Parametric competing risks with simulation based confidence intervals

SDCC
April 2021
http://bendixcarstensen.com/
Version 4

Compiled Monday 5th April, 2021, 18:34
from: `/home/bendix/stat/R/lib.src/Epi/pkg/vignettes/crisk.tex`

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

Contents

Chapter 1

Competing risks in practice

The concept of competing risks is one where persons in a given state, 'alive', say, er subject to a number of different causes of deaths, 'cause1', 'cause2' etc. Causes of death are required to be exhaustive and mutually exclusive. 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 what occurs after.

The likelihood for observations from a competing risk scenario is a function of the cause-specific transition rates, and is *product* of the likelihoods that would emerge if we considered each cause the only one. Thus analysis is in principle straight forward; just 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 will be done.

But most often we would like to have estimates of the cumulative risks, that is the probability of dying from a specific cause before a given time as function of time. Each of these are functions of *all* rates. Specifically, if the cause-specific rates are $\lambda_c(t)$, then:

$$R_c(t) = \int_0^t \lambda_c(s) \exp\left(- \int_0^s \sum_j \lambda_j(u) \,\mathrm{d}u \right) \mathrm{d}s$$

Even if we from the modeling of the λs have standard errors of $\log(\lambda_c)$ the standard errors of R_cs will be analytically intractable from these.

The only viable way to get confidence intervals for the cumulative risks, R_c, is by calculation of the rates $\lambda(t)$ by sampling from the posterior distribution of the parameters in the models for $\log(\lambda(s))$, and computing the integrals numerically for each simulated sample.

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.

1.1 Example data

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

```
> library(Epi)
> library(popEpi)
> 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
 ""  ""  ""
```

```
> Mdm <- mcutLexis( Ldm,
+                    wh = c('dooad','doins'),
+            new.states = c('OAD','Ins'),
+             precursor = 'DM',
+            seq.states = FALSE,
+                  ties = TRUE )
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   22920.25      7532
  OAD        0  992 3327    0    1005      5324     1997   22965.27      5324
  Ins        0  152    0  462     172       786      324    3883.10       786
  Ins+OAD    0  299    0    0     878      1177      299    4504.65      1177
  Sum     2830 2499 6284 1151    2055     14819     7322   54273.27      9996
```

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

```
> Sdm <- splitMulti(factorize(subset(Mdm, lex.Cst == "DM")),
+                   tfd = seq(0, 20, 1/12))
NOTE: lex.Cst and lex.Xst now have levels:
 DM Dead OAD Ins
```

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

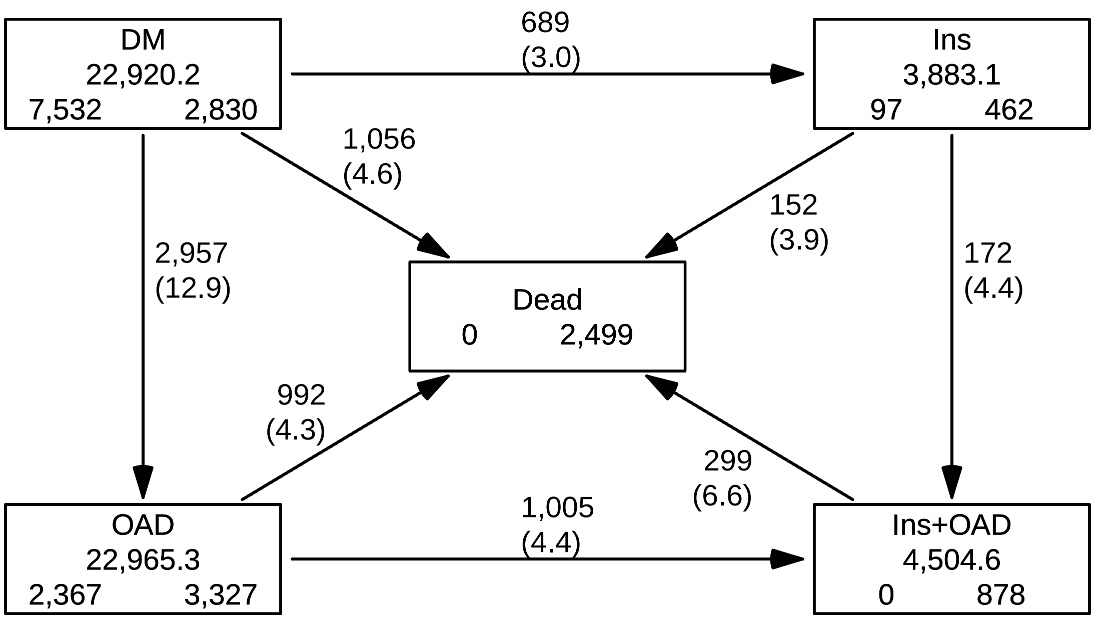

Figure 1.1: *The transitions in the multistate model, where follow-up is extended also after beginning of first drug exposure. Rates in brackets are per 100 PY.* `./crisk-boxes5`

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

```
> boxes(Relevel(Sdm, c(1, 4, 2, 3)),
+       boxpos  = list(x = c(15, 85, 80, 15),
+                      y = c(85, 85, 20, 15)),
+       scale.R = 100,
+       show.BE = TRUE )
```

