11장. 최적화 알고리즘

오일석, 패턴인식, 교보문고, 2008.

들어가는 말

- 패턴인식은 최적화 문제를 많이 다룬다.
  - SVM은 여백을 최대로 하는 결정 직선을 찾는다.
  - 퍼셉트론이나 MLP는 오류를 최소로 하는 가중치 값을 찾는다.
  - …

일반적으로 최적화 문제는 (11.1)과 같이 기술할 수 있다. $J(\boldsymbol{\theta})$는 **목적 함수** target function 또는 비용 함수라 cost function 부르고 $\boldsymbol{\theta}$는 매개 변수라 한다. 경우에 따라 목적 함수를 최대로 하는 **최대화 문제**도 maximization problem 있고 최소로 하는 **최소화 문제**도 minimization problem 있다.

$$\left.\begin{array}{l} \text{최대화 문제:} \\ \quad J(\boldsymbol{\theta})\text{를 최대로 하는 } \hat{\boldsymbol{\theta}}\text{를 찾아라. 즉 } \hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\text{이다.} \\ \text{최소화 문제:} \\ \quad J(\boldsymbol{\theta})\text{를 최소로 하는 } \hat{\boldsymbol{\theta}}\text{를 찾아라. 즉 } \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\text{이다.} \end{array}\right\} \quad (11.1)$$

들어가는 말

- 패턴 인식의 문제 풀이는 크게 두 단계로 나누어 볼 수 있다.
  - 매개 변수 θ는 연속 공간일 수도 이산 공간일 수도 있음
    - SVM, MLP는 연속 공간, 특징 선택은 이산 공간

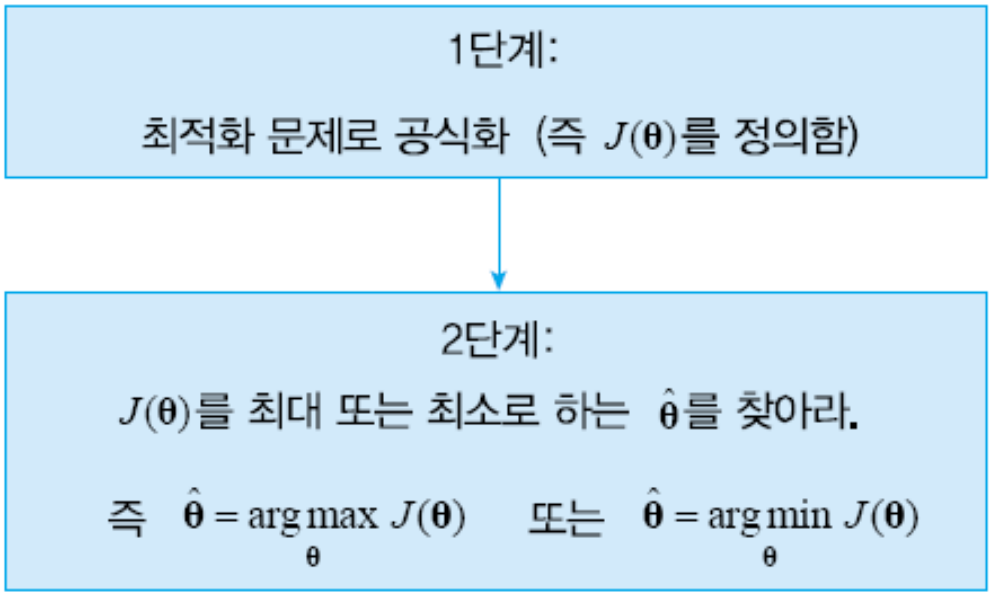


그림 11.1 패턴 인식에서의 문제 풀이 과정

들어가는 말

- 11 장의 목적
  - 이미 1-10 장에서 배운 최적화 알고리즘을 보다 명시적으로 드러내고 그들을 비교함으로써 알고리즘에 대한 이해의 깊이를 더함
  - 새로운 최적화 알고리즘 소개
    - 시뮬레이티드 어닐링, 유전 알고리즘 (기존 알고리즘이 안고 있는 지역 최적점 수렴 문제를 해결할 수 있는 여지를 가짐)

11.1 패턴인식의 최적화 문제와 풀이

이 책의 전반에 걸쳐 아주 많은 최적화 문제를 살펴 보았다. 여기서는 이들을 보다 명시적으로 드러내고 이들의 특성을 분석하고 이들이 서로 어떻게 다른지 살펴보기로 하자.

11.1.1 최적화 문제들

- 분류기 학습
  - 학습 집합 $X=\{(\mathbf{x}_1,\mathbf{t}_1), (\mathbf{x}_2,\mathbf{t}_2), \ldots, (\mathbf{x}_N,\mathbf{t}_N)\}$
  - $\boldsymbol{\theta}$는 가중치 벡터

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} d(\mathbf{o}_i, \mathbf{t}_i) \qquad (11.2)$$

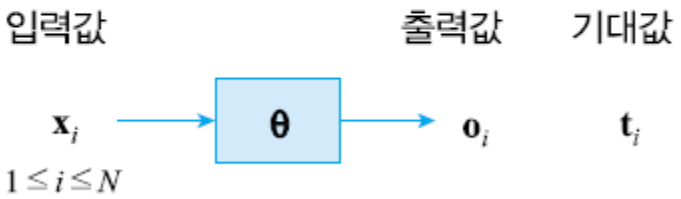


그림 11.2 분류기 학습 문제

그렇다면 $J(\boldsymbol{\theta})$를 최소화 하는 최적의 매개 변수 $\hat{\boldsymbol{\theta}}$ (즉 $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$)를 어떻게 찾을 것인가? 2~7장에서는 대부분 미분을 이용하여 해를 찾았는데 그 방법이 최선인가? 그것 말고 다른 대안은 없는가? 이것에 대한 답을 이 장에서 생각해 보기로 하자.

11.1.1 최적화 문제들

- 선택
  - 특징 선택 (9 장), 분류기 앙상블 선택 (12 장), k-NN을 위한 프로토타입 선택 등

선택 문제도 매개 변수 $\boldsymbol{\theta}$를 도입하여 표현할 수 있다. $\boldsymbol{\theta}$는 n 비트를 갖는 이진열이다. 예를 들어 $n=5$인 경우 부분 집합 $S_1=\{s_2,s_3,s_5\}$는 $\boldsymbol{\theta}=01101$로 표현한다. 즉 선택된 요소는 1을 갖고 제거된 요소는 0을 갖는다.

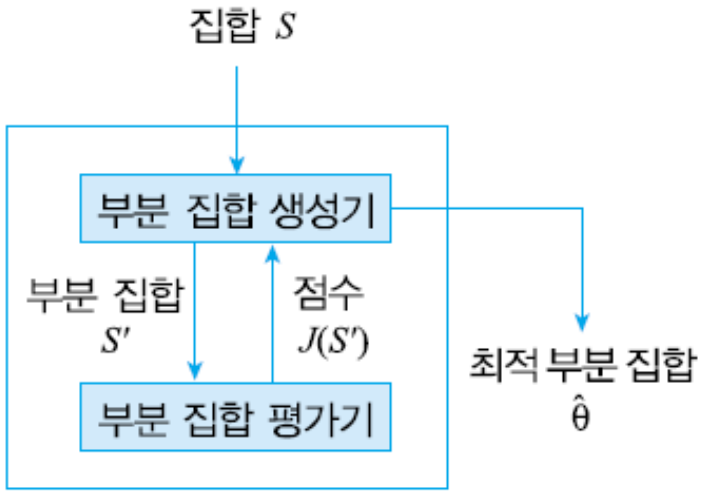
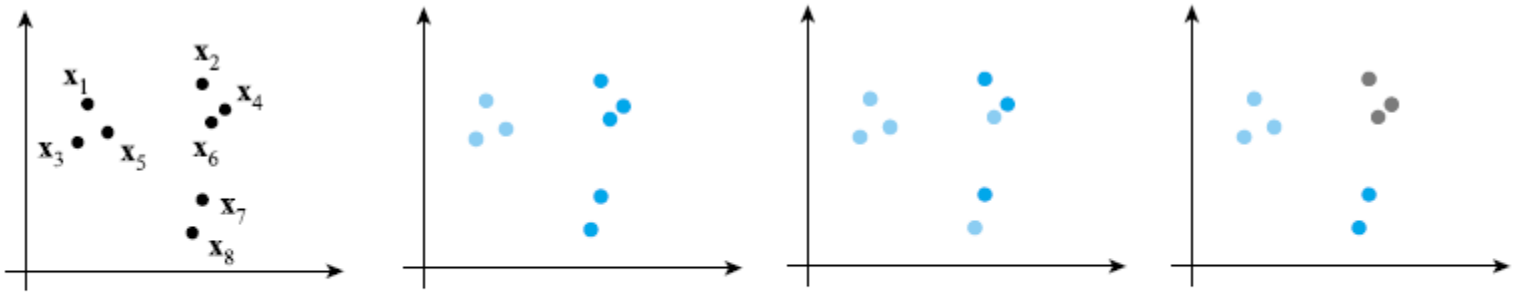


