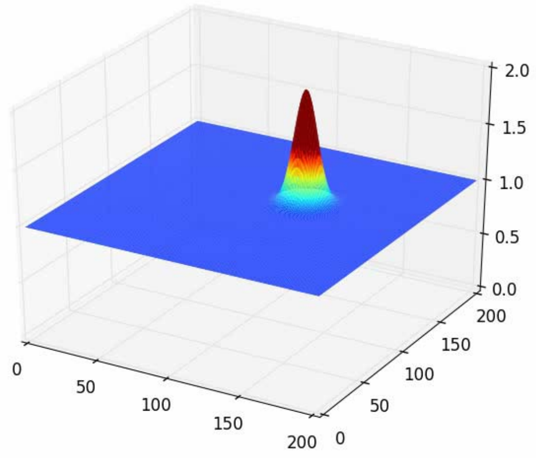


Von Neumann Stability Analyses and Burger’s Equation

Outline of Class

Instructor

- VARQUEZ Alvin Christopher Galang

  Field： Urban meteorology and hydrology

  Department of Transdisciplinary Science and Engineering

  School of Environment and Society

  Ishikawadai Rm. 403 I4-9

  e-mail: varquez.a.aa@m.titech.ac.jp

  office: 03-5734-2768

Recommended Textbook

- Advanced Engineering Mathematics by Erwin Kreyszig (Available in Tokyo Tech library)

Outline of Class

- Fundamental knowledge of PDE, Analytical and Numerical Solutions.
- Lectures and exercises on problems in Science and Engineering, deriving the PDE for physical phenomena, properties of PDE, methods for obtaining analytical and numerical solutions.
- Acquire the ability to perform numerical analyses by combining lectures with computational exercises.

Outline of Class

Schedule:

- Lecture 1 Definition and types of PDE
- Lecture 2 Modeling of flow phenomena with the hyperbolic PDE
- Lecture 3 Analytical solution of the hyperbolic PDE (D'Alembert Solution)
- Lecture 4 Analytical solution of the hyperbolic PDE (Fourier series)
- Lecture 5 Modeling of diffusion phenomena with the parabolic PDE
- Lecture 6 Analytical solution of the parabolic PDE
- Lecture 7 Derivation and solution of elliptical equation Poisson
- Lecture 8 Test level of understanding (Exercise Problems for Lecture 1-7)
- Lecture 9 Introduction to Numerical analysis
- Lesson 10･11 Numerical solution of PDE – Parabolic equation
- Lesson 12･13 Numerical solution of PDE – Combined Parabolic/Hyperbolic
- Lesson 14･15 Numerical solution of PDE – Hyperbolic equation
- Term-end Report

Grading System

mid-term examination (45%), term-end report (45%), short exercises (10%)

Today’s Objectives

- Recap and Von Neumann Stability Analysis
- Burger’s Equation (week’s mission)
- Supplementary: report outline, plotting a 3D surface, quick animation, linux shortcuts.
- Programming time

RECAP & VON NEUMANN STABILITY ANALYSES

Basic FDM Methods

Approximation for the 1st order,

$$\frac{\partial u(x)}{\partial x} = \frac{u_{i+1} - u_i}{\Delta x} + O(\Delta x)$$ Forward Difference

$$\frac{\partial u(x)}{\partial x} = \frac{u_i - u_{i-1}}{\Delta x} + O(\Delta x)$$ Backward Difference

$$\frac{\partial u(x)}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + O(\Delta x^2)$$ Central Difference

Approximation for the 2nd order,

$$\frac{\partial^2 u(x)}{\partial x^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x^2)$$

Mission for the week

$$\frac{\partial T}{\partial t} - \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) = 0$$

Construct a diffusion model for a 2-D heat plate with dimensions 100 m. by 100 m.

1. Desired initial condition (distributed).
2. Investigate w/ or without Dirichlet or Neumann Boundary Conditions.
3. Investigate various influences of $d = \dfrac{\alpha \Delta t}{\Delta x^2}$. What is happening when its values range from 0. to 1.0.
4. Visualize the results by animation.

