

Multi-state modelling with R: the msm package

Christopher Jackson

`chris.jackson@imperial.ac.uk`

Abstract

The multi-state Markov model is a useful way of describing a process in which an individual moves through a series of stages in continuous time. The `msm` package for R allows a general multi-state model to be fitted to longitudinal data. Data consist of observations of the process at arbitrary times, so that the exact times when the state changes are unobserved. For example, the progression of chronic diseases is often described by stages of severity, and the state of the patient may only be known at doctor or hospital visits. Features of `msm` include the ability to model transition rates between stages in terms of covariates. A form of hidden Markov model can also be fitted in which the states are misclassified. This manual introduces the theory behind multi-state models and gives a tutorial in the typical use of the `msm` package. Multi-state models with misclassification are covered in the final section.

Contents

1	Multi-state models	3
1.1	Introduction	3
1.2	Disease progression models	3
1.3	Arbitrary observation times	5
1.4	Likelihood for the multi-state model	5
1.5	Covariates	7
2	Fitting multi-state models with <code>msm</code>	9
2.1	Installing <code>msm</code>	9
2.2	Getting the data in	9
2.3	Specifying a model	10
2.4	Specifying initial values	12
2.5	Running <code>msm</code>	12
2.6	Showing results	13
2.7	Covariates on the transition rates	14
2.8	Fixing parameters at their initial values	16
2.9	Extractor functions	17
2.10	Convergence failure	19
2.11	Model assessment	20

3	Multi-state models with misclassification	23
3.1	Maximum likelihood estimation	24
3.2	Fitting misclassification models with <code>msm</code>	25
3.3	Effects of covariates on misclassification rates	27
3.4	Extractor functions	29
3.5	Recreating the path through underlying states	30
4	<code>msm</code> reference guide	32

1 Multi-state models

1.1 Introduction

Figure 1 illustrates a multi-state model in continuous time. Its four states are labelled **1, 2, 3, 4**. At a time t the individual is in state $S(t)$. The arrows show which transitions are possible between states. The next state to which the individual moves, and the time of the change, are governed by a set of *transition intensities* $q_{rs}(t, z(t))$ for each pair of states r and s . The intensities may also depend on the time of the process t , and a set of individual-specific or time-varying explanatory variables $z(t)$. The intensity represents the instantaneous risk of moving from state r to state s :

$$q_{rs}(t, z(t)) = \lim_{\delta t \rightarrow 0} P(S(t + \delta t) = s | S(t) = r) / \delta t \quad (1)$$

The intensities form a matrix Q whose rows sum to zero, so that the diagonal entries are defined by $q_{rr} = -\sum_{s \neq r} q_{rs}$. To fit a multi-state model to data, we estimate this transition intensity matrix. We concentrate on *Markov* models here. The Markov assumption is that future evolution only depends on the current state. That is, $q_{rs}(t, z(t), \mathcal{F}_t)$ is independent of \mathcal{F}_t , the observation history \mathcal{F}_t of the process up to the time preceding t .

See, for example, Cox and Miller[1] for a thorough introduction to the theory of continuous-time Markov chains.

1.2 Disease progression models

Many chronic diseases have a natural interpretation in terms of staged progression. Multi-state Markov models in continuous time are often used to model the course of diseases. A commonly-used model is illustrated in figure 2. This represents a series of successively more severe disease stages, and an ‘absorbing’ state, often death. The patient may advance into or recover from adjacent disease stages, or die at any disease stage. Observations of the stage $S_i(t)$ are made on a number of individuals i at arbitrary times t , which may vary between individuals. The stages of disease may be modelled as a homogeneous continuous-time Markov process, with a transition matrix Q , pictured below figure 2.

A wide range of medical situations have been modelled using multi-state methods, for example, screening for abdominal aortic aneurysms (Jackson *et al.*[2]), problems following lung transplantation (Jackson and Sharples[3]), problems following heart transplantation (Sharples[4], Klotz and Sharples[5]), hepatic cancer (Kay[6]), HIV infection and AIDS (Longini *et al.*[7], Satten and Longini[8], Guihenneuc-Jouyaux *et al.*[9], Gentleman *et al.*[10]), diabetic complications (Marshall and Jones[11], Andersen[12]), breast cancer screening (Duffy and Chen[13], Chen *et al.*[14]), cervical cancer screening (Kirby and Spiegelhalter[15]) and liver cirrhosis (Andersen *et al.*[16]). Many of these references also describe the mathematical theory, which will be reviewed in the following sections.

A commonly-used model is the *illness-death* model, with three states representing health, illness and death (figure 3). Transitions are permitted from health to illness, illness to death and health to death. Recovery from illness to health may also be permitted.

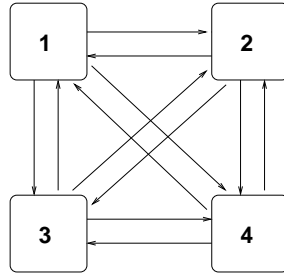
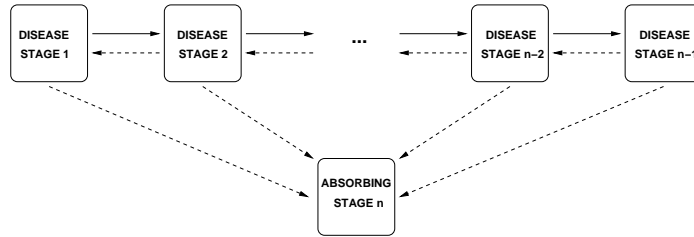


Figure 1: General multi-state model

$$Q = \begin{pmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{pmatrix}$$

Figure 2: General model for disease progression



$$Q = \begin{pmatrix} q_{11} & q_{12} & 0 & 0 & \dots & q_{1n} \\ q_{21} & q_{22} & q_{23} & 0 & \dots & q_{2n} \\ 0 & q_{32} & q_{33} & q_{34} & \ddots & q_{3n} \\ 0 & 0 & q_{43} & q_{44} & \ddots & q_{4n} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 \end{pmatrix}$$

1.3 Arbitrary observation times

Longitudinal data from monitoring disease progression are often incomplete in some way. Usually patients are seen at intermittent follow-up visits, at which monitoring information is collected, but information from the periods between visits is not available. Often the exact time of disease onset is unknown. Thus, the changes of state in a multi-state model usually occur at unknown times. Also a subject may only be followed up for a portion of their disease history. A fixed observation schedule may be specified in advance, but in practice times of visits may vary due to patient and hospital pressures. The states of disease progression models often include death. Death times are commonly recorded to within a day.

A typical sampling situation is illustrated in figure 4. The individual is observed at four occasions through 10 months. The final occasion is the death date which is recorded to within a day. The only other information available is the occupation of stages 1, 1, and 0 at respective times 1.5, 3.5 and 5. The times of movement between stages and the stage occupancy in between the observation times are unknown. Although the patient was in stage 2 between 7 and 9 months this was not observed at all.

Informative sampling times To fit a model to longitudinal data with arbitrary sampling times we must consider the reasons why observations were made at the given times. This is analogous to the problem of missing data, where the fact that a particular observation is missing may implicitly give information about the value of that observation. Possible observation schemes include:

- *fixed*. Each patient is observed at fixed intervals specified in advance.
- *random*. The sampling times vary randomly, independently of the current stage of the disease.
- *doctor's care*. More severely ill patients are monitored more closely. The next sampling time is chosen on the basis of the current disease state.
- *patient self-selection*. A patient may decide to visit the doctor on occasions when they are in a poor condition.

Grüger *et al.* [17] discussed conditions under which sampling times are *informative*. If a multi-state model is fitted, ignoring the information available in the sampling times, then biased inference may occur. Mathematically, because the sampling times are often themselves random, they should be modelled along with the observation process X_t . However the ideal situation is where the joint likelihood for the times and the process is proportional to the likelihood obtained if the sampling times were fixed in advance. Then the parameters of the process can be estimated independently of the parameters of the sampling scheme.

In particular, they showed that fixed, random and doctor's care observation policies are not informative, whereas patient self-selection is informative.

1.4 Likelihood for the multi-state model

Kay [6] described a general method for evaluating the likelihood for a general multi-state model in continuous time, applicable to any form of transition matrix. The only available information is the observed state at a set of times, as in figure 4. The sampling times are assumed to be non-informative.

