An Exercise in Image Processing with Heat Equation in MATLAB

Jeremy Lerner
Department of Applied Mathematics and Statistics
Stony Brook University

August 4, 2014

Contents

Part I
Linear Heat Equation with Finite Differences

1 Black and White

The original image is a high quality black and white picture of my parents' dog, Ben, as seen in figure 1

The image is broken down in Matlab as an array of values between 0 and 255, where 0 represents black and 255 represents white.

Using the two dimensional heat equation, equation 1.

$$\frac{\partial U}{\partial t} = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \tag{1}$$

to operate on the image. The original image, I_0, is successively smoothed. That is, heat equation is applied to the image. Using centered finite difference for the spatial derivatives and a forward difference for the temporal derivative, where $U^n_{i,j}$ is the value in the i^{th} row, j^{th} column at the n^{th} time step. Heat equation can be approximated by equation 2.

$$\frac{U^{n+1}_{i,j} - U^n_{i,j}}{\Delta t} = \frac{U^n_{i+1,j} - 2U^n_{i,j} + U^n_{i-1,j}}{\Delta x^2} + \frac{U^n_{i,j+1} - 2U^n_{i,j} + U^n_{i,j-1}}{\Delta y^2} \tag{2}$$

Which can be rearranged to solve for the updated values, as in equation 3.

$$U^{n+1}_{i,j} = U^n_{i,j} + \frac{\Delta t}{\Delta x^2}\left(\frac{U^n_{i+1,j} - 2U^n_{i,j} + U^n_{i-1,j}}{\Delta x^2}\right) + \frac{\Delta t}{\Delta x^2}\left(\frac{U^n_{i,j+1} - 2U^n_{i,j} + U^n_{i,j-1}}{\Delta y^2}\right) \tag{3}$$

Using $\Delta t = 0.2$ and $\Delta x = \Delta y = 1$, the original image blurs over time, as seen in figures 2, 3 and 4.

Figure 1: Original Image of Ben, I_0

Figure 2: I_{10}

Figure 3: I_{100}

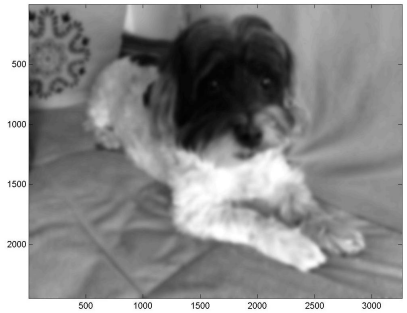
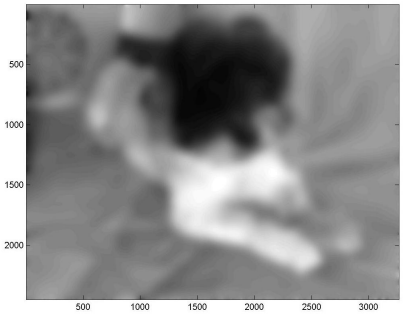


Figure 4: I_{1000}



2 Color

The original image is a color version of the high quality picture of my parents' dog, Ben, as seen in figure 5.

Color images are represented as a matrix of RGB triples. In Matlab, images are represented as a three dimensional matrix and heat equation was applied to each matrix independently.

Using $\Delta t = 0.2$ and $\Delta x = \Delta y = 1$, the original image blurs over time, as seen in figures 6, 7 and 8.

Figure 5: Original Color Image, I_0

Figure 6: I_{10}

Figure 7: I_{100}

Figure 8: I_{1000}

Part II
Convolution

3 Black and White

Applying a Gaussian low pass filter to the image, that is convolving it using the convolution matrix in equation 4, where the x and y matrices in Matlab are given by equations 5 and 6 respectively. The Matlab code is

```
[x,y]=meshgrid(1:n,1:n)
```

$$g(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\left(x - \lceil \frac{p}{2} \rceil\right)^2 + \left(y - \lceil \frac{p}{2} \rceil\right)^2}{2\sigma^2} \right) \tag{4}$$

$$x(i,j) = j \tag{5}$$

$$y(i,j) = i \tag{6}$$

The Gaussian given in equation 4 is centered at $\lceil \frac{p}{2} \rceil$, where p is the size of the square convolution matrix. Equation 4 is the heat kernel and is related to the solution to heat equation (1) by equations 8 and 9 (note $\bigotimes$ is the symbol for mathematic convolution). Where σ is related to t by equation 7. Using equation 7, the same final times as above, 10, 100 and 1000 were used to find σ, as seen in figures 10, 11 and 12.

