數值方法 作業 11

工科系 115　E14116401 張瑋哲

第一題：

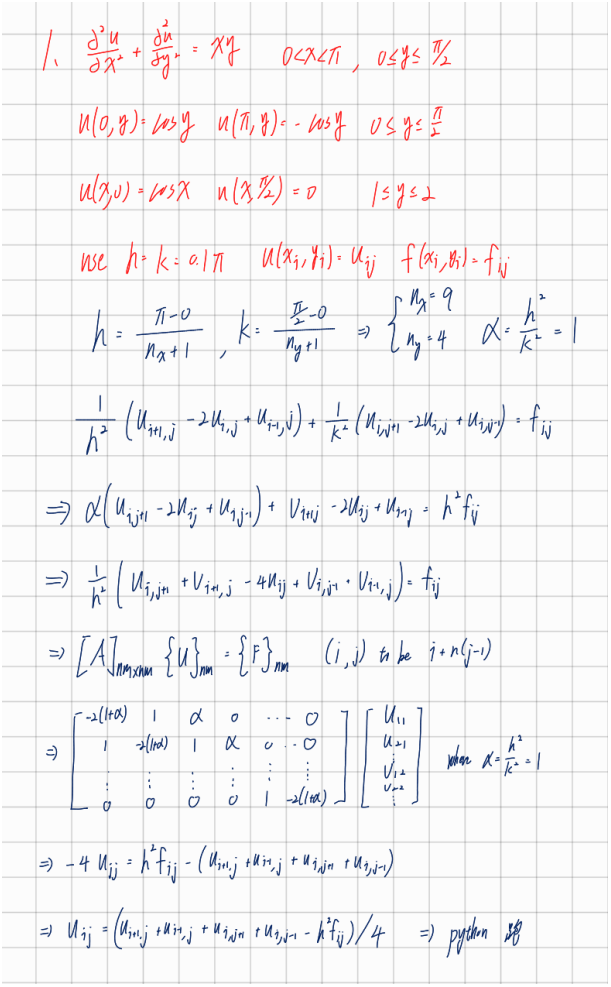

圖 1、第一題計算式

```
PS C:\Users\gunda> & C:/ProgramData/anaconda3/python.exe "d:/ForClass/1132/1132Numerical/HW12/import numpy as np.py"
Converged in 49 iterations.

u(x, y) values:
  1.000000   0.951057   0.809017   0.587785   0.309017   0.000000  -0.309017  -0.587785  -0.809017  -0.951057  -1.000000
  0.951057   0.753226   0.555909   0.333232   0.085807  -0.173196  -0.424257  -0.645220  -0.814475  -0.915787  -0.951057
  0.809017   0.564622   0.347641   0.132648  -0.086862  -0.305632  -0.511151  -0.686178  -0.809947  -0.858894  -0.809017
  0.587785   0.368085   0.176348  -0.004975  -0.182347  -0.353912  -0.511645  -0.642021  -0.724390  -0.725487  -0.587785
  0.309017   0.172806   0.053087  -0.058882  -0.166748  -0.269910  -0.364162  -0.441313  -0.486321  -0.467875  -0.309017
  0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000

Corresponding function indices:
u_00,00 u_01,00 u_02,00 u_03,00 u_04,00 u_05,00 u_06,00 u_07,00 u_08,00 u_09,00 u_10,00
u_00,01 u_01,01 u_02,01 u_03,01 u_04,01 u_05,01 u_06,01 u_07,01 u_08,01 u_09,01 u_10,01
u_00,02 u_01,02 u_02,02 u_03,02 u_04,02 u_05,02 u_06,02 u_07,02 u_08,02 u_09,02 u_10,02
u_00,03 u_01,03 u_02,03 u_03,03 u_04,03 u_05,03 u_06,03 u_07,03 u_08,03 u_09,03 u_10,03
u_00,04 u_01,04 u_02,04 u_03,04 u_04,04 u_05,04 u_06,04 u_07,04 u_08,04 u_09,04 u_10,04
u_00,05 u_01,05 u_02,05 u_03,05 u_04,05 u_05,05 u_06,05 u_07,05 u_08,05 u_09,05 u_10,05
```

