Ecole Normale Supérieure de Cachan
Master recherche 2ème année
**Mathématiques Vision Apprentissage**

# "Stochastic methods for image and signal analysis" Td no. 3

---

## Exercise 1: Simulating discrete random variables with Matlab

In order to understand what kind of features are captured by a stochastic model, one must have a method to get samples from this model and see what they look like. So, one of the main required techniques is to be able to sample from a wide variety of probability distributions. In this exercise, we will study sampling from some discrete distributions, some of which will be used later. The starting point of all the sampling methods will be sampling the uniform distribution on $[0, 1]$. Matlab and all other computer languages have built-in algorithms for doing this.

### Sampling from the binomial distribution

1. How can one use the Matlab function `rand`, to get samples from a Bernoulli distribution of parameter $p \in [0, 1]$ ? We recall that such a distribution is a probability distribution on $\{0, 1\}$, that it is usually denoted by $b(p)$ and that it is defined by $P(\mathcal{X} = 1) = p = 1 - P(\mathcal{X} = 0)$.

2. Use this to sample from a binomial distribution of parameters $n$ and $p$. Recall that such a random variable can be obtained as the sum of $n$ independent Bernoulli random variables with same parameter $p$. Get $N$ samples from the binomial (by using a matrix, and *not* a "for" loop) and compare on the same figure the histogram of the values obtained and the binomial distribution itself.

### Sampling from a Poisson distribution

1. Let $\mathcal{X}$ be a random variable following the uniform distribution on $[0, 1]$. Prove that the random variable $\mathcal{Z} = -\log \mathcal{X}$ has the density function given by $e^{-x}$ on $\{x \geq 0\}$. (This is the exponential distribution of parameter 1).

2. Prove by induction on $k$, that if $\mathcal{Z}_1$, $\mathcal{Z}_2$,... $\mathcal{Z}_k$ are $k$ independent random variables with exponential distribution of parameter 1, then $\mathcal{Z}_1 + \mathcal{Z}_2 + \ldots \mathcal{Z}_k$ has density function $x^{k-1}e^{-x}/(k-1)!$ on $\{x \geq 0\}$. (This is the gamma distribution of parameters $(k, 1)$).

3. Let $\lambda > 0$, and let $(\mathcal{X}_n)_{n\geq 0}$ be a sequence of independent random variables uniformly distributed on $[0,1]$. Let then $\mathcal{Y}$ be the random variable defined by

$$\mathcal{Y} = \min\{n \geq 0 \,|\, \mathcal{X}_0 \times ... \times \mathcal{X}_n \leq e^{-\lambda}\}.$$

Prove that the law of $\mathcal{Y}$ is the Poisson distribution $P_\lambda$ of parameter $\lambda$, which means that

$$\mathbb{P}(\mathcal{Y} = k) = \frac{\lambda^k}{k!} e^{-\lambda}.$$

4. Use this to get an algorithm for sampling from the Poisson distribution $P_\lambda$. Obtain $N$ samples and plot on the same figure their histogram and the Poisson distribution.

**The inversion method**

1. The aim of the inversion method is to get samples from any distribution through the inversion of its repartition or cumulative distribution function. In the discrete framework, the method is the following one.
Let $P = (p_0, p_1, \ldots)$ be a probability distribution on $\mathbb{N}$. Let its repartition function $H$ be defined on $\mathbb{R}_+$ by:

$$\forall k \geq 0, \forall x \in [k, k+1) \text{ then } H(x) = p_0 + p_1 + \ldots p_k.$$

Let $\mathcal{U}$ be uniform on $[0,1]$, then prove that the random variable $\mathcal{X}$ defined by $\mathcal{X} = H^{-1}(\mathcal{U}) = \inf\{k \in \mathbb{N} \,|\, \mathcal{U} \leq H(k)\}$ follows the distribution $P$.

2. Use this to sample from a geometric distribution of parameter $p$ given by $\forall k \geq 1$, $\mathbb{P}(\mathcal{X} = k) = (1-p)^{k-1}p$. Obtain $N$ samples in this way and plot on the same figure the histogram of the obtained values and the geometric distribution.

## Exercise 3: Analyzing $n$-tuples in some database

1. Surfing the web (for instance on http://www.fullbooks.com/), find at least 1 megabyte (5+ is better) of English text. It should be `ascii`, not `html`, so you have the sequence of characters in hand. Below is some MatLab code to do the tedious job of converting an `ascii` text into a string of numbers from 1 to 27 by (i) stripping out all punctuation and numerals, (ii) converting all upper and lower case letters to the numerals 1 to 26, (iii) converting all line returns, tabs to spaces and (iv) collapsing all multiple spaces to single spaces represented by the number 27. The result is one long string of the numbers 1 through 27.

```
label = fopen('mytext.txt');
[hk,count] = fread(label,'uchar');
fclose(label);
F = double(hk);
F(F<33) = 27;                          % line feeds, tabs, spaces --> 27
```

```
F(F>64 & F<91) = F(F>64 & F<91)-64;     % u.c. letters --> [1,26]
F(F>96 & F<123)=F(F>96 & F<123)-96;     % l.c. letters --> [1,26]
F = F(F<28);                            % Throw out numbers, punctuation
F2 = [0;F(1:(size(F)-1))]+F;            % Add consecutive values
G = F(F2<54);                           % The text with one space between words
```

2. Calculate the empirical probability $P_1$ of all single letters (including space) and all consecutive letter pairs $P_2$ and print these out. Check that some obvious things hold, e.g. the conditional probability of 'u' after a 'q' is essentially 1. Now calculate the entropy of both distributions and of the conditional distribution of the second letter, given the first. Compare with Shannon's results in the notes. Depending on your data set, there may be differences; if so, can you see any reasons for differences?

3. Now calculate the distributions $P_3$ on triples and $P_4$ on 4-tuples of consecutive characters. For 4-tuples, you may run into memory and speed problems, so careful MatLab coding is essential. Again calculate the entropies of these distributions.

4. Now play Shannon's game: make new texts, say 200 characters long, by sampling the $n$-gram models (for $n = 0, 1, 2, 3$ and 4): (i) take a random sample of $n$ symbols from your text and (ii) again and again, sample from the conditional distribution $P_n(a_n|a_1...a_{n-1})$ to extend the series. The results, e.g. with the Bible or with IBM technical reports, should be quite suggestive.

5. Finally, process your text to eliminate all spaces! Can we recover the word boundaries? To make prevent problems with memory, take several random chunks, say 80 characters long, of your text and, for each chunk, calculate the mutual information of consecutive pairs, i.e. $\log_2(P_4(abcd)/P_2(ab)P_2(cd))$. Make a figure, as in the notes, showing the graph of this function on top of the actual words (see code below) and see to what extent the local minima predict the word boundaries.

Here is the MatLab code to spare you annoying details:

```
% CODE TO ELIMINATE SPACES
G = G(G<27);

% CODE TO MAKE NICE OUTPUT PLOTTING ARRAY MI ABOVE ACTUAL CHARCATERS
hold off, plot(MI((1:80)));
axis([0 81 -20 20]); axis off, hold on
plot([0 81], [0 0]); plot([0 81], [4 4]);
text(0,0,'0 bits');  text(0,4,'4 bits')
for i=1:80
   text(i+0.15,-2,char(96+G(i)))        % ascii codes 97,..,122 are letters
end
```