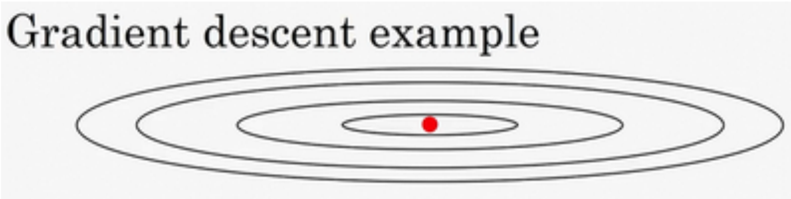
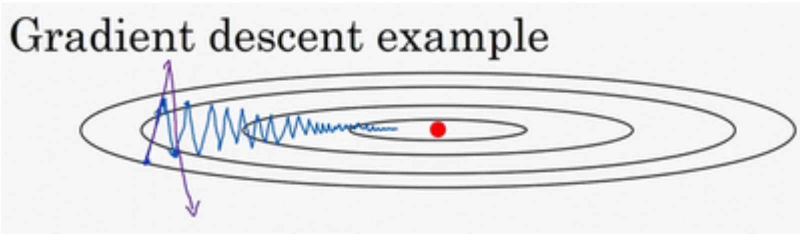
2.6 动量梯度下降法（Gradient descent with Momentum）

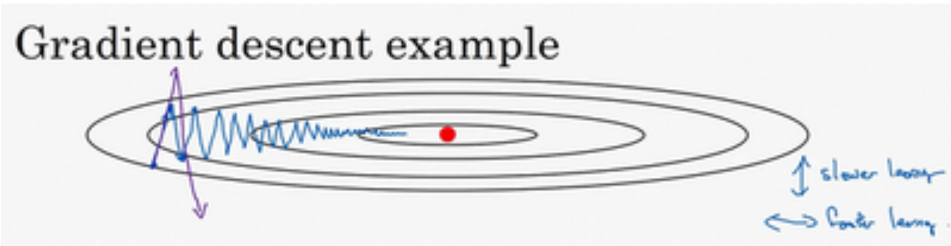
还有一种算法叫做 **Momentum**，或者叫做动量梯度下降法，运行速度几乎总是快于标准的梯度下降算法，简而言之，**基本的想法就是计算梯度的指数加权平均数，并利用该梯度更新你的权重**，在本视频中，我们呢要一起拆解单句描述，看看你到底如何计算。



例如，如果你要优化成本函数，函数形状如图，红点代表最小值的位置，假设你从这里（蓝色点）开始梯度下降法，如果进行梯度下降法的一次迭代，无论是 **batch** 或 **mini-batch** 下降法，也许会指向这里，现在在椭圆的另一边，计算下一步梯度下降，结果或许如此，然后再计算一步，再一步，计算下去，你会发现梯度下降法要很多计算步骤对吧？



慢慢摆动到最小值，这种上下波动减慢了梯度下降法的速度，你就无法使用更大的学习率，如果你要用较大的学习率（紫色箭头），结果可能会偏离函数的范围，为了避免摆动过大，你要用一个较小的学习率。



