

# Débruitage et régularisation par réseau de neurones

## Projet Industriel

ELKATEB Mohamed

FAINELLI Faustine

HOTZEL RUTHER Dominique

MACHADO SANTOS ROHDE Pedro

VILLON HUAMAN Alexandra

Mai 2019

# Table des matières

## 1 Introduction

## 2 Modèle développé dans l'article

- Modèle
- Architecture

## 3 Expériences réalisées

- Entrée du réseau
- Skip connections
- Méthodes d'optimisation

# Table des matières

## 1 Introduction

## 2 Modèle développé dans l'article

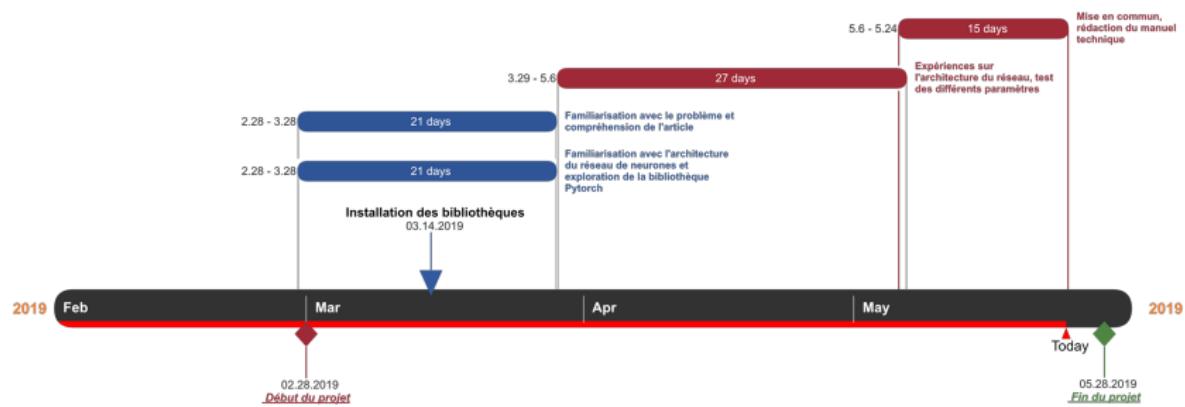
- Modèle
- Architecture

## 3 Expériences réalisées

- Entrée du réseau
- Skip connections
- Méthodes d'optimisation

# Introduction

## Diagramme de Gantt



# Table des matières

1 Introduction

2 Modèle développé dans l'article

- Modèle
- Architecture

3 Expériences réalisées

- Entrée du réseau
- Skip connections
- Méthodes d'optimisation

# Modèle(1)

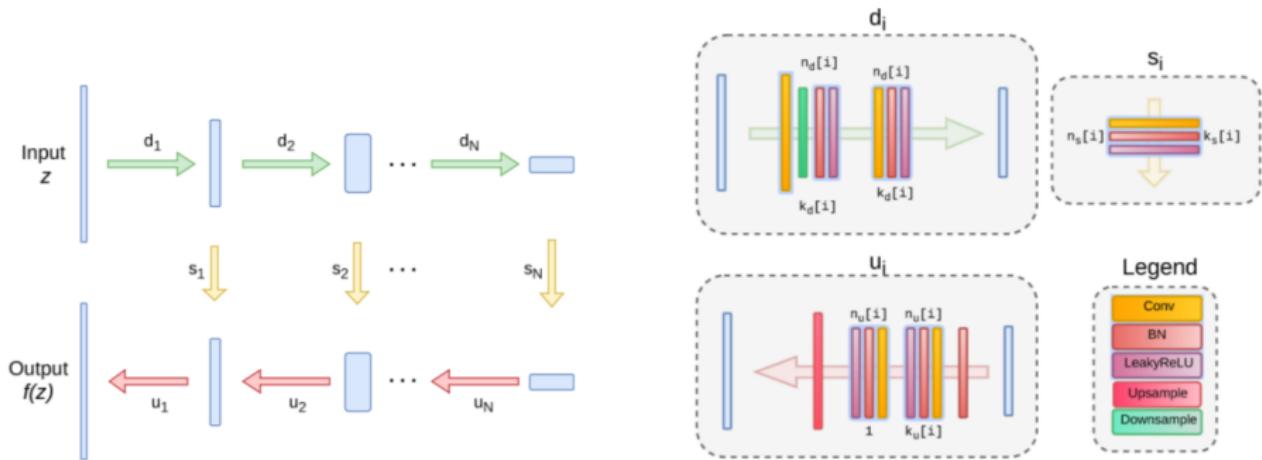
**But :** Montrer que la structure d'un réseau générateur est suffisante pour capturer un grand nombre de statistiques de bas niveau.

