Course 4 - Sampling and probabilities

September 27, 2016

1 Course 4: Probabilities and sampling

In this worksheet, we will play with various probability measures introduced in the Lecture 3 of the course of J. Nzabanita.

- Bernoulli $Ber(p)$
- binomial $B(n, p)$
- geometric $Geom(p)$
- Poisson $Pois(\lambda)$
- uniform $\mathcal{U}(a, b)$
- normal or Gaussian $N(\mu, \sigma^2)$

We will also experiment and illustrate the limit theorems of Lecture 4.

1.1 Plotting discrete probability measures

We will mainly be using the Python libraries `numpy`, `matplotlib.pyplot` and `scipy.stats`. The conventional way to import them is given by the code below

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
```

Exercise: - Copy the above cell and execute it.
We recall that these import statements should be done only once in the whole worksheet!

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        from scipy import stats
```

Exercise: - Do you remember how to access the documentation of a function or a module? - Have a look at the documentation of `stats`. - Could you find the name of the distributions we are interested in (Bernoulli, binomial, geometric, uniform, Poisson and normal)?

```
In [ ]: # to access documentation we use the question mark '?'
        # we can also access the list of functions available in the module stats us
        stats.
```

```
In [2]: stats?
```

In order to construct a probability distribution the syntax is close to the mathematical one

```
P1 = stats.binom(10, 0.3)    # the distribution Binom(10, 0.3)
P2 = stats.poisson(2.3)      # the distribution Pois(2.3)
```

Exercise: - Construct the probabilities `P1` and `P2` given in the code above - Construct the probability `P3` that corresponds to $Geom(0.3)$. - Construct the probability `P4` that corresponds to $Ber(0.7)$. - For each of `P1`, `P2`, `P3` and `P4` print their mean and their variance (*hint: use tab-completion*)

If you have doubts about the mean and variance of a random variable, they were defined in Lecture 2 of J. Nzabanita.

```
In [3]: P1 = stats.binom(10, 0.3)    # binomial
        P2 = stats.poisson(2.3)      # Poisson
        P3 = stats.geom(0.3)         # geometric
        P4 = stats.bernoulli(0.7)    # Bernoulli
```

```
In [29]: # the mean and variance of P1
         print(P1.mean(), P1.var())
```

```
3.0 2.1
```

```
In [50]: discrete_probas = [[P1, 'Binom(10,0.3)'], [P2, 'Pois(2.3)'], [P3, 'Geom(0.
```

```
In [52]: # all at once with a for loop
         print("%14s  %5s  %5s" % ("name", "mean", "variance"))
         for P, name in discrete_probas:
             print("%14s  %.3f  %.3f" % (name, P.mean(), P.var()))
```

```
          name   mean  variance
 Binom(10,0.3)  3.000  2.100
     Pois(2.3)  2.300  2.300
     Geom(0.3)  3.333  7.778
      Ber(0.7)  0.700  0.210
```

```
In [ ]:
```

