

Almost Uniform Sampling From Neural Networks

Changlong Wu Narayana Prasad Santhanam
University of Hawaii at Manoa, Honolulu, HI USA
Email: {wuchangl, nsanthan}@hawaii.edu

Abstract—Given a length n sample from $\mathbb{R}^d$ and a neural network with a fixed architecture with W weights, k neurons, linear threshold activation functions, and binary outputs on each neuron, we study the problem of uniformly sampling from all possible labelings on the sample corresponding to different choices of weights. We provide an algorithm that runs in time polynomial both in n and W such that any labeling appears with probability at least $(\frac{W}{2ekn})^W$ for $W < n$. For a single neuron, we also provide a random walk based algorithm that samples exactly uniformly.

I. INTRODUCTION

Consider a sample $\mathbf{x}_1, \dots, \mathbf{x}_n$, where $\mathbf{x}_i \in \mathbb{R}^d$. We have a feedforward neural network with a given architecture (but the weights are unknown). Each sample point $\mathbf{x}_i$ has binary labels, either +1 or -1. Sauer’s lemma provides an upper bound on the number of possible labelings that could be generated by a hypothesis class (the *growth function*) in terms of the VC dimension of the hypothesis class.

We are interested in hypothesis classes corresponding to neural networks with a fixed architecture but unspecified weights. While it is hard to exactly specify the VC dimension of this class, upper bounds on the VC dimension and the growth function are easily derived, see for example [1, Section 6.2]. The growth function for a feedforward, linear threshold network is upper bounded by $(enk/W)^W$, where k is the number of neurons in the network, and W , the number of weights.

Our goal in this paper is to generate labels of the sample uniformly at random from the set of all possible labelings that a given feedforward architecture can provide. We obtain a polynomial time (in both the number of samples and the size of the network), near uniform sampling from arbitrary feedforward networks. In the special case of a single neuron, we also provide a random walk based algorithm for perfectly uniform sampling, and with polynomial mixing time for the random walk.

Aside from the theoretical interest in generating labelings, we are also motivated by questions in property testing. Namely, we want to estimate the statistics of all labelings generated by a given architecture. As an example, we may want to find out the probability that a subset of samples are all labeled the same if all labels were generated at random from the given architecture. In future work, we intend to leverage these insights into better initializations of neural networks while training.

We obtain these results by developing insights on random walks between chambers of intersecting hyperplanes in high dimensions. This is a well studied area, see for example [2]. General arrangements of these hyperplanes intersect in com-

plicated ways, as in our problem, and random walks between these chambers is nontrivial. It is common to visualize the geometry of these arrangements by means of a *chamber graph*, see Chapter 7 of [3] for a synopsis of such chamber graphs. Random walks over hyperplane arrangements appears in contexts quite different from ours. For example, Bidigare, Hanlon and Rockmore modeled card shuffling in [4], with such random walks. Some other applications are in *e.g.*, [5], [6], [7], [8].

The statistics of the random walks considered in the references above is different from ours. Typically, these authors provide an explicit expression to estimate the eigenvalues of the random walk to bound the mixing time. In our paper, we use conductance to understand the mixing properties of our random walk as in [9] and [10].

II. SETUP AND NOTATIONS

We consider a feed-forward linear threshold neural network with L layers. The input to the network is d -dimensional and there is a single binary output label. Namely, i.e. any neuron with parameters $\mathbf{w}, b$, ($\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$) outputs $\sigma(\mathbf{x}^T \mathbf{w} + b)$ on an input $\mathbf{x} \in \mathbb{R}^d$, where $\sigma(u) = 1$ if $u \geq 0$ and $\sigma(u) = 0$ otherwise. In subsequent work, we extend our results to more general activation functions.

Let N be the graph of the feedforward neural network with a fixed architecture and W different parameters (the weights and thresholds put together). Let $\mathbf{W} \in \mathbb{R}^W$, and let $N_{\mathbf{W}}$ be the neural network which assigns the parameters of N to be $\mathbf{W}$. For any given architecture N , let $f_{\mathbf{W}} : \mathbf{x} \in \mathbb{R}^d \rightarrow \{0, 1\}$ be the function expressed by $N_{\mathbf{W}}$.

The vectors $\mathbf{x} \in \mathbb{R}^d$ are the input and $f_{\mathbf{W}}(\mathbf{x})$ are the labels assigned to $\mathbf{x}$. For a length n sample $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d\}$, let

$$S_X = \{(f_{\mathbf{W}}(\mathbf{x}_1), \dots, f_{\mathbf{W}}(\mathbf{x}_n)) \mid \mathbf{W} \in \mathbb{R}^W\}$$

be the set of all labelings that can be generated on X by the architecture N . Note that the set $S_X \subset \{0, 1\}^n$ and for $W < n$, [1, Section 6.2] (or [11])

$$|S_X| \leq \left(\frac{enk}{W}\right)^W.$$

When $W \geq n$, $|S_X| \leq 2^n$, or X is potentially shattered.

