

Les méthodes Monte-Carlo par Chaîne de Markov (MCMC)



INTRODUCTION

Simulation de
loi

Découverte
du décryptage

Finalisation
du projet



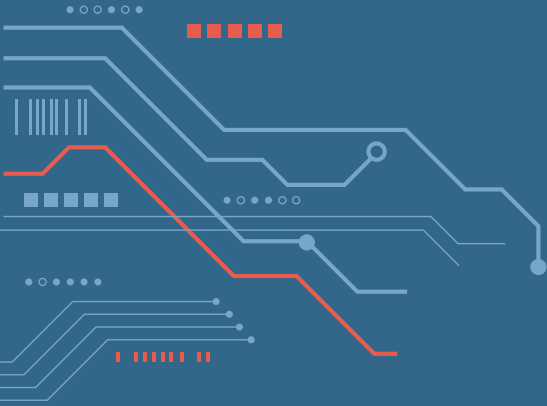
SOMMAIRE



01 Simulation d'une loi
par méthodes MCMC

02 Cryptographie grâce
aux méthodes MCMC





01

I I I I I I I I

SIMULATION D'UNE LOI PAR METHODES MCMC

Introduction aux méthodes MCMC

Loi posteriori : $\pi(\theta|x) = \frac{f(x|\theta) \pi(\theta)}{f_X(x)}$

Problème : $f_X(x) = \int f(x | \theta) \pi(\theta) d\theta$ **Incalculable**



Utilisation des méthodes MCMC

L'algorithme de Metropolis-Hasting

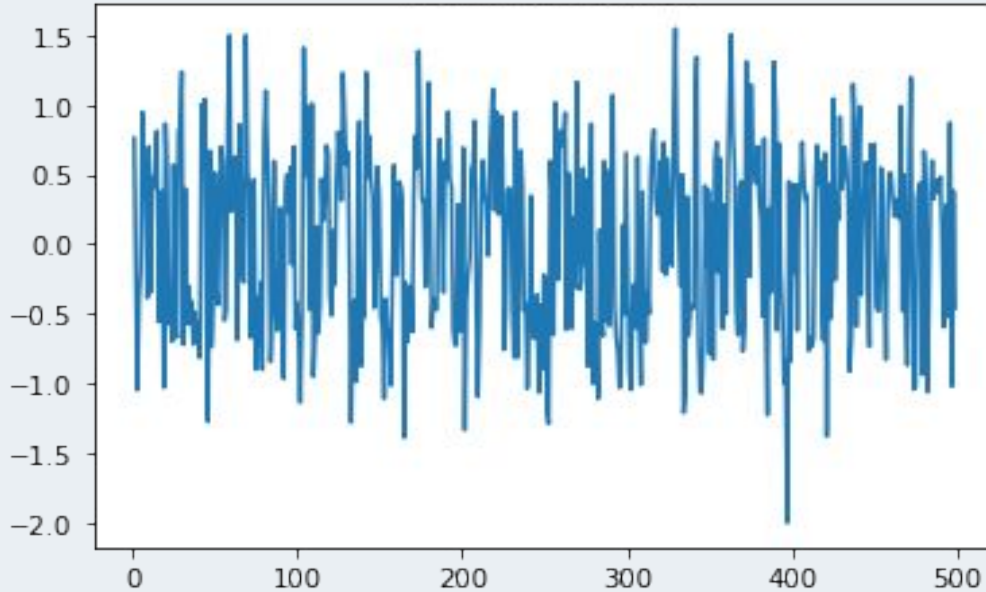
1. Générer $y_{t+1} \sim q(x_t, \cdot)$
2. Calculer la *probabilité d'acceptation*

$$\alpha(x_t, y_{t+1}) = \min \left[\frac{\pi(y_{t+1})q(y_{t+1}, x_t)}{\pi(x_t)q(x_t, y_{t+1})}, 1 \right]$$

3. Prendre $x_{t+1} = \begin{cases} y_{t+1} & \text{avec probabilité } \alpha \\ x_t & \text{avec probabilité } 1 - \alpha \end{cases}$



