Estimating the expected output of wide random MLPs more efficiently than sampling

Wilson Wu[*] wilson@alignment.org
Victor Lecomte[*] victor@alignment.org
Michael Winer[*] mike@alignment.org
George Robinson[*] george@alignment.org
Jacob Hilton[†‡] jacob@alignment.org
Paul Christiano[‡] paul@alignment.org

Alignment Research Center

Abstract

By far the most common way to estimate an expected loss in machine learning is to draw samples, compute the loss on each one, and take the empirical average. However, sampling is not necessarily optimal. Given an MLP at initialization, we show how to estimate its expected output over Gaussian inputs without running samples through the network at all. Instead, we produce approximate representations of the distributions of activations at each layer, leveraging tools such as cumulants and Hermite expansions. We show both theoretically and empirically that for sufficiently wide networks, our estimator achieves a target mean squared error using substantially fewer FLOPs than Monte Carlo sampling. We find moreover that our methods perform particularly well at estimating the probabilities of rare events, and additionally demonstrate how they can be used for model training. Together, these findings suggest a path to producing models with a greatly reduced probability of catastrophic tail risks.

1 Introduction

Stochastic gradient descent is based on a simple idea: to estimate the expected loss of a neural network (and hence its gradient), we draw some samples, compute the loss on each one, and average the results. This sample-based approach is so natural that it is rarely questioned, yet it may be far from optimal. By analogy, consider a one-dimensional definite integral: we could approximate it numerically using Monte Carlo sampling over the domain of integration, but when a closed form for the integral exists, evaluating it directly is far more efficient. Similarly, when the input distribution and loss function of a neural network can be specified formally (such as with board games, formal mathematics, or simulated physical environments), we might hope to use analytic methods to estimate the expected loss much more efficiently than with sampling.

This hope may seem far-fetched for neural networks, since their inputs are high-dimensional and their behavior is highly irregular. Yet we show that, at least in some well-behaved cases, it can in fact be realized. Even though computing the expected loss exactly is typically intractable, there can be tractable analytic approximations that significantly outperform Monte Carlo sampling in terms of mean squared error for a given computational budget. In our definite integral analogy, these might correspond to a rapidly convergent series expansion for the integral.

[*]Joint first authors.
[†]Corresponding author.
[‡]Joint senior authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

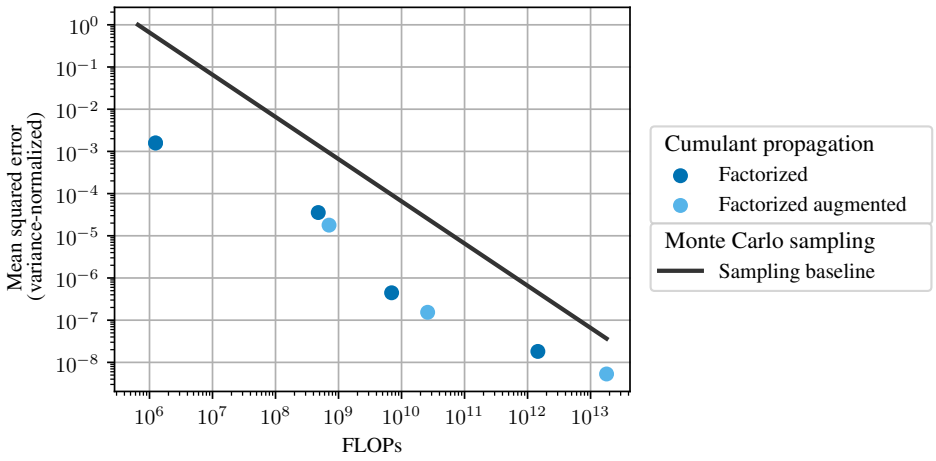


Figure 1: Performance of our best cumulant propagation algorithms at estimating the mean of random ReLU MLPs with 4 hidden layers and width 256. Mean over 5 random seeds shown. (Error bars hidden because they would be too small to be readable, given the logarithmic scale.)

The particular case we study is that of wide random multilayer perceptrons (MLPs): fully-connected feedforward neural networks with random weights, in the large-width limit. Importantly, our algorithms take as input the weights of a *particular* network and estimate the expected output over random inputs, i.e., a *quenched* average, in the language of statistical physics. This is not to be confused with the *annealed* average over the choice of weights, as is commonplace elsewhere in deep learning theory [Roberts, Yaida, and Hanin, 2022]. *After* comparing our estimates with ground truth, we average over the choice of weights, which are drawn independently from $\mathcal{N}\left(0, \frac{c}{n}\right)$ for some constant c, where n is the network width.

Our main result is the design of sample-free estimation procedures that outperform Monte Carlo sampling at this problem, both theoretically and empirically.

Theoretically, we prove that in the average case, for networks with a fixed, constant depth, our best-performing algorithms achieve a mean squared error of $O(\varepsilon^2)$ in time $O(\frac{n}{\varepsilon^2})$.[4] By comparison, Monte Carlo sampling achieves a mean squared error of $\Theta(\varepsilon^2)$ in time $\Theta(\frac{n^2}{\varepsilon^2})$,[5] a factor of n slower. This means that for any given depth, our algorithms are guaranteed to outperform Monte Carlo sampling at sufficiently large width, although the dependence of our algorithms on depth is worse.

Empirically, we validate the performance of our algorithms at realistic widths. Figure 1 shows the mean squared error of our best-performing algorithms versus the number of floating-point operations (FLOPs) they perform, for random ReLU networks with 4 hidden layers and width 256. We are able to outperform Monte Carlo sampling at computational budgets spanning 7 orders of magnitude, in some cases by a factor of over 100.

Furthermore, since our algorithms produce differentiable estimates, they can also be used to train neural networks in certain simple settings. As a proof of concept, we apply our algorithms to distill a random teacher into a student of a different width, a process we refer to as *mechanistic distillation*.

Despite some similarities in performance, our algorithms are very unlike Monte Carlo sampling. Instead of drawing samples, we employ *cumulant propagation* [Christiano et al., 2022, Appendix D]. In cumulant propagation, we form an approximate representation of each activation distribution using its low-order cumulants, which can be thought of as a series expansion about a Gaussian distribution. We start with the input distribution and propagate this forward through the network layer by layer until we reach the output. The smaller the error tolerance ε as a function of the network width n, the higher the order of cumulants we track: roughly speaking, if $\varepsilon(n)^2 \sim \frac{1}{n^K}$, then we track cumulants of order at most K.

[4] Our theoretical results assume polynomial activation functions and conditions on the growth rate of $\varepsilon(n)$.

[5] We assume the use of naive rather than Strassen-type matrix multiplication in all algorithms.

Our motivation for producing analytic loss estimates goes beyond their overall computational efficiency, and is closely related to the problem of *low probability estimation* [Wu and Hilton, 2025]. When we apply our algorithms to estimating the tails of the output distribution, we find that they outperform Monte Carlo sampling to an increasing degree as the true probability decreases. This suggests that using analytic loss estimates to train networks could greatly reduce the probability of catastrophic tail risks.

The remainder of the paper is structured as follows.

- In Section 2, we provide a formal definition of our MLP setup, a precise problem statement, and a description of the Monte Carlo sampling baseline.
- In Section 3, we define the main mathematical constructions used by our algorithms.
- In Section 4, we outline our sample-free estimation procedures, including *augmented* and *factorized* versions. Further explanation and pseudocode for special cases can be found in Appendix B.
- In Section 5, we state our main theoretical results about our algorithms.
- In Section 6, we describe our experimental setup and our main empirical results. Further empirical results can be found in Appendices C, D, E, F, G, H and I.
- In Section 7, we discuss the implications of our results for improving training efficiency for losses dominated by rare events, as well as open problems and conjectures.

We also provide a technical supplement, Appendix S, containing precise descriptions of all of our algorithms and full proofs of all of our theoretical results.

2 Problem statement

2.1 MLP definition

We consider MLPs with equal input dimension, output dimension, and hidden dimensions, and without a final activation function. We do not incorporate bias terms, but we show how to adapt our algorithms to work for these in Appendix F. A formal definition is as follows.

Definition 2.1. The *multilayer perceptron* (MLP) with $L \geq 0$ hidden layers, width $n \geq 1$, activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$ and weights $\boldsymbol{\theta} = \left(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}\right) \in \mathbb{R}^{(L+1)\times n\times n}$ is the function $M_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$M_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{W}^{(L+1)}\phi_L\Big(\mathbf{W}^{(L)}\ldots\phi_2\Big(\mathbf{W}^{(2)}\phi_1\Big(\mathbf{W}^{(1)}\mathbf{x}\Big)\Big)\ldots\Big),$$

where the activation functions are applied coordinatewise.

In our theoretical results, we consider polynomial activation functions, but this is not a practical limitation. In our experiments, our default choice of activation function is the rectified linear unit, $\mathrm{ReLU}\,(z) := \max\,(z, 0)$, and we consider other activation functions in Appendix F.

2.2 Central estimation problem

The central problem we consider is as follows. Given weights $\boldsymbol{\theta} \in \mathbb{R}^{(L+1)\times n\times n}$ drawn independently from $\mathcal{N}\left(0, \frac{2}{n}\right)$ (following He et al. [2015]), estimate

$$\mathbb{E}_{X\sim\mathcal{N}(0,\mathbf{I}_n)}\left[M_{\boldsymbol{\theta}}\left(X\right)\right],$$

where $M_{\boldsymbol{\theta}}$ is the MLP with weights $\boldsymbol{\theta}$. More precisely, an estimation procedure should take as input $\boldsymbol{\theta}$ and $\varepsilon > 0$, and produce an estimate with mean squared error $O(\varepsilon^2)$ over the choice of $\boldsymbol{\theta}$ for each output neuron.

Each entry of the above expectation can be thought of as the expected loss of the network under a simple linear loss function of the network's output. Our methods can easily be adapted to other closed-form loss functions that only depend on the network's output, by treating the loss function as an additional activation function. For more complex formally-defined loss functions that also depend on the network's input, it is often still possible to adapt our methods, although more work may be required. We demonstrate this in the case of a simple distillation loss in Appendix H.

2.3 Monte Carlo sampling baseline

The baseline estimation procedure we consider is Monte Carlo sampling. This procedure samples $X^{(1)}, \ldots, X^{(N)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \mathbf{I}_n)$ for $N = \left\lceil \frac{1}{\varepsilon^2} \right\rceil$ and computes the estimate

$$\frac{1}{N} \sum_{r=1}^{N} M_{\boldsymbol{\theta}}\left(X^{(r)}\right).$$

This achieves the required mean squared error of $\Theta(\varepsilon^2)$, since a single forward pass has variance $\Theta(1)$ under our choice of initialization, and runs in time $\Theta(\frac{n^2}{\varepsilon^2})$ (recall that we treat the number of hidden layers L as a constant in our asymptotics).

3 Mathematical preliminaries

3.1 Cumulants

Our sample-free estimation procedures work by estimating the *cumulants* of the network's activations. The kth-order cumulants of a vector-valued random variable $X \in \mathbb{R}^n$, denoted $\kappa\left[X_{i_1}, \ldots, X_{i_k}\right]$ for $i_1, \ldots, i_k = 1, \ldots, n$, are defined recursively in terms of the moments of X by the partition formula

$$\mathbb{E}[X_{i_1} \cdots X_{i_k}] = \sum_{\substack{\text{partitions} \\ \pi \text{ of } \{1,\ldots,k\}}} \prod_{S \in \pi} \kappa[X_{i_s} : s \in S].$$

For example,

$$\begin{aligned} \mathbb{E}[X_i] &= \kappa[X_i] \qquad \text{and} \\ \mathbb{E}[X_i X_j] &= \kappa[X_i]\,\kappa[X_j] + \kappa[X_i, X_j]. \end{aligned}$$

Thus the first-order cumulants form the mean vector of X, and the second-order cumulants form the covariance matrix of X. The third- and higher-order cumulants of a distribution can be thought of as measuring the deviation of the distribution from Gaussianity, since these cumulants are zero for a Gaussian distribution.

3.2 Hermite expansions

To handle non-linear activation functions, we take *Hermite expansions*. Given $Y \sim \mathcal{N}\left(\mu, \sigma^2\right)$ and $\phi : \mathbb{R} \to \mathbb{R}$ with $\mathbb{E}\left[\phi\left(Y\right)^2\right] < \infty$, the Hermite expansion of ϕ with respect to Y is the series expansion

$$\phi(z) = \sum_{k=0}^{\infty} \frac{1}{k!}\, \widehat{\phi}_k^{(\mu,\sigma^2)}\, \sigma^k\, \mathrm{He}_k\left(\frac{z-\mu}{\sigma}\right)$$

where He_k is the kth probabilist's *Hermite polynomial*, the monic polynomial of degree k that is orthogonal to all polynomials of lower degree under the standard Gaussian measure. The *Hermite coefficients*[6] in this expansion are given by

$$\widehat{\phi}_k^{(\mu,\sigma^2)} := \frac{1}{\sigma^k} \mathbb{E}\left[\phi\left(Y\right) \mathrm{He}_k\left(\frac{Y-\mu}{\sigma}\right)\right]. \tag{1}$$

Truncating the expansion gives the best low-degree approximation to ϕ in mean squared error under Y.

3.3 Diagram summation formula for cumulants

After taking the Hermite expansion of a non-linear activation function $\phi : \mathbb{R} \to \mathbb{R}$, we will need to compute the cumulants of $\phi\left(Z_1\right), \ldots, \phi\left(Z_n\right)$ using the cumulants of $Z_1, \ldots, Z_n$ and the Hermite coefficients of ϕ. To do this, we apply a formula that expresses the desired cumulants as a sum over combinatorial objects we call *diagrams*. We refer to this formula as the *(Hermite-based) diagram summation formula for cumulants*, and we state and explain it in Appendix A.

[6] Other authors use the term *Hermite coefficient* to refer to the same expression divided by $k!$.

3.4 Sizes of cumulants with repeated indices

The typical size of a kth-order cumulant of the network's activations strongly depends on whether a single neuron index appears multiple times. This has an effect on the error of truncating our diagram summation formula for cumulants, which impacts the design of our algorithm.

To illustrate this, we consider the special case of our diagram summation formula in which third- and higher-order cumulants are ignored. This is the covariance propagation formula of Wright et al. [2024, Theorem 1], which we derive as Corollary A.3. This states that when $Z_1, \ldots, Z_n$ are multivariate Gaussian, then

$$\operatorname{Cov}\left[\phi\left(Z_i\right), \phi\left(Z_j\right)\right] = \sum_{k=1}^{\infty} \frac{1}{k!} \widehat{\phi}_k^{(\mathbb{E}[Z_i], \operatorname{Var}[Z_i])} \widehat{\phi}_k^{(\mathbb{E}[Z_j], \operatorname{Var}[Z_j])} \operatorname{Cov}\left[Z_i, Z_j\right]^k .$$

Typically, $|\operatorname{Cov}[Z_i, Z_j]|$ is $O(\frac{1}{\sqrt{n}})$ when $i \neq j$, but $O(1)$ when $i = j$. This means that the error of truncating this formula decays rapidly with n when $i \neq j$, but does not decay with n when $i = j$. This motivates us to change our approach from applying our diagram summation formula naively.

4 Methods

4.1 Basic cumulant propagation algorithm

With the above preliminaries in place, we can now roughly describe our sample-free estimation procedures. We begin with the basic version of our algorithm, before going on to describe variants.

Our algorithm first picks some maximum order K of cumulants to track. In our theoretical results, we choose K to achieve $O\left(\varepsilon^2\right)$ error, and in our experiments, we treat K as a hyperparameter. Given K, we start with the input distribution $\mathcal{N}(0, \mathbf{I}_n)$, which has second-order cumulant matrix $\mathbf{I}_n$ and all other cumulants equal to 0. We then proceed through each operation in the network, updating the estimated cumulants at each step.

- At each matrix multiplication step, we update the estimated cumulants using the multilinearity property of cumulants. This involves taking an einsum of the kth-order estimated cumulant tensor with k copies of the weight matrix, for $k = 1, \ldots, K$.
- At each activation function step, we update the estimated cumulants using a truncation of the diagram summation formula for cumulants. This formula allows us to compute the cumulants of the output in terms of the cumulants of the input and the Hermite coefficients of the activation function.

Once we reach the output of the network, we return the final estimated first-order cumulant vector.

As stated so far, the error of this algorithm is too large, because of the problem with sizes of cumulants with repeated indices described in Section 3.4. To remedy this, we make two further adjustments.

1. **Power cumulants.** Instead of taking Hermite expansions of the activation function only, we take Hermite expansions of *powers* of the activation function, and use these to estimate the *power cumulants* of the activation output. The kth-order power cumulants of X are the cumulants of the form $\kappa\left[X_{i_m}^{\alpha_1}, \ldots, X_{i_m}^{\alpha_m}\right]$ with $\alpha_1, \ldots, \alpha_m \geq 1$, $\alpha_1 + \cdots + \alpha_m = k$ and $i_1, \ldots, i_m$ *distinct*, from which we can obtain the kth-order cumulants of X by applying the partition formula and its inverse version. For example, when $K = 2$, we compute Gaussian variances exactly, while continuing to use an approximation for off-diagonal Gaussian covariances.
2. **Extra full trace for odd K.** When the maximum cumulant order K is odd, we also track a single number corresponding to the *full trace* of the $(K+1)^{\text{th}}$-order cumulant tensor, $\sum_{i_1, \ldots, i_{(K+1)/2}=1}^{n} \kappa\left[X_{i_1}, X_{i_1}, X_{i_2}, X_{i_2}, \ldots, X_{i_{(K+1)/2}}, X_{i_{(K+1)/2}}\right]$. For example, when $K = 1$, we also track the covariance matrix trace $\sum_{i=1}^{n} \operatorname{Var}[X_i]$.

To help explain our sample-free estimation procedures more precisely, we provide pseudocode for the cases $K = 1$ and $K = 2$ in Appendix B. A precise and fully general formulation is significantly more involved, and is given in Appendix S.4. Python implementations are given at `https://github.com/alignment-research-center/mlp_kprop`.

4.2 Augmented algorithm

So far, we have roughly described the basic version of our algorithm. However, we have some flexibility to trade off between mean squared error and runtime by varying exactly which cumulants to track. One possibility (when K is odd, say) is to track not merely the full trace of the $(K+1)^{\text{th}}$-order cumulant tensor, but the entire tensor *except for its traceless component* (a traceless tensor is one for which taking the trace over any pair of indices results in the zero tensor). This improves the mean squared error of the algorithm, but increases its runtime. We refer to this as the *augmented* version of our algorithm, and describe it more precisely in Appendix S.4.2.

The augmented version of our algorithm is not the only possible tradeoff between mean squared error and runtime that can be made. Rather, it serves as an example of how our algorithm can be flexibly adapted based on the computational budget, beyond merely choosing a different value of K.

4.3 Factorized algorithms

Another important change we can make to our algorithm is to modify how the cumulant tensors are represented. For $K \geq 3$, the tensor of Kth-order cumulants admits a factorized representation of size n^{K-1} that can be preserved through each step of the algorithm. By representing this tensor in factorized form, we can improve the runtime of our algorithm by a factor of n, up to constant factors,[7] without changing its final output. We refer to this as the *factorized* version of our algorithm, and describe it more precisely in Appendix S.4.3.

Unlike the augmented version of our algorithm, the factorized version offers a strict performance improvement for sufficiently large n, which allows us to substantially outperform Monte Carlo sampling for large n. We can also apply both modifications at once to obtain a *factorized augmented* version of our algorithm.

5 Theoretical results

Our main theoretical results are as follows.

Theorem 5.1. *Fix an integer $L \geq 0$ and polynomial activation functions $\phi_1, \ldots, \phi_L$. Given weights $\boldsymbol{\theta} \in \mathbb{R}^{(L+1)\times n\times n}$ drawn independently from $\mathcal{N}\left(0, \frac{2}{n}\right)$ for an integer $n \geq 1$, write $M_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ for the MLP with L hidden layers, width n, activation functions $\phi_1, \ldots, \phi_L$ and weights $\boldsymbol{\theta}$. Then there is an estimation procedure $\mathbb{G}(\boldsymbol{\theta}, \varepsilon)$ such that for every noticeable[8] function $0 < \varepsilon(n) \leq 1$ and every output index i, as $n \to \infty$,*

$$\mathbb{E}_{\boldsymbol{\theta}}\left[\left(\mathbb{G}(\boldsymbol{\theta}, \varepsilon(n))_i - \mathbb{E}_{X\sim\mathcal{N}(0,\mathbf{I}_n)}\left[M_{\boldsymbol{\theta}}(X)\right]_i\right)^2\right] = O\Big(\varepsilon(n)^2\Big)$$

and $\mathbb{G}(\boldsymbol{\theta}, \varepsilon)$ runs in time $O\left(\frac{n^2}{\varepsilon^2}\right)$.

Theorem 5.2. *In Theorem 5.1, if we also require that $\varepsilon(n)^2 = O\left(\frac{1}{n^2}\right)$, then $\mathbb{G}(\boldsymbol{\theta}, \varepsilon)$ can be made to run in time $O\left(\frac{n}{\varepsilon^2}\right)$.*

Theorem 5.1 is obtained from the basic version of our sample-free estimation procedure, and Theorem 5.2 from the factorized version. Thus our results show that, up to constant factors, the basic version of our algorithm matches the performance of Monte Carlo sampling in terms of mean squared error for a given computational budget, while the factorized version outperforms Monte Carlo sampling by a factor equal to the network width n, providing ε is not too large.[9]

These results treat the number of hidden layers L as a constant, and choose the maximum cumulant order K based on the error tolerance ε. Expressed in terms of these parameters, we conjecture that our algorithms have variance-normalized mean squared error at most $c_K(L/n)^K$, although the dependence on L is difficult to analyze precisely. The runtime of the basic version of our algorithm is at most $c'_K L n^{K+1}$, and the runtime of the factorized version is at most $c''_K L^2 n^K$ whenever $K \geq 3$. Here, c_K, c'_K and c''_K are constants that depend only on K.

[7] The new constant factors scale quadratically rather than linearly width depth, as can be seen for $K \leq 4$ in Appendix J.

[8] This means that for some $k \geq 0$, $\varepsilon(n) \geq \frac{1}{n^k}$ for all sufficiently large n [Goldreich, 2005, Chapter 1.2].

[9] A mean squared error of $\varepsilon(n)^2 = O(\frac{1}{n^2})$ is achieved with $K \geq 2$, which costs as much as sampling with $\Omega(n)$ samples.

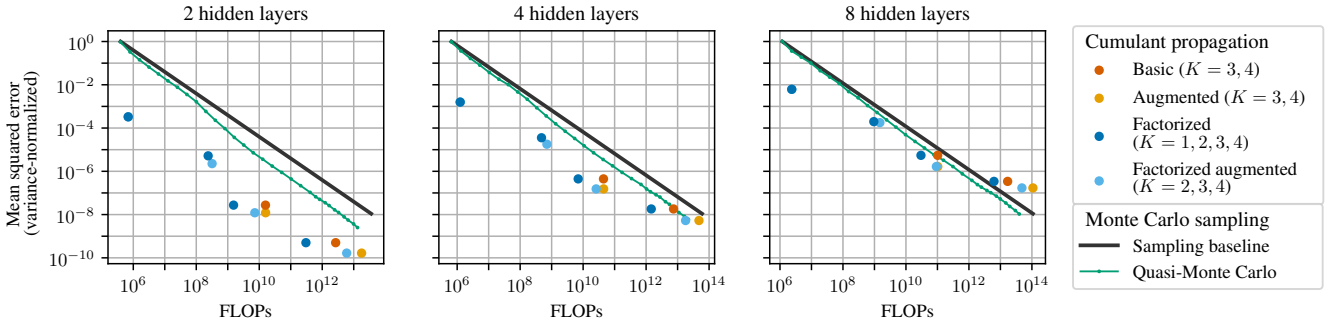


Figure 2: Performance of all versions of our algorithm for different networks of width 256 and K varying from 1 to 4. All 4 versions are identical for $K = 1$, and the factorized versions are identical to their unfactorized counterparts for $K = 2$. (Error bars hidden because they would be too small to be readable.)

We provide full proofs of both results in the technical supplement, Appendix S. We believe that with additional effort, both results can be extended from polynomials to general activation functions.[10]

6 Empirical results

6.1 Experimental setup

We empirically validate the performance of all 4 versions of our algorithm (basic, augmented, factorized and factorized augmented) on ReLU MLPs with a variety of depths and widths. We vary the width n from 4 to 256 and the number of hidden layers L from 2 to 12 (using the same network for each choice of L, but stopping after different numbers of layers). We vary the maximum cumulant order K from 1 to 4. For each configuration, we calculate mean squared error (MSE) over all output neurons and over 5 different random seeds.

To calculate MSE, we estimate ground truth using Monte Carlo sampling with $2^{34} \approx 2 \times 10^{10}$ samples, which greatly exceeds any of our algorithms' computational budgets. We use these same samples to estimate the variance of a single forward pass, and divide MSE by this variance to obtain *variance-normalized MSE*. This choice of units means that Monte Carlo sampling with N samples has a variance-normalized MSE of exactly $\frac{1}{N}$.

We measure the computational budget of our algorithms in floating-point operations (FLOPs), which we count by applying an extension of PyTorch's FLOP counter to our Python implementation. However, since cumulant tensors are symmetric, naively taking cumulant tensor einsums results in redundant computation. Rather than implementing kernels for symmetric tensor einsums, we simply adjust our counts to exclude the redundant FLOPs. We describe this adjustment in more detail in Appendix J.

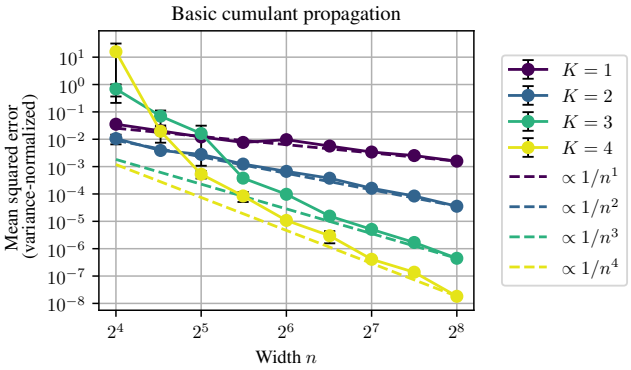
6.2 Comparison with Monte Carlo sampling

We compare the mean squared error of our sample-free estimation procedures to that of Monte Carlo sampling. Our results at width 256 with 2, 4 and 8 hidden layers are shown in Figure 2, and our results at additional widths and depths are given in Appendix C.

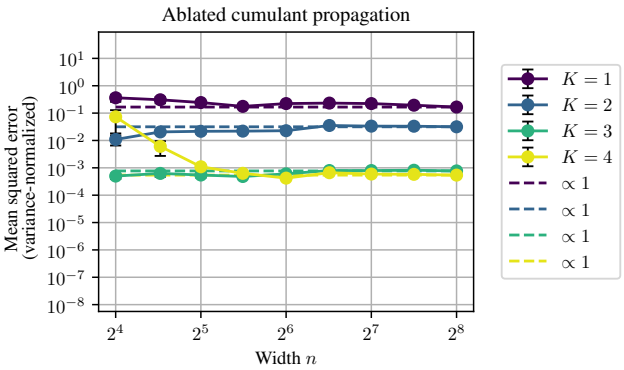
The factorized and factorized augmented versions of our algorithm outperform Monte Carlo sampling by a factor of around 1,000 for 2 hidden layers, and around 10–100 for 4 hidden layers. They start to underperform sampling for 8 hidden layers once the maximum cumulant order K reaches 4, although by our theoretical results, they would outperform sampling once again if we were to increase the width further.

We also tested the performance of Quasi-Monte Carlo sampling based on scrambled Sobol sequences, as implemented in SciPy [Roy et al., 2023]. The results are also shown in Figure 2. Quasi-Monte

[10] We make the mild assumptions that the activation functions are measurable and polynomially bounded.



(a) The mean squared error of the basic (equivalently, factorized) version of our algorithm is $O\left(\frac{1}{n^K}\right)$.



(b) The mean squared error of the power cumulant ablation of our algorithm is $O(1)$ (horizontal).

Figure 3: Width scaling of the basic and ablated versions of our algorithm on networks with 4 hidden layers. Dashed lines are those with the predicted large-n slope passing through the rightmost point. Error bars show ± 1 standard error over 5 random seeds.

Carlo sampling offers a modest benefit of around a factor of 1–10 over simple Monte Carlo sampling, significantly less than the improvement offered by our algorithms.

6.3 Scaling with width and depth

The central finding of our theoretical analysis is that the mean squared error of our algorithm is $O\left(\frac{1}{n^K}\right)$, where n is the width and K is the maximum cumulant order. To validate this finding explicitly, we plot the mean squared error of the basic version of algorithm as a function of the width in Figure 3(a). As expected, the mean squared error closely follows a curve proportional to $\frac{1}{n^K}$ when n is large. Note, however, that increasing K can *worsen* performance at small values of n. We discuss scaling with depth in Appendix D.

6.4 Power cumulant ablation

Our theoretical error analysis was critical to the design of our algorithms. In particular, it led directly to the two adjustments described in Section 4.1: the use of power cumulants, and the inclusion of the full trace of the $(K+1)^{\text{th}}$-order cumulant tensor for odd K. To validate the necessity of these adjustments, we check the performance of our algorithm without them. We plot the mean squared error of this ablated version of our algorithm as a function of the width n in Figure 3(b). As predicted by our theoretical analysis, the mean squared error is $O(1)$ instead of $O\left(\frac{1}{n^K}\right)$, and so the ablated version of our algorithm no longer matches the performance of sampling in the large-width limit. We provide further details of this ablation, including a more direct comparison to Monte Carlo sampling, in Appendix E.

6.5 Other activation functions

Our algorithm extends easily to activation functions other than ReLU. To demonstrate this, we check the performance of our algorithm on GELU [Hendrycks and Gimpel, 2016] (a popular alternative to ReLU) and $\tanh$ (to test a bounded activation function). Our results are given in Appendix F. Results for both activation functions are qualitatively similar to our results for ReLU.

6.6 Low probability estimation

As we discuss later in Section 7.2, we are especially interested in outperforming sampling at estimating low probabilities. To test the performance of our algorithms at this task, we replace the final ReLU activation in our networks with the threshold function $\mathbb{1}_{z>3}$, so that the final expected value can be interpreted as a low probability. We compare the factorized version of our algorithm with maximum

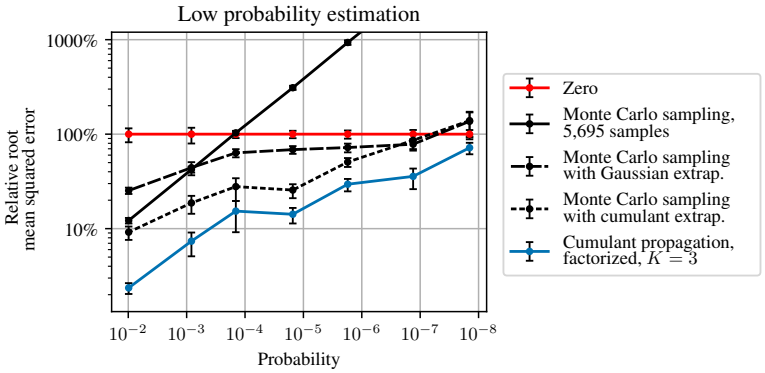


Figure 4: Performance of the factorized version of our algorithm with $K = 3$ at low probability estimation. Error bars show ± 1 standard error of the mean squared error over the random seeds used in each bucket. There were 205 seeds used in total across the 7 buckets.

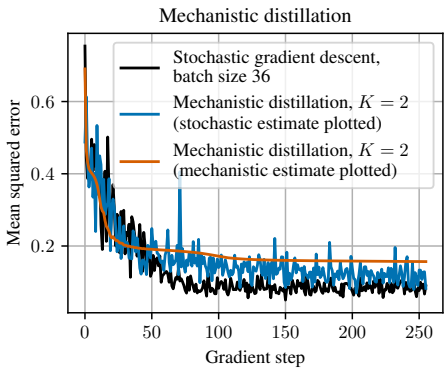


Figure 5: Learning curves for distillation of a random teacher MLP with width 16 into a student MLP with width 8, both with 4 hidden layers.

cumulant order $K = 3$ to Monte Carlo sampling with a comparable number of FLOPs, as well as three other baselines that are explained in Appendix G.

Our results at width 256 with 4 hidden layers (3 ReLUs and one threshold function) are shown in Figure 4. This shows root mean squared error relative to root mean squared probability, with probabilities bucketed into intervals $[10^{k-\frac{1}{2}}, 10^{k+\frac{1}{2}})$ for integers k. Monte Carlo sampling with N samples is worse than simply guessing zero once the probability drops below $1/N$. On the other hand, using a similar number of FLOPs to Monte Carlo sampling with $N = 5,695$ samples, our algorithm achieves a relative error of under 30% for probabilities 100 times lower than $1/N$, and continues to outperform all baselines until the probabilities are nearly 10,000 lower.

6.7 Mechanistic distillation

Since our algorithms produce differentiable estimates, they can be used to train networks by applying gradient descent to the estimate of the expected loss. We demonstrate this approach by using it to distill a random teacher MLP into a student MLP of a different width, a process we refer to as *mechanistic distillation*. We compare this to ordinary distillation using stochastic gradient descent with a comparable number of FLOPs per gradient step.

Our results for a random teacher MLP with width 16 and a student MLP with width 8 are shown in Figure 5. Further details of our experimental setup are given in Appendix H. Mechanistic distillation works well, but is slightly less efficient than distillation using stochastic gradient descent. This may be because our mechanistic estimation algorithms only have good guarantees for random networks, not partially trained ones. We discuss how our algorithms might be adapted to partially-trained networks in Section 7.2.

6.8 Wall-clock time

By default, we measure computational cost in FLOPs, to minimize sensitivity to implementation details. Typically, almost all of the FLOPs both in our algorithms and in Monte Carlo sampling are part of matrix multiplications (or more general einsums), so FLOP counts should be representative of the performance of optimized implementations. However, in practice, our algorithms also include a number of additional FLOPs that, while being a small fraction of the total, take up the majority of the wall-clock time. Although we expect that this issue could be resolved with additional implementation effort, for the interested reader, we show mean squared error against wall-clock time in Appendix I.

7 Discussion

7.1 Outperforming random sampling using mechanistic estimation

We have shown that for wide random MLPs, cumulant propagation outperforms random sampling for estimating the expected loss. This is a striking result, since sampling is the strong default for such problems. Instead of using sampling, cumulant propagation directly analyzes how the weights transform the input distribution. We refer to such methods as *mechanistic*, since they work by reasoning about the mechanisms of the network, in much the same way as methods from the field of mechanistic interpretability [Saphra and Wiegreffe, 2024].

Following Neyman et al. [2025], we conjecture that the performance of *any* estimation procedure can be matched (or exceeded) by a mechanistic one, providing we are given the training trajectory or selection process of any parameters involved. The intuition behind this is that it should never be better to blindly check things than to choose what to check more deliberately. This conjecture is bold: it would imply, for example, that the win rate of one chess-playing network over another can be estimated well without having the networks play chess against each other, by reasoning deductively about the rules of chess and the strengths and weaknesses of different strategies. Nevertheless, the algorithms in this paper give a first glimpse of how this might be possible.

7.2 Mechanistic training and low probability estimation

The obvious application of mechanistic loss estimation is to train neural networks, by applying gradient descent to the mechanistic loss estimate. We refer to this as *mechanistic training*. We have provided a proof of concept of this by demonstrating mechanistic distillation. However, since our method only has good guarantees for networks at initialization, it would not work well for the entirety of training. To adapt our method to partially-trained networks, we may need to adopt an approach akin to the "structure versus randomness" dichotomy [Tao, 2007]: instead of treating the weights as purely random, we could treat them as having a structured component that has been selected to give the model low loss, together with a random component. We leave further elaboration of this idea to future work.

