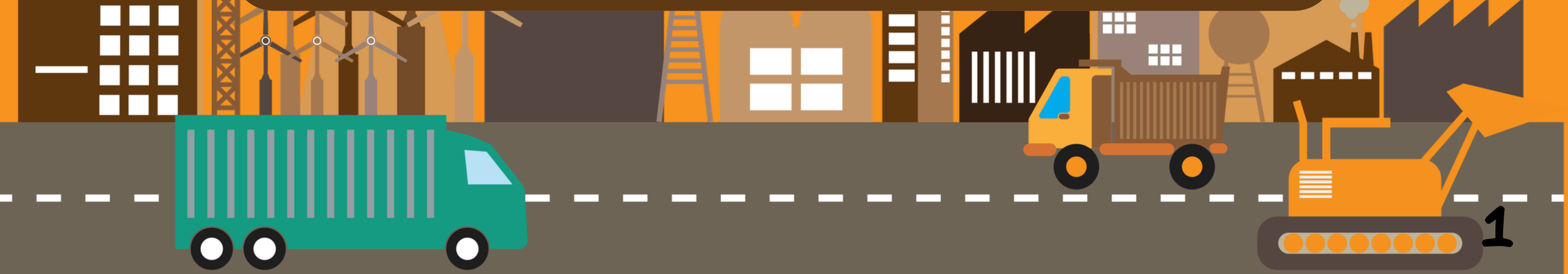
Pollution

Definition, Type & Effect



2D pollution propagation

$$\frac{\partial g}{\partial t} + u\frac{\partial g}{\partial x} + w\frac{\partial g}{\partial z} = \mu\frac{\partial^2 g}{\partial x^2} + \eta\frac{\partial^2 g}{\partial z^2} - \sigma g + f$$

แบบจำลองทางคณิตศาสตร์นี้อธิบายการแพร่กระจายของมลพิษใน 2 มิติเมื่อเวลาผ่านไป
โดยคำนึงถึงการพัดพาของลม การแพร่กระจายตามธรรมชาติ อัตราการสลายตัว
และการปล่อยมลพิษจากแหล่งกำเนิด

ตัวแปร / พารามิเตอร์	ชื่อเรียก	บทบาทในแบบจำลอง (Physical Meaning)
$g(x, z, t)$	ความเข้มข้นของมลพิษ	ตัวแปรหลักที่ต้องการศึกษา (ปริมาณมลพิษต่อพื้นที่ ณ เวลาหนึ่งๆ)
u, w	ความเร็วลม (Advection)	ความเร็วในการพัดพามลพิษไปตามแนวแกน x และแกน z ตามลำดับ
μ, η	สัมประสิทธิ์การแพร่	อัตราการกระจายตัวของมลพิษตามธรรมชาติ (Diffusion) ในแต่ละแนวแกน
σ	อัตราการสลายตัว	ความเร็วที่มลพิษลดน้อยลงอันเนื่องมาจากปฏิกิริยาเคมีหรือการตกตะกอน
f	แหล่งกำเนิด (Source)	การปล่อยมลพิษใหม่เข้าสู่ระบบ (เช่น ปล่องควันโรงงาน หรือยานพาหนะ)
t	เวลา	เวลาที่ใช้ในการคำนวณการเปลี่ยนแปลง (Time dynamic)

Domain and grid

การแบ่งพื้นที่และระยะห่างของแต่ละจุดกริด (grid spacing) ในแกน x และ z

ขอบเขตพื้นที่

$$x \in [0, L_x] \quad \text{และ} \quad z \in [0, H]$$

ขนาดกริด

$$\Delta x = \frac{L_x}{N_x - 1}$$

$$\Delta z = \frac{H}{N_z - 1}$$

กำหนด $L_x = 20.0, H = 6.0, N_x = 240, N_z = 100$

ตัวแปร / พารามิเตอร์	ชื่อเรียก	บทบาทในแบบจำลอง (Physical Meaning)
L_x, H	ขอบเขตพื้นที่ (Domain)	ขนาดความกว้างและความสูงทั้งหมดของพื้นที่จำลอง
N_x, N_z	จำนวนจุดกริด	จำนวนจุดตัวอย่างที่ใช้คำนวณในแต่ละแนวแกน (ยิ่งมากยิ่งละเอียด)
Δx	ขนาดกริดแกน X	ระยะห่างระหว่างจุดสองจุดในแนวนอน
Δz	ขนาดกริดแกน Z	ระยะห่างระหว่างจุดสองจุดในแนวตั้ง
x, z	พิกัดตำแหน่ง	ระบุตำแหน่งใดๆ ภายในพื้นที่จำลอง (Coordinate)

Domain and grid

กำหนดขนาดของพื้นที่จำลองและสร้างตารางกริด

```
# ----------------------------
# Domain and grid
# ----------------------------
Lx = 20.0
H = 6.0

Nx = 240
Nz = 100

dx = Lx / (Nx - 1)
dz = H / (Nz - 1)

x = np.linspace(0, Lx, Nx)
z = np.linspace(0, H, Nz)

X, Z = np.meshgrid(x, z, indexing="ij")
```

