Tutorial in biostatistics: Competing risks and multi-state models
Analyses using the *mstate* package

Hein Putter
Department of Medical Statistics and Bioinformatics
Leiden University Medical Center
Postzone S-5-P
PO Box 9600
2300 RC Leiden
The Netherlands
E-mail: `h.putter@lumc.nl`

December 3, 2016

1 Introduction

This is a companion file both for the *mstate* package and for the Tutorial in Biostatistics: Competing risks and multi-state models (Putter et al. 2007), simply referred to henceforth as the tutorial. Emphasis in this document will be on the use of *mstate*, not on the theory of competing risks and multi-state models. The only exception is that I have added some theory about the Aalen-Johansen estimator that is implemented in *mstate* but did not appear in the tutorial. For other theory on multi-state models, and for interpretation of the results of the analyses, we will repeatedly refer to the tutorial. I will occasionally give more detail and show more analyses than in the tutorial. Also I sometimes give more details on the function in *mstate* than strictly necessary for the analyses in the tutorial, but not all features will be shown either. This file and the *mstate* package, which in turn contains all the data used in the tutorial, can be found at `http://www.msbi.nl/multistate`. This file is also a vignette of the *mstate* package. Type `vignette("Tutorial")` after having installed and loaded *mstate* to access this document within R.

I do not follow the order of the tutorial. Rather, I will start with multi-state models, Section 4 of the tutorial, and finally switch back to the special case of competing risks models. Sections 2, 3 and 4 of this document will discuss data preparation, estimation and prediction, respectively in multi-state models. In Section 5 I illustrate some functions of *mstate* designed especially for competing risks.

After installation, the *mstate* package is loaded in the usual way.

```
> library(mstate)
```

The versions of `R` and *mstate* used in this document are as follows:

```
> R.version$version.string

[1] "R version 3.3.2 (2016-10-31)"

> packageDescription("mstate", fields = "Version")

[1] "0.2.10"
```

2 Data preparation

The data used in Section 4 of the tutorial are 2204 patients transplanted at the EBMT between 1995 and 1998. These data are included in the *mstate* package. For (a tiny bit) more background on the data, refer to the tutorial, or type `help(ebmt3)`.

```
> data(ebmt3)
> head(ebmt3)

  id prtime prstat rfstime rfsstat dissub   age            drmatch    tcd
1  1     23      1     744       0    CML   >40    Gender mismatch No TCD
2  2     35      1     360       1    CML   >40 No gender mismatch No TCD
3  3     26      1     135       1    CML   >40 No gender mismatch No TCD
4  4     22      1     995       0    AML 20-40 No gender mismatch No TCD
5  5     29      1     422       1    AML 20-40 No gender mismatch No TCD
6  6     38      1     119       1    ALL   >40 No gender mismatch No TCD
```

Let us first have a look at the covariates. For instance disease subclassification:

```
> n <- nrow(ebmt3)
> table(ebmt3$dissub)

AML ALL CML
853 447 904

> round(100 * table(ebmt3$dissub)/n)

AML ALL CML
 39  20  41
```

The output of the other covariates is omitted.

```
> table(ebmt3$age)
> round(100 * table(ebmt3$age)/n)
> table(ebmt3$drmatch)
> round(100 * table(ebmt3$drmatch)/n)
> table(ebmt3$tcd)
> round(100 * table(ebmt3$tcd)/n)
```

The first step in a multi-state model analysis is to set up the transition matrix. The transition matrix specifies which direct transitions are possible (those with `NA` are impossible) and assigns numbers to the transitions for future reference. This can be done explicitly.

```
> tmat <- matrix(NA, 3, 3)
> tmat[1, 2:3] <- 1:2
> tmat[2, 3] <- 3
> dimnames(tmat) <- list(from = c("Tx", "PR", "RelDeath"), to = c("Tx",
+     "PR", "RelDeath"))
> tmat

          to
from       Tx PR RelDeath
  Tx       NA  1        2
  PR       NA NA        3
  RelDeath NA NA       NA
```

Steven McKinney has kindly provided a convenient function *transMat* to define transition matrices. The same transition matrix may be constructed as follows.

```
> tmat <- transMat(x = list(c(2, 3), c(3), c()), names = c("Tx",
+     "PR", "RelDeath"))
> tmat

          to
from       Tx PR RelDeath
  Tx       NA  1        2
  PR       NA NA        3
  RelDeath NA NA       NA
```

For common multi-state models, such as the illness-death model (and competing risks models, Section 5) there is a built-in function to obtain these transition matrices more easily.

```
> tmat <- trans.illdeath(names = c("Tx", "PR", "RelDeath"))
> tmat
```

```
          to
from       Tx PR RelDeath
  Tx       NA  1        2
  PR       NA NA        3
  RelDeath NA NA       NA
```

The function *paths* can be used to give a list of all possible paths through the multi-state model. This function should not be used for transition matrices specifying a multi-state model with loops, since there will be infinitely many paths. At the moment there is no check for the presence of loops, but this will be included shortly.

```
> paths(tmat)
```

```
     [,1] [,2] [,3]
[1,]    1   NA   NA
[2,]    1    2   NA
[3,]    1    2    3
[4,]    1    3   NA
```

Time in the `ebmt3` data is reported in days; before doing any analysis, we first convert this to years.

```
> ebmt3$prtime <- ebmt3$prtime/365.25
> ebmt3$rfstime <- ebmt3$rfstime/365.25
```

In order to prepare data in long format, we specify the names of the covariates that we are interested in modeling. Note that I am adding `prtime`, which is not really a covariate, but specifying the time of platelet recovery. The purpose of this will become clear later. The specified covariates are to be retained in the dataset in long format (this is the argument *keep*), which we are going to call `msbmt`. For the original dataset `ebmt3`, each row corresponds to a single patient. For the long format data `msbmt`, each row will correspond to a transition for which a patient is at risk. See the tutorial for more detailed information.

```
> covs <- c("dissub", "age", "drmatch", "tcd", "prtime")
> msbmt <- msprep(time = c(NA, "prtime", "rfstime"), status = c(NA,
+     "prstat", "rfsstat"), data = ebmt3, trans = tmat, keep = covs)
```

The result is an S3 object of class *msdata* and *data.frame*. An *msdata* object is actually only a data frame with a `trans` attribute holding the transition matrix used to define it. A *print* method has been defined for *msdata* objects, which also prints the transition matrix if requested (set argument *trans* to `TRUE`, default is `FALSE`).

```
> head(msbmt)
```

```
An object of class 'msdata'

Data:
  id from to trans     Tstart      Tstop       time status dissub age
1  1    1  2     1 0.00000000 0.06297057 0.06297057      1    CML >40
```

```
2  1    1  3     2 0.00000000 0.06297057 0.06297057      0    CML >40
3  1    2  3     3 0.06297057 2.03696099 1.97399042      0    CML >40
4  2    1  2     1 0.00000000 0.09582478 0.09582478      1    CML >40
5  2    1  3     2 0.00000000 0.09582478 0.09582478      0    CML >40
6  2    2  3     3 0.09582478 0.98562628 0.88980151      1    CML >40
             drmatch    tcd     prtime
1    Gender mismatch No TCD 0.06297057
2    Gender mismatch No TCD 0.06297057
3    Gender mismatch No TCD 0.06297057
4 No gender mismatch No TCD 0.09582478
5 No gender mismatch No TCD 0.09582478
6 No gender mismatch No TCD 0.09582478
```

In the above call of ***msprep***, the *time* and *status* arguments specify the column names in the data `ebmt3` corresponding to the three states in the multi-state model. Since all the patients start in state 1 at time 0, the *time* and *status* arguments corresponding to the first state do not really have a value. In such cases, the corresponding elements of *time* and *status* may be given the value `NA`. An alternative way of specifying *time* and *status* (and *keep* as well) is as matrices of dimension $n \times S$ with S the number of states (and $n \times p$ with p the number of covariates for *keep*). The *data* argument doesn't need to be specified then.

The number of events in the data can be summarized with the function ***events***.

```
> events(msbmt)

$Frequencies
          to
from         Tx   PR RelDeath no event total entering
  Tx          0 1169      458      577           2204
  PR          0    0      383      786           1169
  RelDeath    0    0        0      841            841

$Proportions
          to
from              Tx        PR  RelDeath  no event
  Tx       0.0000000 0.5303993 0.2078040 0.2617967
  PR       0.0000000 0.0000000 0.3276305 0.6723695
  RelDeath 0.0000000 0.0000000 0.0000000 1.0000000
```

For regression purposes, we now add transition-specific covariates to the dataset. For more details on transition-specific covariates, refer to the tutorial. For a numerical covariate `cov`, the names of the expanded (transition-specific) covariates are `cov.1`, `cov.2` etc. The extension `.i` refers to transition number i. First, we define these transition-specific covariates as a separate dataset, by setting *append* to `FALSE`.

```
> expcovs <- expand.covs(msbmt, covs[2:3], append = FALSE)
> head(expcovs)

  age20.40.1 age20.40.2 age20.40.3 age.40.1 age.40.2 age.40.3
1          0          0          0        1        0        0
2          0          0          0        0        1        0
3          0          0          0        0        0        1
```

```
4          0          0          0        1        0        0
5          0          0          0        0        1        0
6          0          0          0        0        0        1
  drmatchGender.mismatch.1 drmatchGender.mismatch.2 drmatchGender.mismatch.3
1                        1                        0                        0
2                        0                        1                        0
3                        0                        0                        1
4                        0                        0                        0
5                        0                        0                        0
6                        0                        0                        0
```

We see that this expanded covariates dataset is quite large, and that the covariate names are quite long. For categorical covariates, the default names of the expanded covariates are a combination of the covariate name, the level (similar to the names of the regression coefficients that you see in regression output), followed by the transition number, in such a way that the combination is allowed as column name. If these names are too long, the user may set the value of *longnames* (default=`TRUE`) to `FALSE`. In this case, the covariate name is followed by `1`, `2` etc, before the transition number. In case of a covariate with only two levels, the covariate name is just followed by the transition number. Confident that this will work out, we also set *append* to `TRUE` (default), which will append the expanded covariates to the dataset.

```
> msbmt <- expand.covs(msbmt, covs, append = TRUE, longnames = FALSE)
> head(msbmt)

An object of class 'msdata'

Data:
  id from to trans     Tstart      Tstop       time status dissub age
1  1    1  2     1 0.00000000 0.06297057 0.06297057      1    CML >40
2  1    1  3     2 0.00000000 0.06297057 0.06297057      0    CML >40
3  1    2  3     3 0.06297057 2.03696099 1.97399042      0    CML >40
4  2    1  2     1 0.00000000 0.09582478 0.09582478      1    CML >40
5  2    1  3     2 0.00000000 0.09582478 0.09582478      0    CML >40
6  2    2  3     3 0.09582478 0.98562628 0.88980151      1    CML >40
             drmatch    tcd     prtime dissub1.1 dissub1.2 dissub1.3 dissub2.1
1    Gender mismatch No TCD 0.06297057         0         0         0         1
2    Gender mismatch No TCD 0.06297057         0         0         0         0
3    Gender mismatch No TCD 0.06297057         0         0         0         0
4 No gender mismatch No TCD 0.09582478         0         0         0         1
5 No gender mismatch No TCD 0.09582478         0         0         0         0
6 No gender mismatch No TCD 0.09582478         0         0         0         0
  dissub2.2 dissub2.3 age1.1 age1.2 age1.3 age2.1 age2.2 age2.3 drmatch.1
1         0         0      0      0      0      1      0      0         1
2         1         0      0      0      0      0      1      0         0
3         0         1      0      0      0      0      0      1         0
4         0         0      0      0      0      1      0      0         0
5         1         0      0      0      0      0      1      0         0
6         0         1      0      0      0      0      0      1         0
  drmatch.2 drmatch.3 tcd.1 tcd.2 tcd.3   prtime.1   prtime.2   prtime.3
1         0         0     0     0     0 0.06297057 0.00000000 0.00000000
```

```
2          1          0      0      0      0 0.00000000 0.06297057 0.00000000
3          0          1      0      0      0 0.00000000 0.00000000 0.06297057
4          0          0      0      0      0 0.09582478 0.00000000 0.00000000
5          0          0      0      0      0 0.00000000 0.09582478 0.00000000
6          0          0      0      0      0 0.00000000 0.00000000 0.09582478
```

The names indeed are quite a bit shorter. The downside however is that we need to remember for ourselves to which category for instance the number `1` in `age1.2` corresponds (age 20-40 with ≤ 20 as reference category).

