Géométrie des Données et Apprentissage Machine

Module 5 – Introduction à l'IA générative

Youssef MESRI - MINES Paris - PSL

November 23, 2025

Plan

Introduction aux modèles génératifs (VAE, GAN, Flows) : plan

- Introduction aux modèles génératifs (VAE, GAN, Flows)
- Définition d'un modèle génératif
- VAE : principe, encodage/décodage, régularisation KL
- VAE : formule de l'ELBO et explication mathématique
- Exemple concret VAE : génération de chiffres MNIST
- GAN : principe, adversarial loss
- GAN : architecture générateur/discriminateur
- GAN : problèmes de convergence et techniques d'amélioration
- Flows : définition et motivation (transformation bijective)
- Flows : formule du changement de variables
- Flows : exemple simple (RealNVP, Glow)
- Comparaison VAE / GAN / Flows
- Applications pratiques : images, texte, molécules
- Évaluation : log-likelihood, FID, inception score
- Limites et challenges des modèles classiques
- Exos

Motivation : Pourquoi les modèles génératifs ?

- Générer de nouvelles données réalistes : images, texte, audio
- Comprendre la distribution sous-jacente des données
- Applications :
  - Synthèse d'images (DeepFake, art AI)
  - Génération de molécules ou structures 3D
  - Data augmentation pour apprentissage supervisé

Définition d'un modèle génératif

- Objectif : approximer la distribution de données $p_{\text{data}}(x)$
- Modèle paramétrique $p_\theta(x)$ ou transformée déterministe $x = f_\theta(z)$
- Deux grandes approches :
  - **Explicit likelihood** : VAE, Flows
  - **Implicit likelihood** : GAN

Variational AutoEncoder (VAE) : motivation

- Objectif : apprendre une représentation latente z continue de données x
- Permet :
  - Génération de nouvelles données réalistes
  - Réduction de dimensionnalité probabiliste
  - Exploration et manipulation de l'espace latent
- Contraintes : différentiabilité, espace latent régulier

Pourquoi "Variational" ?

- Approche bayésienne : approximer la distribution postérieure $p_\theta(z|x)$
- Directement $p_\theta(z|x)$ est intractable
- Solution : utiliser une distribution approchée $q_\phi(z|x)$
- Optimisation via **Variational Inference** :

$$\text{ELBO} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \text{KL}(q_\phi(z|x)||p(z))$$

- "Variational" = on optimise une borne inférieure sur la log-vraisemblance

Encoder : $q_\phi(z|x)$

- Mappe x vers un vecteur latent z
- Approximé par une distribution Gaussienne :

$$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \text{diag}(\sigma^2_\phi(x)))$$

- **Reparameterization trick** pour différentiabilité :

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

- Avantage : on peut backpropager à travers z pour entraîner ϕ

Decoder : $p_\theta(x|z)$

- Reconstruit x à partir du vecteur latent z
- Paramétré par un réseau neuronal (feedforward ou convolutionnel)
- Modélise la probabilité conditionnelle $p_\theta(x|z)$
- Génère $\hat{x}$ ressemblant à x en sortie
- Schéma conceptuel :

$$z \rightarrow \text{Decoder} \rightarrow \hat{x}$$

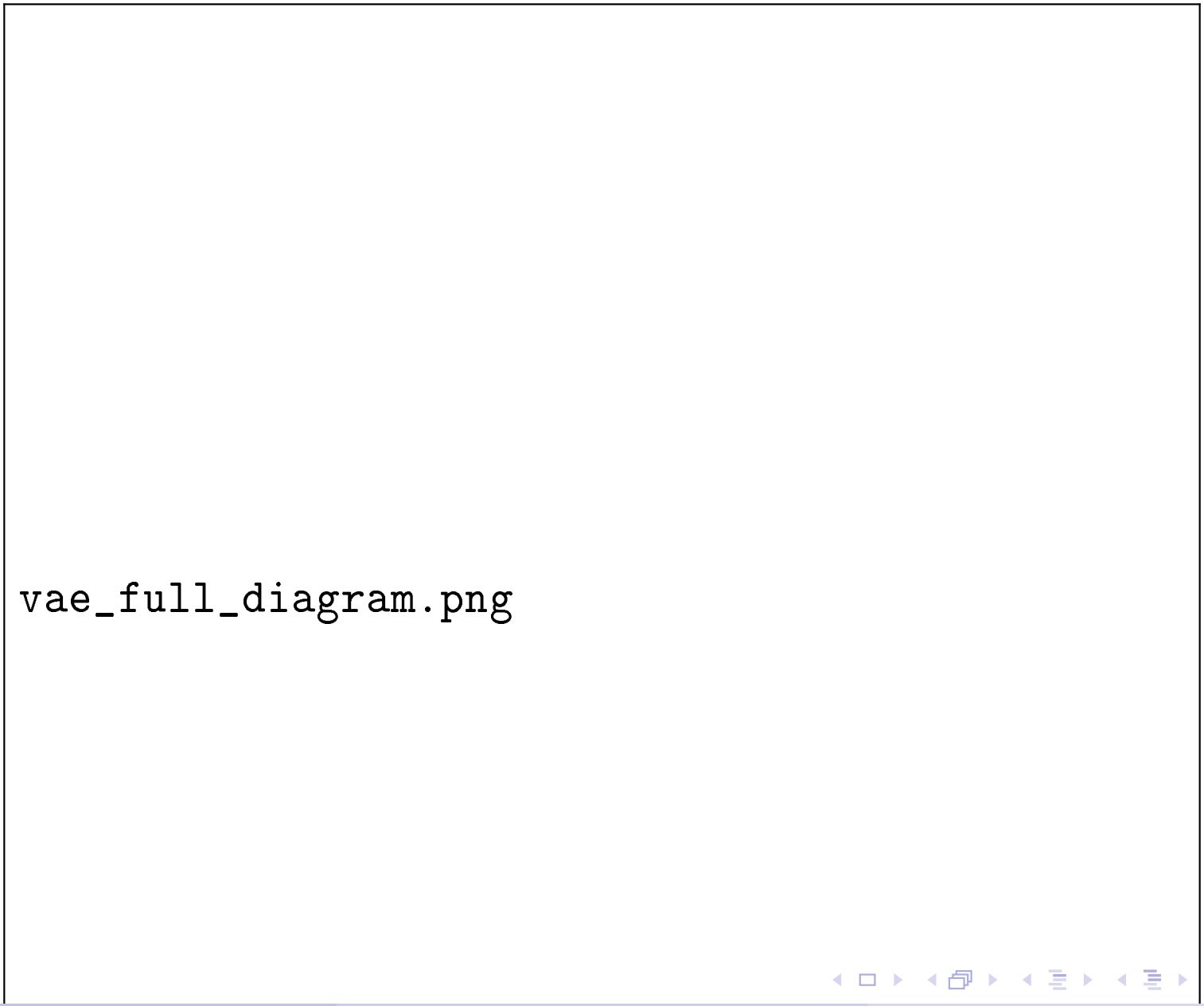
Loss function : ELBO

- Evidence Lower Bound (ELBO) :

$$\mathcal{L}(\theta, \phi; x) = \underbrace{\mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}(q_\phi(z|x)||p(z))}_{\text{régularisation}}$$

