

# A Survey of Optimization Methods froma Machine Learning Perspective

## 출처

<https://ieeexplore.ieee.org/document/8903465/>

---

## 정리

## 제목

### **A Survey of Optimization Methods froma Machine Learning Perspective**

Shiliang Sun, Zehui Cao, Han Zhu, and Jing Zhao

---

---

## 초록

## 번역

기계학습은 빠르게 발전하며, 이로 인해 많은 이론적 돌파구가 마련되고 다양한 분야에 널리 적용되고 있다. 최적화(Optimization)는 기계학습의 중요한 부분으로, 연구자들의 많은 주목을 받고 있다. 데이터 양의 기하급수적 성장과 모델 복잡도의 증가로 인해, 기계학습에서의 최적화 방법은 점점 더 많은 도전에 직면하고 있다. 기계학습에서 최적화 문제를 해결하거나 최적화 방법을 개선하기 위한 많은 연구가 연이어 제안되었다. 기계학습 관점에서의 최적화 방법에 대한 체계적 회고와 요약은 최적화와 기계학습 연구의 발전 모두에 중요한 지침을 제공할 수 있다. 본 논문에서는 먼저 기계학습의 최적화 문제를 설명한다. 이어서 일반적으로 사용되는 최적화 방법의 원리와 발전을 소개한다. 그 다음 일부 인기 있는 기계학습 분야에서의 최적화 방법의 응용과 발전을 요약한다. 마지막으로 기계학습 최적화에서의 도전과 개방형 문제를 탐구하고 제시한다.

색인어—기계학습, 최적화 방법, 심층 신경망(Deep Neural Network), 강화학습(Reinforcement Learning), 근사 베이지안 추론(Approximate Bayesian Inference).

---

## 내용

- **최적화(Optimization)** : 데이터의 양과, 모델 복잡도의 증가로 인해 최적화 문제를 해결하거나, 개선하기 위한 많은 연구가 제안되었음
1. 최적화 문제설명
  2. 최적화 방법의 원리와 발전
  3. 인기 있는 최적화 방법의 응용과 발전
  4. 최적화 문제에서의 도전점과 개방된 문제를 탐구하고 제시
- 

## 포인트

이 논문은 기계학습에서 최적화(Optimization)의 위치를 총정리한 서베이 논문

---

---

## 📌 서론 & 결론 & 고찰

### 번역

#### 서론

최근 기계학습은 놀라운 속도로 성장하며 수많은 연구자와 실무자를 끌어들이고 있다. 기계학습은 기계번역, 음성인식, 이미지 인식, 추천 시스템 등 다양한 분야에서 중요한 역할을 하고 있으며 가장 인기 있는 연구 방향 중 하나가 되었다. 최적화(Optimization)는 기계학습의 핵심 구성 요소 중 하나이다. 대부분의 기계학습 알고리즘의 본질은 최적화 모델을 세우고, 주어진 데이터로부터 목적함수(Objective function)의 매개변수를 학습하는 것이다.

방대한 데이터 시대에 수치적 최적화 알고리즘의 효과성과 효율성은 기계학습 모델의 보급과 활용에 큰 영향을 미친다. 기계학습의 발전을 촉진하기 위해 다양한 효과적인 최적화 방법들이 제안되었고, 이는 성능과 효율성을 향상시켰다.

최적화에서 기울기 정보 관점에서 볼 때, 널리 쓰이는 방법은 세 가지로 나뉜다: (1) **1차 최적화(First-order optimization)**: 확률적 경사하강법(Stochastic Gradient Descent, SGD)과 그 변형, (2) **고차 최적화(High-order optimization)**: 대표적으로 뉴턴 방법(Newton's method), (3) **도함수 비의존 최적화(Derivative-free optimization)**: 대표적으로 좌표 하강법(Coordinate Descent).

1차 최적화의 대표적인 확률적 경사하강법(SGD) [1], [2]과 그 변형들은 최근 널리 쓰이고 빠른 속도로 진화하고 있다. 하지만 많은 사용자들은 이러한 방법들의 특성이나 적용 범위를 깊이 고려하지 않고, 단순히 블랙박스 최적화기로 사용하는 경우가 많아 성능 발휘에 한계가 있을 수 있다. 따라서 본 논문에서는 이러한 근본적 최적화 방법들을 종합적으로 소개한다. 특히 장단점, 적용 범위, 파라미터의 특성을 체계적으로 설명하여,

사용자가 적절한 1차 최적화 방법을 선택하고 학습 과정에서 더 합리적으로 파라미터를 조정할 수 있도록 돋고자 한다.

1차 방법에 비해 고차 방법 [3], [4], [5]은 곡률 정보(curvature information)를 활용하여 탐색 방향을 더 효과적으로 만들기 때문에 더 빠른 속도로 수렴한다. 그러나 고차 최적화는 연산 및 Hessian 역행렬의 저장에서 더 많은 도전에 직면한다. 이를 해결하기 위해 뉴턴 방법을 기반으로 한 다양한 변형법이 제안되었으며 [6], [7], 이후 연구에서는 확률적 준-뉴턴(Stochastic Quasi-Newton) 방법과 그 변형들이 대규모 데이터로 확장되었다 [8], [9], [10].

도함수 비의존 최적화 방법 [11], [12]은 목적함수의 도함수가 존재하지 않거나 계산이 어려울 때 주로 사용된다. 이 방법에는 크게 두 가지 아이디어가 있다. 하나는 경험적 규칙에 기반한 휴리스틱 탐색, 다른 하나는 샘플을 사용해 목적함수를 근사하는 것이다. 또한 이러한 방법은 경사 기반 방법과 함께 사용될 수도 있다.

대부분의 기계학습 문제는 한 번 수식화되면 최적화 문제로 풀 수 있다. 심층신경망(Deep Neural Network), 강화학습(Reinforcement Learning), 메타 학습(Meta Learning), 변분추론(Variational Inference), 마르코프 연쇄 몬테카를로(Markov Chain Monte Carlo) 등 각 분야는 서로 다른 난제와 도전에 직면하며, 그 과정에서 개발된 최적화 기법들이 일반 최적화 방법 발전에도 영감을 줄 수 있다.

심층신경망(DNNs)은 패턴 인식과 기계학습에서 큰 성공을 거두었다. 특히 두 가지 신경망 구조, 즉 합성곱 신경망(Convolutional Neural Networks, CNNs) [13]과 순환 신경망(Recurrent Neural Networks, RNNs)은 여러 분야에서 중요한 역할을 한다. CNN은 합성곱 연산을 사용하는 피드포워드 신경망으로 이미지 처리 [14], [15], 비디오 처리 [16], 자연어 처리(NLP) [17], [18]에 널리 사용된다. RNN은 순차 모델로 NLP [19], [20], [21], [22]뿐만 아니라 이미지 처리 [23], [24], 비디오 처리 [25] 분야에서도 활발히 사용된다. 제약 최적화(constrained optimization)에서도 RNN은 뛰어난 결과를 보인다 [26], [27], [28], [29]. 이러한 연구에서는 RNN 가중치 파라미터가 해석적 방법으로 학습되며, 상태해(solution trajectory)에 따라 최적해를 찾을 수 있다.

확률적 경사 기반 알고리즘들은 DNN에서 널리 사용되지만 [30], [31], [32], [33], 여러 문제점들이 나타난다. 예를 들어, 일부 적응형 방법(adaptive methods)에서는 학습 후반기에 학습률이 요동하며 수렴 실패(non-converging) 문제를 일으킬 수 있다 [34], [35]. 따라서 수렴 속도를 개선하기 위해 분산 감소(variance reduction)를 기반으로 한 추가 알고리즘이 제안되었다 [36], [37]. 또한 SGD와 그 변형의 특성을 결합하는 방향도 고려된다. 특히, 적응형 알고리즘에서 SGD로 전환하는 방식은 알고리즘의 정확성과 수렴 속도를 개선할 수 있다 [38].

강화학습(RL)은 에이전트가 시행착오적 상호작용을 통해 환경과 학습하며 누적 보상을 최대화하는 최적 정책(optimal policy)을 학습하는 기계학습의 한 분야이다 [39]. 심층 강화학습(Deep RL)은 RL과 심층학습을 결합하여 에이전트가 환경을 잘 인식할 수 있도록 한다. 최근 연구는 심층학습이 RL 문제의 표현 학습(representation learning)에 효과적으로 적용될 수 있음을 보여주었다 [40], [41], [42], [43], [44]. RL과 심층 RL 모델에서도 확률적 최적화 알고리즘이 널리 사용된다.

메타 학습(Meta Learning) [45], [46] 역시 최근 기계학습 분야에서 매우 인기 있는 연구 주제다. 목표는 적은 샘플만으로도 새로운 환경에 효율적으로 적응할 수 있는 모델을 설계하는 것이다. 지도학습 내에서의 메타 학습은 소수 샷 학습(few-shot learning) 문제를 해결할 수 있다 [47]. 일반적으로 메타 학습 방법은 세 가지

로 요약된다 [48]: 유사도 기반(**metric-based**) [49], [50], [51], [52], 모델 기반(**model-based**) [53], [54], 최적화 기반(**optimization-based**) [55], [56], [47]. 이 중 최적화 기반 메타 학습은 이후 절에서 자세히 설명한다.

변분추론(Variational Inference)은 베이지안 기계학습에서 사후분포를 근사하는 유용한 방법으로, 최적화 문제로 볼 수 있다. 예를 들어, mean-field 변분추론은 좌표 상승법(coordinate ascent)을 사용하여 이 최적화 문제를 해결한다 [57]. 데이터가 계속 증가하는 상황에서는 전통적 최적화 방식이 비효율적이기에, 확률적 변분추론(Stochastic Variational Inference, SVI)이 제안되었다. 이는 자연 그래디언트(Natural Gradient)를 도입하여 변분추론을 대규모 데이터에 확장했다 [58].

최적화 방법은 여러 기계학습 분야에 큰 영향을 미쳤다. 예를 들어, [5]에서는 Adam 최적화 [33]을 사용한 Transformer 네트워크를 제안하여 기계번역에 적용하였다. [59]에서는 Adam으로 최적화한 초해상도 생성적 적대 신경망(Super-Resolution GAN)을 제안했다. [60]에서는 신뢰 영역 최적화(Trust Region Optimization)를 활용한 Actor-Critic 방법을 통해 Atari 게임 및 MuJoCo 환경에서 심층 RL 문제를 해결했다.

확률적 최적화는 마르코프 연쇄 몬테카를로(MCMC) 샘플링에도 적용되어 효율성을 개선할 수 있다. 대표적으로 확률적 그래디언트 해밀토니안 몬테카를로(Stochastic Gradient Hamiltonian Monte Carlo, SGHMC) [61] 방법이 있다. 여기서 확률적 그래디언트는 대규모 샘플 처리 시 그래디언트 업데이트 속도를 가속화한다. 이때 도입되는 노이즈는 가우시안 노이즈와 마찰항(friction terms)으로 특성화된다. 또한 마찰항은 HMC 이산화(discretization)로 인한 편차를 제거해 메트로폴리스-헤이스팅스(Metropolis-Hastings) 단계를 생략할 수 있게 한다. HMC의 하이퍼파라미터 설정은 모델 성능에 영향을 주며, 자동 조정 기법이 제안되어 샘플러 성능이 향상되었다.

최적화의 발전은 기계학습 진보에 많은 기여를 했다. 그러나 여전히 많은 도전과 개방 문제들이 존재한다. 1) 심층신경망 학습에서 데이터 부족 상황에서의 최적화 성능 개선, 2) 순차모델 학습에서 미니배치 분할로 인한 편차 분석 및 보정, 3) 고차 그래디언트 정보를 SVI에 적용하는 방법, 4) 확률적 기법을 공액경사법(Conjugate Gradient)에 도입해 강력한 최적화 알고리즘을 개발하는 방안 등이 주요 과제이다.

이 논문의 목적은 기계학습 관점에서 고전 및 현대 최적화 방법을 요약·분석하는 것이다. 나머지 구조는 다음과 같다. II장: 최적화 관점에서 본 기계학습 문제, III장: 고전 최적화 알고리즘 및 최신 발전, IV장: 특정 기계학습 분야의 최적화 응용, V장: 도전과 개방 문제, VI장: 결론.

