Survival, mortality, competing risks and expected lifetime

Computer practicals

EDEG 2025 / Umeå University
17 May 2025
http://bendixcarstensen.com/AdvCoh/courses/Um-2025/
Version 6, May 2025

Compiled Saturday 10th May, 2025, 16:14
from: C:\Bendix\teach\AdvCoh\courses\Um-2025\pracs/pracs.tex

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

Contents

Preface

This workshop will provide an overview of the concepts in the title with a special view to generating results (numbers and graphs) using R.

By the title of the workshop it will (hopefully) be relevant for persons that are working with follow-up data over time, be that clinical trials, cohort studies or register-based studies.

- The *target audience* is (young) statisticians and epidemiologists working in (diabetes) epidemiological research
- The *prerequisites* are
  1. a basic knowledge of R,
  2. a working installation of R(latest version, 4.5.0)
  3. a working installation of the latest version of the `Epi` package (2.59)
  4. a working installation of the latest version of the `popEpi` package (0.4.13)
  5. some epidemiological practice
  6. a basic knowledge of elementary biostatistics (you should know what a confidence interval is)
- The *format* of the workshop will be short lectures closely aligned with the topics in the exercises. The exercises will be run in chunks between the short lectures.
- The *mood* of the workshop will be relaxed, encouraging participants to ask questions and bring forward problems they consider relevant for the workshop. Fortunately, there will be the rest of the EDEG to interact.

Exercises are given including substantial parts of the solutions. You can get the exercise code chunks from the workshop website http://bendixcarstensen.com/AdvCoh/courses/Um-2025. So the practicals is essentially to run the code given, which will save you a lot of typing. On the other hand you run the risk of not paying attention to the concepts that is illustrated by the code. So try to add at least one line of code of your own to look into what you have done.

This workshop draws to some extent on the content of the book "Epidemiology with R" [3], (http://bendixcarstensen.com/EwR), but in particular on the draft book (which by no means is sure ever to appear as a book) "Practical multistate modeling with R and `Epi:Lexis`", [4]. The former is available through Oxford University Press, the latter as a draft (updated at unpredictable times) as http://bendixcarstensen.com/MSbook.pdf (200+ pages...).

Program of workshop

Each item on the program is a short(ish) lecture followed by a computer practical in R. The timing of the items is approximate.

The purpose of the workshop is to provide a hands-on experience of computing some the most common quantities in follow-up studies (and some not so common).

Saturday 17 May 2025	
09:00–09:10	Welcome and introdiction
09:10–10:00	Survival and mortality rates Kaplan-Meier survival, mortality function, parametric survival function
10:00–10:30	Cause specific rates
10:30–11:40	Competing risks
11:40–12:20	Lunch
12:20–13:45	Expected lifetime (RMST)
13:45–14:00	Wrap-up and questions

Chapter 1

Survival and mortality

. . . now input from `surv.tex`

```
> library(Epi)
> library(popEpi)
> library(survival)
> library(tidyverse)
> clear()
```

```
     R  Epi popEpi
 4.5.0 2.59 0.4.13
```

1.1 Survival

In epidemiology, survival analysis is about evaluation of variables that influence the survival of persons. Most likely at least age or disease duration, and possibly some exposure variables.

As it happens, the data that underlies survival analysis will normally be derived from a set of dates from registers, trials or cohort studies. For example date of diabetes, date of death and date of last follow-up (not all persons die in the study period).

So what we see in data is *how long* persons have been at risk of dying, and if they die. This is a bivariate outcome: (d, y) of event (death, $d \in \{0, 1\}$), and time at risk (y).

Some covariates, such as age and diabetes duration for example, show *when* persons have been at risk of dying.

1.2 Exercise 1: Diabetes data

Now take a look at the date set `DMlate`, a random sample of Danish diabetes patients diagnosed between 1995 and 2010 (data de-identified)

```
> data(DMlate)
> par(mfrow = 2:1)
> hist(DMlate$dodm, breaks = seq(1995, 2010, 1/4), col = 1)
> abline(v = 1995:2010, col = "red")
> hist(DMlate$dodm, breaks = seq(1995, 2010, 1/12), col = 1)
> abline(v = 1995:2010, col = "red")
```

What do you see from the monthly number of diagnoses?

For convenience we take a random subsample of 1000 patients:

```
> set.seed(19540803)
> DMlate <- DMlate[sample(1:nrow(DMlate), 1000), ]
> str(DMlate)
'data.frame':        1000 obs. of  7 variables:
 $ sex  : Factor w/ 2 levels "M","F": 2 1 1 2 1 1 1 2 1 2 ...
 $ dobth: num  1969 1960 1923 1922 1953 ...
 $ dodm : num  2003 2005 2007 1998 2008 ...
 $ dodth: num  NA NA NA 1999 2009 ...
 $ dooad: num  NA 2005 2007 NA NA ...
 $ doins: num  2003 NA NA NA NA ...
 $ dox  : num  2010 2010 2010 1999 2009 ...
```

The dates in `DMlate` are coded as fractional years, so for example 1 January 2005 is coded 2005.0 and 1 July 2007 is coded 2007.495. It's more convenient to have person-time in units of years, and located at an understandable place on the time axis.

1.3 Survival probability

If we want to see how diabetes patients survived after diagnosis of diabetes we need the survival time (risk time), `dox` − `dodm` and the indicator of whether a person died at exit, `!is.na(dodth)`.

From this we will want the *probability* that a person survives a given time t as a function of t, say:

$$S(t) = \mathrm{P}\{\text{alive at time } t\}$$

We can directly estimate the survival as a function of time since diagnosis of diabetes. Note that the term "survival" only makes sense if we define an *origin*, and we then measure the risk time as the time since that origin. So here we defined the origin as the date of diabetes diagnosis, and t means "time since the origin".

The survival function can be estimated by the Kaplan-Meier estimator using `survfit` from the `survival` package. Here we use the time at diagnosis of diabetes as the origin, so we are estimating the probability of surviving a given span of time after diagnosis of diabetes:

```
> sf <- survfit(Surv(dox - dodm, !is.na(dodth)) ~ 1, data = DMlate)
> sf
Call: survfit(formula = Surv(dox - dodm, !is.na(dodth)) ~ 1, data = DMlate)

        n events median 0.95LCL 0.95UCL
[1,] 1000    231     NA      NA      NA
> plot(sf, yaxs ="i")
> abline(h = 0.5, lty = 3)
```

CODE EXPLAINED: Survival data can be represented in a `Surv` object; `Surv` takes 3 arguments: start time, end time, event indicator. If start time is 0, the first argument can be omitted.

`survfit` produces an object that can be printed or plotted; the Kaplan-Meier estimator.

The curve is a step-curve, although it would seem a more natural assumption that $S(t)$ were a smooth curve.

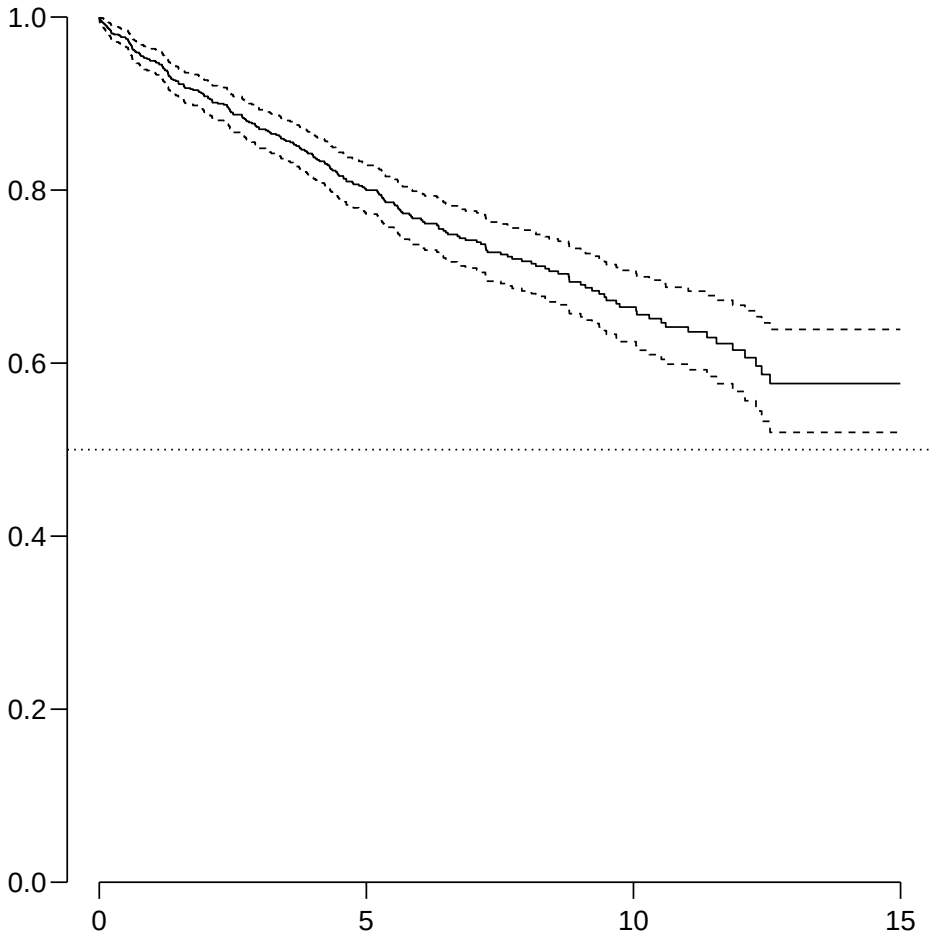

Figure 1.1: *Kaplan-Meier estimator of survival function for Danish diabetes patients.*
`../graph/surv0-KM`

1.4 Exercise 2: Mortality

The data we get from a register, cohort or clinical trial is not survival, but really *mortality data*: events (numerator, 0 or 1) and risk time (denominator).

So it would seem more natural to start with estimation of the mortality rate:

$$\lambda(t) = \lim_{h \to 0} \mathrm{P}\{\text{death in } (t, t+h) | \text{alive at } t\}/h$$

where we now have the liberty to take t to be either time since diagnosis, current age or even calendar time (pretty silly).

It is a bit more complicated to estimate this because it requires statistical modeling of the effect of t on the mortality rate. But let's begin with the simplest case:

1.4.1 Constant mortality

If we make the assumption that the mortality is constant, the estimator of the mortality $\lambda(t) = \lambda$ is just the ratio of the number of deaths to the amount of risk time:

```
> (mort <- with(DMlate, sum(!is.na(dodth)) / sum(dox - dodm)))
[1] 0.04279325
```

In this case mortality is measured in cases per year, because `dox` and `dodm` are measured in years. Its value is 4.3% per year. The *scale* or *units* of the variable is crucial.

The relationship between survival to t since some origin and the mortality rate is

$$S(t) = \exp\Big(-\int_0^t \lambda(u)\,\mathrm{d}u\Big)$$

which, if $\lambda(u) = \lambda$ becomes

$$S(t) = \exp(-\lambda t)$$

so we can put this on top of the Kaplan-Meier curve that we get from `plot(sf)`:

```
> plot(sf, yaxs = "i")
> abline(h = 0.5, col = "gray")
> t = seq(0, 15, 0.1)
> lines(t, exp(-mort * t), col = "red", lwd =2)
```

We see that it is an approximation to the KM estimator to assume constant mortality, but not a good one, so we obviously need to allow mortality to depend on time.

1.4.2 Exercise 3: Time-dependent mortality

Note that when we allow mortality to depend on time, $\lambda(t)$, we are using time (e.g. time since diagnosis) as an *explanatory* variable—a covariate.

1.4.2.1 Subdividing follow-up by time

Time dependent mortality can be modeled by dividing the timescale in small intervals, so that the assumption of constant mortality *within* each interval is necessary; we may then assume that mortality varies (smoothly) *between* the intervals as a function of the position of the intervals on the time scale.

A simple way of doing this is to use the `Lexis` machinery in the `Epi` package:

```
> Lx <- Lexis(exit = list(tfd = dox - dodm),
+      exit.status = factor(!is.na(dodth),
+                           labels = c("DM", "Dead")),
+             data = DMlate)
NOTE: entry.status has been set to "DM" for all.
NOTE: entry is assumed to be 0 on the tfd timescale.
```

CODE EXPLAINED: The `Lexis` function takes the input data set (here `DMlate`) and adds information in the form of a few variables and attributes. In the `Epi` package is a vignette explaining all the features in detail, try: `vignette(package = "Epi")`.

The `exit` argument defines the time of exit from the study on some time scale, in this case time from diabetes, `tfd`, assuming that entry is at 0 on this timescale.

`exit.status` defines the status of each person at exit, in this case `DM` if no date of death (`dodth`) is available otherwise `Dead`. The `labels=` refer to the order of the values of `!is.na(dodth)` where R defines `FALSE` < `TRUE`.

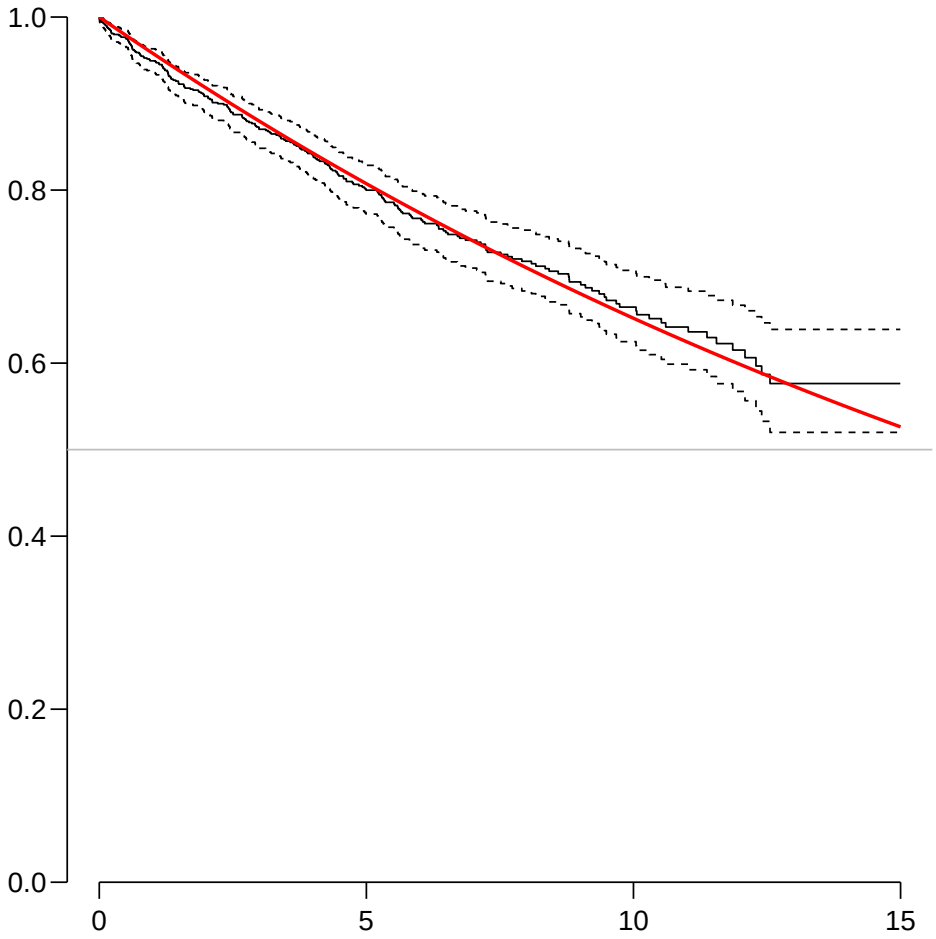

Figure 1.2: *Kaplan-Meier estimator of survival of Danish diabetes patients. The red curve is from the model with constant mortality, also called exponentially distributed survival times.*
../graph/surv0-KMexp

`Lexis` is talkative and tells about the assumptions made; you can do `?Lexis` to get the full set of arguments.

We can get a summary of the status and follow-up in the `Lexis` object:

```
> summary(Lx)
Transitions:
     To
From  DM Dead  Records:  Events: Risk time:  Persons:
  DM 769  231      1000      231    5398.05      1000
```

How many persons are in the dataset—and how many deaths and how much risk time on average?

We can print data for the 5 first persons:

```
> Lx[1:5, ]
 lex.id tfd lex.dur lex.Cst lex.Xst sex   dobth    dodm   dodth   dooad   doins     dox
      1   0    7.08      DM      DM   F 1969.22 2002.92      NA      NA 2003.01 2010.00
      2   0    4.79      DM      DM   M 1959.55 2005.21      NA 2005.22      NA 2010.00
```

```
      3   0    3.00      DM      DM   M 1923.07 2007.00      NA 2007.00      NA 2010.00
      4   0    1.05      DM    Dead   F 1921.69 1998.24 1999.29      NA      NA 1999.29
      5   0    0.89      DM    Dead   M 1952.51 2008.12 2009.01      NA      NA 2009.01
```

The follow-up is still with one record per person, the time at the beginning of the interval is `tfd = 0` for all persons, and the length of the follow-up intervals is in the variable `lex.dur`.

We can split the follow-up (time and events) in smaller intervals, say 0.5 years by using `splitLexis`:

```
> sL <- splitLexis(Lx, seq(0, 15, 0.5))
> subset(Lx, lex.id %in% c(2,4))
 lex.id tfd lex.dur lex.Cst lex.Xst sex   dobth    dodm   dodth   dooad doins     dox
      2   0    4.79      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      4   0    1.05      DM    Dead   F 1921.69 1998.24 1999.29      NA    NA 1999.29
> subset(sL, lex.id %in% c(2,4))
 lex.id tfd lex.dur lex.Cst lex.Xst sex   dobth    dodm   dodth   dooad doins     dox
      2 0.0    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 0.5    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 1.0    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 1.5    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 2.0    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 2.5    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 3.0    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 3.5    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 4.0    0.50      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      2 4.5    0.29      DM      DM   M 1959.55 2005.21      NA 2005.22    NA 2010.00
      4 0.0    0.50      DM      DM   F 1921.69 1998.24 1999.29      NA    NA 1999.29
      4 0.5    0.50      DM      DM   F 1921.69 1998.24 1999.29      NA    NA 1999.29
      4 1.0    0.05      DM    Dead   F 1921.69 1998.24 1999.29      NA    NA 1999.29
```

> CODE EXPLAINED: The result of `splitLexis` is a subdivision of the follow-up time in small intervals, where the `tfd` is now the time since diagnosis at the beginning of each interval. This is what we shall use as a quantitative (metric) variable in modeling mortality.

We see that we now have more records for each person, but the total follow-up time for each person (in `lex.dur`) is the same. Also the number of deaths (`lex.Xst=="Dead"`) is the same for each person (namely 1 for person 2, and 0 for person 4).

Try `summary()` of the split dataset and compare with the results for `Lx`.

1.4.2.2 Exercise 4: Modeling mortality

We can now model the mortality rates as a function of the timescale `tfd`, that is the time where the follow-up is. For each interval of (less than) a 0.5 years length we assume the mortality is constant, and we then model the size of this mortality as a smooth function of the value of `tfd` at the left endpoint of the interval. Because we are using `tfd` as a *quantitative* variable it is allowed to take any value on the `tfd` scale, not only those induced by the `splitLexis`. Modeling the effect of `tfd` as non-linear can be done by using a spline function, `Ns`:

```
> m0 <- glmLexis(sL, ~ Ns(tfd, knots = c(0,1,4,10)))
```

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

CODE EXPLAINED: `glmLexis` fits a models for the rates, using a Poisson likelihood. You are told what transition rates are modeled—the `Lexis` attributes are used to model occurrence rates (mortality that is) using the `poisreg` family. The `glmLexis` call is equivalent to:

```
> mr <- glm(cbind(lex.Xst == "Dead", lex.dur)
+           ~ Ns(tfd, knots = c(0,1,4,10)),
+           family = poisreg,
+             data = sL)
```

or—as people did in the old days—(note the `poisreg` / `poisson` family functions):

```
> mp <- glm(lex.Xst == "Dead"
+           ~ Ns(tfd, knots = c(0,1,4,10)),
+           offset = log(lex.dur),
+           family = poisson,
+             data = sL)
```

This is the point of setting data up as a `Lexis` object: subsequent data handling (`splitLexis`) and model specification (`glmLexis`) becomes much simpler.