- Reconstruction : rapproche $\hat{x}$ de x
- KL : force $q_\phi(z|x)$ proche de prior $p(z)$, régularise espace latent
- Optimisation via gradient descent sur θ et ϕ

VAE : schéma global



Visualisation espace latent

- Exemple MNIST : encoder 784D $\rightarrow$ 2D latent
- Nuage de points : chaque chiffre coloré différemment
- Interpolation linéaire dans latent $\rightarrow$ génération continue de chiffres

vae_latent_mnist.png

Synthèse VAE

- Variational AutoEncoder = combinaison de :
  - Encoder probabiliste
  - Decoder génératif
  - Loss ELBO (reconstruction + KL)
- "Variational" : approximation de la postérieure via optimisation variationnelle
- Applications : génération d'images, réduction de dimensionnalité, interpolation dans l'espace latent

Exemple concret : VAE MNIST

- Encoder : $784 \to 64 \to$ 2D latent
- Decoder : 2D latent $\to 64 \to 784$
- Visualisation : nuage de points latent $\to$ génération de chiffres
- Avantage : structure continue de l'espace latent

Generative Adversarial Networks (GAN) : motivation

- Objectif : générer des échantillons réalistes à partir de bruit latent
- Utilisé pour :
  - Images : photoréalistes ou artistiques
  - Audio, musique, vidéo
  - Données synthétiques pour apprentissage supervisé
- Problème classique : réseaux génératifs simples $\rightarrow$ blurry, distribution approximative

Architecture GAN

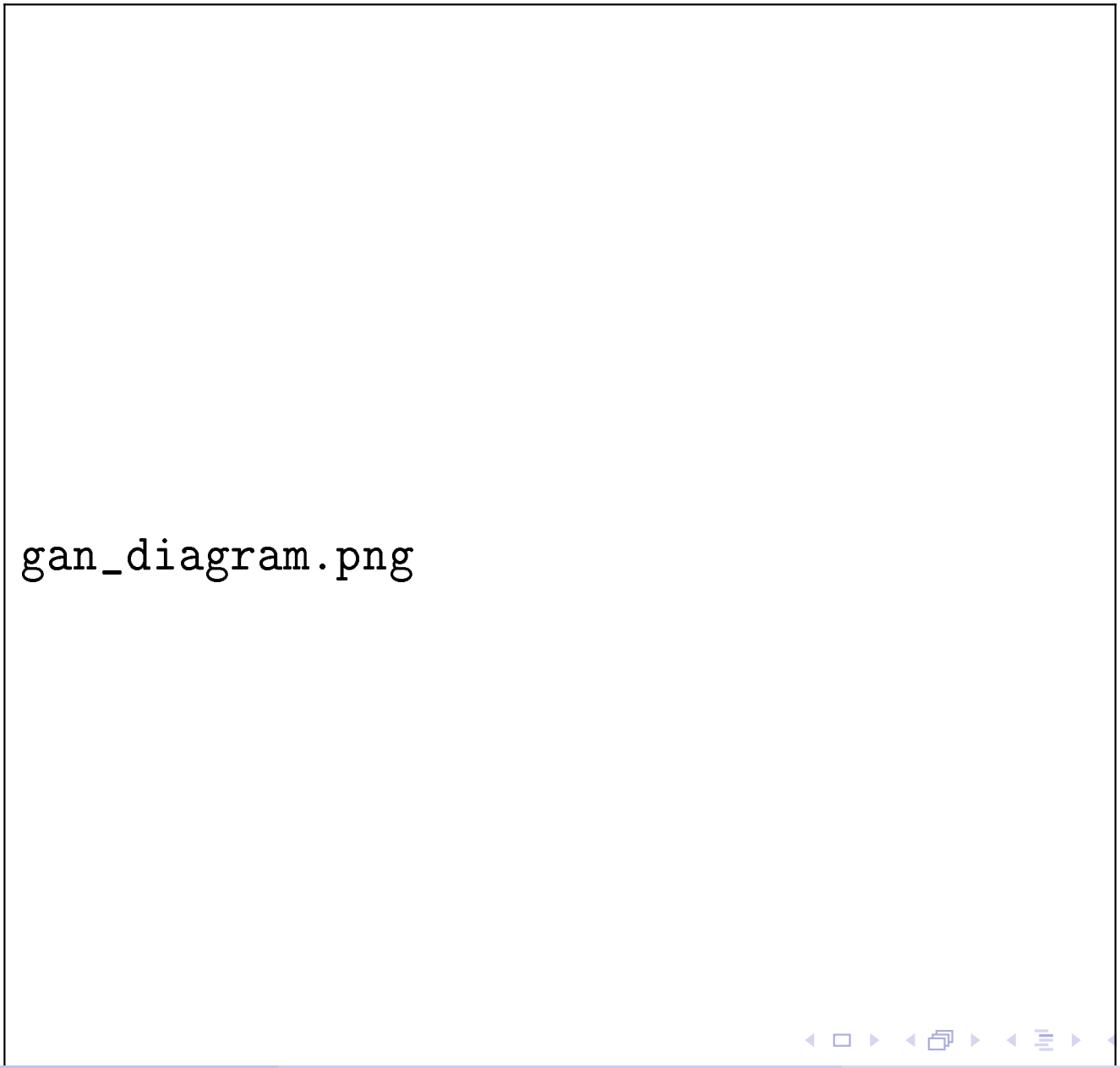
- Deux réseaux en compétition :
  - **Generator** $G(z)$: prend $z \sim p_z$ (bruit) et génère $\hat{x}$
  - **Discriminator** $D(x)$: différencie $x \sim p_{\text{data}}$ et $\hat{x} \sim G(z)$
- Jeu à somme nulle (adversarial)

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

Fonctionnement étape par étape

1. Générateur produit un batch d'échantillons $\hat{x} = G(z)$
2. Discriminateur évalue probabilité de vrai/faux
3. Backpropagation sur D pour distinguer vrai/faux
4. Backpropagation sur G pour tromper D (maximize $\log D(G(z))$)
5. Répéter jusqu'à convergence

