Math 354 Computation Project

Due on Wednesday, April 23

• **Objective** In this project, we are going to experiment some *denoising* technique using the heat kernel. In problem 1, you will recover a segment of music from some noisy sound clip while in Problem 2, a noisy picture is being processed.

• **Heuristics** Convolving the heat kernel with initial data yields the solution to the heat equation $u_t = k\Delta u$ on an infinite line (or 2-D plane or 3-D space).

$$u(x,t) = K(x,t) * f(x) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{4\pi kt}} e^{\frac{-y^2}{4kt}} f(x-y)\, dy,$$

or, equivalently,

$$\hat{u}(\xi,t) = \hat{K}(\xi,t)\hat{f}(\xi) = e^{-\xi^2 kt}\hat{f}(\xi).$$

The multiplication with $\hat{K}(\xi,t)$ has the effect of *damping/filtering out* high frequency components since $\hat{K}$ decays rather quickly as $|\xi| \to \infty$. Moreover, as time t increases, the decay rate gets faster and faster (**why?**) so that, consequently, the damping effect gets more and more significant. This is also considered as a filtering process although the high frequency part is only partially filtered out.

In signal processing, noise – and more generally, nonsmoothness – can often be (roughly) regarded as high frequency components. Applying the heat kernel damps/filters out this high frequency part and therefore smooth out the noise. The variable t now becomes a scaling parameter that indicates the significance of the damping/smoothing effect.

Remark: the name "diffusion process" is often seen in this context since the heat equation also serves as a model for,e.g., diffusion of chemicals in a medium.

• **Digitization** A computer program can only deal with finite and discrete data. Therefore, in order to computerize the above argument, we replace the infinite integral with a finite sum, resulting in the Discrete Fourier Transform (DFT) and discrete convolution. For simplicity, we will not emphasize the replacement of continuum with digital data but rather focus on the commonality such as frequency, filtering, etc.

• **Matlab** First of all, it is always a good habit to create a working directory to store your data and program. Correspondingly, change the value of "current directory" located on the top of the main window.

To plot an 1-D array, use routine $plot(...)$. When several graphs are to be shown within one window, use

$$subplot(number\_of\_coloums, number\_of\_rows, location\_of\_subwindow)$$

preceding the graphing routine. It needs to be called repeatedly for showing graphs at different subwindows. Type *help subplot* for more information.

To compute the convolution of two 1-D arrays, use $conv(... , ...)$. For 2-D matrices, use $conv2(... , ...)$. Notice that matrix convolution results in a matrix larger than any of the input ones. Use $conv2(A, B, 'same')$ to keep only the central part of $conv2(A, B)$ that has the same size as the <u>first</u> input matrix A.

To load sound clips, use $variable = wavread('filename')$. The variable is either an N-by-1 array or an N-by-2 matrix depending on whether the data is monophonic or stereophonic. To play sound clips, use $soundsc(variable)$. In this project, we only deal with monophonic data.

Likewise, for images, use $imread('filename')$ to load image data which can be 2-D matrices for grayscale pictures or 3-D matrices for color pictures. To show images, use $imshow(variable)$. In this project, we only deal with grayscale data and prefer to use $imshow(variable, [0, 255])$ for proper display.

For simplicity, the following predefined routines are available online. Store them in your working directory.

—— $plotfft(...)$. This routine has two forms. When there is one input argument, it will plot the magnitudes of the DFT of the input. When there is an additional argument which should be of the form $start\_index : end\_index$, it will plot the magnitudes of the DFT in the frequency range specified by $start\_index : end\_index$.

—— $K1(...)$ and $K2(...)$. These two generates the heat kernel in 1-D and 2-D. Of course, the output is always discrete, i.e., arrays and matrices. The only parameter they take is t which is the time in heat flow and the scaling parameter in signal processing.

Problem 1 (denoising of sound). A sound file "music.wav" is given online. Denote the original sound data as $x0$. Denoise $x0$ using the 1-D heat kernel. Try parameter values $t = 0.1$ and $t = 1$. Denote the processed data as $x1$ and $x2$ respectively. Listen to the playbacks of $x0$, $x1$, $x2$.

Turn in the following items,

a) Matlab code;

b) Describe the acoustic differences among $x0$, $x1$ and $x2$.

c) With the help of $subplot(2, 1, location\_of\_subwindow)$ and $plotfft(...)$, show the spectra of $x0$ and $x2$ in two subwindows. Namely, show the DFT of $x0$ and $x2$. Explain the observed symmetry using the result from HW8, Problem 6.

d) Plot only the high frequency part of DFT of $x0$ and $x2$. Discuss the difference and explain it using the heat kernel. Here the high frequency part is located around the center of the spectra instead of at the right end.

Problem 2 (denoising of image). A image file "picture.jpg" is given online. Denote it as $y0$. Denoise $y0$ using the 2-D heat kernel. Experiment different values of t,e.g. $t = 1, 2, 3$. Denote them as $y1, y2, y3$.

Turn in the following items:

a) Matlab code;

b) The original picture $y0$ and 3 denoised ones $y1, y2, y3$ with different t values. Discuss the relation of parameter t and the quality of the outcome.

c) Take the difference between the original picture and the most <u>blurred</u> one. Display it using,e.g. $imshow(doubel(y0) - y3, [\,])$ and make comments on the output. Here $double(y0)$ works as data type conversion. The argument [ ] helps dramatize the image.

d) Combining our observations in b) c), we should find that the heat kernel filters out not only the noise but also some details germane to the original picture. What is this lost part of information?

HAVE FUN !