Vincent Vigon

Python

pour les proba-stats

Université de Strasbourg

Table des matières

Chapitre 1

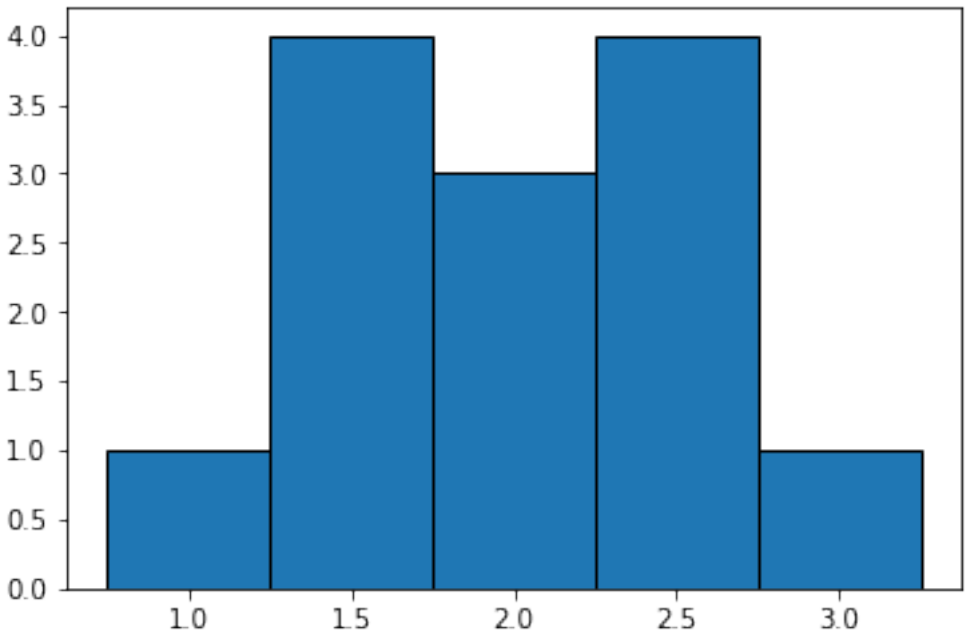
histogrammes et densités

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
```

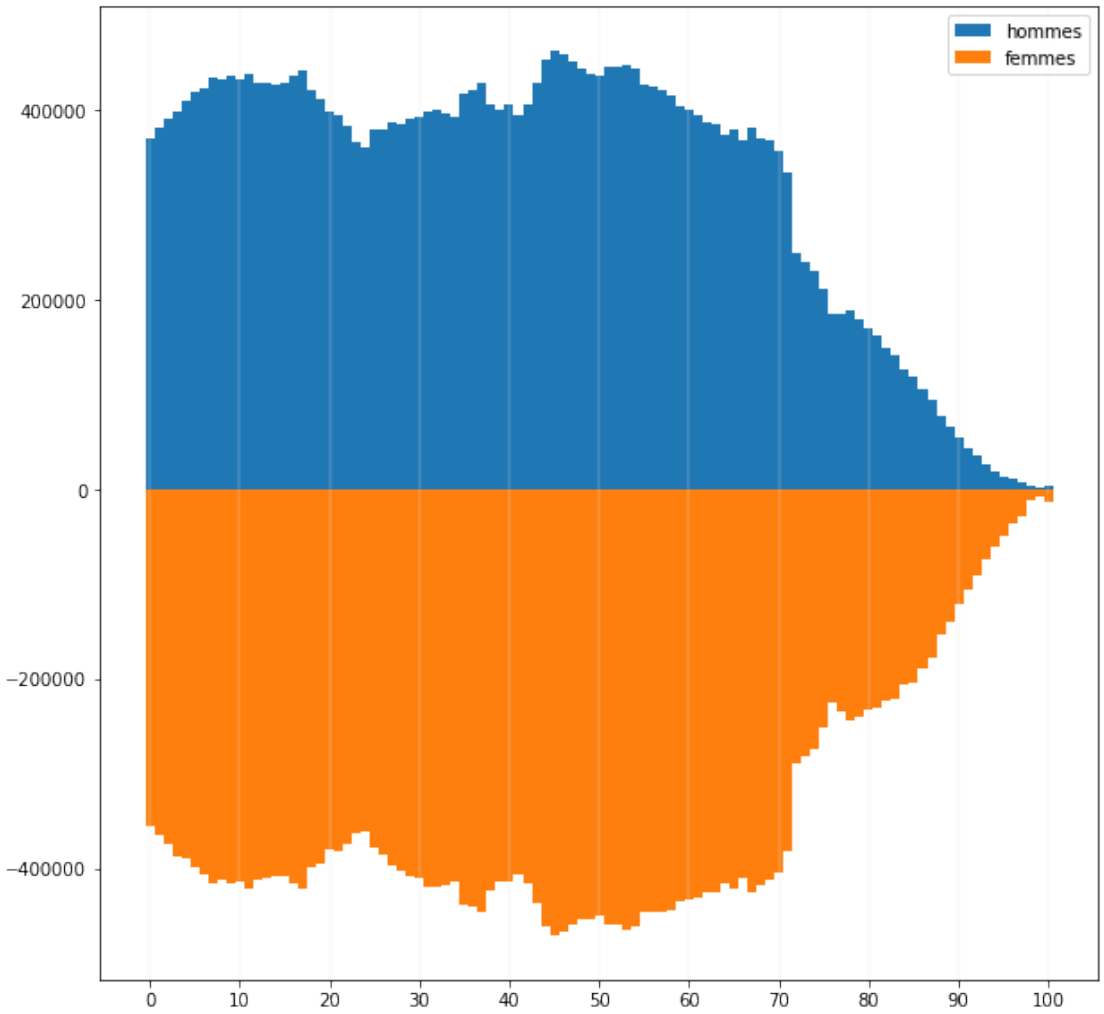
1.1 Graphiques en batons

bonjour toto

```
In [2]: x= [1,1.5,2,2.5,3]
        y= [1,4,3,4,1]

        """ Par défaut  width=0.8, ce qui ne ferait pas joli ici"""
        plt.bar(x,y,edgecolor="k",width=0.5) ; # "k" c'est black
```



In [3]:
```
""" Population totale par sexe et âge au 1er janvier 2018, France. source:
https://www.insee.fr/fr/statistiques/fichier/1892086/pop-totale-france.xls """
nb_hommes=[370264,380786,390091,398757,409096,419458,422341,433633,433293,435936,432382,439050,429300,42792
nb_femmes=[354226,363749,373574,386477,389867,398597,405611,415679,412153,415631,413237,420649,411513,40986
nb_femmes=-np.array(nb_femmes)

ages=range(0,len(nb_hommes))

plt.figure(figsize=(10,10))

