Applied Numerical Methods for Civil Engineering

CGN 3405 - 0002

Review Class for Exam 1

Xinyu Chen

Assistant Professor

University of Central Florida

40% in Exam 1

Components (≈ 10 points for each, 40 points in total):

- Quadratic equations
- Euler's method
- Deflection of cantilever beam
- Taylor series

Format:

- Understand quadratic formula
- Fill in a table of results over different steps (step size $\Delta x = 0.5, 1$) using Euler's methods
- Write down the Taylor series approximation with different orders (one of $f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$, $f(x) = \sin(x)$, and $f(x) = e^x$ will be selected for exam)

Questions:

- Easy to calculate by hand

Roots of Equations

Solve quadratic equations with **a leading coefficient of 1**: $x^2 + bx + c = 0$

- Find two numbers who sum equals b and whose product equals c, i.e.,

$$d + e = b \qquad d \cdot e = c$$

by using the zero-product property:

$$(x+d)(x+e) = x^2 + \underbrace{(d+e)}_{=b}x + \underbrace{d \cdot e}_{=c} = 0$$

Example. Roots of a simple quadratic equation.

Let's solve the quadratic equation $x^2 + x - 6 = 0$.

This is a quadratic equation with

$$d + e = b = 1 \qquad d \cdot e = c = -6$$

So we can factor the equation as:

$$(x-2)(x+3) = 0$$

As a result, we can find the solutions as $x = 2$ and $x = -3$.

Example. Roots of a simple quadratic equation.

Let's solve the quadratic equation $x^2 - 5x + 6 = 0$.

This is standard quadratic equation of the form:

$$ax^2 + bx + c = 0$$

with

$$a = 1, \quad b = -5, \quad c = 6$$

So we can factor the equation as:

$$(x - 2)(x - 3) = 0$$

As a result, we can find the solutions as $x = 2$ and $x = 3$.

Roots of Equations

Quadratic formula. Given $ax^2 + bx + c = 0$ $(a \neq 0)$, we can derive the quadratic formula by completing the square.

1. Move the constant term to the right-hand side:
$$ax^2 + bx = -c$$
2. Divide by a and let the leading coefficient be 1:
$$x^2 + \frac{b}{a}x = -\frac{c}{a}$$
3. Add $\frac{b^2}{4a^2}$ to both sides:
$$x^2 + \frac{b}{a}x + \frac{b^2}{4a^2} = \frac{b^2}{4a^2} - \frac{c}{a}$$
4. Use the square root property:
$$\left(x + \frac{b}{2a}\right)^2 = \frac{b^2 - 4ac}{4a^2} \quad \Rightarrow \quad x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Quadratic formula. Given $ax^2 + bx + c = 0$ $(a \neq 0)$, the quadratic formula is

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Python programming.

```
import numpy as np

def quad_formula(a, b, c):
    term = np.sqrt(b**2 - 4*a*c)
    x1 = (-b + term) / (2*a)
    x2 = (-b - term) / (2*a)
    return x1, x2
```

Euler's Method

- **Example**. Given an ODE:

$$\frac{\mathrm{d}y}{\mathrm{d}x} = f(x, y)$$

  with initial condition $y(x_0) = y_0$

- **Euler's formula**:

$$\underbrace{y_{i+1}}_{\text{next value}} = \underbrace{y_i}_{\text{current value}} + \underbrace{\Delta x}_{\text{step size}} \cdot \underbrace{f(x_i, y_i)}_{\text{slope}}$$

$$x_{i+1} = x_i + \underbrace{\Delta x}_{\text{step size}}$$

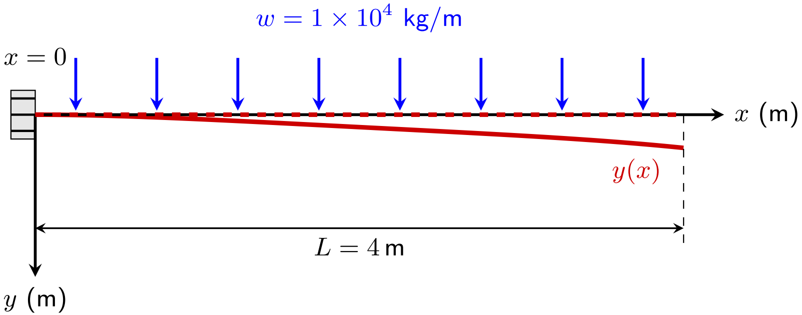
- **Interpretation**:
  - $f(x_i, y_i) =$ slope at current point
  - $\Delta x =$ step size (small values!)
  - step size $\times$ slope $=$ predicted change in y
  - Add to current y to get next y

Euler's Method

- **Toy example**: Solve
$$\frac{\mathrm{d}y}{\mathrm{d}x} = x + y$$
with $y(0) = 1$, find $y(1)$ using step size $\Delta x = 0.5$.
- ❶ Initialize $x_0 = 0$ and $y_0 = 1$
- ❷ First step $(0 \to \Delta x)$
$$f(x_0, y_0) = x_0 + y_0 = 1 \quad y_1 = y_0 + \Delta x \cdot f(x_0, y_0) = 1.5 \quad x_1 = x_0 + \Delta x = 0.5$$
- ❸ Second step $(\Delta x \to 2\Delta x)$
$$f(x_1, y_1) = x_1 + y_1 = 2 \quad y_2 = y_1 + \Delta x \cdot f(x_1, y_1) = 2.5 \quad x_2 = x_1 + \Delta x = 1$$
So we have $y(1) \approx y(x_2) = 2.5$.
- Hint (Keep in mind!):
$$\underbrace{y_{i+1}}_{\text{next value}} = \underbrace{y_i}_{\text{current value}} + \underbrace{\Delta x}_{\text{step size}} \cdot \underbrace{f(x_i, y_i)}_{\text{sloop}} \qquad x_{i+1} = x_i + \Delta x$$

Cantilever Beam Deflection

- Use **Euler's method** to find deflection $y(x)$ from $x = 0$ to $x = L$.
- $y(x)$ is downward deflection at point x (x is distance from fixed end).
- **Given parameters**:
  - Uniform load: $w = 1 \times 10^4$ kg/m
  - Beam length: $L = 4$ m
  - Modulus: $E = 2 \times 10^{11}$ Pa (steel)
  - Moment of inertia: $I = 3.25 \times 10^{-4}$ m^4



Cantilever Beam Deflection

- Use **Euler's method** to find deflection $y(x)$ from $x = 0$ to $x = L$.

$$\boxed{\frac{\mathrm{d}y}{\mathrm{d}x} = \underbrace{\frac{w}{24 \cdot E \cdot I}}_{\text{constant}}(4x^3 - 12Lx^2 + 12L^2x)}$$

- x is distance from fixed end.
- $y(x)$ is downward deflection at point x.
- **Given parameters**:
  - Uniform load: $w = 1 \times 10^4$ kg/m
  - Beam length: $L = 4$ m
  - Modulus: $E = 2 \times 10^{11}$ Pa (steel)
  - Moment of inertia: $I = 3.25 \times 10^{-4}$ m^4
- Compute the **constant factor**:

$$c = \frac{w}{24 \cdot E \cdot I} = \frac{10^4}{24 \times (2 \times 10^{11}) \times (3.25 \times 10^{-4})} = 6.41 \times 10^{-6}$$

Cantilever Beam Deflection

- **Idea**: Given the step size $\Delta x = 0.25\,\text{m}$, we start from $y(0) = 0$ and **update the deflection** by

$$\underbrace{y_{i+1}}_{\text{next deflection}} = \underbrace{y_i}_{\text{current deflection}} + \underbrace{\Delta x}_{\text{step size}} \cdot \underbrace{\frac{\mathrm{d}y}{\mathrm{d}x}}_{\text{sloop}}$$

  update the position by

$$\underbrace{x_{i+1}}_{\text{next position}} = \underbrace{x_i}_{\text{current position}} + \underbrace{\Delta x}_{\text{step size}}$$

  where the sloop is given by

$$\frac{\mathrm{d}y}{\mathrm{d}x} = c(4x^3 - 12Lx^2 + 12L^2x)$$

- Number of steps (repeat `for` loop)

$$\frac{L}{\Delta x} = \frac{4}{0.25} = 16\,\text{steps}$$

- Compute the **constant factor**:

$$c = \frac{w}{24 \cdot E \cdot I} = \frac{10^4}{24 \times (2 \times 10^{11}) \times (3.25 \times 10^{-4})} = 6.41 \times 10^{-6}$$

