Echantillonnage multidimensionnel (= de plusieurs échantillons à la fois) à entropie maximum : définition, propriétés, algorithmes et programmes.

Jean-Claude DEVILLE (*) et Lionel QUALITÉ (**)

(*) CREST/ENSAI, (**) CREST/ENSAI puis Université de Neuchâtel

On désire tirer dans une même population U de taille N, Q échantillons disjoints en même temps (et non pas successivement !) ayant des probabilités d'inclusion π_k^i ($i=1$ à Q, $k=1$ à N) fixées et de tailles fixées n_i . Même dans le cas $Q=2$, les choses sont loin d'être évidentes si on désire conserver une certaine symétrie au problème.

Loin d'être un problème académique, on a besoin de techniques de ce genre pour tirer certains échantillons pour le contrôle de qualité (sans jeu de mot) du recensement. En 1999 (et oui, c'est déjà vieux tout ça) chaque unité de traitement devait être contrôlés dans deux ateliers, l'un s'occupant de l'analyse ménage-famille, l'autre de la qualité des libellés pour la codification automatique. Pour des raisons de commodité, il s'agissait de tirer des districts entiers, les premiers avec des probabilités proportionnelles au nombre de gros ménages, le second avec des probabilités proportionnelles au nombres d'actifs. Bien entendu, une solution honorable avait, en son temps, été donnée et appliquée. Une autre fut trouvée peu après, ce qui permit de rédiger un joli article académique [3] que personne n'a jamais lu, pas même sans doute, les 'referee' chargés d'évaluer la valeur du produit. Ce qui suit décrit sans doute la meilleure solution du problème. Mais personne n'est obligé de l'appliquer.

D'autant plus qu'on obtient en même temps une solution intéressante à la question de l'imputation d'une variable quantitative en cas de non-réponse.

1. Echantillonnage à entropie maximum

On s'intéresse ici au cas d'un seul échantillon tiré avec des probabilités d'inclusion fixées $0<\pi_k<1$ et un support fixé $\boldsymbol{S}$. De plus, on veut que les variables indicatrices I_k ($=1$ si k dans s, 0 sinon) soient les plus indépendantes possibles, ou, ce qui revient sensiblement au même, que les $p(s)$ aient une dispersion minimale. Sans faire trop de philosophie, un bon critère est de maximiser l'entropie du plan de sondage $\sum_{s\in\mathbf{S}} - p(s)\log(p(s))$ avec $p(0)\log p(0) = 0$.

Les N contraintes sont $\pi_k = \sum_{\mathbf{S}} \mathbf{1}(k \in s)\, p(s)$ associées à des multiplicateurs de Lagrange λ_k .

La solution du problème d'optimisation donne $p(s)=C(\mathbf{S})\exp(\lambda.s)$ où apparaît le produit scalaire du N-vecteur des λ_k avec le vecteur s des 'coordonnées de s ': 1 si k est dans s, 0 sinon. La constante $C(\mathbf{S})$ normalise la somme des probabilités à un.

-Supposons que $\mathbf{S}$ soit l'ensemble de toutes les parties de U. Posons $\exp\lambda_k=\omega_k=\pi_k^*/(1-\pi_k^*)$.
On voit que $p(s)=C(\mathbf{S})\prod_s \omega_k=\prod_s \pi_k^* \prod_{U-s}(1-\pi_k^*)$ ce qui signifie que nous avons obtenu un échantillonnage de Poisson et que $\pi_k^*=\pi_k$.

-Si $\mathbf{S}$ est une famille arbitraire de parties de U , on a donc un échantillonnage de Poisson conditionnel à $\mathbf{S}$, et les π_k sont les probabilités conditionnelles, 'sachant ' que s est dans $\mathbf{S}$. Les π_k^* sont maintenant différents des π_k .

