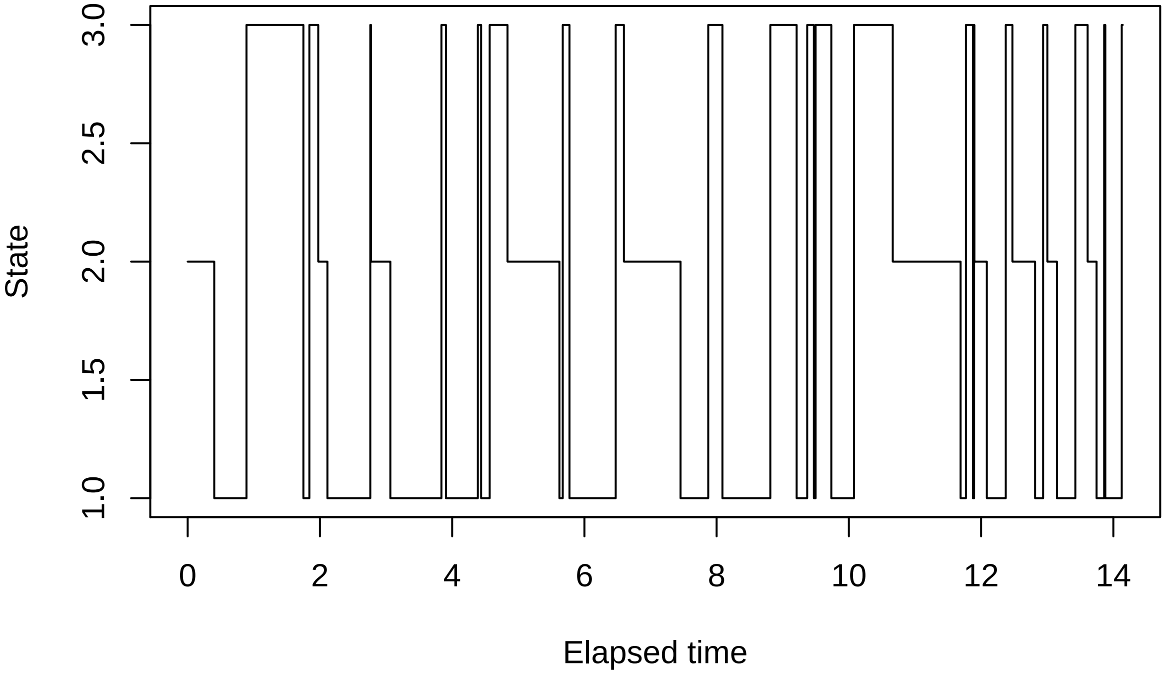
3StateDecayRepair

Chris Kalra

3/6/2019

3 state decay-repair proccess

```
#State 1 = poor, 2 = fair, 3 = good
set.seed(1492)
lam = 3;  mu = 2           # repair and decay rates
kap = c(lam, mu, 2*mu)     # rates for leaving states 1, 2, and 3
m = 50000;  x = t = numeric(m)
x[1] = 2                   # start in state 2
for (i in 2:m)
{
    t[i-1] = rexp(1, kap[x[i-1]])
    if (x[i-1] == 1) x[i] = 3                # moving from 1 to 3
    if (x[i-1] == 2) x[i] = 1                # moving from 2 to 1
    if (x[i-1] == 3) x[i] = sample(1:2, 1)  # moving from 3 to 1 or 2, each with
                                             # equal probability (.5)
}
plot(c(0,cumsum(t)[1:50]), c(x[1],x[1:50]), type="S", ylab = "State", xlab="Elapsed time", main="")
```



```
d = 2*(lam + mu);  p = c(2*mu, lam, lam)/d  # exact p
t.avg = numeric(3)                           # simulated p
for (j in 1:3) {t.avg[j] = sum(t[x==j])/sum(t)}
round(cbind(states=1:3, p, t.avg), 3)
```

```
##      states   p t.avg
## [1,]      1 0.4 0.400
## [2,]      2 0.3 0.301
## [3,]      3 0.3 0.299
```

```
Q = matrix(c(-lam,   0,  lam,
             mu, -mu,  0,
             mu,  mu, -2*mu), byrow=T, nrow=3)
I = diag(3); g = eigen(t(Q+I))$vectors[ ,3]
p = g/sum(g); p                             # produces the p vector
```

```
## [1] 0.4 0.3 0.3
```

```
round(p %*% Q, 5)                           # verification: should produce all 0's
```

```
##      [,1] [,2] [,3]
## [1,]    0    0    0
```