Université de Strasbourg
M1 - Magistère
Statistique - études de cas

Ségolen Geffray
geffray@math.unistra.fr
Année 2015/2016

Sujet 16: Méthodes de Monte-Carlo par chaînes de Markov (MCMC)

On parle de 'méthodes de Monte Carlo' dès que l'on tente d'approximer une valeur numérique au moyen de tirages aléatoires. Cela peut être utile pour des besoins d'intégration ou d'optimisation. Ainsi, en vertu de la loi des grands nombres et du TCL, on peut imaginer approximer

$$I = \int \varphi(x) f(x) dx,$$

où f est une densité par rapport à la mesure de Lebesgue, par

$$I \approx \frac{1}{n} \sum_{i=1}^{n} \varphi(x_i)$$

avec $(x_1, \ldots, x_n)$ obtenus par des tirages indépendants selon la loi de densité f. On peut également envisager de trouver la solution, du moins approximativement, du problème d'optimisation suivant:

$$\arg\max_x g(x).$$

Si $g(.)$ est une densité, on peut prendre le mode d'une estimation de la densité g, laquelle estimation aura été obtenue à partir de tirages indépendants selon la loi de densité g. Si $g \geq 0$ mais $\int gx)dx \neq 1$, on peut alors normer g pourvu que l'on sache calculer la constante de normalisation $\int g(x)dx$. Si g n'est pas positive, on pourra considérer $\exp(g(.))$ puis normer cette fonction à 1.

Supposons maintenant que l'on veuille générer un échantillon selon une loi admettant la densité f par rapport à une mesure μ positive et σ-finie, qui sera ici la mesure de Lebesgue. Supposons également que l'on ne sait pas simuler des variables de loi de densité f directement par les méthodes usuelles et/ou que la densité f n'est connue qu'à une constante de normalisation près auquel cas on note $\pi(.)$ toute fonction positive proportionnelle à la cible f ie telle que $f(x) = c\pi(x)$ pour tout x et pour une constante c (qui est d'ailleurs donnée par $c = \int \pi(x) d\mu(x)$). Pour contourner ce problème, on a recours aux méthodes de Monte Carlo par Chaînes de Markov (Monte-Carlo Markov Chains en anglais ou MCMC). Le principe spécifique des méthodes MCMC est de construire une chaîne de Markov convergeant en un certain sens (ergodicité) vers la loi d'intérêt f. Dans ce cas, après un certain nombre de tirages, on peut considérer que les réalisations de la chaîne sont des réalisations de variables aléatoires (non-indépendantes toutefois) de loi arbitrairement proche de f.

Définitions:

- Un processus stochastique $(X_k)_{k\in\mathbb{N}}$ à valeurs dans un ensemble E est une chaîne de Markov si, pour tout $k \in \mathbb{N}$, la loi de X_{k+1} sachant le passé du processus jusqu'à l'instant k ne dépend que X_k. Cela s'écrit, pour un ensemble mesurable fixé quelconque $A \subset E$,

$$\mathbb{P}(X_{k+1} \in A|X_1, \ldots, X_k) = \mathbb{P}(X_{k+1} \in A|X_k) \text{ p.s.}$$

  Notons $F(k, X_k, A) := \mathbb{P}(X_{k+1} \in A|X_k)$.
  Lorsque $F(k, X_k, A) = F(X_k, A)$, on dit que la chaine est homogène.

- Lorsque $(X_k)_{k\in\mathbb{N}}$ est une chaine de Markov homogène, la fonction définie pour $x \in E$ et pour tout $A \subset E$ ensemble mesurable par

$$P(x, A) = \mathbb{P}(X_{k+1} \in A|X_k = x)$$

  s'appelle le noyau de transition de $(X_k)_{k\in\mathbb{N}}$.
  La mesure de probabilité $\mathbb{P}(X_0 \in .)$ s'appelle la loi initiale de $(X_k)_{k\in\mathbb{N}}$.

- Etant donnée une loi de probabilité P_0 et un noyau de transition Q, existe-t-il une chaîne de Markov de loi initiale P_0 et de noyau de transition Q? Le théorème de Kolmogorov (que nous n'énonçons pas) permet d'affirmer l'existence et l'unicité d'une telle chaîne de Markov.

