MCMC Learning

Varun Kanade
University of California, Berkeley
vkanade@eecs.berkeley.edu

Elchanan Mossel*
University of California, Berkeley
mossel@stat.berkeley.edu

May 7, 2014

Abstract

The theory of learning under the uniform distribution is rich and deep. It is connected to cryptography, computational complexity, and analysis of boolean functions to name a few areas. This theory however is very limited in applications due to the fact that the uniform distribution and the corresponding Fourier basis is rarely encountered as a statistical model.

A family of distributions that vastly generalizes the uniform distribution on the Boolean cube is that of distributions represented by Markov Random Fields (MRF). Markov Random Fields are one of the main tools for modeling high dimensional data in all areas of statistics and machine learning.

In this paper we initiate the investigation of extending central ideas, methods and algorithms from the theory of learning under the uniform distribution to the setup of learning concepts given examples from MRF distributions. In particular, our results establish a novel connection between properties of MCMC sampling of MRFs and learning under the MRF distribution.

1 Introduction

The theory of learning under the uniform distribution is well developed and has rich and beautiful connections to discrete Fourier analysis, computational complexity, cryptography and combinatorics to name a few areas. However, these methods are rarely used in practice since it is almost never the case that examples are drawn from the uniform distribution over the Boolean cube. In this paper we extend ideas, techniques and algorithms from this theory to a much broader family of distributions, namely, to Markov Random Fields.

1.1 Learning Under the Uniform Distribution

Since the seminal work of [Linial et al. (1993)], the study of learning under the uniform distribution has developed into a major area of research. A key feature of learning

*Supported by DOD ONR grant N000141110140 and NSF grant CCF 1320105
under the uniform distribution is the simple and explicit Fourier expansion of functions defined on the boolean cube ($\{-1, 1\}^n$):

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S(x), \quad \chi_S(x) = \prod_{i \in S} x_i.$$ 

This connection allows a rich class of algorithms that are based on learning coefficients of $f$ for several classes of functions. Moreover, this connection allows application of sophisticated results in the theory of Boolean functions including hyper-contractivity, number theoretic properties and invariance, e.g. [O’Donnell et al., 2005, Shpilka and Tal, 2011, Servedio et al., 2012].

On the other hand, the central role of the uniform distribution in computational complexity and cryptography results in deep and beautiful results relating learning under the uniform distribution to key themes in theoretical computer science including de-randomization, hardness and cryptography, e.g. [Klivans and Sherstov, 2009, Alekhnovich et al., 2008, Guruswami and Raghavendra, 2009, Feldman et al., 2012].

Given the beautiful theoretical work in this area, it is a little disappointing that these results are almost never used. The assumption of independent examples sampled from the uniform distribution is an idealization that would rarely, if ever, be applicable in practice. In real distributions, features are correlated. These correlations deem algorithms that assume independence useless. Thus the area of learning under the uniform distribution suffers from the following problem.

**Problem 1:** Algorithms for learning under the uniform distribution are not practical!

### 1.2 Markov Random Fields

Markov random fields are a standard way of representing high dimensional distributions (see e.g. [Kinderman and Snell, 1980]). Recall that a Markov random field, on a finite graph $G = (V, E)$ and taking values in a discrete set $A$, is a probability distribution on $A^V$ of the form $\Pr[(\sigma_v)_{v \in V}] = Z^{-1} \prod_C \phi_C((\sigma_v)_{v \in C})$, where the product is over all cliques $C$ in the graph, $\phi_C$ are some non-negative valued functions and $Z$ is the normalization constant. Here $(\sigma_v)_{v \in V}$ is an assignment from $V \rightarrow A$.

Markov Random Fields are widely used in vision, computational biology, biostatistics, spatial statistics and several other areas. The popularity of Markov Random Fields as modeling tools is coupled with extensive algorithmic theory studying sampling from these models, estimating their parameters and recovering them. However, to the best of our knowledge the following problem has not been studied.

**Problem 2:** Given an unknown function $f : A^V \rightarrow \{0, 1\}$ from a class $C$ and samples from the Markov Random Field, learn the function.

Of course the problem stated above is a special case of learning a function class given a general distribution [Kearns and Vazirani, 1994]. Therefore, a learning algorithm that can be applied for a general distribution can be also applied to MRF distributions. However in problem 2 we mean to ask the following:

Can we utilize the structure of the MRF to obtain better learning algorithms?
1.3 Our results

In this paper we begin to provide an answer to the two problems above. We show how methods that have been used in the theory of learning under the uniform distribution can be also applied for learning from MRF distributions.

This may sound surprising as the theory of learning under the uniform distribution strongly relies on the explicit Fourier representation of functions. Given an MRF distribution, one can also imagine expanding a function in terms of a “Fourier basis” for the MRF. Such a basis will consist of orthogonal functions under the MRF distribution. These functions are the eigenvectors of the transition matrix of the Gibbs MC associated with the MRF. It seems however that this approach is naive since:

(a) Each eigenfunction is of size $|A|^V$. We can’t even store it!

(b) How do we find these eigenfunctions?

(c) How do we find the expansion of a function in terms of these eigenfunctions?

MCMC Learning: The main effort in this paper is to provide an answer to the questions above. For this we use Gibbs sampling, which is a Markov chain Monte Carlo (MCMC) algorithm that is used to sample from an MRF. We will use this MCMC as the main engine in our learning algorithms. Note that

(a) The Gibbs sampler is a reversible Markov chain and therefore its eigenvectors are orthogonal with respect to the MRF distribution.

(b) The MCMC algorithm is easy to implement.

(c) Much is known about the convergence rate of this MCMC algorithm. Our results hold for Gibbs samplers that are rapidly mixing.

In Section 4, we show how the eigenvectors of the transition matrix of the Gibbs MC can be computed implicitly. We focus on the eigenvectors corresponding to the higher eigenvalues. These eigenvectors correspond to the stable part of the spectrum, i.e. the part that is not very sensitive to small errors. We adapt the power iteration method to find these eigenvectors. Observe that the powers of a transition matrix define distributions in time over the state space of the the Gibbs MC. Thus, for any $x$ the value of an eigenfunction at $x$ can be estimated by a careful use of the Gibbs sampler. The eigenfunctions thus obtained can be used as a basis for approximating functions. Note that the inner product between functions can also be estimated by sampling from the stationary distribution of the MRF.

