Competing risks with
`Lexis`, parametric rates and
simulation based confidence intervals

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

Chapter 1

Competing risks

1.1 Technicalities

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

```
> options(width = 90,
+         show.signif.stars = FALSE,
+         SweaveHooks=list(fig = function()
+                          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)
> clear()


     R   Epi   popEpi
 4.4.2  2.59   0.4.12
```

1.2 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:

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

...formally, the limit of this quantity as $h \to 0$.

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.

1

1.2.1 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.

1.2.2 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 is required for the calculations these quantities.

1.2.3 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* 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.

1.2.4 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. Though not a formal mathematical requirement this implies that we should have data starting at time 0.

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 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.

1.3 Rates and cumulative risks

Each of the cumulative risks is a function of the survival function which in turn depends on *all* rates. Specifically, 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:

$$S(t) = \exp\left(-\int_0^t \sum_c \lambda_c(s)\,\mathrm{d}s\right) = \exp\left(-\sum_c \Lambda_c(t)\right) \tag{1.1}$$

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:

$$R_k(t) = \int_{u=0}^{u=t} \lambda_k(u)S(u)\,\mathrm{d}u = \int_{u=0}^{u=t} \lambda_k(u)\exp\left(-\sum_j \Lambda_j(u)\right)\mathrm{d}u \tag{1.2}$$

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 we get 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 cause k at t.

Parametric models for the cause-specific rates can produce estimated transition rates λ_c at closely spaced intervals, 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:

$$1 = S(t) + R_1(t) + R_2(t) + \cdots, \quad \forall t$$

1.3.1   Confidence intervals by simulation

But even if we from the modeling of the λ_cs may have standard errors of $\log\left(\lambda_c(t)\right)$, the standard errors of the R_cs will be analytically intractable from these.

In practice, the only viable way to get confidence intervals for the cumulative risks, R_c, is therefore by calculation of a set of rates $\lambda_c(t)$ by sampling from the posterior distribution of the parameters in the models for $\log\left(\lambda_c(t)\right)$, and then compute the integrals numerically for each simulated sample, according to formulae 1.1 and 1.2. This will produce a so-called 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.

1.3.2   Subdistribution hazard

A common concept seen in competing risks is the subdistribution hazard, and proportional hazards models for this (the Fine-Gray model).

Suppose for a moment we only consider all-cause mortality, $\lambda(t)$. Then the cumulative risk of death is:

$$R(t) = 1 - S(t) = 1 - \exp\left(-\int_0^t \lambda(s)\,\mathrm{d}s\right)$$

Solving this for λ will yield:

$$\lambda(t) = -\frac{\mathrm{d}\log\big(1 - R(t)\big)}{\mathrm{d}t}$$

In the case of multiple causes of death we can define the *subdistribution hazard* for cause c by using the same transformation of the cumulative risk for cause c:

$$\tilde{\lambda}_c(t) = -\frac{\mathrm{d}\log\big(1 - R_c(t)\big)}{\mathrm{d}t}$$

or, for the cause-specific risk:

$$R_c(t) = 1 - \exp\left(-\int_0^t \tilde{\lambda}_c(s)\,\mathrm{d}s\right)$$

Thus, the subdistribution hazard $\tilde{\lambda}_c(t)$ is a function which, when subjected to the (hazard→risk) function from the all-cause mortality case, yields the cause-specific risk.

The Fine-Gray model is a model for the subdistribution hazard $\tilde{\lambda}_c$. It is only a model for *one* cause-specific hazard. Of course it can be applied to all available causes in turn, but the sum of the cumulative risks derived from the models may exceed 1...

Unlike a cause-specific hazard, which can depend on multiple time scales, the subdistribution hazard can only depend on one, since it requires an origin—just like cumulative risks.

But the interpretation of a subdistribution hazard is difficult. I have yet to see one that goes beyond the mathematical formalism above. Therefore the subdistribution is not included in this vignette.

Chapter 2

Example data

2.1 A `Lexis` object

As an illustrative data example we use the (fake) diabetes register data; we set up the Lexis object, an then cut the follow-up time at dates of `OAD` and `Ins` using `mcutLexis`:

```
> data(DMlate)
> Ldm <- Lexis(entry = list(per = dodm,
+                            age = dodm-dobth,
+                            tfd = 0),
+               exit = list(per = dox),
+        exit.status = factor(!is.na(dodth), labels = c("DM", "Dead")),
+               data = DMlate)
NOTE: entry.status has been set to "DM" for all.
NOTE: Dropping  4  rows with duration of follow up < tol
> summary(Ldm, t = T)
Transitions:
     To
From   DM Dead  Records:  Events: Risk time:  Persons:
  DM 7497 2499      9996     2499   54273.27      9996


Timescales:
per age tfd
 ""  ""  ""
> set.seed(1952)
> Mdm <- mcutLexis(Ldm,
+                  wh = c('dooad','doins'),
+           new.states = c('OAD','Ins'),
+           seq.states = FALSE,
+                 ties = TRUE)
NOTE: Precursor states set to DM
NOTE: 15 records with tied events times resolved (adding 0.01 random uniform),
      so results are only reproducible if the random number seed was set.
> summary(Mdm)
Transitions:
     To
From       DM Dead  OAD  Ins Ins+OAD  Records:  Events: Risk time:  Persons:
  DM     2830 1056 2957  689       0      7532     4702   22920.38      7532
  OAD       0  992 3327    0    1005      5324     1997   22965.24      5324
  Ins       0  152    0  462     172       786      324    3883.06       786
```

```
  Ins+OAD     0  299     0     0      878        1177       299     4504.58        1177
  Sum      2830 2499 6284 1151     2055       14819      7322    54273.27        9996
```

before drug inception) in intervals of 1/12 year, creating a `Lexis` object for a competing risks situation with three possible event types:

```
> Sdm <- splitLexis(factorize(subset(Mdm,
+                                  lex.Cst == "DM")),
+                time.scale = "tfd",
+                   breaks = seq(0, 20, 1/12))
> summary(Sdm)
Transitions:
     To
From      DM Dead  OAD Ins  Records:  Events:  Risk time:  Persons:
  DM 274263 1056 2957 689    278965     4702    22920.38      7532
```

We can illustrate the follow-up in the full data frame and in the restricted data frame

```
> boxes(Mdm, boxpos = list(x = c(15, 50, 15, 85, 85),
+                          y = c(85, 50, 15, 85, 15)),
+          scale.R = 100,
+          show.BE = TRUE)
```
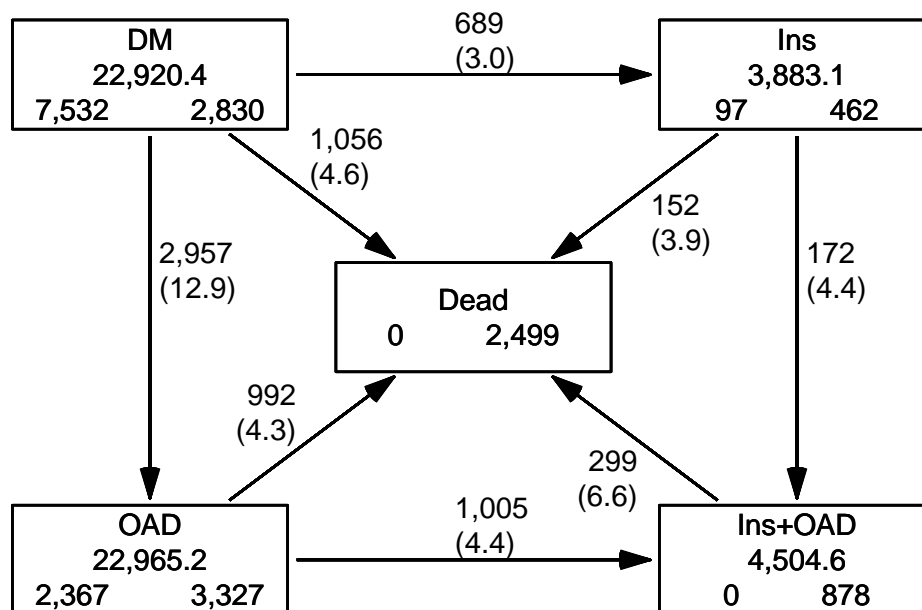

