tpidla_ameliore

July 9, 2020

1 Introduction aux Processus Stochastiques (PRSTO)

P. Carmona

Pour avancer dans le notebook et exécuter les cellules il faut taper Shift+Enter ou utiliser la barre d'outils ci-dessus et choisir Cell, Run Cell and select Below

1.1 Consignes pour ce TP temps réel noté

Vous répondrez aux questions en modifiant ce notebook. En insérant des cellules de type Markdown pour le texte et des cellules de type code pour le code.

Ensuite vous sauvez ce notebook sous le nom Prenom_Nom_tpidla.ipynb et vous le déposez sur Moodle

1.2 La ruine du joueur

Soit S_n une marche aléatoire simple issue de x. On peut réaliser $S_n = x + \xi_1 + \cdots + \xi_n$ avec \$$(\xi i)\{i \geq 1\}$\$$iiddeloiP(\xi_1 = \pm 1) = 1/2$. Etant donnés des entiers $a \leq x \leq b$ on pose $T_x = \inf\{n \geq 0 : S_n = x\}$ et en considérant la martingale S_n arrêtée en $T = T_a \wedge T_b$ on montre que $E_x[S_T] = E_x[S_0]$ et on en déduit (ce résultat a été montré en cours et on ne vous demande pas de le redémontrer)

$$\mathbb{P}_x(T_a < T_b) = \frac{b-x}{b-a}.$$

Illustrer ce résultat par des simulations en traçant, pour $a = -10, b = 10$ une approximation de la fonction \$ x→ $\mathbb{P}$_$x(T_a < T_b)$\$.

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
import numpy.random as rnd
%matplotlib inline
import seaborn as sns
```

1.3 IDLA

On étudie le modèle de croissance/exploration suivant. Une particule part de l'origine et se comporte comme une marche aléatoire simple jusqu'à ce qu'elle rencontre une zone inexplorée, et la elle se fixe sur le premier site inexploré. Si A_n désigne l'ensemble aléatoire des sites de $\mathbb{Z}$ explorés au temps n, on a $A_0 = 0$ et $A_1 = 0, 1$ si la particule fait un premier pas à droite, $A_1 = -1, 0$ sinon. Ainsi étant donné A_n, on fait partir une marche aléatoire de 0 et on pose $T = \inf k \geq 1 : S_n \notin A_n$ et on pose $A_{n+1} = A_n \cup S_T$. On pose $A_n = G_n, D_n$ et on remarque que $D_n - G_n = n$ (pourquoi ?). L'objet de notre étude est $X_n = G_n + D_n$.

Ecrire une fonction spdla(n) qui prend pour paramètre l'instant n et renvoie la valeur de X_n obtenue par la simulation décrite précédemment. Attention: vous devez attendre le premier instant où la marche sort de l'intervalle A_n déjà exploré. Vous devez donc utiliser une boucle while (tant que).

[ ]:

```

```

En utilisant les résultats précédents sur la ruine du joueur on peut montrer que en fait $(X_n, n \in \mathbb{N})$ est une chaîne de Markov inhomogène i.e. une suite $(X_n)_{n\in\mathbb{N}}$ de variables aléatoires entières dont les lois sont données par la condition initiale $X_0 = 0$ p.s. et la récurrence

$$p_{n,i} = X_{n+1} = i + 1 \mid X_n = i = 1 - X_{n+1} = i - 1 \mid X_n = i = \frac{n + 2 - i}{2(n + 2)}$$

En utilisant la relation précédente écrire une fonction mhdla(h) qui prend pour paramètre l'instant n et renvoie la valeur de X_n obtenue. C'est donc un programme de simulation analogue à celui permettant de simuler une chaine de Markov homogène : la différence est que la chaîne est inhomogène, et que donc nous vous conseillons de coder d'abord une fonction pidla(n,i) qui calcule la probabilité $p_{n,i}$

On peut tirer une variable de Bernoulli de parametre p en écrivant

[4]:

