DEep Learning for Image Restoration and Synthesis (MVA)

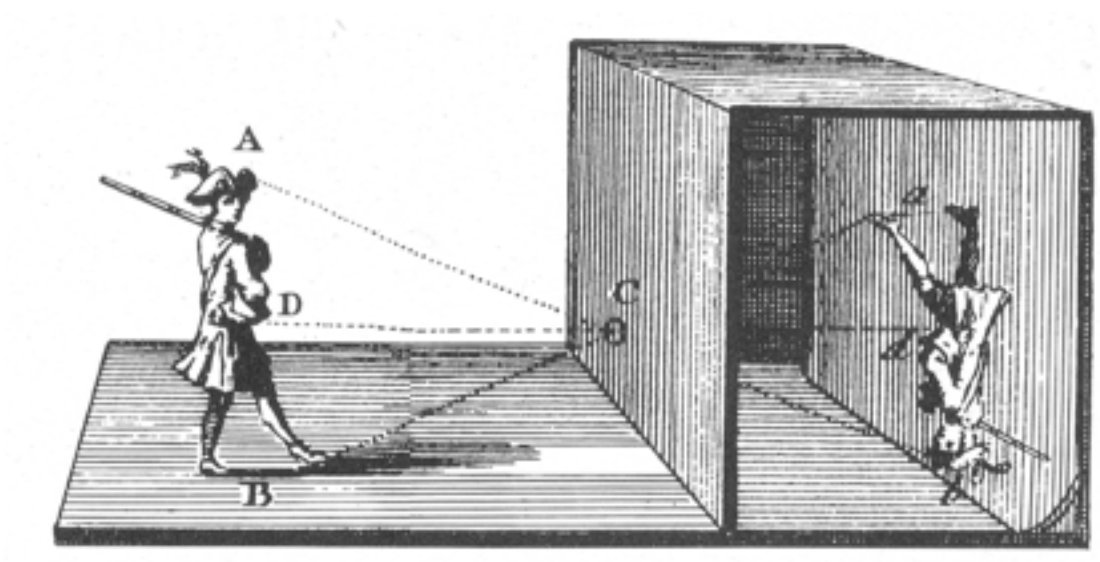
Saïd Ladjal (said.ladjal@telecom-paristech.fr)
Andrés Almansa (andres.almansa@parisdescartes.fr)
Alasdair Newson (alasdair.newson@telecom-paristech.fr)
http://delires.wp.imt.fr

Partie I: Introduction

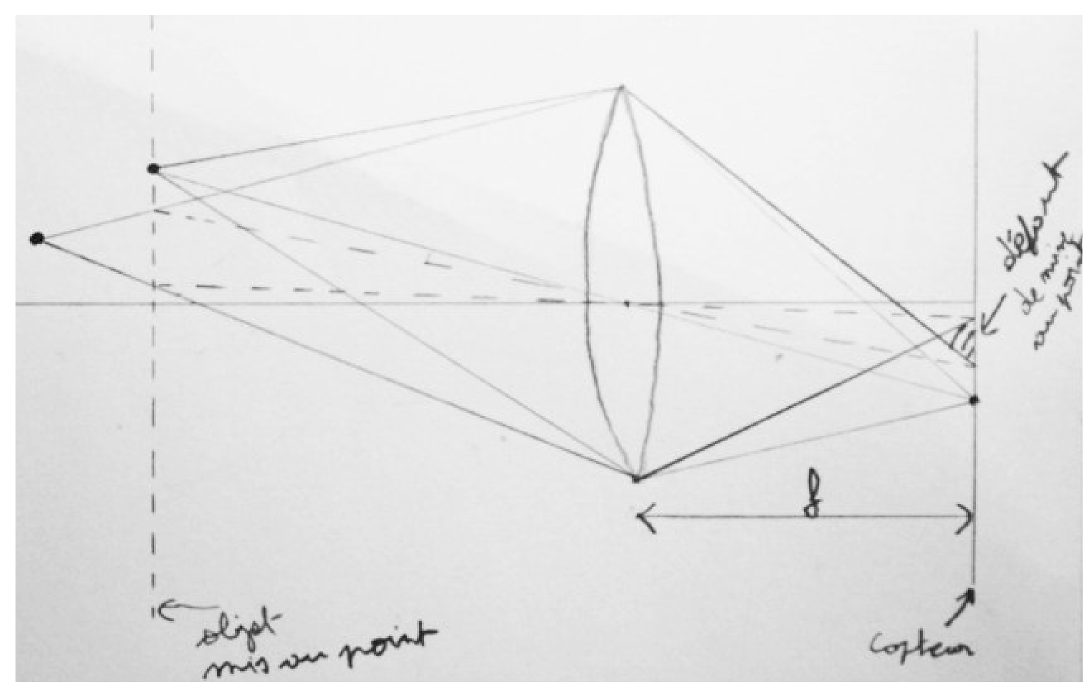
- Définition du problème de restauration.
- Quelques méthodes classiques de restauration.
- Introduction aux réseaux de neurones.
- Application aux problèmes de restauration:
  - Débruitage.
  - Super-résolution mono-image.

Partie I: La restauration

- Une image est acquise par un appareil numérique.
- L’image idéale est ce qui serait obtenu par une *camera obscura*.
- L’image réellement obtenue diffère de l’image idéale pour plusieurs raisons:
  - Flou
  - Bruit
  - Échantillonnage



Camera obscura



Lentille système optique réel

Bruit: Les raisons

- Dans une image numérique le bruit est principalement du a deux phénomènes:
  - Bruit de grenaille (shot noise): du au caractère quantique. De puissance variable.
  - Bruit thermique: De puissance constante sur l’image.

Bruit: Shot noise

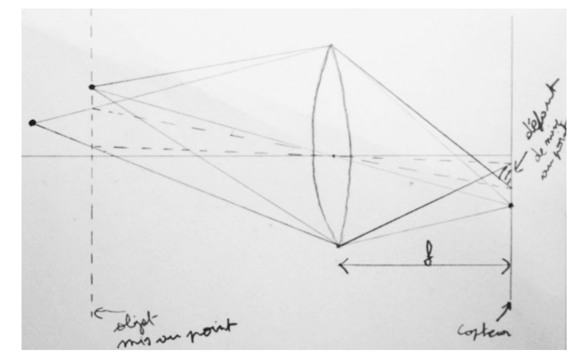
Bruit: Bruit "constant"

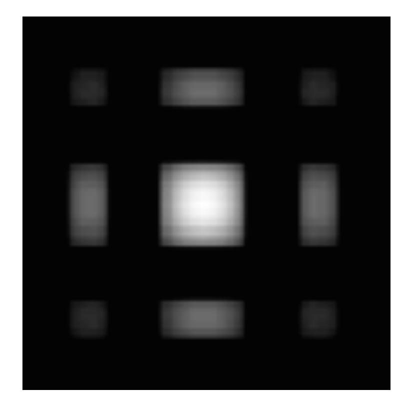
Bruit: À retenir

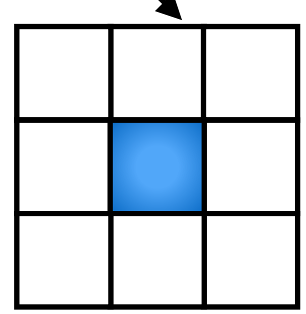
- Le modèle statistique du bruit est bien connu.
- On peut, avec une bonne précision, trouver la puissance du bruit qui affecte une image donnée.

Flou: Raisons du flou

- Le flou a plusieurs raisons possibles:
  - Défaut de mise au point.
  - Bougé.
  - Diffraction.
  - Intégration pixelique.







Flou: À retenir

- Le noyau de flou est souvent inconnu.
- Il est très difficile de retrouver le noyau de flou depuis l’image dégradée

Originale

Gaussien

Bougé

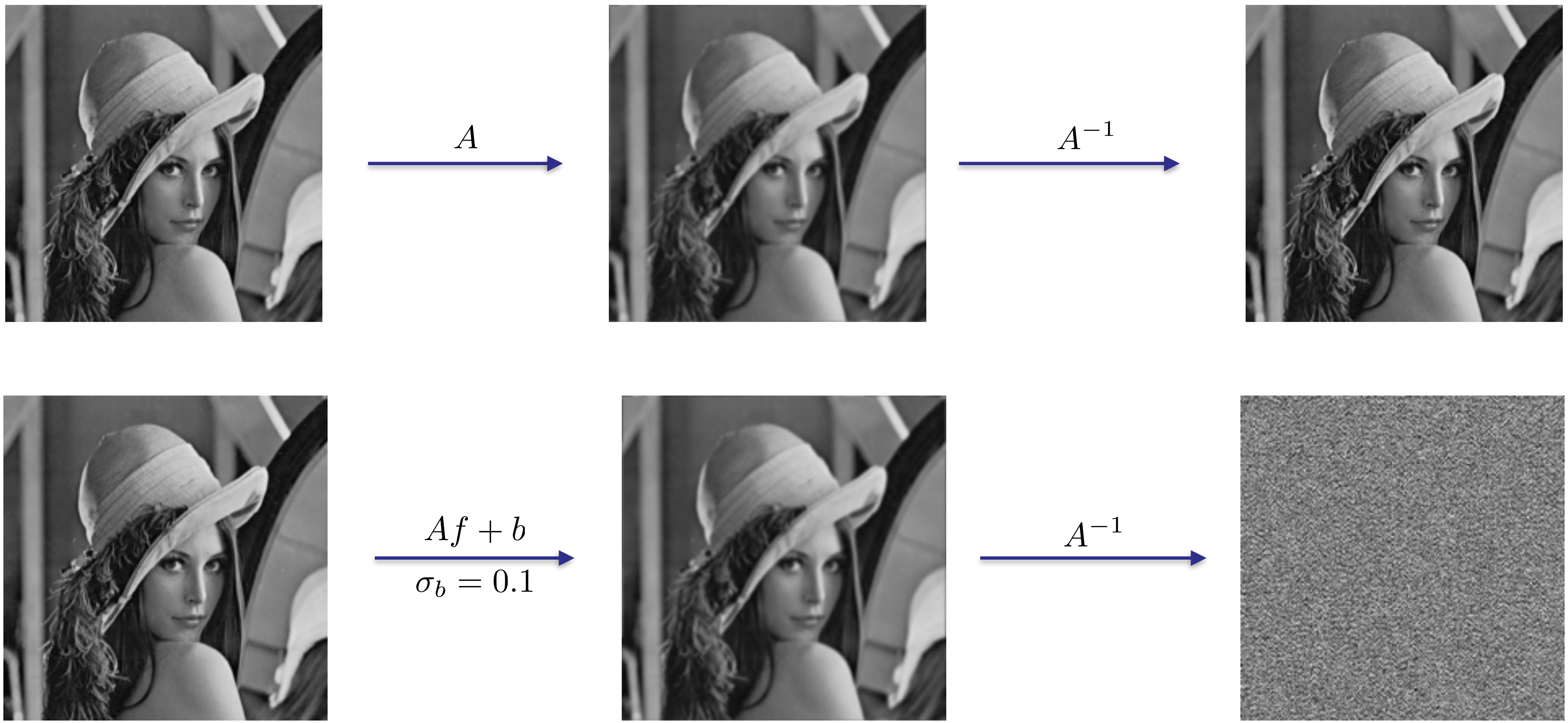
Modélisation de la dégradation

- L'image parfaite est floutée, échantillonnée puis du bruit s'ajoute à la mesure.