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

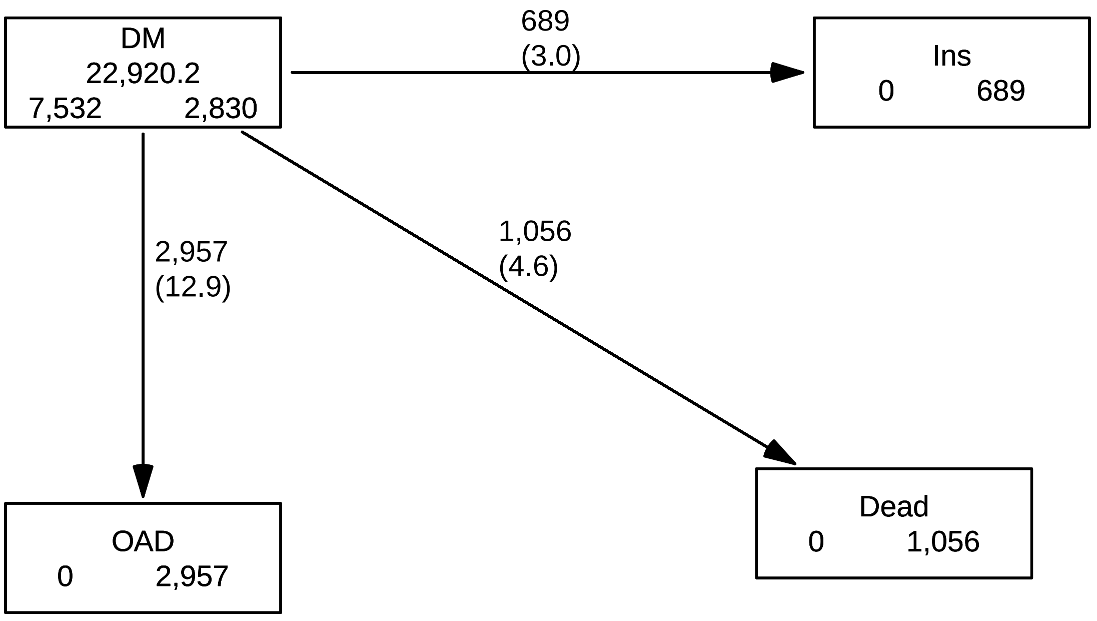

Figure 1.2: *The transitions in the competing risks model, where follow-up is stopped at drug exposure. By that token only the* `DM` *state has person-years; a characteristic of a competing risks situation.* ./crisk-boxes4

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

With these models fitted we can compute the rates, cumulative rates and the cumulative risks an sojourn times in states using the usual formulae. First we compute the rates in intervals of length 1/100 years. Note that these models only have time since diagnosis as covariates, so they are the counterpart of Nelson-Aalen estimates, albeit in a biologically more meaningful guise.

The points where we compute the predicted rates are midpoints of intervals of length 1/100 year. These points are unrelated to the follow-up intervals in which we split the data—they were 1 month intervals, here we use 1/100 year (about 3.7 days):

```
> int <- 1 / 100
> nd <- data.frame(tfd = seq(0, 10, int)) # not the same as the split,
>                                          # and totally unrelated to it
> rownames(nd) <- nd$tfd
> str(nd)
'data.frame':        1001 obs. of  1 variable:
 $ tfd: num  0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 ...
```

With this we can show the rates as a function of the time since diagnosis:

```
> matshade(nd$tfd, cbind(ci.pred(mD, nd),
+                        ci.pred(mI, nd),
+                         ci.pred(m0, nd))*1000,
+           ylim = c(0.02,500), yaxt = "n",
+           ylab = "Rates per 1000 PY",
+           xlab = "Time since DM diagnosis (years)",
+           col = c("black","red","blue"), log = "y", lwd = 3, plot = TRUE)
> 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 = ll<-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)
```

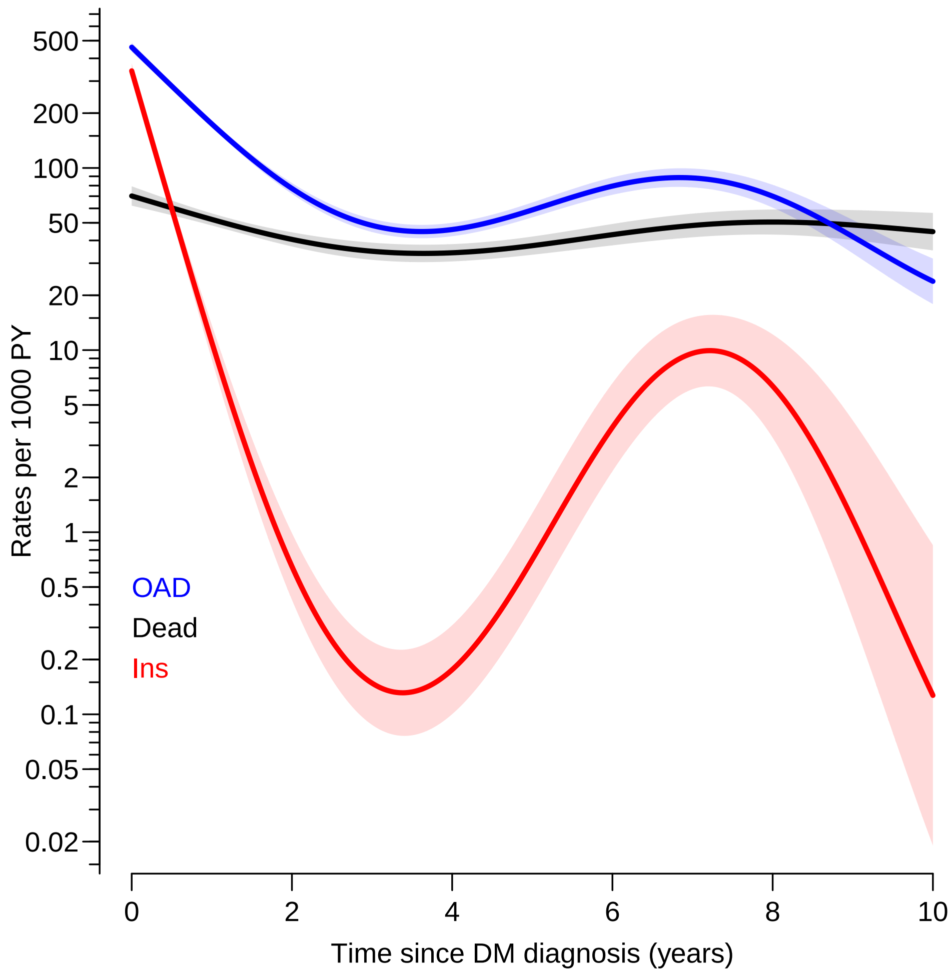

Figure 1.3: *Estimated rates from the* `DM` *state, estimates are from* `gam` *models fitted to data split in 1 month intervals (1/12 year, that is). Rates of* `OAD` *is in the vicinity of 0.1/year, and mortality about half of this. Rates of insulin start among persons on no other drug are beginning high decreasing to about 4 year and then have a peak at 8 years.* `./crisk-rates`

Note that the graph in figure 1.3 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.

