2 พื้นฐานการหาค่าเหมาะที่สุด

ในบทนี้จะกล่าวถึงวิธีการพื้นฐานในการหาค่าเหมาะที่สุด โดยเริ่มจากแนวคิดเบื้องต้นสำหรับปัญหาที่ไม่มีเงื่อนไขบังคับ เป้าหมายคือลดค่าของฟังก์ชันวัตถุประสงค์จนกระทั่งได้ค่าต่ำที่สุด โดยอาจจะเป็นค่าแบบวงกว้าง ค่าเฉพาะที่ หรือไม่มีขอบเขต (เข้าสู่อนันต์ด้านลบ) ขึ้นกับว่าฟังก์ชันเป็นแบบใด ความซับซ้อนของอัลกอริทึมจะเพิ่มขึ้นตามลำดับเมื่อเพิ่มเงื่อนไขบังคับแบบสมการและอสมการ การนำเสนอในส่วนนี้จะยึดแนวทางตามเนื้อหาในรายวิชา [1] เพียงแต่เปลี่ยนแปลงรายละเอียดในตัวอย่าง และใช้ภาษาไพทอนแทนจูเลีย

> ในกรณีปัญหาเหมาะที่สุดคือการหาค่าค่าต่ำสุดของฟังก์ชัน เรานิยมเรียกฟังก์ชันวัตถุประสงค์ว่าฟังก์ชันมูลค่า (cost function)

2.1 อัลกอริทึมสำหรับหาคำตอบเหมาะที่สุด

วิธีการหาคำตอบสำหรับปัญหาการหาค่าเหมาะที่สุดสามารถทำได้หลากหลาย และอาจเลือกตัวหาคำตอบที่เหมาะสมตามประเภทของโจทย์ปัญหา แม้ในปัจจุบันยังมีผู้คิดค้นวิธีการใหม่ขึ้นมาอยู่ตลอด กล่าวได้ว่าเป็นงานวิจัยเปิด แนวทางในบทนี้เพียงนำเสนอวิธีการพื้นฐานที่ช่วยให้เข้าใจหลักการของการหาค่าเหมาะที่สุดเท่านั้น

สมมุติว่าเวลานี้เราไม่มีความรู้เกี่ยวกับการหาค่าเหมาะที่สุดเลย โจทย์กำหนดให้ฟังก์ชันมูลค่า $y=f(x)$ มาให้และถามว่าค่า x ที่ทำให้ y มีค่าต่ำสุดคือเท่าใด วิธีการแบบตรงไปตรงมาที่คิดได้คือกำหนดค่าตัวอย่าง x เป็นเวกเตอร์ในช่วงหนึ่งแล้วแทนค่าลงในฟังก์ชัน อาจพล็อตเป็นกราฟหรือเขียนโปรแกรมหาค่าต่ำสุดในเวกเตอร์เอาต์พุต วิธีการนี้มีข้อด้อยที่ทำให้ไม่สามารถใช้งานได้ดีในทางปฏิบัติ เช่น

- จะแน่ใจอย่างไรว่าย่านที่ทดสอบกว้างเพียงพอ และจำนวนจุดตัวอย่างละเอียดเพียงพอ เราอาจได้ค่าที่คลาดเคลื่อนจากคำตอบจริง
- ในกรณีตัวแปรตัดสินใจมีจำนวนมากและฟังก์ชันมีความซับซ้อน การคำนวณจะใช้เวลามากทำให้ไม่สามารถทำงานได้แบบเรียลไทม์

- ในบางกรณีฟังก์ชันอาจเป็นการประมาณค่าที่ไม่อยู่ในรูปปิด หรือมีการเปลี่ยนแปลงได้ระหว่างการทำงาน

ดังนั้นเราต้องการอัลกอริทึมรูปแบบอื่นที่มิใช่การแทนค่าฟังก์ชัน สามารถประมวลผลได้เร็วเพียงพอสำหรับควบคุมระบบที่ตอบสนองเร็วเช่นหุ่นยนต์ ประยุกต์ใช้กับงานได้หลากหลาย เช่นการเรียนรู้ของเครื่องที่อาจมีตัวแปรเป็นหลักล้านตัว

2.1.1 เงื่อนไขจำเป็นสำหรับค่าต่ำสุด

พื้นฐานการหาค่าต่ำสุดของฟังก์ชันโดยอาศัยหลักการของแคลคูลัสได้มีการประยุกต์ใช้ในหลายสาขาวิชา เช่น วิศวกรรมศาสตร์ วิทยาศาสตร์ เศรษฐศาสตร์ โดยพิจารณาจากความชันหรืออนุพันธ์ของฟังก์ชัน

ตัวอย่าง 2.1 พิจารณาฟังก์ชัน

$$f(x) = x^4 + x^3 - 2x^2 - 0.5x \tag{2.1}$$

เขียนโค้ดไพทอนเพื่อสร้างฟังก์ชัน (2.1)

