Multiplicative Fourier Level of Detail
Supplemental Material

Yishun Dou[2] Zhong Zheng[2] Qiaoqiao Jin[1] Bingbing Ni[1,2]
[1]Shanghai Jiao Tong University, Shanghai 200240, China [2] Huawei
yishun.dou@gmail.com nibingbing@sjtu.edu.cn

1. Supplemental Derivations

1.1. Deriving the Formulation of MFLOD

Lemma 1 *The product of two sines can be transformed to the sum of two sines:*

$$\sin(\tau_1 z_1 + \varphi_1) \circ \sin(\tau_2 z_2 + \varphi_2) = \frac{1}{2}[\sin(\underline{\tau_1 z_1 + \tau_2 z_2} + \varphi_1 + \varphi_2 - \frac{\pi}{2}) + \sin(\underline{\tau_1 z_1 - \tau_2 z_2} + \varphi_1 - \varphi_2 + \frac{\pi}{2})]. \tag{1}$$

Theorem 2 *The entire function of MFLOD at level ℓ can be characterized as:*

$$\sum_{j=0}^{N^{\ell}_{sine}-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}^j_1 \boldsymbol{z}_1 + \bar{\omega}^j_2 \boldsymbol{z}_2 + \cdots + \bar{\omega}^j_\ell \boldsymbol{z}_\ell} + \bar{\phi}^j), \tag{2}$$

where the coefficients $\bar{\alpha}^j$, $\bar{\omega}^j$ and $\bar{\phi}^j$ are determined by the parameters of filters and network.

Proof. Consider the first level $\ell = 1$, the intermediate output $\boldsymbol{t}_1$ is given as

$$\boldsymbol{t}_1 = \sin(\omega_1 \boldsymbol{z}_1 + \phi_1). \tag{3}$$

This expression is a d dimensional vector. Although there is no intermediate output layer at first level in our architecture, it still conforms to the form of Eq. (2).

Consider the MFLOD at second level $\ell = 2$, the intermediate output $\boldsymbol{t}_2$ is given as

$$\begin{aligned} \boldsymbol{t}_2 &= \sin(\omega_2 \boldsymbol{z}_2 + \phi_2) \circ [\boldsymbol{W}_2 \sin(\omega_1 \boldsymbol{z}_1 + \phi_1) + \boldsymbol{b}_2] \qquad (4)\\ &= \sin(\omega_2 \boldsymbol{z}_2 + \phi_2) \circ [\boldsymbol{W}_2 \sin(\omega_1 \boldsymbol{z}_1 + \phi_1)] + \sin(\omega_2 \boldsymbol{z}_2 + \phi_2) \circ \boldsymbol{b}_2, \qquad (5) \end{aligned}$$

where

$$\begin{aligned} &\sin(\omega_2 \boldsymbol{z}_2 + \phi_2) \circ [\boldsymbol{W}_2 \sin(\omega_1 \boldsymbol{z}_1 + \phi_1)] \\ &= \begin{bmatrix} \sin(\omega_2^{(1)} \boldsymbol{z}_2 + \phi_2^{(1)}) \\ \vdots \\ \sin(\omega_2^{(d)} \boldsymbol{z}_2 + \phi_2^{(d)}) \end{bmatrix} \circ \begin{bmatrix} W_2^{(1,1)} \sin(\omega_2^{(1)} \boldsymbol{z}_1 + \phi_1^{(1)}) + \cdots + W_2^{(1,d)} \sin(\omega_2^{(d)} \boldsymbol{z}_1 + \phi_1^{(d)}) \\ \vdots \\ W_2^{(d,1)} \sin(\omega_2^{(1)} \boldsymbol{z}_1 + \phi_1^{(1)}) + \cdots + W_2^{(d,d)} \sin(\omega_2^{(d)} \boldsymbol{z}_1 + \phi_1^{(d)}) \end{bmatrix}. \end{aligned} \tag{6}$$