2. Echantillonnage de Poisson conditionnel à une taille fixe

Occupons nous de cas où $\mathbf{S}$ est l'ensemble des échantillons vérifiant une contrainte de taille fixe $card(s)=n$. La somme des π_k doit donc être égale à n . On sait [6] qu'il existe une infinité de façon de traiter ce problème. Pour de nombreuses raisons, la technique 'à entropie maximum' est vraisemblablement la meilleure [4] et presque la plus simple , si on excepte les variantes du tirage systématique!

Il se trouve que les π_k^* ne sont plus déterminés de façon unique mais dépendent d'un terme additif sur les λ_k , ou, ce qui revient au même, d'un facteur positif sur les ω_k .Mais le calcul des probabilités conditionnelles est étonnamment simple! Notons π_k^n ces probabilités d'inclusion conditionnelles .

On a: $For\ n=0 \quad \pi_k^0=0$ $For\ n=1 \quad \pi_k^1=\omega_k / \sum_U \omega_l$

$For\ n=n+1 \quad \pi_k^{n+1}=\frac{C(n)}{C(n+1)}\omega_k\,(1-\pi_k^n)$ et le facteur $C(n)/C(n+1)$ est identifié par la condition: $\sum_U \pi_k^{n+1}=n+1$.

Pour l'exemple, voici un code matlab de ce calcul :

```
%Calcul les probas d'ordre 1 jusqu'a l, ;retourne le  vecteur z (Nx1) ou la matrice de 0 à l  %
function [z,p]=p0cond(la,l);
 [N,p]=size(la);    if l>N display('err dimension'); else
   %     Constantes  %                lla=la./(1-la);    l0=[2:l+1];    un=ones(N,1);
   %    initialisation  %    p=zeros(N,l+1);
  %     Récurrence %     for i=2:l+1;          ii=i-1;   py=p(:,ii);  v=lla.*(un-py);     p(:,i)=ii*v/sum(v);        end;
 z=p(:,l+1);  end;
```

Ce programme est très rapide est très précis, quasiment comme celui qui calcule une exponentielle.

Pour les probabilités du second ordre, c'est à peine plus compliqué. La différence vient de ce que la récurrence fonctionne de n à $n+2$:

$$\pi_{kl}^{n+2}=\frac{C(n)}{C(n+2)}\omega_k\omega_l(1-\pi_k^n-\pi_l^n+\pi_{kl}^n)$$

On part des évidences $\pi_{kl}^0=\pi_{kl}^1=0$, et on identifie le ratio des constantes en utilisant le fait que la somme des proba doit être égale à $(n+2)(n+1)/2$.

Le code matlab va de soit :

```
function pp0=p2cond(m0,n);
[N,NN]=size(m0);  rat=m0./(1-m0);rat2=rat*rat';rat2=rat2-diag(diag(rat2));   ss=sum(rat);
                  if rem(n,2)==0 pp0=zeros(N,N);s=0;else pp0=diag(rat)/sum(rat); s=1;end;
p0=diag(pp0);
while s<n
s=s+2;
a=repmat(p0,1,N);     pp1=rat2.*(pp0-a-a'+1);pp1=pp1*s*(s-1)/sum(sum(pp1));  p0=sum(pp1)'/(s-1);
pp0=pp1+diag(p0);
end;
```

Il reste à calculer les π_k^* à partir des π_k , c'est à dire à inverser la fonction construite ci-dessus. On va normaliser la somme des π_k^* à n . Comme, dans ce cas, la matrice des dérivées partielles est très proche de l'identité, la méthode de Newton se simplifie et on obtient le simple programme:

```
%Cette fonction calcule les probabilités d'inclusion du sondage poissonnien q dont p est le vecteur de proba
    d'inclusions de taille fixe associé
    function q=poiss(p)
q0=p;n=round(sum(p));test=1;niter=1;
while (test>0.00000000001)&(niter<100)
  p1=p0cond(q0,n);
niter=niter+1;test=max(abs(p-p1));
q0=q0+p-p1;
end;
q=q0;niter
```

De nouveau, cette fonction est rapide et précise. Elle peut être utilisée sur des fichiers de plusieurs milliers d'unités et tourne en moins de dix secondes sur un ordinateur 'normal'.

