Introduction au calcul bayésien

Dhafer Malouche

Ecole Supérieure de la Statistique
et de l'Analyse de l'Information de Tunis

3ième Année, 2012-2013

Plan

Calcul Bayésien avec un échantillonnage indépendant

Premiers pas avec `OpenBUGS`

Comparaison de modèles

Calcul Bayésien avec un échantillonnage indépendant

Premiers pas avec `OpenBUGS`

Comparaison de modèles

Exemple : Élections présidentielles aux USA : Obama-McCain(2007).

- Système électoral aux USA :
  - 50 états + District of Colombia.
  - dans chaque état, le candidat doit gagner les votes des grands électeurs.
  - pour gagner le vote des grands électeurs, il faut que le candidat(te) gagne les élections dans chaque état.
- Formellement :
  - Soit θ_{O_j} (resp. θ_{M_j}) : la probabilité de voter pour Obama (resp. Mc-Cain) dans l'un l'État $j = 1, \ldots, 51$.
  - GE_j le nombre de grands électeurs et GE_O (resp. GE_M) le nombre de votes des grands électeurs obtenus par Obama (resp. Mc-Cain) :

$$GE_O = \sum_{j=1}^{51} GE_j \, \mathbb{1}_{\{\theta_{O_j} > \theta_{M_j}\}}.$$

  - pour que Obama gagne il faut que $GE_0 > 269$ (au total il y a 538 GE).

Problème :

- Comment estimer à partir d'un résultat de sondage la probabilité qu'un des deux candidats gagne ?
- Formellement, Si $\underline{y} = \left((y_{O_j}, y_{M_j}, N - y_{O_j} - y_{M_j}), j = 1, \ldots, 51\right)$ est l'ensemble des données récoltées lors du sondage, comment estimer

$$\mathbb{P}\left(GE_O > 269 \mid \underline{y}\right).$$

- Solution : construction d'un modèle bayésien ?

Le modèle Bayésien

- Sur un échantillon de N électeurs dans un état j : y_{O_j}, y_{M_j} personnes déclarent voter resp. pour Obama et McCain. $N - y_{O_j} - y_{M_j}$ est le nombre d'électeurs qui déclarent ne pas voter pour les deux candidats.

- La vraisemblance :

$$L(y_{O_j}, y_{M_j} \mid \theta_{O_j}, \theta_{M_j}) \propto \theta_{O_j}^{y_{O_j}} \theta_{M_j}^{y_{M_j}} (1 - \theta_{O_j} - \theta_{M_j})^{N - y_{O_j} - y_{M_j}}$$

- La loi *a priori* conjuguée : la loi de Dirichelet.

$$g_j(\theta_{O_j}, \theta_{M_j}, \theta_j) \sim \text{Dirichlet}(\alpha_1, \alpha_2, \alpha_3)$$

où $\theta_j = 1 - \theta_{O_j} - \theta_{M_j}$.

La loi a posteriori

- $X = (X_1, \ldots, X_k) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_k)$,

$$f(x_1, \ldots, x_k) = \frac{1}{\Gamma(\alpha_0)} \prod_{i=1}^{k} \Gamma(\alpha_i)\, x_i^{\alpha_i - 1}$$

où $\forall\, i = 1 \ldots k-1$, $x_i \in [0,1]$, $x_k = 1 - x_1 - \ldots - x_{k-1}$ et $\alpha_0 = \sum_{i=1}^{k} \alpha_i$.

- La loi *a posteriori*

$$g(\theta_{O_j}, \theta_{M_j}, \theta_j \mid y_{O_j}, y_{M_j}, N - y_{O_j} - y_{M_j})$$

$$\propto \quad L(y_{O_j}, y_{M_j} \mid \theta_{O_j}, \theta_{M_j})\, g_j(\theta_{O_j}, \theta_{M_j}, \theta_j)$$

$$\propto \quad \theta_{O_j}^{\alpha_1 + y_{O_j} - 1}\, \theta_{M_j}^{\alpha_2 + y_{M_j} - 1}\, \theta_j^{\alpha_3 + N - y_{O_j} - y_{M_j} - 1}$$

- Donc la loi a posteriori est

$$\text{Dirichlet}(\alpha_1 + y_{O_j},\ \alpha_2 + y_{M_j},\ \alpha_3 + N - y_{O_j} - y_{M_j}).$$

Données : `election.2008`

- Dernier sondage fourni par la CNN des pourcentages d'intention de votes pour chaque candidat dans les 51 état.
- `election.2008` est une matrice à 3 colonnes :
  - les pourcentages de votes en faveur de chaque candidat
  - le nombre des GE dans chaque état.
- On a questionné 500 personnes dans chaque état et on considère une loi *a priori* uniforme sur les paramètres.
  $\alpha_1 = \alpha_2 = \alpha_3 = 1$.
  $\Rightarrow$ La loi *a posteriori* est

$$\text{Dirichlet}(500pct_{O_j}+1,\ 500pct_{M_j}+1,\ 500-500pct_{O_j}-500pct_{M_j}+1).$$

Algorithme d'estimation

- Pour estimer la probabilité $\mathbb{P}\left(GE_O > 269 \mid \underline{y}\right)$ on va utiliser une méthode de Monte Carlo.

