Image denoising by solving a diffusion PDE using the FEM method

www.cs.ualberta.ca/~kpopuri/assign4/image_diffusion.pdf

Karteek Popuri and Martin Jägersand

Introduction

Please take a look at the ``helper code'' for this assignment at:
www.cs.ualberta.ca/~kpopuri/assign4/helper_code Helper code

A *digital image* is a collection of *pixels* arranged in a rectangular two dimensional (2D) array. For a gray-scale (black and white) image, we obtain a scalar intensity value at each of the pixel locations. These intensity values are usually quantized between 0 and 255 . Hence, a digital image corresponds to a matrix of discrete values in the range of $[0\ 255]$. For example, see Figure 1.

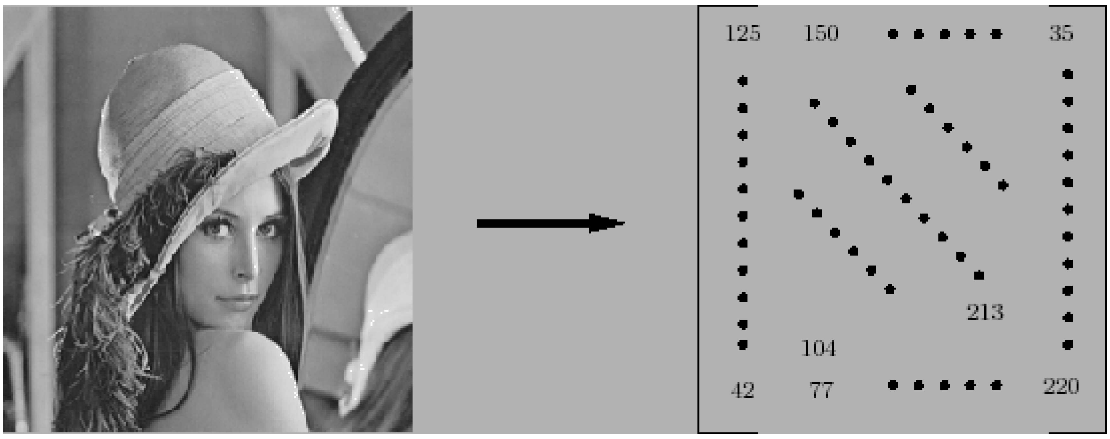


Figure: A digital image is essentially a matrix of integers in $[0\ 255]$.

Images captured of the real world objects (scenes) are prone to random fluctuations in the observed intensity values. This variation in the intensity values is refered to as *image noise*. It is an unavoidable by-product of the image capture process and it predominantly arises from the sensor and circuity of the digital camera. Figure 2 shows a noisy image.

