Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.438 Algorithms for Inference
Fall 2014

18 Markov Chain Monte Carlo Methods and Approximate MAP

In the recent lectures we have explored ways to perform approximate inference in graphical models with structures that make efficient inference intractable. First we saw loopy belief propagation and then found that it fit into a larger framework of variational approximation. Unfortunately, simple distributions may not always approximate our distribution well. In these cases, we may settle for characterizing the distribution with *samples*. Intuitively, if we have enough samples, we can recover any important information about the distribution.

Today, in the first half, we'll see a technique for sampling from a distribution without knowing the partition function. Amazingly, we can easily create a Markov chain whose stationary distribution is the distribution of interest. This area is very rich and we'll only briefly scratch the surface of it. In the second half, we'll switch gears and look at an algorithm for approximately finding the MAP through graph partitioning.

First, we'll see how sampling can capture essentially any aspect of interest of the distribution.

18.1 Why sampling?

Given $\{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ samples from $p_{\mathbf{x}}(\mathbf{x})$. Recall that the sample mean

$$\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)$$

is an unbiased estimator of $\mathbb{E}\left[f(\mathbf{x})\right]$ for any f irrespective of whether the samples are independent because of the linearity of expectation. If the samples are i.i.d. then by the law of large numbers, we have that the sample mean converges to the true expectation

$$\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i) \to \mathbb{E}\left[f(\mathbf{x})\right]$$

as $k \to \infty$. With different choices of f, we can capture essentially any aspect of interest of p. For example, choosing

$f(\mathbf{x}) = (\mathbf{x} - \mathbb{E}\left[\mathbf{x}\right])^2$ gives the variance

$f(\mathbf{x}) = -\log(p(x))$ gives differential entropy

$f(x) = \mathbb{1}_{x>x_*}$ gives $p(x > x_*)$ where x_* is a parameter

So if we have samples from the joint distribution $p(\mathbf{x})$, we can probe aspects of the distribution. However, our previous algorithms focused on the marginals $p(x_i)$. Can we use samples from the joint distribution $p(\mathbf{x})$ to tell us about the marginals? In fact, it is simple to see that if $\mathbf{x}^1, \ldots, \mathbf{x}^N$ are samples from the joint distribution, then $\mathbf{x}_i^1, \ldots, \mathbf{x}_i^N$ are samples from the marginal distribution $p(x_i)$. Hence, if we have samples from the joint distribution, we can project to the components to get samples from the marginals. Alternatively, if we're interested in just the marginals, we might come up with an algorithm to sample from the marginal distributions directly. One might wonder if there is any advantage to doing this. The marginal distributions tend to be much less complex than the joint distribution, so it may turn out to be much simpler to sample from the marginal distributions directly, rather than sampling from the joint distribution and projecting. For the moment, we'll focus on generating samples from $p(\mathbf{x})$ and in the next lecture, we'll return to sampling from the marginal distributions.

It's clear that the sampling framework is powerful, but naively drawing samples from $p(\mathbf{x})$ when we don't know the partition function is intractable. In the next section, we describe one approach to sampling from $p(\mathbf{x})$ called *Metropolis-Hastings.*

18.2 Markov Chain Monte Carlo

Suppose, we are interested in sampling from $p_{\mathsf{x}}(\mathbf{x})$, but we only know p_{x} up to a multiplicative constant (i.e. $p_{\mathsf{x}}(\mathbf{x}) = p_{\mathsf{x}}^*(\mathbf{x})/Z$ and we can calculate $p_{\mathsf{x}}^*(\mathbf{x})$). Initially, this seems like an immensely complicated problem because we do not know Z.

Our approach will be to construct a Markov chain $\mathbf{P}$ whose stationary distribution π is equal to p_{x} while only using p_{x}^* in our construction. Once we have created the Markov chain, we can start from an arbitrary $\mathbf{x}$, run the Markov chain until it converges to π and we will have a sample from p_{x}. Such an approach is called a *Markov Chain Monte Carlo* approach. To develop this, we will have to answer

1. How to construct such a Markov chain $\mathbf{P}$?

2. How long it takes for the Markov chain to converge to its stationary distribution?

We'll describe the Metropolis-Hastings algorithm to answer the first question. To answer the second, we'll look at the "mixing time" of Markov chains through *Cheeger's inequality.*

18.2.1 Metropolis-Hastings

First, we'll introduce some notation to make the exposition clearer. Let Ω be the state space of possible values of x and we'll assume x is discrete. For example, if x was a binary vector of length 10, then Ω would be $\{0,1\}^{10}$ and have 2^{10} elements. We will construct a Markov chain $\mathbf{P}$ which we will represent as a matrix $[\mathbf{P}_{ij}]$ where the (i,j) entry corresponds to the probability of transitioning from state $i \in \Omega$ to state

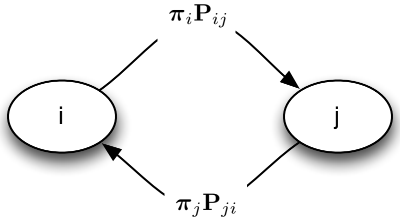

Figure 1: Detailed balance says the $\pi_i\mathbf{P}_{ij} = \pi_j\mathbf{P}_{ji}$, in other words the probability flowing from i to j is the same as the flow from j to i.

$j \in \Omega$. We want the stationary distribution of $\mathbf{P}$ denoted by a vector $\pi = [\pi_i]$ to be equal to p_x (i.e. $\pi_i = p_\mathsf{x}(i)$). Furthermore, we will require $\mathbf{P}$ to be a *reversible* Markov chain.

Definition 1 (Reversible Markov chain)**.** *A Markov Chain* $\mathbf{P}$ *is called reversible with respect to* π *if it satisfies*

$$\pi_i\mathbf{P}_{ij} = \pi_j\mathbf{P}_{ji}.$$

This equation is also referred to as detailed balance.

Intuitively, detailed balance says that the probability "flowing" from i to j is the same amount of probability "flowing" from j to i, where by probability "flow" from i to j we mean $\pi_i\mathbf{P}_{ij}$ (i.e. the probability of being in i and transitioning to j).