- A partir d'un algorithme de simulation de la v.a. GE_O on génère un $n-$échantillon $(GE_O)_1, \ldots, (GE_O)_n$ de même loi que $GE_O \mid \underline{y}$ et

$$\mathbb{P}\left(GE_O > 269 \mid \underline{y}\right) \approx \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{(GE_O)_i > 269\}}.$$

- Algorithme de Simulation
  1. Simulation de la loi *a posteriori* des $(\theta_{O_j}, \theta_{M_j})$
  2. Simulation de $GE_O \mid \underline{y}$

Estimation de $\mathbb{P}\left(GE_O > 269 \mid \underline{y}\right)$

Simulation de la v.a. GE_O à partir de la simulation de la loi *a posteriori*

1. On simule pour chaque $j = 1, \ldots, 51$, $N = 5000$ réplications à partir de la loi a posteriori de θ_{O_j}, θ_{M_j} :

$$(\theta_{O_j}^{(1)}, \theta_{M_j}^{(1)}), \ldots, (\theta_{O_j}^{(N)}, \theta_{M_j}^{(N)})$$

2. On calcule ensuite

$$\overline{p}_{j,N} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\{\theta_{0_j}^{(i)} > \theta_{M_j}^{(i)}\}}$$

$\overline{p}_{j,N}$ s'interprète comme la probabilité *a posteriori* que dans l'État j on vote en faveur de Obama.

Simulation des élections...suite

3. Pour chaque j, on tire $\epsilon_{j,N} \sim \text{Bernoulli}(\overline{p}_{j,N})$, ceci sera fait $n = 1000$ fois. On obtient pour chaque j,

$$\epsilon_{j,N}^{(1)}, \ldots, \epsilon_{j,N}^{(n)}$$

4. le $n-$échantillon de même loi que de GE_O est alors

$$(\widehat{GE}_O)_i = \sum_{j=1}^{51} GE_j \, \epsilon_{j,N}^{(i)}, \ \forall \, i = 1 \ldots n.$$

Mise en œuvre sous R

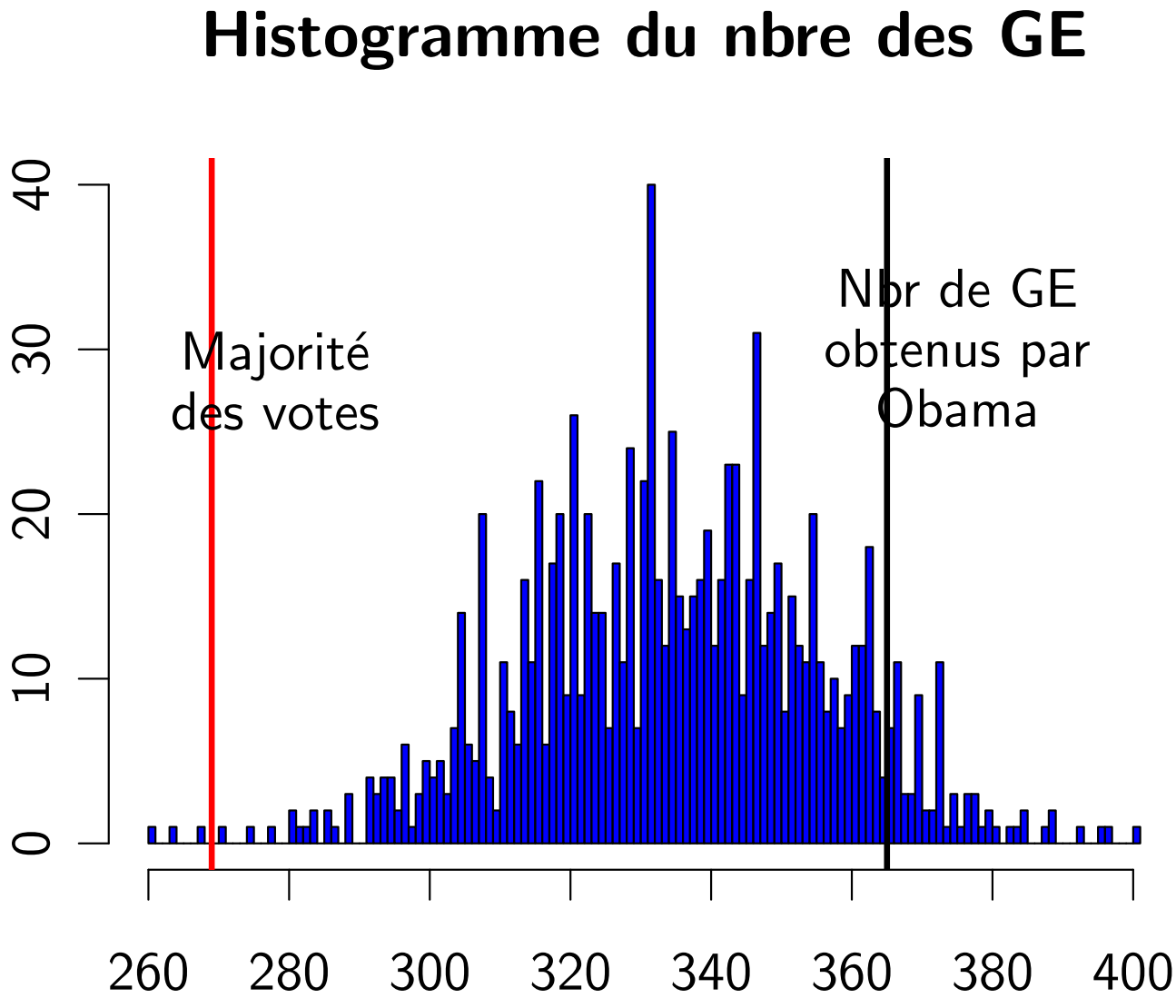
- Calcul de $\overline{p}_{j,N}$

```
> prob.Obama=function(j)
+ {
+ p=rdirichlet(5000,
+ 500*c(M.pct[j],
+ O.pct[j],100-M.pct[j]-O.pct[j])/100+1)
+ mean(p[,2]>p[,1])
+ }
> Obama.win.probs=sapply(1:51,prob.Obama)
```

- Calcul de $\widehat{GE}_0$.

```
> sim.election=function()
+ {
+ winner=rbinom(51,1,Obama.win.probs)
+ sum(GE *winner)
+ }
> sim.GE=replicate(1000,sim.election())
```

Résultats


Histogramme du nbre des GE
Majorité
des votes
Nbr de GE
obtenus par
Obama
0
10
20
30
40
260
280
300
320
340
360
380
400

Exemple : lancement de satellites

- Entre 1980 et 2002 il y a eu 11 lancements de satellites, 3 succès et 8 échecs.
- Objectif : Estimer la probabilité de réussite d'un nouveau lanceur, fournir un intervalle de confiance/crédibilité.

