

TP 4 - Au delà de Fourier

(DCT, DCT locale, multirésolution et ondelettes)

Maxime Descoteaux

17 novembre 2025

Vous devez rédiger un rapport et me remettre un zip avec votre code python. Commentez le code et assurez-vous que je puisse reproduire vos résultats et figures. Séparez votre code en différents fichiers pour faciliter la lecture. Des points seront attribués pour la qualité du document et ses figures, et la qualité du code python (10 points). **La de remise du TP sera déterminée en classe.**

L'approximation linéaire garde que les M coefficients de plus basses fréquences dans l'espace de Fourier, c'est-à-dire les M premiers coefficients. Une approximation nonlinéaire garde que les M coefficients les plus élevés de la décomposition ou transformée. Dans ce devoir, quand vous comparez des transformées, assurez-vous de toujours prendre les mêmes M .

1. **Approximations 1D. [10 points]** À partir du *piece regular*, reproduisez les figures de mon code de DCT locale, disponible dans la Demo08.
 - (a) Illustrez votre signal.
 - (b) Décrivez votre choix de taille de segment et nombres de coefficients à couper M .
 - (c) Faites des approximations linéaires et non-linéaires de votre signal
 - (d) Rapporter les erreurs quadratiques moyennes de toutes les approximations dans un tableau.
2. **Approximations avec des transformées de Fourier et d'ondelettes** sur une image de votre choix. Attention ! Mettez-la en noire et blanc. Faites attention au type de l'image.
 - 5 pts a) **Approximation de Fourier.** L'approximation linéaire garde que les M coefficients de plus basses fréquences dans l'espace de Fourier. Faites l'approximation linéaire dans Fourier gardant que M coefficients, pour 3 M différents et rapportez le SNR des images approximées ainsi que les erreurs relatives d'approximation sur les *plot*. Illustrez les images des coefficients des transformées avant et après approximations linéaires. (Vous avez déjà fait cela dans le TP3)
 - 10 pts b) **Approximation avec cosinus discrets et des cosinus discrets locaux.** En premier temps, faites l'approximation linéaire avec une transformée de cosinus discrets (DCT) et une transformée de cosinus discrets locaux (locDCT), gardant que les M coefficients de plus basses fréquences, pour 3 M différents et rapportez le SNR des images approximées ainsi que les erreurs relatives d'approximation sur les *plot*. Utilisez les fonctions *dct2.py* et *idct2.py*, données en classe (vous devez

étendre ce que j'ai déjà fait en 1D en 2D). Illustrez les images des coefficients des transformées avant et après approximations linéaires.

- 10 pts c) **Approximation d'ondelettes de Haar périodiques.** À partir de votre ondelette de Haar, faites l'approximation linéaire avec cette transformée en utilisant 3 M différents. Rapportez le SNR des images approximées et les erreurs relatives d'approximation sur les *plot*. Illustrez les images des coefficients des transformées avant et après approximations linéaires.
- 10 pts d) **Approximations nonlinéaires.** Une approximation nonlinéaire garde que les M coefficients les plus élevés de la transformée. Faites les approximations nonlinéaires avec toutes les transformées des parties (a) à (d). Illustrez-bien les images des coefficients de la transformée conservées.
- 10 pts e) Faites un **tableau des SNR et des erreurs d'approximations linéaires et nonlinéaires**. Que remarquez-vous ?

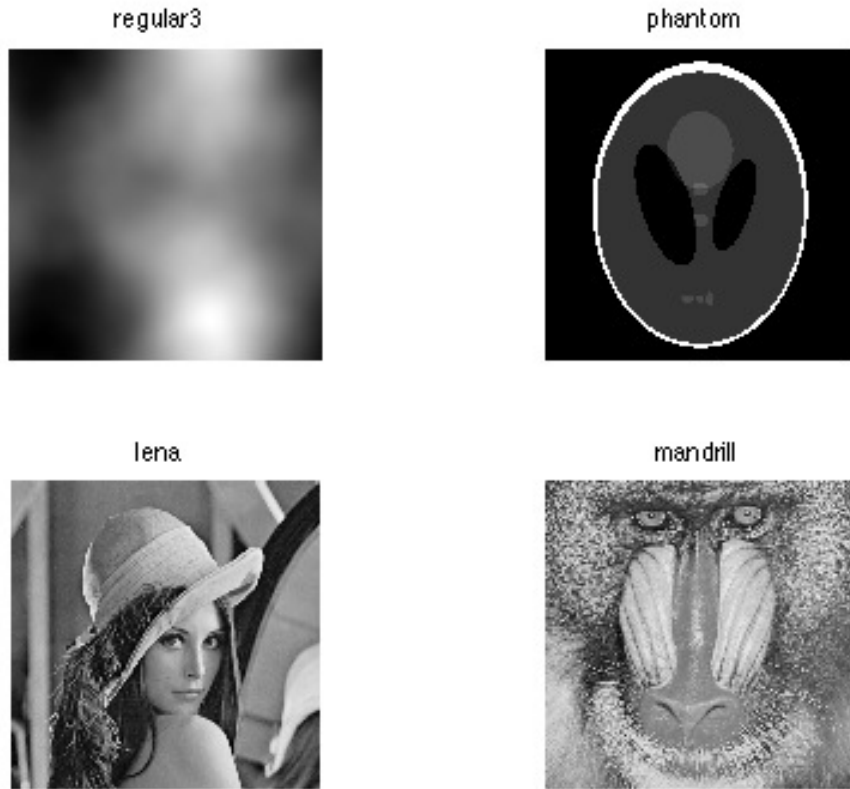


FIGURE 1 – 4 images classiques en traitement d'images à approximer nonlinéairement.

3. Approximations sur des images de différentes complexités.

20 pts i) Faites les approximations nonlinéaires pour les 4 images de la figure 1 (regular, phantom, lena et le mandrill). Que remarquez-vous ? Les approximations dépendent-elles de la "complexité" de l'image ? Expliquez.

5 pts ii) Selon vous, quelle est la meilleure transformée ? Pourquoi ?