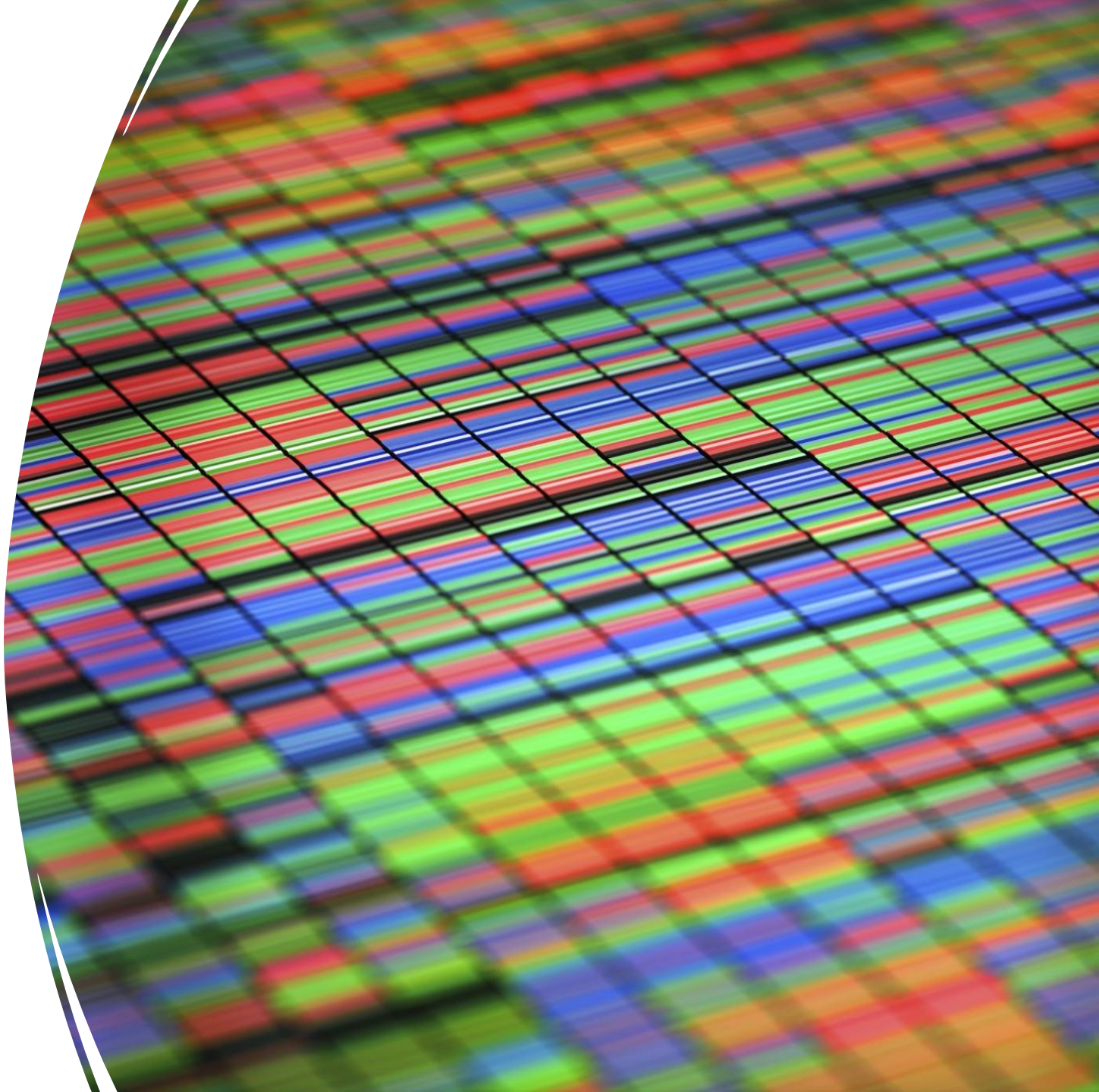
Algorithme de rejet



Introduction

Introduction

. **Contexte :** Générer des v.a avec un générateur de loi uniforme

Introduction

. **Contexte :** Générer des v.a avec un générateur de loi uniforme

. **Connaissances de cours:** méthode de la transformation inverse

Introduction

. **Contexte :** Générer des v.a avec un générateur de loi uniforme

. **Connaissances de cours:** méthode de la transformation inverse

. **Limite :** connaissance nécessaire de la fonction de répartition

Exemple

Exemple

$$f(x) = \begin{cases} e^{-\frac{3}{2}} + e^{-x^2}, x \geq 0 \\ e^{\frac{x}{3}} + e^{x^3}, x < 0 \end{cases}$$

Exemple

$$f(x) = \begin{cases} e^{-\frac{3}{2}} + e^{-x^2}, x \geq 0 \\ e^{\frac{x}{3}} + e^{x^3}, x < 0 \end{cases}$$

$p(x) = \frac{f(x)}{\int_{\mathbb{R}} f(x)dx}$ est bien

une densité de probabilité

Problématique:

Problématique:

Engendrer des variables aléatoires pour toute densité

Problématique:

Engendrer des variables aléatoires pour toute densité

Solution:

Problématique:

Engendrer des variables aléatoires pour toute densité

Solution:

Utiliser une méthode indirecte: l'algorithme de rejet

Explication/Procédure: Premier exemple

Outils : - g *densité candidate*

- $M \in \mathbb{R}$

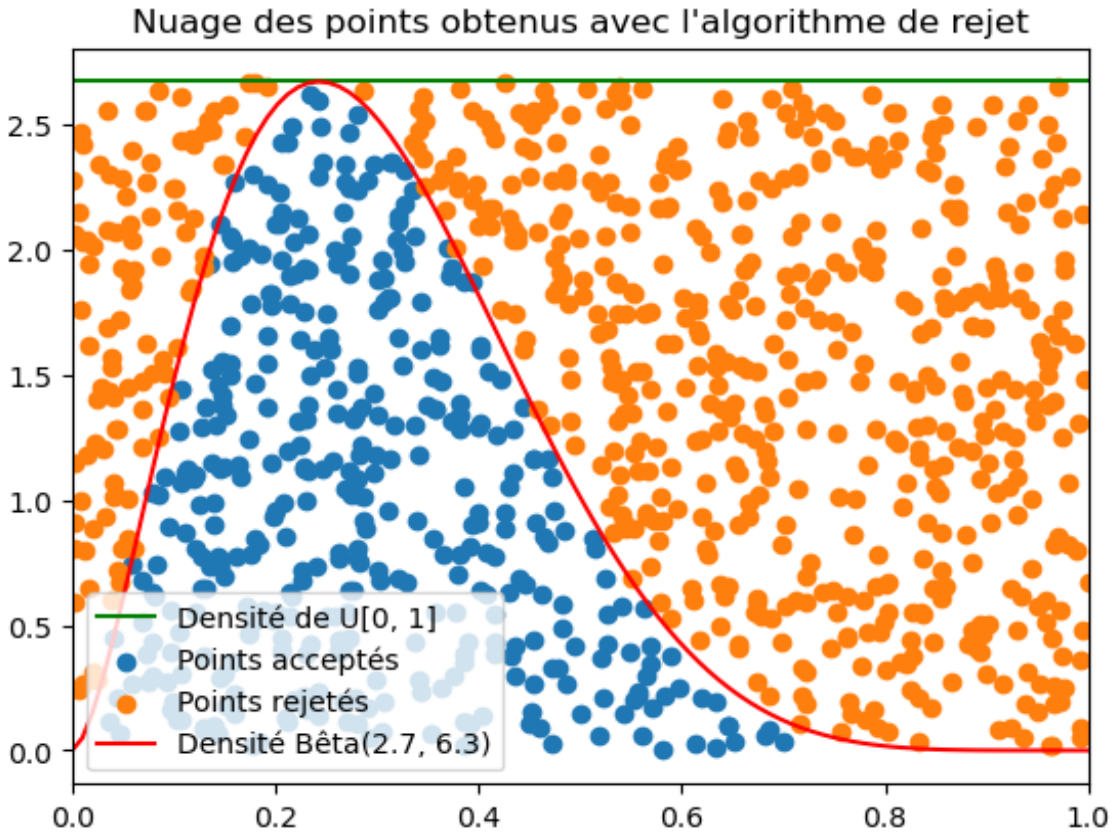
1. $Suite := vrai$
2. **Tant que** *Suite*
   - $Tirer\ U\ selon\ la\ loi\ \mathcal{U}([0,1]), tirer\ Y\ selon\ g$
   - $Z := MUg(Y)$
   - $Suite := \mathbf{SI}(Z \leq f(Y), vrai, faux)$:

