Quatrième partie

Annexes

CHAPITRE 27

Jeux de grattage

27.1 Corrigé possible du texte

27.1.1 Construction d'un modèle statistique général

Dans ce chapitre, on considère deux modèles. Un modèle probabiliste, que nous présenterons dans un second temps, sera celui de répartition uniformément au hasard des lots. Un modèle statistique formera le modèle général, où l'on ne suppose rien de l'uniformité ou de la non-uniformité de cette répartition. Parce qu'ici il est plus facile d'aller du général au particulier, nous construisons d'abord le modèle général (statistique). Nous spécifierons ensuite le modèle particulier (probabiliste) de répartition uniforme.

On part de 100 observations $c_1, \dots, c_{100}$, représentant le nombre de lots significatifs obtenus dans 100 carnets achetés à des moments aléatoires d'une année. Leurs valeurs sont

$$\Big|\big\{j \in [\![1,100]\!] :\ c_j = 0\big\}\Big| = 33 \qquad \text{et} \qquad \Big|\big\{j \in [\![1,100]\!] :\ c_j = 1\big\}\Big| = 67\ .$$

On modélise maintenant ces observations comme la réalisation d'un vecteur aléatoire $C = (C_1, \dots, C_{100})$.

Loi du nombre de lots C_j dans un seul carnet

À cet effet, il faut expliquer davantage comment un carnet se retrouve chez un buraliste : on suppose que la société de jeux conçoit d'un coup l'ensemble d'une tranche d'émission, i.e., 10 000 carnets, contenant exactement les lots discutés au tableau 19.1. Ces carnets sont alors répartis uniformément au hasard entre les buralistes du pays. Même en achetant un carnet chez un buraliste fixé, on prélève donc un carnet uniformément au hasard dans une tranche d'émission.

On suppose également pour le moment que chaque tranche comporte les mêmes proportions de carnets contenant 0, 1, 2, etc., et au plus 50 lots significatifs. (Comme le carnet contient 50 tickets, il ne peut y avoir plus de 50 lots significatifs.) On étudiera le cas général de proportions aléatoires ci-dessous. On note le vecteur de ces proportions déterministes par ν, qui est une probabilité sur $[\![0,50]\!]$; il est évidemment inconnu et on cherche à l'inférer.

Avec ces notations et hypothèses, lorsque l'on prélève un carnet j au hasard dans une tranche, son nombre de lots significatifs C_j est distribué selon ν (où l'aléa est

engendré uniquement par le choix aléatoire d'un élément dans la population donnée par l'ensemble des carnets d'une tranche).

Loi du vecteur de nombre de lots C

On a supposé que pour ce jeu, 10 tranches d'émission étaient réalisées par an et que l'on prélevait 100 carnets, de manière étalée dans le temps. Cela assure qu'on prélève peu de carnets par tranche, de l'ordre d'une dizaine, à comparer à l'ensemble des 10 000 carnets de la tranche. Comme on le rappelle ci-dessous, on peut donc approximer ces tirages, *a priori* sans remise, par des tirages avec remise.

En conséquence, les observations C_j sont indépendantes et identiquement distribuées : le vecteur aléatoire $C = (C_1, \dots, C_{100})$ est un 100–échantillon de loi commune ν.

> Remarque au passage. On définit alors le vecteur du décompte des occurrences $N = (N_0, \dots, N_{50})$ de la manière suivante : pour tout $k \in [\![0, 50]\!]$,
>
> $$N_k = \Big| \big\{ j \in [\![1, 100]\!] : \ C_j = k \big\} \Big| \ .$$
>
> Comme on le rappelle ci-dessous, par définition, la loi de N est alors la loi multinomiale $\mathcal{M}(100, \nu)$ de paramètres 100 et ν. Elle charge uniquement les 51–uplets $(n_0, \dots, n_{50})$ d'entiers tels que $n_0 + \ldots + n_{50} = 100$, et pour un tel 51–uplet,
>
> $$\mathbb{P}\big\{ (N_0, \dots, N_{50}) = (n_0, \dots, n_{50}) \big\} = \frac{100!}{n_0! \ \dots \ n_{50}!} \prod_{j=0}^{50} \nu_j^{n_j} \ .$$
>
> La loi multinomiale est la généralisation de la loi binomiale au décompte d'occurrences dans un échantillon dont les éléments peuvent prendre plus de deux valeurs différentes. Pour $k \in [\![0, 50]\!]$ fixé, la loi de N_k est la loi binomiale $\mathrm{Bin}(100, \nu_k)$ de paramètres 100 et ν_k.

Ici, on a donc affaire à un modèle paramétrique, puisque l'ensemble des lois possibles (l'ensemble des ν possibles) est formé par le simplexe des probabilités de $\mathbb{R}^{51}$, noté $\mathcal{P}\big([\![0, 50]\!]\big)$:

$$\mathcal{P}\big([\![0, 50]\!]\big) = \left\{ \nu = (\nu_0, \dots, \nu_{50}) \in \mathbb{R}^{51} : \quad \sum_{j=0}^{50} \nu_j = 1 \quad \text{et} \quad \nu_i \geqslant 0 \ \text{ pour tout } \ i \in [\![0, 50]\!] \right\} .$$

Pour $\nu \in \mathcal{P}\big([\![0, 50]\!]\big)$ et $j \in [\![0, 50]\!]$, on notera indifféremment la probabilité que ν associe à j par ν_j ou $\nu\{j\}$. Le contexte (la considération de ν comme un vecteur ou comme une probabilité) déterminera quelle est l'écriture naturelle.

Il y a cinquante paramètres, ce qui constitue un nombre fini mais plutôt élevé. Par ailleurs, même si l'on ne rapporte pas d'observation c_j ayant une valeur plus grande que 2, on ne peut pas encore, à ce moment de la modélisation, partir du principe que ν met une probabilité nulle à l'ensemble $[\![3, 50]\!]$ et que le modèle est donc décrit par trois paramètres uniquement. Cela viendra ultérieurement, au paragraphe 27.1.3 et sous une hypothèse supplémentaire à discuter.

Retour sur l'approximation d'un tirage sans remise par un tirage avec remise

On commence par définir les lois associées respectivement aux tirages avec et sans remise de m éléments parmi M. On suppose que les observations peuvent prendre ici k modalités dont les valeurs sont représentées par $\{1, \dots, k\}$ et qu'il y a M_j éléments parmi les M pour la modalité j. Ainsi, $M = M_1 + \ldots + M_k$.

On note $X_1, \dots, X_m$ les résultats des tirages, avec, pour $t = 1, \dots, m$, les valeurs possibles $X_t \in \{1, \dots, k\}$. On définit le vecteur des occurrences $N = (N_1, \dots, N_k)$ de la manière suivante : pour tout $j \in \{1, \dots, k\}$,

$$N_j = \sum_{t=1}^{m} \mathbb{I}_{\{X_t = j\}} \ .$$

On montre par des arguments élémentaires de dénombrement les deux résultats suivants. Pour deux entiers $0 \leqslant r \leqslant s$, on notera

$$\binom{s}{r} = \frac{s!}{r!(s-r)!}$$

le coefficient binomial représentant le nombre de choix de r éléments parmi s.

Définition–Théorème 27.1 Lorsque dans l'expérience définie ci-dessus, les tirages s'effectuent avec remise, la loi de N est appelée la loi multinomiale de paramètres m et

$$\boldsymbol{p} = (p_1, \dots, p_k) = \left(\frac{M_1}{M}, \dots, \frac{M_k}{M} \right) \ .$$

On la note $\mathcal{M}(m, \boldsymbol{p})$. Elle admet pour support les k–uplets d'entiers $(\ell_1, \dots, \ell_k)$ tels que $\ell_1 + \ldots + \ell_k = m$. Elle affecte une probabilité

$$\frac{m!}{\ell_1! \ \ldots \ \ell_k!} \ p_1^{\ell_1} \ \ldots \ p_k^{\ell_k}$$

à un tel k–uplet.

Définition–Théorème 27.2 Lorsque dans l'expérience définie ci-dessus, les tirages s'effectuent sans remise, la loi de N est appelée la loi hypergéométrique de paramètres

m et $(M_1, \ldots, M_k)$. On la note $\mathcal{H}\big(m, (M_1, \ldots, M_k)\big)$. Elle admet pour support les k–uplets d'entiers $(\ell_1, \ldots, \ell_k)$ tels que $\ell_1 + \ldots + \ell_k = m$. Elle affecte une probabilité

$$\frac{\displaystyle\prod_{j=1}^{k} \binom{M_j}{\ell_j}}{\displaystyle\binom{M_1 + \ldots + M_k}{m}}$$

à un tel k–uplet.

Le résultat de convergence ci-dessous justifie l'approximation précédemment considérée d'une expérience de tirages de carnets sans remise par des tirages de carnets avec remise.

Lemme 27.3 Lorsque $M_j \to \infty$ pour tout $j \in \{1, \ldots, k\}$ de sorte qu'il existe un vecteur de probabilité $\boldsymbol{p} = (p_1, \ldots, p_k)$ avec $p_j > 0$ et

$$\frac{M_j}{M_1 + \ldots + M_k} \longrightarrow p_j \tag{27.1}$$

pour tout $j \in \{1, \ldots, k\}$, alors

$$\mathcal{H}\big(m, (M_1, \ldots, M_k)\big) \rightsquigarrow \mathcal{M}(m, \boldsymbol{p}) \, .$$

Preuve Il s'agit de prouver que pour tout k–uplet d'entiers $(\ell_1, \ldots, \ell_k)$ tel que $\ell_1 + \ldots + \ell_k = m$,

$$\frac{\displaystyle\prod_{j=1}^{k} \binom{M_j}{\ell_j}}{\displaystyle\binom{M_1 + \ldots + M_k}{m}} \longrightarrow \frac{m!}{\ell_1! \ldots \ell_k!} \, p_1^{\ell_1} \ldots p_k^{\ell_k} \, .$$

Or, on note que puisque les $M_j \to \infty$, on a les équivalences

$$\binom{M_j}{\ell_j} = \frac{M_j!}{\ell_j!(M_j - \ell_j)!} \sim \frac{M_j^{\ell_j}}{\ell_j!} \quad \text{et} \quad \binom{M_1 + \ldots + M_k}{m} \sim \frac{(M_1 + \ldots + M_k)^m}{m!} \, .$$

La preuve est conclue en injectant les équivalences données par (27.1). □

Extension à des proportions aléatoires dans une tranche

On discute ici le cas où les proportions ne sont plus déterministes et varient d'une tranche à l'autre, selon un certain aléa. C'est le cas notamment lorsque la génération de la position des lots est faite selon un hasard total, ainsi qu'expliqué au paragraphe 27.1.2.

Dans ce cas, on note $\mu_1, \mu_2, \dots, \mu_{10}$ les proportions ν correspondant aux dix émissions dans lesquelles on a prélevé des carnets. On suppose que ces proportions sont les réalisations de variables aléatoires indépendantes et identiquement distribuées selon une loi $\mathcal{V}$ sur $\mathcal{P}\big(\llbracket 0, 50 \rrbracket\big)$. En effet, cela revient simplement à supposer qu'elles ont été engendrées par le même processus aléatoire de fabrication. Nous donnons un exemple d'une telle situation au paragraphe 27.1.2.

Lorsque l'on s'intéresse au nombre C_j de lots significatifs contenus dans un seul carnet, on peut imaginer que le tirage de ce nombre est effectué en deux temps : on tire d'abord la valeur du vecteur de proportions μ selon $\mathcal{V}$, puis on tire C_j selon μ. Il est équivalent d'effectuer directement un tirage de C_j selon la loi

$$\nu = \int_{\mathcal{P}\left(\llbracket 0,50 \rrbracket\right)} \mu \, \mathrm{d}\mathcal{V}(\mu) \ . \tag{27.2}$$

Cela montre que la modélisation précédente, introduite pour des proportions déterministes, est encore valable dans ce cadre de proportions aléatoires, au sens où C est bien encore un échantillon de loi ν. On prendra cependant garde qu'on fera œuvre d'inférence ici sur ν et non sur $\mathcal{V}$. Dit autrement, le vrai modèle sous-jacent (décrit par $\mathcal{V}$) est non-paramétrique et nous ne nous intéressons qu'à ν, qui est une loi image de $\mathcal{V}$.

27.1.2 Construction et rejet d'un modèle probabiliste de référence : répartition totalement aléatoire des lots significatifs

Ici, on définit un cas particulier du modèle statistique général exhibé précédemment et qui nous servira de modèle de référence : on étudie la méthode de génération $\mathcal{V}_{\text{unif}}$ qui place les lots significatifs totalement au hasard dans la tranche d'émission.

Dans un premier temps, on construit précisément ce modèle et on détermine le vecteur ν associé selon la formule (27.2). Dans un second temps, on montre que les observations recueillies $(c_1, \dots, c_{100})$ contredisent ce modèle (au sens des tests statistiques).

Construction du modèle probabiliste de répartition totalement aléatoire

On suppose ici que l'ensemble des lots (et des tickets perdants) est obtenu par considération d'un arrangement aléatoire des lots mineurs et significatifs décrits au tableau 19.1 parmi les 500 000 tickets d'une tranche. Cela définit les gains ou absences de gains $T_1, \dots, T_{500\,000}$ portés par les tickets. Ces derniers sont alors séquentiellement groupés 50 par 50 pour former les carnets. Ainsi, pour $1 \leqslant k \leqslant 10\,000$, le k–ième carnet de la tranche contient les lots $T_{50(k-1)+1}, \dots, T_{50k}$.

Cette procédure décrit en particulier la loi $\mathcal{V}_{\text{unif}}$ dont il a été question plus haut. Nous calculons maintenant la loi ν_{ref} induite selon la formule (27.2), ou tout du moins, dans un premier temps, une bonne approximation. Cette loi formera la loi de référence lors de la construction d'un test ci-dessous.

La méthode de génération selon $\mathcal{V}_{\text{unif}}$ est équivalente au fait de tirer séquentiellement et sans remise les valeurs $T_1, \dots, T_{500\,000}$. On s'intéresse alors à la loi du nombre de lots significatifs présents dans le premier carnet : par symétrie, sa loi est identique à celle du nombre de lots contenus dans tout autre carnet et est donc égale à ν.

Or, comme il découle de la Définition–théorème 27.2, ce nombre suit la loi ν_{ref} donnée par la première marginale de $\mathcal{H}\big(50, (7\,697, 500\,000 - 7\,697)\big)$. Selon les mêmes arguments que précédemment (taille d'échantillon prélevé faible par rapport à la population), ce tirage peut être approximé par un tirage avec remise, voir le Lemme 27.3. Ainsi,

$$\mathcal{H}\big(50, (7\,697, 500\,000 - 7\,697)\big) \approx \mathcal{M}\big(50, (p_0, 1 - p_0)\big)$$

de sorte que $\nu_{\text{ref}} \approx \text{Bin}(50, p_0)$, où l'on a noté $p_0 = 7\,697/500\,000$ la proportion de lots significatifs dans une tranche d'émission. Un petit abus (qu'on quantifiera par le calcul) nous conduira à écrire dans la suite $\nu_{\text{ref}} = \text{Bin}(50, p_0)$.

Quelle que soit la méthode de génération $\mathcal{V}$ des carnets, un ticket pris au hasard dans un carnet pris au hasard porte un lot significatif avec probabilité $p_0 = 7\,697/500\,000$, puisqu'il y a 7 697 lots significatifs dans une tranche de 500 000 tickets. On vient de prouver que la méthode de génération aléatoire $\mathcal{V}_{\text{unif}}$ étend en un sens ce fait à l'ensemble des tickets d'un carnet donné, car le nombre de lots significatifs dans un carnet est donné par la loi binomiale $\nu_{\text{ref}} = \text{Bin}(50, p_0)$: autrement dit, il y a indépendance entre les gains portés par des tickets dans un carnet. C'était bien sûr attendu au vu de la méthode de génération (et au vu de l'approximation des tirages sans remise par des tirages avec remise).

Test d'ajustement au modèle de répartition totalement aléatoire

On souhaite tester ici si l'allocation des lots significatifs par carnet est réalisée par la société de jeux selon le modèle de répartition totalement aléatoire (c'est notre hypothèse H_0) ou si des contraintes sont imposées sur leur répartition par carnet (c'est l'hypothèse alternative H_1). Ces hypothèses se traduisent mathématiquement, avec les notations précédentes, par

$$H_0 : \nu = \nu_{\text{ref}} \big(= \text{Bin}(50, p_0)\big) \qquad \text{contre} \qquad H_1 : \nu \neq \nu_{\text{ref}} \ .$$

En particulier, sous H_0, les nombres de lots significatifs observés $C = (C_1, \dots, C_{100})$ forment un 100–échantillon de loi $\text{Bin}(50, p_0)$.

La probabilité de quelques événements sous la loi $\text{Bin}(50, p_0)$ est rappelée dans le tableau 27.1. Il y a en particulier une probabilité environ égale à 18 % qu'une variable aléatoire de loi $\text{Bin}(50, p_0)$ prenne des valeurs supérieures ou égales à 2. On note

$$S_j = \mathbb{I}_{\{C_j \geqslant 2\}} \ , \qquad \text{pour} \quad j = 1, \dots, 100.$$

$\mathbb{P}\{\mathrm{Bin}(50,\, p_0) = 0\}$	$\mathbb{P}\{\mathrm{Bin}(50,\, p_0) = 1\}$	$\mathbb{P}\{\mathrm{Bin}(50,\, p_0) \geqslant 2\}$
0.4604	0.3599	0.1797

Tableau 27.1. Probabilités de quelques événements sous la loi $\nu_{\mathrm{ref}} = \mathrm{Bin}(50,\, p_0)$, arrondies à 10^{-4} près.

Ces variables aléatoires indiquent s'il y a ou non au moins deux lots significatifs dans un carnet. Sous H_0, on attend donc, par la loi des grands nombres :

$$\overline{S}_{100} = \frac{S_1 + \ldots + S_{100}}{100} \approx 0.18 \ .$$

Or, sur les observations, la réalisation $\overline{s}_{100}$ de $\overline{S}_{100}$ vaut $\overline{s}_{100} = 0$. On soupçonne que H_0 est infondée et l'on va maintenant quantifier mathématiquement ce sentiment, en exhibant une p–valeur.

D'une manière générale (i.e., sous H_0 comme sous H_1), les variables aléatoires $S_1, \ldots, S_{100}$ sont indépendantes et identiquement distribuées selon une loi de Bernoulli $\mathrm{Ber}(q)$ de paramètre

$$q = \nu\big([\![2,\, 50]\!]\big) \ .$$

Sous H_0, on a

$$q = q_{\mathrm{ref}} \overset{\text{not.}}{=} \mathbb{P}\{\mathrm{Bin}(50,\, p_0) \geqslant 2\} \approx 0.18 \ .$$

Un test de H_0 et H_1 est donc donné en particulier par un test de

$$H_0' : q = q_{\mathrm{ref}} \qquad \text{contre} \qquad H_1' : q < q_{\mathrm{ref}} \ .$$

On a pris ici un test unilatère parce que l'on suppose que le détective avait recueilli au préalable des témoignages de buralistes qui laissaient entendre qu'ils observaient rarement, voire jamais, l'obtention de deux lots significatifs dans un carnet. Bien entendu, si cette information externe n'avait pas été disponible, il aurait fallu mener un test bilatère.

Les tests de conformité pour le paramètre d'une loi de Bernoulli ont été vus notamment au paragraphe 4.4. Ils utilisent comme statistique de test la moyenne empirique correctement centrée et renormalisée par le paramètre à tester, ici, q_{ref}. On les met en œuvre au paragraphe suivant.

On va en effet effectuer d'abord un calcul direct, pour changer et parce que l'on a accès à une détermination informatique aisée des quantiles. Ce calcul direct met en jeu une statistique similaire, la somme $S = S_1 + \ldots + S_{100}$, dont on sait qu'elle suit la loi $\mathrm{Bin}(100, q)$; en particulier, sous H_0, elle suit la loi $\mathrm{Bin}(100, q_{\mathrm{ref}})$. Vu la forme des hypothèses, un test de niveau $\alpha \in \,]0, 1[$ rejette H_0 lorsque S est trop petite, *id est*, utilise une zone de rejet sur S de la forme $[\![0,\, r_\alpha]\!]$.

k	12	11	10	9
$\mathbb{P}\{\mathrm{Bin}(100, q_{\mathrm{ref}}) \leqslant k\}$	7×10^{-2}	4×10^{-2}	2×10^{-2}	10^{-2}

k	6	3	1	0
$\mathbb{P}\{\mathrm{Bin}(100, q_{\mathrm{ref}}) \leqslant k\}$	5×10^{-4}	5×10^{-6}	6×10^{-8}	3×10^{-9}

Tableau 27.2. Fonction de répartition de la loi $\mathrm{Bin}(100, q_{\mathrm{ref}})$; valeurs données avec un seul chiffre significatif.

Le seuil r_α est fixé le plus grand possible de telle sorte que l'erreur de première espèce soit inférieure ou égale à α, soit ici,

$$\mathbb{P}_{H_0}\{S \in [\![0, r_\alpha]\!]\} = \mathbb{P}\{\mathrm{Bin}(100, q_{\mathrm{ref}}) \leqslant r_\alpha\} \leqslant \alpha \ .$$

Le tableau 27.2 permet de calculer r_α pour certaines valeurs de α. Par exemple, pour les niveaux usuels $\alpha = 5$ % et $\alpha = 1$ %, on a $r_{5\%} = 11$ et $r_{1\%} = 9$; on a également, entre autres, que $r_{0.05\,\%} = 6$. On a même $r_\alpha = 1$ pour $3 \times 10^{-9} \leqslant \alpha \leqslant 6 \times 10^{-8}$.

Or, les observations sont $s_1 = \ldots = s_{100} = 0$, de sorte que la réalisation s de S vaut $s = 0$. Le test précédent rejette ainsi H_0 à tout niveau α, pour $\alpha \geqslant 3 \times 10^{-9}$. Ce n'est que lorsque l'on s'oblige à avoir une erreur de première espèce inférieure à 3×10^{-9} que le test ne rejette plus H_0. Cette valeur seuil est appelée la p–valeur du test, voir le paragraphe 4.2 ; elle est ici très inférieure aux seuils usuels de 5 % et 1 %.

Conclusion intermédiaire

Sans aucun doute, on rejette H_0 et on peut affirmer que cette dernière contredit gravement les données recueillies : le paramètre q est significativement plus petit que la valeur $q_{\mathrm{ref}} = 18\,\%$.

Si l'on veut reproduire l'expérience (par exemple, pour faire foi, sous contrôle d'huissier), faut-il vraiment acheter 100 carnets, et combien cela coûtera-t-il (hors frais d'huissier) ? Cela revient à se demander quel est le plus petit n tel qu'une réalisation égale à 0 pour $S_1 + \ldots + S_n$ entraîne le rejet de H_0, avec une erreur de première espèce contrôlée, disons, par le seuil très faible 1 ‰. Avec le même raisonnement que ci-dessus, cela consiste à déterminer le plus petit n tel que

$$\mathbb{P}\{\mathrm{Bin}(n, q_0) = 0\} \leqslant 1\,‰ \ .$$

Un calcul informatique montre qu'il s'agit de $n = 35$. Or, il est facile et bon marché d'acheter 35 carnets (dans plusieurs tranches d'émission, pour conserver la qualité

de l'approximation de tirage avec remise, donc, pas tous à la fois). En effet, le paragraphe 27.1.4 indique que l'on peut compter sur un solde (différence entre le gain procuré par le ticket et la mise de 3 €) typique moyen de l'ordre de -1.15 € par ticket. L'achat d'un carnet entier revient donc environ à 57.50 €, et celui de 35 carnets, à environ 2 000 €. C'est un budget raisonnable pour un enquêteur (détective ou journaliste). Pour une garantie théorique moindre, par exemple une majoration de l'erreur de première espèce par le seuil typique de 5 %, un nombre $n = 16$ d'achats de carnets suffit.

Variante du test précédent avec davantage d'approximations

À titre informatif, on regarde ce qu'aurait donné le test asymptotique du paragraphe 4.4 : sa considération consiste essentiellement à ne plus utiliser les quantiles calculés informatiquement et reportés dans le tableau 27.2, mais à prendre à la place les quantiles de la loi normale. On utilise la convergence

$$\frac{\sqrt{n}}{\sqrt{q(1-q)}}\left(\overline{S}_n - q\right) \rightsquigarrow \mathcal{N}(0,1)$$

pour tout $q \in]0,1[$, assurée par le théorème de la limite centrale. En particulier, sous H_0,

$$\frac{\sqrt{n}}{\sqrt{q_{\mathrm{ref}}(1-q_{\mathrm{ref}})}}\left(\overline{S}_n - q_{\mathrm{ref}}\right) \rightsquigarrow \mathcal{N}(0,1)\ ,$$

où l'on note par ailleurs que la statistique donnée par le membre de gauche a tendance à être plus petite sous H_1 que sous H_0. Un test asymptotiquement de taille $\alpha \in]0,1[$ pour H_0 et H_1 est donc donné par

$$\phi(C) = \mathbb{I}_{\left\{\overline{S}_n < q_{\mathrm{ref}} - z_{1-\alpha}\sqrt{q_{\mathrm{ref}}(1-q_{\mathrm{ref}})/n}\right\}}\ .$$

Ici, on effectue l'hypothèse que l'approximation donnée par la convergence en loi est déjà bonne lorsque $n = 100$. La détermination de la p–valeur au vu des observations est alors effectuée selon les indications de la remarque 4.11. Elle vaut

$$p = \mathbb{P}\left\{N \leqslant \frac{\sqrt{100}}{\sqrt{q_{\mathrm{ref}}(1-q_{\mathrm{ref}})}}\left(\overline{s}_{100} - q_{\mathrm{ref}}\right)\right\} = \Phi(-4.68) = 1.43 \times 10^{-6}\ ,$$

où l'on a noté Φ la fonction de répartition de la loi normale standard et où l'on a précisé les valeurs numériques avec trois chiffres significatifs uniquement. Elle est supérieure à celle précédemment calculée mais elle conduit toujours à un rejet clair de H_0.

> Remarque au passage. On se gardera cependant de dire que la procédure précédente fondée sur les quantiles de la loi binomiale est plus exacte ou plus

satisfaisante. Toute modélisation comporte une part d'imprécision ou d'erreur et il est difficile de quantifier quelle modélisation est la moins mauvaise face à un problème concret. On gardera cela en mémoire en lisant les autres variantes discutées ci-dessous.

Une autre approximation possible dans la procédure fondée sur les lois binomiales aurait été de continuer la chaîne d'approximations pour la loi de référence ν_{ref} selon

$$\nu_{\text{ref}} \approx \text{Bin}(50, p_0) \approx \mathcal{P}(50\, p_0) \ ,$$

en effectuant l'approximation dite binomiale–Poisson. On rappelle que cette dernière est justifiée par le résultat suivant.

Lemme 27.4 Soit (p_n) une suite dans $[0,1]$ telle que $np_n \to \lambda$, pour un réel $\lambda > 0$. Alors,

$$\text{Bin}(n, p_n) \ \rightsquigarrow \ \mathcal{P}(\lambda) \ .$$

Des conditions empiriques assurent la validité de l'approximation $\text{Bin}(n,p) \approx \mathcal{P}(np)$ correspondante : que $n \geqslant 30$ et $np \in [1, 10]$. Ici, $n = 50$ et $\lambda_{\text{ref}} = 50\, p_0 = 0.7697$, de sorte que les conditions sont presque respectées.

On a besoin, pour mettre en œuvre le test discuté plus haut, de la valeur de $q_{\text{ref}} = \nu\big(\llbracket 2,\, 50 \rrbracket\big)$, que l'on se propose d'approcher par

$$q_{\text{ref}} \approx \mathbb{P}\{P \geqslant 2\} = 1 - \Big(e^{-\lambda_{\text{ref}}} + e^{-\lambda_{\text{ref}}} \lambda_{\text{ref}}\Big) = 0.1804 \qquad \text{où} \qquad P \sim \mathcal{P}(50\, p_0) \ ,$$

à comparer à la valeur 0.1797 proposée par le tableau 27.1. La différence ténue entre ces deux valeurs conduit aux mêmes ordres de grandeur pour la p–valeur associée au test et aux observations.

Remarque 27.5 L'intérêt des deux approximations présentées (utilisation des quantiles de la loi normale, approximation binomiale–Poisson) est qu'avec elles, une calculatrice ordinaire et une table de la loi normale suffisent à réaliser le test. On n'a alors pas besoin de recourir à un logiciel de statistique comme `Matlab`.

Variante du test précédent avec moins d'approximations

Étant donné le modèle idéal décrit par la génération $\mathcal{V}_{\text{unif}}$, on peut exhiber un test exact, donné par des considérations de dénombrement. Nous allons ici ne faire qu'une allusion brève à sa mise en œuvre et nous concentrer sur le calcul de sa p–valeur. On l'effectue sous l'hypothèse (qui précise la réalité) qu'on a tiré exactement 10 carnets dans 10 tranches d'émission successives.