---

## 결론

본 논문은 기계학습 관점에서 자주 사용되는 최적화 방법을 소개하고 요약하며, 다양한 기계학습 분야에서의 응용을 검토하였다. 먼저 1차, 고차, 도함수 비의존적 관점에서의 최적화 방법의 이론적 기초와 최근 연구 진전을 설명하였다. 이어서 다양한 기계학습 시나리오에서의 최적화 방법의 응용과 성능 개선 접근법을 기술하였다. 마지막으로 기계학습 최적화 방법의 여러 도전과 개방 문제들을 논의하였다.

---

---

# 내용

## 서론

- 최적화는 기계학습의 핵심중의 하나이다. ML의 핵심은 최적화 모델을 세우고, 주어진 데이터로부터 **목적 함수(Objective function)**의 매개변수를 학습

즉 기계학습의 본질은 목적함수의 최적화이며 성능과 속도를 제어한다.

- **기울기(Gradient)**관점에서의 최적화
  - **1차 최적화(First-order optimization)** : 확률적 경사하강법(Stochastic Gradient Descent, SGD)과 그 변형, etc.
    - 널리 쓰이고, 빠른 속도로 진화하고 있지만, 사용자들은 특성이나, 적용범위를 고려하지 않고, 단순히 블랙박스 **optimizer**로 사용하는 경우가 많아 성능 한계에 직면할 수 있다.
  - **고차 최적화(High-order optimization)** : 뉴턴 방법(Newton's method), etc.
    - 곡률 정보(**curvature information**)를 활용해, 탐색방향을 효과적으로 찾아 더 빠른 속도로 수렴한다. 그러나 연산 및 **Hessian** 역행렬의 저장에 큰 어려움이 있다. 이를 해결하기 위해 뉴턴 방법을 기반으로 다양한 변형 및 확률적 준-뉴턴(Stochastic Quasi-Newton) 방법과 그 변형이 개발되었다.
  - **도함수 비의존 최적화(Derivative-free optimization)** : 좌표 하강법(Coordinate Descent), etc.
    - 목적함수의 도함수가 존재하지 않거나, 계산이 어려울때 사용한다. 경험적 규칙에 기반한 휴리스틱 탐색, 다른 하나는 샘플을 사용해 목적함수를 근사하는 두가지 방법을 주로 사용한다. 추가적으로 이러한 방법은 경사하강법과 함께 사용될 수 있다.
- 대부분의 ML문제는 한번 수식화되면 최적화 문제로 풀 수 있다.
  - 심층신경망(Deep Neural Network), 강화학습(Reinforcement Learning), 메타 학습(Meta Learning), 변분추론(Variational Inference), 마르코프 연쇄 몬테카를로(Markov Chain Monte Carlo) 등 각 분야는 서로 다른 난제와 도전에 직면하며, 그 과정에서 개발된 최적화 기법들이 일반 최적화 방법 발전에도 영감을 줄 수 있다.
  - **심층 신경망(Deep Neural Network, DNN)** : 두가지 주요한 신경망인 CNN, RNN은 여러분야에서 주요한 역할을 한다.
    - CNN : 이미지(비디오)처리 뿐만아니라 NLP에서도 뛰어난 효과를 보임
    - RNN : 제약 최적화나 이미지(비디오)처리 분야에서도 뛰어난 효과를 보임
    - **확률적 경사하강법(Stochastic Gradient Descent, SGD)**이 주로 사용되지만 문제점이 발생한다.
      - 일부 적응형 방법(adaptive methods)에서는 학습 후반에 학습률이 요동쳐서 수렴에 실패할 수 있다.
      - 수렴 속도를 개선하기 위해 분산 감소(variance reduction)를 기반으로 한 추가 알고리즘이 제안되었다.

- 적응형 알고리즘을 SGD와 변환하면 수렴속도와 정확도를 높일 수 있다.
- **강화학습(Reinforcement Learning, RL)** : agent가 환경과 상호작용하여 누적 보상을 최대화하는 최적 정책(Optimal Policy)를 학습
  - 요즘은 DL과 합쳐져서 사용됨 : 표현 학습(representation learning)에 효과적으로 적용된다.
- **메타학습(Meta Learning)** : 적은 샘플로 새로운 환경에 효율적으로 적응할 수 있는 모델을 만드는 것
  - 지도학습의 few shot 문제를 해결 가능
  - 유사도 기반(metric-based), 모델 기반(model-based), 최적화 기반(optimization-based)
- **변분추론(Variational Inference)** : 베이지안 ML에서 사후분포를 근사하는 방법으로 최적화 문제로 볼 수 있다.
  - mean-field 변분추론은 좌표 상승법(coordinate ascent)을 사용하여 이 최적화 문제를 해결
  - 데이터가 계속 증가하는 상황에서는 전통적 최적화 방식이 비효율적이기에, 확률적 변분추론 (Stochastic Variational Inference, SVI)이 제안
- 최적화 기법은 ML에 많은 영향을 미쳤다. Adam의 적용, 신뢰 영역 최적화(Trust Region Optimization)를 통한 Actor-Critic 방법이 그 예이다.
- 확률적 최적화는 몬테카를로에도 적용시킬 수 있으며, 확률적 그래디언트 해밀토니안 몬테카를로 (Stochastic Gradient Hamiltonian Monte Carlo, SGHMC)이 그 예이다.
- 이렇게 최적화 기법은 ML의 발전에 많은 기여를 하였으나, 아래와 같은 문제에 직면해있다.
  - 심층신경망 학습에서 데이터 부족 상황에서의 최적화 성능 개선
  - 순차모델 학습에서 미니배치 분할로 인한 편차 분석 및 보정
  - 고차 그래디언트 정보를 SVI에 적용하는 방법
  - 확률적 기법을 공액경사법(Conjugate Gradient)에 도입해 강력한 최적화 알고리즘을 개발하는 방안 등

### Note

#### 제약 최적화(Constrained Optimization)

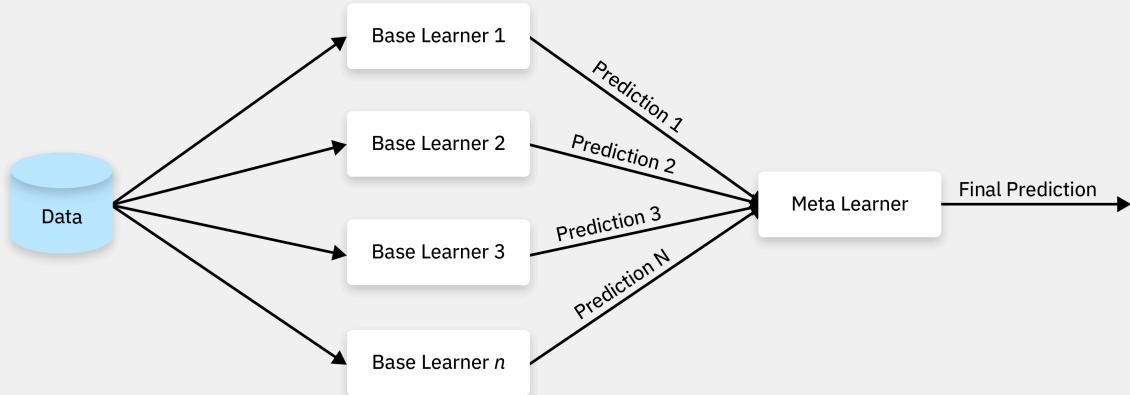
해당 변수에 제약 조건이 있는 경우 일부 변수에 대한 목적 함수를 최적화하는 프로세스이다. RNN의 경우 해석적 방법으로 학습되며, 상태해(solution trajectory)에 따라 최적해를 구한다.

### Note

#### 메타학습(Meta Learning)

**메타** : 한 단계위의 것을 가르킬 때 사용, 데이터의 경우 데이터에 대한 데이터인 메타데이터가 그 예이다.

스스로 학습하는 방법(**learning to learn**)을 배우는 과정을 말한다. 일반적인 머신러닝 모델이 특정 데이터셋을 가지고 '문제 푸는 법'을 직접 배운다면, 메타학습 모델은 여러 다양한 학습 경험을 통해 '새로운 문제를 더 빠르고 효율적으로 푸는 법' 자체를 터득한다.



- **내부 루프 (Inner Loop)**: 개별 '태스크' 내에서 모델이 빠르게 학습을 진행합니다. 예를 들어, '고양이 사진 5장으로 고양이 구분하기'라는 태스크가 주어지면, 모델은 이 5장의 데이터에 맞춰 빠르게 파라미터를 업데이트합니다.
- **외부 루프 (Outer Loop)**: 여러 태스크에 대한 내부 루프의 학습 결과를 종합하여, '어떻게 하면 새로운 태스크를 더 잘 배울 수 있을지'에 대한 **메타 지식(meta-knowledge)**을 업데이트합니다. 이 메타 지식은 보통 모델의 초기 가중치(initial weights) 형태로 나타납니다.

### Note

#### 변분추론

복잡해서 직접 계산하기 어려운 확률 분포를 다루기 쉬운 간단한 근사(approximate) 분포로 대체하여 문제를 해결하는 기법으로 간단하게 "복잡한 분포를 단순한 분포로 근사"하여 문제를 푸는 것

### Note

#### 신뢰 영역 최적화(Trust Region Optimization)를 통한 Actor-Critic

## 액터-크리틱(Actor-Critic)

- **액터 (Actor, 배우):** 현재 상태에서 어떤 행동을 할지 결정하는 정책(Policy)을 담당합니다. 목표는 보상을 최대로 받는 것입니다.
- **크리틱 (Critic, 평론가):** 액터가 한 행동이 얼마나 좋았는지를 평가하는 가치 함수(Value Function)를 담당합니다. 이 평가 결과(주로 어드밴티지 함수, Advantage Function

## TPRO

신뢰 영역의 크기는 이전 정책과 새 정책의 차이가 일정 수준을 넘지 않도록 하는 제약 조건에 해당하며, 정책의 '차이'를 측정하는 거리 단위로는 단순한 파라미터 차이가 아닌, 두 정책의 행동 분포 차이를 나타내는 **KL-Divergence** (쿨백-라이블러 발산)를 사용하여 문제를 처리 쉬운 예를 들자면 안개낀 등산으로 비유한다면

- 등산가의 현재 위치는 현재 정책( $\pi_{old}$ )
- 정상으로 가는 가장 가파른 길은 보상이 최대가 되는 방향
- 하지만 안개가 짙게 껴서 멀리 내다볼 수 없으므로, 현재 위치에서 계산한 경사가 먼 곳에서도 유효 할지 알 수 없다
- 따라서 등산가는 자신의 주변에 안전하다고 믿는 작은 원(신뢰 영역)을 그리고, 그 원 안에서 고도를 가장 많이 높일 수 있는 지점으로 한 발짝만 내딛는다

### Note

## 몬테카를로

무수히 많은 난수(random numbers)를 생성하고, 반복적인 시뮬레이션을 통해 원하는 값의 근사치를 얻어내는 계산 기법

## 결론

(생략)

## 포인트

항목	내용
최적화 분류	1차(경사하강 기반), 고차(뉴턴/준-뉴턴), 도함수 비의존(좌표 하강 등)
응용 분야	DNN, 강화학습, 메타학습, 변분추론, MCMC
기여	방법론적 정리, 응용 사례, 최신 발전, 오픈 문제 정리
핵심 메시지	기계학습의 효율과 성능은 최적화 발전에 크게 의존

## 🔬 실험과정

## 📚 2. MACHINE LEARNING FORMULATED AS OPTIMIZATION

### 번역

거의 모든 머신러닝 알고리즘은 목적함수(**objective function**)의 극값을 찾는 최적화 문제로 정식화될 수 있다. 모델을 구축하고 합리적인 목적함수를 정의하는 것이 머신러닝 방법의 첫 단계이다. 목적함수가 정해지면, 보통 수치적(numerical) 또는 해석적(analytical) 최적화 방법을 사용하여 해당 최적화 문제를 해결한다.

모델링 목적과 해결하고자 하는 문제에 따라, 머신러닝 알고리즘은 지도학습(supervised learning), 반지도 학습(semi-supervised learning), 비지도학습(unsupervised learning), 강화학습(reinforcement learning)으로 구분된다. 특히 지도학습은 다시 분류(classification) 문제(예: 문장 분류(sentence classification) [17], [63], 이미지 분류(image classification) [64], [65], [66] 등)와 회귀(regression) 문제로 나뉜다. 비지도학습은 클러스터링(clustering)과 차원 축소(dimension reduction) [67], [68], [69] 등으로 나뉜다.