Mechanistic training does not only offer to match or improve training efficiency: it should also produce models that generalize very differently. For example, consider a loss function with a certain "dangerous" event that is very rare but has very high loss. Stochastic gradient descent may fail to sample the event even once during training, resulting in a model that does not account for the dangerous event at all. On the other hand, mechanistic training could employ much better estimates for the probability of the rare event, resulting in a model that avoids it far more reliably, despite having potentially never seen the event happen. This could greatly improve safety, especially in the presence of distribution shifts that make the dangerous event more likely. This application of mechanistic estimation is what motivates our study of low probability estimation, and is discussed further in Wu and Hilton [2025].

7.3 Open problems

For a discussion of directions for further research, see the arXiv version of the paper.

8 Related work

For a literature review, see the first appendix.

9 Conclusion

We have demonstrated both theoretically and empirically that our sample-free method for estimating the expected loss of wide MLPs at initialization is substantially more efficient than Monte Carlo sampling. With further work, this could lead to greatly improved training efficiency for losses dominated by rare events.

10 Acknowledgments

We are grateful to Eric Neyman for work that helped to inspire this project, to Ekene Ezeunala for work on mechanistic estimation of the Hermite coefficients of GELU, and to Harshita Khera for invaluable operational support. We are also grateful to Matthew Ding, Geoffrey Irving, Vinayak Pathak, Jess Riedel, Linus Tang and Gabriel Wu for comments on drafts, as well as to the anonymous reviewers for helpful feedback on the first version of the paper.

The Alignment Research Center received funding for this work from the UK AI Security Institute's Alignment Project. WW was partially supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2040434. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. MW acknowledges the Institute for Advanced Study, and DOE grant DE-SC0009988.

References

A. Bibi, M. Alfadly, and B. Ghanem. Analytic expressions for probabilistic moments of PL-DNN with Gaussian input. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9099–9107, 2018.

P. Christiano, E. Neyman, and M. Xu. Formalizing the presumption of independence. *arXiv preprint arXiv:2211.06738*, 2022.

F. Dai and Y. Xu. *Approximation Theory and Harmonic Analysis on Spheres and Balls*. Springer Monographs in Mathematics. Springer, New York, NY, 1 edition, 2013. ISBN 978-1-4614-6659-8. doi: 10.1007/978-1-4614-6660-4.

J. Dunbar and S. Aaronson. Wide neural networks as a baseline for the computational no-coincidence conjecture. *arXiv preprint arXiv:2510.06527*, 2025.

K. Fischer, A. René, C. Keup, M. Layer, D. Dahmen, and M. Helias. Decomposing neural networks as mappings of correlation functions. *Physical review research*, 4(4):043143, 2022.

B. J. Frey and G. E. Hinton. Variational learning in nonlinear gaussian belief networks. *Neural Computation*, 11(1):193–213, 1999.

J. Gast and S. Roth. Lightweight probabilistic deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3369–3378, 2018.

A. Genz and F. Bretz. *Computation of multivariate normal and t probabilities*. Springer Science & Business Media, 2009.

O. Goldreich. Foundations of cryptography – a primer. *Foundations and Trends® in Theoretical Computer Science*, 1(1):1–116, 2005.

P. Hall. *The Bootstrap and Edgeworth Expansion*. Springer Series in Statistics. Springer New York, NY, New York, NY, 1 edition, 1992. ISBN 978-0-387-97720-1. doi: 10.1007/978-1-4612-4384-7.

B. Hanin. Random neural networks in the infinite width limit as gaussian processes. *The Annals of Applied Probability*, 33(6A):4798–4819, 2023.

B. Hanin. Random fully connected neural networks as perturbatively solvable hierarchies. *Journal of Machine Learning Research*, 25(267):1–58, 2024.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

D. Hendrycks and K. Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

A. Jacot, F. Gabriel, and C. Hongler. Neural Tangent Kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

T. Kagawa. The Hermite function expansions of the Heaviside function. *Journal of Pseudo-Differential Operators and Applications*, 6(1):21–32, 2015.

M. Kendall. Proof of relations connected with the tetrachoric series and its generalization. *Biometrika*, 32(2):196–198, 1941.

S. Kuang and X. Lin. Exact gaussian moment matching for residual networks: a second-order method. *arXiv preprint arXiv:2601.22307*, 2026.

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as Gaussian processes. In *International Conference on Learning Representations*, 2018.

V. A. Malyshev. Cluster expansions in lattice models of statistical physics and the quantum theory of fields. *Russian Mathematical Surveys*, 35(2):1–62, 1980.

R. M. Neal. Priors for infinite networks. In *Bayesian learning for neural networks*, pages 29–53. Springer, 1996.

E. Neyman, V. Lecomte, W. Wu, M. Winer, J. Hilton, and G. Robinson. Competing with sampling, 2025. URL `https://www.alignment.org/blog/competing-with-sampling/`.

J. Pennington, S. S. Schoenholz, and S. Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 47884798, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli. Exponential expressivity in deep neural networks through transient chaos. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL `https://proceedings.neurips.cc/paper_files/paper/2016/file/148510031349642de5ca0c544f31b2ef-Paper.pdf`.

D. A. Roberts, S. Yaida, and B. Hanin. *The Principles of Deep Learning Theory*. Cambridge University Press, 2022. `https://deeplearningtheory.com`.

P. T. Roy, A. B. Owen, M. Balandat, and M. Haberland. Quasi-Monte Carlo methods in python. *Journal of Open Source Software*, 8(84):5309, 2023.

N. Saphra and S. Wiegreffe. Mechanistic? In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 480–498, 2024.

S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein. Deep information propagation. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL `https://openreview.net/forum?id=H1W1UN9gg`.

E. Solomonik and J. W. Demmel. Contracting symmetric tensors using fewer multiplications. Technical report, ETH Zürich, 2015.

T. Tao. The dichotomy between structure and randomness, arithmetic progressions, and the primes. In *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006*, pages 581–608. European Mathematical Society-EMS-Publishing House GmbH, 2007.

H. Wang and L. Zhang. Convergence analysis of Hermite approximations for analytic functions. *Mathematics of Computation*, July 2025. doi: 10.1090/mcom/4127. URL `https://doi.org/10.1090/mcom/4127`. Published electronically July 16, 2025.

O. Wright, Y. Nakahira, and J. M. Moura. An analytic solution to covariance propagation in neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4087–4095. PMLR, 2024.

G. Wu and J. Hilton. Estimating the probabilities of rare outputs in language models. In *International Conference on Learning Representations*, 2025.

Related work

R.1 Covariance propagation for neural networks

A very similar approach is taken by Fischer et al. [2022], who explain how cumulant propagation can be applied to MLPs using Hermite expansions (or equivalently Gram–Charlier expansions, in their terminology). Furthermore, they use class-conditional cumulants of the input distribution to perform mechanistic training on small models in settings such as MNIST. However, their implementation and error analysis focuses on covariance propagation, i.e., the special case of cumulant propagation with maximum cumulant order 2, which corresponds to approximating activations as multivariate Gaussian distributions. A similar approach is also taken by Kuang and Lin [2026], who also study covariance propagation, but with exact calculations for off-diagonal covariances as well as variances. Our error analysis for random networks shows that the effect of these exact off-diagonal covariance calculations is lower-order in the width of the network, and would therefore only have a minimal effect on mean squared error in our setting. We build on both of these works by extending covariance propagation to arbitrarily high cumulant order, in the setting of wide random networks: providing an explicit formulation of the algorithm, checking its performance empirically, and conducting the corresponding error analysis.

R.2 Analytic uncertainty propagation

There is a body of work in a similar vein that is motivated by the problem of analytically propagating an uncertain input through a network. The most direct precursor to our own work in this area is that of Wright et al. [2024], who derive formulas for covariance propagation that can be applied to ReLU networks. Indeed, Theorem 1 of Wright et al. [2024] is a special case of our diagram summation formula, as explained in Appendix A.1. Their work builds in turn upon work of Gast and Roth [2018], who derive formulas for propagating means and variances only, but not off-diagonal covariances. Thus our work generalizes these methods to arbitrarily high cumulant order, and shows moreover that this achieves arbitrarily high accuracy in the large-width limit. Separately, Bibi et al. [2018] derive an alternative covariance propagation formula specifically for ReLU networks with 1 hidden layer. This whole line of work dates at least as far back as Frey and Hinton [1999], who already provided analytic formulas for the mean and variance of ReLU and several other activation functions on a Gaussian input distribution.

R.3 Wide random networks

There is an extensive literature on the behavior of neural networks with random weights in the limit as the width tends to infinity, as detailed in Roberts, Yaida, and Hanin [2022]. Some of the key ideas in this area include the view of infinitely-wide random networks as Gaussian processes [Neal, 1996, Lee et al., 2018, Hanin, 2023] and the Neural Tangent Kernel associated with the training of infinitely-wide networks [Jacot et al., 2018]. Notably, Hanin [2024] studies the cumulants of outputs of random networks in the infinite-width limit, and Dunbar and Aaronson [2025] studies how the covariances decay with depth. However, this body of work focuses almost exclusively on the output distribution *for fixed inputs over random weights*, whereas we study the output distribution *for fixed weights over random inputs*. This distinction is critical: instead of studying properties of the network *ensemble*, we study algorithms that take in the weights of a *particular* network.

R.4 Mechanistic estimation and low probability estimation

The question of whether there is a general framework for deductive, probabilistic estimation was posed by Christiano et al. [2022], who also introduced cumulant propagation in the context of arithmetic circuits. Wu and Hilton [2025] studied both mechanistic and non-mechanistic methods for low probability estimation, and their *activation extrapolation* methods have features in common with our low probability estimation methods. The broad conjecture that the performance of any estimation procedure can be matched or exceeded by a mechanistic one was originally posed by Neyman et al. [2025].

A Diagram summation formula for cumulants

The diagram summation formula for cumulants is our key tool for propagating cumulants (as defined in Section 3.1) through non-linear activation functions. Given a random variable $Z \in \mathbb{R}^n$ and an activation function $\phi : \mathbb{R} \to \mathbb{R}$, this formula expresses the cumulants of $\phi(Z_1), \ldots, \phi(Z_n)$ in terms of the cumulants of $Z_1, \ldots, Z_n$ and the Hermite coefficients of ϕ (as defined in Section 3.2). In order to state it, we begin by defining diagrams and their attributes.

Definition A.1. Let $\underline{k} := (k_1, \ldots, k_r)$ be an r-tuple of non-negative integers. A *diagram* on $\underline{k}$ is a partition π of the set

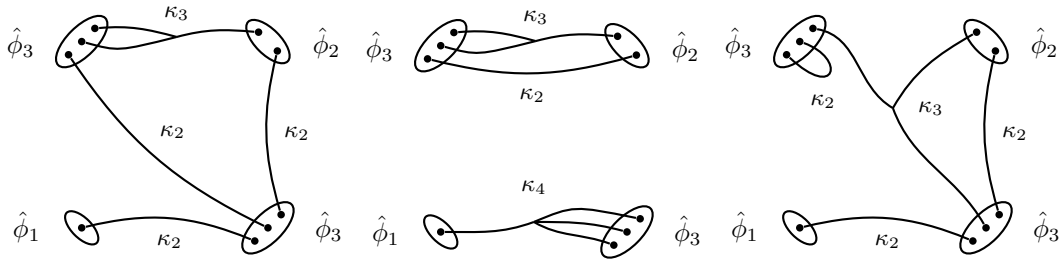
$$V := \bigcup_{v=1}^{r} \{(v,1), (v,2), \ldots, (v,k_v)\}.$$

We say that π is connected to mean that for any two elements $(v_{\text{start}}, w_{\text{start}}), (v_{\text{end}}, w_{\text{end}}) \in V$, there is a sequence $(v_1, w_1), \ldots, (v_m, w_m) \in V$ with $(v_1, w_1) = (v_{\text{start}}, w_{\text{start}})$, $(v_m, w_m) = (v_{\text{end}}, w_{\text{end}})$ and for all $i = 1, \ldots, m-1$, either $v_i = v_{i+1}$ or (v_i, w_i) and (v_{i+1}, w_{i+1}) lie in the same part of π. In other words, every two elements of V are linked by a path for which consecutive points lie either in the same one of the r sets used to define V or in the same part of π. (In the degenerate case in which $k_v = 0$ for some v, we consider π to be connected if and only if $r = 1$.) The set of all connected diagrams on $\underline{k}$ is denoted $\mathrm{cDia}(\underline{k})$.

We say that π is $[\leq M]$-*mixed* to mean that every part $S \in \pi$ of size at most M satisfies $\{(v_1, w_1), (v_2, w_2)\} \subseteq S$ for some $v_1 \neq v_2$.

The set of all connected $[\leq M]$-mixed diagrams on $\underline{k}$ is denoted $\mathrm{cDia}_{[\leq M]}(\underline{k})$.

We visualize diagrams by drawing a vertex for each pair (v, w), encircling the vertex set $\{(v, w) : w \in \{1, \ldots, k_v\}\}$ for each v, and drawing each part of the partition as a hyperedge. We draw some examples of diagrams and explain whether they are connected or $[\leq 2]$-mixed in Figure 6.



(a) The diagram is both connected and $[\leq 2]$-mixed, and so the term **is** included.

(b) The diagram is $[\leq 2]$-mixed but not connected, and so the term **is not** included.

(c) The diagram is connected but not $[\leq 2]$-mixed (see top left loop), and so the term **is not** included.

Figure 6: Drawings of diagrams corresponding to included and excluded terms from the Hermite-based diagram summation formula, for $(k_1, k_2, k_3, k_4) = (3, 2, 3, 1)$ (clockwise from top left). Hermite coefficients correspond to encircled vertex sets and cumulants correspond to hyperedges.

Our diagrams are essentially cluster-expansion diagrams from statistical physics [Malyshev, 1980], except that we generalize pairwise edges to hyperedges in order to account for third- and higher-order cumulants.

With these definitions in place, together with the notation for Hermite coefficients introduced in Section 3.2 (equation (1)), we may now state the formula. We refer to the formula as *Hermite-based* to distinguish it from an analogous but simpler *polynomial* diagram summation formula, which we introduce in Appendix S.2.[11]

[11] We state the diagram summation formula here for polynomial activation functions, but it also applies to non-polynomials under certain analytic conditions. We discuss this in more detail in Appendix S.7.

Theorem A.2 (Hermite-based diagram summation formula for cumulants)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi : \mathbb{R} \to \mathbb{R}$ be a polynomial. Then*

$$\kappa\left[\phi\left(Z_{j_1}\right), \ldots, \phi\left(Z_{j_r}\right)\right] = \sum_{\underline{k} \in \mathbb{Z}_{\geq 0}^r} \left(\prod_{v=1}^{r} \frac{1}{k_v!} \widehat{\phi}_{k_v}^{(\mathbb{E}[Z_{j_v}], \mathrm{Var}[Z_{j_v}])} \right) \sum_{\pi \in \mathrm{cDia}_{[\leq 2]}(\underline{k})} \prod_{S \in \pi} \kappa\left[Z_{j_v} : (v, w) \in S\right].$$

We prove Theorem A.2 in Appendix S.7.

Note that in this formula, we do not use a single Hermite expansion of ϕ, but rather take separate Hermite expansions of ϕ with respect to Gaussian distributions with means and variances matching each of $Z_1, \ldots, Z_n$.

The special case of this formula in which ϕ is the step function $\mathbb{1}_{z \geq 0}$ and Z is multivariate Gaussian appears in Kendall [1941] (see Genz and Bretz [2009, equation (3.2)]).

A.1 Covariance propagation formula

To help understand the Hermite-based diagram summation formula for cumulants, it is instructive to consider the special case applicable to covariance propagation, i.e., cumulant propagation with maximum cumulant order $K = 2$.

Corollary A.3 (Wright et al. [2024])**.** *Let $Y \in \mathbb{R}^n$ be a multivariate Gaussian random variable and let $\phi : \mathbb{R} \to \mathbb{R}$ be a polynomial. Then*

$$\mathrm{Cov}\left[\phi\left(Y_{j_1}\right), \phi\left(Y_{j_2}\right)\right] = \sum_{k=1}^{\infty} \frac{1}{k!} \widehat{\phi}_k^{(\mathbb{E}[Y_{j_1}], \mathrm{Var}[Y_{j_1}])} \widehat{\phi}_k^{(\mathbb{E}[Y_{j_2}], \mathrm{Var}[Y_{j_2}])} \mathrm{Cov}\left[Y_{j_1}, Y_{j_2}\right]^k.$$

This formula exactly matches the covariance propagation formula of Wright et al. [2024, Theorem 1] using the integration-by-parts identity

$$\widehat{\phi}_k^{(\mu, \sigma^2)} := \frac{1}{\sigma^k} \mathbb{E}\left[\phi\left(Y\right) \mathrm{He}_k\left(\frac{Y - \mu}{\sigma}\right)\right] = \mathbb{E}\left[\phi^{(k)}\left(Y\right)\right] = \frac{\partial^k}{\partial \mu^k} \mathbb{E}\left[\phi\left(Y\right)\right]$$

for $Y \sim \mathcal{N}\left(\mu, \sigma^2\right)$.

Proof. Apply Theorem A.2 in the case $r = 2$ to Y, giving an expression for $\mathrm{Cov}\left[\phi\left(Y_{j_1}\right), \phi\left(Y_{j_2}\right)\right]$. Since Y is multivariate Gaussian, its third- and higher-order cumulants vanish, and so the only diagrams on (k_1, k_2) that survive have the form $\{\{(1, \rho(w)), (2, w)\} : w \in \{1, \ldots, k_1\}\}$ for some permutation ρ of $\{1, \ldots, k_1\}$, with $k_1 = k_2$. The number of such permutations is $k_1!$, which cancels out one of the factors of $\frac{1}{k_1!}$. Finally, the connectedness condition eliminates the diagram with $k_1 = k_2 = 0$, leaving a sum starting at $k_1 = k_2 = 1$. □

B Mean and covariance propagation

In this Appendix we provide pseudocode for the basic version of our cumulant propagation algorithm in the cases $K = 1$ (*mean propagation*) and $K = 2$ (*covariance propagation*), where K is the maximum cumulant order. We also indicate roughly how these special cases derive from our general algorithm, and how the ablated versions differ. A precise formulation of our general algorithm is significantly more involved, and is given in Appendix S.4.

B.1 Pseudocode

Mean propagation tracks both the mean and the trace of the covariance matrix of activations.[12] Tracking the entire diagonal of the covariance matrix would have lower mean squared error, but we provide the version that is consistent with our more general formulation. Pseudocode is as follows.

Algorithm 1 Mean propagation ($K = 1$)

Require: weights $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)} \in \mathbb{R}^{n \times n}$, activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$

$\boldsymbol{\mu} \leftarrow \mathbf{0} \in \mathbb{R}^n$
$\sigma^2_{\text{mean}} \leftarrow 1 \in \mathbb{R}$
for $\ell = 1, \ldots, L+1$ **do**
  $\boldsymbol{\mu} \leftarrow \mathbf{W}^{(\ell)} \boldsymbol{\mu}$
  $\sigma_i^2 \leftarrow \sigma^2_{\text{mean}} \sum_{j=1}^n \left(W_{ij}^{(\ell)} \right)^2$ **for** $i = 1, \ldots, n$
  if $\ell \leq L$ **then**
    $a_i \leftarrow \mathbb{E}_{Y \sim \mathcal{N}(\mu_i, \sigma_i^2)} \left[\phi_\ell \left(Y \right) \right]$ **for** $i = 1, \ldots, n$
    $b_i \leftarrow \mathbb{E}_{Y \sim \mathcal{N}(\mu_i, \sigma_i^2)} \left[\phi_\ell \left(Y \right)^2 \right]$ **for** $i = 1, \ldots, n$
    $\mu_i \leftarrow a_i$ **for** $i = 1, \ldots, n$
    $\sigma^2_{\text{mean}} \leftarrow \frac{1}{n} \sum_{i=1}^n \left(b_i - a_i^2 \right)$
  end if
end for
return $\boldsymbol{\mu}$

Covariance propagation tracks both the mean and the full covariance matrix of activations. Pseudocode is as follows.

Algorithm 2 Covariance propagation ($K = 2$)

Require: weights $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)} \in \mathbb{R}^{n \times n}$, activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$

$\boldsymbol{\mu} \leftarrow \mathbf{0} \in \mathbb{R}^n$
$\boldsymbol{\Sigma} \leftarrow \mathbf{I}_n \in \mathbb{R}^{n \times n}$
for $\ell = 1, \ldots, L+1$ **do**
  $\boldsymbol{\mu} \leftarrow \mathbf{W}^{(\ell)} \boldsymbol{\mu}$
  $\boldsymbol{\Sigma} \leftarrow \mathbf{W}^{(\ell)} \boldsymbol{\Sigma} \left(\mathbf{W}^{(\ell)} \right)^\top$
  if $\ell \leq L$ **then**
    $a_i \leftarrow \mathbb{E}_{Y \sim \mathcal{N}(\mu_i, \Sigma_{ii})} \left[\phi_\ell \left(Y \right) \right]$ **for** $i = 1, \ldots, n$
    $b_i \leftarrow \mathbb{E}_{Y \sim \mathcal{N}(\mu_i, \Sigma_{ii})} \left[\phi_\ell \left(Y \right)^2 \right]$ **for** $i = 1, \ldots, n$
    $c_i \leftarrow \frac{1}{\sqrt{\Sigma_{ii}}} \mathbb{E}_{Y \sim \mathcal{N}(\mu_i, \Sigma_{ii})} \left[\phi_\ell \left(Y \right) \left(\frac{Y - \mu_i}{\sqrt{\Sigma_{ii}}} \right) \right]$ **for** $i = 1, \ldots, n$
    $\mu_i \leftarrow a_i$ **for** $i = 1, \ldots, n$
    $\Sigma_{ii} \leftarrow b_i - a_i^2$ **for** $i = 1, \ldots, n$
    $\Sigma_{ij} \leftarrow \Sigma_{ij} c_i c_j$ **for** $i, j = 1, \ldots, n$ **if** $i \neq j$
  end if
end for
return $\boldsymbol{\mu}$

[12] Our usage of the term *mean propagation* differs from that of Christiano et al. [2022], who only track the mean, like in our ablated version of mean propagation.

When the activation functions $\phi_1, \ldots, \phi_L$ are piecewise polynomials, such as when they are all ReLU, the Gaussian expectations in these algorithms can be computed straightforwardly. Each expectation can be decomposed into a finite sum of integrals of polynomials against the standard Gaussian density over finite intervals. By repeated integration by parts, these integrals can be expressed in terms of the standard Gaussian density function and cumulative distribution function evaluated at the interval endpoints. For explicit formulas in the case of ReLU, see See Proposition S.3.8, and for an alternative approach that we use for $\tanh$ and GELU, see Appendix F.3.

B.2 Derivation from cumulant propagation

It is instructive to consider how mean propagation and covariance propagation derive from the more general formulation of our basic cumulant propagation algorithm.

The matrix multiplication step (lines 4–5 in both algorithms) is straightforward. If $X \in \mathbb{R}^n$ is a random variable with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, then $\mathbf{W}X$ has mean $\mathbf{W}\boldsymbol{\mu}$ and covariance matrix $\mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top$. The higher cumulants of X transform similarly. In mean propagation, we avoid computing the full covariance matrix explicitly, and instead approximate $\boldsymbol{\Sigma}$ by $\sigma^2_{\text{mean}}\mathbf{I}_n$.

For the activation function step, we use a Gaussian approximation. Given a random variable $Z \in \mathbb{R}^n$ and an activation function $\phi : \mathbb{R} \to \mathbb{R}$ (to be applied coordinatewise), we approximate the mean and covariance matrix of $\phi(Z)$ by the mean and covariance matrix of $\phi(Y)$, where $Y \in \mathbb{R}^n$ is a multivariate Gaussian with the same mean and covariance matrix as Z.

To compute the means and variances of $\phi(Y)$, we only need to evaluate one-dimensional integrals, and these can be computed exactly when ϕ is a piecewise polynomial. Using power cumulants allows us to take advantage of this by handling these variances separately from the off-diagonal covariances. For the off-diagonal covariances of $\phi(Y)$ (lines 9 and 12 of covariance propagation), we use the leading-order term of the formula given in Corollary A.3.

B.3 Ablated mean and covariance propagation

To help explain the ablated version of cumulant propagation described in Appendix E, we describe the changes that would need to be made to mean and covariance propagation to obtain their ablated versions.

For mean propagation, the ablated version no longer tracks the trace of the covariance matrix. This amounts to omitting lines 2, 5, 8 and 10, and replacing σ_i^2 by the default value 1 in line 7. Since only the mean is tracked, the use of power cumulants makes no difference.

For covariance propagation, the ablated version no longer uses the power cumulants of the outputs of activation functions. This amounts to omitting lines 8 and 11, and removing the condition $i \neq j$ from line 12.

C Additional widths and depths

Figure 7 compares the mean squared error of our sample-free estimation procedures to that of Monte Carlo sampling at different widths from 16 to 256 and different numbers of hidden layers from 2 to 12. Our procedures are exact when there is only 1 hidden layer.

Our algorithms perform poorly at low width, especially when the number of hidden layers L is large, and can even get *worse* with the maximum cumulant order K in this setting. However, for any fixed L and K, our algorithms match or exceed the performance of sampling as the width grows.

Note also that at low depth, factorized algorithms use fewer FLOPs than their unfactorized counterparts, but at high enough depth (for fixed width), they use more FLOPs.

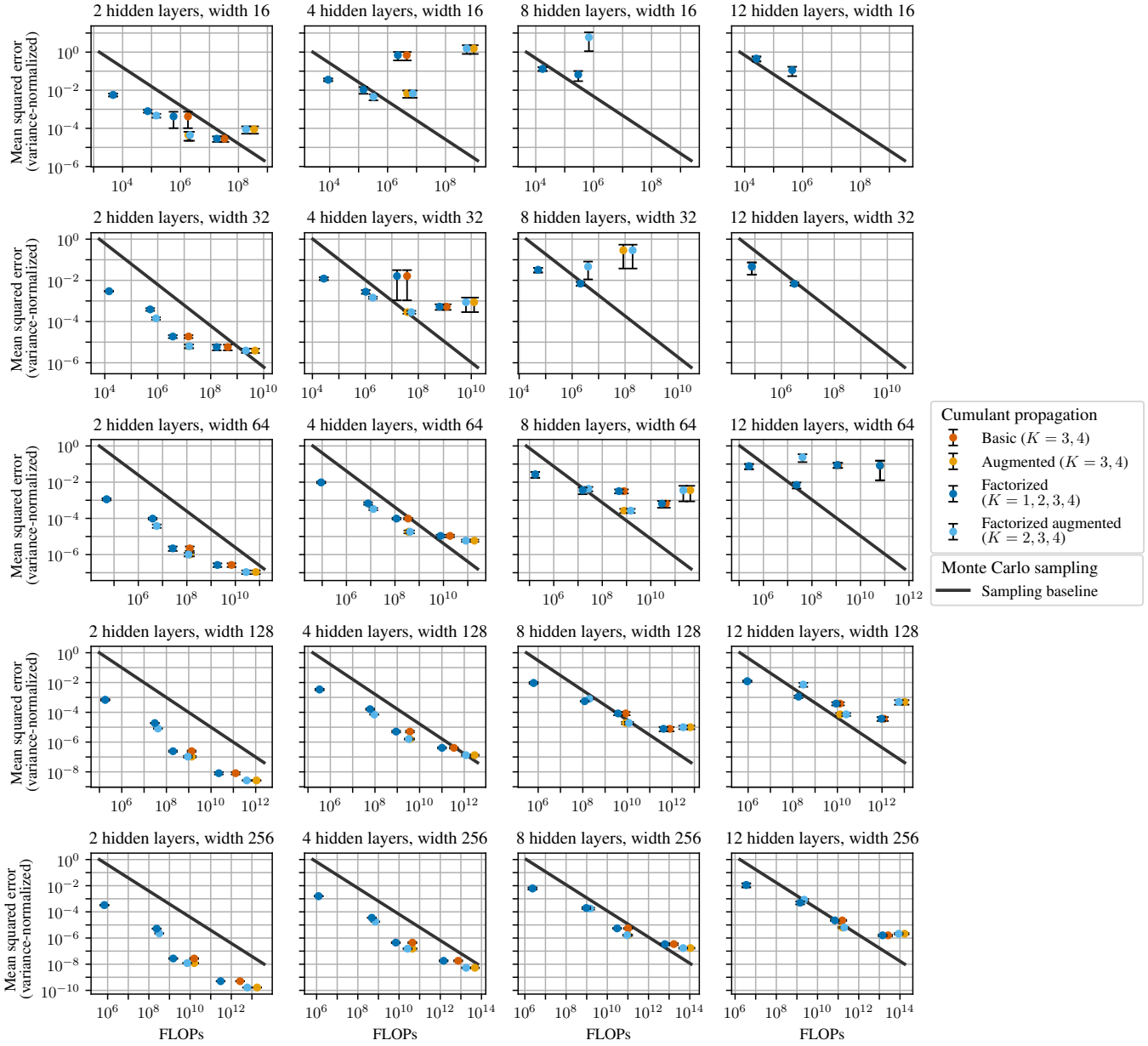


Figure 7: Performance of all versions of our algorithm for networks of different widths and depths and K varying from 1 to 4. Error bars show ± 1 standard error over 5 random seeds. Missing points have variance-normalized mean squared error greater than 10.

D Scaling with depth

As discussed in Section 5, we conjecture that our algorithms have mean squared error at most $c_K(L/n)^K$, where n is the width, L is the number of hidden layers, K is the maximum cumulant order and c_K is a constant that depends only on K. We conjecture this depth scaling based on a heuristic analysis of the error, but we can also test it empirically. To do this, we plot the mean squared error of the basic version of algorithm as a function of the number of hidden layers in Figure 8. As conjectured, the mean squared error roughly follows a curve proportional to L^K when L is large.

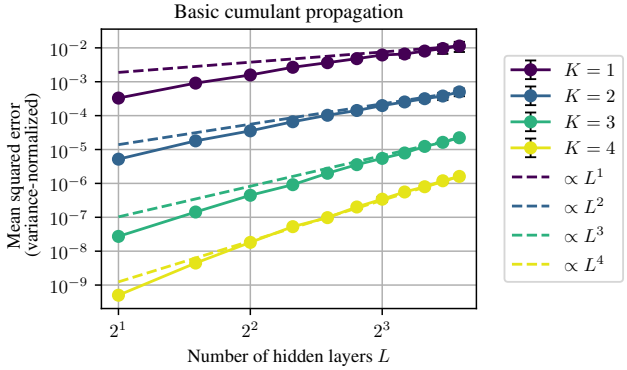


Figure 8: Depth scaling of the basic (equivalently, factorized) version of our algorithm on networks with width 256. Dashed lines are those with the conjectured large-n slope passing through the rightmost point. Error bars show ± 1 standard error over 5 random seeds.

This depth scaling is worse than Monte Carlo sampling, whose error does not increase with depth. We believe that a sample-free algorithm can be developed whose error also does not increase with depth either, but we leave this problem to future work.

E Power cumulant ablation

As discussed in Section 6.4, we checked the performance of an ablated version of our algorithm without the two adjustments described in Section 4.1. Specifically, the differences between the basic and ablated versions of our algorithm are as follows:

1. As discussed in Section 3.3, we use the Hermite-based diagram summation formula for cumulants, Theorem A.2, to compute the cumulants of $\phi(Z_1), \ldots, \phi(Z_n)$ from the cumulants of $Z_1, \ldots, Z_n$ for a non-linear activation function $\phi : \mathbb{R} \to \mathbb{R}$. The basic version of our algorithm uses this formula to compute the power cumulants of $\phi(Z_1), \ldots, \phi(Z_n)$, as defined in Section 4.1, by taking Hermite expansions of *powers* of ϕ. It then uses these power cumulants to obtain the cumulants of $\phi(Z_1), \ldots, \phi(Z_n)$. The ablated version of our algorithm instead uses the Hermite-based diagram summation formula to compute the cumulants of $\phi(Z_1), \ldots, \phi(Z_n)$ directly, only taking Hermite expansions of ϕ itself.
2. When the maximum cumulant order K is odd, the basic version of our algorithm tracks the full trace of the $(K+1)^{\text{th}}$-order cumulant tensor, as defined in Section 4.1. The ablated version of our algorithm instead approximates this as zero. When K is even, there is no difference between the two algorithms in this respect.

To help explain precisely what we mean by this, we explain how to ablate the pseudocode for $K = 1$ (mean propagation) and $K = 2$ (covariance propagation) in Appendix B.3.

We compared the performance of an *ablated factorized* version of our algorithm to the unablated factorized versions and to Monte Carlo sampling. Our results for networks with width 256 and different numbers of hidden layers from 2 to 12 are shown in Figure 9. As expected, the ablated version is significantly worse than the unablated versions, and is no longer competitive with Monte Carlo sampling. The scaling of the ablated version of our algorithm with width (which is unaffected by factorization, since this only affects the runtime) is also discussed in Section 6.4.

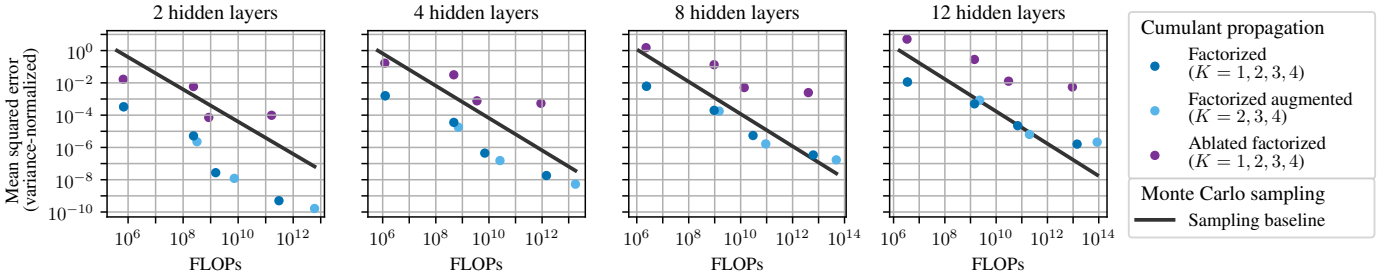


Figure 9: Performance of the ablated factorized version of our algorithm, as well as the unablated factorized versions, for different networks of width 256. Mean over 5 random seeds shown. (Error bars hidden because they would be too small to be readable.)

F Other activation functions

In this Appendix, we validate the performance of our algorithms for activation functions other than ReLU. We consider two additional activation functions:

- $\mathrm{GELU}\,(z) := z\Phi\,(z)$, where Φ is the standard Gaussian cumulative distribution function [Hendrycks and Gimpel, 2016]
- $\tanh\,(z) := \frac{1-e^{-2z}}{1+e^{-2z}}$

For the remainder of this Appendix, fix the number of hidden layers $L \geq 0$.

F.1 Bias terms

For GELU, we consider multilayer perceptrons (MLPs) without bias terms, as defined in Definition 2.1.

For $\tanh$, if we were to consider MLPs without bias terms, then the expectation of every output neuron would be zero by symmetry, making the estimation problem trivial. We therefore incorporate bias terms, meaning that the weights of the MLP are $\boldsymbol{\theta} = \left(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(L+1)}\right)$ with $\mathbf{W}^{(\ell)} \in \mathbb{R}^{n\times n}$ and $\mathbf{b}^{(\ell)} \in \mathbb{R}^{n}$ for $\ell = 1, \ldots, L+1$, and $\mathbf{b}^{(\ell)}$ is added to the ℓth hidden layer's activation vector immediately after multiplying it by $\mathbf{W}^{(\ell)}$.

It is easy to adjust our sample-free estimation procedures to incorporate these bias terms. Our cumulant propagation algorithms simply add the bias vector to the estimate of the mean at the corresponding layer. Equivalently, this can be thought of as using a different activation function for each neuron that has been shifted by the corresponding bias component.

F.2 Weight initialization

For GELU, we use He initialization [He et al., 2015], which ensures that when n is large and the inputs have mean 0 and variance 1, all pre-activations and outputs also have mean 0 and variance $O(1)$ (treating the depth as a constant). This is defined for MLPs without bias terms as follows.