- Voici quelques propriétés importantes pour les chaînes de Markov qui nous intéressent, énoncées de manière informelle:
  - stationnarité: si la variable X_k est distribuée selon f, alors les variables $X_{k+1}, X_{k+2}, ...$ sont distribuées suivant f.
  - irréductibilité: tous les ensembles de probabilité non nulle doivent être atteints à partir de n'importe quel point de départ.
  - récurrence: les suites (X_k) passent une infinité de fois dans tout ensemble de probabilité non nulle.
  - apériodicité: le noyau n'induit pas de comportement périodique pour les suites (X_k).

Le principe général est le suivant:

- on se donne une valeur initiale x_0,

- on construit x_{k+1} à partir de x_k à l'aide d'un noyau de transition Q de sorte que la loi cible est f. Se donner un noyau de transition Q revient à se donner une loi conditionnelle de densité notée $q(.|.)$ appelée aussi loi instrumentale ou loi de proposition, qui sert à proposer une valeur y_k au vu de la valeur de x_k et à adopter un schéma décisionnel relatif à l'acceptation ou au rejet de cette proposition. Si la proposition est rejetée, on conserve la valeur x_k ie on pose $x_{k+1} = x_k$. Si la proposition est acceptée, on pose $x_{k+1} = y_k$.
  NB: Le noyau de proposition Q est valide si $\text{supp}(\pi) \subseteq \cup_x \text{supp}(Q(.|x))$.
  NB2: plus $q(.|.)$ est proche de f, meilleure sera la convergence de l'algorithme...

- à la convergence ie pour k grand, mettons pour $k \geq k_0$, chaque valeur x_k est la réalisation d'une variable X_k distribuée (approximativement) selon la loi de densité f.

NB: les variables X_{k_0}, X_{k_0+1},... ne sont pas indépendantes! Cependant, des résultats théoriques permettent d'affirmer que pour des besoins d'intégration ou d'optimisation, ces simulations sont satisfaisantes.

• **Algorithme de Metropolis (1953)-Hastings (1970)**:

- Initialisation : choisir x_0.
- A chaque étape $k \in \mathbb{N}$:
  - Simuler une valeur y_k de $Y_k \sim q(.|x_k)$.
  - Simuler une valeur u_k de $U_k \sim \mathcal{U}([0,1])$.
  - Poser
$$x_{k+1} = \begin{cases} y_k & \text{si } u_k \leq \rho(x_k, y_k) \\ x_k & \text{sinon,} \end{cases}$$
où $\rho(x_k, y_k) = \min\left(1, \dfrac{\pi(y_k)q(x_k|y_k)}{\pi(x_k)q(y_k|x_k)}\right)$.
- NB: L'algorithme de Metropolis-Hastings ne fait intervenir que les rapports $\dfrac{\pi(y_k)}{\pi(x_k)}$ et $\dfrac{q(x_k|y_k)}{q(y_k|x_k)}$. La connaissance des constantes de normalisation n'est donc pas nécessaire.

Cas particulier: Algorithme de Metropolis-Hastings indépendant: $q(.|x) = g(.)$ pour tout x

- Initialisation : choisir x_0.
- A chaque étape $k \in \mathbb{N}$:
  - Simuler une valeur y_k de $Y_k \sim g(.)$.
  - Simuler une valeur u_k de $U_k \sim \mathcal{U}([0,1])$.
  - Poser
$$x_{k+1} = \begin{cases} y_k & \text{si } u_k \leq \rho(x_k, y_k) \\ x_k & \text{sinon,} \end{cases}$$
où $\rho(x_k, y_k) = \min\left(1, \dfrac{\pi(y_k)g(x_k)}{\pi(x_k)g(y_k)}\right)$.

Cas particulier: Algorithme de Metropolis-Hastings à marche aléatoire: $q(y|x) = g(y - x)$

- Initialisation : choisir x_0.
- A chaque étape $k \in \mathbb{N}$:
  - Simuler une valeur e_k de $\varepsilon_k \sim g(.)$.
  - Simuler une valeur u_k de $U_k \sim \mathcal{U}([0,1])$.
  - Poser $y_k = x_k + e_k$.
  - Poser
$$x_{k+1} = \begin{cases} y_k & \text{si } u_k \leq \rho(x_k, y_k) \\ x_k & \text{sinon,} \end{cases}$$
où $\rho(x_k, y_k) = \min\left(1, \dfrac{\pi(y_k)g(x_k - y_k)}{\pi(x_k)g(y_k - x_k)}\right)$.

Dans le cas d'un noyau symétrique (i.e. $g(t) = g(-t)$, par exemple $g =$ densité d'une loi normale centrée), on obtient la simplification: $\rho(x_k, y_k) = \min\left(1, \frac{\pi(y_k)}{\pi(x_k)}\right)$.