plt.bar(ages,nb_hommes,width=1,label="hommes")
plt.bar(ages,nb_femmes,width=1,label="femmes")
xticks=np.arange(0,101,10)
plt.xticks(xticks)
for x in xticks : plt.axvline(x,color="0.9",linewidth=0.3)
plt.legend() ;
```



Exo : changez les abscisses pour faire apparaitre les années de naissances. Essayez de justifier les trous et les bonnes dans la pyramide d'âge.

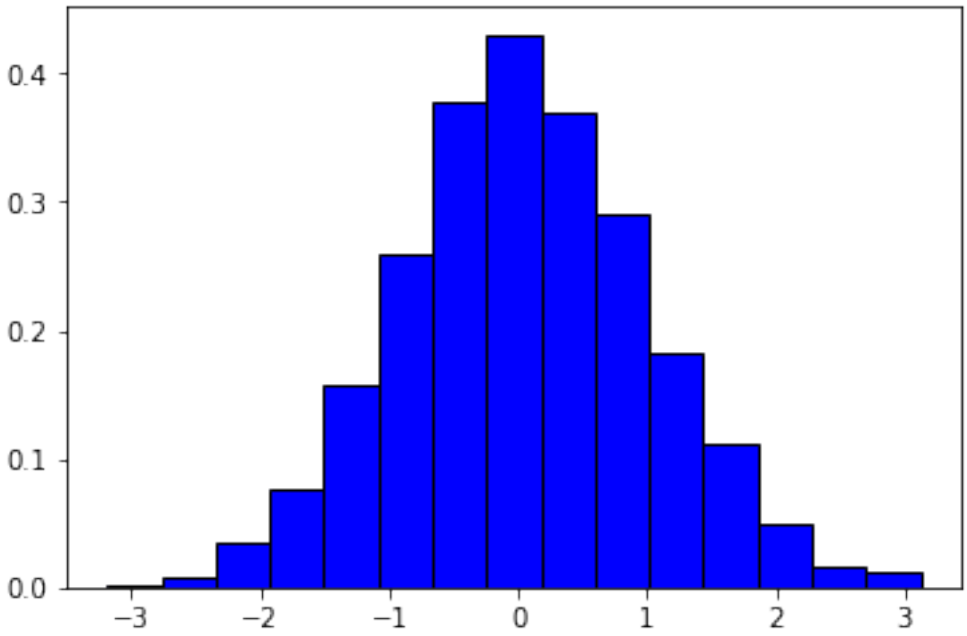
Exo : Comment expliquez-vous le grand nombre de centenaire comparez aux personnes de 98 ou 99 ans ?

1.2 Histogrammes

Considérons maintenant un échantillon c.à.d un ensemble de nombre réels. Dans la plupart des cas les échantillons sont construit : * soit à partir de simulation successive de v.a (ex : v.a gaussienne) * soit à partir d'observations (ex : tailles des gens dans la rue)

Dresser l'histogramme d'un échantillon consiste à découper les réel en sous-interavalles, puis d'afficher des batons qui ont pour base ces sous-intervallse, et comme hauteur le nombre d'élément de l'échantillon contenu dans chaque sous-intervalles.

```
In [4]: """l'échantillon à observer"""
        X=np.random.normal(0,1,size=1000)
        plt.hist(X,bins=15,color='blue',density=True,edgecolor="k");
```

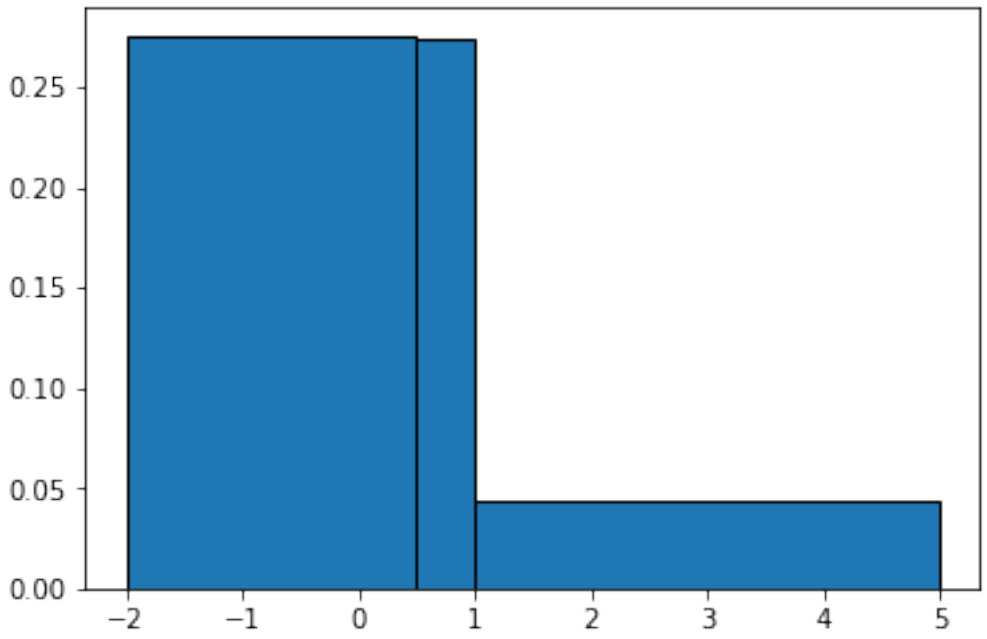


Explication des options de plt.hist :

- bins=10 : on découpe l'intervalle [min(X),max(X)] en dix sous-intervalles.
- normed=True : la hauteur des batons est normalisée pour que cela ressemble à une densité
- rwidth=0.9 : la largeur de chaque baton occupe 90% de chaque sous-intervalle.

Mais parfois il est préférable de préciser nous même les sous-intervalles (=la base des batons)

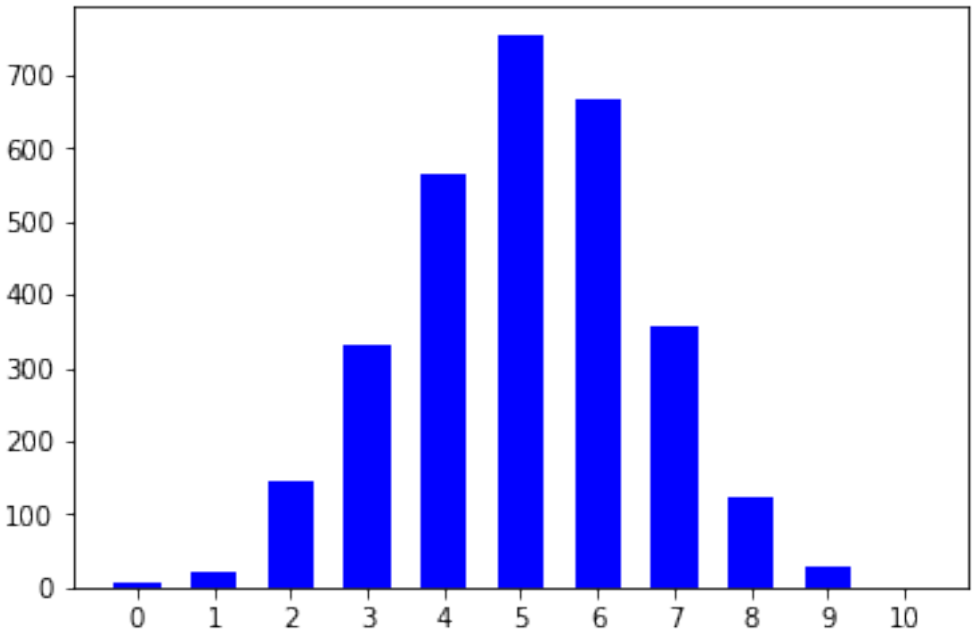
```
In [5]: X=np.random.normal(0,1,size=1000)
        plt.hist(X, bins=[-2,0.5,1,5],  density=True,edgecolor="k") ; #un choix particulièrement idiot de bins
```



1.3 Histogramme de loi discrète

Attention, pour les lois discrètes il faut obligatoirement préciser le découpage. Pour voir une catastrophe, remplacez bins par 11 dans plt.hist(). Expliquez le phénomène.

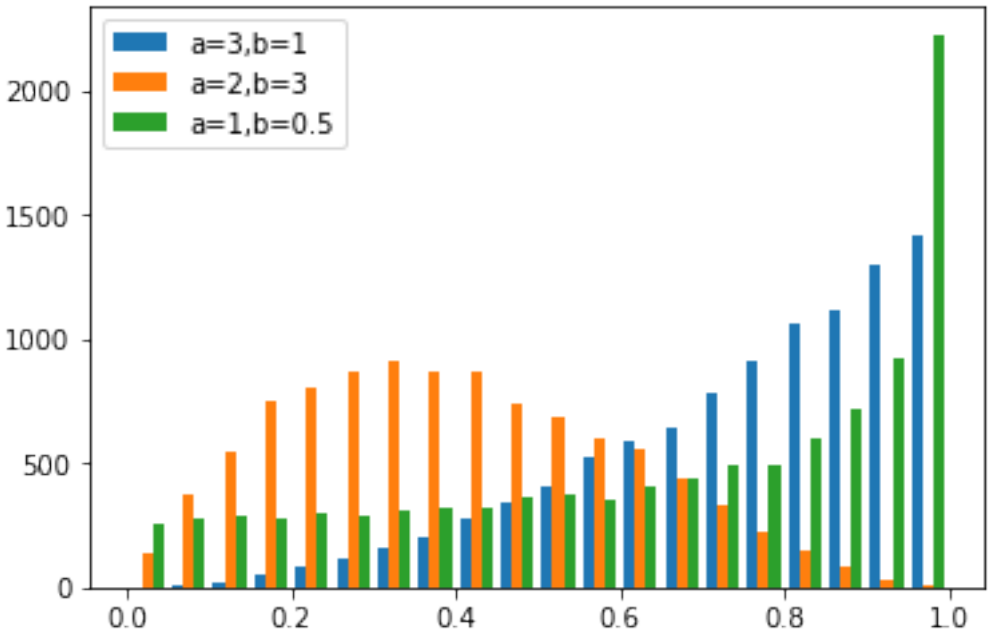
```
In [6]: n=10
        X=np.random.binomial(n,0.5,size=3000)

