

1.

```
main.py x +
main.py > ...
1 import numpy as np
2 import math
3 from scipy.interpolate import lagrange
4
5 def lagrange_interpolation(x_values, y_values, x_target):
6     """
7     使用 SciPy 的 lagrange 函式做拉格朗日插值。
8     回傳在 x_target 的近似值。
9     """
10    poly = lagrange(x_values, y_values)
11    return poly(x_target)
12
13 def error_bound(x_values, x_target):
14     """
15     使用插值誤差公式的上界：
16      $E_n(x) \leq \frac{\max |f^{(n+1)}(\xi)|}{(n+1)!} \cdot \prod |x - x_i|$ 
17     理論上  $\max |f^{(n+1)}(\xi)|$  是在包含所有插值點與目標點的區間內  $f^{(n+1)}(x)$ 
18     的最大值。
19     但對於  $f(x)=\cos(x)$  而言， $\cos$  或  $\sin$  的絕對值最大均不超過 1。
20     故直接取 1 作為保守估計。
21     """
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
>_ Console x +
python3 mai...
實際 cos(0.750) = 0.7316889
1 次插值：
  近似值 = 0.7320771
  誤差上界 = 4.4200000e-04
  真實誤差(參考) = 3.8827398e-04
2 次插值：
  近似值 = 0.7317163
  誤差上界 = 2.6520000e-06
  真實誤差(參考) = 2.7457657e-05
3 次插值：
  近似值 = 0.7317040
  誤差上界 = 3.5139000e-08
  真實誤差(參考) = 1.5086811e-05
注意：給定的點數不足，無法進行四次插值。
```

2.

```
main.py x +
main.py > ...
1 from scipy.interpolate import InterpolatedUnivariateSpline
2 from scipy.optimize import fsolve
3
4 # 數據點 (x -> y = e^(-x))
5 x_vals = np.array([0.3, 0.4, 0.5, 0.6]) # x 正數
6 y_vals = np.exp(-x_vals) # y = e^(-x)
7
8 # 使用插值建立反函數
9 inverse_interp = InterpolatedUnivariateSpline(y_vals, x_vals, kind='linear', fill_value="extrapolate")
10
11 # 解決 x = e^(-x) <=> inverse_interp(x) = x
12 def equation(x):
13     return inverse_interp(x) - x
14
15 # 使用 fsolve 求解，初始猜測值設為 0.5
16 solution = fsolve(equation, 0.5)[0]
17
18 # 輸出結果
19 print(f"使用逆插值法找到的解: x = {solution:.6f}")
20
21
```

```
>_ Console x +
python3 mai...
使用逆插值法找到的解: x ≈ 0.567545
```

3.

```
main.py x +
main.py > ...
1 import numpy as np
2 from scipy.interpolate import CubicHermiteSpline
3
4 # 給定的數據點
5 T = np.array([0, 3, 5, 8, 13]) # 時間 (秒)
6 D = np.array([0, 200, 375, 620, 990]) # 距離 (英尺)
7 V = np.array([75, 77, 80, 74, 72]) # 速度 (英尺/秒)
8
9 # 建立 Hermite 插值
10 hermite_interp = CubicHermiteSpline(T, D, V)
11
12 # (a) 預測 t = 10 的位置和速度
13 t_target = 10
14 D_10 = hermite_interp(t_target)
15 V_10 = hermite_interp.derivative()(t_target)
16
17 # (b) 確認車輛是否超過 55 mi/h (1 mi = 5280 ft, 55 mi/h = 55 * 5280 / 3600 ft/s)
18 speed_limit = 55 * 5280 / 3600 # 80.67 ft/s
19
20
```

```
>_ Console x +
python3 mai...
(a) 當 t = 10 秒時，位置 D(10) ≈ 768.96 英尺，速度 V(10) ≈ 74.64 英尺/秒
(b) 車輛是否超速？是
  最早超速時間：3.15 秒
(c) 預測最大速度：92.04 英尺/秒
```