Grouping

Group			
1	Kiyo Kei	Ratchatawijin Maythawee	
2	Thumwanit Napat	Chin Riyu	
3	Zhou Lijun	Lerdbussarakam Tanad	Shagdar Zolbayar
4	Tsamara Tsani	Ka Kinzei	
5	Pongpattanayok Supatat	Jiang Lechuan	
6	Tumurbaatar Uyanga	Kiyo Yu	
7	Kadohiro Yasuki	Dolgormaa Banzragch	
8	Ka Jiyuntaku	Rudjito Rezkita Ramadhani	
9	Nasution Ghiffari Aby Malik	Kobayashi Yuki	
10	Chaijirawiwat Chawit	Chin Keitoku	
11	Sato Yaoki	Purevsuren Norovsambuu	
12	Do Khanh N	Kazama Tomohiro	

FTCS Method

2-D Diffusion $$\boxed{\frac{\partial T}{\partial t}-\alpha\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}\right)=0}$$

$$\frac{T_{i,j}^{n+1}-T_{i,j}^{n}}{\Delta t}=\alpha\left\{\frac{T_{i+1,j}^{n}-2T_{i,j}^{n}+T_{i-1,j}^{n}}{\Delta x^2}+\frac{T_{i,j+1}^{n}-2T_{i,j}^{n}+T_{i,j-1}^{n}}{\Delta y^2}\right\}$$

$$\Delta x=\Delta y$$

$$\frac{T_{i,j}^{n+1}-T_{i,j}^{n}}{\Delta t}=\frac{\alpha}{\Delta x^2}\left\{T_{i+1,j}^{n}-2T_{i,j}^{n}+T_{i-1,j}^{n}+T_{i,j+1}^{n}-2T_{i,j}^{n}+T_{i,j-1}^{n}\right\}$$

$$T_{i,j}^{n+1}-T_{i,j}^{n}=\frac{\alpha\Delta t}{\Delta x^2}\left\{T_{i+1,j}^{n}-2T_{i,j}^{n}+T_{i-1,j}^{n}+T_{i,j+1}^{n}-2T_{i,j}^{n}+T_{i,j-1}^{n}\right\}$$

$$T_{i,j}^{n+1}=T_{i,j}^{n}+d\left\{T_{i+1,j}^{n}-2T_{i,j}^{n}+T_{i-1,j}^{n}+T_{i,j+1}^{n}-2T_{i,j}^{n}+T_{i,j-1}^{n}\right\}$$

$$\boxed{d=\frac{\alpha\Delta t}{\Delta x^2}}$$ Diffusion Number

Von Neumann Stability Analysis

- A procedure to check the stability of the solution of finite difference equations applied to linear PDE.

- Decay or growth of the error (amplification) determines whether the numerical algorithm is stable.

- Briefly described by British researchers Crank and Nicolson in 1947. Focused in more detail by John von Neumann at a later time.

Von Neumann Stability Analysis

$$\frac{\partial T}{\partial t} - \alpha\left(\frac{\partial^2 T}{\partial x^2}\right) = 0$$

<u>FTCS Method</u>

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \alpha\left(\frac{T_{i+1}^n - 2T_i^n + T_{i-1}^n}{\Delta x^2}\right)$$

Introduce an error term

Error Term

$$T_i^n = \overline{T}_i^n + \varepsilon_i^n$$

Exact solution

Von Neumann Stability Analysis

$$\frac{T_i^{n+1}-T_i^n}{\Delta t}=\alpha\left(\frac{T_{i+1}^n-2T_i^n+T_{i-1}^n}{\Delta x^2}\right)$$

