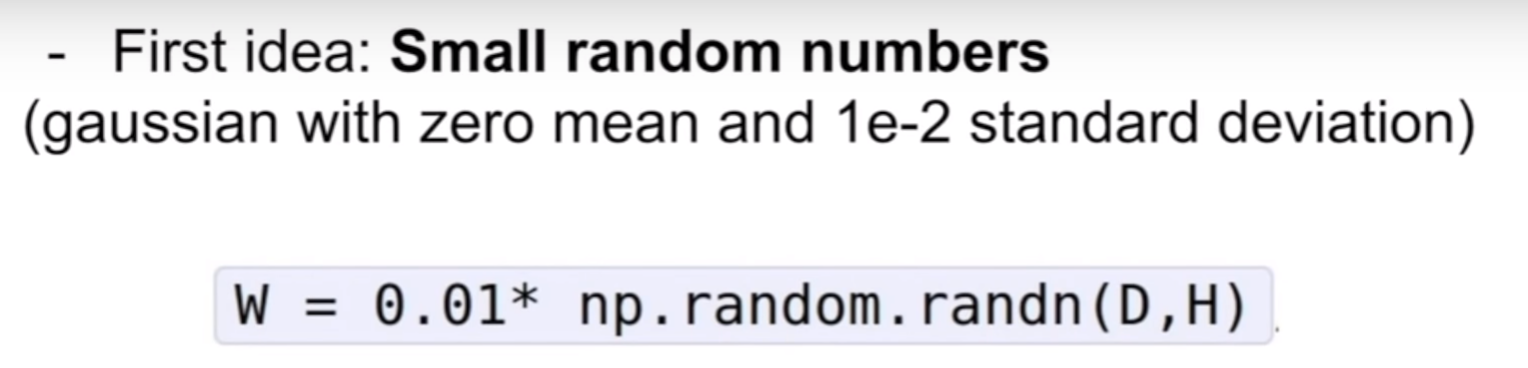
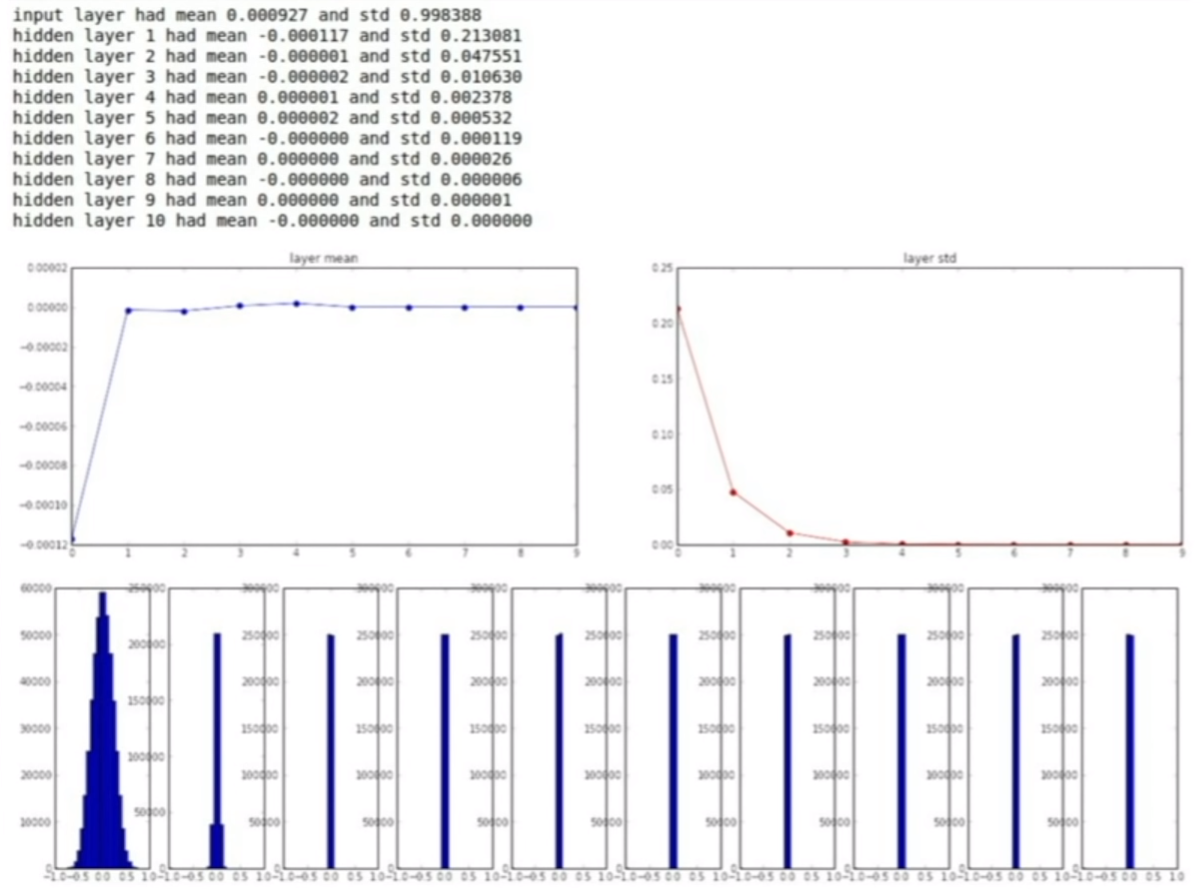