3 Estimation

After having prepared the data in long format, estimation of covariate effects using Cox regression is straightforward using the ***coxph*** function of the *survival* package. This is not at all a feature of the *mstate* package, other than that ***msprep*** has facilitated preparation of the data. Let us consider the Markov model, where we assume different effects of the covariates for different transitions; hence we use the transition-specific covariates obtained by ***expand.covs***. The delayed entry aspect of this model for transition 3 (see discussion in the tutorial) is achieved by specifying *Surv(Tstart,Tstop,status)*, where (this is reflected in the long format data) `Tstart` is the time of entry in the state, and `Tstop` the event or censoring time, depending on the value of `status`. We consider first the model without any proportionality assumption on the baseline hazards; this is achieved by adding *strata(trans)* to the formula, which estimates separate baseline hazards for different values of `trans` (the transitions). The results appear in the left column of Table III of the tutorial.

```
> c1 <- coxph(Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
+     age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
+     age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
+     age1.3 + age2.3 + drmatch.3 + tcd.3 + strata(trans), data = msbmt,
+     method = "breslow")
> c1

Call:
coxph(formula = Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
    age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
    age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
    age1.3 + age2.3 + drmatch.3 + tcd.3 + strata(trans), data = msbmt,
    method = "breslow")

             coef exp(coef) se(coef)     z       p
dissub1.1 -0.0436    0.9573   0.0779 -0.56 0.57570
dissub2.1 -0.2972    0.7429   0.0680 -4.37 1.2e-05
age1.1    -0.1646    0.8482   0.0791 -2.08 0.03732
age2.1    -0.0898    0.9141   0.0865 -1.04 0.29908
drmatch.1  0.0458    1.0468   0.0666  0.69 0.49213
tcd.1      0.4291    1.5358   0.0804  5.33 9.6e-08
dissub1.2  0.2559    1.2916   0.1352  1.89 0.05841
dissub2.2  0.0167    1.0169   0.1084  0.15 0.87719
age1.2     0.2552    1.2907   0.1510  1.69 0.09113
age2.2     0.5265    1.6930   0.1579  3.33 0.00086
```

```
drmatch.2 -0.0753    0.9275   0.1103 -0.68 0.49501
tcd.2      0.2967    1.3454   0.1501  1.98 0.04801
dissub1.3  0.1365    1.1462   0.1480  0.92 0.35663
dissub2.3  0.2469    1.2801   0.1169  2.11 0.03460
age1.3     0.0616    1.0635   0.1534  0.40 0.68824
age2.3     0.5807    1.7874   0.1601  3.63 0.00029
drmatch.3  0.1728    1.1886   0.1145  1.51 0.13132
tcd.3      0.2009    1.2225   0.1264  1.59 0.11187

Likelihood ratio test=118  on 18 df, p=1.11e-16
n= 5577, number of events= 2010
```

The interpretation is discussed in the tutorial.

The next model considered is the Markov model where the transition hazards into relapse or death (these correspond to transitions 2 and 3) are assumed to be proportional. For this purpose transition 1 (transplantation $\rightarrow$ platelet recovery) belongs to one stratum and transitions 2 (transplantation $\rightarrow$ relapse/death) and 3 (platelet recovery $\rightarrow$ relapse/death) belong to a second stratum. Transitions 2 and 3 have the same receiving state, hence the same value of `to`, so the two strata can be distinguished by the variable `to` in our dataset. In order to distinguish between transitions 2 and 3, we introduce a time-dependent covariate `pr` that indicates whether or not platelet recovery has already occurred. For transition 2 (Tx $\rightarrow$ RelDeath) the value of `pr` equals 0, while for transition 3 (PR $\rightarrow$ RelDeath) the value of `pr` equals 1. Results are found in the middle of Table III of the tutorial.

```
> msbmt$pr <- 0
> msbmt$pr[msbmt$trans == 3] <- 1
> c2 <- coxph(Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
+     age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
+     age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
+     age1.3 + age2.3 + drmatch.3 + tcd.3 + pr + strata(to), data = msbmt,
+     method = "breslow")
> c2

Call:
coxph(formula = Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
    age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
    age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
    age1.3 + age2.3 + drmatch.3 + tcd.3 + pr + strata(to), data = msbmt,
    method = "breslow")

              coef exp(coef) se(coef)     z       p
dissub1.1 -0.04359   0.95734  0.07789 -0.56 0.57570
dissub2.1 -0.29724   0.74287  0.06800 -4.37 1.2e-05
age1.1    -0.16461   0.84822  0.07905 -2.08 0.03732
age2.1    -0.08979   0.91412  0.08647 -1.04 0.29908
drmatch.1  0.04575   1.04681  0.06660  0.69 0.49213
tcd.1      0.42907   1.53583  0.08043  5.33 9.6e-08
dissub1.2  0.26097   1.29819  0.13518  1.93 0.05355
dissub2.2  0.00364   1.00364  0.10837  0.03 0.97323
age1.2     0.25089   1.28517  0.15106  1.66 0.09673
```

```
age2.2      0.52579   1.69180  0.15789  3.33 0.00087
drmatch.2 -0.07207   0.93047  0.11026 -0.65 0.51336
tcd.2       0.31854   1.37511  0.14997  2.12 0.03367
dissub1.3  0.13981   1.15006  0.14798  0.94 0.34477
dissub2.3  0.25033   1.28445  0.11679  2.14 0.03208
age1.3      0.05556   1.05713  0.15337  0.36 0.71717
age2.3      0.56248   1.75503  0.15997  3.52 0.00044
drmatch.3  0.16915   1.18430  0.11445  1.48 0.13941
tcd.3       0.21103   1.23495  0.12620  1.67 0.09448
pr         -0.37863   0.68480  0.21152 -1.79 0.07345

Likelihood ratio test=135  on 19 df, p=0
n= 5577, number of events= 2010
```

For a discussion of the results we again refer to the tutorial. The hazard ratio of `pr` (0.685) and its *p*-value (0.073) indicate a trend-significant beneficial effect of platelet recovery on relapse-free survival. Later on we will look at the corresponding baseline transition intensities for these two models and see as a graphical check that the assumption of proportionality of the baseline hazards for transitions 2 and 3 is reasonable. This can also be tested formally using the function *`cox.zph`* (part of the *survival* package, not of *mstate*).

```
> cox.zph(c2)

                rho    chisq        p
dissub1.1  0.05050  5.11474 2.37e-02
dissub2.1 -0.00982  0.19522 6.59e-01
age1.1    -0.03058  1.93805 1.64e-01
age2.1    -0.03957  3.10494 7.81e-02
drmatch.1  0.03315  2.20235 1.38e-01
tcd.1      0.05742  6.74519 9.40e-03
dissub1.2  0.00150  0.00437 9.47e-01
dissub2.2  0.07669 11.86991 5.70e-04
age1.2    -0.03684  2.65186 1.03e-01
age2.2    -0.03593  2.52297 1.12e-01
drmatch.2  0.02100  0.88576 3.47e-01
tcd.2      0.03896  3.10115 7.82e-02
dissub1.3 -0.00338  0.02306 8.79e-01
dissub2.3  0.03787  2.95284 8.57e-02
age1.3    -0.01551  0.49723 4.81e-01
age2.3    -0.01741  0.64403 4.22e-01
drmatch.3  0.00338  0.02321 8.79e-01
tcd.3      0.03959  3.24944 7.14e-02
pr         0.01543  0.46320 4.96e-01
GLOBAL          NA 53.06349 4.58e-05
```

There is no evidence of non-proportionality of the baseline transition intensities of transitions 2 (p=0.496 for `pr`). There is strong evidence that the proportional hazards assumption for `dissub2` (CML vs AML) is violated, at least for the transitions into relapse and death. This makes sense, clinically, since CML and AML are two diseases with completely different biological pathways. It would have been much better to study separate multi-state models for the three disease subclassifications. However, since the purpose of this manuscript is to illustrate the use of *mstate*, we will blatantly ignore the clear evidence of non-proportionality for the disease subclassifications.

Building on the Markov PH model, we can investigate whether the time at which a patient arrived in state 2 (PR) influences the subsequent RFS rate, that is, the transition hazard of PR $\rightarrow$ RelDeath. Here the purpose of expanding `prtime` becomes apparent. Since `prtime` only makes sense for transition 3 (PR $\rightarrow$ RelDeath), we need the transition-specific covariate of `prtime` for transition 3, which is `prtime.3`. The corresponding model is termed the "state arrival extended Markov PH" model in the tutorial, and appears on the right of Table III.

```
> c3 <- coxph(Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
+     age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
+     age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
+     age1.3 + age2.3 + drmatch.3 + tcd.3 + pr + prtime.3 + strata(to),
+     data = msbmt, method = "breslow")
> c3

Call:
coxph(formula = Surv(Tstart, Tstop, status) ~ dissub1.1 + dissub2.1 +
    age1.1 + age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 +
    age1.2 + age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 +
    age1.3 + age2.3 + drmatch.3 + tcd.3 + pr + prtime.3 + strata(to),
    data = msbmt, method = "breslow")

              coef exp(coef) se(coef)     z       p
dissub1.1 -0.04359   0.95734  0.07789 -0.56 0.57570
dissub2.1 -0.29724   0.74287  0.06800 -4.37 1.2e-05
age1.1    -0.16461   0.84822  0.07905 -2.08 0.03732
age2.1    -0.08979   0.91412  0.08647 -1.04 0.29908
drmatch.1  0.04575   1.04681  0.06660  0.69 0.49213
tcd.1      0.42907   1.53583  0.08043  5.33 9.6e-08
dissub1.2  0.26090   1.29810  0.13518  1.93 0.05361
dissub2.2  0.00376   1.00377  0.10837  0.03 0.97232
age1.2     0.25095   1.28525  0.15106  1.66 0.09665
age2.2     0.52577   1.69176  0.15789  3.33 0.00087
drmatch.2 -0.07209   0.93045  0.11026 -0.65 0.51324
tcd.2      0.31824   1.37470  0.14997  2.12 0.03384
dissub1.3  0.13202   1.14113  0.14885  0.89 0.37511
dissub2.3  0.25181   1.28635  0.11682  2.16 0.03112
age1.3     0.05823   1.05996  0.15343  0.38 0.70431
age2.3     0.56575   1.76077  0.16001  3.54 0.00041
drmatch.3  0.16682   1.18154  0.11456  1.46 0.14533
tcd.3      0.20740   1.23048  0.12643  1.64 0.10091
pr        -0.40687   0.66573  0.21908 -1.86 0.06328
prtime.3   0.29523   1.34343  0.59495  0.50 0.61974

Likelihood ratio test=136  on 20 df, p=0
n= 5577, number of events= 2010
```

The influence of the time at which platelet recovery occurred seems small and is not significant (p=0.62, last row).

The clock-reset models may be obtained very similarly to those of the clock-forward models. The only difference is that *Surv(Tstart,Tstop,status)* is replaced by *Surv(time,status)*. This reflects the fact (recall that in our long format data each row corresponds to a transition) that for each transition the time starts at 0, rather than `Tstart`, the time since start of study at which the state has been entered. We will only show the code, not the output; the reader may try this for him-or herself.

```
> c4 <- coxph(Surv(time, status) ~ dissub1.1 + dissub2.1 + age1.1 +
+     age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 + age1.2 +
+     age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 + age1.3 +
+     age2.3 + drmatch.3 + tcd.3 + strata(trans), data = msbmt,
+     method = "breslow")
> c5 <- coxph(Surv(time, status) ~ dissub1.1 + dissub2.1 + age1.1 +
+     age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 + age1.2 +
+     age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 + age1.3 +
+     age2.3 + drmatch.3 + tcd.3 + pr + strata(to), data = msbmt,
+     method = "breslow")
> c6 <- coxph(Surv(time, status) ~ dissub1.1 + dissub2.1 + age1.1 +
+     age2.1 + drmatch.1 + tcd.1 + dissub1.2 + dissub2.2 + age1.2 +
+     age2.2 + drmatch.2 + tcd.2 + dissub1.3 + dissub2.3 + age1.3 +
+     age2.3 + drmatch.3 + tcd.3 + pr + prtime.3 + strata(to),
+     data = msbmt, method = "breslow")
```

4 Prediction

In order to obtain prediction probabilities in the context of the Markov multi-state models discussed in the previous section, basically two steps are involved. The first is to use the estimated parameters and baseline transition hazards and the covariate values of a patient of interest, to obtain patient-specific transition hazards for that patient, for each of the transitions in the multi-state model. This is what the function `msfit` is designed to do. The second step is to use the resulting patient-specific transition hazards (and variances and covariances) as input for `probtrans` to obtain (patient-specific) transition probabilities.

I will first show how `msfit` can be used to obtain the baseline hazards associated with the Markov stratified and PH models. The hazards of the Markov stratified models (and their variances and covariates) are obtained by first creating a new dataset containing the (expanded) covariates along with their values (in this case 0). This is very similar to the use of `survfit` from the *survival* package. The important difference is that for one patient, this *newdata* data frame needs to have exactly one line for each transition. When transition-specific covariates have been used in the model, the easiest way to obtain such a data frame is to first create a data frame with the basic covariates and then using `expand.covs` to obtain the transition-specific covariates. Since `expand.covs` expects an *msdata* object, we set the class of the *newdata* data to *msdata* explicitly. We also copy the levels of the categorical covariates before expanding, although this is not really necessary here.

```
> newd <- data.frame(dissub = rep(0, 3), age = rep(0, 3), drmatch = rep(0,
+     3), tcd = rep(0, 3), trans = 1:3)
> newd$dissub <- factor(newd$dissub, levels = 0:2, labels = levels(ebmt3$dissub))
> newd$age <- factor(newd$age, levels = 0:2, labels = levels(ebmt3$age))
> newd$drmatch <- factor(newd$drmatch, levels = 0:1, labels = levels(ebmt3$drmatch))
> newd$tcd <- factor(newd$tcd, levels = 0:1, labels = levels(ebmt3$tcd))
> attr(newd, "trans") <- tmat
> class(newd) <- c("msdata", "data.frame")
> newd <- expand.covs(newd, covs[1:4], longnames = FALSE)
> newd$strata = 1:3
> newd

An object of class 'msdata'

Data:
  dissub  age            drmatch    tcd trans dissub1.1 dissub1.2 dissub1.3
1    AML <=20 No gender mismatch No TCD     1         0         0         0
2    AML <=20 No gender mismatch No TCD     2         0         0         0
3    AML <=20 No gender mismatch No TCD     3         0         0         0
  dissub2.1 dissub2.2 dissub2.3 age1.1 age1.2 age1.3 age2.1 age2.2 age2.3
1         0         0         0      0      0      0      0      0      0
2         0         0         0      0      0      0      0      0      0
3         0         0         0      0      0      0      0      0      0
  drmatch.1 drmatch.2 drmatch.3 tcd.1 tcd.2 tcd.3 strata
1         0         0         0     0     0     0      1
2         0         0         0     0     0     0      2
3         0         0         0     0     0     0      3
```

The last command where the column `strata` is added is important and points to a second major difference between *`survfit`* and *`msfit`*. The *newdata* data frame needs to have a column `strata` specifying to which stratum in the `coxph` object each transition belongs. Here each transition corresponds to a separate stratum, so we specify 1, 2, and 3.