$g = \Pi(f_0 \star h) + b$	image acquise
f_0	image parfaite
h	noyau de flou
Π	opérateur d'échantillonnage
b	bruit

Inversion du défaut?

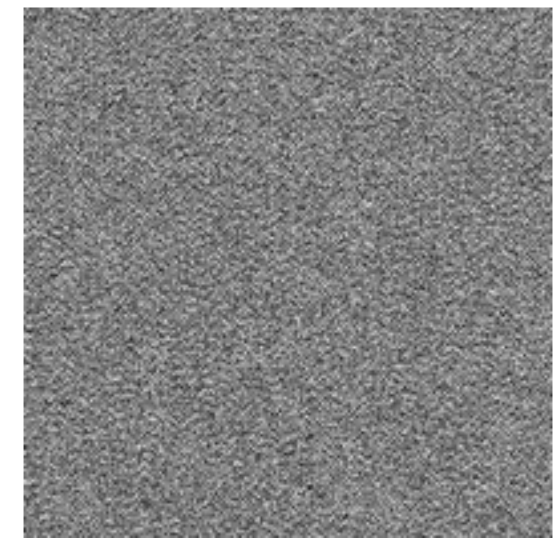
Inversion du défaut?



Inversion du défaut?

$$\xrightarrow[\sigma_b = 0.1]{Af + b}$$

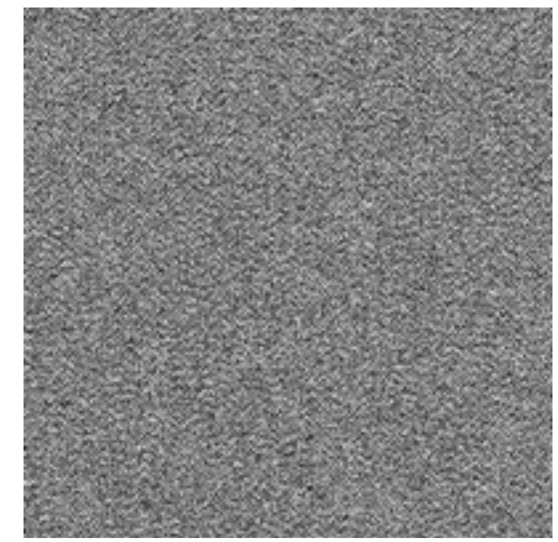
$$\xrightarrow{A^{-1}}$$



Inversion du défaut?

$$\xrightarrow[\sigma_b = 0.1]{Af + b}$$

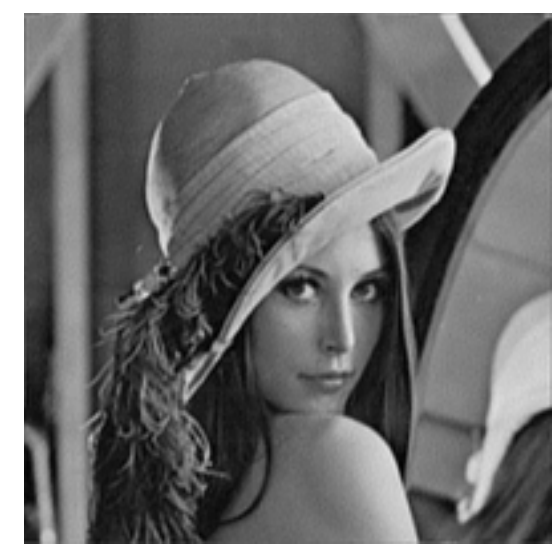
$$\xrightarrow{A^{-1}}$$



$$\xrightarrow[\sigma_b = 2]{Af + b}$$

+Image model

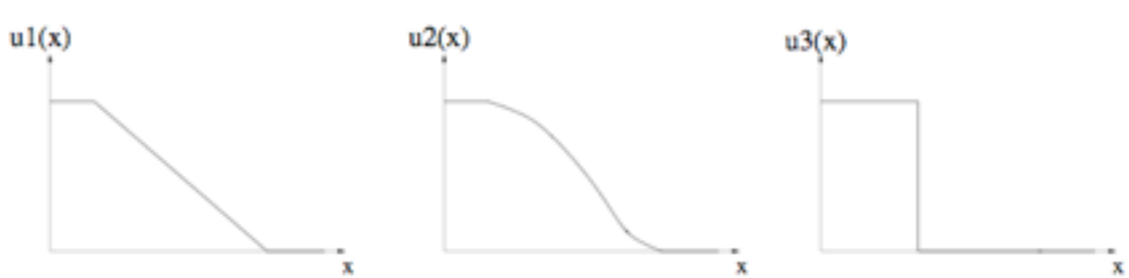
$$\xrightarrow[\mathcal{P}(Af + b)]{}$$



Modèles: La régularisation

- On veut reconstruire une image qui:
  - Explique l'observation
  - Est régulière
- Pour se faire on recherche l'image comme minimiseur de:

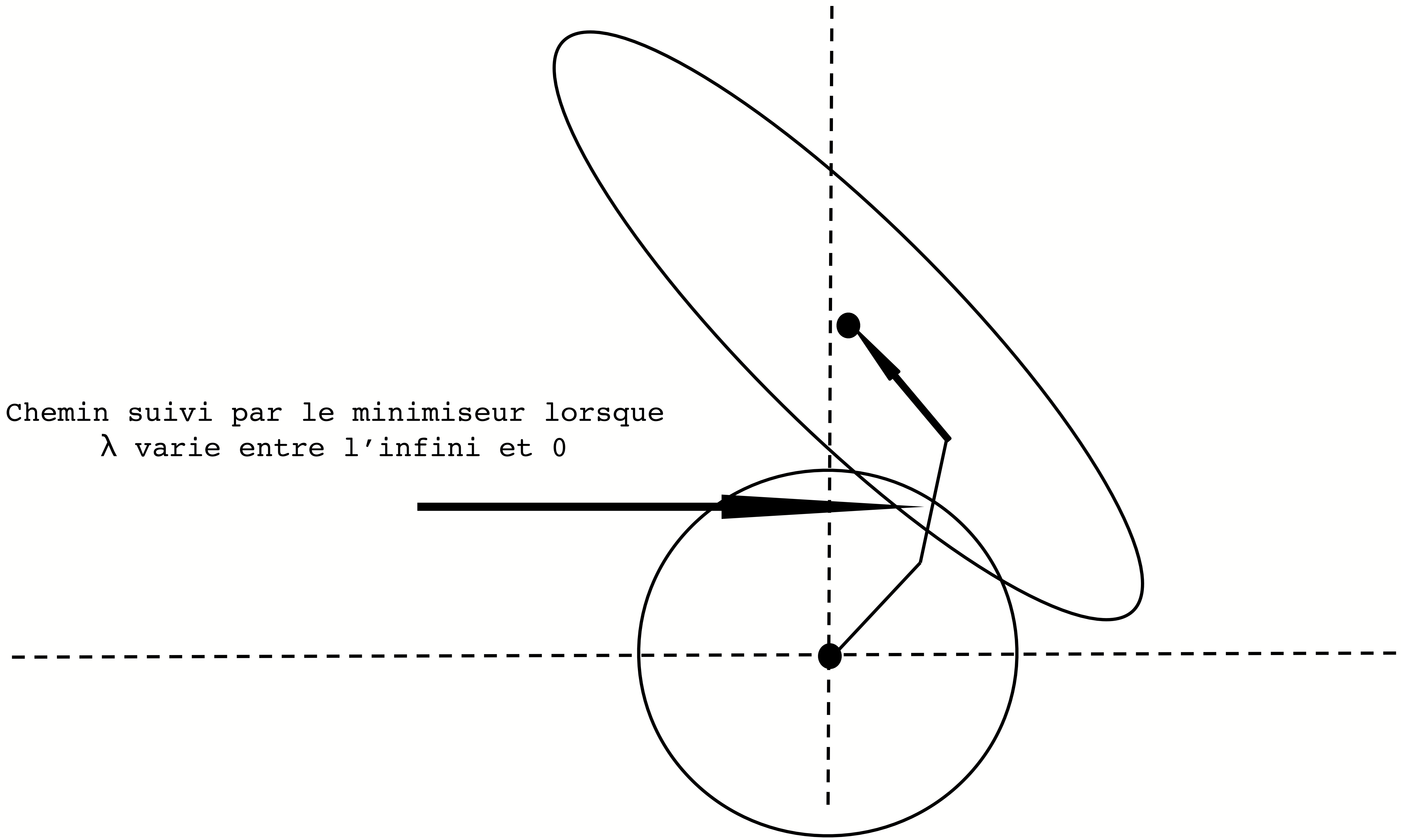
$$E(u) = \|Au - g\|^2 + \lambda R(u) \quad \begin{array}{l} R(u) = \iint \|\nabla u\|^2 \text{ Tychonov} \\ R(u) = \iint \|\nabla u\| \text{ (TV)} \end{array}$$



