


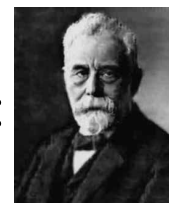


Occurrence rates, cumulative risks,
competing risks, state probabilities
with multiple states and time scales in
in  Register  Research
with  and Epi ::



Computer practicals

SDCstats

October 2021

<http://bendixcarstensen.com/AdvCoh/courses/SDC-2021>

Version 2

Compiled Monday 4th October, 2021, 09:47

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	1
0.1	Program	1
1	Survival and rates: lung	2
1.1	Data and simple survival	2
1.2	Rates and rate-ratios: Simple Cox model	4
1.3	Simple Poisson model	6
1.4	Representation of follow-up: <code>Lexis</code> object	7
1.5	Baseline hazard: splitting time	9
2	Competing risks: <code>DMlate</code>	16
2.1	Data	16
2.2	State probabilities	19
2.3	What not to do	20
2.4	Modeling cause specific rates	21
2.5	Integrals with R	24
2.6	Cumulative risks from parametric models	26
2.7	Expected life time: using simulated objects	29
3	Multistate models: <code>steno2</code>	31
3.1	<code>Lexis</code> object for <code>steno2</code>	31
3.2	Mortality rates: 3 initial states, 2 outcomes, multiple time scales	36
3.3	State probabilities for different <i>baseline</i> values of sex and age.	43
3.4	Simulation of state probabilities	46
3.5	Time spent in albuminuria states	57
3.6	State probabilities using the Aalen-Johansen approach from <code>survival</code>	60
3.7	Clinical variables	65
3.8	A single model for transitions between microvascular complications states: <code>stack</code>	69
	References	73

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 **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>

0.1 Program

Program

Tuesday 5 October

09:00–09:30	L: Introduction to multistate models
09:30–11:30	P: Survival analysis
11:30–12:15	Lunch
12:15–12:45	L: Introduction to competing risks
12:45–16:00	P: Cause-specific rates and competing risks

Wednesday 6 October

09:00–09:30	L: Multistate models in practice
09:30–11:30	P: Multistate models
11:30–12:15	Lunch
12:15–14:15	P: State probabilities

Within each of the the chunks of topics (see the table of contents) there will be a short introductory lecture, introducing the practical.

Chapter 1

Survival and rates: lung

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 Data and simple survival

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,
+                   levels = 1:2,
+                   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?

- 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, 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?

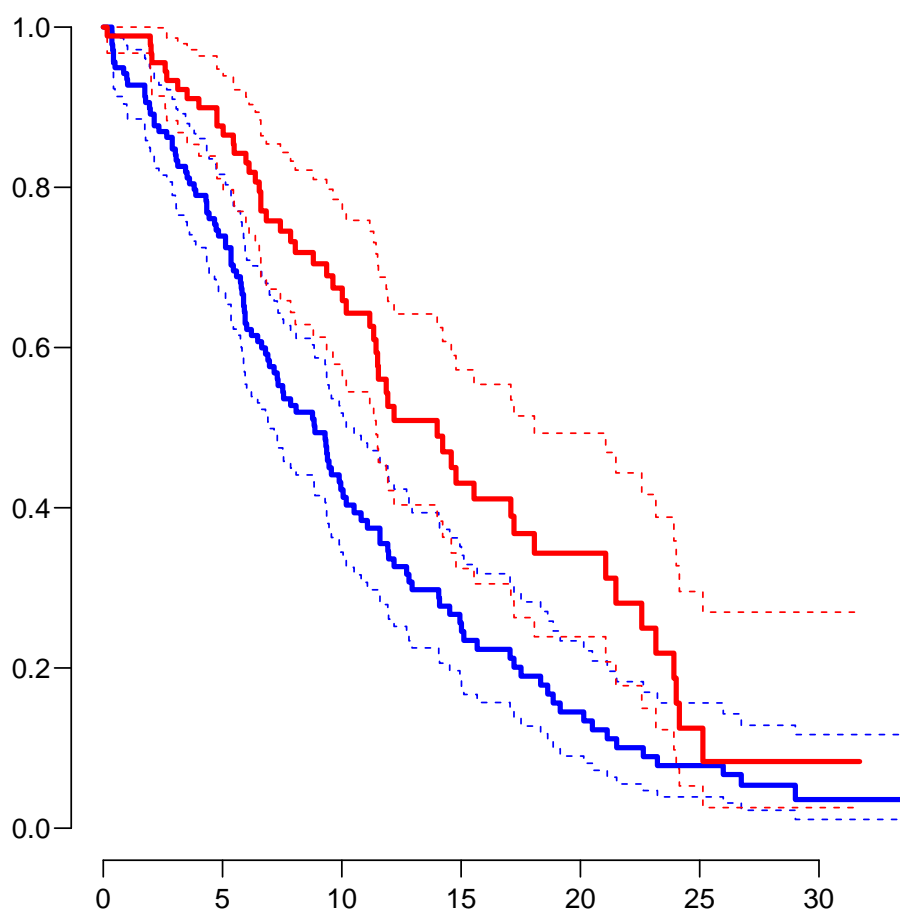


Figure 1.1: *Kaplan-Meier estimators of survival for men (blue) and women (red).* `W`
`../graph/surv-kms`

1.2 Rates and rate-ratios: Simple Cox model

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 1.7% higher mortality rate per year of age at diagnosis (mortality RR or hazard ratio of 1.017).

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)

```

1.3 Simple Poisson model

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)` (taken from the `ci.exp`) 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)
> ## or:
> 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) as a variable in the prediction frame. This is not the case for `poisreg`, where you get the predicted rates per unit in which as you entered the person years when specifying the model.

1.4 Representation of follow-up: *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
```

```
> L1 <- 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 will 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 = L1)
```

but even simpler, by using the *Lexis* features:

```
> ?coxph.Lexis

> cL <- coxph.Lexis(L1, tfl ~ sex + age)

survival::coxph analysis of Lexis object L1:
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(c1)

      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 = L1)

```

or even simpler, by using the Lexis features:

```

> pL <- glm.Lexis(L1, ~ sex + age)

stats::glm Poisson analysis of Lexis object L1 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.

1.5 Baseline hazard: 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 = S1)
> 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 proportional hazards assumption is the same for the Cox model and the Poisson models: The M/W hazard ratio is the same at any time after diagnosis. What differs is a assumed shape of the hazard (not hazard ratio). The Cox model allows the baseline rate to change arbitrarily at every event time time not using the quantitative nature of time, the `ps` Poisson model has a baseline that varies smoothly by time and the `pc` Poisson model has a baseline that is constant over time. The latter is clearly not tenable, whereas the smooth Poisson model and the Cox model give the same regression estimates.

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

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

We can over-plot 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)

```

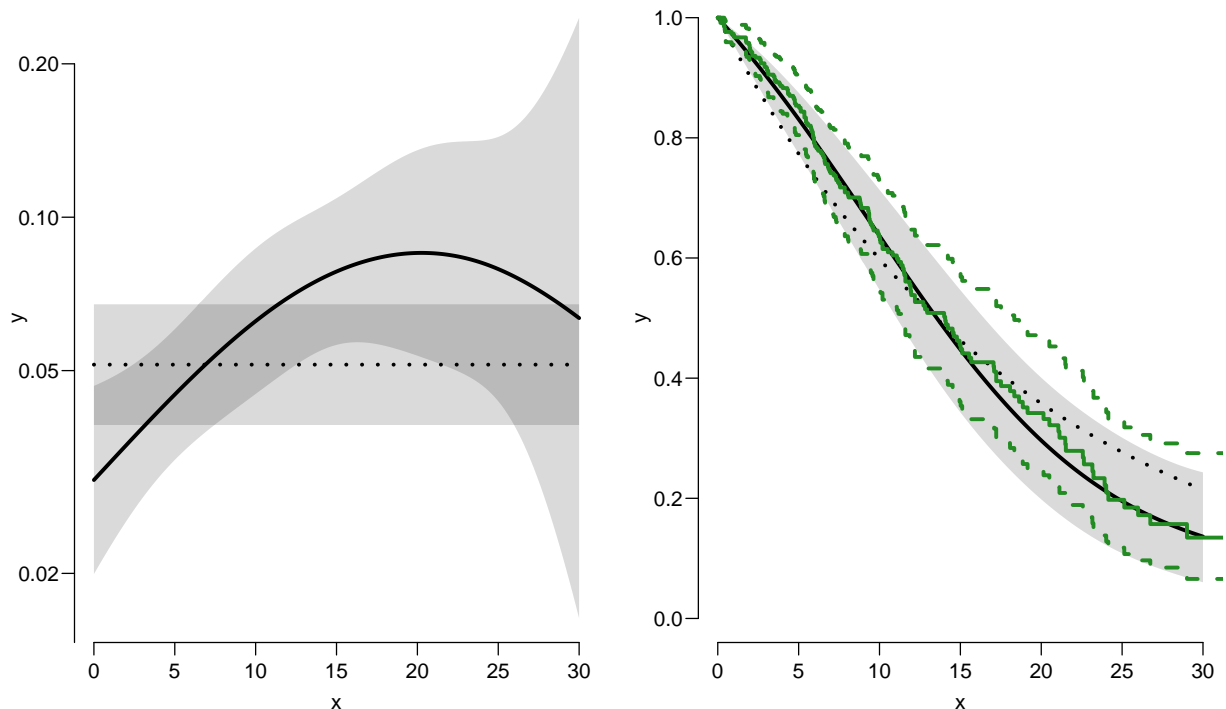


Figure 1.2: Hazards (left) and survival (right) for 60 year old women. The left hand plot is unavailable from the Cox model.

../graph/surv-ratesurv

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

```

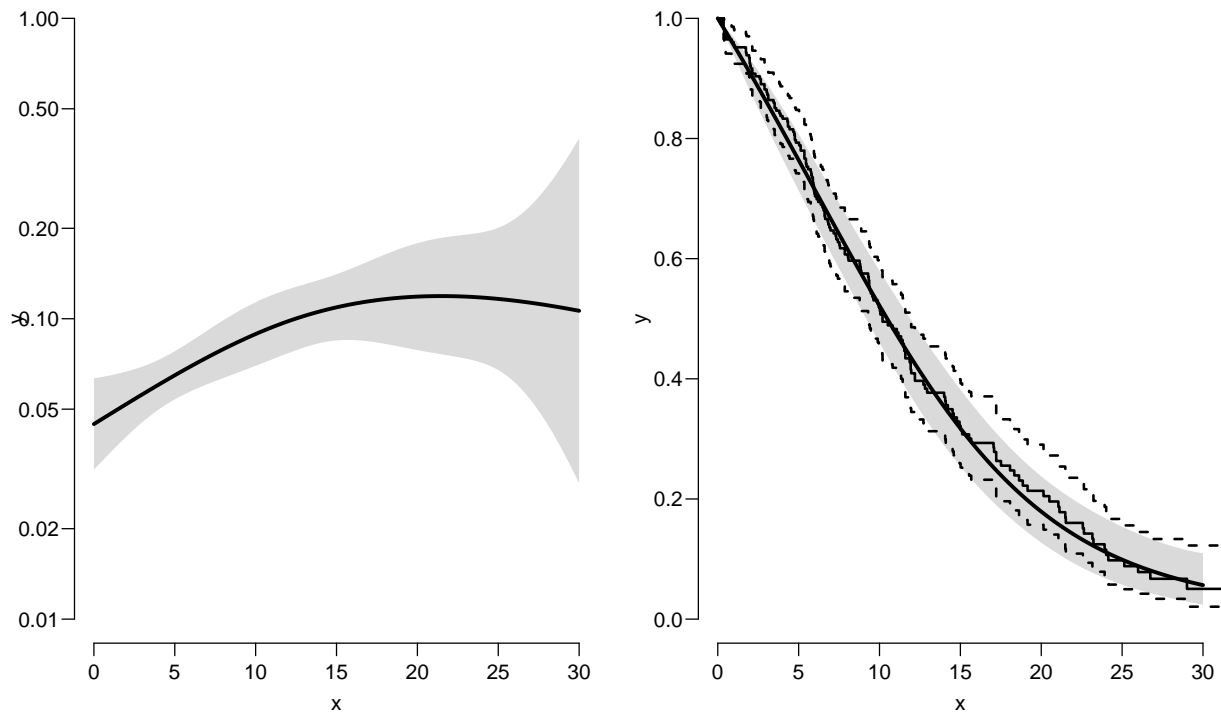


Figure 1.3: *Baseline hazard (left), and corresponding survival function from parametric model and Kaplan-Meier estimator.*

../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 = S1)

```



```

+ matshade(prf$tf1, ci.pred(pk, prf),
+         plot = TRUE, log = "y", lwd = 3, ylim = c(0.01,1))
+ rug(kn, lwd=3)
+
+ matshade(prf$tf1, ci.surv(pk, prf, intl = 0.2) ,
+         plot = TRUE, lwd = 3, ylim = 0:1)
+ lines(km, lwd = 2)
+ }

> zz(12)

> zz(2)

```

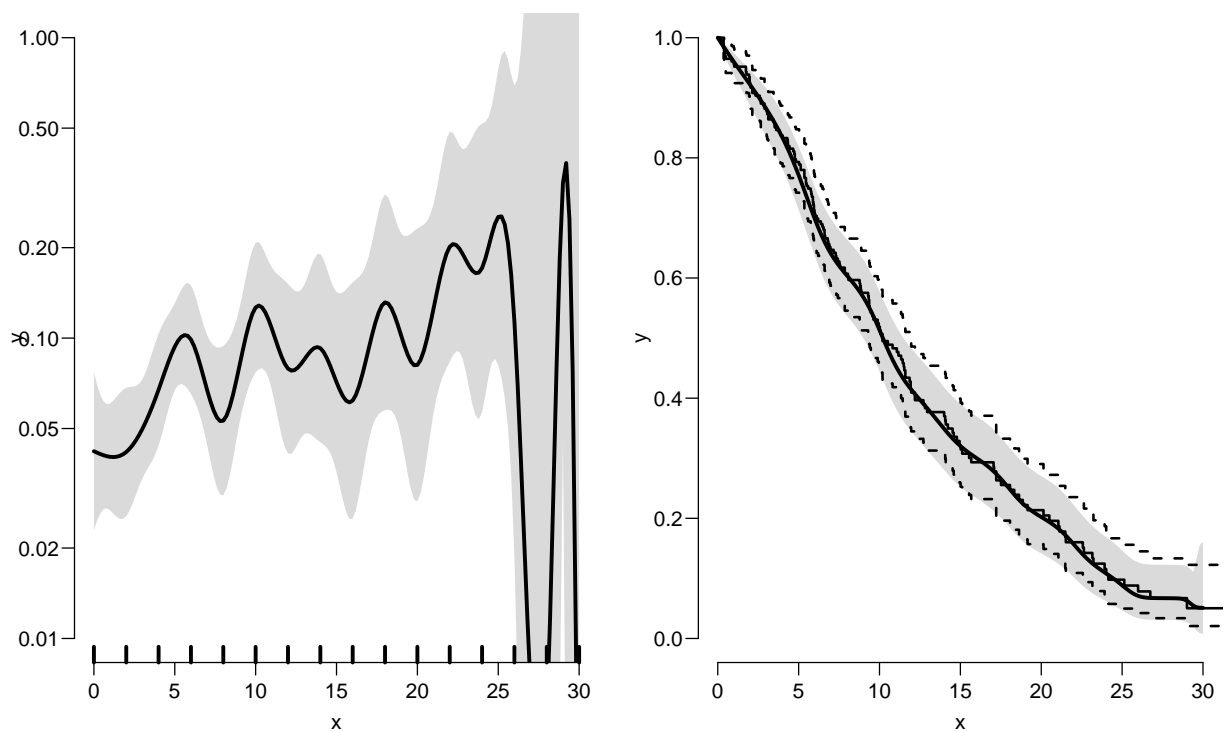


Figure 1.4: *Hazard (left) and survival (right) comparing a parametric model with knots every 2 months and the Kaplan-Meier estimator.*

../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.