Furthermore, the `poisreg` family has the possibility to use an identity link function, leading to additive hazards (rate) models instead of multiplicative rate models (proportional hazards)—there is a `link=` argument to `glmLexis`. The `poisson` family does not have this possibility.

We now have fitted a fully parametric model for the mortality rates as a function of `tfd`, so we can make a *prediction* of the mortality rate at any set of values of `tfd`. `ci.pred` also produces confidence intervals:

```
> nd <- data.frame(tfd = seq(0, 15, 0.1))
> pm <- ci.pred(m0, nd)
> head(cbind(tfd = nd$tfd, pm))
  tfd   Estimate       2.5%      97.5%
1 0.0 0.05454378 0.03944859 0.07541521
2 0.1 0.05359010 0.03985352 0.07206135
3 0.2 0.05265756 0.04018047 0.06900913
4 0.3 0.05175006 0.04040389 0.06628245
5 0.4 0.05087118 0.04049765 0.06390190
6 0.5 0.05002424 0.04043904 0.06188139
```

CODE EXPLAINED: `nd` is a prediction data frame, each row with combination of covariate values for which we want a mortality prediction. In this case there is only one variable in the model, `tfd`. The last statement illustrates the value of `pm` as a function of `tfd`—a 3-column matrix of mortality and confidence intervals.

These are mortality rates per 1 year (because `lex.dur` is in years), but we would want them in units per 100 person-years (% per year), so we multiply by 100 when plotting:

```
> matshade(nd$tfd, pm * 100, plot = TRUE, lwd = 2,
+          ylim = c(0, 10), yaxs = "i",
+          xlab = "Time since diabetes (years)",
+          ylab = "Mortality rate (% / year)")
> abline(v = c(0,1,4,10), lty = 3)
```

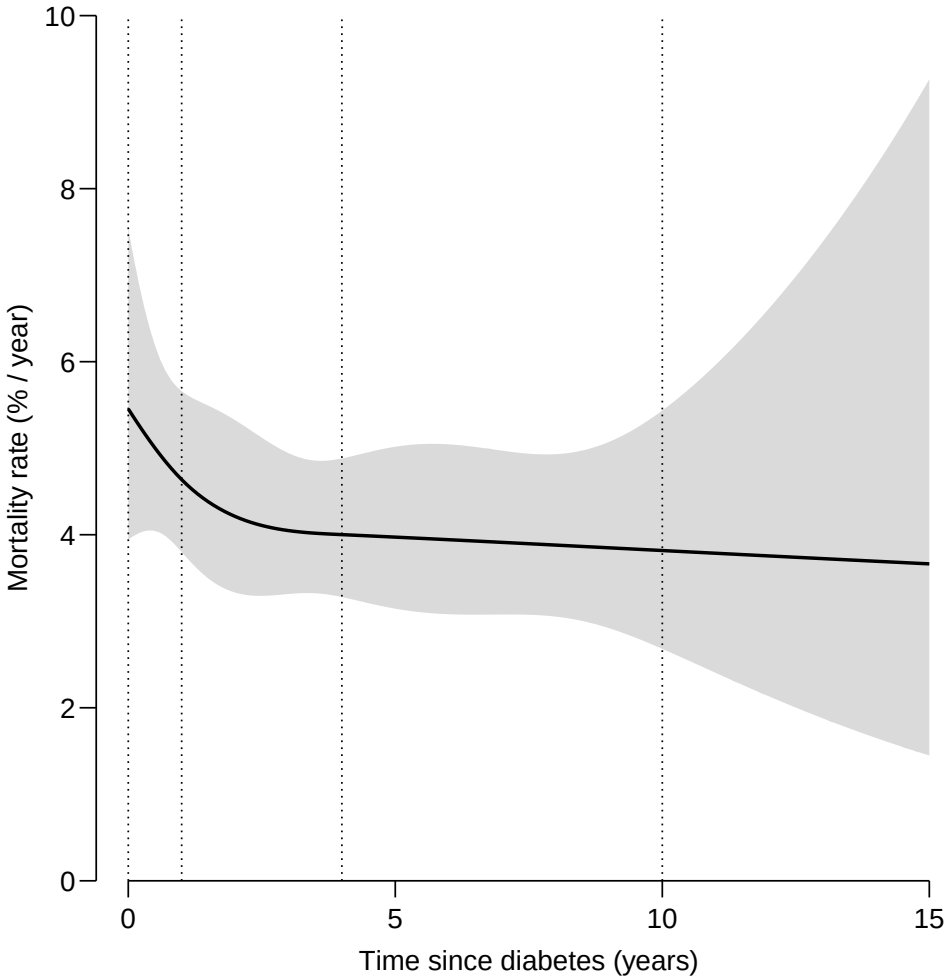

Figure 1.3: *Spline model for the effect of diabetes duration on mortality among Danish diabetes patients. The dotted lines indicate the location of the knots—in the three first intervals the curve is a 3rd degree polynomial, beyond the last it is a straight line.*

../graph/surv0-mort

From figure 1.3 it appears that mortality is declining the first few years after diagnosis and then is approximately constant—how is that, people get older as time goes by?

Here is the mean age at different times since diagnosis, and the successive differences:

```
> am <- with(sL, tapply((dodm - dobth) + tfd, floor(tfd), mean))
> cbind(am, c(NA, diff(am)))
         am
0  60.83603          NA
1  60.94051  0.10447195
2  61.35016  0.40965543
```

```
3  61.50191  0.15174811
4  61.85418  0.35227263
5  61.87411  0.01992934
6  62.02486  0.15074364
7  62.58449  0.55963808
8  62.98958  0.40508265
9  63.60464  0.61506736
10 64.94027  1.33563101
11 65.10760  0.16732416
12 65.68808  0.58047802
13 64.09349 -1.59458354
14 62.56480 -1.52869752
```

CODE EXPLAINED: The age at diagnosis is `dodm - dobth` so when adding `tfd` to this we get the age at follow-up at the beginning of each interval. `tapply` computes the *mean* age for integer values of `tfd`. The function `diff` takes the successive differences of a vector.

We see that the mean age increases by `tfd`, but far from one year per year, so we should not expect the effect of `tfd` to increase as the effect of age. This is a typical for epidemiological studies; in this case we have a massive confounding by age.

1.4.2.3 Exercise 5: Parametric survival function

We would also want to see the transformation to the survival function, which is straight-forward by numerical integration. Confidence limits for the survival function is however not so easy, but there is fortunately a function that does this, `ci.surv`:

```
> plot(sf, yaxs = "i", ylim = c(0.5, 1),
+          xlab = "Time since diabetes (years)",
+          ylab = "Survival probability")
> abline(h = 0.5)
> ps <- ci.surv(m0, nd)
NOTE: interval length chosen from  as tfd[2] - tfd[1]
> matshade(nd$tfd, ps, col = "red", lwd = 2)
```

We now see a much closer agreement between the K-M and the parametric curves, but also a biologically more credible curve.

1.4.3 Relationship between mortality and survival

There is not a one-to-one relationship between mortality and survival; there is a one-to-one relationship between mortality *and* an origin on one hand and the survival on the other. Survival is survival *since* some origin

The parametric mortality and survival curves are obtained by:

- define follow-up as `Lexis`
- split follow-up in small intervals with `splitLexis`
- use `glmLexis` to model mortality as function of time
- use `ci.pred` to get mortality rates—not available using K-M
- use `ci.surv` to get survival function—close to K-M, but biologically credible (continuous smooth function)

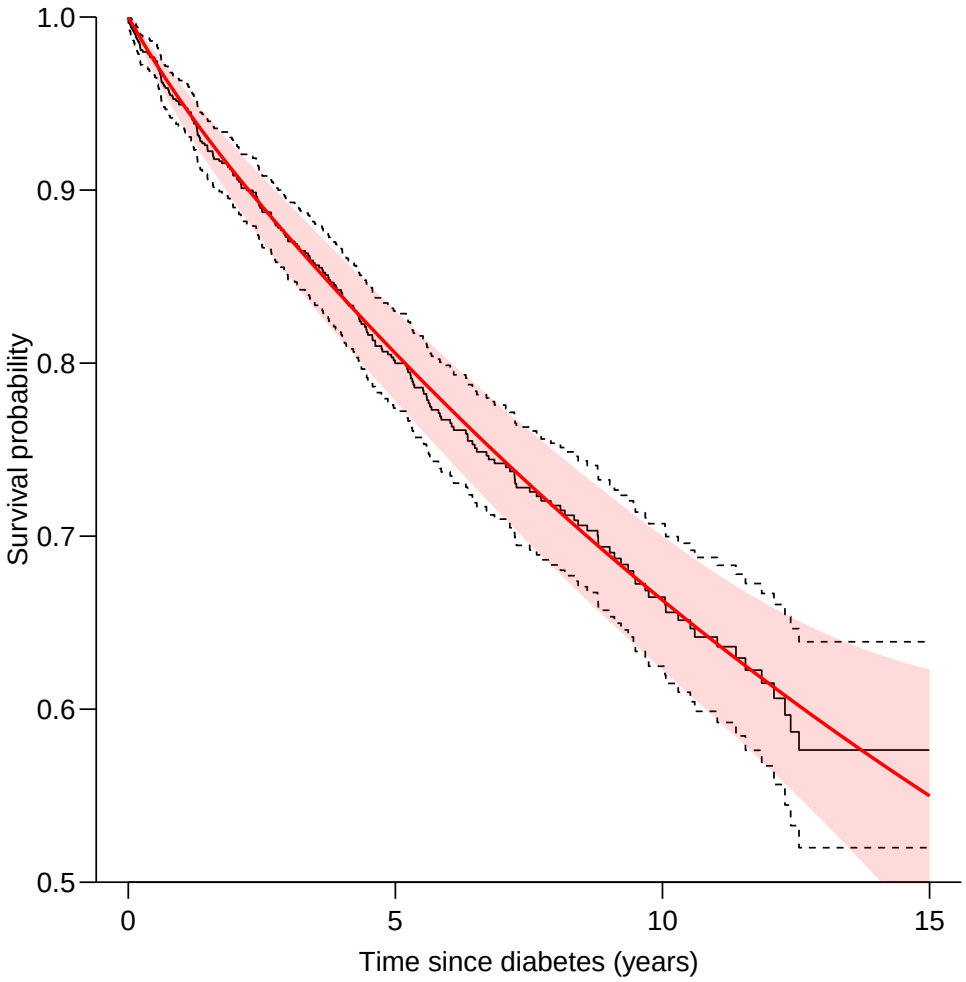

Figure 1.4: *Kaplan-Meier estimator of (black) overlaid with a parametric estimator based on a spline model for mortality (red). Note that the y-axis is not from 0.*

../graph/surv0-KMmod

The Kaplan-Meier curve is obtained by:

- define follow-up using `Surv()`
- use `survfit` get survival function
- use `plot.survfit` to plot the curve

. . . only gives the survival function, not the mortality function.

```
  Start time: 2025-05-10, 16:09:31
    End time: 2025-05-10, 16:09:32
Elapsed time: 0.01 minutes
```

Chapter 2

Cause specific rates and competing risks

. . . now input from `comp.tex`

Technicalities

First we set some output and graphics parameters for convenience and load the packages needed:

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

```
> library(Epi)
> library(popEpi)
> library(survival)
> library(tidyverse)
> library(mgcv)
> clear()
```

```
     R   Epi   popEpi
 4.5.0  2.59   0.4.13
```

This exercise is largely taken from the vignette "Competing risks with `Lexis`, parametric rates and simulation based confidence intervals" which is available in the `Epi` package as (pops up as a `.pdf`):

```
> vignette("crisk", package = "Epi")
```

—you can see all vignettes in the `Epi` package here:

```
> vignette(package = "Epi")
```

or you can access it directly from CRAN:
https://cran.r-project.org/web/packages/Epi/vignettes/crisk.pdf

2.1 Concepts

The concept of competing risks is one where persons in a given state, "alive", say, are subject to a number of different causes of death, "cause1", "cause2" etc. Causes of death are required to be exhaustive and mutually exclusive. That is, you will eventually die from one of the designated causes, and you can only die from one.

The *cause-specific* rate for cause c is defined as:

$$\lambda_c(t) = \mathrm{P}\{\text{death from cause } c \text{ in } (t, t+h] \mid \text{ alive at } t\}/h$$

... formally, the limit of this quantity as $h \to 0$. This is to exclude that no two transitions occur in the same interval with positive probability.

The observed data will be a survival time and an exit status which is either "censored alive" or one of the causes. In situations where the causes are not causes of death but other events, it is implicit that we only consider the first occurrence of an event from the state "alive", and ignore whatever occurs after that.

Two key papers by Per Kragh Andersen (and others) [1, 2] provide an overview of the concepts in competing risks and multistate models. Examples in those are mostly based on semi-parametric models, where the treatise here is based on fully parametric models.

2.1.1 Cause specific rates and likelihood

The likelihood for observations from a competing risk scenario is a function of the cause-specific transition rates, and is a *product* of the likelihoods that would emerge if we considered each cause as being the only possible event. Thus analysis is in principle straight forward: estimate a model for each of the cause-specific rates; these will together form a complete model for the competing risks problem.

If the cause-specific rates are all we want to assess then we are done.

2.1.2 Survival and cumulative risks

In addition to the rates we might however also be interested in the *survival* probability and the *cumulative risks* of each cause of death.

The survival is the probability of still being alive at a given time after entry—a function of time since entry. The cumulative risk of dying from cause c is the probability of having died from cause c as a function of time since entry.

This means that a time of entry (time origin) is required for calculation of these quantities.

2.1.3 Sojourn times

The *sojourn time* for cause c is the time spent in the "cause c" state before a given time, t, say. This is also called the expected lifetime lost to cause c, *truncated* (or restricted) at the time t. For the state "alive" it will be the expected time lived before t. This is also called the restricted mean survival time, RMST.

2.1.4 The time scale

The cause specific rates will be functions of covariates, notably a time scale, be that age or time since entry to the study or even calendar time. But the cumulative risks are probabilities that refer to time since some *origin*. Thus cumulative risks (and survival) are only meaningful in relation to a time that begins at 0 (time since the origin).

If we were to use age as timescale for cumulative risk, we would want data available since birth; if we only had observations where most people entered between 20 and 40 years of age, we could mathematically compute cumulative risk from birth to some age, but it would be nonsense. Instead we would compute the cumulative risk *given* that a person attained age 40, say. In that case the time scale would be age $-$ 40. Note that the origin in this context is taken out of thin air.

2.2 Rates and cumulative risks

Each of the cumulative risks is a function of the cause-specific rate *and* the survival function, which in turn depends on *all* cause-specific rates. If the cause-specific rates are $\lambda_c(t)$, $c = 1, 2, \ldots$, then the survival function (probability of being alive at time t) is:

$$S(t) = \exp\left(-\int_0^t \sum_c \lambda_c(s)\,\mathrm{d}s\right) = \exp\left(-\sum_c \Lambda_c(t)\right) \tag{2.1}$$

The quantities $\Lambda_c(t) = \int_0^t \lambda_c(s)\,\mathrm{d}s$ are called cumulative *rates* (probabilists call them integrated intensities), although they are not rates. Cumulative rates are dimensionless, but they have no probability interpretation of any kind.

The cumulative *risk*, the probability of dying from cause k, say, before time t, $R_k(t)$ is:

$$R_k(t) = \int_{u=0}^{u=t} \lambda_k(u) S(u)\,\mathrm{d}u = \int_{u=0}^{u=t} \lambda_k(u) \exp\left(-\sum_j \Lambda_j(u)\right)\mathrm{d}u \tag{2.2}$$

The rationale is that $\lambda_k(u)\,\mathrm{d}u$ is the probability of death from cause k in the small interval $(u, u + \mathrm{d}u)$, *conditional* on being alive at time u. If this is multiplied with the probability of being alive at u, $S(u)$, Bayes rule tells us that $\lambda_k(u)S(u)\,\mathrm{d}u$ is the *marginal probability* of death from cause k in the small interval $(u, u + \mathrm{d}u)$. This function of u is the argument in the integral; so integration from $u = 0$ to $u = t$, will give the probability of death from cause c anywhere in $(0, t)$—the cumulative risk of dying from cause k at t.

Parametric models for the cause-specific rates can produce estimated transition rates λ_c at closely spaced times, and the cumulative risks can then be estimated from these by simple numerical integration; this is illustrated in the next chapter.

Note that at any one time every person is either alive or dead from one of the causes, so the sum of the survival and the cumulative risks is always 1:

$$1 = S(t) + \sum_j R_j(t), \quad \forall t \tag{2.3}$$

Methods that estimate the cumulative risks should produce estimates that meet this equation.

2.2.1 Confidence intervals by simulation

Even if we from the modeling of the λ_cs may have standard errors of $\log\big(\lambda_c(t)\big)$, the standard errors of the R_cs will be analytically intractable from these. Well, not quite so, but extremely complicated. So we resort to:

2.2.1.1 Parametric bootstrap

which is a method to simulate values of a function of parameter values. Assuming that the parameter estimates in a model follow a multivariate normal distribution with mean equal to the estimated parameters and variance-covariance equal to the estimated variance-covariance of the parameters, we can simulate a large number of parameter vectors from that distribution (sometimes called the posterior distribution of the parameters). All that is needed is an R function that can produce a random sample from the multivariate normal distribution.

From each simulated set of parameters we then compute the desired function, and so we have a sample from the distribution of the desired function. A 95% confidence interval is then simply the empirical 2.5% and 97.5% quantiles in that distribution.

The attractive feature of this method is that we only need to be able to compute the function from the parameters.

2.2.1.2 Confidence intervals

In practice, the only viable way to get confidence intervals for the cumulative risks, R_c, is therefore by calculation of a large set of rates $\lambda_c(t)$ by sampling from the posterior distribution of the parameters in the models for $\log\big(\lambda_c(t)\big)$, and then compute the $R_c(t)$s by numeric integration for each simulated sample, according to formulae 2.1 and 2.2. This will produce a parametric bootstrap sample of the cumulative risks from which we can derive confidence intervals.

The simulation approach also allows calculation of confidence intervals for sums of the cumulative risks, $R_1(t) + R_2(t)$, for example, which will be needed if we want to show stacked cumulative risks.

Finally, it will also allow calculation of standard errors of sojourn times in each of the states "alive" and "cause1", "cause2",.... While the latter two may not be of direct interest, then *differences* between such sojourn times between different groups can be interpreted as years of life lost to each cause between groups.

2.3 Exercise 6: Example data — uptake of insulin or OAD

We use the `DMlate` dataset of Danish diabetes patients where we restrict to the outcomes `OAD` or `Ins` with `Dead` as competing event.