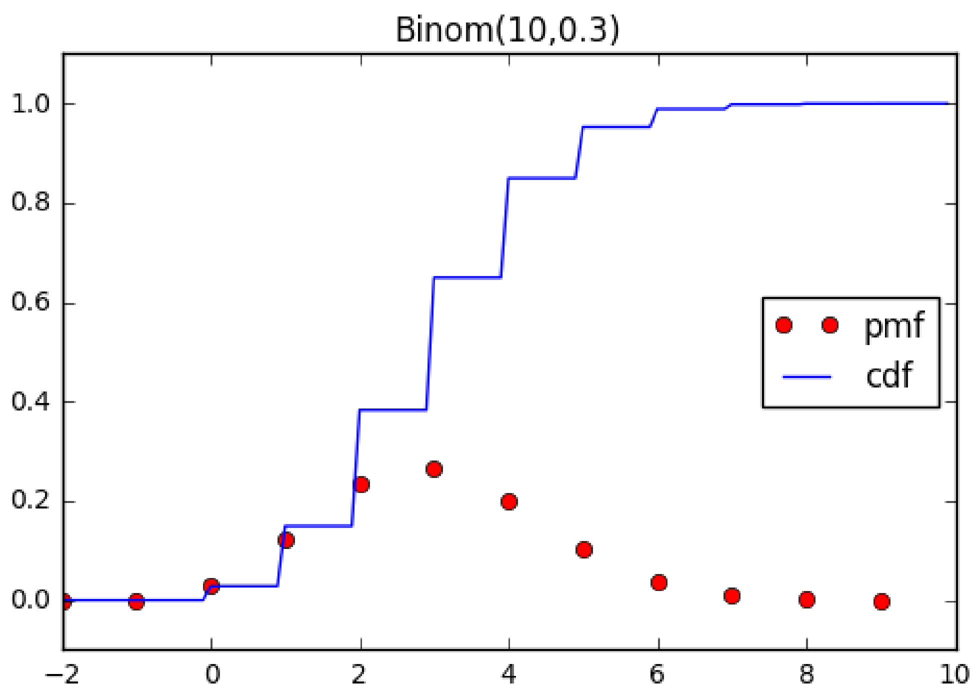
We will now be interested in graphical representation of the probabilities `P1`, `P2`, `P3` and `P4`. We will be using two methods: - `pmf`: for the probablity mass function (i.e. the function $k \mapsto P(\{k\})$) - `cdf`: for the cumulative density function (i.e. the function $k \mapsto P((-\infty, k])$.

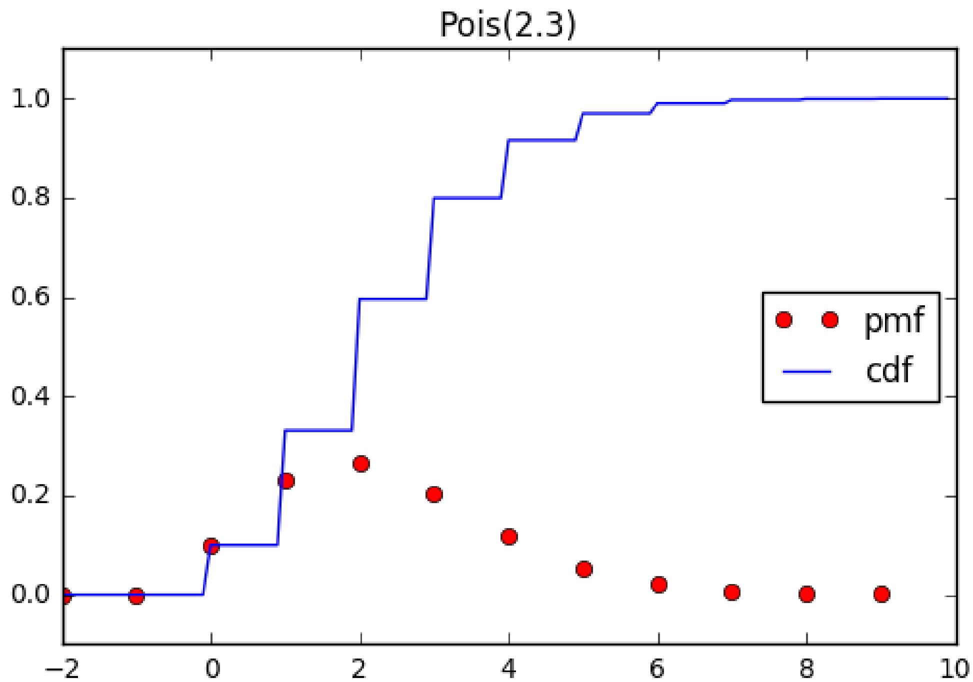
Exercise: - Copy, paste and execute the code below to get a graphical representation of `P1`