### A. Optimization Problems in Supervised Learning

지도학습의 목표는 훈련 샘플의 손실함수를 최소화하는 최적의 매핑 함수  $f(x)$ 를 찾는 것이다.

$$\min \frac{1}{N} \sum L(y_i, f(x_i, \theta)),$$

여기서  $N$ 은 훈련 샘플의 개수,  $\theta$ 는 매핑 함수의 매개변수,  $x_i$ 는  $i$ 번째 샘플의 특징 벡터(feature vector),  $y_i$ 는 그에 대응하는 레이블(label),  $L$ 은 손실 함수(loss function)이다.

지도학습에는 여러 종류의 손실 함수가 있다. 예를 들어 유clidean 거리 제곱(square of Euclidean distance), 교차 엔트로피(crossentropy), 대조 손실(contrast loss), 힌지 손실(hinge loss), 정보 이득 (information gain) 등이 있다. 회귀(regression) 문제에서는 가장 단순하게 유clidean 거리 제곱을 손실 함수로 사용하여, 훈련 샘플의 제곱 오차를 최소화한다. 그러나 이러한 경험적 손실(empirical loss)은 반드시 일반화 성능(generalization performance)이 좋은 것은 아니다. 또 다른 전형적인 형태는 구조적 위험 최소

화(Structured Risk Minimization)이며, 그 대표적인 방법은 서포트 벡터 머신(Support Vector Machine)이다.

목적 함수에는 보통 정규화 항(regularization item)을 추가하여 과적합(overfitting)을 완화한다. 예를 들어 L2 노름(norm)을 사용하면,

$$\min \sum L(y_i, f(x_i, \theta)) + \lambda \|\theta\|_2^2$$

와 같이 쓸 수 있다. 여기서  $\lambda$ 는 절충 파라미터(compromise parameter)이며, 교차 검증(cross-validation)을 통해 결정될 수 있다.

## B. Optimization Problems in Semi-supervised Learning

반지도학습(SSSL)은 지도학습과 비지도학습의 중간에 위치한 방법으로, 학습 과정에서 레이블이 있는 데이터(labeled data)와 레이블이 없는 데이터(unlabeled data)를 모두 활용한다.

이 방법은 분류(classification) [70], [71], 회귀(regression) [72], 군집(clustering) [73], [74], 차원 축소(dimensionality reduction) [75], [76] 등 다양한 작업을 처리할 수 있다. 반지도학습에는 자기학습(self-training), 생성 모델(generative models), 반지도 서포트 벡터 머신(S3VM) [77], 그래프 기반 방법(graph-based methods), 다중 학습(multi-learning) 방식 등이 있다. 여기서는 반지도학습의 최적화를 소개하기 위해 S3VM을 예시로 든다.

S3VM은 이진 분류(binary classification) 문제를 다룰 수 있는 학습 모델이며, 이 문제에서 학습 집합의 일부만 레이블이 부여되어 있다. 레이블이 있는 데이터를  $D_l = \{(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l)\}$ 로, 레이블이 없는 데이터를  $D_u = \{x_{l+1}, x_{l+2}, \dots, x_N\}$ 로 나타낼 수 있으며,  $N = l + u$ 이다. 레이블이 없는 데이터의 정보를 활용하기 위해, 원래 SVM의 목적함수에 슬랙 변수(slack variable)  $\zeta_i$ 와 함께 추가적인 제약을 도입한다. 구체적으로, unlabeled 데이터의 실제 레이블이 양성(positive)일 경우의 오분류(error)를  $q_j$ , 음성(negative)일 경우의 오분류(error)를  $z_j$ 라 정의한다. 제약식은  $\sum_{j=l+1}^N \min(q_j, z_j)$ 를 가능한 작게 만드는 것을 의미한다. 따라서, S3VM 문제는 다음과 같이 표현된다.

$$\min \|\omega\| + C [\sum \zeta_i + \sum \min(q_j, z_j)]$$

제약 조건:

$$\begin{aligned} y_i (\omega \cdot x_i + b) + \zeta_i &\geq 1, \quad \zeta_i \geq 0, \quad i = 1, \dots, l, \\ \omega \cdot x_j + b + q_j &\geq 1, \quad q_j \geq 0, \quad j = l + 1, \dots, N, \\ -(\omega \cdot x_j + b) + z_j &\geq 1, \quad z_j \geq 0. \end{aligned}$$

여기서  $C$ 는 패널티 계수(penalty coefficient)이다. S3VM의 최적화 문제는 혼합정수(mixed-integer) 문제로 다루기 어렵다 [78]. 이 문제를 다루기 위해 [79]에서는 다양한 방법들을 요약하고 있으며, 대표적으로 분기 한계법(branch and bound techniques) [80], 볼록 완화(convex relaxation) [81] 방법 등이 있다.

## C. Optimization Problems in Unsupervised Learning

클러스터링(clustering) 알고리즘 [67], [82], [83], [84]은 샘플 집합을 여러 클러스터로 나누어, 같은 클러스터에 속한 샘플들 간의 차이는 최대한 작게 하고, 다른 클러스터 간의 차이는 최대한 크게 만드는 것을 목표로 한다. k-means 클러스터링 알고리즘의 최적화 문제는 다음 손실함수를 최소화하는 것으로 정식화된다:

$$\min \sum \sum \|x - \mu_k\|^2_2,$$

여기서 K는 클러스터 개수, x는 샘플의 특징 벡터(feature vector),  $\mu_k$ 는 클러스터 k의 중심(center),  $S_k$ 는 클러스터 k에 속한 샘플 집합이다. 이 목적함수의 의미는 모든 클러스터의 분산 합을 최소화한다는 것이다.

차원 축소(dimensionality reduction) 알고리즘은 데이터를 저차원 공간으로 사영(projection)하더라도 원래의 정보를 최대한 보존하는 것을 보장한다. 주성분 분석(Principal Component Analysis, PCA) [85], [86], [87]는 차원 축소 방법의 대표적인 알고리즘이다. PCA의 목적은 재구성 오차(reconstruction error)를 최소화하는 것으로, 다음과 같이 정식화된다:

$$\min \sum \|x_i - \hat{x}_i\|^2_2, i=1\dots N$$

$$\text{여기서 } \hat{x}_i = \sum_{j=1}^D z_{ij} e_j, D \gg D'.$$

N은 샘플 개수를 나타내며,  $x_i$ 는 D차원 벡터,  $\hat{x}_i$ 는  $x_i$ 의 재구성 값이다.  $z'_i = \{z_{1i}, \dots, z_{Di}\}$ 는  $x_i$ 의  $D'$ 차원 좌표에서의 사영이며,  $e_j$ 는  $D'$ 차원 좌표계의 표준 직교 기저(standard orthogonal basis)이다.

확률적(probabilistic) 모델에서 또 다른 일반적인 최적화 목표는 훈련 샘플의 로그 우도 함수(log-likelihood function, MLE)를 최대화하는 확률 밀도 함수  $p(x)$ 를 찾는 것이다:

$$\max \sum \ln p(x_i ; \theta), i=1\dots N.$$

베이지안(Bayesian) 방법의 틀에서는  $\theta$ 에 대해 사전 분포(prior distribution)를 가정하는 경우가 많으며, 이는 과적합(overfitting)을 완화하는 효과를 가진다.

## D. Optimization Problems in Reinforcement Learning

강화학습 [42], [88], [89]은 지도학습과 비지도학습과 달리, 환경에 따라 출력이 달라지는 최적의 전략 함수(strategy function)를 찾는 것을 목표로 한다. 결정적(deterministic) 전략에서는 상태 s로부터 행동 a로의 매핑 함수가 학습 대상이며, 확률적(stochastic) 전략에서는 각 행동이 실행될 확률이 학습 대상이 된다. 각 상태에서 행동은  $a = \pi(s)$ 로 결정되며,  $\pi(s)$ 는 정책 함수(policy function)이다.

강화학습의 최적화 문제는 정책 함수에 의해 결정된 일련의 행동을 실행한 뒤 얻는 누적 보상(cumulative return)을 최대화하는 것으로 정식화된다:

$$V\pi(s) = E [\sum_{k=0}^{\infty} \gamma^k r_{t+k} | S_t = s],$$

여기서  $V\pi(s)$ 는 정책  $\pi$  하에서 상태 s의 가치 함수(value function),  $r$ 은 보상(reward),  $\gamma \in [0, 1]$ 은 할인율(discount factor)이다.

최적화 목표는 다음과 같다:

$$\max_{\pi} V\pi(s).$$

## E. Optimization for Machine Learning

전반적으로 머신러닝의 주요 단계는 (1) 모델 가설(model hypothesis)을 수립하고, (2) 목적함수(objective function)를 정의하며, (3) 목적함수의 최댓값 또는 최솟값을 구해 모델의 파라미터를 결정하는 것이다. 이 세

단계 중 앞의 두 단계는 머신러닝의 모델링 문제에 해당하며, 세 번째 단계는 최적화 방법을 통해 원하는 모델을 푸는 과정이다.

## 내용

- 머신러닝 알고리즘 = 목적함수의 극값을 찾는 최적화 문제
  - 즉, 머신러닝의 첫 단계는 목적함수를 구축하고 합리적으로 정의하는 것
  - 수치적(numerical) 또는 해석적(analytical)으로 문제를 해결

### A. Optimization Problems in Supervised Learning

- 목표 : 훈련 샘플의 손실함수를 최소화하는 최적의 함수를 찾는 것

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N L(y^i, f(x^i, \theta)),$$

$f$  : 매핑 함수 ( $x, y$ ),  $L$  : 손실함수

- e.g. : 유클리드 거리 제곱(square of Euclidean distance), 교차 엔트로피(crossentropy), 대조 손실(contrast loss), 힌지 손실(hinge loss), 정보 이득(information gain) 등
- 회귀 : 경험적 손실 - 일반화 성능이 좋은 것은 아니다.
- 구조적 위험 최소화 : 정규화항을 추가하여 일반화 성능을 높인다.(예시는 L2norm, 람다는 cross validation으로 조정)

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N L(y^i, f(x^i, \theta)) + \lambda \|\theta\|_2^2,$$

*L2 norm*  
*Compromise parameter.*

### B. Optimization Problems in Semi-supervised Learning

- 목표 : 레이블이 있는 데이터와, 없는 데이터를 모두 활용하여 최적의 함수를 찾는 것(일반화 성능 상승)
- e.g. 자기학습(self-training), 생성 모델(generative models), 반지도 서포트 벡터 머신(S3VM), 그래프 기반 방법(graph-based methods), 다중 학습(multi-learning) 방식 등
  - S3VM : 이진 분류문제

### Note

#### S3VM

• labeled  $D^l := \{(x^l, y^l) | (x^l, y^l), \dots, (x^l, y^l)\}$

• unlabeled  $D^u := \{x^{l+1}, x^{l+2}, \dots, x^N\}, N = l + u$

$$\min_{w, b} \|w\| + C \left[ \sum_{i=1}^l \zeta_i + \sum_{j=l+1}^N \min(\epsilon_j, z_j) \right]$$

↑  
penalty      ↓  
slack

S3VM :  $\begin{cases} \text{정답 레이블이 } 1 \text{ 인지 } -1 \text{ 인지 모름} \\ +1 = \epsilon_j (\text{양수}), -1 = z_j (\text{음수}) \end{cases} \therefore \text{언제 레이블 부여시 오류를 별개로 } \downarrow \text{ 계산}$

모든 허용, 그에 대한 별개부과 (SVM)

subject to

$$\begin{aligned} & y^i(\mathbf{w} \cdot \mathbf{x}^i + b) + \zeta^i \geq 1, \zeta^i \geq 0, i = 1, \dots, l, \\ & \mathbf{w} \cdot \mathbf{x}^j + b + \epsilon^j \geq 1, \epsilon^j \geq 0, j = l + 1, \dots, N, \\ & -(\mathbf{w} \cdot \mathbf{x}^j + b) + z^j \geq 1, z^j \geq 0, \end{aligned}$$

SVM의 준지도학습버전으로, 가짜라벨을 붙여보면서 결정경계를 가장 잘 잡을 수 있도록 유도한다. 기본적으로 데이터는 "클러스터 가정"에 뿌리를 두어 결정경계는 밀도가 낮은부분을 통과한다는것을 가정한다. 이때 unlabeled 데이터가 이 밀도에 대한 정보를 제공하여, 일반화된 결정경계를 잡을 수 있도록 돋는다.

