# Visual Computing - Notes Week 14

Ruben Schenk, ruben.schenk@inf.ethz.ch

December 27, 2021

### 8.4.8 Partial Differential Equations

In contrast to ODEs, where an unknown function is described through its derivatives with respect to a single variable, **partial differential equations (PDEs)** describe an unknown function through its partial derivatives with respect to *multiple* variables:

$$\frac{\partial u(t,\, x)}{\partial t^2} = c^2 \frac{\partial u(t,\, x)}{\partial x^2}$$

**Fluid Simulation in Graphics** *Incompressible Navier Stokes Equations:*

$$\nabla \cdot u = 0 \frac{\partial u}{\partial t} + (u \cdot \nabla)u - v\nabla^2 u = -\nabla w + g$$

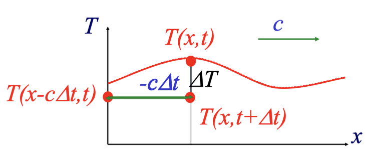
**Elasticity in Graphics** *Governing Equations of Continuum Mechanics:*

$$\nabla \cdot \sigma + f = m \cdot a$$

**Magnetism in Graphics** *Maxwell Equations (static case):*

$$\nabla \cdot B = 0, \quad \nabla \times H = JH = \frac{1}{\mu_0} B - M$$

**1D Advection** Consider the following example, where we are given some initial temperature distribution $T_0(x) = T(x,\, 0)$ and some wind speed $c$. We want to find the temperature distribution $T(x,\, t)$ for any $t$:




$$\boxed{T(x, t+\Delta t)} = \boxed{T(x - c\Delta t, t)}$$
$$\boxed{\Delta T} = T(x, t+\Delta t) - T(x,t)$$

$$T(x - c\Delta t, t) = \boxed{T(x,t)} - \frac{\partial T}{\partial x} c\Delta t + O(\Delta t^2) \ = \boxed{T(x, t+\Delta t)}$$

**1D advection equation**

$$\frac{\Delta T}{\Delta t} \approx -c\frac{\partial T}{\partial x} \quad \xrightarrow{\Delta t \to 0} \quad \frac{\partial T}{\partial t} = -c\frac{\partial T}{\partial x}$$

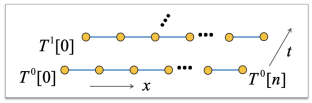
We can solve this problem *analytically:*

- Any $T(x,\, t)$ of the form $T(x,\, t) = f(x - ct)$ solves $\frac{\partial T}{\partial t} = -c\frac{\partial T}{\partial x}$
- The solution also needs to satisfy the initial condition $T(x,\, 0) = T_0(x)$
- The solution therefore is given by $T(x,\, t) = T_0(x - ct)$

*Note:* Only simple PDEs can be solved analytically!

We might also solve the problem *numerically:*

- Sample temperature *T(x,t)* on 1D grids $T^t[i] = T(i \cdot h, t \cdot \Delta t)$ with $i \in (1,..,n), t \in (0,1,2..)$




- Discretize derivatives with **finite differences** (*space & time*)

$$\frac{\partial T}{\partial t} = -c\frac{\partial T}{\partial x} \quad \Rightarrow \quad \frac{T^{t+1}[i] - T^t[i]}{\Delta t} = -c\frac{T^t[i] - T^t[i-1]}{h}$$

- Solving for $T^{t+1}[i]$ yields update rule

$$T^{t+1}[i] = T^t[i] - \Delta t \cdot c\frac{T^t[i] - T^t[i-1]}{h}$$

- Provide initial values $T^0[i]$
- Set boundary conditions, e.g. *periodic* $\quad T^t[0] = T^t[n]$

- Abbreviation $\quad u_{tt} = \dfrac{\partial^2}{\partial t^2}u(t,..), \quad u_{xy} = \dfrac{\partial^2}{\partial x\,\partial y}u(x,y,..)$
- Spatial variables $\quad \mathbf{x} = (x_1,\ldots,x_d)^t$
- Nabla operator $\quad \nabla = \left(\dfrac{\partial}{\partial x_1},\ldots,\dfrac{\partial}{\partial x_d}\right)^t \quad \nabla s = \left(\dfrac{\partial s}{\partial x_1},\ldots,\dfrac{\partial s}{\partial x_d}\right)^t$
- Laplace operator $\quad \Delta = \nabla^t \cdot \nabla = \nabla_{\mathbf{x}}^2 = \displaystyle\sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2}$

(in *d* dimensions)

**Some Notation**

**PDE Classification** The **order** of a PDE is the order of the highest partial derivative. A PDE is said to be **linear** if the unknown function $u$ and its partial derivatives only occur linearly.