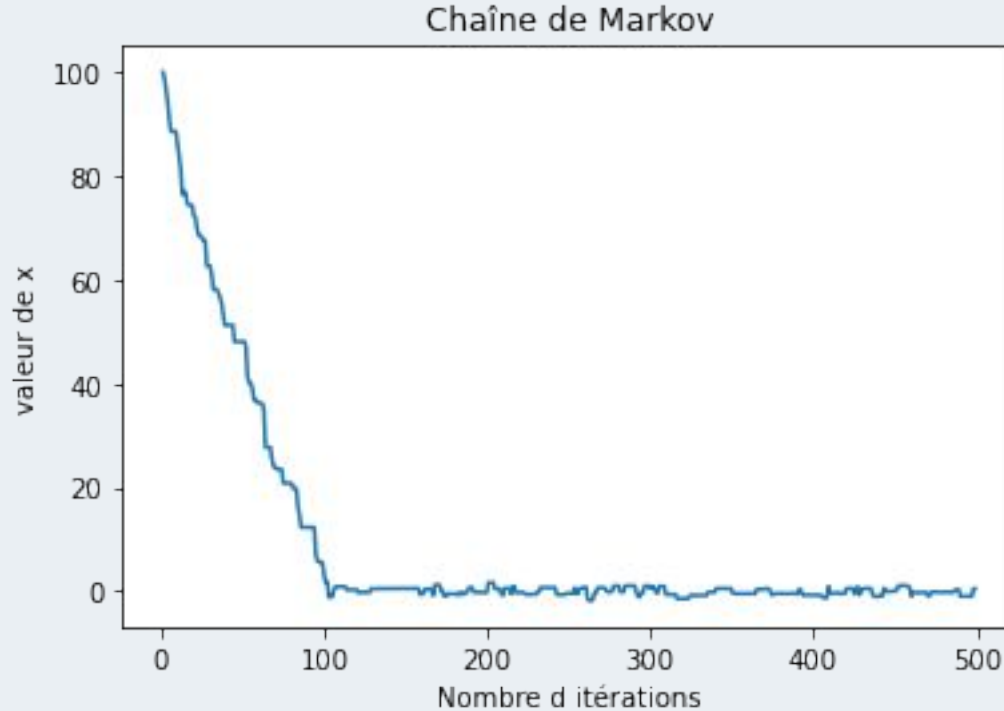
Éviter la corrélation (THINNING)



Ici Thinning = 100



Éviter la corrélation (BURN-IN)



Ici Burn-in = 100

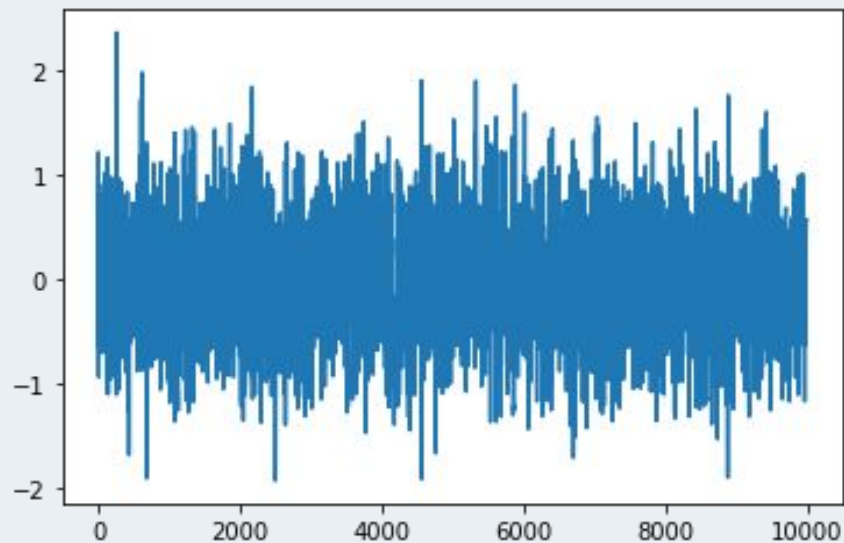
Implémentation

Loi posteriori : $\pi(\theta | x) = \exp(-b * |x|) * |x|^{(a-1)}$, $a = 4$ et $b = 7$

