上海海事大学
SHANGHAI MARITIME UNIVERSITY

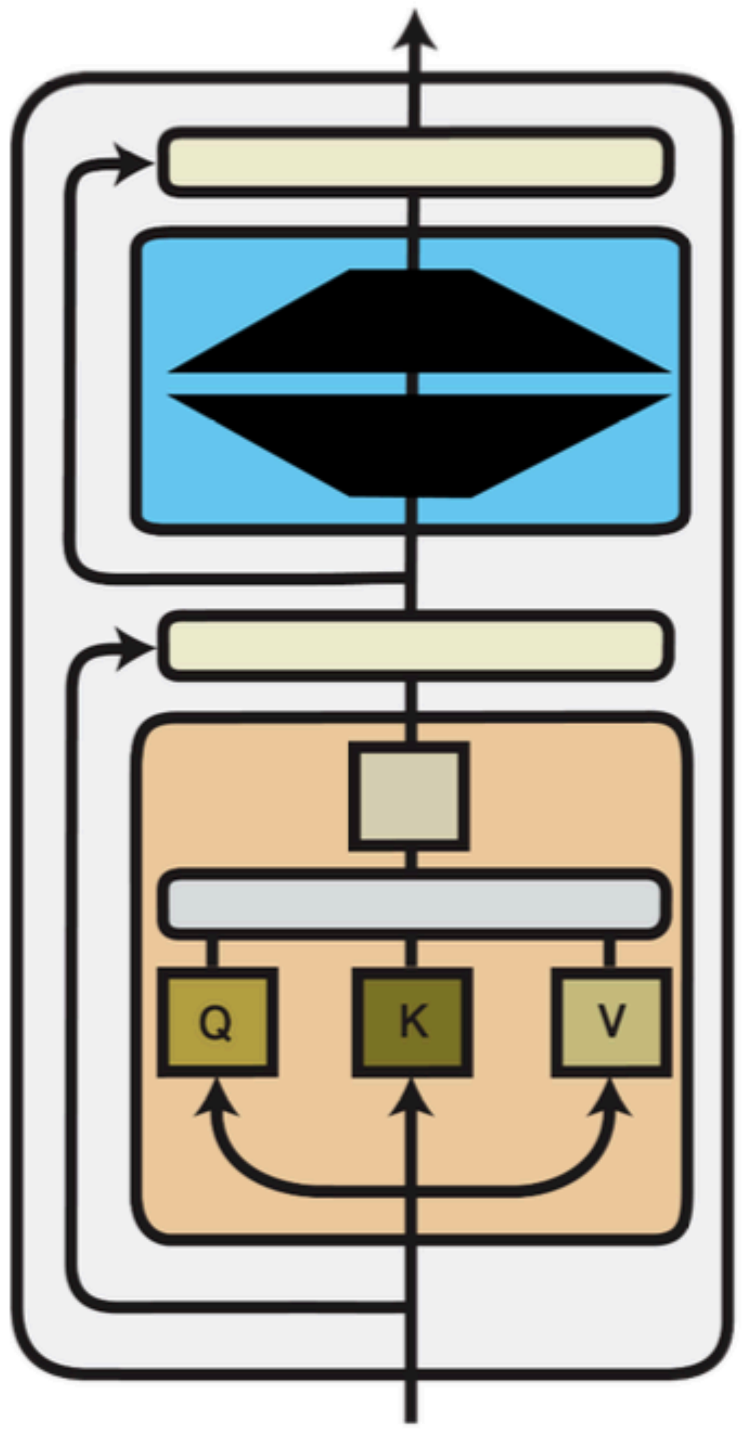
自然语言处理

2024–2025 学年第 2 学期

信息工程学院　谢雨波

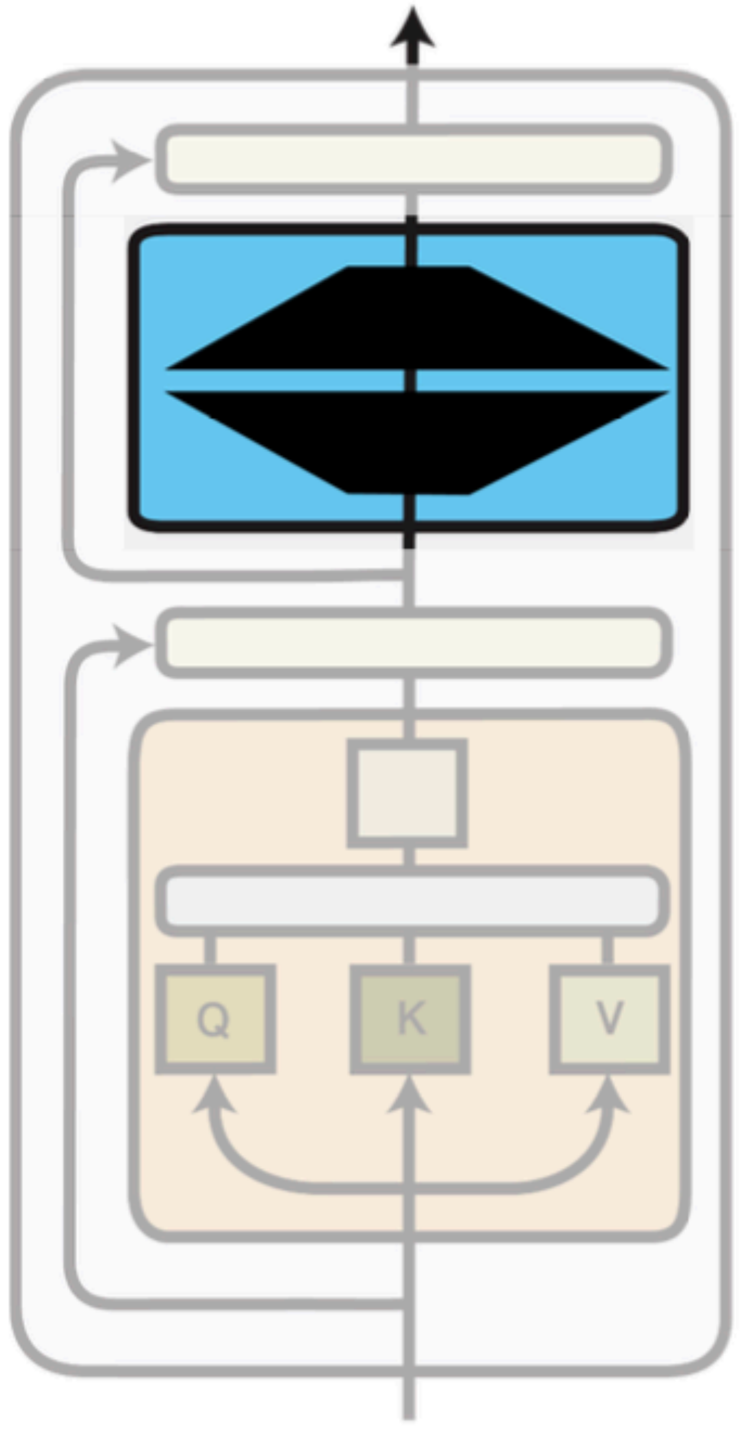
PEFT 和 LoRA

PEFT (Parameter Efficient Fine-Tuning)



整体微调

更新模型的全部参数



PEFT

只更新模型的一小部分参数

PEFT：一类高效微调的方法

为什么只微调模型的一部分参数?

1. 对于大语言模型，微调全部参数不现实（成本太高）
2. 如今的大语言模型大都是超参数化的（Over-parameterized）

- 只微调一部分参数也可以达到整体微调的效果

整体微调

- 假设有一个预训练因果语言模型 $P_\Phi(y|x)$
  - 例如基于 Transformer 解码器的 GPT 模型
- 对模型进行微调，应用于下游 NLP 任务
  - 需要训练数据集 $\{(x_i, y_i)\}_{i=1,\cdots,N}$
- 在进行**整体微调（Full Fine-tuning）**时，使用梯度下降将 Φ_0 更新至 $\Phi_0 + \Delta\Phi$

$$\max_{\Phi} \sum_{(x,y)} \sum_{t=1}^{|y|} \log P_\Phi\left(y_t | x, y_{<t}\right)$$

整体微调

- 对于每一个下游任务，都需要学习不同的 $\Delta\Phi$
  - $|\Delta\Phi| = |\Phi_0|$
  - 对于 GPT-3 来说，$|\Phi_0| = 1750$ 亿
  - 存储和部署模型的成本过高
- 如何进行优化?

LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

- LoRA: **Lo**w-**R**ank **A**daptation

- 核心思想：使用更少数目的参数 Θ 来表示 $\Delta\Phi$

$$\Delta\Phi = \Delta\Phi(\Theta), \quad |\Theta| \ll |\Phi_0|$$

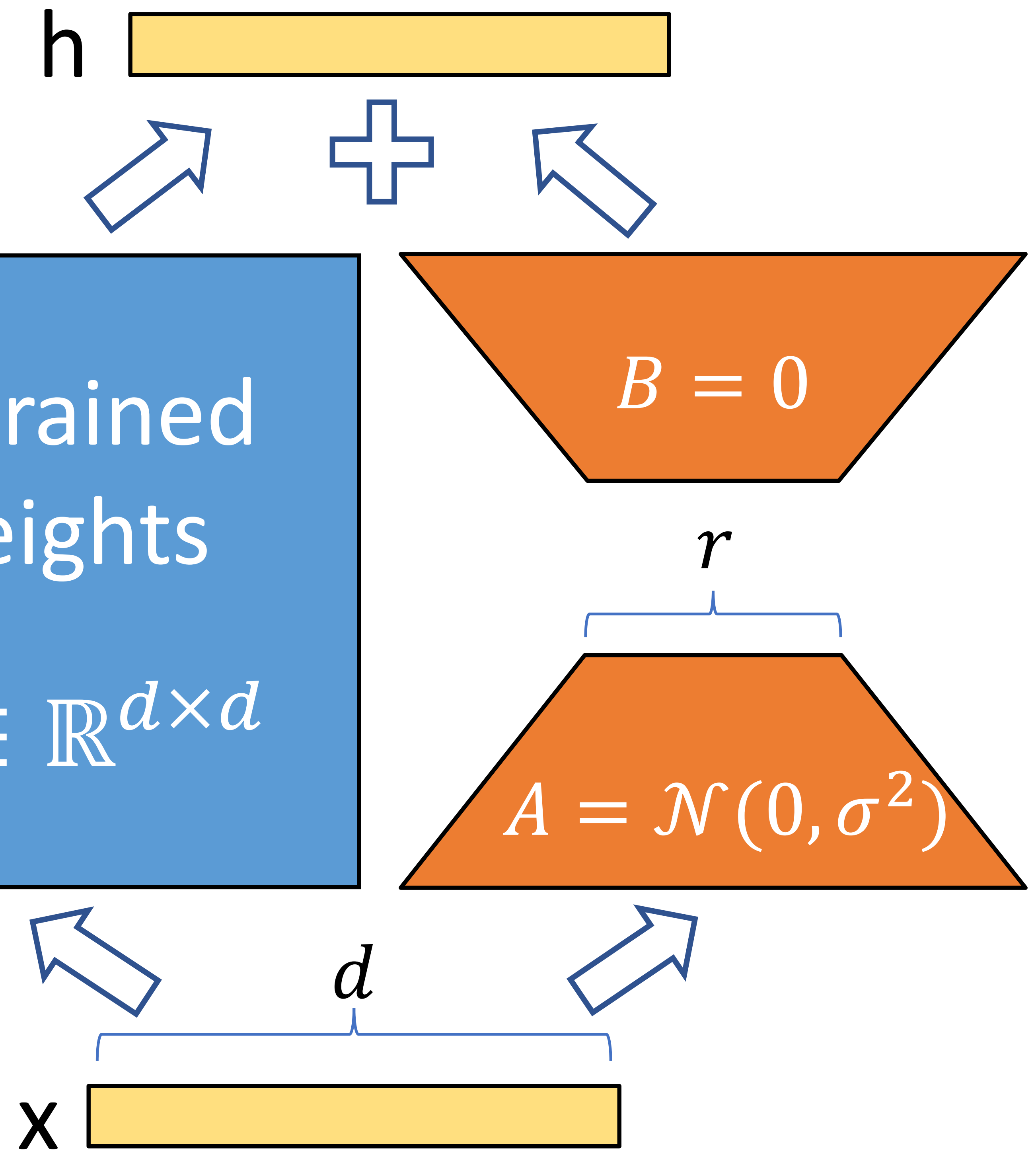
- 因此，求解 $\Delta\Phi$ 的任务可以转化为在 Θ 上进行寻优：

$$\max_{\Theta} \sum_{(x,y)} \sum_{t=1}^{|y|} \log P_{\Phi_0+\Delta\Phi(\Theta)}\left(y_t \mid x, y_{<t}\right)$$

LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

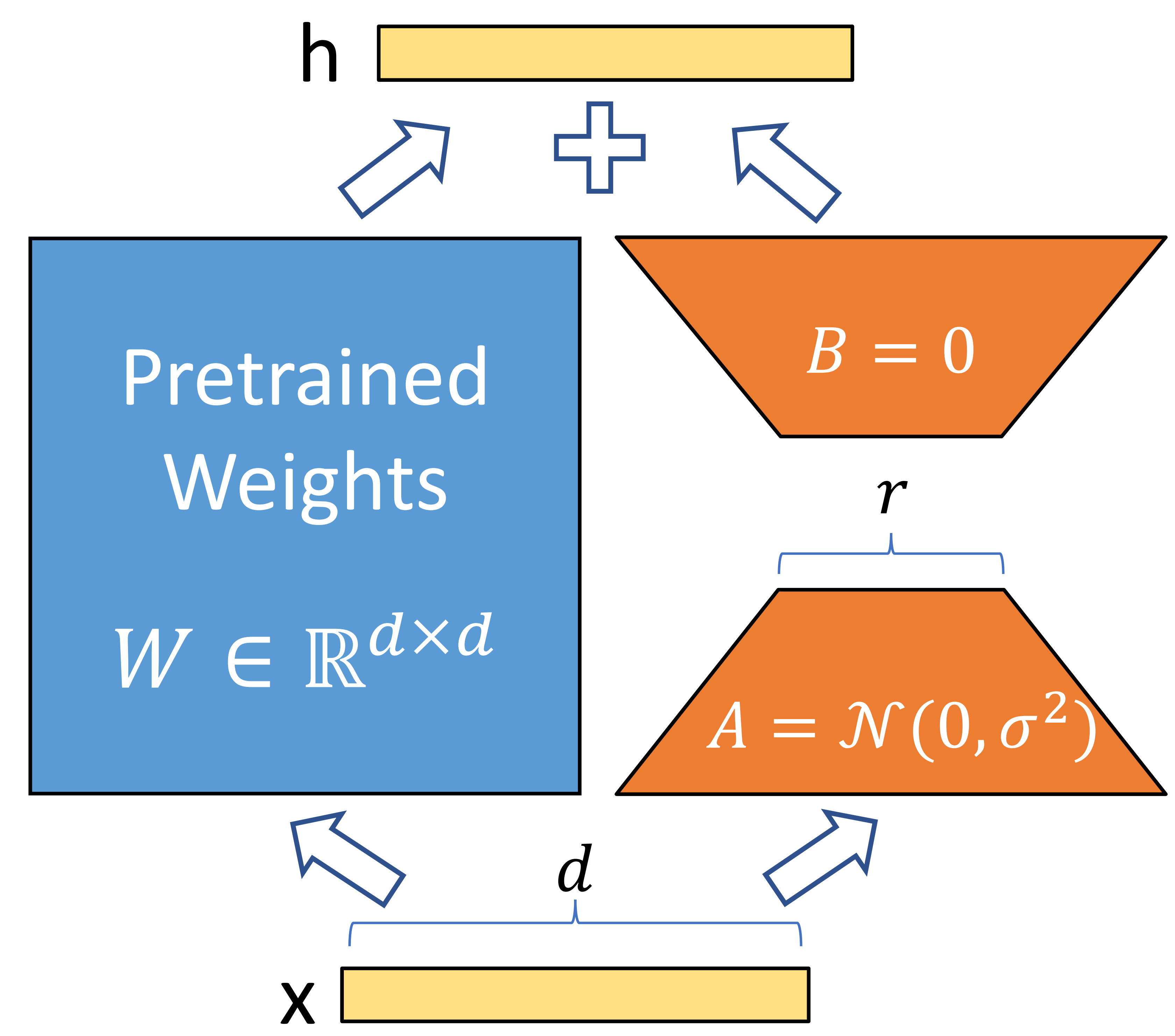
- $W_0 \in \mathbb{R}^{d\times k}$：预训练权重矩阵
- 使用一个低秩分解来限制 ΔW
  - $W_0 + \Delta W = W_0 + BA$
    - 其中 $B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k}, r \ll \min(d, k)$
  - $h = W_0x + \Delta Wx = W_0x + BAx$
- 对 ΔWx 进行缩放：$\frac{\alpha}{r}$（α 和 r 均为超参数）
- 只有 A 和 B 可训练，W_0 不参与训练



LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

- 如果逐渐增加 LoRA 中的可训练参数，则训练 LoRA 将收敛至训练原模型
- **没有额外的推理延迟**：从一个下游任务切换至另一个下游任务时，只需减去 BA 即可恢复 W_0，然后再加上一个不同的 $B'A'$
- 通常将 LoRA 应用于自注意力模块中的权重矩阵



LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

- GPT-2 + LoRA 的实验结果

Model & Method	# Trainable Parameters	E2E NLG Challenge BLEU	NIST	MET	ROUGE-L	CIDEr
GPT-2 M (FT)*	354.92M	68.2	8.62	46.2	71.0	2.47
GPT-2 M (Adapter^{L})*	0.37M	66.3	8.41	45.0	69.8	2.40
GPT-2 M (Adapter^{L})*	11.09M	68.9	8.71	46.1	71.3	2.47
GPT-2 M (Adapter^{H})	11.09M	$67.3_{\pm.6}$	$8.50_{\pm.07}$	$46.0_{\pm.2}$	$70.7_{\pm.2}$	$2.44_{\pm.01}$
GPT-2 M (FT^{Top2})*	25.19M	68.1	8.59	46.0	70.8	2.41
GPT-2 M (PreLayer)*	0.35M	69.7	8.81	46.1	71.4	2.49
GPT-2 M (LoRA)	0.35M	$\mathbf{70.4}_{\pm.1}$	$\mathbf{8.85}_{\pm.02}$	$\mathbf{46.8}_{\pm.2}$	$\mathbf{71.8}_{\pm.1}$	$\mathbf{2.53}_{\pm.02}$
GPT-2 L (FT)*	774.03M	68.5	8.78	46.0	69.9	2.45
GPT-2 L (Adapter^{L})	0.88M	$69.1_{\pm.1}$	$8.68_{\pm.03}$	$46.3_{\pm.0}$	$71.4_{\pm.2}$	$\mathbf{2.49}_{\pm.0}$
GPT-2 L (Adapter^{L})	23.00M	$68.9_{\pm.3}$	$8.70_{\pm.04}$	$46.1_{\pm.1}$	$71.3_{\pm.2}$	$2.45_{\pm.02}$
GPT-2 L (PreLayer)*	0.77M	70.3	8.85	46.2	71.7	2.47
GPT-2 L (LoRA)	0.77M	$\mathbf{70.4}_{\pm.1}$	$\mathbf{8.89}_{\pm.02}$	$\mathbf{46.8}_{\pm.2}$	$\mathbf{72.0}_{\pm.2}$	$2.47_{\pm.02}$

LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. [LoRA: Low-Rank Adaptation of Large Language Models](). ICLR 2022.

- GPT-3 + LoRA：效果达到甚至超过整体微调

Model&Method	# Trainable Parameters	WikiSQL Acc. (%)	MNLI-m Acc. (%)	SAMSum R1/R2/RL
GPT-3 (FT)	175,255.8M	**73.8**	89.5	52.0/28.0/44.5
GPT-3 (BitFit)	14.2M	71.3	91.0	51.3/27.4/43.5
GPT-3 (PreEmbed)	3.2M	63.1	88.6	48.3/24.2/40.5
GPT-3 (PreLayer)	20.2M	70.1	89.5	50.8/27.3/43.5
GPT-3 ($\text{Adapter}^{\text{H}}$)	7.1M	71.9	89.8	53.0/28.9/44.8
GPT-3 ($\text{Adapter}^{\text{H}}$)	40.1M	73.2	**91.5**	53.2/29.0/45.1
GPT-3 (LoRA)	4.7M	73.4	**91.7**	**53.8/29.8/45.9**
GPT-3 (LoRA)	37.7M	**74.0**	**91.6**	53.4/29.2/45.1

LoRA

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

- 将 LoRA 应用于 W_q 和 W_v 时，效果最好

	# of Trainable Parameters = 18M						
Weight Type Rank r	W_q 8	W_k 8	W_v 8	W_o 8	W_q, W_k 4	W_q, W_v 4	W_q, W_k, W_v, W_o 2
WikiSQL ($\pm 0.5\%$)	70.4	70.0	73.0	73.2	71.4	**73.7**	**73.7**
MultiNLI ($\pm 0.1\%$)	91.0	90.8	91.0	91.3	91.3	91.3	**91.7**

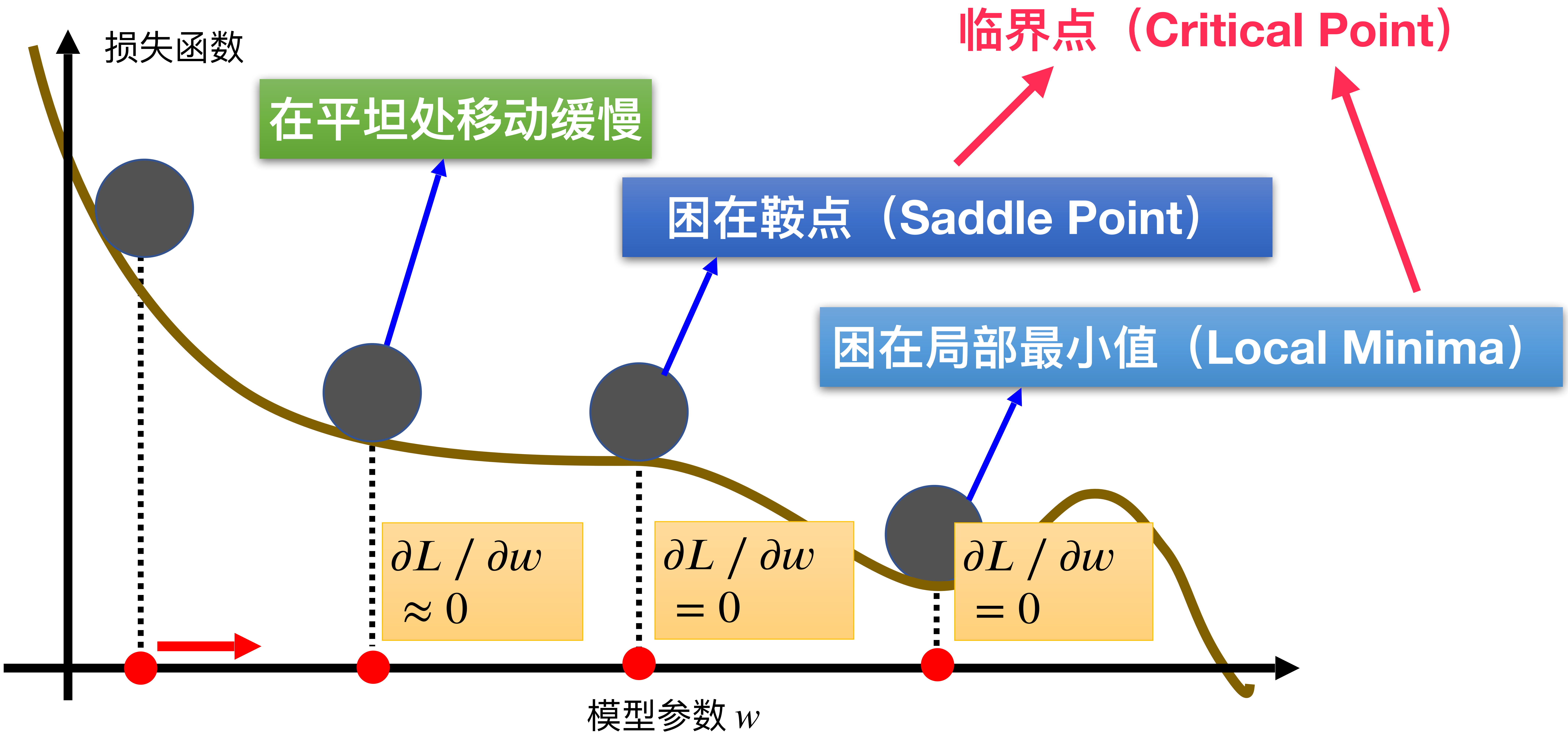
- 即使 r 值很小，效果依旧可观

	Weight Type	$r = 1$	$r = 2$	$r = 4$	$r = 8$	$r = 64$
WikiSQL($\pm 0.5\%$)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI ($\pm 0.1\%$)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