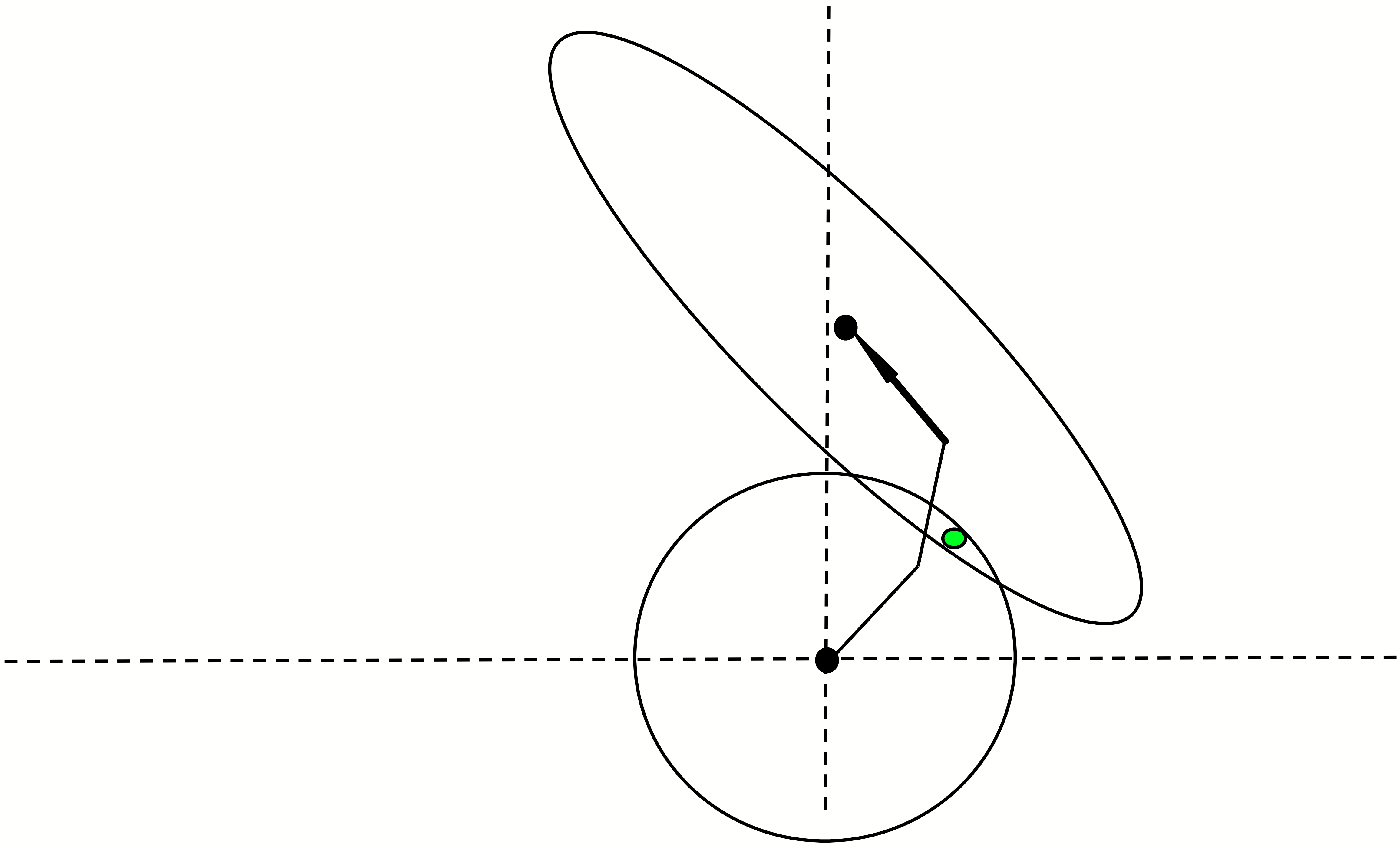
Régularité Thyconov

Régularité TV

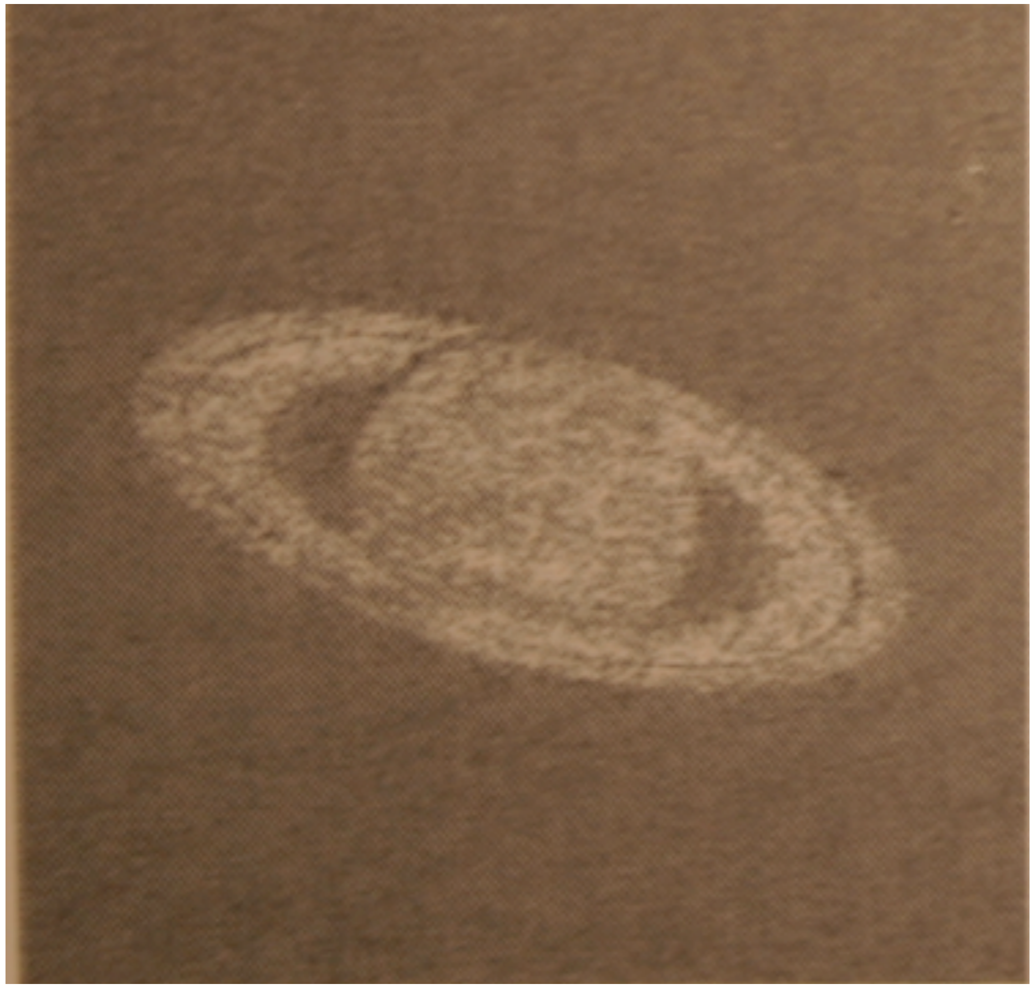
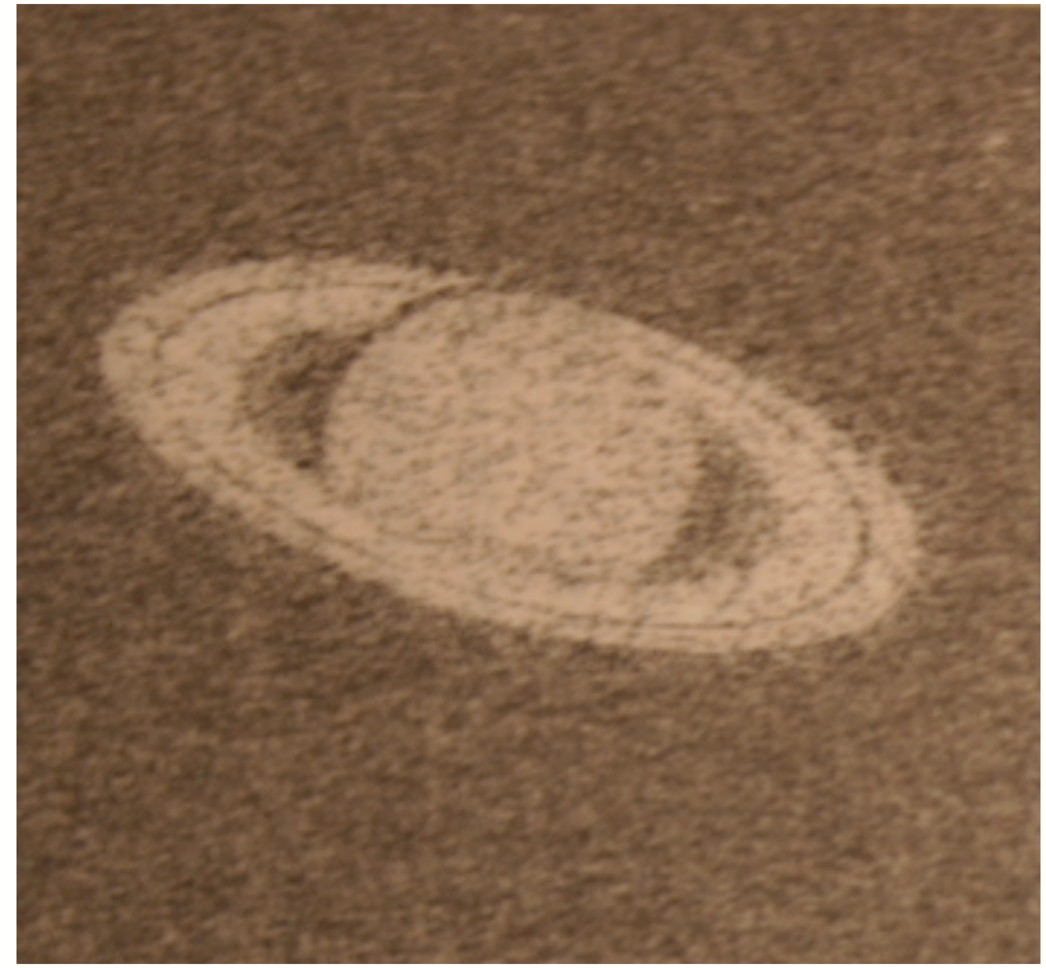
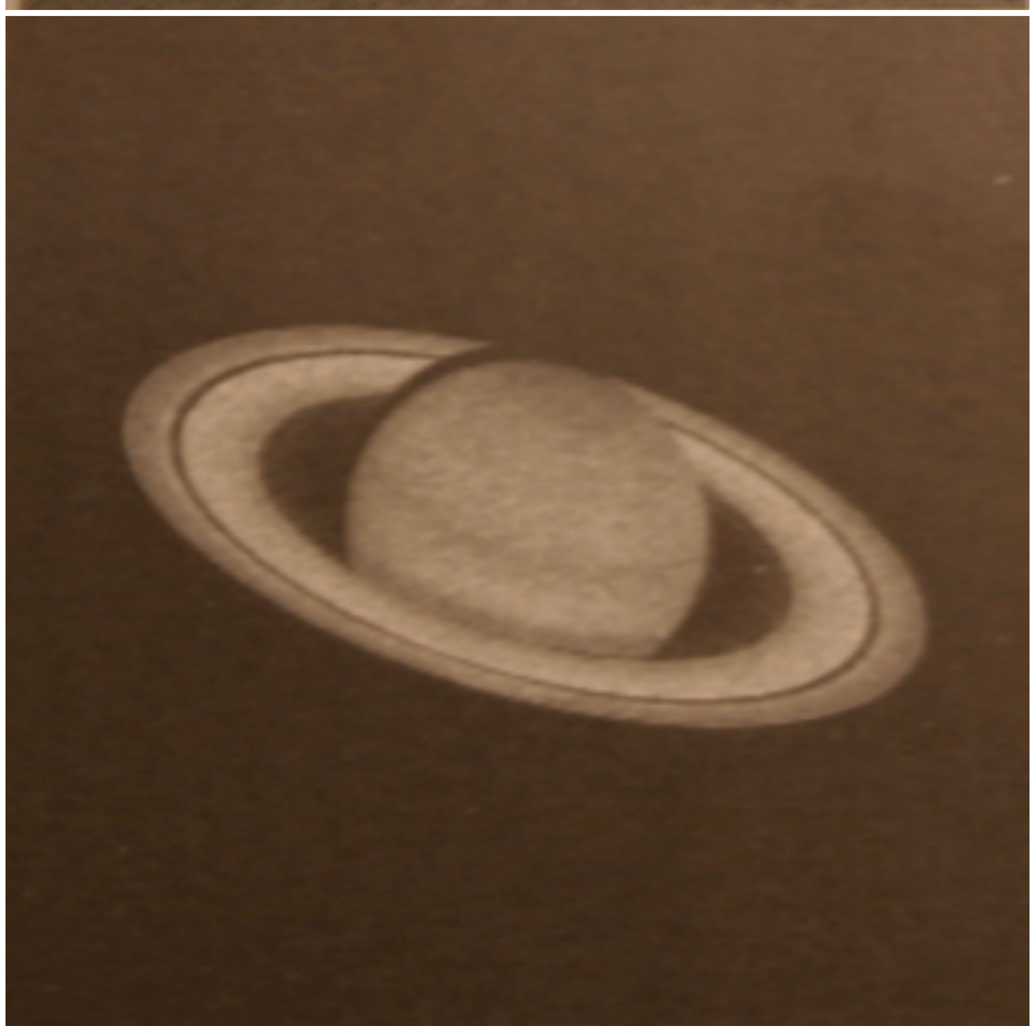
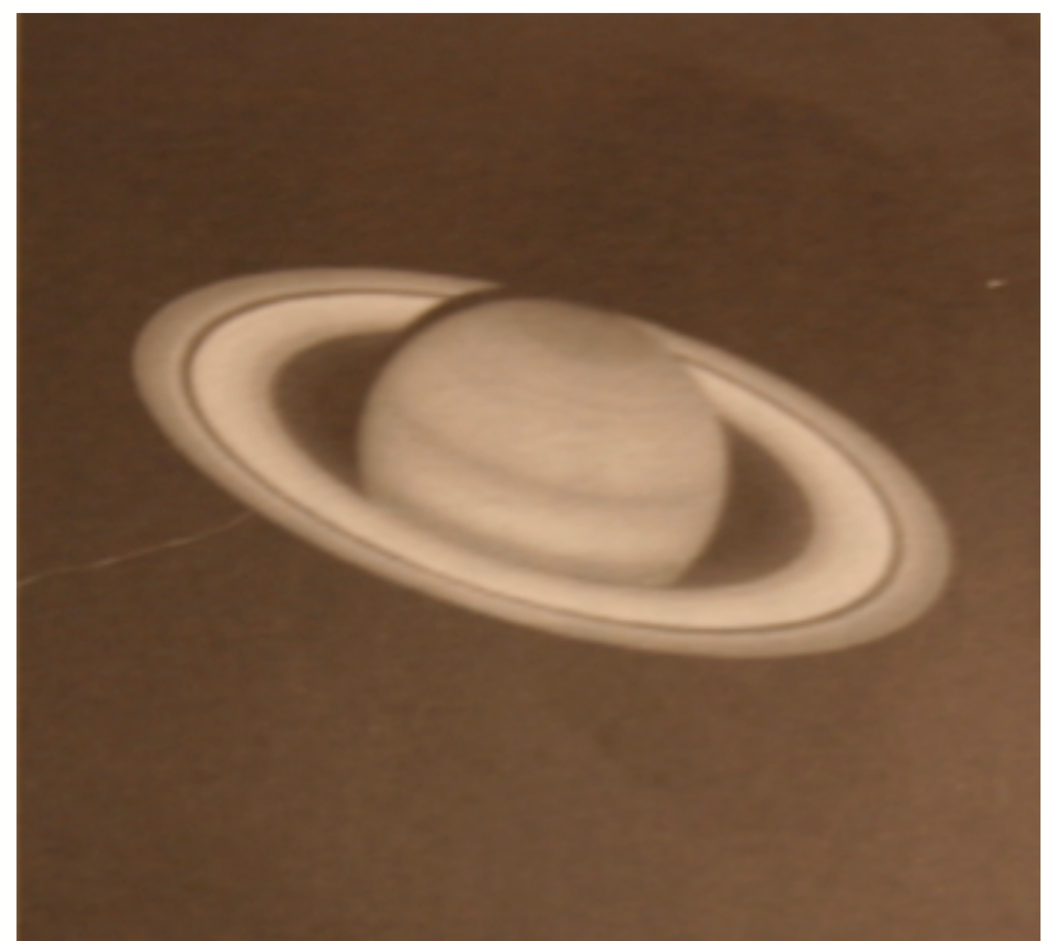
Modèles: La régularisation

