

Simple CFR approach

Here is the basic model. We are interested in estimating the CFR among confirmed cases, ρ . Presume that the time from confirmation to report of death (censoring at recovery) follows an exponential distribution:

$$\Pr(Y_i < t) = F(t; \lambda_1)$$

Where Y_i is the time from confirmation to death for person i .

Moreover, the time from confirmation to death or recovery (composite outcome) follows an exponential distribution:

$$\Pr(T_i < t) = G(t; \lambda)$$

Where T_i is the time from confirmation to death or recovery for person i .

Let d_{jv} be the number of reported deaths in location j and calendar date v . Based on the above:

$$E(d_{jv}) = \sum_{k=0}^v n_{jk} \lambda_1$$

where n_{jv} is the number in the risk set on that day (i.e., cumulative number of confirmed cases reported on day v minus the cumulative number of deaths on that day and the cumulative number of people recovered).

In addition,

$$E(n_{jv}) = c_{jv} - \sum_{k=0}^{v-1} n_{jk} \lambda$$

where n_{jv} is the number in the risk set on that day (i.e., cumulative number of confirmed cases reported on day v minus the cumulative number of deaths on that day and the cumulative number of people recovered).

Finally

$$\rho = \int_0^{\infty} \lambda_1 [1 - G(t)] dt$$

.

First, define the stan model.

```
data {
  int <lower=0> T; //the number of time steps included
  int <lower=0> L; //the number of locations we have data from

  real <lower=0> c[T,L]; // cumulative number of confirmed cases reported on each day.
  real <lower=0> r[T,L]; // cumulative number of recovered cases reported on each day.
  int <lower=0> d[T,L]; // cumulative number of deaths reported on each day.
  int <lower=0> n[T,L]; //number of currently infected people on each day.
  int <lower = 0> V; // max infection duration (we should be able to get rid of this)
}

parameters {
  real <lower=0> lambda1; //parameter for time to death distribution
  real <lower=0> lambda; // parameter for time to death or recovery distribution
}

transformed parameters {
```

```

real <lower=0> expected_deaths[T,L]; // cumulative # deaths BY time T
real <lower=0> expected_riskset[T,L]; // # infected at time T
real <lower=0> expected_exit[T,L]; // cumulative # those who have exited due to death or recovery BY

//this can be made more efficient...but for now./
for (j in 1:L) {
  for (t in 1:T) {
    expected_deaths[t,j] = 0.0001;
    expected_exit[t,j] = 0.0001;
    for (k in 1:t) {
      expected_exit[k,j] += n[k,j] * lambda;
      expected_deaths[k,j] += n[k,j] * lambda1;
    }
    expected_riskset[t,j] = c[t,j] - expected_exit[t,j];
    //expected_recovered[t,j] = (expected_exit[t,j] - expected_deaths[t,j]);
  }
}

model {

  //definitely can be made more efficient.
  for (j in 1:L) {
    for (t in 1:T) {
      //target+=poisson_lpmf(d[t,j]|expected_deaths[t,j]) + poisson_lpmf(n[t,j]|expected_riskset[t,j]);
      target+=poisson_lpmf(d[t,j]|expected_deaths[t,j]) + poisson_lpmf(n[t,j]|expected_riskset[t,j]);
    }
  }
  // print(target());
}

//generated quantities {
//real <lower = 0, upper = 1> rho;
// for (v in 1:V){
//   rho += exp(loglambda1) * exp(-exp(loglambda) * v);
//   rho += lambda1 * exp(-lambda * v);

// }
//print(rho);
//rho = integrate_1d(getft, 0, positive_infinity(), {lambda1, lambda}, t);

//} // The posterior predictive distribution"

Prep data.
cfrmdl_res <- sampling(cfrmdl, data=cfrmdl_data,
                      iter=10000, init=1)
chains <- extract(cfrmdl_res)
lambda1 <- chains$lambda1
lambda <- chains$lambda
outdat <- data.frame(lambda1 = lambda1, lambda = lambda)
save(outdat, file = "outdat.RData")