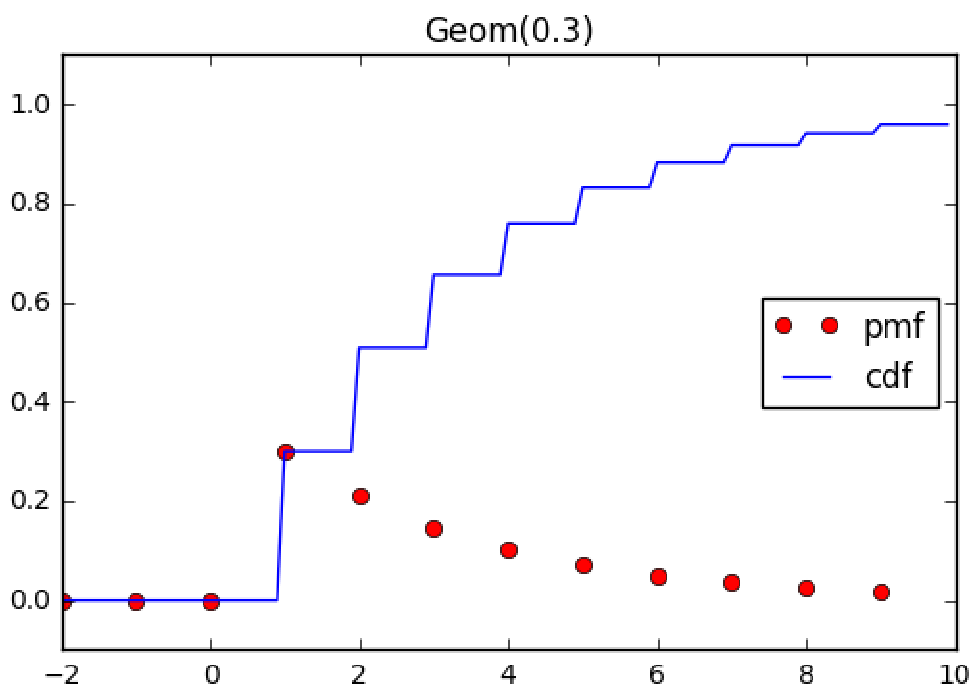
```
x1 = np.arange(-2, 10, 1.0)
x2 = np.arange(-2, 10, 0.1)
y = P1.pmf(x1)
z = P1.cdf(x2)
plt.plot(x1, y, 'or', label='pmf')
plt.plot(x2, z, '-b', label='cdf')
plt.legend(loc=5)
plt.ylim(-0.1, 1.1)
plt.show()
```

