



Normandie Université

THÈSE

Pour obtenir le grade de Docteur de Normandie Université

Spécialité Informatique

l'École Doctorale Mathématiques, Information, Ingénierie des Systèmes

Auxiliary Tasks for the Conditioning of Generative Adversarial Networks Tâches auxiliaires pour le conditionnement des réseaux antagonistes génératifs

Présentée et soutenue par
Cyprien RUFFINO

Dirigée par Gilles GASSO et Romain HÉRAULT

Thèse soutenue publiquement le Friday 15th May, 2020
devant le jury composé de

Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse

Abstract

Résumé

Remerciements

Contents

Contents	VII
List of Figures	IX
List of Tables	XI
List of Acronyms	XIII
Introduction	1
Context	1
Motivations	1
Contributions	1
Outline	1
1 Generative Adversarial Networks : Principles, strengths and limitations	3
1.1 Generative modeling	4
1.2 Adversarial models	7
1.3 The GAN Zoo	13
2 Reconstruction as an Auxiliary Task for Generative Modeling	15
2.1 Image Reconstruction with Generative Models	15
2.2 Conditional generation as a Maximum A Posteriori estimation	16
2.3 Experimental evaluation and application to underground soil generation	16
2.4 Conclusion	16
3 Conditioning generation with multiple task-specific constraints	19
3.1 Introduction	19
3.2 Proximal method for non-Euclidean output space	19
3.3 Application to RGB to Polarimetric domain transfer	19
3.4 Conclusion	19
4 Conclusion and Perspectives	21
Bibliography	21
A Publications	23
B Experiment details for the Pixel-Wise Conditioned GAN	25
C Experiment details for the Polarimetric CycleGAN	27

List of Figures

1.1	Generative modeling	5
1.2	Latent variable model	5
1.3	Variational auto-encoder	6
1.4	Illustration of a divergence	8
1.5	Generative Adversarial Networks framework	8
1.6	CycleGAN framework	11
1.7	Reverse KL (left) and KL (right) divergence between the true blue distribution and the mode-collapsed red distribution . The distance is low in the case of the reverse KL, even if a missing mode is clearly visible.	12
1.8	Classifications of some advances in GANs on the trilemma	14
2.1	The problems of inpainting and image reconstruction	16
2.2	Generation of a sample during training	16

List of Tables

Acronyms

CGAN	Conditional Generative Adversarial Networks
CycleGAN	Cycle-Consistent Generative Adversarial Networks
ELBO	Evidence Lower Bound
FID	Fréchet Inception Distance
GAN	Generative Adversarial Networks
GMM	Gaussian Mixture Model
IS	Inception Score
JS	Jensen-Shannon (Divergence)
KL	Kullback-Leibler (Divergence)
MSE	Mean-Squared Error
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
VAE	Variational Auto-Encoder

Introduction

Context

Generic deep learning introduction, generic introduction to generative modeling (image generation, whichfaceisreal.com, etc...)

Generative Adversarial Networks (GAN) **Goodfellow2014** have been recently highlighted for their ability to generate photo-realistic images. By providing a simple framework for high-quality, high-dimensional generative modeling, they quickly found real-world applications ranging from the notorious "deepfakes" **Vaccari2020** to image super-resolution **Wang2020**.

Introduction to applied conditional generative modeling : examples others than geology

Motivations

Geostatistical application : introduction and needs

- Tuneable (quality vs enforcement of the constraints)
- Pixel-precise
- Keeping diversity

Polarimetry application : introduction and needs

- Designing custom-made constraints for the problem
- Non-euclidian
- Compatible with domain transfer

Contributions

Outline

Chapter 1

Generative Adversarial Networks : Principles, strengths and limitations

Chapter abstract

In this chapter, we first propose an introduction to the problem of generative modeling and some solutions to tackle this problem. We then propose an overview of the Generative Adversarial Networks (Goodfellow2014) framework, which is a recent method to train deep neural networks as generative models that is particularly adapted to the task of image generation. We will introduce some of its theoretical interpretations, as well as some of its variations and applications. We discuss the different limitations of this approach and expose a trilemma between the quality of the generated samples, their diversity and the conditioning of the model. We then discuss the recent advances that have been made to overcome some of these limitations and propose a taxonomy of these advances using the aforementioned trilemma. Finally, we discuss the evaluation of generative models and the difficulties of evaluating the intrinsic quality of a generated sample. We propose an overview of the different classical metrics and discuss their limitations.

Contents

1.1 Generative modeling	4
1.1.1 Generative modeling with maximum likelihood estimation	4
1.1.2 Latent variable models	4
1.2 Adversarial models	7
1.2.1 Generative Adversarial Networks	7
1.2.2 Conditional modeling with CGANs	9
1.2.3 Domain-transfer with GANs	10
1.2.4 Limitations	10
1.2.5 A note on the evaluation of generative models	12
1.3 The GAN Zoo	13
1.3.1 A taxonomy of GANs	13
1.3.2 Architecture variants	14
1.3.3 Divergence variants	14
1.3.4 Task-specific losses	14

1.1 Generative modeling

we will first propose an overview of generative modeling

Generative modeling with deep neural networks has been a challenging task due to the stochastic nature of sampling, which prevents the computation of gradient, thus preventing the training of a deep model with gradient descent. [CR: TODO](#)

1.1.1 Generative modeling with maximum likelihood estimation

Generative modeling is the task of learning the underlying distribution of a dataset in order to generate more samples from that distribution. In other words, it describes how data is generated in terms of a probabilistic model, a distribution from which the whole dataset could have been sampled with a high likelihood.

Indeed, whereas a discriminative model tries to find decision boundaries by fitting a parametric model $p_\theta(y|x)$ to a conditional probability distribution $p(y|x)$ of data $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$ relatively to the data $x \sim p(x)$, a generative model aims to fit $p_\theta(x)$ to $p(x)$ the intrinsic distribution of the data and to provide a sampling mechanism on this distribution (see Figure. 1.1).

