Simulation et Monte Carlo: projets 2021-22

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.

Seeds

Le jeu de données Seeds (disponible ici donne la proportion de graines (de différents types) ayant germé dans 21 pots. On considère le même modèle que celui proposé dans la page où vous pouvez télécharger les données, soit: n_i $Bin(p_i, r_i)$, où r_i (nombre de graines) est observé, n_1 est le nombre de graines germées, et

$$\text{logit}(p_i) = \alpha_0 + \alpha_1 x_{1i} + \alpha_2 x_{2i} + \alpha_{12} x_{1i} x_{2i} + \varepsilon_i, \qquad \varepsilon_i \sim N(0, \tau)$$

avec des prédicteurs x_{1i} et x_{2i} correspondant au type de graine. On considère les mêmes lois a priori que celles proposées sur la page (lois a priori impropre).

1. Mettre en œuvre un algorithme de type random walk Metropolis pour estimer les paramètres α_j à partir du modèle simplifié où $\tau = 0$ (pas d'effet aléatoire). Expliquer comment vous pouvez effectuer une régression logistique (classique, estimée par maximum de vraisemblance) pour choisir le point de départ de la chaîne simulée, et la matrice de covariance de la marche aléatoire.

2. On revient au modèle général. Développer (sur le papier) un algorithme de Gibbs permettant d'estimer le modèle: bien détailler la structure de l'algorithme et les loi que vous devez simuler. Quelles lois vous semblent difficiles à simuler directement?

3. Modifier l'algorithme de Gibbs en remplaçant les lois difficiles à simulées directement par des étapes de Metropolis (approche Metropolis-within-Gibbs). Comparer la performance de l'algorithme ainsi obtenu avec un random walk Metropolis pour même loi (donc la marche aléatoire opère sur toutes les composantes de la loi cible). Bien sûr, essayez de calibrer les deux algorithmes avant de les comparer.

4. Bonus: proposer une approche alternative pour simuler la même loi a posteriori (plusieurs solutions possibles; vous avez le droit d'utiliser des logiciels tels que STAN ou BlackJax).

MCMC sans biais

Le but de ce projet est de mettre en œuvre la méthode de "débiaisage" de MCMC proposé dans l'article suivant: https://arxiv.org/abs/1708.03625.

1. En guise de préliminaire, mettre en œuvre le Gibbs sampler proposé dans la section C de l'article (baseball batting averages), en reprenant les calculs des lois conditionelles à simuler (bien détailler). Les données:

0.395, 0.375, 0.355, 0.334, 0.313, 0.313, 0.291, 0.269, 0.247, 0.247, 0.224, 0.224, 0.224, 0.224, 0.224, 0.200, 0.175, 0.148

2. Expliquer le lien entre la technique du "maximal coupling" (Algorithme 2 de l'article ci-dessus) et l'algorithme d'acceptation - rejet vu en cours. Le mettre en œuvre pour chaque type de loi rencontré dans la question précédente (i.e. commment coupler deux lois Gamma, etc.).

3. Appliquer la méthode de débiaisage présentée dans l'article ci-dessus (en particulier l'estimateur de l'équation 2.1) au Gibbs sampler de la question 1. Faire des expériences numériques (et des graphiques!) pour évaluer l'influence des entiers m et k. Commenter.

4. Bonus: mettre en œuvre un algorithme de random-walk Metropolis pour la même loi a posteriori (expliquer comment vous pouvez choisir les paramètres de la marche aléatoire), puis appliquer la même technique de débiaisage, comparer les résultats, et commenter.

Données angulaires

Une loi classique pour des données correspondant à des angles observés sur l'intervalle $[-\pi, \pi]$ est la loi de von Mises, de densité:

$$\pi(\theta) = \frac{1}{Z(\kappa)} \exp\{\kappa \cos(\theta - \mu)\}$$

où $\kappa > 0$, $\mu \in [-\pi, \pi]$, et $Z(\kappa)$ est une constante de normalisation qui n'admet pas d'expression explicite.

Cette loi est souvent qualifiée de loi normale pour les angles, car $\cos(\theta) \approx 1 - \theta^2$.

