

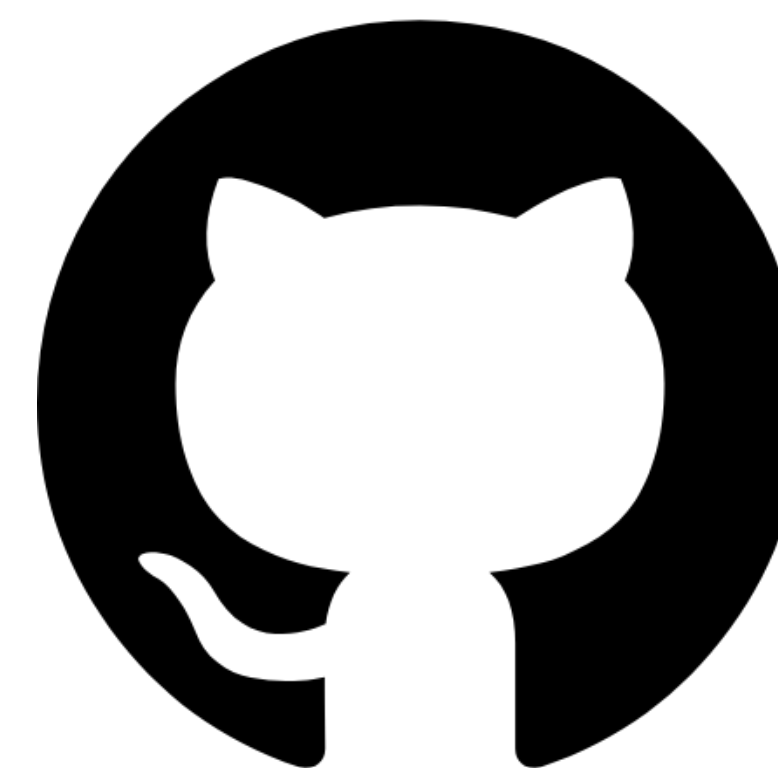
个人简介

唐亘，数据科学家



- 教育背景
 - 复旦大学数学和计算机双学士
 - 巴黎综合理工经济学硕士、ENSAE数据科学硕士
- 项目经验
 - 出版《解构大语言模型：从线性回归到通用人工智能》、《精通数据科学》

从**模型结构**和**数据基础**角度
拆解、重构大语言模型



<https://github.com/GenTang/regression2chatgpt>

视频列表

工程基础

- 最优化算法：梯度下降法及其衍生
- 反向传播算法

模型基础

- 多层感知器 (multilayer perceptron, MLP)
- 卷积神经网络 (CNN) 与残差链接 (Residual Connection)
- 循环神经网络 (RNN) 与自然语言处理
- 从零实现GPT-2

大语言模型

- 大语言模型监督微调 (Supervised Fine-tuning)
- 利用强化学习进一步微调大语言模型 (Reinforcement Learning)

梯度下降法理论基础

以线性回归为例，模型的损失函数：
$$L(a, b) = \frac{1}{n} \sum_{i=1}^n (y_i - ax_i - b)^2$$

梯度下降法理论基础

以线性回归为例，模型的损失函数： $L(a, b) = 1/n \sum_{i=1}^n (y_i - ax_i - b)^2$

选取一个随机的起始点 a_0, b_0 ；当
 a_1, b_1 在其**附近**时

$$\Delta L = L(a_1, b_1) - L(a_0, b_0) \approx \frac{\partial L}{\partial a} \Delta a + \frac{\partial L}{\partial b} \Delta b \longrightarrow \begin{array}{l} \Delta a = a_1 - a_0 \\ \Delta b = b_1 - b_0 \end{array}$$

根据泰勒展开式

梯度下降法理论基础

以线性回归为例，模型的损失函数： $L(a, b) = 1/n \sum_{i=1}^n (y_i - ax_i - b)^2$

选取一个随机的起始点 a_0, b_0 ；当 a_1, b_1 在其附近时

$$\Delta L = L(a_1, b_1) - L(a_0, b_0) \approx \frac{\partial L}{\partial a} \Delta a + \frac{\partial L}{\partial b} \Delta b \longrightarrow \begin{array}{l} \Delta a = a_1 - a_0 \\ \Delta b = b_1 - b_0 \end{array}$$

根据泰勒展开式

不难推导出：

$$(\Delta a, \Delta b) = -\eta \left(\frac{\partial L}{\partial a}, \frac{\partial L}{\partial b} \right) \xrightarrow{\eta > 0} \Delta L \approx -\eta \left[\left(\frac{\partial L}{\partial a} \right)^2 + \left(\frac{\partial L}{\partial b} \right)^2 \right] \leq 0$$