These two learning tasks, the discriminative (Equation. (1.1)) modeling and the generative modeling (Equation. (1.2)) can be formulated as a maximum log-likelihood estimation

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x,y \sim p(y|x)} \log p_\theta(y|x) \quad (1.1) \quad \theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p(x)} \log p_\theta(x) \quad (1.2)$$

An simple example of generative model are Gaussian Mixture Models (GMM) . They consist in a sum of K Gaussian distributions $\mathcal{N}(\mu_k, \sigma_k^2)$, $k \in 1..K$ which are all attributed a selection probability $p(z = k) = \pi_k$, with $z \in \mathcal{Z}$, so that $p(x|z = k) = \mathcal{N}(\mu_k, \sigma_k^2)$. The model is then formulated as

$$p_\theta(x) = \sum_z p(z) p_\theta(x|z) ,$$

with log-likelihood

$$\log \sum_{x \sim p(x)} p_\theta(x) = \sum_{x \sim p(x)} \log \sum_{k=1}^k \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2) .$$

In the case of the GMMs, the Expectation-Maximization (EM) algorithm (**Dempster1977**) can be used to find the parameters θ^* which, at convergence, maximizes the log-likelihood of the model. Once the model is trained, sampling a new data is done by picking a component k from the distribution $p(z)$, then drawing a sample from the Gaussian distribution $p(x|z = k) = \mathcal{N}(\mu_k^*, \sigma_k^{*2})$.

1.1.2 Latent variable models

Latent variable models and marginalization

For GMMs, sampling a new point consists in, once the Gaussian component has been selected, sampling a point on a normal distribution. This sampling can be done by using reparametrization: instead directly sampling $x \sim \mathcal{N}(\mu_k^*, \sigma_k^{*2})$, we can instead sample a latent variable $z \sim \mathcal{N}(0, 1)$ and compute $x = G(z; \mu, \sigma) = \mu + z\sigma$. Such a model, that consists in a deterministic function $G: \mathcal{Z} \rightarrow \mathcal{X}$ with parameters θ applied to a random latent variable drawn from a fixed distribution $p(z)$ is a latent variable model.

Since more complex distributions does not necessarily provide a natural sampling mechanism, using a latent variable model allows to outsource the stochastic part of the sampling process from the learning process and only learn the function $G(z; \theta)$. More formally, instead of directly modeling $p(x)$, a latent variable model learns a deterministic mapping $p_G(x|z)$. From this mapping, the generative model can be obtain through marginalization

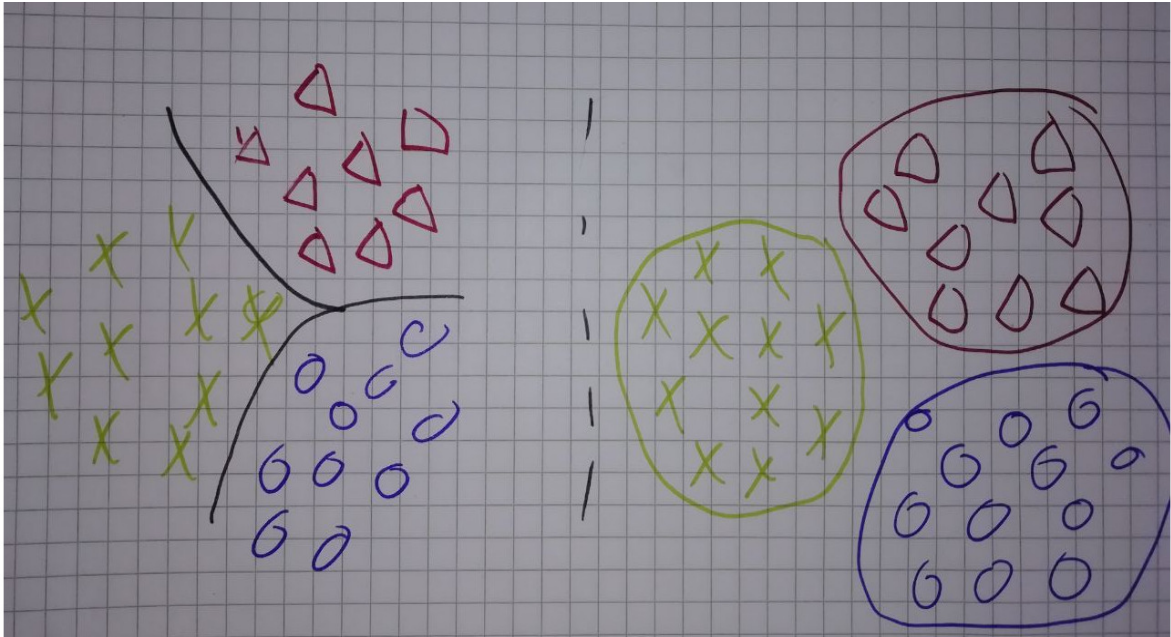


Figure 1.1: Left: Discriminative modeling, the model aims to find decision boundaries between classes. Right: Generative modeling, the model aims to learn the probability distribution of the data.

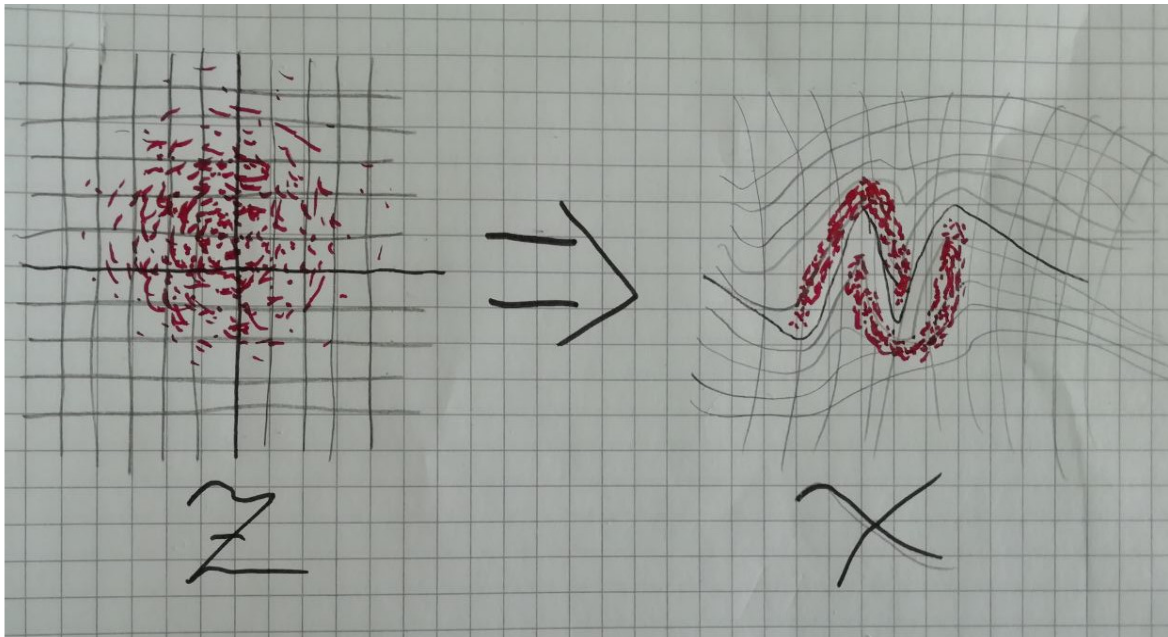


Figure 1.2: A mapping between a latent space \mathcal{Z} and the space of a dataset \mathcal{X} .