The likelihood is calculated from the transition probability matrix $P(t)$. The (r, s) entry of $P(t)$ is the probability of being in state s at a time $t + u$ in the future, given the state at time u is r . It

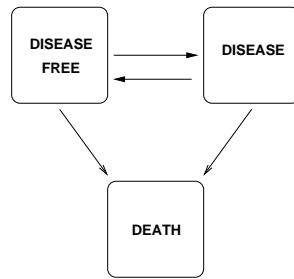


Figure 3: Illness-death model

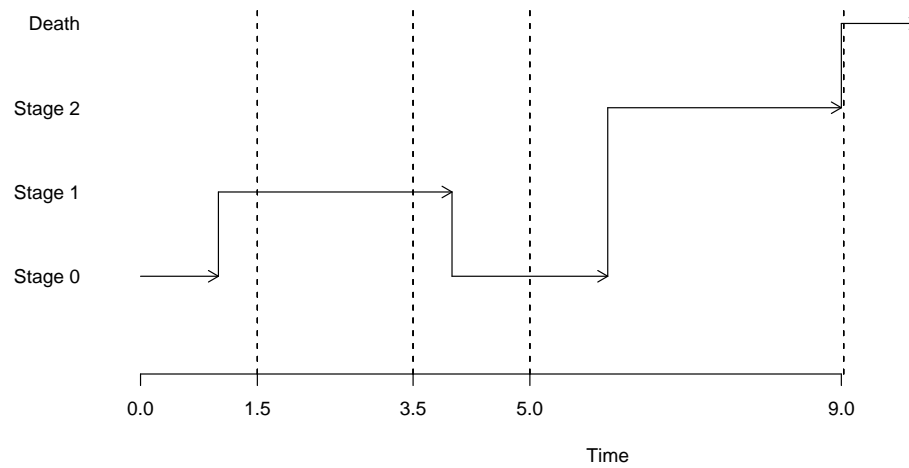


Figure 4: Evolution of a multi-state model. The process is observed on four occasions

does not say anything about the time of transition from r to s , indeed the process may have entered other states between times u and $t + u$. $P(t)$ can be calculated by taking the matrix exponential of the transition intensity matrix (see, for example, Cox and Miller [1]).

$$P(t) = \exp(tQ) \quad (2)$$

Suppose i indexes M individuals. The data for individual i consist of a series of times $(t_{i1}, \dots, t_{in_i})$ and corresponding states $(S(t_{i1}), \dots, S(t_{in_i}))$. Consider a general multi state model, with a pair of successive observed disease states $S(t_j), S(t_{j+1})$ at times t_j, t_{j+1} . The contribution to the likelihood from this pair of states is

$$L_{i,j} = p_{S(t_j)S(t_{j+1})}(t_{j+1} - t_j) \quad (3)$$

This is the entry of the transition matrix $P(t)$ at the $S(t_j)$ th row and $S(t_{j+1})$ th column, evaluated at $t = t_{j+1} - t_j$. Suppose also that the model has an absorbing death state, whose time of entry is observed to within 1 day. If $S(t_{j+1}) = D$ is death, then the contribution to the likelihood is summed over the unknown state on the day before death:

$$L_{i,j} = \sum_{m \neq D} p_{S(t_j),m}(t_{j+1} - t_j - 1)p_{m,D}(1) \quad (4)$$

assuming a time unit of days. The sum is taken over all possible states m which can be visited between $S(t_j)$ and D . If the times $(t_{i1}, \dots, t_{in_i})$ had been the *exact* transition times between the states, with no transitions between the observation times, then the contributions would be of the form

$$L_{i,j} = p_{S(t_j)S(t_j)}(t_{j+1} - t_j)q_{S(t_j)S(t_{j+1})} \quad (5)$$

since the state is assumed to be $S(t_j)$ throughout the interval between t_j and t_{j+1} with a known transition to state $S(t_{j+1})$ at t_{j+1} .

The full likelihood $L(Q)$ is the product of all such terms over all individuals and all transitions. It depends on the unknown transition matrix Q , which was used to determine $P(t)$.

1.5 Covariates

The relation of constant or time-varying characteristics of individuals to their transition rates is often of interest in a multi-state model. Explanatory variables for a particular transition intensity can be investigated by modelling the intensity as a function of these variables. Marshall and Jones [11] described a form of a *proportional hazards* model, where the transition intensity matrix elements q_{rs} which are of interest can be replaced by

$$q_{rs}(z(t)) = q_{rs}^{(0)} \exp(\beta_{rs}^T z(t))$$

The new Q is then used to determine the likelihood. If the covariates $z(t)$ are time dependent, the contributions to the likelihood of the form $p_{rs}(t - u)$ are replaced by

$$p_{rs}(t - u, z(u))$$

although this requires that the value of the covariate is known at every observation time u . Sometimes covariates are observed at different times to the main response, for example recurrent disease events

or other biological markers. In some of these cases it could be assumed that the covariate is a step function which remains constant between its observation times. Marshall and Jones [11] described likelihood ratio and Wald tests for covariate selection and testing hypotheses, for example whether the effect of a covariate is the same for all forward transitions in a disease progression model.

2 Fitting multi-state models with `msm`

`msm` is a package of functions for multi-state modelling using the R statistical software. The `msm` function itself implements maximum-likelihood estimation for general multi-state models with covariates. We illustrate its use with a set of data from monitoring heart transplant patients. Throughout this section “>” is used to indicate the R command prompt, and `typewriter` text indicates R commands or data.

2.1 Installing `msm`

The easiest way to install the `msm` package on a computer connected to the Internet is to run the R command:

```
> install.packages(msm)
```

This downloads `msm` from the CRAN archive of contributed R packages (cran.r-project.org or one of its mirrors) and installs it to the default library location, commonly

```
c:/Program Files/R/rw1061/library
```

on Windows, or

```
/usr/local/lib/R/library
```

on Unix-type systems. To install to a different location, for example if you are a normal user with no administrative privileges, create a directory in which R packages are to be stored, say, `/your/library/dir`, and run

```
> install.packages(msm, library='/your/library/dir')
```

After `msm` has been installed, its functions can be made visible in an R session by

```
> library(msm)
```

or, if it has been installed in a location outside the default library path,

```
> library(msm, lib.loc='/your/library/dir')
```

2.2 Getting the data in

The data can be specified as a series of observations, grouped by patient. It should contain variables indicating the subject identification number, the time of the observation, and the observed state of the process. The subject ID does not need to be numeric, but data must be grouped by subject, and observations must be ordered by time within subjects.

Table 1 is a sample of data taken from a set of heart transplant recipients. This data set is available in the `msm` package, and can be made available to the current R session by issuing the command

```
> data(heart)
```

There are 622 patients in all, but only three patient histories are shown in table 1. PTNUM is the subject identifier. Approximately each year after transplant, each patient has an angiogram, which detects coronary allograft vasculopathy (CAV), a deterioration of the arterial walls. The result of the test is in the variable `state`, with possible values 1, 2, 3 representing CAV-free, mild CAV and moderate or severe CAV respectively. A value of 4 is recorded at the date of death. `years` gives the time of the test in years since the heart transplant. Other variables include `age` (age at screen), `dage` (donor age), `sex` (0=male, 1=female), `pdiag` (primary diagnosis, or reason for transplant - IHD represents ischaemic heart disease, IDC represents idiopathic dilated cardiomyopathy), and `cumrej` (cumulative number of rejection episodes).

The data can also be specified as a list of *transitions*, rather than observations. Each row of the data file then represents a pair of observations. Columns indicate the starting state, the finishing state, and the time difference between the two observations. Additional variables may be given, representing the covariate value which is assumed to be constant within this time interval. Thus the first few rows of the above data set can be expressed as in table 1 (from-to format). We take the covariate value from the start of the interval. However, this form of data can not be used for multi-state models with misclassification (section 3), which do not assume successive transitions within the same individual are independent. It also discards potentially useful information about individual histories, therefore it is recommended to store data as rows representing observation times.

A useful way to summarise multi-state data is as a frequency table of pairs of consecutive states. This counts over all individuals, for each state r and s , the number of times an individual had an observation of state r followed by an observation of state s . The function `statetable.msm` can be used to produce such a table, as follows,

```
> statetable.msm(state, PTNUM, data=heart)
      to
from  1   2   3   4
  1 1367 204  44 148
  2   46 134  54  48
  3    4  13 107  55
```

Thus there were 148 CAV-free deaths, 48 deaths from state 2, and 55 deaths from state 3. On only four occasions was there an observation of severe CAV followed by an observation of no CAV.

