KAMA-NNs: low-dimensional rotation based neural networks

Krzysztof Choromanski[*] Google Brain Robotics
Aldo Pacchiano[*] UC Berkeley
Jeffrey Pennington[*] Google Brain
Yunhao Tang[*] Columbia University

Abstract

We present new architectures for feedforward neural networks built from products of learned or random low-dimensional rotations that offer substantial space compression and computational speedups in comparison to the unstructured baselines. Models using them are also competitive with the baselines and often, due to imposed orthogonal structure, outperform baselines accuracy-wise. We propose to use our architectures in two settings. We show that in the non-adaptive scenario (random neural networks) they lead to asymptotically more accurate, space-efficient and faster estimators of the so-called PNG-kernels (for any activation function defining the PNG). This generalizes several recent theoretical results about orthogonal estimators (e.g. orthogonal JLTs, orthogonal estimators of angular kernels and more). In the adaptive setting we propose efficient algorithms for learning products of low-dimensional rotations and show how our architectures can be used to improve space and time complexity of state of the art reinforcement learning (RL) algorithms (e.g. PPO, TRPO). Here they offer up to 7x compression of the network in comparison to the unstructured baselines and outperform reward-wise state of the art structured neural networks offering similar computational gains and based on low displacement rank matrices.

1 Introduction

Structured transforms play an important role in many machine learning algorithms. Several recently proposed scalable kernel methods using random feature maps [Rahimi and Recht, 2007] apply structured matrices to either reduce time & space complexity of kernels' estimators or improve their accuracy [Choromanska et al., 2016, Choromanski et al., 2018b, Bojarski et al., 2017, Choromanski et al., 2017, Choromanski et al., 2018a, Yu et al., 2016, Choromanski and Sindhwani, 2016, Vybíral, 2011, Zhang and Cheng, 2013]. Structured matrices are also applied in some of the fastest known cross-polytope LSH algorithms [Andoni et al., 2015] and neural networks [Sindhwani et al., 2015, Choromanski et al., 2018a, Choromanski et al., 2018c]. In the latter setting they were used in particular to scale up architectures for mobile speech recognition [Sindhwani et al., 2015], predictive state recurrent neural networks [Choromanski et al., 2018a] and more recently, to encode policy architectures for RL tasks [Choromanski et al., 2018c]. Compressed neural networks encoded by structured matrices enable practitioners to train RL policies with the use of evolutionary strategy algorithms (recently becoming a serious alternative to state of the art policy gradient methods [Salimans et al., 2017, Mania et al., 2018]) on a single machine instead of clusters of thousands of machines. Time & space complexity reduction is obtained by applying structured matrices where matrix-vector multiplication can be conducted in sub-quadratic time with the use of Fast Fourier Transform (e.g. low displacement rank matrices from [Choromanski and Sindhwani, 2016, Sindhwani et al., 2015, Choromanski et al., 2018c]) or Fast Walsh-Hadamard Transform (e.g. random Hadamard matrices from [Andoni et al., 2015, Choromanski et al., 2017]).

However, time & space complexity reduction as well as accuracy improvements over unstructured baselines at the same time were recently theoretically proven only for random Hadamard matrices and only for the linear kernel (see: dimensionality reduction mechanisms in [Choromanski et al., 2017]). Furthermore, it is known that obtained accuracy gains are due to the orthogonality and similar guarantees cannot be achieved for low displacement rank matrices. Other orthogonal transforms for which accuracy improvements were proven

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. [*] Equal contribution.

(angular kernel estimators and asymptotically estimators of certain classes of RBF-kernels) were built from random orthogonal matrices constructed via Gram-Schmidt orthogonalization [Yu et al., 2016]. These do not offer any compression of the number of parameters and computational gains.

We propose here a class of structured neural network architectures, called, KAMA-NNs, where matrices of connections can be decomposed as products of low-dimensional learned or random rotations. We show that the best features of all the aforementioned structured families are encapsulated in this class. First, it provides orthogonality and we show that in the non-adaptive scenario (random neural networks) it consequently leads to the asymptotically more accurate, space-efficient and faster estimators of the so-called PNG (Pointwise Nonlinear Gaussian) kernels (for any activation function defining the PNG) and RBF-kernels. This generalizes several recent theoretical results about orthogonal estimators (e.g. orthogonal JLTs, orthogonal estimators for angular, gaussian kernels and more). Furthermore, it achieves time & space complexity gains over unstructured baselines, matching or outperforming accuracy-wise low displacement rank matrices. Finally, in the adaptive setting, where low-dimensional rotations are learned, it improves state of the art reinforcement learning algorithms (e.g. PPO, TRPO). In the RL setting our architectures offer up to 7x compression of the network in comparison to the unstructured baselines and outperform reward-wise state of the art structured neural networks offering similar computational gains and based on low displacement rank matrices [Choromanski et al., 2018c]. In the adaptive setting, we also give an interesting geometric interpretation of our algorithms. We explain how optimizing over a sequence of low dimensional rotations is akin to performing coordinate descent/ascent on the manifold of all rotation-matrices. This sheds light on the effectiveness of the KAMA-NN mechanism also in the adaptive setting.

We highlight our main contributions below:

- In Section 2 we formally introduce KAMA-NN architectures and discuss their space and time complexity.
- In Section 3 we discuss the capacity of models based on products of low dimensional rotations in both: adaptive and non-adaptive setting and the connection to random matrix theory.
- In Section 4 we establish the connection between random neural networks and PNG kernels and show that random KAMA-NNs lead to asymptotically more accurate estimators of these kernels.
- In Section 5 we analyze adaptive mechanism, where low dimensional rotations defining KAMA-NN architectures are trained and provide convergence results for optimizing certain classes of black-box functions via KAMA-NNs.
- In Section 6 we given an exhaustive empirical evaluation of KAMA-NN architectures. In the non-adaptive setting (random KAMA-NNs) we perform an empirical study of the accuracy of PNG kernels' estimators with KAMA-NNs. In the adaptive setting we apply KAMA-NNs to encode RL policies and compare them to unstructured and other structured architectures for different policy gradient algorithms and on several RL tasks.

