Berkeley
MSSE
MOLECULAR SCIENCE &
SOFTWARE ENGINEERING

Lecture 4:

Numerical Differentiation and Integration

Markus Hohle

University California, Berkeley

Numerical Methods for Computational Science

MSSE 273, 3 Units

Course Map

Week 1:	Introduction to Scientific Computing and Python Libraries
Week 2:	Linear Algebra Fundamentals
Week 3:	Vector Calculus
Week 4:	**Numerical Differentiation and Integration**
Week 5:	Solving Nonlinear Equations
Week 6:	Probability Theory Basics
Week 7:	Random Variables and Distributions
Week 8:	Statistics for Data Science
Week 9:	Eigenvalues and Eigenvectors
Week 10:	Simulation and Monte Carlo Method
Week 11:	Data Fitting and Regression
Week 12:	Optimization Techniques
Week 13:	Machine Learning Fundamentals

Outline

- Finite Differences

- Numerical Integration

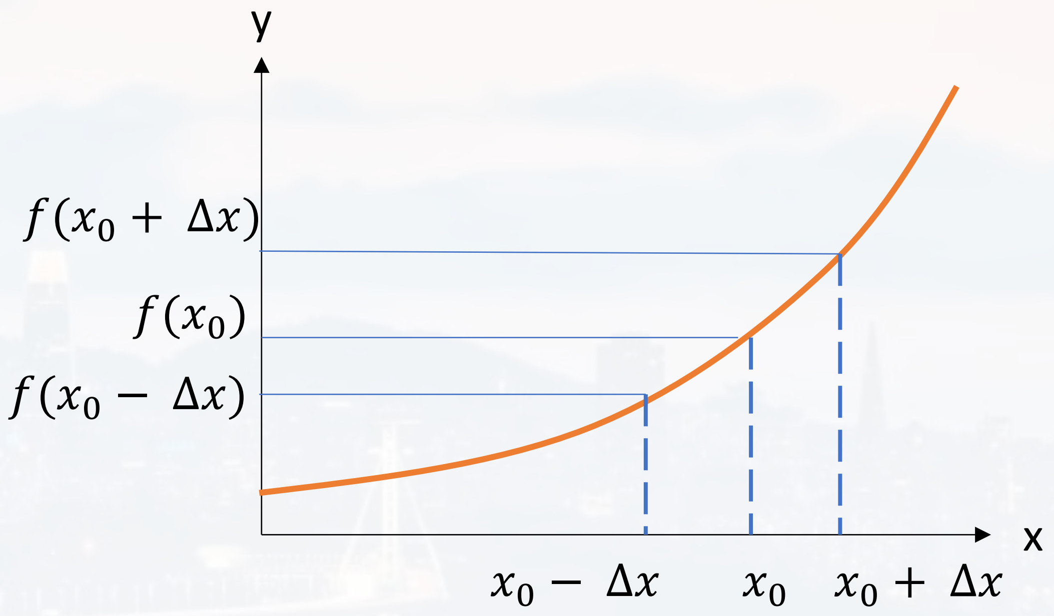
Outline

- **Finite Differences**
- Numerical Integration

slope of a function at $x = x_0$

$$\left.\frac{df^{+}}{dx}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

$$\left.\frac{df^{-}}{dx}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0) - f(x_0 - \Delta x)}{\Delta x}$$



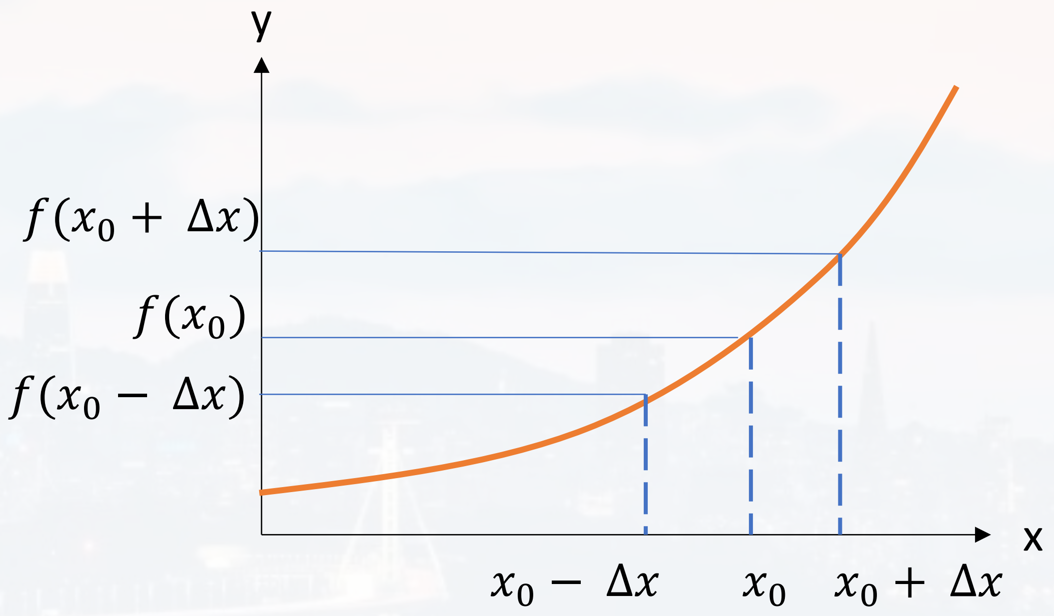
$$\boxed{\left.\frac{df}{dx}\right|_{x=x_0} = \frac{1}{2}\left(\left.\frac{df^{+}}{dx}\right|_{x=x_0} + \left.\frac{df^{-}}{dx}\right|_{x=x_0}\right) = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{2\Delta x}}$$