To obtain an estimate of the baseline cumulative hazard for the "stratified hazards" model, *`msfit`* can be called with the first Cox model, `c1`, as input model, and `newd` as *newdata* argument.

```
> msf1 <- msfit(c1, newdata = newd, trans = tmat)
```

The result is an object of class *msfit*, which is a list with three items, `Haz`, `varHaz`, and `trans`. The item `trans` records the transition matrix used when constructing the *msfit* object. `Haz` contains the estimated cumulative hazard for each of the transitions for the particular patient specified in `newd`, while `varHaz` contains the estimated variances of these cumulative hazards, as well as the covariances for each combination of two transitions. All are evaluated at the time points for which any event in any transition occurs, possibly augmented with the largest (non-event) time point in the data. The *`summary`* method for *msfit* objects is most conveniently used for a summary. If we also would like to have a look at the covariances, we could set the argument *variance* equal to `TRUE`.

```
> summary(msf1)

An object of class 'msfit'

Transition 1 (head and tail):

         time          Haz trans
1 0.002737851 0.0005277714     1
2 0.008213552 0.0010560892     1
```

```
3 0.010951403 0.0010560892     1
4 0.016427105 0.0010560892     1
5 0.019164956 0.0015857558     1
6 0.021902806 0.0015857558     1

...

        time       Haz trans
500 6.253251 0.9513165     1
501 6.357290 0.9513165     1
502 6.362765 0.9513165     1
503 6.798084 0.9513165     1
504 7.110198 0.9513165     1
505 7.731691 0.9513165     1

Transition 2 (head and tail):

           time          Haz trans
506 0.002737851 0.0003046955     2
507 0.008213552 0.0003046955     2
508 0.010951403 0.0006097444     2
509 0.016427105 0.0012203981     2
510 0.019164956 0.0018316171     2
511 0.021902806 0.0024438486     2

...

         time       Haz trans
1005 6.253251 0.5020560     2
1006 6.357290 0.5020560     2
1007 6.362765 0.5248419     2
1008 6.798084 0.5248419     2
1009 7.110198 0.5248419     2
1010 7.731691 0.5248419     2

Transition 3 (head and tail):

            time Haz trans
1011 0.002737851   0     3
1012 0.008213552   0     3
1013 0.010951403   0     3
1014 0.016427105   0     3
1015 0.019164956   0     3
1016 0.021902806   0     3

...

         time       Haz trans
1510 6.253251 0.3291154     3
```

```
1511 6.357290 0.3427115     3
1512 6.362765 0.3427115     3
1513 6.798084 0.3693677     3
1514 7.110198 0.4647197     3
1515 7.731691 0.4647197     3
```

Let us have a closer look at some of the variances and covariances as well.

```
> vH1 <- msf1$varHaz
> head(vH1[vH1$trans1 == 1 & vH1$trans2 == 1, ])

         time       varHaz trans1 trans2
1 0.002737851 2.798518e-07      1      1
2 0.008213552 5.629062e-07      1      1
3 0.010951403 5.629062e-07      1      1
4 0.016427105 5.629062e-07      1      1
5 0.019164956 8.500376e-07      1      1
6 0.021902806 8.500376e-07      1      1

> tail(vH1[vH1$trans1 == 1 & vH1$trans2 == 1, ])

        time      varHaz trans1 trans2
500 6.253251 0.005158522      1      1
501 6.357290 0.005158522      1      1
502 6.362765 0.005158522      1      1
503 6.798084 0.005158522      1      1
504 7.110198 0.005158522      1      1
505 7.731691 0.005158522      1      1

> tail(vH1[vH1$trans1 == 1 & vH1$trans2 == 2, ])

         time varHaz trans1 trans2
1005 6.253251      0      1      2
1006 6.357290      0      1      2
1007 6.362765      0      1      2
1008 6.798084      0      1      2
1009 7.110198      0      1      2
1010 7.731691      0      1      2

> tail(vH1[vH1$trans1 == 1 & vH1$trans2 == 3, ])

         time varHaz trans1 trans2
1510 6.253251      0      1      3
1511 6.357290      0      1      3
1512 6.362765      0      1      3
1513 6.798084      0      1      3
1514 7.110198      0      1      3
1515 7.731691      0      1      3

> tail(vH1[vH1$trans1 == 2 & vH1$trans2 == 3, ])
```

```
         time varHaz trans1 trans2
2520 6.253251      0      2      3
2521 6.357290      0      2      3
2522 6.362765      0      2      3
2523 6.798084      0      2      3
2524 7.110198      0      2      3
2525 7.731691      0      2      3
```

Note that the covariances of the estimated cumulative hazards are practically (apart from rounding errors) 0. Theoretically, they should be 0, because with separate strata and separate covariate effects for the different transitions, the estimates of the three transitions could in fact have been estimated as three separate Cox models (this would give exactly the same results).

The estimated baseline cumulative hazards for the Markov PH model are obtained in mostly the same way. The only exception is the specification of the *strata* argument in `newd`. Instead of taking the values 1, 2, and 3, for the three transitions, they take values 1, 2, 2, to indicate that transition 1 corresponds to stratum 1, and both transitions 2 and 3 correspond to stratum 2 (the order of the strata as defined in the `coxph` object). Also the time-dependent covariate `pr` needs to be included, taking the value 0 for transitions 1 and 2, and 1 for transition 3.

```
> newd$strata = c(1, 2, 2)
> newd$pr <- c(0, 0, 1)
> msf2 <- msfit(c2, newdata = newd, trans = tmat)
> summary(msf2)

An object of class 'msfit'

Transition 1 (head and tail):

         time          Haz trans
1 0.002737851 0.0005277714     1
2 0.008213552 0.0010560892     1
3 0.010951403 0.0010560892     1
4 0.016427105 0.0010560892     1
5 0.019164956 0.0015857558     1
6 0.021902806 0.0015857558     1

...

        time       Haz trans
500 6.253251 0.9513165     1
501 6.357290 0.9513165     1
502 6.362765 0.9513165     1
503 6.798084 0.9513165     1
504 7.110198 0.9513165     1
505 7.731691 0.9513165     1

Transition 2 (head and tail):

           time          Haz trans
506 0.002737851 0.0003053084     2
```

```
507 0.008213552 0.0003053084     2
508 0.010951403 0.0006107971     2
509 0.016427105 0.0012223306     2
510 0.019164956 0.0018344413     2
511 0.021902806 0.0024473467     2

...

         time       Haz trans
1005 6.253251 0.5040408     2
1006 6.357290 0.5146993     2
1007 6.362765 0.5255361     2
1008 6.798084 0.5476683     2
1009 7.110198 0.6357669     2
1010 7.731691 0.6357669     2

Transition 3 (head and tail):

            time          Haz trans
1011 0.002737851 0.0002090742     3
1012 0.008213552 0.0002090742     3
1013 0.010951403 0.0004182719     3
1014 0.016427105 0.0008370481     3
1015 0.019164956 0.0012562195     3
1016 0.021902806 0.0016759351     3

...

         time       Haz trans
1510 6.253251 0.3451655     3
1511 6.357290 0.3524644     3
1512 6.362765 0.3598855     3
1513 6.798084 0.3750415     3
1514 7.110198 0.4353712     3
1515 7.731691 0.4353712     3
```

```
> vH2 <- msf2$varHaz
> tail(vH2[vH2$trans1 == 1 & vH2$trans2 == 2, ])
```

```
         time varHaz trans1 trans2
1005 6.253251      0      1      2
1006 6.357290      0      1      2
1007 6.362765      0      1      2
1008 6.798084      0      1      2
1009 7.110198      0      1      2
1010 7.731691      0      1      2
```

```
> tail(vH2[vH2$trans1 == 1 & vH2$trans2 == 3, ])
```

```
         time varHaz trans1 trans2
1510 6.253251      0      1      3
```

```
1511 6.357290      0      1      3
1512 6.362765      0      1      3
1513 6.798084      0      1      3
1514 7.110198      0      1      3
1515 7.731691      0      1      3
```

```
> tail(vH2[vH2$trans1 == 2 & vH2$trans2 == 3, ])
```

```
         time       varHaz trans1 trans2
2520 6.253251 0.0004142378      2      3
2521 6.357290 0.0005227029      2      3
2522 6.362765 0.0006348311      2      3
2523 6.798084 0.0011112104      2      3
2524 7.110198 0.0088628795      2      3
2525 7.731691 0.0088628795      2      3
```

Note that the estimated cumulative hazards and variances for transition 1 are identical to those from `msf1`. We saw earlier that the estimated regression coefficients were also identical for the Markov stratified and the Markon PH models. Note also that the variance of the cumulative hazard of transition 3 (and 2, not shown) is smaller than with `msf1`. The cumulative hazard estimates of transitions 1 and 2 are still uncorrelated (and 1 and 3), but those of transitions 2 and 3 are correlated now, because they share a common baseline.

Let us compare the baseline hazards of the Markov stratified and PH models graphically. For this we use the ***plot*** method for *msfit* objects. Figure 1 corresponds to Figure 14 in the tutorial.

```
> par(mfrow = c(1, 2))
> plot(msf1, cols = rep(1, 3), lwd = 2, lty = 1:3, xlab = "Years since transplant",
+     ylab = "Stratified baseline hazards", legend.pos = c(2, 0.9))
> plot(msf2, cols = rep(1, 3), lwd = 2, lty = 1:3, xlab = "Years since transplant",
+     ylab = "Proportional baseline hazards", legend.pos = c(2,
+         0.9))
> par(mfrow = c(1, 1))
```

Define the multi-state model as $X(t)$, a random process taking values in $1, \ldots, S$ (S being the number of states). We are interested in estimating so called transition probabilities $P_{gh}(s,t) = P(X(t) = h \,|\, X(s) = g)$, possibly depending on covariates. For instance, $P_{13}(0,t)$ indicates the probability of having relapsed/died (state 3) by time t, given that the individual was alive without relapse or platelet recovery (state 1) at time $s = 0$. By fixing s and varying t, we can predict the future behavior of the multi-state model given the present at time s. For Markov models, these probabilities will depend only on the state at time s, not on what happened before. For these Markov models there is a powerful relation between these transition probabilities and the transition intensities, given by

$$\mathbf{P}(s,t) = \prod_{(s,t]} (\mathbf{I} + d\mathbf{\Lambda}(u)) \tag{1}$$

Here $\mathbf{P}(s,t)$ is an $S \times S$ matrix with as (g,h) element the $P_{gh}(s,t)$ in which we are interested, and $\mathbf{\Lambda}(t)$ is an $S \times S$ matrix with as off-diagonal (g,h) elements the transition intensities $\Lambda_{gh}(t)$ of transition $g \to h$. If such a direct transition is not possible, then $\Lambda_{gh}(t) = 0$. The diagonal elements of $\mathbf{\Lambda}(t)$ are defined as $\Lambda_{gg}(t) = -\sum_{h \neq g} \Lambda_{gh}(t)$, i.e. as minus the sum of the

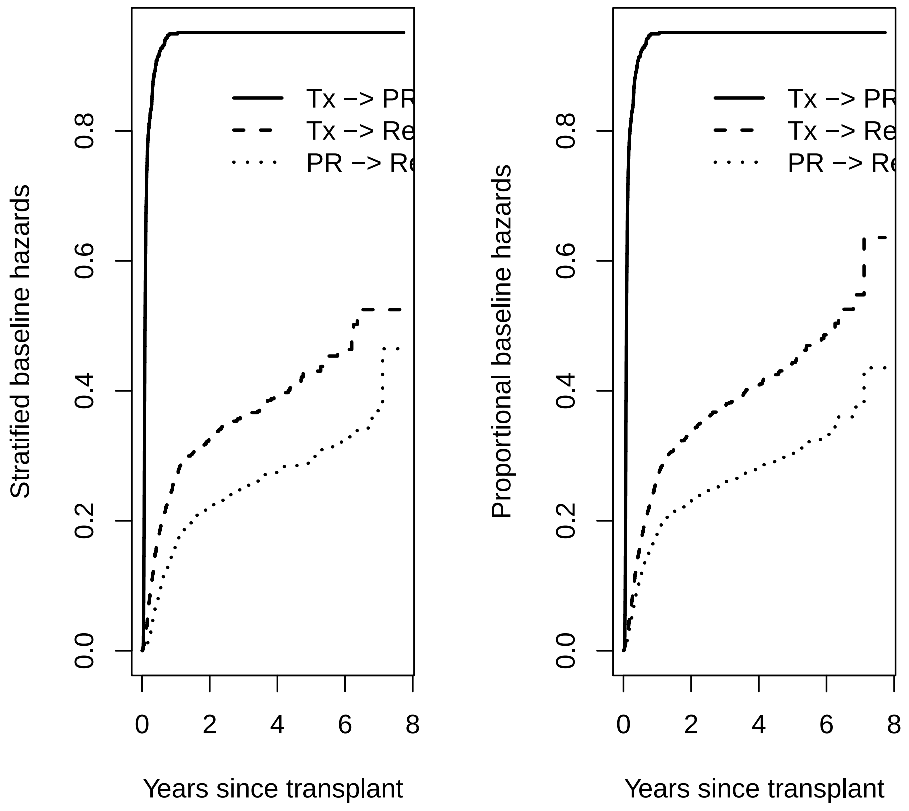

Figure 1: Baseline cumulative hazard curves for the EBMT illness-death model. On the left the Markov stratified hazards model, on the right the Markov PH model.

transition intensities of the transitions out from state g. Finally, $\mathbf{I}$ is the $S \times S$ identity matrix. Equation (1) describes a theoretical relation between the true underlying transition intensities and transition probabilities. The product is a so called product integral (Andersen et al. 1993) when the transition intensities are continuous.