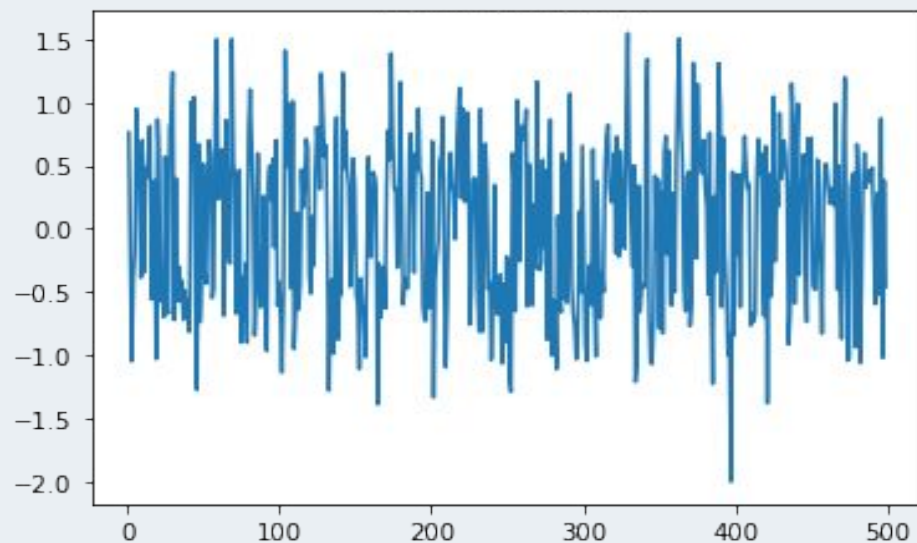
	Série 1	Série 2
Nb itérations	10000	10000
Burn-in	0	500
Thinning	1	10
Variance	5	0.4
Espérance	0.1	-0.02