Problem For a given architecture N and data X , how can we randomly sample from S_X , in time polynomial in both n and W , such that any labeling $v \in S_X$ appears with probability at least $\Omega(1/|S_X|)$?

a) *Background:* A hyperplane in $\mathbb{R}^d$ (or a hyperplane in d dimensions) is the set of all points $\mathbf{w} \in \mathbb{R}^d$ satisfying $\mathbf{x}^T \mathbf{w} = 0$ for some fixed vector $\mathbf{x} \in \mathbb{R}^d$. Let N be a single neuron with input dimension d . As before, $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d\}$ is a length n sample.

Let $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$. Physically, the vector in $d+1$ dimensions, $(\mathbf{w}, b) \in \mathbb{R}^{d+1}$ defines the parameters of the single neuron N . For each sample point $\mathbf{x}_i \in \mathbb{R}^d$, define P_i to be the hyperplane in the parameter space $\mathbb{R}^{d+1}$:

$$\mathbf{x}_i^T \mathbf{w} + b = 0.$$

We start with a visualization from [1].

Theorem 1. *All parameter vectors that belong to the same connected component of $\mathbb{R}^{d+1} \setminus \bigcup_i P_i$ label X in the same way. Conversely, different components have different labelings on X .*

We recall a few standard terms regarding hyperplane arrangements formed by $P_1, \dots, P_n$.

- The connected components in $\mathbb{R}^d \setminus \bigcup_{i=1}^n P_i$ are called *chambers* (or regions).
- The *chamber graph* is constructed as follows: assign a vertex to every chamber. Two vertices are connected if their associated chambers share a common face.
- Any hyperplane arrangement is *centered* if the intersection of the component hyperplanes contains the origin. In our case, $\bigcap_{i=1}^n P_i$ always contains the origin, i.e., the samples generate a *centered* arrangement in the parameter space.
- A collection of n centered hyperplanes in $\mathbb{R}^{d+1}$ is in *general position*, if for all $k \leq d+1$, every intersection of k distinct hyperplanes forms a $d+1-k$ -dimension linear space, and any intersection of more than $d+1$ hyperplanes is contains only the origin. Randomly chosen planes are in general position almost surely.

b) *Pseudo polynomial optimal training algorithm:* A theoretically useful framework was introduced in Theorem 4.1 of [12] for ReLU networks, where the network size W is treated as a constant, and we look at the dependency purely on the sample size n (thereby treating n^W as a polynomial).

We note that our near-uniform polynomial time sampling procedure implies a probabilistic, pseudo-polynomial training algorithm that attains the global minimum for any feedforward linear threshold neural network. This implication is immediate from the coupon collector problem—since given any confidence, generating at most $O(n^W \log n)$ samples guarantees that we have seen every possible labeling that can be produced.

III. PROPERTIES OF HYPERPLANE ARRANGEMENTS

We summarize a few useful properties of hyperplane arrangements that we will use in our arguments in the paper.

Proposition 1 ([1, Theorem 3.1]). *The number of chambers in a centered hyperplane arrangement formed by n hyperplanes in d dimensions in the general position is*

$$2 \sum_{i=0}^{d-1} \binom{n-1}{i}.$$

In fact, even sampling all labels of a sample of size n , even when the network consists of a single neuron, in time polynomial in both n and dimension d of the data points, is non-trivial. The number of chambers by Theorem 1 is the number of labels on a size- n sample, which from the above Proposition is roughly $O(n^d)$. Clearly, trivial enumeration of labels is out of question. As we will see later in Section IV-A, this is not the only difficulty even for a single neuron.

Proposition 2. *Let $Q_1, \dots, Q_n$ form a centered hyperplane arrangement in d dimensions. Let $\mathbf{v}_i \in \mathbb{R}^d$ be any vector normal to the hyperplane Q_i . If $\mathbf{v}_1, \dots, \mathbf{v}_n$ have rank r , then any chamber in the hyperplane arrangement has at least r faces.*

Proof. Let $V = \{\mathbf{v}_1, \dots, \mathbf{v}_n, -\mathbf{v}_1, \dots, -\mathbf{v}_n\}$. Suppose the proposition is false. Then there exists a chamber with exactly $b < r$ faces. Without loss of generality, let $\mathbf{u}_1 \in V, \dots, \mathbf{u}_b \in V$ be the normal vectors of the b different faces of this chamber respectively, such that for any point $\mathbf{x}$ within the chamber

$$\mathbf{u}_i^T \mathbf{x} > 0.$$

Since the rank of $\mathbf{v}_1, \dots, \mathbf{v}_n$ is $r > b$, we can choose a vector $\mathbf{u}_{b+1} \in V$ such that $\mathbf{u}_{b+1}$ is linearly independent from $\mathbf{u}_1, \dots, \mathbf{u}_b$.

