M-seq Initialization: Using Pseudo-Random Binary Sequences to Initialize Deep Neural Networks.

Zanya Gonzalez Tellez ZANYA.GONZALEZTELLEZ@TUE.ELLIS.EU
University of Stuttgart, ELLIS Institute Tübingen, Germany

Antonio Orvieto ANTONIO@TUE.ELLIS.EU
ELLIS Institute Tübingen, MPI-IS
Tübingen AI Center, Germany

Abstract

Random initialization of deep feedforward networks can cause vanishing and exploding gradients. As the network depth increases, adapting the weights standard deviation at initialization mitigates but does not solve this issue. This problem has led to the introduction of several architectural modifications, notably residual connections and normalization layers. In this work, we return to the original problem of poor statistical signal propagation in MLPs and propose an alternative that stabilizes both the forward and backward passes at arbitrary depths. Our approach is similar to orthogonal initialization, yet it is cheaper to implement and based on maximum length sequences (M-seq): pseudo-random binary sequences generated by a linear-feedback shift register.

1. Introduction

Modern neural networks are often extremely deep, but their successful training typically depends on architectural mechanisms such as skip connections [4], normalization layers [1, 7], and carefully scaled parameterizations [9, 17]. While very deep networks are now routinely trained in practice, this success should not be confused with a complete understanding of how to train long vanilla feedforward networks: training those architectures remains challenging due to the high-dimensional non-convex parameter space with saddle points [8]. Among the factors influencing optimization, the proper initialization of the network weights is crucial to achieve convergence.

Standard practice consists of randomly initializing the weights with variance adjustments, such as those proposed by Glorot and Bengio [2] for linear activations and He et al. [5] for ReLU activations, to avoid the vanishing of gradients. Nevertheless, substantial research ([10], [16], [15]) has shown that standard i.i.d initialization prevents efficient convergence due to poor signal propagation. Theoretical analysis by Pennington et al. [12] has further demonstrated that DNNs achieving *dynamical isometry*, reached when all singular values of the input-output Jacobian are 1 at initialization, perform notably better than those initialized from standard Gaussian noise.

Multiple efforts have been made to ensure dynamical isometry, ranging from architectural solutions such as deep residual nets [4], to designing orthogonal weight matrices [14] [6]. In this work, we propose an alternative method to random initialization consisting of employing structured pseudo-random sequences to improve signal propagation in DNNs.

© Z.G. Tellez & A. Orvieto.

2. Preliminaries

Consider an L-layer multilayer perceptron (MLP) with weight matrices $\mathbf{W}_\ell \in \mathbb{R}^{d\times d}$, where $\ell = 1, ..., L$ and d is the width. Given an input vector $\mathbf{x}_i \in \mathbb{R}^{d_{in}}$ from the training data $\{\mathbf{x}_k, \mathbf{z}_k\}_{k=1}^n$, the output $\mathbf{y}_{out}(\mathbf{x}_i) \in \mathbb{R}^d$ of the MLP is given by

$$\mathbf{y}_{out}(\mathbf{x}_i) = \mathbf{W}_{L:1}^{\phi}\mathbf{W}_0\mathbf{x}_i, \qquad \mathbf{W}_{L:1}^{\phi} = \mathbf{W}_L\mathbf{D}_L\mathbf{W}_{L-1} \dots \mathbf{W}_2\mathbf{D}_2\mathbf{W}_1\mathbf{D}_1 \tag{1}$$

where $\mathbf{D}_\ell$ is the diagonal matrix of activation gates, ϕ is the activation function and $\mathbf{W}_0 \in \mathbb{R}^{d\times d_{in}}$ is the input projection matrix. In this work, we focus on linear networks where ϕ is the linear map, which simplifies $\mathbf{D}_\ell = \mathbf{I}$. We aim to minimize the L2 loss $\mathcal{L}(\mathbf{W}) = \frac{1}{2n}\sum_{i=1}^n ||\mathbf{z}_i - \mathbf{y}_{out}(\mathbf{x}_i)||^2$

Let $\{\mathbf{W}_\ell(0)\}_{\ell=1}^{L-1}$ be the set of initial weight matrices. The standard initialization method consists of i.i.d sampling the entries of each matrix from a gaussian distribution $\mathcal{N}(0, \sigma^2)$, with mean $\mu = 0$ and variance σ^2. For a linear activation function, Glorot and Bengio [2] showed that to maintain the variance of activations and gradients, the weights should be initialized with $\sigma^2 = 1/d$. This method is known as Xavier initialization.

