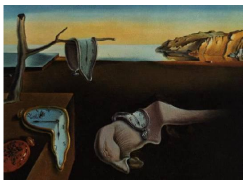

Journal of Statistical Software

January 2011, Volume 38, Issue 7. *http://www.jstatsoft.org/*

mstate: An R Package for the Analysis of Competing Risks and Multi-State Models

Liesbeth C. de Wreede
Leiden University
Medical Center

Marta Fiocco
Leiden University
Medical Center

Hein Putter
Leiden University
Medical Center

Abstract

Multi-state models are a very useful tool to answer a wide range of questions in survival analysis that cannot, or only in a more complicated way, be answered by classical models. They are suitable for both biomedical and other applications in which time-to-event variables are analyzed. However, they are still not frequently applied. So far, an important reason for this has been the lack of available software. To overcome this problem, we have developed the **mstate** package in R for the analysis of multi-state models. The package covers all steps of the analysis of multi-state models, from model building and data preparation to estimation and graphical representation of the results. It can be applied to non- and semi-parametric (Cox) models. The package is also suitable for competing risks models, as they are a special category of multi-state models.

This article offers guidelines for the actual use of the software by means of an elaborate multi-state analysis of data describing post-transplant events of patients with blood cancer. The data have been provided by the EBMT (the European Group for Blood and Marrow Transplantation). Special attention will be paid to the modeling of different covariate effects (the same for all transitions or transition-specific) and different baseline hazard assumptions (different for all transitions or equal for some).

Keywords: competing risks, estimation, multi-state models, prediction, R, survival analysis.

1. Introduction

Recently, multi-state and competing risks models have gained considerable popularity in survival analysis. In the first place, this popularity is due to the fact that in comparison to classical models, these models describe the disease/recovery process of patients in more detail, thus yielding more insight. In the second place, these models are useful in a clinical setting for the prediction of survival duration for specific patients. These predictions can easily be updated if information about events later occurring to such a patient becomes available. The influence of covariates can be taken into account.

Although the relevance of these models is clearly recognized, their application by non-statisticians has so far been limited. An important reason for this is the lack of available software. For this reason, we have developed a software package in R (R Development Core Team 2010), called **mstate**, which can be used for the different phases of the description and analysis of competing risks and multi-state models. It is primarily designed for non- and semi-parametric Cox models. However, we have paid special attention to the flexibility of the functions: they can be used independently of each other to enable users to combine their own functions with those of **mstate** when considering models different from ours. **mstate** is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=mstate`.

The mathematics underlying the package, its philosophy and features and an example have been discussed in our previous paper de Wreede *et al.* (2010). A more general introduction into competing risks and multi-state models can be found in Putter *et al.* (2007). The current article builds on these previous articles. It describes in detail how the functions work by means of a more elaborate example. The mathematical concepts related to the functions will only be mentioned briefly in the relevant places. The use of the package is illustrated by the analysis of a model describing the disease/recovery process of leukemia patients, of which model several variants will be discussed. The code for all of them will be given, and it will be explained how this should be adapted for other models. Data, clinical questions and model are introduced in Section 2. We consider estimation and prediction both for a non-parametric model (Section 3) and a semi-parametric model including several relevant clinical covariates (Section 4.1). The use of transition-specific covariates will be explained. As a special semi-parametric model, we consider a proportional baseline hazards model, which is useful for reducing the number of parameters in the model (Section 4.2). In Section 4.3 we discuss reduced rank models and two functions for simulation and bootstrapping in the context of multi-state models. Finally, the analysis of competing risks models by means of **mstate** is discussed briefly in Section 4.4.

2. Data, questions, and model

We consider survival after a transplant treatment of patients suffering from a blood cancer. The data have been provided by the EBMT (the European Group for Blood and Marrow Transplantation); they are available in **mstate** as `ebmt4`. The present data set has been compiled to illustrate the models and the software. To facilitate this illustration, only patients with complete covariate information and a reasonable amount of information about intermediate events have been included. Although the current data set mimics a real-life situation and although the order of magnitude of the outcomes is correct, the clinical meaning of the results of the analyses is restricted. To avoid misinterpretation, we have abstracted from the actual disease, covariate values and intermediate events.

Three intermediate events are included in the model: Recovery (Rec), an Adverse Event (AE) and a combination of the two (AE and Rec). It is to be expected that recovery improves the prognosis and an adverse event deteriorates it. The model is suitable to show the size of these effects, and to capture the influence of their timing and of the covariates on the prognosis.

Prognostic factor (name in data)	Categories	n (%)
Donor recipient	no gender mismatch	1734 (76)
(`match`)	gender mismatch	545 (24)
Prophylaxis	no	1730 (76)
(`proph`)	yes	549 (24)
Year of transplant	1985-1989	634 (28)
(`year`)	1990-1994	896 (39)
	1995-1998	749 (33)
Age at transplant (years)	≤ 20	551 (24)
(`agecl`)	$20-40$	1213 (53)
	> 40	515 (23)

Table 1: Prognostic factors for all patients

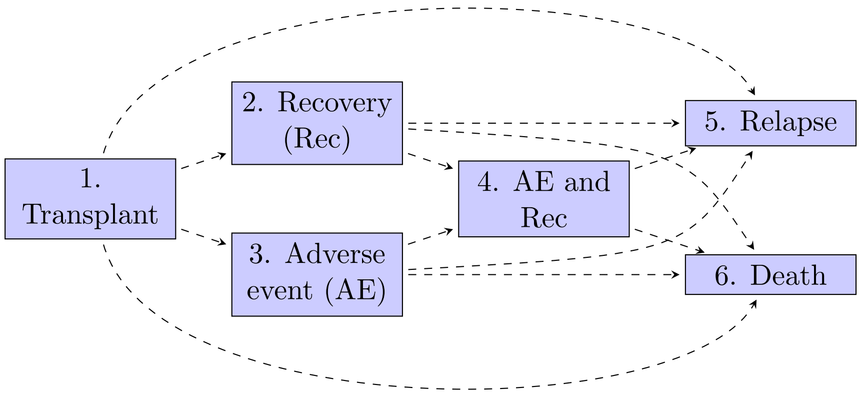

Figure 1: A six-states model for leukemia patients after bone marrow transplantation

Moreover, it shows what happens when both the positive and negative event take place, as compared to one or none of them. The models under study here can easily be made more specific in the case of real applications. For instance, instead of Recovery, Engraftment can be included, and instead of Adverse Event, Acute Graft-versus-Host Disease.

We consider 2279 patients who were treated between 1985 and 1998. Four prognostic factors are known at baseline for all patients (see Table 1). They are: donor-recipient match (where gender mismatch is defined as female donor, male recipient), prophylaxis, year of transplant and age at transplant in years. All these covariates are treated as time-fixed categorical covariates. The distribution of the values of the covariates over the patients in the data set is shown in Table 1.

A multi-state approach is particularly appropriate for these data, since it can help to model both the disease-related and the treatment-related morbidity and mortality. These are modeled here by including the intermediate events recovery and adverse event. Information about the occurrence of these events is used to update the prognosis of the patients.

We consider the following six-states model (see Figure 1):

1. Alive and in remission, no recovery or adverse event;
2. Alive in remission, recovered from the treatment;
3. Alive in remission, occurrence of the adverse event;
4. Alive, both recovered and adverse event occurred;
5. Alive, in relapse (treatment failure);
6. Dead (treatment failure).

All patients start in state 1. States 5 and 6 are called absorbing: once the patient has entered one of them, she/he stays there. This leaves us with a model with 12 transitions.

The data have been made suitable for a multi-state analysis by some small adjustments. Since the model does not allow patients to enter two states at the same time, we have set the death indicator of patients with simultaneous relapse and death to 0, because although these events were reported at the same time, in reality the patients must have experienced the relapse before their death. For those with equal time to the adverse event and time to Rec we have lowered the time to AE by half a day to avoid a transition with only very few events. Two new variables have been created to express the time of entry in state 4 (AE and Rec) and the accompanying status indicator: `recae` and `recae.s` respectively.

The data are available in **mstate** in wide format. This means that each row in the data corresponds to a single subject.

```
R> library("mstate")
R> data("ebmt4")
R> ebmt <- ebmt4
R> head(ebmt)
```

```
  id  rec rec.s   ae ae.s recae recae.s  rel rel.s  srv srv.s
1  1   22     1  995    0   995       0  995     0  995     0
2  2   29     1   12    1    29       1  422     1  579     1
3  3 1264     0   27    1  1264       0 1264     0 1264     0
4  4   50     1   42    1    50       1   84     1  117     1
5  5   22     1 1133    0  1133       0  114     1 1133     0
6  6   33     1   27    1    33       1 1427     0 1427     0
       year agecl proph             match
1 1995-1998 20-40    no no gender mismatch
2 1995-1998 20-40    no no gender mismatch
3 1995-1998 20-40    no no gender mismatch
4 1995-1998 20-40    no    gender mismatch
5 1995-1998   >40    no    gender mismatch
6 1995-1998 20-40    no no gender mismatch
```

The columns `rec`, `ae`, `rel` and `srv` are time variables, indicating the time measured in days post-transplant to recovery, AE, relapse and death respectively in case of an event, or last follow-up otherwise. The `.s`-variables are the corresponding status variables (1 for an event, 0 for censoring). For instance, patient 1 had recovered after 22 days (transition from state 1 to state 2) and was censored after 995 days without a further event. Patient 2 experienced the adverse event after 12 days (transition from state 1 to state 3), then recovery after 29 days (transition from state 3 to state 4) and a relapse after 422 days (transition from state 4 to state 5). Finally, he/she died after 579 days, but this last event is not relevant to the model, because the patient had already reached an absorbing state.

Diverse clinical questions can be answered by fitting this model to the data, such as: how does age influence the different phases of the disease/recovery process? How does the occurrence of the adverse event after 2 months change the prognosis of 10-year survival for a patient? Which risks should be monitored most carefully for a patient who has recovered after one month, taking into account certain covariate values?