We already have estimates of all the transition intensities. If we gather these in a matrix and plug them in equation (1), we get

$$\hat{\mathbf{P}}(s,t) = \prod_{s<u\leq t} \Big(\mathbf{I} + d\hat{\mathbf{\Lambda}}(u)\Big) \tag{2}$$

as an estimate of the transition probabilities. This estimator is called the Aalen-Johansen estimator, and it is implemented in ***probtrans***. By working with matrices, we immediately get all the transition probabilities from all the starting states g to all the receiving states h in one go. When we fix s, we can calculate all these transition probabilities by forward matrix multiplications using the simple recursive relation

$$\hat{\mathbf{P}}(s,t+) = \hat{\mathbf{P}}(s,t) \cdot \Big(\mathbf{I} + d\hat{\mathbf{\Lambda}}(t+)\Big) \ .$$

Andersen et al. (1993) and de Wreede et al. (2009) also describe recursive formulas for the covariance matrix of $\hat{\mathbf{P}}(s,t)$, with and without covariates, which are implemented in *mstate*.

Let us see all this theory in action and let us recreate Figure 15 of the tutorial. For this we need to calculate transition probabilities for a baseline patient, based on the Markov PH model. We thus use `msf2` as input for ***probtrans***. By default, ***probtrans*** uses forward prediction, which means that s is kept fixed and $t > s$. The argument *predt* specifies either s or t. In this case (forward prediction) it specifies s. From version 0.2.3 on, ***probtrans*** no longer needs a *trans* argument, but takes that from the `trans` item of the *msfit* object.

```
> pt <- probtrans(msf2, predt = 0)
```

The result of ***probtrans*** is a *probtrans* object, which is a list, where item `[[i]]` contains predictions from state i. Each item of the list is a data frame with `time` containing all event time points, and `pstate1`, `pstate2`, etc the probabilities of being in state 1, 2, etc, and finally `se1`, `se2` etc the standard errors of these estimated probabilities. The item `[[3]]` contains predictions $\hat{P}_{3h}(0,t)$ (we chose $s = 0$) starting from the RelDeath state, which is absorbing.

```
> head(pt[[3]])

         time pstate1 pstate2 pstate3 se1 se2 se3
1 0.000000000       0       0       1   0   0   0
2 0.002737851       0       0       1   0   0   0
3 0.008213552       0       0       1   0   0   0
4 0.010951403       0       0       1   0   0   0
5 0.016427105       0       0       1   0   0   0
6 0.019164956       0       0       1   0   0   0
```

```
> tail(pt[[3]])

        time pstate1 pstate2 pstate3 se1 se2 se3
501 6.253251       0       0       1   0   0   0
502 6.357290       0       0       1   0   0   0
503 6.362765       0       0       1   0   0   0
504 6.798084       0       0       1   0   0   0
505 7.110198       0       0       1   0   0   0
506 7.731691       0       0       1   0   0   0
```

We see that these prediction probabilities are not so interesting; the probabilities are all 0 or 1, and, since there is no randomness, all the SE's are 0. Item `[[2]]` contains predictions $\hat{P}_{2h}(0,t)$ from state 2.

It is easier to use the ***summary*** method for *probtrans* objects. The user may specify a *from* argument, specifying from which state the predictions are to be printed. The ***summary*** method prints a selection, the ***head*** and ***tail*** by default unless there are fewer than 12 time points. When *complete* is set to `TRUE`, predictions for all time points are printed. If the *from* argument is missing in the function call, then predictions from all states are printed.

```
> summary(pt, from = 2)

An object of class 'probtrans'

Prediction from state 2 (head and tail):

         time pstate1   pstate2      pstate3 se1          se2          se3
1 0.000000000       0 1.0000000 0.0000000000   0 0.0000000000 0.0000000000
```

```
2 0.002737851       0 0.9997909 0.0002090742   0 0.0002115858 0.0002115858
3 0.008213552       0 0.9997909 0.0002090742   0 0.0002115858 0.0002115858
4 0.010951403       0 0.9995818 0.0004182281   0 0.0003028232 0.0003028232
5 0.016427105       0 0.9991632 0.0008368292   0 0.0004382601 0.0004382601
6 0.019164956       0 0.9987444 0.0012556499   0 0.0005486946 0.0005486946

...

        time pstate1   pstate2   pstate3 se1        se2        se3
501 6.253251       0 0.7079572 0.2920428   0 0.03724432 0.03724432
502 6.357290       0 0.7027899 0.2972101   0 0.03803252 0.03803252
503 6.362765       0 0.6975745 0.3024255   0 0.03881087 0.03881087
504 6.798084       0 0.6870020 0.3129980   0 0.04097391 0.04097391
505 7.110198       0 0.6455554 0.3544446   0 0.05856528 0.05856528
506 7.731691       0 0.6455554 0.3544446   0 0.05856528 0.05856528
```

From state 2 it is only possible to visit state 3 or to remain in state 2. The probability of going to state 1 is 0. The predictions $\hat{P}_{1h}(0,t)$ from state 1 in `[[1]]` are perhaps of most interest here.

```
> summary(pt, from = 1)

An object of class 'probtrans'

Prediction from state 1 (head and tail):

         time   pstate1      pstate2      pstate3          se1          se2
1 0.000000000 1.0000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
2 0.002737851 0.9991669 0.0005277714 0.0003053084 0.0006117979 0.0005285695
3 0.008213552 0.9986390 0.0010556490 0.0003053084 0.0008100529 0.0007492497
4 0.010951403 0.9983340 0.0010554282 0.0006106022 0.0008685356 0.0007490930
5 0.016427105 0.9977235 0.0010549862 0.0012215589 0.0009807157 0.0007487794
6 0.019164956 0.9965843 0.0015830048 0.0018327183 0.0012115670 0.0009191199
          se3
1 0.0000000000
2 0.0003082357
3 0.0003082357
4 0.0004401329
5 0.0006342283
6 0.0007908588

...

        time   pstate1   pstate2   pstate3        se1        se2        se3
501 6.253251 0.2308531 0.4336481 0.3354989 0.02448884 0.02974526 0.03063866
502 6.357290 0.2283925 0.4304829 0.3411246 0.02460675 0.03002904 0.03150500
503 6.362765 0.2259175 0.4272883 0.3467942 0.02472281 0.03031296 0.03234850
504 6.798084 0.2209174 0.4208123 0.3582703 0.02518284 0.03119272 0.03507050
505 7.110198 0.2014549 0.3954248 0.4031203 0.03067690 0.03987257 0.05867417
506 7.731691 0.2014549 0.3954248 0.4031203 0.03067690 0.03987257 0.05867417
```

But we see that we do not have enough information to create Figure 15 of the tutorial, since the probability of the relapse/death state (`pstate3`) does not distinguish between relapse/death before or after platelet recovery. The remedy is actually easy in this case. Consider a different multi-state model with two RelDeath states, the first one (state 3) after platelet recovery, the second one (state 4) without platelet recovery. The transition matrix of this multi-state model is defined as

```
> tmat2 <- transMat(x = list(c(2, 4), c(3), c(), c()))
> tmat2
```

```
         to
from      State 1 State 2 State 3 State 4
  State 1      NA       1      NA       2
  State 2      NA      NA       3      NA
  State 3      NA      NA      NA      NA
  State 4      NA      NA      NA      NA
```

The multi-state model has four states and the same three transitions as before. If we apply ***probtrans*** to this new multi-state model with the same estimated cumulative hazards and standard errors as before, we get exactly what we want. Thus, we just have to call ***probtrans*** with the old `msf2` and the new `tmat2`. From version 0.2.3 on, since the transition matrix is in the *msfit* object, we just need to replace the `trans` item of `msf2` by `tmat2`. In the elements of the resulting lists, `pstate3` will indicate the probability of relapse/death after platelet recovery and `pstate4` the probability of relapse/death without platelet recovery.

```
> msf2$trans <- tmat2
> pt <- probtrans(msf2, predt = 0)
> summary(pt, from = 1)
```

```
An object of class 'probtrans'

Prediction from state 1 (head and tail):

         time   pstate1      pstate2      pstate3      pstate4          se1
1 0.000000000 1.0000000 0.0000000000 0.000000e+00 0.0000000000 0.0000000000
2 0.002737851 0.9991669 0.0005277714 0.000000e+00 0.0003053084 0.0006117979
3 0.008213552 0.9986390 0.0010556490 0.000000e+00 0.0003053084 0.0008100529
4 0.010951403 0.9983340 0.0010554282 2.208393e-07 0.0006103813 0.0008685356
5 0.016427105 0.9977235 0.0010549862 6.628276e-07 0.0012208961 0.0009807157
6 0.019164956 0.9965843 0.0015830048 1.105048e-06 0.0018316132 0.0012115670
           se2          se3          se4
1 0.0000000000 0.000000e+00 0.0000000000
2 0.0005285695 1.116923e-07 0.0003080762
3 0.0007492497 1.116923e-07 0.0003080762
4 0.0007490930 2.989514e-07 0.0004397978
5 0.0007487794 6.308958e-07 0.0006336859
6 0.0009191199 1.032427e-06 0.0007900509

...

        time   pstate1   pstate2   pstate3   pstate4        se1        se2
```

```
501 6.253251 0.2308531 0.4336481 0.1681264 0.1673724 0.02448884 0.02974526
502 6.357290 0.2283925 0.4304829 0.1712916 0.1698330 0.02460675 0.03002904
503 6.362765 0.2259175 0.4272883 0.1744862 0.1723080 0.02472281 0.03031296
504 6.798084 0.2209174 0.4208123 0.1809622 0.1773081 0.02518284 0.03119272
505 7.110198 0.2014549 0.3954248 0.2063497 0.1967706 0.03067690 0.03987257
506 7.731691 0.2014549 0.3954248 0.2063497 0.1967706 0.03067690 0.03987257
           se3        se4
501 0.02379684 0.02100629
502 0.02430502 0.02136056
503 0.02480762 0.02170882
504 0.02616939 0.02264879
505 0.03690104 0.02987965
506 0.03690104 0.02987965
```

The reader may check that the pstate3 and pstate4 probabilities of this new Aalen-Johansen estimator sum up to the pstate3 probability of the result of the previous call to *probtrans*, and that the pstate1 and pstate2 probabilities are unchanged.

Figure 2 contains a plot of pt1. For this we use the *plot* method for *probtrans* objects.

```
> plot(pt, ord = c(2, 3, 4, 1), lwd = 2, xlab = "Years since transplant",
+     ylab = "Prediction probabilities", cex = 0.75, legend = c("Alive in remission, no PR",
+         "Alive in remission, PR", "Relapse or death after PR",
+         "Relapse or death without PR"))
```

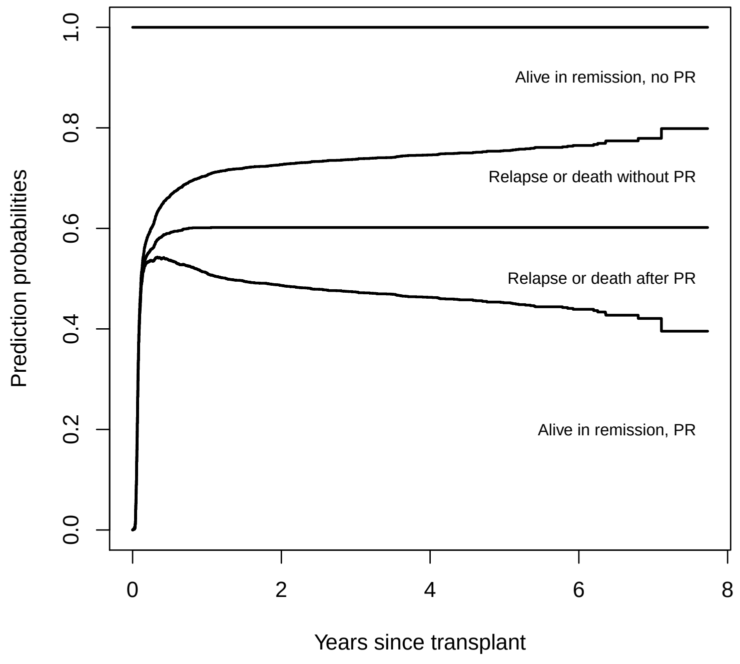

Figure 2: Stacked prediction probabilities at $s = 0$ for a reference patient. PR stands for platelet recovery

The argument *from* determines from which state the transition probabilities are to be plotted. The default is from state 1, which is what we want, so the *from* argument is omitted here. The default *type* of the `plot` method for *probtrans* objects is a "stacked" plot, for which the difference between two adjacent lines represents the probability of being in a state. The argument *ord* specifies the order of the states of which the probabilities are stacked. The present order, 2, 3, 4, 1, allows states 2 and 3 to be combined visually (states with platelet recovery) and states 3 and 4 (death states). Other plot types are "filled", which is like "stacked", but uses colors to fill the space between adjacent lines, "single", which simply plots the transition probabilities as different lines in a single plot, and "separate", which uses separate plots for the transition probabilities.

