Stat260/CS294: Spectral Graph Methods **Lecture 17 - 03/19/2015**

Lecture: Diffusions and Random Walks as Robust Eigenvectors

Lecturer: Michael Mahoney *Scribe: Michael Mahoney*

Warning: these notes are still very rough. They provide more details on what we discussed in class, but there may still be some errors, incomplete/imprecise statements, etc. in them.

17 Diffusions and Random Walks as Robust Eigenvectors

Last time, we talked about electrical networks, and we saw that we could reproduce some of the things we have been doing with spectral methods with more physically intuitive techniques. These methods are of interest since they are typically more robust than using eigenvectors and they often lead to simpler proofs. Today, we will go into more detail about a similar idea, namely whether we can interpret random walks and diffusions as providing robust or regularized or stable analogues of eigenvectors. Many of the most interesting recent results in spectral graph methods adopt this approach of using diffusions and random walks rather than eigenvectors. We will only touch on the surface of this approach.

17.1 Overview of this approach

There are several advantages to thinking about diffusions and random walks as providing a robust alternative to eigenvectors.

- New insight into spectral graph methods.
- Robustness/stability is a good thing in many situations.
- Extend global spectral methods to local spectral analogues.
- Design new algorithms, e.g., for Laplacian solvers.
- Explain why diffusion-based heuristics work as they do in social network, computer vision, machine learning, and many other applications.

Before getting into this, step back for a moment, and recall that spectral methods have many nice theoretical and practical properties.

- **Practically.** Efficient to implement; can exploit very efficient linear algebra routines; perform very well in practice, in many cases better than theory would suggest. (This last claim means, e.g., that there is an intuition in areas such as computer vision and social network analysis that even if you could solve the best expansion/conductance problem, you wouldn't want to, basically since the approximate solution that spectral methods provide is "better.")

- **Theoretically.** Connections between spectral and combinatorial ideas; and connections between Markov chains and probability theory that provides a geometric viewpoint.

Recently, there have been very fast algorithms that combine spectral and combinatorial ideas. They rely on an optimization framework, e.g., solve max flow problems by relating them to these spectral-based optimization ideas. These use diffusion-based ideas, which are a relatively new trend in spectral graph theory.

To understand better this new trend, recall that the classical view of spectral methods is based on Cheeger's Inequality and involves computing an eigenvector and performing sweep cuts to reveal sparse cuts/partitions. The new trend is to replace eigenvectors with vectors obtained by running random walks. This has been used in:

- fast algorithms for graph partitioning and related problems;
- local spectral graph partitioning algorithms; and
- analysis of real social and information networks.

There are several different types of random walks, e.g., Heat Kernel, PageRank, etc., and different walks are better in different situations.

So, one question is: Why and how do random walks arise naturally from an optimization framework?

One advantage of a random walk is that to compute an eigenvector in a very large graph, a vanilla application of the power method or other related iterative methods (especially black box linear algebra methods) might be too slow, and so instead one might run a random walk on the graph to get a quick approximation.

Let $W = AD^{-1}$ be the natural random walk matrix, and let $L = D - A$ be the Laplacian. As we have discussed, it is well-known that the second eigenvector of the Laplacian can be computed by iterating W.

- For "any" vector y_0 (or "any" vector y_0 s.t. $y_0 D^{-1}\vec{1} = 0$ or any random vector y_0 s.t. $y_0 D^{-1}\vec{1} = 0$), we can compute $D^{-1}W^t y_0$; and we can take the limit as $t \to \infty$ to get

$$v_2(L) = \lim_{t\to} \frac{D^{-1}W^t y_0}{\|W^t y_0\|_{D^{-1}}},$$

  where $v_2(L)$ is the leading nontrivial eigenvector of the Laplacian.

- If time is a precious resource, then one alternative is to avoid iterating to convergence, i.e., don't let $t \to \infty$ (which of course one never does in practice, but by this we mean don't iterate to anywhere near machine precision), but instead do some sort of "early stopping." In that case, one does not obtain an eigenvector, but it is of interest to say something about the vector that is computed. In many cases, this is useful, either as an approximate eigenvector or as a locally-biased analogue of the leading eigenvector. This is very common in practice, and we will look at it in theory.

Another nice aspect of replacing an eigenvector with a random walk or by truncating the power iteration early is that the vectors that are thereby returned are more robust. The idea should be

familiar to statisticians and machine learners, although in a somewhat different form. Say that there is a "ground truth" graph that we want to understand but that the measurement we make, i.e., the graph that we actually see and that we have available to compute with, is a noisy version of this ground truth graph. So, if we want to compute the leading nontrivial eigenvector of the unseen graph, then computing the leading nontrivial eigenvector of the observed graph is in general not a particularly good idea. The reason is that it can be very sensitive to noise, e.g., mistakes or noise in the edges. On the other hand, if we perform a random walk and keep the random walk vector, then that is a better estimate of the ground truth eigendirection. (So, the idea is that eigenvectors are unstable but random walks are not unstable.)

A different but related question is the following: why are random walks useful in the design of fast algorithms? (After all, there is no "ground truth" model in this case—we are simply running an algorithm on the graph that is given, and we want to prove results about the algorithm applied to that graph.) The reason is similar, but the motivation is different. If we want to have a fast iterative algorithm, then we want to work with objects that are stable, basically so that we can track the progress of the algorithm. Working with vectors that are the output of random walks will be better in this sense. Today, we will cover an optimization perspective on this. (We won't cover the many applications of these ideas to graph partitioning and related algorithmic problems.)

17.2 Regularization, robustness, and instability of linear optimization

Again, take a step back. What is regularization? The usual way it is described (at least in machine learning and data analysis) is the following.