   Si *Suite* **alors**
     - $Accepter\ X = Y$

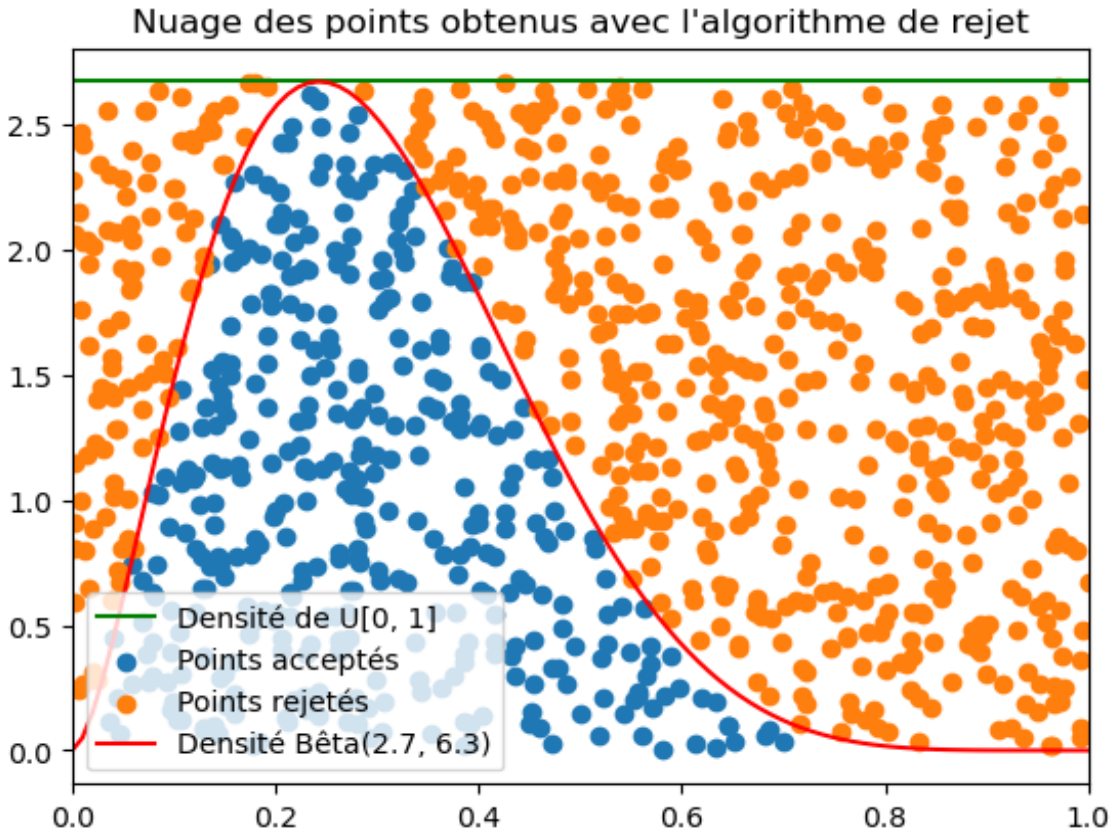
   Fin si
3. **Fin tant que**

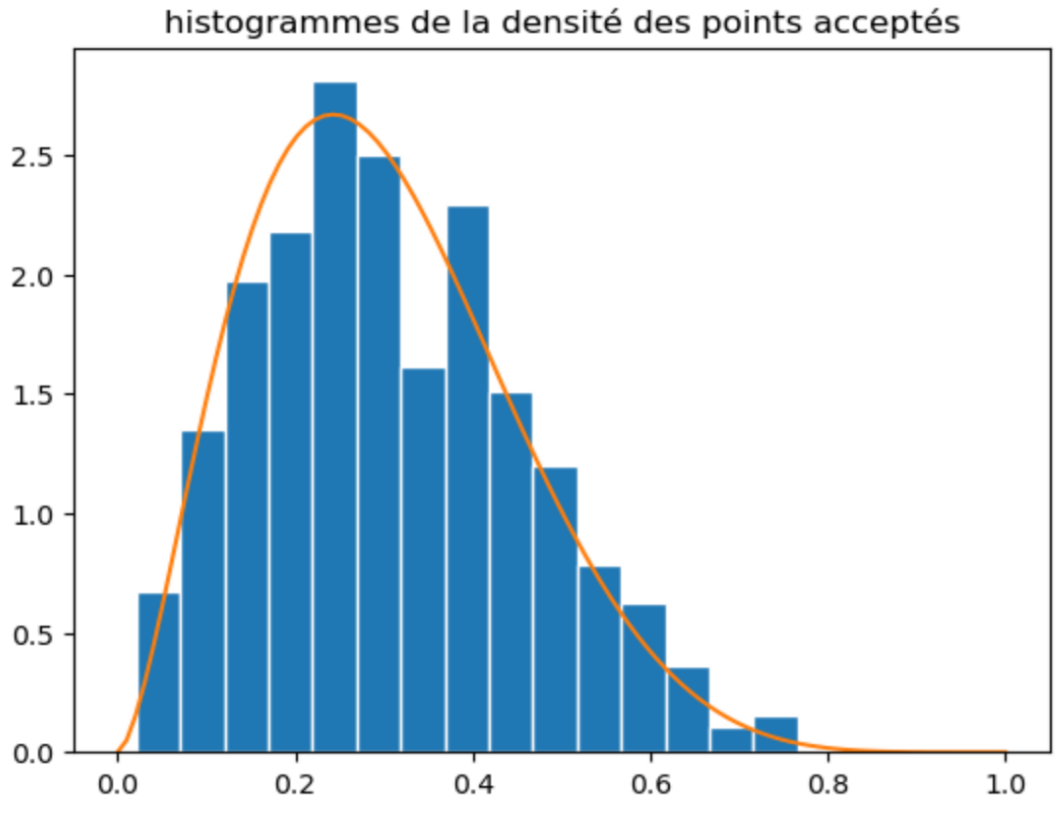
Résultats en Python: Densité candidate g = U[0, 1]

Résultats en Python: Densité candidate g = U[0, 1]