Cantilever Beam Deflection

Deflection table (numerical vs. analytical solution).

$$\text{error} = |y_{\text{analytical}} - y_{\text{numerical}}| \qquad \text{with } |\cdot| \text{ denoting absolute value}$$

Distance x	Analytical solution	Numerical solution	Error
0.25	0.00003688	0	
0.50	0.00014143	0.00007222	0.07 mm
0.75	0.00030491	0.00020763	$\cdots$
1.00	0.00051923	0.00039784	
1.25	0.00077687	0.00063502	
1.50	0.00107091	0.00091196	
1.75	0.00139506	0.00122206	
2.00	0.00174359	0.00155929	
2.25	0.00211140	0.00191827	
2.50	0.00249399	0.00229417	
2.75	0.00288744	0.00268279	
3.00	0.00328846	0.00308053	
3.25	0.00369434	0.00348438	
3.50	0.00410296	0.00389193	0.21 mm
3.75	0.00451285	0.00430138	0.21 mm
4.00	0.00492308	0.00471154	0.21 mm

Note: 1 meter $= 1,000$ millimeter (mm).

Taylor Series Approximation

Intuition: We predict $f(x_{i+1})$ using information at x_i:

- **Zeroth-order** (constant) approximation:

$$f(x_{i+1}) \approx f(x_i)$$

  Only works if f is **constant** between x_i and x_{i+1}

- **First-order** Taylor approximation:
  Add slope information:

$$f(x_{i+1}) \approx f(x_i) + f'(x_i) \underbrace{(x_{i+1} - x_i)}_{\text{step size } \Delta x}$$

  - Represents a **straight line** (linear approximation)
  - Exact if f is linear

- **Euler's formula**:

$$\underbrace{y_{i+1}}_{\text{next value}} = \underbrace{y_i}_{\text{current value}} + \underbrace{\Delta x}_{\text{step size}} \cdot \underbrace{f(x_i, y_i)}_{\text{slope}} \qquad x_{i+1} = x_i + \underbrace{\Delta x}_{\text{step size}}$$

Taylor Series Approximation

Intuition: We predict $f(x_{i+1})$ using information at x_i:

- **Second-order** Taylor approximation:

$$f(x_{i+1}) \approx f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2$$

  - Captures quadratic behavior
  - Better accuracy for smooth functions

Taylor Series Approximation

- **General Taylor polynomial**

$$
\begin{aligned}
f(x_{i+1}) \approx f(x_i) \\
&+ f'(x_i)(x_{i+1} - x_i) \\
&+ \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2 \\
&+ \cdots \\
&+ \frac{f^{(n)}(x_i)}{n!}(x_{i+1} - x_i)^n \\
= \sum_{k=0}^{n} \frac{f^{(k)}(x_i)}{k!}(x_{i+1} - x_i)^k
\end{aligned}
$$

Higher $n \to$ better approximation (if function is smooth)

Taylor Series Approximation of a Polynomial

Problem statement:

- Use zeroth- through fourth-order Taylor series expansions to approximate:

$$f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$$

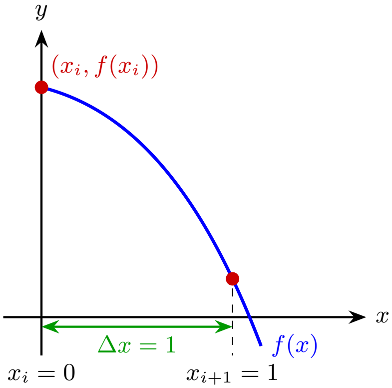
- **Goal**: Predict $f(1)$ using Taylor approximations of increasing order.

  - Function $f(x)$:

$$f(0) = 1.2$$

$$\begin{aligned} f(1) = & -0.1 - 0.15 - 0.5 \\ & -0.25 + 1.2 = 0.2 \end{aligned}$$

  - True value to predict: $f(1) = 0.2$



Taylor Series Approximation of a Polynomial

First-order approximation for $f(1)$:

- Need first-order derivative at $x_i = 0$:

$$\begin{aligned} & f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2 \\ \Rightarrow \quad & f'(x) = -0.4x^3 - 0.45x^2 - x - 0.25 \\ \Rightarrow \quad & f'(0) = -0.25 \end{aligned}$$

- First-order Taylor series:

$$\begin{aligned} & f(x_{i+1}) \approx f(x_i) + f'(x_i) \cdot (x_{i+1} - x_i) \\ \Rightarrow \quad & f(1) \approx 1.2 + (-0.25) \times 1 = 0.95 \end{aligned}$$

Taylor Series Approximation of a Polynomial

Second-order approximation for $f(1)$:

- Need second-order derivative at $x_i = 0$:

$$\begin{aligned} f(x) &= -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2 \\ \Rightarrow \quad f'(x) &= -0.4x^3 - 0.45x^2 - x - 0.25 \\ \Rightarrow \quad f''(x) &= -1.2x^2 - 0.9x - 1 \\ \Rightarrow \quad f''(0) &= -1 \end{aligned}$$

- Second-order Taylor series:

$$\begin{aligned} f(x_{i+1}) &\approx f(x_i) + f'(x_i) \cdot (x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2 \\ \Rightarrow \qquad f(1) &\approx 1.2 - 0.25 \times 1 + \left(\frac{-1}{2}\right) \times 1^2 = 0.45 \end{aligned}$$

Taylor Series Approximation of a Polynomial

Third-order approximation for $f(1)$:

- Need third-order derivative at $x_i = 0$:

$$\begin{aligned} f(x) &= -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2 \\ \Rightarrow \quad f'(x) &= -0.4x^3 - 0.45x^2 - x - 0.25 \\ \Rightarrow \quad f''(x) &= -1.2x^2 - 0.9x - 1 \\ \Rightarrow \quad f'''(x) &= -2.4x - 0.9 \\ \Rightarrow \quad f'''(0) &= -0.9 \end{aligned}$$

- Third-order Taylor series:

$$f(1) \approx 1.2 - 0.25 \times 1 - 0.5 \times 1^2 + \left(\frac{-0.9}{3!}\right) \times 1^3 = 0.3$$

Taylor Series Approximation of a Polynomial

Fourth-order approximation for $f(1)$:

- Need fourth-order derivative at $x_i = 0$:

$$
\begin{aligned}
f(x) &= -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2 \\
\Rightarrow \quad f'(x) &= -0.4x^3 - 0.45x^2 - x - 0.25 \\
\Rightarrow \quad f''(x) &= -1.2x^2 - 0.9x - 1 \\
\Rightarrow \quad f'''(x) &= -2.4x - 0.9 \\
\Rightarrow \quad f^{(4)}(x) &= -2.4 \\
\Rightarrow \quad f^{(4)}(0) &= -2.4
\end{aligned}
$$

- Fourth-order Taylor series:

$$f(1) \approx 1.2 - 0.25 \times 1 - 0.5 \times 1^2 - 0.15 \times 1^3 - \left(\frac{-2.4}{4!}\right) \times 1^4 = 0.2$$

Taylor Series Approximation of $\sin(x)$

Maclaurin series:

- A Taylor series expansion of a function about 0
- Taylor series approximation:

$$f(x_{i+1}) \approx \sum_{k=0}^{n} \frac{f^{(k)}(x_i)}{k!}(x_{i+1} - x_i)^k$$

- Set $x_i = 0$ and $x_{i+1} = x$, then

$$f(x) \approx \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k$$

- Derivatives of $f(x) = \sin(x)$:

$$f'(x) = \underbrace{\cos(x)}_{\cos(0)=1}, \quad f''(x) = -\underbrace{\sin(x)}_{\sin(0)=0}, \quad f'''(x) = -\underbrace{\cos(x)}_{\cos(0)=1}, \quad f^{(4)}(x) = \underbrace{\sin(x)}_{\sin(0)=0}$$