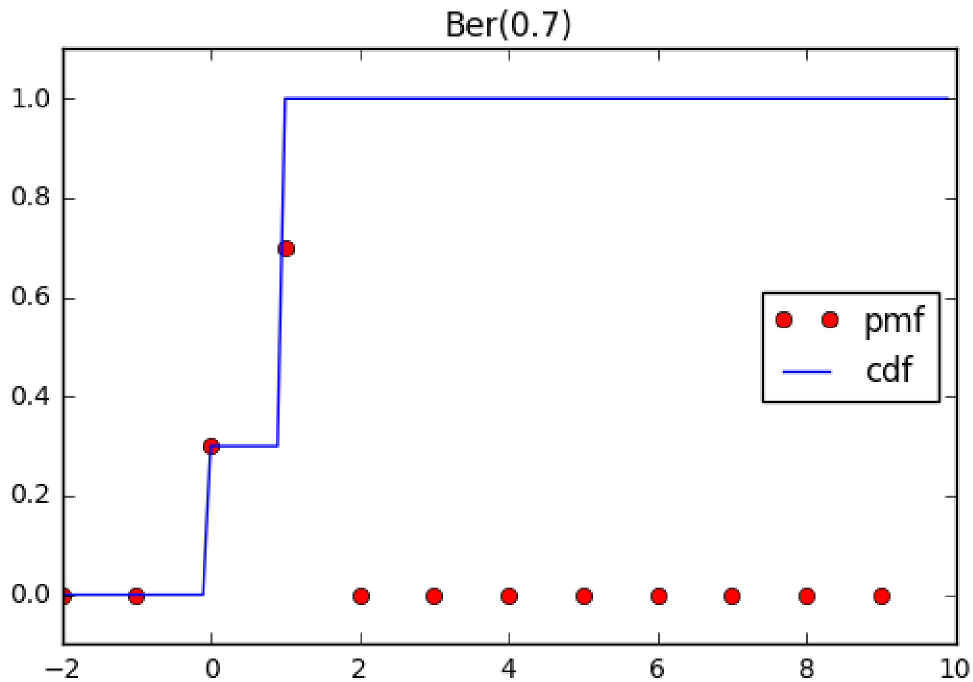
If there are some functions that you do not know in the above code, I recall that you can access their documentation with the question mark `?`. - Could you interpret what you see on the graphics? - Reproduce similar graphics for `P2`, `P3` and `P4` - Add the titles `"Binom(10, 0.3)"`, `"Poisson(2.3)"`, `"Geom(0.4)"` and `"Ber(0.7)"`. - Save these graphics to four files called respectively `binomial.pdf`, `poisson.pdf`, `geometric.pdf` and `bernoulli.pdf` (*hint: look at the function `plt.savefig`) - Do you know how you could include these file into a latex document?

```
In [48]: x1 = np.arange(-2, 10, 1.0)
         x2 = np.arange(-2, 10, 0.1)
         for P,name in discrete_probas:
             y = P.pmf(x1)
             z = P.cdf(x2)
             plt.plot(x1, y, 'or', label='pmf')
             plt.plot(x2, z, '-b', label='cdf')
             plt.legend(loc=5)
             plt.title(name)
             plt.ylim(-0.1, 1.1)
             plt.show()
```




Pois(2.3)
pmf
cdf
1.0
0.8
0.6
0.4
0.2
0.0
−2
0
2
4
6
8
10


Geom(0.3)
pmf
cdf
1.0
0.8
0.6
0.4
0.2
0.0
−2
0
2
4
6
8
10



In [ ]:

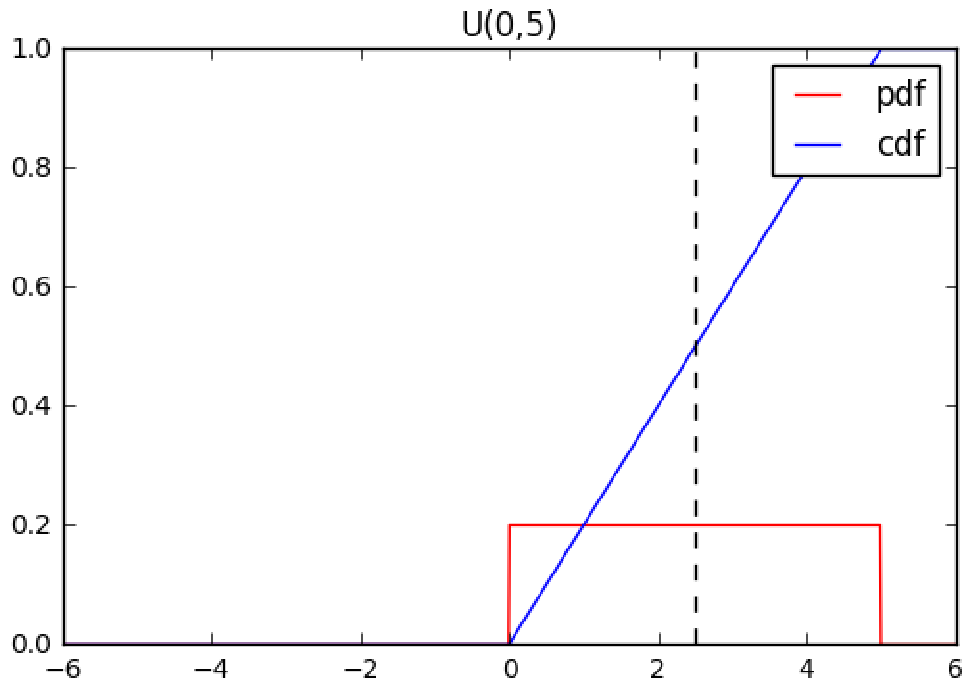
1.2 Plotting continuous probability measure

