Tensor Programs II: Neural Tangent Kernel for Any Architecture

Greg Yang
Microsoft Research AI
gregyang@microsoft.com

Abstract

We prove that a randomly initialized neural network of *any architecture* has its Tangent Kernel (NTK) converge to a deterministic limit, as the network widths tend to infinity. We demonstrate how to calculate this limit. In prior literature, the heuristic study of neural network gradients often assumes every weight matrix used in forward propagation is independent from its transpose used in backpropagation [58]. This is known as the *gradient independence assumption (GIA)*. We identify a commonly satisfied condition, which we call *Simple GIA Check*, such that the NTK limit calculation based on GIA is correct. Conversely, when Simple GIA Check fails, we show GIA can result in wrong answers. Our material here presents the NTK results of Yang [63] in a friendly manner and showcases the *tensor programs* technique for understanding wide neural networks. We provide reference implementations of infinite-width NTKs of recurrent neural network, transformer, and batch normalization at https://github.com/thegregyang/NTK4A.

1 Introduction

Jacot et al. [39] showed that, in the limit of large width, a neural network undergoing training by gradient descent evolves like a linear model. Their argument proceeds in two steps:

NTKINIT If $f(x;\theta)$ is the neural network (with parameters θ and input x), then we can define a kernel called *Neural Tangent Kernel* by

$$\Theta(x,\bar{x}) \stackrel{\text{def}}{=} \langle \nabla_\theta f(x;\theta), \nabla_\theta f(\bar{x};\theta)\rangle, \quad \text{for any inputs } x, \bar{x}.$$

Jacot et al. [39] showed that, if the parameters θ are appropriately randomized, then Θ converges to a deterministic kernel $\mathring{\Theta}$ as the widths of f grow to infinity.

NTKTRAIN In the limit of large width, the NTK in the course of gradient descent stays constant, and remarkably, the network evolves like a linear model under kernel gradient descent with this limiting NTK $\mathring{\Theta}$.

With recent experimental validation [45], NTK promises to shed light on the training and generalization properties of overparametrized neural networks. Yet, it's not clear whether NTK continues to be valid for modern deep learning, such as Faster R-CNN in image segmentation [56], transformer in machine translation [59], or generative adversarial networks for distribution learning [25]. In particular, we ask

Does every modern neural network have an infinite-width NTK? Can we compute it?

Our contributions. In this paper, we show that the NTK for any randomly initialized neural network of *standard architecture*[1] converges almost surely to a deterministic limit, as the network widths[2] tend to infinity, and we show how to exactly compute this limit, i.e. we generalize NTKINIT to standard architectures. *By **standard architecture** we mean any architecture that is some composition*

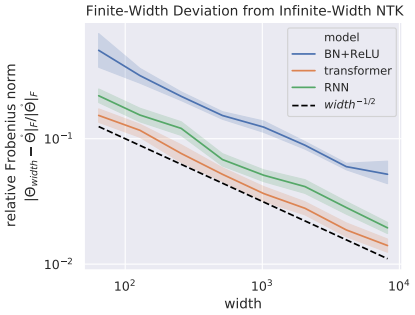
[1] In this work, architecture refers to the network topology along with the ratios of widths of hidden layers.

[2] In fully-connected network, width is the number of neurons in a layer. In a convolutional network, width is the number of channels in a layer.

of multilayer perceptrons (MLPs), recurrent neural networks (RNNs) (e.g., Long-Short Term Memory (LSTM) [33] or Gated Recurrent Unit (GRU) [14]), skip connections [31, 35], convolutions [22, 23, 42, 43, 57] or graph convolutions [11, 17, 20, 32, 40, 46], pooling [42, 43], batch normalization [38], layer normalization [8] and/or attention [9, 59]. More generally, our result applies to any architecture whose forward and backpropagation can be expressed via nonlinearities and matrix multiplication (Definition 7.1).

In the process, we identify a commonly satisfied condition (Simple GIA Check, Condition 1) that rigorously justifies what is known as the *Gradient Independence Assumption (GIA)* [67]. This is the heuristic, used in calculating the statistics of neural network gradients at initialization, that $W^\top$ in backpropagation is independent from W in forwardpropagation. However, without Condition 1, calculation based on GIA can be incorrect (Section 6.3).

We give concrete algorithms to compute the infinite-width NTK for batchnorm-ReLU MLP, transformer, and RNN (Appendix E) and verify they agree with simulations. In the plot on the right, we have computed the deviation of empirical NTKs Θ_{width} over widths $2^6, \ldots, 2^{13}$ from the corresponding limits $\mathring{\Theta}$. The shade represents 95% confidence interval of the mean over 100 seeds.



The *Tensor Programs* Series This paper is the second in the *Tensor Programs* series, following Yang [62]. Here we show NTKINIT holds for any standard architecture, which motivates NETSOR$\top$, an extension of the tensor program language NETSOR in Yang [62] by matrix transposes. Whereas NETSOR can only express the forward propagation of a neural network, NETSOR$\top$ can also express its backpropagation. This allows us to reason about the network's gradients in the infinite-width limit (and hence the NTK), which NETSOR cannot do. In a future paper, we will also generalize NTKTRAIN for any standard architecture, which requires an even more expressive extension of NETSOR$\top$. The results in this paper supercede all results in Yang [63] regarding NTK.

Our results here imply a *universal Neural Network-Tangent Kernel correspondence*. It opens a way toward studying the inductive biases of a neural network of any architecture trained under SGD. We hope this can enable theoretical understanding to catch up to practice even as neural networks manifest in increasingly many varied architectures in modern deep learning.

2 Background

Given a parametrized function $f(x;\theta)$ with parameter θ and with scalar output, we can naively expand f in θ around a base point θ_0

$$f(x;\theta) - f(x;\theta_0) \approx \langle \nabla_\theta f(x;\theta_0), \theta - \theta_0 \rangle \tag{1}$$

for any input x, where $\langle , \rangle$ denotes inner product. The RHS is a linear model, where $\nabla_\theta f(-;\theta_0)$ acts as a input featurizer, and $\theta - \theta_0$ acts as the weights. This is a good approximation as long as θ is not too far from θ_0 — in particular, if f is a neural network and we train it for a short amount of time under gradient descent with a small learning rate. However, at face value, it seems f can never change — and learn — much under such training. Why would such a naive linearization of f be helpful?

Counterintuively, Jacot et al. [39] showed that, as the network widths tend to infinity, f can in fact fit any data *perfectly* while Eq. (1) remains an accurate description of the training dynamics! An explanatory intuition is that, when θ is high dimensional, even a small change in θ can cause a large change in f.

Let's be a bit more precise. Consider the L-hidden-layer MLP $f(x;\theta)$ described below in Eq. (2), with width n^l in layer l. Then Jacot et al. [39] showed that the finite-width NTK $\Theta(x,\bar{x}) \overset{\text{def}}{=} \langle \nabla_\theta f(x;\theta), \nabla_\theta f(\bar{x};\theta) \rangle$ converges in probability

$$\Theta \xrightarrow{\text{p}} \mathring{\Theta} \quad \text{as } n^1, \ldots, n^L \to \infty \text{ in that sequence,} \tag{NTKINIT}$$

for some deterministic $\mathring{\Theta}$ to be described below (Eq. (12)), over the randomness induced by randomly initializing the parameters like $\omega^l_{\alpha\beta}, b^l_\alpha \sim \mathcal{N}(0,1), \forall \alpha, \beta$. This means that the inner product between

every pair of features $\nabla_\theta f(x;\theta_0), \nabla_\theta f(\bar{x};\theta_0)$ of Eq. (1) converges, as widths tend to infinity, even though the parameters θ are random.

Now consider the evolution of the MLP f_t with time t, trained under continuous time gradient descent with loss function $\mathcal{L}$. Let the initial function f_0 be obtained by standard Gaussian random initialization as above. Then Jacot et al. [39] showed that, in the large width limit, for any fixed training time T,

$$f_t \to \mathring{f}_t \quad \text{for all } t < T, \quad \text{where} \quad \mathring{f}_0 = f_0,\ \partial_t \mathring{f}_t = -\eta \mathring{\Theta} \cdot \nabla_f \mathcal{L}(\mathring{f}_t). \tag{NTKTRAIN}$$

Thus, somehow the hopelessly complicated optimization trajectory of an MLP has reduced to a kernel gradient descent with a fixed kernel $\mathring{\Theta}$. For square loss $\mathcal{L}$, this equation further simplifies to a linear differential equation, allowing one to solve for $\mathring{f}_t$ explicitly for all t: if labels are provided by a ground truth function f^*, then

$$\mathring{f}_t - f^* = e^{-\eta t \mathring{\Theta}}(f_0 - f^*).$$

Because one can show $\mathring{\Theta}$ is in general a non-singular kernel, this equation implies that f can fit any training data given it is wide enough [39].

Thus, the infinite-width NTK $\mathring{\Theta}$ reflects an *implicit prior* induced by gradient descent and the choices of architecture and initialization scheme. For example, its spectrum informs us the kind of functions that can be learned quickly and generalize well [65]. Jacot et al. [39] gave us a way into the blackbox of MLPs, and this paper tries to fill the gap for modern architectures. Here we describe a general, rigorous way of computing the infinite-width NTK of a network. In a future paper of this series, we will also show NTKTRAIN holds for any architecture as well. We hope our work here can enable theoretical analyses of state-of-the-art neural networks that contribute to the practice of modern machine learning.

3 Related Works

A much older literature of Gaussian process (GP) behavior of wide neural networks also associates a kernel to each network (the NN-GP correspondence) [16, 30, 41, 44, 50–52, 60]. While the NTK can be thought of as characterizing the behavior of training the full network under gradient descent, the infinite-width GP of a network characterizes the same when training only the last layer.

After Jacot et al. [39] invented NTK, our original paper [63] proved the architectural universality of NTK and NN-GP. However, the results were written densely and in heavy programming language notation. Yang [62] simplified the writing and generalized the results for GP. We do the same here for the NTK results.

After Yang [63], several works dove into specific kinds of NTKs, such as convolutional [6], graph [19], hypernetworks [48], RNNs [2], attention [34], or NTK with orthogonal initialization [37]. Other works studied the higher order terms in the Taylor expansion [21, 36], ensembled NTK [49], or finite width corrections [27, 47].

Closely related is the signal propagation literature, which tries to understand how to prevent pathological behaviors in randomly initialized neural networks when they are deep [13, 26, 28, 29, 53–55, 58, 66–68]. The investigation of *forward signal propagation* corresponds to studying the infinite-depth limit of the associated Gaussian process, and the investigation of *backward signal propagation* corresponds to studying the infinite-depth limit of NTK.

Neural tangent kernel solved an age-old question of "how does training of neural network work so well despite being highly nonconvex?" [3–5, 18, 39, 69]. This in turn has been used for studying convergence questions in deep reinforcement learning [1, 12]. The spectrum of NTK has been analyzed to provide finer-grained answers to these problems [10, 24, 65].

Compared to neural networks, kernel regression with the corresponding NTKs work better in the low data regime [7], consistent with classical observations about kernel methods and previous works on NNGPs [44, 52]. This can be valuable in important settings such as medical data that need to make decisions based on only a few data points.

4 Warmup: Neural Tangent Kernel for a Multi-Layer Perceptron

We first demonstrate the intuitions of our framework by redoing the MLP NTK limit computation. Consider the MLP $f(\xi;\theta) = W^{L+1}x^L(\xi)$ with input $\xi \in \mathbb{R}^{n^0}$ and output dimension $n^{L+1} = 1$,

where we recursively define, for $l = 2, \ldots, L$,

$$h^l(\xi) = W^l x^{l-1}(\xi) + b^l \in \mathbb{R}^{n^l}, \quad x^l(\xi) = \phi(h^l(\xi)), \quad h^1(\xi) = W^1 \xi + b^1 \in \mathbb{R}^{n^1} \tag{2}$$

in which each W^l is factorized as $W^l = \frac{1}{\sqrt{n^{l-1}}}\omega^l$, and the MLP's parameters are $\theta = \{\omega^l \in \mathbb{R}^{n^l \times n^{l-1}}\}_{l=1}^{L+1} \cup \{b^l \in \mathbb{R}^{n^l}\}_{l=1}^{L}$. This style of parametrization of weight matrices is known as the *NTK parametrization*. We shall sample $\omega^l_{\alpha\beta}, b^l_\alpha \sim \mathcal{N}(0,1), \forall \alpha, \beta$. Jacot et al. [39]'s argument for NTKINIT is inductive in the depth of the MLP, which would run into difficulty generalizing to other architectures with weight sharing, like RNNs. Here we show a different technique based on decomposing the NTK into an explicit sum of products of terms whose limits we can evaluate.

4.1 Decomposing NTK

For simplicity, write $f(\xi) = f(\xi; \theta)$ and $\nabla_p f(\xi)$ will denote gradient of the output $f(\xi)$ in some quantity p, given ξ and θ. In the MLP (Eq. (2)) above, we can decompose the NTK into contributions from weights and biases: for inputs $\xi, \bar{\xi} \in \mathbb{R}^{n^0}$ (possibly $\xi = \bar{\xi}$),

$$\Theta(\xi, \bar{\xi}) = \langle \nabla_\theta f(\xi), \nabla_\theta f(\bar{\xi}) \rangle = \sum_{l=1}^{L+1} \left\langle \nabla_{\omega^l} f(\xi), \nabla_{\omega^l} f(\bar{\xi}) \right\rangle + \sum_{l=1}^{L} \left\langle \nabla_{b^l} f(\xi), \nabla_{b^l} f(\bar{\xi}) \right\rangle, \tag{3}$$

where $\langle , \rangle$ denotes (trace) inner product. To see this quantity converges as widths $n^1, \ldots, n^L \to \infty$, it suffices to show that each summand converges. First note the $n^l \times n^{l-1}$ matrix $\nabla_{\omega^l} f(\xi)$ is the product of the $n^l \times 1$ vector $\frac{1}{\sqrt{n^{l-1}}} \nabla_{h^l} f(\xi)$ and the $1 \times n^{l-1}$ vector $x^{l-1}(\xi)^\top$, by chain rule. Abbreviate $\bullet = \bullet(\xi), \bar{\bullet} = \bullet(\bar{\xi})$ for different vectors $\bullet \in \{h^l, x^l\}_l$. Set $dh^l = \sqrt{n^l} \nabla_{h^l} f(\xi)$ and $d\bar{h}^l = \sqrt{n^l} \nabla_{\bar{h}^l} f(\bar{\xi})$. Then we have $\nabla_{\omega^l} f(\bar{\xi}) = \frac{1}{\sqrt{n^l n^{l-1}}} d\bar{h}^l \bar{x}^{l-1\top}$. Using the cyclic property of the trace inner product in the right equality,

$$\left\langle \nabla_{\omega^l} f(\xi), \nabla_{\omega^l} f(\bar{\xi}) \right\rangle = \frac{1}{n^l n^{l-1}} \left\langle dh^l x^{l-1\top}, d\bar{h}^l \bar{x}^{l-1\top} \right\rangle = \left(\frac{dh^{l\top} dh^l}{n^l} \right) \left(\frac{x^{l-1\top} \bar{x}^{l-1}}{n^{l-1}} \right). \tag{4}$$

In the rest of the section we seek to understand the two terms in this product in an intuitive way. The main ingredients in our argument are a central limit heuristic (i.e. the sum of many roughly independent random variables looks like a Gaussian) and gradient independence assumption.

4.2 Limits of Forward Quantities $x^{l\top} \bar{x}^l / n^l$

By the randomness of the initial weight matrices and inductive applications of central limit arguments, $(x^l_\alpha, \bar{x}^l_\alpha)$ is intuitively correlated but roughly iid across $\alpha \in [n^l]$ [55, 58], so

$$\frac{x^{l\top} \bar{x}^l}{n^l} \to C^l(\xi, \bar{\xi}), \tag{5}$$

for some deterministic scalar $C^l(\xi, \bar{\xi})$. Unpacking this a bit: for each α, the coordinate $(W^l x^{l-1})_\alpha = \sum_{\beta=1}^n W^l_{\alpha\beta} x^{l-1}_\beta$ is a sum of a large number n of roughly iid random variables $W^l_{\alpha\beta} x^{l-1}_\beta$. Its variance is $\mathbb{E}(W^l x^{l-1})^2_\alpha = \mathbb{E}(\sum_{\beta=1}^n W^l_{\alpha\beta} x^{l-1}_\beta)^2 = \sum_{\beta=1}^n \mathbb{E}(W^l_{\alpha\beta})^2 \, \mathbb{E}(x^{l-1}_\beta)^2 = \|x^{l-1}\|^2 / n^{l-1} \approx C^{l-1}(\xi, \xi)$. So by a central limit argument, $(W^l x^{l-1})_\alpha$ should look like $\mathcal{N}(0, C^{l-1}(\xi, \xi))$. Similarly, $(W^l \bar{x}^{l-1})_\alpha$ should be roughly $\mathcal{N}(0, C^{l-1}(\bar{\xi}, \bar{\xi}))$ and the pair $((W^l x^{l-1})_\alpha, (W^l \bar{x}^{l-1})_\alpha)$ should be jointly Gaussian with covariance $C^{l-1}(\xi, \bar{\xi})$. Then the pair $(x^l_\alpha, \bar{x}^l_\alpha)$ should be distributed like $(\phi(\zeta), \phi(\bar{\zeta}))$, and C^l satisfies the following recursion (here the $+1$ comes from the bias $b^l \sim \mathcal{N}(0,1)$)

$$C^l(\xi, \bar{\xi}) = \mathbb{E}\, \phi(\zeta)\phi(\bar{\zeta}), \quad \text{where} \quad (\zeta, \bar{\zeta}) \sim \mathcal{N}\left(0, \begin{pmatrix} C^{l-1}(\xi,\xi) & C^{l-1}(\xi,\bar{\xi}) \\ C^{l-1}(\bar{\xi},\xi) & C^{l-1}(\bar{\xi},\bar{\xi}) \end{pmatrix} + 1\right). \tag{6}$$

4.3 Limits of Backward Quantities $dh^{l\top} d\bar{h}^l / n^l$

For simplicity, assume $n^1 = \cdots = n^L$. Then like h^l, we can also expand $dx^l_\alpha \overset{\text{def}}{=} (W^{l+1\top} dh^{l+1})_\alpha = (W^{l+1\top}(dx^{l+1} \odot \phi'(h^{l+1}))_\alpha = \sum_\beta W^{l+1}_{\beta\alpha} dx^{l+1}_\beta \phi'(h^{l+1}_\beta)$. We might hope to say that each term of

this sum is roughly independent so we can apply a central limit heuristic, but h_β^{l+1} actually depends on $W_{\beta\gamma}^{l+1}$ for all γ. Interestingly, the signal propagation literature [58, 61, 67, 68] has found *it's fine to ignore* such dependences: If we adopt the following

Heuristic 4.1 (gradient independence assumption, or GIA [58, 67]). *For any matrix W, we assume $W^\top$ used in backprop is independent from W used in the forward pass.*

then the resulting calculation will still agree with simulation when $n^1, \dots, n^L \gg 1$. With this assumption, we can then proceed as in Section 4.2 and argue dx_α^l is roughly distributed as $\mathcal{N}(0, \|dh^{l+1}\|^2/n^{l+1})$ and iid across $\alpha \in [n^l]$. Likewise, we argue the pair $(dx_\alpha^l, d\bar{x}_\alpha^l) \overset{\text{def}}{=} ((W^{l+1\top} dh^{l+1})_\alpha, (W^{l+1\top} d\bar{h}^{l+1})_\alpha)$ is jointly Gaussian with zero mean and covariance $\|dh^{l+1\top} d\bar{h}^{l+1}\|^2/n^{l+1}$, and is iid across α. Since $(h_\alpha^l, \bar{h}_\alpha^l)$ is also roughly iid across α, we expect $(dh_\alpha^l, d\bar{h}_\alpha^l) = (dx_\alpha^l \phi'(h_\alpha^l), d\bar{x}_\alpha^l \phi'(\bar{h}_\alpha^l))$ to be so as well, and

$$\frac{dh^{l\top} d\bar{h}^l}{n^l} \to D^l(\xi, \bar{\xi}), \tag{7}$$

for some deterministic scalar $D^l(\xi, \bar{\xi})$. Combining our calculations here, we see D^l satisfies the recurrence

$$D^l(\xi, \bar{\xi}) = \mathbb{E}\,\eta\bar{\eta}\,\mathbb{E}\,\phi'(\zeta)\phi'(\bar{\zeta}) = D^{l+1}(\xi, \bar{\xi})\,\mathbb{E}\,\phi'(\zeta)\phi'(\bar{\zeta}) \tag{8}$$

$$\text{where} \quad (\eta, \bar{\eta}) \sim \mathcal{N}\left(0, \begin{pmatrix} D^{l+1}(\xi,\xi) & D^{l+1}(\xi,\bar{\xi}) \\ D^{l+1}(\bar{\xi},\xi) & D^{l+1}(\bar{\xi},\bar{\xi}) \end{pmatrix}\right),$$

$$(\zeta, \bar{\zeta}) \sim \mathcal{N}\left(0, \begin{pmatrix} C^l(\xi,\xi) & C^l(\xi,\bar{\xi}) \\ C^l(\bar{\xi},\xi) & C^l(\bar{\xi},\bar{\xi}) \end{pmatrix} + 1\right)$$

Together with Eq. (5) and Eq. (7), we have

$$\left\langle \nabla_{\omega^l} f(\xi), \nabla_{\omega^l} f(\bar{\xi}) \right\rangle \to C^{l-1}(\xi, \bar{\xi}) D^l(\xi, \bar{\xi}), \quad \forall l \in [2, L].$$

Similarly, because $\nabla_{b^l} f(\xi) = \nabla_{h^l} f(\xi) = dh^l/\sqrt{n^l}$, we have

$$\left\langle \nabla_{b^l} f(\xi), \nabla_{b^l} f(\bar{\xi}) \right\rangle \to D^l(\xi, \bar{\xi}), \quad \forall l \in [2, L].$$

So the NTK should converge like

$$\Theta(\xi, \bar{\xi}) \to \sum_{l=1}^{L+1} C^{l-1}(\xi, \bar{\xi}) D^l(\xi, \bar{\xi}) + \sum_{l=1}^{L} D^l(\xi, \bar{\xi}). \tag{9}$$

Together with Eq. (6) and Eq. (8), this in fact recovers the NTK limit formula in Jacot et al. [39].

5 NTK for Any Architecture? The Issues and the Proposal

The method presented in the last section for computing the MLP NTK already seem easier than that of Jacot et al. [39] to generalize to other architectures, but several thorny issues still remain.

Q1: Can we meaningfully generalize the NTK decomposition in Eq. (3)? For example, for an MLP with weights tied across layers (i.e. $W^l = W^{l+1}$, for all $l = 1, \dots, L-1$), we can generalize Eq. (3) into a similar decomposition, but how do we know terms like $\frac{dh^{l\top} d\bar{h}^l}{n^l}$ will converge or won't blow up to ∞ due to the extra correlations from weight tying?

Q2: Can we continue to assume gradient independence? GIA significantly simplified our calculation above for the MLP. However, at a first glance it would still seem absurd to assume $W^\top$ is independent from W. Now, for example, suppose we tie the weights across layers in the MLP. The additional correlations then make GIA even more questionable. Can we still assume GIA?

Q3: Can we uniformly handle the complexity of modern neural networks? Standard architectures like CNN, RNN, GRU, LSTM, transformer, ResNet, etc contain a wide variety of gadgets, and *a priori* it's not clear there's a systematic way of handling all of them at once.

The techniques in this paper yield the following answers:

A1: Yes. We can generalize Eq. (3) to decompose the NTK into a sum of products of inner products of the form $h^\top \bar{h}/n$ with $h, \bar{h} \in \mathbb{R}^n$, and importantly, each such term will turn out to tend to a deterministic finite constant as $n \to \infty$, implying NTK converges as well. See Eqs. (10) and (12).

A2: Conditional Yes. It turns out, somewhat counterintuitively, whether GIA works doesn't depend on the hidden-to-hidden weight matrices (which GIA concerns) so much as the output layer weights. The following is a general but easily checkable condition that implies GIA:

Condition 1 (Simple GIA Check). The output layer (like W^{L+1} in the MLP example above) is sampled independently and with zero mean from all other parameters and is not used anywhere else in the interior of the network[3].

At a very high level, Condition 1 implies GIA because any weight matrix W can only interact with its transpose $W^\top$ via a path that goes through the last layer weights. If these weights are sampled independently and with zero mean, then such interactions are zeroed out as well. See Eq. (16) for a concrete explanation. In Section 6.3, we also show a counterexample where Condition 1 is violated and GIA doesn't work[4]. For a more general condition guaranteeing GIA, see Definition A.3.

A3: Yes. We introduce a simple and general language, NETSOR$\top$ (extending NETSOR from Yang [62]), expressing compositions of matrix multiplication and nonlinearity application, such that if an NN satisfies Condition 1 and one can write down its forward and backward computations in NETSOR$\top$ (as can be done for standard architectures), then its NTK provably converges under mild regularity conditions (Corollary 7.3). This NETSOR$\top$ program can allow one to mechanistically compute the infinite-width NTK by recursively applying the Master Theorem (Theorem 7.2).

6 Strategy for Computing the Infinite-Width NTK

For general architectures, we can in fact compute the NTK with an overall strategy very similar to Eq. (3) and Eq. (9).

6.1 The Canonical Decomposition

Consider a neural network[5] $f(\xi)$ with input $\xi \in \mathbb{R}^d$, scalar output, and with weights W and biases b such that any weight $W \in \mathbb{R}^{n\times m}$ is always used in the computation of $f(\xi)$ in the form $y(\xi) = Wz(\xi)$, for possibly many different vectors $y(\xi) \in \mathbb{R}^n, z(\xi) \in \mathbb{R}^m$. For example, in the MLP example above, W would be W^l for some l, and $y(\xi) = h^l(\xi), z(\xi) = x^{l-1}(\xi)$. If the MLP weights are tied across layers with $W = W^2 = \cdots = W^L$, then $(y, z) \in \{(h^2, x^1), \ldots, (h^L, x^{L-1})\}$.

Suppose that we adopt the NTK parametrization where W is factored as $W = \frac{1}{\sqrt{m}}\omega$ for $\omega \in \mathbb{R}^{n\times m}$, and ω, instead of W, is trained. Then the NTK Θ of f is a sum

$$\Theta(\xi, \bar{\xi}) = \sum_\omega \left\langle \nabla_\omega f(\xi), \nabla_\omega f(\bar{\xi}) \right\rangle + \sum_b \left\langle \nabla_b f(\xi), \nabla_b f(\bar{\xi}) \right\rangle \tag{10}$$

over biases b and factorized weights ω. In the MLP example with tied-weights $W = W^2 = \cdots = W^L \in \mathbb{R}^{n\times n}$ and $W = \frac{1}{\sqrt{n}}\omega$, we can write $\nabla_\omega f(\xi) = \frac{1}{n}\sum_{l=1}^{L-1} dh^{l+1}\, x^{l\top}$, and

$$\begin{aligned}\left\langle \nabla_\omega f(\xi), \nabla_\omega f(\bar{\xi}) \right\rangle &= \frac{1}{n^2}\left\langle \sum_{l=1}^{L-1} dh^{l+1}\, x^{l\top}, \sum_{\ell=1}^{L-1} d\bar{h}^{\ell+1}\, \bar{x}^{\ell\top} \right\rangle \\ &= \frac{1}{n^2}\sum_{l,\ell=1}^{L-1} \left\langle dh^{l+1}\, x^{l\top}, d\bar{h}^{\ell+1}\, \bar{x}^{\ell\top} \right\rangle = \sum_{l,\ell=1}^{L-1} \frac{dh^{l+1\top} d\bar{h}^{\ell+1}}{n} \frac{x^{l\top}\bar{x}^{\ell}}{n}.\end{aligned}$$

[3] i.e. if the output weight is v and the output is $v^\top x$, then x does not depend on v.

[4] Note that GIA means we can assume the backward weights are independent from the forward weights but multiple usages of backward weights (e.g. in an RNN backprop) are not assumed to be independent from each other.

[5] formally, we consider any neural network whose computation can be expressed in NETSOR$\top$ (Definition 7.1); however, in this section, an intuitive understanding of "neural network" is enough.

In the general case, consider any two inputs $\xi, \bar{\xi}$ to f (possibly equal). If we abbreviate $\bar{y} = y(\bar{\xi}), \bar{z} = z(\bar{\xi}), dy = \sqrt{n}\nabla_y f(\xi), d\bar{y} = \sqrt{n}\nabla_{\bar{y}} f(\bar{\xi})$, then we can express the contribution of $\nabla_\omega f$ to the NTK Θ of f as

$$\left\langle \nabla_\omega f(\xi), \nabla_\omega f(\bar{\xi}) \right\rangle = \frac{1}{m} \left\langle \nabla_W f(\xi), \nabla_W f(\bar{\xi}) \right\rangle = \frac{1}{mn} \left\langle \sum_{y,z} dy\, z^\top, \sum_{\bar{y},\bar{z}} d\bar{y}\, \bar{z}^\top \right\rangle$$
$$= \frac{1}{mn} \sum_{y,z,\bar{y},\bar{z}} \left\langle dy\, z^\top, d\bar{y}\, \bar{z}^\top \right\rangle = \sum_{y,z,\bar{y},\bar{z}} \frac{dy^\top d\bar{y}}{n} \frac{z^\top \bar{z}}{m} \tag{11}$$

where the sum is over all matrix multiplication of the form $y = Wz$ (resp. $\bar{y} = W\bar{z}$) used in the computation of $f(\xi)$ (resp. $f(\bar{\xi})$). Notice how Eq. (10) generalizes Eq. (3), and Eq. (4) is just Eq. (11) where the sum is over the singleton sets $\{h^l, x^{l-1}\}$ and $\{\bar{h}^l, \bar{x}^{l-1}\}$.

We will show below (Theorem 7.2) that $\frac{dy^\top d\bar{y}}{n}$ and $\frac{z^\top \bar{z}}{m}$ both converge almost surely to some deterministic limits $D^{y,\bar{y}}(\xi, \bar{\xi})$ and $C^{z,\bar{z}}(\xi, \bar{\xi})$ if the factored weights and biases ω, b are drawn from standard Gaussians (i.e. in the *NTK parametrization*), as widths tend to infinity. Similarly, we will also show the convergence of $\nabla_b f(\xi)^\top \nabla_b f(\bar{\xi})$ for any bias b of f and compute its limiting value $D^b(\xi, \bar{\xi})$. Then the limiting NTK is given by

$$\mathring{\Theta}(\xi, \bar{\xi}) = \sum_{\text{weight } W} \sum_{\substack{y,z: y = Wz \\ \bar{y},\bar{z}: \bar{y} = W\bar{z}}} D^{y,\bar{y}}(\xi, \bar{\xi}) C^{z,\bar{z}}(\xi, \bar{\xi}) + \sum_{\text{bias } b} D^b(\xi, \bar{\xi}). \tag{12}$$

6.2 Intuitive Rules for Computing Intermediate Kernels C and D

Here we present intuitive rules for computing C and D, which would yield the NTK by Eq. (12). Their justifications will follow in the next section. Consider the first forward and backward propagations of a neural network. Assume for simplicity that the hidden layers all have the same width, denoted n, which tends to infinity. Then under Condition 1[6], the following is the key intuition for computing the kernels C and D for arbitrary architecture.

Box 1 Key Intuitions for Understanding a Wide Neural Network

When the width $n \gg 1$, every (pre-)activation vector $x \in \mathbb{R}^n$ has roughly iid coordinates distributed as some random variable denoted Z^x. The set of random variables $\{Z^x\}_x$ over $x \in \mathbb{R}^n$ in this computation is possibly correlated, as $\{x_\alpha\}_x$ is possibly correlated for each $\alpha \in [n]$, but is roughly iid across α. Thus, for any vectors $x, y \in \mathbb{R}^n$, as $n \to \infty$,

$$x^\top y / n \to \mathbb{E}\, Z^x Z^y,$$

which is the form of the limit (kernels C and D) we want. We can use the following rules to compute Z^x recursively.

1. **(Nonlin)** For any fixed (i.e. constant as $n \to \infty$) k and $\phi : \mathbb{R}^k \to \mathbb{R}$, we have[a]
$$Z^{\phi(x^1, \ldots, x^k)} = \phi(Z^{x^1}, \ldots, Z^{x^k}).$$
2. **(MatMul)** For any set of $\mathbb{R}^n$ vectors $\mathcal{X}$ and a matrix $W \in \mathbb{R}^{n \times n}$ with $W_{\alpha\beta} \sim \mathcal{N}(0, \sigma_W^2/n)$, the set of random variables $\{Z^{Wx} : x \in \mathcal{X}\}$ is jointly Gaussian with zero mean and covariance
$$\mathrm{Cov}(Z^{Wx}, Z^{W\bar{x}}) = \sigma_W^2\, \mathbb{E}\, Z^x Z^{\bar{x}}, \quad \text{for any } x, \bar{x} \in \mathcal{X}.$$
If $\mathcal{Y}$ is any set of $\mathbb{R}^n$ vectors and $\bar{W} \neq W$, then $\{Z^{Wx} : x \in \mathcal{X}\}$ is independent from $\{Z^{\bar{W}y} : y \in \mathcal{Y}\}$.

[a] here ϕ is applied coordinatewise to $x^1, \ldots, x^k$, i.e. $\phi(x^1, \ldots, x^k)_\alpha = \phi(x^1_\alpha, \ldots, x^k_\alpha)$

Remark 6.1. Rule 2 applies even if W is correlated with vectors in $\mathcal{X}$, for example if $x = W\bar{x}$ or $x = W^\top \bar{x}$ for some $x, \bar{x} \in \mathcal{X}$.

[6] or when the associated NETSOR$\top$ program is BP-like (Definition A.3)

Remark 6.2. In Rule 2, if we set $\bar{W} = W^\top$, then the rule implies $\left\{Z^{W^\top y} : y \in \mathcal{Y}\right\}$ is independent from $\left\{Z^{Wx} : x \in \mathcal{X}\right\}$. This is how we use GIA implied by Condition 1.

Remark 6.3. To reason about the computation of a wide neural network on the input ξ of fixed dimension, we apply the above rules to the first layer embedding $W\xi$ into $\mathbb{R}^n$, but not ξ itself.

These rules largely generalize our intuitive treatment of the MLP example above: The "iid coordinates" intuition suggests the limits in Eqs. (5) and (7). The recursive relations in Eqs. (6) and (8) of C and D are then given by Rule 1 and 2. Now let us examine the power of these rules by looking at more advanced weight sharing inside an RNN.

