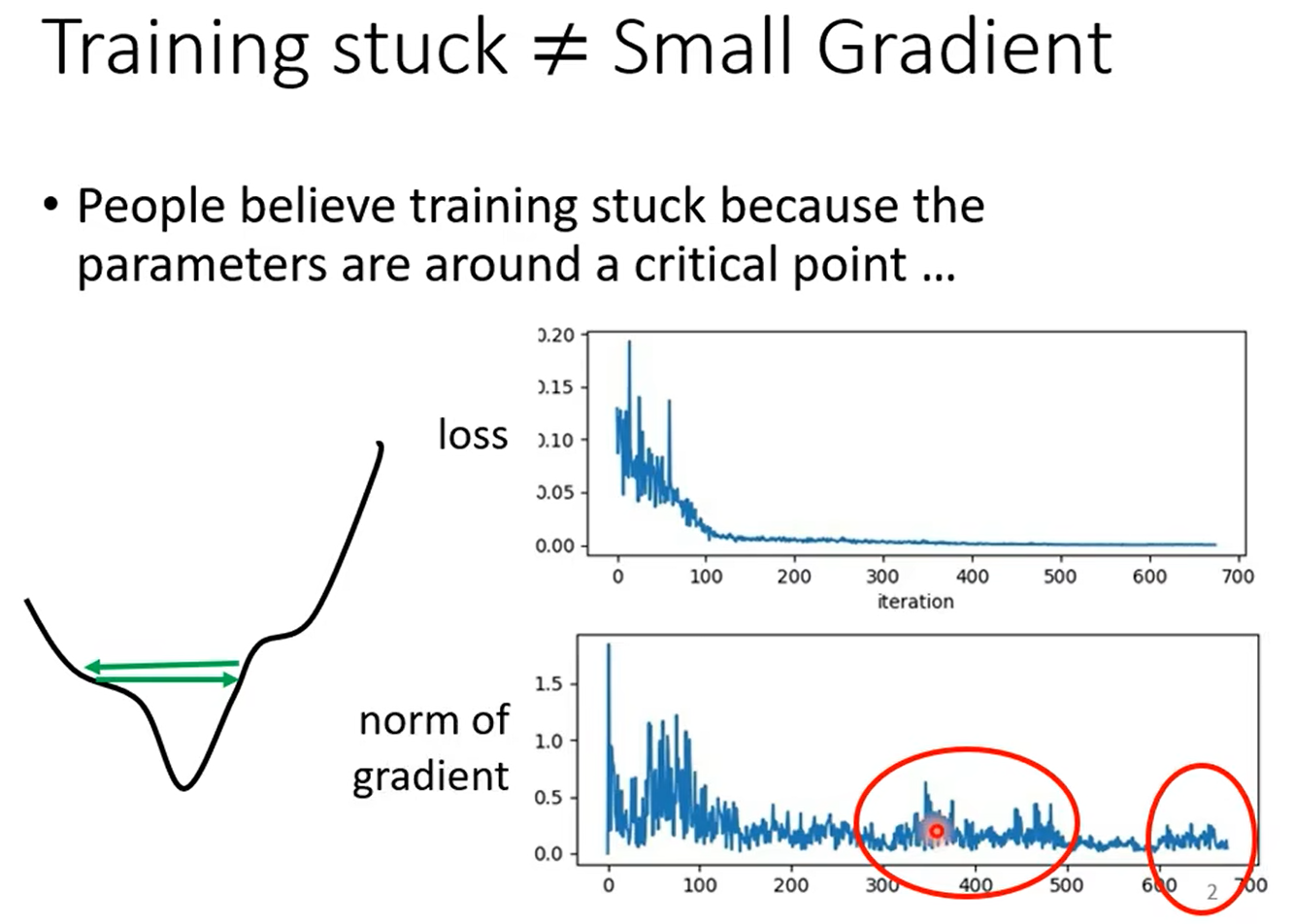
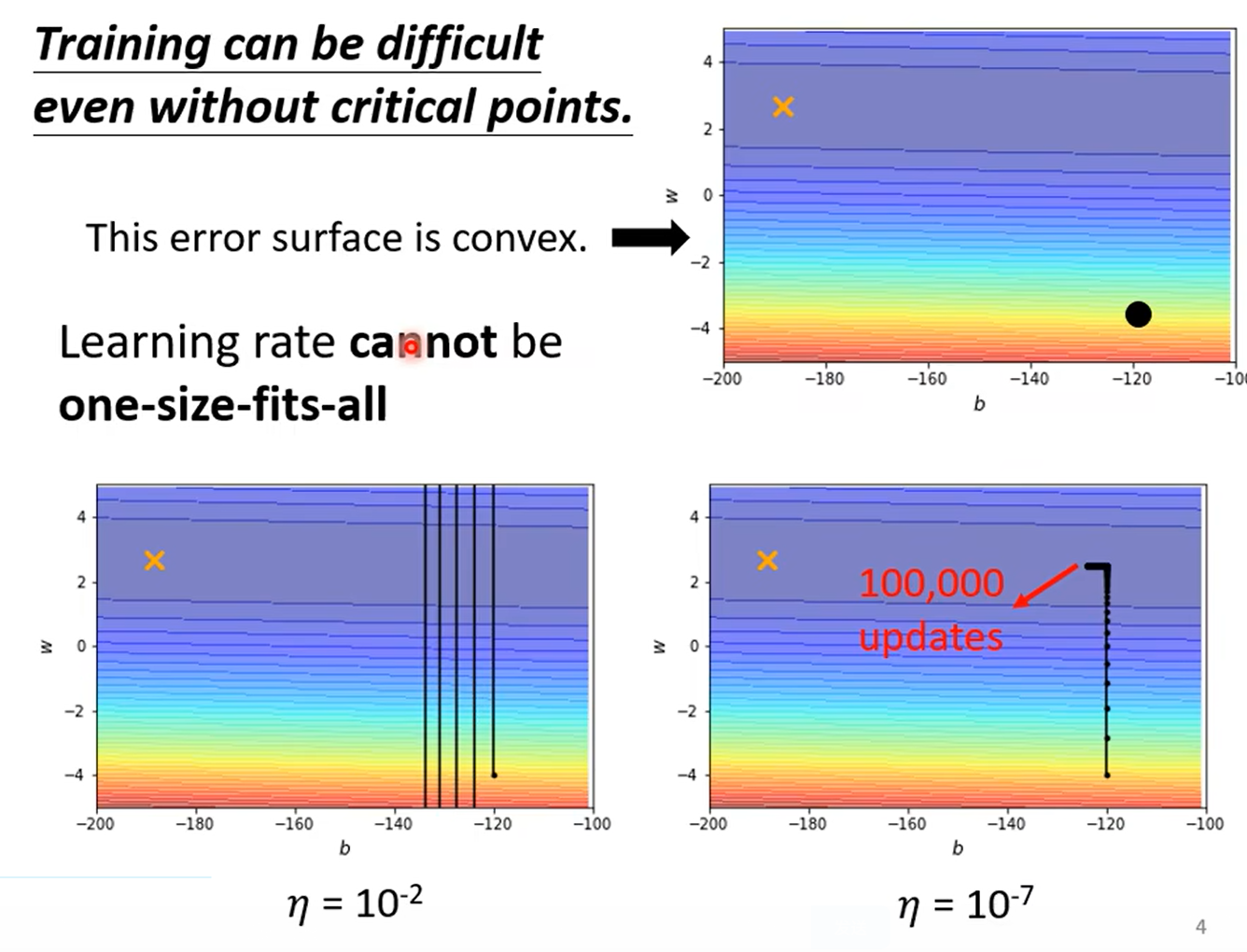
## 1.loss不再下降的时候，并非梯度就完全变成接近0，而是因为学习率有点大，在峡谷中不断振荡