1.3 Cumulative rates and risks

For the calculation of the cumulative rates and state probabilities, we need just the rates without CIs:

```
> # function that calculates the midpoints between sucessive values
> mp <- function(x) x[-1] - diff(x) / 2
> # rates at midpoints
> 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 fuction at right border of the intervals
> LD <- cumsum(lD) * int
> LI <- cumsum(lI) * int
> LO <- cumsum(lO) * int
> Sv <- exp( -LD - LI - LO )
> # but when integrating to get the cumulative risks we use the average
> # of the survival function at the two endpoints (adding 1 as the first)
> 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:

```
> 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. 1.0000010097513576390
Median  1.0000010156375198633
Mean    1.0000009965292544489
3rd Qu. 1.0000010318374084051
Max.    1.0000010387825954972
> options(oo)
```

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)
```

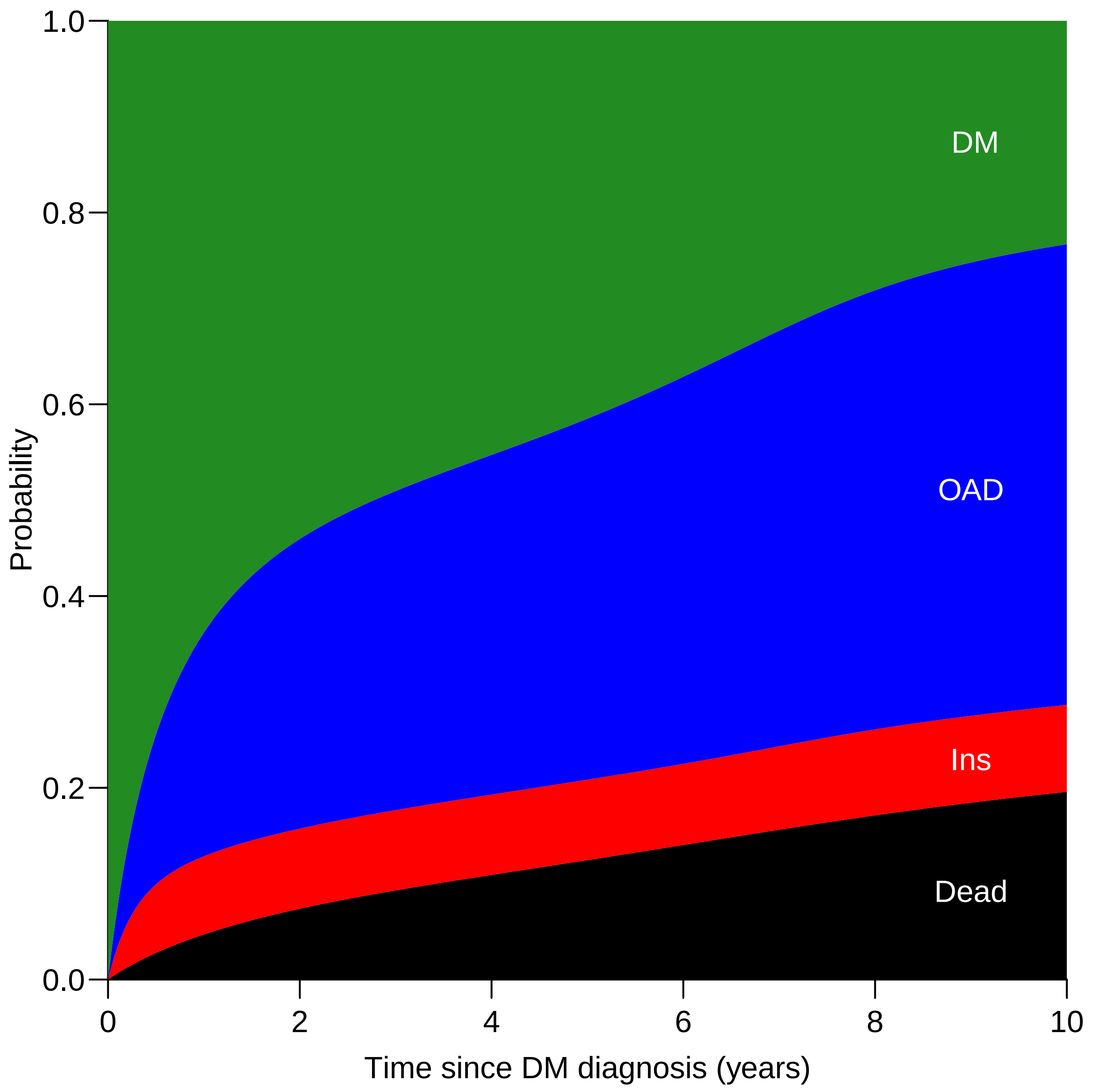

Figure 1.4: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`*. We are not concerned about what occur after these events.* `Dead` *means dead without being on any drug.* `./crisk-stack`

Chapter 2

Confidence intervals

We want confidence intervals for each of the 4 cumulative risks, but we may also be interested in confidence intervals for *sums* of any subset of the cumulative risks, corresponding to the borders between the colours in figure 1.4. 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 of the same structure as the one we just extracted from the model.

Moreover, we might also want confidence intervals for sojourn times (i.e. time spent) in each state up to a given time, which would come almost for free from the simulation approach.

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 is 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 an indication of 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    0.3414494 0.3101362 0.3759241 0.3414494 0.3101362 0.3759241
0.01 0.3297277 0.2999811 0.3624240 0.3297277 0.2999811 0.3624240
0.02 0.3184085 0.2901440 0.3494263 0.3184085 0.2901440 0.3494263
0.03 0.3074779 0.2806147 0.3369126 0.3074779 0.2806147 0.3369126
0.04 0.2969226 0.2713834 0.3248652 0.2969226 0.2713834 0.3248652
```

```
0.05 0.2867299 0.2624406 0.3132671 0.2867299 0.2624406 0.3132671
> head(cbind(ci.pred(mI,nd), exp(ci.lin(mI,nd)[,c(1,5:6)])))
      Estimate      2.5%     97.5%  Estimate      2.5%     97.5%
0    0.3414494 0.3101362 0.3759241 0.3414494 0.3101362 0.3759241
0.01 0.3297277 0.2999811 0.3624240 0.3297277 0.2999811 0.3624240
0.02 0.3184085 0.2901440 0.3494263 0.3184085 0.2901440 0.3494263
0.03 0.3074779 0.2806147 0.3369126 0.3074779 0.2806147 0.3369126
0.04 0.2969226 0.2713834 0.3248652 0.2969226 0.2713834 0.3248652
0.05 0.2867299 0.2624406 0.3132671 0.2867299 0.2624406 0.3132671
```

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