```
p=0.3
rnd.binomial(1,p)
```

[4]: 1

Pour quelques de valeurs de n, allant de $n = 10$ à $n = 100$, (pourquoi ne prend on pas $n = 1000$?) générer un grand échantillon de taille N ($N = 500$ minimum) de variables aléatoires de loi X_n en utilisant les deux fonctions précédentes. Tracer des histogrammes de ces deux échantillons sur un même graphique. Comparer numériquement les moyennes empirique et variances empirique de ces échantillons. Commentez.

Comme superposer des histogrammes n'est pas évident nous vous conseillons d'utiliser des estimations de la densité (kernel density estimator).

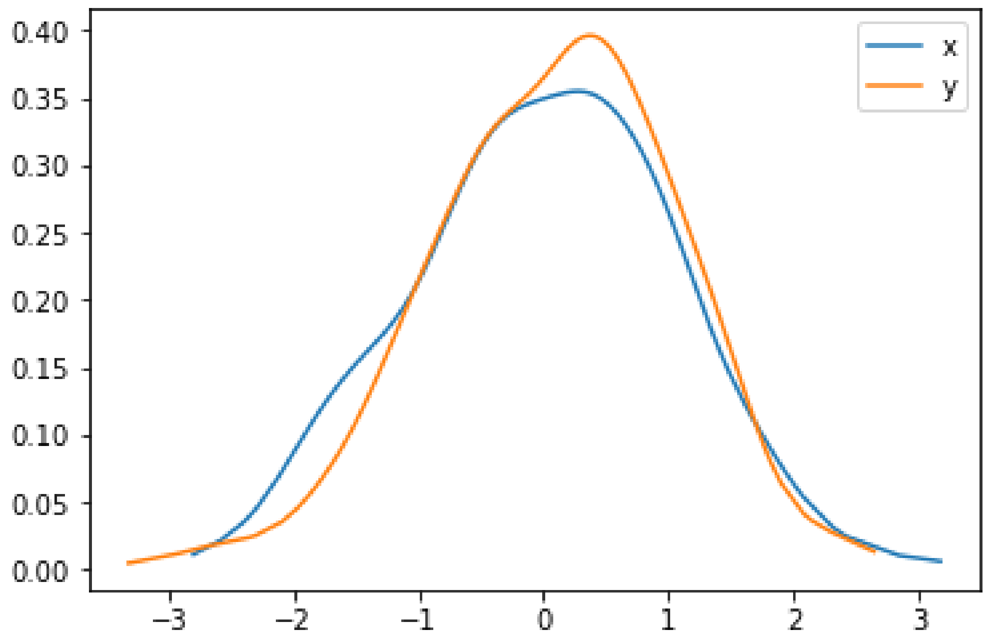
[12]:

```
from scipy.stats import kde

# create data
x = np.random.normal(size=500)
x.sort()
y = np.random.normal(size=500)
y.sort()
# Evaluate a gaussian kde
```

```
k = kde.gaussian_kde(x)
plt.plot(x,k(x),label="x")
kp = kde.gaussian_kde(y)
plt.plot(y,kp(y),label="y")
plt.legend()
```

[12]: <matplotlib.legend.Legend at 0x7f5e861c4e80>



1.4 IDLA Limite

On peut montrer (facilement par récurrence) que $X_n = 0$ et que si $a_n = X_n^2$, alors $a_{n+1} = 1 + \frac{n}{n+2} a_n$. On en déduit que $\lim \frac{a_n}{n} = 1/3$ (un exercice de prépa pas si évident qu'il n'y paraît).

On peut en fait montrer la convergence en loi (admise)

$$\frac{X_n}{\sqrt{n}} \xrightarrow[n \to \infty]{\mathcal{L}} \mathcal{N}(0, 1/3)\,.$$

Pour illustrer la convergence en loi, choisir n assez grand, fabriquer un N échantillon, avec N assez grand, de la loi de X_n,

$$X_n^{(1)}, X_n^{(2)}, \ldots, X_n^{(N)}$$

Comparer la moyenne, la variance de cet échantillon avec la moyenne et la variance de la loi cible.

Enfin, superposer l'histogramme de l'échantillon avec la densité de la loi cible.

[ ]: