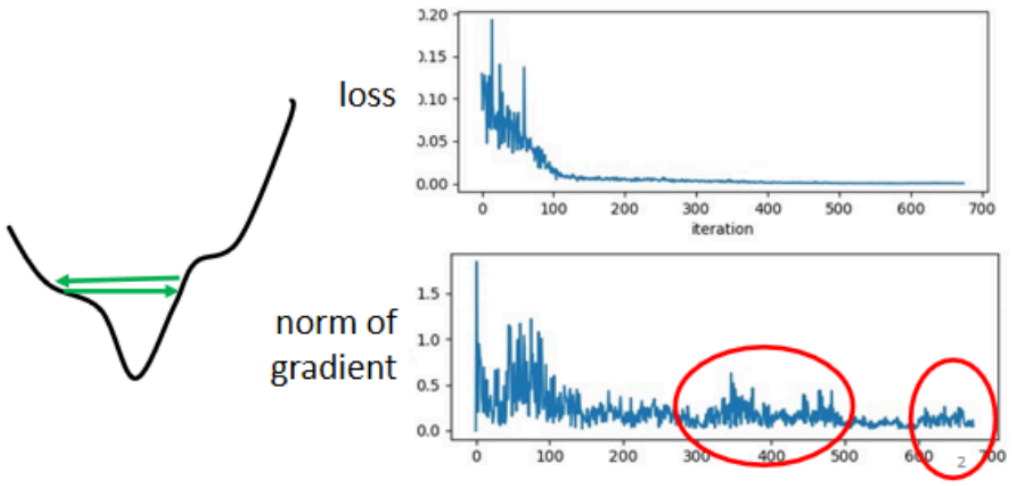
Adaptive Learning Rate

1. 错误认知：训练卡住 ≠ 来到 critical point

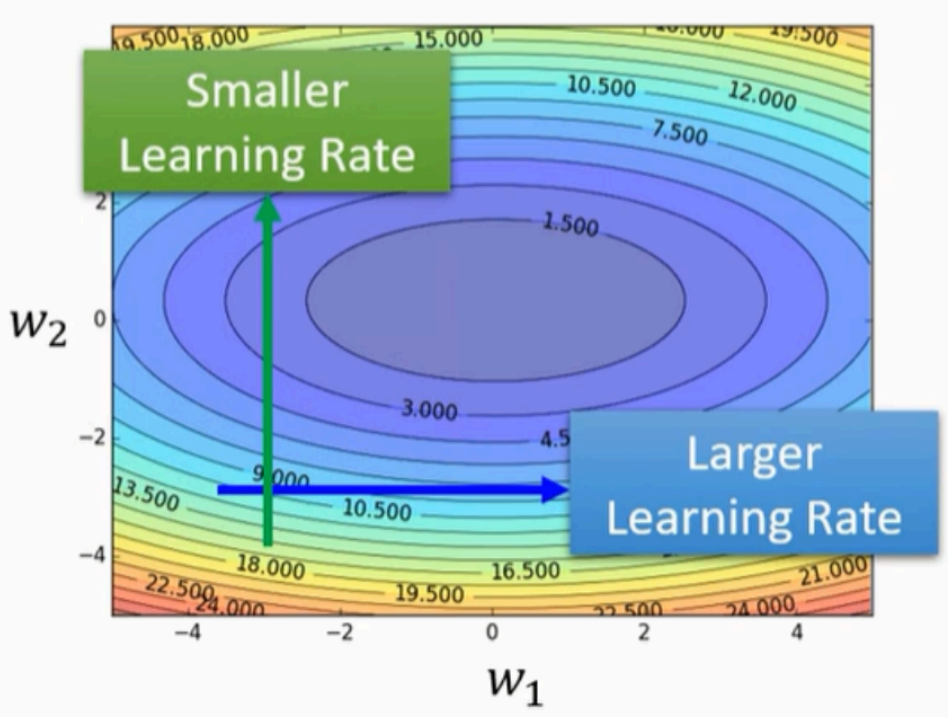
- **训练卡住，不等于梯度很小**



原因：

- 模型可能在 “峡谷两边反复震荡”，梯度方向两侧来回跳，导致 loss 不降
- 训练卡住最常见原因 **不是 critical point，而是学习率的问题**
    - 这就是为什么需要 **Adaptive Learning Rate（自适应学习率）**

2. 关键思想：每个参数需要自己的 learning rate



设参数为 θ_i，其更新式改为：

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

其中：

- g_i^t：该参数方向上的梯度
- σ_i^t：该参数的 “方向尺度”，用来调节步长

机制：

- 若该方向梯度长期 **很小** → σ 小 → learning rate = η/σ 变大
- 若该方向梯度长期 **很大** → σ 大 → learning rate = η/σ 变小