The model can be described by means of a transition matrix, which is in this case a 6-by-6 matrix. A number at entry (g, h) of the matrix represents a possible transition from state g to state h. These numbers range here from 1 to 12, because the model has 12 transitions. These transition numbers are used in the various stages of the analysis. If a transition between two states is not allowed, the entry becomes `NA`. The function `transMat()` creates the transition matrix `tmat`. It has been contributed to **mstate** by Steven McKinney.

```
R> tmat <- transMat(x = list(c(2, 3, 5, 6), c(4, 5, 6), c(4, 5, 6), c(5, 6),
+    c(), c()), names = c("Tx", "Rec", "AE", "Rec+AE", "Rel", "Death"))
R> tmat
```

```
        to
from     Tx Rec AE Rec+AE Rel Death
  Tx     NA   1  2     NA   3     4
  Rec    NA  NA NA      5   6     7
  AE     NA  NA NA      8   9    10
  Rec+AE NA  NA NA     NA  11    12
  Rel    NA  NA NA     NA  NA    NA
  Death  NA  NA NA     NA  NA    NA
```

All possible paths through the multi-state model can be found by the function `paths()`.

In the present format, the data are not yet suitable for a multi-state analysis. First they have to be recoded into 'long format'. In this format, each subject has as many rows as transitions for which he/she is at risk. The function `msprep()` transforms a data frame in wide format into one in long format. Arguments for `msprep()` are a `time` and `status` vector indicating the time of entry in every state and the accompanying status indicator. The `keep` argument contains the names of the covariates that will be used in the analysis. The output is an object of class '`msdata`': a data frame in long format, which has the transition matrix as an attribute. This object has its own `print()` method.

The call to `msprep()` and output for the first patient are as follows (covariate `match` is not shown in the output):

```
R> msebmt <- msprep(data = ebmt, trans = tmat, time = c(NA, "rec", "ae",
+    "recae", "rel", "srv"), status = c(NA, "rec.s", "ae.s", "recae.s",
+    "rel.s", "srv.s"), keep = c("match", "proph", "year", "agecl"))
R> msebmt[msebmt$id == 1, c(1:8, 10:12)]
```

```
An object of class 'msdata'

Data:
  id from to trans Tstart Tstop time status proph      year agecl
1  1    1  2     1      0    22   22      1    no 1995-1998 20-40
2  1    1  3     2      0    22   22      0    no 1995-1998 20-40
3  1    1  5     3      0    22   22      0    no 1995-1998 20-40
4  1    1  6     4      0    22   22      0    no 1995-1998 20-40
5  1    2  4     5     22   995  973      0    no 1995-1998 20-40
6  1    2  5     6     22   995  973      0    no 1995-1998 20-40
7  1    2  6     7     22   995  973      0    no 1995-1998 20-40
```

Consider again the first subject. Starting from state 1, he/she is at risk for transitions $1, \ldots, 4$. This means that she/he can move to states 2, 3, 5 and 6. At time 22, the patient moves to state 2 (Recovery), from where he/she is at risk for a further transition to state 4, 5 and 6 (i.e., transitions 5, 6 and 7). None of these occur and the patient is censored at time 995. The patient has no rows for transitions 8–12 because he/she has never been at risk for these. The value of `time` is equal to `Tstop`−`Tstart`; it is of use in 'clock reset'-models, where the time t refers to the time spent in the current state.

The numbers of transitions, both in terms of frequencies and percentages, are given by the function `events()`.

```
R> events(msebmt)

$Frequencies
        to
from       Tx  Rec   AE Rec+AE  Rel Death no event total entering
  Tx        0  785  907      0   95   160      332           2279
  Rec       0    0    0    227  112    39      407            785
  AE        0    0    0    433   56   197      221            907
  Rec+AE    0    0    0      0  107   137      416            660
  Rel       0    0    0      0    0     0        0              0
  Death     0    0    0      0    0     0        0              0

$Proportions
        to
from             Tx        Rec         AE     Rec+AE        Rel
  Tx     0.00000000 0.34444932 0.39798157 0.00000000 0.04168495
  Rec    0.00000000 0.00000000 0.00000000 0.28917197 0.14267516
  AE     0.00000000 0.00000000 0.00000000 0.47739802 0.06174201
  Rec+AE 0.00000000 0.00000000 0.00000000 0.00000000 0.16212121
  Rel
  Death
        to
from          Death   no event
  Tx     0.07020623 0.14567793
  Rec    0.04968153 0.51847134
  AE     0.21719956 0.24366042
  Rec+AE 0.20757576 0.63030303
  Rel
  Death
```

In Section 4.1, a model will be introduced in which it is assumed that the covariates have different effects on each transition. This can be achieved by creating transition-specific covariates. They are derived from the covariates at baseline as follows: each covariate Z is split up into as many covariates Z_{gh} as there are transitions in the model. For the transition from state g to state h, Z_{gh} is equal to Z; for all other transitions, $Z_{gh} = 0$.

The function `expand.covs()` expands the covariates specified by the user on the basis of an object of class '`msdata`'. Factors are expanded into dummy variables. The option `longnames` controls whether the names of the dummy variables are based on the levels of the categorical covariates (`TRUE`) or on numbers (`FALSE`). In our example, we obtain six new covariates for each transition: one dummy variable each for donor-recipient match and prophylaxis, and two each for year of transplant and age. For the coding of the dummy variables, consider `agecl`, which has three values. These are coded by `agecl1` and `agecl2`. For the reference category (1985–1989) `agecl1` and `agecl2` are both 0; for 1990–1994, `agecl1` $= 1$ and `agecl2` $= 0$ and for 1995–1998, `agecl1` $= 0$ and `agecl2` $= 1$. This results in 72 new covariates altogether.

We illustrate this process by showing a selection of the long format data for the first patient, leaving out the values at baseline and the transition-specific counterparts of `match`, `proph` and `agecl`. We also omit `year1.1` to `year1.12`, because they are all 0.

```
R> covs <- c("match", "proph", "year", "agecl")
R> msebmt <- expand.covs(msebmt, covs, longnames = FALSE)
R> msebmt[msebmt$id == 1, -c(9, 10, 12:48, 61:84)]

An object of class 'msdata'

Data:
  id from to trans Tstart Tstop time status      year year2.1 year2.2
1  1    1  2     1      0    22   22      1 1995-1998       1       0
2  1    1  3     2      0    22   22      0 1995-1998       0       1
3  1    1  5     3      0    22   22      0 1995-1998       0       0
4  1    1  6     4      0    22   22      0 1995-1998       0       0
5  1    2  4     5     22   995  973      0 1995-1998       0       0
6  1    2  5     6     22   995  973      0 1995-1998       0       0
7  1    2  6     7     22   995  973      0 1995-1998       0       0
  year2.3 year2.4 year2.5 year2.6 year2.7 year2.8 year2.9 year2.10
1       0       0       0       0       0       0       0        0
2       0       0       0       0       0       0       0        0
3       1       0       0       0       0       0       0        0
4       0       1       0       0       0       0       0        0
5       0       0       1       0       0       0       0        0
6       0       0       0       1       0       0       0        0
7       0       0       0       0       1       0       0        0
```

```
  year2.11 year2.12
1        0        0
2        0        0
3        0        0
4        0        0
5        0        0
6        0        0
7        0        0
```

The original covariates are maintained in the output. This makes it possible to consider models with any mixture of basic and transition-specific covariates.

Finally, for future modeling we convert time into years.

```
R> msebmt[, c("Tstart", "Tstop", "time")] <- msebmt[, c("Tstart",
+    "Tstop", "time")]/365.25
```

All necessary preparations for estimation have now been made.

3. A non-parametric model

First we consider a non-parametric model, in which we ignore the influence of covariates. The basic quantities of interest are the transition intensities or hazard rates. Denote by $g \to h$ a transition from state g to state h, by $X(t)$ the state occupied at time t and by $\alpha_{gh}(t)$ the corresponding transition intensity. The transition intensity expresses the instantaneous risk of a transition from state g into state h at time t. It is defined as

$$\alpha_{gh}(t) := \lim_{\Delta t \to 0} \frac{\mathsf{P}(X(t+\Delta t) = h \,|\, X(t) = g)}{\Delta t} \; . \tag{1}$$

This definition makes an implicit assumption that the multi-state model is Markovian, which implies that the probability of going to a future state depends only on the present state and not on the history. The cumulative transition hazard for transition $g \to h$ is defined as $A_{gh}(t) = \int_0^t \alpha_{gh}(u) \mathrm{d}u$. Let $\mathbf{A}(t)$ be a matrix with dimensions $S \times S$, in which S denotes the number of states. This matrix has elements $A_{gh}(t)$ $(g \neq h)$, and diagonal elements $A_{gg}(t) = -\sum_{h \neq g} A_{gh}(t)$.

For the estimation of the cumulative hazard, we assume we have data with independent censoring. The Nelson-Aalen estimator $\widehat{A}_{gh}(t)$ of the cumulative hazard for transition $g \to h$ is now given by

$$\widehat{A}_{gh}(t) = \sum_{t_i \leq t} \mathrm{d}N_{gh}(t_i)/Y_g(t_i) \; , \tag{2}$$

where t_i indicates the event times, $\mathrm{d}N_{gh}(t_i)$ is the observed number of transitions from state g to state h at time t_i, and $Y_g(t_i)$ is the number of subjects at risk for a transition from state g at t_i. The Nelson-Aalen estimator makes jumps of size $\Delta\widehat{A}_{gh}(t_i) = \mathrm{d}N_{gh}(t_i)/Y_g(t_i)$ at the event times t_i. The estimates of the diagonal elements of $\mathbf{A}(t)$ are given by $\widehat{A}_{gg}(t) = -\sum_{h \neq g} \widehat{A}_{gh}(t)$.

We use the function `coxph()` from the **survival** package (Therneau and Lumley 2010) to estimate the cumulative hazards.

```
R> c0 <- coxph(Surv(Tstart, Tstop, status) ~ strata(trans), data = msebmt,
+    method = "breslow")
```

This Cox model has separate baseline hazards for each of the transitions and no covariates. In principle, the transition intensities could also be estimated separately, but the combined use of long format data and a single stratified `copxh` object makes further calculations easier.

