Dans ce TP, on part de `numpy.random.rand` comme seule source d'aléatoire, et on construit les briques de simulations des grandes lois probabilistes.

18.1 Fonctions utiles

La fonction `numpy.random.rand()`

La fonction `rand` du module `numpy.random` permet d'obtenir un réel uniformément tiré dans l'intervalle $[0, 1[$.

```
from numpy.random import rand
print(rand())  # random float between 0 and 1
```

On peut accessoirement préciser des dimensions pour obtenir un tableau numpy dont les coefficients sont aléatoirement tirés dans l'intervalle $[0, 1[$.

```
rand(10)      # list of 10 random floats
rand(10, 5)   # 10 X 5 matrix of random floats
```

C'est la seule fonction du module `numpy.random` que l'on s'autorise à utiliser.

Représenter un histogramme

Pour trace un histogramme, on peut utiliser la fonction `bar` du module `matplotlib.pyplot`. La fonction attend deux arguments, `X` la liste des abscisses et `Y` la liste des ordonnées correspondantes. Ainsi le code ci-dessous produit le graphique figure 18.1. Notez qu'il faut apppeler la fonction `show` pour afficher le graphique.

Si vous utilisez `pyzo`, il est possible que le code ne fonctionne pas directement car `matplotlib` n'est pas installé par défaut, nous essaierons de l'installer pendant le TP mais à défaut vous pouvez utiliser replit.com.

```
from matplotlib.pyplot import bar, show
import numpy as np

X = np.arange(10)
Y = [1, 2, 3, 4, 5, 5, 4, 3, 2, 1]

bar(X, Y)
show()
```

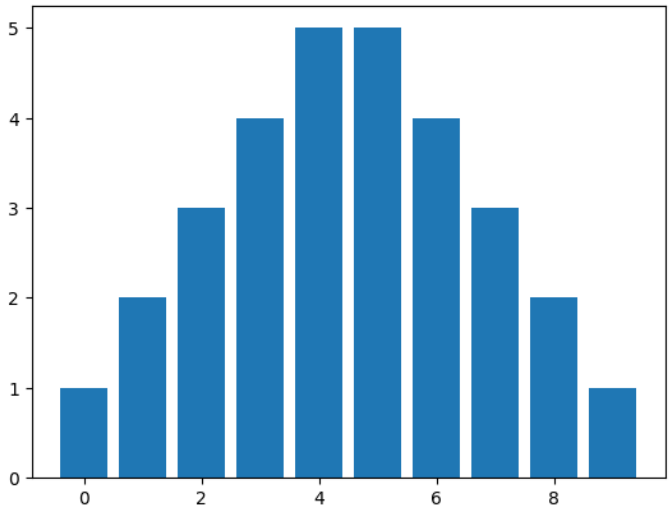

FIGURE 18.1 – Un histogramme de `matplotlib`

18.2 Simulation d'une variable binomiale

Premières fonctions aléatoires

1. Écrire une fonction `bernoulli(p)` qui simule une variable de Bernoulli de paramètre p. Elle renvoie 0 avec probabilité $1-p$ et 1 avec probabilité p.

2. a) Écrire une fonction `uniformInt(a, b)` qui simule une variable aléatoire X uniformément distribuée sur les entiers $a, a+1, \ldots, b-1$.

b) Écrire une fonction `loiUniformInt(a, b, k)` qui renvoie une estimation de $\mathbb{P}(X = k)$ à l'aide de 1000 simulations de X.

c) À l'aide de ces simulations, effectuer le tracé de l'histogramme de la loi de X.

Loi binomiale

La loi binomiale $\mathcal{B}(n, p)$ peut être définie comme la somme de n variables de Bernoulli indépendantes de paramètre p.

3. a) Écrire une fonction `binomial(n, p)` qui simule une variable binomiale X de paramètre n, p. Utiliser pour cela la fonction `bernoulli`.

b) Écrire une fonction `loiBinomiale(n, p, k, N)` qui renvoie une estimation de la probabilité $\mathbb{P}(X = k)$ à l'aide de N simulations de X.

c) À l'aide de ces simulations, effectuer le tracé de l'histogramme de la loi de X. On pourra prendre $n = 30$, $p = 1/2$ et $N = 10\,000$.

Une autre façon de décrire la loi binomiale est la loi du nombre de boule blanche obtenue après n tirages avec remise dans une urne contenant une proportion p de boules blanches (et le reste de boules noires).

4. a) Écrire une fonction `histoBinomial(n, p, N)` qui crée une liste contenant une proportion p de 1 (et le reste de 0), effectue N fois n tirages aléatoires dans la liste, et renvoie le tableau des fréquences du nombre de 1 obtenus pour chacune de ces N simulations. Effectuer à nouveau le tracé de l'histogramme obtenu.

18.3 Pour les plus rapides

5. Écrire une fonction `shuffle(L)` qui effectue une permutation aléatoire des éléments de `L`.