The reason for focusing on the part of the spectrum corresponding to stable eigenvectors is twofold. First, it is technically easier to access this part of the spectrum. Furthermore, we think of eigenfunctions corresponding to small eigenvalues as unstable. Consider Gibbs sampling as a true temporal evolution of the system and let $\nu$ be an eigenfunction corresponding to a small eigenvalue. Then calculating $\nu(x)$ provides very little information on $\nu(y)$ where $y$ is obtained from $x$ after a short evolution of the.
Gibbs sampler. The reasoning just applied is a generalization of the classical reasoning for concentrating on the low frequency part of the Fourier expansion in traditional signal processing.

**Learning Juntas:** We also consider the so-called junta learning problem. A junta is a function that depends only on a small number of (typically many) variables. Learning juntas from i.i.d. examples is a notoriously difficult problem, see (Blum, 1992, Mossel et al., 2004). However, if the learning algorithm has access to labeled examples that are received from a Gibbs sampler, these correlated examples can be useful for learning juntas. We show that under standard technical conditions on the Gibbs MC, juntas can be learned in polynomial time by a very simple algorithm. These results are presented in Section 5.

### 1.4 Related Work

The idea of considering Markov Chains or Random Walks in the context of learning is not new. However, none of the results and models considered before give non-trivial improvements or algorithms in the context of MRFs. Work of [Aldous and Vazirani, 1995] studies a Markov chain based model where the main interest was in characterizing the number of new nodes visited. [Gamarnik, 1999] observed that after the mixing time a chain can simulate i.i.d. samples from the stationary distribution and thus obtained learning results for general Markov chains. [Bartlett et al., 1994] and [Bshouty et al., 2005] considered random walks on the discrete cube and showed how to utilize the random walk model to learn functions that cannot be easily learned from i.i.d. examples from the uniform distribution on the discrete cube. We further note that the problem considered here is unrelated to the currently studied problem of learning MRF structure.

### 2 Preliminaries

Let $X$ be an instance space. In this paper, we will assume that $X$ is finite and in particular we are mostly interested in the case when $X = \mathcal{A}^n$, where $\mathcal{A}$ is some finite set. For $x, x' \in \mathcal{A}^n$, let $d_H(x, x')$ denote the Hamming distance between $x$ and $x'$, i.e. $d_H(x, x') = |\{i \mid x_i \neq x'_i\}|$.

Let $M = (X, P)$ denote a time-reversible discrete time ergodic Markov chain with transition matrix $P$. When $X = \mathcal{A}^n$, we say that $M$ has single-site transitions if for any legal transition $x \rightarrow x'$ it is the case that $d_H(x, x') \leq 1$, i.e. $P(x, x') = 0$ when $d_H(x, x') > 1$. Let $X^0 = x_0$ denote the starting state of a Markov chain $M$. Let $P^t(x_0, \cdot)$ denote the distribution over states at time $t$, when starting from $x_0$. Let $\pi$ denote the stationary distribution of $M$. Denote by $\tau_M(x_0)$ the quantity:

$$\tau_M(x_0) = \min\{t : \|P^t(x_0, \cdot) - \pi\|_{TV} \leq \frac{1}{4}\}$$

\(^1\)For a good reference on algorithmic aspects of Markov chains, readers may refer to the book by [Aldous and Fill, 2002].
Then, define the mixing time of $M$ as $\tau_M = \max_{x_0 \in X} \tau_M(x_0)$. We say that a Markov chain with state space $X = \mathcal{A}^n$ is rapidly mixing if $\tau_M \leq \text{poly}(n)$.

While all the results in this paper are general, we describe two basic graphical models that will aid the discussion. We also use these models for elementary experiments to test the techniques introduced in this paper.

### 2.1 Ising Model

Consider a collection of sites, $[n] = \{1, \ldots, n\}$, and for each pair $i, j$, there is an associated interaction energy, $\beta_{ij}$. Suppose $([n], E)$ denotes the graph, where $\beta_{ij} = 0$ for $(i, j) \notin E$. A state $\sigma$ of the system consists of an assignment of spins, $\sigma_i \in \{+1, -1\}$, to the nodes $[n]$. The Hamiltonian of configuration $\sigma$ is defined as

$$H(\sigma) = - \sum_{(i, j) \in E} \beta_{ij} \sigma_i \sigma_j - B \sum_{i \in [n]} \sigma_i,$$

where $B$ is the external field. The energy of a configuration $\sigma$ is $\exp(-H(\sigma))$. The Glauber dynamics defines a Markov Chain $M = \langle \{-1, 1\}^n, P \rangle$, where the transitions are defined as follows:

(i) In state $\sigma$, pick a node $i \in [n]$ uniformly at random. With probability $1/2$ do nothing, otherwise

(ii) Let $\sigma'$ be obtained by flipping the spin at node $i$. Then, with probability $\exp(-H(\sigma'))/(\exp(-H(\sigma) + \exp(-H(\sigma'))))$, the state at the next time-step is $\sigma'$. Otherwise the state at the next time-step remains unchanged.

The stationary distribution of the above dynamics is the **Gibbs distribution**, where $\pi(\sigma) \propto \exp(-H(\sigma))$. It is known that there exists a $\beta(\Delta) > 0$ such that for all graphs of maximal degree $\Delta$, if $\max |\beta_{ij}| < \beta(\Delta)$ then the dynamics above is rapidly mixing (Dobrushin and Shlosman, 1985, Mossel and Sly, 2013).

### 2.2 Graph Coloring

Let $G = ([n], E)$ be a graph. For any $q > 0$, a valid $q$-coloring of the graph $G$ is a function $C : V \to [q]$ such that for every $(i, j) \in E$, $C(i) \neq C(j)$. For a node $i$, let $N(i) = \{j \mid (i, j) \in E\}$ denote the set of neighbors of $i$. Consider the Markov chain defined by the following transition:

(i) In state (coloring) $C$, choose a node $i \in [n]$ uniformly at random. With probability $1/2$ do nothing, otherwise:

(ii) Let $S \subseteq [q]$ be the subset of colors defined by $S = \{C(j) \mid j \in N(i)\}$. Define $C'$ to be the coloring obtained by choosing a random color $c \in [q] \setminus S$ and set $C'(i) = c$, $C'(j) = C(j)$ for $j \neq i$. The state at the next time-step is $C'$.