2.3 Specifying a model

We now specify the multi-state model to be fitted to the data. A model is governed by a transition intensity matrix Q . For the heart transplant example, there are four possible states through which the patient can move, corresponding to CAV-free, mild/moderate CAV, severe CAV and death. We assume that the patient can advance or recover from consecutive states while alive, and die from any state. Thus the model is illustrated by figure 2 with four states, and we have

$$Q = \begin{pmatrix} -(q_{12} + q_{14}) & q_{12} & 0 & q_{14} \\ q_{21} & -(q_{21} + q_{23} + q_{24}) & q_{23} & q_{24} \\ 0 & q_{32} & -(q_{32} + q_{34}) & q_{34} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

To tell `msm` what the allowed transitions of our model are, we define a matrix, called `twoway4.q`, say, of the same size as Q , containing ones in the positions where the entries of Q

Observation time format

PTNUM	age	years	dage	sex	pdiag	cumrej	state
100002	52.4959	0.00000	21	0	IHD	0	1
100002	53.4986	1.00274	21	0	IHD	2	1
100002	54.4986	2.00274	21	0	IHD	2	2
100002	55.5890	3.09315	21	0	IHD	2	2
100002	56.4959	4.00000	21	0	IHD	3	2
100002	57.4932	4.99726	21	0	IHD	3	3
100002	58.3507	5.85479	21	0	IHD	3	4
100003	29.5068	0.00000	17	0	IHD	0	1
100003	30.6959	1.18904	17	0	IHD	1	1
100003	31.5151	2.00822	17	0	IHD	1	3
100003	32.4986	2.99178	17	0	IHD	2	4
100004	35.8959	0.00000	16	0	IDC	0	1
100004	36.8986	1.00274	16	0	IDC	2	1
100004	37.9068	2.01096	16	0	IDC	2	1
100004	38.9068	3.01096	16	0	IDC	2	1
100004	39.9041	4.00822	16	0	IDC	2	1
100004	40.8877	4.99178	16	0	IDC	2	1
100004	41.9178	6.02192	16	0	IDC	2	2
100004	42.9151	7.01918	16	0	IDC	2	3
100004	43.9123	8.01644	16	0	IDC	2	3
100004	44.7973	8.90137	16	0	IDC	2	4

From-to format

from	to	timelag	age	dage	sex	pdiag	cumrej
1	1	1.0027397	52.49589	21	1	1	0
1	2	1.0000000	53.49863	21	1	1	2
2	2	1.0904110	54.49863	21	1	1	2
2	2	0.9068493	55.58904	21	1	1	2
2	3	0.9972603	56.49589	21	1	1	3
3	4	0.8575342	57.49315	21	1	1	3
1	1	1.1890411	29.50685	17	1	1	0
1	3	0.8191781	30.69589	17	1	1	1
3	4	0.9835616	31.51507	17	1	1	1
1	1	1.0027397	35.89589	16	1	2	0
1	1	1.0082192	36.89863	16	1	2	2
1	1	1.0000000	37.90685	16	1	2	2
1	1	0.9972603	38.90685	16	1	2	2
1	1	0.9835616	39.90411	16	1	2	2
1	2	1.0301370	40.88767	16	1	2	2
2	3	0.9972603	41.91781	16	1	2	2
3	3	0.9972603	42.91507	16	1	2	2
3	4	0.8849315	43.91233	16	1	2	2

Table 1: Three individual histories from the heart transplant data

are non-zero. The diagonal entries of this matrix do not matter, as the diagonal entries of Q are defined as minus the sum of all the other entries in the row.

```
> twoway4.q <- rbind ( c(0, 1, 0, 1),
                        c(1, 0, 1, 1),
                        c(0, 1, 0, 1),
                        c(0, 0, 0, 0) )
```

Fitting the model is a process of finding values of the seven unknown transition intensities: q_{12} , q_{14} , q_{21} , q_{23} , q_{24} , q_{32} , q_{34} , which maximise the likelihood.

2.4 Specifying initial values

For reassurance that the true maximum likelihood estimates have been found, models should be run repeatedly starting from different initial values. However a sensible choice of initial values can be important for unstable models with flat or multi-modal likelihoods. For example, the transition rates for a model with misclassification could be initialised at the corresponding estimates for an approximating model without misclassification. Initial values for a model without misclassification could be set by supposing that transitions between stages take place only at the observation times. If we observe n_{rs} transitions from state r to state s , and a total of n_r transitions from state r , then q_{rs}/q_{rr} can be estimated by n_{rs}/n_r . Then, given a total of T_r years spent in state r , the mean sojourn time $1/q_{rr}$ can be estimated as T_r/n_r . Thus, n_{rs}/T_r is a crude estimate of q_{rs} . The `msm` package provides a function `crudeinits.msm` for calculating initial values in this way.

```
> crudeinits.msm(state, years, PTNUM, data=heart, qmatrix=twoway4.q)
      1-2      1-4      2-1      2-3      2-4      3-2      3-4
0.0679893 0.0493256 0.1168179 0.1371340 0.1218969 0.0490840 0.2076631
```

However, if there are many changes of state in between the observation times, then this crude approach may fail to give sensible initial values. For the heart transplant example we could also guess that the mean period in each state before moving to the next is about 2 years, and there is an equal probability of progression, recovery or death. This gives $q_{rr} = -0.5$ for $r = 1, 2, 3$, and $q_{12} = q_{14} = 0.25$, $q_{21} = q_{23} = q_{24} = 0.166$, $q_{32} = q_{34} = 0.25$.

```
> inits1 <- c(0.25, 0.25, 0.166, 0.166, 0.166, 0.25, 0.25)
```

2.5 Running msm

To fit a model, call the `msm` function with the appropriate arguments. For our running example, we have defined a data set `heart`, a matrix `twoway4.q` indicating the allowed transitions, and a vector of initial values `inits`. We are ready to run `msm`.

Model 1: simple bidirectional model

```
> heart.msm <- msm( state ~ years, subject=PTNUM, data = heart,
                    qmatrix = twoway4.q, inits = inits1, death = TRUE, tunit = 365,
                    control = list (trace=2, REPORT=1) )
```

We specify `death = TRUE` in the call to `msm`. The time of death, the final state in the model, is recorded to within one day. Then `msm` needs to know what a “day” is in this context, in terms of our observed time variable. Thus we specify `tunit = 365`: the unit of our observed time variable in days.

While the `msm` function runs, it searches for the maximum of the likelihood of the unknown parameters. Internally, it uses the R function `optim` to minimise the minus log-likelihood. When the data set, the model, or both are large, then this may take a long time. It is then useful to see the progress of the optimisation algorithm. To do this, we specify a `control` argument, which is passed internally to the `optim` function. See the help page for `optim`,

```
> ?optim
```

for more options to control the optimisation. When completed, the `msm` function returns a value. This value is a list of the important results of the model fitting, including the parameter estimates and their covariances. To keep these results for post-processing, we store them in an R object, here called `heart.msm`. When running several similar `msm` models, it is recommended to store the respective results in informatively-named objects.

2.6 Showing results

To show the maximum likelihood estimates and their standard errors, type the name of the fitted model object at the R command prompt.¹

```
> heart.msm

Multi-state Markov models in continuous time
-----

Maximum likelihood estimates:

* Matrix of transition intensities with covariates set to their means

      Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 -0.1673425  0.1270006  0.0000000  0.04034186
Stage 2  0.2022784 -0.6007947  0.3664707  0.03204556
Stage 3  0.0000000  0.1101204 -0.4303000  0.32017958
Stage 4  0.0000000  0.0000000  0.0000000  0.00000000

      corresponding standard errors

      Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.009226362 0.008827036 0.00000000 0.004448418
Stage 2 0.030413882 0.046488624 0.04055923 0.022987690
Stage 3 0.000000000 0.029423560 0.05206172 0.038305442
Stage 4 0.000000000 0.000000000 0.00000000 0.000000000

* No covariates on transition intensities

* Mean sojourn times in transient states with covariates set to their means
```

¹This is equivalent to typing `print.msm(heart.msm)`. The function `print.msm` formats the important information in the model object for printing on the screen.

```

$estimate
  Stage 1   Stage 2   Stage 3
5.975768 1.664462 2.323960

$SE
  Stage 1   Stage 2   Stage 3
0.3294716 0.1287937 0.2811745

-2 * log-likelihood: 6943.577

```

From the estimated intensity matrix, we see patients are three times as likely to develop symptoms than die without symptoms (first row). After disease onset (stage 2), progression to severe symptoms (stage 3) is twice as likely as recovery, and death from the severe disease stage is rapid (mean of 2.32 years in stage 3).

Section 2.9 describes various functions that can be used to obtain summary information from the fitted model.

2.7 Covariates on the transition rates

We now model the effect of explanatory variables on the rates of transition, using a proportional intensities model. Now we have an intensity matrix $Q(z)$ which depends on a covariate vector z . For each entry of $Q(z)$, the transition intensity for patient i at observation time j is $q_{rs}(z_{ij}) = q_{rs}^{(0)} \exp(\beta_{rs}^T z_{ij})$. The covariates z are specified through the `covariates` argument to `msm`. If z_{ij} is time-dependent, we assume it is constant in between the observation times of the Markov process. By default `msm` calculates the probability for a state transition from times $t_{i,j-1}$ to t_{ij} using the covariate value at time $t_{i,j-1}$, but this can be changed to $t_{i,j}$ by specifying an argument `covmatch="next"` to `msm`.

We consider a model with just one covariate, female sex. Out of the 622 transplant recipients, 535 are male and 87 are female. We give an extra seven initial values of zero, one for the linear effect of sex on each intensity on the log scale.