优化器

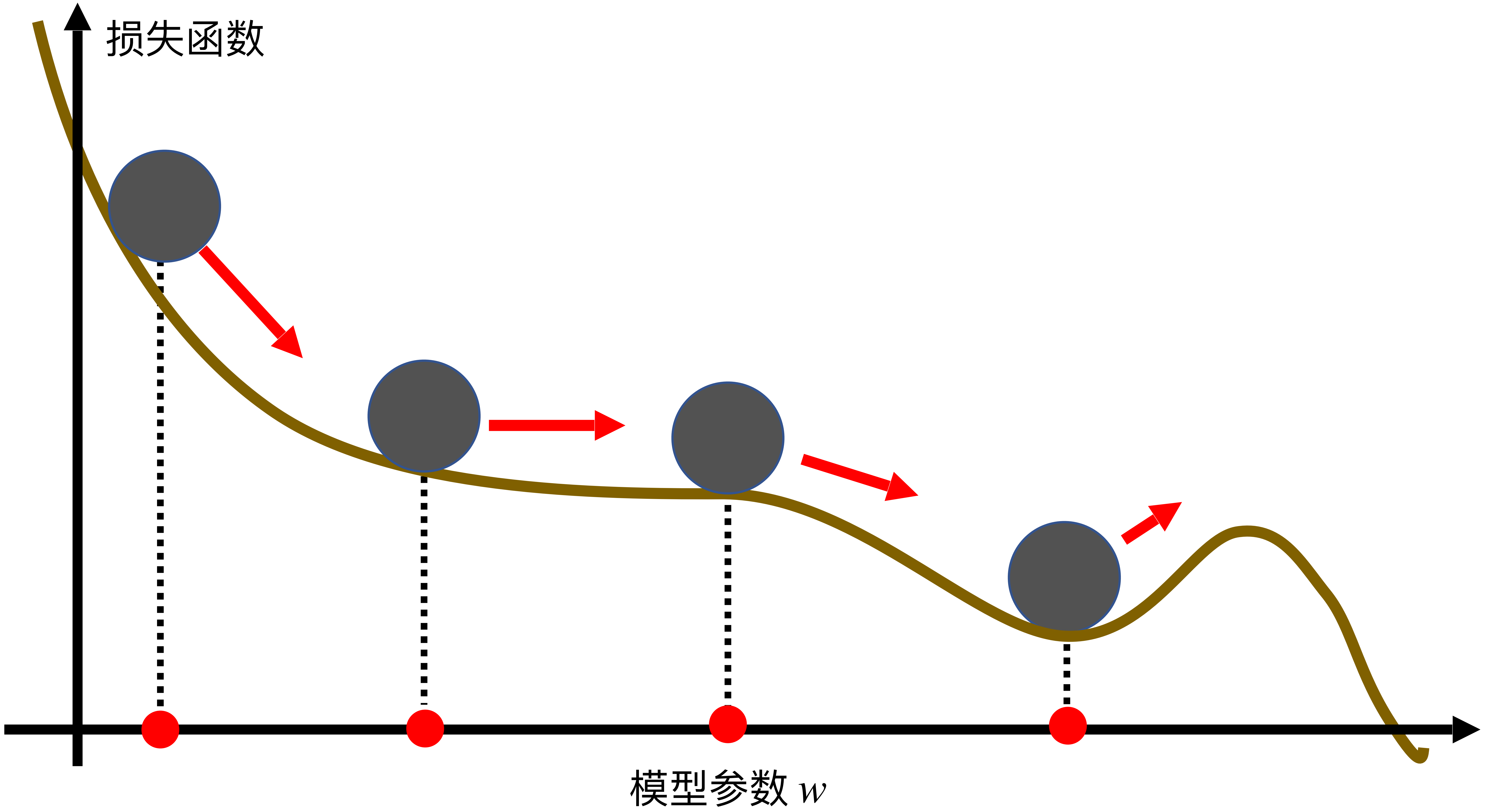
参考：李宏毅《机器学习》2021 春

梯度下降

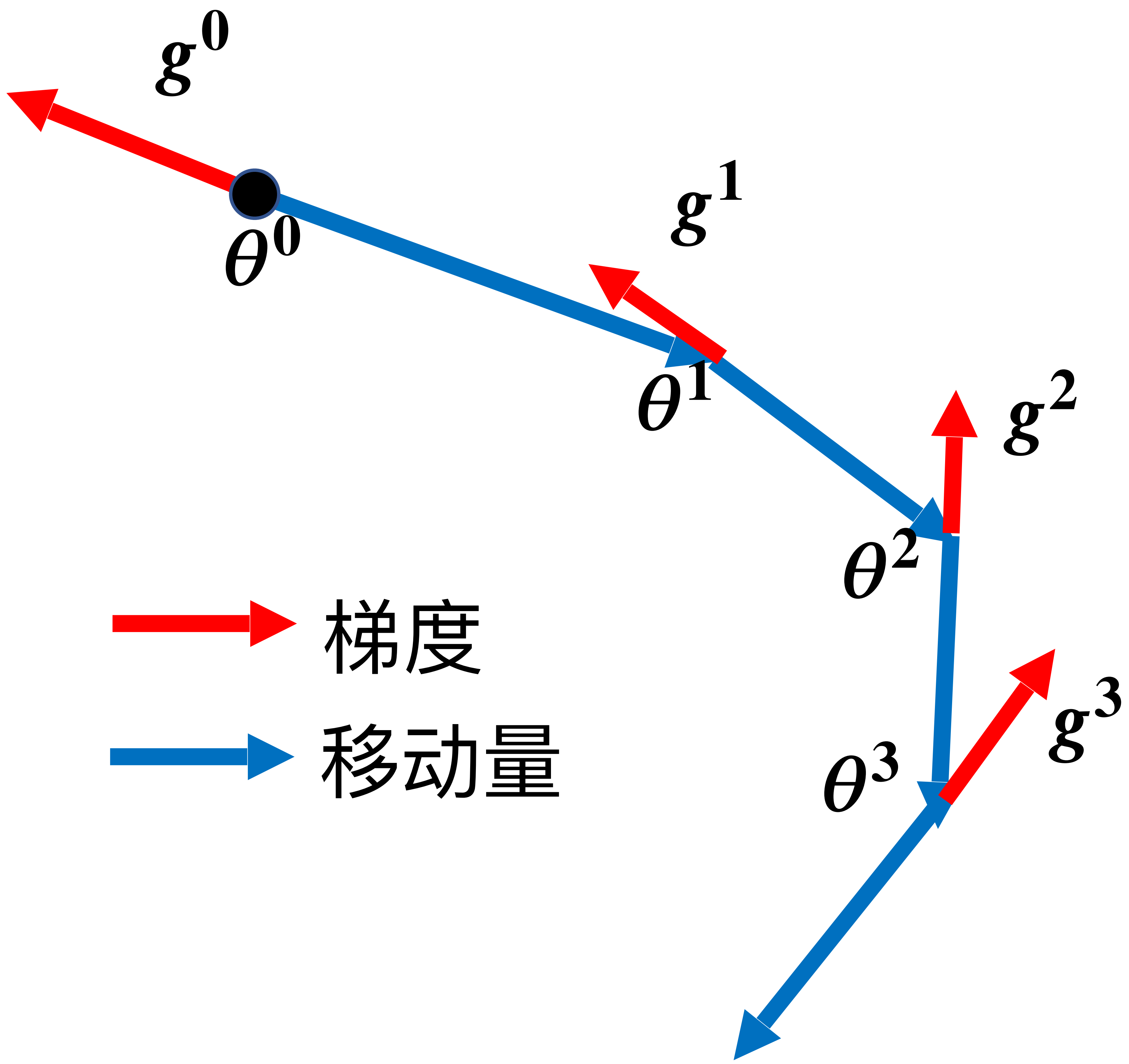


损失函数
在平坦处移动缓慢
临界点（Critical Point）
困在鞍点（Saddle Point）
困在局部最小值（Local Minima）
$\partial L / \partial w \approx 0$
$\partial L / \partial w = 0$
$\partial L / \partial w = 0$
模型参数 w

