Anisotropic Diffusion

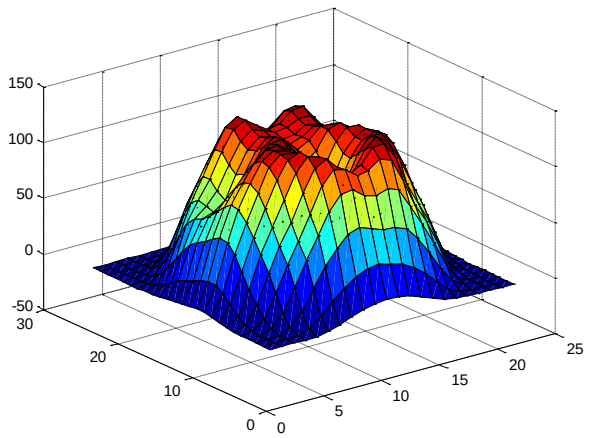
L 24

Goal: To find out how edges can be used to help in image denoising.

First thing's first… what is "diffusion" in the context of image processing?

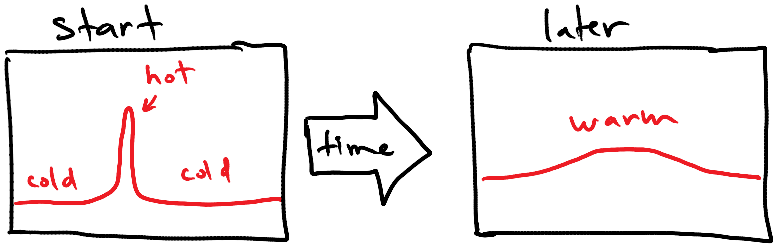
Think of an image as a function indicating the temperature distribution throughout a material.
(eg. temp. of a metal plate).

```
f = imread('t1.jpg');
f = double(f(:,:,1));
surf(MyGaussianBlur(imresize(f,0.1),1));
```



It's not hard to imagine the spreading of that heat over time.

eg. in 1D



Heat diffuses over time. In particular, its time dynamics are modelled by the "heat equation", a partial differential equation (PDE),

$u(x,t)$ is the temperature at location x at time t

equation (PDE),

$u(x,t)$ is the temperature at location x at time t.

$$\frac{\partial u}{\partial t} = c\,\Delta u$$

c: diffusion rate

$$\Delta u \equiv \nabla \cdot \nabla u$$

∇ is the gradient operator

$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$ in 2D

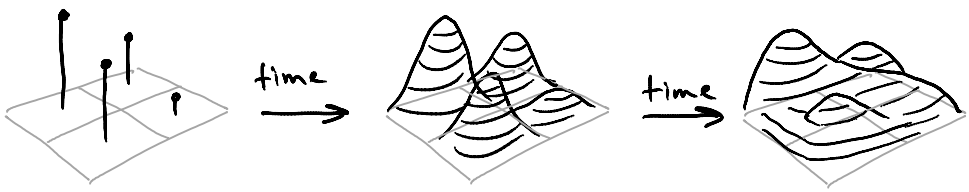
$$= \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right) \cdot \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right)$$

$$= \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

As it turns out, if you start with a single spike of heat (and no heat elsewhere), then the solution is a time progression of Gaussian functions, with larger and larger σ as time goes on.

Any heat distribution can be represented as a combination of heat spikes. Since the differential operators are linear, the evolution of the entire heat distribution is the superposition of a bunch of spreading Gaussian functions.

eg.



This is equivalent to convolving with a Gaussian kernel.

$\Rightarrow$ Diffusion = Blurring

Hence, one way to blur an image is to let it evolve using the heat equation.

$$\frac{\partial f}{\partial t} = c\,\Delta f$$

forward differencing

$$\frac{f_{mn}^{t+1} - f_{mn}^{t}}{\Delta t}$$

$$\therefore f_{mn}^{t+1} = f_{mn}^{t} + \Delta t\, c\, \Delta f$$

$$\frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial y}\right)$$

$$= \frac{\partial}{\partial x}\left[\frac{f_{m+1,n}^{t} - f_{m-1,n}^{t}}{2\Delta x}\right] + \frac{\partial}{\partial y}\left[\frac{f_{m,n+1}^{t} - f_{m,n-1}}{2\Delta y}\right]$$

$$= \frac{\left[\frac{f_{m+2,n}^{t} - f_{mn}^{t}}{2\Delta x}\right] - \left[\frac{f_{m+1,n}^{t} - f_{m-1,n}^{t}}{2\Delta x}\right]}{2\Delta x} + \cdots$$

$$= \frac{f_{m+2,n}^{t} - 2f_{mn}^{t} + f_{m-2,n}^{t}}{(2\Delta x)^2} + \frac{f_{m,n+2}^{t} - 2f_{mn}^{t} + f_{m,n-2}^{t}}{(2\Delta y)^2}$$

Apply central difference approx. twice in sequence.
This is the hard way.

