Embedding Python into Core University Mathematics

Siri Chongchitnan
Warwick Mathematics Institute

TALMO May 2024

Outline

1. Core curriculum
2. Examples from various topics
3. Case study I: 1st-year Vector Calculus
4. Case study II: 3rd-year Problem Solving
5. Practical advice

A typical core maths curriculum (1st/2nd year)

- Pure: analysis, linear & abstract algebra
- Applied: calculus, geometry, dynamical systems*
- Probability and statistics
- Computing

** Dynamical systems, chaos and fractals will be covered in Stephen's talk*

Why Python?

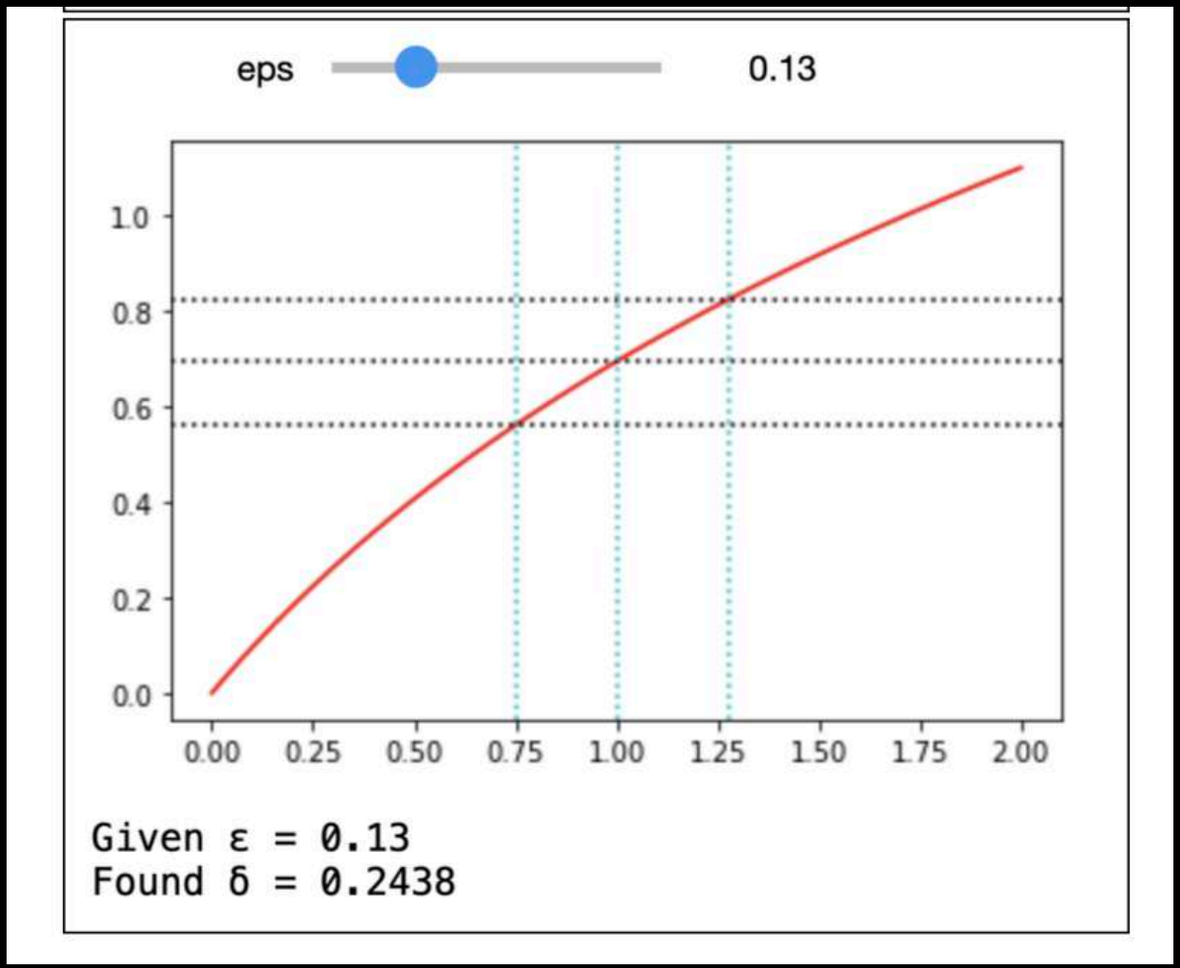
- Calculation: tedious tasks sped up
- Simulation: probabilistic situations, games
- Visualisation: graphs, 3D surfaces, animations
- Extension: use numerical evidence to form conjectures

Analysis

A function $f : A \to \mathbb{R}$ is said to be *continuous at a point* $x_0 \in A$ if, for all $\varepsilon > 0$, there exists $\delta > 0$ such that, for all $x \in A$, we have

$$|x - x_0| < \delta \implies |f(x) - f(x_0)| < \varepsilon.$$

Analysis



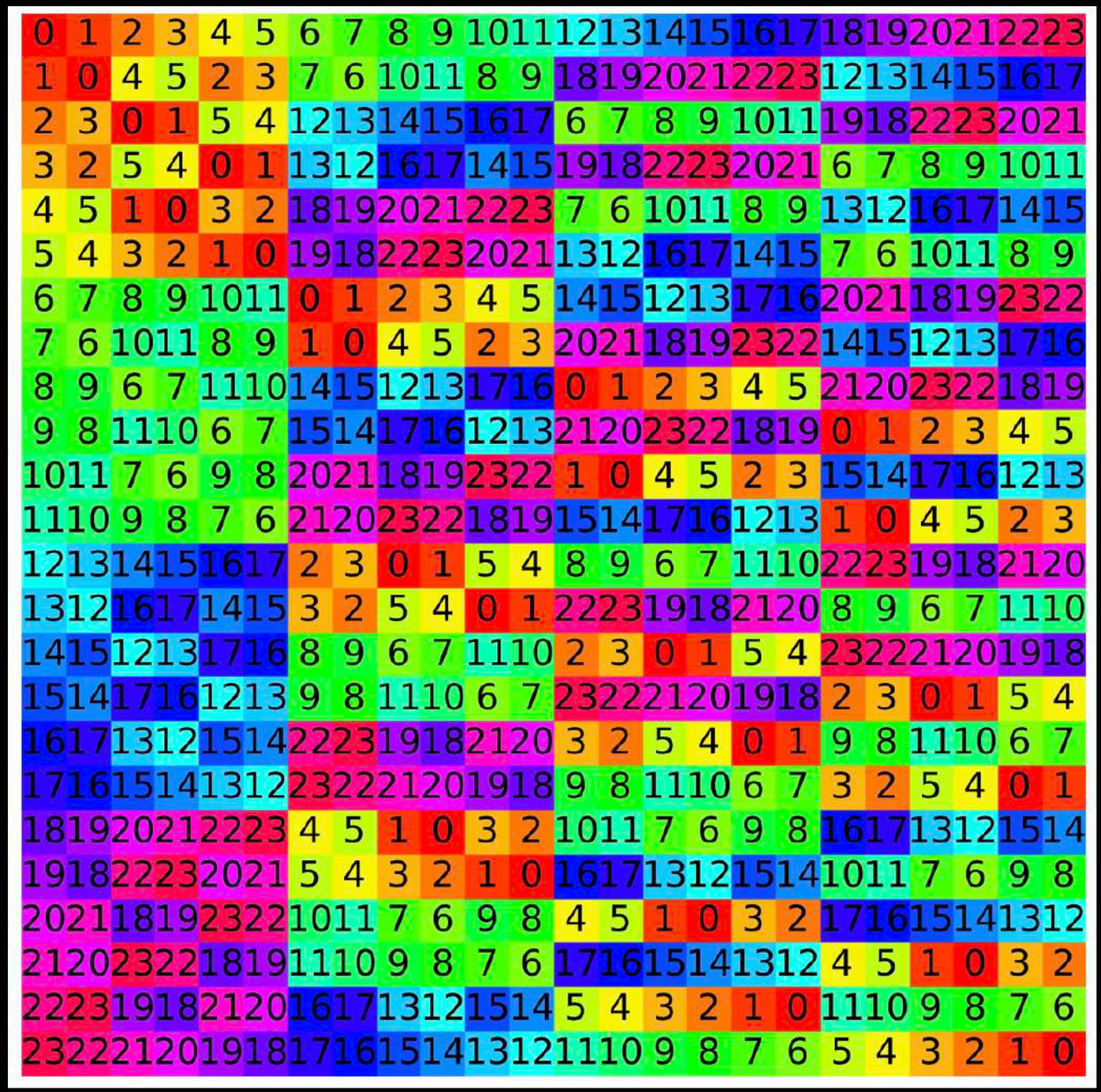
(code demo) The ε-δ game
f(x)=ln(x+1), at x=1

$$\delta_{\max} = 2(1 - e^{-0.13}) \approx 0.2438.$$

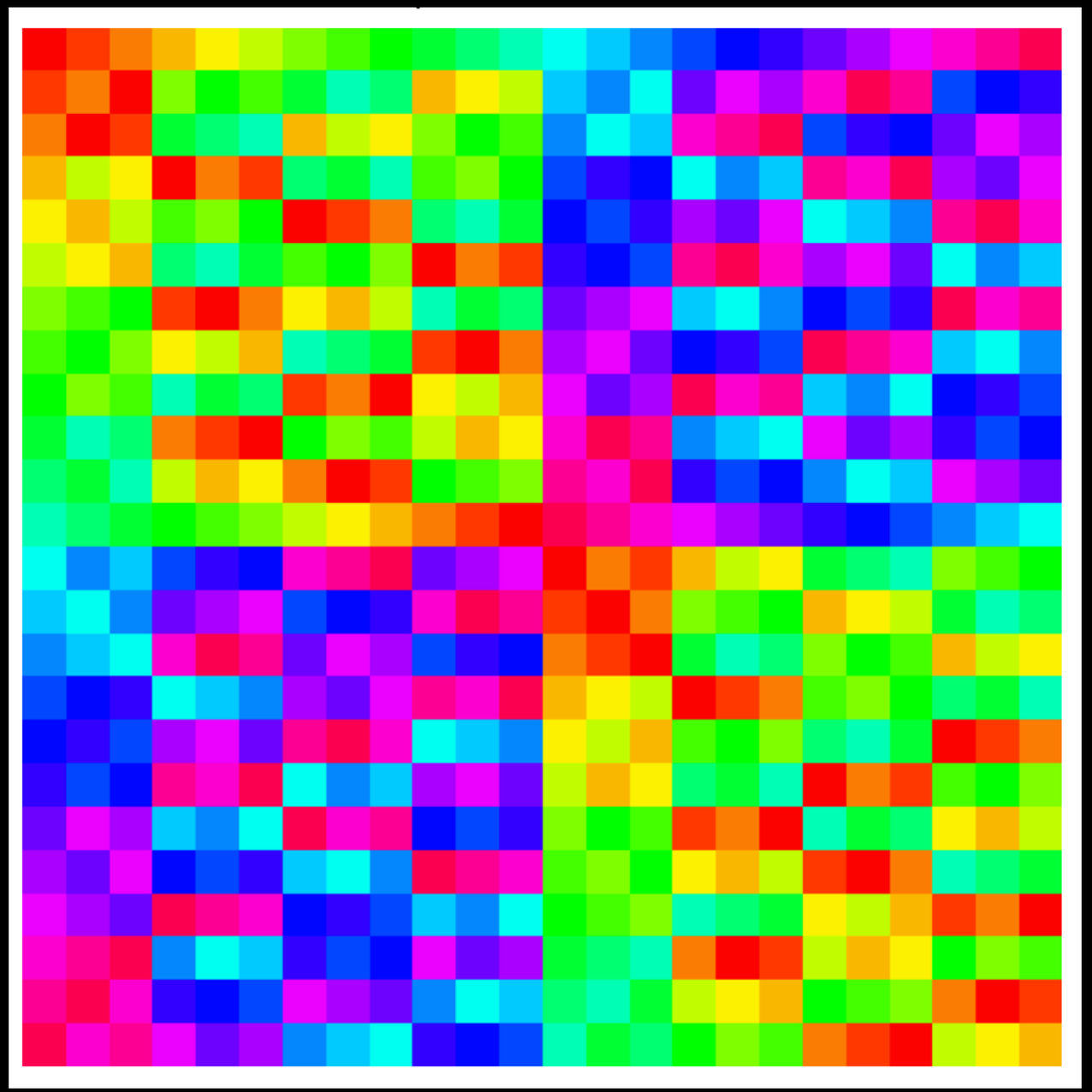
Group theory

Symmetric group S_4

index	element	parity	index	element	parity
0	e	E	12	(0 2 1)	E
1	(2 3)	O	13	(0 2 3 1)	O
2	(1 2)	O	14	(0 2)	O
3	(1 2 3)	E	15	(0 2 3)	E
4	(1 3 2)	E	16	(0 2)(1 3)	E
5	(1 3)	O	17	(0 2 1 3)	O
6	(0 1)	O	18	(0 3 2 1)	O
7	(0 1)(2 3)	E	19	(0 3 1)	E
8	(0 1 2)	E	20	(0 3 2)	E
9	(0 1 2 3)	O	21	(0 3)	O
10	(0 1 3 2)	O	22	(0 3 1 2)	O
11	(0 1 3)	E	23	(0 3)(1 2)	E