3. Methodology

3.1. Maximum Length Sequences

A *maximum length sequence* (m-sequence) is a pseudo-random binary sequence generated by a linear-feedback shift register (LFSR). As described by Golomb and Gong [3], an LFSR is a circuit consisting of N cells, each one containing one bit $S_j \in GF(2) = \{0, 1\}$. Here, $(GF(2), +, \cdot)$ is the finite field modulo 2, which contains 2 elements. LFSRs are regulated by an internal clock which, at each time step, shifts the bits from one cell into another and changes the *initial state* of the shift register. The state of an LFSR at time step t is denoted as $\mathbf{S}^t = (S_{N-1}^t, S_{N-2}^t, ..., S_0^t)$, with its output defined as $a_t = S_{N-1}^t$. To build an LFSR, we must include a feedback loop which computes the next state $\mathbf{S}^{t+1}$ based on the other bits in the circuit according to the feedback function $f(S_{N-1}^t, S_{N-2}^t, ..., S_0^t)$. From their outputs at different time steps, LFSRs generate the sequences $\{a_t\}$. One of the main properties of LFSRs is that, with a properly-chosen feedback function, they will cycle through all possible $2^N - 1$ nonzero configurations of state $\mathbf{S}$. An example for $N = 3$ is

$$(1, 0, 0) \to (0, 1, 1) \to (1, 1, 0) \to (1, 1, 1) \to (1, 0, 1) \to (0, 0, 1) \to (0, 1, 0) \to (1, 0, 0).$$

This paper utilizes the Galois configuration of an LFSR, where given the output $a_{t-1} = S_{N-1}^{t-1}$, the bits in the subsequent state will be given by

$$S_j^t = S_{j-1}^{t-1} \; + \; a_{t-1}q_j, \qquad S_0^t = a_{t-1}q_0. \tag{2}$$

Here, $q_0, q_1, ..., q_{N-1} \in GF(2)$ are the feedback coefficients, which interact with the state bits through the modulo 2 operation. In this representation, the state bits at a given time step are altered according to the feedback function f through which the feedback coefficients q are determined. The diagram in Figure 1 depicts a Galois LFSR. In the representation used, the bits shift from right to left. An example is provided in Appendix A.1.

Mathematically, state changes in Galois LFSRs can be understood as multiplication by x in the quotient ring $GF(2)[x]/<p(x)>$, where $GF(2)[x]$ is the polynomial ring with coefficients in

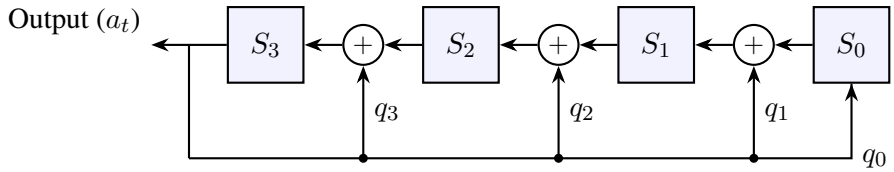

Figure 1: Galois LFSR.

$GF(2)$, and the polynomial $p(x) = \sum_{i=0}^{N-1} q_i x^i$ represents the feedback function f through which the state is updated each time step and is known as the *characteristic polynomial* of the LFSR. The period of the output sequence $\{a_t\}$ generated by the Galois LFSR depends on the features of this polynomial. In particular, if $p(x)$ is a primitive polynomial, the sequence has period $2^N - 1$ and is a maximum length sequence.

3.2. M-Seq Initialization

Motivated by the special properties of the m-sequences, we propose *M-Seq Initialization*, an alternative method to initialize the weights of MLPs of arbitrary depth. Let $\{a_t\}$ be the m-sequence of length $m = 2^N - 1$, with N the degree of the characteristic polynomial of the LFSR that generates the signal. To properly define the weights, we restrict the width of all hidden layers d to be equal to the sequence length. An initial weight matrix $\mathbf{W}_\ell(0)$ is constructed following Algorithm 1. As described, an m-sequence is generated from a random primitive polynomial at a random initial state S^0, and appended as the first row of the weight matrix. By incrementing the initial state for each row, we produce cyclic shifts of the base sequence. We map the binary sequence to a bipolar representation $\{0, 1\} \rightarrow \{-1, +1\}$, to ensure zero-mean properties.

