Méthodes spécifiques pour Simuler des lois de probabilités discrètes

Prof. Mohamed El Merouani

2019/2020

Plan

Nous présentons quelques méthodes qui peuvent être efficientes pour des lois de probabilités particulières :

- Loi binomiale
- Loi de Poisson
- Loi géométrique
- Loi binomiale négative
- Loi hypergéométrique

Loi binomiale :

- Une variable aléatoire X de loi binomiale $B(n,p)$ peut être interprétée comme la somme de "n" expépériences, indépendantes de Bernoulli.
- Ainsi, il existe une façon d'échantillonnage en générant "n" nombres aléatoires $U_1, U_2, ..., U_n \sim \mathcal{U}(0,1)$ et en faisant égaliser X à celles d'entre elles qui sont inférieures à p.
- Si nous considérons la $i^{\text{ème}}$ expérience succès lorsque $U_i < p$ et que la probabilité de cet événement est égale à p, il est claire que l'on obtient une variable aléatoire binomiale de paramètres n et p.
- L'algorithme est :

Faire $X = 0$
Répéter n fois
    Générer $U \sim \mathcal{U}(0,1)$
    Si $U < p$, faire $X = X + 1$
Sortir X

Loi binomiale :

- Cette méthode requiere "n" nombres aléatoires et "n" vérifications (comparaisons, tests).
- Pour cela, on peut opter pour la méthode d'inversion suivante basée sur la formule recurrente qui permet d'obtenir la probabilité $P(X = i + 1) = p_{i+1}$.

$$P(X = i+1) = p_{i+1} = C_n^{i+1} p^{i+1} (1-p)^{n-(i+1)}$$

$$= \frac{(n-i)p}{(i+1)(1-p)} p_i$$

$$= \frac{(n-i)p}{(i+1)(1-p)} P(X = i)$$

Loi binomiale :

- Dans l'algorithme suivant, on note $P = P(X = i)$ et $F = P(X \leq i)$.

Générer $U \sim \mathcal{U}(0,1)$
Faire $i = 0$, $P = F = (1-p)^n$
Jusqu'à ce que $U < F$
    Faire $P = \frac{(n-i)p}{(i+1)(1-p)} P$, $F = F + p$
    $i = i + 1$
Sortir $X = i$

- Le nombre de vérifications (comparaisons) est un plus que la valeur de X, pour cela, on a besoin, en moyenne, de $(1 + np)$ comparaisons pour générer la variable X.
- Évidement, si le procédé est répété plusieurs fois pour les mêmes n et p, les F_i peuvent s'accumuler, comme dans la méthode générale d'inversion.

Loi binomiale :

- Lorsque n est grand, la méthode est peu efficiente et on peut opter pour une méthode générale.
- Autre possibilité, à partir du théorème centrale limite, en se basant sur le fait que $(*) Z = \frac{X - np + 0.5}{\sqrt{np(1-p)}}$ tend vers une loi normale $\mathcal{N}(0,1)$ lorsque n croît indéfiniment.
- Dans ce cas, on génére $Z \sim \mathcal{N}(0,1)$,
- On résoud $(*)$ précédente en X et on arrondit à une valeur entière non négative, en faisant

$$X = \max\{0, ent(-0.5 + np + Z\sqrt{np(1-p)}\}.$$

Loi de Poisson :

- Dans le cas de la loi de Poisson $\mathcal{P}(\lambda)$, pour λ petit, on peut utiliser la méthode d'inversion, selon la formule de récurrence $P(X = i+1) = \frac{\lambda}{i+1} P(X = i)$.
- Avec la même notation F, P d'avant, on obtient :

Générer $U \sim \mathcal{U}(0,1)$
Faire $i = 0$, $F = P = e^{-\lambda}$
Jusqu'à ce que $U < F$
    Faire $P = \frac{\lambda}{(i+1)} P$, $F = F + p$
    $i = i + 1$
Sortir $X = i$

- Le nombre de comparaisons sera "une" plus que la valeur de Poisson générée. En moyenne, il sera nécessaire $1 + \lambda$ comparaisons.