Chaine de Markov

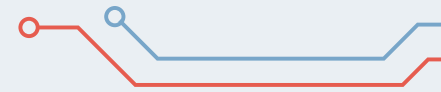


Série 1

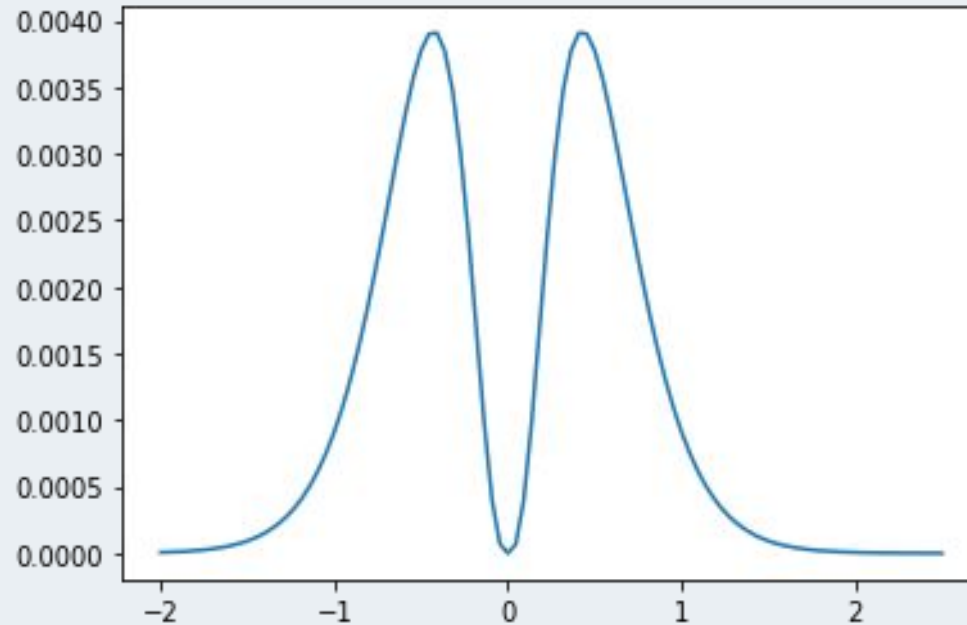


Série 2

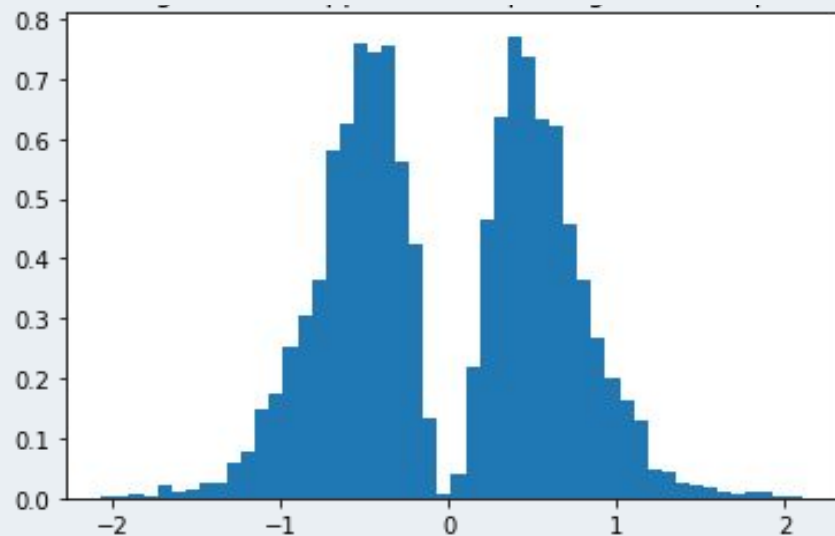




Fonction densité de la loi Gamma

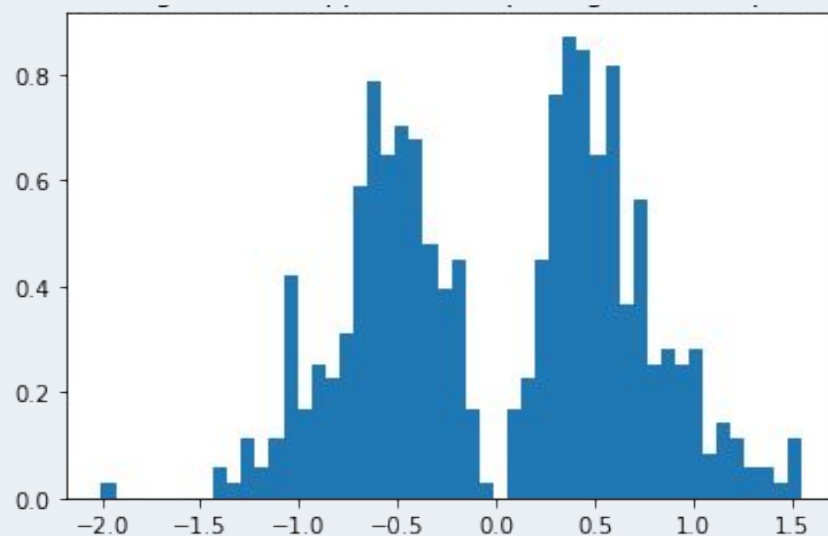


Histogrammes



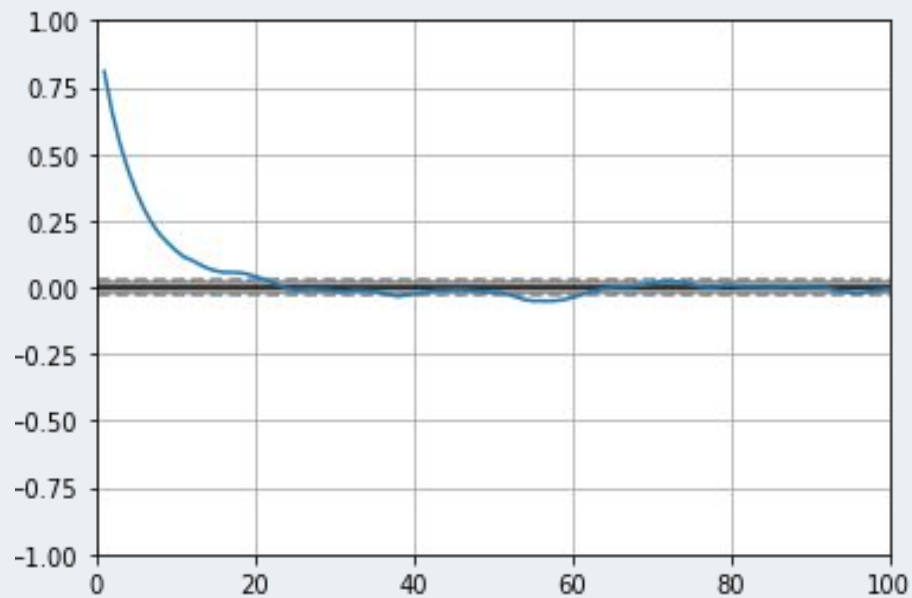
Série 1

I I I I I I I I

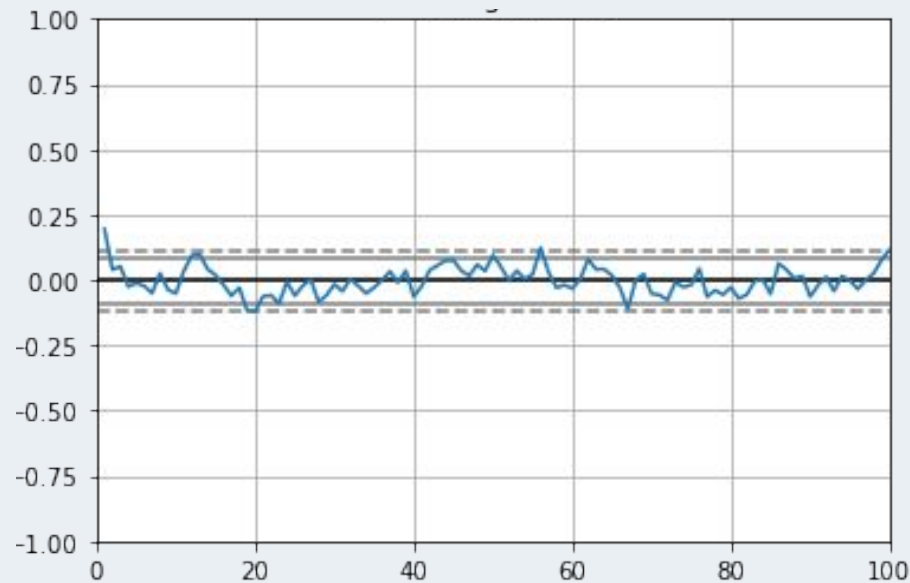