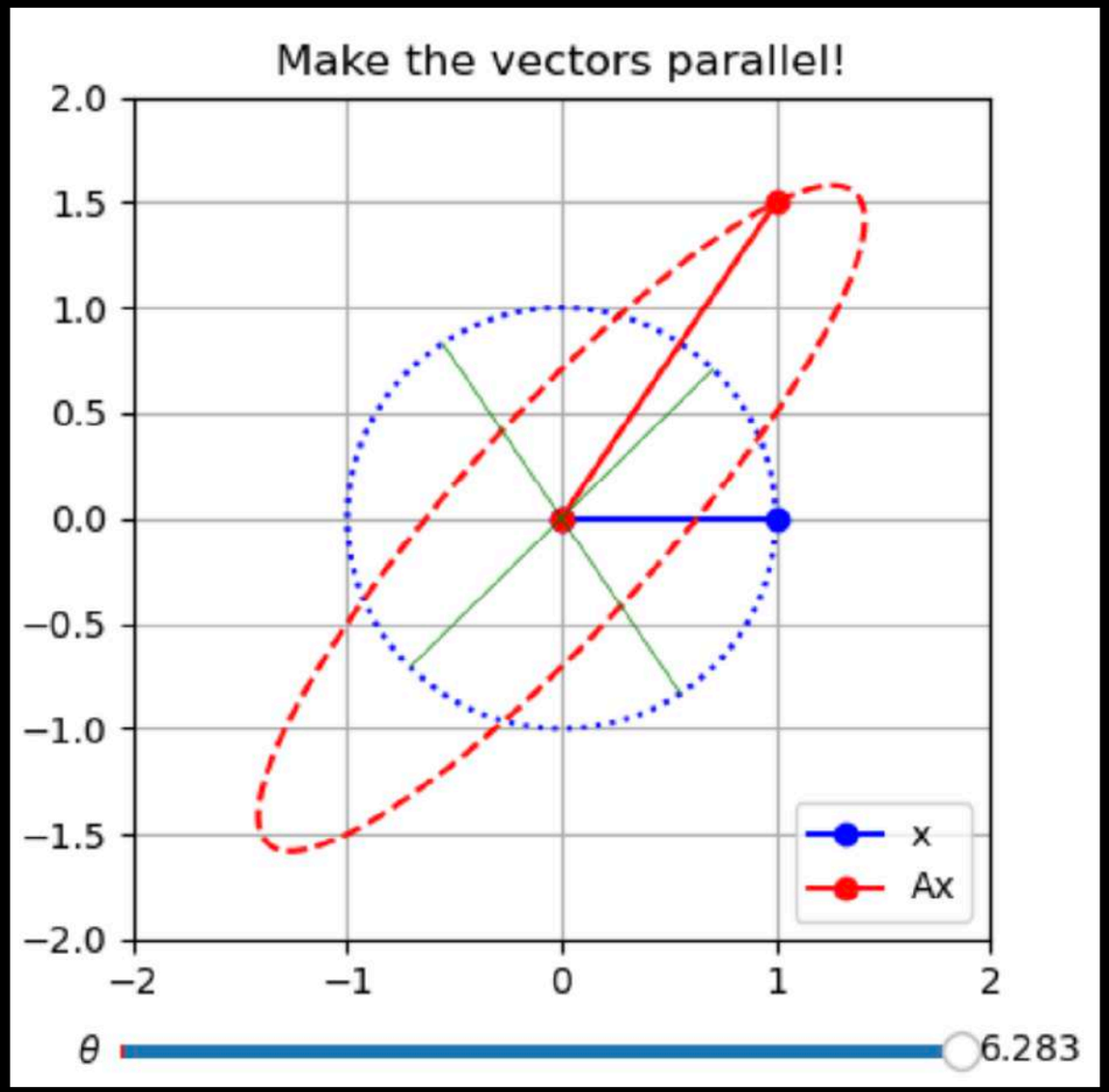


See any subgroups?



$$S_4/A_4 \cong \mathbb{Z}_2$$

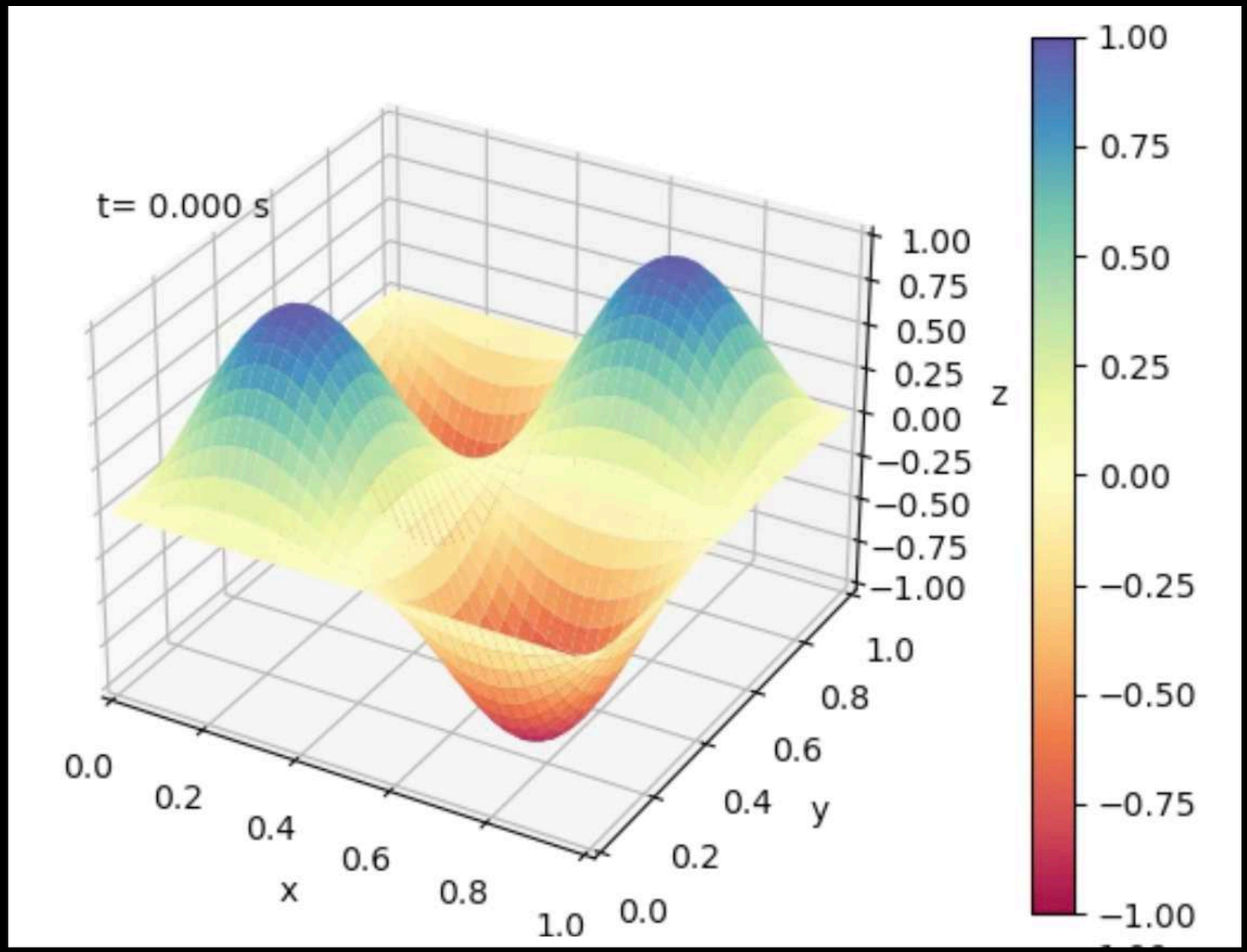
Linear algebra



$$A\mathbf{x} = \lambda\mathbf{x}, \text{ where } A = \begin{pmatrix} 1 & 1 \\ 3/2 & 1/2 \end{pmatrix}$$

(code demo)

PDE



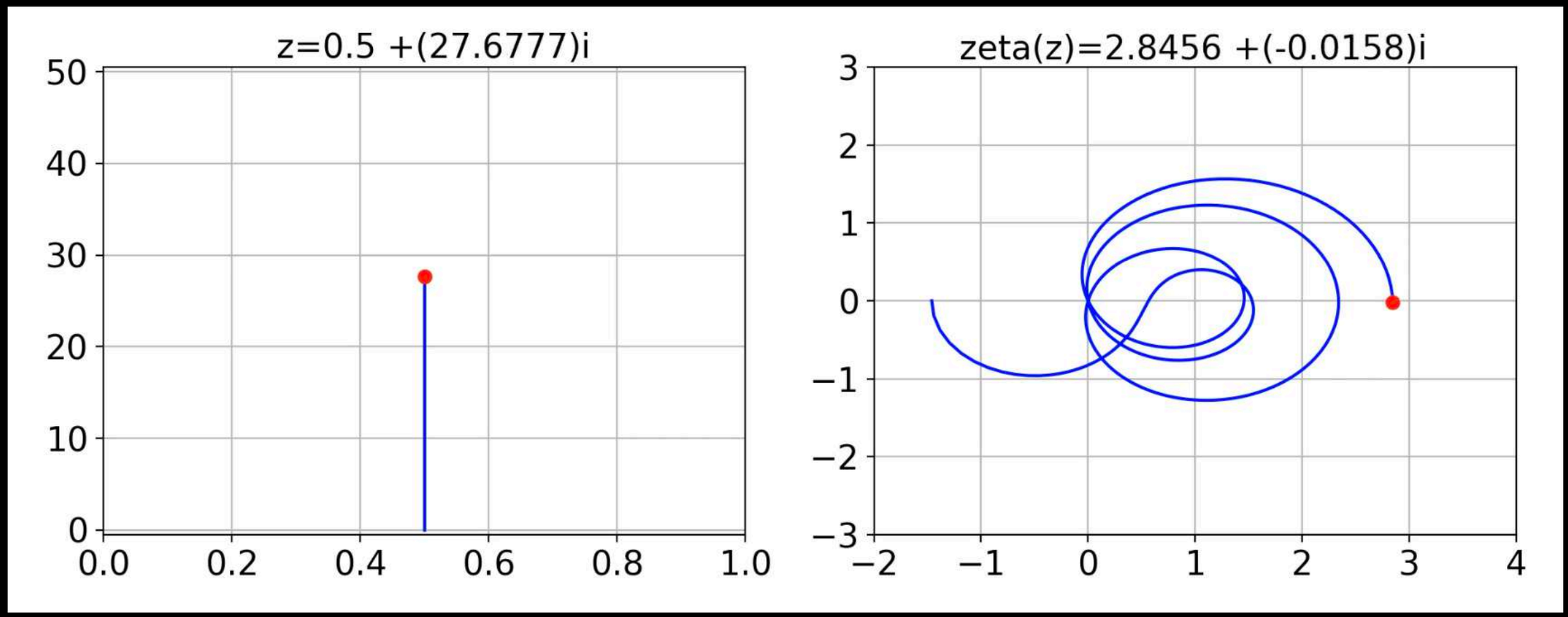
$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \quad (x, y) \in \mathcal{S},$$