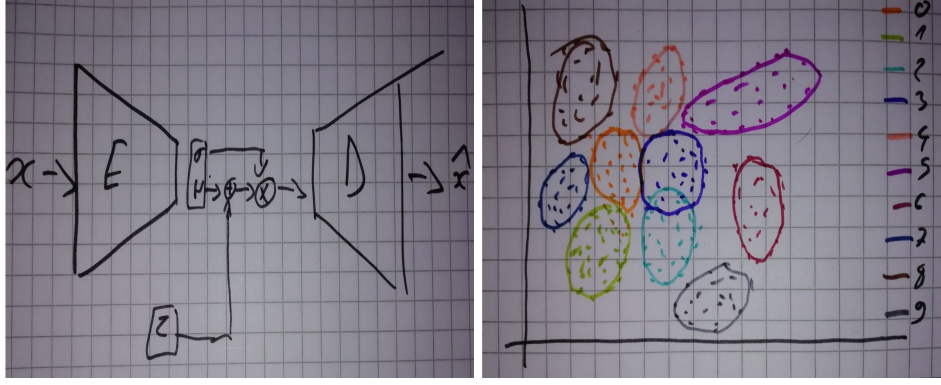


Figure 1.3: Left: the Variational auto-encoder global architecture. Right: the latent space of a VAE trained on the MNIST (**Lecun1998**) dataset. We can observe a Gaussian component per class in the latent space,

$$p_G(x) = \int_{\mathcal{Z}} p(z) p_G(x|z) dz = \int_{\mathcal{Z}} p(z) p(x|G(z; \theta)) dz . \quad (1.3)$$

This marginalization allows for the use of an arbitrary flexible G . However, if G is non-linear, the actual evaluation of $p_G(x)$ is very likely to be intractable due to the integral over \mathcal{Z} , which prevents the training of such a model as is.

While the marginal distribution $p_G(x)$ cannot be explicitly computed for any function G , several solutions exist to overcome this problem and train deep generative models with latent variables anyways.

Variational auto-encoders

Variational Auto-Encoders (VAE) (**Kingma2014b**) are deep latent variable models which tackle the marginalization problem by approximating the integral using a variational approach. To this end, they both learn the distribution of the latent model $p_G(x|z)$ as well as the distribution $q_F(z|x)$. This is done with with two different neural networks, a decoder network $G : \mathcal{Z} \rightarrow \mathcal{X}$ and an encoder network $F : \mathcal{X} \rightarrow \mathcal{Z}$ and allows to compute the distribution $p(x)$ as

$$\log p_G(x) - D_{\text{KL}}(q_F(z|x) \parallel p(z|x)) = \mathbb{E}_{z \sim q_F(z|x)} [\log p_G(x|z)] - D_{\text{KL}}(q_F(z|x) \parallel p(z)) .$$

The KL terms evaluates the distance between the distribution $q_F(z|x)$ learned by the encoder and real distribution $p(z|x)$, and since $p(z)$ is chosen Gaussian, this KL terms can be explicitly computed. The first term, is equivalent to the reconstruction error of an auto-encoder. Hence the model is trained by minimizing

$$L_{\text{VAE}}(F, G) = \mathbb{E}_{z \sim q_F(z|x)} [\|x - G(z)\|_2^2] - D_{\text{KL}}(q_F(z|x) \parallel p(z))$$

However, since sampling $z \sim q_F(z|x)$ is not differentiable, the VAE uses the so-called *reparametrization trick*, that is to have $F(x)$ output the mean and the variance (μ_x, σ_x^2) of a normal distribution for a sample x , so that a $z' \sim \mathcal{N}(0, 1)$ is sampled outside of the model and given as a parameter, thus allowing to compute $z = \mu_x + \sigma_x z'$, which is differentiable by considering z' a parameter.

Finally, generating a sample x with the trained model can be done by sampling a random vector $z \sim \mathcal{N}(0, 1)$ and computing $x = G(z)$.

Normalizing flows

Normalizing flow based techniques is a family of latent variable models that aim to tackle the marginalization problem by using the *change of variable formula*

$$p_G(x) = p(z) \left| \det \left(\frac{\partial G(z)}{\partial z^T} \right) \right|^{-1} = p(G^{-1}(x)) \left| \det \left(\frac{\partial G^{-1}(x)}{\partial x^T} \right) \right| ,$$

with $z \sim p(z)$ a latent variable. This formulation has notable advantages such as explicitly allowing the computation of the exact inference. However, the model has to enforce some tough constraints: the input and output dimensions must be the same; G must be invertible; and computing the determinant of the Jacobian needs to be efficient and differentiable.

These constraints can be enforced through strong restrictions on the architecture of the model. By limiting the transformations to a set of invertible transformations with a tractable Jacobian determinant, the model remains invertible and the determinant of its Jacobian can be computed efficiently.

Real-valued non-volume preserving (RealNVP) normalizing flows (**Dinh2017**) uses affine coupling transformations, which transforms a variable by adding and scaling it by a non-linear transformation of itself, usually computed with deep neural networks. These transformations can be inverted by subtracting and downscaling by the same transformed variables and their Jacobian is triangular, therefore computing its determinant can be done efficiently by computing the product of its diagonal terms. *Glow* (**Kingma2018**) extended this set of transformations to 1×1 invertible convolutions as well as a variant of batch normalization that allows for more expressiveness in the model.

1.2 Adversarial models

In this section, we will focus on the Generative Adversarial Networks (**Goodfellow2014**) framework, their training process and some of their variants, especially their conditional and domain-transfer variants.