The stationary distribution of the above Markov chain is uniform over the valid colorings of the graph. It is known that the above chain is rapidly mixing when the condition $q \geq 3\Delta$ is satisfied, where $\Delta$ is the maximal degree of the graph (in fact better results are known (Jerrum 1995, Vigoda 1999)).
3 Learning Models

Let $X$ be a finite instance space and let $M = (X, P)$ be an irreducible discrete-time reversible Markov chain, where $P$ is the transition matrix. Let $\pi_M$ denote the stationary distribution of $M$, $\tau_M$ the mixing time. We assume that the Markov chain $M$ is rapidly mixing, i.e. $\tau_M \leq \text{poly}(\log(|X|))$ (note that if $X = A^n$, $\log(|X|) = O(n)$).

We consider the problem of learning with respect to stationary distributions of rapidly mixing Markov chains (e.g. defined by an MRF). The two graphical models described in the previous section serve as examples for such settings. The learning algorithm has access to an oracle, the one-step oracle, $OS(\cdot)$, that when queried with a state $x \in X$, returns the state after one step. Thus, $OS(x)$ is a random variable with distribution $P(x, \cdot)$.

Let $C$ be a concept class over $X$. The goal of the learning algorithm is to learn an unknown function, $f \in C$, with respect to the stationary distribution $\pi_M$ of the Markov chain $M$. As described above, the learning algorithm has query access to the transitions of the Markov chain. We consider two possible settings by which the learner may have access to the target function $f$.

3.1 Learning with i.i.d. examples only

In this setting, in addition to having access to the one-step oracle, $OS(\cdot)$, the learning algorithm has access to the standard PAC-example oracle, $EX(f, \pi)$ (Valiant, 1984). On each invocation, $EX(f, \pi)$ returns $(x, f(x))$ where $x$ is distributed according to the stationary distribution $\pi$ and $f$ is the unknown target function.

Our result in Section 4 holds in this model. In order to find the eigenvectors of the transition matrix, we only need unlabeled examples from the Gibbs sampler. Receiving i.i.d. labeled examples from the distribution is sufficient to estimate inner products of the target function with the eigenvectors found by our algorithm.

3.2 Learning with labeled examples from MC

In this setting, the learning algorithm has access to a random walk, $(x^1, f(x^1)), (x^2, f(x^2)), \ldots$, of the Markov chain. Here $x^{i+1}$ is the (random) state one time-step after $x^i$. Thus, the learning algorithm can potentially exploit correlations between consecutive examples. The result in Section 5 is only applicable in this stronger setting.

Since, we assume that the Markov chain is rapidly mixing, the learning algorithm essentially also has access to i.i.d. examples from the stationary distribution $\pi$. These can be obtained as follows: If $\tau_M$ is the mixing time, then after time $\tau_M \log(1/\delta)$, the resulting distribution is $\delta$-close to the stationary distribution, in terms of total variation distance.
4 Harmonic Analysis using Eigenvectors

In this section, we provide a procedure that implicitly computes eigenvectors of the transition matrix. We propose that these eigenvectors be used as basis functions by the learning algorithm for approximating the unknown target function.

Let $M = \langle X, P \rangle$ be a time-reversible discrete Markov chain. Let $\pi$ be the stationary distribution of $M$. We consider the set of right-eigenvectors of the matrix $P$. The largest eigenvalue of $P$ is 1 and the corresponding eigenvector has 1 in each co-ordinate. The left-eigenvector in this case is the stationary distribution. For simplicity of analysis we assume that $P(x, x) = 1/2$ for all $x$ which implies that all the eigenvalues of $P$ are non-negative. We are interested in identifying as many as possible of the remaining eigenvectors with eigenvalues less than 1. For readers unfamiliar with properties of Markov chains, a useful reference is the book by [Aldous and Fill (2002)].

For functions, $f, g : X \to \mathbb{R}$, define the inner-product, $\langle f, g \rangle_\pi = \mathbb{E}_{x \sim \pi}[f(x)g(x)]$, and the norm $\|f\|_\pi = \sqrt{\langle f, f \rangle_\pi}$. When the context is clear, we will drop the subscript $\pi$ from the inner products and norms.

Since $M$ is reversible, the right eigenvectors of $P$ are orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle_\pi$. Thus, these eigenvectors can be used as a basis to represent functions from $X \to \mathbb{R}$. First, we briefly show that this approach generalizes the standard Fourier analysis on the Boolean cube, which is commonly used in uniform-distribution learning.

4.1 Fourier Analysis over the Boolean Cube

Let $\{-1, 1\}^n$ denote the boolean cube. For $S \subseteq [n]$, the parity function over $S$ is defined as $\chi_S(x) = \prod_{i \in S} x_i$. With respect to the uniform distribution $U_n$ over $\{-1, 1\}^n$, the set of parity functions $\{\chi_S \mid S \subseteq [n]\}$ form an orthonormal “Fourier” basis, i.e. for $S \neq T$, $\mathbb{E}_{x \sim U_n}[\chi_S(x)\chi_T(x)] = 0$ and $\mathbb{E}_{x \sim U_n}[\chi_S(x)^2] = 1$.

We can view the uniform distribution over $\{-1, 1\}^n$ as arising from the stationary distribution of the following simple Markov chain. For $x, x'$, such that $x_i \neq x'_i$ and $x_j = x'_j$ for $j \neq i$, let $P(x, x') = 1/(2n)$; $P(x, x) = 1/2$. The remaining values of the matrix $P$ are set to 0. This chain is rapidly mixing with mixing time $O(n \log(n))$ and the stationary distribution is the uniform distribution over $\{-1, 1\}^n$. It is easy to see and well known that every parity function $\chi_S$ is an eigenvector of $P$ with eigenvalue $1 - |S|/n$. Thus, Fourier-based learning under the uniform distribution can be seen as a special case of Harmonic analysis using eigenvectors of the transition matrix.

4.2 Finding Eigenvectors Implicitly

As in the case of the uniform distribution over the boolean cube, we would like to find the eigenvectors of the transition matrix of a general Markov chain, $M$, and use these as an orthonormal basis for learning. Unfortunately, in most cases of interest the size of the set $|X|$ is likely to be prohibitively large, typically exponential in $n$, where $n$ is the length of the vectors in $X$. Thus, it is not possible to use standard techniques
**Inputs:** $x \in X$, $\tau$, access to function $g$, access to $\text{OS}(\cdot)$ oracle for Markov chain $M$

let $\langle (x_i) \rangle_{i=1}^m$ be drawn i.i.d. from stationary distribution $\pi$

let $t = 1$, $\gamma[1] = 0$, $\gamma[0] = 1$

while $|\gamma[t] - \gamma[t-1]| \geq \tau$

for $i = 1, \ldots, m$

$\hat{g}^t(x_i) = \mathbb{E}_{x \sim P^t(x, \cdot)}[g(x)]$

$\text{norm}[t] = \sqrt{\frac{1}{m} \sum_{i=1}^m \hat{g}^t(x_i)^2}$

$\gamma[t] = \frac{\text{norm}[t]}{\text{norm}[t-1]}$

**Output:** $\frac{\hat{g}^t(x)}{\text{norm}[t]}$

Figure 1: Algorithm to extract top eigenvector of $g$

to obtain eigenvectors of $P$. Here, we show how these eigenvectors may be computed implicitly.

