Overview

The `icmstate` package can be used to non-parametrically estimate the transition intensities in interval-censored multi-state models and use these to calculate and plot transition probabilities.

In this vignette, we first give a brief introduction to interval-censored data and multi-state modelling. Panel data is intermittently observed data, which is well described by interval-censored multi-state models. Afterwards, we focus on the possible approaches for non-parametric estimation present in this package and summarize some of the key functions of this package. We demonstrate the use of this package using simulated examples. We finish this vignette with a discussion on the choice of initial estimates.

1 Introduction

1.1 Interval-censoring

The study of partially observed outcomes is called survival analysis. Outcomes are oftentimes partially observed due to a phenomenon known as right-censoring, where an outcomes is only known to be larger than a certain value. This often occurs in medical problems, where the survival time (after surgery, treatment, . . .) of a patient is only known for a set of patients. This mathematical complication can happen for a number of reasons. Either patients are not followed up for a long enough time (study period was too short) or the patient is still alive or the patient was never inspected after a certain time period had passed due to other complications. The analysis of right-censored data is very well studied.

Another commonly encountered problem is interval-censoring. We speak of interval-censoring if the outcome is only known to lie between two values. An example of a setting where interval-censored data is collected is in the case of longitudinal follow-up. A patient might be assessed for the presence of a disease at multiple (scheduled) visits over a period of time. If the disease is detected at a certain visit, this means the patient has fallen ill between the previous two visit times. The exact time of infection is usually not known and nigh impossible to find out. Although data from such studies is clearly better described using interval-censoring, the time of infection is often assumed to be right-censored.

1.2 Multi-state modelling

Sometimes the available data is not well described by the simple survival setting, warranting the use of a more complicated multi-state model. In a multi-state model subjects can transition between different states, with the transition rate between states being of interest for estimation. The (extended) illness-death model is a famous example of a multi-state model, see Figure 1.

1.3 Panel data

When the state of subjects is observed intermittently, we obtain so called *panel data*. Such data is best described by an interval-censored multi-state model, as the exact transition times between states are not known. Additionally, it can be unclear exactly which transitions have been made between two observation times, as different paths can be taken to arrive at the same state in complex multi-state models.

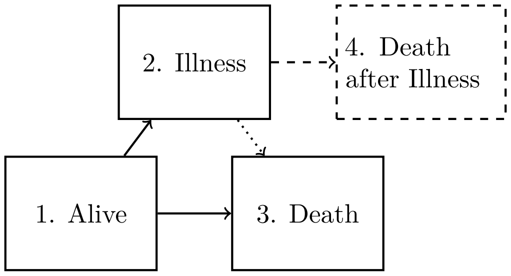


Figure 1: Graphical representation of the (extended) illness-death model. Extended: solid and dashed lines. Standard: solid and dotted lines.

It sometimes occurs that the entry time into a subset of the states is always observed at an exact time. For example, in the illness-death model (Figure 1) it is reasonable to assume that the time of entry into the death state will be observed exactly. We will therefore also consider panel data with a mix of interval- and right-censored data.

2 Estimation

The `icmstate` package can be used to non-parametrically estimate the transition intensities in interval-censored Markov multi-state models. Let $\mathcal{H}$ denote the possible states in the model, so that $\mathcal{H} = \{1, 2, 3\}$ or $\mathcal{H} = \{\text{alive, illness, death}\}$ for the illness-death model. In general, the transition intensities are given by:

$$\alpha_{gh}(t) = \lim_{dt \downarrow 0} \frac{\mathbb{P}(X(t+dt) = h | X(t) = g, \mathcal{F}_{t-})}{dt} = \lim_{dt \downarrow 0} \frac{\mathbb{P}(X(t+dt) = h | X(t) = g)}{dt},$$

with $X(t)$ the state of the process at time t and $g, h \in \mathcal{H}$. Let $A_{gh}(t) = \int_0^t \alpha_{gh}(s) ds$ denote the cumulative transition intensity for the transition $g \to h$.

The transition probabilities are usually of greater interest than the intensities. These are defined as:

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

The transition intensities are related to the transition probabilities via product integration by the Chapman-Kolmogorov Equations:

$$\mathbf{P}(s,t) = \prod_{s<u\leq t} (\mathbf{I} + \mathrm{d}\mathbf{A}(u)), \tag{1}$$

with $\mathbf{P}(s,t)$ the matrix containing the transition probabilities, with rows representing transitions from a state and columns transitions to a state. Letting $H = |\mathcal{H}|$, we have that $\mathbf{I}$ is the $H \times H$ identity matrix and $\mathrm{d}\mathbf{A}(u)$ the matrix containing the transition intensities.

The theory above alows us to relate key quantities to each other. When estimating these quantities, some assumptions usually have to be made. Let us visualise a sample of 5 subjects following an illness-death model (see Figure 2).

In non-parametric estimation of simple interval-censored data, Turnbull [1976] had shown that it suffices to consider only the unique observation times of the subjects, and that the NPMLE of the cumulative intensities can only make jumps at these unique observation times.

Although this hasn't been directly shown to be true for general interval-censored multi-state models, a similar assumption is usually made. We determine all unique observation times, and name them $\tau_1, \ldots, \tau_K$ with K the total number of unique observation times. A visual representation with $K = 13$ of this can be seen in Figure 3.

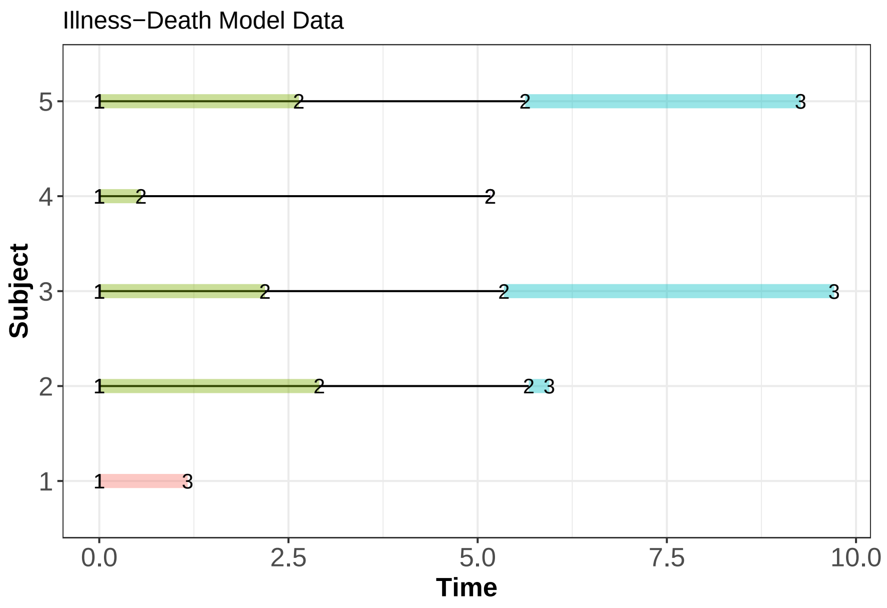


Figure 2: Visualisation of 5 subjects following an illness-death model. Each line represents the trajectory of a single subject, with the numbers indicating the state the subject was observed in at a certain time.