We will then outline some limitations of this framework and propose a formulation of these limitations in the form of a trilemma between the intrinsic quality of the generated samples, their diversity and the quality of the conditioning of the model. With this tool, we propose a taxonomy of the recent GAN approaches and identify trends in these approaches.

1.2.1 Generative Adversarial Networks

In the same fashion as the generative models mentioned in Subsection. 1.1.2, Generative Adversarial Networks (GANs) (**Goodfellow2014**) aims to learn a parameterized mapping $p_G(x|z)$ between a simple distribution $p(z)$ (usually normal or uniform) to the real data distribution $p(x)$. However, instead of relying on the likelihood and trying to estimate the distribution through marginalization, it aims to minimize an estimation of a divergence between $p(x)$ and the mapped distribution $p_G(x)$. Therefore, GANs are often qualified as *likelihood-free* generative models.

Since a divergence $\text{Div}(p(x)||q(x))$ between two distributions $p(x)$ and $q(x)$ is analogous to a distance between these distributions (see Figure. 1.4), minimizing such a divergence allows for a parametric distribution $p_\theta(x)$ to fit a target distribution $p(x)$. When this divergence is both tractable (or estimable) and differentiable w.r.t the parameters θ , it can be directly optimized, allowing for the training of a generative model.

However in practice, such divergences usually intractable in the case of generic distributions. GANs aim to estimate this divergence by relying on a second learned function that will act as a surrogate to the divergence, the discriminator model D . This discriminator is a binary classifier that aims to predict the probability that a sample x was sampled on the real distribution $p(x)$ or was generated from $z \sim p(z)$ and is trained with binary cross-entropy

$$L_D(D, G) = \mathbb{E}_{x \sim p(x)} [\log D(x)] + \mathbb{E}_{x \sim p_G(x)} [1 - \log D(x)] .$$

The intuition behind this model is that once the discriminator is trained, maximizing its error on generated samples $x \sim p_G(x)$ w.r.t the parameters of G should push $p_G(x)$ towards $p(x)$.

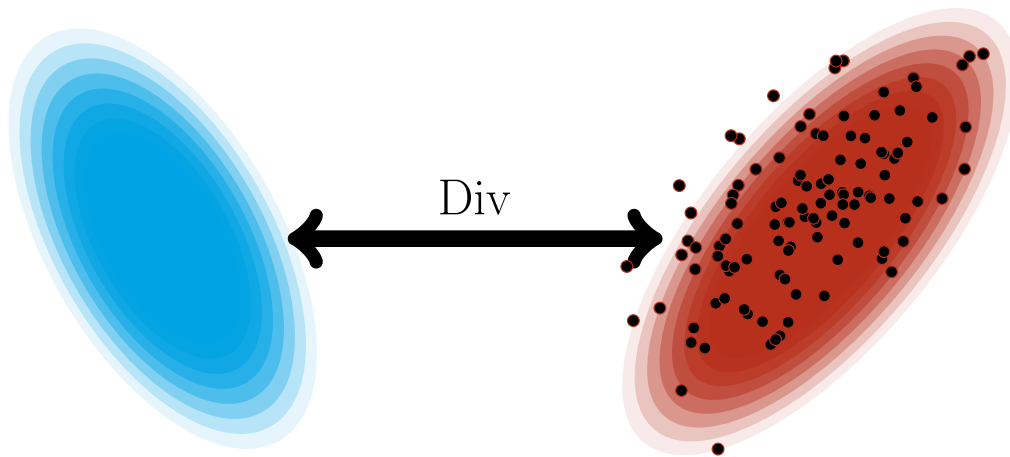


Figure 1.4: A divergence can capture the distance between a (parametric) **distribution** and the estimated distribution of a set of data points \mathcal{X}



Figure 1.5: Generative Adversarial Networks framework

The minimum of $f(x) = a \log(x) + b \log(1 - x)$ is $\frac{a}{a+b}$, so the discriminator that maximizes $L_D(D, G)$ for a fixed G is given by

$$D_G^*(x) = \frac{p(x)}{p(x) + p_G(x)} .$$

By plugging this optimal into the discriminator cost, we get

$$\min_D L_D(D, G) = \mathbb{E}_{x \sim p(x)} \left[\log \frac{p(x)}{p(x) + p_G(x)} \right] + \mathbb{E}_{x \sim p_G(x)} \left[1 - \log \frac{p_G(x)}{p(x) + p_G(x)} \right] .$$

As said previously, the objective of the generator model G will be to maximize the error of the discriminator D . Thus, we can formulate a criterion $L_G(G)$ as $L_G(G) = \min_D L_D(D, G)$. Up to additive and multiplicative constants, the criterion $L_G(G)$ can be reformulated as

$$L_G(G) = D_{\text{KL}}\left(p(x) \left\| \frac{p(x) + p_G(x)}{2} \right\| \right) + D_{\text{KL}}\left(p_G(x) \left\| \frac{p(x) + p_G(x)}{2} \right\| \right) = 2 \cdot D_{\text{JS}}\left(p(x) \left\| p_G(x) \right\| \right) .$$

When the discriminator is trained to convergence, minimizing the criterion $L_G(G) = L_{\text{GAN}}(D^*, G)$ is equivalent to minimizing the Jensen-Shannon (JS) divergence between $p(x)$ and $p_G(x)$. This training process is summed up as a mini-max game in Equation. (1.4)

$$\arg \min_G \max_D L_{\text{GAN}} = \arg \min_G \max_D \mathbb{E}_{x \sim p(x)} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [1 - \log D(G(z))] . \quad (1.4)$$

As shown above, this mini-max game has, assuming infinite capacity for both G and D , a global optimum for $p(x) = p_G(x)$. The GAN training algorithm then consists in alternatively updating the discriminator and the generator via gradient ascent/descent. A summary of this process is presented in Algorithm. 1.

Algorithm 1 The GAN training algorithm

Require: \mathcal{D}_X the real dataset, G the generator model, and D the discriminator model

repeat

 sample a mini-batch $\{x_i\}_{i=1}^m$ from \mathcal{D}_X

 sample a mini-batch $\{z_i\}_{i=1}^m$ from $p(z)$

 update D by stochastic gradient ascent of

$$\sum_{i=1}^m \log(D(x_i)) + \log(1 - D(G(z_i)))$$

 sample a mini-batch $\{z_j\}_{j=1}^n$ from distribution $p(z)$;

 update G by stochastic gradient descent of

$$\sum_{j=1}^n \log(1 - D(G(z_j)))$$

until a stopping condition is met

1.2.2 Conditional modeling with CGANs

While classical generative models such as GANs try to unconditionally approximate the real-data distribution $p(x)$, a conditional generative model aim to learn a model of the conditional distribution $p(x|y)$, where $y \in \mathcal{Y}$ is a label of any kind.

