MÉTHODES DE MONTE-CARLO
CORRIGÉ PARTIEL DES EXERCICES DU 27 FÉVRIER 2012

par Rémi Peyre

EXERCICE 6 — Yahtzee avec double réduction de variance

Question. Reprendre le code du yahtzee *avec échantillonnage préférentiel* pour lui appliquer *en plus* la méthode de conditionnement.

Corrigé. Avant d'implémenter le code, la première étape est de déterminer l'espérance conditionnelle, à l'issue des deux premiers lancers, de la variable à intégrer. Rappelons que dans la méthode d'échantillonnage préférentiel que nous avons prise, la quantité que nous évaluons est en fait la probabilité (non biaisée) d'obtenir un yahtzee de '6', que nous multiplions par 6 ensuite (étant entendu que notre joueur traite tous les chiffres de manière identique), et que l'échantillonnage préférentiel consiste à piper les dés pour qu'ils tombent plus souvent sur '6' que sur un autre chiffre.

A priori on se dit que la situation va être compliquée, car la fonction que nous échantillonnons n'est pas l'indicatrice de « obtenir un yahtzee de '6' », mais cette indicatrice *divisée par la densité de probabilité de la loi biaisée par rapport à la loi non biaisée*. Mais en fait, on n'est pas obligés d'en tenir compte, car il est équivalent d'appliquer le conditionnement *avant* l'échantillonnage préférentiel que de l'appliquer après ! [∗] [†] Conditionnellement aux deux premiers lancers la probabilité d'obtenir un yahtzee de '6' est :

[∗]. Quel que soit le moment auquel on applique le conditionnement, on tombe sur la même fonctionnelle à intégrer ; toutefois les calculs sont plus compliqués quand on applique le conditionnement en dernier. Voici en effet ce qui se passe dans ce cas-là. La quantité dont on cherche à évaluer l'espérance sous la loi biaisée (avant conditionnement) est $f_{\text{pref}} := \mathbf{1}_{\text{y. de 6}} \times (1+5\lambda)^{-D_6} \times (1-\lambda)^{-D_0}$, où D_6 et D_0 sont le nombre total de '6', resp. de 'non-6', tirés au cours des trois lancers. Nous voulons évaluer l'espérance de cette quantité à l'issue des deux premiers lancers, *toujours sous la loi pipée*. Conditionnellement aux deux premiers lancers, l'espérance de f_{pref}, pour commencer, vaut clairement 0 si le chiffre qu'on garde à l'issue du second lancer n'est pas '6'. D'autre part, si le chiffre qu'on garde à l'issue du second lancer est '6', il faut pour obtenir un yahtzee de '6' que les $(K-5)$ dés qu'on relancera au troisième lancer tombent tous sur '6', ce qui arrive avec probabilité $[(1+5\lambda)/6]^{5-K}$; dans ce cas, on aura $D_0 = D_0'$ et $D_6 = D_6' + (5-K)$, d'où $f_{\text{pref}} = (1+5\lambda)^{-D_6'-5+K}(1-\lambda)^{-D_0}$. Au final, notre espérance conditionnelle est

$$[(1+5\lambda)/6]^{5-K} \times (1+5\lambda)^{-D_6'-5+K}(1-\lambda)^{-D_0'}, \tag{1}$$

ce qui est bien la même chose que l'expression (2) obtenue en appliquant le conditionnement en premier.

[†]. Montrons que, plus généralement, cela donne toujours le même résultat d'appliquer le conditionnemement en premier et en dernier. Supposons que notre simulation se fasse à l'aide du tirage de deux variables aléatoires X et Y (la loi conditionnelle de Y étant susceptible de dépendre de la loi de X), et qu'on cherche à conditionner par rapport au tirage de X. On suppose la loi biaisée $\mathbb{P}'$ a une densité $m(x)n(x,y)$ par rapport à la loi $\mathbb{P}$, où $m(x)$ est la densité relative de x (on a donc $\mathbb{E}(m) = 1$) et $n(x,y)$ est, *conditionnellement à l'événement « $X = x$ »*, la densité relative de Y sous $\mathbb{P}'$ par rapport à $\mathbb{P}$ (j'entends par là que c'est la densité de $\mathbb{P}'(\cdot|X=x)$ par rapport à $\mathbb{P}(\cdot|X=x)$). La quantité que nous cherchons à évaluer, quant à elle, est $f(x,y)$. Appliquer le conditionnement seul signifie qu'on cherche à évaluer l'espérance sous $\mathbb{P}$ de $\bar{f}(x) := \mathbb{E}[f(x,y) \mid X = x]$; si on fait l'échantillonnage préférentiel après, cela signifie qu'on évalue $\mathbb{E}'[\bar{f}(x) \,/\, m(x)]$. Si on fait l'échantillonnage préférentiel seul, on évalue $\mathbb{E}'[f(x,y) \,/\, (m(x)n(x,y))]$. Faire le conditionnement après, c'est donc évaluer l'espérance sous $\mathbb{P}'$ de $\mathbb{E}'[f(x,y) \,/\, (m(x)n(x,y)) \mid X = x] = \mathbb{E}[n(x,y) \times f(x,y) \,/\, (m(x)n(x,y)) \mid X = x] = \mathbb{E}[f(x,y) \,/\, m(x) \mid X = x] = \bar{f}(x) \,/\, m(x)$: on tombe bien sur la même quantité à intégrer.