하지만 **non-convex**문제이기 때문에 근사 알고리즘을 사용한다.

## C. Optimization Problems in Unsupervised Learning

- 클러스터링 목표 : 같은 군집간 차이는 최소화, 다른 군집간 차이는 최대화
  - e.g. k-means : 모든 클러스터의 분산합 최소화

**# cluster**

$$\min \sum_{k=1}^K \sum_{x \in S_k} \|x - \mu_k\|_2^2$$

***k*<sup>th</sup> cluster's center**

- 차원축소 목표 : 저차원으로 사영하더라도 정보를 최대한 보존
  - e.g. PCA : 재구성 오차(reconstruction error)를 최소화

$$\min \sum_{i=1}^N \|\bar{x}^i - x^i\|_2^2 \quad \text{where} \quad \bar{x}^i = \sum_{j=1}^{D'} z_j^i e_j. \quad D \gg D'$$

↓ 사영  
 ↓ 고차원 저차원  
**stand. 직교 basis**

- 확률모델 목표 : 훈련 샘플의 로그 우도 함수(log-likelihood function, MLE)를 최대화하는 확률 밀도 함수  $p(x)$ 를 찾는 것 -> 베이지안 접근에서는 사전분포를 추가해 과적합을 방지

## D. Optimization Problems in Reinforcement Learning

- 강화학습 목표 : 환경에 따라 출력이 달라지는 최적의 전략 함수(strategy function)를 찾는 것으로서 정책 함수에 의해 결정된 일련의 행동을 실행한 뒤 얻는 누적 보상(cumulative return)을 최대화하는 것이다.
  - 순차적 의사결정의 최적화
  - 결정적(deterministic) 전략에서는 상태  $s$ 로부터 행동  $a$ 로의 매팅 함수가 학습 대상
  - 확률적(stochastic) 전략에서는 각 행동이 실행될 확률이 학습 대상

**value fn.**                      **reward**  
 $\max_{\pi} V_{\pi}(s) \quad \text{where} \quad \tilde{V}_{\pi}(s) = E \left[ \sum_{k=0}^{\infty} r^k r_{t+k} \mid s_t = s \right]$

**policy  $\pi$**                       **Discount State**

- 할인율 : 미래 보상의 가치를 조절

## E. Optimization for Machine Learning

- ML은 세 단계로 요약됨: 모델 가설  $\rightarrow$  목적함수 정의  $\rightarrow$  최적화
    - 앞 두단계 : 모델링
    - 마지막 단계 : 최적화
- 

## 포인트

### A. Optimization Problems in Supervised Learning

항목	내용
데이터셋	$(x_i, y_i), i=1, \dots, N$
모델 구조	$f(x; \theta)$
학습 방법	손실함수 최소화 + 정규화 항
손실 함수	Euclidean distance <sup>2</sup> , Cross Entropy, Contrast, Hinge, Information Gain 등
정규화	L2 norm, $\lambda$ 파라미터 (cross-validation로 결정)
대표 방법	Support Vector Machine (SRM 개념 기반)
결과 해석	경험적 손실은 일반화 보장 X $\rightarrow$ SRM과 정규화로 개선

### B. Optimization Problems in Semi-supervised Learning

항목	내용
데이터셋	DI (레이블 있음), Du (레이블 없음), $N = l + u$
모델 구조	SVM 확장 (S3VM)
학습 방법	손실 최소화 + unlabeled 데이터 제약
제약 조건	$\sum \min(q_j, z_j)$ 를 최소화 (오분류 최소화)
난이도	혼합정수 문제 (NP-hard 성격)
해결법	Branch & Bound, Convex Relaxation

### C. Optimization Problems in Unsupervised Learning

항목	내용
데이터셋	$N$ 개의 샘플 ( $x_i \in \mathbb{R}^D$ )
모델 구조	k-means ( $\mu_k$ 중심), PCA (투영 $z$ , 기저 $e_j$ ), 확률모델 $p(x; \theta)$

항목	내용
학습 방법	손실 최소화 (k-means, PCA) 또는 로그우도 최대화 (MLE)
평가 지표	클러스터링: 클러스터 내 분산, 차원축소: 재구성 오차, 확률모델: 로그우도
결과 해석	데이터 구조 발견, 저차원 표현 획득, 확률 분포 추정

## D. Optimization Problems in Reinforcement Learning & E. Optimization for Machine Learning

구분	핵심 요소	최적화 목표	수식/조건
강화학습	정책 함수 $\pi(s)$	누적 보상 최대화	$V\pi(s) = E[\sum \gamma^k r_{t+k}]$
머신러닝 일반	모델 가설, 목적함수	목적함수 극값 도출	$\theta^* = \text{argmin/max } f(\theta)$

### 🔬 실험과정

## 📚 3. FUNDAMENTAL OPTIMIZATION METHODS AND PROGRESSES

### 번역

경사 정보(gradient information)의 관점에서, 기초 최적화 방법은 1차 최적화 방법(first-order optimization methods), 고차 최적화 방법(high-order optimization methods), 그리고 미분 불필요 최적화 방법(derivative-free optimization methods)으로 나눌 수 있다. 이러한 방법들은 긴 역사를 가지고 있으며 끊임없이 발전해 왔다. 실제 응용에서도 진보를 거듭하며 좋은 성능을 달성하고 있다.

이러한 기초적인 방법들 외에도, 사전조건화(preconditioning)는 유용한 최적화 기법이다. 적절한 사전조건화를 적용하면 반복 횟수를 줄이고 스펙트럼 특성(spectral characteristics)을 개선할 수 있다. 이 기술들은 실제에서 널리 사용되고 있다. 연구자들의 편의를 위해, 본 절의 마지막 부분에서는 현재 존재하는 공통 최적화 툴킷들을 표로 요약한다.

## A. First-Order Methods

머신러닝 분야에서 가장 널리 사용되는 1차 최적화 방법은 주로 경사하강법(Gradient Descent)에 기반한다. 본 절에서는 경사하강법의 발전과 함께 대표적인 알고리즘들을 소개하며, 수치최적화에서의 고전적 방법인 승수의 교대방향법(Alternating Direction Method of Multipliers, ADMM)과 Frank-Wolfe 방법도 함께 다룬다.

1. 경사하강법(Gradient Descent): 경사하강법은 가장 오래되고 일반적인 최적화 방법이다. 아이디어는 목적함수의 기울기(gradient)의 반대 방향으로 변수를 반복적으로 업데이트하여 점진적으로 최적값에 수렴하게 하는 것이다. 학습률  $\eta$ 는 각 반복의 보폭(step size)을 결정하여 최적값에 도달하는 반복 횟수에 영향을 준다 [90]. 최급강하법(Steepest Descent Algorithm)은 잘 알려진 알고리즘이다. 각 반복에서 목적함수 값을 가장 빠르게 줄이는 탐색 방향을 선택하는 것이다. 경사하강법과 최급강하법은 동일하지 않으며, 음의 기울기 방향이 항상 가장 빠른 하강을 의미하지는 않는다. 경사하강법은 최급강하법에서 유클리드 노름(Euclidean norm)을 사용하는 특별한 경우다 [91]. 선형 회귀(linear regression) 모델에서, 학습 대상 함수  $f(\theta)$ , 손실함수  $L(\theta)$ , 최적화 대상 파라미터  $\theta$ 를 정의하면 목적은 다음을 최소화하는 것이다:

$$L(\theta) = (1/2N) \sum_{i=1}^N (y_i - f(\theta)) ^ 2$$

$$f(\theta) = \sum_{j=1}^D \theta_j x_j$$

여기서  $N$ 은 훈련 샘플 수,  $D$ 는 입력 특징 차원,  $x_i$ 는 독립변수,  $y_i$ 는 목표 출력이다. 경사하강법은 다음 두 단계를 반복한다:

$$\frac{\partial L(\theta)}{\partial \theta_j} = - (1/N) \sum (y_i - f(\theta)) x_{ji}$$

$$\theta'_j = \theta_j + \eta (1/N) \sum (y_i - f(\theta)) x_{ji}$$

경사하강법은 구현이 단순하다. 목적함수가 볼록(convex)일 경우 전역 최적해에 도달한다. 하지만 해에 가까워질수록 수렴 속도가 느려져 세심한 반복이 필요하다. 배치 경사하강법(Batch GD)은 모든 데이터를 매 반복마다 사용하므로 복잡도는  $O(ND)$ 이다. 병렬화 연구도 있었지만 [92], [93], 대규모 데이터에서는 여전히 계산량이 부담되어 확률적 경사하강법(SGD)이 등장했다.

2. 확률적 경사하강법(Stochastic Gradient Descent, SGD): 배치 GD의 높은 계산복잡도를 줄이기 위해 제안된 방법 [1]. SGD는 매 반복마다 무작위 샘플 하나를 이용해 업데이트하며, 이는 진짜 기울기의 불편 추정(unbiased estimate)이다. 반복당 계산복잡도는  $O(D)$ 이다. 수렴 속도는 부분선형(sublinear) [37] 이지만, 대규모 데이터와 온라인 학습에 적합하다. 손실함수는 다음과 같이 표현된다:

$$L(\theta) = (1/N) \sum \text{cost}(\theta, (x_i, y_i))$$

$$\text{단일 샘플 } i \text{의 경우: } L^*(\theta) = (1/2)(y_i - f(\theta)) ^ 2$$

$$\text{SGD의 업데이트 규칙: } \theta' = \theta + \eta (y_i - f(\theta)) x_i$$

SGD는 빠르지만 기울기 방향의 분산이 크고, 탐색 경로가 불안정하다. 이를 완화하기 위해 미니배치 SGD(MSGD)가 제안되었으며, 일반적으로  $b=50\sim256$  [90] 샘플을 사용한다. 본 논문 이후 부분에서는 MSGD를 간단히 SGD라 부른다.

SGD는 잡음으로 인해 지역 최소해(local minimum)에서 벗어나 전역 최적해(global optimum)를 찾을 가능성이 크다. 하지만 이러한 진동(fluctuation)은 수렴을 늦추기도 한다.

학습률(learning rate)의 선택은 매우 중요하다. 너무 작으면 수렴이 느리고, 너무 크면 발산하여 손실 함수가 최소값 근처에서 요동친다. 이를 해결하기 위해 사전에 정의한 학습률 스케줄이나 동적 조정법 [97], [98]이 사용된다. 희소 데이터(sparse data)에서는 특성별 등장 빈도가 달라 같은 학습률을 적용하는 것이 부적절하며, 드물게 등장하는 특성에는 더 큰 학습률이 필요하다 [30], [33].

또 다른 도전 과제는 안장점(saddle point)이다 [99]. 이는 어떤 방향으로는 양의 기울기, 다른 방향으로는 음의 기울기를 가지며 모든 방향에서 기울기가 0인 지점이다. SGD가 이 지점에서 빠져나오는 것이 중요한 문제이며, 이를 다룬 연구가 존재한다 [100], [101].

3. 네스테로프 가속 경사하강법(Nesterov Accelerated Gradient Descent, NAG): SGD는 널리 쓰이지만 학습이 오래 걸릴 때가 있다. 학습률을 어떻게 조절할지, 수렴을 어떻게 가속할지, 탐색 중 지역 최소값 (local minimum)에 갇히는 것을 어떻게 막을지가 중요한 연구 주제다.

SGD를 개선하기 위한 연구가 많다. 예를 들어 모멘텀(momentum) 개념이 SGD에 도입되었다 [102]. 모멘텀의 개념은 물리학의 역학에서 유래하여 물체의 관성(inertia)을 모사한다. SGD에 모멘텀을 적용하는 아이디어는 이전 갱신 방향의 영향을 어느 정도 다음 반복에 보존하는 것이다. 모멘텀 방법은 곡률이 큰 경우, 작지만 일관된 기울기, 또는 노이즈가 있는 기울기를 다룰 때 수렴을 가속할 수 있다 [103]. 모멘텀 알고리즘은 속도(speed) 변수  $v$ 를 도입하는데, 이는 파라미터 공간에서 파라미터 이동의 방향과 속도를 나타낸다. 속도는 음의 기울기의 지수가중 평균(exponential decay)으로 설정된다.