그림 11.3 선택 문제

11.1.1 최적화 문제들

- 군집화



(a) 샘플 집합 X (b) $\theta = 12121222$ (c) $\theta = 12121121$ (d) $\theta = 13131322$

그림 11.4 군집화 예 (연한 파랑은 군집1, 진한 파랑은 군집2, 검정은 군집3)

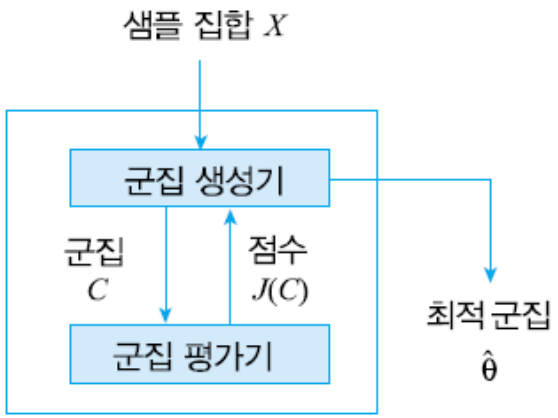


그림 11.5 군집화 문제

11.1.1 최적화 문제들

위에서 살펴본 학습, 선택, 그리고 군집화 문제뿐만 아니라 다른 문제들도 최적화 문제로 만들어 풀었다. 8장에서 다루었던 PCA는 변환된 샘플의 분산을 목적 함수로 정의한 후 이것을 최대로 하는 변환 행렬을 찾았다. PCA에서는 이 변환 행렬이 $\boldsymbol{\theta}$에 해당한다. 3장의 가우시언 혼합 모델도 최적화 문제로 풀었다. 이 최적화 문제를 푸는 알고리즘을 EM 알고리즘이라 불렀다. 이때는 가우시언의 평균 벡터와 공분산 행렬 그리고 가우시언 혼합을 표현하는 혼합 계수가 바로 $\boldsymbol{\theta}$에 해당한다.

11.1.2 문제 풀이

이들 문제를 어떻게 풀어 최적의 매개 변수 값을 찾아 낼 것인가?

- 문제의 난이도
  - $\boldsymbol{\theta}$의 차원. 작게는 수십~수백 , 크게는 수천~수만 차원
  - 목적 함수 $J(\boldsymbol{\theta})$의 복잡도. 지역 최적점이 하나뿐인 single-modal인 경우 미분식을 풀어 해결 가능, multi-modal인 경우는 보다 복잡한 알고리즘 필요
- 해 공간을 효율적으로 탐색하는 알고리즘 필요

11.1.2 문제 풀이

- 문제 풀이
  - 분석적 방법
    - 예, 미분식을 풀어 해를 구함 (11.2.1 절)
  - 수치적 방법 (그림 11.6)
    - 초기 해에서 출발하여 그것을 조금씩 개선해 나가는 방법

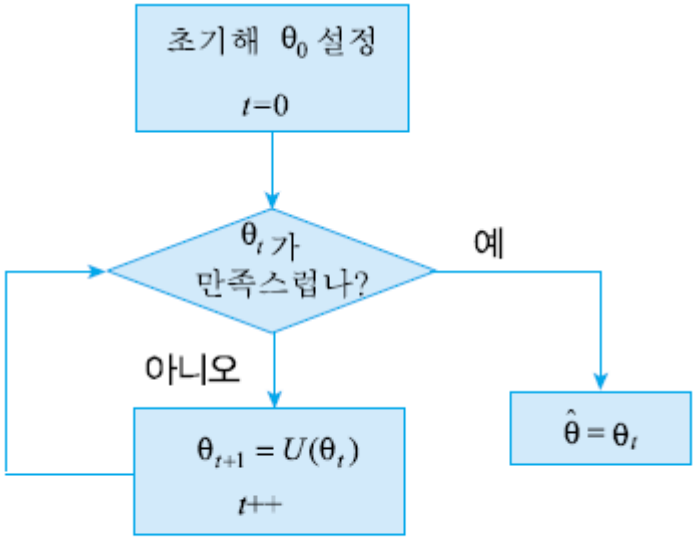


그림 11.6 최적해를 찾기 위한 반복 알고리즘

11.1.2 문제 풀이

- 수치적 방법 (그림 11.6)
  - 세 가지 사항을 고려해야 한다.

1. **초기해 설정.** 탐색의 초기 점을 잡는 과정이다. 이 초기 점에 따라 지역 최적 점에서 끝날 수도 있고 전역 최적 점으로 수렴할 수도 있다. 그럼 초기 점을 어디로 할 것인가?
2. **멈춤 조건.** 현재 해가 만족스러우면 그 해를 결과로 출력하고 멈춘다. 그렇다면 만족스러운지를 어떻게 판단할 것인가?
3. **해 갱신.** 현재 해를 다른 점으로 이동한다. 물론 품질이 개선된 점으로 이동해야 한다. 이동을 책임지는 함수 $U(.)$를 어떻게 만들 것인가?

11.1.2 문제 풀이

- 수치적 방법
  - 최적 해 (전역 최소점, θ_4)을 보장하는 것은 아니다. 지역 최소 점 (θ_1) 또는 최소 점 근방 (θ_2, θ_3, θ_5)을 찾고 멈출 수도 있다.

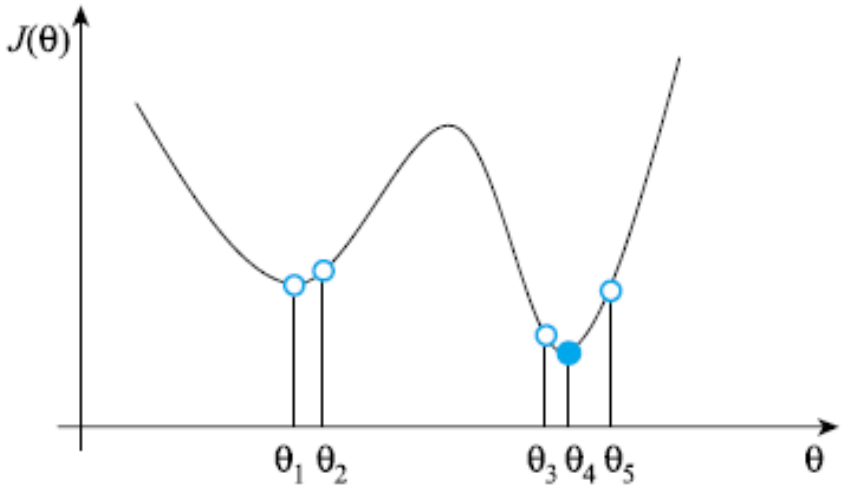


그림 11.7 최소화 문제에서 최적 해 (속이 찬 점)와 부 최적 해 (속이 빈 점)

11.2 미분을 이용한 방법

11.2.1 분석적 풀이

11.2.2 내리막 경사법

11.2.3 라그랑제 승수

11.2.4 최적화 알고리즘과 패턴인식 문제들

11.2.1 분석적 풀이

알고리즘 [11.1]이 미분을 이용하여 최고 점 또는 최소 점을 찾는 과정을 명확하게 설명한다. 이 알고리즘은 도함수의 derivative 값은 기울기 (경사)를 나타내고 최고 점 또는 최저 점에서는 기울기가 0이라는 성질을 이용하고 있다.