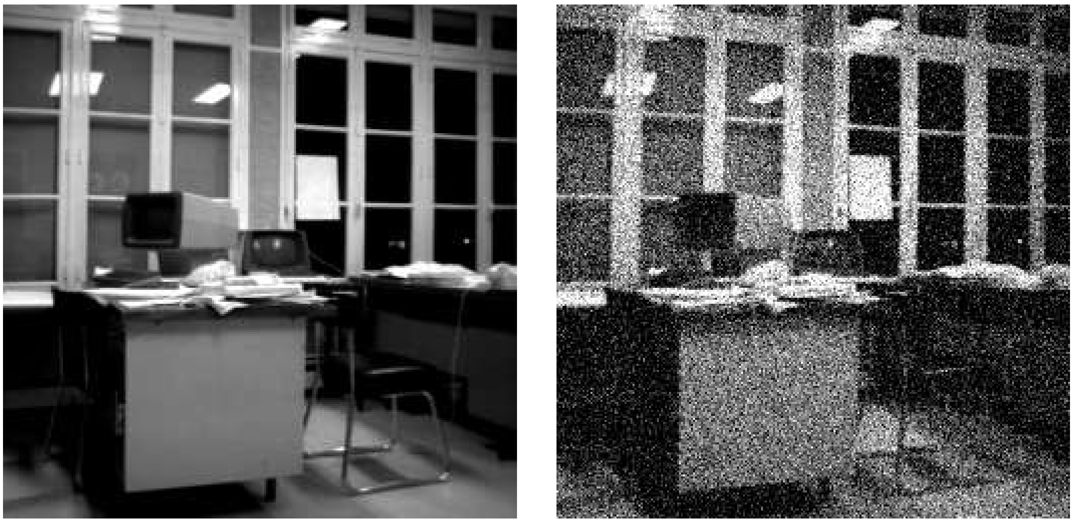


Figure 2: (left) The ``office'' image WITHOUT noise. (Right) The ``office'' image corrupted WITH noise.

Gaussian smoothing

The process of removing noise from an image is known as *noise reduction* or *denoising*. A standard denoising technique is the convolution (see section 5.1 for an explanation) of the image with a 2D Gaussian distribution. The zero mean 2D Gaussian distribution function is given in Figure 3:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

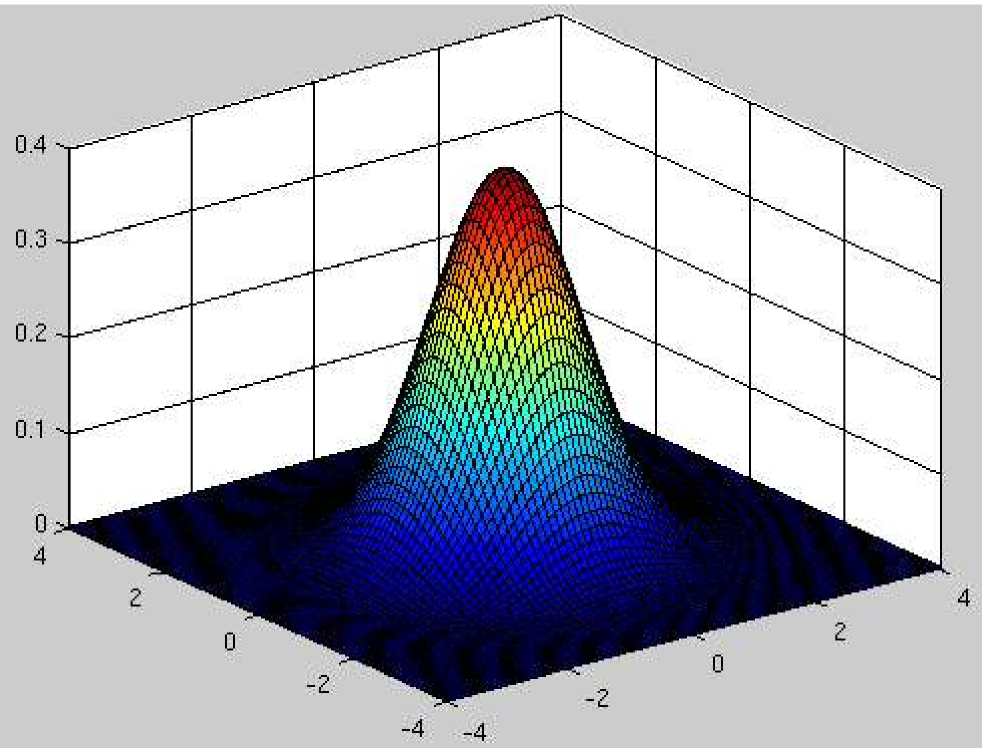
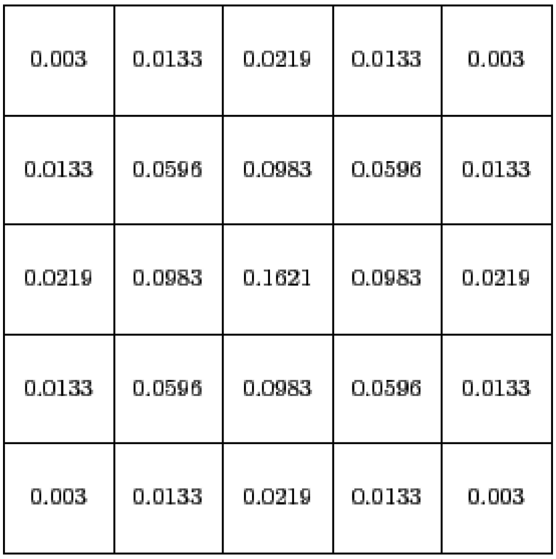


Figure 3: A zero mean 2D Gaussian with $\sigma = 1$.