2 KAMA-NN architectures

KAMA-NNs are feedforward neural networks, with matrices of connections constructed from products of 2-dimensional learned or random rotations called *Givens rotations*. For fixed $I, J \in \{0, 1, ..., d-1\}$ $(I \neq J)$ and $\Theta \in [0, 2\pi)$ we define a Givens rotation $\mathbf{G}^{\Theta}_{I,J} \in \mathbb{R}^{d \times d}$ as follows:

$$\mathbf{G}^{\Theta}_{I,J}[i,j] = \begin{cases} 1, & \text{if } i = j \text{ and } i \notin \{I, J\} \\ 0, & \text{if } i \neq j \text{ and } \{i, j\} \neq \{I, J\} \\ \cos\Theta, & \text{if } i = j \text{ and } i \in \{I, J\} \\ \sin\Theta, & \text{if } i = J, j = I \\ -\sin\Theta, & \text{if } i = I, j = J \end{cases},$$

Givens random rotation is a Givens rotation, where $\Theta \sim Unif[0, 2\pi)$ and I, J are chosen uniformly at random.

Each matrix $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$ of connections of the KAMA-NN is obtained from the product of k Givens rotations of $\max(d_1, d_2)$ rows and columns each (where k may differ form layer to layer) by taking its first d_1 rows (d_2 columns) if $d_1 \leq d_2$ ($d_1 > d_2$) and then renormalizing these $\min(d_1, d_2)$ rows (columns). The renormalization is conducted by multiplying these rows (columns) by fixed $\min(d_1, d_2)$ scalars: $s_1, ..., s_{\min(d_1,d_2)}$. Random KAMA-NNs apply matrices $\mathbf{M}$ using Givens random rotations chosen independently and with scalars $s_1, ..., s_{\min(d_1,d_2)}$ chosen independently from a given probabilistic $1D$-distribution Φ. In general these as well as angles Θ and indices I, J of Givens rotations are learned. Notice that products of independent Givens random rotations are sometimes called *Kac's random walk matrices* since they correspond to the *Kac's random walk* Markov Chain [Kac, 1954]. KAMA stands for: Kacs Asymptotic Matrix Approximators, since, as we will see in Section 3, these constructions can be used to approximate many classes of matrices.

Space and time complexity gains: Matrix-vector mutliplication with matrices $\mathbf{M} = \mathbf{G}_1 \cdot ... \mathbf{G}_k \in \mathbb{R}^d$, where $\mathbf{G}_i$s are Givens rotations, can be conducted in time $O(k)$, since mutliplication by each Givens rotation can be trivially done in time $O(1)$ (it requires exactly four scalar multiplications and two additions). Furthermore, the total number of parameters needed to encode matrix $\mathbf{M}$ together with renormalization parameters s_i (see: above discussion) equals $3k + d$ (two indices and one angle per Givens rotation and d renormalization scalars). We will see later (see: Section 3) that in practice $k = O(d \log(d))$ Givens matrices suffice to encode architectures capable of learning good quality models. Thus KAMA-NNs provide faster inference than unstructured counterparts and form a class of compact yet expressible neural network architectures.

3 Capacity of KAMA-NNs

Products of low dimensional rotations have been the subject of voluminous research, partially because of their applications in physics [Kac, 1954, Janvresse, 2001, Mischler and Mouhot, 2013]. Kac's random walk, where transitions from previous to next d-dimensional states are defined by independent Givens random rotations, was introduced in [Kac, 1954]. It was shown that products of such rotations converge (in certain sense) to the truly random rotation, yet are much more efficient to compute. For instance, it was recently proven that Kac's random walk on the d-sphere mixes in $d \log(d)$ steps [Pillai and Smith, 2015]. This result suggests that products of relatively small number of low dimensional rotations may serve as a good proxy for truly random rotation matrices sampled from the distribution corresponding to the Haar measure and providing solid theoretical guarantees at the same time (as opposed to other structured random matrices giving computational speedups such as random Hadamard matrices, but for which only vague theoretical guarantees were given so far). This suggest straightforward applications in machine learning (for instance to produce fast random feature map based estimators of RBF or PNG kernels [Rahimi and Recht, 2007]), yet surprisingly to the best of our knowledge, so far mechanisms based on Givens random rotations were proposed only in the context of dimensionality reduction and Johnson-Lindenstrauss Transforms [Ailon and Chazelle, 2006].

Not much is also known regarding the adaptive setting, where Givens rotations are learned. In [Mathieu and LeCun, 2014] learned Givens rotations were applied to approximate Hessian matrices for certain optimization problems. It is believed though that products of relatively small number of learned Givens rotations can accurately approximate matrices from many classes of rotations. In particular, we will focus on the following family.

Definition 1. *Denote by* $\mathrm{GIV}(d)$ *the class of all Givens rotations from* $\mathbb{R}^d \times \mathbb{R}^d$. *For a constant* $C > 0$, *let* $\mathcal{G}_C$ *be a family of matrices in* $\mathbb{R}^{d \times d}$ *defined as:* $\mathcal{G}_C = \{\mathbf{G}_1 \cdot ... \cdot \mathbf{G}_k : \mathbf{G}_i \in \mathrm{GIV}(d), i = 1, ..., k\}$, *where* $k = \lceil C \frac{d \log(d)}{2} \rceil$.

Even though those families are not dense in the set of all rotation matrices, as we will show later, in practice they can accurately approximate many rotation matrices in both adaptive and non-adaptive setting. The renormalization scalars s_i defined by us in Section 2 can then "stretch" certain dimensions of the input vectors rotated by such matrices. This mechanism has sufficient capacity to provide accurate and superior performance to unstructured baselines estimators of all PNG kernels that correspond to random neural networks, as we will see next. We then show that it also suffices in the adaptive setting to learn good quality RL policies.

