Simulation parfaite des chaînes de Markov

Ana Bušić

INRIA - ENS

http://www.di.ens.fr/~busic/

ana.busic@inria.fr

Master AMIS - UVSQ

Versailles, novembre 2015

Rappels

Nous avons vu :

- Répresentation des chaînes de Markov par des SED.
- Simulation MCMC.
- Simulation parfaite par couplage depuis le passé.
- Cas monotone (et anti-monotone).

Rappels

Nous avons vu :

- Répresentation des chaînes de Markov par des SED.
- Simulation MCMC.
- Simulation parfaite par couplage depuis le passé.
- Cas monotone (et anti-monotone).

Quelques remarques/problèmes :

- Temps de couplage ?
- Garder la suite aléatoire en mémoire ?
- SED non-monotones ?

Temps de couplage

En général un problème très difficile.

Quelques résultats théoriques dans des files d'attente :

- Une file M/M/1/C. Pire cas : $\lambda = \mu$. $O(C^2)$.
- Réseaux de Jakson avec les capacités finies :

  Pour un réseau acyclique de K files M/M/1/C, on peut montrer que $\mathbb{E}\tau^b \leq \alpha(\lambda, \mu) K C^2$ [Dopper, Gaujal , Vincent, 2006], alors que la taille de l'espace d'état est $N = C^K$.

Read-once randomness (Wilson 1999)

Algorithme Propp & Wilson : on recommence les trajectoires en partant des temps $-N_1, -N_2, \ldots$ (avec $N_1 < N_2 < \ldots$) jusqu'à j tel que $|S_{N_j}| = 1$, où $S_{N_j} \overset{\text{def}}{=} \mathcal{X} \cdot a_{N_j \to 1}$.

Problème : garder la suite aléatoire en mémoire peut être très cher (dépends du temps de couplage !)

Read-once randomness (Wilson 1999)

Algorithme Propp & Wilson : on recommence les trajectoires en partant des temps $-N_1, -N_2, \ldots$ (avec $N_1 < N_2 < \ldots$) jusqu'à j tel que $|S_{N_j}| = 1$, où $S_{N_j} \stackrel{\text{def}}{=} \mathcal{X} \cdot a_{N_j \to 1}$.

Problème : garder la suite aléatoire en mémoire peut être très cher (dépends du temps de couplage !)

En pratique : garder les "seed" utilisées pour générer les sous-séquences - peut mener à toute sorte des comportements inattendus et génants (à éviter ! !)

Wilson's modification

[Wilson 1999] Idée : une sorte de couplage en avant mais où on n'arrete pas la simulation au moment même du couplage - on continue encore pendant un certain temps aléatoire.

Wilson's modification

[Wilson 1999] Idée : une sorte de couplage en avant mais où on n'arrete pas la simulation au moment même du couplage - on continue encore pendant un certain temps aléatoire.

Comment définir ce temps ?

Wilson's modification

[Wilson 1999] Idée : une sorte de couplage en avant mais où on n'arrete pas la simulation au moment même du couplage - on continue encore pendant un certain temps aléatoire.

Comment définir ce temps ?

On commence par faire quelques observations sur le couplage depuis le passé...

Observation I

Pour le cas monotone, on a utilisé les temps $(N_1, N_2, N_3, N_4...) = (1, 2, 4, 8, ...)$ mais toute suite strictement croissante marche aussi.

Observation I

Pour le cas monotone, on a utilisé les temps $(N_1, N_2, N_3, N_4 ...) = (1, 2, 4, 8, ...)$ mais toute suite strictement croissante marche aussi.

On peut même prendre une suite aléatoire strictement croissante !

Observation I

Pour le cas monotone, on a utilisé les temps $(N_1, N_2, N_3, N_4...) = (1, 2, 4, 8, ...)$ mais toute suite strictement croissante marche aussi.

On peut même prendre une suite aléatoire strictement croissante !

Preuve : soit $N_1 < N_2 < ...$ une suite aléatoire d'enties, indépendante de la suite aléatoire des événements tirés. Alors sachant $(N_1, N_2, ...)$ l'algorithme renvoie un échantillon non-biaisé de π. Cela est vrai pour toute réalisation de $(N_1, N_2, ...)$, ainsi l'algorithme (avec les temps $(N_1, N_2, ...)$ aléatoires) donne un échantillon non-biaisé de π.