Figure 2.1: *The transitions in the multistate model, where follow-up is extended also after beginning of first drug exposure. Rates in brackets are per 100 PY.*   ./crisk-boxes5

```
> boxes(Relevel(Sdm, 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 )
```
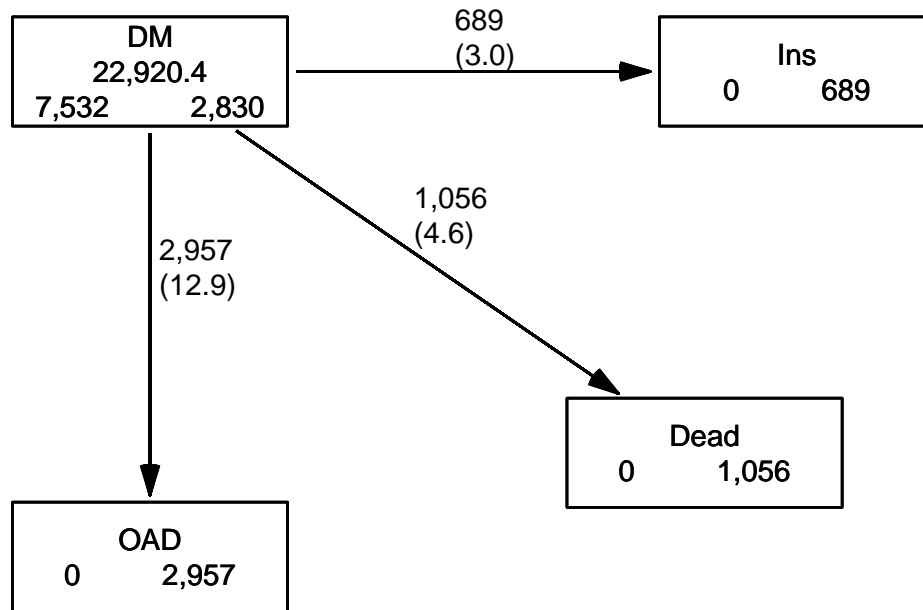
Figure 2.2: *The transitions in the competing risks model, where follow-up is stopped at first drug exposure. By that token only the* DM *state has person-years; a characteristic of a competing risks situation.* ./crisk-boxes4

2.2   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):

```
> mD <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'Dead')
mgcv::gam Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->Dead

> mO <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'OAD' )
mgcv::gam Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->OAD

> mI <- gamLexis(Sdm, ~ s(tfd, k = 5), to = 'Ins' )
mgcv::gam Poisson analysis of Lexis object Sdm with log link:
Rates for the transition:
DM->Ins
```

We see that `gamLexis` (just like `glmLexis` would) 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 (in real life a smaller interval should be used, say 1/50 or 1/100 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 ...
```

With this 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 2.3 is not normally shown in analyses of competing risks; the competing cause-specific *rates* are hardly ever shown. I suspect that this is frequently because they are often modeled by a Cox model and so are buried in the model and hard to get at.

Since we will be integrating the rates, it would also 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.

```
> 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)
```

2.3   Cumulative rates and risks

For the calculation of the cumulative rates and state probabilities, we need just the estimated rates (without CIs). The formulae 1.1 and 1.2 on page 3 are transformed to R-code; starting with the rates, λ_D as `lD` etc:
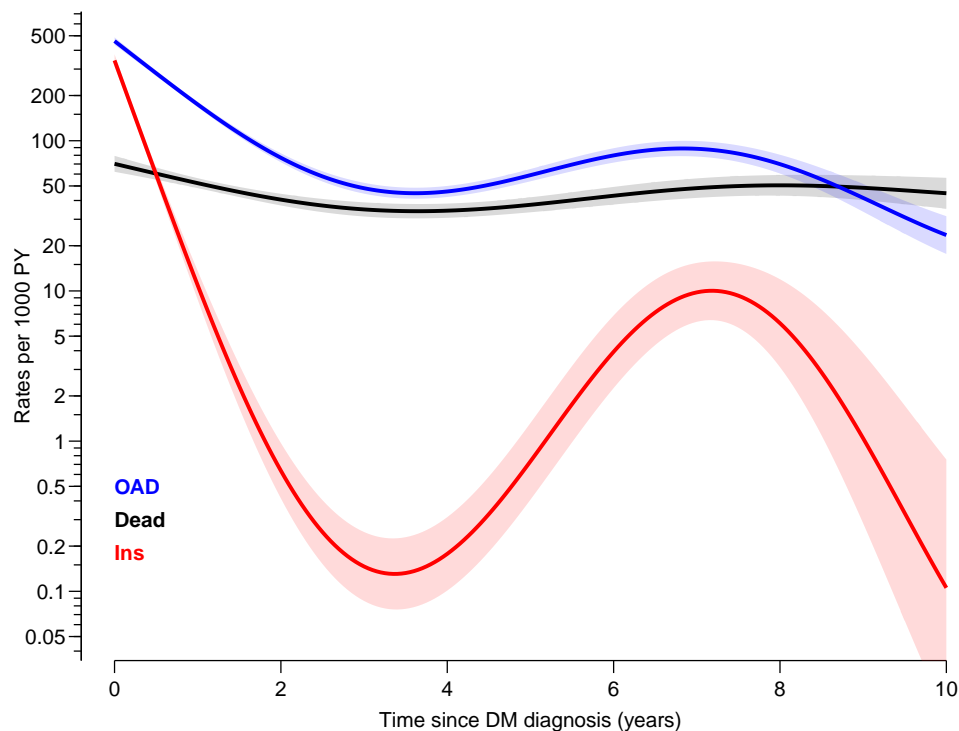
Figure 2.3: *Estimated rates from the* `DM` *state, estimates are from* `gam` *models fitted to data split in 1 month intervals (1/12 year, that is). Rates of* `OAD` *is in the vicinity of 0.1/year, and mortality about half of this. Rates of insulin start among persons on no other drug are beginning high, then decreasing with a nadir at about 4 years and then increase to a peak at 8 years.*   `./crisk-rates`

