MLE

We consider dividing some weighting random variables to the original distance matrix as the process of randomization. That is $d^*_{i,j} = d_{i,j}/w_{i,j}$, where weights $w_{i,j} = e_i + e_j$ have unit-mean of gamma

Alternative we could make $1/w_{i,j}$ having unit-mean.

As an approximation, suppose units i and j are merged into a common cluster if (and only if) $d_{i,j} < c$.

From Bayesian perspective, given the true distance $\Delta_{i,j}$, $d_{i,j}|\Delta_{i,j} \sim \text{Gamma}(a_1, a_1/\Delta_{i,j})$, so that the sampling mean of $d_{i,j}$ is $\Delta_{i,j}$.

Further, the simple analysis would ignore any issues about the d's or Δ's being true distances. The condition for qualifiable distance matrix is the triangle inequality among the pairwise distances, such condition would not affect our clustering results too much. But, the simple analysis might suppose that a-priori

$1/\Delta_{i,j} \sim \text{Gamma}(a_0, d_0)$. The scaling is such that $\text{E}(1/\Delta_{i,j}) = a_0\ d_0$. The posterior, by conjugacy, has $1/\Delta_{i,j}|d_{i,j} \sim \text{Gamma}(a_0 + a_1, d_0 + d_{i,j})$.

Then the posterior probability that i and j should be clustered is the posterior probability that $\Delta_{i,j} < c$, which is ...

We consider estimating those hyper parameters through optimizing the likelihood function.

$$P(d_i j|\Delta_{i,j}) = \frac{(a_1/\Delta_{i,j})^{a_1}}{\Gamma(a_1)} d_{i,j}^{a_1-1} e^{-\frac{a_1}{\Delta_{i,j}} d_{i,j}}$$

$$P(\Delta_{i,j}) = \frac{(d_0)^{a_0}}{\Gamma(a_0)} (1/\Delta_{i,j})^{a_0+1} e^{-d_0/\Delta_{i,j}}$$

And we have the joint pdf

$$P(d_{i,j}, \Delta_{i,j}) = \frac{d_0^{a_0} a_1^{a_1} d_{i,j}^{a_1-1}}{\Gamma(a_0)\Gamma(a_1)} (1/\Delta_{i,j})^{a_1+a_0+1} e^{-(d_0+a_1 d_{i,j})/\Delta i,j}$$

We integral out $\Delta_{i,j}$.

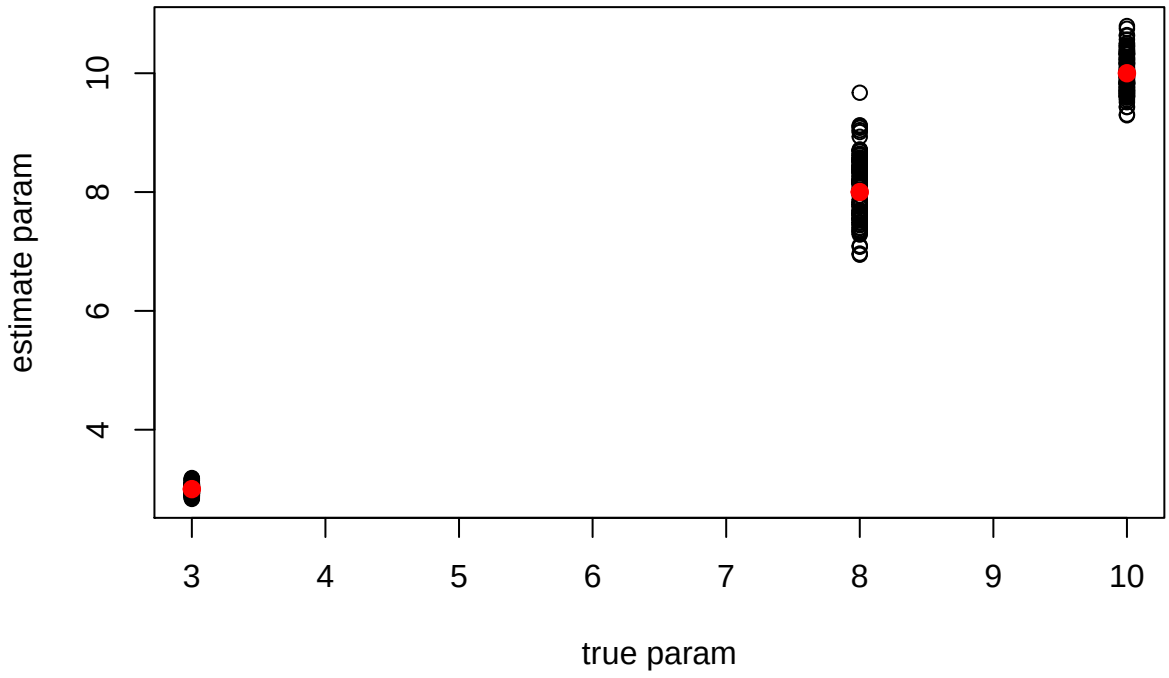
$$P(d_{i,j}|a_0, a_1, d_0) = \frac{\Gamma(a_0+a_1)}{\Gamma(a_0)\Gamma(a_1)} \frac{d_0^{a_0} d_{i,j}^{a_1-1} a_1^{a_1}}{(d_0 + a_1 * d_{i,j})^{a_0+a_1}}$$

Taking log and sum over $d_{i,j}$, we obtain the objective function

$$L = \sum_{i,j} \log(P(d_{i,j}|a_0, a_1, d_0)) = \log(\Gamma(a_0+a_1)) - \log(\Gamma(a_0)) - \log(\Gamma(a_1)) + a_1 \log(d_0) + (a_1 - 1)\log(d_{i,j}) -$$

```
LL = function(param, x){
  a0 = param[1]   #shape for prior
  d0 = param[2]   #rate for prior
  a1 = param[3]   #shape for sampling model

  n = length(x)

  C = d0 + a1 * x
```

```r
  res = n * (lgamma(a0 + a1) - lgamma(a0) - lgamma(a1) + a0 * log(d0) + a1 * log(a1)) + sum((a1 - 1) * l

  return(-res)

}

###true parameter
a0 = 3
d0 = 10
a1 = 8

###optimize N = 100times
N = 100
est.a0 = rep(0,N)
est.d0 = rep(0,N)
est.a1 = rep(0,N)

for(i in 1:100){
  wt <- 1 / rgamma( 10000, shape=a0, rate= d0 )
  dt <- rgamma( 10000, shape=a1, rate=(a1/wt) )
  fit3 <- suppressMessages(nlminb( start=c(1,5,10), objective=LL, x=dt, lower=c(0,0,0) ))
  est.a0[i] = fit3$par[1]
  est.d0[i] = fit3$par[2]
  est.a1[i] = fit3$par[3]
}

plot(c(rep(c(a0,d0,a1),each = N)) ,c(est.a0,est.d0,est.a1),
     xlab = "true param", ylab = "estimate param")
points(a0,a0,col = "red", pch = 19, lwd = 2)
points(d0,d0,col = "red", pch = 19, lwd = 2)
points(a1,a1,col = "red", pch = 19, lwd = 2)
```



“‘