UNIST HeXA 여름 특강

강화학습 기초

Chris Ohk

utilForever@gmail.com

근사함수

다이내믹 프로그래밍의 한계

- 계산 복잡도
- 차원의 저주
- 환경에 대한 완벽한 정보가 필요

몬테카를로, 살사, 큐러닝은 이 세 가지 문제를 다 해결했을까?

근사함수

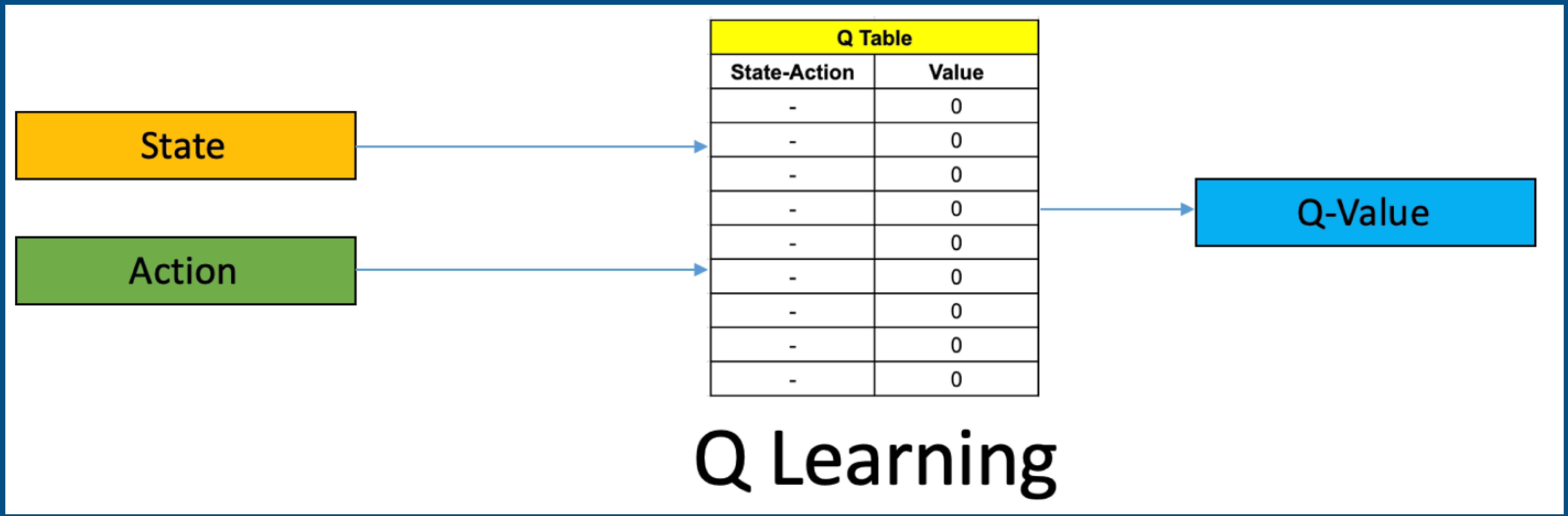
다이내믹 프로그래밍의 한계

- 계산 복잡도 → ? ? ?
- 차원의 저주 → ? ? ?
- 환경에 대한 완벽한 정보가 필요
  → 모델 프리 (환경에 대한 모델 없이 샘플링을 통해 학습)