- ajustement d'un réseau générateur à une seule image dégradée
- poids de réseau = paramétrisation de l'image restaurée, initialisés et ajustés de manière aléatoire
- reconstruction = problème de génération d'image conditionnelle

## Modèle(2)

- Réseau de neurones = paramétrisation :  $x = f_\theta(z)$  de l'image  $x \in \mathbb{R}^{3 \times H \times W}$ , avec  $z$  un tenseur aléatoire fixe,  $\theta$  les paramètres du réseau (poids et biais des filtres).
- Minimisation d'énergie :  $x^* = \min_x E(x; x_0) + R(x)$ , avec  $E(x; x_0)$  un terme de données dépendant de l'application,  $x_0$  une image bruyante et  $R(x)$  un régularisateur.
- Prior implicite :  $\theta^* = \operatorname{argmin}_\theta(f_\theta(z); x_0)$ ,  $x^* = f_{\theta^*}(z)$ .

# Architecture(1)



**Figure –** Architecture de type décodeur-encodeur.  $n_u [i]$ ,  $n_d [i]$ ,  $n_s [i]$  correspondent respectivement au nombre de filtres sur la couche de profondeur  $i$  pour les connexions de suréchantillonnage ( $n_u$ ), de sous-échantillonnage ( $n_d$ ) et de saut ( $n_s$ ). Les valeurs  $k_u [i]$ ,  $k_d [i]$ ,  $k_s [i]$  correspondent aux tailles des noyaux respectifs.

## Architecture(2)

Implémentation par la fonction `get_net.py` :

```
elif fname == 'data/denoising/F16_GT.png':
    num_iter = 3000
    input_depth = 32
    figsize = 4

    net = get_net(input_depth, 'skip', pad,
                  skip_n33d=128,
                  skip_n33u=128,
                  skip_n11=4,
                  num_scales=5,
                  upsample_mode='bilinear').type(dtype)
```

**Figure –** La variable `num_iter` définit le nombre d'itérations lors de la phase d'optimisation, `input_depth` correspond à la profondeur du réseau neuronal : ici nous travaillons avec un réseau de 32 couches. `skip_n11` représente le nombre de skip connections, `skip_33d` et `skip_33u` représentent respectivement le nombre de canaux dans les couches montantes (upsample) et descendentes (downsample).

