Homework #1 STA410H1F/2102H1F

due Friday October 4, 2019 at 11:59pm

Instructions: Solutions to problems 1 and 2 are to be submitted via Quercus and only PDF files will be accepted. (Ideally, you should submit only one file.) You are strongly encouraged to do problems 3 and 4 but these are **not** to be submitted for grading.

1. In this problem, you will use the discrete cosine transform (DCT) to denoise an image. An R function to compute a two-dimensional DCT is available in the R package `dtt` – this package contains functions to compute a number of "trigonometric" transforms. The R function that will be used in this package is `mvdct`.

(a) In lecture, we defined matrix transforms for vectors of length n, focusing on families of matrices $\{A_n\}$ satisfying $A_n^T A_n = D_n$ where D_n is a diagonal matrix.
Suppose that Z is an $m \times n$ pixel image, which we can represent as an $m \times n$ matrix. Then using $\{A_n\}$, we can define a transform of Z as follows:

$$\widehat{Z} = A_m Z A_n^T$$

Given the $m \times n$ matrix $\widehat{Z}$, show that we can reconstruct the original image by $Z = D_m^{-1} A_m^T \widehat{Z} A_n D_n^{-1}$.

(b) To denoise an image using the DCT, we first compute $\widehat{Z}$ and then perform hard- or soft-thresholding to obtain a thresholded (or shrunken) transform $\widehat{Z}^*$ (which will typically be "sparser" than $\widehat{Z}$). Our hope is that the components of $\widehat{Z}$ corresponding to noise in the image are eliminated and so the denoised image can be obtained by applying the inverse transform to $\widehat{Z}^*$.
Using `mvdct`, write R functions to perform both hard- and soft-thresholding (dependent on a parameter λ). Your functions should *not* threshold the $(1, 1)$ component of the DCT matrix. (A simple R function to do hard thresholding will be provided on Quercus; this can be modified to do soft thresholding.)

(c) The file `boats.txt` contains a noisy 256×256 pixel grayscale image of sailboats. Its entries are numbers between 0 and 1 where 0 represents black and 1 represents white. The data can be read into R and displayed as an image as follows:

```
> boats <- matrix(scan("boats.txt"),ncol=256,byrow=T)
> image(boats, axes=F, col=grey(seq(0,1,length=256)))
```

Using your functions from part (b), try to denoise the image as best as possible. (This is quite subjective but try different methods and parameter values.)

Note: You should not expect your noise-reduced image to be a drastic improvement over noisy image; in fact, connaisseurs of pointillism may find prefer the noisy image. There are two issues here: First, we are applying noise reduction to the whole image rather than dividing the image into smaller sub-images and applying noise reduction to each sub-image. Second, even very good noise reduction tends to wash out some details, thereby rendering the noise-reduced image less visually appealing.

2. Suppose that U and V are independent Poisson random variables with means λ_u and λ_v. We then define $X = U + 2V$, which is said to have a Hermite distribution with parameters λ_u and λ_v. (The Hermite distribution is the distribution of a sum of two correlated Poisson random variables.)

(a) Show that the probability generating function of X is

$$g(s) = E(s^X) = \exp\left[\lambda_u(s-1) + \lambda_v(s^2-1)\right].$$

(b) The distribution of X can, in theory, be obtained exactly in closed-form with exact computation for given λ_u and λ_v somewhat more difficult. However, we can approximate the distribution very well by combining the exact probability generating with the discrete Fourier transform. The key to doing this is to find M such that $P(X \geq M)$ is very small so that we can use the discrete Fourier transform to compute (to a very good approximation) $P(X = x)$ for $x = 0, 1, \cdots, M-1$.
One approach to determining M is to use the probability generating function of X and Markov's inequality. Specifically, if $s > 1$ we have

$$P(X \geq M) = P(s^X \geq s^M) \leq \frac{E(s^X)}{s^M} = \frac{\exp\left[\lambda_u(s-1) + \lambda_v(s^2-1)\right]}{s^m}$$

Use this fact to show that for $P(X \geq M) \leq \epsilon$, we can take

$$M = \inf_{s>1} \frac{\lambda_u(s-1) + \lambda_v(s^2-1) - \ln(\epsilon)}{\ln(s)}$$

(c) Given M (which depends on ϵ), the algorithm for determining the distribution of X goes as follows:

1. Evaluate the probability generating function $g(s)$ at $s = s_k = \exp(-2\pi\iota k/M)$ for $k = 0, \cdots, M-1$; the values of s can be created in R as follows:

```
> s <- exp(-2*pi*1i*c(0:(M-1))/M)
```

2. Evaluate $P(X = x)$ by computing the inverse FFT of the sequence $\{g(s_k) : k = 0, \cdots, M-1\}$:

$$P(X = x) = \frac{1}{M} \sum_{k=0}^{M-1} \exp\left(2\pi\iota \frac{x}{M} k\right) g(s_k)$$

Write an R function to implement this algorithm where M is determined using the method in part (b) with $\epsilon = 10^{-5}$. Use this function to evaluate the distribution of X for the following two cases:

(i) $\lambda_u = 1$ and $\lambda_v = 5$;

(ii) $\lambda_u = 0.1$ and $\lambda_v = 2$.

