HAAR WAVELET TRANSFORM
(IMAGE MULTI-RESOLUTION ANALYSIS)

- Atanda Abdullahi Adewale

Introduction:

This work shows that the Haar transformation can be used for image compression. The results and effectiveness is determined in terms of PSNR value between original image and reconstructed image at different thresholds.

A Wavelet transformation converts data from the spatial into the frequency domain and then stores each component with a corresponding matching resolution scale

The **Haar** function:

$$\psi(t) = \begin{cases} 1 & t \in [0, 1/2) \\ -1 & t \in [1/2, 1) \\ 0 & t \notin [0, 1) \end{cases}$$

and

$$\psi_i^j(t) = \sqrt{2^j}\psi(2^j t - i), \; j = 0, 1, \ldots \text{ and } i = 0, 1, \ldots, 2^j - 1 .$$

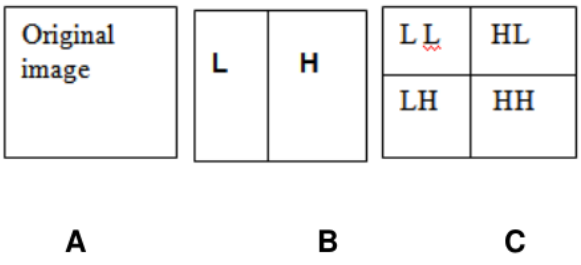
Haar Transform is nothing but averaging and differencing.

This can be explained with a simple 1D image with eight pixels **[3 2 -1 -2 3 0 4 1].** Applying the Haar wavelet transform, we can represent this image in terms of a low-resolution image and a set of detail coefficients.

So the image after one Haar Wavelet Transform is:

Transformed coefficient	=	**[2.5 -1.5 1.5 2.5]**
Detail Coefficients	=	**[0.5 0.5 1.5 1.5]**

The detail coefficients are used in reconstruction of the image. Recursive iterations will reduce the image by a factor of two for every cycle.



A = Original Image
B = First Run along Row
C = First Run Along Column

The image array is split into two halves containing the transformed data and the detail coefficients. The transformed data coefficients are the results of the low-pass filter while the detail coefficients are the results of the high-pass filter.