Parameters

ค่าคงที่ต่างๆ ที่ใช้ในสมการ

$\mu = 0.012$ **คือ สัมประสิทธิ์การแพร่ในแนวแกน x**

$\eta = 0.006$ **คือ สัมประสิทธิ์การแพร่ในแนวแกน z**

$\sigma = 0.015$ **คือ อัตราการสลายตัวของมลพิษ**

$T = 35.0$ **คือ เวลาทั้งหมดที่ใช้จำลอง**

Parameters

กำหนดค่าพารามิเตอร์ต่างๆ ที่ใช้ในสมการ เช่น ค่าสัมประสิทธิ์การแพร่ (DIFFUSION COEFFICIENTS) และเวลาทั้งหมด (T)

```
# ----------------------------
# Parameters
# ----------------------------
mu = 0.012
eta = 0.006
sigma = 0.015
a = 0.25

T = 35.0
```

Time-dependent wind field

สมการความเร็วลมในแนวแกน x (u) และแกน z (w)

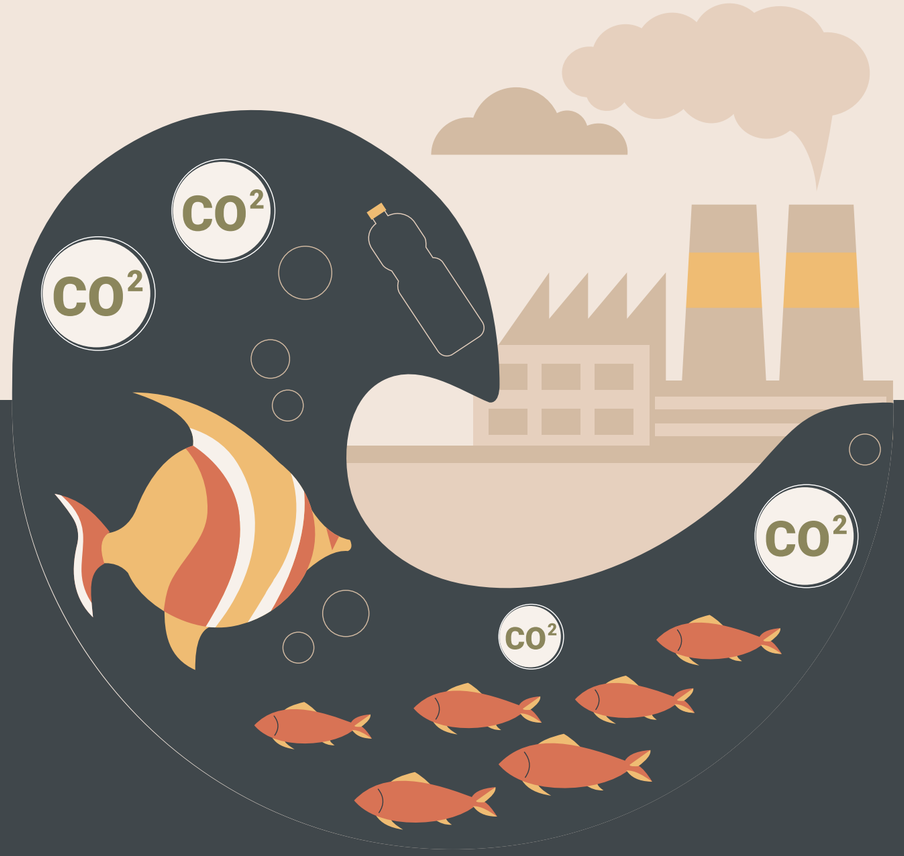
$$u(x,z,t) = 1.0 + 0.45\left(\frac{z}{H}\right) + 0.35\sin\left(\frac{2\pi t}{12.0}\right) + 0.20\sin\left(\frac{2\pi z}{H}\right)$$

1.0 **คือ ลมพื้นฐานคงที่**

$0.45\left(\frac{z}{H}\right)$ **คือ ความชันลม**

$0.35\sin\left(\frac{2\pi t}{12}\right)$ **คือ การแกว่งตามเวลาทุกๆ 12 หน่วยเวลา**