6.2.1 Example: RNN

Consider the RNN with state s^t at time t evolving according to

$$s^t(\xi) = \phi(g^t(\xi) + u^t(\xi) + b), \quad g^t(\xi) = Ws^{t-1}(\xi), \quad u^t(\xi) = U\xi^t \tag{13}$$

with input sequence $\xi = \left\{\xi^1, \ldots, \xi^t, \ldots, \xi^T \in \mathbb{R}^d\right\}$, nonlinearity ϕ, weights $W \in \mathbb{R}^{n\times n}, U \in \mathbb{R}^{n\times d}$, and bias $b \in \mathbb{R}^n$. The RNN outputs $v^\top s^T(\xi)/\sqrt{n} \in \mathbb{R}$ for some output weights $v \in \mathbb{R}^n$ and the last state $s^T(\xi)$. We shall sample $W_{\alpha\beta} \sim \mathcal{N}(0, 1/n), U_{\alpha\beta} \sim \mathcal{N}(0, 1/d), b_\alpha \sim \mathcal{N}(0, 1), v_\alpha \sim \mathcal{N}(0, 1)$. Then Condition 1 becomes true automatically, and we may use the rules in Box 1.

As in Eq. (11), we shall consider a second input sequence $\bar{\xi} = \{\bar{\xi}^1, \ldots, \bar{\xi}^t, \ldots \in \mathbb{R}^d\}$, possibly with $\bar{\xi} = \xi$. There are two weight matrices in this network, W and U. For W, the double sum in Eq. (11) is over $\{g^t, s^{t-1}\}_t$ and $\{\bar{g}^t, \bar{s}^{t-1}\}_t$. Thus we seek to calculate the limits of $\frac{s^{t\top}\bar{s}^r}{n}$ and $\frac{dg^{t\top}d\bar{g}^r}{n}$ for all t and r. Similarly, for U, the double sum in Eq. (11) is over $\{u^t, \xi^t\}_t$ and $\left\{\bar{u}^t, \bar{\xi}^t\right\}_t$. Thus we also seek to calculate the limits of $\frac{du^{t\top}d\bar{u}^r}{n}$, whereas we are already given $\frac{\xi^{t\top}\bar{\xi}^r}{d}$, which is constant in n for each t and r.

Forward As width $n \to \infty$ (but input dimension d fixed), Box 1 says we can think of g^t, u^t, s^t, b as having iid coordinates distributed resp. as some random variables $Z^{g^t}, Z^{u^t}, Z^{s^t}, Z^b$. Of course, $Z^b = \mathcal{N}(0, 1)$ and $\{Z^{u^t}, Z^{\bar{u}^t}\}_t$ is jointly Gaussian with mean zero and covariance $\mathrm{Cov}(Z^{u^t}, Z^{\bar{u}^r}) = \xi^{t\top}\bar{\xi}^r/d$. By Rule 2, $\{Z^{g^t}, Z^{\bar{g}^t}\}_t$ is also jointly Gaussian with mean zero, and it has covariance $\mathrm{Cov}(Z^{g^t}, Z^{\bar{g}^r}) = \mathbb{E}\, Z^{s^{t-1}} Z^{\bar{s}^{r-1}}$. Stringing them together gives us the following recursion

$$\mathbb{E}\, Z^{s^t} Z^{\bar{s}^r} = \mathbb{E}\, \phi(Z^{g^t} + Z^{u^t} + Z^b)\phi(Z^{\bar{g}^r} + Z^{\bar{u}^r} + Z^b) = \mathbb{E}\, \phi(\zeta_1)\phi(\zeta_2),$$
$$\text{where} \quad (\zeta_1, \zeta_2) \sim \mathcal{N}\left(0, \mathbb{E}\begin{pmatrix} \left(Z^{s^{t-1}}\right)^2 & Z^{s^{t-1}} Z^{\bar{s}^{r-1}} \\ Z^{\bar{s}^{r-1}} Z^{s^{t-1}} & \left(Z^{\bar{s}^{r-1}}\right)^2 \end{pmatrix} + \frac{\xi^{t\top}\xi^r}{d} + 1\right).$$

This recursion yields the desired limit

$$C^{s^t, \bar{s}^r}(\xi, \bar{\xi}) = \lim_{n\to\infty} \frac{s^{t\top}\bar{s}^r}{n} = \mathbb{E}\, Z^{s^t} Z^{\bar{s}^r}. \tag{14}$$

Backward The backward equation is given by

$$ds^{t-1} = W^\top dg^t, \quad dg^t = du^t = \phi'\left(g^t + u^t + b\right) \odot ds^t. \tag{15}$$

By Box 1, we should think of ds^t as having iid coordinates distributed like some random variable Z^{ds^t} which satisfy

$$\begin{aligned}
\mathbb{E}\, Z^{ds^t} Z^{d\bar{s}^r} &= \mathbb{E}\, Z^{du^{t+1}} Z^{du^{r+1}} \\
&= \mathbb{E}\, \phi'(Z^{g^{t+1}} + Z^{u^{t+1}} + Z^b) Z^{ds^{t+1}} \phi'(Z^{\bar{g}^{r+1}} + Z^{\bar{u}^{r+1}} + Z^b) Z^{d\bar{s}^{r+1}} \\
&= \mathbb{E}\, Z^{ds^{t+1}} Z^{d\bar{s}^{r+1}}\, \mathbb{E}\, \phi'(Z^{g^{t+1}} + Z^{u^{t+1}} + Z^b)\phi'(Z^{\bar{g}^{r+1}} + Z^{\bar{u}^{r+1}} + Z^b) \\
&= \mathbb{E}\, Z^{ds^{t+1}} Z^{d\bar{s}^{r+1}}\, \mathbb{E}\, \phi'(\zeta_1)\phi'(\zeta_2),
\end{aligned}$$

where $(\zeta_1, \zeta_2) \sim \mathcal{N}\left(0, \mathbb{E}\begin{pmatrix} \left(Z^{s^t}\right)^2 & Z^{s^t} Z^{\bar{s}^r} \\ Z^{\bar{s}^r} Z^{s^t} & \left(Z^{\bar{s}^r}\right)^2 \end{pmatrix} + \frac{\xi^{t\top}\xi^r}{d} + 1\right)$. This recursion yields the desired limit

$$D^{s^t, \bar{s}^r}(\xi, \bar{\xi}) = \lim_{n\to\infty} \frac{ds^{t\top} d\bar{s}^r}{n} = \mathbb{E}\, Z^{ds^t} Z^{d\bar{s}^r}$$
$$= D^{u^{t+1}, \bar{u}^{r+1}}(\xi, \bar{\xi}) = \lim_{n\to\infty} \frac{du^{t+1\top} d\bar{u}^{r+1}}{n} = \mathbb{E}\, Z^{du^{t+1}} Z^{du^{r+1}}.$$

Combined with Eqs. (12) and (14), we can compute the infinite-width NTK. See Appendix E.2 also for generalization to RNN with average pooling.

Other standard architectures follow a similar scheme; see Appendices D and E.

6.3 GIA Makes or Breaks the Intuitive Rules of Box 1

Without Condition 1, rules of Box 1 may not work When the last layer outputs the average of the final embedding, Condition 1 doesn't hold anymore. Let us see how this means we can't treat $W^\top$ as independent from W. Suppose we have a 2-hidden-layer network

$$x^1 = W^1\xi + 1, \quad h^2 = W^2 x^1, \quad x^2 = \phi(h^2), \quad y = 1^\top x^2/n$$

with $\phi(z) = z^2$ being the square function, $\xi = 0 \in \mathbb{R}^d, y \in \mathbb{R}, x^1, h^2, x^2 \in \mathbb{R}^n, W^1 \in \mathbb{R}^{n\times d}, W^2 \in \mathbb{R}^{n\times n}, W^1_{\alpha\beta} \sim \mathcal{N}(0, 1/d), W^2_{\alpha\beta} \sim \mathcal{N}(0, 1/n)$. If we set $dx^2 = n\frac{\partial y}{\partial x^2}$, then backprop yields

$$dx^2 = 1, \quad dh^2 = 2h^2 \odot 1 = 2h^2, \quad dx^1 = W^{2\top} dh^2 = 2W^{2\top} h^2 = 2W^{2\top} W^2 x^1$$

By Rule 2, h^2 should have coordinates distributed like $Z^{h^2} = \mathcal{N}(0, 1)$ and likewise dh^2 has coordinates distributed like $Z^{dh^2} = 2Z^{h^2} = \mathcal{N}(0, 4)$.

If we assumed that $W^{2\top}$ is independent from W^2, then this would imply dx^1 also has coordinates distributed like $\mathcal{N}(0, 4)$. But a simple calculation shows its mean cannot be 0 in reality:

$$\mathbb{E}\, dx^1_\alpha = 2\,\mathbb{E}\sum_{\beta,\gamma} W^2_{\beta\alpha} W^2_{\beta\gamma} x^1_\gamma = 2\sum_\beta \mathbb{E}(W^2_{\beta\alpha})^2 x^1_\alpha + 2\sum_\beta \sum_{\gamma\neq\alpha} \mathbb{E}\, W^2_{\beta\alpha} W^2_{\beta\gamma} x^1_\gamma = 2\,\mathbb{E}\, x^1_\alpha = 2$$

where the second sum vanishes because the terms $W^2_{\beta\alpha}, W^2_{\beta\gamma}, x^1_\gamma$ in the product are independent, while in the first sum we have $\sum_\beta \mathbb{E}(W^2_{\beta\alpha})^2 = 1$.

Intuition for why Condition 1 implies GIA On the other hand, if the last layer is $y = v^\top x^2/\sqrt{n}$ for $v_\alpha \sim \mathcal{N}(0, 1)$ so that Condition 1 holds, then a similar calculation with $dx^2 = \sqrt{n}\frac{\partial y}{\partial x^2}$ yields

$$\mathbb{E}\, dx^1_\alpha = 2\sum_\beta \mathbb{E}\, v_\beta (W^2_{\beta\alpha})^2 x^1_\alpha + 2\sum_\beta \sum_{\gamma\neq\alpha} \mathbb{E}\, v_\beta W^2_{\beta\alpha} W^2_{\beta\gamma} x^1_\gamma = 0 \tag{16}$$

which now vanishes because v_β appears unpaired in the expectation of the first sum and it is independent from everything else. This illustrates an intuition for *why Condition 1 implies GIA*: the last layer weights zero out all potential pathways through which W and $W^\top$ can correlate.

We have demonstrated our intuitive rules for calculating the kernels that combine to form the NTK. Now let us rigorously justify these rules.

7 NETSOR$\top$

To justify our intuitive calculations, we need to pin down the range of architectures they are valid for, and also the precise regularity conditions for the corresponding limits to hold. Here 1) we introduce the NETSOR$\top$ language such that an architecture is covered if its forward and backward propagations are expressible in NETSOR$\top$, and 2) we prove a Master Theorem for NETSOR$\top$ programs that allows us to justify the intuitions of Box 1 rigorously.

Definition 7.1 (Simplified NETSOR$\top$)**.** For simplicity's sake[7], in this section, a NETSOR$\top$ program is just a sequence of $\mathbb{R}^n$ vectors inductively generated via one of the following ways from an initial set $\mathcal{V}$ of random $\mathbb{R}^n$ vectors and a set $\mathcal{W}$ of random $n \times n$ matrices

Nonlin Given $\phi : \mathbb{R}^k \to \mathbb{R}$ and $x^1, \dots, x^k \in \mathbb{R}^n$, we can generate $\phi(x^1, \dots, x^k) \in \mathbb{R}^n$

MatMul Given $W \in \mathbb{R}^{n \times n}$ and $x \in \mathbb{R}^n$, we can generate $Wx \in \mathbb{R}^n$ or $W^\top x \in \mathbb{R}^n$

Note that ϕ in Nonlin is applied coordinatewise. Here, n should be thought of as the width, $\mathcal{W}$ the weight matrices, and $\mathcal{V}$ the biases and the first layer *embeddings* of inputs. Note that ϕ in Nonlin can also be linear, e.g. $x, y \mapsto x + y$ in a skip connection. For example, the RNN equations (Eqs. (13) and (15)) form a natural NETSOR$\top$ program: the initial vectors are $\mathcal{V} = \{u^t(\xi) = U\xi^t\}_{t=1}^T \cup \{v = ds^T, b\}$ and the initial matrix is $\mathcal{W} = \{W\}$, and new vectors $\{g^t, s^t, ds^t, dg^t\}_t$ are formed inductively according to

$$g^t(\xi) = Ws^{t-1}(\xi) \qquad ds^{t-1} = W^\top dg^t \qquad \textbf{MatMul}$$
$$s^t(\xi) = \phi(g^t(\xi) + u^t(\xi) + b) \qquad dg^t = \phi'\left(g^t + u^t + b\right) \odot ds^t. \qquad \textbf{Nonlin}$$

Like NETSOR in Yang [62] for forward propagation, NETSOR$\top$ can express all standard architectures. For example, in a program expressing convolution neural network with width n, the activation vector for each pixel across all channels is represented by an $\mathbb{R}^n$ vector. See Appendix D for more details and other examples of modern deep learning layers. We state the Master Theorem below assuming a generalization of Condition 1 to a condition called *BP-like* (short for "backpropagation-like") for NETSOR$\top$ programs; see Definition A.3. On the first read-through, we recommend the reader to mentally replace *BP-like* with Condition 1 which covers most of the cases we are interested in practice with regard to NTK calculations. In previous sections, we cared about limits of the form $x^\top y/n = \frac{1}{n}\sum_{\alpha=1}^n \psi(x_\alpha, y_\alpha)$ where ψ is the product function. The Master Theorem tells us how to compute this for almost any function ψ.

Theorem 7.2 (BP-like NETSOR$\top$ Master Theorem)**.** *Consider a* NETSOR$\top$ *program. Suppose: 1) for each initial $W \in \mathcal{W}$, $W_{\alpha\beta} \sim \mathcal{N}(0, \sigma_W^2/n)$ for an associated variance σ_W^2; 2) there is a multivariate Gaussian $Z^{\mathcal{V}} = \{Z^g : g \in \mathcal{V}\} \in \mathbb{R}^{|\mathcal{V}|}$ such that the initial set of vectors $\mathcal{V}$ are sampled like $\{g_\alpha : g \in \mathcal{V}\} \sim Z^{\mathcal{V}}$ iid for each $\alpha \in [n]$. If the program is BP-like and all ϕ used in* ***Nonlin*** *are polynomially bounded*[8]*, then*

$$\frac{1}{n}\sum_{\alpha=1}^n \psi(h_\alpha^1, \dots, h_\alpha^k) \xrightarrow{\text{a.s.}} \mathbb{E}\,\psi(Z^{h^1}, \dots, Z^{h^k}), \quad \textit{as} \quad n \to \infty, \tag{17}$$

for any collection of vectors $h^1, \dots, h^k$ in the program and any polynomially bounded $\psi : \mathbb{R}^k \to \mathbb{R}$, where Z^{h^i} are defined in Box 1.[9]

This rigorously justifies the intuitions in the previous section (after checking the regularity conditions).

Back to the MLP example Eq. (2) Assuming $n^1 = \cdots = n^L$, we'd have $\mathcal{W} = \{W^2, \dots, W^L\}$ and $\mathcal{V} = \{b^l\}_l \cup \{W^1\xi, W^1\bar{\xi}\} \cup \{dx^L = \sqrt{n}W^{L+1}\}$. The weight matrices are by default sampled like in Theorem 7.2, and $\mathcal{V}$ is distributed as in Theorem 7.2 with $\left\{Z^{W^1\xi}, Z^{W^1\bar{\xi}}\right\} \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{\dim(\xi)}\begin{pmatrix} \|\xi\|^2 & \xi^\top\bar{\xi} \\ \bar{\xi}^\top\xi & \|\bar{\xi}\|^2 \end{pmatrix}\right)$ and with $Z^{b^l}, Z^{dx^L} \sim \mathcal{N}(0, \sigma_b^2)$ independently. The forward and backpropagation of the MLP (Eq. (2)) form a natural NETSOR$\top$ program, once we unwind it a little: We set $g^1(\xi) = W^1\xi$ via **Nonlin** (with identity as the nonlinearity), and

$$g^l(\xi) = W^l x^{l-1}(\xi) \qquad dx^{l-1}(\xi) = W^{l\top} dg^l(\xi) \qquad \textbf{MatMul}$$
$$x^l(\xi) = \phi(g^l(\xi) + b^l) \qquad dg^l(\xi) = \phi'(g^l(\xi) + b^l) \odot dx^l(\xi) \qquad \textbf{Nonlin}$$

[7] See Appendix A for the formal description of the general notion of NETSOR$\top$; for variable dimension generalization, see Appendix C.

[8] We say a function $\phi : \mathbb{R}^k \to \mathbb{R}$ is *polynomially-bounded* if $|\phi(x)| \le C\|x\|^p + c$ for some $p, C, c > 0$, for all $x \in \mathbb{R}^k$.

[9] Difference with [63, Thm 5.1]: We have gotten rid of the "rank convergence" assumption by showing that it comes for free. See CoreSet and Lemma G.6 in Appendix G.

and similarly for computations on $\bar{\xi}$. This program is BP-like because the MLP satisfies Condition 1. For typical activation function ϕ like ReLU, ϕ and its derivative are both polynomially bounded. Therefore Theorem 7.2 applies: for example, with $\psi(x, y) = xy$ applied to the vectors $dh^l, d\bar{h}^l$, Eq. (17) recovers Eq. (7) rigorously.

Summary So a formal proof of the NTK convergence proceeds as follows:

1. Express the network in NETSOR$\top$
2. Check the network satisfies Condition 1 or more generally the program is BP-like
3. Check that the ϕs of the program (which correspond to both the activation functions in the original network and their derivatives) are all polynomially bounded

This is sufficient to show that the NTK converges almost surely as width goes to infinity. To further compute this limit, follow Eq. (12) and Box 1 as in the RNN example in Section 6.2.1. As a summary:

Corollary 7.3. *Let f be a (possibly recurrent) neural network of standard architecture with scalar output and satisfying Condition 1. If its nonlinearities have polynomially bounded weak derivatives, then its NTK Θ converges almost surely, over any finite set of inputs, to a deterministic kernel $\mathring{\Theta}$*

$$\Theta \xrightarrow{\text{a.s.}} \mathring{\Theta}$$

as its widths go to infinity and each of its factored weights ω and biases b are randomly initialized as $\omega_{\alpha\beta} \sim \mathcal{N}(0, \sigma_\omega^2), b_\alpha \sim \mathcal{N}(0, \sigma_b^2)$ for some $\sigma_\omega, \sigma_b \geq 0$.

See more examples of NTK computations and proofs of convergence (Appendix E) in the appendix.

Remark 7.4 (Importance of BP-like Condition). Recall the counterexample for GIA in Section 6.3, which can be expressed in a valid but not BP-like NETSOR$\top$ program. Therefore Theorem 7.2 is not true when the BP-like condition does not hold. We will extend Theorem 7.2 to cover the non-BP-like cases in a future paper, which requires much more machinery.

Generalizations All results in this section can be generalized to the case where the dimensions in the NETSOR program are not all equal (such as when an NN has varying widths across layers); see Appendix C. It is easy to show Corollary 7.3 also holds when the output is multidimensional (possibly variable-dimensional, like in a language model). Architectural blocks like layernorm or attention requires extending NETSOR$\top$ to a more powerful language NETSOR$\top^+$ (like how NETSOR$^+$ extends NETSOR in Yang [62]), which is discussed in Appendix B.

The NETSOR$\top$ Master Theorem has implications outside of NTK as well. For example, most of the semirigorous computations made in the signal propagation literature [55, 58, 61, 67, 68] can now be justified rigorously. See Yang [63] for more discussions.

Guide to the Appendix

Appendix A Treats NETSOR$\top$ from a formal perspective (similar to the style of Yang [62]).

Appendix B Introduces NETSOR$\top^+$ and proves its Master Theorem.

Appendix C Extends NETSOR and NETSOR$\top^+$ to allow matrices, vectors of variable dimensions.

Appendix D Examples writing forward and backprop of standard architectures in NETSOR$\top$.

Appendix E Examples calculating the limiting NTKs for RNN, CNN, transformer, and batchnorm.

Appendix F Theoretical tools for our main proof.

Appendix G Proof of our main theorem Theorem 7.2

8 Conclusion

We showed that for any randomly initialized feedforward or recurrent neural network of standard architecture, its NTK converges almost surely to a deterministic kernel. We did so by introducing NETSOR$\top$, a language capable of expressing both forward and backward propagation of NNs, along with a tool (Theorem 7.2) for understanding the behavior of such computations. We hope our work lays the foundation for understanding modern overparametrized neural networks.

Acknowledgements

We thank Edward Hu, Judy Shen, Zhiyuan Li, Ilya Razenshteyn, Jason Lee, Huishuai Zhang, Simon Du, Suriya Gunasekar, Etai Littwin, Roman Novak, Jaehoon Lee, Sam Schoenholz, Jascha Sohl-Dickstein, Tomer Galanti, Janardhan Kulkarni, Zeyuan Allen-Zhu, and Jeffrey Pennington for feedback and discussions.

References

[1] Joshua Achiam, Ethan Knight, and Pieter Abbeel. Towards Characterizing Divergence in Deep Q-Learning. *arXiv:1903.08894 [cs]*, March 2019. URL http://arxiv.org/abs/1903.08894.

[2] Sina Alemohammad, Zichao Wang, Randall Balestriero, and Richard Baraniuk. The recurrent neural tangent kernel, 2020.

[3] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers. *arXiv:1811.04918 [cs, math, stat]*, November 2018. URL http://arxiv.org/abs/1811.04918.

[4] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A Convergence Theory for Deep Learning via Over-Parameterization. *arXiv:1811.03962 [cs, math, stat]*, November 2018. URL http://arxiv.org/abs/1811.03962.

[5] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. On the Convergence Rate of Training Recurrent Neural Networks. *arXiv:1810.12065 [cs, math, stat]*, October 2018. URL http://arxiv.org/abs/1810.12065.

[6] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On Exact Computation with an Infinitely Wide Neural Net. *arXiv:1904.11955 [cs, stat]*, April 2019. URL http://arxiv.org/abs/1904.11955.

[7] Sanjeev Arora, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks, 2019.

[8] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv:1607.06450 [cs, stat]*, July 2016. URL http://arxiv.org/abs/1607.06450.

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*, September 2014. URL http://arxiv.org/abs/1409.0473.

[10] Ronen Basri, David Jacobs, Yoni Kasten, and Shira Kritchman. The Convergence Rate of Neural Networks for Learned Functions of Different Frequencies. *arXiv:1906.00425 [cs, eess, stat]*, June 2019.

[11] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral Networks and Locally Connected Networks on Graphs. *arXiv:1312.6203 [cs]*, December 2013. URL http://arxiv.org/abs/1312.6203.

[12] Qi Cai, Zhuoran Yang, Jason D. Lee, and Zhaoran Wang. Neural temporal-difference and q-learning provably converge to global optima, 2019.

[13] Minmin Chen, Jeffrey Pennington, and Samuel Schoenholz. Dynamical Isometry and a Mean Field Theory of RNNs: Gating Enables Signal Propagation in Recurrent Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 873–882, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR. URL http://proceedings.mlr.press/v80/chen18i.html.

[14] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv:1406.1078 [cs, stat]*, June 2014. URL http://arxiv.org/abs/1406.1078.

[15] Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009. URL http://papers.nips.cc/paper/3628-kernel-methods-for-deep-learning.

[16] Amit Daniely, Roy Frostig, and Yoram Singer. Toward Deeper Understanding of Neural Networks: The Power of Initialization and a Dual View on Expressivity. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2253–2261. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6427-toward-deeper-understanding-of-neural-networks-the-power-of-initialization-and-a-dual-view-on-expressivity.pdf.

[17] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *arXiv:1606.09375 [cs, stat]*, June 2016.

[18] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient Descent Provably Optimizes Over-parameterized Neural Networks. *arXiv:1810.02054 [cs, math, stat]*, October 2018. URL http://arxiv.org/abs/1810.02054.

[19] Simon S. Du, Kangcheng Hou, Barnabás Póczos, Ruslan Salakhutdinov, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels, 2019.

[20] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2224–2232. Curran Associates, Inc., 2015.

[21] Ethan Dyer and Guy Gur-Ari. Asymptotics of wide networks from feynman diagrams, 2019.

[22] Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3-4):121–136, 1975.

[23] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[24] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Linearized two-layers neural networks in high dimension. *arXiv:1904.12191 [cs, math, stat]*, April 2019.

[25] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *arXiv:1406.2661 [cs, stat]*, June 2014. URL http://arxiv.org/abs/1406.2661.

[26] Boris Hanin. Which Neural Net Architectures Give Rise To Exploding and Vanishing Gradients? January 2018. URL https://arxiv.org/abs/1801.03744.

[27] Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel, 2019.

[28] Boris Hanin and David Rolnick. How to Start Training: The Effect of Initialization and Architecture. *arXiv:1803.01719 [cs, stat]*, March 2018. URL http://arxiv.org/abs/1803.01719.

[29] Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the Selection of Initialization and Activation Function for Deep Neural Networks. *arXiv:1805.08266 [cs, stat]*, May 2018. URL http://arxiv.org/abs/1805.08266.

[30] Tamir Hazan and Tommi Jaakkola. Steps Toward Deep Kernel Methods from Infinite Neural Networks. *arXiv:1508.05133 [cs]*, August 2015. URL http://arxiv.org/abs/1508.05133.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. pages 770–778, 2016. URL https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

[32] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep Convolutional Networks on Graph-Structured Data. *arXiv:1506.05163 [cs]*, June 2015.

[33] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[34] Jiri Hron, Yasaman Bahri, Jascha Sohl-Dickstein, and Roman Novak. Infinite attention: Nngp and ntk for deep attention networks, 2020.

[35] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. *arXiv:1608.06993 [cs]*, August 2016. URL http://arxiv.org/abs/1608.06993.

[36] Jiaoyang Huang and Horng-Tzer Yau. Dynamics of deep neural networks and neural tangent hierarchy, 2019.

[37] Wei Huang, Weitao Du, and Richard Yi Da Xu. On the neural tangent kernel of deep networks with orthogonal initialization, 2020.

[38] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *PMLR*, pages 448–456, June 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

[39] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. *arXiv:1806.07572 [cs, math, stat]*, June 2018. URL http://arxiv.org/abs/1806.07572.

[40] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, September 2016.

[41] Nicolas Le Roux and Yoshua Bengio. Continuous neural networks. In *Artificial Intelligence and Statistics*, pages 404–411, 2007.

[42] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[43] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[44] Jaehoon Lee, Yasaman Bahri, Roman Novak, Sam Schoenholz, Jeffrey Pennington, and Jascha Sohl-dickstein. Deep Neural Networks as Gaussian Processes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1EA-M-0Z.

[45] Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. *arXiv:1902.06720 [cs, stat]*, February 2019. URL http://arxiv.org/abs/1902.06720.

[46] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated Graph Sequence Neural Networks. *arXiv:1511.05493 [cs, stat]*, November 2015.

[47] Etai Littwin and Lior Wolf. Residual tangent kernels, 2020.

[48] Etai Littwin, Tomer Galanti, and Lior Wolf. On the optimization dynamics of wide hypernetworks, 2020.

[49] Etai Littwin, Ben Myara, Sima Sabah, Joshua Susskind, Shuangfei Zhai, and Oren Golan. Collegial ensembles, 2020.

[50] Alexander G. de G. Matthews, Mark Rowland, Jiri Hron, Richard E. Turner, and Zoubin Ghahramani. Gaussian Process Behaviour in Wide Deep Neural Networks. *arXiv:1804.11271 [cs, stat]*, April 2018. URL http://arxiv.org/abs/1804.11271.

[51] Radford M Neal. *BAYESIAN LEARNING FOR NEURAL NETWORKS*. PhD Thesis, University of Toronto, 1995.

[52] Roman Novak, Lechao Xiao, Jaehoon Lee, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Bayesian Deep Convolutional Networks with Many Channels are Gaussian Processes. *arXiv preprint arXiv:1810.05148*, 2018.

[53] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4788–4798. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7064-resurrecting-the-sigmoid-in-deep-learning-through-dynamical-isometry-theory-and-practice.pdf.

[54] George Philipp and Jaime G. Carbonell. The Nonlinearity Coefficient - Predicting Overfitting in Deep Neural Networks. *arXiv:1806.00179 [cs, stat]*, May 2018. URL http://arxiv.org/abs/1806.00179.

[55] Ben Poole, Subhaneil Lahiri, Maithreyi Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances In Neural Information Processing Systems*, pages 3360–3368, 2016.

[56] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf.

[57] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[58] Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep Information Propagation. 2017. URL https://openreview.net/pdf?id=H1W1UN9gg.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, \Lukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[60] Christopher K I Williams. Computing with Infinite Networks. In *Advances in neural information processing systems*, page 7, 1997.

[61] Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5393–5402, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR. URL http://proceedings.mlr.press/v80/xiao18a.html.

[62] Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes. In *Advances in Neural Information Processing Systems*, pages 9947–9960, 2019.

[63] Greg Yang. Scaling Limits of Wide Neural Networks with Weight Sharing: Gaussian Process Behavior, Gradient Independence, and Neural Tangent Kernel Derivation. *arXiv:1902.04760 [cond-mat, physics:math-ph, stat]*, February 2019.

[64] Greg Yang. Tensor programs iii: Neural matrix laws. 2020.

[65] Greg Yang and Hadi Salman. A fine-grained spectral perspective on neural networks, 2019.

[66] Greg Yang and Sam S. Schoenholz. Deep mean field theory: Layerwise variance and width variation as methods to control gradient explosion, 2018. URL https://openreview.net/forum?id=rJGY8GbR-.

[67] Greg Yang and Samuel S. Schoenholz. Mean Field Residual Network: On the Edge of Chaos. In *Advances in neural information processing systems*, 2017.

[68] Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. A Mean Field Theory of Batch Normalization. *arXiv:1902.08129 [cond-mat]*, February 2019. URL http://arxiv.org/abs/1902.08129.

[69] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic Gradient Descent Optimizes Over-parameterized Deep ReLU Networks. *arXiv:1811.08888 [cs, math, stat]*, November 2018. URL http://arxiv.org/abs/1811.08888.

A NETSOR$\top$: The Formal Version

While we recommend using NETSOR$\top$ defined in Definition 7.1 in practice, we give a formal treatment of the NETSOR$\top$ language here and formulate its corresponding Master Theorem (Theorem A.6), which is equivalent to Theorem 7.2 and which we will prove instead. The type system of this formal NETSOR$\top$ allows us to express the proof of Theorem A.6 more easily.

The formal syntax of NETSOR$\top$ extends NETSOR [62] by a new transpose (`Trsp`) instruction. Compared to Definition 7.1, we allow matrices and vectors to have varying dimension, and explicitly single out the vectors produced by **MatMul** via an elementary type system.

Definition A.1. NETSOR$\top$ *programs* are straightline programs, where each variable follows one of three types, G, H, or A (such variables are called *G-vars*, *H-vars*, and *A-vars*), and after input variables, new variables can be introduced by one of the rules `MatMul` or `Nonlin` to be discussed shortly. Variables of G and H types are vectors, while variables of A type are matrices. Each type is annotated by dimensionality information [10]:

- If x is a (vector) variable of type G (or H) and has dimension n, we write $x : \mathsf{G}(n)$ (or $x : \mathsf{H}(n)$).
- If A is a (matrix) variable of type A and has size $n_1 \times n_2$, we write $A : \mathsf{A}(n_1, n_2)$.

G is a *subtype* of H, which means that $x : \mathsf{G}(n)$ implies $x : \mathsf{H}(n)$. A NETSOR$\top$ program consists of the following two parts.

Input A set of input G- or A-vars (corresponding to the initial set of matrices and vectors in Definition 7.1).

Body New variables can be introduced and assigned via the following rules

`Trsp` if $A : \mathsf{A}(n_1, n_2)$ is an A-var, then we can form its transpose as an A-var:

$$A^\top : \mathsf{A}(n_2, n_1)$$

Naturally we identify $(A^\top)^\top$ with A.

`MatMul` if $A : \mathsf{A}(n_1, n_2)$ and $x : \mathsf{H}(n_2)$, then we can form a G-var via matrix-vector product:

$$Ax : \mathsf{G}(n_1)$$

`Nonlin` If $x^1, \ldots, x^k : \mathsf{G}(n)$ are G-vars with the same dimension n and $\phi : \mathbb{R}^k \to \mathbb{R}$, then we can form an H-var by coordinatewise application of ϕ

$$\phi(x^1, \ldots, x^k) : \mathsf{H}(n)$$

Output For the purpose of this paper[11], the output of a NETSOR$\top$ program is any function of scalars of the form

$$\frac{1}{n} \sum_{\alpha=1}^{n} \psi(h^1_\alpha, \ldots, h^k_\alpha)$$

for some function ψ and collection of H-vars $h^1, \ldots, h^k$.

Remark A.2. In comparison with NETSOR introduced in [62], the language NETSOR$\top$ defined here has additionally the transpose (`Trsp`) instruction and drops the `LinComb` instruction[12], and additionally has no output. While the lack of `LinComb` and output is just taking away some syntactic sugar, the `Trsp` instruction significantly expands the expressivity of the language. Importantly, it allows us to express backpropagation.

[10] formally, we are dealing with dependent types G and H indexed by $\mathbb{N}$ and A indexed by $\mathbb{N}^2$

[11] In general, the output of a tensor program need not be defined, as most of the time we are concerned with how the H-vars produced over the course of the program interact with each other.

[12] NETSOR$\top$ is exactly the NETSOR$^-$ language, introduced in the appendix of Yang [62], augmented with `Trsp`.

