

# Rapport de Projet : Modèles de Diffusion

Implémentation d'un processus de génération par diffusion

Lesech Erwann, Le Riboter Aymeric  
EPITA SCIA-G

10 janvier 2026

## Introduction

Ce projet s'inscrit dans le cadre du cours de processus stochastiques et vise à implémenter un modèle génératif basé sur un processus de diffusion. Notre objectif était de comprendre et de mettre en pratique les mécanismes théoriques des modèles de diffusion probabilistes, en partant d'un cas simple pour progressivement augmenter la complexité des données traitées. Nous avons débuté par une preuve de concept sur des données bidimensionnelles, puis étendu notre approche à la génération d'images en niveaux de gris avec le dataset MNIST, avant de finaliser avec des images RGB du dataset CIFAR-10. Ce rapport présente notre démarche méthodologique, les obstacles rencontrés durant le développement et les résultats obtenus à chaque étape.

## 1 Fondements théoriques et approche méthodologique

Les modèles de diffusion reposent sur l'idée d'apprendre à inverser un processus de bruitage progressif appliqué aux données. Le processus direct, appelé *noising process* ou processus de bruitage, transforme graduellement une donnée  $x_0$  issue de la distribution des données en un bruit gaussien pur  $x_T \sim \mathcal{N}(0, I)$ . Cette transformation suit une dynamique de Langevin discrète où chaque étape ajoute un bruit gaussien contrôlé par un paramètre  $\sigma_t$ . Formellement, nous avons :  $x_t = \sqrt{\sigma_t}x_{t-1} + \sqrt{1 - \sigma_t}\xi$  où  $\xi \sim \mathcal{N}(0, I)$ . En développant récursivement cette relation, on obtient une expression directe de  $x_t$  en fonction de  $x_0$  :  $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\xi_0$  avec  $\bar{\alpha}_t = \prod_{i=1}^t \sigma_i$  le produit cumulé des coefficients.

Le défi principal consiste à inverser ce processus pour générer de nouvelles données en partant d'un bruit aléatoire. Contrairement au processus de bruitage qui est direct et ne nécessite aucun apprentissage, le processus inverse requiert d'estimer la distribution conditionnelle  $p(x_{t-1}|x_t)$  qui n'est pas connue analytiquement. Nous avons alors décidé d'entraîner un réseau de neurones  $\epsilon_\theta(x_t, t)$  pour prédire le bruit  $\xi_0$  ayant été ajouté lors du processus de bruitage. Cette reformulation transforme le problème de débruitage en un problème de régression supervisée où la fonction de perte est simplement l'erreur quadratique moyenne entre le bruit réel et le bruit prédit :  $L = \|\xi_0 - \epsilon_\theta(x_t, t)\|^2$ . L'avantage de cette approche est que prédire le bruit est plus stable numériquement que de prédire directement la donnée débruitée, car le bruit suit une distribution gaussienne standard.

## 2 Démarche progressive : du simple au complexe

Nous avons structuré notre projet en trois phases distinctes, chacune apportant son lot de défis et d'enseignements. Cette approche incrémentale nous a permis de valider les concepts théoriques avant de les appliquer à des données plus complexes.

### 2.1 Phase 1 : Preuve de concept avec la diffusion 2D

Notre point de départ a été le projet ToyDiffusion<sup>1</sup>, qui implémente un modèle de diffusion pour générer des distributions de points en deux dimensions. Nous avons utilisé ce code comme base pour comprendre le fonctionnement concret des algorithmes d'entraînement et de génération.

Nous avons adapté le code initial en implémentant rigoureusement l'algorithme d'entraînement décrit dans le cours. L'algorithme consiste à sélectionner aléatoirement un échantillon  $x_0$  de la distribution des données, tirer uniformément un pas de temps  $t \in \{1, \dots, T\}$ , générer un bruit gaussien  $\xi_0$ , calculer  $x_t$  selon la formule du forward process, puis effectuer une descente de gradient sur la fonction de perte. Pour cette phase, nous avons utilisé un simple perceptron multicouches (MLP) comme architecture de réseau, avec des embeddings sinusoidaux pour encoder le pas de temps  $t$ . Ces embeddings permettent au réseau de distinguer les différents niveaux de bruit et d'adapter sa prédiction en conséquence.