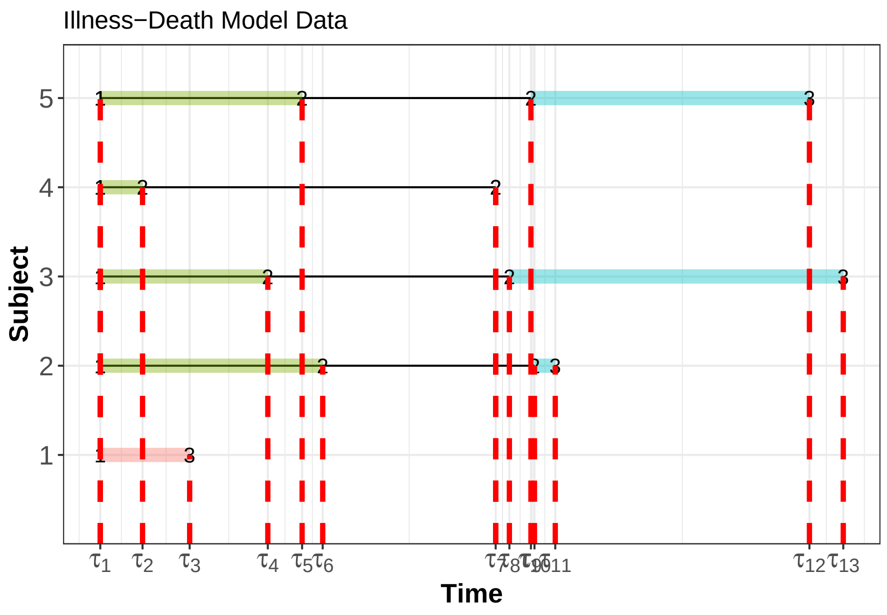


Figure 3: Unique observation times ordered chronologically and named.

Under this assumption, we are then interested in determining the jumps of the cumulative intensities only at these points in time. We therefore define:

$$\alpha_{gh}^k = \alpha_{gh}(\tau_k),$$

and this will be the main interest of our estimation procedure. Having obtained an estimate for these quantities, the Aalen-Johanssen estimator (Equation (1) with estimated intensities) can be used to recover transition probabilities.

2.1 Methods

The transition intensities are estimated using an EM algorithm. Two approaches are implemented in this package, described in the table below.

Method	Likelihood	Exactly observed states	R function
Gomon and Putter [2024]	Multinomial	Implemented	`npmsm(..., method = "multinomial")`
Gu et al. [2023]	Poisson	Not implemented	`npmsm(..., method = "poisson")`

The methods differ mostly in which complete-data likelihood is maximized, for details see the referenced articles. From here on out, we will refer to the two methods as Multinomial EM and Poisson EM. Although the two methods usually yield similar estimates, there are some pros and cons to each method.

- The Poisson EM estimator has been shown to be consistent for the cumulative intensities when subjects can start in all non-absorbing states. This is not likely to be true for most panel data.
- The Multinomial EM algorithm requires way less iterations to converge to an estimate than the Poisson estimator. Due to this, computation times are usually more than twice as fast.
- The Multinomial EM algorithm allows for the incorporation of exactly observed states. Although possible with the Poisson EM algorithm as well, only a smooth approximation has been developed and is not incorporated in this package.
- The estimates for the transition probabilities have been shown to be consistent for both methods in a simulation study (Gomon and Putter [2024]).

For most users, it should suffice to choose the appropriate method as follows:

1. If not all subjects are observed at the beginning of the study (time 0) and the subjects can initially be observed in all non-absorbing states and the cumulative intensities are of interest: Poisson EM.
2. In all other cases, the computational advantage of the Multinomial EM algorithm will make it the more desirable choice.

3 Key functions

A short description of the key functions in this package is given in this section.

Function	Use case
`npmsm()`	Main function: Estimate NPMLE for general multi-state models without loop. Output can be displayed using the `plot()` and `print()` methods.
`transprob()`	Determine transition probabilities from a `npmsm` or `msm` fit. Can `plot()` the result.
`visualise_msm()`	Visualise multi-state data.
`plot_probtrans()`	Plot and compare transition probabilities for multiple `npmsm` fits.
`plot_surv()`	Plot and compare transition specific 'Kaplan-Meier' plots between `npmsm` fits.

4 Examples

We demonstrate basic usage of this package through a few simulated examples.

4.1 Simple interval-censoring

The simplest form of a multi-state model is a model with two states. The NPMLE for this problem was found by Turnbull [1976]. We show how to fit a simple model and compare our result with the well-known Turnbull estimator.

This "standard" survival setting can be represented by using the following *transition* matrix:

```r
library(mstate)
#> Loading required package: survival
tmat <- transMat(x = list( c(2), c() ))
```

We create transition matrices as in the `mstate` package (Wreede et al. [2011]).

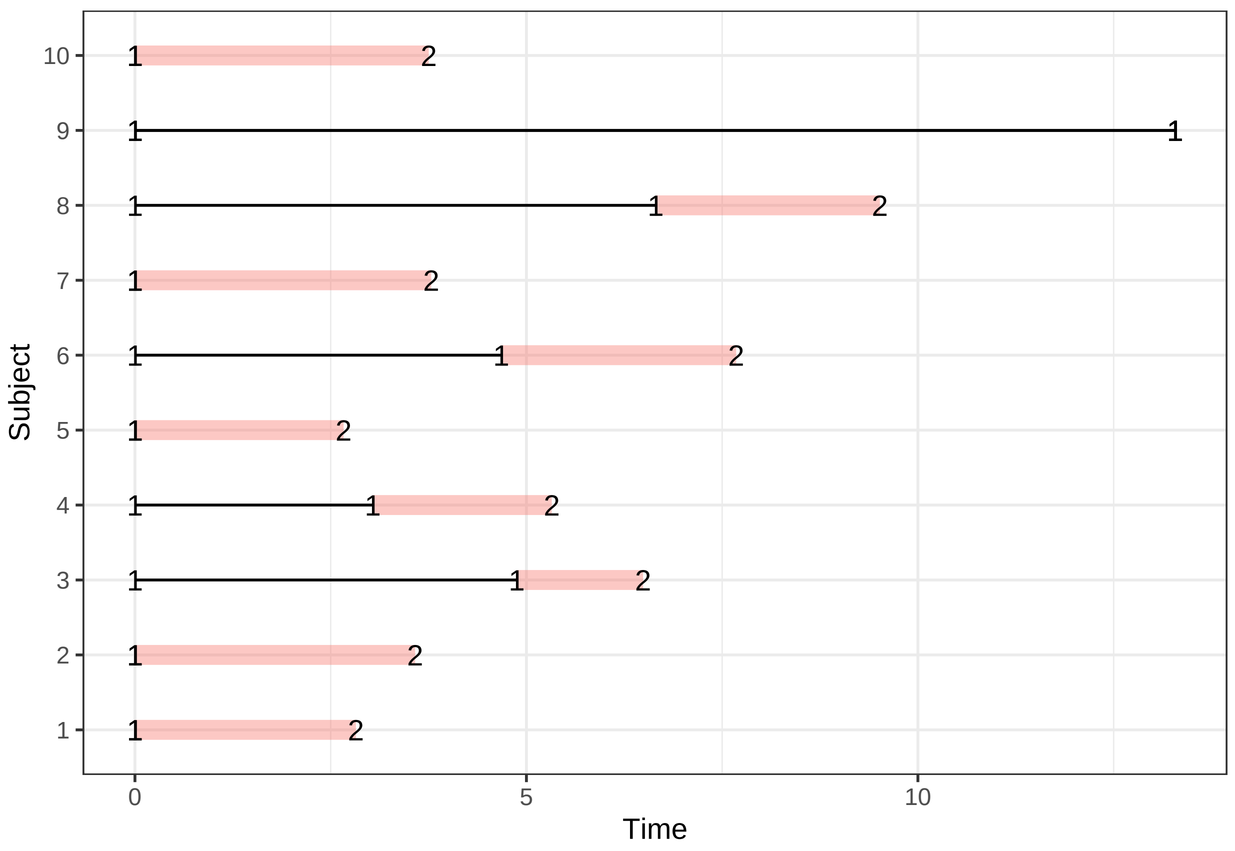
This transition matrix tells us that there are 2 states, with a possible transition to state 2 from state 1 and no transitions out of state 2 (absorbing).