NETSOR$\top$ Program 1 MLP Forward and Backward Computation on Network Input x

Input: $W^1 x : \mathsf{G}(n^1)$ ▷ layer 1 embedding of input
Input: $b^1 : \mathsf{G}(n^1)$ ▷ layer 1 bias
Input: $W^2 : \mathsf{A}(n^2, n^1)$ ▷ layer 2 weights
Input: $b^2 : \mathsf{G}(n^2)$ ▷ layer 2 bias
Input: $v : \mathsf{G}(n^2)$ ▷ readout layer weights
$x^1 := \phi(W^1 x + b^1) : \mathsf{H}(n^1)$ ▷ layer 1 activation; `Nonlin` with $(u, v) \mapsto \phi(u + v)$
$\tilde{h}^2 := W^2 x^1 : \mathsf{G}(n^2)$ ▷ `MatMul`
$x^2 := \phi(\tilde{h}^2 + b^2) : \mathsf{H}(n^2)$ ▷ layer 2 activation; `Nonlin` with $(u, v) \mapsto \phi(u + v)$
▷ output is $v^\top x^2 / \sqrt{n^2}$, but this does not need to be expressed in the program
▷ begin backprop
$W^{2\top} := (W^2)^\top : \mathsf{A}(n^1, n^2)$ ▷ `Trsp`
$dx^2 := v : \mathsf{G}(n^2)$ ▷ gradient wrt x^2 equals the last layer weights, scaled up by $\sqrt{n^2}$
$d\tilde{h}^2 := \phi'(\tilde{h}^2 + b^2) \odot dx^2 : \mathsf{H}(n^2)$ ▷ gradient wrt $\tilde{h}^2$, scaled up by $\sqrt{n^2}$; `Nonlin`
$dx^1 := W^{2\top} d\tilde{h}^2 : \mathsf{G}(n^1)$ ▷ gradient wrt x^1, scaled up by $\sqrt{n^2}$; `MatMul`
$d(W^1 x) := \phi'(W^1 x + b^1) \odot dx^1$ ▷ gradient wrt the vector $W^1 x$, scaled up by $\sqrt{n^2}$; `Nonlin`
▷ Return the NTK value $\Theta(x, x)$; see Eq. (9)

Output: $\frac{\|x^2\|^2}{n^2} + \frac{\|d\tilde{h}^2\|^2}{n^2}\left(1 + \frac{\|x^1\|^2}{n^1}\right) + \frac{\|d(W^1 x)\|^2}{n^1}\left(1 + \frac{\|x\|^2}{\dim(x)}\right)$

NETSOR$\top$ Program 2 Simple RNN Forward and Backward Computation on Two Input Sequences

// Embeddings of sequence 1 tokens
Input: $Ux^{11}, \ldots, Ux^{T_1 1} : \mathsf{G}(n)$
// Embeddings of sequence 2 tokens
Input: $Ux^{12}, \ldots, Ux^{T_2 2} : \mathsf{G}(n)$
// Weight and bias
Input: $W : \mathsf{A}(n, n)$
Input: $b : \mathsf{G}(n)$
// Readout weights
Input: $v : \mathsf{G}(n)$
// The FOR loop is a shorthand for the unrolled straight-line program
for $a = 1, 2$ **do**
$s^{1a} := \phi(Ux^{1a} + b) : \mathsf{H}(n)$
$\tilde{h}^{2a} := W s^{1a} : \mathsf{G}(n)$
$s^{2a} := \phi(\tilde{h}^{2a} + Ux^{2a} + b) : \mathsf{H}(n)$
$\vdots$
$\tilde{h}^{T_a a} := W s^{T_a - 1, a} : \mathsf{G}(n)$
$s^{T_a a} := \phi(\tilde{h}^{T_a a} + Ux^{T_a a} + b) : \mathsf{H}(n)$
// Output is $v^\top s^{T_a a} / \sqrt{n}$, but
// we don't express this in the program
// — Backprop —
// $\forall$ variable u, du represents $\sqrt{n} \nabla_u$out
$ds^{T_a a} := v : \mathsf{G}(n)$
// ϕ' is derivative of ϕ
$d\tilde{h}^{T_a a} := \phi'(\tilde{h}^{T_a a} + Ux^{T_a a} + b) \odot ds^{T_a a} : \mathsf{H}(n)$
$ds^{T_a - 1, a} := W^\top d\tilde{h}^{T_a a} : \mathsf{G}(n)$
$d\tilde{h}^{T_a - 1, a} := \phi'(\tilde{h}^{T_a - 1, a} + Ux^{T_a - 1, a} + b) \odot ds^{T_a - 1, a} : \mathsf{H}(n)$
$ds^{T_a - 2, a} := W^\top d\tilde{h}^{T_a - 1, a} : \mathsf{G}(n)$
$\vdots$
$ds^{1a} := W^\top d\tilde{h}^{2a} : \mathsf{G}(n)$
$d\tilde{h}^{1a} := \phi'(Ux^{1a} + b) \odot ds^{1a} : \mathsf{H}(n)$
// Return NTK evaluated on sequences x^1, x^2
// See Eq. (12)
Output: $\frac{s^{T_1 1 \top} s^{T_2 2}}{n} + \sum_{i=1}^{T_1} \sum_{j=1}^{T_2} \frac{d\tilde{h}^{i1\top} d\tilde{h}^{j2}}{n} \times \left(1 + \frac{s^{i-1,1\top} s^{j-1,2}}{n} + \frac{x^{i1\top} x^{j2}}{\dim(x^{i1})}\right)$

Examples Programs 1 and 2 write out the forward and backward computation of resp. an MLP and a simple RNN. We remark on a few things: First, notice that the new transpose instruction `Trsp` allows us to express backpropagation. Second, as in Yang [62], *we account for the input x through its embedding $W^1 x$, not x itself.* This is because 1) our theorems concern the case where all input G-vars are random; in the context of expressing neural network computation, x is a deterministic input, while $W^1 x$ is a Gaussian vector when W^1 has iid Gaussian entries; 2) x has a fixed dimension, while we intend all dimensions (like n^1, n^2) in the NETSOR program to tend to infinity, as we'll describe shortly. Third, weight-sharing is easily expressed because we can arbitrarily re-use A-vars.

Programs expressing backpropagation have a special property that we would like to isolate, and which will reduce the complexity the NETSOR$\top$ master theorem we need to prove. It is a tensor program generalization of Condition 1 for neural networks.

Definition A.3. A NETSOR$\top$ program is said to be *BP-like* if there is a special nonempty set of input G-vars $v^1, \dots, v^k$ (intuitively, these should be thought of as the readout weights of the forward computation) such that

1. If $W^\top z$ is used in the program for some H-var z, and W is an input A-var, then z must be an odd function of $v^1, \dots, v^k$, in the sense that, fixing all other G-vars, if $v^1, \dots, v^k$ are negated simultaneously, then z is negated as well:
$$z(-v^1, \dots, -v^k, \text{all other G-vars}) = -z(v^1, \dots, v^k, \text{all other G-vars}).$$
2. If Wz is used in the program for some H-var z, and W is an input A-var, then z cannot depend on any of $v^1, \dots, v^k$.
3. $v^1, \dots, v^k$ are sampled with zero mean (but possibly with nontrivial covariances) and independently from all other G-vars.

Remark A.4. A NETSOR$\top$ program expressing backpropagation of a network satisfying Condition 1 can be seen to be BP-like as follows: We can always write the program such that the "forward computation", up to but before applying readout weights, appears first, followed by the "backward computation" (scaled up by $\sqrt{\text{width}}$, as exemplified by Programs 1 and 2). If the network output is a scalar out $= v^\top x/\sqrt{n}$ with readout weights v, then v is not used until the backward computation, where we have $\sqrt{n}\partial\text{out}/\partial x = v$. This fulfills condition 2 in Definition A.3. In the backward computation, only transposed matrices (A-vars) appear in `MatMul` lines, and all vectors (H-vars) are linear functions of (and thus are odd in) v, because backpropagation is linear in output gradients. This fulfills condition 1 in Definition A.3.

Like in Yang [62], the G-vars in a BP-like NETSOR$\top$ program will roughly jointly Gaussian in each coordinate slice. We keep track of their mean and covariance using the usual recursive equations.

$$\mu(g) = \begin{cases} \mu^{\text{in}}(g) & \text{if } g \text{ is input} \\ 0 & \text{otherwise} \end{cases},$$
$$\Sigma(g, \bar{g}) = \begin{cases} \Sigma^{\text{in}}(g, g') & \text{if } g, g' \text{ are inputs} \\ \sigma_W^2 \, \mathbb{E}_Z \, \phi(Z)\bar{\phi}(Z) & \text{if } g = Wh, \bar{g} = W\bar{h}, \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

Setup A.5. *For* NETSOR$\top$ *program: For simplicity, assume all dimensions in the program are equal to n. Suppose for each A-var $W : \mathsf{A}(n, n)$, we sample $W_{\alpha\beta} \sim \mathcal{N}(0, \sigma_W^2/n)$ for some $\sigma_W^2 > 0$, and for each $\alpha \in [n]$, we sample, i.i.d., $\{x_\alpha : x \text{ is input G-var}\} \sim \mathcal{N}(\mu^{\text{in}}, \Sigma^{\text{in}})$ for some mean μ^{in} and (possibly singular) covariance Σ^{in} over input G-vars.*

Finally, we have the BP-like NETSOR$\top$ Master theorem.

Theorem A.6 (BP-like NETSOR$\top$ Master Theorem)**.** *Fix any BP-like* NETSOR$\top$ *program satisfying Setup A.5 and with all nonlinearities polynomially-bounded. If $g^1, \dots, g^M$ are all of the G-vars in the entire program, including all input G-vars, then for any polynomially-bounded $\psi : \mathbb{R}^M \to \mathbb{R}$, as $n \to \infty$,*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z) = \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z^{g^1}, \dots, Z^{g^M}),$$

where $\xrightarrow{\text{a.s.}}$ means almost sure convergence, $Z = (Z^{g^1}, \dots, Z^{g^M}) \in \mathbb{R}^M$, and $\mu = \{\mu(g^i)\}_{i=1}^M \in \mathbb{R}^M$ and $\Sigma = \{\Sigma(g^i, g^j)\}_{i,j=1}^M \in \mathbb{R}^{M\times M}$ are given in Eq. (18). See Fig. 1 for an illustration.

This is equivalent to Theorem 7.2, but emphasizes that the main source of randomness comes from the G-vars (i.e. approximately Gaussian vectors) $g^1, \dots, g^M$ whose distribution can be explicitly computed via Eq. (18).

B NETSOR$\top^+$

In this section we augment NETSOR$\top$ with a constant type to form NETSOR$\top^+$, like how NETSOR is augmented likewise to form self-parametrized NETSOR$^+$ in Yang [62]. The main result here is the *BP-like* NETSOR$\top^+$ *Master Theorem* (Theorem B.2).

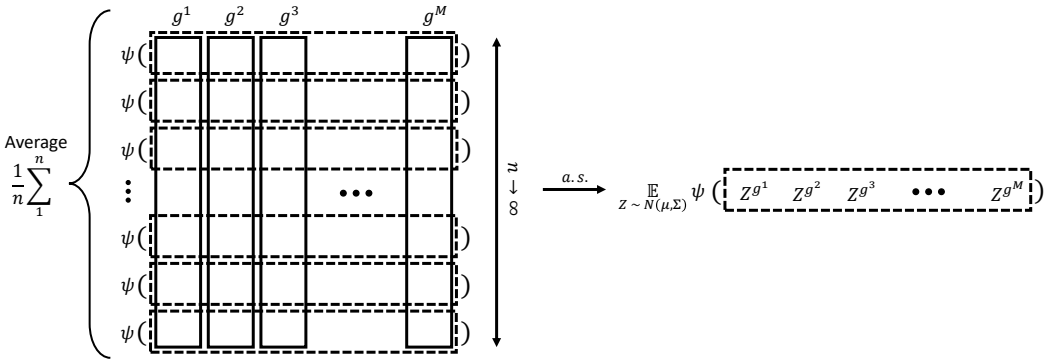


Figure 1: An illustration of the NETSOR$\top$ Master Theorem Theorem A.6.

We first formally define the NETSOR$\top^+$ language.

Definition B.1. A NETSOR$\top^+$ *program*[13] is a NETSOR$\top$ program where we have an additional scalar type, called C, which should intuitively be thought of as a random variable that tends to a deterministic limit (i.e. a C*onstant*) almost surely. Colloquially, we will call variables of type C "C-vars." C-vars can be used as parameters of nonlinearities.

For completeness, we specify a NETSOR$^+$ program as follows:

Input A set of input C-vars, in addition to the G- and A-vars allowed in Definition A.1.

Body New variables can be introduced and assigned via the following rules

`MatMul` Same as in Definition A.1.

`Trsp` Same as in Definition A.1.

`Nonlin`$^+$ If $x^1, \dots, x^k : \mathsf{G}(n)$ are G-vars with the same dimension n, $\theta_1, \dots, \theta_l : \mathsf{C}$ are C-vars, and $\phi(-;-) : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$ is a parametrized function, then we may create an H-var

$$\phi(x^1, \dots, x^k; \theta_1, \dots, \theta_l) : \mathsf{H}(n)$$

where $\phi(-; \theta_1, \dots, \theta_l)$ acts coordinatewise.

`Moment` If $x^1, \dots, x^k : \mathsf{G}(n)$ are G-vars with the same dimension n, $\theta_1, \dots, \theta_l : \mathsf{C}$ are C-vars, and $\phi(-;-) : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$ is a parametrized function, then we may create a C-var

$$\frac{1}{n}\sum_{\alpha=1}^{n} \phi(x^1_\alpha, \dots, x^k_\alpha; \theta_1, \dots, \theta_l) : \mathsf{C}.$$

Output Same as in Definition A.1.

See Appendix D for examples of layernorm and attention written in NETSOR$\top^+$.

The following gives the NETSOR$\top^+$ Master Theorem, which first lists the regularity conditions needed (*rank stability* and *parameter-control*), along with some natural notations defined later, before stating the main convergence results.

Theorem B.2 (BP-like NETSOR$\top^+$ Master Theorem)**.** *Fix any BP-like* NETSOR$\top^+$ *program sampled in the natural way as in Assumption B.3 and also satisfying rank stability (Assumption B.7). Let $\varphi^\bullet, \Theta^\bullet, \mu, \Sigma, (\mathring{\ })$ be as in Definition B.4. Suppose for every H-var or C-var u, $\varphi^u(-; \Theta^u)$ is parameter-controlled at $\mathring{\Theta}^u$. Then the following hold.*

[13] In the language of Yang [62], NETSOR$\top^+$ actually corresponds to self-parametrized NETSOR$^+$ with `Trsp`, but we omit "self-parametrized" in the name because this is the primary form of NETSOR$\top^+$ we will use in practice. What would be the plain NETSOR$\top^+$ in the language of Yang [62] would just be a fragment of of the NETSOR$\top^+$ in Definition B.1, so our Master Theorem will also cover that case.

1. *Let $g^1, \dots, g^M$ be all of the G-vars in the program (including all input G-vars). Then for any polynomially bounded $\psi : \mathbb{R}^M \to \mathbb{R}$, we have*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z).$$

More generally, for any l, for any random vector $\Theta \in \mathbb{R}^l$ that converges almost surely to a deterministic vector $\mathring{\Theta}$, as $n \to \infty$, and for any $\psi(-;-) : \mathbb{R}^M \times \mathbb{R}^l \to \mathbb{R}$ parameter-controlled at $\mathring{\Theta}$,

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M; \Theta) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z; \mathring{\Theta}).$$

2. *Each C-var θ converges to its natural limit $\mathring{\theta}$:*

$$\theta \xrightarrow{\text{a.s.}} \mathring{\theta}.$$

We now expand on the assumptions and definitions used in the Master Theorem.

Assumption B.3. *Fix a self-parametrized* NETSOR$\top^+$ *program satisfying Setup A.5. Assume each input C-var θ is sampled in a way such that $\theta \xrightarrow{\text{a.s.}} \mathring{\theta}$ as $n \to \infty$ for some deterministic scalar $\mathring{\theta} \in \mathbb{R}$.*

Definition B.4. Fix a NETSOR$\top^+$ program with scalar variables satisfying Assumption B.3. For the purpose of this definition, write $g^1, \dots, g^M$ for the entirety of the G-vars in the program, including input G-vars.

New Notations: $\varphi^\bullet, \Theta^\bullet$ For each H-var $h = \phi(g^1, \dots, g^M; \theta_1, \dots, \theta_l)$ introduced by `Nonlin`$^+$, set $\varphi^h \stackrel{\text{def}}{=} \phi$ and $\Theta^h \stackrel{\text{def}}{=} (\theta_1, \dots, \theta_l)$. For each G-var g^i, this means that $\varphi^{g^i}(x^1, \dots, x^M) = x^i$ and $\Theta^{g^i} = () \in \mathbb{R}^0$ is the empty vector. Likewise, for each C-var $c = \frac{1}{n}\sum_{\alpha=1}^n \phi(g_\alpha^1, \dots, g_\alpha^M; \theta_1, \dots, \theta_l)$ introduced by `Moment`, set $\varphi^c \stackrel{\text{def}}{=} \phi$ and $\Theta^c \stackrel{\text{def}}{=} (\theta_1, \dots, \theta_l)$.

Extending the $\mathring{(\ )}$ notation from Assumption B.3 and the Recursive Definition of μ and Σ Given μ^{in} and Σ^{in} as in Setup A.5, we define μ and Σ on G-vars, along with "limit scalars" $\mathring{\theta}$ for each C-var θ (extending $\mathring{\theta}$ given by Assumption B.3 for input θ), as follows: For any pair of G-vars $g, \bar{g}$, we define recursively

$$\mu(g) \stackrel{\text{def}}{=} \begin{cases} \mu^{\text{in}}(g) & \text{if } g \text{ is input} \\ 0 & \text{otherwise} \end{cases}$$

$$\Sigma(g,\bar{g}) \stackrel{\text{def}}{=} \begin{cases} \Sigma^{\text{in}}(g,\bar{g}) & \text{if } g, g' \text{ are inputs} \\ \sigma_W^2 \mathbb{E}_Z \varphi^h(Z; \mathring{\Theta}^h)\varphi^{\bar{h}}(Z; \mathring{\Theta}^{\bar{h}}) & \text{if } g = Wh, \bar{g} = W\bar{h} \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

where W is any A-var, transposed or not[14]; and for each C-var θ introduced by `Moment`,

$$\mathring{\theta} \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{Z} \varphi^\theta(Z; \mathring{\Theta}^\theta). \tag{20}$$

In all of the equations above, $Z \sim \mathcal{N}(\mu, \Sigma)$ is a random Gaussian vector with an entry for each G-var in the program.

Parameter-Control We adapt the definition of *parameter-control* from Yang [62] to our setting.

Definition B.5. We say a parametrized function $\phi(-;-) : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$ is *polynomially parameter-controlled* or just *parameter-controlled* for short[15] , at $\mathring{\Theta} \in \mathbb{R}^l$ if

[14] In Eq. (19), if W is an input A-var, then σ_W is as in Setup A.5; if $W = \bar{W}^\top$ for some input G-var $\bar{W}$, then $\sigma_W = \sigma_{\bar{W}}$. Note: when we allow variable dimensions in the program, this is defined slightly differently; see Eq. (24).

[15] This overloads the meaning of *parameter-controlled* from Yang [62], where the definition replaces the "polynomially bounded" in the definition here with "bounded by $e^{C\|\cdot\|^{2-\epsilon}+c}$ for some $C, c, \epsilon > 0$." In this paper, we shall never be concerned with the latter (more generous) notion of boundedness, so there should be no risk of confusion.

1. $\phi(-; \mathring{\Theta})$ is polynomially bounded, and
2. there are some polynomially bounded $\bar{\phi} : \mathbb{R}^k \to \mathbb{R}$ and some function $f : \mathbb{R}^l \to \mathbb{R}^{\geq 0} \cup \{\infty\}$ that has $f(\mathring{\Theta}) = 0$ and that is continuous at $\mathring{\Theta}$, such that, for all $x^1, \ldots, x^k \in \mathbb{R}$ and $\Theta \in \mathbb{R}^l$,

$$|\phi(x^1, \ldots, x^k; \Theta) - \phi(x^1, \ldots, x^k; \mathring{\Theta})| \leq f(\Theta)\bar{\phi}(x^1, \ldots, x^k).$$

Note that f and $\bar{\phi}$ here can depend on $\mathring{\Theta}$. The following examples come from Yang [62].

Example B.6. Any function that is (pseudo-)Lipschitz[16] in $x^1, \ldots, x^k$ and Θ is polynomially parameter-controlled. An example of a discontinuous function that is polynomially parameter-controlled is $\phi(x; \theta) = \text{step}(\theta x)$. Then for $\mathring{\theta} \neq 0$,

$$|\phi(x; \theta) - \phi(x; \mathring{\theta})| \leq \frac{|\mathring{\theta} - \theta|}{|\mathring{\theta}|},$$

so we can set $f(\theta) = \frac{|\mathring{\theta} - \theta|}{|\mathring{\theta}|}$ and $\bar{\phi} = 1$ in Definition B.5.

Rank Stability The following assumption says that the vectors in a program should not change any linear dependence relations abruptly in the infinite n limit.

Assumption B.7 (Rank Stability)**.** *For any $W : \mathsf{A}(n, m)$ and any collection $\mathcal{S} \subseteq \{(h : \mathsf{H}(m)) \mid \exists (g : \mathsf{G}(n)), g := Wh\}$, let $H \in \mathbb{R}^{m \times |\mathcal{S}|}$ be the matrix whose columns are $h \in \mathcal{S}$. If $\frac{1}{m} H^\top H \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ converges almost surely to some $\mathring{C}$ as $n, m \to \infty$ with convergent ratio $n/m \to \alpha$, then almost surely $\operatorname{rank} H = \operatorname{rank} \mathring{C}$ for all large n and m.*

Some remarks

- An example violating rank stability would be $\mathcal{S} = \{1, 1 + 1/n\}$, vectors with constant entries 1 and $1 + 1/n$, which are linearly independent for any finite n but their kernel matrix becomes singular in the limit $n \to \infty$.
- Note that a common situation where rank stability holds is when all limit $\mathring{C}$ matrices are full rank. By the lower semi-continuity of rank, $\operatorname{rank} H = \operatorname{rank} \mathring{C}$ must hold asymptotically.
- There are counterexamples to the NETSOR$\top^+$ Master Theorem if rank stability is not assumed [62].
- Note that we did not assume Assumption B.7 explicitly in Theorems 7.2 and A.6 because we in fact get it for free (Lemma G.6).

See Yang [62] for further discussions of the rank stability assumption.

B.1 The Simplified NETSOR$\top^+$

We can simplify the above formal description of NETSOR$\top^+$ like how we simplified NETSOR$\top$ in the main text.

Definition B.8 (Simplified NETSOR$\top^+$)**.** A simplified NETSOR$\top^+$ program is just a sequence of vectors and scalars recursively generated from an initial set of random $n \times n$ matrices $\mathcal{W}$, random size n vectors $\mathcal{V}$, and random scalars $\mathcal{C}$ via one of the following ways

Nonlin Given $\phi : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$, previous scalars $\theta_1, \ldots, \theta_l \in \mathbb{R}$ and vectors $x^1, \ldots, x^k \in \mathbb{R}^n$, we can generate a new vector

$$\phi(x^1, \ldots, x^k; \theta_1, \ldots, \theta_l) \in \mathbb{R}^n$$

where $\phi(-; \theta_1, \ldots, \theta_l)$ applies coordinatewise to each α-slice $(x^1_\alpha, \ldots, x^k_\alpha)$.

[16] A pseudo-Lipschitz function $\phi : \mathbb{R}^r \to \mathbb{R}$ is one that satisfies

$$|\phi(x) - \phi(y)| \leq C\|x - y\|(\|x\|^p + \|y\|^q + 1)$$

for some constants $C, p, q \geq 0$. Roughly speaking, pseudo-Lipschitz functions are those that have polynomially bounded weak derivatives.

Moment Given same setup as above, we can also generate a new scalar

$$\frac{1}{n}\sum_{\alpha=1}^{n}\phi(x_\alpha^1,\dots,x_\alpha^k;\theta_1,\dots,\theta_l)\in\mathbb{R}$$

MatMul Given $W\in\mathbb{R}^{n\times n}$ and $x\in\mathbb{R}^n$, we can generate $Wx\in\mathbb{R}^n$ or $W^\top x\in\mathbb{R}^n$

Here the (initial) matrices, vectors, and scalars correspond to the (input) A-, H-, and C-vars in Definition B.1, and the rules with the same name mirror one another. The main difference between this version of NETSOR$\top^+$ and Definition B.1 is that we are implicitly allowing **Nonlin** and **Moment** to take in H-vars here instead of only G-vars. Nevertheless, their expressive powers are equivalent, since any vector generated with Definition B.8 is generated from a chain of **Nonlin** that ends up in G-vars (those vectors created by **MatMul**), and we can just collapse parametrized nonlinearities into a single parametrized nonlinearity that takes in G-vars only. For example, if $z=\phi(x^1,x^2;\theta_1), x^1=Wv, x^2=\psi(y;\theta_2), y=Wu$, then z can be directly expressed in terms of G-vars: $z=\overline{\phi}(Wv,Wu;\theta_1,\theta_2)\overset{\text{def}}{=}\phi(Wv,\psi(Wu;\theta_2);\theta_1)$. Therefore, we make the following definition

Definition B.9 ($\varphi^\bullet,\Theta^\bullet$ Notation)**.** For any size n vector x in Definition B.1, let φ^x and Θ^x be the parametrized nonlinearity and the scalars such that $x=\varphi^x(z^1,\dots,z^k;\Theta^x)$ for some G-vars $z^1,\dots,z^k$. Likewise, for any scalar c in Definition B.1, let φ^c and Θ^c be the parametrized nonlinearity and the scalars such that $c=\frac{1}{n}\sum_{\alpha=1}^n\varphi^x(z_\alpha^1,\dots,z_\alpha^k;\Theta^c)$ for some G-vars $z^1,\dots,z^k$.

Then we can write down a Master Theorem in the style of Theorem 7.2 for Definition B.8-style NETSOR$\top^+$ programs.

Box 2 How to Intuitively Understand a Simplified NETSOR$\top^+$ Program

Consider a NETSOR$\top^+$ program sampled as in Theorem B.10. When $n\gg 1$, each vector $x\in\mathbb{R}^n$ in the program has roughly iid coordinates distributed like a random variable Z^x, and each scalar $\theta\in\mathbb{R}$ is close to a deterministic scalar $\mathring{\theta}$, with Z^x and $\mathring{\theta}$ defined recursively as below.

Nonlin If $y=\phi(x^1,\dots,x^k;\theta_1,\dots,\theta_l)$, then

$$Z^y=\phi(Z^{x^1},\dots,Z^{x^k};\mathring{\theta}_1,\dots,\mathring{\theta}_l).$$

Moment If $\theta=\frac{1}{n}\sum_{\alpha=1}^n\phi(x_\alpha^1,\dots,x_\alpha^k;\theta_1,\dots,\theta_l)$, then

$$\mathring{\theta}=\mathop{\mathbb{E}}_Z\phi(Z^{x^1},\dots,Z^{x^k};\mathring{\theta}_1,\dots,\mathring{\theta}_l).$$

MatMul For any set of infinite vectors $\mathcal{X}$ and matrix $W\in\mathcal{W}$, the set of random variables $\{Z^{Wx}:x\in\mathcal{X}\}$ is jointly Gaussian with zero mean and covariance

$$\operatorname{Cov}\left(Z^{Wx},Z^{W\bar{x}}\right)=\sigma_W^2\,\mathbb{E}\,Z^xZ^{\bar{x}}.$$

If $\mathcal{Y}$ is any set of $\mathbb{R}^n$ vectors and $\bar{W}\neq W$, then $\{Z^{Wx}:x\in\mathcal{X}\}$ is independent from $\{Z^{\bar{W}y}:y\in\mathcal{Y}\}$.

Theorem B.10 (Simplified BP-like NETSOR$\top$ Master Theorem)**.** *Consider a* NETSOR$\top^+$ *program in the style of Definition B.8. Suppose: 1) for each initial $W\in\mathcal{W}$, $W_{\alpha\beta}\sim\mathcal{N}(0,\sigma_W^2/n)$ for an associated variance σ_W^2; 2) there is a multivariate Gaussian $Z^{\mathcal{V}}=\{Z^g:g\in\mathcal{V}\}\in\mathbb{R}^{|\mathcal{V}|}$ such that the initial set of vectors $\mathcal{V}$ are sampled like $\{g_\alpha:g\in\mathcal{V}\}\sim Z^{\mathcal{V}}$ iid for each $\alpha\in[n]$; 3) each initial scalar θ tends to a deterministic constant $\mathring{\theta}$ as $n\to\infty$. Suppose the program is BP-like and $\varphi^u(-;-)$ is parameter-controlled at $\mathring{\Theta}^u$ for all vectors and scalars u. Assume the program satisfies rank stability (Assumption B.7).*

Recursively define Z^h for each vector h and $\mathring{\theta}$ for each scalar θ in the program as in Box 2. Then the following hold.

1. *For any l, for any random vector $\Theta \in \mathbb{R}^l$ that converges almost surely to a deterministic vector $\mathring{\Theta}$ as $n \to \infty$, for any vectors $x^1, \dots, x^M \in \mathbb{R}^n$ in the program, and for any $\psi(-;-) : \mathbb{R}^M \times \mathbb{R}^l \to \mathbb{R}$ parameter-controlled at $\mathring{\Theta}$,*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(x^1_\alpha, \dots, x^M_\alpha; \Theta) \xrightarrow{\text{a.s.}} \mathbb{E}\, \psi(Z^{x^1}, \dots, Z^{x^M}; \mathring{\Theta}).$$

2. *Each C-var θ converges to its natural limit $\mathring{\theta}$:*

$$\theta \xrightarrow{\text{a.s.}} \mathring{\theta}.$$

C Programs with Variable Dimensions

Notation In this section, we let $\dim(x)$ denote the dimension of an H-var x.

As in Yang [62], we focused in the main text on the case where all dimensions of vectors in a NETSOR$\top$ program are equal, but this does not have to be the case. In this section, we state the Master Theorem for BP-like NETSOR$\top$ and NETSOR$\top^+$ programs with variable dimensions. The main idea is exactly the same as before, but we require some more notation to describe how the limit is taken when all the widths vary.

First, there are some obvious dimensionality constraints even when they do vary, induced by the rules we apply to introduce variables:

$$\begin{cases} \text{If } y = \phi(x^1, \dots, x^k), \text{ then } \dim(y) = \dim(x^i), \forall i; \text{ similarly for } \texttt{Nonlin}^+ \text{ and } \texttt{Moment}. \\ \text{If } y = Wx \text{ and } \bar{y} = W\bar{x}, \text{ then } \dim(x) = \dim(\bar{x}) \text{ and } \dim(y) = \dim(\bar{y}). \end{cases} \tag{21}$$

Definition C.1. Given an equivalence relation $\simeq$ on the input G-vars of a program, we extend this to an equivalence relation on all H-vars of the program by

$$h \equiv h' \iff h \simeq h' \text{ OR } h \text{ and } h' \text{ are constrained to have the same dimension by (21).} \tag{22}$$

We call any such equivalence class a *Common Dimension Class*, or CDC.

Intuitively, the dimensions of H-vars in each CDC are all the same but can be different in different CDCs.

Example C.2. In Program 1, if we let different layers have different widths, then the CDCs are $\{W^1x, b^1, x^1, dx^1, d(W^1x)\}$ and $\{b^2, v, \tilde{h}^2, x^2, dx^2, d\tilde{h}^2\}$. If we tie the widths, then all of these H-vars are in the same CDC. In Program 2, all G-vars are in the same CDC, and given the body of the program, this is the only way to partition the H-vars into CDCs, because the reuse of W across time step ties all H-var dimensions to be equal.

The following describes the sampling and CDCs of NETSOR$\top$ programs we are interested in.

Assumption C.3. *Fix a* NETSOR$\top$ *or* NETSOR$\top^+$ *program with some equivalence relation on the input G-vars, and thus with induced CDCs over its H-vars. Assume the dimensions in each CDC are the same, but the dimensions of different CDCs can vary. Suppose for each input A-var $W : \mathsf{A}(m', m)$, we sample $W_{\alpha\beta} \sim \mathcal{N}(\sigma_W^2/m)$ for some $\sigma_W^2 > 0$. For each transpose A-var $W^\top : \mathsf{A}(m, m')$, we also set $\sigma_{W^\top}^2 = \frac{m'}{m}\sigma_W^2$. Suppose further for each CDC $\mathfrak{c}$ with dimension n, for each $\alpha \in [n]$, we sample, i.i.d., $\{x_\alpha : x \in \mathfrak{c} \text{ and } x \text{ is input G-var}\} \sim \mathcal{N}(\mu^{\mathfrak{c}}, \Sigma^{\mathfrak{c}})$ for some mean $\mu^{\mathfrak{c}}$ and covariance $\Sigma^{\mathfrak{c}}$ over input G-vars in $\mathfrak{c}$.*

Then the following result is an easy extension of Theorem A.6.

Theorem C.4 (BP-like NETSOR$\top$ Master Theorem; Variable Dimensions)**.** *Fix any BP-like* NETSOR$\top$ *program satisfying Assumption C.3 and with all nonlinearities polynomially bounded. Consider the limit where all dimensions go to infinity, with their pairwise ratios tending to finite but nonzero values:*

$$\forall W : \mathsf{A}(n', n), \quad \textit{we have} \quad n'/n \to \rho \quad \textit{for some } \rho \in (0, \infty).$$

Then for each transpose $W^\top$ of an input A-var $W : \mathsf{A}(n', n)$, we have

$$\sigma_{W^\top}^2 = \frac{n'}{n}\sigma_W^2 \to \mathring{\sigma}_{W^\top}^2$$

for some limit $\mathring{\sigma}^2_{W^\top}$. For each input A-var W, we also set $\mathring{\sigma}_W = \sigma_W$.

For any CDC $\mathfrak{c}$, if $g^1, \dots, g^M : \mathsf{G}(n)$ are all of the G-vars in $\mathfrak{c}$ (including all input G-vars), then for any polynomially bounded $\psi : \mathbb{R}^M \to \mathbb{R}$, as all dimensions in the program tend to infinity (not just the dimension of $\mathfrak{c}$) in the manner above, we have

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g^1_\alpha, \dots, g^M_\alpha) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu^{\mathfrak{c}},\Sigma^{\mathfrak{c}})} \psi(Z) = \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu^{\mathfrak{c}},\Sigma^{\mathfrak{c}})} \psi(Z^{g^1}, \dots, Z^{g^M}), \tag{23}$$

where $\xrightarrow{\text{a.s.}}$ means almost sure convergence, $Z = (Z^{g^1}, \dots, Z^{g^M}) \in \mathbb{R}^M$, and $\mu^{\mathfrak{c}} = \{\mu^{\mathfrak{c}}(g^i)\}_{i=1}^M \in \mathbb{R}^M$ and $\Sigma^{\mathfrak{c}} = \{\Sigma^{\mathfrak{c}}(g^i, g^j)\}_{i,j=1}^M \in \mathbb{R}^{M\times M}$ are given in Eq. (24).

The mean $\mu^{\mathfrak{c}}$ and covariance $\Sigma^{\mathfrak{c}}$ used in Theorem C.4 are defined as follows.

Definition C.5. For any CDC $\mathfrak{c}$ and G-vars $g, \bar{g}$ in $\mathfrak{c}$, define recursively

$$\mu^{\mathfrak{c}}(g) = \begin{cases} \mu^{\mathfrak{c}}(g) & \text{if } g \text{ is input} \\ 0 & \text{otherwise} \end{cases},$$

$$\Sigma^{\mathfrak{c}}(g, \bar{g}) = \begin{cases} \Sigma^{\mathfrak{c}}(g, \bar{g}) & \text{if } g, \bar{g} \text{ are inputs} \\ \mathring{\sigma}^2_W \mathbb{E}_Z \varphi^h(Z)\varphi^{\bar{h}}(Z) & \text{if } g = Wh, \bar{g} = W\bar{h} \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

where W can be either an input or a transposed A-var (with $\mathring{\sigma}^2_W$ defined in Theorem C.4) and $Z \sim \mathcal{N}(\mu^{\mathfrak{c}'}, \Sigma^{\mathfrak{c}'})$ with $\mathfrak{c}'$ denoting the CDC of h and $\bar{h}$.

