TP DE PROBABILITÉS: SIMULATIONS ET ILLUSTRATION DES GRANDS THÉORÈMES

TABLE DES MATIÈRES

1. OBJECTIFS

— Savoir simuler des lois simples
— Illustrer et comprendre les implications pratiques de la loi des grands nombres et du théorème de la limite centrale

Dans tout ce TP, la seule fonction de génération aléatoire de Matlab qu'on s'autorisera à utiliser est la fonction `rand(n,m)` *qui renvoie une matrice de taille $n \times m$ contenant des nombres réels tirés selon la loi uniforme sur* $]0, 1[$.

2. SIMULATIONS

2.1. **Simulation de pile ou face.** On peut modéliser un jeu de pile ou face par une suite de variables aléatoires indépendantes identiquement distribuées $(X_k)_{k\in\mathbb{N}^*}$ à valeurs dans $\{0, 1\}$ telles que

$$P(\text{le } k\text{-ieme lancé donne 'pile'}) = P(X_k = 1) = p$$
$$P(\text{le } k\text{-ieme lancé donne 'face'}) = P(X_k = 0) = 1 - p$$

où p est un paramètre fixé dans $[0, 1]$. Pour tout k, les variables aléatoires X_k sont indépendantes et identiquement distribuées selon la loi de Bernoulli de paramètre p, on note $\forall k$, $X_k \sim \mathcal{B}(p)$.

(1) Quelle est la moyenne et la variance des X_k ?

(2) Que fait la fonction suivante

```
A=(rand(n,m)<p);
```

(3) Écrire une fonction `pileouface(n,p,m)` qui prend comme entrées les paramètres n (entier) et $0 \leq p \leq 1$ ainsi qu'un entier m, et qui donne en sortie une matrice $n \times m$ dont les coefficients simulent des variables aléatoires indépendantes qui suivent la loi de probabilité de Bernouilli $\mathcal{B}(p)$.

(4) Visualiser dans trois fenêtres graphiques séparées, pour $n = m = 100$ et $p = 0.5$, $p = 0.2$ et $p = 0.8$, un échantillon de 10000 tirages indépendants de loi de probabilité $\mathcal{B}(p)$ sur un diagramme en bâtons.

(5) Calculer la moyenne et la variance empiriques de ces échantillons et les comparer aux valeurs théoriques.

2.2. **Loi binômiale.** La première grandeur à laquelle on s'intéresse est le nombre de 'pile' obtenu lors de n tirages successifs (le nombre de 'face' s'en déduit). On introduit donc la variable aléatoire :

$$S_n = \sum_{k=1}^{n} X_k$$

(1) Quelle est la loi de S_n ? Rappeler en particulier $P(S_n = k)$. On note $S_n \sim \mathcal{B}(n, p)$

(2) Quelle est la moyenne et la variance de S_n ?

(3) Écrire une fonction `binomiale(n,p,m)` qui prend comme entrées les paramètres n (entier) et $p \in [0, 1]$ de la loi $\mathcal{B}(n, p)$, ainsi qu'un entier m, et qui donne en sortie un vecteur de longueur m dont les coefficients sont indépendants et suivent la loi $\mathcal{B}(n, p)$.

(4) Visualiser, pour $n = 10$ et $p = 0.5$, $p = 0.2$ et $p = 0.8$, un échantillon de 10000 tirages indépendants de loi $\mathcal{B}(n, p)$ sur un diagramme en bâtons.

(5) Calculer la moyenne et la variance empiriques de ces échantillons et les comparer aux valeurs théoriques.

2.3. **Loi géométrique.** On peut aussi s'intéresser, dans la suite des tirages $(X_k)_{k\in\mathbb{N}^*}$, au premier tirage donnant 'pile'. On considère donc :

$$Y = \inf\{k \in \mathbb{N}^* \text{ tel que } X_k = 1\} .$$

(1) Quelle est la loi de Y ? Rappeler en particulier $P(Y = k)$. On note $Y \sim \mathcal{G}(p)$.

(2) Quelle est la moyenne et la variance de Y ?

(3) Simuler un échantillon de loi géométrique et le visualiser dans un diagramme en bâtons.

(4) Calculer la moyenne et la variance empiriques de cet échantillon et les comparer aux valeurs théoriques.

2.4. Simulation d'un nombre au hasard sur un intervalle $]a,b[$.

(1) Écrire une fonction `uniforme(n,m,a,b)` qui prend comme entrées les paramètres n et m (entiers) et a et b (réels), et qui donne en sortie une matrice $n \times m$ dont les coefficients simulent des variables aléatoires indépendantes qui suivent la loi de probabilité uniforme $\mathcal{U}(]a,b[)$.

(2) Visualiser, pour $n = m = 100$, et un choix de a et b , un échantillon de 10000 tirages indépendants de loi de probabilité $\mathcal{U}(]a,b[)$ sur un histogramme.

2.5. Simulation d'un dé parfait.

(1) Écrire une fonction `de_parfait(n,m)` qui prend comme entrées les paramètres n et m (entiers) et qui donne en sortie une matrice $n \times m$ dont les coefficients simulent des variables aléatoires indépendantes qui suivent la loi de probabilité uniforme sur l'ensemble $\{1,2,3,4,5,6\}$.

