บทที่ 2

อัลกอริทึมการอบอ่อนจำลอง
Simulated Annealing Algorithm

อัลกอริทึมการอบอ่อนจำลอง (simulated annealing หรือ SA) เป็นเทคนิคการค้นหาคำตอบแบบสุ่ม ซึ่งเลียนแบบกรรมวิธีการควบคุมอุณหภูมิของการอบโลหะให้ร้อนแล้วปล่อยให้เย็นลงช้าๆ เพื่อให้เกิดการเปลี่ยนแปลงทางโครงสร้างผลึกที่มีพลังงานภายในน้อยที่สุด คุณลักษณะดังกล่าวจะทำให้โลหะมีคุณสมบัติทางกายภาพที่แข็งแกร่งและทนทาน ในทางตรงกันข้าม ถ้าไม่ทำการควบคุมการทำให้เย็นของโลหะร้อนแล้ว โครงสร้างที่ได้จะมีจุดตำหนิหรือบกพร่อง เป็นโลหะมีความแข็ง แต่เปราะ กรรมวิธีการควบคุมอุณหภูมิของการอบโลหะดังกล่าวเรียกว่าการอบอ่อน (annealing process) การอบอ่อนจำลองเป็นเทคนิคการค้นหาค่าเหมาะที่สุดสำหรับปัญหาเชิงผสมผสาน โดยมีการควบคุมอุณหภูมิของการอบอ่อนเป็นการปรับค่าพารามิเตอร์การค้นหาคำตอบของระบบนั่นเอง

การอบอ่อนจำลอง (Simulated Annealing หรือ SA) ได้รับการพัฒนาขึ้นในปี 1983 โดย Kirkpatrick และคณะ [Kirkpatrick et al., 1983]เพื่อใช้ในการแก้ปัญหาที่ไม่เป็นเชิงเส้น SA ทำให้ระบบได้รับคำตอบที่เหมาะที่สุดแบบวงกว้าง (global optimum) โดยการเลียนแบบธรรมชาติของกระบวนการอบอ่อนเหล็ก กระบวนการดังกล่าวเริ่มต้นจากการเผาเหล็กด้วยอุณหภูมิสูงจนเกินจุดหลอมเหลว ซึ่งทำให้เหล็กอยู่ในสภาวะของเหลวและมีโครงสร้างระดับโมเลกุลที่มีพลังงานสูง สามารถมีปริมาณการเคลื่อนที่ที่สูงได้ ซึ่งจุดนี้เปรียบได้กับการที่การค้นหาคำตอบ มีพลังงานมากพอที่จะสามารถก้าวหลุดพ้นจากคำตอบที่เหมาะที่สุดแบบวงแคบเฉพาะถิ่นได้ (local optimum) เมื่อทำการลดอุณหภูมิของระบบลง โมเลกุลโครงสร้างของเหล็กจะไม่สามารถเคลื่อนที่ได้เท่าเดิม และจะถูกจำกัดในช่วงคำตอบที่เหมาะที่สุดแบบวงแคบเฉพาะถิ่น เมื่อพิจารณาในระดับโครงสร้างผลึกของเหล็ก การลดอุณหภูมิลงอย่างช้าๆ ทำให้เหล็กมีโครงสร้างผลึกที่แข็งแกร่ง ในขณะเดียวกัน ถ้าอุณหภูมิลดลงอย่างไม่เหมาะสม โครงสร้างผลึกดังกล่าวจะมีการจัดตัวที่ไม่เหมาะสม ส่งผลให้เกิดข้อบกพร่องหรือมีตำหนิในโครงสร้างผลึก ซึ่งจะทำให้เหล็กที่ได้มีความแข็งแกร่งแต่เปราะ เหล็กที่ไม่มีข้อบกพร่องในโครงสร้างผลึกเลยเปรียบได้กับคำตอบที่เหมาะที่สุดแบบวงกว้าง กระบวนการอบอ่อนเหล็กดังกล่าวเปรียบเทียบได้กับการเขย่ากล่องที่มีพื้นผิวการค้นหาคำตอบ และมีลูกบอลกลิ้งไปมาบนพื้นผิวนั้นๆ เมื่อลูกบอลกลิ้งไปติดกับร่อง เราจำเป็นต้องเขย่ากล่องให้แรงขึ้นเพื่อให้ลูกบอลหลุดออกจากร่องให้ได้ ร่องดังกล่าวเปรียบได้กับคำตอบที่เหมาะที่สุดแบบวงแคบเฉพาะถิ่น ร่องที่ลึก ย่อมต้องการการเขย่าที่แรงกว่าในการที่จะทำให้ลูกบอลหลุดออกจากร่องได้ การเขย่าที่แรงคือการให้พลังงานกับลูกบอลที่มากขึ้น เทียบได้กับการให้อุณหภูมิที่สูงกับโมเลกุลของเหล็ก อุณหภูมิที่ลดลงคือการเขย่ากล่องด้วยแรงที่ลดลง ทำให้ลูกบอลเคลื่อนที่ได้ราบเรียบและเข้าสู่เป้าหมายได้ดีขึ้น อย่างไรก็ดี ถึงแม้ว่า SA จะสามารถให้คำตอบที่เหมาะที่สุดแบบวงกว้างได้ แต่เวลาในการค้นหาคำตอบอาจจะไม่เหมาะสมในการใช้งานจริงก็ได้ มีงาน

วิจัยมากมายที่เน้นการปรับปรุงประสิทธิภาพในการค้นหาคำตอบของ SA ให้ดีขึ้นเช่น FSA (Fast Simulated Annealing) หรือ ADA (Adaptive Simulated Annealing) ที่ซึ่งลู่เข้าสู่คำตอบได้เร็วขึ้นแบบเลขยกกำลัง

SA [Kirkpatrick et al., 1983][Rutenbar, 1989][Hajek, 1985] มีความแตกต่างไปจาก**การค้นหาคำตอบแบบวนรอบดีขึ้น** (iterative improvement search) ตรงที่มีการใส่ค่าสุ่มเข้าไปในการค้นหาคำตอบ การสุ่มค่าดังกล่าวทำให้ SA สามารถ "*กระโดด*" หรือหลุดออกจากคำตอบที่เหมาะที่สุดแบบวงแคบเฉพาะถิ่นได้ การกระโดดดังกล่าวคือการปรับอุณหภูมิในกระบวนการอบอ่อนข้างต้นนั่นเอง การปรับอุณหภูมิของ SA ยังสามารถพิจารณาได้อีกสองแง่มุม กล่าวคือที่อุณหภูมิสูง ระบบจะทำการค้นหาแบบหยาบๆ (การกระโดดไปมาอย่างแรง) ในขณะที่การค้นหาแบบละเอียดจะได้จากระบบที่มีอุณหภูมิต่ำๆ ด้วยลักษณะการทำงานแบบสุ่มดังกล่าว เราอาจเรียก SA ว่าเป็นอัลกอริทึมเชิงน่าจะเป็น (probabilistic algorithm)

2.1 อัลกอริทึมการอบอ่อนจำลอง Simulated Annealing Algorithm

อัลกอริทึมการอบอ่อนจำลองได้รับแนวคิดทางธรรมชาติจากการทำให้เหล็กที่กำลังร้อนเย็นตัวลง ในโครงสร้างของเหล็กที่กำลังร้อนนั้น ความน่าจะเป็นของกลุ่มอะตอม ณ ตำแหน่ง r_i จะมีค่าพลังงานเฉพาะค่าหนึ่งคือ $E(r_i)$ ที่อุณหภูมิ T ค่าความน่าจะเป็นดังกล่าวสามารถนิยามได้จากค่าความน่าจะเป็นแบบโบลต์ซมันน์ (Boltzmann probability) ดังนี้

$$Pr[E(r_i)] = e^{-E(r_i)/k_B T} \tag{2.1}$$

โดยที่ k_B คือค่าคงที่โบลต์ซมันน์ จะเห็นได้ว่าค่าความน่าจะเป็นข้างต้นมีค่าขึ้นอยู่กับ 2 ปัจจัย ได้แก่พลังงาน E ของกลุ่มอะตอมที่ตำแหน่ง r_i และอุณหภูมิ T ความน่าจะเป็นของกลุ่มอะตอมที่จะมีพลังงานสูงนั้นจะมีค่าน้อยกว่าที่กลุ่มอะตอมจะมีพลังงานต่ำ ในทางตรงกันข้าม เมื่อพิจารณาตัวแปรอุณหภูมิ T จะเห็นว่าอุณหภูมิที่สูงจะให้ค่าความน่าจะเป็นของกลุ่มอะตอมที่จะมีพลังงานที่สูงด้วย ในขณะที่อุณหภูมิต่ำกว่า ค่าความน่าจะเป็นที่กลุ่มอะตอมจะรักษาระดับพลังงานเอาไว้จะลดลงด้วย