Proof. Trivial adaptation of the proof of Theorem A.6. □

Remark C.6. Eq. (23) only concerns G-vars of the same CDC; what about G-vars of different CDCs? One can quickly see that a convergence statement like in Eq. (23) cannot be made naively just because the dimensions are not equal: we cannot even write down the "empirical average" that should converge. In fact, one can intuitively think of vectors of different CDCs as roughly "independent" because their "main source of randomness" must come from different A-vars.

Likewise we can extend the BP-like NETSOR$\top^+$ Master Theorem to the Variable Dimension case.

Theorem C.7 (BP-like NETSOR$\top^+$ Master Theorem; Variable Dimensions)**.** *Fix any BP-like NETSOR$\top^+$ program sampled in the natural way as in Assumptions B.3 and C.3 and also satisfying rank stability (Assumption B.7). Let $\varphi^\bullet, \Theta^\bullet$ be as in Definition B.4. Suppose for every H-var or C-var u, $\varphi^u(-; \Theta^u)$ is parameter-controlled at $\mathring{\Theta}^u$.*

Consider the limit where all dimensions go to infinity, with their pairwise ratios tending to finite but nonzero values:

$$\text{for all } W : \mathsf{A}(n', n), \quad \text{we have} \quad n'/n \to \rho \quad \text{for some } \rho \in (0, \infty).$$

Then for each transpose $W^\top$ of an input A-var $W : \mathsf{A}(n', n)$, we have

$$\sigma^2_{W^\top} = \frac{n'}{n}\sigma^2_W \to \mathring{\sigma}^2_{W^\top}$$

for some limit $\mathring{\sigma}^2_{W^\top}$. For each input A-var W, we also set $\mathring{\sigma}_W = \sigma_W$.

For any pair of G-vars $g, \bar{g}$, we define recursively

$$\mu(g) \overset{\text{def}}{=} \begin{cases} \mu^{\text{in}}(g) & \textit{if } g \textit{ is input} \\ 0 & \textit{otherwise} \end{cases}$$

$$\Sigma(g, \bar{g}) \overset{\text{def}}{=} \begin{cases} \Sigma^{\text{in}}(g, \bar{g}) & \textit{if } g, g' \textit{ are inputs} \\ \mathring{\sigma}^2_W \mathbb{E}_Z \varphi^h(Z; \mathring{\Theta}^h)\varphi^{\bar{h}}(Z; \mathring{\Theta}^{\bar{h}}) & \textit{if } g = Wh, \bar{g} = W\bar{h} \\ 0 & \textit{otherwise} \end{cases}$$

where W is any A-var, transposed or not; and for each C-var θ introduced by `Moment`*,*

$$\mathring{\theta} \overset{\text{def}}{=} \mathop{\mathbb{E}}_Z \varphi^\theta(Z; \mathring{\Theta}^\theta). \tag{25}$$

Then the following hold.

1. *For any CDC* $\mathfrak{c}$*, let* $g^1, \dots, g^M$ *be all of the G-vars in* $\mathfrak{c}$ *(including all input G-vars). Then for any polynomially bounded* $\psi : \mathbb{R}^M \to \mathbb{R}$*, we have*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu^{\mathfrak{c}},\Sigma^{\mathfrak{c}})} \psi(Z).$$

More generally, for any l*, for any random vector* $\Theta \in \mathbb{R}^l$ *that converges almost surely to a deterministic vector* $\mathring{\Theta}$*, as* $n \to \infty$*, and for any* $\psi(-;-) : \mathbb{R}^M \times \mathbb{R}^l \to \mathbb{R}$ *parameter-controlled at* $\mathring{\Theta}$*,*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M; \Theta) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu^{\mathfrak{c}},\Sigma^{\mathfrak{c}})} \psi(Z; \mathring{\Theta}).$$

2. *Each C-var* θ *converges to its natural limit* $\mathring{\theta}$*:*

$$\theta \xrightarrow{\text{a.s.}} \mathring{\theta}.$$

D Writing Backpropagation of Standard Architectures in NETSOR$\top$

In general, one can observe that, if the forward propagation can be written down in NETSOR (which can be done for all standard architectures as noted in Yang [62]), then the backprop can be written down in NETSOR$\top$. Here we give some explicit examples of this.

Notation If $x \in \mathbb{R}^n$ is an (pre-)activation vector, then dx denotes the gradient of the network output at x.

Dense Matrix Multiplication If $y = Wx$, then $dx = W^\top dy$.

Skip Connection If $z = x + y$, then $dx = dy = dz$.

(Graph) Convolution A convolution can be decomposed as a sum of many weight-shared dense matrix multiplications, as observed in Yang [62]. Combining the above, we can also express convolution and its backpropagation in NETSOR$\top$.

Let $x = \{x_s \in \mathbb{R}^n : s \in P\}$ be the feature maps of a convolutional neural network, where n is the number of channels, P is the set of pixel positions (e.g. $P = [32] \times [32]$), and x_s is the vector of activations at pixel s across all channels. A convolutional layer is given by a set of weights $W = \{W_\kappa \in \mathbb{R}^{n\times n} : \kappa \in K\}$: a dense (#channel-by-#channel) matrix W_κ for each kernel position $\kappa \in K$ (e.g. $K = \{-1, 0, 1\} \times \{-1, 0, 1\}$ for a 3×3 kernel or $K = \{-2, 0, 2\} \times \{-2, 0, 2\}$ for the same with dilation 2), where for simplicity we assume n is also the number of output channels. Assume $W_\kappa \sim \mathcal{N}(0, \sigma_w^2/n)$. A kernel position κ acts on a pixel position s to obtain another pixel position $s + \kappa$. The convolution of x by W (that maintains the pixel positions) can then be written via a combination of **MatMul** and **Nonlin** as $\{h_s \in \mathbb{R}^n : s \in P\}$ where

$$h_s = \sum_\kappa W_\kappa x_{s+\kappa}$$

where the range of κ depends on the padding property of the convolution. Here we will assume the sum ranges over all κ such that $s + \kappa \in P$, which corresponds to the most common zero padding.

Similarly, during backpropagation, given gradients $\{dh_t \in \mathbb{R}^n : t \in P\}$, the gradients $\{dx_t \in \mathbb{R}^n : t \in P\}$ can be computed by

$$dx_t = \sum_\kappa W_\kappa^\top dh_{t-\kappa}$$

where again the sum is over κ such that $t + \kappa \in P$. Both equations are valid NETSOR$\top$ snippets and we may form the associated random variables by

$$Z^{h_s} = \sum_\kappa Z^{W_\kappa x_{s+\kappa}}$$

$$Z^{dx_t} = \sum_\kappa Z^{W_\kappa^\top dh_{t-\kappa}}$$

where each $Z^{W_\kappa x_{s+\kappa}}, Z^{W_\kappa^\top dh_{t-\kappa}}$ is Gaussian.

Let $\bar{x}$ be any second set of input feature maps (possibly $x = \bar{x}$), and $\bar{h}$ is the convolution of W with $\bar{x}$. Then $\{Z^{h_s}\}_s \cup \{Z^{\bar{h}_s}\}_s$ are jointly Gaussian, with

$$\mathbb{E}\, Z^{h_s} Z^{\bar{h}_t} = \sum_{\kappa,\tau} \mathbb{E}\, Z^{W_\kappa x_{s+\kappa}} Z^{W_\tau \bar{x}_{t+\tau}}.$$

But by Rule 2,

$$\mathbb{E}\, Z^{W_\kappa x_{s+\kappa}} Z^{W_\tau \bar{x}_{t+\tau}} = \begin{cases} 0 & \text{if } \kappa \neq \tau \\ \sigma_w^2\, \mathbb{E}\, Z^{x_{s+\kappa}} Z^{\bar{x}_{t+\kappa}} & \text{if } \kappa = \tau \end{cases}$$

so we can simplify

$$\mathbb{E}\, Z^{h_s} Z^{\bar{h}_t} = \sigma_w^2 \sum_{\kappa} \mathbb{E}\, Z^{x_{s+\kappa}} Z^{\bar{x}_{t+\kappa}}. \tag{26}$$

Similarly, $\{Z^{dx_t}\}_t \cup \{Z^{d\bar{x}_t}\}_t$ is jointly Gaussian with

$$\mathbb{E}\, Z^{dx_s} Z^{d\bar{x}_t} = \sigma_w^2 \sum_{\kappa} \mathbb{E}\, Z^{dh_{s+\kappa}} Z^{d\bar{h}_{t+\kappa}}. \tag{27}$$

If we apply nonlinearity to h, then the usual V-transform calculations apply. This routine can be easily generalized to different strides, paddings, dilations, and also to graph convolutions.

Pooling Continuing the notation from convolution above, global average pooling (GAP) can be expressed via **Nonlin** as

$$\mathrm{GAP}(x) = \frac{1}{|P|} \sum_{s \in P} x_s \in \mathbb{R}^n.$$

Likewise, (local) maxpool with kernel positions K can be expressed via **Nonlin** as

$$\mathrm{Maxpool}(x)_s = \max\{x_{s+\kappa} : \kappa \in K, s + \kappa \in P\} \in \mathbb{R}^n,$$

where $\max$ is applied coordinatewise.

Batchnorm and Pooling For $\epsilon > 0$[17], $\zeta = (\zeta^1, \ldots, \zeta^B) \in \mathbb{R}^B$, let $\tilde{\phi} : \mathbb{R}^B \to \mathbb{R}^B$,

$$\tilde{\phi}(\zeta) \overset{\text{def}}{=} \phi\left(\tilde{\zeta}\right), \tilde{\zeta} \overset{\text{def}}{=} \frac{\hat{\zeta}}{\sigma(\hat{\zeta})}, \hat{\zeta} \overset{\text{def}}{=} \zeta - \nu(\zeta) \quad \text{where} \quad \nu(\zeta) \overset{\text{def}}{=} \frac{1}{B}\sum_{i=1}^B \zeta^i, \quad \sigma(\hat{\zeta})^2 \overset{\text{def}}{=} \frac{1}{B}\|\hat{\zeta}\|^2 + \epsilon, \tag{28}$$

be batchnorm followed by coordinatewise nonlinearity ϕ, where $\zeta \in \mathbb{R}^B$ should be interpreted as a single neuron across a batch, and ν and σ are the *batch* mean and standard deviations. Here, B should be thought of as fixed while $n \to \infty$.

If $\gamma = \tilde{\phi}(\zeta) \in \mathbb{R}^B$, and $d\gamma \in \mathbb{R}^B$ is a gradient of some loss wrt γ, then the gradient wrt $d\zeta \in \mathbb{R}^B$ can be written as

$$d\zeta = d\tilde{\phi}(d\gamma \mid \zeta) \overset{\text{def}}{=} \left(I - \frac{1}{B}\right)\left(I - \frac{\tilde{\zeta}\tilde{\zeta}^\top}{B}\right)\frac{d\gamma \odot \phi'(\tilde{\zeta})}{\sigma(\hat{\zeta})}.$$

Then, given a batch of vectors $z^1, \ldots, z^B \in \mathbb{R}^n$ (for example, they could be the preactivations after applying a linear layer), we can express batchnorm via **Nonlin** as coordinatewise applications of $\tilde{\phi}$:

$$y^i := \tilde{\phi}_i(z^1, \ldots, z^B) \in \mathbb{R}^n, \quad i = 1, \ldots, B. \tag{29}$$

Given gradients $dy^1, \ldots, dy^B \in \mathbb{R}^n$, backpropagation can similarly be expressed via **Nonlin** as coordinatewise applications of $d\tilde{\phi}$:

$$dz^i = d\tilde{\phi}_i(dy^1, \ldots, dy^B \mid z^1, \ldots, z^B) \in \mathbb{R}^n, \quad i = 1, \ldots, B.$$

[17] If $\epsilon = 0$ here, then the batchnorm jacobian has a singularity, so the BP-like Master Theorem does not cover it.

GRU and LSTM Since GRU and LSTMs are just recurrent dense matrix mutliplication and coordinatewise nonlinearities, we can naturally write their forward and backprop in NETSOR$\top$. Here we do so concretely for GRU.

GRU evolves according to:

$$\begin{aligned}
z^t &= \sigma(\zeta^t), \quad \zeta^t = U_z x^t + W_z h^{t-1} + b_z \\
r^t &= \sigma(\rho^t), \quad \rho^t = U_r x^t + W_r h^{t-1} + b_r \\
h^t &= z^t \odot h^{t-1} + (1 - z^t) \odot \phi(\gamma^t), \quad \gamma^t = U_h x^t + W_h (r^t \odot h^{t-1}) + b_h
\end{aligned}$$

where σ is sigmoid and ϕ is tanh, x^t, h^t, z^t, r^t are resp. the input, state, update gate, and reset gate vectors, and $W_\bullet, U_\bullet, b_\bullet$ are the weights and biases going into vector $\bullet$. Since these equations only involve **MatMul** and **Nonlin**, they can be expressed in NETSOR$\top$.

If the output is $v^\top h^T$ on the final time step T for some weights $v \in \mathbb{R}^n$, and $d\bullet$ denotes the gradient of this output against $\bullet$, then we can write the backprop as follows

$$\begin{aligned}
dh^T &= v \\
dh^{t-1} &= z^t \odot dh^t + W_z^\top d\zeta^t + r^t \odot W_h^\top d\gamma^t + W_r^\top \left(\sigma'(\rho^t) \odot d\rho^t\right) \\
dz^t &= dh^t \odot \left(h^{t-1} - \phi(\gamma^t)\right) \\
d\zeta^t &= dz^t \odot \sigma'(\zeta^t) \\
d\gamma^t &= dh^t \odot (1 - z^t) \odot \phi'(\gamma^t) \\
dr^t &= h^{t-1} \odot W_h^\top d\gamma^t \\
d\rho^t &= dr^t \odot \sigma'(\rho^t)
\end{aligned}$$

which involves only **MatMul** and **Nonlin** and so is expressible in NETSOR$\top$.

To express layernorm and attention, we need the extension NETSOR$\top^+$ to NETSOR$\top$ so we can express scalars such as the mean and variance of a layer. We will use the simplified version of NETSOR$\top^+$ given in Definition B.8.

Layernorm Given a layer's pre-activation $x \in \mathbb{R}^n$, we can use **Moment** to compute its mean and variance, where we introduce nonlinearities $\phi_\bullet$ to put the expressions in the form of **Moment**:

$$\begin{aligned}
\nu &= \frac{1}{n} \sum_{\alpha=1}^n x_\alpha = \frac{1}{n} \sum_{\alpha=1}^n \phi_{id}(x_\alpha) \in \mathbb{R} \\
\sigma^2 &= \frac{1}{n} \sum_{\alpha=1}^n (x_\alpha - \nu)^2 = \frac{1}{n} \sum_{\alpha=1}^n \phi_{sq}(x_\alpha; \nu) \in \mathbb{R}.
\end{aligned}$$

With these scalars defined, we can then express layernorm of x as

$$y = \text{Layernorm}(x) = \frac{x - \nu}{\sqrt{\sigma^2 + \epsilon}} = \phi_{LN}(x; \nu, \sigma^2, \epsilon) \in \mathbb{R}^n.$$

Now suppose we have a gradient $dy \in \mathbb{R}^n$ at y. Then backpropagating to x through the layernorm yields

$$\begin{aligned}
dx &= d\,\text{Layernorm}(dy \mid x) \in \mathbb{R}^n && (30) \\
&\stackrel{\text{def}}{=} \left(I_n - \frac{1}{n}\right)\left(I_n - \frac{yy^\top}{n}\right) \frac{dy}{\sqrt{\sigma^2 + \epsilon}} \in \mathbb{R}^n \\
&= \frac{1}{\sqrt{\sigma^2 + \epsilon}} \left(I_n - \frac{1}{n}\right)(dy - c \cdot y) \\
&= \frac{1}{\sqrt{\sigma^2 + \epsilon}} (dy - c \cdot y - a) \\
&= \phi_{bp}(dy, y; c, a, \sigma^2, \epsilon) && \textbf{Nonlin} \\
\text{where} \quad c &= \frac{y^\top dy}{n} = \frac{1}{n} \sum_{\alpha=1}^n y_\alpha dy_\alpha \in \mathbb{R} && \textbf{Moment} \\
a &= \frac{1}{n} \sum_{\alpha=1}^n dy_\alpha - c \cdot y_\alpha \in \mathbb{R} && \textbf{Moment}
\end{aligned}$$

In terms of the corresponding random variables, we have

$$Z^y = \mathring{\text{Layernorm}}(x) \stackrel{\text{def}}{=} \frac{Z^x - \mathbb{E}\, Z^x}{\sqrt{\text{Var}(Z^x) + \epsilon}} \tag{31}$$

$$Z^{dx} = d\mathring{\text{Layernorm}}(dy \mid x) \stackrel{\text{def}}{=} \frac{\text{Center}(Z^{dy} - Z^y\, \mathbb{E}\, Z^{dy} Z^y)}{\sqrt{\text{Var}(Z^x) + \epsilon}} \tag{32}$$

where $\text{Center}(X) \stackrel{\text{def}}{=} X - \mathbb{E}\, X$.

Attention Given keys, queries, and values for T tokens, $k^1, \ldots, k^T, q^1, \ldots, q^T, v^1, \ldots, v^T \in \mathbb{R}^n$, attention yields

$$\begin{aligned}
y^i &= \text{Attn}(q^i, \{k^i\}_i, \{v^i\}_i) \\
&= a_1^i v^1 + \cdots a_T^i v^T \in \mathbb{R}^n && \textbf{Nonlin} \\
\text{where} \quad (a_1^i, \ldots, a_T^i) &= \text{SoftMax}(c_1^i, \ldots, c_T^i) \\
a_j^i &= \text{SoftMax}(c_1^i, \ldots, c_T^i)_j \in \mathbb{R} && \textbf{Moment} \\
c_j^i &= q^{i\top} k^j / n = \frac{1}{n} \sum_{\alpha=1}^n q_\alpha^i k_\alpha^j \in \mathbb{R}. && \textbf{Moment}
\end{aligned}$$

In terms of corresponding random variables, we have

$$\begin{aligned}
Z^{y^i} &= \mathring{a}_1^i Z^{v^1} + \cdots + \mathring{a}_T^i Z^{v^T} \\
(\mathring{a}_1^i, \ldots, \mathring{a}_T^i) &= \text{SoftMax}\left(\mathbb{E}\, Z^{q^i} Z^{k^1}, \ldots, \mathbb{E}\, Z^{q^i} Z^{k^T}\right).
\end{aligned}$$

If $dy^1, \ldots, dy^T \in \mathbb{R}^n$ are gradients, and we abbreviate $dy = \{dy^i\}_i, k = \{k^i\}_i, q = \{q^i\}_i, v = \{v^i\}_i$, then backpropagating through the attention yields

$$\begin{aligned}
dv^j &= d_{\text{v}j}\, \text{Attn}(dy \mid k, q, v) \stackrel{\text{def}}{=} a_j^1 dy^1 + \cdots + a_j^T dy^T && \textbf{Nonlin} \\
dq^i &= d_{\text{q}i}\, \text{Attn}(dy \mid k, q, v) \stackrel{\text{def}}{=} \sum_{j,l} e_j^i f_{jl}^i k^l \in \mathbb{R}^n && \textbf{Nonlin} \\
dk^i &= d_{\text{k}i}\, \text{Attn}(dy \mid k, q, v) \stackrel{\text{def}}{=} \sum_{j,l} e_j^l f_{ji}^l q^l \in \mathbb{R}^n && \textbf{Nonlin} \qquad (33) \\
\text{with} \quad e_j^i &= dy^{i\top} v^j / n = \frac{1}{n} \sum_{\alpha=1}^n dy_\alpha^i v_\alpha^j \in \mathbb{R} && \textbf{Moment} \\
f_{jl}^i &= \partial a_j^i / \partial c_l^i = \frac{1}{n} \sum_{\alpha=1}^n \psi_{jl}(; c_1^i, \ldots, c_T^i) \in \mathbb{R} && \textbf{Moment}
\end{aligned}$$

where ψ_{jl} is a "parametrized nonlinearity" that depends only on the parameters:

$$\psi_{jl}(; c_1, \ldots, c_T) \stackrel{\text{def}}{=} \partial \text{SoftMax}_j(c_1, \ldots, c_T) / \partial c_l.$$

If we abbreviate $Z^{dy} = \{Z^{dy^i}\}_i, Z^k = \{Z^{k^i}\}_i, Z^q = \{Z^{q^i}\}_i, Z^v = \{Z^{v^i}\}_i$, then in terms of the corresponding random variables, we have

$$\begin{aligned}
Z^{dv^j} &= d_{\text{v}j}\mathring{\text{Attn}}(Z^{dy} \mid Z^k, Z^q, Z^v) \stackrel{\text{def}}{=} \mathring{a}_j^1 Z^{dy^1} + \cdots + \mathring{a}_j^T Z^{dy^T} \\
Z^{dq^i} &= d_{\text{q}i}\mathring{\text{Attn}}(Z^{dy} \mid Z^k, Z^q, Z^v) \stackrel{\text{def}}{=} \sum_{j,l} \mathring{e}_j^i \mathring{f}_{jl}^i Z^{k^l} \\
Z^{dk^i} &= d_{\text{k}i}\mathring{\text{Attn}}(Z^{dy} \mid Z^k, Z^q, Z^v) \stackrel{\text{def}}{=} \sum_{j,l} \mathring{e}_j^l \mathring{f}_{ji}^l Z^{q^l} \qquad (34) \\
\mathring{e}_j^i &= \mathbb{E}\, Z^{dy^i} Z^{v^j} \\
\mathring{f}_{jl}^i &= \psi_{jl}(; \mathbb{E}\, Z^{q^i} Z^{k^1}, \ldots, \mathbb{E}\, Z^{q^i} Z^{k^T})
\end{aligned}$$

E Example NTK Computations

In this section, we show how to compute the NTK of different architecture.

First, we review the *V-transform* of a nonlinearity.

Definition E.1. Given a multivariate nonlinearity $\Phi : \mathbb{R}^B \to \mathbb{R}^B$, its *V-transform* V_Φ is a function taking $B \times B$ positive semidefinite matrices to $B \times B$ positive semidefinite matrices, and is given by the following formula

$$\mathrm{V}_\Phi(K) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,K)} \Phi(z)\Phi(z)^\top.$$

When $\phi : \mathbb{R} \to \mathbb{R}$, we take V_ϕ to be V-transform of the $\mathbb{R}^B \to \mathbb{R}^B$ function that applies ϕ to each coordinate.

We collect below some of the common V-transforms. Here we describe the V-transforms using the function notation of kernels, but we shall freely switch between the function notation and the matrix notation in what follows.

Fact E.2 ([15])**.** *For any kernel K,*

$$\mathrm{V}_{\mathrm{relu}}(K)(x,x') = \frac{1}{2\pi}(\sqrt{1-c^2} + (\pi - \arccos c)c)\sqrt{K(x,x)K(x',x')}$$
$$\mathrm{V}_{\mathrm{relu}'}(K)(x,x') = \frac{1}{2\pi}(\pi - \arccos c)$$

where $c = K(x,x')/\sqrt{K(x,x)K(x',x')}$.

Fact E.3 ([51])**.** *For any kernel K,*

$$\mathrm{V}_{\mathrm{erf}}(K)(x,x') = \frac{2}{\pi}\arcsin\frac{K(x,x')}{\sqrt{(K(x,x)+0.5)(K(x',x')+0.5)}}$$
$$\mathrm{V}_{\mathrm{erf}'}(K)(x,x') = \frac{4}{\pi\sqrt{(1+2K(x,x))(1+2K(x',x')) - 4K(x,x')^2}}.$$

Fact E.4. *Let $\phi(x) = \exp(x/\sigma)$ for some $\sigma > 0$. For any kernel K,*

$$\mathrm{V}_\phi(K)(x,x') = \exp\left(\frac{K(x,x) + 2K(x,x') + K(x',x')}{2\sigma^2}\right).$$

E.1 MLP

In the main text, we showed how to compute infinite-width NTK of an MLP using the simplified NETSOR$\top$ (Definition 7.1). While this is the recommended way of performing the calculations, here we demonstrate formal NETSOR$\top$ (Definition A.1) calculation of the infinite-width NTK $\mathring{\Theta}(x,x)$ of the MLP described in Program 1, using Theorem A.6. By its nature, this calculation will be more verbose, so the meaning of Theorem A.6 can be seen concretely. We hope this can help readers who find the main text calculations too dense.

For simplicity, let $\phi = \mathrm{ReLU}$, assume the hidden layer widths are equal to a common integer n, $n^1 = n^2 = n$, and suppose $x \in \mathbb{R}^m$. This MLP has 5 parameters: $W^1 \in \mathbb{R}^{n\times m}, W^2 \in \mathbb{R}^{n\times n}, v \in \mathbb{R}^n, b^1 \in \mathbb{R}^n, b^2 \in \mathbb{R}^n$. In the NTK parametrization, we factor $W^1 = \frac{1}{\sqrt{m}}\omega^1$ and $W^2 = \frac{1}{\sqrt{n}}\omega^2$, and we sample $\omega^1_{\alpha\beta}, \omega^2_{\alpha\beta}, v_\alpha, b^1_\alpha, b^2_\alpha \sim \mathcal{N}(0,1)$, iid, for any α, β. This implies that $\sigma_{W^2} = 1$ in Setup A.5.

This implies that each coordinate of the G-var vector W^1x is distributed as $\mathcal{N}(0, \|x\|^2/m)$. Thus, μ^{in} is identically 0, and Σ^{in} takes the following values over pairs of G-vars

$$\Sigma^{\mathrm{in}}(W^1x, W^1x) = \|x\|^2/m, \quad \Sigma^{\mathrm{in}}(b^1,b^1) = \Sigma^{\mathrm{in}}(b^2,b^2) = \Sigma^{\mathrm{in}}(v,v) = 1,$$

and $\Sigma^{\mathrm{in}}(g,g') = 0$ for all other pairs of G-vars.

If we let $f(x)$ denote the network output $v^\top x^2/\sqrt{n}$, then by Eq. (11), the contribution of ω^1's gradient to the NTK is

$$\|\nabla_{\omega^1} f(x)\|^2 = \|\nabla_{W^1x} f(x)\|^2 \frac{\|x\|^2}{m}.$$

In Program 1, the G-var $d(W^1x)$ corresponds to $\sqrt{n}\nabla_{W^1x}f(x)$. Therefore, we can rewrite the above as

$$\|\nabla_{\omega^1}f(x)\|^2 = \frac{\|d(W^1x)\|^2}{n}\frac{\|x\|^2}{m}.$$

Similarly, the contribution of ω^2's and v's gradients to the NTK is

$$\|\nabla_{\omega^2}f(x)\|^2 = \frac{\|d\tilde{h}^2\|^2}{n}\frac{\|x^1\|^2}{n}, \quad \|\nabla_v f(x)\|^2 = \frac{\|x^2\|^2}{n}.$$

Likewise, the contributions of the bias gradients are

$$\|\nabla_{b^1}f(x)\|^2 = \frac{\|d(W^1x)\|^2}{n}, \quad \|\nabla_{b^2}f(x)\|^2 = \frac{\|d\tilde{h}^2\|^2}{n}.$$

Since the NTK can be expressed as

$$\begin{aligned}\Theta(x,x) &= \|\nabla_{\omega^1}f(x)\|^2 + \|\nabla_{\omega^2}f(x)\|^2 + \|\nabla_v f(x)\|^2 + \|\nabla_{b^1}f(x)\|^2 + \|\nabla_{b^2}f(x)\|^2 \\ &= \frac{\|d(W^1x)\|^2}{n}\frac{\|x\|^2}{m} + \frac{\|d\tilde{h}^2\|^2}{n}\frac{\|x^1\|^2}{n} + \frac{\|x^2\|^2}{n} + \frac{\|d(W^1x)\|^2}{n} + \frac{\|d\tilde{h}^2\|^2}{n} \\ &= \frac{\|d(W^1x)\|^2}{n}\left(\frac{\|x\|^2}{m}+1\right) + \frac{\|d\tilde{h}^2\|^2}{n}\left(\frac{\|x^1\|^2}{n}+1\right) + \frac{\|x^2\|^2}{n}, \end{aligned} \tag{35}$$

it suffices to compute the limits of the following squared norms, as $n \to \infty$:

$$\frac{\|d(W^1x)\|^2}{n}, \frac{\|d\tilde{h}^2\|^2}{n}, \frac{\|x^1\|^2}{n}, \frac{\|x^2\|^2}{n}, \frac{\|x\|^2}{m}.$$

Theorem A.6 provides exactly the tool needed for this purpose. Here the last squared norm $\|x\|^2/m$ is constant in n so we will focus on the other ones.

Checking the conditions of Theorem A.6 In order to apply Theorem A.6, we first need to check that 1) Program 1 is BP-like, and 2) its nonlinearities are polynomially-bounded. The latter assumption is obvious since both ReLU and its derivative, the step function, are polynomially-bounded (note that we don't require these functions to be smooth at all). The former condition is already shown in Remark A.4 to be true for any program expressing backpropagation, but we can also reason explicitly as follows: We can take the "special set of input G-vars" in Definition A.3 to be the G-var v in Program 1. Note that the only input A-var in Program 1 is W^2. Then condition 2 of Definition A.3 is satisfied because the only usage of W^2 in Program 1 is in the line $\tilde{h}^2 := W^2x^1$, and here x^1 does not depend on v. Likewise, condition 1 is satisfied because the only usage of $W^{2\top}$ in Program 1 is in the line $dx^1 := W^{2\top}d\tilde{h}^2$, and $d\tilde{h}^2$ depends linearly on, and is thus odd in v.

Limits of x^1 and x^2 In fact, the limits of $\frac{\|x^1\|^2}{n}, \frac{\|x^2\|^2}{n}$ can be computed already with the NETSOR Master Theorem of Yang [62], but for completeness we will present the calculation of their limits.

The variable x^1 has type H but it can be expressed as a function of G-vars as $\phi(W^1x + b^1)$, so by Theorem A.6,

$$\frac{\|x^1\|^2}{n} = \frac{1}{n}\sum_{\alpha=1}^{n}\phi((W^1x)_\alpha + b^1_\alpha)^2 \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z^{W^1x}, Z^{b^1}} \phi(Z^{W^1x} + Z^{b^1})^2$$

$$\text{where} \quad (Z^{W^1x}, Z^{b^1}) \sim \mathcal{N}\left(0, \begin{pmatrix}\Sigma(W^1x, W^1x) & \Sigma(W^1x, b^1) \\ \Sigma(W^1x, b^1) & \Sigma(b^1, b^1)\end{pmatrix}\right).$$

Because W^1x and b^1 are both input G-vars, this covariance matrix is just

$$\begin{pmatrix}\Sigma^{\text{in}}(W^1x, W^1x) & \Sigma^{\text{in}}(W^1x, b^1) \\ \Sigma^{\text{in}}(W^1x, b^1) & \Sigma^{\text{in}}(b^1, b^1)\end{pmatrix} = \begin{pmatrix}\|x\|^2/m & 0 \\ 0 & 1\end{pmatrix}.$$

Furthermore, by linearity of Gaussian variables, we can simplify this expectation to

$$\frac{\|x^1\|^2}{n} \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{\zeta} \phi(\zeta)^2, \quad \text{where} \quad \zeta \sim \mathcal{N}(0, \Sigma(W^1x, W^1x) + 2\Sigma(W^1x, b^1) + \Sigma(b^1, b^1))$$
$$= \mathcal{N}\left(0, \frac{\|x\|^2}{m} + 1\right).$$

Since we have assumed ϕ is ReLU, this expectation is just

$$\frac{\|x^1\|^2}{n} \xrightarrow{\text{a.s.}} \frac{1}{2}\left(\frac{\|x\|^2}{m} + 1\right) = \frac{1}{2}\frac{\|x\|^2}{m} + \frac{1}{2}.$$

To compute the the next limit $\lim_{n\to\infty} \|x^2\|^2/n$, we first need to compute $\Sigma(\tilde{h}^2, \tilde{h}^2)$ and $\Sigma(\tilde{h}^2, b^2)$. By "otherwise" case of Eq. (18), $\Sigma(\tilde{h}^2, b^2) = 0$, and by the `MatMul` case of Eq. (18),

$$\Sigma(\tilde{h}^2, \tilde{h}^2) = \sigma_{W^2} \mathop{\mathbb{E}}_{Z^{W^1x}, Z^{b^1}} \phi(Z^{W^1x} + Z^{b^1})^2 = \frac{1}{2}\frac{\|x\|^2}{m} + \frac{1}{2},$$

as we have computed above already.

Therefore, by Fig. 1,

$$\|x^2\|^2/n = \frac{1}{n}\sum_{\alpha=1}^{n} \phi(\tilde{h}_\alpha^2 + b_\alpha^2)^2 \xrightarrow{\text{a.s.}} \mathbb{E}\, \phi(Z^{\tilde{h}^2} + Z^{b^2})^2,$$
$$\text{where} \quad (Z^{\tilde{h}^2}, Z^{b^2}) \sim \mathcal{N}\left(0, \begin{pmatrix} \Sigma(\tilde{h}^2, \tilde{h}^2) & \Sigma(\tilde{h}^2, b^2) \\ \Sigma(\tilde{h}^2, b^2) & \Sigma(b^2, b^2) \end{pmatrix}\right) = \mathcal{N}\left(0, \begin{pmatrix} \frac{1}{2}\frac{\|x\|^2}{m} + \frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}\right).$$

Again, by linearity of Gaussians, we have

$$\|x^2\|^2/n \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{\zeta \sim \mathcal{N}(0, \|x\|^2/2m + 3/2)} \phi(\zeta) = \frac{\|x\|^2}{4m} + \frac{3}{4}.$$

Limits of $\frac{\|d(W^1x)\|^2}{n}$ and $\frac{\|d\tilde{h}^2\|^2}{n}$ Whereas the limits computed above could already be done using the NETSOR Master Theorem of Yang [62], the limits we will compute here necessarily involve matrix transposes and thus can only be computed using Theorem A.6.