We want to generate some interval-censored data. This is possible by using the `msm` package (Jackson [2011]). We generate data with $\alpha_{12}(t) = 0.2$ for 10 subjects.

```r
library(msm)
#Entry: constant transition rate from row state to column state
#Absorbing state 2
qmatrix <- rbind(
  c(-0.2, 0.2),
  c(0, 0)
)
#Number of subjects in simulated data
n <- 10
#time = observation time, subject = subject identifier
simdat <- data.frame(time = c(replicate(n, c(0, seq(2, 12, by=2) + runif(6, 0, 2)))),
                     subject = rep(1:n, each = 7))
#Simulate interval-censored data. See help(simmulti.msm)
dat <- simmulti.msm(data = simdat, qmatrix = qmatrix, start = 1)[, 1:3]
names(dat)[1] <- "id"
```

We can visualise the data using the `visualise_msm()` function:

```
visualise_msm(dat)
```



The visualisation will only show the observation times which are relevant for transitions, getting rid of repeated observations in the same state and will remove extra observations in an absorbing state if the argument `tmat` is supplied.

We transform the data for use by the `icenReg` package, so we can determine the Turnbull estimator

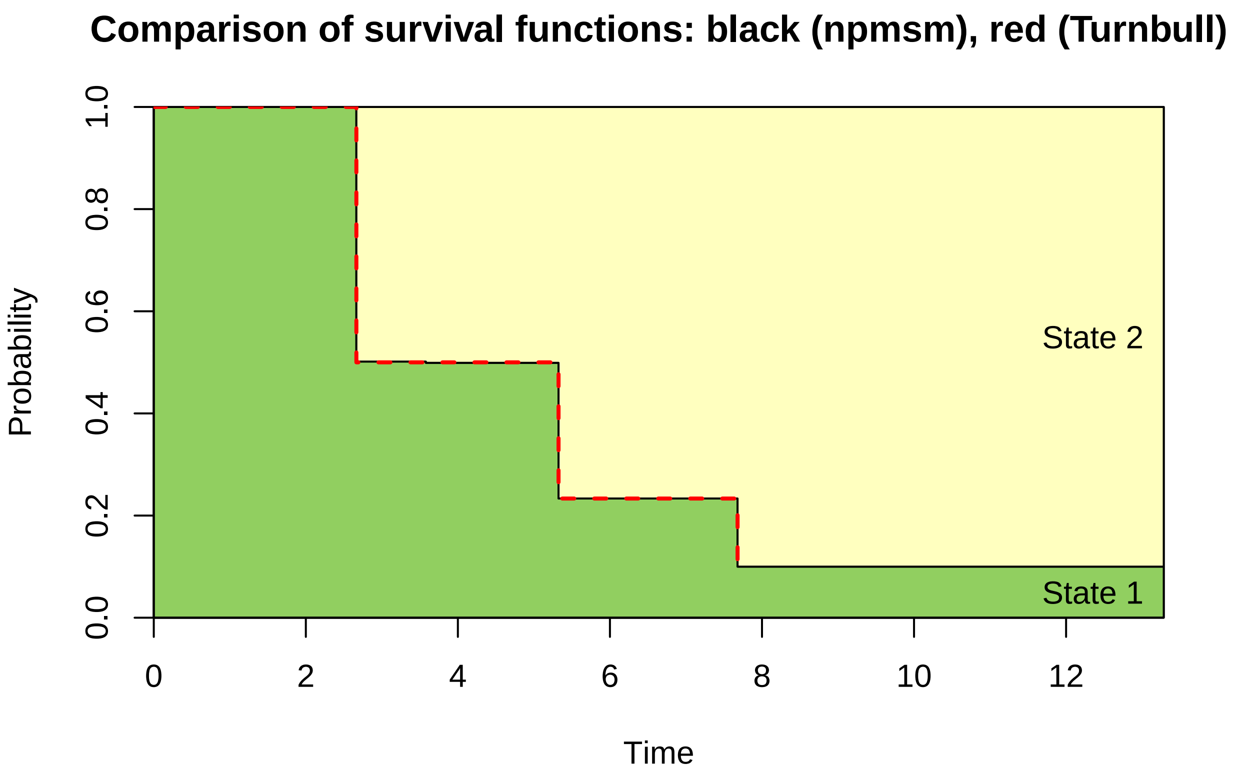
```
icdata <- NULL
for(i in unique(dat$id)){
  gdi <- subset(dat, id == i)
  L_idx <- Position(isTRUE, gdi$state <= 1, right = TRUE)
  L <- gdi$time[L_idx] #Exit time from state 1
  if(L_idx < nrow(gdi)){
    R <- gdi$time[L_idx + 1]
  } else{
    R <- Inf
  }
  icdata <- rbind(icdata, c(L, R))
}
icdata <- as.data.frame(icdata)
colnames(icdata) <- c("L", "R")
```

Now we can determine the NPMLE using both methods:

```r
library(icenReg)
SimpleMult <- npmsm(gd = dat, tmat = tmat, method = "multinomial")
Turnbull <- ic_np(cbind(L, R) ~ 0, data = icdata)
```

The Turnbull estimator usually extracts $p_{12}^k = F(\tau_k) - F(\tau_{k-1})$ instead of the intensities. Note that we cannot compare different models on their estimated intensities if the intensities are calculated over different intervals! We can however compare the resulting survival functions of the two approaches. For the Turnbull estimator, the survival function is given by $S(t) = 1 - F(t) = 1 - \sum_{\tau_k \leq t} p_{12}^k$. For the multi-state estimator, we can extract the survival function using transition probabilities $S(t) = \mathbb{P}(X(t) = 1 | X(0) = 1)$. This can be done using the `transprob()` function.

```r
#Extract Survival function from Turnbull estimate
surv_turnbull <- sapply(1:length(Turnbull$p_hat), function(i) 1 - sum(Turnbull$p_hat[1:i]))
#Extract Survival function using transition probabilities
surv_npmsm <- transprob(SimpleMult, predt = 0, direction = "forward")
#Plot the result
plot(surv_npmsm, main = "Comparison of survival functions: black (npmsm), red (Turnbull)")
lines(c(0, Turnbull$T_bull_Intervals[2,]), c(1, surv_turnbull), type = "s", col = "red",
      lwd = 2, lty = 2)
```



We can see that both methods recover the same survival function, as expected.