• **Algorithme du recuit simulé (1953)**:
Cet algorithme a pour objectif de maximiser une fonction réelle g. Il utilise l'algorithme de Metropolis-Hastings pour simuler la loi $f(x) \propto \exp(g(x))$. Pour estimer du(des) mode(s), on change la loi objectif à chaque étape k de l'algorithme : $f_k(x) \propto e^{g(x)/T_k}$. La suite de paramètres $(T_k)_{k\in\mathbb{N}}$ est une suite décroissante de températures positives qui tend vers 0. En pratique, la température est choisie élévée dans les premières itérations pour pouvoir s'extraire des bassins des minima locaux, puis elle est prise à décroissance lente jusqu'à tendre vers 0. Un choix possible est $T_k = T_0\beta^k$ avec $0 < \beta < 1$.

Algorithme du recuit simulé:

- Initialisation : choisir x_0.
- A chaque étape $k \in \mathbb{N}$:
  - Simuler une valeur y_k de $Y_k \sim q(.|x_k)$.
  - Simuler une valeur u_k de $U_k \sim \mathcal{U}([0,1])$.
  - Poser
$$x_{k+1} = \begin{cases} y_k & \text{si } u_k \leq \rho(x_k, y_k) \\ x_k & \text{sinon,} \end{cases}$$
où $\rho(x_k, y_k) = \min\left(1, \frac{\exp(-g(y_k))q(x_k|y_k)}{\exp(-g(x_k))q(y_k|x_k)}\right)$.
  - Faire décroître la température de T_k à T_{k+1}.

Exercice 1.

1. Faire tourner l'algorithme de Metropolis-Hastings indépendant avec la loi cible de votre choix et avec les lois de proposition suivante: $\mathcal{U}(-4,4)$, $\mathcal{U}(-10,10)$, $\mathcal{N}(0,1)$, $\mathcal{N}(0,20)$. Placer sur un graphique les points (k, x_k) pour $k = 0, \ldots, 20000$. Tracer les 4 histogrammes obtenus à partir des x_k pour $k = 1, \ldots, 5000$, puis $k = 5001 \ldots, 10000$, puis $k = 10001 \ldots, 15000$, puis $k = 15001 \ldots, 20000$. Calculer le taux d'acceptation de l'algorithme. Que constatez-vous?

2. Faire tourner l'algorithme de Métropolis-Hastings indépendant avec $q(y) = \mathcal{N}(0, \sigma^2)$ pour simuler des variables de loi cible $f = \mathcal{N}(0,1)$. Verifier empiriquement la qualité de la simulation. Comment la valeur de σ^2 influe-t-elle sur la simulation?

3. Faire tourner l'algorithme de Métropolis-Hastings à marche aléatoire avec la loi de proposition $\mathcal{N}(0,1)$ pour simuler des variables de loi cible $f \propto \exp(-c|x|^a)$, avec différents choix de a et c.

4. Soit la fonction $h : x \to \sin(100/x)\exp(-(x-1)^2)$. Tracer le graphe de cette fonction sur $[0, 5]$. Déterminer le minimum de la fonction h dans $[0, 5]$ avec la fonction `fminsearch` du package `pracma` du logiciel `R`. Déterminer le minimum de la fonction h dans $[0, 5]$ avec l'algorithme du recuit simulé avec la loi $\mathcal{N}(1.5, 10)$ comme loi de proposition. Qu'en pensez-vous?

5. Soit la fonction $h : x \to (x-3)(x+6)(1+\sin(60x))$. Tracer le graphe de cette fonction sur $[-10, 10]$. Déterminer le(s) zéro(s) de la fonction h dans $[-2, 10]$ et dans $[-8, 1]$ avec la fonction `uniroot` du logiciel `R`. Déterminer le(s) zéro(s) de la fonction h dans $[-2, 10]$ et dans $[-8, 1]$ avec l'algorithme du recuit simulé. Qu'en pensez-vous?

6. Déterminer $\arg\max_x h(x)$ avec $h(x) = (\cos(50x) + \sin(20x))^2$ au moyen de l'algorithme du recuit simulé.

Pistes de réflexion:

- Etudier l'impact du choix de la valeur initiale sur la convergence de l'algorithme.
- Etudier l'impact du choix de la loi de proposition sur la convergence de l'algorithme.
- Etudier l'impact du choix de la température initiale, du schéma de descente de température, de la température d'arrêt sur la convergence de l'algorithme.