$0.20\sin\left(\frac{2\pi z}{H}\right)$ **คือ ความแปรปรวนตามความสูง**



Time-dependent wind field

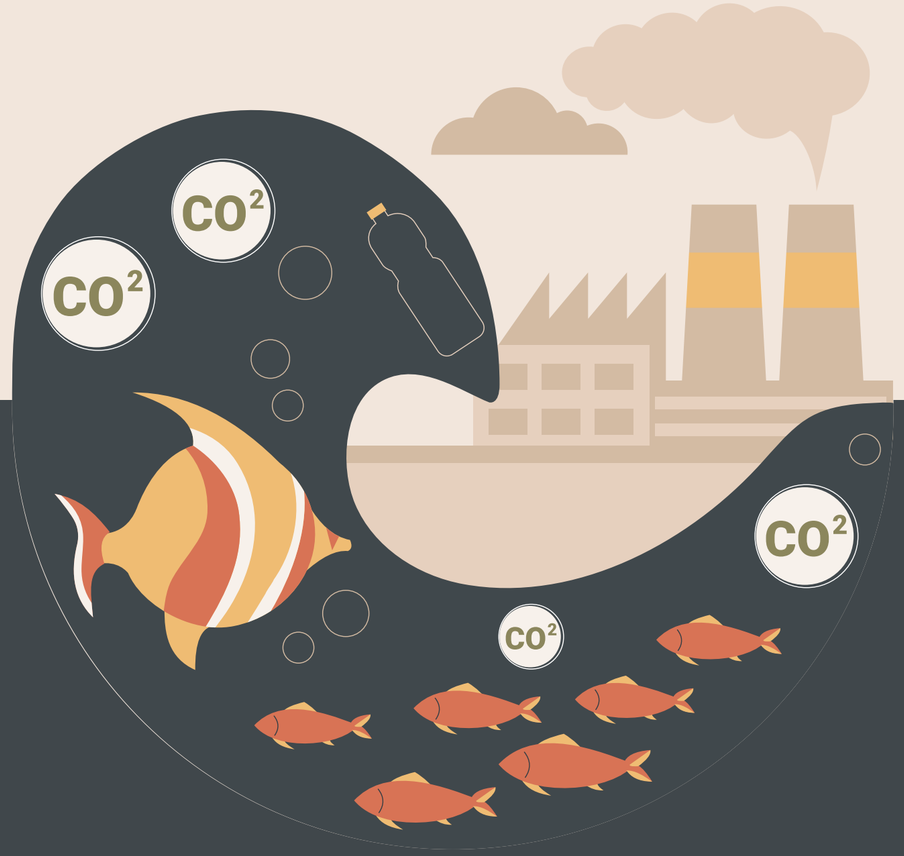
สมการความเร็วลมในแนวแกน x (u) และแกน z (w)

$$w(x,z,t) = 0.15 \sin\left(\frac{\pi x}{L_x}\right) \sin\left(\frac{\pi z}{H}\right) \cos\left(\frac{2\pi t}{10.0}\right)$$

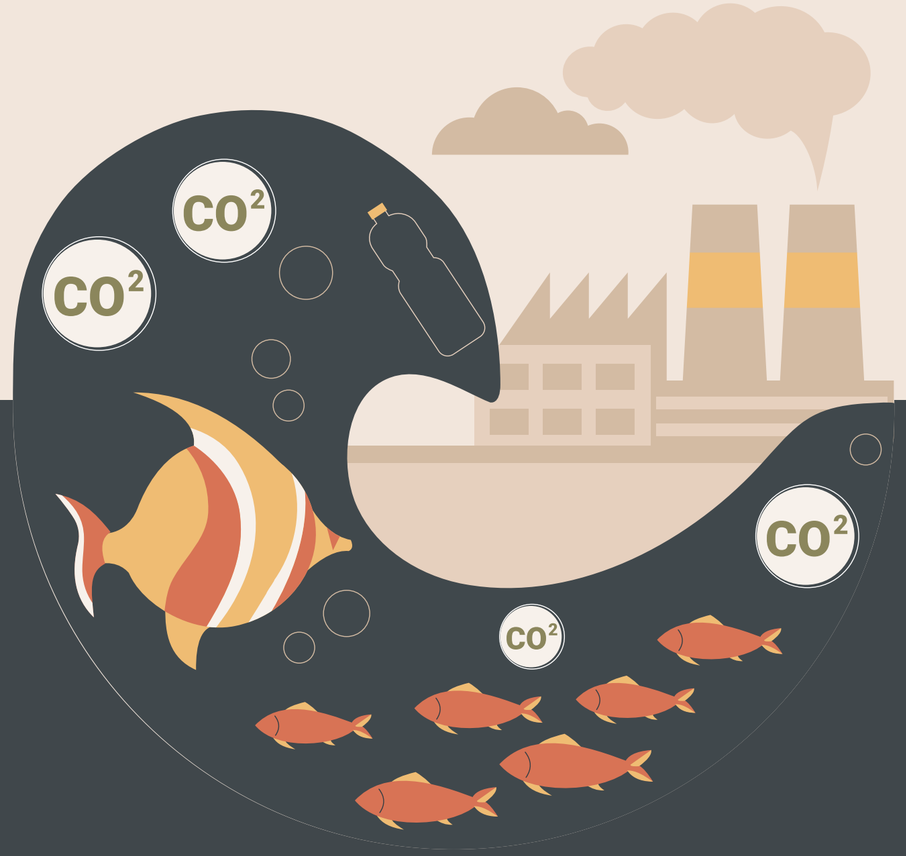
0.15 คือ ความเร็วลมแนวตั้งสูงสุดที่จะเกิดขึ้นได้ (AMPLITUDE)

$\sin\left(\frac{\pi x}{L_x}\right) \sin\left(\frac{\pi z}{H}\right)$ คือ รูปแบบเซลล์อากาศ (SPATIAL CELL)

$\cos\left(\frac{2\pi t}{10.0}\right)$ คือ การสลับทิศทางตามเวลา



Time-dependent wind field



ฟังก์ชันกำหนดความเร็วลมในแกน x (u) และแกน z (w) ซึ่งมีการเปลี่ยนแปลงตามเวลา (t) และพื้นที่

