that a randomly chosen person from the Steno 2 study…". So we are referring to a universe that is not generalizable, the reference is to a particular distribution of ages at entry into the study. The plot is only partially relevant for showing the intervention effect.

    Even if we take the modeling background deeply serious and accept that occurrence rates depend only on current age (`age`), time since entry (`tfi`) and treatment allocation (`allo`), the assumption of age-distribution as in the Steno 2 study is quite absurd; who wants to refer to this? Often this is disguised in terms such as "population averaged".

    Therefore, it would be more relevant to show the results for a homogeneous population of persons aged, say, 50 years a entry. This would just require a different `init` data frame:

```
> ini <- S5[1:10,c("lex.id", "per", "age", "tfi", "lex.Cst", "allo")]
> str(ini)
```

```
Classes ‘Lexis’ and 'data.frame':       10 obs. of  6 variables:
 $ lex.id : num  1 1 1 1 1 1 1 1 1 1
 $ per    : num  1993 1993 1994 1994 1994 ...
 $ age    : num  61.1 61.2 61.3 61.6 61.8 ...
 $ tfi    : num  0 0.118 0.25 0.5 0.75 ...
 $ lex.Cst: Factor w/ 4 levels "Norm","Mic","Mac",..: 2 2 2 2 2 2 2 2 2 2
 $ allo   : Factor w/ 2 levels "Int","Conv": 1 1 1 1 1 1 1 1 1 1
 - attr(*, "time.scales")= chr  "per" "age" "tfi"
 - attr(*, "time.since")= chr  "" "" ""
 - attr(*, "breaks")=List of 3
  ..$ per: NULL
  ..$ age: NULL
  ..$ tfi: num  0 0.25 0.5 0.75 1 1.25 1.5 1.75 2 2.25 ...
```