근사함수

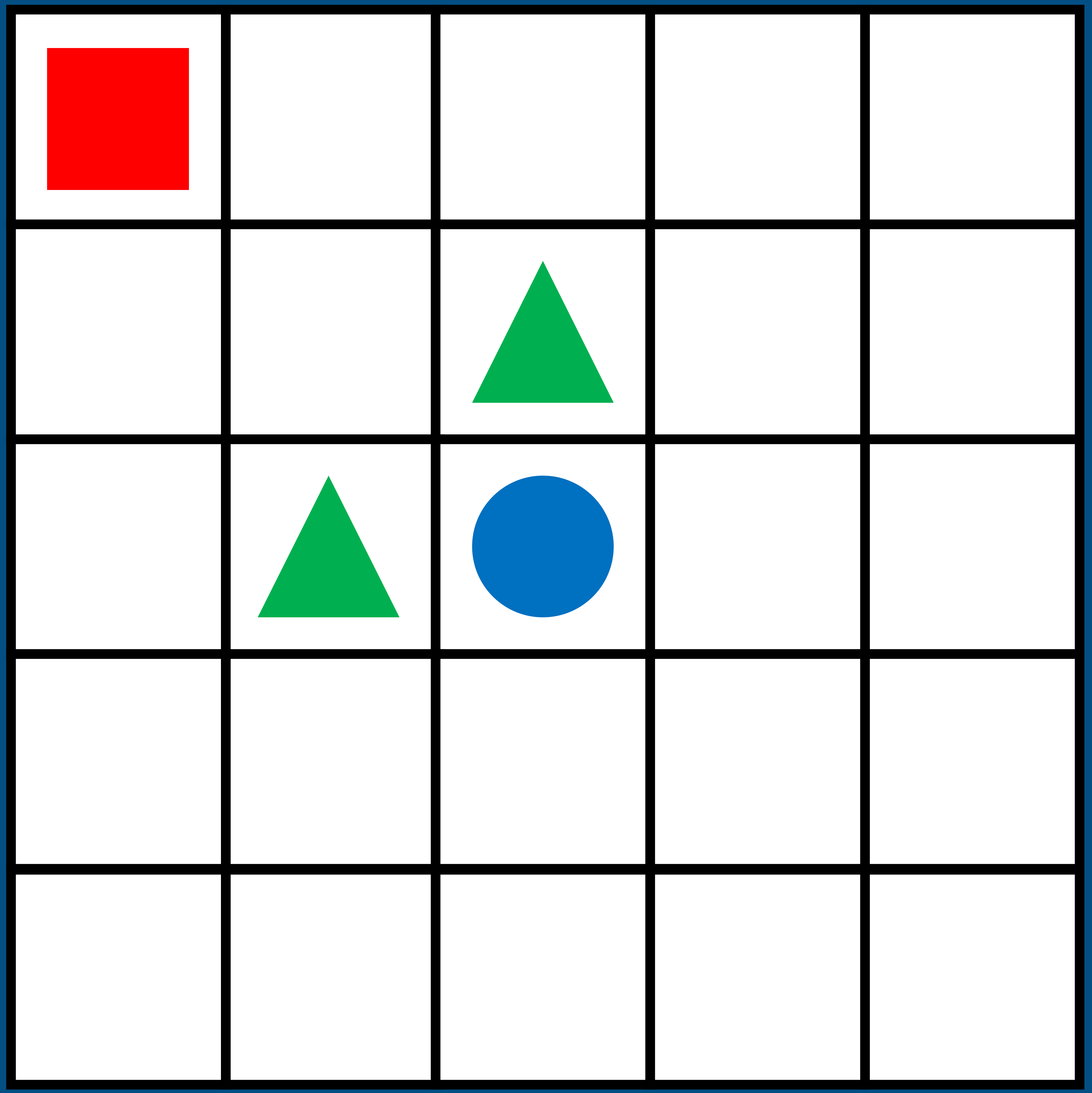
지금까지 살펴본 강화학습 알고리즘 = 테이블 형식의 강화학습

- 그리드월드의 경우 상태는 (x, y) 좌표로 2차원이었고 전체 상태의 수는 25개였다.
- 에이전트가 선택 가능한 행동이 5개였으므로 행동 상태는 125개
  → 테이블 형태로 풀 수 있다.
- 환경의 모델을 안다는 장점을 빼면
  다이내믹 프로그래밍이 훨씬 빠른 속도로 답을 찾아낸다.