Lanceur	Résultat
Pegasus	Succès
Percheron	Echec
AMROC	Echec
Conestoga	Echec
Ariane 1	Succès
India SLV-3	Echec
India ASLV	Echec
India PSMV	Echec
Shavit	Succès
Taepodong	Echec
Brazil VLS	Echec

Approche fréquentiste (Rappel)

- **Distribution des données** (le modèle statistique)
  - y est le nombre de succès parmi n lancements de satelittes,
  - θ est la probabilité de succès d'un lancement

$$y \mid \theta \sim \mathcal{B}(n, \theta);$$

- **Vraisemblance et estimateur du maximum de vraisemblance**
  - $l(y \mid \theta) = C_n^y \theta^y (1-\theta)^{n-y}$,
    $\log\left(l(y \mid \theta)\right) = y \log\theta + (n-y)\log(1-\theta)$.
  - $\widehat{\theta} = \text{argmax}_\theta \log\left(l(y \mid \theta)\right)$

$$\widehat{\theta} = \frac{y}{n}$$

  - Calcul d'intervalle de confiance (asymptotique n grand) :
    $(\widehat{\theta} \pm z_{\alpha/2}\sqrt{\frac{1}{n}\widehat{\theta}(1-\widehat{\theta})}$.

Approche Bayésienne

- On considère une loi a priori Beta sur θ

$$\pi(\theta) \propto \theta^{a-1}(1-\theta)^{b-1}.$$

- Calcul de la loi a posteriori

$$\begin{aligned} \pi(\theta \mid y) &\propto l(y \mid \theta)\pi(\theta) \\ &\propto \theta^{y+a-1}(1-\theta)^{n-y+b-1}. \end{aligned}$$

  Donc $y \mid \theta \sim \text{Beta}(y+a, n-y+b)$.

- La distribution, moyenne, variance, intervalle de crédibilité a posteriori ?

OpenBugs

Présentation OpenBUGS fait partie du projet BUGS (Bayesian inference Using Gibbs Sampler) qui vise à rendre simple la pratique des méthodes MCMC aux statisticiens. Il a été développé par l'unité MRC Biostatistics de l'université de Cambridge. OpenBUGS est un logiciel libre et gratuit.

Utilisation possible

- Via une interface " clique-bouton " qui permet de contrôler l'analyse,
- En utilisant des modèles définis par des interfaces graphiques, appelés DoddleBUGS,
- Via d'autres logiciels tels que R (en particulier via le package R2OpenBUGS)

Algorithme de Gibbs

Supposons que le vecteur paramètre θ est constitué de k composantes, i.e. $\theta^T = (\theta_1, \ldots, \theta_k)$. Les étapes de l'algorithme de Gibbs sont :

