Partial Differential Equations and Image Processing

Eshed Ohn-Bar

OBJECTIVES
In this presentation you will…

1) Learn what **partial differential equations** are and where do they arise

2) Learn how to **discretize** and **numerically approximate** solutions of a particular PDE, the **heat equation**, using MATLAB

3) Learn how energy minimization of the **total variation norm** can be used to de-noise an image

Warm Up – Solving an Ordinary Differential Equation

I.V.P:	SEPERABLE!	INTEGRATE	USE INITIAL CONDITIONS
$\frac{dx}{dt} = kx$ $x(t_0) = x_0$	$\frac{dx}{x} = kdt$	$\ln(x) = kt + c$ $x(t) = Ce^{kt}$ $x > 0$	$x(0) = x_0 = C$ $\boldsymbol{x(t) = x_0 e^{kt}}$

x depends on t only.

What if we have **more than one variable** involved?

- 1st objective: Learn what **partial differential equations** are and where do they arise

Definitions

- ODE: **One** independent variable
- PDE: **Several** independent variables, relationship of functions and their **partial derivatives.**

- Notation: $f_x = \dfrac{\partial f}{\partial x}$
- Gradient (2D): $\nabla f(x, y) = (f_x,\ f_y)$

- Laplacian (2D): $\Delta f(x, y) = f_{xx} + \ f_{yy}$

Discrete derivative

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Finite difference:

First derivative using a forward difference

- **$f_x = f(x+1,y) - f(x,y)$**

In MATLAB: n = length(f); f_x = [f(2:n) f(n)] -f(1:n)

Second Derivative using a 2^{nd} order central difference:

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

In MATLAB: f_xx = f(:,[2:n,n])-2*f +f(:,[1,1:n-1]);

The Heat Equation and Diffusion

In 1D: $u_t = cu_{xx}$
In 2D: $u_t = c\Delta u = c(u_{xx} + u_{yy})$

- $\boldsymbol{u}(\boldsymbol{x}, t)$ – temperature function, at point **x** and time t

- Need initial conditions!

  $u(\boldsymbol{x}, t_0) = f(\boldsymbol{x})$ initial temperature at each point

  Also boundary conditions, when **x**=0 and **x**=L

… To the next objective of discretizing the Heat Equation and the beautiful connection between PDEs and image processing…

Code – Discrete Heat Equation $U_t = \Delta U$

```
dt = 0.1;          <- Time Step
T = 10;          <- Stopping time
[m,n]=size(u);
for t = 0:dt:T
    u_xx = u(:,[2:n,n])-2*u +u(:,[1,1:n-1]);
    u_yy = u([2:m,m],:) - 2*u + u([1,1:m-1],:);
    L = uxx + uyy;
    u = u + dt*L;
end
```