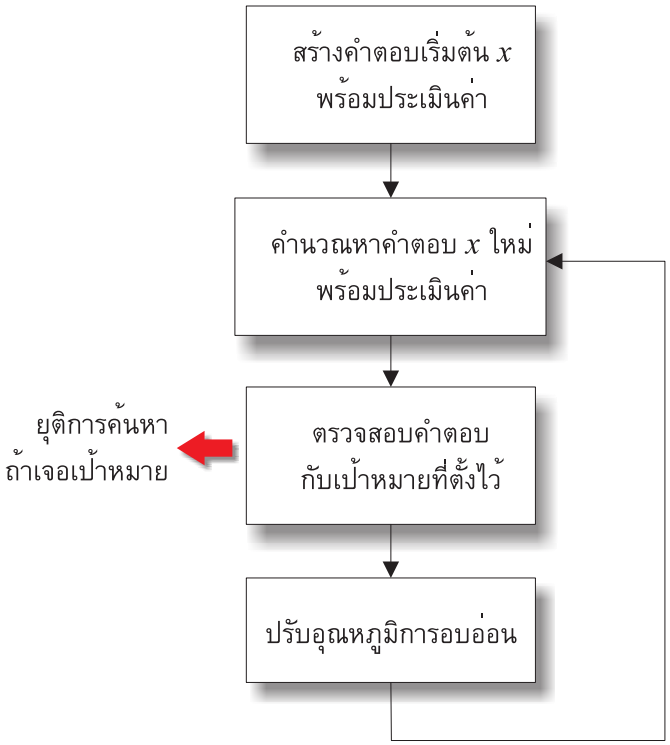
ปรากฏการณ์จากความสัมพันธ์ข้างต้นนำไปสู่การออกแบบอัลกอริทึมสำหรับการอบอ่อนจำลอง ซึ่งเป็นจุดเด่นที่ทำให้ SA ได้รับความสนใจมากมาย จุดเด่นที่สำคัญอย่างหนึ่งคือ SA มีโครงสร้างหรือหลักการทำงานที่ไม่มีความซับซ้อนแต่อย่างใด รูปที่ 2.1 แสดงแผนผังการทำงานของ SA ซึ่งมีองค์ประกอบสำคัญอยู่ 4 อย่างดังนี้ (รายละเอียดดังกล่าวมีที่มาจากอัลกอริทึมเมโทรโพลิสหรือ Metropolis หนึ่งในเทคนิคมอนติคาร์โลหรือ Monte Carlo ซึ่งมีใช้ในกลศาสตร์สถิติ) [Ham and Kostanic, 2001]

- รูปแบบของคำตอบที่ต้องการค้นหา
- ตัวกำเนิดสุ่ม (random generator) หรือตัวปฏิบัติการค้นหาสำหรับทำการ "เดิน" หรือ "move" คำตอบของระบบไปยังคำตอบใหม่
- ฟังก์ชันวัตถุประสงค์ (objective function) ของระบบ
- ตารางจัดการการอบอ่อน (annealing schedule) ซึ่งเป็นส่วนที่ใช้ปรับค่าอุณหภูมิและกำหนดเวลาที่จะให้ระบบเกิดการเปลี่ยนแปลงอุณหภูมิ

รายละเอียดของอัลกอริทึมการอบอ่อนจำลองมีดังต่อไปนี้

▷ **อัลกอริทึมการอบอ่อนจำลอง – Simulated Annealing Algorithm**

1. กำหนดรอบการทำงานของอัลกอริทึมด้วยตัวชี้ k ใช้ค่าเริ่มต้นการทำงานของระบบเป็น $k = 0$



รูปที่ 2.1: อัลกอริทึมการอบอ่อนจำลอง

2. สุ่มค่าเริ่มต้นเวกเตอร์คำตอบ $\mathbf{x}_k$ ในปริภูมิการค้นหา ค่าเวกเตอร์คำตอบ $\mathbf{x}_k$ นี้สามารถพิจารณาว่าเป็นสถานะปัจจุบันของระบบก็ได้

3. กำหนดตารางจัดการการอบอ่อนสำหรับพารามิเตอร์อุณหภูมิ T และตั้งค่าเริ่มต้นของ T ไว้ที่ค่าสูงอย่างเพียงพอ (ในการหลอมละลายเหล็กได้)

4. คำนวณหาเวกเตอร์คำตอบใหม่ $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$ โดยที่ $\Delta\mathbf{x}$ เป็นการเปลี่ยนแปลงที่ถูกนำเสนอต่อระบบ

5. คำนวณหาการเปลี่ยนแปลงของฟังก์ชันวัตถุประสงค์ $\Delta f = f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)$ ถ้า Δf มีค่าเป็นลบ แสดงว่าสถานะใหม่ $\mathbf{x}_{k+1}$ มีค่าวัตถุประสงค์ที่ลดลง ซึ่งเป็นกรณีที่ต้องการถ้าปัญหาที่พิจารณาเป็นปัญหาการหาค่าน้อยที่สุด (minimization) ในทางตรงกันข้ามถ้า Δf มีค่าเป็นบวก แสดงว่าสถานะใหม่ $\mathbf{x}_{k+1}$ มีค่ามากขึ้น ซึ่งตรงกับปัญหาการหาค่ามากที่สุด (maximization)

6. ทำการคำนวณหาว่าควรจะใช้ $\mathbf{x}_{k+1}$ เป็นคำตอบหรือสถานะใหม่ของระบบหรือไม่ กล่าวคือใช้ความน่าจะเป็นในการตัดสินใจที่จะเปลี่ยนสถานะตามเงื่อนไขต่อไปนี้

$$Pr(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}) = \begin{cases} 1 & \text{ถ้า } \Delta f < 0 \\ e^{-\Delta f/T} & \text{ถ้า } \Delta f \geq 0 \end{cases} \tag{2.2}$$

โดยที่ T แทนอุณหภูมิของระบบ พิจารณาในกรณี $\Delta f < 0$ จะได้ว่าค่าวัตถุประสงค์เป็นไปตามที่ต้องการ (minimization) ดังนั้นค่าความน่าจะเป็นในการเปลี่ยนสถานะในกรณีจึงเท่ากับ 1 หรือมีการเปลี่ยนสถานะใหม่อย่างแน่นอน ในกรณีที่ $\Delta f \geq 0$ ค่าความน่าจะเป็นในการเปลี่ยนสถานะของระบบจะต้องถูกสุ่มคำนวณขึ้น กล่าวคือ ค่าตัวเลขสุ่มที่มีการกระจายแบบสม่ำเสมอ (uniform distribution) η จะถูกเลือกจากในช่วง $[0, 1]$ ถ้า $Pr(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}) > \eta$ แล้ว เวกเตอร์คำตอบ $\mathbf{x}_{k+1}$ จะถูกใช้เป็นคำตอบหรือสถานะใหม่ของระบบ มิฉะนั้นแล้ว ให้คงคำตอบของระบบไว้ที่ $\mathbf{x}_k$ เหมือนเดิม จะเห็นได้ว่าถึงแม้ค่าวัตถุประสงค์จะไม่

เป็นตามที่ต้องการ (ในกรณีนี้คือค่าสถานะใหม่มีค่ามากกว่าหรือเท่ากับสถานะเดิม) ระบบยังคงมีโอกาสที่จะเปลี่ยนสถานะได้ ด้วยค่าความน่าจะเป็นตามความสัมพันธ์ที่ (2.2) ซึ่งค่าความน่าจะเป็นจะขึ้นอยู่กับค่าความแตกต่างของค่าวัตถุประสงค์ระหว่างสถานะใหม่และสถานะเก่า และค่าอุณหภูมิของระบบ ค่าความน่าจะเป็นดังกล่าวสอดคล้องกับแนวคิดในความสัมพันธ์ที่ (2.6)

7. ขั้นตอนข้างต้นจะถูกทำซ้ำ (เพิ่มค่าตัวชี้ k) จนกระทั่งระบบเข้าสู่จุดสมดุล ที่ซึ่งรู้ได้จากค่าการเปลี่ยนแปลงสถานะของระบบน้อยกว่าค่าที่กำหนดไว้ สภาวะดังกล่าวนี้จะเกิดขึ้นเมื่อระบบมีคำตอบหรือสถานะเข้าใกล้ค่าเหมาะที่สุดแบบวงแคบเฉพาะถิ่น โดยปกติแล้ว จำนวนครั้งของการทำซ้ำจะถูกกำหนดไว้ก่อนล่วงหน้า
8. ทำการปรับค่าอุณหภูมิ T ตามตารางจัดการการอบอ่อน แล้วเริ่มทำขั้นตอนทำซ้ำทั้งหมดใหม่อีกครั้ง กระบวนการทั้งหมดสามารถยุติได้เมื่อค่าอุณหภูมิ T มีค่าเป็นศูนย์ (ระบบถูกแช่แข็ง) ◁

รายละเอียดพารามิเตอร์ต่างๆ ของ SA มีดังต่อไปนี้

- **ตารางจัดการการอบอ่อน** - Annealing Schedule
  ในความเป็นจริงแล้ว ประสิทธิภาพของ SA จะขึ้นอยู่กับการเลือกตารางจัดการการอบอ่อน (สำหรับพารามิเตอร์อุณหภูมิ T) ตารางจัดการการอบอ่อนจะเป็นตัวกำหนดการเคลื่อนที่หรือ "move" ของคำตอบของระบบในระหว่างการค้นหา หลักในการเลือกตารางจัดการการอบอ่อนที่เหมาะสมคือ "*ค่าอุณหภูมิเริ่มต้นของระบบควรจะสูงพอที่จะหลอมละลายระบบอย่างสมบูรณ์ และในระหว่างขั้นตอนการค้นหาควรจะมีค่าลดลงไปจนถึงจุดเยือกแข็ง*" การเลือกตารางจัดการการอบอ่อนจึงค่อนข้างเป็นงานศิลปะ ถ้าระบบถูกลดอุณหภูมิลงเร็วเกินไป การลู่เข้าสู่คำตอบอาจจะเร็วเกินไปด้วย ทำให้ระบบลู่เข้าสู่คำตอบที่เหมาะที่สุดแบบวงแคบเฉพาะถิ่น ในทางตรงกันข้าม ถ้าระบบมีอุณหภูมิลดลงช้าเกินไป จะทำให้เสียเวลาในการคำนวณมากเกินไปด้วยเช่นกัน การเลือกตารางจัดการการอบอ่อนหลายๆ วิธีได้ถูกนำเสนอ ไม่ว่าจะเป็นวิธีการใช้ลูกโซ่มาร์คอฟ (Markov chain) ขนาดจำกัดในการลดอุณหภูมิ [Isaacson and Madsen, 1976] หรือวิธีการปรับอุณหภูมิจาก [Geman and Geman, 1984] ดังต่อไปนี้ ตารางจัดการการอบอ่อนคือการกำหนดค่า $T(k)$ ซึ่งเป็นค่าอุณหภูมิ ณ รอบการทำงานที่ k ของระบบ