```
> str(ci.lin(mI, nd, sample = 4))
 num [1:1001, 1:4] -0.992 -1.028 -1.064 -1.101 -1.137 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  ..$ : NULL
> head(cbind(ci.pred(mI,nd), exp(ci.lin(mI, nd, sample = 4))))
      Estimate      2.5%     97.5%
0    0.3414494 0.3101362 0.3759241 0.3250063 0.3512494 0.3508379 0.3573523
0.01 0.3297277 0.2999811 0.3624240 0.3142096 0.3392021 0.3385386 0.3452093
0.02 0.3184085 0.2901440 0.3494263 0.3037717 0.3275680 0.3266705 0.3334791
0.03 0.3074779 0.2806147 0.3369126 0.2936805 0.3163331 0.3152185 0.3221475
0.04 0.2969226 0.2713834 0.3248652 0.2839247 0.3054837 0.3041681 0.3112010
0.05 0.2867299 0.2624406 0.3132671 0.2744931 0.2950065 0.2935053 0.3006267
```

The simulation is taking place at the parameter level and the transformation to survival and cumulative risks is simply a function applied to every simulated set of rates.

2.1 Joint models for several transitions

Note that we are implicitly 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, using one model—we are in trouble, because we then need one sample from the posterior generating two 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 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.

Finally, it 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.

2.2 Simulation based confidence intervals

These ideas have been 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 1/100 years, approximately 4 days:

```
> system.time(
+ res <- ci.Crisk(list(OAD = mO,
+                        Ins = mI,
+                       Dead = mD),
+                               nd = data.frame(tfd = 0:1000 / 100),
+                               nB = 1000,
+                             perm = 4:1))
Times are assumed to be in the column tfd at equal distances of 0.01
   user  system elapsed
 15.087   0.754  15.323
```

```
> str(res)
```

```
List of 4
 $ Crisk: num [1:1001, 1:4, 1:3] 1 0.991 0.983 0.975 0.967 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:1001, 1:3, 1:3] 0 0.000699 0.00139 0.002072 0.002747 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:3] "Dead" "Dead+Ins" "Dead+Ins+OAD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:1001, 1:4, 1:3] 0 0.00996 0.01983 0.02962 0.03933 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:1001] "0.0" "0.01" "0.02" "0.03" ...
  .. ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:1001] 0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 ...
 - attr(*, "int")= num 0.01
```

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. This is mainly for plotting convenience.

The three components of `res` represent:

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

The first dimension of each is time corresponding to endpoints of intervals of length `int`, (assumed) starting at 0. 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 intervals as indicated.

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 = 0:1000 / 100),
+                               nB = 2000,
+                          sim.res = 'rates'))
Times are assumed to be in the column tfd at equal distances of 0.01
   user  system elapsed
  0.456   0.500   0.321
> str(rsm)
 num [1:1001, 1:3, 1:2000] 0.471 0.466 0.462 0.457 0.453 ...
 - attr(*, "dimnames")=List of 3
  ..$ time: chr [1:1001] "1" "2" "3" "4" ...
  ..$ mod : chr [1:3] "OAD" "Ins" "Dead"
  ..$ sim : chr [1:2000] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.01
```

This is bootstrap samples of the rates evaluated at the 1000 midpoints of intervals.

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 = 0:1000 / 100),
+                               nB = 2000,
+                          sim.res = 'crisk'))
Times are assumed to be in the column tfd at equal distances of 0.01
   user  system elapsed
  6.126   0.584   6.070
> str(csm)
 num [1:1001, 1:4, 1:2000] 1 0.992 0.984 0.976 0.968 ...
 - attr(*, "dimnames")=List of 3
  ..$ time : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  ..$ sim  : chr [1:2000] "1" "2" "3" "4" ...
 - attr(*, "int")= num 0.01
```

These are the cumulative risks evaluated at the 1001 endpoints of the intervals, and also includes the survival probability in the first slot of the 1st dimension of `rsm`.

In both cases, the first slot of the 3rd dimension, `sim`, is the rates, resp. cumulative risks from the model.

2.3 Simulated confidence intervals for rates

In figure 1.3 we showed the rates with confidence intervals from the model. But in `rsm` we have 2000 (parametric) bootstrap samples of the occurrence rates, so we can derive the bootstrap medians and the bootstrap c.i.—remember that the first slice of the 3rd dimension is the model estimates that should not enter the calculations. We use the function `mnqt` to compute the model estimate and the mean, median and quantiles of the simulated values.

```
> Brates <- aperm(apply(rsm, 1:2, Epi:::mnqt), c(2,3,1))
> str(Brates)
 num [1:1001, 1:3, 1:3] 0.46 0.456 0.451 0.447 0.443 ...
 - attr(*, "dimnames")=List of 3
  ..$ time: chr [1:1001] "1" "2" "3" "4" ...
  ..$ mod : chr [1:3] "OAD" "Ins" "Dead"
  ..$     : chr [1:3] "50%" "2.5%" "97.5%"
```

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:

```
> 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 = ll<-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)
```

2.4 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 immediately plot the three cumulative risks:

```
> 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)
```

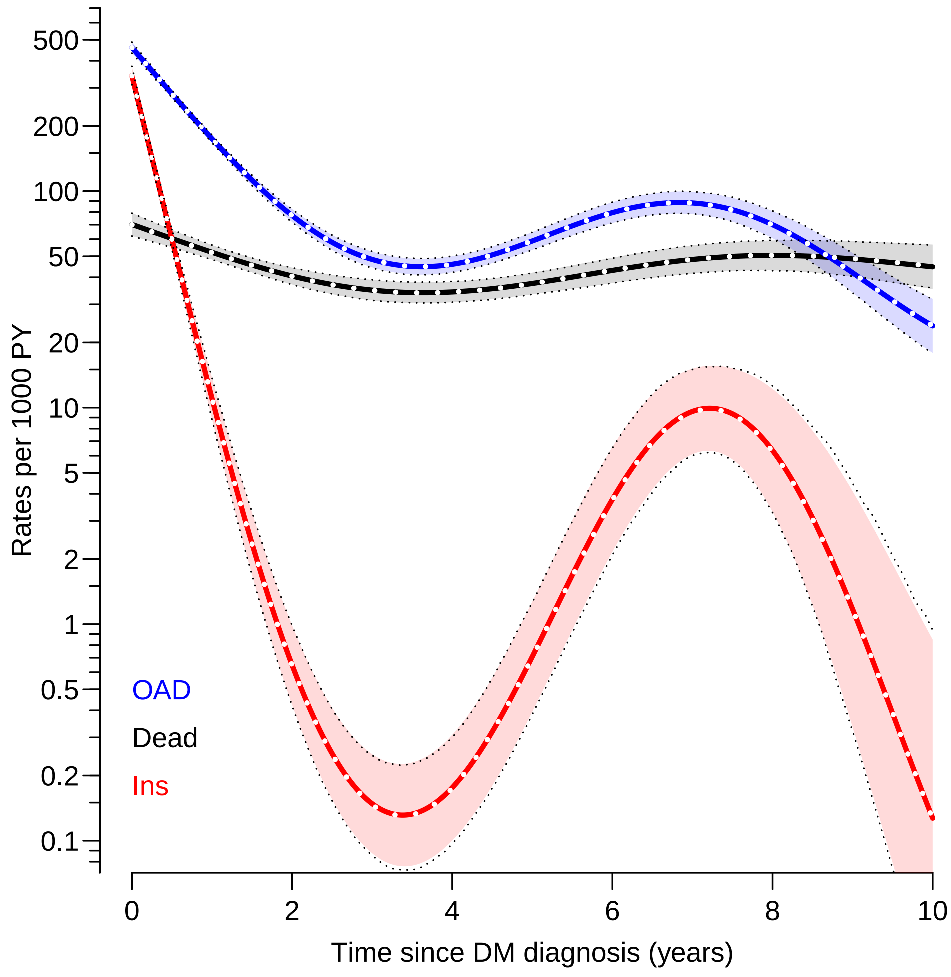

Figure 2.1: *Estimated rates from the* `DM` *state, estimates are from* `gam` *models fitted to data split in 1 month intervals (1/12 year, that is). The white dotted curves are the bootstrap medians, black dotted curves are the bootstrap 95% c.i.s.* `./crisk-rates-ci`

2.5 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$Crisk)
 num [1:1001, 1:4, 1:3] 1 0.991 0.983 0.975 0.967 ...
 - attr(*, "dimnames")=List of 3
  ..$ time : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  ..$      : chr [1:3] "50%" "2.5%" "97.5%"
> str(res$Srisk)
 num [1:1001, 1:3, 1:3] 0 0.000699 0.00139 0.002072 0.002747 ...
 - attr(*, "dimnames")=List of 3
  ..$ time : chr [1:1001] "0" "0.01" "0.02" "0.03" ...
  ..$ cause: chr [1:3] "Dead" "Dead+Ins" "Dead+Ins+OAD"
  ..$      : chr [1:3] "50%" "2.5%" "97.5%"
```

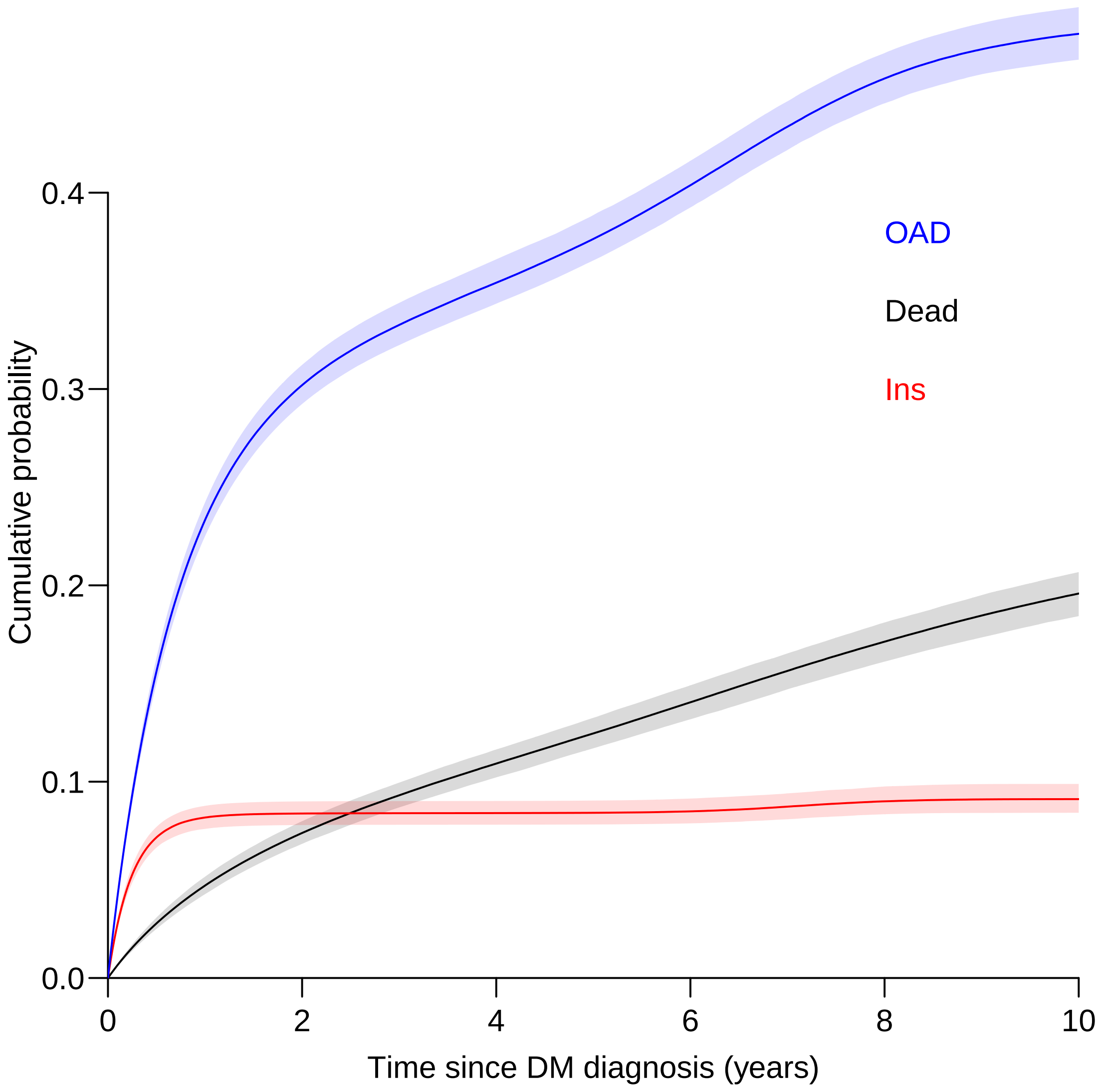

Figure 2.2: *Cumulative risks for the three types of events, with 95% bootstrap-based confidence intervals as shades.* `./crisk-crates`