物理世界中的动量（Momentum）



常规梯度下降



初始位置： θ^0

计算梯度 $\boldsymbol{g}^0$

移动到 $\theta^1 = \theta^0 - \eta \boldsymbol{g}^0$

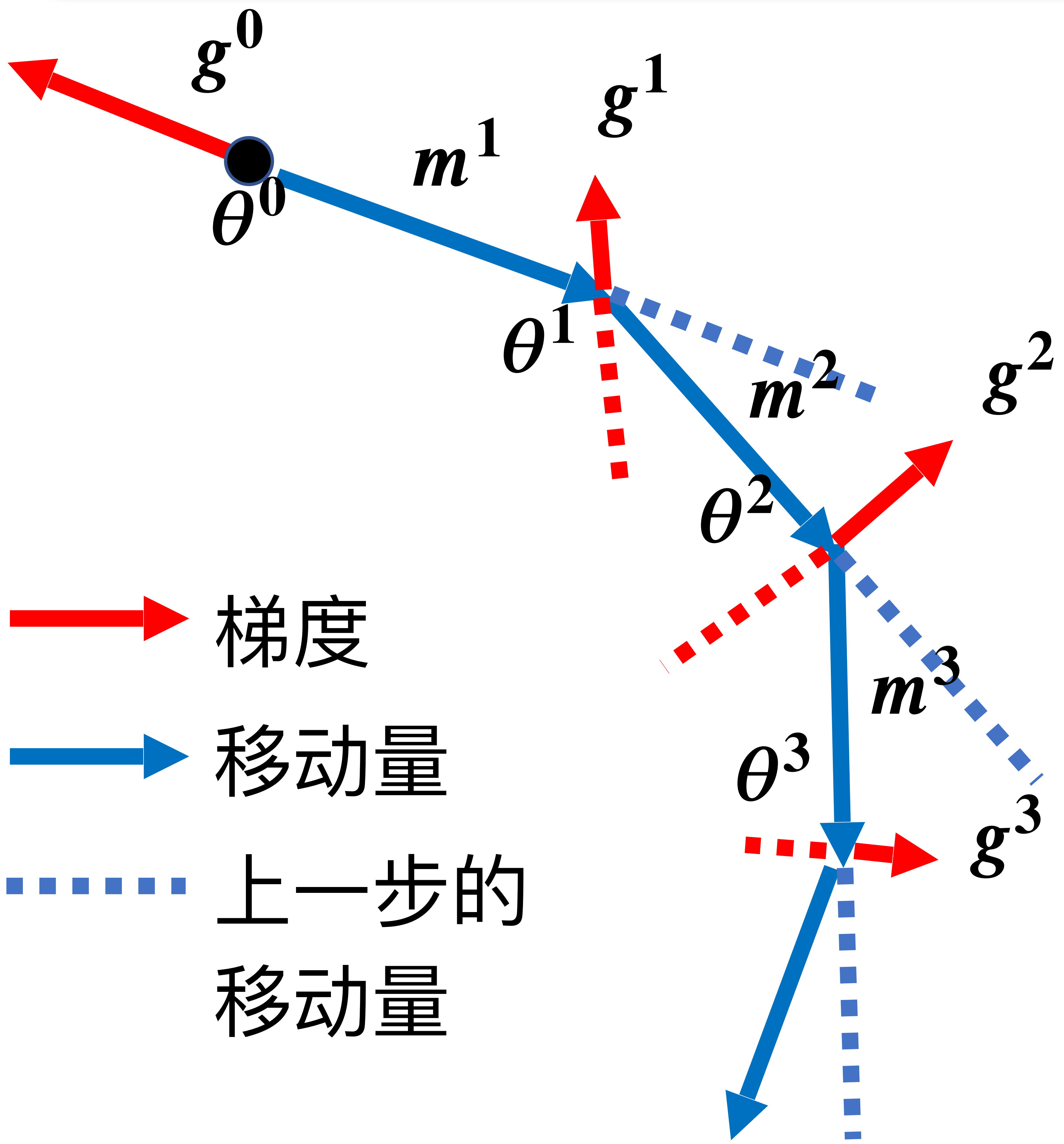
计算梯度 $\boldsymbol{g}^1$

移动到 $\theta^2 = \theta^1 - \eta \boldsymbol{g}^1$

⋮

梯度下降 + 动量

移动量：上一步的移动量 减去当前的梯度



初始位置：θ^0

移动量 $m^0 = 0$

计算梯度 g^0

移动量 $m^1 = \lambda m^0 - \eta g^0$

移动到 $\theta^1 = \theta^0 + m^1$

计算梯度 g^1

移动量 $m^2 = \lambda m^1 - \eta g^1$

移动到 $\theta^2 = \theta^1 + m^2$

当前的移动量不仅依赖于梯度，还依赖于上一步的移动量

梯度下降 + 动量

移动量：上一步的移动量 减去当前的梯度

$\boldsymbol{m}^i$ 是之前所有梯度的加权求和：

$\boldsymbol{g}^0, \boldsymbol{g}^1, \dots, \boldsymbol{g}^{i-1}$

$$\boldsymbol{m}^0 = \boldsymbol{0}$$

$$\boldsymbol{m}^1 = -\eta \boldsymbol{g}^0$$

$$\boldsymbol{m}^2 = -\lambda \eta \boldsymbol{g}^0 - \eta \boldsymbol{g}^1$$

$$\vdots$$

初始位置：θ^0

移动量 $\boldsymbol{m}^0 = \boldsymbol{0}$

计算梯度 $\boldsymbol{g}^0$

移动量 $\boldsymbol{m}^1 = \lambda \boldsymbol{m}^0 - \eta \boldsymbol{g}^0$

移动到 $\theta^1 = \theta^0 + \boldsymbol{m}^1$

计算梯度 $\boldsymbol{g}^1$

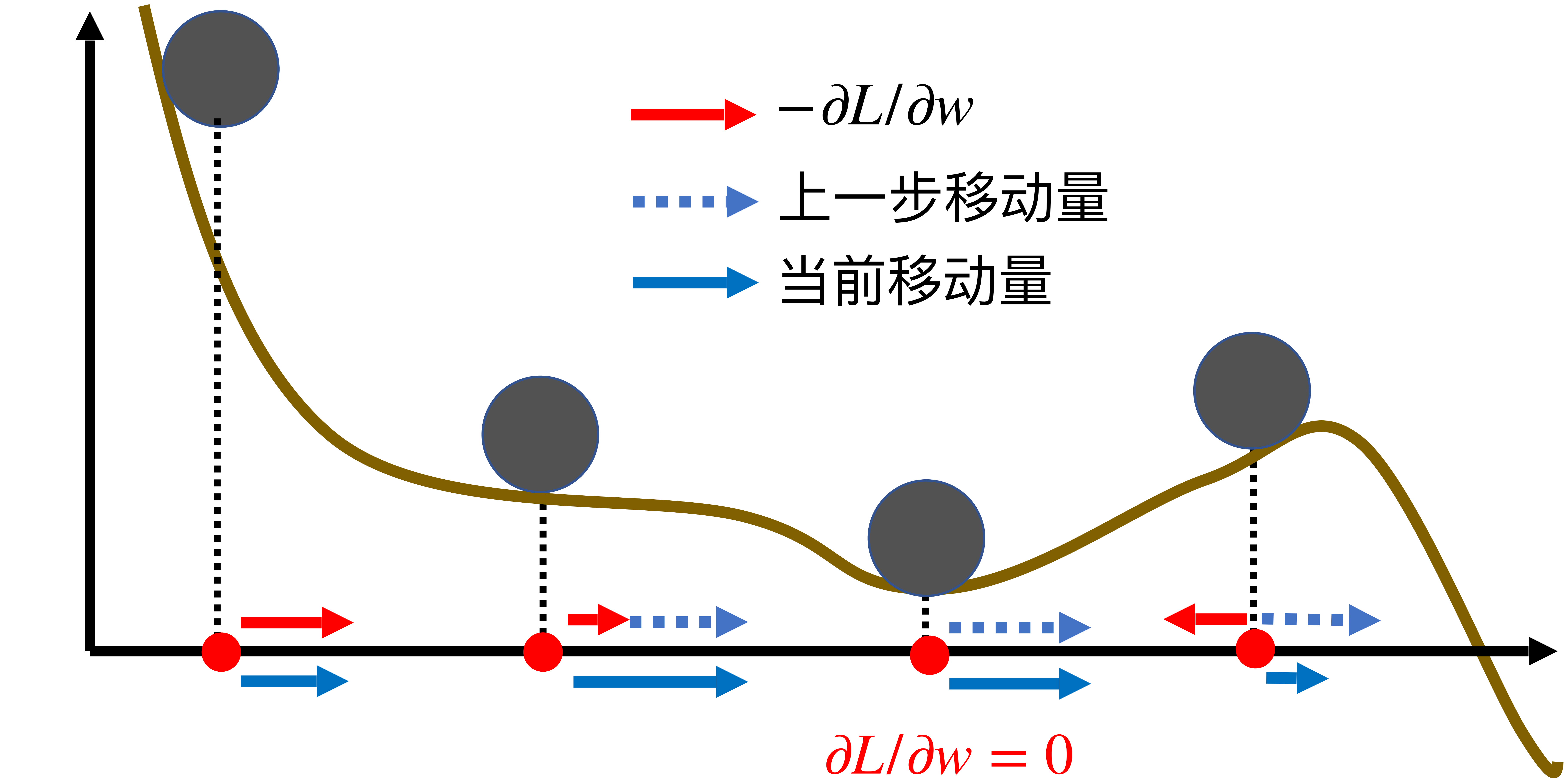
移动量 $\boldsymbol{m}^2 = \lambda \boldsymbol{m}^1 - \eta \boldsymbol{g}^1$

移动到 $\theta^2 = \theta^1 + \boldsymbol{m}^2$

当前的移动量不仅依赖于梯度，还依赖于上一步的移动量

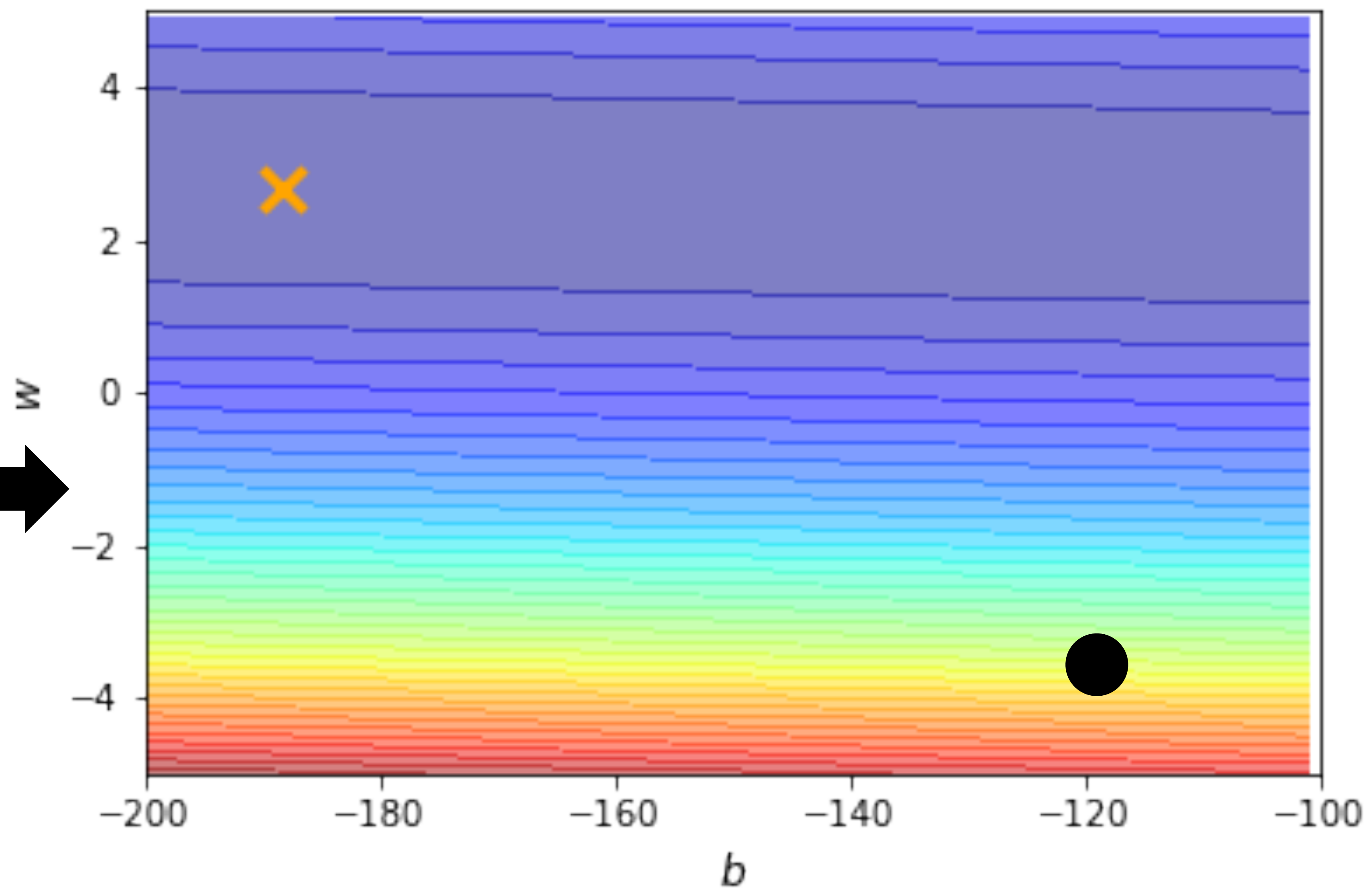
梯度下降 + 动量

移动量 $= -\partial L/\partial w$ + 上一步的移动量

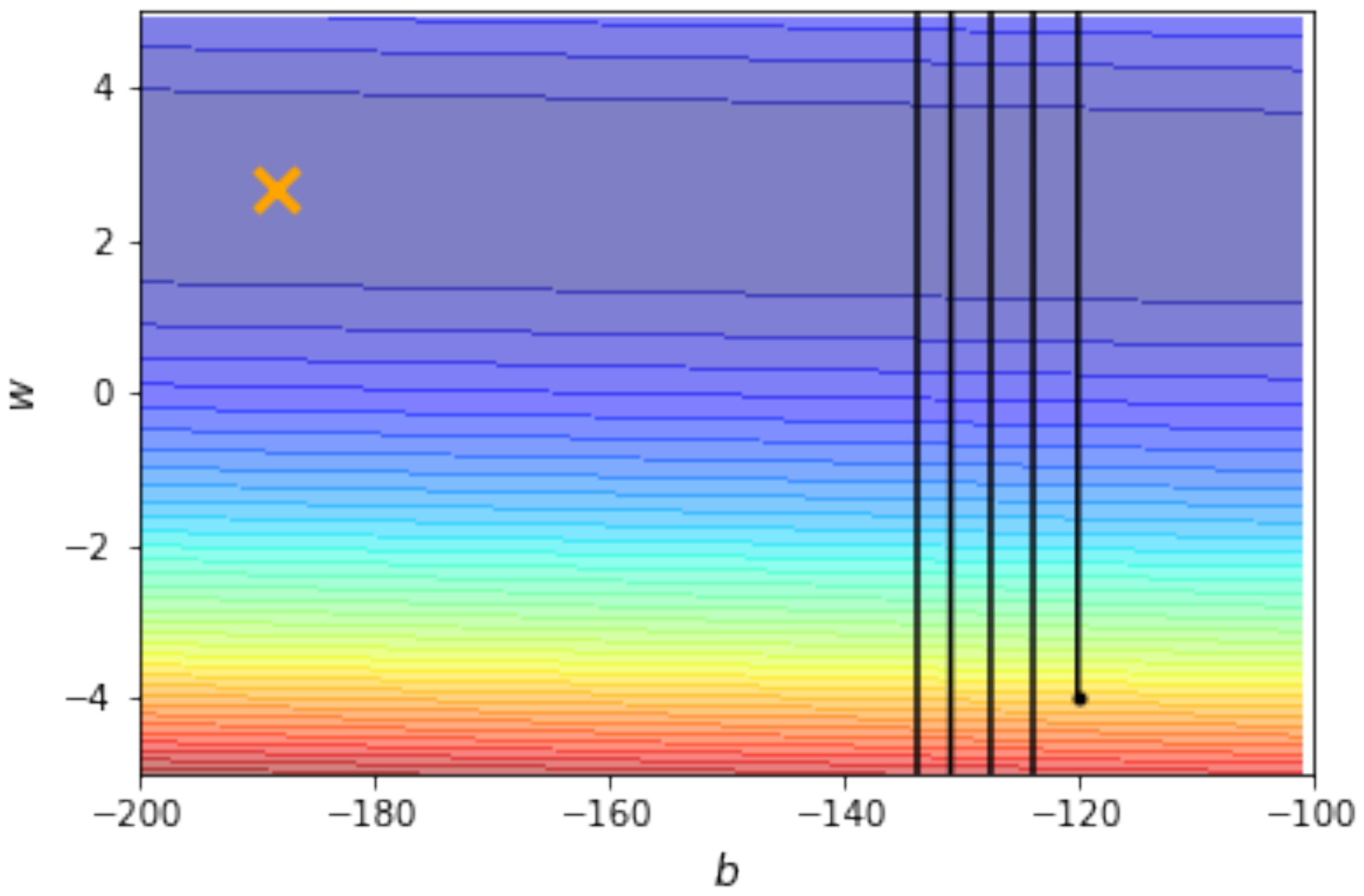


没有临界点也很难训练?