4.2 Time homogeneous example

Let us consider the illness-death model with time-homogeneous transition intensities. We assume that all transitions are interval-censored. This means that $\alpha_{gh}(t) = \lambda_{gh}$ for the possible transitions in the model: $1 \to 2, 2 \to 3$ and $1 \to 3$ so that $\lambda_{gh}(t) = 0.1$ for each transition. We generate some data using the `msm` package:

```r
#Absorbing state 3, transition intensities constant = 0.1
qmatrix <- rbind(
  c(-0.2, 0.1, 0.1),
  c(0, -0.1, 0.1),
  c(0, 0, 0)
)
#Create a transition matrix
tmat_ID <- trans.illdeath()
#Number of subjects in simulated data
n <- 100
#Create data frame for simulation:
simdat_ID <- data.frame(time = c(replicate(n, c(0, seq(2, 12, by=2) + runif(6, 0, 2)))),
                    subject = rep(1:n, each = 7))
dat_ID <- simmulti.msm(data = simdat_ID, qmatrix = qmatrix, start = 1)[, 1:3]
names(dat_ID)[1] <- "id"
```

Let us take a visual look at the data of the first 20 subjects (Figure 4):

```r
visualise_msm(subset(dat_ID, id < 21))
```

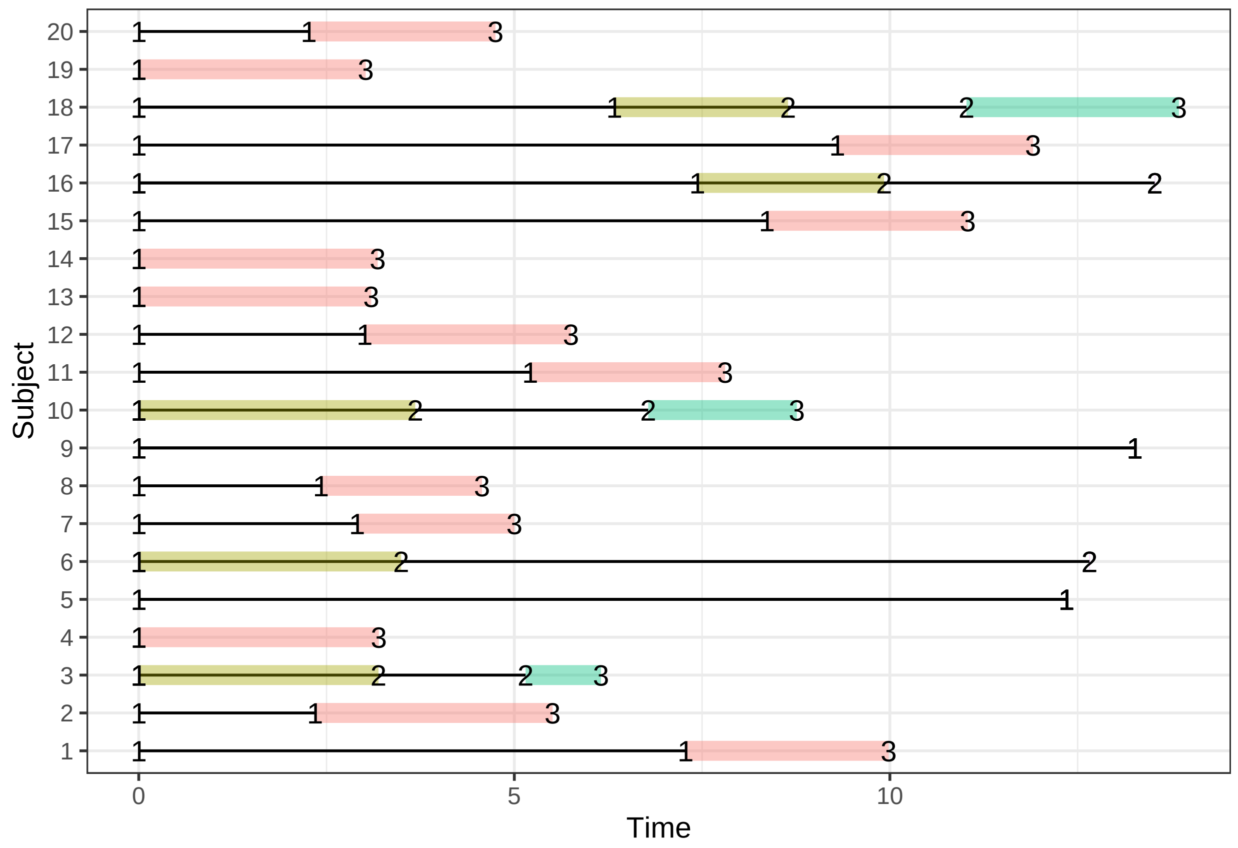


Figure 4: Visualisation of 20 subjects in an illness-death model

We fit the illness-death model using the `npmsm()` function and as the data was simulated using time homogeneous intensities, we can also use the `msm` package (function `msm()`) to fit an appropriate model to the data:

```r
#Fit appropriate MSM using the msm package
ID_msm <- msm(state ~ time, data = dat_ID, subject = id, qmatrix = qmatrix)
#Fit NP MSM using the icmstate package
ID_npmsm <- npmsm(gd = dat_ID, tmat = tmat_ID)
```

We plot the estimated cumulative intensities against each other (see Figure 5):

```r
plot(ID_npmsm, main = "Cumulative intensity: icmstate (solid lines) vs msm (dashed lines)")
#To plot output from the 'msm' function we need to extract the estimates
cols <- c("black", "red", "green")
for(i in 1:length(ID_msm$estimates)){
  abline(a = 0, b = exp(ID_msm$estimates)[i], col = cols[i], lty = 2)
}
```

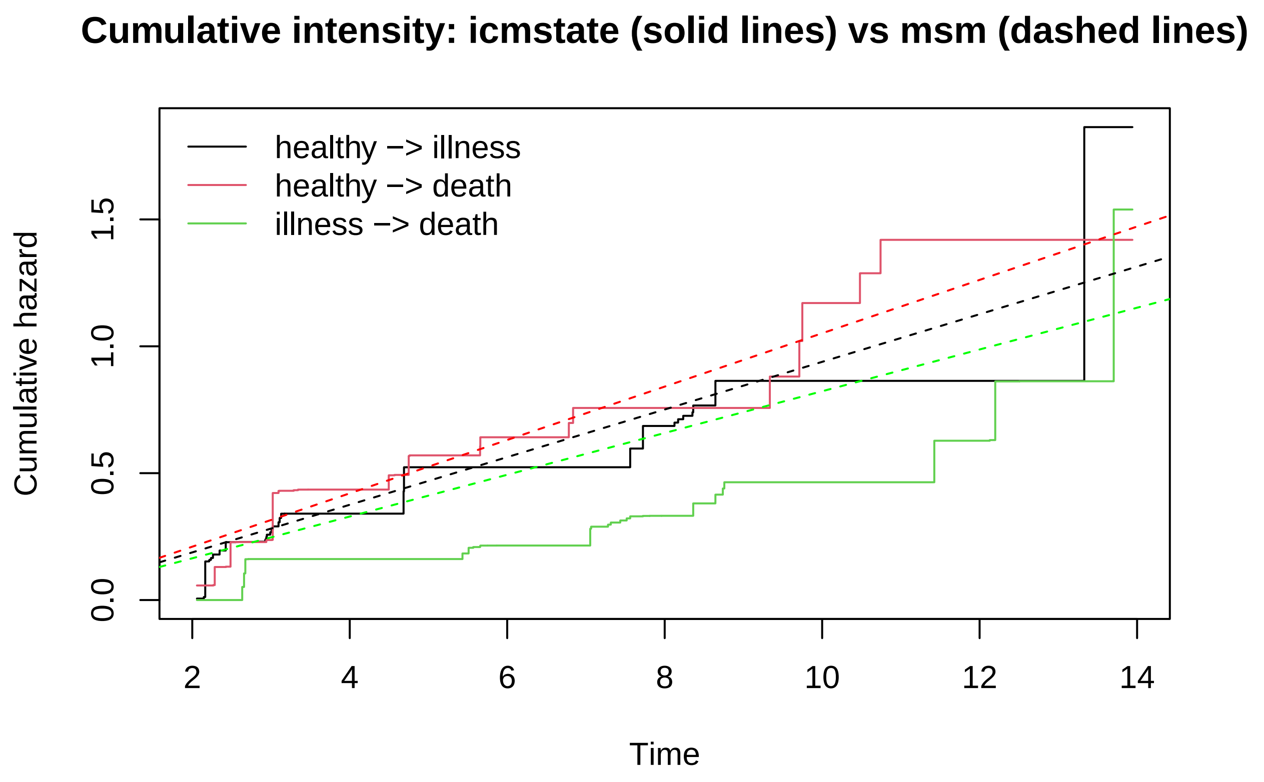


Figure 5: Cumulative intensity estimates using the msm package and icmstate package. True cumulative intensities are straight lines with slope 0.1 for each transition.