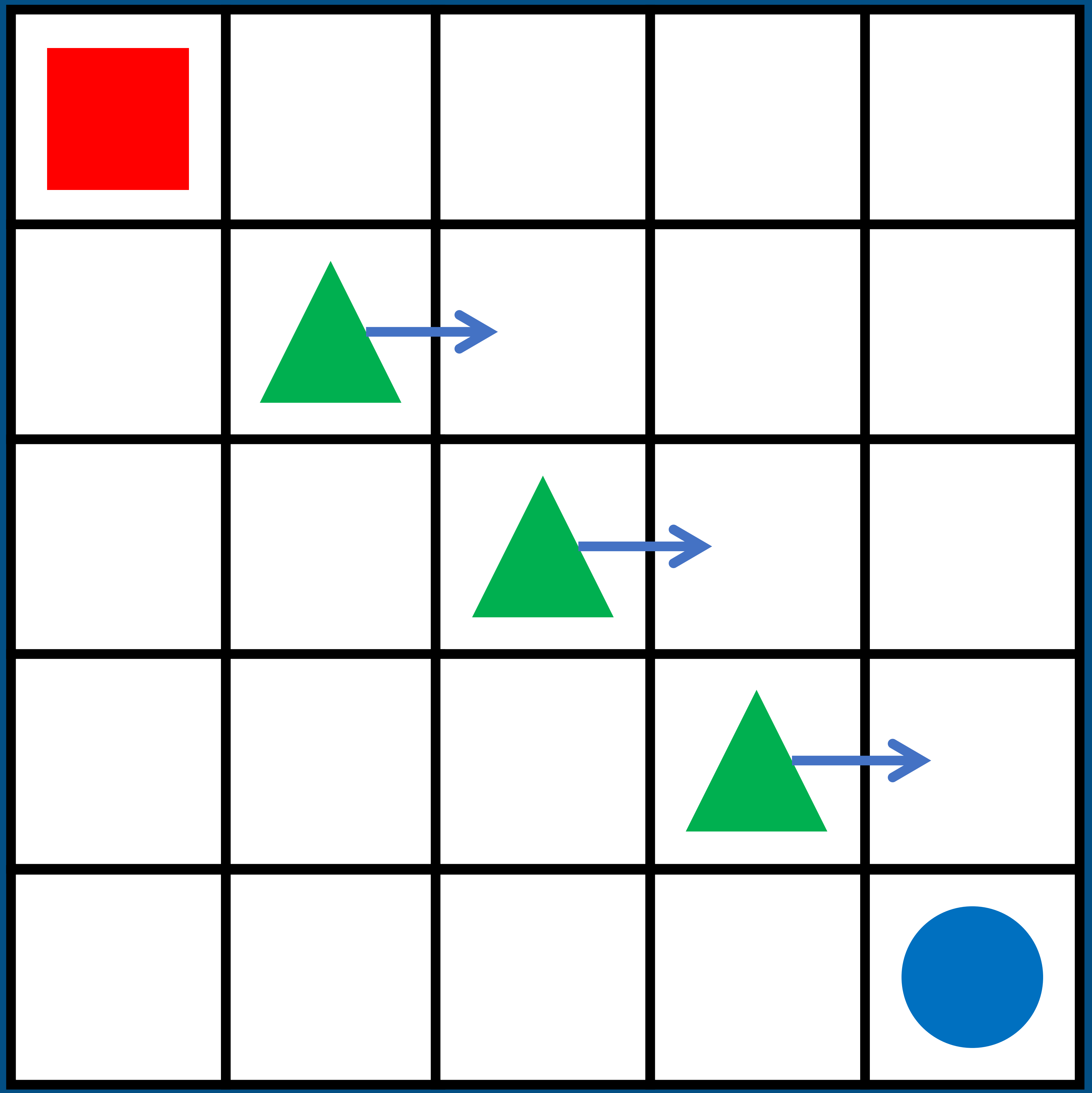
근사함수

하지만 우리가 풀고 싶은 문제는 이런 게 아니다.



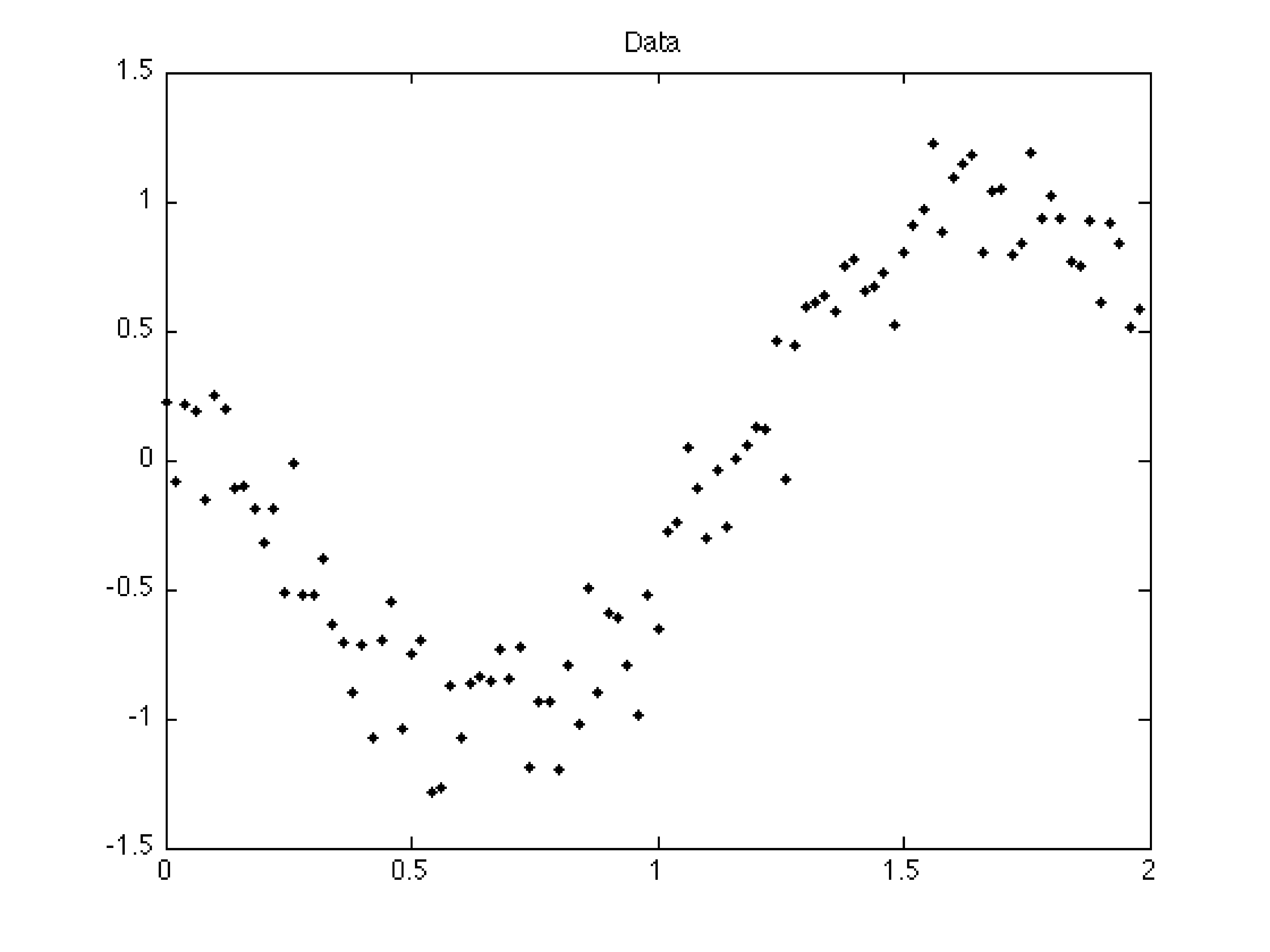
근사함수

하지만 우리가 풀고 싶은 문제는 이런 게 아니다.