$$\sigma = \sqrt{(2t)} \tag{7}$$

$$u(x,y,t) = u(x,y,t=0) \bigotimes K = \int_{\Omega} u(x,y,t=0) \cdot K(x,y,x_2,y_2,t)\, dx_2 dy_2 \tag{8}$$

$$K(x,y,x_2,y_2,t) = \frac{1}{4\pi t} e^{-\frac{(x-x_2)^2+(y-y_2)^2}{4t}} \tag{9}$$

Figure 9: Original Image, I_0

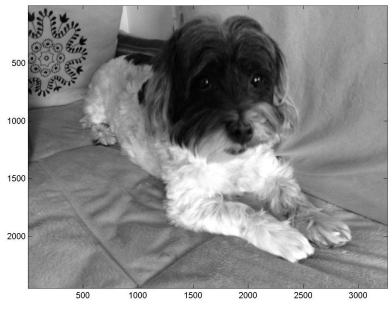
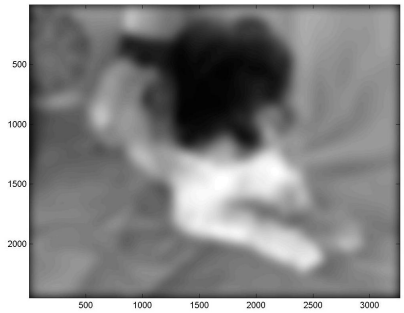


Figure 10: I_{10}

Figure 11: I_{100}

Figure 12: I_{1000}



The size of the convolution matrix became a relevant variable, because when the convolution matrix was too small, for example 9 by 9, the image would blur, but only slightly, as seen in figures 14, 15 and 16. For the image of Ben, which is 2448 by 3264 pixels, a size that worked well was 201 by 201, which was used in the images above. The tradeoff is that the convolution takes more time at each step for larger convolution matrices.

Figure 13: Original Image, I_0

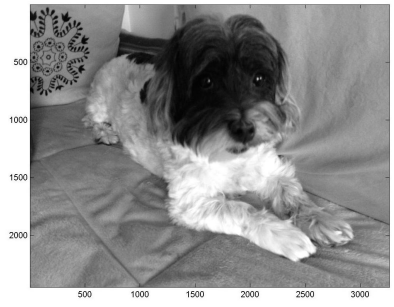


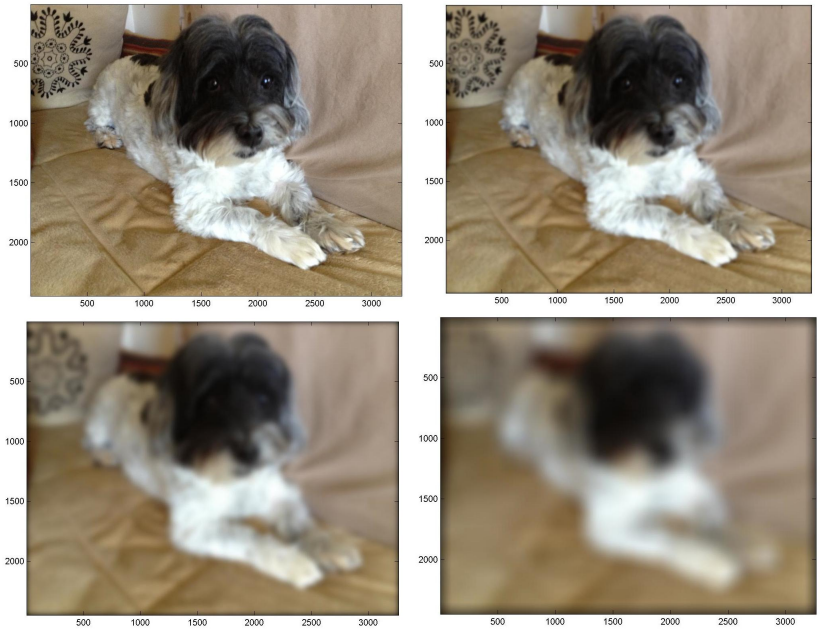
Figure 14: I_{10}

Figure 15: I_{100}

Figure 16: I_{1000}

4 Color

Using a convolution matrix of size 201 by 201, the original image, figure 4 can be smoothed resulting in figures 4, 23 and 4.