Série 2

Corrélogrammes

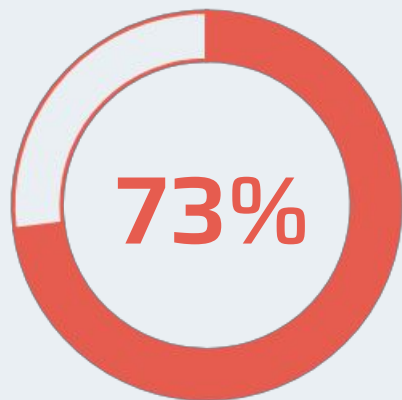


||| ||||| || Série 1

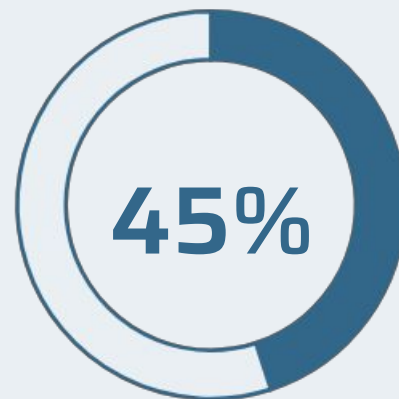


Série 2

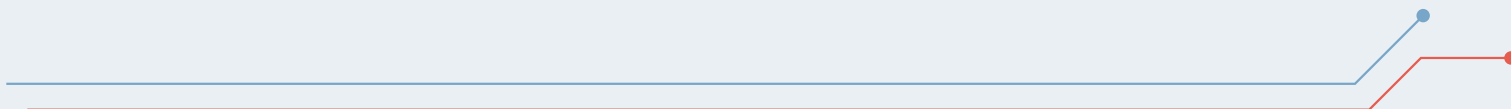
Taux de rejet



Série 1



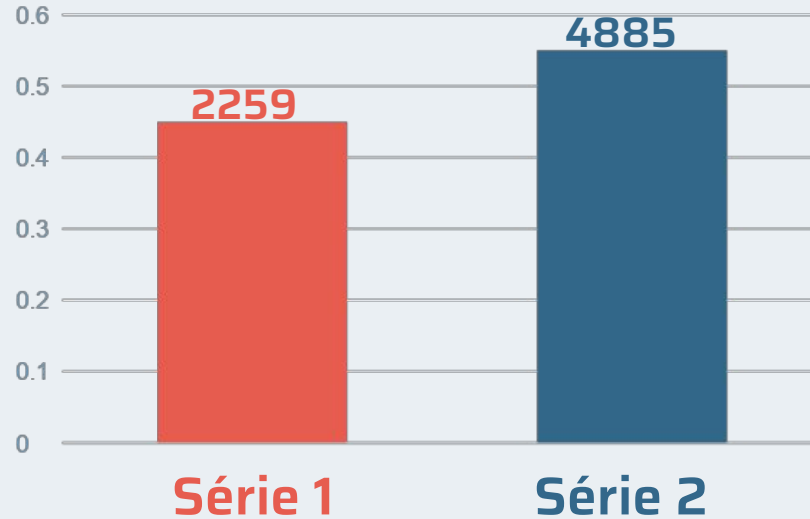
Série 2

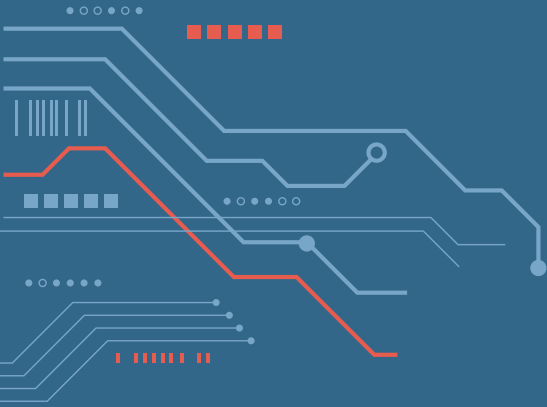


Effective Samples Size (ESS)



$$ESS = \frac{N}{\text{weight}}$$