# Table des matières

## 1 Introduction

## 2 Modèle développé dans l'article

- Modèle
- Architecture

## 3 Expériences réalisées

- Entrée du réseau
- Skip connections
- Méthodes d'optimisation

# Entrée du réseau

- $z \in \mathbb{R}^{C \times H \times W}$
- $z \sim U(0, \frac{1}{10})$

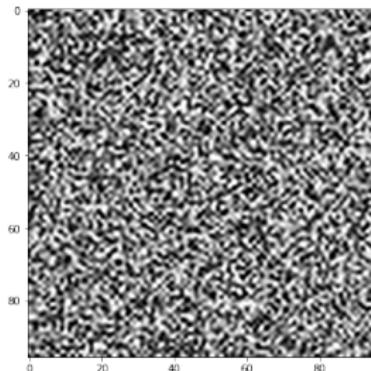


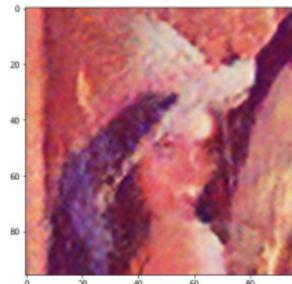
Figure – Un des canaux en entrée

# Entrée du réseau - après optimisation

$Z$  original



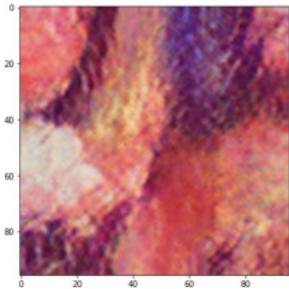
$Z + b, b \sim N(0, \frac{1}{10})$



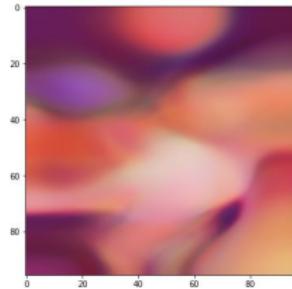
$kZ, k > 0$



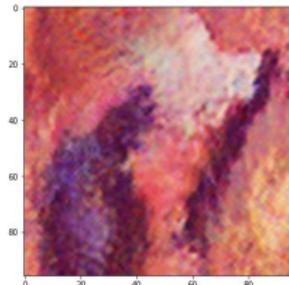
$kZ, k < 0$



$kZ, k = 0$



$Z \sim U(0, \frac{1}{10})$



# Entrée du réseau - régularisation durant l'optimisation

À chaque itération,  $Z' = Z + b$ ,  $b \sim N(0, \sigma)$

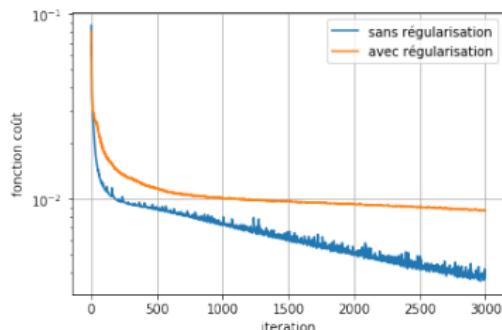


Figure – Fonction coût  
 $\sigma = 0$  vs.  $\sigma = \frac{1}{30}$

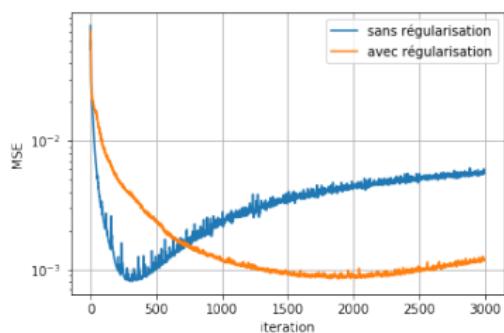


Figure – MSE image générée vs. bruitée  
 $\sigma = 0$  vs.  $\sigma = \frac{1}{30}$