Model 2: sex as a covariate

```

> inits2 <- c(0.25, 0.25, 0.166, 0.166, 0.166, 0.25, 0.25, 0, 0, 0,
0, 0, 0, 0)
> heartsex.msm <- msm( state ~ years, subject=PTNUM, data = heart,
  qmatrix = twoway4.q, inits = inits2,
  death = TRUE, tunit = 365, covariates = ~ sex)

```

The `msm` object will now include the estimated covariate effects and their standard errors.

```

> heartsex.msm

Multi-state Markov models in continuous time
-----

Maximum likelihood estimates:

* Matrix of transition intensities with covariates set to their means

      Stage 1      Stage 2      Stage 3      Stage 4

```

```

Stage 1 -0.1634103  0.1254215  0.0000000  0.03798887
Stage 2  0.1923463 -0.6263536  0.3555461  0.07846121
Stage 3  0.0000000  0.1526604 -0.4758856  0.32322519
Stage 4  0.0000000  0.0000000  0.0000000  0.00000000

```

corresponding standard errors

```

          Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.009108762 0.008807001 0.00000000 0.004458974
Stage 2 0.029461342 0.049254737 0.03989392 0.021684196
Stage 3 0.000000000 0.038576713 0.05623371 0.037557589
Stage 4 0.000000000 0.000000000 0.00000000 0.000000000

```

* Linear effects on log transition intensities of sex

```

          Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.0000000 -0.5920732 0.0000000 0.2645051
Stage 2 0.237829  0.0000000 0.449743 0.4741817
Stage 3 0.0000000 0.5957510 0.0000000 0.5162959
Stage 4 0.0000000 0.0000000 0.0000000 0.0000000

```

corresponding standard errors

```

          Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.0000000 0.2610496 0.0000000 0.3025250
Stage 2 0.5210001 0.0000000 0.4160991 0.7276354
Stage 3 0.0000000 1.1926230 0.0000000 0.4157256
Stage 4 0.0000000 0.0000000 0.0000000 0.0000000

```

* Mean sojourn times in transient states with covariates set to their means

\$estimate

```

Stage 1 Stage 2 Stage 3
6.119565 1.596542 2.101345

```

\$SE

```

Stage 1 Stage 2 Stage 3
0.3411146 0.1255477 0.2483085

```

-2 * log-likelihood: 6938.592

Comparing the estimated log-linear effects of age and their standard errors, we see that the disease onset rate is smaller for females, whereas none of the other effects are large with respect to their standard errors. This output shows the estimated transition intensity matrix $q_{rs}(z) = q_{rs}^{(0)} \exp(\beta_{rs}^T z)$ with the covariate z set to its mean value in the data. Here, this represents an average intensity matrix for the population of 535 male and 87 female patients. To extract separate matrices for male and female patients ($z = 0$ and 1 respectively), use the function `qmatrix.msm`, as shown below. This and similar summary functions will be described in more detail in section 2.9.

```
> qmatrix.msm(heartsex.msm, covariates=list(sex=0)) # Male
```

\$estimates

```

          Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 -0.1714705  0.1346701  0.0000000  0.03680037
Stage 2  0.1869269 -0.5978829  0.3368406  0.07411535
Stage 3  0.0000000  0.1421134 -0.4458943  0.30378089

```

```

Stage 4  0.0000000  0.0000000  0.0000000  0.0000000

$SE
      Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.00996882 0.009706826 0.00000000 0.004712185
Stage 2 0.02998003 0.048085897 0.03882023 0.022293741
Stage 3 0.00000000 0.033936933 0.05334545 0.036564563
Stage 4 0.00000000 0.000000000 0.00000000 0.000000000

> qmatrix.msm(heartsex.msm, covariates=list(sex=1)) # Female
$estimates
      Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 -0.1224397  0.07449672  0.00000000 0.04794301
Stage 2  0.2371153 -0.88433190  0.5281355  0.11908104
Stage 3  0.00000000  0.25784953 -0.7669282  0.50907867
Stage 4  0.00000000  0.00000000  0.00000000  0.00000000

$SE
      Stage 1      Stage 2      Stage 3      Stage 4
Stage 1 0.02071834 0.01869132 0.00000000 0.01314067
Stage 2 0.11753803 0.25563614 0.2111579  0.07889733
Stage 3 0.00000000 0.30128517 0.3495075  0.20257294
Stage 4 0.00000000 0.00000000 0.00000000 0.00000000

```

We may also want to constrain the effect of age to be equal for certain transition rates, to reduce the number of parameters in the model, or to investigate hypotheses on the covariate effects. A `constraints` argument can be used to indicate which of the transition rates have common covariate effects.

Model 3: constrained covariate effects

```

> inits3 <- c(0.25, 0.25, 0.166, 0.166, 0.166, 0.25, 0.25, 0, 0, 0)
> heart3.msm <- msm( state ~ years, subject=PTNUM, data = heart,
+                   qmatrix = twoway4.q, inits = inits3, death = TRUE,
+                   tunit = 365, covariates = ~ sex,
+                   constraint = list(sex=c(1,2,3,1,2,3,2)) )

```

This constrains the effect of age to be equal for the progression rates q_{12}, q_{23} , equal for the death rates q_{14}, q_{24}, q_{34} , and equal for the recovery rates q_{21}, q_{32} . The intensity parameters are assumed to be ordered by reading across the rows of the transition matrix, starting at the first row: $(q_{12}, q_{14}, q_{21}, q_{23}, q_{24}, q_{32}, q_{34})$, giving constraint indicators $(1, 2, 3, 1, 2, 3, 2)$. Any vector of increasing numbers can be used for the indicators. Notice we have four fewer parameters in the model, therefore we give four fewer initial values.

2.8 Fixing parameters at their initial values

For exploratory purposes we may want to fit a model assuming that some parameters are fixed, and estimate the remaining parameters. This may be necessary in cases where there is not enough information in the data to be able to estimate a proposed model, and we have strong prior information about a certain transition rate. To do this, use the `fixedpars` argument to `msm`. For model 1, the following statement fixes the parameters numbered 2, 5, 7 (q_{14}, q_{24}, q_{34}) to their initial values (0.25, 0.166 and 0.25, respectively).

Model 4: fixed parameters

```
> inits4 <- c(0.25, 0.25, 0.166, 0.166, 0.166, 0.25, 0.25)
> heart4.msm <- msm( state ~ years, subject=PTNUM, data = heart,
  qmatrix = twoway4.q, inits = inits4, death = TRUE,
  tunit = 365, control = list (trace=2, REPORT=1),
  fixedpars = c(2, 5, 7) )
```

A `fixedpars` statement can also be useful for fixing covariate effect parameters to zero, that is to assume no effect of a covariate on a certain transition rate.

2.9 Extractor functions

We may want to extract some of the information from the `msm` model fit for post-processing, for example for plotting graphs or generating summary tables. A set of functions is provided for extracting interesting features of the fitted model.

Intensity matrices The function `qmatrix.msm` extracts a transition intensity matrix and the corresponding standard errors for a given set of covariate values, as shown in section 2.7.

Transition probability matrices The function `pmatrix.msm` extracts the estimated transition probability matrix $P(t)$ within a given time. For example, for model 1, the 10 year transition probabilities are given by:

```
> pmatrix.msm(heart.msm, t=10)

           Stage 1   Stage 2   Stage 3   Stage 4
Stage 1 0.30198009 0.09506924 0.09394938 0.5090013
Stage 2 0.15142015 0.05897163 0.07980591 0.7098023
Stage 3 0.04496411 0.02398079 0.04424346 0.8868116
Stage 4 0.00000000 0.00000000 0.00000000 1.0000000
```

Thus, a person in stage 1 has a probability of 0.5 of being dead ten years from now, a probability of 0.3 of being still disease-free, and probabilities of 0.1 of being alive with mild/moderate or severe disease, respectively.

This assumes Q is constant within the desired time interval. For non-homogeneous processes, where Q varies with time-dependent covariates but is piecewise constant, there is an equivalent function `pmatrix.piecewise.msm`.