An eigenvector of the transition matrix $P$ is a function $\nu : X \to \mathbb{R}$. Throughout this section, we will view any function $g : X \to \mathbb{R}$ as an $|X|$-dimensional vector with value $g(x)$ at position $x$. As such, even writing down such a vector corresponding to an eigenvector $\nu$ is not possible in polynomial time. Instead, the goal is to provide a procedure that given $x \in X$ would output (an approximation to) $\nu(x)$ in polynomial time. In order to find eigenvectors of $P$, we implicitly implement the power iteration method. Let $g : X \to \mathbb{R}$ be some function which can be computed efficiently. Suppose $g$ is expressed in terms of eigenvectors as,

$$g = \alpha_1 \nu_1 + \alpha_2 \nu_2 + \cdots + \alpha_k \nu_k$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$ are eigenvalues of $P$ corresponding to the eigenvectors $\nu_1, \ldots, \nu_k$ respectively and $\alpha_i = \langle g, \nu_i \rangle_\pi$.

Let $\mathbb{1}_x$ denote the $|X|$-dimensional vector that has 1 in the position corresponding to $x \in X$ and 0 elsewhere. Then denote by $P^t(x, \cdot)$ the distribution over states at time $t$ starting from state $x$ at time 0. By the assumption that $P(x, x) \geq 0$ it follows that all eigenvalues of $P$ are non negative. Observe that,

$$P^t g = \alpha_1 \lambda_1^t \nu_1 + \alpha_2 \lambda_2^t \nu_2 + \cdots + \alpha_k \lambda_k^t \nu_k \quad (1)$$

Depending on the gap $\lambda_1 - \lambda_2$, for a suitably chosen value of $t$, it will be the case that $P^t g \approx \alpha_1 \lambda_1^t \nu_1$. In Appendix A we characterize certain conditions on $g$ under which we can provably extract eigenvectors from $g$.

Now, it is easy to see how given access to a function $g : X \to \mathbb{R}$ and some $x \in X$, we can efficiently approximate $\nu_1(x)$, where $\nu_1$ is the top eigenvector of $g$. Rearranging
**Inputs:** $\mathcal{G} = \{g_1, \ldots, g_\ell\}$, $\mathbf{EX}(\pi, f)$ oracle

let $\langle (x^j, f(x^j)) \rangle_{j=1}^m$ be drawn from $\mathbf{EX}(\pi, f)$

let $\mathcal{V} = \{\nu_1, \ldots, \nu_k\}$ be extracted from $\mathcal{G}$ and suitably orthogonalized

for $i = 1, \ldots, k$
  \[ \alpha_i = \frac{1}{m} \sum_{j=1}^m \nu_i(x^j)f(x^j) \]

**Output:** $h(x) = \sum_{i=1}^k \alpha_i \nu_i(x)$

![Figure 2: Eigenvector-based Learning Algorithm](image)

terms of $[1]$, we have

$$\alpha_1 \lambda_1^2 \nu_1(x) = g(x) - \sum_{i=1}^k \alpha_i \lambda_i^2 \nu_i(x) \tag{2}$$

Let $\nu'_1$ denote the function $\alpha_1 \lambda_1^2 \nu_1$. Then, $\nu_1 = \nu'_1 / \|\nu'_1\|_\pi$. Thus, if we can (approximately) compute $\nu'_1(x)$ and estimate the norm $\|\nu'_1\|_\pi$, then we can approximate the value $\nu_1(x)$.

Recall that $\mathbb{1}_T P^t$ is simply the distribution $P^t(x, \cdot)$ over states at time $t$ when starting from state $x$ at time $0$. Thus, $\mathbb{1}_T P^t g = \mathbb{E}_{x' \sim P^t(x, \cdot)}[g(x')]$. With access to the one-step oracle, $\mathbf{OS}(\cdot)$, we can easily sample from the distribution $P^t(x, \cdot)$. Let $\hat{\mathbb{E}}^t[g]$ denote the estimate of $\mathbb{E}_{x' \sim P^t(x, \cdot)}[g(x')]$ obtained by sampling. We can obtain successive estimates each time increasing value of $t$ by one, until the ratio $\hat{\mathbb{E}}^{t+1}[g]/\hat{\mathbb{E}}^t[g]$ converges. Figure 1 describes an algorithm that given input $x \in X$, black-box access to $g$, and access to oracle $\mathbf{OS}(\cdot)$, outputs the approximate value of $\nu_1(x)$, where $\nu_1$ is the top eigenvector of $g$.

In order to extract more than one eigenvector from $g$, we consider $g' = g - \alpha_1 \nu_1$. Of course, we can only obtain approximate values for $\alpha_1$ and $\nu_1(x)$ and hence may introduce errors. Appendix $A$ gives conditions under which one can extract multiple eigenvectors from a single function $g$ without the estimation errors blowing up. For the actual learning algorithm, we will use several functions to extract eigenvectors. For two functions, $g_1, g_2$, suppose we extract eigenvectors $\nu_1, \nu_2$ with eigenvalues $\lambda_1, \lambda_2$ respectively. If the difference $|\lambda_1 - \lambda_2|$ is small, they may in fact be the same eigenvalue. We explicitly orthogonalize $\nu_1$ and $\nu_2$ to account for this possibility.

The learning algorithm we propose extracts as many eigenvectors as possible from a collection of functions, $\mathcal{G} = \{g_1, \ldots, g_\ell\}$. Suppose, $\mathcal{V} = \{\nu_1, \ldots, \nu_k\}$ is the resulting set of eigenvectors. Since, the eigenvectors $\mathcal{V}$ are orthonormal, for any function $f : X \to \mathbb{R}$, the best approximation using $\mathcal{V}$ is given by,

$$\tilde{f}(x) = \sum_{i=1}^k \langle f, \nu_i \rangle_\pi \nu_i(x).$$
A key point to note is that to estimate $\langle f, \nu \rangle_\pi$, we only need access to i.i.d. labeled examples from $\text{EX}(f, \pi_M)$. Figure 2 describes the learning algorithm which is an analogue of the low-degree algorithm of Linial et al. (1993) under the uniform distribution.

