STA211 - sujet 1
Estimation d'une taille de population à partir de données de capture-marquage-recapture

Anthony Kalaydjian - Mathieu Occhipinti

2023-05-03

Vraissemblance du modèle

La vraissemblance du modèle $\mathcal{M}$ s'écrit comme suit :

$$\begin{aligned}
[C_1 = c_1, C_{20} = c_{20}, C_{21} = c_{21} | \pi, N] &= [C_1 = c_1 | \pi, N]\,[C_{20} = c_{20}, | \pi, N, C_1 = c_1]\,[C_{21} = c_{21} | \pi, N, C_1 = c_1, C_{20} = c_{20},] \\
&= [C_1 = c_1 | \pi, N]\,[C_{20} = c_{20}, | \pi, N, C_1 = c_1]\,[C_{21} = c_{21} | \pi, N, C_1 = c_1] \\
&= C_{c_1}^{N} \pi^{c_1} (1-\pi)^{N-c_1} C_{c_2 0}^{N-c_1} \pi^{c_{20}} (1-\pi)^{N-c_1-c_{20}} \\
&= C_N^{c_1} \pi^{c_1} (1-\pi)^{N-c_1} C_{N-c_1}^{c_{20}} \pi^{c_{20}} (1-\pi)^{N-c_1-c_{20}} C_{c_1}^{c_{21}} \pi^{c_{21}} (1-\pi)^{C_1-c_{21}}
\end{aligned}$$

On en déduit donc la log-vraissemblance en passant au log :

$$\begin{aligned}
\mathrm{l}(N, \pi) &= \ln\left(C_N^{c_1} C_{N-c_1}^{c_{20}} C_{c_1}^{c_{21}}\right) + (c_1 + c_{20} + c_{21}) \ln(\pi) + (2N - 2c_1 - c_{20} + c_1 - c_{21}) \ln(1-\pi) \\
&= \ln\left(C_N^{c_1} C_{N-c_1}^{c_{20}} C_{c_1}^{c_{21}}\right) + (c_1 + c_2) \ln(\pi) + (2N - c_1 - c_2) \ln(1-\pi)
\end{aligned}$$

car $c_{20} + c_{21} = c_2$

Simulation du tirage de C_1

La fonction de répartition de la loi discrète de $C_1 \sim \mathcal{B}(N, \pi)$ est la suivante :

$\forall x \in [0,1], \qquad F(x) = \sum_{k=0}^{N} \mathbb{P}(C_1 = k) 1_{\{k \leq x\}}$

On remarque que $\forall u \in [0,1], \exists p \in [0, N] \quad / \quad \sum_{k=0}^{p-1} \mathbb{P}(C_1 = k) \leq u \leq \sum_{k=0}^{p} \mathbb{P}(C_1 = k)$ [1]

Ainsi, $\forall x \in [k, k+1], \quad F(x) = \sum_{k=0}^{p} \mathbb{P}(C_1 = k) \geq u$

L'inverse généralisée de la loi discrète s'écrit donc : $F^{-1}(u) = p$

Finalement, on a :

$$\boxed{\forall u \in [0,1], \quad F^{-1}(u) = \inf_{p=1,\ldots,N} \left\{ p \;\middle|\; \sum_{k=0}^{p} \mathbb{P}(C_1 = k) \geq u \right\}}$$

[1] Avec la convention $\sum_{k=0}^{-1} \mathbb{P}(C_1 = k) = 0$

```r
my.qbinom <- function(u, N, pi){
  p <- sapply(c(0:N), FUN=function(n) choose(N, n)*pi^n*(1-pi)^{N-n})
  cdf <- cumsum(p)
  return(findInterval(u, cdf))
}

my.rbinom <- function(N, pi, n.iter=1){
  U <- runif(n=n.iter, min=0, max=1)
  res <- sapply(U, FUN=function(u) my.qbinom(u,N, pi))
  return(res)
}
```

```r
n.iter <- 10000
N <- 125
pi <- 0.15

generated.C1 <- my.rbinom(N, pi, n.iter)
```

```r
resultats <- data.frame(n=1:n.iter, valeurs=factor(generated.C1, levels = 0:N))
```

```r
#frequence theorique
freq_theo =dbinom(0:N, N, pi)

#calcul de la frequence empirique
freq_emp <- c()
for (k in 0:N){
 freq_emp <- c(freq_emp, mean(generated.C1==k))
}
freq_binom <- tibble( x=0:N, freq_emp=freq_emp, freq_theo=freq_theo)

#Représentation graphique
ggplot(freq_binom) + #Tableau représenter
aes(x = x) + #Abscisse commune
geom_col(mapping = aes(y = freq_emp), #Ordonne des frquences empiriques
width = 0.2, fill = "lightblue") +
geom_point(aes(y = freq_theo), #On ajoute le point des frquences thoriques
shape = 3, col = "red", size = 3) +
xlim(0, 40) +
labs(y = "Frequence", x = "Nombre de succes")
```