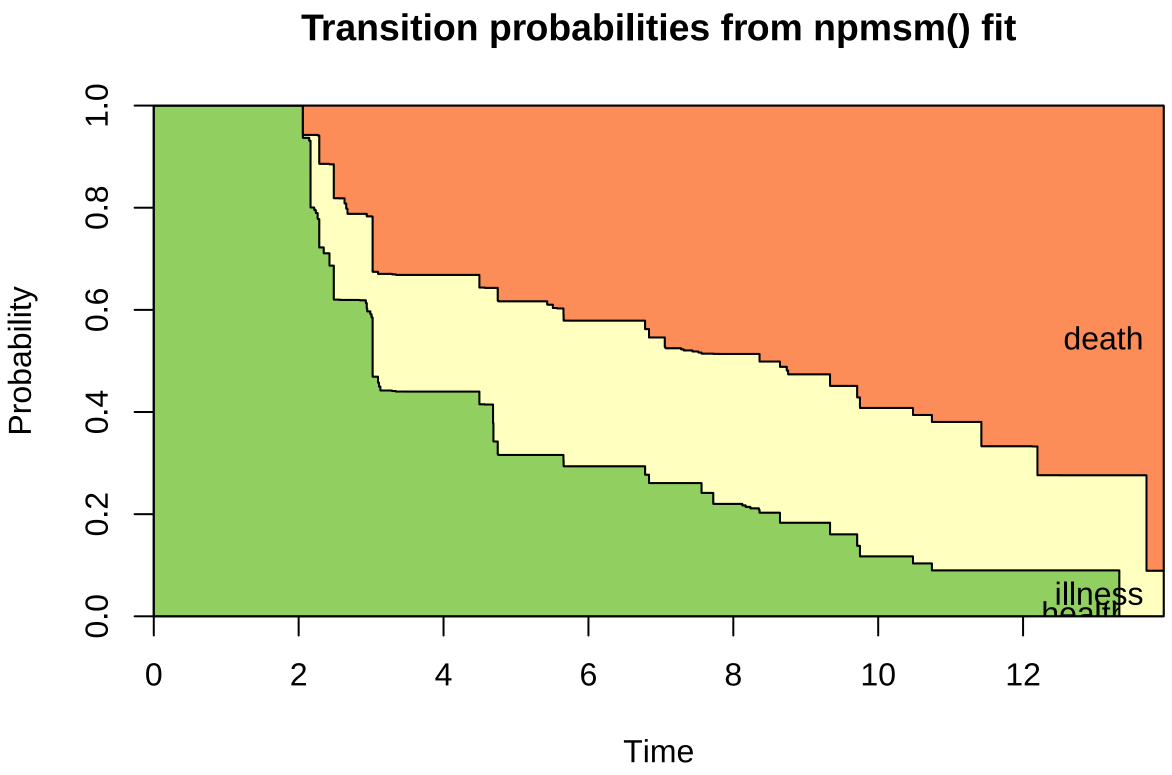
In this simulated example, we know that the true cumulative intensities are $A_{gh}(t) = 0.1t$, and both methods are not far off from the truth. We can check some diagnostics of the EM algorithm run by `icmstate` by simply printing the fitted model:

```r
ID_npmsm
#> Multinomial complete-data likelihood maximized through EM algorithm.
#> Convergence criterion: change in estimated intensities < 1e-04
#> Algorithm failed to converge after 100 out of max 100 iterations.
#> Consider increasing 'maxit'.
#> Log likelihood value: -239.554377546238
#> NPMLE has NOT been reached. Consider reducing 'tol' or increasing 'checkMLE_tol'.
#> Conclusion based on checking reduced_gradient < 1e-04
```

By default, the EM algorithm will only perform 100 iterations. This can be changed by the `maxit` argument. We find that the convergence criterion of a change in $\max_{g,h,k} \alpha_{gh}^k$ smaller than 0.0001 has not been reached in the first 100 iterations, therefore it would be advisable to re-run this model with more iterations. A check on whether the estimated intensities represent the NPMLE is also performed, up to a numerical tolerance given by `checkMLE_tol`. It is advisable to have the tolerance criterion and convergence criterion close to each other. If the tolerance criterion is much smaller than the convergence criterion, the numerical check will always return that the NPMLE has not been reached yet.

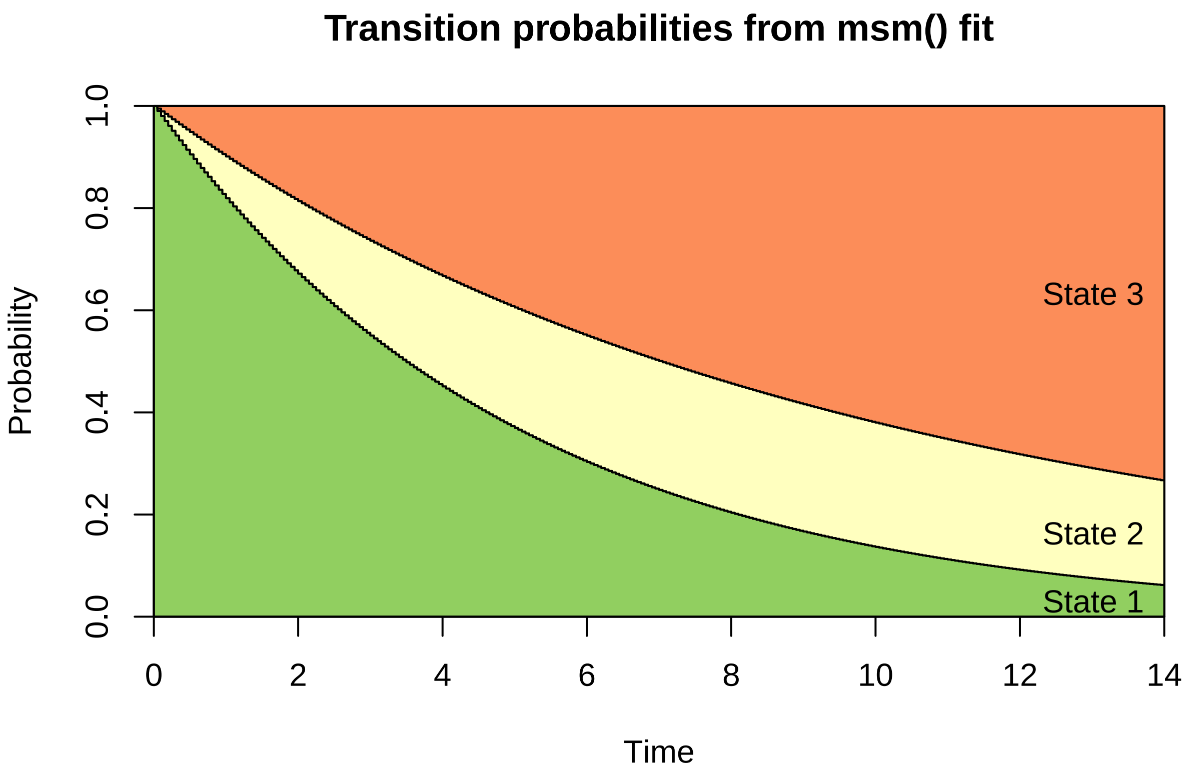
We might also be interested in determining transition probabilities for the estimated models. For this, we first need to use the `transprob()` function (based on `probtrans()` from the `mstate` package) and afterwards we can simply plot the result. For details, see `help(transprob.npmsm)`.

```
plot(transprob(object = ID_npmsm, predt = 0, direction = "forward", variance = FALSE),
     main = "Transition probabilities from npmsm() fit")
```



Similarly, we can plot the transition probabilities for the `msm` fit (see `help(transprob.msm)`).

```
plot(transprob(ID_msm, predt = 0, times = seq(0, 14, 0.05)),
     main = "Transition probabilities from msm() fit")
```



4.3 Time homogeneous example - exact observation times

Sometimes, it might occur that transitions into a certain state are always exactly observed whereas transitions between other states are interval-censored. A good example of this is the illness-death model, where the time of death is usually observed exactly. We show how to fit a suitable model for this situation, again assuming time homogeneous transition intensities.