```

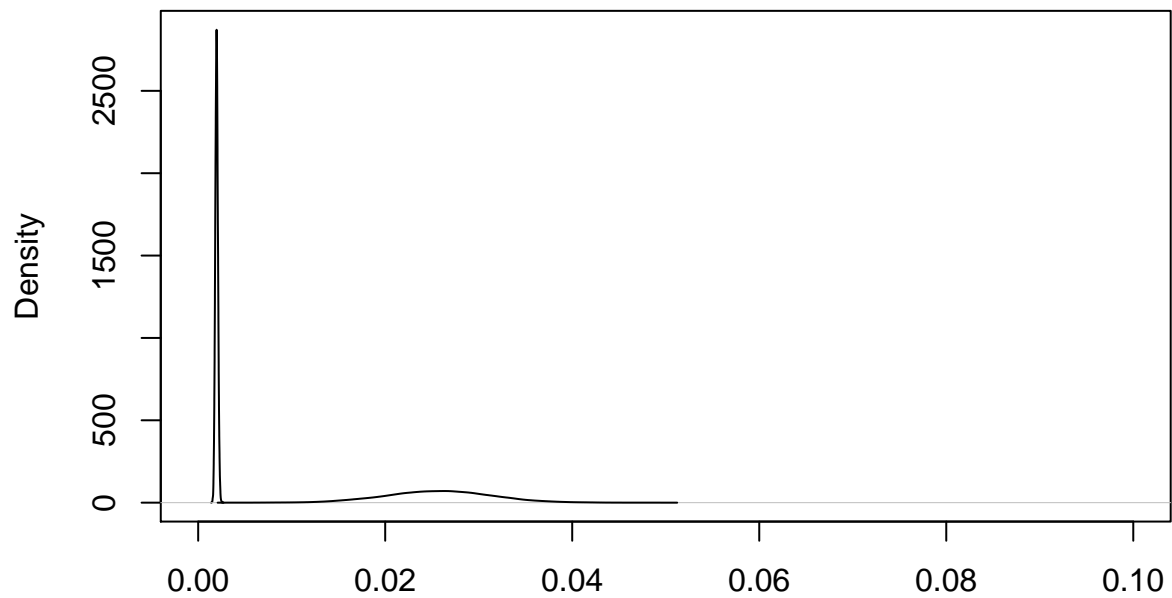
Summarize

```
load("outdat.RData")

lambda1 <- outdat[,1] #chains$lambda1
lambda <- outdat[,2] #chains$lambda

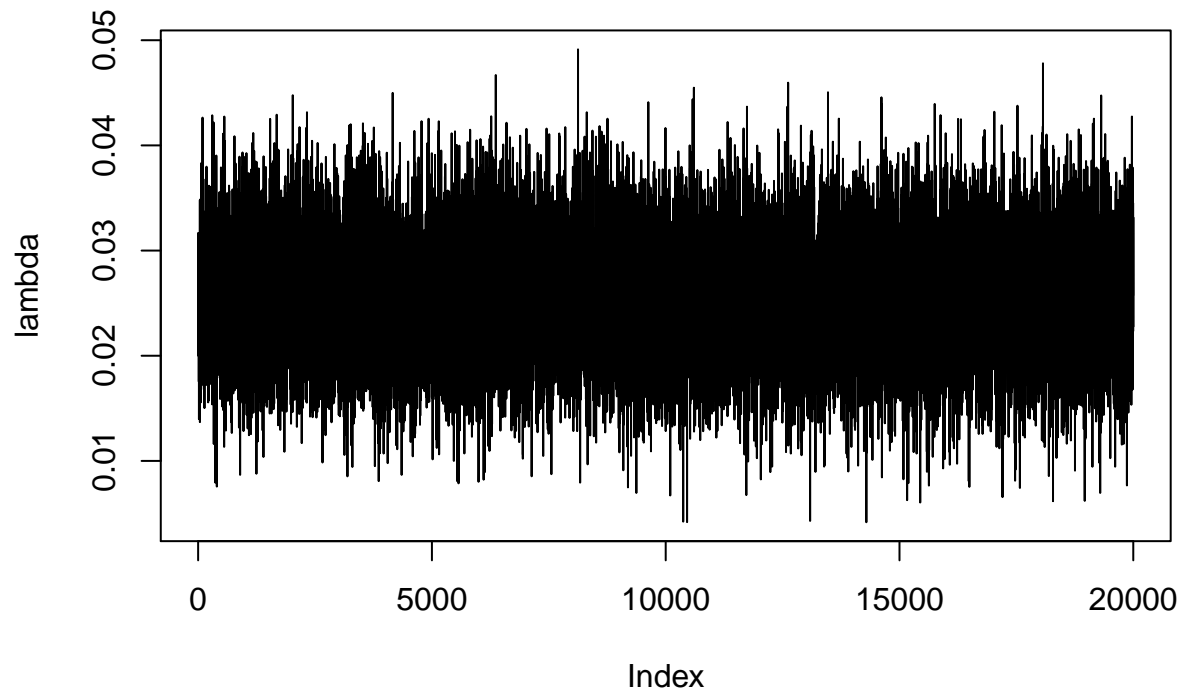
plot(density(lambda1), xlim=c(0,.1), type = "l")
lines(density(lambda))
```