Simulation d'une réalisation possible de capture-marquage-recapture

```r
capture.sim <- function(N, pi){
  C1 <- my.rbinom(N=N, pi=pi)
  C20 <- my.rbinom(N=N-C1, pi=pi)
  C21 <- my.rbinom(N=C1, pi=pi)
  return(list(C1=C1, C20=C20, C21=C21))
}

capture.sim(N, pi) %>% as.data.frame()
```

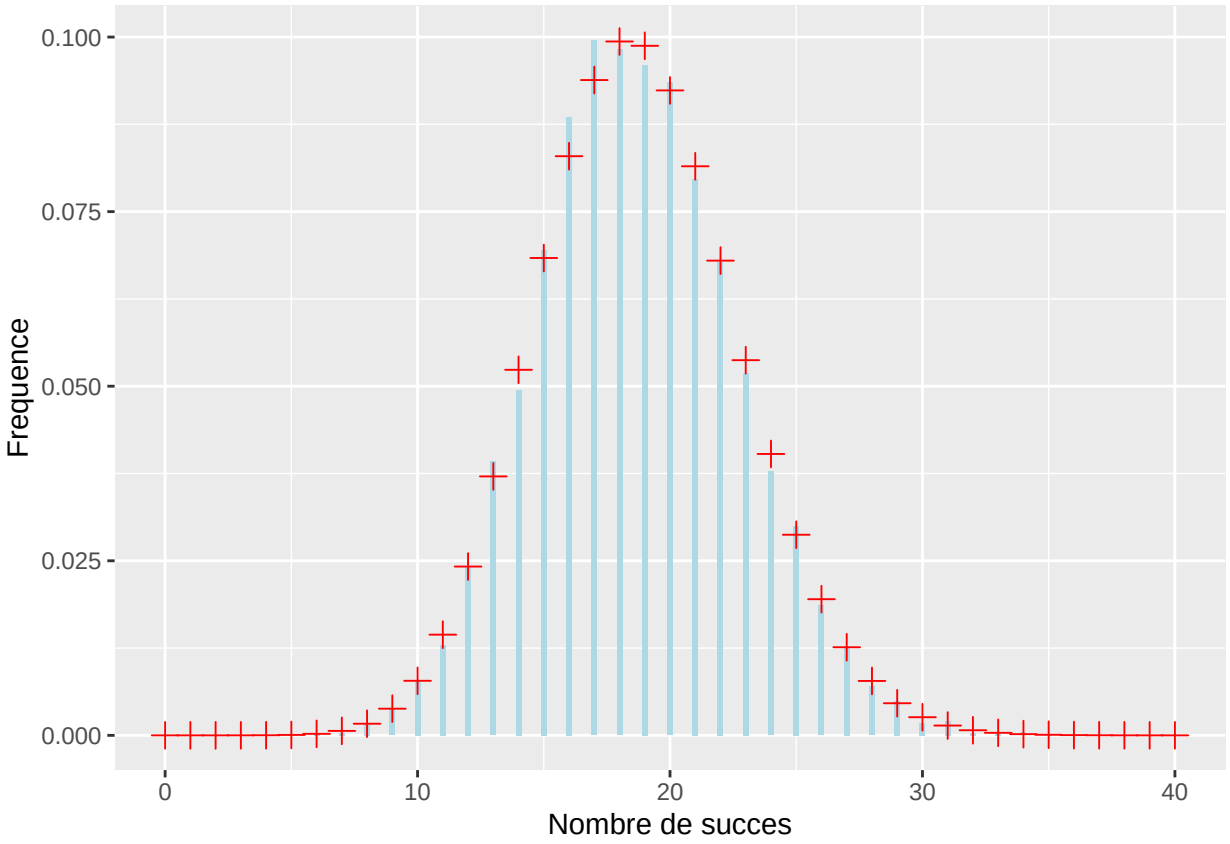

Figure 1: Comparaison des fréquences

```
##   C1 C20 C21
## 1 19  16   3
```

Supposons N connu

Supposons tout d'abord que $N = 950$ (connu) et estimons l'efficacité π.