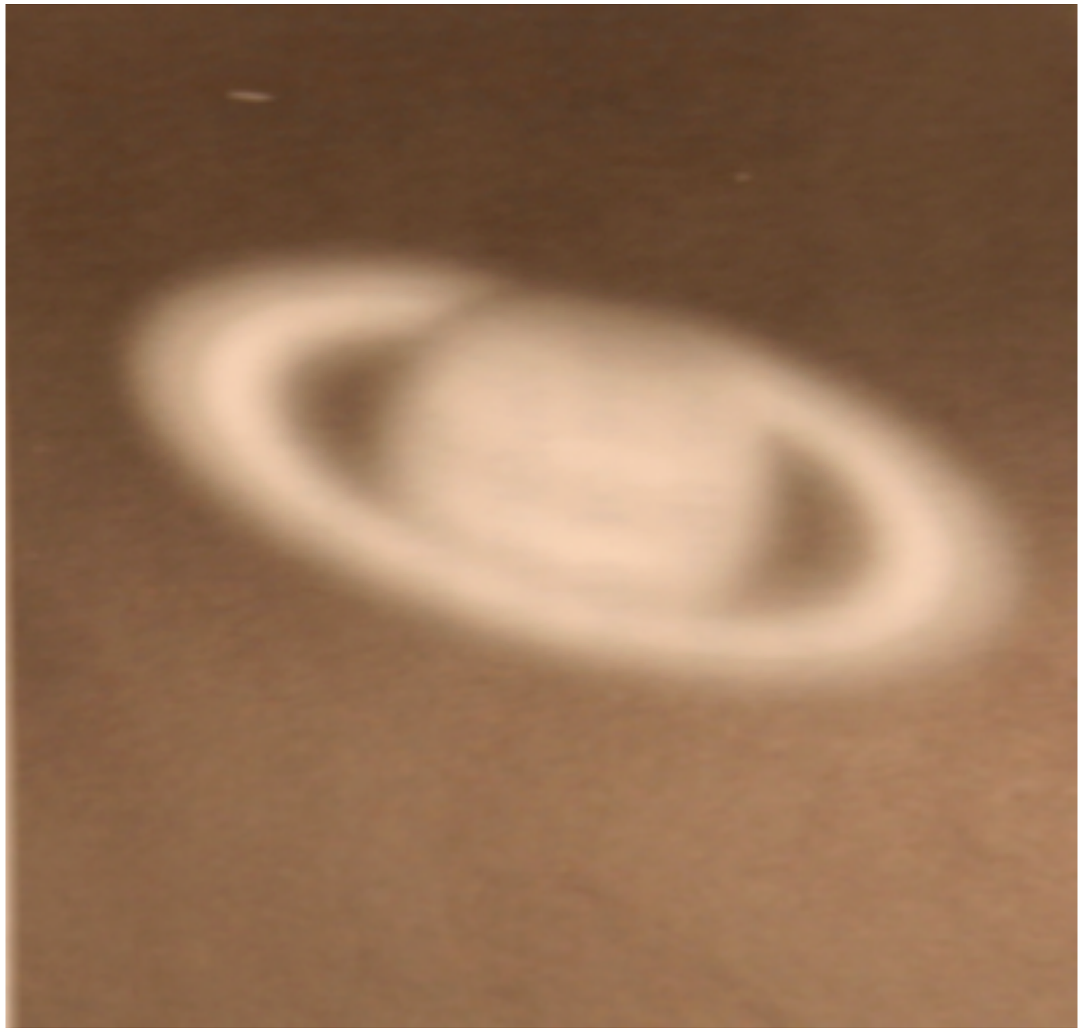


Modèles: La régularisation



La solution s'approche puis s'éloigne de la vraie image.



Modèles: Statistiques

Modèle Gaussien:

X v.a. $\in \mathbb{R}^N$
$\mathbb{P}(X) \sim e^{-\frac{1}{2} X^T C^{-1} X}$
$C = E(X^T X)$ matrice de covariance déf. pos.
$\mathbb{P}(B) \sim e^{-\frac{1}{2}\frac{\|B\|^2}{\sigma_b^2}}$

Problème:

$Y = AX + B$
trouver $\tilde{X} = DY$ t.q.
$E(\|\tilde{X} - X\|^2)$ soit minimale

Solution:

$$\boxed{\tilde{X} = \underbrace{\left(A^T A + \sigma_b^2 C^{-1}\right)^{-1} A^T}_{\text{D}} Y}$$

Modèles: Wiener

- Pour les images, une base de décomposition naturelle: Fourier

- La densité spectrale de puissance:
  - On peut l'imposer comme a priori.
  - Ou utiliser celle de l'image dégradée.

$$\boxed{\tilde{X} = \underbrace{\left(A^T A + \sigma_b^2 C^{-1}\right)^{-1} A^T}_{\text{D}} Y}$$

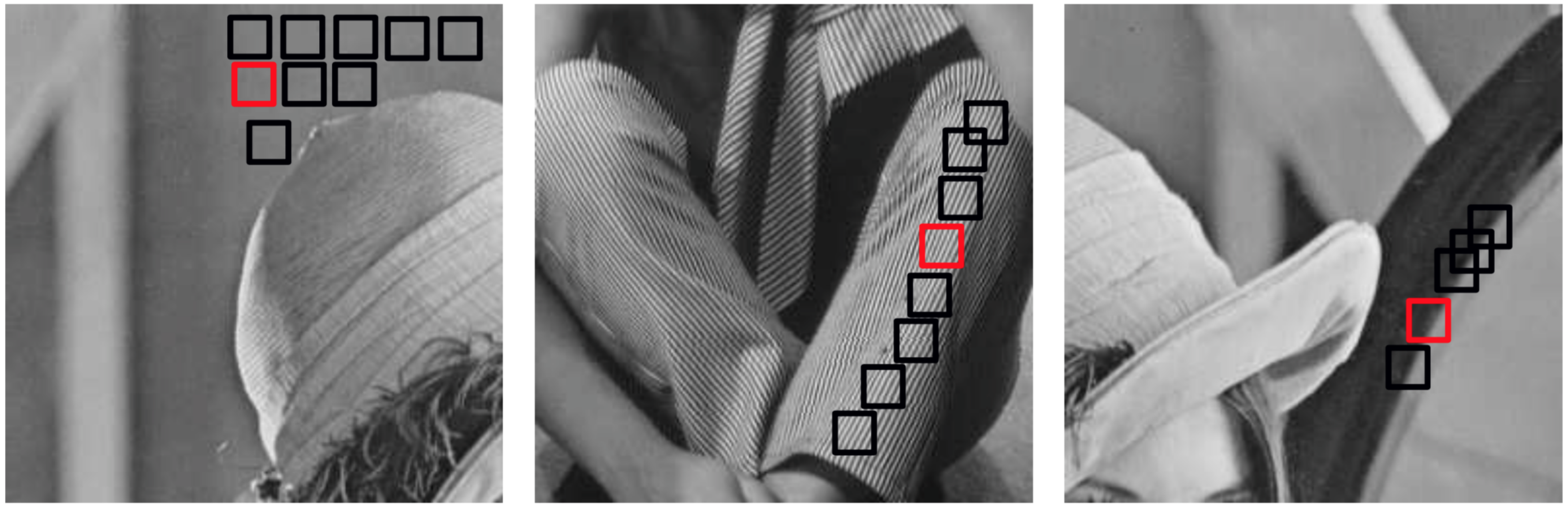
$g = Af + b$
$f \in \mathbb{R}^{N^2}$: image de taille $N \times N$ inconnue
$g \in \mathbb{R}^{N^2}$: image de taille $N \times N$ connue
A : matrice de convolution carrée de taille $N^2 \times N^2$
$b \in \mathbb{R}^{N^2}$: réalisation d'un bruit gaussien

$$\hat{\tilde{f}}(\omega) = \frac{\overline{\hat{K}(\omega)}}{|\hat{K}(\omega)|^2 + \frac{\sigma_b^2}{\sigma_s^2(\omega)}} \hat{g}(\omega)$$