Since the elementwise multiplication between each row in above expression results in a sum of two sines (**Lemma**1), it results in a vector containing $2d^2$ sines, each of which consists a linear combination of modulated $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$. In addition to the bias term with d sines, the intermediate output $\boldsymbol{t}_2$ can be expressed as

$$\begin{aligned}\boldsymbol{t}_2 &= \sum_{j=0}^{2d^2-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_1^j \boldsymbol{z}_1 + \bar{\omega}_2^j \boldsymbol{z}_2} + \bar{\phi}^j) + \sum_{j=2d^2}^{2d^2+d-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_2^j \boldsymbol{z}_2} + \bar{\phi}^j) \\ &= \sum_{j=0}^{2d^2+d-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_1^j \boldsymbol{z}_1 + \bar{\omega}_2^j \boldsymbol{z}_2} + \bar{\phi}^j). \end{aligned} \tag{7}$$

Now, assume the MFLOD at ℓth level. $\sin(\omega_\ell \boldsymbol{z}_\ell + \phi_\ell)$ would be elementwise multiplied with all previous sine terms. Then we have

$$\begin{aligned}\boldsymbol{t}_\ell &= \sum_{j=0}^{2dN_{\text{sine}}^{\ell-1}-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_1^j \boldsymbol{z}_1 + \bar{\omega}_2^j \boldsymbol{z}_2 + \cdots + \bar{\omega}_\ell^j \boldsymbol{z}_\ell} + \bar{\phi}^j) + \sum_{j=2dN_{\text{sine}}^{\ell-1}}^{2dN_{\text{sine}}^{\ell-1}+d-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_\ell^j \boldsymbol{z}_\ell} + \bar{\phi}^j) \\ &= \sum_{j=0}^{N_{\text{sine}}^{\ell}-1} \bar{\alpha}^j \sin(\underline{\bar{\omega}_1^j \boldsymbol{z}_1 + \bar{\omega}_2^j \boldsymbol{z}_2 + \cdots + \bar{\omega}_\ell^j \boldsymbol{z}_\ell} + \bar{\phi}^j), \end{aligned} \tag{8}$$

which is the original result, completing the proof. (See Lindell *et al*. [4] for the proof of the number of sines N_{sine}^{ℓ}.)

1.2. Initialization and Distribution of Activations

We initialize the feature-volume entries using the uniform distribution $\mathcal{U}(-a, a)$, and therefore the variance is $a/\sqrt{3}$. Then the distribution after normalization is $\mathcal{U}(-\sqrt{3}, \sqrt{3})$ at initialization.

Theorem 3 *Let $\boldsymbol{\omega} \in \mathbb{R}^{d\times m}$, $\boldsymbol{z} \in \mathbb{R}^m$, and $\boldsymbol{\phi} \in \mathbb{R}^d$ be independent random variables sampled from continuous uniform distributions as*

$$\boldsymbol{\omega} \sim \mathcal{U}(-B, B) \tag{9}$$

$$\boldsymbol{z} \sim \mathcal{U}(-\sqrt{3}, \sqrt{3}) \tag{10}$$

$$\boldsymbol{\phi} \sim \mathcal{U}(-\pi, \pi) \tag{11}$$

where $B >> \pi$. Then let $\boldsymbol{X} = \boldsymbol{\omega}\boldsymbol{z} + \boldsymbol{\phi}$. When $m = 1$, the probability density function $f_{\boldsymbol{X}}(x)$ of $\boldsymbol{X}$ is approximately

$$f_{\boldsymbol{X}}(x) \approx \frac{1}{2\sqrt{3}B} \log\left(\frac{B}{\min(|x/\sqrt{3}|, B)}\right). \tag{12}$$

As m increases, $f_{\boldsymbol{X}}(x)$ approaches the Gaussian distribution according to the central limit theorem.

(See Lindell *et al*. [4] for the proof of Eq. (12).)