4 Random KAMA-NNs

Consider a random neural network with input and output layer of size d, nonlinearity f applied to neurons in the output layer and weights taken independently at random from the gaussian distribution $\mathcal{N}(0, \frac{1}{\sqrt{d}})$. This is a standard choice for weights initialization in feedforward neural networks. We call it *unstructured random NN*. Several recent results [Pennington and Worah, 2017], [Pennington et al., 2017], [Pennington et al., 2018] focus on understanding statistical properties of unstructured random NNs and their connection to random matrix theory. Recent work [Pennington et al., 2017, Xiao et al., 2018, Chen et al., 2018] shows also that orthogonal random initialization of neural networks leads to better learning profiles, even though not much is known about this phenomenon from the theoretical point of view. We shed light on it, by showing that random KAMA-NNs as well as previously analyzed random orthogonal constructions lead to asymptotically as $d \to \infty$ more accurate estimators of the so-called PNG (Pointwise nonlinear Gaussian) kernels.

Definition 2 (PNG-kernels)**.** *The* PNG *kernel (shortly:* PNG*) defined by the mapping* f *is a function:* $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ *given as follows for* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$*:*

$$K_f(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{\mathbf{g} \sim \mathcal{N}(0, \mathbf{I}_d)}[f(\mathbf{g}^\top \mathbf{x}) f(\mathbf{g}^\top \mathbf{y})]. \quad (1)$$

PNG kernels are important in the analysis of unstructured random NNs since such networks can be equivalently thought of as transformations that translate one of the most basic similarity measures between feature vectors, namely the linear (dot-product) kernel

by the PNG corresponding to the particular mapping f. Indeed, consider a linear kernel between activation vectors $a(\mathbf{x})$ and $a(\mathbf{y})$ corresponding to given input vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. The following is true:

$$a(\mathbf{x})^\top a(\mathbf{y}) = \frac{1}{d}\sum_{i=1}^{d} f(\mathbf{g}_i^\top \mathbf{x}) f(\mathbf{g}_i^\top \mathbf{y}), \tag{2}$$

where $\mathbf{g}_i$ is the i^{th} row of the connection matrix. Therefore unstructured random NNs become unbiased Monte Carlo (MC) estimators of the values of particular PNG kernels. We denote these unbiased estimators as $\mathrm{URN}_f(\mathbf{x}, \mathbf{y})$. Notice that URNs choose values of weights independently from $\mathcal{N}(0, 1)$. The so-called *orthogonal random NN* is obtained by replacing Gaussian matrix of connections $\mathbf{G}$ by its orthogonal variant $\mathbf{G}_{\mathrm{ort}}$. Matrix $\mathbf{G}_{\mathrm{ort}}$ is obtained from $\mathbf{G}$ by conducting Gram-Schmidt orthogonalization and then using scalars $s_1, ..., s_d$ to renormalize rows of the obtained orthonormal matrix, where s_i are sampled independently from $\|\mathbf{g}\|_2$ for $\mathbf{g} \sim \mathcal{N}(0, \mathbf{I}_d)$. We will denote the corresponding estimator as $\mathrm{ORN}_f(\mathbf{x}, \mathbf{y})$. Finally, if instead we use random KAMA-NNs constructed from k blocks and with scalars s_i chosen in the same way as for orthogonal random NNs, then the corresponding estimator will be denoted as $\mathrm{KRN}_f^k(\mathbf{x}, \mathbf{y})$.

For $\epsilon > 0$, we denote by $B(\epsilon)$ a ball centered at 0 and of radius ϵ. Our main theoretical results in this section are given below.

Theorem 1 (orthogonal random NNs for PNGs)**.** *Let $f : \mathbb{R} \to \mathbb{R}$ be a function and $\mathcal{B} \subseteq \mathbb{R}^d$ be bounded region (for instance a unit sphere). Then for every constant $\epsilon > 0$ there exists a constant $K(\epsilon) > 0$ such that for every $\mathbf{x}, \mathbf{y} \in \mathcal{B} \backslash B(\epsilon)$ the following holds for d large enough:*

$$\mathrm{MSE}(\mathrm{ORN}_f(\mathbf{x}, \mathbf{y})) \leq \mathrm{MSE}(\mathrm{URN}_f(\mathbf{x}, \mathbf{y})) - \frac{K(\epsilon)}{d}, \tag{3}$$

where MSE *stands for the mean squared error.*

If instead of orthogonal random NNs, we use KAMA-NNs then the following is true:

Theorem 2 (KAMA-NNs for PNGs)**.** *Let $f : \mathbb{R} \to \mathbb{R}$ be a function and $\mathcal{B} \subseteq \mathbb{R}^d$ be bounded region (for instance a unit sphere). Then for every constant $\epsilon > 0$ there exist constants $K(\epsilon), L(\epsilon) > 0$ such that for $k = L(\epsilon) d \log(d)$ and for every $\mathbf{x}, \mathbf{y} \in \mathcal{B} \backslash B(\epsilon)$ the following holds for d large enough:*

$$\mathrm{MSE}(\mathrm{KRN}_f^k(\mathbf{x}, \mathbf{y})) \leq \mathrm{MSE}(\mathrm{URN}_f(\mathbf{x}, \mathbf{y})) - \frac{K(\epsilon)}{d}. \tag{4}$$

The above theorems show that orthogonal random NNs as well as random KAMA-NNs provide asymptotically as $d \to \infty$ more accurate estimators of PNG kernels for any nonlinear function f. Previously these results were known only for orthogonal random NNs with sin / cos nonlinear mappings corresponding to RBF kernels [Choromanski et al., 2018b] and with $f(x) = \mathrm{sgn}(x)$ corresponding to angular PNG kernels [Choromanski et al., 2017]. Not only do random KAMA-NNs give accuracy gains, but they also lead to faster inference ($O(d \log(d))$ versus $O(d^2)$ time) and compression of the model ($O(d \log(d))$ versus $O(d^2)$ space) that orthogonal random NNs are not capable of. Our empirical results in Section 6 confirm all our theoretical findings and show also that mean squared error guarantees translate to more downstream guarantees.