**Identifying Useful Functions to Extract Eigenvectors:** A key feature of our approach is the use of auxiliary functions from which we extract the eigenvectors. In general, domain knowledge may be helpful to choose such functions. One candidate class of functions, which we use in our experiments is the class of local functions. Suppose $X = A^n$, then for a subset $S \subseteq [n]$, let $b : S \rightarrow A$ be an assignment of values to the variables in $S$. Then, the local function, $g_{S,b} : X \rightarrow \mathbb{R}$ is defined as,

$$g_{S,b}(x) = \prod_{i \in S} (\mathbb{1}(x_i = b(i)) - \Pr(x_i = b(i)))$$

In the case of the boolean cube, these local functions are simply conjunctions of a fixed size, and the algorithm would exactly recover the low-degree parity functions as eigenvectors.

From the point of view of learning, the most useful part of the spectrum is the eigenvectors corresponding to large eigenvalues. These are usually more stable and hence most likely to correspond to the signal in the target function. The eigenvectors corresponding to lower eigenvalues are more unstable and their component in the labels may often be noise. The choice of auxiliary functions above, which depend on a small number of variables, may facilitate finding such stable eigenvectors.

We can summarize the discussion in this section as:

**Theorem 1.** Let $M = (A^n, P)$ be a Markov chain with transition matrix $P$. Let $\pi_M$ be the stationary distribution and $\tau_M$ the mixing time of $M$. Let $\mathcal{G}$ be a set of bounded functions from $A^n \rightarrow \mathbb{R}$ that satisfy the condition of Lemma 1 in Appendix. Then,

(i) There is an efficient algorithm (Fig. 3) that with access to the one-step oracle $\text{OS}(\cdot)$ and black-box access to $g$, extract from each $g \in \mathcal{G}$ a constant number of eigenvectors.

(ii) Let $\mathcal{V}$ be the collection eigenvectors obtained from step (i). Then for any bounded $f : A^n \rightarrow \mathbb{R}$ and $\epsilon > 0$, there is an efficient algorithm (Fig. 2) (with access to $\text{EX}(f, \pi_M)$ oracle) that outputs $\tilde{f} \in \text{span}(\mathcal{V})$ such that, $\mathbb{E}_{\pi_M}[(\tilde{f} - f)^2] \leq \min_{h \in \text{span}(\mathcal{V})} \mathbb{E}_{\pi_M}[(h - f)^2] + \epsilon$.

(iii) The running time of both algorithms is polynomial in $n, \tau_M, 1/\epsilon$.

**4.3 Experiments**

We report results from an elementary experimental study of our proposed techniques. We consider two models: (i) Ising model on a 4x4 2-D grid (ii) Graph coloring on 2x3 2-D grid with 4 colors. The function classes we used are (a) decision trees (b) linear separators (c) arbitrary functions of a small number of linear separators and (d) DNF expressions. For a data point $(x, y)$, there are two choices for features: (i) the component variables $x_1, \ldots, x_n$, or (ii) eigenfeatures $\nu_1(x), \ldots, \nu_k(x)$ for a collection
of eigenvectors. In both cases, we perform degree 1 and degree 2 regression. The results are shown as bar-charts in Figures 3-6. The four methods we use are the following: (i) Linear regression on variables (red), (ii) Degree two regression on the variables (gold) (iii) Regression using eigenvectors (blue) (iv) Degree 2 regression using eigenvectors (green). For each function class, we chose three random functions and used these algorithms for learning. We report both the root-mean square error (RMSE) for approximating the target, and the classification error (CE) of the hypothesis obtained by thresholding the best approximation. The results reported are as follows: (i) Figure 3 – RMSE (4x4 2-D Ising model) (ii) Figure 4 – CE (4x4 2-D Ising model) (iii) Figure 5 – RMSE (colourings of graph on 2x3 2-D grid) (iv) Figure 6 – CE (colourings of graph on 2x3 2-D grid).

Regression using eigenfeatures seems to outperform linear regression. However, degree 2 regression using eigenfeatures seems by far the best. In general for eigenfunctions $\nu_1, \nu_2$ of transition matrix $P$, the product function $\nu_1 \nu_2$ is not an eigenfunction. However, if these are localized in the sense that they essentially depend on a small number of variables, then such a product is most likely close to being an eigenfunction. The rapid mixing condition of the Gibbs sampler together with our choice of auxiliary functions make it likely that the eigenvectors we find are indeed localized. We considered products of all pairs of eigenfunctions extracted by our algorithm and a significant fraction of these products were very close to being eigenvectors. We expect that further theoretical and experimental work will elucidate this interesting direction.

Working with small graphical models allowed us high-precision computation. Otherwise, obtaining numerically stable eigenvectors would require better algorithms than the ones we are currently using. Even with the current algorithms, working with larger models is possible using clusters or GPUs. Most importantly the time required to obtain a single sample typically scales very benignly with the size of the graphical model. Also, the sampling can be easily parallelized since these are essentially independent runs of the Markov chain.

4.4 Discussion.

It would be interesting to study the performance of our techniques on larger models and a wider class of functions. A more theoretical question of interest is – can one show under further suitable assumptions that some classes of functions are well approximated by stable eigenvectors with respect to (certain classes of) MRF distributions?

In this work, we assume that the underlying MRF is known and the learner has access to the one-step oracle $\text{OS}(\cdot)$. There is a large body of work devoted to estimating parameters of MRFs from observed data. Our methods could be used in conjunction with these – first learn the parameters of the graphical model, then use that model to simulate the $\text{OS}(\cdot)$ oracle. It would be interesting to compare how such an approach would fare compared to methods such as PCA, ICA etc.