– 0 si le chiffre que le joueur décide de garder à l'issue du second lancer est différent de '6' ; [‡]
– 6^{-5+K} si le chiffre qu'il garde est '6', où K est le nombre de '6' qu'il a obtenus à l'issue du second lancer.

Reste encore à tenir compte de la densité relative de probabilité due au pipage... C'est la même que dans le cas sans conditionnement, sauf qu'on l'évalue à l'issue de deux lancers. Vu que, pour un paramètre de pipage λ, la probabilité d'obtenir un '6' à chaque lancer est $(1+5\lambda)/6$ et que la probabilité d'obtenir chacun des autres chiffres est $(1-\lambda)/6$. Par conséquent, notant D'_6 le nombre de fois qu'un dé est tombé sur 6 à l'issue des deux premiers lancers et D'_0 le nombre de fois qu'on a obtenu un résultat autre que 6, le densité de probabilité relative est $(1+5\lambda)^{D_6} \times (1-\lambda))^{D_0}$. Au final, la quantité dont on va évaluer l'espérance sous la loi pipée à l'issue du deuxième lancer est

$$\mathbf{1}_{\text{on veut garder les '6'}} \times 6^{-(5+K)} \times (1+5\lambda)^{D_6}(1-\lambda)^{D_0}. \tag{2}$$

Il n'y a plus qu'à coder cela, en s'inspirant des deux codes "`yahtzeepref`" et "`yahtzeecondi`". Voici mon code : [§]

```
function yahtzeecondipref(lambda,N)
% La fonction "yahtzeecondipref" applique à la fois les techniques de
% conditionnement et d'échantillonnage préférentiel au problème du yahtzee,
% selon les calculs théoriques que nous venons d'expliquer. Les
% commentaires expliquent les différences par rapport à "yahtzeepref".

somme = 0;
sommecarres = 0;
tic;
for i = 1:N,
    six = 0;
    nonsix = 0;
    deuxlancers;
    % Par rapport à "yahtzeepref", il n'y a ici que deux lancers à faire.
    densite = (1+5*lambda)^six*(1-lambda)^nonsix;
    x = resultat/densite;
    somme = somme + x;
    sommecarres = sommecarres+x*x;
end
t = toc;
moyenne = 6*somme/N;
variance = 6*6*sommecarres/N-moyenne*moyenne;
disp('Intervalle de confiance à 2 sigmas :');
disp(moyenne+sqrt(variance/N)*[-1,1]);
disp('Efficacité :');
disp(N/t/variance);


    function deuxlancers

        des = [pipe,pipe,pipe,pipe,pipe];
```

[‡]. Rappelons que le joueur ne sait pas que les dés sont pipés en faveur de '6' et qu'il cherche juste à obtenir à obtenir un yahtzee, peu importe son chiffre. Du coup, si par hasard il a obtenu plus de '5' (par exemple) que de '6' à l'issue du second lancer, il gardera les '5' et relancera les '6' !

[§]. À l'utilisation de cet algorithme, je préconise de prendre $\lambda = 0{,}35$.

```
    des = relancer(des);
    % Après avoir relancé les dés une première fois, nous n'avons pas
    % besoin d'effectuer le troisième lancer, mais il faut quand même
    % identifier le chiffre des dés que nous choisissons de garder, et
    % dire combien de fois il apparaît (càd. déterminer K).
    occurrences = rand(1,6);
    for j = 1:5,
        occurrences(des(j)) = occurrences(des(j))+1;
    end
    % Comme nous avons besoin de connaître K, il nous faut récupérer
    % /les deux/ sorties de "max(occurrences)".
    [record, recordman] = max(occurrences);
    desrelances = des;
    for j = 1:5,
        if des(j) ~= recordman
            desrelances(j) = pipe;
        end
    end
    if (recordman ~= 6)
        resultat = 0;
    else
        % Attention, il y a un piège ! "record" n'est /pas/ la valeur
        % de K, à cause du fait que nous lui avons rajouté un nombre
        % aléatoire pour départager uniformément les ex-æquo. K est en
        % fait la /partie entière/ de "record".
        resultat = 6^(-5+floor(record));
    end
end


function desrelances = relancer(des)

    occurrences = rand(1,6);
    for j = 1:5,
        occurrences(des(j)) = occurrences(des(j))+1;
    end
    [~, recordman] = max(occurrences);
    desrelances = des;
    for j = 1:5,
        if des(j) ~= recordman
            desrelances(j) = pipe;
        end
    end
end


function chiffre = pipe

    if(rand<lambda)
        chiffre = 6;
    else
        chiffre = randi([1,6]);
    end
    if (chiffre == 6)
        six = six + 1;
```

```
        else
            nonsix = nonsix + 1;
        end
    end

end
```

✓