        """attention np.arange(0,n+2,1) donne l'intervalle discret [0,n+2[= [0,n+1].
         on lui soustrait ensuite 0.5 pour avoir chaque entier de [0,n] dans un sous-intervalle"""
        bins=np.arange(0,n+2,1)-0.5

        """ rwidth=0.6 (=ratio_width) signifie que la base des batons occupe 60% des sous-intervalles.  """
        plt.hist(X,bins=bins, histtype='bar', color='blue', rwidth=0.6)
        """on précise les graduations en x"""
        plt.xticks(np.arange(0,n+1,1)) ;
```



1.3.1 Plusieurs histogrammes

comparons des lois béta

```
In [7]: nbData=10000
        X1=np.random.beta(3,1,size=nbData)
        X2=np.random.beta(2,3,size=nbData)
        X3=np.random.beta(1,0.5,size=nbData)

        plt.hist([X1,X2,X3],bins=20,label=["a=3,b=1","a=2,b=3","a=1,b=0.5"]);
        plt.legend();
```



La variété des formes possible d'une loi la rend très pratique en modélisation.

Choisissez des lois bêta bien choisies (dilatée par une constante), pour modéliser les variables X suivantes : * X : quantité chocolat consommée par les français (sachant que plus on en mange, et plus on a envie d'en manger) * X : durée de vie des français * X : durée de vie des grenouilles (forte mortalité infantile)

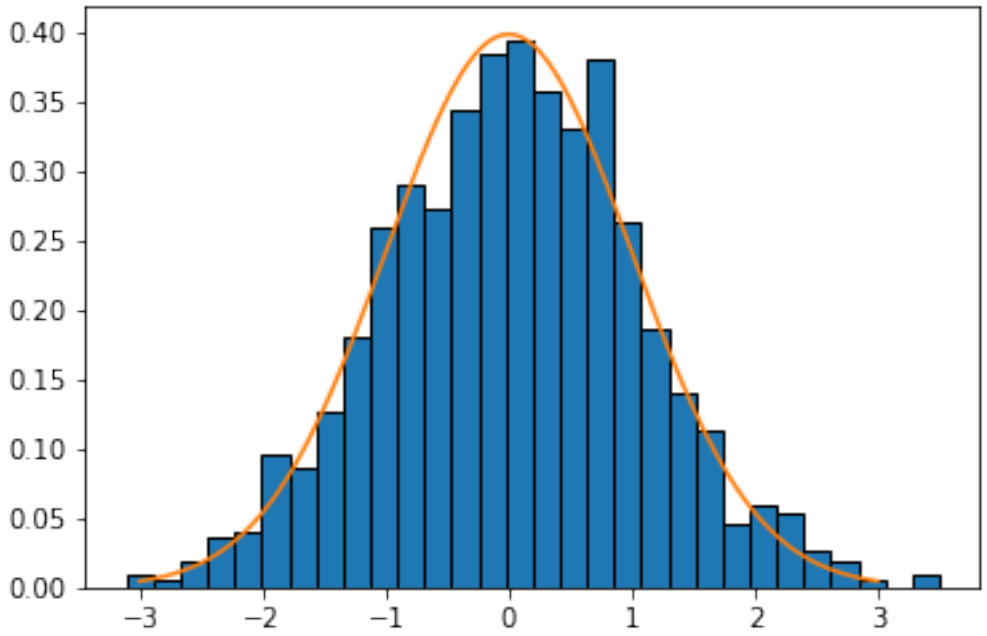
Dressez les histogrammes

Connaissez-vous d'autre loi pour des durées de vie ?

1.4 Superposons histogramme et densité

1.4.1 une loi normale

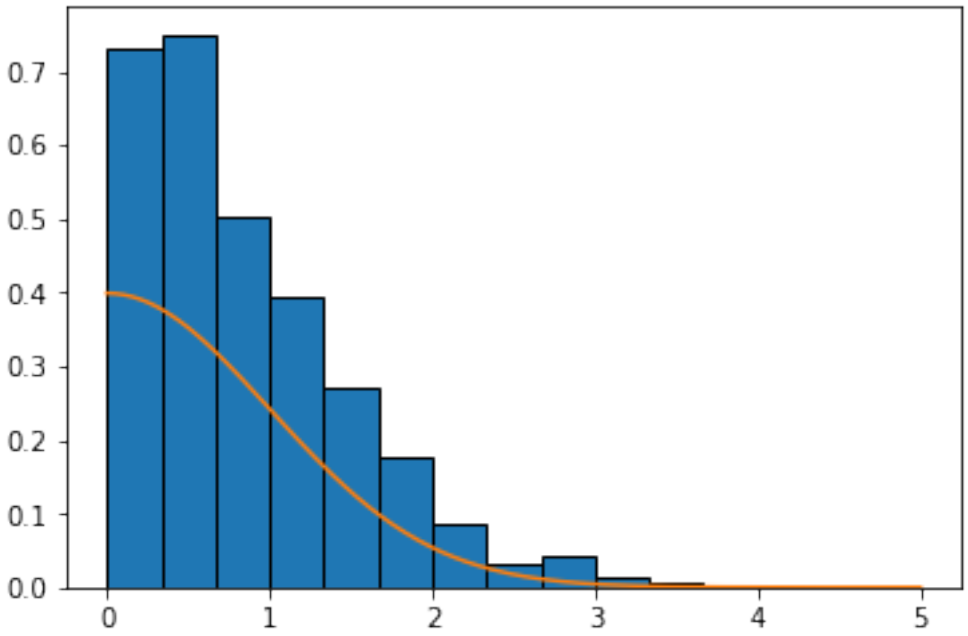
```
In [8]: nbSimu=1000
        Simu=np.random.normal(size=nbSimu)

        """formule à emmener partout avec soi"""
        def gaussian_density(x) :
            return 1/(np.sqrt(2*np.pi))*np.exp(-0.5*x**2)

        plt.hist(Simu,bins=30,density=True,edgecolor="k") ;