\footnote{In the case of regression using eigenfeatures, estimating the inner products with the target function suffices, since these are orthonormal. However, to minimize sampling errors one may wish to implement regularized regression in both cases.}
Figure 3: Bar chart showing RMSE errors for approximating four classes of functions. The groups correspond to three random functions chosen from each class. (a) Red: Linear (Degree 1) Regression on variables (b) Gold: Degree 2 Regression on variables (c) Blue: Regression (Degree 1) on eigenfeatures (d) Green: Degree 2 Regression on eigenfeatures. The graphical model is the Ising model on a 4x4 2-D grid.
Figure 4: Bar chart showing classification errors for learning four classes of functions. The groups correspond to three random functions chosen from each class. (a) Red: Linear (Degree 1) Regression on variables (b) Gold: Degree 2 Regression on variables (c) Blue: Regression (Degree 1) on eigenfeatures (d) Green: Degree 2 Regression on eigenfeatures. The graphical model is the Ising model on a 4x4 2-D grid.
Figure 5: Bar chart showing RMSE errors for approximating four classes of functions. The groups correspond to three random functions chosen from each class. (a) Red: Linear (Degree 1) Regression on variables (b) Gold: Degree 2 Regression on variables (c) Blue: Regression (Degree 1) on eigenfeatures (d) Green: Degree 2 Regression on eigenfeatures. The graphical model is 4-colourings of the 2x3 2-D grid.
Figure 6: Bar chart showing classification errors for learning four classes of functions. The groups correspond to three random functions chosen from each class. (a) Red: Linear (Degree 1) Regression on variables (b) Gold: Degree 2 Regression on variables (c) Blue: Regression (Degree 1) on eigenfeatures (d) Green: Degree 2 Regression on eigenfeatures. The graphical model is 4-colourings of the 2x3 2-D grid.
**Inputs:** Access to labeled examples \((x, f(x))\) from Markov Chain \(M\)

**Identifying Relevant Variables**

1. \(J = \emptyset\)
2. Consider a random walk, \(((x^1, f(x^1)), \ldots, (x^T, f(x^T)))\).
3. For every, \(i\), such that \(f(x^i) \neq f(x^{i+1})\), if \(j\) is the variable such that \(x^i_j \neq x^{i+1}_j\), add \(j\) to \(J\).

**Learning \(f\)**

1. Consider each of the \(|A|^{|J|}\) possible assignments \(b_J \to A\). We will construct a truth table for a function \(h : A^J \to \mathcal{Y}\).
2. For a fixed \(b_J\), let \(h(b_J)\) be the plurality label among the \(x^i\) in the random walk above for which \(x^i_j = b_J(j)\) for all \(j \in J\).

**Output:** Hypothesis \(h\)

---

**Figure 7: Algorithm: Exact Learning \(k\)-juntas**

### 5 Learning Juntas

In this section, we consider the problem of learning the class of \(k\)-juntas. Suppose \(X = A^n\) is the instance space. A \(k\)-junta is a boolean function that depends on only \(k\) out of the \(n\) possible co-ordinates of \(x \in X\). In this section, we consider the model in which we receive labeled examples from a random walk of a Markov chain (see Section 3.2). In this case the learning algorithm can identify the \(k\) relevant variables by keeping track of which variables caused the function to change its value.

For a subset, \(S \subseteq [n]\) of the variables and a function \(b_S : S \to A\), let \(x_S = b_S\) denote the event, \(\bigwedge_{i \in S} x_i = b_S(x_i)\), *i.e.* it fixes the assignment on the variables in \(S\) as given by the function \(b_S\). A set \(S\) is the junta of function \(f\), if the variables in \(S\) completely determine the value of \(f\). In this case, for \(b_S : S \to A\), every \(x\) satisfying \(x_S = b_S\) has the same value \(f(x)\) and by slight abuse of notation we denote this common value by \(f(b_S)\).

Figure 7 describes the simple algorithm for learning juntas. Theorem 2 gives conditions under which Algorithm 7 is guaranteed to succeed. Later, we show that the Ising model and graph coloring satisfy these conditions.

---

3 In the model where labeled examples are received from the only from stationary distribution, it seems unlikely that any learning algorithm can benefit from access to the \(\text{OS}(\cdot)\) oracle. The problem of learning juntas in time \(n^{o(k)}\) is a long-standing open problem even when the distribution is uniform over the Boolean cube, where the \(\text{OS}(\cdot)\) oracle can easily be simulated by the learner itself.
Theorem 2. Let $X = A^n$ and let $M = \langle X, P \rangle$ be a time-reversible rapidly mixing MC. Let $\pi$ denote the stationary distribution of $M$ and $\tau_M$ its mixing time. Furthermore, suppose that $M$ has single-site dynamics, i.e. $P(x, x') = 0$ if $d_H(x, x') > 1$ and that the following conditions hold:

(i) For any $S \subseteq [n]$, $b_S : S \rightarrow A$ either $\pi(x_S = b_S) = 0$ or $\pi(x_S = b_S) \geq 1/(c|A|)^{|S|}$, where $c$ is a constant.

(ii) For any $x, x'$ such that $\pi(x) \neq 0, \pi(x') \neq 0$ and $d_H(x, x') = 1$, $P(x, x') \geq \beta$.

Then Algorithm \ref{alg:junta} exactly learns the class of $k$-junta functions with probability at least $1 - \delta$ and the running time is polynomial in $n, |A|^k, \tau_M, 1/\beta, \log(1/\delta)$.

Proof. Let $f$ be the unknown target $k$-junta function. Let $S$ be the set of variables that influence $f$, $|S| \leq k$. The set $S$ is called the junta for $f$. Note that a variable $i$ is in the junta for $f$, if and only if there exist $x, x' \in A^n$ such that $\pi(x) \neq 0, \pi(x') \neq 0, x, x'$ differ only at co-ordinate $i$ and $f(x) \neq f(x')$. Otherwise, $i$ can have no influence in determining the value of $f$ (under the distribution $\pi$).

We claim that Algorithm \ref{alg:junta} identifies every variable in the junta $S$ of $f$. Let $b_S : S \rightarrow A$, be any assignment of values to variables in $S$. Since $S$ is the junta for $f$, any $x \in X$ that satisfies $x_i = b_S(i)$ for all $i \in S$, has the same value $f(x)$. By slight abuse of notation, we denote this common value by $f(b_S)$.