with initial conditions $u(x, y, 0) = \sin(2\pi x)\sin(2\pi y)$,

$$\frac{\partial u}{\partial t}(x, y, 0) = 0,$$

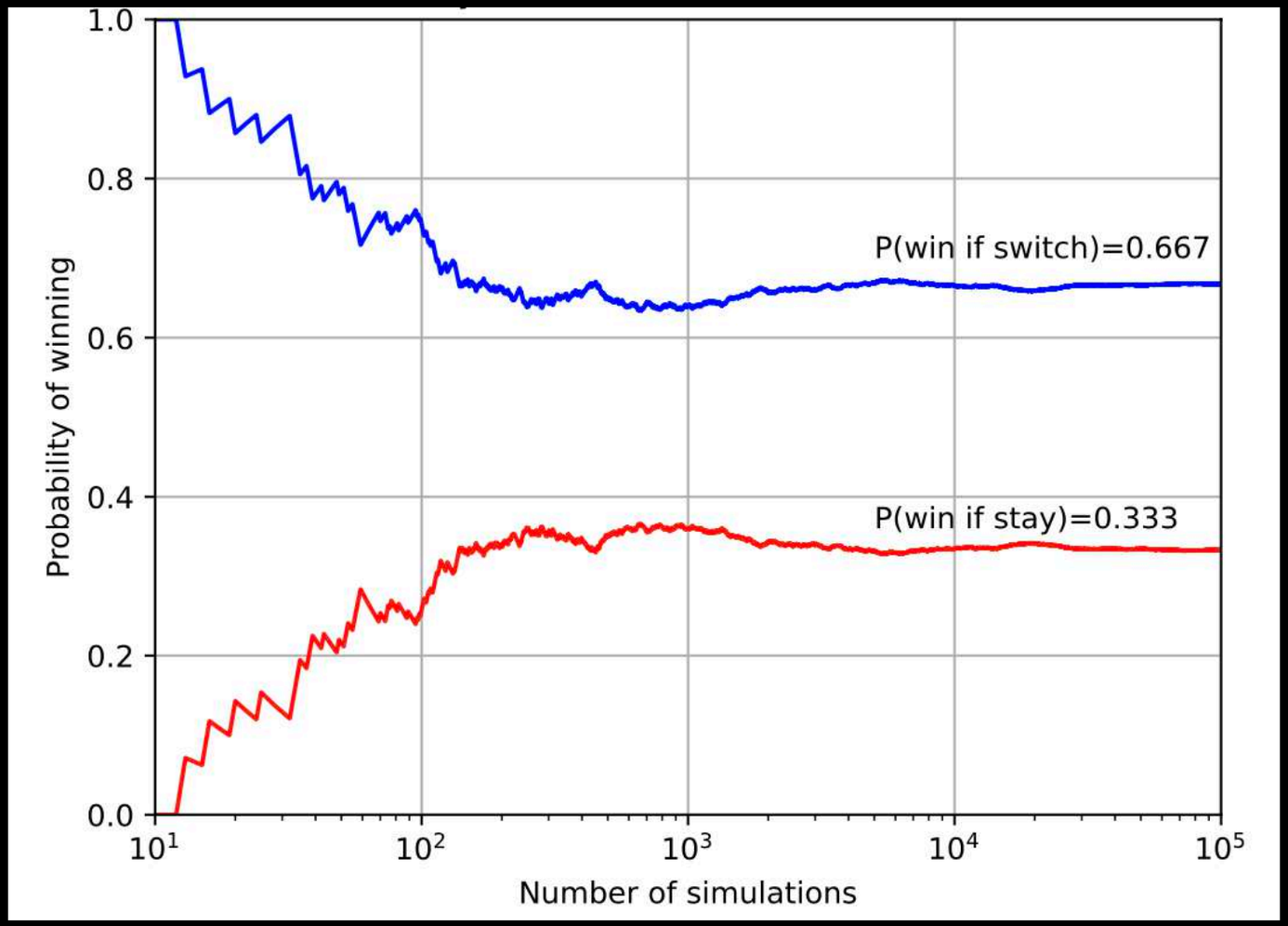
and boundary condition $u(x, y, t) = 0, \quad (x, y) \in \partial\mathcal{S},$

Complex analysis

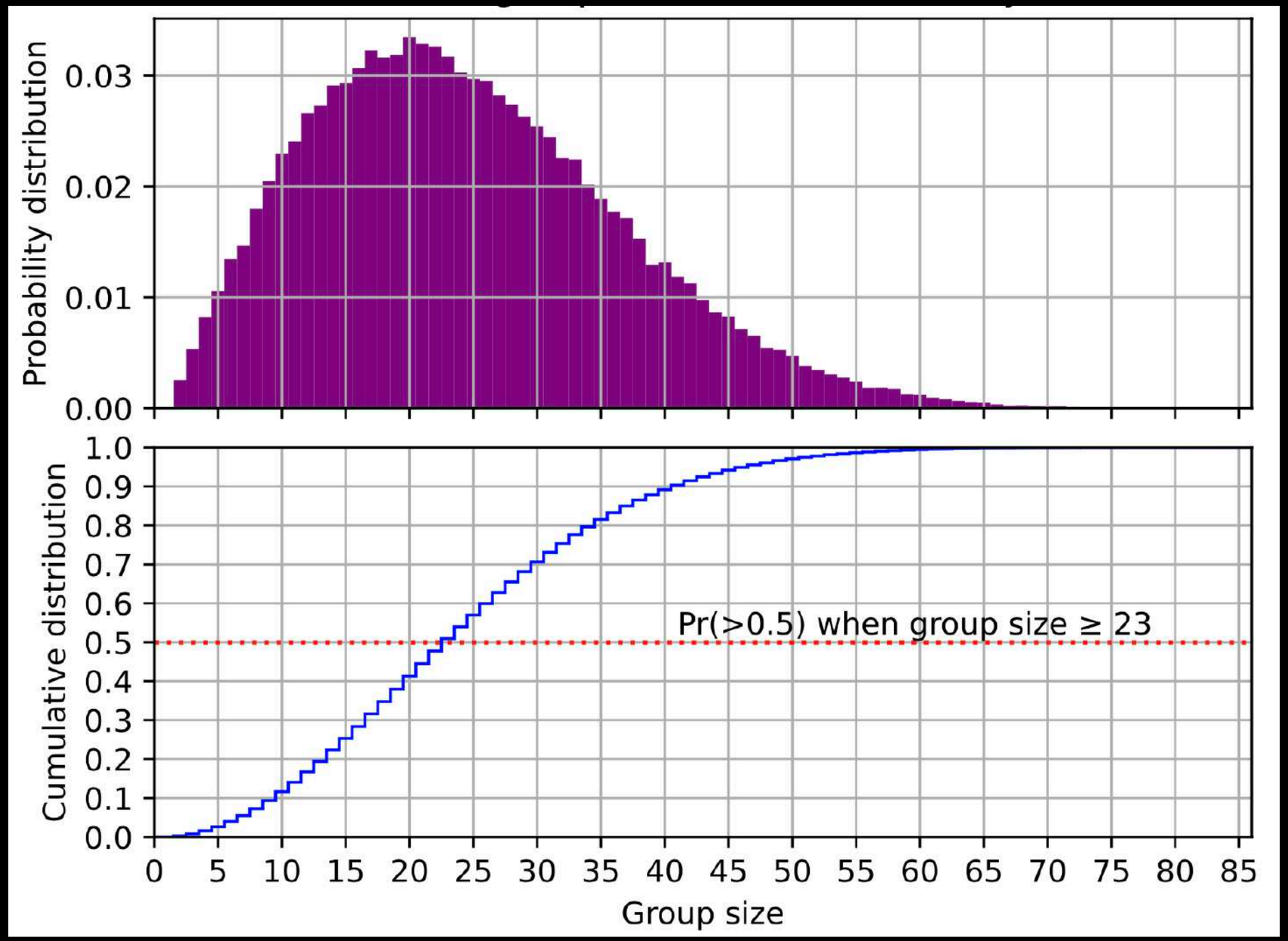


The zeta function

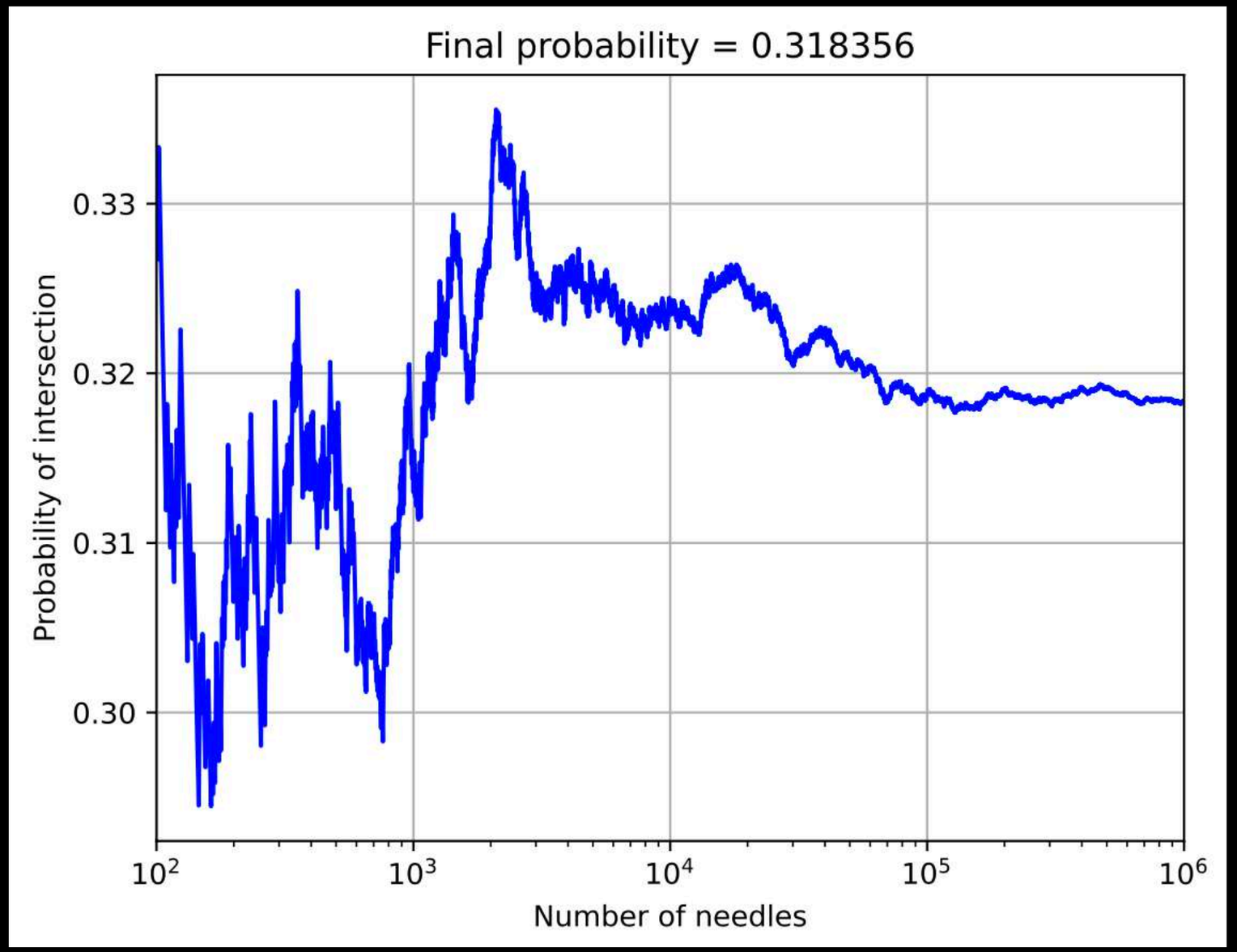
Probability



Probability



Probability



Case Study I
1st-year Vector Calculus

- Curves in 2D and 3D
- Partial differentiation, vector differentiation, grad div curl
- Surfaces, tangent planes, critical points
- Multiple integrals, volume and centre of mass, surface areas, Jacobian
- Line integral, vector fields, flux, circulation
- Integral theorems: Green's, Stokes' and Divergence

4 assignments, with one Python task in every assignment.

