

Deep Learning for Data Science

DS 542

<https://dl4ds.github.io/fa2025/>

Initialization



Plan for Today

- Project 1
- The need for weights initialization
- Expectations Refresher
- The (Kaiming) He Initialization
- Lottery tickets

Initialization

- Consider standard building block of NN in terms of pre-activations:

$$\begin{aligned}\mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k \\ &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{a}[\mathbf{f}_{k-1}]\end{aligned}$$

- How do we initialize the biases and weights?
- Equivalent to choosing starting point in our gradient descent searches

Forward Pass

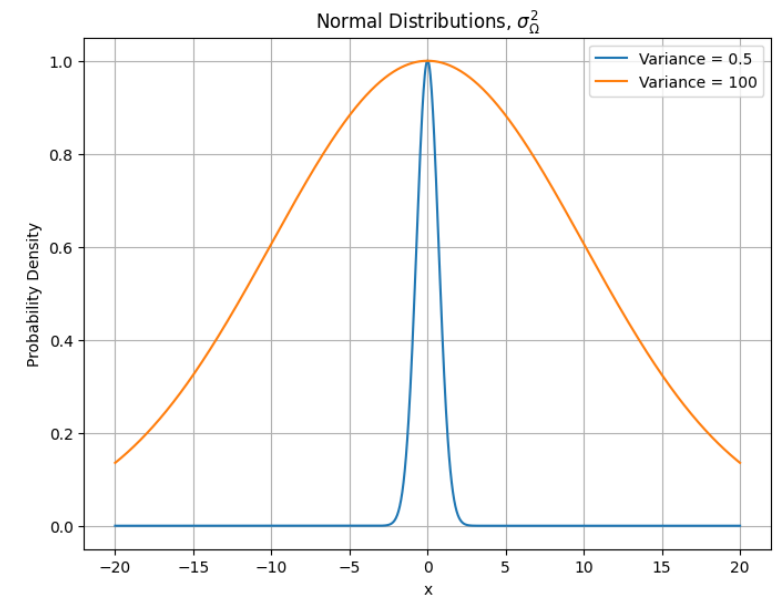
- Consider standard building block of NN in terms of *pre-activations*:

$$\begin{aligned}\mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k \\ &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{a}[\mathbf{f}_{k-1}]\end{aligned}$$