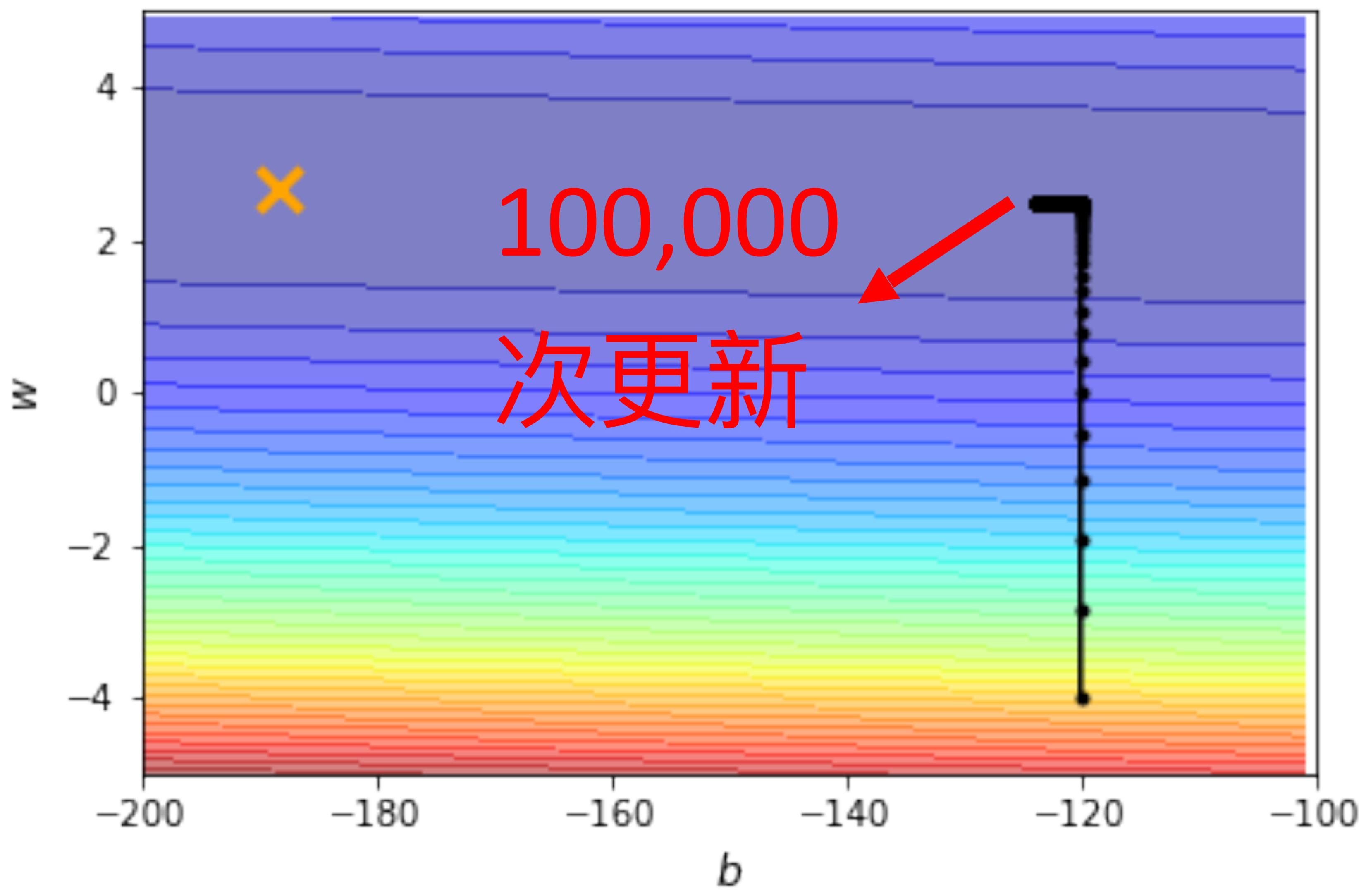
损失函数为凸函数



学习率不能是一成不变的

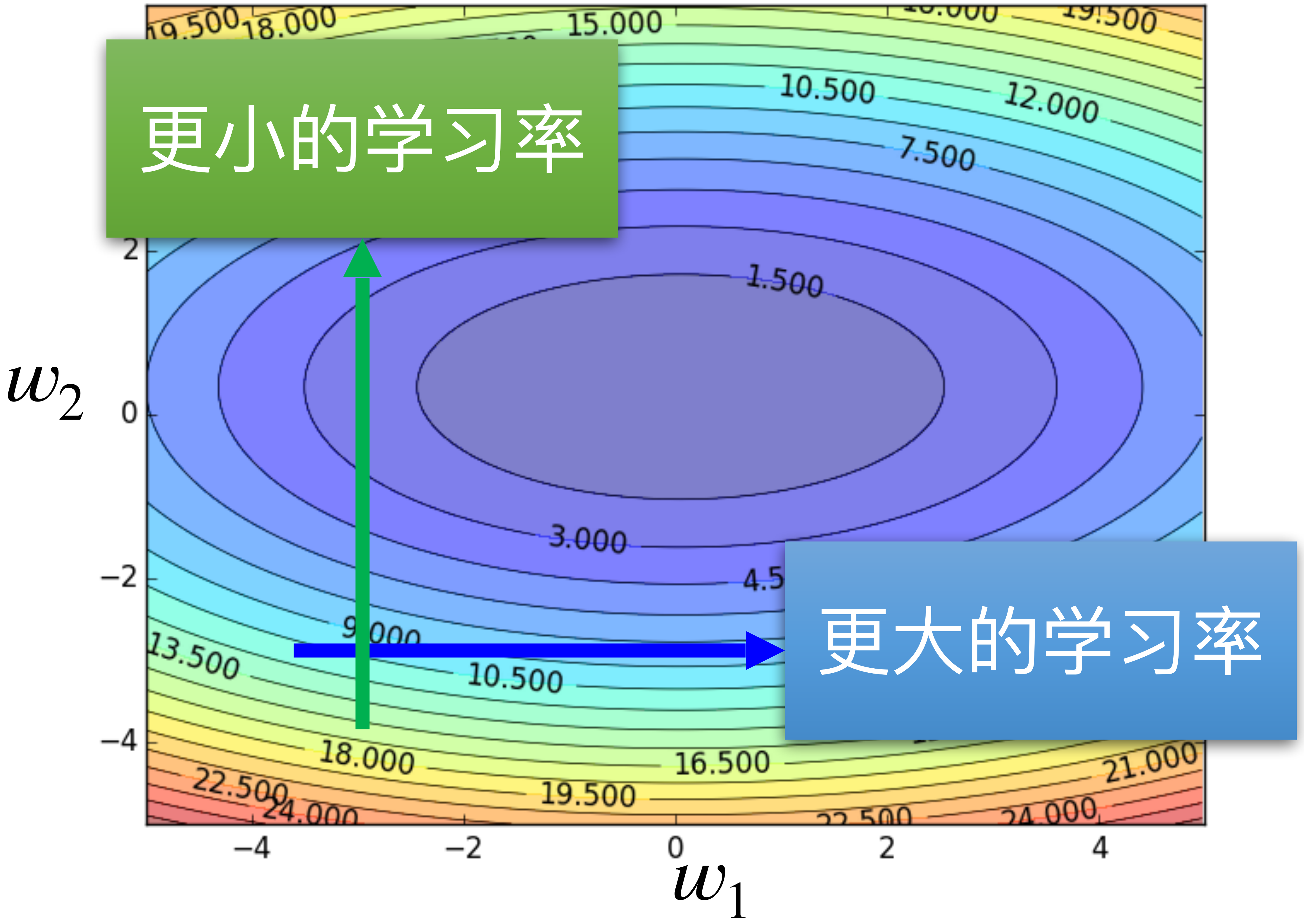


$\eta = 10^{-2}$



$\eta = 10^{-7}$

不同的参数需要不同的学习率



对于某一个参数的梯度更新：

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\eta} \mathbf{g}_i^t$$

$$\mathbf{g}_i^t = \frac{\partial L}{\partial \theta_i} |_{\theta=\theta^t}$$

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} \mathbf{g}_i^t$$

依赖于参数

RMS (Root Mean Square)

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} \mathbf{g}_i^t$$

$$\theta_i^{\mathbf{1}} \leftarrow \theta_i^{\mathbf{0}} - \frac{\eta}{\sigma_i^0} \mathbf{g}_i^{\mathbf{0}} \qquad \sigma_i^0 = \sqrt{\left(\mathbf{g}_i^{\mathbf{0}}\right)^2} = \left|\mathbf{g}_i^{\mathbf{0}}\right|$$

$$\theta_i^{\mathbf{2}} \leftarrow \theta_i^{\mathbf{1}} - \frac{\eta}{\sigma_i^1} \mathbf{g}_i^{\mathbf{1}} \qquad \sigma_i^1 = \sqrt{\frac{1}{2}\left[\left(\mathbf{g}_i^{\mathbf{0}}\right)^2 + \left(\mathbf{g}_i^{\mathbf{1}}\right)^2\right]}$$

$$\theta_i^{\mathbf{3}} \leftarrow \theta_i^{\mathbf{2}} - \frac{\eta}{\sigma_i^2} \mathbf{g}_i^{\mathbf{2}} \qquad \sigma_i^2 = \sqrt{\frac{1}{3}\left[\left(\mathbf{g}_i^{\mathbf{0}}\right)^2 + \left(\mathbf{g}_i^{\mathbf{1}}\right)^2 + \left(\mathbf{g}_i^{\mathbf{2}}\right)^2\right]}$$