The fact that $i \in S$ implies that there exist assignments, $b_S^1, b_S^2$, such that $b_S^1(i) \neq b_S^2(i), \forall j \in S$, such that $j \neq i$, $b_S^1(j) = b_S^2(j)$ and which satisfy the following: $\pi(x_S = b_S^1) \neq 0, \pi(x_S, b_S^2) \neq 0$. Consider the following event: $x$ is drawn from $\pi$, $x'$ is the state after exactly one transition, $x$ satisfies the event $x_S = b_S^1$ and $x'$ satisfies the event $x'_S = b_S^2$. By our assumptions, the probability of this event is at least $\beta/(c|A|)^{|S|}$. Let $\alpha = \beta/(c|A|)^{|S|}$. Then, if we draw $x$ from the distribution $P^t(x_0, \cdot)$ for $t = \tau_M \ln(2/\alpha)$, instead of the true stationary distribution $\pi$, the probability of the above event is still at least $\alpha/2$. This is because when $t = \tau_M \ln(2/\alpha)$, the $\|P^t(x_0, \cdot) - \pi\|_{TV} \leq \alpha/2$.

Thus, by observing a long enough random walk, i.e. one with $2\tau_M \ln(1/\alpha) \log(k/\delta)/\alpha$ transitions, except with probability $\delta/k$, the variable $i$ will be identified as a member of the junta. Since there are at most $k$ such variables, by a union bound all of $S$ will be identified. Once the set $S$ has been identified, the unknown function can be learned exactly by observing an example of each possible assignments to the variables in $S$.

The above argument shows that all such assignments with non-zero measure under $\pi$ already exist in the observed random walk.

Remark 1. We observe that the condition that the MC be rapidly mixing alone is sufficient to identify at least one variable of the junta. However, unlike in the case of learning from i.i.d. examples, in this learning model, identifying one variable of the junta is not equivalent to learning the unknown junta function. In fact, it is quite easy to construct rapidly mixing Markov chains where the influence of some variables on the target function can be hidden, by making sure that the transitions that cause the function to change value happen only on a subset of the variables of the junta.

We now show that the Ising model and graph coloring satisfy the conditions of Theorem 2 as long as the underlying graphs have constant degree.
**Ising Model:** Recall that the state space is $X = \{-1, 1\}^n$. Let $\beta(\Delta)$ be the inverse critical temperature, which is a constant independent of $n$ as long as $\Delta$, the maximal degree, is constant. Let $S \subseteq [n]$ and let $b_S^1 : S \to \{-1, 1\}$ and $b_S^2 : S \to \{-1, 1\}$ be two distinct assignments to variables in $S$. Let $\sigma^1, \sigma^2$ be two configurations of the Ising system such that for all $i \in S$, $\sigma^1_i = b_S^1(i)$, $\sigma^2_i = b_S^2(i)$ and for $i \notin S$, $\sigma^1_i = \sigma^2_i$. Let $d^1 = \sum_{(i,j) \in E; \sigma^1_i \neq \sigma^1_j} \beta_{ij}$ and $d^2 = \sum_{(i,j) \in E; \sigma^2_i \neq \sigma^2_j} \beta_{ij}$. Then, since the maximum degree of the graph $\Delta$ is constant and each $\beta_{ij}$ is constant, $|d^1 - d^2| \leq 2\Delta|S|$. Then, by definition (see Section 2), $\exp(-c\beta\Delta|S|) \leq \pi(\sigma^1)/\pi(\sigma^2) \leq \exp(c\beta\Delta|S|)$. By summing over possible pairs $\sigma^1, \sigma^2$ that satisfy the constraints, we have $\exp(-\beta\Delta|S|) \leq \pi(x_S = b_S^1)/\pi(x_S = b_S^2) \leq \exp(\beta\Delta|S|)$. But, since there are only $2^{|S|}$ possible assignments of variables in $S$, the first assumption of Theorem 2 follows immediately. The second assumption follows from the definition of the transition rate matrix, i.e. each non-zero entry in the transition rate matrix is at least $\exp(-\beta\Delta)/2n$.

**Graph Coloring:** Let $q$ be the number of colors. The state space is $[q]^n$ and invalid colorings have 0 mass under the stationary distribution. We assume that $q \geq 3\Delta$, where $\Delta$ is the maximum degree in the graph. This is also the assumption that ensures rapid mixing. Let $S \subseteq [n]$ be an subset of nodes. Let $C_S^1$ and $C_S^2$ be two assignments of colors to the nodes in $S$. Let $D_1$ and $D_2$ be the set of valid colorings such that for each $x \in D_1$, $i \in S$, $x_i = C_S^1(i)$ and for each $x \in D_2$, $i \in S$, $x_i = C_S^2(i)$. We define a map from $D_1$ to $D_2$ as follows:

1. Starting from $x \in D_1$, first for all $i \in S$, set $x_i = C_S^2(i)$. This may in fact result in an invalid coloring.

2. The invalid coloring is switched to a valid coloring by only modifying neighbors of nodes in $S$. The condition that $q \geq 3\Delta$ ensures that this can always be done.

The above map has the following properties. Let $N(S) = \{j \mid (i,j) \in E, i \in S\}$. Then, the nodes that are not in $S \cup N(S)$ do not change the color. Thus, even though the map may be a many to one map, at most $q^{[S]+|N(S)|}$ elements in $D_1$ may be mapped to a single element in $D_2$. Note that $|S| + |N(S)| \leq (\Delta + 1)|S|$. Thus, we have $\pi(D_1)/\pi(D_2) = |D_1|/|D_2| \leq q^{(\Delta+1)|S|}$. This implies the first condition of Theorem 2. The second condition follows from the definition of the transition matrix, each non-zero entry is at least $1/(2qn)$.

**References**

David Aldous and James Fill. Reversible markov chains and random walks on graphs, 2002. Draft available at [www.stat.berkeley.edu/~aldous/RWG/book.html](http://www.stat.berkeley.edu/~aldous/RWG/book.html).

David Aldous and Umesh Vazirani. A markovian extension of valiant’s learning model. *Inf. Comput.*, 117(2):181–186, 1995.
Michael Alekhnovich, Mark Braverman, Vitaly Feldman, Adam R. Klivans, and Toniann Pitassi. The complexity of properly learning simple concept classes. *J. Comput. Syst. Sci.*, 74(1):16–34, 2008.

Peter L. Bartlett, Paul Fischer, and Klaus-Uwe Höffgen. Exploiting random walks for learning. In *Proceedings of the seventh annual conference on Computational learning theory*, COLT ’94, pages 318–327, 1994.

Avrim Blum. Learning boolean functions in an infinite attribute space. *Mach. Learn.*, 9(4):373–386, 1992.

Nader H. Bshouty, Elchanan Mossel, Ryan O’Donnel, and Rocco A. Servedio. Learning dnf from random walks. *J. Comput. Syst. Sci.*, 71(3):250–265, Oct 2005.