Mean sojourn times The function `sojourn.msm` extracts the estimated mean sojourn times in each transient state, for a given set of covariate values.

```
> sojourn.msm(heart.msm)
$estimate
  Stage 1 Stage 2 Stage 3
5.975768 1.664462 2.323960

$SE
  Stage 1 Stage 2 Stage 3
0.3294716 0.1287937 0.2811745
```

Total length of stay Mean sojourn times describe the average period in a single stay in a state. For processes with successive periods of recovery and relapse, we may want to forecast the total time spent healthy or diseased, before death. The function `totlos.msm` estimates the forecasted total length of time spent in each transient state s between two future time points t_1 and t_2 , for a given set of covariate values. This defaults to the expected amount of time spent in each state between the start of the process (time 0, the present time) and death or a specified future time. This is obtained as

$$L_s = \int_{t_1}^{t_2} P(t)_{r,s} dt$$

where r is the state at the start of the process, which defaults to 1. This is calculated using numerical integration. For model 1, each patient is forecasted to spend 8.7 years disease free, 2.2 years with mild or moderate disease and 1.8 years with severe disease.

```
> totlos.msm(heart.msm)
  Stage 1  Stage 2  Stage 3
8.570903 2.146924 1.828457
```

Ratio of transition intensities The function `gratio.msm` estimates a ratio of two entries of the transition intensity matrix at a given set of covariate values. For example, we may want to estimate the ratio of the progression rate q_{12} into the first stage of disease to the corresponding recovery rate q_{21} . For example in model 1, recovery is 1.6 times as likely as progression.

```
> gratio.msm(heart.msm, ind1=c(2,1), ind2=c(1,2))
$estimate
[1] 1.592735

$se
[1] 0.2218395
```

Hazard ratios for transition The function `hazard.msm` gives the estimated hazard ratios corresponding to each covariate effect on the transition intensities. 95% confidence limits are computed by assuming normality of the log-effect. For example, for model 2 with female sex as a covariate, the following hazard ratios show more clearly that the only transition on which the effect of sex is significant at the 5% level is the 1-2 transition.

```
> hazard.msm(heartsex.msm)
$sex
              HR          L95          U95
Stage 2 - Stage 1 1.2684923 0.4568901 3.5217934
Stage 1 - Stage 2 0.5531792 0.3316347 0.9227239
Stage 3 - Stage 2 1.8143930 0.1752135 18.7886360
Stage 2 - Stage 3 1.5679092 0.6936433 3.5440971
Stage 1 - Stage 4 1.3027860 0.7200512 2.3571260
Stage 2 - Stage 4 1.6066988 0.3859848 6.6880381
Stage 3 - Stage 4 1.6758087 0.7419215 3.7852186
```

All of these extractor functions take an argument called `covariates`. If this argument is omitted, for example,

```
> qmatrix.msm(heart.msm)
```

then the intensity matrix is evaluated as $Q(\bar{x})$ with all covariates set to their mean values \bar{x} in the data. Alternatively, set `covariates` to 0 to return the result $Q(0)$ with covariates set to zero. This will usually be preferable for categorical covariates, where we wish to see the result for the baseline category.

```
> qmatrix.msm(heartsex.msm, covariates = 0)
```

Alternatively, the desired covariate values can be specified explicitly as a list,

```
> qmatrix.msm(heartsex.msm, covariates = list(sex = 1))
```

If a covariate is categorical, that is, an *Rfactor* with k levels, then we use its internal representation as a set of $k - 1$ 0/1 indicator functions. For example, consider a covariate `cov`, with three levels, `VAL1`, `VAL2`, `VAL3`, where the baseline level is `VAL1`. To set the value of `cov` to be `VAL1`, `VAL2` or `VAL3`, respectively, use statements such as

```
> qmatrix.msm(example.msm, covariates = list(age = 60, covVAL2=0, covVAL3=0))
> qmatrix.msm(example.msm, covariates = list(age = 60, covVAL2=1, covVAL3=0))
> qmatrix.msm(example.msm, covariates = list(age = 60, covVAL2=0, covVAL3=1))
```

respectively. (This procedure is likely to be simplified in future versions of the package.)

2.10 Convergence failure

Inevitably if over-complex models are applied with insufficient data then the parameters of the model will not be identifiable. This will result in the optimisation algorithm failing to find the maximum of the log-likelihood, or even failing to evaluate the likelihood. For example, it will commonly be inadvisable to include several covariates in a model simultaneously.

Initial values Make sure that a sensible set of initial values have been chosen. The optimisation may only converge within a limited range of ‘informative’ initial values.

Convergence criteria Sometimes the optimisation may report convergence, but fail to calculate any standard errors. In these cases, the Hessian of the log-likelihood at the converged solution is not positive definite. Thus the reported solution is probably close to the maximum, but not the maximum. This type of problem can often be solved by tightening the criteria (`reltol`, defaults to $1e-08$) for reporting convergence of the optimisation.

```
options = list(reltol = 1e-16)
```

Alternatively consider using smaller step sizes for the numerical approximation to the gradient, used in calculating the Hessian. This is given by the control parameter `ndeps`. For example, for a model with 5 parameters,

```
options = list(ndeps = rep(1e-6, 5))
```

Scaling Consider applying a scaling factor to normalise the likelihood (`fnscale`), or certain individual parameters (`parscale`). This may prevent overflow or underflow problems within the optimisation. For example, if the value of the $-2 \times \log$ -likelihood is around 5000, then the following option leads to a minimisation of the $-2 \times \log$ -likelihood on an approximate unit scale.

```
options = list(fnscale = 5000)
```

Model simplification If none of these numerical adjustments lead to convergence, then the model is probably over-complicated. There may not be enough information in the data on a certain transition rate. It is always recommended to count all the pairs of transitions between states in successive observation times, making a frequency table of previous state against current state (function `statetable.msm`). Although the data are a series of snapshots of a continuous-time process, and the actual transitions take place in between the observation times, this type of table may still be helpful. If there are not many observed ‘transitions’ from state 2 to state 4, for example, then the data may be insufficient to estimate q_4 .

For a staged disease model (figure 2), the number of disease states should be low enough that all transition rates can be estimated. Consecutive stages of disease severity should be merged if necessary. For models with covariates, consider applying constraints on the covariate effects so that the effects are equal for certain transitions, or zero for certain transitions.

2.11 Model assessment

Observed and expected prevalence To compare the relative fit of two nested models, it is easy to compare their likelihoods. However it is not always easy to determine how well a fitted multi-state model describes an irregularly-observed process. Ideally we would like to compare observed data with fitted or expected data under the model. If there were times at which all individuals were observed then the fit of the expected numbers in each state or *prevalences* can be assessed directly at those times. Otherwise, some approximations are necessary. We could assume that an individual’s state at an arbitrary time t was the same as the state at their previous observation time. This might be fairly accurate if observation times are close together. This approach is taken by the function `prevalence.msm`, which constructs a table of observed and expected numbers and percentages of individuals in each state at a set of times.

A set of expected counts can be produced if the process begins at a common time for all individuals. Suppose at this time, each individual is in state 0. Then given $n(t)$ individuals are under observation at time t , the expected number of individuals in state r at time t is $n(t)P(t)_{0,r}$.

For example, we calculate the observed expected numbers and percentages at two-yearly intervals up to 20 years after transplant, for the heart transplant model 1. The number of individuals still alive and under observation decreases from 622 to 251 at year 20.

```
> options(digits=3)
> prevalence.msm(heart.msm, times=seq(0,20,2))
Calculating approximate observed state prevalences...
Forecasting expected state prevalences...
$Observed
  Stage 1 Stage 2 Stage 3 Stage 4 Total
0      622      0      0      0    622
```

2	507	20	7	54	588
4	330	37	24	90	481
6	195	43	28	129	395
8	117	44	21	161	343
10	60	25	22	189	296
12	26	11	12	221	270
14	11	3	6	238	258
16	4	0	3	245	252
18	0	0	2	249	251
20	0	0	0	251	251

\$Expected					
	Stage 1	Stage 2	Stage 3	Stage 4	Total
0	622.0	0.00	0.0	0.0	622
2	437.6	74.37	25.5	50.5	588
4	279.1	68.31	41.8	91.8	481
6	182.6	51.47	41.0	120.0	395
8	127.7	38.64	35.3	141.3	343
10	89.4	28.14	27.8	150.7	296
12	66.4	21.40	22.1	160.1	270
14	51.8	16.93	18.0	171.3	258
16	41.3	13.64	14.7	182.3	252
18	33.7	11.17	12.2	194.0	251
20	27.6	9.18	10.1	204.2	251

\$"Observed percentages"				
	Stage 1	Stage 2	Stage 3	Stage 4
0	100.00	0.00	0.000	0.00
2	86.22	3.40	1.190	9.18
4	68.61	7.69	4.990	18.71
6	49.37	10.89	7.089	32.66
8	34.11	12.83	6.122	46.94
10	20.27	8.45	7.432	63.85
12	9.63	4.07	4.444	81.85
14	4.26	1.16	2.326	92.25
16	1.59	0.00	1.190	97.22
18	0.00	0.00	0.797	99.20
20	0.00	0.00	0.000	100.00

\$"Expected percentages"				
	Stage 1	Stage 2	Stage 3	Stage 4
0	100.0	0.00	0.00	0.00
2	74.4	12.65	4.34	8.59
4	58.0	14.20	8.68	19.09
6	46.2	13.03	10.37	30.38
8	37.2	11.27	10.30	41.19
10	30.2	9.51	9.39	50.90
12	24.6	7.93	8.19	59.30
14	20.1	6.56	6.97	66.41
16	16.4	5.41	5.84	72.35
18	13.4	4.45	4.85	77.28
20	11.0	3.66	4.01	81.35

Comparing the observed and expected percentages in stages 1, 2 and 3, we see that the predicted number of individuals who die is under-estimated by the model from year 8 onwards. Similarly the number of individuals still alive and free of CAV (Stage 1) is over-estimated by the model for year 10

onwards.