We now show that the hyperplane that determined by $\mathbf{u}_{b+1}$ is also a face of the chamber by proving that there is a point $\mathbf{x}'$ in the chamber satisfying

$$\mathbf{u}_i^T \mathbf{x}' > 0 \quad 1 \leq i \leq b \quad \text{and} \quad \mathbf{u}_{b+1}^T \mathbf{x}' = 0. \quad (1)$$

Since $\mathbf{u}_{b+1}$ is linearly independent of $\mathbf{u}_1, \dots, \mathbf{u}_b$, we can choose a vector $\mathbf{y}$ such that $\mathbf{y}^T \mathbf{u}_i = 0$ for $1 \leq i \leq b$ but $\mathbf{y}^T \mathbf{u}_{b+1} \neq 0$. Now let $\mathbf{x}$ be any point in the chamber and set $\mathbf{x}' = \mathbf{x} + t\mathbf{y}$ where $t = -\mathbf{u}_{b+1}^T \mathbf{x} / \mathbf{u}_{b+1}^T \mathbf{y}$. It is easy to verify now that $\mathbf{x}'$ satisfies (1). This contradicts the assumption that the chamber contained b faces, where $b < r$. $\square$

Proposition 3. *The chamber graph of any hyperplane arrangement $Q_1, \dots, Q_n$ in $\mathbb{R}^d$ in general position satisfies (i) the degree of any vertex is at least d and at most n , and (ii) any pair of vertices has graph distance at most n .*

Proof. (i) from Proposition 2, (ii) from [3, Lemma 7.15]. $\square$

IV. SAMPLING LABELINGS

For the sample $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, where $\mathbf{x}_i \in \mathbb{R}^d$, S_X is the set of all possible labels generated on X by the network N . We would like to sample from S_X uniformly.

In Section IV-A, we let N be a single neuron and even this turns out to be non-trivial. Inspired by the inductive approach for computing hyperplane partition number [2, Chapter 2], we derive Algorithm RS (for Recursive Sampling) in Section IV-A that generates a label from S_X almost uniformly.

In Sections IV-B and IV-C we expand in two directions. In Section IV-B, we provide means to perfectly sample from all labelings of a single neuron using a random walk on S_X with a perfectly uniform stationary distribution. This allows us to sample from S_X perfectly uniformly. The mixing time of this random walk is as yet unproven, but we provide partial

evidence (empirical as well as proofs for small dimensions) that this random walk is fast mixing, with mixing time at most linear in the number of dimensions and at most quadratic in the number of samples.

In Section IV-C, we build on our RS approach to sample from arbitrary feedforward networks in time (true) polynomial in the sample size n , network size W and input dimension (d), showing that even for arbitrary networks, we get near-uniform sampling of the possible labels that could be produced by the network.

A. The recursive approach

From Theorem 1, to sample uniformly from S_X we only need to sample the weights uniformly from the connected components of $\mathbb{R}^{d+1} \setminus \bigcup_i P_i$.

However, even for a single neuron, this is not trivial. As already noted from Proposition 2, the basic combinatorial difficulty comes from the fact that there are roughly n^d labelings for almost all samples X —therefore the number of chambers is exponential in the dimension d . Clearly one can not simply enumerate all the possible components.

But a bigger difficulty comes from the fact that the arrangement of the hyperplanes can be very heterogeneous. The volume of some of the chambers can be arbitrarily small and therefore such chambers may be difficult to find. We settle this problem by using a recursive sampling approach that is inspired by the inductive approach for computing hyperplane partition number.

Our recursive algorithm $RS(\mathbf{v}_1, \dots, \mathbf{v}_k)$ (see Algorithm 1) takes as its inputs k unit vectors $\mathbf{v}_1, \dots, \mathbf{v}_k$, all from, say $\mathbb{R}^m$. The vectors $\mathbf{v}_1, \dots, \mathbf{v}_k$ are interpreted as normal vectors of k distinct centered hyperplanes in $\mathbb{R}^m$. For simplicity, the reader can assume that these hyperplanes are in the general position, but they do not have to be. To sample from S_X , we would therefore simply call $RS(\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n)$, where $\tilde{\mathbf{x}}_i = \frac{(\mathbf{x}_i, -1)}{\|(\mathbf{x}_i, -1)\|_2} \in \mathbb{R}^{d+1}$ and $\mathbf{x}_i \in X$.

The call $RS(\mathbf{v}_1, \dots, \mathbf{v}_k)$ works recursively on the dimension m of the vectors $\mathbf{v}_i$ and k , by calling RS with a new set of vectors $\mathbf{u}_1, \dots, \mathbf{u}_{k'}$ in $\mathbb{R}^{m-1}$ with $k' \leq k-1$. The base case is when RS is called with vectors in 1 dimension or when $k = 1$. When RS is called with vectors in 1 dimension, the problem is trivial since there is only one centered hyperplane arrangement in 1-dimension, the origin. When RS is called with $k = 1$ (no matter the dimension of the single input vector), the problem is also trivial since there are only two chambers for one hyperplane.

To generate the vectors $\mathbf{u}_i$ in $\mathbb{R}^{m-1}$, we choose a hyperplane at random from $\mathbf{v}_1, \dots, \mathbf{v}_k$, say $\mathbf{v}_I$, and compute the intersection of $\mathbf{v}_I$ with all the remaining hyperplanes. These intersections are at most $k-1$ hyperplanes in $\mathbb{R}^{m-1}$ and let $\mathbf{u}_1, \dots, \mathbf{u}_{k'}$ be the unit normal vectors of these hyperplanes (written in the specific orthonormal basis indicated).