Definition F.1. Consider an MLP with activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$ and weights $\boldsymbol{\theta} = \left(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}\right) \in \mathbb{R}^{(L+1)\times n\times n}$. We say that $\boldsymbol{\theta}$ is *He-initialized* to mean that $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}$ are independent random variables with

$$W_{ij}^{(1)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{1}{n}\right)$$
$$W_{ij}^{(\ell+1)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{1}{n\,\mathbb{E}_{Z\sim\mathcal{N}(0,1)}[\phi_\ell\,(Z)^2]}\right), \quad \ell = 1, \ldots, L$$

for $i, j = 1, \ldots, n$.

For $\tanh$, we use a *critical initialization* [Poole et al., 2016, Schoenholz et al., 2017, Pennington et al., 2017], defined for MLPs with bias terms as follows.

Definition F.2. Consider an MLP and with activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$ and weights $\boldsymbol{\theta} = \left(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(L+1)}\right)$. We say that $\boldsymbol{\theta}$ is *critical at variance* q^* to mean that $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(L+1)}$ are independent random variables with

$$W_{ij}^{(1)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{q^*}{n}\right)$$
$$b_i^{(1)} = 0$$
$$W_{ij}^{(\ell+1)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{\sigma_{\ell+1,\mathrm{w}}^2}{n}\right), \qquad \ell = 1, \ldots, L$$
$$b_i^{(\ell+1)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \sigma_{\ell+1,\mathrm{b}}^2\right), \qquad \ell = 1, \ldots, L$$

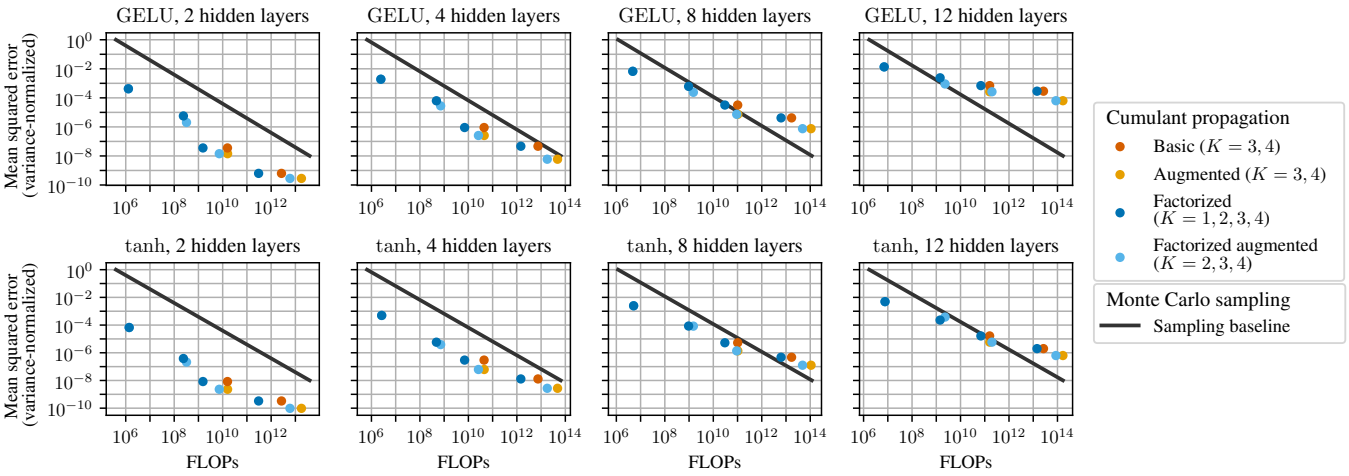


Figure 10: Performance of all versions of our algorithm for different GELU and tanh networks of width 256. Mean over 5 random seeds shown. (Error bars hidden because they would be too small to be readable.)

for $i, j = 1, \ldots, n$, where the initialization scales $\sigma_{\ell,\mathrm{w}}$ and $\sigma_{\ell,\mathrm{b}}$ are chosen to satisfy

$$q^* = \sigma^2_{\ell+1,\mathrm{w}} \mathbb{E}[\phi_\ell(\sqrt{q^*}Z)] + \sigma^2_{\ell+1,\mathrm{b}}, \qquad \ell = 1, \ldots, L$$
$$1 = \sigma^2_{\ell+1,\mathrm{w}} \mathbb{E}[\phi'_\ell(\sqrt{q^*}Z)^2], \qquad \ell = 1, \ldots, L$$

for $Z \sim \mathcal{N}(0, 1)$.

For $\tanh$, we use critical initialization at $q^* = 0.85$, corresponding to $\sigma^2_{\mathrm{w}} = 2.025$ and $\sigma^2_{\mathrm{b}} = 0.111$. This is the "large" $\tanh$ setting used in Pennington et al. [2017, §3].

As an aside, note that He initialization and critical initialization at $q^* = 1$ coincide for ReLU, both giving weight variance $\frac{2}{n}$ and zero bias. Thus our main ReLU experiments can be thought of as being performed in either initialization.

F.3 Hermite coefficient estimation

In order to estimate Hermite coefficients, as defined in Section 3.2, our sample-free estimation procedures must estimate Gaussian expectations of a power of the activation function multiplied by a polynomial. For both GELU and $\tanh$, this amounts to estimating $\mathbb{E}_{Z\sim\mathcal{N}(0,1)}[\psi(Z)]$ for some $\psi : \mathbb{R} \to \mathbb{R}$ that is analytic and of polynomial growth on an infinite strip around the real axis. A standard approach in such cases is Gauss–Hermite quadrature [Wang and Zhang, 2025], which produces the estimate

$$\mathbb{E}_{Z\sim\mathcal{N}(0,1)}[\psi(Z)] \approx \sum_{i=1}^{m} \frac{(m-1)!}{m} \frac{1}{\mathrm{He}_{m-1}(x_i)^2} \psi(x_i),$$

where He_{m-1} is the $(m-1)^{\text{th}}$ probabilist's Hermite polynomial and $x_1, \ldots, x_m$ are its distinct real roots.[13] For our experiments, we take the number of roots to be $m = 100$.

We also use the same technique to estimate the Gaussian expectations used in the definition of He initialization and critical initialization.

F.4 Results

Figure 10 compares the mean squared error of our sample-free estimation procedures to that of Monte Carlo sampling for GELU and tanh networks of width 256 with different numbers of hidden layers from 2 to 12. Our results are qualitatively similar to our results for ReLU.

[13] This method is arguably not "sample-free" and hence not a "mechanistic" way to estimate this one-dimensional integral, but we defer a resolution to this issue to future work.

G Low probability estimation baselines

Our results on low probability estimation are given in Section 6.6. Here we explain our four baseline methods to which we compare cumulant propagation.

1. **Monte Carlo sampling.** This is our usual baseline algorithm. However, it performs very poorly once the probability drops below $1/(\text{number of samples})$, since the squared error is very large in the unlikely event that there is a positive sample.
2. **Monte Carlo sampling with Gaussian extrapolation.** We use Monte Carlo sampling to estimate the mean and variance of each input to the final threshold function $\mathbb{1}_{z>3}$, and output the probability that a Gaussian with that mean and variance would exceed the threshold of 3. This method is similar to the Gaussian Logit Difference method of Wu and Hilton [2025]. Empirically, this method performs similarly to using cumulant propagation to estimate the mean and variance of each of these neurons, i.e., most of the error comes from failing to account for the third- and higher-order cumulants of the inputs to the final threshold function.
3. **Monte Carlo sampling with cumulant extrapolation.** We use Monte Carlo sampling to estimate the mean, variance and third cumulant of each input to the final threshold function $\mathbb{1}_{z>3}$. We then use our usual cumulant propagation algorithm, involving Hermite expansions, to propagate these through the final threshold function. The purpose of this baseline is to check whether it is better to use sampling or cumulant propagation to compute these cumulants.
4. **Zero.** This estimate simply outputs zero, which always has a relative error of 100%. We include this because it outperforms naive Monte Carlo sampling at low probabilities.

Our empirical results show that once the probability drops below $1/(\text{number of samples})$, cumulant propagation performs best, followed by Monte Carlo sampling with cumulant extrapolation, followed by Monte Carlo sampling with Gaussian extrapolation, followed by zero, followed by naive Monte Carlo sampling. This shows that computing the third-order cumulants of the inputs to the threshold function is helpful for low probability estimation, and moreover that cumulant propagation computes these more accurately than Monte Carlo sampling. At sufficiently small probabilities, however, none of the methods significantly outperform zero.

H Mechanistic distillation setup

Our results on mechanistic distillation are given in Section 6.7. Here we provide further details of our experimental setup.

We distill a random teacher MLP M_{θ_T} with hidden dimension n_T and L hidden layers into a student MLP M_{θ_S} with hidden dimension n_S and the same number of hidden layers. Both networks have input and output dimension n, use ReLU activation functions and are initialized using He initialization [He et al., 2015]. We optimize the student parameter θ_S to minimize the loss

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{X\sim\mathcal{N}(0,\mathbf{I}_n)}\left[\left(M_{\theta_T}\left(X\right)_i - M_{\theta_S}\left(X\right)_i\right)^2\right].$$

In our experiments, we take $n = n_T = 16$, $n_S = 8$ and $L = 4$, since the learning problem can be challenging for much larger n_T.

We train the student network in two different ways. The first way uses ordinary stochastic gradient descent, which takes steps of gradient descent on estimates of the loss produced using Monte Carlo sampling. The second way uses mechanistic training, which takes steps of gradient descent on estimates of the loss produced by cumulant propagation. In our experiments, we use the basic version of our cumulant propagation algorithm with maximum cumulant order $K = 2$, which has good mean squared error even for widths as low as 8. For stochastic gradient descent, we use a batch size of 36, which uses a comparable number of FLOPs per gradient step. In both cases, we train for 256 gradient steps and tune the learning rate to within a factor of 2 to optimize the final loss, which amounts to a learning rate of 0.25 for stochastic gradient descent and 0.5 for mechanistic training.

To estimate the above loss with cumulant propagation, we stack the hidden layer activations of the teacher and student networks into vectors of length $n_T + n_S$. This allows the output of both networks to be computed as the output of a single network whose ℓ^{th} weight matrix is the block diagonal matrix

$$\mathbf{W}^{(\ell)} = \begin{pmatrix} \mathbf{W}_T^{(\ell)} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_S^{(\ell)} \end{pmatrix},$$

where $\mathbf{W}_T^{(\ell)}$ and $\mathbf{W}_S^{(\ell)}$ are the ℓth weight matrices of the teacher and student networks respectively. We run cumulant propagation to estimate the mean and covariance matrix of this combined network, starting from a Gaussian input with covariance matrix

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{I}_n \\ \mathbf{I}_n & \mathbf{I}_n \end{pmatrix},$$

corresponding to a duplicated input. We then estimate the loss in terms of the mean and covariance matrix of the output using the identity

$$\mathbb{E}[(A-B)^2] = \left(\mathbb{E}\left[A\right] - \mathbb{E}\left[B\right]\right)^2 + \mathrm{Var}\left(A\right) + \mathrm{Var}\left(B\right) - 2\,\mathrm{Cov}\left(A, B\right).$$

Implemented naively, this computation involves redundant computation because of the zeros in the weight matrices, but in order to determine the appropriate batch size for stochastic gradient descent, we adjust the FLOP count to exclude this redundant computation, in a similar manner to Appendix J.

Our results are shown in Figure 5 and are discussed in Section 6.7. The average variance of an output neuron is 0.23, so both forms of training are producing some non-trivial feature matching. It is also interesting to compare the mechanistic loss estimate used for mechanistic training to the stochastic loss estimate for the same training run produced using Monte Carlo sampling: the mechanistic loss estimate has bias but no variance, whereas the stochastic loss estimate has variance but no bias. This may favor stochastic gradient descent, which can accumulate uncorrelated error across updates. We leave a remedy to this problem for mechanistic training to future work.

I Wall-clock time

We compare the mean squared error of all 4 versions of our algorithm as a function of wall-clock time to that of Monte Carlo sampling. Our implementations were written in PyTorch and run on B200 GPUs. Our results for networks with width 256 and different numbers of hidden layers from 2 to 12 are shown in Figure 11.

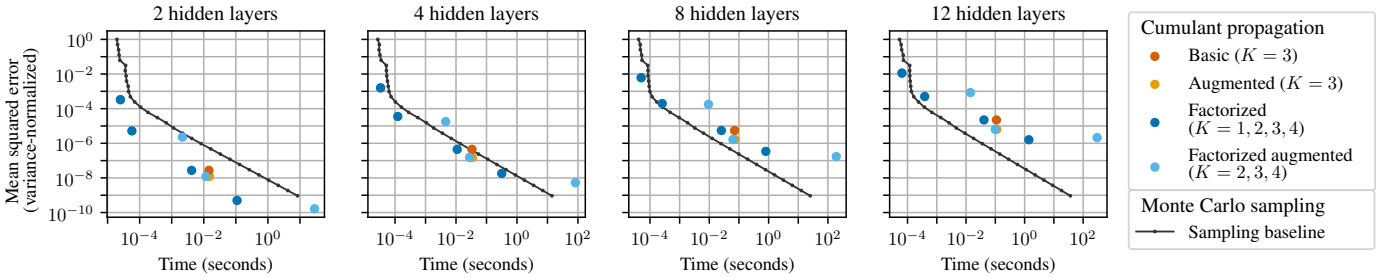


Figure 11: Wall-clock performance of all versions of our algorithm for different networks of width 256. Mean over 5 random seeds shown. (Error bars hidden because they would be too small to be readable.)

Our algorithms still outperform sampling for networks with 2 hidden layers, but their relative performance is generally worse than when runtime is measured in FLOPs. This appears to be largely because our algorithms spend the majority of the wall-clock time on a small fraction of the FLOPs that are implemented inefficiently in pure Python. In addition, they perform a significant amount of redundant computation, as described in Appendix J. We expect that with more implementation effort, wall-clock time could be made to track FLOP count much more closely, especially at large width, but we leave this to future work.

J FLOP counting

To measure the FLOPs used by both our cumulant propagation algorithm and sampling baselines, we use an extension of PyTorch's `FlopCounterMode` with the modifications listed below.

Entrywise operation tracking: Unlike sampling baselines, our cumulant propagation algorithm expends a large portion of its FLOPs in entrywise operations during the nonlinear step.[14] Thus, for an accurate comparison of FLOPs, we add tracking for entrywise arithmetic operations (`+-*/`), which are not counted in the native `FlopCounterMode`.

Mock symmetric tensor kernels: Cumulant tensors are symmetric, but our proof-of-concept implementation of cumulant propagation uses full `torch.Tensor` objects. To simulate the number of FLOPs that would be consumed by an implementation with efficient symmetric tensor CUDA kernels, we multiply FLOP counts for elementwise operations on symmetric tensors by the adjustment factor

$$\alpha_{n,d} := \frac{\dim \mathcal{S}^d(\mathbb{R}^n)}{\dim \mathcal{T}^d(\mathbb{R}^n)} = \binom{n+d-1}{d} n^{-d}.$$

(See Appendix S.3.1 for notation.)

More generally, given an integer partition $\lambda = (1^{c_1} 2^{c_2} \dots d^{c_d}) \vdash d$ with $c = \sum_i c_i$ blocks, the adjustment factor for λ-symmetric tensors is

$$\alpha_{\pi,n} := \frac{\dim \mathcal{S}^\lambda(\mathbb{R}^n)}{\dim \mathcal{T}^d(\mathbb{R}^n)} = \prod_{i=1}^{d} \binom{n+c_i-1}{c_i} n^{-c_i}.$$

Now consider the symmetric contraction operation $- \odot \mathbf{W} \colon \mathcal{S}^d(\mathbb{R}^n) \to \mathcal{S}^d(\mathbb{R}^n)$. The optimal path is to contract the weight matrices $\mathbf{W}$'s into the d-tensor one at a time. Over full tensors, $\mathbf{W}$-contraction takes $2n^{d+1}$ FLOPs: there are n^d output entries, and each takes $2n$ FLOPs to compute (counting add and multiply separately). Thus $- \odot \mathbf{W}$ takes $2dn^{d+1}$ FLOPs total. Over symmetric tensors, observe that the intermediate after contracting in the ith W is $(1^i 2^{d-i})$-symmetric, and thus only needs $\dim \Sigma^{(1^i 2^{d-i})}(\mathbb{R}^n) = \alpha_{(1^i 2^{d-i})} n^d$ entries to represent. The total number of FLOPs over symmetric tensors can thus be multiplied by the adjustment factor

$$\beta_{d,n} := \frac{1}{d} \sum_{i=1}^{d} \alpha_{(1^i 2^{d-i}),n} = \frac{1}{dn^d} \sum_{i=1}^{d} \binom{n+i-1}{i} \binom{n+d-i-1}{d-i}.$$

As $n \to \infty$, the adjustment factors approach

$$\alpha_{\pi,n} \to \alpha_{\pi,\infty} := \prod_{i=1}^{d} \frac{1}{c_i!}$$

$$\beta_{d,n} \to \beta_{d,\infty} := \frac{1}{d} \sum_{i=1}^{d} \frac{1}{i!(d-i)!}.$$

For reference,

$$\beta_{1,\infty} = 1, \quad \beta_{2,\infty} = \frac{3}{4}, \quad \beta_{3,\infty} = \frac{7}{18}, \quad \beta_{4,\infty} = \frac{5}{32}.$$

Note that there may be algorithms for performing symmetric tensor contractions even more efficiently than this [Solomonik and Demmel, 2015], but we disregard these for the sake of fairness and simplicity.

Polynomial fit: Because the FLOP count for cumulant propagation through a depth L MLP is a polynomial in n and L with leading term $n^{K+1}L$ for the unfactored version (and $n^K L^2$ for factored), we can simply fit a polynomial of the specified degree to the FLOP counts for a few small values of n, L to find the exact counts for any values. We check that these fit polynomials are exact on held out values of n, L. See Table 1 for the leading order terms of the FLOP polynomials.

[14] This is a practical weakness of using the Hermite expansion for the nonlinearity which we believe can be corrected in future work.

K	Basic	Augmented	Factorized	Factorized augmented
1	$4n^2L + 66nL + \ldots$			
2	$7n^3L + 26n^2L + \ldots$	$11n^3L + 150n^2L + \ldots$		
3	$\frac{7}{3}n^4L + 252n^3L + \ldots$	$\frac{7}{3}n^4L + 260n^3L + \ldots$	$30n^3L^2 + 39n^3L + \ldots$	$72n^3L^2 + 264n^3L + \ldots$
4	$\frac{5}{4}n^5L + 224n^4L + \ldots$	$\frac{5}{4}n^5L + 3083n^4L + \ldots$	$24n^4L^2 + \frac{79}{3}n^4L + \ldots$	$60n^4L^2 + 1069n^4L + \ldots$

Table 1: Cumulant propagation FLOP counts expressed as polynomials for a ReLU MLP with width n and L hidden layers. All 4 versions are identical for $K = 1$, and the factorized versions are identical to their unfactorized counterparts for $K = 2$. Fractional coefficients arise essentially because the number of distinct entries of a symmetric tensor is a polynomial with fractional coefficients (e.g. $\frac{1}{2}n^2 + \frac{1}{2}n$ for a symmetric matrix).

S Technical supplement

Cumulant propagation algorithms and analysis

Contents

S.1 Introduction **28**

S.2 Polynomial algorithm **33**

S.3 Mathematical preliminaries for Hermite-based algorithms **42**

S.4 Descriptions of Hermite-based algorithms **46**

S.5 Proofs for Hermite-based algorithms with polynomial activation functions **52**

S.6 Discussion of Hermite-based algorithms with non-polynomial activation functions **56**

S.7 Proof of diagram summation formulas **57**

S.8 Proofs of preliminary results **60**

S.1 Introduction

In this technical supplement, we provide precise descriptions of all the algorithms described in Section 4, and prove all of our theoretical results: the results of Section 5 on the performance of our algorithms, and the Hermite-based diagram summation formula for cumulants given in Appendix A.

To begin with, recall the main definition and theorems of the paper:

Definition 2.1. The *multilayer perceptron* (MLP) with $L \geq 0$ hidden layers, width $n \geq 1$, activation functions $\phi_1, \ldots, \phi_L : \mathbb{R} \to \mathbb{R}$ and weights $\boldsymbol{\theta} = \left(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L+1)}\right) \in \mathbb{R}^{(L+1)\times n\times n}$ is the function $M_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$M_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{W}^{(L+1)}\phi_L\Big(\mathbf{W}^{(L)} \ldots \phi_2\Big(\mathbf{W}^{(2)}\phi_1\Big(\mathbf{W}^{(1)}\mathbf{x}\Big)\Big)\ldots\Big),$$

where the activation functions are applied coordinatewise.

Theorem 5.1. *Fix an integer $L \geq 0$ and polynomial activation functions $\phi_1, \ldots, \phi_L$. Given weights $\boldsymbol{\theta} \in \mathbb{R}^{(L+1)\times n\times n}$ drawn independently from $\mathcal{N}\left(0, \frac{2}{n}\right)$ for an integer $n \geq 1$, write $M_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ for the MLP with L hidden layers, width n, activation functions $\phi_1, \ldots, \phi_L$ and weights $\boldsymbol{\theta}$. Then there is an estimation procedure $\mathbb{G}\left(\boldsymbol{\theta}, \varepsilon\right)$ such that for every noticeable[15] function $0 < \varepsilon(n) \leq 1$ and every output index i, as $n \to \infty$,*

$$\mathbb{E}_{\boldsymbol{\theta}}\Big[\left(\mathbb{G}(\boldsymbol{\theta}, \varepsilon(n))_i - \mathbb{E}_{X\sim\mathcal{N}(0,\mathbf{I}_n)}\left[M_{\boldsymbol{\theta}}\left(X\right)\right]_i\right)^2\Big] = O\Big(\varepsilon(n)^2\Big)$$

and $\mathbb{G}(\boldsymbol{\theta}, \varepsilon)$ runs in time $O\Big(\frac{n^2}{\varepsilon^2}\Big)$.

Theorem 5.2. *In Theorem 5.1, if we also require that $\varepsilon(n)^2 = O\big(\frac{1}{n^2}\big)$, then $\mathbb{G}\left(\boldsymbol{\theta}, \varepsilon\right)$ can be made to run in time $O\big(\frac{n}{\varepsilon^2}\big)$.*

More specifically, we prove 5.1 by taking the estimation procedure to be the basic cumulant propagation algorithm described in Section 4, and Theorem 5.2 by taking it to be the factorized version described in Section 4.3.

[15] This means that for some $k \geq 0$, $\varepsilon(n) \geq \frac{1}{n^k}$ for all sufficiently large n [Goldreich, 2005, Chapter 1.2].

In order to better convey the core ideas behind these proofs, we introduce an additional, simplified algorithm that avoids the use of Hermite expansions, which we refer to as the *polynomial algorithm*. This algorithm is sufficient to prove Theorem 5.1 (and its factorized version for Theorem 5.2), but it is of limited practical interest, because it can only be applied to polynomial activation functions. By contrast, the *Hermite-based* algorithms given in Section 4 can be applied to any measurable, polynomially bounded activation functions, although analytic difficulties prevent us from straightforwardly extending our proofs to this wider class of functions. The supplement begins by analyzing the polynomial algorithm, and then describes the changes necessary to analyze the Hermite-based algorithms.

We will now give a motivating overview of the algorithms and proofs, which should serve as a guide to the rest of the supplement.

S.1.1 Basic structure of cumulant propagation

The first intuition is simple: to estimate the mean output of an MLP, we will track estimates for the distribution (or statistics of the distribution) of (pre-)activations at each layer, starting at the input distribution (which we assume is Gaussian), and alternately passing through linear layers and non-linear activation layers, to produce an estimate for the distribution of the final layer. Define the following notation, which will serve to describe the MLP layer-by-layer:

- For each $\ell \in [L+1]$, let $\boldsymbol{\theta}^{(\ell)} = (\mathbf{W}^{(1)}, ..., \mathbf{W}^{(\ell)})$, so $\boldsymbol{\theta} = \boldsymbol{\theta}^{(L+1)}$.
- Let $X = X^{(0)} \sim \mathcal{N}(0, \mathbf{I}_n)$ denote the input to the MLP. Then recursively define

$$\begin{aligned} \forall 1 \le \ell \le L+1, \qquad & Z^{(\ell)} := \mathbf{W}^{(\ell)} X^{(\ell-1)} \\ \forall 1 \le \ell \le L, \qquad & X^{(\ell)} = \phi_\ell(Z^{(\ell)}). \end{aligned}$$

  Then $Z^{(\ell)}$ and $X^{(\ell)}$ are the preactivation and activation at layer ℓ, respectively.

While our estimation procedure applies to any specific set of weights, Theorem 5.1 only requires the estimate to be good *in expectation over random weights and in the limit* $n \to \infty$. Therefore, we can imagine $\mathbf{W}^{(\ell)}$ to be a large, unstructured matrix. By the central limit theorem, the relation $Z^{(\ell)} = \mathbf{W}^{(\ell)} X^{(\ell-1)}$ then suggests that regardless of the distribution of $X^{(\ell-1)}$, we expect the distribution of $Z^{(\ell)}$ to be close to Gaussian. Since distributions that are close to Gaussian are well-specified by their low-degree cumulants, we are led to the idea of tracking the cumulants through the layers.

The intuition that higher cumulants decay as $n \to \infty$ is proven in Corollary S.2.14:

$$\mathbb{E}\left[\left|\kappa_r[Z^{(\ell)}]_{i_1,\ldots,i_r}\right|^2\right] = \begin{cases} O(n^{2-r}) & \text{if } r \text{ is even, and each value in } i_1, \ldots, i_r \\ & \text{appears with even multiplicity,} \\ O(n^{1-r}) & \text{otherwise.} \end{cases} \tag{2}$$

This heuristically suggests that for integers $K \ge 1$, to get mean squared error (MSE) $O(n^{-K})$ we can do the following:

- If K is even, estimate all cumulants of order at most K;
- If K is odd, estimate all cumulants of order at most K *and* the elements $\kappa_{K+1}[Z^{(\ell)}]_{i_1,i_1,\ldots,i_{\frac{K+1}{2}},i_{\frac{K+1}{2}}}$. (As a more efficient bookkeeping device, we track these terms via the scalar harmonic component of κ_{K+1} – see Section S.2.2).

This is indeed what we do in a class of algorithms we call *cumulant propagation* (so K is not literally the maximum cumulant degree propagated, but it serves the same moral purpose). These algorithms estimate the cumulants listed above at each preactivation and activation layer.

In our main results, our algorithm is given only the tolerance parameter ε. In this case, for each n, we take K to be the smallest integer such that $\frac{1}{n^K} \le \varepsilon^2$ (note that we could have chosen any multiple of $\frac{1}{n^K}$ arbitrarily). This choice of K ensures that the mean squared error is $O(\varepsilon^2)$. To be fast enough,

we require the time taken to be $O(n^{K+1})$.[16] Therefore, to prove the validity of cumulant propagation, we wish to prove that for each integer $K \geq 1$,

1. Tracking the cumulants listed above does indeed produce MSE $O(n^{-K})$. This is our main result, Theorem S.2.15.
2. Tracking these cumulants takes time $O(n^{K+1})$. This is easily verified during the description of the algorithm in Sections S.2.6 and S.4.

Note that for this analysis to hold, we require that K is bounded above by a constant. This is exactly where the requirement that $\varepsilon(n)$ is noticeable comes from.

Let us now look at each stage of the algorithms in more detail.

S.1.2 The linear step

Tracking cumulants through the linear step is simple but expensive: cumulants are multi-linear, so the cumulants of degree r transform as

$$\kappa_r[Z^{(\ell)}]_{i_1,\ldots,i_r} = \sum_{j_1,\ldots,j_r=1}^{n} \kappa_r[X^{(\ell-1)}]_{j_1,\ldots,j_r} \prod_{s=1}^{r} \mathbf{W}^{(\ell)}_{i_s,j_s}$$

which takes $O(n^{r+1})$-time, and so can be done for cumulants up to degree K. We call this *symmetric contraction* (see Definition S.2.3). For harmonic components of cumulants of degree $> K$, we must figure out how to propagate the harmonic component itself rather than the whole tensor, but this is relatively routine.

S.1.3 The non-linear step for the polynomial algorithm

The more challenging part of the problem is how to propagate the cumulants through the (possibly non-linear) activation function, i.e. $X^{(\ell)} = \phi_\ell(Z^{(\ell)})$. The polynomial algorithm deals with this non-linearity in the following way: since ϕ_ℓ is a polynomial, the cumulant $\kappa_r[X^{(\ell)}]_{i_1,\ldots,i_r}$ can be expanded out by multi-linearity to a linear combination of cumulants of the form

$$\kappa_r[(Z^{(\ell)}_{i_1})^{k_1}, \ldots, (Z^{(\ell)}_{i_r})^{k_r}]$$

A simple combinatorial argument shows that a cumulant of powers can be written as a polynomial in the cumulants of the underlying random variables (see Lemma S.7.1), in particular

$$\kappa_r[(Z^{(\ell)}_{i_1})^{k_1}, \ldots, (Z^{(\ell)}_{i_r})^{k_r}] = \sum_{\substack{\pi \vdash [k_1] \sqcup \ldots \sqcup [k_r] \\ \text{connected}}} \prod_{S \in \pi} \kappa\left[Z^{(\ell)}_{i_v} : (v,l) \in S\right]$$

where π is a partition (see Section S.2 for the precise definition of the notation). Combining this with the multi-linearity proves that, if $\phi(t) = \sum_{k=0}^{D} \frac{1}{k!} a_k t^k$ is a polynomial,

$$\kappa_r[\phi(Z)]_{i_1,\ldots,i_r} = \sum_{\underline{k} \in \{0,\ldots,D\}^r} \frac{1}{\underline{k}!} \left(\prod_{v=1}^{r} a_{k_v}\right) \sum_{\substack{\pi \vdash [k_1] \sqcup \ldots \sqcup [k_r] \\ \text{connected}}} \prod_{S \in \pi} \kappa[Z_{i_v} : (v,l) \in S]. \tag{3}$$

This is the simplest form of what we call a *diagram summation formula*, in particular this is the polynomial diagram summation formula (see Theorem S.2.1). This diagram summation formula allows us to propagate our estimates for (the harmonic components of) the cumulants through the non-linearity.

S.1.4 The non-linear step for Hermite-based algorithms

It is clear that the algorithm just discussed has a number of flaws:

[16] One might have thought that $O(n^{K+2})$ would be sufficient to achieve a time of $O(\frac{n^2}{\varepsilon^2})$, but this is not the case because K is constrained to be an integer, while ε^2 could be $\Theta(n^{-k})$ for some non-integer k.

- It can only be run for polynomial ϕ_ℓ; and,
- Even for polynomial ϕ_ℓ, the constants in the runtime depend on the degree of the polynomial. A more favorable algorithm might prioritize the larger terms in (3) so that the number of terms computed depends on K but not D.

Let us now consider how to alter the algorithm to fix these two issues. A first attempt could go as follows: expand the activation function as a Taylor series around $x = 0$, apply multilinearity and Lemma S.7.1 to get the same diagram summation formula above (equation (3)) as a formal expansion with infinite summation, and truncate this expression after some finite number of terms.

Now recall equation (2). This tells us that the means and variances of Z are of constant size as $n \to \infty$ (technically we only give an upper bound, but it is, in fact, tight). Therefore, for any diagram (i.e. value of $\underline{k}$ and π) in (3), we can produce an infinite collection of diagrams with the same growth in n by increasing $\underline{k}$ by $\underline{d}$ and partitioning $\underline{d}$ with means and variances. So, any finite truncation will surely fail.

A partial solution to this is to arrange the sum on a "base diagram" of size $\underline{k}$ which contains no part of size 1 or 2 in a single $[k_v]$ (we call such a diagram $[\leq 2]$-mixed), and a diagram of size $\underline{d}$ which contains the remaining parts (and does not affect the connectedness of the complete diagram). We need an extra binomial factor to account for the choice of which shape-$\underline{k}$ subset of $\underline{k} + \underline{d}$ is the base diagram.

$$\kappa_r[\phi(Z)]_{i_1,\ldots,i_r} = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\sum_{\substack{\pi \vdash [k_1] \sqcup \ldots \sqcup [k_r] \\ \text{connected} \\ [\leq 2]\text{-mixed}}} \prod_{S \in \pi} \kappa[Z_{i_v} : (v,l) \in S] \right) \prod_{v=1}^{r} \left(\sum_{d=0}^{\infty} \binom{k_v + d}{k_v} \frac{a_{k_v+d}}{(k_v+d)!} \sum_{\substack{\tau \vdash [d] \\ \text{all parts size} \leq 2}} \prod_{S \in \tau} \kappa_{|S|}[Z_{i_v}] \right) \quad (4)$$

We can now simplify the second term: by the defining property of cumulants, the sum over τ is equal to $\mathbb{E}[(Y_{i_v})^d]$, where Y_{i_v} is the 1d Gaussian with the same mean and variance as Z_{i_v}. This means that the sum over d evaluates to $\frac{1}{k_v!}\mathbb{E}\left[\phi^{(k_v)}(Y_{i_v})\right]$. This expectation is exactly the k_v-th Hermite coefficient of ϕ with respect to Y_{i_v}, which we notate $\widehat{\phi}_{k_v}^{Y_{i_v}}$ (for the definition, see Section 3.2)! Therefore, we have shown

$$\kappa_r[\phi(Z)]_{i_1,\ldots,i_r} = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \frac{1}{\underline{k}!} \left(\prod_{v=1}^{r} \widehat{\phi}_{k_v}^{Y_{i_v}} \right) \left(\sum_{\substack{\pi \vdash [k_1] \sqcup \ldots \sqcup [k_r] \\ \text{connected} \\ [\leq 2]\text{-mixed}}} \prod_{S \in \pi} \kappa[Z_{i_v} : (v,l) \in S] \right) \quad (5)$$

This is the Hermite-based diagram summation formula, Theorem A.2. This is significant progress, since the Hermite coefficients may often be computed exactly (or at least approximated efficiently). In fact, if $i_1, \ldots, i_r$ are distinct, we have collected together all of the $\Theta(1)$-growth terms, and this is the expansion that we truncate in our Hermite-based algorithms.

However, there is still a further adjustment to make: notice that when $i_1 = i_2$ in Equation (5), the "base diagram" part may still contain $\mathrm{Var}[Z_{i_1}]$. For example, the equation simplifies in the case $r = 2, i_1 = i_2 = i$, and $Z = \mathcal{N}(0, I_n)$ to

$$\kappa_2[\phi(Z)]_{i,i} = \sum_{k \geq 1} \frac{1}{k!} \left(\widehat{\phi}_k^Z \right)^2 \mathrm{Var}[Z_i]^k.$$

So once again, when some indices are equal, any truncation of (5) will fail. The fix here can be thought of as one of the following two equivalent observations:

1. We can explicitly sum this infinite collection of $\Theta(1)$ terms. By the orthogonality property for Hermite polynomials,

$$\widehat{\phi^2}_0^Z = \mathbb{E}[\phi^2] = \mathbb{E}\left[\sum_{k_1,k_2=0}^{\infty} \frac{\sigma^{k_1+k_2}}{k_1!k_2!}\widehat{\phi}_{k_1}^Z \widehat{\phi}_{k_2}^Z \mathrm{He}_{k_1}\left(\sigma^{-1}Z\right)\mathrm{He}_{k_2}\left(\sigma^{-1}Z\right)\right] = \sum_{k=0}^{\infty}\frac{1}{k!}\left(\widehat{\phi}_k^Z\right)^2\sigma^{2k}$$

and so

$$\kappa_2[\phi(Z)]_{i,i} = \widehat{\phi^2}_0^Z - (\widehat{\phi}_0^Z)^2.$$

We can always do this summation to remove the $\Theta(1)$ terms in (5) when indices are repeated.[17]

2. Alternatively, by writing the cumulant in terms of expectations and collecting equal indices, we can write any cumulant as a polynomial in cumulants of powers with pairwise distinct indices. For example,

$$\begin{aligned}\kappa_2[\phi(Z)]_{i,i} &= \kappa_1[\phi^2(Z)]_i - \kappa_1[\phi(Z)]_i^2\\ \kappa_3[\phi(Z)]_{i,i,j} &= \kappa_2[\phi^2(Z),\phi(Z)]_{i,j} - 2\kappa_2[\phi(Z)]_{i,j}\kappa_1[\phi(Z)]_i.\end{aligned}$$

We call this construction *power cumulants* (see Section S.3.3).