The output of `coxph()` is the input for **mstate**'s function `msfit()`. It estimates transition hazards and their associated (co)variances. The output of `msfit()` is an object of class '`msfit`'. This is a list with elements `Haz` (with the estimated cumulative hazard values at all event times), `varHaz` (with the covariances of each pair of estimated cumulative hazards at each event time point, i.e., $\widehat{\mathrm{cov}}(\widehat{A}_{gh}(t), \widehat{A}_{kl}(t))$), and `trans`, in which the transition matrix is stored for further use. The (co)variances of the estimated cumulative hazards may be computed in two different ways: by means of the Aalen estimator or by means of the Greenwood estimator. An advantage of the Greenwood estimator is the fact that it yields exact multinomial standard errors for the transition probabilities when there is no censoring. The two estimators give almost equal results in all practical applications. For details see Section IV.4.1.3 of Andersen *et al.* (1993) and de Wreede *et al.* (2010).

```
R> msf0 <- msfit(object = c0, vartype = "greenwood", trans = tmat)
```

The '`msfit`' class has its own `summary()` and `plot()` methods. By default, `summary()` prints head and tail of the cumulative hazards of all transitions. Since there are twelve transitions in the model, its output is not shown here in the interest of space. The head and tail of the `Haz` and `varHaz` items look as follows (`time` is now given in years):

```
R> head(msf0$Haz)

         time         Haz trans
1 0.002737851 0.000000000     1
2 0.008213552 0.000000000     1
3 0.010951403 0.000000000     1
4 0.013689254 0.000000000     1
5 0.016427105 0.000443066     1
6 0.019164956 0.001333142     1

R> tail(msf0$Haz)

         time       Haz trans
6199 12.48460 0.3800455    12
6200 12.61602 0.3800455    12
6201 13.02396 0.3800455    12
6202 13.10609 0.3800455    12
6203 13.12799 0.4255001    12
6204 17.24572 0.4255001    12

R> head(msf0$varHaz)
```

```
         time       varHaz trans1 trans2
1 0.002737851 0.000000e+00      1      1
2 0.008213552 0.000000e+00      1      1
3 0.010951403 0.000000e+00      1      1
4 0.013689254 0.000000e+00      1      1
5 0.016427105 1.962205e-07      1      1
6 0.019164956 5.919853e-07      1      1
```

```
R> tail(msf0$varHaz)
```

```
          time      varHaz trans1 trans2
40321 12.48460 0.002461034     12     12
40322 12.61602 0.002461034     12     12
40323 13.02396 0.002461034     12     12
40324 13.10609 0.002461034     12     12
40325 13.12799 0.004433236     12     12
40326 17.24572 0.004433236     12     12
```

The last time point in the list indicates the last time point in the data, either of an event or of a censoring. The `plot()` function produces a graph of all estimated cumulative hazards in different colors:

```
R> plot(msf0, las = 1, lty = rep(1:2, c(8, 4)),
+    xlab = "Years since transplantation")
```

When we choose the `"aalen"` option for the `vartype` argument in `msfit()`, the estimated cumulative hazards are the same as before, but the standard errors are slightly different.

```
R> msf0a <- msfit(object = c0, vartype = "aalen", trans = tmat)
R> head(msf0a$varHaz)
```

```
         time       varHaz trans1 trans2
1 0.002737851 0.000000e+00      1      1
2 0.008213552 0.000000e+00      1      1
3 0.010951403 0.000000e+00      1      1
4 0.013689254 0.000000e+00      1      1
5 0.016427105 1.963075e-07      1      1
6 0.019164956 5.924248e-07      1      1
```

```
R> tail(msf0a$varHaz)
```

```
          time      varHaz trans1 trans2
40321 12.48460 0.002502928     12     12
40322 12.61602 0.002502928     12     12
40323 13.02396 0.002502928     12     12
40324 13.10609 0.002502928     12     12
40325 13.12799 0.004569044     12     12
40326 17.24572 0.004569044     12     12
```

We proceed to the estimation of the transition probability matrix for which the necessary elements are now available. Denote the transition probability matrix by $\mathbf{P}(s,t)$. It has elements

$$P_{gh}(s,t) = P(X(t) = h|X(s) = g), \tag{3}$$

denoting the transition probability from state g tot state h in time interval $(s,t]$. The transition probability matrix is estimated as

$$\widehat{\mathbf{P}}(s,t) = \Pi_{u\in(s,t]}(\mathbf{I} + \Delta\widehat{\mathbf{A}}(u)), \tag{4}$$

where u indicates the event times and the elements of $\mathbf{A}$ are estimated as in (2). Formula (4) is called the Aalen-Johansen estimator.

The function `probtrans()` calculates the estimated transition probabilities, and optionally the standard errors and/or the covariances of the transition probabilities. In the case of non-parametric models, the user can choose between Greenwood or Aalen standard errors in the `method` argument, in accordance with the choice in `msfit()`. Just as in the case of the estimates of the hazards, the estimates of the transition probabilities themselves do not depend on this choice.

The argument `predt` gives the starting time for prediction, that is, the starting time for the calculation of the transition probabilities. Two directions of prediction have been implemented, which can be specified by the `direction` argument: `"forward"` (the default) and `"fixedhorizon"`. Any string starting with `"fo"` or `"fi"` is sufficient to distinguish between the two options. The `"forward"` option means that the prediction is made from `predt`; in $\mathbf{P}(s,t)$, time s remains fixed at the value `predt`, while time t varies from s to the last (possibly censored) time point in the data. The `"fixedhorizon"` option means that the prediction is made for `predt`; in $\mathbf{P}(s,t)$, time t remains fixed at the value `predt` and time s varies from 0 to `predt`. The use of the fixed horizon option will be illustrated in Section 4.1.

The default output of `probtrans()` is an object of class '`probtrans`', which is a list of $(S+1)$ elements, where S is the number of states. Element g of the list contains all transition probabilities starting from state g and going to all states, and their associated standard errors ($g = 1,\ldots,S$). The last element of the list contains again the transition matrix. If the option `covariance=TRUE` is chosen, an additional element is added to the list containing all estimated covariances of the transition probabilities. The '`probtrans`' class has two methods, `summary()` and `plot()`. By default the `summary()` method prints the head and tail of the estimated transition probabilities for each of the states g. If the transition probabilities from a particular starting state are required, the argument `from` must be added. These results show how the prognosis of a patient depends on his/her starting state and on the moment that is taken as the starting point for prediction.

The code for `probtrans()` and a sample of its output look as follows:

```
R> pt0 <- probtrans(msf0, predt = 0, method = "greenwood")
R> pt0a <- probtrans(msf0a, predt = 0, method = "aalen")
R> summary(pt0, from = 1)

An object of class 'probtrans'

Prediction from state 1 (head and tail):
```

```
         time   pstate1      pstate2      pstate3 pstate4 pstate5
1 0.000000000 1.0000000 0.0000000000 0.0000000000       0       0
2 0.002737851 0.9995610 0.0000000000 0.0004389816       0       0
3 0.008213552 0.9978051 0.0000000000 0.0021949078       0       0
4 0.010951403 0.9956102 0.0000000000 0.0039508341       0       0
5 0.013689254 0.9907814 0.0000000000 0.0079016681       0       0
6 0.016427105 0.9863916 0.0004389816 0.0118525022       0       0
       pstate6          se1          se2          se3 se4 se5
1 0.0000000000 0.0000000000 0.0000000000 0.0000000000   0   0
2 0.0000000000 0.0004388852 0.0000000000 0.0004388852   0   0
3 0.0000000000 0.0009805148 0.0000000000 0.0009805148   0   0
4 0.0004389816 0.0013851313 0.0000000000 0.0013143406   0   0
5 0.0013169447 0.0020023724 0.0000000000 0.0018550683   0   0
6 0.0013169447 0.0024274584 0.0004388852 0.0022674569   0   0
           se6
1 0.0000000000
2 0.0000000000
3 0.0000000000
4 0.0004388852
5 0.0007598375
6 0.0007598375

...

        time   pstate1   pstate2    pstate3   pstate4   pstate5
513 12.48460 0.1366292 0.1650151 0.09331023 0.1593697 0.1811612
514 12.61602 0.1320749 0.1650151 0.09331023 0.1593697 0.1811612
515 13.02396 0.1320749 0.1650151 0.08997772 0.1593697 0.1811612
516 13.10609 0.1265718 0.1650151 0.08997772 0.1593697 0.1811612
517 13.12799 0.1265718 0.1650151 0.08997772 0.1521256 0.1811612
518 17.24572 0.1265718 0.1650151 0.08997772 0.1521256 0.1811612
      pstate6         se1         se2         se3        se4
513 0.2645145 0.008491577 0.009230259 0.006334163 0.01062790
514 0.2690688 0.009350412 0.009230259 0.006334163 0.01062790
515 0.2724013 0.009350412 0.009230259 0.006929354 0.01062790
516 0.2779044 0.010455556 0.009230259 0.006929354 0.01062790
517 0.2851485 0.010455556 0.009230259 0.006929354 0.01236966
518 0.2851485 0.010455556 0.009230259 0.006929354 0.01236966
           se5        se6
513 0.01001397 0.01245932
514 0.01001397 0.01315637
515 0.01001397 0.01352405
516 0.01001397 0.01441637
517 0.01001397 0.01588915
518 0.01001397 0.01588915
```

```
R> summary(pt0a, from = 1)

An object of class 'probtrans'

Prediction from state 1 (head and tail):

         time   pstate1      pstate2      pstate3 pstate4 pstate5
1 0.000000000 1.0000000 0.0000000000 0.0000000000       0       0
2 0.002737851 0.9995610 0.0000000000 0.0004389816       0       0
3 0.008213552 0.9978051 0.0000000000 0.0021949078       0       0
4 0.010951403 0.9956102 0.0000000000 0.0039508341       0       0
5 0.013689254 0.9907814 0.0000000000 0.0079016681       0       0
6 0.016427105 0.9863916 0.0004389816 0.0118525022       0       0
       pstate6          se1          se2          se3 se4 se5
1 0.0000000000 0.0000000000 0.0000000000 0.0000000000   0   0
2 0.0000000000 0.0004387889 0.0000000000 0.0004387889   0   0
3 0.0000000000 0.0009797822 0.0000000000 0.0009797822   0   0
4 0.0004389816 0.0013838512 0.0000000000 0.0013130230   0   0
5 0.0013169447 0.0019989382 0.0000000000 0.0018514699   0   0
6 0.0013169447 0.0024228979 0.0004370375 0.0022626745   0   0
           se6
1 0.0000000000
2 0.0000000000
3 0.0000000000
4 0.0004380161
5 0.0007571011
6 0.0007571011

...

        time   pstate1   pstate2    pstate3   pstate4   pstate5
513 12.48460 0.1366292 0.1650151 0.09331023 0.1593697 0.1811612
514 12.61602 0.1320749 0.1650151 0.09331023 0.1593697 0.1811612
515 13.02396 0.1320749 0.1650151 0.08997772 0.1593697 0.1811612
516 13.10609 0.1265718 0.1650151 0.08997772 0.1593697 0.1811612
517 13.12799 0.1265718 0.1650151 0.08997772 0.1521256 0.1811612
518 17.24572 0.1265718 0.1650151 0.08997772 0.1521256 0.1811612
      pstate6         se1         se2         se3        se4
513 0.2645145 0.008402082 0.009190605 0.006308515 0.01059278
514 0.2690688 0.009238456 0.009190605 0.006308515 0.01059278
515 0.2724013 0.009238456 0.009190605 0.006879823 0.01059278
516 0.2779044 0.010305245 0.009190605 0.006879823 0.01059278
517 0.2851485 0.010305245 0.009190605 0.006879823 0.01224960
518 0.2851485 0.010305245 0.009190605 0.006879823 0.01224960
            se5        se6
513 0.009959111 0.01236101
514 0.009959111 0.01303965
```