- Formula:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \frac{x^{13}}{13!} - \frac{x^{15}}{15!} + \cdots$$

Taylor Series Approximation of e^x

Maclaurin series:

- A Taylor series expansion of a function about 0

$$f(x) \approx \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k$$

- Derivatives of $f(x) = e^x$:

$$f'(x) = f''(x) = \cdots = f^{(n)}(x) = e^x \quad \Rightarrow \quad f^{(n)}(0) = 1 \quad \text{for all } n$$

- Formula:

$$\begin{aligned} f(x) &\approx 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} \\ &= \sum_{k=0}^{n} \frac{x^k}{k!} \end{aligned}$$

Taylor Series Approximation of e^x for $x = 1$

Problem statement:

- Given the exponential function

$$f(x) = e^x \approx 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$

- **Goal**: Predict $f(1)$ using Taylor approximations of increasing order.
- True value: $f(1) = e = 2.71828$

```
import numpy as np

print(np.exp(1))
```

Taylor Series Approximation of e^x for $x = 1$

Predict $f(1)$ (true value $f(1) = 2.71828$):

- First-order approximation:

$$\hat{f}(x) = 1 + x \quad \Rightarrow \quad \hat{f}(1) = 1 + 1 = 2$$

- Second-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} \quad \Rightarrow \quad \hat{f}(1) = 1 + 1 + \frac{1}{2} = 2.5$$

- Third-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} \quad \Rightarrow \quad \hat{f}(1) = 1 + 1 + \frac{1}{2} + \frac{1}{6} = 2.66667$$

- Fourth-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} \quad \Rightarrow \quad \hat{f}(1) = 1 + 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{24} = 2.70833$$

Taylor Series Approximation of e^x for $x = 2$

Predict $f(2)$ (true value $f(2) = 7.38906$, see `print(np.exp(2))`):

- First-order approximation:

$$\hat{f}(x) = 1 + x \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 = 3$$

- Second-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 + \frac{2^2}{2} = 5$$

- Third-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 + \frac{2^2}{2} + \frac{2^3}{6} = 6.33333$$

- Fourth-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 + \frac{2^2}{2} + \frac{2^3}{6} + \frac{2^4}{24} = 7$$

20% in Exam 1

Format (20 points in total):

- Give you Python codes
- Write down the results

Questions:

- **3 questions** are selected from the sample codes
- Parameters might be a little different
- Make sure you can totally understand these sample codes

Sample 1

```
1 a = 2
2 b = 3
3 print(a + b)  # plus
4 print(a - b)  # minus
5 print(a * b)  # product
6 print(a / b)  # division
7 print(a ** 2) # quadratic function
8 print(a ** 3) # cubic function
```

Corresponding **arithmetic operations**:

Line **3**: $a + b$

Line **4**: $a - b$

Line **5**: $a \cdot b$

Line **6**: $\frac{a}{b}$

Line **7**: a^2

Line **8**: a^3

Note: `a ** n` refers to a to the power of n.

Sample 2

Fibonacci Sequence.

- Given $f(1) = f(2) = 1$, the Fibonacci sequence takes the form of

$$f(n) = f(n-1) + f(n-2),\, n > 2$$

```python
import numpy as np

def fib(n):          # Input n>2
    f = np.zeros(n)
    f[0] = 1
    f[1] = 1
    for i in range(2, n):
        f[i] = f[i - 1] + f[i - 2]
    return f[n - 1]

print(fib(5))
print(fib(6))
print(fib(7))
```

Sample 3

A system of linear equations.

- Let's solve:

$$\begin{cases} 3x + 2y = 5 \\ x - y = 0 \end{cases} \qquad \Rightarrow \qquad \begin{bmatrix} 3 & 2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

- Try to solve by hand, and then check with Python.
- Define matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$:

$$\text{Line } \mathbf{3}\text{:} \qquad \boldsymbol{A} = \begin{bmatrix} 3 & 2 \\ 1 & -1 \end{bmatrix}$$

$$\text{Line } \mathbf{4}\text{:} \qquad \boldsymbol{b} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

```python
1 import numpy as np
2
3 A = np.array([[3, 2], [1, -1]])
4 b = np.array([5, 0])
5 solution = np.linalg.solve(A, b)
6 print('Solution (x, y):', solution)
```

Sample 4

- Generate sequences of numbers:

```
1  # Count from 0 to 4
2  for i in range(5):
3      print(i)
4
5  # With start and end
6  for i in range(2, 6):
7      print(i)
8
9  # With step
10 for i in range(0, 10, 2):
11     print(i)
```

Line **2**-**3** Result: 0, 1, 2, 3, 4

Line **6**-**7** Result: 2, 3, 4, 5

Line **10**-**11** Result: 0, 2, 4, 6, 8

Sample 5

- Repeat while condition is true:

```python
a = [1, 2, 3, 4, 5, 6, 7, 8]
i = 0
while a[i] < 6:
    print(a[i])
    i = i + 1
```

Result: 1, 2, 3, 4, 5

Sample 6

- `np.arange()`: Like Python's `range()`, but returns array

```
import numpy as np

# Bungee jumping velocity
delta_t = 0.1
t_start = 0
t_end = 20
time_step = np.arange(t_start, t_end, delta_t)
print(time_step)
```

will not count `t_end = 20`.

- Toy examples:

```
import numpy as np

a = np.arange(1, 10, 2) # step size: 2
b = np.arange(1, 10, 2.5) # step size: 2.5
```

$$\boldsymbol{a} = (1, 3, 5, 7, 9)^\top \qquad \boldsymbol{b} = (1, 3.5, 6, 8.5)^\top$$

Sample 7

- **Converting matrix into vector**

  Given a matrix $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$, there are two strategies:

```python
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6]])
a1 = np.reshape(A, (6)) # C-like index ordering
print(a1)
a2 = np.reshape(A, (6), order = 'F') # Fortran-like
    index ordering
print(a2)
```

$$\boldsymbol{a}_1 = (1, 2, 3, 4, 5, 6)^\top \qquad \boldsymbol{a}_2 = (1, 4, 2, 5, 3, 6)^\top$$

- **Converting vector into matrix**

```python
A1 = np.reshape(a1, (2, 3)) # C-like index ordering
print(A1)
A2 = np.reshape(a1, (2, 3), order = 'F') # Fortran-like
    index ordering
print(A2)
```

Sample 8

- Given a vector

```python
import numpy as np
np.random.seed(0)

a = np.random.rand(10)
print(a)
```

Result:

```
[0.5488135  0.71518937 0.60276338 0.54488318 0.4236548
 0.64589411 0.43758721 0.891773   0.96366276
 0.38344152]
```

- Indexing

```python
i = 1
j = 7
print(a[i])      # 2nd
print(a[j])      # 8th
print(a[i :])    # 2nd to the last
print(a[: j])    # 1st to 7th
print(a[i : j])  # 2nd to 7th
```

Sample 9

Flip or reverse an array:

```python
import numpy as np

a = np.arange(1, 9)
a_flip = np.flip(a)
A = np.arange([[1, 2, 3], [4, 5, 6]])
A0 = np.flip(A, axis = 0)
A1 = np.flip(A, axis = 1)
A_flip = np.flip(A)
print(a_flip)
print(A0)
print(A1)
print(A_flip)
```

Sample 10

Stack two arrays vertically and horizontally:

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
C = np.vstack(A, B)
D = np.hstack(A, B)
print(C)
print(D)
```

Sample 11

- **Factorial of a non-negative integer** n is the product of all positive integers less than or equal to n:

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 3 \times 2 \times 1$$
$$= \begin{cases} 1 & \text{if } n = 1 \\ n \times \underbrace{(n-1)!}_{\text{factorial}} & \text{if } n > 1 \end{cases}$$