1st derivative at $\mathrm{x} = x_0$

slope of a function at $x = x_0$

$$\left.\frac{df^+}{dx}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

$$\left.\frac{df^-}{dx}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0) - f(x_0 - \Delta x)}{\Delta x}$$



Δx

$\Delta f = f(x_0 + \Delta x) - f(x_0)$

finite differences

$$\boxed{\begin{aligned}&\Delta x\\ &\Delta f = f(x_0 + \Delta x) - f(x_0)\end{aligned}}$$

finite differences

$$f(x) = \sum_{n=0}^{\infty} \frac{1}{n!} \frac{d^n f}{dx^n}\bigg|_{x=x_0} (x - x_0)^n$$

Taylor Series:
approximation of n-th order: error $\varepsilon = \varepsilon(\Delta x^{n+1})$

<u>see last lecture exercise</u>

n = 2:

$$E(x) \approx E(x_0) + \frac{dE}{dx}\bigg|_{x=x_0} (x - x_0) + \frac{1}{2} \frac{d^2 f}{dx^2}\bigg|_{x=x_0} (x - x_0)^2$$

$$E(x) \approx E(x_0) + \frac{1}{2} \frac{d^2 f}{dx^2}\bigg|_{x=x_0} (x - x_0)^2$$

$$E(x) \approx E(x_0) + \frac{1}{2} k (x - x_0)^2$$

$$\boxed{\Delta E \approx \frac{1}{2} k \, \Delta x^2}$$

$$\Delta x$$

$$\Delta f = f(x_0 + \Delta x) - f(x_0)$$

finite differences

$$\left.\frac{df}{dx}\right|_{x=x_0} = \frac{1}{2}\left(\left.\frac{df^+}{dx}\right|_{x=x_0} + \left.\frac{df^-}{dx}\right|_{x=x_0}\right) = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{2\Delta x}$$

1st derivative at $x = x_0$

$$\left.\frac{d^2 f}{dx^2}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - 2f(x_0) + f(x_0 - \Delta x)}{\Delta x^2}$$

2nd derivative at $x = x_0$

often, we need to **model diffusion processes** (heat conduction, diffusion reaction, electromagnetism etc)

$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

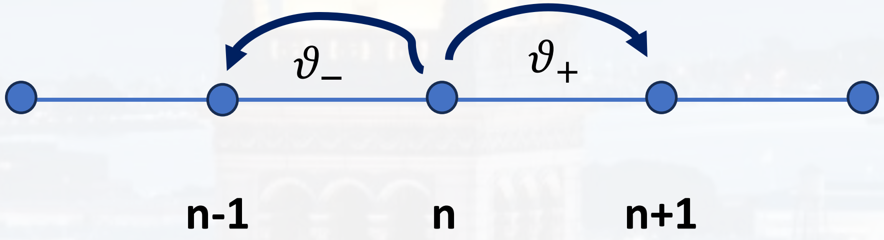
often, we need to **model diffusion processes** (heat conduction, diffusion reaction, electromagnetism etc)

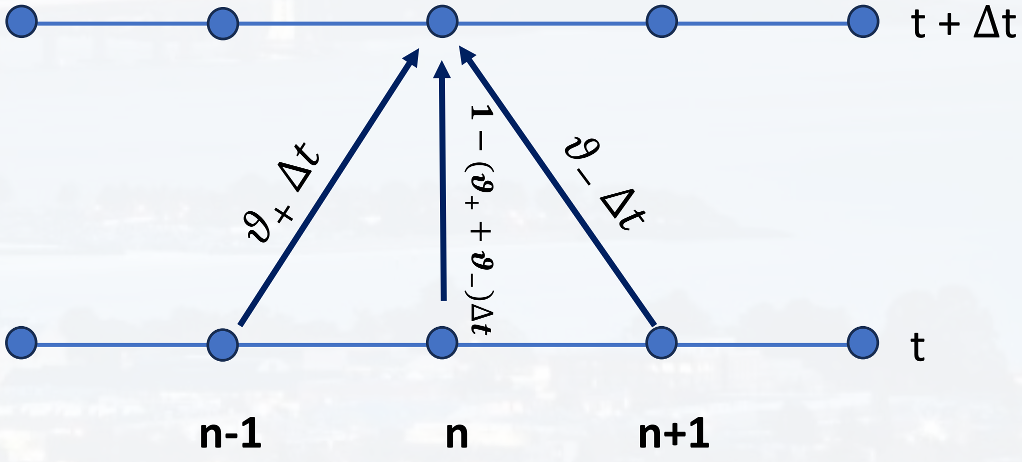
$$\boxed{\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)}$$