2. Neural Tangent Kernel for Hybrid Models

We compute the neural tangent kernel (NTK) [2] using the Jacobian contraction. For the global implicit neural network such as SIREN [7] and FFN [8], the NTK for two data points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is defined as the matrix product between the Jacobian of the model evaluated at $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$:

$$\Theta_\theta^f(\boldsymbol{x}_1, \boldsymbol{x}_2) = [\partial f(\theta, \boldsymbol{x}_1)/\partial\theta] \ \ [\partial f(\theta, \boldsymbol{x}_2)/\partial\theta]^T, \tag{13}$$

where $[\partial f(\theta, \cdot)/\partial\theta]$ is a neural network Jacobian. Base on the fact that the Jacobian is with respect to the neural network parameter according to the NTK theory [2], and that the parametric feature-volume that encodes the queried position into feature vectors is independent of the neural network parameter, the feature-volume does not need to be comprised in the full Jacobian. Let $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ be the encoded/queried features for the input positions $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we can simply compute the gradients with respect to the $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$:

$$\Theta_\theta^f(\boldsymbol{x}_1, \boldsymbol{x}_2; \mathcal{Z}) = [\partial f(\theta, \boldsymbol{z}_1)/\partial\theta] \ \ [\partial f(\theta, \boldsymbol{z}_2)/\partial\theta]^T, \tag{14}$$

where $\mathcal{Z}$ is the parametric feature-volume. It's worth noting that MFLOD includes learnable sinusoidal filters θ^{filter} in addition to the linear layers θ^{net}, and thus in Eq. (14) $\theta = \theta^{\text{filter}} + \theta^{\text{net}}$. In order to compute the NTK efficiently, we reimplement the MFLOD in PyTorch and use the functorch [1] which implements the fast finite width NTK [6].

3. Additional Implementation Details

We jointly train the feature-volume entries, the sinusoidal filters, and the neural network weights by applying Adam [3], where we set $\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 10^{-15}$ following [5]. To keep the learnable filter ω and ϕ distribute around the initialization, we apply a small learning rate to them. The hyperparameters bandwidth B_ℓ are set empirically. For example, given a LOD with maximum level $L = 5$, we set $B_1 = B_2 = B/8, B_3 = B_4 = B_5 = B/4$ such that $\sum_\ell B_\ell = B$, where B is the maximum bandwidth. The setting of LOD with $L = 4$ and $L = 6$ can be simply extended from that of $L = 5$. For image fitting and NeRF, we follow BACON [4] to set the maximum bandwidth B; for 3D shape representation, we use a fixed value (the same with that used for *dragon* in BACON) for all shapes since we found MFLOD is not as sensitive to B as BACON. After training, we found that the filters are distributed around the initialization, thus we can assume that the trained MFLOD has the similar bandlimited property to that at initialization. Table 1a shows comparative results on the effects of the proposed initialization and bandwidth limitation. Additionally, how the dimensions of Fourier space affect the final results is shown in Tab. 1b.

	MFLOD	no $\mathbf{W}_\ell$	no ω_ℓ, ϕ_ℓ	no bandlimit
ShapeNet-200 (IoU)	94.3	93.2	88.9	94.0
NeRF (PSNR)	33.91	33.23	29.62	33.64

(a) From left to right: (1) MFLOD. (2) We replace the initialization distribution of linear layers $\mathbf{W}_\ell$ with xavier uniform. (3) We replace the initialization distribution of filters ω_ℓ, ϕ_ℓ with xavier uniform. (4) All levels have the same bandwidth.

	$d = 16$	$d = 32$ (default)	$d = 64$	$d = 128$
ShapeNet-200 (IoU)	92.3	94.3	94.5	94.4
NeRF (PSNR)	30.87	33.91	34.01	34.06

(b) The results with different Random Fourier Features (RFF) dimension d. We choose d=32 to maximize the quality-cost tradeoff.

Table 1. Ablation studies on the proposed initialization scheme, the bandlimited initialization (a), and the Random Fourier Features (RFF) dimension d (b). We use the results on ShapeNet-200(IoU) and NeRF(PSNR) to demonstrate the effect of different design choices.

References

[1] Richard Zou Horace He. functorch: Jax-like composable function transforms for pytorch. https://github.com/pytorch/functorch, 2021. 2

[2] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018. 2

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[4] David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16252–16262, 2022. 2, 3

[5] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 3

[6] Roman Novak, Jascha Sohl-Dickstein, and Samuel S Schoenholz. Fast finite width neural tangent kernel. In *International Conference on Machine Learning*, pages 17018–17044. PMLR, 2022. 2

[7] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 2

[8] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 2