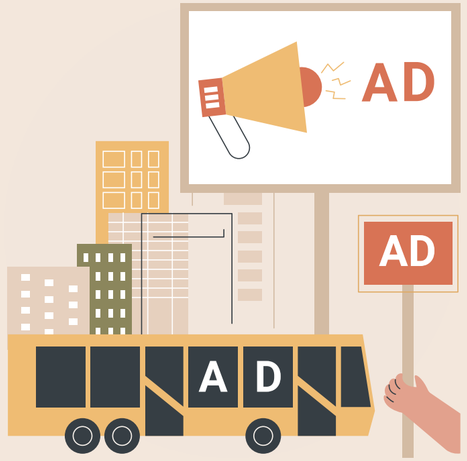
```
def velocity(t):
    u = (
        1.0
        + 0.45 * Z / H
        + 0.35 * np.sin(2 * np.pi * t / 12.0)
        + 0.20 * np.sin(2 * np.pi * Z / H)
    )

    w = (
        0.15
        * np.sin(np.pi * X / Lx)
        * np.sin(np.pi * Z / H)
        * np.cos(2 * np.pi * t / 10.0)
    )

    return u, w
```

Time step

การคำนวณ TIME STEP เพื่อรักษาเสถียรภาพของตัวแบบ (STABILITY CRITERIA)



$$\Delta t_{adv,x} = \frac{\Delta x}{\max(u)}, \quad \Delta t_{adv,z} = \frac{\Delta z}{\max(w)}$$

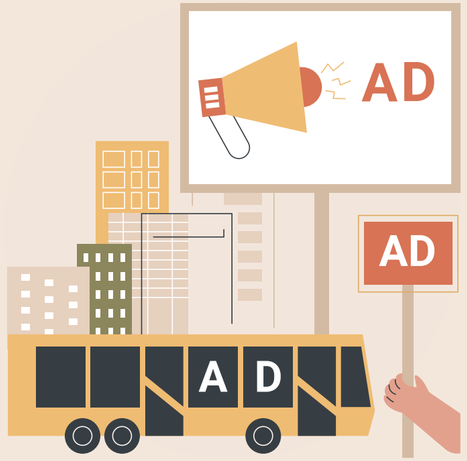
$$\Delta t_{diff,x} = \frac{(\Delta x)^2}{2\mu}, \quad \Delta t_{diff,z} = \frac{(\Delta z)^2}{2\eta}$$

$$N_t = \left\lfloor \frac{\Delta t}{T} \right\rfloor$$

$$\Delta t = 0.35 \times \min(\Delta t_{adv,x}, \Delta t_{adv,z}, \Delta t_{diff,x}, \Delta t_{diff,z})$$

Time step

คำนวณหาค่า TIME STEP (DT) ที่เหมาะสมที่สุด เพื่อให้แบบจำลองมีความเสถียร (STABILITY) โดยพิจารณาจากเงื่อนไข CFL




```
# ----------------------------
# Time step
# ----------------------------
max_u_est = 2.2
max_w_est = 0.2

dt_adv_x = dx / max_u_est
dt_adv_z = dz / max_w_est
dt_diff_x = (dx**2) / (2 * mu)
dt_diff_z = (dz**2) / (2 * eta)
# dt_diff = 0.25 / (mu / dx**2 + eta / dz**2)

dt = 0.35 * min(dt_adv_x, dt_adv_z, dt_diff_x, dt_diff_z)
Nt = int(T / dt)
```

Initial condition

ค่าความเข้มข้นของมลพิษเริ่มต้นที่เวลา t = 0 ซึ่งเกิดจากการรวมกันของฟังก์ชันเกาส์เซียน (GAUSSIAN) 3 กลุ่ม

$$g(x,z,0) = \exp\left(-\left[\frac{(x-3.0)^2}{0.50} + \frac{(z-1.0)^2}{0.20}\right]\right)$$

$$+0.8\exp\left(-\left[\frac{(x-6.0)^2}{0.70} + \frac{(z-2.5)^2}{0.35}\right]\right)$$

$$+0.5\exp\left(-\left[\frac{(x-11.0)^2}{1.20} + \frac{(z-1.8)^2}{0.45}\right]\right)$$

Initial condition

กำหนดค่าเริ่มต้นของมลพิษ (g) ที่เวลา T=0 โดยใช้การกระจายตัวแบบ GAUSSIAN 3 จุด

```
# ----------------------------
# Initial condition
# ----------------------------
g = np.zeros((Nx, Nz))

g += 1.0 * np.exp(-((X - 3.0) ** 2 / 0.50 + (Z - 1.0) ** 2 / 0.20))
g += 0.8 * np.exp(-((X - 6.0) ** 2 / 0.70 + (Z - 2.5) ** 2 / 0.35))
g += 0.5 * np.exp(-((X - 11.0) ** 2 / 1.20 + (Z - 1.8) ** 2 / 0.45))
```

Source term

ฟังก์ชัน F(X, Z, T) ที่กำหนดแหล่งกำเนิดมลพิษตามเวลา โดยกำหนดให้ฟังก์ชัน GAUSSIAN_SOURCE อยู่ในรูปของฟังก์ชัน G

$$G(x, z; x_s, z_s, s_x, s_z) = \exp\left(-\left[\frac{(x - x_s)^2}{s_x} + \frac{(z - z_s)^2}{s_z}\right]\right)$$