Importantly, if $\mathbf{P}$ is reversible with respect to π and we did not assume π was the stationary distribution, detailed balance implies that π is a stationary distribution because

$$\sum_j \pi_j\mathbf{P}_{ji} = \sum_j \pi_i\mathbf{P}_{ij} = \pi_i\left(\sum_j \mathbf{P}_{ij}\right) = \pi_i.$$

So showing that $\mathbf{P}$ satisfies the detailed balance equation with π is one way of showing the π is a stationary distribution of $\mathbf{P}$.

To obtain such a $\mathbf{P}$, we will start with a "proposed" Markov chain $\mathbf{K}$ which we will modify to create $\mathbf{P}$ and as we'll see the conditions that $\mathbf{K}$ must satisfy are very mild and $\mathbf{K}$ may have little or no relation to p_x. Again, we will represent $\mathbf{K}$ as a matrix $[\mathbf{K}_{ij}]$ and we require that

$\mathbf{K}_{ii} > 0$ for all $i \in \Omega$ and

$\mathcal{G}(\mathbf{K}) = (\Omega, \mathcal{E}(\mathbf{K}))$ is connected where $\mathcal{E}(\mathbf{K}) \triangleq \{(i,j) : \mathbf{K}_{ij}\mathbf{K}_{ji} > 0\}$.

In other words, all self-transitions must be possible and it must be possible to move from any state to another state in some number of transitions of the Markov chain.

Then, define

$$R(i,j) \triangleq \min\left(1, \frac{p^*_{\mathbf{x}}(j)\mathbf{K}_{ji}}{p^*_{\mathbf{x}}(i)\mathbf{K}_{ij}}\right) =^{1} \min\left(1, \frac{p_{\mathbf{x}}(j)\mathbf{K}_{ji}}{p_{\mathbf{x}}(i)\mathbf{K}_{ij}}\right)^{2}$$

and

$$\mathbf{P}_{ij} \triangleq \begin{cases} \mathbf{K}_{ij}R(i,j) & j \neq i \\ 1 - \sum_{j\neq i}\mathbf{P}_{ij} & j = i \end{cases}.$$

This is the **Metropolis-Hastings** Markov chain $\mathbf{P}$ that we were after. Now it remains to show that $p_{\mathbf{x}}$ is the stationary distribution of $\mathbf{P}$, and as we commented above it suffices to show that $\mathbf{P}$ satisfies the detailed balance equation with $p_{\mathbf{x}}$.

Lemma 1. *For all $i, j \in \Omega$, detailed balance $p_{\mathbf{x}}(i)\mathbf{P}_{ij} = p_{\mathbf{x}}(j)\mathbf{P}_{ji}$ holds.*

Proof. For $(i,j) \notin \mathcal{E}(\mathbf{K})$ this is trivially true. For $(i,j) \in \mathcal{E}(\mathbf{K})$, without loss of generality let $p^*_{\mathbf{x}}(j)\mathbf{K}_{ji} \geq p^*_{\mathbf{x}}(i)\mathbf{K}_{ij}$. This implies that $R(i,j) = 1$ and $R(j,i) = \frac{p^*_{\mathbf{x}}(i)\mathbf{K}_{ij}}{p^*_{\mathbf{x}}(j)\mathbf{K}_{ji}}$. Then

$$\begin{aligned} p_{\mathbf{x}}(i)\mathbf{P}_{ij} &= p_{\mathbf{x}}(i)\mathbf{K}_{ij} = p_{\mathbf{x}}(i)\mathbf{K}_{ij}\frac{p_{\mathbf{x}}(j)\mathbf{K}_{ji}}{p_{\mathbf{x}}(j)\mathbf{K}_{ji}} = \left(\frac{p_{\mathbf{x}}(i)\mathbf{K}_{ij}}{p_{\mathbf{x}}(j)\mathbf{K}_{ji}}\right)\mathbf{K}_{ji}p_{\mathbf{x}}(j) \\ &= R(j,i)\mathbf{K}_{ji}p_{\mathbf{x}}(j) = \mathbf{P}_{ji}p_{\mathbf{x}}(j). \end{aligned}$$

□

Thus, we conclude that $p_{\mathbf{x}}$ is the stationary distribution of $\mathbf{P}$ as desired.

Because the matrices describing $\mathbf{K}$ and $\mathbf{P}$ are enormously large[3], it can be helpful to think of $\mathbf{K}$ and $\mathbf{P}$ as describing a process that explains how to generate a new state j in the Markov chain given our current state i. From this perspective, the process describing $\mathbf{P}$ is as follows, starting from state i, to generate state j:

```
Generate $j'$ according to $\mathbf{K}$ with current state $i$.
Flip a coin with bias $R(i,j')$
    If heads, then the new state $j = j'$.
    If tails, then the new state $j = i$, the old state.
```

This gives a convenient description of $\mathbf{P}$, which can easily be implemented in code. Also, $R(i,j)$ is commonly referred to as the *acceptance probability* because it describes the probability of accepting the proposed new state j'.

[1] The equality holds because Z cancels out.

[2] The astute reader will notice that the ratio in $R(i,j)$ is directly related to the detailed balance equation.

[3] They're $|\Omega| \times |\Omega|$ and the reason we cannot calculate $p_{\mathbf{x}}$ is that Ω is so large.