Observation II

Continuer l'algorithme depuis encore plus loin dans le passé n'est pas génant !

(à part pour le temps supplémentaire d'exécution de l'algorithme)

Observation II

Continuer l'algorithme depuis encore plus loin dans le passé n'est pas génant !

(à part pour le temps supplémentaire d'exécution de l'algorithme)

Preuve : si $|S_n| = 1$ pour un $n \in \mathbb{N}$, alors pour tout $m > n$, on a $S_m = S_n$.

Choix de la suite aléatoire

Comment choisir la suite aléatoire strictement croissante $N_1 < N_2 < \ldots$?

Soit $(T_1, T_2, T_3, \ldots)$ une suite aléatoire d'entiers i.i.d. avec la distribution égale à celle du couplage en avant.

Choix de la suite aléatoire

Comment choisir la suite aléatoire strictement croissante $N_1 < N_2 < \ldots$?

Soit $(T_1, T_2, T_3, \ldots)$ une suite aléatoire d'entiers i.i.d. avec la distribution égale à celle du couplage en avant.

Question

Comment générer $(T_1, T_2, T_3, \ldots)$ *?*

Choix de la suite aléatoire

Comment choisir la suite aléatoire strictement croissante $N_1 < N_2 < \ldots$?

Soit $(T_1, T_2, T_3, \ldots)$ une suite aléatoire d'entiers i.i.d. avec la distribution égale à celle du couplage en avant.

Question

Comment générer $(T_1, T_2, T_3, \ldots)$ *?*

On fixe :

$$\begin{aligned} N_1 &= T_1 \\ N_2 &= T_1 + T_2 \\ N_3 &= T_1 + T_2 + T_3 \\ &\vdots \quad\quad \vdots \\ N_n &= N_{n-1} + T_n \end{aligned}$$

Observation III

La probabilité que l'algorithme Propp & Wilson en partant du temps initial $-N_1 = -T_1$ se termine (couplage avant le temps 0) est au moins $1/2$.

Observation III

La probabilité que l'algorithme Propp & Wilson en partant du temps initial $-N_1 = -T_1$ se termine (couplage avant le temps 0) est au moins $1/2$.

Preuve : Soit M_1 le temps nécessaire pour coupler en partant du temps $-N_1$ (et dépassant le temps 0 si besoin !)
Alors M_1 et N_1 ont la même distribution et ils sont indépendants, donc :

$$\mathbb{P}(M_1 \leq N_1) = \mathbb{P}(M_1 \geq N_1).$$

Observation III

La probabilité que l'algorithme Propp & Wilson en partant du temps initial $-N_1 = -T_1$ se termine (couplage avant le temps 0) est au moins $1/2$.

Preuve : Soit M_1 le temps nécessaire pour coupler en partant du temps $-N_1$ (et dépassant le temps 0 si besoin !)
Alors M_1 et N_1 ont la même distribution et ils sont indépendants, donc :

$$\mathbb{P}(M_1 \leq N_1) = \mathbb{P}(M_1 \geq N_1).$$

Aussi :

$$\begin{aligned}
\mathbb{P}(M_1 \leq N_1) + \mathbb{P}(M_1 \geq N_1) &= 1 - \mathbb{P}(M_1 > N_1) + 1 - \mathbb{P}(M_1 < N_1) \\
&= 2 - (\mathbb{P}(M_1 > N_1) + \mathbb{P}(M_1 < N_1)) \\
&= 2 - (\mathbb{P}(M_1 \neq N_1)) \\
&\geq 2 - 1 = 1.
\end{aligned}$$

Donc $\mathbb{P}(M_1 \leq N_1) \geq 1/2$.

Observation III - suite

On peut continuer le raisonement...

Si l'algorithme ne termine pas en partant de $-N_1$, alors avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_1$ en partant du temps $-N_2 = -(N_1 + T_2)$. Plus généralement, avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_{j-1}$ en partant du temps $-N_j = -(N_{j-1} + T_j)$.