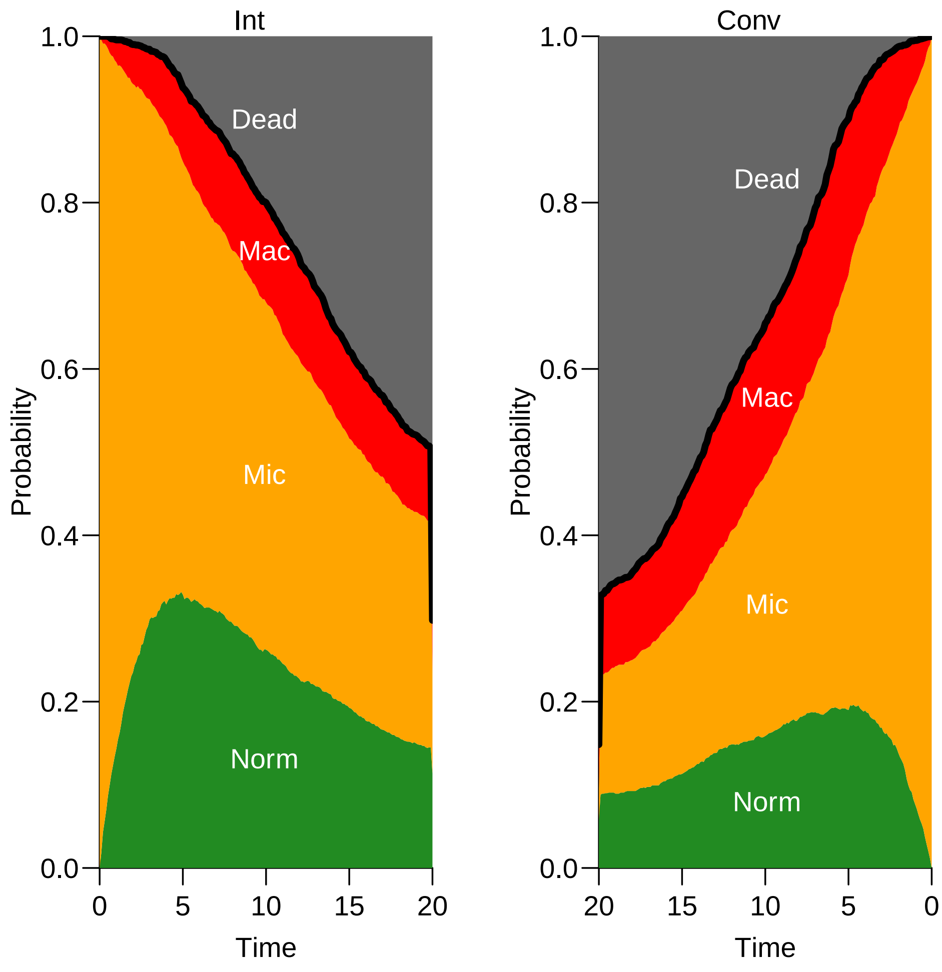

Figure 1.11: *State probabilities for the two intervention groups, for populations of the same structure as the original* total *Steno2 population.*

`../graph/ms-pStates`

```
> ini[,"lex.id"]  <- 1:10
> ini[,"age"]     <-
+ ini[,"ain"]     <- rep(seq(45,65,5), 2)
> ini[,"tfi"]     <- 0
> ini[,"lex.Cst"] <- factor("Mic", levels = levels(S5$lex.Cst))
> ini[,"allo"]    <- rep(c("Int","Conv"), each = 5)
> ini
```

```
   lex.id      per age tfi lex.Cst allo ain
1       1 1993.326  45   0     Mic  Int  45
2       2 1993.444  50   0     Mic  Int  50
3       3 1993.576  55   0     Mic  Int  55
4       4 1993.826  60   0     Mic  Int  60
5       5 1994.076  65   0     Mic  Int  65
6       6 1994.326  45   0     Mic Conv  45
7       7 1994.576  50   0     Mic Conv  50
8       8 1994.826  55   0     Mic Conv  55
```

```
9       9 1995.076  60   0     Mic Conv  60
10     10 1995.326  65   0     Mic Conv  65
```

```
> str(ini)
```

```
Classes ‘Lexis’ and 'data.frame':       10 obs. of  7 variables:
 $ lex.id : int  1 2 3 4 5 6 7 8 9 10
 $ per    : num  1993 1993 1994 1994 1994 ...
 $ age    : num  45 50 55 60 65 45 50 55 60 65
 $ tfi    : num  0 0 0 0 0 0 0 0 0 0
 $ lex.Cst: Factor w/ 4 levels "Norm","Mic","Mac",..: 2 2 2 2 2 2 2 2 2 2
 $ allo   : chr  "Int" "Int" "Int" "Int" ...
 $ ain    : num  45 50 55 60 65 45 50 55 60 65
 - attr(*, "time.scales")= chr  "per" "age" "tfi"
 - attr(*, "time.since")= chr  "" "" ""
 - attr(*, "breaks")=List of 3
  ..$ per: NULL
  ..$ age: NULL
  ..$ tfi: num  0 0.25 0.5 0.75 1 1.25 1.5 1.75 2 2.25 ...
```

Note that it is important that we enter the variable `lex.Cst` as a factor with the same levels as in the `Lexis` object `S5`.