- Set all the biases to 0

$$\boldsymbol{\beta}_k = \mathbf{0}$$

- Set weights to be normally distributed
 - mean 0
 - variance σ_{Ω}^2

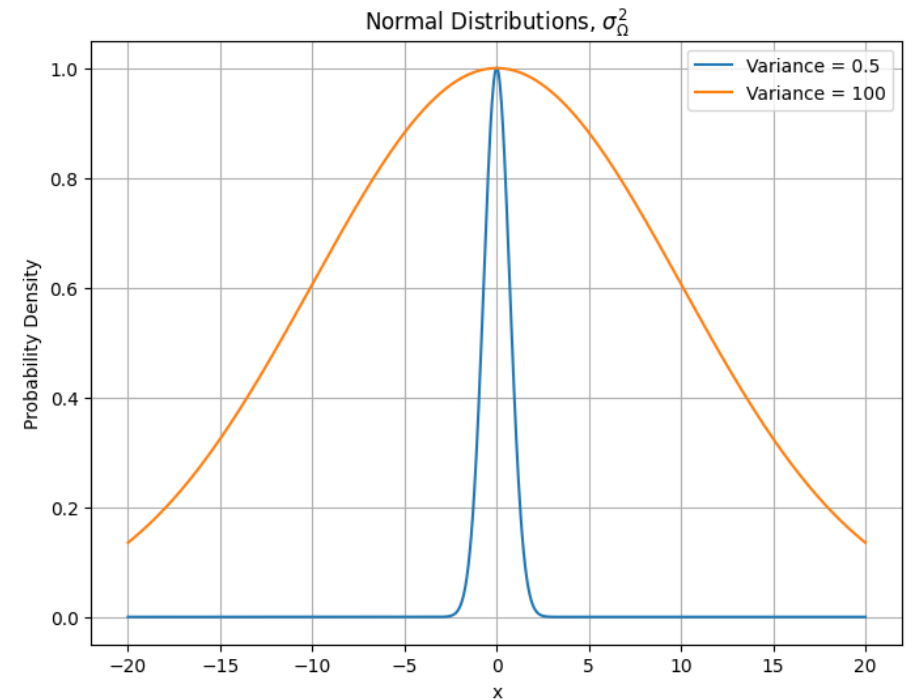


- What will happen as we move through the network if σ_{Ω}^2 is very small?
- What will happen as we move through the network if σ_{Ω}^2 is very large?

Backward Pass

$$\frac{\partial \ell_i}{\partial \mathbf{f}_{k-1}} = \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial \ell_i}{\partial \mathbf{f}_k} \right), \quad k \in \{K, K-1, \dots, 1\} \quad (7.13)$$

- What will happen as we propagate backwards through the network if σ_Ω^2 is very small?
- What will happen as we propagate backwards through the network if σ_Ω^2 is very large?



Initialize weights to different variances

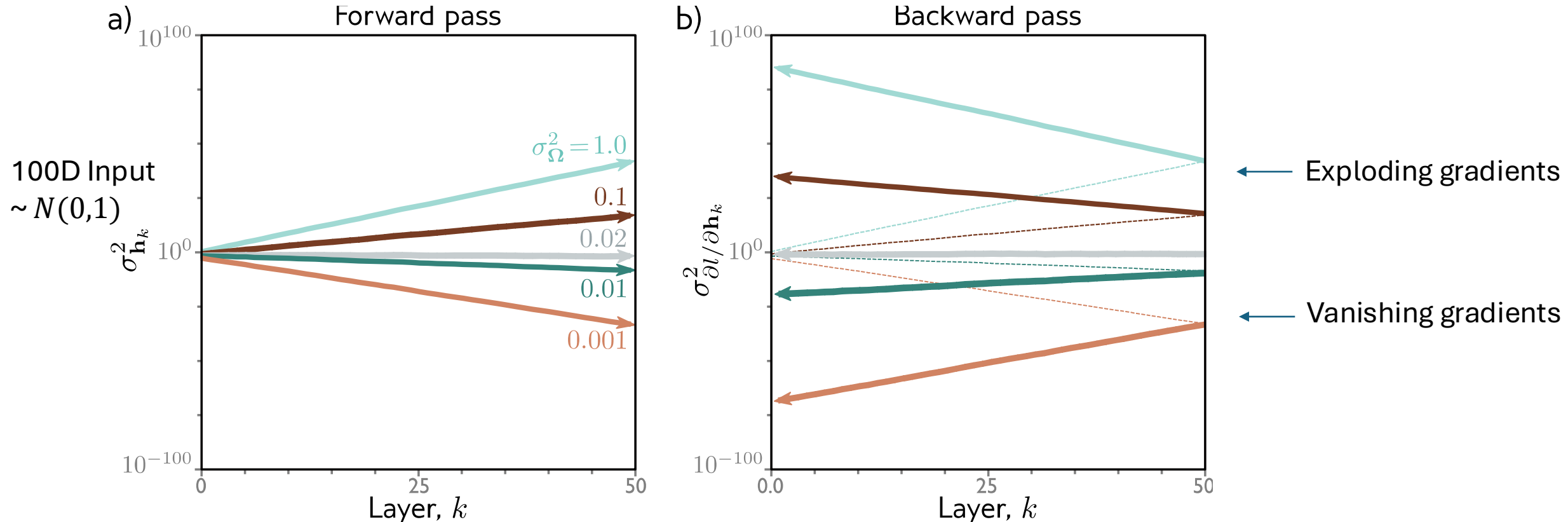


Figure 7.4 Weight initialization. Consider a deep network with 50 hidden layers and $D_h = 100$ hidden units per layer. The network has a 100 dimensional input \mathbf{x} initialized with values from a standard normal distribution, a single output fixed at $y = 0$, and a least squares loss function. The bias vectors β_k are initialized to zero and the weight matrices Ω_k are initialized with a normal distribution with mean zero and five different variances $\sigma^2_{\Omega} \in \{0.001, 0.01, 0.02, 0.1, 1.0\}$. a)

How do we initialize weights to keep variance stable across layers?