Python

3. A circle radius $r < 1$ is rolling on the inside of a circle radius 1, centred at the origin O. The centre C of the smaller circle is initially at $(1-r, 0)$. Let θ be the angle subtended by the line OC measured with respect to the positive x axis. The point P, initially at $(1, 0)$, traces out a curve as the smaller circle rolls inside the unit circle. The setup is shown below.

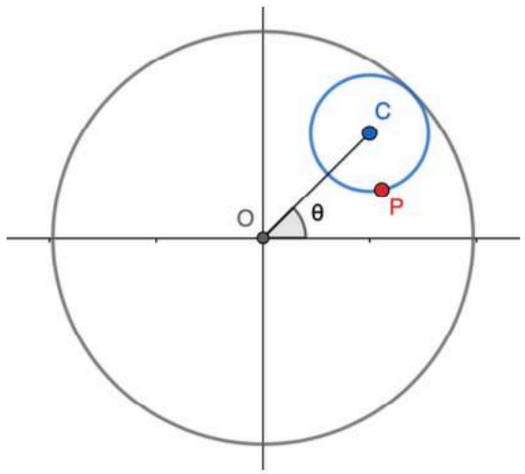


Figure 1: The point P traces out a curve as the smaller circle rolls within the larger one.

The curve traced out by P has the parametric equations:

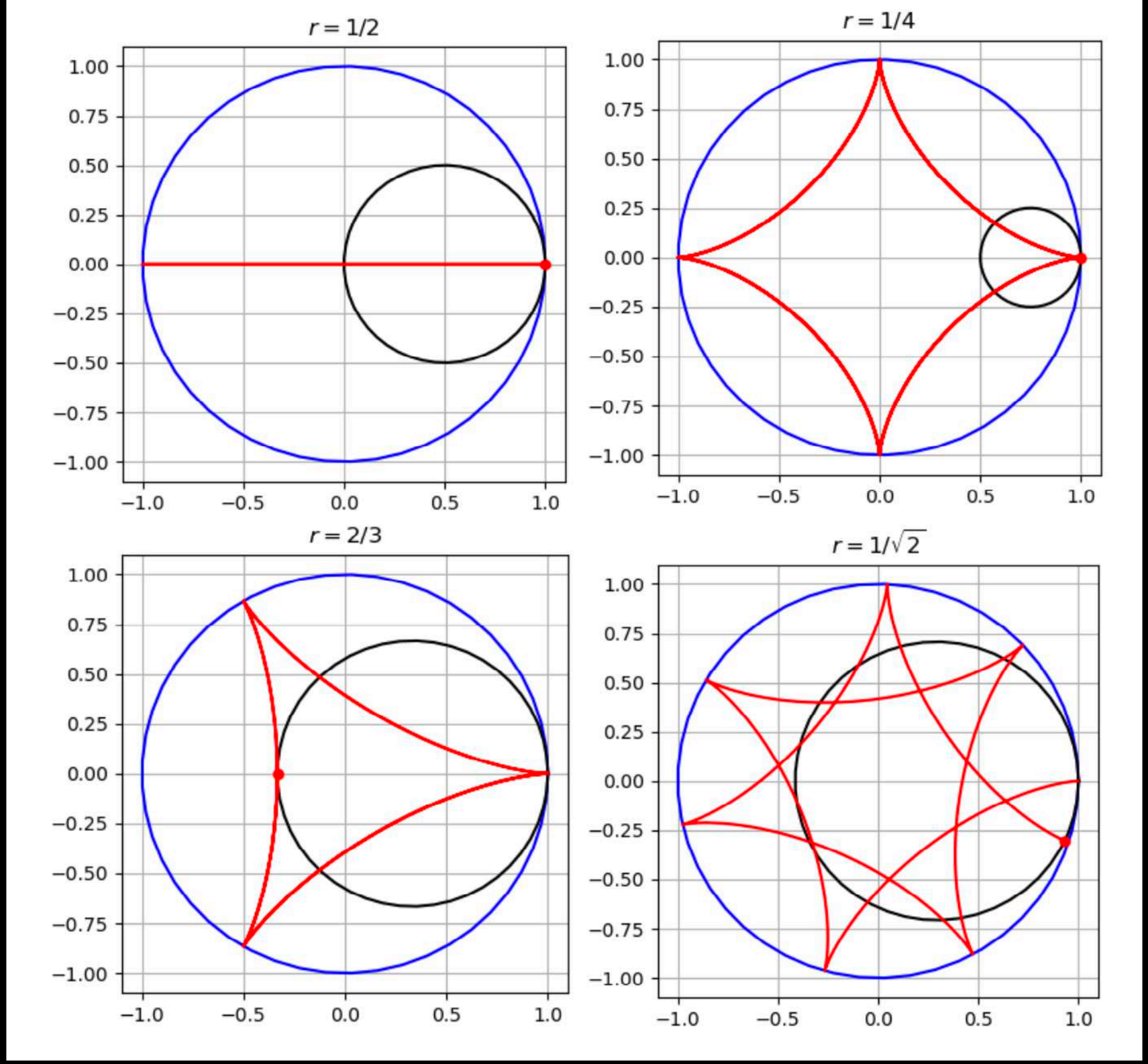
$$x(\theta) = (1-r)\cos\theta + r\cos\left(\frac{1-r}{r}\theta\right),$$
$$y(\theta) = (1-r)\sin\theta - r\sin\left(\frac{1-r}{r}\theta\right).$$

(a) Use Python to plot the curves corresponding to $r = \frac{1}{2}, \frac{1}{4}, \frac{2}{3}$ and $\frac{1}{\sqrt{2}}$, where $\theta \in [0, 10\pi]$. Plot them in 4 separate figures.

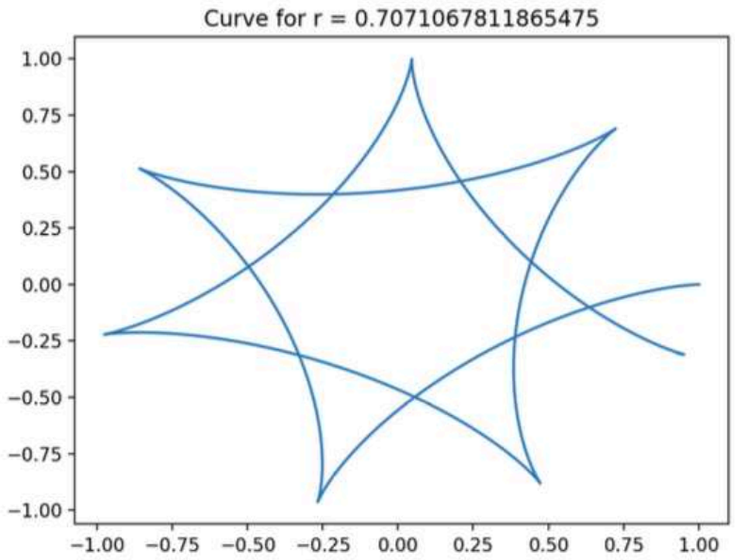
[4 marks]

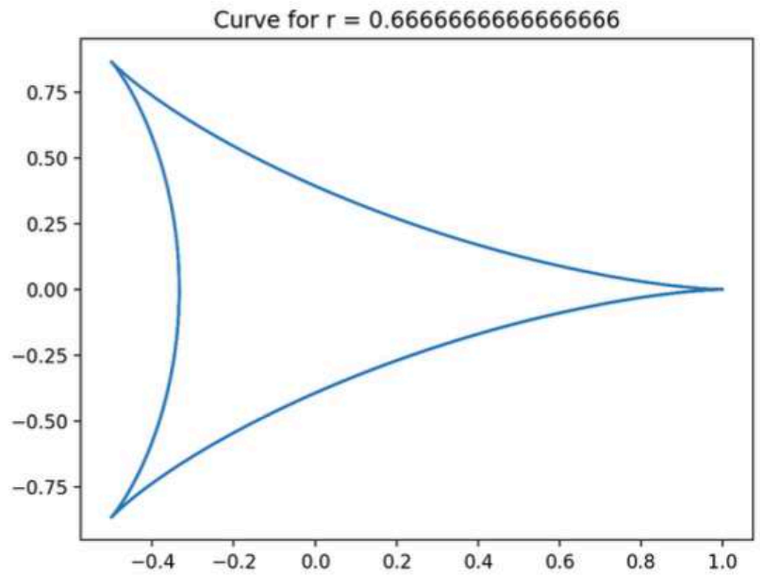
(b) Make one conjecture on the appearance of the curve for an arbitrary $r \in \mathbb{R}$.

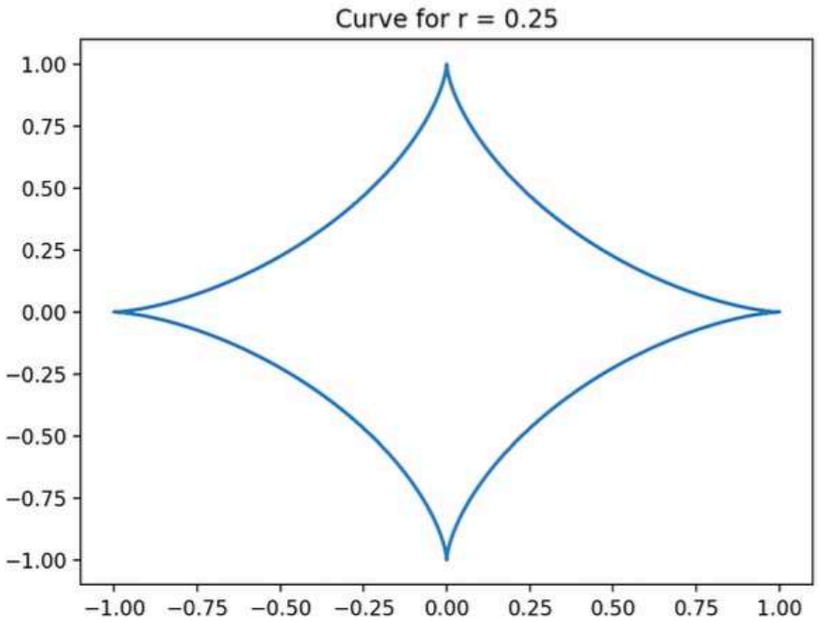
[1 mark]

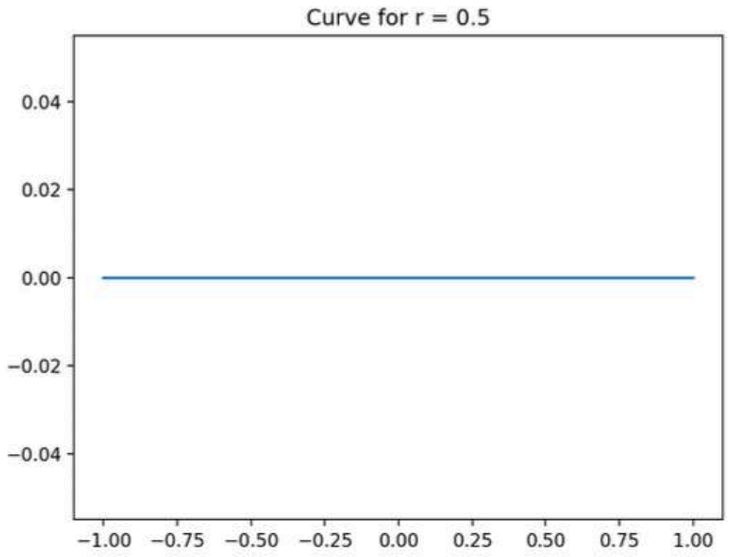


(Code demo)