也就是说很多时候loss还没有走到极小值点或者鞍点，就已经下降不下去了

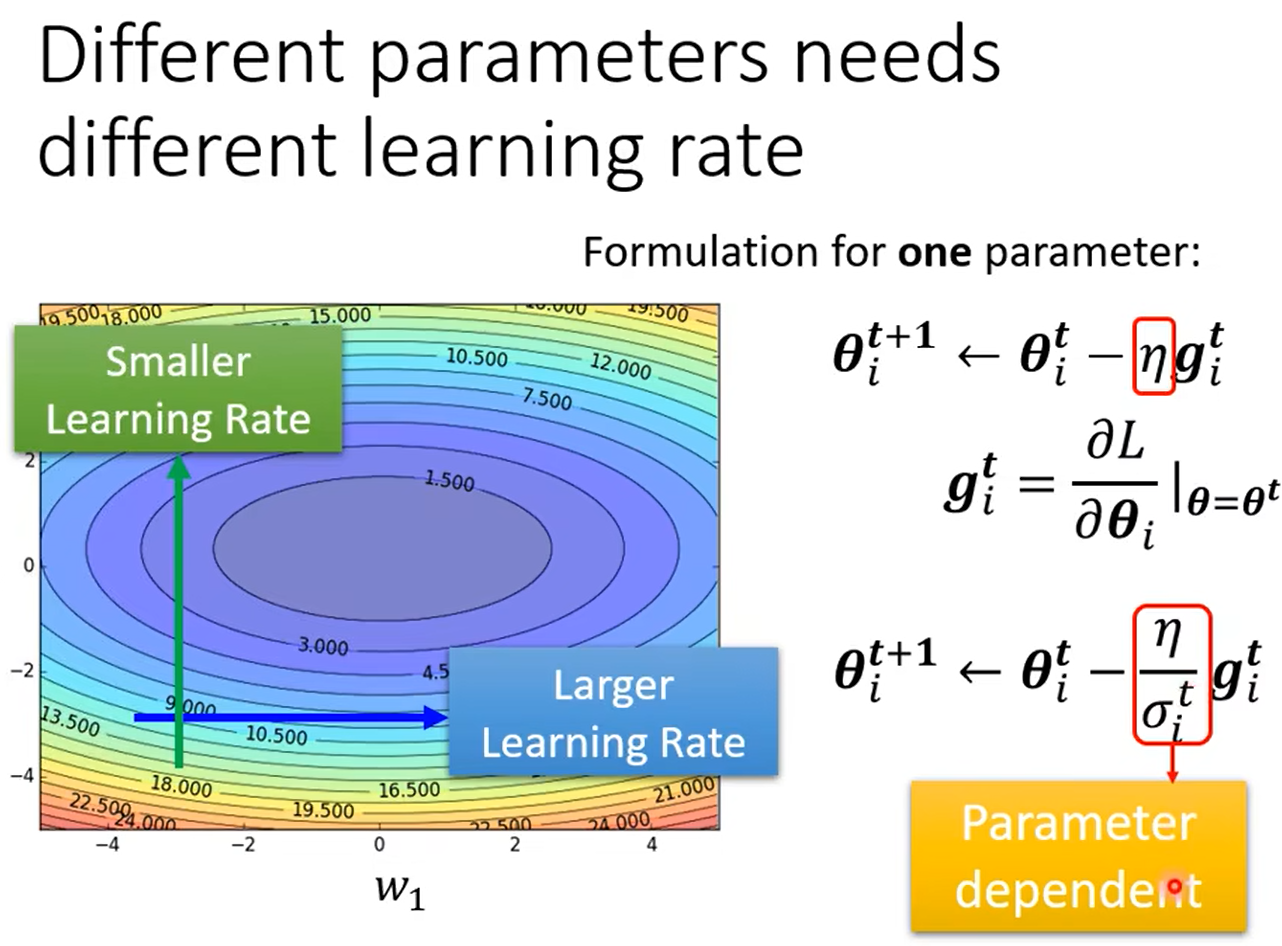
## 2.例子



左边学习率比较大，不断振荡

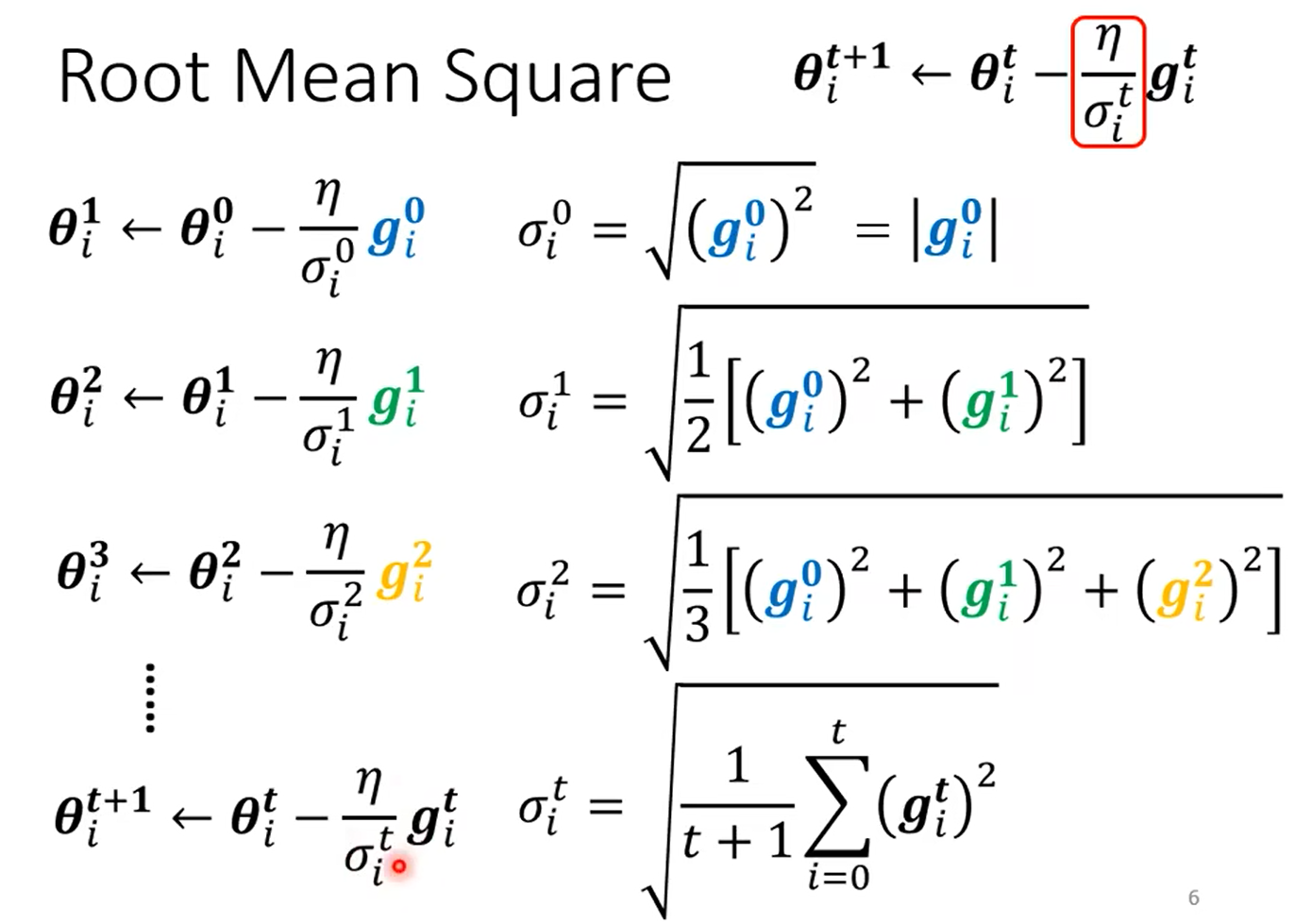
右边学习率比较小，向左边走很慢