$$T(k) = \frac{T(0)}{\log(1+k)} \quad k = 1, 2, \ldots \tag{2.3}$$

  โดยที่ k คือจำนวนรอบที่กำลังทำซ้ำ $T(k)$ เป็นอุณหภูมิลำดับที่ k และ $T(0)$ เป็นค่าอุณหภูมิเริ่มต้น (ควรมีค่าบวกและมากเพียงพอ) อัลกอริทึม SA รับประกันว่าจะลู่เข้าสู่คำตอบที่เหมาะที่สุดแบบวงกว้างด้วยฟังก์ชันวัตถุประสงค์ $f(\mathbf{x})$ เมื่อ $k \to \infty$ หรือในรูปแบบการลดอุณหภูมิอย่างง่ายๆ เช่น

$$T(k) = \alpha T(k-1) \tag{2.4}$$

  โดยที่ α เป็นค่าคงที่ที่มีค่าใกล้เคียงและน้อยกว่า 1 เรียกว่า*การลดอุณหภูมิแบบเลขชี้กำลัง* (exponential cooling scheme หรือ ECS) นำเสนอครั้งแรกด้วยการใช้ค่า $\alpha = 0.95$ ส่วน*การลดอุณหภูมิแบบเชิงเส้น* (linear cooling scheme หรือ LCS) กำหนดการลดค่าอุณหภูมิดังนี้

$$T(k) = T(k-1) - T' \tag{2.5}$$

  โดยที่ค่า T' เป็นค่าอุณหภูมิที่ลดลงไปในช่วงจำนวนการวนรอบที่กำหนด ข้อควรระวังคือค่าอุณหภูมิที่สามารถติดลบได้ในกรณีของ LCS

- **ค่าอุณหภูมิเริ่มต้น** - Initial Temperature
  ค่าอุณหภูมิเริ่มต้น $T(0)$ ที่เหมาะสมจะทำให้ค่าเฉลี่ยความน่าจะเป็นในการยอมรับคำตอบหรือสถานะใหม่

ของระบบสูงถึง 0.8 กล่าวคือมีโอกาส 80% ที่การเปลี่ยนแปลงของคำตอบหรือสถานะของระบบจะถูกยอมรับ ค่าของ $T(0)$ จะขึ้นอยู่กับสัดส่วนของฟังก์ชันวัตถุประสงค์ f ดังนั้นค่าอุณหภูมิเริ่มต้นดังกล่าวจึงมีค่าเฉพาะของแต่ละปัญหา

- **ค่าอุณหภูมิสุดท้าย** - Final Temperature
  ในอัลกอริทึม SA ที่ไม่ซับซ้อน ค่าอุณหภูมิสุดท้ายของระบบสามารถหาได้จากจำนวนรอบของการทำซ้ำ จำนวนคำตอบใหม่ที่ได้ทั้งหมดหรือจำนวนค่าอุณหภูมิที่ใช้ไป ในขณะเดียวกัน เราสามารถยุติขั้นตอนของ SA ได้เมื่อระบบไม่มีการเปลี่ยนแปลงของคำตอบหรือสถานะใหม่

2.2 ตารางจัดการการอบอ่อน

สำหรับ SA แล้ว สิ่งที่เป็นหัวใจสำคัญของอัลกอริทึมก็คือตารางจัดการการอบอ่อน เราจะเห็นได้ว่าถ้ามีการจัดการอุณหภูมิของระบบที่ถูกต้องเหมาะสม ระบบจะสามารถเข้าสู่สภาวะสมดุลได้ กล่าวคือคำตอบที่ได้จากระบบจะเป็นคำตอบที่มีค่าเหมาะที่สุด ในการวิเคราะห์ศึกษา SA จึงได้มีผู้พัฒนาตารางจัดการการอบอ่อนในรูปแบบต่างๆ เพื่อให้ได้กระบวนการอบอ่อนที่สอดคล้องกับวัตถุประสงค์ของการค้นหาคำตอบ นั่นก็คือการหาคำตอบจะต้องรวดเร็วและคำตอบที่ได้จะต้องมีความถูกต้อง ในหัวข้อนี้จะได้กล่าวถึงตารางจัดการการอบอ่อนในรูปแบบต่างๆ ซึ่งรายละเอียดดังกล่าวสามารถนำไปต่อยอดสู่อัลกอริทึม SA แบบต่างๆ ต่อไปได้

2.2.1 การอบอ่อนแบบมาตรฐาน

ใน SA มาตรฐาน [Kirkpatrick et al., 1983] (หรือที่ใช้กันทั่วไป) เดิมทีเป็นเพียงอัลกอริทึมที่เรียกว่า "Metropolis Monte Carlo integration" [Metropolis et al., 1953] ที่รวมเอาตารางจัดการอุณหภูมิไว้ด้วย เพื่อเพิ่มประสิทธิภาพในการค้นหาคำตอบ ต่อมาได้มีการพิสูจน์ไว้ว่าการปรับอุณหภูมิเป็นไปตามสมการที่ 2.3 กล่าวคืออุณหภูมิมีขอบล่างสุดจำกัดไว้ด้วย $1/log(k+1)$ (เลข 1 มีไว้เพื่อไม่ให้ค่าเกิดค่า $\log$ ของศูนย์) โดยที่ k คือค่าเวลาอ้างอิงของตารางจัดการการอบอ่อน หลังจากการนำเสนอดังกล่าว ได้มีผู้พัฒนาอัลกอริทึมต่างๆ เพิ่มเติมขึ้น จนภายหลังเราถึงได้รู้จักงานข้างต้นในชื่อของการอบอ่อนจำลองหรือ simulated annealing

2.2.2 การอบอ่อนแบบโบลต์ซมันน์ Boltzmann Annealing

การอบอ่อนจำลองที่เป็นที่ยอมรับคือการอบอ่อนแบบโบลต์ซมันน์ (Boltzmann Annealing หรือ BA [Szu and Hartley, 1987]) การกระจายของความน่าจะเป็นในการเปลี่ยนสถานะของระบบจะเป็นแบบโบลต์ซมันน์ โดยพิจารณาได้จากสมการที่ 2.6 กำหนดให้ $E(\mathbf{x}_k)$ เป็นพลังงานของระบบขณะเวลา k (หรือของสถานะปัจจุบัน $\mathbf{x}_k$) $E(\mathbf{x}_{k+1})$ เป็นพลังงานของระบบขณะเวลา $k+1$ (หรือสถานะใหม่ของระบบ $\mathbf{x}_{k+1}$) และ ΔE เป็นความแตกต่างของพลังงานของระบบในสถานะปัจจุบันและสถานะก่อนหน้า ($\Delta E = E(\mathbf{x}_{k+1}) - E(\mathbf{x}_k)$) จะได้ว่า

$$\begin{aligned} Pr[\Delta E] &= \frac{e^{(-E(\mathbf{x}_{k+1})/T)}}{e^{(-E(\mathbf{x}_{k+1})/T)} + e^{(-E(\mathbf{x}_k)/T)}} \\ &= \frac{1}{1+e^{(-\Delta E/T)}} \\ &\approx e^{(-\Delta E/T)} \end{aligned} \tag{2.6}$$

พิจารณาการอบอ่อนจำลองดังกล่าวประกอบไปด้วยความสัมพันธ์ของปริมาณ 3 อย่างดังต่อไปนี้

- $Pr[\Delta E]$ - ความน่าจะเป็นในการยอมรับสถานะใหม่ของระบบ โดยพิจารณาสถานะก่อนหน้านี้ของระบบด้วย
- $T(k)$ - ตารางเวลา k ในการอบอ่อนของอุณหภูมิ T

- $g_T(r)$ - ความหนาแน่นของความน่าจะเป็น (probability density) ของปริภูมิสถานะของพารามิเตอร์ของระบบ $r = \{r_i, i = 1, \ldots, M\}$ ณ อุณหภูมิ T โดยที่ M คือจำนวนของพารามิเตอร์ในระบบ (นั่นคือขนาดหรือ dimension ของปริภูมิสถานะของพารามิเตอร์)

ในอัลกอริทึมของ SA สิ่งสำคัญอย่างหนึ่งก็คือการเลือกช่วงของพารามิเตอร์ที่จะทำการค้นหา ดังนั้นจึงได้มีการนำเสนอเทคนิคต่างๆ ที่จะทำให้สามารถเลือกช่วงการค้นหาพารามิเตอร์ที่แคบลงได้ ผลลัพธ์ที่ได้จะปรากฏในรูปของตารางจัดการการอบอ่อนแบบต่างๆ ยกตัวอย่างเช่นในสมการที่ 2.3 สำหรับ BA ที่ซึ่งสามารถพิจารณาฟังก์ชันพลังงานในรูปการกระจายแบบโบลต์ซมันน์ (Boltzmann distribution) ได้นั้น การจัดการอุณหภูมิของการอบอ่อนจะอยู่ในรูปลอการิทึม ตัวอย่างเช่น

$$T(k) = T(0)\frac{\ln k_0}{\ln k} \tag{2.7}$$

โดยที่ T เป็น "อุณหภูมิ" k เป็นตัวชี้ "เวลา" ในการอบอ่อนและ k_0 เป็นค่าตัวชี้เริ่มต้น สำหรับค่า k ที่มากๆ เราสามารถเขียนในรูปต่อไปนี้ได้

$$T(k+1) = T(k) - T(0)\frac{\ln k_0}{k(\ln k)^2} \tag{2.8}$$

หรือในบางครั้งก็มีการใช้ในรูปของเลขชี้กำลังแทนรูปลอการิทึม ยกตัวอย่างเช่น

$$\begin{aligned} T(k+1) &= cT(k), 0 < c < 1 \\ T(k) &= T(0)e^{(c-1)k} \end{aligned} \tag{2.9}$$

เราจะเห็นได้ว่าเงื่อนไขของการปรับอุณหภูมิข้างต้นเป็นสิ่งที่ทำให้อัลกอริทึม BA มีประสิทธิภาพที่แตกต่างกัน อัลกอริทึม SA แบบต่างๆ จึงได้ถูกนำเสนอเพื่อปรับปรุงประสิทธิภาพการค้นหาคำตอบดังกล่าว