By Theorem A.6,

$$\|d\tilde{h}^2\|^2/n = \frac{1}{n}\sum_{\alpha=1}^{n} (\phi'(\tilde{h}_\alpha^2 + b_\alpha^2) dx_\alpha^2)^2 \xrightarrow{\text{a.s.}} \mathbb{E}\, \phi'(Z^{\tilde{h}^2} + Z^{b^2})^2 (Z^{dx^2})^2$$

where

$$(Z^{\tilde{h}^2}, Z^{b^2}, Z^{dx^2}) \sim \mathcal{N}\left(0, \begin{pmatrix} \Sigma(\tilde{h}^2, \tilde{h}^2) & \Sigma(\tilde{h}^2, b^2) & \Sigma(\tilde{h}^2, dx^2) \\ \Sigma(b^2, \tilde{h}^2) & \Sigma(b^2, b^2) & \Sigma(b^2, dx^2) \\ \Sigma(dx^2, \tilde{h}^2) & \Sigma(dx^2, b^2) & \Sigma(dx^2, dx^2) \end{pmatrix}\right)$$
$$= \mathcal{N}\left(0, \begin{pmatrix} \frac{1}{2}\|x\|^2/m + \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right).$$

Since Z^{dx^2} is independent from $Z^{\tilde{h}^2}$ and Z^{b^2}, we have

$$\|d\tilde{h}^2\|^2/n \xrightarrow{\text{a.s.}} \mathbb{E}\, \phi'(\zeta_1)^2(\zeta_2)^2, \quad \text{with} \quad \zeta_1 \sim \mathcal{N}\left(0, \frac{1}{2}\|x\|^2/m + \frac{3}{2}\right), \quad \zeta_2 \sim \mathcal{N}(0,1)$$
$$= \mathbb{E}\, \phi'(\zeta_1)^2\, \mathbb{E}(\zeta_2)^2 = \frac{1}{2} \cdot 1 = \frac{1}{2}.$$

Next, notice that by Eq. (18), for the same ζ_1, ζ_2 above, we have

$$\Sigma(dx^1, dx^1) = \sigma_{W^{2\top}}\, \mathbb{E}\, \phi'(\zeta_1)^2(\zeta_2)^2 = 1 \cdot \frac{1}{2} = \frac{1}{2}$$

as before (in this calculation, the `MatMul` case of Eq. (18) essentially "forgets" the correlation between W^2 and $W^{2\top}$ and treats $W^{2\top}$ as just another independently sampled matrix). In addition, $\Sigma(dx^1, b^1) = \Sigma(dx^1, W^1 x) = 0$ by the "otherwise" case of Eq. (18). Consequently, by Theorem A.6,

$$\|d(W^1x)\|^2/n = \frac{1}{n}\sum_{\alpha=1}^{n} \phi'((W^1x)_\alpha + b^1_\alpha)^2 (dx^1)^2_\alpha \xrightarrow{\text{a.s.}} \mathbb{E}\,\phi'(Z^{W^1x} + Z^{b^1})^2 (Z^{dx^1})^2,$$

$$\text{where} \quad (Z^{W^1x}, Z^{b^1}, Z^{dx^2}) \sim \mathcal{N}\left(0, \begin{pmatrix} \Sigma(W^1x, W^1x) & \Sigma(W^1x, b^1) & \Sigma(W^1x, dx^2) \\ \Sigma(b^1, W^1x) & \Sigma(b^1, b^1) & \Sigma(b^1, dx^2) \\ \Sigma(dx^2, W^1x) & \Sigma(dx^2, b^1) & \Sigma(dx^2, dx^2) \end{pmatrix}\right)$$
$$= \mathcal{N}\left(0, \begin{pmatrix} \|x\|^2/m & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}\right).$$

This expectation is easily evaluated, and we have

$$\|d(W^1x)\|^2/n \xrightarrow{\text{a.s.}} \frac{1}{2}\cdot\frac{1}{2} = \frac{1}{4}$$

Finish computing the infinite-width NTK $\mathring{\Theta}$ In summary, we have

$$\frac{\|x^1\|^2}{n} \xrightarrow{\text{a.s.}} \frac{1}{2}\frac{\|x\|^2}{m} + \frac{1}{2}, \quad \frac{\|x^2\|^2}{n} \xrightarrow{\text{a.s.}} \frac{1}{4}\frac{\|x\|^2}{m} + \frac{3}{4}, \quad \frac{\|d\tilde{h}^2\|^2}{n} \xrightarrow{\text{a.s.}} \frac{1}{2}, \quad \frac{\|d(W^1x)\|^2}{n} \xrightarrow{\text{a.s.}} \frac{1}{4}.$$

Thus, by Eq. (35), we have

$$\Theta(x,x) \xrightarrow{\text{a.s.}} \mathring{\Theta}(x,x) = \frac{1}{4}\left(\frac{\|x\|^2}{m} + 1\right) + \frac{1}{2}\left(\frac{1}{2}\frac{\|x\|^2}{m} + \frac{1}{2} + 1\right) + \left(\frac{1}{4}\frac{\|x\|^2}{m} + \frac{3}{4}\right)$$
$$= \frac{3}{4}\frac{\|x\|^2}{m} + \frac{7}{4}.$$

Generalization to multiple inputs This example only computed $\mathring{\Theta}(x,x)$. The same reasoning can be easily applied to multiple inputs by writing down a program expressing the forward and backward computation of the MLP on two inputs.

E.2 Simple Recurrent Neural Network and Average Pooling

We complete the NTK limit computation from the main text and also generalize it to the case where the output is a projection of the average state instead of just the last state. Recall the RNN we consider is given by the following forward and backward equations

$$s^t(\xi) = \phi(g^t(\xi) + u^t(\xi) + b), \quad g^t(\xi) = W s^{t-1}(\xi), \quad u^t(\xi) = U\xi^t$$
$$ds^{t-1} = W^\top dg^t, \quad du^t = dg^t = \phi'\left(g^t + u^t + b\right) \odot ds^t.$$

For an input sequence $\xi = \left\{\xi^1, \ldots, \xi^t, \ldots, \xi^T \in \mathbb{R}^d\right\}$, we will consider both

LastState the case where the output of the RNN is a projection of the last state

$$f(\xi) = v^\top s^T/\sqrt{n}$$

as in main text and

AvgPool the case where the output of the RNN is a projection of the average state

$$f(\xi) = \frac{1}{T}\sum_{t=1}^{T} v^\top s^t/\sqrt{n}.$$

If we sample

$$W_{\alpha\beta} \sim \mathcal{N}(0, \sigma_W^2/n), U_{\alpha\beta} \sim \mathcal{N}(0, \sigma_U^2/d), b_\alpha \sim \mathcal{N}(0, \sigma_b^2), v_\alpha \sim \mathcal{N}(0, \sigma_v^2)$$

then the recursion equations in the main text can be generalized straightforwardly to the following

$$\begin{aligned} C^{s^t,\bar{s}^r} &= \mathbb{E}\,\phi(\zeta_1)\phi(\zeta_2), \\ D^{s^t,\bar{s}^r} &= D^{s^{t+1},\bar{s}^{r+1}}\,\mathbb{E}\,\phi'(\zeta_1)\phi'(\zeta_2) \\ D^{g^t,\bar{g}^r} &= D^{u^t,\bar{u}^r} = \sigma_W^{-2} D^{s^{t-1},\bar{s}^{r-1}} \end{aligned}$$

where $(\zeta_1,\zeta_2) \sim \mathcal{N}\left(\sigma_W^2 \begin{pmatrix} C^{s^t,s^t} & C^{s^t,\bar{s}^r} \\ C^{s^t,\bar{s}^r} & C^{\bar{s}^r,\bar{s}^r} \end{pmatrix} + \sigma_U^2 \frac{\xi^{t\top}\xi^r}{d} + \sigma_b^2\right)$, and we have abbreviated $C^{s^t,\bar{s}^r} = C^{s^t,\bar{s}^r}(\xi,\bar{\xi}) = \mathbb{E}\, Z^{s^t} Z^{\bar{s}^r} = \lim_{n\to\infty} n^{-1} s^{t\top}\bar{s}^r$, and so on.

Initial condition for LastState If we use the last state for output, then the initial conditions are

$$\begin{aligned} C^{s^0,\bar{s}^r} &= C^{s^t,\bar{s}^0} = 0 \\ D^{s^T,\bar{s}^{\bar{T}}} &= \sigma_v^2 \qquad \text{but} \\ D^{s^T,\bar{s}^r} = D^{s^t,\bar{s}^{\bar{T}}} &= 0, \qquad \text{for all other } r,t. \end{aligned}$$

Here, the initial condition for $D^{s^t,\bar{s}^r}$ reflects the fact that only the last state is used for output. This initial condition in fact implies that most $D^{s^t,\bar{s}^r}$ are 0 by a simple induction:

$$i \neq j \implies D^{s^{-i},\bar{s}^{-j}} = 0$$

where $s^{-i} = s^{T-i}$, $\bar{s}^{-j} = \bar{s}^{\bar{T}-j}$.

Initial condition for AvgPool If instead of projecting the last state, we project the average of all states to get the output, then the initial condition for D is

$$D^{s^T,\bar{s}^r} = D^{s^t,\bar{s}^{\bar{T}}} = \sigma_v^2, \quad \forall r,t$$

In this case we can't zero out the majority of D like the above.

NTK To compute the NTK, we apply Eq. (12) to get

$$\mathring{\Theta}(\xi,\bar{\xi}) = \sum_{t=1}^{T-1}\sum_{r=1}^{\bar{T}-1} D^{g^{t+1},\bar{g}^{r+1}} C^{s^t,\bar{s}^r} + \sum_{t=1}^{T}\sum_{r=1}^{\bar{T}} D^{g^t,\bar{g}^r}\frac{\xi^t\bar{\xi}^r}{d} + \sum_{t=1}^{T}\sum_{r=1}^{\bar{T}} D^{g^t,\bar{g}^r} + C^{s^T,\bar{s}^{\bar{T}}}$$

where the terms in the sum are resp. contributions from W, U, b, and v. As noted above, if the output depends only on the last state, then the double sum above can be replaced with a single sum over the diagonal $D^{s^{-i},\bar{s}^{-i}}$.

E.3 Convolution Neural Network

We continue the notation of the **Convolution** section of Appendix D. We consider a convolutional neural network with width n, with feature map positions given by P, and with convolutional kernel positions K, throughout the network. For example, K can be $\{-1,0,1\}\times\{-1,0,1\}$, and P can be $[32]\times[32]$. For simplicity, we forgo bias. Let $x^l = \{x_s^l \in \mathbb{R}^n : s \in P\}$ and $h^l = \{h_s^l \in \mathbb{R}^n : s \in P\}$ be the activations and preactivations of layer l. Let $W^l = \{W_\kappa^l : \kappa \in K\}$ be the layer l weights. The input $\xi = \{\xi_s \in \mathbb{R}^d : s \in P\}$ to the network is an image with $d = 3$ channels and has pixel positions P . Then the network computation proceeds by

$$x_s^0(\xi) = \xi_s, \qquad h_s^l(\xi) = \sum_\kappa W_\kappa^l x_{s+\kappa}^{l-1}(\xi), \qquad x_s^l(\xi) = \phi(h_s^l(\xi)),$$

for $l = 1,\ldots,L$, where the sum is over κ such that $s+\kappa \in P$. We sample $W_{\kappa\alpha\beta}^l \sim \mathcal{N}(0,\sigma_w^2/n)$ for $l = 2,\ldots,L$, and $W_{\kappa\alpha\beta}^1 \sim \mathcal{N}(0,\sigma_w^2/d)$. For the output, we will consider both

Global Average Pooling (GAP) where output of network is given by

$$f(\xi) = \frac{1}{|P|}\sum_{s\in P}\frac{1}{\sqrt{n}} v^\top x_s^L$$

for $v_\alpha \sim \mathcal{N}(0,\sigma_v^2)$ and

Vectorization where output of network is given by

$$f(\xi) = \frac{1}{\sqrt{|P|}} \sum_{s \in P} \frac{1}{\sqrt{n}} v_s^\top x_s^L$$

for output weights $v = \{v_s \in \mathbb{R}^n : s \in P\}$ sampled like $v_{s\alpha} \sim \mathcal{N}(0, \sigma_v^2)$. This is called "vectorization" because it's equivalent to a linear readout of the flattening (vectorization) of final layer embeddings $x^L = \{x_s^L : s \in P\}$.

Forward Propagation Given another input $\bar{\xi} = \{\bar{\xi}_s \in \mathbb{R}^d : s \in P\}$, we define $\bar{h}_s^l = h_s^l(\bar{\xi}), \bar{x}_s^l = x_s^l(\bar{\xi})$ similarly. Then by the NETSOR$\top$ Master Theorem (Theorem 7.2), there exist some deterministic scalars $C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}), C^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi})$, such that

$$h_s^{l\top} \bar{h}_t^l / n \xrightarrow{\text{a.s.}} \mathbb{E}\, Z^{h_s^l} Z^{\bar{h}_t^l} \stackrel{\text{def}}{=} C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi})$$
$$x_s^{l\top} \bar{x}_t^l / n \xrightarrow{\text{a.s.}} \mathbb{E}\, Z^{x_s^l} Z^{\bar{x}_t^l} \stackrel{\text{def}}{=} C^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi}),$$

for all $s, t \in P, l = 1, \dots, L$. For convenience, we also define $C^{x_{s+\kappa}^0, \bar{x}_{t+\kappa}^0}(\xi, \bar{\xi}) \stackrel{\text{def}}{=} x_{s+\kappa}^{0\top} \bar{x}_{t+\kappa}^0 / d$. As in Eq. (26) and in the MLP case, these scalars are related to each other through a recurrence: for $l = 1, \dots, L$,

$$C^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi}) = \mathbb{E}\, \phi(\zeta)\phi(\bar{\zeta}), \quad (\zeta, \bar{\zeta}) \sim \begin{pmatrix} C^{h_s^l, h_s^l}(\xi, \xi) & C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) \\ C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) & C^{\bar{h}_t^l, \bar{h}_t^l}(\bar{\xi}, \bar{\xi}) \end{pmatrix}$$
$$C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) = \sigma_w^2 \sum_\kappa C^{x_{s+\kappa}^{l-1}, \bar{x}_{t+\kappa}^{l-1}}(\xi, \bar{\xi}).$$

Backpropagation Define $dh_s^l \stackrel{\text{def}}{=} \sqrt{n} \nabla_{h_s^l} f(\xi), d\bar{h}_s^l \stackrel{\text{def}}{=} \sqrt{n} \nabla_{\bar{h}_s^l} f(\bar{\xi})$. Then, the backpropagation of f proceed as follows.

$$dh_s^l = dx_s^l \odot \phi'(h_s^l)$$
$$dx_s^{l-1} = \sum_\kappa W_\kappa^{l\top} dh_{s-\kappa}^l.$$

By the NETSOR$\top$ Master Theorem (Theorem 7.2), there exist some deterministic scalars $D^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}), D^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi})$ such that

$$dh_s^{l\top} d\bar{h}_t^l / n \xrightarrow{\text{a.s.}} \mathbb{E}\, Z^{dh_s^l} Z^{d\bar{h}_t^l} \stackrel{\text{def}}{=} D^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi})$$
$$dx_s^{l\top} d\bar{x}_t^l / n \xrightarrow{\text{a.s.}} \mathbb{E}\, Z^{dx_s^l} Z^{d\bar{x}_t^l} \stackrel{\text{def}}{=} D^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi}),$$

for $l = L, \dots, 1$. These scalars are related to each other through a recurrence: As in the MLP case, we have, for $l = L, \dots, 1$,

$$D^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) = D^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi})\, \mathbb{E}\, \phi'(\zeta)\phi'(\bar{\zeta}), \quad (\zeta, \bar{\zeta}) \sim \begin{pmatrix} C^{h_s^l, h_s^l}(\xi, \xi) & C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) \\ C^{h_s^l, \bar{h}_t^l}(\xi, \bar{\xi}) & C^{\bar{h}_t^l, \bar{h}_t^l}(\bar{\xi}, \bar{\xi}) \end{pmatrix}.$$

As in Eq. (27), we also have, for $l = L, \dots, 2$,

$$D^{x_s^{l-1}, \bar{x}_t^{l-1}}(\xi, \bar{\xi}) = \sigma_w^2 \sum_\kappa D^{h_{s-\kappa}^l, \bar{h}_{t-\kappa}^l}.$$

Now the initial condition for this recurrence depends on whether the network has global average pooling or not:

$$D^{x_s^L, \bar{x}_t^L}(\xi, \bar{\xi}) = \begin{cases} \sigma_v^2 / |P|^2 & \text{if GAP} \\ \sigma_v^2 \mathbb{I}(s = t) / |P| & \text{if vectorization.} \end{cases}$$

Note in the vectorization case, this initial condition implies that, for all l, $D^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi}) = D^{x_s^l, \bar{x}_t^l}(\xi, \bar{\xi}) = 0$ if $s \neq t$.

NTK Finally, we can decompose the NTK into contributions from $\nabla_v f$ and from $\{\nabla_{\omega_\kappa^l} f\}_{\kappa,l}$. If the last layer involves GAP, then the contribution of $\nabla_v f$ is

$$\langle \nabla_v f(\xi), \nabla_v f(\bar\xi)\rangle = \frac{1}{n}\left\langle |P|^{-1}\sum_{s\in P} x_s^L, |P|^{-1}\sum_{t\in P}\bar x_t^L\right\rangle = |P|^{-2}\sum_{s,t\in P}\frac{x_s^{L\top}\bar x_t^L}{n}$$
$$\xrightarrow{\text{a.s.}} |P|^{-2}\sum_{s,t\in P} C^{x_s^L,\bar x_t^L}(\xi,\bar\xi).$$

If the last layer involves vectorization, then

$$\langle \nabla_v f(\xi), \nabla_v f(\bar\xi)\rangle = \sum_{s\in P}\langle \nabla_{v_s} f(\xi), \nabla_{v_s} f(\bar\xi)\rangle = |P|^{-1}\sum_{s\in P}\frac{x_s^{L\top}\bar x_s^L}{n}$$
$$\xrightarrow{\text{a.s.}} |P|^{-1}\sum_{s\in P} C^{x_s^L,\bar x_s^L}(\xi,\bar\xi).$$

For $l>1$, the contribution of $\nabla_{\omega_\kappa^l} f$ is

$$\langle \nabla_{\omega_\kappa^l} f(\xi), \nabla_{\omega_\kappa^l} f(\bar\xi)\rangle = n^{-2}\left\langle \sum_s dh_s^l x_{s+\kappa}^{l-1}, \sum_s d\bar h_s^l \bar x_{s+\kappa}^{l-1}\right\rangle = \sum_{s,t}\frac{dh_s^{l\top} d\bar h_t^l}{n}\frac{x_{s+\kappa}^{l-1\top}\bar x_{t+\kappa}^{l-1}}{n}$$
$$\xrightarrow{\text{a.s.}} D^{h_s^l,\bar h_t^l}(\xi,\bar\xi) C^{x_{s+\kappa}^{l-1},\bar x_{t+\kappa}^{l-1}}(\xi,\bar\xi).$$

For $l=1$, if we define $C^{x_{s+\kappa}^0,\bar x_{t+\kappa}^0}(\xi,\bar\xi)\overset{\text{def}}{=}\frac{x_{s+\kappa}^{0\top}\bar x_{t+\kappa}^0}{d}$, we similarly have

$$\langle \nabla_{\omega_\kappa^1} f(\xi), \nabla_{\omega_\kappa^1} f(\bar\xi)\rangle = n^{-1}d^{-1}\left\langle \sum_s dh_s^1 x_{s+\kappa}^0, \sum_s d\bar h_s^1 \bar x_{s+\kappa}^0\right\rangle = \sum_{s,t}\frac{dh_s^{1\top} d\bar h_t^1}{n}\frac{x_{s+\kappa}^{0\top}\bar x_{t+\kappa}^0}{d}$$
$$\xrightarrow{\text{a.s.}} D^{h_s^1,\bar h_t^1}(\xi,\bar\xi) C^{x_{s+\kappa}^0,\bar x_{t+\kappa}^0}(\xi,\bar\xi).$$

Altogether, the NTK is

$$\Theta(\xi,\bar\xi) = \sum_{l=1}^L\sum_{\kappa\in K}\langle \nabla_{\omega_\kappa^l} f(\xi), \nabla_{\omega_\kappa^l} f(\bar\xi)\rangle + \langle \nabla_v f(\xi), \nabla_v f(\bar\xi)\rangle$$
$$\xrightarrow{\text{a.s.}} \sum_{l=1}^L\sum_\kappa\sum_{s,t} D^{h_s^l,\bar h_t^l}(\xi,\bar\xi) C^{x_{s+\kappa}^{l-1},\bar x_{t+\kappa}^{l-1}}(\xi,\bar\xi) + \begin{cases} |P|^{-2}\sum_{s,t\in P} C^{x_s^L,\bar x_t^L}(\xi,\bar\xi) & \text{if GAP}\\ |P|^{-1}\sum_{s\in P} C^{x_s^L,\bar x_s^L}(\xi,\bar\xi) & \text{else.}\end{cases}$$

Here the sum is over $\kappa\in K$ and $s,t\in P$ such that $s+\kappa, t+\kappa\in P$.

E.3.1 Vectorized NTK Formula

Let $\{\xi_i\}_{i=1}^k$ be a set of inputs. For $l=1,\dots,L$, define the tensors $C^{x^l}, C^{h^l}\in\mathbb{R}^{k\times P\times k\times P}$ by $C^{x^l}_{isjt}\overset{\text{def}}{=}C^{x_s^l,\bar x_t^l}(\xi_i,\xi_j), C^{h^l}_{isjt}\overset{\text{def}}{=}C^{h_s^l,\bar h_t^l}(\xi_i,\xi_j)$. Note that s (and t) is a spatial index that may expand to double indices if P is 2-dimensional (e.g. $P=[32]\times[32]$). For any tensor $C=\{C_{isjt}\}_{isjt}\in\mathbb{R}^{k\times P\times k\times P}$, define the linear operator

$$\mathcal{T}(C)_{isjt}\overset{\text{def}}{=}\sum_\kappa C_{i,s+\kappa,j,t+\kappa}$$

where the sum is over $\kappa\in K$ such that $s+\kappa, t+\kappa$ are both in P. This operator can be easily implemented via convolution in `pytorch` or `tensorflow`. Also define $C^{x^0}\in\mathbb{R}^{k\times P\times k\times P}$ by $C^{x^0}_{isjt}=\xi_{is}^\top\xi_{jt}/d$. Then assuming $K=-K$[18],

$$C^{x^l}=\mathrm{V}_\phi(C^{h^l}) \qquad C^{h^l}=\sigma_w^2\mathcal{T}(C^{x^{l-1}})$$
$$D^{x^{l-1}}=\sigma_w^2\mathcal{T}(D^{h^l}) \qquad D^{h^l}=D^{x^l}\odot \mathrm{V}_{\phi'}(C^{h^l}).$$

[18]Note, in general when $K\neq -K$, we have $D^{x^{l-1}}=\sigma_w^2\mathcal{T}^\dagger(D^{h^l})$, where $\mathcal{T}^\dagger$ is the adjoint of $\mathcal{T}$.

The initial condition is

$$D^{x^L}_{isjt} = \begin{cases} \sigma_v^2/|P|^2 & \text{if GAP} \\ \sigma_v^2 \mathbb{I}(s=t)/|P| & \text{else.} \end{cases}$$

Note in the vectorization case, we only need to compute the entries $C^{x^l}_{isjt}, C^{h^l}_{isjt}$ where $s = t$, as everything else will be 0. Finally, the infinite-width NTK is given by

$$\mathring{\Theta}_{ij} = \sum_{l=1}^{L} D^{h^l} \circ \mathcal{T}(C^{x^{l-1}}) + \begin{cases} |P|^{-2} \sum_{s,t \in P} C^{x^L}_{isjt} & \text{if GAP} \\ |P|^{-1} \sum_{s \in P} C^{x^L}_{isjs} & \text{else.} \end{cases}$$

where $\circ$ contracts the s, t indices: $(A \circ B)_{ij} = \sum_{s,t \in P} A_{isjt} B_{isjt}$.

E.4 Batchnorm

We first detail how to propagate the covariances of activations and of gradients before describing how we can combine them to compute the NTK. We use the notation of Eq. (28) and let $\tilde{\phi}$ denote batchnorm followed by coordinatewise nonlinearity ϕ.

Forward Single Batch If $h^1, \dots, h^B$ are the pre-activations of a layer over a batch of size B, and

$$x^1, \dots, x^B = \widetilde{\text{relu}}(h^1, \dots, h^B),$$

then as discussed in Appendix D, this is a valid tensor program. If $Z^{h^1}, \dots, Z^{h^B}$ are jointly distributed as $\mathcal{N}(0, \Sigma)$, then [62, 68] showed that $Z^{x^1}, \dots, Z^{x^B}$ has the 2nd moment matrix $\Sigma', \Sigma'_{ij} = \mathbb{E}\, Z^{x^1} Z^{x^B}$, given by

$$\Sigma' = B \int_0^\infty \frac{\text{V}_{\text{relu}}(\Sigma^G (I + 2s\Sigma^G)^{-1})}{\sqrt{\det(I + 2s\Sigma^G)}}\, \mathrm{d}s \tag{36}$$

where V_{relu} is as in Fact E.2, and $\Sigma^G = G\Sigma G, G = I_B - \frac{1}{B}\mathbf{1}\mathbf{1}^\top$.

Forward Cross Batch Suppose $\bar{h}^1, \dots, \bar{h}^{\bar{B}}$ are the pre-activations of a layer over another batch of size $\bar{B}$ (possibly $\bar{B} \neq B$), such that $(Z^{h^1}, \dots, Z^{h^B}, Z^{\bar{h}^1}, \dots, Z^{\bar{h}^{\bar{B}}})$ is jointly distributed as $\mathcal{N}\left(0, \begin{pmatrix} \Sigma & \Xi \\ \Xi^\top & \bar{\Sigma} \end{pmatrix}\right)$. Let $\bar{x}^1, \dots, \bar{x}^B = \widetilde{\text{relu}}(\bar{h}^1, \dots, \bar{h}^B)$. Then the *cross-batch moment matrix* $\Xi', \Xi'_{ij} = \mathbb{E}\, Z^{x^i} Z^{\bar{x}^j}$, is given by

$$\Xi' = \sqrt{B\bar{B}}\pi^{-1} \int_0^\infty \mathrm{d}s \int_0^\infty \mathrm{d}t\, (st)^{-1/2} \det(I_{B+\bar{B}} + 2\Omega)^{-1/2} \text{V}_{\text{relu}}(\Pi)_{12} \tag{37}$$

where

$$\begin{aligned} \Omega &= D^{1/2} \begin{pmatrix} G\Sigma G & G\Xi\bar{G} \\ \bar{G}\Xi^\top G & \bar{G}\bar{\Sigma}\bar{G} \end{pmatrix} D^{1/2} \\ \Pi &= D^{-1/2}\Omega(I + 2\Omega)^{-1} D^{-1/2} \\ D &= sI_B \oplus tI_{\bar{B}} = \begin{pmatrix} sI_B & 0 \\ 0 & tI_{\bar{B}} \end{pmatrix} \\ G &= I_B - B^{-1}\mathbf{1}\mathbf{1}^\top \\ \bar{G} &= I_{\bar{B}} - \bar{B}^{-1}\mathbf{1}\mathbf{1}^\top \end{aligned}$$

and $\text{V}_{\text{relu}}(\Pi)_{12}$ is the block of $\text{V}_{\text{relu}}(\Pi)$ on the first row, second column, of size $B \times \bar{B}$.

Backward Single Batch Now, suppose $dx^1, \dots, dx^B$ are gradients such that $Z^{dx^1}, \dots, Z^{dx^B}$ are jointly distributed as $\mathcal{N}(0, \Delta)$ independently from $Z^{h^1}, \dots, Z^{h^B}$. Let $dh^1, \dots, dh^B = d\,\widetilde{\text{relu}}(dx^1, \dots, dx^B \mid h^1, \dots, h^B)$. Then, by Yang et al. [68], $\{Z^{dh^i}\}_i$ has 2nd-moment matrix $\Delta', \Delta'_{ij} = \mathbb{E}\, Z^{dh^i} Z^{dh^j}$, given by

$$\Delta' = B \int_0^\infty \delta(\Lambda_1 + \Lambda_2 - \Lambda_3)^G\, \mathrm{d}s \tag{38}$$

where

$$\Lambda_1 = \Lambda_1(s) = \Delta \odot \mathrm{V}_{\mathrm{step}}(K(s))$$
$$\Lambda_2 = \Lambda_2(s) = \frac{1}{2}s^2(\langle \Delta, \mathrm{V}_{\mathrm{relu}}(K(s))\rangle K(s) + 2K(s)J(s)K(s))$$
$$\Lambda_3 = \Lambda_3(s) = s(K(s)J(s) + J(s)K(s))$$

and

$$\delta(s) = 1/\sqrt{\det(I + 2s\Sigma^G)}$$
$$K(s) = \Sigma^G(I + 2s\Sigma^G)^{-1}$$
$$J(s) = \frac{d\mathrm{V}_{\mathrm{relu}}(K(s))}{dK(s)}^{\dagger}\{\Delta\}.$$

Here, $\frac{d\mathrm{V}_{\mathrm{relu}}(K(s))}{dK(s)}$ is a matrix-to-matrix linear operator, and $\frac{d\mathrm{V}_{\mathrm{relu}}(K(s))}{dK(s)}^{\dagger}$ denotes its adjoint, which "backprops" a gradient of $\mathrm{V}_{\mathrm{relu}}(K(s))$ to a gradient of $K(s)$.

Backward Cross Batch Now, suppose $d\bar{x}^1, \ldots, d\bar{x}^B$ are gradients such that $(Z^{dx^1}, \ldots, Z^{dx^B}, Z^{d\bar{x}^1}, \ldots, Z^{d\bar{x}^{\bar{B}}})$ are jointly distributed as $\mathcal{N}\left(0, \begin{pmatrix} \Delta & \chi \\ \chi^\top & \bar{\Delta} \end{pmatrix}\right)$ independently from $(Z^{h^1}, \ldots, Z^{h^B}, Z^{\bar{h}^1}, \ldots, Z^{\bar{h}^{\bar{B}}})$. Let

$$d\bar{h}^1, \ldots, d\bar{h}^{\bar{B}} = d\,\widetilde{\mathrm{relu}}(d\bar{x}^1, \ldots, d\bar{x}^B \mid \bar{h}^1, \ldots, \bar{h}^B). \tag{39}$$

Then the *cross-batch moment matrix* $\chi', \chi'_{ij} = \mathbb{E}\, Z^{dh^i} Z^{d\bar{h}^j}$, is given by

$$\chi' = \sqrt{B\bar{B}} \int_0^\infty \int_0^\infty \gamma\, G(\Gamma_1 + \Gamma_2 - \Gamma_3)\bar{G}\ \mathrm{d}s\ \mathrm{d}t$$

where

$$\Gamma_1 = \Gamma_1(s,t) = \chi \odot \mathrm{V}_{\mathrm{step}}(\Pi)_{12}$$
$$\Gamma_2 = \Gamma_2(s,t) = 4st(\langle \mathrm{V}_{\mathrm{relu}}(\Pi)_{12}, \chi\rangle \Pi_{12} + (\Pi J \Pi)_{12})$$
$$\Gamma_3 = \Gamma_3(s,t) = 2(t(J\Pi)_{12} + s(\Pi J)_{12})$$

with A_{12} denoting the off-diagonal block of A, and

$$\gamma = \gamma(s,t) = \pi^{-1}s^{-1/2}t^{-1/2}\det(I_{B+\bar{B}} + 2\Omega)^{-1/2}$$
$$D = D(s,t) = \begin{pmatrix} sI_B & 0 \\ 0 & tI_{\bar{B}} \end{pmatrix}$$
$$\Omega = \Omega(s,t) = D^{1/2}\Sigma D^{1/2}$$
$$\Pi = \Pi(s,t) = D^{-1/2}\Omega(I + 2\Omega)^{-1}D^{-1/2}$$
$$J = J(s,t) = \frac{d\mathrm{V}_{\mathrm{relu}}(\Pi)}{d\Pi}^{\dagger}\{\begin{pmatrix} 0 & \chi \\ \chi^\top & 0 \end{pmatrix}\}$$

The backward equations Eqs. (38) and (39) are not explicit in [68] but can be derived from Lemma H.5, Eq (52), Prop G.8, and Lemma G.10 from [68].

E.4.1 NTK

Because batchnorm turns the neural network into a batch-to-batch function, its infinite-width NTK is constructed slightly differently than other networks demonstrated in the main text. We summarize its computation below.

Two inputs of the same batch Let $x_1, \ldots, x_B \in \mathbb{R}^d$ be a batch of inputs.

Consider a batchnorm-ReLU MLP with L hidden layers and width n. Its forward pass is given by

$$h_i^l = \omega^l x_i^{l-1} \in \mathbb{R}^n, \quad x^l = \widetilde{\text{relu}}(h^l) \in \mathbb{R}^{B\times n}, \quad x_i^1 = \omega^1 x_i \in \mathbb{R}^n$$

with weights $\omega_{\alpha\beta}^l \sim \mathcal{N}(0,1)$, and it has output $y_i = \frac{1}{\sqrt{n}} v^\top x^L$ for parameters $v \sim \mathcal{N}(0,1)$.

Starting with the $\Sigma_{ij}^0 = \langle x_i, x_j\rangle/d$, compute $\Sigma^l = (\Sigma^{l-1})', l = 1, \ldots L$, according to Eq. (36).

Suppose we want to compute the NTK's value on two inputs x_i, x_j, possibly the same. Start with $\Delta^{L+1} = \frac{1}{2}(\delta_{ij} + \delta_{ji})$ where δ_{ij} is the matrix with zero everywhere except 1 at the (i,j)th entry. Then compute $\Delta^l = (\Delta^{l+1})', l = L, \ldots, 1$ according to Eq. (38). Then

$$\mathring{\Theta}(x_i, x_j) = \sum_{l=0}^{L} \langle \Sigma^l, \Delta^{l+1}\rangle.$$

Two inputs of different batches Let $\bar{x}_1, \ldots, \bar{x}_B \in \mathbb{R}^d$ be a second batch of inputs, and compute $\bar{\Sigma}^l, l = 0, \ldots, L$ for them just like how Σ^l are computed for $x_1, \ldots, x_B \in \mathbb{R}^d$ above. In addition, starting with $\Xi_{ij}^0 = \langle x_i, \bar{x}_j\rangle/d$, compute $\Xi^l = (\Xi^{l-1})', l = 1, \ldots, L$, according to Eq. (37).