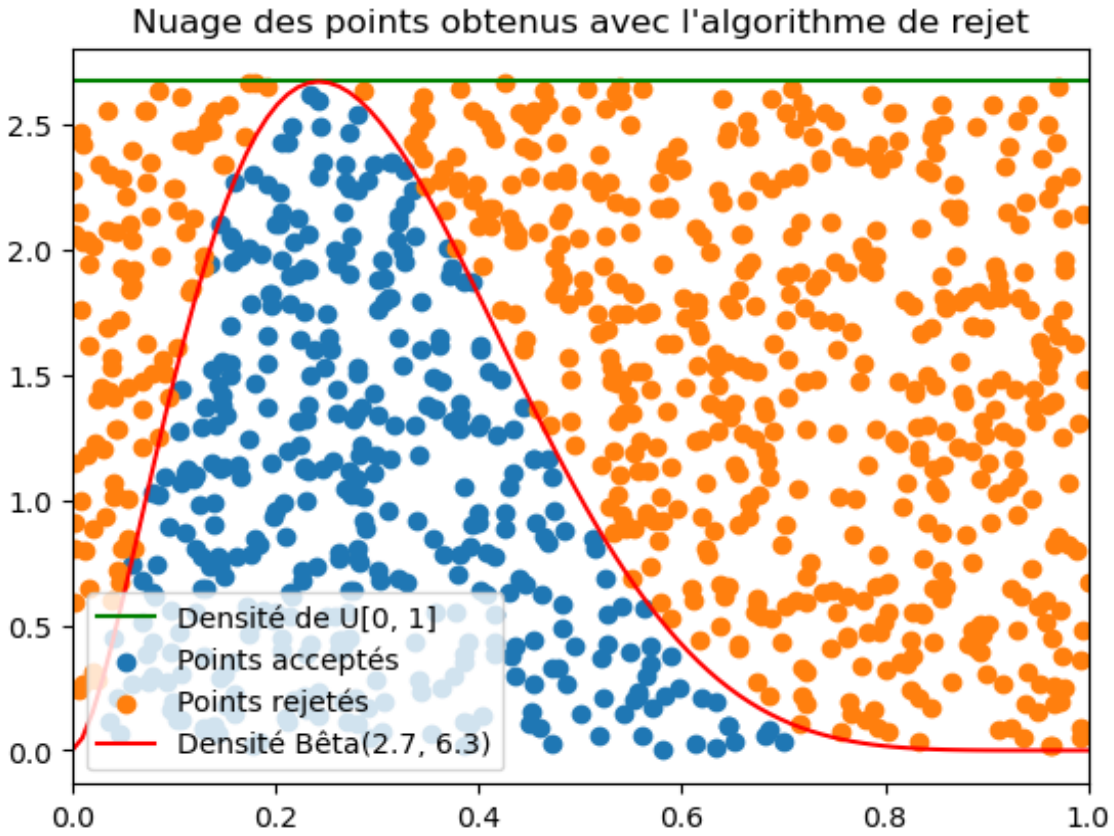


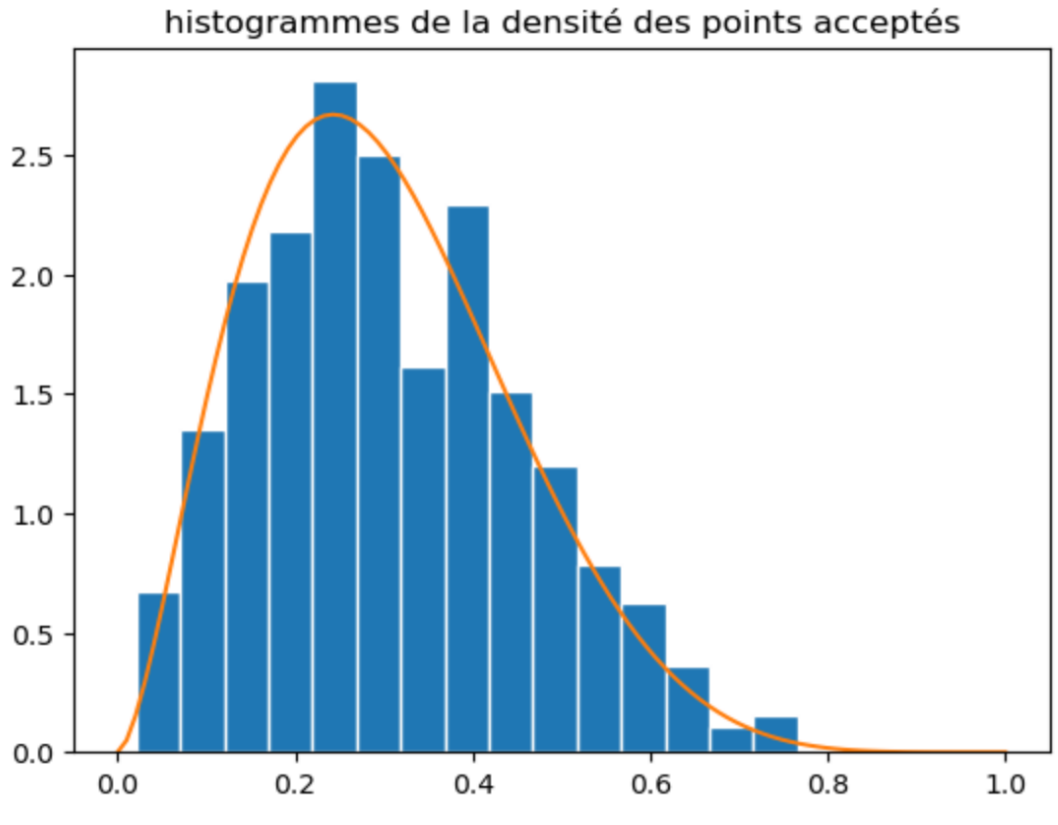
Résultats en Python: Densité candidate g = U[0, 1]





Résultats en Python: Densité candidate g = U[0, 1]