These power cumulants can now be expanded by the same expansion as (5) with the Hermite coefficients of powers of ϕ. All $\Theta(1)$-size cumulants in this expression have now been absorbed into the Hermite coefficients.

These two bookkeeping devices (Hermite coefficients and power cumulants) turn out to be sufficient to deal with both flaws of the polynomial algorithm mentioned at the start of this section. So, to summarize, the outline of the non-linear step for the Hermite-based algorithms is as follows:

- First use the combinatorics of power cumulants to rewrite $\kappa_r[\phi_\ell(Z^{(\ell)})]_{i_1,\ldots,i_r}$ using cumulants of powers of $\phi_\ell(Z^{(\ell)})$ where all indices are distinct.
- For a power cumulant with distinct indices, use the corresponding expansion similar to equation (5).
- Depending on the specific algorithm, truncate this expansion for power cumulants after some finite number of terms so that the tail is sufficiently small in the limit as $n \to \infty$.

Since we can compute Hermite coefficients of powers of ϕ in many cases beyond polynomial ϕ, this algorithm can be applied quite broadly. For example, see Section S.3.4 for the Hermite coefficients of powers of ReLU.

S.1.5 Proof outline

We now give an overview of the proof itself.

As is clear in Section S.1.1 (see equation (2)), it is crucial to understand the asymptotic growth of the true cumulants as $n \to \infty$. While Corollary S.2.14 gives the moral result, the real work is

[17] For a more involved example, consider $\kappa_3[\phi(Z)]_{i,i,j}$ with $i \neq j$ when Z is a mean zero multivariate Gaussian. The Hermite-based diagram summation formula can be factorized as

$$\kappa_3[\phi(Z)]_{i,i,j} = \sum_{k\geq 1}\frac{1}{k!}(\widehat{\phi}_k^Z)_j \mathrm{Cov}[Z]_{i,j}^k \left(\sum_{a\geq 0}\frac{1}{a!}\mathrm{Var}[Z]_i^a \sum_{\substack{0\leq b\leq k\\ a+b(k-b)\neq 0}}\binom{k}{b}(\widehat{\phi}_{a+b}^Z)_i(\widehat{\phi}_{k+a-b}^Z)_i\right)$$

Another application of the orthogonality of Hermite coefficients, and a formula for the Hermite expansion of a product of Hermite polynomials shows that the inner parentheses evaluate to $\left(\widehat{\phi^2}_k^Z\right)_i - 2\left(\widehat{\phi}_0^Z\right)_i\left(\widehat{\phi}_k^Z\right)_i$ and so

$$\begin{aligned}\kappa_3[\phi(Z)]_{i,i,j} &= \sum_{k\geq 1}\frac{1}{k!}\left(\widehat{\phi^2}_k^Z\right)_i(\widehat{\phi}_k^Z)_j\mathrm{Cov}[Z]_{i,j}^k - 2\left(\widehat{\phi}_0^Z\right)_i\sum_{k\geq 1}\left(\widehat{\phi}_k^Z\right)_i(\widehat{\phi}_k^Z)_j\mathrm{Cov}[Z]_{i,j}^k\\ &= \kappa_2[\phi(Z)^2,\phi(Z)]_{i,j} - 2\kappa_1[\phi(Z)]\kappa_2[\phi(Z)]_{i,j}\end{aligned}$$

matching our second interpretation of power cumulants.

done by Proposition S.2.12, which states that any connected scalar contraction of cumulants grows linearly in n as $n \to \infty$. This proceeds via layer-by-layer induction, and is independent of the algorithm, simply depending on the choice of activation function. See Section S.6 for a discussion in the non-polynomial setting.

Next, we compute bounds for the expected sizes of the error cumulants (the difference between our estimated cumulants and the true cumulants). For these bounds, we simply bound the square value of any entry of an error cumulant by n^{-K}. This proof of this result is essentially a simple induction:

- The error tensors of $Z^{(\ell+1)}$ are easily computed in terms of contractions of the error tensors of $X^{(\ell)}$.
- To compute the sizes of these contractions, we use the relevant diagram summation formula (Theorem S.2.1 or A.2) and prove that the terms we drop in the truncation are all sufficiently small (we only drop terms when using the Hermite-based diagram summation formula). These terms include both the error cumulants and true cumulants of $Z^{(\ell)}$, and we use Cauchy-Schwartz to separate the two, bounding the two resulting diagrams individually.

Applying the error bound to the final first-order cumulant gives the required result. See Theorem S.2.15 for the proof for the polynomial algorithm, Section S.5 for the adjustments required for the Hermite-based algorithms, and Section S.6 for a discussion of the non-polynomial setting.

S.1.6 Supplement structure

The supplement is structured as follows.

- In Section S.2, we prove Theorem 5.1 using the polynomial algorithm. We also cover the prerequisites for the proof, including the polynomial diagram summation formula (Theorem S.2.1), harmonic decompositions of symmetric tensors (Theorem S.2.4), and asymptotic growth of tensors and tensor diagrams (Sections S.2.3 and S.2.4).
- In Section S.3, we state some preliminary results required for the proofs of the Hermite-based algorithms. These cover some combinatorial results required for working with cumulants and diagrams, as well as explicit computation of Hermite coefficients.
- In Section S.4, we give a full mathematical description of our Hermite-based algorithms.
- In Section S.5, we give a detailed sketch of the alterations to the proof in Section S.2 in order to prove Theorem 5.1 using the Hermite-based algorithms, still in the case of polynomial activation functions.
- In Section S.6, we discuss the remaining analytic results that we believe would be sufficient to adapt the proof in Section S.5 to apply to non-polynomial activation functions.
- In Section S.7, we prove both the polynomial and Hermite-based versions of the diagram summation formula for cumulants.
- In Section S.8, we prove all of remaining the preliminary results of Sections S.2 and S.3.

S.2 Polynomial algorithm

In this Section we will prove Theorem 5.1, the main theoretical result of this paper, in the case that the algorithm used is the polynomial algorithm (note that as per Section S.4, this algorithm does not agree with the basic algorithm, even in the case that the activation functions are polynomials).

Write $\mathcal{T}^r(\mathbb{R}^n)$ for the space of all *r-tensors* on n real variables (which is an $\mathbb{R}$-inner product space with the Frobenius inner product). Write $\mathcal{S}^r(\mathbb{R}^n)$ for the space of all *symmetric* r-tensors on n real variables.

A *partition* π of $[r]$, denoted $\pi \vdash [r]$, is a set of disjoint subsets (*blocks*) of $[r]$ with union equal to $[r]$. The set of partitions $\{\pi : \pi \vdash [r]\}$ forms a lattice with the following notation:

- Write $\pi \leq \tau$ if $\forall S \in \pi, \exists T \in \tau$ such that $S \subseteq T$.
- For $\pi, \tau \vdash [r]$, write $\pi \wedge \tau$ for the greatest lower bound (called the *meet*) of π and τ, i.e.

$$\pi \wedge \tau = \{S \cap T : S \in \pi, T \in \tau\}$$

- For $\pi, \tau \vdash [r]$, write $\pi \vee \tau$ for the least upper bound (called the *join*) of π and τ.

S.2.1 Polynomial diagram summation formula

The polynomial algorithm uses a significantly simpler version of Theorem A.2 that avoids the need to Hermite coefficients, at the expense of us not being able to truncate the formula (therefore introducing a dependence of the algorithm on the degree of the polynomial). For polynomials, this is not an issue for the theoretical result since both diagram summation formulas have only a finite number of terms, however when we discuss non-polynomial activations we will see that truncating the Hermite-based diagram summation formula is a crucial step.

Theorem S.2.1 (Polynomial diagram summation formula for cumulants)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi : \mathbb{R} \to \mathbb{R}$ be a polynomial. Then*

$$\kappa\left[\phi\left(Z_{j_1}\right), \ldots, \phi\left(Z_{j_r}\right)\right] = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^{r} \frac{1}{k_v!} \phi^{(k_v)}(0)\right) \sum_{\pi \in \mathrm{cDia}(\underline{k})} \prod_{S \in \pi} \kappa\left[Z_{j_v} : (v, w) \in S\right].$$

This is proven in Section S.7. See Definition A.1 for the definition of $\mathrm{cDia}(\underline{k})$.

S.2.2 Harmonic decomposition of symmetric tensors

Definition S.2.2. The *trace* $\mathrm{tr}\colon \mathcal{S}^r(\mathbb{R}^n) \to \mathcal{S}^{r-2}(\mathbb{R}^n)$ is defined by

$$(\mathrm{tr}\,\mathbf{T})_{i_1,\ldots,i_{r-2}} = \sum_{i=1}^{n} \mathbf{T}_{i,i,i_1,\ldots,i_{r-2}}.$$

for $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$. Also, for $r \in \{0, 1\}$, define $\mathrm{tr} : \mathcal{S}^r(\mathbb{R}^n) \to \{0\}$ to be the zero map. The *cup* $g\colon \mathcal{S}^r(\mathbb{R}^n) \to \mathcal{S}^{r+2}(\mathbb{R}^n)$ is defined by

$$g(\mathbf{T})_{i_1,\ldots,i_{r+2}} = \sum_{\substack{u,v \in [r+2] \\ u<v}} \delta_{i_u,i_v} \mathbf{T}_{\{i_s : s \notin \{u,v\}\}},$$

where δ is the Kronecker delta. More generally, for $\mathbf{M} \in \mathcal{S}^2(\mathbb{R}^n)$, the $\mathbf{M}$-*cup* $g_{\mathbf{M}}\colon \mathcal{S}^r(\mathbb{R}^n) \to \mathcal{S}^{r+2}(\mathbb{R}^n)$ is defined by

$$g_{\mathbf{M}}(\mathbf{T})_{i_1,\ldots,i_{r+2}} = \sum_{\substack{u,v \in [r+2] \\ u<v}} \mathbf{M}_{i_u,i_v} \mathbf{T}_{\{i_s : s \notin \{u,v\}\}}.$$

Definition S.2.3. For a rank-r tensor $\mathbf{T}$ and a matrix $\mathbf{W}$ define the *symmetric contraction of* $\mathbf{T}$ *with* $\mathbf{W}$ to be the rank-r tensor

$$(\mathbf{T} \odot \mathbf{W})_{i_1,\ldots,i_r} := \sum_{j_1,\ldots,j_r=1}^{n} \mathbf{T}_{j_1,\ldots,j_r} \prod_{s=1}^{r} \mathbf{W}_{i_s,j_s}$$

This operation can be computed in $O(n^{r+1})$ operations by multiplying in the matrix one index at a time.

Given a symmetric rank-r tensor $\mathbf{T}$, the harmonic decomposition expresses it as a sum of a "generic" part and a lower rank object (which then recursively breaks down into still smaller objects).

Theorem S.2.4 (Harmonic decomposition of symmetric tensors)**.** *Let $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$. Then there exist unique $h_{r-2s}(\mathbf{T}) \in \mathcal{S}^{r-2s}(\mathbb{R}^n)$ for $0 \leq s \leq \left\lfloor \frac{r}{2} \right\rfloor$ satisfying*

$$\mathrm{tr}\left(h_{r-2s}(\mathbf{T})\right) = 0, \qquad \forall 0 \leq s \leq \left\lfloor \frac{r}{2} \right\rfloor - 1$$

$$\mathbf{T} = \sum_{0 \leq s \leq \lfloor \frac{r}{2} \rfloor} g^s(h_{r-2s}(\mathbf{T}))$$

In fact,

$$h_{r-2s}(\mathbf{T}) = \sum_{0 \leq t \leq \lfloor \frac{r-2s}{2} \rfloor} c^{\mathrm{h}}_{s,t} g^t(\mathrm{tr}^{s+t}(\mathbf{T})) \tag{6}$$

where

$$c^{\mathrm{h}}_{s,t} = (-1)^t \frac{r + \frac{n}{2} - 2s - 1}{2^{s+t} s! t! (r + \frac{n}{2} - 2s - t - 1)_{s+t+1}}$$

with $(a)_t := a(a+1)...(a+t-1)$ *(the Pochhammer symbol).*

This is proven in Section S.8.1. For convenience, we notate

$$h_{\leq r-2s}(\mathbf{T}) := \sum_{s'=s}^{\lfloor r/2 \rfloor} g^{s'-s}(h_{r-2s'}(\mathbf{T})) \in \mathcal{S}^{r-2s}(\mathbb{R}^n)$$

$$h_{\geq r-2s}(\mathbf{T}) := \sum_{s'=0}^{s} g^{s'}(h_{r-2s'}(\mathbf{T})) \in \mathcal{S}^{r}(\mathbb{R}^n)$$

To see why this is referred to as the "harmonic" decomposition, treat $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$ as the tensor of coefficients of a degree-r homogeneous polynomial on $\mathbb{R}^n$. (I.e., identify $\mathcal{S}^r(\mathbb{R}^n)$ with $\mathcal{S}^r(\mathbb{R}^{n*})$ via the standard inner product on $\mathbb{R}^n$.) Then the Laplacian operator is $r(r-1)\,\mathrm{tr}$, multiplication by squared radius is $\frac{2}{r(r-1)} g \colon \mathcal{S}^{r-2}(\mathbb{R}^n) \to \mathcal{S}^r(\mathbb{R}^n)$, and the harmonic decomposition of symmetric tensors coincides with the usual decomposition of polynomials into spherical harmonics (cf. Dai and Xu [2013, Theorem 1.1.3, Lemma 1.2.1]).

Another characterization of the harmonic decomposition is as the decomposition of $\mathcal{S}^r(\mathbb{R}^n)$ into irreducible representations of the action of $\mathrm{O}(n)$ via the symmetric contraction $\odot$ [Dai and Xu, 2013, Theorem 1.7.2]. Because the distribution of Gaussian-initialized $\mathbf{W}$ is $\mathrm{O}(n)$-symmetric, this is intuitively the right level of granularity to track cumulant tensors of activations, which directly precede the application of $- \odot \mathbf{W}$.

S.2.3 Tensor growth

Although our algorithm applies to a particular instance of an MLP, Theorem 5.1 refers to the error of these algorithms in expectation over the random weights and in the limit of large width. To prove these results rigorously, we must first define an appropriate notion of the growth of tensors in the limit of width $n \to \infty$.

Let $\Theta^{(\ell)}_n$ be the random variable valued in $\mathbb{R}^{\ell \times n \times n}$ given by the values of $\boldsymbol{\theta}^{(\ell)} = (\mathbf{W}^{(1)}, ..., \mathbf{W}^{(\ell)})$ when these weights are drawn independently from $\mathcal{N}(0, \frac{2}{n})$.

Definition S.2.5. Fix $r \geq 0$. We define a rank-r *tensor sequence* to be a sequence of random variables T_n taking values in $\mathcal{T}^r(\mathbb{R}^n)$. A tensor sequence is said to be *symmetric* if, for every $n \geq 1$, the support of T_n is contained in $\mathcal{S}^r(\mathbb{R}^n)$.

We say that a tensor sequence $(T_n)_n$ is *at layer* ℓ for some $\ell \in [L+1]$ if the random variables T_n are all measurable functions of $\Theta^{(\ell)}_n$.

For example, for each $r \geq 1$ and $\ell \in [L]$, the (pre-)activation cumulants ($\kappa_r[Z^{(\ell)}]$ and $\kappa_r[X^{(\ell)}]$ respectively) form symmetric tensor sequences at layer ℓ.

Definition S.2.6. We say the tensor sequence $T = (T_n)_{n=1}^{\infty}$ has n^ρ*-growth* (written suggestively as $T = O(n^\rho)$) if, for each $1 \leq p < \infty$,

$$\limsup_{n \geq 1} \max_{i_1, ..., i_r \in [n]} \|(T_n)_{i_1,...,i_r}\|_p n^{-\rho} < \infty$$

where $\|\cdot\|_p$ denotes the L_p-norm. Since the L_p-norms increase as p increases, it is sufficient to check this for even integers $p \in 2\mathbb{Z}_{\geq 1}$.

If $r = 0$, we simply have a sequence of 1d random variables X_n, and this has n^ρ-growth if for every $1 \leq p < \infty$, $(L_p(T_n))_{n \geq 1} = O(n^\rho)$ in the traditional sense of big-O notation.

S.2.4 Tensor diagrams

When computing the cumulants of $\phi(Z)$ in terms of the cumulants of Z, we will see tensor networks of cumulants. We formalize these using tensor sequences in the following way:

Definition S.2.7. We define a *tensor diagram* as a tuple $D = (V_1, V_2, E, I, \{T_v\}_{v \in V_1})$ where

1. $B_D := (V_1, V_2, E)$ is a finite bipartite (multi-)graph (i.e. we allow multiple edges),
2. $I \subset V_2$
3. For each $v \in V_1$, T_v is a symmetric tensor sequence with $\deg(v) = \mathrm{rank}(T_v)$.

Let $\mathrm{rank}(D) := |V_2 \setminus I|$. Call D *connected* if B_D is connected; call it *fully-contracted* if $\mathrm{rank}(D) = 0$. We say the diagram D is *at layer* ℓ if every tensor sequence is at layer ℓ.

This should be interpreted as an einsum in the following way: the vertices $v \in V_2$ correspond to indices, those in I are contracted indices, those not in I are "hanging" indices, and the tensor diagram corresponds to a tensor sequence where the contracted indices have been summed over, giving

$$T_{D,n} = \left(\sum_{\forall v \in I, i_v = 1}^{n} \prod_{u \in V_1} ((T_u)_n)_{i_v : (u,v) \in E)} \right)_{i_v : v \notin I} \tag{7}$$

Here we gloss over two details: first, note that in order to produce a truly well-defined tensor sequence, we need to pick a bijection between $V_2 \setminus I$ and $[\deg(D)]$; and second, for this to be a well-defined tensor sequence, we must correctly reconcile the underlying sources of randomness for the different tensor sequences. In all cases we consider, the random variables will have shared randomness.

Sometimes, we will refer to a tensor being written as a sum of diagrams, and we should emphasize here that this means an *element-wise sum* of tensors with equal rank, *not a disjoint sum of diagrams*. In particular, a disjoint sum of fully-contracted diagrams results in multiplication of the corresponding scalars, whereas a sum of fully-contracted diagrams results in a sum of the corresponding scalars.

We now look at the effect of moving from a tensor diagram at layer ℓ to one at layer $\ell + 1$ by contracting each tensor sequence by the transition matrix $\mathbf{W}^{(\ell+1)}$. For this purpose, note that we can extend the notation $- \odot \mathbf{W}^{(\ell+1)}$ of Definition S.2.3 from tensors to tensor sequences at layer ℓ in the obvious way, producing a tensor sequence at layer $\ell + 1$. For a tensor diagram D at layer ℓ, let $D \odot \mathbf{W}^{(\ell+1)}$ be the tensor diagram at layer $\ell + 1$ resulting from replacing each T_v with $T_n \odot \mathbf{W}^{(\ell+1)}$ for all $v \in V_1$ (note that this differs from computing D via an einsum as a rank-$|V_2 \setminus I|$ tensor sequence and applying $- \odot \mathbf{W}^{(\ell+1)}$).

Example S.2.8. The post-activation cumulants at layer ℓ and the pre-activation cumulants at layer $\ell + 1$ are related simply via

$$\kappa_d[Z^{(\ell+1)}] = \kappa_d[X^{(\ell)}] \odot \mathbf{W}^{(\ell+1)}.$$

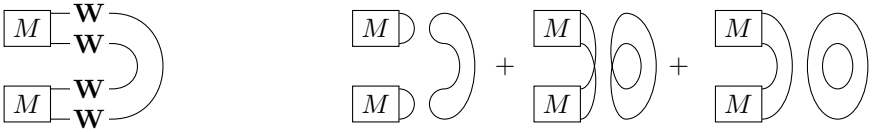
Proposition S.2.9 (Diagram expectations)**.** *Let $D = (V_1, V_2, E, I, \{T_v\}_{v \in V_1})$ be a rank r tensor diagram at layer ℓ. If $|E|$ is odd, then the rank r tensor sequence $\mathbb{E}_{\mathbf{W}^{(\ell+1)}}[D \odot \mathbf{W}^{(\ell+1)}]$ is identically zero. If $|E|$ is even,*

$$\mathbb{E}_{\mathbf{W}^{(\ell+1)}}[D \odot \mathbf{W}^{(\ell+1)}] = \left(\frac{2}{n}\right)^{|E|/2} \sum_{\pi \in \mathcal{P}_2(E)} D_{\pi,1} D_{\pi,2},$$

where $D_{\pi,1}$ is a rank 0 (scalar) tensor diagram at layer ℓ constructed as $(V_1, \pi, \pi, \{(v, S) : v \in e \in S \in \pi\}, \{T_v\}_{v \in V_1})$, and $D_{\pi,2}$ is a constant rank r tensor diagram (constant meaning that the random variables are all constant) constructed as $(\pi, V_2, I, \{(S, w) : w \in e \in S \in \pi\}, (I_2)_{S \in \pi})$. Here, $\mathcal{P}_2(E)$ denotes the set of perfect pairings of E. Furthermore, the number of connected components of $D_{\pi,1} D_{\pi,2}$ is at most $|E|/2$ more than the number of connected components of D.

Example S.2.10. Consider a symmetric $n \times n$-matrix M. Then $M \odot \mathbf{W} = \mathbf{W} M \mathbf{W}^\top$. Therefore $\|\mathbf{W} M \mathbf{W}^\top\|_F^2$ is a diagram of the form $D \odot \mathbf{W}$ for some fully contracted diagram D. There are 4 edges in D, so the right hand side of Proposition S.2.9 has 3 terms. These are most easily understood in diagrammatic form, as in Figure 12. In fact, we see that

$$\mathbb{E}_{\mathbf{W}}\left[\|\mathbf{W} M \mathbf{W}^\top\|_F^2\right] = \left(\frac{2}{n}\right)^2 \left(n \mathrm{Tr}(M)^2 + n\|M\|_F^2 + n^2 \|M\|_F^2\right)$$



(a) $\|\mathbf{W}M\mathbf{W}^\top\|_F^2$ (b) $n(\mathrm{Tr}(M))^2 + n\|M\|_F^2 + n^2\|M\|_F^2$

Figure 12: A diagrammatic proof of the identity $\mathbb{E}_{\mathbf{W}}\left[\|\mathbf{W}M\mathbf{W}^\top\|_F^2\right] = \left(\frac{2}{n}\right)^2\left(n\mathrm{Tr}(M)^2 + n\|M\|_F^2 + n^2\|M\|_F^2\right)$ for a symmetric $n \times n$ matrix M. Each loop in the right hand side diagrams contributes a factor of n. The diagrams on the right are obtained by replacing the four copies of $\mathbf{W}$ with each of the three possible pairings. Explicitly, we replace $\prod_{r=1}^4 \mathbf{W}_{i_r,j_r}$ with $\delta_{i_1,i_2}\delta_{i_3,i_4}\delta_{j_1,j_2}\delta_{j_3,j_4} + \delta_{i_1,i_3}\delta_{i_2,i_4}\delta_{j_1,j_3}\delta_{j_2,j_4} + \delta_{i_1,i_4}\delta_{i_2,i_3}\delta_{j_1,j_4}\delta_{j_2,j_3}$.

We also have a form of Cauchy-Schwartz for tensor diagrams given below. Both results are proven in Section S.8.2.

Lemma S.2.11 (Cauchy-Schwartz for tensor diagrams)**.** *Let $D = (V_1, v_2, E, I, \{T_v\})$ be a tensor diagram, and let $V_1 = U_1 \sqcup U_2$ be a partition of the left vertices of D into two parts. Consider the two tensor diagrams formed as follows: for each $i \in \{1, 2\}$ construct D_i by doubling the set U_i along with all its edges and tensors, formally*

- $V_1(D_i) = [2] \times U_i$
- $V_2(D_i) = V_2(D)$
- $E(D_i) = \{((j, v), w) : (v, w) \in E(D), v \in U_i\}$
- $I(D_i) = I(D)$
- $T_{(j,v)} = T_v$ *for all* $j = 1, 2, v \in U_i$.

Then all entries of D_1, D_2 are non-negative, and

$$D^2_{i_1,\ldots,i_r} \le (D_1)_{i_1,\ldots,i_r}(D_2)_{i_1,\ldots,i_r}.$$

In addition, if D_i have n^{ρ_i}-growth, then D has $n^{\frac{1}{2}(\rho_1+\rho_2)}$-growth.

S.2.5 Cumulant analysis for polynomial activation functions

In this section, we will discuss the cumulant tensors of the layers (both pre- and post-activation). As previously discussed, we are considering the cumulants for fixed weights – the distribution comes from the input $X^{(0)} \sim \mathcal{N}(0, I_n)$. The main result is the following Proposition. The discovery and proof of this Proposition was pivotal to our understanding of the proposed algorithms (both in the polynomial and Hermite-based settings), and so we provide the proof here rather than delaying it to Section S.8.

Proposition S.2.12 (Cumulants growth bounds)**.** *Suppose that D is a fully-contracted connected tensor diagram such that for some $\ell \in [L+1]$, either*

- *every tensor sequence for $v \in V_1$ is $\kappa_{\deg(v)}[X^{(\ell-1)}]$; or*
- *every tensor sequence for $v \in V_1$ is $\kappa_{\deg(v)}[Z^{(\ell)}]$.*

Then, D has n-growth.

Before proving this result, let us first recap some simple facts about the cumulants. For the 0-th layer, the cumulants are trivial:

$$\kappa_r[X^{(0)}] = \begin{cases} I_n, & \text{if } r = 2 \\ 0, & \text{o/w} \end{cases}$$

To compute the cumulants after applying the linear layer, we apply symmetric contraction of the weights matrix $\mathbf{W}^{(\ell)}$,

$$\kappa_r\left[Z^{(\ell)}\right] = \kappa_r\left[X^{(\ell-1)}\right] \odot \mathbf{W}^{(\ell)}$$

for $\ell \in [L+1]$. For the activation step, we have the following simple result.

Lemma S.2.13. *For any rank r, the tensor sequence*

$$\kappa_r\left[\phi^{(\ell)}(Z^{(\ell)})\right]$$

can be written as a finite sum of connected rank-r tensor diagrams involving only tensor sequences from the set $\left\{\kappa_{r'}\left[Z^{(\ell)}\right] : r' \geq 1\right\}$.

Proof. By multi-linearity of κ_r we get a finite sum of cumulants of monomials. Lemma S.7.1 then applies producing a sum of tensor diagrams of $\kappa_{r'}[Z^{(\ell)}]$'s. The identification of indices corresponds to the partition $[r_1] \cup ... \cup [r_d]$, in particular there are d indices, and $I = \phi$, so no indices have been contracted. The cumulant tensors correspond to the partition π. By assumption, these two partitions satisfy that their join is the trivial partition and therefore this tensor diagram is connected. □

We now prove Proposition S.2.12, by induction.

Proof. We prove this by induction, starting with the case that every tensor sequence in D is $\kappa_{\deg(v)}\left[X^{(0)}\right]$. Since only the second cumulant of $X^{(0)}$ is non-zero, and it is equal to I_2, we see that any two indices connected to the same cumulant must be equal, and therefore connectedness implies all indices must be equal, therefore

$$D_n = n.$$

We now prove that if the result holds for $Z^{(\ell)}$-cumulants, then it holds for $X^{(\ell)}$-cumulants. This follows from Lemma S.2.13, since if we expand the $X^{(\ell)}$-cumulants in terms of $Z^{(\ell)}$-cumulants, each is a sum (not disjoint union!) of connected tensor diagrams of $Z^{(\ell)}$-cumulants. Placing these on each node $v \in V_1(D)$ and then expanding the sum, we get a sum of fully-contracted connected tensor diagrams of the $Z^{(l)}$-cumulants. Consequently this has n-growth.

Finally, we prove that if the result holds for $X^{(\ell-1)}$-cumulants, then it holds for $Z^{(\ell)}$-cumulants. Consider the diagram D: taking the p-th power of this quantity corresponds to taking the disjoint sum of p copies of this diagram. By Proposition S.2.9, taking the expectation over $\mathbf{W}^{(\ell)}$ produces a sum

$$\left(\frac{2}{n}\right)^{p|E|/2} \sum_{\pi \text{ pairing of } pE(B)} D_{\pi,1} D_{\pi,2}$$

where the number of connected components of $D_{\pi,1}D_{\pi,2}$ is at most $p(1+\frac{1}{2}|E|)$. Each of the connected components of $D_{\pi,1}$ is a connected fully-contracted tensor diagram of cumulants of $X^{(\ell-1)}$ so contributes a factor of $O(n)$, and each of the connected component of $D_{\pi,2}$ is a connected fully-contracted tensor diagram of I_2's, which also contribute a factor of n. Therefore we get that the whole diagram is $O(n^p)$ as required. □

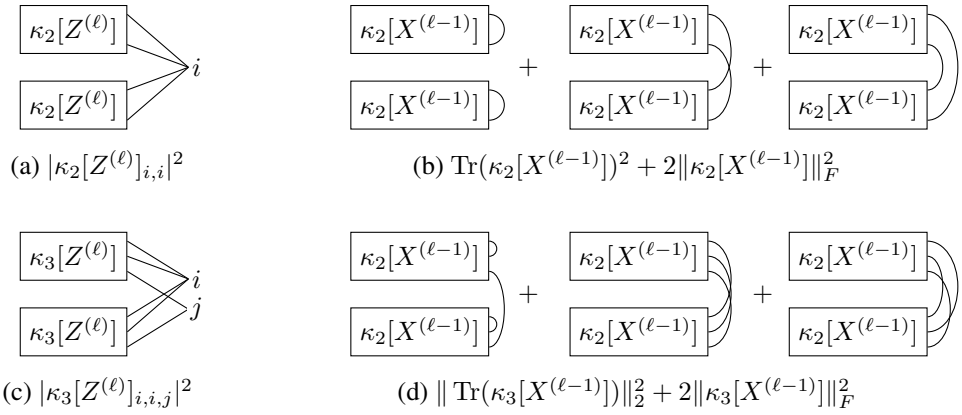
Corollary S.2.14. *Let $r \geq 1, \ell \in [L+1], n \geq 1$ and fix $i_1, ..., i_r \in [n]$. Then*

$$\mathbb{E}\left[\left|\kappa_r[Z^{(\ell)}]_{i_1,\ldots,i_r}\right|^2\right] = \begin{cases} O(n^{2-r}) & \text{if } r \text{ is even, and each value in } i_1,\ldots,i_r \\ & \text{appears with even multiplicity,} \\ O(n^{1-r}) & \text{otherwise.} \end{cases}$$

Proof. Consider $\mathbb{E}\left[\kappa_r[Z^{(\ell)}]^p_{i_1,\ldots,i_r}\right]$ for $p \geq 2$ even. Applying Proposition S.2.9, we get a prefactor of $n^{-\frac{1}{2}pr}$, multiplied by fully-contracted diagram produced by pairing p-copies of $\kappa_r[X^{(\ell)}]$. If r is odd, the maximal number of connected components is $\frac{1}{2}p$, giving $n^{\frac{1}{2}p(1-r)}$. If r is even, the maximal number of connected components is p, giving $n^{p(1-\frac{1}{2}r)}$.

If r is even and some index appears an odd number of times, the maximal number of connected components is again $\frac{1}{2}p$, giving $n^{\frac{1}{2}p(1-r)}$-growth. □

We demonstrate two instances of this Corollary in Figure 13.



(a) $|\kappa_2[Z^{(\ell)}]_{i,i}|^2$ (b) $\operatorname{Tr}(\kappa_2[X^{(\ell-1)}])^2 + 2\|\kappa_2[X^{(\ell-1)}]\|_F^2$

(c) $|\kappa_3[Z^{(\ell)}]_{i,i,j}|^2$ (d) $\|\operatorname{Tr}(\kappa_3[X^{(\ell-1)}])\|_2^2 + 2\|\kappa_3[X^{(\ell-1)}]\|_F^2$

Figure 13: Two square magnitudes of elements of $\kappa[Z^{(\ell)}]$ tensors (a) and (c). Since the first cumulant has an even number of each index, there is a partition splitting the two copies (the first diagram in (b)), and therefore the growth is $O(n^{-2}n^2) = O(1)$ (recall that we suppress by a factor of $\left(\frac{2}{n}\right)^{\#\text{indices}}$). By contrast, in the second diagram we cannot split the two copies (every diagram in (d) is connected), and so has $O(n^{-3}n) = O(n^{-2})$-growth.

S.2.6 Description of polynomial algorithm

For this section, we assume that the activation function is a polynomial $\phi(x) = \sum_{j=0}^{m} a_j x^j$.

The polynomial algorithm propagates tensors $\tilde{\eta}_r[X^{(\ell)}] \in \mathcal{S}^{r-2s_K(r)}(\mathbb{R}^n)$ through MLP layers. These are estimates for $h_{\le r-2s_K(r)}(\kappa_r[X^{(\ell)}])$, where

$$s_K(r) := \begin{cases} 0 & r \le K \\ \dfrac{r}{2} & r = K+1 \text{ and } K \equiv 1 \mod 2 \\ \infty & \text{otherwise.} \end{cases}$$

By convention, if $2s_K(r) > r$, then $\tilde{\eta}_r[X^{(\ell)}] = 0$. Thus, only cumulants with degree r no more than

$$R := \begin{cases} K & K \equiv 0 \mod 2 \\ K+1 & K \equiv 1 \mod 2 \end{cases}$$

are tracked at all.

The algorithm proceeds as follows:

1. For $r \in [R]$, initialize tensors

$$\tilde{\eta}_r[X^{(0)}] := h_{\le r-2s_K(r)}(\kappa_r[X^{(0)}]) = \begin{cases} 1 & r = 2 \text{ and } K = 1 \\ \mathbf{I}_n & r = 2 \text{ and } K > 1 \\ 0 & r \ne 2. \end{cases}$$

2. For $1 \le \ell \le L$, repeat the following:
   (a) Compute the tensors

$$\mathbf{T}_r^{(\ell)} := \tilde{\eta}_r[X^{(\ell-1)}] \odot \mathbf{W}^{(\ell)} \qquad \forall r \in [R]$$
$$\mathbf{M}_{i,j}^{(\ell)} := \begin{cases} (\mathbf{W}^{(\ell)}\mathbf{W}^{(\ell)\top})_{i,j} & K > 1 \text{ or } i = j \\ 0 & \text{otherwise.} \end{cases}$$

   Note that each $\mathbf{T}_r^{(\ell)}$ is a symmetric contraction of a $(r - 2s_K(r))$-tensor, and thus takes time $O(n^{r-2s_K(r)+1}) \le O(n^{K+1})$. The matrix $\mathbf{M}^{(\ell)}$ takes time $O(n^2)$ if $K = 1$ and otherwise $O(n^3)$. In any case, this is no more than the $O(n^{K+1})$ budget.

(b) Define $\tilde{\kappa}_r[Z^{(\ell)}] := g_{\mathbf{M}^{(\ell)}}^{s_K(r)} \mathbf{T}_r^{(\ell)}$. Compute estimated cumulants of X using Lemma S.7.1 and multilinearity:

$$\tilde{\kappa}_r[X^{(\ell)}]_{i_1,\ldots,i_r} = \sum_{\underline{k}\in[m]^r} \left(\prod_{j=1}^r a_{k_j}\right) \sum_{\pi \vdash \bigsqcup_j [k_j]} \prod_{S\in\pi} \tilde{\kappa}_{|S|}[Z^{(\ell)}]_{i_v:(v,l)\in S}, \qquad \forall r \in [R]$$

For $r > R$, define $\tilde{\kappa}_r[X^{(\ell)}] = 0$.