ω parcourt les fréquences de Fourier
$\sigma_s^2(\omega)$ puiss. du signal à la fréq. ω

K : noyau de convolution $Af \hat{=} K * f$

Modèles: Auto-similarité(NLM, BM3D)



Région uniforme — Région texturée — Contour géométrique

$$nouveau(p) = \sum_q ancien(q) \frac{D(V(p),V(q))}{\sum_q D(V(q),V(p))}$$

$V(n)$ voisinage du pixel n

$$D(V1,V2) = e^{-\frac{\|V1-V2\|_2^2}{h^2}}$$

Essayez vous-même

http://demo.ipol.im/demo/bcm_non_local_means_denoising/

http://demo.ipol.im/demo/l_bm3d/

CNN: Résumé

- Définir l'architecture
- Fonction objectif (fonction à minimiser, Loss)
- Comment procéder à l'apprentissage (comment minimiser la Loss)
  - Études de quelques réseaux proposés pour la restauration

CNN: Architecture

- L'image est traitée par le CNN comme une fonction à valeurs vectorielles

  - En entrée une image couleur (profondeur=3): Pour toute position on a un vecteur de R^3 (gris -> R^1)

  - Puis cette image est transformée en une autre image, a priori, de même taille mais avec une profondeur différente.

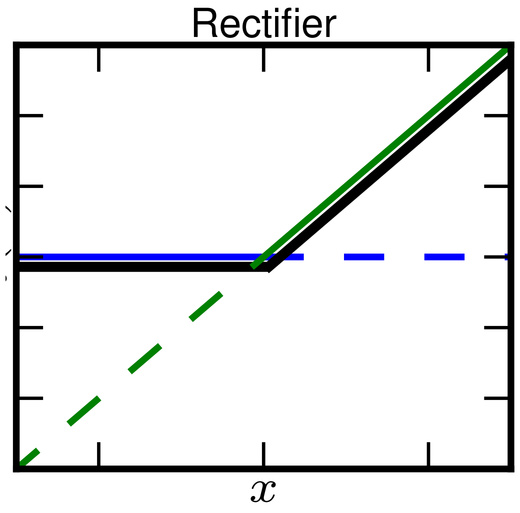
  - Chaque canal de la nouvelle image est obtenu par:

$I_k^{l+1} = a(I^l * W_k^l + b_k)$
I^l image à l'étape l
W_k^l filtre numéro k de la couche l
$*$ est l'opération de convolution où le produit est un produit scalaire
a est la fonction (scalaire) d'activation
b_k est le biais

CNN: Architecture
Les autres composantes

- ReLu: Rectified linear Unit.
  - C'est la non-linéarité d'activation la plus utilisée dans les CNN.
  - Par rapport à la sigmoïde ou tangente hyperbolique elle a l'avantage de ne pas saturer des deux cotés.

$$R(x) = \begin{cases} 0 \text{ si } x < 0 \\ x \text{ sinon} \end{cases}$$

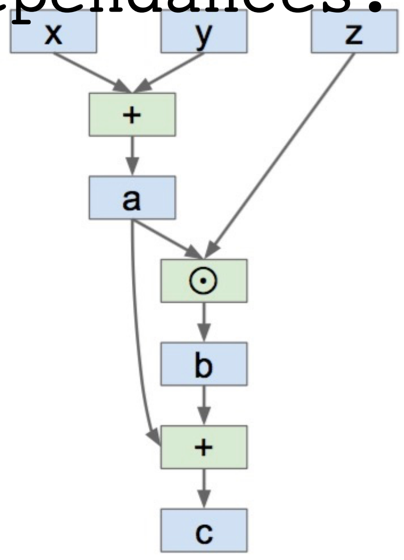


CNN: Architecture
Graphe de calcul

- On regarde l'activation de chaque neurone comme un scalaire.

- L'activation d'un neurone dépend de l'activation des autres via une fonction.

- Tout le réseaux peut être regardé comme un graphe orienté dont les noeuds sont les opérations et les arêtes sont les dépendances.



$$c = b + a = (az) + a = ((x+y)z) + (x+y)$$

CNN: Architecture
Les autres composantes

- MaxPool: Il s'agit de prendre le maximum d'une région comme valeur du nouveau neurone et dans le même temps de sous-échantillonner.

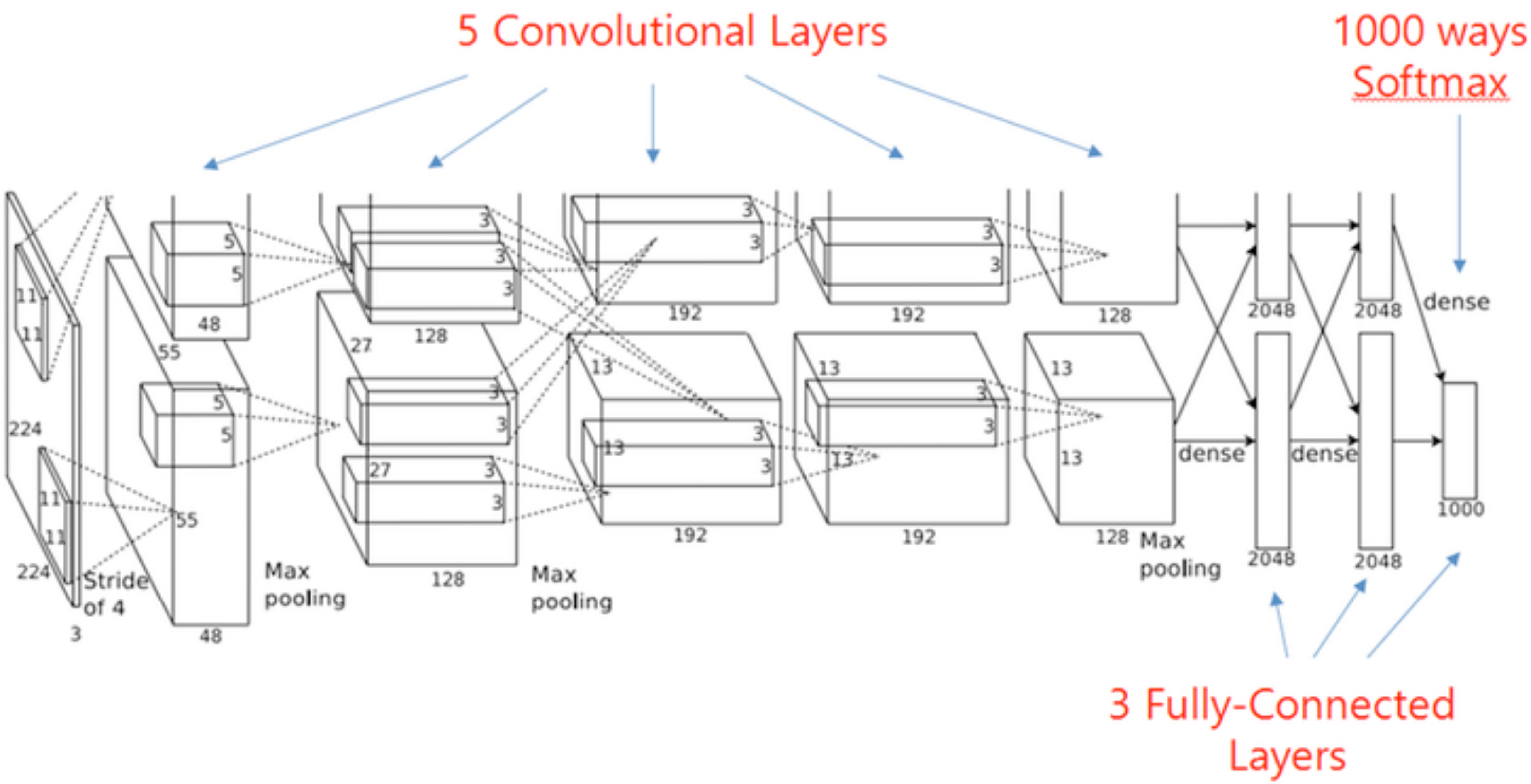
  - Permet de réduire la taille du flux de données à travers le réseau…

  - Tout en gardant la partie la plus significative.