알고리즘 [11.1] 분석적 풀이

입력: 목적 함수 $J(\boldsymbol{\theta})$

출력: $\hat{\boldsymbol{\theta}}$ (최고 점 또는 최저 점)

알고리즘:

1. $J(\boldsymbol{\theta})$를 $\boldsymbol{\theta}$로 미분한다.
2. 방정식 $\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0$ 을 만족하는 $\hat{\boldsymbol{\theta}}$ 를 구한다.
3. **return** $\hat{\boldsymbol{\theta}}$;

예제 11.1 미분에 의한 분석적 방법

- 두 가지 예제

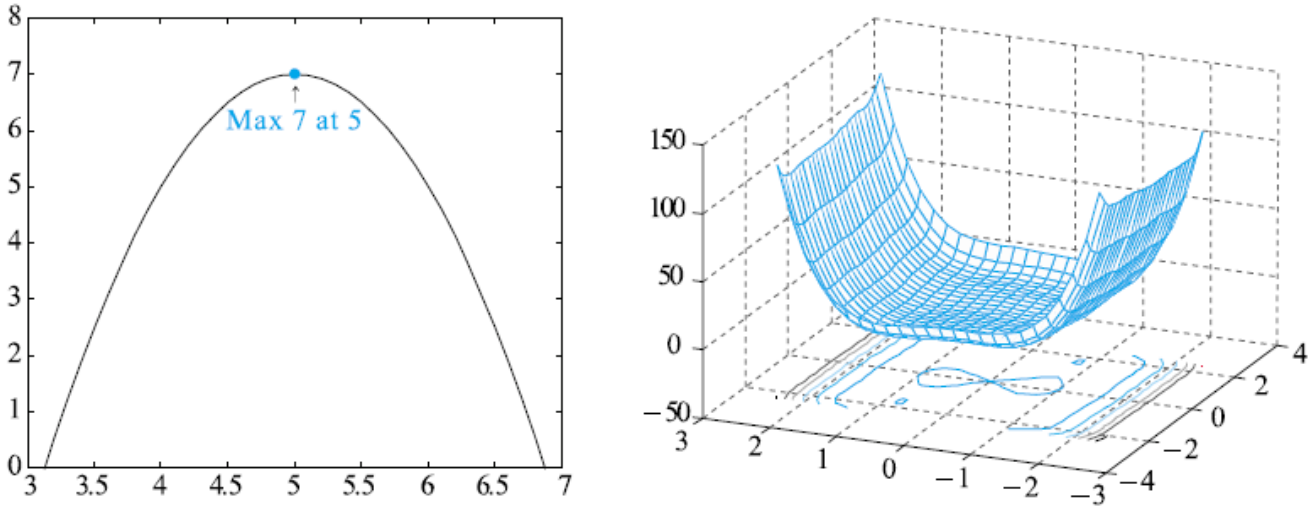
(1) 그림 11.8(a): $J(\theta) = -2\theta^2 + 20\theta - 43$

$\frac{\partial J}{\partial \theta} = -4\theta + 20 = 0$ 을 풀면 $\hat{\theta} = 5$ 이다.

(2) 그림 11.8(b): $J(\boldsymbol{\theta}) = (4 - 2.1\theta_1^2 + \theta_1^4/3)\theta_1^2 + \theta_1\theta_2 + (-4 + 4\theta_2^2)\theta_2^2$

$\frac{\partial J}{\partial \boldsymbol{\theta}} = (\frac{\partial J}{\partial \theta_1}, \frac{\partial J}{\partial \theta_2})^T = (2\theta_1^5 - 8.4\theta_1^3 + 8\theta_1 + \theta_2, 16\theta_2^3 - 8\theta_2 + \theta_1)^T = \mathbf{0}$ 을 풀면

$\hat{\boldsymbol{\theta}} = (-0.0898, 0.7126)^T$ 또는 $\hat{\boldsymbol{\theta}} = (0.0898, -0.7126)^T$ 에서 최소값 -1.0316을 갖는 것을 알 수 있다.



(a) 이차 함수의 최대화 (b) 여섯 혹을 가진 낙타 등 함수의 최소화

그림 11.8 미분에 의한 최적해 구하기

11.2.2 내리막 경사법 gradient descent method

- 내리막 경사법은 해를 반복 개선하는 수치적 방법의 일종
  - 현재 위치에서 경사가 가장 급격하게 떨어지는 방향을 찾고 그 방향으로 해를 약간 이동. 방향은 도함수로 알아냄
  - ρ는 학습률로서 이동량을 조절

내리막 경사법 $$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho \frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}t} \quad (11.3)$$

오르막 경사법 $$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \rho \frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}t} \quad (11.4)$$

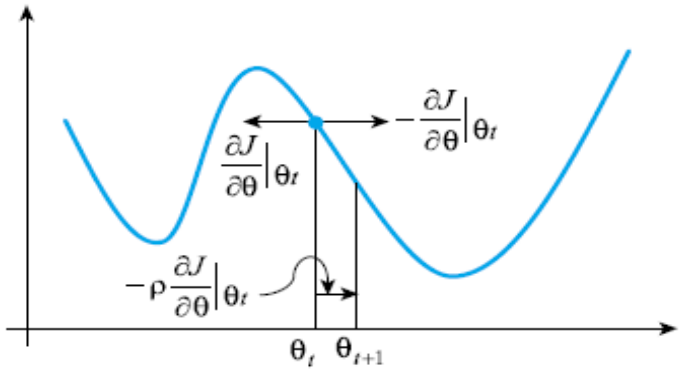


그림 11.9 내리막 경사법

11.2.2 내리막 경사법

알고리즘 [11.2] 내리막 경사법

입력: 목적 함수 $J(\boldsymbol{\theta})$, 학습률 ρ

출력: 최적해 $\hat{\boldsymbol{\theta}}$

알고리즘:

1. 초기해 $\boldsymbol{\theta}_0$을 설정한다.
2. $t = 0$;
3. **repeat** {
4. $\quad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho \frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_t}$;
5. $\quad$ t++;
6. } **until** (멈춤 조건);
7. $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_t$;

- 내리막 경사법은 욕심 알고리즘 (지역 최적점으로 수렴할 위험)

예제 11.2 내리막 경사법 예제 11.1의 낙타 등 함수

- 도함수는 $J'(\boldsymbol{\theta}) = \frac{\partial J}{\partial \boldsymbol{\theta}} = (\frac{\partial J}{\partial \theta_1}, \frac{\partial J}{\partial \theta_2})^{\mathrm{T}} = (2\theta_1^{\,5} - 8.4\theta_1^{\,3} + 8\theta_1 + \theta_2, 16\theta_2^{\,3} - 8\theta_2 + \theta_1)^{\mathrm{T}}$
- 초기값을 $\boldsymbol{\theta}_0=(-0.5,0.5)^{\mathrm{T}}$, 학습률 $\rho=0.01$로 하면,

$$\frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_0} = (-2.5125,-2.5)^{\mathrm{T}}$$

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 - 0.01 * \frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_0} = (-0.5,0.5)^{\mathrm{T}} - 0.01 * (-2.5125,-2.5)^{\mathrm{T}} = (-0.4748,0.525)^{\mathrm{T}}$$

$$\frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_1} = (-2.4228,-2.3596)^{\mathrm{T}}$$

$$\boldsymbol{\theta}_2 = \boldsymbol{\theta}_1 - 0.01 * \frac{\partial J}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_1} = (-0.4748,0.525)^{\mathrm{T}} - 0.01 * (-2.4228,-2.3596)^{\mathrm{T}}$$