# Entrée du réseau - avant optimisation

$Z$  à 90% de coefficients nuls

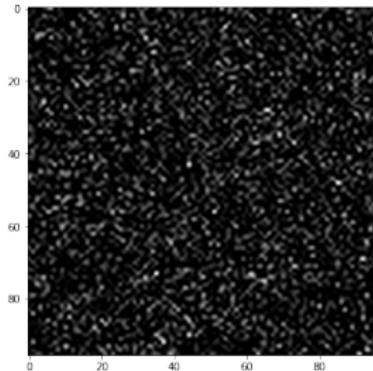


Figure – Un des canaux de l'entrée sparse

# Entrée du réseau - avant optimisation

$Z$  à 90% de coefficients nuls

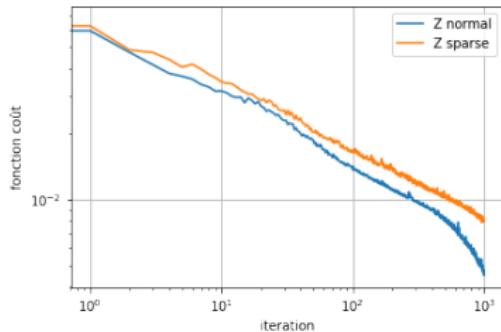


Figure – Fonction coût

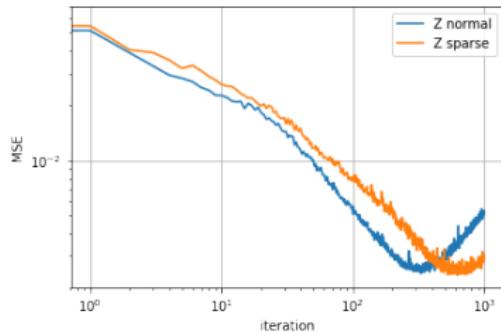


Figure – MSE image générée vs. bruitée

# Impact de la dimension spatiale de l'entrée

- C'est un paramètre couteux pour l'algorithme
- Sa valeur a été choisie de manière empirique

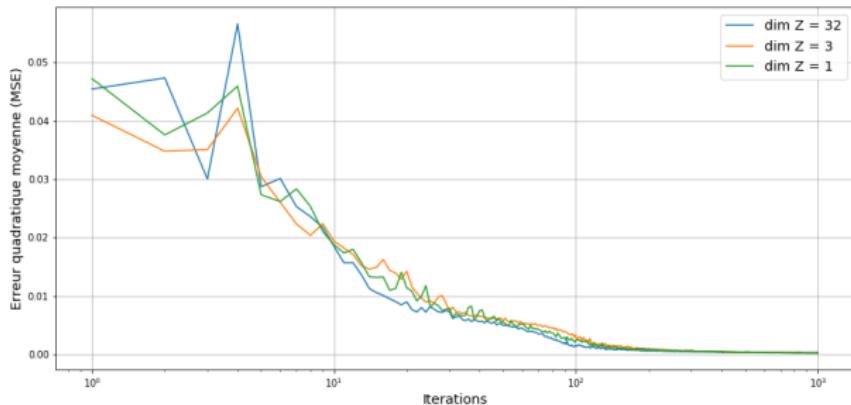


Figure – Evolution de la MSE pour 3 dimensions de Z

# Optimisation sur l'entrée

- $Z$  est un degré de liberté du système
- Certaines transformations sur  $Z$  laissent le résultat inchangé