For each of these combinations of age (at entry) and treatment allocation we will simulate 100 persons (note the we are using the same transition rates, the models in `Tr`):

```
> system.time(
+ Sdef <- simLexis(Tr = Tr,
+                init = ini,
+                   N = 100,
+             t.range = 20,
+               n.int = 200))
```

```
   user  system elapsed
  7.298   4.500   6.264
```

```
> str(Sdef)
```

```
Classes ‘Lexis’ and 'data.frame':       2194 obs. of  10 variables:
 $ lex.id : int  1 1 2 2 3 3 3 4 4 4 ...
 $ per    : num  1993 1994 1993 1999 1993 ...
 $ age    : num  45 45.9 45 50.6 45 ...
 $ tfi    : num  0 0.854 0 5.619 0 ...
 $ lex.dur: num  0.854 7.309 5.619 14.381 4.328 ...
 $ lex.Cst: Factor w/ 4 levels "Norm","Mic","Mac",..: 2 1 2 1 2 1 2 2 1 2 ...
 $ lex.Xst: Factor w/ 4 levels "Norm","Mic","Mac",..: 1 4 1 1 1 2 2 1 2 1 ...
 $ allo   : chr  "Int" "Int" "Int" "Int" ...
 $ ain    : num  45 45 45 45 45 45 45 45 45 45 ...
 $ cens   : num  2013 2013 2013 2013 2013 ...
 - attr(*, "breaks")=List of 3
  ..$ per: NULL
  ..$ age: NULL
  ..$ tfi: num  0 0.25 0.5 0.75 1 1.25 1.5 1.75 2 2.25 ...
 - attr(*, "time.scales")= chr  "per" "age" "tfi"
 - attr(*, "time.since")= chr  "" "" ""
```

```
> summary(Sdef)

Transitions:
     To
From   Norm Mic Mac Dead  Records:  Events: Risk time:  Persons:
  Norm  116 191   0  131       438      322    3677.74       412
  Mic   438 196 422  278      1334     1138    8633.39      1000
  Mac     0 143  96  183       422      326    2366.98       378
  Sum   554 530 518  592      2194     1786   14678.11      1000

> head(Sdef)

  lex.id      per      age       tfi    lex.dur lex.Cst lex.Xst allo ain     cens
1      1 1993.326 45.00000 0.0000000  0.8538224     Mic    Norm  Int  45 2013.326
2      1 1994.180 45.85382 0.8538224  7.3093112    Norm    Dead  Int  45 2013.326
3      2 1993.326 45.00000 0.0000000  5.6192193     Mic    Norm  Int  45 2013.326
4      2 1998.946 50.61922 5.6192193 14.3807807    Norm    Norm  Int  45 2013.326
5      3 1993.326 45.00000 0.0000000  4.3284286     Mic    Norm  Int  45 2013.326
6      3 1997.655 49.32843 4.3284286  2.1929862    Norm     Mic  Int  45 2013.326
```

In real applications we would use 1000 replicates of each to minimize the simulation error.

Now we will repeat the graph above, but for the 10 combinations of age at enrollment (`ain`), and allocation:

```
> P45i <- nState(subset(Sdef, ain == 45 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")]
```

This should then be repeated for 4 other ages at enrollment and the two allocations, plus we will only store the state probabilities:

```
> P45c <- pState(nState(subset(Sdef, ain == 45 & allo == "Conv"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P45i <- pState(nState(subset(Sdef, ain == 45 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P50c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P50i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P55c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+          time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
```

```
> P55i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+         time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P60c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+         time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P60i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+         time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P65c <- pState(nState(subset(Sdef, ain == 65 & allo == "Conv"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+         time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
> P65i <- pState(nState(subset(Sdef, ain == 65 & allo == "Int"),
+                  at = seq(0, 20, 0.1),
+                from = 0,
+         time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")])
```

Then we can plot these:

```
> par(mfrow = c(5,2), mar = c(3,3,1,1),
+       oma = c(0,2,1,0), mgp=c(3,1,0)/1.6)
> plot(P45i, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(0,20))
> plot(P45c, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(20,0))
> plot(P50i, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(0,20))
> plot(P50c, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(20,0))
> plot(P55i, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(0,20))
> plot(P55c, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(20,0))
> plot(P60i, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(0,20))
> plot(P60c, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(20,0))
> plot(P65i, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(0,20))
> plot(P65c, col = c("forestgreen", "orange", "red", gray(0.4)),
+            xlim = c(20,0))
> mtext(c("Int","Conv"), side = 3, at = c(1,3)/4, outer = TRUE, line = 0)
> mtext(paste(seq(45,65,10)), side = 2, at = (5:1*2-1)/10,
+       outer = TRUE, line = 0)
```

e see that the curves are quite ragged; this is the simulation errors, it would be nicer if we simulated 1000 copies of each instead of only 100.