Fick's 2nd law

c:	concentration
D:	diffusion constant
$\Delta \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$:	Laplace operator

n:	state (e.g. location)
ϑ_-, ϑ_+ :	hopping rate (probability per time)





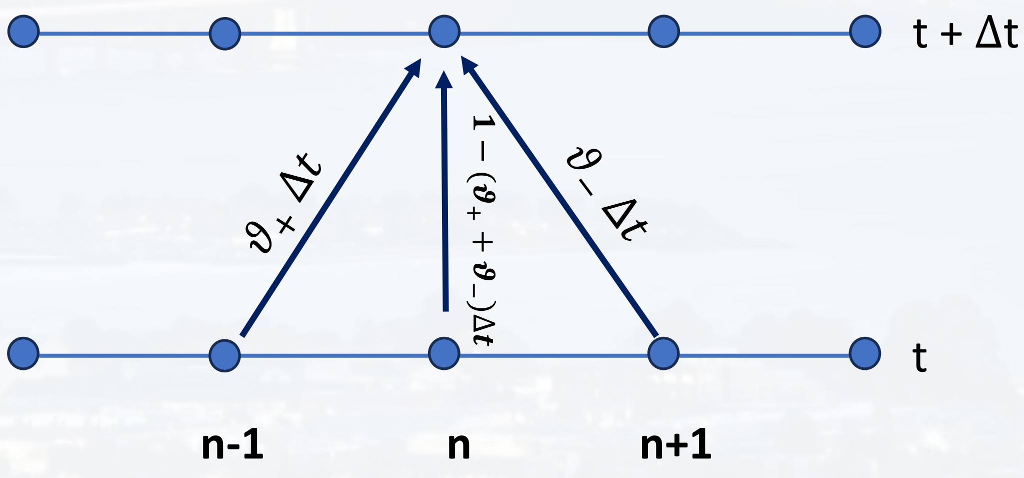
Numerical Differentiation and Integration

$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$ Fick's 2nd law

The probability ***P(n,t)*** that the system is in state ***n*** at time ***t***

$$\frac{\partial}{\partial t}P(n,t) = P(jump\ up|\ at\ n-1)$$
$$+ P(jump\ down|\ at\ n+1)$$
$$- P(jump\ down|\ at\ n)$$
$$- P(jump\ up|\ at\ n)$$

c:	**concentration**
D:	**diffusion constant**
$\Delta \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$:	**Laplace operator**
n:	**state (e.g. location)**
ϑ_-, ϑ_+ :	**hopping rate (probability per time)**



Numerical Differentiation and Integration

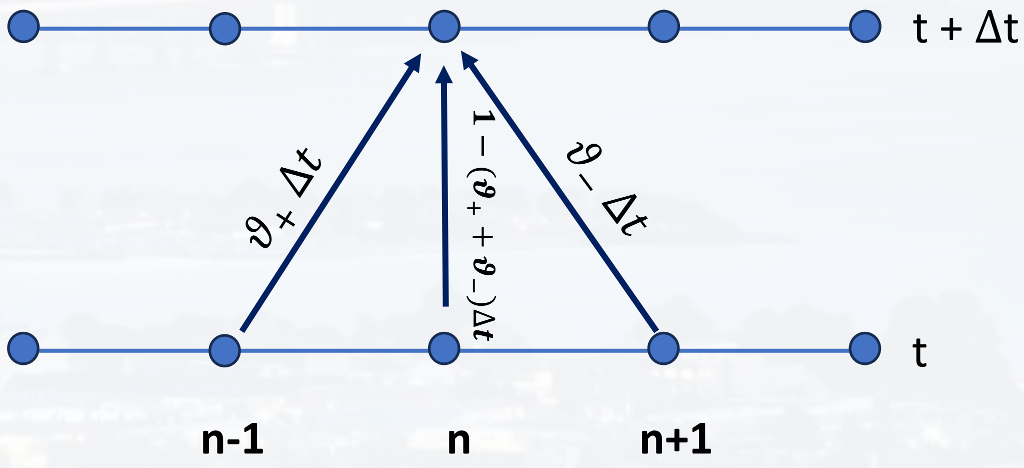
$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

Fick's 2nd law

The probability ***P(n,t)*** that the system is in state ***n*** at time ***t***

$$\frac{\partial}{\partial t}P(n,t) = \vartheta_+ P(n-1,t) + \vartheta_- P(n+1,t) - \vartheta_- P(n,t) - \vartheta_+ P(n,t)$$

c:	**concentration**
D:	**diffusion constant**
$\Delta \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$:	**Laplace operator**
n:	**state (e.g. location)**
ϑ_-, ϑ_+ :	**hopping rate (probability per time)**



Numerical Differentiation and Integration

$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

Fick's 2nd law

c:	**concentration**
D:	**diffusion constant**
$\Delta \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$:	**Laplace operator**
n:	**state (e.g. location)**
ϑ_-, ϑ_+ :	**hopping rate (probability per time)**