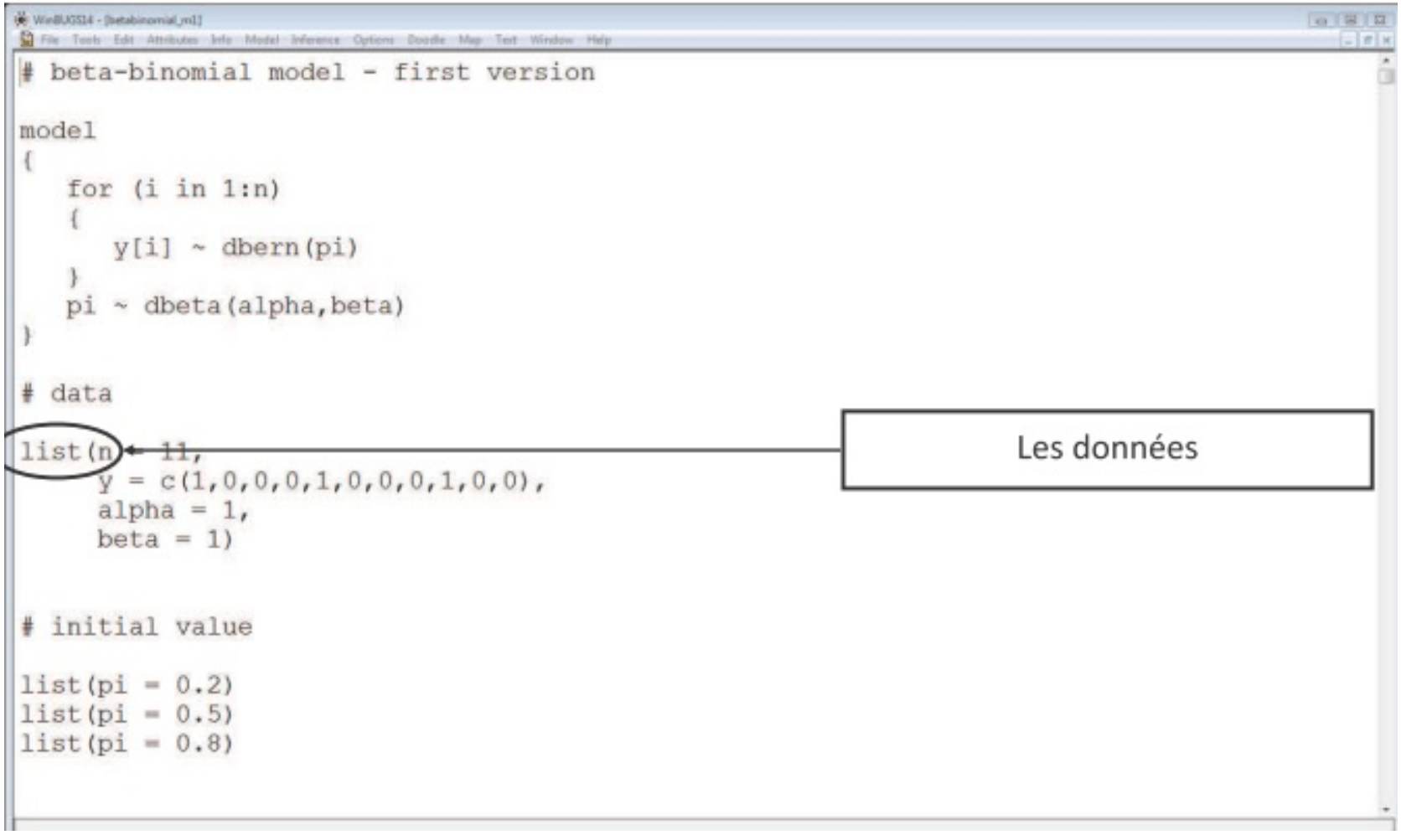
1. Choisir des valeurs initiales pour $\theta_1^{(0)}, \ldots, \theta_k^{(0)}$.
2. Générer $\theta_1^{(1)}$ selon $\pi\left(\theta_1 \mid \theta_2^{(0)}, \ldots, \theta_k^{(0)}, y\right)$.
   Générer $\theta_2^{(1)}$ selon $\pi\left(\theta_2 \mid \theta_1^{(1)}, \theta_3^{(0)} \ldots, \theta_k^{(0)}, y\right)$.
   ...
   Générer $\theta_k^{(1)}$ selon $\pi\left(\theta_k \mid \theta_1^{(1)}, \theta_2^{(1)} \ldots, \theta_{k-1}^{(1)}, y\right)$.
3. Répéter l'étape 2 plusieurs milliers de fois.

Utilisation de OpenBUGS

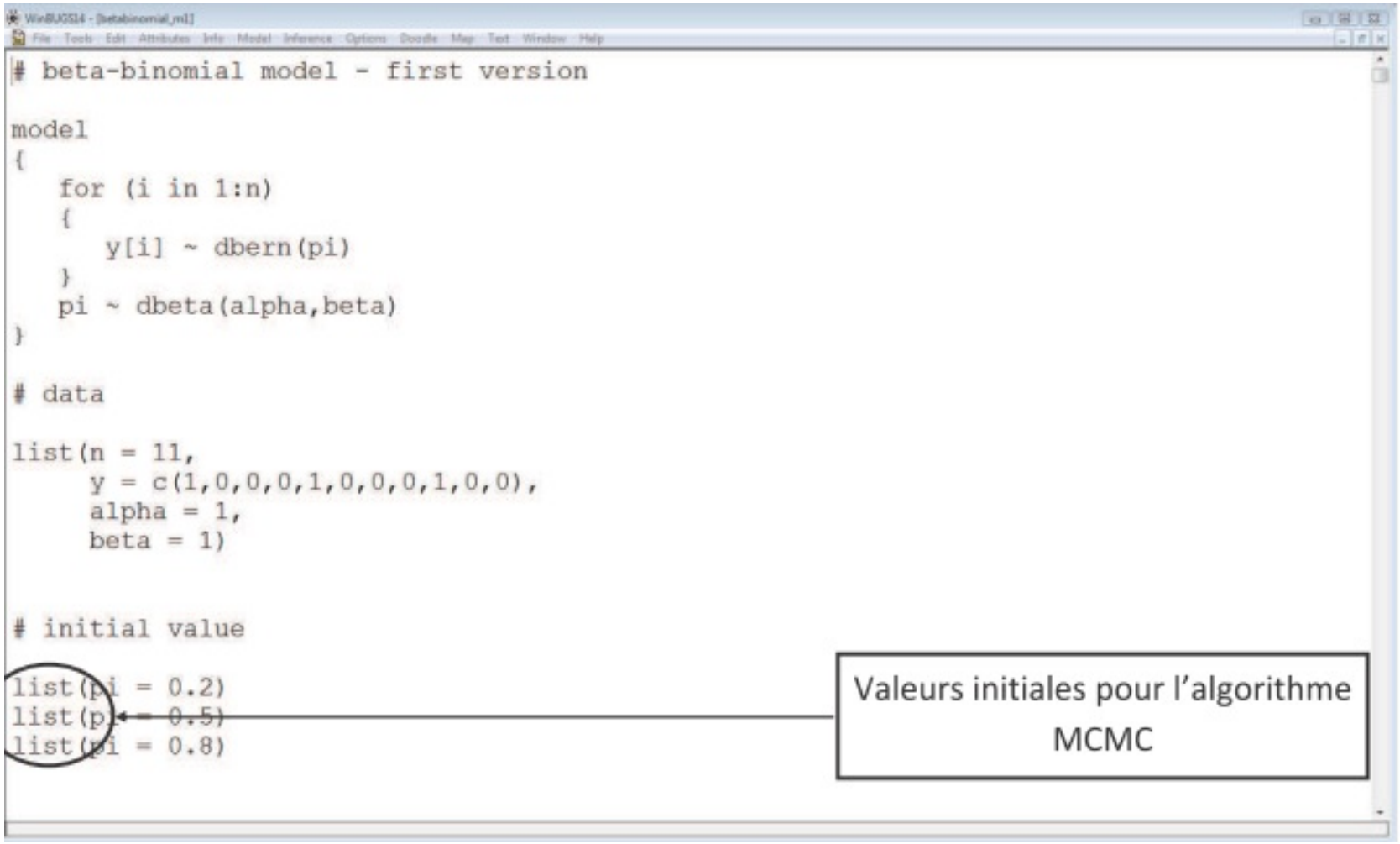
```
# beta-binomial model - first version

model
{
   for (i in 1:n)
   {
      y[i] ~ dbern(pi)
   }
   pi ~ dbeta(alpha,beta)
}

# data

list(n = 11,
     y = c(1,0,0,0,1,0,0,0,1,0,0),
     alpha = 1,
     beta = 1)


# initial value

list(pi = 0.2)
list(pi = 0.5)
list(pi = 0.8)
```