```python
def factorial_r(n):
    if n == 0:
        return 1
    else:
        return n * factorial_r(n-1)
```

- Toy example: $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$

```python
print(factorial_r(5))
```

- Using NumPy

```python
import numpy as np

x = np.prod(np.arange(1, 6))
```

Sample 12

- ℓ_1-norm:

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

- ℓ_2-norm:

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

- ℓ_∞-norm:

$$\|\boldsymbol{x}\|_\infty = \max\{|x_1|, |x_2|, \ldots, |x_n|\}$$

```python
import numpy as np

a = np.arange(1, 5)
x = np.linalg.norm(a, 1)
y = np.linalg.norm(a, 2)
z = np.linalg.norm(a, np.inf)
s = np.sum(a)
print(x)
print(y)
print(z)
print(s)
```

Sample 13

- Inner product:

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x}^\top \boldsymbol{y} = \sum_{i=1}^{n} x_i y_i$$

- Outer product:

$$\boldsymbol{x}\boldsymbol{y}^\top = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix} = \begin{bmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_m y_1 & x_m y_2 & \cdots & x_m y_n \end{bmatrix}$$

```python
import numpy as np

x = np.arange(1, 4)
y = np.arange(4, 7)
print(np.inner(x, y))
print(np.outer(x, y))
```

Sample 14

- Kronecker product:

$$\boldsymbol{X} \otimes \boldsymbol{Y} = \begin{bmatrix} x_{11}\boldsymbol{Y} & x_{12}\boldsymbol{Y} & \cdots & x_{1n}\boldsymbol{Y} \\ x_{21}\boldsymbol{Y} & x_{22}\boldsymbol{Y} & \cdots & x_{2n}\boldsymbol{Y} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1}\boldsymbol{Y} & x_{m2}\boldsymbol{Y} & \cdots & x_{mn}\boldsymbol{Y} \end{bmatrix}$$

- Example:

$$\boldsymbol{X} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad \boldsymbol{Y} = \begin{bmatrix} 5 & 6 & 7 \\ 8 & 9 & 10 \end{bmatrix}$$

```python
import numpy as np

X = np.array([[1, 2], [3, 4]])
Y = np.array([[5, 6, 7], [8, 9, 10]])
print(np.kron(X, Y))
```

Sample 15

- **Example**: For vector $\boldsymbol{x} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$ and matrix $\boldsymbol{A} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$, compute the angle between $\boldsymbol{x}$ and $\boldsymbol{A}\boldsymbol{x}$.
- Matrix-vector multiplication:

$$\boldsymbol{A}\boldsymbol{x} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix}$$

- Angle:

$$\cos(\theta) = \frac{\boldsymbol{x}^\top (\boldsymbol{A}\boldsymbol{x})}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{A}\boldsymbol{x}\|_2} = \frac{1 \times 0 + 2 \times 2 + 1 \times 0}{\sqrt{1^2 + 2^2 + 1^2} \times \sqrt{0^2 + 2^2 + 0^2}} = \frac{\sqrt{6}}{3}$$

```python
import numpy as np

x = np.array([1, 2, 1])
A = np.array([[2, -1, 0], [-1, 2, -1], [0, -1, 2]])
theta = np.arccos(x @ A @ x / (np.linalg.norm(x, 2) * np.
    linalg.norm(A @ x, 2)))
```

Quiz 2

1. What is the formal $f(x) = 0$ for x? **[C]**
   A. Feasibility of equations
   B. Intersections of equations
   C. Roots of equations
   D. Optimum of equations
2. Numerical methods are techniques by which mathematical problems are formulated so that they can be solved with ________? **[D]**
   A. optimization algorithms
   B. matrix computations
   C. black-box tools
   D. arithmetic operations
3. Which of the following best describes a quadratic equation $x^2 - 1 = 0$? **[B]**
   A. A linear equation
   B. A nonlinear equation
   C. A differential equation
   D. A transcendental equation

Quiz 2

4. Is using `import numpy as np` the standard convention within the Python community? **[A]**

   A. True

   B. False

5. Based on the mathematical topics covered in this course, is the equation $f(x) = x^2 - 5x + 6 = 0$ an example of: **[A]**

   A. A root-finding problem for a nonlinear equation

   B. A system of linear equations

   C. An ordinary differential equation

   D. An optimization problem

6. An equation containing a ________ polynomial is called a quadratic equation. **[B]**

   A. first-degree

   B. second-degree

   C. third-degree

   D. fourth-degree

Quiz 2

7. Find all roots of quadratic equation $(x-2)(3x+7)=0$. **[BD]**
   A. -2
   B. 2
   C. $-3/7$
   D. $-7/3$
8. Find all roots of quadratic equation $x^2+8x+15=0$. **[BD]**
   A. 3
   B. -3
   C. 5
   D. -5
9. Find all roots of quadratic equation $x^2-4x-21=0$. **[BC]**
   A. 3
   B. -3
   C. 7
   D. -7

Quiz 2

10. Which line is used to return the roots of quadratic equations? **[D]**

```python
1 import numpy as np
2 
3 def quad_formula(a, b, c):
4     term = np.sqrt(b**2 - 4*a*c)
5     x1 = (-b + term) / (2*a)
6     x2 = (-b - term) / (2*a)
7     return x1, x2
```

A. Line 4

B. Line 5

C. Line 6

D. Line 7

Quiz 2

11. In Python programming, what is the result of `a ** 3` if we give the input `a = 4`? **[C]**

  A. 4

  B. 16

  C. 64

  D. 128

12. In Python programming, what is the result of `a ** 2` if we give the input `a = 3`? **[B]**

  A. 3

  B. 9

  C. 27

  D. 81

Quiz 3

1. How to write $\sqrt{b^2 - 4ac}$ in Python? Note that the usage of NumPy is `import numpy as np` **[C]**

   A. `np.sqrt(b ** 2 - 4ac)`

   B. `np.sqrt(b ** 2 - 4 * ac)`

   C. `np.sqrt(b ** 2 - 4 * a * c)`

   D. `np.sqrt(b ** 2 - 4a * c)`

2. A system of linear equations is a set of multiple linear equations with the different variables. **[B]**

   A. True

   B. False

3. The common methods for solving a system of linear equations by hand are Substitution and Elimination. **[A]**

   A. True

   B. False

Quiz 3

4. Optimization is the process of finding the feasible solution to a problem. **[B]**

   A. True

   B. False

5. Optimization can be simplified as follows. It is used to find the value of x that: **[AD]**

   A. minimizes f(x)

   B. convexifies f(x)

   C. constrains f(x)

   D. maximizes f(x)

6. What is the result of `5 ** n` in Python if $n = 4$? **[D]**

   A. 5

   B. 25

   C. 125

   D. 625

Quiz 3

7. What does `np.linalg.solve(A, b)` return? **[C]**

   A. The determinant of matrix A

   B. The inverse of matrix A

   C. The solution for the linear system A x = b

   D. The eigenvalues of matrix A

8. The physical interpretation of numerical integration is the determination of the slope of a curve at a point. **[B]**

   A. True

   B. False

9. In the context of ordinary differential equations (ODEs), what does the term Δt represent in the formula $y_{i+1} = y_i + f(t_i, y_i)\Delta t$? **[B]**

   A. The slope of the function

   B. The time step size

   C. The initial condition

   D. The exact solution

Quiz 3

10. Optimization in engineering can involve minimizing cost or maximizing efficiency. **[A]**

   A. True

   B. False

11. Which of the following are examples of optimization goals in engineering? (Select all that apply.) **[ABF]**

   A. Minimize material usage

   B. Maximize traffic flow efficiency

   C. Solve linear equations

   D. Fit a curve through data points

   F. Maximize structural strength

Quiz 3

12. What does the following Python code compute? **[B]**