Algorithm 1: M-seq Initialization

Input: width m, list of primitive polynomials of degree N $[p(x) = \sum_{i=0}^{N} q_i x^i]$
Return: $\mathbf{W}_\ell(0) \in \mathbb{R}^{m \times m}$
$p \leftarrow$ random element from $\{p(x)\}$;
$z \leftarrow$ random integer between 1 and m for the initial state;
for *i* ***in*** *m* **do**
  $a \leftarrow$ sequence generated from p and z;
  $a \leftarrow 2a - 1$;
  row[i] of $\mathbf{W}_\ell(0) \leftarrow \frac{a}{\sqrt{m+1}}$;
  $z \leftarrow z + 1$
end

The proposed method generates a row-permuted back-circulant matrix $\mathbf{W} = \mathbf{PC}$ for each weight matrix at initialization. Each row in a back-circulant matrix is a cyclic left shift of the row above. Therefore, the entries of the row-permuted back-circulant matrices obtained through M-seq initialization can be characterized as:

$$W_{ij} = C_{\tau_i j} = a_{(\tau_i + j) \bmod m} \tag{3}$$

where $\{\tau_0, \tau_1, \ldots, \tau_{m-1}\}$ is a permutation of $\{0, 1, \ldots, m-1\}$ determined by the evolution of the LFSR state, with each τ_i corresponding to a distinct cyclic left shift of the m-sequence.

3.3. M-seq Initialization leads to near-orthogonal behavior in linear networks

To preserve the stability of the forward pass in a linear MLP, we need to ensure that for any $\mathbf{x} \in \mathbb{R}^{d_{in}}$

$$\mathbb{E}[||\mathbf{y}_{out}(\mathbf{x})||^2] = \mathbb{E}[\mathbf{x}^\top \mathbf{W}_0^\top \mathbf{W}_1^\top(0) ... \mathbf{W}_L^\top(0)\ \mathbf{W}_L(0) ... \mathbf{W}_1(0)\mathbf{W}_0\mathbf{x}] \approx \mathbb{E}[||\mathbf{W}_0\mathbf{x}||^2]. \tag{4}$$

This depends on the matrix product $\mathbf{W}_\ell^\top(0)\mathbf{W}_\ell(0)$. For our row-permuted back-circulant matrices built through the method here proposed, the product is characterized by

$$(\mathbf{W}_\ell^\top(0)\mathbf{W}_\ell(0))_{jk} = (\mathbf{C}_\ell^\top \mathbf{P}_\ell^\top \mathbf{P}_\ell \mathbf{C}_\ell)_{jk} = (\mathbf{C}_\ell^\top \mathbf{C}_\ell)_{jk} = \sum_{l=0}^{m-1} a_{(l+j) mod\ m}\ a_{(l+k) mod\ m} \tag{5}$$

Since the rows of the weight matrices are m-sequences, they follow specific correlation properties. For instance, the auto-correlation between the output of a maximum-length LFSR $\{a_t\}$ and any shifted version of it $\{a_{t+\Delta t}\}$ is

$$R(\Delta t) = \sum_{t=0}^{m-1} a_t a_{t+\Delta t} = \begin{cases} m, & \text{if } \Delta t = 0 \\ -1, & \text{if } \Delta t \neq 0 \end{cases} \tag{6}$$

Consequently, we find that setting $t = (l+j) \bmod m$ the product yields $\mathbf{W}_\ell^\top(0)\mathbf{W}_\ell(0) = (m+1)\mathbf{I} - \mathbf{J}$ where $\mathbf{J} = \mathbf{1}\mathbf{1}^\top \in \mathbb{R}^{m \times m}$ is the all ones matrix. From this relation it follows that scaling the constructed initial weight matrices by $1/\sqrt{m+1}$ leads to

$$\frac{\mathbf{W}_\ell^\top(0)}{\sqrt{m+1}} \frac{\mathbf{W}_\ell(0)}{\sqrt{m+1}} = \mathbf{I} - \frac{\mathbf{J}}{m+1} \tag{7}$$

We observe that, since the rows and columns of the unscaled weight matrices are m-sequences mapped to the bipolar representation, the sum of the entries of any row i will be $\sum_{j=0}^{m-1} a_{(\tau_i+j) \bmod m} = 1$. This result comes from the balance property of the m-sequences, which ensures that in a sequence of length $m = 2^N - 1$ there are 2^{N-1} bits with value $+1$ and $2^{N-1} - 1$ bits with value -1. The same result holds for the columns of the matrices since they are also m-sequences. Then, the product between any of the constructed weight matrices $\mathbf{W}_\ell(0)$ and $\mathbf{J}$ is

$$(\mathbf{J}\mathbf{W}_\ell(0))_{ij} = \sum_{k=0}^{m-1} a_{(j+k) \bmod m} = 1 \tag{8}$$

Therefore, $\mathbf{J}\mathbf{W}_\ell(0) = \mathbf{W}_\ell^\top(0)\mathbf{J} = \mathbf{J}$, and for scaled matrices the following result holds:

$$\frac{\mathbf{W}_\ell^\top(0)}{\sqrt{m+1}} \mathbf{J} \frac{\mathbf{W}_\ell(0)}{\sqrt{m+1}} = \frac{\mathbf{J}}{(m+1)} \tag{9}$$

We can now employ the results derived to analyze the product of multiple scaled weight matrices.