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



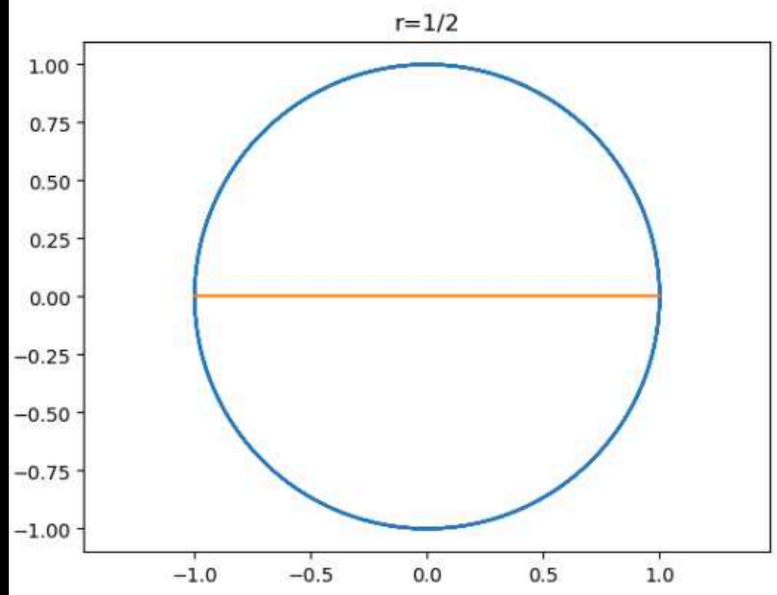
def function(r,p):
    x = (1-r)*np.cos(p) + r*np.cos(((1/r)-1)*p)
    y = (1-r)*np.sin(p) - r*np.sin(((1/r)-1)*p)
    return x,y
arrs = [1/2,1/4,2/3,1/((2)**0.5)]

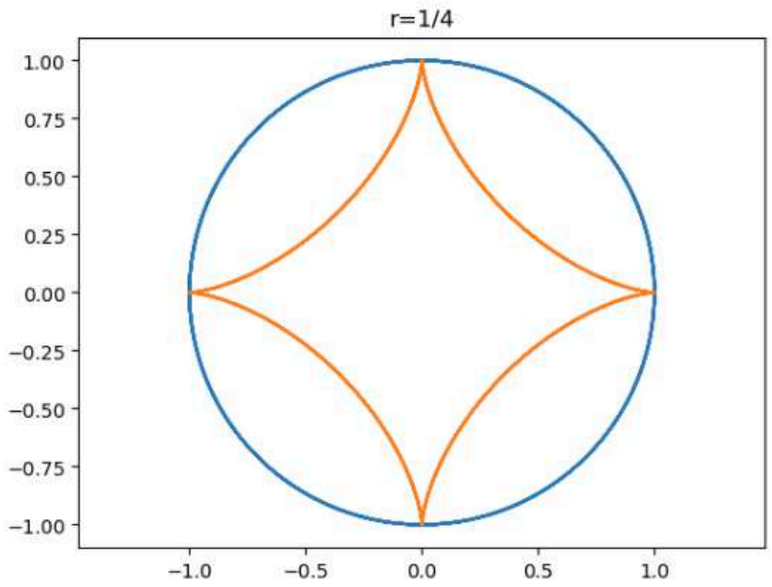
t = np.linspace(0,10*np.pi,1000)

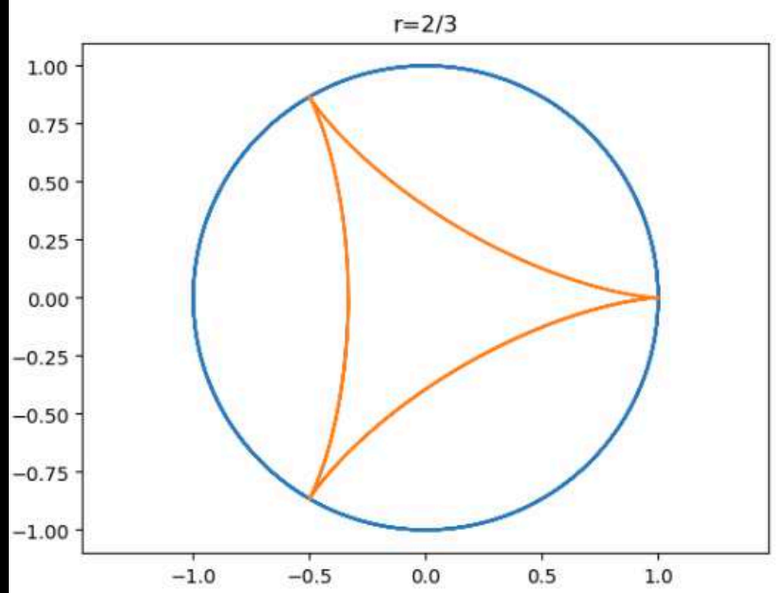
for i,j in enumerate(arrs):
    x_val,y_val = function(j,t)
    plt.figure(i)
    plt.plot(x_val, y_val)
    plt.title(f'Curve for r = {j}')