Setting

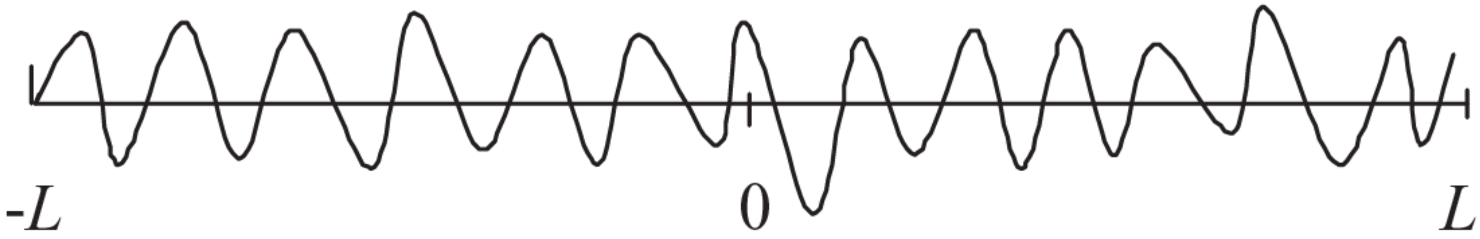
$$T_i^n=\overline{T}_i^n+\varepsilon_i^n$$

$$\underbrace{\cancel{\frac{\overline{T}_i^{n+1}-\overline{T}_i^n}{\Delta t}}}_{0}+\frac{\varepsilon_i^{n+1}-\varepsilon_i^n}{\Delta t}=\alpha\underbrace{\cancel{\left(\frac{\overline{T}_{i+1}^n-2\overline{T}_i^n+\overline{T}_{i-1}^n}{\Delta x^2}\right)}}_{0}+\alpha\left(\frac{\varepsilon_{i+1}^n-2\varepsilon_i^n+\varepsilon_{i-1}^n}{\Delta x^2}\right)$$

$$\frac{\varepsilon_i^{n+1}-\varepsilon_i^n}{\Delta t}=\alpha\left(\frac{\varepsilon_{i+1}^n-2\varepsilon_i^n+\varepsilon_{i-1}^n}{\Delta x^2}\right)$$

Von Neumann Stability Analysis

$$\frac{\varepsilon_i^{n+1}-\varepsilon_i^n}{\Delta t}=\alpha\left(\frac{\varepsilon_{i+1}^n-2\varepsilon_i^n+\varepsilon_{i-1}^n}{\Delta x^2}\right)$$



T.J. Chung (2002)

Considering periodic/cyclic,
the error can be decomposed in a Fourier series for each time n.

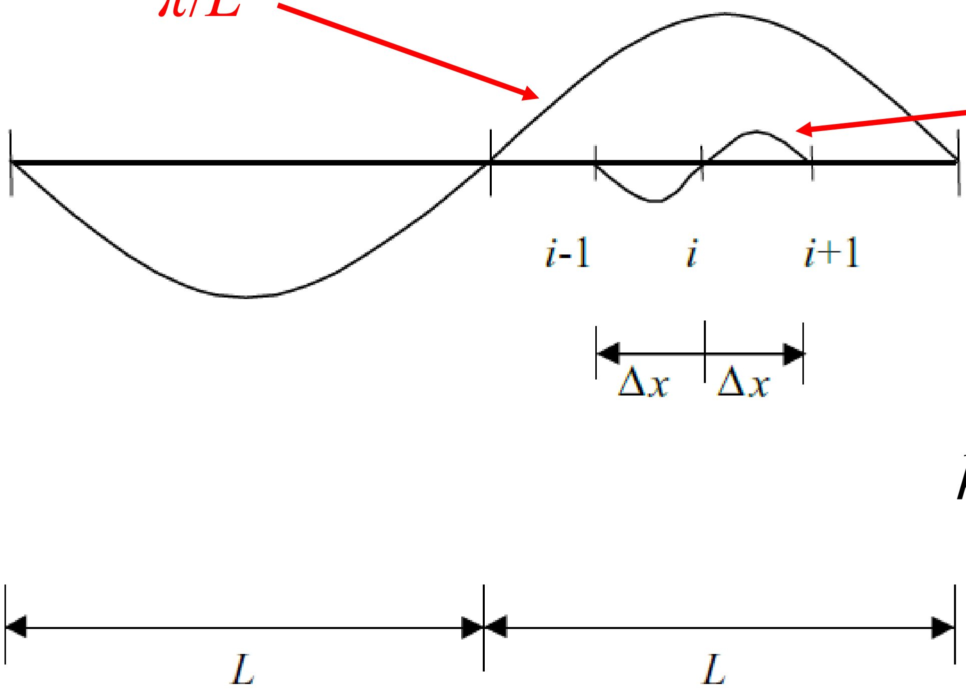
The fundamental frequency in a one-dimension domain between $-L$ and L is the maximum wave length $2L$.