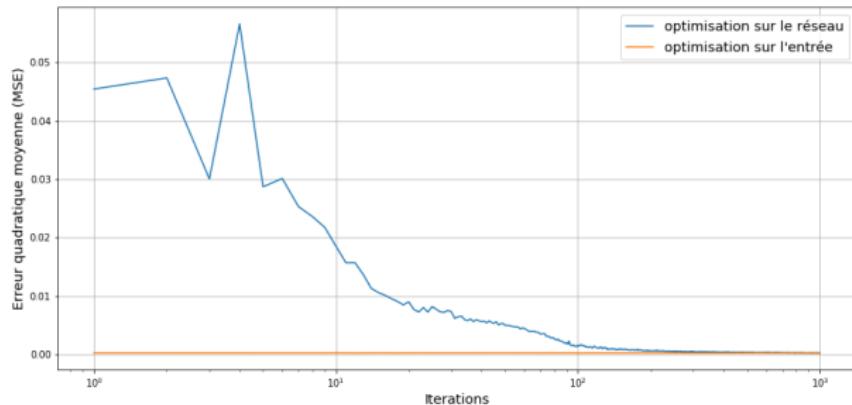


Figure – Optimisation sur le réseau suivie d'une optimisation sur l'entrée

# Optimisation sur l'entrée

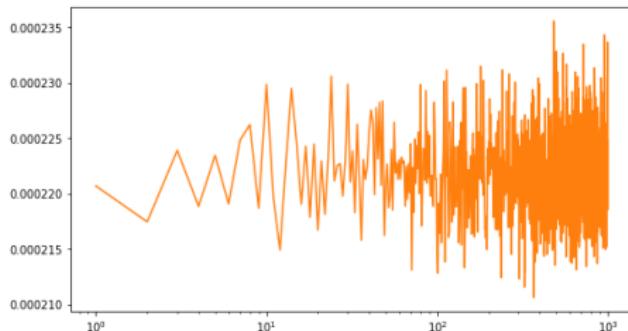


Figure – Optimisation sur l'entrée  $MSE=f(\text{iterations})$

⇒ Z réalise le minimum du MSE après optimisation des paramètres du réseau

# Effet de la diminution du nombre de couches

- $Z$  est un degré de liberté du système
- Certaines transformations sur  $Z$  laissent le résultat inchangé

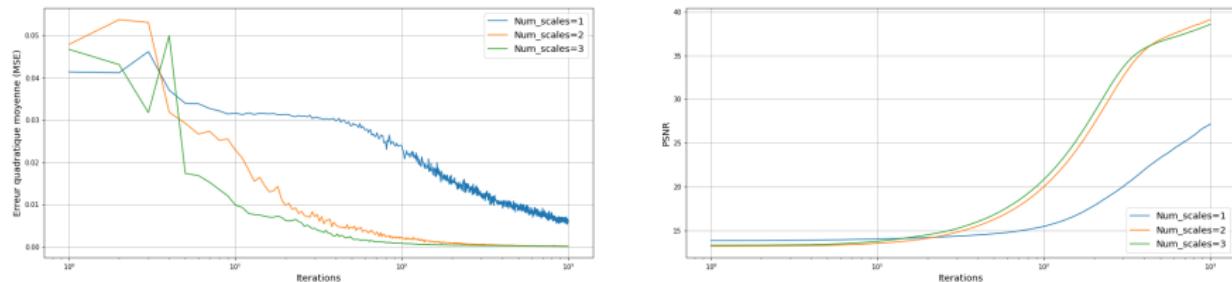


Figure – Evolution du MSE et du PSNR pour 3 profondeurs du réseau

- ⇒ Plus le réseau est profond, meilleure est la convergence
- ⇒ La qualité du débruitage est améliorée
- ⇒ Il existe un certain nombre de couches seuil à partir duquel "l'apprentissage" est similaire.

# Skip connections - Motivation

- Les *Skip connections* transmettent les détails de l'image des couches de convolution aux couches de déconvolution ;
- Convergence plus rapide ;
- Le paramètre *skip\_n11* : Nombre de canaux à sauter ;

