CPE LYON

**Année universitaire**

**2022 - 2023**

---

# Majeure IMI — Partie 3 — 5ETI

# Compression et Techniques Avancées en Image

## TP de Compression

---

**Eric Van Reeth**

# Organisation du TP

## Déroulement

Ce TP s'effectue en binôme par poste informatique (sous LINUX).
Le langage utilisé sera Python (attention à l'interpréteur), avec les librairies suivantes :

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
import pywt # librairie ondelettes PyWavelets
```

L'utilisation de l'IDE `VSCode` est fortement conseillée pour faciliter l'implémentation et le debug des codes. Il est également possible d'implémenter votre TP sous le format Jupyter Notebook, en sauvegardant votre script au format `.ipynb`.

## Évaluation

**Le code développé** ainsi qu'un **compte-rendu au format pdf** devront être rendus pour évaluation dans un délai d'une semaine après le TP.
Le code doit être clair, commenté, et doit pouvoir être exécuté de façon autonome par l'évaluateur.
Le compte-rendu doit décrire de façon claire et illustrée la méthodologie mise en œuvre. Les choix réalisés à chaque étape seront clairement justifiés. Les images intermédiaires obtenues durant le processus de compression seront également insérées pour faciliter l'interprétation de la méthode implémentée. Enfin, les résultats obtenus seront présentés, quantifiés, et analysés de façon critique (avantages, inconvénients).

# Travail demandé

En s'inspirant des méthodes vues en cours, il vous est demandé de **proposer une méthode de compression d'image par transformation avec perte.** Les images que vous aurez à traiter sont initialement codées sur 8 bits (codage uniforme), et vous ne **considérerez que la version en niveaux de gris** dans un premier temps. Chaque binôme traitera une image unique.
Votre méthode sera composée des trois grandes parties suivantes :

1. **transformation** de l'image originale dans un autre domaine
2. **codage et quantification** des coefficients
3. **décodage et décompression** de l'image

De plus, il vous sera demandé de :

- **quantifier le taux de compression** obtenu par rapport à un codage uniforme
- proposer un paramètre permettant de régler le taux de compression obtenu par votre méthode
- **quantifier la dégradation** engendrée par votre méthode

De façon facultative, vous pourrez appliquer votre méthode sur la version en couleur de l'image.

# Annexe : aide de code

## Ondelettes

```
c = pywt.wavedec2() # décomposition en ondelettes 2D d'une image sur
 ↪  plusieurs niveaux de résolution. Coefficients c stockés dans une liste
 ↪  structurée
I = pywt.waverec2(c) # opération inverse
pywt.ravel_coeffs(c) # extraction des coefficients contenus dans c dans un
 ↪  array 1D
pywt.unravel_coeffs() # opération inverse
arr, slices = pywt.coeffs_to_array(c) # extraction des coefficients
 ↪  contenus dans c dans un array 2D
c = pywt.array_to_coeffs(arr, slices) # opération inverse
```

## Autres transformées

```
I_dct = cv2.dct(I.astype(float)) # DCT -> notez la conversion en float
I = cv2.idct(I_dct) # DCT inverse
I_fft = np.fft.fft2(I) # DFT
I = np.fft.ifft2(I_fft) # DFT inverse
```

## Divers

```
hist_I, bin_edges = np.histogram(I,bins=nbBins,range=(iMin,iMax)) # calcul
 ↪  de l'histogramme. Paramètres nbBins, iMin, iMax à spécifier
np.log2() # logarithme en base 2
plt.imsave('figname.png',I,cmap='gray') # sauvegarde sur le disque du
 ↪  contenu de l'image I
plt.savefig('figname.png',bbox_inches='tight',transparent=True) #
 ↪  sauvegarde sur le disque du contenu de la dernière figure affichée
 ↪  (avec les axes)
```