Intuitively, we can think of Metropolis-Hastings as forcing $\mathbf{K}$ to be reversible in a specific way. Given an arbitrary $\mathbf{K}$, there's no reason that $p_{\mathsf{x}}(i)\mathbf{K}_{ij}$ will be equal to $p_{\mathsf{x}}(j)\mathbf{K}_{ji}$, in other words, the probability flow from i to j will not necessarily be equal to the flow from j to i. To fix this, we could scale the flow on one side to be equal to the other and that's what Metropolis-Hastings does. To make this notion more precise, let $R(p_{\mathsf{x}})$ be the space of all reversible Markov chains that have p_{x} as a stationary distribution. Then the Metropolis-Hastings algorithm takes $\mathbf{K}$ and gives us $\mathbf{P} \in R(p_{\mathsf{x}})$ that satisfies

Theorem 1.

$$\mathbf{P} = \arg \min_{\mathbf{Q} \in R(p_{\mathsf{x}})} d(\mathbf{K}, \mathbf{Q}) \triangleq \arg \min_{\mathbf{Q} \in \mathbf{R}(p_{\mathsf{x}})} \sum_i p_{\mathsf{x}}(i) \sum_{j \neq i} |\mathbf{K}_{ij} - \mathbf{Q}_{ij}|.$$

Hence, $\mathbf{P}$ *is the* l_1*-projection of* $\mathbf{K}$ *on* $R(p_{\mathsf{x}})$*.*

18.2.2 Example: MRF

We'll describe a simple example of using Metropolis-Hastings to sample from an MRF. Suppose we have $\mathsf{x}_1, \ldots, \mathsf{x}_n$ binary with

$$p(\mathbf{x}) \propto \exp \underbrace{\left(\sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(x_i, x_j) \right)}_{U(\mathbf{x})},$$

for some graph $\mathcal{G}$. Here $\Omega = \{0,1\}^n$, so it has 2^n elements. Suppose we have $\mathbf{K} = [\frac{1}{2^n}]$ the matrix with all entries equal to $\frac{1}{2^n}$ that is the probability of transitioning from i to j is equally probable for all j. Then the Metropolis-Hastings algorithm would give

$$\begin{aligned} \mathbf{P}_{ij} &= \mathbf{K}_{ij} \min \left(1, \frac{\exp(U(i))}{\exp(U(j))} \right) \\ &= \frac{1}{2^n} \min \left(1, \exp(U(i) - U(j)) \right). \end{aligned}$$

Is there any downside to choosing such a simple $\mathbf{K}$? If i has moderate probability, the chance of randomly choosing a j that has higher probability is very low, so we're very unlikely to transition away from i. Thus it make take a long time for the Markov chain to reach its stationary distribution.

18.2.3 Example: Gibbs Sampling

Gibbs sampling is an example of Metropolis-Hastings, where $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{K}$ is defined by the following process for going from $\mathbf{x} \to \mathbf{x}'$

```
Select $k \in \{1, \dots, n\}$ from a uniform distribution.

Set $\mathbf{x}'_{-k} = \mathbf{x}_{-k}$ and sample $x'_k$ from $p(x'_k|\mathbf{x}_{-k})$.
```

where $\mathbf{x}_{-k} \triangleq x_1, \dots, x_{k-1}, x_{k+1}, \dots, x_n$. The practical applicability of Gibbs sampling depends on the ease with which samples can be drawn from the conditional distributions $p(x_k|\mathbf{x}_{-k})$. In the case of undirected graphical models, the conditional distributions for individual nodes depends only on the neighboring nodes, so in many cases, it is simple to sample from the conditional distributions.

If $p > 0$ then it follows that the graph for $\mathbf{K}$ is connected and all self-transitions are possible. Below, we'll see that $\mathbf{K}$ satisfies detailed balance, so the acceptance probability will always be 1, hence the Metropolis-Hastings transition matrix $\mathbf{P}$ will be equal to $\mathbf{K}$.

Lemma 2. *K satisfies detailed balance with respect to $p_{\mathbf{x}}$.*

Proof. For $\mathbf{x}$ and $\mathbf{x}'$, we must show that $p(\mathbf{x})\mathbf{K}_{\mathbf{x}\mathbf{x}'} = p(\mathbf{x}')\mathbf{K}_{\mathbf{x}'\mathbf{x}}$. If $\mathbf{x} = \mathbf{x}'$, then the equation is satisfied trivially. Suppose $\mathbf{x} \neq \mathbf{x}'$ and suppose they differ in at least two positions. By construction $\mathbf{K}_{\mathbf{x}\mathbf{x}'} = \mathbf{K}_{\mathbf{x}'\mathbf{x}} = 0$, so the equation is satisfied trivially. Lastly, suppose $\mathbf{x} \neq \mathbf{x}'$ and they differ in exactly one position k. Then,

$$\begin{aligned} p(\mathbf{x})\mathbf{K}_{\mathbf{x}\mathbf{x}'} &= \frac{1}{n} p(\mathbf{x}) p(x'_k|\mathbf{x}_{-k}) \\ &= \frac{1}{n} p(x_k|\mathbf{x}'_{-k}) p(\mathbf{x}'_{-k}) p(x'_k|\mathbf{x}'_{-k}) \\ &= \frac{1}{n} p(x_k|\mathbf{x}'_{-k}) p(\mathbf{x}') \\ &= p(\mathbf{x}')\mathbf{K}_{\mathbf{x}'\mathbf{x}}. \end{aligned}$$

using the fact that $\mathbf{x}_{-k} = \mathbf{x}'_{-k}$. □

In practice, Gibbs sampling works well and in many cases it is simple to implement. Explicitly, we start from an initial state $\mathbf{x}^0$ and generate putative samples $\mathbf{x}^1, \dots, \mathbf{x}^T$ according to the following process:

```
for $t = 0, \dots, T-1$:
    Select $i \in \{1, \dots, n\}$ uniformly.
    Set $\mathbf{x}^{t+1}_{-i} = \mathbf{x}^t_{-i}$ and sample $x^{t+1}_i$ from $p(x^{t+1}_i|\mathbf{x}^t_{-i})$.
```

However, all of the caveats about using the samples generated by Metropolis-Hastings apply. For example, we need to run the Markov chain until it has reached its stationary distribution, so we need to toss out a number of initial samples in a process called "burn-in". In the next section, we'll see theoretical results on the time it takes the Markov chain to reach its stationary distribution, but in practice people rely on heuristics. More advanced forms of Gibbs sampling exist, such as *block* Gibbs sampling as seen on Problem Set 8, but their full development is beyond the scope of this class.

18.3 Mixing Time

Now that we've constructed the Markov chain, we can sample from it, but we want samples from the stationary distribution. We turn to the second question: how long does it take for the Markov chain to converge to its stationary distribution?