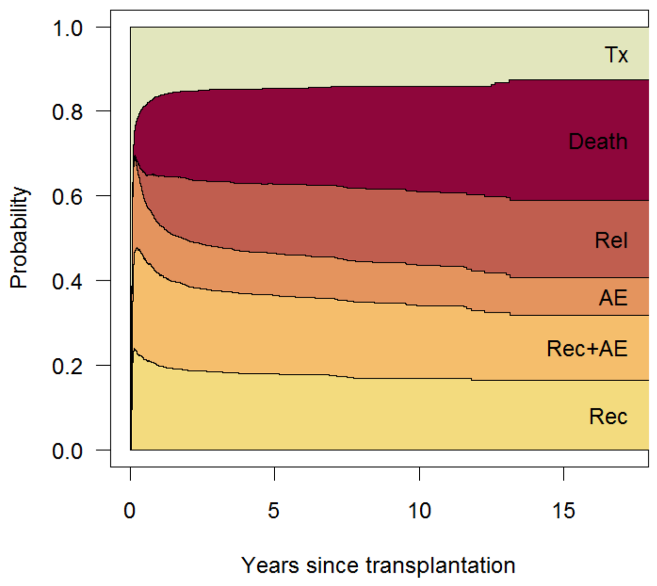

Figure 2: Stacked transition probabilities

```
515 0.009959111 0.01339608
516 0.009959111 0.01425796
517 0.009959111 0.01567654
518 0.009959111 0.01567654
```

The output shows that the `"greenwood"` and `"aalen"` options for `method` give identical estimated transition probabilities but slightly different standard errors.

The `plot()` method enables the user to show the transition probabilities in several ways. The output of `probtrans()` is perhaps most conveniently interpretable when plotted in a figure with stacked transition probabilities. The argument `ord` is used to specify an informative ordering of the transition probabilities to be stacked. The numbers in `ord` correspond to the states specified in the transition matrix. The argument `type` specifies the type of plot; we ask here for the space between adjacent curves to be filled with suitable colors (darker is more serious). For this we used the **colorspace** package (Ihaka *et al.* 2009; Zeileis *et al.* 2009).

```
R> library("colorspace")
R> statecols <- heat_hcl(6, c = c(80, 30), l = c(30, 90),
+    power = c(1/5, 2))[c(6, 5, 3, 4, 2, 1)]
R> ord <- c(2, 4, 3, 5, 6, 1)
R> plot(pt0, ord = ord, xlab = "Years since transplantation",
+    las = 1, type = "filled", col = statecols[ord])
```

Figure 2 shows the result. The distance between two adjacent curves represents the probability of being in the corresponding state. The particular order chosen makes it possible to combine

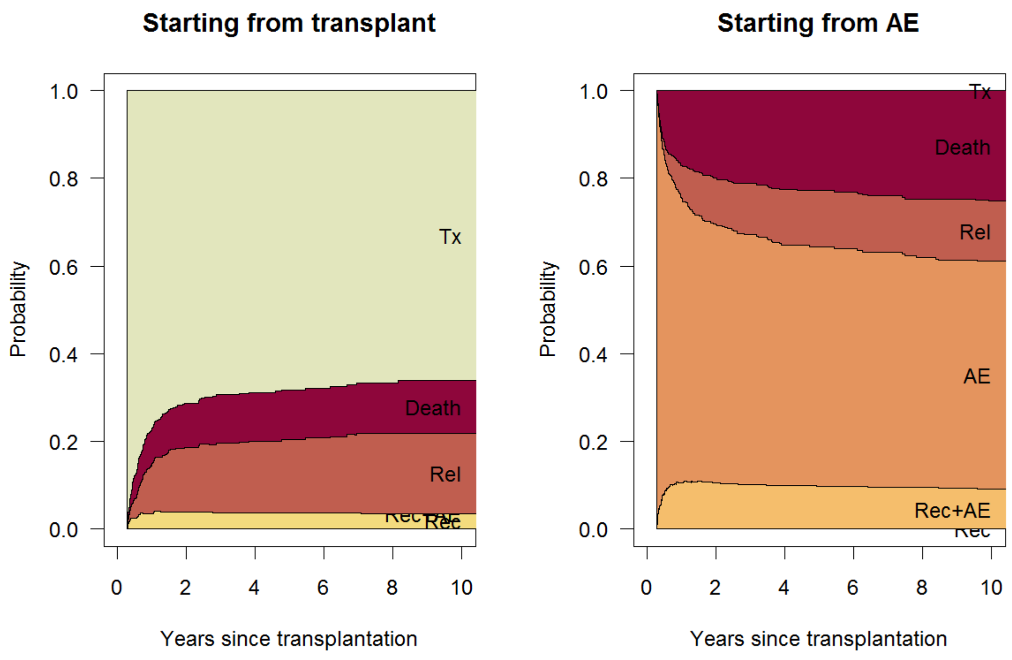

Figure 3: Non-parametric estimates of stacked transition probabilities at 100 days post-transplant. On the left starting state is 1 (transplant), on the right starting state is 3 (AE).

the probabilities of recovery and recovery + AE, and of AE and recovery + AE. Similar figures based on the situation after 100 days can be created by choosing `predt=100`. We compare the prognosis of two patients, without AE (Patient 1; Figure 3 left-hand side) and with AE (Patient 2; Figure 3 right-hand side), both 100 days after transplant.

```
R> pt100 <- probtrans(msf0, predt = 100/365.25, method = "greenwood")
R> plot(pt100, ord = c(2, 4, 3, 5, 6, 1),
+    xlab = "Years since transplantation", main = "Starting from transplant",
+    xlim = c(0, 10), las = 1, type = "filled", col = statecols[ord])
R> plot(pt100, from = 3, ord = c(2, 4, 3, 5, 6, 1),
+    xlab = "Years since transplantation", main = "Starting from AE",
+    xlim = c(0, 10), las = 1, type = "filled", col = statecols[ord])
```

A comparison of Figure 2 and Figure 3 clearly shows that the fact that patient 1 has not had any adverse event in the first 100 days post-transplant has improved his/her prognosis considerably; notably, his/her probability of long-term relapse-free survival has increased significantly. On the other hand, the long-term relapse-free survival of patient 2 is unchanged by the fact that he/she has experienced the adverse event, its negative impact being balanced by the good news of still being alive and relapse-free at 100 days. The relapse probability for patient 2 is somewhat smaller than that for patient 1 and the most likely scenario for 2 is that he/she will have no further events.

When reading the figures, one must keep in mind that we consider no further information about the patients who have had a relapse as this is considered an absorbing state; many of them may die as well. Moreover, being in an intermediary state, such as AE, must be interpreted as being alive after having experienced the entering event; this does not necessarily mean that the patient continues to suffer from the event.

For the analysis of non-parametric multi-state models, several other R packages are also available, but they all have some limitations. Among them are **mvna** (Allignol *et al.* 2008) to calculate the Nelson-Aalen estimator of the cumulative hazard and **etm** (Allignol *et al.* 2011) to calculate the Aalen-Johansen estimator of the transition probability matrix and its variance-covariance matrix. Contrary to **mstate**, **mvna** and **etm** cannot be applied to semi-parametric models.

4. Semi-parametric models

4.1. A model with transition-specific covariates

Next, we will show how the prediction of the transition probabilities can be improved by taking covariates at baseline into account. The **mstate** package supports the analysis of type-specific Cox models. Events are of the same 'type' or 'stratum' if they share a baseline hazard. In this section, we consider a model in which 'type' is equivalent to transition: each transition has its own baseline hazard. In Section 4.2, we consider a so-called 'proportional baseline hazards model'. In both models, covariates can have the same effect for all transitions or different effects for different transitions; in the latter case, transition-specific covariates are needed. For details see de Wreede *et al.* (2010).

The model that we consider in this section is a transition-specific Cox model:

$$\alpha_{gh}(t \,|\, \mathbf{Z}) = \alpha_{gh,0}(t) \exp(\boldsymbol{\beta}^\top \mathbf{Z}_{gh}) \ , \tag{5}$$

where gh indicates a transition from state g to state h, $\alpha_{gh,0}(t)$ is the baseline hazard for this transition, $\mathbf{Z}$ is the vector of covariates at baseline and $\mathbf{Z}_{gh}$ is the vector of transition-specific covariates (see page 7 for covariates expansion). This model specifies different covariate effects for the different transitions, as well as separate baseline transition hazards for each transition. For the estimators of the regression coefficients and baseline hazards, see de Wreede *et al.* (2010). Equation (4) also holds for this model.