Proposition 1 *Let $\tilde{\boldsymbol{W}}_1, ...\tilde{\boldsymbol{W}}_L$ be scaled weight matrices constructed using the M-seq initialization method. The product $G_L := \tilde{\boldsymbol{W}}_1^\top ...\tilde{\boldsymbol{W}}_L^\top \tilde{\boldsymbol{W}}_L ...\tilde{\boldsymbol{W}}_1$ simplifies to $G_L = \boldsymbol{I} - \zeta_L \boldsymbol{J}$, where*

$$\zeta_L = \frac{1}{m}\left(1 - \frac{1}{(m+1)^L}\right) \tag{10}$$

The proof is provided in Appendix A.2. Finally, we show that for the proposed initialization method the forward pass is stabilized.

Theorem 2 *Consider an MLP with initial weight matrices $\boldsymbol{W}_\ell(0)$ constructed through the M-seq Initialization algorithm. Let $\boldsymbol{W}_0$ have i.i.d entries sampled from a distribution with zero mean and variance $\sigma^2 = 1/m$, and let $x \in \mathbb{R}^{d_{in}}$ be a fixed input vector. For large L and as $m \to \infty$,*

$$\mathbb{E}[||\boldsymbol{y}_{out}(\boldsymbol{x})||^2] \approx ||\boldsymbol{x}||^2 \tag{11}$$

Proof *See Appendix A.3* ■

Saxe et al. [14] showed that using random orthogonal matrices ($\mathbf{W}^\top \mathbf{W} = \mathbf{I}$) to initialize the weights of a deep linear neural network ensures that eq.(4) holds, and leads to stable gradient propagation and convergence. Here, we have shown that using m-sequences to build the initial weight matrices leads to the asymptotic preservation of the forward signal. We empirically show in Section 4 that the proposed method achieves results consistent with dynamical isometry, matching the performance of orthogonal matrices.

4. Experiments

In this section, we empirically evaluate M-seq initialization and show that the proposed method leads to the stability and efficient convergence of a deep neural network. All the experiments were conducted in PyTorch [11] and the m-sequences employed were generated through the open source library *libgf2* [13].

We train a linear MLP of width $m = 31$ and depths $L \in \{128, 256\}$ on the Fashion MNIST dataset [18]. We compare our method against standard Xavier initialization and orthogonal initialization. All the networks are trained using Stochastic Gradient Descent (SGD), for 25 epochs and over the set of learning rates 10^{-2}, 10^{-3}, 10^{-4} (see Appendix B.1).

Figure 2 shows that M-seq initialization leads to similar training accuracy and loss as a network initialized using orthogonal weights. Both methods achieve ~ 0.8 accuracy for the depths analyzed, while Xavier initialization achieves at most ~ 0.37 for $L = 128$, degrading to ~ 0.18 for $L = 256$, indicating a failure to propagate signals. Furthermore, gradient norms for M-seq remain stable, mimicking the behavior of orthogonal weights. These results suggest that the additional perturbation introduced by the m-sequences does not impede achieving dynamical isometry and presents M-seq initialization as an alternative to orthogonal initialization.

	$L = 128$			$L = 256$		
Method	$LR = 10^{-2}$	$LR = 10^{-3}$	$LR = 10^{-4}$	$LR = 10^{-2}$	$LR = 10^{-3}$	$LR = 10^{-4}$
M-seq	0.803 ± 0.002	0.746 ± 0.004	0.65 ± 0.01	0.802 ± 0.001	0.75 ± 0.01	0.67 ± 0.01
Orthogonal	0.802 ± 0.001	0.747 ± 0.004	0.65 ± 0.01	0.802 ± 0.002	0.75 ± 0.01	0.67 ± 0.01
Xavier	0.374 ± 0.040	0.248 ± 0.085	0.17 ± 0.04	0.180 ± 0.044	0.14 ± 0.06	0.13 ± 0.03

Table 1: Training accuracy for a linear MLP with width 31 and depth 128 or 256, for different initialization methods on the Fashion MNIST dataset. All methods were evaluated for the learning rates (*LR*) $10^{-2}, 10^{-3}, 10^{-4}$. For $LR \geq 10^{-1}$ the model diverges.