Estimateur de maximum de vraissemblance $\hat{\pi}_{MLE}$ de π

$$\begin{aligned}\frac{dl}{d\pi}(N,\pi) &= (c_1+c_2)\frac{1}{\pi} + (2N-c_1-c_2)\frac{1}{1-\pi}(-1)\\ &= (c_1+c_2)\frac{1}{\pi} - (2N-c_1-c_2)\frac{1}{1-\pi}\end{aligned}$$

$$\begin{aligned}\frac{dl}{d\pi}(N,\pi) > 0 &\iff (c_1+c_2)\frac{1}{\pi} - (2N-c_1-c_2)\frac{1}{1-\pi} > 0\\ &\iff (c_1+c_2)\frac{1}{\pi} > (2N-c_1-c_2)\frac{1}{1-\pi}\\ &\iff \frac{c_1+c_2}{2N-c_1-c_2}(1-\pi) > \pi\\ &\iff \pi\left(1+\frac{c_1+c_2}{2N-c_1-c_2}\right) < \frac{c_1+c_2}{2N-c_1-c_2}\\ &\iff \pi\frac{2N}{2N-c_1-c_2} < \frac{c_1+c_2}{2N-c_1-c_2}\\ &\iff \pi < \frac{c_1+c_2}{2N}\end{aligned}$$

On en déduit que :

$$\boxed{\hat{\pi}_{MLE} = \frac{c_1+c_2}{2N}}$$

Loi beta à priori

On choisit une loi à priori $\beta(a,b)$ pour π, que l'on note $f(\pi) = \pi^{a-1}(1-\pi)^{b-1}$

$$\begin{aligned}\ln\left([\pi|N,C_1,C_{20},C_{21}]\right) &= l(N,\pi) + \ln\left(f(\pi)\right) + cte\\ &= (c_1+c_2)\ln(\pi) + (2N-c_1-c_2)\ln(1-\pi) + (a-1)\ln(\pi) + (b-1)\ln(1-\pi) + cte'\\ &= (c_1+c_2+a-1)\ln(\pi) + (2N-c_1-c_2+b-1)\ln(1-\pi) + cte''\end{aligned}$$

On reconnait, à une constante près, le logarithme d'une loi $\beta(c_1+c_2+a, 2N-c_1-c_2+b)$.

Donc:

$$\boxed{\pi|_N \sim \beta\left(c_1+c_2+a, 2N-c_1-c_2+b\right)}$$

On en déduit son espérance :

$$\mathbb{E}\left(\pi|_N\right) = \frac{c_1 + c_2 + a}{c_1 + c_2 + a + 2N - c_1 - c_2 + b}$$
$$\mathbb{E}\left(\pi|_N\right) = \frac{c_1 + c_2 + a}{2N + a + b}$$