```python
import numpy as np

def quad_formula(a, b, c):
    term = np.sqrt(b**2 - 4*a*c)
    x1 = (-b + term) / (2*a)
    x2 = (-b - term) / (2*a)
    return x1, x2
```

A. The derivative of a quadratic function

B. The roots of a quadratic equation

C. The integral of a quadratic function

D. The minimum of a quadratic function

Quiz 3

13. If `a = 2` and `b = 3` in Python, what will `print(a ** b)` output? **[C]**
    A. 5
    B. 6
    C. 8
    D. 9
14. How to write matrix $\boldsymbol{A} = \begin{bmatrix} 2 & 1 \\ 1 & -3 \end{bmatrix}$ in Python with NumPy? **[B]**
    A. `A = [[2, 1], [1, -3]]`
    B. `A = np.array([[2, 1], [1, -3]])`
    C. `A = np.array([2, 1], [1, -3])`

Quiz 4

1. The bungee jumper's velocity over time can be modeled using an ordinary differential equation. **[A]**

   A. True

   B. False

2. Which forces are acting on the bungee jumper? (Select all that apply.) **[AD]**

   A. Gravitational force

   B. Tension force from the cord

   C. Centrifugal force

   D. Air resistance

3. In the ODE $\frac{dv}{dt} = g - \frac{c_d}{m}v^2$, what does $\frac{dv}{dt}$ represent? **[B]**

   A. Position

   B. Acceleration

   C. Velocity

   D. Drag coefficient

Quiz 4

4. In the Python function for Euler's method, what does the following line compute? **[B]**

```
a = g - cd / m * (v[i] ** 2)
```

A. Position

B. Acceleration

C. Drag force

D. Terminal velocity

5. The terminal velocity occurs when acceleration is zero. **[A]**

A. True

B. False

6. If mass $m = 50\,\text{kg}$, $g = 9.81\,\text{m/s}^2$, and $c_d = 0.25\,\text{kg/m}$, what is the terminal velocity? **[C]**

A. 19.81 m/s

B. 31.32 m/s

C. 44.29 m/s

D. 50.00 m/s

Quiz 4

7. Which factors affect the accuracy of Euler's method? **[A]**
   - A. Time step size
   - B. Initial velocity
   - C. Drag coefficient
   - D. Mass of the jumper

8. A smaller Δt in Euler's method reduces computational error but increases computation time. **[A]**
   - A. True
   - B. False

9. In the Fibonacci sequence Python code, what does `f[i] = f[i-1] + f[i-2]` compute? **[B]**
   - A. The sum of the first n numbers
   - B. The next Fibonacci number
   - C. The average of previous two numbers
   - D. The product of previous two numbers

Quiz 4

10. Which of the following are advantages of numerical methods like Euler's method? (Select all that apply.) **[AC]**

   A. Can solve problems with no analytical solution

   B. Always gives exact results

   C. Easy to implement in code

11. In the error analysis plot, the error is computed as $|v_{\text{numerical}} - v_{\text{analytical}}|$ **[A]**

   A. True

   B. False

12. What does the following Python function return? **[B]**

```python
def analytical_solution(m, g, cd, t):
    v_term = np.sqrt(m * g / cd)
    return v_term * np.tanh(np.sqrt(g * cd / m) * t)
```

   A. Numerical velocity from Euler's method

   B. Analytical velocity as a function of time

   C. Terminal velocity only

   D. Acceleration over time

Quiz 4

13. Which parameters can be adjusted to keep terminal velocity within safe limits? **[B]**

   A. Increase mass
   B. Increase drag coefficient
   C. Decrease gravitational acceleration
   D. Use a longer cord

14. In the parameter sensitivity study, if mass increases and drag coefficient stays the same, terminal velocity: **[A]**

   A. Increases
   B. Decreases
   C. Stays the same
   D. Becomes zero

15. The Fibonacci sequence example was used to introduce recursive functions in Python. **[A]**

   A. True
   B. False

Quiz 5

1. Euler's Method is primarily used to solve which type of mathematical problems? **[C]**
   - A. Algebraic equations
   - B. Partial differential equations
   - C. Ordinary differential equations (ODEs)
   - D. Linear systems only
2. When is Euler's Method most appropriate to use? **[B]**
   - A. When an exact analytical solution is required
   - B. When the rate of change is known and an approximate solution is acceptable
   - C. When the function has no derivative
   - D. When step size is large

Quiz 5

3. In Euler's Method, what does $f(x_i, y_i)$ represent? **[C]**
   A. The next value of y
   B. The step size
   C. The slope at the current point
   D. The numerical error
4. Why should the step size Δx be small when using Euler's Method? **[B]**
   A. To reduce computational cost
   B. To reduce numerical error
   C. To increase instability
   D. To simplify equations
5. What is the effect of increasing the step size in Euler's Method? **[C]**
   A. Higher accuracy
   B. Lower computational speed
   C. Increased numerical error
   D. No effect on results

Quiz 5

6. For the ODE $\frac{dy}{dx} = x + y$ with $y(0) = 1$ and $\Delta x = 0.5$, what is $y(0.5)$ using Euler's method? **[C]**
   A. 1.0
   B. 1.25
   C. 1.5
   D. 2.0
7. Why is calculating cantilever beam deflection important in engineering? **[B]**
   A. To determine beam color
   B. To check safety and meet building codes
   C. To calculate air resistance
   D. To determine beam weight
8. In the cantilever beam problem, what does $y(x)$ represent? **[C]**
   A. Beam stress
   B. Beam length
   C. Downward deflection at distance x
   D. Load intensity

Quiz 5

9. The constant $c = \frac{w}{24 \cdot E \cdot I}$ depends on which parameters? **[B]**
   A. Load, beam length, and position
   B. Load, modulus of elasticity, and moment of inertia
   C. Velocity and acceleration
   D. Step size and position

10. How many Euler steps are required if beam length $L = 4\,\mathrm{m}$ and step size $\Delta x = 0.25\,\mathrm{m}$? **[C]**
   A. 8
   B. 12
   C. 16
   D. 20

11. Why does numerical error increase as distance x increases in the cantilever beam example? **[B]**
   A. Because analytical solutions are unstable
   B. Because Euler's Method accumulates error at each step
   C. Because the load decreases
   D. Because step size changes

40% in Exam 1

Format (40 points in total):

- 20 questions selected from 150 quiz questions
- Including both numerical computing and Python programming

Quiz 6

1. What is the primary programming environment recommended in this course for beginners? **[B]**

   A. PyCharm

   B. Google Colab (Colaboratory)

   C. Visual Studio Code

   D. Jupyter Notebook (local install)

Quiz 6

2. Which of the following are basic data types in Python? (Select all that apply.) **[ABCD]**

   A. Integer

   B. Float

   C. String

   D. Boolean

   E. List

Quiz 6

3. What does the `type()` function do? **[C]**
   A. Prints a value to the screen
   B. Converts a value to a different type
   C. Returns the data type of a value
   D. Defines a new variable

Quiz 6

4. What is the output of the following code? **[C]**

```python
a = 2
b = 3
print(a ** b)
```

A. 5
B. 6
C. 8
D. 9

Quiz 6

5. What will the following code output? **[A]**

```
deflections = [12.3, 15.7, 18.2, 14.9, 16.5]
print(deflections[0])
```

A. 12.3

B. 15.7

C. 16.5

D. Error

Quiz 6

6. Which function gives the number of items in a list? **[B]**

   A. `count()`

   B. `len()`

   C. `size()`

   D. `sum()`

Quiz 6

7. What is the output of this code? **[B]**

```python
stress = 235
if stress > 250:
    print('WARNING')
elif stress > 200:
    print('Alert')
else:
    print('Safe')
```

A. WARNING
B. Alert
C. Safe
D. No output

Quiz 6

8. What does the for loop do? **[B]**
   A. Repeats while a condition is true
   B. Iterates over each item in a sequence
   C. Executes a block of code once
   D. Defines a function

Quiz 6

9. The `while` loop continues until a condition becomes false. **[A]**
   A. True
   B. False

Quiz 6

10. What will this code print? **[B]**