1. Proposer un algorithme d'acceptation-rejet pour simuler ce type de loi, basé sur une loi de proposition uniforme (sur $[-\pi, \pi]$). Le mettre en oeuvre, et discuter comment sa performance dépend des paramètres μ et κ. (On pourra faire une représentation graphique par exemple.)

2. Proposer un algorithme de Metropolis simple pour simuler selon une telle loi; notamment proposer une règle simple pour que la performance de l'algorithme ne dépende pas des paramètres. (Expliquer ce que veut dire le mot "performance" pour un tel algorithme.)

3. Utiliser l'approche MCMC-MLE pour estimer les paramètres μ et κ à partir de données observées.

   Méthode MCMC-MLE: méthode qui revient à remplacer dans une fonction de log-vraisemblance une fonction des paramètres (ici Z) non calculable par une approximation basée sur de l'importance sampling, où la loi de proposition est la loi du modèle pour un paramètre fixe (bien choisi, à vous de réfléchir comment). On maximise ensuite la log-vraisemblance ainsi obtenue. Cette méthode est présentée notamment dans l'article de Geyer & Thompson (1992).

4. Bonus: proposer d'autres algorithmes de rejet plus efficaces pour simuler une loi de von Mises; quelques idées: prendre comme loi de proposition un mélange d'uniformes, ou une loi de Cauchy adaptée aux angles, comme expliqué p. 473 du livre de Devroye disponible à l'adresse suivante: http://luc.devroye.org/rnbookindex.html.

Volatilité stochastique

On considère un modèle simple de volatilité stochatique (populaire en économétrie de la finance); Y_t est le log-rendement d'un actif, de loi $N(0, \exp(X_t))$, X_t est une variable inobservée suivant la loi d'un processus auto-régressif:

$$X_t - \mu = \phi(X_{t-1} - \mu) + \varepsilon_t, \quad \varepsilon_t \sim N(0, \sigma^2)$$

de paramètre $\theta = (\mu, \phi, \sigma^2)$, avec $1 > |\phi|$. Pour X_0, on pourra ruser de la manière suivante: poser $X_{-1} = \mu$, et appliquer l'équation ci-dessus à $t = 0$.

1. Construire un Gibbs sampler pour simuler selon la loi de $(\theta, X_0, \ldots, X_T)$ sachant les données $Y_{0:T}$. Donner les lois à simuler. (Choisir des lois a priori relativement informatives, et donnes des lois conditionnelles simples à simuler.)

2. La loi de $X_t|\theta, X_{-t}, Y_{0:T}$ (où X_{-t} est l'ensemble des X_s pour $s \neq t$) n'est pas une loi connue. Proposer un algorithme de Metropolis pour simuler selon cette loi, et expliquer comment l'utiliser à l'intérieur du Gibbs sampler.

3. Mettre en oeuvre l'algorithme sur des données simulées selon le modèle. Par exemple, prendre $T = 100$, $\phi = 0.95$, $\sigma = 0.1$.

4. Bonus: utiliser l'algorithme sur données réelles (ex. SP500). Pour les gens très très courageux, vous pouvez aussi regarder l'article de Kim et al pour des algorithmes plus efficaces: http://finance.martinsewell.com/stylized-facts/volatility/KimShephardChib1998.pdf

Prévalence de Listeria dans le lait cru

La listeria est une bactérie pathogène qui provoque la listeriose. Le fichier listeria.txt (disponible dans l'espace du cours sous Teams) recense les données de 91 études, reportant dans différents pays le nombre d'échantillons testés (seconde colonne) et le nombre de cas positifs (présence détectée de listeria, première colonne).

1. On suppose pour commencer une probabilité constante de présence de listeria dans toutes les études; soit $r_i \sim Bin(n_i, p)$, avec comme loi a priori $p \sim \alpha, \beta$, $\alpha = \beta = 1$. Déterminer la loi a posteriori, la représenter. Est-ce que ce modèle vous semble raisonnable pour ces données? (On pourra répondre en faisant un graphique.)