경사하강법에서 속도 갱신은 매번  $v = \eta \cdot (-\partial L(\theta)/\partial(\theta))$  이다. 모멘텀 알고리즘을 사용할 때 갱신량  $v$ 는  $\eta \cdot (-\partial L(\theta)/\partial(\theta))$ 로 계산되는 경사하강량만이 아니라,  $[0, 1]$  범위의 모멘텀 계수로 가중된 이전 갱신  $v_{old}$ (마찰 항)도 함께 고려한다. 일반적으로 질량은 1로 둔다. 식은 다음과 같이 표현된다.

$$v = \eta \cdot (-\partial L(\theta)/\partial(\theta)) \cdot mtm + v_{old} \quad (15)$$

여기서  $mtm$ 은 모멘텀 계수다. 현재 기울기가 이전 속도  $v_{old}$ 와 평행이면, 이전 속도가 탐색을 가속한다. 적절한 모멘텀은 학습률이 작을 때 수렴을 가속하는 역할을 한다. 도함수가 0으로 감쇠하면,  $v$ 는 평형에 도달할 때까지 계속 갱신되며 마찰로 감쇠된다. 이는 학습 중 지역 최소값에서 벗어나 더 빠르게 수렴하도록 돋는다 [102], [104]. 현재 기울기가 이전 갱신  $v_{old}$ 와 반대이면,  $v_{old}$ 는 감속 효과를 낸다.

적절한 모멘텀 계수를 쓰면 학습률이 클 때 수렴의 진동을 줄이는 데도 긍정적이다. 다만 계수 선택이 문제다. 너무 작으면 수렴 가속 효과가 약하고, 너무 크면 최적점 부근을 뛰어넘을 수 있다. 많은 실험에서 경험적으로 모멘텀 계수 0.9가 적절하다고 보고되었다 [90].

네스테로프 가속 경사하강법(NAG)은 전통적 모멘텀법보다 한 걸음 더 나아간다 [104], [105]. 네스테로프 모멘텀에서는  $v_{old} \cdot mtm$ 을  $\theta$ 에 대해 얻은  $\sim\theta$ 에서의 기울기를 사용해 갱신한다. 파라미터  $\theta$ 의 상세 갱신식은 다음과 같다:

$$\begin{aligned} \sim\theta &= \theta + v_{old} \cdot mtm, \\ v &= v_{old} \cdot mtm + \eta \cdot (-\partial L(\sim\theta)/\partial(\theta)), \\ \theta' &= \theta + v. \end{aligned} \quad (16)$$

네스테로프 모멘텀의 개선점은 현재 위치가 아니라 “미래 위치”的 기울기를 사용한다는 데 있으며, 전통적 모멘텀법 대비 더 많은 기울기 정보를 포함한다. 네스테로프 모멘텀은 (비확률 최적화일 때) 수렴율을  $O(1/k)$ 에서  $O(1/k^2)$ 로 개선한다 [105].

또 다른 고려 사항은 학습률의 크기를 어떻게 정하느냐이다. 최적점에 가까울수록 진동이 발생하기 쉬우므로 학습률을 조정해야 한다. SGD의 모멘텀 방법에서는 학습률 감쇠 계수  $d$ 를 흔히 사용하여, 반복에 따라 학습

률을 감소시킨다 [106]. 학습률 감쇠식은 다음과 같다:

$$\eta_t = \eta_0 / (1 + d \cdot t) \quad (17)$$

여기서  $\eta_t$ 는  $t$ 번째 반복의 학습률,  $\eta_0$ 는 초기 학습률,  $d$ 는  $[0, 1]$ 의 소수다.  $d$ 가 작을수록 감쇠가 느리고,  $d=0$ 이면 불변,  $d=1$ 이면 가장 빠르게 감소한다.

5. 분산 감소 기법(Variance Reduction Methods): 훈련 샘플에 중복 정보가 많기 때문에, SGD 계열은 제안 이후 널리 사용되어 왔다. 그러나 확률적 경사 기법은 부분선형(sublinear) 수렴만 보장되며, 기울기의 분산이 종종 매우 크다. 분산을 줄여 SGD를 선형 수렴으로 개선하는 것은 줄곧 중요한 문제였다.

Stochastic Average Gradient 확률적 평균 경사(SAG) 방법 [36]은 수렴 속도를 개선하기 위해 제안된 분산 감소 방법이다. SAG 알고리즘은 메모리에 최근  $N$ 개의 기울기  $\{g_i\}$ 의 합을 기록하는 파라미터  $d$ 를 유지하며, 여기서  $g_i$ 는 단일 샘플  $i$  ( $i \in \{1, \dots, N\}$ )로 계산된다. 구체 구현은 매 반복  $t$ 에서 무작위로 샘플  $i_t$ 를 선택해  $d$ 를 갱신하고, 그  $d$ 를 사용해 파라미터  $\theta$ 를 갱신하는 것이다:

□ □ □

$$d = d - g_i^t + g_{i_t}^t(\theta^{t-1}),$$

□ □ □

$$g_i^t = g_{i_t}^t(\theta^{t-1}),$$

(23)

$$\theta^t = \theta^{t-1} - \alpha d,$$

여기서  $d$ 의 갱신 항은, 반복  $t$ 에서  $d$  안의 오래된 기울기  $g_i^t$ 를 새로운 기울기  $g_{i_t}^t(\theta^{t-1})$ 로 치환하여 계산한다.  $\alpha$ 는 학습률을 나타내는 상수다. 그러므로 각 업데이트는 모든 샘플의 기울기를 계산할 필요가 없고, 샘플 하나의 기울기만 계산하면 된다. 계산량은 SGD와 다르지 않지만, 메모리 비용은 훨씬 크다. 이는 시간을 절약하기 위해 공간을 사용하는 전형적 방법이다. SAG는 선형 수렴(linear convergence) 알고리즘이 보였으며 [36], SGD보다 훨씬 빠르고, 다른 확률적 경사 알고리즘 대비 큰 이점을 가진다.

다만, SAG는 손실함수가 매끄럽고(smooth) 목적함수가 볼록(convex)인 경우(예: 볼록 선형 예측 문제)에만 적용 가능하다 [36], [108]. 이 경우, SAG는 SGD보다 더 빠른 수렴 속도를 보인다. 더 나아가 특정 문제들에서는 표준 배치 경사하강법보다도 더 나은 수렴을 보일 수 있다.

6. 승수의 교대 방향법(Alternating Direction Method of Multipliers, ADMM): 증가형 라그랑주 승수법 (Augmented Lagrangian multiplier method)은 선형 제약을 가진 최적화 문제를 푸는 일반적인 방법이다. 순수 라그랑주 승수법에 비해 목적함수에 패널티 항을 추가하여 문제를 더 쉽게 풀 수 있게 한다. 다음 예를 보자.

$$\min \{ \theta_1(x) + \theta_2(y) \mid Ax + By = b, x \in X, y \in Y \}. \quad (27)$$

문제 (27)의 증가형 라그랑주 함수는 다음과 같다.

$$L_\beta(x, y, \lambda) = \theta_1(x) + \theta_2(y) - \lambda^T(Ax + By - b) + \beta/2 \cdot \|Ax + By - b\|_2^2. \quad (28)$$

증가형 라그랑주 승수법으로 풀 때,  $t$ 번째 단계는 주어진  $\lambda^t$ 에서 시작하며, 최적화는 다음과 같이 된다.

$$(x^{t+1}, y^{t+1}) = \arg \min \{ L_\beta(x, y, \lambda^t) \mid x \in X, y \in Y \},$$

$$\{ \lambda^{t+1} = \lambda^t - \beta (Ax^{t+1} + By^{t+1} - b).$$

(29)

(29)에서  $(x, y)$  부분문제를 분리(separating)하면, 증가형 라그랑주 승수법을 다음의 승수의 교대 방향법(ADMM) [112], [113]으로 완화할 수 있다.  $t$ 번째 단계는 주어진  $(y^t, \lambda^t)$ 에서 시작하며, 반복 최적화의 세부는 다음과 같다.

$$x^{t+1} = \arg \min_{x \in X} \{ \theta_1(x) - (\lambda^t)^T A x + \beta/2 \|Con_x\|_2^2 \},$$

$$y^{t+1} = \arg \min_{y \in Y} \{ \theta_2(y) - (\lambda^t)^T B y + \beta/2 \|Con_y\|_2^2 \},$$

$$\lambda^{t+1} = \lambda^t - \beta (A x^{t+1} + B y^{t+1} - b), \quad (30)$$

where  $Con_x = Ax + By^t - b$  and  $Con_y = Ax^{t+1} + By - b$ .

패널티 파라미터  $\beta$ 는 ADMM의 수렴 속도에 일정한 영향을 미친다.  $\beta$ 가 클수록 제약항에 대한 패널티가 커진다. 일반적으로 고정된  $\beta$  대신 단조 증가하는 수열  $\{\beta_t\}$ 를 채택할 수 있다 [114]. 구체적으로, 반복 도중  $\{x^t\}$ 의 현재 값에 기반해  $\{\beta_t\}$ 를 자동 조정하는 기준이 제안되어, 몇몇 볼록 최적화 문제를 푸는 데 적용되었다 [115], [116].

ADMM은 볼록 최적화 문제에서의 분리 가능 연산자(separable operators)를 이용하여 큰 문제를 분산 방식으로 풀 수 있는 여러 개의 작은 문제로 분할한다. 이론적으로, ADMM 프레임워크는 대부분의 대규모 최적화 문제를 풀 수 있다. 그러나 실무 적용에는 여전히 몇 가지 문제가 있다. 예를 들어, 수렴 여부를 판단하기 위해 정지 기준을 사용할 때, 원(primal) 잔차와 쌍대(dual) 잔차는 모두  $\beta$ 에 관련되어 있으며,  $\beta$ 가 큰 값이면 수렴 조건을 만족하기 어려워지는 문제가 발생한다 [117].

프랭크-울프 방법(Frank-Wolfe Method): 1956년 Frank와 Wolfe는 선형 제약 문제를 푸는 알고리즘을 제안했다 [118]. 기본 아이디어는 목적함수를 선형함수로 근사한 뒤, 선형계획(linear programming)을 풀어 실행 가능(feasible) 하강 방향을 찾고, 마지막으로 실행 가능 영역에서 그 방향을 따라 1차원 탐색을 수행하는 것이다. 이 방법을 근사 선형화 방법(approximate linearization method)이라고도 부른다.

여기서는 Frank-Wolfe 방법의 간단한 예를 제시한다. 다음 최적화 문제를 고려하자.

$$\min f(x), \text{s.t. } Ax = b, \quad x \geq 0,$$

$\geq$

0,

(31)

여기서  $A$ 는  $m \times n$ 의 만랭크(full row rank) 행렬이며, 실행 가능 영역은  $S = \{ x \mid Ax = b, x \geq 0 \}$ 이다.  $x_0$ 에서  $f(x)$ 를 1차로 전개하면  $f(x) \approx f(x_0) + \nabla f(x_0)^T (x - x_0)$ 이고, 이를 식 (31)에 대입하면

$$\min f(x_t) + \nabla f(x_t)^T (x - x_t), \{ \text{s.t. } x \in S,$$

(32)

이 된다. 이는 다음과 동치이다

$$\{ \text{s.t. } x \in S.$$

최적해  $y_t$ 가 존재한다고 하자. 그러면 반드시

$$\{ \nabla f(x_t) (y_t - x_t) < 0.$$

(34)

가 성립한다. 따라서  $y_t - x_t$ 는  $x_t$ 에서  $f(x)$ 의 감소 방향이다. 스텝 크기  $\lambda_t$ 를 취해 실행 가능 방향으로 탐색점을 갱신한다. 상세한 절차는 알고리즘 1에 보였다.

Algorithm 1 Frank-Wolfe Method [118], [119]

Input:  $x_0, \varepsilon \geq 0, t := 0$  Output:  $x^*$

```
 $yt \leftarrow \min \nabla f(x_t)^T x$  while  $|\nabla f(x_t)^T (y_t - x_t)| > \varepsilon$  do  $\lambda_t = \operatorname{argmin}\{0 \leq \lambda \leq 1\} f(x_t + \lambda(y_t - x_t))$ 
```

$x_{t+1} \approx x_t$   $t := t + 1$  end while  $x^* \approx x_t$

- $\lambda_t (y_t - x_t)$

$y_t \leftarrow \min \nabla f(x_t)^T x$

이 알고리즘은 다음 수렴 정리를 만족한다 [118].

(1)  $\nabla f(x_t)^T (y_t - x_t) = 0$ 일 때  $x_t$ 는 (31)의 Kuhn-Tucker 점이다.