สมการของแหล่งกำเนิดมลพิษแต่ละจุด (f_1, f_2, f_3, f_4)

$$f_1(x, z, t) = 1.5\left[1.0 + 0.7\sin\left(\frac{2\pi t}{8.0}\right)\right]G(x, z; 2.0, 0.45, 0.35, 0.05)$$

$$f_2(x, z, t) = 1.2\exp\left(-\frac{(t - 10.0)^2}{20.0}\right)G(x, z; 7.0, 0.55, 0.50, 0.06)$$

$$f_3(x, z, t) = 3.0\exp\left(-\frac{(t - 18.0)^2}{1.5}\right)G(x, z; 4.5, 1.4, 0.35, 0.18)$$

สำหรับ F4 จุดกำเนิดมีการเคลื่อนที่ตามแกน X ตามเวลา T

$$x_s(t) = 10.0 + 2.0 \sin\left(\frac{2\pi t}{15.0}\right)$$

$$f_4(x, z, t) = 0.8 \exp\left(-\frac{(t - 24.0)^2}{40.0}\right) \exp\left(-\left[\frac{(x - x_s(t))^2}{0.60} + \frac{(z - 3.0)^2}{0.35}\right]\right)$$

แหล่งกำเนิดรวมในสมการหลักคือผลรวมของทั้ง 4 แหล่ง

$$f(x, z, t) = f_1 + f_2 + f_3 + f_4$$

Source term

ฟังก์ชันกำหนดแหล่งกำเนิดมลพิษที่ปล่อยออกมาตามช่วงเวลาและตำแหน่งต่างๆ

```python
# ----------------------------
# Source term
# ----------------------------
def gaussian_source(xs, zs, sx, sz):
    return np.exp(-((X - xs) ** 2 / sx + (Z - zs) ** 2 / sz))

def source(t):
    f1 = (
        1.5
        * (1.0 + 0.7 * np.sin(2 * np.pi * t / 8.0))
        * gaussian_source(2.0, 0.45, 0.35, 0.05)
    )

    f2 = (
        1.2 * np.exp(-((t - 10.0) ** 2) / 20.0) * gaussian_source(7.0, 0.55, 0.50, 0.06)
    )

    f3 = 3.0 * np.exp(-((t - 18.0) ** 2) / 1.5) * gaussian_source(4.5, 1.4, 0.35, 0.18)

    xs = 10.0 + 2.0 * np.sin(2 * np.pi * t / 15.0)

    f4 = (
        0.8
        * np.exp(-((t - 24.0) ** 2) / 40.0)
        * np.exp(-((X - xs) ** 2 / 0.60 + (Z - 3.0) ** 2 / 0.35))
    )

    return f1 + f2 + f3 + f4
```

Boundary conditions

```python
# ----------------------------
# Boundary conditions
# ----------------------------
def apply_boundary_conditions(g):
    # Side boundaries: g = 0
    g[0, :] = 0.0
    # g[-1, :] = 0.0

    # Bottom boundary: dg/dz = a g
    g[:, 0] = g[:, 1] / (1.0 + a * dz)

    # Top boundary: dg/dz = 0
    g[:, -1] = g[:, -2]

    return g
```

- **ขอบซ้าย (DIRICHLET BOUNDARY):**
  บังคับให้ความเข้มข้นมลพิษที่ขอบฝั่งซ้ายเป็น 0 เสมอ เพื่อจำลองอากาศบริสุทธิ์ที่ถูกพัดเข้ามา

- **ขอบล่างหรือพื้นดิน (ROBIN BOUNDARY):**
  - มาจากสมการ $\frac{\partial g}{\partial z} = a \cdot g$
  - แปลงเป็น FINITE DIFFERENCE: $\frac{g_1 - g_0}{\Delta z} = a \cdot g_0$

  ย้ายข้างจะได้สูตรในโค้ด เพื่อจำลองพื้นดินที่สามารถดูดซับมลพิษไว้ได้บางส่วนตามค่าสัมประสิทธิ์ A

- **ขอบบนหรือเพดานอากาศ (NEUMANN BOUNDARY):**
  - มาจากสมการ $\frac{\partial g}{\partial z} = 0$
  - ความชันของมลพิษเป็นศูนย์ หมายถึงมลพิษสามารถลอยทะลุขอบบนออกไปได้อย่างอิสระ ไม่มีการสะท้อนกลับ

One explicit Euler step

```python
# ----------------------------
# One explicit Euler step
# ----------------------------
def step(g, t):
    g = apply_boundary_conditions(g)

    gn = g.copy()

    u, w = velocity(t)

    G = gn[1:-1, 1:-1]
    U = u[1:-1, 1:-1]
    W = w[1:-1, 1:-1]

    # Upwind derivative in x
    gx_backward = (gn[1:-1, 1:-1] - gn[:-2, 1:-1]) / dx
    gx_forward = (gn[2:, 1:-1] - gn[1:-1, 1:-1]) / dx
    Dx_up = np.where(U >= 0.0, gx_backward, gx_forward)

    # Upwind derivative in z
    gz_backward = (gn[1:-1, 1:-1] - gn[1:-1, :-2]) / dz
    gz_forward = (gn[1:-1, 2:] - gn[1:-1, 1:-1]) / dz
    Dz_up = np.where(W >= 0.0, gz_backward, gz_forward)
```