Algorithmes de tirage :

a) Unité par unité(schéma de l'urne)

Step 0: Compute the ω and the π^* from the π using the preceding algorithm.
Draw 1: one unit k_1 with proba π^{*n}/n in U.
Draw 2: one unit k_2 with proba $\pi^{*n-1}/(n-1)$ in U-k_1 .
…..…
Draw i: one unit k_i with proba π^{*n-i+1}/i in U-{ k_1 ,…,k_{i-1}} .
…..…..
Draw n: one unit k_n with proba π^{*1} in U-{ k_1 ,…,k_{n-1}} .

Justification: Dempster et alli [1], mais faites le plutôt vous-même, ça ira plus vite!

b) Algorithme séquentiel :

C'est l'analogue d'un algorithme bien connu pour le sondage simple :

Unit 1: is included in the sample with proba π_1 ;
Compute the $\pi^{U-1,n-1}$ or $\pi^{U-1,n}$ according to the output of the drawing.
…
Unit k+1 : Let n_k be the number of units selected in the sample until k 'step' (=units). We have computed the proba $\pi^{U-\{1,..k\},\, n-n_k}$. The unit is included with proba $\pi_{k+1}{}^{U-\{1,..k\},\, n-n_k}$.

Ces deux algorithmes sont très efficaces. Le second est plutôt meilleur, comme Chen l'a affirmé [2] sans donner d'arguments très visibles.

3. Echantillonnage 'multidimensionnel'

On s'intéresse maintenant à un échantillon Q-dimensionnel (à Q échantillons pour causer moins pompier) dont on désire contrôler les tailles et les tailles des intersections. Pour deux échantillons, par exemple, on désire avoir des probabilités d'inclusion fixées dans chaque échantillon π^1_k et π^2_k , des tailles fixées $n_{1 ou 2}$, et, de plus, contrôler la taille n_3 de l'intersection. Ceci implique, naturellement, des contraintes sur les probabilités d'inclusion de l 'échantillon *3*=intersection, que nous allons discuter un peu plus tard. Supposons que nous ayons réussi à nous débrouiller de ce problème, on voit qu'on se ramène à tirer trois échantillons <u>disjoints</u>, de taille n_1-n_3, n_2-n_3 et n_3 avec des probabilités d'inclusion que nous savons calculer (patience, ça va venir).

Disjoints ou pas, un échantillonnage Q-dimensionnel est une loi de probabilité sur l'ensemble des suites ordonnées de Q parties de U , vérifiant donc , $p(s_1,...,s_Q) \geq 0$ et $\sum_{s1,...,s_Q} p(s_1,...,s_Q) = 1$. On définit sans aucune des difficultés techniques qui font le charme pervers de la Théorie des Probabilités, des lois marginales comme $p_1(s_1) = \sum_{s2...,s_Q} p(s_1,...,s_Q)$ à une où plusieurs dimensions et des lois conditionnelles comme $p_{1|2}(s_1|s_2) = p_{12}(s_1,s_2)/p_2(s_2)$.

Comme devant, on peut essayer de trouver la forme d'un échantillonnage à entropie maximum, les vecteurs de probabilités d'inclusion étant donnés, soit π^i pour $i= 1$ à Q et le support $\boldsymbol{S}$ de l'échantillonnage étant lui aussi fixé**.** On est donc conduit au problème suivant :

Maximiser $\quad -\sum_{(s1,..,sQ)\in \mathbf{S}} p(s_1,...,s_Q)\, log(p(s_1,...,s_Q))$

sous les QxN contraintes : $\sum_{si \supset k} p_i(s_i) = \pi^i_k = \sum \mathbf{1}(k \in s_i)\ p_i(s_i)$.