2.2.3 การอบอ่อนแบบเร็ว Fast Annealing

ในงาน [Szu and Hartley, 1987] ได้พิจารณาอัลกอริทึม SA ในรูปการกระจายของพลังงานแบบโคชี (Cauchy distribution) และได้รับการบันทึกว่าการกระจายแบบโคชีมีข้อได้เปรียบเหนือการกระจายแบบโบลต์ซมันน์ นำไปสู่การปรับอุณหภูมิของระบบตามความสัมพันธ์ต่อไปนี้

$$T(k) = \frac{T(0)}{k} \tag{2.10}$$

อัลกอริทึม SA ที่มีตารางจัดการการอบอ่อนข้างต้นเรียกว่าเป็นการอบอ่อนแบบเร็ว (Fast Annealing หรือ FA) ซึ่งจะเห็นได้ว่ามีการปรับอุณหภูมิเร็วกว่า BA แบบเลขชี้กำลัง

2.2.4 การอบอ่อนจำลองเชิงปรับตัว Adaptive Simulated Annealing

การอบอ่อนจำลองเชิงปรับตัว (Adaptive Simulated Annealing หรือ ASA [Ingber, 1993]) เป็นอัลกอริทึมการค้นหาค่าเหมาะที่สุดที่ใช้หลักการสุ่มในปริภูมิของพารามิเตอร์ Ingber เองนั้นได้เผยแพร่อัลกอริทึมการอบอ่อนจำลองแบบความเร็วสูง (Very Fast Simulated Reannealing หรือ VFSR [Ingber, 1989]) ไว้ งานดังกล่าวได้มีการนำไปเขียนด้วยโปรแกรมคอมพิวเตอร์และมีผู้สนใจมากมายนำไปใช้งานได้อย่างมีประสิทธิภาพ ASA ที่ได้นำเสนอต่อจาก VFSR เองก็ได้รับความสนใจนำไปเขียนเป็นโปรแกรมคอมพิวเตอร์เพื่อใช้งานเช่นกัน เนื้อหาในส่วนนี้จะได้ทำการกล่าวถึงรายละเอียดของ ASA ซึ่งมีพื้นฐานมาจาก SA นั่นเอง อย่างไรก็ดี ได้มีผู้นำเอา SA มาทำการปรับปรุงในหลายรูปแบบ จุดประสงค์ก็คือเพื่อเพิ่มความสามารถในการค้นหาค่าเหมาะที่สุด ทั้งในทางด้านเวลา

การค้นหาและประสิทธิภาพของคำตอบที่ได้ ตัวอย่างเช่นการอบอ่อนแบบโบลต์ซมันน์ (Boltzmann Annealing หรือ BA [Szu and Hartley, 1987]) หรือการอบอ่อนแบบเร็ว (Fast Annealing หรือ FA [Szu and Hartley, 1987]) เป็นต้น

ASA พิจารณาพารามิเตอร์ r_k^i ของมิติที่ i (พารามิเตอร์มีจำนวนมิติทั้งหมดเท่ากับ D) ณ เวลาการอบอ่อน k ในช่วง

$$r_k^i \in [A_i, B_i] \tag{2.11}$$

จะได้ตารางการจัดการการอบอ่อน ณ อุณหภูมิ T_i สำหรับพารามิเตอร์ตัวที่ i ดังนี้

$$T_i(k) = T_{0i} e^{-c_i k^{1/D}} \tag{2.12}$$

จะเห็นได้ว่า ASA มีความแตกต่างจาก SA อื่นๆ ตรงที่พารามิเตอร์แต่ละตัวจะมีการตารางการปรับอุณหภูมิที่แตกต่างกัน เราสามารถควบคุม c_i ได้เช่น

$$\begin{aligned} T_{fi} &= T_{0i} e^{-m_i} \text{ โดยที่ } k_f = e^{n_i} \\ c_i &= m_i e^{-n_i/D} \end{aligned} \tag{2.13}$$

โดยที่ m_i และ n_i สามารถพิจารณาว่าเป็นพารามิเตอร์อิสระที่ใช้ในการจูน ASA สำหรับแต่ละปัญหาได้

SA แบบต่างๆ ที่ได้กล่าวมาข้างต้น ล้วนแต่มีหลักการทำงานที่คล้ายคลึงกัน จุดประสงค์หลักที่ได้พัฒนา SA แบบต่างๆ นั้นก็เพื่อเพิ่มประสิทธิภาพในการค้นหาคำตอบ โดยเฉพาะในเรื่องของความเร็ว ในหลายๆ โอกาส SA แบบพื้นฐานก็เพียงพอต่อการใช้งาน การค้นหาคำตอบด้วย SA อย่างง่ายๆ แสดงในตัวอย่างต่อไปนี้

■ **ตัวอย่างที่ 2.1** การค้นหาคำตอบด้วย SA

ตัวอย่างนี้เป็นการค้นหาค่าที่น้อยที่สุดของฟังก์ชัน peaks (ดูรูปที่ 2.2) ที่ได้เคยยกตัวอย่างมาแล้วในการค้นหาคำตอบแบบอื่นๆ รูปที่ 2.3-(ก) และ (ข) แสดงผลการค้นหาคำตอบของ SA โดยมีค่าอุณหภูมิเริ่มต้น $T(0) = 20$ และค่าอุณหภูมิสุดท้ายเท่ากับ 0.1 การปรับค่าอุณหภูมิใช้ความสัมพันธ์ $T(k) = \alpha T(k-1)$ โดยใช้ค่า $\alpha = 0.99$ จำนวนการวนรอบก่อนที่จะมีการเปลี่ยนแปลงอุณหภูมิเท่ากับ 75 จากผลที่ได้จะเห็นว่า SA ให้ผลลัพธ์เป็นคำตอบที่น้อยที่สุดแบบวงแคบเฉพาะถิ่น เราสามารถวิเคราะห์ได้ว่าอุณหภูมิเริ่มต้นมีค่าสูงไม่เพียงพอ เมื่อพิจารณาผลลัพธ์ที่ได้ในรูปที่ 2.3-(ค) และ (ง) SA ให้ผลการค้นหาเป็นคำตอบที่น้อยที่สุดแบบวงกว้าง ค่าอุณหภูมิเริ่มต้นของระบบในกรณีนี้คือ $T(0) = 60$

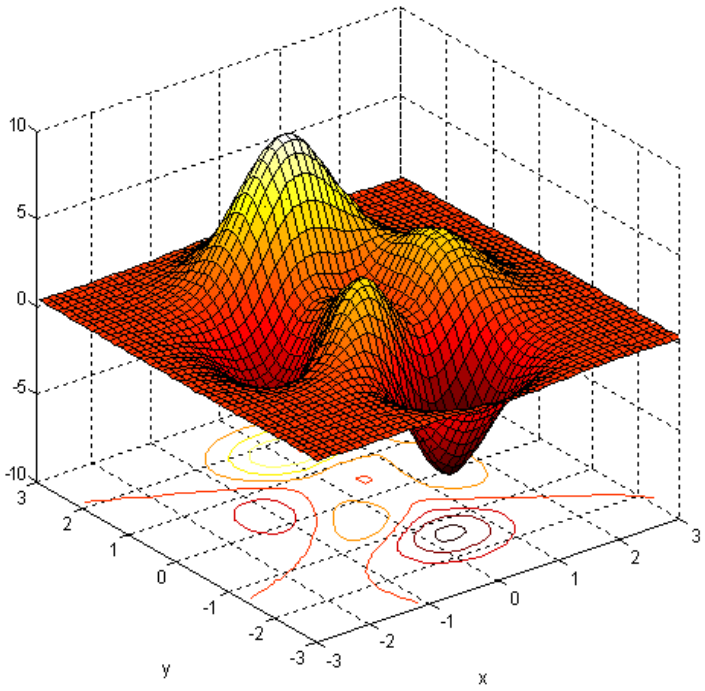
รูปที่ 2.4 แสดงผลการค้นหาด้วยค่าพารามิเตอร์ $\alpha = 0.95$ เราจะเห็นได้ว่าคำตอบที่ได้มีทั้งแบบวงแคบเฉพาะถิ่นและแบบวงกว้าง ผลลัพธ์ดังกล่าวแสดงให้เห็นถึงการใช้งาน SA จะต้องมีการปรับแต่งพารามิเตอร์ค่อนข้างละเอียด ซึ่งค่าของพารามิเตอร์ต่างๆ จะแตกต่างกันไปตามปัญหาที่กำลังพิจารณา

เราจะเห็นได้ว่าพารามิเตอร์ที่สำคัญที่สุดของ SA คืออุณหภูมิ การกำหนดค่าอุณหภูมิเริ่มต้นหรือตารางจัดการการอบอ่อน ล้วนแต่มีผลกระทบต่อประสิทธิภาพการค้นหาคำตอบของระบบ ซึ่งเป็นธรรมชาติของการอบอ่อนจริงๆ ในการออกแบบใช้งาน SA จึงควรเน้นที่พารามิเตอร์อุณหภูมิดังกล่าว ■

ในหัวข้อต่อไปจะได้กล่าวถึงตัวอย่างการนำเอา SA ไปประยุกต์ใช้งานจริง

2.3 การประยุกต์ใช้งาน SA: การจดจำหน้าคนจากภาพดิจิตอล

เนื้อหาในหัวข้อนี้เรียบเรียงจากงานใน [Xu et al., 2007] (โดยได้รับอนุญาติจากเจ้าของผลงาน) การประยุกต์ใช้งาน SA ในที่นี้เกี่ยวกับงานการจดจำใบหน้า (face recognition) โดยใช้การวิเคราะห์องค์ประกอบหลักแบบ 2 มิติ