梯度： ∇L

梯度下降法理论基础

以线性回归为例，模型的损失函数： $L(a, b) = 1/n \sum_{i=1}^n (y_i - ax_i - b)^2$

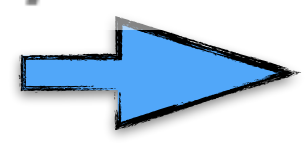
选取一个随机的起始点 a_0, b_0 ；当 a_1, b_1 在其附近时

$$\Delta L = L(a_1, b_1) - L(a_0, b_0) \approx \frac{\partial L}{\partial a} \Delta a + \frac{\partial L}{\partial b} \Delta b \longrightarrow \begin{aligned} \Delta a &= a_1 - a_0 \\ \Delta b &= b_1 - b_0 \end{aligned}$$

根据泰勒展开式

不难推导出：

$$(\Delta a, \Delta b) = -\eta \left(\frac{\partial L}{\partial a}, \frac{\partial L}{\partial b} \right) \xrightarrow{\eta > 0} \Delta L \approx -\eta \left[\left(\frac{\partial L}{\partial a} \right)^2 + \left(\frac{\partial L}{\partial b} \right)^2 \right] \leq 0$$

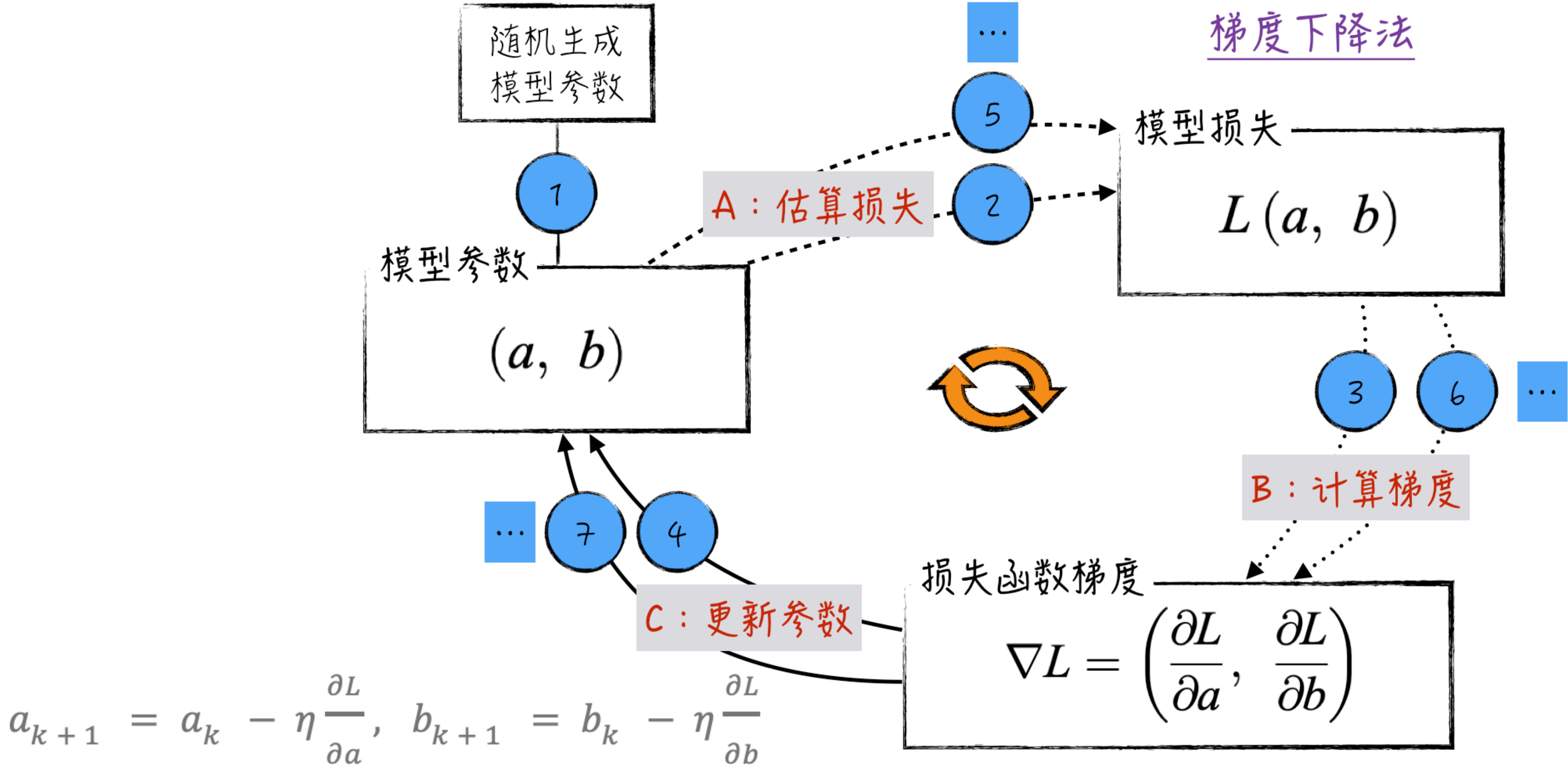


梯度： ∇L

迭代算法：

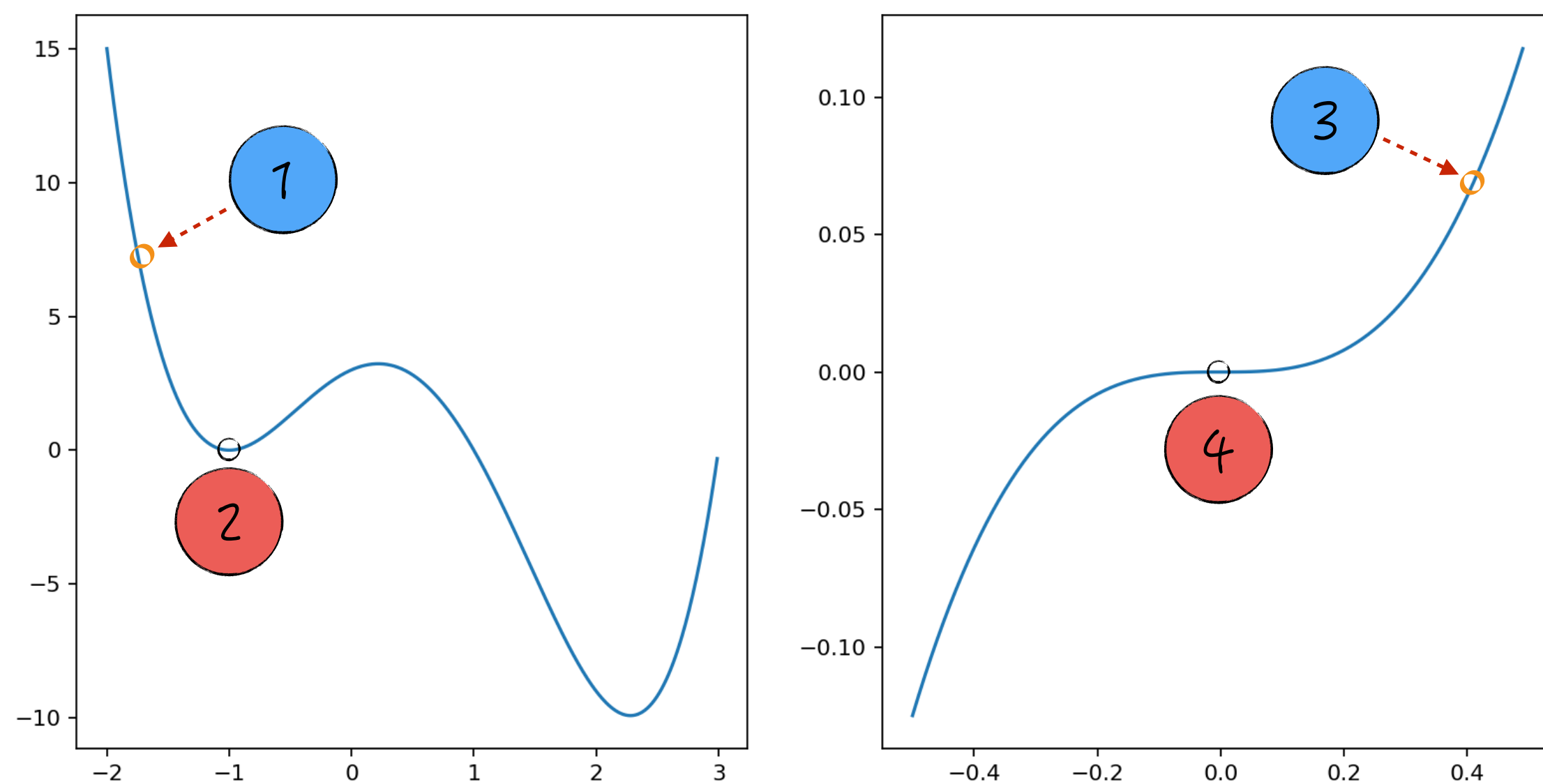
$$a_{k+1} = a_k - \eta \frac{\partial L}{\partial a}, \quad b_{k+1} = b_k - \eta \frac{\partial L}{\partial b}$$