2.3.1 A `Lexis` object

```
> data(DMlate)
> dl <- mutate(DMlate, dofin = pmin(dodth, dooad, doins, dox, na.rm = TRUE),
```

```
+                       xstat = factor(case_when(dofin == dodth ~ "Dead",
+                                                dofin == dooad ~ "OAD",
+                                                dofin == doins ~ "Ins",
+                                                          TRUE ~ "DM"),
+                                      levels = c("DM", "OAD", "Ins", "Dead")))
> str(dl)
'data.frame':        10000 obs. of  9 variables:
 $ sex  : Factor w/ 2 levels "M","F": 2 1 2 2 1 2 1 1 2 1 ...
 $ dobth: num  1940 1939 1918 1965 1933 ...
 $ dodm : num  1999 2003 2005 2009 2009 ...
 $ dodth: num  NA NA NA NA NA ...
 $ dooad: num  NA 2007 NA NA NA ...
 $ doins: num  NA NA NA NA NA NA NA NA NA NA ...
 $ dox  : num  2010 2010 2010 2010 2010 ...
 $ dofin: num  2010 2007 2010 2010 2010 ...
 $ xstat: Factor w/ 4 levels "DM","OAD","Ins",..: 1 2 1 1 1 4 2 2 4 1 ...
> dL <- Lexis(exit = list(tfd = dofin - dodm),
+      exit.status = xstat,
+             data = dl)
NOTE: entry.status has been set to "DM" for all.
NOTE: entry is assumed to be 0 on the tfd timescale.
NOTE: Dropping  2468  rows with duration of follow up < tol
> summary(dL)
Transitions:
     To
From   DM  OAD Ins Dead  Records:  Events: Risk time:  Persons:
  DM 2829 2966 681 1056      7532     4703   22920.27      7532
```

CODE EXPLAINED: In `mutate`, we define the `fin`ishing date as the earliest of the three events or exit (`dox`), and use `case_when` to define the type of event.

The `dL` is defined as a `Lexis` object with three types of event and `DM` as the starting state—hence the `levels=` that defines the ordering of the factor levels; `DM` must be the first of the levels. We only include the follow-up in state `DM`; the dropped persons are those that start with `dodm` equal to either `dooad` or `doins`.

We split the follow-up in the state `DM` (that is before drug inception) in intervals of 1/12 year (months), creating a `Lexis` object for analysis of the three cause specific rates:

```
> Sdm <- splitLexis(dL, time.scale = "tfd", breaks = seq(0, 20, 1/12))
> summary(Sdm)
Transitions:
     To
From     DM  OAD Ins Dead  Records:  Events: Risk time:  Persons:
  DM 274262 2966 681 1056    278965     4703   22920.27      7532
```

We can illustrate the follow-up in state boxes (this would be the same if we used `dL`):

```
> boxes(Relevel(dL, c(1, 4, 2, 3)),
+       boxpos  = list(x = c(15, 85, 75, 15),
+                      y = c(85, 85, 30, 15)),
+       scale.R = 100,
+       show.BE = TRUE )
```

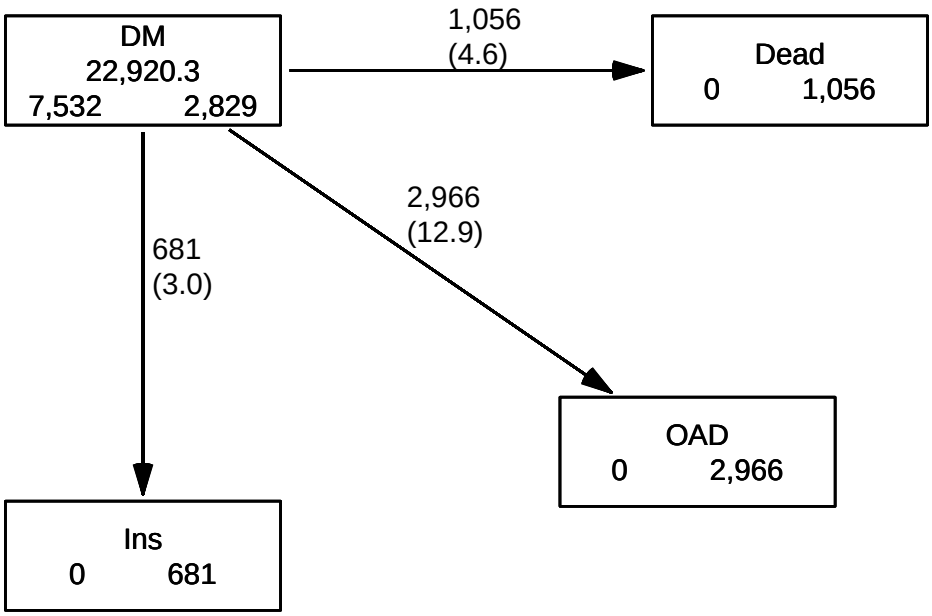

Figure 2.1: *The transitions in the competing risks model, where follow-up is stopped at first drug exposure. By that token only the* `DM` *state has person-years; a characteristic of a competing risks situation.*

`../graph/comp-boxes4`

2.4 Exercise 7: Models for rates

Now that we have set up a dataset with three competing events, we can model the cause-specific rates separately by time from diagnosis as the only underlying time scale.

This is done by Poisson-regression on the time-split data set; since the dataset is in `Lexis` format we can use the convenience wrapper `gamLexis` to model rates as smooth functions of time (`tfd`). Note that we only need to specify the `to=` argument because there is only one possible `from` for each `to` (incidentally the same for all `to` states, namely `DM`):

```
> system.time(
+ mD <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'Dead'))
> mO <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'OAD' )
> mI <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'Ins' )
```

We could alternatively use a predefined parametric form of the time effect—using `glmLexis` also saves a bit of time:

```
> system.time(
+ mD <- glmLexis(Sdm, ~ Ns(tfd, knots = c(0, 1, 3, 6)), to = 'Dead'))
stats::glm Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->Dead

  bruger   system forløbet
    1.67     0.31     2.18
```

```
> mO <- glmLexis(Sdm, ~ Ns(tfd, knots = c(0, 1, 3, 6)), to = 'OAD' )
stats::glm Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->OAD
> mI <- glmLexis(Sdm, ~ Ns(tfd, knots = c(0, 1, 3, 6)), to = 'Ins' )
stats::glm Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->Ins
```

We see that `gamLexis` and `glmLexis` tells us what transition rates are modeled.

With these models fitted we can compute the rates, and from these cumulative rates and the cumulative risks and sojourn times in states using the usual formulae.

First we compute the rates in intervals of length 1/20 years. Note that these prediction points are unrelated to the follow-up intervals in which we split the observed data for analysis—they were 1 month intervals (1/12 year), here we use 1/20 year.

```
> nd <- data.frame(tfd = seq(0, 10, 1/20))
> rownames(nd) <- nd$tfd
> str(nd)
'data.frame':        201 obs. of  1 variable:
 $ tfd: num  0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 ...
> head(nd)
      tfd
0    0.00
0.05 0.05
0.1  0.10
0.15 0.15
0.2  0.20
0.25 0.25
```

With `nd` as prediction data frame we can show the rates as a function of the time since entry (diagnosis of diabetes):

```
> matshade(nd$tfd, cbind(ci.pred(mD, nd),
+                        ci.pred(mI, nd),
+                        ci.pred(mO, nd)) * 1000,
+          col = c("black", "red", "blue"), log = "y", lwd = 3, plot = TRUE,
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Rates per 1000 PY", ylim = c(0.05, 500), yaxt = "n")
> axis(side = 2, at = ll <- outer(c(1,2,5), -2:3, function(x,y) x * 10^y),
+               labels = formatC(ll, digits = 4), las = 1)
> axis(side = 2, at = outer(c(1.5,2:9), -2:3, function(x,y) x * 10^y),
+               labels = NA, tcl = -0.3)
> text(0, 0.5*0.6^c(1,2,0),
+      c("Dead","Ins","OAD"),
+      col = c("black","red","blue"), adj = 0, font = 2)
```

Note that the graph in figure 2.2 is not normally shown in analyses of competing risks; the competing cause-specific *rates* are hardly ever shown. I suspect that this is because they are often modeled by a Cox model and so are buried in the model and hard to get at.

Since we shall compute integrals of the rates, it would be relevant to show the rates on a linear scale instead, de-emphasizing the very small fluctuations of the `Ins` rates that are over-emphasized when using a log-scale for the y-axis.

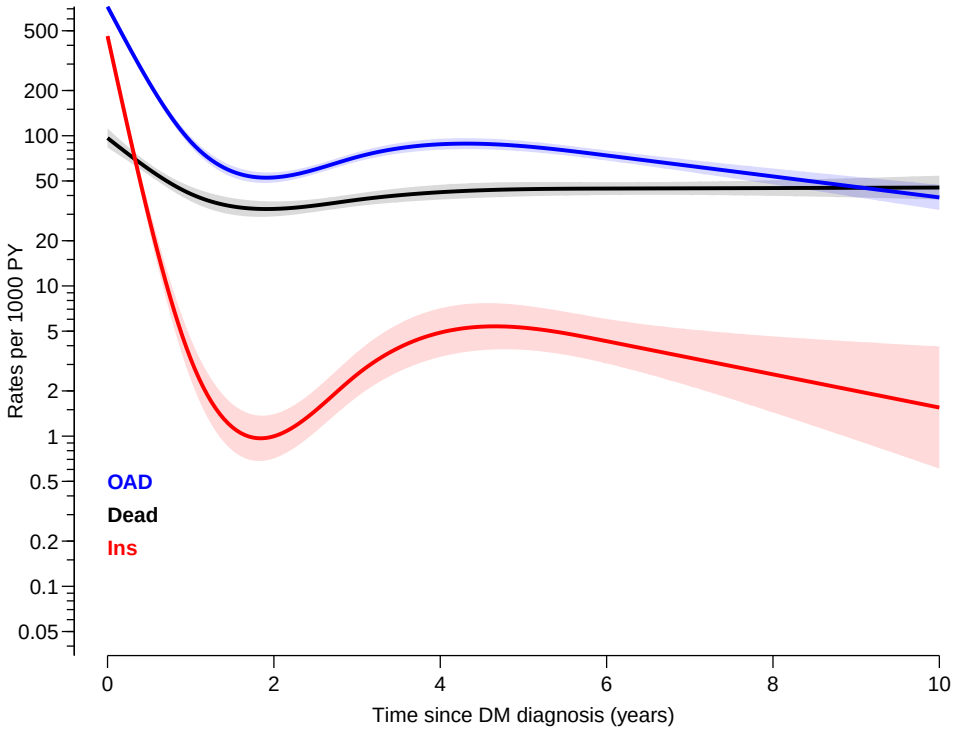

Figure 2.2: *Estimated rates from the* `DM` *state, estimates are from* `gam` *models fitted to data split in 1 month intervals (1/12 year, that is). Rates of* `OAD` *is in the vicinity of 0.1/year, and mortality about half of this. Rates of insulin start among persons on no other drug are beginning high, then decreasing with a nadir at about 4 years and then increase to a peak at 8 years. But note that the rates of* `Ins` *are very small, so the log-axis is a bit deceptive.*

`../graph/comp-rates`

```
> matshade(nd$tfd, cbind(ci.pred(mD, nd),
+                        ci.pred(mI, nd),
+                        ci.pred(mO, nd)) * 1000,
+          col = c("black", "red", "blue"), lwd = 3, plot = TRUE,
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Rates per 1000 PY", ylim = c(0, 500), yaxs = "i")
> text(8, 500 - c(2, 3, 1) * 20,
+      c("Dead","Ins","OAD"),
+      col = c("black","red","blue"), adj = 0, font = 2)
```

2.5 Exercise 8: Cumulative rates and risks

For the calculation of the cumulative rates and state probabilities, we need just the estimated rates (without CIs). The formulae 2.1 and 2.2 on page 15 are transformed to R-code; starting with the rates, λ_D as `lD` etc:

```
> # utility function to compute midpoints between sucessive values in a vector
> mp <- function(x) x[-1] - diff(x) / 2
> #
> int <- 1 / 20
```

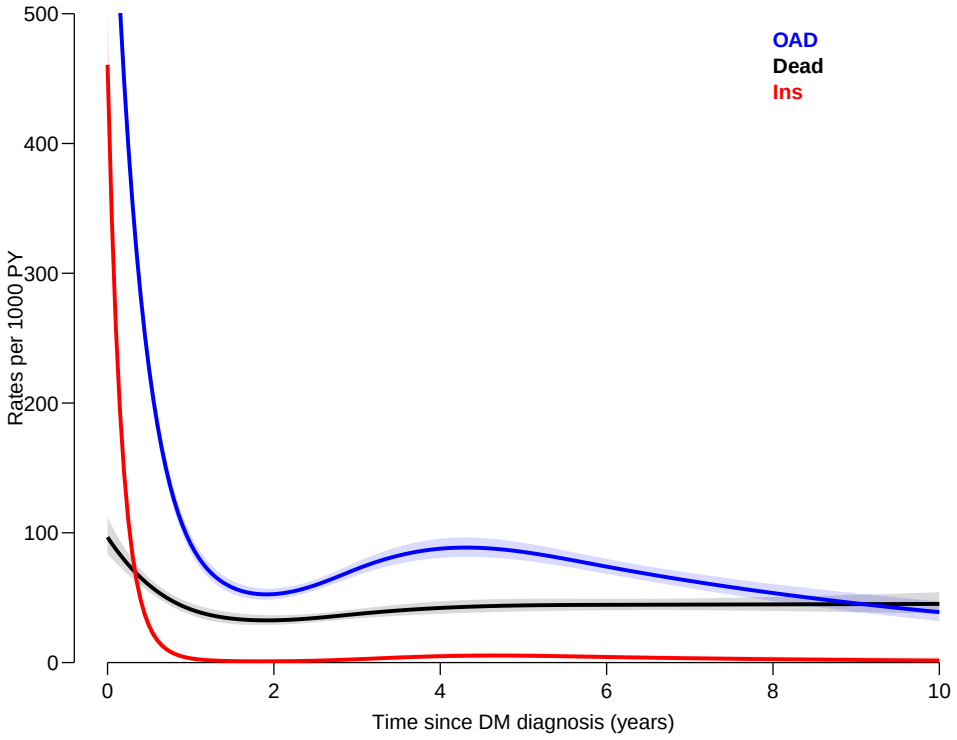

Figure 2.3: *Estimated rates from the* DM *state, estimates are from* gam *models fitted to data split in 1 month intervals (1/12 year, that is). Rates of* OAD *is in the vicinity of 0.1/year, and mortality about half of this.*

../graph/comp-rates-1

```
> # rates at midpoints of intervals
> lD <- mp(ci.pred(mD, nd)[, 1])
> lI <- mp(ci.pred(mI, nd)[, 1])
> lO <- mp(ci.pred(mO, nd)[, 1])
> #
> # cumulative rates and survival function at right border of the intervals
> LD <- cumsum(lD) * int
> LI <- cumsum(lI) * int
> LO <- cumsum(lO) * int
> # survival function, formula (1.1)
> Sv <- exp(- LD - LI - LO)
> #
> # when integrating to get the cumulative *risks* we use the average
> # of the survival function at the two endpoints
> # (adding 1 as the first), formula (1.2)
> Sv <- c(1, Sv)
> rD <- c(0, cumsum(lD * mp(Sv)) * int)
> rI <- c(0, cumsum(lI * mp(Sv)) * int)
> rO <- c(0, cumsum(lO * mp(Sv)) * int)
```

Now we have the cumulative risks for the three causes and the survival, computed at the end of each of the intervals. At any time point the sum of the 3 cumulative risks and the survival should be 1 (see equation 2.3, p. 15:

```
> summary(rD + rI + rO + Sv)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1       1       1       1       1       1
> oo <- options(digits = 20)
> cbind(summary(Sv + rD + rI + rO))
                          [,1]
Min.    1.0000000000000000000
1st Qu. 1.0000414528256968971
Median  1.0000419578588912728
Mean    1.0000413283034843559
3rd Qu. 1.0000422868418281652
Max.    1.0000424106033540816
> options(oo)
```

. . . and bar a small rounding error, they are.

We can then plot the 3 cumulative risk functions stacked together using `mat2pol` (`mat`rix to `pol`ygons):

```
> zz <- mat2pol(cbind(rD, rI, rO, Sv), x = nd$tfd,
+               xlim = c(0,10), xaxs = "i", yaxs = "i", las = 1,
+               xlab = "Time since DM diagnosis (years)",
+               ylab = "Probability",
+                col =  c("black","red","blue","forestgreen"))
> text(9, mp(zz["9", ]), c("Dead", "Ins", "OAD"," DM"), col = "white")
> box(col = "white", lwd = 3)
```

2.6 Exercise 9: Sojourn times

The sojourn times in each of the states is just the area of each of the colored parts of figure 2.4. Since the y-dimension of the plot is probability (dimensionless) and the x-axis has dimension time, the computed areas will have dimension time.

Normally we will not report the sojourn times as functions of (truncation) time, but only report them at a few select truncation points, such as 5 or 10 years. Calculation of the 10 year sojourn times would be straight-forward as integrals from 0 to 10—these calculations rely on the predicted rates from `nd` being for the first 10 years:

```
> Sj <- c(sjA = sum(Sv * int),
+         sjD = sum(rD * int),
+         sjI = sum(rI * int),
+         sjO = sum(rO * int))
> c(Sj, sum(Sj))
       sjA        sjD        sjI        sjO
 4.4747073  1.2272756  0.7551487  3.5932837 10.0504153
```

We see that there is a some rounding error in the calculations; the sum should really be exactly 10.

That was a demonstration of how to compute the rates, cumulative risks and sojourn times using simple numerical integration. But no confidence intervals for the computed quantities.

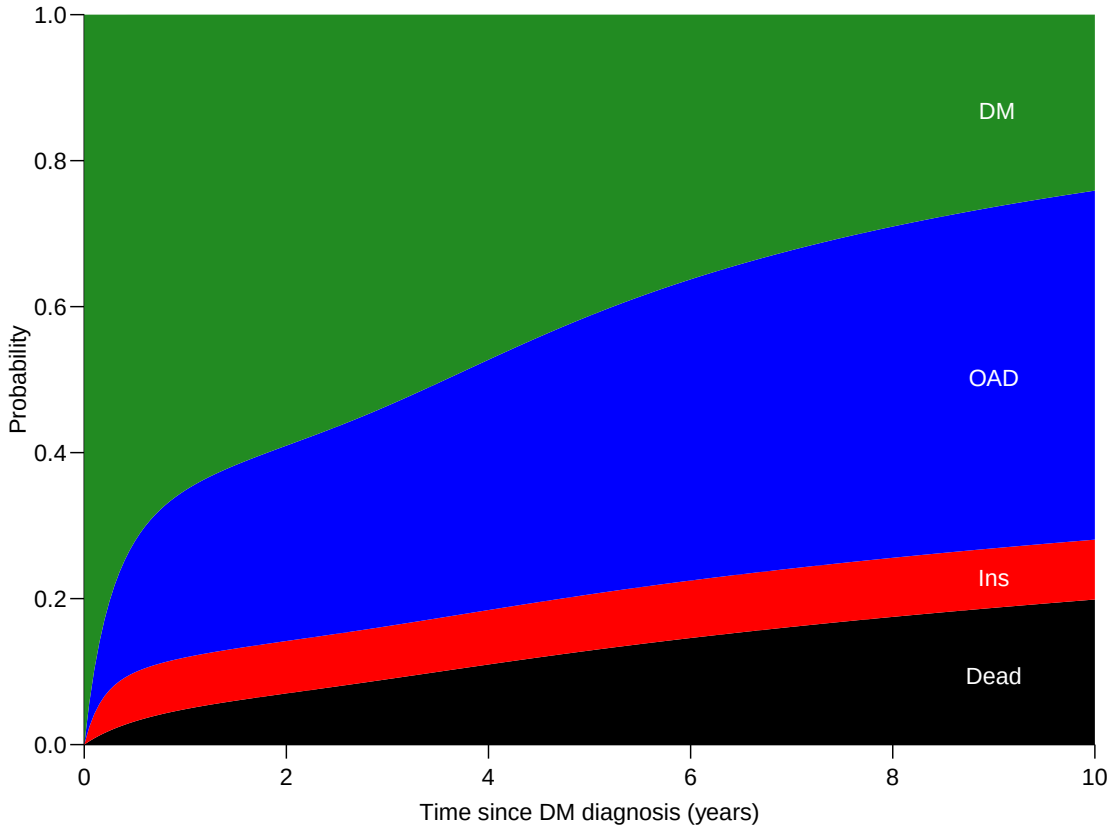

Figure 2.4: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`*. We are not concerned about what occurs after any these events.* `Dead` *means dead without initiating any of the two drugs.*

`../graph/comp-stack`

Chapter 3

Confidence intervals for cumulative risks

Besides confidence intervals for each of the 4 cumulative risks, we will also be interested in confidence intervals for *sums* of any subset of the cumulative risks, corresponding to the borders between the colors in figure 2.4. If we only had two competing risks (and hence three states) the latter would not be an issue, because the sum of any two cumulative risks will be 1 minus the cumulative risk of the remainder, so we could get away with the confidence intervals for the single cumulative risks. This is the reason we have chosen an example with 3 competing risks and not just 2; we then have 4 probabilities to sum in different order.

