Continuous-time HMMs

Jan-Ole Koslik

The regular HMM formulation needs a key assumption to be applicable, namely the data need to be observed at regular, equidistant time-points such that the transition probabilties can be interpreted meaningfully w.r.t. a specific time unit. If this is not the case, the model used should acocunt for this by building on a mathematical formulation in continuous time. The obvious choice here is to retain most of the HMM model formulation, but replace the unobserved discrete-time Markov chain with a continuous-time Markov chain.

As transition probabilities can only be specified w.r.t. to some time unit, but we have irregular observation times $t_1, \dots, t_T$, such a chain is characterized by a so-called **(infinitesimal) generator matrix**

$$Q = \begin{pmatrix} -\lambda_1 & \lambda_1\omega_{12} & \lambda_1\omega_{13} & \cdots \\ \lambda_2\omega_{21} & -\lambda_2 & \lambda_2\omega_{23} & \\ \lambda_3\omega_{31} & \lambda_3\omega_{32} & -\lambda_3 & \\ \vdots & & & \ddots \end{pmatrix} = \begin{pmatrix} q_{11} & q_{12} & q_{13} & \cdots \\ q_{21} & q_{22} & q_{23} & \\ q_{31} & q_{32} & q_{33} & \\ \vdots & & & \ddots \end{pmatrix},$$

where the diagonal entries are $q_{ii} = -\sum_{j \neq i} q_{ij}$, $q_{ij} \geq 0$ for $i \neq j$. This matrix can be interpreted as the derivative of the transition probability matrix and completely describes the dynamics of the state process. For such a chain, the time-spent in a state i is exponentially distributed with rate $-q_{ii}$ and conditional on leaving the state, the probability to transition to a state $j \neq i$ is $\omega_{ij} = q_{ij}/\lambda_i$. For a more detailed introduction see Dobrow (2016) (pp. 265 f.). For observation times t_1 and t_2, we can then obtain the transition probability matrix between these points via the identity

$$\Gamma(t_1, t_2) = \exp(Q(t_2 - t_1)),$$

where exp() is the matrix expoential. This follows from the so-called Kolmogorov forward equations, but again, for more details see Dobrow (2016). We can easily see that such a model now accounts for irregular observation times, while not explicitely modelling them as a random variable.

Example 1: two states

Setting parameters for simulation

```
# 2-state example

# generator matrix Q:
Q = matrix(c(-0.5,0.5,1,-1), nrow = 2, byrow = TRUE)
# state 1 has a smaller rate (dwell-time in state one ~ Exp(1)), i.e. it exhibits
# longer dwell times than state 3 with rate 3.

# parameters for the state-dependent (normal) distributions
mu = c(5, 20)
sigma = c(2, 5)
```

Simulating data

We simulate the continuous-time Markov chain by drawing the exponentially distributed state dwell-times. Within a stay, we can assume whatever structure we like for the observation times, as these are not explicitly modeled. Here we choose to generate them by a Poisson process with rate $\lambda = 1$, but this choice is arbitrary. For more details on Poisson point processes, see the MM(M)PP vignette.