Schéma conceptuel GAN



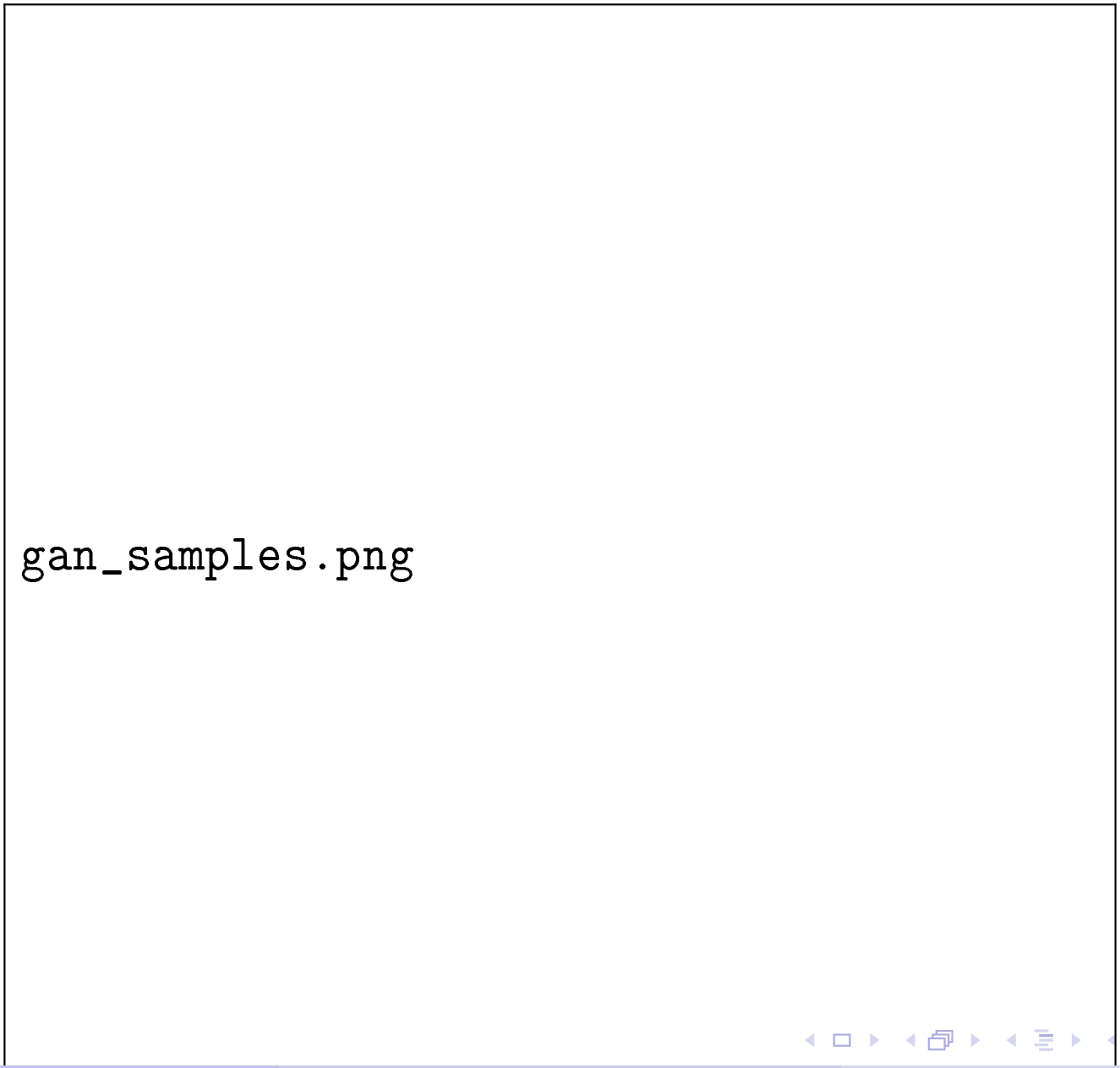
Points importants et difficultés

- Mode collapse : G génère un petit nombre de modes
- Instabilité : oscillations durant l'entraînement
- Solutions :
  - Wasserstein GAN : utiliser distance Wasserstein
  - Gradient penalty, batch norm, label smoothing

Applications GAN

- Image synthesis : photoréaliste, anime, art
- Super-resolution : améliorer résolution d'images
- Data augmentation : synthèse pour apprentissage supervisé
- Audio et musique : génération de sons, voix

Exemple de génération GAN



Synthèse GAN

- Deux réseaux en compétition : Generator et Discriminator
- Jeu min-max $\rightarrow$ optimisation adversariale
- Points clés : mode collapse, instabilité, solutions modernes
- Applications variées : images, audio, données synthétiques

Generative Adversarial Networks (GAN)

- Deux réseaux en compétition :
  - **G** : générateur, produit $x = G(z)$
  - **D** : discriminateur, distingue $x_{\text{réel}}$ et x_{fake}
- Loss adversariale :

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

$$\texttt{Random noise z} \rightarrow \texttt{[G]} \rightarrow \texttt{x}_{fake} \texttt{ß[}D\texttt{]ß}Probabilit fakeour el$$

G apprend à tromper D

D apprend à détecter les faux

Entraînement alterné G/D

Challenges GAN

- Instabilité de l'entraînement
- Mode collapse : G produit peu de variations
- Techniques d'amélioration :
  - Wasserstein GAN
  - Gradient penalty
  - Label smoothing

Normalizing Flows : motivation

- Objectif : modéliser des distributions complexes via transformations bijectives simples
- Avantages :
  - Likelihood exacte
  - Génération de données et évaluation de densité
  - Différentiable et entraînable par gradient
- Applications : génération d'images, audio, densités multi-dimensionnelles

Principe des Normalizing Flows

- Transformer une distribution simple $z \sim p_Z(z)$ en distribution complexe $x \sim p_X(x)$
- Via une suite de transformations bijectives :