รูปที่ 2.2: ฟังก์ชัน peaks สำหรับสาธิตการทำงานของ SA

(2D Principal Component Analysis หรือ 2DPCA) รายละเอียดต่างๆ มีดังต่อไปนี้

2.3.1 การจดจำหน้าด้วย 2DPCA

งานในด้านการจดจำหน้านั้นเดิมทีมีการใช้ PCA เป็นหลักในการแทนข้อมูลภาพมาเป็นเวลานาน จนกระทั่งมีการนำเสนอ 2DPCA ที่ซึ่งพิจารณาข้อมูลในรูปแบบ 2 มิติ คุณลักษณะดังกล่าวแตกต่างไปจาก PCA ซึ่งพิจารณาข้อมูลในรูปของเวกเตอร์ นั่นคือ 1 มิติ ดังนั้น 2DPCA จึงเหมาะสมในการนำมาลดมิติข้อมูลภาพกว่า PCA

พิจารณาเซตของภาพหน้าคน $\{X_1, X_2, \ldots, X_N\}$ ทั้งหมด N ภาพสำหรับใช้ฝึกสอน (รูปที่ 2.5 2DPCA ใช้ภาพทั้งหมดในการคำนวณเมตริกซ์ความแปรปรวนร่วมเกี่ยว (covariance matrix) ของภาพดังนี้

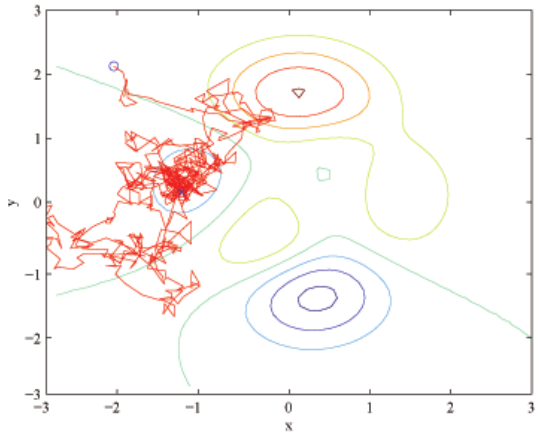
$$\begin{aligned} C &= E[(X - E(X))^T (X - E(X))] \\ &= \tfrac{1}{N} \textstyle\sum_{i=1}^{N} (X_i - \bar{X})^T (X_i - \bar{X}) \end{aligned} \tag{2.14}$$

โดยที่ X_i เป็นภาพฝึกสอนภาพที่ i มีขนาดเท่ากับ $h \times w$ $\bar{X}$ เป็นเมตริกซ์ภาพเฉลี่ยของภาพฝึกสอนทั้งหมด N ภาพ ถ้าพิจารณาภาพที่ต้องการคำนวณหา 2DPCA เป็นเมตริกซ์ A ใดๆ ที่มีขนาดเท่ากับ $h \times w$ และให้ **x** เป็นคอลัมภ์เวกเตอร์หนึ่งหน่วยที่มีมิติเท่ากับ w เราสามารถคำนวณภาพฉายของ A ไปบน **x** และได้ผลเป็นเวกเตอร์ **y** ที่มีมิติเท่ากับ h ดังนี้

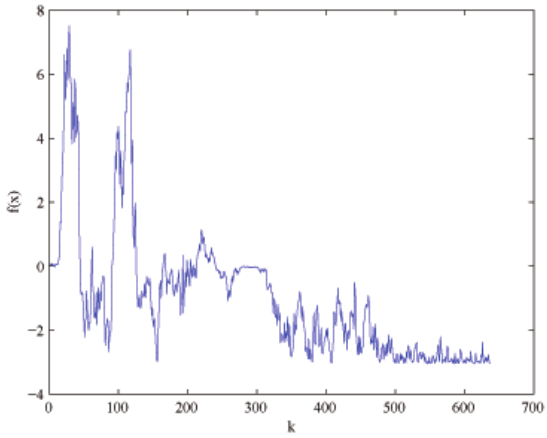
$$\mathbf{y} = A\mathbf{x} \tag{2.15}$$

แนวคิดหลักของ 2DPCA คือการเลือกเวกเตอร์ภาพฉาย **x** (projection vector) ที่ดี (เพื่อเป็นตัวแทนเมตริกซ์ A) โดยสามารถพิจารณาจากค่ารวมการกระจายของตัวอย่างเวกเตอร์ภาพที่ถูกฉาย (projected vector) ที่มากที่สุด ซึ่งเป็นไปตามเงื่อนไขการหาค่ามากที่สุดของสมการต่อไปนี้

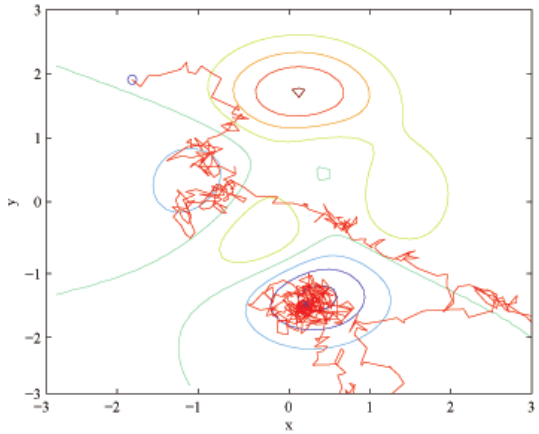
$$J(w) = tr(S_w) \tag{2.16}$$



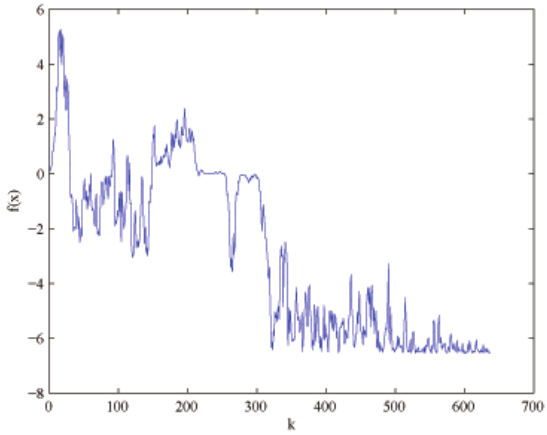
(ก) ผลการค้นหาคำตอบด้วย $T(0) = 20$ และ $\alpha = 0.99$



(ข) ค่าฟังก์ชันวัตถุประสงค์



(ค) ผลการค้นหาคำตอบด้วย $T(0) = 60$ และ $\alpha = 0.99$



(จ) ค่าฟังก์ชันวัตถุประสงค์

รูปที่ 2.3: ตัวอย่างการค้นหาค่าน้อยที่สุดของฟังก์ชัน peaks เครื่องหมาย o คือคำตอบหรือสถานะเริ่มต้นของระบบ และเครื่องหมาย x คือคำตอบหรือสถานะสุดท้ายของระบบ เส้นระหว่างเครื่องหมายทั้งสองคือเส้นทางเดินของคำตอบของระบบในระหว่างการค้นหาคำตอบของอัลกอริทึมการอบอ่อนจำลอง

โดยที่ S_w เป็นเมตริกซ์ความแปรปรวนร่วมเกี่ยวของเวกเตอร์ภาพที่ถูกฉายจากเซตภาพฝึกสอนทั้งหมด และ $tr(S_w)$ เป็นรอย (trace) ของเมตริกซ์ S_w เมตริกซ์ความแปรปรวนร่วมเกี่ยว S_w สามารถเขียนได้ดังนี้

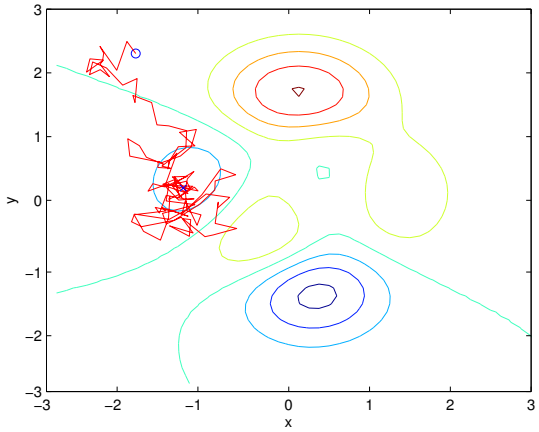
$$\begin{aligned} S_w &= E[y - E(y)]^T[y - E(y)] \\ &= E[(X - E(X))w]^T[(X - E(X))w] \end{aligned} \tag{2.17}$$

ดังนั้นจะได้ว่า

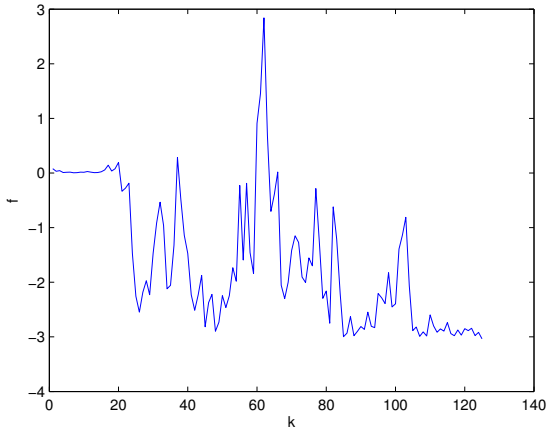
$$J(w) = w^T C w \tag{2.18}$$

เวกเตอร์ภาพฉายที่เหมาะที่สุด (optimal projection vector) $w_1, w_2, \ldots, w_d$ คือเมตริกซ์ลักษณะเฉพาะเชิงตั้งฉากปกติ (orthonormal eigenvector) ของเมตริกซ์ความแปรปรวนร่วมเกี่ยว C สำหรับค่าลักษณะเฉพาะ (eigenvalue) ที่มีค่ามากที่สุด d ค่าแรก