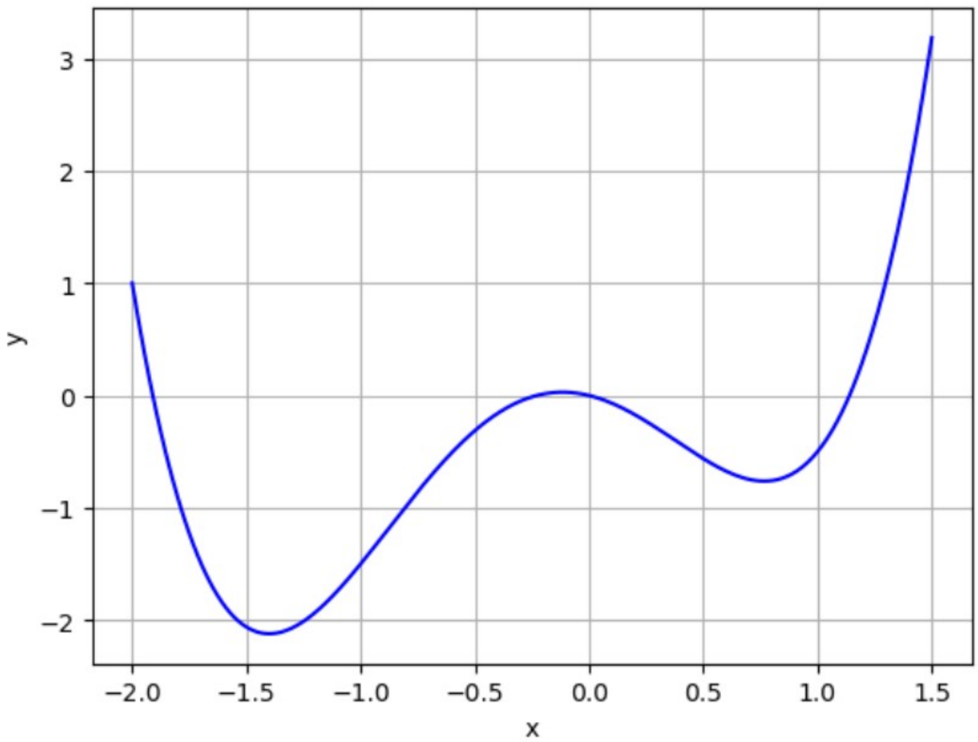
```python
def f(x):
    return x**4 + x**3 -2*x**2 - 0.5*x
```