Alternatively, we could go back to Taylor's theorem:

$$f(x+\Delta x) = f(x) + \frac{\partial f}{\partial x}\Delta x + \frac{\partial^2 f}{\partial x^2}\frac{\Delta x^2}{2} + O(\Delta x^3)$$

$$+\quad f(x-\Delta x) = f(x) - \frac{\partial f}{\partial x}\Delta x + \frac{\partial^2 f}{\partial x^2}\frac{\Delta x^2}{2} + O(\Delta x^3)$$

$$f(x+\Delta x) + f(x-\Delta x) = 2f(x) + 2\,\frac{\partial^2 f}{\partial x^2}\frac{\Delta x^2}{2} + O(\Delta x^3)$$

$$\Rightarrow \frac{\partial^2 f}{\partial x^2} \approx \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2}$$

The same as applying central differencing twice, but using a Δx that is half as big.

Anisotropic Diffusion

The diffusion process described above is isotropic.

American Heritage Dictionary – *Cite This Source*
i·so·tro·pic (ī'sə-trō'pĭk, -trŏp'ĭk) Pronunciation Key
adj. Identical in all directions; invariant with respect to direction.

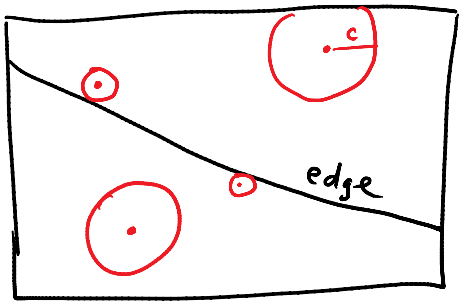
One of the problems is that, while it filters out noise, it also blurs the edges of the image, thereby reducing the resolution. To get around this, we can make the rate of diffusion vary spatially, so that diffusion is slower across edges.

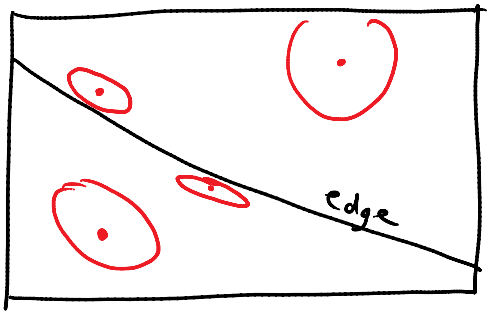
The idea behind anisotropic diffusion (AD) is to let the diffusion-rate constant *c* be a function of space so that

$$c(x) = \begin{cases} \text{large in homogeneous regions} \\ \text{small near edges} \end{cases}$$

We have two ways of using this spatially-varying *c*:

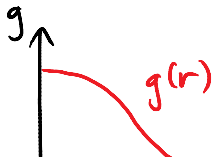
$$\frac{\partial f}{\partial t} = c(x)\,\Delta f \qquad \frac{\partial f}{\partial t} = \nabla \cdot (c(x)\,\nabla f)$$



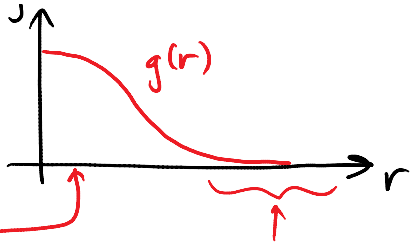


We use the gradient magnitude to construct $c(x)$. In particular, we will use

$$c(x) = g(\|\nabla f\|)$$



$$c(x) = g(\|\nabla f\|)$$



When $\|\nabla f\|$ is small, then diffusion is fast.

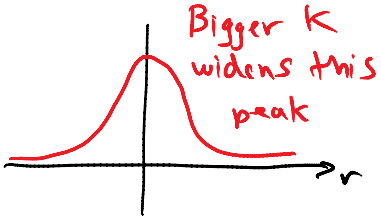
When $\|\nabla f\|$ is large (near an edge), then diffusion is slow.

Replacing $c(x)$ with $g(\|\nabla f\|)$,

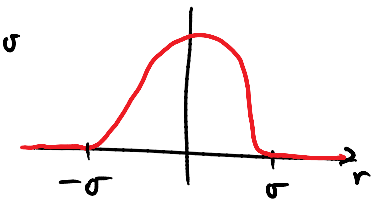
$$\frac{\partial f}{\partial t} = g(\|\nabla f\|)\Delta f \quad \text{or} \quad \frac{\partial f}{\partial t} = \nabla \cdot \left(g(\|\nabla f\|) \nabla f \right)$$

Some common g functions:

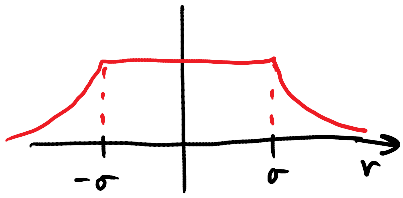
Perona & Malik's Lorentzian function $g(r) = \dfrac{1}{1 + \frac{r^2}{k^2}}$



Tukey's Biweight $g(r) = \begin{cases} \frac{1}{2}\left(1 - \left(\frac{r}{\sigma}\right)^2\right)^2 & \|r\| < \sigma \\ 0 & \text{otherwise} \end{cases}$

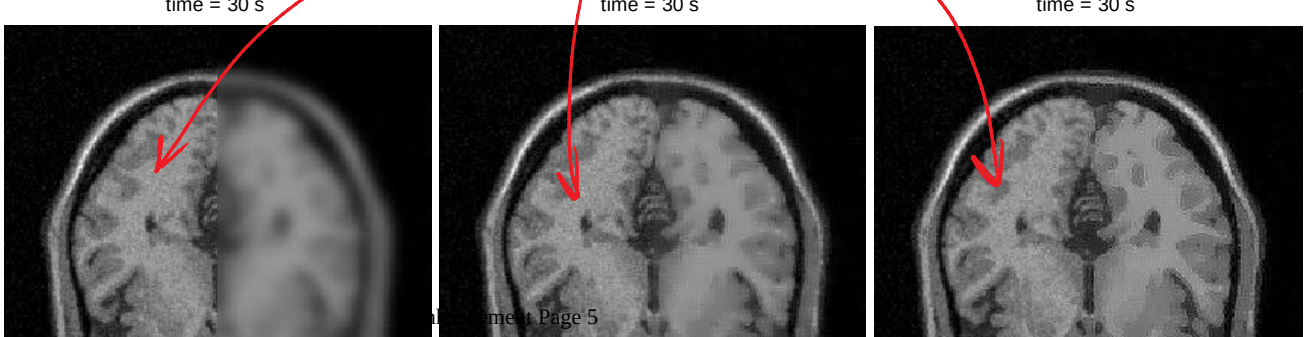


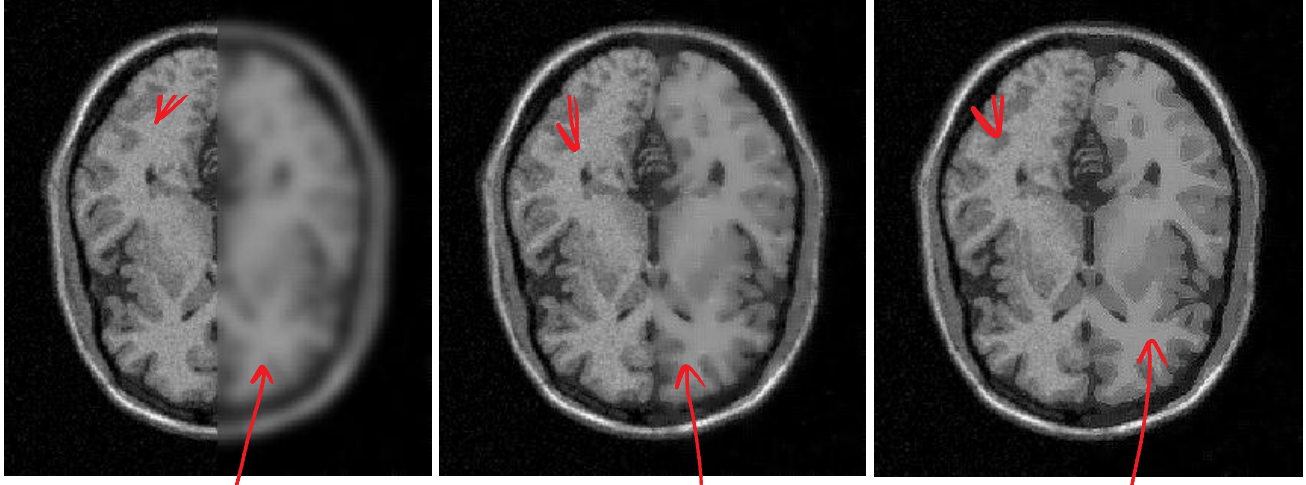
Huber's Minimax Estimator $g(r) = \begin{cases} \frac{1}{\sigma} & |r| \le \sigma \\ \frac{\text{sign}(r)}{r} & |r| > \sigma \end{cases}$



Samples (see Matlab demo)

original noisy





$\frac{\partial f}{\partial t} = c\,\Delta f$

$\frac{\partial f}{\partial t} = g(\|\nabla f\|)\,\Delta f$

$\frac{\partial f}{\partial t} = \nabla \cdot \left(g(\|\nabla f\|)\,\nabla f \right)$

END