```
> # utility function to compute midpoints between sucessive values in a vector
> mp <- function(x) x[-1] - diff(x) / 2
> #
> int <- 1 / 20
> # rates at midpoints of intervals
> lD <- mp(ci.pred(mD, nd)[, 1])
> lI <- mp(ci.pred(mI, nd)[, 1])
> l0 <- mp(ci.pred(m0, nd)[, 1])
> #
> # cumulative rates and survival function at right border of the intervals
> LD <- cumsum(lD) * int
> LI <- cumsum(lI) * int
> L0 <- cumsum(l0) * int
> # survival function, formula (1.1)
> Sv <- exp(- LD - LI - L0)
> #
> # 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)
> r0 <- c(0, cumsum(l0 * mp(Sv)) * int)
```

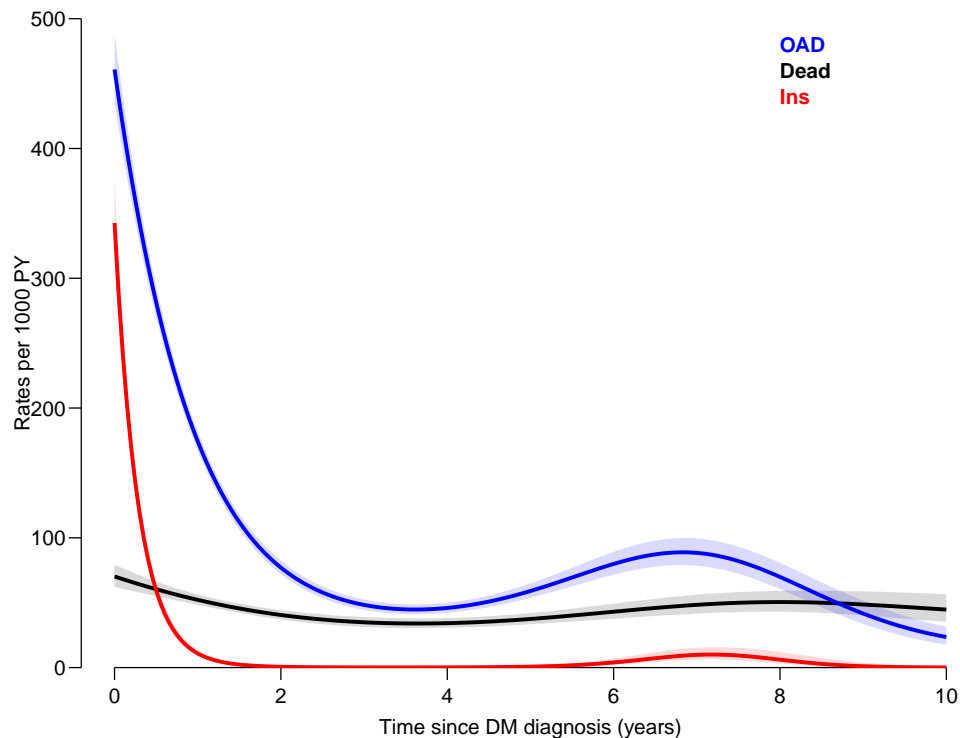Now we have the cumulative risks for the three causes and the survival, computed at the

Figure 2.4: *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. .*   ./crisk-rates-l

end of each of the intervals. At any time point the sum of the 3 cumulative risks and the survival should be 1:

```
> 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. 1.0000252933615676465
Median  1.0000254405444473083
Mean    1.0000249140597869957
3rd Qu. 1.0000258488927371570
Max.    1.0000260207157360970
> options(oo)
```

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

We can then plot the 3 cumulative risk functions stacked together using `mat2pol` (`matrix` to `polygons`):

```
> 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)
```
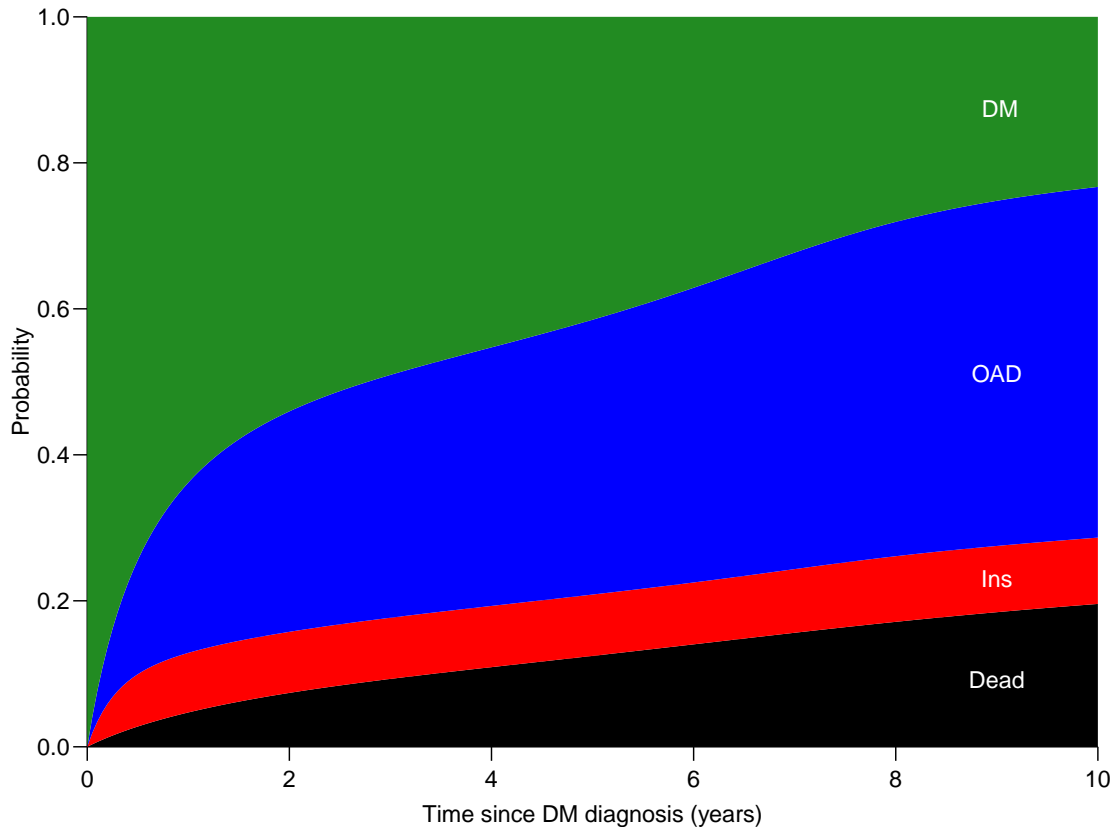
Figure 2.5: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`*. We are not concerned about what occurs after any these events.* `Dead` *means dead without initiating any of the two drugs.* ./crisk-stack

2.4   Sojourn times

The sojourn times in each of the states is just the area of each of the coloured parts of figure 2.5. 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
 4.3486390 1.2071800 0.8396038 3.6548276 10.0502504
```

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

This was a demonstration on how to compute the rates, cumulative risks and sojourn times. But no confidence intervals.

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 colours in figure 2.5. 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 simulated quantities.

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    0.3425874 0.3111484 0.3772030 0.3425874 0.3111484 0.3772030
0.05 0.2874359 0.2630855 0.3140401 0.2874359 0.2630855 0.3140401
0.1  0.2411669 0.2221042 0.2618657 0.2411669 0.2221042 0.2618657
0.15 0.2023535 0.1871526 0.2187890 0.2023535 0.1871526 0.2187890
```

```
0.2  0.1697982 0.1573575 0.1832226 0.1697982 0.1573575 0.1832226
0.25 0.1424958 0.1319999 0.1538263 0.1424958 0.1319999 0.1538263
```

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] -1.04 -1.23 -1.41 -1.6 -1.79 ...
 - 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    0.3425874 0.3111484 0.3772030 0.3596667 0.3664201 0.3287678 0.3524361
0.05 0.2874359 0.2630855 0.3140401 0.2986143 0.3035827 0.2774065 0.2910810
0.1  0.2411669 0.2221042 0.2618657 0.2479294 0.2515260 0.2340728 0.2404116
0.15 0.2023535 0.1871526 0.2187890 0.2058554 0.2084048 0.1975157 0.1985709
0.2  0.1697982 0.1573575 0.1832226 0.1709334 0.1726899 0.1666793 0.1640248
0.25 0.1424958 0.1319999 0.1538263 0.1419513 0.1431134 0.1406722 0.1355053
```

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 cauese 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 risk`s) 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:

```
> res <- ci.Crisk(list(OAD = mO,
+                      Ins = mI,
+                     Dead = mD),
+                        nd = data.frame(tfd = seq(0, 10, 1/20)),
+                        nB = 500,
+                      perm = 4:1)
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05
> str(res)
List of 4
 $ Crisk: num [1:201, 1:4, 1:3] 1 0.959 0.923 0.892 0.864 ...
  ..- 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 0.00341 0.00664 0.00971 0.01264 ...
  ..- 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 0.049 0.096 0.141 0.185 ...
  ..- 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'`):