Von Neumann Stability Analysis

$$\frac{\varepsilon_i^{n+1} - \varepsilon_i^n}{\Delta t} = \alpha \left(\frac{\varepsilon_{i+1}^n - 2\varepsilon_i^n + \varepsilon_{i-1}^n}{\Delta x^2} \right)$$

Wave number is minimum

π/L



Wave number is maxed depending on Δx

$\pi/\Delta x$

Harmonics:

$$k_j = j\pi / L = j\pi / (N\Delta x)$$

$$j = 0,1,\ldots N$$

T.J. Chung (2002)

Von Neumann Stability Analysis

$$\frac{\varepsilon_i^{n+1}-\varepsilon_i^n}{\Delta t}=\alpha\left(\frac{\varepsilon_{i+1}^n-2\varepsilon_i^n+\varepsilon_{i-1}^n}{\Delta x^2}\right)$$

$$k_j = j\pi/L = j\pi/\left(N\Delta x\right) \qquad j=0,1,\ldots N$$ j-th Harmonic

Representing the error at n by a Fourier series with complex coefficients.

$$\varepsilon_i^n=\sum_{j=-N}^{N}\bar{\varepsilon}_j^n e^{Ik_j\left(i\Delta x\right)}=\sum_{j=-N}^{N}\bar{\varepsilon}_j^n e^{Iji\pi/N}$$

$I=\sqrt{-1}$ $\qquad \bar{\varepsilon}_j^n$ - Amplication of the j-th harmonic.

Setting a phase angle $\phi = j\pi/N$

$$\varepsilon_i^n=\sum_{j=-N}^{N}\bar{\varepsilon}_j^n e^{Ii\phi}$$

Von Neumann Stability Analysis

$$\frac{\varepsilon_i^{n+1}-\varepsilon_i^n}{\Delta t}=\alpha\left(\frac{\varepsilon_{i+1}^n-2\varepsilon_i^n+\varepsilon_{i-1}^n}{\Delta x^2}\right)$$

$$\varepsilon_i^n=\sum_{j=-N}^{N}\bar{\varepsilon}_j^{\,n}e^{Ii\phi}$$ $\bar{\varepsilon}_j^{\,n}$ - Amplification of the *j*-th harmonic.

Substituting and looking at a harmonic *j*,

$$\frac{\bar{\varepsilon}^{\,n+1}-\bar{\varepsilon}^{\,n}}{\Delta t}e^{Ii\phi}=\alpha\left(\frac{\bar{\varepsilon}^{\,n}e^{I(i+1)\phi}-2\bar{\varepsilon}^{\,n}e^{Ii\phi}+\bar{\varepsilon}^{\,n}e^{I(i-1)\phi}}{\Delta x^2}\right)$$

$$\bar{\varepsilon}^{\,n+1}-\bar{\varepsilon}^{\,n}=d\left(\frac{\bar{\varepsilon}^{\,n}e^{I(i+1)\phi}-2\bar{\varepsilon}^{\,n}e^{Ii\phi}+\bar{\varepsilon}^{\,n}e^{I(i-1)\phi}}{e^{Ii\phi}}\right)$$

$$\bar{\varepsilon}^{\,n+1}-\bar{\varepsilon}^{\,n}=d\left(\bar{\varepsilon}^{\,n}e^{I\phi}-2\bar{\varepsilon}^{\,n}+\bar{\varepsilon}^{\,n}e^{-I\phi}\right)$$

Von Neumann Stability Analysis

$$\overline{\varepsilon}^{n+1} - \overline{\varepsilon}^{n} = d\left(\overline{\varepsilon}^{n} e^{I\phi} - 2\overline{\varepsilon}^{n} + \overline{\varepsilon}^{n} e^{-I\phi}\right)$$

$\overline{\varepsilon}_j^{n}$ - Amplication of the *j*-th harmonic.

