Homework 1

Umbertol Junior Mele 1388371

30 ottobre 2017

Ex. 1

Assume a Dirichlet process (DP) prior, DP(M;G0(·)), for distributions G on X. Show that for any (measurable) disjoint subsets B1 and B2 of X , Corr(G(B1);G(B2)) is negative. Is the negative correlation for random probabilities induced by the DP prior a restriction? Discuss.

A Dirichlet Process is defined in this way:

Let Ω be a space and $\mathbf{A}$ a σ-field of subsets, and let be α be a finite non-null measure on (Ω,$\mathbf{A}$). Then a stocastic process P indexed by elements a of $\mathbf{A}$, is said to be a Dirichlet process on (Ω,$\mathbf{A}$) with parameters α if for any measurable partition (a_1,a_2, ... ,a_k) of Ω, the random vector ($P(a_1)$, $P(a_2)$, ... , $P(a_k)$) has a Dirichlet distribution with parameter ($\alpha(a_1)$, $\alpha(a_2)$, ... , $\alpha(a_k)$). So P may be considered a random probability measure on (Ω,$\mathbf{A}$).

Then taking:$\Omega = \mathbf{R}$, $\mathbf{A} = \mathbf{B}(R)$

Let $G()$ be a function from $R \to [0,1]$, and $G_0(a) = \frac{\alpha(a)}{\alpha(\Omega)}$,(where $M = \alpha(\Omega)$ then we can say that:

$$G \sim DP(M, G_0)$$

So, if we made the partition: $B_1, B_2, (B_1 \cup B_2)^C$, then the $Corr[G(B_1), G(B_2)]$ is:

$$Cov[G(B_1), G(B_2)] = -\frac{G_0(B_1) \cdot G_0(B_2)}{(M+1)}$$

$$Var[G(B_i)] = \frac{G_0(B_i)\Big(1 - G_0(B_i)\Big)}{M+1}$$

$$Corr[G(B_1), G(B_2)] = \frac{Cov[G(B_1), G(B_2)]}{\sqrt{Var[G(B_1)] \cdot Var[G(B_2)]}} \leq 0$$

And this properties is peculiar of Dirichlet process. Infact usually we aspect that for a random probability distribution the masses assigned to nearby places increase or decrease together, and this can be a problem that needs to be keeped in mind... because in Dirichlet Process setup if the observations of B_1 increase, then the observations of B_2, although the sets are close, has to decrease.

Ex. 2

Simulation of Dirichlet process prior realizations. Consider a $DP(M; G_0)$ ***prior over the space of distributions (equivalently c.d.f.s)*** G ***on*** $\mathbf{R}$***, with*** $G_0 = N(0; 1)$***.***

a) To implement Ferguson definition we need to make a partion of R so using intervals like:

$$[-\infty, x_1], [x_1, x_2], ...[x_n, \infty]$$

we are taking disjoint subset from the $\mathbf{B}(R)$ and since we know that $G \sim DP(M, G_0)$; then :

$$(g([-\infty, x_1]), g([x_1, x_2]), ..., g([x_n, \infty])) \sim Dir(M \cdot g_0([-\infty, x_1]), M \cdot g_0([x_1, x_2]), ..., M \cdot g_0([x_n, \infty]))$$

or better:

$$(G(x_1), G(x_2) - G(x_1), ..., 1 - G(x_n)) \sim Dir(M \cdot G_0(x_1), M \cdot (G_0(x_2) - G_0(x_1)), ..., M \cdot (1 - G_0(x_n)))$$

So, using Ferduson^1 definition we can sample from a dirichlet distribution using gammas distributions.

$$Y_i \sim Gamma(1, \alpha(A_1)) \quad for \quad i = 1, 2, ..., k$$

$$S = \sum_{i=1}^{k} Y_i \sim Gamma\big(1, \alpha(\Omega) = M\big)$$

$$Z_i = \frac{Y_i}{S} \quad for \quad i = 1, 2, ..., k$$
$$\mathbf{Z} \sim Dir(\alpha(A_1), \alpha(A_2), ..., \alpha(A_k))$$

```
ferguson.def <- function(number.obs, M){
  sample <- seq(from=-5, to=5, length.out = number.obs)
  par <- rep(0,number.obs)
  sim <- rep(0,number.obs)
  par[1] <- M*pnorm(sample[1])
  sim[1] <- rgamma(1,par[1])
  for(i in 2:number.obs){
    par[i] <- M*(pnorm(sample[i]) - pnorm(sample[i-1]))
    sim[i] <- rgamma(1,par[i])
  }
  tot <- sum(sim)
  ret <- sim/tot
  return(ret)
}

M <- c(5, 20 , 50, 100)
number.obs=100
sim= 100
```

```
a <- list()
for(i in 1:4){
  a[[i]] <- matrix(nrow = sim, ncol = number.obs)
  for(l in 1:sim)
  a[[i]][l,] <- ferguson.def(number.obs,M=M[i])
}
```

Since from Theorem 3 and Theorem 4 of Ferguson-s paper (1973), we know that:

- if $\int |Z| d\alpha < \infty$, then $\int |Z| dP < \infty$ and...

$$E\left[\int Z dP\right] = \int Z dE[P] = \alpha(\Omega)^{-1} \int Z d\alpha$$

- if $\int |Z_1| d\alpha < \infty$, $\int |Z_2| d\alpha < \infty$ and $\int |Z_1 Z_2| d\alpha < \infty$, then:

$$E\left[\int Z_1 dP \int Z_2 dP\right] = \frac{\sigma_{12}}{\alpha(\Omega)+1} + \mu_1 \mu_2$$

where:

$$\mu_i = \alpha(\Omega)^{-1} \int Z_i d\alpha$$

$$\sigma_{12} = \alpha(\Omega)^{-1} \cdot \int Z_1 Z_2 d\alpha - \mu_1 \mu_2$$

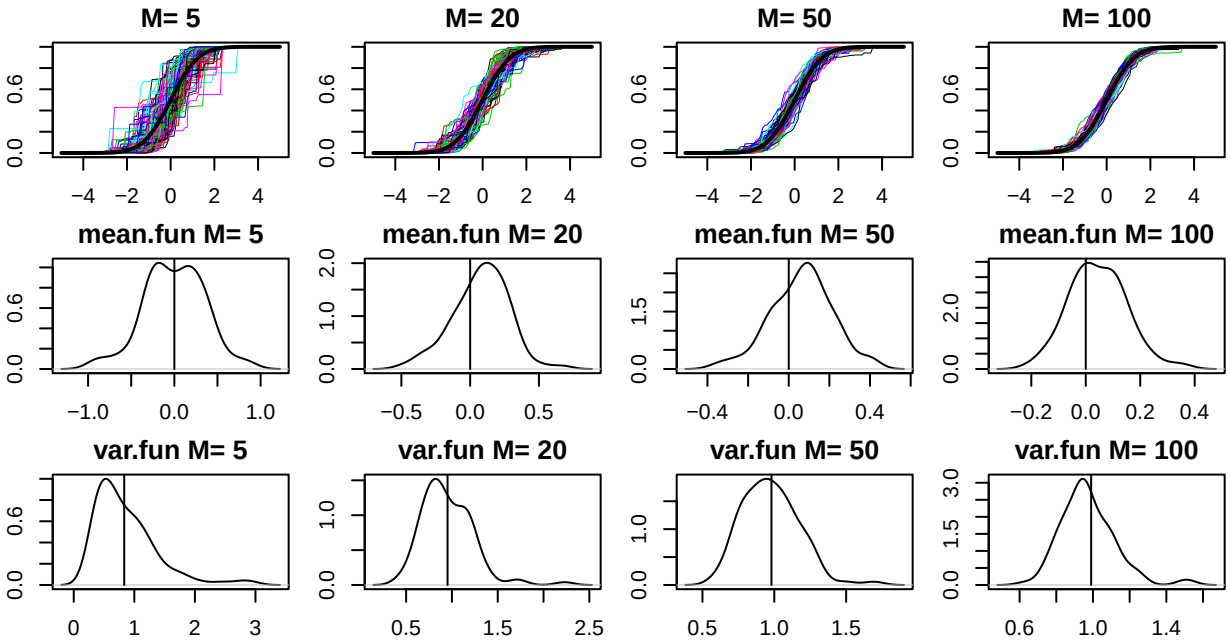
Then we can use this results to estimate the mean and the Variance, for prior realization for our case:

$$E\left[\int t dP(t)\right] = M^{-1} \int t dM G_0(t) = M^{-1} \cdot M \int t dG_0(t) = 0$$

$$E\left[VarP\right] = E[\int t^2 dP(t)] - \left(E[\int t dP(t)]\right)^2 = (\sigma_0^2 + \mu_0^2) - \left(\frac{\sigma_0^2}{M+1} + \mu_0^2\right) = \frac{M}{M+1}\sigma_0^2 = \frac{M}{M+1}$$

```
sample <- seq(from=-5, to=5, length.out = number.obs)
par(mfrow=c(4,4), mar=c(2,2,2,1))
for(i in 1:4){
  el <- apply(a[[i]],1,cumsum )
  matplot(sample,el, type = 'l', lwd = 0.00001, lty=3,main = paste('M=', M[i]))
  curve(pnorm, add=T, col='black', lwd=2)
}

mu.fun <- list(rep(0,sim),rep(0,sim),rep(0,sim),rep(0,sim))
var.fun <- list(rep(0,sim),rep(0,sim),rep(0,sim),rep(0,sim))
for(m in 1:4){
  for(i in 1:sim){
    mu.fun[[m]][i] <- sample%*%a[[m]][i,]
    var.fun[[m]][i] <- (sample)^2%*%a[[m]][i,] - (sample%*%a[[m]][i,])^2
  }
  plot(density(mu.fun[[m]]), main = paste('mean.fun M=', M[m]))
  segments(x0 = 0, y0 = 0,x1 = 0,y1 = 10)
}
```

```
for(m in 1:4){
  plot(density(var.fun[[m]]), main = paste('var.fun M=', M[m]))
  segments(x0 = (M[m]/(M[m]+1)), y0 = 0,x1 = (M[m]/(M[m]+1)),y1 = 10)
}
```



b) While using Sethuraman construction:

$$Y_i \sim G_0$$

$$\theta_i \sim Beta(1, M)$$

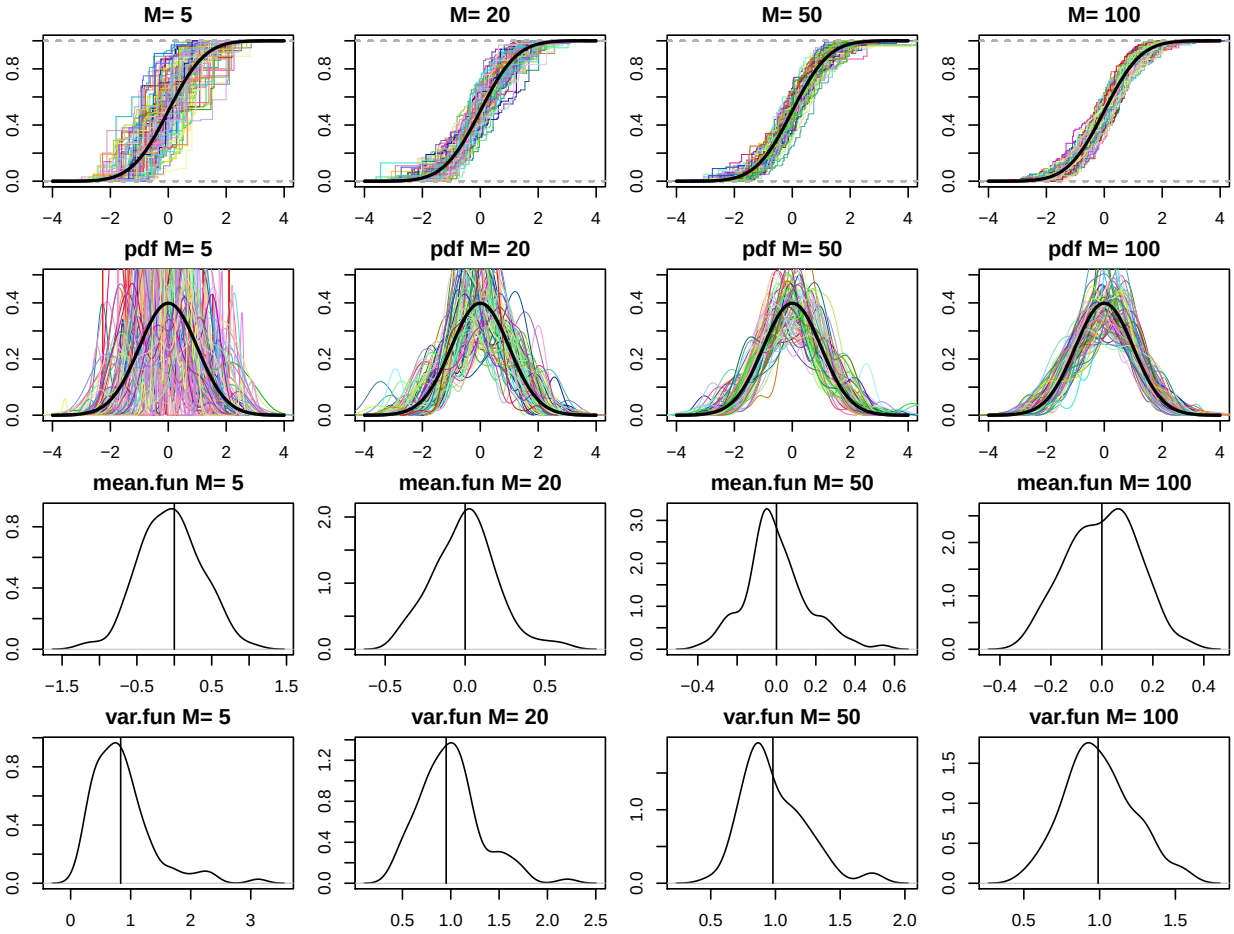
$$\pi_1 = \theta_1$$

$$\pi_i = \theta_i \prod_{j=1}^{i-1}(1 - \theta_i) \quad for \quad i \geq 2$$

$$G(A) = \sum_{1}^{\infty} \pi_i \cdot \delta_{Y_i}(A) \sim DP(M, G_0)$$

```
sethuraman.cost <- function(number.obs, M){
  n <- 5000
  y <- rnorm(n)
  thet <- rbeta(n,shape1 = 1, shape2 = M)
  prob <- rep(0,n)
  prob[1] <- thet[1]
  for(i in 2:n){
    prob[i]<- thet[i]*prod(1 - thet[1:i-1])
  }
  dat <- sample(y,size= number.obs, prob=prob,replace=T)
  return(dat)
}

M <- c(5, 20 , 50, 100)
number.obs=100
sim= 100
a <- list()
for(i in 1:4){
  a[[i]] <- matrix(nrow = sim, ncol = number.obs)
  for(l in 1:sim)
    a[[i]][l,] <- sethuraman.cost(number.obs,M=M[i])
}
```

```
sample <- seq(from=-5, to=5, length.out = number.obs)
par(mfrow=c(4,4), mar=c(2,2,2,1))
for(i in 1:4){
  curve(pnorm, col='black', lwd=2, from = -4, to=4,main = paste('M=', M[i]))
  for(l in 1:100){
    plot(ecdf(a[[i]][l,]),verticals=TRUE, do.points=FALSE,
         lwd = 0.00001, lty=1, add=T,col=randomColor())
  }
  curve(pnorm, add=T, col='black', lwd=2)
}
for(i in 1:4){
  curve(dnorm, col='black', lwd=2, from = -4, to=4,main =paste('pdf M=', M[i]),
        ylim=c(0,.5))
  for(l in 1:100){
    lines(density(a[[i]][l,]), lwd = 0.00001, lty=1
          ,col=randomColor())
  }
  curve(dnorm, add=T, col='black', lwd=2)

}

mu.fun <- list(rep(0,sim),rep(0,sim),rep(0,sim),rep(0,sim))
var.fun <- list(rep(0,sim),rep(0,sim),rep(0,sim),rep(0,sim))
for(m in 1:4){
  for(i in 1:sim){
    mu.fun[[m]][i] <- mean(a[[m]][i,])
    var.fun[[m]][i] <- var(a[[m]][i,])
  }
  plot(density(mu.fun[[m]]), main = paste('mean.fun M=', M[m]))
  segments(x0 = 0, y0 = 0,x1 = 0,y1 = 10)
}

for(m in 1:4){
  plot(density(var.fun[[m]]), main = paste('var.fun M=', M[m]))
  segments(x0 = (M[m]/(M[m]+1)), y0 = 0,x1 = (M[m]/(M[m]+1)),y1 = 10)
}
```

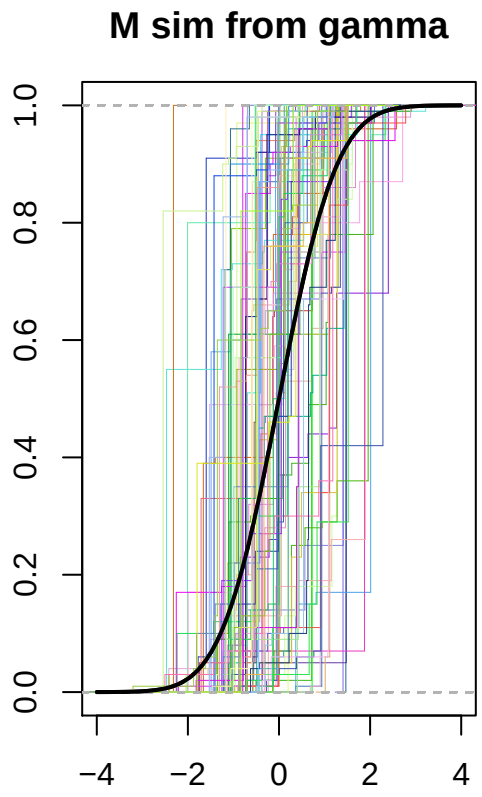

M= 5
M= 20
M= 50
M= 100
pdf M= 5
pdf M= 20
pdf M= 50
pdf M= 100
mean.fun M= 5
mean.fun M= 20
mean.fun M= 50
mean.fun M= 100
var.fun M= 5
var.fun M= 20
var.fun M= 50
var.fun M= 100

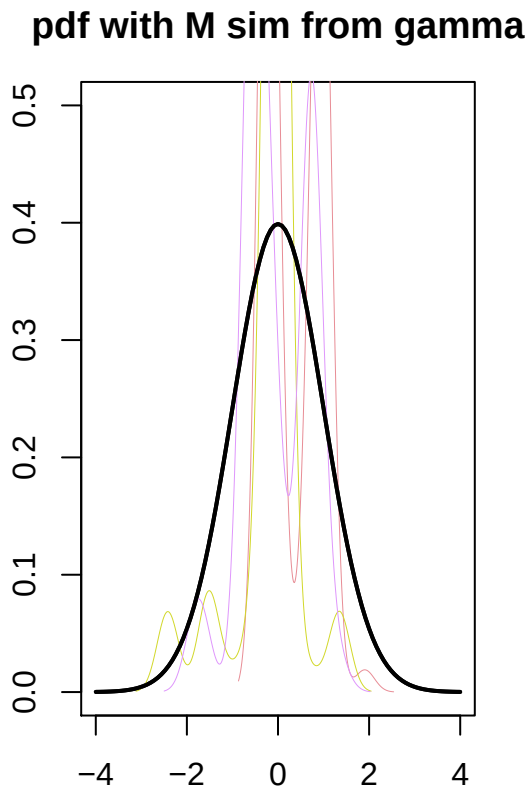
c) Finally, the simulation under a mixture of DPs prior, using a gamma prior for M, that is $M \sim Gamma(3,3)$, that has $E[M] = 1$, and $Var[M] = \frac{1}{3}$.

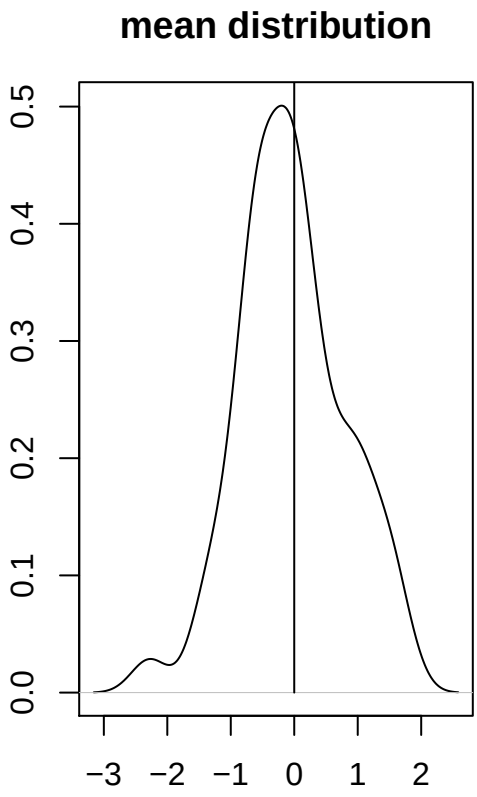
$$M \sim Gamma(3,3)$$
$$G|M \sim DP(M, G_0)$$

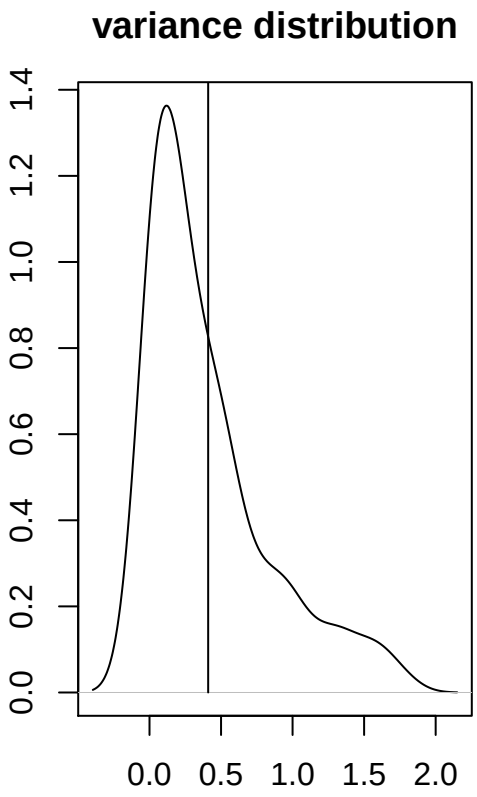
Of course, we can play with the hyperparameters to make stronger believe on the prior distribution.

```
sim = 100
G <- matrix(nrow = sim,ncol = number.obs)
for(t in 1:sim){
  M <- rgamma(1,3,3)
  G[t,] <- sethuraman.cost(number.obs,M)
}
```









From the last plot, we can see that p.d.f. are really strange and this result comes from the fact that sampling from $Gamma(3, 3)$, some values of M can be less than 1 and this is the reason of such particular p.d.f sampled from the DP. In this case, furthermore, is hard to compute analitically the mean of the mean functional and variance functional, since M is a random variable and not more a fixed value... so I used a sample estimator.