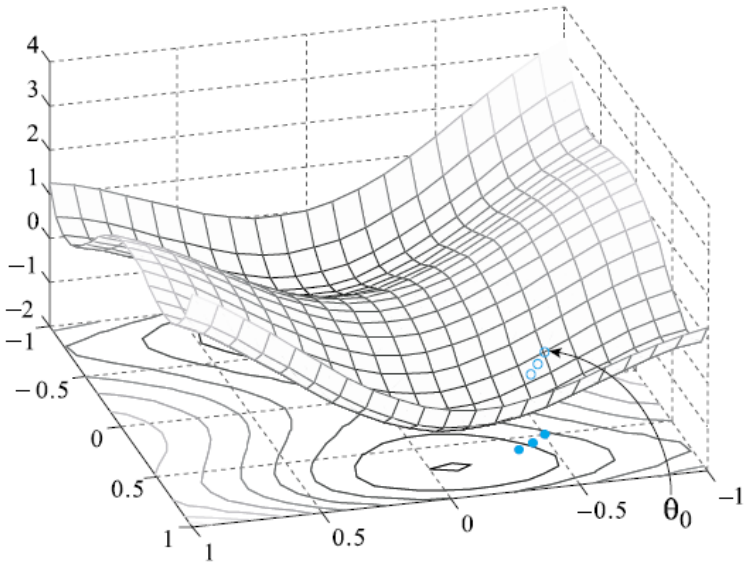
$$= (-0.4506,0.5486)^{\mathrm{T}}$$

점점 낮은 곳을 찾아간다.

$$J(\boldsymbol{\theta}_0) = -0.12604$$

$$J(\boldsymbol{\theta}_1) = -0.24906$$

$$J(\boldsymbol{\theta}_2) = -0.36036$$



Gottfried Wilhelm Leibniz

(1646년 7월 1일 ~ 1716년 11월 14일) 독일

Leibniz는 Isaac Newton (1643년 1월 4일 ~ 1727년 3월 31일, 영국)과 독립적으로 미적분학을 calculus 창안한 것으로 잘 알려져 있다. Newton은 Leibniz가 자신의 아이디어를 도용했다고 주장하기도 하였다. 그 때문에 영국과 유럽 대륙의 자존심 싸움으로까지 번졌는데 결국 독립적으로 창안한 것으로 판명되었다. 미적분학은 인류가 만들어 낸 학문 영역에 영향을 끼치지 않는 곳이 없을 정도로 그 영향력이 크다. 패턴 인식에서는 주로 최대 또는 최소 값을 갖는 매개 변수를 찾는데 미분을 많이 사용한다. Leibniz는 컴퓨터 개발의 토대가 된 이진 시스템을 정립하였으므로 컴퓨터 역사에 꼭 등장하기도 한다. 그는 과학 영역뿐 아니라 철학에서도 중요한 위치를 차지한다. 그는 Descartes와 Spinoza와 함께 17 세기 3대 이성론자로 rationalist 꼽히기도 한다. Leibniz에 대해 보다 상세한 내용은 [Brown(웹)]을 참고하기 바란다.

[Brown(웹)] Gregory Brown, *Online biography of Leibniz*, (http://www.gwleibniz.com/).

11.2.3 라그랑제 승수

- 조건부 최적화 문제 예 $J(\boldsymbol{\theta}) = \theta_1^2 + 2\theta_2^2$
  - 최소 점은 $(0,0)^T$
  - 하지만 '$2\theta_1 + \theta_2 = 0$이라는 조건 하'에 최소점을 구하는 문제라면?

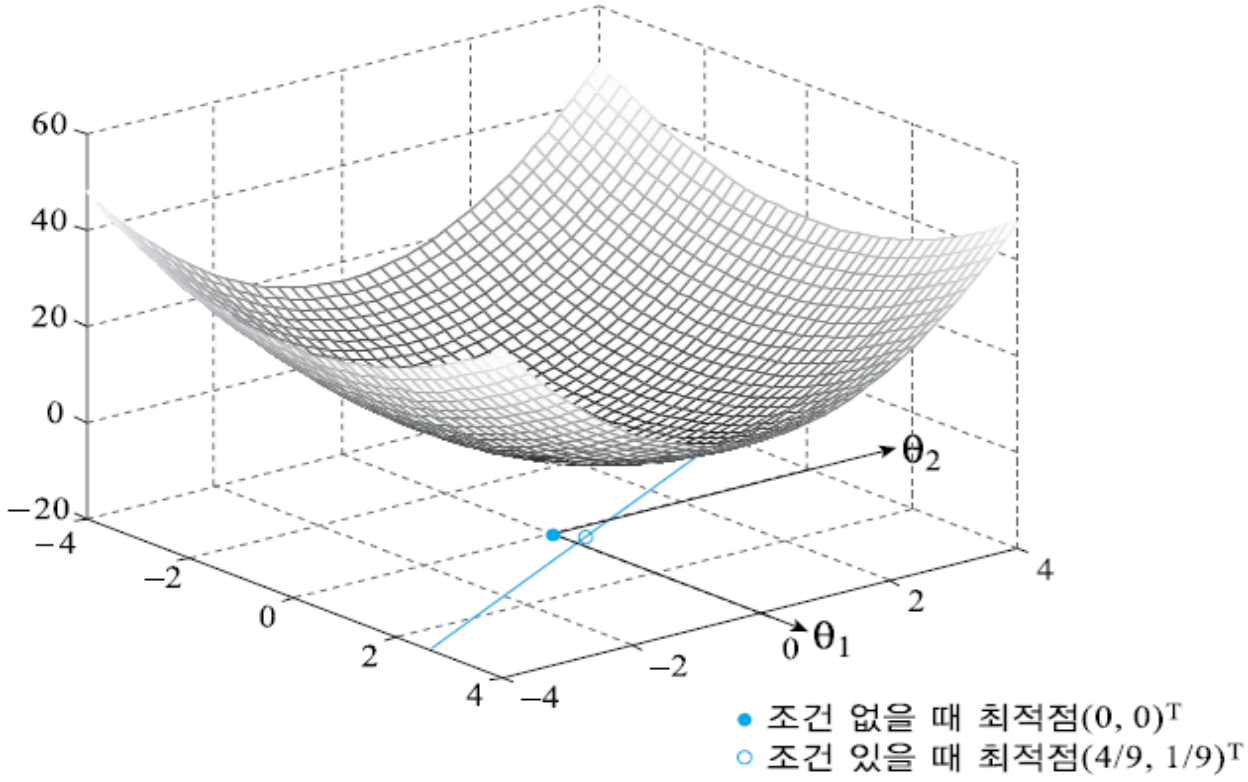


그림 11.11 등식 조건부 최적화 문제의 예

11.2.3 라그랑제 승수

- 등식 조건부 최적화 문제 equality constrained optimization problem
  - 등식 조건부 최소화 문제

아래 조건하에,

$$f_i(\boldsymbol{\theta}) = 0, i = 1, \cdots, n$$

$J(\boldsymbol{\theta})$를 최소화하는 해 $\hat{\boldsymbol{\theta}}$를 구하라. (11.5)

  - 라그랑제 함수와 라그랑제 승수
    - (11.6)은 라그랑제 함수
    - 조건식마다 라그랑제 승수 λ_i를 할당

$$L(\boldsymbol{\theta}, \boldsymbol{\lambda}) = J(\boldsymbol{\theta}) - \sum_{i=1}^{n} \lambda_i f_i(\boldsymbol{\theta}) \quad (11.6)$$

11.2.3 라그랑제 승수

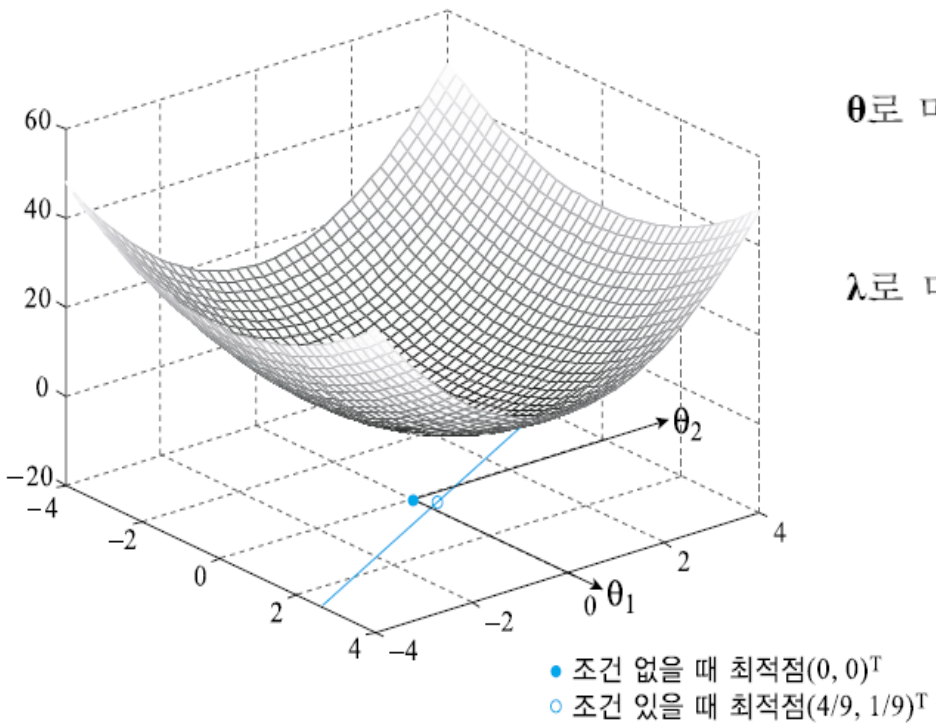
- 이제 라그랑제 함수를 최소화하는 해를 구하면 된다!
  - 결국 (11.7)을 만족하는 해를 구하면 됨

