Aix Marseille Université — 2024-2025
Master IAAA — 2ème année

UE SOAP

Séance 2

Exercice 1 (*Filtrage 2D : extraction de contours dans une image.*)

Complétez le fichier `Seance 2 - detection de contours.ipynb` où l'on effectue des filtrages 2D sur des images pour estimer les contours. On utilisera la fonction `scipy.signal.convolve2d` pour effectuer la convolution 2D.

Exercice 2 (*Débruitage par CNN*)

On considère le problème de débruitage de sons introduit en début de séance :

- chaque son original dégradé par l'ajout d'un bruit gaussien
- on entraine un CNN à débruiter les sons, les CNN ayant une couche avec un seul filtre et une fonction d'activation linéaire ; l'entrainement consiste à donner en entrée des exemples de sons bruités et en sortie les sons non-bruités associés.

On réalise cela dans plusieurs contextes différents afin d'interpréter les résultats : filtre appris, performances obtenues.
Dans chaque question, on fera varier un paramètre en prenant 3 ou 4 valeurs différentes et en gardant les autres paramètres constants.

1. En utilisant des sinusoïdes comme dans l'exemple du cours, faites varier la largeur `df` de la plage de fréquences des sinusoïdes : qu'observez-vous sur la réponse fréquentielle du filtre appris et sur les performances obtenues (SNR) ? Expliquez.
2. En utilisant des sinusoïdes comme dans l'exemple du cours, faites varier la taille du filtre appris : qu'observez-vous sur sa réponse fréquentielle et sur les performances obtenues (SNR) ? Expliquez.
3. En utilisant des sinusoïdes comme dans l'exemple du cours, faites varier la fréquence centrale de la plage de fréquence des sinusoïdes : qu'observez-vous sur la réponse fréquentielle du filtre appris et sur les performances obtenues (SNR) ? Expliquez.
4. Avec de vrais sons bruités (ou des images), faites varier la taille du filtre et commentez les résultats obtenus (réponse fréquentielle du filtre, performance). Vous pouvez comparer vos résultats ceux obtenus avec deux filtres moyenneurs courramment utilisés : un filtre de taille M et de coefficients constants $1/M$, et un filtre de taille M obtenu en appelant la fonction `scipy.signal.windows.hamming(M)` et en normalisant les coefficients pour que leur somme soit égale à un.
5. Question subsidiaire : apprenez un RNN, le plus simple possible, permettant de réaliser le débruitage dans le cas des sinusoïdes du cours.