In practice as the image contains discrete pixel locations, the Gaussian distribution needs to be approximated using a convolution kernel before the convolution operation can be performed. Figure 4 shows a 2D Gaussian convolution kernel.



0.003	0.0133	0.0219	0.0133	0.003
0.0133	0.0596	0.0983	0.0596	0.0133
0.0219	0.0983	0.1621	0.0983	0.0219
0.0133	0.0596	0.0983	0.0596	0.0133
0.003	0.0133	0.0219	0.0133	0.003

Figure 4: The 5×5 convolution kernel of a Gaussian with $\sigma = 1$.

As described in section 5.1, the Gaussian convolution operation basically replaces the intensity values of the pixels in the original noisy image with a weighted average of the intensity values of the pixels in their respective neighborhoods (defined by the size of the kernel). Thus, the Gaussian smoothing technique incorporates the intensity information from the neighboring pixels to reduce noise in the image. In fact, the idea of using neighborhood pixel intensity information for noise reduction is at the core of image denoising techniques in general.

Gaussian smoothing can be performed very easily using MATLAB as follows:

```
I = imread('office_noisy.png');  %load the noisy image

h = fspecial('gaussian',[3*ceil(sigma) 3*ceil(sigma)],sigma);
                                 %create the 3\sigma X
                                 %3\sigma Gaussian kernel with \sigma = 1

I_smooth = imfilter(I,h);        %perform Gaussian smoothing

figure;                          %display the images
subplot(1,2,1);imshow(I,[]);title('Noisy image');
subplot(1,2,2);imshow(I_smooth,[]);title('Smooth image')
```

Question 0 :

Setting the parameter $\sigma = \{0.5, 1, 2, 5, 10, 50\}$ respectively perform Gaussian smoothing on the input noisy ``office" image. Looking at the sequence of output images corresponding to the increasing σ values answer the following:

- How does the increase in parameter σ effect the noise reduction performance of Gaussian smoothing ? What happens to the edges (discontinuities) and other details in the image with the increase of σ ?
- Now can you guess why denoising by a Gaussian kernel is also called Gaussian *smoothing* (or in general denoising is also refered to as *smoothing*)?

Isotropic linear diffusion smoothing

In this problem, we consider a *linear* isotropic diffusion process (see section 5.2 for notes on diffusion) on an image domain for the task of denoising. The linear isotropic diffusion process can be described by:

$$\begin{aligned}\frac{\partial u}{\partial t} &= \operatorname{div}(d\,\nabla u)\\ u(x,y,0) &= I(x,y)\end{aligned} \qquad (1)$$

where d is a scalar constant diffusivity, $I(x,y)$ is the initial noisy image, $u(x,y,t)$ is the image obtained after a diffusion time t . Note that here $u(x,y,t)$ represents the evolving intensity distribution corresponding to the evolving concentration distribution $c(x,y,t)$ in section 5.2.

Question 1

Using the MATLAB PDE toolbox solve the diffusion PDE in equation 1 by the FEM method. Your initial condition is the noisy ``office'' image. Choose $d = 1$. Report the output images obtained after a diffusion time $t = \{1, 5, 10, 30, 100\}$. What type of boundary conditions should you choose here ?

Note: You should use a sufficiently refined mesh so as to avoid any ``triangular'' artifacts in the output image.

Now looking at the sequence of output images obtained from the above answer the following:

- Are the output images less noisy (``smoother'') than the input image ? In other words, is the diffusion process really performing noise reduction on the input image ?
- If your answer is yes to the above question, what is the effect of diffusion time on denoising ? What happens to the edges (discontinuities) and other details in the with increase in diffusion time ?