To obtain the predictions $\hat{P}_{1h}(s,t)$ for $s = 0.5$, which are plotted in Figure 16 of the tutorial, we simply change the value of *predt* in the call to `probtrans`.

```
> pt <- probtrans(msf2, predt = 0.5)
> summary(pt, from = 1)

An object of class 'probtrans'

Prediction from state 1 (head and tail):

       time   pstate1     pstate2      pstate3     pstate4         se1
1 0.5000000 1.0000000 0.000000000 0.000000e+00 0.000000000 0.000000000
2 0.5010267 0.9985898 0.000000000 0.000000e+00 0.001410218 0.003237571
3 0.5037645 0.9976488 0.000000000 0.000000e+00 0.002351164 0.004183373
4 0.5065024 0.9955387 0.001639506 0.000000e+00 0.002821775 0.006169060
5 0.5092402 0.9938957 0.003282495 0.000000e+00 0.002821775 0.007422321
6 0.5119781 0.9915469 0.003277183 5.312169e-06 0.005170580 0.008513835
          se2          se3         se4
1 0.000000000 0.000000e+00 0.000000000
2 0.000000000 0.000000e+00 0.003237571
3 0.000000000 0.000000e+00 0.004183373
4 0.004136138 2.101143e-06 0.004583357
5 0.005848968 2.101143e-06 0.004583357
6 0.005839510 1.353036e-05 0.006209919

...

        time   pstate1    pstate2     pstate3   pstate4        se1        se2
330 6.253251 0.6872018 0.02597812 0.005991102 0.2808290 0.05248379 0.01448894
331 6.357290 0.6798772 0.02578851 0.006180714 0.2881535 0.05348008 0.01438691
332 6.362765 0.6725095 0.02559713 0.006372091 0.2955212 0.05445049 0.01428397
333 6.798084 0.6576254 0.02520918 0.006760043 0.3104053 0.05723289 0.01407791
334 7.110198 0.5996895 0.02368832 0.008280903 0.3683412 0.07993696 0.01332734
335 7.731691 0.5996895 0.02368832 0.008280903 0.3683412 0.07993696 0.01332734
            se3        se4
330 0.003565503 0.05117341
331 0.003675647 0.05224080
332 0.003786522 0.05327926
333 0.004019125 0.05620683
334 0.005060910 0.07944552
```

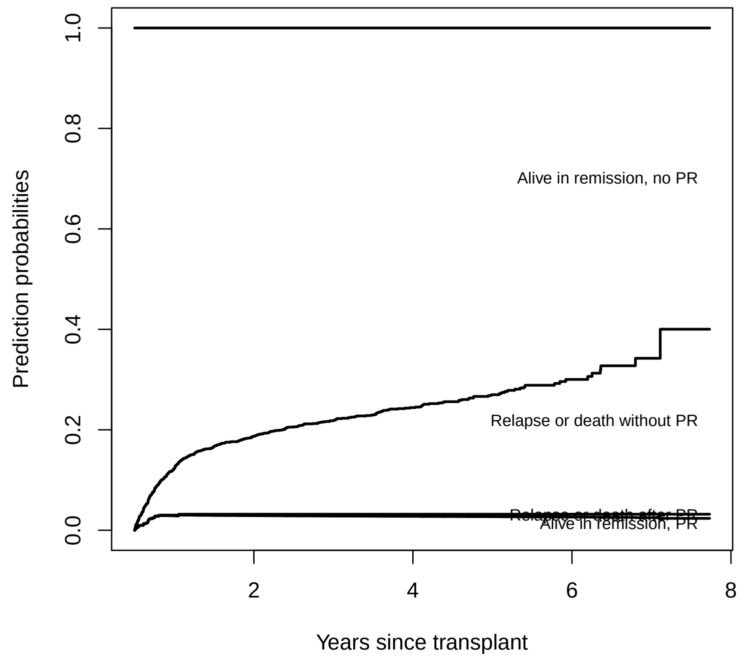

Figure 3: Stacked prediction probabilities at $s = 0.5$ for a reference patient

```
335 0.005060910 0.07944552
```

The result now contains only time points $t \geq 0.5$. Figure 3 contains a plot of `pt1`.

```
> plot(pt, ord = c(2, 3, 4, 1), lwd = 2, xlab = "Years since transplant",
+     ylab = "Prediction probabilities", cex = 0.75, legend = c("Alive in remission, no PR",
+         "Alive in remission, PR", "Relapse or death after PR",
+         "Relapse or death without PR"))
```

Figure 17 of the tutorial distinguishes between three patients, one being the good old (or rather young) reference patient, for which we have already calculated the probabilities, one for a patient in the age category 20-40, and one for a patient older than 40. To obtain prediction probabilities for the latter two patients as well, we have to repeat part of the calculations, changing only the value of age in the `newdata` data frame.

```
> msf2$trans <- tmat
> msf.20 <- msf2 # copy msfit result for reference (young) patient
> newd <- newd[,1:5] # use the basic covariates of the reference patient
> newd2 <- newd
> newd2$age <- 1
> newd2$age <- factor(newd2$age,levels=0:2,labels=levels(ebmt3$age))
> attr(newd2, "trans") <- tmat
> class(newd2) <- c("msdata","data.frame")
> newd2 <- expand.covs(newd2,covs[1:4],longnames=FALSE)
> newd2$strata=c(1,2,2)
> newd2$pr <- c(0,0,1)
> msf.2040 <- msfit(c2, newdata=newd2, trans=tmat)
> newd3 <- newd
> newd3$age <- 2
> newd3$age <- factor(newd3$age,levels=0:2,labels=levels(ebmt3$age))
> attr(newd3, "trans") <- tmat
> class(newd3) <- c("msdata","data.frame")
> newd3 <- expand.covs(newd3,covs[1:4],longnames=FALSE)
> newd3$strata=c(1,2,2)
> newd3$pr <- c(0,0,1)
> msf.40 <- msfit(c2, newdata=newd3, trans=tmat)
> pt.20 <- probtrans(msf.20,predt=0) # original young (<= 20) patient
> pt.201 <- pt.20[[1]]; pt.202 <- pt.20[[2]]
> pt.2040 <- probtrans(msf.2040,predt=0) # patient 20-40
> pt.20401 <- pt.2040[[1]]; pt.20402 <- pt.2040[[2]]
> pt.40 <- probtrans(msf.40,predt=0) # patient > 40
> pt.401 <- pt.40[[1]]; pt.402 <- pt.40[[2]]
```

The 5-years transition probabilities $P_{13}(0,5)$ and $P_{23}(0,5)$ are estimated as 0.30275 and 0.26210 respectively.

```
> pt.201[488:489,] # 5 years falls between 488th and 489th time point

        time   pstate1   pstate2   pstate3        se1        se2        se3
488 4.985626 0.2452605 0.4519872 0.3027523 0.02411439 0.02853645 0.02693539
489 5.084189 0.2445602 0.4511034 0.3043365 0.02412385 0.02858110 0.02707436
```

```
> pt.202[488:489,] # 5-years probabilities

        time pstate1   pstate2   pstate3 se1        se2        se3
488 4.985626       0 0.7378970 0.2621030   0 0.03339911 0.03339911
489 5.084189       0 0.7364541 0.2635459   0 0.03356217 0.03356217
```

Figure 4 shows relapse-free survival probabilities without distinction between before or after platelet recovery, so we can use the first transition matrix `tmat`. The probabilities we want are $1 - \hat{P}_{13}(0,t)$ and $1 - \hat{P}_{23}(0,t)$, the first one conditioning on being in state 1 (transplantation, i.e. no PR), the second in being in state 2 (PR).

```
> plot(pt.201$time, 1 - pt.201$pstate3, ylim = c(0.425, 1), type = "s",
+     lwd = 2, col = "red", xlab = "Years since transplant", ylab = "Relapse-free survival")
> lines(pt.20401$time, 1 - pt.20401$pstate3, type = "s", lwd = 2,
+     col = "blue")
> lines(pt.401$time, 1 - pt.401$pstate3, type = "s", lwd = 2, col = "green")
> lines(pt.202$time, 1 - pt.202$pstate3, type = "s", lwd = 2, col = "red",
+     lty = 2)
> lines(pt.20402$time, 1 - pt.20402$pstate3, type = "s", lwd = 2,
+     col = "blue", lty = 2)
> lines(pt.402$time, 1 - pt.402$pstate3, type = "s", lwd = 2, col = "green",
+     lty = 2)
> legend(6, 1, c("no PR", "PR"), lwd = 2, lty = 1:2, xjust = 1,
+     bty = "n")
> legend("topright", c("<=20", "20-40", ">40"), lwd = 2, col = c("red",
+     "blue", "green"), bty = "n")
```

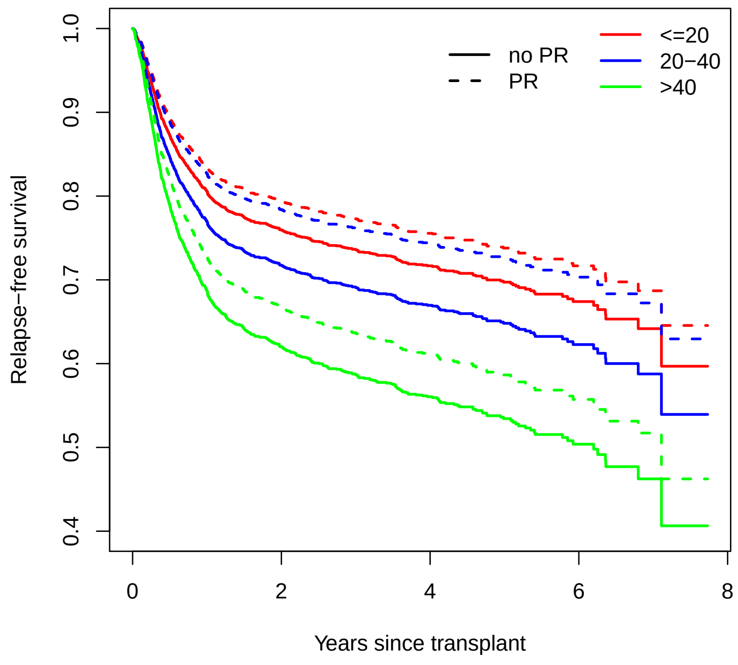

Figure 4: Predicted relapse-free survival probabilities for three patients in different age categories, given platelet recovery (dashed) and given no platelet recovery (solid). The time of prediction was at transplant (note: in the tutorial this was at 1 month after transplant).

It is also possible to do prediction with a fixed horizon. This should not be understood as attempting to predict the past. It means that in our prediction probabilities $P_{gh}(s,t)$, we fix t, a time horizon, and we want to study how $P_{gh}(s,t)$ changes as more and more information on a patient becomes available. From a computational point of view this just means that the order of the matrix multiplication in (2) is reversed. We will plot $1-\hat{P}_{13}(s,5)$ and $1-\hat{P}_{23}(s,5)$, the 5-years relapse-free survival probabilities given that the patient is in state 1 (no PR) and in state 2 (PR), respectively, for the same three patients as before.

```
> pt.20 <- probtrans(msf.20, direction = "fixedhorizon", predt = 5)
> pt.201 <- pt.20[[1]]
> pt.202 <- pt.20[[2]]
> head(pt.201)

         time   pstate1   pstate2   pstate3        se1        se2        se3
1 0.000000000 0.2452605 0.4519872 0.3027523 0.02411439 0.02853645 0.02693539
2 0.002737851 0.2454650 0.4519742 0.3025608 0.02413403 0.02854695 0.02694328
3 0.008213552 0.2455948 0.4518230 0.3025823 0.02414644 0.02854909 0.02694380
4 0.010951403 0.2456698 0.4519611 0.3023691 0.02415369 0.02855746 0.02695114
5 0.016427105 0.2458201 0.4522376 0.3019422 0.02416821 0.02857418 0.02696574
6 0.019164956 0.2461011 0.4523628 0.3015361 0.02419520 0.02859303 0.02698076

> head(pt.202)

         time pstate1   pstate2   pstate3 se1        se2        se3
1 0.000000000       0 0.7378970 0.2621030   0 0.03339911 0.03339911
2 0.002737851       0 0.7380513 0.2619487   0 0.03340572 0.03340572
3 0.008213552       0 0.7380513 0.2619487   0 0.03340572 0.03340572
4 0.010951403       0 0.7382057 0.2617943   0 0.03341233 0.03341233
5 0.016427105       0 0.7385150 0.2614850   0 0.03342551 0.03342551
6 0.019164956       0 0.7388247 0.2611753   0 0.03343863 0.03343863
```

Here item `[[1]]` gives estimates $\hat{P}_{1h}(s,5)$ and `[[2]]` gives estimates $\hat{P}_{2h}(s,5)$. For item `[[g]]`, the column `time` gives the different values of s and `pstate1` etc give the estimated probabilities of being in state 1 etc at 5 years, conditional on being in state g at time s. In `pt.201` we recognize at `time` (s)=0) 0.30275 as $\hat{P}_{1h}(0,5)$ and in `pt.202` we see 0.26210 as $\hat{P}_{2h}(0,5)$. The backward transition probabilities for the other two patients are calculated similarly.

```
> pt.2040 <- probtrans(msf.2040, direction = "fixedhorizon", predt = 5)
> pt.20401 <- pt.2040[[1]]
> pt.20402 <- pt.2040[[2]]
> pt.40 <- probtrans(msf.40, direction = "fixedhorizon", predt = 5)
> pt.401 <- pt.40[[1]]
> pt.402 <- pt.40[[2]]
```

As mentioned before, in $s=0$, these probabilities are the same as the five-years probabilities of Figure 4, and as s approaches 5, the probabilities approach 1, since both $\hat{P}_{13}(s,5)$ and $\hat{P}_{23}(s,5)$ approach 0. Figure 5 shows 5-years relapse-free survival probabilities, both with and without platelet recovery, with the prediction time s varying.

```
> plot(pt.201$time, 1 - pt.201$pstate3, ylim = c(0.425, 1), type = "s",
+     lwd = 2, col = "red", xlab = "Years since transplant", ylab = "Relapse-free survival")
```