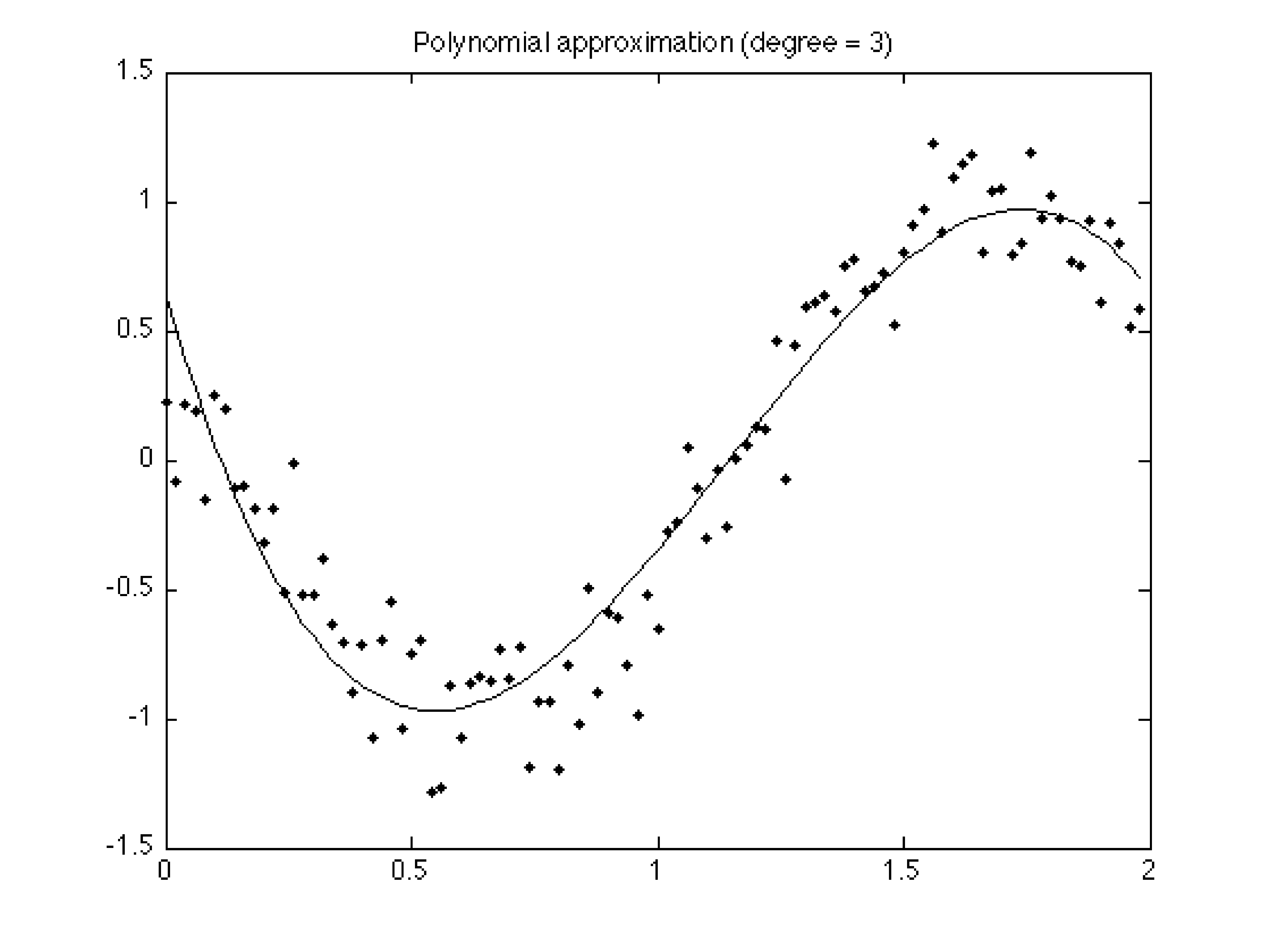
근사함수

근사함수를 통한 가치함수의 매개변수화



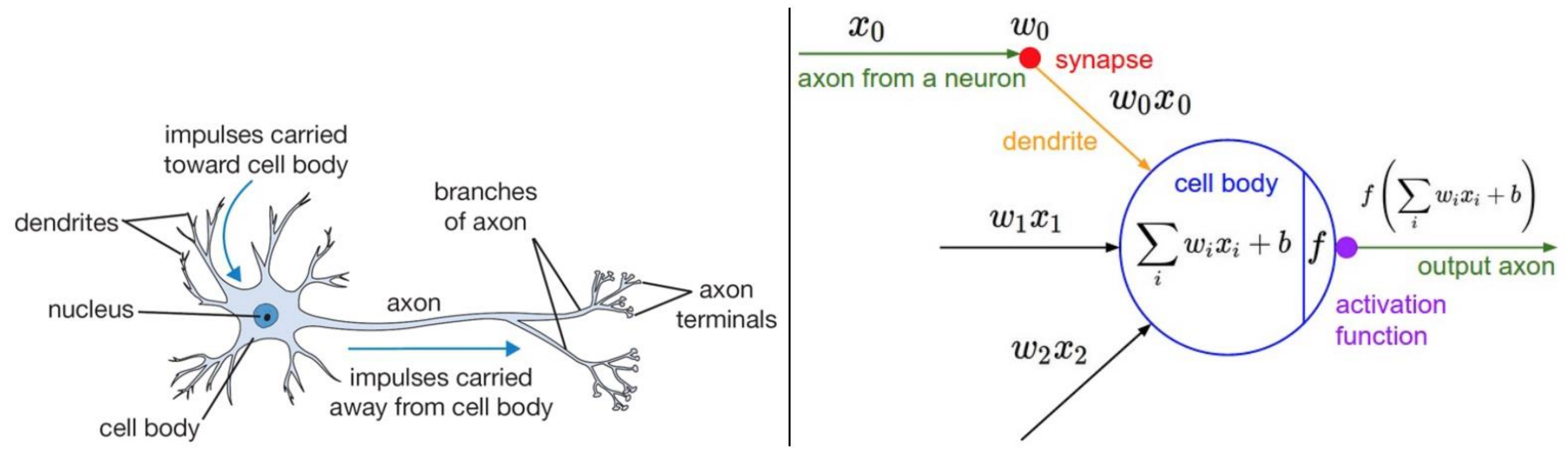
근사함수

근사함수를 통한 가치함수의 매개변수화



인공신경망

인간의 뇌를 구성하는 신경세포에서 영감을 받아 만든 수학적 모델