(c) Project $\tilde{\kappa}_r[X^{(\ell)}]$ onto $(\leq r - 2s_K)$-harmonics using Theorem S.2.4:

$$\tilde{\eta}_r[X^{(\ell)}] := \sum_{s'=s}^{\lfloor r/2 \rfloor} g^{s'-s}(h_{r-2s'}(\tilde{\kappa}_r[X^{(\ell)}])),$$

where $h_{r-2s'}$ is defined in (6). By the definition of s_K, this simplifies to

$$\tilde{\eta}_r[X^{(\ell)}] = \begin{cases} \tilde{\kappa}_r[X^{(\ell)}] & r \leq K \\ h_0(\tilde{\kappa}_r[X^{(\ell)}]) & r = K+1 \equiv 0 \mod 2 \\ 0 & \text{otherwise.} \end{cases}$$

3. Return $\mathbf{W}^{(L+1)}\tilde{\eta}_1[X^{(L)}]$.

S.2.7 Error analysis

In this section, we prove that the polynomial algorithm described above satisfies the conditions of Theorem 5.1. Define the error tensors

$$\Delta_r[X^{(\ell)}] := \kappa_r[X^{(\ell)}] - \tilde{\kappa}_r[X^{(\ell)}]$$
$$\Delta_r[Z^{(\ell)}] := \kappa_r[Z^{(\ell)}] - \tilde{\kappa}_r[Z^{(\ell)}]$$

These are tensor sequences. Our main result is the following theorem:

Theorem S.2.15. *For all $r \geq 1$ and $\ell \in [L+1]$, the tensor sequence $\Delta_r[Z^{(\ell)}]$ has $n^{-(K/2)}$-growth.*

First, we will prove a useful lemma.

Lemma S.2.16. *For any rank r, the tensor sequence*

$$\Delta_r[X^{(\ell)}]$$

can be written as a finite sum of connected rank-r tensor diagrams involving tensor sequences from $\left\{\kappa_{r'}[Z^{(\ell)}] : r' \geq 1\right\} \cup \left\{\Delta_{r'}\left[Z^{(\ell)}\right] : r' \geq 1\right\}$. Furthermore, each term must contain at least one tensor sequence of the form $\Delta_{r'}\left[Z^{(\ell)}\right]$.

Proof. By definition of $\Delta_r\left[X^{(\ell)}\right]$, we get directly that

$$\Delta_r[X^{(\ell)}]_{i_1,\ldots,i_r} = \sum_{\underline{k}\in[m]^r} \left(\prod_{j=1}^r a_{k_j}\right) \sum_{\pi} \left(\prod_{S\in\pi} \kappa\left[Z_{i_v}^{(\ell)} : (v,l) \in S\right] - \prod_{S\in\pi} \widetilde{\kappa}\left[Z_{i_v}^{(\ell)} : (v,l) \in S\right]\right)$$

Consider the difference inside the π summation: since $\widetilde{\kappa} = \kappa - \Delta$, we can use the binomial theorem to rewrite this as

$$\sum_{\pi' \subsetneq \pi} (-1)^{|\pi|-|\pi'|} \left(\prod_{S\in\pi'} \kappa\left[Z_{i_v}^{(\ell)} : (v,l) \in S\right]\right) \left(\prod_{S\notin\pi'} \Delta\left[Z_{i_v}^{(\ell)} : (v,l) \in S\right]\right).$$

For each π', the quantity inside is (the $i_1, ..., i_r$-entry of) a tensor diagram formed as follows:

- $V_1 = \pi, V_2 = [r]$, the edges consist of (S, v) such that $(v, l) \in S$ for some l, with multiplicity.
- $I = \emptyset$

- $T_v = \kappa_{\deg(v)}[Z^{(\ell)}]$ if $v \in \pi'$, and $T_v = \Delta_{\deg(v)}[X^{(\ell)}]$ otherwise.

The number of diagrams, and their coefficients, are independent of n. This completes the proof of the Lemma. □

Proof of Theorem S.2.15. We will do this by induction on ℓ. By defining $\tilde{\kappa}_r[Z^{(0)}] := \tilde{\kappa}_r[X^{(0)}]$, and $\phi^{(0)} := \mathrm{id}$, we may start the induction by proving the result for $\Delta_r[Z^{(0)}]$, which is clear since at $\ell = 0$ all the error tensors are identically zero.

For the inductive step, assume that $\Delta_r[Z^{(\ell)}]$ has $n^{-K/2}$-growth for some $\ell \in 0, ..., L-1$. We would like to prove the same result for $\Delta_r[Z^{(\ell+1)}]$. Let $p \geq 2$ be an even integer (the exponent in the definition of tensor growth).

General case: K is even or $r \neq K+1$

It is clear from the definitions that

$$\Delta_r[Z^{(\ell+1)}] = \Delta_r[X^{(\ell)}] \odot \mathbf{W}^{(\ell+1)}$$

We will now apply Proposition S.2.9 to the following tensor diagram: $V_1 = [p], V_2 = [r], E = [p] \times [r], I = \emptyset, T_v = \Delta_r[X^{(\ell)}]$. Since p is even,

$$\mathbb{E}[\Delta_r[Z^{(\ell+1)}]^p_{i_1,...,i_r}] = \left(\frac{2}{n}\right)^{rp/2} \sum_{\pi} \mathbb{E}[D_{\pi,1}](D_{\pi,2})_{i_1,...,i_r}$$

where the sum is over pairings of $[p] \times [r]$, the $D_{\pi,1}$ are arbitrary tensor diagrams on p copies of $\Delta_r[X^{(\ell)}]$ where every vertex on the right has degree 2, and $I = V_2$, and the value of $(D_{\pi,2})_{i_1,...,i_r}$ is always 1 or 0.

Therefore, to complete the result, we are reduced to proving that tensor diagrams of the form $D_{\pi,1}$ above have $n^{\frac{p}{2}(r-K)}$-growth. From Lemma S.2.16, we know that each terms $\Delta_r[X^{(\ell)}]$ can be written as a sum of connected diagrams of the $\kappa[Z^{(\ell)}]$ and $\Delta[Z^{(\ell)}]$ tensors with at least one $\Delta[Z^{(\ell)}]$ included, and the number and coefficients of these diagrams are independent of n. After replacing each copy of $\Delta_d[X^{(\ell)}]$ by one of these diagrams, we are reduced to proving that each of the resulting fully-contracted tensor diagrams has $n^{\frac{p}{2}(r-K)}$-growth. Let us recap which diagrams these are:

- $V_1 = \bigsqcup_{i=1}^p V_{1,i}$
- $|V_2| = \frac{1}{2}pr$.
- The edges are arbitrary except for the condition that the restriction to $V_{1,i}$ and its neighbors must be connected for all $i \in [p]$, and for each $v \in V_2$ there are exactly two values of i for which $V_{1,i}$ is connected to v by an edge.
- $I = V_2$, the tensor diagram is fully contracted.
- All of the tensors are either $\kappa[Z^{(\ell)}]$ or $\Delta[Z^{(\ell)}]$, and for each $i \in [p]$ at least one of the tensors attached to $V_{1,i}$ is a $\Delta[Z^{(\ell)}]$.

Now, we will apply Lemma S.2.11 by using the decomposition $V_1 = U_1 \sqcup U_2$ with U_1 consisting of all vertices with a $\kappa[Z^{(l)}]$ tensor attached, and U_2 all vertices with a $\Delta[Z^{(l)}]$ tensor. This tells us that

$$D^2_{\pi,1} \leq D^{\kappa}_{\pi,1} D^{\Delta}_{\pi,1}.$$

What can we say about the tensor diagram $D^{\kappa}_{\pi,1}$? It is a fully-contracted tensor diagram where every tensor is of the form $\kappa[Z^{(l)}]$. The cardinality of $V_2(D^{\kappa}_{\pi,1})$ is the same as $|V_2(D_{\pi,1})| = \frac{1}{2}pr$, and therefore we know by Proposition S.2.12 that this diagram has $n^{\frac{p}{2}r}$-growth.

It remains to prove that $D^{\Delta}_{\pi,1}$ has $n^{\frac{1}{2}pr - pK}$-growth. This diagram evaluates to a sum of $n^{\frac{1}{2}pr}$ terms each of which is a product of at least $2p$ tensors $\Delta[Z^{(\ell)}]$ (the factor 2 comes from the fact that the $D^{\Delta}_{\pi,1}$ is has double the number of Δ's as $D_{\pi,1}$). If we take an individual term, we can bound its

expectation using Hölder's inequality in terms of the bounds for moments of the $\Delta[Z^{(\ell)}]$'s. Since each term has at least $2p$ tensors, we get a uniform bound of $O(n^{-pK})$ where the constant is independent of the term we chose.

Since the bound is uniform, we see that $D^{\Delta}_{\pi,1}$ has $n^{\frac{1}{2}pr-pK}$-growth.

Special case: $4 \leq r = K+1 \equiv 0 \bmod 2$

For this case, note that

$$\Delta_{K+1}[Z^{(\ell+1)}] = \left(h_{\geq 2}\kappa_{K+1}[X^{(\ell)}] + g^{\frac{K+1}{2}} h_0 \Delta_{K+1}[X^{(\ell)}]\right) \odot \mathbf{W}^{(\ell+1)}$$

Consider the p-th power of $h_{\geq 2}\kappa_{K+1}[X^{(\ell)}] \odot \mathbf{W}^{(\ell+1)}$. Applying Proposition S.2.9, we see that we require all fully-contracted diagrams with p copies of $h_{\geq 2}\kappa_{K+1}[X^{(\ell)}]$ to have $n^{\frac{1}{2}p}$-growth.

A fully-contracted single copy of $h_{\geq 2}\kappa_{K+1}[X^{(\ell)}]$ is 0, by definition of the harmonic decomposition. Therefore, it is sufficient to prove that a connected fully-contracted diagram of $h_{\geq 2}\kappa_{K+1}[X^{(\ell)}]$'s must have n-growth (since no tensor can be a single connected component, the number of connected components must be at most $\frac{1}{2}p$). Suppose that some such connected fully-contracted diagram has $m \geq 1$ tensors. Using the expression

$$h_{\geq 2}\kappa_{K+1}[X^{(\ell)}] = \kappa_{K+1}[X^{(\ell)}] - c^{\mathrm{h}}_{\frac{K+1}{2},0} g^{\frac{K+1}{2}} \mathrm{tr}^{\frac{K+1}{2}} \kappa_{K+1}[X^{(\ell)}]$$

we can expand using the binomial theorem, and assume that $0 \leq a \leq m$ of the tensors are of the second type. Since $c^{\mathrm{h}}_{\frac{K+1}{2},0} = O(n^{-\frac{K+1}{2}})$, we gain a factor of $n^{-a\frac{K+1}{2}}$, but the number of connected components can increase by $a\frac{K+1}{2}$. Therefore it suffices to prove that any connected fully-contracted tensor diagram of $\kappa_{K+1}[X^{(\ell)}]$'s has n-growth. This follows directly from Proposition S.2.12.

Now consider the p-th power of $g^{\frac{K+1}{2}} h_0 \Delta_{K+1}[X^{(\ell)}] \odot \mathbf{W}^{(\ell+1)}$. This is the p-th power of the scalar value $h_0\Delta_{K+1}[X^{(\ell)}]$ multiplied by $p\frac{K+1}{2}$ copies of $(\mathbf{W}^{(\ell+1)}\mathbf{W}^{(\ell+1)\top})_{i,j}$, which is easily seen to be $O(1)$. So we simply need to show that $h_0\Delta_{K+1}[X^{(\ell)}]$ has $n^{-K/2}$-growth. Given that

$$h_0\Delta_{K+1}[X^{(\ell)}] = c^{\mathrm{h}}_{\frac{K+1}{2},0} \mathrm{Tr}^{\frac{K+1}{2}} \Delta_{K+1}[X^{(\ell)}]$$

and $c^{\mathrm{h}}_{\frac{K+1}{2},0} = O(n^{-\frac{K+1}{2}})$, it suffices to prove that $\Delta_{K+1}[X^{(\ell)}]$ has $n^{-\frac{K}{2}}$-growth. This is true since Lemma S.2.16 implies that the elements of $\Delta_{K+1}[X^{(\ell)}]$ are products of elements of $\kappa_{r'}[Z^{(\ell)}]$ and at least one element of $\Delta_{K+1}[Z^{(\ell)}]$, so the inductive hypothesis and Corollary S.2.14 complete the argument.

Special case: $K = 1, r = 2$

In this case,

$$\Delta_2[Z^{(\ell+1)}] = \left(h_2\kappa_2[X^{(\ell)}] + g h_0 \Delta_2[X^{(\ell)}]\right) \odot \mathbf{W}^{(\ell+1)} + (g_{\mathbf{M}^{(\ell+1)}} - g_{\mathbf{W}^{(\ell+1)}\mathbf{W}^{(\ell+1)\top}}) h_0 \tilde{\kappa}_2[X^{(\ell)}]$$

The first term is dealt with exactly as in the previous special case. We simply need to show that the final term has $n^{-\frac{1}{2}}$-growth. The off-diagonal entries of $\mathbf{W}^{(\ell+1)}\mathbf{W}^{(\ell+1)\top}$ have $n^{-\frac{1}{2}}$-growth. Using $\tilde{\kappa}_2[X^{(\ell)}] = \kappa_2[X^{(\ell)}] - \Delta_2[X^{(\ell)}]$, is suffices to show each of these terms has $O(1)$-growth. This follows exactly as in the final part of the previous special case, along with Corollary S.2.14. □

S.3 Mathematical preliminaries for Hermite-based algorithms

In this section we introduce the notation and basic results needed to follow the precise description of our cumulant propagation algorithms given in Section S.4.

S.3.1 Partitions and diagonals

Here, we introduce basic terminology for tensors, partitions, and diagonals.

Definition S.3.1 (Diagonals of tensors)**.** Given $\pi \vdash [r]$ and indices $j_1, j_2 \in [r]$, we say $j_1 \sim_\pi j_2$ when j_1 and j_2 lie in the same block of π, and we say the indices $(i_1, \dots, i_r)$ are of *type* π to mean that, for all $j_1, j_2 \in [r]$, $i_{j_1} = i_{j_2}$ if and only if $j_1 \sim_\pi j_2$. Then, for $\mathbf{T} \in \mathcal{T}^r(\mathbb{R}^n)$, the π-*diagonal* of $\mathbf{T}$ is defined by

$$(\mathfrak{D}_\pi \mathbf{T})_{i_1,\dots,i_r} := \begin{cases} \mathbf{T}_{i_1,\dots,i_r} & \mathrm{type}(i_1, \dots, i_r) = \pi \\ 0 & \text{otherwise.} \end{cases}$$

In this way, each integer partition $\pi \vdash [r]$ defines an orthogonal projection $\mathfrak{D}_\pi \colon \mathcal{T}^r(\mathbb{R}^n) \to \mathcal{T}^r(\mathbb{R}^n)$. It is also clear that, for any $\mathbf{T}$,

$$\mathbf{T} = \sum_{\pi \vdash [d]} \mathfrak{D}_\pi \mathbf{T},$$

whence the images of the $\mathfrak{D}_\pi$'s in fact form an orthogonal decomposition of $\mathcal{T}^r(\mathbb{R}^n)$.

Definition S.3.2 (Diagonal slices of symmetric tensors)**.** We say that $\pi \vdash [r]$ has *type* equal to the *integer partition* $\lambda = (\lambda_1, \dots, \lambda_b) \vdash r$, written $\mathrm{type}(\pi) = \lambda$, if λ_i is the size of the ith largest block of π and b is the number of blocks in π.[18] By convention, $\lambda_1 \geq \ldots \geq \lambda_b$. Suppose now $\mathbf{T}$ is *symmetric*, which we write $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$. Then, all $\mathfrak{D}_\pi \mathbf{T}$'s for π's associated with the same λ are equal up to permutation, whence

$$\mathfrak{D}_\lambda \mathbf{T} := \sum_{\substack{\pi \vdash [r] \\ \mathrm{type}(\pi) = \lambda}} \mathfrak{D}_\pi \mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n). \tag{8}$$

When manipulating diagonals algorithmically, it suffices to track only *diagonal slices*

$$(\tilde{\mathfrak{D}}_\lambda \mathbf{T})_{i_1,\dots,i_b} := (\mathfrak{D}_\lambda \mathbf{T})_{\underbrace{i_1,\dots,i_1}_{\lambda_1 \text{ times}},\dots,\underbrace{i_b,\dots,i_b}_{\lambda_b \text{ times}}}. \tag{9}$$

Note that $\tilde{\mathfrak{D}}_\lambda \mathbf{T}$ is *not* symmetric. In fact, writing λ in the alternative notation $\lambda = (1^{b_1} 2^{b_2} \dots r^{b_r})$ (denoting that there are b_i blocks of size i), we find that $\tilde{\mathfrak{D}}_\lambda \mathbf{T}$ is invariant with respect to the action of the Young subgroup $\prod_i S_{b_i}$ induced by the integer partition $(b_1, \dots, b_d) \vdash b$. That is, $\tilde{\mathfrak{D}}_\lambda \mathbf{T}$ is only invariant to permutations of indices corresponding to equally sized blocks of λ. We call such tensors λ-symmetric and denote their subspace $\mathcal{S}^\lambda(\mathbb{R}^n) \subseteq \mathcal{T}^b(\mathbb{R}^n)$.

We write the Moore-Penrose pseudoinverse of $\tilde{\mathfrak{D}}_\lambda$ as $\tilde{\mathfrak{D}}_\lambda^{-1} \colon \mathcal{S}^\lambda(\mathbb{R}^n) \to \mathcal{S}^r(\mathbb{R}^n)$, which has the following explicit form:

$$(\tilde{\mathfrak{D}}_\lambda^{-1} \mathbf{T})_{\underbrace{i_1,\dots,i_1}_{\lambda_1 \text{ times}},\dots,\underbrace{i_b,\dots,i_b}_{\lambda_b \text{ times}}} = \mathbf{T}_{i_1,\dots,i_b}$$
$$(\tilde{\mathfrak{D}}_\lambda^{-1} \mathbf{T})_{i_1,\dots,i_r} = 0 \qquad \text{if } \mathrm{type}(\mathrm{type}(i_1, \dots, i_r)) \neq \lambda.$$

Then

$$\tilde{\mathfrak{D}}_\lambda \tilde{\mathfrak{D}}_\lambda^{-1} = \mathrm{id}_{\mathcal{S}^\lambda(\mathbb{R}^n)}, \qquad \tilde{\mathfrak{D}}_\lambda^{-1} \tilde{\mathfrak{D}}_\lambda = \mathfrak{D}_\lambda.$$

As a slight abuse of notation, we may consider $\tilde{\mathfrak{D}}_{\underline{u}} \mathbf{T}$ for arbitrary (possibly unsorted, not necessarily positive) $\underline{u} \in \mathbb{Z}^b_{\geq 0}$ satisfying $\sum_i u_i = r$. In this case, we treat the diagonal slice as constant along indices corresponding to zero entries of $\underline{u}$, excepting that we still enforce that any repeated-index entry is zero. That is,

$$(\tilde{\mathfrak{D}}_{\underline{u}} \mathbf{T})_{i_1,\dots,i_b} := \mathbb{1}[\text{all } j_i \text{ distinct}] \mathbf{T}_{\underbrace{i_1,\dots,i_1}_{u_1 \text{ times}},\dots,\underbrace{i_b,\dots,i_b}_{u_b \text{ times}}}.$$

(Note that (9) also gives zero entries when any j_i is repeated—this follows from the definition of $\mathfrak{D}_\lambda$. Thus the two definitions of $\tilde{\mathfrak{D}}$ we give coincide for $\lambda \vdash d$.) We let $\tilde{\mathfrak{D}}_{\underline{u}}^{-1}$ denote the Moore-Penrose pseudoinverse of $\tilde{\mathfrak{D}}_{\underline{u}}$.

[18] Here we overload the type operator. The type of a tuple of indices is a set partition, and the type of a set partition is an integer partition.

S.3.2 Combinatorial restatement of Hermite-based diagram summation formula

Theorem A.2, the Hermite-based diagram summation formula for cumulants, serves as our basic formula for propagating cumulants through non-linear activation functions. We restate the requisite definitions and provide a combinatorial restatement here, and defer the proof to S.7. This combinatorial restatement is possible simply due to the fact that in a diagram (see Definition A.1) there is an ordering of the points in each $[k_j]$ which doesn't affect the value of the corresponding term. Grouping terms that are equal up to permuting these points gives the combinatorial statement below.

Definition S.3.3. Let $r \in \mathbb{Z}_{>0}$ and $\underline{k} \in \mathbb{Z}^r_{\geq 0}$. A *vector partition* of $\underline{k}$ is a tuple of tuples in lexicographically descending order that sum to $\underline{k}$:

$$\begin{aligned} &\nu = (\underline{u}_1, \dots, \underline{u}_b) \in (\mathbb{Z}^r_{\geq 0})^b \\ \text{s.t.} \quad &k_j = \sum_{i=1}^{b} u_{i,j} \qquad \forall j \in [r]. \\ &\underline{u}_i \geq_{\text{lex}} \underline{u}_{i+1} \qquad \forall i \in [b-1]. \end{aligned}$$

We call the $\underline{u}_i$ *blocks* of the vector partition. Each vector partition is an equivalence class of diagrams modulo the forgetting of the distinction between vertices in the same group $\{(v,w) : w \in \{1, \dots, k_v\}\}$. In this way, the relationship between diagrams and vector partitions is analogous to the relationship between set partitions and integer partitions; we thus write $\nu = \text{type}(\pi)$ if the diagram π is in the equivalence class induced by the vector partition ν. We also say that the type of each block $S \in \pi$ is its corresponding vector in ν:

$$\text{type}(S)_i = \#\{w \in [k_i] : (v,w) \in S\}.$$

Taking this perspective, we can define the properties *connected* and $[\leq M]$*-mixed* analogously to the definitions for diagrams. We denote the set of all vector partitions of $\underline{k}$ by $\text{Vec}(\underline{k})$, and the subset of connected $[\leq M]$-mixed vector partitions by $\text{cVec}_{[\leq M]}(\underline{k})$.

Proposition S.3.4. *Fix $\nu \in \text{Vec}(\underline{k})$, and for each $\underline{0} \leq \underline{u} \leq \underline{k}$ (entrywise) let $\nu(\underline{u})$ denote the number of times $\underline{u}$ appears in ν. Then, the number of diagrams with type ν is*

$$c_\nu^{\text{vec}} := \frac{\prod_{i=1}^r k_i!}{\prod_{\underline{u}=\underline{0}}^{\underline{k}} \left(\nu(\underline{u})! \prod_{i=1}^r (u_i!)^{\nu(\underline{u})}\right)}.$$

Theorem S.3.5 (Combinatorial restatement of A.2)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi \colon \mathbb{R} \to \mathbb{R}$ be a polynomial, which we think of as acting coordinatewise on Z. Then*

$$\kappa_r[\phi(Z)]_{j_1,\dots j_r} = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^r \frac{1}{k_v!} \widehat{\phi}_{k_v}^{(\mathbb{E}[Z_{j_v}], \text{Var}(Z_{j_v}))} \right) \sum_{\nu \in \text{cVec}_{[\leq 2]}(\underline{k})} c_\nu^{\text{vec}} \prod_{\underline{u} \in \nu} (\tilde{\mathfrak{D}}_{\underline{u}} \kappa_{|\underline{u}|}[Z])_{j_1,\dots,j_r}. \tag{10}$$

Both of these results are proven in Section S.7.4. When ϕ is not a polynomial, the sum in (10) is infinite and possibly diverges (for polynomials ϕ, all Hermite coefficients of degree larger than $\deg(\phi)$ are zero, so this summation is finite). It turns out that, for large enough n, a careful truncation of the series via a condition (14) on vector partitions results in an approximation of the cumulant that is sufficient to obtain the desired MSE bound; see Section S.6 for the analysis. Our cumulant propagation algorithm uses this truncation.

S.3.3 Power cumulants

Section S.3.2 provides a summation formula for cumulants after applying a coordinatewise activation function. As mentioned in Section 4.1, we will apply this formula to powers of the activation function in order to compute power cumulants.

Definition S.3.6. Let X be an $\mathbb{R}^n$-valued random variable, and $r \geq 1$. For a multi-exponent $\underline{\alpha} \in \mathbb{Z}^r_{\geq 1}$, define the power cumulant

$$\kappa^{\mathsf{P}}_{\underline{\alpha}}[X]_{i_1,\dots,i_r} = \begin{cases} \kappa[X_{i_1}^{\alpha_1}, \dots, X_{i_r}^{\alpha_r}] & \text{if } i_1, \dots, i_r \text{ are pairwise distinct} \\ 0 & \text{o/w.} \end{cases}$$

This is a tensor in $\mathcal{S}^\alpha(\mathbb{R}^n)$, the set of tensors invariant under permutations in $\text{Stab}_{S_r}(\alpha)$.

The standard cumulant tensors can be written as a sum of products of these tensors with delta functions.

Proposition S.3.7 (Power cumulant partition formula)**.** *Cumulants can be computed from power cumulants slicewise: for each $\lambda \vdash r$ with b blocks,*

$$\tilde{\mathfrak{D}}_\lambda \kappa_r[X] = \sum_{\nu \in \mathrm{Vec}(\lambda)} c_\nu^{\mathrm{vec}} c_\nu^{\mathsf{P}} \prod_{\underline{u} \in \nu} \tilde{\mathfrak{D}}_{\underline{u}} \kappa^{\mathsf{P}}_{|\underline{u}|}[X],$$

where the product is taken elementwise and

$$c_\nu^{\mathsf{P}} := \sum_{\omega \vdash \nu} \mathbb{1}[\omega \text{ totally disconnected}](-1)^{|\omega|-1}(|\omega|-1)!.$$

Here the sum is over set partitions ω of ν (which itself is a set of tuples in $[b]$), $|\omega|$ is the number of blocks in ω, and ω is said to be totally disconnected if for each block of ω, all tuples in that block have disjoint support in $[b]$.

We defer the proof of Proposition S.3.7 to Section S.8.3.

S.3.4 Hermite coefficients of ReLU

In this section, we provide explicit expressions for the Hermite coefficients of ReLU.

Proposition S.3.8. *Let $\phi = \mathrm{ReLU}$. Fix $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}_{>0}$ and let $Z \sim \mathcal{N}(\mu, \sigma^2)$. Write $\alpha = \mu/\sigma$, let He_i denote the ith standard probabilist's Hermite polynomial, and let φ and Φ denote the standard Gaussian probability density function and cumulative distribution function, respectively. Then, for any $k \in \mathbb{Z}_{\geq 0}$ and $p \in \mathbb{Z}_{>0}$,*

$$\widehat{\phi^p}_k^{(\mu,\sigma^2)} = \begin{cases} p!\Phi(\alpha) & k = p \\ p!(-1)^{k-p-1}\sigma^{-(k-p)}\,\mathrm{He}_{k-p-1}(\alpha)\varphi(\alpha) & k > p \\ \sigma^{p-k}(P_{1,p,k}(\alpha)\varphi(\alpha) + P_{2,p,k}(\alpha)\Phi(\alpha)) & k < p \end{cases}$$

where

$$P_{1,p,k}(\alpha) := (p-k+1)_k \sum_{j=0}^{p-k} \binom{p-k}{j} \alpha^{p-k-j} \left(\sum_{m=0}^{\lfloor (j-1)/2 \rfloor} \binom{j}{2m} (2m-1)!!\,\mathrm{He}_{j-2m-1}(-\alpha) \right)$$

$$P_{2,p,k}(\alpha) := (p-k+1)_k \sum_{j=0}^{p-k} \mathbb{1}[j \text{ even}] \binom{p-k}{j} \alpha^{p-k-j}(j-1)!!.$$

and $(a)_t := a(a+1)...(a+t-1)$ is the Pochhammer symbol.

Proof. This is a scaling and shifting of the $(\mu, \sigma^2) = (0, 1)$ case, which is calculated in Kagawa [2015]. □

S.3.5 Going between diagonals and harmonics

Definition S.3.9. A *multigraph* γ on a vertex set $[b]$ is a multiset of unordered pairs of (possibly nondistinct) elements from $[b]$, or equivalently a mapping $\gamma\colon \{\{i,j\} \subseteq [b]\} \to \mathbb{N}$. The total number of edges of γ is $|\gamma| := \sum_{\{i,j\} \subseteq [b]} \gamma(\{i,j\})$, and the degree of a vertex $j \in [b]$ is

$$d_\gamma(j) := \sum_{j' \neq j} \gamma(\{j,j'\}) + 2\gamma(\{j,j\}).$$

We say γ is *totally disconnected (t.d.)* if it contains only self-loops, i.e. $\gamma(\{i,j\}) = 0$ for any $i \neq j$. Let $\Gamma([b], m)$ denote the set of multigraphs on $[b]$ with m edges, and $\Gamma_{\text{t.d.}}([b], m)$ the subset of those multigraphs that are totally disconnected.

Proposition S.3.10. *There exist coefficients $c^{\mathrm{g}}_{\lambda,\gamma}$ such that, for any $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$, any integer $0 \le s \le \lfloor \frac{r}{2} \rfloor$, and any $\lambda = (\lambda_1, \dots, \lambda_b) = (1^{b_1} \dots r^{b_r}) \vdash r$ with b blocks,*

$$\mathrm{tr}^s \, \mathfrak{D}_\lambda \mathbf{T} = \sum_{\gamma \in \Gamma_{t.d.}([b],s)} \left(c^{\mathrm{g}}_{\lambda,\gamma} \mathbb{1}_{\lambda \ge d_\gamma} \tilde{\mathfrak{D}}^{-1}_{(\lambda - d_\gamma)} \operatorname*{Sum}_{\{j : \lambda_j = d_\gamma(j)\}} \tilde{\mathfrak{D}}_\lambda \mathbf{T} \right), \tag{11}$$

where $\mathrm{Sum}_{\{j:\lambda_j = d_\gamma(j)\}}$ is the operator that sums along indices $\{j : \lambda_j = d_\gamma(j)\}$ and the difference between λ and d_γ is taken entrywise. Explicitly,

$$c^{\mathrm{g}}_{\lambda,\gamma} := \frac{2^s s! r! \prod_{i \in [b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}}{(r - 2s + 1)_{2s} \prod_{j \in [r]} b_j! \prod_{i \in [b]} \lambda_i! 2^{\gamma(\{i\})} \prod_{\{i,j\} \subseteq [b]} \gamma(\{i,j\})!}.$$

Provided $\tilde{\mathfrak{D}}_\lambda \mathbf{T}$, (11) can be computed in $O(n^{r-2s})$ time, faster than the $O(n^r)$ time of naïvely applying $\tilde{\mathfrak{D}}^{-1}_\lambda$ then tr^s in sequence.

Proposition S.3.11. *There exist coefficients $\tilde{c}^{\mathrm{g}}_{\lambda,\gamma}$ such that, given an integer $0 \le s \le \lfloor \frac{r}{2} \rfloor$ and $\lambda = (\lambda_1, \dots, \lambda_b) = (1^{b_1} \dots r^{b_r}) \vdash r$ with b blocks, for any $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$ and $\mathbf{M} \in \mathcal{S}^2(\mathbb{R}^n)$,*

$$\tilde{\mathfrak{D}}_\lambda g^s_{\mathbf{M}}(\mathbf{T}) = \sum_{\gamma \in \Gamma([b],s)} \tilde{c}^{\mathrm{g}}_{\lambda,\gamma} \mathbf{T} \odot_\gamma \mathbf{M}, \tag{12}$$

where $(\mathbf{T} \odot_\gamma \mathbf{M}) \in \mathcal{S}^\lambda(\mathbb{R}^n)$ is defined by

$$(\mathbf{T} \odot_\gamma \mathbf{M})_{i_1,\dots,i_b} = \begin{cases} (\tilde{\mathfrak{D}}_{\lambda - d_\gamma} \mathbf{T})_{i_1,\dots,i_b} \prod_{\{u,v\} \in \gamma} \mathbf{M}_{i_u, i_v} & \underline{i} \text{ all distinct} \\ 0 & \text{otherwise.} \end{cases}$$

Explicitly,

$$\tilde{c}^{\mathrm{g}}_{\lambda,\gamma} := \frac{\prod_{i \in [b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}}{\prod_{i \in [b]} 2^{\gamma(\{i\})} \prod_{\{i,j\} \subseteq [b]} \gamma(\{i,j\})!}.$$

Equation 12 can be computed in $O(n^b)$ time, faster than the $O(n^{r+2s})$ time of naïvely applying $g_{\mathbf{W}}$ and $\mathfrak{D}_\lambda$ in sequence.

We defer the proofs to Section S.8.4.

S.4 Descriptions of Hermite-based algorithms

In this section, we describe in detail the basic cumulant propagation algorithm (Section S.4.1), as well as the augmented (Section S.4.2)and factorized (Section S.4.3) variants.

Our basic cumulant propagation estimator takes $O(n^{K+1})$ time and incurs $O(n^{-K})$ MSE for all *integral* $K \ge 0$, thus matching sampling for non-integral K as well (by taking the floor). The factorized version of this algorithm takes only $O(n^K)$ time to compute the same estimate (and thus obtain the same MSE). The augmented version improves the constant factor of the leading-order term in MSE without worsening the leading-order term in time (assuming unfactorized).

Propagating the harmonic projections through a linear layer $\mathbf{W}$ is conceptually easy (but is asymptotically the most expensive step): we simply apply the symmetric contraction $\odot\mathbf{W}$ and replace cups with $\mathbf{W}\mathbf{W}^\top$-cups. For nonlinearities, we use the formulae in Section S.3.5 to move from the harmonic to the diagonal decomposition, apply the power cumulant combinatorics in the diagonal setting, then project back to harmonics.

S.4.1 Basic algorithm

The input to the algorithm is the following data:

- The parameters of an MLP (as defined in Section 2). That is, $L \ge 0$, $n \ge 1$, activation functions $\phi_1, ..., \phi_L : \mathbb{R} \to \mathbb{R}$, and weights $\boldsymbol{\theta} = (\mathbf{W}^{(1)}, ..., \mathbf{W}^{(L+1)}) \in \mathbb{R}^{(L+1) \times n \times n}$.
- The "maximum degree" $K \ge 1$. The choice of K as a function of the error tolerance ε is discussed in Section S.1.1.

Given these parameters, we define the following notation:

- Let $X = X^{(0)} \sim \mathcal{N}(0, \mathbf{I}_n)$ denote the input to the MLP. Then recursively define

$$\forall 1 \le \ell \le L+1, \qquad Z^{(\ell)} := \mathbf{W}^{(\ell)} X^{(\ell-1)}$$
$$\forall 1 \le \ell \le L, \qquad X^{(\ell)} = \phi_\ell(Z^{(\ell)}).$$

- Recall $(\widehat{\phi_\ell^p})_k^{(\mu,\sigma^2)}$ denotes the k-th probabilist's Hermite coefficient of ϕ_ℓ^p (the pointwise pth power of the ℓ-th layer activation ϕ_ℓ) with respect to $\mathcal{N}(\mu, \sigma^2)$, as defined in (1).

Our algorithm then propagates tensors $\tilde{\eta}_r[X^{(\ell)}] \in \mathcal{S}^{r-2s_K(r)}(\mathbb{R}^n)$ through MLP layers. These are estimates for $h_{\ge r-2s_K(r)}(\kappa_r[X^{(\ell)}])$, where

$$s_K(r) := \begin{cases} 0 & r \le K \\ \dfrac{r}{2} & r = K+1 \equiv 0 \mod 2 \\ \infty & \text{otherwise.} \end{cases}$$

By convention, if $2s_K(r) > r$, then $\tilde{\eta}_r[X^{(\ell)}] = 0$. Thus, only cumulants with degree r no more than

$$R := \begin{cases} K & K \equiv 0 \mod 2 \\ K+1 & K \equiv 1 \mod 2 \end{cases}$$

are tracked at all.