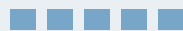




02

I I I I I I I I

DECRYPTAGE GRÂCE AUX MÉTHODES MCMC



Le message crypté du tueur du Zodiaque

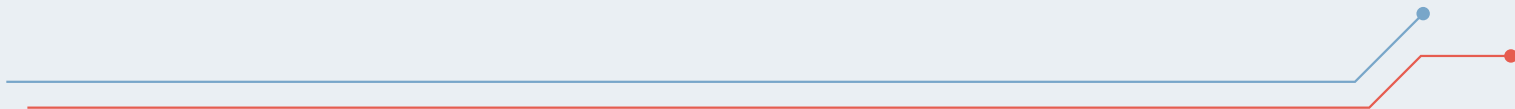
HER>9 JΛVPXI ⊙LT6 ⊙Q
N9+Bϕ ■O▣DWY •< ▣K 7⊕
BΥ⌘∩M+UZ6Wϕ⊕L ■⊕HJ
S99ΔΛJ▲▣V⊙9O++RK⊙
□ΔM+⊕⊥τQI ●FP+P⊙X/
9▲RΛFJO-▣QCXF>⊙Dϕ
■●+K⊙▣⌘⊙U∩X6V •⊕LI
ϕ6⊙J7τ■O+□NY⊕+▣LΔ
Q<M+B+ZR⊙FB∩YA⊙●K



