Tutorial for how to use this code with real data sets

This is a tutorial which explains how you can modify the code used for the simulations such that you can directly apply it to other data sets.

What you need first, is the function for data generation (because you'll perform a bootstrap later on!). This function (`sim.csh.const`) can deal with uniform, exponential and administrative right censoring (other types have to be implemented by yourself). Further it's possible to include one binary covariate if you want (we won't do this here). Basically, you don't have to modify this function. The arguments of the function are given below.

```
sim.csh.const <- function(n.indiv,
                          cr.haz = c(),
                          beta = c(),
                          cova.prob,
                          cens = list(model="unif", param=NULL)
)
```

Now specify the number of states and the censoring mechanism. You'll need your dataset with named columns "time" and "obs.event", where "0" stands for censoring.
The log-likelihood function below is for the case of (exponential) right censoring and 3 states, this can easily be adapted (at least the number of states, for another type of censoring distribution the likelihood has to be adapted which is technically not difficult but needs more effort).

```
#
(negative)
log-
likelihood
(for 3
states and
the
censoring
rate)

             loglikelihood <- function(data){
               state1 <- filter(data, obs.event==1)
               state2 <- filter(data, obs.event==2)
               state3 <- filter(data, obs.event==3)
               cens <- filter(data, obs.event==0)
               f <- function(theta){
                 theta1 <- theta[1]
                 theta2 <- theta[2]
                 theta3 <- theta[3]
                 theta4 <- theta[4]
                 -(dim(state1)[1]*log(theta1)-
             theta1*sum(data$obs.time)+dim(state2)[1]*log(theta2)-
```

```
            theta2*sum(data$obs.time)+dim(state3)[1]*log(theta3)-
            theta3*sum(data$obs.time)+dim(cens)[1]*log(theta4)-
            theta4*sum(data$obs.time))
              }
              return(f)
            }
```

We further define a joint log likelihood for both groups (`joint_loglikelihood`). This will be important for the constraint optimization. Further we define the constraints, one for each test, i.e. comparison of states (in this case we have 3 constraints as we compare transition intensities of 3 states). For instance, this constraint is given for the first test by

```
const1 <-
function(theta){
                    theta1 <- theta[1:4]
                    theta2 <- theta[5:8]
                    abs(theta1[1]-theta2[1])-delta1
                  }
```

In order ot gurantee that the estimated parameters fulfill the null hypothesis. Now the preparations are done and we start with the analysis. Putting your dataset into the negative log-likelihood and minimizing this yields the estimates of the transition intensities and your censoring rate.

```
#we
estimate
the
transition
intensities
and the
rates of
censoring
                alpha1 <- optim(par=start,fn=loglikelihood(dat1))$par
                alpha2 <- optim(par=start,fn=loglikelihood(dat2))$par

                # calculating the test statistics
                t.stat1 <- abs(alpha1[1]-alpha2[1])
                t.stat2 <- abs(alpha1[2]-alpha2[2])
                t.stat3 <- abs(alpha1[3]-alpha2[3])

                # the censoring rates
                censoring_rate_est1 <- alpha1[4]
                censoring_rate_est2 <- alpha2[4]
```

Now you have done a big piece of work already, the rest will be easier. Depending on your number of states/comparisons you are interested in, we will now run several tests. In this example we do 3 tests. For illustration, the first test looks like this:

```
###
State
1 ###

          #constrained optimization

          if (t.stat1>=delta1){
            min_cons <- c(alpha1,alpha2)}else{
              min_cons <-
        auglag(par=c(alpha1,alpha2),fn=joint_loglikelihood,heq=const1,control.ou
        ter=list(trace=FALSE))$par
            }

          alpha1_cons <- min_cons[1:3]
          alpha2_cons <- min_cons[5:7]
```

The first "if…" checks whether the test statistic is larger than the threshold. If so, we don't have to do anything else, we are ready for the bootstrap.
If not, we do a constrained optimization- meaning that we minimize the negative log-likelihood under the condition that const1=0. This is because want to generate data under the null hypothesis.
Finally, as we now have derived our parameters, we can use them to run the bootstrap. Therefore we generate data using these parameters and the estimated censoring rates (these are the same as in the original estimation because we don't put any constraints on them). Further this is where the data generation function defined above is needed. For the generated datasets we (again) estimate the transition intensities and the maximum absolute difference between the ones corresponding to State 1 (note that we are still at the test for the first state of three). Repeating this B times yields an approximation of the null distribution and we can use the empirical distribution function of the bootstrap or calculate the corresponding quantile in order to get a test decision. The bootstrap is described in the code below:

```
  # bootstrap (create new datasets with the estimated parameters and estimate
the transition intensities)
  for(k in 1:B){
    dat1b <- sim.csh.const(n.indiv=n1, cr.haz = alpha1_cons, beta = c(0,0,0),
cova.prob=0.5, cens=list("exp",censoring_rate_est1))
    dat2b <- sim.csh.const(n.indiv=n2, cr.haz = alpha2_cons, beta = c(0,0,0),
cova.prob=0.5, cens=list("exp",censoring_rate_est2))

    alpha1b <-
optim(par=c(alpha1_cons,censoring_rate_est1),fn=loglikelihood(dat1b))$par
    alpha2b <-
optim(par=c(alpha1_cons,censoring_rate_est2),fn=loglikelihood(dat2b))$par

    tb <- abs(alpha1b[1]-alpha2b[1])
    boot1[k] <- tb
  }

  # decision rule
  crit_val <- quantile(boot1,alpha)
  #pval <- ecdf(boot)(t.stat1)

  if(t.stat1<crit_val){
    rej1[j]=1
  }else{rej1[j]=0}
```

We now know whether the test for State 1 rejects (i.e. the transition rates to the first states are similar in both groups) or not. In order to find out whether the whole multi-state models are similar, we have to perform this test for all states (in our case for all three states). This is why we need 3 constraints, 3 constrained optimizations and finally 3 bootstraps. The global null hypothesis can only be rejected, if all 3 tests reject, that is, the maximum p-value of all three tests is smaller than the pre-specified significance level alpha.
The more states we consider the longer the procedure will take but in general all numbers of states can be implemented, as well as different censoring mechanisms. In this case the log-likelihood has to be adapted.