5 Learning KAMA-NNs

5.1 Givens rotations and manifold coordinate ascent

The space of $d \times d$ dimensional orthogonal matrices with positive determinant form a connected manifold known as the special orthogonal matrix group, also denoted as $\mathbb{SO}(d)$ [Gallier and Xu, 2003]. This set is a $\frac{(d-1)d}{2}$ dimensional manifold with the property that each point $\mathbf{M} \in \mathbb{SO}(d)$ has an associated tangent space, which is a $\frac{(d+1)d}{2}$ dimensional vector space where the tangent directions to $\mathbb{SO}(d)$ live. The tangent space is denoted as $T_{\mathbf{M}}\mathbb{SO}(d)$ is defined as $T_{\mathbf{M}}\mathbb{SO}(d) = \{A \in \mathbb{R}^{d \times d}, A = \mathbf{M}\Omega : \Omega = -\Omega^\top\}$, the set of Skew Symmetric matrices premultiplied by $\mathbf{M}$.

We show that maximizing a function $F : \mathbb{SO}(d) \to \mathbb{R}$ by performing coordinate gradient ascent over the manifold $\mathbb{SO}(d)$ naturally yields a solution that equals a product of Givens rotations.

5.1.1 Ascent and descent directions in the manifold.

When moving along a manifold, the right generalization of a straight line between two points is the notion of geodesic curves. For any given point $\mathbf{M} \in \mathbb{SO}(d)$ and direction $\mathbf{M}\Omega \in T_{\mathbf{M}}\mathbb{SO}(d)$, there is a single geodesic that passes through $\mathbf{M}$ in direction $\mathbf{M}\Omega$. In the case of the Special Orthogonal Group of matrices these curves can be written in the parameteric form $\gamma_\Omega : \mathbb{R} \to \mathbb{SO}(d)$, such that $\gamma_\Omega(\Theta) = \mathbf{M}\exp(\Theta\Omega)$. The expression $\exp(A)$ for $A \in \mathbb{R}^{d \times d}$ denotes the matrix exponential of A.

$\exp(\cdot)$ maps any skew symmetric matrix Ω to an orthogonal matrix. As a consequence, $\gamma_\Omega(\Theta) = \mathbf{M}\exp(\Theta\Omega)$ is an orthogonal matrix for all $\Theta \in \mathbb{R}$ provided $\mathbf{M} \in \mathbb{SO}(d)$. Let $\{\mathbf{A}_{I,J}\}_{1 \leq I < J \leq d}$ be the following basis for the space of skew symmetric $d \times d$ matrices:

$$\mathbf{A}_{I,J}[i,j] = \begin{cases} 1 & \text{if } i = I, j = J \\ -1 & \text{if } i = J, j = I \\ 0 & \text{o.w.} \end{cases}$$

The exponential of scalar multiples of these basis elements equal Givens rotations: $\exp(-\Theta \mathbf{A}_{I,J}) = \mathbf{G}^{\Theta}_{I,J}$. As a result, the geodesic passing through $\mathbf{M}$ in direction $\mathbf{A}_{I,J}$ equals $\gamma_{I,J}(\Theta) = M\mathbf{G}^{\Theta}_{J,I}$ [Gallier and Xu, 2003].

Let $F : \mathbb{SO}(d) \to \mathbb{R}$ be a differentiable function over the manifold of Special Orthogonal matrices. Similar to the Euclidean space definition, the directional derivative along $\mathbf{A}_{I,J}$ and evaluated at $\mathbf{M} \in \mathbb{SO}(d)$ is a scalar taking the value:

$$\begin{aligned} \nabla_{I,J} F(\mathbf{M}) &:= \frac{d}{d\Theta} F(\gamma_{I,J}(\Theta))|_{\Theta=0} \\ &:= \frac{d}{d\Theta} F(\mathbf{M}\mathbf{G}^{\Theta}_{J,I})|_{\Theta=0} \end{aligned}$$

The Reimannian gradient of F at $\mathbf{M}$, denoted as $\nabla F(\mathbf{M})$ is a matrix in $T_{\mathbf{M}}\mathbb{SO}(d)$ of the form $\mathbf{M}\Omega$ with $\Omega = \sum_{1 \le I < J \le d} \mathbf{A}_{I,J} \nabla_{I,J} F(\mathbf{M})$. The Frobenius norm of $\nabla F(\mathbf{M})$ equals the Frobenius norm of Ω.

Algorithm 1 Givens coordinate gradient descent on $\mathbb{SO}(d)$.

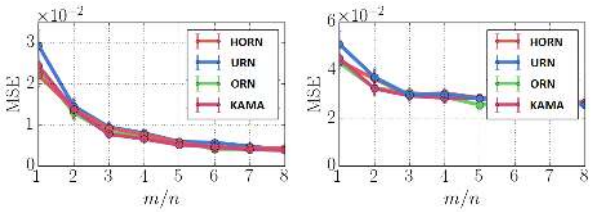
Input: $\mathbf{M}_0 \in \mathbb{SO}(d)$, $F : \mathbb{SO}(d) \to \mathbb{R}$
for $t = 1, 2, \ldots$ **do**
  for I, J *s.t.* $1 \le I < J \le d$ **do**
    2. $\Theta_t^{I,J} = \arg\min_{\Theta} F(\mathbf{M}_{t-1}\mathbf{G}^{\Theta}_{J,I})$.
    3. Let $B_t^{I,J} = F(\mathbf{M}_{t-1}\mathbf{G}^{\Theta_t^{I,J}}_{J,I})$
  4. Let $I_t, J_t = \arg\min_{I,J} B_t^{I,J}$
  5. $\mathbf{M}_t = \mathbf{M}_{t-1}\mathbf{G}^{\Theta_t^{I_t,J_t}}_{J_t,I_t}$

Assume that for all I, J and $\mathbf{M} \in \mathbb{SO}(d)$, the function $\Phi_{I,J}(\Theta) = F(\mathbf{M}G^{\Theta}_{I,J})$ has second derivative bounded by a constant B. The following theorem holds, a derandomized version of Theorem 2 in [Shalit and Chechik, 2014]:

Theorem 3.

$$F(\mathbf{M}_0) - F(\mathbf{M}_t) \ge \sum_{u=0}^{t-1} \frac{\|\nabla F(\mathbf{M}_u)\|^2}{2d(d-1)B}$$

As a consequence, if throughout the algorithm's run $\|\nabla F(\mathbf{M}_u)\|^2 \ge D$ for some constant D, then an objective gain of $\Omega(1/d^2)$ is guaranteed at each step. Since $\mathbb{SO}(d)$ is compact and F continuous, it must be lower bounded by a finite value and therefore Algorithm 1 must converge to a stationary point $\mathbf{M}^*$ with gradient $\|\nabla F(\mathbf{M}^*)\|_F^2 = 0$.