density.default(x = lambda1)

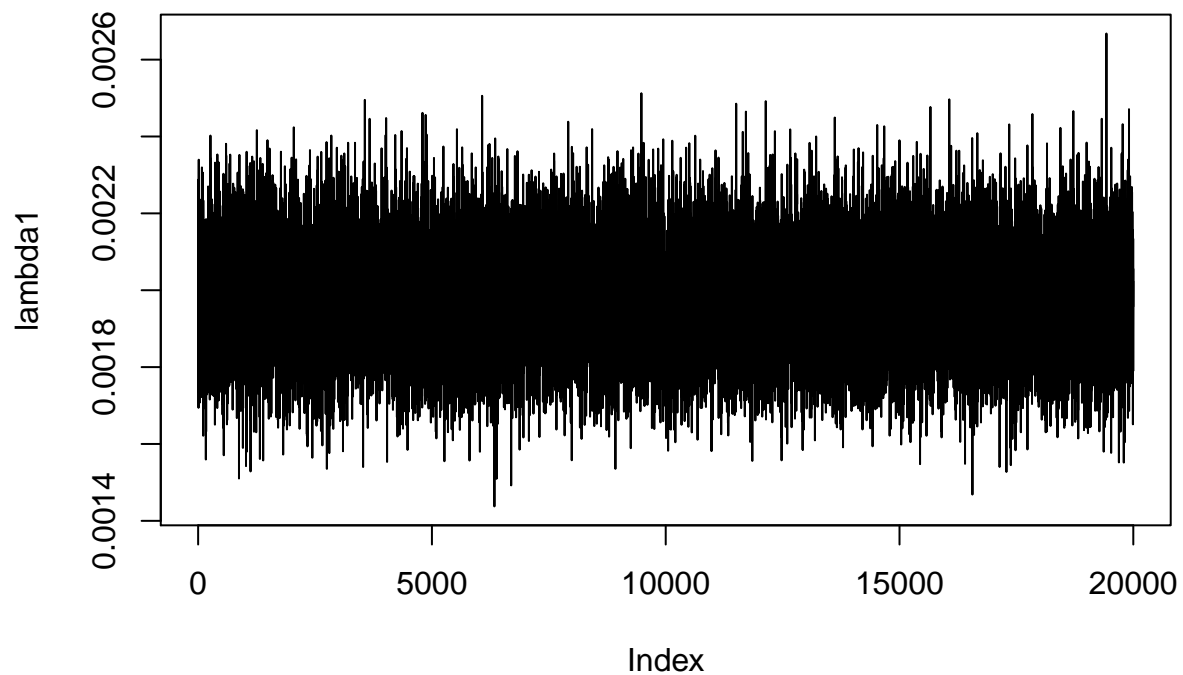


N = 20000 Bandwidth = 1.741e-05

```
plot(lambda, type = "l")
```



```
plot(lambda1, type = "l")
```



```
#move this into stan model later
rho <- matrix(nrow = length(lambda1))
for(i in 1:length(lambda1)){
  rho[i] <- integrate(function(x) {lambda1[i]*exp(-lambda[i]*x)}, lower = 0, upper = Inf)$value
}
```

```
median(rho)
```

```
## [1] 0.07659594
```

```
quantile(rho, probs = c(.025, .975))
```

```
##          2.5%          97.5%  
## 0.05189345 0.13708947
```