- **การเตรียมข้อมูล (SLICE ARRAY):** โค้ดส่วน [1:−1, 1:−1] คือการตัดข้อมูลขอบทิ้ง เพื่อคำนวณเฉพาะ "พื้นที่ด้านในกริด" เท่านั้น
- **ทฤษฎี UPWIND DIFFERENCING:** การหาอนุพันธ์อันดับหนึ่ง $(\frac{\partial g}{\partial x}$ และ $\frac{\partial g}{\partial z})$ สำหรับพจน์การพัดพา
- **การเลือกทิศทาง (NP.WHERE):**
  - ถ้าลมพัดไปข้างหน้า $(U \geq 0)$
    ใช้ข้อมูลจากจุดด้านหลัง (BACKWARD DIFFERENCE)
  - ถ้าลมพัดย้อนกลับ $(U < 0)$
    ใช้ข้อมูลจากจุดด้านหน้า (FORWARD DIFFERENCE)
  - **สาเหตุที่ต้องใช้ทฤษฎีนี้:** เพื่อรักษาเสถียรภาพ ป้องกันไม่ให้เกิดการคำนวณที่ทำให้มลพิษมีค่าแกว่ง (NUMERICAL OSCILLATIONS)

ทฤษฎีสมการระเบียบวิธี Upwind Scheme Theory

สมการตั้งต้น (1D ADVECTION EQUATION) การพิจารณาการพัดพามวลสารใน 1 มิติ (ไม่คิดการแพร่) จะอยู่ภายใต้สมการเชิงอนุพันธ์:

$$\frac{\partial g}{\partial t} + u\frac{\partial g}{\partial x} = 0$$

โดยที่ g คือความเข้มข้น
u คือความเร็วในการพัดพา

แนวคิดของ UPWIND SCHEME (การหาพจน์ $\frac{\partial g}{\partial x}$) หลักการสำคัญคือ "ทิศทางของข้อมูลต้องไหลตามทิศทางของความเร็ว"

การหาความชันเชิงพื้นที่จึงต้องเลือกใช้จุดข้อมูลที่อยู่ "เหนือลม (UPWIND)" เสมอ แบ่งได้เป็น 2 กรณี:

กรณีที่ 1: ความเร็วพัดไปข้างหน้า (u > 0)

$$u\frac{\partial g}{\partial x} \approx u\frac{g_i - g_{i-1}}{\Delta x}$$

ข้อมูลถูกพัดมาจากด้านหลัง (ซ้ายไปขวา) จึงต้องใช้

ข้อมูลจากจุดปัจจุบัน (i) และจุดก่อนหน้า (i-1)

หรือเรียกว่า Backward Difference

กรณีที่ 2: ความเร็วพัดย้อนกลับ (u < 0)

$$u\frac{\partial g}{\partial x} \approx u\frac{g_{i+1} - g_i}{\Delta x}$$

ข้อมูลถูกพัดมาจากด้านหน้า (ขวาไปซ้าย) จึงต้องใช้

ข้อมูลจากจุดกริดปัจจุบัน (i) และจุดถัดไป (i+1)

หรือเรียกว่า Forward Difference

ทำไมต้อง UPWIND?

- มีทิศทางชัดเจน: ข้อมูลต้องไหลตามกระแสลม จึงต้องดึงค่าจากฝั่ง "เหนือลม" มาใช้ ตามหลักฟิสิกส์ที่ว่า "สิ่งที่อยู่ปลายลมไม่สามารถส่งผลกระทบต่อสิ่งที่อยู่ต้นลมได้"
- ป้องกันกราฟแกว่ง: ช่วยเพิ่มความหนืดเทียม (NUMERICAL DIFFUSION) ทำให้ผลลัพธ์นิ่งและเสถียรไม่สั่นเป็นคลื่น

One explicit Euler step

```
# Central diffusion
Dxx = (gn[2:, 1:-1] - 2.0 * G + gn[:-2, 1:-1]) / dx**2
Dzz = (gn[1:-1, 2:] - 2.0 * G + gn[1:-1, :-2]) / dz**2

F = source(t)[1:-1, 1:-1]

g[1:-1, 1:-1] = (
    G - dt * (U * Dx_up + W * Dz_up) + dt * (mu * Dxx + eta * Dzz - sigma * G + F)
)

g[g < 0.0] = 0.0

return apply_boundary_conditions(g)
```

กรองความผิดพลาดจากการคำนวณเชิงตัวเลข โดยบังคับไม่ให้ความเข้มข้นของมลพิษติดลบ ซึ่งเป็นไปไม่ได้ในทางฟิสิกส์

- **การแพร่ (CENTRAL DIFFUSION):** การหาอนุพันธ์อันดับสอง $(\frac{\partial^2 g}{\partial x^2})$ โดยใช้วิธี CENTRAL DIFFERENCE FORMULA:

$$D_{xx} = \frac{g_{i+1} - 2g_i + g_{i-1}}{\Delta x^2}$$

- **สมการก้าวหน้าเวลา (EXPLICIT EULER METHOD):** นี่คือการรวมพจน์ทั้งหมดเพื่อหาค่ามลพิษในอนาคต:

$$g^{new} = g^{old} + \Delta t \times (-\text{Advection} + \text{Diffusion} - \text{Decay} + \text{Source})$$