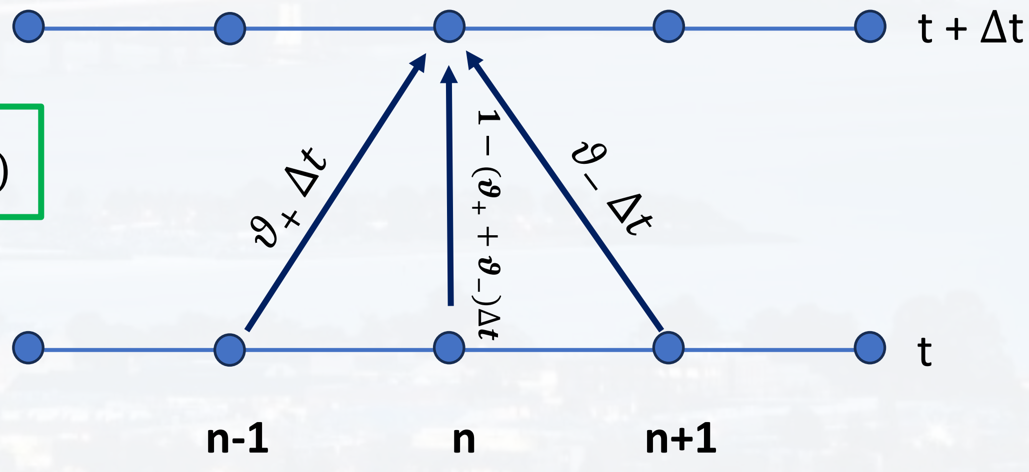
The probability ***P(n,t)*** that the system is in state ***n*** at time ***t***

$$\frac{\partial}{\partial t} P(n,t) = \vartheta_+ \, P(n-1,t) + \vartheta_- \, P(n+1,t) - [\vartheta_- + \vartheta_+]\, P(n,t)$$

in case $\vartheta_- = \vartheta_+ = \vartheta$

$$\boxed{\frac{\partial}{\partial t} P(n,t) = \vartheta \, P(n-1,t) + \vartheta \, P(n+1,t) - 2\vartheta P(n,t)}$$

Master Equation



Numerical Differentiation and Integration

$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

Fick's 2nd law

$$\boxed{\frac{\partial}{\partial t}P(n,t) = \vartheta[P(n-1,t) + P(n+1,t) - 2P(n,t)]}$$

Master Equation

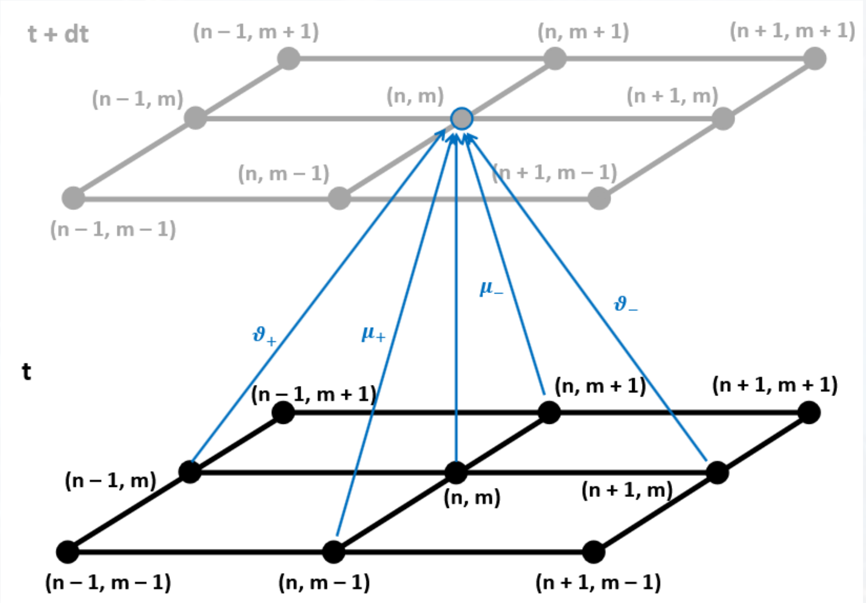
$$\left.\frac{d^2 f}{dx^2}\right|_{x=x_0} = \lim_{\Delta x \to 0} \frac{f(x_0+\ \Delta x)\ - 2f(x_0)\ + f(x_0 -\ \Delta x)}{\Delta x^2}$$

2nd derivative at $x = x_0$

$$\frac{\partial}{\partial t}P(n,t) = \vartheta \frac{\partial^2}{\partial n^2}P(n,t)$$