Several extensions of the GAN framework allow for conditional modeling. First introduced, Conditional GANs (CGANs) (Goodfellow2014) (Mirza2014) simply adds the label y as an input for both the discriminator and the generator. The new optimization problem that results from this change is summed-up in Equation. (1.5) as follows

$$\arg \min_G \max_D L_{\text{CGAN}} = \arg \min_G \max_D \mathbb{E}_{x, y \sim p(x, y)} [\log D(x, y)] + \mathbb{E}_{\substack{y \sim p(y) \\ z \sim p(z)}} [1 - \log D(G(y, z), y)] \quad (1.5)$$

While this approach is trivially simple to implement, it relies entirely on the discriminator to use the label. Other approaches try to learn the conditional distribution by adding an explicit loss term to the optimization problem, such as Auxillary Classifier GAN (ACGAN) (Odena2016). This approach aims to learn a conditional generative model with discrete labels by adding another output to the discriminator that acts as a classifier. The model is then trained by having both the generator and the discriminator minimize the categorical cross-entropy between the real and predicted labels.

1.2.3 Domain-transfer with GANs

Domain-transfer is the task of learning a mapping $p(x|y)$ between two high-dimensional distributions $p(x)$ and $p(y)$ that maintains semantic information, for example changing the color palette of an image while keeping the same objects at the same position. CGANs already learn to model the conditional distribution $p(x|y)$, and adding a way to enforce the consistency of the semantic information enables domain-transfer.

Pix2Pix (Isola2016) implemented this approach explicitly by using paired samples $(x, y) \sim p(x|y)$ forcing the generator to minimize the ℓ_1 reconstruction term between x and $G(y, z)$ (Equation. (1.6)).

$$\arg \min_G \max_D L_{p2p} = \arg \min_G \max_D L_{CGAN}(D, G) + \lambda \mathbb{E}_{\substack{x \sim p(x) \\ y \sim p(y) \\ z \sim p(z)}} \|x - G(y, z)\|_1 \quad (1.6)$$

However, this kind of approaches rely on paired data which can be very hard to obtain, especially in the case of natural images. When trying for example to transfer images of zebras to images of horses, you need a dataset of very similar zebras and horses in the exact same position for the ℓ_1 term to work.

This problem of paired data was solved by CycleGAN (Zhu2017) using cycle-consistency. Instead of training a single model G with reconstruction between x and $G(y, z)$, the CycleGAN approach train two domain-transfer models simultaneously: G_{YX} and G_{XY} that map samples from $p(y)$ onto $p(x)$ and $p(x)$ onto $p(y)$, respectively (see Figure. 1.6). This allows to compute the ℓ_1 reconstruction errors $\|x - G_{YX}(G_{XY}(x))\|_1$ and $\|y - G_{XY}(G_{YX}(y))\|_1$, thus completely removing the need for paired data (x, y) . The training of the two models is done in an adversarial setup, with two discriminators D_X and D_Y , and is summed up as an optimization problem in Equation. (1.7)

$$\arg \min_{G_{XY}, G_{YX}} \max_{D_X, D_Y} L_{CycleGAN} = \arg \min_{G_{XY}, G_{YX}} \max_{D_X, D_Y} L_{GAN}(D_X, G_{YX}) + L_{GAN}(D_Y, G_{XY}) \\ + \lambda \mathbb{E}_{x \sim p(x)} \|x - G_{YX}(G_{XY}(x))\|_1 + \lambda \mathbb{E}_{y \sim p(y)} \|y - G_{XY}(G_{YX}(y))\|_1 \quad (1.7)$$

The CycleGAN training process then consists in alternatively updating the two discriminator and the two generators via gradient ascent/descent. A summary of this process is presented in Algorithm. 2.

1.2.4 Limitations

GANs have shown strong advantages over the classical generative modeling methods, such as generating sharper samples than VAEs or taking significantly less time to train and to generate a sample than normalizing flows. They however bear limitations, namely: the instability of the training process; the lack diversity of the generated samples (*mode-collapse*); and finally the problems induced by the conditioning.

The instability of the GAN training process has first been conjectured to be caused by the bad quality of the gradients obtained when G generates bad samples, which makes D strongly reject these samples and therefore saturating the loss. The first solution proposed (Goodfellow2014)



Figure 1.6: CycleGAN framework

Algorithm 2 CycleGAN training algorithm

Require: \mathcal{X} and \mathcal{Y} two unpaired datasets, G_{XY} and G_{YX} the mapping networks, D_X and D_Y the discrimination models, m the mini-batch size

repeat

sample a mini-batch $\{x_i\}_{i=1}^m$ from \mathcal{X}

sample a mini-batch $\{y_i\}_{i=1}^m$ from \mathcal{Y}

update D_X by stochastic gradient descent of

$$\sum_{i=1}^m (D_X(x_i) - 1)^2 + (D_X(G_{YX}(y_i)))^2$$

update D_Y by stochastic gradient descent of

$$\sum_{i=1}^m (D_Y(y_i) - 1)^2 + (D_Y(G_{XY}(x_i)))^2$$

sample a mini-batch $\{x_i\}_{i=1}^m$ from X

sample a mini-batch $\{y_i\}_{i=1}^m$ from Y

update G_{XY} by stochastic gradient descent of

$$\sum_{i=1}^n (D_Y(G_{XY}(x_i)) - 1)^2 + \lambda(\|x_i - G_{YX}(G_{XY}(x_i))\|_1 + \|y_i - G_{XY}(G_{YX}(y_i))\|_1)$$

update G_{YX} by stochastic gradient descent of

$$\sum_{i=1}^n (D_X(G_{YX}(y_i)) - 1)^2 + \lambda(\|x_i - G_{YX}(G_{XY}(x_i))\|_1 + \|y_i - G_{XY}(G_{YX}(y_i))\|_1)$$