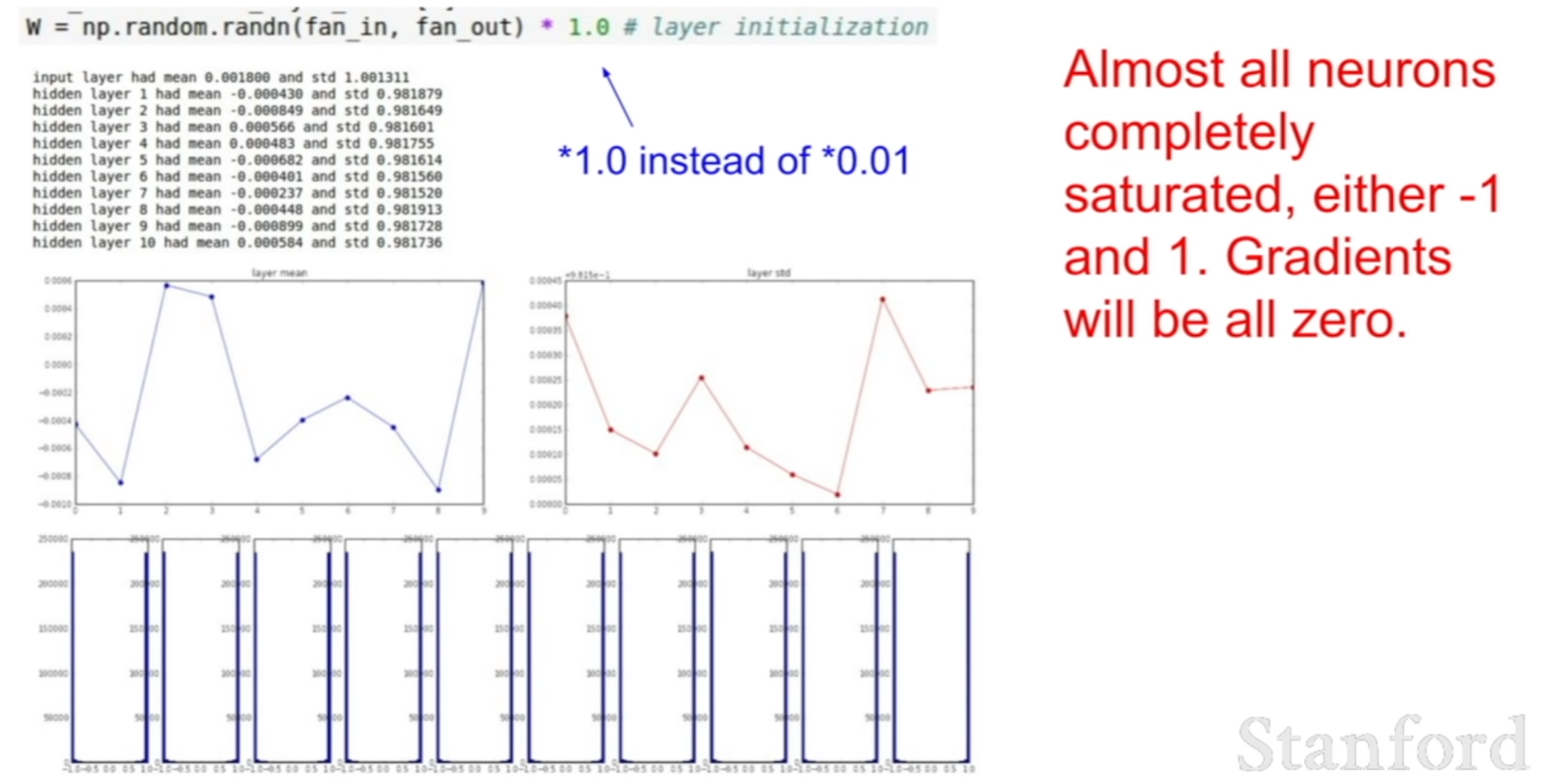