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 can add the confidence intervals

```
> 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 = 0.3)
```

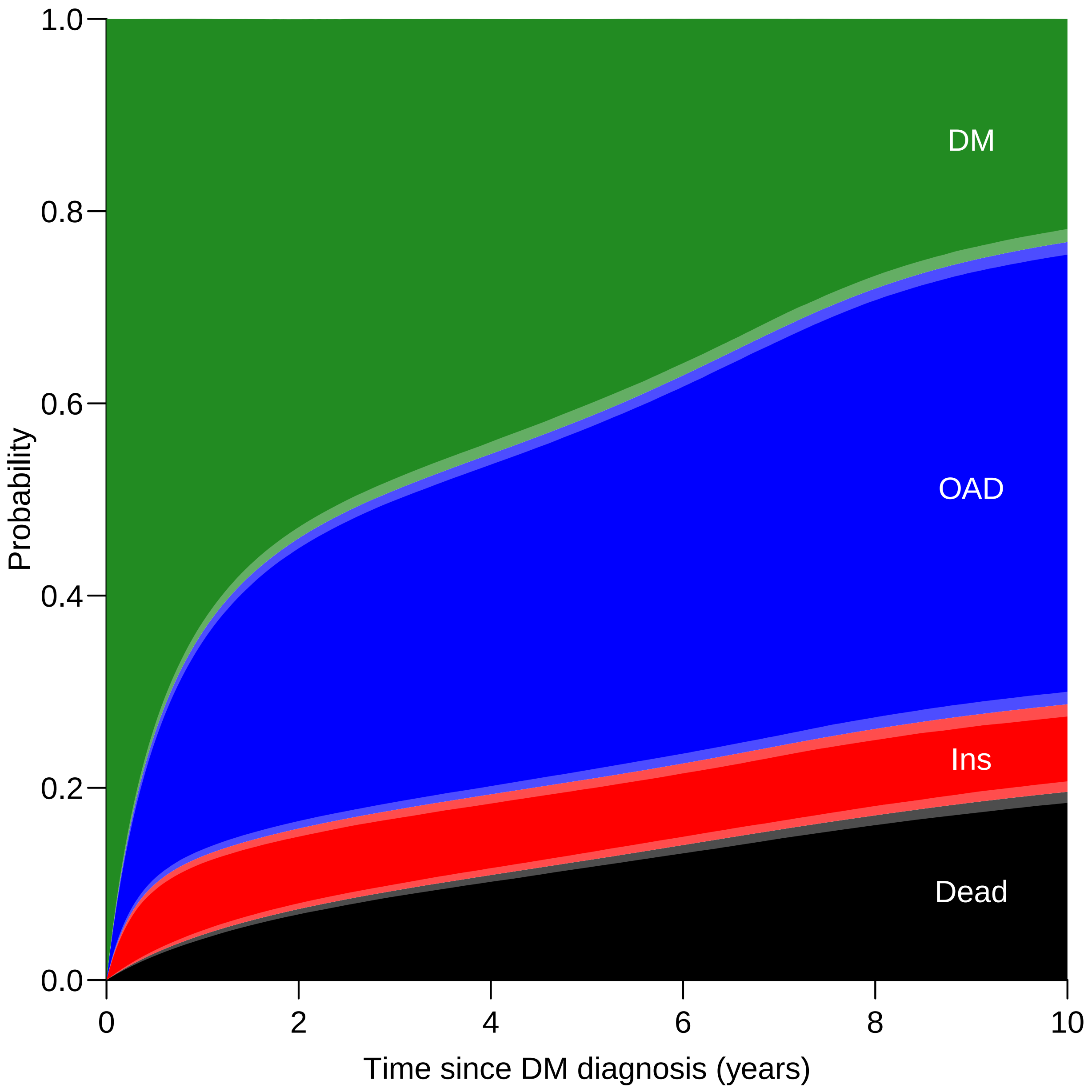

Figure 2.3: *Probabilities of being in the 4 different states as a function of time since diagnosis. Note that* `OAD` *means that OAD was initiated first, and similarly for* `Ins`*. We are not concerned about what occurs* after *these events.* `Dead` *means dead without being on any drug.*
The white shadings around the borders between coloured areas represent the 95% confidence intervals for the (sum of) probabilities. `./crisk-stack-ci`

2.6 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:

```
> str(res$Stime)
 num [1:1001, 1:4, 1:3] 0 0.00996 0.01983 0.02962 0.03933 ...
 - attr(*, "dimnames")=List of 3
```

```
  ..$ time : chr [1:1001] "0.0" "0.01" "0.02" "0.03" ...
  ..$ cause: chr [1:4] "Surv" "OAD" "Ins" "Dead"
  ..$       : chr [1:3] "50%" "2.5%" "97.5%"
```

We extract the 5 and 10 years components:

```
> s510 <- res$Stime[1:2*500,,]
> dimnames(s510)[[1]] <- c(" 5 yr","10 yr")
> round(ftable(s510, row.vars=1:2), 2)
              50% 2.5% 97.5%
time  cause
 5 yr Surv   2.76 2.71  2.81
      OAD    1.44 1.40  1.49
      Ins    0.40 0.37  0.43
      Dead   0.39 0.36  0.42
10 yr Surv   4.32 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;4.42), and during the first 5 years 2.77 (2.72;2.82).

The quantity `OAD` is the years lived without medication that has been terminated by OAD inception, and similarly for `Ins` and `Dead`.

2.7 A simple illustration

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

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

```
> data(DMlate)
> set.seed(7465)
> wh <- sample(1:3, nrow(DMlate), r=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
```

`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.

2.7.1 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)
```

