Simulation et Monte Carlo: projets 2018-19

Nicolas Chopin

Vous devez former un groupe de trois étudiants au sein de votre groupe de TD, choisir un des projets suivants, et le traiter d'ici la dernière séance de TD, où vous ferez une présentation orale de 15 minutes devant les autres étudiants. Pas besoin de rendre un rapport rédigé: vous pouvez cependant rendre le jour de la soutenance un document contenant certains graphiques et résultats, mais surtout, vous devez envoyer vos programmes à votre chargé de TD, qui vérifiera que ce programme fonctionne bien.

Vous avez tout à fait le droit et même je vous encourage à chercher sur internet ou dans la littérature scientifique des inspirations pour effectuer votre projet. Une seule obligation: citez vos sources!

Si vous bloquez, contactez votre chargé de TD (qui me contactera si nécessaire).

Point Essentiel: toujours évaluer (d'une façon ou d'une autre: intervalles de confiance, box-plots, etc.) l'erreur de Monte Carlo de vos résultats. Dans le cas du MCMC, pensez aussi aux ACF (graphe de la fonction d'autocorrélation pour chaque composantes) et aux "traces" (valeur de chaque composante en fonction du temps, notamment pour déterminer le burn-in). Faites preuve d'un esprit scientifique!

Les questions bonus sont facultatives: elles sont réservées aux étudiants qui veulent s'investir plus dans leur projet. Leur bonne résolution sera récompensée par une meilleure note, mais uniquement si le reste du projet a été bien traité.

Black rain

Le jeu de données "tokyo rainfall" est disponible notamment ici: https://bitbucket.org/hrue/r-inla/src/0d9389b41497/r-inla.org/examples/tokyo/

Il contient le nombre de jours de pluie (0, 1, 2) pour chaque jour de l'année, sur la période 1983-1984. Ces données peuvent être modélisées comme suit: $Y_t \sim \text{Binomial}(2, p_t)$ avec $p_t = 1/(1+\exp(-X_t))$, où X_t suit une marche aléatoire: $X_t = X_{t-1} + \sigma U_t$, $U_t \sim N(0,1)$. (On peut prendre $X_0 \sim N(0, \sigma^2)$ pour faire simple.) On cherche à déterminer la loi a posteriori du paramètre σ, pour une loi a priori de type inverse-gamma; $1/\sigma^2 \sim \text{Gamma}(a, b)$, avec par exemple $a = b = 1$.

1. Calculer les lois conditionnelles de X_t sachant X_s, $s \neq t$, et σ, et σ sachant tous les X_t.
2. Proposer et implémenter un algorithme de type Gibbs pour simuler selon la loi a posteriori du modèle. (Pour X_t sachant les autres variables, vous pouvez considérer plusieurs alternatives; notamment un algorithme de rejet, ou un pas de Metropolis, expliquez et comparez).

3. Même exercice pour $X_t = \rho X_{t-1} + \sigma U_t$, où ρ est un paramètre additionnel, de loi a priori uniforme sur $[-1, 1]$. Quelle loi proposez-vous dans ce cas pour X_0? Justifier.

4. Bonus: proposer un algorithme de type Metropolis-within-Gibbs, où tous les X_t seraient modifiés "en bloc". Plusieurs solutions possibles; par exemple vous pouvez prendre inspiration de la méthode "over-relaxation": pour simplifier, considérer le bloc $U_{1:T}$ (qui est une transformation bijective du bloc $X_{1:T}$ pour $\theta = (\rho, \sigma)$ fixé), et la loi de proposition, qui remplace chaque U_t par $\rho U_t + \sqrt{1-\rho^2} V_t$ avec $V_t \sim N(0, 1)$. Montrer que cette loi de proposition laisse la loi **a priori** invariante, et que le rapport de Metropolis se simplifie en un rapport de vraisemblance; essayer plusieurs valeurs de ρ et discuter.

   (D'autres lois de propositions sont possibles, comme par exemple une normale multivariée construite à partir d'un algorithme de maximisation, comme discuté en cours.)

The silence of the lambs

Le jeu de données "lamb" (disponible par exemple dans le package label.switchhing sur CRAN) est une série temporelle du nombre de mouvements d'un agneau dans le ventre de sa mère sur 240 pas de temps consécutifs. On le modélise couramment sous la forme d'un mélange de lois de Poisson: soit $Y_t | X_t = k \sim \text{Poisson}(\lambda_k)$ où X_t est la composante du mélange; soit $q_k = P(X_t = k)$ et $K \geq 2$ le nombre de composantes.

1. Construire un Gibbs sampler pour l'estimation des paramètres q_k et λ_k. Bien donner les détails. (Proposer au préalable des lois a priori donnant des lois conditionnelles simples à simuler.)