ทฤษฎีคำนวณการแพร่ด้วย Central Diffusion Scheme

สมการตั้งต้น (1D DIFFUSION EQUATION) การพิจารณากระบวนการแพร่กระจายของมวลสาร (โดยไม่มีลมพัด) จะอยู่ภายใต้สมการเชิงอนุพันธ์อันดับสอง: $\frac{\partial g}{\partial t} = \mu \frac{\partial^2 g}{\partial x^2}$ โดยที่ g คือความเข้มข้น μ คือสัมประสิทธิ์การแพร่

แนวคิดของ CENTRAL DIFFERENCE (การหาพจน์ $\frac{\partial^2 g}{\partial x^2}$) พจน์การแพร่คือ "ความโค้งของการกระจายตัว" (อนุพันธ์อันดับสอง) ซึ่งเกิดจากการกระจายตัวจากจุดที่มีความเข้มข้นสูงไปยังจุดที่ต่ำกว่าในทุกทิศทาง การหาค่านี้จึงต้องใช้สมการ CENTRAL DIFFERENCE FORMULA ซึ่งดึงข้อมูลจากจุดรอบข้างทั้ง 2 ฝั่งมาประเมินร่วมกับจุดปัจจุบัน:

$$\frac{\partial^2 g}{\partial x^2} \approx \frac{g_{i+1} - 2g_i + g_{i-1}}{\Delta x^2}$$

ทำไมต้อง CENTRAL?

- กระจายรอบทิศทาง: การแพร่บานออกทุกทิศเท่าๆ กัน จึงต้องดึงข้อมูลทั้ง "ข้างหน้าและข้างหลัง" มาหาค่าเฉลี่ย
- ความแม่นยำสูงสุด: เป็นวิธีที่คำนวณการบานออกได้แม่นยำที่สุด และเหมาะสมที่สุดเมื่อไม่มีทิศทางลมมาเกี่ยวข้อง

Scientific Foundations

การจำลองการแพร่กระจายของมลพิษคือการแก้ปัญหาทางคณิตศาสตร์ของสมการ ADVECTION-DIFFUSION:

$$\frac{\partial C}{\partial t} + u \cdot \nabla C = D \nabla^2 C + S$$

Advection Term

การที่มลพิษถูกพัดพาไปตามทิศทางและความเร็วของกระแสลม (Vector u)

Diffusion Term

การที่มลพิษแพร่จากจุดที่มีความเข้มข้นสูงไปยังจุดที่ต่ำกว่า (Coefficient D)

Types of Dispersion Models

Model Type	Core Concept
Box Model	ถือว่ามลพิษผสมกันอย่างรวดเร็วในกล่องพิกัด
Gaussian Model	ติดตามการเคลื่อนที่ของกลุ่มก้อนอากาศ (Puffs)
CFD Model	แก้สมการ Fluid Dynamics บน Grid ละเอียด

ในโค้ดชุดนี้ เราใช้แนวทาง CFD-LITE โดยการแก้สมการบนตารางพิกัดแบบ 2D GRID เพื่อดูการเปลี่ยนแปลงเชิงพื้นที่และเวลา

1. Time Integration Loop

```python
frames = []
times = []

save_every = 8

for n in range(Nt):
    t = n * dt
    g = step(g, t)

    if n % save_every == 0:
        frames.append(g.copy())
        times.append(t)

print(f"Stored {len(frames)} frames.")
```

NUMERICAL STEP :

- วนลูปตามจำนวน NT
- ใช้ STEP(G, T) เพื่อขยับสถานะ
- บันทึกเฉพาะบางเฟรมเพื่อลดภาระ RAM
- G.COPY() สำคัญมากเพื่อไม่ให้ข้อมูลทับซ้อนกัน

Numerical Stability

The CFL Condition

การเลือก dt (Time Step) ต้องสัมพันธ์กับความเร็วลมเพื่อป้องกัน "Numerical Instability" หรือการคำนวณที่ค่าระเบิดออก