```
a = [1, 2, 3, 4, 5]
for x in a:
    if x > 3:
        print(x)
```

A. 1, 2, 3

B. 4, 5

C. 5

D. 1, 2, 3, 4, 5

Quiz 6

11. What does `print('Hello Civil Engineering!')` do? **[B]**
    A. Defines a string variable
    B. Displays text to the screen
    C. Creates a file
    D. Nothing, it's invalid syntax

Quiz 6

12. How do you access the last element of a list named `data`?

    A. `data[last]`

    B. `data[-1]`

    C. `data[len(data)]`

    D. `data.end()`

Quiz 6

13. What will this code output? **[A]**

```
numbers = [1, 2, 3, 4, 5]
total = 0
for num in numbers:
    total += num
print(total)
```

A. 15
B. 10
C. 5
D. Error

Quiz 7

1. How should NumPy be imported in a Python script according to the slides? **[C]**

   A. `import numpy`

   B. `import np as numpy`

   C. `import numpy as np`

   D. `from numpy import *`

Quiz 7

2. Which of the following is NOT a built-in array creation function in NumPy? **[D]**

   A. `np.ones()`

   B. `np.zeros()`

   C. `np.eye()`

   D. `np.matrix()`

Quiz 7

3. If `np.eye(4)` is used, what type of matrix is created? **[B]**
   A. 4×4 matrix of ones
   B. 4×4 identity matrix
   C. 4×4 random matrix
   D. 4×4 matrix of zeros

Quiz 7

4. In `np.arange(start, stop, step)`, does the array include the `stop` value? **[B]**

   A. Yes

   B. No

   C. Only if step is 1

   D. Only if stop is an integer

Quiz 7

5. What does the following code produce? **[A]**

```
print(np.linspace(0, 4, 5))
```

A. `[0, 1, 2, 3, 4]`

B. `[0, 0.8, 1.6, 2.4, 3.2, 4]`

C. `[0, 1, 2, 3]`

D. `[0, 4]`

Quiz 7

6. What is the result of element-wise multiplication of two NumPy arrays `a` and `b` of the same size? **[C]**

   A. Matrix multiplication

   B. Dot product

   C. Element-by-element multiplication

   D. Cross product

Quiz 7

7. Which operator is used for matrix multiplication in NumPy? **[C]**
   A. `*`
   B. `**`
   C. `@`
   D. `&`

Quiz 7

8. What does `np.random.seed(0)` ensure? **[B]**
   A. Faster random number generation
   B. Reproducibility of random numbers
   C. Only integers are generated
   D. No random numbers are generated

Quiz 7

9. If `A` is a 2$\times$3 matrix created with `np.random.rand(2, 3)`, what values does it contain? **[B]**

   A. Integers from 0 to 1

   B. Random floats between 0 and 1

   C. Random integers between 0 and 10

   D. Random floats between -1 and 1

Quiz 7

10. If `a1 = np.array([1, 2, 3, 4, 5, 6])` is reshaped to `(2, 3)` with `order='C'`, what is the resulting matrix? **[A]**

   A. `[[1, 2, 3], [4, 5, 6]]`

   B. `[[1, 3, 5], [2, 4, 6]]`

   C. `[[1, 2], [3, 4], [5, 6]]`

   D. `[[1, 4], [2, 5], [3, 6]]`

Quiz 7

11. Given `A = np.array([[1, 2, 3], [4, 5, 6]])`, what does `A[1, 2]` return? **[D]**

    A. 2

    B. 3

    C. 5

    D. 6

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & \color{red}{6} \end{bmatrix}$$

Quiz 7

12. How do you slice a 2D array to get rows 2 to 3 and columns 3 to 4? **[B]**
    A. `A[2:4, 3:5]`
    B. `A[1:3, 2:4]`
    C. `A[2:3, 3:4]`
    D. `A[2-4, 3-5]`

13. What is the output of `np.arange(2, 6)`? **[A]**
    - A. `[2, 3, 4, 5]`
    - B. `[2, 3, 4, 5, 6]`
    - C. `[2, 6]`
    - D. `[3, 4, 5]`

14. Which NumPy function is used to generate sequences of numbers similar to Python's `range()` but returns an array? **[C]**

A. `np.linspace()`

B. `np.sequence()`

C. `np.arange()`

D. `np.range()`

Quiz 8

1. What does the function `np.where(a > 0.5)` return? **[A]**
   A. The indices where the condition is True
   B. The actual values where the condition is True
   C. A Boolean array
   D. The number of True conditions

Quiz 8

2. Given `A = np.array([[1, 2, 3], [4, 5, 6]])`, what is the result of `print(A.T)`? **[A]**

   A. `[[1, 4], [2, 5], [3, 6]]`

   B. `[[1, 2, 3], [4, 5, 6]]`

   C. `[[3, 2, 1], [6, 5, 4]]`

   D. `[[6, 5, 4], [3, 2, 1]]`

Quiz 8

3. What does `print(np.flip(a))` do to a 1D array `a = np.array([1, 2, 3, 4, 5])`? **[A]**

   A. `[5, 4, 3, 2, 1]`

   B. `[1, 2, 3, 4, 5]`

   C. `[2, 1, 3, 4, 5]`

   D. `[1, 5, 2, 4, 3]`

Quiz 8

4. To flip only the rows of a matrix `A`, which parameter should be used with `np.flip()`? **[A]**

   A. `axis=0`

   B. `axis=1`

   C. `axis=-1`

   D. `axis=2`

Quiz 8

5. What does `np.vstack((A, B))` require about arrays `A` and `B`? **[B]**
   A. Same number of rows
   B. Same number of columns
   C. Same shape
   D. Same data type

Quiz 8

6. Which function loads a NumPy array from a CSV file? **[B]**
   A. `np.readcsv()`
   B. `np.loadtxt()`
   C. `np.fromcsv()`
   D. `np.importcsv()`

Quiz 8

7. Given `A[:, 4]`, what does this indexing return? **[C]**
   A. The 4th row
   B. The 4th column
   C. The 5th column
   D. The 5th row

Quiz 8

8. If `A` is a 3x4 matrix, what does `A[1:3, 2:4]` select? **[B]**
   A. Rows 1–2 and columns 2–3
   B. Rows 2–3 and columns 3–4
   C. Rows 1–2 and columns 3–4
   D. Rows 2–3 and columns 2–3

Quiz 8

9. Given `A = np.random.rand(3, 2)`, what is `A.shape`? **[B]**
   A. `(2, 3)`
   B. `(3, 2)`
   C. `(6,)`
   D. `(3, 3)`

Quiz 9

1. Why are functions used in programming? **[B]**
   A. To make code longer
   B. For reusability, modularity, abstraction, and easier testing
   C. To avoid using variables
   D. Only for mathematical operations

Quiz 9

2. What is the basic syntax of a Python function definition? **[B]**
   A. `function_name():`
   B. `def function_name(parameters):`
   C. `define function_name(parameters):`
   D. `func function_name(parameters):`

Quiz 9

3. In the normal stress function `def normal_stress(F, A):`, what does it return? **[B]**

   A. `F * A`

   B. `F / A`

   C. `F + A`

   D. `F - A`

Quiz 9

4. What is a lambda function in Python? **[B]**
   A. A function that can only be used once
   B. A quick, one-line anonymous function
   C. A function that requires a docstring
   D. A function that calls itself

Quiz 9

5. How would you define a lambda function to calculate the square of a number? **[B]**

   A. `square = lambda x:  x*2`

   B. `square = lambda x:  x**2`

   C. `square = def(x):  x**2`

   D. `square = lambda:  x**2`

6. If `g = lambda r:  np.pi * r**2 / 4`, what is `g(2)`? **[A]**
   A. π
   B. 2π
   C. 4π
   D. 8π

$$g(r) = \frac{\pi r^2}{4} \qquad \Rightarrow \qquad g(2) = \pi$$

Quiz 9

7. What is a recursive function? **[B]**
   A. A function that runs in a loop
   B. A function that calls itself
   C. A function that uses lambda
   D. A function that returns multiple values

Quiz 9

8. What is the Taylor series expansion for sin(x)? **[B]**
   A. Sum of even-powered terms
   B. Sum of odd-powered terms with alternating signs
   C. Sum of all integer powers
   D. A polynomial with only positive coefficients

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

Quiz 9

9. In the Taylor series for sin(x), what is the denominator of each term? **[B]**
   A. Factorial of an even number
   B. Factorial of an odd number
   C. The term index
   D. The value of x

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

Quiz 9

10. In the function `sin_taylor(x, num_term)`, what does `num_term` represent? **[A]**

A. The number of terms to use in the approximation

B. The value of x

C. The factorial limit

D. The maximum error allowed