Ex.3

Posterior inference for one-sample problems using DP priors.

1) Simulation for a $N(0,1)$.

Since from theory we know that:

$$G|Y_1, Y_2, ..., Y_n \sim DP\left(M+n, \frac{1}{M+n}(M \cdot G_0 + \sum_{i=1}^{n} \delta_{Y_i})\right)$$

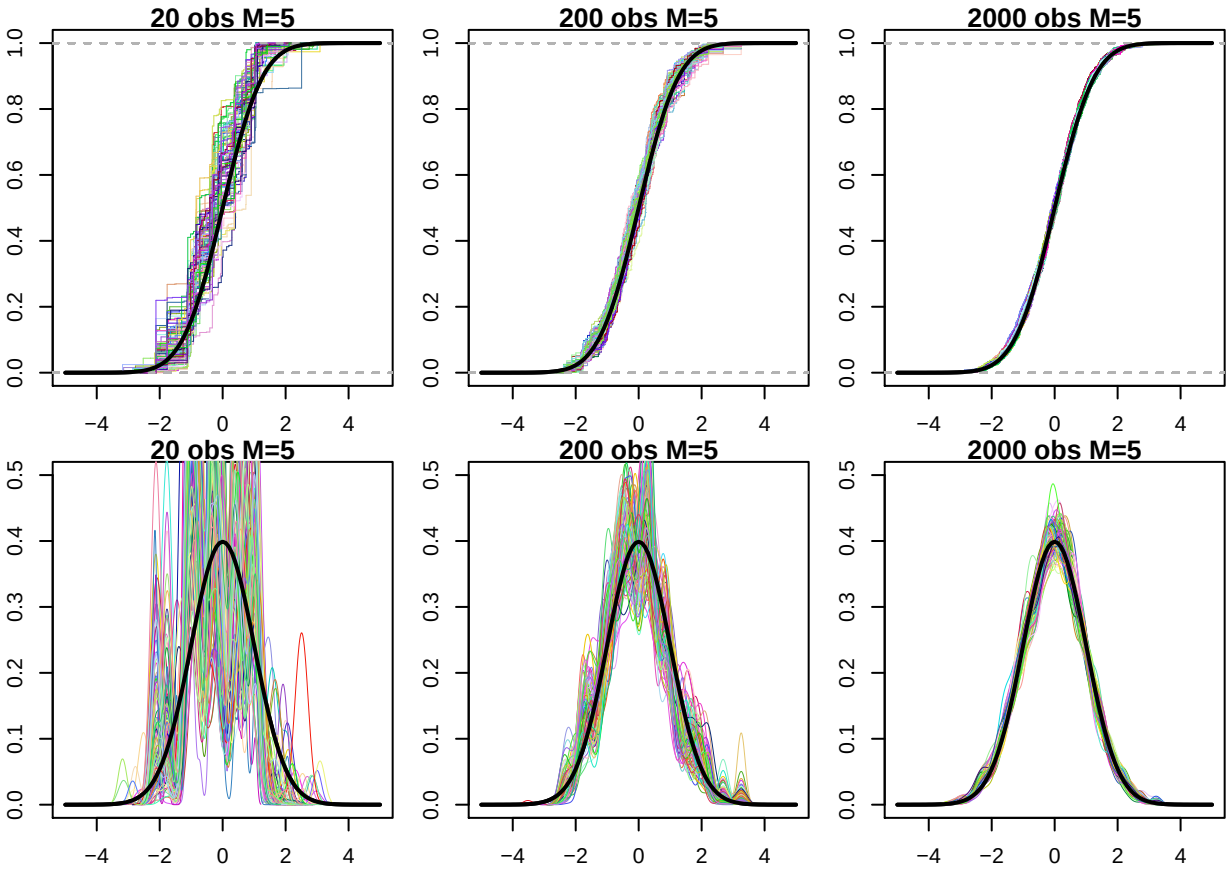
then using Sethuraman contraction we can sampling from the posterior measure $\frac{1}{M+n}(M \cdot G_0 + \sum_{i=1}^{n} \delta_{Y_i})$ implementing the *Pòlya urn and the Chinese Restaurant Process.*

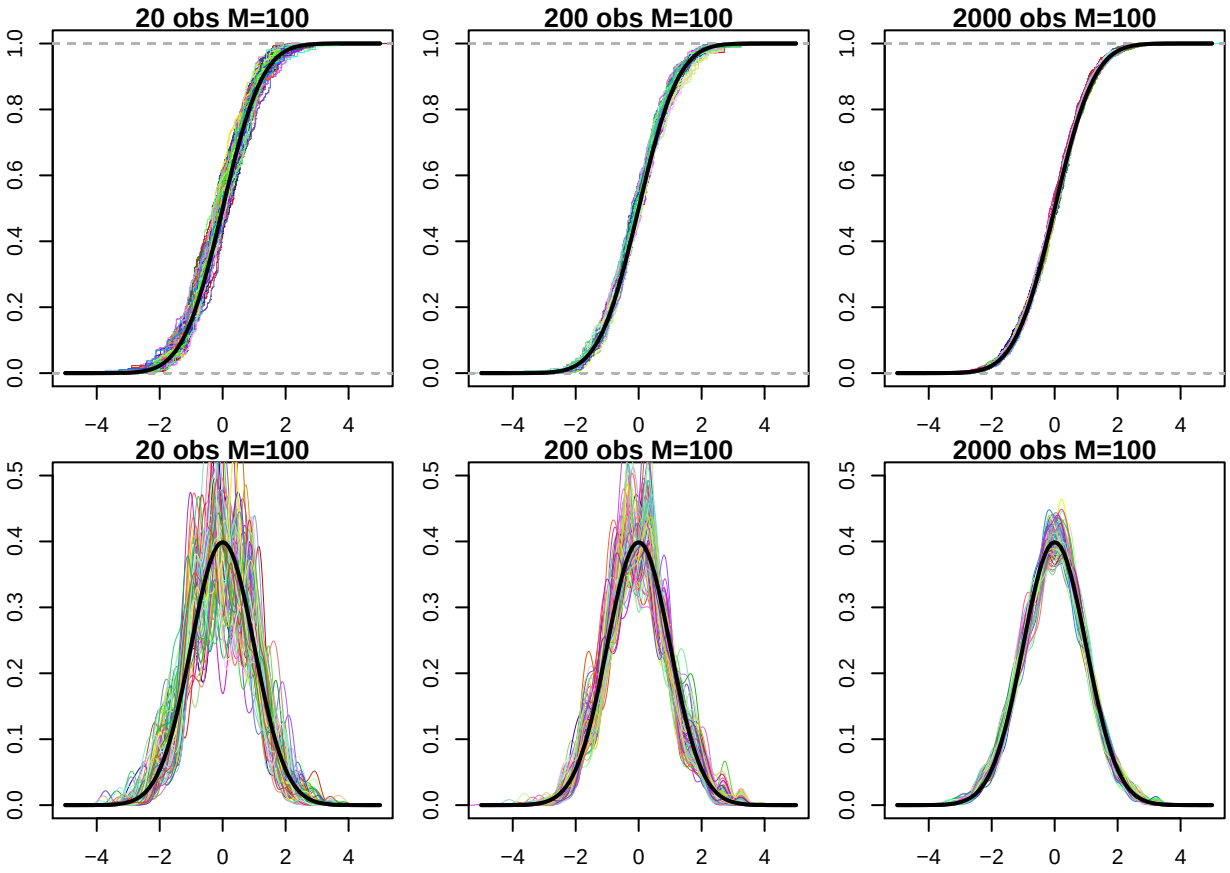
```
chinese.rest <- function(dati, M, mu, s){
  pr= M/(M+length(dati))
  sim <- rep(0,5000)
  for(i in 1:5000){
    do <- rbinom(1,1,pr)
    if(do){
      sim[i] <- rnorm(1,mu,s)
    }else{
      sim[i] <- sample(dati, size=1, replace=T)
    }
  }
  thet <- rbeta(5000,shape1 = 1, shape2 = M+length(dati))
  prob <- rep(0,5000)
  prob[1] <- thet[1]
  for(i in 2:5000){
    prob[i]<- thet[i]*prod(1 - thet[1:i-1])
  }
  dat <- sample(sim, size= 5000, prob=prob,replace=T)
  return(dat)
}
```