In our leading example, we have tested for each covariate separately whether its effect is the same for all transitions or different across transitions by means of a likelihood ratio test. The results suggest a model in which all covariates have transition-specific effects. We refer to this model as the full model. In the call to `coxph()`, each of the expanded covariates has to be included. To specify that each transition has its own baseline transition intensity, `+ strata(trans)` has to be added to the covariates.

```
R> cfull <- coxph(Surv(Tstart, Tstop, status) ~ match.1 +
+    match.2 + match.3 + match.4 + match.5 + match.6 +
+    match.7 + match.8 + match.9 + match.10 + match.11 +
+    match.12 + proph.1 + proph.2 + proph.3 + proph.4 +
+    proph.5 + proph.6 + proph.7 + proph.8 + proph.9 +
+    proph.10 + proph.11 + proph.12 + year1.1 + year1.2 +
+    year1.3 + year1.4 + year1.5 + year1.6 + year1.7 +
+    year1.8 + year1.9 + year1.10 + year1.11 + year1.12 +
+    year2.1 + year2.2 + year2.3 + year2.4 + year2.5 +
+    year2.6 + year2.7 + year2.8 + year2.9 + year2.10 +
+    year2.11 + year2.12 + agecl1.1 + agecl1.2 + agecl1.3 +
+    agecl1.4 + agecl1.5 + agecl1.6 + agecl1.7 + agecl1.8 +
+    agecl1.9 + agecl1.10 + agecl1.11 + agecl1.12 + agecl2.1 +
+    agecl2.2 + agecl2.3 + agecl2.4 + agecl2.5 + agecl2.6 +
+    agecl2.7 + agecl2.8 + agecl2.9 + agecl2.10 + agecl2.11 +
+    agecl2.12 + strata(trans), data = msebmt, method = "breslow")
```

The estimated regression coefficients of the covariates and their standard errors for each of the transitions are shown in Table 2. For each covariate, the estimated effects are positive for some transition hazards and negative for others. The use of transition-specific covariates is very convenient to observe such effects. A model without transition-specific covariates could be estimated by substituting these by the basic covariates in the function call above.

We will now illustrate the use of `probtrans()` to do prediction for two example patients (see Table 3). Patient A has a (relatively) good prognosis and patient B has a bad prognosis. The prediction will be done in two steps. First, `msfit()` is used to estimate the transition intensities specific to these two patients. Similar to `survfit()` in the **survival** package, a `newdata` argument can be defined, specifying the values of the covariates of the patient. This `newdata` data frame looks somewhat different from `newdata` in `survfit()`. In `msfit()`, `newdata` needs to have as many rows as the number of transitions and each row needs to contain the values of all the (transition-specific) covariates used in the `coxph` object. An additional column `strata` specifies to which stratum in the `coxph` object each transition corresponds. In this case, the values of `strata` are equal to those of `trans` (from the '`msdata`' object) because every transition has its own baseline hazard. The fastest way to create data frames for these example patients is by selecting a patient from the data set with the correct characteristics of interest. Alternatively one could specify the basic covariates and apply `expand.covs()` to obtain transition-specific covariates.

For the current model, `msfit()` can only calculate Aalen-type standard errors, because the Greenwood estimator is not defined for models with covariates. The code for patient A is shown below; the code for patient B is similar. The first commands select the covariates and inherit the factor levels of the first patient in the data set with the required characteristics.

```
R> whA <- which(msebmt$proph == "yes" & msebmt$match == "no gender mismatch"
+    & msebmt$year == "1995-1998" & msebmt$agecl == "<=20")
R> patA <- msebmt[rep(whA[1], 12), 9:12]
R> patA$trans <- 1:12
R> attr(patA, "trans") <- tmat
R> patA <- expand.covs(patA, covs, longnames = FALSE)
R> patA$strata <- patA$trans
R> msfA <- msfit(cfull, patA, trans = tmat)
```

The second step in obtaining the predictions is to use the '`msfit`' object as input for `probtrans()`, which calculates $\widehat{\mathbf{P}}(s,t)$ from $\widehat{\mathbf{A}}$. Although usually $\widehat{\mathbf{A}}$ will have been created by a call to `msfit()`, this is not necessary. Any self-created object of class '`msfit`' which contains the estimated cumulative hazards for all transitions and their variances (the latter only if standard errors of $\widehat{\mathbf{P}}$ are required) can be used.

Tran-	Match	Prophylaxis	Year of transplant		Age at transplant	
sition			1990–1994	1995–1998	20–40	> 40
1	*-0.167* (0.085)	***-0.366*** (0.093)	***0.401*** (0.100)	***0.521*** (0.103)	0.049 (0.089)	0.199 (0.102)
2	-0.111 (0.079)	***-0.278*** (0.083)	0.023 (0.084)	-0.114 (0.091)	0.123 (0.083)	0.067 (0.101)
3	0.196 (0.224)	0.385 (0.227)	0.442 (0.245)	0.221 (0.302)	-0.094 (0.232)	-0.232 (0.322)
4	-0.003 (0.181)	-0.056 (0.179)	-0.359 (0.193)	*-0.476* (0.218)	***0.766*** (0.229)	***0.934*** (0.264)
5	0.190 (0.153)	-0.282 (0.196)	-0.095 (0.191)	-0.151 (0.190)	0.292 (0.188)	*0.470* (0.205)
6	*0.426* (0.214)	0.268 (0.221)	-0.210 (0.263)	0.055 (0.259)	-0.255 (0.223)	-0.101 (0.264)
7	0.244 (0.405)	-0.008 (0.378)	*-0.836* (0.398)	*-0.980* (0.442)	0.150 (0.491)	***1.465*** (0.481)
8	0.126 (0.113)	0.125 (0.125)	***0.528*** (0.135)	***0.930*** (0.141)	***-0.393*** (0.116)	*-0.328* (0.142)
9	-0.414 (0.352)	0.159 (0.321)	-0.311 (0.300)	-0.580 (0.433)	0.173 (0.367)	0.423 (0.433)
10	0.008 (0.168)	0.324 (0.166)	***-0.644*** (0.173)	-0.213 (0.195)	0.238 (0.205)	*0.495* (0.237)
11	-0.301 (0.248)	0.012 (0.247)	-0.024 (0.253)	-0.390 (0.277)	0.414 (0.250)	0.256 (0.304)
12	***0.572*** (0.179)	-0.118 (0.217)	-0.362 (0.228)	-0.352 (0.238)	***0.760*** (0.272)	***1.337*** (0.287)

Table 2: Regression coefficients (and standard errors) for the full model; covariate effects significant at 0.05 and 0.01 levels are shown in italics and in boldface italics, respectively.

Prognostic factor	Patient A	Patient B
Donor recipient	No gender mismatch	Gender mismatch
Prophylaxis	Yes	No
Year of transplant	1995-1998	1985-1989
Age at transplant (years)	18	43

Table 3: Covariates for patient A and patient B

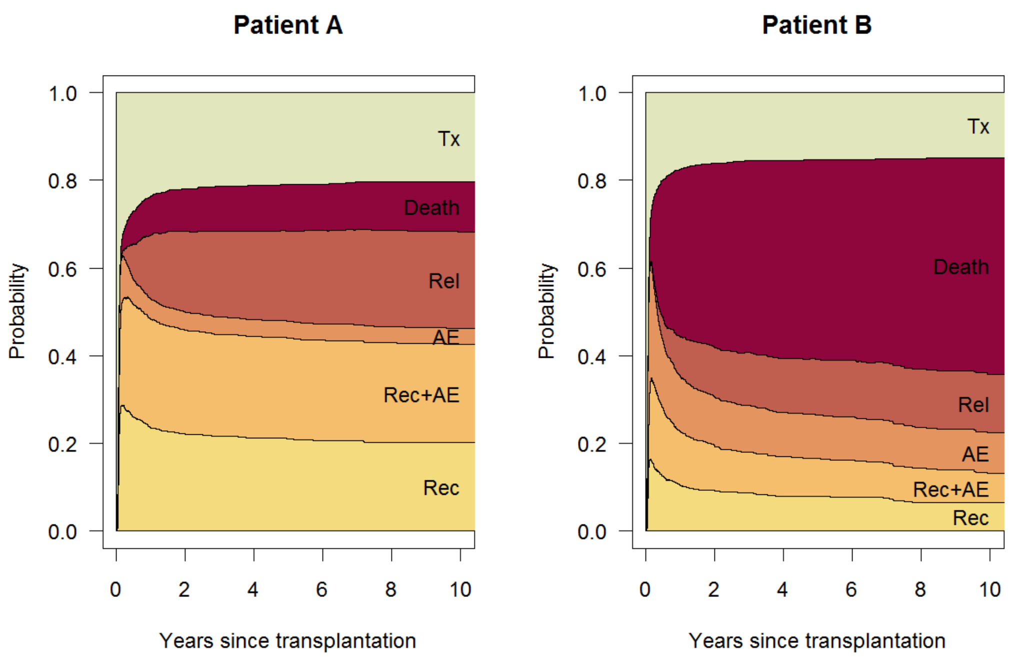

Figure 4: Stacked transition probabilities, starting from state 1 at $t = 0$, semi-parametric model

```
R> ptA <- probtrans(msfA, predt = 0)
```

Figure 4 shows the predictions for patients A and B from time 0, starting from the post-transplant state. Again, the code for patient B is similar to that for patient A.

```
R> plot(ptA, ord = c(2, 4, 3, 5, 6, 1), main = "Patient A",
+    las = 1, xlab = "Years since transplantation", xlim = c(0, 10),
+    type = "filled", col = statecols[ord])
```

As was to be expected on the basis of the covariates, the prospects for patient B are indeed worse than those for patient A, the former having a far larger probability of relapsing or dying.

From a clinical point of view, it is worthwhile to update this prediction if more information becomes known. Assume that both patients are in state 3 (AE, no Rec or relapse) at 100 days post-transplant (compare Figure 3, right-hand side).

```
R> pt100A <- probtrans(msfA, predt = 100/365.25)
```

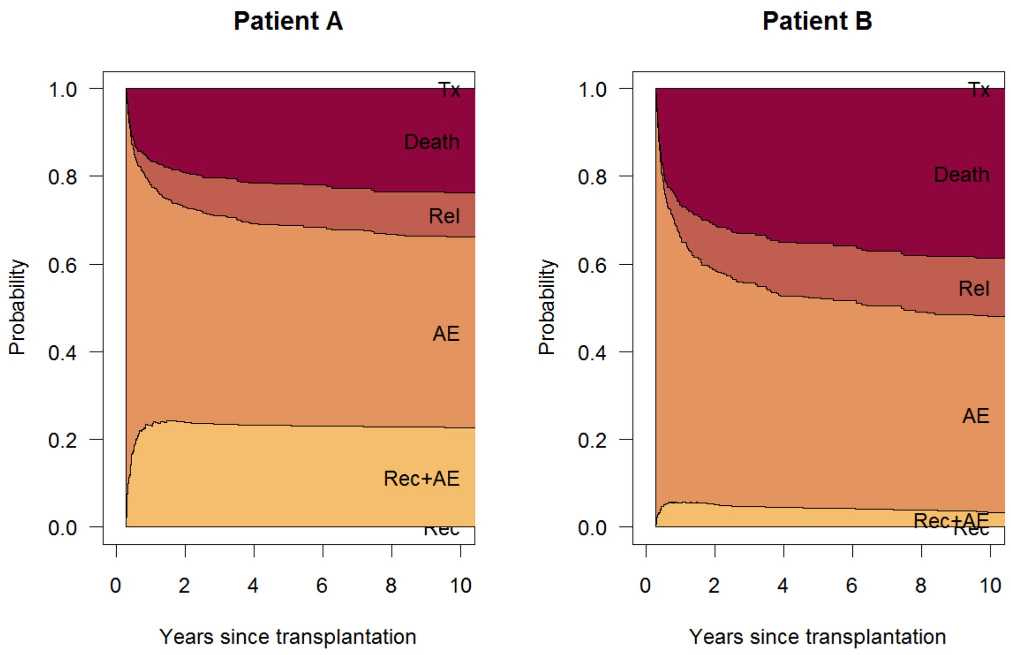

Figure 5: Stacked transition probabilities, starting from state 3 at $t = 100$, semi-parametric model

Their updated predictions are shown in Figure 5.

```
R> plot(pt100A, from = 3, ord = c(2, 4, 3, 5, 6, 1), main = "Patient A",
+    las = 1, xlab = "Years since transplantation", xlim = c(0, 10),
+    type = "filled", col = statecols[ord])
```

For patient A the prognosis of relapse-free survival (i.e., being in state 1, 2, 3 or 4) is about the same as the prognosis just after transplant, but the distribution of the probabilities is different from the previous situation. By surviving the first 100 days, the prospects of patient B regarding relapse-free survival have somewhat improved.

