ASSIGNMENT #9 (EC520)
“Digital Image Processing and Communication”
Date: April 6, 2022
Due date: April 13, 2022

1. *DCT and DWT basis restriction error – Matlab* (40 points)
   For image *barbara.tif* perform the following experiments.

   (a) *DCT*:

      i. Apply 2-D DCT (`dct2`) to the complete image (not block-by-block).
      ii. Keep $p\%$ of the largest-energy coefficients (squared value) unchanged and set the remaining ones to zero. *Hint*: Perhaps the easiest way to accomplish this is to re-shape the 2-D array of squared coefficients into a 1-D vector, apply `sort` to order the coefficients, find a threshold above which there are $p\%$ of squared coefficients and zero all coefficients in the 2-D array below this threshold. Of course, the result would have to be “unsorted” before taking inverse DCT.
      iii. Apply 2-D inverse DCT (`idct2`) to the partially-zeroed coefficient array. This results in an *approximate* image.
      iv. Compute the mean-squared error (MSE) between the original and approximate images. Also, express the MSE as PSNR (defined for pixel values 0-255).
      v. Perform steps (ii-iv) for $p = 10, 20, ..., 90\%$.

   (b) *DWT - Haar*:

      i. Apply 2-D, two-level “Haar” DWT (`dwt2`) to the image using the so-called periodization mode “per” (to handle the filters’ boundary conditions). A two-level DWT means that after you have computed the four half-sized images (LL, HL, LH, HH), you apply the same procedure to the LL image.
      ii. Keep $p\%$ of the largest-energy coefficients **among all 7 bands** (not in each band separately) unchanged and set the remaining ones to zero.
      iii. Apply 2-D, two-level inverse “Haar” DWT with periodization mode “per” (`idwt2`) to the truncated coefficient array.
      iv. Again, compute MSE and PSNR between the original and approximate images.
      v. Perform steps (ii-iv) for $p = 10, 20, ..., 90\%$.

   (c) *DWT - Daubechies 4*:
   Repeat the experiment from part (b) for DWT using the Daubechies filter “db4”.

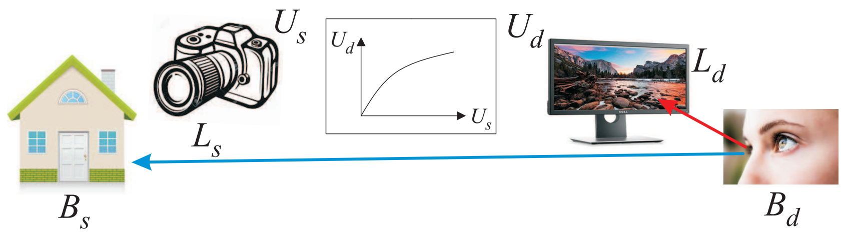
   With your solutions include 2 pages of results. On the first page include:

   - `subplot(2,2,1)`: original image,
   - `subplot(2,2,2)`: DCT-based approximation for $p = 10\%$,
   - `subplot(2,2,3)`: DWT/”Haar”-based approximation for $p = 10\%$,
   - `subplot(2,2,4)`: DWT/”db4”-based approximation for $p = 10\%$.

   On the second page, include MSE and PSNR plots each with three graphs (DCT, “Haar”, “db4”) as a function of p. What conclusions can you draw from the images (carefully inspect them, preferably switching between them in-place) and from MSE/PSNR plots?

2. *Image enhancement* (20 points)

   Gamma correction is needed to compensate non-linearities in a visual communication system, such as the one shown below.



   Assume the following *approximate* models in the above diagram:

   - human visual system outdoors in daylight $\approx$ brightness/luminance relationship: $B_s = 106 \log_{10} L_s$,
   - camera electronics $\approx$ voltage/luminance relationship: $U_s = \sqrt{L_s}/16$,
   - display electronics $\approx$ luminance/voltage relationship: $L_d = 255 \cdot U_d^{5/2}$,
   - human visual system indoors in semi-darkness $\approx$ brightness/luminance relationship: $B_d = 255^{2/3} \cdot L_d^{1/3}$.

   Find the non-linearity $U_s = f(U_d)$ that needs to be inserted between the camera and the display so that users experience a displayed image as if they were present in the scene, i.e., $B_d = B_s$.

   Also, using *Matlab*, plot the following relationships for $0.1 < U_s, U_d \leq 1.0$:

   - `subplot(1,3,1)`: B_s as a function of U_s,
   - `subplot(1,3,2)`: U_s as a function of U_d,
   - `subplot(1,3,3)`: B_d as a function of U_d.

   How is the plot of U_s in terms of U_d related to the other two plots? Comment.