```
> rsm <- ci.Crisk(list(OAD = mO,
+                      Ins = mI,
+                     Dead = mD),
+                         nd = data.frame(tfd = seq(0, 10, 1/20)),
+                         nB = 500,
+                      sim.res = 'rates')
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05

> str(rsm)
 num [1:201, 1:3, 1:500] 0.464 0.442 0.422 0.402 0.383 ...
 - 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 [1:500] "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 500 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'`:

```
> csm <- ci.Crisk(list(OAD = mO,
+                      Ins = mI,
+                     Dead = mD),
+                         nd = data.frame(tfd = seq(0, 10, 1/20)),
+                         nB = 500,
+                      sim.res = 'crisk')
NOTE: Times are assumed to be in the column tfd at equal distances of 0.05

> str(csm)
 num [1:201, 1:4, 1:500] 1 0.959 0.923 0.891 0.862 ...
 - 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 [1:500] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.05
```

These are 500 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 2.3 we showed the rates with confidence intervals from the model. But in `rsm` we have 500 parametric bootstrap samples of the occurrence rates, so we can derive the bootstrap medians and the bootstrap c.i.s:

```
> Brates <- aperm(apply(rsm,
+                       1:2,
+                       quantile,
+                       probs = c(.5, .025, .975)),
+              c(2, 3, 1))
> str(Brates)
 num [1:201, 1:3, 1:3] 0.462 0.439 0.418 0.398 0.379 ...
 - 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%"
```

(`aperm` permutes the dimensions of the array). 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:
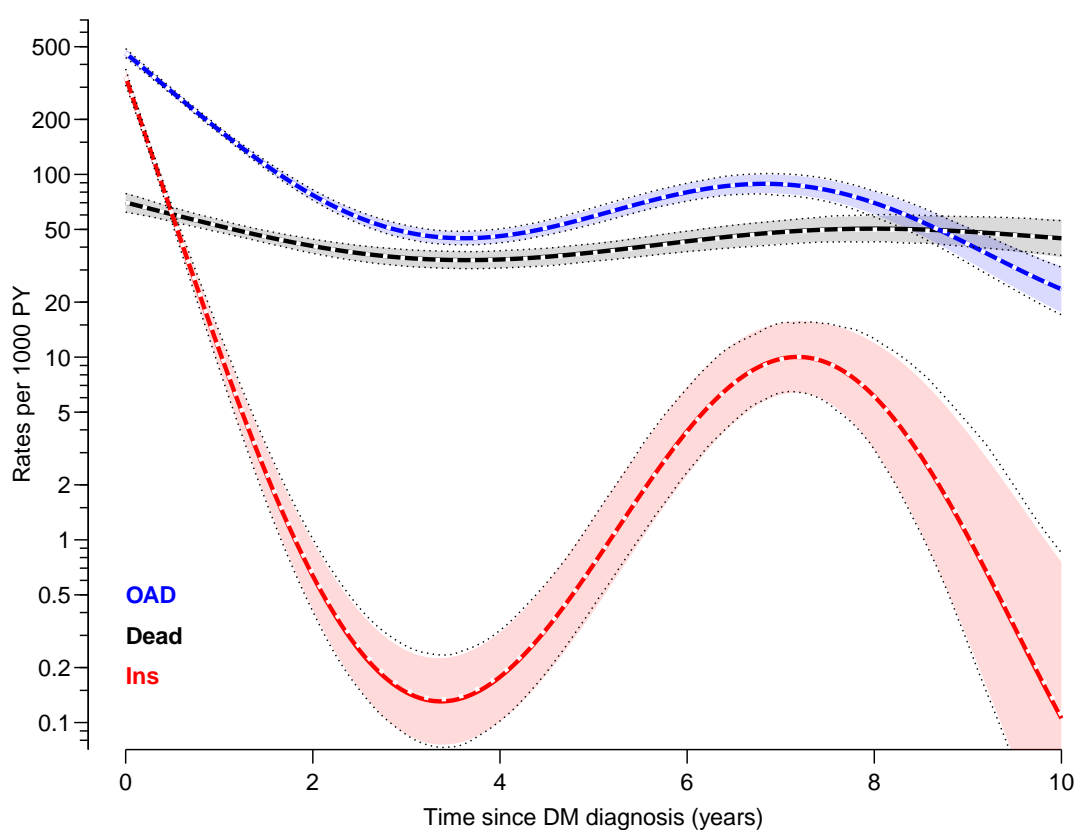

Figure 3.1: *Estimated rates from the* `DM` *state, estimates are from* `gam` *models fitted to data
split in 1 month intervals (1/12 year, that is). The white dotted curves are the bootstrap
medians, black dotted curves are the bootstrap 95% c.i.s.*   `./crisk-rates-ci`

```
> 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)
```

3.4   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:
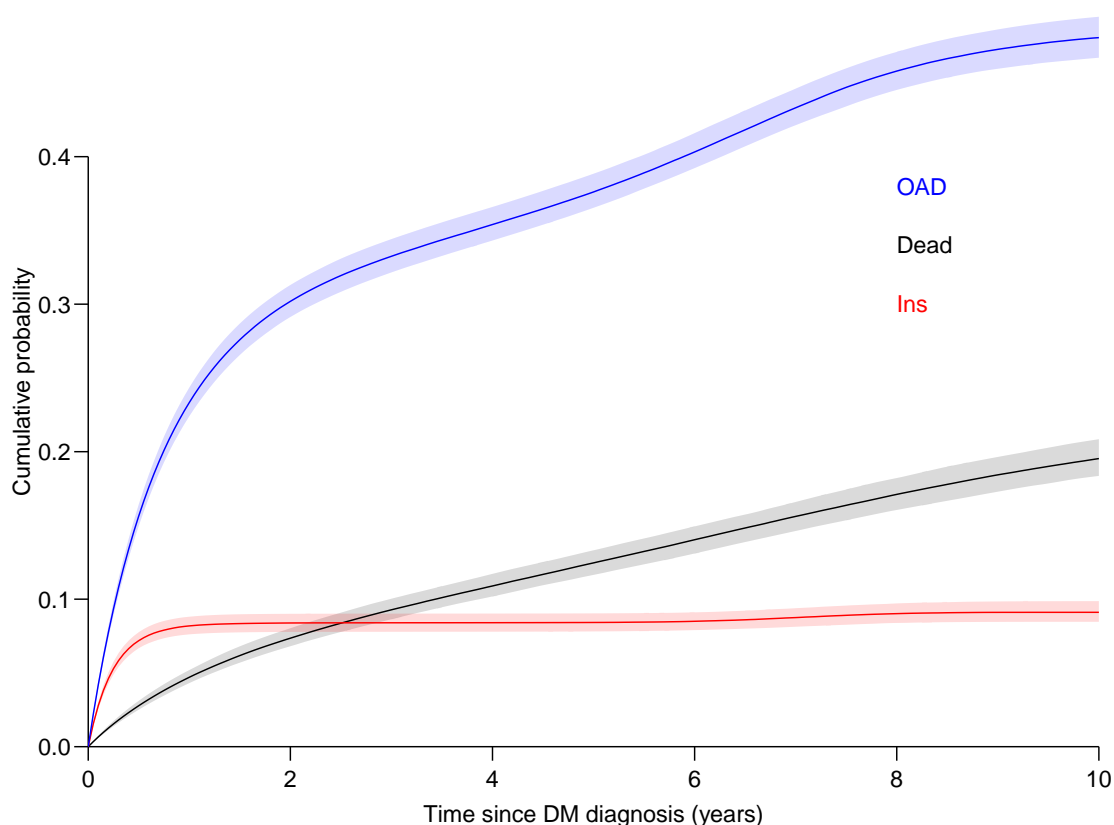