$$x = f_K \circ f_{K-1} \circ \cdots \circ f_1(z)$$

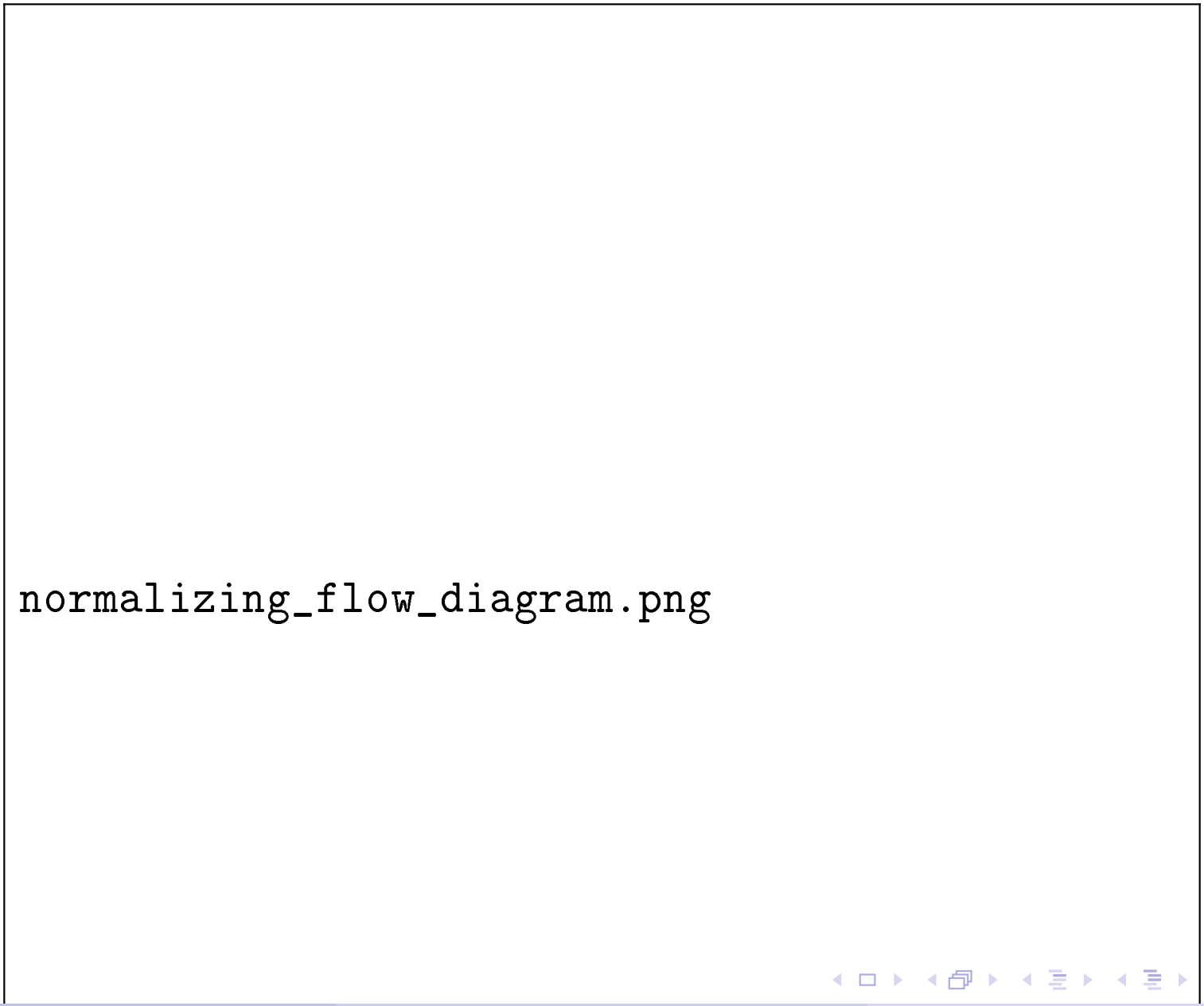
- Densité exacte via la formule du changement de variable :

$$p_X(x) = p_Z(z) \prod_{k=1}^{K} \left| \det \frac{\partial f_k}{\partial h_{k-1}} \right|^{-1}$$

Transformation bijective simple

- Chaque f_k : transformation bijective et différentiable
- Exemples :
  - Affine coupling layers (RealNVP)
  - Actnorm, 1x1 convolutions (Glow)
- Calcul du déterminant jacobien facile pour tractabilité

Schéma conceptuel des Flows



Loss function : Maximum Likelihood

- Objectif : maximiser log-likelihood sur données

$$\max_{\theta} \sum_{i=1}^{N} \log p_X(x_i) = \sum_{i=1}^{N} \left[\log p_Z(z_i) - \sum_{k=1}^{K} \log \left| \det \frac{\partial f_k}{\partial h_{k-1}} \right| \right]$$

- Optimisation directe via backpropagation
- Avantage : pas de perte adversariale, pas d'instabilité comme GAN

Génération et inversion

- Génération : $z \sim p_Z(z) \to x = f_K \circ ... \circ f_1(z)$
- Inversion : x donné $\to z = f_1^{-1} \circ ... \circ f_K^{-1}(x)$
- Permet sampling et évaluation de densité
- Schéma :

$$z \leftrightarrow x \text{ via Flow}$$

Applications des Normalizing Flows

- Génération d'images et d'audio
- Modélisation de densités multi-dimensionnelles
- Approximation de posterior en bayésien (variational inference)
- Pré-processing ou génération de features pour modèles downstream

Synthèse : Normalizing Flows

- Transformer une distribution simple en complexe via bijections différentiables
- Likelihood exacte, entraînement stable
- Complète VAE et GAN : avantages différents
- Applications variées et flexible pour IA générative

Normalizing Flows : introduction

- Transformation bijective $x = f_\theta(z)$
- Distribution exacte via changement de variables :

$$p_\theta(x) = p(z) \left| \det \frac{\partial f_\theta^{-1}}{\partial x} \right|$$

- Avantages : likelihood exacte, sampling direct

Exemple Flow : RealNVP

- Couche affine bijective : $y = s(x) \odot x + t(x)$
- Chaîne de transformations $\rightarrow$ modèle complexe
- Applications : génération d'images, densité estimation

Comparaison des modèles génératifs

Modèle	Likelihood	Latent	Sampling
VAE	explicite	continu	direct
GAN	implicite	non défini	via G
Flow	explicite	continu	direct

- VAE : stabilité mais blurry samples
- GAN : samples réalistes mais instables
- Flow : exact likelihood mais plus coûteux

Applications des modèles génératifs

- Images : MNIST, CIFAR, CelebA
- Texte : GPT, Transformer VAE
- Audio : WaveGAN, Flow-based TTS
- Chimie : génération de molécules, protéines

Évaluation des modèles génératifs

- Log-likelihood
- Frechet Inception Distance (FID)
- Inception score
- Qualité subjective et diversité

Limitations et challenges

- VAE : échantillons flous
- GAN : instabilité, mode collapse
- Flows : complexité computationnelle
- Défi global : générer des données structurées, respecter contraintes

Étude de cas : VAE sur MNIST

- Encoder 784$\rightarrow$64$\rightarrow$2D latent
- Decoder 2D latent$\rightarrow$784
- Exercice : visualiser le latent, générer de nouveaux chiffres

Exercice GAN simple

- Implémenter un GAN sur MNIST ou FashionMNIST
- Observer mode collapse et tenter WGAN ou gradient penalty

Exercice Flow

- Implémenter RealNVP simple sur un dataset 2D toy
- Visualiser la transformation bijective et densité

Mini-exemple VAE + GAN

- Comparer échantillons VAE vs GAN sur FashionMNIST
- Noter différences de qualité et diversité

Mini-exemple Flow

- Appliquer Flow sur données 2D simples
- Visualiser densité exacte et échantillons