How is stability defined?

$$|g| = \left|\frac{\overline{\varepsilon}^{n+1}}{\overline{\varepsilon}^{n}}\right| \leq 1$$

For all ϕ.

Von Neumann Stability Analysis

$$\overline{\varepsilon}^{n+1} - \overline{\varepsilon}^{n} = d\left(\overline{\varepsilon}^{n} e^{I\phi} - 2\overline{\varepsilon}^{n} + \overline{\varepsilon}^{n} e^{-I\phi}\right)$$

$$|g| = \left|\frac{\overline{\varepsilon}^{n+1}}{\overline{\varepsilon}^{n}}\right|$$

$$\overline{\varepsilon}^{n+1} = \overline{\varepsilon}^{n} + d\left(\overline{\varepsilon}^{n} e^{I\phi} - 2\overline{\varepsilon}^{n} + \overline{\varepsilon}^{n} e^{-I\phi}\right)$$

$$\frac{\overline{\varepsilon}^{n+1}}{\overline{\varepsilon}^{n}} = \frac{\overline{\varepsilon}^{n} + d\left(\overline{\varepsilon}^{n} e^{I\phi} - 2\overline{\varepsilon}^{n} + \overline{\varepsilon}^{n} e^{-I\phi}\right)}{\overline{\varepsilon}^{n}} = g$$

$$g = 1 + d\left(e^{I\phi} - 2 + e^{-I\phi}\right)$$

Von Neumann Stability Analysis

$$g = 1 + d\left(e^{I\phi} - 2 + e^{-I\phi}\right)$$

Recall $e^{I\phi} = \cos\phi + I\sin\phi$

$$g = 1 + d\left(\cos\phi + I\sin\phi - 2 + \cos\phi - I\sin\phi\right)$$

$$g = 1 + 2d\left(\cos\phi - 1\right)$$

$$\left|g\right| \le 1$$

$$1 + 2d\left(\cos\phi - 1\right) \le 1 \qquad 1 + 2d\left(\cos\phi - 1\right) \ge -1$$

Von Neumann Stability Analysis

$$-1 \le 1 + 2d(\cos\phi - 1) \le 1$$

What is the maximum possible value for $\cos\phi - 1$?

$$1 + 2d(-2) \ge -1$$

Test:

$$1 + 2d(-2) \ge -1$$

$$-4d \ge -2 \qquad d \le 0.5$$

$$\boxed{d = \frac{\alpha \Delta t}{\Delta x^2} \le 0.5} \quad \text{for} \quad \frac{\partial T}{\partial t} - \alpha\left(\frac{\partial^2 T}{\partial x^2}\right) = 0$$

Von Neumann Stability Analysis

1-D Diffusion

$$\boxed{d = \frac{\alpha \Delta t}{\Delta x^2} \leq 0.5} \quad \text{for} \quad \frac{\partial T}{\partial t} - \alpha \left(\frac{\partial^2 T}{\partial x^2} \right) = 0$$

2-D Diffusion

$$\frac{\partial T}{\partial t} - \alpha \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) = 0$$

If grid-spacing along x and y are equal, what is the stable range for d?

$$\boxed{d = \frac{\alpha \Delta t}{\Delta x^2} \leq 0.25}$$

Summary

$$\frac{\partial T}{\partial t} - \alpha\left(\frac{\partial^2 T}{\partial x^2}\right) = 0$$

Generalized form of the finite difference equation

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \alpha\left[\underbrace{\frac{\beta\left(T_{i+1}^{n+1} - 2T_i^{n+1} + T_{i-1}^{n+1}\right)}{\Delta x^2}}_{\text{implicit}} + \underbrace{\frac{(1-\beta)\left(T_{i+1}^{n} - 2T_i^{n} + T_{i-1}^{n}\right)}{\Delta x^2}}_{\text{explicit}}\right]$$

If $\beta = 0$, FTCS method.

If $\beta = 1/2$, Crank-Nicolson scheme

For $0.5 \le \beta \le 1.0$, the method is unconditionally stable.