(2)  $y_t$ 가 문제 (33)의 최적해이므로, 벡터  $d_t = y_t - x_t$ 는  $\nabla f(x_t)^T (y_t - x_t) \neq 0$ 일 때 점  $x_t$ 에서  $f$ 의 실행 가능 하강 방향이다.

Frank-Wolfe 알고리즘은 제약이 있는 볼록 최적화 문제를 푸는 1차 반복 방법이다. 실행 가능 하강 방향을 결정하고 탐색 스텝을 계산하는 것으로 구성된다. 초반 반복에서 빠르게 수렴하고 후반에는 느려지는 특성이 있다. 반복점이 최적해에 가까워지면 탐색 방향과 목적함수의 기울기 방향이 서로 직교에 가까워진다. 이러한 방향은 최적의 하강 방향이 아니므로, 하강 방향 선택 측면에서 Frank-Wolfe 알고리즘은 개선·확장될 수 있다 [120], [121], [122]. 9

## 내용

- Gradient

- 1차 최적화(first-order optimization) : 단순, 확장성 높음(대규모 문제에 강함)
- 고차 최적화(high-order optimization) : 수렴빠름, Hessian(계산량이 큼)
- 미분 불필요 최적화(derivative-free optimization) : 미분불가 대응(noisy 포함)
- 사전조건화(preconditioning) : 반복 횟수를 줄이고, 스펙트럼의 특성을 개선할 수 있다.

## A. First-Order Methods

- 주로 경사하강법에 기반

### 1. 경사하강법(Gradient Descent)

- 목적함수의 기울기의 반대방향으로 변수를 반복적으로 업데이트하여 점진적으로 최적값에 수렴하게 만드는 방법
  - 학습률(learning rate,  $\eta$ ) : 반복의 보폭을 결정
  - e.g. 선형회귀문제 목적

$$L(\theta) = \frac{1}{2N} \sum_{i=1}^N (y^i - f_\theta(x^i))^2,$$

$$f_\theta(x) = \sum_{j=1}^D \theta_j x_j,$$

- 경사하강법 적용

- 1) Derive  $L(\theta)$  for  $\theta_j$  to get the gradient corresponding to each  $\theta_j$ :

$$\frac{\partial L(\theta)}{\partial \theta_j} = -\frac{1}{N} \sum_{i=1}^N (y^i - f_\theta(x^i)) x_j^i. \quad (10)$$

- 2) Update each  $\theta_j$  in the negative gradient direction to minimize the risk function:

$$\theta_j' = \theta_j + \eta \cdot \frac{1}{N} \sum_{i=1}^N (y^i - f_\theta(x^i)) x_j^i. \quad (11)$$

- 1. 현 상태에서 가장 가파른 기울기 계산
  2. 경사의 반대방향으로 조금씩(학습률 만큼)이동
- 목적함수가 convex일 경우 최적해에 도달하지만 해에 도달할수록 수렴속도가 느려짐

- e.g. 배치 경사하강법은 매 반복마다 사용하여 복잡도가  $O(ND)$ 이며 병렬화 연구도 있었지만 효율적이지 않아 SGD가 개발되었다.

### Note

#### 최급하강법

각 반복에서 목적함수의 값을 가장 빠르게 줄이는 탐색방향을 선택

- 경사하강법과 최급하강법은 동일하지 않음
- 경사하강법은 최급하강법에서 유클리드 노름(Euclidean norm)을 사용하는 특별한 경우

## 2. 확률적 경사하강법(Stochastic Gradient Descent, SGD)

- 배치 경사하강법의 높은 복잡도를 개선하기 위해 개발되어짐
- 매 반복마다 무작위 샘플하나를 이용해 업데이트하며 이는 기울기의 불편추정량이다
- 복잡도 :  $O(D)$ , 수렴속도는 부분선형

$$L(\theta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2} (y^i - f_\theta(x^i))^2 = \frac{1}{N} \sum_{i=1}^N \text{cost}(\theta, (x^i, y^i)). \quad (12)$$

If a random sample  $i$  is selected in SGD, the loss function will be  $L^*(\theta)$ :

$$L^*(\theta) = \text{cost}(\theta, (x^i, y^i)) = \frac{1}{2} (y^i - f_\theta(x^i))^2. \quad (13)$$

The gradient update in SGD uses the random sample  $i$  rather than all samples in each iteration,

$$\theta' = \theta + \eta (y^i - f_\theta(x^i)) x^i. \quad (14)$$

- 빠르지만 기울기 방향의 분산이 크고, 탐색 경로가 불안정하다. 이를 해결하기 위해 미니배치가 개발되었고 **SGD**의 기본값처럼 되었다.
- **local minima**에서 **global minima**로 탈출할 가능성이 높은데, 이는 진동때문이다. 하지만 이러한 진동은 수렴속도를 늦추기도 한다. 따라서 학습률 조정이 중요하며 동적 스케줄링이 사용되기도 한다.
- 특히 sparse data에서는 드물게 등장하는 특성은 학습 기회가 귀하니, 한번 학습시킬 때 과감하게 업데이트 시킴(Adam, Adagrad 등)
- 기울기가 0인 지점인 안정점을 빠져나오는 것에 대한 문제도 있기에 이를 해결하는 것도 중요함.
  - 안정점탈출은 중요한 연구과제임

## Note

### SGD의 장점 - 볼록 vs 비볼록

노이즈덕분에 비볼록 문제에서 탈출이 용이

## 3. 네스테로프 가속 경사하강법(Nesterov Accelerated Gradient Descent, NAG)

- SGD가 많이 사용되지만, 학습이 오래걸릴때가 있음. 아래는 주요 논의사항이다.
  - 학습률 조절
  - 수렴속도를 가속
  - local minima 탈출
- 이를 해결하기 위하여 많은 방법들이 소개되었음.

### 3.1. 모멘텀(Momentum)

- Momentum : 갱신 방향의 영향을 어느 정도 다음 반복에 보존
  - 곡률이 큰 경우, 작지만 기울기가 일관된 경우, 노이즈가 있는 기울기를 다룰 때 수렴을 가속화시킬 수 있음
  - 속도변수  $v$ 를 도입하여 파라미터의 속도와 방향을 나타냄(속도는 음의 기울기로 지수가중평균 적용)

$$v = \eta \cdot \left( -\frac{\partial L(\theta)}{\partial(\theta)} \right) + v^{old} \cdot mtm,$$

- $v$ 를 갱신하기위해서, 학습률뿐만아니라 마찰항( $v^{old}$ ), 질량=1로 적용, mtm은 모멘텀 계수

$$\eta_t = \frac{\eta_0}{1 + d \cdot t},$$

- 적절한 모멘텀계수는 학습률이 작을 때 빠르게 수렴하도록 도와줌. 도함수가 0으로 감쇠하면,  $v$ 는 평형에 도달할 때까지 계속 갱신되며 마찰로 감쇠된다. 이는 학습 중 지역 최소값에서 벗어나 더 빠르게 수렴
- 기울기와 마찰항이 반대면, 감속효과를 낸다

$$\begin{cases} \tilde{\theta} = \theta + v^{old} \cdot mtm, \\ v = v^{old} \cdot mtm + \eta \cdot \left( -\frac{\partial L(\tilde{\theta})}{\partial (\theta)} \right), \\ \theta' = \theta + v. \end{cases}$$

- 현재 위치의 기울기가 아니라 "미래위치"의 기울기를 활용 : 모멘텀방법의 기울기정보보다 더 많은 정보를 사용한다

## 4. 적응형 학습률 방법(Adaptive Learning Rate Method)

- SGD와 비교하여 자동으로 학습률을 조정하는 방법들임

### 4.1. AdaGrad

$$\begin{cases} g_t = \frac{\partial L(\theta_t)}{\partial \theta}, \\ V_t = \sqrt{\sum_{i=1}^t (g_i)^2 + \epsilon}, \\ \theta_{t+1} = \theta_t - \eta \frac{g_t}{V_t}, \end{cases} \quad (18)$$

- 이전 반복의 기울기를 바탕으로 동적으로 조절 이때,  $V_t$ 는 누적 기울기이다.
- GD와 가장 큰 차이점은, 현재까지 사용되었던 기울기들을 이용해 학습률을 조정하는것
- 단점
  - 광역 학습률  $\eta$ 는 여전히 수동으로 설정
  - 학습이 진행될수록 누적 기울기가 커져 학습률이 0에 수렴해 갱신이 비효율적

### 4.2. RMSProp, AdaDelta

$$V_t = \sqrt{\beta V_{t-1} + (1 - \beta)(g_t)^2},$$

- 모든 역사적 기울기를 누적하지 않고, 일정 기간의 "창(window)"에 있는 기울기만 고려하며, 지수가중 이동평균(exponential moving average, ema)으로 제곱 기울기의 누적 모멘텀을 계산
- 이때 beta는 감쇠 계수임

### 4.3. Adam

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)g_t, \quad (20)$$

$$V_t = \sqrt{\beta_2 V_{t-1} + (1 - \beta_2)(g_t)^2}, \quad (21)$$

where  $\beta_1$  and  $\beta_2$  are exponential decay rates. The final update formula for the parameter  $\theta$  is

$$\theta_{t+1} = m_t - \eta \frac{\sqrt{1 - \beta_2}}{1 - \beta_1} \frac{m_t}{V_t + \epsilon}. \quad (22)$$

- AdaDelta/RMSProp처럼 과거 제곱 기울기의 지수가중 평균 저장할 뿐 아니라, 모멘텀 방식과 유사하게 과거 기울기의 지수가중 평균도 함께 유지

## 5. 분산 감소 기법(Variance Reduction Methods)

- SGD계열은 부분선형의 수렴만 보장되며, 기울기의 분산이 종종 매우큰 단점이 있다. 그렇기 때문에 선형 수렴을 가지게 하는것은 매우중요하다.

### 5.1. 확률적 평균 경사(Stochastic Average Gradient , SAG)

$$\begin{cases} d = d - \hat{g}_{i_t} + g_{i_t}(\theta_{t-1}), \\ \hat{g}_{i_t} = g_{i_t}(\theta_{t-1}), \\ \theta_t = \theta_{t-1} - \frac{\alpha}{N}d, \end{cases}$$

- 메모리에 최근 N개의 기울기의 합( $g_i$ )를 기록하는  $d$ 를 사용한다. -> "기억력"
- 매 반복에서 무작위로 샘플  $i_t$ 를 사용하여  $d$ 를 갱신하고,  $d$ 를 이용해  $\theta$ 를갱신
- 새로운 데이터가 들어오면 과거의 기울기를 새로운 기울기로 교체하고 전체기울기의 합에서 과거의 기울기를 빼고 새로운 기울기를 더하여 빠르게 평균 기울기를 계산
- 단점 : 시간복잡도 계선을위해 공간복잡도를 사용, Convex인경우 효과적

### 5.2. 확률적 분산 감소 경사 (Stochastic Variance Reduction Gradient, SVRG)

## SAG의 기억력 컨셉은 유지하되, 메모리 문제 및 Non-Convex를 해결

- 체크포인트 : 일정 반복마다 전체 샘플의 기울기를 계산하여 저장

$$\tilde{\mu} = \frac{1}{N} \sum_{i=1}^N g_i(\tilde{\theta}),$$

$$\theta_t = \theta_{t-1} - \eta \cdot (g_{i_t}(\theta_{t-1}) - g_{i_t}(\tilde{\theta}) + \tilde{\mu}).$$

$$g_{i_t}(\theta_{t-1}) - g_{i_t}(\tilde{\theta}) + \tilde{\mu} \rightarrow g_{i_t}(\theta_{t-1}) - g_{i_t}(\theta^*) \rightarrow 0.$$

- 이후 업데이트는, 체크포인트에 저장된 전체 기울기를 기준점으로 삼고, 현재 데이터의 기울기와 체크포인트 시점의 기울기 간의 차이를 보정해주는 방식으로 진행한다.
- 현재 스텝의 기울기'에서 '체크포인트 스텝의 (동일 샘플) 기울기'를 뺀 값을 '기울기 변화량'으로 계산한다. 이 '기울기 변화량'을 '체크포인트의 전체 기울기'에 더하여 최종 업데이트 방향을 결정한다.
- 특징
  - 하나의 업데이트에서 기울기는 많아야 두번 계산됨 : ('현재 기울기' - '체크포인트 시점 기울기')
  - 분산 감소(**variance reduction**) : SGD는 그래디언트 추정량의 분산(variance)이 0으로 수렴하지 않아, 일정한 학습률(learning rate) 사용 시 선형 수렴을 달성하기 어렵다. 하지만 SVRG는 수렴 과정에서 분산의 상한을 지속적으로 줄여나가기 때문에 선형 수렴이 가능하다.
- 변형 : SAGA