### 2.2 Phase 2 : Extension aux images avec MNIST

Fort de notre compréhension acquise sur les données 2D, nous avons entrepris la génération d'images en utilisant le dataset MNIST de chiffres manuscrits. Cette transition a nécessité plusieurs adaptations importantes. Premièrement, l'architecture du réseau : nous sommes passés d'un MLP à un U-Net, une architecture convolutionnelle spécialement conçue pour les tâches de génération d'images.

Nous avons débuté avec des images de faible résolution ( $16 \times 16$  pixels) pour limiter les coûts de calcul et accélérer les itérations de développement. Nous nous sommes d'ailleurs limités initialement à la génération d'une seule classe de chiffres (le chiffre 6) pour simplifier le problème. Cette approche uni-classe nous a permis de valider notre implémentation et d'ajuster les hyperparamètres avant de généraliser à toutes les classes.

Notre première approche utilisait un scheduler linéaire avec  $\sigma_t$  variant de 0.0001 à 0.2 sur 50 étapes. Les résultats montraient une tendance à générer des images trop floues, suggérant que le modèle avait du mal à capturer les détails fins. Après analyse, nous avons réalisé que ce scheduler était trop agressif : le bruit s'accumulait trop rapidement, rendant l'inversion difficile. Nous avons donc expérimenté avec des schedulers plus doux en augmentant le nombre d'étapes à 200. Cette modification a significativement amélioré la netteté des images générées.

---

<sup>1</sup><https://github.com/ThiagoLira/ToyDiffusion/tree/main>

### 2.3 Phase 3 : Généralisation avec CIFAR-10

La dernière phase du projet consistait à appliquer notre approche à des images RGB de résolution  $32 \times 32$  issues du dataset CIFAR-10 (dataset <https://www.cs.toronto.edu/~kriz/cifar.html> représentant 10 classes d'images incluant avions, voitures, animaux, etc.). Cette transition vers des images couleur a triplé la dimensionnalité des données (passage de 1 canal à 3 canaux), augmentant considérablement la complexité du problème d'apprentissage. De plus, CIFAR-10 contient des images bien plus variées que MNIST (avions, voitures, animaux, etc.), ce qui représente un défi supplémentaire pour le modèle.

Nous avons adapté notre architecture U-Net pour traiter des entrées à 3 canaux. L'entraînement s'est révélé plus long et plus instable que pour MNIST. Nous avons observé que le modèle avait tendance à générer des images très floues et manquant de cohérence structurelle. Cette difficulté s'explique par deux facteurs : d'une part, la diversité des classes dans CIFAR-10 avec 3 canaux de couleurs rend la distribution des données plus ardue ; d'autre part, la résolution accrue amplifie les erreurs de prédiction du bruit.

Pour améliorer les résultats, nous avons suivi le modèle décrit dans le papier de recherche <sup>2</sup> en intégrant des mécanismes d'attention dans notre U-Net. L'attention permet au modèle de se concentrer sur des régions spécifiques de l'image lors de la prédiction du bruit, améliorant ainsi la qualité des détails générés.

## 3 Obstacles rencontrés et solutions apportées

Au cours de ce projet, nous avons rencontré plusieurs difficultés techniques. La première difficulté majeure concernait l'implémentation correcte du bruitage. Nous avons initialement commis l'erreur de calculer  $x_t$  en appliquant récursivement la formule  $x_t = \sqrt{\sigma_t}x_{t-1} + \sqrt{1 - \sigma_t}\xi$  à partir de  $x_0$ , ce qui était inefficace. Après relecture du cours, nous avons compris qu'il fallait utiliser la formule utilisant directement  $x_0$ , ce qui a considérablement simplifié le code et amélioré les performances.

Une autre difficulté concernait le choix des hyperparamètres, en particulier le scheduler de bruit. Nous avons constaté empiriquement qu'un scheduler trop agressif (avec des valeurs de  $\sigma_t$  décroissant rapidement) rendait le processus inverse très difficile à apprendre, tandis qu'un scheduler trop conservateur nécessitait un nombre d'étapes prohibitif. Le compromis optimal dépend fortement de la complexité des données : MNIST tolère des schedulers plus agressifs étant des images en noir et blanc d'une dimensionnalité plus faible que CIFAR-10.