## 3.定制化学习率（平坦的方向学习率应该大，陡峭的方向学习率应该小）



Parameter-dependent的learning-rate，跟参数本身有关，跟训练轮次t有关

### 3.1利用RMSE计算，称为adagrad

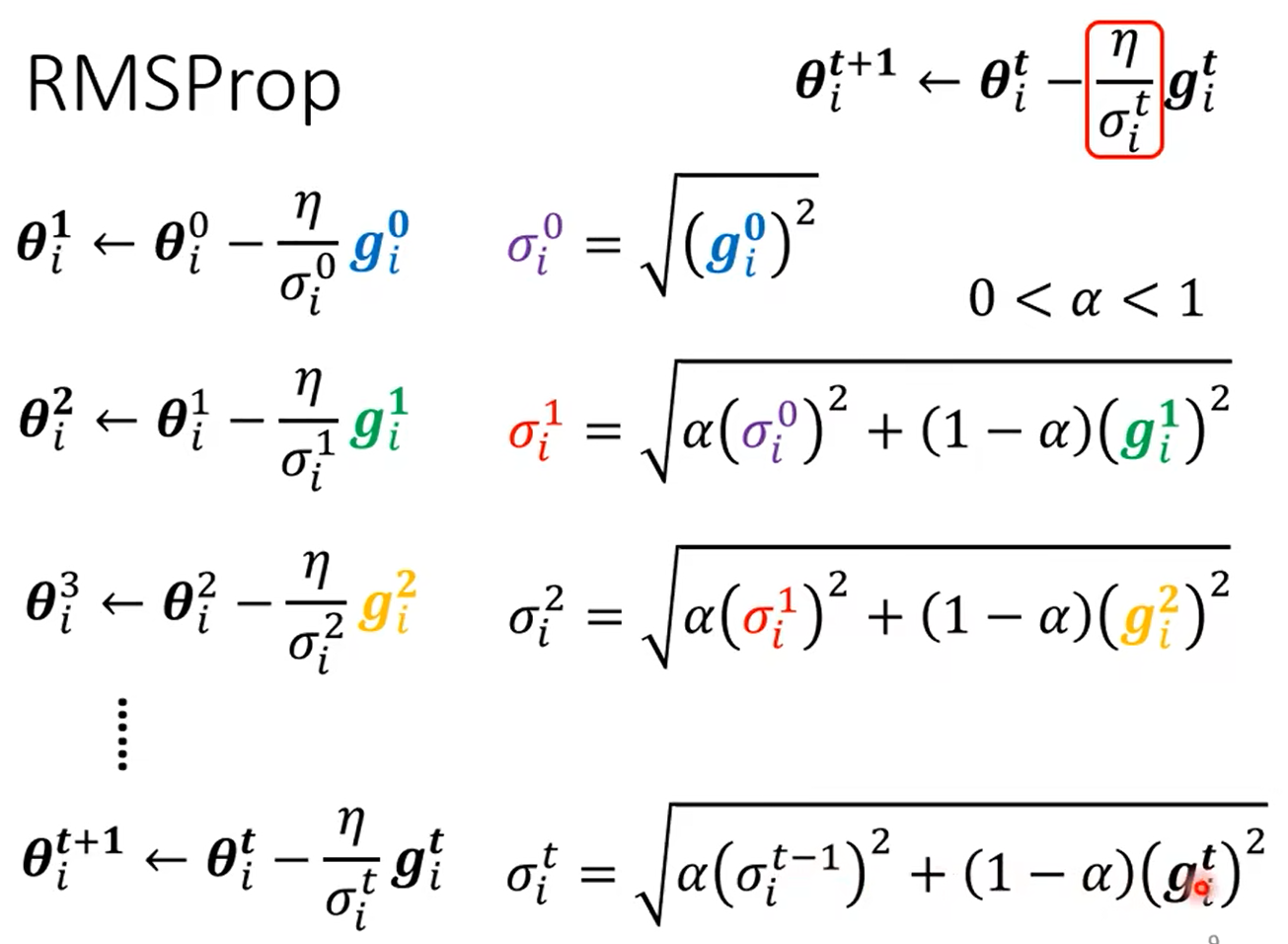


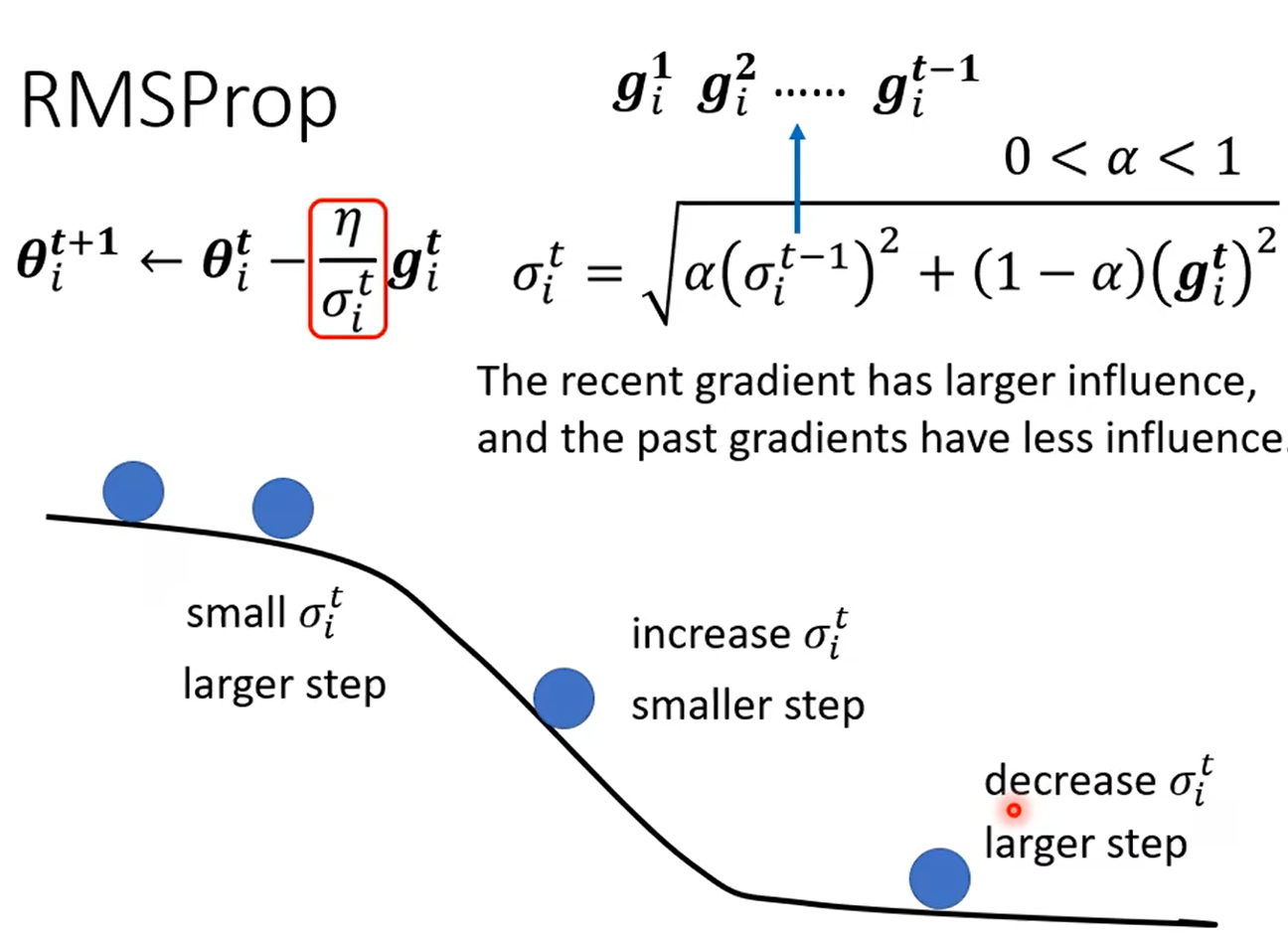


可见平坦的地方，比较小，则真实的步长比较大，加速收敛

陡峭的地方，比较大，则真实的步长比较小，避免振荡

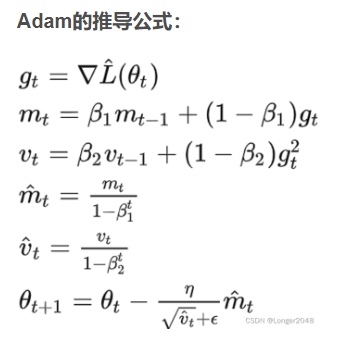
### 3.2 rmsprop在adagrad的基础上，使得关注不同时间的g的重要性，对最新的g很敏感（22分47秒左右有很好的的说明）