For simplicity of exposition, we'll focus on a generic reversible Markov chain $\mathbf{P}$ with state space Ω and unique stationary distribution $\boldsymbol{\pi}$. We'll also assume that $\mathbf{P}$ *regular*, that is $\mathbf{P}^k > 0$ for some $k > 0$. Intuitively, that means that for some $k > 0$, it is possible to transition from any $i \in \Omega$ to any $j \in \Omega$ in exactly k steps. Additionally, we'll assume that $\mathbf{P}$ is a *lazy* Markov chain, which means that $\mathbf{P}_{ii} > 0$ for all $i \in \Omega$. This is a mild condition because we can take any Markov chain $\mathbf{Q}$ and turn it into a lazy Markov chain without changing its stationary distribution by considering $\frac{1}{2}(\mathbf{Q}+\mathbf{I})$. The lazy condition ensures that all of the eigenvalues of $\mathbf{P}$ are positive and it does not substantially increase the mixing time.

We're interested in measuring the time it takes $\mathbf{P}$ to go from any initial state to its stationary distribution. Precisely, we'll define

Definition 2 (ϵ-mixing time of $\mathbf{P}$)**.** *Given $\epsilon > 0$, $T_{mix}(\epsilon)$ is the smallest time such that for $t \geq T_{mix}(\epsilon)$*

$$|\boldsymbol{\mu}\mathbf{P}^t - \boldsymbol{\pi}|_{TV} \leq \epsilon,$$

for any initial distribution $\boldsymbol{\mu}$ where $|\boldsymbol{\mu}\mathbf{P}^t - \boldsymbol{\pi}|_{TV} = \sum_i |(\mu\mathbf{P}^t)_i - \boldsymbol{\pi}_i|$ is the total variation.

To get a bound on $T_{mix}(\epsilon)$, we will focus our attention how what the operation of multiplying with $\mathbf{P}$ does. Recall that as we apply $\mathbf{P}$ to any vector, the eigenvector with the largest eigenvalue dominates and how quickly it dominates is determined by the second largest eigenvalue. We will exploit this intuition to get a bound on $T_{mix}(\epsilon)$ that depends on the difference between the largest and second largest eigenvalues.

The following is technical and is included for completeness. First, we'll the total variation by a term that does not depend on the initial distribution $\boldsymbol{\mu}$.

$$\begin{aligned}
\sum_i |(\boldsymbol{\mu}\mathbf{P}^t)_i - \boldsymbol{\pi}_i| &= \sum_i |\sum_j \boldsymbol{\mu}_j(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| \\
&= \sum_i |\sum_j \boldsymbol{\mu}_j((\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i)| \\
&\leq \sum_{ij} \boldsymbol{\mu}_j |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| \\
&= \sum_j \boldsymbol{\mu}_j \sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| \\
&\leq ||\boldsymbol{\mu}||_1 \max_j \sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| \\
&= \max_j \sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i|.
\end{aligned}$$

where used Holder's inequality. Now we show how to bound $\sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i|$ for every $j \in \Omega$.

$$\begin{aligned}\sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| &= \sum_i |\frac{(\mathbf{P}^t)_{ij}}{\boldsymbol{\pi}_i} - 1|\sqrt{\boldsymbol{\pi}_i}\sqrt{\boldsymbol{\pi}_i} \\ &\leq \left[\left(\sum_i \left|\frac{(\mathbf{P}^t)_{ij}}{\boldsymbol{\pi}_i} - 1\right|^2 \boldsymbol{\pi}_i\right)\left(\sum_i \boldsymbol{\pi}_i\right)\right]^{\frac{1}{2}},\end{aligned}$$

where we used Cauchy-Schwarz to get the inequality. After some algebraic manipulation

$$\begin{aligned}\left[\left(\sum_i \left|\frac{(\mathbf{P}^t)_{ij}}{\boldsymbol{\pi}_i} - 1\right|^2 \boldsymbol{\pi}_i\right)\left(\sum_i \boldsymbol{\pi}_i\right)\right]^{\frac{1}{2}} &= \left(\sum_i \left(1 - \frac{2(\mathbf{P}^t)_{ji}}{\boldsymbol{\pi}_i} + \frac{(\mathbf{P}^t)_{ji}^2}{\boldsymbol{\pi}_i^2}\right)\boldsymbol{\pi}_i\right)^{\frac{1}{2}} \\ &= \left(\sum_i \frac{(\mathbf{P}^t)_{ji}^2}{\boldsymbol{\pi}_i} - 1\right)^{\frac{1}{2}}.\end{aligned}$$

Now we'll use the reversibility of $\mathbf{P}$,

$$\begin{aligned}\left(\sum_i \frac{(\mathbf{P}^t)_{ji}^2}{\boldsymbol{\pi}_i} - 1\right)^{\frac{1}{2}} &= \left(\sum_i \frac{(\mathbf{P}^t)_{ji}(\mathbf{P}^t)_{ji}\boldsymbol{\pi}_j}{\boldsymbol{\pi}_i\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}} \\ &= \left(\sum_i \frac{(\mathbf{P}^t)_{ji}(\mathbf{P}^t)_{ij}\boldsymbol{\pi}_i}{\boldsymbol{\pi}_i\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}} \\ &= \left(\sum_i \frac{(\mathbf{P}^t)_{ji}(\mathbf{P}^t)_{ij}}{\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}} \\ &= \left(\frac{(\mathbf{P}^{2t})_{jj}}{\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}},\end{aligned}$$

where we used the reversibility of $\mathbf{P}$ to exchange $\boldsymbol{\pi}_j(\mathbf{P}^t)_{ji}$ for $\boldsymbol{\pi}_i(\mathbf{P}^t)_{ij}$. Putting this together, we conclude that

$$\sum_i |(\mathbf{P}^t)_{ji} - \boldsymbol{\pi}_i| \leq \left(\frac{(\mathbf{P}^{2t})_{jj}}{\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}}.$$