圖 2、第一題計算結果

```
D: > ForClass > 1132 > 1132Numerical > HW12 > import numpy as np.py > solve_pde
```

```python
import numpy as np

def solve_pde():
    Nx = 11  # x: 0 to π (Δx=0.1π*10)
    Ny = 6   # y: 0 to π/2 (Δy=0.1π*5)
    h = 0.1 * np.pi
    tolerance = 1e-6
    max_iter = 10000

    def f(x, y):
        return x * y

    u = np.zeros((Nx, Ny))
    x = np.array([i * h for i in range(Nx)])
    y = np.array([j * h for j in range(Ny)])

    # BC
    for j in range(Ny):
        u[0, j] = np.cos(y[j])
        u[Nx-1, j] = -np.cos(y[j])
    for i in range(Nx):
        u[i, 0] = np.cos(x[i])
        u[i, Ny-1] = 0.0

    converged = False
    for iter in range(max_iter):
        max_error = 0.0
        for i in range(1, Nx-1):
            for j in range(1, Ny-1):
                u_old = u[i, j]
                u[i, j] = 0.25 * (u[i+1,j] + u[i-1,j] + u[i,j+1] + u[i,j-1] - h**2 * f(x[i], y[j]))
                error = abs(u[i, j] - u_old)
                max_error = max(max_error, error)
        if max_error < tolerance:
            print(f"Converged in {iter+1} iterations.")
            converged = True
            break
```

Ctrl+L to chat, Ctrl+K to generate

```python
    #print u(x, y)
    print("\nu(x, y) values:")
    for j in (range(Ny)):
        print(" ".join([f"{u[i,j]:10.6f}" for i in range(Nx)]))

    #print u_x,y
    print("\nCorresponding function indices:")
    for j in (range(Ny)):
        print(" ".join([f"u_{i:02d},{j:02d}" for i in range(Nx)]))

if __name__ == "__main__":
    solve_pde()
```

圖 3、第一題程式碼

第二題：

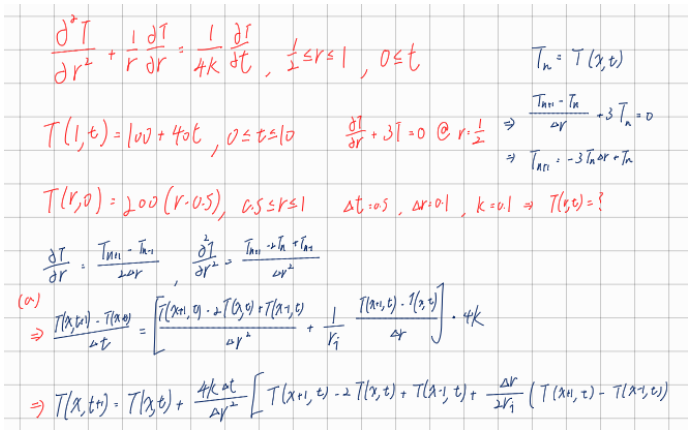

圖 4、第二題(a)計算式

```
PS C:\Users\gunda> & C:/ProgramData/anaconda3/python.exe d:/ForClass/1132/1132Numerical/HW12/2a1.py
參數設定:
dr = 0.1, dt = 0.5, K = 0.1
N_r = 6, N_t = 21
r範圍: [0.5, 1.0]
alpha = 20
結果顯示
========================================================================
     r    t=0.0       t=2.0                t=4.0                        t=6.0                          t=8.0                                t=10.0
0.5000   0.0000 -214390.5977   2605824273774.0347  14833350342014920294 4.0000   505902920829598265957469388 8.0000  160525312351596662437666182999834624.0000
0.6000  20.0000 -278707.7770   3387571555906.2451  19283355444619395072 0.0000   657673797078477811715407872 0.0000  208682906057075675926361296867426304.0000
0.7000  40.0000  106215.8346 -106021021501 35.6777 -41864155079022464204 8.0000 -136194165680799083079023984 64.0000 -428266737553268985685062628283514880.0000
0.8000  60.0000  697419.3702  160757901633 15.3105  45767170262812858777 6.0000  139862958744569584631341383 68.0000  434224354754990022665650196392706048.0000
0.9000  80.0000 -912056.9943 -124192926397 52.4121 -29132640472040754388 8.0000 -846638375218136529298364825 6.0000 -259980765375755607371555230242045952.0000
1.0000 100.0000     180.0000             260.0000                     340.0000                       420.0000                             500.0000
```