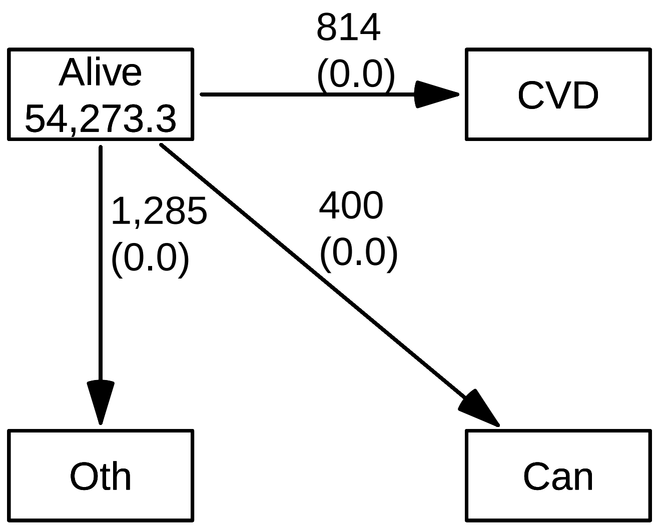

Figure 2.4: *Transitions from live to different causes of death.* `./crisk-boxes`

2.7.2 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 <- splitMulti(dmL, age = 0:120)#seq(0, 120, 1/2))
> summary(sL)
Transitions:
     To
From    Alive CVD Can  Oth  Records:  Events: Risk time:  Persons:
  Alive 61627 814 400 1285     64126     2499   54273.27      9996
```

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

2.7.3 Derived measures

With these three models for the occurrence rates we can compute the cumulative risks of dying from each of the causes. We just need a prediction data frame that gives the rates at closely spaced times, in this case for men:

```
> nm <- data.frame(tfD = seq(0, 15, 0.1), sex = "M")
```

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

```
> cR <- ci.Crisk(list(CVD = mCVD,
+                       Can = mCan,
+                     Other = mOth),
+                  nd = nm)
Times are assumed to be in the column tfD at equal distances of 0.1
> str(cR)
List of 4
 $ Crisk: num [1:151, 1:4, 1:3] 1 0.994 0.988 0.983 0.978 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:151] "0" "0.1" "0.2" "0.3" ...
  .. ..$ cause: chr [1:4] "Surv" "CVD" "Can" "Other"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Srisk: num [1:151, 1:3, 1:3] 0 0.00314 0.00616 0.00907 0.01189 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:151] "0" "0.1" "0.2" "0.3" ...
  .. ..$ cause: chr [1:3] "Other" "Other+Can" "Other+Can+CVD"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ Stime: num [1:151, 1:4, 1:3] 0 0.0997 0.1988 0.2974 0.3954 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ time : chr [1:151] "0.0" "0.1" "0.2" "0.3" ...
  .. ..$ cause: chr [1:4] "Surv" "CVD" "Can" "Other"
  .. ..$      : chr [1:3] "50%" "2.5%" "97.5%"
 $ time : num [1:151] 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 ...
 - attr(*, "int")= num 0.1
```

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.

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 dimnames of the `Crisk` component:

```
> 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 - 1:3/30, c("CVD","Can","Oth"),
+      col = clr, adj = 0)
```

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

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 = clr, 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]])))
```

It is really not a good idea to color the curves, they do not refer to the causes of death, it is the areas *between* the curves that refer to causes.

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$Crisk[,c("Other","Can","CVD","Surv"),"50%"],
+               x = cR$time,
+            xlim = c(0,15), xaxs = "i", yaxs = "i", las = 1,
+            xlab = "Time since DM diagnosis (years)",
```

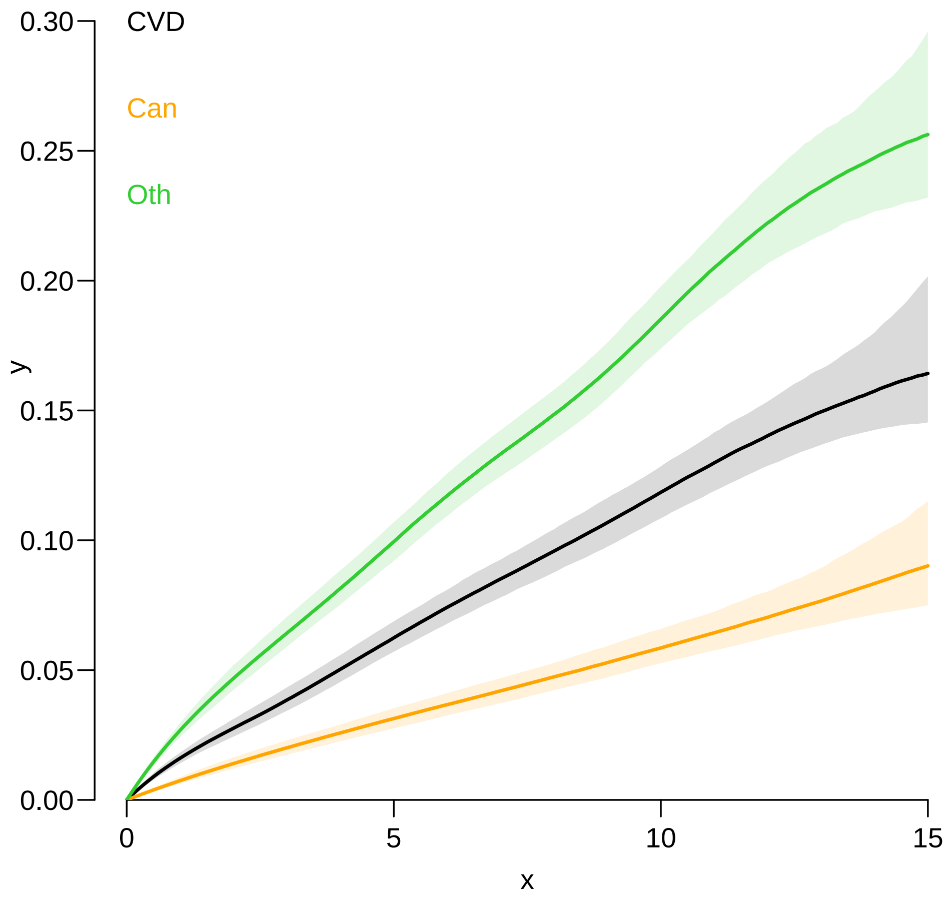

Figure 2.5: *Cumulative risks of each cause of death based on* gam *models for the cause-specific rates.* ./crisk-cR

```
+             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")
```

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%
time
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
```