BURGER’S EQUATION

Target PDEs

- 2-D diffusion phenomena
  - November 2, 2017 (Thursday)
- 2-D Burger's equation
  - November 9, 2017 (Thursday)
- 1-D Advection (other numerical methods)
  - November 16, 2017 (Thursday)

BURGER’S EQUATION

Burger's Equation

- Mixed hyperbolic, parabolic, and elliptic.

1-Dimension

$$\frac{\partial u}{\partial t}+a\frac{\partial u}{\partial x}=v\frac{\partial^2 u}{\partial x^2}$$

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

2-Dimension

$$\frac{\partial u}{\partial t}+a\left(\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}\right)=v\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial u}{\partial t}+u\left(\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}\right)=v\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

Burger's Equation

Mission for the week

$$\frac{\partial u}{\partial t}+a\left(\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}\right)=v\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

Using forward-in-time and backward-in-space for the 1st derivative, and centered difference for the 2nd derivative, construct a numerical model for the Burger's equation. Decide on your own initial conditions and values for *v*.

1. Derive and write the discretization as an algebraic equation.
2. Investigate by modelling the differences when *a* (linear) is a constant and when *a* is *u* (non-linear) for a cyclic condition.
3. What happens when you set a Dirichlet boundary?
4. List potential applications for Burger's equation. Properly write the reference (e.g. T.J. Chung, 2002).

SUPPLEMENTARY

Final Report Outline

- Refer to OCW for template.

- To format code, use:
http://www.planetb.ca/syntax-highlight-word

Animating directly within python. How?

```
import numpy as np
import matplotlib.pyplot as plt
import os

directory  = os.path.dirname(os.path.realpath(__file__))
os.chdir(directory)

d = 0.1
alpha = 1.
grid_x = 100
grid_y = 100
nt = 100
dx = 0.01
dy = dx
dt = d*(dx**2)/alpha

x = np.linspace(0,dx*(grid_x-1),grid_x)
y = np.linspace(0,dy*(grid_y-1),grid_y)
X,Y = np.meshgrid(x,y)

#colorbar settings
cmap = plt.cm.get_cmap("jet")
#cmap.set_under
cmap.set_over('grey')

levels = np.arange(0.,32.,0.2)
#Cyclic boundary condition
T = np.zeros((grid_x,grid_y))
Tn = np.zeros((grid_x,grid_y))
T[90:99,50:60] = 30. #initial condition
icount = 1
for n in range(1,nt):
    plt.cla()
    plt
    Tn
    T=T                                                                  +
    plt                          Censored
    plt
    cl
    plt
    plt.text(np.max(x)*0.8,np.max(y)+dy,"t=%01.5f"%(dt*n))
    plt.savefig('cyclic_%04i.jpg'%(icount))
    icount += 1
```

Animating directly within python. How?

- Import the animation module.

  from matplotlib import animation

- Convert the time iteration into a function.

  for n in range(1,nt): → def animatefunc(n):
  global (changing variable)

- Call the animator.

  fig = plt.figure()

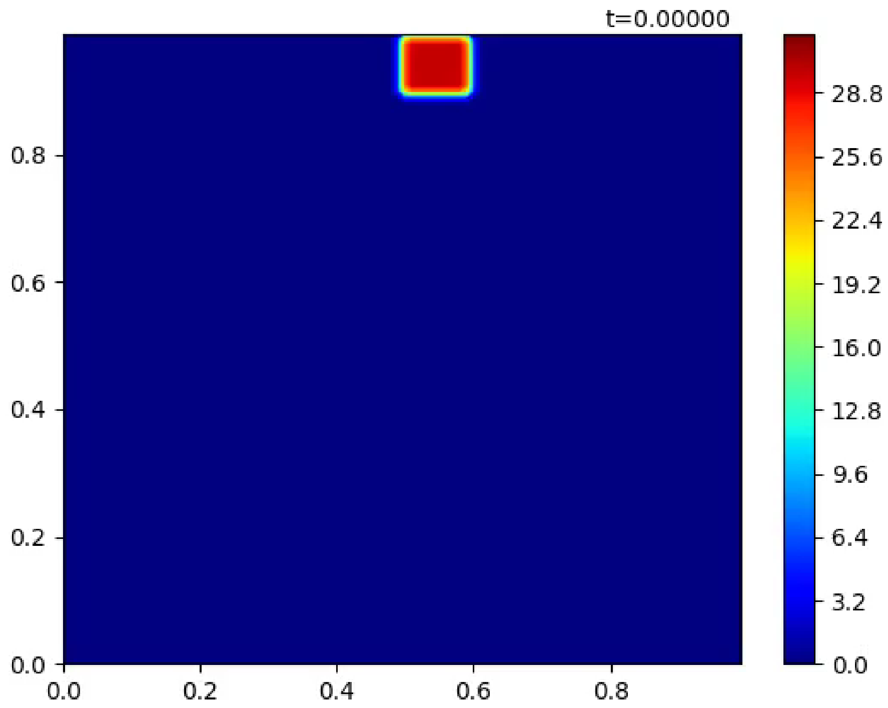
  animation.FuncAnimation(fig, animatefunc, frames=200, interval=20)