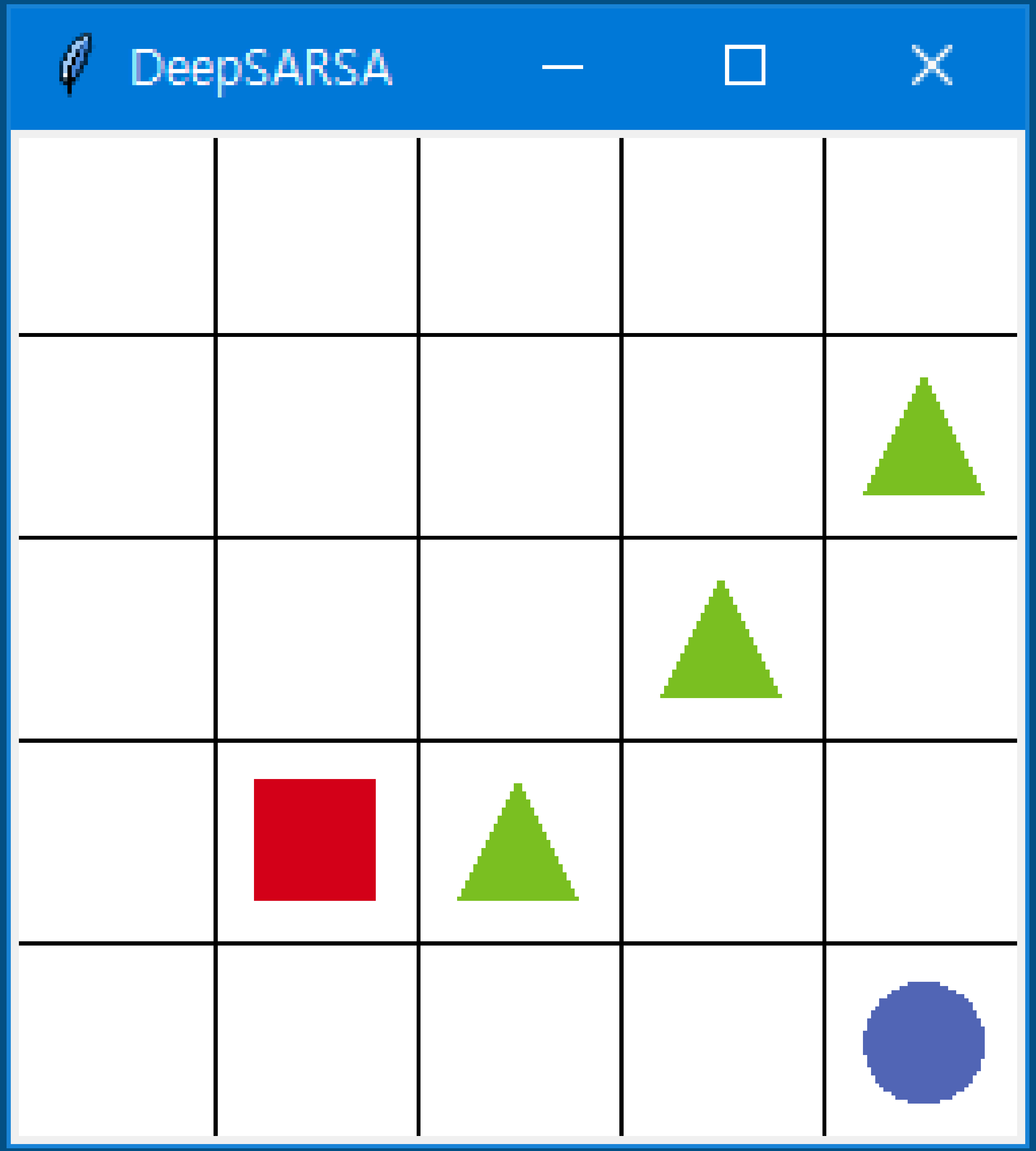


A cartoon drawing of a biological neuron (left) and its mathematical model (right).

딥살사

환경이 복잡하면 기존에 사용했던 살사 알고리즘으로 풀기는 어렵다.