## 6. 승수의 교대 방향법(Alternating Direction Method of Multipliers, ADMM)

### Divide and Conquer

- $f(x), g(y)$ 를 최소화 하고 싶을때, 단,  $x$ 와  $y$ 가  $Ax + By = c$ 와 같은 제약 조건으로 연결되어 있을 때 각각 교대로 풀어 해결한다. 왜냐하면 각각의 함수를 푸는것은 쉽기 때문이다
- e.g. 증가형 라그랑주 승수법

$$\min \{ \theta_1(x) + \theta_2(y) | Ax + By = b, x \in \mathcal{X}, y \in \mathcal{Y} \}. \quad (27)$$

The augmented Lagrange function for problem (27) is

$$\begin{aligned} \mathcal{L}_\beta(x, y, \lambda) = & \theta_1(x) + \theta_2(y) - \lambda^\top(Ax + By - b) \\ & + \frac{\beta}{2} \|Ax + By - b\|^2. \end{aligned} \quad (28)$$

$$\begin{cases} x_{t+1} = \arg \min \left\{ \theta_1(x) - (\lambda_t)^\top Ax + \frac{\beta}{2} \|\text{Con}_x\|^2 | x \in \mathcal{X} \right\}, \\ y_{t+1} = \arg \min \left\{ \theta_2(y) - (\lambda_t)^\top By + \frac{\beta}{2} \|\text{Con}_y\|^2 | y \in \mathcal{Y} \right\}, \\ \lambda_{t+1} = \lambda_t - \beta(Ax_{t+1} + By_{t+1} - b), \end{cases}$$

- 증가형 라그랑주 승수법을 2개의 식으로 나눠 업데이트 하는것과 같다.

- 장점
  - 규모 문제를 여러 개의 작은 부분 문제로 분해하여 병렬 처리를 용이하게 함
- 단점
  - 페널티의 강도를 조절하는  $\beta$  값을 튜닝하기가 까다로워 수렴이 힘들수도 있다.(원시-쌍대(primal-dual) 잔차 규칙 어려워짐) - 진동 혹은 발산 가능

## 7. 프랭크-울프 방법(Frank-Wolfe Method)

Frank-Wolfe 방법은 복잡한 제약 조건이 있는 언덕에서 가장 낮은 지점을 찾는, 영리하지만 단순한 "나침반과 직선 코스" 전략

- 선형근사와 직선탐색
  - 가장 좋아 보이는 방향 찾기, 현재시점에서 가장 가파른 기울기를 찾고 그 기울기 방향으로 가장 멀리 있는 지점을 찾음(제약조건 성립하는 목표 지점)
  - 목표 지점에 돌진하는게 아니라, 현재시점과 목표지점까지의 직선에서 가장 낮은 점을 찾아 이동

## 요약

### 1. 배치 경사하강법(Batch Gradient Descent, GD)

$$\theta_{t+1} = \theta_t - \eta \nabla f(\theta_t) = \theta_t - \eta g_t$$

- $\theta_0$ : 초기값     $g_t = \nabla f(\theta_t)$

- 아이디어 : 전체 데이터를 활용해 가장 가파른 기울기방향으로 이동

**360도를 돌아보고, 완벽한 기울기(가장 가파른 기울기)로 내려감**

- 알고리즘
  1. 초기값  $\theta_0$  설정
  2. 반복 : 전체 데이터로  $g_t = \nabla f(\theta_t)$ (이후 기울기는  $g_t$ 로 통일) 계산
  3.  $\theta_{t+1} = \theta_t - \eta g_t$ 로 갱신
  4. 수렴(기울기 작아지면/변화율 작으면)하면 종료
- 장점
  - 볼록(convex)이면 전역최적점으로 수렴
  - 잡음이 없어 스텝이 안정적
- 단점
  - 매 스텝마다 전체 데이터 기울기  $\rightarrow$  계산 비용 큼
  - 비현실적

## 2. 확률적 경사하강법(Stochastic Gradient Descent/Mini-batch Gradient Descent, SGD)

$$\theta_{t+1} = \theta_t - \eta g_t$$

$$g_t = \frac{1}{|B|} \sum_{i \in B} \nabla f_i(\theta_t)$$

- 아이디어 : 데이터의 일부를 활용하여(샘플 하나 또는 미니배치)로 근사 기울기를 써서 자주, 싸게 업데이트

아무곳이나 일단 내려감

- 알고리즘(mini batch)
  - 초기값  $\theta_0$  설정 & 배치 크기 설정  $B$
  - 반복(epoch마다 데이터를 섞고, 미니배치로 순회) :  $g_t$  계산
  - $\theta$  업데이트
  - 학습률  $\eta$  : 스케줄링 / early stopping으로 관리
- 장점
  - 업데이트가 빠름, 대용량 데이터에 유리
  - 자주 갱신에 초반에 목표로 빠르게 접근
- 단점
  - 학습률 선택이 어려움, 고정 학습률을 비효율적
  - 지그재그(진동)으로 인해 최적의 해 근처에서 헤맬 수 있음

### 3. 네스테로프 가속 경사 (Nesterov Accelerated Gradient, NAG)

$$V_{t+1} = \text{mtm} \cdot V_t - \eta \frac{\nabla f(\theta_t + \text{mtm} \cdot V_t)}{\boxed{\nabla f(\theta_t)}}$$

↓ look ahead (예상위치)

$$\theta_{t+1} = \theta_t + V_{t+1}$$

$\text{mtm}$ : 모멘텀 계수 (practically use  $\tilde{0.9}$ ),  $V_t$ : 시점  $t$ 에서의 속도

- 아이디어 : 모멘텀을 사용하되, “미리 한 발 나간 위치”에서 기울기를 본 뒤 조정(진동 억제 및 가속)

### 자율주행기능을 탑재한 축구 공

모멘텀을 이용해서 경사를 내려가되, 살짝 이동 후 경사를 확인하여 그 방향으로 이동

- 알고리즘
  - $\theta_0, v_0 = 0, \eta, \mu$  설정
  - 반복 :
    - look ahead :  $\tilde{\theta} = \theta_t + \text{mtm} v_t$
    - $g_t = \Delta f(\tilde{\theta})$
    - $v_{t+1} = \text{mtm} v_t - \eta g_t$
    - $\theta_{t+1} = \theta_t + v_{t+1}$
  - 수렴시 종료
- 장점
  - 방향이 바뀔 때 속도를 줄여 진동이 감소, 같은 방향이면 가속  $\rightarrow$  단순 모멘텀보다 진동감소, 수렴속도 증가
  - local minima는 모멘텀을 통해 탈출 가능
- 단점
  - 학습률 선택이 어려움

## 4. AdaGrad (Adaptive Gradient)

## 기울기 제곱의 누적합(각 좌표별)

$$r_t = r_{t-1} + g_t \odot g_t, \quad g_t \odot g_t = \{g_{t,1}^2, g_{t,2}^2, \dots\}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{r_t} + \epsilon} \odot g_t$$

보정항 (e.g.  $10^{-8}$ )

element-wise product (원소별 곱)

- 아이디어 : 좌표별로 기울기 제곱을 누적해, 자주 큰 기울기가 나온 좌표는 스텝을 자동으로 작게

## 기억력좋은 스키선수

자주 다닌 길(가파른 방향)은 더 위험할 수 있으니 보폭을 줄임, 그와 반대로 처음가는 길(완만한 방향)은 위험하지 않을테니 보폭을 늘림

- 알고리즘
  - $\theta_0, r_0 = 0, \eta, \epsilon$  설정
  - 반복
    - $g_t$  계산
    - $r_t = r_{t-1} + g_t \odot g_t$
    - 좌표별로 업데이트
  - 종료조건까지 반복
- 장점
  - 희소/불균형 기울기에 강함(자주 갱신되는 좌표는 자동 감속)
  - 초기 학습이 빠름
  - 편리함
- 단점
  - $r_t$ 가 계속 커져  $\sqrt{r_t}$ 가 커지면 학습률이 0에 수렴  $\rightarrow$  후반 학습 정체.
  - 비볼록(non-convex) 문제엔 한계, 보정 필요.

## 5. RMSProp & AdaDelta

### 5.1. RMSProp

- 공통 아이디어 : AdaGrad의 무한 기울기 누적대신, 최근 기울기만 반영한 EMA를 적용

## 기억력이 적당한 스키선수

기억력이 적당하기 때문에, 모든 기울기를 기억하지 않는다.

$$r_t = \rho r_{t-1} + (1-\rho) g_t^2$$

↳ 가중치의 차수가 중평균 (EWMA) : 가중치 제곱의 이동평균

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{r_t} + \epsilon} \Theta g_t$$

$\rho$  : 감쇠율 (decay rate, usually use  $\tilde{0.9}$ )

- 알고리즘
  1.  $\theta_0, r_0 = 0, \rho, \eta$  설정
  2. 반복 :
    1.  $g_t$  계산
    2.  $r_t = \rho r_{t-1} + (1 - \rho) g_t^2$ 로 EWMA 업데이트
    3. 좌표별로 정규화 업데이트
  3. 종료 조건까지 반복
- 장점
  - AdaGrad의 학습률 소실 문제 완화
  - convex, non-convex 둘 다 안정적
- 단점
  - 학습 후반부 local minima에서 진동 가능성 존재

## 5.2. AdaDelta

기울기<sup>2</sup>의 cma (과거 기울기 추적)

$$\cdot \underline{E(g^2)_t} = \rho E[g^2]_{t-1} + (1-\rho)g_t^2 : \text{기울기 } g^2 \text{ 기대치}$$

Root mean square

$$\Delta\theta_t = - \frac{\overline{\text{RMS}[\Delta\theta]_{t-1}}}{\overline{\text{RMS}[g]_t}} g_t : \text{Step 크기 계산}$$

step<sup>2</sup> 기대치

$$\underline{E[\Delta\theta^2]_t} = \rho E[\Delta\theta^2]_{t-1} + (1-\rho)\Delta\theta_t^2$$

step<sup>2</sup>의 cma

$$\theta_{t+1} = \theta_t + \underline{\Delta\theta_t}$$

업데이트 step

- 알고리즘

1.  $E[g^2], E[\Delta\theta^2] = 0$  으로 시작

2. 반복

1.  $g_t$

2.  $E[g^2]_t$  갱신

3.  $\Delta\theta_t$  산출

4.  $E[\Delta\theta^2]_t$  갱신

5.  $\theta$  업데이트

3. 종료조건까지 반복

- 장점

- 좌표 스케일 자동화

- 명시적  $\eta$ 의존도↓.
- AdaGrad의 누적 문제 해소.
- non-convex 문제에서 좋음
- 단점
  - 최적 하이퍼(특히  $\rho, \epsilon$ ) 세팅 필요.
  - local minima 주변 진동 가능

## 6. ✖ Adam (Adaptive Moment Estimation)

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad \text{momentum}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad \text{RMSProp}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad \text{bias correction}$$

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

$\beta_1$ : momentum decay rate (usually 0.9),  $\beta_2$ : RMSProp decay rate (usually 0.999)

- 아이디어 : 모멘텀(1차 모멘트) + RMSProp(2차 모멘트) 를 결합하고 바이어스 보정

좋은것만 넣은 스키선수

모멘텀을 통해서 빠르게 나가면서, 지형에 따라 step길이가 자동 조절(RMSProp)

- 알고리즘
  1.  $\theta_0, m_0, v_0 = 0$  초기값 설정 및  $\beta_1, \beta_2, \eta$  설정
  2. 반복
    1.  $g_t$
    2.  $m_t, v_t$  를 EMA
    3.  $\hat{m}_t, \hat{v}_t$  계산
    4. 파라미터 업데이트
  3. 종료조건 종료
- 장점
  - 안정적

- 튜닝 쉬움, non-convex에 강함, 대규모 데이터에 강함
- 단점
  - 특정 문제에 대해서 수렴이슈 보고됨

## 7. SAG (Stochastic Average Gradient)

각 샘플  $i_t$ 에 대한 기울기  $\phi_i$  저장

wave

최근에 생산된 기울기 저장 vec.

$$\underline{g_{new}} = \nabla f_{i_t}(\theta_t)$$

새 기울기

$$\bar{\phi}_t = \bar{\phi}_{t-1} + \frac{1}{n}(g_{new} - \underline{\phi}_{it}^{\text{old}})$$

기울기 평균 계산

업데이트 직전의 저장값

$$\underline{\phi}_{it} \leftarrow g_{new}$$

저장치 갱신

$$\theta_{t+1} = \theta_t - \gamma \bar{\phi}_t$$

- 아이디어 : 각 샘플의 최신 기울기를 메모리에 저장, 그 평균으로 업데이터  $\rightarrow$  분산 감소, 선형 수렴가능성

모든 선수의 의견을 수렴하는 탐사대장(스키)

- 알고리즘
  1. 모든  $\phi_i$  를 0 또는 초기 기울기로 설정.
  2. 반복
    1. 샘플  $i_t$  선택,  $g_{new}$  계산
    2.  $\bar{\phi}$  업데이트
    3. 파라미터 업데이트
  3. 수렴 시 종료.