        x=np.linspace(-3,3,200)
        plt.plot(x, gaussian_density(x)) ;
```



1.4.2 Tronquer un histogramme

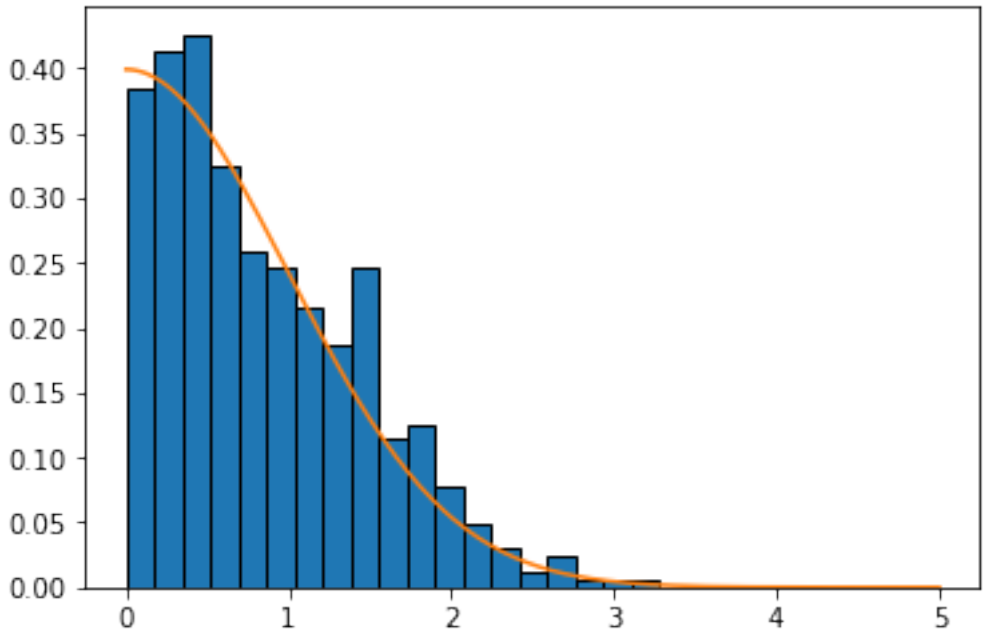
Parfois on a envit de ne montrer qu'une partie de l'histogramme. plt.hist dispose d'une option range qui ignore les valeurs de l'échantillon en dehors du range. Mais malheureusement, plt.hist utlise la normalisation 'naturelle' qui n'est pas compatible avec la superposition avec la densité théorique.

```
In [9]: X=np.random.normal(0,1,size=1000)
        plt.hist(X,bins=15,range=[0,5],density=True,edgecolor="k");
        x=np.linspace(0,5,200)
        plt.plot(x, gaussian_density(x));
```



Il faut donc faire la normalisation à la main, en précisant le poids de chaque observation.

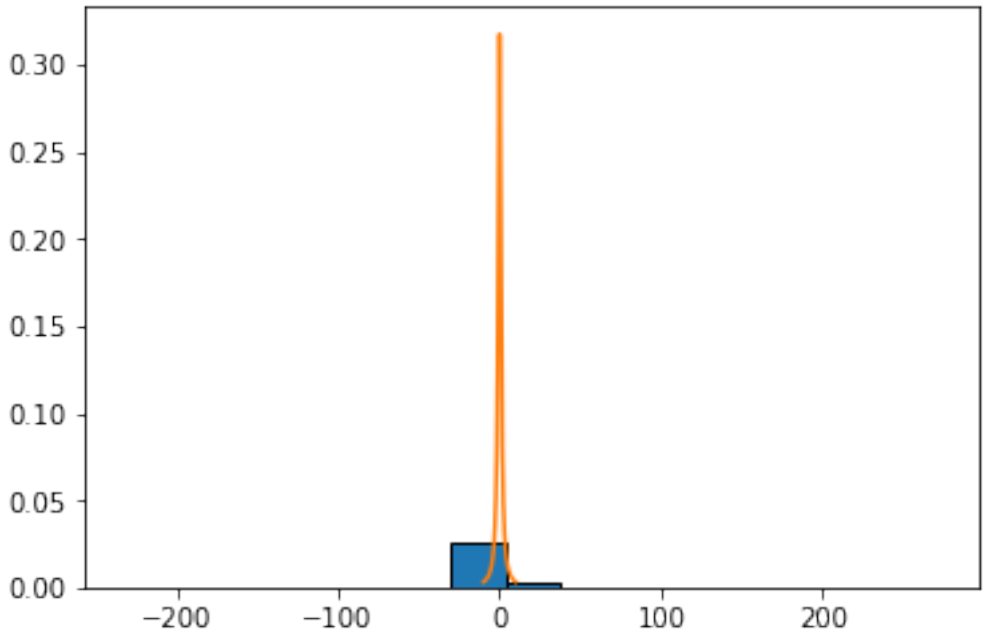
```
In [10]: X=np.random.normal(0,1,size=1000)

         nb_batons=30
         gauche=0
         droite=5
         bins=np.linspace(gauche,droite,nb_batons)
         step=(droite-gauche)/nb_batons
         weights=np.ones_like(X)/step/len(X)

         plt.hist(X,bins=bins,weights=weights,range=[gauche,droite],edgecolor="k")
         x=np.linspace(gauche,droite,200)
         plt.plot(x, gaussian_density(x)) ;
```



Exo : améliorer l'histogramme ci-dessous en tronquant les grandes valeurs de l'échantillon de lois de Cauchy.

Plutôt que de recopier le code précédent, créez une petite fonction qui trace un histogramme tronqué. Elle pourrait par exemple avoir comme signature : `hist_tronc( ech,gauche,droite,nb_batons )`

```
In [11]: def cauchy_density(x) :
             return 1/np.pi/(1+x**2)


         X=np.random.standard_cauchy(size=200)
         plt.hist(X,bins=15,density=True,edgecolor="k");
         x=np.linspace(-10,10,200)
         plt.plot(x, cauchy_density(x));
```



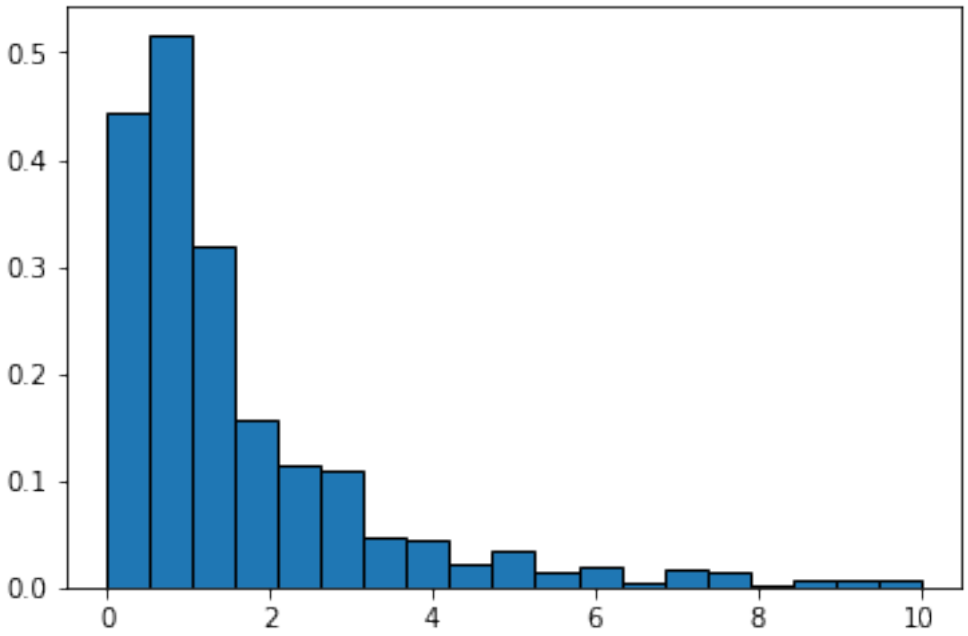
1.4.3 Une loi log-normale

Exo : Que représente une distribution log-normale ?

Réponse : c'est la distribution de $f(X)$ avec f ... et X ...

Vérifiez votre réponse en superposant histogramme de $f(X)$ à l'histogramme proposé ci-dessous.

```
In [12]: size=1000
         X=np.random.lognormal(size=size)
         bins=np.linspace(0,10,20)
         plt.hist(X, bins=bins, density=True,edgecolor="k") ;
```



*** Exo suite :*** Réponse de la première partie : la loi log-normale c'est la loi de $exp(X)$ avec $X \sim Normale(0,1)$.

A partir de cette description, vous devez pouvoir intuitivement comprendre que la densité de la log-normale en 0 vaut ... Ce fait est très mal illustrer par l'histrogramme ci-dessus. Modifiez-le !

Complétez le calcul de la densité de la log-normale :

Considérons ϕ une fonction teste et X une v.a de loi normale.

$$\mathbf{E}[\phi(\exp(X))] = cst \int \phi(e^x)\, e^{-\frac{1}{2}x^2}\, dx$$

on fait le changement de variable $e^x \to y$ on trouve ...

donc la densité de $\exp(X)$ est ...

Superposez cette densité avec l'histogramme précédent pour valider votre calcul.

1.4.4 Une loi binomiale

Quelle est le lien entre loi binomiale et loi de bernouilli ?