R. L. Dobrushin and S. B. Shlosman. Constructive criterion for uniqueness of a Gibbs field. In J. Fritz, A. Jaffe, and D. Szasz, editors, *Statistical Mechanics and dynamical systems*, volume 10, pages 347–370. 1985.

Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM J. Comput.*, 41(6):1558–1590, 2012.

David Gamarnik. Extension of the pac framework to finite and countable markov chains. In *Proceedings of the twelfth annual conference on Computational learning theory*, COLT ’99, pages 308–317, 1999.

Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. *SIAM J. Comput.*, 39(2):1558–1590, 2009.

Mark Jerrum. A very simple algorithm for estimating the number of k-colorings of a low-degree graph. *Random Structures and Algorithms*, 7(2):157–165, 1995.

Michael J. Kearns and Umesh Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, 1994.

Ross Kinderman and J. Laurie Snell. *Markov Random Fields and Their Applications*. AMS, 1980.

Adam R. Klivans and Alexander A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. *J. Comput. Syst. Sci.*, 75(1):2–12, 2009.

Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993.

Elchanan Mossel and Allan Sly. Exact thresholds for ising-gibbs samplers on general graphs. *The Annals of Probability*, 41(1):294–328, 2013.

Elchanan Mossel, Ryan O’Donnell, and Rocco A. Servedio. Learning functions of k relevant variables. *J. Comput. Syst. Sci.*, 69(3):421–434, 2004.
Ryan O’Donnell, Michael E. Saks, Oded Schramm, and Rocco A. Servedio. Every decision tree has an influential variable. In *FOCS*, pages 31–39, 2005.

Rocco A. Servedio, Li-Yang Tan, and Justin Thaler. Attribute-efficient learning and weight-degree tradeoffs for polynomial threshold functions. *Journal of Machine Learning Research - Proceedings Track*, 23:14.1–14.19, 2012.

Amir Shpilka and Avishay Tal. On the minimal fourier degree of symmetric boolean functions. In *IEEE Conference on Computational Complexity*, pages 200–209, 2011.

Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, Nov 1984.

E. Vigoda. Improved bounds for sampling coloring. In *40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 51–59, 1999.
A Finding Eigenvectors

Here, we formally state conditions under which multiple eigenvectors may be extracted from a fixed function $g$. Let $M = \langle X, P \rangle$ be a reversible rapidly mixing Markov chain. Let $\nu_1, \nu_2, \ldots$ denote the eigenvectors of $P$.

**Lemma 1.** Let $g$ be a function of the form

$$g = \alpha_1 \nu_1 + \alpha_2 \nu_2 + \cdots + \alpha_k \nu_k + \sum_{i > k} \alpha_i \nu_i + \eta$$

where $k$ is a constant and the following conditions hold,

1. $1 \geq \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k \geq \lambda_{k+1} \geq \cdots$.
2. For $1 \leq i \leq k$, $\alpha_i \geq 1/A$
3. $\sum_{i \geq 1} |\alpha_i| \leq B$
4. For all $x$, for $1 \leq i \leq k$, $|\nu_i(x)| \leq C$
5. For $1 \leq i \leq k$, $\frac{\ln(\lambda_i)}{\ln(\lambda_{i+1}) - \ln(\lambda_i)} \leq \zeta$.
6. For all $x$, $|\eta(x)| \leq \epsilon^{(\zeta+1)k} \frac{(2A(BC)^{\zeta/(\zeta+1)})^{\sum_{i=1}^{k}(\zeta+1)^i}}{(2A(BC)^{\zeta/(\zeta+1)})^{\sum_{i=1}^{k}(\zeta+1)^i}}$

Then, it is possible to extract $k$ eigenvectors from $g$ using the techniques described in Section 4.

**Proof.** Recall that, if $1_x$ is the vector that has 1 in position $x$ and 0 elsewhere, $1_x P^t = P^t(x, \cdot)$ is the distribution over states at time $t$ if the state was $x$ at time 0. Suppose $g$ is represented as a vector in $|X|$ dimensions, we have:

$$1_x P^t g = \alpha_1 \lambda_1^t \nu_1(x) + \sum_{i \geq 2} \alpha_i \lambda_i^t \nu_i(x) + 1_x P^t \eta$$

Multiplying both sides by $\alpha_1^{-1} \lambda_1^{-t}$ and rearranging terms, we get

$$\nu_1(x) = \alpha_1^{-1} \lambda_1^{-t} 1_x P^t g - \sum_{i \geq 2} \alpha_1^{-1} \alpha_i \lambda_i^{-t} \nu_i(x) - \alpha_1^{-1} \lambda_1^{-t} 1_x P^t \eta$$

Now, if $t = \ln \left( \frac{\lambda_1}{\lambda_2} \right) \ln \left( \frac{2ABC}{\tau} \right)$, the sum,

$$\sum_{i \geq 2} \alpha_1^{-1} \alpha_i \lambda_i^{-t} \lambda_1^t \nu_i(x) \leq \tau/2.$$  

Also, the quantity,

$$|\alpha_1^{-1} \lambda_1^{-t} 1_x P^t \eta| \leq A \left( \frac{2ABC}{\tau} \right)^{\zeta} \frac{\epsilon^{(\zeta+1)k}}{(2A(BC)^{\zeta/(\zeta+1)})^{\sum_{i=1}^{k}(\zeta+1)^i}}.$$
Now, setting \( \tau = \epsilon(\zeta+1)^{k-1} \)

\[
\frac{\epsilon(\zeta+1)^{k-1}}{(2A(BC)(\zeta/(\zeta+1)))\sum_{i=1}^{k-1}(\zeta+1)^i},
\]

we can see that if we use the function \( g_1 = g - \hat{\nu}_1 \), where \( |\hat{\nu}_1 - \nu_1|_{\infty} \leq \tau \), \( g_1 \) satisfies the conditions of the Lemma with \( k \) reduced to \( k - 1 \). Thus, by induction, we can obtain \( k - 1 \) eigenvectors from \( g_1 \), and hence \( k \) eigenvectors from \( g \).

We observe that although we’ve not explicitly accounted for errors that may arise due to sampling in order to estimate \( \mathbb{1}_xP^tg \) using MCMC, it can easily be argued using Chernoff bounds (\( g \) is bounded) that polynomially many samples suffice to ensure that the sampling error is smaller than \( \tau \) with high probability. \( \square \)