Theorem 2. Let $V = \{\mathbf{v}_1, \dots, \mathbf{v}_k\}$ where $\mathbf{v}_i \in \mathbb{R}^m$ and rank of V is m . Let C_V be the set of non-empty chambers induced by the k centered hyperplanes orthogonal to the vectors in

Algorithm 1 $RS(\mathbf{v}_1, \dots, \mathbf{v}_k)$

Input: $\mathbf{v}_1, \dots, \mathbf{v}_k \in \mathbb{R}^m$, interpreted as unit normal vectors of k (distinct) centered hyperplanes in an m -dimensional space.

Output: point $\mathbf{y} \in \mathbb{R}^m$ representing a chamber in the hyperplane arrangement formed by $\mathbf{v}_1, \dots, \mathbf{v}_k$.

Let P_i be the hyperplane in $\mathbb{R}^m$ orthogonal to $\mathbf{v}_i$.

1. If $m = 1$ output -1 or 1 with equal probability. If $m > 1$ but $k = 1$, output $\mathbf{v}_1$ or $-\mathbf{v}_1$ with equal probability.
 2. Uniformly choose an index I from $\{1, \dots, k\}$.
 3. For hyperplane P_I , choose an arbitrary orthonormal basis $B \in \mathbb{R}^{(m-1) \times m}$. Note that P_I is a $(m-1)$ -dimensional linear space in $\mathbb{R}^m$, and the $m-1$ rows of B contain the orthonormal basis vectors, each being a vector in $\mathbb{R}^m$.
 4. Compute the intersection of P_I with P_j , $j \in \{1, \dots, k\} \setminus \{I\}$.
 5. Set $\mathbf{v}'_j$ to be the unit vector in P_I normal to $P_I \cap P_j$ (written using the basis B), $j \in \{1, \dots, k\} \setminus \{I\}$. Note $\mathbf{v}'_j \in \mathbb{R}^{m-1}$.
 6. $\mathbf{x} = RS(\mathbf{u}_1, \dots, \mathbf{u}_{k'})$, where $\mathbf{u}_1, \dots, \mathbf{u}_{k'}$ are the distinct vectors among $\{\mathbf{v}'_j \mid j \neq I\}$. Note $k' \leq k-1$.
 7. Compute the smallest distance δ of $\mathbf{x}^T B$ to the planes P_j with $j \neq I$.
 8. Let t be -1 or 1 with equal probability, output $\mathbf{y} = \mathbf{x}^T B + t\delta \mathbf{v}_I$.
-

V. Algorithm $RS(\mathbf{v}_1, \dots, \mathbf{v}_k)$ runs in $O(km^3)$ time and any chamber in the hyperplane arrangement induced by V is sampled with probability at least

$$\frac{1}{2^m \binom{k}{m}} \geq \left(\frac{m}{2ek}\right)^m, \text{ where } e \text{ is the base of nature logarithm.}$$

Proof. (Outline only) The algorithm will run at most m recursive iterations. For each iteration, we need $O(m^2)$ to compute the base of the null space (Step 3) and $O(km^2)$ time in Step 7 to compute the projection of each input vectors to the plane chosen in Step 2. This yields the total complexity to be $O(km^3)$.

To see the probability lower bound, define

$$p(m, k) = \min_{V, c \in C_V} \Pr[RS(V) = c], \text{ with rank}(V) = m \text{ and } |V| \leq k.$$

We now claim that

$$p(m, k) \geq \frac{m}{2k} p(m-1, k-1).$$

This is because any chamber $c \in C_V$ has at least m faces by Proposition 2. For any chamber $c \in C_V$, we therefore have probability at least $\frac{m}{2k}$ of choosing both a hyperplane that forms the face of c and the direction of the hyperplane that faces the chamber c . Conditioned on this choice of hyperplane and direction, we need to obtain the probability that the recursive call in step in Step 6 returns a point in the face of c .

Observe that the face of c is a $m-1$ -dimensional linear space. In Step 6, note that the rank of $\{\mathbf{u}_1, \dots, \mathbf{u}_{k'}\}$ is exactly $m-1$, but k' can be less than $k-1$. The theorem follows by

Algorithm 2 NRW

Input: walk length T and hyperplanes $P_1, \dots, P_n$ in $\mathbb{R}^m$ **Output:** point $\mathbf{w} \in \mathbb{R}^m$ and chamber c

1. Initialize $\mathbf{w}_0 = RS(\mathbf{v}_1, \dots, \mathbf{v}_n)$, where $\mathbf{v}_i$ is a normal vector of P_i
 2. Set $c_0 = \text{Chamber}(\mathbf{w}_0)$. c_0 will be the chamber in the arrangement $\{P_i\}$ that contains $\mathbf{w}_0$.
 3. For $t = 1$ through T , do
 - a. Uniformly choose a face of c_{t-1}
 - b. Set c_t to be the chamber adjacent to c_{t-1} and across the face chosen in step (a.)
 - c. Set $\mathbf{w}_t$ to any point in the chamber c_t
 4. Output $\mathbf{w}_T$ and c_T
-

solving the recursive inequality, standard approximations on binomial coefficient and by noting that when $m = 1$, there are two chambers, thus yielding $p(1, k) = 1/2$ for all k . $\square$

Note that when $\text{rank}(X) = d$ the above probability is $\mathcal{O}\left(\frac{d!}{2^d n^d}\right)$, a factor $\frac{1}{2^d}$ off the hyperplane slicing bound $2 \sum_{i=0}^{d-1} \binom{n-1}{i}$ in Proposition 1. Note also that if the input vectors in $\mathbb{R}^d$ have rank $m < d$, the above approach still works. We can effectively project down the inputs into $\mathbb{R}^m$ by choosing a basis for $\mathbb{R}^d$ that contains $d - m$ vectors that are orthogonal to the span of the input vectors.