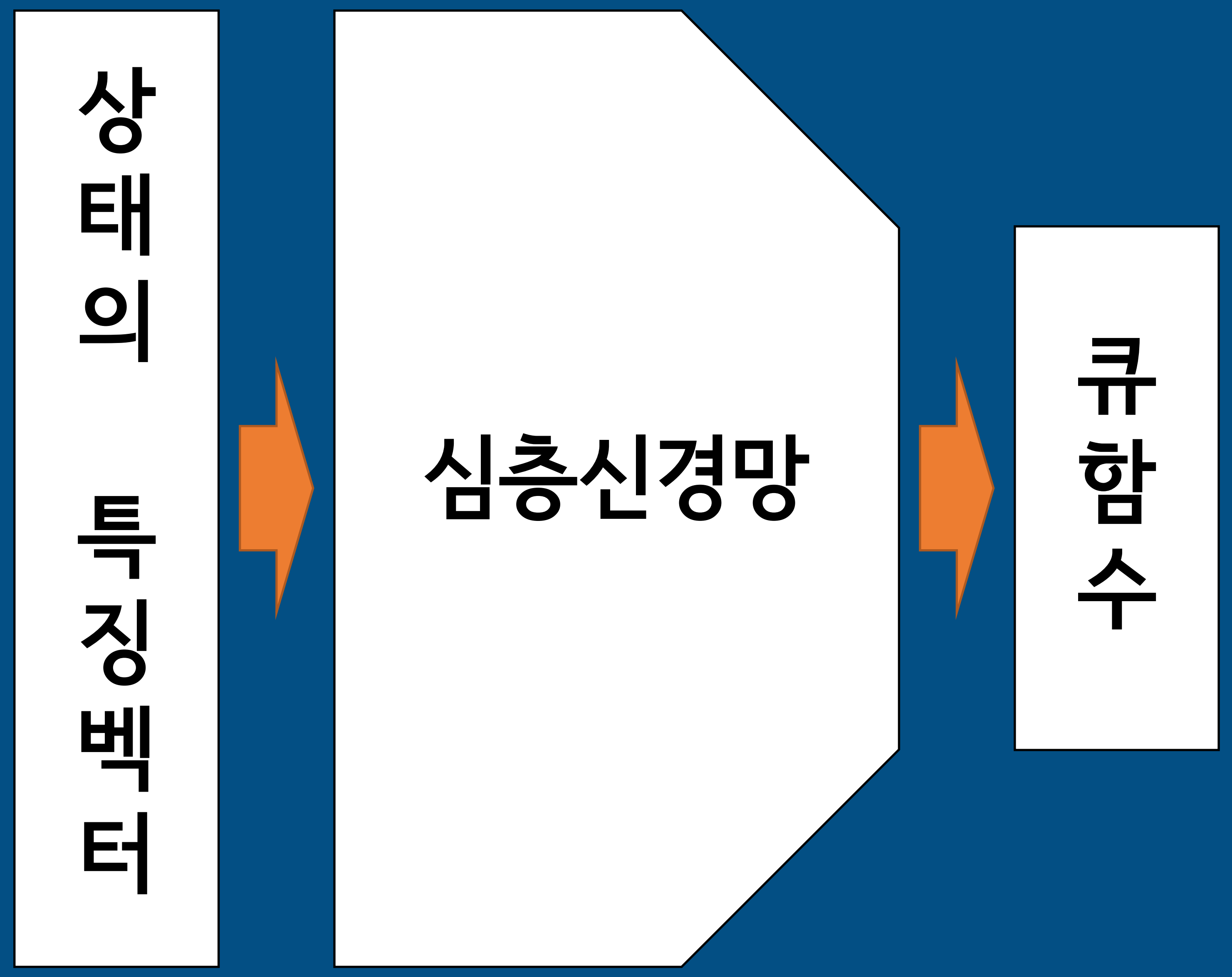
- 움직이는 세 삼각형의 경우의 수 5
- 빨간 사각형이 있을 위치의 수 25 → 총 상태의 개수 125가지



딥살사

하지만 큐함수를 인공신경망으로 근사할 수 있다.

→ 딥살사 = 살사 알고리즘 + 인공신경망



우선 MDP를 정의해야 한다.

- 상태 이외의 다른 요소는 이전의 그리드월드 예제와 유사하다.
- 따라서 상태에 대해 정의해 보자.

딥살사

상태를 정의하기 위해 어떤 정보가 필요할까?

→ 장애물을 피하려면 장애물의 상대적인 거리와 방향이 필요

따라서 상태를 다음과 같이 정의할 수 있음

- 에이전트에 대한 도착지점의 상대 위치 x, y
- 도착지점의 라벨
- 에이전트에 대한 장애물의 상대 위치 x, y
- 장애물의 라벨
- 장애물의 속도

딥살사

입력이 준비됐으니 이제 정답을 알아보자.

살사의 큐함수 업데이트 식은

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\big)$$

위 수식에서 큐함수의 업데이트에서 정답의 역할을 하는 것은

$$R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$$

예측에 해당하는 것은

$$Q(S_t, A_t)$$

딥살사

인공신경망의 출력은 값이므로 선형 함수를 사용한다.