$$\vdots$$

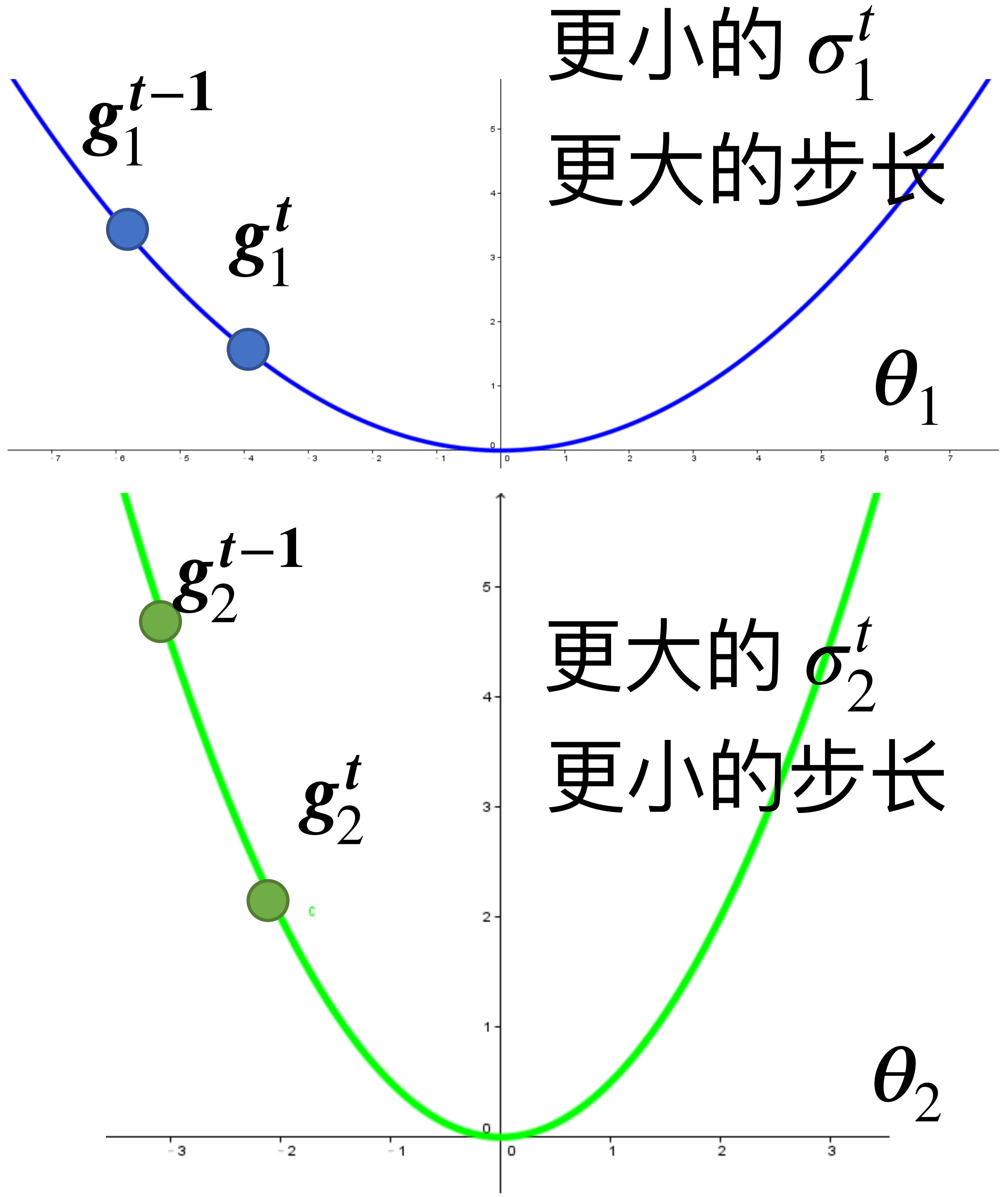
$$\theta_i^{t+1} \leftarrow \theta_i^t - \frac{\eta}{\sigma_i^t} \mathbf{g}_i^t \qquad \sigma_i^t = \sqrt{\frac{1}{t+1} \sum_{i=0}^{t} \left(\mathbf{g}_i^t\right)^2}$$

RMS (Root Mean Square)

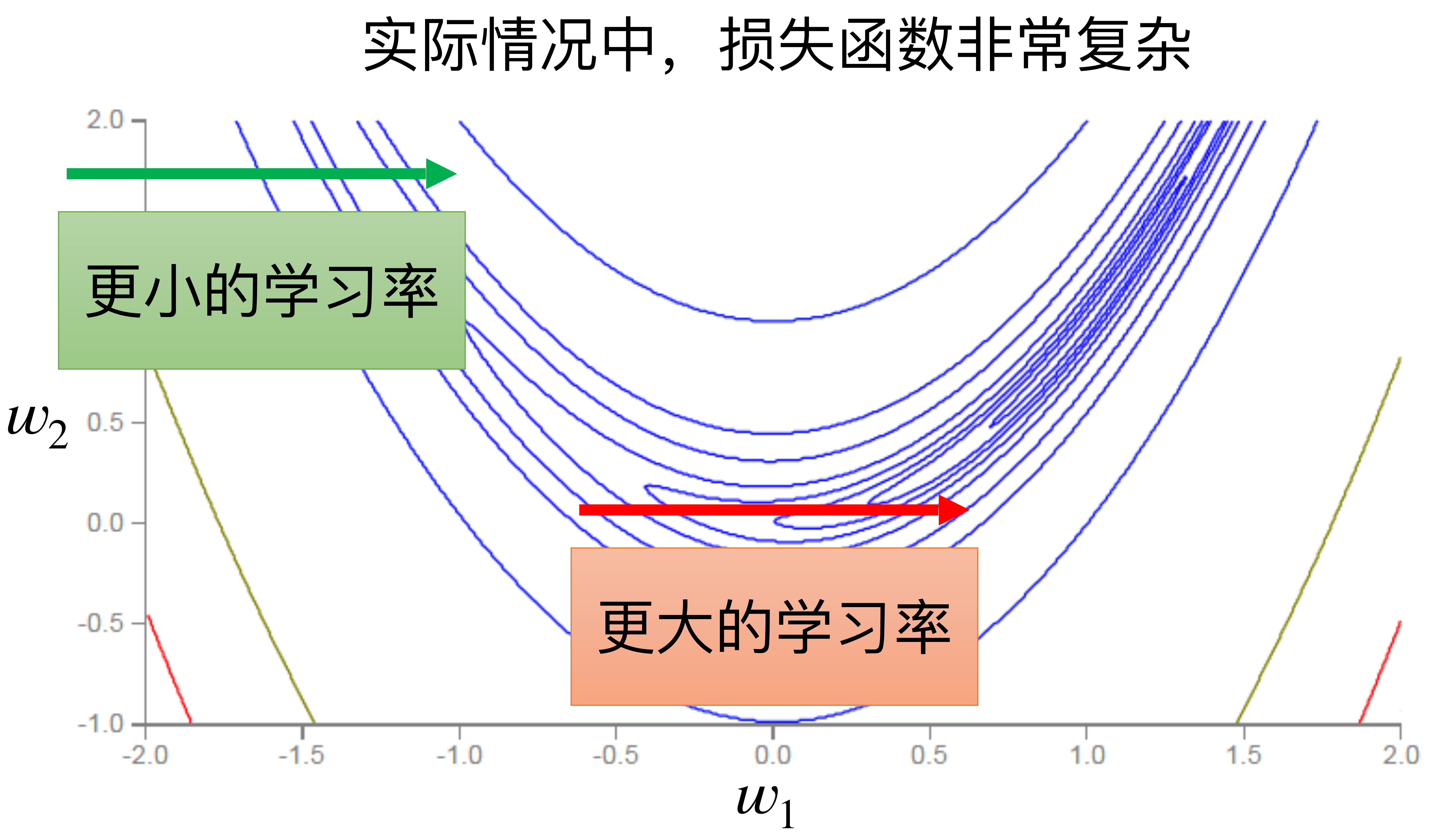
$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} \boldsymbol{g}_i^t$$

$$\sigma_i^t = \sqrt{\frac{1}{t+1}\sum_{i=0}^{t}\left(\boldsymbol{g}_i^t\right)^2}$$

在 AdaGrad 中使用



学习率根据参数动态地变化



RMSProp

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} \boldsymbol{g}_i^t$$

$$\theta_i^{1} \leftarrow \theta_i^0 - \frac{\eta}{\sigma_i^0} \boldsymbol{g}_i^{\boldsymbol{0}} \qquad \sigma_i^0 = \sqrt{\left(\boldsymbol{g}_i^{\boldsymbol{0}}\right)^2}$$

$$0 < \alpha < 1$$

$$\theta_i^{2} \leftarrow \theta_i^1 - \frac{\eta}{\sigma_i^1} \boldsymbol{g}_i^{\boldsymbol{1}} \qquad \sigma_i^1 = \sqrt{\alpha\left(\sigma_i^0\right)^2 + (1-\alpha)\left(\boldsymbol{g}_i^{\boldsymbol{1}}\right)^2}$$

$$\theta_i^{3} \leftarrow \theta_i^2 - \frac{\eta}{\sigma_i^2} \boldsymbol{g}_i^{\boldsymbol{2}} \qquad \sigma_i^2 = \sqrt{\alpha\left(\sigma_i^1\right)^2 + (1-\alpha)\left(\boldsymbol{g}_i^{\boldsymbol{2}}\right)^2}$$

$$\vdots$$

$$\theta_i^{t+1} \leftarrow \theta_i^t - \frac{\eta}{\sigma_i^t} \boldsymbol{g}_i^t \qquad \sigma_i^t = \sqrt{\alpha\left(\sigma_i^{t-1}\right)^2 + (1-\alpha)\left(\boldsymbol{g}_i^t\right)^2}$$

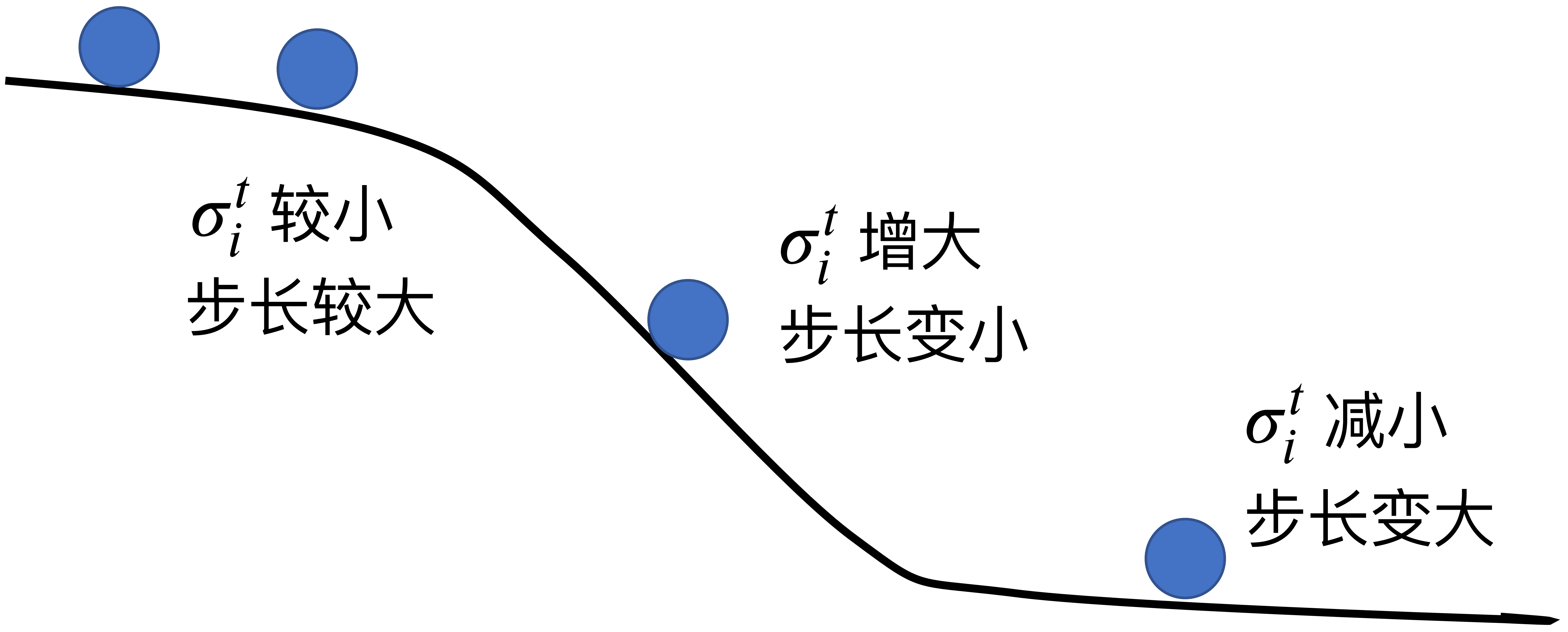
RMSProp

$$\mathbf{g}_i^1 \quad \mathbf{g}_i^2 \quad \cdots\cdots \quad \mathbf{g}_i^{t-1}$$

$$0 < \alpha < 1$$

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} \mathbf{g}_i^t \qquad \sigma_i^t = \sqrt{\alpha\left(\sigma_i^{t-1}\right)^2 + (1-\alpha)\left(\mathbf{g}_i^t\right)^2}$$

最近的梯度影响较大，过往的梯度影响较小



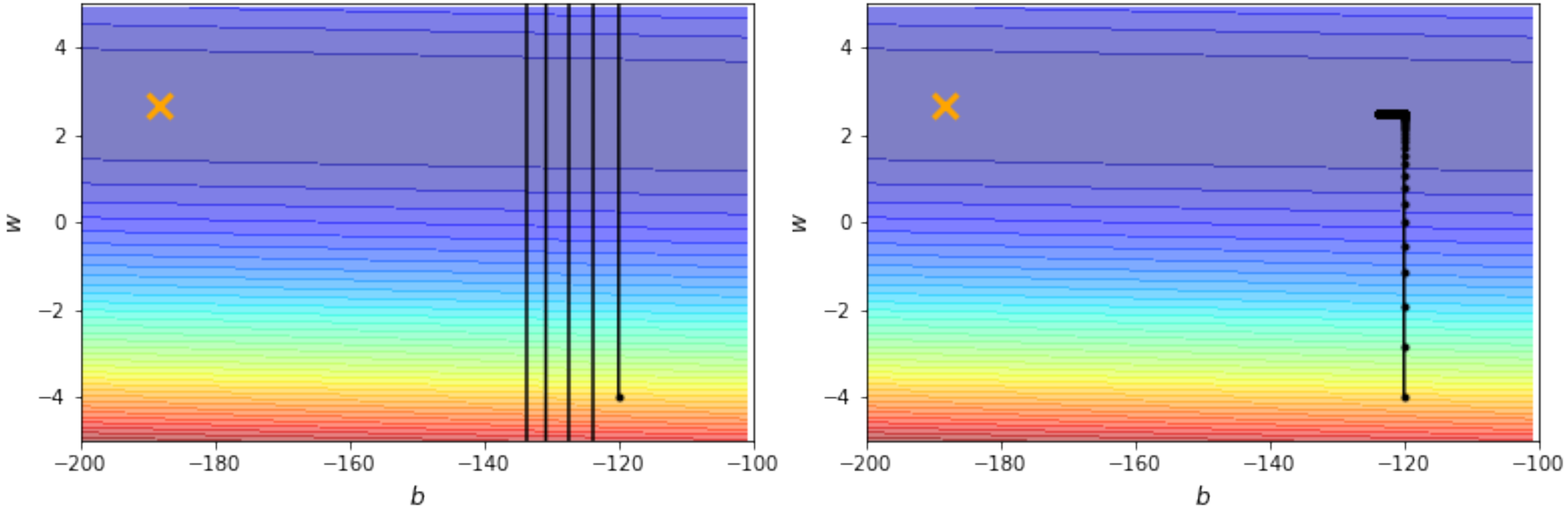
Adam：RMSProp + 动量

Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization. ICLR 2015.