First, we generate some data under these assumptions. Again, we use the `msm` package for this, with the argument `death = 3` to let the data generation mechanism know that the 3rd state is a "death'' state (and therefore exactly observed).

```r
#Absorbing state 3, transition intensities constant = 0.1
qmatrix <- rbind(
  c(-0.2, 0.1, 0.1),
  c(0, -0.1, 0.1),
  c(0, 0, 0)
)
#Create a transition matrix
tmat_ID <- trans.illdeath()
#Number of subjects in simulated data
n <- 100
#Create data frame for simulation:
simdat_ID_exact <- data.frame(time = c(replicate(n, c(0, seq(2, 12, by=2) + runif(6, 0, 2)))),
                   subject = rep(1:n, each = 7))
dat_ID_exact <- simmulti.msm(data = simdat_ID_exact, qmatrix = qmatrix, start = 1,
                            death = 3)[, 1:3]
names(dat_ID_exact)[1] <- "id"
```

Again, we fit a multi-state model using `npmsm()` (icmstate) as well as `msm()` (msm). For `icmstate`, we must specify the argument `exact` as the column number of the exactly observed state. In the cases of the illness-death model, this is the 3rd column. For `msm`, we similarly supply the argument `deathexact`.

```r
#Fit appropriate MSM using the msm package
ID_msm_exact <- msm(state ~ time, data = dat_ID_exact, subject = id, qmatrix = qmatrix,
                    deathexact = 3)
#Fit NP MSM using the icmstate package
ID_npmsm_exact <- npmsm(gd = dat_ID_exact, tmat = tmat_ID, exact = 3)
```

We plot the estimated cumulative intensities against each other (see Figure 6):

```r
plot(ID_npmsm_exact, main = "Cumulative intensity (exactly observed death):\n icmstate (solid lines) vs
cols <- c("black", "red", "green")
for(i in 1:length(ID_msm_exact$estimates)){
  abline(a = 0, b = exp(ID_msm_exact$estimates)[i], col = cols[i], lty = 2)
}
```

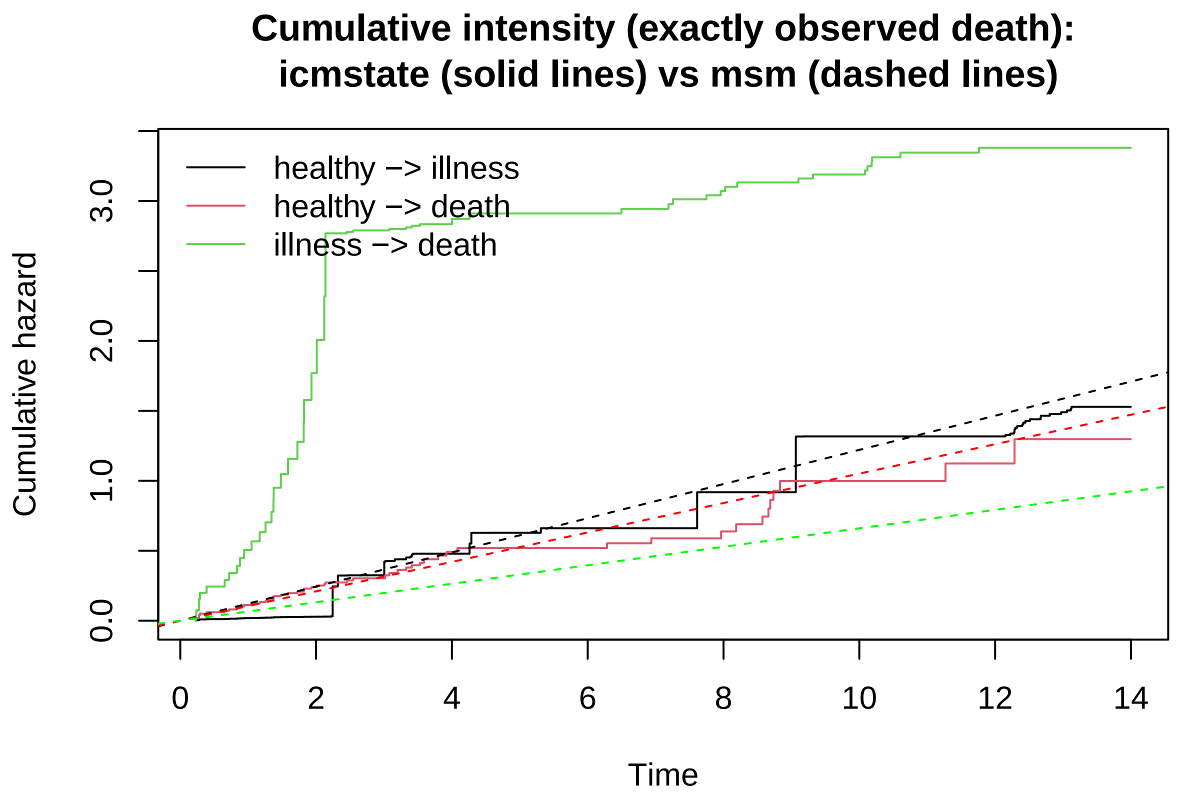
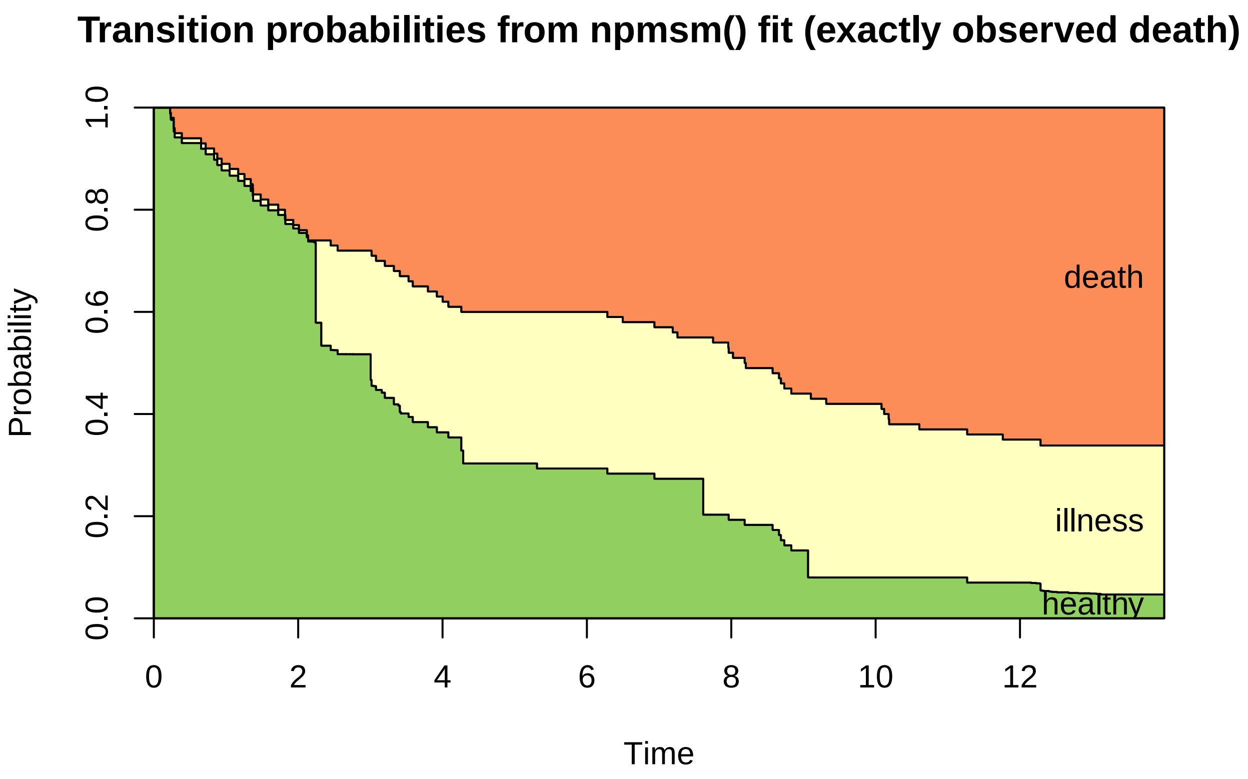