The software also enables the user to make a different kind of predictions for individual patients. Suppose we are interested in dynamic prediction of 10-year relapse-free survival (RFS) probabilities. The question is how these prediction probabilities change as more information about intermediate events becomes known in the course of time. We consider again patient A and we want to study how 10-year RFS probabilities change when the patient experiences the adverse event 60 days (0.164 years) post-transplant and is recovered from the treatment 80 days (0.219 years) post-transplant. The 10-year survival probabilities can all be calculated by setting the `direction` in `probtrans()` to `"fixedhorizon"` (or in fact any string starting with `"fi"`).

```
R> ptA10yrs <- probtrans(msfA, predt = 10, direction = "fixedhorizon")
R> head(ptA10yrs[[1]])
```

```
         time   pstate1   pstate2    pstate3   pstate4   pstate5
1 0.000000000 0.2027821 0.2011028 0.03643172 0.2243403 0.2202891
2 0.002737851 0.2028442 0.2011644 0.03642153 0.2242138 0.2203215
3 0.008213552 0.2030930 0.2014111 0.03638068 0.2237070 0.2204516
4 0.010951403 0.2033788 0.2016946 0.03634618 0.2232383 0.2206213
5 0.013689254 0.2040161 0.2023266 0.03626654 0.2221685 0.2209954
6 0.016427105 0.2046623 0.2028231 0.03618690 0.2209705 0.2212890
    pstate6        se1        se2        se3        se4        se5
1 0.1150540 0.03251177 0.03348491 0.01310601 0.03201052 0.03621101
2 0.1150345 0.03252166 0.03349510 0.01310245 0.03200975 0.03621862
3 0.1149566 0.03256129 0.03353594 0.01308821 0.03200663 0.03624914
4 0.1147208 0.03260681 0.03358284 0.01307626 0.03200914 0.03628623
5 0.1142269 0.03270819 0.03368732 0.01304868 0.03201329 0.03636863
6 0.1140682 0.03281083 0.03377366 0.01302117 0.03200179 0.03644010
         se6
1 0.01778362
2 0.01778126
3 0.01777181
4 0.01776477
5 0.01774802
6 0.01772962
```

We can extract the dynamic predictions of 10-year RFS probabilities for this specific patient directly. The '`probtrans`' object `ptA10yrs` contains estimates of $P_{gh}(s, 10)$. For the 10-year RFS probabilities the quantity $1 - (P_{g5}(s, 10) + P_{g6}(s, 10))$ is needed, in which g indicates the state in which the patient is at time s. For $0 \leq s < 0.164$ (time of the AE) patient A is in state 1 and we extract $1 - (P_{15}(s, 10) + P_{16}(s, 10))$ from the first list element `ptA10yrs[[1]]`. For $0.164 \leq s < 0.219$ (time of Recovery), patient A is in state 3 and now we need $1-(P_{35}(s, 10)+P_{36}(s, 10))$ from the third list element `ptA10yrs[[3]]`. Finally, for $0.219 \leq s < 10$, we extract the relevant information from `ptA10yrs[[4]]`. The results of this fixed horizon prediction are plotted in Figure 6 ($s \leq 2$ years). They show the 10-year RFS probabilities for patient A based on his/her history as described above. The dashed lines show what the survival probabilities would have been if the patient had not had a transition. We see that her/his prognosis becomes worse the moment when the adverse event occurs and improves when recovery takes place. When time progresses, the prognosis improves irrespective of the current state simply because he/she has already survived a potentially dangerous period. Note that these curves, although tending to increase in general, do not have to be monotonely increasing, as the shape of the black curve shows (see also van Houwelingen and Putter 2008).

The function `probtrans()` only needs to be called once, since all the information from different starting states is present in a single '`probtrans`' object. Without further computations we could study changes in 10-year RFS probabilities for patients with the same covariate values but different event histories.

4.2. A proportional baseline hazards model

In the full model discussed in Section 4.1, 12 baseline hazards and 72 covariate effects have to be estimated. These numbers can be reduced in several ways. One of them is to consider a

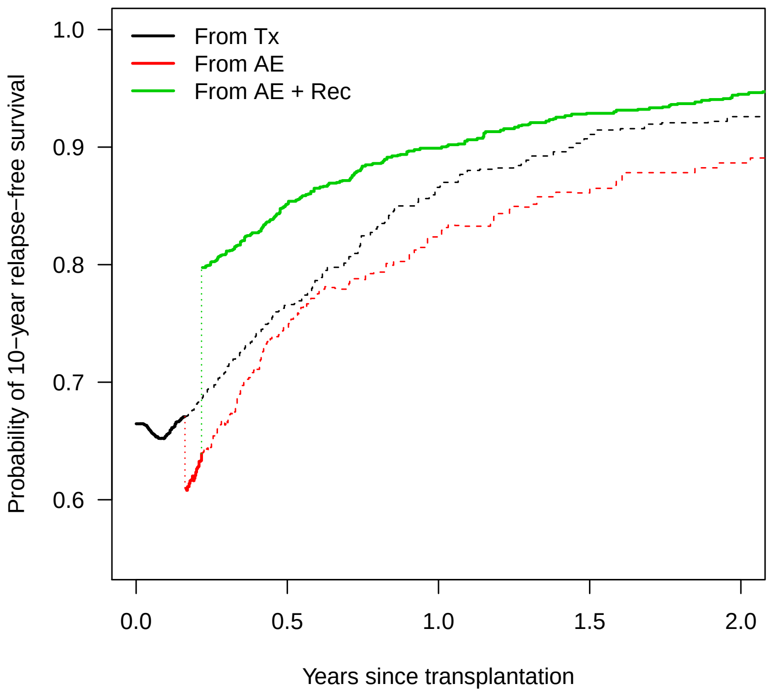

Figure 6: Dynamic prediction of 10-year relapse-free survival for patient A, semi-parametric model

model in which some of the baseline hazards are assumed to be proportional. Alternatively, the number of regression parameters to be estimated can be reduced by applying reduced rank techniques; these techniques, well known in regression theory, have been adapted to the multi-state context. This will be explored in Section 4.3.

In the current section, we consider a model in which we assume that all transitions into the Relapse state have a common baseline hazard and that all transitions into the Death state have another common baseline hazard. These are reasonable assumptions from a clinical point of view and they can be checked using standard methods. In formula, these are expressed as $\alpha_{gh,0}(t) = \tilde{\delta}_{gh}\alpha_{1h,0}(t)$, $g = 2, 3, 4; h = 5, 6$. The transition intensities from Tx to Relapse and to Death serve as baseline transition intensities. To estimate the $\tilde{\delta}_{gh}$s, we need a specific kind of time-dependent covariates $\tilde{Z}_g(t)$ in the regression model, $g = 2, 3, 4$. Within a stratum or type, this covariate distinguishes between different transitions into the same state: $\tilde{Z}_g(t)$ is 1 after the patient has moved to state g and 0 otherwise. These covariates may be expanded into transition-specific covariates as before. The proportionality is then expressed by the coefficient $\tilde{\boldsymbol{\beta}}_{gh}$ of $\tilde{Z}_{gh}(t)$, in which $\tilde{Z}_{gh}(t)$ indicates the transition-specific covariate for the transition $g \to h$ expanded from $\tilde{Z}_g(t)$: $\exp(\tilde{\boldsymbol{\beta}}_{gh}) = \tilde{\delta}_{gh}$ (for details see de Wreede *et al.* (2010)). Each of the other transitions also determines one stratum.

Other covariates in the new model can again be analyzed both as generic covariates and as transition-specific covariates. For clinical reasons, the last option will again be further pursued here. The new model has 6 baseline hazards instead of the 12 of the model of Section 4.1. The hazard rates, transitions probabilities and their asymptotic standard errors can be calculated as before.

The data now have to be extended with a `strata` column (the column name is irrelevant), indicating which transitions are together in one type or stratum, and with the time-dependent covariates indicating for which transition within a stratum the patient is at risk. Once the data have been adjusted, the analysis is very similar to that shown before.

```
R> msebmt$strata <- msebmt$trans
R> msebmt$strata[msebmt$trans %in% c(6, 9, 11)] <- 3
R> msebmt$strata[msebmt$trans %in% c(7, 10, 12)] <- 4
R> msebmt$strata[msebmt$trans == 8] <- 6
R> msebmt$Z2 <- 0
R> msebmt$Z2[msebmt$trans %in% c(6, 7)] <- 1
R> msebmt$Z3 <- 0
R> msebmt$Z3[msebmt$trans %in% c(9, 10)] <- 1
R> msebmt$Z4 <- 0
R> msebmt$Z4[msebmt$trans %in% c(11, 12)] <- 1
R> msebmt <- expand.covs(msebmt, covs = c("Z2", "Z3", "Z4"))
```

In this example, the coefficient of the time-dependent transition-specific covariate `Z2.6` measures the change in the relapse hazard after recovery (`Z2`=1, transition 6, from state 2 to state 5) compared to the relapse hazard without recovery (`Z2`=0, transition 3, from state 1 to state 5). In formula: $\alpha_{25,0}(t) = \exp(\tilde{\beta}_{2,25})\alpha_{15,0}(t)$.