18. Detour: The previous is a lot of hard-coding, we would like to be able to easily get a plot with only a subset of the ages. To this end it is more convenient to collect this in an array:

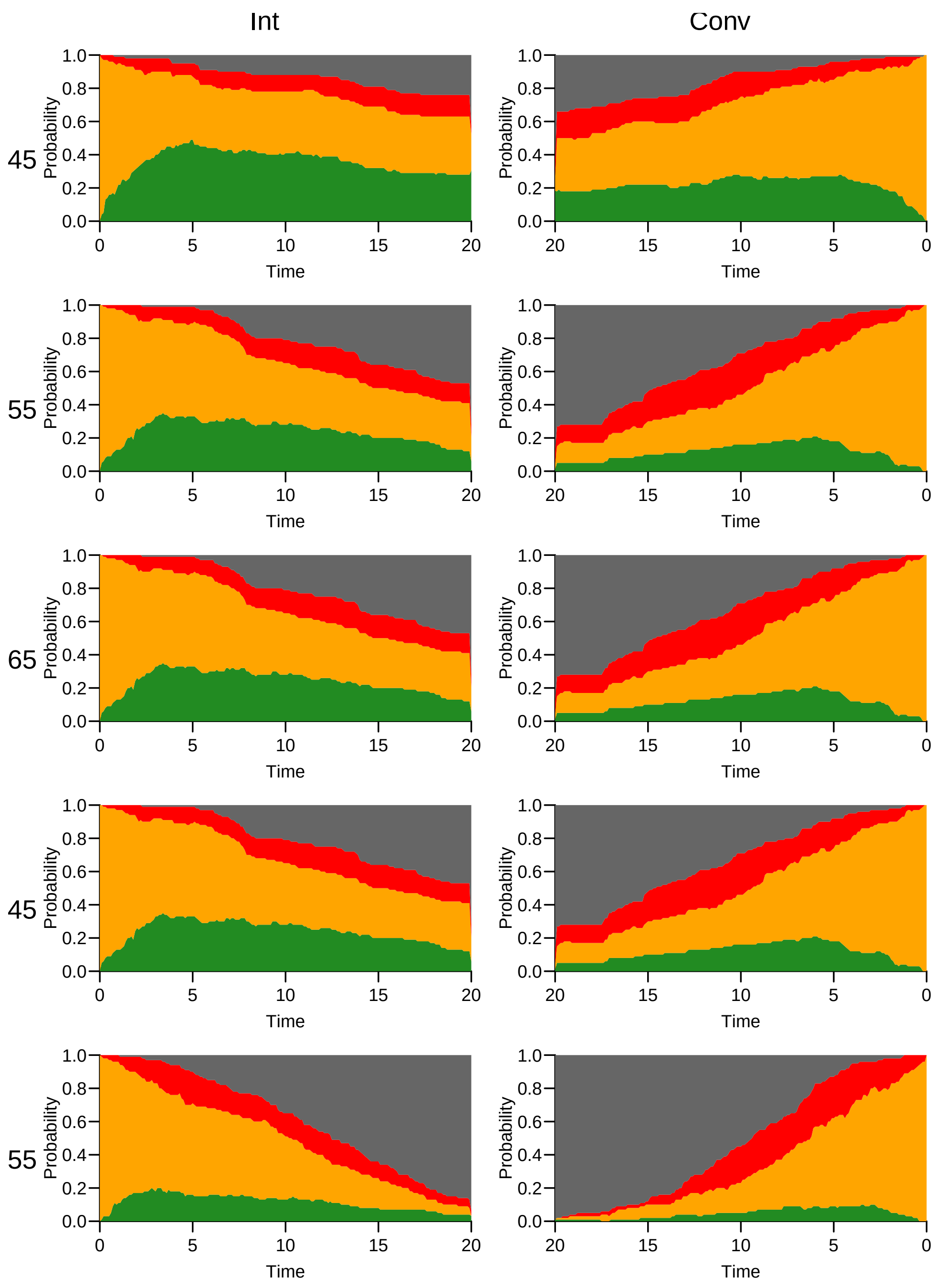

Figure 1.12: *Predicted probabilities of being in each of the states for persons aged 45, 50, 55, 60 and 65 at entry, separately for the two intervention groups. W* ../graph/ms-panel5

```
> (clr <- c("forestgreen", "orange", "red", gray(0.4)))
[1] "forestgreen" "orange"      "red"         "#666666"
> (ain <- seq(45, 65, 10))
[1] 45 55 65
```

```
> (all <- levels(S5$allo))

[1] "Int"  "Conv"

> pdef <- NArray(c(list(ain = ain,
+                       allo = all),
+                  dimnames(P45i)))
> str(pdef)

 logi [1:3, 1:2, 1:201, 1:4] NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 4
  ..$ ain  : chr [1:3] "45" "55" "65"
  ..$ allo : chr [1:2] "Int" "Conv"
  ..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
  ..$ State: chr [1:4] "Norm" "Mic" "Mac" "Dead"
```