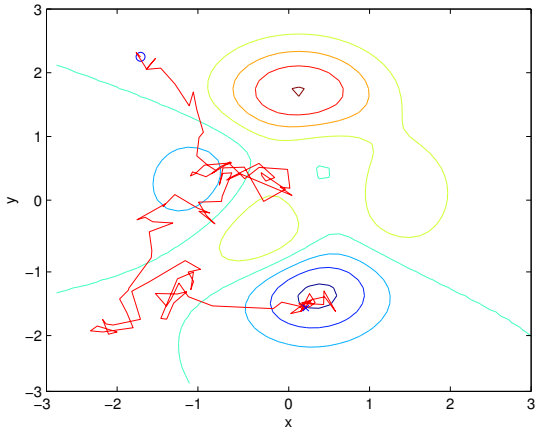
ดังนั้นแต่ละภาพที่ใช้ฝึกสอนจะมีตัวแทนเป็นเวกเตอร์คุณลักษณะ (feature vector) $Y_i = [y_{i1}, y_{i2}, \ldots, y_{id}]$ ที่สามารถคำนวณได้จาก $y_{ik} = X_i w_k$ โดยที่ $k = 1, 2, \ldots, d$ (พิจารณาขนาดของเวกเตอร์คุณลักษณะที่ลดน้อยลง



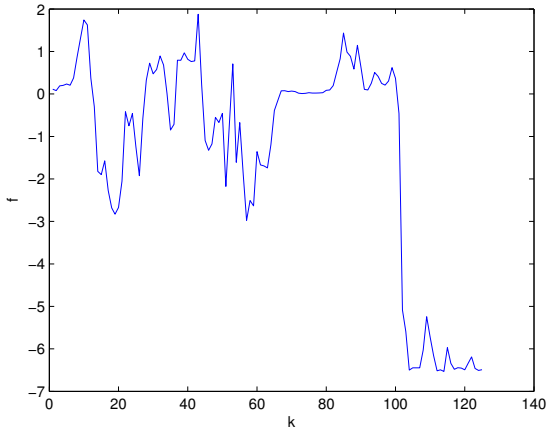
(ก) ผลการค้นหาคำตอบด้วย $T(0) = 20$ และ $\alpha = 0.95$



(ข) ค่าฟังก์ชันวัตถุประสงค์



(ค) ผลการค้นหาคำตอบด้วย $T(0) = 60$ และ $\alpha = 0.95$



(จ) ค่าฟังก์ชันวัตถุประสงค์

รูปที่ 2.4: ตัวอย่างการค้นหาค่าน้อยที่สุดของฟังก์ชัน peaks เครื่องหมาย o คือคำตอบหรือสถานะเริ่มต้นของระบบ และเครื่องหมาย x คือคำตอบหรือสถานะสุดท้ายของระบบ เส้นระหว่างเครื่องหมายทั้งสองคือเส้นทางเดินของคำตอบของระบบในระหว่างการค้นหาคำตอบของอัลกอริทึมการอบอ่อนจำลอง

รูปที่ 2.5: ตัวอย่างภาพหน้าคนสำหรับฝึกสอน (พิมพ์ซ้ำจาก [Xu et al., 2007])

เมื่อเทียบกับขนาดของเมตริกซ์ภาพต้นฉบับ)

ในทำนองเดียวกัน เราสามารถคำนวณหาเวกเตอร์คุณลักษณะของภาพที่จะใช้ทดสอบ $Y_t = [y_{t1}, y_{t2}, \ldots, y_{td}]$ เพื่อใช้ในการเปรียบเทียบความคล้ายกับภาพที่ใช้ฝึกสอนและนำไปทำการคัดแยกต่อไปได้

2.3.2 การวัดความคล้ายด้วย 2DPCA

การวัดความคล้ายของ 2DPCA ที่นิยมใช้กันมีดังนี้

- การวัดระยะแบบ Frobenius

$$d_F(Y_i, Y_j) = \left[\sum_{k=1}^{d}\sum_{l=1}^{m}(y_{ik}(l) - y_{jk}(l))^2\right]^{1/2} \tag{2.19}$$

- การวัดระยะแบบ Yang

$$d_Y(Y_i, Y_j) = \sum_{k=1}^{d}\|y_{ik} - y_{jk}\|_2 \tag{2.20}$$

- การวัดระยะแบบ Volume

$$d_V(Y_i, Y_j) = \sqrt{|(Y_i - Y_j)^T(Y_i - Y_j)|} \tag{2.21}$$

อย่างไรก็ดี การวัดความคล้ายข้างต้นมีลักษณะเป็นการวัดที่ไม่สามารถปรับเปลี่ยนได้ จึงได้มีการนำเสนอวิธีการวัดความคล้ายแบบใหม่โดยใช้ SA ดังรายละเอียดต่อไปนี้

2.3.3 การเรียนรู้สำหรับการวัดความคล้ายด้วย SA

พิจารณาเซตของภาพหน้าคนที่ใช้ในการฝึกสอนทั้งหมด N ภาพคือ $\{X_1, X_2, \ldots, X_N\}$ โดยที่ X_i เป็นภาพฝึกสอนที่ i และมีขนาดเท่ากับ $m \times n$ กำหนดให้เมตริกซ์คุณลักษณะของภาพ X_i ที่คำนวณได้จาก 2DPCA คือ $Y_i = [y_{i1}, y_{i2}, \ldots, y_{id}]$ โดยที่ $i = 1, 2, \ldots, N$ และ $y_{ik}, k = 1, 2, \ldots, d$ คือเวกเตอร์ขนาดเท่ากับ m ดังนั้น ภาพฝึกสอน X_i และ X_j จะมีเมตริกซ์คุณลักษณะ Y_i และ Y_j เราสามารถเรียนรู้ฟังก์ชันการวัดความคล้ายระหว่างข้อมูลเมตริกซ์คุณลักษณะทั้งสองได้

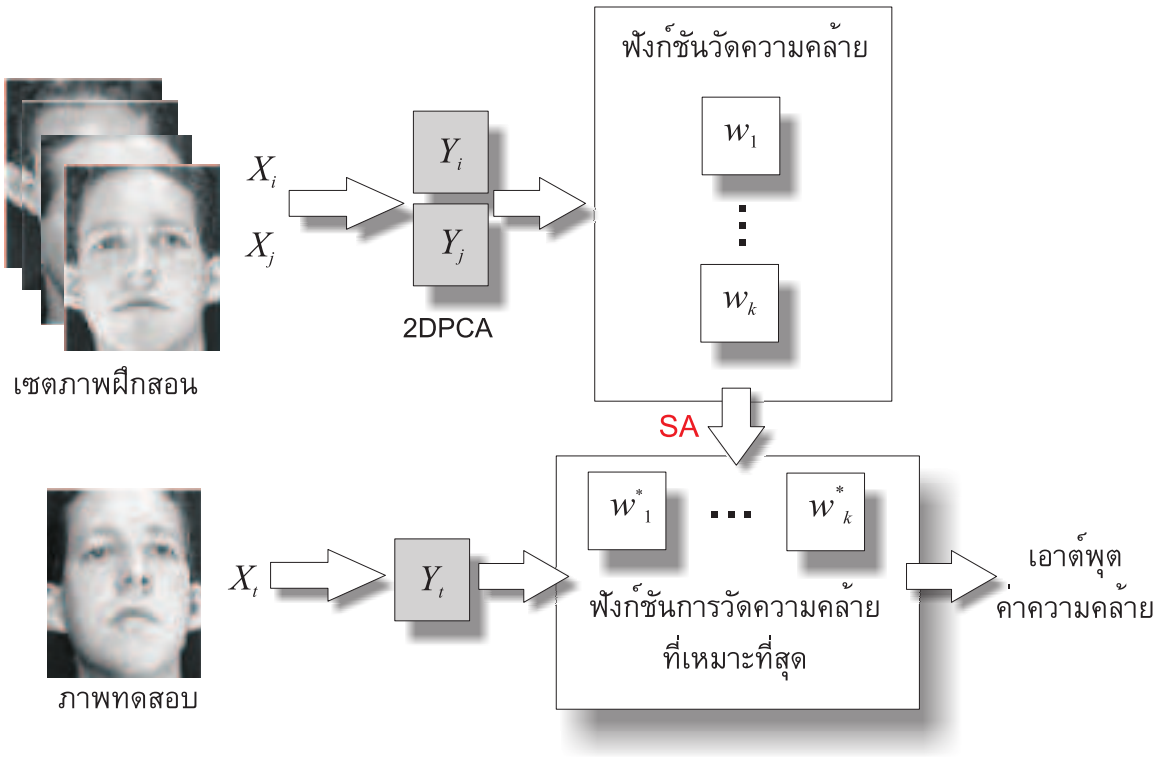
โดยปกติแล้วปริมาณ $\|y_{ik} - y_{jk}\|_2, k = 1, 2, \ldots, d$ สามารถใช้แสดงถึงความแตกต่างระหว่างภาพ X_i และ X_j (ด้วยการวัดความคล้ายระหว่างเมตริกซ์คุณลักษณะ Y_i และ Y_j) อย่างไรก็ดี การนำเอาปริมาณดังกล่าวมาใช้ในการวัดความคล้ายโดยตรงอาจจะให้การวัดความคล้ายที่ยังไม่ดีที่สุด จึงได้มีการนำเอาอัลกอริทึม SA มาเรียนรู้หาฟังก์ชันการวัดความคล้ายที่เหมาะที่สุด กล่าวคือใช้อัลกอริทึม SA ในการค้นหาค่าน้ำหนัก $w_k, k = 1, 2, \ldots, d$ เพื่อใช้ในการปรับฟังก์ชันการวัดความคล้ายดังนี้

$$d(Y_i, Y_j) = \sum_{k+1}^{d} w_k\|y_{ik} - y_{jk}\|_2 / \sum_{k=1}^{d} w_k \tag{2.22}$$

ฟังก์ชันการวัดความคล้ายข้างต้นเป็นการรวมกันแบบเชิงเส้นของแต่ละเวกเตอร์คุณลักษณะ $y_{ik}, k = 1, 2, \ldots, d$ ของเมตริกซ์คุณลักษณะ Y_i ด้วยค่าน้ำหนัก w_k ที่เหมาะสมที่สุด ฟังก์ชันดังกล่าวนี้จะถูกใช้เป็นฟังก์ชันวัตถุประสงค์ โดยมีค่าความผิดพลาดจากการวัดความคล้ายของภาพเทียบกับภาพจากเซตฝึกสอนเป็นค่าวัตถุประสงค์ ซึ่งจะใช้สำหรับประเมินสถานะของระบบใน SA แผนผังการวัดความคล้ายแบบเหมาะสมที่สุดของ 2DPCA ด้วย SA มีแสดงในรูปที่ 2.6 อัลกอริทึมของการใช้ SA ในการเรียนรู้การวัดความคล้ายสรุปได้ดังนี้