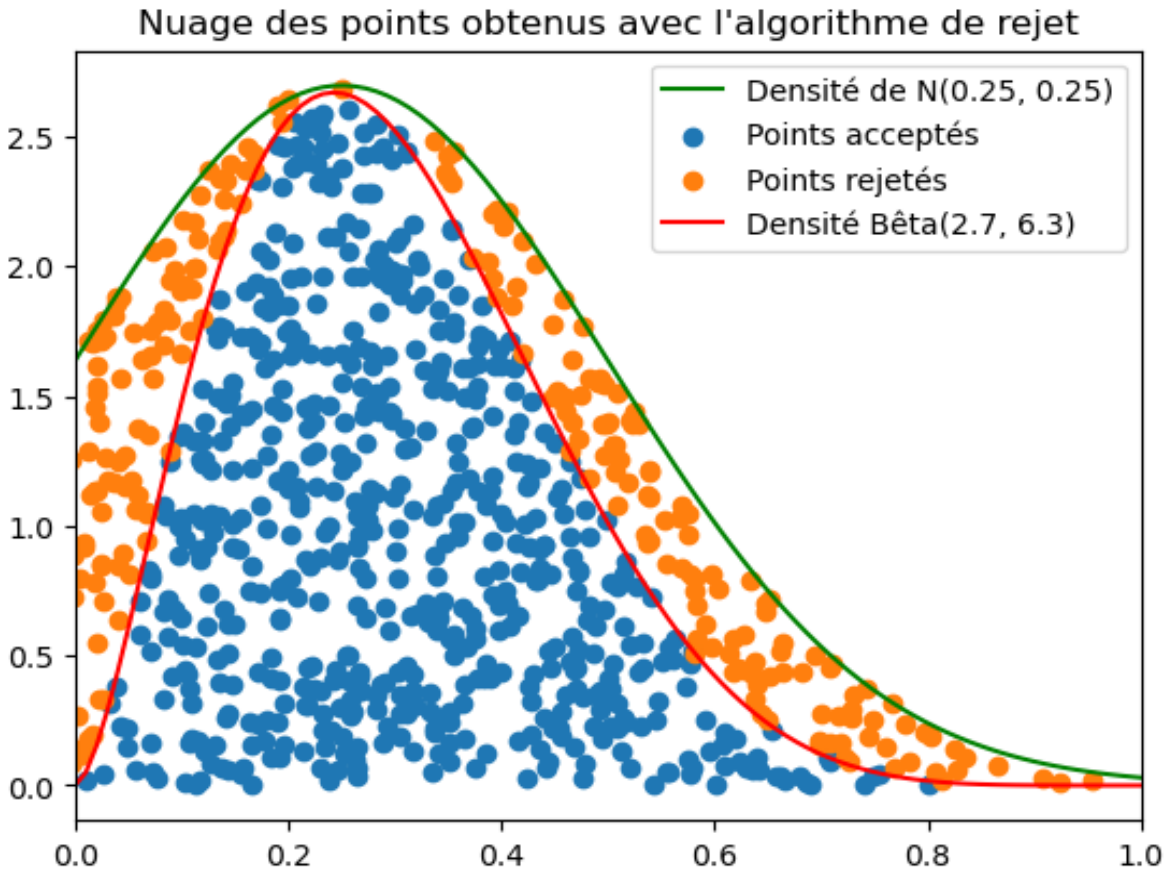




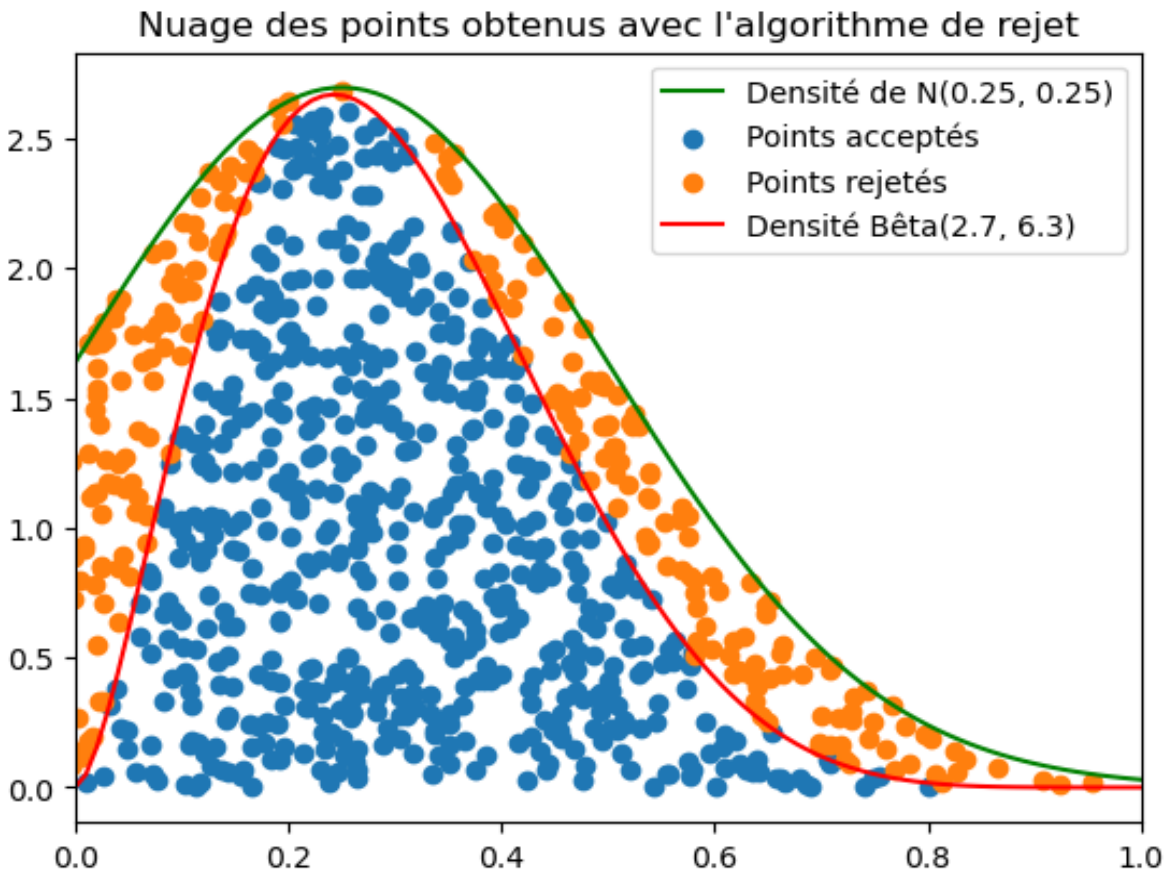
Problème: 60% d'acceptation

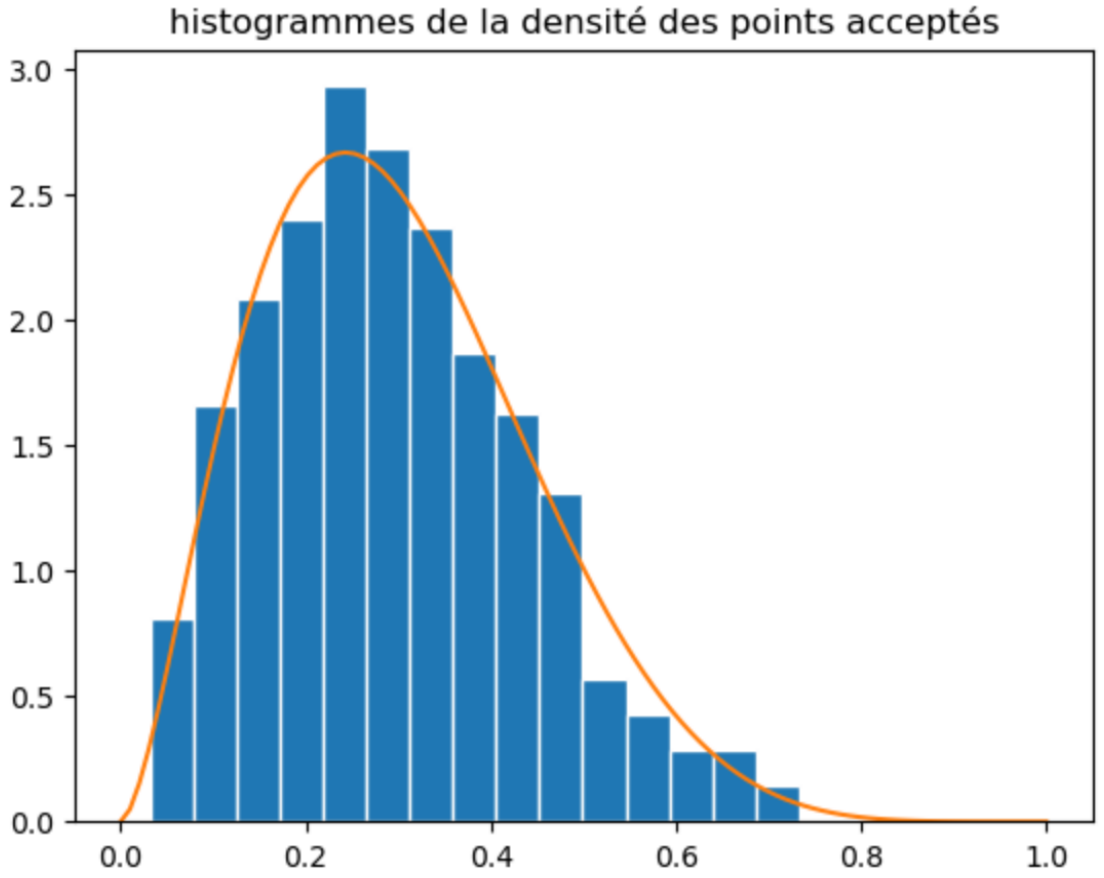
Un bon choix de g: N(0.25, 0.25)

Un bon choix de g: N(0.25, 0.25)