Aim: keep variance same between two layers

$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$$

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$

Definition of variance:

$$\sigma_{f'}^2 = \mathbb{E}[(f'_i - \mathbb{E}[f'_i])^2]$$

Any Questions?



- The need for weights initialization
- **Expectations Refresher**
- The (Kaiming) He initialization
- Lottery tickets

Expectations

$$\mathbb{E}[g[x]] = \int g[x] Pr(x) dx,$$

Interpretation: what is the average value of $g[x]$ when taking into account the probability of x ?

Consider discrete case and assume uniform probability so calculating $g[x]$ reduces to taking average:

$$\mathbb{E}[g[x]] \approx \frac{1}{N} \sum_{n=1}^N g[x_n^*] \quad \text{where} \quad x_n^* \sim Pr(x)$$

Common Expectation Functions

Function $g[\bullet]$	Expectation
x	mean, μ
x^k	k th moment about zero
$(x - \mu)^k$	k th moment about the mean
$(x - \mu)^2$	variance
$(x - \mu)^3$	skew
$(x - \mu)^4$	kurtosis

Table B.1 Special cases of expectation. For some functions $g[x]$, the expectation $\mathbb{E}[g[x]]$ is given a special name. Here we use the notation μ_x to represent the mean with respect to random variable x .

Rules for manipulating expectation

$$\mathbb{E}[k] = k$$

$$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]] \quad \text{if } x, y \text{ independent}$$

Any Questions?



- The need for weights initialization
- Expectations Refresher
- The (Kaiming) He initialization
- Lottery tickets

Aim: keep variance same between two layers

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$
$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$$

Definition of variance:

$$\sigma_{f'_i}^2 = \mathbb{E}[(f'_i - \mathbb{E}[f'_i])^2]$$

Now let's prove:

$$\mathbb{E} [(x - \mu)^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

Keeping in mind:

$$\mathbb{E}[x] = \mu$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$

$$\mathbb{E}[(x - \mu)^2] = \mathbb{E}[x^2 - 2x\mu + \mu^2]$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$

$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\ &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2]\end{aligned}$$

Rule 1:	$\mathbb{E}[k] = k$	←
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$	←
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$	
Def'n	$\mathbb{E}[x] = \mu$	


$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\ &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\ &= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2\end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$



$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\ &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\ &= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\ &= \mathbb{E}[x^2] - 2\mu^2 + \mu^2\end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$

$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\ &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\ &= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\ &= \mathbb{E}[x^2] - 2\mu^2 + \mu^2 \\ &= \mathbb{E}[x^2] - \mu^2\end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$



$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\&= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\&= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\&= \mathbb{E}[x^2] - 2\mu^2 + \mu^2 \\&= \mathbb{E}[x^2] - \mu^2 \\&= \mathbb{E}[x^2] - E[x]^2\end{aligned}$$

Aim: keep variance same between two layers

$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$$

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$

$$\sigma_{f'}^2 = \mathbb{E}[(f'_i - \mathbb{E}[f'_i])^2]$$

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$

$$\longrightarrow \mathbb{E}[(x - \mu)^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

Aim: keep variance same between two layers

$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$$

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$

$$\sigma_{f'}^2 = \mathbb{E}[(f'_i - \mathbb{E}[f'_i])^2]$$

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$


 Focus on this term.