Fréquence de chaque lettre

Du livre « L'appel de l'ange » de Guillaume Musso

A	B	C	D	E	F	G	H	...	Z
39397	3727	13474	15430	68057	4579	4136	4264	...	509

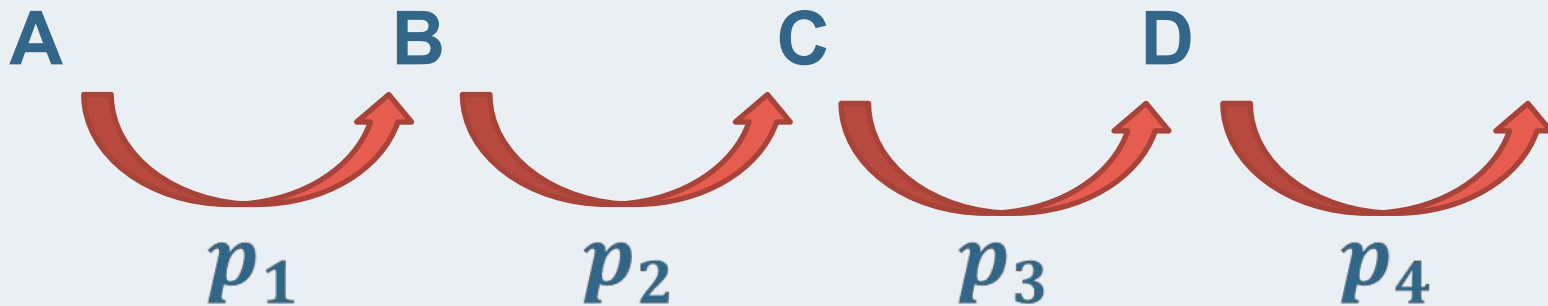




Score de plausibilité

$$Score1(x) = \frac{1}{N} \sum_i \log(p_i + \epsilon)$$

Avec $\epsilon = 10^{-6}$
Et N longueur
de la chaîne



Mise en place d'un message crypté

"Cette année, nous avons un projet important pour valider notre quatrième année d'école d'ingénieur. Nous avons un rapport et un oral pour noter ce projet. Nous avons eu un tuteur de projet spécialiste en statistique pour nous aider. Nous avons choisi d'implémenter un code permettant de déchiffrer un message codé. Nous sommes fiers de comprendre la plupart des messages"