Question 2

Now solve the diffusion PDE in equation 1 using $d = \{1, 5, 10\}$ and compare the output images at $t = 10$. What is the effect of d on deniosing ?

Question 3

It can be proved that a unique solution exists for the PDE in equation 1 which is given by:

$$u(x,y,t) = \begin{cases} I(x,y) & (t = 0) \\ (G_{\sqrt{2t}} * I)(x,y) & (t > 0) \end{cases} \qquad (2)$$

where $G_{\sqrt{2t}}(x,y)$ is the Gaussian kernel. This proves that performing isotropic linear diffusion for a time t with $d = 1$ is exactly equivalent to performing Gaussian smoothing with a $\sigma = \sqrt{(2t)}$. Verify this fact using the noisy ``office'' image.

Isotropic non-linear diffusion smoothing

In this problem, we consider a *non-linear* isotropic diffusion process (see section 5.2 for notes on diffusion) on an image domain for the task of denoising. The non-linear isotropic diffusion process can be described by:

$$\begin{aligned}\frac{\partial u}{\partial t} &= \operatorname{div}(D\,\nabla u)\\ u(x,y,0) &= I(x,y)\end{aligned} \tag{3}$$

Here, D is not a constant but varies across the image domain. A popular choice is the Perona-Malik diffusivity which is given as :

$$D(x,y) = \frac{1}{1+\frac{|\nabla u(x,y)|^2}{\lambda^2}} \tag{4}$$

where λ is the contrast parameter. $\nabla u(x,y)$ is the gradient of the image at pixel (x,y)

Question 4

Using the MATLAB PDE toolbox solve the diffusion PDE in equation 3 by the FEM method. Your initial condition is the noisy ``office" image. Report the output images obtained after a diffusion time $t=\{1,5,10,30,100\}$. Use $\lambda = 0.5$. What type of boundary conditions should you choose here ?

Note: You should use a sufficiently refined mesh so as to avoid any ``triangular" artifacts in the output image.

Now looking at the sequence of output images obtained from the above answer the following:

- Are the output images less noisy (``smoother") than the input image ? In other words, is the diffusion process really performing noise reduction on the input image ?
- If your answer is yes to the above question, what is the effect of diffusion time on denoising ? What happens to the edges (discontinuities) and other details in the with increase in diffusion time ?
- Non-linear diffusion with a diffusivity like the Perona-Malik diffusivity is called ``edge-preserving" diffusion. This is because the value of the Perona-Malik diffusivity is low in the presence of edges, i.e. when $\nabla u$ is large $D(x,y)$ is small (see 4). For $\lambda = 0.5$ , compute the diffusivity $D(x,y)$ on the ``office" image WITHOUT noise in Figure 2 and visualize the diffusivity as a gray-scale image. What do you observe in this diffusivity ``image" ?
- Compare the edges in the output images obtained from non-linear diffusion with the corresponding output images obtained from linear diffusion in the previous problem. In which of the images are the edges better preserved ?

Question 5

Now solve the diffusion PDE in equation 1 using $\lambda = \{0.5,1,2,5,10\}$ and compare the output images at $t=10$. What is the effect of the contrast parameter λ on deniosing and edge preservation ?

(While preparing this assignment, I refered to Dr.Weickert`s notes from the course Differential equations in Image processing and Computer vision [1]. You can access the relevant lectures from here `www.cs.ualberta.ca/~kpopuri/dic/`.