Enfin, sur le plan computationnel, l'entraînement des modèles, particulièrement pour CIFAR-10, s'est révélé très coûteux en temps de calcul. Sans accès à des GPUs performants, chaque expérience nécessitait plusieurs heures d'entraînement, ce qui a limité le nombre de configurations que nous avons pu tester. Nous avons pallié cette limitation en effectuant d'abord des tests sur de petits sous-ensembles de données. De plus, nous avons pris soin de sauvegarder les poids de nos modèles afin de pouvoir faire de l'inférence sans avoir à réentraîner à chaque fois.

---

<sup>2</sup><https://arxiv.org/pdf/2006.11239.pdf>

## 4 Résultats et analyse

Les résultats obtenus valident la faisabilité de notre approche sur différents types de données, tout en révélant les limites inhérentes aux contraintes computationnelles et à la complexité des données.

### 4.1 Évaluation quantitative : la métrique FID

Pour évaluer objectivement la qualité des images générées, nous avons utilisé la métrique Fréchet Inception Distance (FID). Cette métrique mesure la distance entre la distribution des images générées et celle des images réelles. Concrètement, le FID calcule la distance de Fréchet entre deux distributions gaussiennes. Mathématiquement, le FID est défini comme suit :

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr} \left( \Sigma_r + \Sigma_g - 2\sqrt{\Sigma_r \Sigma_g} \right)$$

où  $\mu_r$  et  $\Sigma_r$  sont la moyenne et la matrice de covariance des features des images réelles, et  $\mu_g$  et  $\Sigma_g$  celles des images générées. Un FID plus faible indique une meilleure qualité de génération, avec des valeurs proches de zéro signifiant que les distributions réelles et générées sont quasiment identiques.

### 4.2 Résultats sur les données 2D

Pour la diffusion 2D, le modèle a réussi à reproduire une distribution cible similaire. On arrive à retrouver le chiffre 6.

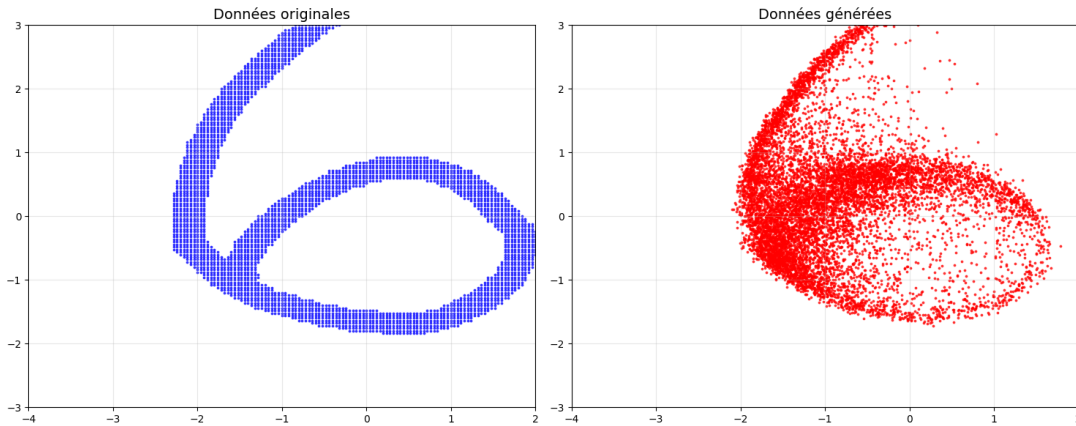


Figure 1: Visualisation du 6 original et du 6 généré par le modèle de diffusion 2D. Le modèle capture bien la forme générale et la variabilité des données.