So we need to find a bound on the diagonal entries of $\mathbf{P}^{2t}$. Consider the following matrix

$$\mathbf{M} \triangleq \operatorname{diag}(\sqrt{\boldsymbol{\pi}})\mathbf{P}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})$$

where $\operatorname{diag}(\sqrt{\boldsymbol{\pi}})$ is the diagonal matrix with $\sqrt{\boldsymbol{\pi}}$ along the diagonal. Note that $\mathbf{M}$ is symmetric because

$$\begin{aligned}\mathbf{M}_{ij} &= \sqrt{\frac{\boldsymbol{\pi}_i}{\boldsymbol{\pi}_j}}\mathbf{P}_{ij} = \frac{\boldsymbol{\pi}_i\mathbf{P}_{ij}}{\sqrt{\boldsymbol{\pi}_i\boldsymbol{\pi}_j}} = \frac{\boldsymbol{\pi}_j\mathbf{P}_{ji}}{\sqrt{\boldsymbol{\pi}_i\boldsymbol{\pi}_j}} \\ &= \sqrt{\frac{\boldsymbol{\pi}_j}{\boldsymbol{\pi}_i}}\mathbf{P}_{ji} = \mathbf{M}_{ji},\end{aligned}$$

using the reversibility of $\mathbf{P}$. Recall the *spectral theorem* from linear algebra which says that for any real symmetric matrix $\mathbf{A}$, there exist orthonormal eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n$ such that

$$\mathbf{A} = \sum_{k=1}^{n} \lambda_k \mathbf{v}_k \mathbf{v}_k^{\mathrm{T}} = \mathbf{V}\operatorname{diag}(\lambda)\mathbf{V}^{\mathrm{T}},$$

where $\mathbf{V}$ is the matrix with $\mathbf{v}_1, \ldots, \mathbf{v}_n$ as columns and $\lambda = (\lambda_1, \ldots, \lambda_n)$.

Decomposing $\mathbf{M}$ in this way, shows that

$$\mathbf{M}^t = (\mathbf{V}\operatorname{diag}(\lambda)\mathbf{V}^{\mathrm{T}})^t = \mathbf{V}\operatorname{diag}(\lambda)^t\mathbf{V}^{\mathrm{T}}$$

because $\mathbf{V}$ is orthogonal. Hence

$$\begin{aligned}\mathbf{P}^t &= \operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})\mathbf{M}^t\operatorname{diag}(\sqrt{\boldsymbol{\pi}}) \\ &= \operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})\mathbf{V}\operatorname{diag}(\lambda)^t\mathbf{V}^{\mathrm{T}}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}) \\ &= \operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})\mathbf{V}\operatorname{diag}(\lambda)^t\mathbf{V}^{\mathrm{T}}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}).\end{aligned}$$

From this representation of $\mathbf{P}^t$, we conclude that

$$(\mathbf{P}^t)_{jj} = \sum_i \lambda_i^t(\mathbf{v}_i)_j^2.$$

It can be shown by the Perron-Frobenius theorem, that $\lambda_1 = 1$ and $|\lambda_k| < 1$ for $k < 1$ and the first eigenvector of $\mathbf{P}$ is $\boldsymbol{\pi}$. $\mathbf{M}$ is similar[4] to $\mathbf{P}$ hence they have the same eigenvalues and their eigenspaces have the same dimensions. By construction, an eigenvector $\mathbf{u}$ of $\mathbf{P}$ implies that $\mathbf{u}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})$ is an eigenvector of $\mathbf{M}$ with the same eigenvalue. Thus $\boldsymbol{\pi}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})$ is eigenvector of $\mathbf{M}$ and because the eigenspace corresponding to the eigenvalue 1 has dimension 1, $\mathbf{v}_1 \propto \boldsymbol{\pi}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})$. Furthermore, $||\mathbf{v}_1||_2 = 1$ because $\mathbf{V}$ is orthogonal, so we conclude that $\mathbf{v}_1 = \pm\boldsymbol{\pi}\operatorname{diag}(\sqrt{\boldsymbol{\pi}}^{-1})$. Using

[4] Matrices $\mathbf{A}$ and $\mathbf{B}$ are similar if $\mathbf{A} = \mathbf{C}^{-1}\mathbf{B}\mathbf{C}$ for some invertible matrix $\mathbf{C}$.

this, we can simplify the expression for $(\mathbf{P}^{2t})_{jj}$ and upper bound it

$$\begin{aligned}(\mathbf{P}^{2t})_{jj} &= \boldsymbol{\pi}_j + \sum_{i=2}^{n} \lambda_i^{2t} (\mathbf{v}_i)_j^2 \\ &\le \boldsymbol{\pi}_j + \lambda_2^{2t} \sum_{i=2}^{n} (\mathbf{v}_i)_j^2 \\ &\le \boldsymbol{\pi}_j + \lambda_2^{2t} \sum_{i=1}^{n} (\mathbf{v}_i)_j^2 \\ &= \boldsymbol{\pi}_j + \lambda_2^{2t},\end{aligned}$$

because $\mathbf{V}$ is orthogonal. Putting this into our earlier expression yields

$$\begin{aligned}\left(\frac{(\mathbf{P}^{2t})_{jj}}{\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}} &\le \left(\frac{\boldsymbol{\pi}_j + \lambda_2^{2t}}{\boldsymbol{\pi}_j} - 1\right)^{\frac{1}{2}} \\ &= \left(\frac{\lambda_2^{2t}}{\boldsymbol{\pi}_j}\right)^{\frac{1}{2}}.\end{aligned}$$

Putting all of the bounds together we have

$$|\boldsymbol{\mu}\mathbf{P}^t - \boldsymbol{\pi}|_{TV} \le \max_j \left(\frac{\lambda_2^{2t}}{\boldsymbol{\pi}_j}\right)^{\frac{1}{2}} = \lambda_2^t \left(\frac{1}{\min_j \boldsymbol{\pi}_j}\right)^{\frac{1}{2}}.$$

Setting the right hand side equal to ϵ and solving for t gives

$$t = \frac{\log \epsilon + \frac{1}{2}\log(\min_j \boldsymbol{\pi}_j)}{\log \lambda_2}.$$