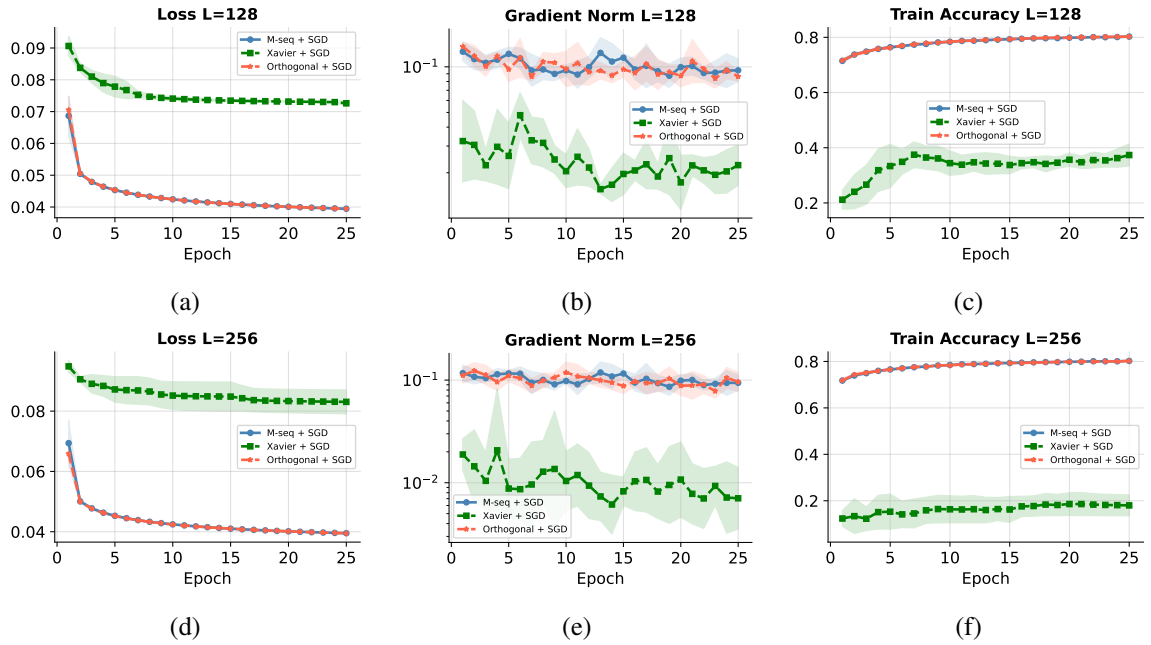

Figure 2: Fashion MNIST on a linear MLP with 31 hidden units and 128 (top) or 256 (bottom) layers. Note the similar performance between a MLP initialized with m-sequences and one initialized with orthogonal weights. The learning rate was set to 10^{-2}, which achieved the best training accuracy among all three methods. For other learning rates see B.1. Average and deviation taken over 5 random seeds.

5. Conclusion

In this work, we proposed a novel initialization method based on the correlation properties of pseudo-random bit sequences generated via LFSRs. We showed that, due to their two-level autocorrelation, using m-sequences leads to constructing weight matrices that behave as orthogonal matrices with an additional perturbation. In addition, our empirical results confirm that through the proposed initialization we can achieve optimization dynamics consistent with dynamical isometry, even in narrow networks.

ACKNOWLEDGMENTS

Antonio Orvieto acknowledges the financial support of the Hector Foundation.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, March 2010. URL https://proceedings.mlr.press/v9/glorot10a.html.

[3] Solomon W. Golomb and Guang Gong. *Feedback Shift Register Sequences*, page 81–116. Cambridge University Press, 2005.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. URL http://arxiv.org/abs/1512.03385. arXiv:1512.03385 [cs].

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, Santiago, Chile, December 2015. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.123. URL http://ieeexplore.ieee.org/document/7410480/.

[6] Wei Hu, Lechao Xiao, and Jeffrey Pennington. Provable Benefit of Orthogonal Initialization in Optimizing Deep Linear Networks, January 2020. URL http://arxiv.org/abs/2001.05992. arXiv:2001.05992 [cs].

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[8] Kenji Kawaguchi. Deep learning without poor local minima. *Advances in neural information processing systems*, 29, 2016.

[9] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *Advances in Neural Information Processing Systems*, 35:27198–27211, 2022.

[10] Antonio Orvieto, Jonas Kohler, Dario Pavllo, Thomas Hofmann, and Aurelien Lucchi. Vanishing curvature in randomly initialized deep relu networks. In *International Conference on Artificial Intelligence and Statistics*, pages 7942–7975. PMLR, 2022.

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[12] Jeffrey Pennington, Samuel S. Schoenholz, and Surya Ganguli. The Emergence of Spectral Universality in Deep Networks, February 2018. URL http://arxiv.org/abs/1802.09979. arXiv:1802.09979 [stat].