We lose the `pState` class of the results, so we resort to the `mat2pol` function that stacks probabilities and plots them, so we simply take the result from `nState` and divide by the number in the initial state (`Mic`) using `sweep`:

```
> for(aa in ain)
+ for(gg in all)
+    pdef[paste(aa), gg, ,] <-
+    nState(subset(Sdef, ain == aa & allo == gg),
+           at = as.numeric(dimnames(pdef)[["when"]]),
+         from = 0,
+   time.scale = "tfi")[,c("Norm","Mic","Mac","Dead")]
> pdef <- sweep(pdef, 1:2, pdef[,,1,"Mic"], "/")
```

```
> par(mfrow = c(length(ain),2),
+       mar = c(3,3,1,1),
+       oma = c(0,2,1,0),
+       mgp = c(3,1,0) / 1.6)
> for(aa in ain)
+    {
+ mat2pol(pdef[paste(aa),"Int" ,,], col = clr, xlim = c(0,20))
+ mat2pol(pdef[paste(aa),"Conv",,], col = clr, xlim = c(20,0))
+    }
> mtext(c("Int","Conv"), side = 3, at = c(1,3)/4, outer = TRUE, line = 0)
> mtext(ain, side = 2, at = (length(ain):1 * 2 - 1) / (length(ain) * 2),
+       outer = TRUE, line = 0)
```

1.5 Time in normo-albuminuria

We have observation time till almost 22 years, but we have predictions of state probabilities in `pdef`:

```
> str(pdef)
```