$$\frac{\partial}{\partial t}P(n,t) = \vartheta\,\Delta P(n,t)$$

same in 2D



Numerical Differentiation and Integration

$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

Fick's 2nd law

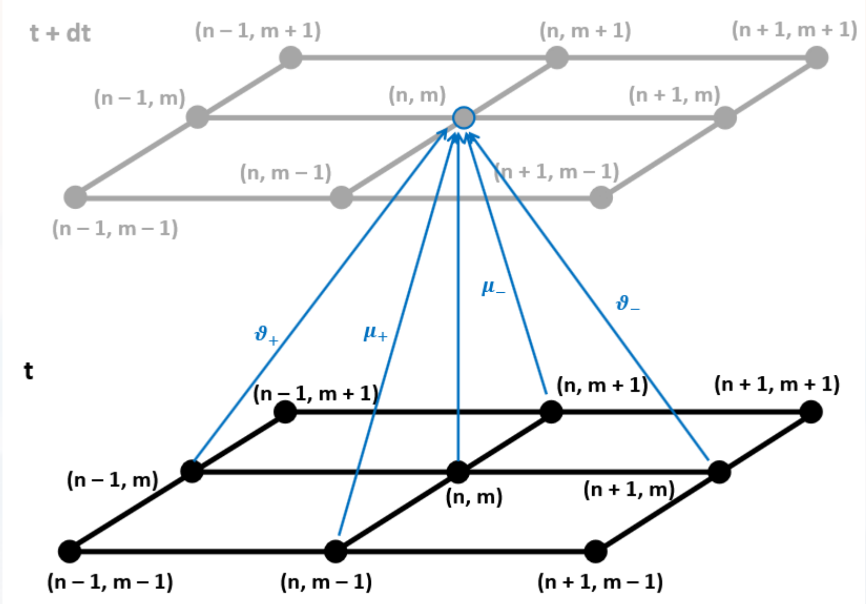
The Laplace operator indeed describes diffusion!

numerically:

$$\frac{c(x_0,\ y_0,\ t_0+\ \Delta t) - c(x_0, y_0,\ t_0-\ \Delta t)}{2\Delta t} = \ D[$$

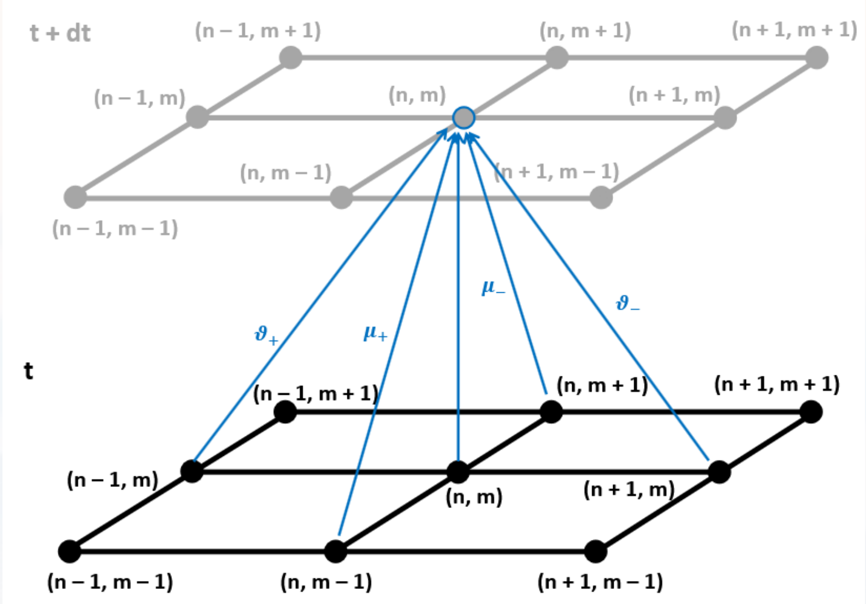
$$\frac{c(x_0+\ \Delta x,\ y_0, t_0)\ - 2c(x_0,\ y_0, t_0) + c(x_0-\ \Delta x,\ y_0, t_0)}{\Delta x^2} +$$

$$\frac{c(x_0,\ y_0+\ \Delta y, t_0)\ - 2c(x_0,\ y_0, t_0) + c(x_0,\ y_0+\ \Delta y, t_0)}{\Delta y^2} \quad]$$



$$\frac{\partial c(x,y,z,t)}{\partial t} = D\,\Delta c(x,y,z,t)$$

Fick's 2nd law



The Laplace operator indeed describes diffusion!

numerically:

$$c(x_0, y_0, t_0 + \Delta t) =$$

We can calculate *c* in the ***future***

$$2\Delta t D \left[\frac{c(x_0+\Delta x,\ y_0, t_0) - 2c(x_0,\ y_0, t_0) + c(x_0 - \Delta x,\ y_0, t_0)}{\Delta x^2} + \right.$$

by using all adjacent ***current*** values

$$\left. \frac{c(x_0,\ y_0 + \Delta y, t_0) - 2c(x_0,\ y_0\ , t_0) + c(x_0,\ y_0 + \Delta y, t_0)}{\Delta y^2}\right]$$