$$\left.\begin{aligned}\frac{\partial L}{\partial \theta_i} &= 0, i = 1, \cdots, k \\ \frac{\partial L}{\partial \lambda_i} &= 0, i = 1, \cdots, n\end{aligned}\right\} \qquad (11.7)$$

예제 11.3 등식 조건부 최적화 문제

- 그림 11.11의 풀이
  - '$2\theta_1+\theta_2=1$이라는 조건 하'에 $J(\boldsymbol{\theta})=\theta_1^2+2\theta_2^2$의 최소점을 구하라.

라그랑제 함수: $L(\boldsymbol{\theta},\lambda_1)=(\theta_1^2+2\theta_2^2)-\lambda_1(2\theta_1+\theta_2-1)$



$\boldsymbol{\theta}$로 미분한 식을 $\mathbf{0}$으로 둠: $\begin{cases}\dfrac{\partial L}{\partial\theta_1}=2\theta_1-2\lambda_1=0\\ \dfrac{\partial L}{\partial\theta_2}=4\theta_2-\lambda_1=0\end{cases}$

$\boldsymbol{\lambda}$로 미분한 식을 $\mathbf{0}$으로 둠: $\dfrac{\partial L}{\partial\lambda_1}=-(2\theta_1+\theta_2-1)=0$

이 식을 풀면, $\hat{\boldsymbol{\theta}}=(\frac{4}{9},\frac{1}{9})^T$

그림 11.11 등식 조건부 최적화 문제의 예

11.2.3 라그랑제 승수

- 부등식 조건부 최적화 문제 inequality constrained optimization problem
  - 부등식 조건부 최소화 문제

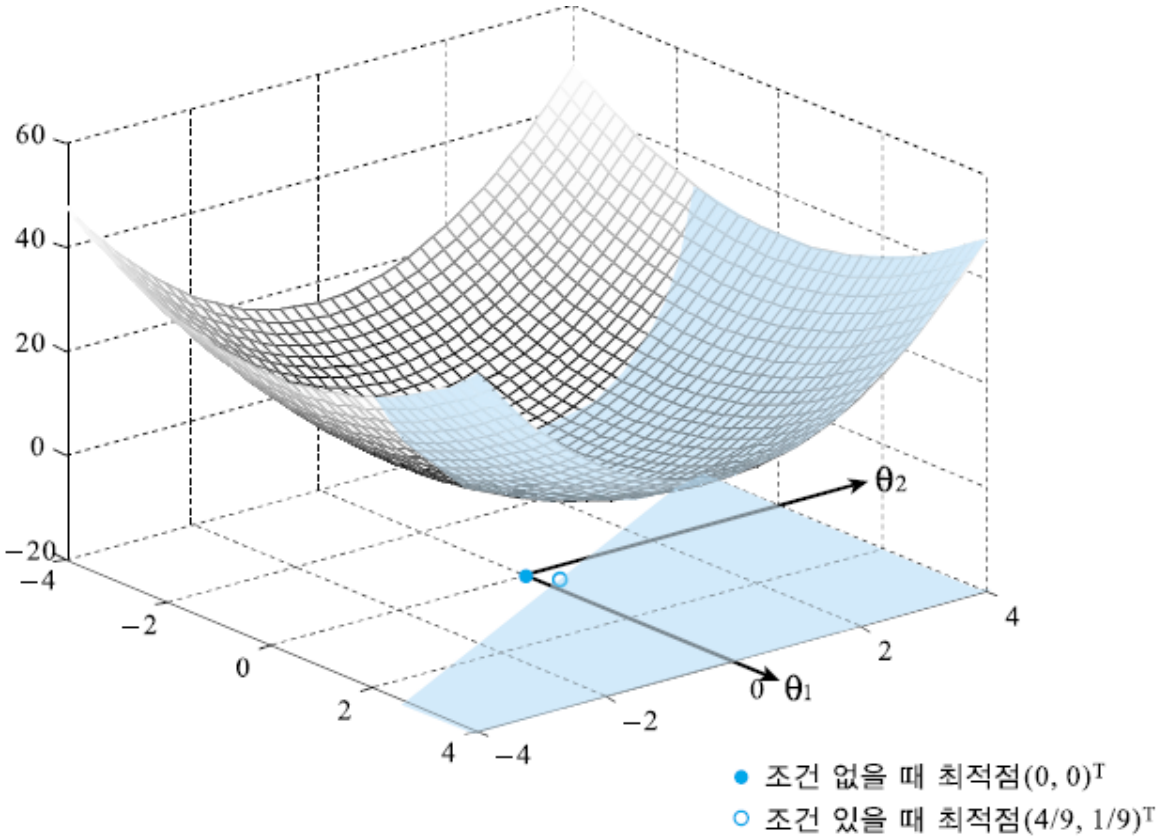
$$\left.\begin{array}{l} \text{아래 조건하에,} \\ \quad f_i(\boldsymbol{\theta}) \geq 0, i = 1, \cdots, n \\ J(\boldsymbol{\theta})\text{를 최소화하는 해 } \hat{\boldsymbol{\theta}}\text{를 구하라.} \end{array}\right\} \qquad (11.8)$$

이 문제는 Karush-Kuhn-Tucker (KKT) 조건을 사용하여 푼다. KKT 조건은 (11.9)와 같이 세 종류의 조건으로 구성된다.

$$\left.\begin{array}{l} \dfrac{\partial L(\boldsymbol{\theta}, \boldsymbol{\lambda})}{\partial \boldsymbol{\theta}} = 0 \\ \lambda_i \geq 0, i = 1, \cdots, n \\ \lambda_i f_i(\boldsymbol{\theta}) = 0, i = 1, \cdots, n \end{array}\right\}$$

예제 11.4 부등식 조건부 최적화 문제

- 그림 11.11을 부등식 조건으로 바꾸어 보면,
  - '$2\theta_1+\theta_2>1$이라는 조건 하'에 $J(\boldsymbol{\theta})=\theta_1^2+2\theta_2^2$ 의 최소점을 구하라.



(a) $f_1(\boldsymbol{\theta})=2\theta_1+\theta_2-1\geq 0$

예제 11.4 부등식 조건부 최적화 문제

- KKT 조건을 구하고 그것을 풀면,

라그랑제 함수: $L(\boldsymbol{\theta}, \lambda_1) = (\theta_1{}^2 + 2\theta_2{}^2) - \lambda_1(2\theta_1 + \theta_2 - 1)$

KKT 조건:
$$\begin{cases} \dfrac{\partial L}{\partial \theta_1} = 2\theta_1 - 2\lambda_1 = 0 \\ \dfrac{\partial L}{\partial \theta_2} = 4\theta_2 - \lambda_1 = 0 \\ \lambda_1 \geq 0 \\ \lambda_1(2\theta_1 + \theta_2 - 1) = 0 \end{cases}$$

이제 KKT 조건을 풀어 이를 만족하는 해 $\boldsymbol{\theta}$를 구하면 된다. 이 예는 라그랑제 승수를 하나만 가지므로 쉽게 풀 수 있다. 마지막 식에서 $\lambda_1 = 0$이거나 $2\theta_1+\theta_2-1 = 0$이어야 한다. 먼저 $\lambda_1 = 0$이라고 가정해 보자. 그럼 $\theta_1 = \theta_2 = 0$이 되어 주어진 조건 $f_1(\boldsymbol{\theta}) = 2\theta_1 + \theta_2 - 1 \geq 0$을 만족하지 못한다. 두 번째 경우를 가지고 풀어 보면 $\hat{\boldsymbol{\theta}} = (\frac{4}{9}, \frac{1}{9})^T$을 얻는다.