### 3.3 adam=rmsprop+动量，内置的超参数其实就很好，adam其实没有考虑学习率衰减或者warm up

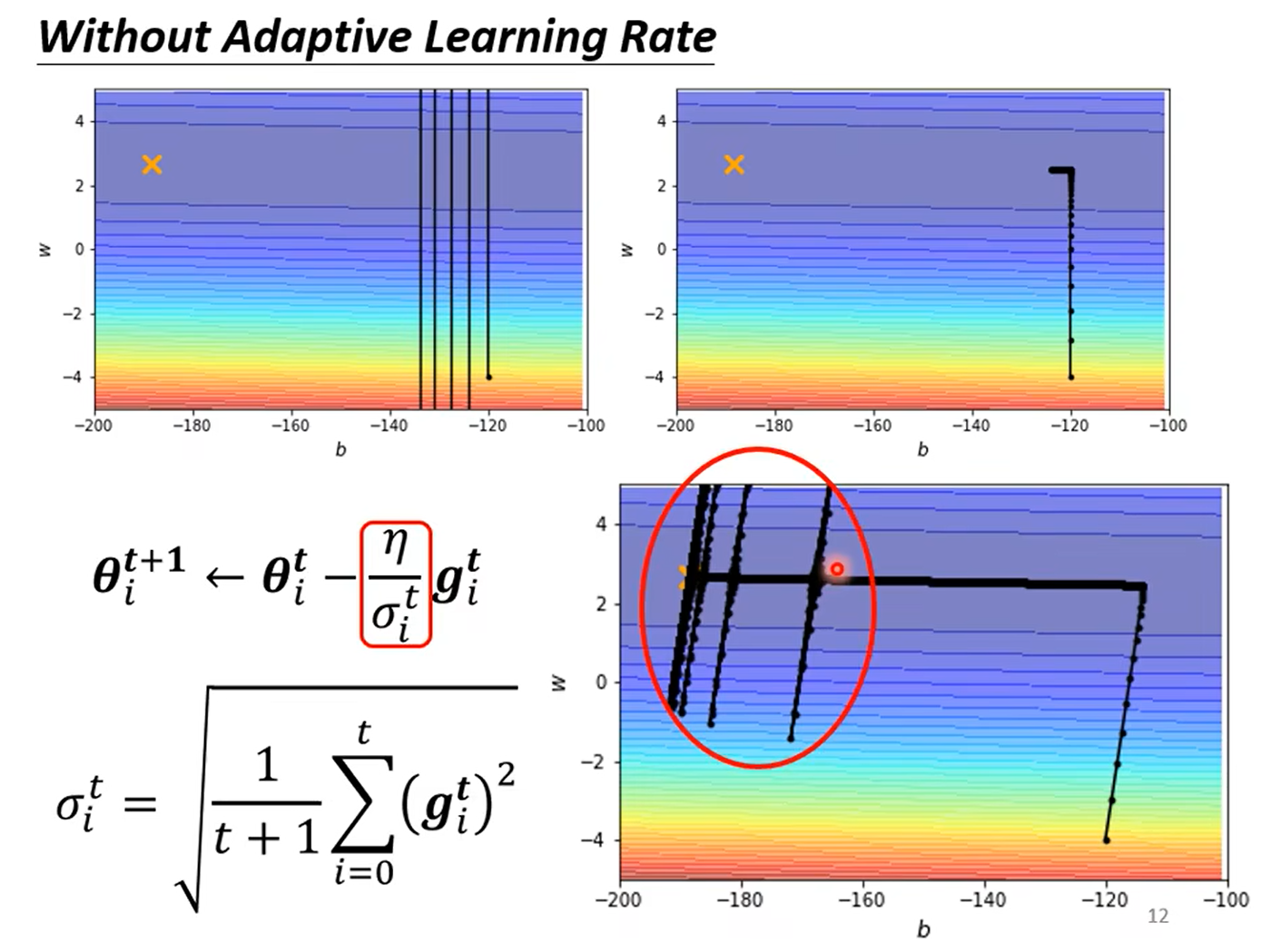




只是不是，也就是没有使用学习率衰减

## 4.利用了定制化学习率对2例子的影响

### 4.1 adagrad

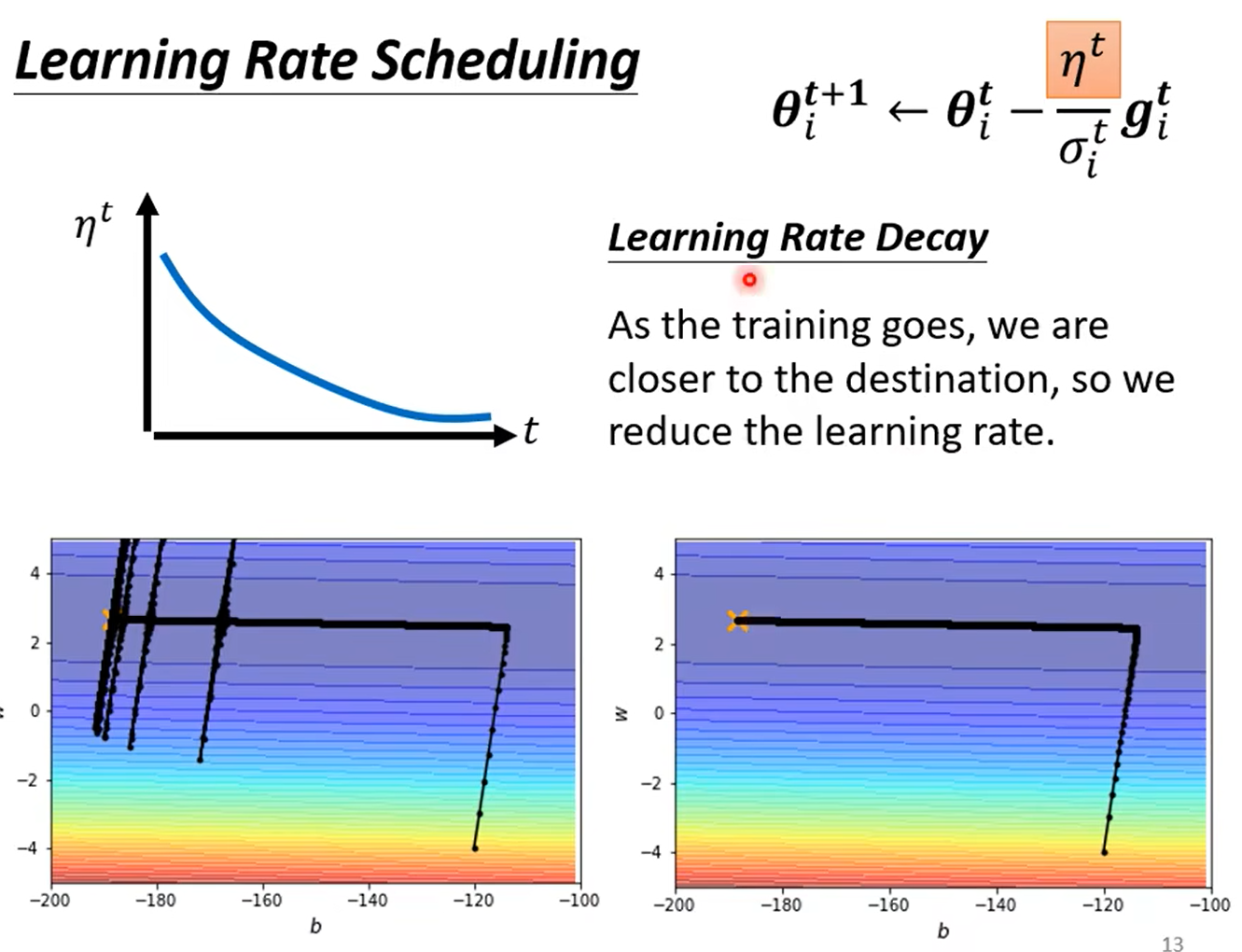