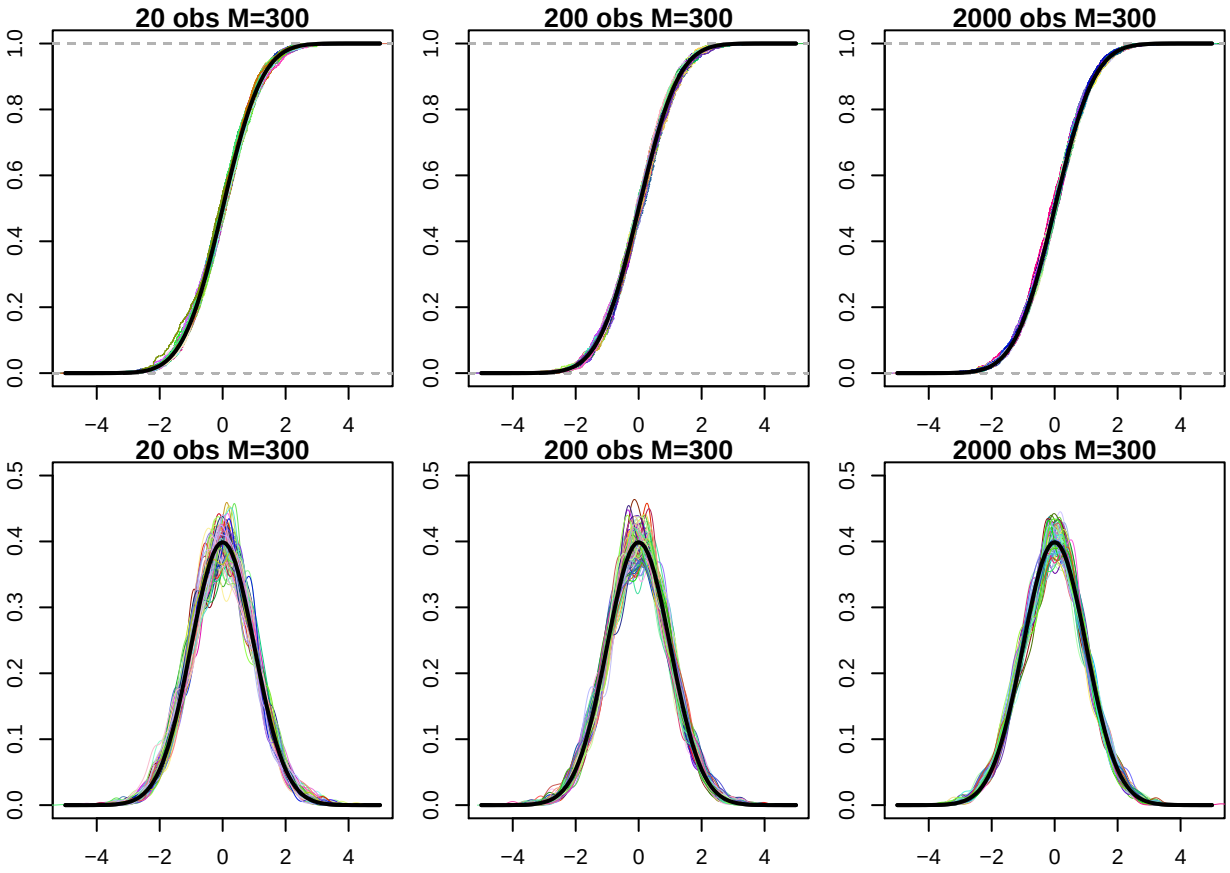
```
# simulation

s.20 <- rnorm(20)
s.200 <- rnorm(200)
s.2000 <- rnorm(2000)


sim.20 <- matrix(nrow = 100, ncol = 5000)
sim.200 <- matrix(nrow = 100, ncol = 5000)
sim.2000 <- matrix(nrow = 100, ncol = 5000)
for(s in 1:100){
  sim.20[s,]<- chinese.rest(s.20,M= 10,0,1)
  sim.200[s,]<- chinese.rest(s.200,M= 10,0,1)
  sim.2000[s,]<- chinese.rest(s.2000,M= 10,0,1)
}
```

```
cases <- list(sim.20,sim.200,sim.2000)
casi <- c('20 obs','200 obs','2000 obs')
```

```r
par(mfrow=c(2,3), mar=c(2,2,1,1))
for(m in 1:3){
  curve(pnorm,  col='black', lwd=2, from = -5,to=5,main = paste(casi[m], 'M=5'))
  for(i in 1:100){
    plot(ecdf(cases[[m]][i,]),verticals=TRUE, do.points=FALSE, lwd = 0.00001, lty=1
         , add=T,col=randomColor())
  }
  curve(pnorm,  col='black', lwd=2, from = -5,to=5, add=T)

}

for(m in 1:3){
  curve(dnorm,  col='black', lwd=2, from = -5,to=5,main = paste(casi[m], 'M=5'),
        ylim=c(0,.5))
  for(i in 1:100){
    lines(density(cases[[m]][i,]), lwd = 0.00001,
          lty=1,col=randomColor())
  }
  curve(dnorm,  col='black', lwd=2, from = -5,to=5, add=T)
}
```



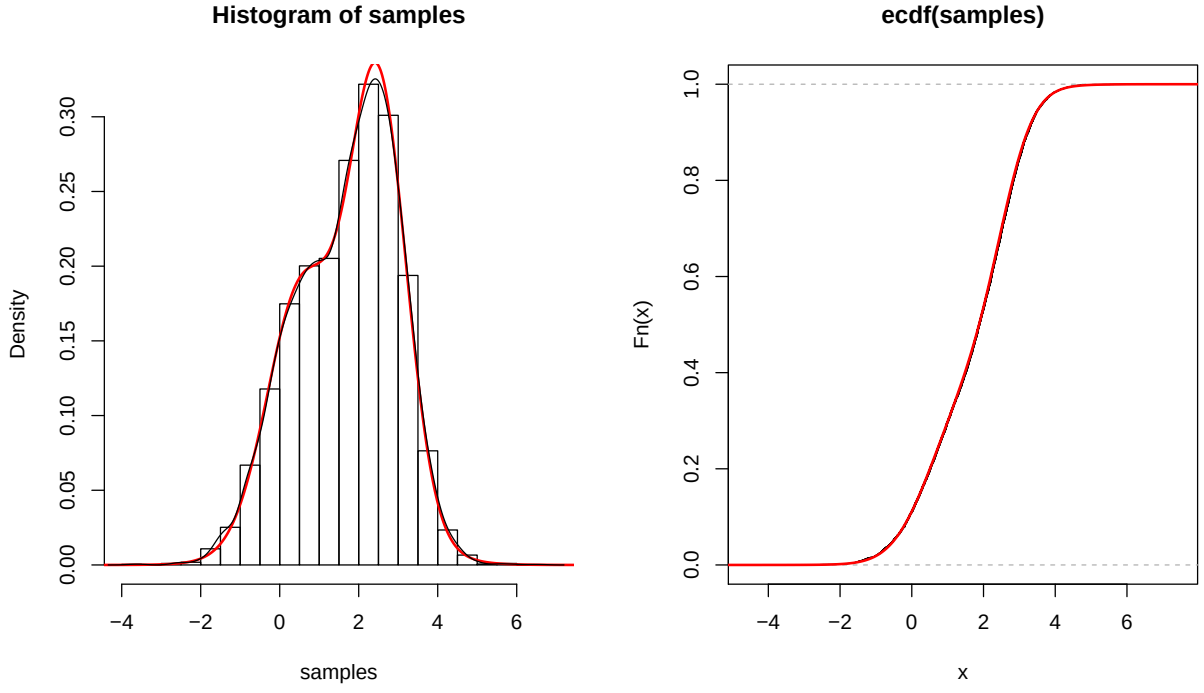

20 obs M=100
200 obs M=100
2000 obs M=100
20 obs M=100
200 obs M=100
2000 obs M=100