A short look at the formulae for cumulative risks will reveal that analytic approximation to the standard error of these probabilities (or some transform of them) is not really a viable way to go. Particularly if we also want confidence intervals of sums of the state probabilities as those shown in stacked plots.

So in practice, if we want confidence intervals not only for the state probabilities, but also for any sum of subsets of them we would want a large number of simulated copies of the cumulative risks, each copy being of the same structure as the one we just extracted from the models.

Confidence intervals for sojourn times (i.e. time spent) in each state up to a given time, would come almost for free from the simulation approach, by taking the relevant quantiles of the sojourn times computed from the simulated parameters.

This means that we must devise a method to make a prediction not from the estimated model, but where we instead of the model parameters use a sample from the posterior distribution of the estimated parameters. Here, the posterior distribution of the parameters will be taken to be the multivariate normal distribution with mean equal to the vector of parameter estimates and variance-covariance matrix equal to the estimated variance-covariance matrix of the parameters.

Precisely this approach is implemented in `ci.lin` via the `sample` argument; we can get a predicted value from a given prediction data frame just as from `ci.pred` resp. `ci.exp`; here is shown two different ways of getting predicted values of the cause-specific rates:

```
> head(cbind(ci.pred(mI, nd),
+            ci.exp (mI, nd)))
      Estimate       2.5%     97.5% exp(Est.)       2.5%     97.5%
0    0.4606488 0.41862132 0.5068956 0.4606488 0.41862132 0.5068956
0.05 0.3446838 0.31671741 0.3751196 0.3446838 0.31671741 0.3751196
0.1  0.2580771 0.23857940 0.2791681 0.2580771 0.23857940 0.2791681
```

```
0.15 0.1934787 0.17885751 0.2092952 0.1934787 0.17885751 0.2092952
0.2  0.1453282 0.13354631 0.1581495 0.1453282 0.13354631 0.1581495
0.25 0.1094403 0.09950469 0.1203679 0.1094403 0.09950469 0.1203679
```

Here is an illustration of the prediction with model based confidence intervals for the rates of insulin start (model `mI`), alongside predictions based on samples from the posterior distribution of the parameters in the model:

```
> str(ci.lin(mI, nd, sample = 4))
 num [1:201, 1:4] -0.763 -1.042 -1.32 -1.598 -1.873 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ : NULL
> head(cbind(ci.pred(mI, nd), exp(ci.lin(mI, nd, sample = 4))))
      Estimate       2.5%     97.5%
0    0.4606488 0.41862132 0.5068956 0.4433666 0.4788047 0.4816796 0.4513492
0.05 0.3446838 0.31671741 0.3751196 0.3348991 0.3587378 0.3573160 0.3355503
0.1  0.2580771 0.23857940 0.2791681 0.2531191 0.2689508 0.2652355 0.2496305
0.15 0.1934787 0.17885751 0.2092952 0.1915385 0.2018936 0.1971427 0.1859638
0.2  0.1453282 0.13354631 0.1581495 0.1452002 0.1518459 0.1468199 0.1388177
0.25 0.1094403 0.09950469 0.1203679 0.1103364 0.1144963 0.1096299 0.1039064
```

Note that we use `exp(ci.lin(...`—this is because the `sample=` argument does not work with `ci.exp`.

The simulation (parametric bootstrapping) is taking place at the parameter level and the transformation to survival and cumulative risks is simply by a function applied to each simulated set of rates.

3.1 Common parameters across cause-specific rates

Note that we have implicitly been assuming that the transitions are being modeled separately. If some transitions are modeled jointly—for example assuming that the rates of `OAD` and `Ins` are proportional as functions of time since entry, we are in trouble, because we then need one sample from the posterior generating two different predictions, one for each of the transitions modeled together. Moreover the model will have to be a model fitted to a `stack.Lexis` object, so a little more complicated to work with.

A simple way to program this would be to reset the seed to the same value before simulating with different values of `nd`, this is what is intended to be implemented, but is not yet. This is mainly the complication of having different prediction frames for different risks in this case.

However, this is not a very urgent need, since the situation where you want common parameters for different rates out of a common state is quite rare. It would for example be quite odd to assume the the M/W rate ratio were the same across different causes of death. By that token the facility is not likely to be implemented anytime soon, if ever.

3.2 Simulation based confidence intervals

The parametric bootstrap is implemented in the function `ci.Crisk` (confidence intervals for Cumulative risks) in the `Epi` package:

We can now run the function using the model objects for the three competing events, using a common prediction data frame, `nd` for the rates. The time points in the frame must be so closely spaced that it makes sense to assume the rates constant in each interval; here we use intervals of length 1/20 years, in real applications we would use 1/50 (about 1 week) or less:

```
> system.time(
+ res <- ci.Crisk(list(OAD = mO,
+                       Ins = mI,
+                      Dead = mD),
+                          nd = data.frame(tfd = seq(0, 10, 1/20)),
+                          nB = 10000,
+                        perm = 4:1))
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05
  bruger   system forløbet
   48.79     3.87    61.44
```

```
> str(res)
```

```
List of 4
 $ Crisk: num [1:201, 1:4, 1:3] 1 0.943 0.897 0.86 0.829 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:201, 1:3, 1:3] 0 0.00457 0.00869 0.01243 0.01586 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:3] "Dead" "Dead+Ins" "Dead+Ins+OAD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:201, 1:4, 1:3] 0 0.0486 0.0946 0.1385 0.1807 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:201] 0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 ...
 - attr(*, "int")= num 0.05
```

As we see, the returned object (`res`) is a list of length 4, the first 3 components are 3-way arrays, and the last the vector of times of the first dimension of the arrays. The latter is mainly for convenience in further processing—it is easier to write `res$time` than `as.numeric(dimnames(res$Crisk)[[1]])`.

The three first components of `res` represent:

- `Crisk`: Cumulative `risk`s for each state
- `Srisk`: Stacked cumulative `risk`s across states
- `Stime`: Sojourn `time`s in each state, truncated at each point of the time dimension.

The first dimension of each array is time corresponding to endpoints of intervals of length `int`, (normally assumed starting at 0, but not necessarily). The second dimension is states (or combinations thereof). The last dimension of the arrays is the type of statistic; `50%` is the median of the samples, and the bootstrap confidence intervals as indicated; taken from the `alpha` argument to `ci.Crisk` (defaults to 0.05).

The argument `perm` governs in which order the state probabilities are stacked in the `Srisk` element of the returned list, the default is the states in the order given in the list of models in the first argument to `ci.Crisk` followed by the survival.

If we want the bootstrap samples to make other calculations we can ask the function to return the bootstrap samples of the rates by using the argument `sim.res = 'rates'` (defaults to `'none'`):

```
> system.time(
+ rsm <- ci.Crisk(list(OAD = mO,
+                       Ins = mI,
+                      Dead = mD),
+                           nd = data.frame(tfd = seq(0, 10, 1/20)),
+                           nB = 10000,
+                      sim.res = 'rates'))
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05
  bruger   system forløbet
    0.30     0.09     0.43
> str(rsm)
 num [1:201, 1:3, 1:10000] 0.743 0.655 0.578 0.51 0.45 ...
 - attr(*, "dimnames")=List of 3
  ..$ time: chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ mod : chr [1:3] "OAD" "Ins" "Dead"
  ..$ sim : chr [1:10000] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.05
 - attr(*, "time")= num [1:201] 0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 ...
```

This is 10,000 bootstrap samples (defined by `nB=`) of the rates evaluated at the 201 endpoints of the intervals (defined in `nd`).

Alternatively we can get the bootstrap samples of the cumulative risks by setting `sim.res = 'crisk'`:

```
> system.time(
+ csm <- ci.Crisk(list(OAD = mO,
+                       Ins = mI,
+                      Dead = mD),
+                           nd = data.frame(tfd = seq(0, 10, 1/20)),
+                           nB = 10000,
+                      sim.res = 'crisk'))
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05
  bruger   system forløbet
   21.08     1.13    22.97
> str(csm)
 num [1:201, 1:4, 1:10000] 1 0.943 0.898 0.861 0.831 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  ..$ sim  : chr [1:10000] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.05
```

These are 10,000 simulated samples of the cumulative risks evaluated at the 201 endpoints of the intervals, and also includes the survival probability in the first slot of the 2nd dimension of `csm`.

3.3 Simulated confidence intervals for rates

In figure 2.2 we showed the rates with confidence intervals from the model. But in `rsm` we have 10,000 parametric bootstrap samples of the occurrence rates, so we can derive the bootstrap medians and the bootstrap c.i.s:

```
> system.time(
+ Brates <- aperm(apply(rsm,
+                        1:2,
+                        quantile,
+                        probs = c(.5, .025, .975)),
+                  c(2, 3, 1)))
  bruger   system forløbet
    1.73     0.16     1.95
> str(Brates)
 num [1:201, 1:3, 1:3] 0.723 0.641 0.568 0.504 0.448 ...
 - attr(*, "dimnames")=List of 3
  ..$ time: chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ mod : chr [1:3] "OAD" "Ins" "Dead"
  ..$     : chr [1:3] "50%" "2.5%" "97.5%"
```

. . . or alternatively with piping the `tidyverse` way.

```
> system.time(
+ apply(rsm,
+        1:2,
+        quantile,
+        probs = c(.5, .025, .975)) |>
+ aperm(c(2, 3, 1)) -> Brates)
  bruger   system forløbet
    1.78     0.06     1.90
> str(Brates)
 num [1:201, 1:3, 1:3] 0.723 0.641 0.568 0.504 0.448 ...
 - attr(*, "dimnames")=List of 3
  ..$ time: chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ mod : chr [1:3] "OAD" "Ins" "Dead"
  ..$     : chr [1:3] "50%" "2.5%" "97.5%"
```

> CODE EXPLAINED: `apply` applies the function in the third argument (`quantile`) over the margins given in the second. The fourth argument (`probs`) is passed on to the function (`quantile`). So for each combination of the two first dimensions of `rsm`, the three quantiles are computed across the 10,000 bootstrap samples in the third. The three quantiles will be returned as the first dimension of the result. Therefore it will have dimensions (3, 201, 3). `aperm` permutes the sequence of the dimension of the array, so the three quantiles becomes the last dimension.

Then we can plot the bootstrap estimates on top of the estimates based on the normal approximation to distribution of the parameters. They are—not surprisingly—in close agreement since they are both based on an assumption of normality of the parameters on the log-rate scale:

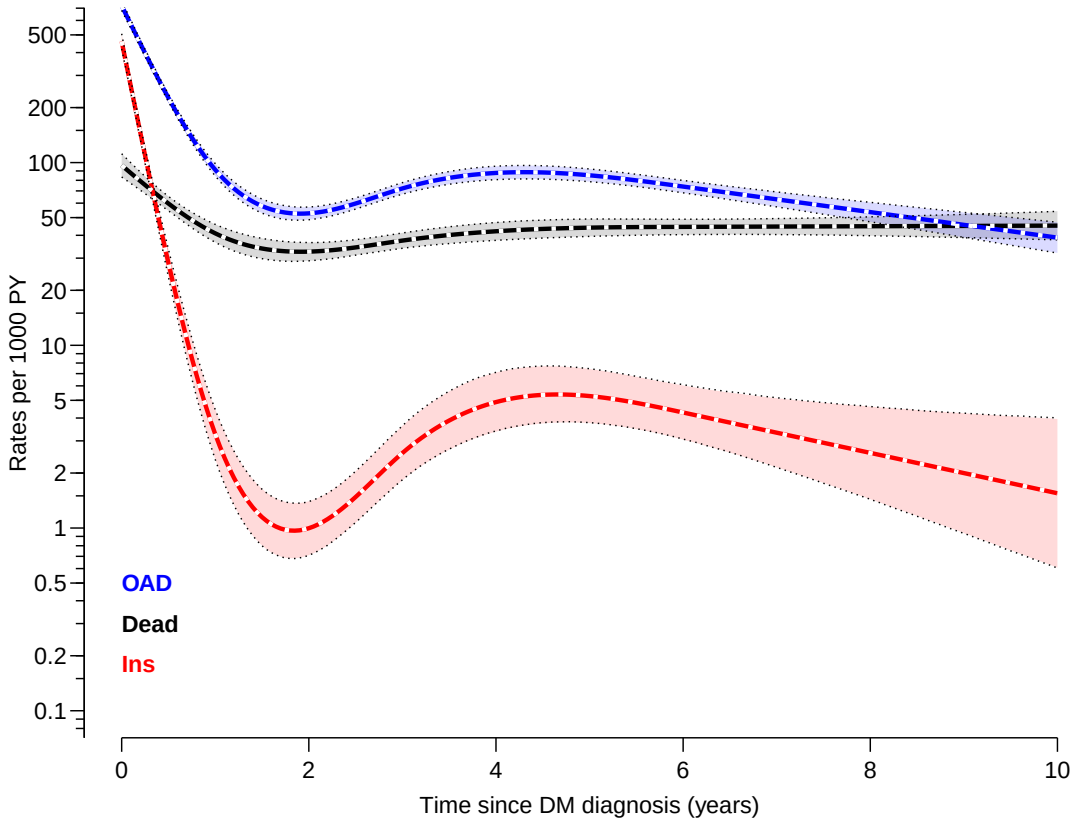

Figure 3.1: *Estimated rates from the* DM *state, estimates are from* gam *models fitted to data split in 1 month intervals (1/12 year, that is). The white dotted curves are the bootstrap medians, black dotted curves are the bootstrap 95% c.i.s.*

../graph/comp-rates-ci

```
> matshade(nd$tfd, cbind(ci.pred(mD, nd),
+                        ci.pred(mI, nd),
+                        ci.pred(mO, nd)) * 1000,
+          ylim = c(0.1,500), yaxt = "n",
+          ylab = "Rates per 1000 PY",
+          xlab = "Time since DM diagnosis (years)",
+          col = c("black", "red", "blue"), log = "y", lwd = 3, plot = TRUE)
> matlines(nd$tfd,
+          cbind(Brates[, "Dead", ],
+                Brates[, "Ins" , ],
+                Brates[, "OAD" , ]) * 1000,
+          col = c("white", "black", "black"), lty = 3, lwd = c(3,1,1))
> axis(side = 2, at = ll <- outer(c(1,2,5), -2:3, function(x, y) x * 10^y),
+                labels = formatC(ll, digits = 4), las = 1)
> axis(side = 2, at = outer(c(1.5, 2:9), -2:3, function(x, y) x * 10^y),
+                labels = NA, tcl = -0.3)
> text(0, 0.5 * 0.6^c(1,2,0),
+      c("Dead", "Ins", "OAD"),
+      col = c("black", "red", "blue"), adj = 0, font = 2)
```

We see that with 10,000 bootstrap samples there is no difference between the estimates and confidence intervals calculated by bootstrapping. This is really not a surprise; the

confidence intervals from the model is computed using the same distributional assumptions (multivariate normal) as the simulation machinery is using.

3.4 Exercise 10: Confidence intervals for cumulative risks

In the `Crisk` component of `res` we have the cumulative risks as functions of of time, with bootstrap confidence intervals, so we can easily plot the three cumulative risks:

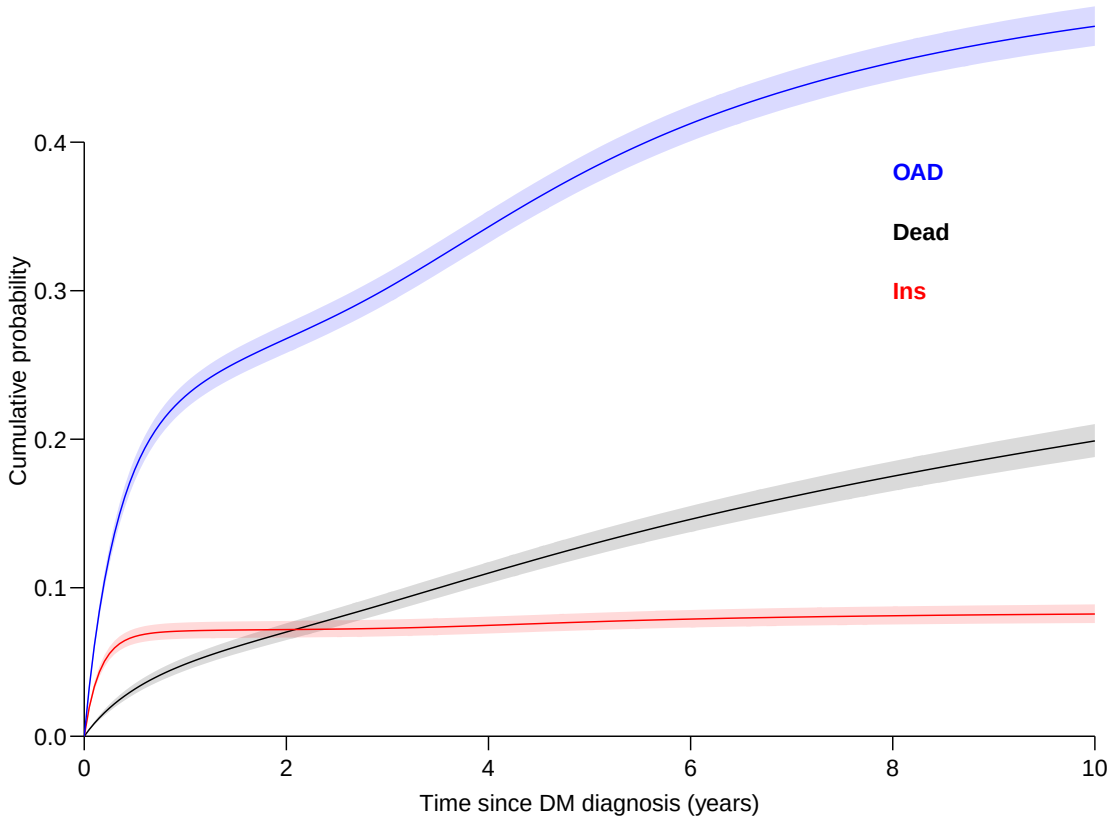

Figure 3.2: *Cumulative risks for the three types of events, with 95% bootstrap-based confidence intervals as shades.*

`../graph/comp-crates`

```
> matshade(res$time,
+          cbind(res$Crisk[,"Dead",],
+                res$Crisk[,"Ins" ,],
+                res$Crisk[,"OAD" ,]), plot = TRUE,
+          xlim = c(0,10), xaxs = "i", yaxs = "i", las = 1,
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Cumulative probability",
+           col = c("black","red","blue"))
> text(8, 0.3 + c(1, 0, 2) / 25,
+      c("Dead", "Ins", "OAD"),
+      col = c("black", "red", "blue"), adj = 0, font = 2)
```