Comme précédemment, dans notre cadre et au vu de nos observations, la p–valeur est donnée par la probabilité qu'avait de se produire le fait que dans tous les carnets tirés, on ait eu aucun ou un seul lot significatif. Soit o la probabilité que dans une

tranche d'émission, les 10 carnets tirés aient chacun contenu au plus un lot significatif : on a alors $p = o^{10}$.

On calcule maintenant o de manière exacte, en recourant à des lois hypergéométriques. Vu la symétrie du processus de génération, la probabilité o est la probabilité des enchaînements suivants :

- on tire sans remise 50 tickets parmi les 500 000 tickets et il y en a $k_1 \in \{0, 1\}$ contenant un lot significatif ;
- on tire sans remise 50 tickets parmi les $500\,000 - 50$ tickets restants (dont $7\,697 - k_1$ portent un lot significatif) et parmi eux, $k_2 \in \{0, 1\}$ portent un lot significatif ;
- plus généralement, pour $t \in [\![2, 10]\!]$, on tire sans remise 50 tickets parmi les $500\,000 - 50(t-1)$ tickets restants (dont $7\,697 - (k_1 + \ldots + k_{t-1})$ portent un lot significatif) et parmi eux, $k_t \in \{0, 1\}$ portent un lot significatif.

Soit $h(k, 50, M', M)$ la probabilité qu'une variable aléatoire de loi $\mathcal{H}\big(50, (M', M)\big)$ vale $(k, 50 - k)$. On réécrit alors o comme

$$o = \prod_{(k_1,\ldots,k_{10})\in\{0,1\}^{10}} \prod_{t=1}^{10} h\big(k_t,\, 50,\, 7\,697 - (k_1 + \ldots + k_{t-1}),\, 500\,000 - 50(t-1)\big)\ .$$

Ainsi formulée, la probabilité o est calculable par un logiciel informatique aux problèmes de précision près et bien que cela requière de l'ordre de 2^{10}, donc plusieurs milliers, de calculs. Les méthodes précédentes consistaient à dire que les termes h dans le produit ci-dessus définissant o ne dépendaient que de leur premier argument k_t et non des suivants : c'était l'approximation de la loi hypergéométrique par la loi binomiale. On pourrait également déterminer par le calcul un majorant commun à tous ces termes. Dans les deux cas, le lecteur se persuadera avec l'utilisation de la fonction `dhypg` de la boîte à outils `StixBox` de `Matlab` (voir le paragraphe 27.3.2) que l'ordre de grandeur 10^{-9} pour la p–valeur est préservé.

Nous ne recommandons pas ce schéma exact, qui effectue en réalité un dénombrement efficace des arrangements des lots significatifs parmi les tickets tels que les dix premiers carnets ainsi construits ne contiennent pas plus d'un lot significatif. Il présente donc un argument combinatoire. Nous trouvons, mais c'est là une question de goût, que le schéma par approximation binomiale a l'avantage de nous ramener à un cadre parfaitement stochastique. Par ailleurs, et c'est une remarque plus objective, s'il est facile de calculer ici la p–valeur au vu des observations, il ne serait pas du tout aisé de préciser de manière un peu automatique un test de taille α, pour toute valeur α.

27.1.3 Inférence sur ν

Majoration de la probabilité q d'avoir deux lots significatifs dans un carnet tiré au hasard

Les tests du paragraphe précédent nous ont appris quelque chose, puisque tous, et sans hésitation, ont rejeté l'hypothèse H_0. On sait comment il ne faut pas modéliser la

répartition ν : elle n'est pas issue du seul hasard $\mathcal{V}_{\text{unif}}$. Cela ne nous donne pas encore la loi $\mathcal{V}$ de son engendrement. En particulier, le paramètre

$$q = \nu\big(\llbracket 2,\, 50 \rrbracket\big)$$

est certes significativement plus petit que $q_{\text{ref}} \approx 18\,\%$, mais on voudrait quantifier cette assertion et en donner un majorant précis.

On utilise à cet effet la dualité entre tests et intervalles de confiance, vue au paragraphe 4.4. L'intervalle de confiance (au niveau $1-\alpha$) sur le paramètre q est donné par tous les paramètres q' qui seraient acceptés par un test unilatère donné de

$$H_0 : q = q' \qquad \text{contre} \qquad H_1 : q < q' \ ,$$

par exemple celui utilisant la statistique $S = S_1 + \ldots + S_{100}$, où $S \sim \text{Bin}(100,\, q')$ sous H_0.

Le même raisonnement qu'au paragraphe précédent montre que H_0 est rejetée au vu des observations $c_1, \ldots, c_{100}$ et au niveau α lorsque

$$\mathbb{P}\big\{\text{Bin}(100,\, q') = 0\big\} = (1-q')^{100} < \alpha \ ,$$

soit lorsque

$$q' > q_\alpha = 1 - \exp\left(\frac{\log \alpha}{100}\right) \ .$$

La réalisation de l'intervalle de confiance au niveau $1-\alpha$ sur q est alors $[0,\, q_\alpha]$. Pour l'erreur de première espèce typique $\alpha = 5\,\%$, on a la réalisation $q_{5\%} = 0.030 \approx 3\,\%$.

> Remarque au passage. On ne peut évidemment jamais montrer par analyse statistique que cette valeur de q est nulle. Même avec davantage d'observations rapportant qu'aucun carnet gratté ne contenait strictement plus d'un lot significatif, l'intervalle de confiance se rétrécirait autour de 0, à vitesse $1/\sqrt{n}$, où n est le nombre de telles observations, mais ne sera jamais réduit au singleton $\{0\}$. Cependant, la borne de $3\,\%$ exhibée ci-dessus est à apprécier à l'aune de la valeur $q_{\text{ref}} \approx 18\,\%$ correspondant au modèle de répartition totalement aléatoire.

Nécessité (au vu des observations) de l'existence de carnets contenant au moins deux lots significatifs

On montre ici qu'il est improbable, au vu des observations, que la valeur $q_{\text{ref}} = 0$ convienne pour la probabilité q qu'un carnet tiré au hasard contienne deux lots significatifs ou plus. À cet effet, on va construire un test ϕ de

$$H_0 : q = 0 \qquad \text{contre} \qquad H_1 : q > 0 \ .$$

Dans la suite, on regardera essentiellement l'image suivante de ν,

$$(a, b, q) = \Big(\nu\{0\},\, \nu\{1\},\, \nu\big(\llbracket 0, 50 \rrbracket\big)\Big) \ .$$

On voit qu'ici on va donner une place particulière à deux paramètres a et b.

Le cas $q = 0$ est celui où tous les carnets contiennent aucun ou un unique lot significatif. Le tableau 19.1 indique que cela est équivalent à

$$b = \frac{7\,697}{10\,000} \overset{\text{not.}}{=} b_0 \qquad \text{et} \qquad a = 1 - b_0 \ .$$

Pour construire le test ϕ, on se ramène alors, par exemple, à tester l'ajustement de a à la valeur de référence $a_{\text{ref}} = 1 - b_0$, i.e., à construire un test ϕ' de

$$H_0' : a = 1 - b_0 \qquad \text{contre} \qquad H_1 : a > 1 - b_0 \ .$$

La forme de l'alternative est justifiée par les faits suivants. Le nombre total de lots significatifs étant constant, si des carnets en contiennent deux ou plus (cas $q > 0$), c'est nécessairement qu'une proportion b plus faible de l'ensemble des carnets en contient un exactement. Cette proportion b diffère de la proportion b_0 d'un facteur additif $-2q$ au moins. À cause de la contrainte $a + b + q = 1$, c'est finalement que a est plus grand, d'au moins un facteur additif q. (On considère ici a plutôt que b pour des raisons qui deviendront claires ci-dessous : essentiellement, on veut exhiber un minorant de a par dualité entre tests et intervalles de confiance.)

Le test ϕ est alors donné par $\phi = \phi'$. Comme H_0 implique H_0', la taille de ϕ est plus petite que celle de ϕ' ; en particulier, si ϕ' est de niveau α, alors c'est également le cas pour ϕ.

On note

$$T_j = \mathbb{I}_{\{C_j = 0\}} \ , \qquad \text{pour} \quad j = 1, \dots, 100,$$

et on procède comme au paragraphe 27.1.2, en considérant cette fois-ci la statistique de test $T = T_1 + \ldots + T_{100}$. Sous H_0, elle suit une loi $\text{Bin}(100,\, 1 - b_0)$ tandis que sous H_1, elle tend à prendre des valeurs plus grandes. L'intervalle de rejet est de la forme $[\![r_\alpha,\, 100]\!]$, où, pour une majoration $\alpha \in \,]0, 1[$ de l'erreur de première espèce, r_α est choisi le plus petit possible tel que

$$\mathbb{P}\big\{\text{Bin}(100,\, 1 - b_0) \geqslant r_\alpha\big\} \leqslant \alpha \ .$$

Par exemple, pour $\alpha = 5\ \%$, on a $r_{5\%} = 31$.

Or, les observations indiquent que la réalisation t de T vaut $t = 33$. On rejette donc H_0' puis H_0 au niveau $5\,\%$. On peut quantifier ce rejet en calculant la p–valeur. La valeur seuil p est égale à

$$p = \mathbb{P}\big\{\text{Bin}(100,\, 1 - b_0) \geqslant 33\big\} \approx 1.5\,\% \ ,$$

la p–valeur est donc de $1.5\,\%$. H_0 est contredite, mais pas trop gravement. On suspecte que la vraie proportion a de carnets ne contenant aucun unique lot significatif est un peu supérieure, mais pas dramatiquement supérieure, à $1 - b_0$.

En fait, en utilisant ici encore la dualité entre tests et intervalles du paragraphe 4.4, on montre que la réalisation d'un intervalle de confiance à 95 % sur a est $[0.252, 1]$. En effet, le test de

$$H_0' : a = a' \qquad \text{contre} \qquad H_1 : a > a'$$

fondé sur la statistique T conserve H_0 au vu des observations et au niveau 5 % si et seulement si

$$\mathbb{P}\big\{\mathrm{Bin}(100,\, a') \geqslant 33\big\} > 5\,\% \;;$$

avec l'aide des fonctions quantiles implémentées dans `Matlab`, on voit que cette inégalité est vérifiée pour les a' tels que $a' \geqslant 0.252 \approx 25\,\%$.

Minoration de q

Il n'est pas facile de minorer directement q par une valeur non triviale, i.e., strictement positive, car les observations ne comportent aucun carnet avec deux lots significatifs ou plus. On va utiliser une méthode indirecte utilisant des encadrements pour a et b, ce qui, combiné à un système de contraintes linéaires que l'on peut écrire sur a, b et q, suffira pour conclure.

On commence par majorer b : cela se fait, comme pour a et q, par dualité entre tests et intervalles. On omet les détails et indique simplement qu'au niveau 5 % l'intervalle est donné par $[0, 0.748]$, où cette valeur $0.748 \approx 75\,\%$ est obtenue comme la valeur seuil b' telle que

$$\mathbb{P}\big\{\mathrm{Bin}(100,\, b') \leqslant 67\big\} = 5\,\% \;.$$

On a donc dans un premier temps les trois encadrements suivants, chacun issu de la réalisation d'un intervalle de confiance de niveau 95 %,

$$\begin{cases} a & \geqslant & 25.2\,\% \;, \\ b & \leqslant & 74.8\,\% \;, \\ q & \leqslant & 3.0\,\% \;. \end{cases}$$

Le système formé par ces derniers est la réalisation d'une région de confiance de niveau au moins 85 % sur (a, b, q).

On veut écrire maintenant un système de contraintes linéaires sur a, b et q. À cet effet, on suppose qu'il n'existe pas de carnet contenant trois lots significatifs ou plus. On a déjà vu, au paragraphe 27.1.2 que la société de jeux émettait des contraintes techniques sur la répartition, puisque le mode de génération selon $\mathcal{V}_{\mathrm{unif}}$ avait été clairement rejeté.

On peut imaginer avec raison qu'elle le fait pour étaler les lots dans le plus grand nombre de carnets possibles et qu'à cause de ce souhait, elle élude également le cas de carnets contenant trois lots significatifs ou plus. Ainsi, on peut réécrire la loi ν comme

$$\nu = (a,\, b,\, q,\, 0,\, \ldots,\, 0) \;.$$

On aboutit alors au système de contraintes

$$\left\{\begin{array}{rcl} a,\, b,\, q & \geqslant & 0\ , \\ a+b+q & = & 1\ , \\ b+2q & = & b_0\ , \end{array}\right.$$

où les deux premières contraintes viennent de ce que ν est une probabilité, et la troisième assure qu'il y a bien le bon nombre de lots significatifs dans une tranche d'émission de 10 000 carnets.

Ce système est équivalent à

$$\left\{\begin{array}{rcl} a,\, b,\, q & \geqslant & 0 \\ q & = & (b_0-1)+a\ , \\ 2q & = & b_0-b\ . \end{array}\right.$$

En le combinant à la réalisation de la région de confiance et en réinjectant les valeurs numériques, on aboutit à l'encadrement de q suivant,

$$2.2\,\% \leqslant q \leqslant 3.0\,\%\ .$$

En fait, ici, on utilise la minoration de a, qui est plus efficace que la majoration de b pour exhiber le minorant sur q. De même, de $a = 1 - b_0 + q$ pour la majoration et de la minoration initiale issue de la région de confiance, on déduit

$$25.2\,\% \leqslant a \leqslant 26.0\,\%\ ;$$

tandis que de $b = b_0 - 2q$ ou $b = 1 - a - q$ et des encadrements précédents vient

$$71.0\,\% \leqslant b \leqslant 72.6\,\%\ .$$

On est ainsi parvenu à

$$\left\{\begin{array}{rcccl} a+b+q & = & 1\ , & & \\ 25.2\,\% & \leqslant & a & \leqslant & 26.0\,\%\ , \\ 71.0\,\% & \leqslant & b & \leqslant & 72.6\,\%\ , \\ 2.2\,\% & \leqslant & q & \leqslant & 3.0\,\%\ . \end{array}\right.$$

L'excellente nouvelle est qu'en particulier il existe une modélisation qui soit en accord avec les données ; en fait, il en existe toute une famille. Pour fixer les idées, et parce qu'il n'y a de toute façon une incertitude qu'à quelques dixièmes de pour cent, on supposera par la suite, pour étudier les espaces d'opportunités, que l'on a affaire à une génération aléatoire sous-jacente telle que

$$(a, b, q) = \big(25.6\,\%,\, 71.8\,\%, 2.6\,\%\big)\ . \tag{27.3}$$

Ce choix est motivé par le fait que la valeur retenue pour chacun des trois paramètres est au centre de l'intervalle exhibé plus haut sur sa vraie valeur.

Remarque au passage. Le résultat justifié ci-dessus peut sembler refléter la dictature éclairée du niveau de confiance 95 %, à ceci près bien sûr que la réalisation de la région finalement exhibée sur (a, b, q) est issue d'une région de confiance de niveau simplement au moins égal à 85 %. Cette région n'étant pas limitée à un point, on pourrait réduire un peu la valeur de α et obtenir encore des résultats raisonnables. Cela étant, il faut bien reconnaître ici que nous avons joué de chance avec les données pour obtenir un système final donnant une image aussi précise de ν. La pratique statistique est parfois plus rude et peut conduire à des images plus floues des phénomènes en jeu. Par ailleurs, il ne faut pas oublier qu'une fois sur vingt, on doit jouer de malchance avec une région de confiance à 95 %.

27.1.4 Étude de la redistribution des sommes mises en jeu

Les jeux à gratter sont des jeux dits par redistribution : les sommes misées sont rassemblées, une fraction d'entre elles revient aux joueurs sous la forme des lots, et la société de jeux empoche le reste. Dans le cas du jeu considéré ici, le tableau 19.1 indique que 1 020 500 € sont reversés pour un montant total joué (correspondant au prix des tickets) de $500\,000 \times 3 = 1\,500\,000$ €. Cela forme un taux de redistribution de 68 % (et donc un bénéfice brut de 32 %). Du point de vue des joueurs, cela signifie que l'espérance de gain, pour un ticket, soit 3 € misés, est de 2.04 €, d'où un solde moyen égal à −0.96 €. (Un mathématicien commenterait froidement : le prix du frisson du jeu avoisine donc 1 €.)

Mais regardons plus précisément le ressenti des joueurs : l'analyse précédente tient trop compte d'événements rares, comme le fait de gagner le gros lot de 40 000 €, ce qui arrive avec une probabilité très faible, mais change malgré tout significativement l'espérance du gain. Nous expliquons ici pourquoi et comment un joueur typique ne doit pas compter sur des gains supérieurs ou égaux à 500 €. Prenons-en un qui achète par exemple deux tickets par semaine.

On commence par déterminer la loi du gain G d'un ticket pris au hasard. On va appliquer des arguments similaires à ceux du paragraphe 27.1.2. Un ticket acheté chez un buraliste fixé est tiré au hasard dans la tranche d'émission (enfin, c'est le cas pour peu que le buraliste ne mette pas en œuvre les stratégies de joueur informé dont il sera question au paragraphe 27.1.5). La loi du gain de ce ticket est donc une loi multinomiale $\mathcal{M}(1, T)$ de paramètres 1 et T, où

$$T = \big(T_0,\, T_3,\, T_4,\, T_6,\, \ldots,\, T_{40\,000}\big)$$

est un vecteur de probabilité donné par le tableau 19.1 : par exemple,

$$\mathbb{P}\{G = 0\} = 1 - \frac{117\,197}{500\,000} \overset{\text{not.}}{=} T_0\ , \qquad \mathbb{P}\{G = 3\} = \frac{47\,500}{500\,000} \overset{\text{not.}}{=} T_3\ ,$$

$$\mathbb{P}\{G = 4\} = \frac{25\,000}{500\,000} \overset{\text{not.}}{=} T_4\ , \qquad \ldots\ , \qquad \mathbb{P}\{G = 40\,000\} = \frac{1}{500\,000} \overset{\text{not.}}{=} T_{40\,000}\ .$$

Gain max.	sur 1 an	sur 10 ans	Gain max.	sur 1 an	sur 10 ans
0	0.00	0.00	100	13.28	15.63
3	0.00	0.00	200	14.72	75.64
4	0.01	0.00	500	0.36	3.47
6	3.34	0.00	1 000	0.10	0.99
10	17.85	0.00	10 000	0.04	0.40
20	26.41	0.06	20 000	0.02	0.20
50	23.86	3.41	40 000	0.02	0.20

Tableau 27.3. Table donnant les lois P_{100} et $P_{1\,000}$, i.e., les fréquences (en %) qu'un joueur grattant deux tickets par semaine gagne au moins une fois une certaine somme. Valeurs arrondies à 0.01 % près.

On note G_j le gain du j–ième ticket acheté par le joueur. Comme le joueur prélève peu de tickets dans une tranche, puisqu'il n'achète que deux tickets par semaine, les gains G_1, $G_2, \dots$ qu'il obtient sont indépendants et identiquement distribués selon $\mathcal{M}(1, T)$. On utilise ici les arguments désormais habituels d'approximation de tirage *a priori* sans remise par un tirage avec remise. Ils sont en fait valables pour la plupart des gains, sauf les plus gros d'entre eux pour lesquels les T_j sont très petits. En revenant à la preuve du Lemme 27.3, on note que pour eux l'approximation n'est pas satisfaisante. De manière équivalente et plus intuitive, il est clair que si $G_1 = 40\,000$ par exemple, alors G_2 et quelques tickets suivants ne pourront prendre cette valeur, puisque le gros lot est unique. Il faut attendre une nouvelle émission pour que la valeur 40 000 puisse être à nouveau prise. Ces très gros gains arrivant rarement (une assertion que nous quantifions plus bas), nous négligeons leur impact dans la modélisation et menons donc l'approximation citée.

En particulier, sous cette modélisation d'échantillon indépendant et identiquement distribué, la loi du gain le plus élevé parmi les n premiers tickets a pour fonction de répartition

$$k \longmapsto F_n(k) \overset{\text{not.}}{=} \mathbb{P}\left\{\max_{t=1,\dots,n} G_t \leqslant k\right\} = \left(\mathbb{P}\{G_1 \leqslant k\}\right)^n .$$

La loi s'en déduit : $P_n(0) = F_n(0)$ et pour tout $k = 3,\ 4,\ 6,\ \dots,\ 40\,000$,

$$P_n(k) \overset{\text{not.}}{=} \mathbb{P}\left\{\max_{t=1,\dots,n} G_t = k\right\} = F_n(k) - F_n(k-1)$$

(la formule étant valable parce que les gains sont séparés d'au moins 1 €).

On calcule cette loi pour $n = 100$ (un an) et $n = 1\,000$ (dix ans) dans le tableau 27.3. On la représente à la figure 27.1, *via* sa fonction de survie

$$k \longmapsto R_n(k) = \mathbb{P}\left\{\max_{t=1,\dots,n} G_t \geqslant k\right\} = 1 - F_n(k-1)$$

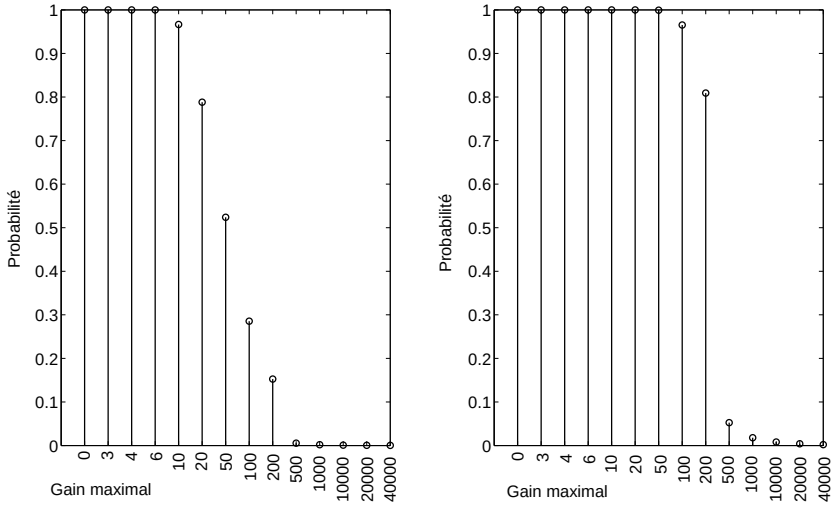

Figure 27.1. Fonction de survie du montant du gain maximal, pour un joueur grattant deux tickets par semaine pendant un an (à gauche), pendant dix ans (à droite).

(où pour la dernière égalité, on a de nouveau utilisé que les gains sont séparés d'au moins 1 €). On remarque qu'en un an, le joueur a toutes les chances (probabilité de 78.8 %) de voir passer un gain d'au moins 20 €, mais qu'il n'a presque aucune chance (probabilité 0.54 %) de gagner 500 € ou plus. Sur une période de dix ans, le gain maximal typique (probabilité 75.64 %) est de 200 €, mais le joueur a toujours peu de chances (probabilité 5.26 %) de gagner 500 € ou plus.

On retient donc que les gains d'un joueur typique varient entre 0 et 200 €. En particulier, à destination du paragraphe 27.1.5, on note qu'il ne faut pas s'attendre à obtenir des gains supérieurs ou égaux à 500 € lorsque l'on mènera notre analyse d'existence d'espaces d'opportunité.

Un autre argument montre qu'il est intéressant de se restreindre à ces gains et de ne pas considérer ceux qui sont supérieurs. Dans la suite, au paragraphe 27.1.5, nous effectuerons un raisonnement en espérance. Pour qu'il reflète le comportement réel, il faut utiliser un argument indiquant que les moyennes empiriques d'échantillons sont proches de l'espérance de la loi les ayant générés. Cette proximité est d'autant plus grande que la variance est petite, comme l'indiquent les intervalles de confiance du chapitre 3.

Or, la loi $\mathcal{M}(1,T)$ du gain d'un ticket admet pour espérance $\mu \approx 2.04$ tandis que son écart-type est $\sigma = 67.2$; cette valeur est très grande par rapport à l'espérance, il y a un facteur multiplicatif de plus de 30. Mais au vu des gains typiques, il semble plus intéressant de considérer la loi du gain d'un ticket conditionnellement au fait que

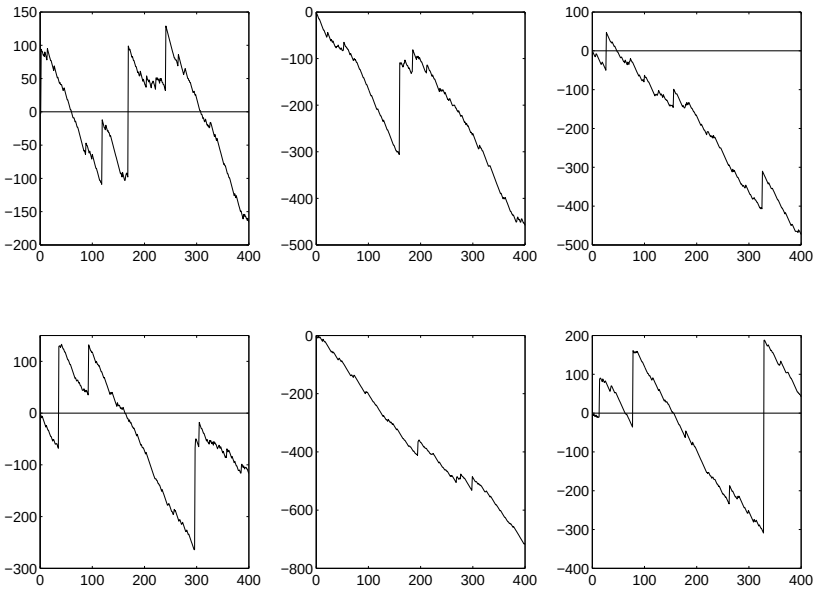

Figure 27.2. Soldes mobiles (en ordonnée) des gains et mises de joueurs individuels, en fonction du temps (en abscisse) : six réalisations aléatoires.

ce dernier soit inférieur ou égal à 200 €. Cette loi conditionnelle admet $\mu' = 1.85$ et $\sigma' = 9.81$ comme espérance et écart-type. Cette dernière n'est en particulier plus que 5 fois plus grande que l'espérance, ce qui est beaucoup plus satisfaisant.

En résumé, un joueur typique perdra donc plutôt $3 - \mu' = 1.15$ € par ticket ; un joueur atypique gagnera un gros lot et aura éventuellement un solde total positif pendant un certain temps. Les courbes des soldes mobiles de la figure 27.2 l'illustrent : un seul gain supérieur ou égal à 500 € apparaît, pour un total de 2 400 tickets grattés, et encore a-t-il fallu relancer les simulations plusieurs fois ! À l'opposé, plusieurs tickets étaient porteurs d'un lot de 200 €.

Pour s'amuser et dans un but pédagogique envers les joueurs invétérés, on peut alors exhiber un intervalle de confiance sur le montant perdu par un joueur typique (i.e., un joueur se trouvant parmi les 94.74 % de joueurs ne gagnant jamais 500 € ou plus en $n = 1\,000$ grattages). Le théorème de la limite centrale assure que

$$\sqrt{n}\left(\frac{G_1\,\mathbb{I}_{\{G_1\leqslant 200\}} + \ldots + G_n\,\mathbb{I}_{\{G_n\leqslant 200\}}}{n} - \mu''\right) \rightsquigarrow \mathcal{N}\big(0,\,(\sigma'')^2\big)\ ,$$

où μ'' et σ'' sont l'espérance et l'écart-type de $G\,\mathbb{I}_{\{G\leqslant 200\}}$ pour $G \sim \mathcal{M}(1, T)$; un

calcul simple montre que $\mu' \approx \mu'' \approx 1.85$ et $\sigma' \approx \sigma''$ (il y a égalité à 2×10^{-4} près, ce qui est moins que la précision avec laquelle on avait donné μ' et σ'). Ainsi, avec une probabilité pas trop éloignée de 95 % (l'écart étant induit par le caractère asymptotique de la convergence en loi ci-dessus),

$$G_1 \mathbb{I}_{\{G_1 \leqslant 200\}} + \ldots + G_n \mathbb{I}_{\{G_n \leqslant 200\}} \leqslant n\mu'' + z_{95\,\%}\, \sigma'' \sqrt{n}$$

où $z_{95\,\%} \approx 1.65$ est le quantile à 95 % de la loi normale standard.

Par union d'événements, avec probabilité proche de 90 %, pour n modérément grand (de l'ordre d'une ou plusieurs centaines, mais plus petit que 1 000), on a la majoration pour les soldes de gains eux-mêmes,

$$G_1 + \ldots + G_n \leqslant n(\mu'' - 3) + z_{95\,\%}\, \sigma'' \sqrt{n}\ .$$

Pour $n \geqslant 200$, cette borne est négative, i.e., les déviations non improbables envers la perte moyenne $-1.15\, n$ ne permettent même pas d'espérer un gain total positif. Autrement dit, environ 90 % de la population perd à jouer régulièrement pendant deux ans, certains perdant beaucoup d'argent, en moyenne ici, 330 €. Seuls 10 % au mieux réussissent à maintenir, pour un temps, un solde positif entre leurs mises et leurs gains. Mais ce dernier est voué, par loi des grands nombres, à devenir négatif.