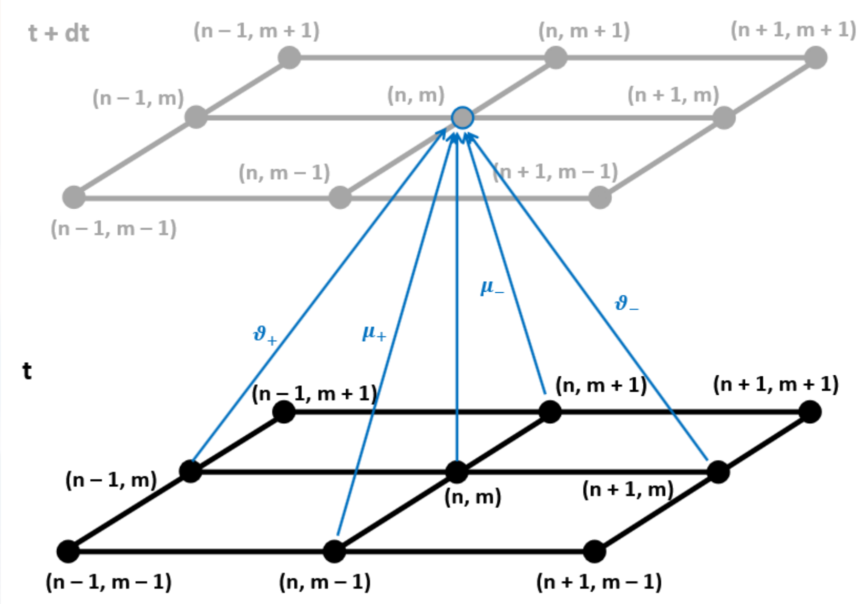
$$+\ c(x_0, y_0, t_0 - \Delta t)$$

and the immediate ***past*** value

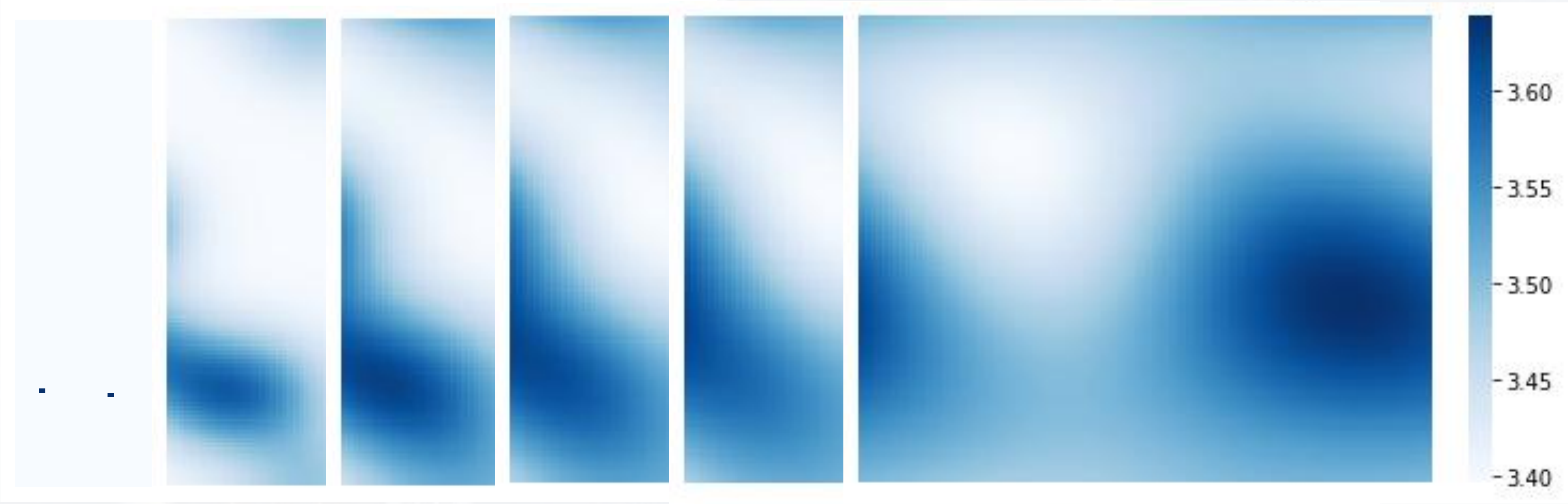
Numerical Differentiation and Integration

run the script `Diffusion2D.py`

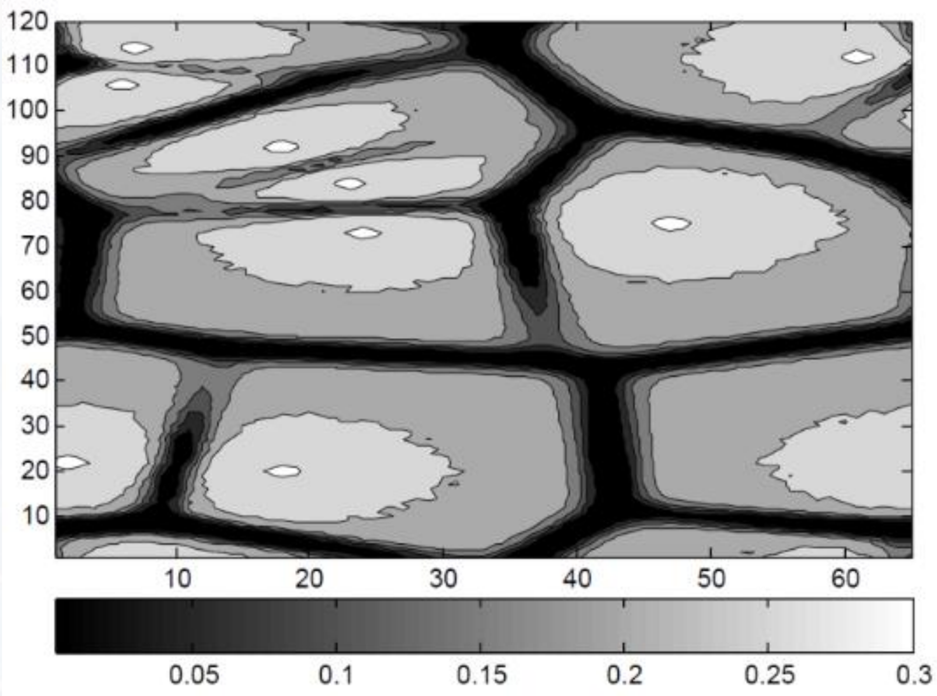
```
from Diffusion2D import *

D = Diffusion2D()
D.RunSimulation()
```