But fortunately the baseline hazard underlying the Kaplan-Meier estimator is rarely shown.

Chapter 2

Competing risks: DMlate

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)
> clear()
```

2.1 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	doad	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	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$doad,
+                 timescale = "per",
+                 new.state = "OAD")
> summary(Cdm)
```

Transitions:

	To	From	DM	OAD	Dead	Records:	Events:	Risk time:	Persons:
	DM	DM	685	634	226	1545	860	5414.29	1545
	OAD	OAD	0	836	252	1088	252	5328.05	1088
	Sum	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)
```

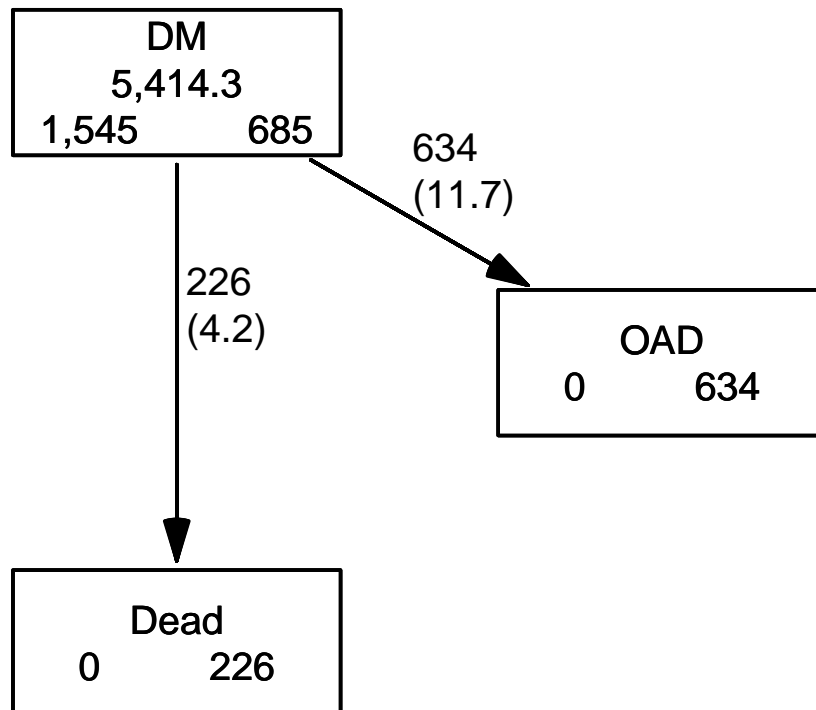


Figure 2.1: *Competing risks set-up for events OAD and Dead.*

../graph/cmpr-boxCR

As shown in figure 2.1 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 oral antidiabetic treatment, disregarding subsequent death”. The other event considered is Dead which here means “dead without initiating oral antidiabetic treatment”.

2.2 State probabilities

We can compute the (correct) counterpart of the survival function for this competing risks setup. The survival function we saw in the previous exercise gives the probability of being alive, and the complement is the probability of being dead.

3. `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 so it is required by the function `survfit`.

Also note that the initial state (DM) must be the first level of the factor `lex.Xst`:

```
> levels(Adm$lex.Xst)
[1] "DM" "OAD" "Dead"

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

Compare `m3$transitions` to `summary(Adm)`.

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()
```

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

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 *overestimate* 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"),
```

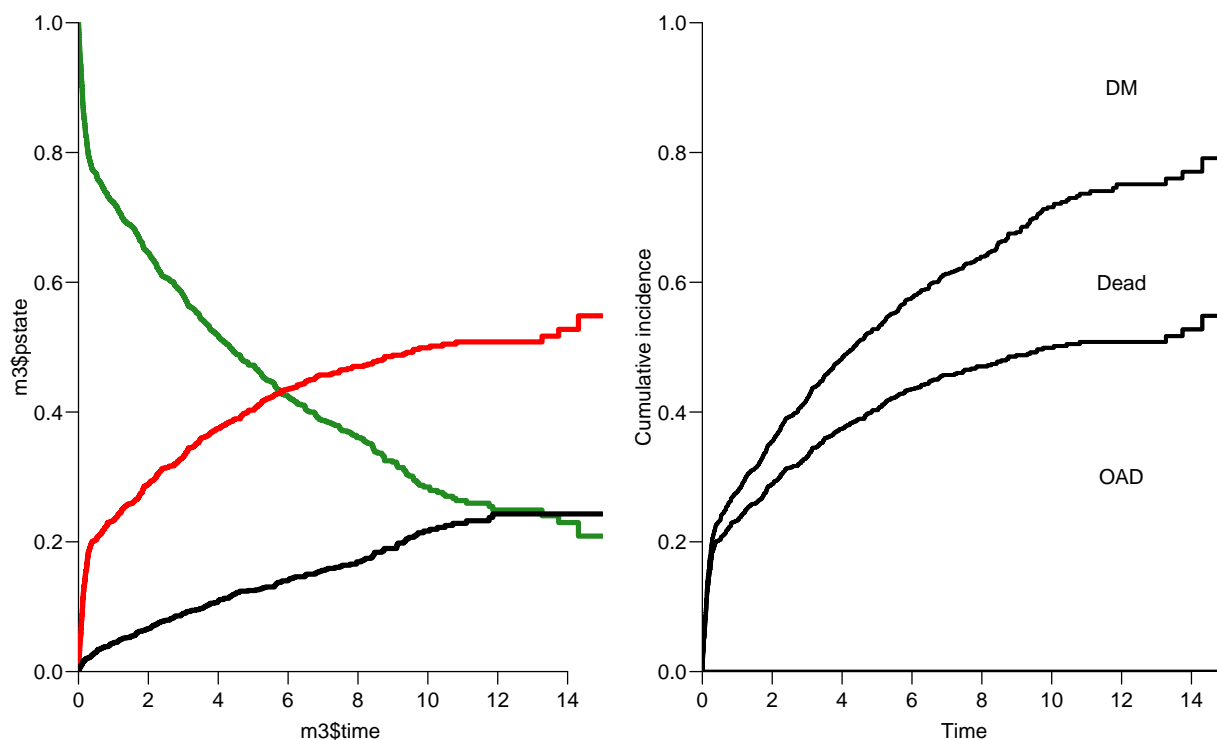


Figure 2.2: *Separate state probabilities (left) and stacked state probabilities (right). In the left panel, Alive is green, OAD is red and Dead is black.*

../graph/cmpr-surv2

```
+ 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.

2.4 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

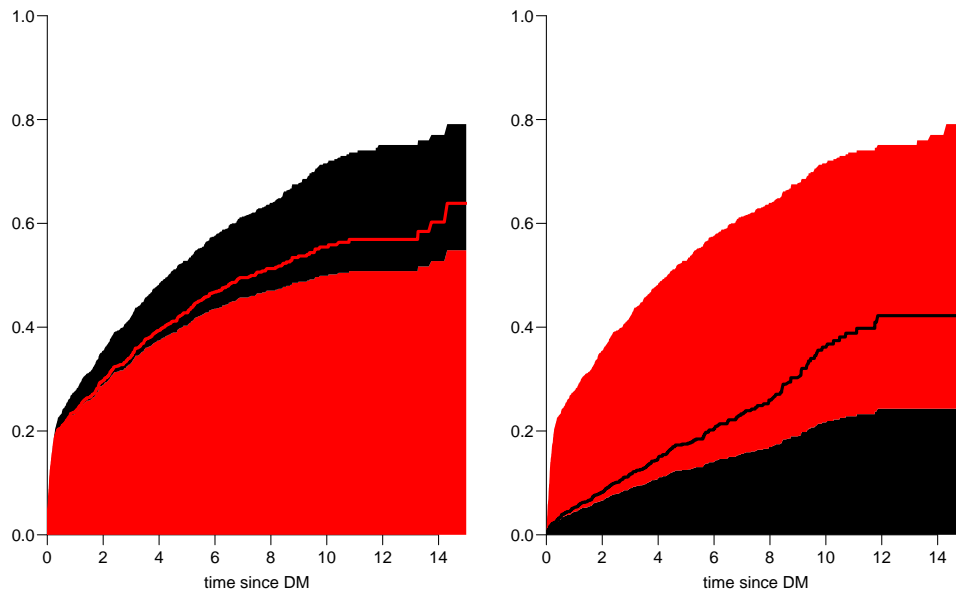


Figure 2.3: Stacked state probabilities Alive is white, OAD is red and Dead is black. The red line in the left panel is the wrong (but often computed) “cumulative risk” of OAD, and the black line in the right panel is the wrong (but often computed) “cumulative risk” of Death. The black and the red areas 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`

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.

- Now model the two sets of rates by parametric models; this must be based on a time-split data set:

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

Transitions:
  To
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
```

- We will use natural splines for the effect of diabetes duration in a model using `glm`. The `Ns` 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)
> OAD.glm <- 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

> Dead.glm <- 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))
> l.glm <- ci.pred( OAD.glm, nd)
> m.glm <- ci.pred(Dead.glm, 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(l.glm, m.glm) * 100,
+         plot = TRUE,
+         log = "y", ylim = c(2, 20),
+         col = rep(c("red", "black"), 2), lwd = 3)

```

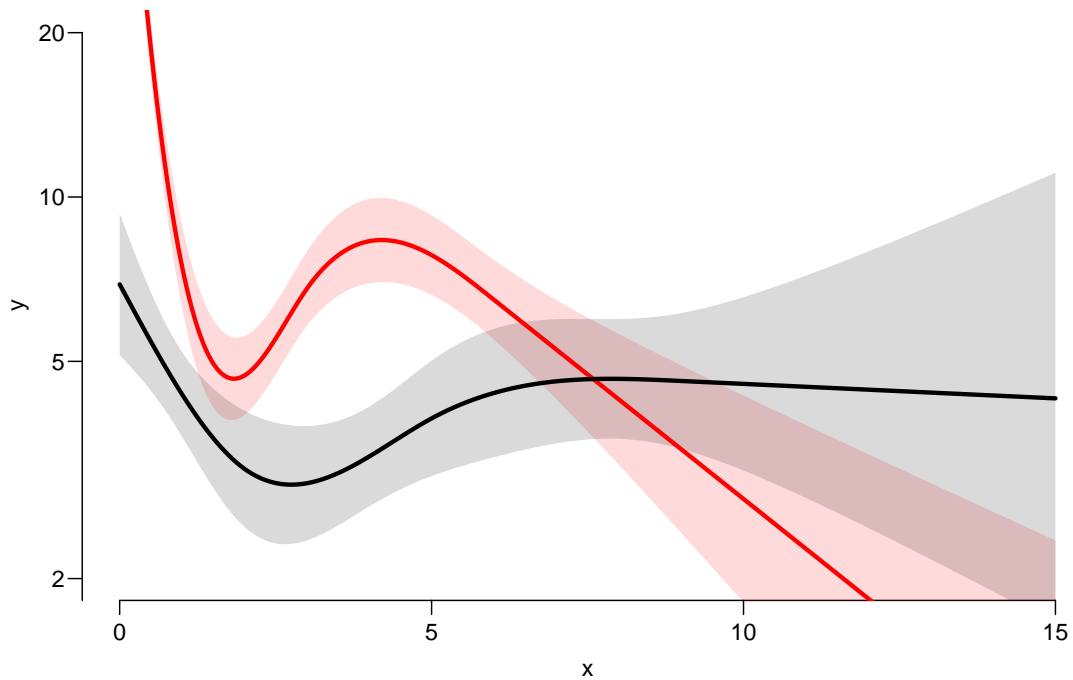


Figure 2.4: *Mortality rates (black) and OAD-rates (red), from a `glm` model with natural splines.*

../graph/cmpr-OAD-mort

2.5 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 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. So here is a digression showing how to calculate integrals as areas under a curve.

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

```
> t <- seq(0, 15, 0.01)
> nd <- data.frame(tfd = t)
> mu <- ci.pred(Dead.glm, nd)[,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

> plot(t, mu, type="l", lwd = 3,
+       xlim = c(0, 7), xaxs = "i",
+       ylim = c(0, max(mu)), yaxs = "i")
> polygon(t[c(1:501,501:1)], c(mu[1:501], rep(0, 501)),
+         col = "gray", border = "transparent")

```

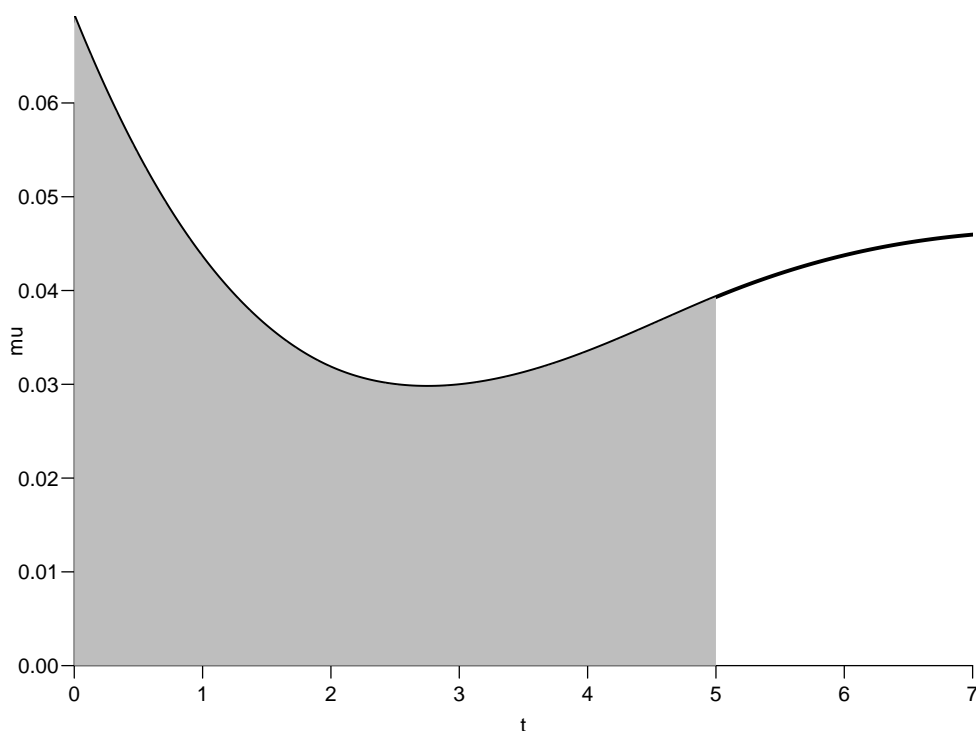


Figure 2.5: *Mortality function and integral from 0 to 5 years.*

../graph/cmpr-int-ill

This is a representation of the points $(t, \mu(t))$; if we want the integral of μ over the interval $[0, 5]$, say, $M(5) = \int_0^5 \mu(s) ds$, we are just asking for the area under the curve. Each t represents an endpoint of an interval, but 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
> (x <- c(1:5, 7, 10))
[1] 1 2 3 4 5 7 10
> mid(x)
[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 the first 501 interval endpoints:

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

So now we have computed $\int_0^5 \mu(s) ds$. This is called the cumulative *rate* over the interval $[0, 5]$ years.

It is important to get the units right. In the modeling we entered the risk time (“person-years”) in units of 1 year, so the unit of predicted mortality function, `mu`, is events per 1 person-year. Therefore, the units of `t` must be year too; otherwise we will introduce a scaling.

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

```
> Mu <- c(0, cumsum(diff(t) * mid(mu)))
> head(cbind(t, Mu))
      t      Mu
1 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.