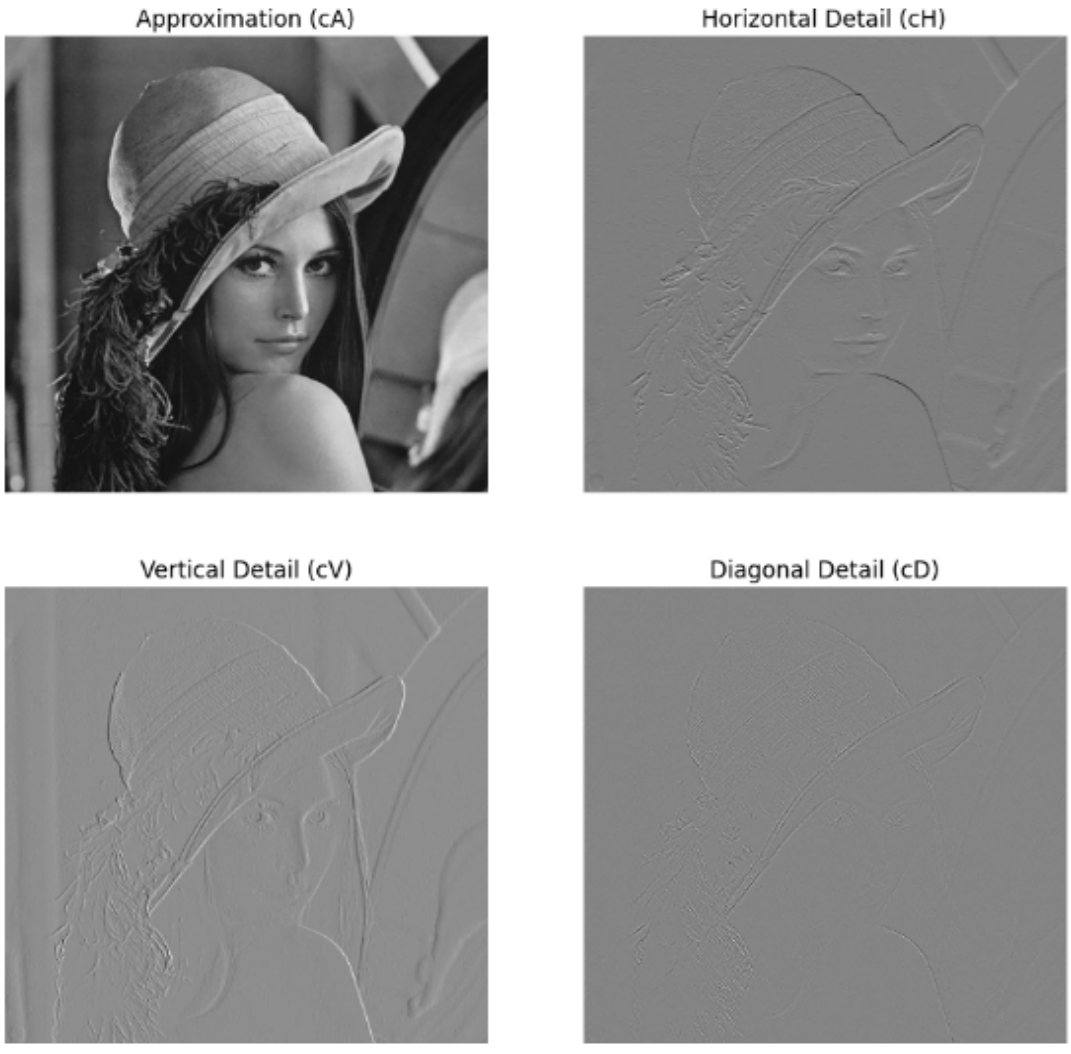
After transforming the image in the row, the image is then transformed along the column. This process repeats till three iterations

Methods:

1. Choose a gray-scale image with a maximum value (intensity) M - this value is used in the definition of the below PSNR

Grayscale image (‘lena.jpg’)

2. Apply the DWT (multiresolution analysis with the Haar Wavelet) on 1 level



3. Apply a threshold T

```
def dwt_threshold(img, threshold):
    coeffs = pywt.dwt2(img, 'haar')
    cA, (cH, cV, cD) = coeffs
    cA_thresh = np.where(np.abs(cA) < threshold, 0, cA)
    cH_thresh = np.where(np.abs(cH) < threshold, 0, cH)
    cV_thresh = np.where(np.abs(cV) < threshold, 0, cV)
    cD_thresh = np.where(np.abs(cD) < threshold, 0, cD)
    coeffs_thresh = (cA_thresh, (cH_thresh, cV_thresh, cD_thresh))
    img_thresh = pywt.idwt2(coeffs_thresh, 'haar')
    return img_thresh, (cA, (cH, cV, cD))
```

The absolute values of cA, cH, cV, cD are compared to the given threshold value. If the absolute value is less than the threshold, it is set to 0; otherwise, it remains unchanged.

```
dwt_img, coeffs = dwt_threshold(img, M)
cA, (cH, cV, cD) = coeffs
```

dwt_threshold value is M: maximum intensity value in the grayscale image.
It is used as the threshold for the DWT coefficients. Thresholding compares the absolute values of the *coefficients* to the threshold (M) and sets them to 0 if they are below the threshold.

img_thresh = pywt.idwt2(coeffs_thresh, 'haar')
Performs the inverse Discrete wavelet Transform (IDWT) using the thresholded coefficients (coeffs_thresh) and the Haar wavelet. It reconstructs the denoised image.