3. *Image restoration* (40 points).
   Consider the discrete-space image formation model from slide #21 without point-wise non-linearity: $\boldsymbol{v} = \boldsymbol{H}\boldsymbol{u} + \boldsymbol{\eta}$, where $\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{\eta}$ are N^2-long vectors respectively representing the observed, original and noise images of size $N \times N$ that are scanned row by row, and $\boldsymbol{H}$ is an $N^2 \times N^2$ circulant matrix representing acquisition system's blur (convolution with impulse response h). In class, we developed image restoration methods using random-field models (ML and MAP estimation). In this problem, you will develop least-squares solutions.

   (a) Assume $h[m], m = 0, ..., M-1$ is the impulse response of a *horizontal* (row) filter and $M << N$. Express convolution of this filter with the original image: $h[m] * u[n, m]$ in matrix form: $\boldsymbol{H}\boldsymbol{u}$, clearly showing elements of matrix $\boldsymbol{H}$.

   (b) Suppose that during image formation the blurred image $\boldsymbol{H}\boldsymbol{u}$ undergoes downsampling by a factor of 2 both horizontally and vertically. Write an $N^2/4 \times N^2$ sampling matrix $\boldsymbol{D}$ that implements this downsampling. Then, filtering followed by downsampling can be implemented by matrix multiplication $\boldsymbol{v} = \boldsymbol{D}\boldsymbol{H}\boldsymbol{u}$ or $\boldsymbol{v} = \boldsymbol{G}\boldsymbol{u}$ with $\boldsymbol{G} = \boldsymbol{D}\boldsymbol{H}$.

(c) The least-squares image restoration for the case of filtering and downsampling modeled as $\boldsymbol{v} = \boldsymbol{G}\boldsymbol{u} + \boldsymbol{\eta}$ can be formulated as follows:

$$\widehat{\boldsymbol{u}} = \arg\min_{\boldsymbol{u}} \|\boldsymbol{v} - \boldsymbol{G}\boldsymbol{u}\|_2^2.$$

Show that $\widehat{\boldsymbol{u}} = (\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{v}$. You have likely seen this formulation and result in other classes but I would like you to derive it. <u>*Hint:*</u> Here are key relationships from vector calculus that are needed in this derivation: $\|\boldsymbol{x}\|_2^2 = \boldsymbol{x}^T\boldsymbol{x}$, $\frac{\partial \boldsymbol{a}^T\boldsymbol{x}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{x}^T\boldsymbol{a}}{\partial \boldsymbol{x}} = \boldsymbol{a}$, $\frac{\partial \boldsymbol{x}\boldsymbol{A}^T\boldsymbol{x}}{\partial \boldsymbol{x}} = (\boldsymbol{A} + \boldsymbol{A}^T)\boldsymbol{x}$, where $\boldsymbol{a}, \boldsymbol{A}$ are not functions of $\boldsymbol{x}$. I expect you have seen these formulas in your multivariable calculus or linear algebra class.

(d) Since the least squares solution from part (c) is sensitive to noise, often the magnitude of the solution is constrained by means of Tikhonov regularization:

$$\widehat{\boldsymbol{u}} = \arg\min_{\boldsymbol{u}} \|\boldsymbol{v} - \boldsymbol{G}\boldsymbol{u}\|_2^2 + \lambda\|\boldsymbol{u}\|_2^2.$$

Show that in this case $\widehat{\boldsymbol{u}} = (\boldsymbol{G}^T\boldsymbol{G} + \lambda\boldsymbol{I})^{-1}\boldsymbol{G}^T\boldsymbol{v}$, where $\boldsymbol{I}$ is the identity matrix.

(e) Consider the minimization from part (d) above in the context of the MAP minimization $\min_{\boldsymbol{u}} E_P(\boldsymbol{u}; \boldsymbol{v})$ from slide #32 on image restoration. Clearly the two formulations are very similar. What size of cliques and what potential function would have to be used in the MAP formulation to implement the Tikhonov regularization?

(f) The above least-squares solutions are easy to implement by constructing matrix $\boldsymbol{G}$ and performing simple matrix manipulations. However, substantial computer resources may be needed. Estimate the computer memory needed to implement the solution from part (c) for a modest-size 512×512-pixel image using 4 bytes for each matrix element.

Important: If you collaborated with other EC520 students, you must list their names at the top of the first page. You must submit your solutions to Gradescope by midnight of the due date as follows:

- upload your analytic solutions and *Matlab* plots as a single PDF file; **remember to assign pages in your solutions to problems on Gradescope so that you receive credit for all work**,
- upload your *Matlab* or *Python* source code as a **single file**; include your name and homework number as a comment at the top of the source code; if you developed functions that you are using in your main code, paste them at the end of the file; your code should run by executing the file.