Figure 6: Cumulative intensity estimates using the msm package and icmstate package. True cumulative intensities are straight lines with slope 0.1 for each transition.

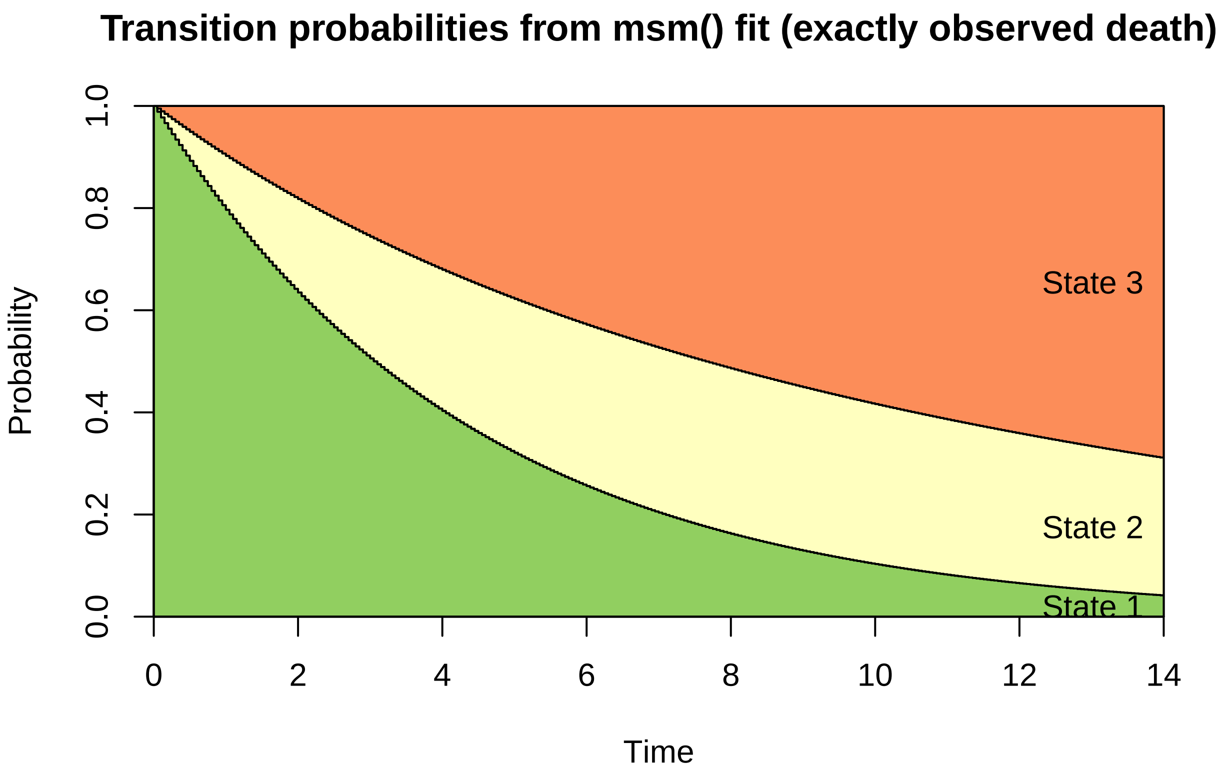
We can see that the estimated cumulative intensity for the illness to death transition (green) is completely wrong for the `icmstate` fit. This happens due to the fact that in the simulated data, no subject has entered the illness state before time 2.3. Because of this, the estimated cumulative intensity for the healthy to illness transition is zero until this point in time. As a consequence, the intensity of transitioning from alive to death is therefore overestimated as we now observe exact event times for death. If we look at the transition probabilities, the underlying issue becomes very apparent.

```r
plot(transprob(object = ID_npmsm_exact, predt = 0, direction = "forward", variance = FALSE),
     main = "Transition probabilities from npmsm() fit (exactly observed death)")
```



Similarly, we can plot the transition probabilities for the `msm` fit (see `help(transprob.msm)`).

```r
plot(transprob(ID_msm_exact, predt = 0, times = seq(0, 14, 0.05)),
     main = "Transition probabilities from msm() fit (exactly observed death)")
```



Although the estimated cumulative intensities are incorrect using the `icmstate` fit, the recovered transition probabilities align with the truth after time 2.3. We can see that after this time point, the estimated transition probabilities do not differ much between the models.

We therefore always recommend to inspect both the cumulative intensities as well as the transition probabilities to get a full picture of what the estimates actually mean.

4.4 Time inhomogeneous example

The two examples above have been using a time-homogeneous assumption. The power of this package lies in the fact that we do not require any assumption on the form of the underlying intensities. In this section, we therefore simulate data from a time-inhomogeneous model.

Let us simulate data from an illness-death model where the transition intensities between the states follow a Weibull distribution. This can be achieved using the `sim_id_weib()` function. The function requires the definition of an extra function which indicates at what times the simulated subjects are to be observed. Additionally, we need to choose a shape and scale parameter for the Weibull distribution for each of the three possible transitions. We specify these through the `shape` and `scale` arguments respectively. We choose the shape and scale parameters similarly to Gomon and Putter [2024].

