IMECC-UNICAMP
Atividade 1 - Métodos computacionais
Professor: Dr. Guilherme Ludwig
Vitor Macedo Rocha RA: 216394

1 Distribuições condicionais completas e implementação Gibbs

Considere $\mathbf{p} = [p_1, ..., p_{14}]^\top$, $\mathbf{x} = [n_1, ..., n_{14}, m_1, ..., m_{14}]^\top$ e $\mathbb{B} = \{0, 1, 2, ..., N\}$, temos que a verosimilhança e as distribuições a priori são dadas por

$$\mathcal{L}(N, \mathbf{p}|\mathbf{x}) \propto \frac{N!}{(N-r)!} \prod_{i=1}^{14} p_i^{n_i}(1-p_i)^{N-n_i} \mathbb{I}(N \in \mathbb{N})\mathbb{I}(p_i \in [0,1])\mathbb{I}(r \in \mathbb{B}) \tag{1}$$

$$\pi(N) = \frac{e^{-\lambda}\lambda^N}{N!}\mathbb{I}(N \in \mathbb{N}) \tag{2}$$

$$\pi(p_i) = \mathbb{I}(p_i \in [0,1]), i = 1, 2, ..., 14. \tag{3}$$

A seguir vamos derivar a distribuição condicional completa de $N|\mathbf{p}, \mathbf{x}$. Note que $\mathbb{I}(N \in \mathbb{N})\mathbb{I}(p_i \in [0,1])\mathbb{I}(r \in \mathbb{B}) = \mathbb{I}(N \in r, r+1, ...)\mathbb{I}(p_i \in [0,1])$, pois $\mathbb{I}(r \in \mathbb{B}) = \mathbb{I}(N \in \{r, r+1, ...\})$.

$$\begin{aligned}
\pi(N|\mathbf{p}, \mathbf{x}) &\propto \mathcal{L}(N, \mathbf{p}|\mathbf{x})\pi(N) \\
&= \frac{N!}{(N-r)!} \prod_{i=1}^{14} p_i^{n_i}(1-p_i)^{N-n_i} \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \frac{e^{-\lambda}\lambda^N}{N!}\mathbb{I}(N \in \mathbb{N}) \\
&= \frac{e^{-\lambda}\lambda^N}{(N-r)!} \prod_{i=1}^{14} p_i^{n_i}(1-p_i)^{N}(1-p_i)^{-n_i} \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \\
&= \frac{e^{-\lambda}\lambda^N}{(N-r)!} \prod_{i=1}^{14} [p_i^{n_i}] \prod_{i=1}^{14} \left[(1-p_i)^{N}\right] \prod_{i=1}^{14} \left[(1-p_i)^{-n_i}\right] \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \\
&\propto \frac{e^{-\lambda}\lambda^N}{(N-r)!} \left[\prod_{i=1}^{14}(1-p_i)\right]^N \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \\
&= \frac{e^{-\lambda}\left[\lambda \prod_{i=1}^{14}(1-p_i)\right]^N}{(N-r)!} \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \frac{e^{-\lambda \prod_{i=1}^{14}(1-p_i)}}{e^{-\lambda \prod_{i=1}^{14}(1-p_i)}} \\
&\propto \frac{\exp\{-\lambda \prod_{i=1}^{14}(1-p_i)\}\left[\lambda \prod_{i=1}^{14}(1-p_i)\right]^N}{(N-r)!} \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \\
&\propto \frac{\exp\{-\lambda \prod_{i=1}^{14}(1-p_i)\}\left[\lambda \prod_{i=1}^{14}(1-p_i)\right]^{N-r}}{(N-r)!} \mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_i \in [0,1]) \\
&= \frac{\exp\{-\lambda \prod_{i=1}^{14}(1-p_i)\}\left[\lambda \prod_{i=1}^{14}(1-p_i)\right]^{N-r}}{(N-r)!} \mathbb{I}(N - r \in \mathbb{N})\mathbb{I}(p_i \in [0,1])
\end{aligned}$$

Portanto

$$\pi(N|\mathbf{p}, \mathbf{x}) \propto \frac{\exp\{-\lambda \prod_{i=1}^{14}(1-p_i)\}\left[\lambda \prod_{i=1}^{14}(1-p_i)\right]^{N-r}}{(N-r)!} \mathbb{I}(N - r \in \mathbb{N})\mathbb{I}(p_i \in [0,1]) \tag{4}$$

Temos que $(N-r)|(\mathbf{p}, \mathbf{x}) \sim Poisson(\lambda \prod_{i=1}^{14}(1-p_i))$, identificado pelo *kernel* apresentado em 4.