This gives an upper bound on the time it takes the Markov chain to mix so that $|\boldsymbol{\mu}\mathbf{P}^t - \boldsymbol{\pi}|_{TV} < \epsilon$. Because we used inequalities to arrive at t, we can only conclude that

$$\begin{aligned}T_{mix}(\epsilon) \le t = \frac{\log \epsilon + \frac{1}{2}\log(\min_j \boldsymbol{\pi}_j)}{\log \lambda_2} &= \frac{\log \frac{1}{\epsilon} + \frac{1}{2}\log \frac{1}{\min_j \boldsymbol{\pi}_j}}{\log 1/\lambda_2} \\ &\le \frac{\log \frac{1}{\epsilon} + \frac{1}{2}\log \frac{1}{\min_j \boldsymbol{\pi}_j}}{1 - \lambda_2}.\end{aligned}$$

So, as expected, the mixing time depends on the difference between the largest and second largest eigenvalues. In this case, Cheeger's celebrated inequality states that

$$\frac{1}{1 - \lambda_2} \le \frac{2}{\Phi^2},$$

where the *conductance* Φ of $\mathbf{P}$ is defined as

$$\Phi = \Phi(\mathbf{P}) = \min_{\mathcal{S} \subset \Omega} \frac{\sum_{i \in \mathcal{S}, j \in \mathcal{S}^c} \boldsymbol{\pi}_i \mathbf{P}_{ij}}{\boldsymbol{\pi}(\mathcal{S}) \boldsymbol{\pi}(\mathcal{S}^c)}$$

where $\boldsymbol{\pi}(\mathcal{S}) = \sum_{k \in \mathcal{S}} \boldsymbol{\pi}_k$ and similarly for $\boldsymbol{\pi}(\mathcal{S}^c)$. Conductance takes the minimum over $\mathcal{S}$ of the probability of starting in $\mathcal{S}$ and transitioning to $\mathcal{S}^c$ in one time step normalized by the "sizes" of $\mathcal{S}$ and $\mathcal{S}^c$. If the conductance is small, then there is a set $\mathcal{S}$ such that transitioning out of $\mathcal{S}$ is difficult, so if the Markov chain gets stuck in $\mathcal{S}$, it will be unlikely to leave $\mathcal{S}$, hence we would expect the mixing time to be large.

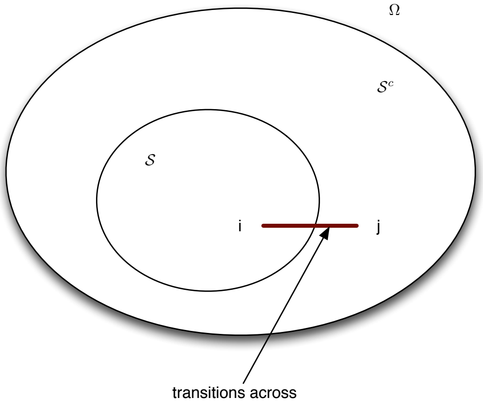

Figure 2

Thus we conclude that

$$T_{mix}(\epsilon) \le \frac{2}{\Phi^2} \left(\log \frac{1}{\min_i \boldsymbol{\pi}_i} + \log \frac{1}{\epsilon} \right).$$

In fact, it can be shown that without converting our Markov chain to a lazy Markov chain, we improve the bound by a factor of 2

$$T_{mix}(\epsilon) \le \frac{1}{\Phi^2} \left(\log \frac{1}{\boldsymbol{\pi}_{min}} + \log \frac{1}{\epsilon} \right).$$

18.3.1 Two-state example

Consider the simple Markov chain depicted in Figure 3 with a single binary variable x. By symmetry, the stationary distribution $\boldsymbol{\pi} = [0.5, 0.5]$. It's clear that Φ is minimized when $\mathcal{S} = \{0\}$ and $\mathcal{S}^c = \{1\}$ so that

$$\Phi = \frac{0.5\delta}{0.5 \times 0.5} = 2\delta.$$

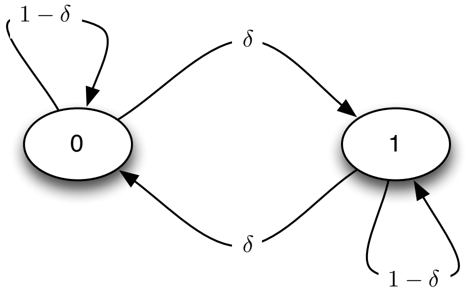

Figure 3

18.3.2 Concluding Remarks

We've seen how to sample from a distribution without knowing the partition function via Metropolis-Hastings. And we saw how we could bound the time it takes for the Markov chain to reach its stationary distribution from any initial distribution. In the next lecture we'll continue to discuss sampling techniques and in particular, we'll focus on techniques for a restricted set of models. For now, we'll take a brief aside to talk about approximate MAP algorithms.

18.4 Approximate MAP and Partitioning

Analogously to loopy belief propagation, we can run max-product (or min-sum) on a loopy graph to approximate the MAP. Unfortunately, we have limited understanding of the approximation it generates, as in the case with loopy belief propagation. The cases that are well understood suggest that max-product is akin to linear programming relaxation, but the discussion of this is beyond the scope of this class[5].

As we saw earlier, in the Gaussian setup, MAP and inference are equivalent. An interesting fact is that if Gaussian BP converges on a loopy graph, the estimated means are always correct[6].

Instead of pursuing loopy max-product, we'll focus on another generic procedure for approximating the MAP based on graph partitionings. The key steps in this approach are:

1. Partition the graph into small disjoint sets.

2. Estimate the MAP for each partition independently.

[5] For more information see Sanghavi, S.; Malioutov, D.; Willsky, A.'s "Belief Propagation and LP Relaxation for Weighted Matching in General Graphs" (2011).

[6] For more information see Weiss, Y; Freeman, W.'s "Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology" (2001).

3. Concatenate the MAPs for the subproblems to form a global estimate.

Because of its simplicity, this algorithm seems almost too good to be true and in fact given a specific partition on a graph, we can choose the clique potentials so that this algorithm will give a poor approximation to the MAP. The key is to exploit randomness! Instead of choosing a single partition, we'll define a distribution on partitions and when we select a partition from this distribution, we can guarantee that on average the algorithm will do well.

When a clever distribution on partitions of the graph exists, this produces a linear time algorithm and can be quite accurate. Unfortunately, not all graphs admit a good distribution on partitions, but in this case, we will can produce a bound on the approximation error. In the following section, we'll precisely define the algorithm and derive bounds on the approximation error. At the end, we'll explore how to find a clever distribution on partitions.

