TP0 Régression bayésienne

ENSAI - 2022/2023

L'objectif de ce TP est de s'initier à la programmation bayésienne. Nous étudierons le modèle linéaire suivant :

$$Y = \mu\mathbb{I} + X\beta + \epsilon$$

Nous générons un jeu de données de 200 observations. Nous découperons en deux l'échantillon pour avoir un jeu «d'apprentissage » qui servira à construire les modèles et un jeu « test » qui servira à comparer nos modèles. On coupera ainsi l'échantillon en deux : 100 observations pour l'apprentissage et 100 pour le test. On génère 300 variables explicatives i.i.d., chacune suivant une loi uniforme entre -5 et 5 (on ne standardisera pas car i.i.d.).

```
simuvarexpl <- matrix(rep(0,300*200),ncol=300,nrow=200)
for (i in 1 :300) simuvarexpl[,i] <- runif(200,-5,5)
simuvarexpl <- as.data.frame(simuvarexpl)
```

Pour générer un vecteur d'observation Y nous utilisons uniquement 5 des 300 variables générées : celles d'indices 10, 20, 30, 40 et 50, que l'on associe au vecteur de paramètres de régression $\beta = (1, -1, 2, -2, 3)$. Nous supposons que $\mu = 0$.

```
trueind <- c(10,20,30,40,50)
beta <- c(1,-1,2,-2,3)
ysimu <- as.matrix(simuvarexpl)[,trueind]%*% beta + rnorm(200,0,2)
```

1 Fonctions utiles

1.1 Prédictions

La fonction suivante sera utilisée pour prédire une variable d'intérêt à partir de variables explicatives et à partir d'estimateurs de μ et β (dans le modèle linéaire).

```
predictions <- function(matableTest,muchap,betachap){
ychap <- muchap * rep(1,dim(matableTest)[1])+ as.matrix(matableTest[,]) %*% betachap
return(ychap)}
```

2 Simulation d'un coefficient aléatoire

On suppose ici que $\mu = 0$ (modèle sans constante). On considère les lois suivantes :

$$\beta \sim \mathcal{N}(0, G) \qquad \epsilon \sim \mathcal{N}(0, R)$$

avec $G = \sigma_\beta^2 I$ et $R = \sigma_\epsilon^2 I$ (I désigne l'identité).

- Calculer la loi de $Y|\beta$.
- Calculer la loi a posteriori de $\beta|Y$.
- Simuler par Gibbs sampler la chaîne de Markov associée à $\beta|Y$.
- Sensibilité : étudier l'impact de la valeur de σ_β^2 sur l'estimation de $\beta|Y$.
- Représentez des trajectoires de "vrais" coefficients et de "faux" coefficients.

3 Ajout d'un a priori de Zellner

On suppose ici que $\mu = 0$ (modèle sans constante). On considère les lois suivantes :

$$\beta \sim \mathcal{N}(0, \sigma^2(X'X)^{-1}) \qquad \epsilon \sim \mathcal{N}(0, R)$$

avec $R = \sigma_\epsilon^2 I$. En général on choisit $\sigma^2 = c\sigma_\epsilon^2$. Proposez une version ridge de la loi de β pour pouvoir simuler la loi a posteriori de $\beta|Y$.

4 Ajout d'une loi sur σ_ϵ^2

On suppose ici que $\mu = 0$ (modèle sans constante). L'inconvénient des modèles précédents est de ne pouvoir estimer la variance résiduelle σ_ϵ^2. On peut soit faire des estimations RR-BLUP. Soit continuer en bayésien. Ici on ajoute une loi (a priori)sur σ_ϵ

$$\beta \sim \mathcal{N}(0, R) \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I) \quad \sigma_\epsilon^2 \sim \mathcal{IG}(c, d)$$

avec c, d les hyperparamètres de la loi Inverse Gamma. Simulez une chaîne de Markov par Gibbs sampler pour estimer les paramètres du modèle.

5 Ajout d'une loi sur σ_β^2

L'a priori sur β a une forte influence. On peut ajouter des hyperparamètres et s'approcher du Bayes A.

$$\begin{aligned} \beta \sim \mathcal{N}(0, diag(\sigma_{\beta_1}^2, \cdots, \sigma_{\beta_p}^2) \qquad & \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I) \\ \sigma_{\beta_j}^2 \sim \mathcal{IG}(a, b) \qquad & \sigma_\epsilon^2 \sim \mathcal{IG}(c, d) \end{aligned}$$

avec a, b, c, d les hyperparamètres de la loi Inverse Gamma. On prend ici des σ_{β_j} différents pour laisser chaque coefficients varier librement. On peut ajouter également une moyenne μ et une loi vague, par exemple uniforme.

---

6 Exemple de simulation par ABC

On considère le modèle linéaire univarié suivant :

$$Y = \mu_0 + \beta x + \epsilon,$$

avec $Y|\beta \sim \mathcal{N}(\beta x + \mu_0, \sigma^2)$.

On suppose que $\beta \sim \mathcal{N}(0, \sigma_\beta^2)$. On sait donc que $\beta|Y \sim \mathcal{N}(m, v)$, avec $v = (\sum_{i=1}^n x_i^2/\sigma^2 + 1/\sigma_\beta^2)^{-1}$ et $m = v(\sum_{i=1}^n x_i yi/2 + 2\sum_{i=1}^n x_i)/\sigma^2$.

On simule ce modèle pour $n = 20$, avec $\sigma_\beta^2 = 2$, $\sigma^2 = 1$, et $\mu_0 = 1$ (à faire varier). On choisit un design uniforme pour x entre -5 et 5.

On simule notre jeu de données suivant le modèle $y|\beta$, avec $\beta = 1$, donc $Y \sim \mathcal{N}(2x+1, 1)$.

On propose un algorithme ABC pour simuler les valeurs de $\beta|Y$. On va donc proposer des valeurs de β^* suivant une $\mathcal{N}(0, \sigma_\beta^2)$ et les garder seulement si la simulation des $y^*|\beta^*$ est proche des y observés (par rapport à un seuil à faire varier suivant le taux d'acceptation de l'algorithme). On notera K le nombre de propositions.

```
abc=function(x,y,mb,sb,K,seuil)
{
compt=0
n=length(y)
res=rep(NA,n)
for (i in 1 :K)
{
betasim=rnorm(1,mb,sb)
my=x * betasim + m0
ysim=rnorm(n,my,se)
if (mean((y-ysim)**2) < seuil)
{
res[i]=betasim
compt = compt+1
}
}
res=res[ !is.na(res)]
accept=compt/K*100
return(list(res,accept))
}
```

On peut proposer un autre critère qu'une distance entre les y simulées et les y observées, en utilisant par exemple une summary statistic (moyenne, variance, un compromis des deux...).

On comparera la distribution des beta retenus à celle théorique.

Commenter les résultats. Un déséquilibre peut venir du fait que l'on présente plus souvent des β plus à gauche ou plus à droite de la valeur inconnue. Ce déséquilibre s'estompe si on diminue le seuil ou si on augmente la variance. On peut faire varier le seuil, les paramètres, le nombre de propositions.