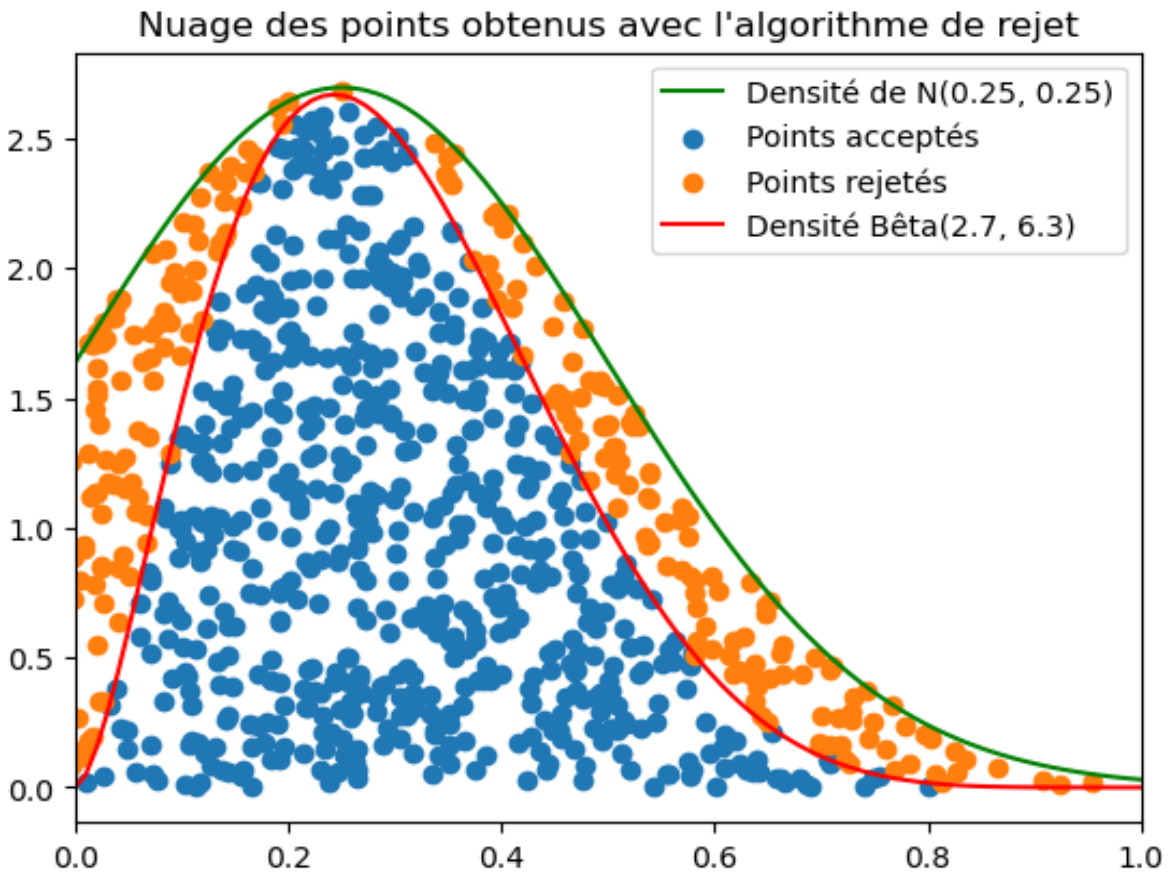


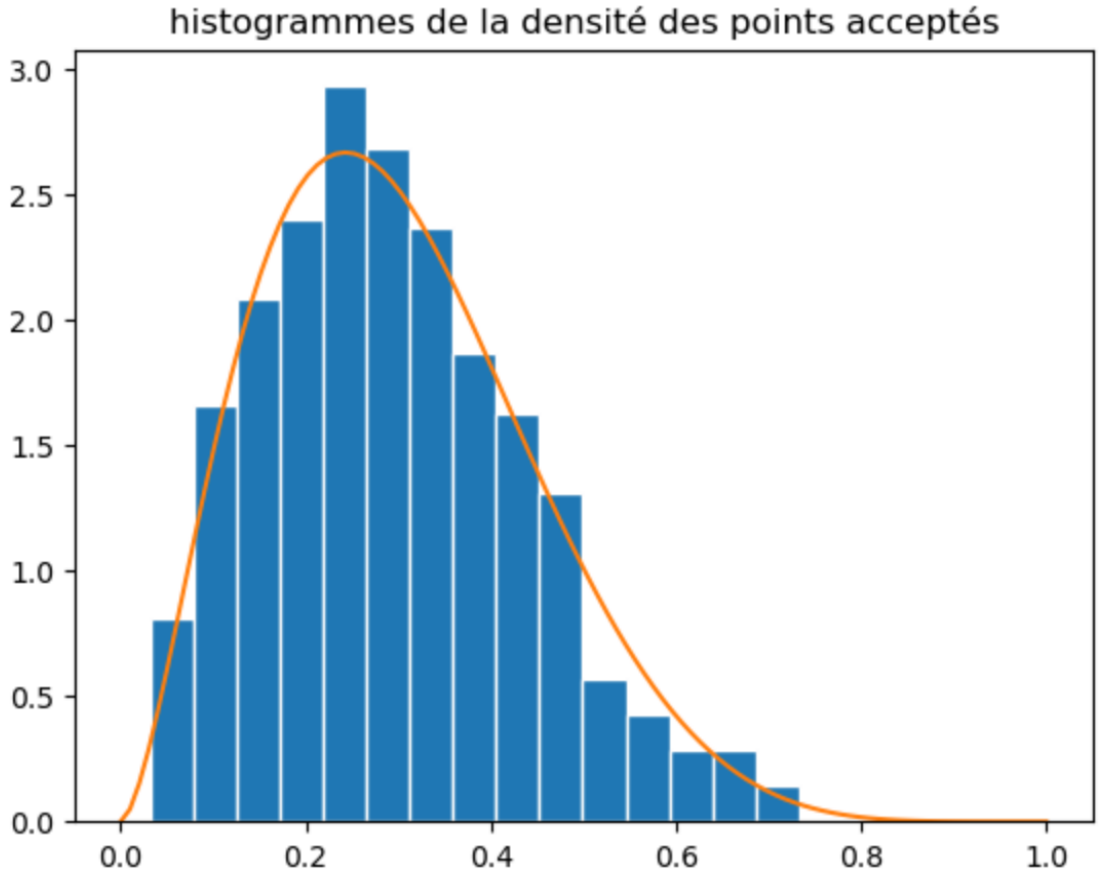
Un bon choix de g: N(0.25, 0.25)





Un bon choix de g: N(0.25, 0.25)





60% d'acceptation

Avantages

Avantages

- **Méthode puissante et simple** : marche pour n'importe quelle distribution

Avantages

- **Méthode puissante et simple** : marche pour n'importe quelle distribution
- **Prérequis** : forme fonctionnelle de f

Limitations

Limitations

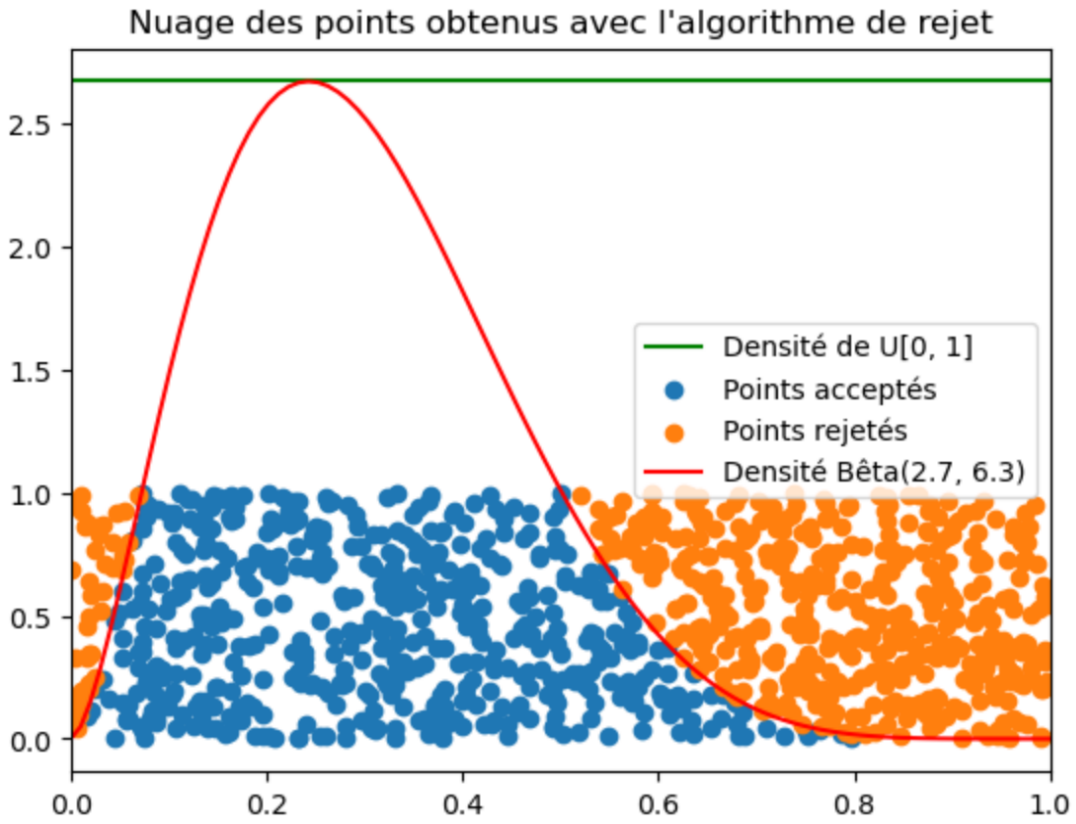
- Pertinence de g et M

Mauvais choix de M

Loi Uniforme [0,1]:

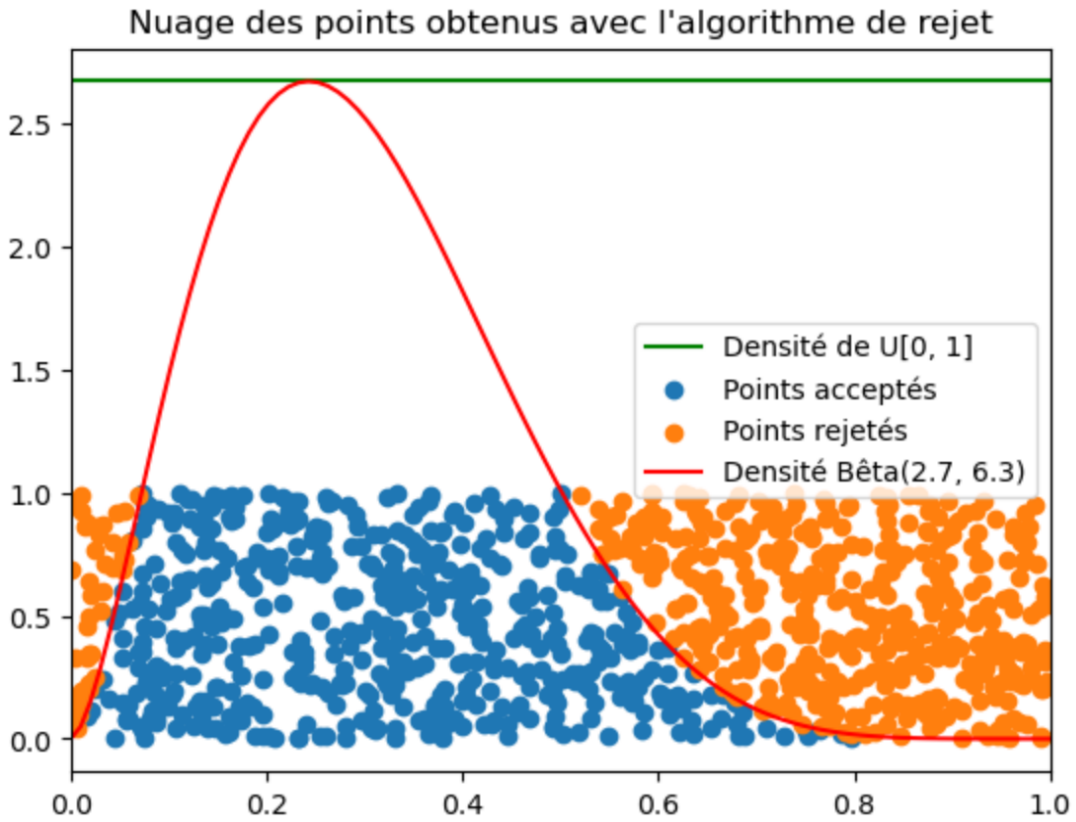
Mauvais choix de M

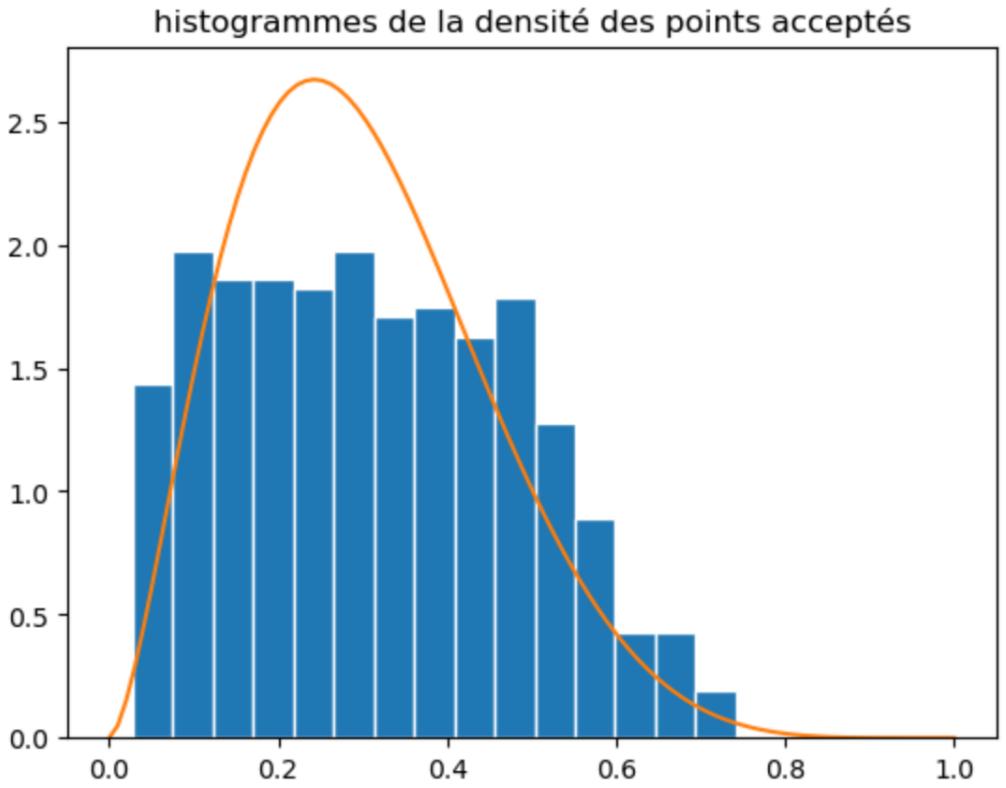
Loi Uniforme [0,1]:



Mauvais choix de M

Loi Uniforme [0,1]:



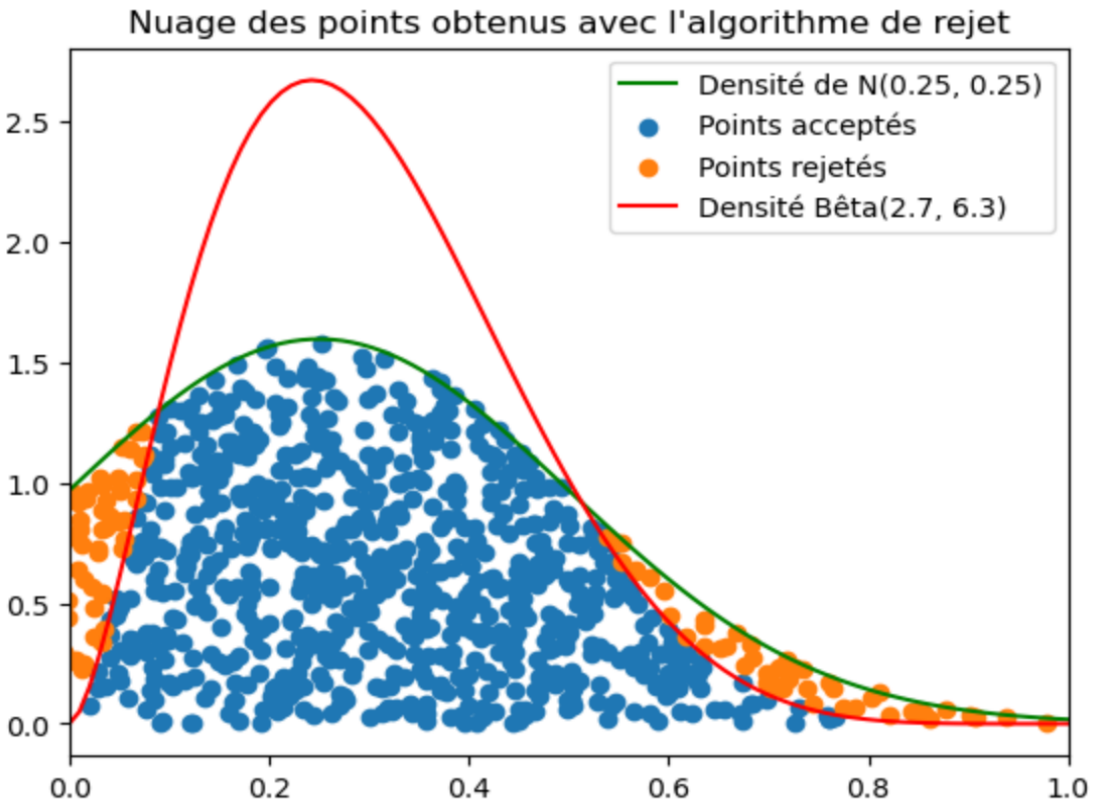


Mauvais choix de M

Loi Normale (0.25,0.25):