好处：向左边明显加快了，因为向左的梯度很小，则除以之后，真实的步长很大，加速了收敛

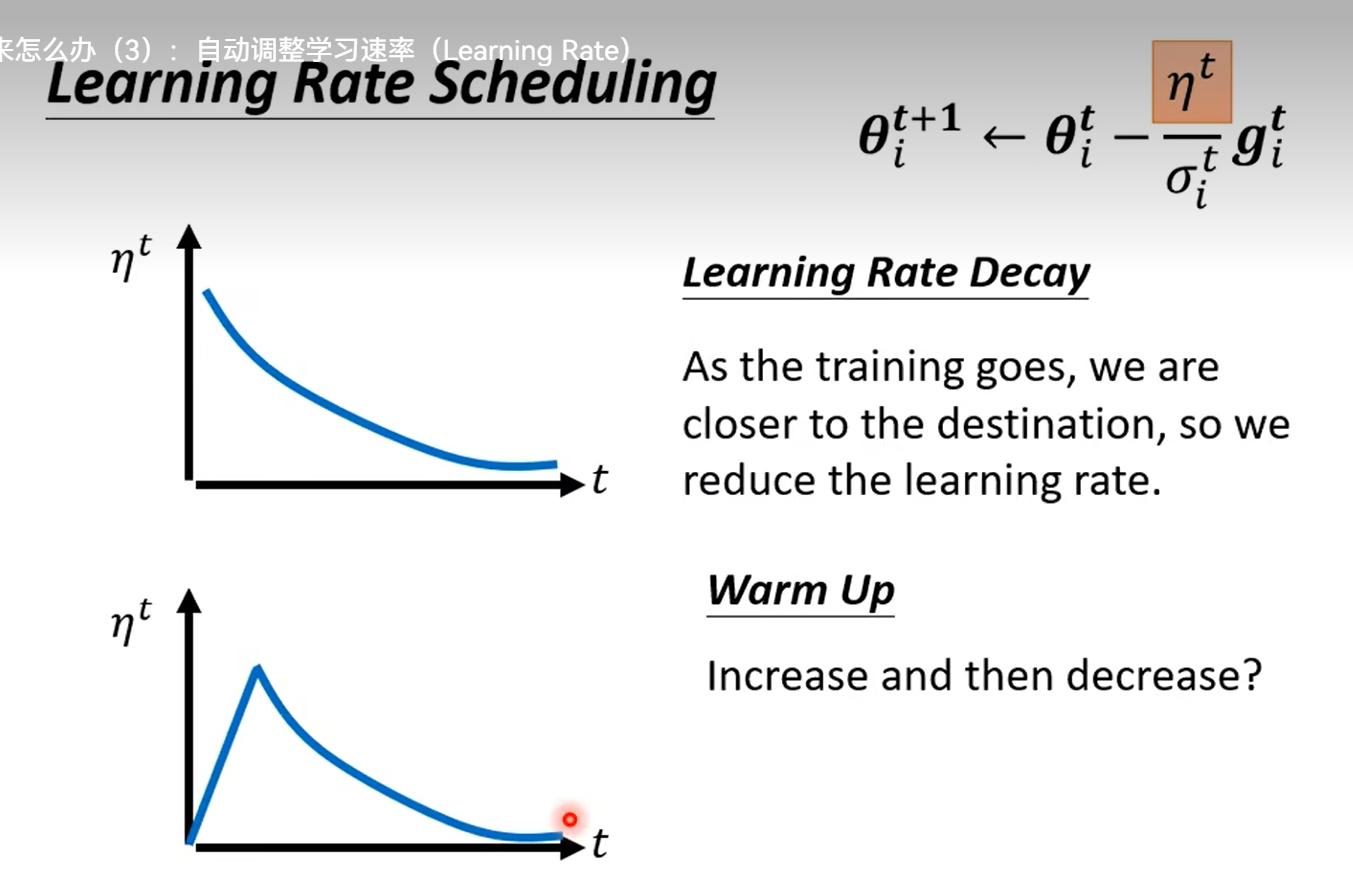
还存在的问题：在某些地方会突然振荡，因为向左边走了很长一段之后，在上下方向累积了很多很小的，从而实际的步长很大，造成了振荡。但是并不是一直上下震荡，因为上下碰壁之后g变大，则变大，则实际的步长很小，则上下方向的振荡幅度又会明显减小。