- Save the animation as mp4.

  a = animation.FuncAnimation(fig, animatefunc, frames=200, interval=20)

  a.save('test.mp4',fps=30,extra_args=['-vcodec','libx264'])

Animating directly within python. How?



Additional reference:

http://jakevdp.github.io/blog/2012/08/18/matplotlib-animation-tutorial/

Plotting a 3D surface instead?

- Import the module for 3D

  from mpl_toolkits.mplot3d import Axes3D
- Initiate a figure with 3D axes,

  fig = plt.figure()

  ax = fig.gca(projection='3d')
- In stead of using plt.plot directly, use ax objects instead:

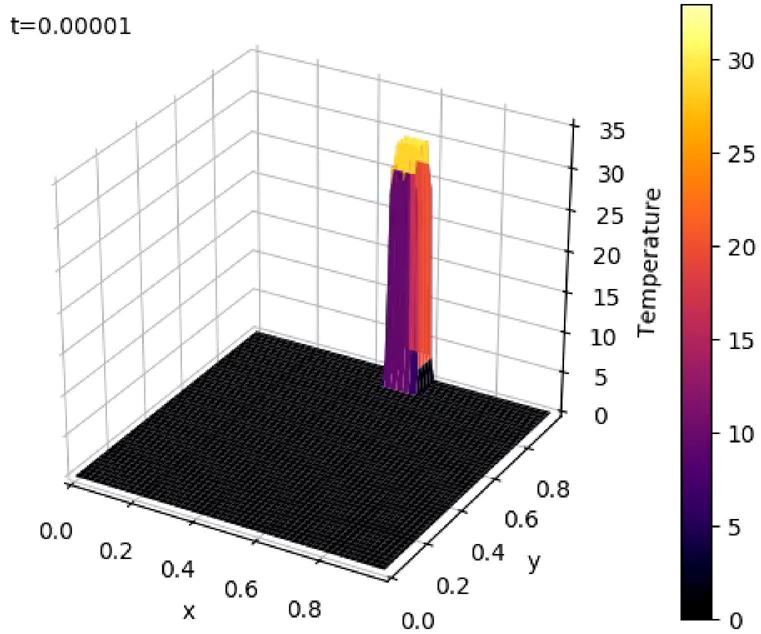
  ax.plot_surface(

  ax.set_ylim(

  ax.set_zlim(

  ax.text2D(
- Make sure to clear the figure as always after or before saving figure
- Set the angle of view: e.g. ax.view_init(elev=10., azim=10.)

Plotting a 3D surface instead?



Be creative!
There are a lot of resources online.

Reference:
https://matplotlib.org/examples/mplot3d/surface3d_demo.html

Linux shortcuts

- In addition to running python and ffmpeg.
- The following can be useful:

  move – “mv (file from) (file to)”

  copy – “cp (file from) (file to)”

  list files – “ls”

  Adding * as wildcard

  Compressing files by zip:

  e.g. zip group_images.zip ./*.jpg

Textbook reference

- Computational Fluid Dynamics by T.J. Chung
  - Downloadable via Tokyo Tech OPAC