2. On considère maintenant un modèle de type "hidden Markov": (X_t) est une chaîne de Markov homogène à K états, caractérisée par les probabilités $q_{kl} := P(X_t = l | X_{t-1} = k)$. Adapter le Gibbs sampler de la question précédente à ce nouveau modèle.

3. Mettre en oeuvre les deux algorithmes, et déterminer leur performance, notamment en fonction de K.

4. Question bonus: montrer qu'il est aussi possible de construire un Gibbs sampler qui génère 'en bloc' (de façon jointe) les variables latentes $X_1, \ldots, X_T$, expliquer son intérêt par rapport à l'algorithme initial, et comparer numériquement. On pourra se baser sur l'article suivant: http://infohost.nmt.edu/~olegm/489/Scott2002HMM.pdf

Metropolis

On considère la loi a posteriori d'un modèle de régression logistique, i.e. pour n individus, on observe (X_i, Y_i) à valeurs dans $\mathbb{R}^p \times \{-1, 1\}$, et la probabilité que $Y_i = 1$ est égale à $1/(1 + \exp(-\beta^T X_i))$. (La première composante des X_i peut être prise comme égale à 1, pour inclure un "intercept".) On pourra prendre comme jeu de données "German credit", disponible sur le site UCI: https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data) et pour la loi a priori une loi impropre de densité constante.

1. Comparer les performances des deux algorithmes suivants:

(a) random walk Metropolis, avec une matrice de covariance pour la loi de proposition égale τI_p, et τ choisi "à la main" (en se basant sur des simulations préliminaires);
(b) une version adaptative de l'algorithme précédent, où la matrice de covariance de la marche aléatoire est adaptée à chaque étape, comme expliqué par exemple dans l'article suivant: [https://people.eecs.berkeley.edu/~jordan/sail/readings/andrieu-thoms.pdf]

2. Lire l'article suivant: https://arxiv.org/abs/1206.1901 et vérifier qu'il est possible d'appliquer l'algorithme HMC (Hamiltonian Monte Carlo) pour simuler la même loi a posteriori. Comparer aux algorithmes précédents, pour différentes valeurs des paramètres (L, ε) (où L est le nombre d'étapes "leapfrog" à chaque itération, et $\varepsilon > 0$ est le pas de discrétisation).

The social network

On représente un réseau de n individus par un graphe à n sommets, et des arrêtes aléatoires ($X_{ij} = 1$ si les individus i et j sont reliés par une arrête). On modélise la loi jointe des X_{ij} de la façon suivante:

$$p(x|\theta) = \frac{1}{Z(\theta)} \exp\{\theta S(x)\}$$

où la statistique résume certaines caractéristiques du réseau; par exemple, $S(x) = (\sum_{i<j} x_{ij}, \sum_{i<j<k} x_{ij}x_{jk}x_{ik})$ (nombre d'arrêtes, et nombre de triangles).

1. Proposer et mettre en oeuvre un algorithme pour simuler selon $p(x|\theta)$ pour θ fixé.

2. On cherche maintenant à estimer θ par maximum de vraisemblance, à partir d'un x (unique) observé. (On pourra par exemple considérer le jeu de données des mariages entre familles florentines: http://www.casos.cs.cmu.edu/computational_tools/datasets/sets/padgett/) Exprimer la vraisemblance du modèle, et montrer que la constante de normalisation, non calculable, peut être approchée (à une constante près) par une méthode d'importance sampling, où la loi de proposition est la loi du modèle pour un certain θ_0. (Il faut donc utiliser le Gibbs sampler de la question précédente pour simuler cette loi.) Mettre en oeuvre une méthode de maximisation de la fonction ainsi obtenue (où la constante de normalisation est remplacée par une approximation de Monte Carlo; noter que l'approximation varie selon θ, mais que les simulations du Gibbs sampler sont générées une fois pour toute.) On pourra par exemple utiliser une méthode de type gradient. (L'approche décrite dans cette section est communément appelée MC-MLE ou MCMC-MLE.)

3. Bonus: chercher dans la littérature un meilleur algorithme pour simuler selon $p(x|\theta)$ (par exemple l'algorithme "tie-no-tie" dans "Specification of exponential-family random graph models: terms and computational aspects" de Morris et al)

Wall street

On cherche à évaluer le prix d'une option asiatique:

$$C = \mathbb{E}\left[e^{-rT}\left(\frac{1}{k}\sum_{i=1}^{k} S(t_i) - K\right)^+\right]$$

avec $t_0 = 0 < t_1 < \ldots < t_k = T$, $r > 0$, K fixe, et S un processus défini sur $[0, T]$; notation: $x^+ = \max(x, 0)$.