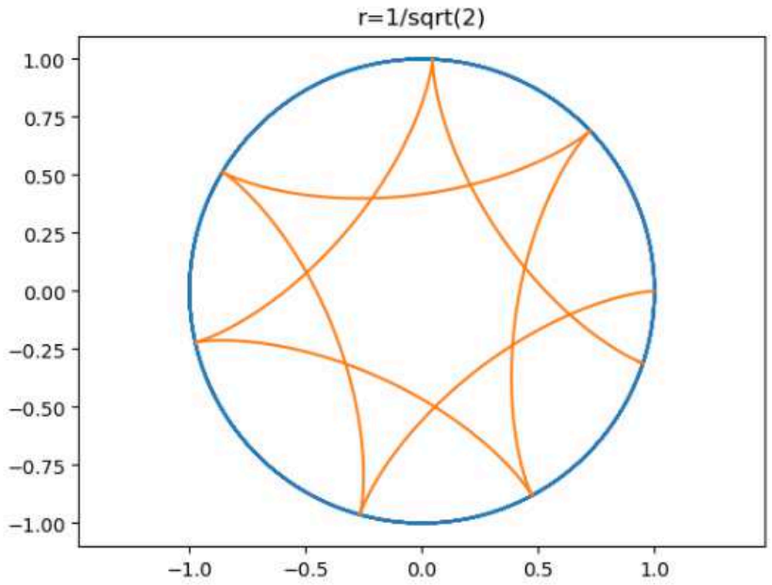
plt.show()
```

3b) The curve will not form a closed loop when the value of r is irrational, and will do so when r is rational.









Conjecture: If $r \in \mathbb{Q}$ i.e $r = \frac{p}{q}$, then the point P 'touches' the outer circle in q unique points.

(or $t \in [0, \infty)$) If $r \notin \mathbb{Q}$, then the point P touches the outer circle in infinitely many unique points.

Python

3. A *torus* has the following parametrisation:

$$\mathbf{r}(u,v) = ((2+\cos u)\cos v,\ (2+\cos u)\sin v,\ \sin u)\,, \quad \text{where } u,v \in [0,2\pi].$$

(a) Plot the torus using Python. See plotting tips below.
Include only screenshot(s) - do not submit the *.ipynb* file.
Make sure the aspect ratio of your plot is equal in all x, y, z directions so the torus doesn't appear stretched. Don't forget to label the axes.

[3 marks]

(b) Using the Divergence Theorem with $\mathbf{F}(x,y,z) = (x,0,0)$, calculate the volume of the torus.

[5 marks]

Plotting a parametric surface in Python

A parametric surface $\mathbf{r} : A \to \mathbb{R}^3$ is defined by

$$\mathbf{r}(u,v) = (x(u,v),\ y(u,v),\ z(u,v))\,,$$

where $(u,v) \in A \subseteq \mathbb{R}^2$.

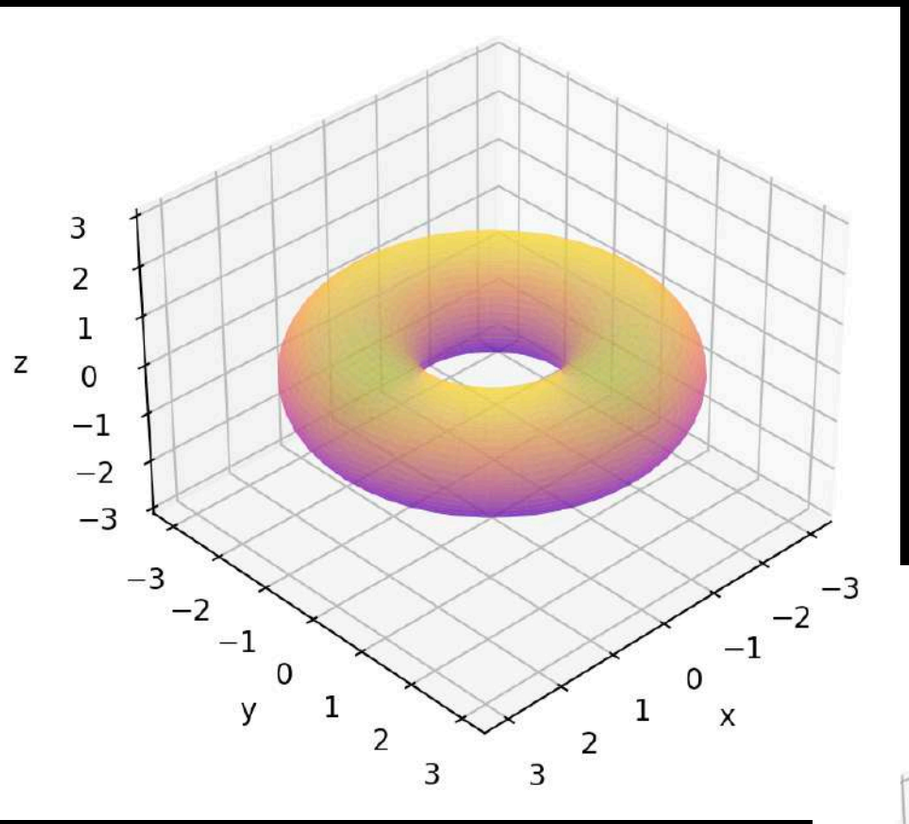
In the (minimal) code below, we use the command `meshgrid` to create all possible ordered pairs of u and v. The surface is created using the `plot_surface` command.

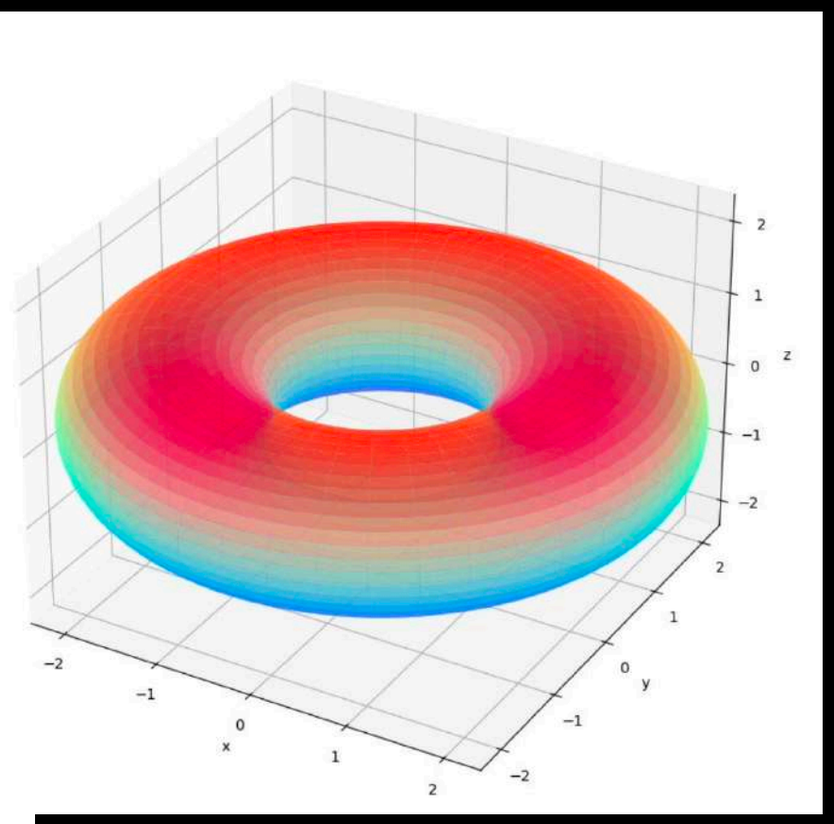
Plotting a parametric surface (minimal code)

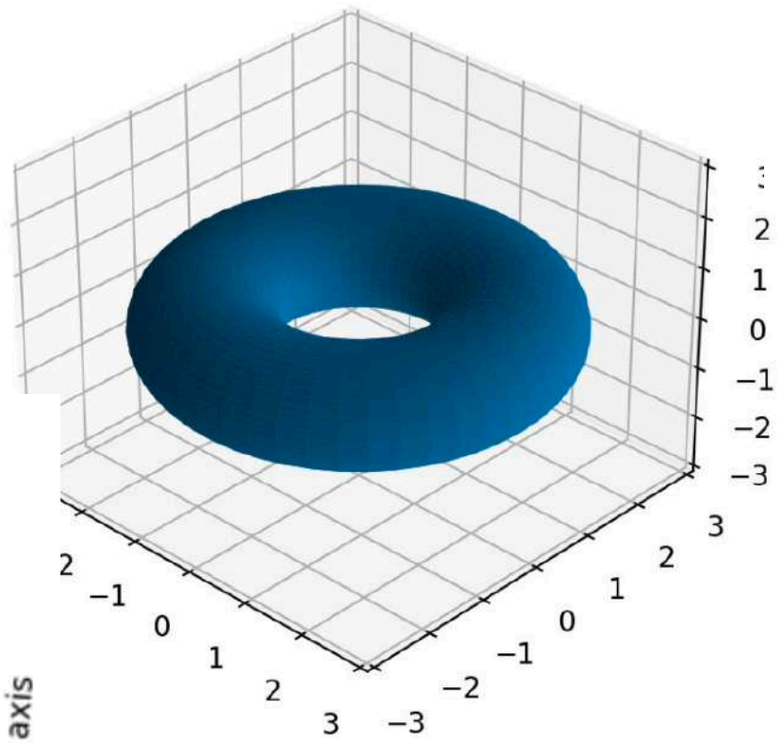
	`import numpy as np`
	`import matplotlib.pyplot as plt`
	`%matplotlib`

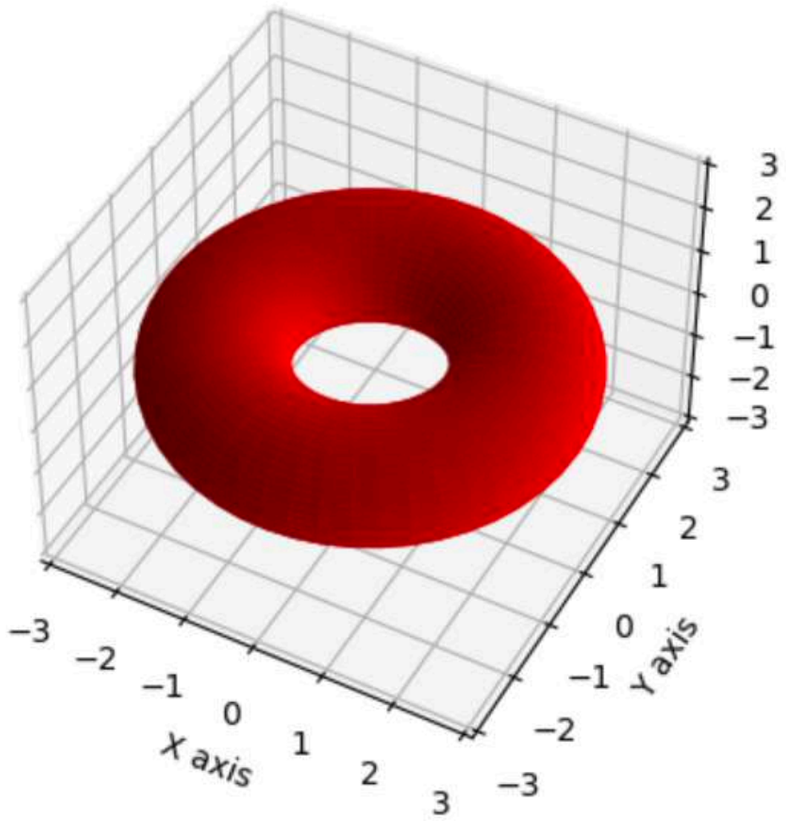
$u \in [-1,1]$
$v \in [0,2]$
Create 50×50 coordinate grid
	`u = np.linspace(-1,1)`
	`v = np.linspace(0,2)`
	`U, V = np.meshgrid(u,v)`

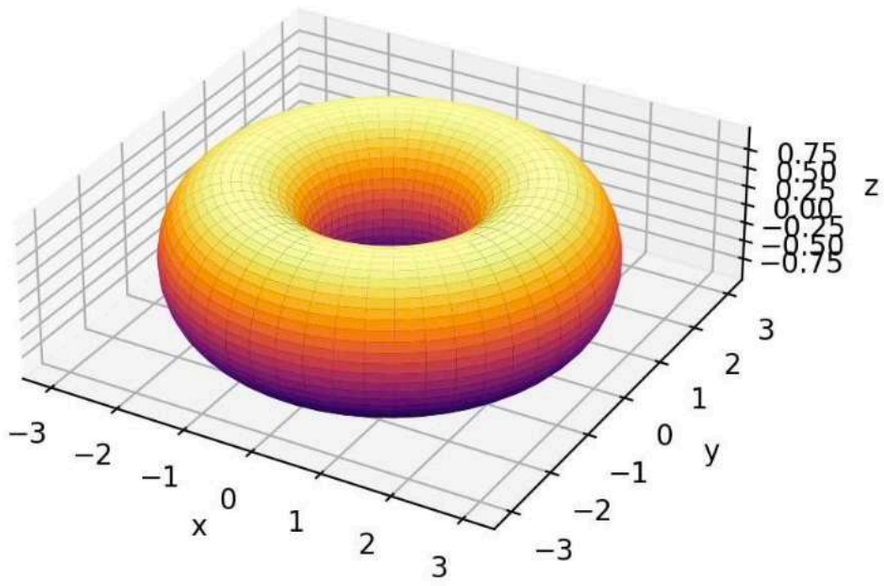
Plot $\mathbf{r}(u,v) = (u,\ 2v,\ 1-u-v)$
(a plane)
	`ax = plt.axes(projection='3d')`
	`ax.plot_surface(U, 2*V, 1-U-V)`
	`plt.show()`


z
y
x

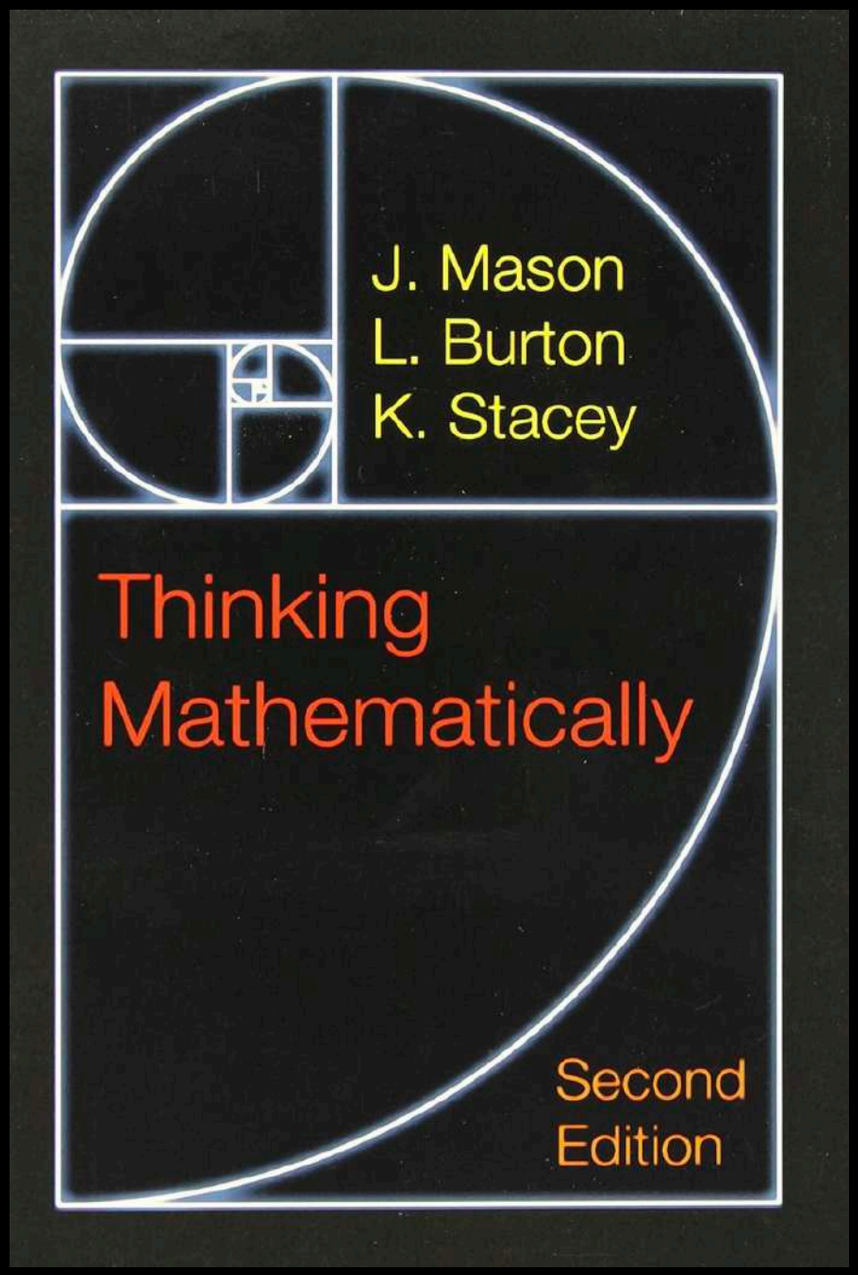

z
y
x

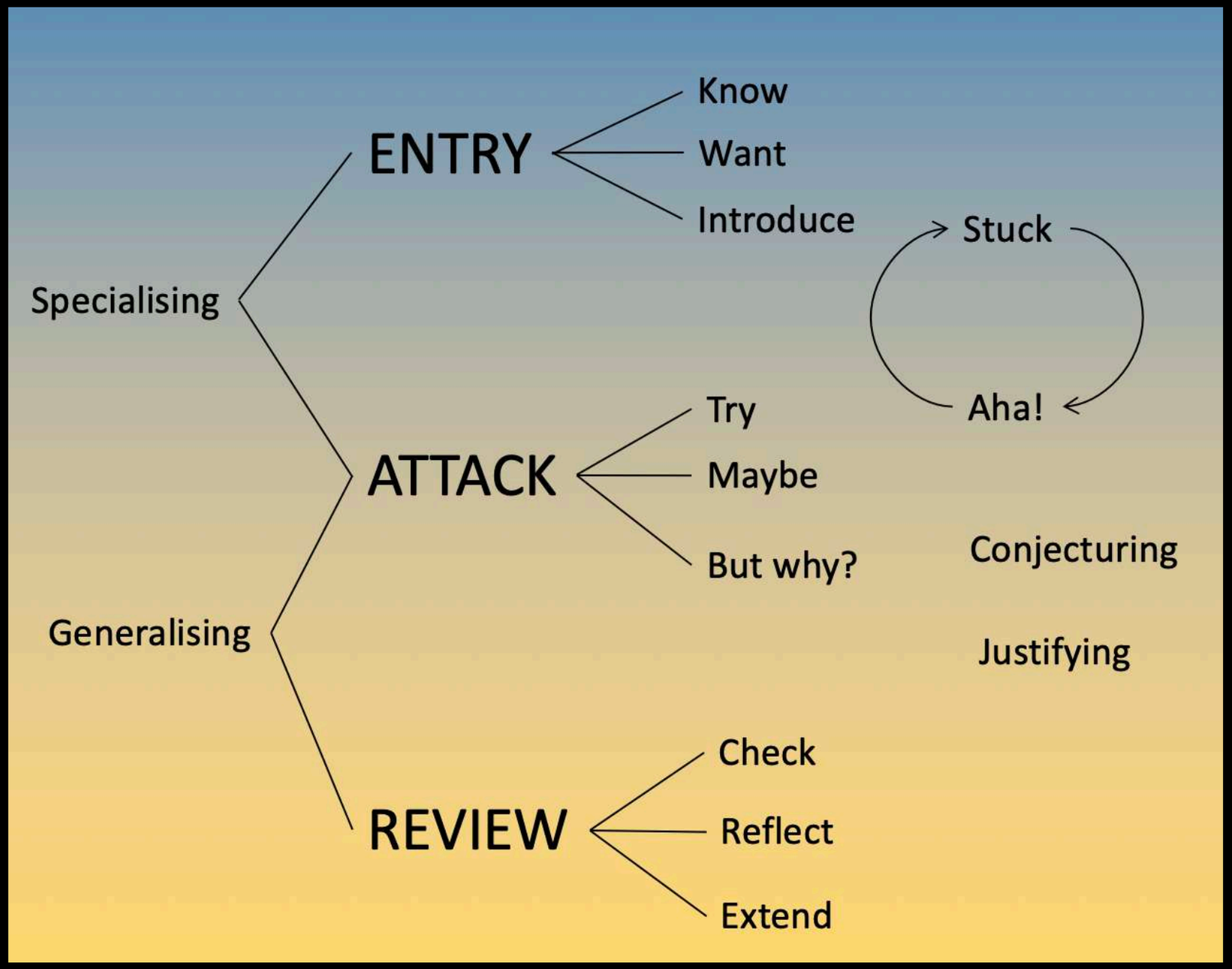

Z axis


X axis
Y axis


z
y
x

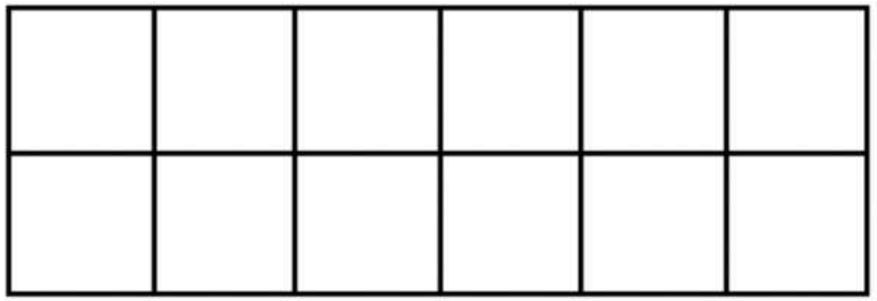
Case Study II
3rd-year Problem Solving

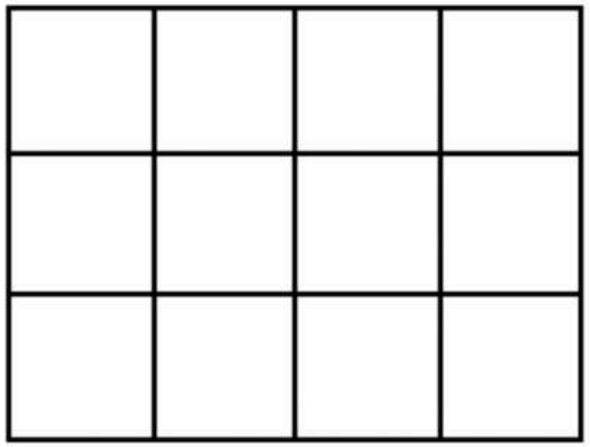




Lines and Squares