To make the notion of a "good" distribution on partitions precise, we'll define

Definition 3. *An (ϵ, k)-partitioning of graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a distribution on finite partitions of $\mathcal{V}$ such that for any partition $\{\mathcal{V}_1, \ldots, \mathcal{V}_M\}$ with non zero probability, $|\mathcal{V}_m| \leq k$ for all $1 \leq m \leq M$. Furthermore, we require that for any $e \in \mathcal{E}$, $p(e \in \mathcal{E}^c) \leq \epsilon$ where $\mathcal{E}^c = \mathcal{E} \setminus \cup_m (\mathcal{V}_m \times \mathcal{V}_m)$ the set of cut edges and the probability is with respect to the distribution on partitions.*

Intuitively, an (ϵ, k)-partitioning is a weighted set of partitions, such that in every partition all of the $\mathcal{V}_m$ are small and the set of cut edges is small. This aligns well with our algorithm because it means that the subproblems will be small because $\mathcal{V}_m$ is small, so our algorithm will be efficient. Because our algorithm evaluates the MAP for each partition independently, it misses out on the information contained on the cut edges, so as long as the set of cut edges is small, we do not miss much by ignoring them.

Let us consider a simple example of an $\sqrt{N} \times \sqrt{N}$ grid graph $\mathcal{G}$. We'll show that it's possible to find an $(\epsilon, \left(\frac{1}{\epsilon}\right)^2)$-partitioning for $\mathcal{G}$ for any $\epsilon > 0$. In this case, $k = \frac{1}{\epsilon^2}$. Our strategy will be to first construct a single partition has $|\mathcal{V}_m| \leq k$ and a small $|\mathcal{E}^c|$. Then we will construct a distribution on partitions that satisfies the constraint that for any $e \in \mathcal{E}$, $p(e \in \mathcal{E}^c)$.

Sub-divide the grid into $\sqrt{k} \times \sqrt{k}$ squares, each containing k nodes (Figure 5). There are $M \triangleq \frac{N}{k}$ such sub-squares; call them $\mathcal{V}_1, \ldots, \mathcal{V}_M$. By construction $|\mathcal{V}_m| \leq k$. The edges in $\mathcal{E}^c$ are the ones that cross between sub-squares. The number of edges crossing out of each such square is at most $4\sqrt{k}$, so the total number of such edges are at most $4M\sqrt{k}\frac{1}{2}$ where $\frac{1}{2}$ is for doubling counting edges. The total number of edges in the grid is roughly $2N$. Therefore, the fraction of cut edges is $2\sqrt{k}\frac{N}{k}\frac{1}{2N} = \frac{1}{\sqrt{k}} = \epsilon$.

Thinking of the sub-division into $\sqrt{k} \times \sqrt{k}$ squares as a coarse grid, we could shift the grid to the right and/or down to create a new partition. If we randomly shift the entire sub-grid uniformly $0, \ldots, \sqrt{k} - 1$ to the right and then uniformly

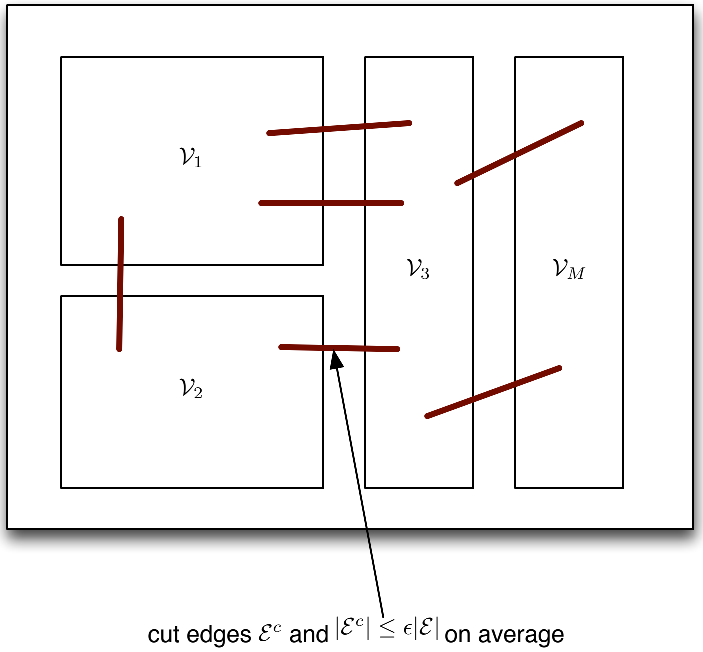
$\mathcal{V}_1$
$\mathcal{V}_2$
$\mathcal{V}_3$
$\mathcal{V}_M$
cut edges $\mathcal{E}^c$ and $|\mathcal{E}^c| \leq \epsilon|\mathcal{E}|$ on average

Figure 4: The nodes are partitioned into subsets $\mathcal{V}_1, \ldots, \mathcal{V}_M$ and the red edges correspond to cut edges.

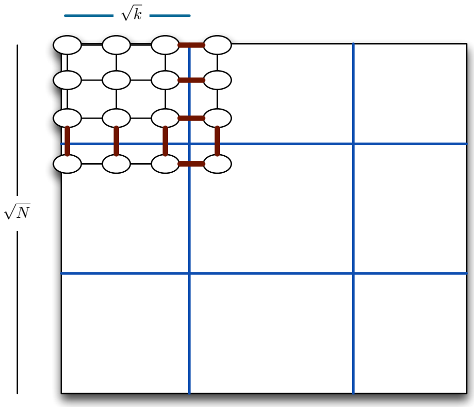

Figure 5: The original grid is sub-divided into a grid of $\sqrt{k} \times \sqrt{k}$ squares.

$0, \ldots, \sqrt{k} - 1$ down this gives a distribution on partitions. By symmetry, it ensures the distributional guarantee that $p(e \in \mathcal{E}^c) \leq \epsilon$. Thus the grid graph admits an $(\epsilon, \left(\frac{1}{\epsilon}\right)^2)-$partitioning for any $\epsilon > 0$.