圖 5、第二題(a)計算結果(數值爆炸)

```python
import numpy as np
import pandas as pd

def solve_heat_equation():

    # 參數設定
    dr = 0.1
    dt = 0.5
    K = 0.1
    N_r = 6          # 範圍 [0.5, 1.0]，共 6 點（包含端點）
    N_t = 21         # t 從 [0,10]，Δt=0.5 → 21 步
    r_min = 0.5
    r_max = 1.0
    alpha = 20

    print("參數設定:")
    print(f"dr = {dr}, dt = {dt}, K = {K}")
    print(f"N_r = {N_r}, N_t = {N_t}")
    print(f"r範圍: [{r_min}, {r_max}]")
    print(f"alpha = {alpha}")

    # 建立 r 座標
    r = np.zeros(N_r)
    for i in range(N_r):
        r[i] = r_min + i * dr

    # 建立 T 矩陣 T[時間][空間]
    T = np.zeros((N_t, N_r))

    # 初始條件 T(r,0) = 200(r - 0.5)
    for i in range(N_r):
        T[0][i] = 200 * (r[i] - 0.5)

    # 時間迴圈
    for n in range(N_t - 1):
        t = n * dt

        # 更新內部節點 i = 1 到 N_r - 2
        for i in range(1, N_r - 1):
            r_i = r[i]
            term1 = T[n][i + 1] - 2 * T[n][i] + T[n][i - 1]
            term2 = (T[n][i + 1] - T[n][i]) * (dr / r_i)
            T[n + 1][i] = T[n][i] + alpha * (term1 + term2)

        # 邊界條件：右邊 T(1, t) = 100 + 40t
        T[n + 1][N_r - 1] = 100 + 40 * (t + dt)

        # 邊界條件：左邊 ∂T/∂r + 3T = 0
        T[n + 1][0] = T[n + 1][1] / (1 + 3 * dr)

    return r, T

# 執行計算
r, T = solve_heat_equation()

# 表格顯示結果
print("結果顯示")
print("="*80)
time_indices = [0, 4, 8, 12, 16, 20]
evolution_data = {'r': r}
for t_idx in time_indices:
    t_val = t_idx * 0.5
    evolution_data[f't={t_val:.1f}'] = T[t_idx]

evolution_table = pd.DataFrame(evolution_data)
print(evolution_table.to_string(index=False, float_format='%.4f'))
```

圖 6、第二題(a)程式碼

```
PS C:\Users\gunda> & C:/ProgramData/anaconda3/python.exe d:/ForClass/1132/1132Numerical/HW12/2b.py
 t\r      0.5000          0.6000          0.7000          0.8000          0.9000          1.0000
 0.0      0.0000         20.0000         40.0000         60.0000         80.0000        100.0000
 0.5      0.0000         29.9295         55.7128         78.7915        100.0395        120.0000
 1.0      0.0000         35.6690         66.1153         92.9876        117.3665        140.0000
 1.5      0.0000         40.9794         75.8993        106.6198        134.3677        160.0000
 2.0      0.0000         46.2413         85.6113        120.1853        151.3301        180.0000
 2.5      0.0000         51.4976         95.3150        133.7430        168.2880        200.0000
 3.0      0.0000         56.7532        105.0177        147.2997        185.2454        220.0000
 3.5      0.0000         62.0088        114.7203        160.8564        202.2027        240.0000
 4.0      0.0000         67.2643        124.4229        174.4131        219.1600        260.0000
 4.5      0.0000         72.5199        134.1254        187.9697        236.1173        280.0000
 5.0      0.0000         77.7755        143.8280        201.5264        253.0746        300.0000
 5.5      0.0000         83.0310        153.5306        215.0830        270.0319        320.0000
 6.0      0.0000         88.2866        163.2332        228.6397        286.9892        340.0000
 6.5      0.0000         93.5421        172.9357        242.1963        303.9466        360.0000
 7.0      0.0000         98.7977        182.6383        255.7530        320.9039        380.0000
 7.5      0.0000        104.0533        192.3409        269.3096        337.8612        400.0000
 8.0      0.0000        109.3088        202.0435        282.8663        354.8185        420.0000
 8.5      0.0000        114.5644        211.7460        296.4229        371.7758        440.0000
 9.0      0.0000        119.8199        221.4486        309.9796        388.7331        460.0000
 9.5      0.0000        125.0755        231.1512        323.5362        405.6904        480.0000
10.0      0.0000        130.3311        240.8538        337.0929        422.6477        500.0000
```