Un modèle composé de la vraisemblance et de la distribution a priori

Utilisation de OpenBUGS



Utilisation de OpenBUGS



Les étapes de l'exécution sous OpenBUGS

1. Vérifier le modèle,
2. Charger les données,
3. Spécifier le nombre de chaînes MCMC,
4. Compiler le modèle,
5. Charger les conditions initiales,
6. Générer les réalisations "burnin",
7. Spécifier les paramètres/quantités à conserver,
8. Générer les réalisations à conserver,
9. Vérifier la convergence et mise en place des résultats.

Diagnostique de la convergence des chaînes de Markov

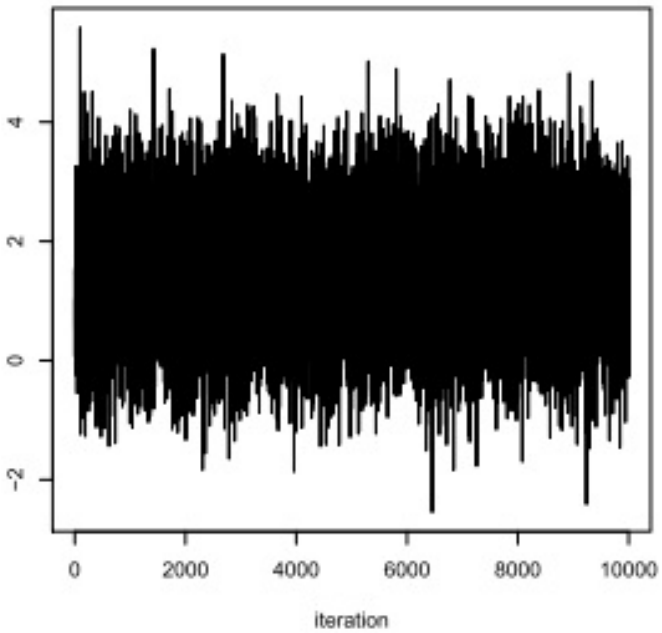
- A partir de la théorie des chaînes de Markov, nous espérons que ces chaînes éventuellement convergent vers la distribution stationnaire, qui est aussi notre distribution cible (la loi *a posteriori*).
- Il n'y a aucune garantie que les chaînes de Markov convergent après un certain nombre de tirages.
- Comment savoir que ces chaînes convergent ?
- Réponse : on est jamais sûr, mais il y a certain nombre de tests et de graphiques qui permettent de diagnostiquer la convergence de la chaîne.
- Outils : package `coda` sous `R`.

Diagnostique visuel

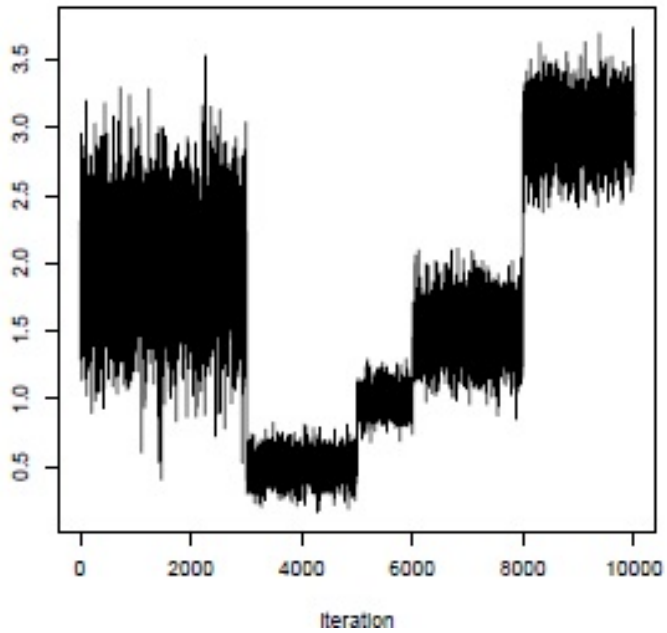
Si $(\theta_i)_{i=1,\ldots,M}$ est la chaîne de Markov simulée.