### 4.3 Résultats sur MNIST

Sur MNIST en  $16 \times 16$ , nous avons adopté une approche progressive en deux étapes. Dans un premier temps, nous avons développé un modèle uni-classe focalisé uniquement sur la génération du chiffre 6. Cette approche ciblée nous a permis de valider notre implémentation et d’optimiser les hyperparamètres sur un problème de complexité réduite. Le modèle uni-classe a rapidement

convergé et générait des chiffres 6 de qualité satisfaisante après seulement 100 époques d'entraînement. Cette expérience nous a permis d'affiner notre compréhension du scheduler de bruit et du nombre d'étapes de débruitage optimal.

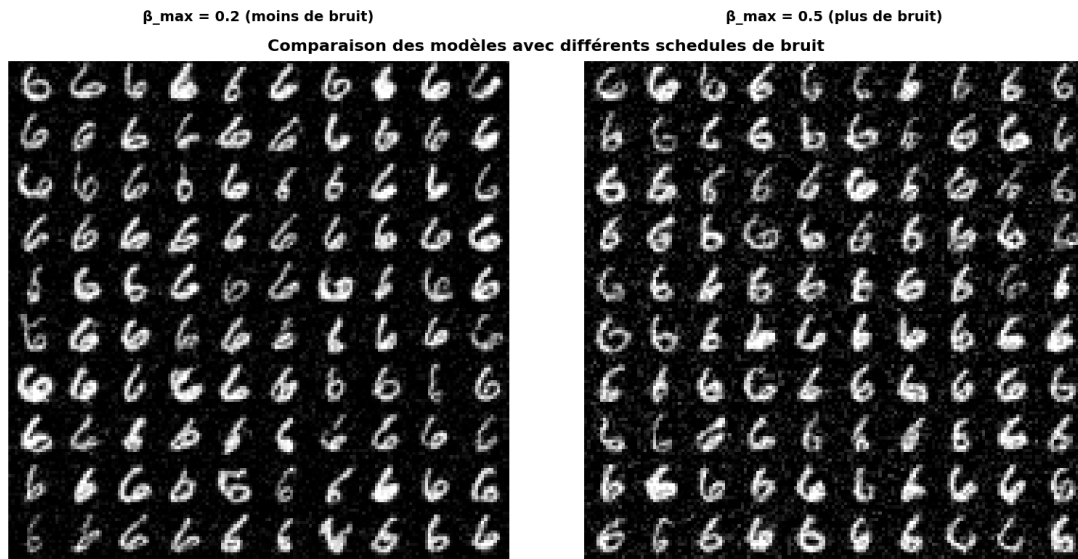


Figure 2: Échantillons générés par le modèle uni-classe (chiffre 6) avec deux schedules de bruit différents.

On peut également observer les résultats intermédiaires de reconstruction à différentes étapes de débruitage ci-dessous :

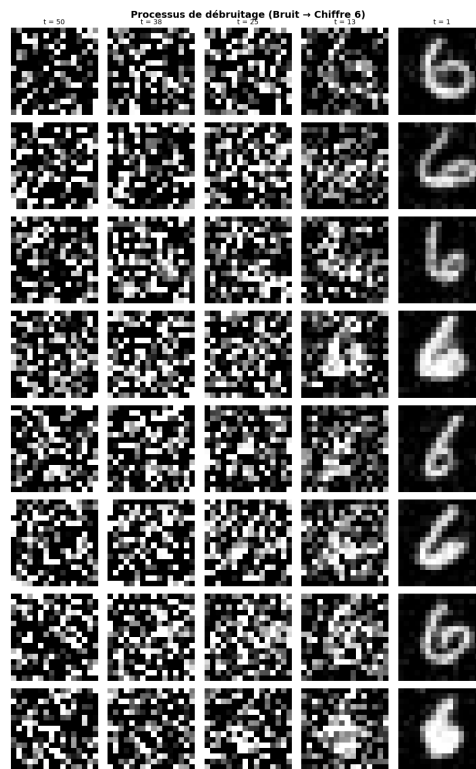


Figure 3: Processus de débruitage du modèle uni-classe.

Fort de cette réussite, nous avons ensuite étendu notre approche à un modèle multi-classe capable de générer n'importe quel chiffre de 0 à 9.



Figure 4: Échantillons générés par le modèle multi-classe pour chaque chiffre de 0 à 9.