(2) Visualiser, pour $n = m = 100$ un échantillon de 10000 tirages indépendants de cette loi sur un diagramme en bâtons de couleur rouge.

2.6. Simulation d'un dé quelconque.

(1) Écrire une fonction `de_quelconque(n,m,p1,p2,p3,p4,p5)` qui prend comme entrées les paramètres n et m (entiers), $p1$, $p2$, $p3$, $p4$, $p5$ entre 0 et 1 de somme plus petite que 1 et qui donne en sortie une matrice $n \times m$ dont les coefficients simulent une série de lancers indépendants du dé. On aura une probabilité $p1$ de tomber sur la 1ère face, $p2$ sur la 2ème etc. et $1 - (p1 + p2 + p3 + p4 + p5)$ de tomber sur la 6ème face.

(2) Visualiser, pour $n = m = 100$ un échantillon de 10000 lancers indépendants de ce dé sur un diagramme en bâtons de couleur verte.

3. Les grands théorèmes

3.1. Loi des grands nombres. On souhaite ici étudier le comportement asymptotique de $\frac{S_n}{n}$, c'est-à-dire l'évolution du nombre moyen d'apparitions de 'pile' dans n tirages quand n tend vers $+\infty$.

(1) Tester le script `LGN_pileface` suivant

```
% LGN_PILEFACE
% trace, pour 10 experiences successives,
% l'evolution du nombre de 'pile' obtenu parmi n tirages de Bernoulli de parametre p,
% renormalise par n, pour n variant de 1 a 300
p=input('entrer le parametre p : ');

for i=1:10
    disp('appuyer sur entree pour la simulation suivante...')
    pause;
    A=pileouface(1,p,300);
    V=cumsum(A);
    D=1:300;
    V=V./D;
    clf;
```

```
        hold on
        plot(V);
        W=p*ones(1,300);
        plot(W,'r');
        hold off
    end
```

(2) Modifier le script précédent pour qu'il permette de choisir le nombre *nmax* de simulations et la longueur L de la série de pile ou face, trace la valeur de p en rouge, et trace les courbes dans la même fenêtre graphique (commande `subplot`).

(3) On vient de voir une illustration d'un théorème fondamental : la loi des grands nombres. Dans ce cadre particulier, elle s'énonce dans les termes suivants :

Théorème 1 (Loi des grands nombres pour une variable de Bernoulli)**.** *Soit $(X_k)_{k\in\mathbb{N}^*}$ une suite de variables aléatoires indépendantes, identiquement distribuées suivant la loi de Bernoulli de paramètre p. Alors, presque-surement et dans L^1*

$$\frac{1}{n}\sum_{k=1}^{N} X_k \longrightarrow p$$

3.2. **Théorème de la limite centrale.** La vitesse de convergence dans le théorème précédent est lente (de l'ordre de $1/\sqrt{n}$). Le théorème suivant (De Moivre–Laplace) est un cas particulier du théorème de la limite centrale ; il permet de préciser la convergence, et justifie l'apparition de la 'courbe en cloche'.

Théorème 2 (De Moive–Laplace)**.** *Soit $(X_k)_{k\in\mathbb{N}^*}$ une suite de variables aléatoires indépendantes, identiquement distribuées suivant la loi de Bernoulli de paramètre p, alors*

$$P\left(a < \frac{S_n - np}{\sqrt{n}} < b\right) \longrightarrow \int_a^b g_\sigma(t)\,\mathtt{d}\,t\ ,$$

où $g_\sigma(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{t^2}{2\sigma^2}}$

(1) Que vaut σ^2 dans l'énoncé ci-dessus ?

(2) Étude de la gaussienne. Tracer, sur un même graphe, la courbe $t \mapsto g_\sigma(t)$ pour $\sigma = 0.5, 1$ et 2.

(3) Que vaut la moyenne et la variance de la loi de densité g_σ ?

(4) Écrire un script `tcl_pileface` qui demande à l'utilisateur la valeur du paramètre p, classe dans un histogramme 1000 réalisations aléatoires de $\frac{S_{400}-400p}{20}$, et affiche en surimpression sur l'histogramme , la courbe de la densité gaussienne g_σ avec la valeur convenable de σ.

ANNEXE : BONNES HABITUDES

Matlab est **optimisé** pour travailler sur les matrices. Il faut donc exploiter cette possibilité au maximum, et donc éviter les boucles `for` quand on peut, pour au moins deux raisons

(1) Le code est **beaucoup** plus lisible qu'avec des boucles `for`

(2) Le code est **beaucoup** plus performant !

Par exemple, si l'on veut savoir si les entrées d'une matrice A sont plus grandes qu'un nombre p, il suffit de faire

```
>> B = A > p ;
```

On obtient alors un résultat sous la forme d'une matrice B de la même taille que A, qui contient des 1 et des 0 : $B_{i,j} = 1 \Leftrightarrow A_{i,j} > p$ et $B_{i,j} = 0 \Leftrightarrow A_{i,j} \leq p$.

De même, si l'on veut accéder seulement aux éléments de A qui sont plus grand que p, il suffit de faire

```
>> A(A>p) = x ;
```

Ainsi, tous les éléments de A qui sont plus grand que p sont remplacés par la valeur x.