27.1.5 Sur l'existence et la pertinence des stratégies de joueur informé

On étudie ici si l'on peut utiliser l'inférence (27.3) pour construire une stratégie qui aurait un solde positif en espérance. On rappelle que l'espérance du solde de la stratégie qui consiste à acheter n tickets au hasard est de $-0.96\, n$ (ou même, plutôt typiquement de l'ordre de $-1.15\, n$), ainsi qu'on l'a vu au paragraphe 27.1.4 On suppose à cet effet deux choses.

Premièrement, même s'il existe des contraintes techniques fortes qui réduisent l'aléa et entraînent qu'un carnet ne peut contenir qu'au plus deux lots significatifs, aucune manipulation n'est effectuée sur les lots mineurs, qui sont répartis dans la tranche d'émission selon un hasard uniforme. De plus, la société de jeu tâche de conserver autant d'aléa que possible et place les lots, et en particulier, le ou les lot(s) significatif(s), au hasard dans le carnet. Autrement dit, une fois l'ensemble des lots constitué pour un carnet, la place de ces derniers à l'intérieur du carnet est donnée par un arrangement aléatoire.

Deuxièmement, le buraliste retenu pour mettre en œuvre notre stratégie, à réception d'un carnet de 50 tickets, ne le découpe pas pour mettre les tickets individuels sur un présentoir, mais accroche le carnet avec une pince à linge et le débite au fur et à mesure des ventes. En outre, il suit le résultat des gains de ses clients. Ainsi, à tout moment, lorsqu'il reste x tickets dans le carnet, il peut déterminer s'il y avait eu un lot significatif dans les $50 - x$ tickets déjà vendus.

La stratégie proposée est de la forme : s'il reste x tickets dans un carnet dans lequel aucun lot significatif n'est encore tombé, alors on achète ces x tickets. Nous allons

tout d'abord prouver qu'il existe des valeurs de x (celles plus petites que 9) telles que l'espérance du solde de cette stratégie est strictement positif. Nous montrerons ensuite qu'il arrive suffisamment fréquemment qu'on soit dans les conditions de mise en œuvre de la stratégie.

Espérance du solde de gain de la stratégie en fonction de x

On dispose de trois types de carnets, selon qu'ils contiennent en leur sein zéro, un, ou deux lots significatifs. À un carnet c donné, on associe alors son nombre de lots significatifs $N(c) \in \{0,1,2\}$. On a vu en (27.3) qu'un carnet C tiré au hasard a un type $N(C)$ distribué selon (a,b,q) :

$$\mathbb{P}\big\{N(C)=k\big\} = \begin{cases} a & \text{si } k=0,\\ b & \text{si } k=1,\\ q & \text{si } k=2. \end{cases}$$

On note qu'ici on ne fait plus l'identification entre C et $N(C)$ qu'on faisait au début du corrigé, par exemple au paragraphe 27.1.1.

On note E_x l'événement selon lequel aucun des $50-x$ premiers tickets d'un carnet C pris au hasard n'a été porteur d'un lot significatif. C'est une observation que le buraliste peut effectuer. On rappelle que l'emplacement du ou des lots significatifs, quand ils existent dans un carnet, est aléatoire. Étant donné un carnet avec un seul lot significatif, la probabilité que celui-ci soit dans les x derniers tickets est de $x/50$. De même, si le carnet contient deux lots significatifs, la probabilité que ces deux lots se trouvent dans les x derniers tickets est $d_1 = 0$ si $x = 1$ et est donnée par le rapport

$$d_x \overset{\text{not.}}{=} \binom{x}{2} \Big/ \binom{50}{2}$$

pour $x \geqslant 2$. Cela entraîne que conditionnellement à $N(C)=1$, l'événement E_x a une probabilité $x/50$, tandis que conditionnellement à $N(C)=2$, cette probabilité est d_x. Enfin, il est clair que lorsque $N(C)=0$, l'événement E_x est certain. Il vient alors

$$r_x \overset{\text{not.}}{=} \mathbb{P}(E_x) = \sum_{k\in\{0,1,2\}} \mathbb{P}\big\{N(C)=k\big\}\, \mathbb{P}\Big(E_x \,\Big|\, \big\{N(C)=k\big\}\Big) = a + b\,\frac{x}{50} + q\,d_x\ .$$

On détermine alors la loi conditionnelle de C sachant E_x (il s'agit d'une loi dite *a posteriori*). Par la formule de Bayes et en réinjectant les valeurs précédentes, on a

$$\mathbb{P}\big\{N(C)=k \,|\, E_x\big\} = \frac{\mathbb{P}\big\{N(C)=k\big\}\, \mathbb{P}\Big(E_x \,\Big|\, \big\{N(C)=k\big\}\Big)}{\mathbb{P}(E_x)}\ ,$$

soit

$$\mathbb{P}\big\{N(C)=k \,|\, E_x\big\} = \begin{cases} a/r_x & \text{si } k=0,\\ (b/r_x)\,(x/50) & \text{si } k=1,\\ (q/r_x)\,d_x & \text{si } k=2. \end{cases}$$

On calcule maintenant, dans chaque cas, l'espérance du gain typique. Avec les notations du paragraphe 27.1.4, et en considérant toujours une variable aléatoire $G \sim \mathcal{M}(1, T)$ donnant le gain d'un ticket pris au hasard, on note respectivement

$$m = \mathbb{E}\big[G \,|\, G \leqslant 10\big] \approx 1.02 \qquad \text{et} \qquad M = \mathbb{E}\big[G \,|\, G \in \{20, 50, 100, 200\}\big] \approx 55.54$$

l'espérance du gain conditionnellement au fait que le ticket gagne au mieux un lot mineur ou est perdant et à celui que le ticket porte un lot significatif typique. On élimine donc ici également les gains atypiques, i.e., ceux supérieurs ou égaux à 500 €: on les a caractérisés au paragraphe 27.1.4 comme suffisamment rares pour que même en jouant régulièrement, la probabilité qu'on les rencontre soit extrêmement faible.

Vu les hypothèses effectuées sur la fabrication des carnets, lorsque E_x est réalisé sur un carnet C aléatoire, l'espérance des gains portés par les x tickets restants est, conditionnellement à $N(C)$ et au fait qu'il n'y a pas de gains atypiques, de

$$\big(x - N(C)\big)m + N(C)\, M$$

pour $x \geqslant 1$ (on rappelle que $N(C) = 2$ est incompatible avec E_1).

En combinant avec la loi *a posteriori*, il vient que l'espérance des gains d'un carnet pris au hasard, conditionnellement à E_x avec $x \geqslant 2$ (et toujours au fait qu'il n'y a pas de gains atypiques), vaut

$$e_x = \frac{1}{r_x}\Big(axm + b(x/50)\big(M + (x-1)m\big) + qd_x\big(2M + (x-2)m\big)\Big)\ .$$

Cette formule est valable également pour $x = 1$, puisque l'on a défini $d_1 = 0$.

Il faut bien entendu s'intéresser en pratique au solde de jeu, c'est-à-dire aux gains déduction faite des mises, à savoir : $s_x = e_x - 3x$. La figure 27.3 donne le tracé de s_x comme fonction de x. On y voit, sans surprise, et conformément aux résultats du paragraphe 27.1.4, que la perte moyenne lorsque l'on achète 50 tickets vaut (aux erreurs d'arrondis près) $50 \times 1.15 = 57.50$ €.

La valeur de s_x est maximale pour $x = 4$ et vaut $s_4 = 2.10$ € (aux arrondis près).

Pertinence de la stratégie précédente

Le solde admet une espérance strictement positive : chaque fois que E_4 survient, on gagne donc en moyenne 2.10 €. La question est de déterminer la fréquence de survenance de E_4. Si c'est une fois l'an, l'investissement en temps et en énergie du buraliste (qui, rappelons-le, doit suivre les gains ou absences de gains des joueurs) n'est par exemple pas rentable. Le calcul a déjà été effectué : cette fréquence vaut

$$\mathbb{P}(E_4) = r_4 \approx 31.7\,\%\ .$$

En termes simples : plus de 30 % des carnets conduisent à la réalisation de E_4. Or, on peut supposer que le buraliste vend plusieurs carnets par semaine, disons, vingt. Dans ce cas, la stratégie précédente gagne en moyenne

$$20\, r_4\, e_4 \approx 20 \times 0.317 \times 2.10 = 13.31\ €$$

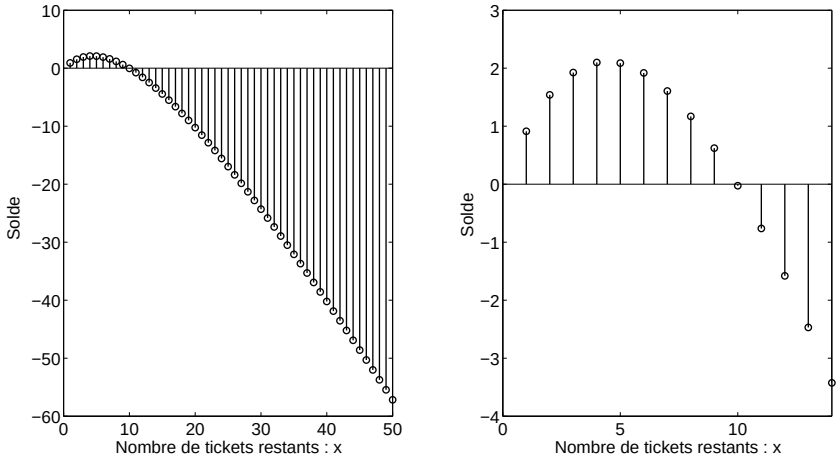

Figure 27.3. Espérance du solde de jeu (en ordonnée) pour un détaillant fictif mettant en œuvre la stratégie de joueur informé, en fonction (en abscisse) du nombre de tickets restant dans un carnet dans lequel aucun gain significatif n'est sorti.

x	1	2	3	4	5	6	7	8	9	10 et plus
$r_x\, s_x$	0.25	0.44	0.58	0.66	0.69	0.66	0.57	0.43	0.24	$\leqslant 0$

Tableau 27.4. Valeurs de la quantité $r_x\, s_x$ en fonction de x (présentées avec deux chiffres significatifs) ; pour $x \geqslant 10$, les valeurs sont négatives.

par semaine. (Le fait que la stratégie de joueur informé puisse être appliquée régulièrement justifie, par la loi des grands nombres et le théorème de la limite centrale, que le solde moyen des gains soit proche de s_4 avec grande probabilité.)

En réalité, on peut faire un tout petit peu mieux. La meilleure valeur de x a été déterminée uniquement pour maximiser l'espérance de gain lors d'une réalisation de la stratégie, alors que le buraliste est amené à la mettre en œuvre plusieurs fois par semaine. Il s'agit en fait de maximiser la quantité $r_x\, e_x$, qui tient compte non seulement des gains réalisés à chaque fois, mais aussi du nombre de fois où on les réalise. Le tableau 27.4 reporte les valeurs prises par $r_x\, e_x$ en fonction de x. On en déduit que la valeur $x = 5$ obtient des performances légèrement meilleures en situation de répétitions que la valeur $x = 4$ calibrée pour une mise en œuvre unitaire de la stratégie.

Dans tout ce paragraphe, nous avons écarté les gains atypiques. En fait, il se trouve que le buraliste mettant en œuvre la stratégie pour $x = 4$ ou $x = 5$ achète de nombreux

tickets par semaine : en moyenne, $x\, r_x$ fois le nombre de carnets vendus. Lorsque ce dernier vaut 20 comme supposé précédemment et que $x = 5$, cela consiste en plus de 30 tickets par semaine. En un an, il effectue un peu plus d'achats que le joueur invétéré du paragraphe 27.1.4 en dix ans. L'hypothèse que les gains typiques sont tous inférieurs ou égaux à 200 serait peut-être à revoir ; mais nous préférons la conserver afin de montrer que même sous cette hypothèse pessimiste, il y a bien opportunité de gain.

27.1.6 Conclusions ?

La conclusion est laissée au lecteur : d'après vous, l'investissement du buraliste pour gagner ces 13 € supplémentaires par semaine en vaut-il la peine, ou le temps et l'énergie consacrés au suivi trouveraient-ils une meilleure rémunération dans une autre activité accessoire ?

Par ailleurs, si une telle affaire était portée devant un tribunal et que vous y soyez cité comme expert, que diriez-vous ? Le hasard dans la répartition des lots, à défaut d'être total, vous semble-t-il suffisant ? Si l'affaire avait lieu en France, on porterait notamment à la défense de la société de jeux que selon un décret de 2002, il n'est pas demandé une répartition des lots selon un hasard total, mais un « hasard prépondérant » – une notion floue qu'il nous est délicate de quantifier mathématiquement. On peut sans doute dire, par exemple, que le résultat d'un tirage selon une loi de Bernoulli Ber(0.49) est causé essentiellement par le hasard, mais que ce n'est pas le cas pour une réalisation de Ber(0.80). On pourrait relier la place prépondérante du hasard au fait que les déviations de part et d'autre de la valeur moyenne rendent difficile la prédiction des réalisations. Dans ce cas, la quantité-clé serait le rapport entre cette espérance et l'écart-type de la distribution, c'est elle qui mesurerait la prépondérance ou non du hasard.

27.2 Questions possibles, conseils aux étudiants et bibliographie

27.2.1 Questions éventuelles d'un enseignant

S'agissant ici d'effectuer des tests de conformité sur les paramètres de lois multinomiales, un enseignant ne pourrait réprimer une question sur les tests du χ^2 ; comme par exemple, rappeler la forme générale de la statistique du χ^2 et l'énoncé de sa convergence en loi vers une loi du χ^2 (de degrés de liberté à préciser). Il reviendra également sur tous les points mal maîtrisés dans la méthodologie des tests.

On peut également s'attendre à une discussion sur la forme du tirage stochastique sous-jacent des lots, qui risque de tourner un peu au débat d'opinions, dans la mesure où l'on n'a que peu de données précises et où l'on est obligé, à plusieurs endroits du texte, d'imaginer comment procède la société de jeux et d'émettre des hypothèses supplémentaires sur la bonne génération aléatoire de la répartition des lots. Dans tous les cas, il est important d'argumenter son opinion et de rester honnête dans l'échange

en reconnaissant une contradiction soulevée par un enseignant et en essayant, en retour, de mettre en difficulté, si cela est possible, son idée de la modélisation.

Enfin, sur un tout autre plan, à propos de jeux de hasard du monde réel, on pourrait demander s'il existe des stratégies de joueur informé au Loto par exemple, qui est un célèbre jeu organisé par la Française des Jeux. Il est clair que compte tenu de la méthode de tirage, on ne peut pas augmenter sa probabilité de gagner. En revanche, comme il s'agit d'un jeu par répartition et que les gains sont divisés entre les vainqueurs, il vaut mieux être seul ou peu nombreux à gagner le gros lot. Pour assurer cela, il faut jouer des nombres peu retenus par les autres joueurs (par exemple les nombres plus grands que 32, qui ne correspondent ni à des jours ni à des mois de naissance). Ainsi, on augmente l'espérance de son gain.

Pour mettre cette stratégie en œuvre, il faudrait cependant disposer d'une étude statistique donnant les répartitions des numéros joués. Or, la Française des Jeux ne communique pas ces données. En revanche, elle procure les résultats des tirages passés et tous les tests statistiques montrent que les résultats obtenus semaines après semaines sont bien totalement indépendants et uniformément répartis.

27.2.2 Conseils aux étudiants

Le texte paraît inoffensif au premier abord, mais il teste l'étendue des connaissances dans la méthodologie des tests. Tous les étudiants raisonnablement à l'aise avec le cours de statistique doivent pouvoir rejeter l'hypothèse de répartition totalement uniforme.

On rappelle que lorsque l'on rejette une hypothèse H_0 aussi franchement que celle de répartition uniforme ici, la forme de H_1 importe peu, et c'est pourquoi nous avons parfois pris H_1 fantôme. La plupart des tests suivants (de conformité) ont été également pris unilatères : là, en revanche, la forme unilatère de H_1 est fort importante pour avoir une meilleure p–valeur (souvent, pour la diminuer d'un facteur 2 par rapport au cas bilatère) et il faut la justifier précisément au cas par cas.

Au passage, rappelons que la notion de p–valeur (voir chapitre 3) n'est pas au programme de l'agrégation ; elle est pourtant bien utile pour quantifier l'attachement à H_0, surtout que l'on obtient ici une p–valeur, lors du test de la répartition totalement aléatoire, inhabituellement petite. On pourra utilement, lors du calcul d'une p–valeur, faire un dessin pour montrer la forme de la zone de rejet (déterminée en fonction de l'hypothèse alternative H_1), la valeur observée de la statistique de test, et la p–valeur (l'aire située sous une certaine partie de la courbe de densité de la loi de la statistique).

Le point crucial du corrigé est la modélisation stochastique sur laquelle s'ouvre le texte ; il faut bien la détailler, et notamment expliquer précisément pourquoi et comment l'on approxime, à deux moments, des tirages sans remise par des tirages avec remise. On ne peut pas et on ne doit pas commencer de but en blanc en supposant que l'on a affaire à des échantillons d'observations indépendantes et identiquement distribuées : c'est là tout le sel de la modélisation, passer d'une situation concrète à

une situation mathématiquement agréable. Nous n'avons vu ce point que trop peu souvent suffisamment détaillé par nos étudiants.

27.2.3 Bibliographie

Ce texte est *inspiré* de faits réels, mais ne prétend en aucun cas les reconstituer. Au contraire, des modifications et simplifications notoires ont été considérées, pour rendre le propos plus exploitable dans le cadre d'un manuel de niveau master. Gilles Stoltz est reconnaissant aux sources qui l'avaient contacté pour lui présenter l'histoire réelle sous-jacente. Certaines idées de simulation proviennent d'un énoncé de travaux pratiques d'Alain Trouvé.

27.3 Commentaires sur les simulations `Matlab`

27.3.1 Architecture générale des programmes

Une fois n'est pas coutume, le texte utilise plutôt `Matlab` comme une calculatrice géante (de quantiles et de fonctions de répartition). On a groupé dans le fichier maître ci-dessous une partie seulement des calculs informatiques menés pour écrire le corrigé ; on a essentiellement repris ceux qui ont conduit aux tableaux et graphiques des paragraphes 27.1.4 et 27.1.5. Une seule fonction auxiliaire, `Ticket`, est utilisée, elle simule un échantillon de tickets distribués selon la loi multinomiale d'intérêt.

27.3.2 Commentaires sur le code `Matlab`

La boîte à outils `StixBox`, dont nous avons déjà parlé aux chapitres 9.6 et 22, nous est très utile ici ; elle permet notamment, pour les lois les plus communes, par exemple la loi binomiale, de simuler un échantillon (`rbinom`), de calculer la probabilité d'une valeur (`dbinom`) ou la densité en un point donné (lorsque la loi est continue), la valeur de la fonction de répartition (`pbinom`) ou de déterminer les quantiles (`qbinom`). Autres exemples, pour la loi hypergéométrique (pour $k = 2$ modalités seulement), on utilise respectivement `rhypg`, `dhypg`, `phypg` et `rhypg`. Pour voir la liste de toutes les fonctions implémentées dans la boîte à outils `StixBox`, qui, on le rappelle, est disponible à l'oral de l'agrégation, on pourra taper `help Contents` dans une fenêtre de commande `Matlab` (en vérifiant au préalable que le répertoire contenant les fichiers de `StixBox` est dans la liste des chemins disponibles). Ainsi, on a par exemple obtenu le tableau 27.1 en calculant un vecteur `B` selon

```
B = dbinom([0 1], 50, 7697/500000);
B = [B 1-sum(B)];
```

Quant au tableau 27.2, il procède de

```
pbinom([12 11 10 9 6 3 1 0], 100, B(3));
```

En particulier, on n'a pas besoin de déterminer les différentes probabilités à la main en calculant des coefficients binomiaux ; ceci étant, il est utile de savoir que ceux-ci sont

obtenus par la commande `nchoosek` (c'est comme cela que l'on appelle les coefficients binomiaux en anglais, cela réfère au nombre de choix possibles de k éléments dans un ensemble en contenant n).

D'autres remarques plus rapides sont que d'une part, au lieu de décrire explicitement notre matrice `T` de répartition dans le code, on aurait pu la lire à partir d'un fichier (par exemple, un fichier texte, avec deux colonnes séparées, de manière cohérente, par des virgules, espaces ou tabulations), en employant la commande `dlmread`. La commande inverse permettant d'écrire une matrice dans un fichier texte est `dlmwrite`.

Du point de vue des graphiques, `stem` semble ici plus adapté que `plot` et fournit des figures souvent plus jolies que `bar` ; cette dernière commande est vraiment à réserver aux cas où l'on est intéressé par l'aire des bâtons (comme c'est le cas pour les histogrammes estimant les fonctions de densité, au chapitre 24, et encore dans ce cas lui préfère-t-on `histo`). Les lignes

```
set(gca, 'XTick', [1:length(R)]);
set(gca, 'XTickLabel', [0; T(end:-1:1, 1)]);
```

sont très techniques et permettent de changer la légende des bâtons en abscisses ; ceux-ci sont par défaut indexés par 0, 1, 2, etc, et l'on veut plutôt remplacer cette numérotation par la valeur des lots, 0, 3, 4, 6, 10, etc. `gca` désigne l'objet figure. `XTick` et `XTickLabel` sont les propriétés de cet objet que l'on veut modifier, et le troisième argument de `set` donne les nouvelles valeurs. On voit apparaître là le côté orienté objet de `Matlab`, que l'on avait déjà évoqué au paragraphe 25.3.2.

27.3.3 Codes des simulations

Ils sont disponibles au chapitre 35, page 539.

CHAPITRE 28

Machines à sous (compléments)

Résumé. Ce qui suit complète le chapitre 20. On explique ici brièvement comment rendre non-asymptotiques les résultats de convergence qui reposaient sur la loi des grands nombres pour les martingales ; au passage, nous allons même exhiber des vitesses de convergence. Pour cela, nous allons recourir aux inégalités de Hoeffding–Azuma et de Bernstein pour les martingales. (Tout ceci n'a évidemment qu'une visée uniquement culturelle et est d'un niveau master 2.) On conclut en donnant les codes `Matlab` des simulations effectuées au chapitre 20.

28.1 Compléments mathématiques

28.1.1 Inégalité de Hoeffding–Azuma

Énoncé

Nous allons employer cette inégalité pour préciser (20.2). À cet effet, nous l'énonçons tout d'abord, en soulignant qu'elle forme une généralisation de l'inégalité de Hoeffding (présentée au Lemme 3.7).

Théorème 28.1 (Inégalité de Hoeffding–Azuma) *Soit $(Y_t)_{t\geqslant 1}$ une suite d'accroissements de martingale par rapport à la filtration $(\mathcal{F}_t)_{t\geqslant 1}$, tels que pour tout $t \geqslant 1$, il existe a_t, b_t avec $a_t \leqslant Y_t \leqslant b_t$ p.s. Alors, pour tout $\lambda > 0$,*

$$\mathbb{P}\left\{\sum_{t=1}^{n} Y_t \geqslant \lambda\right\} \leqslant \exp\left(-\frac{2\lambda^2}{\sum_{t=1}^{n}(b_t-a_t)^2}\right) .$$

En particulier, sous les hypothèses du théorème, si $a_t = a$ et $b_t = b$ pour tout $t \geqslant 1$, on a, pour tout $n \geqslant 1$ et avec probabilité au moins $1-\delta$,

$$\sum_{t=1}^{n} Y_t \leqslant (b-a)\sqrt{\frac{n}{2}\log\frac{1}{\delta}} .$$

Application

On reprend les notations de la preuve du Corollaire 20.3 et il s'agit de préciser l'énoncer $M_n = o(n)$ p.s. : on va montrer que

$$M_n = O\Big(\sqrt{n\log n}\Big) \qquad \text{p.s.}$$

Preuve On a $-1 \leqslant Y_t \leqslant 1$ p.s., de sorte que $M_n \leqslant \sqrt{2n\log(2/\delta)}$ avec probabilité au moins $1-\delta/2$. Symétriquement, en appliquant l'inégalité aux $-Y_t$, on a également $M_n \geqslant -\sqrt{2n\log(2/\delta)}$ avec probabilité au moins $1-\delta/2$. Au final, avec probabilité au moins $1-\delta$,

$$\frac{|M_n|}{\sqrt{2n\log\dfrac{2}{\delta}}} \leqslant 1 \ .$$

En choisissant $\delta = \delta_n = 1/n^2$, le lemme de Borel–Cantelli indique que

$$\mathbb{P}\left\{\limsup \frac{|M_n|}{\sqrt{2n\log\left(2n^2\right)}} > 1\right\} \leqslant \mathbb{P}\left(\limsup\left\{\frac{|M_n|}{\sqrt{2n\log\left(2n^2\right)}} > 1\right\}\right) = 0 \ .$$

Ainsi, on vient de montrer que

$$\limsup \ \frac{|M_n|}{\sqrt{2n\log\bigl(2n^2\bigr)}} \leqslant 1 \quad \text{p.s.},$$

ce qui était le résultat recherché. □

Cette vitesse, qui est essentiellement $\sqrt{n\log n}$, ne doit pas être surprenante : c'est, à facteur logarithmique près, celle du théorème de la limite centrale et celle de la loi du logarithme itéré. À vrai dire, une formulation dite maximale de l'inégalité de Hoeffding–Azuma permettrait de renforcer le résultat précédent en remplaçant la vitesse $\sqrt{n\log n}$ obtenue par $\sqrt{n\log\log n}$, qui est cette fois-ci exactement celle de la loi du logarithme itéré. Cette dernière est bien connue pour des sommes de variables aléatoires indépendantes et identiquement distribuées, voir par exemple [vdV98, Proposition 2.26], mais elle est également vraie pour les martingales.

28.1.2 Inégalité de Bernstein–Freedman pour les martingales

Énoncé

Nous allons l'employer pour préciser les vitesses de convergence des estimateurs $\widehat{\theta}_n^A$ et $\widehat{\theta}_n^B$ vers, respectivement, θ^A et θ^B. Étant donnée une filtration $(\mathcal{F}_t)$, on rappelle qu'on note $\mathbb{E}_t$ l'espérance conditionnelle par rapport à $\mathcal{F}_{t-1}$.

Théorème 28.2 (Inégalité de Bernstein–Freedman) *Soient $(Y_t)_{t\geqslant 1}$ une suite d'accroissements de martingale par rapport à la filtration $(\mathcal{F}_t)_{t\geqslant 1}$ et K une constante telles que $Y_t \leqslant K$ p.s. pour tout $t \geqslant 1$. On note $\operatorname{Var}_t Y_t$ la variance conditionnelle de Y_t,*

$$\operatorname{Var}_t Y_t = \mathbb{E}_t\left[Y_t^2\right] \ .$$

Alors, pour tout $n \geqslant 1$, avec probabilité au moins $1-\delta$,

$$\sum_{t=1}^{n} Y_t \leqslant \square \sqrt{\left(K^2 + \sum_{t=1}^{n} \operatorname{Var}_t Y_t\right) \log \frac{n}{\delta}} + \square\, K \log \frac{n}{\delta}$$

où les $\square$ désignent des constantes universelles que l'on n'a pas cru bon d'expliciter.

Notons que l'inégalité de Bernstein est généralement plus fine que celle de Hoeffding–Azuma, en tout cas, dès que

$$\sum_{t=1}^{n} \operatorname{Var}_t Y_t \ll \frac{n}{\log n} \qquad \text{p.s.}$$

Application

On reprend les notations de la preuve du Lemme 20.4 et il s'agit d'en préciser l'assertion $M_n^A = o\big(N_n^A\big)$ située à sa fin ; on va montrer que sous l'hypothèse $N_n^A \gg \log n$, on a

$$M_n^A = O\left(\sqrt{N_n^A \log n}\right) \qquad \text{p.s.}$$

L'hypothèse est vérifiée dès que (c_k) croît sous-exponentiellement, par exemple, polynomialement : on a vu en effet au paragraphe 20.1.7 que $N_n^A \geqslant c_A^{-1}(n) \geqslant \big(c^{-1}(n)-1\big)/2$.

> Remarque au passage. Avant de prouver ce raffinement par application de l'inégalité de Bernstein, on peut se demander pourquoi l'on n'appliquerait pas l'inégalité de Hoeffding–Azuma ; c'est parce que cette dernière donnerait, de la même manière qu'au paragraphe précédent, $M_n^A = O\big(\sqrt{n \log n}\big)$ p.s. Or, cela n'est pas assez précis, puisque l'on a $N_n^A \ll n$ dès que A n'est pas le bon bras et que la stratégie remplit l'objectif du paragraphe 20.1.4.