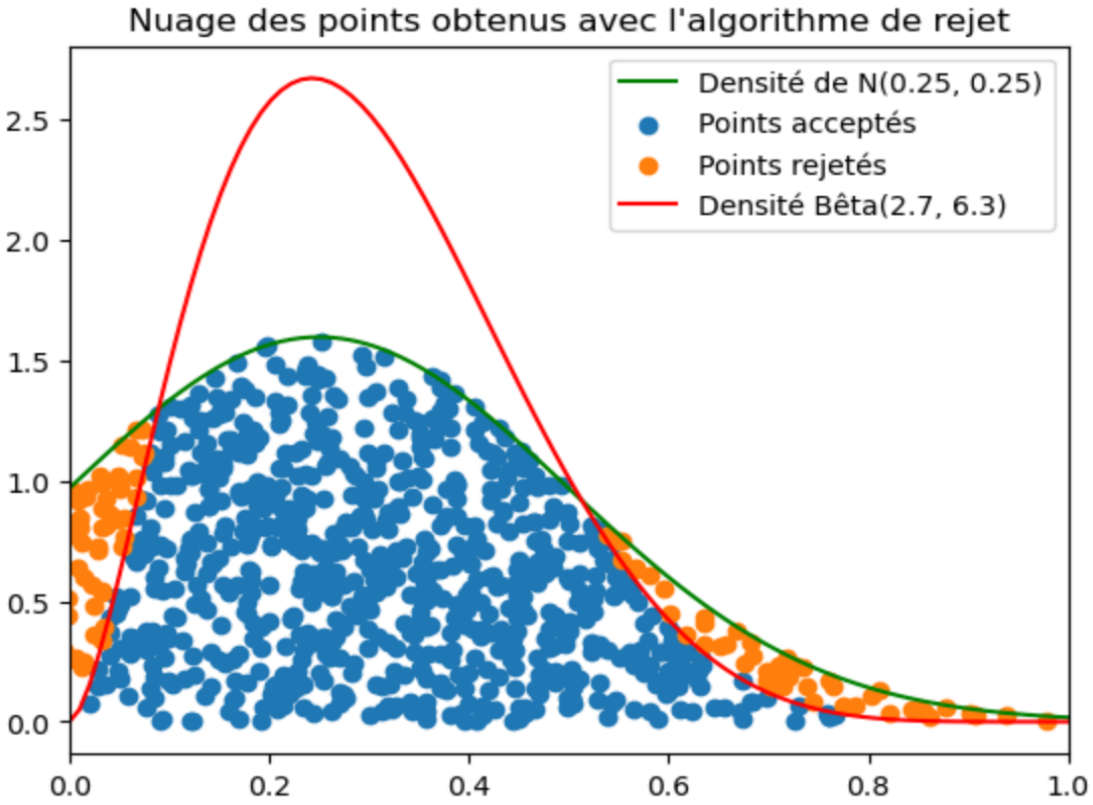
Mauvais choix de M

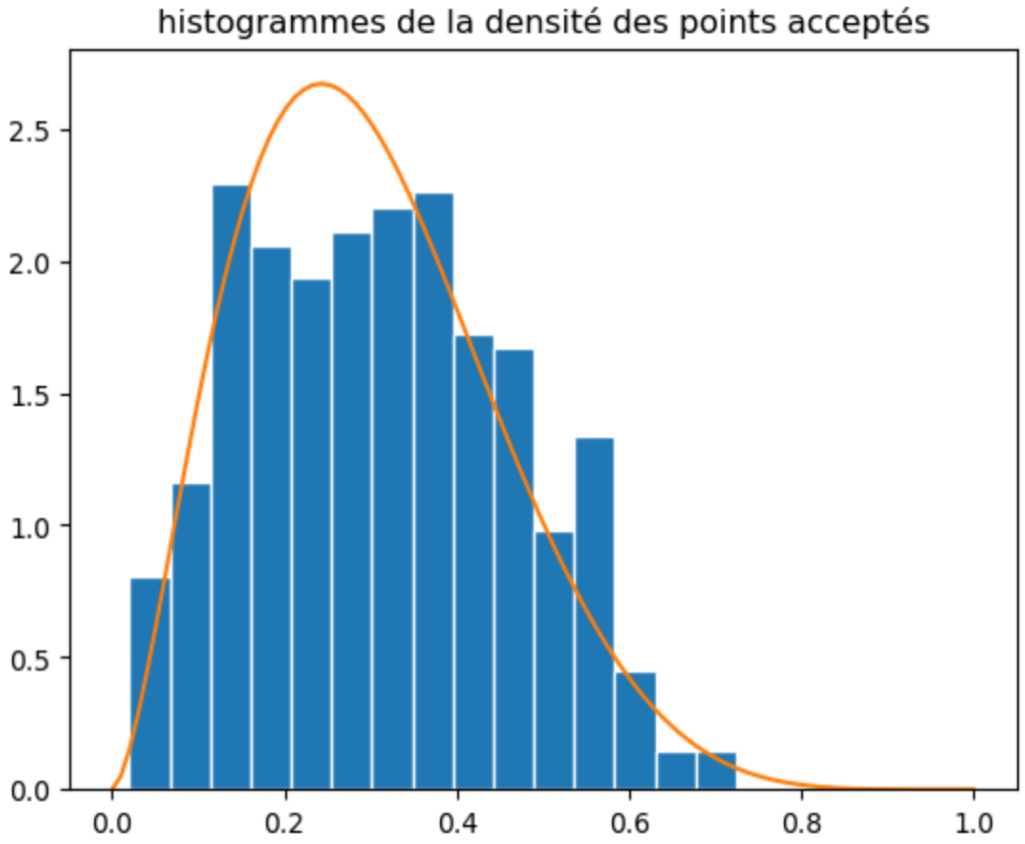
Loi Normale (0.25,0.25):