梯度下降法理论基础



梯度下降法的局限

1、无法保证最低点



在梯度计算时，引入随机性

2、计算量太大，需要遍历所有数据点

$$\nabla L = 1/n \sum_i \nabla L_i$$

减少计算时使用的数据量

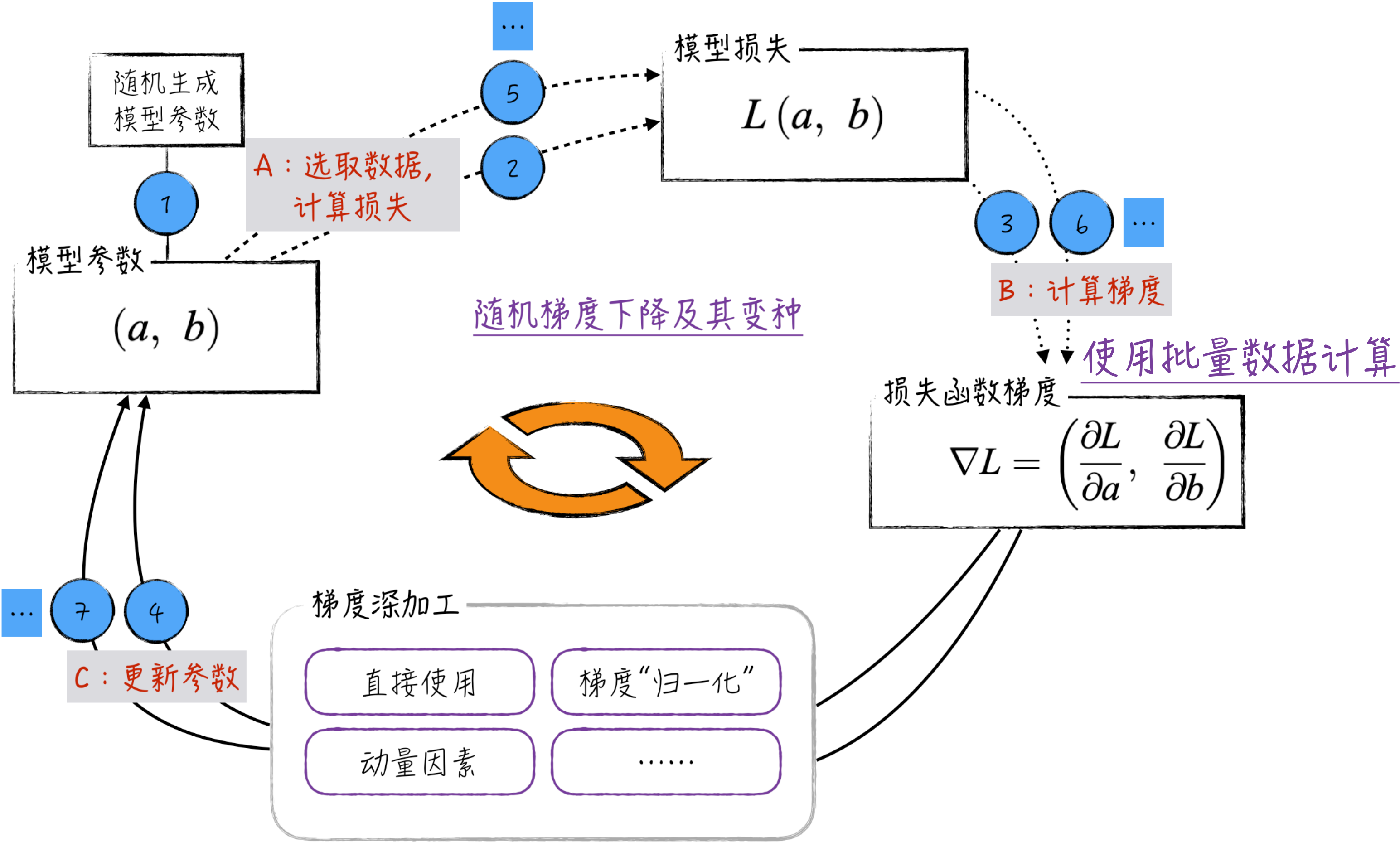
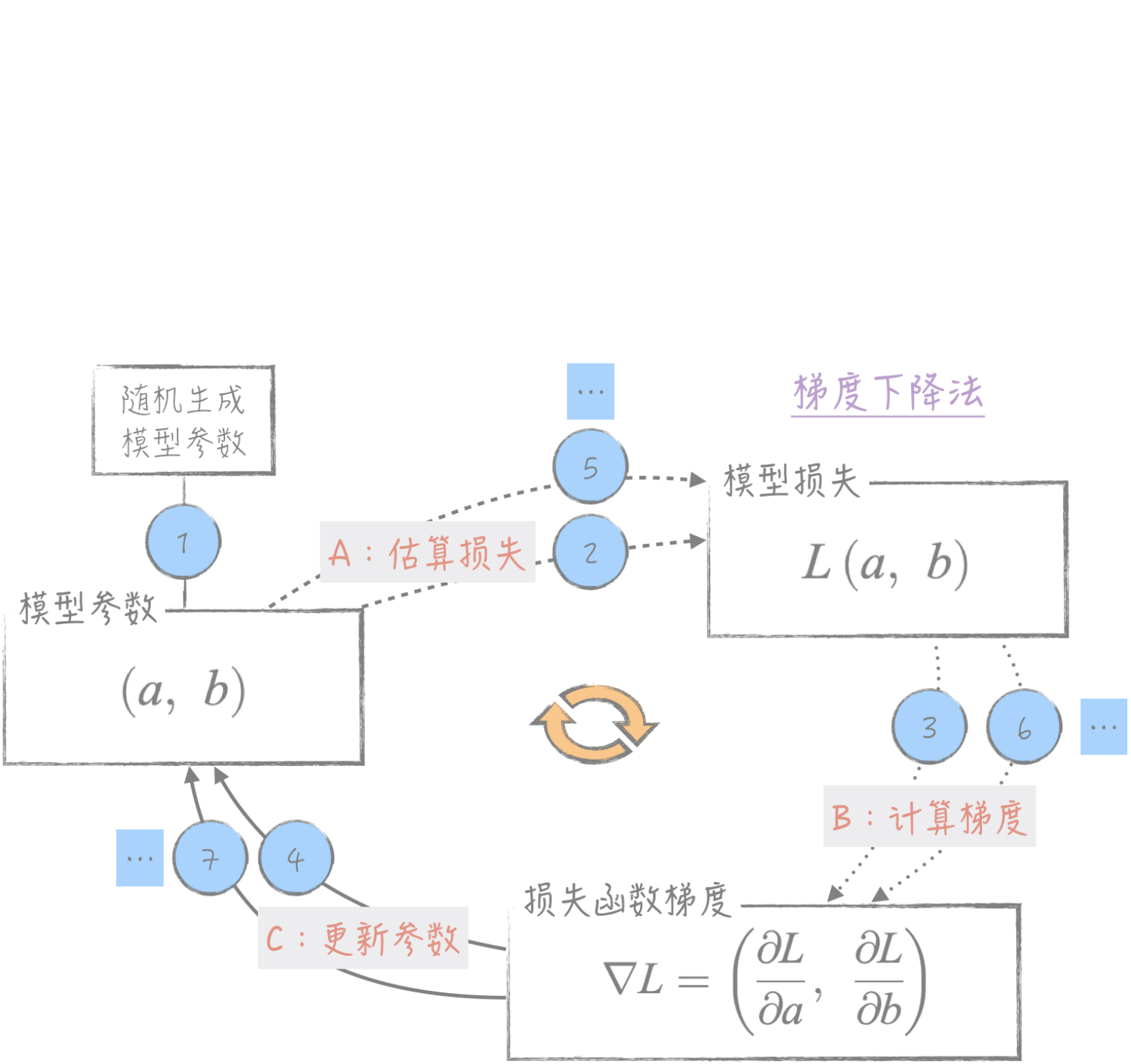
随机梯度下降法