Résumé Sous-section 1

- VAE : latent continu, ELBO, reconstruction
- GAN : générateur/adversaire, loss implicite
- Flow : transformation bijective, likelihood exacte
- Applications : images, texte, audio, chimie

Préparer transition vers diffusion models

- Limites des modèles classiques : blurry, mode collapse, complexité
- Motivation pour diffusion models et modèles fondation

Diffusion models et modèles fondation (GPT, Stable Diffusion, etc) : plan

- Introduction : motivation des diffusion models
- Processus de diffusion : forward noising process
- Processus de reverse denoising
- Loss fonctionnelle : score matching
- Architecture typique : U-Net
- Exemples d'application : images, audio, molécules
- Stable Diffusion : workflow, text-to-image
- GPT / LLM : modèle autoregressif, transformer
- Embeddings et attention mechanism
- Exemples d'usage GPT : génération texte, code
- Diffusion vs GAN vs VAE : comparaison
- Exos

Diffusion Models : motivation

- Nouveaux modèles génératifs (2020+) surpassant GAN/VAE sur qualité d'image
- Idée clé : générer en inversant un processus de diffusion bruité
- Avantages :
  - Échantillons haute qualité
  - Apprentissage stable (pas de min-max adversarial)
  - Flexibilité (conditionnement, multimodalité)

Principe des Diffusion Models

- Processus avant (forward) : bruit progressif

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right)$$

- Processus inverse (backward) : apprendre à débruiter

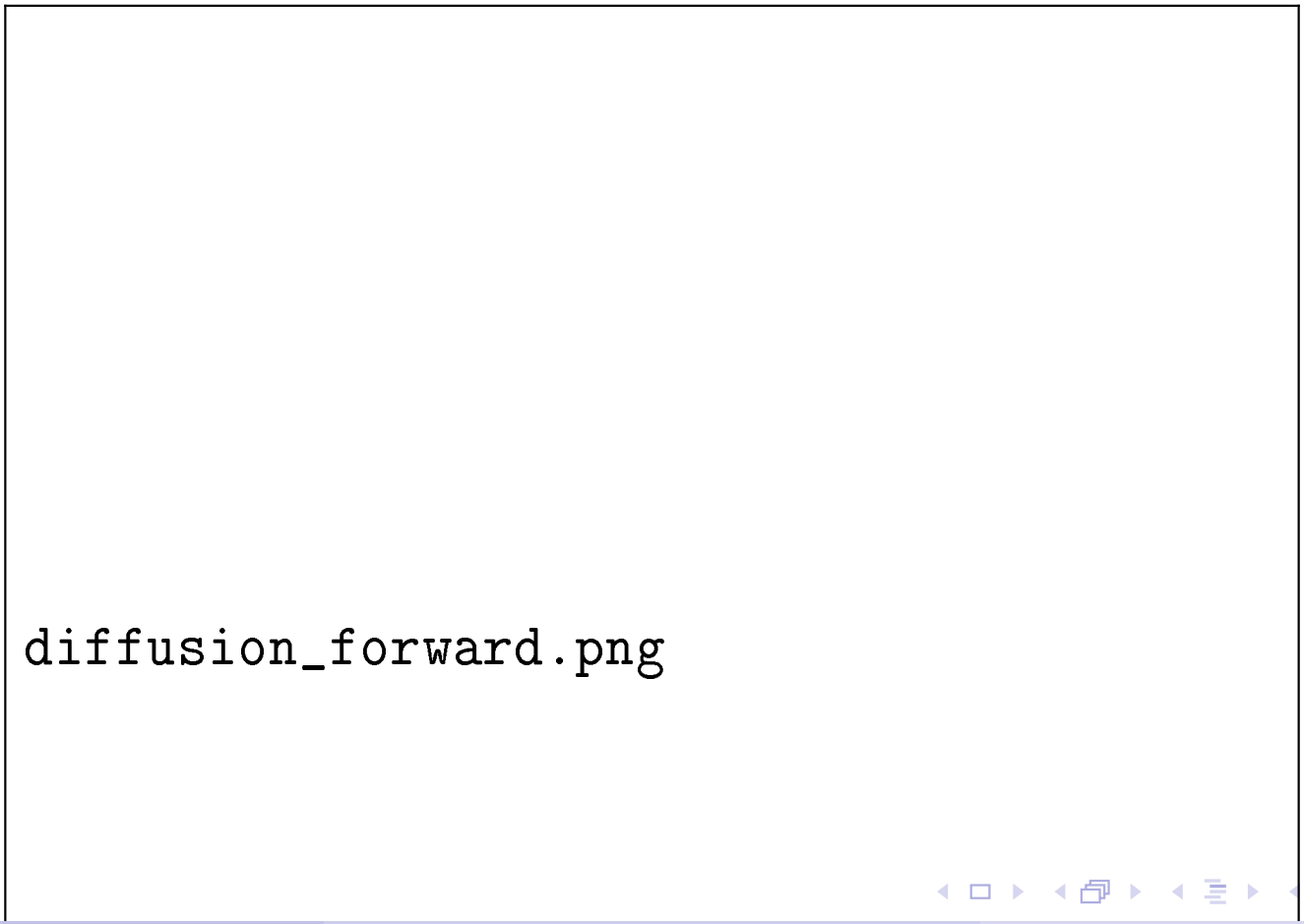
$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right)$$

- L'IA apprend à approximer la dynamique inverse

Forward process : ajout de bruit

- On part d'une donnée réelle x_0
- On ajoute progressivement du bruit gaussien sur T étapes
- Après assez d'étapes : $x_T \sim \mathcal{N}(0, I)$ (pure noise)

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)I\right)$$



Reverse process : apprentissage

- Objectif : échantillonner x_{t-1} à partir de x_t
- Approximé par un réseau neuronal $\epsilon_\theta(x_t, t)$
- Intuition : apprendre à prédire et enlever le bruit

$$p_\theta(x_{t-1}|x_t) \approx \mathcal{N}\left(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right), \beta_t I\right)$$

Fonction de perte simplifiée

- Loss d'entraînement = MSE entre bruit réel et bruit prédit

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{x_0,t,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$$