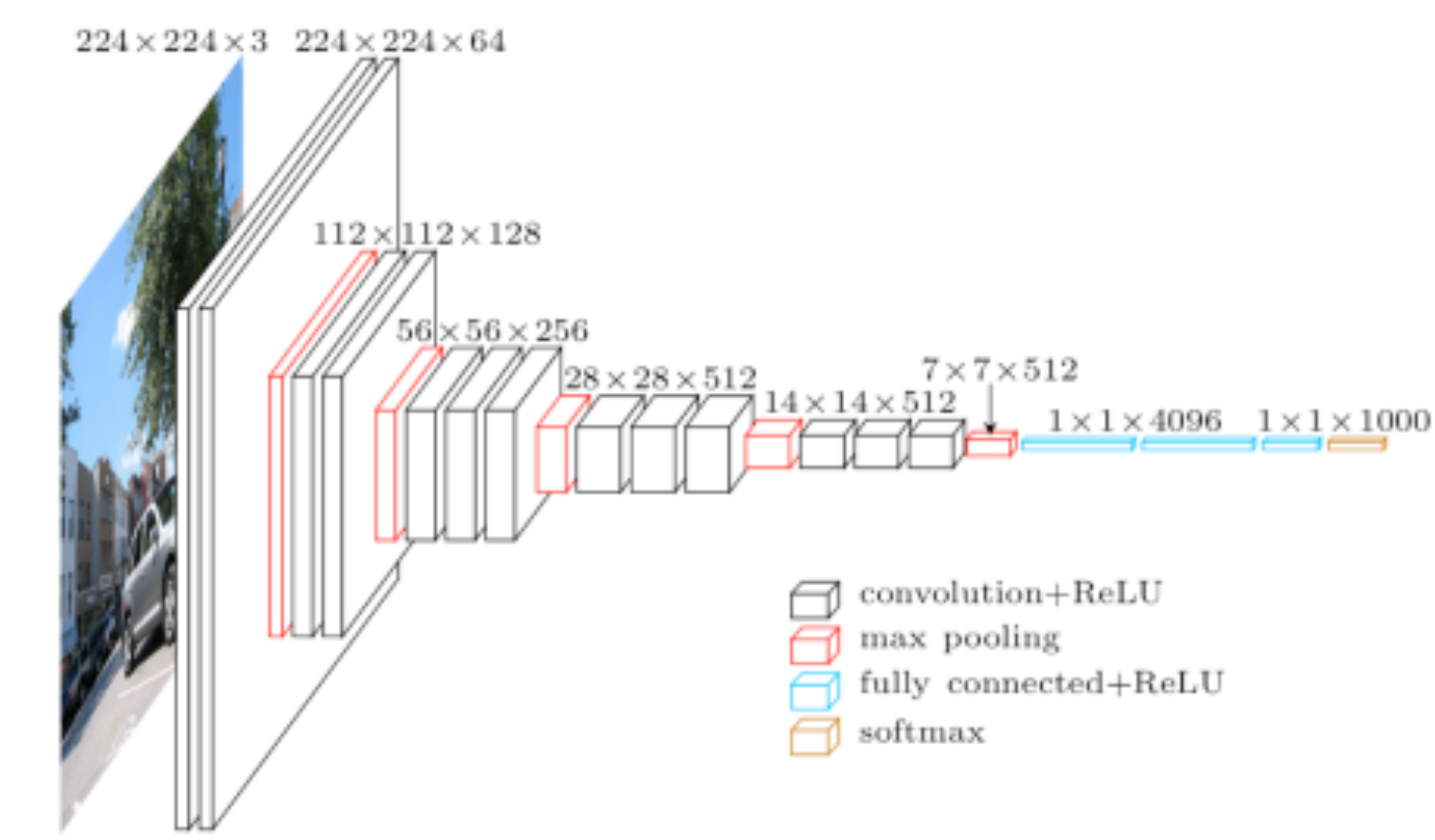
  - D'autres formes de sous-échantillonnage sont possibles.

  - Rappel les filtres morphologiques et participe au comportement non linéaire.

CNN: Exemples
AlexNet



CNN: Exemples
VGG-16



500Mo sur le disque dur

CNN: Fonction objectif (loss)

- Suivant la tâche à accomplir on choisit la fonction objectif.
- Le but de l’entraînement est de minimiser cette fonction objectif
  - Pour la classification softmax_logit
  - Pour la restauration L^2

CNN: Fonction objectif (loss)

$$L(I) = \|I^l\|_2^2$$

CNN: Fonction objectif (loss)

- La base de données: $(f_n, g_n = Af_n + b)$
- L'architecture du réseau: $\mathcal{C}(g, W)$
- La loss: $\mathcal{L}(W) = \sum_n \|\mathcal{C}(g_n, W) - f_n\|^2$

Rappel du modèle statistique: $\underset{\mathcal{P}}{argmin}\, \mathbb{E}\left(\|\mathcal{P}(g) - f\|^2\right)$

CNN: Fonction objectif (loss) exemple en débruitage

- Si on veut construire un réseau qui débruite les images:
  - On se donne une base de données d'images parfaites fn
  - On ajoute du bruit et pour obtenir un image gn
  - On veut retrouver les fn à partir des gn:

$$\mathcal{L}(W) = \sum_n \|\mathcal{C}(g_n, W) - f_n\|^2$$

CNN: Fonction objectif (loss) exemple en restauration

- Si on veut construire un réseau qui interpole les images:
  - On se donne une base de données d'images parfaites fn
  - Chaque image est sous-échantillonnée en une image gn
  - On veut retrouver fn à partir de gn, par exemple:

$$\mathcal{L}(W) = \sum_n \|\mathcal{C}(g_n, W) - f_n\|^2$$

CNN: Fonction objectif

- Comment la minimiser? On cherche à minimiser par morceaux (gradient stochastique):
  - mini-batch: Un sous ensemble de la base d'apprentissage donne une fonction.
  - La somme de ces fonctions donne la fonction objectif (loss)
  - Gradient stochastique: En minimisant alternativement chacune de ces fonctions on espère minimiser la somme (la fonction objectif)

$$\mathcal{L}(W) = \sum_n F_n(W) = \sum_m F_{J_m}(W), \text{ avec } F_{J_m} = \sum_{n \in J_m} F_n(W) \text{ et } \cup_m J_m = [1, N]$$

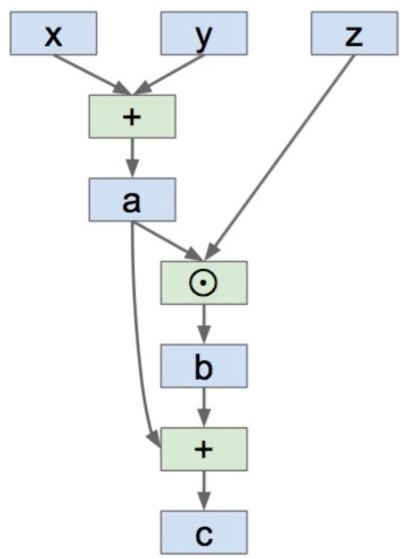
CNN: Entraînement
Le gradient (rappels)

$$L(\Theta + d\Theta) \approx L(\Theta) + DL(\Theta)(d\Theta) = L(\Theta) + < \nabla L | (d\Theta) >$$

La direction dans laquelle **L** croît le plus vite est le gradient

$$\mathrm{argmax}_v \frac{|L(\Theta + tv) - L(\Theta)|}{t} = \nabla L(\Theta)$$

CNN: Entraînement Rétropropagation (backprop)

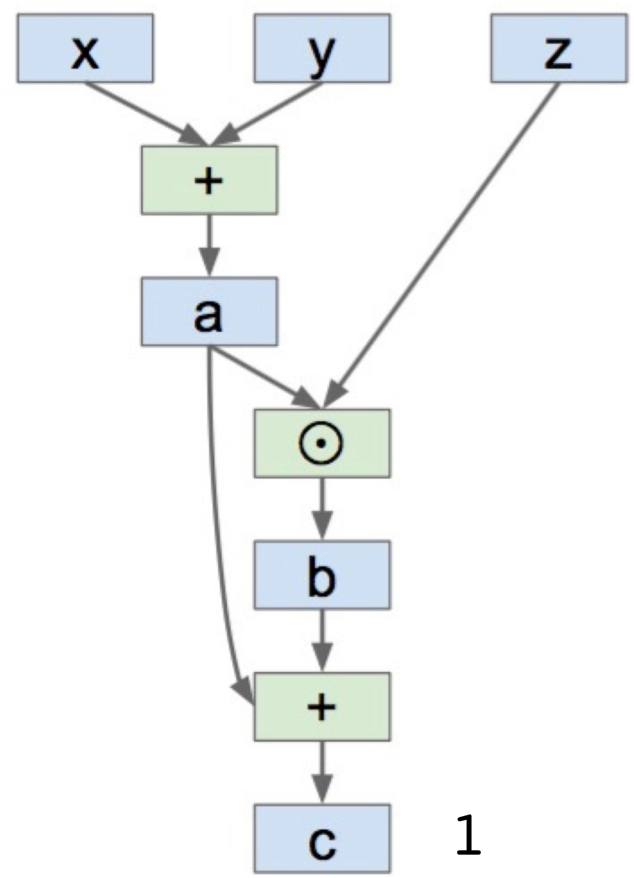


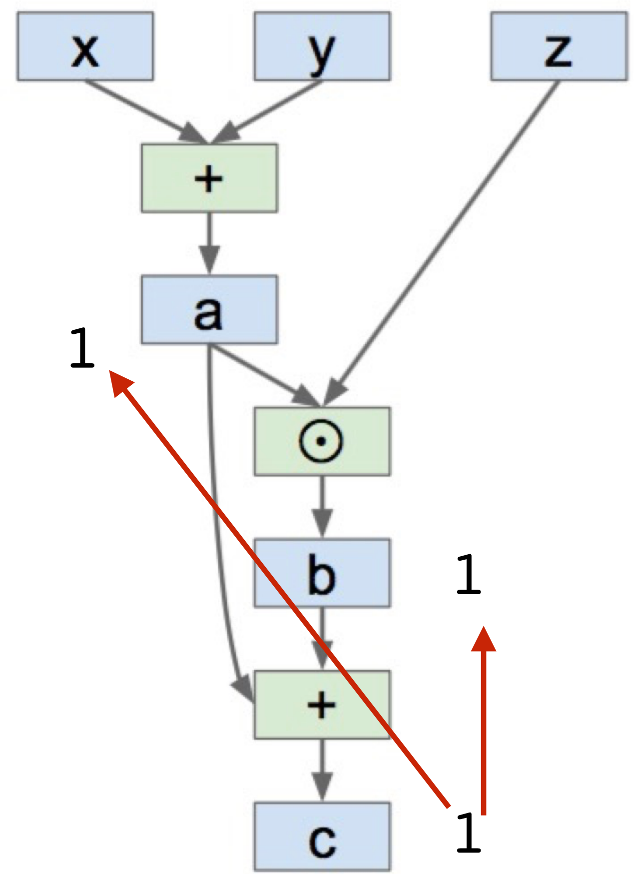
$$c = b + a = (az) + a = ((x+y)z) + (x+y)$$

