Advanced Computational Statistics (STAT 7730)
Homework 6

This homework is due by 11:30 AM on Wednesday, October 30. Be sure to explain your reasoning in your solutions, label your figures and attach the code at the end. When you submit your homework, you must bring it (hard copy) to our class.

1. (20 pts) Define the function

$$h(\boldsymbol{x}) = \left|\sin\left(2\pi x_1 \sum_{i=1}^{100} x_i\right)\right| \left(\cos\left(2\pi x_2 \sum_{j=1}^{100} x_j^2\right)\right)^2$$

for $\boldsymbol{x} = (x_1, \ldots, x_{100}) \in \mathbb{R}^{100}$.

(a) Compute an approximate 95% confidence interval for the value of

$$\int_{[0,1]^{100}} h(\boldsymbol{x})\, d\boldsymbol{x}$$

The interval should be no wider than 0.01.

(b) Compute an approximate 95% confidence interval for the value of

$$\int_{[0,1.05]^{100}} h(\boldsymbol{x})\, d\boldsymbol{x}$$

The interval should be no wider than 1. Your answer for part (b) should be a lot larger than for part (a). Why?

(c) Compute an approximate 95% confidence interval for the value of

$$\int_{[0,1]^{100}\cap B} h(\boldsymbol{x})\, d\boldsymbol{x}$$

where B is the ball of radius 6 centered at $(0, 0, \cdots, 0)$ in $\mathbb{R}^{100}$. The interval should be no wider than 0.01.

(d) What goes wrong if you try to do (c) using the ball of radius 4?

2. (16 pts) Define the function

$$h(x, y, z) = \begin{cases} z\left(\cos(3xy)\right)^2 e^{-z(x^2+y^2)} & \text{if } 0 \le \sqrt{x^2+y^2} \le \frac{1}{2} \\ z e^{xy} e^{-z(x^2+y^2)} & \text{if } \frac{1}{2} < \sqrt{x^2+y^2} \le 1 \\ 0 & \text{if } 1 < \sqrt{x^2+y^2} \end{cases}$$

for $x, y, z \in \mathbb{R}$.

(a) Compute an approximate 95% confidence interval for the value of

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x, y, 1)\, dx\, dy$$

The interval should be no wider than 0.05.

(b) Compute an approximate 95% confidence interval for the value of

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x, y, 50)\, dx\, dy$$

Use importance sampling. The interval should be no wider than 0.0001.

3. (20 pts) *Image Restoration.* The file "Wedding.zip" on Carmen contains three 540×360 images from a wedding scene: an original gray-level image ("Wedding"), a thresholded version of the original, set to ±1 ("Binary_Wedding"), and a version of Binary_Wedding that has been corrupted by adding iid Gaussian noise with mean $\mu = 0$ and standard deviation $\sigma = 1.5$ ("Noisy_Binary_Wedding"). The original gray-level image is included for reference, but isn't used in this experiment. The idea is to recover the binary image from the corrupted image.

   The model for the (latent) binary image is the Ising model:

$$p(x) = \frac{1}{Z}\exp\{\frac{1}{Temp}\sum_{<s,t>} x_s x_t\} \quad x_s \in \{-1, 1\}\ \forall s \in S$$

   where S is the 540×360 square lattice, and where $\sum_{<s,t>}$ means the sum over all pairs of neighboring sites, meaning all pairs of horizontal and vertical neighbors. Thus the neighbors of an interior pixel (r, c) are $(r+1, c)$, $(r-1, c)$, $(r, c+1)$, and $(r, c-1)$. Each pair only appears once in the summation. $Temp$ plays the role of temperature, so that the model strongly encourages neighboring pixels to have the same value at low temperatures, and only weakly encourages this at high temperatures.

   At each of temperatures $Temp = 0.5,\ 1.0$, and 1.5 use the Gibbs sampler to draw an approximate sample from the **"conditional"** distribution on x given the corrupted image, 'y'. Display the three samples, along with both the uncorrupted binary image and the corrupted image. Comment on the different restorations. (Notice that the parameters, the standard error of the added noise and the temperature of the Ising model, could be estimated directly from the noisy image using EM or MM Algorithms.)

   In running the sampler, each pixel will need to be visited enough times to have a reasonable chance of being close to equilibrium.

   You can run a typical Gibbs sampler but I recommend "checkerboard" sampling, which makes for very efficient code in this and related restoration problems. The idea is to label the pixels as "red" or "black" in a checkerboard pattern. Notice that under the latent model, $p(x)$, the red pixels are conditionally independent of each other given the black pixels, and vice versa. This means that all of the red pixels can be updated *simultaneously*, followed by simultaneous updating of all of the black pixels. (Larger neighborhoods in the latent structure require more "colors," but the same trick of replacing large blocks of conditionally independent pixels still applies.) If you use arrays, and avoid loops over array elements, the code can be quite fast. I could easily do 100 replacements of each pixel (i.e. 100 iterations of red/black replacements) in a few seconds. But 10 iterations of red/black replacements is probably enough to get a good sense of what the samples should look like. (The noisy image amounts to an "external field" in the posterior distribution, substantially diminishing the long-range correlations that slow convergence.)

   A convenient way to handle boundaries is to pad the lattice of ±1 values with a boundary of zeros. Then all pixels can be treated as "interior" and your code will not need to distinguish corner and boundary pixels from interior pixels.