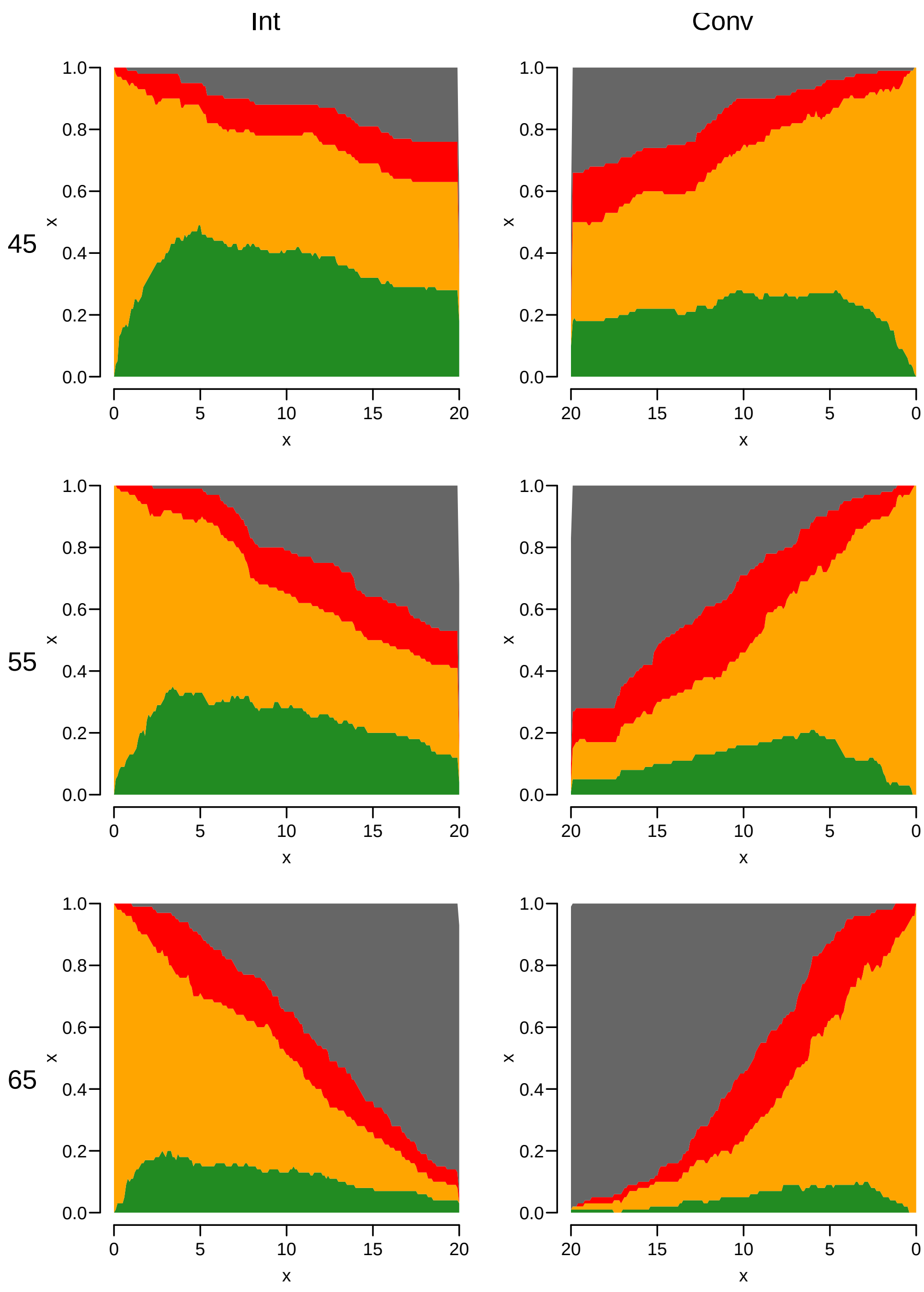

Figure 1.13: *Predicted probabilities of being in each of the states for persons aged 45, 55 and 65 at entry, separately for the two intervention groups.*

`../graph/ms-panel3`

```
num [1:3, 1:2, 1:201, 1:4] 0 0 0 0 0 0 0.04 0.05 0.01 0.01 ...
- attr(*, "dimnames")=List of 4
  ..$ ain  : chr [1:3] "45" "55" "65"
  ..$ allo : chr [1:2] "Int" "Conv"
  ..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
  ..$ State: chr [1:4] "Norm" "Mic" "Mac" "Dead"
```

```
> ftable(pdef[,,1:10,], row.vars = c(1,3))
```

```
         allo   Int                    Conv
         State Norm  Mic  Mac Dead Norm  Mic  Mac Dead
ain when
45  0           0.00 1.00 0.00 0.00 0.00 1.00 0.00 0.00
    0.1         0.04 0.94 0.02 0.00 0.01 0.99 0.00 0.00
    0.2         0.05 0.92 0.03 0.00 0.03 0.96 0.00 0.01
    0.3         0.13 0.84 0.03 0.00 0.04 0.95 0.00 0.01
    0.4         0.14 0.83 0.03 0.00 0.04 0.94 0.01 0.01
    0.5         0.16 0.80 0.04 0.00 0.06 0.92 0.01 0.01
    0.6         0.16 0.80 0.04 0.00 0.07 0.90 0.02 0.01
    0.7         0.17 0.79 0.04 0.00 0.08 0.89 0.02 0.01
    0.8         0.16 0.79 0.04 0.01 0.09 0.86 0.04 0.01
    0.9         0.19 0.75 0.05 0.01 0.09 0.85 0.05 0.01
55  0           0.00 1.00 0.00 0.00 0.00 1.00 0.00 0.00
    0.1         0.05 0.95 0.00 0.00 0.00 1.00 0.00 0.00
    0.2         0.06 0.93 0.01 0.00 0.00 0.99 0.01 0.00
    0.3         0.08 0.91 0.01 0.00 0.02 0.96 0.02 0.00
    0.4         0.09 0.89 0.02 0.00 0.03 0.94 0.03 0.00
    0.5         0.09 0.89 0.02 0.00 0.03 0.94 0.03 0.00
    0.6         0.09 0.89 0.02 0.00 0.03 0.94 0.03 0.00
    0.7         0.11 0.87 0.02 0.00 0.03 0.94 0.03 0.00
    0.8         0.12 0.86 0.02 0.00 0.03 0.93 0.04 0.00
    0.9         0.13 0.84 0.03 0.00 0.03 0.94 0.03 0.00
65  0           0.00 1.00 0.00 0.00 0.00 1.00 0.00 0.00
    0.1         0.01 0.98 0.01 0.00 0.00 0.96 0.04 0.00
    0.2         0.03 0.95 0.02 0.00 0.00 0.96 0.04 0.00
    0.3         0.03 0.95 0.02 0.00 0.00 0.95 0.05 0.00
    0.4         0.03 0.95 0.02 0.00 0.00 0.94 0.06 0.00
    0.5         0.03 0.94 0.03 0.00 0.02 0.91 0.07 0.00
    0.6         0.05 0.92 0.03 0.00 0.02 0.90 0.08 0.00
    0.7         0.09 0.87 0.04 0.00 0.02 0.89 0.09 0.00
    0.8         0.11 0.85 0.04 0.00 0.03 0.88 0.09 0.00
    0.9         0.10 0.86 0.04 0.00 0.03 0.87 0.10 0.00
```