- Le `traceplot` est une représentation graphique de (i, θ_i).
- Nous pouvons voir si notre chaîne se coince dans une certaine région de l'espace des paramètres

Convergence possible



Convergence non évidente



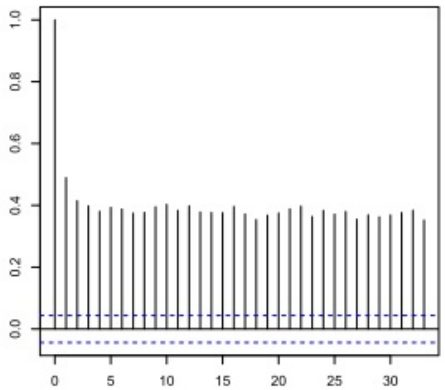
Autocorrélation

- Une autre façon d'évaluer la convergence est d'estimer les autocorrélations qui est une corrélation de la chaîne de Markov avec elle-même avec un $k-$ième décalage
- L'autocorrélation ρ_k s'exprime

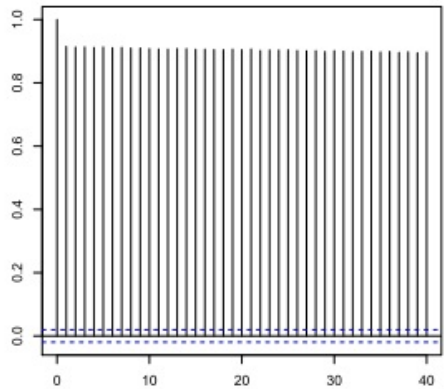
$$\rho_k = \frac{\sum_{i=1}^{n-k}(x_i - \overline{x})(x_{i+k} - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

- On s'attend de voir que l'autocorrélation du $k-$ième décalage de la chaîne décroît quand k croît.

Convergence possible



Convergence non évidente



Critères de diagnostiques : Gelman et Rubin

Les étapes de la méthode (pour chaque paramètre).

1. Simuler $m \geq 2$ chaînes de Markov de longueur $2n$ avec valeurs initiales différentes.
2. Ecarter les n premières simulations des chaînes.
3. Calculer les variances inter et intra chaînes.
4. Calculer l'estimation de la variance du paramètre comme une somme pondérées des deux variances inter et intra chaînes.
5. Calculer un facteur de réduction d'échelle.

Variance intra chaînes

$$W = \frac{1}{m}\sum_{j=1}^{m} s_j^2$$

où

$$s_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\theta_{ij} - \overline{\theta}_j)^2$$

- s_j^2 est la variance de la $j-$ième chaîne.
- W est la moyenne des variances des m chaînes.
- W sous-estime la variance du paramètre puisqu'il est probable que la chaîne n'a pas atteint son état de stationnarité.

Variance inter chaînes

$$B = \frac{n}{m-1} \sum_{j=1}^{m} (\bar{\theta}_j - \bar{\bar{\theta}})^2$$

où

$$\bar{\bar{\theta}} = \frac{1}{m} \sum_{j=1}^{m} \bar{\theta}_j,$$

C'est la variance des moyennes des chaînes.
L'estimation de la variance de la distribution stationnaire

$$\widehat{\mathrm{Var}(\theta)} = \left(1 - \frac{1}{n}\right) W + \frac{1}{n} B.$$

Facteur de réduction d'échelle

- Le facteur de réduction d'échelle s'exprime

$$\widehat{R} = \sqrt{\frac{\widehat{\text{Var}(\theta)}}{W}}$$

- Si $\widehat{R}$ est grand (plus grand que 1,1) on devrait faire tourner la chaîne un peu plus longtemps jusqu'à atteindre la convergence.
- Si on a plusieurs paramètres à estimer, il faut alors calculer un facteur pour chaque paramètre.
- Il faut aussi faire tourner la chaîne un peu plus longtemps pour que $\widehat{R}$ devient de plus en petit.
- Commande `R` : (package `coda`) `gelman.diag`, `gelman.plot`

Diagnostique de Geweke

- Cette méthode est basée sur un test d'égalité des moyennes entre une première partie de la chaîne (par défaut les premiers 10%) et une seconde partie (les 50% dernières simulations de la chaînes).
- $Z-$score est alors calculé à partir de la différence des moyennes empiriques divisées par l'écart-type empirique.
- Sous l'hypothèse nulle (les deux parties de la chaînes suivent la même loi de probabilité a posteriori), Z suit une loi normale $\mathcal{N}(0,1)$.
- Commande `R` : (package `coda`) `geweke.diag`

Calcul de probabilité a posteriori

- Cas simple
  Supposons que l'on soit intéressé par calculer la probabilité a posteriori que θ soit supérieure à 0.5. Il est alors nécessaire de créer une nouvelle variable, e.g. p, dans le modèle OpenBUGS via le code :
  ```
  p <- step(pi - 0.5)
  ```
  La moyenne a posteriori de p fournira le résultat voulu.
- Cas complexe
  On souhaite calculer la probabilité a posteriori que θ appartienne à $[0.2, 0.5]$. Il est alors nécessaire de créer trois nouvelles variables, e.g. p1, p2 et p3, dans le modèle OpenBUGS via le code :
  ```
  p1 <- step(pi - 0.2)
  p2 <- step(0.5 - pi)
  p3 <- p1*p2
  ```
  La moyenne a posteriori de p3 fournira le résultat voulu.

Utilisation de R2OpenBUGS

```
> require(R2OpenBUGS)
> data=list(n=11, y=c(1,0,0,0,1,0,0,0,1,0,0),a=1,b=1)
> inits1=list(pi=0.2)
> inits2=list(pi=0.6)
> inits3=list(pi=0.9)
> parameters.to.save<-("pi")
> inits=list(inits1,inits2,inits3)
> fit<-bugs(data=data,parameters.to.save="pi",
+ inits=inits,
+ model.file="betabinomialmodel.txt",debug=F,
+ n.chains=length(inits),
+ n.iter=11000,
+ n.burnin=1000,
+ n.thin=1,
+ working.directory=getwd())
```

Exemples Applications OpenBUGS

- **Données :** Durées en minutes de 20 coureurs pour parcourir le Marathon de New-York, notée $(y_i)_{i=1,\ldots,20}$

```
> marathontimes
   time
1   182
2   201
3   221
4   234
.
.
19  359
20  365
```

- **Objectifs :**
  - Estimer la durée moyenne du parcours, estimer la variance,
  - Donner des intervalles de confiance/crédibilité,
  - Calcul de probabilités.