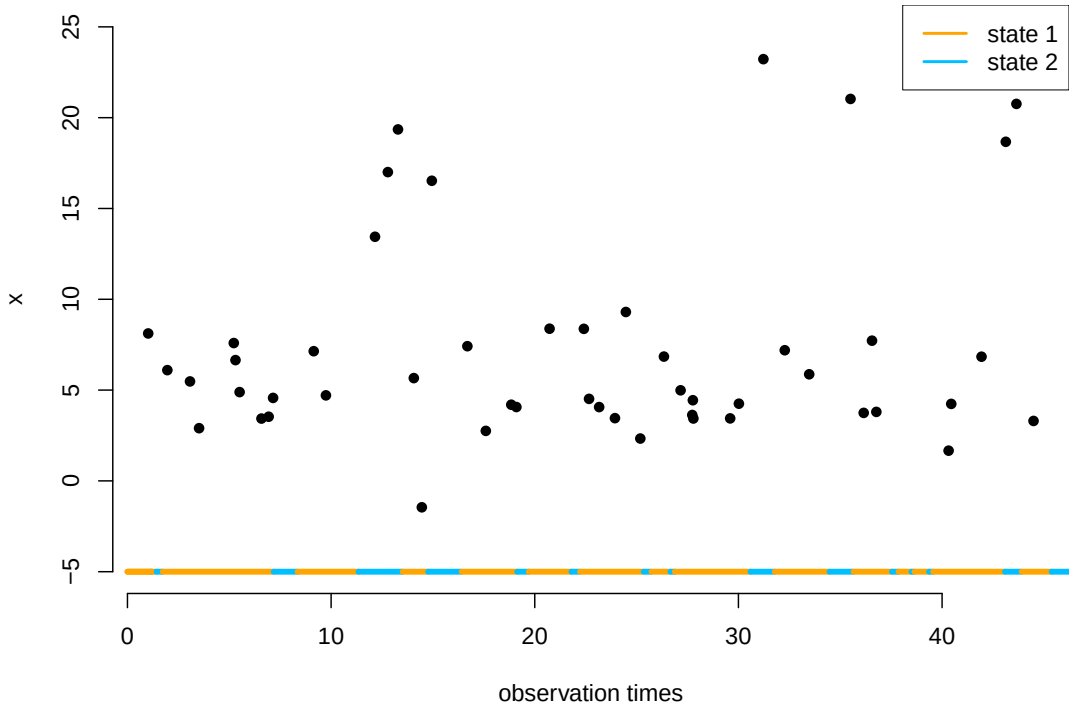
```
set.seed(123)

k = 200 # number of state switches
trans_times = s = rep(NA, k) # time points where the chain transitions
s[1] = sample(1:2, 1) # initial distribuion c(0.5, 0.5)
# exponentially distributed waiting times
trans_times[1] = rexp(1, -Q[s[1],s[1]])
n_arrivals = rpois(1, trans_times[1])
obs_times = sort(runif(n_arrivals, 0, trans_times[1]))
x = rnorm(n_arrivals, mu[s[1]], sigma[s[1]])
for(t in 2:k){
  s[t] = c(1,2)[-s[t-1]] # for 2-states, always a state swith when transitioning
  # exponentially distributed waiting times
  trans_times[t] = trans_times[t-1] + rexp(1, -Q[s[t], s[t]])
  n_arrivals = rpois(1, trans_times[t]-trans_times[t-1])
  obs_times = c(obs_times,
                sort(runif(n_arrivals, trans_times[t-1], trans_times[t])))
  x = c(x, rnorm(n_arrivals, mu[s[t]], sigma[s[t]]))
}
```

Let's visualize the simulated continuous-time HMM:

```
color = c("orange", "deepskyblue")

n = length(obs_times)
plot(obs_times[1:50], x[1:50], pch = 16, bty = "n", xlab = "observation times",
     ylab = "x", ylim = c(-5,25))
segments(x0 = c(0,trans_times[1:48]), x1 = trans_times[1:49],
         y0 = rep(-5,50), y1 = rep(-5,50), col = color[s[1:49]], lwd = 4)
legend("topright", lwd = 2, col = color,
       legend = c("state 1", "state 2"), box.lwd = 0)
```



Writing the negative log-likelihood function

The likelihood of a continuous-time HMM for observations $x_{t_1}, \ldots, x_{t_T}$ at irregular time points $t_1, \ldots, t_T$ has the exact same structure as the regular HMM likelihood:

$$L(\theta) = \delta^{(1)}\Gamma(t_1, t_2)P(x_{t_2})\Gamma(t_2, t_3)P(x_{t_3})\ldots\Gamma(t_{T-1}, t_T)P(x_{t_T})1^t,$$

where $\delta^{(1)}$, P and 1^t are as usual. Thus we can fit such models using the standard implementation of the general forward algorithm `forward_g()` with time-varying transition probability matrices.

```
mllk = function(theta.star, timediff, x, N=2){
  mu = theta.star[1:N]
  sigma = exp(theta.star[N+1:N])
  Q = diag(N) # generator matrix
  Q[!Q] = exp(theta.star[2*N+1:(N*(N-1))])
  diag(Q) = 0
  diag(Q) = -rowSums(Q)
  delta = solve(t(Q+1), rep(1,N), tol = 1e-20) # stationary distribution of the
  # continuous-time Markov chain
  Qube = Lcpp::tpm_cont(Q, timediff) # this computes exp(Q*timediff)
  allprobs = matrix(1, nrow = length(x), ncol = N)
  ind = which(!is.na(x))
  for(j in 1:N){
    allprobs[ind,j] = dnorm(x[ind], mu[j], sigma[j])
  }
  -Lcpp::forward_g(delta, Qube, allprobs)
}
```