Agora vamos derivar a distribuição condicional completa de $p_i|N, \mathbf{x}, i = 1, 2, ..., 14$. Considere $\mathbf{p_{(i)}} = (p_1, p_2, ..., p_{i-1}, p_{i+1}, ..., p_{14}$, isto é, o vetor de parâmetros $\mathbf{p}$ sem o i-ésimo elemento.

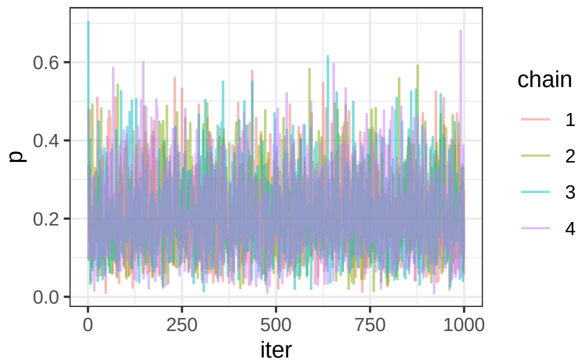
$$
\begin{aligned}
\pi(p_i|N, \mathbf{p_{(i)}}, \mathbf{x}) &\propto \mathcal{L}(N, \mathbf{p}|\mathbf{x})\pi(\mathbf{p_{(i)}}) \\
&= \frac{N!}{(N-r)!}\prod_{l=1}^{14} p_i^{n_l}(1-p_l)^{N-n_l}\mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_l \in [0,1]) \left[\prod_{j\neq i}\mathbb{I}(p_j \in [0,1])\right] \\
&= \frac{N!}{(N-r)!}\prod_{l=1}^{14} p_l^{n_l}(1-p_l)^{N-n_l}\mathbb{I}(N \in \{r, r+1, ...\})\mathbb{I}(p_l \in [0,1]) \\
&\propto p_i^{n_i}(1-p_i)^{N-n_i}\mathbb{I}(p_i \in [0,1]) \\
&= p_i^{(n_i-1)+1}(1-p_i)^{(N-n_i-1)+1}\mathbb{I}(p_i \in [0,1])
\end{aligned}
$$

Podemos então identificar que $p_i|N, \mathbf{x} \sim \text{Beta}(n_i + 1, N - n_i + 1), i = 1, 2, ..., 14$ pelo kernel que apresentado acima. Chegamos a conclusão que

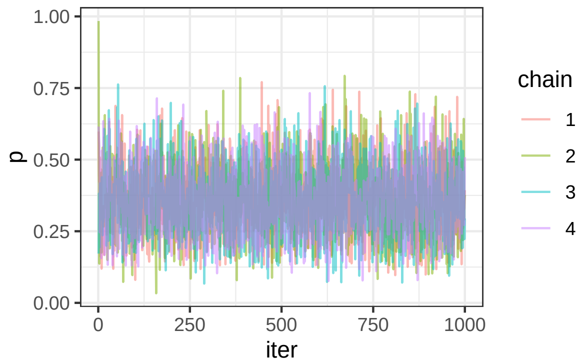
$$
(N-r)|(\mathbf{p}, \mathbf{x}) \sim Poisson(\lambda\prod_{i=1}^{14}(1-p_i)) \tag{5}
$$

$$
p_i|N, \mathbf{x} \sim Beta(n_i + 1, N - n_i + 1), i = 1, 2, ..., 14 \tag{6}
$$

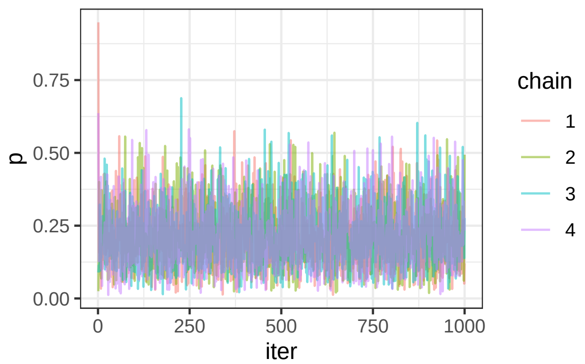
Para avaliar a convergência para a distribuição estacionária, vamos gerar as 4 cadeias com diferentes pontos iniciais e avaliar visualmente se elas se misturam.