```
In [13]: """ échantillon d'une binomiale """
         X=np.random.binomial(n=10,p=0.5,size=100)
         np.set_printoptions(linewidth=2000)
         print("X:",X)
```

```
X: [6 5 4 7 3 2 3 4 5 4 6 3 5 6 5 2 5 5 5 4 7 6 6 5 5 5 6 7 6 6 7 4 9 4 6 3 3 5 7 4 5 4 3 4 5 6 6 6 6 5
```

Dressez l'histogramme de ces simulations. Superposez cet histogramme avec la densité discrète de la loi binomiale.

On calculera cette densité point par point (sans chercher de package particulier). Vous aurez seulement besoin de la fonction factorielle.

Remarque : Il est plus élégant de ne pas relier les points d'une densité discrète.

```
In [14]: import math
         print("10!:",math.factorial(10))
```

```
10!: 3628800
```

Chapitre 2

Plus de lois avec scipy

```
In [1]: import numpy as np
        np.set_printoptions(precision=2,suppress=True)
        import matplotlib.pyplot as plt

        """Voici un autre import qu'il faut connaître :
        scipy= sci-entific py-thon """
        import scipy.stats as stats
        %matplotlib inline
```

4 mots clefs à retenir (qui permettent aussi d'améliorer son anglais scientifique) :

- pdf -> Probability density function. -> densité (prend des réels en argument)
- pmf -> Probability mass function -> densité discrète (prend des entiers en argument)
- cdf -> Cumulative density function. -> fonction de répartition
- ppf -> Percent point function (inverse of cdf) -> fonction quantile (ou percentile)
- rvs -> Random variates. -> simulation d'un échantillon de va ayant la loi donnée

A ce point du TP, vous vous dites qu'il y a vraiment trop de choses à retenir. Mais nous allons les pratiquez très souvent : cela rentrera tout seul.

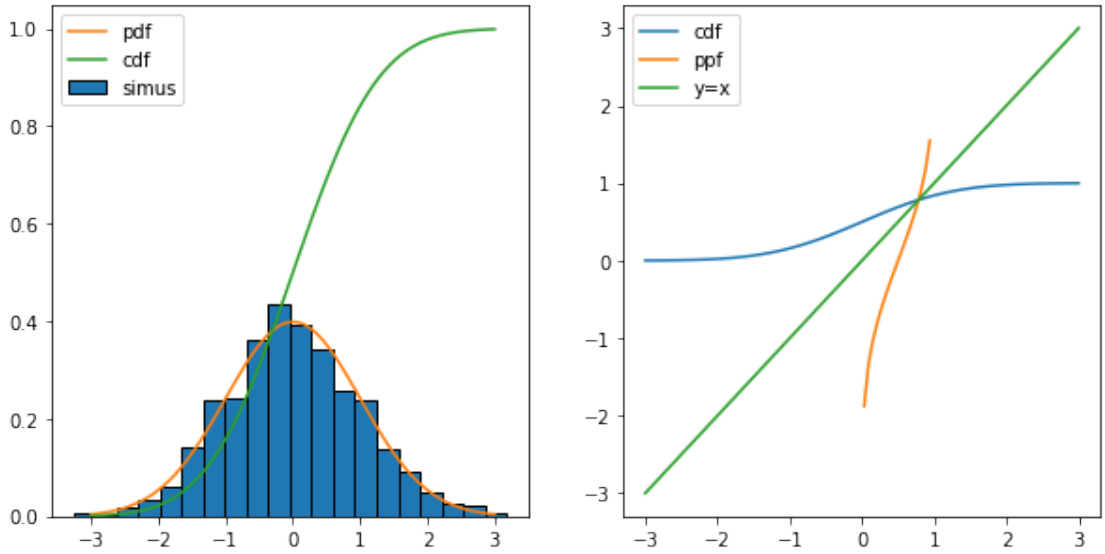
2.1 Densité, fonction de réparition, quantiles

2.1.1 une loi continue

```
In [2]: simus=stats.norm.rvs(loc=0, scale=1, size=1000)
        """
        c'est tout à fait identique à :
        simus=np.random.normal(loc=0,scale=1,size=1000)

        cependant scipy.stats contient encore plus de loi que numpy.random.
        Par exemple la loi t de student utile pour les stats.
        """
        plt.figure(figsize=(10,5))
```

```
x=np.linspace(-3,3,100)
pdf=stats.norm.pdf(x, loc=0, scale=1)
cdf=stats.norm.cdf(x, loc=0, scale=1)
ppf=stats.norm.ppf(x, loc=0, scale=1)

plt.subplot(1,2,1)
plt.hist(simus,20,normed=True,label="simus",edgecolor="k")
plt.plot(x,pdf,label="pdf")
plt.plot(x,cdf,label="cdf")
plt.legend()

plt.subplot(1,2,2)
plt.plot(x, cdf,label="cdf")
plt.plot(x, ppf,label="ppf")
plt.plot(x, x,label="y=x")
plt.legend();
```

```
/Users/vigon/Library/Python/3.6/lib/python/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



Le tracé de la ppf ci-dessus n'est pas très joli, on a l'impression qu'il est incomplet. Changez cela.

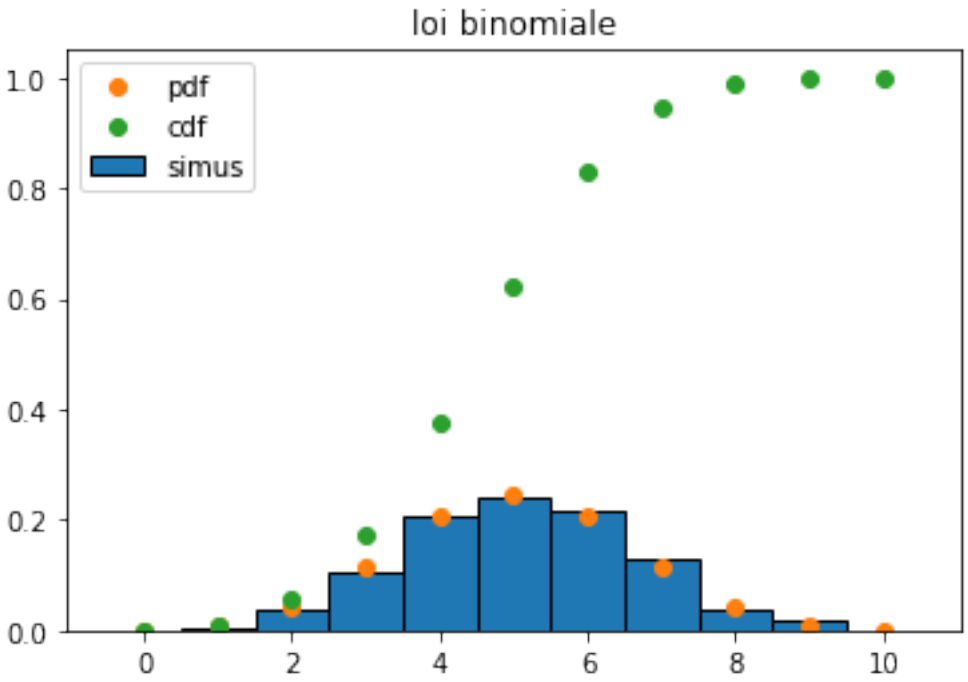
Aide : n'oubliez pas qu'une courbe, c'est des points qu'on relie entre eux, et que ces points, c'est vous qui les spécifiez.