Preuve L'inégalité de Bernstein pour les martingales livre que pour tout n fixé, avec probabilité au moins $1-\delta$,

$$M_n^A \leqslant \square \sqrt{\big(1+N_n^A\big) \log \frac{n}{\delta}} + \square \log \frac{n}{\delta}$$

(où les $\square$ désignent des constantes universelles), et une inégalité symétrique sur $-M_n^A$. En appliquant le lemme de Borel–Cantelli avec $\delta = \delta_n = 1/n^2$, il vient

$$\limsup \frac{M_n^A}{\log n + \sqrt{N_n^A \log n}} \leqslant \square \qquad \text{p.s.}$$

où l'on a utilisé que N_n^A domine 1, et où $\square$ désigne toujours une constante universelle. Ainsi, $M_n^A = O\big(\log n + \sqrt{N_n^A \log n}\big)$ est $O\big(\sqrt{N_n^A \log n}\big) = o\big(N_n^A\big)$ dès que la condition $N_n^A \gg \log n$ est remplie. □

28.2 Codes des simulations

28.2.1 Fonctions auxiliaires : génération des paiements, moyenne d'un vecteur éventuellement vide

```
function P = Paiements(p, q, n)
%
% Entrées :
%   p est le paramètre de Bernoulli du premier bras,
%   q celui du second,
%   n est le nombre de tours de jeu
% Sortie :
%   Une matrice P à deux lignes et n colonnes,
%   composée de 0 et de 1
%

  P = [rand(1,n) < p;
       rand(1,n) < q];

end



function m = meanV(P)
%
% Entrées :
%   P est un vecteur de nombres réels
% Sortie :
%   La moyenne m des éléments de P si P est non vide,
%   m = 0 sinon
%
  if (sum(P) == 0)
    m = 0;
  else
    m = mean(P);
  end

end
```

28.2.2 Stratégie N, et stratégies dérivées EN et S

```
function [G, C] = Algo_N(P, C, G)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
%   Le vecteur C contient la suite des bras déjà tirés (index 1 ou 2)
%   en cas d'exploration préalable, C = [] sinon lors de l'appel
```

```
%   Le vecteur G contient la suite des gains déjà tirés
%   en cas d'exploration préalable, G = [] sinon lors de l'appel
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme naif de pure exploitation (N)
%   C, la suite des bras choisis
%

  n = size(P, 2);
  % Exploitation pure (à tous les tours de jeu après l'exploration
  % éventuelle incluse dans C)
  for j = (1+length(C)):n
    % On calcule les moyennes des bras et on les compare,
    % ce qui donne l'action c (aléatoire en cas d'égalité)
    mA = meanV(G(C == 1));
    mB = meanV(G(C == 2));
    c = 1 + ((mB > mA) + (rand < 1/2) * (mB == mA));
    C = [C c];
    G = [G P(c, j)];
  end

end
```

```
function [G, C] = Algo_EN(P)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme (EN) composé de 2 x 10 coups
%   d'exploration (non alternés), suivis par une procédure
%   du type algorithme naif de pure exploitation
%   C, la suite des bras choisis
%

  n = size(P, 2);
  G = [P(1, 1:10), P(2, 11:20)];
  C = [ones(1,10), ones(1,10)*2];
  [G, C] = Algo_N(P, C, G);

end
```

```
function [G, C] = Algo_S(P)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
%   (pour cet algo, il est bon qu'elle soit composée de 0 et 1 uniquement)
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme de type théorème du scrutin (S)
%   C, la suite des bras choisis
%

  % On convertit les paiements 0/1 en paiements -1/1,
  % on lance l'algorithme naïf de décision par exploitation
  % pure, et on re-convertit les paiements
  [G, C] = Algo_N(2*P-1, [], []);
  G = (G + 1)/2;

end
```

28.2.3 Stratégie excitée X

```
function [G, C] = Algo_X(P, EA, EB)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
%   EA, temps d'exploration de A (éléments entre 1 et n)
%   EB, temps d'exploration de B (éléments entre 1 et n)
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme avec excitation (X)
%   C, la suite des bras choisis
%

  n = size(P, 2);
  C = [];
  G = [];
  for j = 1:n
    if any(EA == j)
      c = 1;
    elseif any(EB == j)
      c = 2;
    else
      % On calcule les moyennes des bras et on les compare,
      % ce qui donne l'action c (aléatoire en cas d'égalité)
      mA = meanV(G(C == 1));
      mB = meanV(G(C == 2));
      c = 1 + ((mB > mA) + (rand < 1/2) * (mB == mA));
```

```
    end
    C = [C c];
    G = [G P(c, j)];
  end

end
```

28.2.4 Stratégies U et E

```
function [G, C] = Algo_UCB(P)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme naif de pure exploitation (N)
%   C, la suite des bras choisis
%

  n = size(P, 2);
  % On joue d'abord chaque bras, puis on exploite
  % en considérant des bornes supérieures d'intervalles de confiance
  C = [1 2];
  G = [P(1,1) P(2,2)];
  for j = 3:n
    nA = (C == 1);
    mA = mean(G(nA)) + sqrt(2*log(j-1)/length(nA));
    nB = (C == 2);
    mB = mean(G(nB)) + sqrt(2*log(j-1)/length(nB));
    c = 1 + ((mB > mA) + (rand < 1/2) * (mB == mA));
    C = [C c];
    G = [G P(c, j)];
  end

end
```

```
function [G, crit] = Algo_E(P, eps)
%
% Entrées :
%   P est la matrice des paiements, à deux lignes et n colonnes
%   eps > 0 est le paramètre d'approximation
% Sortie :
%   G, le vecteur des paiements obtenus à chaque tour
%   par l'algorithme de pure exploration (E)
```

```
%   crit, variable booléenne indiquant si l'exploration
%   a eu le temps de prendre fin
%

  n = size(P, 2);
  C = [];
  G = [];
  % Phase préalable d'exploitation pure
  % crit est une variable booléenne indiquant si l'exploration doit
  % être poursuivie
  crit = 1;
  % Pas de temps courant
  t = 1;
  while crit & (t <= n-1)
    C = [C 1 2];
    G = [G P(1,t) P(2,t+1)];
    t = t + 2;
    mA = meanV(G(C == 1));
    mB = meanV(G(C == 2));
    crit = (abs(mA - mB) <= sqrt((2/t)*log(pi^2*t^2/(3*eps))));
  end
  % On est ensuite en exploitation pure
  if not(crit)
    c = 1 + (mB > mA);
    G = [G P(c, t:n)];
  end

end
```

28.2.5 Fichier maître (script) pour le choix de la forme des instants d'excitation

```
m = 1000;
n = 300;

% Différentes suites d'instants d'excitation
% On joue le premier bras aux temps définis par
% les Ej et le second aux temps Ej+1
E1 = (1:sqrt(n)).^2;
E2 = [(1:4)*2-1 8+(1:sqrt(n)).^2];
E3 = 2.^(1:(log2(n)+1));
E4 = (1:n^(1/3)).^3;

% On refait m fois la simulation pour calculer
% une approximation de l'espérance du gain moyen
R = zeros(4, n);
for j = 1:m
  P = Paiements(0.45, 0.30, n);
```

```
  R(1,:) = R(1,:) + Algo_X(P, E1, E1+1);
  R(2,:) = R(2,:) + Algo_X(P, E2, E2+1);
  R(3,:) = R(3,:) + Algo_X(P, E3, E3+1);
  R(4,:) = R(4,:) + Algo_X(P, E4, E4+1);
end

% Suite des approximations des gains moyens en fonction du temps
Z = cumsum(R, 2)./repmat(m*(1:n), [4 1]);
% On élimine les premiers pas de temps et on
% commence le dessin au 50ème tour
d = 50;
plotstyle = {'--', '-', '-.', ':'};
clf; hold on;
for k = 1:4
  plot(d:n, Z(k, d:n), plotstyle{k}, 'LineWidth', 1.7);
end
legend('c_k = k^2', 'Expl. puis c_k = k^2', '2^k', 'k^3')
xlabel('Pas de temps')
ylabel('Gain moyen')
```

28.2.6 Fichier maître (script) pour le tracé de la loi de la statistique asymptotiquement normale

```
clf, clc, clear;

m = 10000;
n = 300;

G = [];
% On refait simplement m simulations de l'algorithme
% par excitation
for j = 1:m
  P = Paiements(0.45, 0.30, n);
  E = 2.^(1:(log2(n)+1));
  G = [G sum(Algo_X(P, E, E+1))];
end

% Et on construit alors l'histogramme de la statistique d'intérêt
G = (G - 0.45*n)/sqrt(n);
histo(G, 100, [], 1);
title(strcat([ 'Variance (obs. non seuillées) ', num2str(var(G),3), ...
               ' -- Variance (obs. seuillées) ', ...
               num2str(var(G(G >= -1.5)), 3) ]));
```

28.2.7 Fichier maître (script) pour la comparaison des six stratégies

```
clc, clear;
```

```
m = 1000; % Ou 1, ou 100, ou 10000, selon les simulations
n = 300;

% On initialise les variables
R = zeros(6, n);
prb = zeros(6, 1);
M = zeros(6, 1);

% On répète m fois la simulation : on tire une suite de couples
% de paiements, on lance les algorithmes et on évalue leur performance :
% suites des gains moyens, probabilité de jouer le bon bras au tour n,
% probabilité d'être le meilleur algorithme
for j = 1:m
  % Cf. les paramètres doivent être < 1/2, sinon le casino est perdant
  % On a fait varier les couples de paramètres : (0.45, 0.30) mais
  % aussi (0.45, 0.05) et (0.45, 0.40)
  P = Paiements(0.45, 0.05, n);
  [R1, C1] = Algo_N(P, [], []);
  [R2, C2] = Algo_EN(P);
  E = 2.^(1:(log2(n)+1));
  [R3, C3] = Algo_X(P, E, E+1);
  [R4, crit] = Algo_E(P, 0.10);
  [R5, C5] = Algo_S(P);
  [R6, C6] = Algo_UCB(P);
  % Pour le calcul de l'espérance des gains
  R = R + [R1; R2; R3; R4; R5; R6];
  % Pour le calcul de la probabilité, on prend celle
  % de jouer le bon bras au dernier tour (i.e., de ne pas
  % être dans un piège et/ou d'avoir convergé)
  prb = prb + [C1(end) == 1;
               C2(end) == 1;
               C3(end) == 1;
               1 - crit;
               C5(end) == 1;
               C6(end) == 1];
  % Détermination du meilleur algorithme
  % (on traite les cas d'égalité en mettant +1 à tous les ex-aequo !)
  r = [sum(R1); sum(R2); sum(R3); sum(R4); sum(R5); sum(R6)];
  M(r == max(r)) = M(r == max(r)) + 1;
end

% Tracés des gains moyens
Z = cumsum(R, 2)./repmat(m*(1:n), [6 1]);
% On ne représente pas les premiers pas
d = 21;
textalgo = {'N', 'EN', 'X', 'E', 'S', 'U'};
```

```
clf; hold on;
for k = 1:6
  plot(d:n, Z(k, d:n), 'b', 'LineWidth', 1.7);
  % Permet d'annoter les courbes à la main
  gtext(textalgo{k});
end
xlabel('Pas de temps')
ylabel('Gains moyens')
```

CHAPITRE 29

Estimation non-paramétrique pour le modèle de régression (compléments)

Résumé. Nous présentons ici les codes des simulations informatiques relatives au chapitre 21.

29.1 Codes des simulations

29.1.1 Construction des données artificielles

```
function Y = observations_regression(f, sigma)
%
% Entrées :
%   sigma, un réel > 0 donnant un écart-type
%   f, un vecteur de taille n représentant un un signal f échantillonné
%      aux points  t_k = k/n pour k = 1,...,n
% Sortie :
%   Y, un vecteur d'observations de taille n
%

  n = length(f);
  Y = f + sigma * randn(1,n);

end
```

29.1.2 Calcul des estimateurs

```
function [hat_theta, Phi] = FourierSurDesign(Y)
%
% Entrée :
%   Y, un vecteur d'observations de taille n
% Sorties :
%   hat_theta, un vecteur de taille n-1 donnant les estimées
%              des n-1 premiers coefficients de Fourier
%   Phi, une matrice de taille n-1 x n dont la j-ième ligne est
%        le vecteur des valeurs du j-ième élément de la base de Fourier
%        sur le design
%

  n = length(Y);
```

```
  t = (1:n)/n;     % design uniforme
  N = n-1;
  % Calcul des valeurs des vecteurs de la base sur les points du design
  Phi = ones(N, n);
  for j = 2:N
    % Note : rem(a,b) donne le reste de le division euclidienne de a par b
    %        de sorte que rem(j+1,2) est p.ex. une indicatrice du
    %        fait que j+1 est impair
    Phi(j, :) =  sqrt(2)*(cos(pi*j*t)*rem(j+1,2)+sin(pi*(j-1)*t)*rem(j,2));
  end;
  % Calcul du vecteur hat_theta des estimées des coefficients de Fourier
  for j = 1:N
    hat_theta(j) = mean(Y .* Phi(j, :));
  end;

end



function hat_f = estimation_regression(Y, N);
%
% Entrées :
%   Y, un vecteur d'observations
%   N, le nombre de coefficients de Fourier à estimer
% Sortie :
%   hat_f, un vecteur de taille n donnant la valeur de l'estimée de f
%          en les points du design
%

  n = length(Y);
  [hat_theta, Phi] = FourierSurDesign(Y);
  for j = 1:n
    hat_f(j) = hat_theta(1:N) * Phi(1:N, j);
  end

end



function N_mallows = mallows_regression(Y, sigma);
%
% Entrées :
%   Y, un vecteur d'observations de taille n
%   sigma, un niveau de bruit
% Sortie :
%   N_mallows, un nombre entier entre 1 et n-1 donné par le critère
```

```
%              de Mallows
%

  n = length(Y);
  [hat_theta, Phi] = FourierSurDesign(Y);
  CritMallows = 2*sigma^2/n * (1:(n-1)) - cumsum(hat_theta.^2);
  [C, N_mallows] = min(CritMallows);
  disp(['Valeur de N donnée par le critère de Mallows --> N = ', ...
        num2str(N_mallows)]);

end
```

29.1.3 Fonction principale de tracé et de comparaison des méthodes

```
function principal_regression(sigma, n, t, f, p)
%
% Entrées :
%   sigma, un réel donnant un niveau de bruit
%   n, un nombre d'observations,
%   t, le design
%   f, la fonction à estimer échantillonnée sur t et passée en argument
%      comme un vecteur de taille n
%   p, un nombre donnant le numéro du signal
% Sortie :
%   Aucune sortie numérique mais des tracés graphiques
%

  disp([char(13), 'Pour la fonction numéro ', num2str(p)]);
  % Calcul approché de la norme L2 du signal par méthode des rectangles
  energie = sqrt(1/n * sum(f.^2));
  disp(['La norme L2 du signal vaut environ : ', num2str(energie)]);
  % On simule le vecteur des observations
  Y = observations_regression(f, sigma);

  % Tracé du signal p et des observations associées
  figure(1); subplot(2,2,p);
  plot(t, f, 'k', t, Y, 'k--');
  title(['n = ', num2str(n), ' et \sigma = ', num2str(sigma)]);

  % Calcul des estimateurs pour toutes les valeurs N = 1, ... n-1
  % (on en a besoin pour déterminer N_opt)
  % Au passage, on calcule également leur erreur quadratique
  hat_f = zeros(n-1, n);
  e = zeros(1, n-1);
  for N = 1:(n-1)
    hat_f(N, :) = estimation_regression(Y, N);
    e(N) = sum( (f-hat_f(N,:)) .^2 ) /n;
```

```
  end;

  % Tracé de la courbe de l'erreur quadratique en fonction de N
  figure(2); subplot(2,2,p);
  plot([1:n-1], e, 'k');
  xlabel('dimensionnalité N'); ylabel('erreur');

  % Détermination de N_opt
  [e_min, N_opt] = min(e);
  disp(['Valeur optimale de N --> N_opt = ', num2str(N_opt)]);
  % Détermination de N_mallows
  N_mallows = mallows_regression(Y,sigma);
  % Autres valeurs de N
  N_floor = floor(n^(1/3));
  N_max = n-1;

  % Tracés des reconstructions associées aux différents N retenus
  N = [N_opt N_floor N_max N_mallows];
  Tt = {' N_{opt}', '\lfloor n^{1/3} \rfloor', 'n-1', 'N_{mallows}'};
  for k = 1:4
    tr_f = hat_f(N(k), :);
    figure(p+2); subplot(2,2,k);
    plot(t, f, 'k', t, tr_f, 'k--');
    title(['signal et estimateur avec N = ', Tt{k}, ' = ', num2str(N(k))]);
  end;
```

29.1.4 Programme maître de simulation (script)

```
clc; clf;

% Signal #1
clear;
p = 1; n = 100; sigma = 0.1;
t = (1:n)/n;
f = (t >= 0.2) .* (t <= 0.3);
principal_regression(sigma, n, t, f, p);

% Signal #2
clear;
p = 2; n = 100; sigma = 0.1;
t = (1:n)/n;
f = exp(-t) .* sin(4*pi*t);
principal_regression(sigma, n, t, f, p);

% Signal #3
clear;
p = 3; n = 250; sigma = 1;
```

```
t = (1:n)/n;
f = min(10, 1./abs(cos(4*pi*t)));
principal_regression(sigma, n, t, f, p);

% Signal #4
clear;
p = 4; n = 200; sigma = 3;
t = (1:n)/n;
f = 4*sin(4*pi*t) + 3*cos(6*pi*t) - 2*sin(6*pi*t);
principal_regression(sigma, n, t, f, p);
```

29.1.5 Traitement des données réelles

Fonction pour l'estimation et le tracé de l'estimateur de l'écart-type

```
function s = estimation_ec(Y);
%
% Entrée :
%   Y, un vecteur d'observations (de taille n)
% Sortie :
%   s, le vecteur des estimées de sigma (de taille n-1)
%

  n = length(Y);
  [hat_theta, Phi] = FourierSurDesign(Y);
  % On détermine la trajectoire de l'estimateur de l'écart-type
  P = cumsum(hat_theta([end:-1:1]).^2);
  s = sqrt(n * P ./ (1:(n-1)));

end
```

Fichier maître (script) pour les données réelles

```
% Etude des performances de la procédure associée au critère de Mallows
% sur le jeu de vraies données  *Sunspot_month*
clc; clear;

% Chargement des données
load Sunspot_month.mat;
Y = Sunspot_month;
% Tracé des observations
n = length(Y);
t = [1:n];
figure(1); subplot(2,2,1);
plot(t, Y, '-');
title('Observations');
```

```
% Tracé de la trajectoire de l'estimateur de l'écart-type
s = estimation_ec(Y);
figure(1); subplot(2,2,2);
plot([1:n-1], s, '-');
title('Tracé de la trajectoire de l''estimateur de \sigma');

% On va donner deux fois la main à l'utilisateur pour le choix
% délicat des estimées de l'écart-type
for j = 1:2
  sigma = input('Estimée de sigma ?  ');
  N_mallows = mallows_regression(Y, sigma);
  hat_f = estimation_regression(Y, N_mallows);
  % Tracé de l'estimateur hat_f
  figure(1); subplot(2,2,2+j);
  plot(t, hat_f, '-');
  title(['Estimateur pour \sigma = ', num2str(sigma), ...
         '  (N_{mallows} = ', num2str(N_mallows), ')']);
  % Tracés des résidus
  figure(2); subplot(2,3,j);
  plot(t, Y-hat_f);
  title(['Vraies données : \sigma = ', num2str(sigma)]);
  subplot(2,3,3+j);
  plot(hat_f,Y-hat_f,'.');
  title(['Vraies données : \sigma = ', num2str(sigma)]);
end;

% On renouvelle l'expérience avec un signal connu
Y = [];
n = 200;
sigma = 3;
t = (1:n)/n;
f = 4*sin(4*pi*t) + 3*cos(6*pi*t) - 2*sin(6*pi*t);
Y = observations_regression(f,sigma);
N_mallows = mallows_regression(Y, sigma);
hat_f = estimation_regression(Y, N_mallows);
% Tracé des résidus
figure(2); subplot(2,3,3);
plot(t, Y-hat_f);
title(['Données simulées : \sigma = ', num2str(sigma)]);
subplot(2,3,6);
plot(hat_f, Y-hat_f, '.');
title(['Données simulées : \sigma = ',num2str(sigma)]);
```

CHAPITRE 30

Inférence statistique pour des modèles censurés (compléments)

RÉSUMÉ. Nous présentons ici le code des simulations informatiques relatives au chapitre 22.

30.1 Codes des simulations

30.1.1 Fonctions auxiliaires

```
function [Y, d, hl] = donnees_censure(n)
%
% Entrée :
%   n est la taille d'échantillon désirée
% Sorties :
%   Y  est le vecteur des observations censurées
%   d  est un vecteur dont chaque coordonnée vaut 0 ou 1 selon
%      qu'il y a eu censure (Y_i = C_i) ou non (Y_i = T_i)
%   hl est le vecteur des estimateurs de lambda, le t-ième
%      élément du vecteur étant construit uniquement
%      avec les t premiers éléments de Y et d
%

  % Génération de n variables exponentielles de
  % paramètres 1 (durées de vie) puis 0.5 (censure)
  T = -log(rand(1,n));
  C = -2*log(rand(1,n));
  % Observations des données censurées uniquement
  Y = min(C,T);
  d = (Y == T);
  % On utilise ici le fait que presque sûrement, Y(1) est non nul,
  % pour ne pas avoir de problème de division par 0 dans la ligne suivante
  hl = cumsum(d)./cumsum(Y);

end



function [KM1, KM2] = Kaplan_Meier(Y, d)
```

```
%
% Entrée :
%    Y et d, vecteurs d'observations (de même longueur n)
% Sortie :
%         Estimateur de Kaplan-Meier, sous la forme suivante.
%         Etant une fonction en escalier, on donne
%    KM1, les instants de sauts (vecteur de taille n) et
%    KM2, les valeurs de l'estimateur sur les n+1 intervalles
%         ainsi définis (vecteur de taille n+1)
%

  n = length(Y);
  % Réordonnement des observation et détermination de KM1
  [KM1, I] = sort(Y);
  dord = d(I);
  % Calcul de KM2
  KM2 = ones(1,n+1);
  for i = 1:n
    KM2(i+1) = KM2(i)*(1 - dord(i)/(n+1-i));
  end;

end
```

30.1.2 Illustration de la normalité asymptotique de l'estimateur de Kaplan–Meier

```
% Illustration de la normalité asymptotique de l'estimateur de Kaplan-Meier
clf, clc, clear;

N = 1000;
figure(3);
for k = 1:2
  % Valeur de n
  n = 10*(k==1) + 1000*(k==2);
  for t = 1:2
    sigma = 2/3 * exp(-2*t) * (exp(3/2*t)-1);
    % Calcul d'un N-échantillon
    Knorm = zeros(1, N);
    for m = 1:N
      [Y, d] = donnees_censure(n);
      [KM1, KM2] = Kaplan_Meier(Y, d);
      J = max(find(KM1 <= t));
      Knorm(m) = sqrt(n) * (KM2(J+1) - exp(-t)) / sqrt(sigma);
    end;
    % Tracés de la distribution empirique associée à Knorm
    % et de la distribution limite sur l'intervalle [-3 3]
    subplot(2, 2, t+2*(k-1));
    hold on;
```

```
    stairs(sort(Knorm), [1:N]/N, 'k--');
    axis([-3 3 0 1]);
    fplot('pnorm', [-3 3], 'k');
    legend('KM','limite');
    title(['t = ', num2str(t), ' et n = ',num2str(n)]);
  end;
end;
```

30.1.3 Illustration des propriétés asymptotiques des $\widehat{\lambda}_n$

```
% Illustration 1. de la consistance et 2. de la normalité asymptotique
% de \hat\lambda_n vers \lambda, dans le cas du modèle avec lois
% exponentielles
clf, clc, clear;

% 1. Illustration de la convergence presque-sûre
n = 1000;
[Y, d, hl] = donnees_censure(n);
figure(1);
plot(10:n, hl(10:n), 'k', [10, n], [1, 1], 'k:');
legend('\lambda_n', '\lambda = 1');
title('Convergence ps de l''estimateur de \lambda');

% 2. Illustration de la normalité asymptotique
%    Pour les valeurs n = 10, n = 100, n = 1000, on calcule un
%    échantillon de taille N = 1000 de \hat\lambda_n, puis
%    les deux variables aléatoires d'intérêt
N = 1000;
figure(2);
for k = [1, 2, 3]          %n sera 10^k
  hlnorm = zeros(N, 2);
  for m = 1:N
    [Y, d, hl_all] = donnees_censure(10^k);
    hl = hl_all(end);
    hlnorm(m, 1) = sqrt(2/3*10^k) * (hl - 1);
    hlnorm(m, 2) = sqrt(sum(d)) * (hl - 1) / hl;
  end;
  % Tracé des distributions empiriques et de la distribution limite
  % sur l'intervalle [-3, 3]
  hlnorm = sort(hlnorm, 1);
  figure(2);
  subplot(3,1,k);
  hold on;
  stairs(hlnorm(:,1), [1:N]/N, 'k--');
  stairs(hlnorm(:,2), [1:N]/N, 'k-.');
  axis([-2.5 2.5 0 1]);
  fplot('pnorm', [-2.5 2.5], 'k');
```

```
  title(['n = ', num2str(10^k)]);
end;
legend('Empirique 1','Empirique 2','Limite');
```

30.1.4 Comparaison entre estimateurs de Kaplan–Meier et par plug-in

```
% Comparaison entre estimateurs de Kaplan-Meier et par plug-in
% 1.  Consistance de ces estimateurs
% 2.  Régions de confiance
%     2.1  Consistance de l'estimateur de Greenwood
%     2.2  Comparaison des régions de confiance
clc, clear, clf;

% 1. Illustration de la consistance
n = 200;
[Y, d, hl] = donnees_censure(n);
hl = hl(end);
[KM1, KM2] = Kaplan_Meier(Y, d);
figure(4);
stairs([0,KM1], [KM2], 'k--');
hold on;
fplot(@(t)exp(-hl*t), [0 KM1(end)], 'k-.')
fplot('exp(-t)', [0 KM1(end)], 'k');
legend('Kaplan-Meier', 'plug-in', 'limite');
title(['Consistance pour n = ', num2str(n)]);


% 2.1 Calcul de l'estimateur de Greenwood pour la variance asymptotique
%     (et représentation de sa convergence ps)
[Yord, I] = sort(Y);
dord = d(I);
% G contiendra les n+1 valeurs associées aux n+1 intervalles
% définis par les n points de KM1
G = zeros(1,n+1);
% Estimation des intégrales I(t)
for i = 1:n
  G(i+1) = G(i) + dord(i)/((n-i+1)*(n-i));
end;
G = n * KM2.^2 .* G;
figure(5);
stairs([0,KM1], G, 'k-.');
hold on;
fplot(@(t)(2/3*(exp(3/2*t)-1)*exp(-2*t)), [0 KM1(end)], 'k');
legend('Greenwood', 'Var W(t)')
title(['n = ', num2str(n)]);

% 2.2 Tracés
```

```
%     - de la vraie fonction de survie
%     - de la bande de confiance (niveau = 95%) associée à l'estimateur
%       plug-in
%     - de la collection des intervalles de confiance associés
%       à l'estimateur de Kaplan-Meier
figure(6);
hold on;
IKM = qnorm(.975)*sqrt(G/n);
stairs([0,KM1], KM2 - IKM ,'k--');
IPI = qnorm(.975)/sqrt(sum(d));
fplot(@(t)(exp(-hl*t) - IPI), [0 KM1(end)], 'k-.')
fplot('exp(-x)', [0 KM1(end)], 'k');
title(['Niveau 0.95 et n = ',num2str(n)]);
legend('Kaplan-Meier', 'plug-in', 'limite');
stairs([0,KM1], KM2 + IKM ,'k--');
fplot(@(t)(exp(-hl*t) + IPI), [0 KM1(end)], 'k-.')
```

30.1.5 Illustration sur un jeu de réelles données

```
clc, clear, clf;

% On exploite les données fournies par l'ouvrage  "Survival Analysis" de
% Miller qui donne les données de durée de vie partiellement censurées pour
% 23 patients atteints de leucémie aiguë myéloïde
D = [9 13 13 18 23 28 31 34 45 48 161 5 5 8 8 12 16 23 27 30 33 43 45;
     1  1  0  1  1  0  1  1  0  1   0 1 1 1 1  1  0  1  1  1  1  1  1];
% Tri des données
Y = D(1,:);
[Y,S] = sort(Y);
d = D(2,S);
% Tracé des données
hold on;
I = find(d == 0); J = find(d == 1);
stem(I, Y(I), 'filled');
stem(J, Y(J));
legend('Non censurées', 'Censurées');
xlabel('Patients'); ylabel('Durées de survie');

% 2. Calcul de l'estimateur de Greenwood pour la variance asymptotique
n = length(Y);
hl = sum(d)/sum(Y);
[KM1, KM2] = Kaplan_Meier(Y, d);
%    Cf. le script principal :
[Yord, I] = sort(Y);
dord = d(I);
G = zeros(1,n+1);
for i = 1:n
```

```
  G(i+1) = G(i) + dord(i)/((n-i+1)*(n-i));
end;
G = n * KM2.^2 .* G;
%
figure;
stairs([0,KM1], G, 'k-.');
legend('Greenwood')
title(['n = ', num2str(n)]);

% 2.2 Tracés
%     - de la bande de confiance (niveau = 95%) associée à l'estimateur
%       plug-in
%     - de la collection des intervalles de confiance associés
%       à l'estimateur de Kaplan-Meier
figure;  hold on;
title(['Niveau 0.95 et n = ',num2str(n)]);
stairs([0,KM1], [KM2], 'k');
IKM = qnorm(.975)*sqrt(G/n);
stairs([0,KM1], KM2 - IKM ,'k--');
stairs([0,KM1], KM2 + IKM ,'k--');
fplot(@(t)exp(-hl*t), [0 KM1(end)], 'k-.')
IPI = qnorm(.975)/sqrt(sum(d));
fplot(@(t)(exp(-hl*t) - IPI), [0 KM1(end)], 'k:')
fplot(@(t)(exp(-hl*t) + IPI), [0 KM1(end)], 'k:')
hold off;
```

CHAPITRE 31

Étude du nombre de renouvellements (compléments)

RÉSUMÉ. Nous présentons ici le code des simulations informatiques relatives au chapitre 23.