圖 7、第二題(b)計算結果

D: > ForClass > 1132 > 1132Numerical > HW12 > 2b.py > ...

```python
import numpy as np

dr = 0.1
dt = 0.5
K = 0.1
N_r = 6     # 範圍 [0.5, 1.0]
N_t = 21
r_min = 0.5

alpha = 4 * K * dt / (dr**2)  # α = 4KΔt/Δr²

# 初始化網格
r = np.array([r_min + i*dr for i in range(N_r)])
T = np.zeros((N_t, N_r))

# 初始條件
T[0, :] = 200 * (r - 0.5)

for n in range(N_t-1):
    t = n * dt
    a = np.zeros(N_r-2)
    b = np.zeros(N_r-2)
    c = np.zeros(N_r-2)
    d = np.zeros(N_r-2)

    for i in range(1, N_r-1):
        ri = r[i]
        coeff1 = alpha
        coeff2 = alpha * dr / (2 * ri)

        a[i-1] = -coeff1 + coeff2
        b[i-1] = 1 + 2*coeff1
        c[i-1] = -coeff1 - coeff2
        d[i-1] = T[n, i]

    # 邊界條件處理
    T[n+1, 0] = T[n+1, 1] / (1 + 3*dr)
    d[0] -= a[0] * T[n+1, 0]

    T[n+1, -1] = 100 + 40*(t + dt)
    d[-1] -= c[-1] * T[n+1, -1]

    for i in range(1, N_r-2):
        m = a[i] / b[i-1]
        b[i] -= m * c[i-1]
        d[i] -= m * d[i-1]

    x = np.zeros(N_r-2)
    x[-1] = d[-1] / b[-1]
    for i in range(N_r-3, 0, -1):
        x[i-1] = (d[i-1] - c[i-1]*x[i]) / b[i-1]

    T[n+1, 1:-1] = x

header = [' t\\r'] + [f'{val:>8.4f}' for val in r]
print('\t'.join(header))
for j in range(N_t):
    current_t = 0.5 * j
    row = [f'{current_t:>4.1f}'] + [f'{T[j][i]:>8.4f}' for i in range(N_r)]
    print('\t'.join(row))
```

圖 8、第二題(b)程式碼

```
PS C:\Users\gunda> & C:/ProgramData/anaconda3/python.exe d:/ForClass/1132/1132Numerical/HW12/2c.py
T(r, t) 結果表格:
```