将数据分为m个批次（m份）： I_1, I_2, \dots, I_m

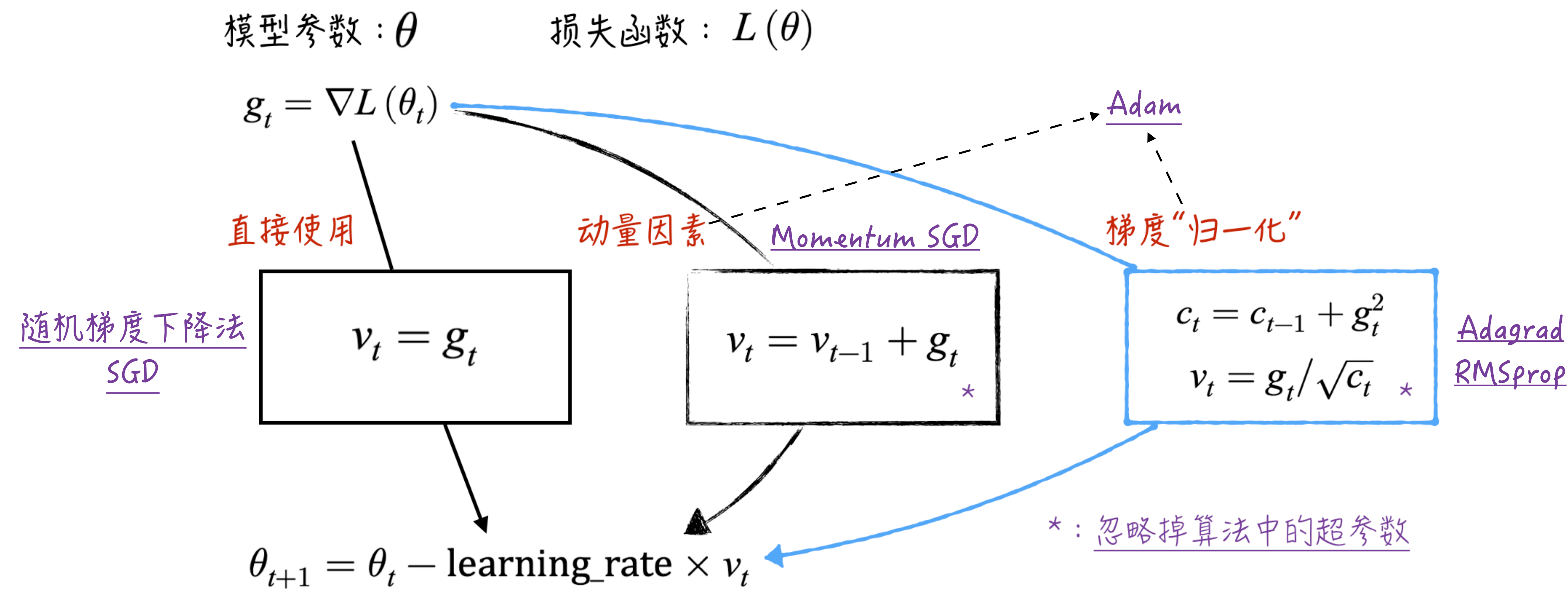
每次只使用一个批次的数据： $\nabla L = 1/n \sum_i \nabla L_i \approx 1/m \sum_{j=1}^m \nabla L_{I_j}$

迭代算法： $a_{k+1} = a_k - \frac{\eta}{m} \sum_{j=1}^m \frac{\partial L_{I_j}}{\partial a}, b_{k+1} = b_k - \frac{\eta}{m} \sum_{j=1}^m \frac{\partial L_{I_j}}{\partial b}$