Représentation graphique

On choisit $a = 1$, $b = 3$.

```
curve(dbeta(x, 1, 3))
```

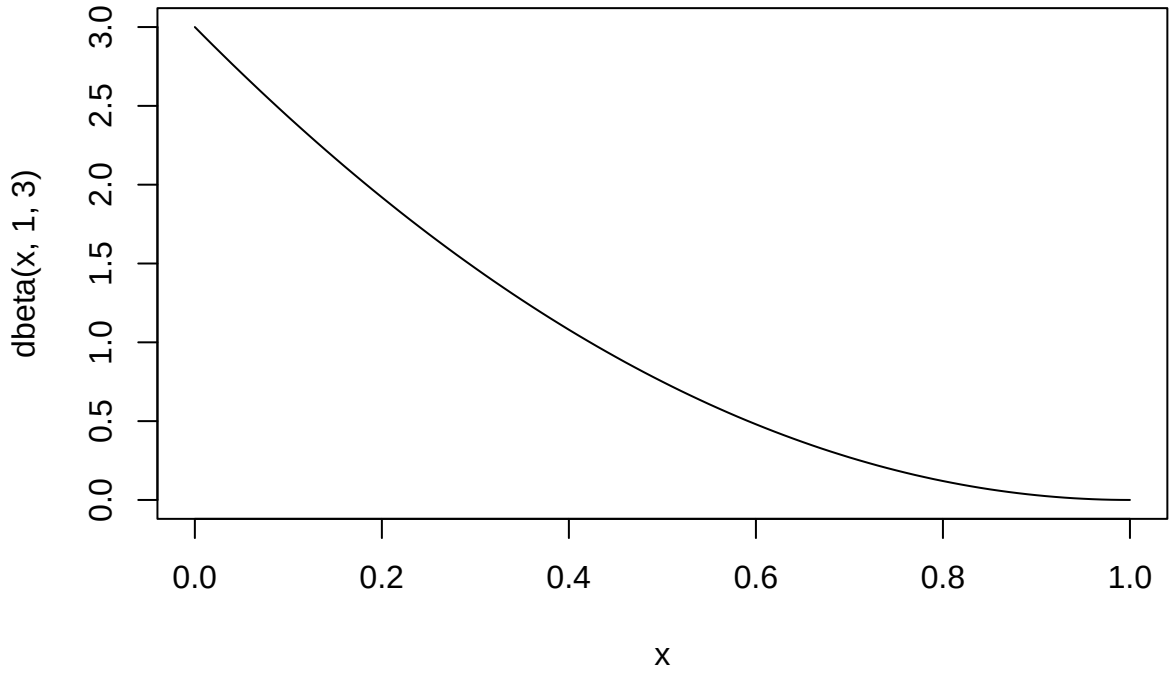

Figure 2: loi Beta(1, 3)

```
set.seed(32)
N = 950
pi = 0.3
df <- capture.sim(N, pi)
C1 <- df$C1
C2 <- df$C20 + df$C21
```

```
a <- 1
b <- 3
```

```r
a.post <- C1 + C2 + a
b.post <- 2*N - C1 - C2 + b
pi.MLE <- (C1 + C2)/(2*N)

p_val<- seq(0, 1, length.out=100)
```

```r
plot(p_val, dbeta(p_val, a, b),type="l",col="red",ylab="Densité",xlab="pi",
     main=paste("a=",a,"b=",b),ylim=c(0,55))
curve(dbeta(x,a.post,b.post),add=TRUE, col="blue")
abline(v=pi.MLE)
legend("topright", legend=c("a priori", "a posteriori", "max.vraiss"), bty='n',
       pch=rep('_',3), col=c("red","blue","black"))
```