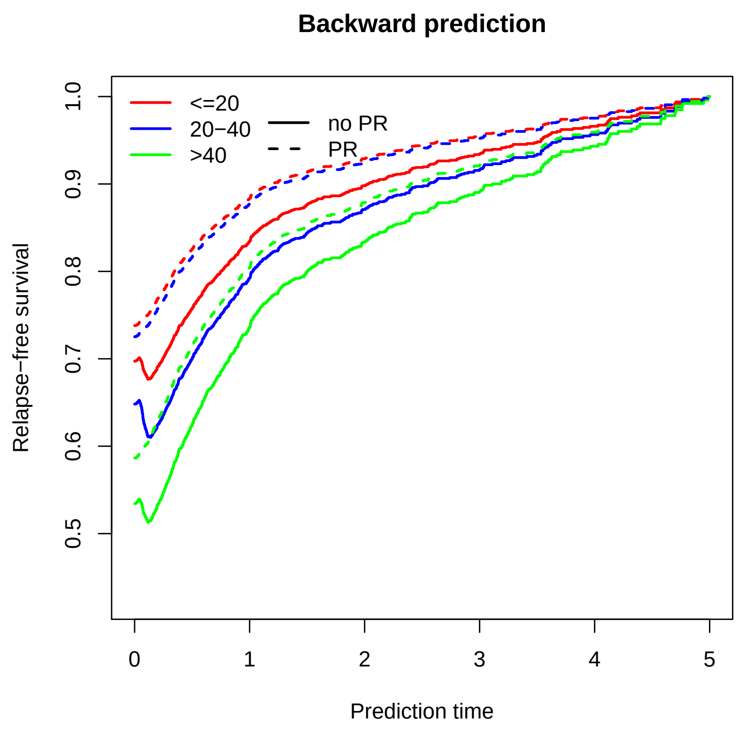

Figure 5: Predicted probabilities of 5-years relapse-free survival, conditional on being alive without relapse with (PR) and without platelet recovery (no PR). Patients in three age categories.

```
> lines(pt.20401$time, 1 - pt.20401$pstate3, type = "s", lwd = 2,
+     col = "blue")
> lines(pt.401$time, 1 - pt.401$pstate3, type = "s", lwd = 2, col = "green")
> lines(pt.202$time, 1 - pt.202$pstate3, type = "s", lwd = 2, col = "red",
+     lty = 2)
> lines(pt.20402$time, 1 - pt.20402$pstate3, type = "s", lwd = 2,
+     col = "blue", lty = 2)
> lines(pt.402$time, 1 - pt.402$pstate3, type = "s", lwd = 2, col = "green",
+     lty = 2)
> legend("topleft", c("<=20", "20-40", ">40"), lwd = 2, col = c("red",
+     "blue", "green"), bty = "n")
> legend(1, 1, c("no PR", "PR"), lwd = 2, lty = 1:2, bty = "n")
> title(main = "Backward prediction")
```

5 Competing risks

The data used in Section 3 of the tutorial is available in *mstate* under the name `aidssi`. See the help file for more information.

```
> data(aidssi)
> si <- aidssi # Just a shorter name
> head(si)
```

```
  patnr   time status      cause ccr5
1     1  9.106      1       AIDS   WW
2     2 11.039      0 event-free   WM
3     3  2.234      1       AIDS   WW
4     4  9.878      2         SI   WM
5     5  3.819      1       AIDS   WW
6     6  6.801      1       AIDS   WW
```

```
> table(si$status)
```

```
  0   1   2
107 114 108
```

To prepare data in long format, it is possible to use *msprep*. In this case there is not a huge advantage in using *msprep*; the long data may just as easily be prepared directly. Nevertheless we will illustrate the use of *msprep* to obtain data in long format. The function *trans.comprisk* prepares a transition matrix for competing risks models. The first argument is the number of causes of failure; in the *names* argument a character vector of length three (the total number of states in the multi-state model including the failure-free state) may be given. The transition matrix has three states with stte 1 being the failure-free state and the subsequent sttes representing the different causes of failure.

```
> tmat <- trans.comprisk(2, names = c("event-free", "AIDS", "SI"))
> tmat
```

```
            to
from         event-free AIDS SI
  event-free         NA    1  2
  AIDS               NA   NA NA
  SI                 NA   NA NA
```

Now follows the actual call to *msprep*.

```
> si$stat1 <- as.numeric(si$status == 1)
> si$stat2 <- as.numeric(si$status == 2)
> silong <- msprep(time = c(NA, "time", "time"), status = c(NA,
+     "stat1", "stat2"), data = si, keep = "ccr5", trans = tmat)
```

We can use *events* to check whether the number of events from original data (`si`) corresponds with long data.

```
> events(silong)
```

```
$Frequencies
            to
from         event-free AIDS  SI no event total entering
  event-free          0  114 108      107            329
  AIDS                0    0   0      114            114
  SI                  0    0   0      108            108

$Proportions
            to
```

```
from         event-free      AIDS        SI  no event
  event-free  0.0000000 0.3465046 0.3282675 0.3252280
  AIDS        0.0000000 0.0000000 0.0000000 1.0000000
  SI          0.0000000 0.0000000 0.0000000 1.0000000
```

For the regression analyses to be performed later we add transition-specific covariates. In the context of competing risks one could call them cause-specific covariates. Since the factor levels of CCR5 are quite short we keep the default setting (`TRUE`) of *longnames*.

```
> silong <- expand.covs(silong, "ccr5")
> silong[1:8, ]
```

```
An object of class 'msdata'

Data:
  id from to trans Tstart  Tstop   time status ccr5 ccr5WM.1 ccr5WM.2
1  1    1  2     1      0  9.106  9.106      1   WW        0        0
2  1    1  3     2      0  9.106  9.106      0   WW        0        0
3  2    1  2     1      0 11.039 11.039      0   WM        1        0
4  2    1  3     2      0 11.039 11.039      0   WM        0        1
5  3    1  2     1      0  2.234  2.234      1   WW        0        0
6  3    1  3     2      0  2.234  2.234      0   WW        0        0
7  4    1  2     1      0  9.878  9.878      0   WM        1        0
8  4    1  3     2      0  9.878  9.878      1   WM        0        1
```

To illustrate the fact that naive Kaplan-Meiers are biased estimators of the probabilities of failing from the different causes of failure, we just make use of the functions in the *survival* package. I am using `coxph` below, probably this could be done quicker.

```
> c1 <- coxph(Surv(time, status) ~ 1, data = silong, subset = (trans ==
+     1), method = "breslow")
> c2 <- coxph(Surv(time, status) ~ 1, data = silong, subset = (trans ==
+     2), method = "breslow")
> h1 <- survfit(c1)
> h1 <- data.frame(time = h1$time, surv = h1$surv)
> h2 <- survfit(c2)
> h2 <- data.frame(time = h2$time, surv = h2$surv)
```

These naive Kaplan-Meier curves are shown in Figure 6 (Figure 2 in the tutorial). The Kaplan-Meier estimate of AIDS is plotted as a survival curve, while that of SI appearance is shown as a distribution function. There is some extra code to chop the time at 13 years. This was just done to make the picture prettier.

```
> idx1 <- (h1$time<13) # this restricts the plot to the first 13 years
> plot(c(0,h1$time[idx1],13),c(1,h1$surv[idx1],min(h1$surv[idx1])),type="s",
+     xlim=c(0,13),ylim=c(0,1),xlab="Years from HIV infection",ylab="Probability",lwd=2)
> idx2 <- (h2$time<13)
> lines(c(0,h2$time[idx2],13),c(0,1-h2$surv[idx2],max(1-h2$surv[idx2])),type="s",lwd=2)
> text(8,0.71,adj=0,"AIDS")
> text(8,0.32,adj=0,"SI")
```

Cumulative incidence functions can be computed using the function `Cuminc`. It takes as main arguments *time* and *status*, which can be provided as vectors

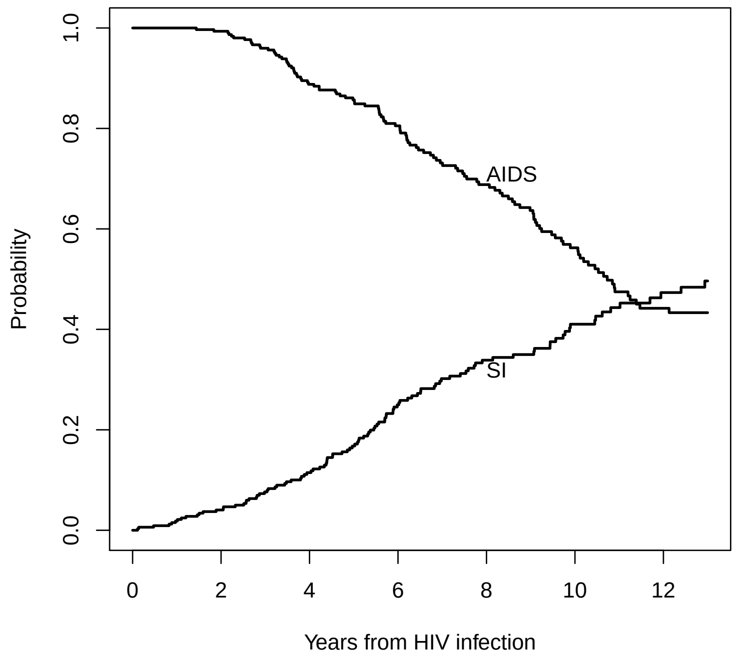

Figure 6: Estimated survival curve for AIDS and probability of SI appearance, based on the naive Kaplan-Meier estimator.

```
> ci <- Cuminc(time = si$time, status = si$status)
```

or, alternatively, as column names representing time and status, along with a *data* argument containing these column names.

```
> ci <- Cuminc(time = "time", status = "status", data = aidssi)
```

The result is a data frame containing the failure-free probabilities (`Surv`) and the cumulative incidence functions with their standard errors. Other arguments allow to specify the codes for the causes of failure and a group identifier.

```
> head(ci)

   time      Surv CI.1        CI.2      seSurv seCI.1      seCI.2
1 0.112 0.9969605    0 0.003039514 0.003034891      0 0.003034891
2 0.137 0.9939210    0 0.006079027 0.004285436      0 0.004285436
3 0.474 0.9908628    0 0.009137246 0.005251290      0 0.005251290
4 0.824 0.9877760    0 0.012224046 0.006074796      0 0.006074796
5 0.884 0.9846795    0 0.015320522 0.006799283      0 0.006799283
6 0.969 0.9815830    0 0.018416998 0.007449696      0 0.007449696
```

```
> tail(ci)

      time      Surv      CI.1      CI.2     seSurv     seCI.1     seCI.2
211 11.943 0.2312339 0.4035707 0.3651954 0.02638091 0.02978948 0.02881464
```

```
212 12.129 0.2266092 0.4081954 0.3651954 0.02625552 0.02989297 0.02881464
213 12.400 0.2219845 0.4081954 0.3698201 0.02612382 0.02989297 0.02896110
214 12.936 0.2165702 0.4081954 0.3752344 0.02604167 0.02989297 0.02919663
215 13.361 0.2067261 0.4180395 0.3752344 0.02665370 0.03089977 0.02919663
216 13.936 0.0000000 0.4180395 0.5819605 0.00000000 0.03089977 0.03089977
```

The cumulative incidence functions just obtained can be used to reproduce Figure 3 of the tutorial. The plots are shown in Figure 7.

```
> idx0 <- (ci$time < 13)
> plot(c(0, ci$time[idx0], 13), c(1, 1 - ci$CI.1[idx0], min(1 -
+     ci$CI.1[idx0])), type = "s", xlim = c(0, 13), ylim = c(0,
+     1), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> idx1 <- (h1$time < 13)
> lines(c(0, h1$time[idx1], 13), c(1, h1$surv[idx1], min(h1$surv[idx1])),
+     type = "s", lwd = 2, col = 8)
> lines(c(0, ci$time[idx0], 13), c(0, ci$CI.2[idx0], max(ci$CI.2[idx0])),
+     type = "s", lwd = 2)
> idx2 <- (h2$time < 13)
> lines(c(0, h2$time[idx2], 13), c(0, 1 - h2$surv[idx2], max(1 -
+     h2$surv[idx2])), type = "s", lwd = 2, col = 8)
> text(8, 0.77, adj = 0, "AIDS")
> text(8, 0.275, adj = 0, "SI")
```

The stacked plots of Figure 4 of the tutorial are shown in Figure 8.

```
> idx0 <- (ci$time < 13)
> plot(c(0, ci$time[idx0]), c(0, ci$CI.1[idx0]), type = "s", xlim = c(0,
+     13), ylim = c(0, 1), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> lines(c(0, ci$time[idx0]), c(0, ci$CI.1[idx0] + ci$CI.2[idx0]),
+     type = "s", lwd = 2)
> text(13, 0.5 * max(ci$CI.1[idx0]), adj = 1, "AIDS")
> text(13, max(ci$CI.1[idx0]) + 0.5 * max(ci$CI.2[idx0]), adj = 1,
+     "SI")
> text(13, 0.5 + 0.5 * max(ci$CI.1[idx0]) + 0.5 * max(ci$CI.2[idx0]),
+     adj = 1, "Event-free")
```

Regression

The section on regression in the tutorial already shows some R code and occasional output. Because of the fact that I used *msprep* to prepare the long data, occasionally there will be very small differences with the code in the tutorial. We start with regression on cause-specific hazards. Using the original dataset, we can apply ordinary Cox regression for cause 1 (AIDS), taking only the AIDS cases as events. This is done by specifying `status==1` below (observations with status=0 (true censorings) and status=2 (SI) are treated as censorings). Similarly for cause 2 (SI appearance), where `status==2` indicates that only failures due to SI appearance are to be treated as events.

```
> coxph(Surv(time, status == 1) ~ ccr5, data = si) # AIDS
```