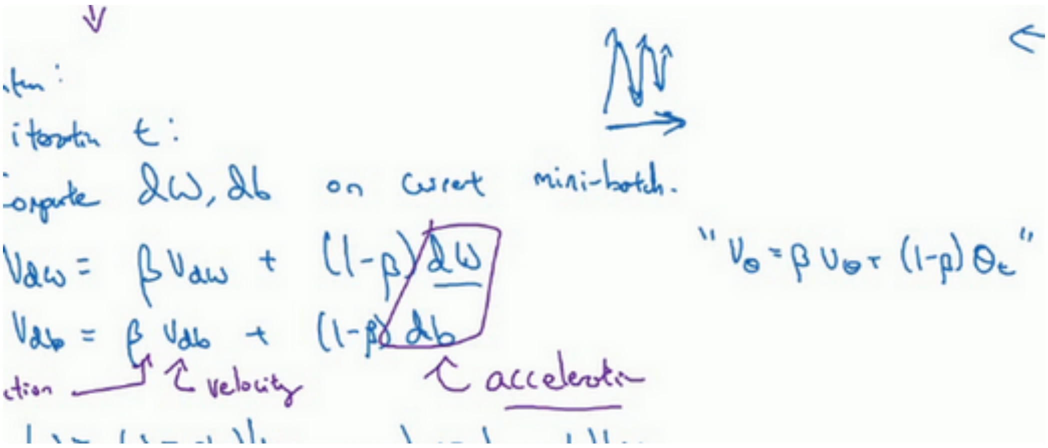
另一个看待问题的角度是，在纵轴上，你希望学习慢一点，因为你不想要这些摆动，但是在横轴上，你希望加快学习，你希望快速从左向右移，移向最小值，移向红点。所以使用动量梯度下降法，你需要做的是，在每次迭代中，确切来说在第t次迭代的过程中，你会计算微分dW，db，我会省略上标$[l]$，你用现有的 **mini-batch** 计算dW，db。如果你用 **batch** 梯度下降法，现在的 **mini-batch** 就是全部的 **batch**，对于 **batch** 梯度下降法的效果是一样的。如果现有的 **mini-batch** 就是整个训练集，效果也不错，你要做的是计算$v_{dw} = \beta v_{dw} +$

$(1-\beta)dW$，这跟我们之前的计算相似，也就是$v=\beta v+(1-\beta)\theta_t$，$dW$的移动平均数，接着同样地计算$v_{db}$，$v_{db}=\beta v_{db}+(1-\beta)db$，然后重新赋值权重，$W:=W-av_{dW}$，同样$b:=b-av_{db}$，这样就可以减缓梯度下降的幅度。

例如，在上几个导数中，你会发现这些纵轴上的摆动平均值接近于零，所以在纵轴方向，你希望放慢一点，平均过程中，正负数相互抵消，所以平均值接近于零。但在横轴方向，所有的微分都指向横轴方向，因此横轴方向的平均值仍然较大，因此用算法几次迭代后，你发现动量梯度下降法，最终纵轴方向的摆动变小了，横轴方向运动更快，因此你的算法走了一条更加直接的路径，在抵达最小值的路上减少了摆动。

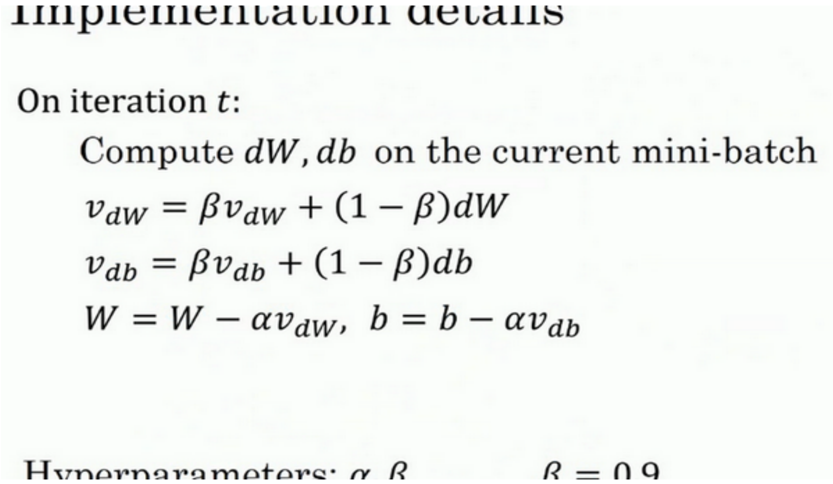
动量梯度下降法的一个本质，这对有些人而不是所有人有效，就是如果你要最小化碗状函数，这是碗的形状，我画的不太好。