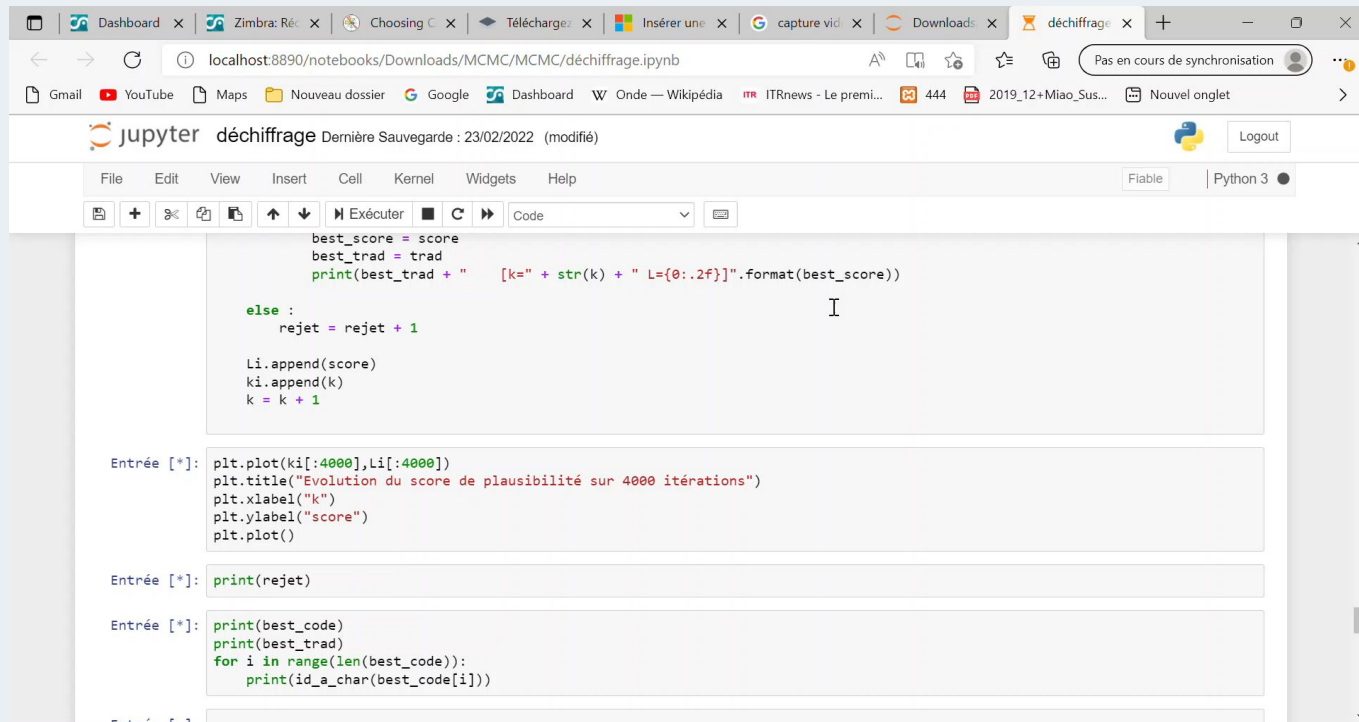


"EYVV MSSVV SNXU MQNSU XS IHNKVY
RCINHYMSY INXH QMPRFVH SNYHV
LXMYHRVCV MSSVV F VENPV F RSZVSRVXH
SNXU MQNSU XS HMIINHY VY XS NHMP
INXH SNYVH EV IHNKVY SNXU MQNSU VX
XS YXYVXH FV IHNKVY UIVERMPRUYV VS
UVMYRUYLXV INXH SNXU MRFVH SNXU
MQNSU EONRUR F RCIPVCVSYVH XS ENFV
IVHCVYMSY FV FVEORBBHVH XS CVUUMZV
ENFVH SNXU UNCCVU BRVHV FV
ENCIHVSFHV PM IPXIMHY FVU CVUUMZVU"

L'algorithme de Metropolis-Hasting pour le décryptage



La traduction du message



```
best_score = score
best_trad = trad
print(best_trad + "    [k=" + str(k) + " L={0:.2f}]" .format(best_score))

else :
    rejet = rejet + 1

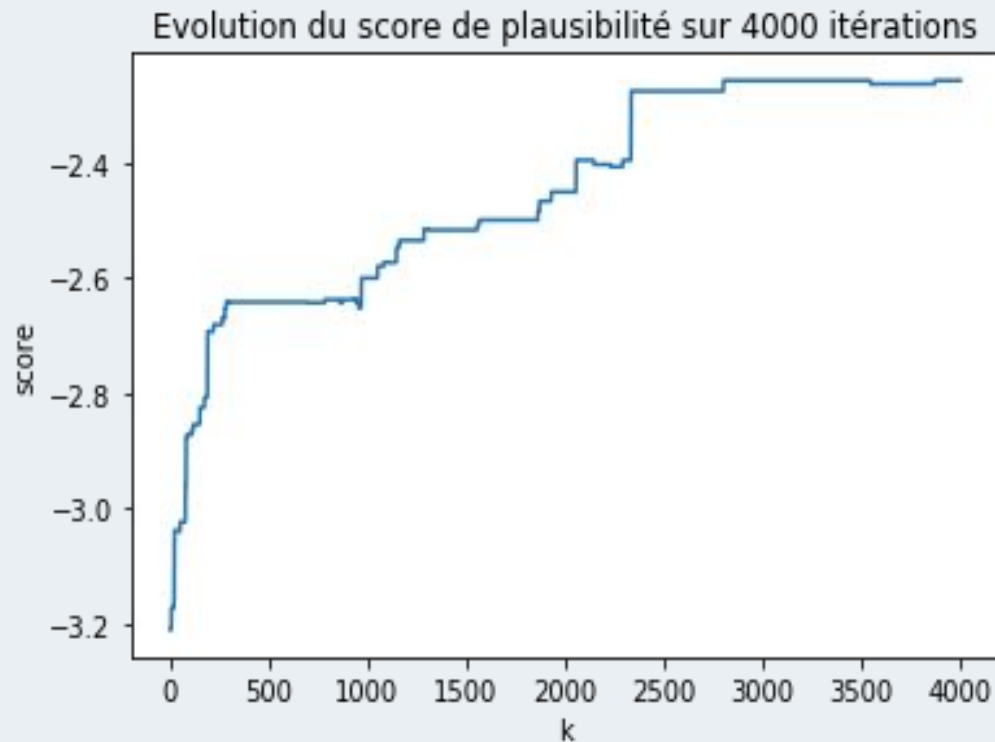
Li.append(score)
ki.append(k)
k = k + 1
```

Entrée [*]: `plt.plot(ki[:4000],Li[:4000])`
`plt.title("Evolution du score de plausibilité sur 4000 itérations")`
`plt.xlabel("k")`
`plt.ylabel("score")`
`plt.plot()`

Entrée [*]: `print(rejet)`

Entrée [*]: `print(best_code)`
`print(best_trad)`
`for i in range(len(best_code)):`
 `print(id_a_char(best_code[i]))`

Analyse de l'évolution du score de plausibilité





Conclusion

Simulation de loi

Difficulté



Plaisir



Décryptage