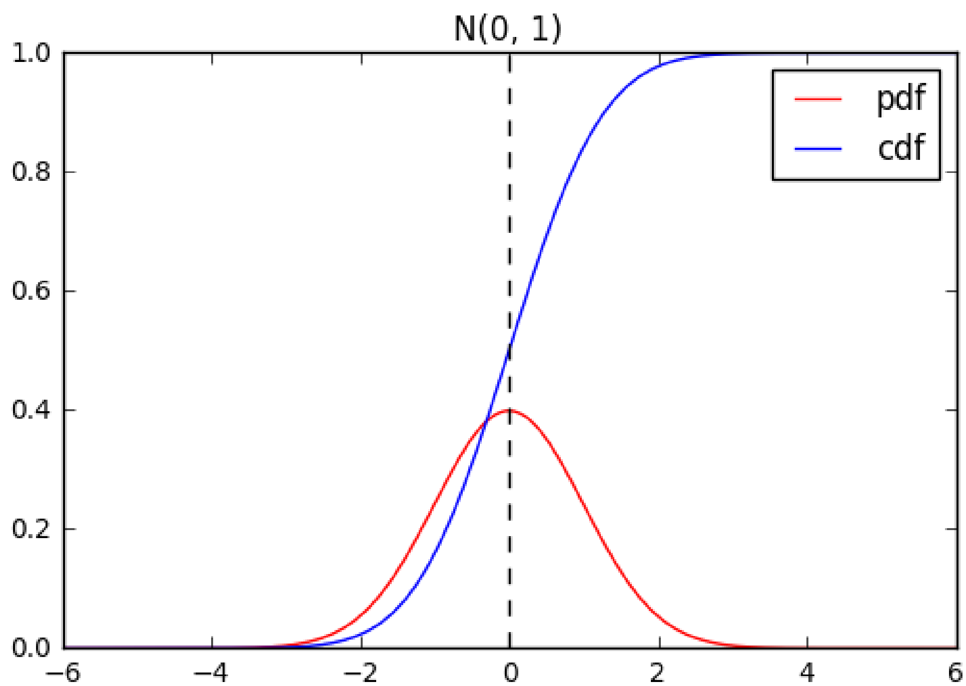
For continuous probability measures the main methods are - `pdf`: probability density function - `cdf`: cumulative density function (also called cumulative distribution function in Lecture 2 of J. Nzabanita)

Exercise: - Set `U` to be the uniform probability measure on $[0,5]$. Plot on the same graphics its density (in red) and its cumulative density function (in lue). - Let `N` be the standard normal distribution ($\mu = 0$ and $\sigma^2 = 1$). Plot on the same graphics its density (in red) and its cumulative density function (in blue). - On both graphics add a black vertical dashed line at the position of the mean.

In [53]:
```
U = stats.uniform(0, 5)
N = stats.norm(0, 1)
continuous_probas = [[U, "U(0,5)"], [N, "N(0, 1)"]]
```

In [60]:
```
x = np.linspace(-6, 6, 1000)
for P, name in continuous_probas:
    plt.plot(x, P.pdf(x), 'r', label='pdf')
    plt.plot(x, P.cdf(x), 'b', label='cdf')
    m = P.mean()
    plt.plot([m, m], [0, 1], '--k')
    plt.title(name)
    plt.legend()
    plt.show()
```






```
In [ ]:
```