ECE 285 – Assignment #6
Wiener Deconvolution

Written by Charles Deledalle on May 14, 2019.

In this assignment we will implement the Wiener deconvolution algorithm as part of our image manipulation library `imagetools`.

First, start a Jupyter Notebook, go into the subdirectory `ece285_IVR_assignments` (or whatever you named it), and create a new notebook `assignment6_wiener.ipynb` with

```
%load_ext autoreload
%autoreload 2
import numpy as np
import numpy.fft as npf
import matplotlib
import matplotlib.pyplot as plt
import time
import imagetools.assignment6 as im

%matplotlib notebook
```

We will use

- `assets/eagle.png`
- `assets/owls_blur.png`
- `assets/plane.png`
- `assets/sheeps.png`

For the following questions, please write your code and answers directly in your notebook. Organize your notebook with headings, markdown and code cells (following the numbering of the questions).

1 Spectral deconvolution

An important ingredient for spectral deconvolution is the concept of mean power spectral density. The mean power spectral density is the expectation of the power spectral density for the given population (subset) of images that we are targeting. The formal definition is

$$S_{u,v} = \mathbb{E}[(S_x)_{u,v}] \quad \text{where} \quad (S_x)_{u,v} = |\hat{x}_{u,v}|^2 = \left| \sum_{k=0}^{n_1-1} \sum_{l=0}^{n_2-1} x_{k,l} e^{-i2\pi\left(\frac{uk}{n_1} + \frac{vl}{n_2}\right)} \right|^2 . \tag{1}$$

where the expectation $\mathbb{E}$ is relative to a probability space on which images x are seen as random vectors. In theory, to determine the mean power spectral density, one has to choose a probability space for x and evaluate the above expectation (which usually leads to the computation of a difficult integral). Instead, we will assume that $S_{u,v}$ follows a power law of the form

$$S_{u,v} = n_1 n_2 e^{\beta} \omega_{u,v}^{\alpha} \quad \text{for} \quad \omega_{u,v} \neq 0 \quad \text{where} \quad \omega_{u,v} = \sqrt{\left(\frac{u}{n_1}\right)^2 + \left(\frac{v}{n_2}\right)^2} . \tag{2}$$

In the first part, we are going to estimate α and β on a subset of three training images. Once α and β have been determined, we will use them to deconvolve a test image in the second part.

1.1 Part I (mean power spectral density)

1. Create in `imagetools/assignment6.py`, a function

   ```
   def average_power_spectral_density(x)
   ```

   that takes a list of images `x[1]`, `x[2]`, ..., `x[K]`, and compute the average of their power spectral density

   $$S^{(\text{avg})}_{u,v} = \frac{1}{K} \sum_{k=1}^{K} |\hat{x}^{(k)}_{u,v}|^2 \tag{3}$$

   For color images, average the power spectral density of each channel.

2. Run your new function on the three images `x[1] = eagle`, `x[2] = plane` and `x[3] = sheeps`, and display the logarithm of $S^{(\text{avg})}$ as

   ```
   im.showfft(apsd, apply_log=True, vmin=-10, vmax=5)
   ```

3. Show that except for the 0 frequency ($\omega_{u,v} \neq 0$), we have $\alpha t_{u,v} + \beta = s_{u,v}$ where $t_{u,v} = \log \omega_{u,v}$ and $s_{u,v} = \log S_{u,v} - \log n_1 - \log n_2$.

4. We now assume that $S \approx S^{(\text{avg})}$, and based on the above equation, we are going to estimate α and β from $S^{(\text{avg})}$ by least square linear regression. In our case, the least square linear regression consists in looking for α and β that minimizes the following sum of square errors (SSE)

   $$\text{SSE}(\alpha, \beta) = \frac{1}{2} \sum_{\substack{0 \leqslant u < n_1 \\ 0 \leqslant v < n_2 \\ (u,v) \neq (0,0)}} (\alpha t_{u,v} + \beta - s^{(\text{avg})}_{u,v})^2 \ . \tag{4}$$

   where $s^{(\text{avg})}_{u,v} = \log S^{(\text{avg})}_{u,v} - \log n_1 - \log n_2$. Find the mathematical expression of α and β that minimizes the SSE.