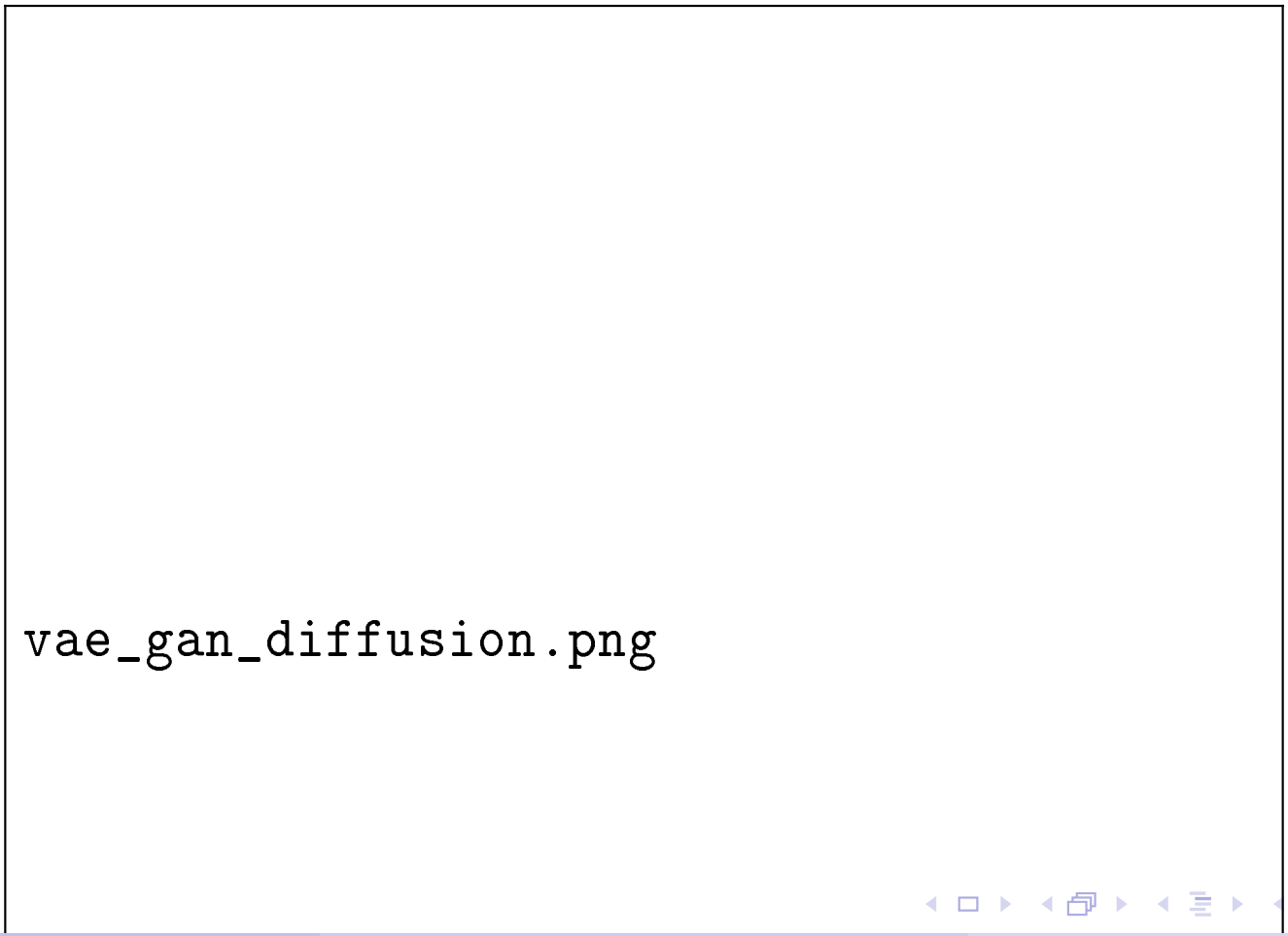
- x_t est une version bruitée de x_0
- ϵ est le bruit ajouté
- $\epsilon_\theta(x_t, t)$ = prédiction du réseau

Sampling avec Diffusion Models

- Étapes de sampling :
  1. Tirer $x_T \sim \mathcal{N}(0, I)$
  2. Appliquer séquentiellement $p_\theta(x_{t-1}|x_t)$
  3. Obtenir x_0 (image générée)
- Processus lent (1000 étapes typiques)
- Optimisation : DDIM, samplers accélérés

Comparaison : Diffusion vs GAN/VAE

- GAN : échantillons rapides, mais entraînement instable
- VAE : entraînement stable, mais qualité floue
- Diffusion : entraînement stable + haute qualité, mais sampling coûteux



Applications des Diffusion Models

- Génération d'images (Stable Diffusion, Imagen, DALL·E 3)
- Audio et musique (AudioLM, Riffusion)
- Modélisation moléculaire, biologie (AlphaFold avec diffusion)
- IA multimodale (texte $\rightarrow$ image, image $\rightarrow$ 3D, vidéo générative)

Extensions et variantes

- Score-based models (SDE + Langevin dynamics)
- Diffusion déterministe (DDIM)
- Latent diffusion models (Stable Diffusion) : diffusion dans un espace latent
- Conditionnement : texte, étiquettes, audio, structures

Synthèse : Diffusion Models

- Apprentissage d'un processus inverse au bruit gaussien
- Entraînement stable via prédiction du bruit
- Échantillons haute qualité mais sampling coûteux
- Révolution dans l'IA générative moderne

Motivation : Pourquoi les diffusion models ?

- Génération de données de haute qualité (images, audio, molécules)
- Modèles stables et contrôlables, surmontant certains problèmes des GAN
- Applications : text-to-image, audio synthesis, IA générative scientifique

Diffusion Models : processus de diffusion

- Forward process : ajout progressif de bruit à une donnée x_0

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

- Transforme distribution originale en bruit pur
- Objectif : apprendre à inverser ce processus

Reverse process : génération

- Reverse process : retirer le bruit pas à pas

$$p_\theta(x_{t-1}|x_t) \approx \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

- Réseau entraîné à prédire le bruit ou la distribution conditionnelle
- Génération finale : x_0 à partir de $x_T \sim \mathcal{N}(0, I)$

Loss fonctionnelle : score matching

- Objectif : prédire bruit ϵ ajouté

$$L(\theta) = \mathbb{E}_{x_0,\epsilon,t}\|\epsilon - \epsilon_\theta(x_t, t)\|^2$$

- Optimisation via backpropagation
- Permet un training stable comparé aux GAN

Architecture typique : U-Net

- Encodeur-décodeur avec skip connections
- Conditionnement sur timestep t
- Peut être combiné avec attention (Transformer-like)

Exemples d'application des diffusion models

- Images : MNIST, CIFAR, ImageNet
- Audio : génération de musique ou voix
- Molécules : génération de structures chimiques

Stable Diffusion

- Text-to-image diffusion model
- Conditionnement sur embeddings textuels (CLIP)
- Workflow :
  1. Encode texte $\rightarrow$ embedding
  2. Noising forward process sur image
  3. Denoising reverse process conditionné par texte

Modèles Fondations : Introduction

- Définition : modèles entraînés sur de larges corpus génériques, réutilisables pour de multiples tâches
- Exemples :
  - LLM (GPT-4, LLaMA, Mistral) pour le texte
  - Stable Diffusion, Imagen pour les images
  - AudioLM, MusicGen pour l'audio
- Idée clé : pré-entraînement massif + fine-tuning ou prompting