3. Adagrad：第一种 Adaptive Learning Rate

Adagrad 的思想：

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

特点：

- 用 **所有过去的梯度平方平均** 来调整步长
- 越 “陡峭” 的方向 → 梯度大 → σ 大 → 步子小
- 越 “平坦” 的方向 → 梯度小 → σ 小 → 步子大

优点：

- 自动调节学习率
- 不需要人为为每个参数调 learning rate

4. RMSProp：解决 Adagrad 的缺陷

Adagrad 的问题是“所有梯度权重一样大”，RMSProp 引入 **指数加权平均**：

$$\sigma_i^t = \sqrt{\alpha(\sigma_i^{t-1})^2 + (1-\alpha)(g_i^t)^2}$$

α 是超参数，通常接近 1（如 0.9）。
含义：

- 最近的梯度更重要
- 过去的梯度影响逐渐衰减

优势：

- 可以 **快速反应梯度变化**
- 在突然变陡峭的地方，σ 变大 → 自动 “踩刹车”
- 在变平滑时，σ 变小 → 自动 “加速”

5. Adam：Momentum + RMSProp

Adam = RMSProp + Momentum

- Momentum 累积 **方向信息**（有正负）
- RMSProp 累积 **大小信息**（只看幅度）

Adam 是目前使用最广的优化器：

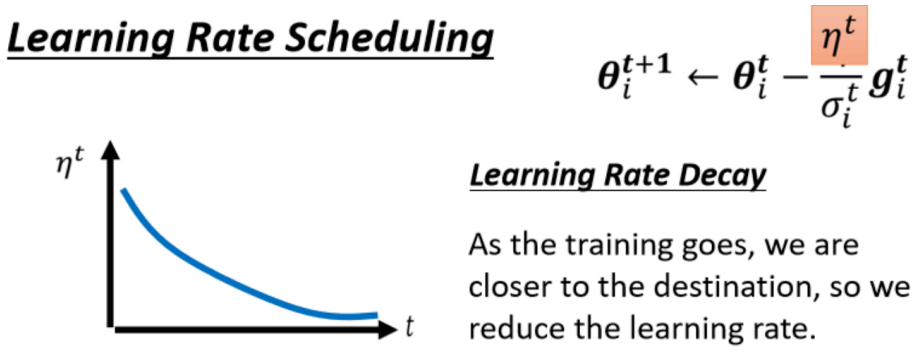
- PyTorch 默认参数一般无需调整
- 收敛快、稳定性高

6. Learning Rate Scheduling

自适应学习率之外，还需学习率随时间变化

> **LR Scheduling 与 Adaptive LR 是分层叠加的机制**

6.1 Learning Rate Decay

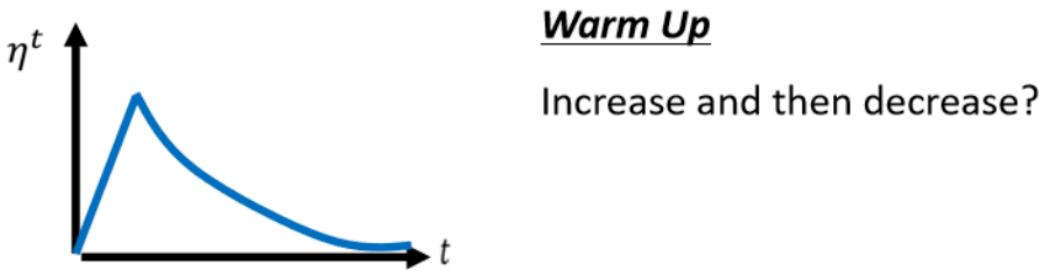


随时间逐渐降低 η。

理由：

- 越接近 minimum → 需要越小的步长
- 可以平滑训练曲线、避免震荡

6.2 Warm Up（黑科技）



Warm Up 的学习率策略：

- **前几步先用非常小的 learning rate**
- 再逐渐升到正常 learning rate
- 之后再 Decay

7. 总结

机制	本质作用
Momentum	累积历史梯度，稳定方向，加速下降
Adaptive LR（σ）	用梯度大小调整步伐（RMS / RMSProp / Adagrad）
LR Scheduling	随时间调整全局学习率（Decay / Warm Up）

> 训练卡住常常不是 critical point，而是“学习率在不同方向不合适”
> 因此现代优化方法通过：
>
> **每个参数一个学习率（Adaptive） + 时间调整（Schedule） + 方向平均（Momentum）**
>
> 来提升训练速度与稳定性