[13] Jason Sachs. libgf2: A python library for computations in gf(2), 2013-2017. URL https://foss.heptapod.net/math/libgf2. Apache License 2.0.

[14] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, February 2014. URL http://arxiv.org/abs/1312.6120. arXiv:1312.6120 [cs].

[15] Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep Information Propagation, April 2017. URL http://arxiv.org/abs/1611.01232. arXiv:1611.01232 [stat].

[16] Ohad Shamir. Exponential convergence time of gradient descent for one-dimensional deep linear neural networks. In *Conference on Learning Theory*, pages 2691–2713. PMLR, 2019.

[17] Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. *Advances in Neural Information Processing Systems*, 38:163104–163136, 2026.

[18] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Appendix A.

A.1. Example of M-seq construction

An LFSR has N binary cells. At time t, its *state* is the vector of current bits

$$S^t = (S^t_{N-1}, \ldots, S^t_1, S^t_0) \in 0, 1^N.$$

The *output* associated with this state is one selected bit of the state. In the convention used here, the output is the leftmost bit:

$$a_t = S^t_{N-1}.$$

The next state S^{t+1} is obtained by shifting the bits and feeding the output bit back into the register through modulo-2 additions. The feedback coefficients are determined by a polynomial. If the polynomial is primitive, then starting from any nonzero state, the LFSR visits all $2^N - 1$ nonzero states before repeating.

Take $N = 3$, so $m = 2^3 - 1 = 7$, and use the primitive polynomial

$$p(x) = x^3 + x + 1.$$

For a state $S^t = (S^t_2, S^t_1, S^t_0)$, the output is

$$a_t = S^t_2,$$

and the update, modulo 2, is

$$(S_2^t, S_1^t, S_0^t) \mapsto (S_1^t, ; S_0^t + S_2^t, ; S_2^t).$$

Starting from $S^0 = (1, 0, 0)$, this gives

$$(1,0,0) \to (0,1,1) \to (1,1,0) \to (1,1,1) \to (1,0,1) \to (0,0,1) \to (0,1,0) \to (1,0,0).$$

The output at each step is the leftmost bit of the current state, so the generated binary sequence is

$$a = (1, 0, 1, 1, 1, 0, 0).$$

Mapping $0, 1$ to $-1, +1$ gives

$$\tilde{a} = 2a - 1 = (1, -1, 1, 1, 1, -1, -1).$$

Now define W by cyclic shifts of $\tilde{a}$:

$$W = \frac{1}{\sqrt{8}} \begin{pmatrix} 1 & -1 & 1 & 1 & 1 & -1 & -1 \\ -1 & 1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ -1 & -1 & 1 & -1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 & 1 & 1 & -1 \end{pmatrix}.$$

The key m-sequence property is

$$\sum_{t=0}^{6} \tilde{a}_t \tilde{a}_{t+\Delta} = \begin{cases} 7, & \Delta = 0, \\ -1, & \Delta \neq 0, \end{cases}$$

where indices are modulo 7. Therefore

$$W^\top W I - \frac{1}{8} J,$$

where $J = \mathbf{1}\mathbf{1}^\top$. Hence W is almost orthogonal: it preserves all directions orthogonal to $\mathbf{1}$ exactly, and only perturbs the constant direction.

A.2. Proof Proposition 1

Proof By induction on L, it is clear that the base case for $L = 1$ holds from eq.(7). Now, assuming $G_{L-1} = \mathbf{I} - \zeta_{L-1}\mathbf{J}$, the product G_L over layers 1,...,L is:

$$\begin{aligned} G_L &= \tilde{\mathbf{W}}_1^\top ... \tilde{\mathbf{W}}_L^\top \tilde{\mathbf{W}}_L ... \tilde{\mathbf{W}}_1 \\ &= \tilde{\mathbf{W}}_1^\top ... \tilde{\mathbf{W}}_{L-1}^\top \left(\mathbf{I} - \frac{\mathbf{J}}{(m+1)} \right) \tilde{\mathbf{W}}_{L-1} ... \tilde{\mathbf{W}}_1 \\ &= \tilde{\mathbf{W}}_1^\top ... \tilde{\mathbf{W}}_{L-1}^\top \tilde{\mathbf{W}}_{L-1} ... \tilde{\mathbf{W}}_1 - \frac{\tilde{\mathbf{W}}_1^\top ... \tilde{\mathbf{W}}_{L-1}^\top \mathbf{J} \tilde{\mathbf{W}}_{L-1} ... \tilde{\mathbf{W}}_1}{(m+1)} \end{aligned} \tag{12}$$