# Skip connections - Résultats

## Résultats Quantitatifs

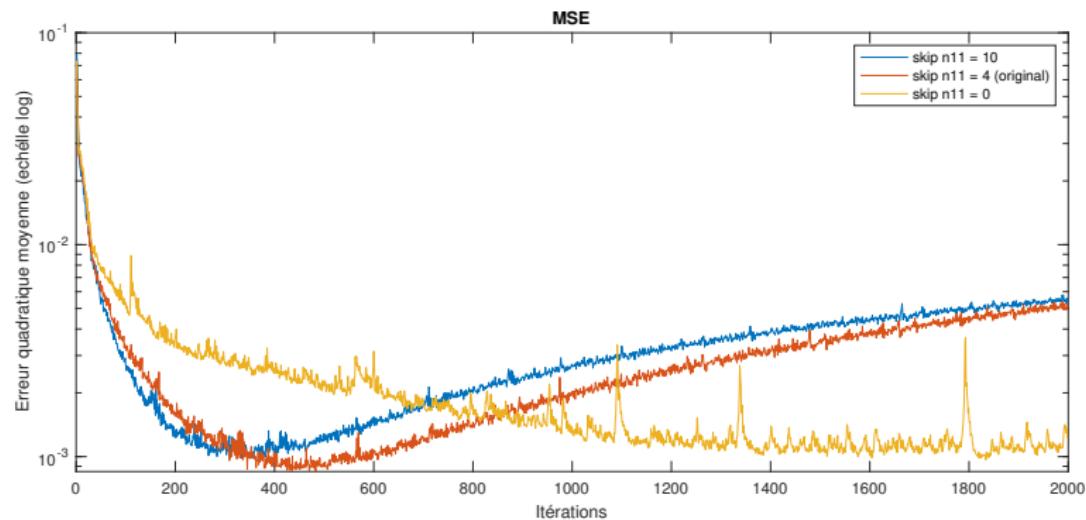


Figure – Erreur Quadratique Moyenne

# Skip connections - Résultats

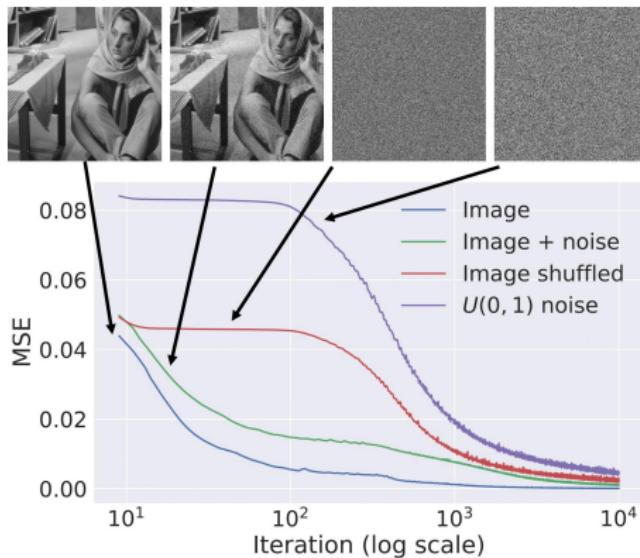


Figure – Courbes d'apprentissage pour la tâche de reconstruction en utilisant : une image naturelle, la même image plus du bruit i.i.d., la même avec les pixels mélangés aléatoirement et du bruit blanc.