11.2.4 최적화 알고리즘과 패턴인식 문제들

- 2~10 장의 문제들에 적용된 최적화 알고리즘들을 정리해 보면,

표 11.1 패턴 인식에 나타난 다양한 최적화 문제들과 그들이 사용한 최적화 알고리즘

최적화 문제	매개 변수 θ	문제 풀이	관련 수식	알고리즘
최대 우도 (3.2절)	pdf의 매개 변수	분석적 방법	문제 (3.4) 해 (3.6)	수식 (3.6)을 계산 (pdf가 가우시언일 때)
GMM (3.4절)	$(\mu_k, \Sigma_k), \pi_k$ $1 \le k \le K$	분석적 방법과 라그랑제 승수, 내리막 경사법	문제 (3.23) 해 (3.26), (3.28), (3.29)	알고리즘 [3.3] (EM 알고리즘)
퍼셉트론 학습 (4.2절)	**w**	내리막 경사법	문제 (4.4) 해 (4.7)	알고리즘 [4.1]
다층 퍼셉트론 학습 (4.3절)	**u**, **v**	내리막 경사법	문제 (4.16) 해 (4.18), (4.19), (4.20), (4.21)	알고리즘 [4.5] (오류 역전파 알고리즘)
SVM 학습 (5장)	**w**	라그랑제 승수	문제 (5.13), (5.27), (5.35) 해 SMO 방법 등	알고리즘 [5.1]
PCA (8.5절)	고유벡터	라그랑제 승수	문제 (8.32) 해 (8.33)	알고리즘 [8.3]
특징 선택 (9.4절)	부분집합	내리막 경사법	문제 (9.5) 해 (9.7), (9.8)	알고리즘 [9.5]~[9.8]

Joseph Louis Lagrange

(1736년 1월 25일 ~ 1813년 4월 10일) 사르디니아 왕국

사르디니아는 이탈리아 반도 서쪽에 있는 섬이다. Lagrange는 그곳에서 태어났는데 그 당시는 독립된 사르디니아 왕국을 이루고 있었다. 나중에 이 왕국이 이탈리아 반도를 통일하여 이탈리아라는 나라가 탄생하게 된다. 그는 유명한 스위스 수학자 L.P. Euler의 권유로 1766년에 베를린에 있는 Prussian Academy of Science로 이적한다. 20년간 그곳에 머문 뒤 프랑스로 옮긴다. 루브르 궁에 연구실을 마련할 정도로 프랑스 정부의 대우를 받으며 왕성한 연구 활동을 펼친다. Lagrange는 조건부 최적화 문제 해결을 위한 라그랑제 승수를 Lagrange multiplier 고안하였다. 패턴 인식에서는 SVM의 풀이뿐 아니라 곳곳에서 이 방법을 활용하고 있다. 그는 수학뿐 아니라 천문학과 물리학에도 큰 공헌을 하였다. Lagrange에 대한 보다 자세한 내용은 [OConnor(웹)]을 참고하기 바란다.

[Oconnor(웹)] John J. O'Connor and Edmund F. Robertson, "Joseph Louis Lagrange," (*MacTutor History of Mathematics Archive*, http://www-history.mcs.st-andrews.ac.uk/Biographies/Lagrange.html).

11.3 시뮬레이티드 어닐링simulated annealing

- 시뮬레이티드 어닐링의 기본 원리
  - 내리막 경사법과 비슷함
  - (해를 하강시키기만 하는 내리막 경사법과 달리) 조건에 따라 해를 상승시키기도 함 (이 기능으로 지역 최저점 탈출하는 여지 생김)
  - 온도라는 뜻을 갖는 변수 T를 조절함으로써 상승 확률 조절 (초기에는 온도가 높아 상승 확률이 크지만 시간이 지남에 따라 온도를 낮추어 확률을 줄임)

알고리즘 [11.3] 시뮬레이티드 어닐링

입력: 목적 함수 $J(\boldsymbol{\theta})$

출력: 최적해 $\hat{\boldsymbol{\theta}}$

알고리즘:

1. 초기해 $\boldsymbol{\theta}$를 설정한다.
2. *score* = goodness($J(\boldsymbol{\theta})$);
3. 초기 온도 T를 설정한다.
4. *best_score* = 0;
5. **repeat** {
6. $\boldsymbol{\theta}_{new}$ = random_neighbor($\boldsymbol{\theta}$);
7. *new_score* = goodness($J(\boldsymbol{\theta}_{new})$);
8. **if** (random() < accept_level(*score*, *new_score*, T))
   { $\boldsymbol{\theta} = \boldsymbol{\theta}_{new}$; *score* = *new_score*;}
9. **if** (*new_score* > *best_score*) // 새로운 최적 점 발견
   { $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_{new}$; *best_score* = *new_score*;}
10. 온도 T를 낮춘다.
11. } **until** (stop-condition);
12. **return** $\hat{\boldsymbol{\theta}}$;

13. accept_level($s1$, $s2$, T) {
14. **if** ($s2 > s1$) **return** 1;
15. **else return** $e^{-(s1-s2)/T}$;
16. }

11.4 유전 알고리즘genetic algorithm

- 발상 !

1960년대 독일의 대학원 학생인 Rechenberg와 Schwefel은 풍동 문제를 해결하는 데 있어 미분에 기반한 방법의 한계를 느껴 새로운 방법을 고안하였다. 새로운 방법은 (11.10)의 연산을 추가로 사용하였다. 이 연산은 현재 해에 정규 분포로 얻은 임의 값을 합하는 것으로서 유전 알고리즘의 변이에 해당한다. 이 연산은 지역 최적점에서 탈출하도록 도와 준다. 이들의 방법을 진화 전략이라 evolution strategy 부른다. 이것은 유전 알고리즘의 모태가 되었다.

$$\theta_{t+1} = \theta_t + N(0, \sigma) \qquad (11.10)$$

11.4.1 원리: 내리막 경사법, 시뮬레이티드 어닐링, 유전 알고리즘

- 유전 알고리즘의 차별성
  - 해 집단을 가짐 (단일 해가 아닌 다수 해를 개선해 나가는 전략)
  - 생물 진화와 마찬가지로 교배와 돌연변이 연산을 통해 해를 개선해 나감

알고리즘 [11.6] 유전 알고리즘 (추상화 버전)

1. 초기 해 집단 P를 설정한다. // P는 여러 개의 해를 가짐
2. P의 해들을 평가하여 적합도를 부여한다.
3. **repeat** {
4. P에서 두 개의 해 θ_1과 θ_2를 선택한다. // 높은 적합도의 해가 선택 확률 높음
5. θ_1과 θ_2를 교차시키고 변이 연산을 가하여 자식 해 θ'를 얻는다.
6. θ'를 P에 대치한다.
7. } **until** (멈춤 조건);

11.4.1 원리: 내리막 경사법, 시뮬레이티드 어닐링, 유전 알고리즘

- 내리막 경사법, 시뮬레이티드 어닐링과 비교

알고리즘 [11.4] 내리막 경사법 (추상화 버전)

1. 초기해 **θ**를 설정한다.
2. **repeat** {
3. **θ**의 이웃 중에 가장 좋은 것을 **θ**로 대치한다.
4. } **until** (멈춤 조건);

알고리즘 [11.5] 시뮬레이티드 어닐링 (추상화 버전)

1. 초기해 **θ**와 초기 온도를 설정한다.
2. **repeat** {
3. **θ**의 이웃 중에 임의의 해를 선택하여 **θ'**라 한다.
4. **if** (온도에 따른 조건이 참) **θ** = **θ'**; // 온도가 높을수록 참일 확률 큼
5. 온도를 낮춘다.
6. } **until** (멈춤 조건);