(a) boston (b) wine

Figure 1: Empirical MSE (mean squared error) for the pointwise evaluation of the angular kernel. The following estimators are compared: baseline using independent Gaussian vectors (URN), structured using random Hadamard matrices with renormalized rows (HORN), structured using $k = 5d\log(d)$ Givens random rotations (KAMA) and structured using matrices $\mathbf{G}_{\text{ort}}$ (ORN). We use datasets: boston and wine.

6 Experiments

6.1 The non-adaptive setting: random KAMA-NNs

Here we consider KAMA-NNs using products of random Givens rotations (Kac's random walk matrices) and show that they outperform unstructured PNG kernel estimators on the example of the angular kernel, matching the most accurate orthogonal estimators proposed recently, yet superior to them in terms of space and time complexity.

Pointwise kernel approximation: In the first set of experiments we computed empirical mean squared error (MSE) of several random matrix based estimators of the angular kernel defined as $K_{\text{ang}}(\mathbf{x}, \mathbf{y}) = 1 - \frac{2\theta_{\mathbf{x},\mathbf{y}}}{\pi}$, where $\theta_{\mathbf{x},\mathbf{y}}$ stands for an angle between $\mathbf{x}$ and $\mathbf{y}$. This kernel can be equivalently rewritten as: $K_{\text{ang}}(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{\mathbf{g} \sim \mathcal{N}(0, \mathbf{I}_d)}[f(\mathbf{g}^\top \mathbf{x}) f(\mathbf{g}^\top \mathbf{y})]$, where $f(x) = \text{sgn}(x)$. We compared the following estimators: KRN_f^k with $k = 5d\log(d)$ Givens random rotations and corresponding to random KAMA-NNs, baseline URN using unstructured Gaussian matrices, estimator ORN built on matrices $\mathbf{G}_{\text{ort}}$ as well as estimators applying random Hadamard matrices (HORN) [Yu et al., 2016]. Experiments were conducted on the following datasets: boston and wine. Results are presented on Figure 1. We see that KAMA-NNs provide moderate accuracy gains over unstructured baselines. Next we show that these moderate gains translate to more substantial accuracy gains on more downstream tasks.

Approximating kernel matrices: Here we test the relative error of kernel matrix estimation via different angular kernel estimators based on random matices, in

particular those applying random KAMA-NNs.

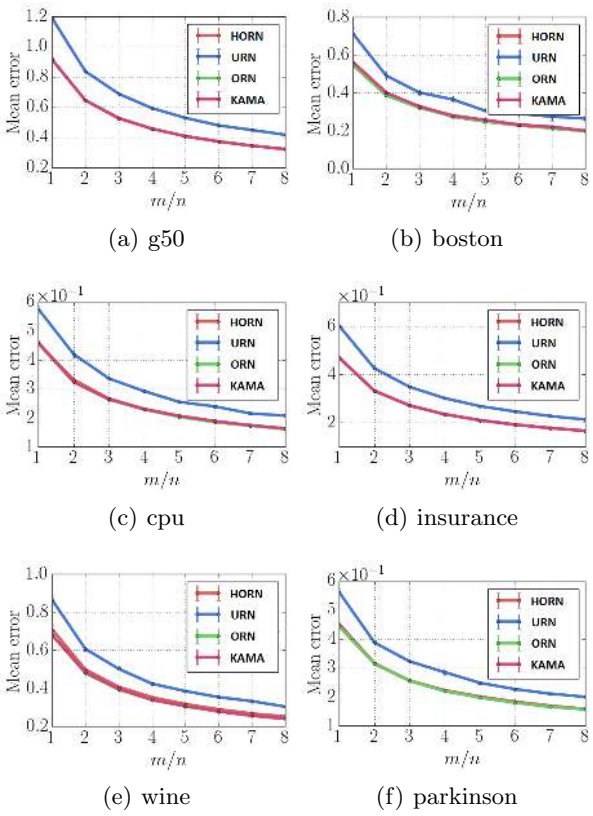
For a given dataset $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ and an angular kernel K_{ang} denote by $K(\mathcal{X}) = \{K_{\text{ang}}(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j \in \{1,...,N\}}$ the corresponding kernel matrix and by $\widehat{K}(\mathcal{X})$ its approximate version obtained by using values proposed by a given estimator. The relative error of the kernel matrix estimation is given as: $\epsilon = \frac{\|K(\mathcal{X}) - \widehat{K}(\mathcal{X})\|_F}{\|K(\mathcal{X})\|_F}$, where $\|\|_F$ stands for the Frobenius norm. We use the following datasets: g50, boston, cpu, insurance, wine and parkinson. Following [Choromanski et al., 2017], we plot the mean error obtained from $r = 1000$ repetitions for each mechanism. Kernel matrices are computed on a randomly selected subset of $N = 550$ datapoint from each datasets.

Results are presented on Figure 2. In all plots different orthogonal mechanisms show similar performance (almost identical curves substantially better than for the URN mechanism), while KAMA-NNs outperform other orthogonal transforms speed-wise.