<- Finite Differences for U_{xx} and U_{yy}

<- Finite Difference for U_t

- 2^{nd} objective: Learn how to **discretize** the **heat equation**

Heat Equation on an Image

- What would happen if we evolve the heat equation on an image? dt = 0.2

(a) Original Image

(b) Time = 5

(c) Time = 10

(d) Time = 30

- 2nd objective: Learn how to **discretize** the **heat equation**

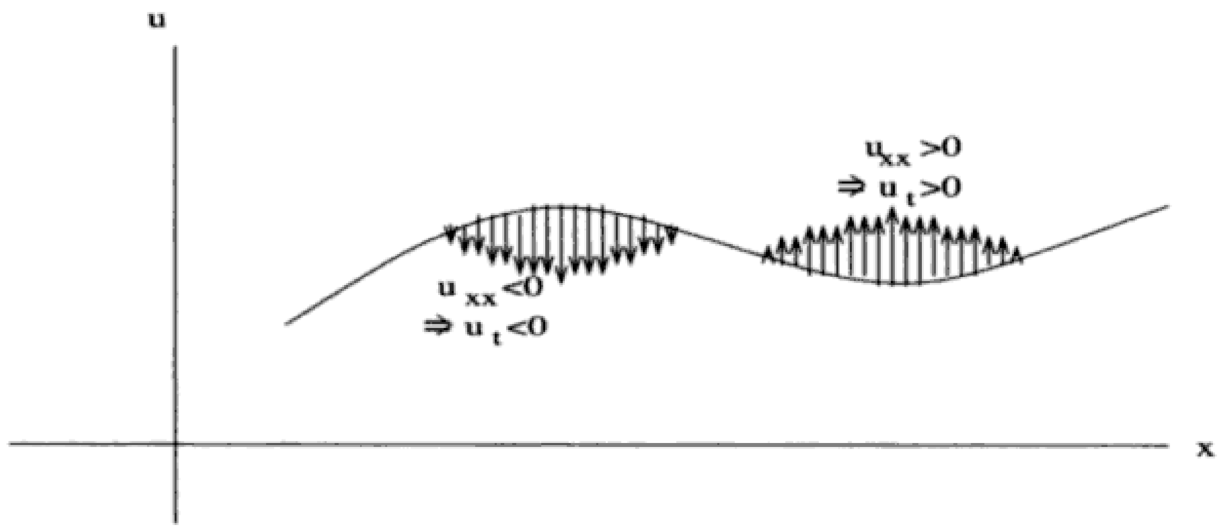
Heat Equation on an Image

- Applying the heat equation causes blurring. Why?

- <u>Graphical interpretation of the heat equation</u>

U concave down ⟹ **$U_t < 0$** ⟹ **U decreasing**

U concave up ⟹ **$U_t > 0$** ⟹ **U increasing**



- 2nd objective: Learn how to **<u>discretize</u>** the **<u>heat equation</u>**

Heat Equation on an Image

What's going to happen as t->∞ ?

Diffusion of heat smoothes the temperature function

Equivalent to **minimizing** the L-2 norm of the **gradient**:

$$u_t = \Delta u \leftrightarrow \min \int ||\nabla u||^2$$

- Problem: **Isotropic** diffusion, uniform, doesn't consider **shapes** and **edges.**

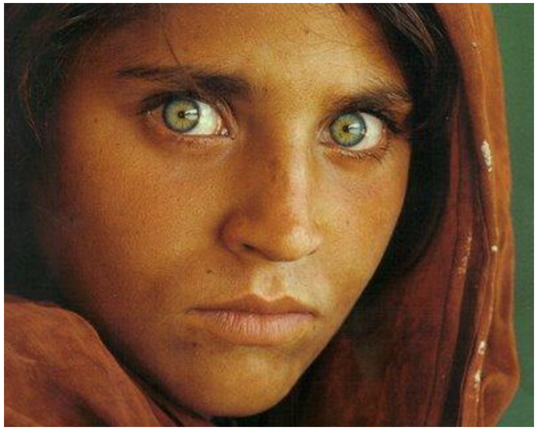
Anisotropic Diffusion

$$u_t = \nabla \cdot \left(\frac{\nabla u}{\|\nabla u\|}\right)$$

Slows down diffusion at the edges

Anisotropic Diffusion

(a) Original Image



(b) Time = 5

(c) Time = 10

(d) Time = 30

- 3rd objective: Learn how energy minimization of **total variation** can de-noise an image

Anisotropic Diffusion

(a) Original Image

(b) Time = 5

(c) Time = 10

(d) Time = 30

- 3rd objective: Learn how energy minimization of **total variation** can de-noise an image

Anisotropic Diffusion – Total Variation (TV)[1]

Goal: remove noise without blurring object boundaries.

We add a regularization term to change the steady state solution.
Minimize the **total variation** energy:

$$\min_u E[u] = \int |\nabla u| + \lambda \int (u - u_0)^2$$

Using the Euler – Lagrange equation

$$\nabla E = -\frac{u_y^2 u_{xx} - 2u_x u_y u_{xy} + u_x^2 u_{yy}}{\left(u_x^2 + u_y^2\right)^{3/2}} + 2\lambda(u - u_0)$$

$$\frac{\partial u}{\partial t} = -\nabla E\,.$$

[1] Rudin, L. I.; Osher, S. J.; Fatemi, E. Nonlinear total variation based noise removal algorithms. Phys. D 60 (1992), 259–268

3rd objective: Learn how energy minimization of **total variation** can de-noise an image