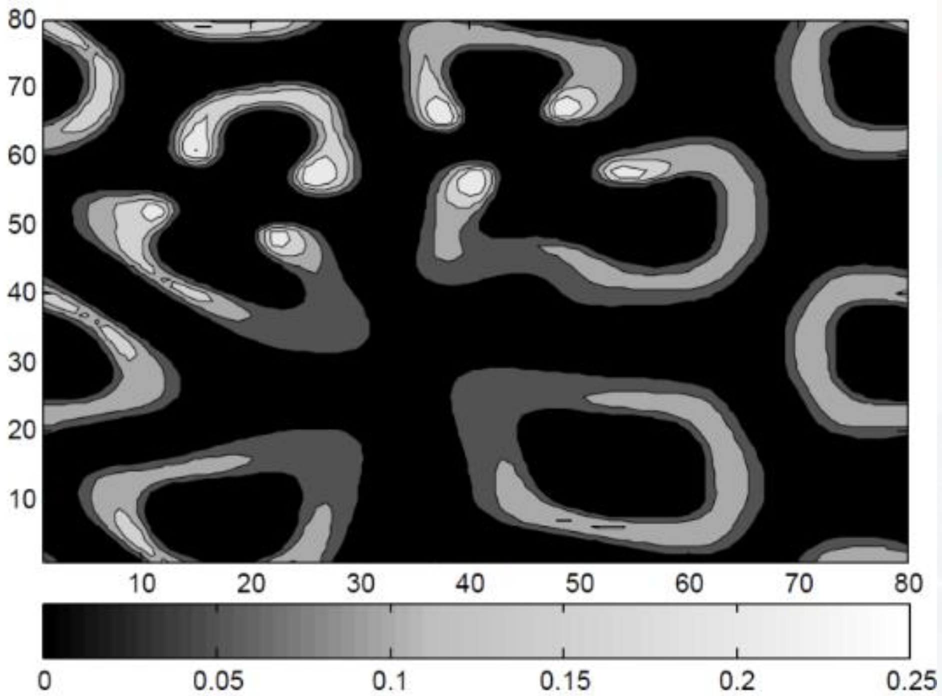


initial condition



Project I (Module 9): modelling fur and skin pattern:





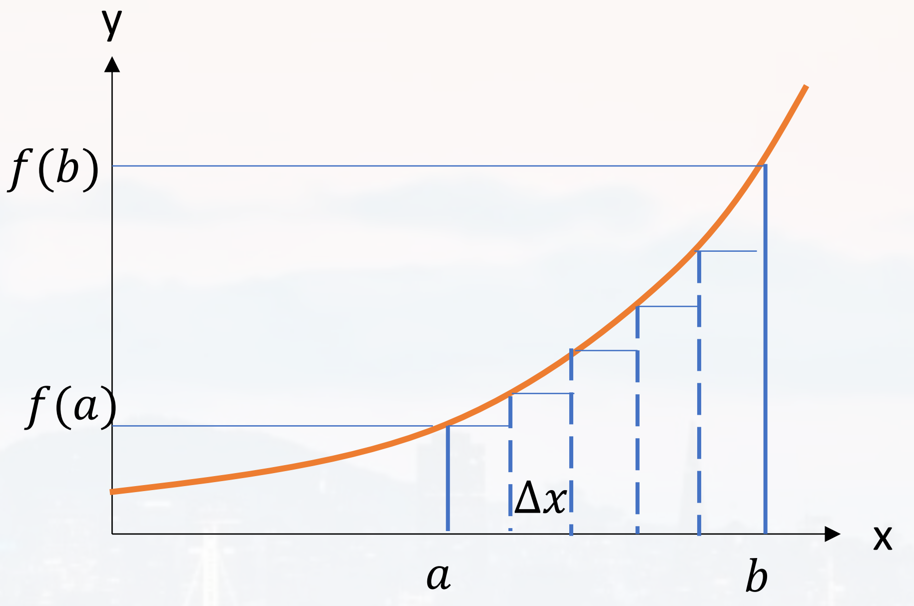
Outline

- Finite Differences

- **Numerical Integration**

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum_{i=0}^{N-1} f(a + i\,\Delta x)\,\Delta x$$

$$N = \frac{b - a}{\Delta x}$$



more accurate:

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum_{i=0}^{N-1} [f(a + i\,\Delta x) + f(a + (i+1)\,\Delta x)]\,\frac{\Delta x}{2}$$

trapezoidal rule

error (for large N):

$$\varepsilon = -\frac{(b-a)^2}{12\,N^2}\,[f'(b) - f'(a)] + O(N^{-3})$$

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum_{i=0}^{N-1} f(a + i\,\Delta x)\,\Delta x \qquad N = \frac{b - a}{\Delta x}$$