until a stopping condition is met

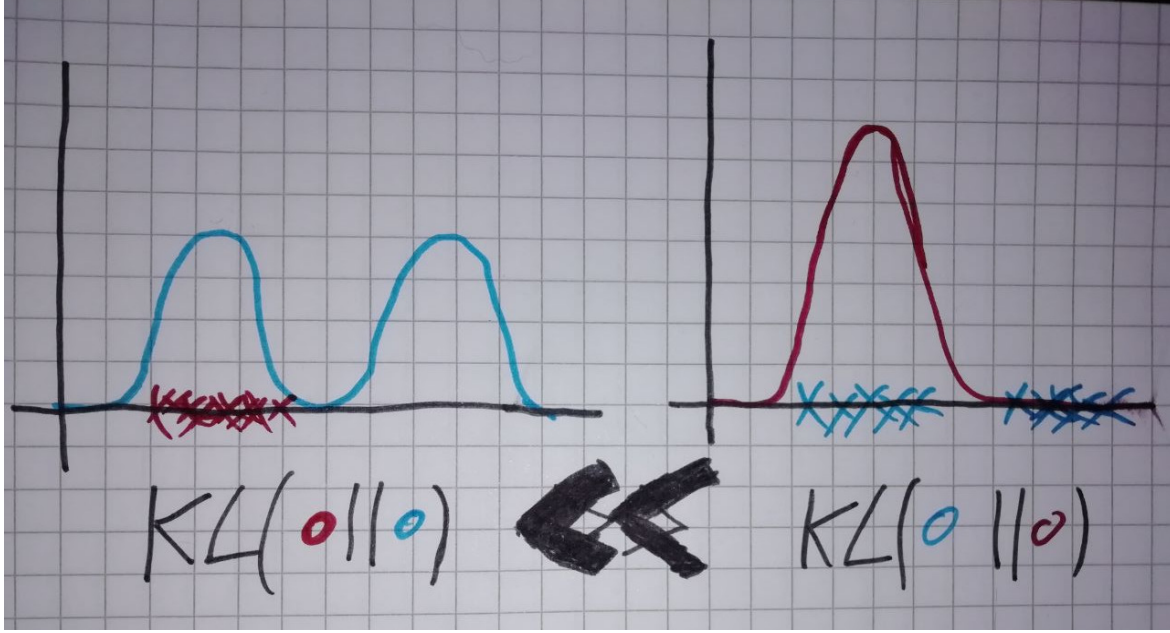


Figure 1.7: Reverse KL (left) and KL (right) divergence between the true blue distribution and the mode-collapsed red distribution. The distance is low in the case of the reverse KL, even if a missing mode is clearly visible.

was to slightly change the generator's loss function from $\log(1 - D(G(z)))$ to $-\log(D(G(z)))$, which helped considerably to avoid failures of the training process and was then widely used

While this loss term converges to the same minimum as the original loss term, minimizing it no longer correspond to minimizing a JS divergence but the non-symmetric reverse KL divergence (minus a JS term) (Arjovsky2017a). More formally,

$$\mathbb{E}_{z \sim p(z)} \left[\nabla_G \log D^*(G(z)) \right] = \nabla_G \left[D_{KL}(p_G(x) || p(x)) - 2D_{JS}(p_G(x) || p(x)) \right]$$

The main issue of the reverse KL divergence is that it does not penalize "missing" parts of the learned distribution $p_G(x)$, which are some points in the support of $p(x)$ that have zero (or near-zero) probability on $p_G(x)$ (see Figure. 1.7). This phenomenon leads to (*mode collapse*: different z_1, z_2 are mapped to samples $G(z_1)$ and $G(z_2)$ that are very close; and *mode dropping*: only a localized subset of the distribution can actually be mapped to. This issues raise a first trade-off, since using the original GAN creates instability which lead to a drop of visual quality, and using the non-saturating variant creates a lack of diversity.

While these issues already arise in the unconditional setting, su

CR:

Instability, catastrophic forgetting and the mode collapse problem

Trade-off image quality/diversity : Explanation through the loss term and distribution coverage

Black-box approach to conditioning, no tuning possible, no interpretability

1.2.5 A note on the evaluation of generative models

Unlike discriminative models, evaluating and comparing GAN approaches is a non-trivial task. Two approaches can be envisioned: evaluating the *intrinsic* quality of generated samples with ad-hoc criteria or directly evaluating the likelihood of the generated samples. However, unlike VAEs and flow-based models, GANs offer no explicit way to evaluate or approximate the likelihood of the generated samples. Thus, a significant part of the GAN literature resorted to a subjective visual evaluation of the generated samples.

In order to provide a more precise evaluation of the visual quality of generated samples, two ad-hoc methods Inception Score (IS) (**Salimans2016**) and Fréchet Inception Distance (FID) (**Heusel2017**) were proposed, which both make use of a pre-trained Inception v3 model (**Szegedy2016**), a deep classifier trained on the ImageNet dataset (**Deng2009**).

Inception Score (IS) (**Salimans2016**) is based on the evaluation of the entropy of the labels y predicted by the Inception classifier of generated data. High-fidelity samples should be easier to classify and therefore have a conditional label distribution $p_G(y|x)$ with low entropy. In addition to the high quality, the samples should be diverse, therefore the marginal distribution $p_G(y) = \int_{\mathcal{Z}} p_G(y|x = G(z)) dz$ should have a high entropy. By combining these two requirements, the IS is formulated as

$$\text{IS}(y) = \exp \left[\mathbb{E}_{x \sim p_G(x)} \text{D}_{\text{KL}} \left(p_G(y|x) \parallel p_G(y) \right) \right] .$$

Although it has been widely used, IS has shown major issues (**Barratt2018**) that raise from the use of the conditional label distribution. Most notably, examples that are correctly classified are not necessarily of the highest quality and the pre-determined label classes can skew the estimation of the marginal distribution $p_G(y)$.

The **Fréchet Inception Distance (FID)** (**Heusel**) differs from IS since it evaluates a distance between the distributions of visual features computed on real and generated data, instead of relying on the labels. These features are extracted at the penultimate layer of the Inception classifier. The distributions of these features are assumed Gaussian, so that the Fréchet distance (or Wasserstein-2 distance) can be computed as

$$\text{FID} = \|\mu - \mu_G\|^2 + \text{Tr}(\Sigma + \Sigma_G - 2\sqrt{\Sigma * \Sigma_G}) ,$$

where $\mathcal{N}(\mu, \Sigma)$ and $\mathcal{N}(\mu_G, \Sigma_G)$ are the distributions of the extracted features of the real and generated data, respectively. FID is considered more robust than IS and has been either completing or replacing the use of IS in recent works.