TV - Code

- T = 100; <- Time Step
- dt = 0.2; <- Stopping time
- epsilon = 0.01;
- for t = 0:dt:T
- u_x = (u(:,[2:n,n]) - u(:,[1,1:n-1]))/2;
- u_y = (u([2:m,m],:) - u([1,1:m-1],:))/2; <- Finite Differences
- u_xx = u(:,[2:n,n]) - 2*u + u(:,[1,1:n-1]); for Partial Derivatives
- u_yy = u([2:m,m],:) - 2*u + u([1,1:m-1],:);
- u_xy = (u([2:m,m],[2:n,n]) + u([1,1:m-1],[1,1:n-1]) - u([1,1:m-1],[2:n,n]) - u([2:m,m],[1,1:n-1])) / 4;
- Numer = u_xx.*u_y.^2 - 2*u_x.*u_y.*u_xy + u_yy.*u_x.^2;
- Deno = (u_x.^2 + u_y.^2).^(3/2) + epsilon; <- Finite Difference
- u = u + dt*(Numer./Deno)- 2*lambda*(u-u0(:,:,1)); for U_t

TV Denoising **Lambda = 0.01**

Original Image

Gaussian Noise

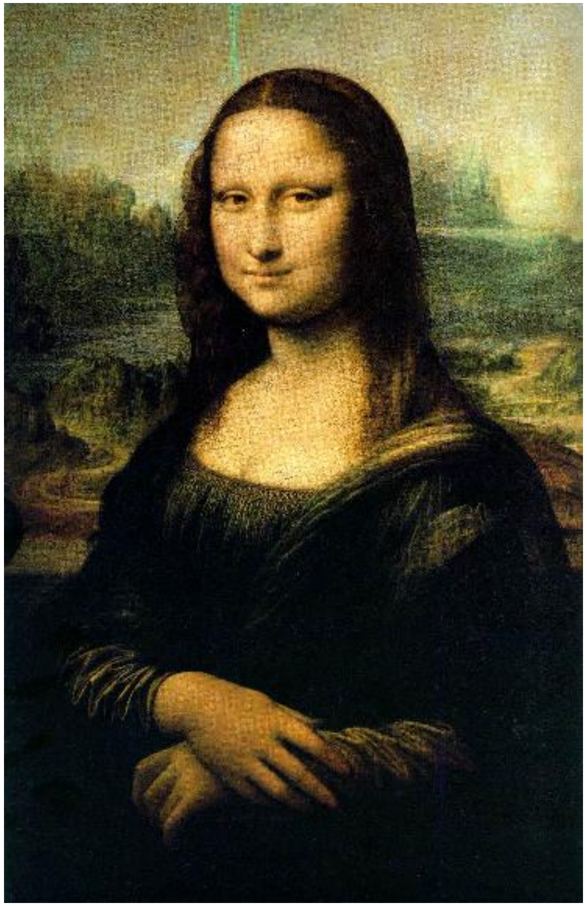
Time = 70

Time = 200

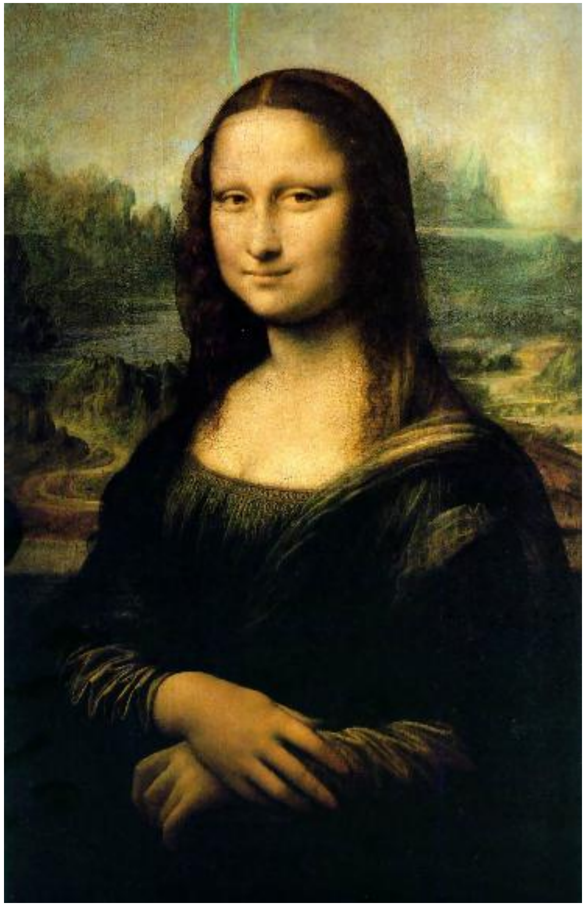
- 3^{rd} objective: Learn how energy minimization of **total variation** can de-noise an image

