A numerical example of uniformization

Yuxi Zhu

A numerical 3-state progressive model is used as an example to show how uniformization works. As an example, let's presume, the intensity matrix is

$$Q = \begin{bmatrix} -(0.2+0.1) & 0.2 & 0.1 \\ 0 & -0.35 & 0.35 \\ 0 & 0 & 0 \end{bmatrix}.$$

Now, we have $q_{12} = 0.2$, $q_{13} = 0.1$ and $q_{23} = 0.35$. The total rate out of state 1 is $q_1 = q_{12} + q_{13} = 0.2$. The total rate out of state 2 is $q_2 = q_{23} = 0.35$.

We can pick a Poisson rate $\lambda \geq max\{q_1, q_2\}$ as the uniformization rate to illustrate the whole process. For example, we pick $\lambda = 0.5$, then it means we can add self-transition rate to state 1 as $\lambda_{11} = \lambda - q_1 = 0.5 - 0.3 = 0.2$ and add self-transition rate to state 2 as $\lambda_{22} = \lambda - q_2 = 0.5 - 0.35 = 0.15$. Now, the new $q_1^{'} = q_{12} + q_{13} + \lambda_{11} = 0.5$ and the new $q_2^{'} = q_{23} + \lambda_{22} = 0.35 + 0.15 = 0.5$. So

$$q_1^{'} = q_2^{'} = \lambda.$$

By adding self-transitions to state 1 and state 2, the rates are uniformed across all states as $\lambda = 0.5$. This is where the name of uniformization comes from.

Let's numerically show the equation below by R.

$$e^{Qt} = \sum_{k=0}^{\infty} \tilde{P}^k \frac{(\lambda t)^k e^{-\lambda t}}{k!}.$$

Using the R package "expm" we can compute e^{Qt} directly using the ward approximation.Let's say t=2.

```
t=2
Q=matrix(c(-0.3,0.2,0.1,0,-0.35,0.35,0,0,0),3,3,byrow = T)
```

```r
Ptran=expm::expm(Q*t,method = "Ward77")
print(Ptran)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.5488116 0.2089053 0.2422830
## [2,] 0.0000000 0.4965853 0.5034147
## [3,] 0.0000000 0.0000000 1.0000000
```

For the uniformization method, the jump probability now is

$$\tilde{P} = I + \frac{Q}{\lambda} = \begin{bmatrix} 0.4 & 0.4 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{bmatrix}.$$

Let's compute the transition probability by uniformization equation. We can choose $M_1 = 10$, via the uniformization equation,

```r
M1=10
jump_p=matrix(c(0.4,0.4,0.2,0,0.3,0.7,0,0,1),3,3,byrow = T)
Ptran_unif=0
for (i in 0 : M1){
  Ptran_unif=Ptran_unif+((jump_p)%^%i)*((0.5*2)^i)*exp(-0.5*2)/factorial(i)
}
print(Ptran_unif)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.5488116 0.2089053 0.2422830
## [2,] 0.0000000 0.4965853 0.5034147
## [3,] 0.0000000 0.0000000 1.0000000
```

We can find the results between the package expm and uniformization is the same.

Now, let's choose the smallest $\lambda = 0.35$. Then the jump probability is

$$\tilde{P} = I + \frac{Q}{\lambda} = \begin{bmatrix} 1/7 & 4/7 & 2/7 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Let $M_1 = 10$, we have the same results as package expm.

```
M1=10
jump_p=matrix(c(1/7,4/7,2/7,0,0,1,0,0,1),3,3,byrow = T)
Ptran_unif=0
for (i in 0 : M1){
  Ptran_unif=Ptran_unif+((jump_p)%^%i)*((0.35*2)^i)*exp(-0.35*2)/factorial(i)
}
print(Ptran_unif)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.5488116 0.2089053 0.2422830
## [2,] 0.0000000 0.4965853 0.5034147
## [3,] 0.0000000 0.0000000 1.0000000
```

This numerical example shows the uniformization process step by step. And we can see that the choice of λ has no influence on the computational results of the estimated transition probability. In each iteration of the Newton-Raphson method, we will have an updated estimate of transition intensities. By setting $\lambda = max(q_i)$, we will enter into the next iteration. So we do not need to estimate λ.