1 Theory

1. Suppose $f(w)$ has the Fourier series representation $f(w) = \sum_{k\in\mathbb{Z}} c_k e^{ikw}$ and suppose that $g(w) = e^{imw} f(w)$ for some $m \in \mathbb{Z}$, show that
$$g(w) = \sum_{k=-\infty}^{\infty} d_k e^{ikw}, \text{ where } d_k = c_{k-m}.$$

2. Find the Fourier series for the 2π-periodic square wave function
$$f(\omega) = \begin{cases} -k & \text{if} \quad -\pi < \omega < 0 \\ k & \text{if} \quad 0 < \omega < \pi \end{cases} \quad \text{and} \quad f(\omega + 2\pi) = f(\omega)$$

3. Compute by hand the Haar wavelet decomposition (Pyramidal decomposition) of the vector $\mathbf{x}^T = [1, 7, -3, 2]$ by viewing it as
$$f(x) = \begin{cases} 1 & \text{if} x \in \left[0, \frac{1}{4}\right), \\ 7 & \text{if} x \in \left[\frac{1}{4}, \frac{1}{2}\right), \\ -3 & \text{if} x \in \left[\frac{1}{2}, \frac{3}{4}\right), \\ 2 & \text{if} x \in \left[\frac{3}{4}, 1\right). \end{cases}$$
Graphically show the projections onto the scaling and wavelet subspaces.

2 Computing

1. Implement a 3×3 *median filter* and apply the filtering process on a corrupted image of "app-ndt-Chip-5.JPG" located via the course website. Specifically, corrupt "app-ndt-Chip-5.JPG" with *salt-and-pepper* noise, where the corrupted pixels are either set to the maximum value (which looks like snow in the image) or have single bits flipped over. In some cases, single pixels can be set alternatively to zero or to the maximum value (i.e., 255 on a 8-bit machine). Then apply the median filter to de-noise the corrupted image. Compare your result with the original.

2. Given an image "CTimage.JPG" on the course website. Perform the following operations:
   (a) Construct a 3×3 average filter to smooth the image.
   (b) Then use a $2D$ Laplacian filter mask to extract the edges of the smoothed image.
   (c) Finally, enhance the smoothed image with the result from part (b). How does this image compare to the original?

3. Test your codes with your favorite image for this problem. Make sure your codes are as general as possible and be sure to plot the results.
   (a) Write a function in MATLAB to compute the *approximation* and each of the three *detail* components of an image. (i.e., you will produce 4 *extremely* short codes here) Notice that the resolution of LL, HL, LH, and HH will be $M/2 \times N/2$, where (M,N) is the resolution of the original image.
   (b) Write a subroutine to reconstruct an image from **only** the *approximation* component as a function of level. Notice that the resolution of the reconstructed image will be of size $M \times N$.

(c) Compress the image up to level 3 using **only the approximation component**. Compute the compression ratio as a function of each compression level. Plot the compressed image for each level along with their compression ratio. Note that the compression ratio in this case can be defined as

$$CR = \frac{\text{\# of nonzero entries in the original}}{\text{\# of nonzero entries in the compressed}}.$$

Some useful MATLAB commands for this problem:

```
>> dwt2
>> wavedec2
>> wrcoef2
```