2.6 Cumulative risks from parametric models

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

$$R_{\text{OAD}}(t) = \int_0^t \lambda(u) S(u) du = \int_0^t \lambda(u) \exp\left(-\int_0^u \lambda(s) + \mu(s) ds\right) du$$

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 practical 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 $(\lambda(t), \mu(t))$, evaluated at 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 at these times
- (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 = OAD.glm,
+                           Dead = Dead.glm),
+               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(as.numeric(dimnames(cR$Crisk)[[1]]),
+          cbind(cR$Crisk[,1,],
+               cR$Crisk[,2,],
+               cR$Crisk[,3,]), plot = TRUE,
+          lwd = 2, col = c("limegreen", "red", "black"))
```

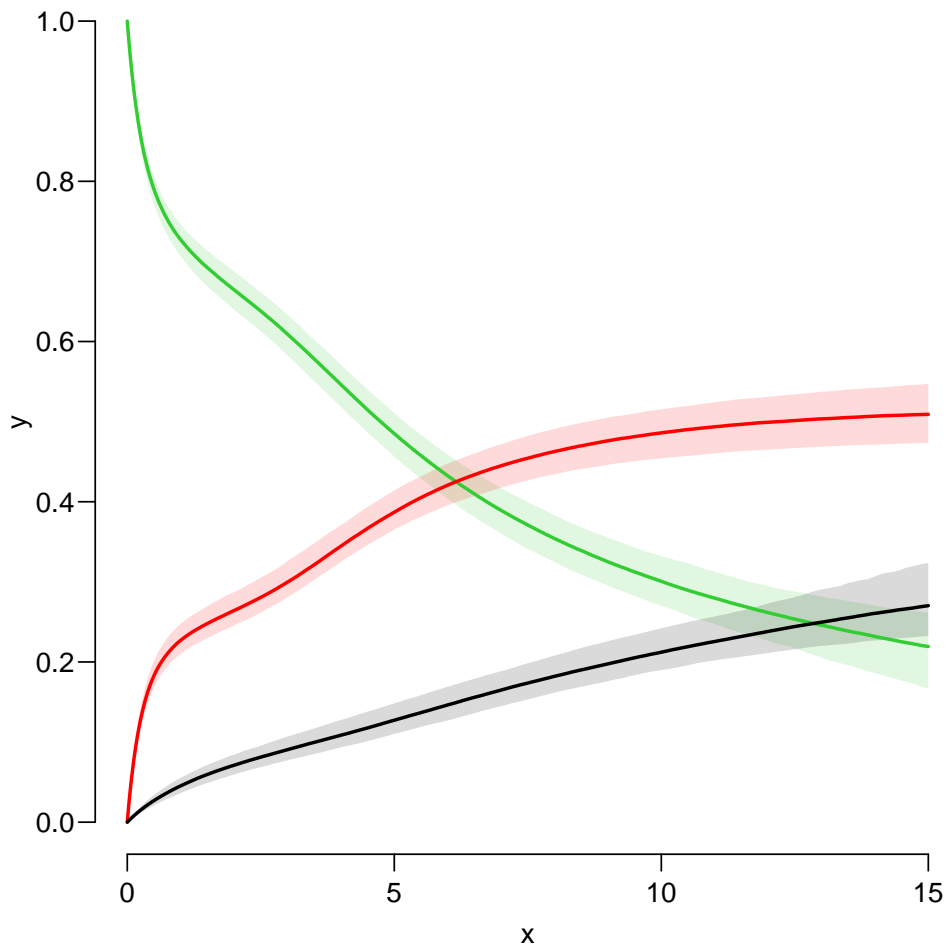


Figure 2.6: Cumulative risks of being in each of the states DM (green), OAD (red) and Dead
 ../graph/cmpr-crisk

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(as.numeric(dimnames(cR$Srisk)[[1]]),
+         cbind(cR$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.

2.7 Expected life time: using simulated objects

12. It is not only the cumulative risks of being in different states that may 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 (remember that the first dimension of `Stime` is in units of 1/100 year):

```
> 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[1:3*500+1,"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[1:3*500+1,"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:

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

Amend the plot with proper axis labels.

Chapter 3

Multistate models: steno2

Paraphernalia

First we load the relevant packages and set some options:

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

For later convenience we devise a function that prints a data frame with all its numerical values rounded—this is particularly useful for `Lexis` objects with time scales calendar time and say, age.

```
> nround <-
+ function(df, dec = 2)
+ {
+   wh.num <- sapply(df, is.numeric)
+   df[,wh.num] <- round(df[,wh.num], dec)
+   print(df)
+ }
```

3.1 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 at entry—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?

Explain the coding of `exit.status`.

How many person-years?

What are the time scales?

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 each date of albuminuria measurement allowing the patients to change between states Normoalbuminuria, Microalbuminuria and Macroalbuminuria at each of these dates, possibly several times per person. To this end we use the function `rcutLexis` (recurrent cuts), 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), and in order to comply with the requirements of `rcutLexis` rename the id variable `id` to `lex.id`, the date variable `doTr` to `cut` and the state variable `state` to `new.state`:

```
> 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 ...
 $ 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 first timescale by default, and the first of the timescales in L2 is `per`):

```
> 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
```

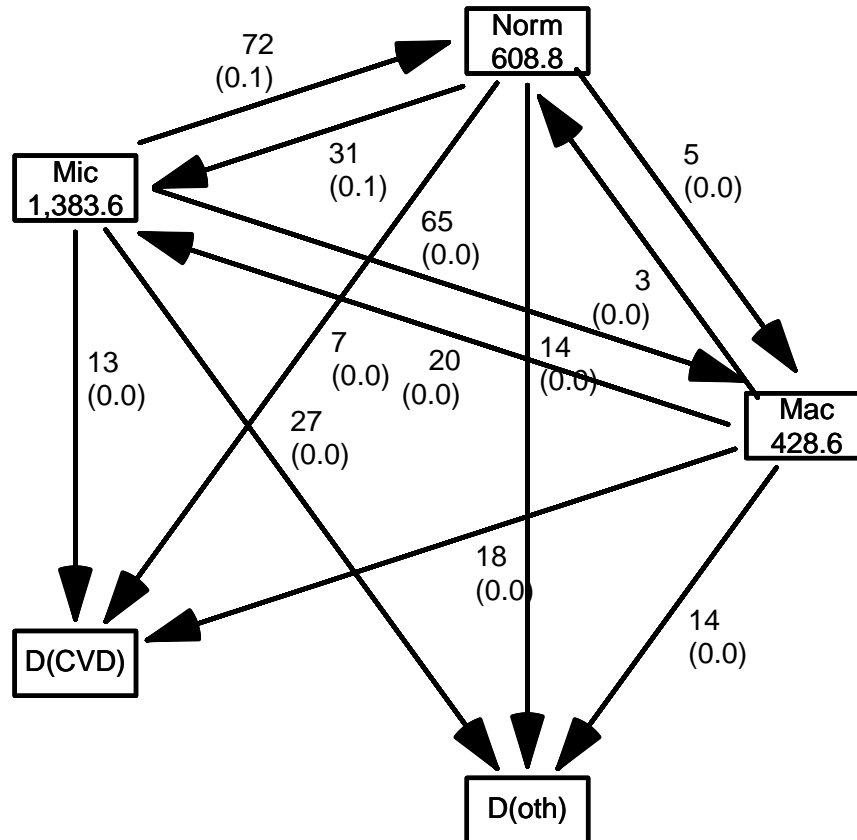


Figure 3.1: *The default lay-out of the 5 boxes, including the jumps directly between Norm and Mac.*

../graph/ms-boxL3

```
> 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, so transitions from Norm to Mac (and vice versa) should go through Mic

5. In order to remedy this anomaly we find all transitions Norm → Mac and provide a

transition Norm \rightarrow Mic in between. And of course similarly for transitions Mac \rightarrow 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 each of these “jumps” is to provide an extra transition to Mic at a time during the stay in either Norm or Mac, i.e. somewhere 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 generated when cutting the follow-up?

- Now make extra cuts (transitions to Mic) 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(50,90,10,75,25)),
+       show.BE = "nz",
+       scale.R = 100,
+       cex = 0.9)

```

Explain the arguments used to `boxes`.

Explain the numbers in the graph.

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.

3.2 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 each interval):

```

> S4 <- splitMulti(L4, tfi = seq(0, 25, 1/2))
> 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)

```

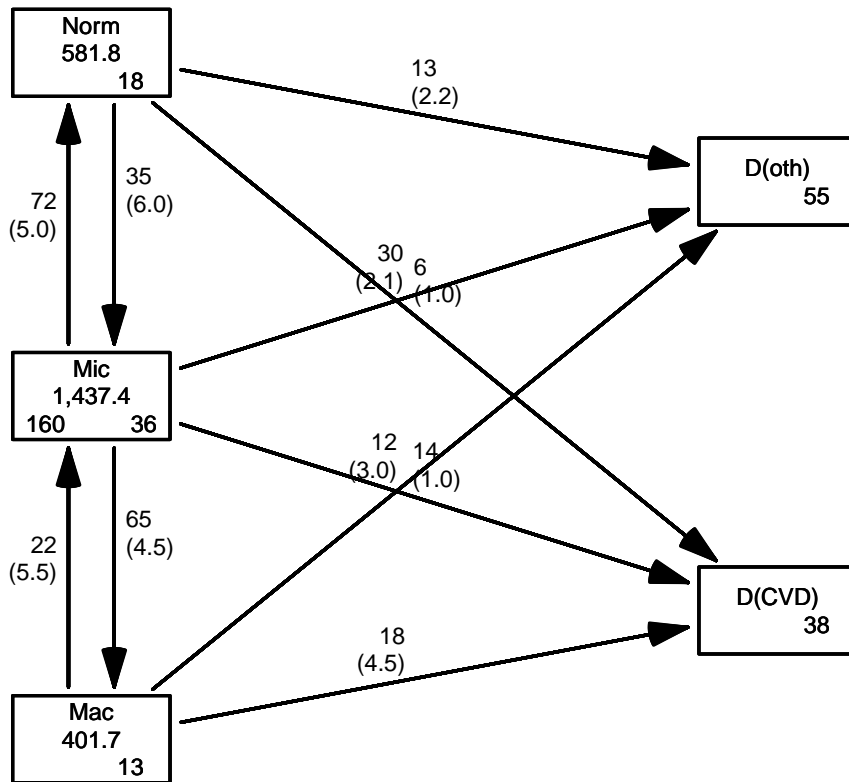


Figure 3.2: Transitions between states in the Steno2 study.

../graph/ms-b4

Transitions:

From	To	Mic	Norm	Mac	D(oth)	D(CVD)	Records:	Events:	Risk time:	Persons:
Mic		3107	72	65	30	14	3288	181	1437.39	160
Norm		35	1254	0	13	6	1308	54	581.83	66
Mac		22	0	847	12	18	899	52	401.70	60
Sum		3164	1326	912	55	38	5495	287	2420.91	160

We see that the number of events (transitions) and person-years are the same, but the number of records in S4 is substantially larger than in L4.

We can now 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(50, 80, 10)) +
+                               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
```

```
> ci.exp(ma)
```

	exp(Est.)	2.5%	97.5%
(Intercept)	0.002050421	0.0003671892	1.144975e-02
Ns(tfi, knots = seq(0, 20, 5))1	5.586238327	1.1524085205	2.707899e+01
Ns(tfi, knots = seq(0, 20, 5))2	3.948224386	0.9544630678	1.633219e+01
Ns(tfi, knots = seq(0, 20, 5))3	34.408040078	0.8997125880	1.315879e+03
Ns(tfi, knots = seq(0, 20, 5))4	0.466409150	0.1500745257	1.449530e+00
Ns(age, knots = seq(50, 80, 10))1	3.269829526	1.3358892679	8.003497e+00
Ns(age, knots = seq(50, 80, 10))2	11.582318649	1.4600392048	9.188117e+01
Ns(age, knots = seq(50, 80, 10))3	12.640207886	5.6379476934	2.833919e+01
lex.CstNorm	1.041469079	0.6062915725	1.789004e+00
lex.CstMac	1.772156120	1.1036543651	2.845580e+00

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 overall mortality rates are proportional between states of albuminuria.

The default for `glm.Lexis` is to model all transitions to absorbing states which in this case are the two “dead” states, `D(oth)` and `D(CVD)`.

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(50, 80, 10)) +
+            lex.Cst,
+          family = poisreg,
+          data = subset(S4, lex.Cst %in% c("Norm", "Mic", "Mac")))
```

which will also give the same results 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(50, 80, 10)) +
+            lex.Cst,
+          offset = log(lex.dur),
+          family = poisson,
+          data = subset(S4, lex.Cst %in% c("Norm", "Mic", "Mac")))
```

—note the difference between the families `poisreg` and `poisson` syntax; `poisson` enters person-years (`lex.dur`) via the `offset`, whereas `poisreg` enters it more logically as part of the outcome.