Aim: keep variance same between two layers


$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$$

Consider the mean of the pre-activations:

$$\mathbb{E}[f'_i] = \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right]$$

- Rule 1: $\mathbb{E}[k] = k$
- Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
- Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
- Rule 4: $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent
- 

$$\begin{aligned}\mathbb{E}[f'_i] &= \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right] \\ &= \mathbb{E}[\beta_i] + \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij} h_j]\end{aligned}$$

- Rule 1: $\mathbb{E}[k] = k$
- Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
- Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
- Rule 4: $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent
- 

$$\begin{aligned}\mathbb{E}[f'_i] &= \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right] \\ &= \mathbb{E} [\beta_i] + \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij} h_j] \\ &= \mathbb{E} [\beta_i] + \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij}] \mathbb{E} [h_j]\end{aligned}$$

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\mathbb{E}[f'_i] = \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right]$$

Start making initialization choices.

- Set all the biases to 0
- Weights normally distributed
 - mean 0
 - variance σ_{Ω}^2

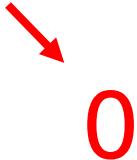
$$\begin{aligned}
 &= \mathbb{E}[\beta_i] + \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij} h_j] \\
 &= \mathbb{E}[\beta_i] + \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij}] \mathbb{E}[h_j] \\
 &= 0 + \sum_{j=1}^{D_h} 0 \cdot \mathbb{E}[h_j] = 0
 \end{aligned}$$

Aim: keep variance same between two layers

$$\mathbf{f}' = \beta + \Omega \mathbf{h}$$

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$

$$\sigma_{f'}^2 = \mathbb{E}[(f'_i - \mathbb{E}[f'_i])^2]$$

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 = \mathbb{E}[f_i'^2]$$


Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\begin{aligned}\sigma_{f'}^2 &= \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 \\ &= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0\end{aligned}$$

Set all the biases to 0

Weights normally distributed
mean 0
variance σ_{Ω}^2

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\begin{aligned}
 \sigma_{f'}^2 &= \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 \\
 &= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0 \\
 &= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right]
 \end{aligned}$$

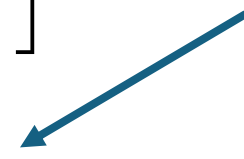
Initialization choices.

- Set all the biases to 0
- Weights normally distributed
 - mean 0
 - variance σ_{Ω}^2

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent



$$\begin{aligned}
 \sigma_{f'}^2 &= \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 \\
 &= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0 \\
 &= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] \\
 &= \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij}^2] \mathbb{E} [h_j^2]
 \end{aligned}$$



For all the cross terms, $E[\Omega_{ij}] = 0$ so only the squared terms are left, then use independence.

Initialization choices.


- Set all the biases to 0
- Weights normally distributed
 - mean 0
 - variance σ_{Ω}^2

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\begin{aligned}\sigma_{f'}^2 &= \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 \\ &= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0 \\ &= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right]\end{aligned}$$

Initialization choices.

- Set all the biases to 0
- Weights normally distributed
 - mean 0
 - variance σ_{Ω}^2

$$\begin{aligned}&= \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij}^2] \mathbb{E}[h_j^2] \\ &= \sum_{j=1}^{D_h} \sigma_{\Omega}^2 \mathbb{E}[h_j^2] = \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \mathbb{E}[h_j^2]\end{aligned}$$


Because the Ω 's are zero mean, this is the variance.

$$\begin{aligned}
\sigma_{f'}^2 &= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2] \\
&= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \mathbb{E} [\text{ReLU}[f_j]^2] \\
&= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \int_{-\infty}^{\infty} \text{ReLU}[f_j]^2 \text{Pr}(f_j) df_j \quad \leftarrow \text{From the definition of expectation.} \\
&= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \int_{-\infty}^{\infty} (\mathbb{I}[f_j > 0] f_j)^2 \text{Pr}(f_j) df_j \\
&= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \int_0^{\infty} f_j^2 \text{Pr}(f_j) df_j \quad \leftarrow \text{Only positive integral limits because of ReLU} \\
&= \sigma_{\Omega}^2 \sum_{j=1}^{D_h} \frac{\sigma_f^2}{2} = \frac{D_h \sigma_{\Omega}^2 \sigma_f^2}{2} \quad \leftarrow \frac{1}{2} \text{ of the variance for zero mean distribution}
\end{aligned}$$