Mauvais choix de M

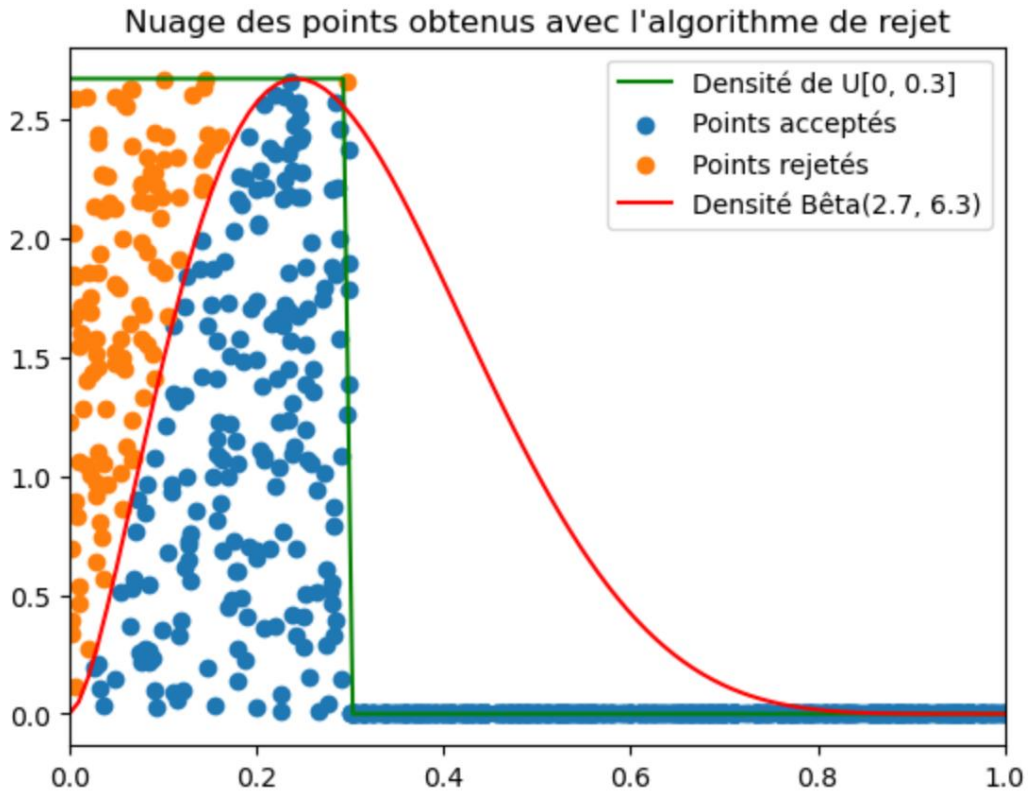
Loi Normale (0.25,0.25):



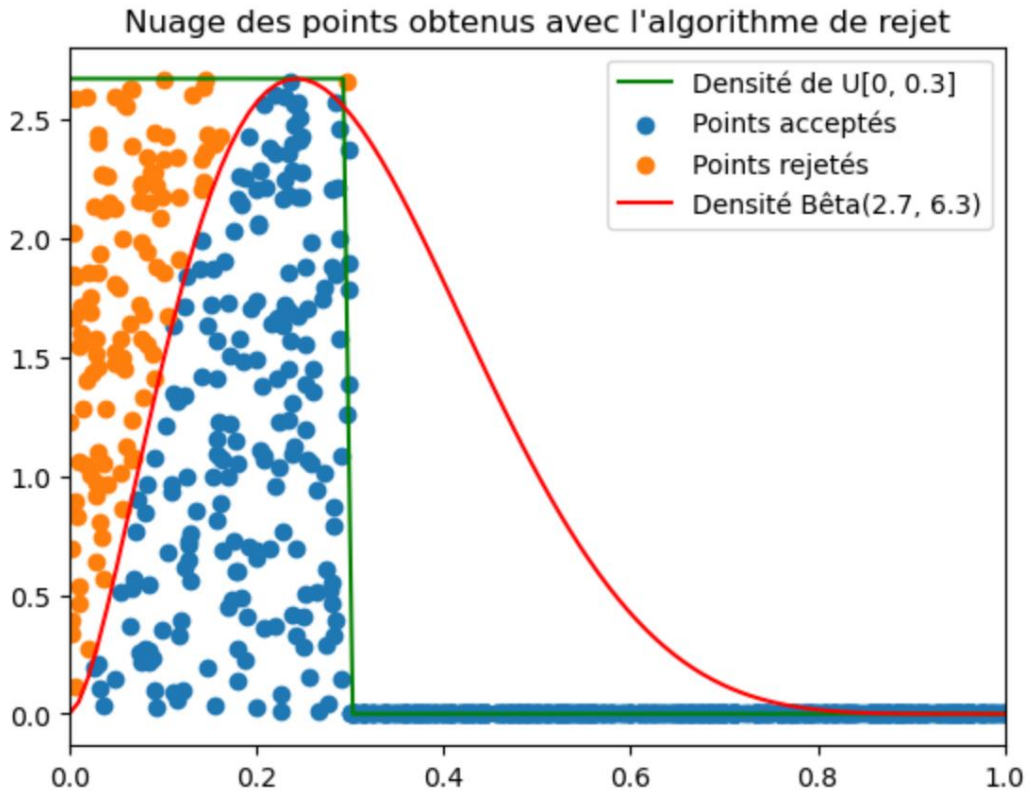


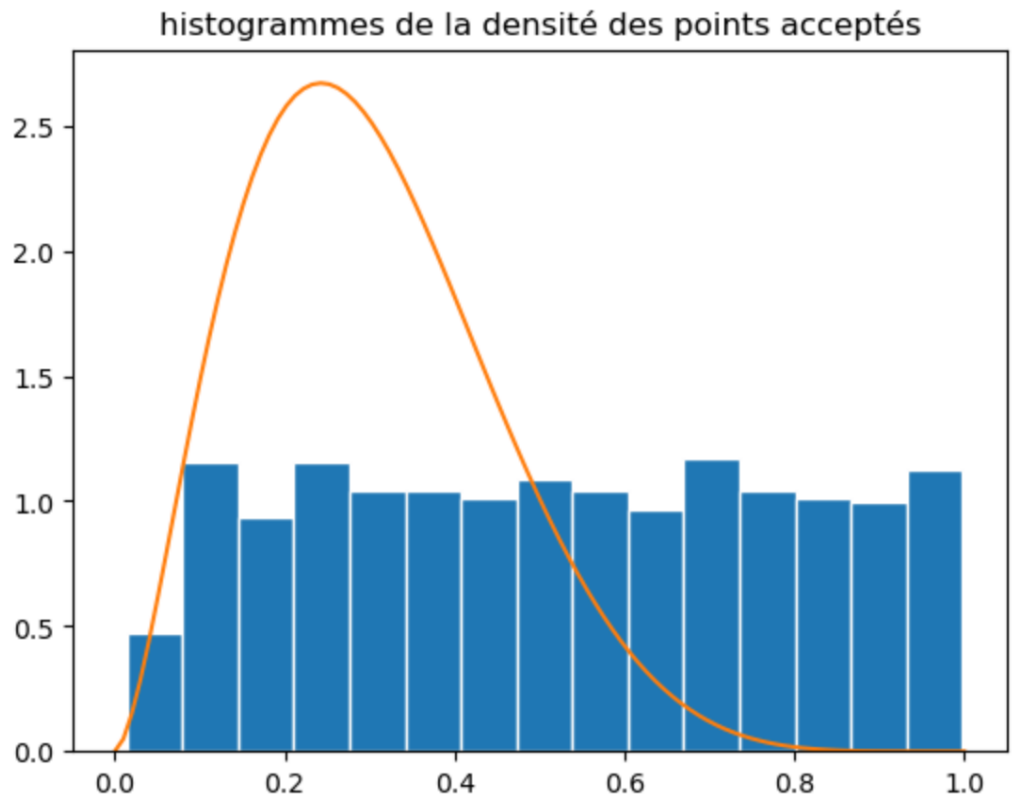
Mauvais choix de g

Mauvais choix de g



Mauvais choix de g





Limitations

- Pertinence de g et M

Limitations

- Pertinence de g et M

Critères pour un bon choix de g et M:

- supp(g) = supp(f)
- M par tâtonnement ou minimisation

Limitations

- Pertinence de g et M

<u>Critères pour un bon choix de g et M</u>:

- supp(g) = supp(f)

- M par tâtonnement ou minimisation

- Temps d'exécution
  - U[0,1]: `23.3 ms ± 2.76 ms per loop`
  - N(0.25, 0.25): `37.9 ms ± 3.59 ms per loop`

Limitations

- Pertinence de g et M

Critères pour un bon choix de g et M:

- supp(g) = supp(f)
- M par tâtonnement ou minimisation

- Temps d'exécution
  - U[0,1]: `23.3 ms ± 2.76 ms per loop`
  - N(0.25, 0.25): `37.9 ms ± 3.59 ms per loop`

Dilemme: choisir entre rapidité et haut taux d'acceptation

Sources

- Simulation des variables aléatoires, Mohamed El Merouani

- Méthodes de Monte Carlo avec R, Christian P Robert

- La méthode de simulation d'acceptation-rejet