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	0.01
Ns(tfi, knots = seq(0, 20, 5))1	5.59	1.15	27.08
Ns(tfi, knots = seq(0, 20, 5))2	3.95	0.95	16.33
Ns(tfi, knots = seq(0, 20, 5))3	34.41	0.90	1315.88
Ns(tfi, knots = seq(0, 20, 5))4	0.47	0.15	1.45
Ns(age, knots = seq(50, 80, 10))1	3.27	1.34	8.00
Ns(age, knots = seq(50, 80, 10))2	11.58	1.46	91.88
Ns(age, knots = seq(50, 80, 10))3	12.64	5.64	28.34
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
6.11103822 2.00000000 0.04709827
```

So the mortality rates from the three states are not the same, but it is also quite clear that the mortality from state Mac is higher than from the two other states, that really do not differ.

- 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(50, 80, 10)) +
+                      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 7.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1  115.151 2.779 4.770588e+03
Ns(tfi, knots = seq(0, 20, 5))2   30.897 1.466 6.512260e+02
Ns(tfi, knots = seq(0, 20, 5))3 23342.027 4.716 1.155320e+08
Ns(tfi, knots = seq(0, 20, 5))4    1.737 0.302 1.000100e+01
Ns(age, knots = seq(50, 80, 10))1   2.745 0.901 8.360000e+00
Ns(age, knots = seq(50, 80, 10))2   2.053 0.208 2.028900e+01
Ns(age, knots = seq(50, 80, 10))3  12.979 4.637 3.633200e+01
lex.CstNorm      1.000 0.518 1.929000e+00
lex.CstMac       0.994 0.503 1.965000e+00

> mC <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      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)	0.001	0.000	0.012
Ns(tfi, knots = seq(0, 20, 5))1	1.079	0.165	7.069
Ns(tfi, knots = seq(0, 20, 5))2	1.932	0.305	12.254
Ns(tfi, knots = seq(0, 20, 5))3	1.143	0.018	73.365
Ns(tfi, knots = seq(0, 20, 5))4	0.129	0.016	1.065
Ns(age, knots = seq(50, 80, 10))1	6.419	1.069	38.564
Ns(age, knots = seq(50, 80, 10))2	417.853	1.795	97264.177
Ns(age, knots = seq(50, 80, 10))3	14.997	3.545	63.443
lex.CstNorm	1.091	0.416	2.859
lex.CstMac	3.513	1.719	7.179

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

Can you make a formal test of relevant hypotheses?

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

	Chisq	d.f.	P
	0.0002966161	2.0000000000	0.9998517030

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

	Chisq	d.f.	P
	13.764652275	2.0000000000	0.001025755

What is the conclusion w.r.t. mortality dependence on albuminuria status?

9. We can show how mortality rates look for persons currently in state Mic entering the study at a set of specific ages. The entry ages are in L2\$age:

```
> summary(L2$age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
37.39	48.52	56.60	55.13	61.06	67.50

So we shall use ages 45, 55 and 65, and show mortality rates for persons at these ages at entry as functions of their current age. We need a prediction data frame, with values for all variables in the model; here `expand.grid` is our friend:

```
> expand.grid(tfi = c(NA, seq(0, 20, 5)),
+           ain = c(45, 55, 65))
```

```

      tfi ain
1   NA  45
2    0  45
3    5  45
4   10  45
5   15  45
6   20  45
7   NA  55
8    0  55
9    5  55
10   10  55
11   15  55
12   20  55
13  NA  65
14   0  65
15   5  65
16  10  65
17  15  65
18  20  65

```

—it will give all combinations of the values in the vectors supplied as a `data.frame`. The NAs are there for plotting purposes— we get a break in plotted curves if there is an NA in the data. We want the `tfi` points to be closer than in the illustrative example:

```

> prf <- transform(expand.grid(tfi = c(NA, seq(0, 20, 0.5)),
+                               ain = c(45, 55, 65))[-1,],
+                   age = ain + tfi,
+                   lex.Cst = "Mic")
> head(prf)

```

```

      tfi ain age lex.Cst
2  0.0  45 45.0     Mic
3  0.5  45 45.5     Mic
4  1.0  45 46.0     Mic
5  1.5  45 46.5     Mic
6  2.0  45 47.0     Mic
7  2.5  45 47.5     Mic

```

```

> prf[40:44,]

```

```

      tfi ain age lex.Cst
41 19.5  45 64.5     Mic
42 20.0  45 65.0     Mic
43  NA  55  NA     Mic
44  0.0  55 55.0     Mic
45  0.5  55 55.5     Mic

```

```

> matshade(prf$age, cbind(ci.pred(mo, prf),
+                          ci.pred(mC, prf)) * 100,
+           lwd = 3, col = c("black", "blue"),
+           log = "y", ylim = c(0.01, 50), plot = TRUE)

```

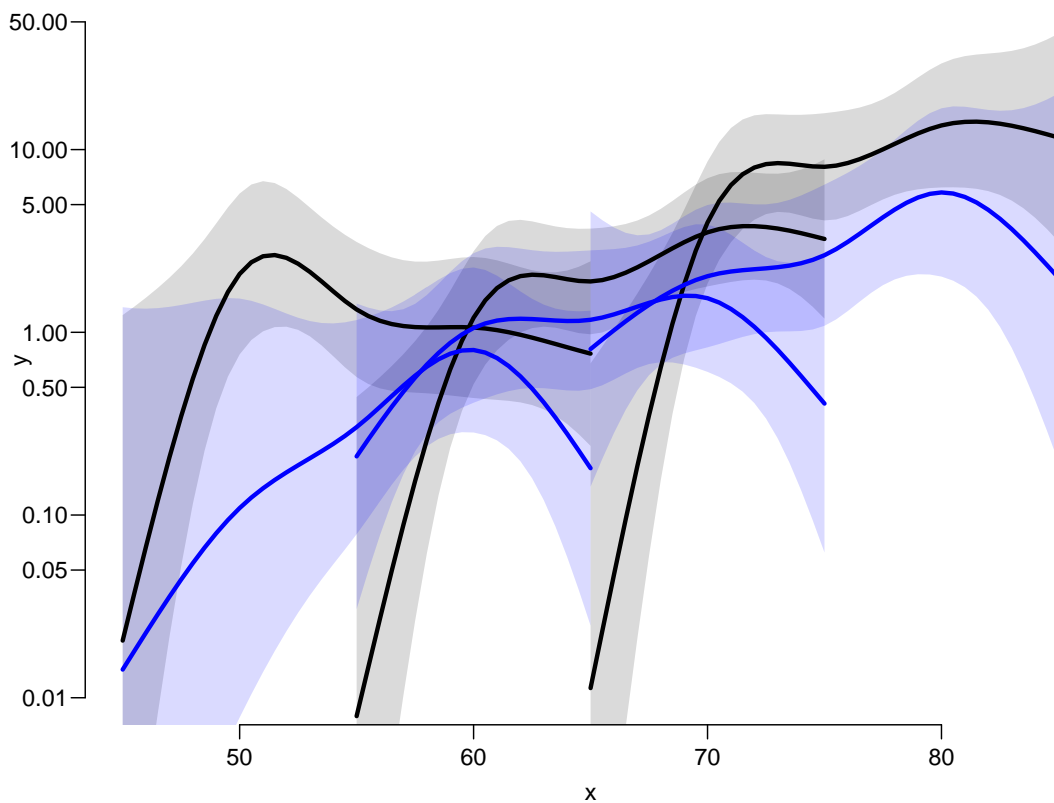


Figure 3.3: CVD mortality rates (blue) and non-CVD mortality rates (black), with 95% confidence intervals as shades. Curve represent persons entering the study at ages 45, 55 and 65 respectively.

../graph/ms-mort1

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 an informal description of the curves, and a possible reason for the shape 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,
+               lwd = 3, col = c("black","blue"),
+               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?

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

We would like to see how the probabilities of being in each of the states in figure 3.2 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) the cause-specific mortality rates and 2) transition rates between albuminuria states. And a model which includes the effect of `allo`.

11. We first model the mortality rates using a proportional hazards model, but allowing different mortality between the two allocation groups (in `allo`), and the three albuminuria states (in `lex.Cst`):

```
> mix <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst * allo,
+                      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(mix), 3)
```

	exp(Est.)	2.5%	97.5%
(Intercept)	0.000	0.000	5.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1	138.431	3.177	6.032407e+03
Ns(tfi, knots = seq(0, 20, 5))2	36.322	1.653	7.981850e+02
Ns(tfi, knots = seq(0, 20, 5))3	35690.096	6.479	1.965958e+08
Ns(tfi, knots = seq(0, 20, 5))4	2.183	0.378	1.259800e+01
Ns(age, knots = seq(50, 80, 10))1	2.746	0.909	8.295000e+00
Ns(age, knots = seq(50, 80, 10))2	1.627	0.159	1.666400e+01
Ns(age, knots = seq(50, 80, 10))3	11.953	4.268	3.347400e+01
lex.CstNorm	1.039	0.388	2.786000e+00
lex.CstMac	1.686	0.665	4.275000e+00
alloConv	1.931	0.927	4.022000e+00
lex.CstNorm:alloConv	0.929	0.244	3.544000e+00
lex.CstMac:alloConv	0.336	0.086	1.314000e+00

We would however like to see the allocation effect on mortality separately for each albuminuria state; this is done by the `/` operator in the model formula (pronounced `allo` effect within `lex.Cst`):

```
> mox <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo,
+                      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(mox), 3)

              exp(Est.)  2.5%          97.5%
(Intercept)           0.000 0.000 5.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1  138.431 3.177 6.032407e+03
Ns(tfi, knots = seq(0, 20, 5))2   36.322 1.653 7.981850e+02
Ns(tfi, knots = seq(0, 20, 5))3 35690.096 6.479 1.965958e+08
Ns(tfi, knots = seq(0, 20, 5))4    2.183 0.378 1.259800e+01
Ns(age, knots = seq(50, 80, 10))1   2.746 0.909 8.295000e+00
Ns(age, knots = seq(50, 80, 10))2   1.627 0.159 1.666400e+01
Ns(age, knots = seq(50, 80, 10))3  11.953 4.268 3.347400e+01
lex.CstNorm              1.039 0.388 2.786000e+00
lex.CstMac                1.686 0.665 4.275000e+00
lex.CstMic:alloConv      1.931 0.927 4.022000e+00
lex.CstNorm:alloConv     1.794 0.590 5.455000e+00
lex.CstMac:alloConv      0.649 0.204 2.065000e+00

> c(deviance(mox), deviance(mix))

[1] 554.6063 554.6063

```

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

What is the meaning of the parameters?

12. We also need a similar model for the CVD-mortality:

```

> mCx <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo,
+                      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(mCx), 3)

              exp(Est.)  2.5%          97.5%
(Intercept)           0.000 0.000    0.009
Ns(tfi, knots = seq(0, 20, 5))1   0.928 0.141    6.105
Ns(tfi, knots = seq(0, 20, 5))2   2.202 0.357   13.586
Ns(tfi, knots = seq(0, 20, 5))3   1.012 0.016   65.082
Ns(tfi, knots = seq(0, 20, 5))4   0.110 0.012    0.976
Ns(age, knots = seq(50, 80, 10))1   6.836 1.113   41.984
Ns(age, knots = seq(50, 80, 10))2 558.246 2.052 151860.752
Ns(age, knots = seq(50, 80, 10))3  20.881 4.798   90.877
lex.CstNorm              1.244 0.307    5.044
lex.CstMac                1.544 0.380    6.272
lex.CstMic:alloConv      1.684 0.579    4.894
lex.CstNorm:alloConv     1.392 0.276    7.016
lex.CstMac:alloConv      4.880 1.372   17.355

```

What is the conclusion for the intervention effect on CVD mortality rates?

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

```
> det <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo,
+                      from = c("Norm","Mic"),
+                      to = c("Mic","Mac"))
```

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

```
> imp <- glm.Lexis(S4, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo,
+                      from = c("Mic","Mac"),
+                      to = c("Norm","Mic"))
```

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

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

	exp(Est.)	2.5%	97.5%
(Intercept)	0.030	0.015	0.060
Ns(tfi, knots = seq(0, 20, 5))1	0.606	0.232	1.584
Ns(tfi, knots = seq(0, 20, 5))2	0.264	0.075	0.931
Ns(tfi, knots = seq(0, 20, 5))3	0.243	0.041	1.440
Ns(tfi, knots = seq(0, 20, 5))4	0.218	0.061	0.784
Ns(age, knots = seq(50, 80, 10))1	2.029	0.852	4.831
Ns(age, knots = seq(50, 80, 10))2	3.477	0.927	13.042
Ns(age, knots = seq(50, 80, 10))3	2.712	0.762	9.645
lex.CstNorm	2.560	1.448	4.525
lex.CstMic:alloConv	1.964	1.178	3.277
lex.CstNorm:alloConv	0.488	0.221	1.080

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

	exp(Est.)	2.5%	97.5%
(Intercept)	0.207	0.131	0.326
Ns(tfi, knots = seq(0, 20, 5))1	0.255	0.079	0.825
Ns(tfi, knots = seq(0, 20, 5))2	0.059	0.009	0.383
Ns(tfi, knots = seq(0, 20, 5))3	0.042	0.007	0.240
Ns(tfi, knots = seq(0, 20, 5))4	0.201	0.039	1.050
Ns(age, knots = seq(50, 80, 10))1	0.825	0.281	2.420
Ns(age, knots = seq(50, 80, 10))2	0.351	0.070	1.763
Ns(age, knots = seq(50, 80, 10))3	0.583	0.070	4.873
lex.CstMac	1.064	0.469	2.415
lex.CstMic:alloConv	0.526	0.324	0.855
lex.CstMac:alloConv	1.338	0.543	3.294

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

```

                                exp(Est.) 2.5% 97.5%
lex.CstMic:alloConv             2.0  1.2  3.3
lex.CstNorm:alloConv            0.5  0.2  1.1

> 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  3.3

```

What was the meaning of “different models for `det` and `imp`”?

What do the parameters in the models represent?

Label the transitions in figure 3.2 with the models for each of the transitions.

3.4 Simulation of state probabilities

We now have statistical models for all transitions, two models for the cause specific 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. These probabilities depend on the age at entry to the study (because current age (`age`) and time since entry, (`tfi`) are both in the model). This problem can be approached in (at least) two different ways:

- Use a population with the same age-distribution as the entire study population
- 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 simulation¹.

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)

```

¹A 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>


```

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.

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

```

> Tr <- list(Norm = list(Mic = det,
+                       "D(oth)" = mox,
+                       "D(CVD)" = mCx),
+           Mic = list(Mac = det,
+                     Norm = imp,
+                     "D(oth)" = mox,
+                     "D(CVD)" = mCx),
+           Mac = list(Mic = imp,
+                     "D(oth)" = mox,
+                     "D(CVD)" = mCx))
> lapply(Tr, names)

$Norm
[1] "Mic"      "D(oth)"   "D(CVD)"

$Mic
[1] "Mac"      "Norm"     "D(oth)"   "D(CVD)"

$Mac
[1] "Mic"      "D(oth)"   "D(CVD)"

```

For example, the object `Tr$Norm$Mic` is a model for the transition rate $\text{Norm} \rightarrow \text{Mic}$; we see that there are 10 entries in the specification of `Tr`, corresponding to each of the 10 transitions in the diagram in figure 3.2. Note that some of the entries in `Tr` point to the same model; all the models fitted were actually joint models for more than one transition.

3. 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 person in ini
+               t.range = 20, # how long should we simulate before censoring
+               n.int = 200))# how many intervals for evaluating rates

   user  system elapsed
27.300  11.334  24.789

```

There is no guaranteed order of the states in the `Sorg` object, so we explicitly reorder the states:

```

> Sorg <- Relevel(Sorg,c("Norm", "Mic", "Mac", "D(CVD)", "D(oth)"))
> summary(Sorg, t = T)

Transitions:
  To
From  Norm  Mic  Mac D(CVD) D(oth)  Records:  Events: Risk time:  Persons:
  Norm  398  640   0    119   282     1439     1041   11621.26    1310
  Mic  1439  622 1311   279   580     4231     3609   27091.33    3200
  Mac   0    391  302   380   238     1311     1009   7604.01    1217
  Sum  1837 1653 1613   778  1100     6981     5659  46316.59    3200

```

```

Timescales:
per age tfi
"" "" ""

```

```

> nround(subset(Sorg, lex.id %in% 28:32), 2)

  lex.id  per  age  tfi  lex.dur  lex.Cst  lex.Xst  allo  cens
79     28 1993.37 49.94 0.00    0.78    Mic    Norm  Int 2013.37
80     28 1994.15 50.72 0.78    5.57   Norm    Mic  Int 2013.37
81     28 1999.72 56.29 6.35    1.10    Mic    Norm  Int 2013.37
82     28 2000.82 57.39 7.45    5.74   Norm  D(oth)  Int 2013.37
83     29 1993.37 49.94 0.00    2.11    Mic    Norm  Int 2013.37
84     29 1995.48 52.06 2.11    2.79   Norm  D(oth)  Int 2013.37
85     30 1993.37 49.94 0.00    7.15    Mic    Norm  Int 2013.37
86     30 2000.53 57.10 7.15    3.14   Norm  D(CVD)  Int 2013.37
87     31 1993.34 48.50 0.00    5.14    Mic    Norm  Int 2013.34
88     31 1998.47 53.64 5.14   14.86   Norm    Norm  Int 2013.34
89     32 1993.34 48.50 0.00    4.64    Mic    Mac   Int 2013.34
90     32 1997.98 53.14 4.64    0.65    Mac    Mic   Int 2013.34
91     32 1998.62 53.79 5.28   14.18   Mic  D(oth)  Int 2013.34

```

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

```

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

  1    2    3    4    5    6    7    8  Sum
874 1397  534  297   70   24    3    1 3200

```

What does this table mean?

4. Now we can just count how many of the original 3200 persons are in each of the states at each of a set of times; 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
 2.673  0.000  2.670

> str(Nst)

'table' int [1:201, 1:5] 0 42 88 118 168 197 230 259 290 310 ...
- attr(*, "dimnames")=List of 2
 ..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
 ..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

> head(Nst)

      State
when  Norm  Mic  Mac D(CVD) D(oth)
 0      0 3200   0      0      0
0.1    42 3131  25      2      0
0.2    88 3077  33      2      0
0.3   118 3030  46      6      0
0.4   168 2965  60      7      0
0.5   197 2914  81      8      0
```

This is however not necessarily a relevant summary; we would be interested in seeing how things look in each of the allocation groups, `Int` and `Conv`.

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

      State
when  Norm  Mic  Mac D(CVD) D(oth)
 0      0 1600   0      0      0
0.1    24 1569   6      1      0
0.2    55 1533  11      1      0
0.3    77 1506  15      2      0
0.4   105 1472  21      2      0
0.5   121 1443  34      2      0

> head(Nconv)
```

	State				
when	Norm	Mic	Mac	D(CVD)	D(oth)
0	0	1600	0	0	0
0.1	18	1562	19	1	0
0.2	33	1544	22	1	0
0.3	41	1524	31	4	0
0.4	63	1493	39	5	0
0.5	76	1471	47	6	0

If we divide each of these by 1600, we get the probabilities of being in each if the states at the different times since entry.

- If we want the cumulated state probabilities over states we can derive these by `pState`, that yields a matrix with the cumulative state probabilities.

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

'pState' num [1:201, 1:5] 0 0.015 0.0344 0.0481 0.0656 ...
- attr(*, "dimnames")=List of 2
..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

> head(Pint)
```

	State				
when	Norm	Mic	Mac	D(CVD)	D(oth)
0	0.000000	1.000000	1.000000	1	1
0.1	0.015000	0.995625	0.999375	1	1
0.2	0.034375	0.992500	0.999375	1	1
0.3	0.048125	0.989375	0.998750	1	1
0.4	0.065625	0.985625	0.998750	1	1
0.5	0.075625	0.977500	0.998750	1	1

Describe the structure of `Pst`.

There is a standard plotting method for a `pState` object, it will plot the stacked state probabilities stacked in the order given by the `perm` argument (not used here because they are already in the order we want):

```
> par(mfrow = c(1,2), mar=c(3,3,2,2))
> clr <- c("forestgreen", "orange", "red", "blue", gray(0.4))
> plot(Pint, col = clr, xlim = c(0, 20))
> # the survival curve
> lines(as.numeric(rownames(Pint)), Pint[, "Mac"], lwd = 4, col = "white")
> text(rownames(Pint)[150],
+       Pint[150,] - diff(c(0, Pint[150,]))/2,
+       colnames(Pint),
+       col = "white")
> plot(Pconv, col = clr, xlim = c(20, 0))
> # the survival curve
> lines(as.numeric(rownames(Pconv)), Pconv[, "Mac"], lwd = 4)
```

```

> text(rownames(Pconv)[150],
+      Pconv[150,] - diff(c(0, Pconv[150,]))/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. Describe

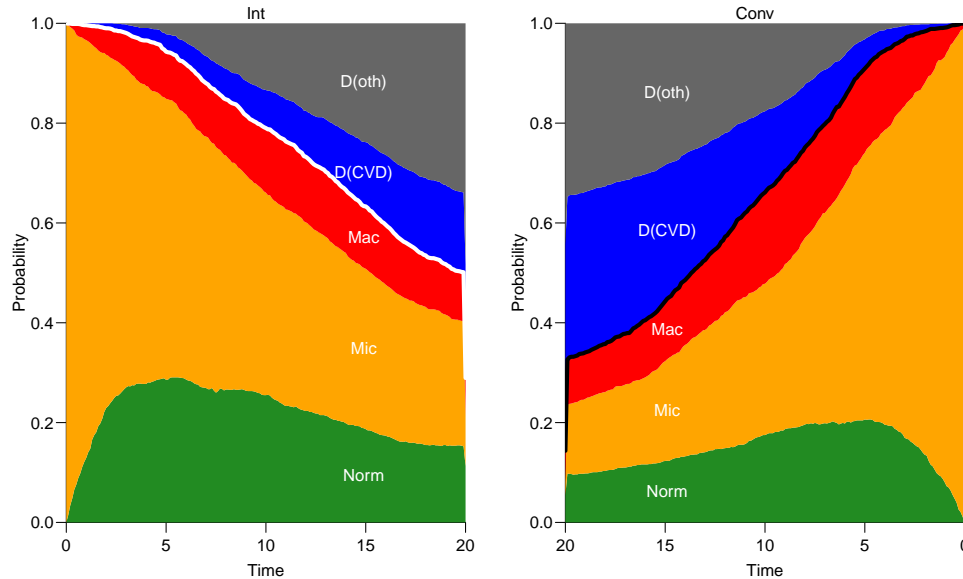


Figure 3.4: *State probabilities for the two intervention groups, for populations of the same structure as the original total Steno2 population.*

../graph/ms-pStates

the results and conclude on the probabilities shown.

- The plot 3.4 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, the absolute sizes of the state probabilities are irrelevant.

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 at select ages at entry. This would just require a different `init` data frame:

```

> ini <- S4[1:10,c("lex.id", "per", "age", "tfi", "lex.Cst", "allo")]
> ini[, "lex.id"] <- 1:10
> ini[, "per"] <- 1993 # not used but it is a time scale in S4
> ini[, "age"] <-
+ ini[, "ain"] <- rep(seq(45,65,5), 2)
> ini[, "tfi"] <- 0
> ini[, "lex.Cst"] <- factor("Mic",
+                           levels = c("Norm", "Mic", "Mac", "D(CVD)", "D(oth)"))
> ini[, "allo"] <- factor(rep(c("Int", "Conv"), each = 5))
> ini

  lex.id per age tfi lex.Cst allo ain
1      1 1993 45  0     Mic  Int  45
2      2 1993 50  0     Mic  Int  50
3      3 1993 55  0     Mic  Int  55
4      4 1993 60  0     Mic  Int  60
5      5 1993 65  0     Mic  Int  65
6      6 1993 45  0     Mic Conv 45
7      7 1993 50  0     Mic Conv 50
8      8 1993 55  0     Mic Conv 55
9      9 1993 60  0     Mic Conv 60
10     10 1993 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 1993 1993 1993 ...
 $ 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/ 5 levels "Norm","Mic","Mac",...: 2 2 2 2 2 2 2 2 2 2
 $ allo   : Factor w/ 2 levels "Conv","Int": 2 2 2 2 2 1 1 1 1 1
 $ 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 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...

```

Note that it is important that we enter the variable `lex.Cst` as a factor with the same levels as in the Lexis object `S4`, in the order we want the states when reporting results. `allo` must also be entered as a factor, otherwise it is not possible to compute predictions from the models where `allo` were included as a factor.

For each of these combinations of age (at entry) and treatment allocation we will simulate 100 persons (note that 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
      9.715  6.828   8.165

> # str(Sdef)
> summary(Sdef)

Transitions:
  To
From  Norm Mic Mac D(CVD) D(oth)  Records:  Events: Risk time:  Persons:
Norm  126 210  0     42    76     454     328     3667.94     407
Mic   454 210 402     89   180    1335    1125     8756.18    1000
Mac    0 125  87    122    68     402     315     2296.99     365
Sum   580 545 489    253   324    2191    1768    14721.11    1000

> nround(head(Sdef))

lex.id   per   age  tfi lex.dur lex.Cst lex.Xst allo ain cens
1        1 1993.00 45.00 0.00   0.06   Mic   Norm  Int  45 2013
2        1 1993.06 45.06 0.06  19.94  Norm  Norm  Int  45 2013
3        2 1993.00 45.00 0.00  20.00  Mic   Mic   Int  45 2013
4        3 1993.00 45.00 0.00  20.00  Mic   Mic   Int  45 2013
5        4 1993.00 45.00 0.00   3.94  Mic  D(oth) Int  45 2013
6        5 1993.00 45.00 0.00   8.19  Mic   Mac   Int  45 2013

```

In real applications we would use at least 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; we start with the 45 year old allocated to *Int*:

```

> P45i <- nState(subset(Sdef, ain == 45 & allo == "Int"),
+               at = seq(0, 20, 0.1),
+               from = 0,
+               time.scale = "tfi")
> head(P45i)

```

```

      State
when  Norm Mic Mac D(CVD) D(oth)
0     0 100  0     0     0
0.1   3  97  0     0     0
0.2   8  92  0     0     0
0.3  11  89  0     0     0
0.4  13  87  0     0     0
0.5  14  86  0     0     0

```

```

> head(pState(P45i))

```

```

      State
when  Norm Mic Mac D(CVD) D(oth)
0     0.00  1  1     1     1
0.1  0.03  1  1     1     1
0.2  0.08  1  1     1     1
0.3  0.11  1  1     1     1
0.4  0.13  1  1     1     1
0.5  0.14  1  1     1     1

```

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"))
> P45i <- pState(nState(subset(Sdef, ain == 45 & allo == "Int"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P50c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P50i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P55c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P55i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P60c <- pState(nState(subset(Sdef, ain == 55 & allo == "Conv"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P60i <- pState(nState(subset(Sdef, ain == 55 & allo == "Int"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P65c <- pState(nState(subset(Sdef, ain == 65 & allo == "Conv"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
> P65i <- pState(nState(subset(Sdef, ain == 65 & allo == "Int"),
+                       at = seq(0, 20, 0.1),
+                       from = 0,
+                       time.scale = "tfi"))
```

Then we can plot these in a multiple lay-out:

```
> par(mfrow = c(5,2), mar = c(1,1,0,0),
+     oma = c(3,3,1,0), mgp=c(3,1,0)/1.6)
> plot(P45i, col = clr, xlim = c(0,20))
> plot(P45c, col = clr, xlim = c(20,0))
> plot(P50i, col = clr, xlim = c(0,20))
> plot(P50c, col = clr, xlim = c(20,0))
> plot(P55i, col = clr, xlim = c(0,20))
> plot(P55c, col = clr, xlim = c(20,0))
```



```

> plot(P60i, col = clr, xlim = c(0,20))
> plot(P60c, col = clr, xlim = c(20,0))
> plot(P65i, col = clr, xlim = c(0,20))
> plot(P65c, col = clr, xlim = c(20,0))
> mtext(c("Int","Conv"), side = 3, at = c(1,3)/4, outer = TRUE, line = 0)
> mtext(paste(seq(45,65,5)), 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.