Le problème étant convexe, sa solution est trouvée sans ennui par la technique de Monsieur Lagrange (qu'on remercie et qu'on applaudit bien fort !). Au prix de l'introduction de Q N-vecteurs λ^i , on trouve que $p(s_1,...,s_Q) = C(\mathbf{S})\ \exp(\sum_{i=1}^{Q} \lambda^i.s_i)$ où $C(\boldsymbol{S})$ est une constante de normalisation, et où les vecteurs lambda s'identifient à partir des contraintes. C'est comme au paragraphe 1, juste un petit peu plus compliqué. Cependant les conclusions générales reste les mêmes :

-Supposons que $\boldsymbol{S}$ soit l'ensemble de toutes les suites de Q parties de U. Posons $exp\lambda^i_k = \omega^i_k = \pi^i_k{}^*(1-\pi^i_k{}^*)$.

On voit que $p(s_1,...,s_Q) = C(\mathbf{S}) \prod_{i=1}^{Q} (\prod_{si} \omega^i_k) = \prod_{i=1}^{Q} (\prod_{si} \pi^i_k{}^* \prod_{U-si} (1-\pi^i_k{}^*))$ ce qui signifie que nous avons obtenu Q échantillonnages de Poisson <u>*indépendants*</u> et que $\pi^i_k{}^* = \pi^i_k$.

On a donc en particulier $p(s_1,...,s_Q) = \prod_{i=1}^{Q} (p_i(s_i))$. Par suite, les plans 'intersection' du genre $p(s_1 \cap s_2)$ sont aussi poissonnien pour les probabilités d'inclusion $\pi_k{}^1 \pi_k{}^2$.

- Si $\boldsymbol{S}$ est une famille arbitraire de suites de Q parties de U , on a donc une famille d'échantillonnages de Poisson indépendants conditionnellement à $\boldsymbol{S}$ **.** Les π^i_k sont les probabilités conditionnelles, 'sachant ' que ($s_1,...,s_Q$) est dans $\boldsymbol{S}$. Les $\pi^i_k{}^*$ sont maintenant différents des π^i_k .

4. Echantillonnage de Poisson multiple disjoint à tailles fixes

Occupons nous maintenant du cas où $\boldsymbol{S}$ est contraint par des tailles fixes $card(s_i)=n_i$ et par le fait que les échantillons sont disjoints. La somme en k des π^i_k doit donc être égale à n_i . Les $\pi^i_k{}^*$ ne sont plus déterminés de façon unique mais dépendent de termes additifs dépendant de i sur les λ^i_k , ou, ce qui revient au même, d'un facteur positif sur les ω^i_k . Il se trouve que le calcul des probabilités conditionnelles reste étonnamment simple, du moins sur le plan des principes! Notons $\pi_k^{i;\underline{n}}$ ces probabilités d'inclusion conditionnelles, avec $\underline{n}=(n_1,\ldots,n_Q)$ et $\underline{i}=(0,\ldots,1,..,0)$ le vecteur avec *1* à la $i^{ème}$ position et des zéros ailleurs. A partir de la relation de définition des probabilités d'inclusion, on obtient :

$$\pi_k^{i;\bar{n}}=C_{\bar{n}}\sum_{Dik}\exp\left(\sum_{j=1}^{Q}\lambda^j.s_j\right)=C_{\bar{n}}\omega_k^i\sum_{Eik}\exp\left(\sum_{j=1}^{Q}\lambda^j.s_j\right)=(C_{\bar{n}}/C_{\bar{n}-\bar{i}})\omega_k^i(1-\sum_j\pi_k^{j;\bar{n}-\bar{i}})\ ,$$