時間 t	r=0.5	r=0.6	r=0.7	r=0.8	r=0.9	r=1.0
0.0	0.0000	20.0000	40.0000	60.0000	80.0000	100.0000
0.5	0.0000	33.7339	60.4688	82.8827	102.4605	120.0000
1.0	0.0000	33.6374	63.9000	91.2042	116.2785	140.0000
1.5	0.0000	42.3815	77.2496	107.6380	135.0157	160.0000
2.0	0.0000	45.2722	84.8394	119.6699	150.9709	180.0000
2.5	0.0000	52.2038	95.7562	133.9816	168.4886	200.0000
3.0	0.0000	56.2150	104.7719	147.2160	185.1335	220.0000
3.5	0.0000	62.4364	114.8510	160.8563	202.2662	240.0000
4.0	0.0000	66.9126	124.3602	174.4565	219.1220	260.0000
4.5	0.0000	72.8173	134.1477	187.9060	236.1425	280.0000
5.0	0.0000	77.5187	143.8297	201.5977	253.0552	300.0000
5.5	0.0000	83.2560	153.5147	215.0111	270.0492	320.0000
6.0	0.0000	88.0874	163.2574	228.7086	286.9723	340.0000
6.5	0.0000	93.7199	172.9067	242.1320	303.9640	360.0000
7.0	0.0000	98.6382	182.6699	255.8119	320.8857	380.0000
7.5	0.0000	104.1970	192.3081	269.2561	337.8801	400.0000
8.0	0.0000	109.1790	202.0766	282.9146	354.7990	420.0000
8.5	0.0000	114.6820	211.7131	296.3795	371.7956	440.0000
9.0	0.0000	119.7132	221.4809	310.0185	388.7132	460.0000
9.5	0.0000	125.1725	231.1197	323.5013	405.7102	480.0000
10.0	0.0000	130.2428	240.8842	337.1241	422.6281	500.0000

圖 9、第二題(c)計算結果

```python
import numpy as np

# 常數定義
dr = 0.1
dt = 0.5
K = 0.1
N_r = 6         # 範圍 [0.5, 1.0]
N_t = 21
r_min = 0.5

alpha = 4 * K * dt / (dr * dr)  # α = 4KΔt / Δr²

def main():
    # 初始化半徑陣列
    r = np.array([r_min + i * dr for i in range(N_r)])

    # 初始化溫度矩陣
    T = np.zeros((N_t, N_r))

    # 初始條件
    for i in range(N_r):
        T[0, i] = 200 * (r[i] - 0.5)

    # Crank-Nicolson
    for n in range(N_t - 1):
        t = n * dt

        # 三對角係數
        a = np.zeros(N_r - 2)
        b = np.zeros(N_r - 2)
        c = np.zeros(N_r - 2)
        d = np.zeros(N_r - 2)

        for i in range(1, N_r - 1):
            ri = r[i]
            coeff1 = alpha / 2.0
            coeff2 = coeff1 * dr / (2 * ri)

            a[i - 1] = -coeff1 + coeff2
            b[i - 1] = 1 + 2 * coeff1
            c[i - 1] = -coeff1 - coeff2
```

```python
            # RHS 計算 (forward half)
            term1 = alpha / 2 * (T[n, i + 1] - 2 * T[n, i] + T[n, i - 1])
            term2 = (alpha * dr / (4 * ri)) * (T[n, i + 1] - T[n, i - 1])
            d[i - 1] = T[n, i] + term1 + term2

        # 邊界條件
        T[n + 1, 0] = T[n + 1, 1] / (1 + 3 * dr)
        d[0] -= a[0] * T[n + 1, 0]
        T[n + 1, N_r - 1] = 100 + 40 * (t + dt)
        d[N_r - 3] -= c[N_r - 3] * T[n + 1, N_r - 1]


        for i in range(1, N_r - 2):
            m = a[i] / b[i - 1]
            b[i] -= m * c[i - 1]
            d[i] -= m * d[i - 1]

        x = np.zeros(N_r - 2)
        x[N_r - 3] = d[N_r - 3] / b[N_r - 3]
        for i in range(N_r - 4, -1, -1):
            x[i] = (d[i] - c[i] * x[i + 1]) / b[i]

        for i in range(1, N_r - 1):
            T[n + 1, i] = x[i - 1]

    print("T(r, t) 結果表格:")
    print("=" * 100)
    print(f"{'時間 t':>6} | {'r=0.5':>11} | {'r=0.6':>11} | {'r=0.7':>11} | {'r=0.8':>11} | {'r=0.9':>11} | {'r=1.0':>11} |")
    print("-" * 100)

    for j in range(N_t):
        t_val = j * dt
        print(f"{t_val:8.1f} |", end="")
        for i in range(N_r):
            print(f"{T[j, i]:12.4f} |", end="")
        print()

if __name__ == "__main__":
    main()
```