Pour faire simple, on considère le modèle de Black et Scholes:

$$S(t) = \exp\{(\mu - \sigma^2/2)t + \sigma W_t\}$$

où W_t est le mouvement brownien. On pourra prendre par ex. $T = 1$, $r = 0.05$, $\sigma = 0.3$, $K = 5$, $k = 20$, $t_i = i/20$. Notez que ce processus ne nécessite pas de discrétisation d'Euler, mais nous allons ignorer ce point, et le traiter comme un processus général qu'il n'est pas possible de simuler sans erreur de discrétisation.

1. Comparer différentes méthodes de calcul de C en combinant les différentes méthodes de réduction de variance vues en cours: variables antithétiques, variables de contrôle, et Quasi-Monte Carlo. Pour QMC, je vous conseille fortement d'utiliser des packages qui implémentent déjà ces méthodes, comme le package randtoolbox sous R. Pensez aussi à la variante RQMC. Vous pouvez essayer aussi de faire varier les différents paramètres pour voir à quel point vos conclusions en dépendent.
2. Comparer avec MLMC (Multi-Level Monte Carlo), méthode expliquée en cours, et dans l'article suivant http://statweb.stanford.edu/~owen/courses/362/readings/GilesMultilevel.pdf
3. Reprendre la comparaison en se basant sur du Quasi-Monte Carlo.

Travelling salesman

Ce projet est inspiré du Chapitre 4 du livre "The Cross-Entropy Method" de Rubinstein et Kroese.

Le problème du voyageur de commerce (en anglais, TSP, travelling salesman problem) revient à minimiser le trajet effectué entre M villes, sous la contrainte de passer par chaque ville exactement une fois. Soit D la matrice telle que D_{ij} est la distance entre la ville i et la ville j; noter que $D_{ii} = 0$ et D est symétrique.

1. Rappeler (ou retrouver!) la méthode permettant de simuler efficacement N vecteurs selon la loi Multinomiale définie par le vecteur de probabilité $p = (p_1, \ldots, p_M)$. La mettre en œuvre.
2. Utiliser l'algorithme CE vu en cours pour résoudre le problème TSP. Pour ce faire, proposer une famille paramétrique de lois pour la simulation de trajets. Programmer cet algorithme, et le tester sur quelques exemples.
3. Concernant l'estimation bayésienne de vecteurs de probabilité d'une loi multinomiale, retrouver le fait que la famille des lois de Dirichlet est conjuguée pour ce modèle. Expliquer comment en pratique vous pouvez utiliser cette propriété pour améliorer votre algorithme.
4. Bonus: en fonction du temps disponible, vous pouvez tester votre algorithme sur de "gros" problèmes (par ex. trajet optimal du tour de France en se basant sur la liste des villes de l'année dernière), ou comparer à d'autres algorithmes que vous trouverez dans la littérature.

Catch me if you can

La méthode statistique dite de "capture-recapture" est couramment utilisée en écologie pour estimer la taille M d'une population animale. Dans un premier temps, une partie de la population est capturée, marquée et relâchée. Dans un deuxième temps, une autre partie est capturée et le nombre d'individus marqués dans l'échantillon est compté. Le modèle correspondant s'écrit sous la forme suivante:

$$m_1 \sim \text{Bin}(M, p), \quad m_{12}|m_1 \sim \text{Bin}(m_1, p), \quad m_2|m_1 \sim \text{Bin}(M - m_1, p)$$

où p la probabilité de capture, m_1 (resp. m_2) le nombre d'animaux capturés uniquement à la première (resp. deuxième) étape, et m_{12} le nombre d'animaux qui ont été capturés les deux fois.

1. Prenant pour loi a priori la loi impropre $\pi(p, M) \propto 1$, calculer les lois conditionelles de p et M sachant les données (i.e. la loi de p sachant M et les données, puis la loi de M, sachant p et les données.)
2. Expliquer comment simuler les lois que vous avez obtenu, et programmer l'algorithme correspondant.
3. Mettre en oeuvre l'algorithme de Gibbs correspondant (en se basant sur les algorithmes de simulation obtenus en 2), pour $m_1 = 22$, $m_2 = 60$, $m_{12} = 11$. (Données réelles pour une expérience sur des passereaux.)
4. Bonus: en fonction du temps disponible, vous pouvez consulter le Chapitre 5 de "Bayesian Essentials with R" de Marin et Robert, pour voir comment généraliser aux modèles à k étapes, et mettre en oeuvre l'algorithme de Gibbs correspondant, sur les données Eurodip.