19. We may want to compare groups by the expected time spent in the normoabuminuric state. The expected time in a state is simply the time-integral of the probabilities, so we can easily compute it from `pdef`; each probability represents an interval of length 0.1, so we just take the midpoint of the probabilities at the ends of each interval:

```
> mid <- function(x) x[-1] - diff(x) / 2
> pmid <- apply(pdef, c(1,2,4), mid)
> str(pmid)
```

```
 num [1:200, 1:3, 1:2, 1:4] 0.02 0.045 0.09 0.135 0.15 0.16 0.165 0.165 0.175 0.205 ..
 - attr(*, "dimnames")=List of 4
  ..$      : chr [1:200] "0.1" "0.2" "0.3" "0.4" ...
  ..$ ain  : chr [1:3] "45" "55" "65"
  ..$ allo : chr [1:2] "Int" "Conv"
  ..$ State: chr [1:4] "Norm" "Mic" "Mac" "Dead"
```

```
> pyr <- apply(pmid, 2:4, sum) * 0.1
> str(pyr)
```

```
 num [1:3, 1:2, 1:4] 7.1 4.78 2.31 4.37 2.31 ...
 - attr(*, "dimnames")=List of 3
  ..$ ain  : chr [1:3] "45" "55" "65"
  ..$ allo : chr [1:2] "Int" "Conv"
  ..$ State: chr [1:4] "Norm" "Mic" "Mac" "Dead"
```

```
> round(ftable(pyr[,,-4], col.vars = 3:2), 1)
```

```
    State Norm      Mic       Mac
    allo   Int Conv  Int Conv  Int Conv
ain
45         7.1  4.4  8.4 10.2  2.0  2.5
55         4.8  2.3  8.8  8.1  2.3  3.1
65         2.3  0.9  7.7  6.1  2.4  2.7
```

These numbers are the expected time (in years) spent in each state during the first 20 years after enrollment; we see that the intervention group spend far more time in `Norm` than do the conventional group.

20. The study intervention lasted some 7 years, after which time all persons were shifted to intensive care. So we might ask the question of how long time persons spend in the normoalbuminuric state in the period from 10 to 20 years, *given* that they are still alive 10 years after enrollment.

    This is particularly simple given that we have the simulated data; we can just restrict the simulated data to the follow-up available after 10 years after enrollment.

    Do this.

1.5.1 State probabilities only using time since entry; comparison to Aalen-Johansen approach from `survival`

1.5.2 `addCov` using `st2clin`

1.5.3 Mortality by `chol` and `bp`

1.5.4 Limitations in using clinical measurements as time-dependent variables without a model for the clinical variables

1.5.5 Transitions between microvascular complications states, one model, use `stack`

…comparison of smooth modeling to Cox-models, using the `mstate` machinery.

References

[1] Bendix Carstensen. *Epidemiology with R*. Number ISBN: 978-0-19-884133-3. Oxford University Press, 2020.