```python
import numpy as np

def sin_taylor(x, num_term):
    result = 0
    for n in range(num_term):
        # Term index: 0, 1, 2, ... corresponds to x^1, x^3,
            x^5, ...
        exp = 2*n + 1
        factorial = np.prod(np.arange(1, exp + 1))
        result += ((-1) ** n) * (x ** exp) / factorial
    return result
```

Quiz 10

1. What does the ℓ_1-norm measure mathematically? **[B]**
   A. The sum of squared components
   B. The sum of absolute values of components
   C. The maximum absolute component
   D. The Euclidean length

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

Quiz 10

2. Which norm is also known as the "Manhattan norm"? **[C]**

   A. ℓ_2-norm

   B. ℓ_∞-norm

   C. ℓ_1-norm

   D. Frobenius norm

Quiz 10

3. In engineering, the ℓ_1-norm is used to compute: **[B]**
   A. Peak stress
   B. Total resource consumption or total absolute error
   C. Resultant force magnitude
   D. Matrix eigenvalues

Quiz 10

4. What is the ℓ_2-norm of the vector [1, 2, 3, 4]? **[B]**
   A. 10
   B. $\sqrt{30}$
   C. 30
   D. 4

$$\sqrt{1^2 + 2^2 + 3^2 + 4^2} = \sqrt{30}$$

Quiz 10

5. What does the ℓ_∞-norm measure? **[C]**
   A. Sum of absolute values
   B. Square root of sum of squares
   C. Maximum absolute value in the vector
   D. Average of absolute values

Quiz 10

6. In civil engineering, the ℓ_∞-norm can be used to find: **[B]**
   A. Total material used
   B. Maximum stress or worst-case error
   C. Average deflection
   D. Resultant force

Quiz 10

7. Which Python function is used to compute the ℓ_1-norm of a vector `a` in NumPy? **[A]**

   A. `np.linalg.norm(a, 1)`

   B. `np.linalg.norm(a, 2)`

   C. `np.linalg.norm(a, np.inf)`

   D. `np.sum(np.square(a))`

Quiz 10

8. If `errors = np.array([-1.2, 0.8, -2.1, 1.5])`, what is the ℓ_∞-norm?
   A. 1.5
   B. 2.1
   C. 5.6
   D. 0.8

$$\max\{|-1.2|, |0.8|, |-2.1|, |1.5|\} = \max\{1.2, 0.8, 2.1, 1.5\} = 2.1$$

Quiz 10

9. The Frobenius norm is used for:
   A. Vectors only
   B. Matrices
   C. Scalars
   D. Complex numbers only

Quiz 10

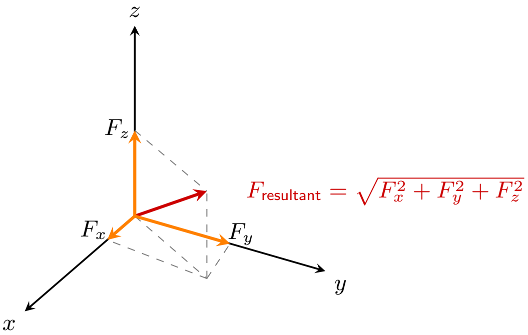
10. In the context of structural engineering, the ℓ_2-norm can represent: **[B]**

   A. Total weight of materials

   B. Magnitude of a force vector

   C. Maximum stress

   D. Average strain



Quiz 10

11. What does `np.linalg.norm(a, np.inf)` compute? **[C]**

    A. ℓ_1-norm

    B. ℓ_2-norm

    C. ℓ_∞-norm

    D. Frobenius norm

Quiz 10

12. The Mean Absolute Error (MAE) is proportional to: **[C]**

   A. ℓ_2-norm of error

   B. ℓ_∞-norm of error

   C. ℓ_1-norm of error

   D. Frobenius norm of error

Quiz 10

13. Which norm would be most appropriate for a "worst-case" safety analysis? **[C]**

   A. ℓ_1-norm

   B. ℓ_2-norm

   C. ℓ_∞-norm

   D. Frobenius norm

14. The ℓ_2-norm of a vector `x` in Python can be computed without `np.linalg.norm` using: **[C]**

A. `np.sum(np.abs(x))`

B. `np.max(np.abs(x))`

C. `np.sqrt(np.sum(x**2))`

D. `np.prod(np.abs(x))`

Quiz 11

1. Which of the following correctly computes the inner product of vectors x and y in NumPy? **[D]**

   A. `np.sum(x * y)`

   B. `np.inner(x, y)`

   C. `x @ y`

   D. `All of the above`

Quiz 11

2. Given `x = np.arange(1, 5)` and `y = np.arange(6, 10)`, what is the value of `np.inner(x, y)`? **[C]**

   A. 30

   B. 50

   C. 80

   D. 100

$$\boldsymbol{x} = (1, 2, 3, 4)^\top \qquad \boldsymbol{y} = (6, 7, 8, 9)^\top$$

$$\Rightarrow \quad \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 1 \times 6 + 2 \times 7 + 3 \times 8 + 4 \times 9 = 6 + 14 + 24 + 36 = 80$$

Quiz 11

3. The inner product of a vector with itself is equal to: **[B]**
   A. The ℓ_1-norm squared
   B. The ℓ_2-norm squared
   C. The ℓ_∞-norm squared
   D. The Frobenius norm squared

$$\langle \boldsymbol{x}, \boldsymbol{x} \rangle = \|\boldsymbol{x}\|_2^2$$

Quiz 11

4. Which code correctly computes the outer product of vectors `x` and `y`?
   A. `np.outer(x, y)`
   B. `np.kron(x, y)`
   C. `np.dot(x, y.T)`
   D. `x @ y.T`

Quiz 11

5. If `x = np.arange(1, 4)` and `y = np.arange(4, 6)`, what is the shape of `np.outer(x, y)`? **[A]**

   A. (3, 2)

   B. (2, 3)

   C. (3, 3)

   D. (2, 2)

$$\boldsymbol{x} = (1, 2, 3)^\top \qquad \boldsymbol{y} = (4, 5)^\top$$

Quiz 11

6. The Kronecker product of two matrices `X` and `Y` in NumPy is computed using: **[B]**

   A. `np.outer(X, Y)`

   B. `np.kron(X, Y)`

   C. `np.dot(X, Y)`

   D. `X @ Y`

Quiz 11

7. Given `X = np.array([[1, 2], [3, 4]])` and `Y = np.array([[5, 6, 7], [8, 9, 10]])`, what is the shape of `np.kron(X, Y)`? **[C]**

   A. (2, 3)

   B. (3, 6)

   C. (4, 6)

   D. (6, 4)

8. A matrix `A` is positive definite if for any nonzero vector `x`: **[C]**
   A. `x.T @ A @ x < 0`
   B. `x.T @ A @ x = 0`
   C. `x.T @ A @ x > 0`
   D. `A @ x = 0`

$$\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} > 0$$

Quiz 11

9. For `A = np.array([[2, -1, 0], [-1, 2, -1], [0, -1, 2]])`, the expression `x.T @ A @ x` simplifies to: **[B]**

   A. $x_1^2 + x_2^2 + x_3^2$

   B. $x_1^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + x_3^2$

   C. $2x_1^2 + 2x_2^2 + 2x_3^2$

   D. $x_1x_2 + x_2x_3$

   - matrix-vector multiplication:

$$\boldsymbol{A}\boldsymbol{x} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2x_1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 \end{bmatrix}$$

   - inner product:

$$\begin{aligned} \boldsymbol{x}^\top(\boldsymbol{A}\boldsymbol{x}) &= x_1(2x_1 - x_2) + x_2(-x_1 + 2x_2 - x_3) + x_3(-x_2 + 2x_3) \\ &= 2x_1^2 - 2x_1x_2 + 2x_2^2 - 2x_2x_3 + 2x_3^2 \\ &= x_1^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + x_3^2 > 0 \end{aligned}$$

Quiz 11

10. The cosine of the angle between two vectors `a` and `b` is given by: **[A]**

   A. `np.inner(a, b) / (np.linalg.norm(a, 2) * np.linalg.norm(b, 2))`

   B. `np.dot(a, b) / np.sum(a * b)`

   C. `np.kron(a, b) / (np.linalg.norm(a, 1) * np.linalg.norm(b, 1))`

   D. `np.outer(a, b) / np.abs(a) @ np.abs(b)`

$$\cos(\theta) = \frac{\boldsymbol{a}^\top \boldsymbol{b}}{\|\boldsymbol{a}\|_2 \cdot \|\boldsymbol{b}\|_2}$$

