Correction du TP n° 1

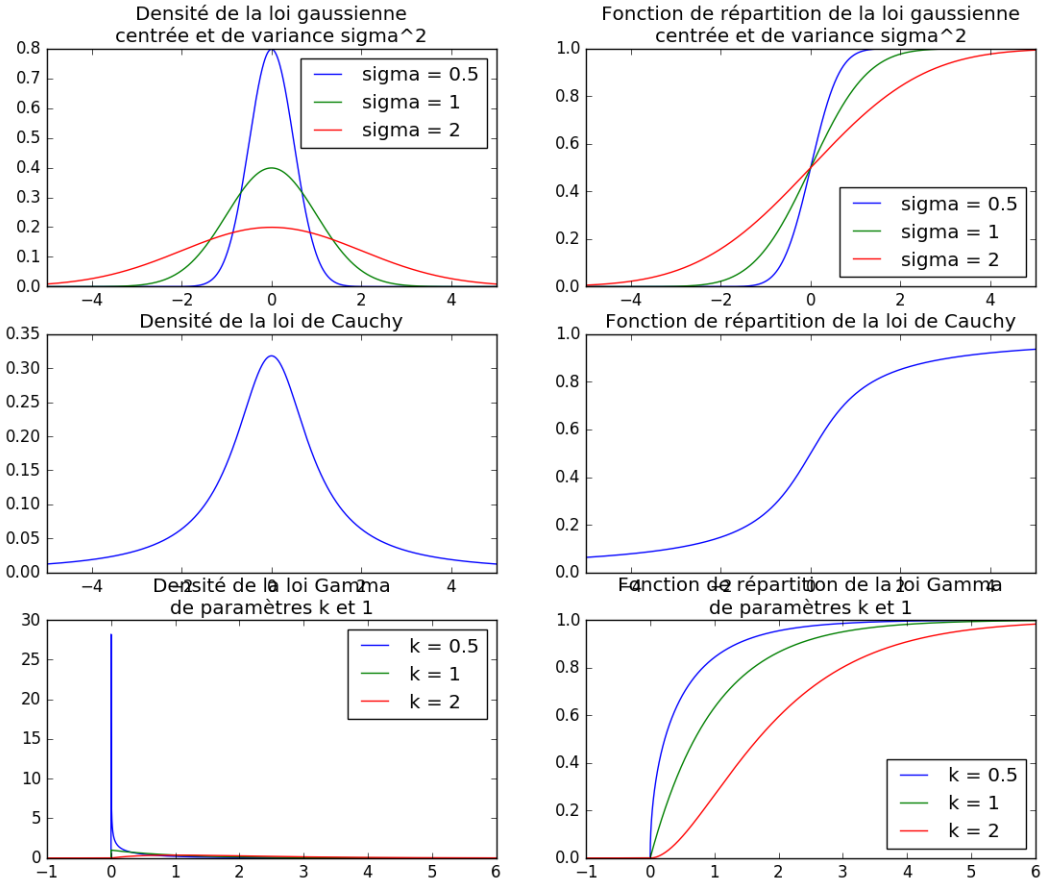
Exercice 1. Le but de cet exercice est de visualiser quelques lois continues importantes : les lois gaussiennes, Cauchy, et Gamma.

1. Nous souhaitons tracer les fonctions de densité et les fonctions de répartition des lois gaussiennes, Cauchy et Gamma. Pour cela, diviser la fenêtre en 6 cases (3 lignes et 2 colonnes) avec `plt.subplot`. Puis tracer (à l'aide de `plt.plot`) :
   - Dans la première ligne, la fonction de densité et la fonction de répartition de la loi gaussienne de moyenne nulle et écart-type $\sigma = 0.5$, 1 et 2 (superposer les trois tracés), sur l'intervalle $[-5, 5]$.
   - Dans la seconde ligne, la fonction de densité et la fonction de répartition de la loi de Cauchy, sur l'intervalle $[-5, 5]$.
   - Dans la troisième ligne, la fonction de densité et la fonction de répartition de la loi Gamma de premier paramètre $k = 0.5$, 1 et 2 et de deuxième paramètre 1 (superposer les trois tracés), sur l'intervalle $[0, 6]$.
2. Superposer sur l'intervalle $[-10, 10]$ les densités des lois gaussiennes standard et Cauchy. Comparer les deux : quels sont les points communs, quelles sont les différences ?
3. Superposer sur l'intervalle $[-5, 5]$ les densités des lois gaussiennes de moyenne nulle et d'écart-type 1, 0.5, 0.25, 0.1 et 0.05. Faire de même pour la fonction de répartition. Qu'observe-t-on quand l'écart-type tend vers zéro ?
4. *Bonus :* Superposer sur l'intervalle $[0, 6]$ les densités des lois Gamma de paramètres $\alpha = \beta = k$ pour $k \in \{1, 2, 4, 8, 16, 32, 64\}$. Qu'observe-t-on quand k tend vers l'infini ? Pour $k = 100$, superposer la densité avec la densité d'une autre loi que l'on connaît et qui y est proche.

Correction. *1.*

```
>>> plt.figure()
>>>
>>> plt.subplot(3,2,1)
>>> X = np.linspace(-5,5,10000)
>>> plt.plot(X,scs.norm.pdf(X,0,0.5))
>>> plt.plot(X,scs.norm.pdf(X,0,1))
>>> plt.plot(X,scs.norm.pdf(X,0,2))
>>> plt.title(u'Densité de la loi gaussienne centrée et de variance sigma^2')
>>> plt.legend([u'sigma = 0.5',u'sigma = 1',u'sigma = 2'])
>>>
>>> plt.subplot(3,2,2)
>>> plt.plot(X,scs.norm.cdf(X,0,0.5))
>>> plt.plot(X,scs.norm.cdf(X,0,1))
>>> plt.plot(X,scs.norm.cdf(X,0,2))
>>> plt.title(u'Fonction de répartition de la loi gaussienne
centrée et de variance sigma^2')
>>> plt.legend([u'sigma = 0.5',u'sigma = 1',u'sigma = 2'],loc='lower right')
>>>
>>> plt.subplot(3,2,3)
>>> plt.plot(X,scs.cauchy.pdf(X))
>>> plt.title(u'Densité de la loi de Cauchy')
>>>
>>> plt.subplot(3,2,4)
>>> plt.plot(X,scs.cauchy.cdf(X))
```

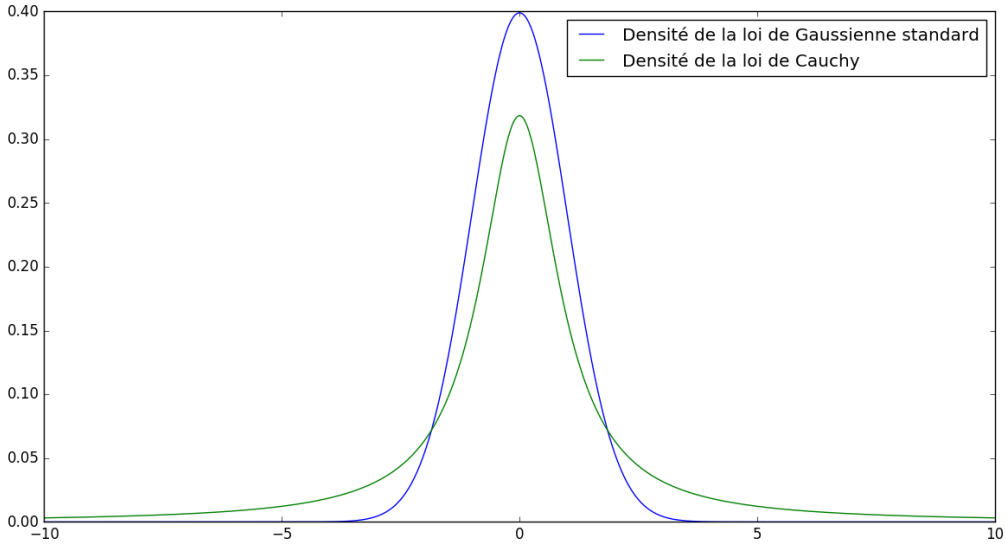
```
>>> plt.title(u'Fonction de répartition de la loi de Cauchy')
>>>
>>> plt.subplot(3,2,5)
>>> X = np.linspace(-1,6,10000)
>>> plt.plot(X,scs.gamma.pdf(X,0.5,0,1))
>>> plt.plot(X,scs.gamma.pdf(X,1,0,1))
>>> plt.plot(X,scs.gamma.pdf(X,2,0,1))
>>> plt.legend([u'k = 0.5',u'k = 1',u'k = 2'])
>>> plt.title(u'Densité de la loi Gamma de paramètres k et 1')
>>>
>>> plt.subplot(3,2,6)
>>> plt.plot(X,scs.gamma.cdf(X,0.5,0,1))
>>> plt.plot(X,scs.gamma.cdf(X,1,0,1))
>>> plt.plot(X,scs.gamma.cdf(X,2,0,1))
>>> plt.legend([u'k = 0.5',u'k = 1',u'k = 2'],loc='lower right')
>>> plt.title(u'Fonction de répartition de la loi Gamma de paramètres k et 1')
>>>
>>> plt.show()
```



2.

```
>>> plt.figure()
>>> X = np.linspace(-10,10,10000)
>>> plt.plot(X,scs.norm.pdf(X,0,1))
>>> plt.plot(X,scs.cauchy.pdf(X))
>>> plt.legend([u'Densité de la loi gaussienne standard',u'Densité
```

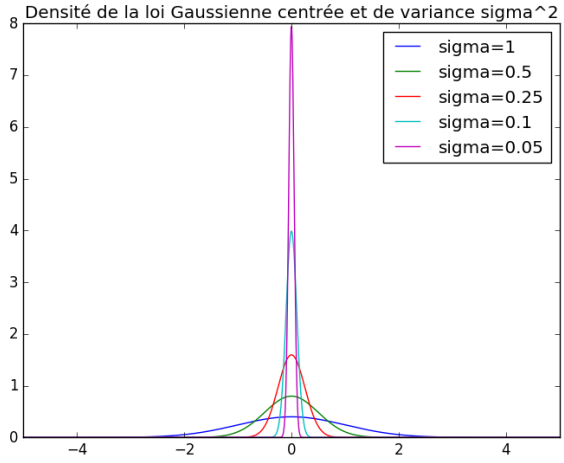
```
de la loi de Cauchy'])
>>> plt.show()
```



Les deux densités sont des fonctions symétriques. Néanmoins, la densité de la loi de Cauchy décroît beaucoup moins vite que celle de la loi gaussienne.

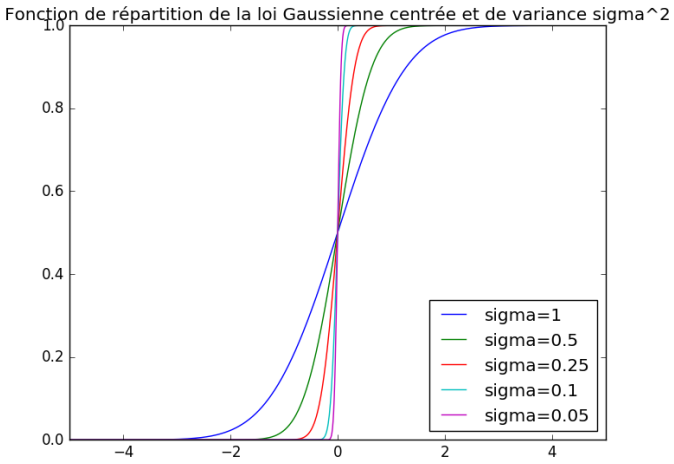
3.

```
>>> plt.figure()
>>> X = np.linspace(-5,5,10000)
>>> sigmas = [1,0.5,0.25,0.1,0.05]
>>> for s in sigmas:
...     plt.plot(X,scs.norm.pdf(X,0,s))
>>> plt.legend(['sigma='+str(s) for s in sigmas])
>>> plt.title(u'Densité de la loi Gaussienne centrée et de variance sigma^2')
>>> plt.show()
```




```
>>> plt.figure()
>>> for s in sigmas:
...     plt.plot(X,scs.norm.cdf(X,0,s))
>>> plt.legend(['sigma='+str(s) for s in sigmas],loc='lower right')
>>> plt.title(u'Fonction de répartition de la loi Gaussienne centrée
et de variance sigma^2')
```

```
>>> plt.xlim(-5,5)
>>> plt.show()
```



Quand l'écart type devient de plus en plus petit, on observe que la courbe de densité se concentre de plus en plus autour de 0. Cela était attendu puisqu'un petit écart type signifie que les valeurs typiques sont peu dispersées autour de la moyenne (0 dans le cas d'une Gaussienne centrée). On observe aussi cela avec les courbes des fonctions de répartition : quand l'écart type tend vers 0, la fonction de répartition tend à se rapprocher de la fonction $t \longmapsto \mathbb{1}_{[0,+\infty[}(t)$*, qui est la fonction de répartition d'une variable aléatoire constante égale à* 0*.*

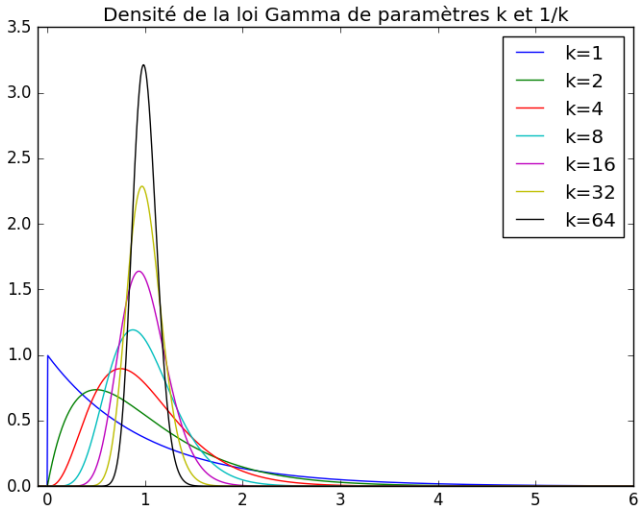
Notons Φ_σ *la fonction de répartition d'une loi Normale centrée et de variance* σ^2*. Soit* $t \in \mathbb{R}$*. A l'aide du changement de variable* $y = x/\sigma$*, nous avons*

$$\Phi_\sigma(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-x^2/(2\sigma^2)}\, dx = \int_{-\infty}^{t/\sigma} \frac{1}{\sqrt{2\pi}} e^{-y^2/2}\, dy = \Phi_1\left(\frac{t}{\sigma}\right) \underset{\sigma \downarrow 0}{\longrightarrow} \begin{cases} 0 & si \quad t < 0, \\ 1/2 & si \quad t = 0, \\ 1 & si \quad t > 0. \end{cases}$$

Ainsi, pour tout $t \neq 0$*,* $\Phi_\sigma(t)$ *tend vers* $\mathbb{1}_{[0,+\infty[}(t)$ *quand* σ *tend vers* 0*. On dit qu'une variable aléatoire de loi Normale centrée et de variance* σ^2 converge en loi *vers* 0 *(nous verrons cela prochainement en cours).*

4.

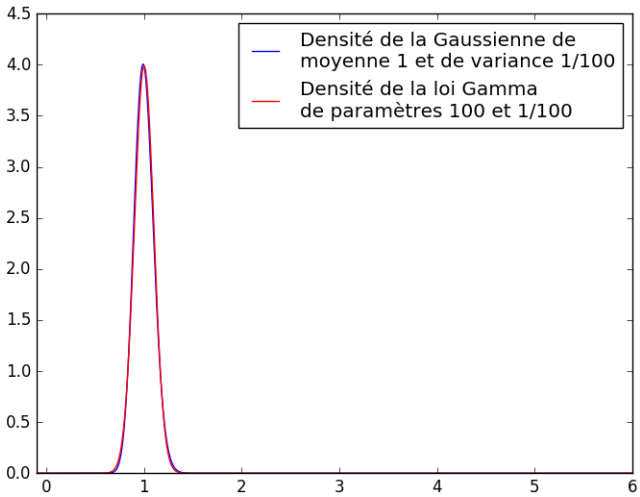
```
>>> plt.figure()
>>> X = np.linspace(-0.1,6,1000)
>>> for k in [1,2,4,8,16,32,64]:
...     plt.plot(X,scs.gamma.pdf(X,k,0,1./k))
...     plt.legend(['k='+str(k) for k in [1,2,4,8,16,32,64]])
>>> plt.title(u'Densité de la loi Gamma de paramètres k et 1/k')
>>> plt.show()
```



Quand k tend vers $+\infty$, on remarque la courbe de densité d'une loi Gamma de paramètres k et $1/k$ tend à se rapprocher d'une courbe en cloche, caractéristique de la courbe de densité d'une loi Gaussienne.

Une variable aléatoire de loi Gamma de paramètres k et $1/k$ admet pour espérance 1 et pour variance $1/k$ (cf. TD 4). Comparons la densité d'une loi Gamma de paramètres 100 et $1/100$ avec la densité d'une loi Gaussienne de moyenne 1 et de variance $1/100$:

```
>>> plt.figure()
>>> X = np.linspace(-0.1,6,1000)
>>> plt.plot(X,scs.gamma.pdf(X,100,0,0.01))
>>> plt.plot(X,scs.norm.pdf(X,1,0.1),color='r')
>>> plt.legend([u'Densité de la Gaussienne\ncentrée et de variance 1/100',
u'Densité de la loi Gamma\nde paramètres 100 et 1/100'])
>>> plt.show()
```



Exercice 2. Le but de cet exercice est de visualiser quelques lois discrètes importantes : les lois de Poisson, binomiale et géométrique.

1. Comme dans l'Exercice 1.1, diviser la fenêtre en 6 cases (3 lignes et 2 colonnes) et tracer :
   — Dans la première ligne, la fonction de masse (avec `plt.stem`) et la fonction de répartition (avec `plt.step`) de la loi de Poisson de paramètre $\lambda = 0.5$, 2 et 5 (superposer les trois tracés), sur l'intervalle $[0, 10]$.

— Dans la deuxième ligne, la fonction de masse et la fonction de répartition de la loi binomiale de paramètres $n = 20$ et $p = 0.2$, 0.5, 0.8 (superposer les trois tracés), sur l'intervalle $[0, 20]$. Utiliser `scs.binom`.

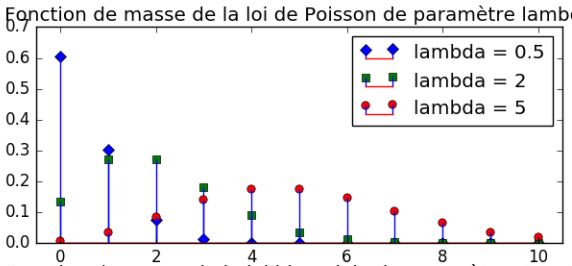
— Dans la troisième ligne, la fonction de masse et la fonction de répartition de la loi géométrique de paramètre $p = 0.7, 0.5, 0.2$ (superposer les trois tracés), sur l'intervalle $[1, 10]$. Utiliser `scs.geom`.

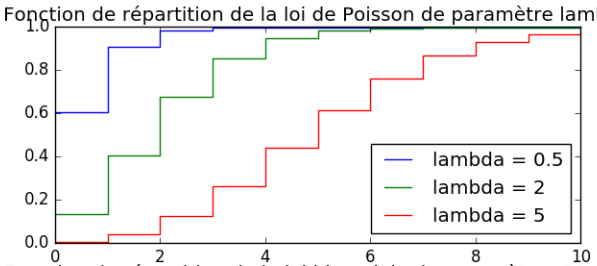
2. Superposer sur l'intervalle $[0, 10]$ les fonctions de masse des lois binomiales de paramètres n et $p = 1/n$ pour $n \in \{10, 20, 40\}$. Faire de même pour la fonction de répartition. Qu'observe-t-on quand n tend vers l'infini ? Peut-on approcher la loi par une autre loi discrète que l'on connaît ?
3. Répéter la partie précédente avec $n = 10$, 30, 100 et $p = 0.5$, sur l'intervalle $[0, 100]$. Peut-on cette fois-ci approcher la loi par une loi continue ?

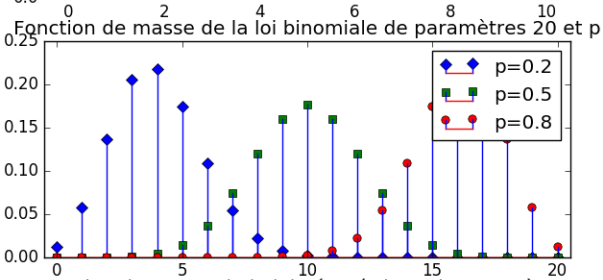
Correction. *1.*

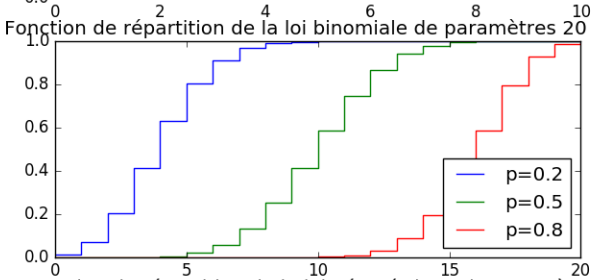
```
>>> plt.figure()
>>>
>>> plt.subplot(3,2,1)
>>> X = range(0,11)
>>> plt.stem(X,scs.poisson.pmf(X,0.5),markerfmt='bD')
>>> plt.stem(X,scs.poisson.pmf(X,2),markerfmt='gs')
>>> plt.stem(X,scs.poisson.pmf(X,5),markerfmt='ro')
>>> plt.legend([u'lambda = 0.5',u'lambda = 2',u'lambda = 5'])
>>> plt.title(u'Fonction de masse de la loi de Poisson de paramètre lambda')
>>> plt.xlim(-0.5,10.5)
>>>
>>> plt.subplot(3,2,2)
>>> plt.step(X,scs.poisson.cdf(X,0.5), where='post')
>>> plt.step(X,scs.poisson.cdf(X,2), where='post')
>>> plt.step(X,scs.poisson.cdf(X,5), where='post')
>>> plt.legend([u'lambda = 0.5',u'lambda = 2',u'lambda = 5'],loc='lower right')
>>> plt.title(u'Fonction de répartition de la loi
de Poisson de paramètre lambda')
>>>
>>> plt.subplot(3,2,3)
>>> X = range(0,21)
>>> plt.stem(X,scs.binom.pmf(X,20,0.2),markerfmt='bD')
>>> plt.stem(X,scs.binom.pmf(X,20,0.5),markerfmt='gs')
>>> plt.stem(X,scs.binom.pmf(X,20,0.8),markerfmt='ro')
>>> plt.legend([u'p=0.2',u'p=0.5',u'p=0.8'])
>>> plt.title(u'Fonction de masse de la loi binomiale de paramètres 20 et p')
>>> plt.xlim(-0.5,20.5)
>>>
>>> plt.subplot(3,2,4)
>>> plt.step(X,scs.binom.cdf(X,20,0.2), where='post')
>>> plt.step(X,scs.binom.cdf(X,20,0.5), where='post')
>>> plt.step(X,scs.binom.cdf(X,20,0.8), where='post')
>>> plt.legend([u'p=0.2',u'p=0.5',u'p=0.8'],loc='lower right')
>>> plt.title(u'Fonction de répartition de la loi binomiale
 de paramètres 20 et p')
>>>
>>> plt.subplot(3,2,5)
>>> X = range(0,11)
>>> plt.stem(X,scs.geom.pmf(X,0.7),markerfmt='bD')
>>> plt.stem(X,scs.geom.pmf(X,0.5),markerfmt='gs')
>>> plt.stem(X,scs.geom.pmf(X,0.2),markerfmt='ro')
>>> plt.legend([u'p = 0.7',u'p = 0.5',u'p = 0.2'])
>>> plt.title(u'Fonction de masse de la loi géométrique de paramètre p')
```

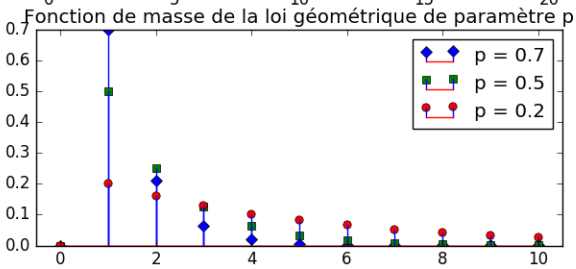
```
>>> plt.xlim(-0.5,10.5)
>>>
>>>
>>> plt.subplot(3,2,6)
>>> plt.step(X,scs.geom.cdf(X,0.7), where='post')
>>> plt.step(X,scs.geom.cdf(X,0.5), where='post')
>>> plt.step(X,scs.geom.cdf(X,0.2), where='post')
>>> plt.legend([u'p = 0.7',u'p = 0.5',u'p = 0.2'],loc='lower right')
>>> plt.title(u'Fonction de répartition de la loi géométrique de paramètre p')
>>>
>>> plt.show()
```

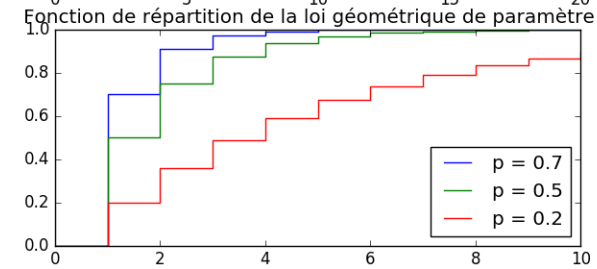






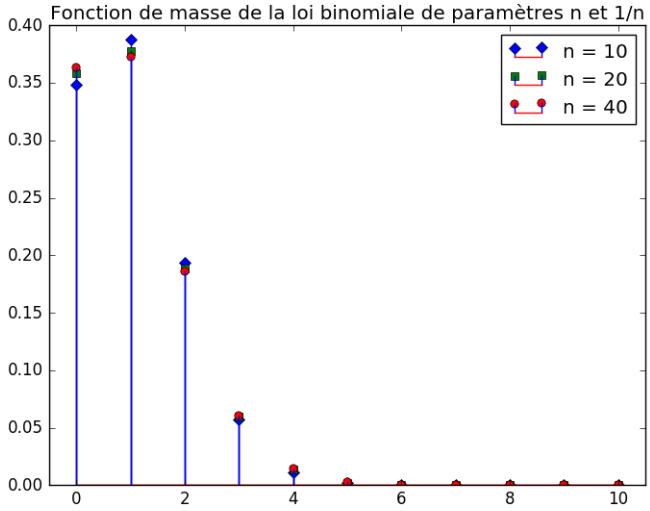




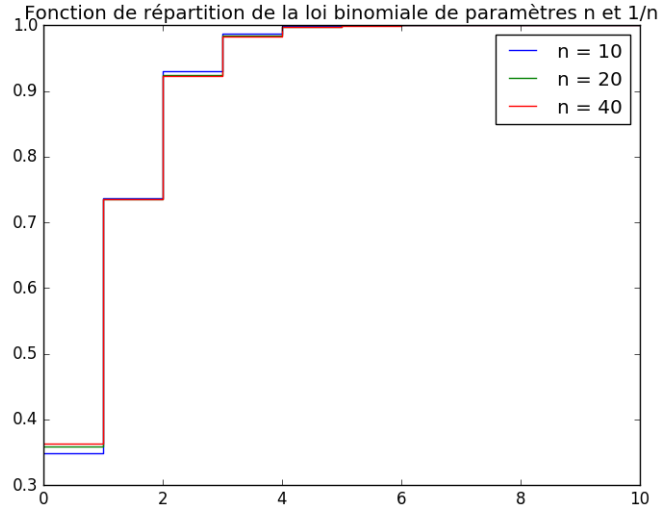


2.

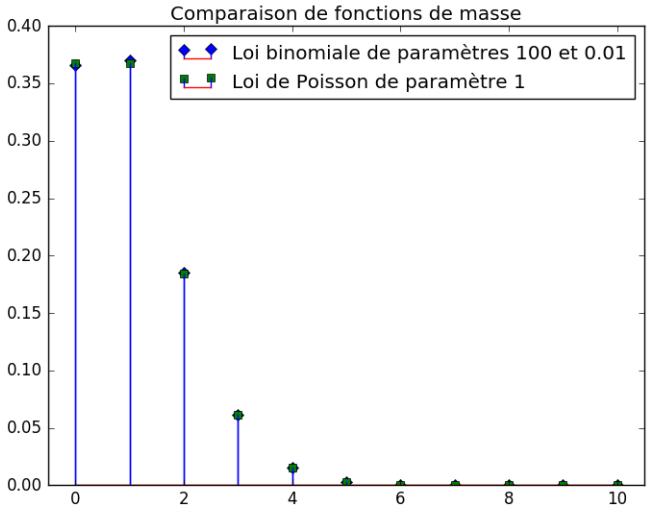
```
>>> plt.figure()
>>> X = range(0,11)
>>> plt.stem(X,scs.binom.pmf(X,10,1./10),markerfmt='bD')
>>> plt.stem(X,scs.binom.pmf(X,20,1./20),markerfmt='gs')
>>> plt.stem(X,scs.binom.pmf(X,40,1./40),markerfmt='ro')
>>> plt.title(u'Fonction de masse de la loi binomiale de paramètres n et 1/n')
>>> plt.legend([u'n = 10',u'n = 20',u'n = 40'])
>>> plt.xlim(-0.5,10.5)
>>> plt.show()
```




```
>>> plt.figure()
>>> plt.step(X,scs.binom.cdf(X,10,1./10), where='post')
>>> plt.step(X,scs.binom.cdf(X,20,1./20), where='post')
>>> plt.step(X,scs.binom.cdf(X,40,1./40), where='post')
>>> plt.title(u'Fonction de répartition de la loi binomiale
de paramètres n et 1/n')
>>> plt.legend([u'n = 10',u'n = 20',u'n = 40'])
>>> plt.show()
```



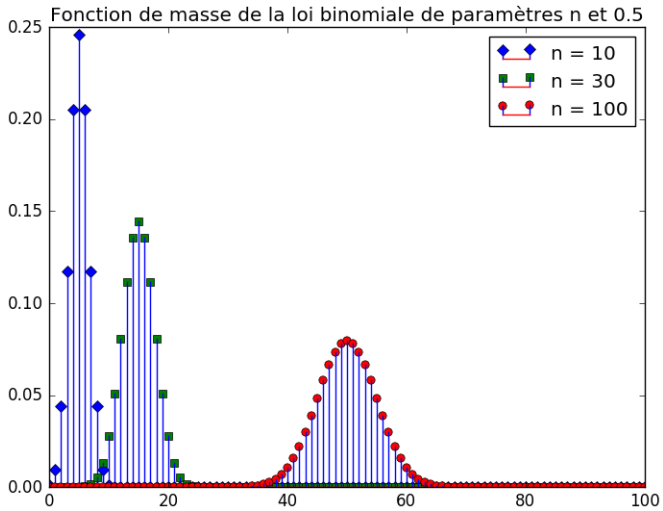

```
>>> plt.figure()
>>> plt.stem(X,scs.binom.pmf(X,100,1./100),markerfmt='bD')
>>> plt.stem(X,scs.poisson.pmf(X,1),markerfmt='gs')
>>> plt.title(u'Comparaison de fonctions de masse')
>>> plt.legend([u'Loi binomiale de paramètres 100 et 0.01',
u'Loi de Poisson de paramètre 1'])
>>> plt.xlim(-0.5,10.5)
>>> plt.show()
```



On voit que lorsque n croît, les fonctions de masse et fonctions de répartition de la loi binomiale de paramètres n et $p = 1/n$ semblent converger vers une limite. On identifie cette limite comme étant les fonctions de masse et la fonction de répartition de la loi de Poisson de paramètre 1.

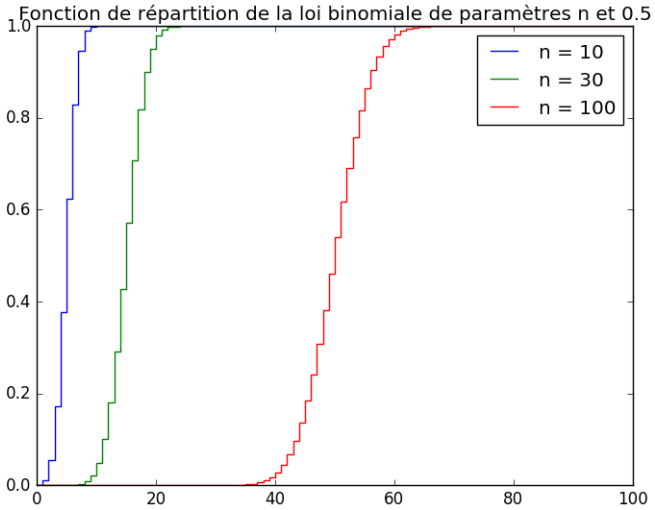
3.

```
>>> plt.figure()
>>> X = range(0,101)
>>> plt.stem(X,scs.binom.pmf(X,10,0.5),markerfmt='bD')
>>> plt.stem(X,scs.binom.pmf(X,30,0.5),markerfmt='gs')
>>> plt.stem(X,scs.binom.pmf(X,100,0.5),markerfmt='ro')
>>> plt.title(u'Fonction de masse de la loi binomiale de
paramètres n et 0.5')
>>> plt.legend([u'n = 10',u'n = 30',u'n = 100'])
>>> plt.show()
```

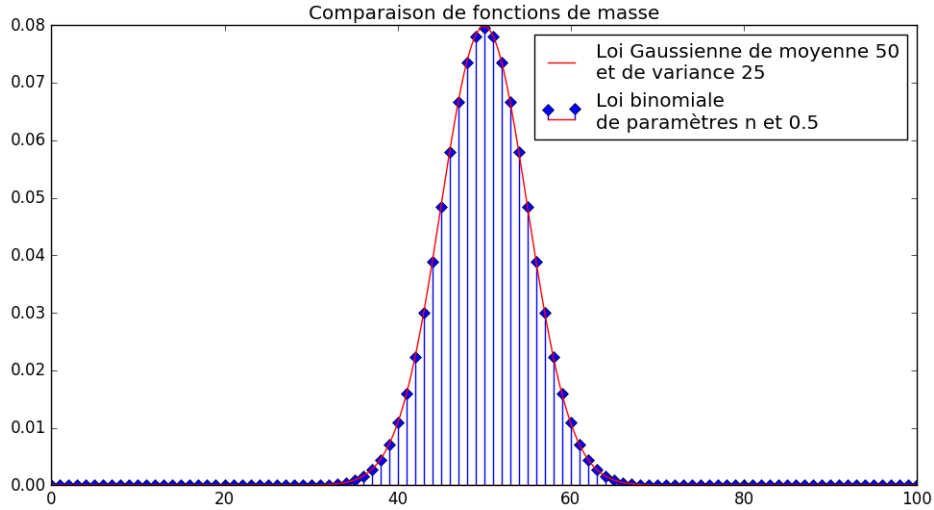



```
>>> plt.figure()
>>> plt.step(X,scs.binom.cdf(X,10,0.5), where='post')
>>> plt.step(X,scs.binom.cdf(X,30,0.5), where='post')
>>> plt.step(X,scs.binom.cdf(X,100,0.5), where='post')
>>> plt.title(u'Fonction de répartition de la loi binomiale
de paramètres n et 0.5')
```

```
>>> plt.legend([u'n = 10',u'n = 30',u'n = 100'])
>>> plt.show()
```




```
>>> plt.figure()
>>> plt.stem(X,scs.binom.pmf(X,100,0.5),markerfmt='bD')
>>> Y = np.linspace(0,100,500)
>>> plt.plot(Y,scs.norm.pdf(Y,50,5),c='r')
>>> plt.title(u'Comparaison de fonctions de masse')
>>> plt.legend([u'Loi Gaussienne de moyenne 50\net de variance 25',
u'Loi binomiale\nde paramètres n et 0.5'])
>>> plt.show()
```



Quand $p = 0.5$ est fixé, la fonction de masse de la loi binomiale de paramètres n et p semble approcher la densité de la loi gaussienne de moyenne $n/2$ et variance $n/4$, ce qu'on voit bien déjà avec $n = 100$.

Exercice 3. (Analyse d'échantillon) Marie rapporte d'une randonnée en montagne un bloc de pierre contenant des substances radioactives qui émettent des rayons α. Sa mère, chimiste à l'Université Paul Sabatier, lui prête un compteur Geiger pour étudier la radioactivité de cette pierre. Marie branche alors le compteur à son ordinateur et enregistre les espaces de temps entre deux rayons α. Elle remarque que ces espaces de temps ont l'air aléatoires. Aidez-la à découvrir la loi sous-jacente.

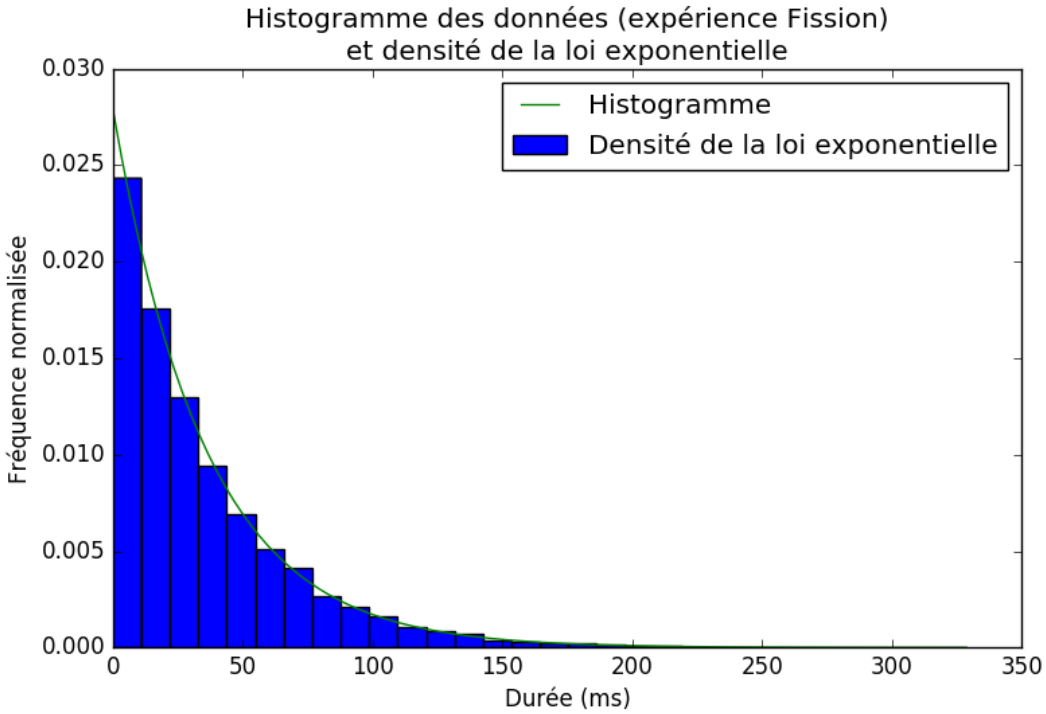
1. Charger le contenu du fichier `TP1_Fission.txt` dans un vecteur `Fission`. Il contient les espaces de temps (en ms) entre deux rayons α. Afficher la taille de l'échantillon, moyenne, quartiles, écart interquartile, écart-type, maximum et minimum.
2. Diviser la fenêtre en deux parties et afficher dans la première un histogramme de l'échantillon avec 30 classes (avec `plt.hist`), et dans la deuxième la fonction de répartition empirique de l'échantillon (avec `plt.step`).
3. Emettre une conjecture sur la loi de l'espace de temps entre deux rayons α. Superposer la densité de cette loi à l'histogramme normalisé. Superposer la fonction de répartition de cette loi à la fonction de répartition empirique.

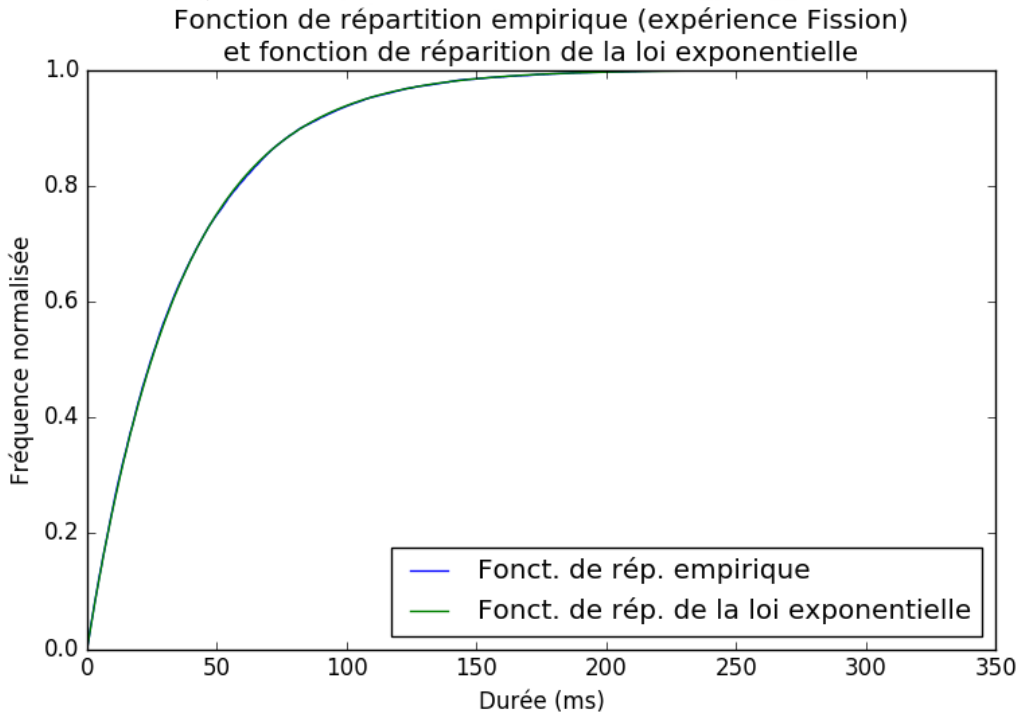
Correction. *1.*

```
>>> Fission = np.loadtxt(u'TP2_Fission.txt')
>>> print 'Taille de l\'échantillon : ', np.size(Fission)
>>> print 'Moyenne : ', np.mean(Fission)
>>> print 'Maximum : ', np.max(Fission)
>>> print 'Minimum : ', np.min(Fission)
Taille de l'échantillon :  20000
Moyenne :  35.915045953
Maximum :  328.84022184
Minimum :  0.00236665372508
```

2-3. La fonction de répartition empirique semble approcher celle d'une loi continue, de plus l'histogramme ressemble à la densité de la loi exponentielle d'une certaine moyenne μ. Comment deviner μ ? Il est naturel de la choisir comme étant la moyenne empirique de l'échantillon que nous avons calculée ci-dessous. Cela donne le code suivant :

```
>>> mu = np.mean(Fission)
>>>
>>> plt.figure()
>>>
>>> plt.subplot(2,1,1)
>>> X = np.linspace(0,np.max(Fission),100)
>>> plt.hist(Fission,bins=30,normed=True)
>>> plt.plot(X,scs.expon.pdf(X,0,mu))
>>> plt.xlabel(u'Durée (ms)')
>>> plt.ylabel(u'Fréquence normalisée')
>>> plt.title(u'Histogramme des données (expérience Fission)\n
et densité de la loi exponentielle')
>>> plt.legend([u'Histogramme', u'Densité de la loi exponentielle'])
>>>
>>> plt.subplot(2,1,2)
>>> plt.step(np.sort(Fission),np.arange(0,1,1./len(Fission)))
>>> plt.plot(X,scs.expon.cdf(X,0,mu))
>>> plt.xlabel(u'Durée (ms)')
>>> plt.ylabel(u'Fréquence normalisée')
>>> plt.title(u'Fonction de répartition empirique (expérience Fission)\n
et fonction de réparition de la loi exponentielle')
>>> plt.legend([u'Fonct. de rép. empirique', u'Fonct. de rép. de la loi
exponentielle'],loc='lower right')
>>>
>>> plt.show()
```





Exercice 4. (Bonus)

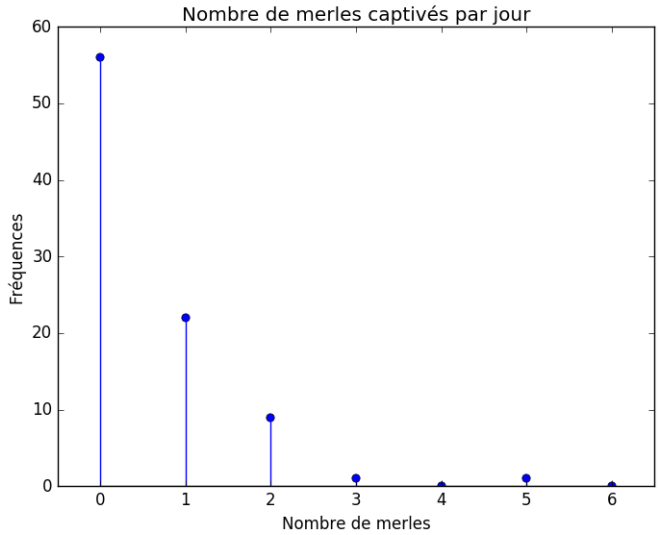
En 1968, à la station ornithologique du col de Golèze situé dans les Alpes françaises, 48 merles à plastron ont été capturés au filet pour être marqués, durant les 89 jours d'ouverture de la station. Les merles sont réputées pour vivre en solitude. On s'intéresse au nombre de merles capturés par jour. Les données sont reproduites dans la table suivante :

Nombre de merles	0	1	2	3	4	5	6
Fréquences	56	22	9	1	0	1	0

1. Représenter graphiquement ces données par un diagramme en bâtons en annotant les axes.
2. Argumenter s'il fait sens de modéliser ces données par un échantillon de loi Poisson d'un certain paramètre λ.
3. Calculer (en Python, avec les fonctions `np.dot` ou `np.average`), le nombre moyen de merles capturés par jour. Appelons ce nombre $\hat{\lambda}$.
4. Dans une nouvelle fenêtre, superposer au diagramme de la première partie la fonction de masse de la loi de Poisson de paramètre $\hat{\lambda}$, convenablement mise à échelle.

Correction. *1.*

```
>>> N = np.arange(0,7)
>>> Frequences = np.array([56,22,9,1,0,1,0])
>>>
>>> plt.figure()
>>> plt.stem(N,Frequences)
>>> plt.xlabel(u'Nombre de merles')
>>> plt.ylabel(u'Fréquences')
>>> plt.title(u'Nombre de merles captivés par jour')
>>> plt.xlim(-0.5,6.5)
>>> plt.show()
```



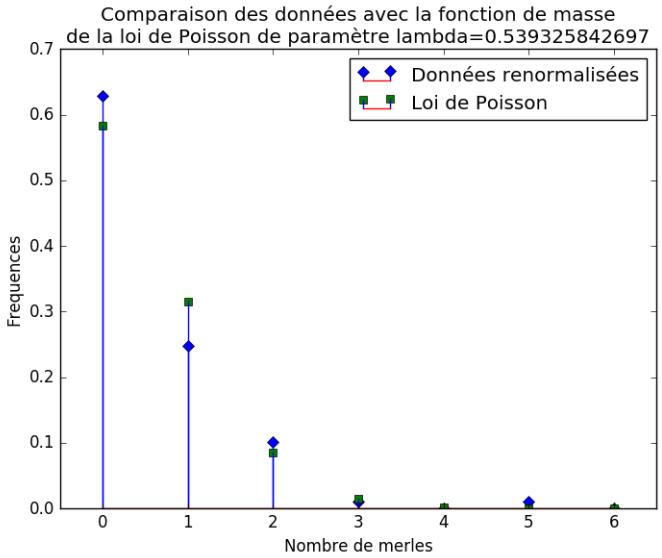
2. *Les merles vivant en solitude, on peut supposer que pour un jour donné, chaque merle et capturé indépendamment et ce avec une faible probabilité. Le nombre de merles capturés peut alors se modéliser par une loi binomiale dont le paramètre p est petit, et donc par la loi de Poisson d'un certain paramètre λ. Pour supposer que le paramètre ne change par jour par jour, il faut supposer de plus que le comportement des merles reste constant durant la durée de l'expérience.*

3.

```
>>> L = np.dot(N,Frequences)/float(sum(Frequences))
>>> print L
0.539325842697
```

4.

```
>>> plt.figure()
>>> plt.stem(N,Frequences/float(sum(Frequences)),markerfmt='bD')
>>> plt.xlabel(u'Nombre de merles')
>>> plt.ylabel(u'Frequences')
>>> plt.title(u'Comparaison des données avec la fonction de masse\nde
la loi de Poisson de paramètre lambda='+str(L))
>>> plt.stem(N,scs.poisson.pmf(N,L),markerfmt='gs')
>>> plt.xlim(-0.5,6.5)
>>> plt.legend([u'Données renormalisées', u'Loi de Poisson'])
>>> plt.show()
```



Le paramètre de la loi de Poisson étant aussi sa moyenne (=son espérance), il est naturel de comparer les données à la loi de Poisson de paramètre égal à la moyenne du nombre de merles capturés par jour, donc $\lambda \approx 0.54$. *On voit que les deux correspondent assez bien, suggèrent que le nombre de merles capturés en une journée peut être modélisé par une loi de Poisson de ce paramètre.*