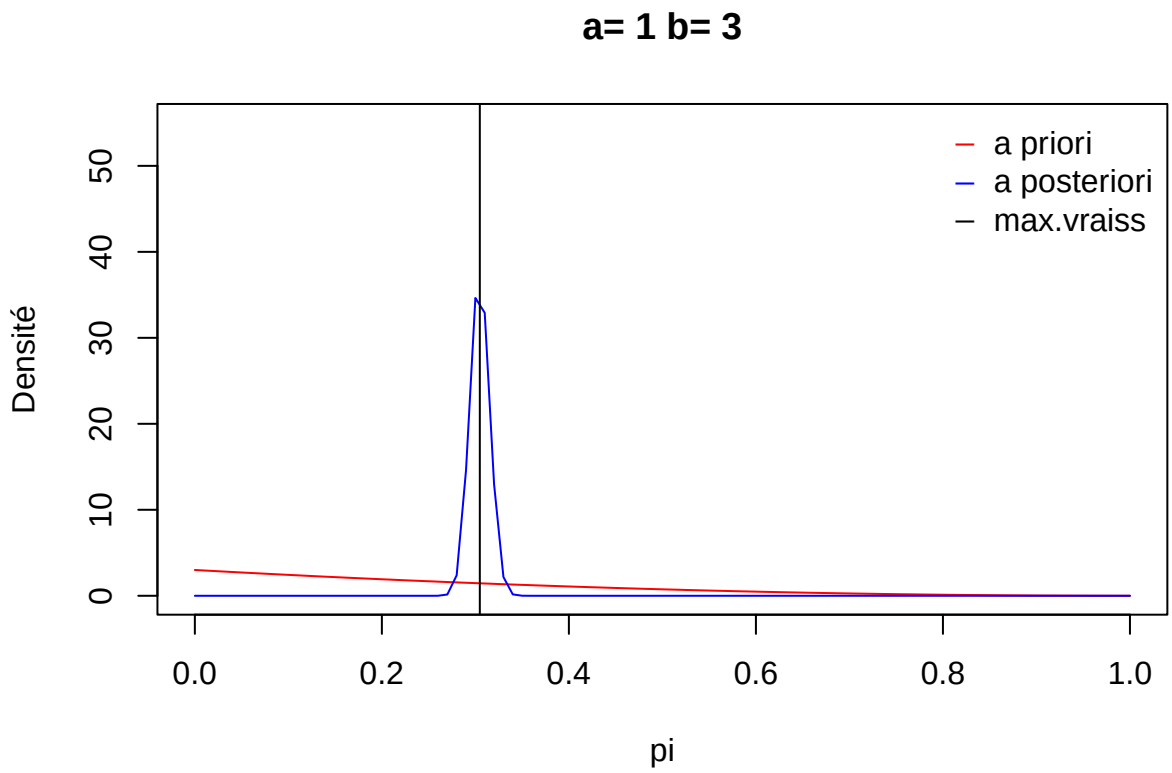

Figure 3: Densités à priori, à postériori et estimateur de maximum de vraissemblance

On observe que le seul mode de la loi à postériori coincide avec la valeur de l'estimateur de maximum de vraissemblance de π. C'est bon signe.

Supposons N et π connus

Approche fréquentiste

Pour évaluer le nombre d'individus N dans une population d'intérêt à partir de deux expériences de pêche de type capture-marquage-recapture, un estimateur fréquentiste naïf est l'estimateur de "Petersen" défini

par :

$$\hat{N} = \frac{C_1 C_2}{C_{21}}$$

Les données disponibles proviennent d'une expérience réelle "miniature" de capture-marquage-recapture réalisée par des étudiants à l'aide d'un saladier ("le lac") rempli de riz ("l'eau du lac") et de haricots blancs ("les poissons"). Les données observées par les étudiants sont les suivantes : $C_1 = 125$, $C_{20} = 134$ et $C_{21} = 21$.

```
C1 <- 125
C20 <- 134
C21 <- 21
C2 <- C20 + C21

N.hat <- C1*C2/C21
round(N.hat, 0)
```

```
## [1] 923
```

L'estimateur naïf de Petersen nous indique qu'il y a 923 "poissons" dans le lac.

Supposons ici que les "vraies" valeurs des paramètres soient $N_{true} = 923$ et $\pi_{true} = 0.15$.

```
n.iter <- 100
N.true <- 923
pi.true <- 0.15

N.MC <- function(N, pi, n.iter){
  df.repeated <-
    replicate(n.iter, capture.sim(N, pi)) %>% drop() %>% t() %>% as.data.frame()
  N.hat.repeated <- apply(df.repeated, MARGIN=1,
                          FUN = function(x) x$C1*(x$C20+x$C21)/x$C21)
  df.repeated$N.hat <- N.hat.repeated
  N.monte.carlo <- mean(df.repeated$N.hat)
  return(N.monte.carlo)
}

N.MC(N.true, pi.true, n.iter)
```

```
## [1] 960.7279
```

```
set.seed(57)
N.true.seq <- seq(100, 1000, 10)
N.evolution <- sapply(N.true.seq, FUN=function(n) N.MC(n, pi.true, n.iter))
```

```
plot(N.true.seq, N.evolution-N.true, xlab="N.true")
```

On remarque que plus N_{true} est grand, plus le biais est faible. Mais cette erreur semble avoir une tendance linéaire qui dépasse 0...

