TP Modélisation à événements discrets

Implémentation de lois de probabilité

TP à rendre

Un compte-rendu de TP est à rédiger à l'issue de ce TP. Il devra expliquer la démarche suivie pour la conception des programmes ainsi que les éventuelles difficultés rencontrées. Le compte-rendu (si possible sous forme PDF) ainsi que le code source des programmes devront être envoyés à l'adresse `Claude.Duvallet@univ-lehavre.fr` avec pour sujet "[M1INFO] TP MED 2". Au minimum, les parties suivantes sont obligatoires pour votre compte-rendu :

1. Rappel du sujet.
2. Présentation du programme réalisé.
3. Donner les réponses aux questions posées dans le sujet.
4. Résultats et tests (courbes).
5. La conclusion inclura les éventuelles difficultés rencontrées, les limites du programmes et les perspectives d'évolution du programme (s'il en existe) et tout ce qui vous semblera relever de cette conclusion.

L'ensemble de vos fichiers (et éventuels répertoires) devront être placés dans un répertoire nommé "MED-TP2-PrenomNom" puis compressés avant d'être joints au courriel que vous allez m'envoyer. Le format de compresssion peut être ZIP ou TAR.GZ. Attention de remplacer, Prenom et Nom, par les vôtres.

Exercice 1 : la loi Binomiale

La loi binomiale possède deux paramètres n et p. On réitère n une épreuve de Bernouilli de paramètre p (compris entre 0 et 1). Cette épreuve comporte deux résultats représentant soit un succès soit un échec. p représente la probabilité d'un succès alors que $q = 1 - p$ représente la probabilité d'un échec.
Soit X, la variable aléatoire qui indique alors le nombre de succès. Elle suit alors une loi de probabilité qui est de la forme suivante :

$$P(X = k) = \binom{n}{k} . p^k . q^{(n-k)} \tag{1}$$

où

$$\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!} \tag{2}$$

Il s'agit donc d'implémenter une classe `Binomial` comportant les méthodes suivantes :

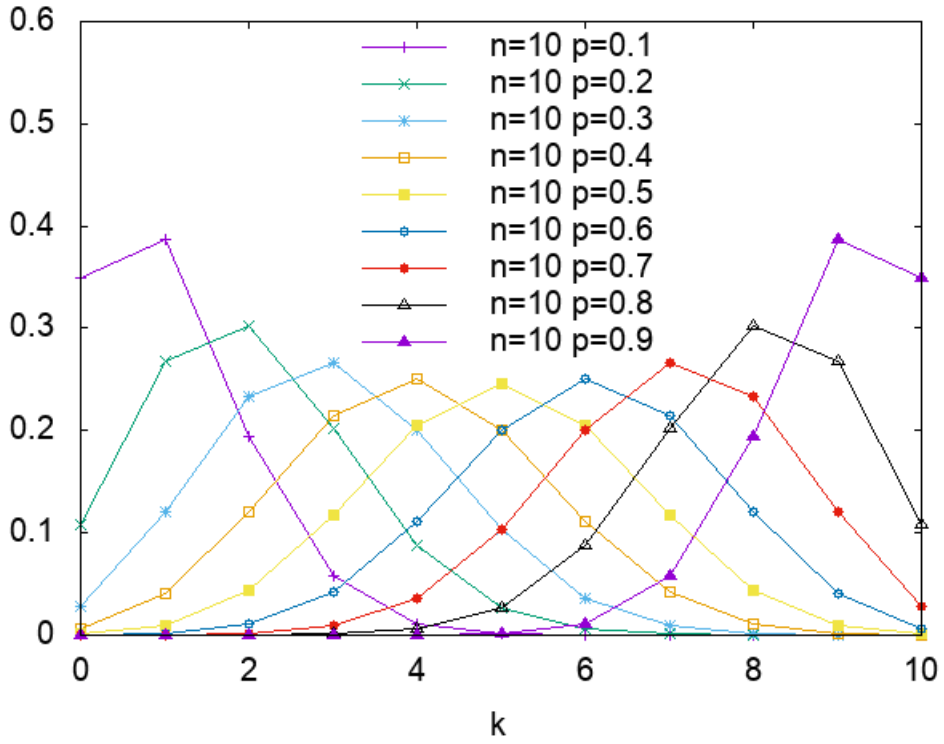
- Une méthode de calcul récursif de la factorielle. Quel est l'inconvénient de cette méthode en Java ?