3.5 Confidence intervals for stacked cumulative risks

Unlike the single cumulative risks where we have a confidence interval for each cumulative risk, when we want to show the stacked probabilities we must deliver the confidence intervals for the relevant sums, they are in the `Srisk` component of `res`.

```
> str(res$Crisk)
 num [1:201, 1:4, 1:3] 1 0.943 0.897 0.86 0.829 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  ..$      : chr [1:3] "50%" "2.5%" "97.5%"
> str(res$Srisk)
 num [1:201, 1:3, 1:3] 0 0.00457 0.00869 0.01243 0.01586 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfd  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ cause: chr [1:3] "Dead" "Dead+Ins" "Dead+Ins+OAD"
  ..$      : chr [1:3] "50%" "2.5%" "97.5%"
```

But we start out by plotting the stacked probabilities using `mat2pol` (`mat`rix to `pol`ygon), the input required is the single components from the `Crisk` component. Then we add the confidence intervals as white shades (using `matshade`):

```
> zz <- mat2pol(res$Crisk[,c("Dead", "Ins", "OAD", "Surv"),1],
+               x = res$time,
+            xlim = c(0, 10), xaxs = "i", yaxs = "i", las = 1,
+            xlab = "Time since DM diagnosis (years)",
+            ylab = "Probability",
+             col =  c("black","red","blue","forestgreen") )
> text(9, mp(zz["9",]), c("Dead", "Ins", "OAD", "DM"), col = "white" )
> matshade(res$time,
+          cbind(res$Srisk[, 1, ],
+                res$Srisk[, 2, ],
+                res$Srisk[, 3, ]),
+          col = 'transparent', col.shade = "white", alpha = 0.4)
```

3.6 Exercise 11: Sojourn times

From the `Stime` component of the `res` we can derive the estimated time spent in each state during the first, say, 5 or 10 years:

When referring to the times, we use *character* values—5 and 10 years are not necessarily at the 5th and 10th positions of the first dimension of the `Stime` array:

```
> s510 <- res$Stime[c("5", "10"),,]
> dimnames(s510)[[1]] <- c(" 5 yr","10 yr")
> round(ftable(s510, row.vars=1:2), 2)
              50% 2.5% 97.5%
tfd   cause
 5 yr Surv   2.88 2.83  2.93
      OAD    1.38 1.33  1.43
      Ins    0.35 0.33  0.38
      Dead   0.39 0.36  0.42
```

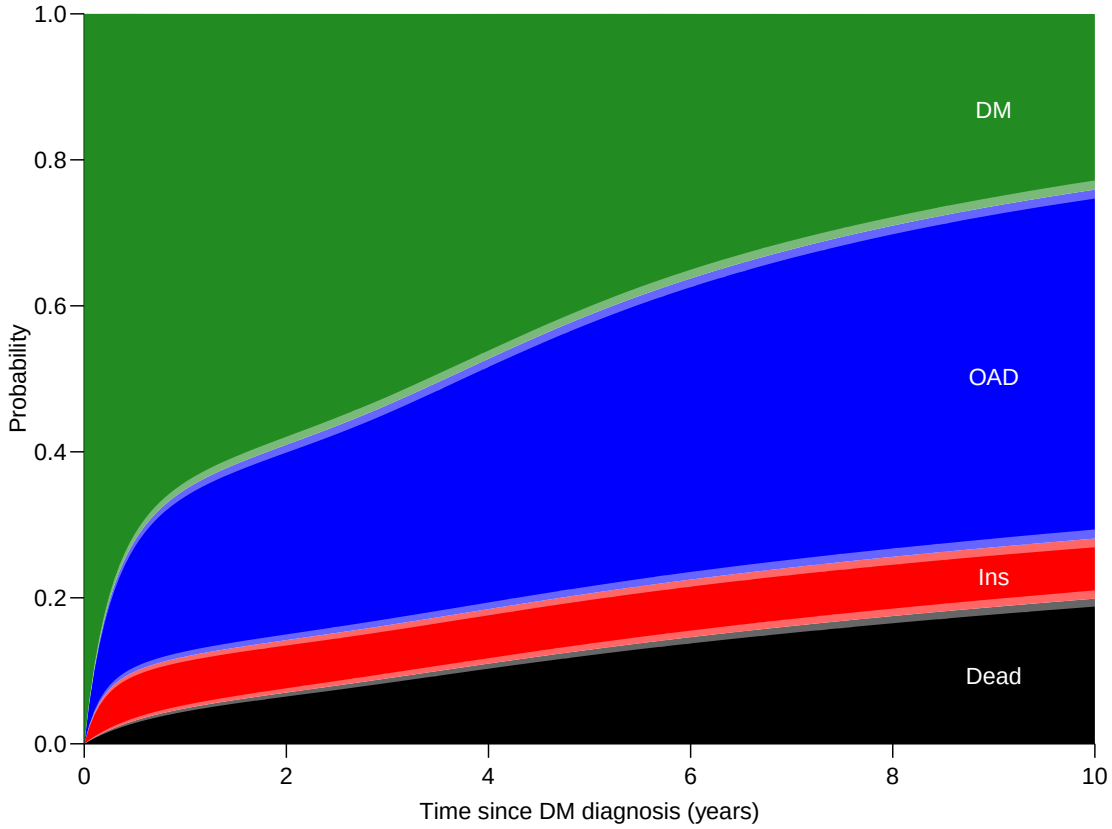

Figure 3.3: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`*. We are not concerned about what occurs* after *these events.* `Dead` *means dead without being on any drug. The white shadings around the borders between colored areas represent the 95% confidence intervals for the (sum of) probabilities.*

`../graph/comp-stack-ci`

```
10 yr Surv   4.44 4.34  4.54
      OAD    3.58 3.48  3.68
      Ins    0.75 0.70  0.81
      Dead   1.22 1.15  1.29
```

So we see that the expected life lived without pharmaceutical treatment during the first 10 years after DM diagnosis is 4.44 years with a 95% CI of (4.34, 4.54), and during the first 5 years 2.88 years with a 95% CI of (2.83, 2.93).

Chapter 4

A simple illustration of `ci.Crisk`

The following is a terse cook-book illustration of how to use the `ci.Crisk` function.

4.1 Exercise 12: A `Lexis` object with 3 causes

For illustration we define a `Lexis` object similar to `dl`, but expanded by two extra time scales:

```
> data(DMlate)
> dl <- mutate(DMlate, dofin = pmin(dodth, dooad, doins, dox, na.rm = TRUE),
+                      xstat = factor(case_when(dofin == dodth ~ "Dth",
+                                               dofin == dooad ~ "OAD",
+                                               dofin == doins ~ "Ins",
+                                                         TRUE ~ "DM"),
+                                     levels = c("DM", "OAD", "Ins", "Dth")))
> dmL <- Lexis(entry = list(per = dodm,
+                           age = dodm - dobth,
+                           tfD = dodm - dodm),
+              exit = list(per = dofin),
+       exit.status = xstat,
+              data = dl)
NOTE: entry.status has been set to "DM" for all.
NOTE: Dropping  2468  rows with duration of follow up < tol

> summary(dmL, t = T)

Transitions:
     To
From   DM  OAD Ins  Dth  Records:  Events: Risk time:  Persons:
  DM 2829 2966 681 1056      7532     4703   22920.27      7532

Timescales:
per age tfD
 ""  ""  ""
```

We can show the overall rates (the default `boxes` is *very* primitive):

```
> boxes(dmL, boxpos = TRUE)
```

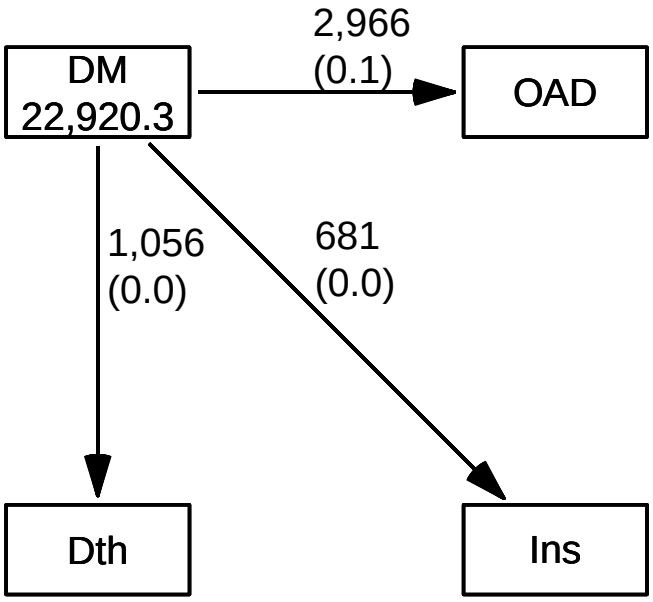

Figure 4.1: *Transitions from DM to different causes of exit. You probably want to explore the other arguments to* `boxes`.

../graph/comp-boxes

4.2 Models for the rates

In order to model the cause-specific mortality rates by sex and time from diagnosis (`tfD`), we first split the data in 6-month intervals

```
> sL <- splitLexis(dmL, time.scale = "age", breaks = seq(0, 120, 1/2))
> summary(sL)
Transitions:
     To
From    DM  OAD Ins  Dth  Records:  Events: Risk time:  Persons:
  DM 48630 2966 681 1056     53333     4703   22920.27      7532


> mOAD <- gamLexis(sL, ~ s(tfD, by=sex), to = "OAD")
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->OAD
> mIns <- gamLexis(sL, ~ s(tfD, by=sex), to = "Ins")
```

```
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->Ins
> mDth <- gamLexis(sL, ~ s(tfD, by=sex), to = "Dth")
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->Dth
```

4.3 Derived measures

With these three models for the occurrence rates we can compute the cumulative risks of exiting each of the causes. We need a prediction data frame that gives the rates at closely spaced times, in this case for men. For women the code would be practically the same:

```
> nm <- data.frame(tfD = seq(0, 10, 1/20), sex = "M")
```

Note that we can rename the states as we please by naming the entries in the list of models we supply to `ci.Crisk`:

```
> system.time(
+ cR <- ci.Crisk(list(OAD = mOAD,
+                      Ins = mIns,
+                      Dth = mDth),
+                 nB = 5000,
+                 nd = nm))
NOTE: Times are assumed to be in the column tfD at equal distances of 0.05
  bruger   system forløbet
   24.56     1.28    26.96
> str(cR)
List of 4
 $ Crisk: num [1:201, 1:4, 1:3] 1 0.952 0.913 0.883 0.858 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dth"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:201, 1:3, 1:3] 0 0.00431 0.00822 0.01179 0.01508 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:3] "Dth" "Dth+Ins" "Dth+Ins+OAD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:201, 1:4, 1:3] 0 0.0488 0.0954 0.1403 0.1838 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dth"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:201] 0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 ...
 - attr(*, "int")= num 0.05
```

Note that we get three arrays: `Crisk`, the cumulative risks; `Srisk`, the stacked risks and `Stime`, the sojourn times in each state. Finally, for convenience we also have the component `time`, the times at which the cumulative risks are computed. It is also available as the clumpy expression `as.numeric(dimnames(cR$Crisk)[[1]])`, but `cR$time` is easier.

4.3.1 Cumulative risks

We can plot the cumulative risks for death from each of the three causes, note we use the colors from last. Note that the time points are in the `time` component of the `Crisk` object:

```
> clr <- c("limegreen", "orange", "black")
> matshade(cR$time, cbind(cR$Crisk[, "OAD", ],
+                         cR$Crisk[, "Ins", ],
+                         cR$Crisk[, "Dth", ]),
+          col = clr, lty = 1, lwd = 2,
+          plot = TRUE, ylim = c(0, 0.5), yaxs = "i",
+          xlab = "Time since DM (years)", ylab = "Probability of state")
> text(0, 1/3 - c(1,3,2)/30, c("OAD", "Ins", "Dth"),
+      col = clr, adj = 0, font = 2)
```

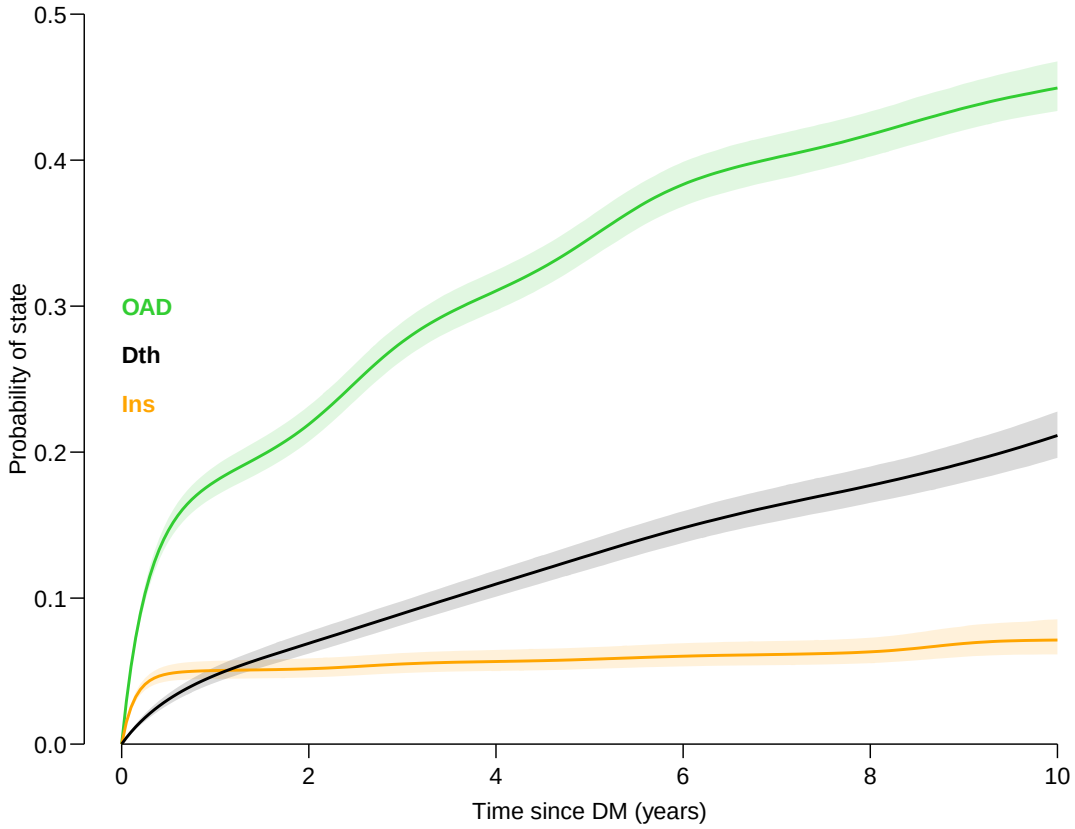

Figure 4.2: *Cumulative risks of each cause of exit based on* `gam` *models for the cause-specific rates.*

`../graph/comp-cR`

We also have the stacked probabilities so we can show how the population is distributed across the states at any one time:

4.3.2 Stacked cumulative risks

We also get the stacked probabilities in the order that we supplied the models, so that if we plot these we get the probabilities of being dead from each cause as the *difference* between

the curves. And the confidence intervals are confidence intervals for the cumulative sums of probabilities.

```
> matshade(cR$time, cbind(cR$Srisk[,1,],
+                         cR$Srisk[,2,],
+                         cR$Srisk[,3,]),
+          col = "black", lty = 1, lwd = 2,
+          plot = TRUE, ylim = c(0,1), xaxs = "i", yaxs = "i")
> text(9, mp(c(0, cR$Srisk["9", , 1], 1)),
+      rev(c(dimnames(cR$Crisk)[[2]])))
> box(bty = "o")
```

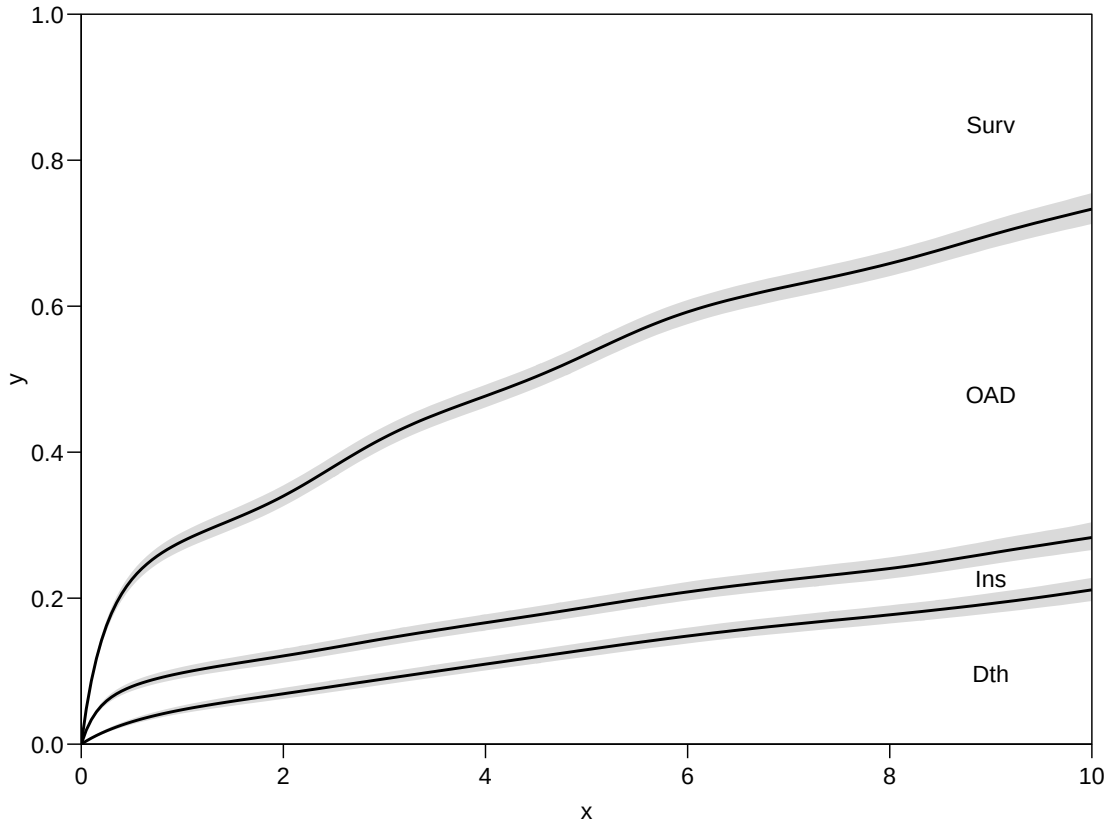

Figure 4.3: *Stacked cumulative risks.*

`../graph/comp-Sr1`

It is not a good idea to color these curves, they do not refer to the causes of exit, it is the areas *between* the curves that refer to causes. By the same token, since the quantity of interest is the area between the curves and horizontal lines at 0 and 1, it is important that the horizontal axes are placed at precisely 0 and 1 on the vertical axis. This is what `yaxs = "i"` achieves.