# Skip connections - Résultats

## Résultats Quantitatifs

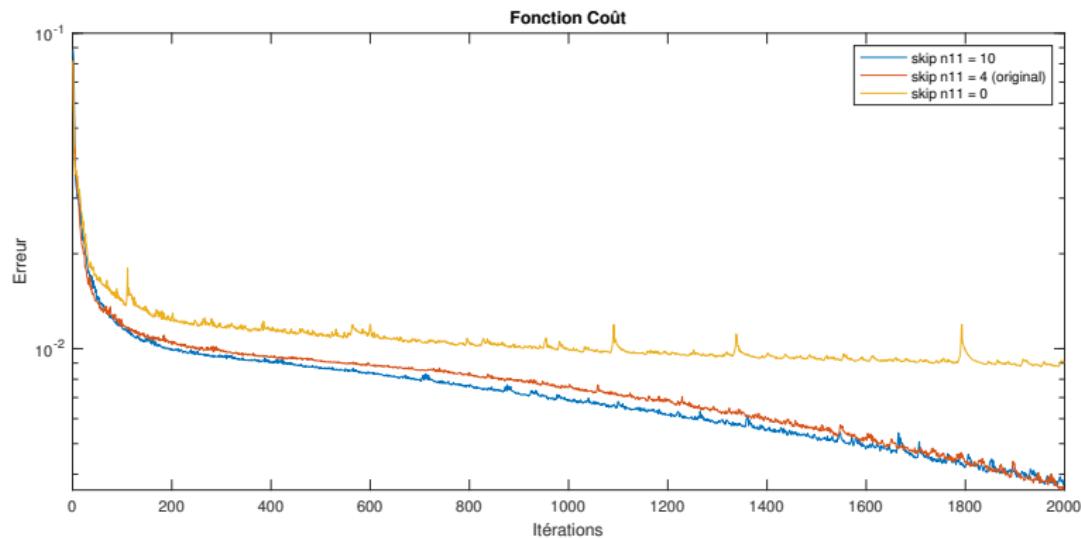


Figure – Fonction Coût

# Skip connections - Résultats

## Résultats Qualitatifs - Dernières sorties du réseau



Figure – Image final  $skip\_n11 = 0$   
PSNR = 29.129339

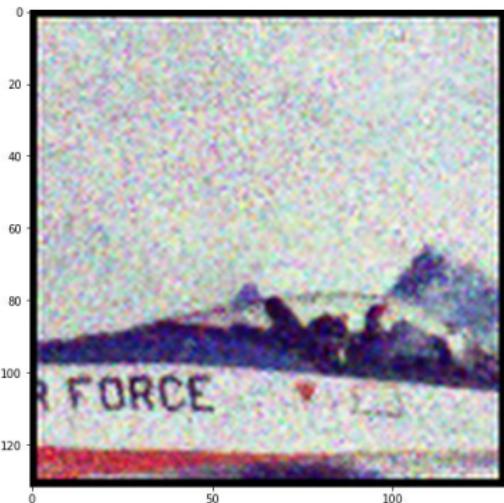


Figure – Image final  $skip\_n11 = 10$   
PSNR = 22.666066

# Performance du réseau sur du bruit réel

- La méthode est utilisée sur des images contenant du bruit
- Le bruit réel ne peut pas être modélisé par une distribution explicite



Figure – Débruitage d'images contenant du bruit non synthétique (A gauche : l'image débruitée)

# Méthode d'optimisation(1)

## Algorithme ADAM

**Soit** les taux de dégradation exponentiels pour l'estimation des moments :  $\beta_1$  et  $\beta_2$

**Soit**  $\phi$  une petite constante de stabilisation numérique

**Soit** le paramètre initial  $\theta$

Initialisation de  $m_t$  et  $v_t$

**Tant que**  $t < \text{num\_max\_itérations}$  faire

Calculer le gradient:  $\mathbf{g} = \nabla_{\theta} \Sigma E(f_{\theta}(z), x_0)$

$t=t+1$

Mise à jour des moments:  $\mathbf{m} = \beta_1 * \mathbf{m} + (1 - \beta_1) \mathbf{g}$

$\mathbf{v} = \beta_2 * \mathbf{v} + (1 - \beta_2) \mathbf{g} \cdot \mathbf{g}$

Correction des biais:  $\hat{\mathbf{m}} = \frac{\mathbf{m}}{(1 - \beta_1)}$  et  $\hat{\mathbf{v}} = \frac{\mathbf{v}}{(1 - \beta_2)}$

Calcul de mise à jour:  $\Delta \theta = \frac{-\epsilon * \hat{\mathbf{m}}}{\sqrt{(\hat{\mathbf{v}})} + \phi}$

Mise à jour des paramètres :  $\theta = \theta + \Delta \theta$

**fin tant que**

# Méthode d'optimisation(2)

## Algorithme LBFGS

Algorithme basé sur la méthode Newton :

$$J(\boldsymbol{\theta}) = J(\boldsymbol{\theta}_0) + (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) + \frac{1}{2} (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T \mathbf{H} (\boldsymbol{\theta} - \boldsymbol{\theta}_0)$$

En cherchant le point critique de cette fonction :

on obtient la règle de mise à jour :  $\boldsymbol{\theta}^o = \boldsymbol{\theta}_0 - \mathbf{H}^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$