The algorithm proceeds as follows:

1. Initialize tensors

$$\tilde{\eta}_r[X^{(0)}] = h_{\ge r-2s_K(r)}(\kappa_r[X^{(0)}]) = \begin{cases} 1 & r = 2 \text{ and } K = 1 \\ \mathbf{I}_n & r = 2 \text{ and } K > 1 \\ 0 & r \ne 2. \end{cases}$$

2. For $1 \le \ell \le L$, repeat the following:
   (a) Compute the tensors

$$\mathbf{T}_r^{(\ell)} := \tilde{\eta}_r[X^{(\ell-1)}] \odot \mathbf{W}^{(\ell)} \qquad \forall r \in [R]$$
$$\mathbf{M}^{(\ell)} := \begin{cases} \mathfrak{D}_{(2)}(\mathbf{W}^{(\ell)}\mathbf{W}^{(\ell)\top}) & K = 1 \\ \mathbf{W}^{(\ell)}\mathbf{W}^{(\ell)\top} & K > 1. \end{cases}$$

   Note that each $\mathbf{T}_r^{(\ell)}$ is a symmetric contraction of a $(r - 2s_K(r))$-tensor, and thus takes time $O(n^{r-2s_K(r)+1}) \le O(n^{K+1})$. The matrix $\mathbf{M}^{(\ell)}$ takes time $O(n^2)$ if $K = 1$, otherwise $O(n^3)$. In any case, this is no more than the $O(n^{K+1})$ budget.
   (b) Write the estimated pre-activation mean and variance vectors

$$\mu := \mathbf{T}_1^{(\ell)} \in \mathbb{R}^n, \qquad \sigma^2 := \tilde{\mathfrak{D}}_{(2)} g_{\mathbf{M}^{(\ell)}}^{s_K(2)} \mathbf{T}_2^{(\ell)} \in \mathbb{R}^n.$$

   For each $r \in [K]$ and $\lambda \vdash r$, compute the λ-slice of the activation power cumulant using a truncation of (10) applied to powers of the activation function: [19]

$$\tilde{\mathfrak{D}}_\lambda(\tilde{\kappa}^{\mathsf{P}}[X^{(\ell)}])_{i_1,\dots,i_b} \tag{13}$$
$$:= \sum_{\substack{\underline{k} \in \mathbb{Z}_{\ge 0}^b \\ |\underline{k}| \le 2K-1}} \frac{1}{\underline{k}!} \left(\prod_{a=1}^{b} \left(\widehat{(\phi_\ell^{\lambda_a})}_{k_a}^{(\mu_{k_a}, \sigma_{k_a}^2)} \right) \right)_{i_a} \sum_{\substack{\nu \in \mathrm{cVec}_{[\le 2]}(\underline{k}) \\ k(\nu) \le K}} \prod_{\underline{u} \in \nu} (\tilde{\mathfrak{D}}_{\underline{u}} g_{\mathbf{M}^{(\ell)}}^{s_K(|\underline{u}|)} \mathbf{T}_{|\underline{u}|}^{(\ell)})_{i_1,\dots,i_b},$$

[19] In practice, instead of summing diagrams directly, we sum orbits of diagrams under permutations of indices then λ-symmetrize the result.

where each $\tilde{\mathfrak{D}}_{\underline{u}} g_{\mathbf{M}^{(\ell)}}^{s_K(|\underline{u}|)} \mathbf{T}_{|S|}^{(\ell)}$ is computed using (12), each Hermite coefficient is computed using Proposition S.3.8 if $\phi_\ell = \mathrm{ReLU}$ and Gauss-Hermite otherwise (see Section F), and

$$1 + \sum_{\underline{u} \in \nu} \left(\sum_{j=1}^{b} u_j - 1 - \mathbb{1}[u_j \text{ all even}] \right) =: k(\nu) \le K \tag{14}$$

is the condition for including vector partition ν. If K is odd, do the same for all $\lambda \vdash K+1$ having no singleton blocks.

(c) For the same set of λ as in the previous step, convert power cumulants to cumulants slicewise:

$$\tilde{\mathfrak{D}}_\lambda \tilde{\kappa}_r[X^{(\ell)}] := \sum_{\nu \in \mathrm{Vec}(\lambda)} c_\nu^{\mathrm{vec}} c_\nu^{\mathsf{P}} \prod_{\underline{u} \in \nu} \tilde{\mathfrak{D}}_{\underline{u}} \kappa_r^{\mathsf{P}}[X^{(\ell)}],$$

(This is (S.3.7) applied to the estimated power cumulants.)

(d) For each $r \in [R]$, project the estimated cumulants onto the $(\le r - 2s_K)$-harmonics:

$$\eta_r[X^{(\ell)}] := \sum_{s'=s}^{\lfloor r/2 \rfloor} g^{s'-s} \left(\sum_{t=0}^{\lfloor (r-2s)/2 \rfloor} c_{s',t}^{\mathrm{h}} g^t \operatorname{tr}^{s'+t} \left(\sum_{\lambda \vdash [r]} \tilde{\mathfrak{D}}_\lambda \tilde{\kappa}_r[X^{(\ell)}] \right) \right)$$

where the trace is distributed over the sum of diagonals then computed with (11).

3. Return $\mathbf{W}^{(L+1)} \tilde{\eta}_1[X^{(L)}]$.

S.4.2 Augmented algorithm

The basic algorithm in Section S.4.1 works by tracking every harmonic part that would incur at least $\Omega(n^{-K+1})$ MSE if dropped; the largest of these parts takes $\Theta(n^{K+1})$ time to propagate. We may consider an algorithm that additionally tracks every harmonic part that can be propagated in $O(n^K)$ time and contributes at least $\Omega(n^{-K})$ MSE. This augmented algorithm would improve the constant factor of the leading-order term in MSE, while keeping the leading-order term in FLOPs the same.[20] It follows from our bounds on cumulant sizes (Proposition S.2.12) that the augmented algorithm corresponds to replacing s_K in the basic algorithm with

$$s_K^{\mathrm{aug}}(r) := \begin{cases} 0 & r \le K \\ 1 & r = K+1 \\ \dfrac{r}{2} & r = K+2 \equiv 0 \mod 2 \\ \infty & \text{otherwise.} \end{cases}$$

All other parts of cumulant propagation remain the same.

S.4.3 Factorized algorithm

The time complexity of our cumulant propagation algorithm is dominated by the $O(n^{K+1})$ symmetric contraction of the full degree-K cumulant tensor during the linear step. By representing this tensor in a factorized form, we are able to reduce the linear step's runtime, and thus that of the entire algorithm, to $O(n^K)$.

Note that this improvement concerns only the width dependence—recall that our big-O notation treats the depth L as constant. If we track the dependence on L explicitly, the time complexity of basic cumulant propagation through the entire MLP scales linearly with L, while that of the factored version scales quadratically. The reason is that the factored version propagates a running list of factors is propagated through the network, adding a constant number of new factors at each layer.

For ease of exposition, we demonstrate the factorized form of the basic algorithm when $K = 3$, and defer the proof that general K can be factorized to Section S.8.5. With some additional effort, augmented cumulant propagation can also be factorized; we refer interested readers to the code (`https://github.com/alignment-research-center/mlp_kprop`).

[20] As long as $K \ge 3$. For lower K, the linear step is not the only computation that is leading order in FLOPs, so augmenting cumulant propagation in fact increases the leading-order FLOPs term.

Factoring tensors When $K = 3$, we factor $T \in \mathcal{S}^3(\mathbb{R}^n)$ as follows:

$$\mathbf{T}_{i_1,i_2,i_3} = \frac{1}{6} \sum_{\sigma \in S_3} \sum_{j=1}^{J} \mathbf{A}_{i_{\sigma(1)},j} \mathbf{B}_{i_{\sigma(2)},j} \mathbf{C}_{i_{\sigma(3)},j} \tag{15}$$

where J is the number of factors and $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{R}^{n \times J}$. Throughout, $J = O(n)$. We notate (15) sans indices as

$$\mathbf{T} = F(\mathbf{A}, \mathbf{B}, \mathbf{C}).$$

It is easily verified that sums of factored tensors are factored by concatenation:

$$F(\mathbf{A}, \mathbf{B}, \mathbf{C}) + F(\mathbf{A}', \mathbf{B}', \mathbf{C}') = F\left([\mathbf{A} \quad \mathbf{A}'], [\mathbf{B} \quad \mathbf{B}'], [\mathbf{C} \quad \mathbf{C}']\right).$$

Hence summing factored tensors sums the number of factors.

The symmetric contraction of a factored tensor is

$$F(\mathbf{A}, \mathbf{B}, \mathbf{C}) \odot \mathbf{W} = F(\mathbf{WA}, \mathbf{WB}, \mathbf{WC}).$$

When $J = O(n)$, the factors of the symmetric contraction can be computed in $O(n^3)$ time—this is what gives us the asymptotic improvement over the naïve $O(n^4)$ linear step.

Diagonals of factored tensors Let $*$ denote the entrywise (Hadamard) matrix product. Then,

$$\begin{aligned}
(\tilde{\mathfrak{D}}_{(3)} F(\mathbf{A}, \mathbf{B}, \mathbf{C}))_i &= \sum_{j=1}^{J} (\mathbf{A} * \mathbf{B} * \mathbf{C})_{i,j}, \\
(\tilde{\mathfrak{D}}_{(2,1)} F(\mathbf{A}, \mathbf{B}, \mathbf{C}))_{i_1,i_2} &= \mathbb{1}[i_1 \neq i_2] \frac{1}{3} \sum_{j=1}^{J} \left((\mathbf{B} * \mathbf{C})_{i_1,j} \mathbf{A}_{i_2,j} + (\mathbf{A} * \mathbf{C})_{i_1,j} \mathbf{B}_{i_2,j} + (\mathbf{A} * \mathbf{B})_{i_1,j} \mathbf{C}_{i_2,j} \right).
\end{aligned} \tag{16}$$

When $J = O(n)$, these can be computed in $O(n^3)$ time.

Factoring diagonals For $\mathbf{T} \in \mathcal{S}^3(\mathbb{R}^n)$,

$$\begin{aligned}
\mathfrak{D}_{(3)} \mathbf{T} &= F((\tilde{\mathfrak{D}}_{(3)} \mathbf{T}) \mathbf{1}_n^\top, \mathbf{I}_n, \mathbf{I}_n), \\
\mathfrak{D}_{(2,1)} \mathbf{T} &= 3F(\tilde{\mathfrak{D}}_{(2,1)} \mathbf{T}, \mathbf{I}_n, \mathbf{I}_n).
\end{aligned} \tag{17}$$

where $\mathbf{1}_n \in \mathbb{R}^n$ is the all-ones vector and $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ is the identity matrix.

Factoring the diagram summation formula We now consider factoring (13) when $\lambda = (1, 1, 1)$. Suppose we are at layer ℓ of cumulant propagation. Let μ and σ^2 be the propagated estimates for mean and variance respectively, and write

$$\mathbf{f}_k^{(\ell)} := \widehat{\phi}_{\ell k}^{(\mu_k, \sigma_k^2)}, \qquad \mathbf{Q}_{\underline{u}}^{(\ell)} := \tilde{\mathfrak{D}}_{\underline{u}} g_{\mathbf{M}^{(\ell)}}^{s_3(\underline{u})} \mathbf{T}_{|\underline{u}|}^{(\ell)}, \qquad \mathbf{M}^{(\ell)} := \mathbf{W}\mathbf{W}^\top.$$

where $\mathbf{T}$ comes from Step 2a. Recall each $\mathbf{Q}_{\underline{u}}^{(\ell)}$ is computed using (12). Then, grouping diagrams by orbits under permutations of indices,

$$\begin{aligned}
(\tilde{\mathfrak{D}}_{(1,1,1)} \tilde{\kappa}^{\mathsf{P}}[X^{(\ell)}])_{i_1,i_2,i_3} = \mathbb{1}[i_j \text{ all distinct}] \Bigg(& \\
\sum_{\sigma \in A_3} \sum_{k_1,\ldots,k_4 \in [2]} \frac{1}{\underline{k}!} & (\mathbf{f}_{k_1}^{(\ell)})_{i_{\sigma(1)}} (\mathbf{f}_{k_2+k_3}^{(\ell)})_{i_{\sigma(2)}} (\mathbf{f}_{k_4}^{(\ell)})_{i_{\sigma(3)}} (\mathbf{Q}_{(k_1,k_2)}^{(\ell)})_{i_{\sigma(1)},i_{\sigma(2)}} (\mathbf{Q}_{(k_3,k_4)}^{(\ell)})_{i_{\sigma(2)},i_{\sigma(3)}} \\
+ (\mathbf{f}_1^{(\ell)})_{i_{\sigma(1)}} & (\mathbf{f}_1^{(\ell)})_{i_{\sigma(2)}} (\mathbf{f}_1^{(\ell)})_{i_{\sigma(1)}} (\mathbf{Q}_{(1,1,1)}^{(\ell)})_{i_1,i_2,i_3} \\
+ \sum_{\sigma \in A_3} \frac{1}{2} & (\mathbf{f}_2^{(\ell)})_{i_{\sigma(1)}} (\mathbf{f}_1^{(\ell)})_{i_{\sigma(2)}} (\mathbf{f}_1^{(\ell)})_{i_{\sigma(3)}} (\mathbf{Q}_{(2,1,1)}^{(\ell)})_{i_{\sigma(1)},i_{\sigma(2)},i_{\sigma(3)}} \\
\Bigg), &
\end{aligned}$$

where $A_3 \subseteq S_3$ is the group of 3-cycles.[21] We factor each summand, ignoring for now the $\mathbb{1}[\text{all } i_j \text{ distinct}]$ factor.

- The sum of two-legged diagrams factors as

$$\begin{aligned}
&\sum_{\sigma\in A_3}\sum_{k_1,\dots,k_4\in[2]} \frac{1}{k!}(\mathbf{f}^{(\ell)}_{k_1})_{i_{\sigma(1)}}(\mathbf{f}^{(\ell)}_{k_2+k_3})_{i_{\sigma(2)}}(\mathbf{f}^{(\ell)}_{k_4})_{i_{\sigma(3)}}(\mathbf{Q}^{(\ell)}_{(k_1,k_2)})_{i_{\sigma(1)},i_{\sigma(2)}}(\mathbf{Q}^{(\ell)}_{(k_3,k_4)})_{i_{\sigma(2)},i_{\sigma(3)}} \\
&= \sum_{k_2\in[2]} F\left(\frac{3}{k_2!}\mathbf{I}_n, \sum_{k_1\in[2]}\frac{1}{k_1!}(\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_{k_1})\mathbf{Q}^{(\ell)}_{(k_1,k_2)}, \sum_{k_3,k_4\in[2]}\frac{1}{k_3!k_4!}(\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_{k_2+k_3})(\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_{k_4})\mathbf{Q}^{(\ell)}_{(k_3,k_4)}\right)_{i_1,i_2,i_3} \\
&=: \sum_{k_2\in[2]} F\left(\mathbf{A}^{(\ell)}_{\text{two},k_2}, \mathbf{B}^{(\ell)}_{\text{two},k_2}, \mathbf{C}^{(\ell)}_{\text{two},k_2}\right)_{i_1,i_2,i_3}. \qquad (18)
\end{aligned}$$

  (Since F by definition symmetrizes, the sum over A_3 in the left hand side turns into a factor of 3, which is pulled into the first factor of the right hand side.) This is $2n$ factors total.

- By induction, the previous layer's activation cumulants corresponding to $(1,1,1)$ are factored as $\tilde{\kappa}_3[X^{(\ell-1)}] = F(\mathbf{A}^{(\ell-1)}, \mathbf{B}^{(\ell-1)}, \mathbf{C}^{(\ell-1)})$. Then, since $\mathbf{Q}^{(\ell)}_{(1,1,1)} = \tilde{\kappa}_3[X^{(\ell-1)}] \odot \mathbf{W}$,

$$\begin{aligned}
&(\mathbf{f}^{(\ell)}_1)_{i_{\sigma(1)}}(\mathbf{f}^{(\ell)}_1)_{i_{\sigma(2)}}(\mathbf{f}^{(\ell)}_1)_{i_{\sigma(3)}}(\mathbf{Q}^{(\ell)}_{(1,1,1)})_{i_1,i_2,i_3} \\
&\quad= F((\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{W}\mathbf{A}^{(\ell-1)}, (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{W}\mathbf{B}^{(\ell-1)}, (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{W}\mathbf{C}^{(\ell-1)})_{i_1,i_2,i_3} \\
&\quad=: F\left(\mathbf{A}^{(\ell)}_{(1,1,1)}, \mathbf{B}^{(\ell)}_{(1,1,1)}, \mathbf{C}^{(\ell)}_{(1,1,1)}\right)_{i_1,i_2,i_3}. \qquad (19)
\end{aligned}$$

  This is the same number of factors that $\tilde{\kappa}_3[X^{(\ell-1)}]$ had in the previous layer.

- To factor the $(2,1,1)$ diagram, write $(j_1,j_2,j_3,j_4) = (1,1,2,3)$. Then, since $\mathbf{T}^{(\ell)}_4$ is a scalar and $\mathbf{M}^{(\ell)}$ is symmetric,

$$\begin{aligned}
(\mathbf{Q}^{(\ell)}_{(2,1,1)})_{i_1,i_2,i_3} &= \left(\tilde{\mathfrak{D}}_{(2,1,1)} g^2_{\mathbf{M}^{(\ell)}}\mathbf{T}^{(\ell)}_4\right)_{i_1,i_2,i_3} \\
&= \mathbf{T}^{(\ell)}_4 \sum_{\substack{u,v\in[4]\\ u<v}} \mathbf{M}^{(\ell)}_{i_{j_u},i_{j_v}}\mathbf{M}^{(\ell)}_{\{i_{j_s}:s\notin\{u,v\}\}} \\
&= \mathbf{T}^{(\ell)}_4\left(2\mathbf{M}^{(\ell)}_{i_1,i_1}\mathbf{M}^{(\ell)}_{i_2,i_3} + 4\mathbf{M}^{(\ell)}_{i_1,i_2}\mathbf{M}^{(\ell)}_{i_1,i_3}\right).
\end{aligned}$$

  Hence,

$$\begin{aligned}
&\sum_{\sigma\in A_3}(\mathbf{f}^{(\ell)}_2)_{i_{\sigma(1)}}\frac{1}{2}(\mathbf{f}^{(\ell)}_1)_{i_{\sigma(2)}}(\mathbf{f}^{(\ell)}_1)_{i_{\sigma(3)}}(\mathbf{Q}^{(\ell)}_{(2,1,1)})_{i_{\sigma(1)},i_{\sigma(2)},i_{\sigma(3)}} \\
&\quad= F(3\mathbf{T}^{(\ell)}_4(\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_2)(\mathfrak{D}_{(2)}\mathbf{M}^{(\ell)})\mathbf{1}^\top_n, (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1), (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{M}^{(\ell)})_{i_1,i_2,i_3} \\
&\qquad+ F(6\mathbf{T}^{(\ell)}_4(\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_2), (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{M}^{(\ell)}, (\tilde{\mathfrak{D}}^{\text{-}1}_{(2)}\,\mathbf{f}^{(\ell)}_1)\mathbf{M}^{(\ell)})_{i_1,i_2,i_3} \\
&\quad=: \sum_{k=1}^{2} F\left(\mathbf{A}^{(\ell)}_{(2,1,1),k}, \mathbf{B}^{(\ell)}_{(2,1,1),k}, \mathbf{C}^{(\ell)}_{(2,1,1),k}\right)_{i_1,i_2,i_3}. \qquad (20)
\end{aligned}$$

  This is $2n$ factors total.

Factorized cumulant propagation algorithm We get the factorized version of the $K = 3$ basic cumulant propagation algorithm by replacing Steps 2a–2d with the following:

[21] In general, the orbit-stabilizer theorem tells us to sum over coset representatives of the quotient of S_3 by the stabilizer of the diagram (or sum of diagrams) under consideration. For example, the stabilizer of $\{(2,1,1)\}$ is $\langle(23)\rangle$ (the subgroup generated by the transposition of 2 and 3), while the stabilizer of $\{(1,1,1)\}$ is all of S_3.

(a) From the previous layer, we have

$$\begin{aligned}
&\tilde{\eta}_1[X^{(\ell-1)}] \in \mathcal{S}^1(\mathbb{R}^n),\\
&\tilde{\eta}_2[X^{(\ell-1)}] \in \mathcal{S}^2(\mathbb{R}^n)\\
&\mathbf{A}^{(\ell-1)}, \mathbf{B}^{(\ell-1)}, \mathbf{C}^{(\ell-1)} \in \mathbb{R}^{n \times J^{(\ell-1)}}\\
&\tilde{\eta}_4[X^{(\ell-1)}] \in \mathcal{S}^1(\mathbb{R}^n).
\end{aligned}$$

where implicitly $\tilde{\eta}_3[X^{(\ell-1)}] = F(\mathbf{A}^{(\ell-1)}, \mathbf{B}^{(\ell-1)}, \mathbf{C}^{(\ell-1)})$, but we never explicitly compute this tensor. Compute and store

$$\begin{aligned}
\mathbf{T}_r^{(\ell)} &:= \tilde{\eta}_r[X^{(\ell-1)}] \odot \mathbf{W}^{(\ell)} \qquad \forall r \in \{1,2,4\}\\
\mathbf{M}^{(\ell)} &:= \mathbf{W}^{(\ell)} \mathbf{W}^{(\ell)\top},
\end{aligned}$$

which take time $O(n^3)$, as well as $\mathbf{W}^{(\ell)}\mathbf{A}^{(\ell-1)}, \mathbf{W}^{(\ell)}\mathbf{B}^{(\ell-1)}, \mathbf{W}^{(\ell)}\mathbf{C}^{(\ell-1)}$. which take time $O(n^2 J^{(\ell-1)})$.

(b) Compute power cumulants via diagram summation:

i. As in Step 2b, compute the estimated pre-activation mean and variance vectors. Then, for each $r \in [4]$ and $\lambda \vdash r$, if λ has no more than 2 blocks and $\lambda \neq (3,1)$, compute $\tilde{\mathfrak{D}}_\lambda(\tilde{\kappa}^{\mathsf{P}}[X^{(\ell)}])$ using (13). Note that this sum involves terms of the form

$$\tilde{\mathfrak{D}}_\lambda \mathbf{T}_3^{(\ell-1)} = \tilde{\mathfrak{D}}_\lambda F(\mathbf{W}^{(\ell)}\mathbf{A}^{(\ell-1)}, \mathbf{W}^{(\ell)}\mathbf{B}^{(\ell-1)}, \mathbf{W}^{(\ell)}\mathbf{C}^{(\ell-1)}).$$

for $\lambda \in \{(3), (2,1)\}$. Compute these terms using (16) applied to the cached values of the $\mathbf{W}^{(\ell)}$-transformed previous-layer factors from Step (b) i.

ii. Compute the $(1,1,1)$ slice in factored form. Explicitly,

$$\begin{aligned}
\hat{\mathbf{A}}^{(\ell)} &:= \begin{bmatrix} \mathbf{A}^{(\ell)}_{\text{two},1} & \mathbf{A}^{(\ell)}_{\text{two},2} & \mathbf{A}^{(\ell)}_{(1,1,1)} & \mathbf{A}^{(\ell)}_{(2,1,1),1} & \mathbf{A}^{(\ell)}_{(2,1,1),2} \end{bmatrix}\\
\hat{\mathbf{B}}^{(\ell)} &:= \begin{bmatrix} \mathbf{B}^{(\ell)}_{\text{two},1} & \mathbf{B}^{(\ell)}_{\text{two},2} & \mathbf{B}^{(\ell)}_{(1,1,1)} & \mathbf{B}^{(\ell)}_{(2,1,1),1} & \mathbf{B}^{(\ell)}_{(2,1,1),2} \end{bmatrix}\\
\hat{\mathbf{C}}^{(\ell)} &:= \begin{bmatrix} \mathbf{C}^{(\ell)}_{\text{two},1} & \mathbf{C}^{(\ell)}_{\text{two},2} & \mathbf{C}^{(\ell)}_{(1,1,1)} & \mathbf{C}^{(\ell)}_{(2,1,1),1} & \mathbf{C}^{(\ell)}_{(2,1,1),2} \end{bmatrix}
\end{aligned}$$

with the factors defined as in (18), 19, 20. (The $(1,1,1)$ factors make use of the $\mathbf{W}^{(\ell)}$-transformed previous-layer factors from Step (b) i.) Then

$$\tilde{\mathfrak{D}}_{(1,1,1)}(\tilde{\kappa}^{\mathsf{P}}[X^{(\ell)}])_{i_1,i_2,i_3} = \mathbb{1}[\text{all } i_j \text{ distinct}] F(\hat{\mathbf{A}}^{(\ell)}, \hat{\mathbf{B}}^{(\ell)}, \hat{\mathbf{C}}^{(\ell)})_{i_1,i_2,i_3},$$

though we never explicitly evaluate this expression for the full tensor $\tilde{\mathfrak{D}}_{(1,1,1)}(\tilde{\kappa}^{\mathsf{P}}[X^{(\ell)}])$. Note that this step adds $4n$ factors to those propagated from the previous layer.

(c) For $r \in [4]$ and $\lambda \vdash r$ with no more than 2 blocks and excluding $(3,1)$, compute $\tilde{\mathfrak{D}}_\lambda \tilde{\kappa}_r[X^{(\ell)}]$ using (S.3.7) as in Step 2c of the basic algorithm.

(d) Form the $\tilde{\eta}_r[X^{(\ell)}]$ tensors:

i. For $r \in \{1,2,4\}$, compute $\eta_r[X^{(\ell)}]$ as in Step 2d of the basic algorithm, using $\tilde{\mathfrak{D}}_\lambda \tilde{\kappa}_r[X^{(\ell)}]$ computed in Step (c).

ii. On the $(1,1,1)$-diagonal, cumulants and power cumulants coincide. Thus the factorization $F(\hat{\mathbf{A}}^{(\ell)}, \hat{\mathbf{B}}^{(\ell)}, \hat{\mathbf{C}}^{(\ell)})$ computed in Step (b) ii agrees with $\eta_3[X^{(\ell)}]$ on the $(1,1,1)$-diagonal, but we need to subtract out the other λ-diagonals and replace them with the values from Step (c). That is, for $\lambda \in \{(3,), (2,1)\}$, define the diagonal adjustment

$$\mathbf{D}_\lambda := \tilde{\mathfrak{D}}_\lambda \tilde{\kappa}_r[X^{(\ell)}] - \tilde{\mathfrak{D}}_\lambda F(\hat{\mathbf{A}}^{(\ell)}, \hat{\mathbf{B}}^{(\ell)}, \hat{\mathbf{C}}^{(\ell)})$$

where the first term on the right hand side was computed in Step (c) and the second is computed using (16). Factor $\mathbf{D}_\lambda$ using (17):

$$\mathbf{D}_\lambda = F(\mathbf{A}^{(\ell)}_\lambda, \mathbf{B}^{(\ell)}_\lambda, \mathbf{C}^{(\ell)}_\lambda) \qquad \forall \lambda \in \{(3), (2,1)\}$$

then append the factored diagonals to the existing factors:

$$\mathbf{A}^{(\ell)} := \begin{bmatrix} \hat{\mathbf{A}}^{(\ell)} & \mathbf{A}^{(\ell)}_{(3)} & \mathbf{A}^{(\ell)}_{(2,1)} \end{bmatrix},$$
$$\mathbf{B}^{(\ell)} := \begin{bmatrix} \hat{\mathbf{B}}^{(\ell)} & \mathbf{B}^{(\ell)}_{(3)} & \mathbf{B}^{(\ell)}_{(2,1)} \end{bmatrix},$$
$$\mathbf{C}^{(\ell)} := \begin{bmatrix} \hat{\mathbf{C}}^{(\ell)} & \mathbf{C}^{(\ell)}_{(3)} & \mathbf{C}^{(\ell)}_{(2,1)\cdot} \end{bmatrix}$$

We thus obtain factors $\mathbf{A}^{(\ell)}, \mathbf{B}^{(\ell)}, \mathbf{C}^{(\ell)}$ satisfying

$$\tilde{\eta}_3[X^{(\ell)}] = F(\mathbf{A}^{(\ell)}, \mathbf{B}^{(\ell)}, \mathbf{C}^{(\ell)}),$$

The diagonal adjustment adds another $2n$ factors. In total, $J^{(\ell)} = J^{(\ell-1)} + 6n$.

The number of factors at layer ℓ is then $J^{(\ell)} = 6n\ell$. The leading-order in time computation at each layer is the propagation of cumulant factors through the linearity (Step (b) i), which takes time $O(n^2 J^{(\ell-1)}) = O(n^3\ell)$. Thus, the time complexity for an L-layer MLP is $O(n^3 L^2)$, which is an improvement in n-dependence but worsening in L-dependence compared to the basic algorithm's $O(n^4 L)$.

S.5 Proofs for Hermite-based algorithms with polynomial activation functions

In this Section we will prove Theorem 5.1, using the basic algorithm described in Section S.4.1. Note that we still assume that the activation functions $\phi_1, ..., \phi_L$ are all polynomials.

S.5.1 Sizes of true cumulants

The results of Section S.2.5 were independent of the algorithm, depending only on the underlying MLP with polynomial activations, and so hold again. We restate the result for convenience:

Proposition S.2.12 (Cumulants growth bounds)**.** *Suppose that D is a fully-contracted connected tensor diagram such that for some $\ell \in [L+1]$, either*

- *every tensor sequence for $v \in V_1$ is $\kappa_{\deg(v)}[X^{(\ell-1)}]$; or*
- *every tensor sequence for $v \in V_1$ is $\kappa_{\deg(v)}[Z^{(\ell)}]$.*

Then, D has n-growth.

S.5.2 Error analysis

As before, define

$$\Delta_r[X^{(\ell)}] := \kappa_r[X^{(\ell)}] - \tilde{\kappa}_r[X^{(\ell)}]$$
$$\Delta_r[Z^{(\ell)}] := \kappa_r[Z^{(\ell)}] - \tilde{\kappa}_r[Z^{(\ell)}]$$

To analyze the basic algorithm, we must understand the error incurred by truncating the Hermite-based diagram summation formula for cumulants (see step (2b) of the description of the basic algorithm in Section S.4.1), as well as the asymptotic behavior of the Hermite coefficients. This can be easily done for polynomial activation functions (see Proposition S.5.3), and we believe it to be possible for quite general activation functions (see the discussion in Section S.6). For now, we formalize this analytic requirement in the following definition.

Definition S.5.1 (Tameness)**.** Let $(Z_n)_{n\geq 1}$ be a sequence of $\mathbb{R}^n$-valued random variables. We say that a function $\phi : \mathbb{R} \to \mathbb{R}$ is *tame with respect to* $(Z_n)_{n\geq 1}$ if:

1. for all $k_0 \geq 0$, $r \geq 1$ and $\underline{\alpha} \in \mathbb{Z}^r_{\geq 1}$, the tensor sequence

$$R^{\mathsf{P}}_{\underline{\alpha},k_0}[\phi(Z)]_{j_1,...,j_r} := 1(j_1, ..., j_r \text{ distinct})$$
$$\cdot \left(\kappa^{\mathsf{P}}_{\underline{\alpha}}[\phi(Z)] - \sum_{\substack{\underline{k}\in\mathbb{Z}^r_{\geq 0} \\ |\underline{k}|\leq k_0}} \frac{1}{\underline{k}!} \prod_{a=1}^{r} \left(\widehat{\phi^{\alpha_a}}^{(\mathbb{E}[Z],\mathrm{Var}[Z])}_{k_a} \right)_{j_a} \sum_{\pi\in \mathrm{cDia}_{[\leq 2]}(\underline{k})} \prod_{S\in\pi} \kappa\left[Z_{j_v} : (v,w) \in S\right] \right)$$

has $n^{-\frac{k_0+1}{4}}$-growth.

2. For any $\alpha, k \geq 0$, the rank-1 tensor sequences $\widehat{\phi^\alpha}_k^{(\mathbb{E}[Z],\mathrm{Var}[Z])}$ have $O(1)$-growth.
3. If $\mu, \sigma^2 \in \mathbb{R}^n$ are rank-1 tensor sequences such that $\mathbb{E}[Z] - \mu$ and $\mathrm{Var}[Z] - \sigma^2$ have n^{-k_0}-growth, then
$$\Delta\widehat{\phi^\alpha}_k := \widehat{\phi^\alpha}_k^{(\mathbb{E}[Z],\mathrm{Var}[Z])} - \widehat{\phi^\alpha}_k^{(\mu,\sigma^2)}$$
has n^{-k_0}-growth.

We prove that polynomials are tame with respect to $Z^{(\ell)}$ in Section S.5.3. Our main result is the following theorem, which does not assume that the activation functions are polynomials:

Theorem S.5.2. *Suppose that we use the basic algorithm to estimate cumulants of an MLP. Assume that*

1. *for all $\ell \in [L]$, ϕ_ℓ is tame with respect to $Z^{(\ell)}$; and*
2. *for all $\ell \in [L+1]$, the conclusion of Proposition S.2.12 holds: connected fully-contracted diagrams of (pre-)activation cumulants have n-growth.*

Then, for all $r \geq 1$ and $\ell \in [L+1]$, the tensor sequence $\Delta_r[Z^{(\ell)}]$ has $n^{-(K/2)}$ growth.

As stated, we are focusing just on the basic algorithm (the error of the factorized algorithm is precisely the same, the only difference is in the run-time, which is discussed in Section S.4.3, and the extra harmonic components computed in the augmented algorithm are dealt with using identical ideas to those presented here).

Proof of Theorem S.5.2. We prove this by induction on ℓ. We can define $Z^{(0)} := X^{(0)}$ and $\phi_0 = \mathrm{id}$ to make the base case $\ell = 0$ which is trivial, since the error tensors are all identically 0. For simplicity, we will just deal with the case $r \neq K+1 \neq 2$. The other cases are dealt with in exactly the same way, with some additional notational difficulty due to the harmonic projections.

Assume that for some $\ell \in 0, ..., L-1$, the conclusion holds that for all $r \geq 1$, the tensor $\Delta_r[Z^{(\ell)}]$ has $n^{-(K/2)}$-growth. We wish to prove the result for the errors $\Delta_d[Z^{(\ell+1)}]$. Let $p \geq 2$ be an even integer (the exponent in the definition of tensor growth).

For $r \geq K+2$, note that the basic algorithm assigns $\tilde{\kappa}_r[Z^{(\ell+1)}] = 0$, so it suffices to prove that the tensor $\kappa_r[Z^{(\ell+1)}]$ has $n^{-K/2}$-growth. This follows precisely from Corollary S.2.14. For $1 \leq r \leq K$, we simply have
$$\Delta_r[Z^{(\ell+1)}] = \Delta_r[X^{(\ell)}] \odot \mathbf{W}^{(\ell+1)}.$$
Therefore, by Proposition S.2.9,
$$\mathbb{E}\left[\Delta_r[Z^{(\ell+1)}]^p_{i_1,\ldots,i_r}\right] = \left(\frac{2}{n}\right)^{pr/2} \sum_{\pi \in \mathcal{P}_2([p]\times[r])} \mathbb{E}[D_{\pi,1}](D_{\pi,2})_{i_1,\ldots,i_r}$$
where the diagrams $D_{\pi,1}$ are now fully-contracted tensor diagrams with p copies of $\Delta_r[X^{(\ell)}]$, and the value of $D_{\pi,2}$ is always 0 or 1. We must prove that each of the diagrams $D_{\pi,1}$ has $n^{p(r-K)/2}$-growth.