B. A random walk approach

To mitigate the fact that the recursive approach above only yields approximately uniform sampling, We introduce a random walk based algorithm that samples arbitrarily close to uniform. Specifically, we run Algorithm NRW on a *lazy chamber graph*, both outlined below. One component of Algorithm NRW is Algorithm Chamber, that determines which chamber an input point belongs to.

Theorem 3. *Algorithm Chamber runs in polynomial time both on d and n .*

Proof. The theorem follows since linear programming can be solved in polynomial time [13]. $\square$

a) Analysis: We first analyze random walk defined by Algorithm NRW over the simple chamber graph, assuming the hyperplanes are in general position. With this assumption any vertex in the *chamber graph* has degree at least d and at most

Algorithm 3 Chamber

Input: point $\mathbf{w} \in \mathbb{R}^m$ and hyperplanes $P_1, \dots, P_n$ **Output:** The faces $P_{i_1}, \dots, P_{i_k}$ of the chamber containing $\mathbf{w}$.

1. Compute $\sigma_i = \text{sign}(\mathbf{w}^T \mathbf{v}_i)$.
 2. For $1 \leq i \leq n$ do:
 3. Define a linear program with $\mathbf{w}^T \mathbf{v}_i = 0$ and $\sigma_j(\mathbf{w}^T \mathbf{v}_j) > 0$ for $j \neq i$.
 4. If the linear programming in step 3 has a solution, add P_i to the collection.
-

n from Proposition 3. Furthermore, from Proposition 3 the graph is connected and the distance between any two vertexes is at most n .

Since the random walk is a reversible Markov chain, the stationary distribution π of the random walk will be proportional to the degree of the vertices [9, Chapter 1.6]. From our observation on the bounds of degrees in Proposition 3, we will therefore have for any two vertices u and v

$$\frac{d}{n} \leq \frac{\pi(u)}{\pi(v)} \leq \frac{n}{d}.$$

The more fundamental question is the mixing time of the random walk, or how quickly the walk generates stationary samples. While there are several approaches to analyze the mixing time, we focus on Cheeger's inequality [9, Theorem 13.14] that bounds the spectral gap of the random walk's transition matrix using the *conductance* of the graph. Recall that the conductance of a graph is

$$\min_{A \subset V, \text{vol}|A| \leq \frac{1}{2} \text{vol}|V|} \frac{|\partial A|}{\text{vol}|A|},$$

where V is the vertex set, ∂A is size of the cut between A and $V \setminus A$, $\text{vol}|A|$ is the sum of degrees of vertexes in A . The following theorem gives a lower bound on the conductance of chamber graph when dimension $d = 2$.

Theorem 4. *The chamber graph of 2-dimensional hyperplane arrangement with size n that is in the general position has conductance lower bounded by $\frac{1}{2n}$.*

Proof. For any set A of vertices in the chamber graph with size no greater than $\frac{1}{2}|V|$, we will show that the conductance of A , $\frac{|\partial A|}{\text{vol}|A|}$, is lower bounded as follows

$$\frac{|\partial A|}{\text{vol}|A|} \geq \frac{1}{2n}.$$

Let X be the set with smallest volume satisfying

$$X = \arg \min_{\substack{A \\ \text{vol}(A) \leq \frac{1}{2} \text{vol}(V)}} \frac{|\partial A|}{\text{vol}|A|}.$$

We first claim that X must be connected. If not, we can write X as the union of (maximally) connected components, i.e., $X = \bigcup_{i=1}^r X_i$, where X_i are the maximally connected components within X (in particular, note that there are no edges between distinct X_i). Then, if $a_i = |\partial X_i|$ and $b_i = \text{vol}(X_i)$, then

$$\frac{|\partial X|}{\text{vol}|X|} = \frac{a_1 + a_2 + \dots + a_r}{b_1 + b_2 + \dots + b_r} \geq \min_{i \in [r]} \frac{a_i}{b_i},$$

implying that X_i has lower conductance than X and is smaller in size than X , a contradiction.

Let S be the boundary surface of the chambers corresponding to vertexes in X . Since X is connected, we must have S to be piece-wise line segments.

We now claim that S will partition the chamber graph into two connected components. Since X is connected, we just have to show that $V \setminus X$ is also connected.