Part III
Non Linear Heat Equation

5 Black and White

Equation 10 can also be used to smooth an image.

$$\frac{\partial U}{\partial t} = \operatorname{div}\left(\frac{\nabla U}{\|\nabla U\|}\right) \cdot \|\nabla U\| \tag{10}$$

Equation 10 can be rewritten as equation 12, as shown in the derivation in 11.

$$\begin{aligned}
\frac{\partial U}{\partial t} = \operatorname{div}\left(\frac{\nabla U}{\|\nabla U\|}\right) \cdot \|\nabla U\| &= \operatorname{div}\left(\frac{\langle U_x, U_y\rangle}{\left\|U_x\hat{i} + U_y\hat{j}\right\|}\right)\left\|U_x\hat{i} + U_y\hat{j}\right\| \\
&= \operatorname{div}\left(\frac{\langle U_x, U_y\rangle}{\sqrt{U_x^2 + U_y^2}}\right)\sqrt{U_x^2 + U_y^2} \\
&= \frac{\partial}{\partial x}\left(\frac{U_x}{\sqrt{U_x^2 + U_y^2}}\right)\sqrt{U_x^2 + U_y^2} + \frac{\partial}{\partial y}\left(\frac{U_y}{\sqrt{U_x^2 + U_y^2}}\right)\sqrt{U_x^2 + U_y^2} \\
&= \frac{U_{xx}\sqrt{U_x^2 + U_y^2} - U_x\left(U_xU_{xx} + U_yU_{xy}\right)\left(U_x^2 + U_y^2\right)^{-\frac{1}{2}}}{\sqrt{U_x^2 + U_y^2}} \\
&\quad + \frac{U_{yy}\sqrt{U_x^2 + U_y^2} - U_y\left(U_xU_{xy} + U_yU_{yy}\right)\left(U_x^2 + U_y^2\right)^{-\frac{1}{2}}}{\sqrt{U_x^2 + U_y^2}} \\
&= \frac{U_{xx}\left(U_x^2 + U_y^2\right) - U_x\left(U_xU_{xx} + U_yU_{xy}\right) + U_{yy}\left(U_x^2 + U_y^2\right) - U_y\left(U_xU_{xy} + U_yU_{yy}\right)}{U_x^2 + U_y^2} \\
&= \frac{1}{U_x^2 + U_y^2}\left[U_{xx}U_y^2 - 2U_xU_yU_{xy} + U_x^2U_{yy}\right] \qquad (11)
\end{aligned}$$

$$\frac{\partial U}{\partial t} = \frac{1}{U_x^2 + U_y^2}\left[U_{xx}U_y^2 - 2U_xU_yU_{xy} + U_x^2U_{yy}\right] \tag{12}$$

Using centered finite differences, equation 10 can be approximated by equation 13.

$$\frac{U_{i,j}^{n+1} - U_{i,j}^{n}}{\Delta t} = \Bigg[\left(\frac{U_{i+1,j}^{n} - 2U_{i,j}^{n} + U_{i-1,j}^{n}}{\Delta x^2} \right) \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)^2 -$$
$$2 \left(\frac{U_{i+1,j+1} - U_{i-1,j+1} + U_{i-1,j-1} - U_{i+1,j-1}}{4\Delta x \Delta y} \right) \left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right) \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)$$
$$+ \left(\frac{U_{i,j+1}^{n} - 2U_{i,j}^{n} + U_{i,j-1}^{n}}{\Delta y^2} \right) \left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right)^2 \Bigg]$$
$$/ \left(\left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right)^2 + \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)^2 \right) \quad (13)$$

In Matlab, the denominator was potentially problematic, as it could have been near zero, so ϵ was added to the denominator, as in equation 14, where $\epsilon = 10^{-14}$, to avoid division by zero.

$$U_{i,j}^{n+1} = U_{i,j}^{n} + \Delta t \Bigg[\left(\frac{U_{i+1,j}^{n} - 2U_{i,j}^{n} + U_{i-1,j}^{n}}{\Delta x^2} \right) \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)^2 -$$
$$2 \left(\frac{U_{i+1,j+1} - U_{i-1,j+1} + U_{i-1,j-1} - U_{i+1,j-1}}{4\Delta x \Delta y} \right) \left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right) \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)$$
$$+ \left(\frac{U_{i,j+1}^{n} - 2U_{i,j}^{n} + U_{i,j-1}^{n}}{\Delta y^2} \right) \left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right)^2 \Bigg]$$
$$/ \left(\left(\frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \right)^2 + \left(\frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y} \right)^2 + \epsilon \right) \quad (14)$$

Using $\Delta t = 0.1$ and $\Delta x = \Delta y = 1$, the original image, figure 17, can be smoothed by operating on it with equation 14, resulting in figures 18, 19 and 20.