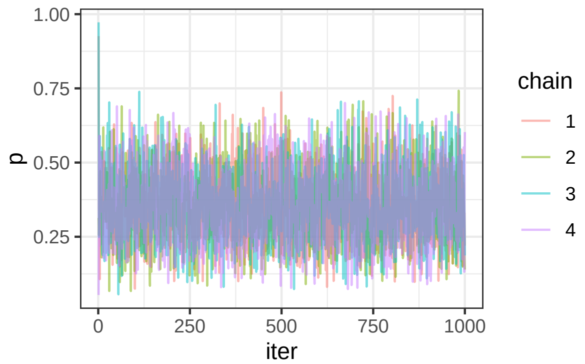
(a) p_1



(b) p_2

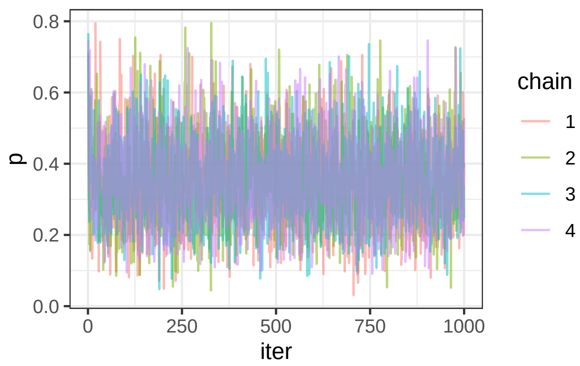


(c) p_3

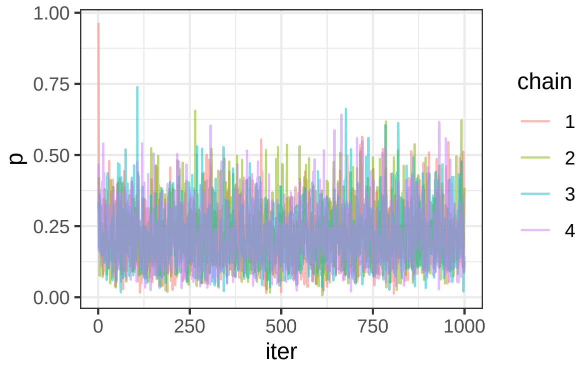


(d) p_4

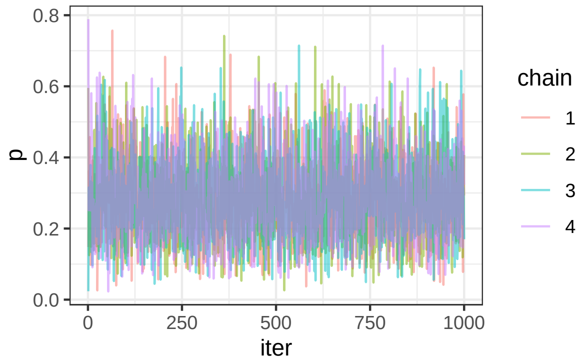
Figura 1:



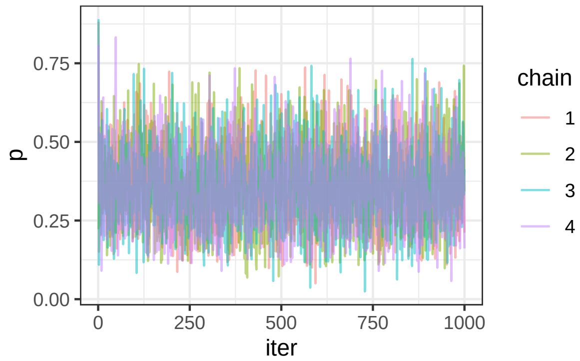
(a) p_5



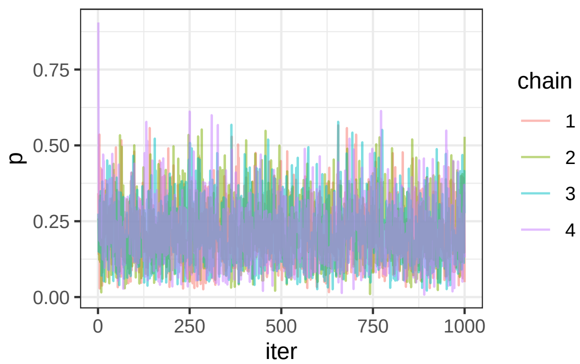
(b) p_6



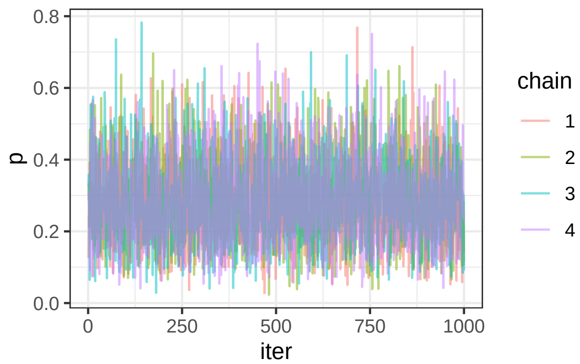
(c) p_7



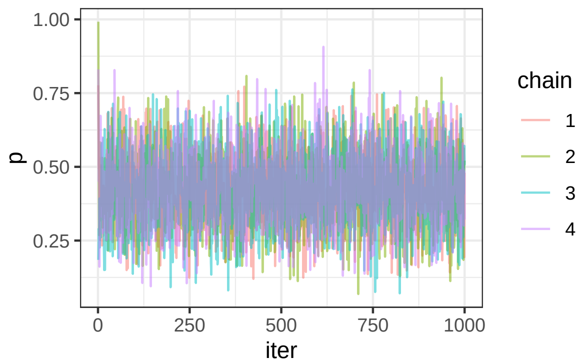
(d) p_8



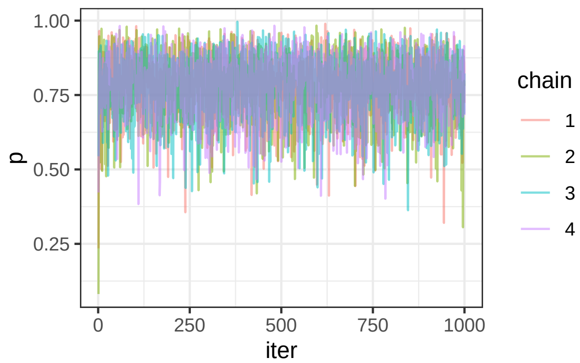
(e) p_9



(f) p_{10}

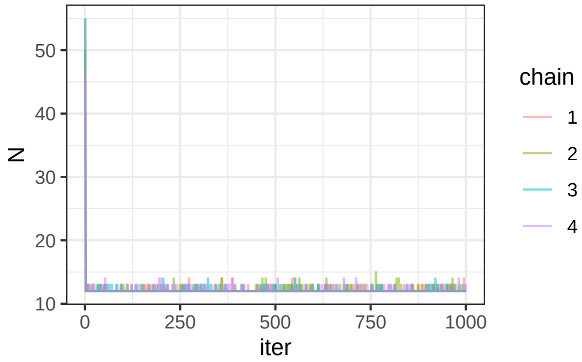


(g) p_{11}

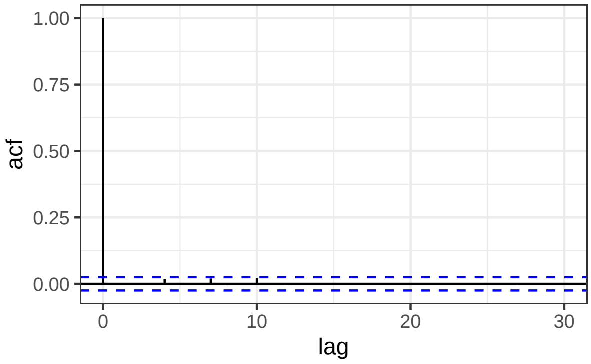


(h) p_{12}

Figura 2:



(a) N



(b) Auto correlação do parâmetro N.