Suppose we want to compute the NTK's value on two inputs $x_i, \bar{x}_j$ from different batches. Start with $\chi^{L+1} = \delta_{ij}$, compute $\chi^l = (\chi^{l+1})', l = L, \ldots, 1$, according to Eq. (39). Then

$$\mathring{\bar{\Theta}}(x_i, \bar{x}_j) = \sum_{l=0}^{L} \langle \Xi^l, \chi^{l+1}\rangle.$$

E.5 Transformer

We'll work with the following transformer variant. Let $x_1^0, \ldots, x_T^0 \in \mathbb{R}^d$ be a sequence of inputs (the superscript will be layer index, and the subscript will be token index). Then each layer l of our transformer computes the following

$$\begin{aligned} k_i^l &= U^l x_i^{l-1} \in \mathbb{R}^n & y_i^l &= \text{Attn}(k_i^l, k^l, k^l) + k_i^l & z_i^l &= \text{L}(y_i^l) \\ g_i^l &= W^{l1} z_i^l & h_i^l &= W^{l2}\phi(g_i^l) & x_i^l &= \text{L}(h_i^l + z_i^l) \end{aligned}$$

where U^l, W^{l1}, W^{l2} are weights, ϕ is nonlinearity (e.g. relu), $k^l = \{k_j^l\}_{j=1}^T$ and Attn and L are Attention and Layernorm as in Appendix D. The network outputs a single scalar from the average of the final embeddings x_i^L:

$$o = \frac{1}{T}\sum_{i=1}^{T} v^\top x_i^L/\sqrt{n}$$

To compute the Transformer-NTK, we will need to use the (simplified) NETSOR$\top^+$ Master Theorem (Theorem B.10) due to the presence of Layernorm and Attention.

Setup assume for all $\alpha, \beta \in [n]$,

- $W_{\alpha\beta}^{l1}, W_{\alpha\beta}^{l2} \sim \mathcal{N}(0, \sigma_w^2/n)$ for all $l \geq 1$
- $U_{\alpha\beta}^l \sim \mathcal{N}(0, \sigma_u^2/n)$ for all $l \geq 2$ and $U_{\alpha\beta}^1 \sim \mathcal{N}(0, \sigma_u^2/d)$
- $v_\alpha \sim \mathcal{N}(0, \sigma_v^2)$
- Assume Layernorm $\epsilon = 0$

Forward pass Suppose we have two sequences $\{x_1^0, \ldots, x_T^0\}$ and $\{\bar{x}_1^0, \ldots, \bar{x}_T^0\}$, and we use $\bar{\bullet}$ to denote quantities $\bullet$ computed on the second sequence. Then we see that $\{Z^{h_i^l}, Z^{\bar{h}_j^l}\}_{i,j}, \{Z^{g_i^l}, Z^{\bar{g}_j^l}\}_{i,j}, \{Z^{k_i^l}, Z^{\bar{k}_j^l}\}_{i,j}$ are mutually independent sets of random variables, each of which is jointly Gaussian with zero mean. Their covariances are given by

$$\begin{aligned}
\mathrm{Cov}(Z^{h_i^l}, Z^{\bar{h}_j^l}) &= \sigma_w^2\, \mathbb{E}\, \phi(Z^{g_i^l})\phi(Z^{\bar{g}_j^l}) \\
\mathrm{Cov}(Z^{g_i^l}, Z^{\bar{g}_j^l}) &= \sigma_w^2\, \mathbb{E}\, Z^{z_i^l} Z^{\bar{z}_j^l} \\
\mathrm{Cov}(Z^{k_i^l}, Z^{\bar{k}_j^l}) &= \sigma_u^2\, \mathbb{E}\, Z^{x_i^{l-1}} Z^{\bar{x}_j^{l-1}}
\end{aligned}$$

In addition,

$$Z^{x_i^l} = \frac{Z^{h_i^l} + Z^{z_i^l} - \mathbb{E}\, Z^{h_i^l} + Z^{z_i^l}}{std\left(Z^{h_i^l} + Z^{z_i^l}\right)} \qquad Z^{z_i^l} = \frac{Z^{y_i^l} - \mathbb{E}\, Z^{y_i^l}}{std(Z^{y_i^l})} \qquad Z^{y_i^l} = \sum_{j=1}^{T} \mathring{a}_{ij}^l Z^{k_j^l} + Z^{k_i^l}$$

where

$$(\mathring{a}_{i1}^l, \ldots, \mathring{a}_{iT}^l) = \mathrm{SoftMax}(\mathring{c}_{i1}^l, \ldots, \mathring{c}_{iT}^l), \qquad \mathring{c}_{ij}^l = \mathbb{E}\, Z^{k_i^l} Z^{k_j^l}.$$

We can easily see that $\mathbb{E}\, Z^{h_i^l} + Z^{z_i^l} = \mathbb{E}\, Z^{y_i^l} = 0$. So we can simplify

$$Z^{x_i^l} = \frac{Z^{h_i^l} + Z^{z_i^l}}{\sqrt{\mathbb{E}\left(Z^{h_i^l} + Z^{z_i^l}\right)^2}}, \qquad Z^{z_i^l} = \frac{Z^{y_i^l}}{\sqrt{\mathbb{E}(Z^{y_i^l})^2}}.$$

These equations yield a recursive way of computing all random variable $Z^{\bullet}$ associated to vector $\bullet$ in the forward pass.

Backward pass Backprop is given by the following equations.

$$\begin{aligned}
dx_i^L &= \frac{1}{T} v & dh_i^l &= d\mathrm{L}(dx_i^l \mid h_i^l + z_i^l) & dg_i^l &= \phi'(g_i^l) \odot W^{l2\top} dh_i^l \\
dz_i^l &= W^{l1\top} dg_i^l + dh_i^l & dy_i^l &= d\mathrm{L}(dz_i^l \mid y_i^l) & dx_i^{l-1} &= U^{l\top} dk_i^l
\end{aligned}$$

and

$$\begin{aligned}
dk_i^l = dy_i^l &+ d_{\mathrm{q}i}\, \mathrm{Attn}(\{dy_i^l\}_i \mid \{k_i^l\}_i, \{k_i^l\}_i, \{k_i^l\}_i) \\
&+ d_{\mathrm{k}i}\, \mathrm{Attn}(\{dy_i^l\}_i \mid \{k_i^l\}_i, \{k_i^l\}_i, \{k_i^l\}_i) \\
&+ d_{\mathrm{v}i}\, \mathrm{Attn}(\{dy_i^l\}_i \mid \{k_i^l\}_i, \{k_i^l\}_i, \{k_i^l\}_i)
\end{aligned}$$

This implies the following equations (via Box 2) for the associated random variables.

$$Z^{dh_i^l} = d\mathring{\mathrm{L}}\left(Z^{dx_i^l} \mid Z^{h_i^l} + Z^{z_i^l}\right), \quad Z^{dy_i^l} = d\mathring{\mathrm{L}}\left(Z^{dz_i^l} \mid Z^{y_i^l}\right), \quad \mathbb{E}\, Z^{dx_i^l} Z^{d\bar{x}_j^l} = \sigma_u^2\, \mathbb{E}\, Z^{dk_i^l} Z^{d\bar{k}_j^l}$$

$$\begin{aligned}
Z^{dk_i^l} = Z^{dy_i^l} &+ d_{\mathrm{q}i} \mathring{\mathrm{Attn}}(Z^{dy^l} \mid Z^{dk^l}, Z^{dk^l}, Z^{dk^l}) \\
&+ d_{\mathrm{k}i} \mathring{\mathrm{Attn}}(Z^{dy^l} \mid Z^{dk^l}, Z^{dk^l}, Z^{dk^l}) \\
&+ d_{\mathrm{v}i} \mathring{\mathrm{Attn}}(Z^{dy^l} \mid Z^{dk^l}, Z^{dk^l}, Z^{dk^l})
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}\, Z^{dx_i^L} Z^{d\bar{x}_j^L} &= T^{-2}\sigma_v^2 \\
\mathbb{E}\, Z^{dg_i^l} Z^{d\bar{g}_j^l} &= \sigma_w^2\, \mathbb{E}\, Z^{dh_i^l} Z^{d\bar{h}_j^l}\, \mathbb{E}\, \phi'(Z^{g_i^l})\phi'(Z^{\bar{g}_j^l}) \\
\mathbb{E}\, Z^{dz_i^l} Z^{d\bar{z}_j^l} &= \sigma_w^2\, \mathbb{E}\, Z^{dg_i^l} Z^{d\bar{g}_j^l} + \mathbb{E}\, Z^{dh_i^l} Z^{d\bar{h}_j^l}
\end{aligned}$$

where $Z^{dy^l} = \{Z^{dy_i^l}\}_i$, $Z^{dk^l} = \{Z^{dk_i^l}\}_i$, and $d\mathring{\mathrm{Attn}}$ and $d\mathring{\mathrm{L}}$ are as in Eqs. (32) and (34).

Some Simplifications Now we can write

$$Z^{dh_i^l} = d\mathring{\mathrm{L}}\left(Z^{dx_i^l} \mid Z^{h_i^l} + Z^{z_i^l}\right) = \frac{1}{std\left(Z^{h_i^l} + Z^{z_i^l}\right)}\mathrm{Center}\left(Z^{dx_i^l} - Z^{x_i^l}\,\mathbb{E}\, Z^{dx_i^l} Z^{x_i^l}\right)$$

as in Eq. (31). But because of the **MatMul** rule of Box 2, $Z^{dx_i^l}$ is a zero mean Gaussian independent from $Z^{x_i^l}$, and $\mathbb{E}\, Z^{dx_i^l} Z^{x_i^l} = 0$. Therefore $\mathrm{Center}\left(Z^{dx_i^l} - Z^{x_i^l}\,\mathbb{E}\, Z^{dx_i^l} Z^{x_i^l}\right) = Z^{dx_i^l}$, and

$$Z^{dh_i^l} = \frac{Z^{dx_i^l}}{std\left(Z^{h_i^l} + Z^{z_i^l}\right)}.$$

Likewise,

$$Z^{dy_i^l} = d\mathring{\mathrm{L}}\left(Z^{dz_i^l} \mid Z^{y_i^l}\right) = \frac{Z^{dz_i^l}}{std(Z^{y_i^l})}.$$

Finally, from Eq. (34), the $d_{\mathrm{q}i}\mathring{\mathrm{A}}\mathrm{ttn}$ and $d_{\mathrm{k}i}\mathring{\mathrm{A}}\mathrm{ttn}$ terms in $Z^{dk_i^l}$ depend linearly on $\{\mathbb{E}\, Z^{dy_i^l} Z^{k_j^l}\}_j$ which vanish again by the **MatMul** rule of Box 2. Therefore,

$$Z^{dk_i^l} = Z^{dy_i^l} + d_{\mathrm{v}i}\mathring{\mathrm{A}}\mathrm{ttn}(Z^{dy^l} \mid Z^{dk^l}, Z^{dk^l}, Z^{dk^l}) = \sum_j \mathring{a}_{ji}^l Z^{dy_j^l} + Z^{dy_i^l}$$

with $\mathring{a}_{ji}^l$ as computed in the forward pass.

The complete simplification:

$$Z^{dh_i^l} = \frac{1}{std\left(Z^{h_i^l} + Z^{z_i^l}\right)} Z^{dx_i^l} \qquad Z^{dy_i^l} = \frac{1}{std\left(Z^{z_i^l}\right)} Z^{dz_i^L} \qquad Z^{dk_i^l} = \sum_j \mathring{a}_{ji}^l Z^{dy_j^l} + Z^{dy_i^l}$$

$$\begin{aligned}
\mathbb{E}\, Z^{dg_i^l} Z^{d\bar{g}_j^l} &= \sigma_w^2\, \mathbb{E}\, Z^{dh_i^l} Z^{d\bar{h}_j^l}\, \mathbb{E}\, \phi'(Z^{g_i^l})\phi'(Z^{\bar{g}_j^l}) \\
\mathbb{E}\, Z^{dz_i^l} Z^{d\bar{z}_j^l} &= \sigma_w^2\, \mathbb{E}\, Z^{dg_i^l} Z^{d\bar{g}_j^l} + \mathbb{E}\, Z^{dh_i^l} Z^{d\bar{h}_j^l} \\
\mathbb{E}\, Z^{dx_i^l} Z^{d\bar{x}_j^l} &= \sigma_u^2\, \mathbb{E}\, Z^{dk_i^l} Z^{d\bar{k}_j^l}
\end{aligned}$$

As in Yang [62], all nonlinearities of the NETSOR$\top^+$ program (corresponding to nonlinearities and their derivatives in the network) are parameter controlled, and Assumption B.7 is satisfied. So Master Theorem holds, and the NTK has a well-defined almost sure limit which is given by Eq. (12). We can then summarize the above into the following vectorized formulas for computing this NTK limit.

E.5.1 NTK

Suppose we have a collection of M sequences $\{x_{a1}^0, \ldots, x_{aT}^0\}_{a=1}^M$ each with T tokens. We will use $a, b, \ldots$ as sequence indices and $i, j, \ldots$ as token indices. We will work with 4-tensors in $\mathbb{R}^{M\times T\times M\times T}$, which can be also thought of as $M \times M$ blocks of $T \times T$ matrices.

Notations For 4-tensor $C \in \mathbb{R}^{M\times T\times M\times T}$:

1. $BlockDiag(C)$ is the 4-tensor with $BlockDiag(C)_{aibi} = C_{aibj}\mathbb{I}(a = b)$.
2. $Diag(C)$ is the 4-tensor with $Diag(C)_{aibj} = C_{aibj}\mathbb{I}(a = b)\mathbb{I}(i = j)$.
3. Juxtaposition represents multiplication of tensors reshaped as matrices $C\bar{C} =$ `einsum('aibj,bjck->aick', C, C̄)`.
4. $Corr(C) = Diag(C)^{-1/2} C\, Diag(C)^{-1/2}$.
5. $\mathrm{SoftMax}(C)$ applies SoftMax to C in the last dimension.

NTK Computation the NTK, as a $M \times M$ matrix, is

$$\mathring{\Theta} = \frac{1}{T^2} \circ C^{x^L} + \sum_{l=1}^{L} D^{k^l} \circ C^{x^{l-1}} + D^{g^l} \circ C^{z^l} + D^{h^l} \circ V_\phi(C^{g^l})$$

where $X \circ Y$ is a matrix for 4-tensors X, Y, with $(X \circ Y)_{ab} = \sum_{ij} X_{aibj} Y_{aibj}$, and the relevant tensors are computed by

Forward:

$$\begin{aligned}
C^{x^0}_{aibj} &= x^\top_{ai} x_{bj}/d \\
C^{k^l} &= \sigma_u^2 C^{x^{l-1}} \\
A^l &= BlockDiag(\mathrm{SoftMax}(C^{k^l})) \\
C^{y^l} &= (A^l + I) C^{k^l} (A^{l\top} + I) \\
C^{z^l} &= Corr(C^{y^l_i}) \\
\Delta^{z^l} &= Diag(C^{y^l_i})^{-1/2} \\
C^{g^l} &= \sigma_w^2 C^{z^l} \\
C^{h^l} &= \sigma_w^2 V_\phi(C^{g^l}) \\
C^{x^l} &= Corr(C^{h^l} + C^{z^l}) \\
\Delta^{x^l} &= Diag(C^{h^l} + C^{z^l})^{-1/2}
\end{aligned}$$

Backward:

$$\begin{aligned}
D^{x^L}_{aibj} &= \sigma_v^2/T^2 \\
D^{h^l} &= \Delta^{x^l} D^{x^l} \Delta^{x^l} \\
D^{g^l} &= \sigma_w^2 D^{h^l} \odot V_{\phi'}(C^{g^l}) \\
D^{z^l} &= \sigma_w^2 D^{g^l} + D^{h^l} \\
D^{y^l} &= \Delta^{z^l} D^{z^l} \Delta^{z^l} \\
D^{k^l} &= (A^{l\top} + I) D^{y^l} (A^l + I) \\
D^{x^{l-1}} &= \sigma_u^2 D^{k^l}
\end{aligned}$$

F Theoretical Tools

We will use the following trivial but useful fact repeatedly.

Lemma F.1. *For an integer m, and complex numbers $a_i \in \mathbb{C}$, $i \in [k]$,*

$$\left| \sum_{i=1}^{k} a_i \right|^m \le k^{m-1} \sum_{i=1}^{k} |a_i|^m .$$

Proof. Expand the power in the LHS using the multinomial theorem, apply AM-GM to each summand, and finally aggregate using triangle inequality. □

F.1 Probability Facts

This section is largely the same as section G.1 of Yang [62]. All proofs can be found there.

Notations Given two random variables X, Y, and a σ-algebra $\mathcal{A}$, the notation $X \overset{\mathrm{d}}{=}_{\mathcal{A}} Y$ means that for any integrable function ϕ and for any random varible Z measurable on $\mathcal{A}$, $\mathbb{E}\,\phi(X)Z = \mathbb{E}\,\phi(Y)Z$. We say that X is distributed as (or is equal in distribution to) Y conditional on $\mathcal{A}$. In case $\mathcal{A}$ is the trivial σ-algebra, we just write $X \overset{\mathrm{d}}{=} Y$. The expression $X \xrightarrow{\mathrm{d}} Y$ (resp. $X \xrightarrow{\text{a.s.}} Y$) means X converges to Y in distribution (resp. almost surely).

Lemma F.2. *Let $\{X_n\}_{n \ge 1}$ be a sequence of random variables with zero mean. If for some $p \in \mathbb{N}$ and for all n, $\mathbb{E}\, X_n^{2p} \le c n^{-1-\rho}$, for some $\rho > 0$, then $X_n \to 0$ almost surely.*

The following is a standard fact about multivariate Gaussian conditioning

Proposition F.3. *Suppose $\mathbb{R}^{n_1+n_2} \ni x \sim \mathcal{N}(\mu, K)$, where we partition $x = (x_1, x_2) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$, $\mu = (\mu_1, \mu_2) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$, and $K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$. Then $x_1 \overset{\mathrm{d}}{=}_{x_2} \mathcal{N}(\mu|_{x_2}, K|_{x_2})$ where*

$$\begin{aligned}
\mu|_{x_2} &= \mu_1 - K_{12} K_{22}^{+} (x_2 - \mu_2) \\
K|_{x_2} &= K_{11} - K_{12} K_{22}^{+} K_{21}.
\end{aligned}$$

Lemma F.4 (Stein's lemma)**.** *For jointly Gaussian random variables Z_1, Z_2 with zero mean, and any function $\phi : \mathbb{R} \to \mathbb{R}$ where $\mathbb{E}\,\phi'(Z_1)$ and $\mathbb{E}\,Z_1\phi(Z_2)$ exists, we have*

$$\mathbb{E}\,Z_1\phi(Z_2) = \mathrm{Cov}(Z_1, Z_2)\,\mathbb{E}\,\phi'(Z_2).$$

Lemma F.5. *Let $\Phi : \mathbb{R}^n \to \mathbb{R}$ be measurable. Then for $z \sim \mathcal{N}(\zeta, \Sigma)$, the following Hessian and gradient matrices are equal:*

$$\frac{\mathrm{d}^2}{\mathrm{d}\zeta^2}\,\mathbb{E}\,\Phi(z) = 2\frac{\mathrm{d}}{\mathrm{d}\Sigma}\,\mathbb{E}\,\Phi(z)$$

whenever both sides exist.

F.2 Gaussian Conditioning Trick

Review of Moore-Penrose Pseudoinverse Let A^+ denote the Moore-Penrose pseudo-inverse of a matrix A.

Lemma F.6. *Let $A \in \mathbb{R}^{n\times m}$ be a matrix with random Gaussian entries, $A_{ij} \sim \mathcal{N}(0, \sigma^2)$. Consider fixed matrices $Q \in \mathbb{R}^{m\times q}, Y \in \mathbb{R}^{n\times q}, P \in \mathbb{R}^{n\times p}, X \in \mathbb{R}^{m\times p}$. Suppose there exists a solution in A to the equations $Y = AQ$ and $X = A^\top P$. Then the distribution of A conditioned on $Y = AQ$ and $X = A^\top P$ is*

$$A \overset{\mathrm{d}}{=}_{Y=AQ, X=A^\top P} E + \Pi_P^\perp \tilde{A} \Pi_Q^\perp$$

where

$$E = YQ^+ + P^{+\top}X^\top - P^{+\top}P^\top YQ^+,$$

$\tilde{A}$ is an iid copy of A, and $\Pi_P^\perp = I - \Pi_P$ and $\Pi_Q^\perp = I - \Pi_Q$ in which $\Pi_P = PP^+$ and $\Pi_Q = QQ^+$ are the orthogonal projection to the space spanned by the column spaces of P and Q respectively.

Proof. See Yang [62, Lemma G.7]. □

F.3 Law of Large Numbers for Images of Weakly Correlated Gaussians

Lemma F.7. *Let $\Pi \in \mathbb{R}^{n\times n}$ be an orthogonal projection matrix. Then each diagonal entry $\Pi_{ii} \in [0, 1]$.*

Proof. Because $\Pi = \Pi^2$, we have for each i, $\Pi_{ii} = \sum_j \Pi_{ij}^2 \implies \Pi_{ii}(1 - \Pi_{ii}) = \sum_{j\neq i} \Pi_{ij}^2 \geq 0 \implies \Pi_{ii} \in [0, 1]$. □

Theorem F.8. *Let $z \sim \mathcal{N}(0, \Pi)$ where $\Pi \in \mathbb{R}^{n\times n}$ is a matrix such that its correlation matrix $C = D^{-1/2}\Pi D^{-1/2}, D = \mathrm{Diag}(\Pi)$, has $\sum_{i<j} C_{ij}^2 \leq R$ for some constant R as $n \to \infty$. (So an orthogonal projection matrix of rank $n - O(1)$ satisfies this condition). Consider functions $\phi_i : \mathbb{R} \to \mathbb{R}$ for each $i \in [n]$ with mean $\mu_i = \mathbb{E}_x\,\phi_i(x)$ under $x \sim \mathcal{N}(0, \Pi_{ii})$. Suppose each ϕ_i has finite $(2p)$th centered moment $\mathbb{E}_x(\phi_i(x) - \mu_i)^{2p}$, for $x \sim \mathcal{N}(0, \Pi_{ii})$, where $p \geq 6$. Then for $Q \overset{\mathrm{def}}{=} \frac{1}{n}\sum_{i=1}^n \phi_i(z_i)$, as $n \to \infty$,*

$$\mathbb{E}[(Q - \mathbb{E}\,Q)^{2p}] \leq \mathcal{C}n^{-1.5} \max_{i\in[n]} \mathop{\mathbb{E}}_{x\sim\mathcal{N}(0,\Pi_{ii})} (\phi_i(x) - \mu_i)^{2p}$$

for some constant $\mathcal{C}$ depending on p and R, but not on n or the functions ϕ_i. If in addition, each ϕ_i has finite centered moments of order $2pL$ for some $L > 1$, then

$$\mathbb{E}[(Q - \mathbb{E}\,Q)^{2p}] \leq \mathcal{C}n^{-1.5+1/L} \sqrt[L]{\frac{1}{n}\sum_{i=1}^n \mathop{\mathbb{E}}_{x\sim\mathcal{N}(0,\Pi_{ii})} (\phi_i(x) - \mu_i)^{2pL}}.$$

Proof. See Yang [64]. □

G Proof of Main Theorem

In this section, we will give the proof for Theorem A.6 which is equivalent to Theorem 7.2. We reproduce the statement below

Theorem A.6 (BP-like NETSOR$\top$ Master Theorem)**.** *Fix any BP-like* NETSOR$\top$ *program satisfying Setup A.5 and with all nonlinearities polynomially-bounded. If $g^1, \dots, g^M$ are all of the G-vars in the entire program, including all input G-vars, then for any polynomially-bounded $\psi : \mathbb{R}^M \to \mathbb{R}$, as $n \to \infty$,*

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^M) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z) = \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z^{g^1}, \dots, Z^{g^M}),$$

where $\xrightarrow{\text{a.s.}}$ means almost sure convergence, $Z = (Z^{g^1}, \dots, Z^{g^M}) \in \mathbb{R}^M$, and $\mu = \{\mu(g^i)\}_{i=1}^M \in \mathbb{R}^M$ and $\Sigma = \{\Sigma(g^i, g^j)\}_{i,j=1}^M \in \mathbb{R}^{M\times M}$ are given in Eq. (18). See Fig. 1 for an illustration.

Comparison against NETSOR Master Theorem [62] We will follow the general outline of the inductive proof of NETSOR Master Theorem in Yang [62]. Here, the correlation of a matrix with its transpose (Remark G.1) causes additional difficulty in proving rank stability and zero stability properties (Appendix G.3) as well as the induction hypothesis (Moments(m)). The main sections dealing with these difficulties are Appendix G.2 which describes the setup for the induction, Appendix G.4.2 which proves part of rank and zero stability properties, and Appendices G.5.2 and G.5.3 which prove parts of the inductive step using the law of large numbers for weakly correlated random variables Theorem F.8.

A Bit of Notation and Terminology Note that, for each n, the randomness of our program specified by Theorem A.6 comes from the sampling of the input variables. Let $\mathcal{U}$ be the product space obtained from multiplying together the corresponding probability space for each n. Each sample from this product probability space thus correspond to a sequence $\{S(n)\}_n$ of instantiations of input variables. Below, when we say "almost surely" (often abbreviated "a.s."), we mean "almost surely over the probability of $\mathcal{U}$." We will also often make statements of the form

$$\textit{almost surely (or, a.s.), for all large } n, \quad \mathcal{A}(n) \textit{ is true}$$

where $\mathcal{A}(n)$ is a claim parametrized by n. This means that for all but a $\mathcal{U}$-probability-zero set of sequences $\{S(n)\}_n$ of input variable instantiations, $\mathcal{A}(n)$ is true for large enough n. Note that the order of the qualifiers is very important here.

We induct, but on what? A natural way of going about proving Theorem A.6 is by inducting on the number of variables in a program. It turns out this is not enough to prove our claim in its full generality, and it would be more fruitful to perform a simultaneous induction on our claim (Moments) along with another statement, parametrized by m,

Moments(m) For any polynomially-bounded $\psi : \mathbb{R}^m \to \mathbb{R}$, as $n \to \infty$,

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^m) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z).$$

CoreSet(m) There exists a "core set" $\mathcal{M} \subseteq [m]$ such that,

- **Basis**(m) almost surely, for large enough n, for every $i \in [m]$, there exist *unique* constants (not depending on n) $\{a_j\}_{j\in\mathcal{M}}$ such that $g^i = \sum_{j\in\mathcal{M}} a_j g^j$. Note the uniqueness implies that $\{g^i\}_{i\in\mathcal{M}}$ is linearly independent.
- **NullAvoid**(m) for every triangular array of Lesbegue measure zero sets $\{A_{n\alpha} \in \mathbb{R}^{\mathcal{M}}\}_{n\in\mathbb{N},\alpha\in[n]}$, almost surely for all large enough n, for all $\alpha \in [n]$, we have

  $$\{g_\alpha^i\}_{i\in\mathcal{M}} \notin A_{n\alpha}.$$

  In other words, the values $\{g_\alpha^i\}_{\alpha\in\mathcal{M}}$ of the core set "avoid" Lebesgue measure zero sets asymptotically. Intuitively, this says that the distribution of these values are not singular. (Note the LHS depends on n although we are suppressing it notationally)

Let us explain in brief why we need to consider CoreSet satisfying Basis and NullAvoid.

- Basis reduces the consideration of Moments to only the core set G-vars, since every other G-var is asymptotically a linear combination of them.
- When we apply the Gaussian conditioning technique Proposition F.3, we need to reason about the pseudo-inverse Λ^+ of some submatrix Λ of a covariance matrix. Each entry of Λ is of the form $\frac{1}{n}\sum_{\alpha=1}^{n}\phi_i(g_\alpha^1,\dots,g_\alpha^{m-1})\phi_j(g_\alpha^1,\dots,g_\alpha^{m-1})$ for a collection of polynomially bounded scalar functions $\{\phi_i\}_i$. This Λ will be a random variable which converges a.s. to a determinstic limit $\mathring{\Lambda}$ as $n\to\infty$. It should be generically true that $\Lambda^+ \xrightarrow{\text{a.s.}} \mathring{\Lambda}^+$ as well, which is essential to make the Gaussian conditioning argument go through. But in general, this is guaranteed only if Λ's rank doesn't drop suddenly in the $n\to\infty$ limit. We thus need to guard against the possibility that $g^1,\dots,g^m$, in the limit, suddenly concentrate on a small set on which $\{\phi_i(g^1,\dots,g^m)\}_i$ are linearly dependent. This is where NullAvoid comes in. It tells us that $g^1,\dots,g^m$ will avoid any such small set asymptotically, so that indeed the rank of Λ will not drop in the limit.

Proof organization We will show that Moments and CoreSet are true for input variables, as the base case, and

$$\text{Moments}(m-1) \text{ and } \text{CoreSet}(m-1) \implies \text{Moments}(m) \text{ and } \text{CoreSet}(m)$$

as the inductive step. By induction, we obtain Moments(M), which is Theorem A.6.

The base cases are easy and we will dispatch with them immediately after this in Appendix G.1, but the inductive step is much more complicated, and we will need to set up notation in Appendix G.2. During this setup, we prove some basic limit theorems using the induction hypothesis. However, the full generality of these claims requires some consequences of CoreSet, which we call "rank stability" and "zero stability." These notions are introduced and proved in Appendix G.3.

We would then finally be able to handle the inductive steps at this point. We first prove

$$\text{Moments}(m-1) \text{ and } \text{CoreSet}(m-1) \implies \text{CoreSet}(m)$$

in Appendix G.4 because it is easier. Then we prove

$$\text{Moments}(m-1) \text{ and } \text{CoreSet}(m-1) \implies \text{Moments}(m)$$

in Appendix G.5.

G.1 Base Cases: Moments and CoreSet for Input Variables

Base case: Moments(input vars) Suppose the input variables are $x^1,\dots,x^k : \mathsf{G}(n)$ (so that $\mu\in\mathbb{R}^k, \Sigma\in\mathbb{R}^{k\times k}$). We need to show that for any polynomially-bounded function $\psi:\mathbb{R}^k\to\mathbb{R}$,

$$\frac{1}{n}\sum_{\alpha=1}^{n}\psi(x_\alpha^1,\dots,x_\alpha^k) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)}\psi(Z),$$

where ψ on the RHS ignores all coordinates corresponding to non-input G-vars. Since μ and Σ restricted to input variables are just μ and Σ (see Eq. (18)), the RHS expectation is just

$$\mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)}\psi(Z) = \mathop{\mathbb{E}}_{Z^{in}\sim\mathcal{N}(\mu,\Sigma)}\psi(Z^{in})$$

and the almost sure convergence we desire is just a result of the law of large numbers.

Base Case: CoreSet(input vars) Let $x^1,\dots,x^k$ be the input G-vars as above. Pick the core set $\mathcal{M}$ to be any subset of $[k]$ such that $\operatorname{rank}\Sigma|_{\mathcal{M}} = \operatorname{rank}\Sigma$. Then it's straightforward to verify Basis and NullAvoid.

G.2 Inductive Case: Setup

We now assume Moments$(m-1)$ and CoreSet$(m-1)$ and want to reason about g^m to show Moments(m) and CoreSet(m). Suppose

$$g^m := Ah \quad \text{where} \quad A : \mathsf{A}(n,n) \text{ and } h : \mathsf{H}(n) \text{ was introduced by } h := \phi(g^1,\dots,g^{m-1})$$

(WLOG padding input slots if necessary; if $h = g^i$ is a G-var, then just let ϕ be the projection to the ith coordinate). For brevity, we will just write $g = g^m$. Consider all previous instances where A or $A^\top$ is used:

$$\hat{g}^i := A\hat{h}^i, i = 1, \dots, r, \quad \text{and} \quad \check{g}^j := A^\top \check{h}^j, j = 1, \dots, s.$$

Define

$$\hat{G} \stackrel{\text{def}}{=} [\hat{g}^1 | \dots | \hat{g}^r] \in \mathbb{R}^{n \times r}, \check{G} \stackrel{\text{def}}{=} [\check{g}^1 | \dots | \check{g}^s] \in \mathbb{R}^{n \times s}, \hat{H} \stackrel{\text{def}}{=} [\hat{h}^1 | \dots | \hat{h}^r], \check{H} \stackrel{\text{def}}{=} [\check{h}^1 | \dots | \check{h}^s]. \quad (40)$$

We will also use $\hat{G}$ to denote the *set* of G-vars $\{\hat{g}^1, \dots, \hat{g}^r\}$ when we later write expressions like $\Sigma(\hat{G}, \hat{G})$. Let $\mathcal{B}$ be the σ-algebra spanned by all previous G-vars $g^1, \dots, g^{m-1}$ (and hence also all previous H-vars). Conditioning on $\mathcal{B}$, A is constrained by $\hat{G} = A\hat{H}, \check{G} = A^\top \check{H}$, and we have by Lemma F.6,

$$g \stackrel{\text{d}}{=}_{\mathcal{B}} (E + \Pi_{\hat{H}}^\perp \tilde{A} \Pi_{\check{H}}^\perp) h$$

where

$$\begin{aligned} E &= \hat{G}\hat{H}^+ + \check{H}^{+\top}\check{G}^\top - \check{H}^{+\top}\check{G}^\top \hat{H}\hat{H}^+ \\ &= \hat{G}(\hat{H}^\top \hat{H})^+ \hat{H}^\top + \check{H}(\check{H}^\top \check{H})^+ \check{G}^\top - \check{H}(\check{H}^\top \check{H})^+ \check{G}^\top \hat{H}(\hat{H}^\top \hat{H})^+ \hat{H}^\top, \end{aligned} \quad (41)$$

$\tilde{A}$ is an independent copy of A and $\Pi_{\hat{H}} = \hat{H}\hat{H}^+ = \hat{H}(\hat{H}^\top \hat{H})^+ \hat{H}^\top$ is the projection to the column space of $\hat{H}$ (likewise for $\Pi_{\check{H}}$).

Remark G.1. Note if `Trsp` is not allowed (as in NETSOR Master Theorem [62]), then this would simplify a lot to $g \stackrel{\text{d}}{=}_{\mathcal{B}} (\hat{G}\hat{H}^+ + \tilde{A}\Pi_{\hat{H}}^\perp)h$. In particular, compared to the NETSOR Master Theorem, we cannot straightforwardly think of g as having iid Gaussian coordinates because of the projection $\Pi_{\check{H}}^\perp$ in front of $\tilde{A}$. Most of the added complexity of this proof of NETSOR$\top$ Master Theorem comes from this fact.

Remark G.2. On the other hand, with the BP-like assumption, E is roughly equal to $\hat{G}\hat{H}^+$; see Lemma G.5. However, this is very far from true when we don't assume the program is BP-like.