20 obs M=300
200 obs M=300
2000 obs M=300
20 obs M=300
200 obs M=300
2000 obs M=300

2) Simulation for a mixture of normal distribution.

$$0.5 \cdot N(2.5, 0.5^2) + 0.3 \cdot N(0.5, 0.7^2) + 0.2 \cdot N(1.5, 2^2)$$

```r
# simulation of data ------------------------------------------------------
N <- 10000
components <- sample(1:3,prob=c(0.5,0.3,0.2),size=N,replace=TRUE)
mus <- c(2.5,0.5,1.5)
sds <- sqrt(c(0.5,0.7,2))

samples <- rnorm(n=N,mean=mus[components],sd=sds[components])

par(mfrow=c(1,2))
hist(samples, probability = T)
x <- seq(-20,20,0.01)

truth <- 0.5*dnorm(x,mean=mus[1],sd=sds[1])+
  0.3*dnorm(x,mean=mus[2],sd=sds[2])+0.2*dnorm(x,mean=mus[3],sd=sds[3])
lines(x,truth,col="red",lwd=2)
lines(density(samples), type = 'l', col='black')

truth.cdf <- 0.5*pnorm(x,mean=mus[1],sd=sds[1])+
  0.3*pnorm(x,mean=mus[2],sd=sds[2])+0.2*pnorm(x,mean=mus[3],sd=sds[3])
plot(ecdf(samples))
lines(x,truth.cdf,col="red",lwd=2)
```



So, using this simulation algorithm, we can create our sample of size:{20,200,2000}

Since from theory we know that:

$$G|Y_1, Y_2, ..., Y_n \sim DP\left(M+n, \frac{1}{M+n}(M \cdot G_0 + \sum_{i=1}^{n} \delta_{Y_i})\right)$$

then using Sethuraman contraction we can sampling from the posterior mesure $\frac{1}{M+n}(M \cdot G_0 + \sum_{i=1}^{n} \delta_{Y_i})$ implementing the **Pòlya urn and the Chinese Restaurant Process**

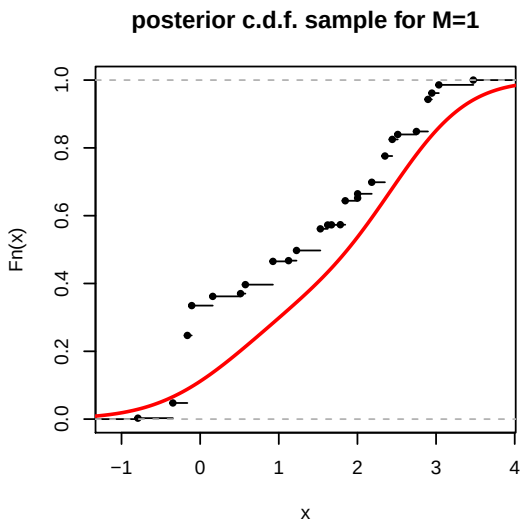
The best non-informative values for m and s^2 are the sample mean and the sample variance, so I choose to start with this hyperparameters.

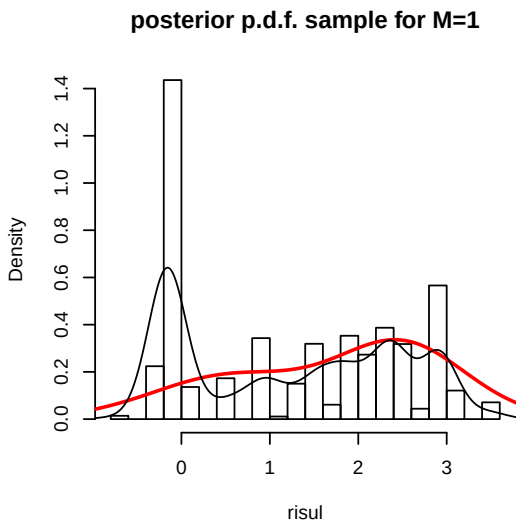
While the choose of M is low since we know that the true distribution is a mixture of models!

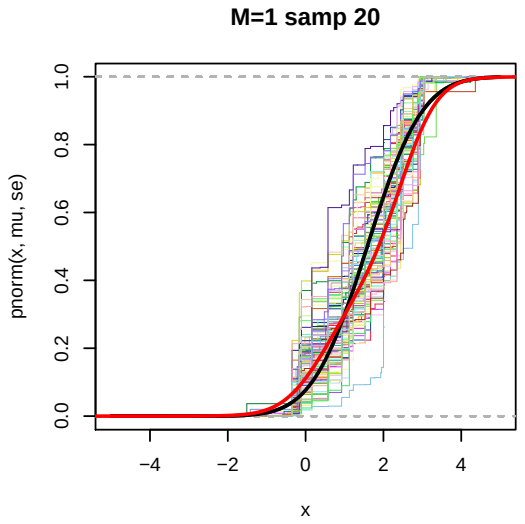
So for small values of M the model will fit the data aproximately well if we have enough observed data.

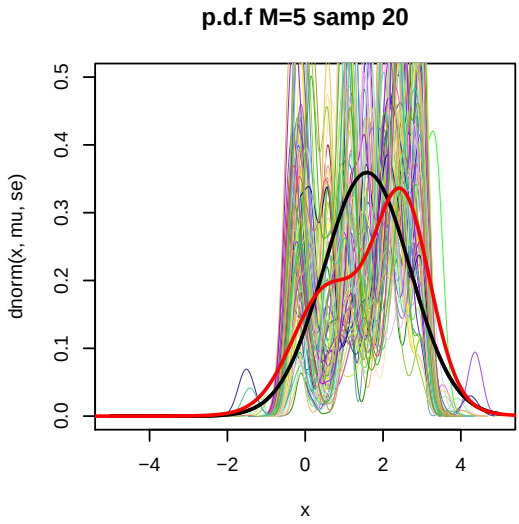
```
mu <- mean(samples.20)
se <- sd(samples.20)

risul <- chinese.rest(samples.20, M = 1, mu = mu, s = se)

par(mfrow=c(3,2))
plot(ecdf(risul), main = 'posterior c.d.f. sample for M=1')
lines(x,truth.cdf,col="red",lwd=2)
hist(risul, probability = T, breaks = 20, main = 'posterior p.d.f. sample for M=1')
lines(x,truth,col="red",lwd=2)
lines(density(risul))

load(file = 'simulation_for_m1_20.RData')

curve(pnorm(x,mu, se),  col='black', lwd=2, from = -5,to=5,main = 'M=1 samp 20')
for(i in 1:100){
  plot(ecdf(samp[i,]),verticals=TRUE, do.points=FALSE, lwd = 0.00001, lty=1
       , add=T,col=randomColor())
}
curve(pnorm(x,mu, se),  col='black', lwd=2, from = -5,to=5, add=T)
lines(x,truth.cdf,col="red",lwd=2)

curve(dnorm(x,mu, se),  col='black', lwd=2, from = -5,to=5,main =  'p.d.f M=5 samp 20',
      ylim=c(0,.5))
for(i in 1:100){
  lines(density(samp[i,]), lwd = 0.00001,
        lty=1,col=randomColor())
}
curve(dnorm(x,mu, se),  col='black', lwd=2, from = -5,to=5, add=T)
lines(x,truth,col="red",lwd=2)

mu.fun <- rep(0,sim)
var.fun <- rep(0,sim)

for(i in 1:sim){
  mu.fun[i] <- mean(samp[i,])
  var.fun[i] <- var(samp[i,])
}
```