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector) 动量
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector) RMSProp
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

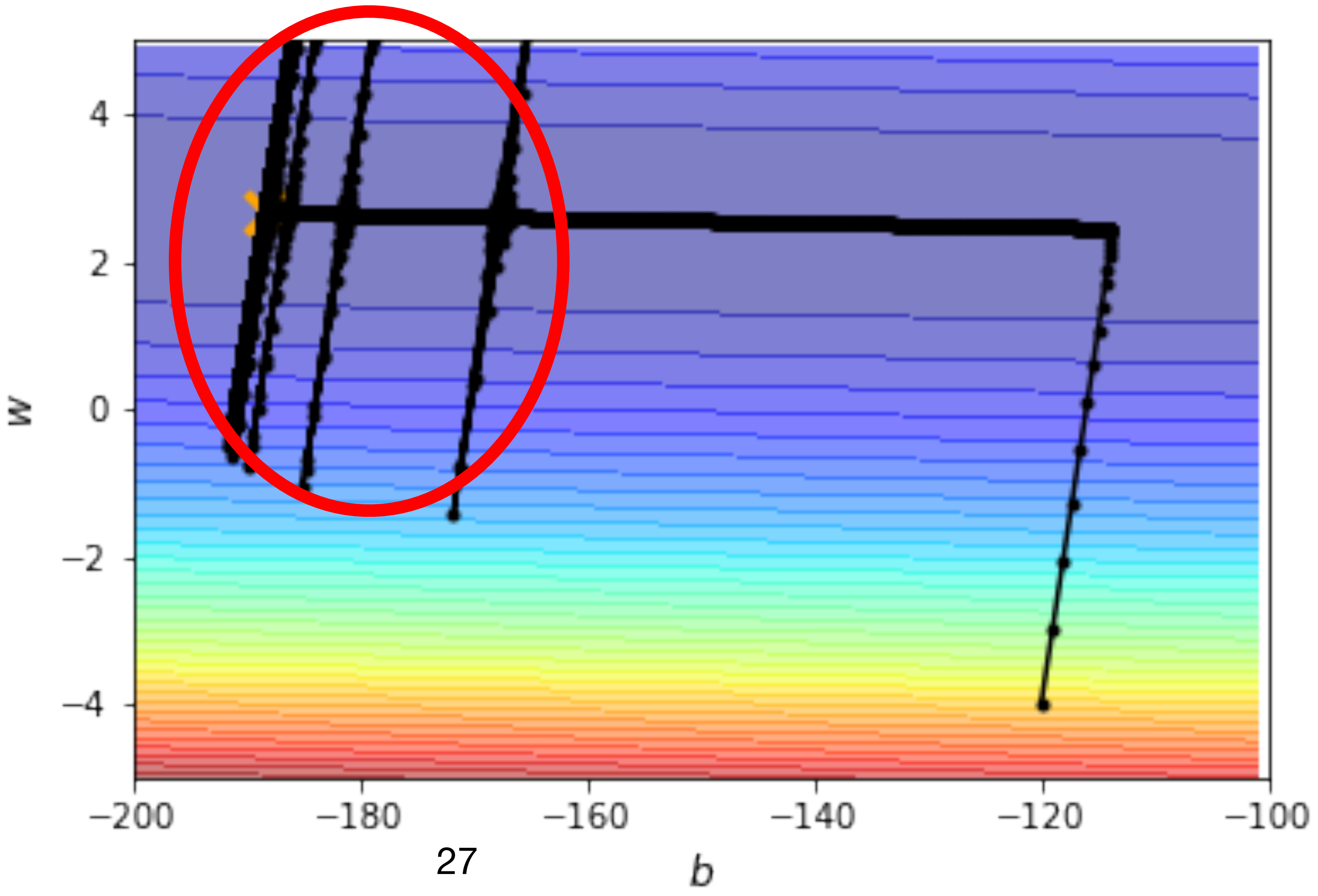
适应性学习率（Adaptive Learning Rate）



无适应性学习率

$$\theta_i^{t+1} \leftarrow \theta_i^t - \boxed{\frac{\eta}{\sigma_i^t}} g_i^t$$

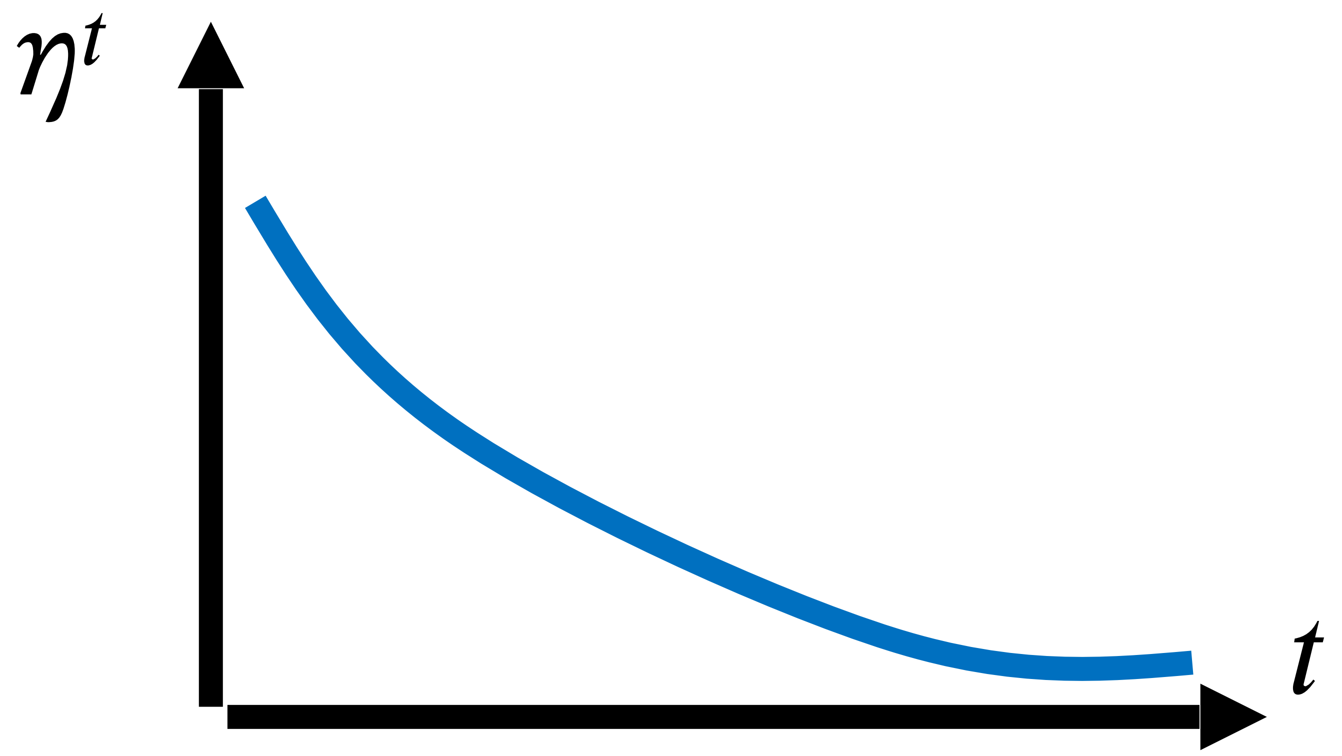
$$\sigma_i^t = \sqrt{\frac{1}{t+1}\sum_{i=0}^{t}\left(g_i^t\right)^2}$$