Caractéristiques des Modèles Fondations

- Taille massive (milliards de paramètres)
- Données hétérogènes (texte, image, code, multimodal)
- Capacité d'émergence : comportements non anticipés
- Adaptabilité via prompt engineering ou fine-tuning

LLM : GPT et Transformers

- Architecture Transformer (auto-attention)
- Pré-entraînement : prédire le token suivant (causal LM)
- Fine-tuning par RLHF (alignment humain)

$$p(x_1, \ldots, x_T) = \prod_{t=1}^{T} p(x_t | x_{<t}; \theta)$$

Applications des LLM

- Dialogue (ChatGPT, assistants IA)
- Traduction, résumé, génération de texte
- Génération de code (Copilot, Code Llama)
- Raisonnement scientifique et mathématique

Stable Diffusion : principe

- Latent Diffusion Model (LDM)
- Applique la diffusion dans un espace latent compressé
- Conditionné par du texte via CLIP embeddings

$$z_0 \xrightarrow{\text{diffusion inverse}} z_T \quad \rightarrow \quad \text{decode}(z_T) \approx \text{image générée}$$

Applications IA générative

- Texte $\rightarrow$ image (Stable Diffusion, DALL·E)
- Image $\rightarrow$ image (inpainting, style transfer)
- Texte $\rightarrow$ vidéo (Sora, Pika)
- Multimodalité (texte $\leftrightarrow$ audio, 3D, simulation)

Lien Diffusion $\leftrightarrow$ Modèles Fondations

- Les LDM (Stable Diffusion) combinent :
  - Encodeur VAE pour espace latent
  - Diffusion pour génération
  - Conditionnement par un LLM/CLIP
- Exemples : texte $\rightarrow$ image réaliste en quelques secondes

Défis des Modèles Fondations

- Coût d'entraînement (centaines de GPU, mois de calcul)
- Biais et toxicité hérités des données
- Contrôle et interprétabilité
- Durabilité énergétique

Impact sociétal

- Révolution de la productivité (IA copilote)
- Transformation de la recherche et de l'éducation
- Questions éthiques et juridiques (plagiat, droits d'auteur)
- Vers des IA généralistes (AGI ?)

Synthèse : Modèles Fondations

- Entraînement massif sur données génériques
- Réutilisables pour de multiples tâches
- Exemples : GPT (texte), Stable Diffusion (images)
- Défis : coût, biais, contrôle, énergie

Modèles fondation : GPT

- Transformer autoregressif
- Pré-entraînement sur large corpus : langage, code, texte-image
- Génération séquentielle : $x_t \sim p_\theta(x_t|x_{<t})$
- Applications : texte, code, multimodal (image+texte)

Transformer : attention mechanism

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- Q = queries, K = keys, V = values
- Permet de capturer dépendances longues dans la séquence
- Base de GPT, Stable Diffusion (cross attention)

Diffusion vs GAN vs VAE

Modèle	Stabilité	Qualité	Complexité
VAE	élevée	moyenne	faible
GAN	faible	élevée	moyenne
Diffusion	élevée	élevée	élevée

- Diffusion models : stable, génératif haute qualité
- GAN : rapide, mais instable
- VAE : stable, latent interpretable, mais blurry

Exemple pratique : Diffusion sur MNIST

- Forward noising : ajouter bruit progressif aux images
- Reverse denoising : prédire le bruit à chaque timestep
- Visualiser évolution image : de bruit $\rightarrow$ chiffre clair

Exercice : implémentation U-Net

- Construire un U-Net simple
- Conditionner sur timestep t
- Entraîner sur MNIST ou FashionMNIST

Exercice : Stable Diffusion mini-projet

- Text-to-image simple
- Préparer embeddings texte
- Observer influence du prompt sur génération

Exercice : visualiser score-based diffusion

- Implémenter forward/noise process sur 2D toy dataset
- Visualiser les trajectoires de reverse process

Exemple multimodal

- Texte $\rightarrow$ image (Stable Diffusion)
- Image $\rightarrow$ image (inpainting)
- Comparer qualité génération vs GAN classique

Exercice : comparaison VAE / GAN / Diffusion

- Générer échantillons pour chaque modèle
- Comparer qualité, diversité, stabilité

Mini-cas : génération molécules

- Forward process : ajouter bruit aux positions atomiques
- Reverse process : prédire positions correctes
- Objectif : molécules valides

Résumé sous-section 2

- Diffusion models : forward noise + reverse denoising
- Stable Diffusion : text-to-image, embeddings textuels
- GPT : autoregressive transformer
- Diffusion > GAN/VAE en stabilité et qualité
- Exercices : implémentation U-Net, visualisation forward/reverse

Liens avec l'optimisation (score-based models, transport optimal) : plan

- Score-based generative modeling : définition et intuition
- Relation avec gradient de log-densité
- Optimisation stochastique et Langevin dynamics
- Transport optimal : distance Wasserstein et applications
- Relation OT / diffusion models
- Algorithmes : Sinkhorn, OT régularisé
- Génération guidée par transport optimal
- Exemples : transfert de style, morphing de distributions
- Exos

IA générative et optimisation

- Beaucoup de modèles génératifs peuvent être vus comme des problèmes d'optimisation.
- Deux liens majeurs :
  1. Modèles basés sur le score (score-based generative models)
  2. Transport optimal et distances de Wasserstein

Score matching : principe

- Idée : apprendre le gradient du log de densité $\nabla_x \log p(x)$
- Approche introduite par Hyvärinen (2005)
- Fonction de coût :

$$\mathcal{L}(\theta) = \mathbb{E}_{x \sim p_{data}} \left[\frac{1}{2} \| s_\theta(x) - \nabla_x \log p(x) \|^2 \right]$$

où $s_\theta(x)$ est le réseau de score.

De score matching à génération

- Si on connaît le score $\nabla_x \log p(x)$, on peut générer via Langevin dynamics :

$$x_{t+1} = x_t + \eta\, s_\theta(x_t) + \sqrt{2\eta}\,\epsilon_t$$

- Procédé stochastique qui converge vers la distribution $p(x)$
- Base des modèles de diffusion actuels

Lien diffusion – score

- Modèles de diffusion = apprentissage du score à chaque niveau de bruit
- Score network : $s_\theta(x, t) \approx \nabla_x \log p_t(x)$
- Génération = résolution d'une équation différentielle stochastique (SDE)

Transport optimal : idée générale

- Problème : comment transformer une distribution μ en une autre ν au coût minimal ?
- Distance de Wasserstein $W_c(\mu, \nu)$:

$$W_c(\mu, \nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int c(x, y)\, d\gamma(x, y)$$

où $\Pi(\mu, \nu) =$ couplages de μ et ν.