Consider the pictures below.





In the first picture, there are 10 straight lines and 17 squares.

In the second picture, there are 9 straight lines and 20 squares.

Find the smallest number of lines needed to make exactly 100 squares.

Python-led extension

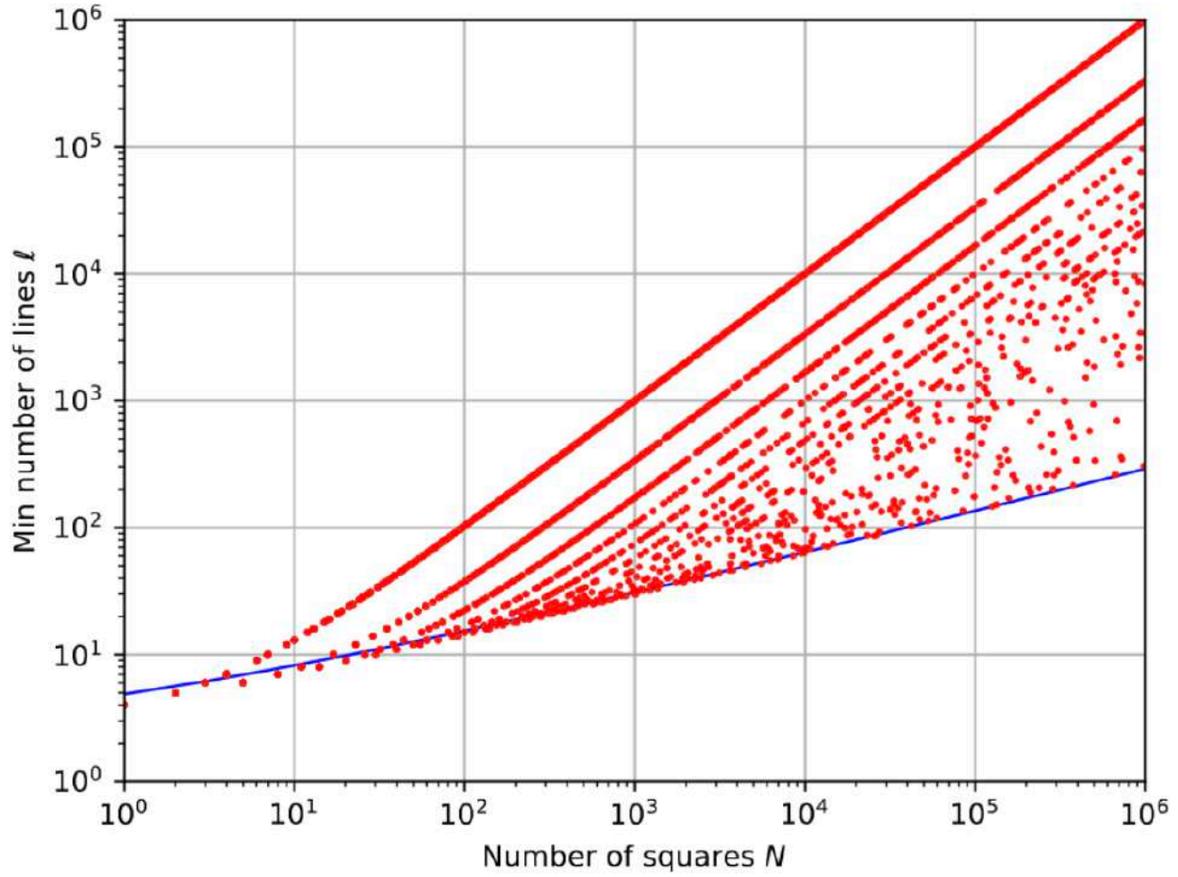


Figure 1: Scatter plot of $\ell_{\min}$ (minimum number of lines) against N (number of squares required). 5000 points are plotted. The equation of the blue line is given by eq. 4.

$$\ell = 2(3N)^{1/3} + 2. \tag{4}$$

We can see from the figure that this is a very good estimate of the lower bound.

In a big bowl, there are a number of strings all jumbled up. The strings are of the same length.

You put on a blindfold and reach into the bowl of strings. You randomly choose two ends and tie them together. You repeat this process of tying two ends a number of times.

You then remove your blindfold and find that some closed string loops have been formed.

Investigate.

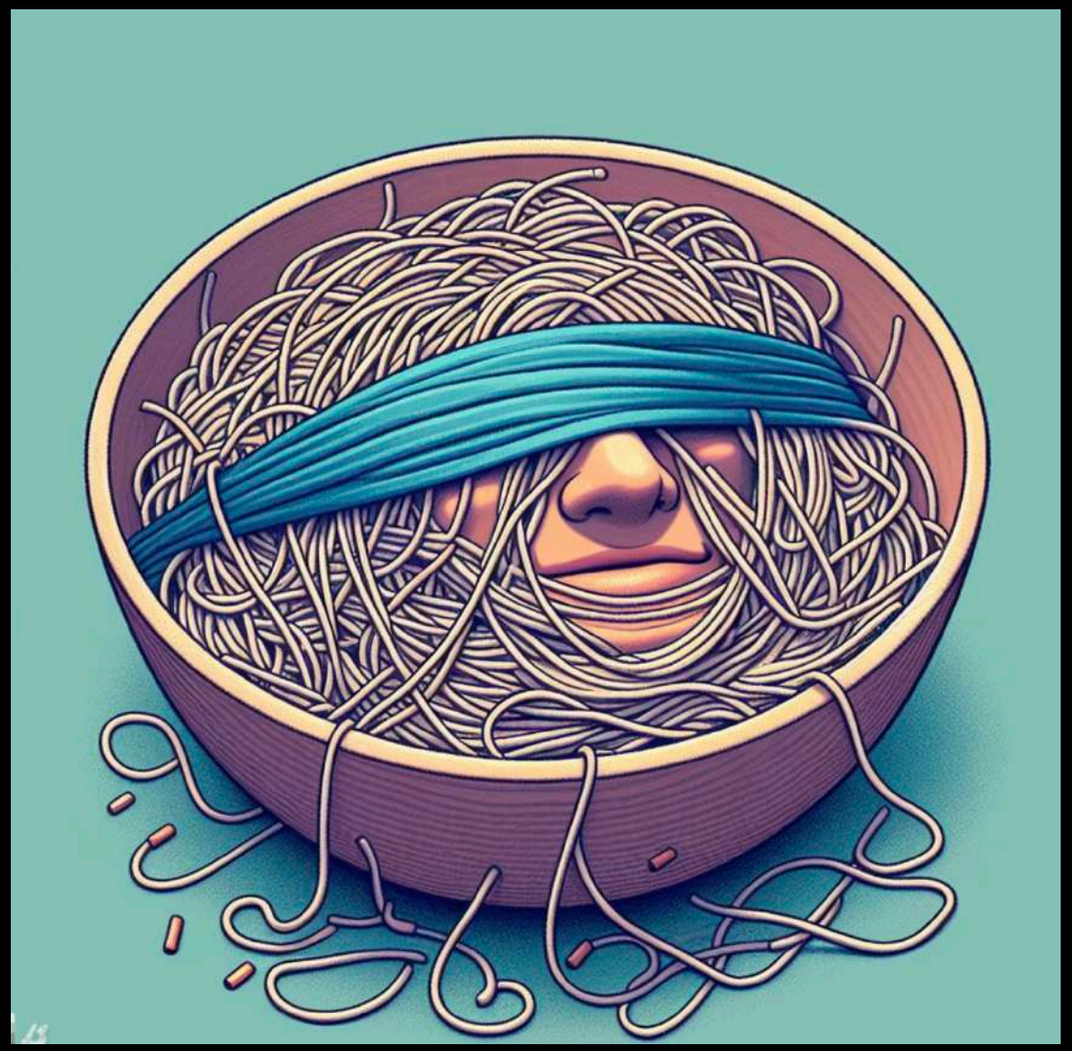


In a big bowl, there are a number of strings all jumbled up. The strings are of the same length.

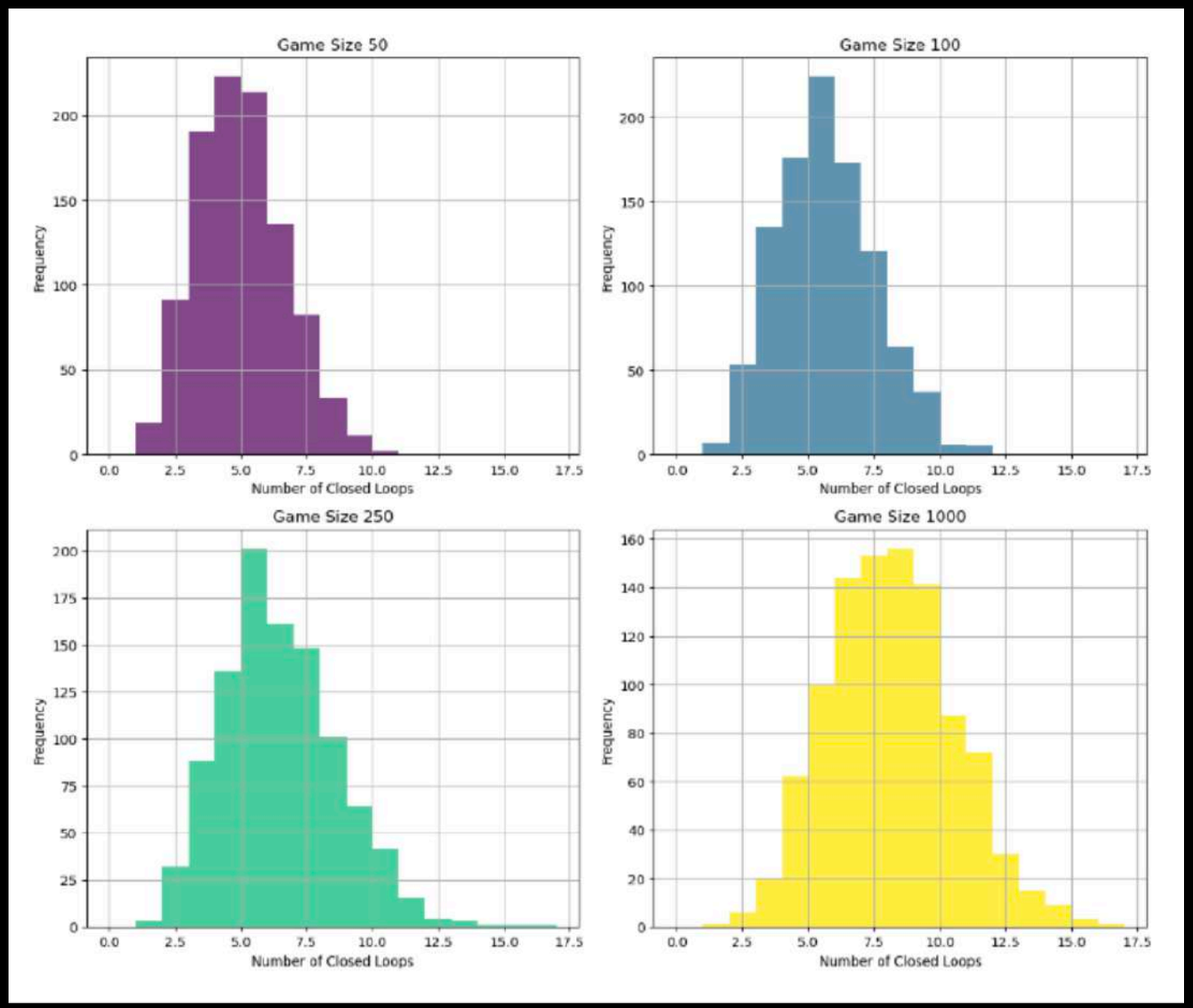
You put on a blindfold and reach into the bowl of strings. You randomly choose two ends and tie them together. You repeat this process of tying two ends a number of times.

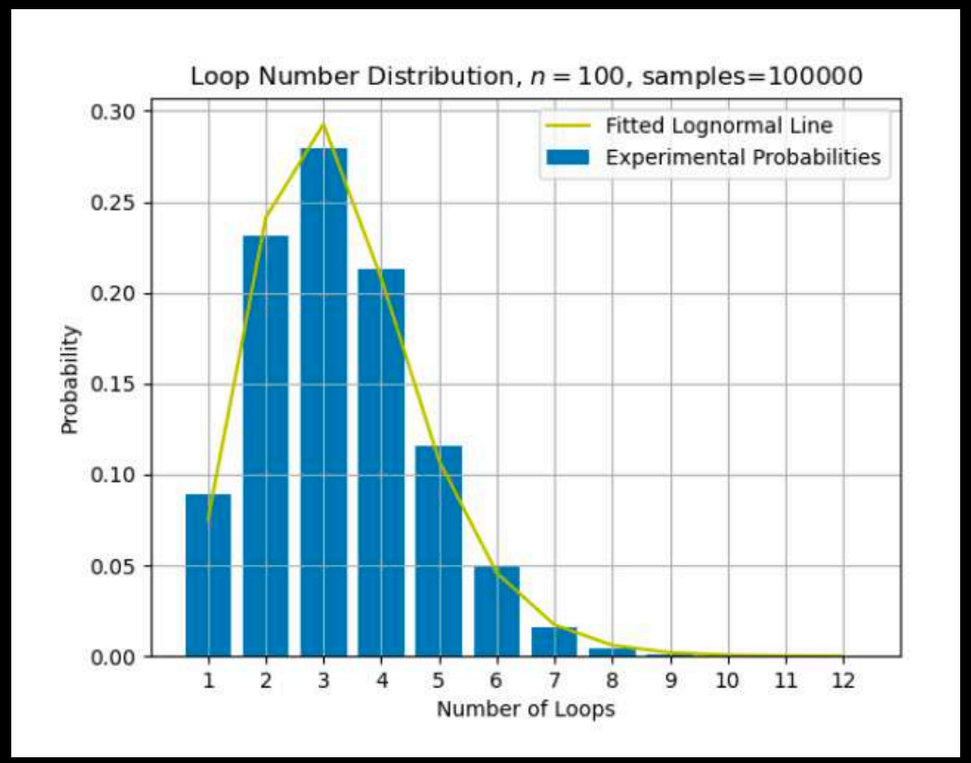
You then remove your blindfold and find that some closed string loops have been formed.

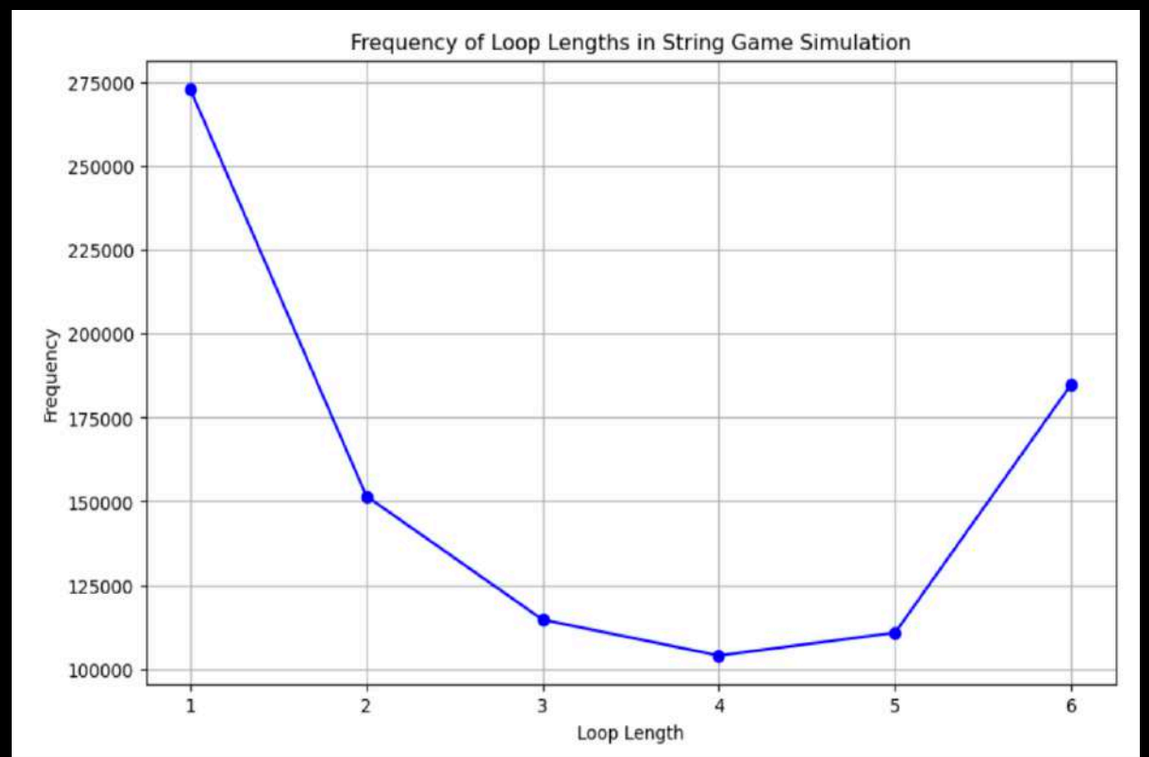
Investigate.



[Bing copilot]







Examples of students' Python-led extensions

Practical advice

1. Python enriches maths courses, deepens students' understanding and encourages mathematical curiosity

2. Start by including a small Python task in your homework (calculate/simulate/visualise)

3. If you're not confident with Python, get students to co-create tasks (e.g. summer internships)

4. Coding isn't the main point because...

ChatGPT(4o)

- Accept that AI coding is going to be the norm
- Many students use AI to create a rough code, then embellish. Why not give students basic code? Then ask them to modify.
- Beautiful coding isn't important (unless you teach computing)