Figure 3.2: *Cumulative risks for the three types of events, with 95% bootstrap-based confidence intervals as shades.*   `./crisk-crates`

```
> 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)
```

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$Crisk)
 num [1:201, 1:4, 1:3] 1 0.959 0.923 0.892 0.864 ...
 - attr(*, "dimnames")=List of 3
  ..$ 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$Srisk)
 num [1:201, 1:3, 1:3] 0 0.00341 0.00664 0.00971 0.01264 ...
 - attr(*, "dimnames")=List of 3
  ..$ 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` (`matrix` to `polygon`), 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   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:
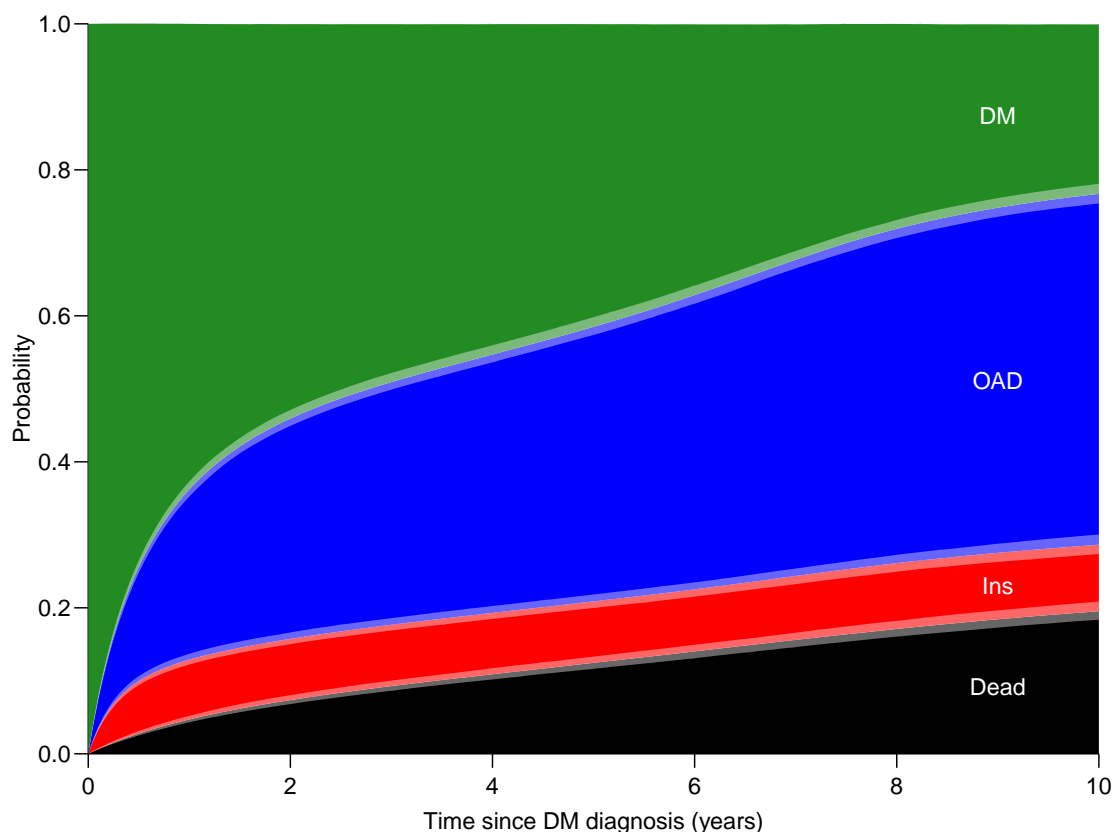
Figure 3.3: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`. *We are not concerned about what occurs after these events.* `Dead` *means dead without being on any drug. The white shadings around the borders between coloured areas represent the 95% confidence intervals for the (sum of) probabilities.* `./crisk-stack-ci`

```
> 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   2.77 2.72  2.82
       OAD    1.44 1.39  1.49
       Ins    0.40 0.37  0.43
       Dead   0.39 0.36  0.42
10 yr  Surv   4.31 4.21  4.41
       OAD    3.64 3.54  3.75
       Ins    0.84 0.78  0.90
       Dead   1.20 1.13  1.27
```

So we see that the expected life lived without pharmaceutical treatment during the first 10 years after DM diagnosis is 4.31 years with a 95% CI of (4.21; 4.41), and during the first 5 years 2.77 (2.72; 2.82).

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 Data

For illustration we simulate some causes of death in the `DMlate` data set; first we sample numbers 1, 2, 3 representing 3 different causes of death in `DMlate`:

```
> data(DMlate)
> set.seed(7465)
> wh <- sample(1:3, nrow(DMlate), replace = T, prob = c(4, 2, 6))
```

Those not dead are changed to 0:

```
> wh[is.na(DMlate$dodth)] <- 0
```

Define a factor in `DMlate` defining exit status as either alive or one of the three causes of death, and check by a `table` that all dead have a cause:

```
> DMlate$codth <- factor(wh, labels = c("Alive", "CVD", "Can", "Oth"))
> with(DMlate, table(codth, isDead = !is.na(dodth)))
        isDead
codth    FALSE TRUE
  Alive   7497    0
  CVD        0  815
  Can        0  401
  Oth        0 1287
```

It is important that the `"Alive"` state is the `first` level if the factor `codth`; the `Lexis` function will assign this the all persons at start of follow-up.

   `DMlate` now looks like a typical data set with cause of death in a separate variable; in this case we also added a state, `Alive`, for those without a recorded death:

```
> str(DMlate)
'data.frame':          10000 obs. of  8 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 ...
 $ codth: Factor w/ 4 levels "Alive","CVD",..: 1 1 1 1 1 4 1 1 4 1 ...
```

```
> head(DMlate, 12)
       sex    dobth     dodm     dodth    dooad doins      dox codth
50185    F 1940.256 1998.917       NA       NA    NA 2009.997 Alive
307563   M 1939.218 2003.309       NA 2007.446    NA 2009.997 Alive
294104   F 1918.301 2004.552       NA       NA    NA 2009.997 Alive
336439   F 1965.225 2009.261       NA       NA    NA 2009.997 Alive
245651   M 1932.877 2008.653       NA       NA    NA 2009.997 Alive
216824   F 1927.870 2007.886 2009.923       NA    NA 2009.923   Oth
24969    M 1946.498 2007.216       NA 2007.248    NA 2009.997 Alive
325465   M 1940.475 2006.674       NA 2006.674    NA 2009.997 Alive
430048   F 1937.083 1998.956 2008.464       NA    NA 2008.464   Oth
362980   M 1933.154 2009.784       NA       NA    NA 2009.997 Alive
279976   F 1928.300 2000.949       NA       NA    NA 2009.997 Alive
279271   M 1951.213 2008.114       NA 2009.743    NA 2009.997 Alive
```