où D_{ik} est l'ensemble des $\underline{n}$-échantillons tels que s_i contienne k , et E_{ik} l'ensemble des $\underline{n}$-$\underline{i}$-échantillons dont aucun ne contient k (on notera que l'éditeur d'équations de Word m'a +ou- obligé à mettre les barres au dessus des n et des j) . La constante multiplicative s'identifie en ajustant la somme des $\pi_k^{i;\underline{n}}$ à n_i . De ce fait, on obtient un programme de calcul relativement simple, le plus délicat étant l'initialisation de la récurrence 'à double coque' et la gestion des multi-indices. Pour cela, on utilise les deux fonctions utilitaires spécifiques 'coord' et 'reduc' listée ci-dessous. On remarquera qu'elles utilisent toutes les deux les merveilleuses fonctions de matlab 'reshape' et 'permute' (ceci était un message publicitaire désintéressé).

```
%Calcul les probas d'ordre 1 de d' ech disjoints jusqu'a n (vecteur d'entiers);
%retourne les d vecteurs (z matrice Nxd ou les n matrices de (0,0,0) à (n)

function [z,p]=pdcond(la,n);
   [N,d]=size(la);[y,dd]=size(n);
   if d~=dd, error('les arguments sont incompatibles'); end
   if d==1 la=[la,la]; n=[n,0];d=2;id=1;else id=0;end
   py=zeros(N,d); qy=zeros(N,d);ir=zeros(1,d);unv=ones(N,1);
if sum(n)>N error('échantillons trop gros'); end


      %      Constantes
   lla=la./(1-la);     n1=n+1;   n2=n+2;
   cum=cumprod(n2);   ss=[1,cum(1:d-1)];

                  %initialisation
   p=zeros(N,d,prod(n2)); p=reshape(p,[N,d,n2]);

                   %Récurrence
 list=1:prod(n2);  cord=coord(n2);
 list=reshape(list,n2);   list=reduc(list);

 for j=1:prod(n1);   i=list(j);ir(:)=[i-ss(:)];
          for k=1:d
            qy=p(:,:,ir(k));py(:,k)=sum(qy')';
          end;
     nrm=cord(i,:)-2;      mnrm=unv*nrm;
     py=lla.*(1-py);      div=unv*sum(py);
     p(:,:,i)=mnrm.*py./div;
   end;
     z=p(:,:,prod(n2));
     if id==1 z=z(:,1);p=p(:,1,:);end
```

Remarques: Cette récurrence calcule, pour chaque 'boite' $\underline{n}$, Q vecteurs de probabilités d'inclusion π_k^i en fonction de ceux des Q boites $\underline{n}$-$\underline{i}$. La récurrence doit donc gérer le 'complexe symplicial' (désolé, ça a l'air de se dire comme ça!) de type 'coin', 'arête', 'mur','salle'. L'astuce et d'initialiser

par un indice virtuel égal à *-1*, qui, dès qu'il est présent dans un multi-indices $\underline{n}$ rend les $\pi_k^{i,\underline{n}}$ nuls pour tout k et tout i . C'est ce qu'on peut appeler la deuxième coque de la récurrence. Ca complique la gestion des indices (matlab n'aime pas beaucoup les indices égaux à zéros, mais alors moins un, la, ça hurle!), d'où les deux utilitaires.

```
%n vecteur ligne d'entiers;
%coo est la liste lexcographique des multi-entiers
%inférieurs ou egaux à n.

function  coo=coord(n)
d=length(n);N=prod(n);cum=cumprod(n);
prodap=N./cum;prodav=cum./n;
coo=[];
for k=1:d
   mat=repmat([1:n(k)],prodav(k),prodap(k));
         v=reshape(mat,N,1);coo=[coo,v];
   end


%Cette fonction permet de gérer les décalages d'indices dus à
 l'utilisation des –1 %

function a=reduc(b)
n=size(b);d=ndims(b);permut=[2:d,1];
for k=1:d
      b(1,:)=[]; n(1)=n(1)-1; b=permute(b,permut);
  n=[n(2:d),n(1)];  b=reshape(b,n);
end
a=b;
```

Il reste à calculer les π_k^{i*} du tirage multipoissonnien sous jacent, c'est à dire les $\lambda_k^{\,i}$ des équations, à partir des π_k^i , c'est à dire à inverser la fonction construite ci-dessus. On va normaliser la somme des π_k^{i*} à n_i . Comme dans le cas d'un seul échantillon de taille fixe, la matrice des derivées partielles est très proche de l'identité, et on peut utiliser la méthode de Newton qui se simplifie ; de fait, on peut utiliser le programme 'poiss' ci-dessus en lui donnant comme entrée l'empilement des π_k^i . Rien de neuf donc (toujours grâce au'reshape' !).