Loi de Poisson :

- Un autre algorithme intéressant se base sur la relation qui existe entre la loi de Poisson et la loi exponentielle.
- Si les durées (temps) T_i entre les événements d'un processus ponctuel sont i.i.d. (indépendantes identiquement distribués) selon une loi exponentielle $\mathcal{E}xp(\lambda)$, le nombre X des événements dans un intervalle de durée 1 suit la loi $\mathcal{P}(\lambda)$.
- Alors $\sum_{i=1}^{X} T_i \leq 1 \leq \sum_{i=1}^{X+1} T_i$.
- comme $T_i = \frac{-1}{\lambda} \ln(U_i)$, on peut reécrire la formule antérieure

$$\frac{-1}{\lambda} \sum_{i=1}^{X} \ln(U_i) \leq 1 \leq \frac{-1}{\lambda} \sum_{i=1}^{X+1} \ln(U_i)$$

- ou bien $\Pi_{i=1}^{X} U_i \geq e^{-\lambda} \geq \Pi_{i=1}^{X+1} U_i$

Loi de Poisson :

- D'où, on obtient, l'algorithme

Faire $a = 1$, $k = 0$
Jusqu'à que $a < e^{-\lambda}$
    Générer $U_k \sim \mathcal{U}(0,1)$
    Faire $a = aU_k$
    $k = k + 1$
Sortir $X = k$

- Pour λ grand, on peut approximer la loi de Poisson par une loi normale. Lorsque $\lambda > 10$, la loi de $(**) Z = \frac{X - \lambda + 0{,}5}{\sqrt{\lambda}}$ tend vers une loi normale $\mathcal{N}(0,1)$.
- On génère $Z \sim \mathcal{N}(0,1)$, on résoud $(**)$ précédente, par rapport à X et on arrondit à une valeur entière non négative, on obtient $X = \max\{0, ent(-0.5 + \lambda + Z\sqrt{\lambda}\}$.

Loi géométrique :

- La loi géométrique $\mathcal{G}e(p)$ est connue parfois comme la loi exponentielle discrète, puisqu'elle peut être générer en échantillant par discrétisation d'une loi exponentielle. En effet, supposons $Y \sim \mathcal{E}xp(\lambda)$.
- Soit $X = ent(Y)$. Alors,

$$P(X = r) = P(r \leq Y < r + 1) = \int_r^{r+1} \lambda e^{-\lambda s} ds$$

$$= e^{-\lambda r} - e^{-\lambda(r+1)} \qquad (***)$$

$$= \left(e^{-\lambda}\right)^r \left(1 - e^{-\lambda}\right)$$

pour $r = 0, 1, \cdots$, est la fonction de loi de probabilité $\mathcal{G}e(p = 1 - e^{-\lambda})$.

Loi géométrique :

- En prenant $\lambda = -\ln(1-p)$, on remarque que $(***)$ précédente est identique à la fonction de probabilité d'une loi géométrique de paramètre p. Donc,

$$X = ent\left(\frac{\ln U}{\ln(1-p)}\right) = ent\left(\frac{-V}{\ln(1-p)}\right)$$

  où $V = -\ln U \sim \mathcal{E}xp(1)$.
- On conclut que pour générer une variable géométrique $\mathcal{G}e(p)$, premièrement, on génère une variable exponentielle $\mathcal{E}xp(-\ln(1-p))$, et on arrondit à la valeur entière.

Loi binomiale négative :

- Il y a plusieurs algorithmes qui ont été proposés pour la génération d'une loi binomiale négative $\mathcal{BN}(k, p)$.
- Ici, nous citons seulement un seul algorithme qui utilise la relation donnée par l'égalité $P(X \leq r) = P(Y \geq k)$ où $X \sim \mathcal{BN}(k, p)$ et $Y \sim \mathcal{B}(p, k + r)$.
- Si nous disposons d'un procédé pour générer la variable binomiale, il est immédiat de construire un algorithme basé sur l'égalité précédente.

Loi hypergéométrique :

- On veut échantillonner à partir d'une loi hypergéométrique $\mathcal{H}(m, n, p)$, c'est-à-dire, on fait n tirages sans remise d'un ensemble de m éléments de deux classes, avec p la fraction d'éléments d'une classe et $q = 1 - p$ la fraction de l'autre.
- Un procédé simple pour générer la variable X, en notant $c_i, i = 1, 2$ le nombre restant d'éléments de chaque classe dans la population après X tirages, est le suivant :

Faire $X = 0$, $c_1 = mp$, $c_2 = m - c_1$
Répéter n fois
    Générer $U \sim \mathcal{U}(0, 1)$
    Si $U \leq \frac{c_1}{m}$, faire $X = X + 1$ et $c_1 = c_1 - 1$
        sinon, $c_2 = c_2 - 1$
    Faire $m = m - 1$
Sortir X