2.1.2 une loi discrète

```
In [3]: n=10
        p=0.5
```

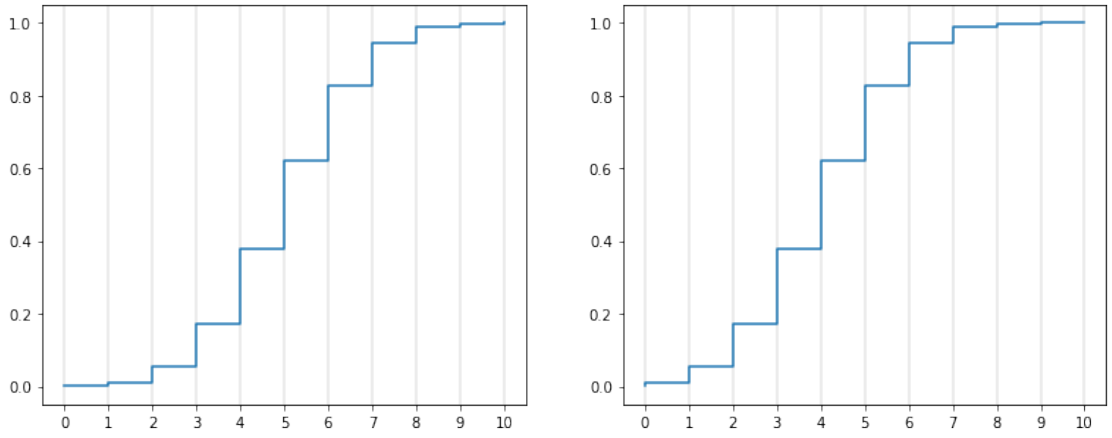
```
simus=stats.binom.rvs(n, p, size=1000)


x=np.arange(0,n+1)
bins=np.arange(0,n+2)-0.5
""" attention, la densité discréte en scipy c'est pmf et pas pdf """
pdf=stats.binom.pmf(x, n, p)
cdf=stats.binom.cdf(x, n, p)


plt.hist(simus, bins, normed=True, label="simus",edgecolor="k")
plt.plot(x, pdf,'o' , label="pdf")
plt.plot(x, cdf,'o', label="cdf")
plt.title("loi binomiale")
plt.legend();
```

```
/Users/vigon/Library/Python/3.6/lib/python/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



In [4]:
```
""" variante. On trace la fonction de répartition en escalier.
On rajoute des lignes verticales pour vous aidez dans l'exo suivant"""
plt.figure(figsize=(13,5))

plt.subplot(1,2,1)
""" lignes verticales.  """
for xc in x :plt.axvline(x=xc, color='0.9') #0.9 => un gris très clair (0=noir,1=blanc)
plt.plot(x, cdf,drawstyle='steps-post')
```

```
plt.xticks(x)


plt.subplot(1,2,2)
for xc in x :plt.axvline(x=xc, color='0.9')
plt.plot(x, cdf,drawstyle='steps-pre')
plt.xticks(x) ;
```



Exo : laquelle de ces deux représentations correpond à la fonction de répartition, à savoir : $x \to P[X \leq x]$

Conseil : lisez bien les messages d'erreur. Par exemple si vous écrivez : `plt.plot(x,y,drawstyle='steps-after',color="red")` cela provoque un message d'erreur :

```
ValueError: Unrecognized drawstyle steps-pre default steps-post steps-mid steps
```

Ce message nous indique que l'on c'est tromper de mot clef, et nous propose les mot clefs valides. D'après vous, quand on écrit `drawstyle=default`, comment les points sont reliés ?

2.2 Conseil python : les arguments facultatifs

Considérons un appel d'une fonction de scipy :

```
In [5]: stats.norm.rvs(loc=-1,scale=3,size=100) ;
```

Tous les arguments sont facultatifs. Les valeurs par défaut sont logiquement `loc=0,scale=1,size=1` On peut écrire par exemple

```
simus=stat.norm.rvs(size=1000)
```

pour

```
simus=stat.norm.rvs(loc=0,scale=1,size=1000)
```

Mais attention : si on ne précise pas le nom des arguments, ils sont pris dans l'ordre `1:loc 2:scale 3:size` Par exemple, si je veux tirer 1000 gaussienne et que j'écris

```
simus=stat.norm.rvs(1000)
```

mon programme bug car cela correspond à

```
simus=stat.norm.rvs(loc=1000)
```

Je vous conseille d'écrire quasi tout le temps le nom des arguments pour éviter ce genre de confusion ; sauf quand il s'agit d'un argument obligatoire évident comme dans `plt.plot(x,y)`.

Ou bien, conseil plus simple : adopter dans un premier temps la façon dont sont codés ces TP. Quand vous serez des vieux routier du python, vous créerez votre propre style.

2.3 lois continues classiques

2.3.1 paramètres de localisation et d'échelle

Toutes les lois dans scipy ont un paramètre de localisation `loc` que nous notons ici μ et un paramètre d'échelle `scale` que nous notons ici σ. Leur interprétation est la suivante :

Si l'appel de `stats.xxx.rvs()` renvoie une v.a X alors `stats.xxx.rvs(loc=mu,scale=sigma)` renvoie une v.a ayant la même loi que sigma $\sigma X + \mu$. Ainsi ces deux paramètres effectue à des translation et dilatation de la densité comme l'indique la proposition suivante :

Proposition : si $x \to f(x)$ est la densité d'une va X, alors la densité de $\sigma X + \mu$ est :

$$\frac{1}{\sigma} f\left(\frac{x - mu}{\sigma}\right)$$

Astuce : Pour ne pas s'encombrer la mémoire, retenez uniquement les densités des lois dans le cas $\mu = 0$ et $\sigma = 1$. Par exemple :

$$a \quad b$$

nom de la loi	simplifié	complète
exponentielle	$e^{-x} 1_{\{x>0\}}$	$\frac{1}{\sigma} \exp(-\frac{x-\mu}{\sigma})$
normale	$\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$	...

Démo de la proposition : Considérons ϕ fonction teste et X une va de densité f

$$\mathbf{E}[\phi(\sigma X + mu)] = \int \phi(\sigma x + \mu)\, f(x)\, dx$$

On effectue le changement de variable $\sigma x + \mu \to y$

$$\mathbf{E}[\phi(\sigma X + mu)] = \int \phi(y)\ f(...)\ dy...$$

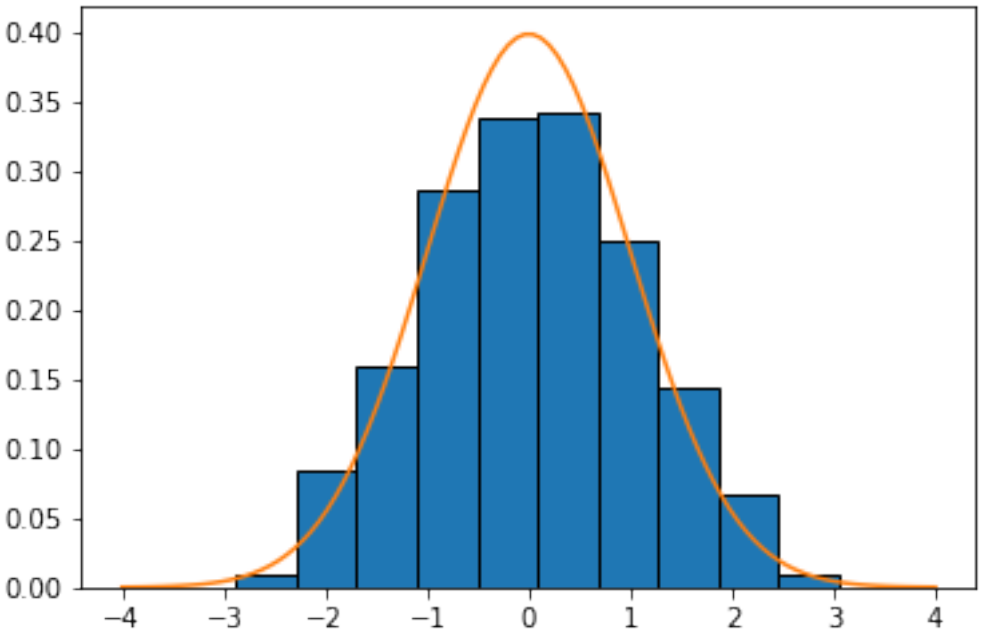
On en déduit que ...

2.3.2 Loi Normale

```
norm.pdf(x) = exp(-x**2/2)/sqrt(2*pi)
```

```
In [6]: plt.hist(stats.norm.rvs(loc=0,scale=1,size=400),normed=True,edgecolor="k")

        x=np.linspace(-4,4,100)
        plt.plot(x,stats.norm.pdf(x,loc=0,scale=1)) ;
```

```
/Users/vigon/Library/Python/3.6/lib/python/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



EXO : Reprenez le dernier programme, améliorez-le car il souffre d'un défaut classique. Faites varier les paramètres loc et scale. Supperposer plusieurs graphique pour que l'on comprennent bien les effets de dilatation et de translation de ces paramètres.