$$C = \frac{u \Delta t}{\Delta x} \leq C_{max}$$

Step Function

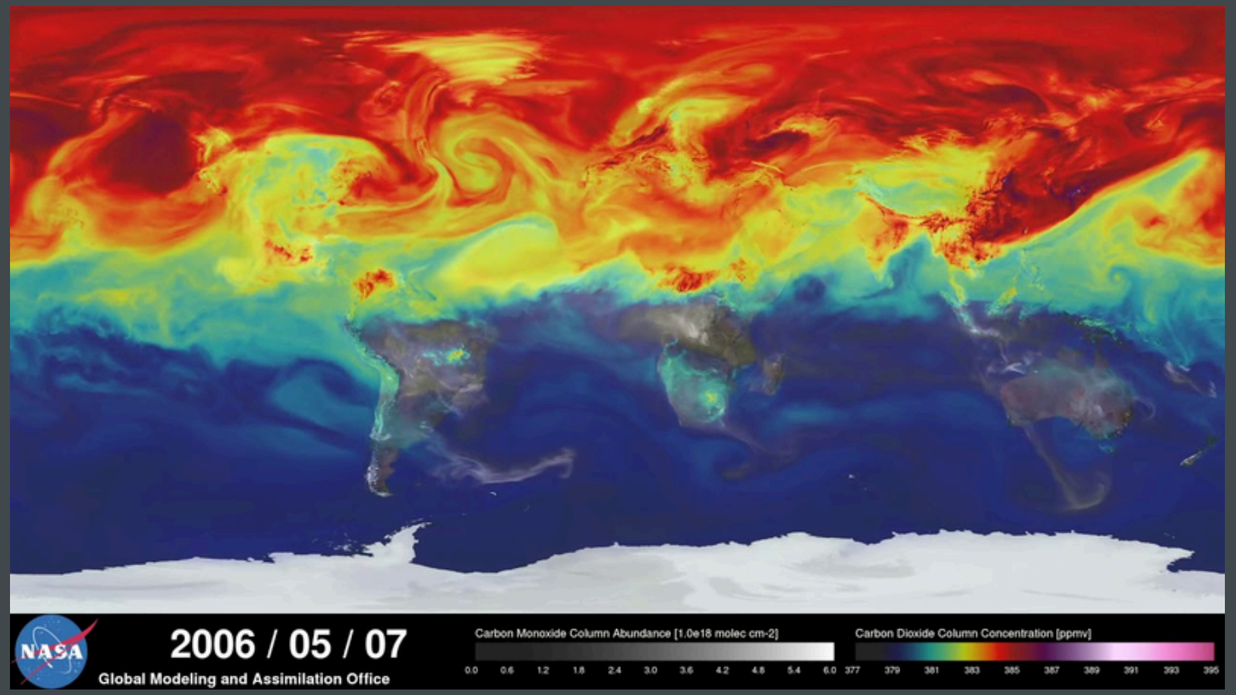
ฟังก์ชัน step ทำการอัปเดตค่าความเข้มข้นในทุกช่องพิกัด โดยมักใช้ Finite Difference Method ในการหาค่าอนุพันธ์

2. Concentration Plot

```python
# Concentration plot
vmax_g = np.percentile(np.array(frames), 99.5)

im = ax1.imshow(
    frames[0].T,
    origin="lower",
    extent=[0, Lx, 0, H],
    aspect="auto",
    vmin=0,
    vmax=vmax_g,
    cmap="viridis",
)

cbar1 = plt.colorbar(im, ax=ax1)
cbar1.set_label("pollutant concentration")
```



การใช้ PERCENTILE ๙๙.5 ช่วยให้การตั้งค่าสี (DYNAMIC RANGE) ครอบคลุมข้อมูลส่วนใหญ่โดยไม่ถูกกวนด้วยค่าสุดโต่ง

Scientific Color Selection

VIRIDIS (CONCENTRATION)

เป็น COLORMAP ที่มีความเป็น PERCEPTUALLY UNIFORM คือสายตามนุษย์รับรู้การเปลี่ยนแปลงของค่าตามสีได้อย่างแม่นยำและเป็นมิตรต่อผู้ที่มีอาการตาบอดสี

PLASMA (WIND SPEED)

ให้ CONTRAST ที่สูงขึ้นสำหรับข้อมูล VECTOR FIELD ช่วยให้เห็นความแตกต่างของความเร็วลมได้ชัดเจนในจุดที่มีค่าต่ำ

3. Wind Field Dynamics

```
skip = (slice(None, None, 12), slice(None, None, 6))

u0, w0 = velocity(times[0])
speed0 = np.sqrt(u0**2 + w0**2)

quiv = ax2.quiver(
    X[skip],
    Z[skip],
    u0[skip],
    w0[skip],
    speed0[skip],
    cmap="plasma",
    scale=25,
)
```

VECTOR ANALYSIS :

- SKIP: ลดความหนาแน่นลูกศร
- QUIVER: วาดลูกศรตามพิกัด X, Z
- SPEED0: ใช้กำหนดสีตามความเร็ว

4. Animation Logic

```python
def update(k):
    t = times[k]

    im.set_data(frames[k].T)
    title1.set_text(f"Pollution, t = {t:.2f}")

    u, w = velocity(t)
    speed = np.sqrt(u**2 + w**2)

    quiv.set_UVC(u[skip], w[skip], speed[skip])
    title2.set_text(f"Wind, t = {t:.2f}")

    return im, quiv, title1, title2
```

ในทุกๆ เฟรม UPDATE จะทำการเปลี่ยนข้อมูล DATA BUFFER ของ MATPLOTLIB OBJECTS โดยไม่สร้างกราฟใหม่ ช่วยเพิ่มประสิทธิภาพให้แอนิเมชันลื่นไหล

5. Final Deployment

```
ani = FuncAnimation(
    fig,
    update,
    frames=len(frames),
    interval=60,
    blit=False,
)

plt.tight_layout()
plt.show()
```

DEPLOYMENT OPTIONS

คุณสามารถบันทึกผลลัพธ์เป็นไฟล์ GIF หรือ MP4 ได้ด้วย:

```
ANI.SAVE("POLLUTION.GIF", WRITER="PILLOW", FPS=20)
```

Real-World Impact

การจำลองนี้ไม่ได้เป็นเพียงแค่กราฟ แต่คือเครื่องมือสำคัญใน AIR QUALITY POLICY:

- EMERGENCY RESPONSE : จำลองการรั่วไหลของก๊าซพิษ
- URBAN PLANNING : วางแผนสร้างโรงงานตามทิศทางลมหลัก
- COMPLIANCE : ตรวจสอบค่ามลพิษตามมาตรฐาน NAAQS

Thank you