- 장점
  - 선형 수렴(매끈 · convex)로 SGD보다 빠름.
  - 분산이 작아 업데이트가 안정적.
- 단점
  - 각 샘플 기울기 저장 메모리 필요(공간복잡도 상승)
  - convex·매끈 가정에 의존, non-convex문제 발생

## 8. SVRG (Stochastic Variance Reduced Gradient)

$\tilde{\mu} \approx \nabla f(\tilde{\theta})$ ,  $\tilde{\theta}$  : checkpoint에서의 param.

$$\tilde{g} = \nabla f_i(\theta) - \nabla f_i(\tilde{\theta}) + \tilde{\mu}, \quad \theta \leftarrow \theta - \eta \tilde{g}$$

- 아이디어 : 주기적으로 기준점  $\tilde{\theta}$ 에서 전체 기울기  $\tilde{\mu}$ 를 계산해, 미니배치 기울기에서 분산 감소 보정. - 체크포인트 활용

### 주기적으로 의견을 확인하는 탐사대장(스키)

- 알고리즘
  1. 외부 루프에서 기준점  $\tilde{\theta} \leftarrow \theta$  고정,  $\tilde{\mu} = \nabla f(\tilde{\theta})$  계산(풀 배치 1회).
  2. 샘플  $i$ 를 뽑아  $\tilde{g}$  계산.
  3.  $\theta \leftarrow \theta - \eta \tilde{g}$
  4. 필요 시  $\tilde{\theta} \leftarrow \theta$ 로 갱신하고 반복.
- 장점
  - 모든 샘플 기울기 저장 불필요(SAG 대비 메모리 절약).
  - 선형 수렴(매끈·강볼록), 분산 크게 감소.
- 단점
  - 기준점 계산에 풀 배치 기울기 필요
  - 대용량 데이터 실용성 문제 발생
  -

## 9. ADMM (Alternating Direction Method of Multipliers)

- Standard

$$\min_{x,z} f(x) + g(z) \quad \text{s.t. } Ax + Bz = c$$

- 라그랑주(증강)

$$x^{k+1} = \arg \min_x (f(x) + \frac{\rho}{2} \|Ax + Bz^k - c + u^k\|_2^2)$$

*x-서브문제*

$$z^{k+1} = \arg \min_z (g(z) + \frac{\rho}{2} \|Ax^{k+1} + Bz - c + u^k\|_2^2)$$

*z-서브문제*

$$u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c$$

*primal residual*

*규반정도*

- 정지:  $\|r_{k+1}\|_2 \leq \epsilon_{pri}$

- 아이디어 : 제약  $Ax + Bz = c$ 가 있는 분할가능 문제를, 증강 라그랑지안으로 만들고  $x$ -서브문제,  $z$ -서브문제를 번갈아 푼 뒤 라그랑주 승수(이중변수) 갱신

## 예산을 맞춰야 하는 두 명의 쇼핑객(Divide and Conquer)

- 알고리즘(위의 식은 프라이멀 잔차)
  1.  $x^0, z^0, u^0$  초기화, 벌점  $\rho > 0$  설정.
  2. **x-업데이트**: 다른 변수 고정,  $x$  부분문제 최소화.
  3. **z-업데이트**: 새  $x$  고정,  $z$  부분문제 최소화.
  4. 이중 갱신:  $u \leftarrow u + (\text{제약 잔차})$
  5. 잔차(듀얼/프라이멀)가 허용 오차 이하이면 종료
- 장점
  - 큰 문제를 분할정복하여, 작은문제로 풀기에 병렬/분산으로 풀기 좋음
  - 대규모 볼록 최적화에서 적용가능
- 단점
  - 수렴 속도와 정확도가  $\rho$ 에 민감, 튜닝이 어렵다
  - 잔차해석, 정지조건 필요

## 10. Frank-Wolfe (Conditional Gradient)

$$s_t = \arg \min_{s \in D} (\nabla f(\theta_t), s)$$

$$\theta_{t+1} = (1 - \gamma_t) \theta_t + \gamma_t s_t$$

$\gamma_t$  : 선택색      or       $\frac{2}{t+2}$

- 아이디어 : 제약집합  $D$ (볼록·콤팩트) 위에서, 현재 점에서 목적함수를 선형 근사한 뒤 그 선형문제 해 방향으로 볼록 결합 이동.

### 울타리 쳐진 공원에서 물건찾기

울타리 끝까지 갔을 때의 지점을 목표로 삼습니다. 그리고 현재 위치와 목표 지점을 잇는 직선 경로 상에서 가장 낮은 지점까지만 이동하는 것을 반복

- 알고리즘
  - $\theta_0 \in D$  로 시작.
  - 반복
    - 선형화 문제로  $s_t$  계산(보통 선형계획이 쉬운  $D$ 에서 유리).
    - 스텝  $\gamma_t$  정하고 볼록 결합으로 이동.
    - 수렴 시 종료
  - 장점

- 선형 제약/단체(simplex)/트레이스 노름 볼 등에서 투영(projection) 불필요.
  - 초기 수렴이 빠른 편
  - 단점
    - 최적 부근에서 느린(아차원) 수렴.
    - 최적에 가까우면  $\nabla f$  와 탐색방향이 직교 경향 → 비효율.
- 

## 포인트

구분	설명	대표 특징
1차 최적화	Gradient Descent 계열	단순, 확장성 높음
고차 최적화	Newton, Quasi-Newton 계열	수렴 빠름, Hessian 필요
미분 불필요	Evolutionary, Random search	미분 불가 문제 대응
사전조건화	스펙트럼 개선, 반복 횟수 감소	실제 문제에서 효과적

## A. First-Order Methods

구분	핵심 요소	최적화 목표	계산 복잡도	난제
배치 GD	모든 데이터 사용	손실 최소화	$O(ND)$	대규모 데이터 불가
SGD	무작위 샘플 1개	손실 근사 최소화	$O(D)$	진동, 학습률 문제
MSGD	b개 샘플 사용	안정된 수렴	$O(bD)$	b 선택 문제
공통	학습률 $\eta$	빠른/안정적 수렴	—	튜닝 필요
심화	안장점	전역 해 탐색	—	탈출 전략 필요

범주	핵심 개념	업데이트/수식(원문 표기 기준)	하이퍼파라미터	장단점/주의
모멘텀	관성 유지로 가속	$v = \eta(-\partial L/\partial \theta) \cdot mtm + v_{old}$	$mtm \approx 0.9$	진동 감소·가속 / 과대면 뛰어넘음
NAG	미래 위치 기울기 사용	$\sim\theta = \theta + v_{old} \cdot mtm;$ $v = v_{old} \cdot mtm + \eta(-\partial L(\sim\theta)/\partial \theta);$ $\theta' = \theta + v$	$mtm$	수렴가속 $O(1/k^2)$ / 구현 주의
LR decay	$\eta_t$ 스케줄	$\eta_t = \eta_0 / (1 + d \cdot t)$	$d \in [0, 1]$	진동 완화 / d 튜닝 필요

범주	핵심 개념	업데이트/수식(원문 표기 기준)	하이퍼파라미터	장단점/주의
AdaGrad	누적 기울기 기반 $\eta$	$\theta_{t+1} = \theta_t - \eta \cdot g_t / V_t$ , $V_t = \sqrt{(\sum(g_i)^2 + \epsilon)}$	$\eta, \epsilon$	튜닝 감소 / $\eta \rightarrow 0$ 문제
RMSProp/AdaDelta	EMA 창 누적	$V_t = \sqrt{\beta V_{t-1}^2 + (1-\beta)g_t^2}$	$\beta, \epsilon$	안정적 / $\beta$ 민감
Adam	1차+2차 모멘트	$m_t, V_t \text{ EMA};$ $\theta_{t+1} = \theta_t - \eta \cdot m_t / (\sqrt{V_t} + \epsilon)$	$\beta_1, \beta_2, \eta, \epsilon$	실무 표준 / 과적합 시 warmup 등 필요

범주	내용
목적	SGD의 큰 분산을 감소시켜 수렴 속도를 선형으로 향상
업데이트 규칙	$d$ 에서 오래된 $g_i^t$ 를 $g_i^t(\theta_{t-1})$ 로 교체 $\rightarrow \theta_t = \theta_{t-1} - \alpha d$
연산 복잡도	반복당 $O(D)$ 수준으로 SGD와 유사(샘플 1개만 미분)
메모리 비용	각 샘플 기울기 $g_i$ 저장 필요. 대규모 $N$ 에서 부담 큼
적용 조건	손실 smooth, 목적 볼록(예: convex linear prediction)
기대 효과	SGD 대비 빠른 수렴, 경우에 따라 배치 GD보다도 우월

범주	내용
구간 평균 $\tilde{\mu}$	매 $w$ 스텝마다 $\tilde{\theta}$ 에서 전체 데이터 평균 기울기 계산
보정 항	$g_{i-t}(\theta_{t-1}) - g_{i-t}(\tilde{\theta}) + \tilde{\mu}$ 로 분산 감소
업데이트 규칙	$\theta_t = \theta_{t-1} - \eta \cdot (g_{i-t}(\theta_{t-1}) - g_{i-t}(\tilde{\theta}) + \tilde{\mu})$
연산량	1 스텝당 최대 2회 기울기 계산(기준점/샘플)
메모리	전체 $g_i$ 저장 불필요(SAG 대비 절약)
수렴	선형 수렴(조건 하), 분산 상계 지속 감소
적용	비볼록 신경망 포함 복잡 모델에 효율적(실험 보고)

범주	내용
문제 형식	$\min \theta_1(x) + \theta_2(y) \text{ s.t. } Ax + By = b, x \in X, y \in Y$
증가형 라그랑주	$L_\beta = \theta_1 + \theta_2 - \lambda^\top (Ax + By - b) + (\beta/2)$
ADMM 갱신(식 (30))	x-서브문제: $y^t$ 고정, $Con_x$ 사용 / y-서브문제: $x^{t+1}$ 고정, $Con_y$ 사용 / $\lambda$ -갱신
$\beta$ 의 역할	제약 위반 패널티 크기 조절, 수렴 속도·잔차 스케일에 영향

범주	내용
장점	분해·분산 가능, 대규모 문제 실용적
유의점	정지 기준의 원·쌍대 잔차가 $\beta$ 에 의존 → $\beta$ 가 크면 수렴 판정 어려움

---

---

## 실험과정



(예시)

번역

---

내용

---

포인트

---

---

## 실험과정



(예시)

번역

---

내용

---

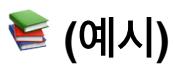
포인트

---

---



실험과정



(예시)

번역

---

내용

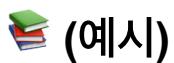
---

포인트

---



실험과정



(예시)

번역

---

내용

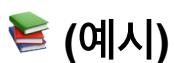
---

포인트

---



실험과정



(예시)

번역

---

**내용**

---

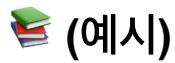
**포인트**

---

---



**실험과정**



**번역**

---

**내용**

---

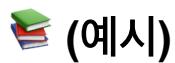
**포인트**

---

---



**실험과정**



**번역**

---

**내용**

---

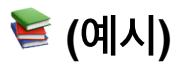
**포인트**

---

---



## 실험과정



### 번역

---

### 내용

---

### 포인트

---

---