▷ อัลกอริทึมการค้นหาค่าน้ำหนักการวัดความคล้ายของ 2DPCA ด้วย SA

1. กำหนดค่าเริ่มต้น $T_{\max}, l_{\max}, k_{\max}, T(l), w_0, N(x), T = T_{\max}, w = w_0$ โดยที่ $w = (w_1, \ldots, w_d)^T$ และ w_0 เป็นค่าสุ่มเริ่มต้นของเวกเตอร์คุณลักษณะมิติ d
2. ทำการสุ่มเลือกคำตอบเวกเตอร์คุณลักษณะรอบข้างจาก w ซึ่งกำหนดโดย $w' = N(w)$ ที่ซึ่ง $N(\cdot)$ เป็นฟังก์ชันหาเวกเตอร์รอบข้าง



รูปที่ 2.6: แผนผังการวัดความคล้ายของ 2DPCA แบบเหมาะที่สุดด้วย SA

3. คำนวณหาค่าวัตถุประสงค์ระหว่าง w และ w' ที่ให้ค่าระยะทางการวัดความคล้ายด้วยสมการที่ 2.22 ที่น้อยที่สุด โดยพิจารณาค่าวัตถุประสงค์ของ w และ w' คือ $\Delta c = c(w') - c(w)$

4. ถ้า $\Delta c < 0$ แล้ว กำหนด $w = w'$ มิฉะนั้นถ้า $e^{-\Delta c/T} > rand$ แล้วกำหนดให้ $w = w'$ โดยที่ $rand$ คือค่าสุ่มตัวเลข

5. ปรับค่าอุณหภูมิตามตารางการจัดการการอบอ่อนของ SA นั่นคือ $T = T(l)$ แล้วเริ่มการค้นหาด้วยค่าอุณหภูมิใหม่อีกครั้ง ◁

กำหนดให้ค่าน้ำหนักการเรียนรู้ที่ได้จาก SA คือ

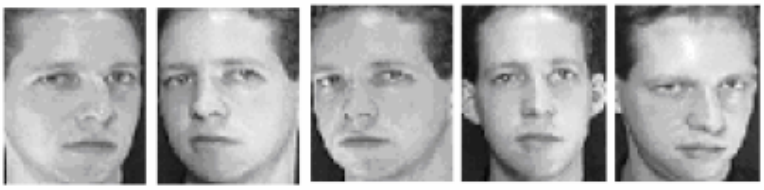
$$w^* = (w_1^*, \ldots, w_d^*)^T \tag{2.23}$$

ค่าของน้ำหนัก w^* ที่ได้จะถูกใช้ในการวัดความคล้ายดังนี้

$$d_{SA}(Y_i, Y_j) = \sum_{k=1}^{d} w_k^* \|y_{ik} - y_{jk}\|_2 / \sum_{k=1}^{d} w_k^* \tag{2.24}$$

ดังนั้น ถ้ากำหนดให้เมตริกซ์คุณลักษณะ Y_t สำหรับทดสอบการจดจำภาพใบหน้า และกำหนดเมตริกซ์คุณลักษณะ Y_i จากเซตของภาพที่ใช้ฝึกสอน การจดจำภาพใบหน้าของ Y_t เทียบกับภาพในเซตฝึกสอนสามารถคำนวณได้จาก

$$c = \arg\min_i d_{SA}(Y_t, Y_i) \tag{2.25}$$



รูปที่ 2.7: ตัวอย่างภาพหน้าคนสำหรับทดสอบ (พิมพ์ซ้ำจาก [Xu et al., 2007])

ตารางที่ 2.1: อัตราการจดจำที่ดีที่สุดพร้อมจำนวนเวกเตอร์ภาพฉายของการวัดความคล้ายแบบต่างๆ

วิธีการวัด	จำนวนของเวกเตอร์ภาพฉาย	อัตราการจดจำ (%)
การวัดระยะแบบ Frobenius	4,7,8	95.5
การวัดระยะแบบ Yang	3	96.0
การวัดระยะแบบ Volume	2,3	97.0
การวัดความคล้ายด้วย SA	3	97.5

หรือ

$$c = \arg\min_i \sum_{k=1}^{d} w_k^* \|y_{ik} - y_{tk}\|_2 / \sum_{k=1}^{d} w_k^* \tag{2.26}$$

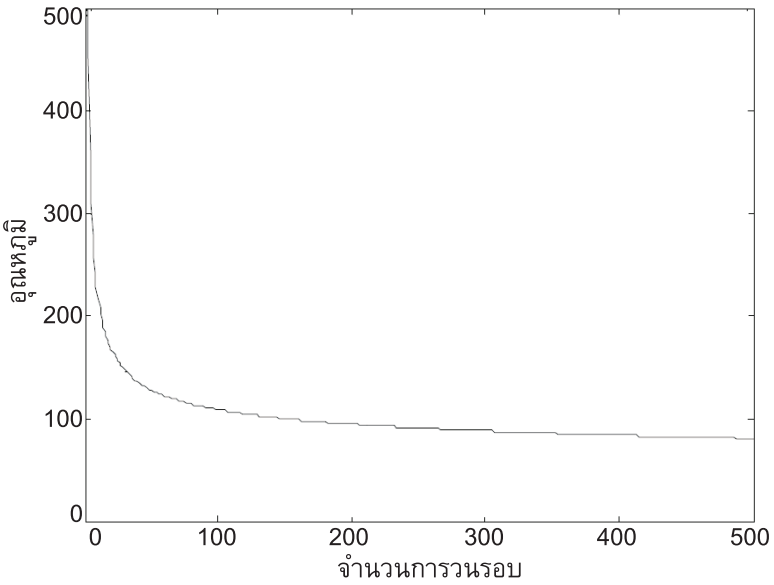
โดยที่ $c \in [1, 2, \ldots, N]$ และ Y_t จะถูกคัดแยกในกลุ่มของ c

2.3.4 ผลการทดลองการจดจำใบหน้า

ในงานข้างต้นได้รายงานผลการจดจำภาพใบหน้าจากการใช้ SA ช่วยในการวัดความคล้าย รูปภาพใบหน้าที่ใช้ในการทดสอบแสดงในรูปที่ 2.7 ตารางที่ 2.1 แสดงผลการจดจำเทียบกับการวัดความคล้ายแบบอื่นๆ จะเห็นได้ว่าการใช้การวัดความคล้าย ที่ซึ่งผ่านการเรียนรู้เพื่อหาค่าน้ำหนักที่เหมาะที่สุดด้วย SA ให้ผลการจดจำที่ดีกว่าการวัดความคล้ายแบบอื่นๆ ตารางการจัดการการอบอ่อนที่ใช้เป็นแบบลอการิทึม โดยมีค่าเริ่มต้นของอุณหภูมิ ($T_{\max}$) ที่ 500 และค่าสูงสุดของตัวชี้เวลาของ SA ($k_{\max}$) ในแต่ละรอบอุณหภูมิ รูปที่ 2.8 แสดงการปรับอุณหภูมิที่ใช้ในงานนี้ SA ในงานนี้ถูกใช้เป็นเครื่องมือหาค่าเหมาะที่สุด สำหรับวัดความคล้ายของภาพใบหน้าคน เพื่อใช้ในการจดจำใบหน้าได้อย่างมีประสิทธิภาพ

2.4 สรุป

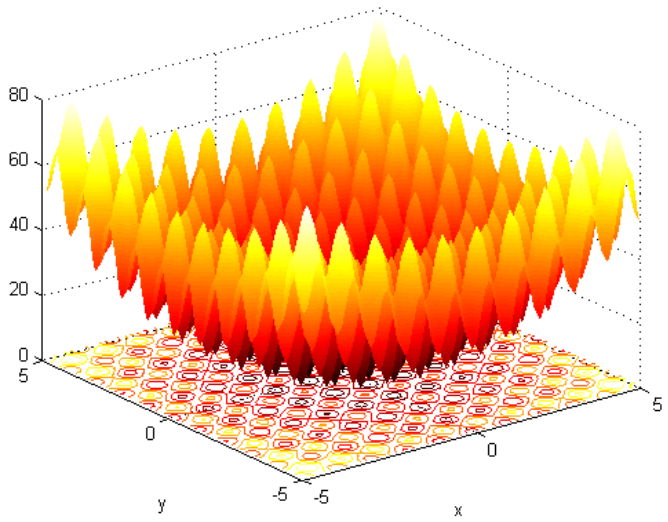
SA มีข้อได้เปรียบในการใช้งานที่เห็นได้ชัดคือความยืดหยุ่นและทนทานในการค้นหาคำตอบที่เหมาะที่สุดแบบวงกว้าง ตัว SA เองไม่ได้ใช้ข้อมูลของความชันหรือเกรเดียนต์ (gradient) เหมือนกับในหลายๆ วิธีของการค้นหา SA เองสามารถใช้งานกับปัญหาที่มีความไม่เป็นเชิงเส้นสูง รวมไปถึงกับปัญหาที่อยู่ในรูปของฟังก์ชันที่ไม่สามารถหาอนุพันธ์ได้ (นั่นคือไม่สามารถใช้เกรเดียนต์ในการค้นหาคำตอบได้) การประยุกต์ใช้งาน SA เองยังมีความหลากหลาย ข้อเสียของ SA นั้นอยู่ที่ต้องมีการคำนวณค่อนข้างมาก ถึงแม้ว่าจะมีการนำเสนออัลกอริทึมของ SA แบบดัดแปรเพื่อให้ทำงานได้เร็วขึ้น แต่การนำไปโปรแกรมใช้งานจริงยังมีความยุ่งยากและยังไม่เป็นที่นิยมใช้กันอย่างกว้างขวางนัก อย่างไรก็ดี SA ยังถือเป็นเครื่องมือในการค้นหาคำตอบที่ใช้งานง่ายและมีประสิทธิภาพสูงเพียงพอต่อการประยุกต์ใช้งานได้อย่างมากมาย