These discrepancies could have many causes. One possibility is that the transition rates vary with the time since the beginning of the process, or the age of the patient, so that the Markov model is *non-homogeneous*. This could be accounted for by modelling the intensity as a function of age, such as a piecewise-constant function. Another cause of poor fit may be the failure of the Markov assumption. That is, the transition intensities may depend on the time spent in the current state (a semi-Markov process) or other characteristics of the process history. Accounting for the process history is difficult as the process is only observed through a series of snapshots. For a multi-state model with one-way progression through states, and frequent observations, we may be able to estimate the time spent in each state by each individual.

3 Multi-state models with misclassification

Hidden Markov models In a *hidden Markov model* (HMM) the states of the Markov chain are not observed. The observed data are governed by some probability distribution conditionally on the unobserved state. The evolution of the underlying Markov chain is still governed by a transition intensity matrix Q (figure 5). This class of model is commonly used in areas such as speech and signal processing [18] and the analysis of biological sequence data [19]. In engineering and biological sequencing applications, the Markov process usually evolves over an equally-spaced, discrete ‘time’ space. Therefore most of the theory of HMM estimation was developed for discrete-time models. HMMs have less frequently been used in medicine, where continuous time processes are often more suitable. A disease process evolves in continuous time, and patients are often monitored at irregular and differing intervals.

The `msm` package can fit continuous-time multi-state models with misclassification, which are a special case of HMM. In these models the observed data are states, assumed to be misclassifications of the true, underlying states.

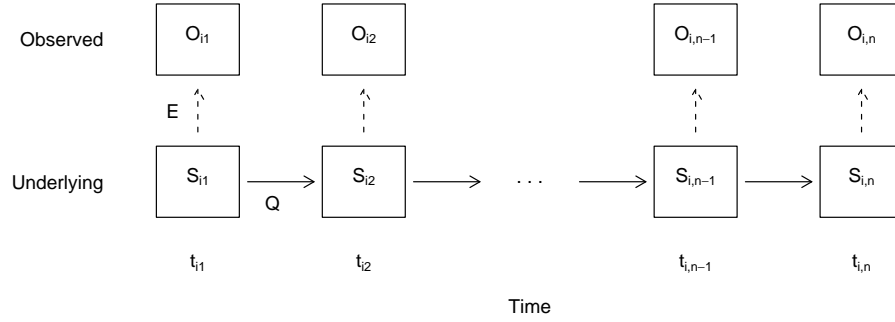


Figure 5: A hidden Markov model in continuous time. Observed states are generated conditionally on an underlying Markov process.

Screening tests with misclassification Consider a Markov model with at least a disease-free and a disease state. When screening for the presence of the disease, the screening process can sometimes be subject to error. Then the Markov disease process $S_i(t)$ for individual i is not observed directly, but through realisations $O_i(t)$. The quality of a diagnostic test is often measured by the probabilities that the true and observed states are equal, $Pr(O_i(t) = r | S_i(t) = r)$. Where r represents a ‘positive’

disease stage, this is the *sensitivity*, or the probability that a true positive is detected by the test. Where r represents a ‘negative’ or disease-free stage, this represents the *specificity*, or the probability that, given the condition of interest is absent, the test produces a negative result.

General misclassification model As an extension to the simple multi-state model described in section 1, we describe a general multi-state model with misclassification. For patient i , observation time t_{ij} , observed states O_{ij} are generated conditionally on true states S_{ij} according to a *misclassification matrix* E . This is a $n \times n$ matrix, whose (r, s) entry is

$$e_{rs} = \Pr(O(t_{ij}) = s | S(t_{ij}) = r), \quad (6)$$

which we first assume to be independent of time t . Analogously to the entries of Q , some of the e_{rs} may be fixed to reflect knowledge of the diagnosis process. For example, the probability of misclassification may be negligibly small for non-adjacent states of disease. Thus the progression through underlying states is governed by the transition intensity matrix Q , while the observation process of the underlying states is governed by the misclassification matrix E .

To investigate explanatory variables $w(t)$ for the probability of misclassification e_{rs} for each pair of stages r and s , a logistic model can be used,

$$\log \frac{e_{rs}(t)}{1 - e_{rs}(t)} = \gamma_{rs}^T w(t). \quad (7)$$

3.1 Maximum likelihood estimation

A type of EM algorithm known as the *Baum-Welch* or *forward-backward* algorithm [20, 21], is commonly used for hidden Markov model estimation in discrete-time applications. See, for example, Durbin *et al.* [19], Albert [22]. A generalisation of this algorithm to continuous time was described by Bureau *et al.* [23].

The `msm` function uses a direct method of calculating likelihoods in discrete or continuous time based on matrix products. This type of method has been described by Macdonald and Zucchini [24, pp. 77–79], Lindsey [25, p.73] and Guttorp [26]. Satten and Longini [8] used this method to calculate likelihoods for a hidden Markov model in continuous time with observations of a continuous marker generated conditionally on underlying discrete states. The matrix-product method is now illustrated for the misclassification model. However it can be generalised to any form of data generated conditionally on states of a hidden Markov process.

Patient i ’s contribution to the likelihood is

$$\begin{aligned} L_i &= \Pr(O_{i1}, \dots, O_{im_i}) \\ &= \sum \Pr(O_{i1}, \dots, O_{im_i} | S_{i1}, \dots, S_{im_i}) \Pr(S_{i1}, \dots, S_{im_i}) \end{aligned} \quad (8)$$

where the sum is taken over all possible paths of underlying states S_{i1}, \dots, S_{im_i} . Assume that the observed states are conditionally independent given the values of the underlying states. Also assume the Markov property, $\Pr(S_{ij} | S_{i,j-1}, \dots, S_{i1}) = \Pr(S_{ij} | S_{i,j-1})$. Then the contribution L_i can be written as a product of matrices, as follows. To derive this matrix product, decompose the overall sum in equation 9 into sums over each underlying stage. The sum is accumulated over the unknown

first state, the unknown second state, and so on until the unknown final state:

$$L_i = \sum_{S_{i1}} Pr(O_{i1}|S_{i1})Pr(S_{i1}) \sum_{S_{i2}} Pr(O_{i2}|S_{i2})Pr(S_{i2}|S_{i1}) \sum_{S_{i3}} Pr(O_{i3}|S_{i3})Pr(S_{i3}|S_{i2}) \dots \sum_{S_{im_i}} Pr(O_{im_i}|S_{im_i})Pr(S_{im_i}|S_{im_i-1}) \quad (9)$$

where $Pr(O_{ij}|S_{ij})$ is the misclassification probability $e_{S_{ij}O_{ij}}$. $Pr(S_{i,j+1}|S_{ij})$ is the $(S_{ij}, S_{i,j+1})$ entry of the Markov chain transition matrix $P(t)$ evaluated at $t = t_{i,j+1} - t_{ij}$. Let f be the vector of initial stage occupation probabilities $Pr(S_{i1})$, and let $\mathbf{1}$ be a column vector consisting of ones. For $j = 2, \dots, m_i$ let T_{ij} be the $n \times n$ matrix with (r, s) entry

$$e_{sO_{ij}}p_{rs}(t_{ij} - t_{i,j-1})$$

Then subject i 's likelihood contribution is

$$L_i = fT_{i2}T_{i3} \dots T_{im_i}\mathbf{1} \quad (10)$$

If $S(t_j) = D$ is an absorbing state such as death, measured without error, whose entry time is known to within one time unit, then the contribution to the likelihood is summed over the unknown state on the day before entry. Assuming a time unit of days, the (r, s) entry of T_{ij} is then

$$p_{rs}(t_j - 1 - t_{j-1})p_{s,D}(1)$$

3.2 Fitting misclassification models with msm

In fact, in the heart transplant example from section 2.2, it is not realistic for patients to recover from a diseased state to a healthy state. Progression of coronary artery vasculopathy is thought to be an irreversible process. The angiography scan for CAV is also subject to error, which leads to some false measurements of CAV states and apparent recoveries. Thus we replace the two-way multi-state model by a one-way model with transition intensity matrix

$$Q = \begin{pmatrix} -(q_{12} + q_{14}) & q_{12} & 0 & q_{14} \\ 0 & -(q_{23} + q_{24}) & q_{23} & q_{24} \\ 0 & 0 & -q_{34} & q_{34} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

We also assume that true state 1 (CAV-free) can be classified as state 1 or 2, state 2 (mild/moderate CAV) can be classified as state 1, 2 or 3, while state 3 (severe CAV) can be classified as state 2 or 3. Recall that state 4 represents death. Thus our misclassification probability matrix is

$$E = \begin{pmatrix} 1 - e_{12} & e_{12} & 0 & 0 \\ e_{21} & 1 - e_{21} - e_{23} & e_{23} & 0 \\ 0 & e_{32} & 1 - e_{32} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

with underlying states as rows, and observed states as columns.