If we define

$$\omega \stackrel{\text{def}}{=} Eh, \quad \sigma \stackrel{\text{def}}{=} \sigma_A \sqrt{\|\Pi_{\check{H}}^\perp h\|^2 / n} \quad (42)$$

then

$$g \stackrel{\text{d}}{=}_{\mathcal{B}} \omega + \sigma \Pi_{\check{H}}^\perp y, \text{ with } y \sim \mathcal{N}(0, I_n) \quad (43)$$

For brevity, we will define the following matrices and vectors of fixed dimension

$$\begin{aligned} &\hat{\Lambda} \stackrel{\text{def}}{=} \hat{H}^\top \hat{H}/n \in \mathbb{R}^{r \times r} \quad \check{\Lambda} \stackrel{\text{def}}{=} \check{H}^\top \check{H}/n \in \mathbb{R}^{s \times s} \quad \Gamma \stackrel{\text{def}}{=} \check{G}^\top \hat{H}/n \in \mathbb{R}^{s \times r} \\ &\hat{\eta} \stackrel{\text{def}}{=} \hat{H}^\top h/n \in \mathbb{R}^r \qquad \check{\eta} \stackrel{\text{def}}{=} \check{G}^\top h/n \in \mathbb{R}^s. \end{aligned} \quad (44)$$

Suppose $\hat{h}^i$ was introduced by $\hat{h}^i := \hat{\phi}^i(g^1, \dots, g^M)$, and $\check{h}^j$ was introduced by $\check{h}^j := \check{\phi}^j(g^1, \dots, g^M)$, where $\hat{\phi}^i$ and $\check{\phi}^j$ depend at most on $g^1, \dots, g^{m-1}$. By induction hypothesis Moments$(m-1)$, $\hat{\Lambda}, \check{\Lambda}, \Gamma, \hat{\eta}, \check{\eta}$ all converge a.s. to corresponding limit values $\mathring{\hat{\Lambda}}, \mathring{\check{\Lambda}}, \mathring{\hat{\eta}}$, since their entries are moments of $Z^1, \dots, Z^{m-1}$:

$$\begin{aligned} \hat{\Lambda}_{ij} &\xrightarrow{\text{a.s.}} \mathring{\hat{\Lambda}}_{ij} \stackrel{\text{def}}{=} \mathbb{E}\, \hat{\phi}^i(Z)\hat{\phi}^j(Z) = (\sigma_A)^{-2}\Sigma(\hat{g}^i, \hat{g}^j) \\ \check{\Lambda}_{ij} &\xrightarrow{\text{a.s.}} \mathring{\check{\Lambda}}_{ij} \stackrel{\text{def}}{=} \mathbb{E}\, \check{\phi}^i(Z)\check{\phi}^j(Z) = (\sigma_A)^{-2}\Sigma(\check{g}^i, \check{g}^j) \\ \hat{\eta}_i &\xrightarrow{\text{a.s.}} \mathring{\hat{\eta}}_i \stackrel{\text{def}}{=} \mathbb{E}\, \hat{\phi}^i(Z)\phi(Z) = (\sigma_A)^{-2}\Sigma(\hat{g}^i, g) \end{aligned}$$

and $\Gamma \xrightarrow{\text{a.s.}} 0, \check{\eta} \xrightarrow{\text{a.s.}} 0$ because, by the BP-like assumption, $\check{G}$ is odd in some input G-vars $v^1, \dots, v^k$ independent from all other input G-vars, so that the limiting expectation is 0.

It turns out that, as a consequence of Lemma G.6 below, a.s. for all large enough n, $\operatorname{rank} \hat{\Lambda} = \operatorname{rank} \mathring{\hat{\Lambda}}$ and $\operatorname{rank} \check{\Lambda} = \operatorname{rank} \mathring{\check{\Lambda}}$. Therefore, as pseudoinverse is continuous on matrices of fixed rank, we get the following proposition

Proposition G.3. $\hat{\Lambda}^+ \xrightarrow{\text{a.s.}} \mathring{\hat{\Lambda}}^+$ *and* $\check{\Lambda}^+ \xrightarrow{\text{a.s.}} \mathring{\check{\Lambda}}^+$.

Using this proposition, we compute the limits of the conditional mean ω and variance σ^2.

Lemma G.4. $\sigma^2 \xrightarrow{\text{a.s.}} \mathring{\sigma}^2 \stackrel{\text{def}}{=} \Sigma(g,g) - \Sigma(g,\hat{G})\Sigma(\hat{G},\hat{G})^+\Sigma(\hat{G},g)$

Proof. Note that

$$\sigma^2 = \frac{\sigma_A^2}{n}(h^\top h - h^\top \Pi_{\hat{H}} h) = \frac{\sigma_A^2}{n}(h^\top h - h^\top \hat{H}(\hat{H}^\top \hat{H})^+ \hat{H}^\top h) = \frac{\sigma_A^2}{n}(h^\top h - \hat{\eta}^\top \hat{\Lambda}^+ \hat{\eta}).$$

Because ϕ is polynomially-bounded, so is $\phi(z)^2$ as well. By induction hypothesis,

$$\frac{1}{n}h^\top h = \frac{1}{n}\sum_{\alpha=1}^{n} \phi(g_\alpha^1, \ldots, g_\alpha^{m-1})^2 \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \phi(Z)^2 = \sigma_A^{-2}\Sigma(g,g).$$

Likewise, $\hat{\eta} \xrightarrow{\text{a.s.}} \mathring{\hat{\eta}}$ and $\hat{\Lambda} \xrightarrow{\text{a.s.}} \mathring{\hat{\Lambda}}$. By Proposition G.3, $\hat{\Lambda}^+ \xrightarrow{\text{a.s.}} \mathring{\hat{\Lambda}}^+$. Combining all of these limits together yields the desired claim. □

Lemma G.5. *Let* $v \stackrel{\text{def}}{=} \hat{\Lambda}^+\hat{\eta}$, *so that* $v \xrightarrow{\text{a.s.}} \mathring{v} \stackrel{\text{def}}{=} \mathring{\hat{\Lambda}}^+\mathring{\hat{\eta}}$. *Then for some vector* $\hat{\varepsilon} \in \mathbb{R}^r, \check{\varepsilon} \in \mathbb{R}^s$ *that go to 0 a.s. with* n, $\omega = Eh = \hat{G}(\mathring{v} + \hat{\varepsilon}) + \check{H}\check{\varepsilon}$

Proof. Using Eqs. (41) and (44), we can re-express ω as

$$\omega = \hat{G}\hat{\Lambda}^+\hat{\eta} + \check{H}\check{\Lambda}^+\check{\eta} - \check{H}\check{\Lambda}^+\Gamma\hat{\Lambda}^+\hat{\eta}.$$

Because $\Gamma \xrightarrow{\text{a.s.}} 0, \check{\eta} \xrightarrow{\text{a.s.}} 0$ as discussed above, we can set $\check{\varepsilon} \stackrel{\text{def}}{=} \check{\Lambda}^+\check{\eta} - \check{\Lambda}^+\Gamma\hat{\Lambda}^+\hat{\eta}$, so that $\check{\varepsilon} \xrightarrow{\text{a.s.}} 0$.

In addition, by Proposition G.3, $\hat{\Lambda}^+ \xrightarrow{\text{a.s.}} \mathring{\hat{\Lambda}}^+$, so that setting $\hat{\varepsilon} \stackrel{\text{def}}{=} v - \mathring{v}$, we get $\hat{\varepsilon} \xrightarrow{\text{a.s.}} 0$.

Altogether, we have

$$\omega = \hat{G}(\mathring{v} + \hat{\varepsilon}) + \check{H}\check{\varepsilon}$$

as desired. □

G.3 Rank Stability and Zero Stability

In this section, we prove the following consequence of CoreSet$(m-1)$ and Moments$(m-1)$.

Lemma G.6 (Rank Stability)**.** *For any collection of polynomially-bounded functions* $\{\psi_j : \mathbb{R}^{m-1} \to \mathbb{R}\}_{j=1}^l$, *let* $K \in \mathbb{R}^{l\times l}$ *be the random matrix (depending on* n*) defined by*

$$K_{ij} = \frac{1}{n}\sum_{\alpha=1}^{n} \psi_i(g_\alpha^1, \ldots, g_\alpha^{m-1})\psi_j(g_\alpha^1, \ldots, g_\alpha^{m-1}).$$

By Moments$(m-1)$,

$$K \xrightarrow{\text{a.s.}} \mathring{K}$$

for some matrix $\mathring{K} \in \mathbb{R}^{l\times l}$.

1. *Then, almost surely, for large enough* n,

   $$\ker K = \ker \mathring{K}, \quad \operatorname{im} K = \operatorname{im} \mathring{K}, \quad \textit{and} \quad \operatorname{rank} K = \operatorname{rank} \mathring{K}.$$

   Here ker *denotes null space and* im *denotes image space.*

2. *Suppose* $I \subseteq [l]$ *is any subset such that* $\mathring{K}|_I$, *the restriction of* $\mathring{K}$ *to rows and columns corresponding to* I, *satisfies*

   $$|I| = \operatorname{rank} \mathring{K}|_I = \operatorname{rank} \mathring{K}.$$

   There are unique coefficients $\{F_{ij}\}_{i\in[l], j\in I}$ *that expresses each row of* $\mathring{K}$ *as linear combinations of rows corresponding to* I*:*

   $$\forall i \in [l], \quad \mathring{K}_i = \sum_{j\in I} F_{ij}\mathring{K}_j.$$

Then, a.s. for all large n, for all $\alpha \in [n]$,

$$\psi_i(g_\alpha^1, \dots, g_\alpha^{m-1}) = \sum_{j \in I} F_{ij} \psi_j(g_\alpha^1, \dots, g_\alpha^{m-1}).$$

This will be primarily a corollary of the following Lemma G.7.

Lemma G.7 (Zero Stability)**.** *If $\psi : \mathbb{R}^{m-1} \to \mathbb{R}^{\geq 0}$ is a nonnegative function such that*

$$\frac{1}{n} \sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^{m-1}) \xrightarrow{\text{a.s.}} 0$$

then, almost surely, for large enough n,

$$\psi(g_\alpha^1, \dots, g_\alpha^{m-1}) = 0$$

for all $\alpha \in [n]$.

We give the proof of Lemma G.6 now, assuming Lemma G.7.

Proof. Let $v \in \mathbb{R}^l$ be in the null space of $\mathring{K}$, i.e. $v^\top \mathring{K} v = 0$. Then we also have $v^\top K v \xrightarrow{\text{a.s.}} v^\top \mathring{K} v = 0$. But

$$v^\top K v = \frac{1}{n} \sum_{\alpha=1}^{n} \Psi(g_\alpha^1, \dots, g_\alpha^{m-1}), \quad \text{where} \quad \Psi(g_\alpha^1, \dots, g_\alpha^{m-1}) \stackrel{\text{def}}{=} \left(\sum_{i=1} v_i \psi_i(g_\alpha^1, \dots, g_\alpha^{m-1}) \right)^2$$

and Ψ is a nonnegative function. By Lemma G.7, we have that: almost surely, for large enough n,

$$\Psi(g_\alpha^1, \dots, g_\alpha^{m-1}) = 0 \quad \text{for all } \alpha \in [n] \quad \implies v^\top K v = 0$$

Claim 1. If we apply this argument to a basis $\{v^1, \dots, v^t\}$ of $\ker \mathring{K}$, then we get,

$$\text{a.s. for all large } n, \quad \ker \mathring{K} \subseteq \ker K,$$

so that

$$\text{a.s. for all large } n, \quad \operatorname{rank} \mathring{K} \geq \operatorname{rank} K.$$

Because the rank function is lower semicontinuous (i.e. the rank can drop suddenly, but cannot increase suddenly), and $K \xrightarrow{\text{a.s.}} \mathring{K}$, we also have

$$\text{a.s. for all large } n, \quad \operatorname{rank} \mathring{K} \leq \operatorname{rank} K.$$

Combined with the above, this gives the desired result on rank. The equality of null space then follows from the equality of rank, and the equality of image space follows immediately, as the image space is the orthogonal complement of the null space.

Claim 2. If we apply the above argument to each v^i defined by inner product as

$$\forall x \in \mathbb{R}^l, \quad x^\top v^i \stackrel{\text{def}}{=} x_i - \sum_{j \in I} F_{ij} x_j,$$

(note that only for $i \notin I$ is v^i nonzero), then we have, a.s. for large n, ${v^i}^\top K v^i = 0$, or

$$\psi_i(g_\alpha^1, \dots, g_\alpha^{m-1}) = \sum_{j \in I} F_{ij} \psi_j(g_\alpha^1, \dots, g_\alpha^{m-1}).$$

□

In the rest of this section, we prove Lemma G.7. It helps to first show that the linear relations given in Basis carries over to the $n \to \infty$ limit.

Proposition G.8. *Let $\Sigma|_{\mathcal{M}}$ be the submatrix of Σ with rows and columns corresponding to $\{g^i : i \in \mathcal{M}\}$. Then $\operatorname{rank} \Sigma = \operatorname{rank} \Sigma|_{\mathcal{M}} = |\mathcal{M}|$. Furthermore, if $Z = (Z^1, \dots, Z^{m-1}) \sim \mathcal{N}(\mu|_{m-1}, \Sigma|_{m-1})$, where $\mu|_{m-1}, \Sigma|_{m-1}$ are the restrictions of μ, Σ to $g^1, \dots, g^{m-1}$, then for each i,*

$$Z^i \stackrel{\text{d}}{=} \sum_{j \in \mathcal{M}} a_j Z^j$$

where $\{a_j\}_{j \in \mathcal{M}}$ are the coefficients corresponding to g^i given in Basis.

Proof. By Basis property, each $g^i, i \in \mathcal{M}$, has a set of unique constants $\{a_j\}_{j\in\mathcal{M}}$ (independent of n) such that, almost surely, for large enough n,

$$g^i = \sum_{j\in\mathcal{M}} a_j g^j.$$

Let $\psi(x^1, \dots, x^{m-1}) \stackrel{\text{def}}{=} (x^i - \sum_{j\in\mathcal{M}} a_j x^j)^2$. Then by Basis$(m-1)$ and Moments$(m-1)$,

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^{m-1}) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu|_{m-1},\Sigma|_{m-1})} \psi(Z) = 0.$$

where $\mu|_{m-1}, \Sigma|_{m-1}$ are the restrictions of μ, Σ to $g^1, \dots, g^{m-1}$. This implies that for $Z = (Z^1, \dots, Z^{m-1}) \sim \mathcal{N}(\mu|_{m-1}, \Sigma|_{m-1})$,

$$Z^i \stackrel{\text{d}}{=} \sum_{j\in\mathcal{M}} a_j Z^j.$$

Repeating this argument for all $i \in [m-1]$ implies that $\{Z^j\}_{j\in\mathcal{M}}$ is a "spanning set" of $Z^1, \dots, Z^{m-1}$. Furthermore, by the uniqueness of the coefficients, we also have that $\{Z^j\}_{j\in\mathcal{M}}$ is linearly independent as well. This then implies the rank consequence we want. □

Now we show Lemma G.7.

Proof of Lemma G.7. By Moments$(m-1)$,

$$\frac{1}{n}\sum_{\alpha=1}^{n} \psi(g_\alpha^1, \dots, g_\alpha^{m-1}) \to \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu|_{m-1},\Sigma|_{m-1})} \psi(Z).$$

By Proposition G.8, if $Z \sim \mathcal{N}(\mu|_{m-1}, \Sigma|_{m-1})$ and $Z|_{\mathcal{M}}$ is the part of Z corresponding to $\mathcal{M}$, then

$Z|_{\mathcal{M}}$ has density. The law of $Z|_{\mathcal{M}}$ (namely $\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})$, where $\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}}$ are the restriction of μ and Σ to $\mathcal{M}$) is absolutely continuous against the Lebesgue measure of $\mathbb{R}^{\mathcal{M}}$ and vice versa, so that a set of Lebesgue measure zero is measure zero under $\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})$, and vice versa; and

$Z|_{\mathcal{M}}$ is basis of Z. Basis yields a linear function λ such that $\lambda(\{g_\alpha^j\}_{j\in\mathcal{M}}) = \{g_\alpha^i\}_{i=1}^{m-1}$ for all α, almost surely asymptotically, and $\lambda(Z|_{\mathcal{M}}) \stackrel{\text{d}}{=} Z$, so that

$$\mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu|_{m-1},\Sigma|_{m-1})} \psi(Z) = \mathop{\mathbb{E}}_{Z'\sim\mathcal{N}(\mu|_{\mathcal{M}},\Sigma|_{\mathcal{M}})} \psi\circ\lambda(Z').$$

This expectation is 0 by our premise.

Because ψ, and thus $\psi\circ\lambda$, is a nonnegative function, the nullity of the expectation implies that, other than a set U of $\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})$-measure zero, $\psi\circ\lambda$ is 0. This set U also has Lebesgue measure zero as $Z|_{\mathcal{M}}$ has density, by our reasoning above.

If in NullAvoid, we set $A_{n\alpha} = U$ for all n and all $\alpha \in [n]$, then we get that: almost surely, for all large enough n, for all $\alpha \in [n]$,

$$\{g_\alpha^i\}_{i\in\mathcal{M}} \notin U \iff \psi\circ\lambda(\{g_\alpha^i\}_{i\in\mathcal{M}}) = 0 \iff \psi(g_\alpha^1, \dots, g_\alpha^{m-1}) = 0,$$

as desired. □

G.4 Inductive Step: CoreSet(m)

In this section, we show

$$\text{Moments}(m-1) \text{ and } \text{CoreSet}(m-1) \implies \text{CoreSet}(m).$$

More explicitly, we need to think about whether to add m to the core set $\mathcal{M}$ of $[m-1]$ in order to maintain the Basis and NullAvoid properties.

We proceed by casework on whether $\mathring{\sigma} = 0$.

G.4.1 If $\mathring{\sigma} = 0$.

We will show that the core set properties are maintained if we don't add m to the core set.

Consider the space $\mathcal{L} \stackrel{\text{def}}{=} L^2(\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}}))$ of square-integrable real functions against the measure $\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})$ defined on $\mathbb{R}^{\mathcal{M}}$. Let $\langle \phi, \psi \rangle = \mathbb{E}_{Y \sim \mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})} \phi(Y)\psi(Y)$ be the inner product of this space. Just like in a finite-dimensional inner product space, given a finite collection of functions $S = \{\psi^i\}_{i=1}^k$, the orthogonal projection operator Π_S to the span of S (inside $\mathcal{L}$) is given by

$$\Pi_S \phi = \sum_{i=1}^{k} a_i \psi^i,$$

for any $\phi \in \mathcal{L}$, where

$$\begin{aligned} a &= \Lambda^+ b \in \mathbb{R}^k, \\ b_j &= \langle \psi^j, \phi \rangle, b \in \mathbb{R}^k, \\ \Lambda_{ij} &= \langle \psi^i, \psi^j \rangle, \Lambda \in \mathbb{R}^{k \times k}. \end{aligned}$$

Recall that $g = Ah$ where h was introduced by $h := \phi(g^1, \ldots, g^{m-1})$, for some polynomially-bounded ϕ, and likewise $\hat{g}^i = A\hat{h}^i$ where $\hat{h}^i = \hat{\phi}^i(g^1, \ldots, g^{m-1})$, for each $i \in [r]$. By Basis, we know that, a.s. for large enough n, each of $g^1, \ldots, g^{m-1}$ is a (unique, constant-in-n) linear combination of $\{g^j\}_{j \in \mathcal{M}}$. Therefore, we can express

$$h = \underline{\phi}(\{g^j\}_{j \in \mathcal{M}}), \quad \text{and} \quad \forall i \in [r], \hat{h}^i = \underline{\hat{\phi}^i}(\{g^j\}_{j \in \mathcal{M}})$$

for some functions $\underline{\phi}, \underline{\hat{\phi}^i} \in \mathcal{L}$. For convenience, set $S \stackrel{\text{def}}{=} \{\underline{\hat{\phi}^i}\}_i$.

One can see then, as in the proof of Lemma G.4,

$$\mathring{\sigma}^2 = \sigma_A^2(\mathbb{E}\,\phi(Z)^2 - \mathring{\hat{\eta}}^\top \mathring{\hat{\Lambda}}^+ \mathring{\hat{\eta}}) = \sigma_A^2(\langle \underline{\phi}, \underline{\phi} \rangle - \langle \underline{\phi}, \Pi_S \underline{\phi} \rangle)$$

by expanding the definition of $\mathring{\hat{\eta}}$ and $\mathring{\hat{\Lambda}}$. Therefore, $\mathring{\sigma} = 0$ implies that

$$\langle \underline{\phi}, \underline{\phi} \rangle = \langle \underline{\phi}, \Pi_S \underline{\phi} \rangle$$

so that: after changing its values on a set U of measure zero under $\mathcal{N}(\mu|_{\mathcal{M}}, \Sigma|_{\mathcal{M}})$ (and thus also under Lebesgue measure by Lemma G.6), $\underline{\phi}$ is a linear combination of $\{\underline{\hat{\phi}^i}\}_{i=1}^r$, i.e.

$$\forall \vec{x} \notin U, \underline{\phi}(\vec{x}) = \sum_{i \in [r]} c_i \underline{\hat{\phi}^i}(\vec{x})$$

for some coefficients $\{c_i\}_{i \in [r]}$. By NullAvoid applied to $A_{n\alpha} = U$ for all n and $\alpha \in [n]$, we also have that: a.s. for large enough n,

$$\phi(g^1, \ldots, g^\alpha) = \underline{\phi}(\{g^j\}_{j \in \mathcal{M}}) = \sum_{i \in [r]} c_i \underline{\hat{\phi}^i}(\{g^j\}_{j \in \mathcal{M}}) = \sum_{i \in [r]} c_i \hat{\phi}^i(g^1, \ldots, g^\alpha),$$

and therefore, under the same condition, (recall A is the matrix giving rise to g in $g := Ah$)

$$g = A\phi(g^1, \ldots, g^\alpha) = \sum_{i \in [r]} c_i A\hat{\phi}^i(g^1, \ldots, g^\alpha) = \sum_{i \in [r]} c_i \hat{g}^i.$$

This shows that, if we keep the core set as $\mathcal{M}$, then Basis is still satisfied. Since the core set is not changing, NullAvoid just follows from the induction hypothesis.

For usage later in the proof of Moments(m), we record our observation here as follows

Lemma G.9. *If $\mathring{\sigma} = 0$, then there are coefficients $\{c_i\}_{i=1}^r$ independent of n such that a.s. for large enough n,*

$$g = \sum_{i \in [r]} c_i \hat{g}^i.$$

G.4.2 If $\mathring{\sigma} > 0$.

It's clear that g cannot be in the linear span of $\{\hat{g}^i\}_{i\in[r]}$ asymptotically, so we will add g to the core set, and the Basis property follows immediately. In the below, we shall write $\mathcal{M}$ for the old core set, and $\mathcal{M}' \stackrel{\text{def}}{=} \mathcal{M} \cup \{g\}$ for the new one.

It remains to show NullAvoid for $\mathcal{M}'$. First, let's assume that, a.s. for large enough n, $\Pi_{\check{H}}^{\perp}$ has no zero diagonal entry; we shall show this fact below in Lemma G.10. Because the conditional variance of g_α^m given $g^1, \dots, g^{m-1}$ is $\sigma^2(\Pi_{\check{H}}^{\perp})_{\alpha\alpha}$, and because $\mathring{\sigma} > 0$, this assumption implies that, a.s. for all large enough n,

$$g_\alpha^m | g^1, \dots, g^{m-1} \text{ has density for all } \alpha \in [n]. \tag{45}$$

By "has density" here, we in particular mean that any Lesbegue measure zero set in $\mathbb{R}$ has zero probability under the conditional distribution of g_α^m given $g^1, \dots, g^{m-1}$.

Now, assuming Lemma G.10, we prove NullAvoid holds for $\mathcal{M}'$.

Let $\{A_{n\alpha} \subseteq \mathbb{R}^{\mathcal{M}'}\}_{n\in\mathbb{N},\alpha\in[n]}$ be a triangular array of Lesbegue measure zero sets. For each $A_{n\alpha}$, define $B_{n\alpha} \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{R}^{\mathcal{M}} : \lambda(A_{n\alpha}|_{\vec{x}}) \neq 0\}$, where $A_{n\alpha}|_{\vec{x}} = \{y \in \mathbb{R} : (\vec{x}, y) \in A_{n\alpha} \subseteq \mathbb{R}^{\mathcal{M}} \times \mathbb{R}\}$ is the "slice" of $A_{n\alpha}$ at $\vec{x}$, and λ is the 1-dimensional Lebesgue measure. Because each $A_{n\alpha}$ has measure zero in $\mathbb{R}^{\mathcal{M}'}$, necessarily each $B_{n\alpha}$ also has measure zero in $\mathbb{R}^{\mathcal{M}}$. Applying NullAvoid to the triangular array $\{B_{n\alpha} \subseteq \mathbb{R}^{\mathcal{M}}\}_{n\in\mathbb{N},\alpha\in[n]}$, we get that: a.s. for large enough n,

$$\forall \alpha \in [n], \{g_\alpha^i\}_{i\in\mathcal{M}} \notin B_{n\alpha}.$$

Therefore, by Eq. (45), a.s. for large enough n,

$$\forall \alpha \in [n], \{g_\alpha^i\}_{i\in\mathcal{M}'} \notin A_{n\alpha}.$$

This finishes the proof of NullAvoid for $\mathcal{M}'$, and also CoreSet(m), save for Lemma G.10 below.

Lemma G.10. *Almost surely, for large enough n, $\Pi_{\check{H}}^{\perp}$ has no zero diagonal entry.*

Proof. WLOG, assume $\mathring{\check{\Lambda}}$ is full rank. Otherwise, by Lemma G.6(2), we can replace $\check{h}^1, \dots, \check{h}^s$ by a linearly independent spanning set $\check{h}^{i_1}, \dots, \check{h}^{i_k}$ such that 1) each $\check{h}^j$ is almost surely, for all large n, a linear combination of them and such that 2) their 2nd moment matrix is full rank in the limit. Then the projection matrix associated to $\check{h}^{i_1}, \dots, \check{h}^{i_k}$ is, almost surely, for all large n, the same as $\Pi_{\check{H}}$.

By the Sherman-Morrison formula (Fact G.11),

$$(\Pi_{\check{H}})_{\alpha\alpha} = f\left(\frac{1}{n}\check{h}_\alpha{}^\top \check{\Lambda}_{-\alpha}^{-1}\check{h}_\alpha\right)$$

where $f(x) = x/(1+x)$, $\check{h}_\alpha$ is the column vector $(\check{h}_\alpha^1, \dots, \check{h}_\alpha^s)^\top$, and $\check{\Lambda}_{-\alpha} = \frac{1}{n}\sum_{\beta\neq\alpha}\check{h}_\beta\check{h}_\beta{}^\top$. Thus, unless $\check{\Lambda}_{-\alpha}$ is singular for some α, all diagonal entries of $\Pi_{\check{H}}^{\perp} = I - \Pi_{\check{H}}$ are nonzero. So it suffices to show that,

$$\text{a.s. for large enough } n, \quad \check{\Lambda}_{-\alpha} \text{ is nonsingular for all } \alpha.$$

To do this, it pays to note that $\check{\Lambda}_{-\alpha} = \check{\Lambda} - \frac{1}{n}\check{h}_\alpha\check{h}_\alpha^\top$, so that

$$|\lambda_{\min}(\check{\Lambda}_{-\alpha}) - \lambda_{\min}(\check{\Lambda})| \leq \|\frac{1}{n}\check{h}_\alpha\check{h}_\alpha^\top\|_{\text{op}} = \frac{1}{n}\check{h}_\alpha^\top\check{h}_\alpha.$$

By Lemma G.12 below (which bounds the max by a high moment),

$$\max_{\alpha\in n}\frac{1}{n}\check{h}_\alpha^\top\check{h}_\alpha \xrightarrow{\text{a.s.}} 0,$$

and consequently

$$\max_{\alpha\in[n]}|\lambda_{\min}(\check{\Lambda}_{-\alpha}) - \lambda_{\min}(\check{\Lambda})| \xrightarrow{\text{a.s.}} 0.$$

Because $\check{\Lambda} \xrightarrow{\text{a.s.}} \mathring{\check{\Lambda}}$, we know that, a.s. for large enough n, $\lambda_{\min}(\check{\Lambda})$ is bounded away from 0 by a constant (independent of n). Altogether, this implies that all $\check{\Lambda}_{-\alpha}$ are nonsingular, as desired. □

Fact G.11 (Sherman-Morrison formula)**.** *For any nonsingular matrix $A \in \mathbb{R}^{l\times l}$ and vector $a \in \mathbb{R}^l$, we have*

$$a^\top (A + aa^\top)^{-1} a = \frac{a^\top A^{-1} a}{1 + a^\top A^{-1} a}.$$

Consequently, for any full rank matrix H, the αth diagonal entry of its associated projection matrix $\Pi_H = H(H^\top H)^{-1} H^\top$ can be written as

$$(\Pi_H)_{\alpha\alpha} = \frac{H_\alpha (H_{-\alpha}^\top H_{-\alpha})^{-1} H_\alpha^\top}{1 + H_\alpha (H_{-\alpha}^\top H_{-\alpha})^{-1} H_\alpha^\top}$$

where H_α is the αth row of H, and $H_{-\alpha}$ is H with the αth row removed.

Lemma G.12. *Assume Moments$(m-1)$. Suppose $\psi : \mathbb{R}^{m-1} \to \mathbb{R}$ is polynomially bounded. Then as $n \to \infty$,*

$$\frac{1}{n^p} \max_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})| \xrightarrow{\text{a.s.}} 0$$

for any $p > 0$.

Proof. For any $q > 0$, we have the elementary bound

$$\max_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})| \le \sqrt[q]{\sum_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})|^q}.$$

Thus, for any $q > 0$,

$$\frac{1}{n^p} \max_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})| \le \frac{1}{n^{p-1/q}} \sqrt[q]{\frac{1}{n}\sum_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})|^q}.$$

Because, by Moments$(m-1)$, $\frac{1}{n}\sum_{\alpha\in[n]} |\psi(g_\alpha^1, \dots, g_\alpha^{m-1})|^q \xrightarrow{\text{a.s.}} C$ for some constant C as $n \to \infty$, the RHS above converges a.s. to 0 as soon as we take $q > 1/p$, and therefore so does the LHS.

□

G.5 Inductive Step: Moments(m)

In this section, we show

$$\text{Moments}(m-1) \text{ and } \text{CoreSet}(m-1) \implies \text{Moments}(m).$$

More specifically, we will show that for any polynomially-bounded $\psi : \mathbb{R}^m \to \mathbb{R}$,

$$\frac{1}{n}\sum_{\alpha=1}^n \psi(g_\alpha^1, \dots, g_\alpha^m) \xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)} \psi(Z)$$

where again on the RHS ψ ignores all coordinates $Z^{m+1}, \dots, Z^M$ (corresponding to $g^{m+1}, \dots, g^M$).

By Lemma G.9, if $\mathring{\sigma} = 0$, then almost surely, for large enough n, $g = g^m$ is just a (fixed) linear combination of $g^1, \dots, g^{m-1}$, so Moments is trivially true. Therefore, in the below, we assume

$$\mathring{\sigma} > 0. \tag{$\star$}$$

This assumption will be crucial for our arguments involving smoothness induced by Gaussian averaging.

To clarify notation in the following, we will write $\mathbb{E}_X[expression]$ to denote the expectation over only the randomization in X, and $\mathbb{E}[expression|\,\mathcal{B}]$ to denote the expectation taken over all randomness except those in $\mathcal{B}$.

Proof Plan Note that

$$\left|\frac{1}{n}\sum_{\alpha=1}^{n}\psi(g_\alpha^1,\dots,g_\alpha^m) - \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)}\psi(Z)\right| \le \mathsf{A}+\mathsf{B}+\mathsf{C}$$

where

$$\mathsf{A} \stackrel{\text{def}}{=} \left|\frac{1}{n}\sum_{\alpha=1}^{n}\psi(g_\alpha^1,\dots,g_\alpha^m) - \mathop{\mathbb{E}}_{z}\psi\left(g_\alpha^1,\dots,g_\alpha^{m-1},\omega_\alpha+\sigma z\sqrt{(\Pi_{\hat{H}}^{\perp})_{\alpha\alpha}}\right)\right|$$

$$\mathsf{B} \stackrel{\text{def}}{=} \left|\frac{1}{n}\sum_{\alpha=1}^{n}\mathop{\mathbb{E}}_{z}\psi\left(g_\alpha^1,\dots,g_\alpha^{m-1},\omega_\alpha+\sigma z\sqrt{(\Pi_{\hat{H}}^{\perp})_{\alpha\alpha}}\right) - \mathop{\mathbb{E}}_{z}\psi\left(g_\alpha^1,\dots,g_\alpha^{m-1},\sum_{i=1}^{r}\mathring{v}_i\hat{g}_\alpha^i+\mathring{\sigma}z\right)\right|$$

$$\mathsf{C} \stackrel{\text{def}}{=} \left|\frac{1}{n}\sum_{\alpha=1}^{n}\mathop{\mathbb{E}}_{z}\psi\left(g_\alpha^1,\dots,g_\alpha^{m-1},\sum_{i=1}^{r}\mathring{v}_i\hat{g}_\alpha^i+\mathring{\sigma}z\right) - \mathop{\mathbb{E}}_{Z\sim\mathcal{N}(\mu,\Sigma)}\psi(Z)\right|$$

with $z\sim\mathcal{N}(0,1)$. Note that B and C are random variables in $\mathcal{B}$. We will show that each of $\mathsf{A},\mathsf{B},\mathsf{C}$ goes to 0 almost surely, which would finish the proof of Theorem A.6.

Roughly speaking, $\mathsf{A}\xrightarrow{\text{a.s.}}0$ because of a law of large number, $\mathsf{B}\xrightarrow{\text{a.s.}}0$ because of the smoothness in $\mathbb{E}_z\,\psi$ induced by Gaussian averaging, and $\mathsf{C}\xrightarrow{\text{a.s.}}0$ by induction hypothesis. We start with the last item, since it's the easiest.

G.5.1 C Converges Almost Surely to 0

In this section we show that $\mathsf{C}\xrightarrow{\text{a.s.}}0$ by a straightforward reduction to the inductive hypothesis.