更优化的算法

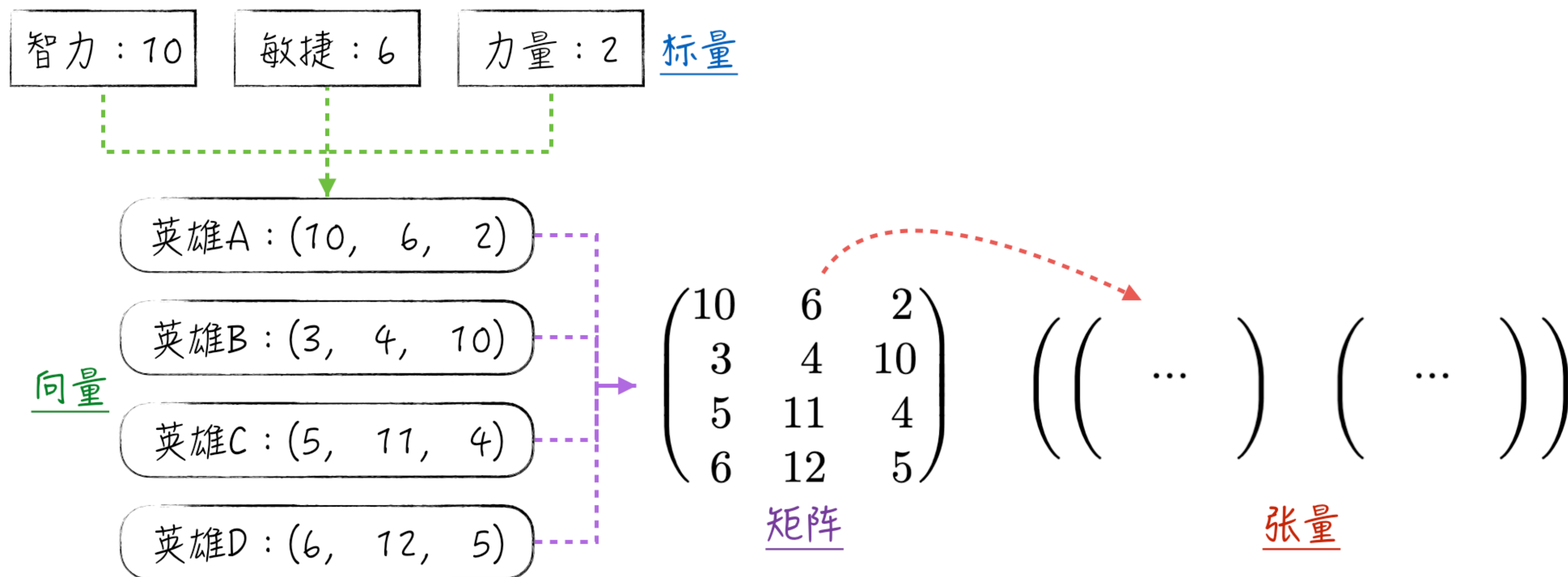


更优化的算法



PyTorch基础

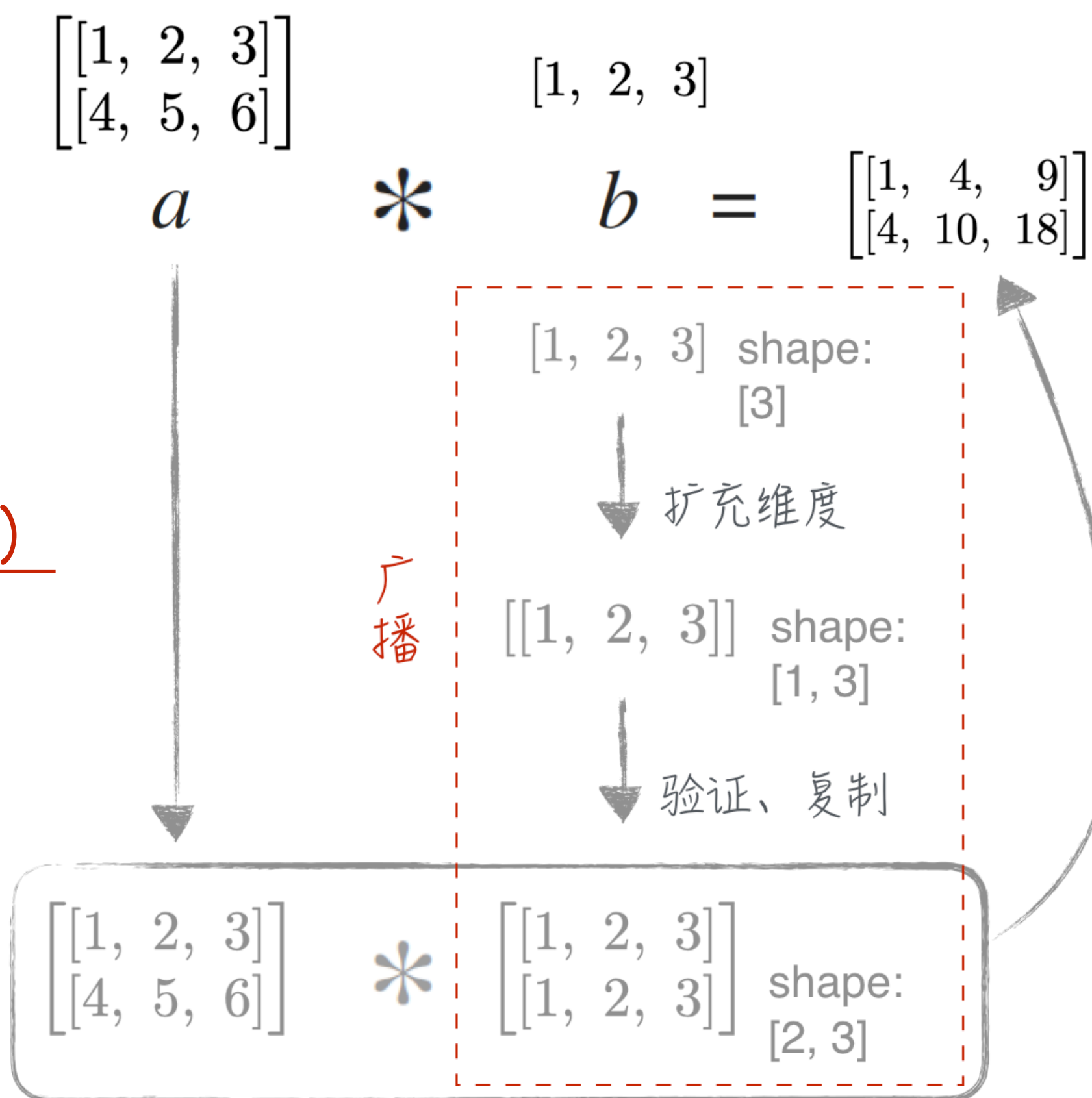
PyTorch的基础数据结构是张量 (Tensor)