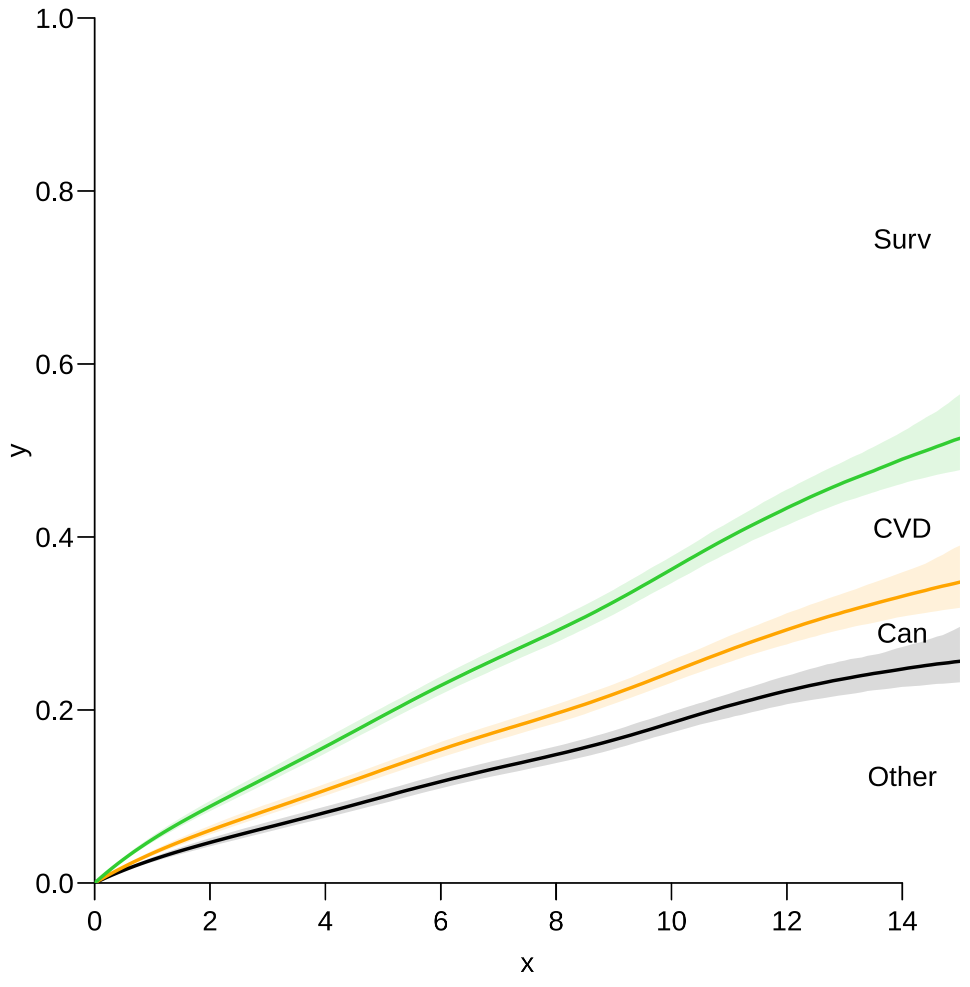

Figure 2.6: *Stacked cumulative risks — not a good graph* `./crisk-Sr1`

```
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
9           7.4  7.4   7.5  0.5  0.5   0.5  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  1.2   1.4  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 years life lost is bogus in this case as the causes of death were randomly generated.

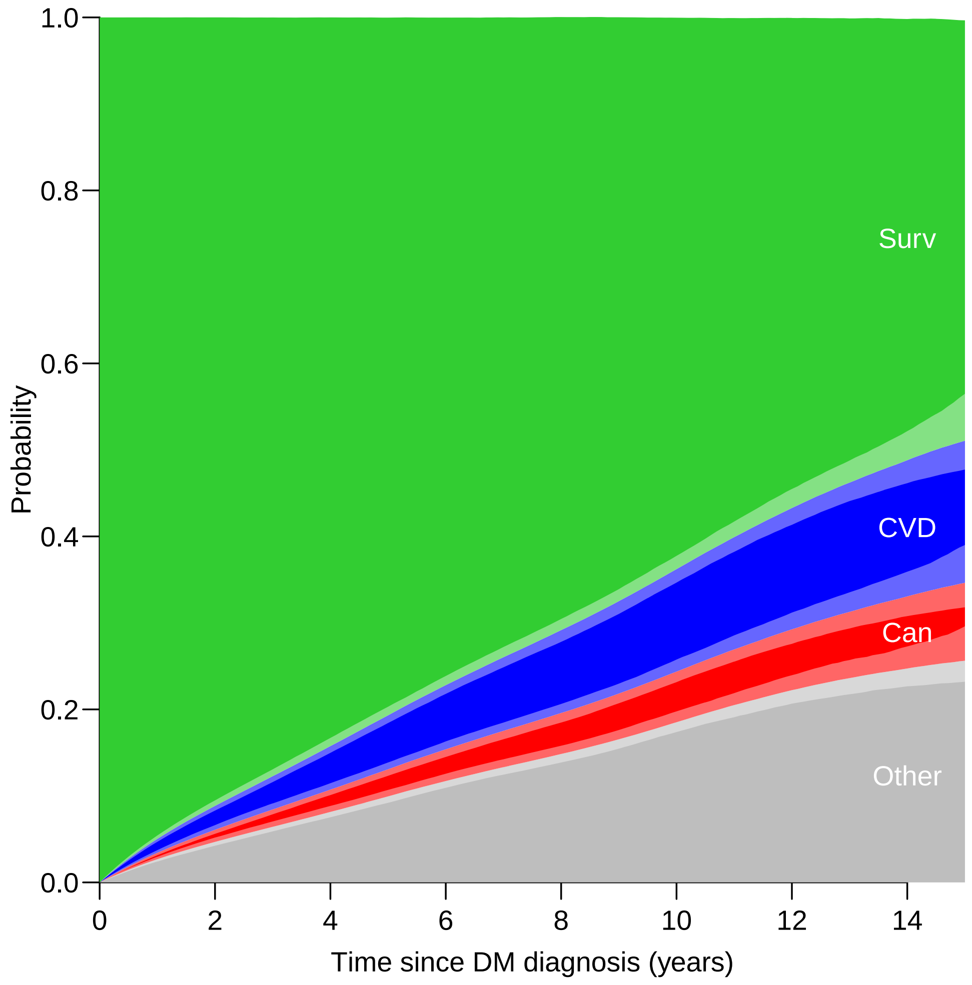

Figure 2.7: *Stacked cumulative risks with coloring of states and overlaid with confidence intervals for the probabilities shown; that is the relevant sums.* `./crisk-Sr2`