31.1 Codes des simulations

31.1.1 Fonctions auxiliaires

```
function W = weibull(a, b, m, n)
%
% Entrées :
%   a et b sont les paramètres de la loi de Weibull
%   m et n désignent la taille du tableau de variables à obtenir
% Sortie :
%   W, une matrice m x n dont les éléments sont iid selon la
%   loi de Weibull de paramètres a et b
%

  % On simule un tableau de taille (m,n) de variables uniformes sur [0,1]
  U = rand(m, n);
  % Et on en déduit W en appliquant l'inverse de la fonction de répartition
  W = b^(-1) * (-log(U)).^(1/a);

end
```

```
function [N1, N2, N3] = Nmax(t, m, v, alpha)
%
% Entrées :
%   t est un vecteur de points où l'on veut calculer les Nmax
%   m et v sont respectivement la moyenne et la variance
%   1-alpha est le niveau de confiance
% Sortie :
%   N1, N2 et N3 sont trois vecteurs de même taille que t
%
```

```
  N1 = t/m + qnorm(1-alpha)*sqrt(v)*sqrt(t)*m^(-3/2);
  M = t/m + v/(m^2);
  N21 = M + alpha^(-1/2) * sqrt(M.^2 + M);
  N22 = M + alpha^(-1/2) * sqrt(t/m*(4*v/(m^2)+3)+v/(m^2)*(2*v/(m^2)+1)-1);
  N2 = N21.*(t<m) + N22.*(t>=m);
  N3 = t/m + alpha^(-1/2)*sqrt(t/m);
end
```

31.1.2 Illustration des convergences presque sûres

```
% On illustre la convergence presque sûre de N(t) vers l'infini
clear, clc, clf;
a = 2; b = 1;
m = gamma(1+1/a)/b;
% On construit la suite de nos temps de saut
N = 2500;
W = weibull(a, b, 1, N);
S = [0 cumsum(W)];
% On représente les 20 premiers sauts du processus N
subplot(1, 2, 1);
stairs(S(1:21), 0:20)
title('Processus N');
% On représente t --> N(t)/t aux instants de saut
% et on trace la droite d'ordonnée 1/mu
subplot(1, 2, 2);
% On omet les premiers pas de temps, pour que ce soit plus clair
n = 50;
stairs(S((n+1):end), (n:N)./S((n+1):end));
hold on;
plot([0 S(end)], [1/m 1/m], '-.');
legend('N(t)/t', 'y = 1/\mu');
title('Convergence p.s. de N(t)/t');
```

31.1.3 Illustration de la convergence en loi

```
% On illustre la convergence en loi énoncée au début du texte
clear, clc, clf;
a = 0.5; b = 2;
m = gamma(1+1/a)/b;
v = (gamma(1+2/a) - gamma(1+1/a)^2)/(b^2);
% On calcule 400 échantillons de vecteurs de 5000 sauts
T = 400;
Nmax = 5000;
W = weibull(a, b, T, Nmax);
S = cumsum(W, 2);
% On choisit des t suffisamment petits pour que N(t) soit plus
% petit que l'instant du dernier (N-ième, où N = 5000) saut
```

```
t = [25, 500];
for k = 1:2
  % Calcul du 400-échantillon de N(t)
  N = sum(S <= t(k), 2);
  % Tracé de la fonction de répartition de la variable d'intérêt,
  % approchée par méthode de Monte-Carlo
  A = sort(sqrt(t(k))*(N/t(k)-1/m));
  subplot(2, 1, k);
  stairs(A, (1:T)/T);
  % Tracé de la loi limite
  hold on;
  plot(A, pnorm(sqrt(m^3/v)*A), '-.');
  legend('empirique', 'limite');
  title(['Fonctions de répartition empirique et limite avec t = ', ...
         num2str(t(k))]);
end;
```

31.1.4 Tracé des intervalles de prévision

```
clc, clf, clear;
% Calcul et tracé des différentes valeurs de N_max pour différentes lois
% de Weibull -- sur l'intervalle [0,500] et pour alpha = 0.05
t = 0:0.1:500;
alpha = 0.05;
% On considère trois valeurs possibles de a
a = [1/2, 1, 2];
b = 1;
for k = 1:3
  % Calcul de la moyenne et de la variance
  m = gamma(1+1/a(k))/b;
  v = (gamma(1+2/a(k))-gamma(1+1/a(k))^2)/(b^2);
  % Calcul des trois valeurs de N_max
  [N1, N2, N3] = Nmax(t, m, v, alpha);
  % Tracé des fonctions
  subplot(3, 1, k);
  hold on;
  plot(t, N1, '--');
  plot(t, N2, '-.');
  if (a(k) >= 1)
    plot(t, N3, '-');
  end
  title(['a = ', num2str(a(k)), ' et b = ', num2str(b)]);
  legend('N1', 'N2', 'N3');
end
```

31.1.5 Calcul ou tracé des niveaux des intervalles de prévision ; illustration des inégalités NBU entre variance, espérance et borne déterministe

```
% 1. Calcul d'une estimation par méthode de Monte-Carlo de la probabilité
%    empirique d'appartenir (niveau) aux intervalles de prévision
%    construits avec alpha = 0.05 et pour t dans [0, 10]
%  2. Classement entre les trois quantités du théorème du paragraphe NBU
%     (variance et espérance de N(t), et borne t/mu)
clc, clf, clear;
t = 0:0.1:10;
alpha = 0.05;
% On considère trois valeurs possibles de a
a = [1/2, 1, 2];
b = 1;
for j = 1:3
  % Calcul de la moyenne et de la variance
  m = gamma(1+1/a(j))/b;
  v = (gamma(1+2/a(j))-gamma(1+1/a(j))^2)/(b^2);
  % On calcule 1000 échantillons de suites de 5000 sauts
  W = weibull(a(j), b, 1000, 5000);
  S = cumsum(W, 2);
  % On construit un 1000-échantillon de N évalué régulièrement
  % sur l'intervalle [0, 10] et on regarde combien de valeurs sont
  % dans les intervalles de prévision en moyenne
  R1 = []; R2 = []; R3 = []; N = [];
  for k = t
    % 1000-échantillon du processus N au pas de temps k
    Nk = sum(S <= k, 2);
    N = [N Nk];
    % Calcul des intervalles de prévision en k
    [N1, N2, N3] = Nmax(k, m, v, alpha);
    % Probabilités empiriques en k
    R1 = [R1 mean(Nk > N1)];
    R2 = [R2 mean(Nk > N2)];
    R3 = [R3 mean(Nk > N3)];
  end;
  % 1. Niveaux (ou plutôt, 1 - niveaux)
  %    Tracé de R1, sorties numériques pour R2 et R3
  figure(1);
  subplot(3, 1, j);
  disp(['Erreur maximale pour N2 et N3 (a = ', ...
        num2str(a(j)), ' et b = ', num2str(b), ') : ', ...
        num2str(max(R2)), ' et ', num2str(max(R3))]);
  plot(t, R1, '-');
  hold on;
  title(['a = ', num2str(a(j)), ' et  b = ', num2str(b)]);
  plot([min(t) max(t)], [alpha alpha], '-.');
```

```
  % 2. Tracé des trois quantités à classer
  figure(2);
  subplot(3, 1, j);
  plot(t, mean(N), 'b--', t, var(N), 'b-.', t, t/m, 'b-')
  title(['a = ', num2str(a(j)), ' et  b = ', num2str(b)]);
  legend('espérance', 'variance', 't/\mu');
end
```

31.1.6 Illustration de nos résultats sur un jeu de réelles données

```
clc, clear, clf;
x = SYS1;
alpha = 0.05; % correspond à 0.10/2
n = length(x);

% Utilisation d'un jeu de données antérieures pour
% effectuer une estimation
D = x(1:(n/2)); m = length(D);
figure(1); hold on;
fplot(@(t)(1-exp(-t/mean(D))),[0 2500],'b');
stairs(sort(D),(1:m)/m)
% Estimations : ponctuelle, par intervalle
l = [1/mean(D) exp(-(log(mean(D))-qnorm(1-alpha)))];

% Jeu de données d'évaluation
T = x((1+n/2):end);
t = [0:1:sum(T)];
figure(2);
for k = 1:2
  subplot(1,2,k);
  hold on;
  [N1,N2,N3] = Nmax(t,1/l(k),(1/l(k))^2,alpha);
  stairs(cumsum([0 T]), [0:(length(T))]);
  plot(t,N1,'--b');
  plot(t,N3,'-.b');
  legend('N','N1','N3');
end;
```

```
function x = SYS1;
%
% Voici la fonction chargeant les vraies données.
%
% Sortie :
%   x, le vecteur des vraies données (rangées dans un ordre aléatoire)
```

```
%

x = [3, 30, 113, 81, 115, 9, 2, 20, 20, 15, 138, 50, 77, 24, 108, 88, ...
    670, 120, 26, 114, 325, 55, 242, 68, 422, 180, 10, 1146, 600, 15, ...
    36, 4, 0, 8, 227, 65, 176, 58, 457,300, 97, 263, 452, 255, 197, ...
    193, 6, 79, 816, 1351, 148, 21, 233, 134, 357, 193, 236, 31, 369, ...
    748, 0, 232, 330, 365, 1222, 543, 10, 16, 529, 379, 44, 129, 810, ...
    290, 300, 529, 281, 160, 828, 1011, 445, 296, 1755, 1064, 1783, 860, ...
    983, 707, 33, 868, 724, 2323, 2930, 1461, 843, 12, 261, 1800, 865, ...
    1435, 30, 143, 108, 0, 3110, 1247, 943, 700, 875, 245, 729, 1897, ...
    447, 386, 446, 122, 990, 948, 1082, 22, 75, 482, 5509, 100, 10, ...
    1071, 371, 790, 6150, 3321, 1045, 648, 5485, 1160, 1864, 4116];

[X,I] = sort(rand(1,length(x)));
x = x(I);

end
```

CHAPITRE 32

Estimation de densité de probabilité (compléments)

RÉSUMÉ. Nous fournissons ici quelques extensions et commentaires quant aux résultats du chapitre 24 : l'énoncé et la preuve du lemme de Scheffé et du théorème de Glick ; d'autres applications de l'estimation de densité ; ainsi qu'une discussion des vitesses de convergence possibles dans les résultats d'estimation consistante des densités. Nous présentons enfin les codes des simulations informatiques.

32.1 Compléments mathématiques

32.1.1 Lemme de Scheffé et théorème de Glick

Le lemme de Scheffé ne doit pas être confondu avec le théorème de Scheffé prouvé au début du chapitre 24 (le Théorème 24.1). C'est un résultat analogue au théorème de convergence dominée de Lebesgue, puisqu'il renforce une convergence presque partout ou presque sûre en une convergence $\mathbb{L}^1$.

Lemme 32.1 (de Scheffé) Soit μ une mesure positive sur un espace mesurable $(\Omega, \mathcal{A})$. On considère une fonction $\ell \in \mathbb{L}^1(\mu)$ et une suite (ℓ_n) dans $\mathbb{L}^1(\mu)$ vérifiant d'une part la convergence μ–presque partout $\ell_n \to \ell$ et d'autre part,

$$\int_\Omega |\ell_n| \, \mathrm{d}\mu \longrightarrow \int_\Omega |\ell| \, \mathrm{d}\mu \ .$$

Alors $\ell_n \to \ell$ dans $\mathbb{L}^1(\mu)$.

Les situations les plus souvent considérées sont celle où $\mu = \lambda$, la mesure de Lebesgue, et celles où μ est une probabilité (bien entendu, on parle alors de convergence presque sûre). Dans ce dernier cas, on montre comme dans la preuve du Théorème 10.9 que le résultat du lemme de Scheffé vaut encore lorsque les ℓ_n convergent vers ℓ en $\mathbb{P}$–probabilité plutôt que $\mathbb{P}$–presque sûrement.

Preuve (générale) Il suffit d'appliquer le lemme de Fatou à la suite (g_n) définie par $g_n = |\ell_n| + |\ell| - |\ell_n - \ell| \geqslant 0$: d'une part, μ–presque partout, $g_n \to 2\,|\ell|$, et ainsi

$$2 \int_\Omega |\ell| \, \mathrm{d}\mu = \int_\Omega \big(\liminf g_n\big) \, \mathrm{d}\mu \leqslant \liminf \int_\Omega g_n \, \mathrm{d}\mu = 2 \int_\Omega |\ell| \, \mathrm{d}\mu - \limsup \int_\Omega |\ell_n - \ell| \, \mathrm{d}\mu \ ,$$

soit

$$\|\ell_n - \ell\|_1 = \int_\Omega |\ell_n - \ell| \, \mathrm{d}\mu \longrightarrow 0$$

après simplification des intégrales en $|\ell|$ (qui ont bien une valeur finie). □

Preuve (dans le cas des densités de probabilité) Lorsque $\mu = \lambda$ et que les ℓ_n et ℓ sont des densités de probabilité, la preuve est encore plus directe. On applique le théorème de convergence dominée aux $g_n = (\ell - \ell_n)_+$, dominées par ℓ et convergeant presque partout vers 0 :

$$\|\ell_n - \ell\|_1 = 2 \int_{\mathbb{R}} (\ell - \ell_n)_+ \, \mathrm{d}\lambda \longrightarrow 0 \ ,$$

où l'on a utilisé également une des égalités données par le théorème de Scheffé (Théorème 24.1). □

On généralise cette preuve de la manière suivante. Dans le théorème suivant, il y a une double intégration, selon la mesure de Lebesgue λ et la probabilité $\mathbb{P}$.

Théorème 32.2 (de Glick) *Si l'on dispose d'une suite d'estimateurs $\widehat{f}_n$ définissant chacun une densité de probabilité et convergeant en presque tout point en $\mathbb{P}$–probabilité vers une densité de probabilité f, alors $\widehat{f}_n \to f$ dans $\mathbb{L}^1(\mathbb{P} \otimes \lambda)$. De manière équivalente,*

$$\mathrm{d}_{\mathrm{VT}}\Big(\widehat{f}_n, \, f\Big) \xrightarrow{\mathbb{P}} 0 \ .$$

Preuve On emploie la même technique de preuve que pour le lemme de Scheffé dans le cas des densités de probabilité : deux applications du théorème de convergence dominée. D'une part, pour x tel que $\widehat{f}_n(x) \to f(x)$ en $\mathbb{P}$–probabilité, on a, par l'extension du théorème de convergence dominée aux convergences en probabilité (voir chapitre 10) et par domination par la constante $f(x)$,

$$\varphi_n(x) \overset{\text{not.}}{=} \mathbb{E}\left[\Big(f(x) - \widehat{f}_n(x)\Big)_+\right] \longrightarrow 0 \ .$$

Ainsi, φ_n tend presque partout vers 0 tout en étant dominée par la fonction intégrable f. Par le théorème de Scheffé (on utilise là que f et les $\widehat{f}_n$ sont des densités de probabilité), le théorème de Fubini–Tonelli puis une seconde application du théorème de convergence dominée, il vient finalement

$$\frac{1}{2} \, \mathbb{E}\left[\int_{\mathbb{R}} \Big|f - \widehat{f}_n\Big| \, \mathrm{d}\lambda\right] = \mathbb{E}\left[\int_{\mathbb{R}} \Big(f(x) - \widehat{f}_n(x)\Big)_+ \, \mathrm{d}x\right] = \int_{\mathbb{R}} \varphi_n(x) \, \mathrm{d}x \longrightarrow 0 \ .$$

On vient de prouver la convergence $\widehat{f}_n \to f$ dans $\mathbb{L}^1(\mathbb{P} \otimes \lambda)$. Elle entraîne évidemment la convergence en $\mathbb{P}$–probabilité des distances en variation totale, simplement parce que

de manière générale, la convergence $\mathbb{L}^1(\mathbb{P})$ entraîne la convergence en $\mathbb{P}$–probabilité. Il y a ici équivalence entre les deux convergences car on manipule des variables aléatoires bornées (par 1, s'agissant de distances en variation totale ; on pourra effectuer un raisonnement direct ou recourir au Théorème 10.9). □

32.1.2 Un autre schéma de preuve pour le Théorème 24.7

Sans surprise, il est dans l'esprit des démonstrations concernant les estimateurs par fenêtres glissantes (qui sont, on le rappelle, un cas particulier d'estimateurs par noyaux) et consiste à établir des résultats ponctuels.

On commence par remarquer qu'en tout point de Lebesgue x de f, donc presque partout, $f \star K_h(x) \to f(x)$ lorsque $h \to 0$. Une démonstration possible de ce fait, suggérée par [DL01, exercice 5.9], repose sur une approximation de K en norme $\mathbb{L}^1$ par une combinaison linéaire d'indicatrices de segments. On montre alors qu'en tous ces points, $\mu_n \star K_{h_n}(x) \to f(x)$ en $\mathbb{P}$–probabilité, lorsque $n \to \infty$ avec $h_n \to 0$: on écrit ici la décomposition biais–variance ponctuelle

$$\mathbb{E}\Big[\big(\mu_n \star K_{h_n}(x) - f(x)\big)^2\Big] = \big(f \star K_{h_n}(x) - f(x)\big)^2 + \mathrm{Var}\big(\mu_n \star K_{h_n}(x)\big)$$

et on a indiqué ci-dessus que le premier terme (de biais) tendait vers 0 presque partout. On a montré à la fin du paragraphe 24.1.6 que le second terme (de variance) était égal à $\big(\mathrm{Var}\, Y_1^{n,x}\big)/n$ et qu'on pouvait le majorer, avec les notations utilisées là-bas, par $\big(K'_{h_n} \star f(x)\big)\,\sigma^2/nh_n$. Ce majorant tend vers 0 dès que $nh_n \to \infty$ car en un point de Lebesgue de f, on a également $f \star K'_h(x) \to f(x)$ lorsque $h \to 0$. Le théorème de Glick (Théorème 32.2) permet alors de passer de la convergence $\mu_n \star K_{h_n}(x) \to f(x)$ en $\mathbb{P}$–probabilité en tous les points de Lebesgue à la convergence

$$\mathbb{E}\Big[\big\|\mu_n \star K_{h_n} - f\big\|_1\Big] \to 0\ .$$

Ce schéma de preuve, proposé par [DL01, Théorème 9.2], a l'avantage de ressembler fort au schéma de l'estimateur par fenêtres glissantes mais a l'inconvénient majeur de reposer sur deux lemmes assez techniques, le théorème de densité de Lebesgue (sa version forte avec les ensembles rétrécissant convenablement, voir [Rud98, paragraphes 7.9 et 7.10]) et sa conséquence, que $f \star K_{h_n}(x) \to f(x)$ en tout point de Lebesgue x de f. Nous n'avons pas montré cette dernière mais comme on s'en doute, la preuve n'est pas difficile, c'est essentiellement une question d'écriture et d'emploi des bons théorèmes (inégalité de convolution) et définitions (pour choisir l'approximation de K, il faut se souvenir des caractérisations et constructions de l'intégrale de Lebesgue). Dans la preuve détaillée au paragraphe 24.1.6, nous n'avions pas eu à utiliser le théorème de densité de Lebesgue et des techniques tout à fait élémentaires ont suffi à prouver le théorème de consistance pour les estimateurs par noyaux.

32.1.3 À propos des autres applications de l'estimation de densité

Simulation de variables

Imaginons que l'on dispose d'un n–échantillon $X_1, \dots, X_n$ (de taille modérée) distribué selon une loi μ de densité f et que l'on veuille simuler de nouvelles variables aléatoires $X_{n+1}, \dots, X_m$ indépendantes et identiquement distribuées selon une loi aussi proche que possible de celle de μ. On requiert cela à des fins d'estimation de l'intégrale sous μ d'une fonction compliquée, pour laquelle il est nécessaire d'avoir une grande taille d'échantillon m, voir la fin du chapitre 9.

On estime alors f par $\widehat{f}_n$ sur le n–échantillon initial, puis, en utilisant une méthode de simulation de variables aléatoires mettant en jeu la densité (la plus connue est la méthode de rejet, voir le paragraphe 9.3), on obtient de nouvelles variables aléatoires, qui, conditionnellement à $X_1, \dots, X_n$, sont indépendantes et identiquement distribuées selon la loi de densité $\widehat{f}_n$. Par cette méthode, on a un contrôle asymptotique de l'écart de la distribution des nouvelles observations à celle de l'échantillon initial et par conséquent, pour peu que par exemple la fonction à intégrer soit bornée, de l'écart des intégrales sous μ et sous ses estimées. Cet écart est en effet quantifié par des distances en variation totale.

Notons que la méthode qui aurait consisté à estimer la fonction de répartition (par la fonction de répartition empirique) pour simuler les nouvelles variables par méthode d'inversion (voir chapitre 9) aurait été décevante, puisqu'elle consiste à simplement tirer au hasard et avec remise les $X_{n+1}, \dots, X_m$ parmi les $X_1, \dots, X_n$. On retrouve le fait que la mesure empirique est concentrée en un nombre fini de points. En particulier, dans cette méthode naïve, les queues de distribution posent problème. La loi associée à la fonction de répartition empirique canonique a en effet un support borné, alors que ce n'est pas nécessairement le cas de la loi $\widehat{\mu}_n$ donnée par un bon estimateur de la densité $\widehat{f}_n$, surtout si l'on a une idée *a priori* de la forme ou du poids de la queue et qu'on l'exploite. C'est le cas notamment si l'on est dans un cadre d'estimation paramétrique et que l'on connaît la forme de f.

Illustrations informatiques de convergence en loi

Dans les illustrations informatiques, on représente souvent la convergence en loi par une convergence de densités. Typiquement, si on établit, par exemple, qu'une suite d'estimateurs converge en loi, $X^{(n)} \rightsquigarrow \mathcal{N}(0,1)$ pour fixer les idées, alors, pour illustrer cette convergence, on va fixer une grande valeur de n et simuler un N_n–échantillon distribué selon la loi de $X^{(n)}$, noté

$$X_1^{(n)}, \dots, X_{N_n}^{(n)} ;$$

après quoi, on établira l'histogramme des valeurs du N_n–échantillon et on essaiera de se convaincre qu'il a une forme proche d'une courbe en cloche, qui est précisément la

forme de la densité de la loi normale standard. C'est ce que nous avons réalisé par exemple dans les simulation des chapitres 22 et 23.

Mais ces histogrammes, nous l'avons vu, sont en fait des estimateurs de la densité de la loi de $X^{(n)}$ lorsque celle-ci existe et que la taille des classes est bien choisie en fonction de N_n. On essaie donc de représenter une convergence en loi par une proximité de densités. Or, *a priori*, la convergence en loi est équivalente à la convergence des fonctions de répartitions, c'est-à-dire, à celle des intégrales des densités (lorsque ces dernières existent). Cela ne signifie généralement pas que les densités elles-mêmes convergent ponctuellement. C'est, en conclusion, une illustration dangereuse...

... Sauf qu'elle est beaucoup plus agréable et plus parlante visuellement qu'une convergence de fonctions de répartitions : plus parlante parce que les différences d'aires entre la densité (estimée sur le N_n–échantillon) au rang n et la densité de la loi limite forment exactement la distance en variation totale. On peut donc lire cette dernière graphiquement et quantifier facilement les écarts entre les deux distributions.

Il arrive en fait que la convergence des lois s'accompagne de celles des densités de probabilité associées. Il est clair que si ces densités convergent dans $\mathbb{L}^1(\lambda)$ (ce qui découle d'une convergence ponctuelle, *via* le lemme de Scheffé), alors les fonctions de répartition convergent ponctuellement et il y a convergence des lois. Réciproquement, une version plus forte du théorème de la limite centrale aboutissant à une convergence $\mathbb{L}^1$ des densités (et donc une convergence en variation totale des lois) s'appelle le théorème de la limite locale ; il est présenté par [vdV98, paragraphe 2.9].

32.1.4 Quelques commentaires sur l'optimalité des procédures d'estimation

On donne ici quelques indications sur la vitesse de convergence des estimateurs par noyaux. Les preuves des résultats avancés ne sont pas si difficiles mais elles sont trop longues pour pouvoir être reproduites ici. L'objectif de ce qui suit est de mettre en lumière la difficulté intrinsèque du problème d'estimation de densité.

En termes de performances individuelles, *id est*, pour une densité f donnée, on a le résultat suivant.

Théorème 32.3 *Pour toute densité f et pour tout noyau K, il existe une constante $A(f,K) \geqslant 0.86$ telle que les estimateurs par noyaux construits sur K vérifient*

$$\liminf_{n\to\infty} \inf_{h>0} \quad n^{2/5}\, \mathbb{E}\left[\left\| \widehat{f}_{n,h} - f \right\|_1 \right] \;\geqslant\; A(f,K)\ .$$

En particulier, la distance en variation totale entre tout estimateur par noyau $\widehat{f}_{n,h_n}$ et la vraie densité f est (asymptotiquement) plus grande que $0.43\, n^{-2/5}$, où n désigne toujours la taille de l'échantillon. Si par exemple, on veut que les probabilités de tous les événements soient estimées avec un écart (absolu) d'au plus 1 % par rapport à leur vraie fréquence, il faudra que n soit (c'est ce que suggère l'asymptotique) tel que

$0.43n^{-2/5} \leqslant 0.01$, i.e., $n \geqslant 12\,000$. C'est un nombre d'observations très grand, que l'on rencontre peu souvent en pratique. Et ce résultat vaut pour toutes les densités de probabilité, même les plus régulières !

> Remarque au passage. Lorsque l'on a affaire à des probabilités discrètes, on peut estimer simultanément, avec les fréquences empiriques, la probabilité de tous les événements à un écart $n^{-1/2}$ près. Le théorème de la limite centrale en donne l'intuition et on peut le prouver en utilisant des inégalités de concentration, par exemple celle de McDiarmid (voir par exemple [DL01, Théorème 2.2]). Ici, on a affaire à des lois continues et estimer leur densité se fait au mieux à la vitesse non pas $n^{-1/2}$ mais $n^{-2/5}$: on perd donc au moins un ordre de grandeur $n^{1/2-2/5} = n^{1/10}$.

Est-ce que cette vitesse $n^{-2/5}$ est atteignable ? En général, non, et il faudra restreindre la classe des densités considérées au vu du résultat suivant. On peut en effet montrer que pour toute suite (a_n) tendant vers 0, pour toute stratégie d'estimation (f_n) de la densité, il existe une densité f (que l'on peut prendre bornée par 3, à support dans $[0,1]$ et infiniment dérivable sur $[0,1[$) telle que

$$\limsup_{n\to\infty} \frac{1}{a_n} \mathbb{E}\Big[\|f_n - f\|_1\Big] = \infty \ .$$

On a vu plus haut qu'on pouvait toujours faire en sorte que la distance en variation totale tende vers 0, quelle que soit la densité de probabilité sous-jacente f, en retenant des estimateurs de la forme

$$f_n = \widehat{f}_{n,h_n} \ .$$

Nous apprenons ici que cette convergence peut être arbitrairement lente. Plus lente, par exemple, que $a_n = 1/(\log\log\log n)\ldots$ pour laquelle il faudrait recueillir un nombre phénoménal d'observations pour obtenir un résultat non trivial.

Par conséquent, pour garantir des vitesses de convergence uniformes, on doit se restreindre à des sous-classes de densités régulières. Nous présentons quelques exemples de résultats que l'on peut exhiber en ce sens. Considérons par exemple la classe $\mathcal{F}$ des densités f à support compact, absolument continues de dérivée f' également absolument continue, de dérivée f'' vérifiant $c(f) = \int |f''| \, \mathrm{d}\lambda < \infty$. Pour $f \in \mathcal{F}$, on note $s(f)$ la mesure de Lebesgue du support de f. Pour tout $C > 0$, on introduit alors la sous-classe $\mathcal{F}_C$ des densités $f \in \mathcal{F}$ telles que $s^2(f)\, c(f) \leqslant C$. Pour cette sous-classe, on a le résultat de convergence suivant.