Appendix

Convolution

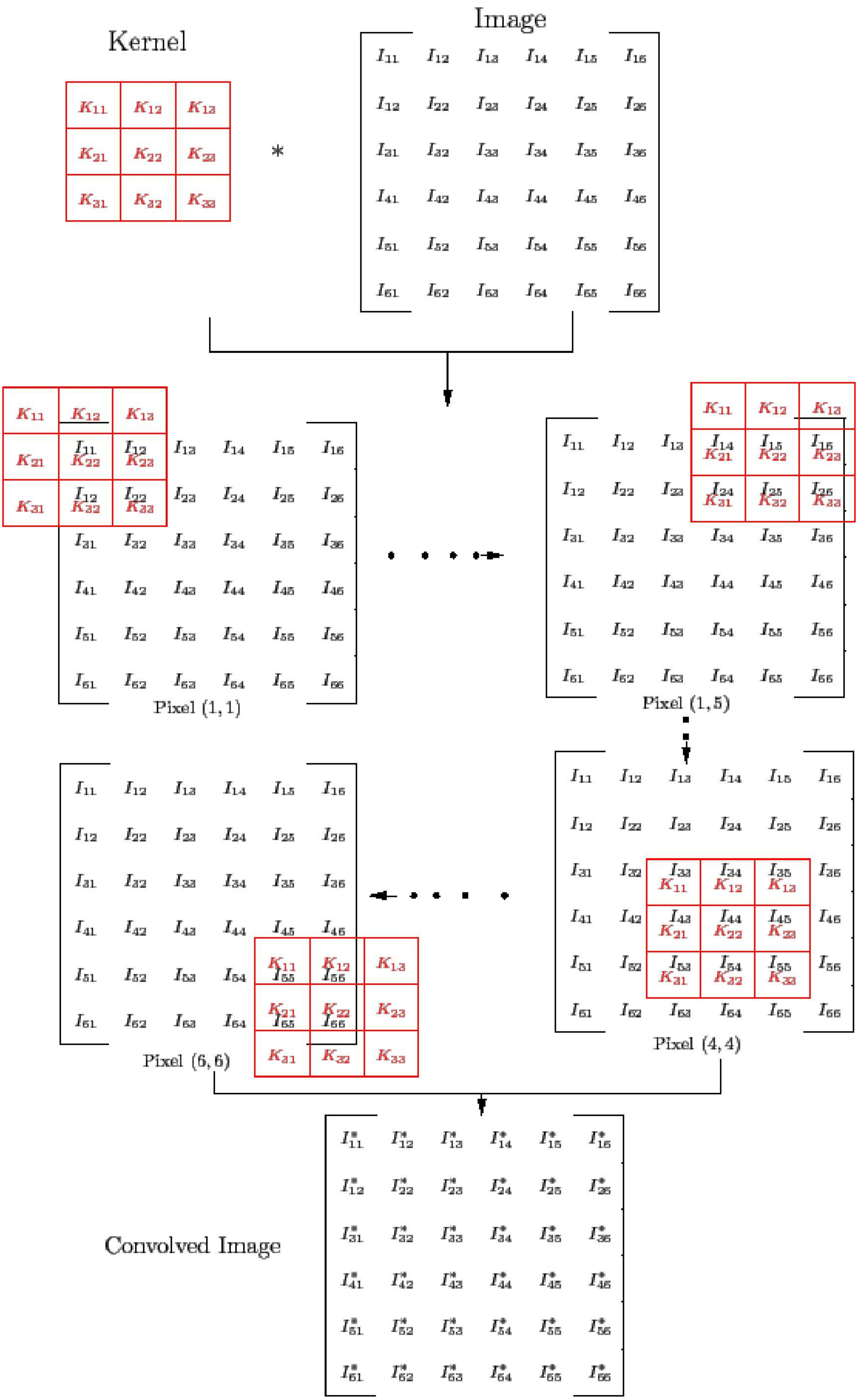


Figure 5: The convolution of a $6X6$ image with a $3X3$ kernel.