2.3.3 Paramètres de forme

Pour toutes les loi suivantes, nous ne nous occuperons plus des paramètres loc et scale, pour nous concentrer sur les autres paramètres (quand ils existent). Ces autres paramètres sont appelé paramètres de forme.

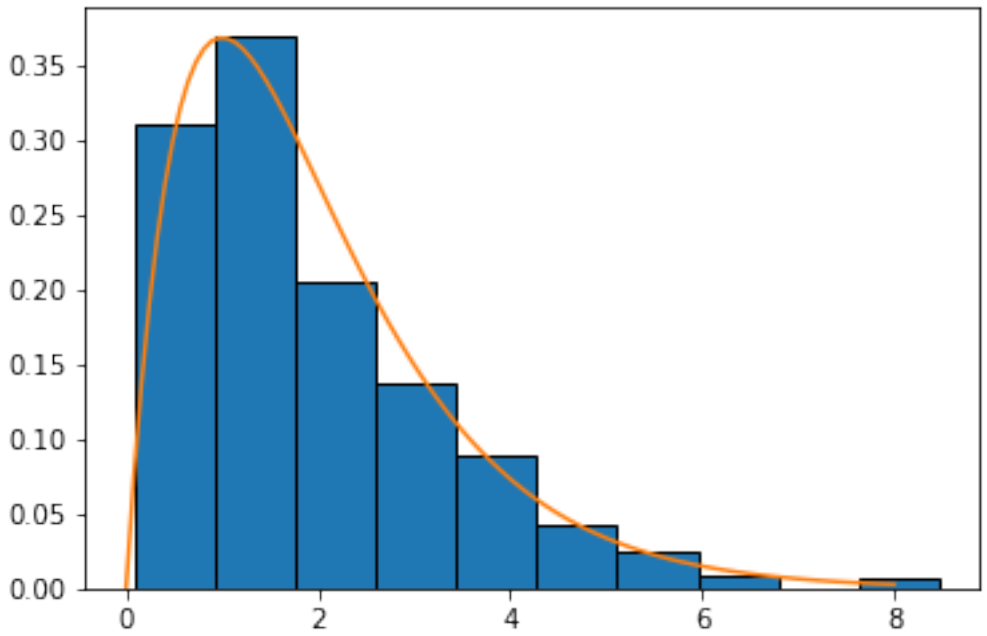
2.3.4 Loi gamma

```
gamma.pdf(x, a) = x**(a-1) * exp(-x) / gamma(a)
```

Pour quelle valeur de a retrouve-t-on une loi exponentielle ?

```
In [7]: """paramètre de forme"""
        a=2
        plt.hist(stats.gamma.rvs(a=a,size=400),normed=True,edgecolor="k")

        x=np.linspace(0,8,100)
        plt.plot(x,stats.gamma.pdf(x,a=a)) ;
```

```
/Users/vigon/Library/Python/3.6/lib/python/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



Exo : Le paramètre des forme est a. Quand il est entier il a l'interprétation suivante : gamma(a) est la loi de la somme de a v.a. exponentielle indépendantes. Illustrer ce fait par des simulations.

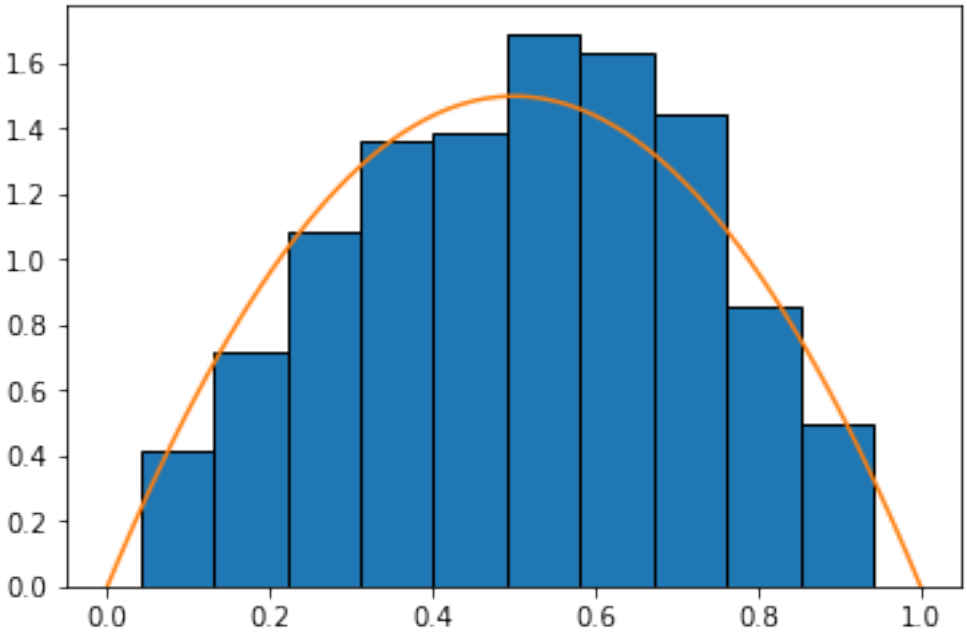
Exo : pour quels paramètres a la densité est-elle monotone ?

2.3.5 Loi Beta

$$\text{beta.pdf(x, a, b)} = \frac{\text{gamma(a+b)} * \text{x**(a-1)} * \text{(1-x)**(b-1)}}{\text{gamma(a)*gamma(b)}}$$

```
In [8]: """ deux paramètres de formes"""
        a=2
        b=2
        plt.hist(stats.beta.rvs(a=a,b=b,size=400),normed=True,edgecolor="k")
        x=np.linspace(0,1,100)
        plt.plot(x,stats.beta.pdf(x,a=a,b=b)) ;
```

```
/Users/vigon/Library/Python/3.6/lib/python/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



Exo : faites varier a et b de manière à faire apparaitres tous les 'type' possible de loi béta : cloche, smiley ,décroissant, croissant

2.4 Lois à Queues lourdes

Une loi est dites à queue lourde lorsque les v.a qui ont cette loi peuvent prendre, de temps en temps, des grandes valeurs positives ou négatives. Elle servent à modéliser des évènements rare et violent (ex : crue d'un fleuve).

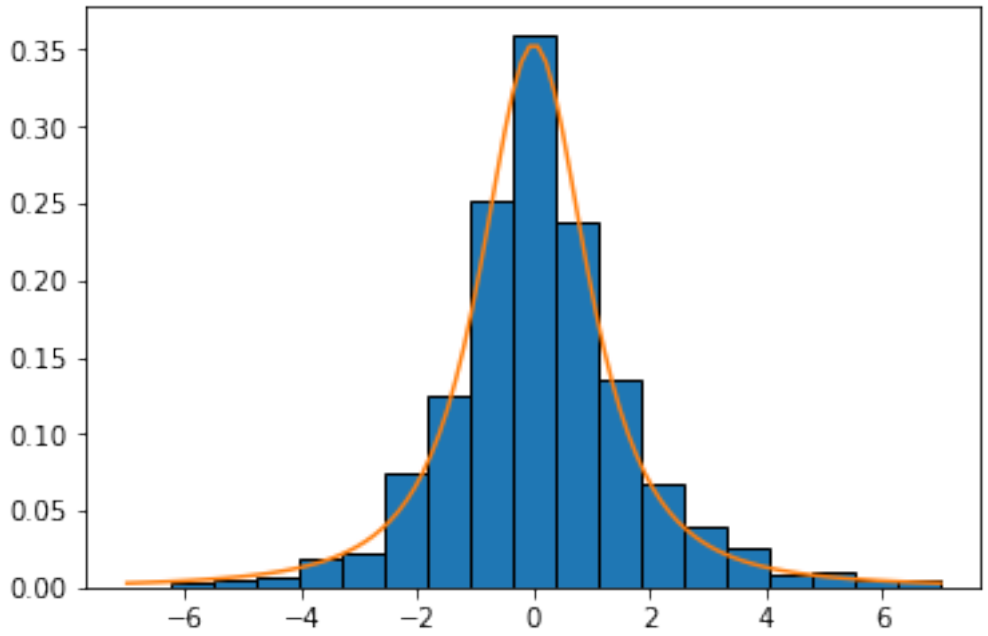
```
In [9]: """ une fonction effectuant un histogramme tronqué """
        def hist_trunc(ech,gauche,droite,nb_batons) :
            bins=np.linspace(gauche,droite,nb_batons)
            interval_width=(droite-gauche)/nb_batons
            weigh=np.ones_like(ech)/len(ech)/interval_width
            plt.hist(ech,bins=bins,weights=weigh,edgecolor="k")
```