Théorème 32.4 *Pour tout noyau K borné, symétrique et à support dans $[-1,1]$, il existe une constante $B(K)$ telle que pour la suite des estimateurs par noyaux construits sur K :*

$$\sup_{f\in\mathcal{F}_C} \inf_{h>0} \mathbb{E}\left[\left\|\widehat{f}_{n,h} - f\right\|_1\right] \leqslant \frac{B(K)\, C^{1/5}}{n^{2/5}} \ .$$

Le théorème précédent n'indique pas directement comment choisir h. Un coup d'œil à la preuve montre en revanche qu'on a les majorations suivantes pour les termes de biais et variance respectivement,

$$h^2 \frac{c(f)}{2} \int_{\mathbb{R}} x^2 K(x)\,\mathrm{d}x \qquad \text{et} \qquad \sqrt{\frac{s(f)+2h}{nh}} \sqrt{\int_{\mathbb{R}} K^2\,\mathrm{d}\lambda}\,.$$

Ainsi, le choix $h_n = n^{-1/5}$ conduit bien à la vitesse uniforme $n^{-2/5}$. Cependant, si l'on voulait optimiser le majorant, il s'agirait de prendre h_n égal à $n^{-1/5}$ à une constante multiplicative près, cette dernière dépendant de $c(f)$ et $s(f)$, que l'on ne connaît pas. C'est ce qui explique que dans le théorème on ne puisse échanger le supremum et l'infimum.

On atteint également des vitesses uniformes sur des classes comme celle des densités lipschitiziennes de constante de Lipschitz et de support compact fixés (vitesse $n^{-1/3}$), celle des densités monotones et bornées par c fixé sur un intervalle $[a,b]$ fixé (vitesse $n^{-1/3}$), etc.

Le Théorème 32.4 est, en un sens, inefficace, parce qu'il ne donne pas de formule explicite pour la fenêtre h_n en fonction de $\mathcal{F}_C$. C'est pourquoi, au vu des bonnes performances des estimateurs par noyaux et pour se débarrasser de ce problème délicat qu'est le choix de h, [DL01] résout le problème suivant : trouver une stratégie d'estimation (f_n) telle que pour toute densité f,

$$\mathbb{E}\Big[\,\|f_n - f\|_1\Big] \leqslant \big(3+o(1)\big) \inf_{h>0} \mathbb{E}\left[\left\|\widehat{f}_{n,h} - f\right\|_1\right] + o\big(n^{-2/5}\big)\,.$$

C'est ce que l'on appelle une inégalité d'oracle. En particulier, sur la classe $\mathcal{F}_C$, elle conduit à

$$\mathbb{E}\Big[\,\|f_n - f\|_1\Big] \leqslant \big(3+o(1)\big)\frac{B(K)\,C^{1/5}}{n^{2/5}} + o\big(n^{-2/5}\big) = \big(3+o(1)\big)\frac{B(K)\,C^{1/5}}{n^{2/5}}\,.$$

La suite d'estimateurs (f_n) n'est pas nécessairement à noyaux, mais elle est définie à partir des briques fondamentales que sont les estimateurs à noyaux $\widehat{f}_{n,h}$.

32.2 Codes des simulations

32.2.1 Calcul des estimations

```
function [f, f_inf, f_sup] = FenGliss(x, D, h)
%
% Calcul de l'estimateur par fenêtres glissantes,
% et des intervalles de confiance ponctuels associés
%
```

```
% Entrées :
%   x est le vecteur des observations
%   D (pour design) est l'ensemble des points où l'on veut
%     calculer l'estimateur de la densité
%   h est un réel positif donnant la taille de fenêtre
% Sortie :
%   f, f_inf, f_sup sont des vecteurs de même taille que D
%   f est le vecteur des estimées de la densité
%   f_inf, f_sup sont les bornes des intervalles de confiance
%                ponctuels à 95 %
%

  N = length(D);
  n = length(x);

  f = zeros(1,N);
  for j = 1:N
    f(j) = sum((D(j)-h < x) & (D(j)+h >= x));
  end
  f = f / (2*n*h);

  q = qnorm(.975);
  f_sup = f + q * sqrt(f/(2*n*h));
  f_inf = f - q * sqrt(f/(2*n*h));

end




function f = EstNoyau(x, h, D, Kfun)
%
% Estimation par noyaux
%
% Entrées :
%   x est le vecteur des observations,
%   h est un réel donnant le paramètre de bande,
%   D (pour design) est l'ensemble des points où l'on veut
%     calculer l'estimateur de la densité,
%   Kfun est le nom d'une fonction MATLAB donnant un noyau
%        (p.ex. 'dnorm' pour le noyau gaussien standard)
% Sortie :
%   Un vecteur f de même taille que D, où chaque élément
%   est l'estimée de la densité au point correspondant de D
%

  n = length(D);
```

```
  K = str2func(Kfun);
  f = zeros(1,n);
  for j = 1:n
    f(j) = sum(K((D(j)-x)/h));
  end
  f = f/(length(x)*h);

end



function f = Histogr(x, D, h)
%
% Estimation par histogrammes
%
% Entrées :
%   x est le vecteur des observations
%   D (pour design) est l'ensemble des points où l'on veut
%     calculer l'estimateur de la densité
%   h est un réel donnant le paramètre de bande
% Sortie :
%   Un vecteur f de même taille que D, où chaque élément
%   est l'estimée de la densité au point correspondant de D
%   sur la partition régulière de pas h
%

  N = length(D);
  n = length(x);

  f = zeros(1,N);
  for j = 1:N
    kh = B(D(j),h);
    f(j) = sum((kh <= x) & (kh+h > x));
  end
  f = f / (n*h);

end

function b = B(y,h)
  % Calcul de l'index de la partition régulière de pas h
  % où se trouve y
  b = floor(y/h)*h;
end
```

32.2.2 Densités et noyaux

```
function d = dEpa(x)
```

```
%
% Noyau d'Epanechnikov
%
% Entrée :
%   x un vecteur
% Sortie :
%   d vecteur de même taille que x
%

  d = 3/4 * max(1-x.^2, 0);

end
```

```
function d = dgauss(D, p, m, s)
%
% Mélange de densités gaussiennes
%
% Entrées :
%   D un vecteur
%   p, m, s trois vecteurs de même taille précisant respectivement
%           les coefficients du mélange, les moyennes des lois
%           normales et leurs écarts-types
% Sortie :
%   d vecteur de même taille que D donnant la densité en chaque point
%

  d = zeros(1, length(D));
  for j = 1:length(p)
    d = d + p(j) * dnorm(D, m(j), s(j));
  end;

end
```

```
function d = dunif(D, p, a, b)
%
% Mélange de densités uniformes
%
% Entrées :
%   D un vecteur
%   p, a, b trois vecteurs de même taille précisant respectivement
%           les coefficients du mélange, les bornes inférieures et
%           supérieures des intervalles de lois uniformes
```

```
% Sortie :
%   d vecteur de même taille que D donnant la densité en chaque point
%

  d = zeros(1, length(D));
  for j = 1:length(a)
    ind = (D >= a(j)) & (D <= b(j));
    d(ind) = d(ind) + p(j)/(b(j)-a(j));
  end;

end
```

32.2.3 Tirages aléatoires

```
function X = MelGauss(n, p, m, s)
%
% Tirage aléatoire selon un mélange de lois gaussiennes
%
% Entrées :
%   n, un entier
%   p, m, s trois vecteurs de même taille précisant respectivement
%           les coefficients du mélange, les moyennes des lois
%           normales et leurs écarts-types
% Sortie :
%   X, un vecteur de taille n échantillonné selon la loi de mélange
%

  y = rand(1,n);
  X = zeros(1,n);
  q = [0 cumsum(p)];
  for j = 1:length(p)
    ind = (y <= q(j+1)) & (y > q(j));
    X(ind) = rnorm(sum(ind+0), m(j), s(j));
  end;

end
```

```
function X = MelUnif(n, p, a, b)
%
% Tirage aléatoire selon un mélange de lois uniformes
%
% Entrées :
%   n, un entier
%   p, a, b trois vecteurs de même taille précisant respectivement
```

```
%           les coefficients du mélange, les bornes inférieures et
%           supérieures des intervalles de lois uniformes
% Sortie :
%   X, un vecteur de taille n échantillonné selon la loi de mélange
%

  y = rand(1,n);
  X = zeros(1,n);
  q = [0 cumsum(p)];
  for j = 1:length(p)
    ind = (y <= q(j+1)) & (y > q(j));
    X(ind) = runif(sum(ind+0), a(j), b(j));
  end;

end

function d = runif(k, a, b)
  d = a + (b-a)*rand(1,k);
end
```

32.2.4 Fonctions auxiliaires

```
function d = dvt(D, delta, f, g)
%
% Approximation de la distance en variation
% entre la vraie densité et une estimation
%
% Entrées :
%   D doit être un vecteur de points régulièrement espacés
%   delta est cet espacement régulier
%   g est la vraie densité
%   f est un vecteur d'estimations (de g, calculées en les éléments de D)
% Sortie :
%   d réel positif
%

  d = delta * trapz(abs(f-g(D))) /2;

end
```

32.2.5 Fonction de comparaison des quatre méthodes

```
function R = compareGr(x, D, delta, true_f, h, trace)
%
% Comparaison des quatres méthodes d'estimation de densité
%
% Entrées :
```

```
%   x est le vecteur des observations,
%   D (pour design) est un ensemble de points régulièrement espacés
%     où l'on veut calculer l'estimateur de la densité,
%   delta est l'espacement uniforme entre les points de D,
%   true_f est une fonction donnant la vraie densité,
%   h est le paramètre réel précisant la largeur de bande
%   trace est un booléen indiquant si un tracé graphique est souhaité
% Sortie :
%   R est un vecteur à quatre éléments précisant quatre distances en
%     variation
%

  if trace
    clf;
    dmin = min(D);
    dmax = max(D);
    ymax = max(true_f(D))*1.25;
  end;

  f = Histogr(x, D, h);
  dH = dvt(D, delta, f, true_f);
  if trace
    subplot(2,2,1)
    box on; hold on;
    plot(D, f, 'k');
    plot(D, true_f(D), 'k-.');
    axis([dmin dmax -0.05 ymax]);
    title(strcat(['Par histogrammes (d_{VT} = '  num2str(dH,2)  ')']));
  end;

  [f, f_inf, f_sup] = FenGliss(x, D, h);
  dF = dvt(D, delta, f, true_f);
  if trace
    subplot(2,2,2)
    box on; hold on;
    plot(D, f, 'k');
    plot(D, true_f(D), 'k-.');
    axis([dmin dmax -0.05 ymax]);
    title(strcat(['Fenêtres glissantes (d_{VT} = ' num2str(dF,2) ')']));
  end;

  f = EstNoyau(x, h, D, 'dnorm');
  dN1 = dvt(D, delta, f, true_f);
  if trace
    subplot(2,2,3)
    box on; hold on;
    plot(D, f, 'k');
```

```
    plot(D, true_f(D), 'k-.');
    axis([dmin dmax -0.05 ymax]);
    title(strcat(['Par noyau gaussien (d_{VT} = ' num2str(dN1,2) ')']));
  end;

  f = EstNoyau(x, h, D, 'KEpa');
  dN2 = dvt(D, delta, f, true_f);
  if trace
    subplot(2,2,4)
    box on; hold on;
    plot(D, f, 'k');
    plot(D, true_f(D), 'k-.');
    axis([dmin dmax -0.05 ymax]);
    title(strcat(['Par noyau d''Epanechnikov (d_{VT} = ', ...
                  num2str(dN2,2), ')']));
  end;

  R = [dH dF dN1 dN2];

end
```

32.2.6 Programme maître (script)

```
clc, clf, clear;

% Paramètres du mélange uniforme
p = [0.25 0.45 0.30];
a = [-1 0 2.5];
b = [ 1 1 3.5];
% Paramètres du mélange gaussien
q = [0.45 0.20 0.35];
m = [-1.5 0.5 2.5];
s = [ 0.75 0.60 0.45];

% 1. Intervalles de confiance pour fenêtres glissantes
%    et mélange de lois uniformes

N = 800;
h = [0.05 0.15 0.25 0.35];

figure(1);
clf;
delta = 0.01;
D = (min(a)-0.5):delta:(max(b)+0.5);
for i = 1:4;
  subplot(2,2,i)
  box on; hold on;
```

```
  x = MelUnif(N, p, a, b)';
  [f, f_inf, f_sup] = FenGliss(x, D, h(i));
  plot(D, dunif(D, p, a, b), 'k');
  plot(D, f, 'k:', D, f_inf, 'k', D,f_sup, 'k');
  title(strcat(['Paramètre de bande h = ' num2str(h(i))]));
  axis([min(a)-0.5 max(b)+0.5 -0.05 0.9]);
end

% 2. Comparaisons graphiques entre les quatre méthodes

N = 400;

% Mélange de lois uniformes

h = 0.15;
figure(2);
x = MelUnif(N, p, a, b)';
f = @(x)dunif(x, p, a, b);
compareGr(x, D, delta, f, h, true);

% Mélange de lois gaussiennes

h = 0.30;
figure(3);
ymin = min(m)-3*max(s);
ymax = max(m)+3*max(s);
D = ymin:delta:ymax;
x = MelGauss(N, q, m, s)';
g = @(x)dgauss(x, q, m, s);
compareGr(x, D, delta, g, h, true);

% 3. Comparaison numérique

% Selon que l'on veut comparer les performances pour l'estimation
% du mélange de densités uniformes ou de densités gaussiennes
u = false;

N = 400;      % Taille de chaque échantillon
K = 1000;     % Nombre d'itérations de Monte-Carlo
h = [0.02 0.05 0.15 0.20 0.30 0.40 0.50 0.60];

% Dans les deux cas, on augmente D pour mesurer les écarts
% sur toute la droite réelle
if u
  D = (-3):delta:(5.5);
  ff = f;
else
```

```
  D = (-5):delta:5;
  ff = g;
end;

% Boucle d'estimation par méthode de Monte-Carlo
R = zeros(length(h), 4, K);
for j = 1:length(h)
  for k = 1:K
    if u
      x = MelUnif(N, p, a, b)';
    else
      x = MelGauss(N, q, m, s)';
    end;
    R(j,:,k) = compareGr(x, D, delta, ff, h(j), false);
  end;
end;

% Affichage des résultats : estimations des moyennes et écart-types
disp(mean(R, 3));
disp(std(R, [], 3));
```

CHAPITRE 33

Classification de données (compléments)

RÉSUMÉ. Dans cette annexe au chapitre 25, on s'attache à montrer que le problème de classification considéré, qui est un problème d'apprentissage, est en fait beaucoup plus facile que le problème auxiliaire d'estimation de la fonction de régression et qui donne lieu à certaines règles de classification. On énonce en premier lieu un argument général, puis on se place dans le cadre paramétrique : on y oppose la vitesse d'estimation $1/\sqrt{n}$ à la vitesse d'apprentissage $1/n$. Plus précisément, on prouve alors le Théorème 25.7 et on donne des éléments de preuve pour le Théorème 25.8. On procure également les codes des simulations `Matlab`.

33.1 Compléments mathématiques

33.1.1 Il est plus facile de classer que d'estimer la fonction de régression

On étoffe ici la discussion de la fin du paragraphe 25.1.4. On note $\mathbb{L}^1(\mu)$ et $\mathbb{L}^2(\mu)$ les classes de fonctions sur $\mathbb{R}^d$ respectivement intégrables et de carré intégrable par rapport à la loi μ de X. L'inégalité (25.3) entraîne en particulier que

$$L(g_n) - L(g^*) \leqslant 2\,\mathbb{E}\Big[\big|\widehat{\eta}_n(X) - \eta(X)\big| \;\Big|\; Z_1^n\Big] = 2\int_{\mathbb{R}^d} \big|\widehat{\eta}_n(x) - \eta(x)\big|\,\mathrm{d}\mu(x)$$
$$\overset{\text{not.}}{=} 2\,\big\|\widehat{\eta}_n - \eta\big\|_{\mathbb{L}^1(\mu)} \leqslant 2\,\big\|\widehat{\eta}_n - \eta\big\|_{\mathbb{L}^2(\mu)} \tag{33.1}$$

(par l'inégalité de Cauchy–Schwarz). La majoration en norme $\mathbb{L}^2(\mu)$ est proposée essentiellement parce que s'agissant d'une fonction de régression, caractérisée par une propriété de meilleure approximation au sens $\mathbb{L}^2$, il est naturel de mesurer l'écart de l'estimation par $\widehat{\eta}_n$ (construite sur les données) à η en norme $\mathbb{L}^2(\mu)$.

Il découle de (33.1) une majoration de l'espérance du risque de classification,

$$\mathbb{E}\big[L(g_n)\big] - L(g^*) \leqslant 2\,\mathbb{E}\Big[\big\|\widehat{\eta}_n - \eta\big\|_{\mathbb{L}^2(\mu)}\Big]\;;$$

et, par une seconde application de l'inégalité de Jensen,

$$\mathbb{E}\Big[\big\|\widehat{\eta}_n - \eta\big\|_{\mathbb{L}^2(\mu)}\Big] \leqslant \sqrt{\mathbb{E}\Big[\big(\widehat{\eta}_n(X) - \eta(X)\big)^2\Big]}\;. \tag{33.2}$$

C'est la vitesse de convergence vers 0 de ce majorant, qui est une quantité naturelle à étudier s'agissant d'estimateurs définis par une propriété de meilleure approximation $\mathbb{L}^2$, que nous comparons à celle du risque de classification.

Le membre de droite de (33.2) tend vers 0 si et seulement si $\|\widehat{\eta}_n - \eta\|_{\mathbb{L}^2(\mu)} \to 0$ en $\mathbb{P}$–probabilité : au vu de (33.2), il suffit de prouver le sens direct, ce qui se fait en remarquant, comme dans la preuve de la Définition–Théorème 25.2, qu'une suite de variables aléatoires positives bornées converge en $\mathbb{P}$–probabilité vers 0 si et seulement si elle converge vers 0 dans $\mathbb{L}^2(\mathbb{P})$. L'hypothèse sur les suites considérées dans le théorème suivant est donc nécessaire pour que le résultat ait un sens.

Théorème 33.1 *Soient $(\widehat{\eta}_n)$ une suite d'estimateurs $\mathbb{P}$–consistants en norme $\mathbb{L}^2(\mu)$ de la fonction de régression η,* id est*,*

$$\|\widehat{\eta}_n - \eta\|_{\mathbb{L}^2(\mu)} \longrightarrow 0 \qquad \text{en } \mathbb{P}\text{–probabilité,}$$

et (g_n) la règle de classification associée par plug-in. Alors

$$\lim_{n\to\infty} \frac{\mathbb{E}\big[L(g_n)\big] - L(g^*)}{\sqrt{\mathbb{E}\left[\big(\widehat{\eta}_n(X) - \eta(X)\big)^2\right]}} = 0 \ .$$

Ce théorème indique ainsi que le risque de classification converge plus rapidement vers 0 que l'erreur d'estimation de la fonction de régression : le problème de classification est plus simple que le problème d'estimation (non-paramétrique) de la fonction de régression. On en rappelle l'explication heuristique donnée dans le corrigé principal : pour bien classifier, il suffit d'estimer η de manière précise dans la zone où η est proche de $1/2$ alors qu'on peut s'autoriser une estimation moins précise autre part, tant qu'elle ne nous fait pas franchir la dite barre des $1/2$.

Preuve On utilise l'inégalité de Cauchy–Schwarz et un découpage du membre de droite de (25.3) selon que $|\eta - 1/2|$ est plus petit ou plus grand que ε :

$$\begin{aligned}
&\mathbb{E}\big[L(g_n)\big] - L(g^*) \\
&\qquad \leqslant 2\sqrt{\mathbb{E}\left[\big(\widehat{\eta}_n(X) - \eta(X)\big)^2\right]}\sqrt{\mathbb{P}\Big\{g_n(X) \neq g^*(X), \quad \eta(X) \neq 1/2\Big\}} \\
&\qquad \leqslant 2\sqrt{\mathbb{E}\left[\big(\widehat{\eta}_n(X) - \eta(X)\big)^2\right]}\left(\sqrt{\mathbb{P}\Big\{\big|\eta(X) - 1/2\big| \leqslant \varepsilon, \quad \eta(X) \neq 1/2\Big\}}\right. \\
&\qquad\qquad \left. + \sqrt{\mathbb{P}\Big\{\big|\eta(X) - 1/2\big| > \varepsilon, \quad g_n(X) \neq g^*(X)\Big\}}\right) \ .
\end{aligned}$$

Il suffit de montrer la convergence vers 0 de la somme entre parenthèses, lorsque $n \to +\infty$ puis $\varepsilon \to 0$. D'une part, par convergence monotone,

$$\lim_{\varepsilon\to 0} \mathbb{P}\Big\{\big|\eta(X) - 1/2\big| \leqslant \varepsilon, \ \eta(X) \neq 1/2\Big\} = 0 \ .$$

D'autre part, par un fait déjà utilisé en (25.3),

$$\Big\{ \big|\eta(X) - 1/2\big| > \varepsilon,\ g_n(X) \neq g^*(X) \Big\} \subseteq \Big\{ \big|\eta(X) - \widehat{\eta}_n(X)\big| > \varepsilon \Big\} \ ,$$

qui est de $\mathbb{P}$–probabilité tendant vers 0 lorsque $n \to +\infty$, puisque la suite des $\widehat{\eta}_n(X)$ converge dans $\mathbb{L}^2(\mathbb{P})$ et donc en $\mathbb{P}$–probabilité vers $\eta(X)$. □

On voit donc qu'il est plus simple de classifier que d'estimer la fonction de régression ; et paradoxalement, nous nous sommes plutôt intéressés dans le corrigé du chapitre 25 à obtenir des estimateurs de la fonction de régression universellement consistants (*id est*, consistants dans le modèle où toutes les lois jointes ν sont possibles). Nous en avons construit par histogrammes et par noyaux. Nous avons procédé ainsi simplement parce qu'il est à la portée d'un étudiant de master 1 ou d'un agrégatif d'en montrer l'universelle consistance. Ce n'est pas le cas de règles plus efficaces.

33.1.2 Preuve du Théorème 25.7

Le Théorème 25.8 améliore le résultat du Théorème 25.7, mais c'est au prix d'une preuve plus complexe. Nous prouvons donc d'abord ce dernier et étudions ensuite comment l'améliorer pour trouver la vitesse d'apprentissage $1/n$. Dans ce paragraphe et le suivant, on reprend les notations du paragraphe 25.1.10.

Les techniques employées ici sont exactement les mêmes que celles du Lemme 22.5 et du Théorème 22.6.

Preuve Dans ce modèle paramétrique, il vient, par (25.1), (25.6) et le rappel fait au début du paragraphe 25.1.10 selon lequel la loi μ de X est absolument continue par rapport à la mesure de Lebesgue de densité $(1-p)\, f_{\lambda_0} + p\, f_{\lambda_1}$, que l'erreur de classification de toute fonction de classification g vaut

$$\begin{aligned} L(g) &= \int_{\mathbb{R}} \Big(\eta(x)\, \mathbb{I}_{\{g(x)=0\}} + \big(1-\eta(x)\big)\, \mathbb{I}_{\{g(x)=1\}} \Big)\, \mathrm{d}\mu(x) \\ &= \int_{\mathbb{R}} \Big(p\, f_{\lambda_1}(x)\, \mathbb{I}_{\{g(x)=0\}} + (1-p)\, f_{\lambda_0}(x)\, \mathbb{I}_{\{g(x)=1\}} \Big)\, \mathrm{d}x \\ &= 1 - \int_{\mathbb{R}} \Big(p\, f_{\lambda_1}(x)\, \mathbb{I}_{\{g(x)=1\}} + (1-p)\, f_{\lambda_0}(x)\, \mathbb{I}_{\{g(x)=0\}} \Big)\, \mathrm{d}x \ . \end{aligned}$$

Par un calcul déjà effectué page 344 (pour la classification multi-classes), on voit que la différence entre les erreurs de classification de g_n et g^* est donnée par

$$\begin{aligned} & L\big(g_n\big) - L(g^*) \\ &= \int_{\mathbb{R}} \Big(p\, f_{\lambda_1}(x)\, \mathbb{I}_{\{g^*(x)=1\}} + (1-p)\, f_{\lambda_0}(x)\, \mathbb{I}_{\{g^*(x)=0\}} \Big)\, \mathrm{d}x \\ &\quad - \int_{\mathbb{R}} \Big(\widehat{p}_n\, f_{\widehat{\lambda}_{1,n}}(x)\, \mathbb{I}_{\{g_n(x)=1\}} + \big(1-\widehat{p}_n\big)\, f_{\widehat{\lambda}_{0,n}}(x)\, \mathbb{I}_{\{g_n(x)=0\}} \Big)\, \mathrm{d}x \end{aligned}$$

$$\leqslant 4\left|\widehat{p}_n - p\right| + 2\int_{\mathbb{R}} \left|f_{\lambda_0}(x) - f_{\widehat{\lambda}_{0,n}}(x)\right| \mathrm{d}x + 2\int_{\mathbb{R}} \left|f_{\lambda_1}(x) - f_{\widehat{\lambda}_{1,n}}(x)\right| \mathrm{d}x \,. \qquad (33.3)$$

On montre que (deux fois) la distance en variation entre deux lois exponentielles de paramètres $\lambda < \lambda'$ est majorée par l'écart entre λ et λ' ; par le théorème de Scheffé (Théorème 24.1) et la forme des densités considérées, il vient

$$\begin{aligned}\int_{\mathbb{R}} \left|f_\lambda(x) - f_{\lambda'}(x)\right| \mathrm{d}x &= 2\int_0^{\varphi(\lambda,\,\lambda')} \left(f_\lambda(x) - f_\mu(x)\right) \mathrm{d}x \\ &= 2\left[e^{-\lambda x} - e^{-\lambda' x}\right]_0^{\varphi(\lambda,\,\lambda')} = 2\,e^{-\lambda'\,\varphi(\lambda,\,\lambda')}\left(e^{-(\lambda-\lambda')\,\varphi(\lambda,\,\lambda')} - 1\right)\end{aligned}$$

où

$$\varphi(\lambda,\,\lambda') = \psi(\lambda,\,\lambda',\,1/2) = \frac{1}{\lambda - \lambda'}\,\log\frac{\lambda}{\lambda'} \geqslant 0$$

est le point x où les densités sont égales, $f_\lambda(x) = f'_\lambda(x)$. En substituant cette valeur et en majorant le facteur exponentiel par 1, il reste

$$\int_{\mathbb{R}} \left|f_\lambda(x) - f_{\lambda'}(x)\right| \mathrm{d}x \leqslant 2\left(\frac{\lambda'}{\lambda} - 1\right) \,.$$

Par symétrie dans le raisonnement précédent entre λ et λ', on a donc majoré l'erreur de classification selon

$$L\big(g_n\big) - L(g^*) \leqslant M_n \overset{\text{not.}}{=} 4\left|\widehat{p}_n - p\right| + 4\left|\frac{\widehat{\lambda}_{0,n} - \lambda_0}{\min\left\{\lambda_0,\,\widehat{\lambda}_{0,n}\right\}}\right| + 4\left|\frac{\widehat{\lambda}_{1,n} - \lambda_1}{\min\left\{\lambda_1,\,\widehat{\lambda}_{1,n}\right\}}\right| \,. \quad (33.4)$$

On montre maintenant que $\sqrt{n}\,M_n$, le membre de droite de cette inégalité multiplié par $\sqrt{n}$, converge en loi (vers une loi que l'on va préciser par l'analyse).