It would be more logical to color the areas *between* the curves. which can be done by `mat2pol` (`mat`rix to `pol`ygons) using the `Crisk` component. We can then superpose the confidence intervals for the sum of the state probabilities using `matshade` by adding white shades:

```
> zz <- mat2pol(cR$Crisk[, c("Dth", "Ins", "OAD", "Surv"), "50%"],
+               x = cR$time,
+            xlim = c(0,10), xaxs = "i", yaxs = "i", las = 1,
+            xlab = "Time since DM diagnosis (years)",
+            ylab = "Probability",
+             col =  c(gray(0.3), "red", "blue", "limegreen"))
> matshade(cR$time, cbind(cR$Srisk[,1,],
+                         cR$Srisk[,2,],
+                         cR$Srisk[,3,]),
+          col = "transparent", col.shade = "white", alpha = 0.4)
> text(9, mp(c(0, cR$Srisk["9", , 1], 1)),
+      rev(c(dimnames(cR$Crisk)[[2]])), col = "white")
> box(bty = "o", col ="white")
```

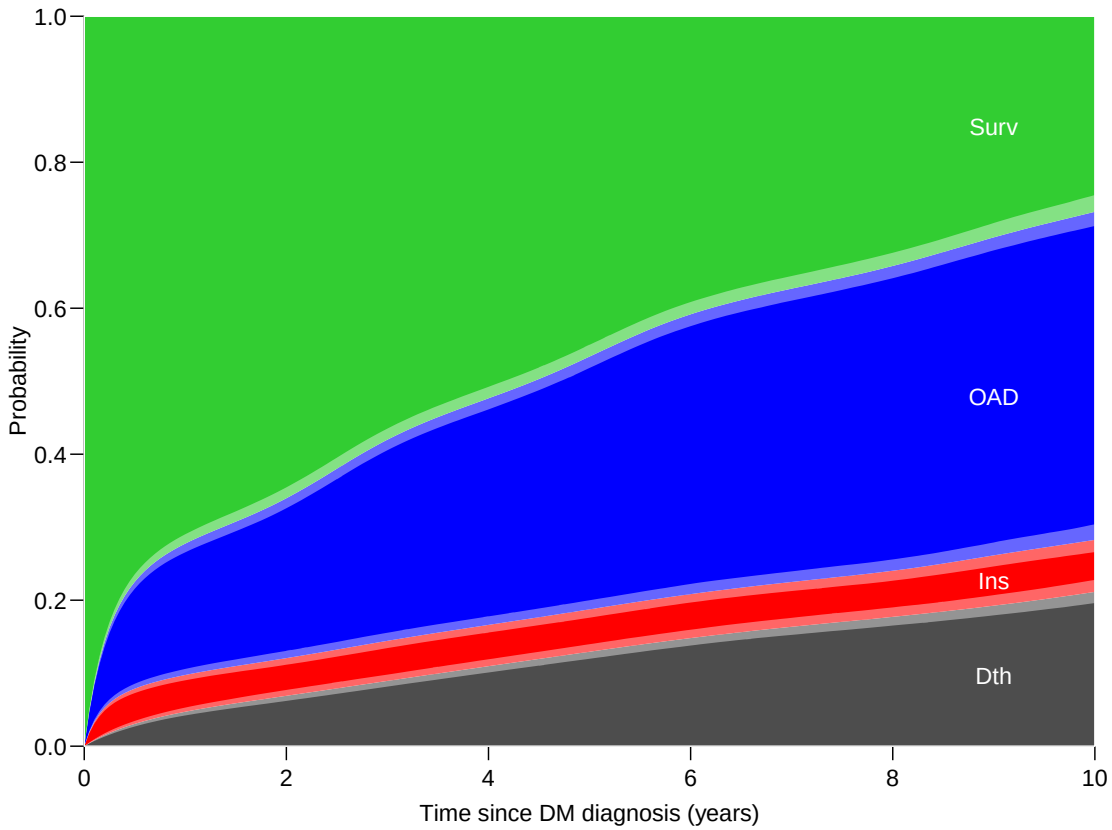

Figure 4.4: *Stacked cumulative risks with coloring of states and overlaid with confidence intervals for the probabilities shown; that is the relevant sums.*

../graph/comp-Sr2

4.3.3 Sojourn times

The third component of the result, `Stime` is an array of sojourn times over intervals starting at 0 and ending at the time indicated by the first dimension:

```
> ftable(round(cR$Stime[paste(c(2, 5, 10)), , ], 2), row.vars = 1)
```

```
    cause Surv             OAD             Ins             Dth
           50% 2.5% 97.5%  50% 2.5% 97.5%  50% 2.5% 97.5%  50% 2.5% 97.5%
tfD
2         1.49 1.47  1.51 0.33 0.31  0.35 0.09 0.08  0.11 0.09 0.08  0.10
5         3.16 3.10  3.21 1.20 1.15  1.25 0.26 0.23  0.29 0.39 0.36  0.42
10        4.94 4.83  5.06 3.24 3.13  3.35 0.58 0.51  0.65 1.24 1.16  1.31
```

The sojourn times in the three exit states can be taken as the years of life without drugs (that is time in the `Surv` state) lost to each of the exit causes. The sum of the medians for the three causes equals the time frame (2, 5, 10) minus the `Surv` component.

So we see that during the first 5 resp. 10 years after diagnosis of diabetes, the expected years alive without medication is 3.16 and 4.95 years.

4.4 Exercise 13: Comparing groups

Finally, we may want to see the *difference* (or ratio) of survival/exit probabilities between men and women, say. This can be derived from two bootstrap samples using different prediction frames (the argument `nd=` to `ci.Crisk`). But the two bootstrap samples of parameters must be the same, i.e. come from the *same* stream of samples from the multivariate normal. This way the differences between the bootstrap samples will represent bootstrap samples of the differences.

This can be obtained by explicitly setting the seed for the random number generator to the *same* value before calling `ci.Crisk` with each of the two different prediction frames as `nd` argument:

```
> nm <- data.frame(tfD = seq(0, 10, 1/20), sex = "M")
> nw <- data.frame(tfD = seq(0, 10, 1/20), sex = "F")
> # set the seed
> set.seed(1952)
> mR <- ci.Crisk(list(OAD = mOAD,
+                     Ins = mIns,
+                     Dth = mDth),
+                nd = nm,
+                nB = 500,
+           sim.res = "crisk" )
NOTE: Times are assumed to be in the column tfD at equal distances of 0.05
> # REset the seed to get the same sequence
> set.seed(1952)
> wR <- ci.Crisk(list(OAD = mOAD,
+                     Ins = mIns,
+                     Dth = mDth),
+                nd = nw,
+                nB = 500,
+           sim.res = "crisk" )
NOTE: Times are assumed to be in the column tfD at equal distances of 0.05
> str(wR)
 num [1:201, 1:4, 1:500] 1 0.967 0.939 0.916 0.895 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfD  : chr [1:201] "0" "0.05" "0.1" "0.15" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dth"
  ..$ sim  : chr [1:500] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.05
```

The two samples are now from identical streams of random numbers, so we can get differences and ratios of the survival curves between men and women:

```
> dS <- mR[,"Surv",] - wR[,"Surv",]
> dS <- apply(dS, 1, quantile, probs = c(.5, .025, .975)) * 100
> str(dS)
 num [1:3, 1:201] 0 0 0 -1.76 -2.11 ...
 - attr(*, "dimnames")=List of 2
  ..$    : chr [1:3] "50%" "2.5%" "97.5%"
  ..$ tfD: chr [1:201] "0" "0.05" "0.1" "0.15" ...
> rS <- mR[,"Surv",] / wR[,"Surv",]
> rS <- apply(rS, 1, quantile, probs = c(.5, .025, .975))
```

We can then plot the differences and the ratios of the probabilities—note that the dimension of the function `apply`ed becomes the first dimension of the result:

```
> par(mfrow = c(1,2))
> matshade(as.numeric(colnames(dS)), t(dS), plot = TRUE,
+          lwd = 3, ylim = c(-7, 7),
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Men - Women survival difference (%)")
> abline(h = 0)
> matshade(as.numeric(colnames(rS)), t(rS), plot = TRUE,
+          lwd = 3, ylim = c(1/1.2, 1.2), log ="y",
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Men - Women survival ratio")
> abline(h = 1)
```

To illustrate the effect of *not* pairing the random samples we can generate a fresh sample for women from a different stream (by `not` setting the seed) and do the calculations to illustrate the excess we get from not aligning samples.

```
> fR <- ci.Crisk(list(OAD = mOAD,
+                     Ins = mIns,
+                     Dth = mDth),
+                nd = nw,
+                nB = 500,
+            sim.res = "crisk" )
NOTE: Times are assumed to be in the column tfD at equal distances of 0.05
> dxS <- mR[,"Surv",] - fR[,"Surv",]
> dxS <- apply(dxS, 1, quantile, probs = c(.5, .025, .975)) * 100
> rxS <- mR[,"Surv",] / fR[,"Surv",]
> rxS <- apply(rxS, 1, quantile, probs = c(.5, .025, .975))
```

Note that the simulation approach makes it easy to evaluate the difference between men and women on both the relative risk (ratio of cumulative risks) or risk difference scales.

```
> par(mfrow = c(1,2))
> matshade(as.numeric(colnames(dS)), t(dS), plot = TRUE,
+          lwd = 3, ylim = c(-7, 7),
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Men - Women drug-free survival difference (%)")
> matshade(as.numeric(colnames(dxS)), t(dxS),
+          lty = "21", lend = "butt", lwd = 3, col = "forestgreen")
> abline(h = 0)
> matshade(as.numeric(colnames(rS)), t(rS), plot = TRUE,
```

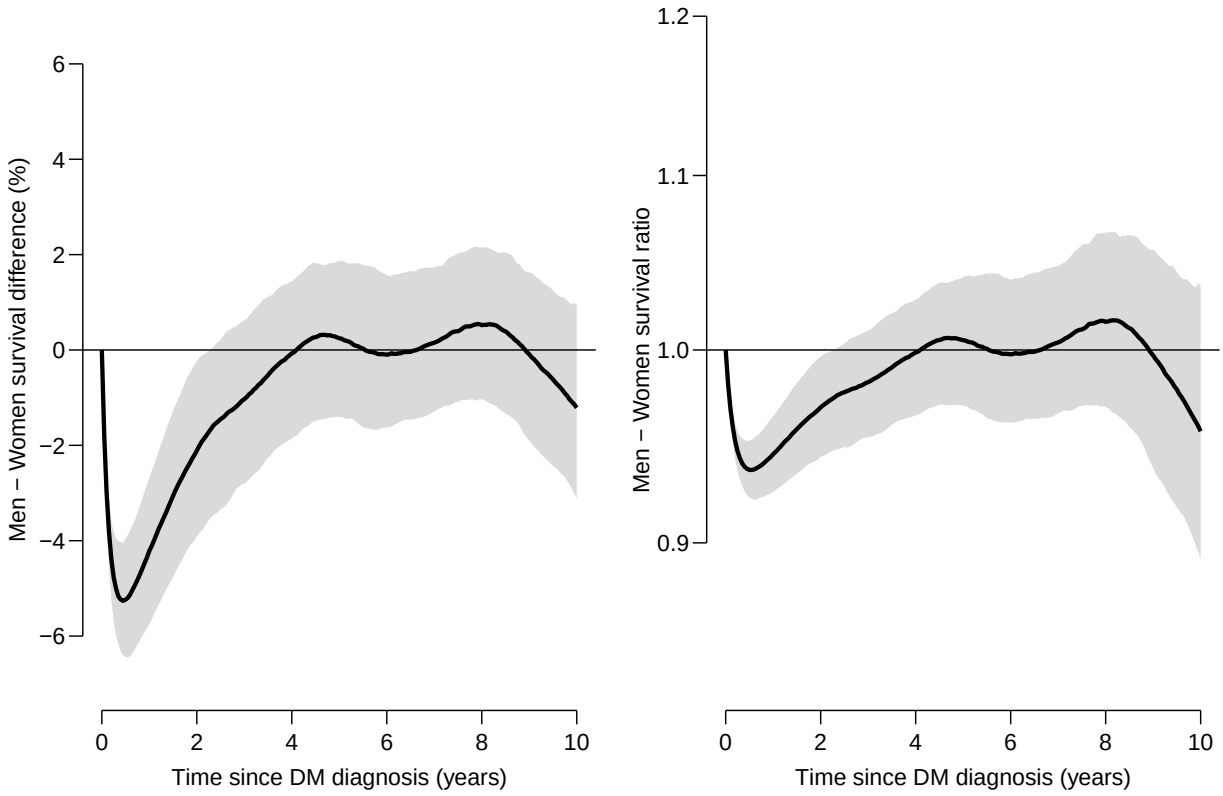

Figure 4.5: *Differences and ratios of drug-free survival between men and women, derived from the same set of bootstrap samples from the parameter vector.*

`../graph/comp-difrat`

```
+          lwd = 3, ylim = c(1/1.2, 1.2), log ="y",
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Men - Women drug-free survival ratio")
> matshade(as.numeric(colnames(rxS)), t(rxS),
+          lty = "21", lend = "butt", lwd = 3, col = "forestgreen")
> abline(h = 1)
```

We see that by using different random streams for evaluation of the cumulative risks for men and women, we introduce a small bias, but a substantial underestimation of the precision of the risk differences differences.

The main conclusion is that during the first two years, men have a smaller probability of remaining drug-free than women, but the difference has disappeared after about 4 years.

```
  Start time: 2025-05-05, 14:52:55
    End time: 2025-05-05, 14:55:33
Elapsed time: 2.63 minutes
```

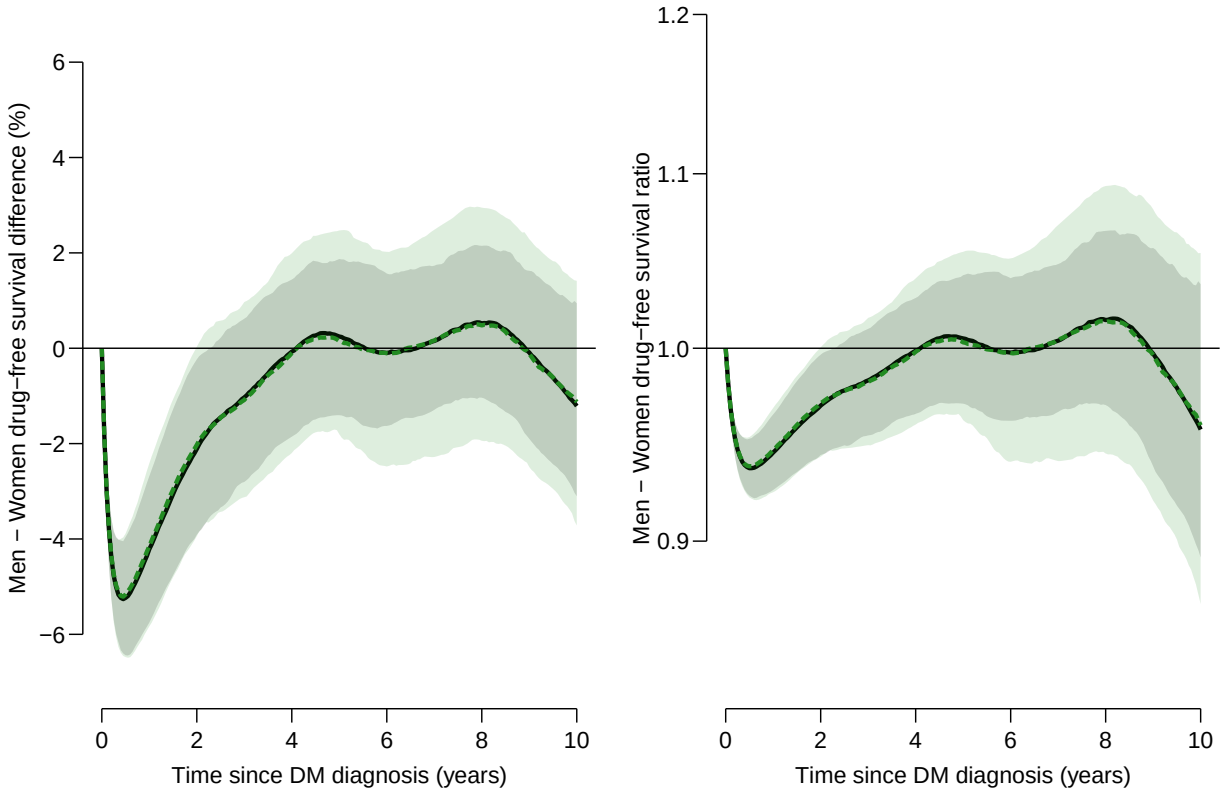

Figure 4.6: *Differences and ratios of drug-free survival between men and women, derived from separate bootstrap samples. The outer confidence bands are from bootstrap samples not properly paired between men end women.*

`../graph/comp-difratx`

Chapter 5

Expected survival time (RMST)

. . . now input from `rmst.tex`

```
> library(Epi)
> library(popEpi)
> library(survival)
> library(tidyverse)
> clear()
```

```
     R  Epi popEpi
 4.5.0 2.59 0.4.13
```

5.1 Exercise 14: Expected lifetime or RMST — preliminaries

The term RMST — Restricted Mean Survival Time — has become a popular term for (a variant of) expected lifetime, or more precisely expected residual lifetime as has been available in published life tables for eons. The term "sojourn time" is also used for the time spent in a given state.

Suppose we look at the mortality rates among diabetes patients of the two different sexes. We can then compare men to women by looking at the mortality rate-ratio (M/W HR, typically a function of time), but we might also compare men and women by 5 or 10 year survival or by the RMST during the next, say, 10 years for a given age, say, 60. Note that RMST refers to an *interval*, in this case age 60 to $60 + 10$

For illustration we again use a random sample of the diabetes patient data set, but this time also include age as a time scale:

```
> data(DMlate)
> set.seed(19540803)
> DMlate <- DMlate[sample(1:nrow(DMlate), 1000), ]
> Lx <- Lexis(entry = list(age = dodm - dobth,
+                          tfd = 0),
+             exit = list(tfd = dox - dodm),
+       exit.status = factor(!is.na(dodth),
+                            labels = c("DM", "Dead")),
+             data = DMlate)
NOTE: entry.status has been set to "DM" for all.
```

```
> sL <- splitLexis(Lx, seq(0, 15, 0.5), "tfd")
> summary(Lx)
Transitions:
     To
From  DM Dead  Records:  Events: Risk time:  Persons:
  DM 769  231      1000      231    5398.05      1000
> summary(sL)
Transitions:
     To
From    DM Dead  Records:  Events: Risk time:  Persons:
  DM 11063  231     11294      231    5398.05      1000
```

We can model mortality rates as a function of age, diabetes duration and sex — the absence of interaction terms qualifies this as a proportional hazards model:

```
> m1 <- glmLexis(sL, ~ Ns(age, knots = c(30, 50, 70))
+                    + Ns(tfd, knots = c(0, 1, 4, 10))
+                    + sex)
stats::glm Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->Dead
> round(ci.exp(m1, subset = "sex"), 3)
     exp(Est.)  2.5% 97.5%
sexF     0.937 0.723 1.215
```

So as a summary we might assert that women have a mortality that is about 6% smaller that that of men (HR is 0.937).

What hazards are proportional here?

Comparative measures on other possible outcome scales are:

- differences in survival probabilities at certain *times*
- differences in expected life times during certain *time intervals*

But in order to compute these we need to specify times and the intervals of interest:

- *at* what times since diagnosis do we want comparison of survival between men and women
- *from* what time and *to* what time do we want the expected lifetime computed?
- for what age (`adx`, age at diagnosis) do we want the comparison

Note that when we say "what", we do not necessarily refer to a single value of covariates in the rate model, we can refer to a distribution of values, for example the empirical distribution in the study population (if anyone would be interested in that).