2.4.1 Loi t de Student

$$t.pdf(x,\ df) = \frac{\Gamma\left(\frac{df+1}{2}\right)}{\sqrt{\pi \cdot df}\ \Gamma\left(\frac{df}{2}\right)\ \left(1+\frac{x^2}{df}\right)^{\frac{df+1}{2}}}$$

C'est une loi très utile en statistique.

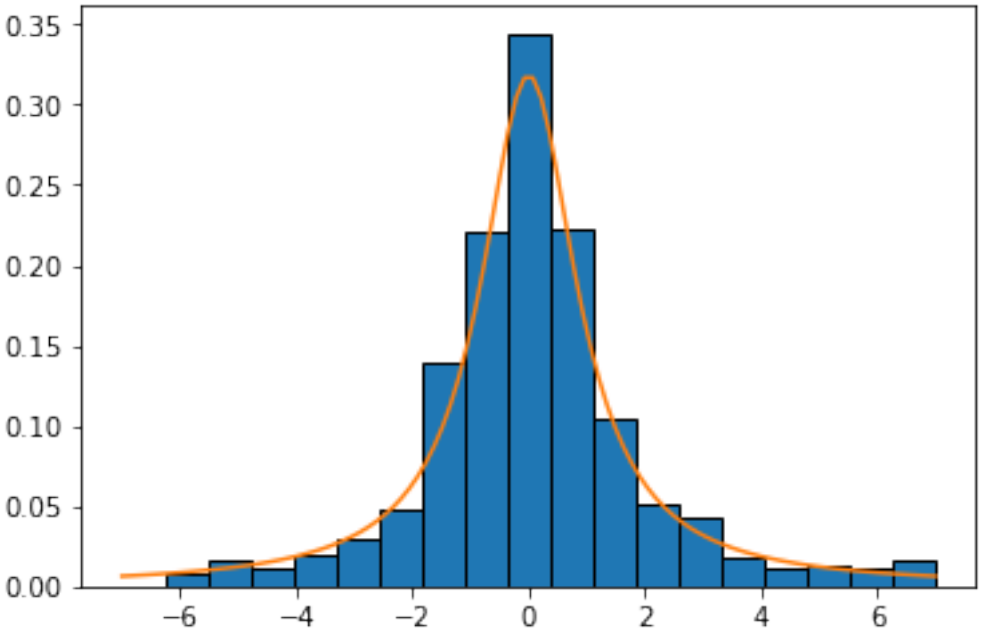
```
In [10]: """ df = degree of freedom"""
         df=2
         X=stats.t.rvs(df=df,size=1000)
         hist_trunc(X,-7,7,20)
         x=np.linspace(-7,7,100)
         plt.plot(x,stats.t.pdf(x,df=df)) ;
```



Pour df petit (ex :2), c'est une loi à queue lourde. Pour df grand, elle ressemble à la loi normale.

2.4.2 Loi de cauchy

```
In [11]: df=2 #degree au freedom
         X=stats.cauchy.rvs(size=1000)
         hist_trunc(X,-7,7,20)
         x=np.linspace(-7,7,100)
         plt.plot(x,stats.cauchy.pdf(x)) ;
```



Proposition : Soit X,Y deux gaussiennes indépendantes. Alors le rapport X/Y suit une loi de Cauchy. (on comprend aussi ici que les va de Cauchy peuvent prendre des valeurs très grandes).

Mais en attendant, on peut facilement faire la preuve avec la technique habituelle :

Soit phi une fonction teste et X,Y deux gaussiennes indépendantes. On a

```
E[X/Y] = cst  int int   phi(x/y)  e^{-0.5 x^2} e^{-0.5 y^2} dx dy
```

En posant x/y = z et donc en faisant le changement de variable x -> yz (ou bien y-> x/z) on obtient . . .

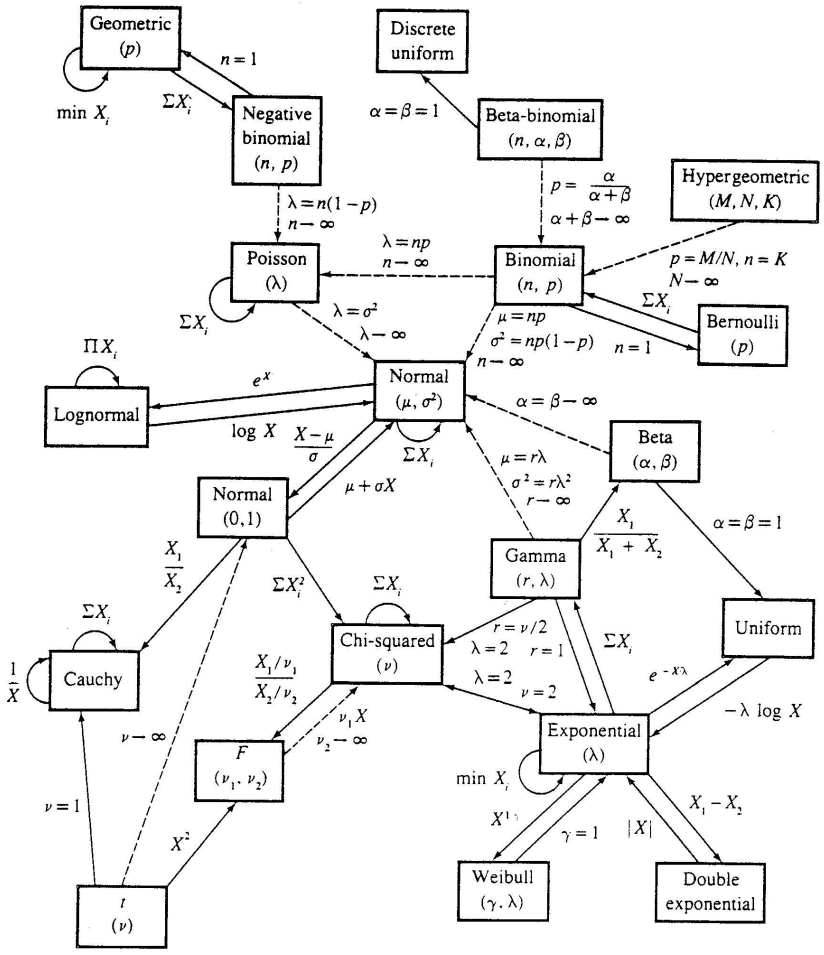
Ensuite, on peut calculer explicitement l'une des deux intégrale, et on tombe sur la densité de la Cauchy.

Exo 1 : Complétez cette démonstration.

Exo 2 : Illustrer la proposition par des simulations.

2.5 Relations entre les lois

Il est important de retenir les relations entre les principales lois, l'image ci-dessous peut vous aider à cela. Sinon, sur cette page contient encore plus de relations, ainsi que leur justifications mathématique.



Relationships among common distributions. Solid lines represent transformations and special cases, dashed lines represent limits. Adapted from Leemis (1986).

Relationships_among_some_of_univariate_probability_distributions