- More important: interpret code output, verify theoretical results, make bold conjectures and dig deeper *("critical thinking")*

Conclusions

- Python can help students gain solid understanding and develop mathematical curiosity
- Start small (one homework problem) and go big (whole module assisted by Python)
- Students can help co-create and test content - it benefits everyone.
- Your students aren't afraid of Python and neither should you be! Lots of great resources…

Siri Chongchitnan

Exploring University Mathematics with Python

This book provides a unique tour of university mathematics with the help of Python. Written in the spirit of mathematical exploration and investigation, the book enables students to utilise Python to enrich their understanding of mathematics through:

- Calculation: performing complex calculations and numerical simulations instantly
- Visualisation: demonstrating key theorems with graphs, interactive plots and animations
- Extension: using numerical findings as inspiration for making deeper, more general conjectures.

This book is for all learners of mathematics, with the primary audience being mathematics undergraduates who are curious to see how Python can enhance their understanding of core university material. The topics chosen represent a mathematical overview of what students typically study in the first and second years at university, namely analysis, calculus, vector calculus and geometry, differential equations and dynamical systems, linear algebra, abstract algebra and number theory, probability and statistics. As such, it can also serve as a preview of university mathematics for high-school students. The prerequisites for reading the book are a familiarity with standard A-Level mathematics (or equivalent senior high-school curricula) and a willingness to learn programming.

For mathematics lecturers and teachers, this book is a useful resource on how Python can be seamlessly incorporated into the mathematics syllabus, assuming only basic knowledge of programming.

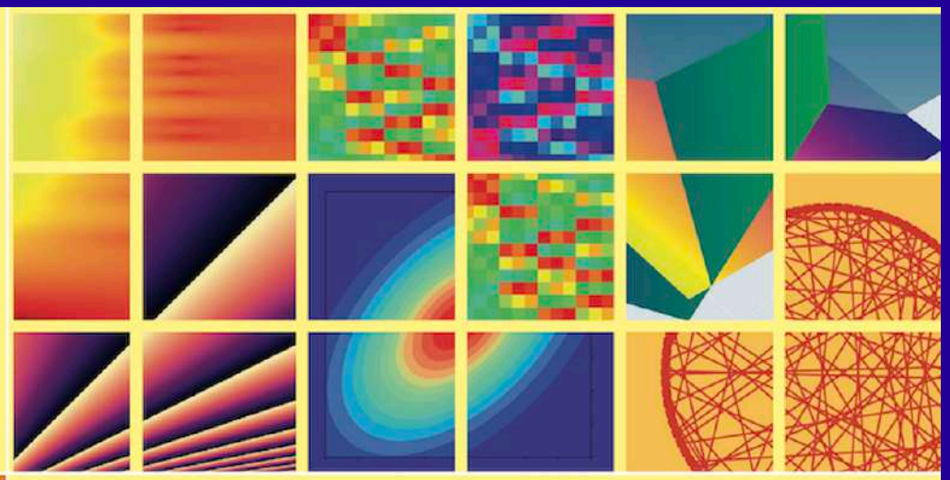
ISBN 978-3-031-46269-6



▸ springer.com

Chongchitnan

Exploring University Mathematics with Python



Siri Chongchitnan

Exploring University Mathematics with Python

Springer