오차함수로는 가장 많이 쓰이는 MSE를 이용해 큐함수를 업데이트한다.

$$\mathrm{MSE} = \left(\text{정답} - \text{예측}\right)^2$$

$$= \left(\mathrm{R}_{t+1} + \gamma \mathrm{Q}(\mathrm{S}_{t+1}, \mathrm{A}_{t+1}) - \mathrm{Q}(\mathrm{S}_t, \mathrm{A}_t)\right)^2$$

이제 위 오차함수를 이용해 큐함수를 업데이트 할 수 있다.

정책 기반 강화학습

지금까지의 강화학습 알고리즘은 가치 함수를 바탕으로 동작
→ 가치 기반 강화학습(Value-based Reinforcement Learning)

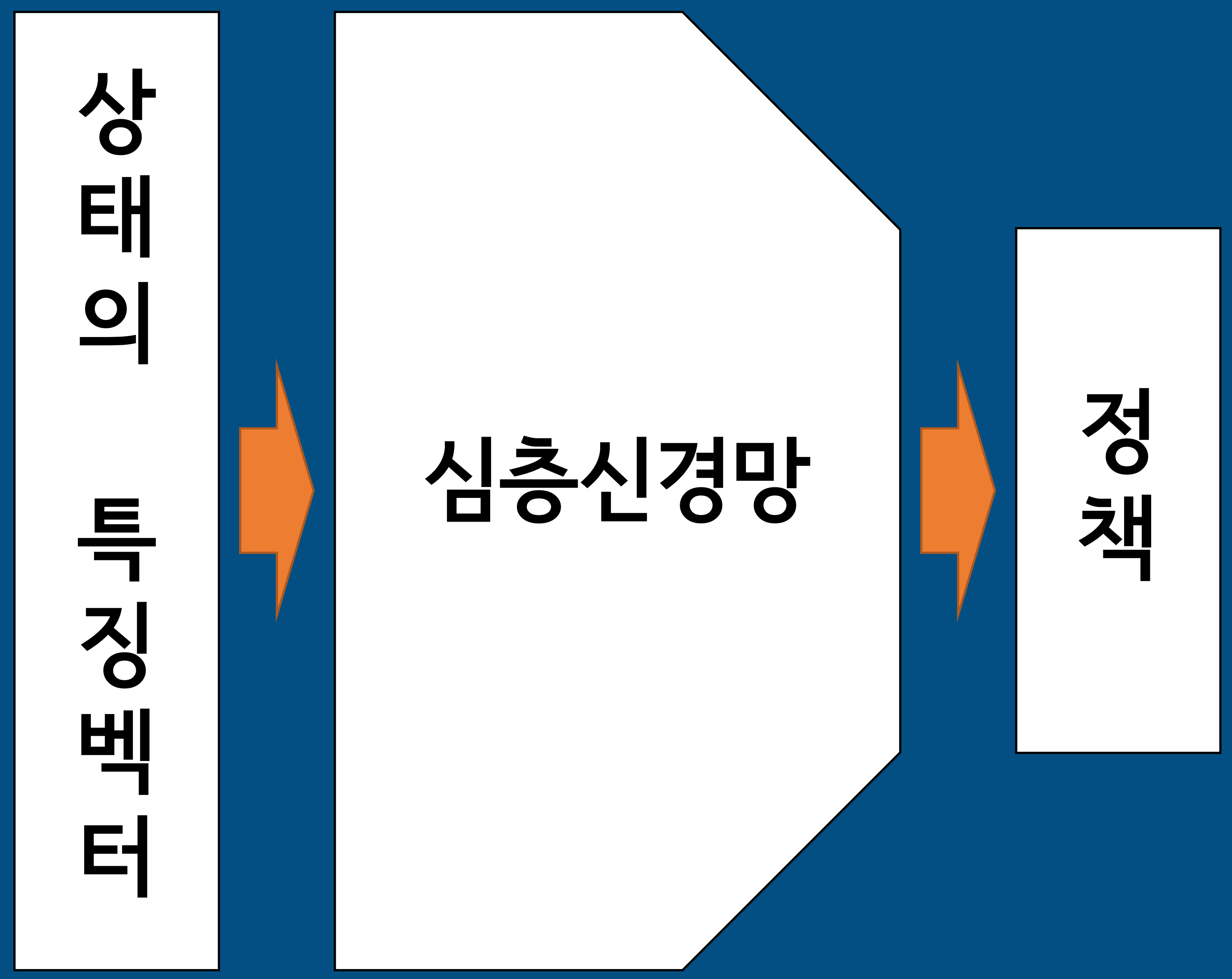
한편, 정책을 기반으로 한 강화학습 알고리즘도 생각해볼 수 있다.
→ 정책 기반 강화학습(Policy-based Reinforcement Learning)

정책 기반 강화학습

UNIST HeXA
Week 6

상태에 따라 바로 행동을 선택한다.
→ 가치함수를 토대로 행동을 선택하지 않는다.

대신 정책을 직접적으로 근사한다.



인공신경망으로 정책을 근사하고, 인공신경망의 출력은 정책이 된다.

정책 기반 강화학습

정책을 근사하는 인공신경망의 출력층 활성함수는 Softmax를 사용한다.
→ 가장 최적의 행동을 선택하는 분류 문제로 생각할 수 있다.