- Coût typique : $c(x, y) = \|x - y\|^2$

Distance de Wasserstein : interprétation

- Vue comme « effort de transport » pour transformer une distribution en une autre.
- Structure géométrique sur l'espace des mesures de probabilité.
- Fournit un cadre naturel pour l'IA générative.

Wasserstein GAN (WGAN)

- Amélioration des GAN classiques par distance Wasserstein
- Critère d'entraînement :

$$\min_{G} \max_{D \in \text{Lip}(1)} \mathbb{E}_{x \sim p_{data}}[D(x)] - \mathbb{E}_{z \sim p(z)}[D(G(z))]$$

- Donne gradients plus stables et convergence meilleure

Optimal transport et diffusion

- Les modèles de diffusion peuvent être vus comme un chemin optimal entre distributions
- Lien avec la formulation de Schrödinger bridge (transport stochastique)
- Apporte une vision unifiée entre diffusion et transport optimal

Applications IA générative

- Génération d'images : WGAN, diffusion models
- Modélisation moléculaire : transport optimal pour aligner distributions
- Simulation physique : apprentissage de dynamiques via score matching

Synthèse : IA générative et optimisation

- Score matching : apprentissage du gradient de log densité
- Transport optimal : géométrie des distributions
- Modèles modernes = combinaison des deux (diffusion, Schrödinger bridge, WGAN)

Motivation : optimisation et IA générative

- Générer des données réalistes nécessite d'optimiser sur la distribution cible
- Score-based models et transport optimal offrent une formulation géométrique et mathématiquement solide
- Applications : diffusion guided generation, morphing, transfert de style

Score-based generative modeling

- Score : gradient log-densité de la distribution

$$s_\theta(x) = \nabla_x \log p_\theta(x)$$

- Modèle entraîne $s_\theta(x)$ sur données bruitées
- Génération via Langevin dynamics :

$$x_{t+1} = x_t + \frac{\eta}{2} s_\theta(x_t) + \sqrt{\eta}\epsilon_t$$

Forward / Reverse Dynamics

- Forward : ajouter bruit progressif (diffusion)
- Reverse : enlever bruit via score-based update
- Connection : score-based models = continuous limit de diffusion models

Langevin dynamics pour génération

- Discrétisation Euler-Maruyama

$$x_{t+1} = x_t + \frac{\eta}{2} s_\theta(x_t) + \sqrt{\eta}\epsilon_t$$

- $\epsilon_t \sim \mathcal{N}(0, I)$
- η : step size
- Permet sampling à partir de $p_\theta(x)$

Transport optimal et génération

- Comparer distributions μ et ν
- Distance de Wasserstein :

$$W_p(\mu,\nu) = \left(\inf_{\gamma \in \Pi(\mu,\nu)} \int \|x-y\|^p d\gamma(x,y) \right)^{1/p}$$

- Génération guidée : rapprocher distribution générée p_θ de p_{data} en OT

Sinkhorn algorithm pour OT régularisé

- OT régularisé entropiquement :

$$\min_{\gamma} \int c(x,y)d\gamma + \epsilon \mathrm{KL}(\gamma||\mu \otimes \nu)$$

- Sinkhorn iterations pour calcul efficace
- Applications : transfert de style, morphing distributions

Combinaison score-based + OT

- Score-based : optimiser densité locale
- OT : optimiser distance globale entre distributions
- Ensemble : génération stable et réaliste

Exercice : score-based sampling

- Implémenter Langevin dynamics sur dataset 2D toy
- Visualiser convergence vers distribution cible

Exercice : OT régularisé

- Calculer Wasserstein distance entre deux distributions 2D
- Implémenter Sinkhorn iterations
- Visualiser plan de transport optimal

Exercice : combinaison Score + OT

- Générer des points avec score-based Langevin
- Régulariser avec plan OT vers distribution cible
- Observer effet sur distribution finale

Applications IA générative + OT

- Image : morphing, style transfer
- Texte $\rightarrow$ texte / image $\rightarrow$ image : distribution matching
- Physique : génération molécules, particules simulées

Diffusion models guidés par OT

- Reverse diffusion avec penalité Wasserstein
- Objectif : préserver structure globale des données
- Meilleure fidélité distributionnelle

Visualisation : Score-based + OT

- Points initiaux : bruit
- Reverse process : Langevin dynamics
- OT reg. : rapprochement distribution cible
- Schéma : trajectoires convergentes

Exercice avancé 1 : Morphing distributions

- Générer distribution A $\rightarrow$ B
- Utiliser score-based update + OT regularization
- Visualiser trajectoires

Exercice avancé 2 : Transfert de style images

- Dataset source : MNIST
- Dataset target : FashionMNIST
- Génération guidée par score + OT

Exercice avancé 3 : Score-based diffusion sur molécules

- Forward process sur positions atomiques
- Reverse process avec score-based Langevin
- OT pour rapprocher distribution moléculaire cible

Mini-cas : comparaison avec GAN et VAE

- Visualiser qualité et diversité
- Observer stabilité des modèles score-based + OT

Mini-cas : visualisation trajectoires

- Forward : diffusion du bruit
- Reverse : score-based update
- OT : ajustement global
- Observer convergence distribution cible

Résumé Sous-section 3

- Score-based generative models : gradient log-densité, Langevin dynamics
- Transport optimal : Wasserstein, Sinkhorn
- Combinaison : génération stable, fidèle à distribution cible
- Applications : morphing, style transfer, diffusion guided, molécules

Transition vers synthèse du module IA générative

- Récapitulatif :
  - Modèles classiques : VAE, GAN, Flows
  - Modèles diffusion et fondation : Stable Diffusion, GPT
  - Optimisation avancée : score-based + OT
- Module complet : théorie, mathématiques, applications, exercices

Synthèse du module : IA générative

- **Modèles classiques : VAE, GAN, Flows**
  - VAE : latent continu, ELBO, reconstruction
  - GAN : générateur/adversaire, échantillons réalistes mais instables
  - Flows : transformations bijectives, likelihood exacte
- **Diffusion models et modèles fondation**
  - Forward noise + reverse denoising
  - Stable Diffusion : text-to-image, embeddings textuels
  - GPT : autoregressive Transformer
- **Optimisation avancée et génération guidée**
  - Score-based models : gradient log-densité, Langevin dynamics
  - Transport optimal : Wasserstein distance, Sinkhorn iterations
  - Combinaison score + OT : génération stable et fidèle à la distribution cible
- **Applications**
  - Images, audio, texte, molécules, style transfer
  - Études de cas et exercices pratiques pour compréhension et visualisation

Prochaines étapes

- Intégration des concepts de géométrie des données et IA générative
- Exploration avancée :
  - Deep learning géométrique (GNN, graphes et variétés)
  - Applications scientifiques et industrielles
- Projet final : combinaison théorie + implémentation + visualisation