รูปที่ 2.8: การลดลงของอุณหภูมิแบบลอการิทึม (พิมพ์ซ้ำจาก [Xu et al., 2007])

โจทย์คำถาม

2.1. พิจารณาฟังก์ชันของ Rastrigin $f(x,y) = 20 + x^2 + y^2 - 10(\cos 2\pi x + \cos 2\pi y)$ ในรูปที่ 2.9 ที่ซึ่งประกอบไปด้วยจุดต่ำสุดทั้งแบบวงแคบเฉพาะถิ่น (local minima) และแบบวงกว้าง (global minima) จงออกแบบใช้ SA ในการหาค่าต่ำสุดแบบวงกว้างของฟังก์ชันดังกล่าว พร้อมทั้งหาค่าอุณหภูมิเริ่มต้นที่เหมาะสมของ SA อธิบายผลลัพธ์ที่ได้



รูปที่ 2.9: ฟังก์ชัน Rastrigin

2.2. จงออกแบบ SA สำหรับทำการคัดแยกกลุ่มข้อมูล (clustering data) โดยทำการสุ่มค่าเวกเตอร์ (x_1, x_2) จำนวนทั้งหมด 100 เวกเตอร์ให้มีตำแหน่งอยู่กันเป็นกลุ่มๆ กำหนดค่าจำนวนกลุ่มสูงสุดที่จะทำการคัดแยกแล้วใช้ SA หาตำแหน่งที่เป็นตัวแทนกลุ่มข้อมูล พร้อมทั้งระบุกลุ่มของเวกเตอร์ที่ทำการคัดแยกทั้งหมด

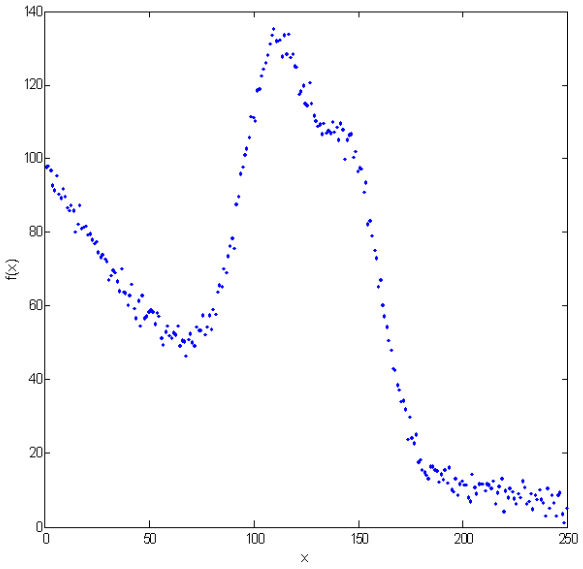
2.3. จงออกแบบใช้ SA ในการหาสัมประสิทธิ์การปรับเส้นโค้ง (curve fitting) ของข้อมูลจาก `gauss3.mat` ใน MATLAB (แสดงในรูปที่ 2.10) โดยใช้ฟังก์ชันต่อไปนี้

$$f(x) = a_0 e^{-b_0 x} + a_1 e^{-\left(\frac{x-b_1}{c_1}\right)^2} + a_2 e^{-\left(\frac{x-b_2}{c_2}\right)^2}$$

2.4. SA ได้รับความนิยมในการนำไปใช้ในกระบวนการฝึกสอนเครือข่ายประสาทเทียมได้ มีตัวอย่างงานมากมายที่ประยุกต์ใช้งาน SA ดังกล่าว จงอธิบายการนำเอา SA ไปใช้ในการหาโครงสร้างเหมาะที่สุดของเครือข่ายประสาทเทียม และทำการจำลองการทำงานโดยพิจารณาใช้ SA ในการค้นหาค่าน้ำหนักประสาทของเครือข่ายเพอร์เซ็พตรอน (perceptron) ขนาด 2 นิวรอนสำหรับปัญหา XOR

2.5. จากการค้นหาค่าเหมาะที่สุดของฟังก์ชัน $peaks$ ให้ทำการออกแบบปรับเปลี่ยนตารางจัดการการอบอ่อนใหม่ พร้อมทั้งเปรียบเทียบผลลัพธ์ที่ได้

2.6. จงออกแบบการเล่นเกมส์ XO ขนาด 3×3 ด้วยอัลกอริทึม SA และทำการเปรียบเทียบประสิทธิภาพด้านต่างๆ กับกลยุทธ์การค้นหาคำตอบแบบใช้ข้อมูลความรู้ต่างๆ ในการต่อสู้ในเกมส์ การออกแบบดังกล่าวสามารถพิจารณาระบบเป็นแบบดิสครีตหรือเป็นแบบเครื่องจักรสถานะ (state machine) สิ่งสำคัญที่สุดคือการออกแบบฟังก์ชันการประเมินหรือฟังก์ชันวัตถุประสงค์ ในการประเมินค่าสถานะปัจจุบันของเกมส์

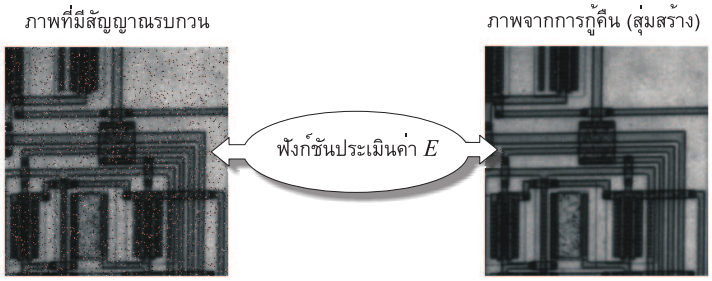


รูปที่ 2.10: ข้อมูลสำหรับการปรับเส้นโค้ง

2.7. SA สามารถนำไปใช้ในการสร้างข้อมูลภาพดิจิตอลได้ เช่นงานใน [Sundermann and Lemahieu, 1995] ซึ่งเป็นการกู้คืนภาพแบบ Positron Emission Tomography (PET) จากภาพที่มีสัญญาณรบกวน จงออกแบบใช้ SA ในการกู้คืนภาพระดับเทา ที่มีสัญญาณรบกวนแบบ Salt & Pepper (ดูแผนผังในรูปที่ 2.11 ประกอบ) พิจารณาใช้ฟังก์ชันการประเมินเป็นค่าความแตกต่างของรูปภาพที่มีสัญญาณรบกวน I^n กับรูปภาพที่สร้างขึ้นเพื่อการกู้คืน $\hat{I}$ ดังนี้

$$F = \sum_{i,j} [I_{ij}^n - \hat{I}_{ij}]^2$$

จงออกแบบ SA ในการกู้คืนภาพที่มีสัญญาณรบกวน (ลดสัญญาณรบกวนนั่นเอง) โดยทดสอบกับสัญญาณรบกวนแบบต่างๆ พร้อมทั้งแสดงผลเปรียบเทียบและวิเคราะห์ผลการปรับแต่งพารามิเตอร์ของ SA



รูปที่ 2.11: แผนผังการกู้คืนภาพด้วย SA

2.8. การจดจำภาพใบหน้าคนใน [Xu et al., 2007] ใช้ SA ในการเรียนรู้ค่าน้ำหนักในฟังก์ชันวัดความคล้าย จงอภิปรายผลของพารามิเตอร์ต่างๆ ของ SA ต่อประสิทธิภาพของการค้นหาค่าน้ำหนักที่เหมาะที่สุด รวมไปถึงผลของการปรับตารางการจัดการการอบอ่อนของ SA ในรูปแบบต่างๆ

บรรณานุกรม

J. A. Anderson and E. Rosenfeld, editors. *Neurocomputing: Foundation of Research*. M.I.T. Press, Cambridge, MA, 1988.

S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721--741, 1984.

B. Hajek. A tutorial survey of theory and applications of simulated annealing. In *Proceedings of the 24th IEEE Conference on Decision and Control*, volume 2, pages 755--760, 1985.

Fredric M. Ham and Ivica Kostanic. *Principle of Neurocomputing for Science & Engineering*, chapter 5, pages 209--215. McGraw-Hill, 2001.

L. Ingber. Very fast simulated re-annealing. In *Mathl. Comput. Modelling*, volume 12, pages 967--973, 1989.

L. Ingber, 1993. URL `ftp.alumni.caltech.edu:/pub/ingber/ASA-shar,ASA-shar.Z,ASA.tar.Z,ASA.tar.gz,ASA.zip`.

D. L. Isaacson and R. W. Madsen. *Markov Chains: Theory and Applications*. New York: Wiley, 1976.

S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220: 671--680, 1983. Reprinted in 1988, Anderson and Rosenfeld [1988], pp. 554-567.

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and F. Teller. Equation of state calculations by fast computing machines. In *J. Chem. Phys.*, volume 21, pages 1087--1092, 1953.

R. A. Rutenbar. Simulated annealing algorithms: An overview. *IEEE Circuits and Devices Magazine*, pages 19--26, January 1989.

Erik Sundermann and Ignace L. Lemahieu. Pet image reconstruction using simulated annealing. In Murray H. Loew, editor, *Proceedings of SPIE*, volume 2434 of *Medical Imaging 1995: Image Processing*, pages 378--386, May 1995.

H. Szu and R. Hartley. Fast simulated annealing. In *Phys. Lett.*, volume A 122, pages 157--162, 1987.

Z. Xu, L. Wang, and L. Yang. A simulated annealing and 2dpca based method for face recognition. In *3rd International IEEE Conference on Signal-Image Technologies and Internet-Based System*, pages 777--782, 2007.