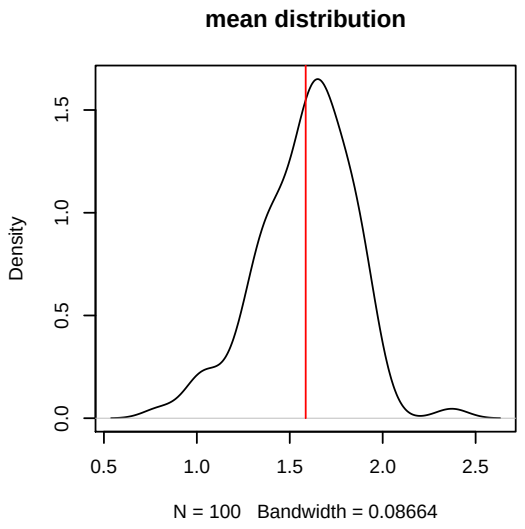
```r
plot(density(mu.fun), main = 'mean distribution')
segments(x0 = mu,y0 = 0,x1 = mu,y1 = 10, col='red')
plot(density(var.fun), main = 'variance distribution')
segments(x0 = se^2,y0 = 0,x1 = se^2,y1 = 10, col = 'red')
```

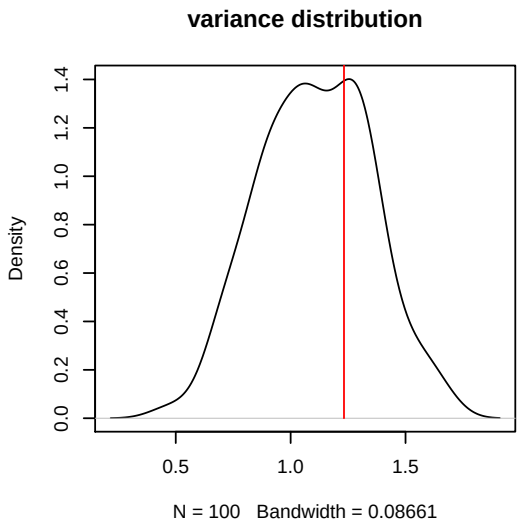

posterior c.d.f. sample for M=1
Fn(x)
x

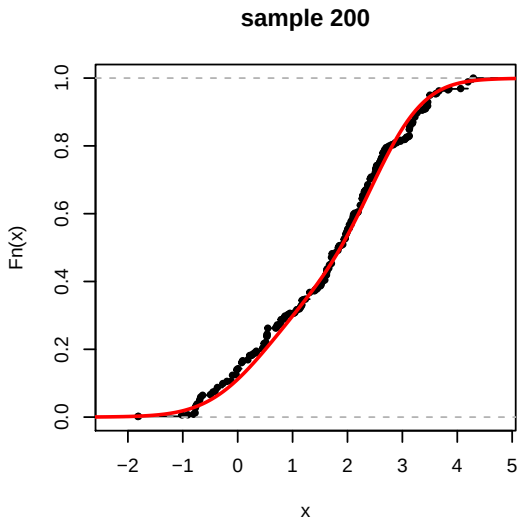

posterior p.d.f. sample for M=1
Density
risul

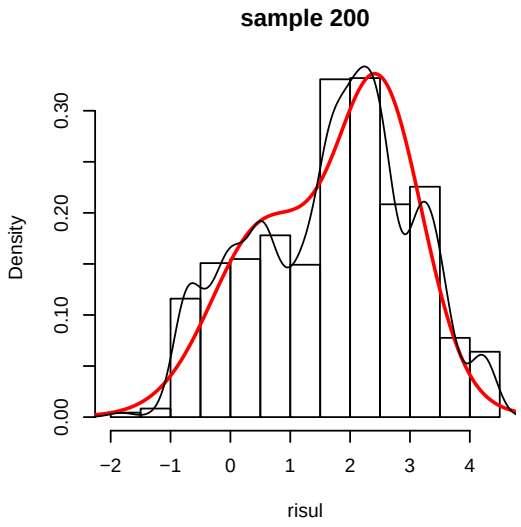

M=1 samp 20
pnorm(x, mu, se)
x

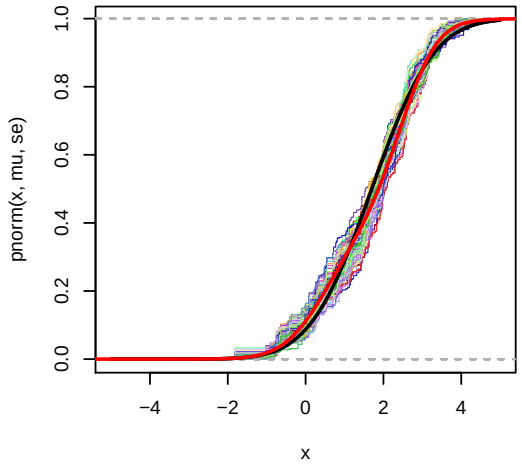

p.d.f M=5 samp 20
dnorm(x, mu, se)
x

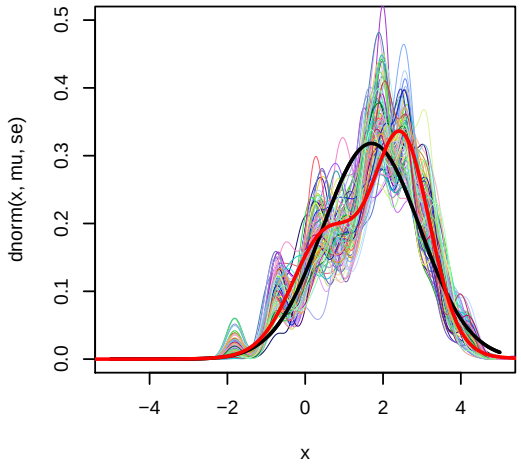

mean distribution
Density
N = 100 Bandwidth = 0.08664

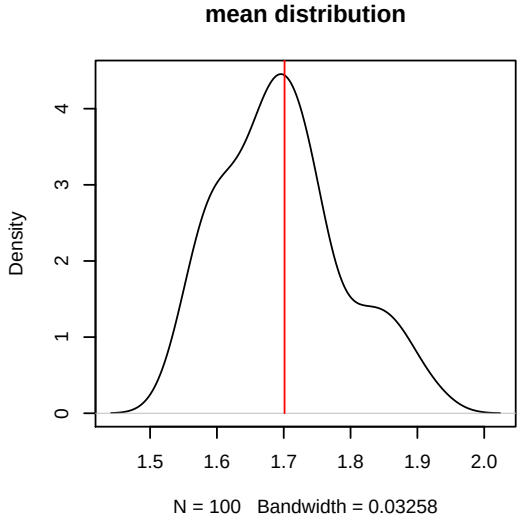

variance distribution
Density
N = 100 Bandwidth = 0.08661

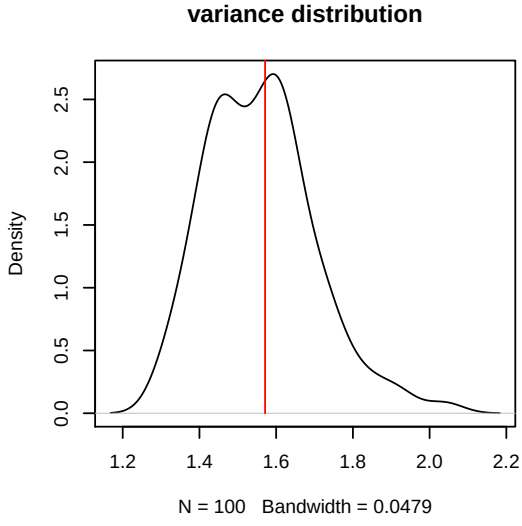

sample 200
Fn(x)
x

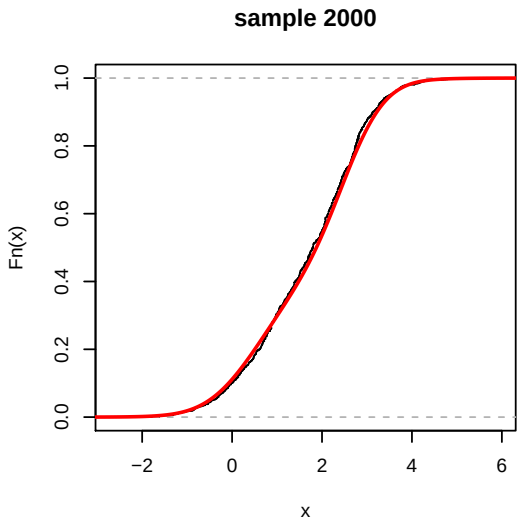

sample 200
Density
risul

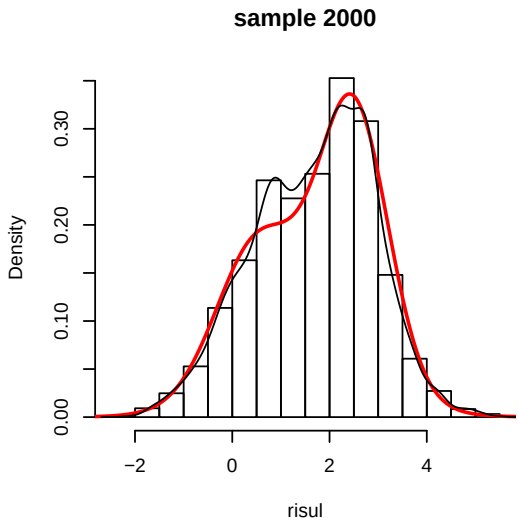

M=1 samp 200
pnorm(x, mu, se)
x

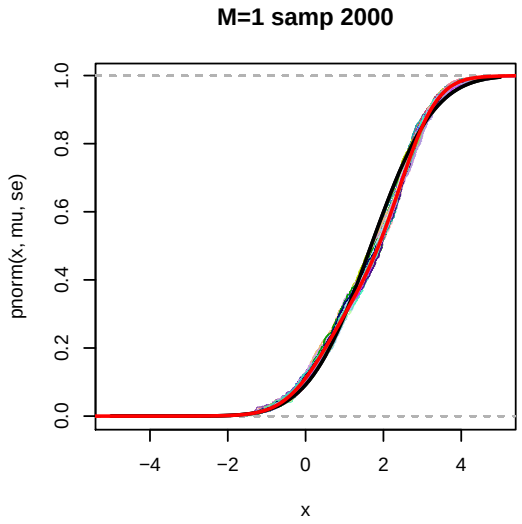

p.d.f M=5 samp 200
dnorm(x, mu, se)