- Une méthode permettant d'effectuer le calcul présenté dans l'équation (2). Cette méthode reposant sur la méthode de calcul factorielle, en donner les limites ?
- Une autre façon de calculer la combinaison est possible :

$$\binom{n}{k} = \begin{cases} 1 & \text{si } k == 0 \text{ ou } k == n \\ \binom{n-1}{k} + \binom{n-1}{k-1} & \text{sinon} \end{cases} \tag{3}$$

- Créer une méthode *compute* permettant de calculer la loi Binomiale avec pour paramètres n, p et k (cf. Equation (1)).
- Créer une méthode *generate* permettant de générer t lignes pour k variant de 0 à t (par pas de 1).
- Créer une méthode permettant d'enregistrer les données générées dans un fichier. Un fichier comportera les résultats obtenus (colonne 2) lorsque lorsque l'on fait varier la valeur de k (colonne 1).
- Dans la méthode principale, répéter la même expérience en faisant varier p de 0 à 1 par pas de 0.1.

À partir des différents fichiers de données générés, tracer les courbes suivantes au moyen de l'outil GnuPlot.

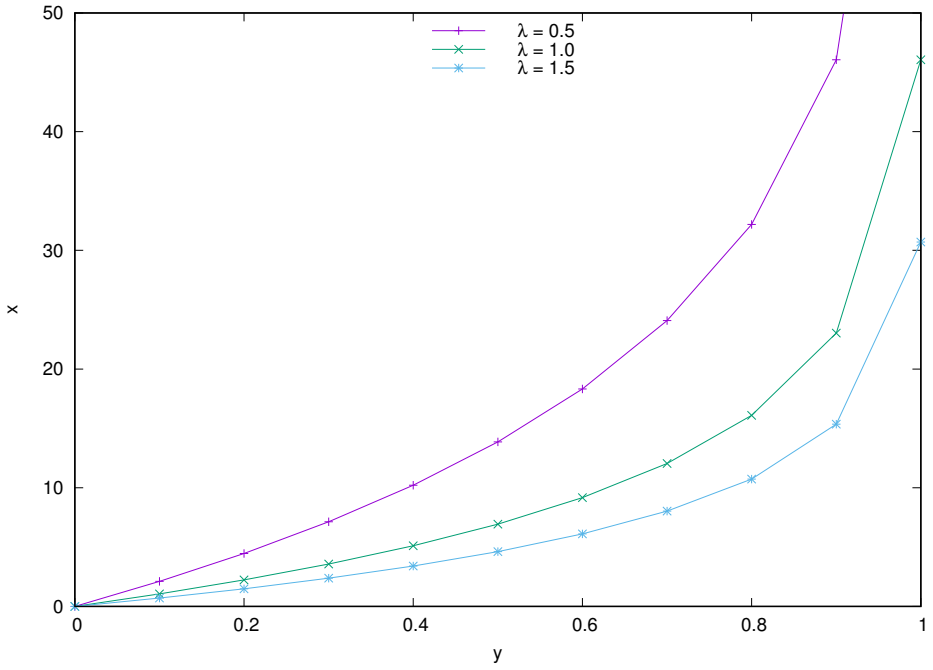


Exercice 2 : la loi exponentielle

Une variable aléatoire x obéit à une loi exponentielle de paramètre λ lorsque sa densité est $d(x) = \lambda \times e^{-\lambda x}$ pour $x \geq 0$. Elle a une fonction de répartition $f(x)$ telle que : $f(x) = 1 - e^{-\lambda x}$ pour x compris entre 0 et 1.
Nous faisons ensuite le lien avec la loi uniforme qui est à la base de l'implémentation du générateur de nombres aléatoires dans Java. On a l'équivalence suivante :

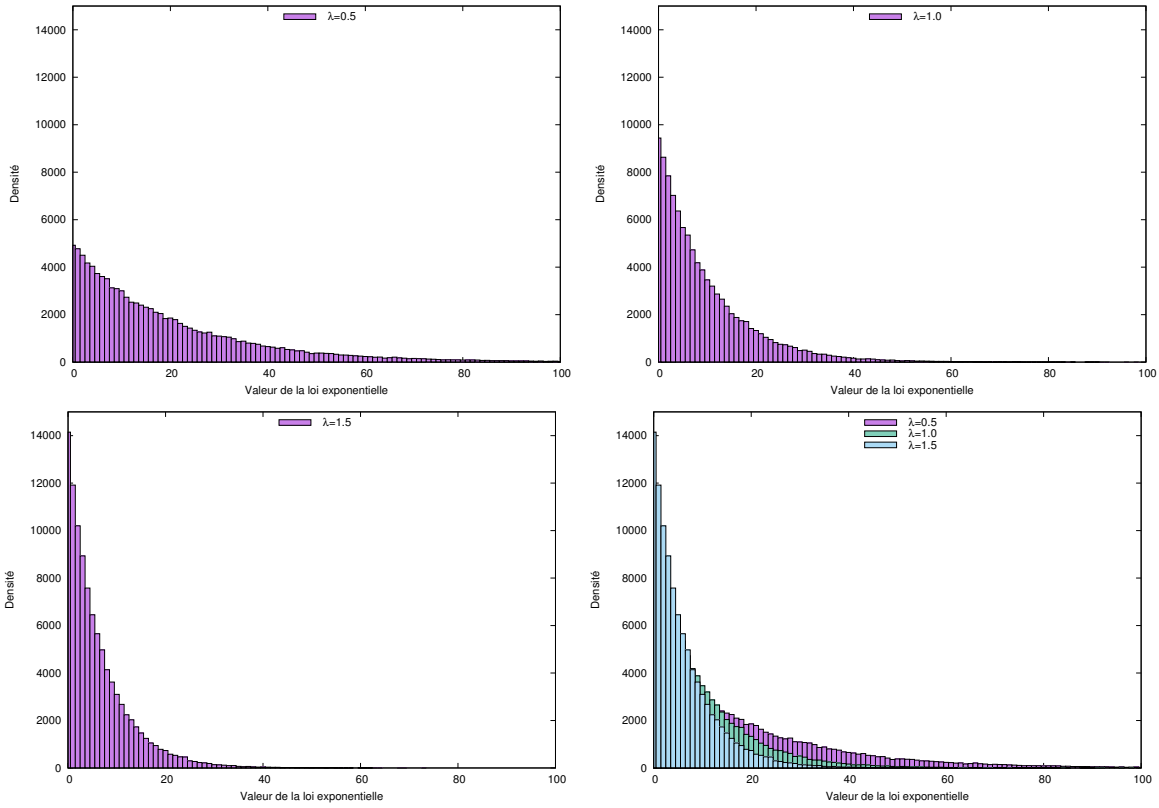
$$y = 1 - e^{-\lambda x} \longleftrightarrow x = -\frac{ln(1-y)}{\lambda} \tag{4}$$

Tracer les trois courbes de l'exponentielle pour y variant entre 0 et 1 par pas de 0.1 et pour $\lambda = 0.5$, $\lambda = 1.0$ et $\lambda = 1.5$. Il vous faudra inclure le graphique correspondant dans votre compte-rendu. Cela vous permettra de déterminer la valeur maximale pour la question suivante. Que remarquez vous pour y = 1.0 ? Si la valeur de y est supérieur ou égale à "1", on la raménera à 0.999. Quelle est la valeur maximale de la loi exponentielle ?



En se basant sur la loi uniforme (générateur aléatoire de Java), il s'agit de tracer les trois histogrammes suivants pour $\lambda = 0.5$, $\lambda = 1.0$ et $\lambda = 1.5$. En réalité, il s'agit de calculer l'exponentielle à partir d'un nombre y tiré aléatoirement suivant la loi uniforme. Il faut ensuite répéter un grand nombre de fois cette expérience et compter combien de fois on obtient la même valeur suivant la loi exponentielle. **Pour avoir une densité sur un plus grand nombre de valeurs, on multiplie l'exponentielle par 10**.

Exemple : On tire 100000 fois un nombre aléatoire suivant la loi uniforme. On calcul x pour ce nombre et on le multiplie par 10. Puis, on prends la partie entière du nombre. Combien de fois va-t-on obtenir 5 ? Il faut donc calculer le nombre d'occurences de chaque nombre.



Exercice 3 : la loi de Poisson

La loi de Poisson est aussi appelée loi des événements rares. Une variable X suit une loi de Poisson de paramètre λ si elle suit :

$$P(X = k) = e^{-\lambda}.\frac{\lambda^k}{k!} \tag{5}$$

Nous souhaitons simuler les arrivées dans une file d'attente suivant une loi de Poisson. Ce que nous souhaitons simuler, ce sont donc les instants d'arrivée. La probabilité d'avoir N arrivées pendant une durée D est alors formulée de la façon suivante :

$$P(N_D = k) \ = \ e^{-\lambda D}.\frac{(\lambda D)^k}{k!} \tag{6}$$

Soit t_1, t_2, ..., t_n les instants d'arrivée avec $t_1 < t_2 < ... < t_n$ et $N_D \in \{t_i \leq D\}$
Nous avons $t_{i+1} - t_i \sim exp(\lambda)$ **(loi exponentielle)**. Soit une variable aléatoire X $= t_{i+1} - t_i$ dont la fonction de répartition est :

$$F_x(D) \ = \ -\lambda D \ si \ x \geq 0 \tag{7}$$

$$F_x(D) = \begin{cases} -\lambda D & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases} \tag{8}$$

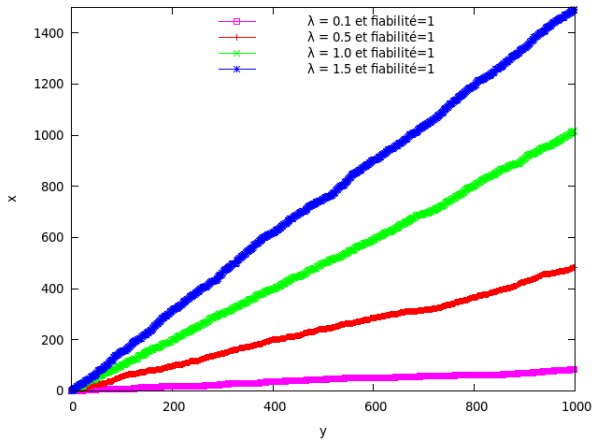
La fonction de répartition étant une fonction exponentionnelle, sa variable aléatoire X peut donc être générée par la transformation inverse. De cette façon, l'addition des variables exponentielles X donne un processus de Poisson stationnaire de moyenne λ sur un intervalle de temps D. Les

dates d'arrivées sont déterminées par la variable aléatoire X et les instants d'arrivées sont déduits par un calcul d'addition des variables aléatoires générées.

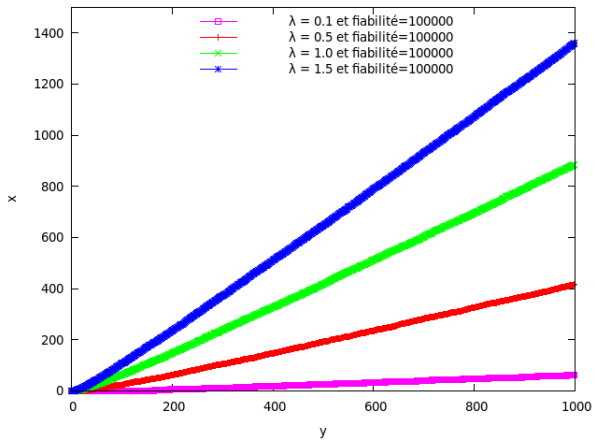
Construisez une fonction permettant de générer des dates d'arrivées sur une période données. Pour nous faciliter la tâche, nous utiliserons une approximation des dates d'arrivées générées sous forme d'un entier. Attention, on ne connaitra le nombre total d'arrivées qu'à l'issue de l'exécution de la fonction car les inter-arrivées étant aléatoires on peut donc avoir un nombre plus ou moins grand d'arrivées.

Sur une période D=1000, générer des dates d'arrivées pour λ=0.5, λ=1.0, λ=1.5.

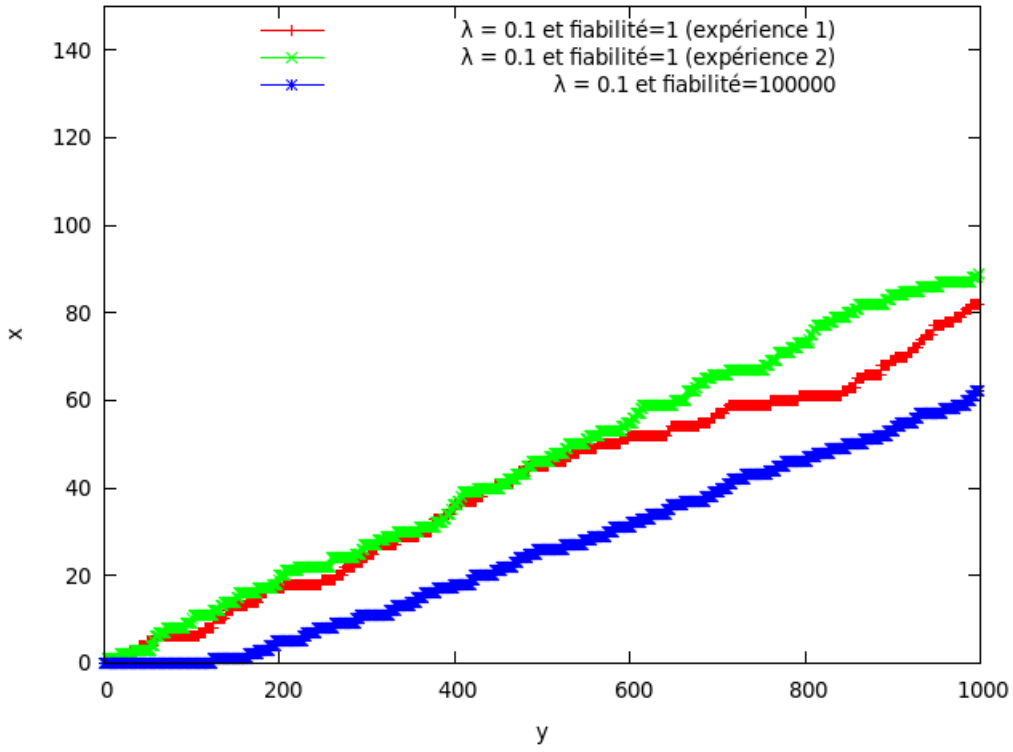
Pour obtenir, une certaine fiabilité statistique, il est conseillé de répéter un grand nombre de fois la génération des arrivées.



(a) Fiabilité = 1



(b) Fiabilité = 100000



(c) Comparaison entre les fiabilités