The call to `coxph()` now includes the same 72 transition-specific covariates as in the full model (`cfull`), plus the six covariates measuring the effects of occurrences of intermediate events on relapse and death. Instead of stratifying by transition, we stratify by `strata`, which means that six baseline hazards are estimated instead of twelve. The first part of the function call with all transition specific covariates is equal to that of `cfull`, the last part becomes `+Z2.6+Z2.7+Z3.9+Z3.10+Z4.11+Z4.12+strata(strata)`. The coefficients of this Cox model are very close to those of the full model of Section 4.1. Of the new covariates Z_{gh}, only the coefficient of `Z3.10` is significant ($p = 0.0024$); the hazard ratio of `Z3.10` equals 2.60, which means that, adjusted for the other covariates, the occurrence of the adverse event increases the rate of dying with a factor of 2.60. The main advantage of proportional baseline hazards models is precisely that we obtain such a measure for the impact of intermediate events on the final outcome.

The prediction for Patient A is very similar to the procedure discussed before. Only the values of the `strata` column need to be adjusted and the time-dependent covariates `Z2.6`, ..., `Z4.12` need to be added.

```
R> patAPH <- patA
R> patAPH$strata[6:12] <- c(3, 4, 6, 3, 4, 3, 4)
R> patAPH$Z2 <- 0
R> patAPH$Z2[patAPH$trans %in% c(6, 7)] <- 1
R> patAPH$Z3 <- 0
R> patAPH$Z3[patAPH$trans %in% c(9, 10)] <- 1
R> patAPH$Z4 <- 0
```

```
R> patAPH$Z4[patAPH$trans %in% c(11, 12)] <- 1
R> patAPH <- expand.covs(patAPH, covs = c("Z2", "Z3", "Z4"))
R> msfAPH <- msfit(coxPH, patAPH, trans = tmat)
R> ptAPH <- probtrans(msfAPH, predt = 0)
```

Finally, Figure 7 shows the standard errors of the transition probabilities from state 3 (AE) at time 100 days to states 5 (Relapse) and 6 (Death), for the full Cox model (non-PH) and for the proportional baseline hazards model (PH). It is obtained by the following code:

```
R> pt100APH <- probtrans(msfAPH, predt = 100/365.25)
R> ptA3 <- pt100A[[3]]
R> ptAPH3 <- pt100APH[[3]]
R> plot(ptA3$time, ptA3$se6, xlim = c(0, 2), type = "s", lwd = 2,
+    xlab = "Years since transplantation", ylab = "Standard errors", las = 1)
R> lines(ptAPH3$time, ptAPH3$se6, type = "s", lwd = 2, lty = 2)
R> text(2, 0.085, "Death", adj = 1)
R> lines(ptA3$time, ptA3$se5, type = "s", lwd = 2, col = 8)
R> lines(ptAPH3$time, ptAPH3$se5, type = "s", lwd = 2, lty = 2, col = 8)
R> text(2, 0.0425, "Relapse", adj = 1)
R> legend("topleft", c("non-PH", "PH"), lwd = 2, lty = c(1, 2), bty = "n")
```

The figure shows that the proportional hazards assumption does not always decrease the standard errors of the predictions.

4.3. Reduced rank models and simulation

In this section two more specialized functions of **mstate** are illustrated. The first of these is useful to obtain a lower dimensional representation of the regression coefficients of the full model of Section 4.1. In our example, 72 coefficients were estimated. Table 2 does not give a clear overview of the structure of the covariate effects on the transitions. By reducing the rank R of the matrix $\mathbf{B}$ of regression coefficients we reduce the number of parameters to be estimated. The matrix $\mathbf{B}$ can be factorized as $\mathbf{B} = \mathbf{A}\mathbf{\Gamma}^\top$. This implies that the number of free parameters that need to be estimated is reduced from $p \times K$ to $R(p + K - R)$, where p and K denote respectively the number of covariates and the number of transitions in the model. For more details see Fiocco *et al.* (2005) and Fiocco *et al.* (2008).

We will illustrate the rank 1 model, which is formulated as

$$\alpha_k(t \,|\, \mathbf{Z}) = \alpha_{k,0}(t) \exp(\gamma_k \boldsymbol{\alpha}^\top \mathbf{Z}) \ ,$$

for transitions numbered $k = 1, \ldots, K$. In this model all covariates have the same effect (given by the parameter vector $\boldsymbol{\alpha}$) on each transition apart from the proportionality coefficients γ_k. The factor $\boldsymbol{\alpha}^\top \mathbf{Z}$ can be seen as a prognostic score for a patient with a vector of covariates $\mathbf{Z}$; this prognostic score determines how likely a patient is to experience an event. The parameter γ_k determines the size of the effect of the prognostic score on transition k. In this example, we use a model in which each transition has its own baseline hazard, but this is not necessary.

The function `redrank()` estimates the parameters of a reduced rank model:

```
R> rr1 <- redrank(Surv(Tstart, Tstop, status) ~ match + proph +
+    year + agecl, data = msebmt, R = 1, print.level = 0)
```

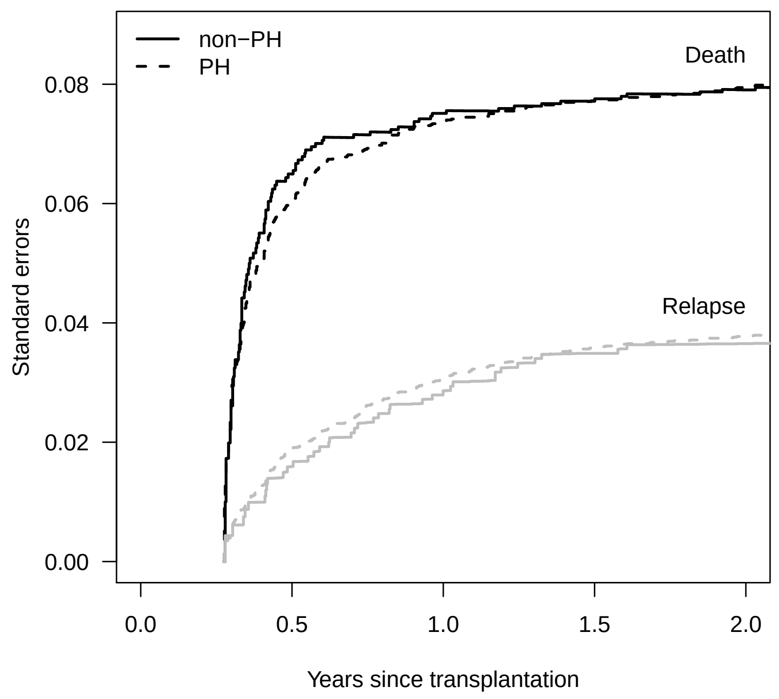

Figure 7: Standard errors for the transition probabilities AE → Relapse and AE → Death (starting time: 100 days), both for the full model and the proportional baseline hazards model

```
R> rr1$Alpha
```

```
                               r1
matchgender.mismatch -0.09565419
prophyes             -0.19069469
year1990.1994         0.51451609
year1995.1998         0.72012922
agecl20.40           -0.24985943
agecl.40             -0.32976231
```

```
R> rr1$Gamma
```

```
   Tx -> Rec   Tx -> AE  Tx -> Rel Tx -> Death Rec -> Rec+AE
r1  0.757664 0.06114974 0.06000156  -0.6898289    -0.1808698
    Rec -> Rel Rec -> Death AE -> Rec+AE  AE -> Rel AE -> Death
r1 -0.03471962    -1.244175     1.016525 -0.6485648  -0.8150607
   Rec+AE -> Rel Rec+AE -> Death
r1    -0.4651913      -0.8487646
```

The prognostic index in the `Alpha` item of `rr1` is higher for later years of transplantation and for younger age. The coefficients for this risk score in `Gamma` are negative and of substantial size for all transitions into death (and for many into relapse). This means that in this case lower values of `Alpha` (for instance higher age) correspond to higher death rates, showing that a higher prognostic index does not necessarily imply a higher risk of an adverse event. In contrast, the effects of `Alpha` on the "good" event of recovery (Tx $\rightarrow$ Rec and AE $\rightarrow$ Rec+AE) are positive. The matrix of regression coefficients $\mathbf{B}$ obtained by $\mathbf{B} = \mathbf{A}\mathbf{\Gamma}^\top$ is given in `rr1$Beta`.

```
R> rr1$Beta
```

```
                      Tx -> Rec     Tx -> AE   Tx -> Rel Tx -> Death
matchgender.mismatch -0.07247373 -0.005849229 -0.00573940  0.06598502
prophyes             -0.14448250 -0.011660931 -0.01144198  0.13154671
year1990.1994         0.38983030  0.031462526  0.03087177 -0.35492807
year1995.1998         0.54561595  0.044035715  0.04320887 -0.49676595
agecl20.40           -0.18930949 -0.015278840 -0.01499195  0.17236026
agecl.40             -0.24984902 -0.020164880 -0.01978625  0.22747958
                     Rec -> Rec+AE   Rec -> Rel Rec -> Death
matchgender.mismatch    0.01730095  0.003321077    0.1190106
prophyes                0.03449091  0.006620848    0.2372576
year1990.1994          -0.09306041 -0.017863804   -0.6401481
year1995.1998          -0.13024961 -0.025002615   -0.8959669
agecl20.40              0.04519202  0.008675025    0.3108689
agecl.40                0.05964404  0.011449223    0.4102821
                     AE -> Rec+AE   AE -> Rel AE -> Death
matchgender.mismatch   -0.0972349  0.06203794  0.07796397
prophyes               -0.1938460  0.12367786  0.15542775
year1990.1994           0.5230186 -0.33369701 -0.41936183
year1995.1998           0.7320296 -0.46705043 -0.58694900
agecl20.40             -0.2539884  0.16205002  0.20365060
agecl.40               -0.3352117  0.21387222  0.26877629
                     Rec+AE -> Rel Rec+AE -> Death
matchgender.mismatch    0.04449750      0.08118789
prophyes                0.08870952      0.16185491
year1990.1994          -0.23934843     -0.43670305
year1995.1998          -0.33499787     -0.61122020
agecl20.40              0.11623244      0.21207184
agecl.40                0.15340257      0.27989058
```

This matrix $\mathbf{B}$ may be compared with that of the full model shown in Table 2. Differences between the reduced rank matrix and Table 2 may be explained both by parameter estimate uncertainty of both models and by a possible lack of fit of the reduced rank model of rank 1. If the rank 1 model is too simplistic, reduced rank models of higher rank may be investigated.