However, while these two metrics are considered to be the standard method for evaluating GANs, their reliance on the pre-trained Inception model can prove to be an issue. Indeed, they behave well when used to compare models learned on natural images datasets such as ImageNet, but they cannot directly be applied to other datasets. A solution to consider can be the training of another classifier network on a more adapted dataset, but this solution cannot be applied when no labeled data is available.

For completeness, we can also refer to less used metrics for evaluating visual quality: The Parzen window (or kernel density estimation) (**Parzen1962**) aim to estimate the likelihood of the generated samples; the Sliced Wasserstein Distance (**Julien2011**) is an efficient approximation of the Earth-Mover (or Wasserstein) distance; the Kernel Inception Distance (**Binkowski2018**) is a recent metric that evaluates the maximum mean discrepancy between Inception features with a polynomial kernel.

Finally it is to note that for conditioned models, evaluating the aforementioned metrics does not inform about the quality of the conditioning. However, since the conditioning usually requires either labels or prior information, these can often be evaluated by, for example, predicting the labels of generated samples with a pre-trained classifier and computing the error between the predicted label and the original one.

1.3 The GAN Zoo

1.3.1 A taxonomy of GANs

Enorme nombre de variantes de GANs

Taxonomie des approches GANs (pour s'éviter une liste des différents GANs)

Schéma pour définir les grandes familles de GAN (évoquer les AmbientGAN / UNIR)

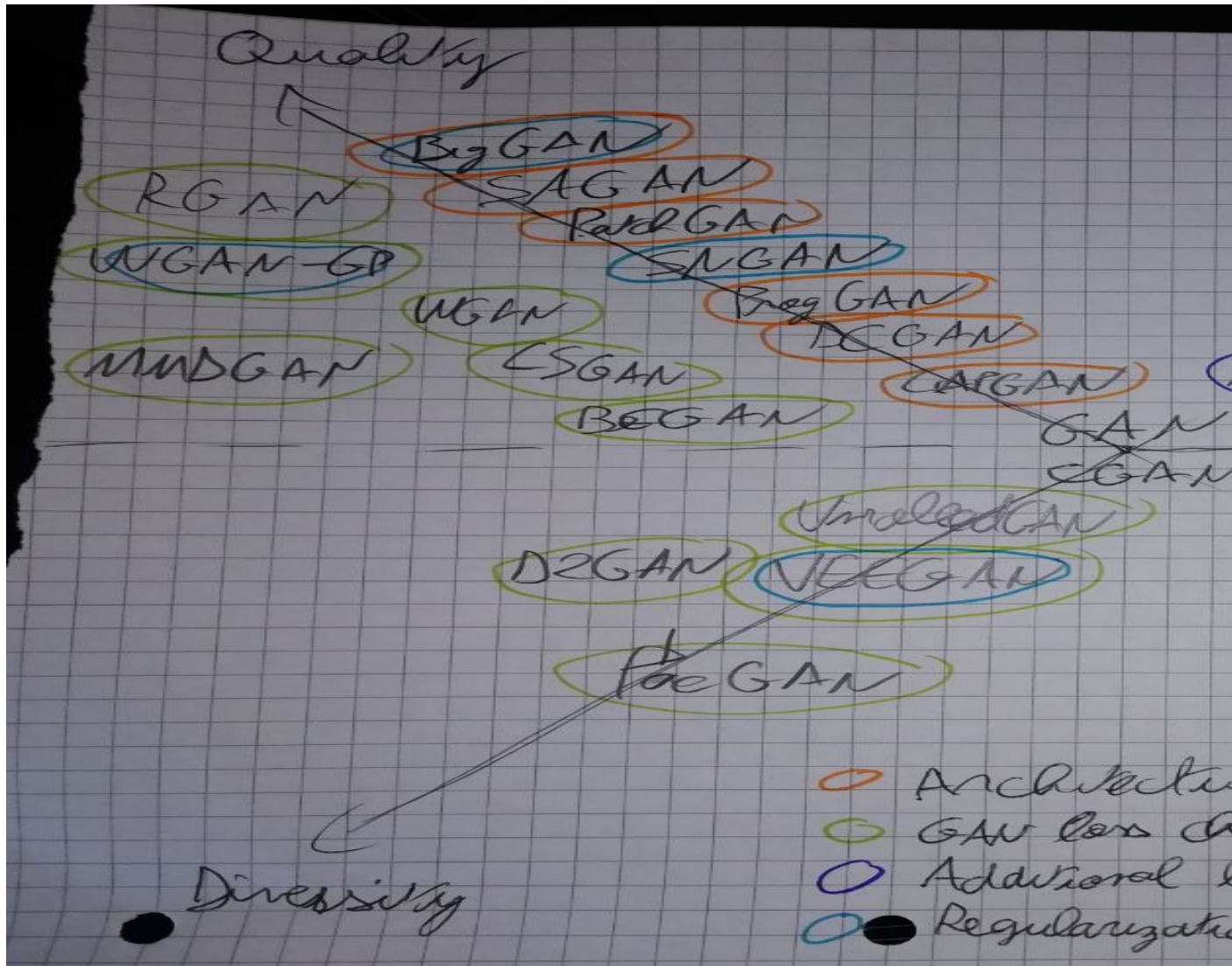


Figure 1.8: Classifications of some advances in GANs on the trilemma

1.3.2 Architecture variants

1.3.3 Divergence variants

Table des loss alternatives (f-divergences + transport optimal)

1.3.4 Task-specific losses

Chapter 2

Reconstruction as an Auxiliary Task for Generative Modeling

Chapter abstract

In this chapter, we propose an approach for conditioning a GAN model to reconstruct images from a very sparse set of randomly-positioned pixels known beforehand. This approach, based on a Maximum A Posteriori estimation, takes the form of an explicit auxiliary reconstruction task which adds to the GAN objective as an additional loss term. Complemented with the PacGAN variant for training GANs, this approach enables the generation of diverse samples from a sparse pixel map. As opposed to the more classical Conditional GAN approach, this auxiliary task is interpretable and a hyperparameter allows to control the importance of the conditioning in the learning process. We evaluate our approach on the classical MNIST, FashionMNIST and CIFAR10 datasets, as well as a custom-made texture dataset. Finally, we apply this approach to a task of geostatistical simulation.