Figura 3:

Podemos ver que as cadeias para todos os parâmetros se misturaram, trazendo indícios da convergência para a distribuição estacionária. Além disso vemos pelo gráfico da autocorrelação da cadeia do parâmetro N que não precisamos realizar o processo de *decorrelation* para garantir uma amostra pseudo-independente. Os gráficos de autocorrelação dos parâmetros $p_i, i = 1, 2, ..., 14$. apresentaram um comportamento similar.

Parâmetro	Média	Mediana	intervalo de credibilidade 95%
N	12.140	12.000	[12;13]
p1	0.211	0.196	[0.051;0.452]
p2	0.352	0.346	[0.136;0.608]
p3	0.212	0.197	[0.052;0.446]
p4	0.355	0.348	[0.139;0.604]
p5	0.356	0.349	[0.139;0.613]
p6	0.213	0.199	[0.05;0.456]
p7	0.284	0.277	[0.088;0.531]
p8	0.356	0.347	[0.137;0.614]
p9	0.214	0.200	[0.051;0.452]
p10	0.285	0.276	[0.091;0.531]
p11	0.426	0.419	[0.187;0.684]
p12	0.778	0.791	[0.538;0.946]
p13	0.213	0.199	[0.051;0.458]
p14	0.427	0.425	[0.188;0.678]

2 Implementação *Hamiltonian Monte Carlo*

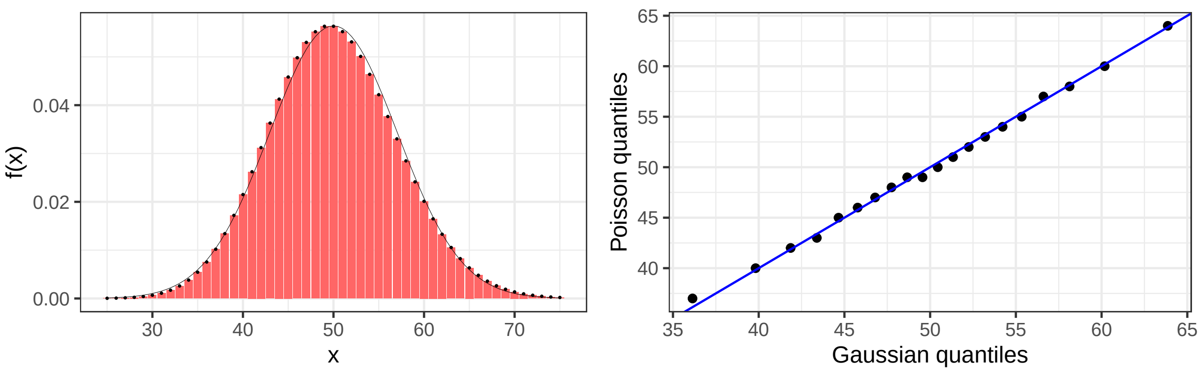
Como bem sabemos, o HMC *sampler* não comporta suportes discretos pela fato de que a função que descreve a energia cinética há de ser derivável em relação a "trajetória"da cadeia. Por esse fato, optei por procurar uma solução na própria documentação do stan e acabei encontrando a opção de marginalização do parâmetro discreto.

Para realizar a implementação no R, procurei artigos na web que tratassem desse assunto e encontrei uma extensão do HMC chamado de *Discontinuous Hamiltonian* Monte Carlo (NISHIMURA; DUNSON; LU, 2020). Nele é discutido também o porquê o HMC falha quando temos interesse em densidades alvo descontínuas, resumidamente a etapa da solução das equações diferenciais via *leapfrog* apresenta uma instabilidade levando ao erro.

Outra ideia que tive foi utilizar a aproximação da distribuição poisson pela Normal, da seguinte maneira $Poisson(\lambda) \approx Normal(\mu = \lambda, \sigma^2 = \lambda)$. Como essa última foi a solução mais simples que achei, vou implementar primeiro e se tiver tempo hábil tentarei implementar as outras.

2.1 HMC via aproximação Poisson-Normal

No gráfico a seguir podemos ver que a aproximação é realmente razoável.



(a) (b)

Figura 4: Aproximação da distribuição $Poisson(\lambda = 50)$ pela distribuição $Normal(\mu = 50, \sigma^2 = 50)$.