7. 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 the state probabilities in an array:

```

> (ain <- seq(45, 65, 5))
[1] 45 50 55 60 65

> (all <- levels(S4$allo))
[1] "Int" "Conv"

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

logi [1:5, 1:2, 1:201, 1:5] NA NA NA NA NA NA ...
- attr(*, "dimnames")=List of 4
..$ ain : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

```

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")
> pdef <- sweep(pdef, 1:2, pdef[,,1,"Mic"], "/")
> str(pdef)

num [1:5, 1:2, 1:201, 1:5] 0 0 0 0 0 0 0 0 0 0 0 ...
- attr(*, "dimnames")=List of 4
..$ ain : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

```

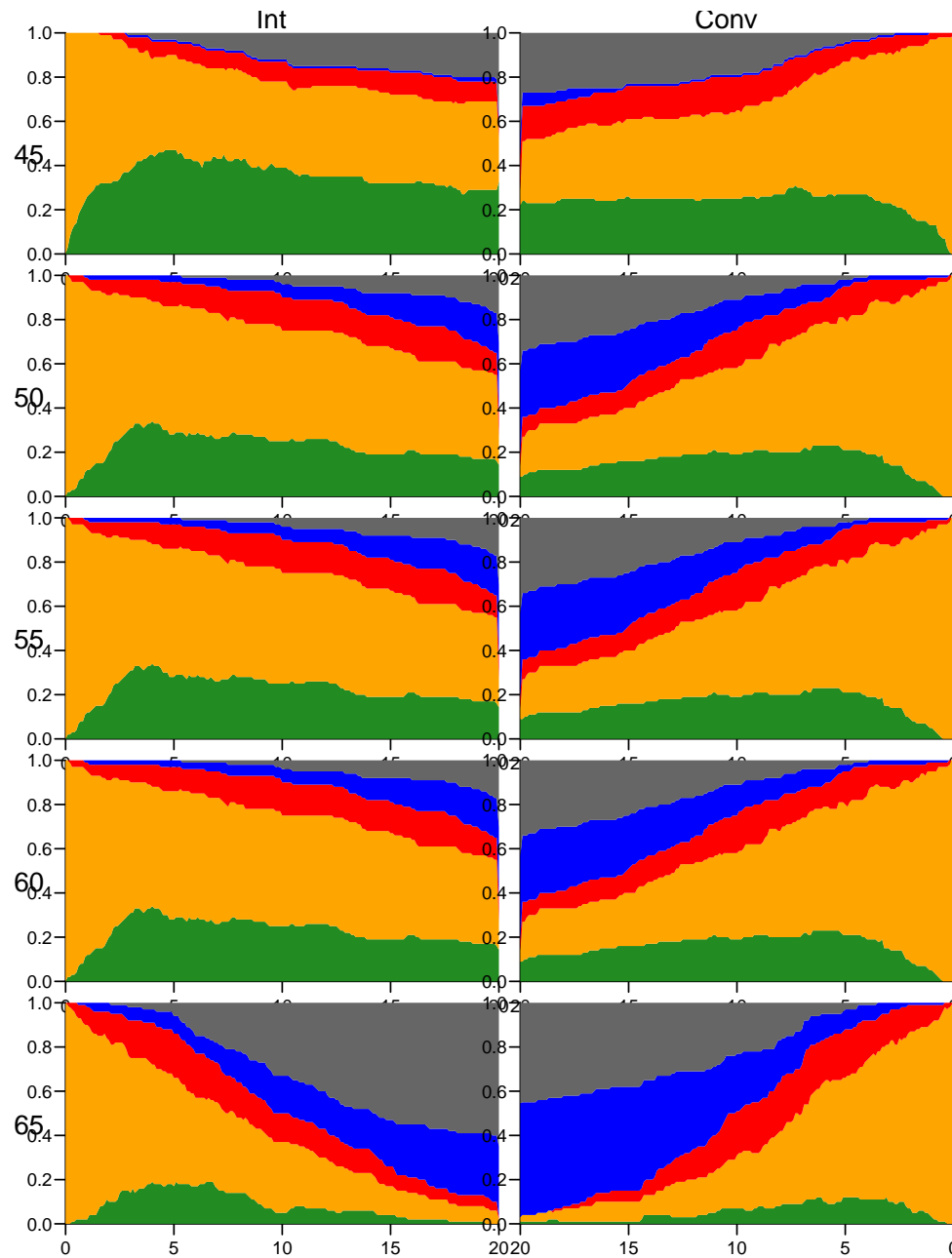


Figure 3.5: Predicted probabilities of being in each of the states for persons aged 45, 50, 55, 60 and 65 at entry, separately for the two intervention groups. `W ../graph/ms-panel5`

```
> ain <- seq(45, 65, 10)
> 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)

```

3.5 Time spent in albuminuria states

We have observation time till almost 22 years, but we have only made predictions of state probabilities in pdef till 20 years:

```

> str(pdef)
num [1:5, 1:2, 1:201, 1:5] 0 0 0 0 0 0 0 0 0 0 0 ...
- attr(*, "dimnames")=List of 4
..$ ain : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

> ftable(pdef[, , 1:5, ], row.vars = c(1,3))

```

ain	when	allo					Conv					
		State	Norm	Mic	Mac	D(CVD)	D(oth)	Norm	Mic	Mac	D(CVD)	D(oth)
45	0		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.1		0.03	0.97	0.00	0.00	0.00	0.00	0.98	0.02	0.00	0.00
	0.2		0.08	0.92	0.00	0.00	0.00	0.01	0.97	0.02	0.00	0.00
	0.3		0.11	0.89	0.00	0.00	0.00	0.03	0.95	0.02	0.00	0.00
	0.4		0.13	0.87	0.00	0.00	0.00	0.07	0.91	0.02	0.00	0.00
50	0		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.1		0.04	0.96	0.00	0.00	0.00	0.00	0.98	0.02	0.00	0.00
	0.2		0.05	0.95	0.00	0.00	0.00	0.00	0.98	0.02	0.00	0.00
	0.3		0.05	0.94	0.01	0.00	0.00	0.02	0.96	0.02	0.00	0.00
	0.4		0.07	0.92	0.01	0.00	0.00	0.03	0.94	0.03	0.00	0.00
55	0		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.1		0.02	0.98	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.2		0.02	0.97	0.01	0.00	0.00	0.00	0.98	0.02	0.00	0.00
	0.3		0.03	0.94	0.03	0.00	0.00	0.00	0.97	0.02	0.01	0.00
	0.4		0.03	0.94	0.03	0.00	0.00	0.00	0.97	0.02	0.01	0.00
60	0		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.1		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.2		0.03	0.97	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.3		0.04	0.96	0.00	0.00	0.00	0.01	0.96	0.03	0.00	0.00
	0.4		0.05	0.95	0.00	0.00	0.00	0.03	0.93	0.04	0.00	0.00
65	0		0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
	0.1		0.00	0.99	0.01	0.00	0.00	0.00	0.99	0.01	0.00	0.00
	0.2		0.00	0.98	0.02	0.00	0.00	0.00	0.98	0.02	0.00	0.00
	0.3		0.01	0.96	0.03	0.00	0.00	0.00	0.97	0.03	0.00	0.00
	0.4		0.01	0.95	0.04	0.00	0.00	0.01	0.97	0.02	0.00	0.00

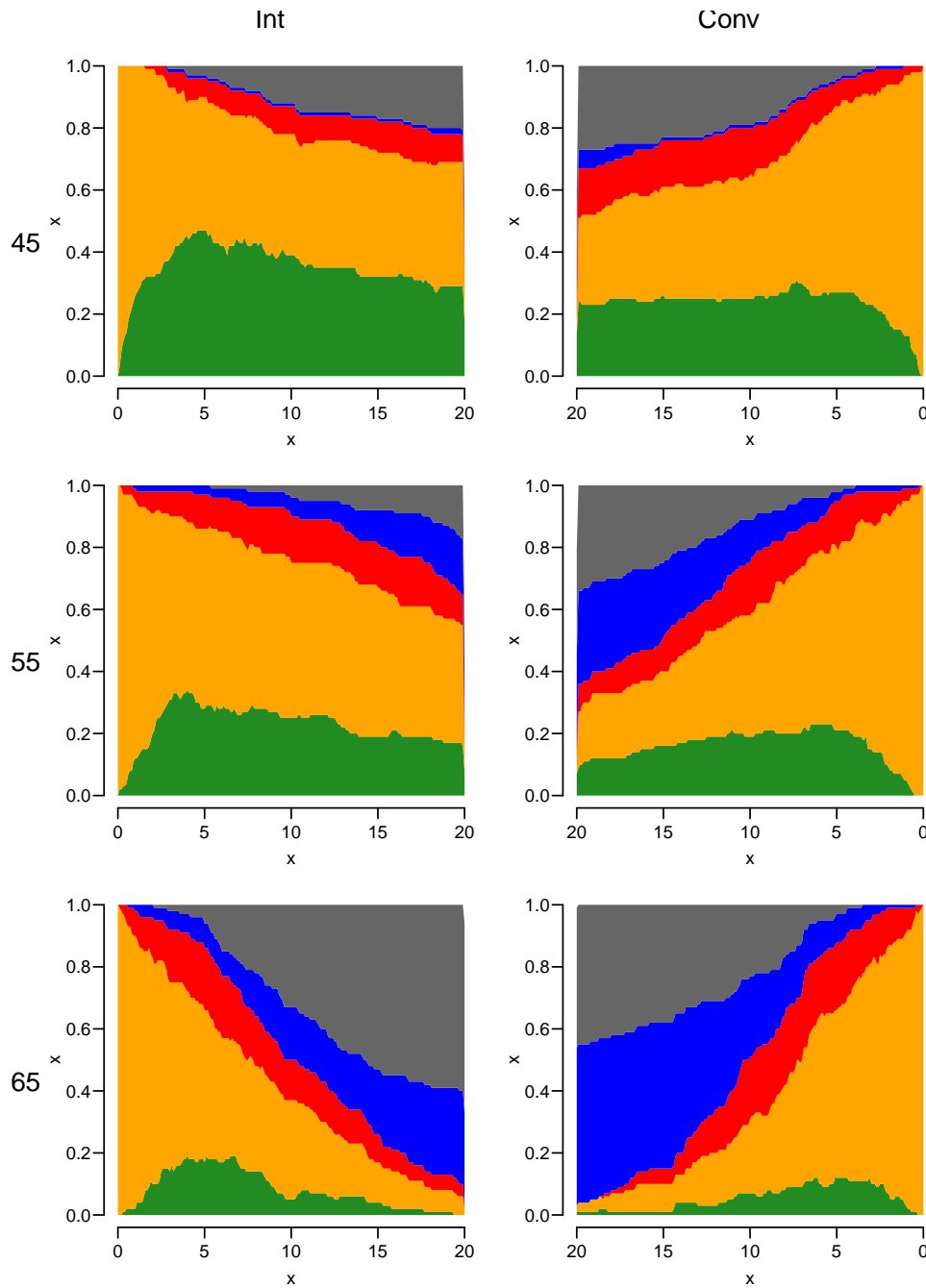


Figure 3.6: *Predicted probabilities of being in each of the states for persons aged 45, 55 and 65 at entry, separately for the two intervention groups.*

../graph/ms-panel13

8. We may want to compare groups by the expected time spent in the normoalbuminuric state during the first, say, 20 years. 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.

Be careful inspecting the results, it is not entirely obvious what apply does, keep track of the dimensions of each new table:

```
> mid <- function(x) x[-1] - diff(x) / 2
> str(pdef)

num [1:5, 1:2, 1:201, 1:5] 0 0 0 0 0 0 0 0 0 0 0 ...
- attr(*, "dimnames")=List of 4
..$ ain : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ when : chr [1:201] "0" "0.1" "0.2" "0.3" ...
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

> pmid <- apply(pdef, c(1,2,4), mid)
> str(pmid)

num [1:200, 1:5, 1:2, 1:5] 0.015 0.055 0.095 0.12 0.135 0.16 0.19 0.21 0.23 0.25 ...
- attr(*, "dimnames")=List of 4
..$      : chr [1:200] "0.1" "0.2" "0.3" "0.4" ...
..$ ain  : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

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

num [1:5, 1:2, 1:5] 7.03 5.43 4.53 3.87 1.61 ...
- attr(*, "dimnames")=List of 3
..$ ain : chr [1:5] "45" "50" "55" "60" ...
..$ allo : chr [1:2] "Int" "Conv"
..$ State: chr [1:5] "Norm" "Mic" "Mac" "D(CVD)" ...

> round(ftable(pyr, col.vars = 3:2), 1)

      State Norm      Mic      Mac      D(CVD)      D(oth)
      allo  Int Conv  Int Conv  Int Conv  Int Conv  Int Conv
ain
45      7.0  4.7  9.1  9.5  1.5  2.4  0.2  0.3  2.1  3.0
50      5.4  3.3  9.7  9.0  2.0  3.5  0.8  2.5  2.0  1.7
55      4.5  3.2 10.7  9.1  2.4  2.2  1.4  2.9  0.9  2.7
60      3.9  1.8  8.9  7.8  1.7  2.6  2.7  4.5  2.9  3.4
65      1.6  1.1  6.9  6.8  2.4  2.4  3.0  5.4  6.1  4.4
```

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, regardless of the age at entry.

The time spent in the two dead states are not really interpretable, it would be something like the number of years (during the first 20 years after enrollment) lost to each of the causes. We see that the most dramatic differences are for the CVD deaths.

Look at the differences:

```
> round(pyr[, "Int", ] - pyr[, "Conv", ], 1)

      State
ain  Norm  Mic  Mac D(CVD) D(oth)
45   2.3 -0.4 -0.9  -0.1  -0.9
50   2.1  0.8 -1.5  -1.7   0.2
55   1.3  1.7  0.2  -1.4  -1.7
60   2.0  1.1 -0.9  -1.8  -0.5
65   0.5  0.1 -0.1  -2.3   1.7
```

These are estimated times spent (sojourn times they are called) in each state. It is a bit strange to say that 55 year old enrollees in the intervention group spent 2.0 years less being dead from CVD than persons from the conventional group.

3.6 State probabilities using the Aalen-Johansen approach from survival

The `survival` package allows estimation of state probabilities by the Aalen-Johansen estimator similar to what we did in competing risks.

9. A direct application gives the wrong result:

```
> AaJ <- survfit(Surv(tfi, tfi + lex.dur, lex.Xst) ~ 1,
+               id = lex.id,
+               data = L4)
> AaJ$transitions
```

from	to				
	Norm	Mac	D(oth)	D(CVD)	(censored)
(s0)	63	55	19	9	14
Norm	96	5	17	10	16
Mac	3	46	19	19	6
D(oth)	0	0	0	0	0
D(CVD)	0	0	0	0	0

```
> summary(L4)
```

Transitions:

From	To					Records:	Events:	Risk time:	Persons:
	Mic	Norm	Mac	D(oth)	D(CVD)				
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

Comparing with the summary of L4 we see that we get the wrong number of transitions; we must use the `istate` argument:

```
> survfit(Surv(tfi, tfi + lex.dur, lex.Xst) ~ 1,
+         id = lex.id,
+         istate = lex.Cst,
+         data = L4)
```

...but this will crash. This is because the machinery does not allow records with null transitions, that is records that is just a transition from a given state to the same if it is the *last* record for a person (i.e. censorings in the last state). We therefore must rename these levels of `lex.Xst` to, say, `cens` (for `censored`, but any name will do), and this state must then be the `first` level of `lex.Xst`:

```
> R4 <- sortLexis(L4)
> last <- rev(!duplicated(rev(R4$lex.id)))
> R4$lex.Xst <- ifelse(last & R4$lex.Cst == R4$lex.Xst,
+                    "cens",
+                    as.character(R4$lex.Xst))
> R4 <- Relevel(factorize(R4), "cens")
```

NOTE: `lex.Cst` and `lex.Xst` now have levels:
 Mic Norm Mac cens D(CVD) D(oth)

```
> 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(R4)
```

Transitions:

	To										
From	cens	Mic	Norm	Mac	D(CVD)	D(oth)	Records:	Events:	Risk time:	Persons:	
Mic	36	276	72	65	14	30	493	217	1437.39	160	
Norm	18	35	72	0	6	13	144	72	581.83	66	
Mac	13	22	0	28	18	12	93	65	401.70	60	
Sum	67	333	144	93	38	55	730	354	2420.91	160	

Describe how the two Lexis objects are related.

- As mentioned, we must tell what state each record starts in, this is conveyed in the argument `istate` (`initial state`):

```
> AaJ <- survfit(Surv(tfi, tfi + lex.dur, lex.Xst) ~ 1,
+               id = lex.id,
+               istate = lex.Cst,
+               data = R4)
```

We see that we get the correct number of transitions when we merge the initial state `s(0)` with `Mic`:

```
> AaJ$transitions[,c(6,1:5)]
```

```

      to
from  (censored) Mic Norm Mac D(CVD) D(oth)
  Mic      36 276   72  65   14   30
  Norm     18  35   72   0    6   13
  Mac      13  22    0  28   18   12
  D(CVD)    0  0    0  0    0    0
  D(oth)    0  0    0  0    0    0

```

```
> summary(R4)
```

```
Transitions:
```

```

      To
From  cens Mic Norm Mac D(CVD) D(oth)  Records:  Events: Risk time:  Persons:
  Mic   36 276   72  65    14    30     493     217   1437.39     160
  Norm  18  35   72   0     6    13     144     72    581.83     66
  Mac   13  22    0  28    18    12     93     65    401.70     60
  Sum   67 333  144  93    38    55     730    354   2420.91    160

```

The predicted state probabilities are in the slot called `pstate`, and the confidence intervals in the corresponding slots `lower` and `upper`.