Fitting a continuous-time HMM to the data

```
theta.star = c(5, 15, log(3), log(5), # mu and sigma
               log(1), log(0.5)) # off-diagonals of Q

timediff = diff(obs_times)

t1 = Sys.time()
mod = stats::nlm(mllk, theta.star, timediff=timediff, x=x, stepmax = 10)
# we often need the stepmax, as the matrix exponential can be numerically unstable
Sys.time()-t1
#> Time difference of 0.06164193 secs
```

Results

```
N = 2
# mu
round(mod$estimate[1:N],2)
#> [1]  5.06 20.24
# sigma
round(exp(mod$estimate[N+1:N]))
#> [1] 2 5
Q = diag(N) # generator matrix
Q[!Q] = exp(mod$estimate[2*N+1:(N*(N-1))])
diag(Q) = 0
diag(Q) = -rowSums(Q)
round(Q,3)
#>        [,1]   [,2]
#> [1,] -0.479  0.479
#> [2,]  0.905 -0.905
```

Example 2: three states

Setting parameters for simulation

```
# 2-state example

# generator matrix Q:
Q = matrix(c(-0.5,0.2,0.3,
             1,-2, 1,
             0.4, 0.6, -1), nrow = 3, byrow = TRUE)

# parameters for the state-dependent (normal) distributions
mu = c(5, 15, 30)
sigma = c(2, 3, 5)
```

Simulating data

The simulation is very similar but we now also have to draw which state to transition to, as explained in the beginning.

```
set.seed(123)

k = 200 # number of state switches
trans_times = s = rep(NA, k) # time points where the chain transitions
s[1] = sample(1:3, 1) # uniform initial distribuion
# exponentially distributed waiting times
trans_times[1] = rexp(1, -Q[s[1],s[1]])
n_arrivals = rpois(1, trans_times[1])
obs_times = sort(runif(n_arrivals, 0, trans_times[1]))
x = rnorm(n_arrivals, mu[s[1]], sigma[s[1]])
for(t in 2:k){
  # off-diagonal elements of the s[t-1] row of Q divided by the diagonal element
  # give the probabilites of the next state
  s[t] = sample(c(1:3)[-s[t-1]], 1, prob = Q[s[t-1],-s[t-1]]/-Q[s[t-1],s[t-1]])
  # exponentially distributed waiting times
  trans_times[t] = trans_times[t-1] + rexp(1, -Q[s[t], s[t]])
  n_arrivals = rpois(1, trans_times[t]-trans_times[t-1])
  obs_times = c(obs_times,
                sort(runif(n_arrivals, trans_times[t-1], trans_times[t])))
  x = c(x, rnorm(n_arrivals, mu[s[t]], sigma[s[t]]))
}
```

Fitting a 3-state continuous-time HMM to the data

```
theta.star = c(5, 10, 25, log(2), log(2), log(6), # mu and sigma
               rep(0, 6)) # off-diagonals of Q

timediff = diff(obs_times)

t1 = Sys.time()
mod2 = stats::nlm(mllk, theta.star, timediff=timediff, x=x, N = 3, stepmax = 10)
Sys.time()-t1
#> Time difference of 0.239269 secs
```

Results

```
N = 3
# mu
round(mod2$estimate[1:N],2)
#> [1]  4.90 15.45 29.10
# sigma
round(exp(mod2$estimate[N+1:N]),2)
#> [1] 1.80 2.58 5.06
Q = diag(N) # generator matrix
Q[!Q] = exp(mod2$estimate[2*N+1:(N*(N-1))])
```

```r
diag(Q) = 0
diag(Q) = -rowSums(Q)
round(Q, 3)
#>        [,1]   [,2]   [,3]
#> [1,] -0.888  0.565  0.323
#> [2,]  2.821 -3.469  0.647
#> [3,]  0.000  0.770 -0.770
```

References

Dobrow, Robert P. 2016. *Introduction to Stochastic Processes with r*. John Wiley & Sons.