Suppose not, and let $V \setminus X = \bigcup_{i=1}^m Y_i$, where Y_i are maximally connected, and $V \setminus X$ is the union of m different connected components. Let $c_i = |\partial Y_i|$ and $d_i = \text{vol}(Y_i)$. Then we have

$$\sum_{i=1}^m c_i = |\partial X|,$$

and since $\text{vol}(V \setminus X) = \text{vol}(V) - \text{vol}(X)$ and $\text{vol}(X) \leq \frac{1}{2} \text{vol}(V)$, we have

$$\sum_{i=1}^m d_i \geq \frac{1}{2} \text{vol}(V) \geq \text{vol}(X).$$

Therefore, there must be some component i such that

$$\frac{c_i}{d_i} \leq \frac{\sum c_i}{\sum d_i} \leq \frac{|\partial X|}{\text{vol}(X)}.$$

If Y_i satisfies $\text{vol}(Y_i) \leq \frac{1}{2} \text{vol}(V)$, then again we have a contradiction because of the following. If $c_i/d_i < \frac{|\partial X|}{\text{vol}(X)}$, we are done. If $c_i/d_i = \frac{|\partial X|}{\text{vol}(X)}$, it means that every component in $V \setminus X$ has conductance $\frac{|\partial X|}{\text{vol}(X)}$. But if there are more than two components in $V \setminus X$, then X has a larger cut ∂X than each of the components, and therefore must have a larger volume as well, contradicting the assumption on X .

If $\text{vol}(Y_i) \geq \frac{1}{2} \text{vol}(V)$, then consider the set $Z = V \setminus Y_i$. Note that

$$Z = X \bigcup (\bigcup_{j \neq i} Y_j).$$

Now $|\partial Z| = |\partial Y_i| \leq |\partial X|$. This follows since there is no boundary between Y_i and any of the other Y_j , and the only boundary Y_i has is with X . Furthermore, $\text{vol}(Z) > \text{vol}(X)$, implying that Z has lower conductance than X , again a contradiction.

Now, we know that the boundary S between the chambers in X and the rest of the hyperplane arrangement is exactly a piece-wise line segment that separates $\mathbb{R}^2$ into two connected components. There are only 3 possibilities, as shown in Figure 1. We now observe $\text{vol}(X)$ is exactly the sum of the 1-dimensional faces in the arrangement that intersect with X . Since there are at most n lines in the arrangement, there exist a line P that intersect with X (or $V \setminus X$) by at least $\frac{\text{vol}(X)}{n}$ many faces, see figure 1. The number of faces in S is no less than the number of faces in P , because any line that intersects with P in X must also intersect with S , and at most two lines can intersect at the same point on S by our general position assumption. The theorem now follows. $\square$

For the general dimension case, we have the following conjecture. See Appendix for justification and partial proofs.

Conjecture 1. *The conductance of any d -dimensional general position hyperplane arrangement of size n is lower bounded by $\frac{1}{\text{poly}(n,d)}$.*

Remark 1. *Note that the requirement for general position of the hyperplanes is necessary for fast mixing given by the Conjecture above. Else it is easy to construct a hyperplane*

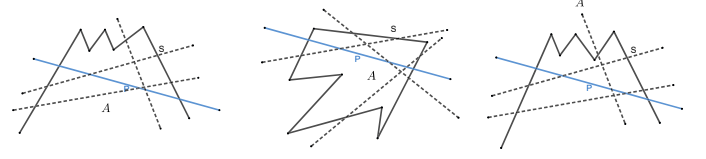


Fig. 1. Possibility of piece-wise linear partition

arrangement with mixing time lower bounded by $O(\frac{n^d}{2^d})$. As shown in Figure 2, the cut made by the gray shaded top plane has only 4 boundary chamber but the total number of chambers below the plane is roughly n^2 (in two dimensions, while in $d+1$ dimensions, we will have the cut and volume to be 2^{d+1} and $O(n^d)$ respectively).

b) *Lazy Chamber graph:* Algorithm NRW on the regular chamber graph will not give an *exact* uniform sampling, but is off by a factor of d/n as mentioned above. This is easily fixed by adding dummy vertices and dummy edges to each vertex in the chamber graph raising the degree of every vertex in the original chamber graph to $4n$. Call such a graph to be *lazy chamber graph*.

We will call the vertex in the original chamber graph to be *chamber vertex* and the dummy vertices added to be *augmentation vertices*. The stationary probability of the new random walk, restricted on the chamber vertices, is exactly uniform. If the Algorithm NRW on the chamber graph is fast mixing, we can show that Algorithm NRW on the lazy chamber graph is also fast mixing:

Lemma 1. *If the conductance of the chamber graph is g , the lazy chamber graph has conductance $\geq \frac{g}{8n^2}$.*

Proof. We only need to show that any subset A of vertex in the lazy chamber graph we have $\frac{|\partial A|}{|A|} \geq \frac{g}{8n}$. We observe that if an augmentation vertex is in A , then the chamber vertex attached to it must also be included in A . We denote $A' \subset A$ to be the set of all chamber vertexes in A .