4.2   A `Lexis` object with 3 causes of death

With cause of death in a separate variable it is easy to set up a `Lexis` object:

```
> dmL <- Lexis(entry = list(per = dodm,
+                           age = dodm - dobth,
+                           tfD = 0),
+               exit = list(per = dox),
+         exit.status = codth,
+                data = DMlate)
NOTE: entry.status has been set to "Alive" for all.
NOTE: Dropping  4  rows with duration of follow up < tol
> summary(dmL, t = T)
Transitions:
     To
From    Alive CVD Can  Oth   Records:  Events: Risk time:  Persons:
  Alive  7497 814 400 1285       9996     2499   54273.27      9996

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

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

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

4.3   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     Alive CVD Can  Oth  Records:  Events: Risk time:  Persons:
  Alive 115974 814 400 1285     118473     2499   54273.27      9996
```
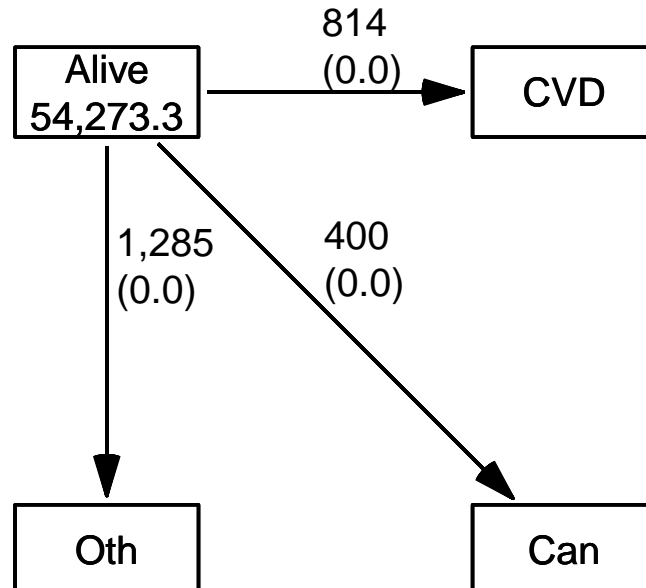
Figure 4.1: *Transitions from live to different causes of death. You probably want to explore the other arguments to* `boxes`. `./crisk-boxes`

```
> mCVD <- gamLexis(sL, ~ s(tfD, by=sex), to = "CVD")
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
Alive->CVD
> mCan <- gamLexis(sL, ~ s(tfD, by=sex), to = "Can")
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
Alive->Can
> mOth <- gamLexis(sL, ~ s(tfD, by=sex), to = "Oth")
mgcv::gam Poisson analysis of Lexis object sL with log link:
Rates for the transition:
Alive->Oth
```

4.4   Derived measures

With these three models for the occurrence rates we can compute the cumulative risks of dying from 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, 15, 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`:

```
> cR <- ci.Crisk(list(CVD = mCVD,
+                     Can = mCan,
+                   Other = mOth),
+                nB = 500,
+                nd = nm)
NOTE: Times are assumed to be in the column tfD at equal distances of 0.05
> str(cR)
List of 4
 $ Crisk: num [1:301, 1:4, 1:3] 1 0.997 0.993 0.99 0.987 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:301] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "CVD" "Can" "Other"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:301, 1:3, 1:3] 0 0.00179 0.00353 0.00523 0.00688 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:301] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:3] "Other" "Other+Can" "Other+Can+CVD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:301, 1:4, 1:3] 0 0.0499 0.0997 0.1492 0.1987 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ tfD  : chr [1:301] "0" "0.05" "0.1" "0.15" ...
  .. ..$ cause: chr [1:4] "Surv" "CVD" "Can" "Other"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:301] 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.

4.4.1   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("black", "orange", "limegreen")
> matshade(cR$time, cbind(cR$Crisk[, "CVD"  , ],
+                         cR$Crisk[, "Can"  , ],
+                         cR$Crisk[, "Other", ]),
+         col = clr, lty = 1, lwd = 2,
+         plot = TRUE, ylim = c(0, 1/3), yaxs = "i")
> text(0, 1/3 - c(2,3,1)/30, c("CVD", "Can", "Oth"),
+     col = clr, adj = 0, font = 2)
```

We also have the stacked probabilities so we can show how the population is distributed across the states at any one time:
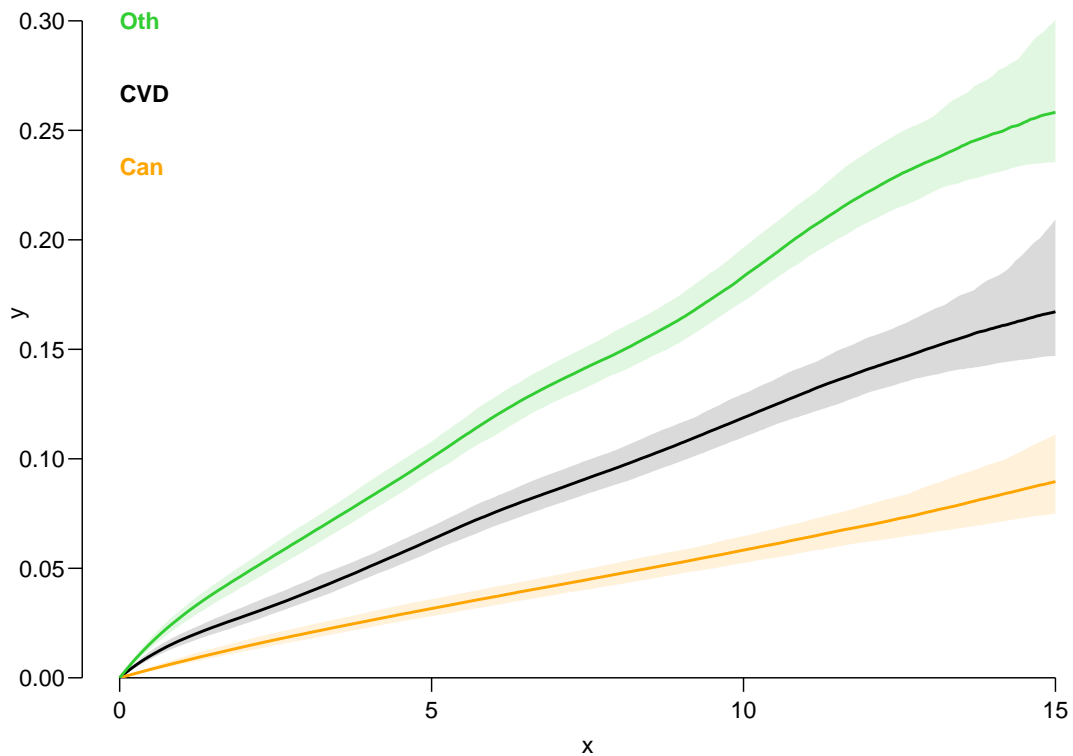
Figure 4.2: *Cumulative risks of each cause of death based on* `gam` *models for the cause-specific rates.*   `./crisk-cR`
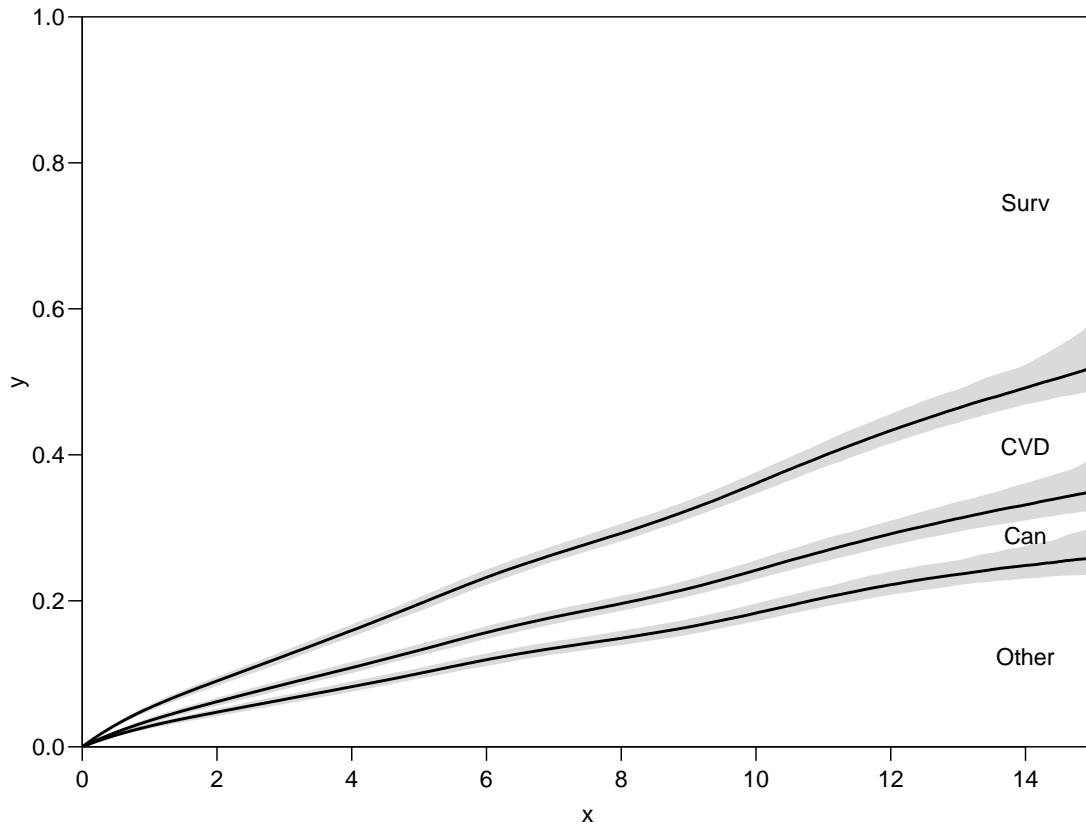
4.4.2   Stacked cumulative risks