4. Compute the PSNR (reference image = orignal image) of the reconstruction (DWT-1)

```python
def compute_psnr(img1, img2):
    # Resize images to match dimensions
    img1 = cv2.resize(img1, (img2.shape[1], img2.shape[0]))

    mse = np.mean((img1 - img2) ** 2)
    max_value = np.max(img1)
    psnr = 10 * np.log10((max_value ** 2) / mse)
    return psnr
```

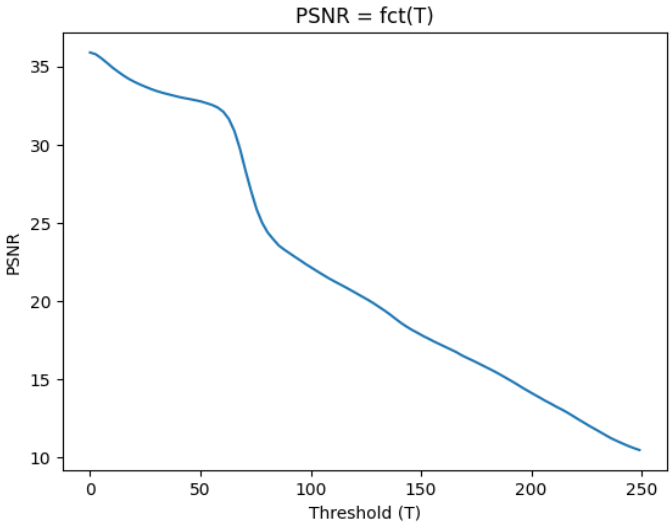
```python
# Compute PSNR for reconstruction
psnr_reconstruction = compute_psnr(img, dwt_img)
print("PSNR for reconstruction (DWT-1):", psnr_reconstruction)
```

```
PSNR for reconstruction (DWT-1): 10.47258377947826
```

5. Plot PSNR = fct(T)

```python
thresholds = np.linspace(0, M, 100)
```

So, **thresholds** array will contain 100 values starting from 0 and ending at M, with evenly spaced intervals between them.



PSNR as a function of the threshold values (T).

6. Repeat the process with the noisy image (add a Gaussian White Noise, noise level-> standard deviation D)

```python
# Add Gaussian white noise to the image
std_dev = 20
noisy_img = add_gaussian_noise(img, std_dev)
```

7. Find the optimal threshold To for denoising (PSNR). Note the reference image is the noise-free image

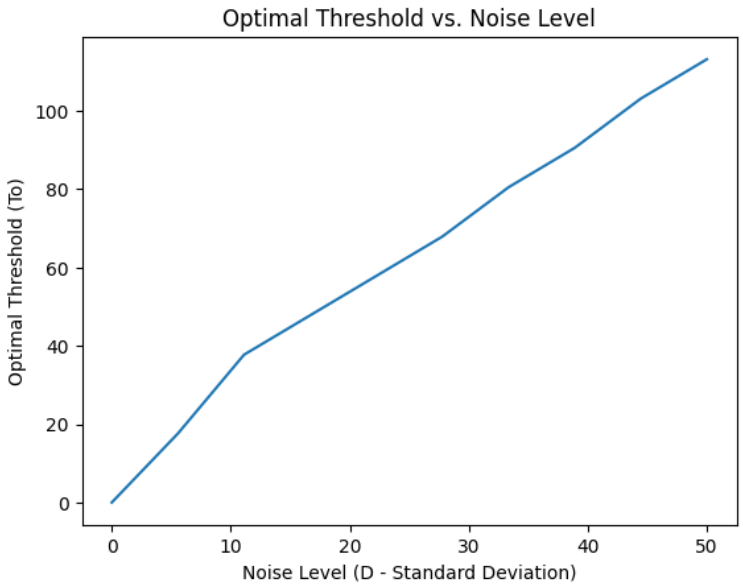
```
# Find the optimal threshold for denoising
optimal_threshold, psnr_values = optimal_threshold_denoising(noisy_img, img, M)
print("Optimal threshold for denoising (PSNR):", optimal_threshold)
```

```
Optimal threshold for denoising (PSNR): 52.81818181818182
```