6.2 The adaptive setting: learning RL policies

We show that KAMA-NNs can substantially reduce the number of policy parameters in reinforcement learning (RL) benchmark tasks, while still providing good performance. We choose standard unstructured fully connected feedforward neural network policy architectures and structured neural network policies with Toeplitz matrices as baselines [Choromanski et al., 2018c]. With many more parameters, fully-connected policies can represent a much larger policy space, which facilitates easier optimization and leads to better performance in practice. On the other hand, Toeplitz policies greatly compress parameters' space, but at the cost of significant degradation of policy performance. We show that KAMA-NNs policy achieves a desirable middle ground between these two extremes: drastically reducing the number of parameters compared to a fully-connected policy, while achieving better performance than Toeplitz policy and providing the same computational speed-ups.

Our fully-connected neural network policies consist of two hidden layers, each with $h = 64$ hidden units for PPO algorithm and $h = 32$ hidden units for TRPO algorithm (see: below). Let $\mathbf{x}$ and $\mathbf{y} = \sigma(\mathbf{W}\mathbf{x}+\mathbf{b})$ be the activations at the first and second hidden layer respectively, where $\mathbf{W} \in \mathbb{R}^{h \times h}$ is a weight matrix, $\mathbf{b} \in \mathbb{R}^h$ is a bias vector and $\sigma(\cdot)$ is the non-linear activation function. To construct a compact policy using KAMA mechanism, we replace the unstructured weight matrix $\mathbf{W}$ by a sequence of K Givens rotations. We do the same for the first and last matrix of connections, but



(a) g50 (b) boston

(c) cpu (d) insurance

(e) wine (f) parkinson

Figure 2: Normalized Frobenius norm error for the angular PNG kernel matrix approximation. The following estimators are compared: baseline using independent Gaussian vectors (URN), structured using random Hadamard matrices with renormalized rows (HORN), structured using $k = 5d\log(d)$ Givens random rotations (KAMA) and structured using matrices $\mathbf{G}_{\text{ort}}$ (ORN). Experiments are run on six datasets: g50, boston, cpu, insurance, wine and parkinson.

this time apply also the truncation mechanism, as described in Section 2. While using Toeplitz mechanism, we replace all three matrices of connections by Toeplitz matrices.

Learning low dimensional rotations: We now introduce the way to parameterize and learn rotations for the KAMA mechanism. Upon initialization, we randomly sample $2D$-linear subspaces, where rotations defined by Givens matrices $\mathbf{G}_i$ are conducted, from the set of subspaces spanned by two vectors from the canonical basis $\{\mathbf{e}_1, ..., \mathbf{e}_n\}$. For each $\mathbf{G}_i$ only angle Θ_i of the rotation is learned. Unstructured matrices are replaced by (truncated) matrices of the form $\mathbf{G}_1\mathbf{G}_2...\mathbf{G}_K$. All rotation angles Θ_i are learned by back-propagation. For each matrix we also learn renormalization scalars s_i (see: Section 2).

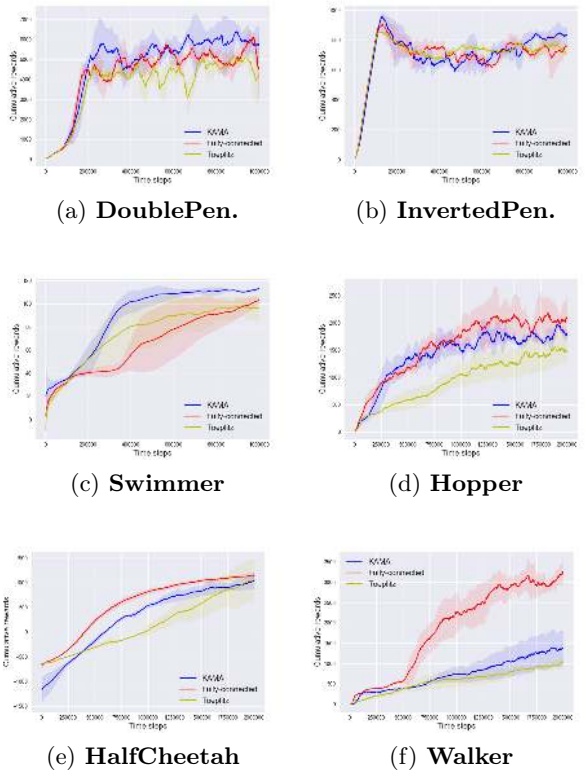
Remark 1. *Note that in the above setting we do*

not need to explicitly store structured matrices $\mathbf{S} = \mathbf{G}_1\mathbf{G}_2...\mathbf{G}_K$. *It suffices to keep:* $\theta_1, ..., \theta_K$ *to efficiently compute* $\mathbf{Sx}$ *for any input* $\mathbf{x}$.

Algorithms and Tasks: We test our policy for state of the art RL algorithms: Trust Region Policy Optimization [Schulman et al., 2015] (TRPO) and Proximal Policy Optimization [Schulman et al., 2017] (PPO). Using these two algorithms, we compare different policy architectures. All implementations are using OpenAI baseline [Dhariwal et al., 2017]. The benchmark tests are based on MuJoCo locomotion tasks provided by OpenAI Gym [Brockman et al., 2016, Todorov et al., 2012] and Roboschool [Schulman et al., 2017]. We take the following environments: Double Pendulum, Inverted Pendulum, Swimmer, Hopper, HalfCheetah.

6.2.1 Proximal Policy Optimization (PPO)

In Figure 3, we show training results on MuJoCo benchmark tasks with Proximal Policy Optimization (PPO) [Schulman et al., 2017]. Here we use $K = 200$ Givens rotations to construct all three structured matrices in the policy. We train the policy on each task for a fixed number of time steps and record the cumulative rewards during training. We show the mean $\pm$ std performance across 5 random seeds. As seen from Figure 3, across most tasks KAMA-NNs policies achieve better performance than Toeplitz policies.



(a) **DoublePen.** (b) **InvertedPen.** (c) **Swimmer** (d) **Hopper** (e) **HalfCheetah** (f) **Walker**

Figure 3: Illustration of KAMA-NNs policies on MuJoCo benchmarks with PPO. KAMA-NNs are compared with unstructured baselines and architectures based on low displacement rank matrices (Toeplitz). For each task we train the policy with PPO for a fixed number of steps and show the mean $\pm$ std performance. Vertical axis is the cumulative reward and horizontal axis stands for the # of time steps.