TV Denoising **lambda = 0.1**

Original



Time = 5



Time = 10

- 3rd objective: Learn how energy minimization of **total variation** can de-noise an image

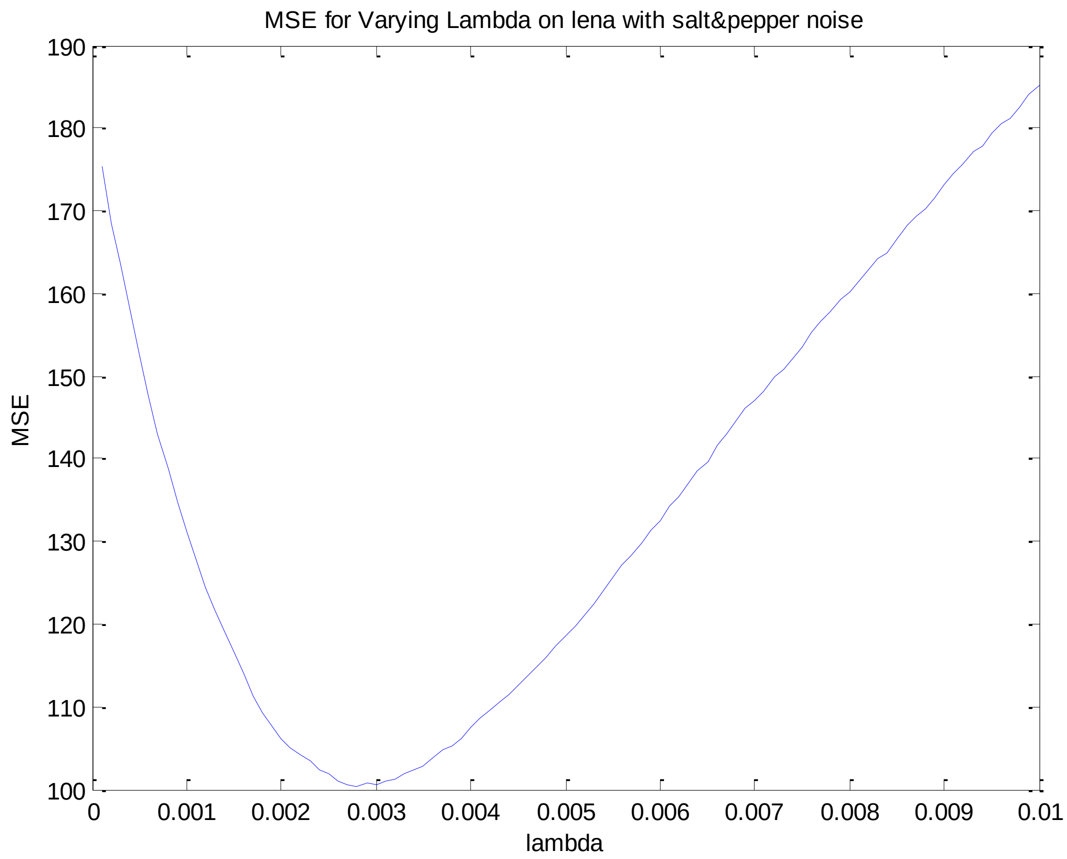
How to choose Lambda?

- There are various optimization and ad-hoc methods, beyond the scope of this project.

- In this project, the value is determined by **pleasing results.**

- Lambda too large -> may not remove all the noise in the image.
- Lambda too small -> it may distort important features from the image.

How to choose Lambda?



Original

Salt & Pepper Noise

De-noised

Summary

- Energy minimization problems can be translated to a PDE and applied to de-noise images
- We can use the magnitude of the gradient to produce anisotropic diffusion that preserves edges
- TV energy minimization uses the L1-norm of the gradient, which produces nicer results on images than the L2-norm