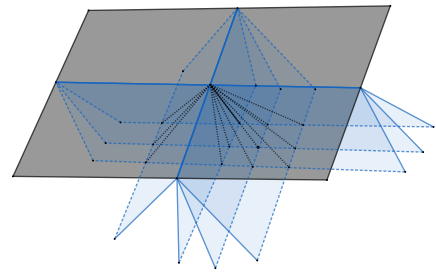


Fig. 2. Hyperplane arrangement with small conductance

The vertices in A' can be partitioned into two classes, $A' = B' \cup C'$ where B' is the set of all chamber vertices that have all their attached augmentation vertices in A and C' is the complement of B' in A' . Similarly, B, C to be the sets that contains also the attached new vertex of B' and C' in A). We have

$$\frac{|\partial A'|}{|A'|} = \frac{|\partial B'| + |\partial C'|}{|B'| + |C'|} \geq \frac{|\partial B'|}{|B'|},$$

since all vertices in C' are boundary vertexes. Note that $3n * |B'| \leq |A| \leq \frac{4n * |V|}{2}$ since any vertex in B' will attach at least $3n$ new vertex in order to make degree $4n$, we have $|B'| \leq \frac{2}{3}|V|$. Now, by the definition of conductance we have $\frac{|\partial B'|}{|B'|} \geq g/2$. This is because, if $|B'| \leq \frac{|V|}{2}$ then $\frac{|\partial B'|}{|B'|} \geq g$. Otherwise, we have $\frac{1}{3}|V| \leq A' \setminus B' \leq \frac{1}{2}|V|$, thus $|\partial B'| \geq g|A' \setminus B'| \geq \frac{1}{3}g|V|$ and $|B'| \leq \frac{2}{3}|V|$, we have $\frac{|\partial B'|}{|B'|} \geq \frac{g}{2}$. Therefore, we have

$$\frac{|\partial A|}{|A|} \geq \frac{|\partial B'| + |\partial C|}{|B| + |C|} \geq \frac{|\partial B'|}{|B|} \geq \frac{|\partial B'|}{4n * |B'|} \geq \frac{g}{8n}.$$

Now since $\text{vol}|A| \leq n|A|$, the theorem follows. $\square$

Combining all the results, we have

Theorem 5. *Assuming conjecture 1. For an given parameter $\epsilon > 0$ and X in the general position, Algorithm NRW run on the lazy chamber graph generated by S_X can generate labels from S_X with distribution ϵ close (in variational distance) to uniform, and runs in time $\text{poly}(d, n, \log(1/\epsilon))$.*

Proof. By the relationship between mixing time and spectral gap [10, Theorem 2.2], we have

$$t_{\text{mix}}(\epsilon) \leq \frac{1}{g} \log \left(\frac{1}{2\epsilon n^d} \right).$$

The theorem follows since the spectral gap is lower bounded by square of conductance by Cheeger's inequality [9, Theorem 13.14]. $\square$

C. Sampling for arbitrary neural networks

We now consider the sampling for arbitrary neural networks. Let $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ be the samples, we choose the weights of the network layer by layer. At layer ℓ we use the previous sampled weights in layers $1, \dots, \ell - 1$ to generate outputs $\mathbf{x}_1^\ell, \dots, \mathbf{x}_n^\ell$, where $\mathbf{x}_i^\ell$ is output of layer $\ell - 1$ with input $\mathbf{x}_i$, a binary vector. For each neuron in layer ℓ we *independently* sample weights using Algorithm RS with input $\{(1, \mathbf{x}_1^\ell), \dots, (1, \mathbf{x}_n^\ell)\}$.

To illustrate the idea more concretely, consider neural networks with one hidden layer. Let X to be the input samples of dimension d , for each neuron in the hidden layer, we use Algorithm RS to generate the weights *independently*. We now fix the weights we sampled for the neuron in the hidden layer and view the function that expressed by the hidden layer to be some function $h := \mathbb{R}^d \rightarrow \{0, 1\}^{u_2}$, where u_2 is the number of neurons in the hidden layer. We now define $\mathbf{x}_i' = h(\mathbf{x}_i)$ to be the new input sample for the output layer, and again use

Algorithm RS to sample the weights for the output neuron with input X' .

Theorem 6. *For a neural network with fixed architecture, k neurons and W parameters, the above sampling procedure runs in $O(nW^3)$ time. Given a sample X , each labeling in S_X produced by this architecture appears with probability at least $(\frac{W}{2enk})^W$.*

Proof. We use induction on the layers. For any given labeling produced by weights $\mathbf{w}$, let $p(\ell)$ to be the probability that the output of layer ℓ is consistent with the output on weight $\mathbf{w}$. We have

$$p(\ell) \geq p(\ell - 1) \prod_{i=1}^{u_\ell} \left(\frac{d_i}{2en} \right)^{d_i},$$

where d_i is the input dimension of the i th neuron in layer ℓ , and the product term comes from Theorem 2 and independence. Note that the rank of the outputs X^ℓ may reduced after passing the previous layers, however, this will only make the probability larger than $(\frac{d_i}{2en})^{d_i}$ by Theorem 2. Now, the theorem follows with the same argument as in [1, Theorem 6.1] for bounding VC dimension of linear threshold neural networks. $\square$