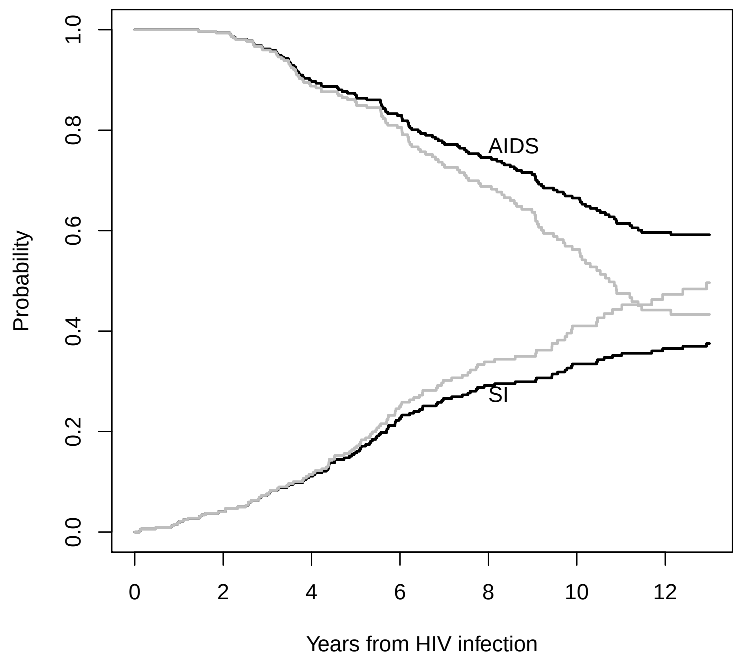

Figure 7: Estimates of probabilities of AIDS and SI appearance, based on the naive Kaplan-Meier (grey) and on cumulative incidence functions (black).

```
Call:
coxph(formula = Surv(time, status == 1) ~ ccr5, data = si)

         coef exp(coef) se(coef)     z       p
ccr5WM -1.236     0.291    0.307 -4.02 5.7e-05

Likelihood ratio test=22  on 1 df, p=2.76e-06
n= 324, number of events= 113
   (5 observations deleted due to missingness)

> coxph(Surv(time, status == 2) ~ ccr5, data = si) # SI appearance

Call:
coxph(formula = Surv(time, status == 2) ~ ccr5, data = si)

         coef exp(coef) se(coef)     z    p
ccr5WM -0.254     0.776    0.238 -1.07 0.29

Likelihood ratio test=1.19  on 1 df, p=0.275
n= 324, number of events= 107
   (5 observations deleted due to missingness)
```

The same analysis can be performed using the long format dataset `silong` in several ways. For instance, as separate Cox regressions.

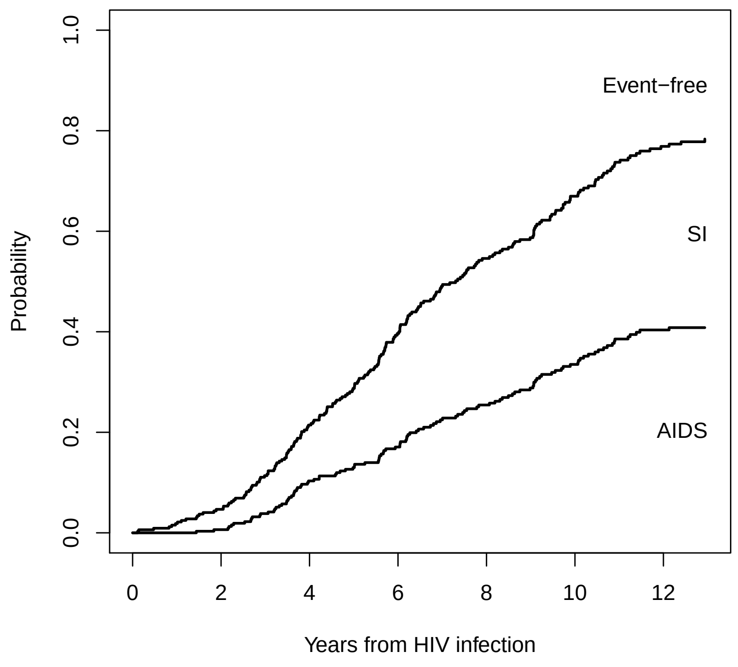

Figure 8: Cumulative incidence curves of AIDS and SI appearance. The cumulative incidence functions are stacked; the distances between two curves represent the probabilities of the different events.

```
> coxph(Surv(time, status) ~ ccr5, data = silong, subset = (trans ==
+     1), method = "breslow")

Call:
coxph(formula = Surv(time, status) ~ ccr5, data = silong, subset = (trans ==
    1), method = "breslow")

         coef exp(coef) se(coef)     z    p
ccr5WM -0.604     0.546    0.235 -2.57 0.01

Likelihood ratio test=7.4  on 1 df, p=0.00654
n= 323, number of events= 113
   (6 observations deleted due to missingness)

> coxph(Surv(time, status) ~ ccr5, data = silong, subset = (trans ==
+     2), method = "breslow")

Call:
coxph(formula = Surv(time, status) ~ ccr5, data = silong, subset = (trans ==
    2), method = "breslow")

         coef exp(coef) se(coef)    z    p
ccr5WM -0.357     0.700    0.238 -1.5 0.13

Likelihood ratio test=2.39  on 1 df, p=0.122
n= 323, number of events= 104
   (6 observations deleted due to missingness)
```

And in a single analysis, using the expanded covariates.

```
> coxph(Surv(time, status) ~ ccr5WM.1 + ccr5WM.2 + strata(trans),
+     data = silong)

Call:
coxph(formula = Surv(time, status) ~ ccr5WM.1 + ccr5WM.2 + strata(trans),
    data = silong)

           coef exp(coef) se(coef)     z    p
ccr5WM.1 -0.604     0.546    0.235 -2.58 0.01
ccr5WM.2 -0.357     0.700    0.238 -1.50 0.13

Likelihood ratio test=9.79  on 2 df, p=0.00749
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

The same model, but now using a covariate by cause interaction.

```
> coxph(Surv(time, status) ~ ccr5 * factor(trans) + strata(trans),
+     data = silong)
```

```
Call:
coxph(formula = Surv(time, status) ~ ccr5 * factor(trans) + strata(trans),
    data = silong)

                        coef exp(coef) se(coef)     z    p
ccr5WM                -0.604     0.546    0.235 -2.58 0.01
factor(trans)2            NA        NA    0.000    NA   NA
ccr5WM:factor(trans)2  0.248     1.281    0.334  0.74 0.46

Likelihood ratio test=9.79  on 2 df, p=0.00749
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

In the model below we assume that the effect of CCR5 on the two cause-specific hazards is equal. The significant effect of the interaction in the model we just saw indicates that this is not a good idea. But, again, this is just for educational purposes.

```
> coxph(Surv(time, status) ~ ccr5 + strata(trans), data = silong)

Call:
coxph(formula = Surv(time, status) ~ ccr5 + strata(trans), data = silong)

         coef exp(coef) se(coef)     z      p
ccr5WM -0.487     0.615    0.167 -2.91 0.0036

Likelihood ratio test=9.24  on 1 df, p=0.00237
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

There are two alternative ways yielding the same result. First, we can actually leave out the *strata* term.

```
> coxph(Surv(time, status) ~ ccr5, data = silong)

Call:
coxph(formula = Surv(time, status) ~ ccr5, data = silong)

         coef exp(coef) se(coef)     z      p
ccr5WM -0.489     0.613    0.167 -2.93 0.0034

Likelihood ratio test=9.34  on 1 df, p=0.00225
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

Second, since the *strata* term is not needed we can use `si`.

```
> coxph(Surv(time, status != 0) ~ ccr5, data = si)

Call:
coxph(formula = Surv(time, status != 0) ~ ccr5, data = si)
```

```
         coef exp(coef) se(coef)     z       p
ccr5WM -0.701     0.496    0.186 -3.77 0.00016

Likelihood ratio test=16.5  on 1 df, p=4.95e-05
n= 324, number of events= 220
   (5 observations deleted due to missingness)
```

Note: the actual estimated baseline hazards may be different, whether or not the strata term is used.

Assuming that baseline hazards for AIDS and SI are proportional (this is generally not a realistic assumption by the way, but just for illustration purposes).

```
> coxph(Surv(time, status) ~ ccr5WM.1 + ccr5WM.2 + factor(trans),
+     data = silong)

Call:
coxph(formula = Surv(time, status) ~ ccr5WM.1 + ccr5WM.2 + factor(trans),
    data = silong)

                 coef exp(coef) se(coef)     z     p
ccr5WM.1       -0.560     0.571    0.234 -2.39 0.017
ccr5WM.2       -0.415     0.661    0.237 -1.75 0.080
factor(trans)2 -0.120     0.887    0.153 -0.78 0.434

Likelihood ratio test=9.96  on 3 df, p=0.0189
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

Or, again using covariate by cause (transition) interaction.

```
> coxph(Surv(time, status) ~ ccr5 * factor(trans), data = silong)

Call:
coxph(formula = Surv(time, status) ~ ccr5 * factor(trans), data = silong)

                        coef exp(coef) se(coef)     z     p
ccr5WM                -0.560     0.571    0.234 -2.39 0.017
factor(trans)2        -0.120     0.887    0.153 -0.78 0.434
ccr5WM:factor(trans)2  0.146     1.157    0.332  0.44 0.661

Likelihood ratio test=9.96  on 3 df, p=0.0189
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

Note that, even though patients are replicated in the long format, it is not necessary to use robust standard errors. Any of the previous analyses with the silong dataset gives identical results when a cluster(id) term is added. For instance,

```
> coxph(Surv(time, status) ~ ccr5 * factor(trans) + cluster(id),
+     data = silong)
```

```
Call:
coxph(formula = Surv(time, status) ~ ccr5 * factor(trans) + cluster(id),
    data = silong)

                        coef exp(coef) se(coef) robust se     z     p
ccr5WM                -0.560     0.571    0.234     0.228 -2.46 0.014
factor(trans)2        -0.120     0.887    0.153     0.153 -0.78 0.434
ccr5WM:factor(trans)2  0.146     1.157    0.332     0.313  0.46 0.642

Likelihood ratio test=9.96  on 3 df, p=0.0189
n= 646, number of events= 217
   (12 observations deleted due to missingness)
```

gives the same result as before.

So far in the regression context we have just used the `coxph` function of the *survival* package. In order to obtain predicted cumulative incidences, `msprep` is useful. First let us store our analysis with separate covariate effects for the two causes.

```
> c1 <- coxph(Surv(time, status) ~ ccr5WM.1 + ccr5WM.2 + strata(trans),
+     data = silong, method = "breslow")
```

If we want the predicted cumulative incidences for an individual with CCR5 wild-type (WW), we make a *newdata* data frame containing the (transition-specific) covariate values for each of the transitions for the individual of interest. Then we apply `msfit` as illustrated earlier in the context of multi-state models.

```
> WW <- data.frame(ccr5WM.1 = c(0, 0), ccr5WM.2 = c(0, 0), trans = c(1,
+     2), strata = c(1, 2))
> msf.WW <- msfit(c1, WW, trans = tmat)
```

And finally, to obtain the cumulative incidences we apply `probtrans`. Item `[[1]]` is selected because the prediction starts from state 1 (event-free) at time $s = 0$.

```
> pt.WW <- probtrans(msf.WW, 0)[[1]]
```

Similarly for an individual with the CCR5 mutant (WM) genotype.

```
> WM <- data.frame(ccr5WM.1 = c(1, 0), ccr5WM.2 = c(0, 1), trans = c(1,
+     2), strata = c(1, 2))
> msf.WM <- msfit(c1, WM, trans = tmat)
> pt.WM <- probtrans(msf.WM, 0)[[1]]
```

We now plot these cumulative incidence curves for AIDS (`pstate2`) and SI appearance (`pstate3`), for wild-type (WW) and mutant (WM) in Figure 9 (Figure 5 in the tutorial).

```
> idx1 <- (pt.WW$time < 13)
> idx2 <- (pt.WM$time < 13)
> plot(c(0, pt.WW$time[idx1]), c(0, pt.WW$pstate2[idx1]), type = "s",
+     ylim = c(0, 0.5), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> lines(c(0, pt.WM$time[idx2]), c(0, pt.WM$pstate2[idx2]), type = "s",
+     lwd = 2, col = 8)
> title(main = "AIDS")
```

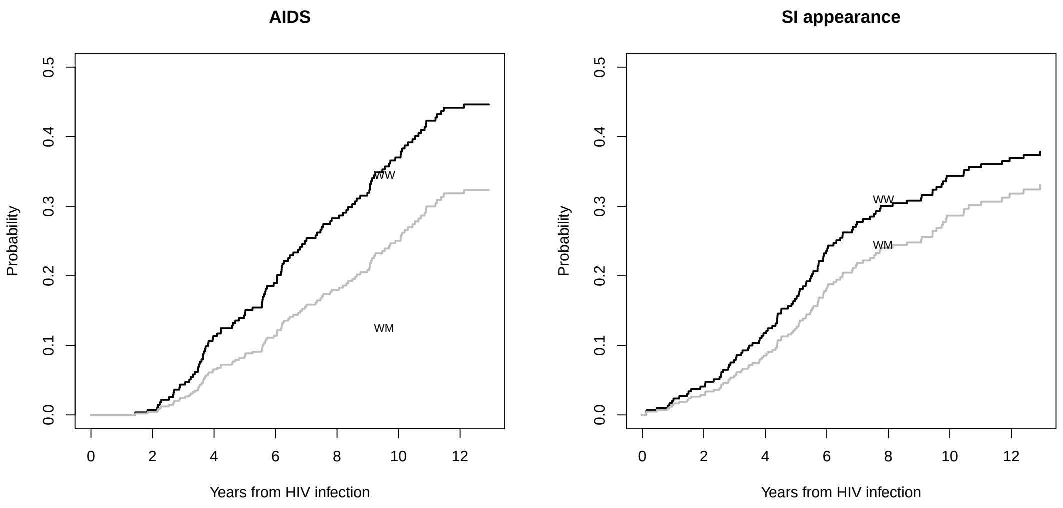

Figure 9: Cumulative incidence functions for AIDS (left) and SI appearance (right), for wild-type (`WW`) and mutant (`WM`) CCR5 genotype, based on a proportional hazards model on the cause-specific hazards.

```
> text(9.2, 0.345, "WW", adj = 0, cex = 0.75)
> text(9.2, 0.125, "WM", adj = 0, cex = 0.75)
> plot(c(0, pt.WW$time[idx1]), c(0, pt.WW$pstate3[idx1]), type = "s",
+     ylim = c(0, 0.5), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> lines(c(0, pt.WM$time[idx2]), c(0, pt.WM$pstate3[idx2]), type = "s",
+     lwd = 2, col = 8)
> title(main = "SI appearance")
> text(7.5, 0.31, "WW", adj = 0, cex = 0.75)
> text(7.5, 0.245, "WM", adj = 0, cex = 0.75)
```