The standard errors of the estimates can be obtained by means of a bootstrap procedure. The bootstrap is useful in cases where asymptotic variances of estimators are not available in closed form or may be very complicated to compute. In **mstate**, a non-parametric bootstrap procedure has been implemented through the function `msboot()`. Its main argument `theta` is a function that should return a real-valued scalar or vector. The function `theta` itself should have an object of class ‘`msdata`’ as its first argument. `msboot()` is modeled after `boot()` in the **boot** package, but one important difference is that in the multi-state context of `msboot()`, the dataframe contains multiple rows of data for the same individual.

As an example, we calculate the standard errors of the **B** matrix of the reduced rank regression of rank 1 illustrated above. For this purpose, we define `theta` as the stacked vector representation of **B**. First, a function is defined that turns **B** into a vector:

```
R> rr1beta <- function(data) {
+    rr1 <- redrank(Surv(Tstart, Tstop, status) ~ match +
+    proph + year + agecl, data = data, R = 1, print.level = 0)
+    return(as.vector(rr1$Beta))
+  }
R> th <- rr1beta(msebmt)
```

Now `msboot()` is called with `rr1beta()` as `theta` argument. The other arguments include `data` (an object of class ‘`msdata`’), `id` (needed to identify individuals in the data), and `B`, which specifies the number of bootstrap simulations.

```
R> set.seed(1234)
R> msb <- msboot(theta = rr1beta, data = msebmt, id = "id",
+    B = 500, verbose = 0)
```

The result is a matrix with `B` columns, each of which contains the result of the `theta` function applied to a bootstrap data set (see Fiocco *et al.* (2008) for details). It can be used to assess the bias and compute the standard errors of **B**. For instance, the standard errors are calculated as

```
R> sqrt(apply(msb, 1, var))
```

Finally, it is possible to use simulation to calculate transition probabilities. In Markov models this is not necessary, because calculation of transition probabilities is implemented in `probtrans()`, but especially in models where the Markov assumption is not fulfilled, simulation is very useful. In **mstate**, a function `mssample()` is provided to do this. We will illustrate this function by recomputing transition probabilities for patient A in Section 4. The code below uses simulation to approximate the probabilities already calculated in `ptA` (see p. 17). The argument `tvec` indicates a vector of time points at which the probabilities are to be calculated.

```
R> set.seed(1234)
R> msfAsample <- mssample(Haz = msfA$Haz, trans = tmat,
+    tvec = 1:10, M = 10000)
R> msfAsample

   time pstate1 pstate2 pstate3 pstate4 pstate5 pstate6
1     1  0.2305  0.2353  0.0473  0.2430  0.1526  0.0913
2     2  0.2134  0.2196  0.0418  0.2324  0.1929  0.0999
3     3  0.2084  0.2150  0.0408  0.2286  0.2040  0.1032
```

```
4     4  0.2064  0.2116  0.0393  0.2267  0.2098  0.1062
5     5  0.2045  0.2096  0.0391  0.2261  0.2137  0.1070
6     6  0.2028  0.2054  0.0391  0.2253  0.2201  0.1073
7     7  0.1977  0.2051  0.0386  0.2241  0.2252  0.1093
8     8  0.1977  0.2008  0.0378  0.2239  0.2281  0.1117
9     9  0.1973  0.2008  0.0377  0.2230  0.2282  0.1130
10   10  0.1973  0.2008  0.0377  0.2209  0.2282  0.1151
```

4.4. Competing risks models

The **mstate** package has been designed for general multi-state models. Since competing risks models are special cases of multi-state models, all the functionality for general multi-state models also applies to competing risks models. In particular, cumulative incidences for cause-specific proportional hazards models may be obtained using `msfit()` and `probtrans()`. Two functions in **mstate** are designed specifically for competing risks models: `trans.comprisk()`, which defines a transition matrix for competing risks models, and `Cuminc()`, which calculates non-parametric cumulative incidence functions and associated standard errors, possibly for subgroups defined by a categorical covariate. The vignette of **mstate** contains example code for the analysis of competing risks data using **mstate**; type `vignette("Tutorial", package= "mstate")` to access the vignette.

5. Discussion

Although multi-state models are a very useful tool to answer a wide range of questions in survival analysis, they are not frequently applied. So far, an important reason for this has been the lack of available software. For this reason we have developed a package in R that offers the user the opportunity to explore different kinds of multi-state models and estimate their parameters of interest on the basis of a regular data set containing the times to event of the events of interest and optionally covariate values. The functions in the package are flexible, which means that they can easily be combined with user-written software in cases when models not covered by **mstate** are studied.

In **mstate**, we restrict ourselves to non- and semi-parametric models. This means that for parametric models other software is needed, such as the R package **msm** developed by Christopher Jackson (see Jackson 2010, 2011). In this article, we have explored the role of the **mstate** functions in the different phases of a multi-state analysis: model building, data preparation, exploration of different covariate effects and baseline assumptions, estimation of hazards, transition probabilities and associated standard errors. In particular, it has been explained how predictions can be updated if more information becomes known (dynamic prediction). This possibility is an important extra feature of multi-state models compared to classical survival models. Moreover, several ways of presenting the outcomes in figures have been shown. Finally, we have presented some functions for reduced rank modeling and for bootstrap and simulation procedures.

In future releases of **mstate** we plan to implement prediction in Markov renewal models, with and without covariates. Formulas similar to the Aalen-Johansen estimator currently implemented in `probtrans()` also exist for this type of models, but are not yet available in any software package. Other methods are also planned to be included in **mstate**: dynamic prediction using landmarking of van Houwelingen (2007), vertical modeling in competing risks models of Nicolaie *et al.* (2010), the Fine-Gray regression method for competing risks with left truncation developed by Geskus (2010) and methods for quality-of-life adjusted survival in the context of multi-state models.

Acknowledgments

Research leading to this paper was supported by the Netherlands Organization for Scientific Research Grant ZONMW-912-07-018 "Prognostic modeling and dynamic prediction for competing risks and multi-state models". The authors are grateful to the EBMT for providing the data.

This document was prepared using `Sweave` (Leisch 2002).

References

Allignol A, Beyersmann J, Schumacher M (2008). "**mvna**: An R Package for the Nelson-Aalen Estimator in Multistate Models." *R News*, **8**, 48–50.

Allignol A, Schumacher M, Beyersmann J (2011). "Empirical Transition Matrix of Multistate Models: The **etm** Package." *Journal of Statistical Software*, **38**(4), 1–15. URL `http://www.jstatsoft.org/v38/i04/`.

Andersen PK, Borgan Ø, Gill RD, Keiding N (1993). *Statistical Models Based on Counting Processes*. Springer Series in Statistics, 2nd edition. Springer-Verlag.

de Wreede LC, Fiocco M, Putter H (2010). "The **mstate** Package for Estimation and Prediction in Non- and Semi-Parametric Multi-State and Competing Risks Models." *Computer Methods and Programs in Biomedicine*, **99**, 261–274.

Fiocco M, Putter H, van Houwelingen HC (2008). "Reduced Rank Proportional Hazards Regression and Simulation-Based Prediction for Multi-State Models." *Statistics in Medicine*, **27**, 4340–58.

Fiocco M, Putter H, van Houwelingen JC (2005). "Reduced Rank Proportional Hazards Model for Competing Risks." *Biostatistics*, **6**, 465–478.

Geskus R (2010). "Cause-Specific Cumulative Incidence Estimation and the Fine and Gray Model Under Both Left Truncation and Right Censoring." *Biometrics*. `doi:10.1111/j.1541-0420.2010.01420.x`.

Ihaka R, Murrell P, Hornik K, Zeileis A (2009). ***colorspace**: Color Space Manipulation*. R package version 1.0-1, URL `http://CRAN.R-project.org/package=colorspace`.

Jackson C (2010). *Multi-State Modelling with R: The **msm** Package*. R package version 1.0, URL `http://CRAN.R-project.org/package=msm`.

Jackson C (2011). "Multi-State Models for Panel Data: The **msm** Package for R." *Journal of Statistical Software*, **38**(8), 1–29. URL `http://www.jstatsoft.org/v38/i08/`.

Leisch F (2002). "Dynamic Generation of Statistical Reports Using Literate Data Analysis." In W Härdle, B Rönz (eds.), *COMPSTAT 2002 – Proceedings in Computational Statistics*, pp. 575–580. Physica-Verlag, Heidelberg.

Nicolaie MA, van Houwelingen HC, Putter H (2010). "Vertical Modeling: A Pattern Mixture Approach for Competing Risks Modeling." *Statistics in Medicine*, **29**, 1190–205.

Putter H, Fiocco M, Geskus RB (2007). "Tutorial in Biostatistics: Competing Risks and Multi-State Models." *Statistics in Medicine*, **26**, 2389–2430.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL `http://www.R-project.org/`.

Therneau T, Lumley T (2010). ***survival**: Survival Analysis Including Penalised Likelihood.* R package version 2.36-1, URL `http://CRAN.R-project.org/package=survival`.

van Houwelingen HC (2007). "Dynamic Prediction by Landmarking in Event History Analysis." *Scandinavian Journal of Statistics*, **34**, 70–85.

van Houwelingen HC, Putter H (2008). "Dynamic Predicting by Landmarking as an Alternative for Multi-State Modeling: An Application to Acute Lymphoid Leukemia Data." *Lifetime Data Analysis*, **14**, 447–463.

Zeileis A, Hornik K, Murrell P (2009). "Escaping RGBland: Selecting Colors for Statistical Graphics." *Computational Statistics & Data Analysis*, **53**, 3259–3270. `doi:10.1016/j.csda.2008.11.033`.

Affiliation:

Liesbeth C. de Wreede
Department of Medical Statistics and Bioinformatics
Leiden University Medical Center
P.O. Box 9600
2300 RC Leiden, The Netherlands
E-mail: `l.c.de_wreede@lumc.nl`
URL: `http://www.msbi.nl/multistate`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the American Statistical Association `http://www.amstat.org/`
Volume 38, Issue 7 *Submitted:* 2010-01-17
January 2011 *Accepted:* 2010-08-20