On applique pour cela le théorème de la limite centrale dans sa version vectorielle, en considérant les vecteurs $U_1, \ldots, U_n$, indépendants et identiquement distribués, où pour $j = 1, \ldots, n$,

$$U_j = \begin{bmatrix} U_{j,1} \\ U_{j,2} \\ U_{j,3} \end{bmatrix} = \begin{bmatrix} \mathbb{I}_{\{Y_j = 1\}} \\ X_j\,\mathbb{I}_{\{Y_j = 0\}} \\ X_j\,\mathbb{I}_{\{Y_j = 1\}} \end{bmatrix} \,.$$

On note $\overline{U}_n$ la moyenne empirique des U_j, c'est-à-dire, $\overline{U}_n = (U_1 + \ldots + U_n)/n$; on a

$$\sqrt{n}\left(\overline{U}_n - \underline{m}\right) \rightsquigarrow \mathcal{N}\left(0,\,\Gamma\right) \qquad (33.5)$$

où

$$\underline{m} = \begin{bmatrix} p \\ (1-p)/\lambda_0 \\ p/\lambda_1 \end{bmatrix}$$

est l'espérance commune des U_j et Γ leur matrice de variance–covariance :

$$\Gamma = \begin{bmatrix} p(1-p) & -p(1-p)/\lambda_0 & p(1-p)/\lambda_1 \\ -p(1-p)/\lambda_0 & (1-p)(1+p)/\lambda_0^2 & -p(1-p)/(\lambda_0\lambda_1) \\ p(1-p)/\lambda_1 & -p(1-p)/(\lambda_0\lambda_1) & p(2-p)/\lambda_1^2 \end{bmatrix} .$$

On indique les éléments qui ont permis de calculer Γ : l'espérance et la variance d'une loi exponentielle $\mathcal{E}(\lambda)$ sont respectivement $1/\lambda$ et $2/\lambda^2$, celle d'une loi de Bernoulli de paramètre p est $p(1-p)$. Alors d'une part pour les éléments diagonaux, on a par exemple

$$\begin{aligned} \Gamma_{2,2} = \mathbb{E}\Big[X^2\,\mathbb{I}_{\{Y=0\}}\Big] - \Big(\mathbb{E}\big[X\,\mathbb{I}_{\{Y=0\}}\big]\Big)^2 &= \mathbb{P}\{Y=0\}\;\mathbb{E}\Big[X^2 \;\Big|\; Y=0\Big] - \left(\frac{1-p}{\lambda_0}\right)^2 \\ &= (1-p)\,\frac{2}{\lambda_0^2} - \frac{(1-p)^2}{\lambda_0^2} = \frac{(1-p)(1+p)}{\lambda_0^2} , \end{aligned}$$

de quoi on déduit la valeur de $\Gamma_{3,3}$ par symétrie. D'autre part, pour les éléments hors diagonale, on utilise les valeurs des composantes de $\underline{m}$ et le fait que vu les produits de fonctions indicatrices,

$$\mathbb{E}\big[U_{j,1}\,U_{j,2}\big] = 0 , \qquad \mathbb{E}\big[U_{j,1}\,U_{j,3}\big] = \mathbb{E}\big[U_{j,3}\big] = p/\lambda_1 , \qquad \mathbb{E}\big[U_{j,2}\,U_{j,3}\big] = 0 .$$

On note $\Phi : \mathbb{R}^3 \to \mathbb{R}^3$ la fonction vectorielle

$$\Phi : \begin{bmatrix} x \\ y \\ z \end{bmatrix} \longmapsto \begin{bmatrix} x \\ (1-x)/y \\ x/z \end{bmatrix} ;$$

elle a été prise de telle sorte que

$$\Phi(\underline{m}) = \begin{bmatrix} p \\ \lambda_0 \\ \lambda_1 \end{bmatrix} \qquad \text{et} \qquad \Phi\big(\overline{U}_n\big) = \begin{bmatrix} \widehat{p}_n \\ \widehat{\lambda}_{0,n} \\ \widehat{\lambda}_{1,n} \end{bmatrix} .$$

Φ est différentiable sur $\mathbb{R} \times \mathbb{R}_+^* \times \mathbb{R}_+^*$, de matrice différentielle égale à

$$\mathrm{D}\Phi(x,y,z) = \begin{bmatrix} 1 & 0 & 0 \\ -1/y & x/y^2 & 0 \\ 1/z & 0 & -x/z^2 \end{bmatrix} .$$

La méthode delta (voir le Lemme 2.9) permet alors de passer de la convergence (33.5) à

$$\sqrt{n}\left(\begin{bmatrix} \widehat{p}_n \\ \widehat{\lambda}_{0,n} \\ \widehat{\lambda}_{1,n} \end{bmatrix} - \begin{bmatrix} p \\ \lambda_0 \\ \lambda_1 \end{bmatrix}\right) \rightsquigarrow \mathrm{D}\Phi(\underline{m})\,\mathcal{N}\big(0,\,\Gamma\big) . \tag{33.6}$$

En particulier, $\widehat{\lambda}_{0,n}$ et $\widehat{\lambda}_{1,n}$ sont des estimateurs consistants de λ_0 et λ_1, convergent en probabilité vers eux, de sorte que le lemme de Slutzky (utilisé dans sa forme multidimensionnelle, Lemme 10.8) garantit finalement que

$$\sqrt{n}\left(\begin{bmatrix} \widehat{p}_n \\ \widehat{\lambda}_{0,n}/\min\left\{\lambda_0,\, \widehat{\lambda}_{0,n}\right\} \\ \widehat{\lambda}_{1,n}/\min\left\{\lambda_1,\, \widehat{\lambda}_{1,n}\right\} \end{bmatrix} - \begin{bmatrix} p \\ \lambda_0/\min\left\{\lambda_0,\, \widehat{\lambda}_{0,n}\right\} \\ \lambda_1/\min\left\{\lambda_1,\, \widehat{\lambda}_{1,n}\right\} \end{bmatrix}\right) \rightsquigarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/\lambda_0 & 0 \\ 0 & 0 & 1/\lambda_1 \end{bmatrix} \mathrm{D}\Phi(\underline{m})\, \mathcal{N}\big(0,\, \Gamma\big)\ .$$

Enfin, l'application de la fonction continue $g : (x, y, z) \mapsto 4\big(|x| + |y| + |z|\big)$ montre que $\sqrt{n}\, M_n$ converge en loi,

$$\sqrt{n}\, M_n = 4\sqrt{n}\left(|\widehat{p}_n - p| + \left|\frac{\widehat{\lambda}_{0,n} - \lambda_0}{\min\left\{\widehat{\lambda}_{0,n},\, \lambda_0\right\}}\right| + \left|\frac{\widehat{\lambda}_{1,n} - \lambda_1}{\min\left\{\widehat{\lambda}_{1,n},\, \lambda_1\right\}}\right|\right) \rightsquigarrow \mathcal{L}$$

où $\mathcal{L}$ est la loi de

$$g\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/\lambda_0 & 0 \\ 0 & 0 & 1/\lambda_1 \end{bmatrix} \mathrm{D}\Phi(\underline{m})\, \mathcal{N}\big(0,\, \Gamma\big)\right)\ ,$$

et est donc une loi dérivée de la loi normale. □

33.1.3 Éléments de preuve pour le Théorème 25.8

Si l'on regarde attentivement comment (33.3) a été obtenue, on constate qu'on a négligé un terme $\mathbb{I}_{\{g^*(x) \neq g_n(x)\}}$, et c'est pourquoi on est retombé sur un problème d'estimation pure et que l'on a obtenu la vitesse paramétrique $1/\sqrt{n}$. S'agissant d'un problème d'apprentissage, la vitesse est plus rapide : nous allons montrer comment considérer le terme $\mathbb{I}_{\{g^*(x) \neq g_n(x)\}}$ permet de montrer que la vitesse est en fait au moins en $1/n$.

On suppose ici $\lambda_0 \neq \lambda_1$; on peut traiter le cas $\lambda_0 = \lambda_1$ de manière similaire à ce qui suit et nous laissons ce travail en exercice.

Plus précisément, les majorations des intégrales de la page 344 procèdent de majoration des intégrandes uniquement ; or, lorsque $g^*(x)$ et $g_n(x)$ sont égales, l'erreur commise est nulle. On aboutit donc par cet argument plus fin à la majoration

$$\begin{aligned} L\big(g_n\big) - L(g^*) \ \leqslant\ M_n'' &\overset{\text{not.}}{=} 4\,|\widehat{p}_n - p| \int_{\mathbb{R}} \mathbb{I}_{\{g^*(x) \neq g_n(x)\}}\, \mathrm{d}x \\ &\quad + 2 \int_{\mathbb{R}} \left|f_{\lambda_0}(x) - f_{\hat{\lambda}_{0,n}}(x)\right| \mathbb{I}_{\{g^*(x) \neq g_n(x)\}}\, \mathrm{d}x \end{aligned}$$

$$+\,2\int_{\mathbb{R}}\left|f_{\lambda_1}(x)-f_{\widehat{\lambda}_{1,n}}(x)\right|\,\mathbb{I}_{\{g^*(x)\neq g_n(x)\}}\,\mathrm{d}x\ .$$

On appelle M_n'' ce majorant car on va le majorer à son tour par la quantité M_n' apparaissant dans le Théorème 25.8.

Vu (25.7) et l'expression similaire pour g_n en remplaçant les trois paramètres par leurs estimateurs, on a deux situations :

– un cas très défavorable où $\lambda_0-\lambda_1$ et $\widehat{\lambda}_{0,n}-\widehat{\lambda}_{1,n}$ sont de signes contraires ; les fonctions de classification g^* et g_n sont alors différentes partout, sauf sur l'intervalle

$$I_n=\left[\min\left\{\psi_0,\widehat{\psi}_n\right\},\ \max\left\{\psi_0,\widehat{\psi}_n\right\}\right]$$

$$\text{où}\qquad \psi_0=\psi(\lambda_0,\,\lambda_1,\,p)\qquad\text{et}\qquad \widehat{\psi}_n=\psi\left(\widehat{\lambda}_{0,n},\,\widehat{\lambda}_{1,n},\,\widehat{p}_n\right)\ ,$$

ce qui conduit à une grosse erreur de classification ; cette situation arrive heureusement exponentiellement peu souvent ;

– le cas le plus fréquent où les deux fonctions de classification ne diffèrent que sur l'intervalle I_n.

L'heuristique est que la longueur de I_n est de l'ordre de $1/\sqrt{n}$ et que les différences $f_{\lambda_0}-f_{\hat{\lambda}_{0,n}}$ et $f_{\lambda_1}-f_{\hat{\lambda}_{1,n}}$ sont aussi de ce même ordre de grandeur sur cet intervalle, ce qui donne la vitesse $1/n$ au final. On va maintenant détailler ce raisonnement.

On note à cet effet D_n l'événement donné par le cas très défavorable et on va majorer notre borne intermédiaire M_n'' sur D_n et D_n^c pour obtenir la borne M_n' du Théorème 25.8.

Premièrement, une étude de fonctions rapide montre que pour $0<\lambda<\lambda'$,

$$\left\|\,f_\lambda-f_{\lambda'}\,\right\|_\infty=e^{-\lambda y^*}\left(\lambda-\lambda' e^{-(\lambda'-\lambda)y^*}\right)\leqslant\frac{\lambda}{\lambda'}\left(\lambda'-\lambda\right)\leqslant|\lambda'-\lambda|$$

où le supremum dans le membre de gauche est atteint en $y^*>0$ tel que

$$e^{-(\lambda'-\lambda)y^*}=\left(\frac{\lambda}{\lambda'}\right)^2\ .$$

Ainsi, sur l'événement D_n^c, comme g^* et g_n ne diffèrent que sur I_n, de longueur $\left|\widehat{\psi}_n-\psi_0\right|$ (et en majorant des facteurs 2 par un facteur 4),

$$M_n''\leqslant 4\left|\widehat{\psi}_n-\psi_0\right|\left(\left|\widehat{p}_n-p\right|+\left|\widehat{\lambda}_{0,n}-\lambda_0\right|+\left|\widehat{\lambda}_{1,n}-\lambda_1\right|\right)\ ;$$

les deux termes du produit présent dans la majoration étant fonctions continues de $p-\widehat{p}_n$, $\lambda_0-\widehat{\lambda}_{0,n}$ et $\lambda-\widehat{\lambda}_{1,n}$, et puisque la convergence en loi passe aux fonctions

continues, on déduit, comme précédemment, de (33.6) une convergence en loi

$$\sqrt{n}\begin{bmatrix} \left|\widehat{\psi}_n - \psi_0\right| \\ \left|\widehat{p}_n - p\right| + \left|\widehat{\lambda}_{0,n} - \lambda_0\right| + \left|\widehat{\lambda}_{1,n} - \lambda_1\right| \end{bmatrix} \rightsquigarrow \dots$$

(vers une loi qu'il ne serait pas difficile, mais pénible, d'exhiber explicitement) et en particulier, on a également la convergence en loi

$$4n\left|\widehat{\psi}_n - \psi_0\right|\left(\left|\widehat{p}_n - p\right| + \left|\widehat{\lambda}_{0,n} - \lambda_0\right| + \left|\widehat{\lambda}_{1,n} - \lambda_1\right|\right) \rightsquigarrow \dots$$

Comme l'erreur de classification est toujours majorée par 1, on procède à une majoration grossière sur D_n et on pose donc

$$M_n' = \mathbb{I}_{D_n} + 4\left|\widehat{\psi}_n - \psi_0\right|\left(\left|\widehat{p}_n - p\right| + \left|\widehat{\lambda}_{0,n} - \lambda_0\right| + \left|\widehat{\lambda}_{1,n} - \lambda_1\right|\right)\mathbb{I}_{D_n^c} .$$

La preuve sera conclue par lemme de Slutzky pour peu que $\mathbb{P}\big(D_n\big) = o(1/n)$.

À cet effet, on rappelle qu'au vu de (25.7) le mauvais cas indiqué par D_n arrive seulement lorsque

– λ_0 et λ_1 d'une part,
– $\widehat{\lambda}_{0,n}$ et $\widehat{\lambda}_{1,n}$ d'autre part,

sont classés dans des ordres différents. C'est équivalent au fait que

$$(\lambda_1 - \lambda_0)\Big(\widehat{\lambda}_{1,n} - \widehat{\lambda}_{0,n}\Big) < 0 \ , \qquad \text{ou encore,} \qquad \left(\frac{1}{\lambda_1} - \frac{1}{\lambda_0}\right)\left(\frac{1}{\widehat{\lambda}_{1,n}} - \frac{1}{\widehat{\lambda}_{0,n}}\right) < 0 \ .$$

En particulier, une condition nécessaire est que

$$\text{ou} \quad \left|\frac{1}{\widehat{\lambda}_{1,n}} - \frac{1}{\lambda_1}\right| \geqslant \frac{1}{2}\left|\frac{1}{\lambda_0} - \frac{1}{\lambda_1}\right| \ , \qquad \text{ou} \quad \left|\frac{1}{\widehat{\lambda}_{0,n}} - \frac{1}{\lambda_0}\right| \geqslant \frac{1}{2}\left|\frac{1}{\lambda_0} - \frac{1}{\lambda_1}\right| \ .$$

Nous allons utiliser une inégalité de Bernstein (voir le Théorème 28.2, qui se place cependant dans le cadre plus général des martingales), à ceci près qu'il nous en faut une version pour des variables non bornées : ici, nous avons essentiellement affaire à des lois exponentielles. Nous recourons par conséquent à la version avec condition de moments exponentiels énoncée, par exemple, dans [Mas07, chapitre 2]. Elle indique qu'avec probabilité au moins $1 - 2\,\delta$,

$$\left|\frac{np}{\lambda_1} - \sum_{t=1}^{n} X_t\,\mathbb{I}_{\{Y_t=1\}}\right| \leqslant \square\sqrt{n\log\frac{1}{\delta}} + \square\log\frac{1}{\delta}$$

où les $\square$ désignent des quantités dépendant des paramètres p et λ_1, mais pas de n. De même, l'inégalité de Hoeffding (voir le Lemme 3.7) donne qu'avec probabilité au moins $1 - 2\,\delta$,

$$\left|np - \sum_{t=1}^{n}\mathbb{I}_{\{Y_t=1\}}\right| \leqslant \sqrt{\frac{n}{2}\,\log\frac{2}{\delta}} \ .$$

Au final, en combinant ces deux inégalités, on a avec probabilité au moins $1 - 4\,\delta$ que

$$\left| \frac{1}{\widehat{\lambda}_{1,n}} - \frac{1}{\lambda_1} \right| \leqslant \square \sqrt{\frac{\log 1/\delta}{n}} + \square \frac{\log 1/\delta}{n} \; ;$$

en particulier, la probabilité que

$$\left| \frac{1}{\widehat{\lambda}_{1,n}} - \frac{1}{\lambda_1} \right| \geqslant \frac{1}{2} \left| \frac{1}{\lambda_0} - \frac{1}{\lambda_1} \right| > 0$$

est plus petite que $\square\, e^{-\square\, n} = o(1/n)$, où les constantes $\square$ dépendent de λ_0, λ_1, p mais pas de n.

De même pour l'écart entre $1/\widehat{\lambda}_{0,n}$ et $1/\lambda_0$; cela conclut cette preuve (qui est plutôt du niveau master 2, comme on l'aura remarqué par le recours à l'inégalité de Bernstein).

33.2 Codes des simulations

33.2.1 Fonctions de simulations des données artificielles

```
function [X, Y] = SimExp(N)
%
% Simulation de couples observation--étiquette
% selon le modèle exponentiel
%
% Entrée :
%   N, un entier naturel
% Sorties :
%   X et Y, deux vecteurs de taille N contenant
%           respectivement des observations (selon
%           des lois exponentielles) et des étiquettes
%

  X = zeros(1,N);
  Y = [rand(1,N) <= 0.5];

  c = (Y == 1);
  n = sum(c);

  X(c) = rexpweib([1 n], 1);
  X(not(c)) = rexpweib([1 N-n], 0.4);

end
```

```
function [X, Y] = SimGauss(N)
%
% Simulation de couples observation--étiquette
% selon le modèle gaussien
%
% Entrée :
%   N, un entier naturel
% Sorties :
%   X et Y, deux vecteurs de taille 2 x N contenant
%           respectivement des observations (selon des lois
%           normales bi-dimensionnelles) et des étiquettes
%

  X = zeros(2,N);
  Y = [rand(1,N) <= 0.4];

  c = (Y == 1);
  n = sum(c);

  X(:,c) = [2 0; 0.5 1] * randn([2 n]);
  X(1,c) = X(1,c) + 2;
  X(2,c) = X(2,c) - 1.5;

  X(:,not(c)) = [1 -1; 0 1] * randn([2 N-n]);

end
```

33.2.2 Fonction de classification par histogrammes cubiques

```
function D = CHistoCubiq(X, x, y, h)
%
% Classification par histogrammes cubiques
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
%   h, le paramètre de grille
% Sortie :
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0, 0.5, 1)
%

  n = size(x); n = n(end);
  c = floor(X/h);
  [U, I, J] = unique(c','rows');
  D = [];
```

```
  for u = U'
    ind = (repmat(u*h, 1, n) <= x) & (x < repmat((u+1)*h, 1, n));
    ind = logical(prod(ind+0, 1));
    D = [D compare(sum(y(ind)),sum(1-y(ind)))];
  end;

  if (length(D) >= 2)
    D = D(J);
  else
    D = D(J)';
  end;

end
```

33.2.3 Fonction auxiliaire utile pour les méthodes décidant selon une majorité

```
function r = compare(a, b)
%
% Fonction auxiliaire nécessitée lors de toute décision majoritaire
% Il y a indétermination si les deux termes à comparer sont issus
% tous deux de sommes sur aucun élément (tous deux nuls) ou, plus
% généralement, s'ils sont égaux
%
% Entrées :
%   a et b deux vecteurs de scalaires positifs
% Sortie :
%   r un vecteur comparant a et b terme à terme
%

  r = (a > b) + 0.5 * (a == b);

end
```

33.2.4 Fonction de classification par méthodes de noyaux

```
function D = CNoyaux(X, x, y, k, h)
%
% Classification par méthodes de noyaux (gaussien ou uniforme symétrique)
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
%   k, booléen, indique le type de noyau
%      (k = true donne le noyau uniforme, k = false le gaussien)
%   h, le paramètre de bande
% Sortie :
```

```
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0, 0.5, 1)
%

  n = size(x);
  n = n(end);

  b = (y == 1);
  D = [];
  for u = X
    M = (repmat(u, 1, n) - x)/h;
    if k
      K = logical(prod(0+(abs(M) <= 1), 1));
    else
      K = exp(-sum(M.^2, 1)/2);
    end;
    D = [D compare(sum(K(b)), sum(K(not(b))))];
  end;

end
```

33.2.5 Fonctions de classification par plug-in paramétrique

```
function D = PluginExp(X, x, y, estimer)
%
% Classification par plug-in paramétrique (cas du modèle exponentiel)
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
%   estimer, true si l'on ne connaît pas les paramètres et qu'il faut
%            les estimer (cas de la construction d'une fonction de
%            classification); false si l'on veut approximer L^*
% Sortie :
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0 et 1)
%

  if estimer
    l0 = 1/mean(x(:, y == 0));
    l1 = 1/mean(x(:, y == 1));
    p = mean(y);
  else
    l0 = 0.4;
    l1 = 1;
    p = 0.5;
```

```
  end;

  D = compare(p*dexp(X,l1), (1-p)*dexp(X,l0));

end

function d = dexp(X, l)
% Densité de la loi exponentielle de paramètre l en X

  d = l * exp(-l * X);

end
```

```
function D = PluginExpSeuil(X, x, y)
%
% Classification par détermination du meilleur seuil empirique
% (cas du modèle exponentiel)
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
% Sortie :
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0 et 1, ici, pas d'indétermination : p.s. les
%       nouvelles observations X ne tombent pas pile sur le seuil)
%

  [x, I] = sort(x);
  y = y(I);
  P = [0 (x(1:(end-1))+x(2:end))/2 x(end)+1];

  R = [];
  for p = P
    d = (x <= p);
    R = [R; sum(abs(d-y)) sum(abs((1-d)-y))];
  end;

  R = R';
  [m, J] = min(R(:));
  p = P(floor((J+1)/2));

  if mod(J,2)
    D = (X <= p);
```

```
  else
    D = (X >= p);
  end;

end



function D = PluginGaussien(X, x, y, estimer)
%
% Classification par plug-in paramétrique (cas du modèle gaussien)
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
%   estimer, true si l'on ne connaît pas les paramètres et qu'il faut
%            les estimer (cas de la construction d'une fonction de
%            classification); false si l'on veut approximer L^*
% Sortie :
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0 et 1)
%

  if estimer
    C1 = cov(x(:,y==1)');
    m1 = mean(x(:,y==1), 2);
    C0 = cov(x(:,y==0)');
    m0 = mean(x(:,y==0), 2);
    p = mean(y);
  else
    C1 = [2 0; 0.5 1]; C1 = C1*C1';
    m1 = [2; -1.5];
    C0 = [1 -1; 0 1]; C0 = C0*C0';
    m0 = [0; 0];
    p = 0.4;
  end;

  D = compare(p*dnorm2(X,m1,C1), (1-p)*dnorm2(X,m0,C0));

end

function d = dnorm2(X, m, C)
% A un facteur multiplicatif près, la densité de la loi normale
% de moyenne m et de matrice de covariance C aux points X

  I = inv(C);
```

```
  isd = 1/sqrt(det(C));
  d = [];
  for x = X
    d = [d isd * exp(-(x-m)'*I*(x-m)/2)];
  end;

end
```

33.2.6 Fonction de classification par plus proches voisins

```
function D = kNN(X, x, y, k)
%
% Classification par les plus proches voisins
%
% Entrées :
%   X, des observations à classer
%   x, des observations pour lesquelles on a déjà :
%   y, le vecteur des étiquettes des éléments de x
%   k, le nombre de voisins à considérer
% Sortie :
%   D, le vecteur des étiquettes pour les éléments de X
%      (valeurs possibles : 0 et 1 ; on prendra garde à lancer
%       cette fonction avec k impair pour éviter les indéterminations)
%

  n = size(x);
  n = n(end);
  %
  D = [];
  for u = X
    dist = sum((x-repmat(u, 1, n)).^2, 1);
    [dist, I] = sort(dist);
    D = [D (mean(y(I(1:k))) >= 1/2)];
  end;

end
```

33.2.7 Fonction de tracé des résultats

```
function [] = TraceEG(unidim, x, y, X, Y, D, grille)
%
% Fonction de tracé des résultats d'une classification
%
% Entrées :
%   unidim, vaut true si on a affaire au modèle exponentiel et false pour
%           le modèle gaussien
%   x, des observations pour lesquelles on a déjà vu :
```

```
%   y, le vecteur des étiquettes des éléments de x
%   X, des observations classées par une fonction de classification en :
%   D, le vecteur des étiquettes proposées pour les éléments de X
%   Y, le vecteur des vraies étiquettes pour les éléments de X
%   grille, true si une grille doit être tracée et false sinon
% Sorties :
%   Deux figures, une pour les vraies étiquettes 1, et l'autre, pour les 0
%

  symbol = {'^', 'x', '+'};
  taille = {6, 7, 6};
  % '^' = triangle : quand bien classifié
  % 'x' = croix diagonale : quand mal classifié
  % '+' = croix verticale : quand indétermination

  if unidim
    X = [X; jitter(length(X))];
    x = [x; jitter(length(x))];
  end;

  for i = [0 1]
    figure(1+i);
    clf;
    hold on;
    for k = [0 1]
      gca1 = plot(x(1, y == k), x(2, y == k), 'ok', 'MarkerSize', 5);
      if k
        set(gca1, 'MarkerFaceColor', 'k');
      end;
    end;
    for d = [0 1 0.5]
      p = (Y == i) & (D == d);
      if not(isempty(p))
        gca2 = plot(X(1,p), X(2,p), [symbol{indice(i,d)} 'k'], ...
                    'MarkerSize', taille{indice(i,d)});
        if grille
            grid on;
        end;
        if i
          set(gca2, 'MarkerFaceColor', 'k');
        end;
      end;
    end;
    if unidim
      axis([-0.5 6 -0.2 0.2]);
      plot([-0.5 6], [0 0], 'k');
      set(get(gca1, 'Parent'), 'YTick', [])
```

```
    end;
  end;

end

function V = jitter(n)
% Choix d'une ordonnée aléatoire

  jitter = 0.1;
  V = 2 * jitter * rand(1, n) - jitter;

end

function I = indice(i,d)
% Choix du symbole selon l'étiquette lue 0, 0.5 ou 1, et sa
% correspondance à la vraie étiquette i

  I = 3*(d==0.5) + (d~=0.5)*(abs(i-d)+1);

end
```

33.2.8 Fichier script permettant de réaliser des simulations et tracés

```
% Exemple de fonction maître pour réaliser les tracés
clc, clear;
N = 50;
M = 50;
unidim = false;
if unidim
    sim = @SimExp;
else
    sim = @SimGauss;
end;
[x, y] = sim(N);
[X, Y] = sim(M);

  D = CHistoCubiq(X, x, y, 1);   grille = true;
% D = CNoyaux(X, x, y, true, 1); grille = false;
% D = PluginExp(X, x, y, true);  grille = false;

TraceEG(unidim, x, y, X, Y, D, grille);
```

33.2.9 Fichier script permettant une comparaison numérique des performances des différentes méthodes

```
% Fichier maître pour l'évaluation et la comparaison des performances
% numériques des différentes méthodes de classification
```

```
clc, clear;
unidim = true;
N = 50;
M = 200;
m = 200;

tic;
E = [];
% Double moyenne de Monte-Carlo pour calculer l'erreur moyenne
% Une boucle sur j pour la seconde moyenne, la première
% moyenne est effectuée lors du calcul des composantes de R
for j = 1:m
  % Simulation des données
  if unidim
    sim = @SimExp;
  else
    sim = @SimGauss;
  end;
  [x, y] = sim(N);
  [X, Y] = sim(M);
  %
  % Vecteur des résultats pour un j donné
  R = [];
  % Histogrammes cubiques
  for h = [0.5 1 2 3]
    D = CHistoCubiq(X, x, y, h);
    R = [R mean(abs(D-Y))];
  end;
  for h = [0.5 1 2 3]
  % Par noyau uniforme
    D = CNoyaux(X, x, y, true, h);
    R = [R mean(abs(D-Y))];
    % Par noyau gaussien
    D = CNoyaux(X, x, y, false, h);
    R = [R mean(abs(D-Y))];
  end;
  % Méthode k-NN
  for k = [1 3 11]
    D = kNN(X, x, y, k);
    R = [R mean(abs(D-Y))];
  end;
  % Par plug-in
  % Et on calcule au passage une estimation de L^*
  if unidim
    D1 = PluginExp(X, x, y, true);
    D2 = PluginExpSeuil(X, x, y);
```

```
    Dtheo = PluginExp(X, x, y, false);
    R = [R mean(abs(D1-Y)) mean(abs(D2-Y)) mean(abs(Dtheo-Y))];
  else
    D = PluginGaussien(X, x, y, true);
    Dtheo = PluginGaussien(X, x, y, false);
    R = [R mean(abs(D-Y)) mean(abs(Dtheo-Y))];
  end;
  %
  E = [E; R];
end;
disp(mean(E));
toc;
```

CHAPITRE 34

Compression de données (compléments)

Résumé. Ce chapitre complète le chapitre 26. On étudie ici le modèle formé par une source émettant des symboles selon une loi stationnaire. On commence notamment par définir le taux d'entropie d'un processus stationnaire. On procure également les codes des simulations `Matlab`.

34.1 Compléments mathématiques

34.1.1 Sous-additivité de l'entropie, taux d'entropie d'un processus stationnaire

Dans ce paragraphe, on veut définir ce qu'est le taux d'entropie d'un processus stationnaire $X = (X_1, X_2, \ldots)$. On rappelle qu'on note $\mathbb{P}_N$ la loi des N premières marginales X_1^N de ce processus et $\mathbb{P}_\infty$ sa loi sur $\mathcal{A}^{\mathbb{N}}$.