From eq.(9) we know that $\tilde{\mathbf{W}}_\ell^\top \mathbf{J}\tilde{\mathbf{W}}_\ell = \mathbf{J}/(m+1)$, and therefore:

$$\begin{aligned} G_L &= \mathbf{I} - \zeta_{L-1}\mathbf{J} - \frac{\mathbf{J}}{(m+1)^L} \\ &= \mathbf{I} - \frac{\mathbf{J}}{m} + \frac{\mathbf{J}}{m(m+1)^{L-1}} - \frac{\mathbf{J}}{(m+1)^L} \\ &= \mathbf{I} - \mathbf{J}\frac{1}{m}\left(1 - \frac{1}{(m+1)^L}\right) \\ &= \mathbf{I} - \zeta_L \mathbf{J} \end{aligned} \tag{13}$$

■

A.3. Proof Theorem 2

Proof Using proposition 1, we obtain

$$\begin{aligned} \mathbb{E}[||\mathbf{y}_{out}(\mathbf{x})||^2] &= \mathbb{E}[\mathbf{x}^\top \mathbf{W}_0^\top \left(\mathbf{I} - \zeta_L \mathbf{J}\right) \mathbf{W}_0 \mathbf{x}] \\ &= \mathbb{E}[||\mathbf{W}_0\mathbf{x}||^2 - \zeta_L(\mathbf{x}^\top \mathbf{W}_0^\top \mathbf{1}\mathbf{1}^\top \mathbf{W}_0\mathbf{x})] \\ &= \mathbb{E}[||\mathbf{W}_0\mathbf{x}||^2] - \mathbb{E}[\zeta_L(\mathbf{1}^\top \mathbf{W}_0\mathbf{x})^2] \end{aligned} \tag{14}$$

Considering that $\mathbf{W}_0$ has entries sampled i.i.d from a normal distribution $\mathcal{N}(0, \sigma^2)$ and $\mathbf{x}$ is a fixed vector, we obtain

$$\mathbb{E}[||\mathbf{y}_{out}(\mathbf{x})||^2] = m\sigma^2||\mathbf{x}||^2 - \zeta_L \mathbb{E}[(\mathbf{1}^\top \mathbf{W}_0\mathbf{x})^2] \tag{15}$$

We observe that the product $\mathbf{1}^\top \mathbf{W}_0\mathbf{x} = \sum_{i=0}^{m-1}(\mathbf{W}_0\mathbf{x})_i$ is the sum of m zero-mean random variables with variance $\sigma^2||\mathbf{x}||^2$, therefore:

$$\begin{aligned} \mathbb{E}[||\mathbf{y}_{out}(\mathbf{x})||^2] &= m\sigma^2||\mathbf{x}||^2 - \zeta_L m\sigma^2||\mathbf{x}||^2 \\ &= m\sigma^2||\mathbf{x}||^2 - \left(1 - \frac{1}{(m+1)^L}\right)\sigma^2||\mathbf{x}||^2 \\ &= m\sigma^2||\mathbf{x}||^2 - \sigma^2||\mathbf{x}||^2 + \frac{\sigma^2||\mathbf{x}||^2}{(m+1)^L} \end{aligned} \tag{16}$$

The third term on the right hand side of the equation scales with order $O((m+1)^{-L})$, and can be neglected when analyzing deep networks with large L. The second term has order $O(1)$ and, by choosing $\sigma^2 = 1/m$ as the variance of the projection matrix $\mathbf{W}_0$, its effect decreases for large m. This leads to $\mathbb{E}[||\mathbf{y}_{out}(\mathbf{x})||^2] \approx ||\mathbf{x}||^2$, which confirms that the proposed initialization asymptotically preserves the norm of the forward signal.

■

Appendix B.