Et les probabilités d'ordre deux, direz vous ? Sur le plan des principes, il y a la même analogie que pour les probas d'ordre un. La forme exponentielle des probabilités du tirage poissonnien conduit à une récurrence de même nature, et toujours de deux en deux :

$$\pi_{kl}^{ij;\bar{n}} = C_{\bar{n}} \sum_{Dik} \exp\left(\sum_{j=1}^{Q} \lambda^j . s_j\right) = C_{\bar{n}} \omega_k^i \omega_l^j \sum_{Eik} \exp\left(\sum_{h=1}^{Q} \lambda^h . s_h\right) = \left(C_{\bar{n}} / C_{\bar{n}-\bar{i}-\bar{j}}\right) \omega_k^i \omega_l^j \left(1 - \sum_h \pi_{kl}^{h;\bar{n}-\bar{i}-\bar{j}}\right)$$

où D_{ik} est l'ensemble des $\underline{n}$-échantillons tels que s_i contienne k et l , et E_{ik} l'ensemble des $\underline{n}$-$\underline{i}$-$\underline{j}$-échantillons dont aucun ne contienne k ou l . La constante multiplicative s'identifie en ajustant la somme des $\pi_k^{i,\underline{n}}$ à $n_i(ni-1)$. De ce fait, on obtient un programme de calcul relativement simple pour sa partie récursive. On observera, cependant, que chaque 'boite' indicée par $\underline{n}$ contient les $Q(Q-1)/2$ matrices NxN de probabilités d'inclusions doubles $\pi_k^{i}{}_l^j$ (sur ces questions on pourra se reporter à [8]) et que les problèmes d'initialisation, d'une part, de volume de calcul, d'autre part, deviennent assez casse-bonbons. Le rédacteur de ces lignes s'est arrêté au cas de deux échantillons disjoints (et donc de trois matrices de probas d'inclusion doubles - le programme est long mais devrait pouvoir se simplifier ; en tous cas, il n'a aucune vertu pédagogique !) , ce qui peut toujours servir.

Algorithmes de tirage :

Dans le cas multidimensionnel, c'est algorithme séquentiel qui est le plus naturel et le plus simple à écrire:

Unité 1: elle est attribuée à l'échantillon i avec la proba π_1^i (et à aucun d'eux avec la proba 1-$\Sigma_i \pi_1^i$) ;

- Calculer $\pi_k^{i,\ U-1\ ,\underline{n}-i}$ ou les $\pi_k^{i,\ U-1\ ,\underline{n}}$ selon le résultat du tirage.

…

- Unité k+1 : soit $\underline{n}_k = (\ n_k^i)$ le nombre d'unités selectionnées dans l'échantillon i après k 'étapes' (=unités). On calcule les proba $\pi_k^{i,\ U-\{1,\ldots k\}\ ,\ \underline{n}-\underline{n}_k}$. L'unité k+1 est attribuée à léchantillon i avec ces probabilités (où n'est pas attribuée avec la proba complémentaire de leur somme).

Cet algorithme est très efficace, rapide, précis et sur. En revanche, bien qu'on puisse imaginer un tirage 'unité par unité', il serait bien plus complexe et n'aurait quasiment aucun avantage sur le précédent. En particulier, il n'y a aucun moyen de tirer le premier échantillon, puis le second et ainsi de suite.

5. Applications et compléments

a) Tirage de deux échantillons avec une intersection contrôlée.

Il suffit de tirer trois échantillons. Si les deux échantillons de base sont Poissoniens, le paramètre pour l'intersection sera $\lambda^1 + \lambda^2$. Ceci dit on peut même spécifier les probas d'inclusion dans l'intersection (sauf 0 sinon ça gueule). Ceci dit, pour limiter la portée du miracle, on remarquera que les échantillons marginaux ne sont pas poissonniens. Il en va de même, par suite, de l'échantillon intersection. En revanche, l'échantillon 'réunion' de deux poissonniens conditionnels disjoints est un poissonnien conditionnel si les deux échantillonnages parents ont le même paramètre, ce qui est assez restrictif mais arrive parfois dans la nature.