Observation III - suite

On peut continuer le raisonement...

Si l'algorithme ne termine pas en partant de $-N_1$, alors avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_1$ en partant du temps $-N_2 = -(N_1 + T_2)$. Plus généralement, avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_{j-1}$ en partant du temps $-N_j = -(N_{j-1} + T_j)$.

On va dire qu'on a un succès à l'étape j si, en partant du temps $-N_j$, on observe le couplage avant le temps $-N_{j-1}$.

Observation III - suite

On peut continuer le raisonement...

Si l'algorithme ne termine pas en partant de $-N_1$, alors avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_1$ en partant du temps $-N_2 = -(N_1 + T_2)$. Plus généralement, avec la probabilité au moins $1/2$ on observe le couplage avant le temps $-N_{j-1}$ en partant du temps $-N_j = -(N_{j-1} + T_j)$.

On va dire qu'on a un succès à l'étape j si, en partant du temps $-N_j$, on observe le couplage avant le temps $-N_{j-1}$.

Et pour avoir la probabilité de succès égale à $1/2$, au cas ou on observe le couplage exactement au temps $-N_{j-1}$, on tire une piece non-biaisé et avec proba $1/2$ on déclare le succès, et sinon l'échec.

Modification de Wilson

L'algorithme (juste, mais un peu artificiel !) qui consiste à recommencer depuis les temps $-N_j$ jusqu'au succès donne bien un échantillon non-biaisé de π.

Modification de Wilson

L'algorithme (juste, mais un peu artificiel !) qui consiste à recommencer depuis les temps $-N_j$ jusqu'au succès donne bien un échantillon non-biaisé de π.

Notons le nombre (aléatoire) d'échec avant le succès par Y. Alors Y est une variable aléatoire géométrique de paramètre $p = 1/2$:

$$\mathbb{P}(Y = n) = p(1-p)^n.$$

L'itération finale (et succès) commence en $-N_{Y+1}$.

Modification de Wilson

L'algorithme (juste, mais un peu artificiel !) qui consiste à recommencer depuis les temps $-N_j$ jusqu'au succès donne bien un échantillon non-biaisé de π.

Notons le nombre (aléatoire) d'échec avant le succès par Y. Alors Y est une variable aléatoire géométrique de paramètre $p = 1/2$:

$$\mathbb{P}(Y = n) = p(1-p)^n.$$

L'itération finale (et succès) commence en $-N_{Y+1}$.

Idée de modification de Wilson : trouver un moyen de simuler d'abord les trajectoires depuis $-N_{Y+1}$ jusqu'à $-N_Y$, puis de $-N_Y$ jusqu'à $-N_{Y-1}$... et cela jusqu'au temps 0.

Twin run

Executer deux copies indépendantes de couplage en avant - On s'arrête quand les deux couplent !

On déclare la copie qui a couplé d'abord le gagnant et l'autre le perdant (avec une pièce non-biaisée pour départager si égalité).

Twin run

Executer deux copies indépendantes de couplage en avant - On s'arrête quand les deux couplent !

On déclare la copie qui a couplé d'abord le gagnant et l'autre le perdant (avec une pièce non-biaisée pour départager si égalité).

Observation : l'évolution de nos trajectoires depuis le passé en partant de $-N_{Y+1}$ jusqu'à $-N_Y$ a justement la même distribution que celle du gagnant.

Algorithme

- Choisir Y selon la loi géometrique avec $p = 1/2$.
- Si $Y = 0$ on a fini. On renvoie la valeur finale du gagnant au moment de couplage du perdant.
- Sinon on doit continuer la simulation depuis $-N_Y$ jusqu'à 0. Pour simuler les trajectoires de $-N_Y$ jusqu'à $-N_{Y-1}$, on fait un autre twin run et en prend l'évolution du perdant, ou le perdant évolue de 0 jusqu'au couplage du gagnant !

  Puis on continue avec un autre twin run indépendant pour $-N_{Y-1}$ jusqu'à $-N_{Y-2}$...

  On renvoie la valeur finale du perdant du dernier twin run (au moment du couplage du gagnant du même twin run).