The illustration of the phenomenon that the same cause-specific hazard ratio may have different effects on the cumulative incidences (Figure 7 in the tutorial) may be performed as well, by replacing the appropriate parts of the cumulative hazard of AIDS (`trans`=1), and calling ***probtrans***. We are interested in SI appearance and adjust the hazards of the competing risk (AIDS) while keeping the remainder the same (Figure 7 in the tutorial). The result is shown in Figure 10. We multiply the baseline hazard of AIDS with factors (`ff` = 0, 0.5, 1, 1.5, 2, 4).

```
> ffs <- c(0, 0.5, 1, 1.5, 2, 4)
> newmsf.WW <- msf.WW
> newmsf.WM <- msf.WM
> par(mfrow = c(2, 3))
> for (ff in ffs) {
+     newmsf.WW$Haz$Haz[newmsf.WW$Haz$trans == 1] <- ff * msf.WW$Haz$Haz[msf.WW$Haz$trans ==
+         1]
+     pt.WW <- probtrans(newmsf.WW, 0, variance = FALSE)[[1]]
+     newmsf.WM$Haz$Haz[newmsf.WM$Haz$trans == 1] <- ff * msf.WM$Haz$Haz[msf.WM$Haz$trans ==
+         1]
+     pt.WM <- probtrans(newmsf.WM, 0, variance = FALSE)[[1]]
+     idx1 <- (pt.WW$time < 13)
```

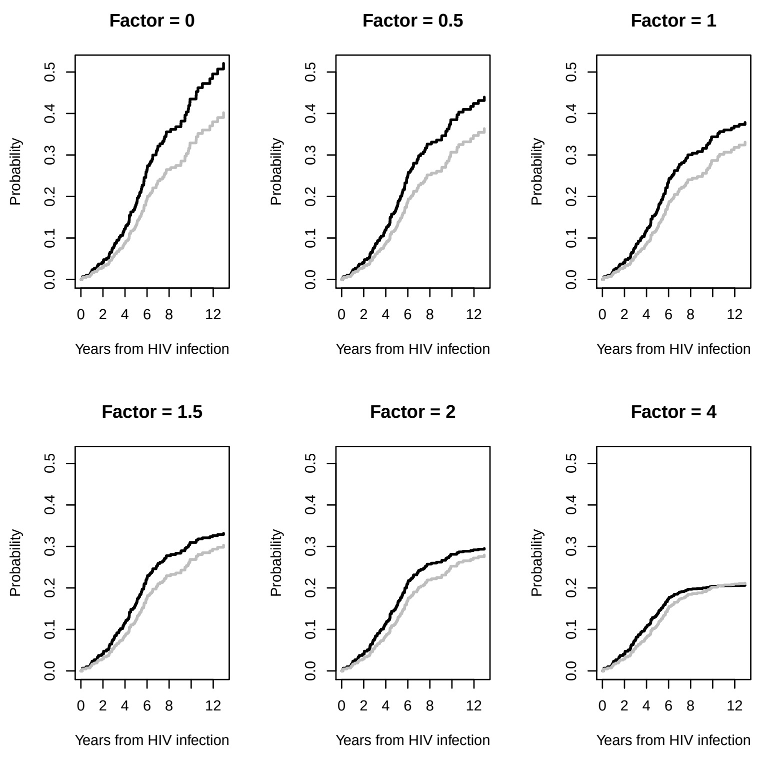

Figure 10: Cumulative incidence functions for Si appearance, for CCR5 wild-type `WW` (black) and mutant `WM` (grey). The baseline hazard of AIDS was multiplied with different factors, while keeping everything else the same.

```
+     idx2 <- (pt.WM$time < 13)
+     plot(c(0, pt.WW$time[idx1]), c(0, pt.WW$pstate3[idx1]), type = "s",
+         ylim = c(0, 0.52), xlab = "Years from HIV infection",
+         ylab = "Probability", lwd = 2)
+     lines(c(0, pt.WM$time[idx2]), c(0, pt.WM$pstate3[idx2]),
+         type = "s", lwd = 2, col = 8)
+     title(main = paste("Factor =", ff))
+ }
> par(mfrow = c(1, 1))
```

Fine and Gray regression on cumulative incidence functions is not implemented in *mstate*, but in the `R` package *cmprsk*. Since our main purpose here is illustration of *mstate*, we just give the code and the output.

```
> library(cmprsk)
> sic <- si[!is.na(si$ccr5),]
> ftime <- sic$time
> fstatus <- sic$status
> cov <- as.numeric(sic$ccr5)-1
> # for failures of type 1 (AIDS)
> z1 <- crr(ftime,fstatus,cov)
> z1
```

```
convergence:  TRUE
coefficients:
  cov1
-1.004
standard errors:
[1] 0.295
two-sided p-values:
   cov1
0.00066

> # for failures of type 2 (SI)
> z2 <- crr(ftime,fstatus,cov,failcode=2)
> z2

convergence:  TRUE
coefficients:
   cov1
0.02359
standard errors:
[1] 0.2266
two-sided p-values:
cov1
0.92
```

The result (Figure 8 in the tutorial) is shown in Figure 11.

```
> z1.pr <- predict(z1,matrix(c(0,1),2,1))
> # this will contain predicted cum inc curves, both for WW (2nd column) and WM (3rd)
> z2.pr <- predict(z2,matrix(c(0,1),2,1))
> # Standard plots, not shown
> par(mfrow=c(1,2))
> plot(z1.pr,lty=1,lwd=2,color=c(8,1))
> plot(z2.pr,lty=1,lwd=2,color=c(8,1))
> par(mfrow=c(1,1))
> ## AIDS
> n1 <- nrow(z1.pr) # remove last jump
> plot(c(0,z1.pr[-n1,1]),c(0,z1.pr[-n1,2]),type="s",ylim=c(0,0.5),
+     xlab="Years from HIV infection",ylab="Probability",lwd=2)
> lines(c(0,z1.pr[-n1,1]),c(0,z1.pr[-n1,3]),type="s",lwd=2,col=8)
> title(main="AIDS")
> text(9.3,0.35,"WW",adj=0,cex=0.75)
> text(9.3,0.14,"WM",adj=0,cex=0.75)
> ## SI appearance
> n2 <- nrow(z2.pr) # again remove last jump
> plot(c(0,z2.pr[-n2,1]),c(0,z2.pr[-n2,2]),type="s",ylim=c(0,0.5),
+     xlab="Years from HIV infection",ylab="Probability",lwd=2)
> lines(c(0,z2.pr[-n2,1]),c(0,z2.pr[-n2,3]),type="s",lwd=2,col=8)
> title(main="SI appearance")
> text(7.9,0.28,"WW",adj=0,cex=0.75)
> text(7.9,0.31,"WM",adj=0,cex=0.75)
```

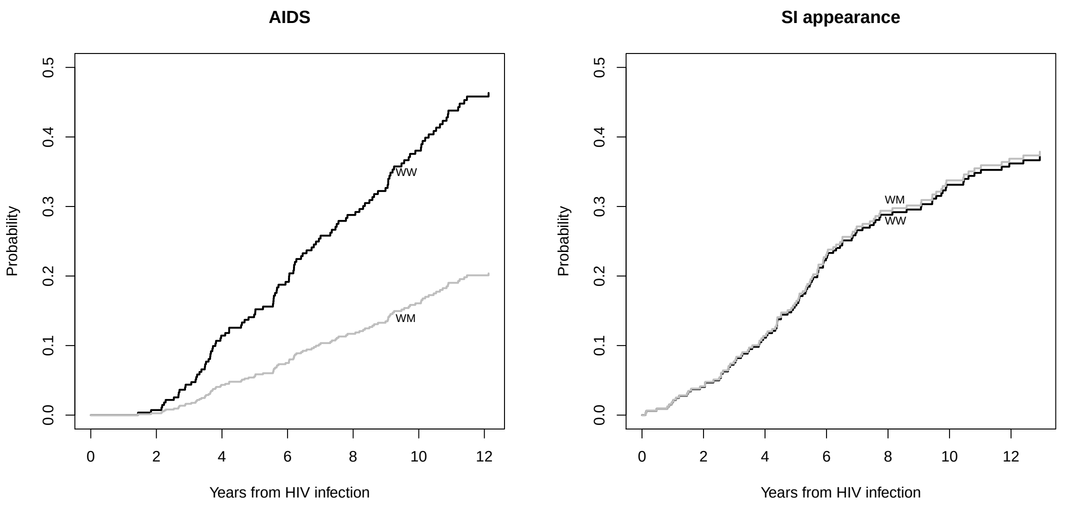

Figure 11: Cumulative incidence functions for AIDS (left) and SI appearance (right), for CCR5 wild-type `WW` and mutant `WM`, based on the Fine and Gray model.

To judge the "fit" of the cause-specific and Fine & Gray regression models we estimate cumulative incidence curves nonparametrically, i.e., for two subgroups of WW and WM CCR5-genotypes. Here we can use the *group* argument of `Cuminc`.

```
> ci <- Cuminc(si$time, si$status, group = si$ccr5)
> ci.WW <- ci[ci$group == "WW", ]
> ci.WM <- ci[ci$group == "WM", ]
```

We show these nonparametric estimates in Figure 12 (Figure 9 in the tutorial).

```
> idx1 <- (ci.WW$time < 13)
> idx2 <- (ci.WM$time < 13)
> plot(c(0, ci.WW$time[idx1]), c(0, ci.WW$CI.1[idx1]), type = "s",
+     ylim = c(0, 0.5), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> lines(c(0, ci.WM$time[idx2]), c(0, ci.WM$CI.1[idx2]), type = "s",
+     lwd = 2, col = 8)
> title(main = "AIDS")
> text(9.3, 0.35, "WW", adj = 0, cex = 0.75)
> text(9.3, 0.11, "WM", adj = 0, cex = 0.75)
> plot(c(0, ci.WW$time[idx1]), c(0, ci.WW$CI.2[idx1]), type = "s",
+     ylim = c(0, 0.5), xlab = "Years from HIV infection", ylab = "Probability",
+     lwd = 2)
> lines(c(0, ci.WM$time[idx2]), c(0, ci.WM$CI.2[idx2]), type = "s",
+     lwd = 2, col = 8)
> title(main = "SI appearance")
> text(7.9, 0.32, "WW", adj = 0, cex = 0.75)
> text(7.9, 0.245, "WM", adj = 0, cex = 0.75)
```

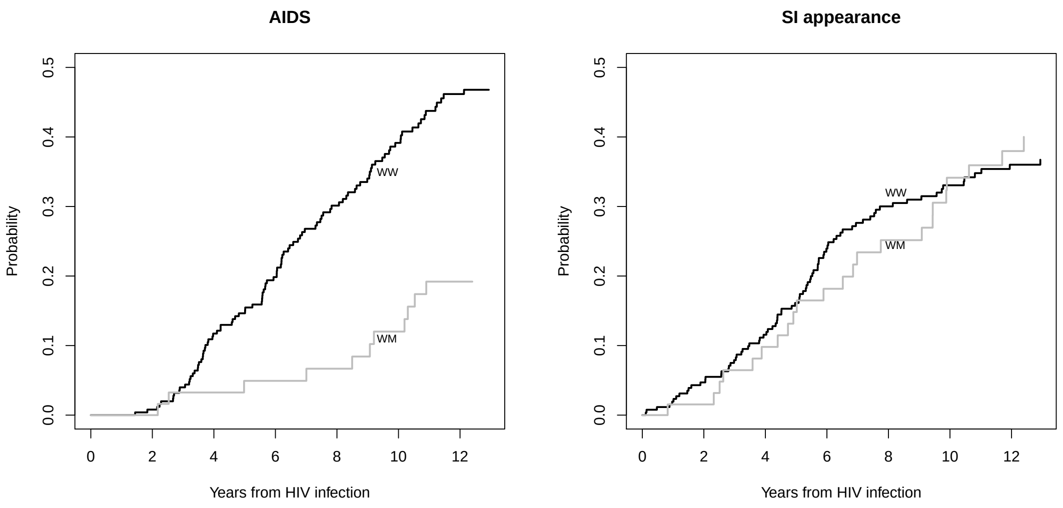

Figure 12: Non-parametric cumulative incidence functions for AIDS (left) and SI appearance (right), for CCR5 wild-type `WW` and mutant `WM`.

References

Andersen, P. K., Borgan, Ø., Gill, R. D. & Keiding, N. (1993), *Statistical Models Based on Counting Processes*, Springer-Verlag.

de Wreede, L., Fiocco, M. & Putter, H. (2009), 'The mstate package for estimation and prediction in non- and semi-parametric multi-state models'. Submitted.

Putter, H., Fiocco, M. & Geskus, R. B. (2007), 'Tutorial in biostatistics: Competing risks and multi-state models', *Statist Med* **26**, 2389–2430.