Difficulté : Calcul de  $\mathbf{H}^{-1}$

# Méthode d'optimisation ADAM Résultats

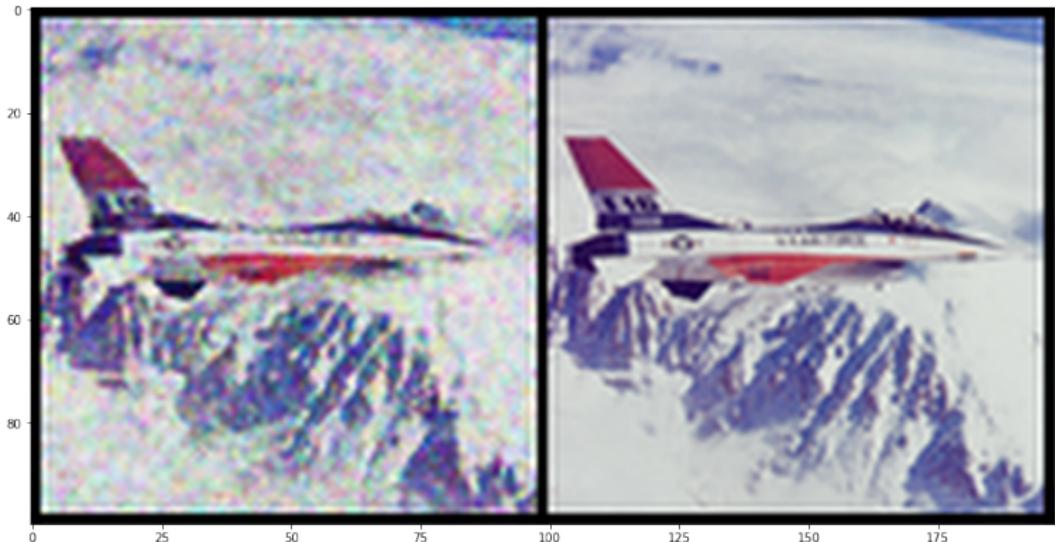


Figure – PSRN1 :24.7 PSRN2 :24.6

# Méthode d'optimisation LBFGS Résultats

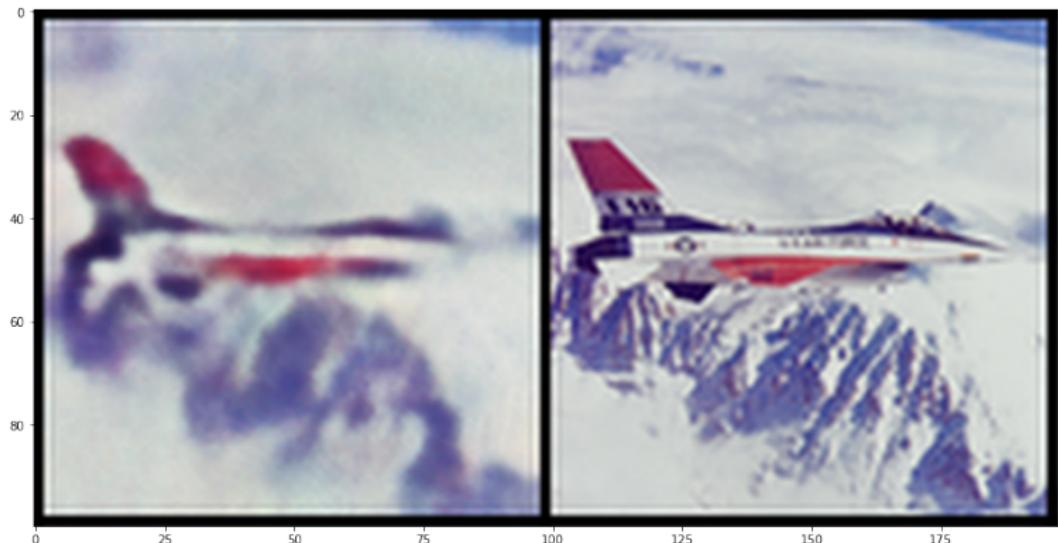


Figure – PSRN1 :22.8 PSRN2 :23

Merci de votre attention.