x

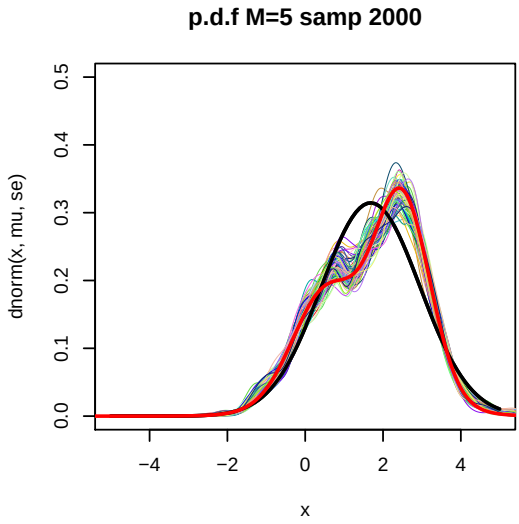

mean distribution
Density
N = 100 Bandwidth = 0.03258

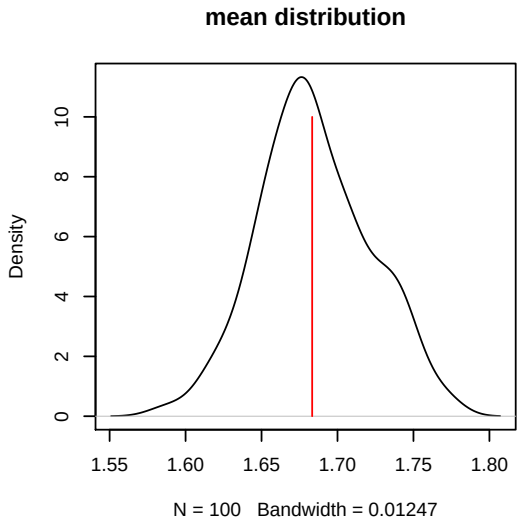

variance distribution
Density
N = 100 Bandwidth = 0.0479

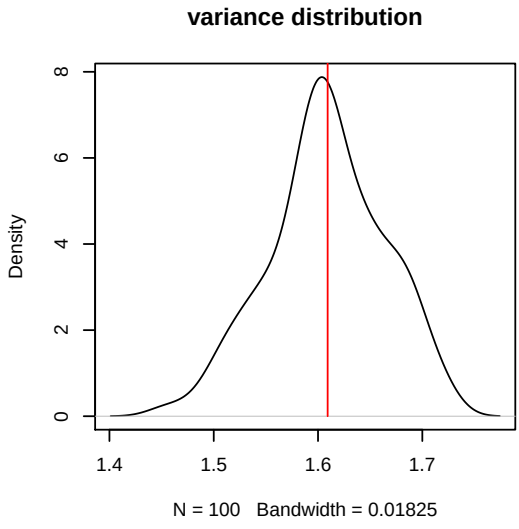

sample 2000
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
−2
0
2
4
6
x


sample 2000
Density
0.00
0.10
0.20
0.30
−2
0
2
4
risul


M=1 samp 2000
pnorm(x, mu, se)
0.0
0.2
0.4
0.6
0.8
1.0
−4
−2
0
2
4
x


p.d.f M=5 samp 2000
dnorm(x, mu, se)
0.0
0.1
0.2
0.3
0.4
0.5
−4
−2
0
2
4
x


mean distribution
Density
0
2
4
6
8
10
1.55
1.60
1.65
1.70
1.75
1.80
N = 100 Bandwidth = 0.01247


variance distribution
Density
0
2
4
6
8
1.4
1.5
1.6
1.7
N = 100 Bandwidth = 0.01825

Reference

- 1 *A Bayesian Analysis of some nonparametric problems*, by Thomas S. Ferguson.
- 2 *A constructive definition of Dirichlet Priors*, by Jayaram Sethuraman
- 3 *Bayesian Nonparametrics*, by Nils Lid Hjort, Chris Holmes, Peter Muller and Sthephen G. Walker.
- 4 *A simple proof of the Stick-Breacking construction of Dirichlet Process*, by John Paisley.