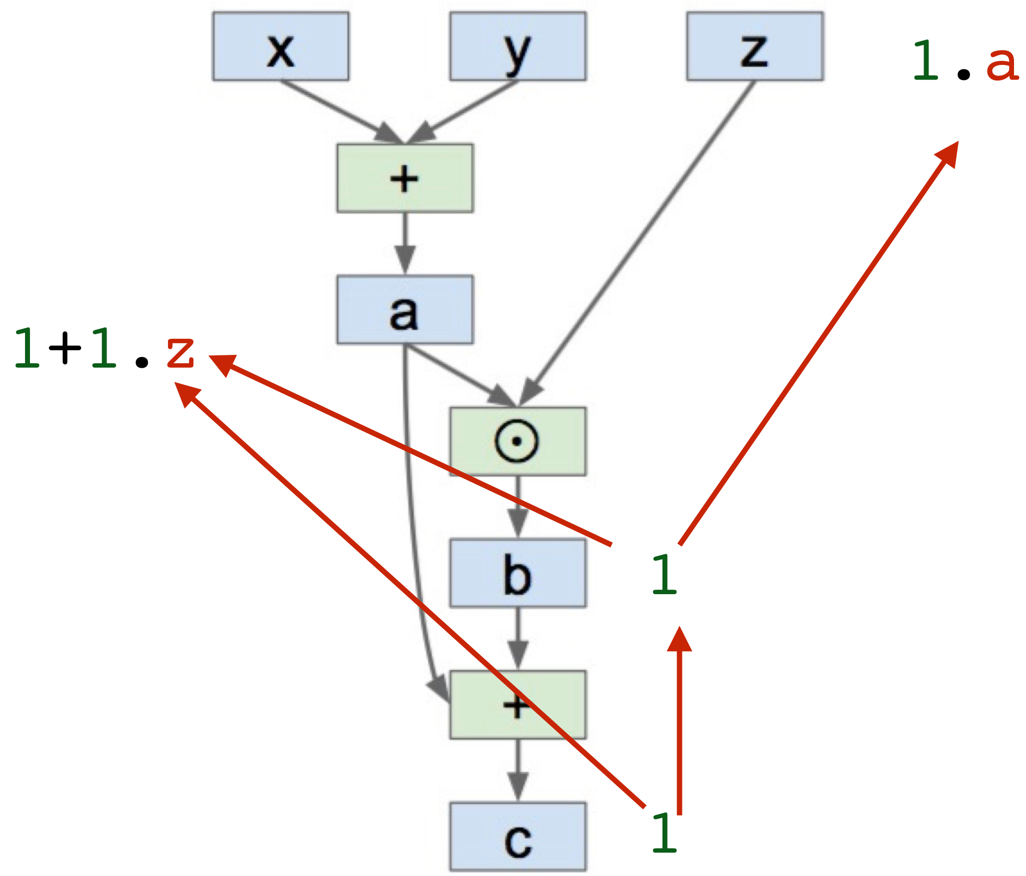
- Backpropagation: On cherche à calculer les variations de la loss par rapport aux variables du réseau (gradient)

$$\Delta c = \Delta b + \Delta a$$
$$\Delta b = \Delta a.z + \Delta z.a$$
$$\Delta a = \Delta x + \Delta z$$
$$\frac{\partial c}{\partial z} = a$$

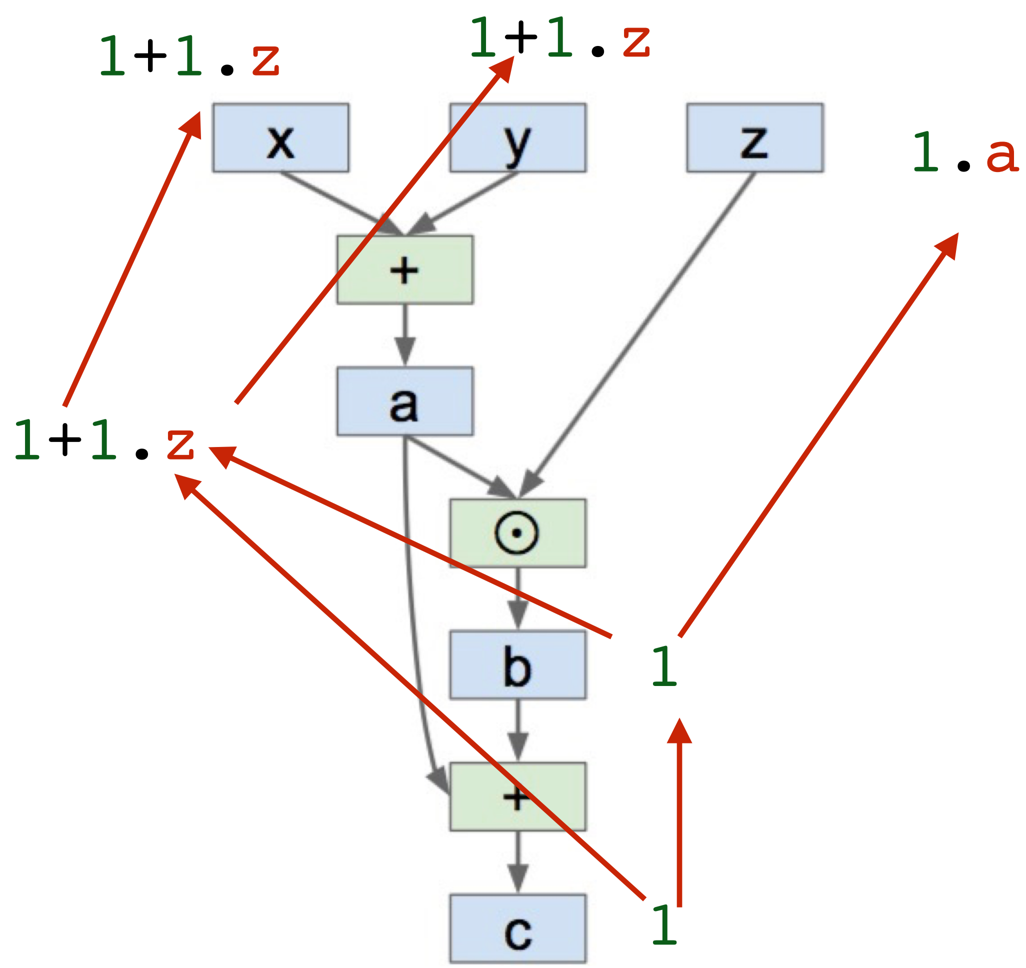
- La loss varie comme 1 fois elle-même.
- Elle envoie le message vous "je varie de tant en fonction de vous aux noeuds dont elle dépend".
- Arrivé à un noeud final, on a le gradient!


x
y
z
+
a
⊙
b
+
c
1


x
y
z
+
a
1
⊙
b
1
+
c
1


x
y
z
1.a
+
a
1+1.z
⊙
b
1
+
c
1

$$c = b + a = (az) + a = ((x+y)z) + (x+y)$$



$$\frac{\partial c}{\partial z} = a = x + y$$

$$\frac{\partial c}{\partial x} = 1 + z$$

Débruitage: rappel

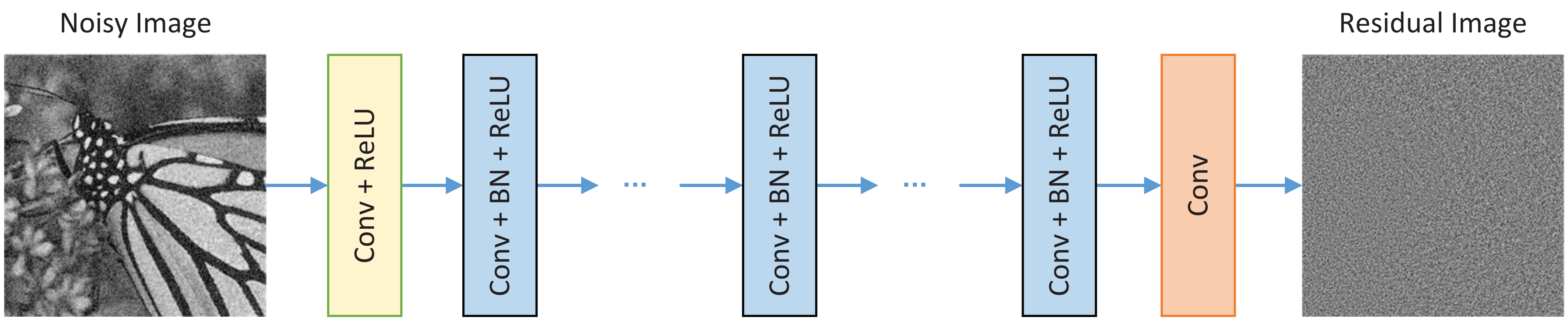
- Si on veut construire un réseau qui débruite les images:
  - On se donne une base de données d'images parfaites fn
  - On ajoute du bruit et pour obtenir un image gn
  - On veut retrouver les fn à partir des gn:

$$\mathcal{L}(W) = \sum_n \|\mathcal{C}(g_n, W) - f_n\|^2$$

Débruitage: Points clés

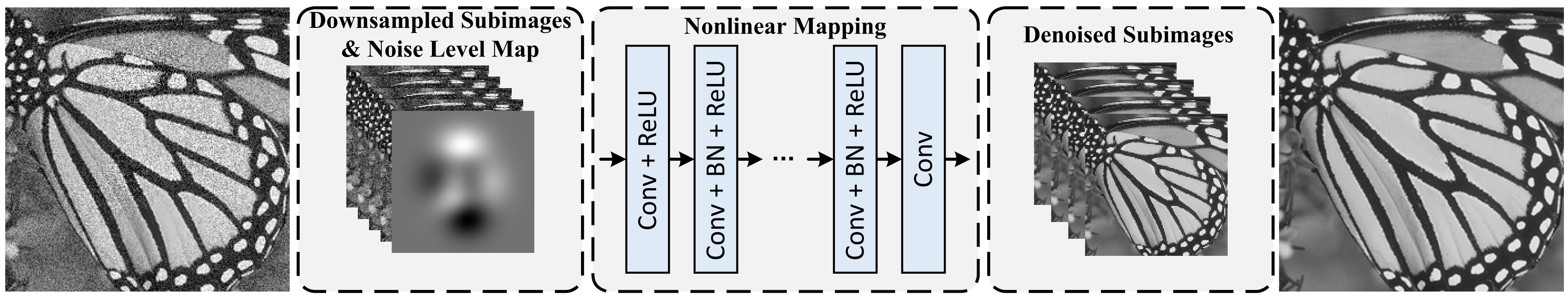
- Dans le cas du débruitage on sait parfaitement synthétiser le bruit.
- On peut donc construire une base de données qui échantillonne bien la densité de probabilité du couple (f,g). (à condition de bien choisir les f)

Débruitage: DnCNN



- Il y a une vingtaines de couches.
- On apprend non pas l'image restaurée mais le bruit.
- La batch normalisation (astuce de minimisation) est utilisée à toutes les couches internes.

Débruitage: FFDNet (mêmes auteurs que DnCNN)



- Il y a une 15 couches.
- On reconstruit l'image et non pas le résidu.
- L'image est fournie comme 4 sous-images+estimations du bruit.
- La batch normalisation (astuce de minimisation) est utilisée à toutes les couches internes.

Débruitage: Niveau de bruit

- A priori, il faut apprendre pour un niveau de bruit fixé.
- Dans une méthode variationnelle, l'inclusion du niveau de bruit est souvent simple.
- Dans FFDNet les auteurs incluent un pré-calcul du bruit comme entrée du réseau.