Softmax 함수의 식은 다음과 같다.

$$\mathrm{S}(\mathrm{y}_i) = \frac{e^{\mathrm{y}_i}}{\sum_j e^{\mathrm{y}_j}}$$

위 수식에서 $s(\mathrm{y}_i)$는 에이전트가 i번째 행동을 할 확률이 된다.

폴리시 그레이디언트

정책을 인공신경망으로 근사했기 때문에 정책을 다음과 같이 표현할 수 있다.

$$\pi_\theta(\mathrm{a}|\mathrm{s})$$

- 정책을 인공신경망으로 근사했기 때문에 θ는 정책 신경망의 가중치가 된다.
- 목표함수는 $\mathrm{J}(\theta)$로 표현할 수 있다.

폴리시 그레이디언트

강화학습의 목표는 <u>누적 보상을 최대로 하는 최적 정책</u>을 찾는 것이다.

따라서 정책 기반 강화학습의 목표를 수식으로 표현하면 다음과 같다.

$$\text{Maximize J}(\theta)$$

목표함수 $\mathrm{J}(\theta)$의 최대화는 미분을 통해 미분한 값에 따라 업데이트하면 된다.

→ 경사를 따라 올라가는 것이므로 "경사상승법"이라고 한다.

폴리시 그레이디언트

미분을 통해 정책 신경망을 업데이트해보자.

어느 시간 $t+1$에서 정책 신경망의 계수 θ_{t+1}은 다음과 같이 구할 수 있다.

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \mathrm{J}(\theta)$$

목표함수는 $\mathrm{J}(\theta) = \mathrm{v}_\pi(\mathrm{s}_0)$이므로 목표함수의 미분은 다음과 같다.

$$\nabla_\theta \mathrm{J}(\theta) = \nabla_\theta \mathrm{v}_\pi(\mathrm{s}_0)$$

폴리시 그레이디언트

계속하면…

$$\nabla_\theta J(\theta) = \sum_{s} d_{\pi_\theta}(s) \sum_{a} \nabla_\theta \pi_\theta(a|s)\, q_\pi(s, a)$$

- $d_{\pi_\theta}(s)$는 s라는 상태에 에이전트가 있을 확률
- 정책에 따라 각 상태에 에이전트가 있을 확률이 달라진다.
- 위 함수는 가능한 모든 상태에 대해 각 상태에서 특정 행동을 했을 때 큐함수의 기댓값
  → 에이전트의 선택에 대한 좋고 나쁨의 지표

폴리시 그레이디언트

계속하면…

$$\nabla_\theta J(\theta) = \sum_s d_{\pi_\theta}(s) \sum_a \pi_\theta(a|s) \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)} q_\pi(s,a)$$

log 미분으로 표현하면

$$\nabla_\theta J(\theta) = \sum_s d_\pi(s) \sum_a \pi_\theta(a|s) \times \nabla_\theta \log \pi_\theta(a|s)\, q_\pi(s,a)$$

이를 기댓값의 형태로 표현할 수 있다.

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s)\, q_\pi(s,a)]$$

폴리시 그레이디언트

최종적으로 폴리시 그레이디언트의 업데이트 식은

$$\theta_{t+1} = \theta_t + \alpha\nabla_\theta J(\theta) \approx \theta_t + \alpha[\nabla_\theta \log \pi_\theta(a|s)\, q_\pi(s,a)]$$

하지만 에이전트에 가치함수나 큐함수가 없기 때문에

$q_\pi(s, a)$를 구할 수 없다는 문제가 있다.

폴리시 그레이디언트

목표함수의 미분값 $\nabla_\theta J(\theta)$를 잘 근사하는 게 중요하다.

→ 가장 고전적인 방법으로는 큐함수를 반환값 G_t로 대체하는 방법이 있다.
이를 REINFORCE 알고리즘이라고 한다.

REINFORCE 알고리즘의 업데이트 식은 다음과 같다.

$$\theta_{t+1} \approx \theta_t + \alpha[\nabla_\theta \log \pi_\theta(a|s) G_t]$$

- REINFORCE 알고리즘에선 실제로 얻은 보상으로 학습한다.
  → 몬테카를로 폴리시 그레이디언트라고도 한다.

폴리시 그레이디언트

오차함수의 관점에서 보자.

분류 문제에서 가장 많이 쓰이는 오류함수인 크로스 엔트로피는 다음과 같다.

$$-\sum_i \mathrm{y}_i \log \mathrm{p}_i$$

- y_i와 p_i가 얼마나 가까운지를 나타낸다.
- y_i와 p_i가 같아지면 식의 값은 최소가 된다.
- 지도학습에선 y_i는 정답, p_i는 예측값을 사용한다.

폴리시 그레이디언트

REINFORCE의 오류함수는 다음과 같다.

$$\log \pi_\theta(\mathrm{a}|\mathrm{s})\mathrm{G}_t$$

- 크로스 엔트로피와 위 수식은 비슷하게 생겼다.
- $\log \pi_\theta(a|s)$는 실제로 한 행동을 정답으로 둔 것임
- 하지만 잘못된 선택을 할 수도 있어 반환값을 곱해준다.
  → 부정적인 보상을 받게 됐다면 그 행동을 선택할 확률을 낮춘다.

감사합니다!
스터디 듣느라 고생 많았습니다.