A partir da verosimilhança apresentada em 1 e as distribuições a priori $\pi(N) \sim Normal(50, 50)$ e $\pi(p_i) \sim U(0,1), i = 1, 2, ..., 14$. Obtemos a seguinte função proporcional a distribuição a posteriori.

$$\pi(N, \mathbf{p}|\mathbf{x}) \propto \left[\frac{N!}{(N-r)!}\prod_{i=1}^{14} p_i^{n_i}(1-p_i)^{N-n_i}\mathbb{I}(p_i \in [0,1])\mathbb{I}(r \in [0,N])\right] \times \pi(N) \times \pi(\mathbf{p})$$

Precisamos então das quantidade referentes a "força cinética" $K(\boldsymbol{\rho})$ e "força potencial" $U(\mathbf{y})$. Definimos

$$\begin{aligned} U(\mathbf{y}) &= -\log(\pi(N, \mathbf{p}|\mathbf{x})) \\ &- \left[\log(\Gamma(N+1)) - \log(\Gamma(N-r+1)) + \sum_{i=1}^{14} n_i \log(p_i) + (N-n_i)\log(1-p_i)\right] \end{aligned}$$

$$K(\boldsymbol{\rho}) = -\log(\pi(\boldsymbol{\rho}|N, \mathbf{p}))$$

3 Implementação *Hamiltonian Monte Carlo* no STAN

Consegui fazer a implementação do modelo de captura-marcação-recaptura no STAN com as prioris de interesse usando a aproximação da poisson-Normal sugerida anteriormente. A seguir são apresentados os resultados, notamos que foram coerente com os métodos implementados no R.

Referências

NISHIMURA, A.; DUNSON, D. B.; LU, J. Discontinuous hamiltonian monte carlo for discrete parameters and discontinuous likelihoods. *Biometrika*, Oxford University Press, v. 107, n. 2, p. 365–380, 2020.

Apêndice A

A.1 Implementação Gibbs Sampler

```r
library(readr)
library(tidyverse)
dados <- read_table("./Atividade1/resolucao/dados.txt",
                    col_names = FALSE)
colnames(dados) <- c('onca',paste0('armadilha',1:14))
nj <- dados%>%
  pivot_longer(-onca,values_to = "capturada",
               names_to = "armadilha",names_prefix = 'armadilha',
               names_transform = list(armadilha = as.numeric))%>%
  group_by(armadilha)%>%
  summarise(nj=sum(capturada))
mj <- dados%>%
  pivot_longer(-onca,values_to = "capturada",names_to = "armadilha",
               names_prefix = 'armadilha',
               names_transform = list(armadilha = as.numeric))%>%
  group_by(onca)%>%
  mutate(mj=case_when(cumsum(capturada)*capturada>1 ~ 1 , TRUE ~ 0))%>%
  group_by(armadilha)%>%
  summarise(mj=sum(mj))
Ngibbs=10e4
iter0 <- c(rpois(1,50),runif(14))
chain <- matrix(0,ncol = Ngibbs,nrow=length(iter0))
chain[,1] <- iter0
rownames(chain) <- c("N",paste0("p",1:14))
lambda=50
r <- sum(nj$nj)-sum(mj$mj)
nj <- nj$nj
for (i in 2:Ngibbs) {
  lambda_pos <- lambda * prod(1-chain[-1,i-1])
  N_i_plus_1 <- rpois(n=1,lambda = lambda_pos) + r
  chain[1,i] <- N_i_plus_1
  for (j in 1:length(nj)) {
    aux <- rbeta(n = 1,shape1 = nj[j]+1,shape2 = N_i_plus_1-nj[j]+1)
    chain[j+1,i] <- aux
  }
}
```

A.1 Implementação STAN

```
data{
  int j;
  vector[j] ni;
  vector[j]mi;
}
transformed data{
  real r;
  r = sum(ni) - sum(mi);
}
parameters{
  vector<lower=0.00001,upper=.9999>[j] p;
  real<lower=r> N;
}

transformed parameters{
  real Nt;
  Nt= N-r;
}

model{
  target+= normal_lpdf(Nt|50,sqrt(50));
  target+=uniform_lpdf(p|0,1);
  target += log(tgamma(N+1)) - log(tgamma(N - r + 1)) + sum(ni .* log(p)  + (N-ni).*log(1-p));
}
```