```
> names(AaJ)
```

```

 [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"

```

```
> AaJ$states
```

```
[1] "Mic" "Norm" "Mac" "D(CVD)" "D(oth)"
```

```
> head(AaJ$pstate)
```

```

      [,1] [,2] [,3] [,4] [,5]
[1,] 0.99375 0.00000 0.00625 0 0
[2,] 0.98750 0.00625 0.00625 0 0
[3,] 0.98750 0.00625 0.00625 0 0
[4,] 0.98125 0.01250 0.00625 0 0
[5,] 0.98125 0.01250 0.00625 0 0
[6,] 0.98125 0.01250 0.00625 0 0

```

```
> head(AaJ$lower)
```

```

      [,1] [,2] [,3] [,4] [,5]
[1,] 0.9816133 NA 0.0008858142 NA NA
[2,] 0.9704340 0.0008858142 0.0008858142 NA NA
[3,] 0.9704340 0.0008858142 0.0008858142 NA NA
[4,] 0.9604561 0.0031535032 0.0008858142 NA NA
[5,] 0.9604561 0.0031535032 0.0008858142 NA NA
[6,] 0.9604561 0.0031535032 0.0008858142 NA NA

```

```
> head(AaJ$upper)
```



```

      [,1]      [,2]      [,3] [,4] [,5]
[1,] 1          NA 0.04409785 NA  NA
[2,] 1 0.04409785 0.04409785 NA  NA
[3,] 1 0.04409785 0.04409785 NA  NA
[4,] 1 0.04954807 0.04409785 NA  NA
[5,] 1 0.04954807 0.04409785 NA  NA
[6,] 1 0.04954807 0.04409785 NA  NA

```

We can now show the Aalen-Johansen estimator of the state probabilities:

```

> mat2pol(AaJ$pstate, perm = c(2,1,3,5,4), x = AaJ$time,
+         col = clr)
> lines(AaJ$time, apply(AaJ$pstate[,1:3], 1, sum), lwd = 5)

```

11. But as above, we are interested in seeing the results from each of the allocation groups:

```

> AaJ <- survfit(Surv(tfi, tfi + lex.dur, lex.Xst) ~ allo,
+              id = lex.id,
+              istate = lex.Cst,
+              data = R4)

```

The result is in a long vector of `time` and `pstate`, the two parts corresponding to `Int` and `Conv` put after one another, with the length of each part in `strata`

```

> AaJ$states
[1] "Mic" "Norm" "Mac" "D(CVD)" "D(oth)"
> AaJ$strata
      allo=Int allo=Conv
      375      337
> wh <- rep(substr(names(AaJ$strata), 6, 9), AaJ$strata)
> table(wh)
wh
Conv Int
 337 375

```

So we just make the plots for the two subsets and place them next to each other as before:

```

> par(mfrow = c(1,2), mar=c(3,3,2,2))
> mat2pol(AaJ$pstate[wh=="Int",],
+         perm = c(2,1,3:5),
+         x = AaJ$time[wh=="Int"],
+         col = clr, xlim = c(0,21), xaxs = "i", yaxs = "i")
> lines(AaJ$time[wh=="Int"],
+       apply(AaJ$pstate[,1:3], 1, sum)[wh=="Int"], lwd = 4)
> mat2pol(AaJ$pstate[wh=="Conv",],

```

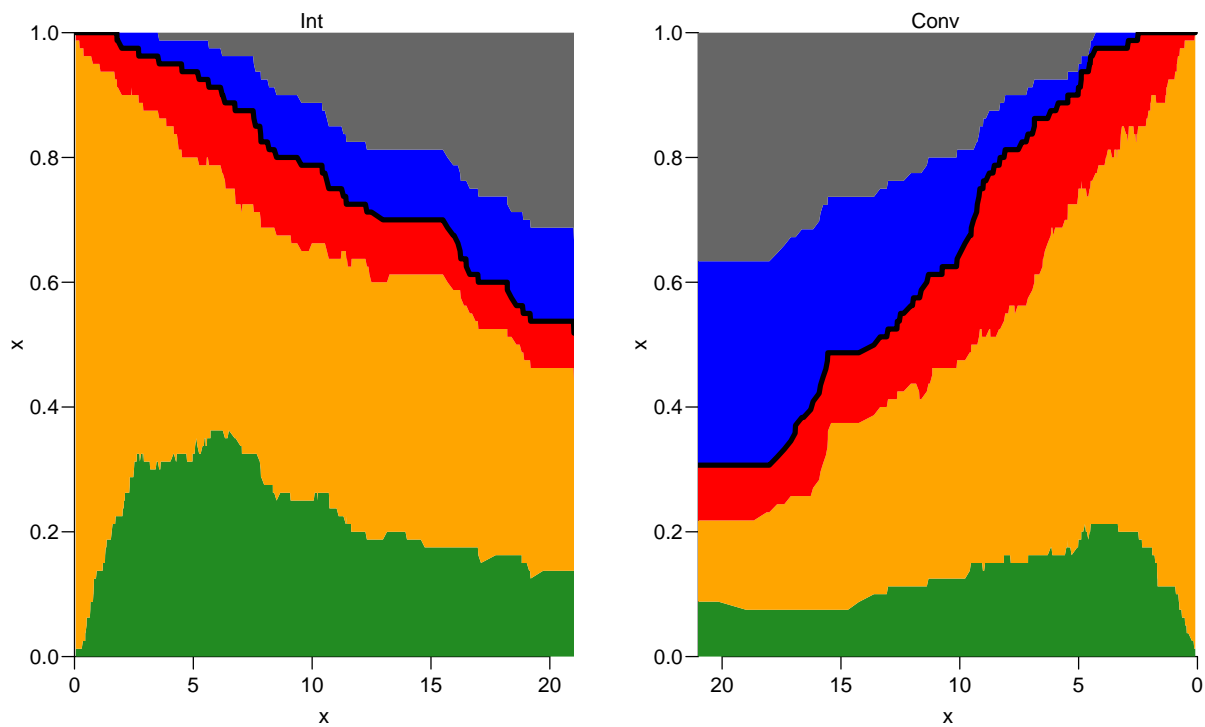


Figure 3.7: Aalen-Johansen estimator of the state probabilities for the two intervention groups, for the original total Steno2 population, subdivided by intervention allocation.

../graph/ms-AaJstates

```

+     perm = c(2,1,3:5),
+     x = AaJ$time[wh=="Conv"],
+     col = clr, xlim = c(21,0), xaxs = "i", yaxs = "i")
> lines(AaJ$time[wh=="Conv"],
+       apply(AaJ$pstate[,1:3], 1, sum)[wh=="Conv"], lwd = 4)
> mtext(c("Int","Conv"), side = 3, at = c(1,3)/4, outer = TRUE, line = -2)

```

This can be considered the empirical counterpart of figure 3.4; the state probabilities for a population as the one in the study. However not quite so; the models underlying figure 3.4 are proportional hazards in the sense that the effects of age and time since enrollment are proportional between state by allocation (6 groups for mortality, 4 groups for albuminuria state), whereas the figures in 3.7 are based on separate models for each transition and allocation.

12. We have confidence intervals for each of the state probabilities in the slots **lower** and **upper**, but not for the sums of these. And is the sums of state probabilities we have shown in the graph.

Moreover we would also want confidence intervals for areas under the curves. Neither are available from the Aalen-Johansen nor from the simulation approach. The simulation approach does not even give confidence intervals

3.7 Clinical variables

So far we have only considered covariates that we know the value of at any time point, including future time points, that is the allocation status and timescales such as age and time since inclusion.

1. In the dataset `st2clin` are clinical measurements taken at different dates, up to six different occasions per person:

```
> data(st2clin)
> str(st2clin)

'data.frame':      750 obs. of  5 variables:
 $ id   : num  1 2 3 4 5 6 7 8 9 10 ...
 $ doV  : Date, format: "1993-05-07" "1993-05-05" ...
 $ a1c  : num  87.3 66.5 73 61.2 102.7 ...
 $ chol : num  3.9 6.6 5.6 5.2 6 4.8 8.6 5.1 4.2 5.4 ...
 $ crea : num  83 83 68 97 149 55 56 78 123 79 ...

> st2clin <- rename(cal.yr(st2clin),
+                  lex.id = id,
+                  per = doV)
> summary(st2clin)

      lex.id      per      a1c      chol      crea
Min.   : 1.00   Min.   :1993   Min.   : 32.80   Min.   : 2.200   Min.   : 28.00
1st Qu.: 39.00   1st Qu.:1995   1st Qu.: 54.80   1st Qu.: 4.000   1st Qu.: 67.00
Median : 84.50   Median :1997   Median : 66.35   Median : 4.800   Median : 88.00
Mean   : 85.81   Mean   :2000   Mean   : 68.22   Mean   : 4.941   Mean   : 99.16
3rd Qu.:131.00   3rd Qu.:2002   3rd Qu.: 79.38   3rd Qu.: 5.700   3rd Qu.:115.25
Max.   :176.00   Max.   :2015   Max.   :147.60   Max.   :14.000   Max.   :1067.00
                NA's   :4           NA's   :3           NA's   :2

> addmargins(table(table(st2clin$lex.id)))

  1  2  3  4  5  6 Sum
  2  6 23 38 31 60 160
```

Explain the contents of the table.

2. We can use `addCov.Lexis` to amend the follow-up data with the clinical measurements:

```
> S5 <- addCov.Lexis(S4, st2clin, "per")
> tt <- table(st2clin$lex.id)
> (who <- names(tt[tt == 3])[1])

[1] "5"

> subset(st2clin, lex.id == who)

  lex.id      per      a1c chol crea
5       5 1993.151 102.7  6.0 149
165     5 1995.511  54.7  8.8 140
321     5 1997.496  41.9  5.8 141
```

```

> nround(subset(S5,
+             lex.id == who,
+             select = c(lex.id,per,tfi,tfc,exnam,a1c,chol,crea)))

  lex.id    per  tfi  tfc exnam   a1c chol  crea
159     5 1993.22 0.00 0.07  ex1 102.7  6.0  149
160     5 1993.72 0.50 0.57  ex1 102.7  6.0  149
161     5 1993.77 0.55 0.62  ex1 102.7  6.0  149
162     5 1994.22 1.00 1.07  ex1 102.7  6.0  149
163     5 1994.72 1.50 1.57  ex1 102.7  6.0  149
164     5 1995.22 2.00 2.07  ex1 102.7  6.0  149
165     5 1995.51 2.29 0.00  ex2  54.7  8.8  140
166     5 1995.72 2.50 0.21  ex2  54.7  8.8  140
167     5 1996.22 3.00 0.71  ex2  54.7  8.8  140
168     5 1996.72 3.50 1.21  ex2  54.7  8.8  140
169     5 1997.07 3.85 1.56  ex2  54.7  8.8  140
170     5 1997.22 4.00 1.71  ex2  54.7  8.8  140
171     5 1997.50 4.27 0.00  ex3  41.9  5.8  141
172     5 1997.72 4.50 0.23  ex3  41.9  5.8  141

> timeScales(S5)

[1] "per" "age" "tfi" "tfc"

> timeSince(S5)

per age tfi tfc
""  ""  ""  ""

```

We see that `tfc` is included as a time scale, but it is not a proper time scale; it is reset to 0 at every clinical visit, and it also has some missing values, as do the clinical variables. The missing values are where there is follow-up before the earliest clinical measurement for a person.

But it needs to be a time scale in the `Lexis` object in order to be properly handled when subsequently cutting and splitting a `Lexis` object.

3. The values of the clinical measurements in `st2clin` are added to the follow-up data: extra cutpoints at the measurement dates are added, and the values of the clinical variables are propagated as LOCF (Last Observation Carried Forward), so it is possible to model the effect of these clinical variables on transition rates—creatinine is traditionally modeled on a log-scale, here we use the base 2 logarithm.