11.4.1 원리: 내리막 경사법, 시뮬레이티드 어닐링, 유전 알고리즘

- 내리막 경사법, 시뮬레이티드 어닐링, 유전 알고리즘의 비유
  - 히말라야 산맥에서 가장 높은 에베레스트 봉을 찾는 문제

"오르막 경사법에서는 캥거루 한 마리가 가장 가파른 경사면만 따라 계속 산을 올라 꼭대기에 도달한다. 처음 시작한 곳이 에베레스트 봉에 속한 곳이 아니라면 절대 에베레스트 꼭대기에 도달하지 못한다. 시뮬레이티드 어닐링에서는 캥거루가 술에 취해 이리 저리 날 뛰다가 시간이 지남에 따라 술이 조금씩 깨어 가파른 경사면을 따라 올라 꼭대기에 도착한다. 시작점이 에베레스트가 아니더라도 날 뛰다가 우연히 에베레스트의 어느 곳에 도달하여 결국 에베레스트 꼭대기에 도착할 수도 있다. 유전 알고리즘에서는 비행기가 등장한다. 히말라야 산맥 곳곳에 여러 마리의 캥거루를 떨어뜨린다. 이들은 기를 쓰고 꼭대기로 올라가지 않는다. 그냥 있는 곳에서 오르락 내리락 거리며 풀을 뜯어 먹고 자식을 낳으며 산다. 몇 년에 한번씩 포수가 나타나 낮은 곳에 있는 캥거루를 잡아간다. 따라서 캥거루들이 사는 곳은 세대를 거듭하면서 높아진다."

11.4.2 유전 알고리즘의 구조

- 용어
  - 해집단population: 해의 집합
  - 적합도: 해의 품질
  - 세대
  - 자식 해
  - 유전 연산: 교차, 돌연 변이

알고리즘 [11.7] 유전 알고리즘

입력: 목적 함수 $J(\boldsymbol{\theta})$

출력: 최적해 $\hat{\boldsymbol{\theta}}$

알고리즘:

1. 초기 해 집단 P를 설정한다. // P는 여러 개의 해를 가짐
2. P의 해들의 품질을 평가한다.
3. P에서 가장 우수한 해를 $\hat{\boldsymbol{\theta}}$에 저장한다.
4. **repeat** {
5. P의 해들에게 적합도를 부여한다.
6. **for** ($i = 1$ **to** k) {
7. P에서 두 개의 해를 선택하여 이들을 $\boldsymbol{\theta}_1$과 $\boldsymbol{\theta}_2$라 한다.
8. $\boldsymbol{\theta}_i' = \text{crossover}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$; // 부모 해로부터 교차와 변이를 통해
9. mutation($\boldsymbol{\theta}_i'$); // 자식 해를 만든다.
10. $\boldsymbol{\theta}_i'$의 품질을 평가한다.
11. }
12. $\boldsymbol{\theta}_1', \cdots, \boldsymbol{\theta}_k'$를 P에 대치한다.
13. $\boldsymbol{\theta}_1', \cdots, \boldsymbol{\theta}_k'$에서 가장 우수한 해가 $\hat{\boldsymbol{\theta}}$보다 좋으면 그것을 $\hat{\boldsymbol{\theta}}$로 한다.
14. } **until** (멈춤 조건);
15. **return** $\hat{\boldsymbol{\theta}}$;

11.4.2 유전 알고리즘의 구조

- 원리
  - 라인 7의 선택
    - 적합도가 높은 해일수록 선택될 확률이 높다.
    - 하지만 못난 해도 0 이상의 확률을 갖는다.
  - 라인 8 의 교차로 만들어진 자식 해는 부모 해의 형질을 이어 받는다.
  - 라인 9의 돌연 변이는 지역 최적점을 벗어날 가능성을 제공한다. (조숙한 수렴premature convergence 방지 역할)
  - 자식 해의 품질은 해 집단의 평균 품질을 넘을 가능성이 높다. 따라서 세대를 거듭하면 해 집단의 품질은 점점 좋아진다.

11.4.2 유전 알고리즘의 구조

- 안정 상태형과steady-state 세대형generational
  - 안정 상태형: 라인 6-11에서 k=1로 두어 한 세대에 하나의 해 생산
  - 세대형: k=n으로 두어 한 세대에 해집단 전체를 대치
- 해 표현과 초기해 생성
  - 문제에 적합한 해 표현을 개발해야 한다.
    - 예) 특징 선택: 이진 열
    - 예) 신경망 훈련: 실수 코딩
  - 초기해는 보통 난수를 이용하여 생성
- 멈춤 조건
  - 수렴이 되면 멈춘다. 그런데 수렴 여부는 어떻게 알 수 있나?
  - 주어진 문제에 적합하게… 보통 아래 조건을 적절히 and 또는 or 하여 사용
    - 주어진 계산 시간이 지났다.
    - 세대가 주어진 최대 세대에 도달하였다.
    - 여러 세대에 걸쳐 더 이상 좋은 해가 발생하지 않는다.
    - 해 집단의 평균 적합도가 여러 세대에 걸쳐 더 이상 향상되지 않는다.

11.4.3 유전 연산

알고리즘 [11.7]은 여러 종류의 연산을 사용하고 있다. 유전 알고리즘을 제대로 활용하기 위해서는 그들에 대한 정확한 이해가 필수적이다. 이들 연산은 여러 가지 매개 변수를 가지는데 이들의 설정도 매우 중요하다. 매개 변수 설정은 선택 압력과 밀접한 관련을 갖는데 이에 대해서는 11.4.4절에서 구체적으로 설명할 것이다.

11.4.3 유전 연산

- 적합도 계산
  - 해 품질을 그대로 사용
    - 적절할수도 그렇지 않을수도...
    - 초기에는 우열이 두드러지지만 세대가 지남에 따라 품질 값의 차이가 미세해져 더 이상 진화가 안 일어 날 수 있음
  - 품질 비례 방법
    - f_i는 i 번째 해의 적합도. q_i는 i 번째 해의 품질

$$f_i = (q_i - q_{worst}) + (q_{best} - q_{worst})/(r-1) \qquad (11.11)$$

    - q_{best}와 q_{worst}는 각각 가장 좋은 해와 가장 나쁜 해의 품질
    - 가장 좋은 해는 가장 나쁜 해의 r 배 적합도 (r이 클수록 '선택 압력' 높다.)
  - 순위 기반 방법
    - 해를 품질에 따라 정렬하고 순위에 따라 적합도를 부여

$$f_i = q(1-q)^{i-1} \qquad (11.12)$$

    - q가 클수록 선택 압력 높다.

예제 11.5 적합도 계산

해 집단의 크기가 $n = 4$라 하고 네 개의 해가 아래와 같은 품질을 가진다고 가정하자.

$$\text{해1} = 32,\ \text{해2} = 69,\ \text{해3} = 63,\ \text{해4} = 72$$

해의 품질을 그대로 적합도로 취하는 첫 번째 방법에서는 적합도 f_i와 정규화 적합도 nf_i가 아래와 같다.

$$f_1 = 32,\ f_2 = 69,\ f_3 = 63,\ f_4 = 72$$
$$nf_1 = 0.1356,\ nf_2 = 0.2924,\ nf_3 = 0.2669,\ nf_4 = 0.3051$$

품질 비례 방법 ($r = 5$)과 순위 기반 방법 ($q = 0.3$)에 의한 적합도와 정규화 적합도는 아래와 같다.

품질 비례 방법: $f_1 = 10,\ f_2 = 47,\ f_3 = 41,\ f_4 = 50$

$$nf_1 = 0.0676,\ nf_2 = 0.3176,\ nf_3 = 0.2770,\ nf_4 = 0.3378$$

순위 기반 방법: $f_1 = 0.1029,\ f_2 = 0.21,\ f_3 = 0.147,\ f_4 = 0.30$

$$nf_1 = 0.1354,\ nf_2 = 0.2764,\ nf_3 = 0.1934,\ nf_4 = 0.3948$$

11.4.3 유전 연산

- 공유sharing
  - 해 집단의 다양성을 높이려는 목적 (다양성이 높다는 것은 탐색 공간을 골고루 살핀다는 뜻이므로 전역 최적 점에 도달할 가능성을 높여줌)
  - 원리
    - 식 (11.13)으로 서로 비슷한 해의 적합도를 낮추어 줌

$$\hat{f}_i = \frac{f_i}{\sum_{j=1}^{n} s(d_{ij})} \qquad (11.13)$$

    - d_{ij}는 해 i와 j 사이의 거리
    - $s(d)$는 d가 클수록 작아지는 함수

11.4.3 유전 연산

- 선택

선택을 위해서도 여러 방법이 있는데 모든 방법이 공유하는 기본 원칙이 있다. 그것은 우수한 해가 선택될 확률이 커야 한다는 것이다. 그리고 열등한 해도 낮은 확률이지만 선택될 기회가 주어져야 한다.