Approche Fréquentiste

Si on suppose que les $(y_i)_{i=1,\dots,20}$ est un échantillon i.i.d. $\mathcal{N}(\mu, \sigma^2)$, alors

- les estimateurs de μ et de σ^2 sont

$$\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \widehat{\mu})^2.$$

- Distribution des estimateurs

$$\widehat{\mu} \sim \mathcal{N}\left(\mu, \frac{1}{n}\sigma^2\right), \quad (n-1)\frac{S^2}{\sigma^2}\chi^2(n-1), \quad \sqrt{n}\frac{\widehat{\mu}-\mu}{S} \sim \mathcal{T}(n-1)$$

Estimation

```
> mean(marathontimes[,1])
[1] 277.6
> var(marathontimes[,1])
[1] 2454.042
> t.test(marathontimes,conf.level=0.95)$conf.int
[1] 254.4154 300.7846
attr(,"conf.level")
[1] 0.95
> source("D:\\Enseignements2012\\StatBayesienne
+ \\sigma2tools.r")
> sigma2.test(marathontimes[,1],conf.level=0.95)$conf
[1] 1419.285 5235.133
attr(,"conf.level")
[1] 0.95
```

Approche Bayésienne

- **Le modèle bayésien** :

$$\begin{cases} y_i \mid \mu, \tau & \sim & \mathcal{N}(\mu, \tau^{-1}) \\ \mu & \sim & \mathcal{N}(\mu_0, \tau_0^{-1}) \\ \tau & \sim & \text{Gamma}(a; b) \end{cases}$$

- **Les lois a priori** : On considère des lois a priori non-informatives sur μ et τ :

$$\mu_0 = 0, \quad \tau_0 = 10^{-6}, \quad \text{et} \quad a = b = 10^{-6}.$$

- Problématique : Estimer μ, τ, Calcul de l'estimation a posteriori que μ soit supérieur à 290 minutes.

Régression logistique

- **Données** : On observe à chaque niveau de dose (en log g/ml), le nombre d'animaux exposés et le nombre d'animaux décédé à chaque niveau de dose.

Dose	Sample size	Deaths
-0,86	5	0
-0,30	5	1
-0,05	5	3
0,73	5	5

- **Problématique** :
  - Estimer la probabilité de décès en fonction de la dose.
  - Quel est le niveau de dose pour que la probabilité de décès ne dépasse pas les 50%.

Modèle statistique

- On note y_i le nombre d'animaux morts parmi les n_i exposés au niveau de dose x_i.
- $y_i \sim$ Binomiale(n_i, π_i) où π_i est la probabilité de décès.
- On suppose la relation suivante entre π_i et x_i

$$\text{logit}(\pi_i) = \alpha + \beta x_i.$$

- **Vraisemblance** :

$$p(y \mid \alpha, \beta) \propto \prod_{i=1}^{4} \pi_i^{y_i} (1 - \pi_i)^{n_i - y_i}$$

$$\propto \prod_{i=1}^{4} \frac{(\exp(\alpha + \beta x_i))^{y_i}}{(1 + \exp(\alpha + \beta x_i))^{n_i}}$$

Mise en œuvre sous R de l'approche fréquentiste

```
> x=c(-.86,-.3,-.05,.73)
> n=rep(5,4)
> y=c(0,1,3,5)
> obj= glm(cbind(y,n-y)~x, family="binomial")
> summary(obj)
Call:
glm(formula = cbind(y, n - y) ~ x, family = "binomial")
Deviance Residuals:
       1         2         3         4
-0.17236   0.08133  -0.05869   0.12237
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.8466     1.0191   0.831    0.406
x             7.7488     4.8728   1.590    0.112
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 15.791412  on 3  degrees of freedom
Residual deviance:  0.054742  on 2  degrees of freedom
AIC: 7.9648
Number of Fisher Scoring iterations: 7
```

Modèle Bayésien

- Il est nécessaire de fournir des lois a priori sur les paramètres α et β.

- Un choix standard consiste à considérer que la distribution a priori pour α (et pour β) est suivant une distribution normale dont la moyenne et la précision sont fixées par l'utilisateur. En cas d'absence d'information a priori, la moyenne peut être fixée à 0 et la précision à une très petite valeur.

$$\begin{cases} y_i \mid \pi_i & \sim & \text{Binomial}(n_i; \pi_i) \\ \text{logit}(\pi_i) & = & \alpha + \beta x_i \\ \alpha & \sim & \mathcal{N}(\mu_\alpha, \tau_\alpha^{-1}) \\ \beta & \sim & \mathcal{N}(\mu_\beta, \tau_\beta^{-1}) \end{cases}$$

Exercice

On considère le code BUGS suivant

```
model
{
x ~ dunif(-1, 1)
y ~ dunif(-1, 1)
0 ~ dbern(constraint)
constraint <- step(x * x + y * y - 1)
}
```

1. Faites tourner le code dans OpenBUGS et tracer le diagramme de dispersion $x \times y$ quand la variable O prend la valeur 1 et 0.
2. Montrer que la loi du couple (x, y) sachant $O = 1$ ou 0 est une loi uniforme dans un sous ensemble de $[-1, 1] \times [-1, 1]$.
3. Utiliser ce code pour simuler un couple de variable aléatoire de loi uniforme sur un anneau de $[-1, 1] \times [-1, 1]$.