To model observed states with misclassification, we define an indicator matrix `ematrix` for the states that can be misclassified. We then call `msm` as before, but set `misc=TRUE` and specify

the `ematrix`. We also need initial values for five unknown transition intensities (given in the order $q_{12}, q_{14}, q_{23}, q_{24}, q_{34}$) and four unknown misclassification probabilities (given in the order $e_{12}, e_{21}, e_{23}, e_{32}$). The indicator matrix `qmatrix` for the permitted transition intensities also changes to correspond to the new Q representing the progression-only model for the underlying states.

We also use an alternative quasi-Newton optimisation algorithm (`method="BFGS"`) which can often be faster than the default Nelder-Mead simplex-based algorithm. An optional argument `initprobs` could also have been given here, representing a vector f of the probabilities of occupying each true state at the start of the process. If not given, all individuals are assumed to be in true state 1 at the initial time.

Model 5: multi-state model with misclassification

```
> oneway4.q <- rbind(c(0, 1, 0, 1),
                    c(0, 0, 1, 1),
                    c(0, 0, 0, 1),
                    c(0, 0, 0, 0))

> ematrix <- rbind(c(0, 1, 0, 0),
                  c(1, 0, 1, 0),
                  c(0, 1, 0, 0),
                  c(0, 0, 0, 0))

> miscinits <- c(0.148, 0.0171, 0.202, 0.081, 0.126,
                0.1, 0.1, 0.1, 0.1)

> heartmisc.msm <- msm(state ~ years, misc=TRUE, subject = PTNUM,
                      data = heart, inits = miscinits,
                      qmatrix = oneway4.q, ematrix = ematrix, tunit=365,
                      method="BFGS")

> heartmisc.msm
```

Multi-state Markov models in continuous time

Maximum likelihood estimates:

* Matrix of transition intensities with covariates set to their means

	Stage 1	Stage 2	Stage 3	Stage 4
Stage 1	-0.145325	0.0985626	0.000000	0.0467619
Stage 2	0.000000	-0.2633697	0.201217	0.0621530
Stage 3	0.000000	0.000000	-0.367096	0.3670965
Stage 4	0.000000	0.000000	0.000000	0.000000

corresponding standard errors

	Stage 1	Stage 2	Stage 3	Stage 4
Stage 1	0.00871773	0.0078822	0.000000	0.0047952
Stage 2	0.00000000	0.0261764	0.0285534	0.0205069
Stage 3	0.00000000	0.000000	0.0488109	0.0488109
Stage 4	0.00000000	0.000000	0.000000	0.000000

* No covariates on transition intensities

* Matrix of misclassification probabilities with covariates set to their means

	Stage 1	Stage 2	Stage 3	Stage 4
Stage 1	0.991914	0.00808636	0.0000000	0
Stage 2	0.237922	0.71084776	0.0512306	0
Stage 3	0.000000	0.11272389	0.8872761	0
Stage 4	0.000000	0.00000000	0.0000000	1

corresponding standard errors

	Stage 1	Stage 2	Stage 3	Stage 4
Stage 1	0.00339671	0.00339671	0.0000000	0
Stage 2	0.03737863	0.03540776	0.0139780	0
Stage 3	0.00000000	0.03844006	0.0384401	0
Stage 4	0.00000000	0.00000000	0.0000000	0

* No covariates on misclassification probabilities

* Mean sojourn times in transient states with covariates set to their means

\$estimate

Stage 1	Stage 2	Stage 3
6.88115	3.79694	2.72408

\$SE

Stage 1	Stage 2	Stage 3
0.412786	0.377379	0.362207

-2 * log-likelihood: 3973.99

Thus there is an estimated probability of 0.01 that mild/moderate CAV will be diagnosed erroneously, but a rather higher probability of 0.24 that underlying mild/moderate CAV will be diagnosed as CAV-free. Between the two CAV stages, the mild stage will be misdiagnosed as severe with a probability of 0.05, and the severe state will be misdiagnosed as mild with a probability of 0.11.

The model also estimates the progression rates through underlying stages. An average of 6.9 years is spent disease-free, an average of 3.8 years is spent with mild/moderate disease. Spells of severe disease last 2.7 years on average before death.

3.3 Effects of covariates on misclassification rates

We can investigate how the probabilities of misclassification depend on covariates in a similar way to the transition intensities, using a `misccovariates` argument to `msm`. For example, we now include female sex as a covariate for the misclassification probabilities. This requires an extra four initial values for the linear effect for each of the logit-probabilities, which we set to zero.

Model 6: misclassification model with misclassification probabilities modelled on sex

```
> miscinits <- c(0.148, 0.0171, 0.202, 0.081, 0.126,
  0.1, 0.1, 0.1, 0.1, 0, 0, 0, 0)

> heartmiscsex.msm <- msm(state ~ years, misc=TRUE, subject = PTNUM,
  data = heart, inits = miscinits,
```

```

qmatrix = oneway4.q, ematrix = ematrix, tunit=365,
misccovariates = ~sex,
control = list(trace=1, REPORT=1), method="BFGS")

> heartmiscsex.msm

Multi-state Markov models in continuous time
-----

Maximum likelihood estimates:

* Matrix of transition intensities with covariates set to their means

      Stage 1   Stage 2   Stage 3   Stage 4
Stage 1 -0.144477  0.0977739  0.000000  0.0467036
Stage 2  0.000000 -0.2706537  0.207638  0.0630159
Stage 3  0.000000  0.0000000 -0.360329  0.3603287
Stage 4  0.000000  0.0000000  0.000000  0.0000000

      corresponding standard errors

      Stage 1   Stage 2   Stage 3   Stage 4
Stage 1 0.00886372 0.00808819 0.0000000 0.00480154
Stage 2 0.00000000 0.02741103 0.0298579 0.02116196
Stage 3 0.00000000 0.00000000 0.0485460 0.04854599
Stage 4 0.00000000 0.00000000 0.0000000 0.00000000

* No covariates on transition intensities

* Matrix of misclassification probabilities with covariates set to their means

      Stage 1   Stage 2   Stage 3 Stage 4
Stage 1 0.994768 0.00523219 0.0000000 0
Stage 2 0.237247 0.71060880 0.0521446 0
Stage 3 0.000000 0.13138136 0.8686186 0
Stage 4 0.000000 0.00000000 0.0000000 1

      corresponding standard errors

      Stage 1   Stage 2   Stage 3 Stage 4
Stage 1 0.0102221 0.0102221 0.0000000 0
Stage 2 0.0419666 0.0396737 0.0144309 0
Stage 3 0.0000000 0.0418224 0.0418224 0
Stage 4 0.0000000 0.0000000 0.0000000 0

* Linear effects on logit misclassification probabilities of sex

      Stage 1   Stage 2 Stage 3 Stage 4
Stage 1 0.00000 -6.25291  0.0000  0
Stage 2 1.08258  0.00000 -0.5951  0
Stage 3 0.00000  1.72107  0.0000  0
Stage 4 0.00000  0.00000  0.0000  0

      corresponding standard errors

      Stage 1   Stage 2 Stage 3 Stage 4
Stage 1 0.000000 16.08321 0.00000  0
Stage 2 0.747098  0.00000 1.07230  0

```

```

Stage 3 0.000000 1.04111 0.00000 0
Stage 4 0.000000 0.00000 0.00000 0

* Mean sojourn times in transient states with covariates set to their means

$estimate
Stage 1 Stage 2 Stage 3
6.92149 3.69476 2.77524

$SE
Stage 1 Stage 2 Stage 3
0.424635 0.374195 0.373900

-2 * log-likelihood: 3968.2

```

Therefore, considering the large standard errors relative to their estimates, we do not see any significant effect of sex on the fitted misclassification probabilities.