- 룰렛 휠 방법

룰렛 휠 선택:

r = random(); // [0,1] 사이의 난수

$a[i-1] < r \le a[i]$인 해 i를 선택하라.

예) $nf_1 = 0.1354,\ nf_2 = 0.2764,\ nf_3 = 0.1934,\ nf_4 = 0.3948$

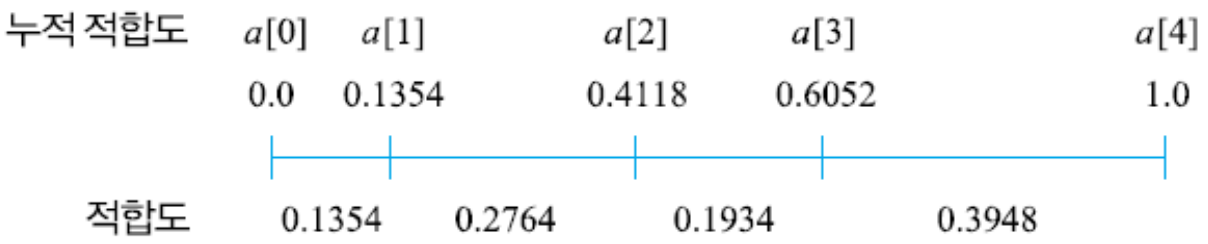


그림 11.14 룰렛 휠 방법

11.4.3 유전 연산

- 선택 (계속)
  - 토너먼트 방법

토너먼트 선택:

[1,n] 사이의 서로 다른 두 개의 정수 i_1과 i_2를 임의로 생성하고 해 i_1과 i_2를 선택한다.

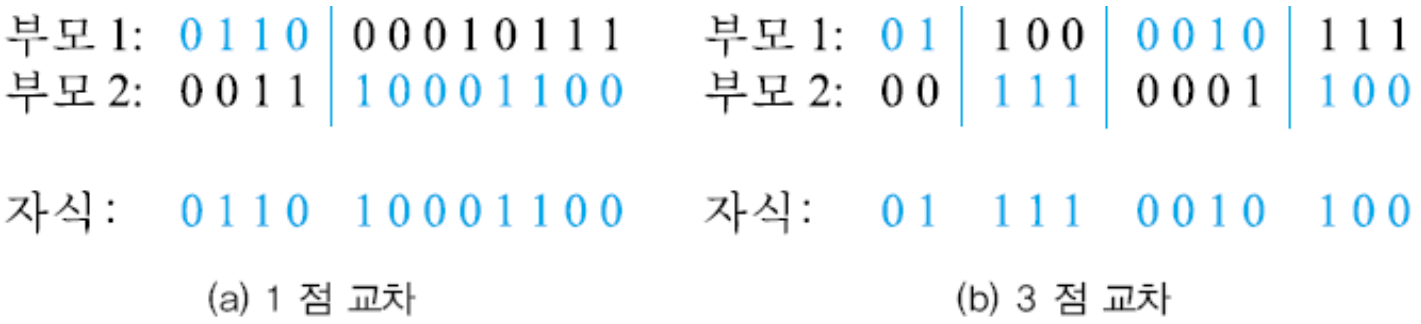
r = random(); // [0,1] 사이의 난수

if ($r < t$) i_1과 i_2 중에 우수한 것을 선택하고,

else 열등한 것을 선택한다.

11.4.3 유전 연산

- 교차 (부모해로부터 자식 해 생성. 부모 형질 이어받음)
  - 자름선 교차 (이진 염색체)



(a) 1 점 교차 (b) 3 점 교차

그림 11.15 이진 열로 표현된 염색체의 교차 연산

  - 자름선 개수가 많아지면 선택 압력은 어떻게 될까?

11.4.3 유전 연산

- 교차 (계속)
  - 균등 교차 (이진 염색체)

부모 1: 0 1 1 0 0 0 0 1 0 1 1 1
부모 2: 0 0 1 1 1 0 0 0 1 1 0 0
난수 : 0.3 0.8 0.4 0.9 0.4 0.6 0.5 0.7 0.4 0.3 0.1 0.9 ($t = 0.5$)

자식 : 0 0 1 1 0 0 0 0 0 1 1 0

그림 11.16 균등 교차

  - 산술 교차 (실수 염색체)

부모 1: 0.1 −0.5 0.3 0.7 −0.8
부모 2: 0.3 −0.3 0.5 0.9 0.6

자식 : 0.2 −0.4 0.4 0.8 −0.1

그림 11.17 산술 교차

11.4.3 유전 연산

- 변이 (부모해가 가지지 못한 형질을 부여하여 더 광범위한 공간 탐색)
  - 이진 염색체의 변이 (변이 확률 p_m과 선택 압력의 관계는?)

변이전 :	0	0	1	1	0	0	0	0	0	0	1	0
난수 :	0.30	0.89	0.45	0.01	0.42	0.63	0.54	0.70	0.48	0.11	0.18	0.93
변이후 :	0	0	1	0	0	0	0	0	0	0	1	0

그림 11.18 변이 ($p_m = 0.02$)

  - 실수 염색체의 변이

변이 전 :	0.2	−0.4	0.4	0.8	−0.1	
난수 :	0.876	0.653	0.103	0.004	0.719	(p_m=0.005)
난수 :				−0.1		
변이 후 :	0.2	-0.4	0.4	0.7	−0.1	

그림 11.19 실수 염색체의 변이

11.4.4 매개 변수 설정과 선택 압력

- 선택 압력selection pressure
  - 선택 압력이 높다는 말은 우수한 해에게 기회를 더 줌을 뜻함
  - 너무 높으면 조기 수렴의 위험. 너무 낮으면 수렴 시간이 너무 긴 문제
  - 다양성과 선택 압력은 반비례 관계이다.
- 탐사와exploitation 탐험exploration
  - 탐사는 가능성 높아 보이는 지점을 집중적으로 찾아보는 성향
  - 탐험은 탐색 공간 전체를 고루고루 찾아보는 성향
- 유전 알고리즘이 추구하는 바는
  - 탐사와 탐험의 절묘한 조화
  - 그러기 위해서는 여러 매개 변수를 조화롭게 설정해야 한다!

11.4.4 매개 변수 설정과 선택 압력

표 11.2 유전 알고리즘의 매개 변수와 선택 압력

연산 종류	매개 변수	선택 압력
해 집단	해의 개수 n이 클수록	낮음
순위 기반 적합도 계산	q가 클수록	높음
품질 비례 적합도 계산	r이 클수록	높음
토너먼트 선택	t가 클수록	높음
교차	자름 선이 많을수록	낮음
변이	p_m이 클수록	낮음
공유	사용하면	낮음
엘리티즘	사용하면	높음
대치	가장 나쁜 해를 대치하면	높음

11.4.5 찬반 논쟁

- 찬
  - 새로운 개념과 연산들이 재미있다.
  - 기존 알고리즘의 한계를 극복할 것 같다.
  - 시간을 더 주면 더 좋은 해를 기대할 수 있다.
  - 문제에 대한 제약 조건이 적고 응용 범위가 넓다.
  - 실제 실험해 보니 성능이 기존 알고리즘보다 좋다.
- 반
  - 수학적 토대가 약하여 미덥지 못하다.
  - 수행할 때마다 다른 해를 내면 어떡하나?
  - 수행 시간이 길어 실시간 환경에서는 활용이 불가능하다.
  - 실제 실험해 보니 성능이 기존 알고리즘보다 나쁘다.