### 4.2 在自适应学习率的时候，还可以加上learning rate scheduling，也就是学习率衰减

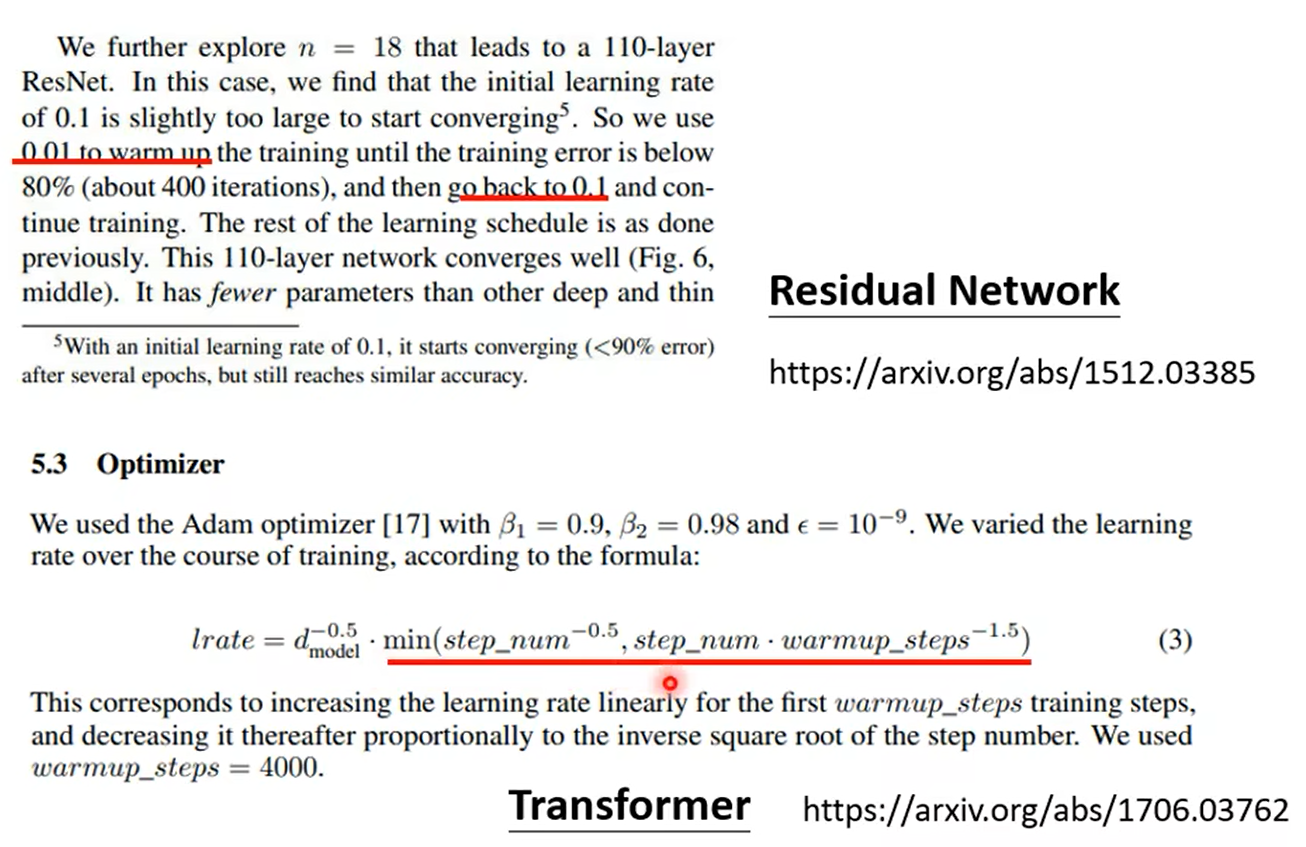
#### 4.2.1 学习率衰减



#### 4.2.2 warm up

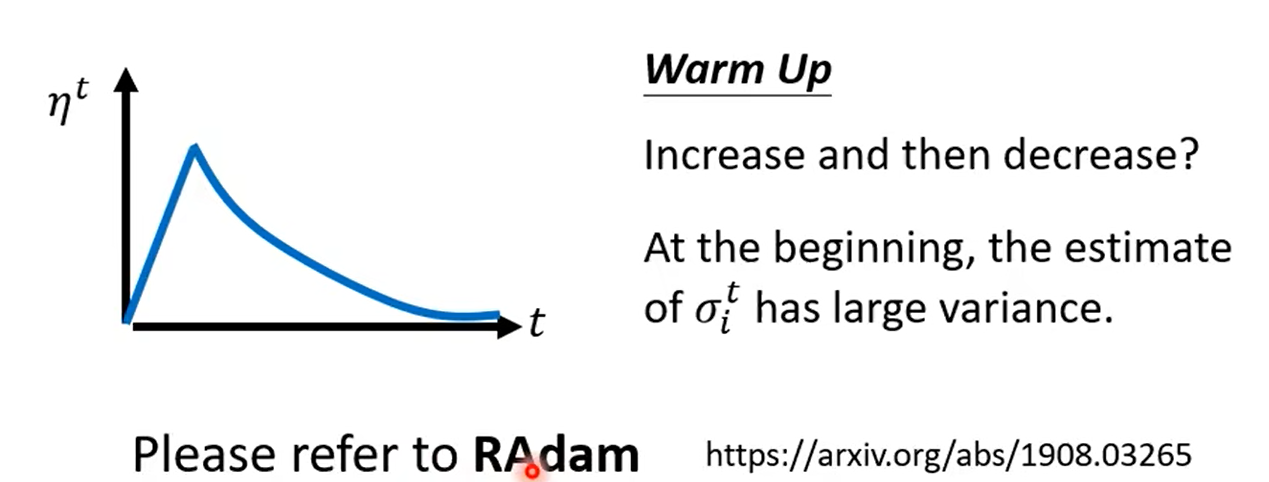


其实warm up在resnet和transformer中就有使用



为什么要用warm up？

一开始的（是对之间的g的统计，例如RMSE）不是很精准，实际有很大的偏差，因此让刚开始的学习率更小一点



## 5.结论