   Hint: they simultaneously cancel the partial derivatives of SSE with respect to α and β.

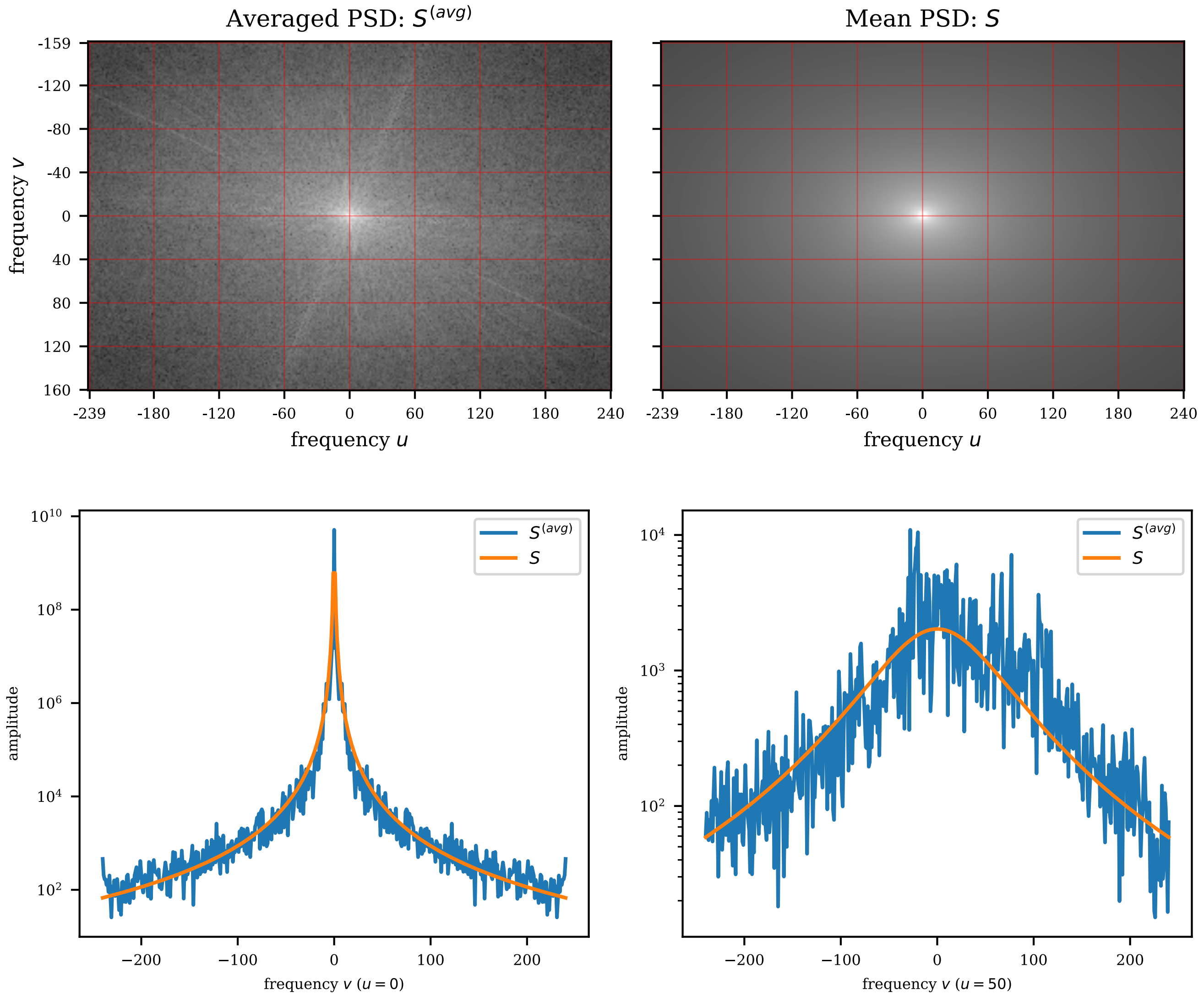
5. Based on these expressions, in `imagetools/assignment6.py`, write a function

   ```
   def mean_power_spectrum_density(apsd):
       ...
       return mpsd, alpha, beta
   ```

   that returns the estimated mean power spectral density S and the values of α and β.
   Set `mpsd[0, 0] = np.inf`.

