Quantitative Methods in Neuroscience

(neu 466M)

Homework 4

Due: Tuesday March 6 by 5 pm in NHB 3.128 (or by 2 in class if preferred)

In this assignment you will further explore the STA, comparing its predictions with those from Wiener-Hopf filtering and studying the spike-triggered average with the 2-spike triggered average. You will also explore edge detection with convolution with a difference-of-Gaussians kernel, and do a little matrix algebra. *General guidelines:* Read through each complete problem carefully before attempting any parts. Feel free to collaborate in groups of size 2-3, but always note the names of your collaborators on your submitted homework. For graphs: clearly label your axes and use good color and symbol choices. Print out your matlab code (in the form of a script file). For derivations you're asked to do 'by hand' (in other words, analytically, using paper and pencil) feel free to turn in handwritten or typed-out work.

1) **The Wiener-Hopf filter estimate for neural kernels when the stimulus is a non-white (temporally correlated) Gaussian, and comparison with the STA.** This problem is an extension of Problem 3 from Homework 3.

   a. Download `generate_STAdata.m` from the course webpage and set `whitenoise = 0`. As you did in Homework 3, plot the STA estimate of the model neuron kernel and compare with the true kernel used to generate the data. Change the stimulus correlation time from `taustim =0.01` (s) to `taustim =0.02` and repeat; compare the STA estimates with the true kernels in plots, and explain the discrepancies.

   b. Now generate the Wiener-Hopf result for the model neuron kernel (with `taustim =0.01`). To generate the autocorrelation matrix, use the command `toeplitz`. In Matlab, to implement $A^{-1}\mathbf{b}$, where A is a matrix and $\mathbf{b}$ is a vector, use `A\b`. Plot the true kernel used to generate the data in the file `generate_STAdata.m`, plot over it the Wiener-Hopf kernel estimate, and finally plot over those the STA kernel estimate. Compare all three. Explain why you see the differences you do, and which is the better estimate of the true kernel. Submit your code and your plots.

2) **Convolution of 2D images with a 2D difference-of-Gaussians filter for edge detection.**

   a. Download the file `plant.mat` from the course webpage. Load it into matlab using `load plant.mat`. This file contains a grayscale image `plant`, which you can view using `imagesc(plant); colormap(gray)`. The command `colorbar` gives a scale for the colormap image.

b. Generate and plot a 2D difference-of-Gaussians kernel with parameters σ_1, σ_2 specifying the widths of the two Gaussians. Set the normalizations of the two Gaussians to 1, and let the parameter a control the height of the broad Gaussian relative to the narrow one. Hints: to generate a normalized 2D Gaussian kernel of length `2*L` in each dimension, with width `sigma` (make sure you choose a length that is much larger than the Gaussian kernel weidths, `L >> sigma`, so that the Gaussian decays to zero by the edge), you can use the commands `x=-L:1:L-1;` followed by
`kernel = (1/(2*pi*sigma^2))*exp(-((ones(2*L,1)*x).^2 + (x'*ones(1,2*L)).^2)/2/sigma^2)`. You can plot 2D kernels using `imagesc(kernel)`.

c. Make some choices for a, σ_1, σ_2, and generate `convimg` by convolving the image `img` with your difference-of-Gaussians kernel. E.g., start with $a = 1$ and some small values for the σ's, depending on how many pixels in the image each edge spans: for a sharp, high-resolution image, you might want values as small as 1 or 2. Next, generate `thres_convimg` by thresholding `convimg` at some appropriate value. Finally, detect the zero-crossings of the thresholded image using `diff`. (`diff` computes the derivative of the image along one of the two dimensions. When there is a step in the image from dark to light or light to dark, the magnitude of the derivative is non-zero. Thus, you can define an edge as any point in the image where `diff(thres_convimg)` is non-zero. We are interested in steps along both dimensions of the image. Thus, you will want to find edges along each dimension, and sum both to plot the final edge-detected image. Using `subplot`, make a set of four subplots: the original image, `convimg`, `thres_convimg`, and finally the detected images.

d. Make some choices for a, σ_1, σ_2, and generate `convimg` by convolving the image `img` with your difference-of-Gaussians kernel. Next, generate `thres_convimg` by thresholding `convimg` at some appropriate value. Finally, detect the zero-crossings of the thresholded image using `diff`. `diff` computes the derivative of the image along one of the two dimensions. When the magnitude of the derivative is non-zero, there is a step in the image from dark to light or light to dark. You can define an edge as any point in the image where `diff(thres_convimg)` is non-zero. We are interested in steps along both directions. Thus, you will want to find edges along each dimension, and sum both to plot the final edge-detected image.

e. Re-do the steps in d., using different values of the σ's and a, trying to obtain some systematic insight into what values work well for what purpose and why. Plot and explain.

3) **Matrix algebra: identity and inverses.**

a. Using the definition of matrix multiplication (in sum notation), prove that right-multiplication of an $n \times m$ matrix times the $(m \times m)$ identity matrix always returns the original matrix. Do the same for left-multiplication by the $(n \times n)$ identity matrix.

b. Consider the 2×2 matrix

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Assume M is invertible, and let the inverse be written as

$$M^{-1} = \begin{bmatrix} e & f \\ g & h \end{bmatrix}.$$

Solve for e, f, g, h in terms of a, b, c, d, thus find the inverse matrix M^{-1}. What is the condition for M^{-1} to exist? In other words, what condition on a, b, c, d allows the entries of M^{-1} to be finite?

c. Suppose the row (c, d) in M is simply a constant multiple of the row (a, b) (in other words, $c = \beta a$ and $d = \beta b$, where β is any real number. Will M have an inverse?