Si on veut trois échantillons, on devra spécifier les tailles et les probabilités d'inclusion des sept sous-échantillons intersections, des 2^Q-1 dans le cas de Q échantillons.

b) Imputation d'une variable qualitative.

Une variable qualitative, CSP ou niveau d'instruction par exemple, est manquante et on désira imputer une valeur aux N individus qui sont dans ce triste état. Supposons ajusté un modèle qui donne, pour chaque individu k pour lequel la valeur est manquante, les probabilités π_k^i d'être affecté à la valeur i ($\Sigma_i \pi_k^i = 1$). Les sommes $\Sigma_k \pi_k^i = n_i$ sont supposées être des entiers (sinon on peut bricoler un vague raking-ratio pour s'y ramener ou faire plus rusé, mais passons). Il est bien connus que n_i est l'espérance du nombre d'unités imputées à i, et qu'il est naturel de respecter cette contrainte (en plus ça fait diminuer les variances, mais c'est une autre session). On peut donc appliquer ce qui vient d'être décrit à ce problème. Pour être exactement dedans il suffit de supprimer une des catégories i (pour des raisons de stabilité numérique, plutôt la plus grosse). Si on garde toutes les catégories (somme des probas individuelles égales à 1) , ça marche aussi, mais des problèmes de troncature numérique peuvent parfois se manifester pour des 'populations' pas trop grandes (quelques dizaines).

Une amusante remarque : si l'ajustement des probabilitéss a été réalisé à l'aide de régression logistiques , les λ^i sont tous calculés.

c) Supposons qu'on aie deux échantillons disjoints de tailles fixées n_1 et n_2. Dans le cas des probabilités égales et d'entropie maximale, les questions des probabilités diverses et variées et de l'algorithmique ne se posent pas tellement elles sont triviales. On sait, de plus, que la covariance entre les moyennes $\bar{y}_1$ et $\bar{y}_2$ sur les deux échantillons est négative et vaut $-1/N\ S_Y^2$,

c'est à dire que c'est tout petit. On a un résultat analogue pour $Cov(\overline{y}_1, \overline{x}_2)$, ce qui permet de négliger ce terme qu'on a aucun moyen d'estimer. On aimerait bien que ça se généralise au cas des probabilités inégales, en tous cas dans le cas entropiques (sinon c'est assez facile de trouver des contre-exemples). Pas de chance, ça ne marche pas en général. C'est vrai, en revanche, quand les échantillonnages poissoniens 'parents' ont le même paramètre λ (à une constante près , bien sur). Cela vient de deux fait :

- l'échantillon réunion est poissonien , comme on l'a déjà vu,
- conditionnellement à la réunion de s_1 et s_2 , s_1 (resp s_2) est un échantillon aléatoire simple. On se référera à [7] pour plus de précision.

Références :

[1] Chen,S.X., Dempster, A.P., et LIU,J.S. (1994). Weighting finite population sampling to maximize entropy. *Biometrika* ***81*** pp457-469.

[2] Chen, S.X. ,(1998). Weighting polynomial models and weighting sampling schemes for finite population. *The Annals of Statistics,* ***26,*** pp 1894-1915

[3] Deville, J.C., and Tillé, Y., (2000) Selection of several unequal probability samples from the same population. *Journal of Statistical Planning and Inference,* ***86*** pp 215-227

[4] Hajek, J.,(1981) Sampling from a finite population, *New-York, Marcel Dekker*

[5] Joe,H. (1990). A winning strategy for lotto game? *Canadian Journal of Statistics,****18*** pp233-244

[6] Tillé,Y. (2006?). Sampling algorithms with equal or unequal probabilities, *to appear*

[7] Qualité, L. (2004). Echantillonnage à entropie maximale, *mémoire de dea, université de Rennes 1*.

[8] Goga, C. (2003). Estimation de la variance dans les sondages à plusieurs échantillons et prise en compte de l'information auxiliaire par des modèles non-paramétriques, *Thèse de doctorat, université de. Rennes 2.*