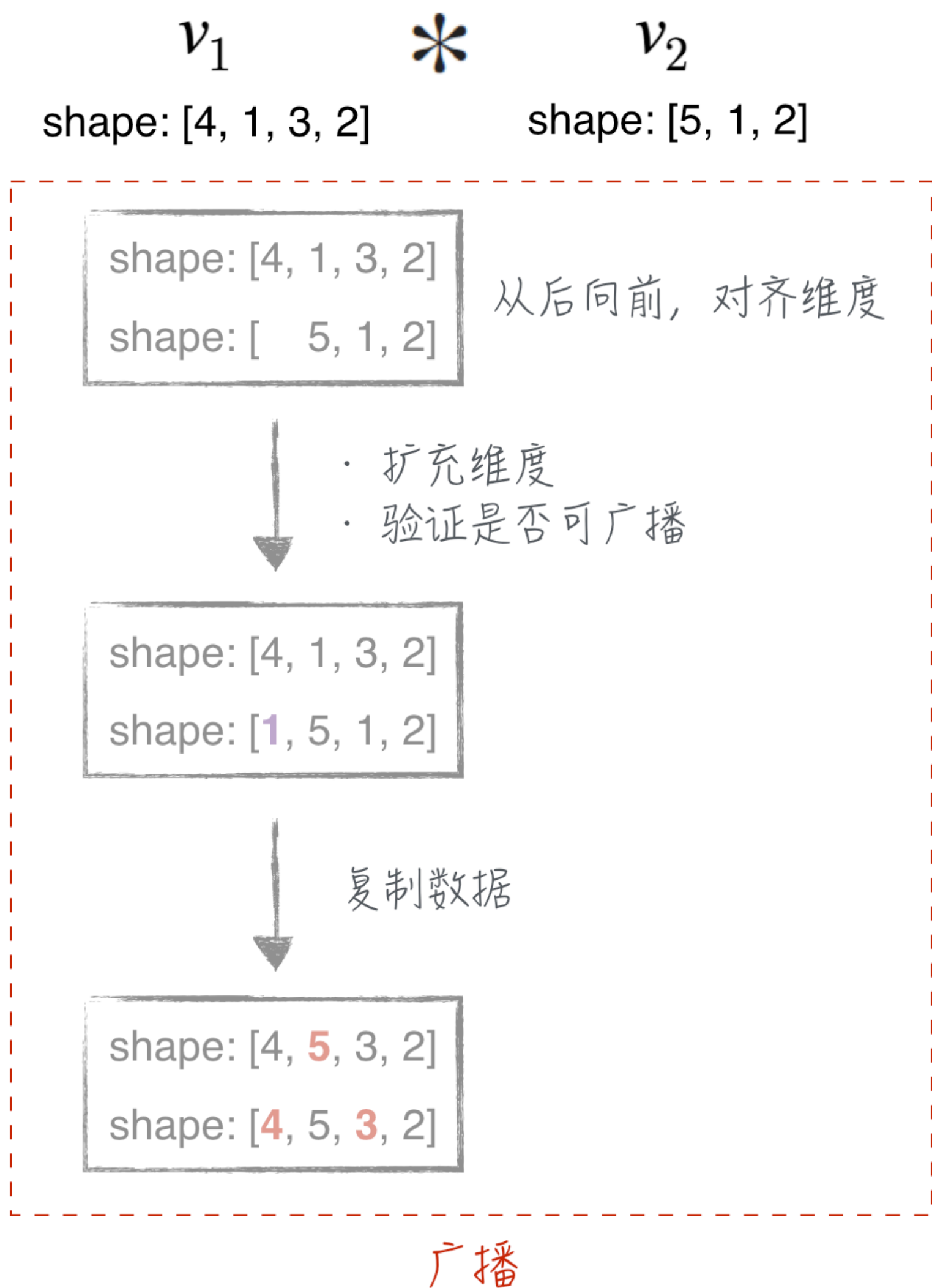
PyTorch基础

简单例子

广播机制 (Broadcasting)



复杂例子



总结