18.4.1 Approximate MAP using (ϵ, k)-partitioning

In this section, we'll prove a bound on the approximation error when we have an (ϵ, k)-partitioning. For our analysis we will restrict our attention to pairwise MRFs that have non-negative potentials, so p takes the form

$$p_{\mathbf{x}}(\mathbf{x}) \propto \underbrace{\exp\left(\sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(x_i, x_j)\right)}_{\triangleq U(\mathbf{x})}$$

for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and where $\psi_i, \phi_{ij} \geq 0$. Then formally, the approximate MAP algorithm is

1. Given an (ϵ, k)-partitioning of $\mathcal{G}$, sample a partition $\{\mathcal{V}_1, \ldots, \mathcal{V}_M\}$ of $\mathcal{V}$.

2. For each $1 \leq m \leq M$: Using max-product on $\mathcal{G}_m = (\mathcal{V}_m, \mathcal{E} \cap \mathcal{V}_m \times \mathcal{V}_m)$ find

$$\hat{\mathbf{x}}_m \in \arg\max_{\mathbf{y} \in \mathcal{X}^{|\mathcal{V}_m|}} \underbrace{\sum_{i \in \mathcal{V}_m} \phi_i(y_i) + \sum_{\substack{(i,j) \in \mathcal{E} \\ i,j \in \mathcal{V}_m}} \psi_{ij}(y_i, y_j)}_{\triangleq U_m(\mathbf{y})}$$

3. Set $\hat{\mathbf{x}} = ((\hat{\mathbf{x}}_m)_m)$ as an approximation of the MAP.

We can get a handle on the approximation error by understanding how much error arises from ignoring the edge potentials corresponding to $\mathcal{E}^c$. If we use an (ϵ, k)-partitioning, then we expect $\mathcal{E}^c$ to be small, so we can bound our approximation error. The following theorem makes this intuition rigorous.

Theorem 2 (Jung-Shah)**.**

$$\mathbb{E}\left[U(\hat{\mathbf{x}})\right] \geq U(\mathbf{x}^*)(1 - \epsilon).$$

where the expectation is taken over the (ϵ, k)-partitioning.

Proof.

$$\begin{aligned}
U(\mathbf{x}^*) &= \sum_{i \in \mathcal{V}} \phi_i(x_i^*) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(x_i^*, x_j^*) \\
&= \sum_{m=1}^{M} \left[\sum_{i \in \mathcal{V}_m} \phi_i(x_i^*) + \sum_{\substack{(i,j) \in \mathcal{E} \\ i,j \in \mathcal{V}_m}} \psi_{ij}(x_i^*, x_j^*) \right] + \sum_{(i,j) \in \mathcal{E}^c} \psi_{ij}(x_i^*, x_j^*) \\
&= \sum_{m=1}^{M} U_m(\mathbf{x}^*) + \sum_{(i,j) \in \mathcal{E}^c} \psi_{ij}(x_i^*, x_j^*) \\
&\leq U(\hat{\mathbf{x}}) + \sum_{(i,j) \in \mathcal{E}} \mathbb{1}_{(i,j) \in \mathcal{E}^c} \psi_{ij}(x_i^*, x_j^*).
\end{aligned}$$

Therefore, by taking expectation with respect to randomness in partitioning and using the fact that $\phi_i, \psi_{i,j} \geq 0$ and $U(\mathbf{x}^*) \geq \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(x_i^*, x_j^*)$,

$$U(\mathbf{x}^*) \leq \mathbb{E}\left[U(\hat{\mathbf{x}})\right] + \epsilon U(\mathbf{x}^*).$$

□

This means that given our choice of ϵ, we can ensure that $\mathbb{E}\left[U(\hat{\mathbf{x}})\right]$ is close to the correct answer (i.e. the approximation error is small).

18.4.2 Generating (ϵ, k)-partitionings

We've seen that as long as we have an (ϵ, k)-partitioning for an MRF, then we can make guarantees about the approximation algorithm. Now we will show that a large class of graphs have (ϵ, k)-partitionings. First we'll describe a procedure for generating a **potential** (ϵ, k)-paritioning and then we'll see which class of graphs this realizes an (ϵ, k)-partitioning.

The procedure for generating a potential (ϵ, k)-partitioning on a graph is given by the following method for sampling a partition

1. Given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, k, and $\epsilon > 0$. Define the *truncated geometric distribution* with parameter ϵ truncated at k as follows

$$p(x = l) = \begin{cases} (1-\epsilon)^{l-1}\epsilon & l < k \\ (1-\epsilon)^{k-1} & l = k \end{cases}.$$

2. Order the nodes $\mathcal{V}$ arbitrarily $1, \ldots, N$. For node i:

   Sample R_i from a truncated geometric distribution with parameter ϵ truncated at k.

Assign all nodes within distance[7] R_i from i color i. If the node is already colored, recolor it to i.

3. All nodes with the same color form a partition.

This gives a partition and defines a distribution on partitions. The questions is: for what graphs $\mathcal{G}$ is this distribution an (ϵ, k)-partitioning?

Intuitively, for any given node, we want the number of nodes within some distance of it not to grow too quickly. Precisely,

Definition 4 (Poly-growth graph)**.** *A graph $\mathcal{G}$ is a poly-growth graph if there exists $\rho > 0, C > 0$ such that for any vertex v in the graph,*

$$|N_v(r)| \leq Cr^\rho,$$

where $N_v(r)$ is the number of nodes within distance r of v in $\mathcal{G}$.

In this case, we know that

Theorem 3 (Jung-Shah)**.** *If $\mathcal{G}$ is a poly-growth graph then by selecting $k = \Theta(\frac{\rho}{\epsilon} \log \frac{\rho}{\epsilon})$,* [8] *the above procedure results in an (ϵ, Ck^ρ) partition.*[9]

This shows that we have a large class of graphs where we can apply the procedure to generate an (ϵ, k)-partitioning, which guarantees that our approximation error is small and controlled by our choice of ϵ.

[7] Where distance is defined as the path length on the graph.

[8] This notation means that k is asymptotically bounded above and below by $\frac{\rho}{\epsilon} \log \frac{\rho}{\epsilon}$.

[9] A similar procedure exists for all planar graphs.