Aim: keep variance same between two layers

Since:

$$\sigma_{f'}^2 = \frac{D_h \sigma_{\Omega}^2 \sigma_f^2}{2}$$

Should choose:

$$\sigma_{\Omega}^2 = \frac{2}{D_h}$$

To get:

$$\sigma_{f'}^2 = \sigma_f^2$$

Kaiming He 何恺明



<https://people.csail.mit.edu/kaiming/>

This is called **He initialization** or **Kaiming initialization**.

He initialization (assumes ReLU)

- Forward pass: want the variance of hidden unit activations in layer $k+1$ to be the same as variance of activations in layer k :

$$\sigma_{\Omega}^2 = \frac{2}{D_h} \quad \leftarrow \text{Number of units at layer } k$$

- Backward pass: want the variance of gradients at layer k to be the same as variance of gradient in layer $k+1$:

$$\sigma_{\Omega}^2 = \frac{2}{D_{h'}} \quad \leftarrow \text{Number of units at layer } k+1$$

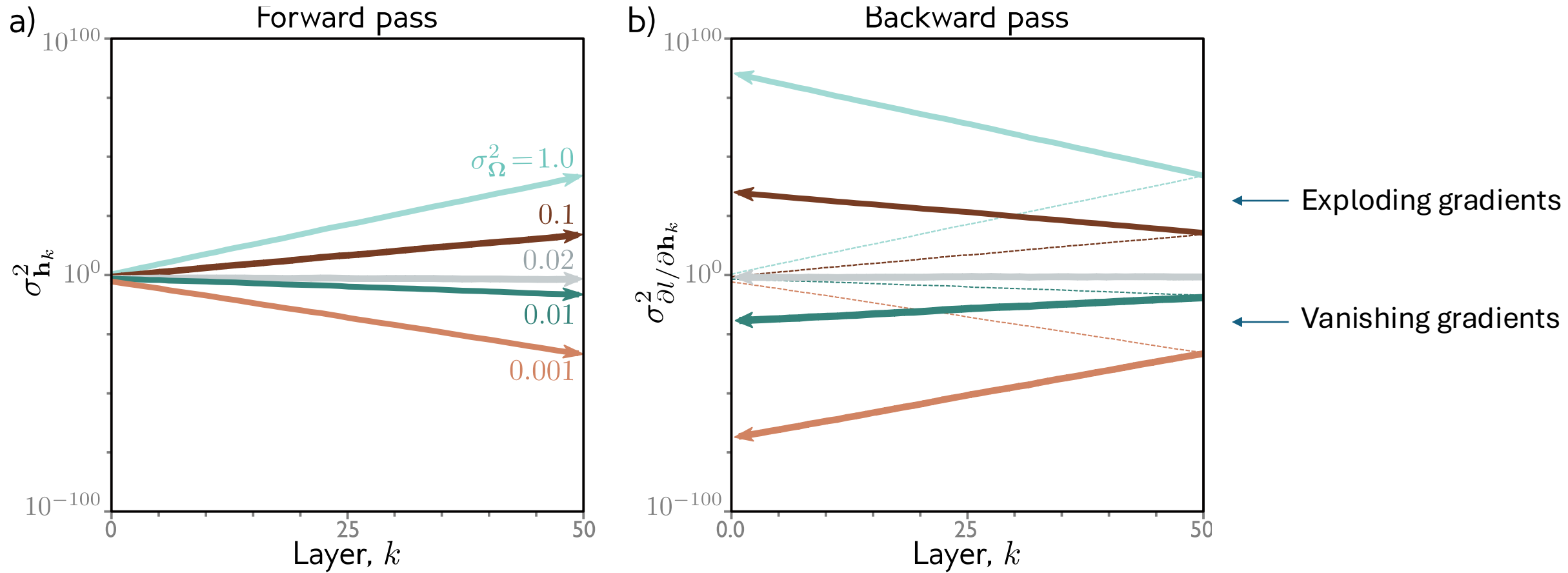


Figure 7.4 Weight initialization. Consider a deep network with 50 hidden layers and $D_h = 100$ hidden units per layer. The network has a 100 dimensional input \mathbf{x} initialized with values from a standard normal distribution, a single output fixed at $y = 0$, and a least squares loss function. The bias vectors β_k are initialized to zero and the weight matrices Ω_k are initialized with a normal distribution with mean zero and five different variances $\sigma_{\Omega}^2 \in \{0.001, 0.01, 0.02, 0.1, 1.0\}$. a)

$$\sigma_{\Omega}^2 = \frac{2}{D_h} = \frac{2}{100} = 0.02$$

Default Initialization in PyTorch

https://pytorch.org/docs/stable/nn.init.html#torch.nn.init.kaiming_uniform_

```
torch.nn.init.kaiming_uniform_(tensor, a=0, mode='fan_in', nonlinearity='leaky_relu',  
generator=None) [SOURCE]
```

Fill the input *Tensor* with values using a Kaiming uniform distribution.

The method is described in *Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification* - He, K. et al. (2015). The resulting tensor will have values sampled from $\mathcal{U}(-\text{bound}, \text{bound})$ where

$$\text{bound} = \text{gain} \times \sqrt{\frac{3}{\text{fan_mode}}}$$

Also known as He initialization.

Any Questions?



- The need for weights initialization
- Expectations Refresher
- The (Kaiming) He initialization
- Lottery tickets

Initialization Note

A good initialization does not prevent gradient descent from changing the weights a lot.

- A good initialization keeps the initial gradients modestly sized,
- And modest gradients reduce wild swings in parameters with gradient descent
- Smaller learning rates also help with this.
- Next week's topic, regularization, will directly address this.