B.1. Additional Results

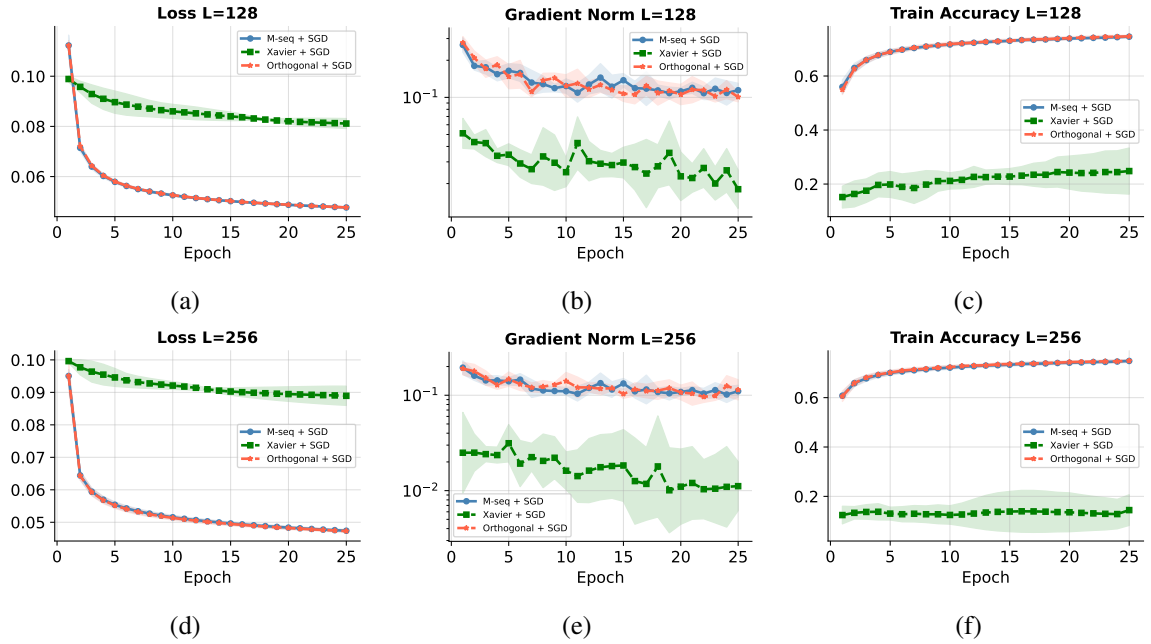

Figure 3: Fashion MNIST on a linear MLP with 31 hidden units and 128 (top) or 256 (bottom) layers. Note the similar performance between an MLP initialized with m-sequences and one initialized with orthogonal weights. The learning rate is set to 10^{-3}. Average and deviation taken over 5 random seeds.

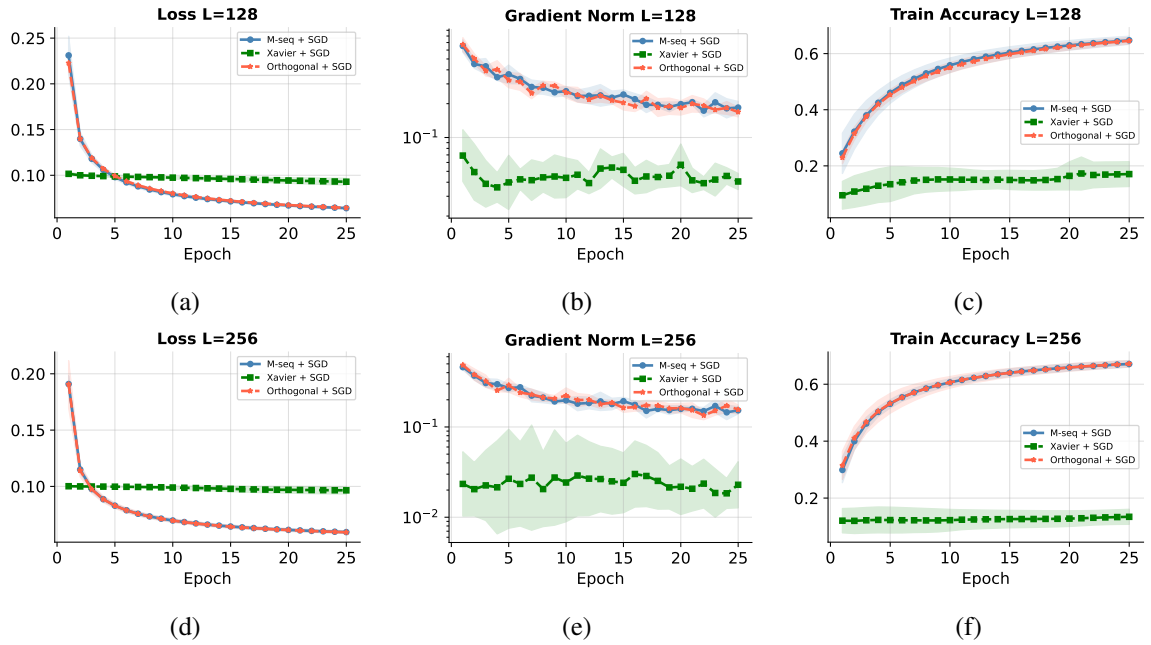

Figure 4: Fashion MNIST on a linear MLP with 31 hidden units and 128 (top) or 256 (bottom) layers. Note the similar performance between an MLP initialized with m-sequences and one initialized with orthogonal weights. The learning rate is set to 10^{-4}. Average and deviation taken over 5 random seeds.