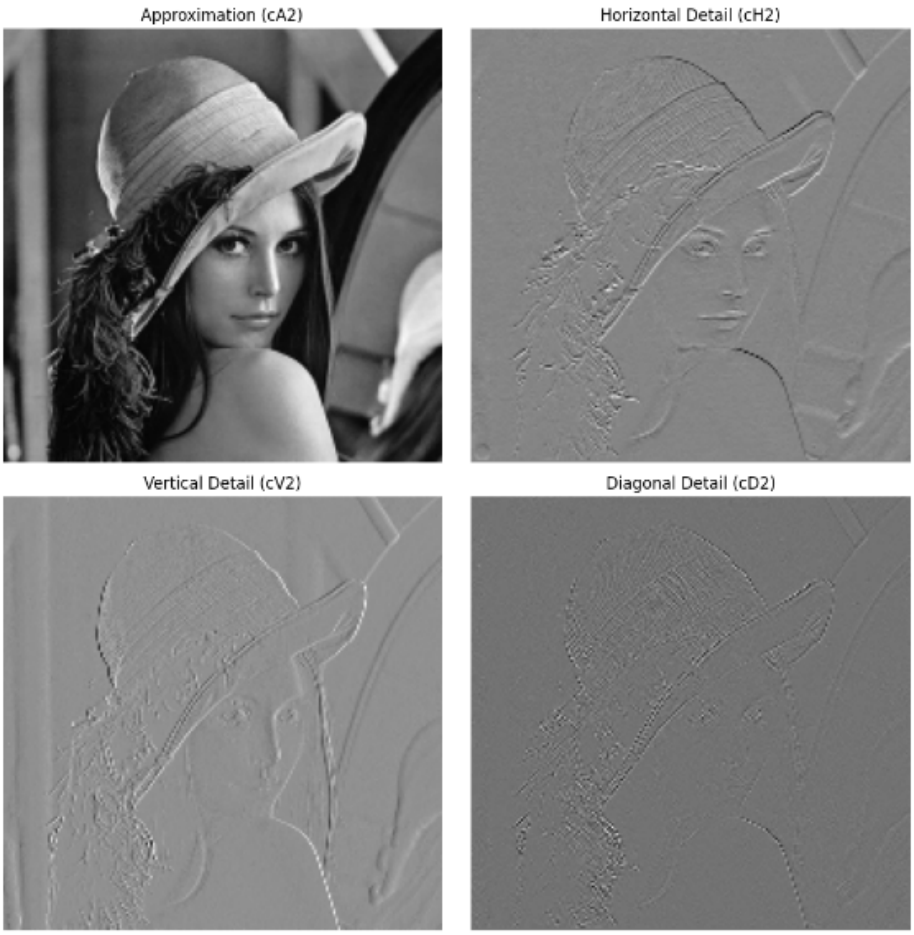
8. Study the evolution of optimal threshold To in respect of the noise level (D)

```
# Study the evolution of To in respect of the noise level (D)
std_dev_values = np.linspace(0, 50, 10)
optimal_thresholds = []
for std_dev in std_dev_values:
    noisy_img = add_gaussian_noise(img, std_dev)
    optimal_threshold, _ = optimal_threshold_denoising(noisy_img, img, M)
    optimal_thresholds.append(optimal_threshold)
```

Iterates over a range of standard deviation values (0, 50, 10), adds Gaussian noise to the image, and then finds the optimal threshold using the optimal_threshold_denoising function. The optimal thresholds are collected in the optimal_thresholds list for visualization.



9. Repeat all the process with two levels (L1 and L2) in the multiresolution analysis. Find the optimal thresholds T1 and T2. What is the relation between these T1 and T2 ?



DWT-2 Coefficients

Conclusion

For optimal thresholds T1 and T2, i apply the DWT with two levels, perform thresholding on the coefficients at each level, and then evaluate the denoising performance using PSNR.

The relation between T1 and T2 depend on the specific characteristics of the image and the noise such as the noise level, image content, and the desired balance between denoising and preserving image details.

From PSNR, T2 yield the highest denoising quality because the higher-level coefficients capture finer details and noise, so a higher threshold is needed to remove the noise without excessively distorting the image.