结果：所有神经元都会做相同的事情，但它们不一定全部都die了，主要取悦于输入数据。权重梯度都会是一样，都会以相同的方式进行更新，但是我们希望神经元之间的操作是不同的。



注：这个初始化方式可行，尤其是小网络，但是对层级比较深的网络就会产生问题，为什么？下图所以为10层网络，500个神经元的实验。在前向传播过程中，从图中可以看出，mean总是为0是合理的，但是std收缩很快然后坍塌为0，所以的激活值都变成0。在反向传播过程中，因为输入数据总是很小，当我们计算梯度时是通过upstream和local的梯度之积得到的，而local的梯度结果就是输入X，当X很小时，我们得到这个梯度也将会更小，那么权重更新将会非常慢。

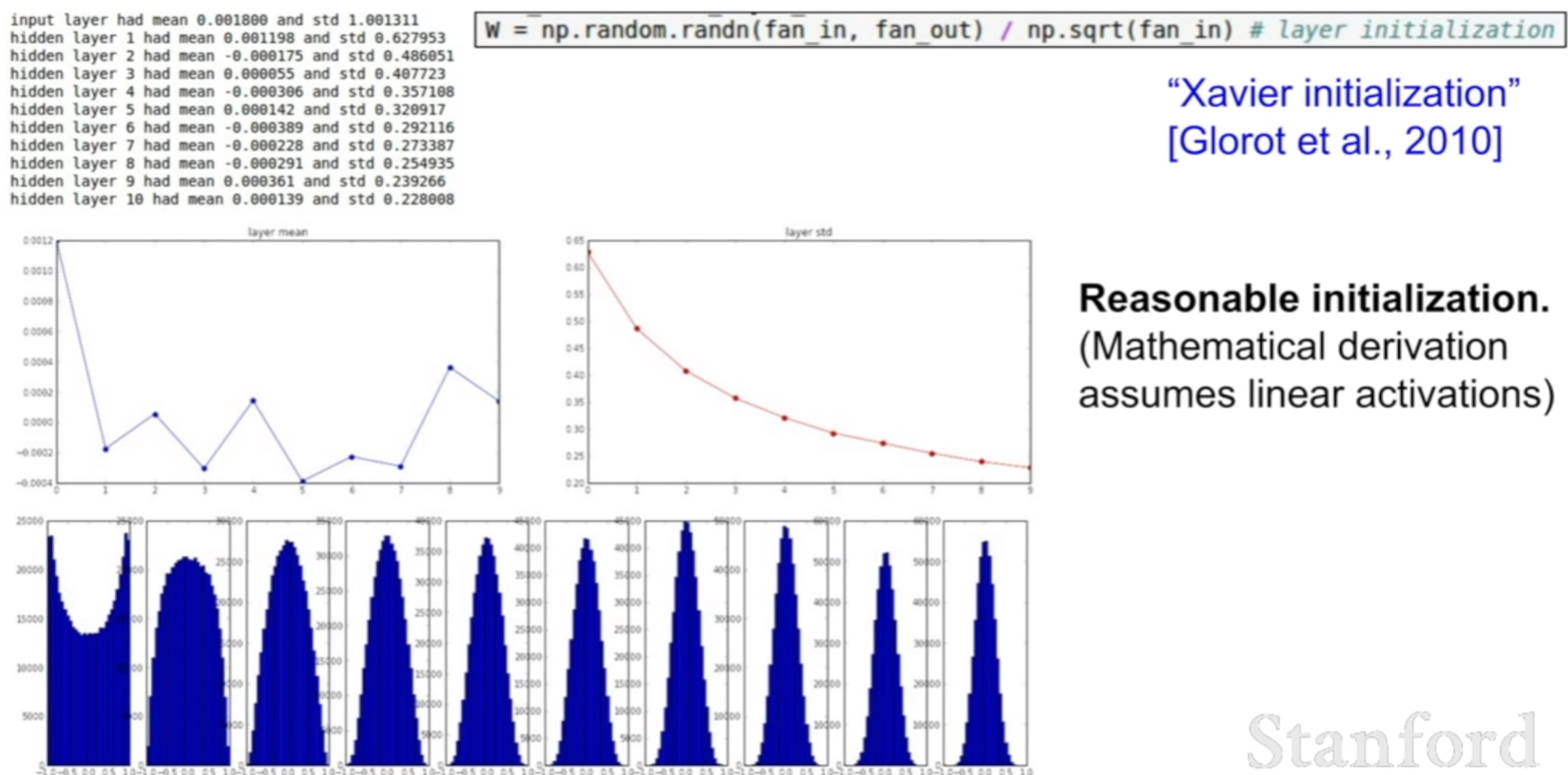




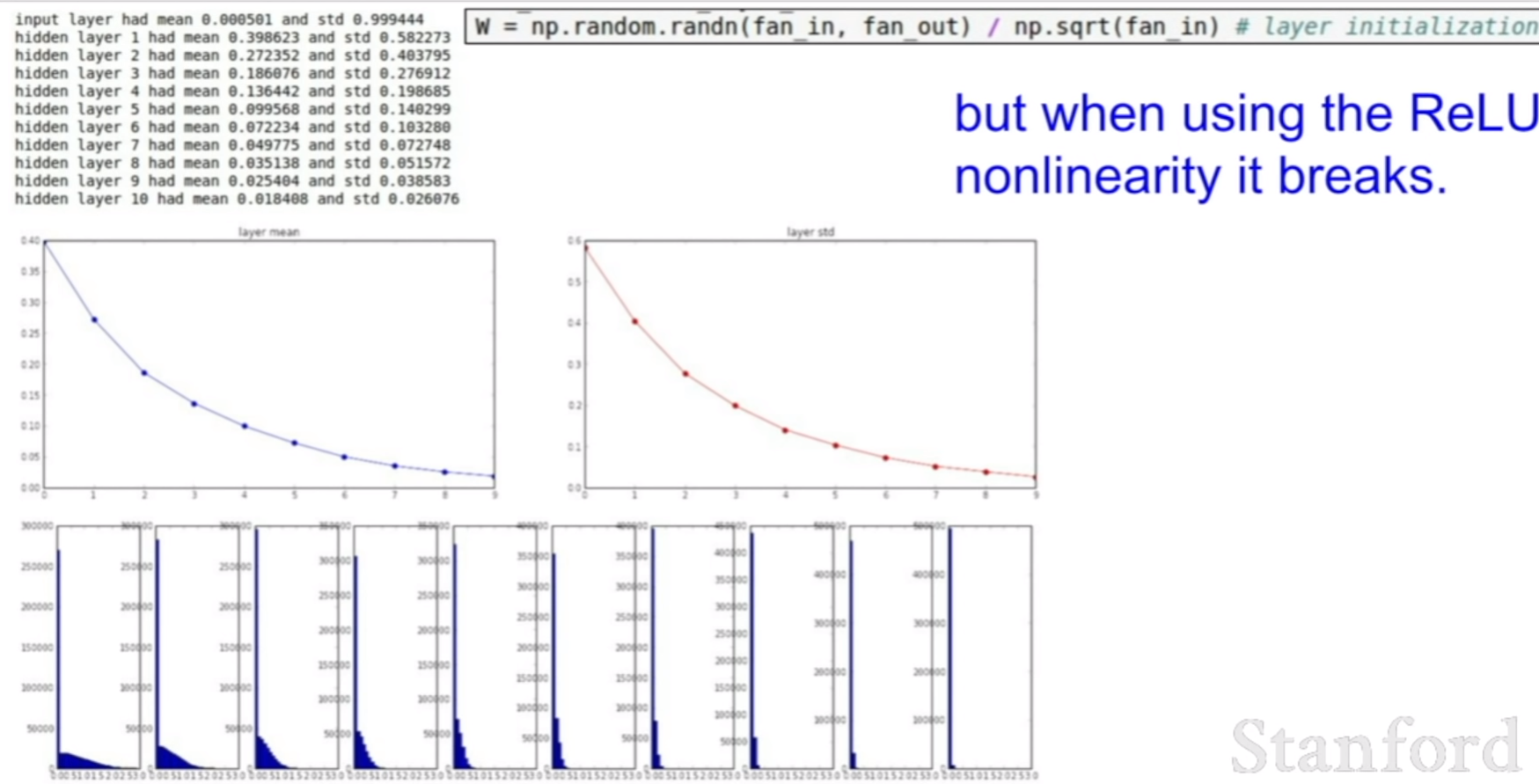
如果把这个小的初始化值从0.01改到1.0，结果则是所有的神经元激活值都会饱和，要么-1，要么1，因为在这个实验中使用的是tanh激活函数，所以它们的梯度都为0。

**可行的权重初始化方式：1. Xavier Initialization**

在高斯分布的基础上采用输入size的开方进行scale。这种操作的方式就是希望输入的方差与输出的方差一致。直观理解：如果输入数据集很小，那么就会获得较大的权重初始化；如果输入数据集很大，那么就会获得较小的权重初始化。



***但是这种初始化公式的数学推导是假设采用线性激活函数的，但激活函数不是线性的，就会产生其他问题，阐述如下。***



为什么ReLU会产生这种情况？因为ReLU会砍掉一半的神经元，本质上就相当于每次都对方差减半，所以就跟之前较小值初始化差不多，方差减小程度很大，最终又消失为0。

**可行的权重初始化方式：2. 何凯明 Initialization**

就是在Xavier的基础上在分母除以2再开方，为什么这样就有效？因为我们上面分析说的是ReLU的方差会被砍掉一半，那么我们默认原始输入的方差本来就是砍掉后的一半，那么输出的就会跟输入的方差一致，所以才在输入size提前除以2后再进行开方。