พล็อตค่าในช่วง [-2, 1.5] ได้ดังรูปที่ 2.1

```python
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-2, 1.5, 1000)
y = f(x)
plt.plot(x,y,'b-')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```

จากกราฟเมื่ออ่านค่าโดยสายตา มีจุดต่ำสุดที่ค่า x ประมาณ -1.4 ซึ่ง ณ จุดนี้ ความชันของกราฟเท่ากับศูนย์ ดังนั้นเงื่อนไขจำเป็นในกรณีทั่วไปสำหรับค่าต่ำสุดคือค่าอนุพันธ์

$$\frac{\partial f(x)}{\partial x} \tag{2.2}$$



รูปที่ 2.1 กราฟของฟังก์ชัน (2.1) ในช่วง [-2, 1.5]

เหตุผลที่ (2.2) เป็นเพียงเงื่อนไขจำเป็นแต่ยังไม่เพียงพอพิจารณาได้จากกราฟในรูปที่ 2.1 จะเห็นว่ามี 3 ตำแหน่งที่ (2.2) เป็นจริง แต่มีเพียงตำแหน่งเดียวที่เป็นค่าต่ำสุดวงกว้าง ส่วนที่ค่า x ประมาณ 0.8 คือค่าต่ำสุดเฉพาะที่ และ x ประมาณ -0.1 เป็นค่ายอดของเส้นโค้งบริเวณส่วนกลางของกราฟ

จากหลักการของแคลคูลัส เราต้องการเงื่อนไขอีกข้อหนึ่งสำหรับค่าต่ำสุด (ไม่จำเป็นต้องเป็นแบบวงกว้าง) คืออนุพันธ์อันดับสองต้องมีค่าเป็นบวก ซึ่งจะขยายหลักการนี้ให้ครอบคลุมกรณีทั่วไป

2.1.2 การหาค่ารากโดยวิธีนิวตัน

ในการหาจุดที่ค่าความชันเป็นศูนย์ สามารถประยุกต์ใช้ *วิธีนิวตัน (Newton's method)* ที่ใช้ในการวิเคราะห์เชิงเลขสำหรับหาค่ารากของฟังก์ชันจำนวนจริง ซึ่งก็คือจุดที่ฟังก์ชันมีค่าเป็นศูนย์ นิยามเชิงคณิตศาสตร์คือ สำหรับฟังก์ชัน $f(x)$ หาค่าของ x^* ที่ทำให้ $f(x^*) = 0$

ในบริบทของระบบควบคุมที่เป็นเนื้อหาหลักของหนังสือ อาจเปรียบเทียบได้กับจุดสมดุล (equilibrium) ของพลวัตในระบบเวลาต่อเนื่อง หรือหากพิจารณาในระบบเวลาวิยุต (discrete-time) จุดสมดุลที่เสถียรเรียกว่า *จุดตรึง (fixed point)* ที่ทำให้ $f(x^*) = x^*$ สังเกตว่าทั้งสอง

รูปแบบมีความสัมพันธ์กัน เพราะเราสามารถแปลงปัญหาจุดตรึงเป็นการหาค่ารากได้โดยเขียนอยู่ในรูป $f(x^*) - x^* = 0$

วิธีการของนิวตันเริ่มต้นจากการประมาณค่าเชิงเส้นฟังก์ชัน $f(x)$

$$f(x + \Delta x) \approx f(x) + \frac{\partial f(x)}{\partial x}\Delta x \tag{2.3}$$

ตั้งให้การประมาณค่านี้เท่ากับศูนย์

$$f(x) + \frac{\partial f(x)}{\partial x}\Delta x = 0 \tag{2.4}$$

หาคำตอบ Δx ได้ดังนี้

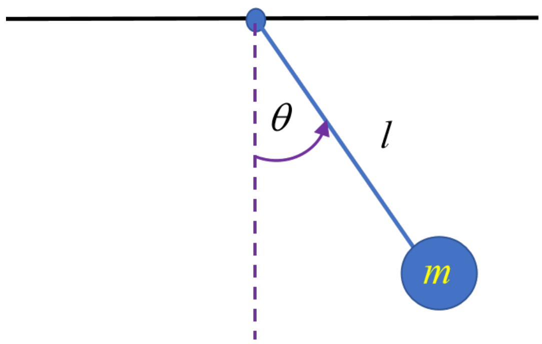
$$\Delta x = -\left(\frac{\partial f(x)}{\partial x}\right)^{-1} f(x) \tag{2.5}$$

สมการ (2.5) คือหัวใจสำคัญของอัลกอริทึมหาค่ารากโดยวิธีนิวตัน มีโครงสร้างดังนี้

- คาดเดาค่าเริ่มต้น x
- คำนวณขั้นการเปลี่ยนค่า Δx จาก (2.5)
- ปรับค่า $x \leftarrow x + \Delta x$
- วนซ้ำจนลู่เข้า คือค่า $f(x)$ เข้าสู่ศูนย์โดยมีค่าผิดพลาดอยู่ในพิกัดที่ต้องการ

ตัวอย่าง 2.2 พิจารณาพลวัตของลูกตุ้มในรูปที่ 2.2 เมื่อไม่มีอินพุต สามารถบรรยายโดยสมการการเคลื่อนที่ดังนี้

$$ml^2\ddot{\theta} + mglsin(\theta) = 0 \tag{2.6}$$



รูปที่ 2.2 ลูกตุ้มสำหรับตัวอย่าง 2.2

เมื่อกำหนดสถานะ $x = [\theta \;\; \dot{\theta}]^T$ จะเขียนอยู่ในรูปเมทริกซ์ได้เป็น

$$\dot{x} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \frac{-g}{l} sin(\theta) \end{bmatrix} \tag{2.7}$$

หากต้องการจำลองพลวัตของลูกตุ้มบนคอมพิวเตอร์ จะต้องแปลงระบบให้อยู่ในรูปเวลาวิยุตซึ่งทำได้หลายวิธี วิธีการหนึ่งที่จะแสดงในตัวอย่างนี้โดยการหาค่ารากหรือวิธีจุดตรึงเรียกว่า *ออยเลอร์ย้อนหลัง (backward Euler)* เขียนได้ดังนี้

$$x_{k+1} = x_k + hf(x_{k+1}) \tag{2.8}$$

โดย h คือค่าขั้นเวลา (time step) สังเกตว่าค่าของสถานะ x_{k+1} ใน (2.5) ขึ้นกับ $f(x_{k+1})$ ซึ่งไม่สามารถคำนวณได้โดยตรง แต่เมื่อจัดรูปใหม่เป็น

$$x_k + hf(x_{k+1}) - x_{k+1} = 0 \tag{2.9}$$

จะกลายเป็นโจทย์ที่สามารถใช้วิธีหาค่ารากหรือวิธีจุดตรึงได้

บรรณานุกรม

1. Z. Manchester et.al. 16-745 Optimal Control & Reinforcement Learning, Course materials, Carnegie Mellon University. 2024.