圖 10、第二題(c)程式碼

第三題：

```
PS C:\Users\gunda> & C:/ProgramData/anaconda3/python.exe d:/ForClass/1132/1132Numerical/HW12/3.py
Converged in 38 iterations.
r\theta      0.0°          12.0°          24.0°          36.0°          48.0°          60.0°
--------------------------------------------------
   0.5 |    0.000         50.000         50.000         50.000         50.000          0.000
   0.6 |    0.000         35.772         48.075         48.075         35.772          0.000
   0.7 |    0.000         37.453         53.637         53.637         37.453          0.000
   0.8 |    0.000         48.144         65.485         65.485         48.144          0.000
   0.9 |    0.000         67.905         81.753         81.753         67.905          0.000
   1.0 |    0.000        100.000        100.000        100.000        100.000          0.000
```

圖 11、第三題計算結果

```python
import numpy as np
import math

# 常數定義
Nr = 6          # r: 0.5 to 1, step = 0.1
Nth = 6         # theta: 0 to pi/3, step = pi/15
r0 = 0.5
dr = 0.1
dtheta = math.pi / 15
tol = 1e-5
max_iter = 10000

def main():
    # 初始化溫度矩陣 T[i][j] = T(r_i, theta_j)
    T = np.zeros((Nr, Nth))

    # 初始化 r, theta 陣列
    r = np.array([r0 + i * dr for i in range(Nr)])
    theta = np.array([j * dtheta for j in range(Nth)])

    # 邊界條件設定
    # r = 0.5 和 r = 1.0 的邊界
    for j in range(Nth):
        T[0, j] = 50     # r = 0.5
        T[Nr-1, j] = 100  # r = 1.0

    # theta = 0 和 theta = pi/3 的邊界
    for i in range(Nr):
        T[i, 0] = 0      # theta = 0
        T[i, Nth-1] = 0   # theta = pi/3

    # Gauss-Seidel 迭代求解
    for iteration in range(max_iter):
        max_err = 0.0

        for i in range(1, Nr - 1):
            for j in range(1, Nth - 1):
                ri = r[i]

                # 計算各項係數
                term_r = (T[i+1, j] + T[i-1, j]) / (dr * dr)
                term_rr = (T[i+1, j] - T[i-1, j]) / (2 * dr)
                term_th = (T[i, j+1] + T[i, j-1]) / (dtheta * dtheta)

                # 新的溫度值計算
                T_new = (
                    term_r +
                    (1.0 / ri) * term_rr +
                    (1.0 / (ri * ri)) * term_th
                ) / (2.0 / (dr * dr) + 2.0 / (ri * ri * dtheta * dtheta))

                # 計算誤差並更新
                err = abs(T_new - T[i, j])
                T[i, j] = T_new
                if err > max_err:
                    max_err = err

        # 檢查收斂條件
        if max_err < tol:
            print(f"Converged in {iteration + 1} iterations.")
            break


    print("r\\theta", end="")
    for j in range(Nth):
        theta_deg = theta[j] * 180 / math.pi  # 轉換為度數顯示
        print(f"{theta_deg:8.1f}°\t", end="")
    print()

    print("-" * 50)

    # 輸出每一行的結果
    for i in range(Nr):
        print(f"{r[i]:6.1f} |", end="")
        for j in range(Nth):
            print(f"{T[i, j]:8.3f}\t", end="")
        print()

    print("=" * 50)

if __name__ == "__main__":
    main()
```