REFERENCES

- [1] M. Anthony and P. L. Bartlett, *Neural network learning: Theoretical foundations*. Cambridge university press, 2009.
- [2] R. P. Stanley *et al.*, "An introduction to hyperplane arrangements," *Geometric combinatorics*, vol. 13, pp. 389–496, 2004.
- [3] S. Ovchinnikov, *Graphs and cubes*. Springer Science & Business Media, 2011.
- [4] P. Bidigare, P. Hanlon, D. Rockmore *et al.*, "A combinatorial description of the spectrum for the tsetlin library and its generalization to hyperplane arrangements," *Duke Mathematical Journal*, vol. 99, no. 1, pp. 135–174, 1999.
- [5] K. S. Brown and P. Diaconis, "Random walks and hyperplane arrangements," *Annals of Probability*, pp. 1813–1854, 1998.
- [6] C. A. Athanasiadis and P. Diaconis, "Functions of random walks on hyperplane arrangements," *Advances in Applied Mathematics*, vol. 45, no. 3, pp. 410–437, 2010.
- [7] J. Pike, "Eigenfunctions for random walks on hyperplane arrangements," Ph.D. dissertation, University of Southern California, 2013.
- [8] A. Björner, "Random walks, arrangements, cell complexes, greedoids, and self-organizing libraries," in *Building bridges*. Springer, 2008, pp. 165–203.
- [9] D. A. Levin and Y. Peres, *Markov chains and mixing times*. American Mathematical Soc., 2017, vol. 107.
- [10] N. Berestycki, "Mixing times of markov chains: Techniques and examples," *Alea-Latin American Journal of Probability and Mathematical Statistics*, 2016.
- [11] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [12] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee, "Understanding deep neural networks with rectified linear units," *arXiv preprint arXiv:1611.01491*, 2016.
- [13] N. Megiddo *et al.*, *On the complexity of linear programming*. IBM Thomas J. Watson Research Division, 1986.
- [14] V. Koltun, "The arrangement method for linear programming," *Computer Science Department, Stanford University*, 2005.

APPENDIX

In order to provide a convincing reason as to why we believe the Conjecture 1, we provide a proof of the following partial result: suppose we cut a general position hyperplane arrangement with another hyperplane. The conductance of such a cut is lower bounded by $\Omega(1/n^2)$ (no matter the number of dimensions).

Proposition 4. Let $Q_1, \dots, Q_n$ be a general position hyperplane arrangement in dimension d . P is another hyperplane. Then the number of chambers in $Q_1 \cap P, \dots, Q_n \cap P$ (viewed as a hyperplane arrangement in P) is lower bounded by

$$\frac{1}{n} \binom{n-1}{d-1}.$$

Proof. A set of hyperplane in dimension d is said to be in *almost full rank position* if any $k \leq d$ planes has rank at least $k-1$. Note that the hyperplanes $Q_1 \cap P, \dots, Q_n \cap P$ are in *almost full rank position*, since the projection on to P can only reduce the rank by 1. Note that there may be two $Q_i \cap P$ and $Q_j \cap P$ coincident, but we treat them as different planes.

Denote $Q'_i = Q_i \cap P$, we show that the intersection $\{Q'_j \cap Q'_1\}$ for $j \neq 1$ is also in *almost full rank position*. We only need to show that any $k \leq d-2$ planes has rank at least $k-1$.

Suppose not, w.l.o.g. $Q'_2 \cap Q'_1, \dots, Q'_{k+1} \cap Q'_1$ has rank at most $k-2$ and $k \leq d-2$.

But we will show that $Q'_1, \dots, Q'_{k+1}$ will have rank $k-1$, thus obtaining a contradiction. To see this, let B be the base of the linear space Q'_1 , v_i a normal vector to Q'_i . We have $Bv_1 = 0$ and $\lambda_2 Bv_2 + \dots + \lambda_{k+1} Bv_{k+1} = 0$, implying that $B(\lambda_2 v_2 + \dots + \lambda_{k+1} v_{k+1}) = 0$, which in turn implies $\lambda_2 v_2 + \dots + \lambda_{k+1} v_{k+1} = \lambda_1 v_1$ for two different set of λ_i s. Meaning that $v_1, \dots, v_{k+1}$ has rank of $k-1$.

The proposition now follows by induction. $\square$

a) *A random walk on vertexes:* For hyperplane arrangement $Q_1, \dots, Q_n$ that is in the general position. We define the vertexes of the arrangement to be all the intersections $v_I = \bigcap_{i \in I} Q_i$ with $I \subset [n]$ and $|I| = d$. By the general position assumption, we know that there are exactly $\binom{n}{d}$ many vertexes. Two vertexes $v_I, v_{I'}$ is said to be connected if they are connected by a 1-dimensional face (intersection by $d-1$ hyperplanes) of the hyperplane arrangement. There are $(n-d)\binom{n}{d-1}$ many edges and each vertex adjacent to at most $2d$ and at least d edges. The graph that defined by the vertexes and edges is known as the *arrangement graph* and studied in [14]. Where the author obtained the following conductance bound using a coupling argument:

Theorem 7 ([14, Theorem 4.3]). *The conductance of the arrangement graph is lower bounded by*

$$\Omega\left(\frac{n-d}{n^3 \log n}\right).$$

Note that theorem 1 will implies conjecture 1 if we also know that the number of vertexes in any cut of the chamber do not much greater than the number of faces. Proposition 1 shows that this is satisfied if the cut is a plane, since there are exactly $\binom{n}{d}$ vertices but at least $\frac{1}{n} \binom{n-1}{d-1}$ faces).