5.1.1 Exercise 15: Survival comparison

Now, say we want to compare 5 and 10 year survival for men and women, diagnosed with diabetes at ages 50, 60 and 70. To that end we need the survival curves (say, evaluated at times from 0 through 15 years in steps of .1 year) for men and women aged 50, 60 and 70 at diagnosis, that is the *predicted* survival, so we specify prediction data frames for each combination of these variable values.

```
> surv.arr <- NArray(list(adx = c(50, 60, 70),
+                         sex = c("M", "F"),
+                         tfd = tfd <- seq(0, 15, .1),
+                        surv = c("surv", "lo", "up")))
> str(surv.arr)
 logi [1:3, 1:2, 1:151, 1:3] NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 4
  ..$ adx : chr [1:3] "50" "60" "70"
  ..$ sex : chr [1:2] "M" "F"
  ..$ tfd : chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ surv: chr [1:3] "surv" "lo" "up"
```

CODE EXPLAINED: We use `NArray` to define a 4-dimensional array—a 4-dimensional table, `surv.arr`. We can refer to elements or "slices" of the array using "`[ , , , ]`"—note there are 3 commas needed to allow to refer to the 4 dimensions. We can refer to these either by name or number; for example, `surv.arr["60", , , ]` is the same as `surv.arr[2, , , ]`. Note the quoted `"60"`. Using `surv.arr[60, , , ]` would be asking for the 60th element of the first dimension which would crash the command because there are only 3 elements. `surv.arr["60", , , ]` is the human readable version.

Note that in the definition of the 3rd dimension of the array we use an assignment to define `tfd` as a vector in the global environment—to be used later.

We want to fill the array with the survival functions (with confidence intervals) for each combination of values of age at diagnosis (`adx`) and sex (`sex`)

```
> for(adx in c(50, 60, 70))
+ for( sx in c("M", "F"))
+    {
+    nd <- data.frame(tfd = tfd,
+                     age = adx + tfd,
+                     sex = sx)
+    surv.arr[paste(adx), sx, , ] <- ci.surv(m1, nd)
+    }
NOTE: interval length chosen from  as tfd[2] - tfd[1]
NOTE: interval length chosen from  as tfd[2] - tfd[1]
NOTE: interval length chosen from  as tfd[2] - tfd[1]
NOTE: interval length chosen from  as tfd[2] - tfd[1]
NOTE: interval length chosen from  as tfd[2] - tfd[1]
NOTE: interval length chosen from  as tfd[2] - tfd[1]
```

CODE EXPLAINED: We use loops over age at diagnosis, `adx`, and sex, `sx`, and for each combination of these we compute the survival function at times `tfd` from diagnosis.

We need `adx` as numeric in definition of `nd`, but when we use it to refer to the position in the array `surv.arr` we must convert it to character by `paste(adx)`.

The loop variable `sx` is of mode character, so needs no converting when used for indexing.

Note that we use "`tfd = tfd`" to define the variable `tfd` in `nd` with the values from the global environment variable we defined above.

We can show the survival probabilities and the differences in survival probabilities between men and women.

```
> ftable(surv.arr[,,c("5","10"),], row.vars = c(1,3,2))
            surv      surv        lo        up
adx tfd sex
50  5   M        0.9597302 0.9721581 0.9419227
        F        0.9622098 0.9744000 0.9443826
    10  M        0.9078803 0.9328699 0.8742350
        F        0.9134051 0.9382469 0.8792326
60  5   M        0.8971219 0.9207994 0.8668942
        F        0.9032569 0.9272499 0.8719147
    10  M        0.7755516 0.8215218 0.7199083
        F        0.7880259 0.8346398 0.7305444
70  5   M        0.7530853 0.7941397 0.7054767
        F        0.7666117 0.8078805 0.7181366
    10  M        0.5150269 0.5821581 0.4431678
        F        0.5369331 0.6035610 0.4648897
> # difference in survival probabilities (%)
> round(100 * ftable(surv.arr[,"F",c("5","10"),1] -
+                    surv.arr[,"M",c("5","10"),1]), 2)
    tfd    5   10
adx
50      0.25 0.55
60      0.61 1.25
70      1.35 2.19
> # remember the brackets when using the pipe, |>
> ((surv.arr[,"F",c("5","10"),1] -
+   surv.arr[,"M",c("5","10"),1]) * 100) |>
+                                ftable() |> round(2)
    tfd    5   10
adx
50      0.25 0.55
60      0.61 1.25
70      1.35 2.19
```

So we see there is remarkably little difference between the survival probabilities between men and women. But we do not have any confidence interval for the differences. We can get that by simulation (parametric bootstrap), which is also the case for the RMST, so let us consider that first.

5.1.2 Exercise 16: RMST

We can use `ci.Crisk` to get estimates of the restricted mean survival time for men and women respectively:

```
> head(nd)
  tfd  age sex
1 0.0 70.0   F
2 0.1 70.1   F
3 0.2 70.2   F
4 0.3 70.3   F
5 0.4 70.4   F
6 0.5 70.5   F
> msM <- ci.Crisk(list(Mort = m1), mutate(nd, sex = "M"))$Stime
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
```

```
> msF <- ci.Crisk(list(Mort = m1), mutate(nd, sex = "F"))$Stime

NOTE: Times are assumed to be in the column tfd at equal distances of 0.1

> str(msF)

 num [1:151, 1:2, 1:3] 0 0.0997 0.199 0.2977 0.396 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfd  : chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ cause: chr [1:2] "Surv" "Mort"
  ..$      : chr [1:3] "50%" "2.5%" "97.5%"
```

These are the *sojourn times* (the `Stime` component) from time ($\texttt{tfd} = 0, \texttt{age} = 70$) to the times given in first dimension of `msF`, so the estimated sojourn times or RMST during the first 5, resp. 10 years are for men and women (and the difference):

```
> msM[c("5","10"), "Surv", ]

tfd       50%     2.5%    97.5%
  5  4.376054 4.235329 4.493654
  10 7.528717 7.119472 7.896698

> msF[c("5","10"), "Surv", ]

tfd       50%     2.5%    97.5%
  5  4.410727 4.260937 4.531647
  10 7.648984 7.240340 8.026692

> msF[c("5","10"), "Surv", 1] - msM[c("5","10"), "Surv", 1]

         5         10
0.03467254 0.12026705
```

So we can see that women only have a very slightly longer expected lifetime during the first 5, resp. 10 years, than men diagnosed at age 70.

But this approach does not give us any confidence intervals for the difference. It would be nonsense to add or subtract confidence limits.

5.1.3 Exercise 17: Confidence intervals for RMST

We can get confidence intervals from (parametric) bootstrap samples of the cumulative rates. This is done by simulation from the distribution of the model parameters. Well, assumed distribution, taken to be multivariate normal. But this is just using the usual assumptions that underlie calculation of confidence intervals as ± twice the standard error—the 2 (or 1.96 as often used) is a characteristic of the normal distribution.

But first we set up an array to store the simulated cumulative risks:

```
> head(nd)
  tfd  age sex
1 0.0 70.0   F
2 0.1 70.1   F
3 0.2 70.2   F
4 0.3 70.3   F
5 0.4 70.4   F
6 0.5 70.5   F
```

```
> nB <- 10000 # no of bootstrap samples
> ain <- 5:7 * 10
> sex <- c("M", "F")
> simres <- NArray(list(adx = ain,
+                       sex = sex,
+                       tfd = nd$tfd,
+                       sim = 1:nB))
> str(simres)
 logi [1:3, 1:2, 1:151, 1:10000] NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> length(simres)
[1] 9060000
```

CODE EXPLAINED: `NArray` sets up an array of `NA`s; in this case classified by age at diagnosis, sex, time since diagnosis and bootstrap sample number. This is to be used to hold derived RMST for persons of given sex and age at diagnosis, as well as comparisons of these.

```
> for (adx in ain)
+ for ( sx in sex)
+     {
+ set.seed(20250503)
+ simres[paste(adx), sx, , ] <- ci.Crisk(list(Mort = m1),
+                                        nd = mutate(nd, sex = sx,
+                                                    age = adx + tfd),
+                                        nB = nB,
+                                        sim.res = "crisk")[, "Surv", ]
+ cat(adx, sx, format(Sys.time(), format = "%T"), "\n")
+     }
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
50 M 16:09:49
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
50 F 16:09:57
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 M 16:10:05
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 F 16:10:13
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 M 16:10:21
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 F 16:10:28
```

CODE EXPLAINED: In order to be able to compare men and women based on simulated samples of cumulative risks, the cumulative risks must be computed from the *same* stream of random numbers. In other words, the random number streams for men and women must be the same. And so for ages too.

For each combination of sex (`sx`) and age at diagnosis (`adx`), we therefore reset the random number generator seed to the same number, and simulate cumulative risks.

Resetting the random generator seed to the *same* number in each interaction of the loop has the effect that the random number streams used will be identical, so the only difference between the results will be the the different prediction frames—the second argument `nd`. Therefore, it will be possible to derive valid combinations of statistics from each simulated set of cumulative risks.

There is about 9 mil. entries in `simres`.

We now have 6 samples of 10,000 cumulative risk functions: for men and women at three different ages at entry (diagnosis), coming from synchronized streams of random numbers. In this context the cumulative risks are the state probabilities for the dead state (`Mort`) and the default alive state (`Surv`). The latter is the survival function, and the only one we extract.

From this array we want the expected time spent in state `Surv` between 0 and 5 years, and between 0 and 10 years. These are just the integrals of the state probabilities, so we devise a function that returns these quantities:

```
> mid <- function(z) z[-1] - diff(z) / 2
> get5.10 <-
+     function(x)
+         {
+         c("5" = sum(mid(x[1:51])),
+           "10"= sum(mid(x[1:101]))) / 10
+         }
```

CODE EXPLAINED: The `mid` function just computes the midpoint between successive values in a vector. The `get5.10` function takes the sum of the midpoint values, corresponding to the integral, and multiplies these by the interval length which was taken to be 1/10.

So these are typical examples of functions designed *ad hoc*, just for this use. This is why R is called a functional language.

With the function that computes the integral to 5 resp. 10 years we can employ this to each of the simulated cumulative risk functions and get simulated values of the RMST from diagnosis and 5 resp. 10 years on.

```
> str(simres)
 num [1:3, 1:2, 1:151, 1:10000] 1 1 1 1 1 ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> zz <- apply(simres, c(1,2,4), get5.10)
> str(zz)
 num [1:2, 1:3, 1:2, 1:10000] 4.9 9.56 4.73 8.91 4.35 ...
 - attr(*, "dimnames")=List of 4
  ..$    : chr [1:2] "5" "10"
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sex: chr [1:2] "M" "F"
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
```

CODE EXPLAINED: The `apply` function will use the function `get5.10` for each combination of the dimensions 1, 2 and 4 of the `simres` array, so we get 10,000 samples of the 5 and 10 year RMST for each combination of sex and age at diagnosis.

We can then derive the relevant quantiles for each combination of the first 3 dimensions:

```
> round(ftable(apply(zz,
+                    1:3,
+                    quantile, c(50, 5, 95)/100),
+              col.vars = 2:1), 2)
            5             10
          50%   5%  95%  50%   5%  95%
adx sex
50  M    4.90 4.86 4.93 9.57 9.43 9.68
    F    4.91 4.87 4.94 9.60 9.46 9.70
60  M    4.75 4.68 4.80 8.93 8.68 9.12
    F    4.76 4.69 4.82 8.99 8.74 9.19
70  M    4.38 4.26 4.47 7.52 7.18 7.83
    F    4.41 4.29 4.51 7.65 7.30 7.97

> # the tidyverse version would be
> apply(zz, 1:3, quantile, c(50, 5, 95)/100) |>
+                     ftable(col.vars = 2:1) |>
+                                    round(2)

            5             10
          50%   5%  95%  50%   5%  95%
adx sex
50  M    4.90 4.86 4.93 9.57 9.43 9.68
    F    4.91 4.87 4.94 9.60 9.46 9.70
60  M    4.75 4.68 4.80 8.93 8.68 9.12
    F    4.76 4.69 4.82 8.99 8.74 9.19
70  M    4.38 4.26 4.47 7.52 7.18 7.83
    F    4.41 4.29 4.51 7.65 7.30 7.97

> # the usual tidyverse version takes up a bit more space
> zz |> apply(1:3, quantile, c(50, 5, 95) / 100) |>
+                          ftable(col.vars = 2:1) |>
+                                          round(2)

            5             10
          50%   5%  95%  50%   5%  95%
adx sex
50  M    4.90 4.86 4.93 9.57 9.43 9.68
    F    4.91 4.87 4.94 9.60 9.46 9.70
60  M    4.75 4.68 4.80 8.93 8.68 9.12
    F    4.76 4.69 4.82 8.99 8.74 9.19
70  M    4.38 4.26 4.47 7.52 7.18 7.83
    F    4.41 4.29 4.51 7.65 7.30 7.97
```

CODE EXPLAINED: This use of `apply` finds the relevant quantiles from the 10,000 simulated samples of the RMST.

`ftable` produces a flat `table` that is readable.

So we see that men and women aged 70 at diagnosis have approximately the same expected lifetime over the next 5 and 10 years.

5.2 Exercise 18: predicted mortality

The model for mortality that underlies the calculations was a proportional hazards model; the effect of time since diagnosis and age were supposed to be the same for men and women:

```
> m1 <- glmLexis(sL, ~ Ns(age, knots = c(50, 60, 70, 80))
+                    + Ns(tfd, knots = c(0, 1, 5))
+                    + sex)
stats::glm Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->Dead
```

```
> round(ci.exp(m1, subset = "sex"), 3)
```

```
     exp(Est.)  2.5% 97.5%
sexF     0.961 0.741 1.248
```

We did not even inspect how mortality rates looked, so let's do that by devising a prediction frame: duration of :

```
> pd <- expand.grid(tfd = c(NA, seq(0, 35, .5)),
+                   ain = c(50, 55, 60, 65, 70, 75)) |>
+       mutate(age = ain + tfd,
+              age = ifelse(age < 86, age, NA))
> head(pd)
  tfd ain  age
1  NA  50   NA
2 0.0  50 50.0
3 0.5  50 50.5
4 1.0  50 51.0
5 1.5  50 51.5
6 2.0  50 52.0
```

```
> matshade(pd$age, ci.pred(m1, mutate(pd, sex = "M")) * 1000,
+          xlab = "Attained age", ylab = "Mortality per 1000 PY",
+          plot = TRUE, col = "blue", lwd = 3, log = "y")
> matshade(pd$age, ci.pred(m1, mutate(pd, sex = "F")) * 1000,
+          col = "red", lwd = 3, log = "y")
```

5.3 Exercise 19: Interaction model

We see there is a minimal difference in mortality rates between men and women among diabetes patients in Denmark, let's see if there are any interactions between sex and time from diagnosis:

```
> mi <- glmLexis(sL, ~ (Ns(age, knots = c(50, 60, 70, 80)) +
+                       Ns(tfd, knots = c(0, 1, 5)))
+                    * sex)
stats::glm Poisson analysis of Lexis object sL with log link:
Rates for the transition:
DM->Dead
```

```
> anova(m1, mi, test = "Chisq")
```

```
Analysis of Deviance Table

Model 1: cbind(trt(Lx$lex.Cst, Lx$lex.Xst) %in% trnam, Lx$lex.dur) ~ Ns(age,
    knots = c(50, 60, 70, 80)) + Ns(tfd, knots = c(0, 1, 5)) +
    sex
Model 2: cbind(trt(Lx$lex.Cst, Lx$lex.Xst) %in% trnam, Lx$lex.dur) ~ (Ns(age,
    knots = c(50, 60, 70, 80)) + Ns(tfd, knots = c(0, 1, 5))) *
    sex
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1     11287     1979.0
2     11282     1977.1  5   1.9074   0.8618
```

The likelihood ratio test indicates no interaction between sex and diabetes duration, but the predicted rates do exhibit a consistent deviations from the proportional hazards assumption:

```
> round(ci.exp(mi)[,1, drop = FALSE], 3)
                                          exp(Est.)
(Intercept)                                   0.007
Ns(age, knots = c(50, 60, 70, 80))1           3.946
Ns(age, knots = c(50, 60, 70, 80))2          48.384
Ns(age, knots = c(50, 60, 70, 80))3           7.728
Ns(tfd, knots = c(0, 1, 5))1                  0.648
Ns(tfd, knots = c(0, 1, 5))2                  0.808
sexF                                          1.412
Ns(age, knots = c(50, 60, 70, 80))1:sexF      0.534
Ns(age, knots = c(50, 60, 70, 80))2:sexF      0.472
Ns(age, knots = c(50, 60, 70, 80))3:sexF      0.706
Ns(tfd, knots = c(0, 1, 5))1:sexF             1.254
Ns(tfd, knots = c(0, 1, 5))2:sexF             0.969
> par(mfrow = c(2,2))
> # main effects model
> matshade(pd$age, ci.pred(m1, mutate(pd, sex = "M")) * 1000,
+          plot = TRUE, ylim = c(2, 200),
+          xlab = "Attained age", ylab = "Mortality per 1000 PY",
+          col = "blue", lwd = 2, log = "y", alpha = .1)
> matshade(pd$age, ci.pred(m1, mutate(pd, sex = "F")) * 1000,
+          col = "red", lwd = 2, alpha = .1)
> matshade(pd$age, ci.pred(mi, mutate(pd, sex = "M")) * 1000,
+          plot = TRUE, ylim = c(2, 200),
+          xlab = "Attained age", ylab = "Mortality per 1000 PY",
+          col = "blue", lwd = 2, log = "y", alpha = .1)
> matshade(pd$age, ci.pred(mi, mutate(pd, sex = "F")) * 1000,
+          col = "red", lwd = 2, alpha = .1)
> # interaction model
> matshade(pd$tfd, ci.pred(m1, mutate(pd, sex = "M")) * 1000,
+          plot = TRUE, ylim = c(2, 200),
+          xlab = "Diabetes duration", ylab = "Mortality per 1000 PY",
+          col = "blue", lwd = 2, log = "y", alpha = .1)
> matshade(pd$tfd, ci.pred(m1, mutate(pd, sex = "F")) * 1000,
+          col = "red", lwd = 2, alpha = .1)
> matshade(pd$tfd, ci.pred(mi, mutate(pd, sex = "M")) * 1000,
+          plot = TRUE, ylim = c(2, 200),
+          xlab = "Diabetes duration", ylab = "Mortality per 1000 PY",
+          col = "blue", lwd = 2, log = "y", alpha = .1)
> matshade(pd$tfd, ci.pred(mi, mutate(pd, sex = "F")) * 1000,
+          col = "red", lwd = 2, alpha = .1)
```