6.2.2 Trust Region Policy Optimization (TRPO)

In Figure 4, we show training results on MuJoCo benchmarks with Trust Region Policy Optimization (TRPO) [Schulman et al., 2015]. Here we use $K = 100$ Givens rotations to construct all three structured matrices. As before, we train the policy on each task for a fixed number of time steps and record the cumulative rewards during training. We show the mean $\pm$ std performance across 5 random seeds. As seen from Figure 4, across most tasks KAMA-NNs policies achieve significantly betters performance than Toeplitz policies.

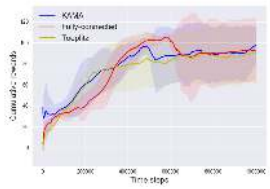
6.2.3 Parameter Compression

By replacing unstructured matrices in the fully connected architecture, structured policies can achieve significant compression in the number of parameters. In the settings where unstructured models are large, structured models can offer much faster inference during training and require much less storage. In Table 1, we list the ratio of the total number of parameters used by structured policies relative to unstructured policies. The two structured policies that we compare (built from KAMA-NNs and Toeplitz networks) provide the same computational speed-ups for the inference (similar number of floating point multiplications).

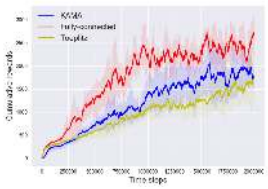
On benchmark tasks, KAMA-NN based policies achieve 7x compression relative to the unstructured model. Though Toeplitz policy reduces the number of parameters even further, the significant drop in performance observed in Figure 3 and Figure 4 is not desirable. We also show in the Appendix that we can further reduce the number of the parameters in the KAMA-NNs by decreasing the number of Givens rotations in the first and third structured matrix, without affecting learned policy and at the same time, further compressing the model.

6.2.4 Ablation Analysis

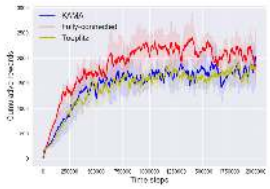
One advantage of KAMA-NN architectures over structured architectures based on low displacement rank matrices (such as Toeplitz), is that it easily allows to adjust the trade-off between the capacity of the



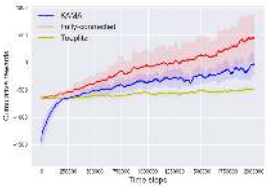
(a) **Swimmer**



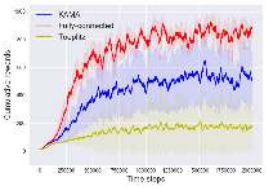
(b) **Walker**



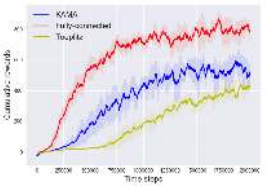
(c) **Hopper**



(d) **HalfCheetah**



(e) **Hopper (R)**



(f) **HalfCheetah (R)**

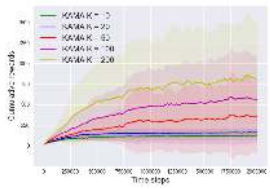
Figure 4: Illustration of KAMA-NNs policies on MuJoCo benchmarks with TRPO. The same setup as for PPO experiments from Fig 3. Experiments with (R) are taken from Roboschool.

model and its compactness by simply varying the number of rotations K. Intuitively, when K is large, the architecture becomes more expensive to use and the performance improves; when K is small, it becomes more compact at the cost of worse performance.

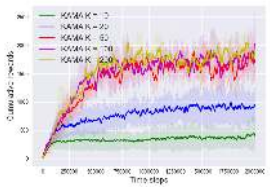
We carry out an ablation study on the effect of the number of rotations K. In Figure 5, we show the training curves of varying $K \in \{10, 20, 50, 100, 200\}$ with both PPO and TRPO on a set of benchmark tasks from OpenAI Gym. We see that the policy performance improves as the number of rotations K increases.

PPO	**HalfCheetah**	**Walker**	**Hopper**
KAMA-NN	15 %	15%	17%
Toeplitz	7 %	7%	8%
TRPO	**HalfCheetah**	**Walker**	**Hopper**
KAMA-NN	24 %	24%	28%
Toeplitz	12 %	12%	13%

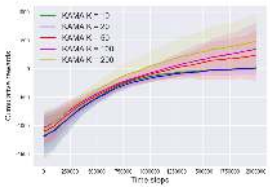
Table 1: Ratio of the total number of parameters used in the structured matrices relative to the unstructured model. KAMA-NN architectures apply $K = 200$ Givens rotations for PPO and $K = 100$ rotations for TRPO.



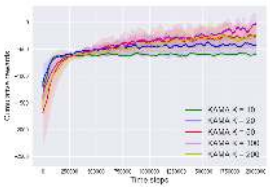
(a) **PPO-Hopper**



(b) **TRPO-Hopper**



(c) **PPO-HalfCheetah**



(d) **TRPO-HalfCheetah**

Figure 5: Ablation study on the effect of the number of rotations K. The experiments are performed with both PPO and TRPO and on a set of benchmark tasks from OpenAI Gym. Vertical axis is the cumulative reward and horizontal axis stands for the # of time steps.

6.3 Learning low-dimensional subspaces for rotations

We also conducted experiments, where not only rotation angles θ, but also 2-dimensional subspaces where rotations were conducted, were learned. We did not observe any quality gains in comparison to the proposed algorithm (where 2-dimensional subspaces were chosen randomly), proving empirically that models with random subspaces and learned angles have sufficient capacity.

7 Conclusions

We presented a new class of compact architectures for feedforward fully connected neural networks based on low dimensional learned or random rotations. We empirically showed their advantages over state of the art in both: adaptive (where the parameters are learned) as well as non-adaptive regime on various tasks such as PNG-kernel approximation and RL policies learning. We further provided theoretical guarantees shedding a new light on the effectiveness of (random) orthogonal compact transforms in machine learning. In particular, we showed that KAMA-NNs lead to asymptotically faster and more accurate estimators of PNG-kernels related to random neural networks. Our architectures provide practitioners with an easy way of adjusting the complexity (and thus also capacity) of the neural network model to their needs by changing the number of low dimensional rotations used (that are building blocks of KAMA-NNs).