6. In your notebook, call your new function in order to estimate S from $S^{(\text{avg})}$. What are the values of α and β?

7. Display next to $S^{(\text{avg})}$, the logarithm of S (using the same colormap and range). Display also one-dimensional slices of $S^{(\text{avg})}$ and S (by superimposition), for the frequency $u = 0$ and $u = 50$. Check that your results are consistent with the following ones:



1.2 Part II (Wiener deconvolution)

8. Load and display the image `y = owls_blur` and the logarithm of its spectrum. This image is corrupted by a blur of spread $\tau = 2$, and a noise with standard deviation $\sigma = 1/255$. Our purpose is to recover the sharp underlying image.

9. Use `im.kernel` and `im.kernel2fft` to create the frequency response λ corresponding to an exponential blur with $\tau = 2$.

10. In the first naive approach, we are going to deconvolve this image by simply dividing its spectrum by λ or, equivalently, multiplying it by the transfer function $\hat{h} = \lambda^*/|\lambda|^2$. Write in `imagetools/assignment6.py` the function

```python
def deconvolve_naive(y, lbd, return_transfer=False):
    ...
    if return_transfer:
        return xdec, hhat
    else:
        return xdec
```

that computes $\hat{h}$ and the deconvolved image as $x^{(\text{dec})} = \mathcal{F}^{-1}(\hat{h} \cdot \hat{y})$ (don't forget to take the real part). Display x and the modulus $|\hat{h}|$ of $\hat{h}$. Interpret the results.

11. Repeat this experiment with the Gaussian and the box kernel for $\tau = 2$. What is the condition number of each of these three deconvolution problems? Interpret the results.

12. The Wiener deconvolution consists of the same thing as the naive version, but instead with the following transfer function

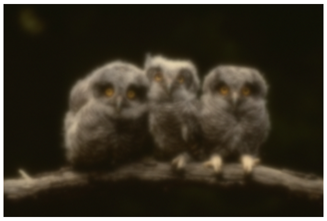
$$\hat{h}_{u,v} = \frac{\lambda^*_{u,v}}{|\lambda_{u,v}|^2 + n_1 n_2 \sigma^2 / S_{u,v}} \tag{5}$$

What does this filter tend to do when σ approaches 0? or $+\infty$?

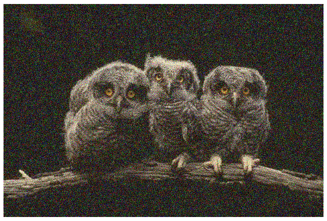
13. Create in `imagetools/assignment6.py` the function

```python
def deconvolve_wiener(x, lbd, sig, mpsd, return_transfer=False)
```

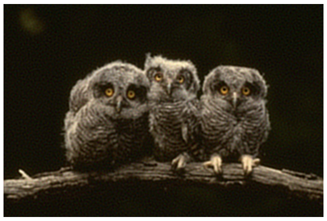
that implements the Wiener deconvolution, returns the result and the transfer function $\hat{h}$. Complete your notebook, to deconvolve x using the mean power spectrum density S that was learned in the first part of this exercise, and assuming an exponential kernel. Display $|\hat{h}|$, and the deconvolved image x. Check that your results are consistent with the following ones:



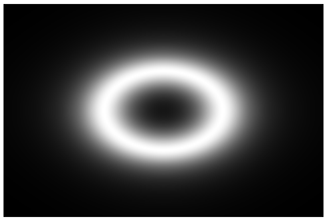
(a) Blurry image



(b) Naive deconvolution



(c) Wiener deconvolution



(d) Transfer function

14. What does Wiener deconvolution do?

15. Repeat for the Gaussian and the box kernel.
Which of these three kernel functions fits the best on our deconvolution problem?