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(14, mp(c(0, cR$Srisk["14", , 1], 1)),
+       rev(c(dimnames(cR$Crisk)[[2]])))
> box(bty = "o")
```

It is not a good idea to color these curves, they do not refer to the causes of death, 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` (matrix to `polygons`) using the `Crisk` component. We can then superpose the confidence intervals for the sum of the state probabilities using `matshade` by adding white

Figure 4.3: *Stacked cumulative risks.* `./crisk-Sr1`

shades:

```
> zz <- mat2pol(cR$Crisk[, c("Other", "Can", "CVD", "Surv"), "50%"],
+               x = cR$time,
+           xlim = c(0,15), xaxs = "i", yaxs = "i", las = 1,
+           xlab = "Time since DM diagnosis (years)",
+           ylab = "Probability",
+             col =  c("gray", "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(14, mp(c(0, cR$Srisk["14", , 1], 1)),
+     rev(c(dimnames(cR$Crisk)[[2]])), col = "white")
```

4.4.3 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(1:5 * 3), , ], 1), row.vars = 1)
    cause Surv            CVD             Can             Other
          50% 2.5% 97.5%  50% 2.5% 97.5%  50% 2.5% 97.5%   50% 2.5% 97.5%
tfD
3         2.8  2.8   2.8  0.1  0.1   0.1  0.0  0.0   0.0   0.1  0.1   0.1
6         5.3  5.2   5.3  0.2  0.2   0.3  0.1  0.1   0.1   0.4  0.4   0.4
```
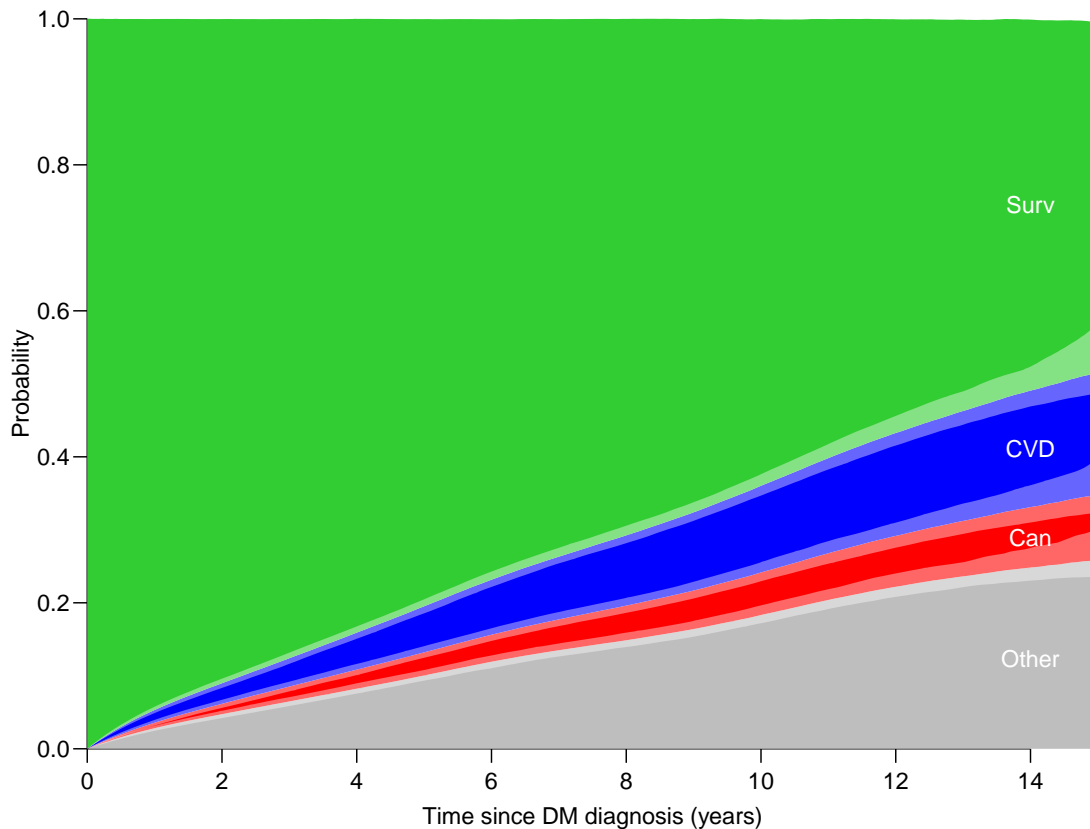
Figure 4.4: *Stacked cumulative risks with coloring of states and overlaid with confidence intervals for the probabilities shown; that is the relevant sums.*   `./crisk-Sr2`

```
9               7.4  7.4   7.5  0.5  0.5   0.6  0.3  0.2   0.3  0.8  0.8   0.9
12              9.3  9.2   9.4  0.9  0.8   0.9  0.4  0.4   0.5  1.4  1.3   1.5
15             10.9 10.7  11.0  1.3  1.3   1.5  0.7  0.6   0.8  2.1  2.0   2.2
```

The sojourn times in the three dead states can be taken as the years of life lost to each of the causes, the sum of the medians for the three causes equals the time frame (5, 10, 15) minus the `Surv` component.

So we see that during the first 15 years after diagnosis of diabetes, the expected years alive is 10.9 years. The distribution of lifetime lost between the causes is bogus in this case as the causes of death were randomly generated.