Exercice

On considère les données sur 27 vaches marines (dugongs, halicores) Age $\times$ Taille et on s'intéresse à estime leur courbe de croissance.

Données :

```
list(x = c( 1.0,  1.5,  1.5,  .......
          .... 17.0, 22.5, 29.0, 31.5),
     Y = c(1.80, 1.85, 1.87, ..............
           2.56, 2.70, 2.72, 2.57), N = 27)
```

On considère le modèle suivant : $y_i \sim \text{Normal}(\mu_i, \tau),\ i = 1, \ldots, 27$ tel que $\mu_i = \alpha - \beta\gamma^{x_i},\ \alpha > 1,\ \beta > 1$ et $0 < \gamma < 1$.
En considérant des lois a priori non-informatives, donner une estimation à l'aide de OpenBUGS du modèle ? Tracer l'estimation de la courbe de croissance.

Exercice

- On considère le nombre de désastres, par an, qui se sont produits dans les mines de charbons en GB entre les années 1851 et 1962. Des progrès ont été réalisé dans la sécurité des mines à partir de 19ième siècle où on a observé une nette diminution du nombre des désastres.
- Objectif du problème est estimer l'année τ où la baisse des incidents est devenue notable (ou le contraire).
- Le modèle bayésien :
  - y_t est le nombre de désastres à l'année t où $t =$ Année actuelle $-$ 1850. On suppose

$$y_t \sim \text{Poisson}(\mu_t), \quad \text{où } \log(\mu_t) = \beta_0 + \beta_1 \mathbb{1}\{t \geq \tau\}$$

  - Lois a priori : β_j, $j = 0, 1$ des lois a priori non-informatives à partir de la loi normale et la loi uniforme sur τ dans $[1, N]$ N étant le nombre d'année d'études.

La déviance

- Pour une fonction vraisemblance $l(y \mid \theta)$ la déviance est définie par
$$D(\theta) = -\log l(y \mid \theta).$$
- Dans `OpenBUGS` la déviance est calculée automatiquement pour chaque simulation du paramètre θ.
- La constante de normalisation de $l(y \mid \theta)$ est incluse dans le calcul de la déviance.
- Exemple, si $y_i \sim \text{Bin}(\theta_i, n_i)$ la déviance est
$$-2\left[\sum_i y_i \log\theta_i + (n_i - y - i)\log(1-\theta_i) + \log\begin{pmatrix} n_i \\ y_i \end{pmatrix}\right]$$

Comparaison des modèles bayésiens utilisant DIC

- La mesure de la complexité est faite en estimant le "nombre effectif de paramètres" :

$$p_D = E_{\theta|y}[D(\theta)] - D\left[E_{\theta|y}[\theta]\right] = \overline{D} - D(\bar{\theta})$$

- Critère d'information de la déviance (*Deviance Information Criteria*, DIC)

$$DIC = D(\bar{\theta}) + 2p_D = \overline{D} + p_D.$$

  Modèles avec des valeurs de DIC faibles ajustent les données.

- Dans `OpenBUGS`, le DIC peut être calculé à partir de `Inference/DIC`.

Exemple

- Supposons qu'on observe $r = 1$ succès dans $n = 2$ tirage de Bernoulli, donc $r \sim \text{Bin}[\theta, n]$.
- On considère une reparamétrisation du modèle $\psi = \theta^a$. Si $\theta \sim [0, 1]$, alors $\psi \sim \text{Beta}(a^{-1}, 1)$.
- Nous allons faire tourner le modèle sous OpenBUGS avec trois valeurs différentes de a ($a = 1, 5, 20$).

```
r <- 1;  n<- 2; a[1]<-1 ; a[2] <- 5; a[3] <- 20
for (i in 1:3){
a.inv[i]<- 1/a[i]
theta[i] <- pow(psi[i], a.inv[i])
psi[i]  ~ dbeta(a.inv[i] , 1)
}
r1<- r; r2<-r ; r3 <- r
r1 ~ dbin(theta[1],n)
r2 ~ dbin(theta[2],n)
r3 ~ dbin(theta[3],n)
```

Exercice

On considère le code WinBUGS suivant

```
model  {
for  (i  in  1:N)  {
y[i]  ~  dnorm(mu[i],p[i])
p[i]  <-  tau*lam[i]
lam[i]  ~  dgamma(2,2)
mu[i]  <-  b[1]+b[2]*x[i]}
for  (j  in  1:2)  {b[j]  ~  dnorm(0,0.001)}
tau  ~  dgamma(0.001,0.001)
%}
```

1. Ecrire le modèle Bayésien correspondant à ce code.

Exercice

2 Ce modèle est utilisé pour comprendre la relation entre le nombre de votes du candidat Pat Buchanan (2000, partie réformateur) en fonction du nombre des votes de Ross Perot (1996). En effet, y_i (Buchana) et x_i (Perot) sont les racines du nombre de votes.
Ecrire le code `R` qui permet de simuler la chaîne de Markov des paramètres du modèle.

3 Ecrire un code `R` pour tracer une bande de confiance de la ligne de régression.

Exercice

Que permet de calculer le code OpenBUGS suivant ?

```
 model{
for(i in 1:2){
 r[i] ~ dbin(p[i],n[i])
}
for (i in 1:2){ p[i] ~ dunif(0,1)}
delta <- p[1] - p[2]
delta.up <- step(delta)
lambda <- log( (p[1]/(1-p[1])) / (p[2]/(1-p[2])) );
lambda.up <- step(lambda)
}
## data
list(r=c(83,72),n=c(86,86))
```