In the convolution operation illustrated in Figure 5, the intensity value I^*_{11} at the pixel $(1,1)$ in the convolved image is computed as follows:

$$\begin{aligned} I^*_{11} = &(K_{11} \times 0) + (K_{12} \times 0) + (K_{13} \times 0) + \\ &(K_{21} \times 0) + (K_{22} \times I_{11}) + (K_{23} \times I_{12}) + \\ &(K_{31} \times 0) + (K_{32} \times I_{21}) + (K_{33} \times I_{22}) \end{aligned}$$

Similarly, the intensity values I^*_{15} , I^*_{44} , I^*_{66} are computed as:

$$\begin{aligned} I^*_{15} = &(K_{11} \times 0) + (K_{12} \times 0) + (K_{13} \times 0) + \\ &(K_{21} \times I_{14}) + (K_{22} \times I_{15}) + (K_{23} \times I_{16}) + \\ &(K_{31} \times I_{24}) + (K_{32} \times I_{25}) + (K_{33} \times I_{26}) \\ I^*_{44} = &(K_{11} \times I_{33}) + (K_{12} \times I_{34}) + (K_{13} \times I_{35}) + \\ &(K_{21} \times I_{43}) + (K_{22} \times I_{44}) + (K_{23} \times I_{45}) + \\ &(K_{31} \times I_{53}) + (K_{32} \times I_{54}) + (K_{33} \times I_{55}) \\ I^*_{66} = &(K_{11} \times I_{55}) + (K_{12} \times I_{56}) + (K_{13} \times 0) + \\ &(K_{21} \times I_{65}) + (K_{22} \times I_{66}) + (K_{23} \times 0) + \\ &(K_{31} \times 0) + (K_{32} \times 0) + (K_{33} \times 0) \end{aligned}$$

In general, given a kernel of size $(2P+1) \times (2P+1)$ and an image of size $N \times M$, the intensity value I^*_{ij} at the pixel (i,j) in the convolved image is given by:

$$I^*_{ij} = \sum_{k=-P}^{P} \sum_{l=-P}^{P} K_{2P-k\,2P-l}\, I_{i-k\,j-l} \qquad \forall i \in \{1,2,\ldots,N\}\, j \in \{1,2,\ldots,M\}$$

Hence, the basic idea of the convolution operation is to replace the intensity value at a particular pixel by a weighted average of all the pixels in its $(2P+1) \times (2P+1)$ neighborhood.

Notes on Diffusion

Diffusion is the flow of molecules (mass) from a place of high concentration to a place of low concentration. Some examples of diffusion are: diffusion of perfume molecules from one part of the room to the other when a bottle of perfume is opened, the diffusion of tea molecules into the surrounding water when a tea bag is inserted into a cup of water. Diffusion processes strive to equilibrate the concentration differences in the system whilst preserving the total mass of the system.

Fick`s law of diffusion relates the diffusive flux to the concentration gradient:

$$\mathbf{j} = -D\,(\frac{\partial c}{\partial x} + \frac{\partial c}{\partial y})$$

where $\mathbf{j}$ is the diffusive flux, $c(x,y,t)$ is the concentration and D is the diffusivity. The diffusivity D is a scalar in the case of *isotropic* diffusion (considered in this assignment) and it is a matrix in the case of *anisotropic* diffusion.

The principle of conservation of mass states that:

$$\frac{\partial c}{\partial t} = -\mathrm{div}\,\mathbf{j}$$

Combining the above two equations we obtain a the diffusion PDE:

$$\frac{\partial c}{\partial t} = \mathrm{div}(D\nabla c)$$

In the case of isotropic diffusion we have following two cases:

Linear diffusion
: D is a constant, i.e. $D(x,y) = d$, a scalar constant for all x,y in the problem domain.

Non-Linear diffusion
: $D \equiv D(x,y)$ is scalar but varies over the problem domain.

Bibliography

1
: Dr. Joachim Weickert.
Differential equtions in image processing and computer vision, lecture notes apr-aug, 2009.
`http://www.mia.uni-saarland.de/Teaching/dic09.shtml.`

About this document ...

Image denoising by solving a diffusion PDE using the FEM method
`www.cs.ualberta.ca/~kpopuri/assign4/image_diffusion.pdf`

This document was generated using the **LaTeX**2`HTML` translator Version 2002-2-1 (1.71)

The command line arguments were:
latex2html `-split 0 image_diffusion`

The translation was initiated by Karteek Popuri on 2010-04-08

Karteek Popuri 2010-04-08