4.4.4 Comparing groups

Finally, we may want to see the *difference* (or ratio) of survival 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 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, 15, 1/20), sex = "M")
> nw <- data.frame(tfD = seq(0, 15, 1/20), sex = "F")
```

```
> # set the seed
> set.seed(1952)
> mR <- ci.Crisk(list(CVD = mCVD,
+                         Can = mCan,
+                       Other = mOth),
+                  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
> set.seed(1952)
> wR <- ci.Crisk(list(CVD = mCVD,
+                         Can = mCan,
+                       Other = mOth),
+                  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 [1:301, 1:4, 1:500] 1 0.997 0.994 0.991 0.988 ...
 - attr(*, "dimnames")=List of 3
  ..$ tfD  : chr [1:301] "0" "0.05" "0.1" "0.15" ...
  ..$ cause: chr [1:4] "Surv" "CVD" "Can" "Other"
  ..$ 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, 1:301] 0 0 0 -0.0334 -0.0869 ...
 - attr(*, "dimnames")=List of 2
  ..$    : chr [1:3] "50%" "2.5%" "97.5%"
  ..$ tfD: chr [1:301] "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 `applyed` 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(-5, 5),
+          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)
```
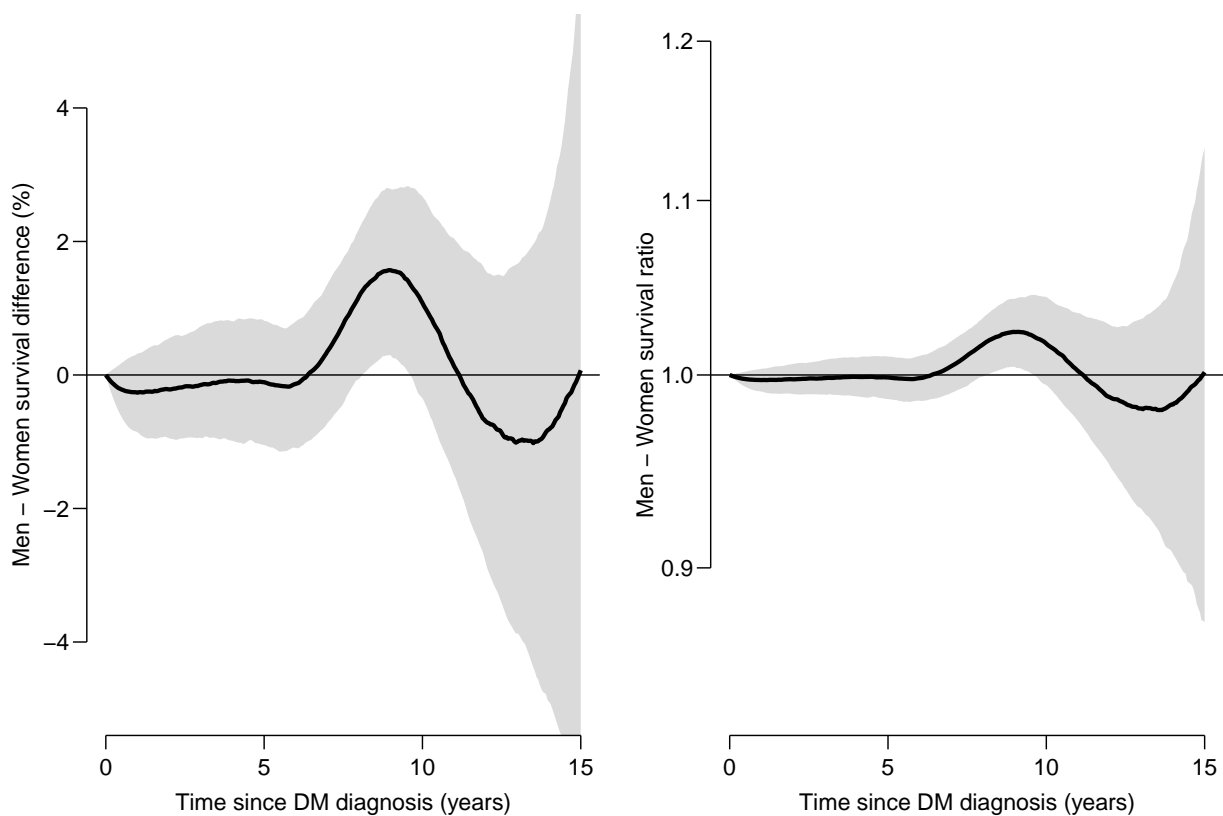
Figure 4.5: *Differences and ratios of survival between men and women, derived from the same set of bootstrap samples from the parameter vector.*   `./crisk-difrat`

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(CVD = mCVD,
+                      Can = mCan,
+                    Other = mOth),
+              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))


> par(mfrow = c(1,2))
> matshade(as.numeric(colnames(dS)), t(dS), plot = TRUE,
+          lwd = 3, ylim = c(-5, 5),
+          xlab = "Time since DM diagnosis (years)",
+          ylab = "Men - Women survival difference (%)")
> matshade(as.numeric(colnames(dxS)), t(dxS), lty = 3, col = "forestgreen")
> 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")
> matshade(as.numeric(colnames(rxS)), t(rxS), lty = 3, col = "forestgreen")
> abline(h = 1)
```
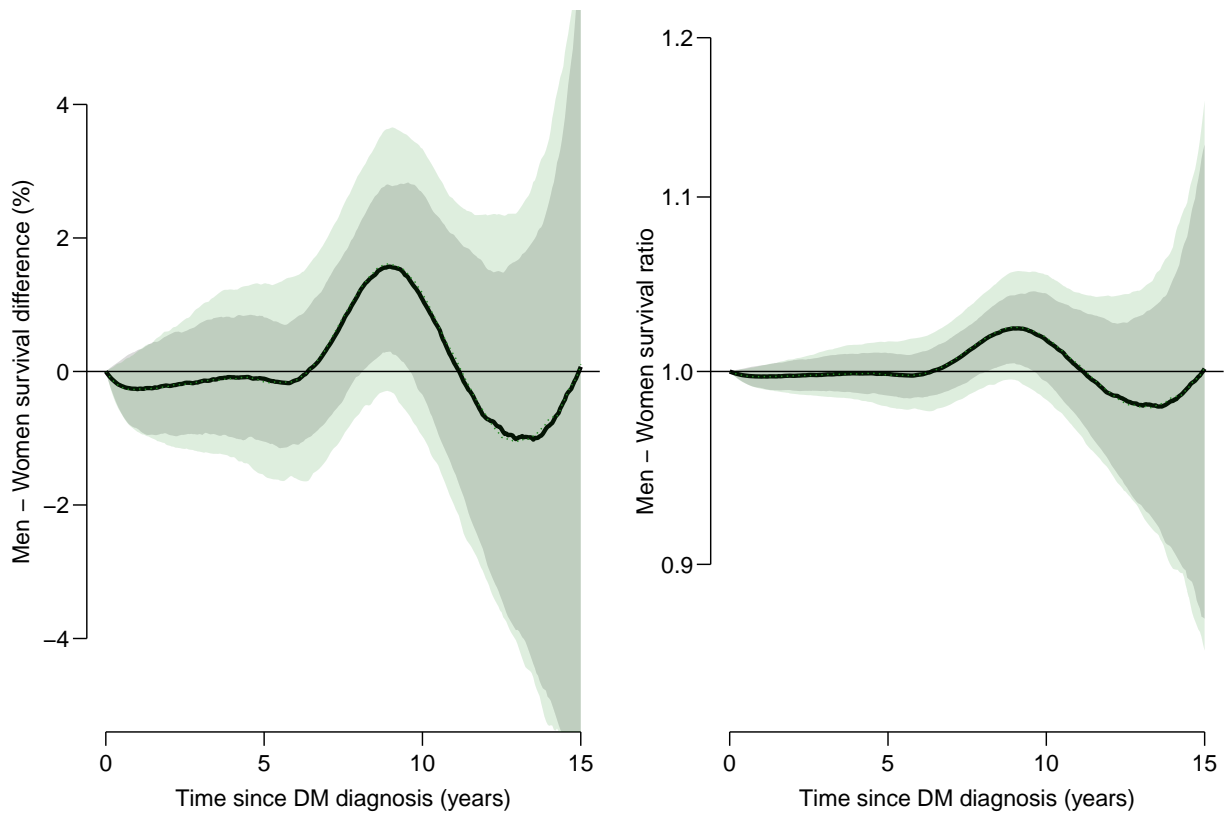

Figure 4.6: *Differences and ratios of survival between men and women, derived from separate bootstrap samples. The outer confidence bands are from bootstrap samples not properly paired between men end women.* ./crisk-difratx

```
  Start time: 2025-01-13, 09:21:57
    End time: 2025-01-13, 09:23:21
Elapsed time: 1.39 minutes
```