2. On suppose ensuite que cette probabilité varie d'une étude à l'autre; soit $r_i \sim Bin(n_i, p_i)$ où les p_i sont distincts mais suivent la même loi a priori Beta(α, β). De plus, on suppose que α et β sont eux-aussi inconnus, et re-paramétrés ainsi: $\mu = \alpha/(\alpha + \beta)$, avec $\mu \sim$ U$[0, 1]$, et $\kappa = \alpha + \beta$, avec $\kappa \sim$ Exp(0.1). Construire et mettre en oeuvre un algorithme de Metropolis-within-Gibbs pour simuler la loi a posteriori de $(\mu, \kappa, p_1, \ldots, p_n)$ sachant les données. Bien expliquer les détails.

3. On modifie le modèle comme suit: on suppose désormais que les p_i suivent, a priori, un mélange de deux lois Beta. Adapter l'algorithme précédent à ce nouveau modèle.

4. Bonus: proposer une méthode pour comparer les trois modèles. Pour ce faire, on pourra utiliser la vraisemblance marginale (la densité des données après intégration sur les paramètres), et par exemple la méthode suivante: importance sampling avec comme loi de proposition la loi a priori. Bien expliquer les détails.

Option pricing et modèle d'Heston

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: $x^+ = \max(x, 0)$.

On considère le modèle d'Heston:

$$dS_t = rS_t dt + \sqrt{V_t} S_t (\rho dW_t^1) + \sqrt{1-\rho^2} dW_t^2 \quad (1)$$

$$dV_t = \kappa(\theta - V_t)dt + \xi\sqrt{V_t} dW_t^1 \quad (2)$$

avec $r = 0.03$, $\rho = 0.2$, $\kappa = 2$, $\theta = 0.3$, et $\xi = 0.5$ constantes fixes, et W_t^1, W_t^2$ des mouvements browniens indépendants.

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). Vous pouvez essayer aussi de faire varier les différents paramètres (ainsi que le pas de discrétisation) pour voir à quel point vos conclusions en dépendent.
2. Expliquer et illustrer comment la méthode MLMC (multi-level Monte Carlo) vue en cours, et présentée dans l'article suivant http://statweb.stanford.edu/~owen/courses/362/readings/GilesMultilevel.pdf peut réduire le temps de calcul.
3. Reprendre la comparaison en se basant sur du quasi-Monte Carlo.
4. Bonus: Vérifier à quel point les résultats dépendent de la façon utilisée pour construire les trajectoires des mouvements browniens (cf cours: random walk, Brownian bridge, etc.).

Modèle d'Ising avec bruit

Une méthode classique pour débruiter une image est de considérer un modèle de la forme: $y_i | x_i = k \sim N(\mu_k, \tau^2)$, où y_i est le niveau de gris du pixel i, et les x_i (à valeurs dans $\{0, 1\}$) suivent une loi dite d'Ising:

$$p(x) = \frac{1}{Z(\alpha, \beta)} \exp\{\alpha \sum_i x_i + \beta \sum_{i \sim j} \mathbf{1}\{x_i = x_j\}\}$$

où la deuxième somme porte sur les paires de voisins (deux pixels sont voisins s'ils ont un côté adjacent), et $Z(\alpha, \beta)$ est une constante de normalisation (difficile à évaluer, expliquer pourquoi).

1. On suppose τ, α et β connus. Mettre en oeuvre un Gibbs sampler pour simuler selon loi des x_i sachant les y_i. On pourra l'appliquer à une version bruitée d'une image de votre choix. Par exemple: https://images.fineartamerica.com/images/artworkimages/mediumlarge/3/op-art-black-and-white-infinity-whirl-tom-hill.jpg

2. On suppose τ inconnu. Adapter l'algorithme de la question précédente pour estimer τ. On pourra prendre une loi a priori de type inverse-gamma (expliquer pourquoi). Quels problèmes se posent si on choisit mal les hyper-paramètres de cette loi a priori?

3. On suppose α et β eux aussi inconnus. Est-il possible de construire de généraliser à nouveau l'algorithme des questions précédentes pour estimer α et β (en plus de τ et des variables x_i). (Attention, la réponse dépend de la loi a priori choisie). L'implementer si vous avez répondu par l'affirmative.

4. Bonus: proposer un algorithme de type ABC pour simuler la loi jointe de tous les paramètres et des x_i, sans contrainte sur la loi a priori de (α, β). Discuter et illustrer la performance de l'algorithme. En fonction du temps, vous pouvez aussi généraliser à des images avec K couleurs (loi a priori de type Potts).