Error of power cumulants:

The first step is to show that $\Delta^{\mathsf{P}}_{\underline{\alpha}}[X^{(\ell)}]$ has $n^{-K/2}$-growth, where this tensor is defined in the obvious way. By tameness of $\phi^{(\ell)}$, for any $j_1, ..., j_r$ distinct,
$$\kappa^{\mathsf{P}}_{\underline{\alpha}}[X^{(\ell)}] - \sum_{\substack{\underline{k}\in\mathbb{Z}^r_{\geq 0} \\ |\underline{k}|\leq 2K-1}} \frac{1}{\underline{k}!}\left(\prod_{a=1}^r \widehat{\phi^{\alpha_a}_\ell}_{k_a}^{\left(\mathbb{E}\left[Z^{(\ell)}_{j_a}\right],\mathrm{Var}\left[Z^{(\ell)}_{j_a}\right]\right)}\right) \sum_{\pi\in \mathrm{cDia}_{[\leq 2]}(\underline{k})} \prod_{S\in\pi} \kappa\left[Z^{(\ell)}_{j_v} : (v,w)\in S\right]$$
has $n^{-K/2}$-growth, and therefore to prove that the power cumulants have $n^{-K/2}$-growth, it suffices to prove this for the difference
$$\prod_{a=1}^r \widehat{\phi^{\alpha_a}_\ell}_{k_a}^{\left(\mathbb{E}\left[Z^{(\ell)}_{j_a}\right],\mathrm{Var}\left[Z^{(\ell)}_{j_a}\right]\right)} \prod_{S\in\pi} \kappa\left[Z^{(\ell)}_{j_v} : (v,w)\in S\right] - \prod_{a=1}^r \widehat{\phi^{\alpha_a}_\ell}_{k_a}^{\left(\tilde{\kappa}_1\left[Z^{(\ell)}_{j_a}\right],\tilde{\kappa}_2\left[Z^{(\ell)}\right]_{j_a,j_a}\right)} \prod_{S\in\pi} \tilde{\kappa}\left[Z^{(\ell)}_{j_v} : (v,w)\in S\right]$$

for any $\underline{k} \in \mathbb{Z}^r_{\geq 0}$ with $|\underline{k}| \leq 2K-1$, and $\pi \in \mathrm{cDia}_{[\leq 2]}(\underline{k})$ satisfying $k(\pi) \leq K$ where

$$k(\pi) := 1 + |\underline{k}| - |\pi| - \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\}$$

and for

$$\prod_{a=1}^{r} \widehat{\phi_\ell^{\alpha_a}}_{k_a}^{\left(\mathbb{E}\left[Z_{j_a}^{(\ell)}\right], \mathrm{Var}\left[Z_{j_a}^{(\ell)}\right]\right)} \prod_{S \in \pi} \kappa\left[Z_{j_v}^{(\ell)} : (v,w) \in S\right]$$

for any $\underline{k} \in \mathbb{Z}^r_{\geq 0}$ with $|\underline{k}| \leq 2K-1$, and $\pi \in \mathrm{cDia}_{[\leq 2]}(\underline{k})$ satisfying $k(\pi) > K$. The second case is simplest so we deal with that first. Let $p' \geq 2$ be even. Taking the p'-th power of the product of cumulants and then taking the expected value, Proposition S.2.9 tells us that we get a power $n^{-\frac{1}{2}p'|\underline{k}|}$ in front of a sum of fully-contracted diagrams of cumulants of $X^{(\ell-1)}$, and we need to compute the number of connected components. Any cumulant corresponding to $S \in \pi$ which has an odd intersection with some $[k_v]$ must be connected to some other cumulant, therefore the maximal number of connected components is

$$\frac{p'}{2}\left(|\pi| + \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\}\right)$$

Consequently (by assumption (2)), we see that the growth is

$$n^{\frac{p'}{2}(|\pi| + \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\} - |\underline{k}|)} = n^{\frac{p'}{2}(1-k(\pi))} = O(n^{-\frac{Kp'}{2}}),$$

as required, provided that the product of Hermite coefficients has n^0-growth, which is indeed condition (2) of tameness.

For the first case, we use the binomial theorem to expand the product after writing $\tilde{\phi} = \phi - \Delta\phi$ and $\tilde{\kappa} = \kappa - \Delta$, and so we can look simply at a term of the form

$$\prod_{a \notin T} \widehat{\phi_\ell^{\alpha_a}}_{k_a}^{\left(\mathbb{E}\left[Z_{j_a}^{(\ell)}\right], \mathrm{Var}\left[Z_{j_a}^{(\ell)}\right]\right)} \prod_{S \notin \pi'} \kappa\left[Z_{j_v}^{(\ell)} : (v,w) \in S\right] \prod_{a \in T} \widehat{\Delta\phi_\ell^{\alpha_a}}_{k_a} \prod_{S \in \pi'} \Delta\left[Z_{j_v}^{(\ell)} : (v,w) \in S\right]$$

where $T \subseteq [r], \pi' \subseteq \pi$ and at least one is non-empty. We know that the Hermite terms are n^0-growth, and the error terms have $n^{-K/2}$-growth (for the Hermite terms, this is condition (3) of tameness). The cumulant term has growth $\#\{S \notin \pi'\} + \#\{S \notin \pi' : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\}$. Since π is $[\leq 2]$-mixed, any S in the first group has size at least 2, and any S in the second group has size at least 4, therefore,

$$\#\{S \notin \pi'\} + \#\{S \notin \pi' : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\} \leq \frac{1}{2}\sum_{S \notin \pi'} |S|.$$

Applying Proposition S.2.9 (which holds by assumption (2)) shows that the product over $S \notin \pi'$ has $O(1)$-growth, and this term is complete.

Reduction to power cumulants: By applying Proposition S.3.7 to both $\kappa_r[X^{(\ell)}]$ and $\tilde{\kappa}_r[X^{(\ell)}]$, we see that $\Delta_r[X^{(\ell)}]$ can be replaced by a sum of connected diagrams including the power cumulant errors $\Delta^{\mathsf{P}}_\alpha[X^{(\ell)}]$ and the cumulants $\kappa_{r'}[X^{(\ell)}]$. Replacing all p copies of $\Delta_r[X^{(\ell)}]$ with this sum, we get a collection of diagrams which have the following form:

- The left vertices V_1 partition into s parts $V_{1,1}, ..., V_{1,p}$, each of which contains at least one Δ^P power cumulant.
- There are at most $pr/2$ indices in V_2.
- Each index connects to at most two parts $V_{1,i}$.
- Each part $V_{1,i}$ together with its neighbors forms a connected subgraph.

Again, we are required to prove that these diagrams have $n^{p(r-K)/2}$-growth. By Cauchy-Schwartz, Lemma S.2.11, we can relate the square of diagrams to a diagram that contains Δ's and the diagram that contains κ's. The κ part will have at most $pr/2$ connected components, and so will have $n^{pr/2}$-growth (by assumption (2)), which means that we are required to prove that the Δ-diagram has $n^{p(\frac{1}{2}r-K)}$-growth. They are fully contracted tensor diagrams with at least $2p$ tensors, each with $n^{-K/2}$-growth, and at most $pr/2$ indices, proving the result. $\square$

S.5.3 Proof of tameness for polynomial activations

Proposition S.5.3. *Polynomials are tame. More precisely, if Z is the output distribution of an MLP with polynomial activation functions, then any polynomial ϕ is tame with respect to Z.*

Proof. First, we prove condition (2) of the tameness condition. Let $\alpha, k \geq 0$, then

$$(\mu, \sigma^2) \mapsto \widehat{\phi^{\alpha}}_k^{(\mu,\sigma^2)}$$

is a polynomial map. We can consider each monomial individually, for example $\mu^a(\sigma^2)^b$ for $a, b \geq 1$. We get, for $p \geq 2$ even, by Proposition S.2.9,

$$\mathbb{E}\left[\left(\kappa_1[Z^{(\ell)}]_i^a \kappa_2[Z^{(\ell)}]_{i,i}^b\right)^p\right] = \left(\frac{2}{n}\right)^{\frac{1}{2}p(a+2b)} \sum_{\pi} D_\pi$$

where D_π is the diagram formed from pairing pa copies of κ_1 and pb copies of κ_2. The most connected components we can produce is $\frac{1}{2}pa + pb$, and so this term is $O(1)$, completing the proof of (2).

Next, we prove condition (3). By expanding the Hermite coefficient into monomials, and using the binomial theorem to express every term as a monomial in $\Delta\mu_i, (\Delta\kappa_2)_{i,i}, \kappa_1[Z^{(\ell)}]$ and $\kappa_2[Z^{(\ell)}]_{i,i}$, we have to prove that if $a, b, c, d \geq 0$ with $a + b \geq 0$, then

$$(\Delta\mu_i)^a (\Delta\kappa_2)_{i,i}^b \kappa_1[Z^{(\ell)}]_i^c \kappa_2[Z^{(\ell)}]_{i,i}^d$$

has $n^{-K/2}$-growth. Since, by assumption, the Δ vectors have $n^{-K/2}$-growth, we are done by applying the already proven part (2) of tameness for polynomials.

Now we prove condition (1) of tameness for polynomials. Only a finite number of Hermite coefficients are non-zero, so we see that for any $\underline{\alpha}, k_0, \ell$, the quantity $R^{\mathsf{P}}_{\underline{\alpha},k_0}[\phi(Z^{(\ell)})]$ is a sum of terms of the form

$$\frac{1}{\underline{k}!}\left(\prod_{a=1}^{r} \widehat{\phi^{\alpha_a}}_{k_a}^{(\mathbb{E}[Z],\mathrm{Var}[Z])}\right) \prod_{S\in\pi} \kappa[Z_{j_v} : (v,w) \in S] \tag{21}$$

for a connected $[\leq 2]$-mixed partition π of $\{(v,w) : v \in \{1,...,m\}, w \in \{1,...,k_v\}\}$ for some $\underline{k} \in \mathbb{Z}^m_{\geq 0}$ with $|\underline{k}| > k_0$. It suffices to prove that the term above has $n^{-\frac{k_0+1}{4}}$-growth. The Hermite product has n^0-growth by condition (2).

Let $p' \geq 2$ be even. Taking the p'-th power of the product of cumulants and then taking the expected value, Proposition S.2.9 tells us that we get a power $n^{-\frac{1}{2}p'|\underline{k}|}$ in front of a sum of fully-contracted diagrams of cumulants of $X^{(\ell-1)}$, and we need to compute the number of connected components. Any cumulant corresponding to $S \in \pi$ which has an odd intersection with some $[k_v]$ must be connected to some other cumulant, therefore the maximal number of connected components is

$$\frac{p'}{2}\left(|\pi| + \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\}\right).$$

Since π is $[\leq 2]$-mixed, any set $S \in \pi$ satisfying $\forall v, |S \cap [k_v]| \equiv 0 \bmod 2$ has size at least 4, and any other set has size at least 2. Therefore

$$|\pi| + \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\} \leq \frac{1}{2}|\underline{k}|$$

and so the growth of this diagram has exponent

$$-\frac{1}{2}|\underline{k}| + \frac{1}{2}(|\pi| + \#\{S \in \pi : \forall v, |S \cap [k_v]| \equiv 0 \bmod 2\}) \leq -\frac{1}{4}|\underline{k}| \leq -\frac{1}{4}(k_0 + 1)$$

as required. □

Corollary S.5.4. *Theorem 5.1 is satisfied by the basic algorithm.*

Proof. By Propositions S.5.3 and S.2.12, the conditions of Theorem S.5.2 are satisfied when the ϕ_ℓ are all polynomials. □

In Section S.8.5, we will describe the general form of the factorized algorithm, and show that it computes the same estimates as the basic algorithm, while saving a factor of n in the runtime for all $K \geq 3$. This will prove that Theorem 5.2 is satisfied by the factorized algorithm.

S.6 Discussion of Hermite-based algorithms with non-polynomial activation functions

In the previous section, Theorem S.5.2 did not assume that the activation functions are polynomials. For polynomials, we proved assumption (1) in Proposition S.5.3 and assumption (2) in Proposition S.2.12, thus completing the proof of Theorem 5.1, the main theoretical result of the paper. We now discuss how we might go about proving assumptions (1) and (2) in the non-polynomial setting.

S.6.1 Cumulant growth

We start with assumption (2), which we expect to follow from tameness. We will try to prove this by induction. The first step is independent of the activation function and follows exactly as in the proof of Proposition S.2.12.

Lemma S.6.1. *Suppose that for some $\ell \in [L+1]$, every connected fully-contracted tensor diagram of cumulants of $X^{(\ell-1)}$ has n-growth. Then the same holds for connected fully-contracted tensor diagrams of cumulants of $Z^{(\ell)}$.*

Proof. Consider the diagram D: taking the p-th power of this quantity corresponds to taking the disjoint sum of p copies of this diagram. By Proposition S.2.9, taking the expectation over $\mathbf{W}^{(\ell)}$ produces a sum

$$\left(\frac{2}{n}\right)^{p|E|/2} \sum_{\pi \text{ pairing of } pE(B)} D_{\pi,1} D_{\pi,2}$$

where the number of connected components of $D_{\pi,1}D_{\pi,2}$ is at most $p(1+\frac{1}{2}|E|)$. Each of the connected components of $D_{\pi,1}$ is a connected fully-contracted tensor diagram of cumulants of $X^{(\ell-1)}$ so contributes a factor of $O(n)$, and each of the connected component of $D_{\pi,2}$ is a connected fully-contracted tensor diagram of I_2's, which also contribute a factor of n. Therefore we get that the whole diagram is $O(n^p)$ as required. □

The second part of the induction depends on ϕ_ℓ. Let us first replace each $\kappa_r[X^{(\ell)}]$ with a sum of connected contractions of power cumulants $\kappa^{\mathsf{P}}_{\underline{\alpha}}[X^{(\ell)}]$. Suppose that ϕ_ℓ is tame, then we can pick $k_0 \geq 0$ and replace $\kappa^{\mathsf{P}}_{\underline{\alpha}}[X^{(\ell)}]$ using part (1) of the definition of tameness, which will produce a sum of diagrams containing tensors $R^{\mathsf{P}}_{\underline{\alpha},k_0}[\phi_\ell(Z^{(\ell)})]$, Hermite coefficient vectors, and cumulants $\kappa[Z^{(\ell)}]$. Consider the r-rank tensor

$$\prod_{a=1}^{r} \left(\widehat{\phi^{\alpha_a}}^{(\mathbb{E}[Z],\mathrm{Var}[Z])}_{k_a}\right)_{j_a} \prod_{S\in\pi} \kappa[Z_{j_v} : (v,w) \in S]$$

The cumulant part has growth at most $n^{-\frac{1}{4}|\underline{k}|}$, and so by Cauchy-Schwartz and condition (2) of tameness, this rank-r tensor has growth at most $n^{\frac{1}{2}(r-|\underline{k}|)}$. Since this is bounded above independently of k_0, we can choose k_0 large enough that any diagram containing a residual $R^{\mathsf{P}}_{\underline{\alpha},k_0}$ has at most n-growth. It remains to deal with the terms involving just Hermite coefficient vectors and cumulants $\kappa[Z^{(\ell)}]$.

These terms look like connected fully-contracted tensor diagrams of $\kappa[Z^{(\ell)}]$'s, except each index in the summation is attached to a product of Hermite coefficients, each of which is $O(1)$. If all the Hermite coefficients are a rank-1 tensor diagram in the cumulants $\kappa_r[Z^{(\ell)}]$ as is the case for polynomial activation functions, we would have a complete proof. We expect that for tame activation functions, the well-behaved nature of the Hermite coefficients should be sufficient to prove the same result, however we have not been able to provide a comprehensive proof of this fact.

S.6.2 Tameness

Tameness provides a much more significant analytic challenge. Here we simply give an informal discussion on how one might formally prove tameness.

Conditions (2) and (3) roughly correspond to a bounded, Lipschitz property of the Hermite coefficients of ϕ around $(\mathbb{E}[Z], \mathrm{Var}[Z])$. For example, it is easy to prove that

$$\mathbb{E}\left[\left(\kappa_2[Z^{(\ell)}]_{i,i} - \frac{1}{n}\mathrm{Tr}\kappa_2[Z^{(\ell)}]\right)^2\right] = 8n^{-4}(n-1)^2\|\kappa_2[X^{(\ell)}]\|_F^2 \tag{22}$$

which has n^{-1}-growth. Provided the activation functions are normalized correctly to keep $\mathrm{Tr}\kappa_2[Z^{(\ell)}]$ constant, this suggests that as $n \to \infty$ the variances will concentrate, and then provided the Hermite coefficients are well-behaved in a rectangle $(-C, C) \times (1-\epsilon, 1+\epsilon)$, the properties (2) and (3) should follow. This is simply a sketch, since we need significantly better control on the variances than (22). For example, the Hermite coefficients of ReLU have a singularity at $\sigma^2 = 0$, so much more care is required (see Section S.3.4).

Condition (1) of tameness is a statement concerning the validity of Theorem A.2, the diagram summation formula for cumulants, when extended to non-polynomial functions ϕ. We do not, in fact, expect this infinite summation formula to converge at any fixed width n, but instead that a fixed truncation should be an asymptotically good approximation as $n \to \infty$. This is the property of an asymptotic expansion, and in fact we can motivate the diagram summation formula through the theory of Edgeworth expansions, a special form of asymptotic series [Hall, 1992]. Note that in the proof of Proposition S.5.3, we proved exactly that every term of the diagram summation formula which remains in $R^{\mathsf{P}}_{\underline{\alpha},k_0}$ in condition (1) of tameness has $n^{-\frac{k_0+1}{4}}$-growth. For polynomials, there are only finitely many terms, which completes the proof, but for general activation functions the definition of tameness simply says that the asymptotic growth of the error of truncation is no larger than any particular term that has been removed. We expect the theory of Edgeworth expansions in Hall [1992] to apply to our situation, since the application of the linear layer produces a collection of almost iid random variables, however we have not performed this analysis.

S.7 Proof of diagram summation formulas

In this section we prove of our diagram summation formulas. Specifically, we have the polynomial diagram summation formula, Theorem S.2.1, the Hermite-based diagram summation formula, Theorem A.2, and its combinatorial restatement, Theorem S.3.5. We begin by restating Theorems S.2.1 and A.2.

Theorem S.2.1 (Polynomial diagram summation formula for cumulants)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi : \mathbb{R} \to \mathbb{R}$ be a polynomial. Then*

$$\kappa\left[\phi\left(Z_{j_1}\right), \ldots, \phi\left(Z_{j_r}\right)\right] = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^{r} \frac{1}{k_v!}\phi^{(k_v)}(0)\right) \sum_{\pi \in \mathrm{cDia}(\underline{k})} \prod_{S \in \pi} \kappa\left[Z_{j_v} : (v,w) \in S\right].$$

Theorem A.2 (Hermite-based diagram summation formula for cumulants)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi : \mathbb{R} \to \mathbb{R}$ be a polynomial. Then*

$$\kappa\left[\phi\left(Z_{j_1}\right), \ldots, \phi\left(Z_{j_r}\right)\right] = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^{r} \frac{1}{k_v!}\widehat{\phi}_{k_v}^{(\mathbb{E}[Z_{j_v}], \mathrm{Var}[Z_{j_v}])}\right) \sum_{\pi \in \mathrm{cDia}_{[\leq 2]}(\underline{k})} \prod_{S \in \pi} \kappa\left[Z_{j_v} : (v,w) \in S\right].$$

The cumulants and Hermite coefficients used in these formulas are defined in Section 3, and the definitions of the set $\mathrm{cDia}(\underline{k})$ of all connected diagrams on $\underline{k}$ and the set $\mathrm{cDia}_{[\leq M]}(\underline{k})$ of all connected $[\leq M]$-mixed diagrams on $\underline{k}$ are given in Appendix A. Some further intuition for these formulas is given in Section S.1.4.

Both Theorem S.2.1 and Theorem A.2 assume that $\phi : \mathbb{R} \to \mathbb{R}$ is a polynomial. We make this assumption to avoid convergence issues, since it ensures that the expansions are finite sums. For non-polynomials, the formulas still hold in a formal sense, but can fail to converge. Convergence of these formal expansions is fragile: we do not expect these expansions to converge for a fixed n, but instead truncate to a finite summation such that the error converges to zero as $n \to \infty$. We discuss how this could be done using the notion of tameness in Section S.6.2.

S.7.1 Polynomial diagram summation formula for cumulants

Let $\phi(t) = \sum_{k=0}^{\infty} \frac{1}{k!} a_k t^k$ be a polynomial (so $a_k = 0, \forall k \gg 0$). Then by multi-linearity of cumulants, we can expand

$$\kappa[\phi(Z_{j_1}), ..., \phi(Z_{j_r})] = \sum_{\underline{k} \in \mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^{r} \frac{1}{k_v!} a_{k_v} \right) \kappa_r[Z_{j_1}^{k_1}, ..., Z_{j_r}^{k_r}] \tag{23}$$

We now explicitly compute the cumulants of powers using the following Lemma.

Lemma S.7.1. *Let $X_1, ..., X_r$ be $\mathbb{R}$-valued random variables, and $k_1, ..., k_r \in \mathbb{Z}_{\geq 0}$. Then*

$$\kappa\left[Z_{j_1}^{k_1}, ..., Z_{j_r}^{k_r}\right] = \sum_{\pi \in \mathrm{cDia}(\underline{k})} \prod_{S \in \pi} \kappa\left[Z_{j_v} : (v,l) \in S\right] \tag{24}$$

where we say $\pi \vdash [k_1] \sqcup ... \sqcup [k_r]$ is connected if $\pi \vee \{[k_1], ..., [k_r]\}$ has a single block.

Proof. Consider the following two ways of expanding

$$\mathbb{E}\left[\prod_{v=1}^{r} Z_{j_v}^{k_v}\right]$$

with the defining formula for cumulants. Firstly, consider this as an expectation of a product of r random variables $Z_{j_1}^{k_1}, ..., Z_{j_r}^{k_r}$, giving

$$\mathbb{E}\left[\prod_{v=1}^{r} Z_{j_v}^{k_v}\right] = \sum_{\pi \vdash [r]} \prod_{S \in \pi} \kappa\left[Z_{j_v}^{k_v} : v \in S\right]$$

Secondly, consider it as an expectation of a product of $k_1 + ... + k_r$ random variables, and we get

$$\begin{aligned}\mathbb{E}\left[\prod_{v=1}^{r} Z_{j_v}^{k_v}\right] &= \sum_{\tau \in \mathrm{Dia}(\underline{k})} \prod_{T \in \tau} \kappa\left[Z_{j_v} : (v,l) \in T\right] \\ &= \sum_{\pi \vdash [r]} \prod_{S \in \pi} \sum_{\tau \in \mathrm{cDia}(\underline{k}_S)} \prod_{T \in \tau} \kappa\left[Z_{j_v} : (v,l) \in T\right]\end{aligned}$$

Here $\underline{k}_S := (k_v : v \in S) \in \mathbb{Z}_{\geq 0}^{|S|}$. In these decompositions, uniqueness implies that

$$\kappa\left[Z_{j_v}^{k_v} : v \in S\right] = \sum_{\tau \in \mathrm{cDia}(\underline{k_S})} \prod_{T \in \tau_S} \kappa\left[Z_{j_v} : (v,l) \in T\right]$$

and in particular the required result follows (with $S = [r]$). □

Substituting this result into (23) completes the proof of Theorem S.2.1.

S.7.2 Hermite coefficients of polynomials

Recall from Section 3 the definition of the Hermite coefficients: for $(\mu, \sigma) \in \mathbb{R} \times \mathbb{R}_{>0}$ and $Y \sim \mathcal{N}(\mu, \sigma^2)$,

$$\widehat{\phi}_s^{(\mu,\sigma^2)} := \frac{1}{\sigma^s} \mathbb{E}\left[\phi(Y) \mathrm{He}_s\left(\frac{Y-\mu}{\sigma}\right)\right].$$

For a polynomial, we get a simple expression.

Lemma S.7.2. *Let $\phi(t) = \sum_{k=0}^{\infty} \frac{1}{k!} a_k t^k$ be a polynomial. Then*

$$\widehat{\phi}_s^{(\mu,\sigma^2)} = \sum_{k=0}^{\infty} \frac{1}{k!} a_{s+k} \mathbb{E}[Y^k]$$

Proof. The defining equation for the probabilist's Hermite polynomial is

$$\mathrm{He}_s(x) = (-1)^s e^{\frac{1}{2}x^2} \frac{d^s}{dx^s} e^{-\frac{1}{2}x^2}.$$

Therefore,

$$\begin{aligned}
\frac{1}{\sigma^s}\mathbb{E}\left[\phi(Y)\mathrm{He}_s\left(\frac{Y-\mu}{\sigma}\right)\right] &= \frac{1}{\sigma^s\sqrt{2\pi}}\int_{-\infty}^{\infty}\phi(\mu+\sigma x)\mathrm{He}_s(x)e^{-\frac{1}{2}x^2}dx \\
&= (-1)^s\frac{1}{\sigma^s\sqrt{2\pi}}\int_{-\infty}^{\infty}\phi(\mu+\sigma x)\left(\frac{d^s}{dx^s}e^{-\frac{1}{2}x^2}\right)dx \\
&= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\phi^{(s)}(\mu+\sigma x)e^{-\frac{1}{2}x^2}dx \\
&= \mathbb{E}\left[\phi^{(s)}(Y)\right] \\
&= \sum_{k=0}^{\infty}\frac{1}{k!}a_{s+k}\mathbb{E}[Y^k] \qquad \square
\end{aligned}$$

The terms $\mathbb{E}[Y^k]$ can be computed using the defining formula for cumulants, using the fact that the only non-zero cumulants for a Gaussian are κ_1 and κ_2, so

$$\mathbb{E}[Y^k] = \sum_{\substack{\pi\vdash[k] \\ \forall S\in\pi, |S|\leq 2}} \prod_{S\in\pi}\kappa_{|S|}[Y]. \tag{25}$$

S.7.3 Hermite-based diagram summation formula for cumulants

We start at the polynomial diagram summation formula, for a polynomial $\phi(t) = \sum_{k=0}^{\infty}\frac{1}{k!}a_k t^k$,

$$\kappa\left[\phi\left(Z_{j_1}\right),\ldots,\phi\left(Z_{j_r}\right)\right] = \sum_{\underline{k}\in\mathbb{Z}^r_{\geq 0}}\left(\prod_{v=1}^{r}\frac{1}{k_v!}a_{k_v}\right)\sum_{\pi\in\mathrm{cDia}(\underline{k})}\prod_{S\in\pi}\kappa\left[Z_{j_v}:(v,w)\in S\right].$$

We now divide the diagram $\pi\in\mathrm{cDia}(\underline{k})$ into $r+1$ diagrams,

- $\pi'\in\mathrm{cDia}_{[\leq 2]}(\underline{k}')$, consisting of all $S\in\pi$ such that $|S|\geq 3$ or $\#\{v : S\cap[k_v]\neq\emptyset\} > 1$,
- for each $v\in[r]$, $\pi_v\vdash[d_v]$ consisting of all $S\in\pi$ such that $|S|\leq 2$, and $S\subseteq[k_v]$.

The pre-image of $(\pi',\pi_1,...,\pi_r)$ under the map $\pi\mapsto(\pi',\pi_1,...,\pi_r)$ has size $\prod_{v=1}^{r}\binom{k'_v+d_v}{d_v}$ since the only choice is the image $\bigcup_{S\in\pi'}S\subset[k_1]\sqcup...\sqcup[k_r]$. Therefore, we get

$$\begin{aligned}
\kappa\left[\phi\left(Z_{j_1}\right),\ldots,\phi\left(Z_{j_r}\right)\right] &= \sum_{\underline{k}'\in\mathbb{Z}^r_{\geq 0}}\prod_{v=1}^{r}\left(\sum_{d_v=0}^{\infty}\binom{k'_v+d_v}{d_v}\frac{1}{(k'_v+d_v)!}a_{k'_v+d_v}\sum_{\substack{\pi_v\vdash[d_v] \\ \forall S\in\pi_v,|S|\leq 2}}\prod_{S\in\pi_v}\kappa_{|S|}[Z_{j_v}]\right) \\
&\qquad\cdot\sum_{\pi'\in\mathrm{cDia}_{[\leq 2]}(\underline{k}')}\prod_{S\in\pi'}\kappa\left[Z_{j_v}:(v,w)\in S\right] \\
&= \sum_{\underline{k}'\in\mathbb{Z}^r_{\geq 0}}\frac{1}{k_v!}\prod_{v=1}^{r}\left(\sum_{d_v=0}^{\infty}\frac{1}{d_v!}a_{k'_v+d_v}\mathbb{E}[Y_v^k]\right)\sum_{\pi'\in\mathrm{cDia}_{[\leq 2]}(\underline{k}')}\prod_{S\in\pi'}\kappa\left[Z_{j_v}:(v,w)\in S\right]
\end{aligned}$$

where $Y_v = \mathcal{N}(\mathbb{E}[Z_{j_v}],\mathrm{Var}[Z_{j_v}])$, and we have applied (25). Comparing the summation in parentheses with Lemma S.7.2 completes the proof of Theorem A.2.

S.7.4 Proof of combinatorial restatement

Recall the combinatorial restatement, Theorem S.3.5, of Theorem A.2.

Theorem S.3.5 (Combinatorial restatement of A.2)**.** *Let $Z \in \mathbb{R}^n$ be a random variable with finite moments and let $\phi\colon \mathbb{R} \to \mathbb{R}$ be a polynomial, which we think of as acting coordinatewise on Z. Then*

$$\kappa_r[\phi(Z)]_{j_1,\dots j_r} = \sum_{\underline{k}\in\mathbb{Z}^r_{\geq 0}} \left(\prod_{v=1}^{r} \frac{1}{k_v!}\widehat{\phi}_{k_v}^{(\mathbb{E}[Z_{j_v}],\mathrm{Var}(Z_{j_v})}\right) \sum_{\nu\in \mathrm{cVec}_{[\leq 2]}(\underline{k})} c_\nu^{\mathrm{vec}} \prod_{\underline{u}\in\nu} (\tilde{\mathfrak{D}}_{\underline{u}}\kappa_{|\underline{u}|}[Z])_{j_1,\dots,j_r}. \tag{10}$$

This form of the diagram summation formula is obtained by rewriting Theorem A.2 in tensor notation, grouping diagrams by their vector partition type, and applying Proposition S.3.4, which we prove now.

Proposition S.3.4. *Fix $\nu \in \mathrm{Vec}(\underline{k})$, and for each $\underline{0} \leq \underline{u} \leq \underline{k}$ (entrywise) let $\nu(\underline{u})$ denote the number of times $\underline{u}$ appears in ν. Then, the number of diagrams with type ν is*

$$c_\nu^{\mathrm{vec}} := \frac{\prod_{i=1}^r k_i!}{\prod_{\underline{u}=\underline{0}}^{\underline{k}} \left(\nu(\underline{u})! \prod_{i=1}^r (u_i!)^{\nu(\underline{u})}\right)}.$$

Proof of Proposition S.3.4. Fix $\underline{k} \in \mathbb{Z}^r_{>0}$ and $\nu \in \mathrm{Vec}(\underline{k})$. We wish to compute

$$c_\nu^{\mathrm{vec}} := \#\{\pi \vdash \{(v,w) : v \in [r], w \in [k_v]\} : \mathrm{type}(\pi) = \nu\},$$

which is precisely the cardinality of the orbit of some π satisfying $\mathrm{type}(\pi) = \nu$ under the action of $H_{\underline{k}} := \prod_{v=1}^r S_{k_v}$ by permuting elements within each block $\{(v,w) : w \in [k_v]\}$ for $v \in [r]$. By orbit-stabilizer,

$$c_\nu^{\mathrm{vec}} = [H_{\underline{k}} : \mathrm{Stab}_{H_{\underline{k}}}(\pi)] = \frac{\prod_{v=1}^r k_v!}{|\mathrm{Stab}_{H_{\underline{k}}}(\pi)|}. \tag{26}$$

To find the cardinality of the stabilizer of π, note that any element of $H_{\underline{k}}$ that fixes π must map each block of π to a block of the same type. It may also arbitrarily permute elements within intersections of π-blocks and $\underline{k}$-blocks. Thus, letting $\nu(\underline{u})$ denote the number of times $\underline{u}$ appears in ν,

$$|\mathrm{Stab}_{H_{\underline{k}}}(\pi)| = \prod_{\underline{u}} \underbrace{\nu(\underline{u})!}_{\#\text{ ways to permute blocks of type } \underline{u}} \prod_{i=1}^{r} \underbrace{(u_i!)^{\nu(\underline{u})}}_{\#\text{ ways to permute elements of form } (i,v) \text{ in each block of type } \underline{u}},$$

where the product is over all $\underline{u}$ appearing in ν. Substituting into (26) completes the proof. □

S.8 Proofs of preliminary results

S.8.1 Proofs for harmonic decompositions

Here we prove Theorem S.2.4.

Theorem S.2.4 (Harmonic decomposition of symmetric tensors)**.** *Let $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$. Then there exist unique $h_{r-2s}(\mathbf{T}) \in \mathcal{S}^{r-2s}(\mathbb{R}^n)$ for $0 \leq s \leq \left\lfloor \frac{r}{2} \right\rfloor$ satisfying*

$$\mathrm{tr}\left(h_{r-2s}(\mathbf{T})\right) = 0, \qquad \forall 0 \leq s \leq \left\lfloor \frac{r}{2} \right\rfloor - 1$$

$$\mathbf{T} = \sum_{0\leq s\leq \lfloor \frac{r}{2}\rfloor} g^s(h_{r-2s}(\mathbf{T}))$$

In fact,

$$h_{r-2s}(\mathbf{T}) = \sum_{0\leq t\leq \lfloor \frac{r-2s}{2}\rfloor} c^{\mathrm{h}}_{s,t} g^t(\mathrm{tr}^{s+t}(\mathbf{T})) \tag{6}$$

where

$$c^{\mathrm{h}}_{s,t} = (-1)^t \frac{r + \frac{n}{2} - 2s - 1}{2^{s+t} s!t! (r + \frac{n}{2} - 2s - t - 1)_{s+t+1}}$$

with $(a)_t := a(a+1)...(a+t-1)$ (the Pochhammer symbol).

Proof. First, for a rank r tensor over $\mathbb{R}^n$, a simple induction proves that for all $k \geq 0$,

$$\mathrm{tr}(g^k(\mathbf{T})) - g^k(\mathrm{tr}(\mathbf{T})) = (2kr + kn + 2k(k-1))\, g^{k-1}(\mathbf{T}).$$

Let us solve for the coefficients: the defining equations are

$$\forall s, \sum_{0 \leq t \leq \lfloor \frac{r}{2} \rfloor - s} c^{\mathrm{h}}_{s,t} \mathrm{tr}(g^t(\mathrm{tr}^{s+t}(\mathbf{T}))) = 0$$

$$c^{\mathrm{h}}_{0,0} = 1$$

$$\forall k \geq 1, \qquad \sum_{s+t=k} c^{\mathrm{h}}_{s,t} = 0$$

Applying the commutation relation to the first of these equations results in

$$\sum_{1 \leq t \leq \lfloor \frac{r}{2} \rfloor - s} \left(2tc^{\mathrm{h}}_{s,t}\left(r - 2s - t + \frac{n}{2} - 1\right) + c^{\mathrm{h}}_{s,t-1}\right) g^{t-1}\mathrm{tr}^{s+t}(\mathbf{T}) = 0.$$

The fact that the coefficients here must be zero for this to hold for arbitrary $\mathbf{T}$ means that, by recursion,

$$c^{\mathrm{h}}_{s,t} = (-1)^t \frac{(r - 2s + \frac{n}{2} - 2 - t)!}{2^t t!(r - 2s + \frac{n}{2} - 2)!} c^{\mathrm{h}}_{s,0}$$

Now we simply need to figure out the components, $c^{\mathrm{h}}_{s,0}$, which satisfy the third condition,

$$\sum_{s+t=k} (-1)^j \frac{(r - 2s + \frac{n}{2} - 2 - t)!}{2^t t!(r - 2s + \frac{n}{2} - 2)!} c^{\mathrm{h}}_{s,0} = 0.$$

This allows us to recursively find the base coefficients, and it can be verified that

$$c^{\mathrm{h}}_{s,0} = \frac{(r + \frac{n}{2} - 2s - 1)!}{2^s s!(r + \frac{n}{2} - s - 1)!}.$$

Therefore,

$$c^{\mathrm{h}}_{s,t} = (-1)^t \frac{r + \frac{n}{2} - 2s - 1}{2^{s+t} s! t! (r + \frac{n}{2} - 2s - t - 1)_{s+t+1}}.$$ □

S.8.2 Diagram expectations

Here we prove Proposition S.2.9.

Proposition S.2.9 (Diagram expectations)**.** *Let $D = (V_1, V_2, E, I, \{T_v\}_{v \in V_1})$ be a rank r tensor diagram at layer ℓ. If $|E|$ is odd, then the rank r tensor sequence $\mathbb{E}_{\mathbf{W}^{(\ell+1)}}[D \odot \mathbf{W}^{(\ell+1)}]$ is identically zero. If $|E|$ is even,*

$$\mathbb{E}_{\mathbf{W}^{(\ell+1)}}[D \odot \mathbf{W}^{(\ell+1)}] = \left(\frac{2}{n}\right)^{|E|/2} \sum_{\pi \in \mathcal{P}_2(E)} D_{\pi,1} D_{\pi,2},$$

where $D_{\pi,1}$ is a rank 0 (scalar) tensor diagram at layer ℓ constructed as $(V_1, \pi, \pi, \{(v, S) : v \in e \in S \in \pi\}, \{T_v\}_{v \in V_1})$, and $D_{\pi,2}$ is a constant rank r tensor diagram (constant meaning that the random variables are all constant) constructed as $(\pi, V_2, I, \{(S, w) : w \in e \in S \in \pi\}, (I_2)_{S \in \pi})$. Here, $\mathcal{P}_2(E)$ denotes the set of perfect pairings of E. Furthermore, the number of connected components of $D_{\pi,1} D_{\pi,2}$ is at most $|E|/2$ more than the number of connected components of D.