## 4.4 Résultats sur CIFAR-10

Les résultats sur CIFAR-10 sont plus mitigés. Bien que le modèle génère des images qui possèdent des structures de couleurs cohérentes et suggèrent parfois la présence d'objets, la netteté reste limitée et les détails fins sont absents. Cette différence de performance par rapport à MNIST s'explique par plusieurs facteurs : la dimensionnalité accrue (3 canaux versus 1 et  $32 \times 32$  au lieu de  $16 \times 16$ ), la plus grande variabilité inter-classes de CIFAR-10. Notre modèle de diffusion pour CIFAR-10 utilise une U-Net avec un mécanisme d'attention qui permet de capturer plus d'informations. Néanmoins, il faudrait entraîner des modèles plus profonds pour de meilleurs résultats, ce qui dépasse largement nos ressources disponibles.

Tout au long de nos expérimentations, nous avons pu observer une amélioration drastique du FID passant de 19 à 1.8 en ajoutant l'attention et ajustant les schedulers de bruit.

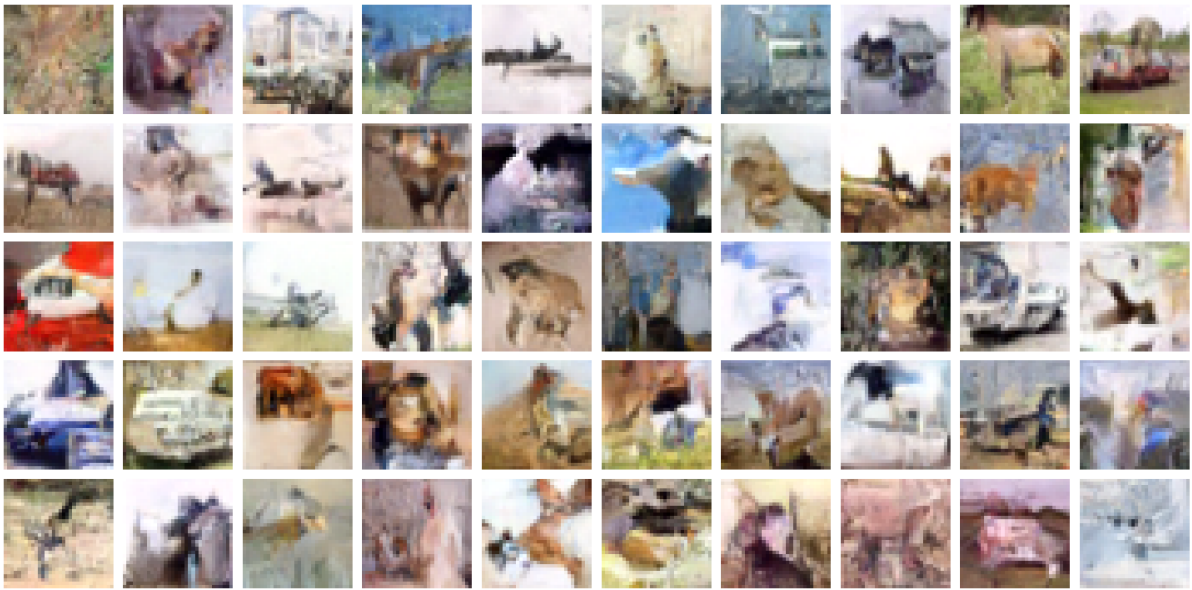


Figure 5: Échantillons générés par le meilleur modèle à partir de CIFAR-10.

## Conclusion

Ce projet nous a permis de comprendre en profondeur le fonctionnement des modèles de diffusion, depuis les fondements mathématiques jusqu'aux détails pratiques de l'implémentation. Notre démarche progressive, partant de données 2D simples pour aboutir à des images RGB complexes, nous a confrontés aux défis réels de l'apprentissage génératif : choix des hyperparamètres, stabilité numérique, coût computationnel et compromis qualité-efficacité. Les résultats obtenus, bien qu'imparfaits pour CIFAR-10, démontrent que nous avons acquis une bonne compréhension des mécanismes de diffusion et les capacités nécessaires pour implémenter ces concepts.