它们能够最小化碗状函数，这些微分项，想象它们为你从山上往下滚的一个球，提供了加速度，**Momentum** 项相当于速度。

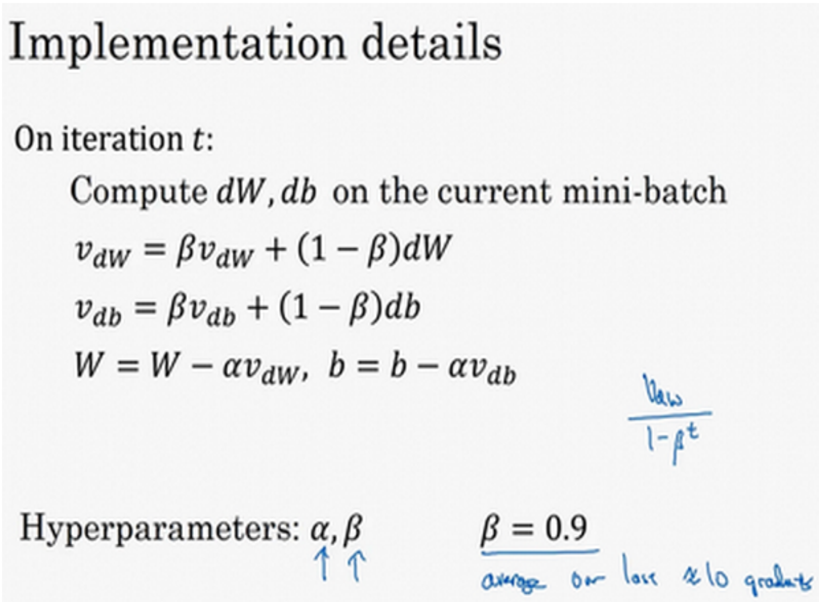


想象你有一个碗，你拿一个球，微分项给了这个球一个加速度，此时球正向山下滚，球因为加速度越滚越快，而因为β 稍小于 1，表现出一些摩擦力，所以球不会无限加速下去，所以不像梯度下降法，每一步都独立于之前的步骤，你的球可以向下滚，获得动量，可以从碗向下加速获得动量。我发现这个球从碗滚下的比喻，物理能力强的人接受得比较好，但不是所有人都能接受，如果球从碗中滚下这个比喻，你理解不了，别担心。

最后我们来看具体如何计算，算法在此。



所以你有两个超参数，**学习率a以及参数β**，β控制着指数加权平均数。β最常用的值是0.9，我们之前平均了过去十天的温度，所以现在平均了前十次迭代的梯度。实际上β为 0.9 时，效果不错，你可以尝试不同的值，可以做一些超参数的研究，不过 0.9 是很棒的鲁棒数。那么关于偏差修正，所以你要拿v_{dW}和v_{db}除以$1-\beta^t$，实际上人们不这么做，因为 10 次迭代之后，因为你的移动平均已经过了初始阶段。实际中，在使用梯度下降法或动量梯度下降法时，人们不会受到偏差修正的困扰。当然v_{dW}初始值是 0，要注意到这是和dW拥有相同维数的零矩阵，也就是跟W拥有相同的维数，v_{db}的初始值也是向量零，所以和db拥有相同的维数，也就是和b是同一维数。



最后要说一点，如果你查阅了动量梯度下降法相关资料，你经常会看到一个被删除了的专业词汇，$1-\beta$被删除了，最后得到的是$v_{dW} = \beta v_{dW} + dW$。用紫色版本的结果就是，所以$v_{dW}$缩小了$1-\beta$倍，相当于乘以$\frac{1}{1-\beta}$，所以你要用梯度下降最新值的话，$a$要根据$\frac{1}{1-\beta}$相应

变化。实际上，二者效果都不错，只会影响到学习率α的最佳值。我觉得这个公式用起来没有那么自然，因为有一个影响，如果你最后要调整超参数β，就会影响到v_{dW}和v_{db}，你也许还要修改学习率α，所以我更喜欢左边的公式，而不是删去了$1-\beta$的这个公式，所以我更倾向于使用左边的公式，也就是有$1-\beta$的这个公式，但是两个公式都将β设置为 0.9，是超参数的常见选择，只是在这两个公式中，学习率α的调整会有所不同。

所以这就是动量梯度下降法，这个算法肯定要好于没有 **Momentum** 的梯度下降算法，我们还可以做别的事情来加快学习算法，我们将在接下来的视频中探讨这些问题。