Rappel Wiener:

$$\hat{\tilde{f}}(\omega) = \frac{\overline{\hat{K}(\omega)}}{|\hat{K}(\omega)|^2 + \frac{\sigma_b^2}{\sigma_s^2(\omega)}} \hat{g}(\omega)$$

ω parcourt les fréquences de Fourier
$\sigma_s^2(\omega)$ puiss. du signal à la fréq. ω

Super-résolution: rappel

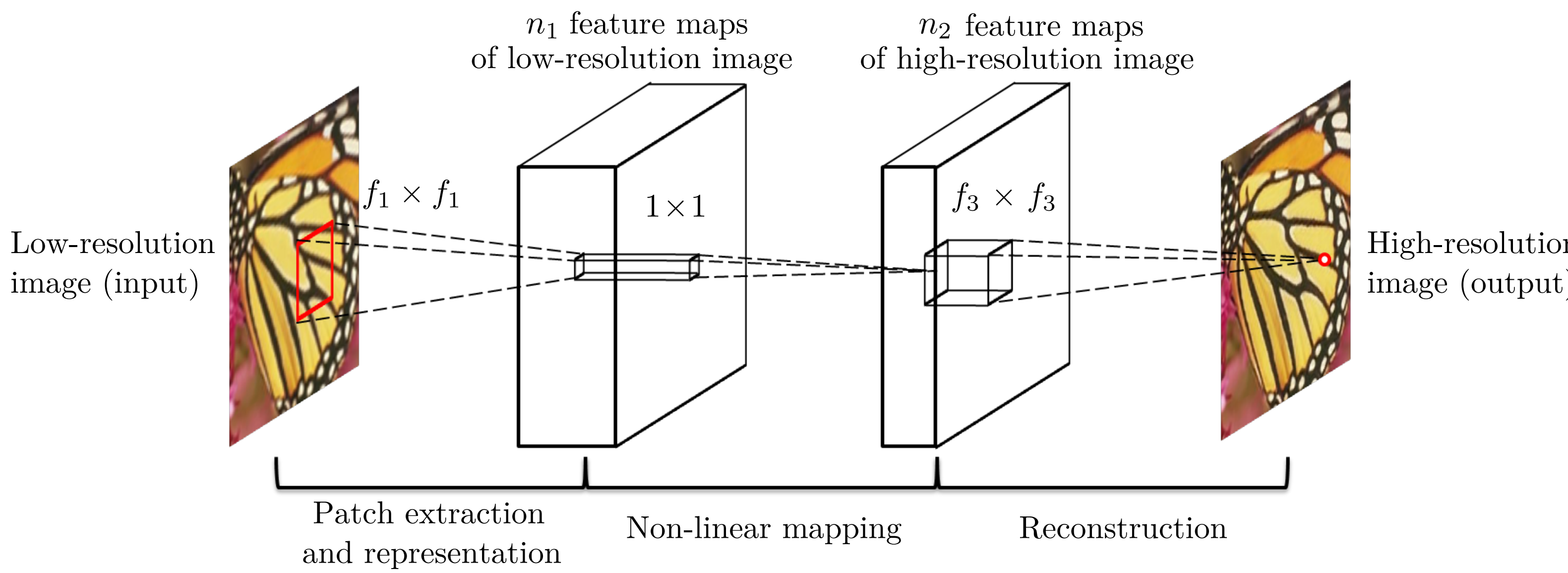
- Si on veut construire un réseau qui interpole les images:
  - On se donne une base de données d'images parfaites fn
  - Chaque image est sous-échantillonnée en une image gn
  - On veut retrouver fn à partir de gn, par exemple:

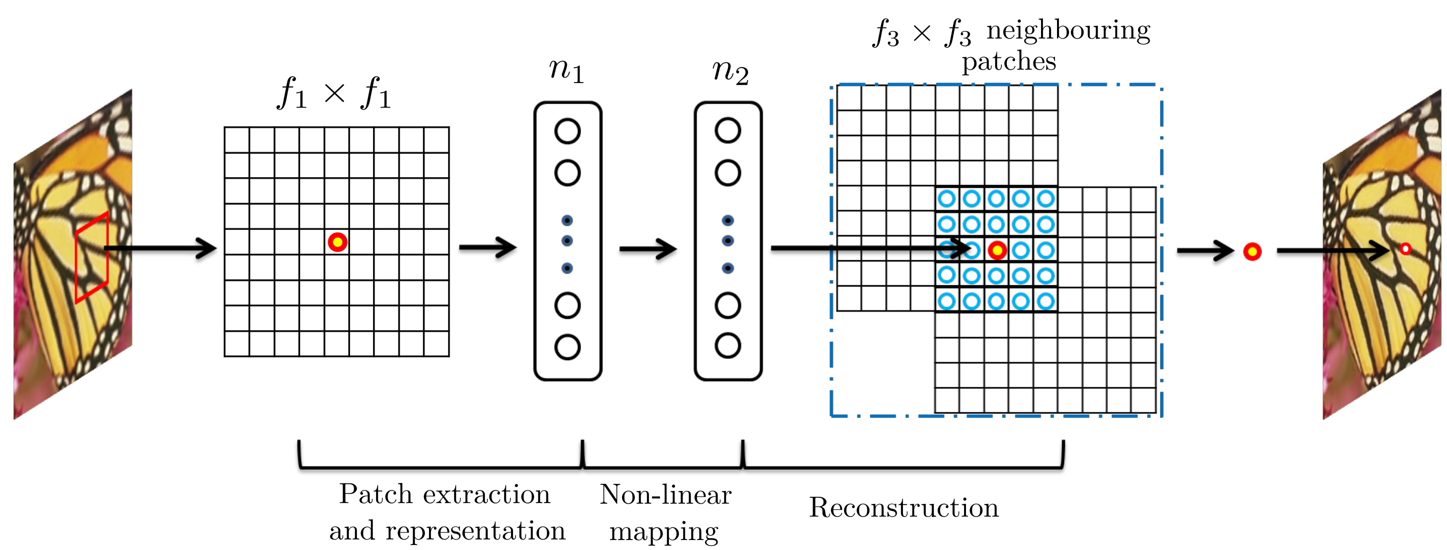
$$\mathcal{L}(W) = \sum_n \|\mathcal{C}(g_n, W) - f_n\|^2$$

Super-résolution: Points clés

- Il y a une plus grande variabilité des dégradations (flou).
- Généralement un réseau de zoom s’adapte mal à un flou inconnu.

Super-résolution: SRCNN





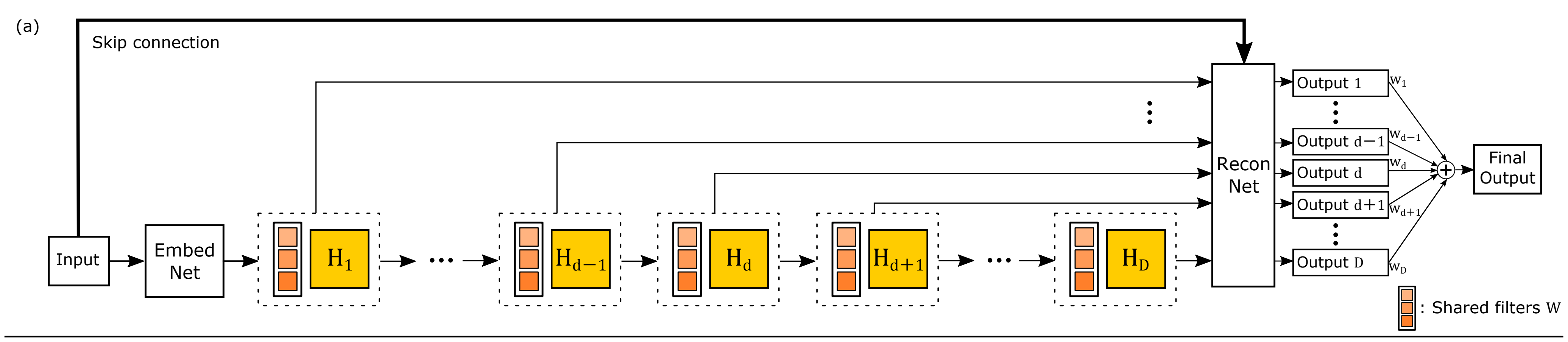
Super-résolution: SRCNN

- C'est la plus simple application des CNN au problème du zoom.

- La profondeur réduite ne lui permet pas d'atteindre de grandes performances.

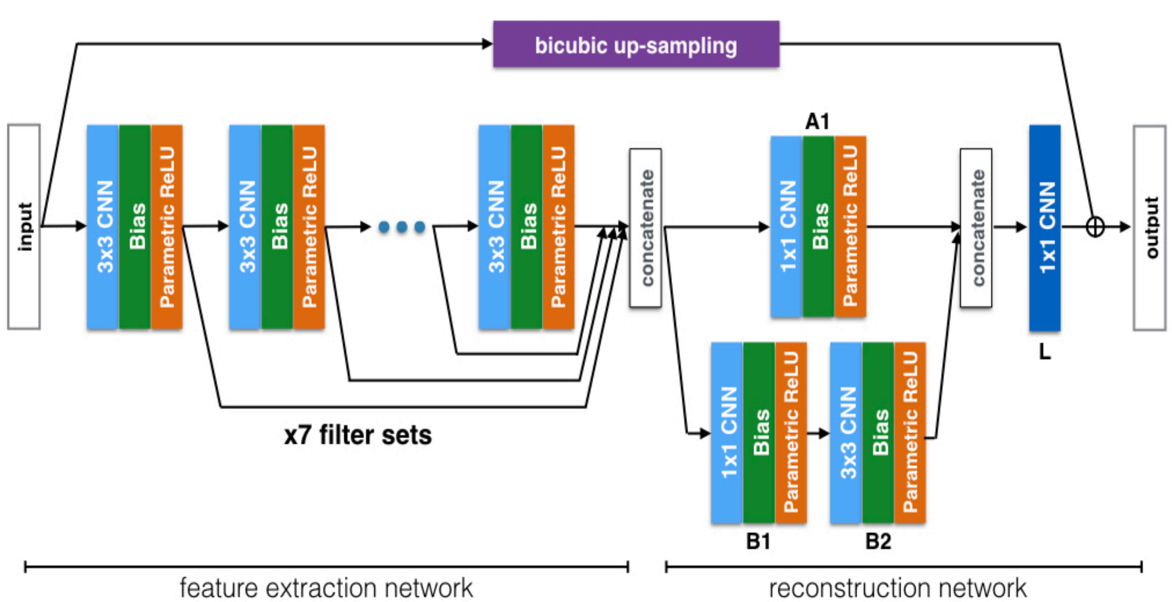
Super-résolution: DRCN

- Une couche récursive est apprise et répétée une vingtaine de fois.
- Chaque étape de la récursion produit un résultat.
- Les 20 résultats sont agrégés suivant des poids appris (20 poids).



Super-résolution: DCSCN

- Son but est la légèreté:
    - En particulier il prend en entrée l'image sous-échantillonnée.
    - La récursion est remplacée par une agrégation des résultats intermédiaires.



Travaux Pratiques

- Introduction à tensorflow.
- Test de modèles.
- Un entraînement pour reverse-engineerer le noyau de floutage.

Références:

- SRCNN: Dong et al. : Learning a Deep Convolutional Network for Image Super-Resolution. (ECCV 2014)
- DRCN: Kim et al. : Deeply-Recursive Convolutional Network for Image Super-Resolution. (CVPR 2016)
- DCSCN: Yamanaka et al.: Fast and Accurate Image Super Resolution by Deep CNN with Skip Connection and Network in Network (Neural Information Processing 2017)
- DnCNN: Zhang et al.: Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising (IEEE TIP 2017)
- FFDnet: Zhang et al.: FFDNet: Toward a Fast and Flexible Solution for CNN-Based Image Denoising (IEEE TIP 2017)