Proof. The key result for the proof is Isserlis's Theorem (which can in fact be seen directly from the defining property of cumulants): if $(G_1, ..., G_r)$ is a multivariate zero-mean Gaussian,

$$\mathbb{E}[G_1 ... G_r] = \sum_{\pi \in \mathcal{P}_2([r])} \prod_{(u,v) = S \in \pi} \mathbb{E}[G_u G_v].$$

Applying this to entries of $\mathbf{W}^{(\ell+1)}$, and using the relation $\mathbb{E}[\mathbf{W}^{(\ell+1)}_{i,j} \mathbf{W}^{(\ell+1)}_{i',j'}] = \frac{2}{n} \delta_{i,i'} \delta_{j,j'}$, we get

$$\mathbb{E}\left[\prod_{i=1}^{r} \mathbf{W}^{(\ell+1)}_{i_r, j_r}\right] = \left(\frac{2}{n}\right)^{r/2} \sum_{\pi \in \mathcal{P}_2([r])} \prod_{(u,v) = S \in \pi} \delta_{i_u, i_v} \delta_{j_u, j_v}.$$

Now, we apply S.2.3 and (7), and get

$$\begin{aligned}
\mathbb{E}\left[D \odot \mathbf{W}^{(\ell+1)}\right] &= \sum_{\substack{i_v=1 \\ \forall v \in I}}^{n} \mathbb{E}\left[\prod_{u \in V_1} \sum_{\substack{j_e=1 \\ \forall e=(u,v) \in E}}^{n} (T_u)_{\{j_e:(u,v)\in E\}} \prod_{v:(u,v)\in E} \mathbf{W}^{(\ell+1)}_{i_v,j_e}\right] \\
&= \sum_{\substack{i_v=1 \\ \forall v \in I}}^{n} \sum_{\substack{j_e=1 \\ \forall e \in E}}^{n} \left(\prod_{u \in V_1} (T_u)_{\{j_v:(u,v)\in E\}}\right) \mathbb{E}\left[\prod_{e=(u,v)\in E} W^{(\ell+1)}_{i_v,j_e}\right] \\
&= \left(\frac{2}{n}\right)^{|E|/2} \sum_{\pi \in \mathcal{P}_2(E)} \left(\sum_{\substack{j_S=1 \\ \forall S \in \pi}}^{n} \prod_{u \in V_1} (T_u)_{j_S:u \in e \in S \in \pi}\right) \left(\sum_{\forall v \in I, i_v=1}^{n} \prod_{S \in \pi} \prod_{(u,v)=e \in S} \delta_{i_u,i_v}\right)
\end{aligned}$$

Now we simply recognize the left bracket inside the sum as $D_{\pi,out}$ and the right bracket as $D_{\pi,in}$.

The connected components statement follows from the fact that replacing a pair of edges of E by a pair of identity matrices can increase the number of connected components by at most 1. □

We also prove Lemma S.2.11.

Lemma S.2.11 (Cauchy-Schwartz for tensor diagrams)**.** *Let $D = (V_1, v_2, E, I, \{T_v\})$ be a tensor diagram, and let $V_1 = U_1 \sqcup U_2$ be a partition of the left vertices of D into two parts. Consider the two tensor diagrams formed as follows: for each $i \in \{1, 2\}$ construct D_i by doubling the set U_i along with all its edges and tensors, formally*

- $V_1(D_i) = [2] \times U_i$
- $V_2(D_i) = V_2(D)$
- $E(D_i) = \{((j, v), w) : (v, w) \in E(D), v \in U_i\}$
- $I(D_i) = I(D)$
- $T_{(j,v)} = T_v$ *for all* $j = 1, 2, v \in U_i$.

Then all entries of D_1, D_2 are non-negative, and

$$D^2_{i_1,\ldots,i_r} \le (D_1)_{i_1,\ldots,i_r}(D_2)_{i_1,\ldots,i_r}.$$

In addition, if D_i have n^{ρ_i}-growth, then D has $n^{\frac{1}{2}(\rho_1+\rho_2)}$-growth.

Proof. To actually talk about entries of D, we need to pick an ordering of $V_2 \setminus I$, say $\sigma : [\mathrm{rank}(D)] \to V_2 \setminus I$. Then,

$$D_{i_{\sigma(1)},\ldots,i_{\sigma(r)}} = \sum_{(i_w)\in[n]^I} \prod_{v \in V_1} (T_v)_{\{i_w:(v,w)\in E\}}.$$

Splitting this product over $V_1 = U_1 \sqcup U_2$ and applying Cauchy-Schwartz (on vectors indexed by $[n]^I$) implies

$$D^2_{i_{\sigma(1)},\ldots,i_{\sigma(r)}} \le \left(\sum_{(i_w)\in[n]^I} \prod_{v \in U_1} (T_v)^2_{\{i_w:(v,w)\in E\}}\right) \left(\sum_{(i_w)\in[n]^I} \prod_{v \in U_2} (T_v)^2_{\{i_w:(v,w)\in E\}}\right)$$

These bracketed values are clearly equal to $(D_1)_{i_{\sigma(1)},\ldots,i_{\sigma(r)}}$ and $(D_2)_{i_{\sigma(1)},\ldots,i_{\sigma(r)}}$ respectively. This proves non-negativity and the inequality.

Raising this inequality to the power p and taking expectations, we can apply Cauchy-Schwartz again,

$$\mathbb{E}_{\boldsymbol{\theta}}\left[\left|D_{i_{\sigma(1)},\ldots,i_{\sigma(r)}}\right|^p\right] \le \mathbb{E}_{\boldsymbol{\theta}}\left[(D_1)^p_{i_{\sigma(1)},\ldots,i_{\sigma(r)}}\right]^{\frac{1}{2}} \mathbb{E}_{\boldsymbol{\theta}}\left[(D_2)^p_{i_{\sigma(1)},\ldots,i_{\sigma(r)}}\right]^{\frac{1}{2}}$$

which completes the growth claim. □

S.8.3 Proof of power cumulant formula

Here, we restate and prove Proposition S.3.7.

Proposition S.3.7 (Power cumulant partition formula)**.** *Cumulants can be computed from power cumulants slicewise: for each $\lambda \vdash r$ with b blocks,*

$$\tilde{\mathfrak{D}}_\lambda \kappa_r[X] = \sum_{\nu \in \mathrm{Vec}(\lambda)} c_\nu^{\mathrm{vec}} c_\nu^{\mathsf{P}} \prod_{\underline{u} \in \nu} \tilde{\mathfrak{D}}_{\underline{u}} \kappa^{\mathsf{P}}_{|\underline{u}|}[X],$$

where the product is taken elementwise and

$$c_\nu^{\mathsf{P}} := \sum_{\omega \vdash \nu} \mathbb{1}[\omega \text{ totally disconnected}](-1)^{|\omega|-1}(|\omega|-1)!.$$

Here the sum is over set partitions ω of ν (which itself is a set of tuples in $[b]$), $|\omega|$ is the number of blocks in ω, and ω is said to be totally disconnected if for each block of ω, all tuples in that block have disjoint support in $[b]$.

Proof. Fix $\pi \vdash [r]$ and $(i_1, \ldots, i_r)$ of type π. Let μ denote the Möbius function on the partition lattice, so that $\mu(\pi) = (-1)^{|\pi|-1}(|\pi|-1)!$. Then,

$$\kappa[X_{i_1}, \ldots, X_{i_r}] = \sum_{\tau \vdash [r]} \mu(\tau) \prod_{B \in \tau} \mathbb{E}\left[\prod_{j \in B} X_{i_j}\right] \tag{27}$$

$$= \sum_{\tau \vdash [r]} \mu(\tau) \prod_{B \in \tau} \mathbb{E}\left[\prod_{C \in \pi|_B} X_{i_{\min(C)}}^{|C|}\right] \tag{28}$$

$$= \sum_{\tau \vdash [r]} \mu(\tau) \prod_{B \in \tau} \left(\sum_{\rho_B \geq \pi|_B} \prod_{D \in \rho_B} \kappa^{\mathsf{P}}_{|D|}[X]_{i_j : j \in D}\right) \tag{29}$$

$$= \sum_{\tau \in [r]} \mu(\tau) \sum_{\pi \wedge \tau \leq \rho \leq \tau} \prod_{D \in \rho} \kappa^{\mathsf{P}}_{|D|}[X]_{i_j : j \in D} \tag{30}$$

$$= \sum_{\rho \vdash [r]} \left(\sum_{\substack{\tau \vdash [r] \\ \tau \geq \rho \\ \pi \wedge \tau \leq \rho}} \mu(\tau)\right) \prod_{D \in \rho} \kappa^{\mathsf{P}}_{|D|}[X]_{i_j : j \in D}. \tag{31}$$

where

- (27) is the usual Möbius inversion to go from moments to cumulants,
- (28) is grouping X_{i_j} by $\mathrm{type}(i_j : j \in B) = \pi|_B$,
- (29) is the usual sum over partitions to go from cumulants to moments (separately for each B),
- (30) is the observation that the product of partition lattices of blocks of τ is in bijection with the sublattice beneath τ in the partition lattice of $[r]$.

Now, let $\lambda \vdash r$ be the type of π. Then (31) is written in diagonal slice notation as

$$\tilde{\mathfrak{D}}_\lambda \kappa_r[X] = \sum_{\rho \vdash [r]} \left(\sum_{\substack{\tau \vdash [r] \\ \tau \geq \rho \\ \pi \wedge \tau \leq \rho}} \mu(\tau)\right) \prod_{D \in \rho} \tilde{\mathfrak{D}}_{\mathrm{type}(\pi|_D)} \kappa^{\mathsf{P}}_{|D|}[X],$$

and the Proposition follows from grouping (π, ρ) of the same vector partition type. (Note that $\{\tau \vdash [r] : \tau \geq \rho\}$ is in bijection with $\{\tilde{\tau} \vdash \rho\}$, and $\pi \wedge \tau \leq \rho$ iff each block of $\tilde{\tau}$ contains blocks of ρ such that no pair of them intersects the same block of π, whence the sum over totally disconnected set partitions of the vector partition corresponding to (π, ρ).) □

This Proposition tells us that each entry of the cumulants tensors is calculated in $O(1)$ time from the power cumulants.

S.8.4 Proofs of diagonal-harmonic formulae

This section is devoted to proofs of Propositions S.3.10 and S.3.11. Before restating and proving the results, we prove some necessary preliminaries.

Lemma S.8.1. *Let $\lambda = (\lambda_1, \ldots, \lambda_b) = (1^{b_1} \ldots r^{b_r}) \vdash r$. Then*

$$c_\lambda^{\mathrm{part}} := \#\{\pi \vdash [r] : \mathrm{type}(\pi) = \lambda\} = \frac{r!}{\prod_{i=1}^{b} \lambda_i! \prod_{j=1}^{r} b_j!}.$$

Proof. By orbit-stabilizer applied to the action of S_n. Fix a particular $\pi \vdash [r]$ with $\mathrm{type}(\pi) = \lambda$. A permutation that stabilizes π can only permute blocks of the same size ($\prod_j b_j!$ permutations) and then permute elements within each block ($\prod_i m_i!$ permutations). Thus the stabilizer has cardinality $\prod_i \lambda_i! \prod_j b_j!$. □

Definition S.8.2. A *m-partial pairing* of $[r]$ is a set of m disjoint subsets $[r]$ each of cardinality two. We notate the set of m-partial pairings of $[r]$ as $\mathrm{Pair}(r, m)$. Given $\lambda \vdash r$ with b blocks, write $\pi_\lambda \vdash [r]$ for the set partition of type λ formed by partitioning $[r]$ into consecutive contiguous blocks of cardinality $\lambda_1, \ldots, \lambda_b$:

$$\pi_\lambda = (\{1, \ldots, \lambda_1\}, \{\lambda_1 + 1, \ldots, \lambda_1 + \lambda_2\}, \ldots, \{r - \lambda_b + 1, \ldots, r\}).$$

We call this the *sorted set partition* of type λ. The *λ-type* of $\rho \in \mathrm{Pair}([r], m)$, which we denote $\mathrm{type}_\lambda(\rho)$, is the multigraph with m edges on $[b]$ induced by forgetting the ordering on elements within each block of π_λ.

Lemma S.8.3. *Let $\lambda \vdash [r]$ with b blocks, and $\gamma \in \Gamma([b], m)$. Then*

$$\tilde{c}_{\lambda,\gamma}^{\mathrm{g}} := \#\{\rho \in \mathrm{Pair}([r], m) : \mathrm{type}_\lambda(\rho) = \gamma\} = \frac{\prod_{i \in [b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}}{\prod_{i \in [b]} 2^{\gamma(\{i\})} \prod_{\{i,j\} \subseteq [b]} \gamma(\{i,j\})!}. \tag{32}$$

Proof. One approach is another orbit-stabilizer argument. Alternatively, $\prod_{i \in [b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}$ is the number of ways to assign endpoints $s_i, s_j \in [r]$ to each edge $\{i, j\} \subseteq [b]$ such that each s_i is in block i and s_j is in block j. This double-counts the loops (since in that case $i = j$), which contributes the factor $\prod_i 2^{\gamma(\{i\})}$. We also overcount by permutations of edges between the same two vertices, which contributes the factor $\prod_{\{i,j\}} \gamma(\{i,j\})!$ □

Definition S.8.4. Let $\rho \in \mathrm{Pair}([r], m)$. Then $\mathrm{tr}_\rho \colon \mathcal{T}^r(\mathbb{R}^n) \to \mathcal{T}^{r-2m}(\mathbb{R}^n)$ is the linear map that traces out indices corresponding to each pair in ρ. To be more precise, fix an ordering of pairs $\rho = \{\{u_1, v_1\}, \ldots, \{u_m, v_m\}\}$, and sort the vertices

$$\{u_1, v_1, \ldots, u_m, v_m\} = \{c_1, \ldots, c_{2m}\} \qquad 0 = c_0 < c_1 < \ldots < c_m.$$

Then, for $\mathbf{T} \in \mathcal{S}^n(\mathbb{R}^r)$,

$$(\mathrm{tr}_\rho \mathbf{T})_{i_1, \ldots, i_{r-2m}} = \sum_{i_{r-2m+1}, \ldots, i_{r-m}=1}^{n} \mathbf{T}_{i_{j_1}, \ldots, i_{j_r}},$$

where the indices j satisfy

$$j_s = \begin{cases} r - 2m + k & s \in \{u_k, v_k\} \\ s - t & c_t < s < c_{t+1}. \end{cases}$$

As a slight abuse of notation, we may think of the standard tr operator as mapping $\mathcal{T}^r(\mathbb{R}^n) \to \mathcal{T}^{r-2}(\mathbb{R}^n)$ by contracting the first two indices, i.e.

$$\mathrm{tr} := \mathrm{tr}_{\{\{1,2\}\}}. \tag{33}$$

This agrees with the original definition of tr when restricted to $\mathcal{S}^r(\mathbb{R}^n) \subseteq \mathcal{T}^r(\mathbb{R}^n)$.

Definition S.8.5. For $\mathbf{M} \in \mathcal{S}^2(\mathbb{R}^r)$, define $g_{\mathbf{M},\rho} \colon \mathcal{T}^{r-2m}(\mathbb{R}^n) \to \mathcal{T}^r(\mathbb{R}^n)$ by

$$(g_{\mathbf{M},\rho}\mathbf{T})_{i_1,\dots,i_r} = \mathbf{T}_{i_{j_1'},\dots,i_{j_{r-2m}'}} \prod_{\{u,v\}\in\rho}^{m} \mathbf{M}_{i_u,i_v}$$

where

$$j_s' = s + t \quad \text{when} \quad c_t < s < c_{t+1}.$$

Note then that

$$g_{\mathbf{M}}^m = \sum_{\rho\in\mathrm{Pair}([r],m)} g_{\mathbf{M},\rho}. \tag{34}$$

Definition S.8.6. For $\sigma \in S_r$, the operator $\mathrm{P}_\sigma\, \mathcal{T}^r(\mathbb{R}^n) \to \mathcal{T}^r(\mathbb{R}^n)$ acts by permuting indices, i.e.

$$(\mathrm{P}_\sigma\, \mathbf{T})_{i_1,\dots,i_r} = \mathbf{T}_{i_{\sigma(1)},\dots,i_{\sigma(r)}}.$$

Now we can prove the required results. Specifically,

Proposition S.3.10. *There exist coefficients $c^{\mathrm{g}}_{\lambda,\gamma}$ such that, for any $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$, any integer $0 \le s \le \lfloor \frac{r}{2} \rfloor$, and any $\lambda = (\lambda_1, \dots, \lambda_b) = (1^{b_1} \dots r^{b_r}) \vdash r$ with b blocks,*

$$\mathrm{tr}^s\, \mathfrak{D}_\lambda \mathbf{T} = \sum_{\gamma\in\Gamma_{t.d.}([b],s)} \left(c^{\mathrm{g}}_{\lambda,\gamma}\, \mathbb{1}_{\lambda\ge d_\gamma}\, \tilde{\mathfrak{D}}^{-1}_{(\lambda-d_\gamma)} \operatorname*{Sum}_{\{j:\lambda_j=d_\gamma(j)\}} \tilde{\mathfrak{D}}_\lambda \mathbf{T} \right), \tag{11}$$

where $\mathrm{Sum}_{\{j:\lambda_j=d_\gamma(j)\}}$ is the operator that sums along indices $\{j : \lambda_j = d_\gamma(j)\}$ and the difference between λ and d_γ is taken entrywise. Explicitly,

$$c^{\mathrm{g}}_{\lambda,\gamma} := \frac{2^s s! r! \prod_{i\in[b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}}{(r-2s+1)_{2s} \prod_{j\in[r]} b_j! \prod_{i\in[b]} \lambda_i! 2^{\gamma(\{i\})} \prod_{\{i,j\}\subseteq[b]} \gamma(\{i,j\})!}.$$

Proof of Proposition S.3.10. Let $\lambda \vdash r$ with b blocks, and let $\pi_\lambda \vdash [r]$ be the sorted set partition of type λ. For $0 \le m \le \lfloor r/2 \rfloor$ and $\rho \in \mathrm{Pair}([r], m)$ and any $\mathbf{S} \in \mathcal{S}^r(\mathbb{R}^n)$

Let us consider the operator $\mathrm{tr}_\rho\, \mathfrak{D}_{\pi_\lambda} \mathbf{S}$ for some $\rho \in \mathrm{Pair}([r], m)$ and $\mathbf{S} \in \mathcal{S}^r(\mathbb{R}^n)$. If $\mathrm{type}_\lambda(\rho)$ is not totally disconnected, then there exists some $\{u, v\} \in \rho$ such that $u \not\sim_{\pi_\lambda} v$. By definition tr_ρ sums over entries corresponding to indices with $i_u = i_v$, and the output of $\mathfrak{D}_{\pi_\lambda}$ is nonzero only when $i_u \ne i_v$, whence $\mathrm{tr}_\rho\, \mathfrak{D}_{\pi_\lambda} \mathbf{S} = 0$.

What happens when $\mathrm{type}_\lambda(\rho)$ is totally disconnected? The result depends only on $\gamma = \mathrm{type}_\lambda(\rho) \le \lambda$, where the inequality is taken elementwise. For $j \in [b]$ such that $\lambda_j = d_\gamma(j)$, all indices in the jth block are summed over, corresponding to summing over the jth index of $\tilde{\mathfrak{D}}_\lambda \mathbf{S}$. On the other hand, if $\lambda_j > d_\gamma(j)$, there are $\lambda_j - d_\gamma(j)$ free indices in block j not summed over. By the definition of $\mathfrak{D}_{\pi_\lambda}$, if there are multiple such indices, they must be set to the same value for the corresponding entry to be nonzero. Across all indices, this corresponds to applying $\tilde{\mathfrak{D}}^{-1}_{\lambda_j - d_\gamma(j)}$ to the result of summing over indices of $\tilde{\mathfrak{D}}_\lambda \mathbf{S}$ with $\lambda_j > d_\gamma(j)$. We conclude

$$\mathrm{tr}_\rho\, \mathfrak{D}_{\pi_\lambda} \mathbf{S} = \tilde{\mathfrak{D}}^{-1}_{\lambda - d_\gamma} \operatorname*{Sum}_{\{j:\lambda_j=d_\gamma(j)\}} \tilde{\mathfrak{D}}_\lambda \mathbf{S}, \tag{35}$$

Hence,

$$\begin{aligned}
\operatorname{tr}^m \mathfrak{D}_\lambda \mathbf{T} &\overset{(8)}{=} \operatorname{tr}^m \left(\sum_{\substack{\pi \vdash [r] \\ \operatorname{type}(\pi) = \lambda}} \mathfrak{D}_\pi \mathbf{T} \right) \\
&= \operatorname{tr}^m \left(\frac{c_\lambda^{\mathrm{part}}}{r!} \sum_{\sigma \in S_r} (\mathrm{P}_\sigma \, \mathfrak{D}_{\pi_\lambda}) \mathbf{T} \right) \\
&= \frac{c_\lambda^{\mathrm{part}}}{r!} \left(\sum_{\sigma \in S_r} (\operatorname{tr}^m \mathrm{P}_\sigma) \mathfrak{D}_{\pi_\lambda} \right) \mathbf{T} \\
&\overset{(33)}{=} \frac{c_\lambda^{\mathrm{part}}}{\binom{r}{2m}(2m-1)!!} \left(\sum_{\rho \in \mathrm{Pair}([r],m)} \operatorname{tr}_\rho \mathfrak{D}_{\pi_\lambda} \right) \mathbf{T} \\
&\overset{(35)}{=} \frac{c_\lambda^{\mathrm{part}}}{\binom{r}{2m}(2m-1)!!} \left(\sum_{\rho \in \mathrm{Pair}([r],m)} \mathbb{1}_{\operatorname{type}_\lambda(\rho) \text{ t.d.}} \tilde{\mathfrak{D}}^{-1}_{\lambda - d_{\operatorname{type}_\lambda(\rho)}} \operatorname*{Sum}_{\{j : \lambda_j = d_{\operatorname{type}_\lambda(\rho)}(j)\}} \right) \tilde{\mathfrak{D}}_\lambda \mathbf{T} \\
&\overset{(32)}{=} \frac{c_\lambda^{\mathrm{part}}}{\binom{r}{2m}(2m-1)!!} \left(\sum_{\gamma \in \Gamma_{\text{t.d.}}([b],m)} \tilde{c}^{\mathrm{g}}_{\lambda,\gamma} \mathbb{1}_{\lambda \geq d_\gamma} \tilde{\mathfrak{D}}^{-1}_{\lambda - d_\gamma} \operatorname*{Sum}_{\{j : \lambda_j = d_\gamma(j)\}} \right) \tilde{\mathfrak{D}}_\lambda \mathbf{T} \\
&= \left(\sum_{\gamma \in \Gamma_{\text{t.d.}}([b],m)} c^{\mathrm{g}}_{\lambda,\gamma} \mathbb{1}_{\lambda \geq d_\gamma} \tilde{\mathfrak{D}}^{-1}_{\lambda - d_\gamma} \operatorname*{Sum}_{\{j : \lambda_j = d_\gamma(j)\}} \right) \tilde{\mathfrak{D}}_\lambda \mathbf{T}. \qquad \square
\end{aligned}$$

Proposition S.3.11. *There exist coefficients $\tilde{c}^{\mathrm{g}}_{\lambda,\gamma}$ such that, given an integer $0 \leq s \leq \lfloor \frac{r}{2} \rfloor$ and $\lambda = (\lambda_1, \ldots, \lambda_b) = (1^{b_1} \ldots r^{b_r}) \vdash r$ with b blocks, for any $\mathbf{T} \in \mathcal{S}^r(\mathbb{R}^n)$ and $\mathbf{M} \in \mathcal{S}^2(\mathbb{R}^n)$,*

$$\tilde{\mathfrak{D}}_\lambda g^s_{\mathbf{M}}(\mathbf{T}) = \sum_{\gamma \in \Gamma([b],s)} \tilde{c}^{\mathrm{g}}_{\lambda,\gamma} \mathbf{T} \odot_\gamma \mathbf{M}, \tag{12}$$

where $(\mathbf{T} \odot_\gamma \mathbf{M}) \in \mathcal{S}^\lambda(\mathbb{R}^n)$ is defined by

$$(\mathbf{T} \odot_\gamma \mathbf{M})_{i_1,\ldots,i_b} = \begin{cases} (\tilde{\mathfrak{D}}_{\lambda - d_\gamma} \mathbf{T})_{i_1,\ldots,i_b} \prod_{\{u,v\} \in \gamma} \mathbf{M}_{i_u,i_v} & \underline{i} \text{ all distinct} \\ 0 & \text{otherwise.} \end{cases}$$

Explicitly,

$$\tilde{c}^{\mathrm{g}}_{\lambda,\gamma} := \frac{\prod_{i \in [b]} (\lambda_i - d_\gamma(i) + 1)_{d_\gamma(i)}}{\prod_{i \in [b]} 2^{\gamma(\{i\})} \prod_{\{i,j\} \subseteq [b]} \gamma(\{i,j\})!}.$$

Proof of Proposition S.3.11. First, for $\lambda \vdash r$ with b blocks and $\rho \in \mathrm{Pair}([r], m)$, calculate

$$\begin{aligned}
(\tilde{\mathfrak{D}}_\lambda g_{\mathbf{M},\rho} \mathbf{T})_{i_1,\ldots,i_b} &= (g_{\mathbf{M},\rho} \mathbf{T})_{\underbrace{i_1,\ldots,i_1}_{\lambda_1 \text{ times}},\ldots,\underbrace{i_b,\ldots,i_b}_{\lambda_b \text{ times}}} \\
&= \mathbf{T}_{\underbrace{i_1,\ldots,i_1}_{\lambda_1 - d_{\operatorname{type}_\lambda(\rho)}(1) \text{ times}},\ldots,\underbrace{i_b,\ldots,i_b}_{\lambda_b - d_{\operatorname{type}_\lambda(\rho)}(b) \text{ times}}} \prod_{\{u,v\} \in \operatorname{type}_\lambda(\rho)} \mathbf{M}_{i_u,i_v} \mathbb{1}[\underline{i} \text{ distinct}] \\
&= (\tilde{\mathfrak{D}}_{\lambda - d_{\operatorname{type}_\lambda(\rho)}} \mathbf{T})_{i_1,\ldots,i_b} \prod_{\{u,v\} \in \gamma} \mathbf{M}_{i_u,i_v} \mathbf{1}[\underline{i} \text{ all distinct}] \\
&=: (\mathbf{T} \odot_{\operatorname{type}_\lambda \rho} \mathbf{M})_{i_1,\ldots,i_b}.
\end{aligned} \tag{36}$$

Hence,

$$\begin{aligned}\tilde{\mathfrak{D}}_\lambda g^m_{\mathbf{M}}(\mathbf{T}) &\overset{(34)}{=} \sum_{\rho\in\mathrm{Pair}([r],m)} \tilde{\mathfrak{D}}_\lambda g_{\mathbf{M},\rho}\mathbf{T} \\ &\overset{(36)}{=} \sum_{\rho\in\mathrm{Pair}([r],m)} \mathbf{T}\odot_{\mathrm{type}_\lambda\,\rho}\mathbf{M} \\ &\overset{(32)}{=} \sum_{\gamma\in\Gamma([b],m)} \tilde{c}^{\mathrm{g}}_{\lambda,\gamma}\mathbf{T}\odot_\gamma \mathbf{M}.\end{aligned}$$

□

S.8.5 Factorized cumulant propagation for arbitrary $K \geq 3$

Proof of Theorem 5.2. Given Theorem 5.1 (proved in Section S.5), it suffices to show that the factorization strategy exhibited in Section S.4.3 for $K = 3$ can be extended to $K > 3$. The largest cumulant tensor has arity $K > 3$ and we factor it as

$$\mathbf{T}_{i_1,\ldots,s_K} = \frac{1}{K!}\sum_{\sigma\in S_K}\sum_{s=2}^{\lfloor r/2\rfloor}\sum_{t=1}^{T}\mathbf{A}^{(s)}_{i_{\sigma(1)}\ldots i_{\sigma(s)},t}\mathbf{B}^{(s)}_{i_{\sigma(s+1)}\ldots i_{\sigma(K)},t}. \tag{37}$$

Lower-degree cumulants are left unfactored.

For factorized cumulant propagation through each layer to run in time $O(n^K)$, we need to verify each of the following:

- **Symmetric contraction $-\odot\mathbf{W}$ can be done in $O(n^K)$ time when $T = O(n)$**
  Symmetric contraction of the factorized representation is done by contracting $\mathbf{W}$ into each of the indices other than t of all of the factors $\mathbf{A}^{(s)}$ and $\mathbf{B}^{(s)}$ of (37). Since the largest such factor has arity $K-1$, this takes time $O(n^{K-1}T) = O(n^K)$.
- **Hermite coefficients can be multiplied in in time $O(n^K)$ time when $T = O(n)$**
  This is clear because multiplying in Hermite coefficients is an elementwise operation on the factors.
- **Non-discrete diagonals can be extracted in $O(n^K)$ time when $T = O(n)$**
  Let $\lambda \vdash r$ be other than the discrete partition $(1,\ldots,1)$. Then $\tilde{\mathfrak{D}}_\lambda\mathbf{T}$ has no more than n^{K-1} entries, each of which can be computed from (37) in time T. Thus the total time is $O(n^{K-1}T) = O(n^K)$.
- **Non-discrete diagonals can be factored into $T = \Theta(n)$ factors**
  Given $\lambda \vdash r$ other than the discrete partition $(1,\ldots,1)$, we claim $\mathfrak{D}_\lambda\mathbf{T} = \tilde{\mathfrak{D}}^{-1}_\lambda\tilde{\mathfrak{D}}_\lambda\mathbf{T}$ has a factorization in terms of $\tilde{\mathfrak{D}}_\lambda\mathbf{T}$ with $\Theta(n)$ factors. And indeed, since $\lambda_1 > 1$,
  $$\begin{aligned}(\mathfrak{D}_\lambda\mathbf{T})_{i_1,\ldots,i_r} &= \mathbb{1}[i_1 = \cdots = i_{\lambda_1}](\tilde{\mathfrak{D}}^{-1}_{(1,\lambda_2,\ldots,\lambda_n)})\tilde{\mathfrak{D}}_\lambda\mathbf{T})_{i_1,i_{\lambda_1+1},\ldots,i_r} \\ &= \sum_{t=1}^{n}\delta^{(\lambda_1)}_{i_1,\ldots,i_{\lambda_1},t}(\tilde{\mathfrak{D}}^{-1}_{(1,\lambda_2,\ldots,\lambda_n)})\tilde{\mathfrak{D}}_\lambda\mathbf{T})_{t,i_{\lambda_1+1},\ldots,i_r}\end{aligned}$$
  is a factorization with n factors, where $\delta^{(\lambda_1)} \in \mathcal{S}^{\lambda_1}(\mathbb{R}^n)$ is the Kronecker delta, i.e. the tensor that is 1 along the (λ_1)-diagonal and zero elsewhere.
- **Power cumulant diagrams can be factored into $T = \Theta(n)$ factors**
  Consider the set of diagrams contributing to the $(1,\ldots,1) \vdash K$ diagonal slice of the activation power cumulant, which is the only slice that needs to be factored. The key observation is that, because we are dealing with diagrams with the maximum possible arity, if a vector partition of $(1,\ldots,1)$ is to satisfy (14), then it cannot have any cycles when viewed as a multi-hypergraph on $[K]$. Indeed, an acyclic connected diagram ν already makes (14) tight:[22]
  $$k(\nu) = 1 + \sum_{\underline{u}\in\nu}\left(\sum_{j=1}^{K}\left\lfloor\frac{u_j}{2}\right\rfloor - 1\right) \geq 1 + \sum_{\underline{u}\in\nu}\left(\sum_{j=1}^{K}u_j - 1\right) = K,$$

[22]This is a generalization of the standard fact that a tree on V vertices has exactly $V-1$ edges. It can be proved using a straightforward inductive argument.

and completing a cycle with a hyperedge would cause the condition to be broken.

This leaves only a few possibilities for ν:

(a) Each block of the ν has degree no more than $K-1$, in which case there must be at least two blocks, and thus there exists a block with degree $\leq K-2$. (Since two blocks each of degree $\geq K-1$ must form a cycle.) Call this block $\mathbf{C} \in \mathcal{S}^c(\mathbb{R}^n)$ and the rest of the diagram $\mathbf{D} \in \mathcal{S}^d(\mathbb{R}^n)$, with $c, d \geq 2$ and $c \leq K-2$. By the acyclicity and connectedness conditions, $\mathbf{C}$ and $\mathbf{D}$ must be incident along exactly one vertex. Grouping by orbits under the action of S_K on indices, the contribution corresponding to this diagram is then

$$\begin{aligned}\mathbf{T}_{i_1,\ldots,s_K} &= c_\nu \sum_{\sigma \in S_K} \mathrm{P}_\sigma \left(\mathbf{C}_{i_1,\ldots,i_c} \mathbf{D}_{i_c,i_{c+1},\ldots,s_K}\right) \\ &= c_\nu \sum_{\sigma \in S_K} \mathrm{P}_\sigma \left(\sum_{i=1}^{n} (\mathbf{C}_{i_1,\ldots,i_{c-1},i} \delta^{(2)}_{i_c,i} \mathbf{D}_{i,i_{c+1},\ldots,s_K}\right),\end{aligned}$$

where $c_{\sigma,\nu}$ is the appropriate coefficient coming from orbit-stabilizer. This is a factorization into n factors. Note that the first factor $\mathbf{C}_{i_1,\ldots,i_c} \delta^{(2)}_{i_c,i}$ has arity $c+1 \leq K-1$, and so is indeed a valid factorization.

(b) $\nu = \{(1,\ldots,1)\}$, and the corresponding tensor is the already factorized $(1,\ldots,1)$-slice from the previous layer.

(c) K is odd and $\nu = \{(2,\ldots,1)\}$ or one of its permutations. Let $\mathbf{M} = \mathbf{W}\mathbf{W}^\top$. Since in this case $i_K(K+1) = (K+1)/2$, the corresponding tensor is $g_{\mathbf{M}}^{(K+1)/2} \tilde{\eta}_{K+1}[X]$, with $\eta_{K+1}[X]$ a scalar. Hence it suffices to write $g_{\mathbf{M}}^{(K+1)/2} 1$ as a single factor

$$(g_{\mathbf{M}}^{(K+1)/2} 1)_{i_1,\ldots,i_{K+1}} = \frac{(K-1)!!}{K!} \sum_{\sigma \in S_K} \sum_{t=1}^{1} \left(\mathbf{M}_{i_{\sigma(1)},i_{\sigma(2)}} \mathbf{1}_t\right) \left(\mathbf{1}_t \prod_{j=2}^{(K+1)/2} \mathbf{M}_{i_{\sigma(2j-1)},i_{\sigma(2j)}}\right).$$

□

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly summarize the main theoretical and empirical results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The assumption that the weights are randomly initialized, as well as the reliance on large width, are clearly stated in the title, abstract, introduction and throughout the paper. The poor depth dependence is mentioned in the introduction and discussed further in Appendix D. The potential for extending the results to partially-trained networks is discussed in Section 7.2.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Full assumptions are given in theorem statements. Full proofs of all theoretical results are given in the technical supplement, Appendix S.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Precise descriptions of all algorithms are given in the technical supplement, Appendix S, and code is provided.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: No data is required, and code is provided.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup is clearly described in Section 6, and no data is required.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Error bars are included in all plots, except in cases where they would be too small to be readable, where this is explicitly noted.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Experiments can easily be replicated on CPU with minimal compute resources.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: There are no immediate ethical implications of the results, and the safety motivation for the paper is discussed in Section 7.2.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The implications of the results for improving training efficiency and reducing catastrophic tail risk are discussed in Section 7.2.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: There is no direct risk of misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: No existing assets are used.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The provided code includes documentation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were used to help solve mathematical problems in the course of the research, but the final results including proofs were composed by the authors, with non-essential assistance from LLMs. LLMs were also used to help implement algorithms, but all code was carefully reviewed by the authors.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.