```

> detc <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo +
+                      a1c + chol + log2(crea),
+                      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

```

```
> impc <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo +
+                      a1c + chol + log2(crea),
+                      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(detc), 3)
```

	exp(Est.)	2.5%	97.5%
(Intercept)	0.033	0.002	0.553
Ns(tfi, knots = seq(0, 20, 5))1	0.680	0.259	1.788
Ns(tfi, knots = seq(0, 20, 5))2	0.280	0.078	1.008
Ns(tfi, knots = seq(0, 20, 5))3	0.244	0.040	1.478
Ns(tfi, knots = seq(0, 20, 5))4	0.228	0.063	0.830
Ns(age, knots = seq(50, 80, 10))1	2.096	0.880	4.993
Ns(age, knots = seq(50, 80, 10))2	4.283	1.096	16.735
Ns(age, knots = seq(50, 80, 10))3	3.358	0.906	12.447
lex.CstNorm	2.587	1.459	4.589
a1c	1.005	0.993	1.018
chol	1.090	0.910	1.307
log2(crea)	0.866	0.583	1.285
lex.CstMic:alloConv	1.702	0.977	2.964
lex.CstNorm:alloConv	0.433	0.193	0.973

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

	exp(Est.)	2.5%	97.5%
(Intercept)	1.085	0.061	19.162
Ns(tfi, knots = seq(0, 20, 5))1	0.247	0.076	0.804
Ns(tfi, knots = seq(0, 20, 5))2	0.059	0.009	0.386
Ns(tfi, knots = seq(0, 20, 5))3	0.041	0.007	0.248
Ns(tfi, knots = seq(0, 20, 5))4	0.190	0.036	1.001
Ns(age, knots = seq(50, 80, 10))1	0.838	0.288	2.442
Ns(age, knots = seq(50, 80, 10))2	0.363	0.071	1.848
Ns(age, knots = seq(50, 80, 10))3	0.592	0.071	4.927
lex.CstMac	1.059	0.468	2.396
a1c	0.991	0.978	1.003
chol	0.963	0.803	1.155
log2(crea)	0.872	0.580	1.313
lex.CstMic:alloConv	0.598	0.359	0.996
lex.CstMac:alloConv	1.523	0.610	3.799

```
> moc <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo +
+                      a1c + chol + log2(crea),
+                      to = "D(oth)")
```

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

```
> mCc <- glm.Lexis(S5, ~ Ns(tfi, knots = seq( 0, 20, 5)) +
+                      Ns(age, knots = seq(50, 80, 10)) +
+                      lex.Cst / allo +
+                      a1c + chol + log2(crea),
+                      to = "D(CVD)")
```

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

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

	exp(Est.)	2.5%	97.5%
(Intercept)	0.000	0.000	1.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1	145.699	3.077	6.898293e+03
Ns(tfi, knots = seq(0, 20, 5))2	25.400	1.041	6.198740e+02
Ns(tfi, knots = seq(0, 20, 5))3	36604.436	5.048	2.654028e+08
Ns(tfi, knots = seq(0, 20, 5))4	1.751	0.286	1.074000e+01
Ns(age, knots = seq(50, 80, 10))1	2.115	0.676	6.610000e+00
Ns(age, knots = seq(50, 80, 10))2	1.119	0.108	1.161500e+01
Ns(age, knots = seq(50, 80, 10))3	8.945	2.938	2.723000e+01
lex.CstNorm	1.033	0.384	2.778000e+00
lex.CstMac	1.341	0.504	3.567000e+00
a1c	1.005	0.987	1.024000e+00
chol	0.845	0.635	1.124000e+00
log2(crea)	1.849	1.140	3.002000e+00
lex.CstMic:alloConv	1.936	0.875	4.288000e+00
lex.CstNorm:alloConv	1.887	0.602	5.920000e+00
lex.CstMac:alloConv	0.771	0.233	2.553000e+00

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

	exp(Est.)	2.5%	97.5%
(Intercept)	0.000	0.000	0.011
Ns(tfi, knots = seq(0, 20, 5))1	0.873	0.131	5.824
Ns(tfi, knots = seq(0, 20, 5))2	1.883	0.275	12.886
Ns(tfi, knots = seq(0, 20, 5))3	0.802	0.011	58.470
Ns(tfi, knots = seq(0, 20, 5))4	0.108	0.012	1.000
Ns(age, knots = seq(50, 80, 10))1	6.213	0.975	39.600
Ns(age, knots = seq(50, 80, 10))2	525.886	1.826	151495.801
Ns(age, knots = seq(50, 80, 10))3	19.071	4.167	87.283
lex.CstNorm	1.248	0.307	5.069
lex.CstMac	1.416	0.343	5.853
a1c	0.999	0.980	1.019
chol	1.007	0.738	1.374
log2(crea)	1.346	0.755	2.399
lex.CstMic:alloConv	1.674	0.550	5.091
lex.CstNorm:alloConv	1.384	0.269	7.115
lex.CstMac:alloConv	5.068	1.386	18.529

Only `crea` has any effect; a doubling of creatinine is associated with a 1.85 times higher mortality rate from other (non-CVD) causes. Confidence interval is (1.14,3.00), so not terribly precisely determined.

There are limitations in using clinical measurements as time-dependent variables without a model for the clinical variables. In order to simulate events based on

models for transition rates we must know all covariates at all times, so models with non-deterministically varying are not usable. Timescales are time-varying covariate, but they vary deterministically, so their value for each person will be known at any time of follow-up.

So the models with effects of clinical variables as presented here cannot be used for prediction of state probabilities—that would requires some kind of model for the clinical variables over time as well.

3.8 A single model for transitions between microvascular complications states: `stack`

So far, we have only jointly modeled transitions that originated in *different* states:

```
Mic → Mac and Norm → Mic;
Norm → D(CVD), Mic → D(CVD) and Mac → D(CVD).
```

As long as the different rates modeled are originating in *different* states, the likelihood will have at most one contribution from each record in the `Lexis` follow-up data set. But if we want to create a joint model for more than one rate originating in a given state we must repeat some of risk time in different contributions to the likelihood. This means that the modeling cannot be based on (subsets of) a `Lexis` object, we must repeat some records. This is detailed in dection 3.2.2 on compring risks in the PMM (Practical Multistate Modeling, <http://bendixcarstensen.com/MSbook.pdf>, very preliminary).

This is achieved by the `stack.Lexis` function:

```
> St4 <- stack(S4)
NOTE: lex.Cst and lex.Xst now have levels:
  Mic Norm Mac D(oth) D(CVD)
> c(nrow(S4), nrow(St4))
[1] 5495 19773
> table(S4$lex.Cst)
  Mic  Norm  Mac D(oth) D(CVD)
3288  1308  899     0     0
> table(St4$lex.Tr, St4$lex.Cst)
      Mic Norm  Mac D(oth) D(CVD)
Mic->Norm 3288  0     0     0     0
Mic->Mac   3288  0     0     0     0
Mic->D(oth) 3288  0     0     0     0
Mic->D(CVD) 3288  0     0     0     0
Norm->Mic   0 1308  0     0     0
Norm->D(oth) 0 1308  0     0     0
Norm->D(CVD) 0 1308  0     0     0
Mac->Mic    0  0  899  0     0
Mac->D(oth) 0  0  899  0     0
Mac->D(CVD) 0  0  899  0     0
> ftable(St4$lex.Tr, St4$lex.Xst, St4$lex.Fail, col.vars = 2:3)
```

	Mic		Norm		Mac		D(oth)		D(CVD)	
	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE
Mic->Norm	3107	0	0	72	65	0	30	0	14	0
Mic->Mac	3107	0	72	0	0	65	30	0	14	0
Mic->D(oth)	3107	0	72	0	65	0	0	30	14	0
Mic->D(CVD)	3107	0	72	0	65	0	30	0	0	14
Norm->Mic	0	35	1254	0	0	0	13	0	6	0
Norm->D(oth)	35	0	1254	0	0	0	0	13	6	0
Norm->D(CVD)	35	0	1254	0	0	0	13	0	0	6
Mac->Mic	0	22	0	0	847	0	12	0	18	0
Mac->D(oth)	22	0	0	0	847	0	0	12	18	0
Mac->D(CVD)	22	0	0	0	847	0	12	0	0	18

We see that the `lex.Fail` is only TRUE where `lex.Xst` is equal to the second part of the `lex.Tr`.

```
> table(tt <- table(S4$lex.id))
 4  7  9 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 33 35 36 37 38 39
 2  3  1  1  5  3  2  3  3  1  3  2  3  3  5  4  6  2  3  2  3  2  1  2  1  1  1  4  8  6  1
40 41 42 43 45 46 47 48 49
 1  3  3  1  5  8 23 19 11
> nround(subset(S4 , lex.id == 102)[,1:8], 1)
lex.id per age tfi lex.dur lex.Cst lex.Xst id
3348 102 1993.5 58.3 0.0 0.5 Mic Mic 102
3349 102 1994.0 58.8 0.5 0.5 Mic Mic 102
3350 102 1994.5 59.3 1.0 0.5 Mic Mic 102
3351 102 1995.0 59.8 1.5 0.3 Mic D(CVD) 102
> nround(subset(St4, lex.id == 102)[,1:9], 1)
lex.id per age tfi lex.dur lex.Cst lex.Xst lex.Tr lex.Fail
3348 102 1993.5 58.3 0.0 0.5 Mic Mic Mic->Norm FALSE
3349 102 1994.0 58.8 0.5 0.5 Mic Mic Mic->Norm FALSE
3350 102 1994.5 59.3 1.0 0.5 Mic Mic Mic->Norm FALSE
3351 102 1995.0 59.8 1.5 0.3 Mic D(CVD) Mic->Norm FALSE
33481 102 1993.5 58.3 0.0 0.5 Mic Mic Mic->Mac FALSE
33491 102 1994.0 58.8 0.5 0.5 Mic Mic Mic->Mac FALSE
33501 102 1994.5 59.3 1.0 0.5 Mic Mic Mic->Mac FALSE
33511 102 1995.0 59.8 1.5 0.3 Mic D(CVD) Mic->Mac FALSE
33482 102 1993.5 58.3 0.0 0.5 Mic Mic Mic->D(oth) FALSE
33492 102 1994.0 58.8 0.5 0.5 Mic Mic Mic->D(oth) FALSE
33502 102 1994.5 59.3 1.0 0.5 Mic Mic Mic->D(oth) FALSE
33512 102 1995.0 59.8 1.5 0.3 Mic D(CVD) Mic->D(oth) FALSE
33483 102 1993.5 58.3 0.0 0.5 Mic Mic Mic->D(CVD) FALSE
33493 102 1994.0 58.8 0.5 0.5 Mic Mic Mic->D(CVD) FALSE
33503 102 1994.5 59.3 1.0 0.5 Mic Mic Mic->D(CVD) FALSE
33513 102 1995.0 59.8 1.5 0.3 Mic D(CVD) Mic->D(CVD) TRUE
```

Suppose we wanted to fit a model for the two types of mortality assuming that, say, the effect of sex was the same.

Since some of the transitions we put in the same model originate from the same state we need the stacked data representation where each record corresponds to a likelihood term.


```
> cbind(with(subset(St4, grepl("D", lex.Tr)), table(lex.Tr)))
      [,1]
Mic->Norm      0
Mic->Mac        0
Mic->D(oth)  3288
Mic->D(CVD)  3288
Norm->Mic       0
Norm->D(oth) 1308
Norm->D(CVD) 1308
Mac->Mic        0
Mac->D(oth)   899
Mac->D(CVD)   899
```

We can then fit a model with common effect of

```
> stD <- glm(cbind(lex.Fail, lex.dur)
+           ~ Ns(tfi, knots = seq( 0, 20,  5)) * lex.Tr +
+           Ns(age, knots = seq(50, 80, 10)) * lex.Tr +
+           lex.Tr / allo + sex,
+           family = poisreg,
+           offset = log(lex.dur),
+           data = subset(St4, grepl("D", lex.Tr)))
> round(ci.exp(stD)[,1,drop=F],3)
exp(Est.)
(Intercept) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))1 8.353772e+03
Ns(tfi, knots = seq(0, 20, 5))2 7.012630e+02
Ns(tfi, knots = seq(0, 20, 5))3 9.369302e+07
Ns(tfi, knots = seq(0, 20, 5))4 1.088500e+01
lex.TrMic->D(CVD) 2.180430e+02
lex.TrNorm->D(oth) 3.843160e+02
lex.TrNorm->D(CVD) 0.000000e+00
lex.TrMac->D(oth) 0.000000e+00
lex.TrMac->D(CVD) 6.140100e+01
Ns(age, knots = seq(50, 80, 10))1 1.918000e+00
Ns(age, knots = seq(50, 80, 10))2 1.377000e+00
Ns(age, knots = seq(50, 80, 10))3 1.176700e+01
sexM 1.457000e+00
Ns(tfi, knots = seq(0, 20, 5))1:lex.TrMic->D(CVD) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))2:lex.TrMic->D(CVD) 3.000000e-03
Ns(tfi, knots = seq(0, 20, 5))3:lex.TrMic->D(CVD) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))4:lex.TrMic->D(CVD) 9.000000e-03
Ns(tfi, knots = seq(0, 20, 5))1:lex.TrNorm->D(oth) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))2:lex.TrNorm->D(oth) 7.000000e-03
Ns(tfi, knots = seq(0, 20, 5))3:lex.TrNorm->D(oth) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))4:lex.TrNorm->D(oth) 1.300000e-02
Ns(tfi, knots = seq(0, 20, 5))1:lex.TrNorm->D(CVD) 2.102100e+01
Ns(tfi, knots = seq(0, 20, 5))2:lex.TrNorm->D(CVD) 1.752356e+03
Ns(tfi, knots = seq(0, 20, 5))3:lex.TrNorm->D(CVD) 2.128717e+03
Ns(tfi, knots = seq(0, 20, 5))4:lex.TrNorm->D(CVD) 0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))1:lex.TrMac->D(oth) 6.686980e+64
Ns(tfi, knots = seq(0, 20, 5))2:lex.TrMac->D(oth) 3.825428e+46
Ns(tfi, knots = seq(0, 20, 5))3:lex.TrMac->D(oth) 8.032855e+125
Ns(tfi, knots = seq(0, 20, 5))4:lex.TrMac->D(oth) 6.162919e+26
Ns(tfi, knots = seq(0, 20, 5))1:lex.TrMac->D(CVD) 1.000000e-03
```

```

Ns(tfi, knots = seq(0, 20, 5))2:lex.TrMac->D(CVD)      1.900000e-02
Ns(tfi, knots = seq(0, 20, 5))3:lex.TrMac->D(CVD)      0.000000e+00
Ns(tfi, knots = seq(0, 20, 5))4:lex.TrMac->D(CVD)      3.500000e-02
lex.TrMic->D(CVD):Ns(age, knots = seq(50, 80, 10))1    7.103000e+00
lex.TrNorm->D(oth):Ns(age, knots = seq(50, 80, 10))1   4.103000e+00
lex.TrNorm->D(CVD):Ns(age, knots = seq(50, 80, 10))1   8.326000e+00
lex.TrMac->D(oth):Ns(age, knots = seq(50, 80, 10))1    7.770000e-01
lex.TrMac->D(CVD):Ns(age, knots = seq(50, 80, 10))1    3.751000e+00
lex.TrMic->D(CVD):Ns(age, knots = seq(50, 80, 10))2    1.085309e+03
lex.TrNorm->D(oth):Ns(age, knots = seq(50, 80, 10))2   2.612000e+00
lex.TrNorm->D(CVD):Ns(age, knots = seq(50, 80, 10))2   6.400000e-02
lex.TrMac->D(oth):Ns(age, knots = seq(50, 80, 10))2    1.408000e+00
lex.TrMac->D(CVD):Ns(age, knots = seq(50, 80, 10))2    1.853050e+02
lex.TrMic->D(CVD):Ns(age, knots = seq(50, 80, 10))3    1.191000e+00
lex.TrNorm->D(oth):Ns(age, knots = seq(50, 80, 10))3   1.672000e+00
lex.TrNorm->D(CVD):Ns(age, knots = seq(50, 80, 10))3   0.000000e+00
lex.TrMac->D(oth):Ns(age, knots = seq(50, 80, 10))3    8.400000e-01
lex.TrMac->D(CVD):Ns(age, knots = seq(50, 80, 10))3    6.247000e+00
lex.TrMic->D(oth):alloConv                             2.125000e+00
lex.TrMic->D(CVD):alloConv                             1.699000e+00
lex.TrNorm->D(oth):alloConv                             1.788000e+00
lex.TrNorm->D(CVD):alloConv                             2.063000e+00
lex.TrMac->D(oth):alloConv                             6.290000e-01
lex.TrMac->D(CVD):alloConv                             9.182000e+00

```

So under the assumption that the sex-effects are the same for all 6 sets of mortality rates the M/W rate ratio is 1.45.

But it is only rare that we want to model different rates out of the same state, so the actual use of `stack(.Lexis)` is seldom needed.

You should however be aware that when using the `mstate` package, follow-up is stored as stacked objects

References

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