3.4 Extractor functions

Misclassification matrix The function `ematrix.msm` gives the estimated misclassification probability matrix at the given covariate values. For example, the fitted misclassification probabilities for men and women in model 6 are given by

```

> ematrix.msm(heartmiscsex.msm, covariates=list(sex=0))
$estimates
      Stage 1      Stage 2      Stage 3 Stage 4
Stage 1 0.988973 0.0110275 0.0000000 0
Stage 2 0.214514 0.7296914 0.0557945 0
Stage 3 0.000000 0.1095234 0.8904766 0
Stage 4 0.000000 0.0000000 0.0000000 1

$SE
      Stage 1      Stage 2      Stage 3 Stage 4
Stage 1 0.00433906 0.00433906 0.0000000 0
Stage 2 0.03947125 0.03716733 0.0157755 0
Stage 3 0.00000000 0.03760249 0.0376025 0
Stage 4 0.00000000 0.00000000 0.0000000 0

> ematrix.msm(heartmiscsex.msm, covariates=list(sex=1))
$estimates
      Stage 1      Stage 2      Stage 3 Stage 4
Stage 1 0.999979 2.14624e-05 0.0000000 0
Stage 2 0.446370 5.22069e-01 0.0315608 0
Stage 3 0.000000 4.07443e-01 0.5925575 0
Stage 4 0.000000 0.00000e+00 0.0000000 1

$SE
      Stage 1      Stage 2      Stage 3 Stage 4
Stage 1 0.000345081 0.000345081 0.0000000 0
Stage 2 0.181070516 0.178663859 0.0315352 0
Stage 3 0.000000000 0.239361085 0.2393611 0
Stage 4 0.000000000 0.000000000 0.0000000 0

```

Odds ratios for misclassification The function `odds.msm` gives the estimated odds ratios corresponding to each covariate effect on the misclassification probabilities.

```
> odds.msm(heartmiscsex.msm)
$sex
              OR              L95              U95
Obs Stage 1 | Stage 2 2.952283591 6.826960e-01 1.276700e+01
Obs Stage 2 | Stage 1 0.001924820 3.932326e-17 9.421730e+10
Obs Stage 2 | Stage 3 5.590513158 7.265350e-01 4.301766e+01
Obs Stage 3 | Stage 2 0.551507369 6.742341e-02 4.511199e+00
```

Observed and expected prevalences The function `prevalence.msm` is intended to assess the goodness of fit of the hidden Markov model for the *observed* states to the data. Tables of observed prevalences of observed states are calculated as described in section 2.11, by assuming that observed states are retained between observation times.

The expected numbers of individuals in each observed state are calculated similarly. Suppose the process begins at a common time for all individuals, and at this time, the probability of occupying *true* state r is f_r . Then given $n(t)$ individuals under observation at time t , the expected number of individuals in true state r at time t is the r th element of the vector $n(t)fP(t)$. Thus the expected number of individuals in *observed* state r is the r th element of the vector $n(t)fP(t)E$, where E is the misclassification probability matrix.

3.5 Recreating the path through underlying states

In speech recognition and signal processing, *decoding* is the procedure of determining the underlying states that are most likely to have given rise to the observations. The most common method of reconstructing the most likely state path is the *Viterbi* algorithm. Originally proposed by Viterbi [27], it is also described by Durbin *et al.* [19] and Macdonald and Zucchini [24] for discrete-time hidden Markov chains. For continuous-time models it proceeds as follows. Suppose that a hidden Markov model has been fitted and a Markov transition matrix $P(t)$ and misclassification matrix E are known. Let $v_i(k)$ be the probability of the most probable path ending in state k at time t_i .

1. Estimate $v_k(t_1)$ using known or estimated initial-state occupation probabilities.
2. For $i = 1 \dots N$, calculate $v_l(t_i) = e_{l,O_{t_i}} \max_k v_k(t_{i-1})P_{kl}(t_i - t_{i-1})$. Let $K_i(l)$ be the maximising value of k .
3. At the final time point N , the most likely underlying state S_N^* is the value of k which maximises $v_k(T_N)$.
4. Retrace back through the time points, setting $S_{i-1}^* = K_i(S_i^*)$.

The computations should be done in log space to prevent underflow. The `msm` package provides the function `viterbi.msm` to implement this method. For example, the following is an extract from a result of calling `viterbi.msm` to determine the most likely underlying states for all patients. The results for patient 100103 are shown, who appeared to ‘recover’ to a less severe stage of disease while in stage 3. We assume this is not biologically possible for the true states, so we expect that either the observation of state 3 at time 4.98 was an erroneous observation of state 2, or their apparent state 2 at time 5.93 was actually state 3. According to the expected path constructed using the Viterbi algorithm, it is the observation at time 5.93 which is misclassified.

```

> vit <- viterbi.msm(heartmisc.msm)
> vit[vit$subject==100103,]
  subject      time observed fitted
567 100103 0.00000         1      1
568 100103 2.04110         1      1
569 100103 4.08219         2      2
570 100103 4.98356         3      3
571 100103 5.93699         2      3
572 100103 7.01370         3      3
573 100103 8.04658         3      3
574 100103 8.43562         4      4

```

4 **msm** reference guide

The R help page for `msm` gives details of all the allowed arguments and options to the `msm` function. To view this online in R, type:

```
> ?msm
```

Similarly all the other functions in the package have help pages, which should always be consulted in case of doubt about how to call them. The web-browser based help interface may often be convenient - type

```
> help.start()
```

and navigate to **Packages ... msm**, which brings up a list of all the functions in the package with links to their documentation, and a link to this manual in PDF format.

References

- [1] D. R. Cox and H. D. Miller. *The Theory of Stochastic Processes*. Chapman and Hall, London, 1965.
- [2] C. H. Jackson, L. D. Sharples, S. G. Thompson, S. W. Duffy, and E. Couto. Multistate Markov models for disease progression with classification error. *Journal of the Royal Statistical Society, Series D - The Statistician*, 52(2):193–209, July 2003.
- [3] C. H. Jackson and L. D. Sharples. Hidden Markov models for the onset and progression of bronchiolitis obliterans syndrome in lung transplant recipients. *Statistics in Medicine*, 21:113–128, 2002.
- [4] L. D. Sharples. Use of the Gibbs sampler to estimate transition rates between grades of coronary disease following cardiac transplantation. *Statistics in Medicine*, 12:1155–1169, 1993.
- [5] J. H. Klotz and L. D. Sharples. Estimation for a Markov heart transplant model. *The Statistician*, 43(3):431–436, 1994.
- [6] R. Kay. A Markov model for analysing cancer markers and disease states in survival studies. *Biometrics*, 42:855–865, 1986.
- [7] I. M. Longini, W. S. Clark, R. H. Byers, J. W. Ward, W. W. Darrow, G. F. Lemp, and H. W. Hethcote. Statistical analysis of the stages of HIV infection using a Markov model. *Statistics in Medicine*, 8:851–843, 1989.
- [8] G. A. Satten and I. M. Longini. Markov chains with measurement error: Estimating the ‘true’ course of a marker of the progression of human immunodeficiency virus disease. *Applied Statistics - Journal of the Royal Statistical Society Series C*, 45(3):275–295, 1996.
- [9] C. Guichenneuc-Jouyaux, S. Richardson, and I. M. Longini. Modelling markers of disease progression by a hidden Markov process: Application to characterising CD4 cell decline. *Biometrics*, 56:733–741, 2000.
- [10] R. C. Gentleman, J. F. Lawless, J. C. Lindsey, and P. Yan. Multi-state Markov models for analysing incomplete disease history data with illustrations for HIV disease. *Statistics in Medicine*, 13(3):805–821, 1994.
- [11] G. Marshall and R. H. Jones. Multi-state Markov models and diabetic retinopathy. *Statistics in Medicine*, 14, 1995.
- [12] P. K. Andersen. Multistate models in survival analysis: a study of nephropathy and mortality in diabetes. *Statistics in Medicine*, 7(6):661–670, 1988.
- [13] S. W. Duffy and H. H. Chen. Estimation of mean sojourn time in breast cancer screening using a Markov chain model of entry to and exit from preclinical detectable phase. *Statistics in Medicine*, 14:1531–1543, 1995.
- [14] H. H. Chen, S. W. Duffy, and L. Tabar. A Markov chain method to estimate the tumour progression rate from preclinical to clinical phase, sensitivity and positive predictive value for mammography in breast cancer screening. *The Statistician*, 45(3):307–317, 1996.

- [15] A. J. Kirby and D. J. Spiegelhalter. Statistical modelling for the precursors of cervical cancer. In *Case Studies in Biometry*. Wiley, New York, 1994.
- [16] P. K. Andersen, L. S. Hansen, and N. Keiding. Assessing the influence of reversible disease indicators on survival. *Statistics in Medicine*, 10:1061–1067, 1991.
- [17] J. Grüger, R. Kay, and M. Schumacher. The validity of inferences based on incomplete observations in disease state models. *Biometrics*, 47:595–605, 1991.
- [18] B. H. Juang and L. R. Rabiner. Hidden Markov models for speech recognition. *Technometrics*, 33:251–272, 1991.
- [19] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.
- [20] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.
- [21] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximisation technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41:164–171, 1970.
- [22] P. S. Albert. A mover-stayer model for longitudinal marker data. *Biometrics*, 55(4):1252–1257, 1999.
- [23] A. Bureau, J. P. Hughes, and S. C. Shiboski. An S-Plus implementation of hidden Markov models in continuous time. *Journal of Computational and Graphical Statistics*, 9:621–632, 2000.
- [24] I. L. Macdonald and W. Zucchini. *Hidden Markov and Other Models for Discrete-Valued Time Series*. Chapman and Hall, London, 1997.
- [25] J. K. Lindsey. *Models for Repeated Measurements*. Oxford Statistical Science Series. Oxford University Press, second edition, 1999.
- [26] P. Guttorp. *Stochastic Modeling of Scientific Data*. Chapman and Hall, London, 1995.
- [27] J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.