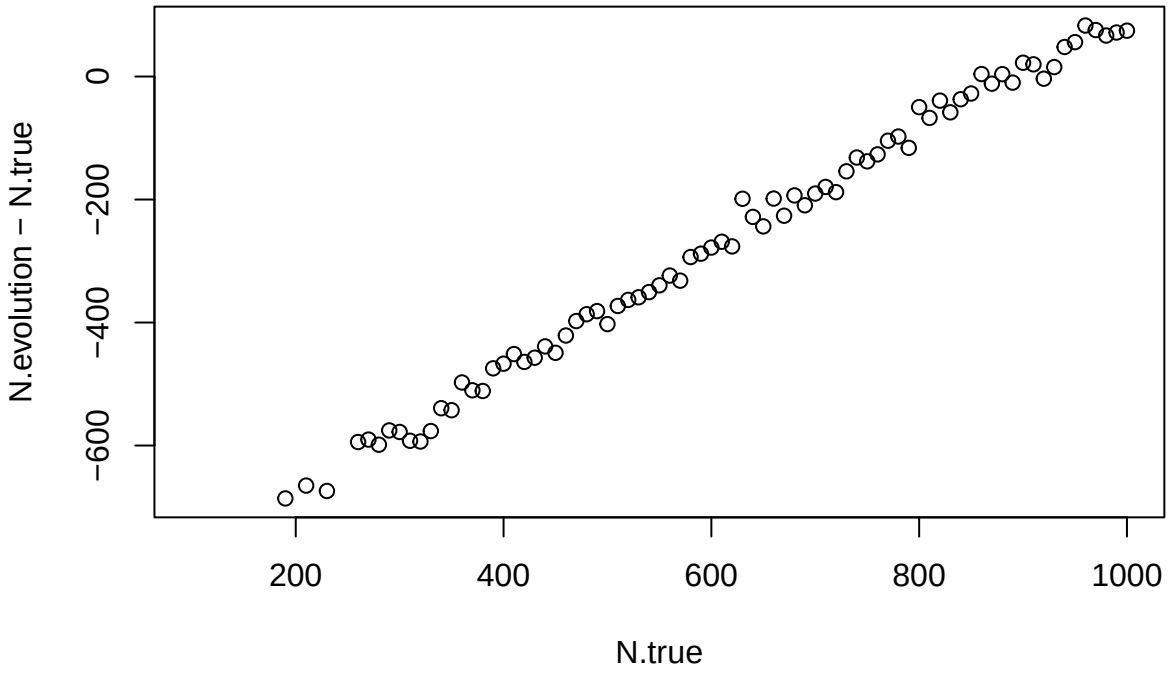

Figure 4: Evolution du biais sur N, en fonction de N.true

Approche bayésienne

On choisit une loi *à priori* uniforme sur l'ensemble discret $\{1, ..., 2000\}$ pour N. Sa densité à priori est donc $f(n) = \frac{1}{2000}$, $\quad \forall n \in \{1, ..., 2000\}$.

On rappelle que la vraissemblance du modèle $\mathcal{M}$, dont la log a été calculée précédemment est la suivante :

$$[y|N, \pi] = C_N^{c_1} C_{N-c_1}^{c_{20}} C_{c_1}^{c_{21}} \pi^{c_1+c_2} (1-\pi)^{2N-c_1-c_2}$$

On en déduit donc la forme de la loi à postériori.

$$\begin{aligned}
[N|y, \pi] &\propto f(n)[y|N, \pi] \\
&\propto \frac{1}{2000} C_N^{c_1} C_{N-c_1}^{c_{20}} C_{c_1}^{c_{21}} \pi^{c_1+c_2} (1-\pi)^{2N-c_1-c_2} \\
&\propto C_N^{c_1} C_{N-c_1}^{c_{20}} (1-\pi)^{2N-c_1-c_2} \\
&\propto C_N^{c_1} C_{N-c_1}^{c_{20}} (1-\pi)^{2N} \\
&\propto \frac{N!}{c_1!(N-c_1)!} \frac{(N-c_1)!}{(c_{20})!(N-c_1-c_{20})!} (1-\pi)^{2N} \\
&\propto \frac{N!}{(N-c_1-c_{20})!} (1-\pi)^{2N} \\
[N|y, \pi] &\propto C_N^{c1+c_{20}} (1-\pi)^{2N}
\end{aligned}$$

Bien qu'elle ressemble à une loi binomiale, cette loi ne fait pas partie des lois analytiques simples connues. Il sera donc nécessaire d'utiliser un algorithme ne dépendant pas de la simulation de cette loi, lorsque l'on souhaitera simuler sa densité.

Algorithme de Metropolis within Gibbs On se propose maintenant d'échantilloner dans la loi jointe à postériori du couple (N, π) sachant $y = (c_1, c_{20}, c_{21})$ à l'aide de l'algoritheme de Meetropolis within Gibbs dont l'itération k est la suivante :

- Mise à jour du paramètre π en tirant dans sa loi conditionnelle complète.
- Mise à jour du paramètre N avec l'algorithme de Metropolis-Hastings (MH), en utilisant comme loi de proposition une loi uniforme (discrète) sur $\{N^{curr} - k, N^{curr} + k\}$ où N^{curr} désigne la valeur courrante du paramètre N à une itération donnée et k est un paramètre de saut.

La loi de proposition $Q(x, y) := 1_{\{x-k,x+k\}}(y)$ étant symétrique, le ratio de Métropolis-Hastings se simplifie pour devenir, à l'ittération k :

$$r_k = \frac{[N^{cand}|\pi, y]}{[N^{k-1}|y]}$$

où N^{cand} a été tiré selon la loi $Q(N^{k-1}, \cdot)$

```
N.law <- function(N, pi, c1, c20){
  density <- choose(N, c1+c20)*(1-pi)^(2*N)
  return(density)
}
```