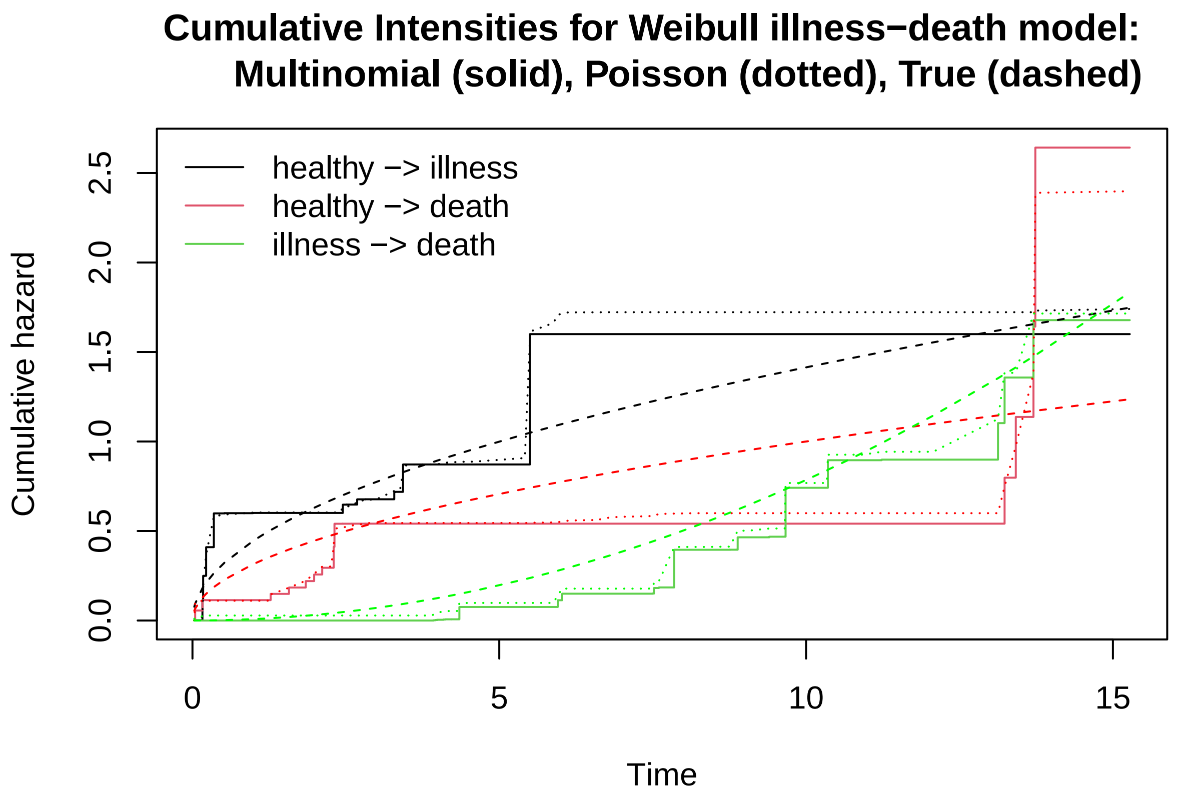
```
#When do we observe the subjects? n_obs times with uniform inter observation times
eval_times <- function(n_obs, stop_time){
  cumsum( c( 0,  runif( n_obs-1, 0, 2*(stop_time-4)/(n_obs-1) ) ) )
}

#Simulate data with Weibull shapes and scales.
dat_inhom <- sim_id_weib(n = 50, n_obs = 6, stop_time = 15, eval_times = eval_times,
start_state = "stable", shape = c(0.5, 0.5, 2), scale = c(5, 10, 10/gamma(1.5)))
```

We fit the model as usual, but now we consider both the Poisson and Multinomial EM approach:

```r
mult_fit <- npmsm(gd = dat_inhom, tmat = tmat_ID)
pois_fit <- npmsm(gd = dat_inhom, tmat = tmat_ID, method = "poisson")
```

We plot the cumulative intensities against each other, and add the true underlying cumulative intensities as well:

```r
plot(mult_fit, main = "Cumulative Intensities for Weibull illness-death model:
     Multinomial (solid), Poisson (dotted), True (dashed)")
shapes <- c(0.5, 0.5, 2)
scales <- c(5, 10, 10/gamma(1.5))
for(i in 1:3){
  trans_dat <- subset(pois_fit$A$Haz, trans == i)
  #Add poisson estimates
  lines(trans_dat$time, trans_dat$Haz, lty = 3, col = cols[i])
  #Add true cumulative intensities
  lines(trans_dat$time,
        -pweibull(trans_dat$time, shapes[i], scales[i], lower = FALSE, log = TRUE),
        lty = 2, col = cols[i])
}
```



We can see that the estimates align with the true intensities. Some instabilities are observed at the end of the time period, as is expected due to the small sample size.

5 Initial estimates

The EM algorithms implemented in this package require the choice of initial estimates for the parameters α_{gh}^k. The package currently allows for four different choices of initial estimates. The choices are:

- `equalprob`: This amounts to choosing $\alpha_{gh}^k = \frac{1}{K}$, assigning an initial estimate based on the total number of bins.
- `homogeneous`: An estimate of the intensities is made assuming time-homogeneous transition rates. See `help(crudeinits.msm)` from the `msm` package for more information. If λ is the estimated intensity of the transition, we assign $\alpha_{gh}^k = \lambda \cdot (\tau_k - \tau_{k-1})$.
- `unif`: We generate α_{gh}^k from a Unif$[0, 1]$ distribution.
- `beta`: We generate α_{gh}^k from a Beta(a,b) distribution. Specify a and b through `beta_params`, see `help(pbeta)`.

A different choice of initial estimates can lead to a different result. Choosing an appropriate starting point can also help the algorithm to converge faster. As an example, for the time-homogeneous case choosing time-homogeneous initial estimates can speed up the estimation process. Let us fit a multi-state model using different initial estimates for a subset of the simulated data-set considered in Section time homogeneous example.

```r
#Fit NP MSM using the icmstate package
dat_ID_small <- subset(dat_ID, id < 21)
ID_npmsm_equalprob <- npmsm(gd = dat_ID_small, tmat = tmat_ID, maxit = 300)
ID_npmsm_hom <- npmsm(gd = dat_ID_small, tmat = tmat_ID, inits = "homogeneous", maxit = 300)
ID_npmsm_unif <- npmsm(gd = dat_ID_small, tmat = tmat_ID, inits = "unif", maxit = 300)
ID_npmsm_beta <- npmsm(gd = dat_ID_small, tmat = tmat_ID, inits = "beta", beta_params = c(3, 6), maxit
```

We compare the value of the likelihood at convergence and the number of iterations required to reach convergence:

```r
fit_summary <- sapply(list(ID_npmsm_equalprob, ID_npmsm_hom, ID_npmsm_unif, ID_npmsm_beta),
       function(x) c(x$it, x$ll))
attr(fit_summary, "dimnames") <- list("summary" = c("iterations", "likelihood"),
                                      "Initial" = c("EqProb", "Hom", "Unif", "Beta"))
fit_summary
#>              Initial
#> summary          EqProb       Hom      Unif      Beta
#>   iterations  97.00000 144.00000 152.00000 109.00000
#>   likelihood -33.49685 -33.49669 -33.49672 -33.49687
```

In our experience, `equalprob` usually yields the best estimates in the smallest number of iterations. In this case, the `homogeneous` initial conditions yield the largest log-likelihood value, but after the most iterations.

References

Daniel Gomon and Hein Putter. Non-parametric estimation of transition intensities in interval censored markov multi-state models without loops, 2024. URL https://arxiv.org/abs/2409.07176.

Y. Gu, D. Zeng, G. Heiss, and D. Y. Lin. Maximum likelihood estimation for semiparametric regression models with interval-censored multistate data. *Biometrika*, November 2023. ISSN 1464-3510. doi: 10.1093/biomet/asad073.

C. H. Jackson. Multi-state models for panel data: The msm package for R. *Journal of Statistical Software*, 38(8), 2011. ISSN 1548-7660. doi: 10.18637/jss.v038.i08.

B. W. Turnbull. The empirical distribution function with arbitrarily grouped, censored and truncated data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 38(3):290–295, July 1976. ISSN 2517-6161. doi: 10.1111/j.2517-6161.1976.tb01597.x.

Liesbeth C. de Wreede, Marta Fiocco, and Hein Putter. mstate: An r package for the analysis of competing risks and multi-state models. *Journal of Statistical Software*, 38(7), 2011. ISSN 1548-7660. doi: 10.18637/jss.v038.i07.