Définition 34.1 On appelle taux d'entropie H_∞ du processus X la limite des entropies moyennes $H(\mathbb{P}_N)/N$, lorsque cette dernière existe :

$$H_\infty = \lim_{N\to\infty} \frac{H(\mathbb{P}_N)}{N} \ .$$

Par exemple, quand le processus est formé de variables aléatoires indépendantes et identiquement distribuées selon une loi commune $\boldsymbol{p}$, on a pour tout entier $N \geqslant 1$ l'égalité $H(\mathbb{P}_N)/N = H(\boldsymbol{p})$. Le taux d'entropie existe alors et est égal à cette valeur commune : $H_\infty = H(\boldsymbol{p})$.

On va montrer que plus généralement, le taux d'entropie est défini pour tout processus stationnaire. À cet effet, on se propose de montrer au préalable que la suite formée par les entropies $H(\mathbb{P}_N)$ est sous-additive.

Sous-additivité de l'entropie

On l'établit dans un cadre général. On considère deux ensembles finis $\mathcal{X}$ et $\mathcal{Y}$, et une probabilité $\boldsymbol{p}$ sur $\mathcal{X} \times \mathcal{Y}$. Les marginales sur $\mathcal{X}$ et $\mathcal{Y}$ sont notées respectivement $\boldsymbol{p}_\mathcal{X}$ et $\boldsymbol{p}_\mathcal{Y}$.

Proposition 34.2 L'entropie de $\boldsymbol{p}$ est majorée par la somme des entropies de ses marginales $\boldsymbol{p}_\mathcal{X}$ et $\boldsymbol{p}_\mathcal{Y}$,

$$H(\boldsymbol{p}) \leqslant H(\boldsymbol{p}_\mathcal{X}) + H(\boldsymbol{p}_\mathcal{Y}) \ ,$$

avec égalité si et seulement si $\boldsymbol{p}$ est une loi–produit, *id est*, $\boldsymbol{p} = \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}$ est le produit de ses lois marginales.

Preuve L'entropie de la loi jointe $H(\boldsymbol{p})$ vaut, par définition,

$$\begin{aligned} H(\boldsymbol{p}) &= -\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{(x,y)} \log_2 p_{(x,y)} \\ &= -\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{(x,y)} \log_2 \frac{p_{(x,y)}}{p_x p_y} - \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{(x,y)} \left(\log_2 p_x + \log_2 p_y\right) \\ &= -\mathcal{K}\left(\boldsymbol{p},\ \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}\right) + H(\boldsymbol{p}_{\mathcal{X}}) + H(\boldsymbol{p}_{\mathcal{Y}}) \ . \end{aligned}$$

La troisième égalité procède de la définition des marginales, en particulier, on a par exemple $\sum_y p_{(x,y)} = p_x$. La deuxième égalité est légitimée par le fait que si $p_{(x,y)} > 0$, alors $p_x > 0$ et $p_y > 0$ (soit $\boldsymbol{p} \ll \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}$) ; il n'y a pas d'introduction de formes indéterminées. En utilisant la positivité de $\mathcal{K}$ (voir le Lemme 26.6), il vient alors

$$H(\boldsymbol{p}) \leqslant H(\boldsymbol{p}_{\mathcal{X}}) + H(\boldsymbol{p}_{\mathcal{Y}}) \ .$$

On note que l'égalité a lieu si et seulement si $\mathcal{K}\left(\boldsymbol{p},\ \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}\right) = 0$, soit, si et seulement si $\boldsymbol{p} = \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}$ (voir, là encore, le Lemme 26.6). □

> Remarque au passage. La quantité $\mathcal{K}\left(\boldsymbol{p},\ \boldsymbol{p}_{\mathcal{X}} \otimes \boldsymbol{p}_{\mathcal{Y}}\right)$ est appelée information mutuelle (entre les deux marginales de $\boldsymbol{p}$). On définit également une notion d'entropie conditionnelle : la différence $H(\boldsymbol{p}) - H(\boldsymbol{p}_{\mathcal{X}})$ est l'entropie conditionnelle de la loi jointe par rapport à sa première marginale. Pour les liens entre entropie d'une loi jointe, entropie conditionnelle et information mutuelle, on lira avec profit [CT91, chapitre 2].

Taux d'entropie d'un processus stationnaire

On rappelle qu'on dit que X est stationnaire si pour tous entiers $t \geqslant 1$ et $k \geqslant 1$, on a l'égalité des lois

$$(X_1, \ldots, X_k) \overset{(\mathrm{d})}{=} (X_{t+1}, \ldots, X_{t+k}) \ .$$

Dans ce cas, la suite des entropies moyennes $H(\mathbb{P}_N)/N$ admet un comportement asymptotique similaire à celui d'une suite décroissante.

Proposition 34.3 Si X est stationnaire, alors il admet un taux d'entropie, égal à

$$H_\infty = \lim_{N\to\infty} \frac{H(\mathbb{P}_N)}{N} = \inf_{N\in\mathbb{N}^*} \frac{H(\mathbb{P}_N)}{N} \ .$$

Preuve On rappelle que $\mathbb{P}_N$ désigne la loi $\mathbb{P}^{X_1^N}$ des N premières marginales X_1^N. Par la Proposition 34.2, pour N et M deux entiers non nuls, on a

$$H(\mathbb{P}_{N+M}) = H\left(\mathbb{P}^{\left(X_1^N,\, X_{N+1}^{N+M}\right)}\right) \leqslant H\left(\mathbb{P}^{X_1^N}\right) + H\left(\mathbb{P}^{X_{N+1}^{N+M}}\right) = H(\mathbb{P}_N) + H(\mathbb{P}_M)\ ,$$

la dernière égalité utilisant l'hypothèse de stationnarité. La suite de terme général $u_N = H(\mathbb{P}_N)$ est donc sous-additive, et par conséquent, (u_N/N) est convergente. On note H_∞ sa limite, qui est égale à $\inf_k\ u_k/k$. □

Remarque au passage. Rappelons la preuve de ce résultat élémentaire d'analyse. On fixe un entier $n \geqslant 1$ et on considère un entier $N \geqslant n$, que l'on écrit sous la forme $N = qn + r$ où $q \geqslant 1$ et $r \leqslant n-1$ sont des entiers. Par récurrence immédiate, il vient $u_N \leqslant q u_n + u_r$, d'où

$$\frac{u_N}{N} \leqslant \frac{q}{N}\, u_n + \frac{u_r}{N} \leqslant \frac{u_n}{n} + \frac{u_r}{N} \qquad \text{puis} \qquad \limsup_{N\to\infty} \frac{u_N}{N} \leqslant \frac{u_n}{n}\ ,$$

et ce, pour tout entier n. Ainsi,

$$\limsup_{N\to\infty} \frac{u_N}{N} \leqslant \inf_{n\geqslant 1} \frac{u_n}{n} \leqslant \liminf_{N\to\infty} \frac{u_N}{N}\ ,$$

ce qui montre que

$$\frac{u_N}{N} \underset{N\to\infty}{\longrightarrow} \inf_{n\geqslant 1} \frac{u_n}{n}\ .$$

Exemple 34.4 On considère une chaîne de Markov X sur un espace d'états fini E, de matrice de transition Q irréductible. La chaîne est donc irréductible récurrente positive et admet une unique probabilité stationnaire π. On va voir que pour que le taux d'entropie soit défini, on n'a en fait pas besoin ici de supposer que X est stationnaire, i.e., que X_1 admet pour loi π. Pour tout $N \geqslant 1$, on note en effet p_N la densité de $\mathbb{P}_N$ par rapport à la mesure de comptage sur E^N et on a alors l'écriture

$$\begin{aligned} H(\mathbb{P}_{N+1}) &= -\sum_{x_1^{N+1}\in E^{N+1}} p_{N+1}\big(x_1^{N+1}\big)\log_2 p_{N+1}\big(x_1^{N+1}\big) \\ &= -\sum_{x_1^N\in E^N}\ \sum_{x_{N+1}\in E} \Big(p_N\big(x_1^N\big)\, Q(x_N, x_{N+1})\Big)\log_2\Big(p_N\big(x_1^N\big)\, Q(x_N, x_{N+1})\Big) \\ &= -\sum_{x_1^N\in E^N} p_N\big(x_1^N\big)\log_2 p_N\big(x_1^N\big) + \sum_{x_1^N\in E^N} p_N\big(x_1^N\big)\, h_Q(x_N) \end{aligned}$$

où l'on a défini pour tout $x \in E$,

$$h_Q(x) = -\sum_{y\in E} Q(x,y)\log_2 Q(x,y)\ .$$

Avec ces notations, on a ainsi prouvé la décomposition

$$H(\mathbb{P}_{N+1}) = H(\mathbb{P}_N) + \mathbb{E}\big[h_Q(X_N)\big] \ .$$

Par conséquent, pour tout entier $N \geqslant 2$,

$$H(\mathbb{P}_N) = H(\mathbb{P}_1) + \sum_{t=1}^{N-1} \mathbb{E}\big[h_Q(X_t)\big] \ .$$

La fonction h_Q étant bornée (voir figure 26.1) donc intégrable pour la probabilité stationnaire π, les théorèmes de type ergodique pour les chaînes de Markov (voir par exemple [MPB98, Théorème 4.25]) assurent la convergence

$$\frac{1}{N}\sum_{t=1}^{N} h_Q(X_t) \underset{N\to\infty}{\longrightarrow} \sum_{x\in E} \pi(x) h_Q(x) \qquad \mathbb{P}\text{–p.s.},$$

qui entraîne également une convergence dans $\mathbb{L}^1(\mathbb{P})$ par théorème de convergence dominée. On a ainsi en particulier

$$\frac{H(\mathbb{P}_N)}{N} = \frac{H(\mathbb{P}_1)}{N} + \frac{1}{N}\sum_{t=1}^{N-1} \mathbb{E}\big[h_Q(X_t)\big] \underset{N\to\infty}{\longrightarrow} H_\infty \overset{\text{not.}}{=} \sum_{x\in E} \pi(x) h_Q(x) \ .$$

En reprenant les calculs précédents, on remarque que la limite ainsi obtenue est une entropie conditionnelle : si $Y = (Y_1, Y_2, \ldots)$ est une chaîne de Markov stationnaire de transition Q et de loi initiale π, alors H_∞ est égale à l'entropie conditionnelle de la loi de (Y_1, Y_2) par rapport à la loi de Y_1.

34.1.2 Codes universels (un aperçu de la recherche en codage)

Au paragraphe 18.1, on avait parlé de deux situations menant à une modélisation stochastique naturelle, le cas du codage de données déjà disponibles et qu'on peut lire une première fois avant de les coder, d'une part, et d'autre part, celui de données émises de manière séquentielle par une source et qu'on doit coder à la volée par blocs de N caractères. Le chapitre 26 s'est essentiellement intéressé au premier cas et nous indiquons maintenant quel est le type de résultats établis pour le second modèle. Pour cela, on suppose avoir affaire à un processus stationnaire $X = (X_1, X_2, \ldots)$, pour lequel on reprend les notations du paragraphe 34.1.1.

Le Théorème 18.5 et la Proposition 34.3 assurent que lorsque $\mathbb{P}_\infty$ et donc tous les $\mathbb{P}_N$ sont connus, il existe une suite de codes $c_N^*(\mathbb{P}_N)$ construits sur les $\mathbb{P}_N$ et tels que les facteurs de compression vérifient

$$\gamma\big(c_N^*(\mathbb{P}_N), \mathbb{P}_N\big) \longrightarrow H_\infty \qquad \text{lorsque} \quad N \to \infty \ ,$$

ce qui est le résultat limite optimal. Ici, pour bien indiquer la dépendance des facteurs de compression en la loi $\mathbb{P}_\infty$, on les a indexés d'une part par le code considéré et d'autre part, par la loi des mots.

La question est de savoir si l'on peut se passer de la connaissance de $\mathbb{P}_\infty$ et faire asymptotiquement aussi bien que précédemment, i.e., exhiber une suite de codes $\widehat{c}_N$ ne requérant aucune connaissance préliminaire et telle que leurs facteurs de compression vérifient que pour toute loi stationnaire $\mathbb{P}_\infty$,

$$\gamma\big(\widehat{c}_N, \mathbb{P}_N\big) \longrightarrow H_\infty \qquad \text{lorsque} \quad N \to \infty \ .$$

Au vu des commentaires autour de la remarque 26.9, cela revient à exhiber une suite de probabilités $\widehat{\mathbb{Q}}_N$, chacune définie sur $\mathcal{A}^N$, telle que pour toute loi stationnaire $\mathbb{P}_\infty$,

$$\frac{1}{N}\, \mathcal{K}\Big(\widehat{\mathbb{Q}}_N, \mathbb{P}_N\Big) \longrightarrow 0 \qquad \text{lorsque} \quad N \to \infty \ .$$

On appelle code universel un code associé à une telle suite de probabilités.

Le fait remarquable est qu'il existe de tels codes. On peut même les construire explicitement et de manière efficace, par double mélange utilisant des arbres de contexte (voir par exemple [Cat04, chapitre 1] pour un résumé de plusieurs décennies de recherches à ce sujet).

34.2 Codes des simulations

34.2.1 Fonctions de conversion ANSI – chaînes de caractères

```
function A = str2ansi(E)
%
% Entrée :
%   E, une chaîne de caractères de '0' et de '1', dont les éléments sont
%      lus 8 par 8 ; sa longueur doit donc être multiple de 8
% Sortie :
%   A, un vecteur de nombres entre 0 et 255, chacun des nombres correspondant
%      à un groupe de 8 caractères '0' ou '1'

  A = bin2dec(reshape(E, 8, length(E)/8)');

end



function S = ansi2str(A)
%
% Entrée :
%   A, un vecteur de nombres entre 0 et 255
% Sortie :
```

```
%   S, une chaîne de caractères de '0' et de '1', dont les éléments sont
%      lus 8 par 8 ; chaque groupe de 8 correspondont à un élément de A

  x = dec2bin(A, 8)';
  S = num2str(x(:))';

end
```

34.2.2 Construction du code de Huffman et représentation de l'arbre associé

```
function T = HuffmanTree(M, P)
%
% Entrées :
%   M, objet de type cell, contenant les mots du dictionnaire
%   P, vecteur de nombres, contenant les probabilités *ou* les nombres
%      d'occurrences associés à chacun des mots (P n'a pas besoin d'être
%      renormalisé de sorte que ses composantes se somment à 1)
% Sortie :
%   T, un arbre, i.e., un objet de type cell contenant ou
%      des objets de type cell ou des mots

  [P, I] = sort(P);
  % S'il ne reste plus que deux éléments dans M,
  % alors l'arbre associé est M lui-même.
  if length(M) == 2
    T = M(I);
  % Sinon on utilise la description récursive du code
  % de Huffman
  else
    T = HuffmanTree({{M{I(1)}, M{I(2)}}, M{I(3:end)}}, [P(1)+P(2) P(3:end)]);
  end;

end
```

```
function R = drawTree(T, varargin)
%
% Entrée :
%   T, l'arbre à représenter, ne contenant que des mots d'une et une seule
%      lettre *ou* que des nombres entre 0 et 255 (on représentera alors
%      les caractères ANSI associés)
%   L'objet de type cell nommé varargin contient tous les éventuels
%   arguments optionnels ; c'est ainsi que l'on peut transmettre un
%   nombre variable d'arguments sous Matlab ; seuls les appels récursifs
%   utilisent ces arguments optionnels
```

```
% Sortie :
%   R, un vecteur de chaînes de caractères toutes de même longueur

  % Lorsque lancé par l'utilisateur, il n'y a pas encore d'arguments
  % optionnels, on les détermine en vue des appels récursifs
  if isempty(varargin)
    depth = profondeur(T);
    current = 0;
    varargin = {depth, current};
  end;

  if not(iscell(T))
    d = varargin{1} - varargin{2};
    S = repmat(' ', 1, 2^d-1);
    R = [S char(T) S;
         repmat(' ', d, 2^(d+1)-1)];
  else
    Rgauche = drawTree(T{1}, varargin{1}, varargin{2}+1);
    Rdroite = drawTree(T{2}, varargin{1}, varargin{2}+1);
    d = varargin{1} - varargin{2};
    R = [repmat(' ', 1, 2^d-1) '*' repmat(' ', 1, 2^d-1);
         Rgauche, repmat(' ', d, 1), Rdroite];
  end;

end

function depth = profondeur(T)
% Calcul récursif de la profondeur d'un arbre

  if not(iscell(T))
    depth = 0;
  else
    depth = max(profondeur(T{1}), profondeur(T{2})) + 1;
  end;

end
```

34.2.3 Fonction pour le codage et le décodage d'une chaîne de caractères

```
function [C, D] = convert(T)
%
% Entrée :
%   T, un arbre de mots
% Sorties :
%   C, un objet de type cell contenant la suite des codes
%   D, un objet de type cell contenant la suite des mots
```

```
  LR = ['0', '1'];
  C = {};
  D = {};
  for k = 1:2
    if iscell(T{k})
      [Cchild, Dchild] = convert(T{k});
    else
      Cchild = '';
      Dchild = {T{k}};
    end;
    C = [C strcat(LR(k), Cchild)];
    D = [D Dchild];
  end;

end




function D = decode(S, T)
%
% Entrées :
%   S, la chaîne des caractères à décoder
%   T, l'arbre de codage (contenant pour mots ou uniquement des chaînes
%      de caractères, ou uniquement des nombres)
% Sortie :
%   D, une chaîne de caractères ou un vecteur de nombres, contenant
%      la suite des mots décodés

  t = T;
  D = [];
  for i = 1:length(S)
    % Lire la chaîne de gauche à droite fait parcourir l'arbre
    t = t{1+str2num(S(i))};
    if not(iscell(t))
      % Chaque fois que l'on atteint une feuille, on note sa valeur
      D = [D t];
      t = T;
    end;
  end;

end
```

34.2.4 Fonction pour encoder et décoder un arbre de code au début d'un fichier

```
function [S, W] = encodeTree(T);
%
```

```
% Entrée :
%   T, l'arbre à encoder
% Sortie :
%   S, une chaîne de caractères contenant des 0 et des 1 et
%      encodant la structure de T
%   W, un objet de type cell dressant la liste des mots présents
%      aux feuilles, de gauche à droite

  if iscell(T)
    [Sleft, Wleft] = encodeTree(T{1});
    [Sright, Wright] = encodeTree(T{2});
    S = ['1' Sleft Sright];
    W = [Wleft Wright];
  else
    S = '0';
    W = { T };
  end;

end



function [T, C, W] = decodeTree(S, M);
%
% Entrées :
%   S, une chaîne de caractères '0' et '1' contenant en préfixe la
%      structure de l'arbre à décoder
%   M, un objet de type cell contenant au moins autant d'éléments que
%      l'arbre a de feuilles ; ses éléments, lus de gauche à droite, seront
%      les feuilles de l'arbre
% Sortie :
%   T, l'arbre correspondant à S et M
%   C, le reste de la chaîne S une fois le préfixe codant l'arbre lu
%   W, les éléments de M qui étaient en trop (dans le cas où M avait
%      plus d'éléments que T n'a de feuilles)

  if S(1) == '0'
    T = M{1};
    W = M(2:end);
    C = S(2:end);
  else
    [Tleft, Cright, Wright] = decodeTree(S(2:end), M);
    [Tright, C, W] = decodeTree(Cright, Wright);
    T = {Tleft, Tright};
  end;

end
```

34.2.5 Script de codage d'un fichier

```
clc, clear;
% Déterminer le fichier à coder
[fname, pname] = uigetfile('*.*', 'Choisissez le fichier à coder');

% Lire fichier (sous la forme du vecteur contenant
% la suite des caractères ANSI du fichier)
fid = fopen([pname fname]);
ansi = fread(fid, inf);
fclose(fid);

% Construction de la table des fréquences des codes ANSI,
% non renormalisées
freqtable = histc(ansi, -0.5 + 0:256);
% Matlab ajoutant la classe [255.5 -> infini] :
freqtable = freqtable(1:256);
garde = (freqtable >= 1);
freqtable = freqtable(garde);
% Calcul de l'entropie et indication d'un taux de compression prévisionnel
f = freqtable/sum(freqtable);
H = - sum(f .* log(f))/log(2);
disp(['Taux de codage attendu (hors encodage de la table) : entre ', ...
      num2str(100*H/8, 2), ' % et ', num2str(min(100,100*(H+1)/8), 2), ' %']);

% Pour obtenir l'arbre, en plus de ces nombres d'occurrences,
% on doit transmettre un objet de type cell, contenant
% les index des codes ANSI ; ce sont eux qui forment nos mots
X = 0:255;
M = num2cell(X(garde), 1);
T = HuffmanTree(M, freqtable');

% Pour coder (mais pas pour décoder), il faut convertir l'arbre
% en deux objets cell, contenant d'une part, les codes (dans C),
% et d'autre part, les mots correspondants (dans D)
[C, D] = convert(T);
s = cell(1,256);
s(cell2mat(D)+1) = C;
% La chaîne codant le fichier étant donné l'arbre T est alors :
R = strcat(s{ansi+1});

% On code maintenant l'arbre
[S, W] = encodeTree(T);
W = ansi2str([W{:}]);

% On met les codes bout à bout, et on ajoute autant de bits qu'il faut
% (maximum 7) pour que la longueur du tout soit multiple de 8
```

```
% On indique par les trois premiers bits combien on en a ajouté à la fin
E = [S W R];
r = mod(length(E) + 3, 8);
d = (8 - r) * (r > 0);
E = [num2str(dec2bin(d, 3)) E repmat('0',1,d)];

Final = str2ansi(E);

% Déterminer sous quel nom on veut sauvergarder le fichier encodé
[fname, pname] = uiputfile('*.huf', 'Enregistrer le fichier encodé sous...');
fid2 = fopen([pname fname], 'w');
fwrite(fid2, Final);
fclose(fid2);

disp(['Taux de codage final (y compris encodage de la table) : ', ...
    num2str(100*length(Final)/length(ansi),2), ' %']);
```

34.2.6 Script de décodage d'un fichier

```
clc, clear;
% Déterminer le fichier à décoder
[fname, pname] = uigetfile('*.huf', 'Choisissez le fichier à décoder');
fid = fopen([pname fname]);
ansi = fread(fid, inf);
fclose(fid);

% On convertit la suite des caractères ANSI en une chaîne
% de 0 et de 1
S = ansi2str(ansi);
% Les 3 premiers 0 ou 1 nous indiquent combien de caractères jeter à la fin
% de la chaîne
d = bin2dec(S(1:3));
S = S(4:(end-d));

% On parcourt une première fois le début de la chaîne pour
% identifier le préfixe correspondant au code de l'arbre T
% (de longueur n)
[T, C, W] = decodeTree(S, num2cell(0:255));
n = length(S) - length(C);
TreeCode = S(1:n);
% Il y a un mot par feuille, les feuilles sont codées par 0
numberWords = sum(TreeCode == '0');
% Les mots ont été placés à la suite de l'arbre
TreeWords = S((n+1):(n+8*numberWords));
M = str2ansi(TreeWords);
% On peut maintenant décoder l'arbre, puisqu'on sait la vraie étiquette
% de chaque feuille
```

```
T = decodeTree(S(1:n), num2cell(M));
disp(['Table lue, format de fichier correct -- ', ...
      'Décodage en cours, veuillez patienter']);

% Et il suffit de décoder le reste de la chaîne en utilisant l'arbre obtenu
Res = decode(S((n+8*numberWords+1):end), T);

% Déterminer sous quel nom on veut sauvergarder le fichier décodé
[fname, pname] = uiputfile('*.*', 'Enregistrer le fichier décodé sous...');
fid2 = fopen([pname fname], 'w');
fwrite(fid2, Res);
fclose(fid2);
```

CHAPITRE 35

Jeux de grattage (compléments)

RÉSUMÉ. Nous présentons ici le code des simulations informatiques relatives au chapitre 27.

35.1 Codes des simulations

35.1.1 Fonction auxiliaire de tirage d'un échantillon de tickets

```
function G = Ticket(T, n)
%
% Tirage du lot d'un ticket selon une loi multinomiale
%
% Entrée :
%   T un tableau de lots possibles
%   n taille de l'échantillon désiré
% Sortie :
%   G vecteur de lots de taille n
%

  Z = ceil(rand(1,n) * 500000);
  R = cumsum(T(:,2));
  G = zeros(1,n);
  for k = R'
    G = G + (Z > k);
  end
  J = (G <= 12);
  if any(J)
    G(J) = T(G(J)+1, 1)';
  end;
  G(G == 13) = 0;

end
```

35.1.2 Fichier maître de calculs de différentes valeurs numériques (probabilités et espérances) et de tracés de graphiques

```
T = [40000, 1;
     20000, 1;
```

```
     10000, 2;
      1000, 5;
       500, 18;
       200, 800;
       100, 850;
        50, 2020;
        20, 4000;
        10, 9000;
         6, 28000;
         4, 25000;
         3, 47500];

% Calcul du nombre L (= 7697) de lots significatifs
L = sum(T((T(:,1) >= 20),2));
p = L / 500000;

N = 500000;

% A propos des calculs autour de l'espérance des gains
% Pourcentage de redistribution
r = sum(T(:,1) .* T(:,2))/(N*3);
% Ecart-type du gain d'un ticket
s = sqrt(sum(T(:,1).^2 .* T(:,2))/N - (sum(T(:,1) .* T(:,2))/N)^2);
% Montant moyen et écart-type des gains typiques (i.e., inférieurs
% ou égaux à 200 euros)
J = (T(:,1) <= 200);
NT = N - sum(T(not(J),2));
mT = sum(T(J,1) .* T(J,2))/NT;
sT = sqrt(sum(T(J,1).^2 .* T(J,2))/NT - mT^2);
mA = sum(T(J,1) .* T(J,2))/N;
sA = sqrt(sum(T(J,1).^2 .* T(J,2))/N - mA^2);


% Fonction de répartition du montant du gain maximal
F = (1 - cumsum(T(:,2))/N).^100;
F = [F(end:-1:1); 1]; % sur 1 an
FF = F.^10; % sur 10 ans
% Fonctions de 'survie'
R = [1; 1-F(1:(end-1))]; % sur 1 an
RR = [1; 1-FF(1:(end-1))]; % sur 10 ans
RRR = [R RR];
for k = 1:2
  subplot(1,2,k)
  stem(RRR(:,k));
  % On effectue une nouvelle graduation de l'axe des abscisses, passant de
  % 0, 1, 2, 3, 4 ... à 0, 3, 4, 6, 10, 20, 50 ... (i.e., les gains
  % potentiels)
```

```
  set(gca, 'XTick', [1:length(RRR(:,k))])
  set(gca, 'XTickLabel', [0; T(end:-1:1, 1)]);
  % Une version modifiée de xticklabel_rotate90 (de Denis Gilbert, Ph.D.)
  % permet de mieux présenter la graduation des abscisses
  % en la tournant de 90 degrés
  % xticklabel_rotate90(get(gca,'XTick'), get(gca,'XTickLabel'));
  xlabel('Gain maximal');
  ylabel('Probabilité');
end;

% Calcul des probabilités qu'un joueur grattant deux tickets par
% semaine gagne au moins une fois une certaine somme
G = [F(1); F(2:end) - F(1:(end-1))];
GG = [FF(1); FF(2:end) - FF(1:(end-1))];
disp([ G GG ]);

% Tracés des six graphiques de soldes totaux des gains et mises
% de joueurs individuels, en fonction du temps
m = 400;
for k = 1:6
  subplot(2,3,k);
  hold on;
  plot(cumsum(Ticket(T, m)) - 3*(1:m), 'color', 'k');
  line([0 400], [0 0], 'color', 'k');
end

% Calculs relatifs à l'existence de stratégies gagnantes
% Montant moyen d'un lot significatif typique
JM = (T(:,1) <= 200) & (T(:,1) >= 20);
NM = sum(T(JM,2));
M = sum(T(JM,1) .* T(JM,2))/NM;
% Montant moyen gagné par un ticket perdant ou comportant
% une misère
Jm = (T(:,1) <= 10);
Nm = N - sum(T(:,2)) + sum(T(Jm,2));
m = sum(T(Jm,1) .* T(Jm,2))/Nm;
% Tentative de modélisation proposée dans le corrigé
a = 0.256; b = 0.718; q = 0.026;
e = []; r = [];
% Calcul de l'espérance de gain de la stratégie consistant
% en l'achat des x derniers tickets d'un carnet où aucun lot
% significatif n'est encore sorti
% Pour x >= 2
for x = (2:50)
  r = [r, a + b*x/50 + q*nchoosek(x,2)/nchoosek(50,2)];
  e = [e, (a*x*m + b*(x/50)*((x-1)*m+M) + ...
           q*(nchoosek(x,2)/nchoosek(50,2))*((x-2)*m+2*M))/r(end)];
```

```
end
% Pour x = 1
r = [a + b/50, r];
e = [(a*m + (b/50)*M)/(a + b/50), e];
% Et on enlève les mises (prix des tickets)
e = e - 3*(1:50);
subplot(1,2,1)
stem(e);
subplot(1,2,2)
stem(e(1:14));
% Si l'on tient compte de la répétition
gr = r.*e;
I = (gr > 0);
ind = 1:50;
disp(ind(I));
disp(gr(I));
```