Note that you do not need to evaluate the bound M with great precision; for example, a simple approach is to take a discrete set of points $\mathcal{S} = \{1 < s_1 < s_2 < \cdots < s_k\}$ and define

$$M = \min_{s \in \mathcal{S}} \frac{\lambda_u(s-1) + \lambda_v(s^2-1) - \ln(\epsilon)}{\ln(s)}$$

where $\delta = s_{i+1} - s_i$ and s_k are determined graphically (that is, by plotting the appropriate function) so that you are convinced that the value of M is close to the actual infimum.

Supplemental problems (not to hand in):

3. As noted in lecture, catastrophic cancellation in the subtraction $x - y$ can occur when x and y are subject to round-off error. Specifically, if $\text{fl}(x) = x(1+u)$ and $\text{fl}(y) = y(1+v)$ then

$$\text{fl}(x) - \text{fl}(y) = x - y + (xu - yv)$$

where the absolute error $|xu - yv|$ can be very large if both x and y are large; in some cases, this error may swamp the object we are trying to compute, namely $x - y$, particularly if $|x - y|$ is relatively small compared to $|x|$ and $|y|$. For example, if we compute the sample variance using the right hand side of the identity

$$\sum_{i=1}^{n} (x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2, \tag{1}$$

a combination of round-off errors from the summations and catastrophic cancellation in the subtraction may result in the computation of a negative sample variance! (In older versions of Microsoft Excel, certain statistical calculations were prone to this unpleasant phenomenon.) In this problem, we will consider two algorithms for computing the sample variances that avoid this catastrophic cancellation. Both are "one pass" algorithms, in the sense that we only need to cycle once through the data (as is the case if we use the right hand side of (1)); to use the left hand side of (1), we need two passes since we need to first compute $\bar{x}$ before

computing the sum on the left hand side of (1). In parts (a) and (b) below, define $\bar{x}_k$ be the sample mean of $x_1, \cdots, x_k$ and note that

$$\bar{x}_{k+1} = \frac{k}{k+1}\bar{x}_k + \frac{1}{k+1}x_{k+1}$$

with $\bar{x} = \bar{x}_n$.

(a) Show that $\sum_{i=1}^{n}(x_i - \bar{x})^2$ can be computed using the recursion

$$\sum_{i=1}^{k+1}(x_i - \bar{x}_{k+1})^2 = \sum_{i=1}^{k}(x_i - \bar{x}_k)^2 + \frac{k}{k+1}(x_{k+1} - \bar{x}_k)^2$$

for $k = 1, \cdots, n-1$. (This is known as West's algorithm.)

(b) A somewhat simpler one-pass method replaces $\bar{x}$ by some estimate x_0 and then corrects for the error in estimation. Specifically, if x_0 is an arbitrary number, show that

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n}(x_i - x_0)^2 - n(x_0 - \bar{x})^2.$$

(c) The key in using the formula in part (b) is to choose x_0 to avoid catastrophic cancellation, that is, x_0 should be close to $\bar{x}$. How might you choose x_0 (without first computing $\bar{x}$) to minimize the possibility of catastrophic cancellation? Ideally, x_0 should be calculated using $o(n)$ operations.

(An interesting paper on computational algorithms for computing the variance is "Algorithms for computing the sample variance: analysis and recommendations" by Chan, Golub, and LeVeque; this paper is available on Quercus.)

4. (a) Suppose that A, B, C, and D are matrices so that AC and BD are both well-defined. Show that

$$(AC) \otimes (BD) = (A \otimes B)(C \otimes D)$$

(Hint: This is easier than it looks — the key is to start with the right hand side of the identity.)

(b) Use the result of part (a) to show that

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$$

for invertible matrices A and B.

(c) Suppose that H_{2^k} is a $2^k \times 2^k$ Hadamard matrix. Prove the claim given in lecture:

$$H_{2^k} = \prod_{j=1}^{k}(I_{2^{j-1}} \otimes H_2 \otimes I_{2^{k-j}})$$

(Hint: Use induction, starting with the fact that the identity holds trivially for $k = 1$.)