

Week1_Lab_Random_Variables

January 4, 2018

Queuing theory: from Markov chains to multiserver systems

Python Lab

Week I: simulation of random variables

In this first lab, we are going to introduce basics of random variables simulation, focusing on the simulation of [exponential](#) and [Poisson](#) distributions, that play a central role in mathematical modelling of queues.

We will see how to draw samples from distributions by the [inverse transform sampling method](#) or by using the [Statistics sublibrary of Scipy](#). We will use the inverse transform sampling method to draw samples from the exponential distribution. Then we will introduce the Poisson distribution.

As explained in the general introduction to the labs (Week 0), to complete a lab, you will have to fill in undefined variables in the code. Then, the code will generate some variables named **Vi**, with **i=1,...** . You will find all the **Vis** generated from your results by running the last cell of code of the lab.

You can check your results by answering to the exercise at the end of the lab section where you will be asked for the values of the **Vis**.

Let $F(x) = P(X \leq x)$ denote the distribution function of some random variable X . When F is continuous and strictly monotone increasing on the domain of X , then the random variable $U = F(X)$ with values in $[0, 1]$ satisfies

$$P(U \leq u) = P(F(X) \leq u) = P(X \leq F^{-1}(u)) = F(F^{-1}(u)) = u, \quad \forall u \in [0, 1].$$

Thus, U is a uniform random variable over $[0, 1]$, what we note $U \sim \mathcal{U}_{[0,1]}$. In other words, for all a, b , with $0 \leq a \leq b \leq 1$, then $P(U \in [a, b]) = b - a$. Conversely, the distribution function of the random variable $Y = F^{-1}(U)$ is F when $U \sim \mathcal{U}_{[0,1]}$.

1) Based on the above results explain how to draw samples from an $\text{Exp}(\lambda)$ distribution. Draw $N = 10^5$ samples of an $\text{Exp}(\lambda)$ distribution, for $\lambda = 2$.

Calculate the mean m and variance σ^2 of an $\text{Exp}(\lambda)$ distribution and compute the sample estimates of m and σ^2 .

```
In [ ]: %matplotlib inline
        from pylab import *
```

```

In [ ]: N = 10**5
        lambda_ = 2.0

        #####
        # Supply the missing coefficient herein below
        V1      = ...
        data     = V1*log(rand(N))
        #####
        m        = mean(data)
        v         = var(data)
        print("\u03BB={0}: m={1:1.2f}, \u03C3\u00B2={2:1.2f}"
              .format(lambda_,m,v)) #\u... for unicode characters

```

2) The discrete valued random variable X follows a [Poisson distribution](#) if its probabilities depend on a parameter λ and are such that

$$P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}, \quad \text{for } k = 0, 1, 2, \dots$$

We denote by $\mathcal{P}(\lambda)$ the Poisson distribution with parameter λ .

As for continuous valued distributions, samples from discrete distributions can be obtained via the inverse transform sampling method. Alternatively, one can use the *statistics* sublibrary of *Scipy* to draw samples from the a Poisson distribution (<https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.poisson.html>).

Draw $N = 10^5$ samples from the Poisson distribution with parameter $\lambda = 20$ and compute their sample mean and variance. Check that they are close to their theoretical values that are both equal to λ

0.0.1 Answer to question 2

```

In [ ]: from scipy.stats import poisson

        lambda_ = 20
        N        = 10**5
        #####
        # Give parameters mu and size in function poisson.rvs
        # (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.poisson.html)
        sample = poisson.rvs(mu= ..., size= ...)
        #####
        # mean and variance of sample vector
        mean_sample = mean(sample)
        var_sample  = var(sample)
        print((" \u03BB = {0} \nestimated mean = {1:1.2f} \n"
              + "estimated var = {2:1.2f} ")
              .format(lambda_,mean_sample, var_sample))

        #-----
        V2 = mean_sample

```

1 Your answers for the exercise

```
In [ ]: print("-----\n"
            +"RESULTS SUPPLIED FOR LAB 1:\n"
            +"-----")
results = ("V"+str(k) for k in range(1,3))
for x in results:
    try:
        print(x+" = {0:.2f}".format(eval(x)))
    except:
        print(x+": variable is undefined")
```