圖 12、第三題程式碼

第四題：

x/t	t=0.0	t=0.1	t=0.2	t=0.3	t=0.4	t=0.5	t=0.6	t=0.7	t=0.8	t=0.9	t=1.0
x=0.0	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
x=0.1	0.80902	1.02382	1.03855	1.03855	1.02382	1.00000	0.97618	0.96145	0.96145	1.47618	2.19098
x=0.2	0.30902	0.84757	1.06237	1.06237	1.03855	1.00000	0.96145	0.93763	1.43763	2.15243	2.69098
x=0.3	-0.30902	0.34757	0.87139	1.06237	1.03855	1.00000	0.96145	1.43763	2.12861	2.65243	3.30902
x=0.4	-0.80902	-0.28519	0.34757	0.84757	1.02382	1.00000	1.47618	2.15243	2.65243	3.28519	3.80902
x=0.5	-1.00000	-0.80902	-0.30902	0.30902	0.80902	1.50000	2.19098	2.69098	3.30902	3.80902	4.00000
x=0.6	-0.80902	-1.02382	-0.84757	-0.34757	0.78519	2.00000	2.71481	3.34757	3.84757	4.02382	3.80902
x=0.7	-0.30902	-0.84757	-1.06237	-0.37139	0.84342	2.00000	3.15658	3.87139	4.06237	3.84757	3.30902
x=0.8	0.30902	-0.34757	-0.37139	0.12861	0.84342	2.00000	3.15658	3.87139	3.87139	3.34757	2.69098
x=0.9	0.80902	0.78519	0.84342	0.84342	1.28519	2.00000	2.71481	3.15658	3.15658	2.71481	2.19098
x=1.0	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000	2.00000

圖 13、第四題計算結果

```python
import math
# 常數
Nx = 11        # x: 0.0 to 1.0, step = 0.1
Nt = 21        # 模擬 t=0 ~ 2.0, step = 0.1
dx = 0.1
dt = 0.1
pi = math.pi
lambda2 = (dt * dt) / (dx * dx)  # λ^2 = 1

def initial_u(x):
    return math.cos(2 * pi * x)

def initial_ut(x):
    return 2 * pi * math.sin(2 * pi * x)

def print_horizontal_table(time_values, x_values, u_values_matrix):

    col_width = 12    # 計算每個欄位的寬度

    header = "x/t".ljust(col_width)     # 輸出表頭 - 時間值
    for t in time_values:
        header += f"| t={t:.1f}".ljust(col_width)
    print(header)

    separator = "-" * col_width     # 輸出分隔線
    for _ in range(len(time_values)):
        separator += "+" + "-" * (col_width - 1)
    print(separator)

    for i, x_val in enumerate(x_values):     # 輸出每一行（對應每個x值）
        row = f"x={x_val:.1f}".ljust(col_width)
        for t_idx in range(len(time_values)):
            row += f"| {u_values_matrix[t_idx][i]:.5f}".ljust(col_width)
        print(row)
```

```python
u_prev = [0.0] * Nx  # p^0
u_curr = [0.0] * Nx  # p^1
u_next = [0.0] * Nx  # p^{n+1}
x = [i * dx for i in range(Nx)]
# 初始狀態 t = 0
for i in range(Nx):
    u_prev[i] = initial_u(x[i])
# 邊界條件 t=0
u_prev[0] = 1.0
u_prev[Nx - 1] = 2.0

# 使用 Taylor 展開計算 t = dt 的值 (u_curr)
for i in range(1, Nx - 1):
    utt = (u_prev[i+1] - 2*u_prev[i] + u_prev[i-1]) / (dx*dx)
    u_curr[i] = u_prev[i] + dt * initial_ut(x[i]) + 0.5 * dt * dt * utt

# 邊界值
u_curr[0] = 1.0
u_curr[Nx - 1] = 2.0

# 儲存所有時間點的結果
all_time_values = [0.0, 0.1]  # 初始包含 t=0 和 t=0.1
all_u_values = [u_prev.copy(), u_curr.copy()]  # 儲存每個時間點的u值

for n in range(2, 11):
    current_time = n * dt
    all_time_values.append(current_time)

    # 計算內部節點
    u_next = [0.0] * Nx
    for i in range(1, Nx - 1):
        u_next[i] = (2 * u_curr[i] - u_prev[i] +
                     lambda2 * (u_curr[i+1] - 2 * u_curr[i] + u_curr[i-1]))

    u_next[0] = 1.0
    u_next[Nx - 1] = 2.0

    all_u_values.append(u_next.copy())
    u_prev = u_curr.copy()
    u_curr = u_next.copy()

print_horizontal_table(all_time_values, x, all_u_values)
print("=" * 130)
```

圖 14、第四題程式碼