Second order linear PDEs are of high practical relevance. A second order linear PDE in 2 variables has the following form:

$$Au_{xx} + 2Bu_{xy} + Cu_{yy} = F(x,\, y,\, u,\, u_x,\, u_y)$$

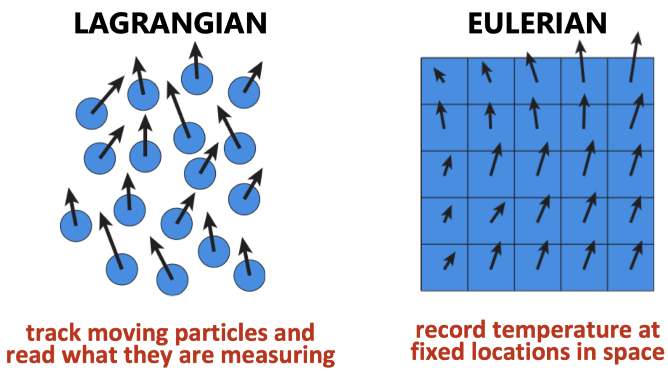
A second order linear PDE in 2 variables can be classified into:

- *Hyperbolic:* $B^2 - AC > 0$ (wave equation)
- *Parabolic:* $B^2 - AC = 0$ (heat equation)
- *Elliptic:* $B^2 - AC < 0$ (Laplace equation)

**Solving PDEs** Like ODEs, many interesting PDEs are difficult or impossible to solve analytically. The basic strategy is as follows:

- Pick a spatial discretization
- Pick a time discretization (forward Euler, backward Euler, etc.)
- As with ODEs, run a time-stepping algorithm

**Spatial Discretization** Two basic ways to **discretize space** are the Lagrangian and the Eulerian approach:




We observe the following trade-offs:

- Lagrangian:
  - Conceptually easy
  - Resolution/domain not limited by grid
  - Good particle distribution can be tough
  - Finding neighbors can be expensive
- Eulerian:
  - Fast, regular computation
  - Easy to represent
  - Simulation is "trapped" in a grid

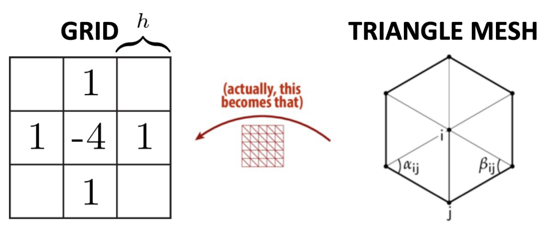
**Nabla operator** $$\nabla = \left(\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_d}\right)^t \qquad \nabla \boldsymbol{u} = \left(\frac{\partial u}{\partial x_1}, \dots, \frac{\partial u}{\partial x_d}\right)^t$$

**Laplace operator** $$\Delta = \nabla \cdot \nabla = \sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2} \qquad \Delta u = \overset{\text{div}}{\nabla} \cdot \overset{\text{grad}}{\nabla} u$$

$$\Delta u = \frac{\partial u^2}{\partial x_1^2} + \cdots + \frac{\partial u^2}{\partial x_n^2}$$

### The Laplace Operator

*Discretization:*




$$\frac{4u_{ij} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}}{h^2} \qquad \frac{1}{2}\sum_j (\cot \alpha_{ij} + \cot \beta_{ij})(u_j - u_i)$$

*Numerically solving the Laplace equation:*

- Want to solve Δu = 0
- Plug in one of our discretizations, e.g.,

| | | |
|---|---|---|
| | $c$ | |
| $d$ | $a$ | $b$ |
| | $e$ | |

$$\frac{4a - b - c - d - e}{h} = 0$$

$$\iff a = \tfrac{1}{4}(b + c + d + e)$$

- At solution that solves the Laplace Equation, each value is the average of neighboring values.
- How do we solve this?
- One idea: keep averaging with neighbors! (“Jacobi method”)
- Correct, but *slow* convergence