Quiz 11

11. For `x = np.array([1, 2, 1])` and `A = np.array([[2, -1, 0], [-1, 2, -1], [0, -1, 2]])`, what is `print(A @ x)`? **[A]**

   A. `[0, 2, 0]`

   B. `[1, 2, 1]`

   C. `[2, 2, 2]`

   D. `[0, 0, 0]`

$$\boldsymbol{A}\boldsymbol{x} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix}$$

Quiz 11

12. Which of the following is the correct way to import Matplotlib in Python? **[A]**

A. `import matplotlib.pyplot as plt`

B. `import matplotlib as mpl`

C. `from matplotlib import pyplot`

D. `All are valid`

Quiz 11

13. In Matplotlib, which function is used to save a figure as a PDF? **[A]**
    A. `plt.savefig('file.pdf')`
    B. `plt.export('file.pdf')`
    C. `plt.write('file.pdf')`
    D. `plt.download('file.pdf')`

Quiz 12

1. Which of the following is an example of truncation error? **[A]**
   A. Approximating a derivative using a finite difference formula
   B. Representing π as 3.14
   C. Storing a large number in floating-point format
   D. All of the above

Quiz 12

2. According to the Taylor theorem, a smooth function can be approximated by: **[B]**

A. A trigonometric series

B. A polynomial

C. An exponential function

D. A logarithmic function

Quiz 12

3. The zeroth-order Taylor approximation of a function predicts: **[C]**
   A. The function's slope
   B. The function's curvature
   C. The function's constant value
   D. The function's integral

Quiz 12

4. The first-order Taylor approximation adds which term to improve the estimate? **[B]**

   A. The second derivative term

   B. The slope term

   C. The constant term

   D. The remainder term

Quiz 12

5. In Euler's formula for numerical integration, what does the term $\Delta x \cdot f(x_i, y_i)$ represent? **[B]**

   A. Current value

   B. Slope multiplied by step size

   C. Truncation error

   D. Exact solution

Quiz 12

6. In the Taylor series example, the function $f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$ was approximated at $x = 1$ starting from $x_i = 0$. What was the step size? **[C]**

   A. 0

   B. 0.5

   C. 1

   D. 2

Quiz 12

7. Why did the fourth-order Taylor approximation give an exact value for $f(1)$? **[B]**

A. Because the function is linear

B. Because the function is a 4th-degree polynomial

C. Because the step size was 1

D. Because higher-order terms cancel out

Quiz 12

8. The Maclaurin series is a special case of the Taylor series where the expansion point is: **[B]**

A. $x = 1$

B. $x = 0$

C. $x \to \infty$

D. $x = \pi$

9. Which derivative of $\sin(x)$ evaluated at $x = 0$ equals 1? **[B]**
   - A. $f(x)$
   - B. $f'(x)$
   - C. $f''(x)$
   - D. $f'''(x)$

Derivatives of $f(x) = \sin(x)$:

$$f'(x) = \underbrace{\cos(x)}_{\cos(0)=1}, \quad f''(x) = -\underbrace{\sin(x)}_{\sin(0)=0}, \quad f'''(x) = -\underbrace{\cos(x)}_{\cos(0)=1}, \quad f^{(4)}(x) = \underbrace{\sin(x)}_{\sin(0)=0}$$

Quiz 12

10. In general, increasing the order n in a Taylor series approximation: **[B]**
    A. Always increases error
    B. Always decreases error if the function is smooth
    C. Has no effect on error
    D. Makes the approximation linear

Quiz 13

1. For $f(x) = x^2$, what is the true value of $f(2)$ and the first-order Taylor approximation at $x = 2$ starting from $x_i = 1$? **[A]**

   A. True = 4, Approximation = 3

   B. True = 4, Approximation = 2

   C. True = 3, Approximation = 4

   D. True = 4, Approximation = 5

$$f(x_{i+1}) \approx f(x_i) + \Delta x \cdot f'(x_i) = 1 + 1 \times 2 = 3$$

Quiz 13

2. What is the absolute error for the first-order Taylor approximation of $f(x) = x^3$ at $x = 2$ starting from $x_i = 1$? **[B]**

   A. 1

   B. 4

   C. 8

   D. 11

$$f(x_{i+1}) \approx f(x_i) + \Delta x \cdot f'(x_i) = 1 + 1 \times 3 = 4$$

Quiz 13

3. For $f(x) = x^4$, what is the true value of $f(2)$ and the first-order Taylor approximation starting from $x_i = 1$? **[B]**

A. True = 8, Approximation = 4

B. True = 16, Approximation = 5

C. True = 16, Approximation = 4

D. True = 11, Approximation = 16

$$f(x_{i+1}) \approx f(x_i) + \Delta x \cdot f'(x_i) = 1 + 1 \times 4 = 5$$

Quiz 13

4. As the degree m of the function $f(x) = x^m$ increases, what happens to the error of the first-order Taylor approximation at a fixed step size? **[C]**

A. Error decreases

B. Error remains the same

C. Error increases

D. Error becomes zero

Quiz 13

5. For $f(x) = x^4$, what is the approximation of $f(1.5)$ using a first-order Taylor series starting from $x_i = 1$? **[A]**

A. 3

B. 4

C. 5

D. 6

$$f(x_{i+1}) \approx f(x_i) + \Delta x \cdot f'(x_i) = 1 + 0.5 \times 4 = 3$$

Quiz 13

6. For $f(x) = x^4$, what is the approximation error of $f(1.25)$ using a first-order Taylor series starting from $x_i = 1$? **[B]**

A. 0.101807

B. 0.441406

C. 0.25

D. 0.5

$$f(x_{i+1}) \approx f(x_i) + \Delta x \cdot f'(x_i) = 1 + 0.25 \times 4 = 2$$

$$f(1.25) = 1.25^4 = 2.4414$$

Quiz 13

7. What happens to the error of a first-order Taylor approximation as the step size Δx is reduced? **[B]**

A. Error increases

B. Error decreases

C. Error remains constant

D. Error becomes unpredictable

Quiz 13

8. What is the true value of e? **[C]**
   - A. 2.5
   - B. 2.66667
   - C. 2.71828
   - D. 2.70833

Quiz 13

9. What is the third-order Taylor approximation of e^2? **[B]**

   A. 5

   B. 6.33333

   C. 7

   D. 7.38906

- First-order approximation:

$$\hat{f}(x) = 1 + x \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 = 3$$

- Second-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 + \frac{2^2}{2} = 5$$

- Third-order approximation:

$$\hat{f}(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} \quad \Rightarrow \quad \hat{f}(2) = 1 + 2 + \frac{2^2}{2} + \frac{2^3}{6} = 6.33333$$

Quiz 13

10. What is the key engineering implication from the Taylor series analysis of functions e^x and x^m? **[C]**

A. Step size does not affect accuracy

B. Nonlinearity reduces error

C. Keep step size small for accuracy; use higher-order methods for larger steps

D. Higher-order approximations are always less accurate