Contents

2.1 Image Reconstruction with Generative Models	15
2.2 Conditional generation as a Maximum A Posteriori estimation	16
2.3 Experimental evaluation and application to underground soil generation	16
2.4 Conclusion	16

2.1 Image Reconstruction with Generative Models

Image reconstruction is the task of completing an image from a very small subset of the pixels. Such source data can usually be found in domains where the measurement process is very noisy or where measurements are expensive. This task differs from image inpainting since the source data is usually unstructured and very scarce, as in this chapter we will consider randomly scattered measurements of less than a percent of the image. While our discussion focus on image reconstruction, it is noteworthy to mention that this applies to other kinds of signals.

The task of image reconstruction is challenging since very few information is available for use. To overcome this lack of information, generative models such as GANs leverage on existing datasets to learn the distribution of the real images. By conditioning the learned distribution, a GAN could learn to generate an image while enforcing the constraint that the pixels known beforehand must remain similar in the generated image.

Similarly as in the GAN setup, we denote by $X \in \mathcal{X}$ a random variable and x its realization. Let p_X be the distribution of X over \mathcal{X} and $p_X(x)$ be its evaluation at x . Similarly $p_{X|Y}$ represents the distribution of X conditioned on the random variable $Y \in \mathcal{Y}$.

We denote by $x \in \mathcal{X} = [-1, 1]^{n \times p \times c}$ (see Figure 2.1a) an image sampled from an unknown distribution p_X and a sparse matrix $y \in \mathcal{Y} = [-1, 1]^{n \times p \times c}$ (Figure 2.1c) as the given constrained pixels.

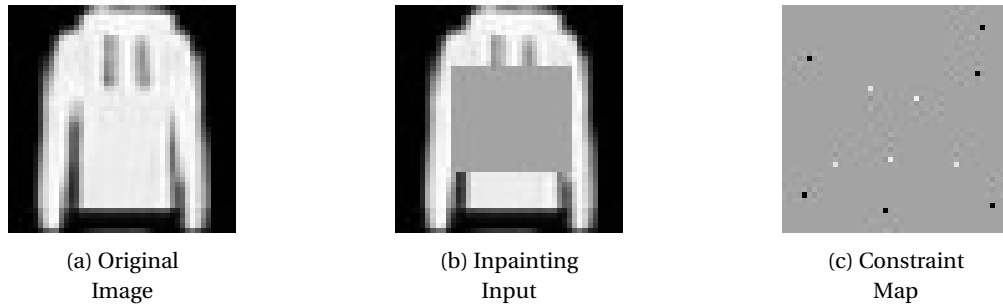


Figure 2.1: Difference between regular inpainting (2.1b) and the problem undertaken in this work (2.1c) on a real sample (2.1a).

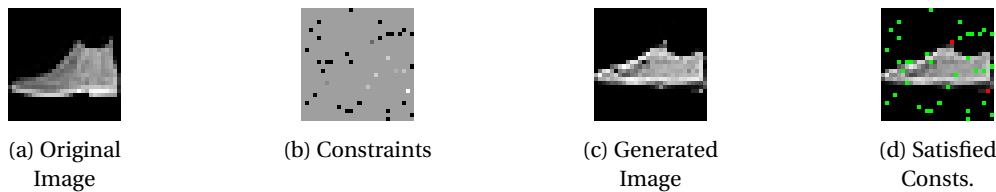


Figure 2.2: Generation of a sample during training. We first sample an image from a training set (2.2a) and we sample the constraints (2.2b) from it. Then our GAN generates a sample (2.2c). The constraints with squared error smaller than $\epsilon = 0.1$ are deemed satisfied and shown by green pixels in (2.2d) while the red pixels are unsatisfied.

The problem then consists in finding a generative model G with inputs z (a random vector sampled from a known distribution p_Z over the space \mathcal{Z}) and constrained pixel values $y \in [-1, 1]^{n \times p \times c}$ that maps the distribution p_Z onto the conditional distribution $p_{X|Y}$ of the real images given the constraints y (see Figure 2.2).

CR: [Related works : CGAN, AmbientGAN, UNIR, Compressed Sensing with Meta-Learning](#)
[Limitations of these models](#)

2.2 Conditional generation as a Maximum A Posteriori estimation

Approche de l'article NeuCom :

Formulation as a Maximum A Posteriori Estimation, assumptions (normal error)
 Construction of the loss term using bayes rule and least squares
 PacGAN for keeping the diversity

2.3 Experimental evaluation and application to underground soil generation

Datasets : MNIST/FashionMNIST/CelebA/Texture
 Evaluation : MSE/FID; Epoch selection criterion
 Architectures : Appendix ? DCGAN + SGAN (encoder-decoder)
 Results : visible trade-off, good fidelity overall
 Application to hydro-geology : subsurface dataset
 Evaluation : MSE/HOG+LBP

2.4 Conclusion

Objective reached : tuneable loss, pixel-wise, keeping diversity
 Applications in hydro-geology : papier Eric

Future works : other distributions (modelling error using Laplacian, beta or Poisson distributions)

Chapter 3

Conditioning generation with multiple task-specific constraints

Chapter abstract

content...

Contents

3.1 Introduction	19
3.2 Proximal method for non-Euclidean output space	19
3.3 Application to RGB to Polarimetric domain transfer	19
3.4 Conclusion	19

3.1 Introduction

Formulation as a constrained optimization problem

Reformulation of CycleGAN as a constrained optimization problem

Relaxation of the constraints

Ici, expérimenter sur des datasets artificiels ?

3.2 Proximal method for non-Euclidean output space

Travail sur le proximal ?

Envelope theorem application

3.3 Application to RGB to Polarimetric domain transfer

Introduction to polarimetry-specific physical constraints (briefly, no need to write a physics essay)

Reformulation as constraints on the output space

Relaxations : L_2 term + rectified term

Dataset, Evaluation

Experiments and results

3.4 Conclusion

Relaxation of the constraints works even when a lot of constraints are applied

The application to the polarimetric dataset works

Future works : using adapted metrics for the non-euclidean outspace

Chapter 4

Conclusion and Perspectives

Appendix A

Publications

Appendix B

Experiment details for the Pixel-Wise Conditionned GAN

Appendix C

Experiment details for the Polarimetric CycleGAN