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum_{i=0}^{N-1} [f(a + i\,\Delta x) + f(a + (i+1)\,\Delta x)]\,\frac{\Delta x}{2}$$

trapezoidal rule

even more accurate:

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum_{i=0}^{N-1} [f(a + i\,\Delta x) + f(a + (i+1)\,\Delta x) + 4\,f(a + i\,\Delta x/2)]\,\frac{\Delta x}{6}$$

Simpson rule

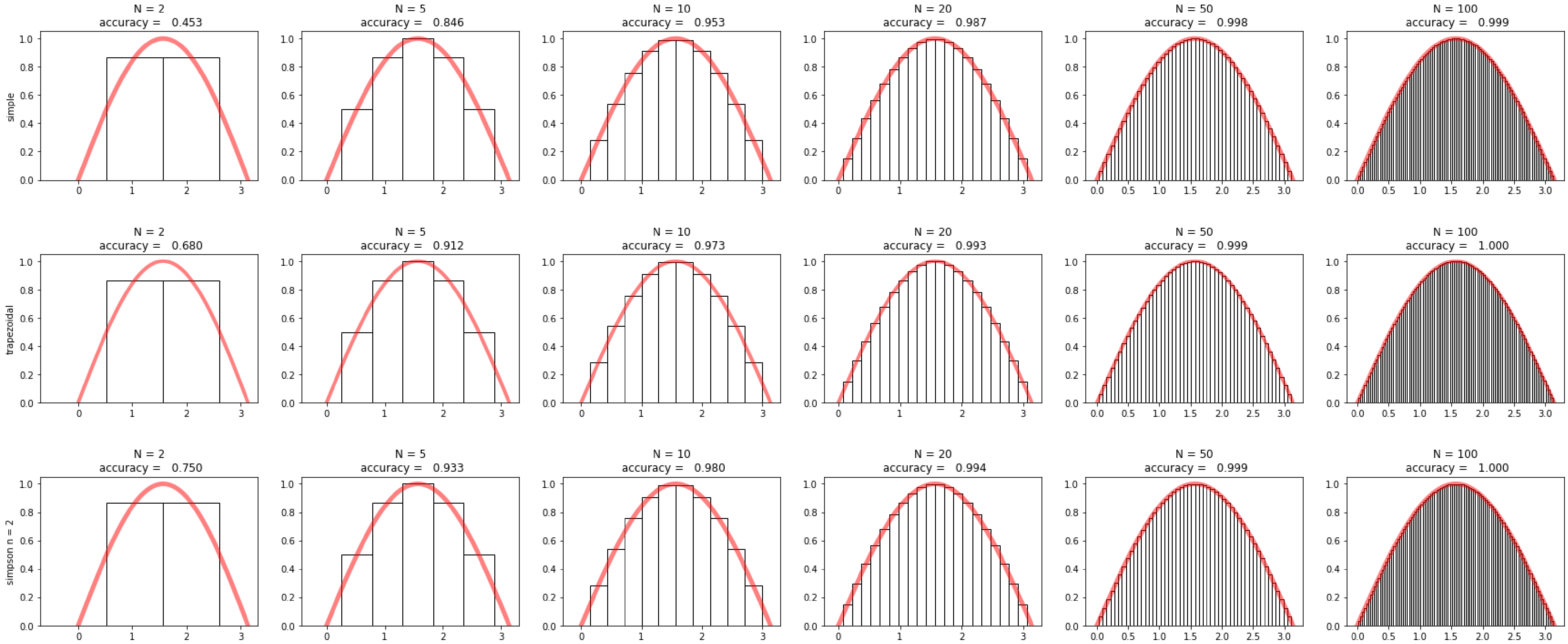
Note: there are different Simpson rules, depending on how many subintervals are included

approximation	error	
$\frac{1}{2}\,\Delta x\,(f_i \;+ f_{i+1})$	$\varepsilon \sim \frac{\Delta x^3}{12}$	trapezoidal
$\frac{1}{6}\,\Delta x\,(f_i \;+ f_{i+2} + 4f_{i+1})$	$\varepsilon \sim \frac{\Delta x^5}{90}$	Simpson
$\frac{1}{8}\,\Delta x\,(f_i \;+ 3f_{i+1} + 3f_{i+2} + f_{i+3})$	$\varepsilon \sim \frac{3\,\Delta x^5}{80}$	Simpson 3/8
$\frac{1}{90}\,\Delta x\,(7f_i \;+ 32f_{i+1} + 12f_{i+2} + 32f_{i+3} + 7f_{i+4})$	$\varepsilon \sim \frac{8\,\Delta x^7}{945}$	Boole

Note: *i* here refers to subinterval **within** Δx

run the function `IntegrationAccuracy.py`

integrating sin(x)



SciPy

```
from scipy.integrate import *

simpson
trapezoid
quad
```

…and more.
See lecture exercise!

```
I = simpson(y, x)
```

Thank you very much for your attention!