Figure 17: Original Image, I_0

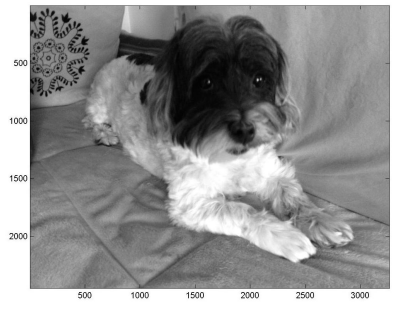
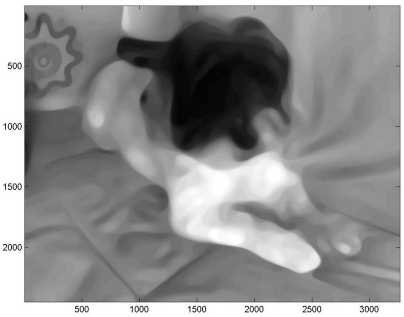


Figure 18: I_{10}

Figure 19: I_{100}

Figure 20: I_{1000}



6 Color

Using $\Delta t = 0.1$ and $\Delta x = \Delta y = 1$, the original image, figure 21 can be smoothed by operating on it with equation 13, resulting in figures 22, 23 and 24. Figure 24 has an end time of $t = 200$ instead of $t = 1000$, due to compounding error. In Matlab, the values in the matrices that represent a color image must be between 0 and 1. While in a black and white image, the values are between 0 and 255, so the compounding error in a black and white image is not as significant.

Figure 21: Original Image, I_0

Figure 22: I_{10}

Figure 23: I_{100}

Figure 24: I_{200}

Part IV
Comparisons

The tables below contains images blurred with multiple methods, first black and white, then color.

Figure 25: Original Image, I_0

Figure 26: I_{10}

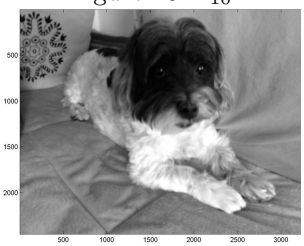
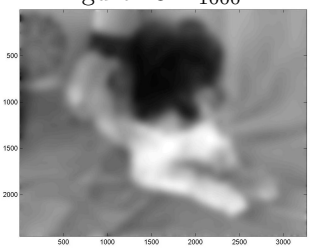


Figure 27: I_{100}

Figure 28: I_{1000}



Linear Heat Equation with Finite Difference Method

Figure 29: I_{10}

Figure 30: I_{100}

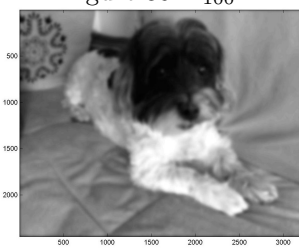
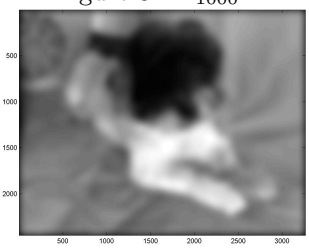


Figure 31: I_{1000}

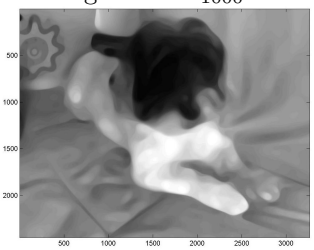


Convolution with a 201 by 201 matrix

Figure 32: I_{10}

Figure 33: I_{100}

Figure 34: I_{1000}



Non Linear Heat Equation with Finite Difference Method

Figure 35: Original Image, I_0

Figure 36: I_{10}

Figure 37: I_{100}

Figure 38: I_{1000}

Linear Heat Equation with Finite Difference Method

Figure 39: I_{10}

Figure 40: I_{100}

Figure 41: I_{1000}

Convolution with a 201 by 201 matrix

Figure 42: I_{10}

Figure 43: I_{100}

Figure 44: I_{200}

Non Linear Heat Equation with Finite Difference Method