Acknowledgements. The authors would like to acknowledge the cloud credits provided by Amazon Web Services.

References

[Abadi et al., 2016] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

[Ailon and Chazelle, 2006] Ailon, N. and Chazelle, B. (2006). Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *STOC*.

[Andoni et al., 2015] Andoni, A., Indyk, P., Laarhoven, T., Razenshteyn, I. P., and Schmidt, L. (2015). Practical and optimal LSH for angular distance. In *NIPS*.

[Bojarski et al., 2017] Bojarski, M., Choromanska, A., Choromanski, K., Fagan, F., Gouy-Pailler, C., Morvan, A., Sakr, N., Sarlos, T., and Atif, J. (2017). Structured adaptive and random spinners for fast machine learning computations. In *AISTATS*.

[Brockman et al., 2016] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

[Chen et al., 2018] Chen, M., Pennington, J., and Schoenholz, S. S. (2018). Dynamical isometry and a mean field theory of rnns: Gating enables signal propagation in recurrent neural networks. *arXiv preprint arXiv:1806.05394*.

[Choromanska et al., 2016] Choromanska, A., Choromanski, K., Bojarski, M., Jebara, T., Kumar, S., and LeCun, Y. (2016). Binary embeddings with structured hashed projections. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 344–353.

[Choromanski et al., 2018a] Choromanski, K., Downey, C., Boots, B., Holtmann-Rice, D., and Kumar, S. (2018a). Initialization matters: Orthogonal predictive state recurrent neural networks. In *to appear at ICLR 2018*.

[Choromanski et al., 2018b] Choromanski, K., Rowland, M., Sarlos, T., Sindhwani, V., Turner, R., and Weller, A. (2018b). The geometry of random features. In *AISTATS 2018*.

[Choromanski et al., 2018c] Choromanski, K., Rowland, M., Sindhwani, V., Turner, R. E., and Weller, A. (2018c). Structured evolution with compact architectures for scalable policy optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 969–977.

[Choromanski and Sindhwani, 2016] Choromanski, K. and Sindhwani, V. (2016). Recycling randomness with structure for sublinear time kernel expansions. In *ICML*.

[Choromanski et al., 2017] Choromanski, K. M., Rowland, M., and Weller, A. (2017). The unreasonable effectiveness of structured random orthogonal embeddings. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 218–227.

[Dhariwal et al., 2017] Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. (2017). Openai baselines. https://github.com/openai/baselines.

[Gallier and Xu, 2003] Gallier, J. and Xu, D. (2003). Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation*, 18(1):10–20.

[Janvresse, 2001] Janvresse, E. (2001). Spectral gap for kac's model of boltzmann equat ion. *The Annals of Probability*, 29(1).

[Kac, 1954] Kac, M. (1954). Foundations of kinetic theory. *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, 3.

[Mania et al., 2018] Mania, H., Guy, A., and Recht, B. (2018). Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*.

[Mathieu and LeCun, 2014] Mathieu, M. and LeCun, Y. (2014). Fast approximation of rotations and hessians matrices. *CoRR*, abs/1404.7195.

[Mischler and Mouhot, 2013] Mischler, S. and Mouhot, C. (2013). Kac's program in kinetic theory. *Inventiones mathematicae*, 193(1).

[Patrascu and Necoara, 2015] Patrascu, A. and Necoara, I. (2015). Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization. *Journal of Global Optimization*, 61(1):19–46.

[Pennington et al., 2017] Pennington, J., Schoenholz, S., and Ganguli, S. (2017). Resurrecting the sigmoid

in deep learning through dynamical isometry: theory and practice. In *Advances in neural information processing systems*, pages 4785–4795.

[Pennington et al., 2018] Pennington, J., Schoenholz, S. S., and Ganguli, S. (2018). The emergence of spectral universality in deep networks. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, pages 1924–1932.

[Pennington and Worah, 2017] Pennington, J. and Worah, P. (2017). Nonlinear random matrix theory for deep learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 2634–2643.

[Pillai and Smith, 2015] Pillai, N. and Smith, A. (2015). Kac's walk on n-sphere mixes in $n \log(n)$ steps. In *arxiv*.

[Rahimi and Recht, 2007] Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. In *NIPS*.

[Salimans et al., 2017] Salimans, T., Ho, J., Chen, X., and Sutskever, I. (2017). Evolution strategies as a scalable alternative to reinforcement learning. *CoRR*, abs/1703.03864.

[Schulman et al., 2015] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897.

[Schulman et al., 2017] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

[Shalit and Chechik, 2014] Shalit, U. and Chechik, G. (2014). Coordinate-descent for learning orthogonal matrices through givens rotations. In *International Conference on Machine Learning*, pages 548–556.

[Sindhwani et al., 2015] Sindhwani, V., Sainath, T. N., and Kumar, S. (2015). Structured transforms for small-footprint deep learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3088–3096.

[Todorov et al., 2012] Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE.

[Vybíral, 2011] Vybíral, J. (2011). A variant of the Johnson-Lindenstrauss lemma for circulant matrices. *Journal of Functional Analysis*, 260(4):1096–1105.

[Xiao et al., 2018] Xiao, L., Bahri, Y., Sohl-Dickstein, J., Schoenholz, S. S., and Pennington, J. (2018). Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. *arXiv preprint arXiv:1806.05393*.

[Yu et al., 2016] Yu, F., Suresh, A., Choromanski, K., Holtmann-Rice, D., and Kumar, S. (2016). Orthogonal random features. In *NIPS*, pages 1975–1983.

[Zhang and Cheng, 2013] Zhang, H. and Cheng, L. (2013). New bounds for circulant Johnson-Lindenstrauss embeddings. *CoRR*, abs/1308.6339.