有适应性学习率

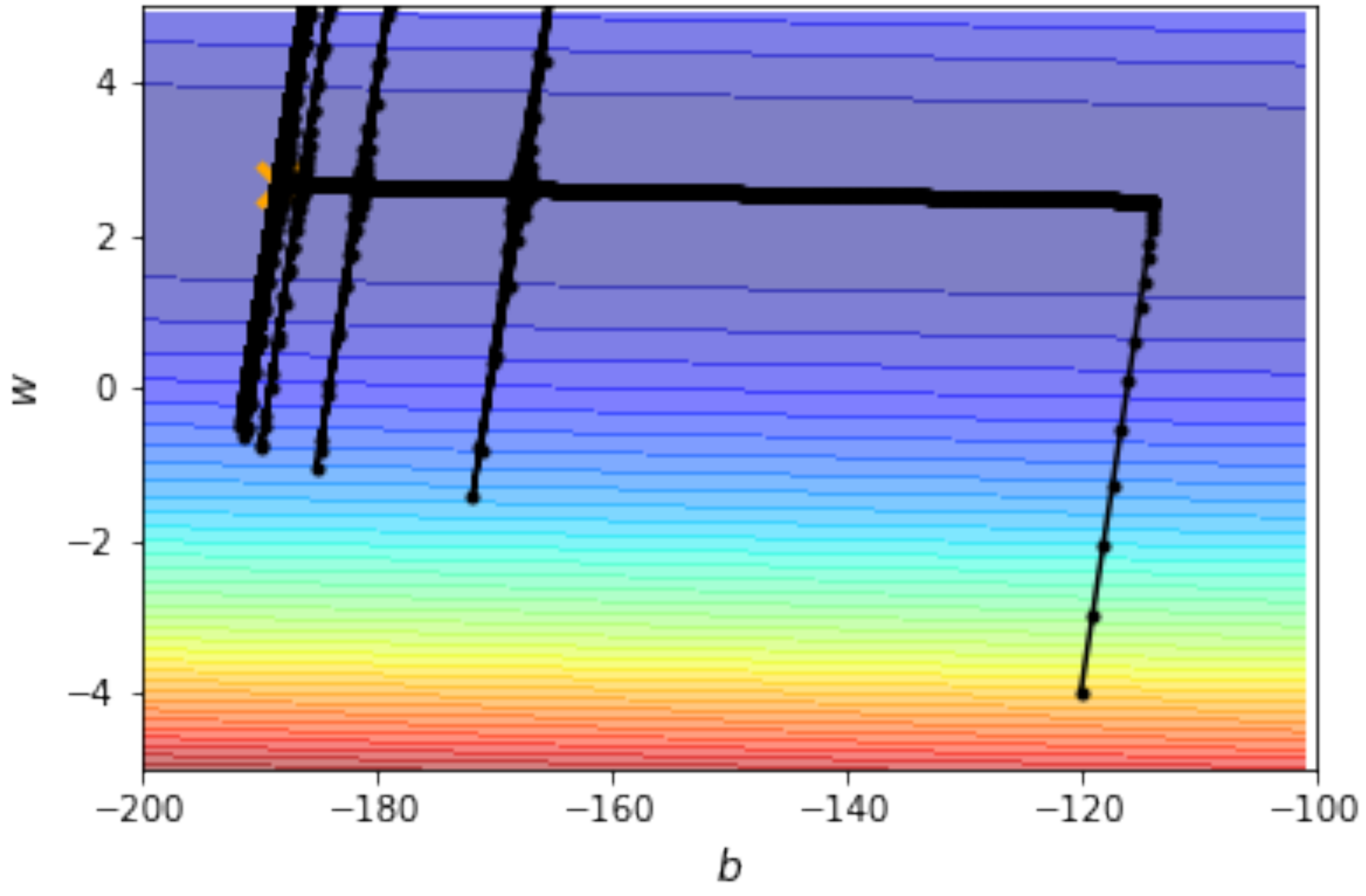
学习率规划（Scheduling）

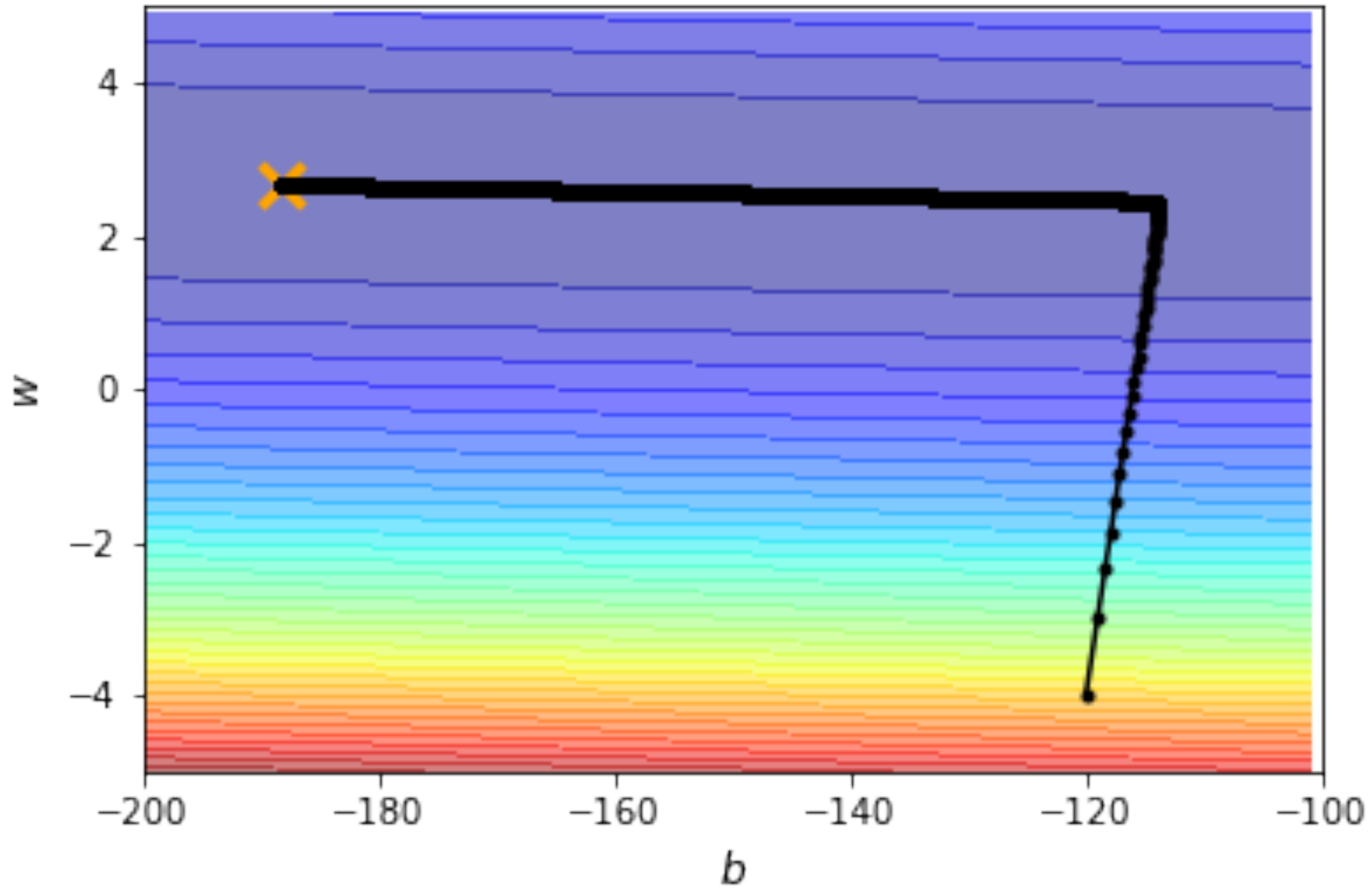
$$\theta_i^{t+1} \leftarrow \theta_i^t - \frac{\boxed{\eta^t}}{\sigma_i^t} g_i^t$$



学习率衰减（Learning Rate Decay）

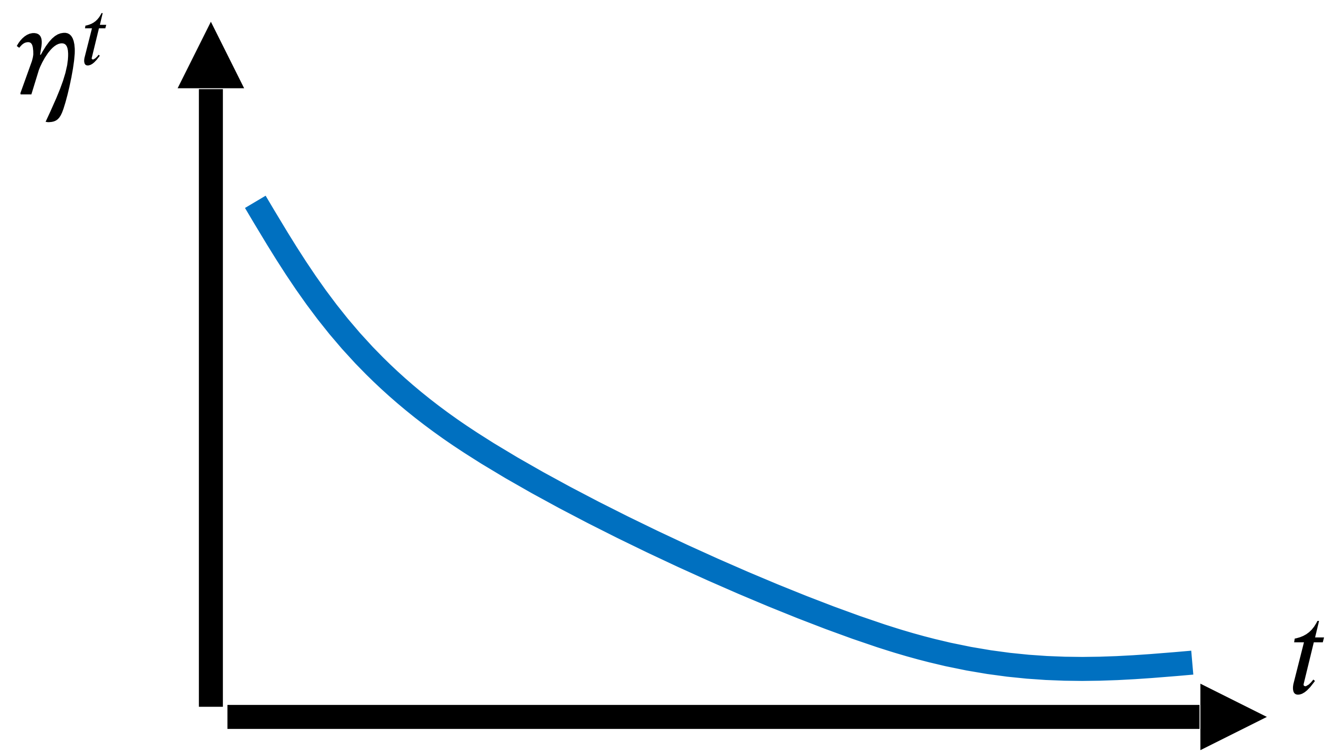
随着训练过程的推进，我们离目标越来越近，因此需要逐步降低学习率





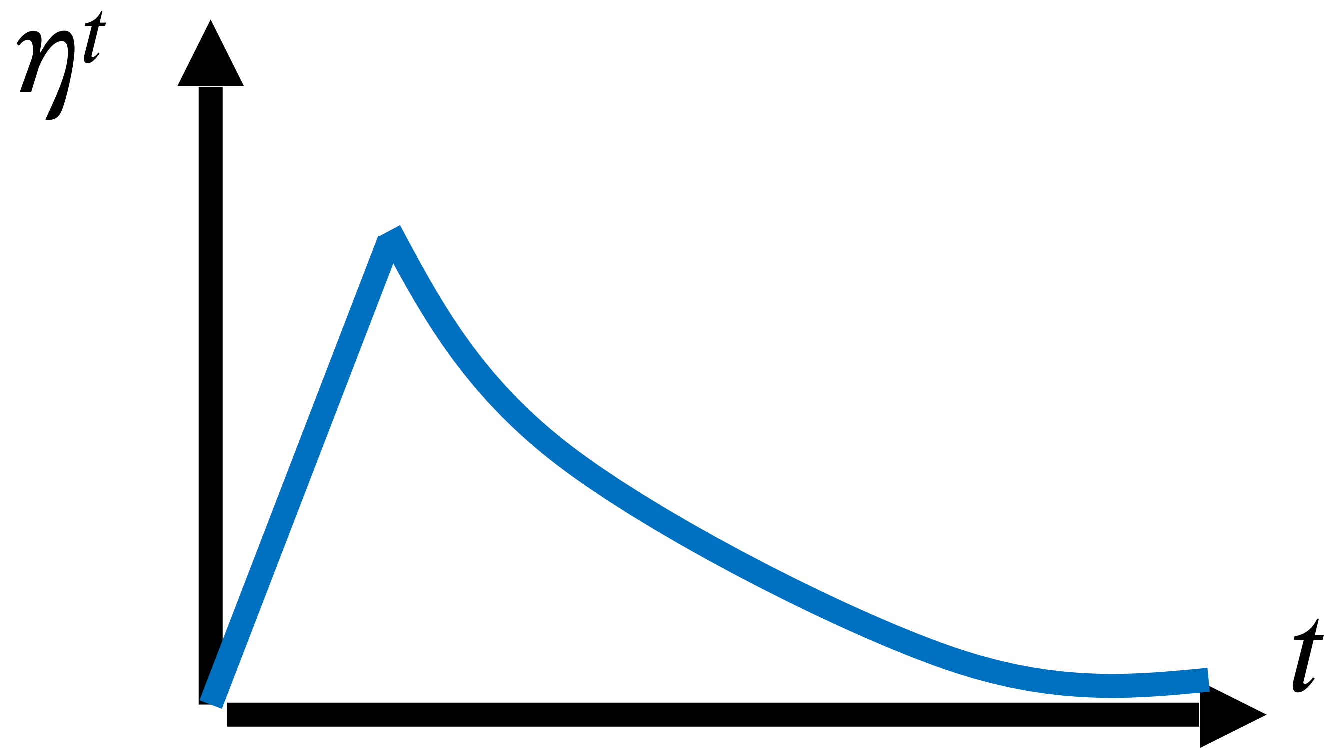
学习率规划（Scheduling）

$$\theta_i^{t+1} \leftarrow \theta_i^t - \frac{\boxed{\eta^t}}{\sigma_i^t} g_i^t$$



学习率衰减（Learning Rate Decay）

随着训练过程的推进，我们离目标越来越近，因此需要逐步降低学习率



学习率预热（Warm Up）

先增大学习率，再减小学习率

在初始阶段，σ_i^t 的估计（Estimation）有着较大的方差

AdamW

Ilya Loshchilov, Frank Hutter. Decoupled Weight Decay Regularization. ICLR 2019.

- AdamW: Adam + Weight Decay（权重衰减）

模型泛化能力和收敛性能更好

Algorithm 2 Adam with L_2 regularization and Adam with decoupled weight decay (AdamW)

1: **given** $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$
2: **initialize** time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
3: **repeat**
4: $t \leftarrow t + 1$
5: $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
6: $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) \; +\lambda\boldsymbol{\theta}_{t-1}$
7: $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t$ ▷ here and below all operations are element-wise
8: $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t^2$
9: $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t/(1-\beta_1^t)$ ▷ β_1 is taken to the power of t
10: $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t/(1-\beta_2^t)$ ▷ β_2 is taken to the power of t
11: $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ can be fixed, decay, or also be used for warm restarts
12: $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left(\alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) \; +\lambda\boldsymbol{\theta}_{t-1} \right)$
13: **until** *stopping criterion is met*
14: **return** optimized parameters $\boldsymbol{\theta}_t$