Let $\hat{Z}^1,\dots,\hat{Z}^r$ be the components of $Z\sim\mathcal{N}(\mu,\Sigma)$ corresponding to $\hat{g}^1,\dots,\hat{g}^r$, and let $\hat{Z}$ be the column vector with these entries. Note that, by Proposition F.3, Z^m (corresponding to g^m), conditioned on $Z^1,\dots,Z^{m-1}$, is distributed as a Gaussian with mean $\Sigma(g,\hat{G})\Sigma(\hat{G},\hat{G})^+\hat{Z} = \mathring{\eta}^\top\mathring{\Lambda}^+\hat{Z} = \mathring{v}^\top\hat{Z}$ and variance $\Sigma(g,g)-\Sigma(g,\hat{G})\Sigma(\hat{G},\hat{G})^+\Sigma(\hat{G},g) = \mathring{\sigma}$. Thus

$$\begin{aligned}\mathop{\mathbb{E}}_{Z}\psi(Z) &= \mathop{\mathbb{E}}_{Z^1,\dots,Z^{m-1}}\mathbb{E}[\psi(Z)|Z^1,\dots,Z^{m-1}]\\ &= \mathop{\mathbb{E}}_{Z^1,\dots,Z^{m-1}}\mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)}\psi(Z^1,\dots,Z^{m-1},\mathring{v}^\top\hat{Z}+\mathring{\sigma}z)\\ &= \mathop{\mathbb{E}}_{Z^1,\dots,Z^{m-1}}\Psi(Z^1,\dots,Z^{m-1})\end{aligned}$$

where we have set $\Psi(Z^1,\dots,Z^{m-1})\stackrel{\text{def}}{=}\mathbb{E}_{z\sim\mathcal{N}(0,1)}\psi(Z^1,\dots,Z^{m-1},\mathring{v}^\top\hat{Z}+\mathring{\sigma}z)$. Ψ is a polynomially bounded function since ψ is. Applying the induction hypothesis to Ψ, we obtain

$$\begin{aligned}&\frac{1}{n}\sum_{\alpha=1}^{n}\mathop{\mathbb{E}}_{z}\psi\left(g_\alpha^1,\dots,g_\alpha^{m-1},\sum_{i=1}^{r}\mathring{v}_i\hat{g}_\alpha^i+\mathring{\sigma}z\right)\\ &= \frac{1}{n}\sum_{\alpha=1}^{n}\Psi\left(g_\alpha^1,\dots,g_\alpha^{m-1}\right)\\ &\xrightarrow{\text{a.s.}} \mathop{\mathbb{E}}_{Z^1,\dots,Z^{m-1}}\Psi(Z^1,\dots,Z^{m-1})\\ &\qquad\qquad\text{by induction hypothesis}\\ &= \mathop{\mathbb{E}}_{Z^1,\dots,Z^{m-1}}\mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)}\psi(Z^1,\dots,Z^{m-1},\mathring{v}^\top\hat{Z}+\mathring{\sigma}z)\\ &= \mathop{\mathbb{E}}_{Z}\psi(Z)\end{aligned}$$

as desired.

G.5.2 A Converges Almost Surely to 0

In this section we show $\mathsf{A}\xrightarrow{\text{a.s.}}0$.

For each $\alpha \in [n]$, let $\psi_\alpha(x) \stackrel{\text{def}}{=} \psi(g_\alpha^1, \dots, g_\alpha^{m-1}, \omega_\alpha + \sigma x)$, with ω and σ defined in Eq. (42). This is a random function depending on the randomness of $g_\alpha^1, \dots, g_\alpha^{m-1}$, and it changes with n as well. Also consider the "centered version" of ψ_α, $\tilde{\psi}_\alpha(x) \stackrel{\text{def}}{=} \psi_\alpha(x) - \mathbb{E}\,\psi_\alpha(x')$ with expectation taken over $x' \sim \mathcal{N}(0, (\Pi_{\tilde{H}}^{\perp})_{\alpha\alpha})$ (but not $g_\alpha^1, \dots, g_\alpha^{m-1}$). Note by Eq. (43),

$$\mathsf{A} \stackrel{\mathrm{d}}{=}_{\mathcal{B}} \frac{1}{n} \sum_{\alpha=1}^{n} \tilde{\psi}_\alpha(\xi_\alpha)$$

where $\xi \sim \mathcal{N}(0, \Pi_{\tilde{H}}^{\perp})$.

Proof idea. To prove our claim, we will show that, for almost all (i.e. probability 1 in $\mathcal{U}$) sequences of $(g^1, \dots, g^{m-1}) = (g^1(n), \dots, g^{m-1}(n))$ in n — which we shall call *amenable sequences of* $g^1, \dots, g^{m-1}$ — we have a moment bound

$$\mathbb{E}[\mathsf{A}^{2\lambda}|\mathcal{B}] = \mathop{\mathbb{E}}_{\xi \sim \mathcal{N}(0, \Pi_{\tilde{H}}^{\perp})} \left(\frac{1}{n} \sum_{\alpha=1}^{n} \tilde{\psi}_\alpha(\xi_\alpha) \right)^{2\lambda} < C n^{-1.25} \tag{46}$$

for some large λ and some constant $C > 0$ depending only on λ and the particular sequence of $\{(g^1(n), \dots, g^{m-1}(n))\}_n$. Then we apply Lemma F.2 to show that, conditioned on any amenable sequence, A converges to 0 almost surely over all randomness remaining after conditioning. Since almost all sequences are amenable, this shows that the convergence is also almost sure without the conditioning.

The moment bound. For $\lambda \geq 6$ and any $q > 1$, we first apply Theorem F.8 to get the bound

$$\mathop{\mathbb{E}}_{\xi} \left(\frac{1}{n} \sum_{\alpha=1}^{n} \tilde{\psi}_\alpha(\xi_\alpha) \right)^{2\lambda} \leq c n^{-1.5+1/q} \sqrt[q]{\frac{1}{n} \sum_{\alpha=1}^{n} \mathbb{E}\, \tilde{\psi}_\alpha(\xi_\alpha)^{2\lambda q}}$$

where on both sides $\xi \sim \mathcal{N}(0, \Pi_{\tilde{H}}^{\perp})$, and c is a constant depending only on λ and m, but not on n, the functions ψ_α, or $g^1, \dots, g^{m-1}$. To obtain Eq. (46), we will show that

$$\frac{1}{n} \sum_{\alpha=1}^{n} \mathbb{E}\, \tilde{\psi}_\alpha(\xi_\alpha)^{2\lambda q} \tag{47}$$

is uniformly bounded (in n), almost surely over the randomness of the sequences $\{g^1(n), \dots, g^{m-1}(n)\}_n$. We take all such sequences to be the *amenable sequences*. For $q > 4$, we then get the desired moment bound Eq. (46).

It remains to show the almost sure uniform boundedness.

Almost sure uniform boundedness. Intuitively, Eq. (47) should converge almost surely to a deterministic value by applying some version of the induction hypothesis, so it should be almost surely uniformly bounded in n. The obstacle is that ξ_α is not purely a function of $g_\alpha^1, \dots, g_\alpha^{m-1}$, and *a priori* it is not clear how to apply the induction hypothesis in a straightforward way. We thus first process Eq. (47) a bit. Let $\mu_\alpha \stackrel{\text{def}}{=} \mathbb{E}_{x \sim \mathcal{N}(0, (\Pi_{\tilde{H}}^{\perp})_{\alpha\alpha})}\, \psi_\alpha(x)$. Then, abbreviating $\mathbb{E}$ for expectation taken

over $\xi \sim \mathcal{N}(0, \Pi_{\mathring{H}}^{\perp})$, we have the following inequalities of random variables in $\mathcal{B}$:

$$\begin{aligned}
\frac{1}{n}\sum_{\alpha=1}^{n} \mathbb{E}\,\tilde{\psi}_\alpha(\xi_\alpha)^{2\lambda q} &= \frac{1}{n}\sum_{\alpha=1}^{n} \mathbb{E}(\psi_\alpha(\xi_\alpha) - \mu_\alpha)^{2\lambda q} \\
&\le \frac{1}{n} 2^{2\lambda q - 1} \sum_{\alpha=1}^{n} \mathbb{E}\left[\psi_\alpha(\xi_\alpha)^{2\lambda q} + \mu_\alpha^{2\lambda q}\right] \\
&\qquad\qquad \text{by Lemma F.1} \\
&\le \frac{1}{n} 2^{2\lambda q} \sum_{\alpha=1}^{n} \mathbb{E}\,\psi_\alpha(\xi_\alpha)^{2\lambda q} \\
&\text{by power mean inequality } \mu_\alpha \le \sqrt[2\lambda q]{\mathbb{E}\,\psi_\alpha(\xi_\alpha)^{2\lambda q}} \\
&= \frac{1}{n} 2^{2\lambda q} \sum_{\alpha=1}^{n} \mathbb{E}\,\psi(g_\alpha^1, \ldots, g_\alpha^{m-1}, \omega_\alpha + \sigma\xi_\alpha)^{2\lambda q}.
\end{aligned}$$

Suppose, WLOG, that ψ is polynomially bounded by an inequality $|\psi(x)| \le C\|x\|_p^p + c$ for some $p, C, c > 0$. In the below, we will silently introduce constants $C_1, C_2, \ldots$ via Lemma F.1 and merge with old constants, such that they will only depend on λ, p, q. Continuing the chain of inequalities above

$$\begin{aligned}
\frac{1}{n}\sum_{\alpha=1}^{n} \mathbb{E}\,\tilde{\psi}_\alpha(\xi_\alpha)^{2\lambda q} &\le c + \frac{1}{n} C 2^{2\lambda q} \sum_{\alpha=1}^{n} \mathbb{E}\left(|g_\alpha^1|^p + \ldots + |g_\alpha^{m-1}|^p + |\omega_\alpha + \sigma\xi_\alpha|^p\right)^{2\lambda q} \\
&\le c + \frac{1}{n} C_1 \sum_{\alpha=1}^{n} \mathbb{E}\left(|g_\alpha^1|^p + \ldots + |g_\alpha^{m-1}|^p + |\omega_\alpha|^p + |\sigma\xi_\alpha|^p\right)^{2\lambda q} \\
&\le c + \frac{1}{n} C_2 \sum_{\alpha=1}^{n} |g_\alpha^1|^{2\lambda qp} + \ldots + |g_\alpha^{m-1}|^{2\lambda qp} + |\omega_\alpha|^{2\lambda qp} + \mathbb{E}\,|\sigma\xi_\alpha|^{2\lambda qp}.
\end{aligned} \tag{48}$$

We now proceed to show that the summands of Eq. (48) are almost surely uniformly bounded, which finishes our proof of $\mathsf{A} \xrightarrow{\text{a.s.}} 0$.

- By induction hypothesis,
$$\frac{1}{n}\sum_{\alpha=1}^{n} \mathbb{E}\,|g_\alpha^1|^{2\lambda qp} + \ldots + |g_\alpha^{m-1}|^{2\lambda qp}$$
almost surely converges to a deterministic value, so that it is almost surely uniformly bounded in n.
- In addition, $\sigma \xrightarrow{\text{a.s.}} \mathring{\sigma}$, so that, almost surely, for large enough n, $\sigma \le \mathring{\sigma} + 1$. (The order of the qualifiers is important here; in general this statement cannot be made uniformly in n). Therefore, *almost surely, for large enough* n,
$$\frac{1}{n}\sum_{\alpha=1}^{n} \mathbb{E}\,|\sigma\xi_\alpha|^{2\lambda qp} \le \frac{1}{n}\sum_{\alpha=1}^{n} |\mathring{\sigma} + 1|^{2\lambda qp}\,\mathbb{E}\,|\xi_\alpha|^{2\lambda qp}.$$
This is almost surely uniformly bounded in n because $\mathrm{Var}(\xi_\alpha) = (\Pi_{\mathring{H}}^{\perp})_{\alpha\alpha} \in [0, 1]$ for all α by Lemma F.7.
- It remains to bound $\frac{1}{n}\sum_{\alpha=1}^{n} |\omega_\alpha|^{2\lambda qp}$. We extract our reasoning here into the Lemma G.13 below, as we will need to reuse this for later. This finishes the proof of $\mathsf{A} \xrightarrow{\text{a.s.}} 0$.

Lemma G.13. *For any polynomially bounded function $\varphi : \mathbb{R} \to \mathbb{R}$,*

$$\frac{1}{n}\sum_{\alpha=1}^{n} |\varphi(\omega_\alpha)|$$

is almost surely uniformly bounded in n.

Proof. It suffices to prove this for $\varphi(x) = |x|^d$ for any $d > 0$.

Expanding ω according to Lemma G.5, we get

$$\frac{1}{n}\sum_{\alpha=1}^{n}|\omega_\alpha|^d = \frac{1}{n}\sum_{\alpha=1}^{n}\left|\sum_{i=1}^{r}\hat{g}_\alpha^i(\mathring{v}_i+\hat{\epsilon}_i)+\sum_{j=1}^{s}\check{h}_\alpha^j\check{\epsilon}_j\right|^d$$

for (fixed dimensional) $\hat{\epsilon}\in\mathbb{R}^r, \check{\epsilon}\in\mathbb{R}^s$ that go to 0 almost surely with n. Applying Lemma F.1, we get

$$\frac{1}{n}\sum_{\alpha=1}^{n}|\omega_\alpha|^d \le \frac{1}{n}C_3\sum_{\alpha=1}^{n}\left|\sum_{i=1}^{r}\hat{g}_\alpha^i\mathring{v}_i\right|^d + \left|\sum_{i=1}^{r}\hat{g}_\alpha^i\hat{\epsilon}_i\right|^d + \left|\sum_{j=1}^{s}\check{h}_\alpha^j\check{\epsilon}_j\right|^d.$$

We bound each summand separately.

- By induction hypothesis,

  $$\frac{1}{n}\sum_{\alpha=1}^{n}\left|\sum_{i=1}^{r}\hat{g}_\alpha^i\mathring{v}_i\right|^d$$

  converges a.s. to a deterministic value, so it is a.s uniformly bounded in n.

- By the a.s. decaying property of $\hat{\epsilon}$, we have almost surely, for large enough n, $|\sum_{i=1}^{r}\hat{g}_\alpha^i\hat{\epsilon}_i| \le \sum_{i=1}^{r}|\hat{g}_\alpha^i|$ (again, the order of qualifier is very important here). By induction hypothesis,

  $$\frac{1}{n}\sum_{\alpha=1}^{n}\left(\sum_{i=1}^{r}|\hat{g}_\alpha^i|\right)^d$$

  converges a.s. to a deterministic value, yielding the a.s. uniformly-boundedness of it and of

  $$\frac{1}{n}\sum_{\alpha=1}^{n}\left|\sum_{i=1}^{r}\hat{g}_\alpha^i\hat{\epsilon}_i\right|^d.$$

- Likewise, because for each j, $\check{h}^j$ is a polynomially-bounded function of $g^1,\dots,g^{m-1}$ [19], the summands of

  $$\frac{1}{n}\sum_{\alpha=1}^{n}\left(\sum_{j=1}^{s}|\check{h}_\alpha^j|\right)^d$$

  are polynomially-bounded functions of $g^1,\dots,g^{m-1}$ too. So by induction hypothesis, this sum converges a.s., implying the a.s. uniform boundedness of it and

  $$\frac{1}{n}\sum_{\alpha=1}^{n}\left|\sum_{j=1}^{s}\check{h}_\alpha^j\check{\epsilon}_j\right|^d.$$

□

G.5.3 B Converges Almost Surely to 0

In this section we show $\mathsf{B}\xrightarrow{\text{a.s.}}0$.

[19] This is the only place where we need the assumption that all nonlinearities in the program are polynomially bounded. Otherwise, the compositions of such nonlinearities might not be integrable against the Gaussian measure

Some Notations For brevity, we will set $d_\alpha \stackrel{\text{def}}{=} (\Pi^{\perp}_{\check{H}})_{\alpha\alpha}$. In addition, for each $\alpha \in [n]$, $w \in \mathbb{R}$, $\tau \geq 0$, let

$$\Psi_\alpha(w;\tau^2) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)} \psi\left(g^1_\alpha, \ldots, g^{m-1}_\alpha, w + \tau z\right).$$

(Here and in all that follows, τ^2 is the square of τ, and the 2 is not an index). This is a random function, with randomness induced by $g^1, \ldots, g^{m-1}$.

Our proof idea is to write

$$\begin{aligned}
\mathsf{B} &= \left|\frac{1}{n}\sum_{\alpha=1}^{n} \Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right) - \Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right| \\
&\leq \frac{1}{n}\sum_{\alpha\in U} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right)\right| + \left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right| \qquad (49) \\
&\quad + \frac{1}{n}\sum_{\alpha\in V} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right) - \Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right| \qquad (50)
\end{aligned}$$

where $U \sqcup V = [n]$ is a partition of $[n]$ with $U \stackrel{\text{def}}{=} \{\alpha : d_\alpha < 1/2\}$ and V is its complement. Note that $|U| \leq 2 \operatorname{rank} \check{H} \leq 2s$ is uniformly bounded in n. We then show each summand of Eq. (49) goes to 0 a.s. independently. Finally we use the smoothness of Ψ_α (Eq. (52)) induced by the Gaussian averaging in Ψ_α to show each summand of Eq. (50) is almost surely $o(1/n)$, finishing the proof.

Eq. (49) converges to 0 a.s. We first look at the term

$$\begin{aligned}
\frac{1}{n}\sum_{\alpha\in U} \left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right| &\leq \frac{|U|}{n} \max_{\alpha\in[n]} \left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right| \\
&\leq \frac{2s}{n^{1-1/q}} \sqrt[q]{\frac{1}{n}\sum_{\alpha\in[n]} \left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right|^q} \qquad (51)
\end{aligned}$$

for any $q > 0$. Now $\left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right|^q$ is a fixed (independent of α) polynomially-bounded function of $g^1_\alpha, \ldots, g^{m-1}_\alpha$, so by induction hypothesis,

$$\frac{1}{n}\sum_{\alpha\in[n]} \left|\Psi_\alpha\left(\sum_{i=1}^{r} \mathring{v}_i \hat{g}^i_\alpha; \mathring{\sigma}^2\right)\right|^q$$

is a.s. uniformly bounded in n, so that using a large $q \geq 2$, we see Eq. (51) converges a.s. to 0.

Next, we apply a similar reasoning to the other term and obtain

$$\frac{1}{n}\sum_{\alpha\in U} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right)\right| \leq \frac{2s}{n^{1-1/q}} \sqrt[q]{\frac{1}{n}\sum_{\alpha\in[n]} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right)\right|^q}$$

We in fact already know that

$$\frac{1}{n}\sum_{\alpha\in[n]} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right)\right|^q$$

is a.s. uniformly bounded in n from Eq. (48) in Appendix G.5.2, so that

$$\frac{1}{n}\sum_{\alpha\in U} \left|\Psi_\alpha\left(\omega_\alpha; \sigma^2 d_\alpha\right)\right| \xrightarrow{\text{a.s.}} 0$$

from which follows the same for Eq. (49).

Eq. (50) converges to 0 a.s. As mentioned above, to prove this we will use the following smoothness bound of Ψ_α, whose proof will be delayed to the end of the section. Suppose, WLOG, that the polynomially boundedness of ψ presents itself in an inequality $|\psi(x)| \le C\|x\|_p^p + C$, for some $p, C > 0$, where p is an integer. This p will appear explicitly in this smoothness bound below.

Lemma G.14 (Smoothness of Ψ_α)**.** *Let $w, \Delta w \in \mathbb{R}, \tau^2, \Delta\tau^2 \in \mathbb{R}^{\ge 0}$. Then*

$$\begin{aligned}&\left|\Psi_\alpha(w+\Delta w;\tau^2+\Delta\tau^2) - \Psi_\alpha(w;\tau^2)\right| \\ &\qquad \le R(|\Delta w| + \Delta\tau^2)(1+\tau^{-2})\left(S_\alpha + |w|^p + |\Delta w|^p + \tau^p + (\Delta\tau^2)^{p/2}\right) \end{aligned} \tag{52}$$

for some constant $R > 0$, and where

$$S_\alpha \overset{\text{def}}{=} 1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p.$$

To bound Eq. (50), first we expand

$$\omega_\alpha = \sum_{i=1}^r \hat{g}_\alpha^i(\mathring{v}_i + \hat{\epsilon}_i) + \sum_{j=1}^s \check{h}_\alpha^j \check{\epsilon}_j$$

where, by Lemma G.5, $\hat{\epsilon} \in \mathbb{R}^r, \check{\epsilon} \in \mathbb{R}^s$ are vectors that go to 0 almost surely with n. Then we apply the smoothness bound Eq. (52) to get, for each $\alpha \in V$

$$\begin{aligned}\left|\Psi_\alpha\left(\omega_\alpha;\sigma^2 d_\alpha\right) - \Psi_\alpha\left(\sum_{i=1}^r \mathring{v}_i\hat{g}_\alpha^i;\mathring{\sigma}^2\right)\right| &\le R\left(1+\min(\sigma^2 d_\alpha, \mathring{\sigma}^2)^{-1}\right) X_\alpha Y_\alpha \\ &\le R\left(1+\min(\sigma^2/2, \mathring{\sigma}^2)^{-1}\right) X_\alpha Y_\alpha\end{aligned}$$

using the fact that $d_\alpha \ge 1/2, \forall \alpha \in V$. Here

$$\begin{aligned}X_\alpha &\overset{\text{def}}{=} |\omega_\alpha - \sum_{i=1}^r \mathring{v}_i\hat{g}_\alpha^i| + |\sigma^2 d_\alpha - \mathring{\sigma}^2| \\ &= \left|\sum_{i=1}^r \hat{g}_\alpha^i\hat{\epsilon}_i + \sum_{j=1}^s \check{h}_\alpha^j\check{\epsilon}_j\right| + |\sigma^2 d_\alpha - \mathring{\sigma}^2| \\ Y_\alpha &\overset{\text{def}}{=} S_\alpha + |\omega_\alpha|^p + \left|\sum_{i=1}^r \hat{g}_\alpha^i\hat{\epsilon}_i + \sum_{j=1}^s \check{h}_\alpha^j\check{\epsilon}_j\right|^p + \max(\sigma^2 d_\alpha, \mathring{\sigma}^2)^{p/2} + |\sigma^2 d_\alpha - \mathring{\sigma}^2|^{p/2}.\end{aligned}$$

Thus,

$$\begin{aligned}\textit{Eq. (50)} &= \frac{1}{n}\sum_{\alpha\in V}\left|\Psi_\alpha\left(\omega_\alpha;\sigma^2 d_\alpha\right) - \Psi_\alpha\left(\sum_{i=1}^r \mathring{v}_i\hat{g}_\alpha^i;\mathring{\sigma}^2\right)\right| \\ &\le R\frac{1}{n}\left(1+\min(\sigma^2/2, \mathring{\sigma}^2)^{-1}\right)\sum_{\alpha\in V} X_\alpha Y_\alpha \\ &\le R\left(1+\min(\sigma^2/2, \mathring{\sigma}^2)^{-1}\right)\sqrt{\frac{1}{n}\sum_{\alpha\in V} X_\alpha^2}\sqrt{\frac{1}{n}\sum_{\alpha\in V} Y_\alpha^2}.\end{aligned}$$

Since $\sigma \xrightarrow{\text{a.s.}} \mathring{\sigma}$ and we have assumed $\mathring{\sigma} > 0$ by Eq. ($\star$), we have $\left(1+\min(\sigma^2/2, \mathring{\sigma}^2)^{-1}\right)$ is almost surely uniformly bounded in n.

Thus, Eq. (50) can be shown to converge a.s. to 0 if we show

$$\begin{aligned}&\sqrt{\frac{1}{n}\sum_{\alpha\in V} Y_\alpha^2} \quad \text{is a.s. uniformly bounded in } n, \text{ and} \\ &\sqrt{\frac{1}{n}\sum_{\alpha\in V} X_\alpha^2} \xrightarrow{\text{a.s.}} 0\end{aligned}$$

We prove these two claims in Lemmas G.15 and G.16 below, which would finish our proof of $\mathsf{B} \xrightarrow{\text{a.s.}} 0$, and of our main theorem Theorem A.6 as well.

Lemma G.15. $\sqrt{\frac{1}{n}\sum_{\alpha\in V} X_\alpha^2} \xrightarrow{\text{a.s.}} 0.$

Proof. Note that

$$\begin{aligned} X_\alpha &\le \left|\sum_{i=1}^r \hat{g}_\alpha^i \hat{\epsilon}_i + \sum_{j=1}^s \check{h}_\alpha^j \check{\epsilon}_j\right| + |\mathring{\sigma}^2 - \sigma^2| + |\sigma^2 - \sigma^2 d_\alpha| \\ &\stackrel{\text{def}}{=} P_\alpha + Q_\alpha + R_\alpha. \end{aligned}$$

Then by triangle inequality (in ℓ_2-norm),

$$\sqrt{\frac{1}{n}\sum_{\alpha\in V} X_\alpha^2} \le \sqrt{\frac{1}{n}\sum_{\alpha\in V} P_\alpha^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in V} Q_\alpha^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in V} R_\alpha^2}.$$

We now show that each term above converges a.s. to 0, which would finish the proof of Lemma G.15.

- Because $\hat{\epsilon} \xrightarrow{\text{a.s.}} 0$ and $\check{\epsilon} \xrightarrow{\text{a.s.}} 0$, we have

$$\begin{aligned} \frac{1}{n}\sum_{\alpha\in V} P_\alpha^2 &\le C_8 \frac{1}{n}\sum_{\alpha\in V}\left(\sum_{i=1}^r (\hat{g}_\alpha^i \hat{\epsilon}_i)^2 + \sum_{j=1}^s (\check{h}_\alpha^j \check{\epsilon}_j)^2\right) \\ &\le C_8 \max_{i,j}\{|\hat{\epsilon}_i|, |\check{\epsilon}_j|\} \times \frac{1}{n}\sum_{\alpha\in V}\left(\sum_{i=1}^r (\hat{g}_\alpha^i)^2 + \sum_{j=1}^s (\check{h}_\alpha^j)^2\right) \\ &\le C_8 \max_{i,j}\{|\hat{\epsilon}_i|, |\check{\epsilon}_j|\} \times \frac{1}{n}\sum_{\alpha\in [n]}\left(\sum_{i=1}^r (\hat{g}_\alpha^i)^2 + \sum_{j=1}^s (\check{h}_\alpha^j)^2\right) \\ &\xrightarrow{\text{a.s.}} C_8 \times 0 \times \mathcal{E} = 0 \end{aligned}$$

  where $\mathcal{E}$ is the Gaussian expectation that $\frac{1}{n}\sum_{\alpha\in[n]}\left(\sum_{i=1}^r (\hat{g}_\alpha^i)^2 + \sum_{j=1}^s (\check{h}_\alpha^j)^2\right)$ converges a.s. to, by inductive hypothesis.

- The quantity Q_α actually doesn't depend on α, so that

$$\sqrt{\frac{1}{n}\sum_{\alpha\in V} Q_\alpha^2} \le |\mathring{\sigma}^2 - \sigma^2| \xrightarrow{\text{a.s.}} 0$$

  by Lemma G.4.

- Notice $R_\alpha^2 = \sigma^4(1-d_\alpha)^2 \le \sigma^4(1-d_\alpha)$ because $1 - d_\alpha \in [0, 1/2]$. Thus,

$$\begin{aligned} \frac{1}{n}\sum_{\alpha\in V} R_\alpha^2 &\le \sigma^4 \frac{1}{n}\sum_{\alpha\in V} 1 - d_\alpha \\ &\le \sigma^4 \frac{1}{n}\sum_{\alpha\in [n]} 1 - d_\alpha \\ &= \sigma^4 \frac{1}{n} \operatorname{rank} \check{H} \end{aligned}$$

  by the definition that $d_\alpha = (\Pi_{\check{H}}^\perp)_{\alpha\alpha}$. But of course $\operatorname{rank} \check{H} \le s$ is bounded relative to n. So this quantity goes to 0 (surely) as desired.

□

Lemma G.16. $\sqrt{\frac{1}{n}\sum_{\alpha\in V} Y_\alpha^2}$ *is a.s. uniformly bounded in* n.

Proof. We have

$$\sqrt{\frac{1}{n}\sum_{\alpha\in V} Y_\alpha^2} \le \sqrt{\frac{1}{n}\sum_{\alpha\in V} S_\alpha^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in V} |\omega_\alpha|^{2p}} + \sqrt{\frac{1}{n}\sum_{\alpha\in V} {X'_\alpha}^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in V} \max(\sigma^2 d_\alpha, \mathring{\sigma}^2)^p}$$
$$\le \sqrt{\frac{1}{n}\sum_{\alpha\in [n]} S_\alpha^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in [n]} |\omega_\alpha|^{2p}} + \sqrt{\frac{1}{n}\sum_{\alpha\in [n]} {X'_\alpha}^2} + \sqrt{\frac{1}{n}\sum_{\alpha\in [n]} \max(\sigma^2 d_\alpha, \mathring{\sigma}^2)^p}$$

where

$$X'_\alpha \stackrel{\text{def}}{=} \left|\sum_{i=1}^r \hat{g}_\alpha^i \hat{\epsilon}_i + \sum_{j=1}^s \check{h}_\alpha^j \check{\epsilon}_j\right|^p + |\sigma^2 d_\alpha - \mathring{\sigma}^2|^p$$

We proceed to show that each of 4 summands above are individually a.s. uniformly bounded in n.

- S_α^2 is a polynomially bounded function of $g_\alpha^1, \dots, g_\alpha^{m-1}$, so that by Moments$(m-1)$,
$$\frac{1}{n}\sum_{\alpha\in[n]} S_\alpha^2 \xrightarrow{\text{a.s.}} C$$
for some constant C, so it is also a.s. uniformly bounded in n.
- By Lemma G.13, we get
$$\frac{1}{n}\sum_{\alpha\in[n]} |\omega_\alpha|^{2p}$$
is a.s. uniformly bounded in n.
- Using the same reasoning as in the proof of Lemma G.15, one can easily show
$$\frac{1}{n}\sum_{\alpha\in[n]} {X'_\alpha}^2 \xrightarrow{\text{a.s.}} 0$$
so it is also a.s. uniformly bounded.
- Since $d_\alpha \le 1$, we have $\max(\sigma^2 d_\alpha, \mathring{\sigma}^2) \le \max(\sigma^2, \mathring{\sigma}^2)$, which is independent of α. Therefore,
$$\frac{1}{n}\sum_{\alpha\in[n]} \max(\sigma^2 d_\alpha, \mathring{\sigma}^2)^p \le \frac{1}{n}\sum_{\alpha\in[n]} \max(\sigma^2, \mathring{\sigma}^2)^p$$
$$= \max(\sigma^2, \mathring{\sigma}^2)^{p/2} \xrightarrow{\text{a.s.}} \mathring{\sigma}^p.$$
Therefore, it is also a.s. uniformly bounded in n.

□

Finally, we deliver the promised proof of Lemma G.14.

Proof of Lemma G.14. By Lemma F.4, Ψ_α is differentiable in w, and

$$\partial_w \Psi_\alpha(w;\tau^2) = \tau^{-1} \mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)} z\psi(g_\alpha^1, \dots, g_\alpha^{m-1}, w+\tau z) \tag{53}$$

$$\partial_{\tau^2} \Psi_\alpha(w;\tau^2) = \frac{1}{2}\tau^{-2} \mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)} (z^2-1)\psi(g_\alpha^1, \dots, g_\alpha^{m-1}, w+\tau z). \tag{54}$$

Recall that $|\psi(x)| \le C\|x\|_p^p + C$. We will silently introduce constants $C_1, C_2, \dots$ depending only on p, merging with old constants, typically via Lemma F.1 or by integrating out some integrands

depending only on p. With $z \sim \mathcal{N}(0,1)$,

$$\begin{aligned}
|\partial_w \Psi_\alpha(w;\tau^2)| &\le \tau^{-1} \mathop{\mathbb{E}}_z |z||\psi(g_\alpha^1,\dots,g_\alpha^{m-1}, w+\tau z)| \\
&\le \tau^{-1} C \mathop{\mathbb{E}}_z |z| \left(1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p + |w+\tau z|^p\right) \\
&\le \tau^{-1} C_1 \mathop{\mathbb{E}}_z |z| \left(1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p + |w|^p + \tau^p |z|^p\right) \\
&\le \tau^{-1} C_2 \left(1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p + |w|^p + \tau^p\right).
\end{aligned}$$

Similarly,

$$\begin{aligned}
|\partial_{\tau^2} \Psi_\alpha(w;\tau^2)| &\le \frac{1}{2}\tau^{-2} \mathop{\mathbb{E}}_z |z^2-1||\psi(g_\alpha^1,\dots,g_\alpha^{m-1}, w+\tau z)| \\
&\le \tau^{-2} C_3 \left(1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p + |w|^p + \tau^p\right).
\end{aligned}$$

Therefore, for any $\Delta w \in \mathbb{R}, \Delta\tau^2 \in \mathbb{R}^{\ge 0}$, we have

$$\begin{aligned}
&\left|\Psi_\alpha(w+\Delta w;\tau^2+\Delta\tau^2) - \Psi_\alpha(w;\tau^2)\right| \\
&= \left|\int_0^1 \mathrm{d}t \left(\Delta w \cdot \partial_w \Psi_\alpha(w+\Delta w t;\tau^2+\Delta\tau^2 t) + \Delta\tau^2 \cdot \partial_{\tau^2}\Psi_\alpha(w+\Delta w t;\tau^2+\Delta\tau^2 t)\right)\right| \\
&\le \int_0^1 \mathrm{d}t \left(|\Delta w| \cdot |\partial_w \Psi_\alpha(w+\Delta w t;\tau^2+\Delta\tau^2 t)| + |\Delta\tau^2| \cdot |\partial_{\tau^2}\Psi_\alpha(w+\Delta w t;\tau^2+\Delta\tau^2 t)|\right) \\
&\le (C_2+C_3)(|\Delta w| + |\Delta\tau^2|) \\
&\quad \int_0^1 \mathrm{d}t ((\tau^2+\Delta\tau^2 t)^{-1/2} + (\tau^2+\Delta\tau^2 t)^{-1}) \times \left(S_\alpha + |w+\Delta w t|^p + (\tau^2+\Delta\tau^2 t)^{p/2}\right)
\end{aligned}$$

where for brevity we have set

$$S_\alpha \stackrel{\text{def}}{=} 1 + |g_\alpha^1|^p + \cdots + |g_\alpha^{m-1}|^p,$$

which is independent of t.

Since $\Delta\tau^2 \ge 0$, $(\tau^2+\Delta\tau^2 t)^{-1} \le \tau^{-2}$, and we get

$$\begin{aligned}
&\left|\Psi_\alpha(w+\Delta w;\tau^2+\Delta\tau^2) - \Psi_\alpha(w;\tau^2)\right| \\
&\le C_4(|\Delta w| + \Delta\tau^2)(\tau^{-1}+\tau^{-2}) \int_0^1 \mathrm{d}t \left(S_\alpha + |w+\Delta w t|^p + (\tau^2+\Delta\tau^2 t)^{p/2}\right) \\
&\le C_5(|\Delta w| + \Delta\tau^2)(\tau^{-1}+\tau^{-2}) \int_0^1 \mathrm{d}t \left(S_\alpha + |w|^p + |\Delta w|^p t^p + \tau^p + (\Delta\tau^2)^{p/2} t^{p/2}\right) \\
&\le C_6(|\Delta w| + \Delta\tau^2)(\tau^{-1}+\tau^{-2}) \left(S_\alpha + |w|^p + |\Delta w|^p + \tau^p + (\Delta\tau^2)^{p/2}\right)
\end{aligned}$$

where in the end we have integrated out t^p and $t^{p/2}$. We finally apply the simplification $\tau^{-1} \le \frac{1}{2} + \frac{1}{2}\tau^{-2}$ by AM-GM to get the desired Eq. (52).

□