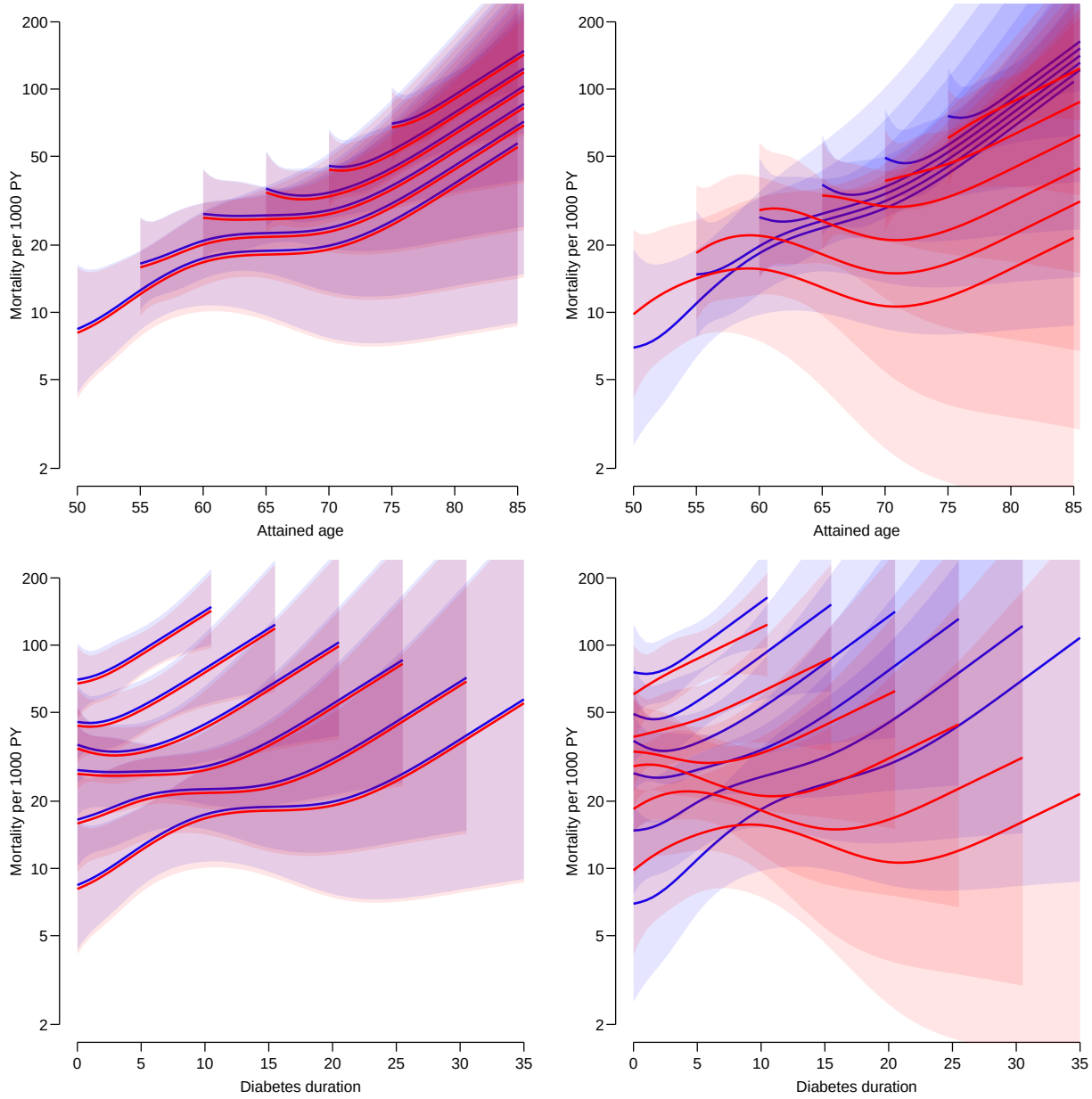

Figure 5.1: *Mortality rates for men and women diagnosed at ages 50, 55, 60,. . . ,75. The left plots is from the proportional hazards model, right plot is from a model with interactions between time since diagnosis (diabetes duration) and sex. The top plots are by attained age, the bottom plots by diabetes duration, but curves in top and bottom plots are the same, just differently aligned.*

`../graph/rmst-mi`

From figure 5.1 it is difficult to see and summarize the age and sex effects, so this is a case where a summary measure such as RMST might be useful. The code to get the quantities from the interaction model is precisely the same as for the proportional hazards model:

```
> simresi <- simres * NA # a quick way to get an array with same dimensions
> for (adx in ain)
+ for ( sx in sex)
```

```
+     {
+ set.seed(20250503)
+ cat(adx, sx, format(Sys.time(), format = "%T"), "\n")
+ simresi[paste(adx), sx, , ] <- ci.Crisk(list(Mort = mi),
+                                         nd = mutate(nd, sex = sx,
+                                                     age = adx + tfd),
+                                         nB = nB,
+                                         sim.res = "crisk")[, "Surv", ]
+     }
50 M 16:10:32
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
50 F 16:10:40
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 M 16:10:48
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 F 16:10:56
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 M 16:11:03
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 F 16:11:11
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1

> #
> # tidyverse
>    apply(simresi, c(1, 2, 4), get5.10) |> # bootstrap sample of 5 and 10 y RMST
+ apply(1:3, quantile, c(50, 5, 95)/100) |> # quantiles of these
+                 ftable(col.vars = 2:1) |> # nice formatting of resulting array
+                                   round(2)

            5              10
          50%   5%  95%  50%   5%  95%
adx sex
50  M    4.90 4.82 4.95 9.54 9.25 9.71
    F    4.86 4.75 4.92 9.39 9.00 9.62
60  M    4.69 4.56 4.78 8.76 8.36 9.07
    F    4.66 4.46 4.78 8.75 8.18 9.13
70  M    4.44 4.29 4.57 7.77 7.29 8.18
    F    4.52 4.38 4.63 8.09 7.62 8.49

> #
> # old fashioned: one at a time
> aa <- apply(simresi, c(1,2,4), get5.10)
> bb <- apply(aa, 1:3, quantile, c(50, 5, 95)/100)
> cc <- ftable(bb, col.vars = 2:1)
> round(cc, 2)

            5              10
          50%   5%  95%  50%   5%  95%
adx sex
50  M    4.90 4.82 4.95 9.54 9.25 9.71
    F    4.86 4.75 4.92 9.39 9.00 9.62
60  M    4.69 4.56 4.78 8.76 8.36 9.07
    F    4.66 4.46 4.78 8.75 8.18 9.13
70  M    4.44 4.29 4.57 7.77 7.29 8.18
    F    4.52 4.38 4.63 8.09 7.62 8.49
```

From this we see that the differences between men and women seem to be a bit larger when using the interaction model.

5.3.1 Exercise 20: Difference between sexes

If we want to assess the difference in RMST between men and women (with confidence limits) we will need simulated values of the differences, not separate simulations for men and women. Now, because we have the simulated survival functions from the *same* simulation streams we can also use these to derive confidence intervals for the difference in RMST between women and men.

```
> str(simres)
 num [1:3, 1:2, 1:151, 1:10000] 1 1 1 1 1 ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> simdif <- simres[,"F",,] - simres[,"M",,]
> str(simdif)
 num [1:3, 1:151, 1:10000] 0 0 0 0.000137 0.000365 ...
 - attr(*, "dimnames")=List of 3
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> dd <- apply(simdif, c(1,3), get5.10)
> str(dd)
 num [1:2, 1:3, 1:10000] 0.0194 0.0813 0.0492 0.1952 0.114 ...
 - attr(*, "dimnames")=List of 3
  ..$    : chr [1:2] "5" "10"
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> dd <- apply(dd, 1:2, quantile, c(50, 5, 95)/100)
> round(ftable(dd, col.vars = 2:1), 2)
         5                10
       50%    5%   95%   50%    5%   95%
adx
50    0.01 -0.02  0.03  0.03 -0.06  0.12
60    0.02 -0.04  0.07  0.06 -0.15  0.27
70    0.04 -0.09  0.16  0.13 -0.30  0.56
```

and we can do the same for the interaction model:

```
> simdifi <- simresi[,"F",,] - simresi[,"M",,]
> ii <- apply(simdifi, c(1,3), get5.10)
> ii <- apply(ii, 1:2, quantile, c(50, 5, 95)/100)
> #
> # main effects model (proportional hazards)
> round(ftable(dd, col.vars = 2:1), 2)
         5                10
       50%    5%   95%   50%    5%   95%
adx
50    0.01 -0.02  0.03  0.03 -0.06  0.12
60    0.02 -0.04  0.07  0.06 -0.15  0.27
70    0.04 -0.09  0.16  0.13 -0.30  0.56
> #
> # interaction model
> round(ftable(ii, col.vars = 2:1), 2)
```

```
        5                10
      50%    5%   95%   50%    5%   95%
adx
50   -0.04 -0.16  0.06 -0.15 -0.56  0.22
60   -0.03 -0.24  0.15 -0.02 -0.65  0.54
70    0.07 -0.11  0.27  0.31 -0.30  0.96
```

So we see there is a bit larger differences when using the interaction model but also wider confidence intervals—this is because we have a model with more parameters and this larger uncertainty in model predictions in the simulations

The main conclusion id that there is no difference between men and women in terms of RMST during 5 or 10 years when starting at ages 50, 60 or 70.

5.3.2 Exercise 21: Difference between ages

We can use the same machinery to see the differences between ages at diagnosis, based on the interaction model:

```
> str(simresi)
```

```
 num [1:3, 1:2, 1:151, 1:10000] 1 1 1 1 1 ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:3] "50" "60" "70"
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
```

```
> simadif <- simresi[2:3,,,] - simresi[1:2,,,]
> dimnames(simadif)[[1]] <- paste0(dimnames(simresi)[[1]][2:3], " - ",
+                                  dimnames(simresi)[[1]][1:2])
> str(simadif)
```

```
 num [1:2, 1:2, 1:151, 1:10000] 0 0 0 0 -0.00241 ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:2] "60 - 50" "70 - 60"
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:151] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
```

```
> aa <- apply(simadif, c(1,2,4), get5.10)
> bb <- apply(aa, 1:3, quantile, c(50, 5, 95)/100)
> round(ftable(bb, col.vars = 2:1), 2)
```

```
                5                10
              50%    5%   95%   50%    5%   95%
adx     sex
60 - 50 M   -0.21 -0.34 -0.10 -0.76 -1.13 -0.43
        F   -0.20 -0.37 -0.08 -0.63 -1.06 -0.28
70 - 60 M   -0.24 -0.41 -0.08 -0.98 -1.51 -0.47
        F   -0.13 -0.31  0.06 -0.64 -1.21 -0.04
```

Quantitatively the results are remarkable; the RMST are not very different between ages; 10 years later diagnosis of diabetes only incur less than .5 years decline in 5 years RMST and less than 1.5 years in 10 years RMST.

5.3.3 Exercise 22: Expanding the scope

Use your code to derive the RMST with horizons of 10, 20 and 30 years.

When the horizon for calculation of the RMST is expanded we will be relying on knowledge of the survival and hence the mortality rates till the horizon, in this example from ages 50, 60 and 70 and 20 or 30 years on, that is till age 90 or 100.

If we were relying on non-parametric models for the baseline, we would be dependent on data all the way to 100. With parametric models we will instead be relying on the validity of extrapolations beyond data. That is tenable in the case of mortality; it is well known that mortality increases exponentially (linearly on a log-scale that is) by age so predictions of mortality using natural splines will presumably be OK. This is not the case for incidence rates of chronic diseases as cancer or diabetes.

5.3.3.1 Repeating the code

For simplicity we make the predictions for horizons of 10, 20 and 30 years:

```
> nB <- 10000 # no of bootstrap samples
> nd <- data.frame(tfd = seq(0, 30, .1))
> (ain <- seq(50, 75, 5))
[1] 50 55 60 65 70 75
> sex <- c("M", "F")
> simres <- NArray(list(adx = ain,
+                       sex = sex,
+                       tfd = nd$tfd,
+                       sim = 1:nB))
> str(simres)
 logi [1:6, 1:2, 1:301, 1:10000] NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:6] "50" "55" "60" "65" ...
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:301] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> #
> # simulated cumulative risks
> for (adx in ain)
+ for ( sx in sex)
+     {
+ set.seed(20250503)
+ simres[paste(adx), sx, , ] <- ci.Crisk(list(Mort = m1),
+                                        nd = mutate(nd, sex = sx,
+                                                    age = adx + tfd),
+                                        nB = nB,
+                                        sim.res = "crisk")[, "Surv", ]
+ cat(adx, sx, format(Sys.time(), format = "%T"), "\n")
+     }
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
50 M 16:11:49
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
50 F 16:12:05
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
55 M 16:12:20
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
```

```
55 F 16:12:34
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 M 16:12:48
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
60 F 16:13:02
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
65 M 16:13:17
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
65 F 16:13:31
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 M 16:13:45
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
70 F 16:14:00
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
75 M 16:14:14
NOTE: Times are assumed to be in the column tfd at equal distances of 0.1
75 F 16:14:28
> #
> # function to derive integrals from function values 0.1 apart
> getRMST <-
+     function(x)
+         {
+         c("10" = sum(mid(x[1:101])),
+           "20" = sum(mid(x[1:201])),
+           "30" = sum(mid(x[1:301]))) / 10
+         }
> str(simres)
 num [1:6, 1:2, 1:301, 1:10000] 1 1 1 1 1 1 1 1 1 1 ...
 - attr(*, "dimnames")=List of 4
  ..$ adx: chr [1:6] "50" "55" "60" "65" ...
  ..$ sex: chr [1:2] "M" "F"
  ..$ tfd: chr [1:301] "0" "0.1" "0.2" "0.3" ...
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> #
> # compute integrals to chosen horizons
> aa <- apply(simres, c(1,2,4), getRMST)
> str(aa)
 num [1:3, 1:6, 1:2, 1:10000] 9.53 17.95 25.27 9.2 16.87 ...
 - attr(*, "dimnames")=List of 4
  ..$    : chr [1:3] "10" "20" "30"
  ..$ adx: chr [1:6] "50" "55" "60" "65" ...
  ..$ sex: chr [1:2] "M" "F"
  ..$ sim: chr [1:10000] "1" "2" "3" "4" ...
> #
> # confidence intervals for RMST
> bb <- apply(aa,
+             1:3,
+             quantile, c(50, 5, 95) / 100)
> str(bb)
 num [1:3, 1:3, 1:6, 1:2] 9.46 9.25 9.62 17.48 16.65 ...
 - attr(*, "dimnames")=List of 4
  ..$    : chr [1:3] "50%" "5%" "95%"
  ..$    : chr [1:3] "10" "20" "30"
  ..$ adx: chr [1:6] "50" "55" "60" "65" ...
  ..$ sex: chr [1:2] "M" "F"
```

```
> round(ftable(bb, col.vars = 2:1), 2)
            10                20                30
           50%    5%   95%   50%    5%   95%   50%    5%   95%
adx sex
50  M     9.46  9.25  9.62 17.48 16.65 18.12 23.92 21.44 25.66
    F     9.49  9.26  9.64 17.56 16.69 18.22 24.10 21.63 25.87
55  M     9.10  8.80  9.32 16.34 15.37 17.13 21.51 18.80 23.59
    F     9.13  8.81  9.36 16.45 15.43 17.28 21.76 19.03 23.89
60  M     8.75  8.42  9.01 15.17 14.14 16.03 18.82 16.11 21.26
    F     8.79  8.43  9.07 15.31 14.27 16.22 19.11 16.35 21.56
65  M     8.44  8.14  8.70 13.72 12.56 14.72 15.72 13.37 18.34
    F     8.50  8.19  8.76 13.89 12.76 14.90 16.02 13.65 18.66
70  M     7.87  7.47  8.22 11.42 10.25 12.60 12.12 10.47 14.35
    F     7.94  7.55  8.27 11.64 10.49 12.76 12.38 10.75 14.63
75  M     6.76  6.30  7.19  8.49  7.55  9.55  8.62  7.59 10.00
    F     6.86  6.42  7.25  8.70  7.79  9.69  8.84  7.84 10.22
```

So we see that the RMST over the next 20 years for a 60 year old man is 15.17 years, with a 90% confidence interval of (14.14 ; 16.03).

Here is a simple graphical overview of the table printed above:

```
> par(xaxt = "n")
>  plotEst(t(bb["10",,"M"]), y=6:1, col = "blue",
+          xlim = c(0,30), xaxt = "n", grid = seq(5, 30, 5),
+          xlab = "RMST the next 10, 20 or 30 years")
> linesEst(t(bb["20",,"M"]), y=6:1 - 0.1, col = "blue")
> linesEst(t(bb["30",,"M"]), y=6:1 - 0.2, col = "blue")
> linesEst(t(bb["10",,"F"]), y=6:1 - 0.3, col = "red")
> linesEst(t(bb["20",,"F"]), y=6:1 - 0.4, col = "red")
> linesEst(t(bb["30",,"F"]), y=6:1 - 0.5, col = "red")
> par(xaxt = "s")
> axis(side = 1, at = seq(5, 25, 5))
> axis(side = 1, at = 0:30, labels = NA, tcl = -0.3)
```

From figure 5.2 it is clear that there is not much difference between men and women and that age at diagnosis (here also age at start of the interval used to compute RMST) is a major driver. Needless to say, the *length* of the interval is of course the main determinant at younger ages.

The simulation is over the posterior distribution of the parameters in the mortality model, so not taking into account the possible misfit of the model, only the rate variation assuming the model to be correct.

```
  Start time: 2025-05-10, 16:09:35
    End time: 2025-05-10, 16:14:43
Elapsed time: 5.14 minutes
```

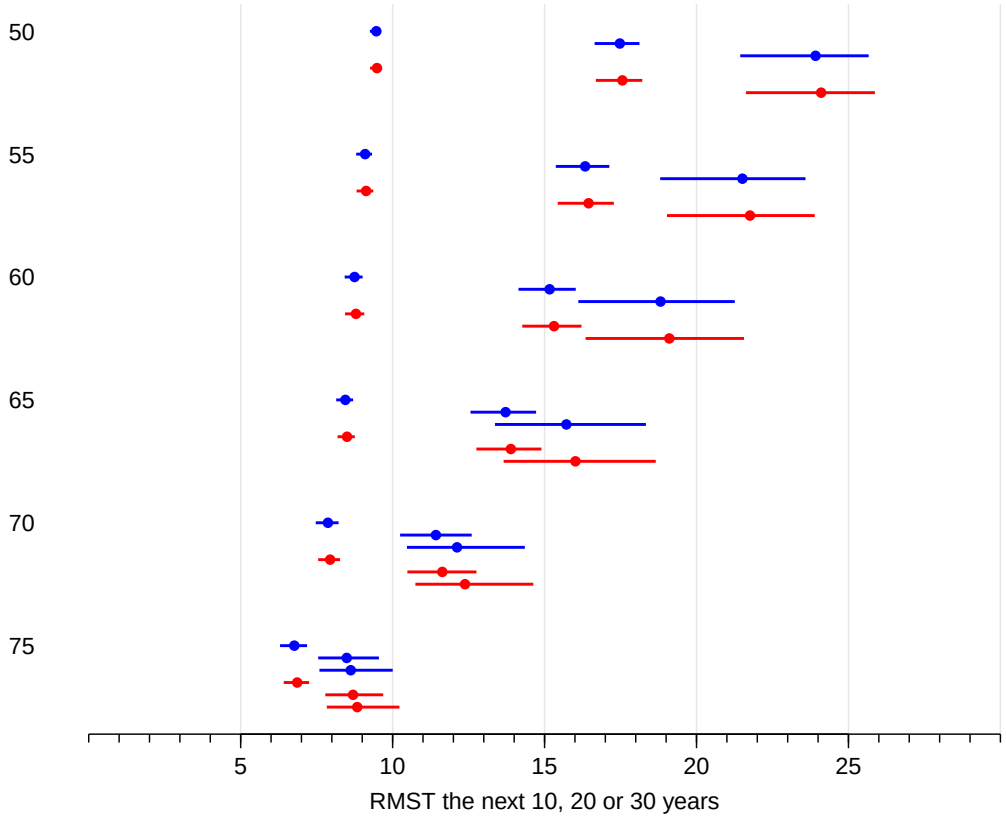

Figure 5.2: *Residual mean survival time during 10, 20 and 30 years (top to bottom within each age at diagnosis) for Danish diabetes patients diagnosed at ages 50, 55,...,75 years; men in blue women in red .* `../graph/rmst-forest`

References

[1] P. K. Andersen, R. B. Geskus, T. de Witte, and H. Putter. Competing risks in epidemiology: possibilities and pitfalls. *Int J Epidemiol*, Jan 2012.

[2] P. K. Andersen and N. Keiding. Interpretability and importance of functionals in competing risks and multistate models. *Stat Med*, 31:1074–1088, 2012.

[3] Bendix Carstensen. *Epidemiology with R*. Number ISBN: 978-0-19-884133-3. Oxford University Press, 2021.

[4] Bendix Carstensen. Practical multistate modeling with r and epi::lexis. Technical report, https://bendixcarstensen.com/MSbook.pdf, 2025.