Limitations of Initialization

- No guarantees that the model will train to low losses
- No guarantees that training process won't lead to large values or gradients
- No guarantees that the model won't have lots of inactive units
 - In fact, the estimates adjusted for half being inactive!
- In fact, much of the network is often useless, and could be pruned away!

The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks

Neural network pruning techniques can **reduce the parameter counts** of trained networks **by over 90%**, decreasing storage requirements and improving computational performance of inference **without compromising accuracy**. However, contemporary experience is that the sparse architectures produced by pruning are difficult to train from the start, which would similarly improve training performance.

We find that a standard pruning technique naturally uncovers subnetworks whose initializations made them capable of training effectively. Based on these results, we articulate the "lottery ticket hypothesis:" **dense, randomly-initialized, feed-forward networks contain subnetworks** ("winning tickets") that - **when trained in isolation - reach test accuracy comparable to the original network in a similar number of iterations**. The winning tickets we find have won the initialization lottery: their connections have initial weights that make training particularly effective.

We present an algorithm to identify winning tickets and a series of experiments that support the lottery ticket hypothesis and the importance of these fortuitous initializations. **We consistently find winning tickets that are less than 10-20% of the size of several fully-connected and convolutional feed-forward architectures** for MNIST and CIFAR10. **Above this size, the winning tickets that we find learn faster than the original network and reach higher test accuracy.**

Any Questions?



- The need for weights initialization
- Expectations Refresher
- The (Kaiming) He initialization
- Lottery tickets

Disclaimer

- Just because variance of gradients starts the same does not mean that the variance of gradients stays the same.
- You should still check the gradients if you are having training difficulties...

Bonus Tip

- If you are trying to implement a model based on a paper, and you are having trouble training, check if they shared their code.
 - Many papers omit important initialization details.



- Especially if they say that their method is not sensitive to initialization.



- Also, some paper descriptions of initialization don't match their code.