We have an optimization problem

$$\min_{x \in \mathcal{S}} f(x),$$

where $f(x)$ is a (penalty) function, and where $\mathcal{S}$ is some constraint set. This problem might not be particularly well-posed or well-conditioned, in the sense that the solution might change a lot if the input is changed a little. In order to get a more well-behaved version of the optimization problem, e.g., one whose solution changes more gradually as problem parameters are varied, one might instead try to solve the problem

$$\min_{x \in \mathcal{S}} f(x) + \lambda g(x),$$

where $\lambda \in \mathbb{R}^+$ is a parameter, and where $g(x)$ is (regularization) function. The idea is that $g(x)$ is "nice" in some way, e.g., it is convex or smooth, and λ governs the relative importance of the two terms, $f(x)$ and $g(x)$. Depending on the specific situation, the advantage of solving the latter optimization problem is that one obtains a more stable optimum, a unique optimum, or smoothness conditions. More generally, the benefits of including such a regularization function in ML and statistics is that: one obtains increased stability; one obtains decreased sensitivity to noise; and one can avoid overfitting.

Here is an illustration of the instability of eigenvectors. Say that we have a graph G that is basically an expander except that it is connected to two small poorly-connected components. That is, each of the two components is well-connected internally but poorly-connected to the rest of G, e.g., connected by a single edge. One can easily choose the edges/weights in G so that the leading non-trivial eigenvector of G has most of its mass, say a $1 - \epsilon$ fraction of its mass, on the first

small component. In addition, one can then easily construct a perturbation, e.g., removing one edge from G to construct a graph G' such that G' has a $1-\epsilon$ fraction of its mass on the second component. That is, a small perturbation that consists of removing one edge can completely shift the eigenvector—not only its direction but also where in $\mathbb{R}^n$ its mass is supported.

Let's emphasize that last point. Recalling our discussion of the Davis-Kahan theorem, as well as the distinction between the Rayleigh quotient objective and the actual partition found by performing a sweep cut, we know that if there is a small spectral gap, then eigenvectors can swing by 90 degrees. Although the example just provided has aspects of that, this example here is even more sensitive: not only does the direction of the vector change in $\mathbb{R}^n$, but also the mass along the coordinate axes in $\mathbb{R}^n$ where the eigenvector is localized changes dramatically under a very minor perturbation to G.

To understand this phenomenon better, here is the usual quadratic optimization formulation of the leading eigenvector problem. For simplicity, let's consider a d-regular graph, in which case we get the following.

$$\begin{aligned} \text{Quadratic Formulation}: \quad \tfrac{1}{d}\min_{x\in\mathbb{R}^n} \quad & x^T L x \\ \text{s.t.} \quad & \|x\| = 1 \\ & x \perp \vec{1}. \end{aligned} \tag{1}$$

This is an optimization over vectors $x \in \mathbb{R}^n$. Alternatively, we can consider the following optimization problem over SPSD matrices.

$$\begin{aligned} \text{SDP Formulation}: \quad \tfrac{1}{d}\min_{X\in\mathbb{R}^{n\times n}} \quad & L \bullet X \\ \text{s.t.} \quad & I \bullet X = 1 \\ & J \bullet X = 0 \\ & X \succeq 0, \end{aligned} \tag{2}$$

Recall that $I \bullet X = \mathbf{Tr}(X)$ and that $J = 11^T$. Recall also that if a matrix X is rank-one and thus can be written as $X = xx^T$, then $L \bullet X = x^T L x$.

These two optimization problems, Problem 1 and Problem 2, are equivalent, in that if x^* is the vector solution to the former and X^* is a solution of the latter, then $X^* = x^* x^{*T}$. In particular, note that although there is no constraint on the SDP formulation that the solution is rank-one, the solution turns out to be rank one.

Observe that this is a linear SDP, in that the objective and all the constraints are linear. Linear SDPs, just as LPs, can be very unstable. To see this in the simpler setting of LPs, consider a convex set $S \subset \mathbb{R}^n$ and a linear optimization problem:

$$f(c) = \arg\min_{x\in S} c^T x.$$

The optimal solution $f(c)$ might be very unstable to perturbations of c, in that we can have

$$\|c' - c\| \le \delta \quad \text{and} \quad \|f(c') - f(c)\| \gg \delta.$$

(With respect to our Linear SDP, think of the vector x as the PSD variable X and think of the vector c as the Laplacian L.) That is, we can change the input c (or L) a little bit and the solution changes a lot. One way to fix this is to introduce a regularization term $g(x)$ that is strongly convex.

So, consider the same convex set $S \subset \mathbb{R}^n$ and a regularized linear optimization problem

$$f(c) = \arg\min_{x \in S} c^T x + \lambda g(x),$$

where $\lambda \in \mathbb{R}^+$ is a parameter and where $g(x)$ is σ-strongly convex. Since this is just an illustrative example, we won't define precisely the term σ-strongly convex, but we note that σ is related to the derivative of $f(\cdot)$ and so the parameter σ determines how strongly convex is the function $g(x)$. Then, since σ is related to the slope of the objective at $f(c)$, and since the slope of the new objective at $f(c) < \delta$, strong convexity ensures that we can find a new optimum $f(c')$ at distance $< \frac{\delta}{\sigma}$. So, we have

$$\|c' - c\| \leq \delta \Rightarrow \|f(c') - f(c)\| < \delta/\sigma, \tag{3}$$

i.e., the strong convexity on $g(x)$ makes the problem stable that wasn't stable before.

17.3 Structural characterization of a regularized SDP

How does this translate to the eigenvector problem? Well, recall that the leading eigenvector of the Laplacian solves the SDP, where X appears linearly in the objective and constraints, as given in Problem 2. We will show that several different variants of random walks exactly optimize regularized versions of this SDP. In particular they optimize problems of the form

$$\text{SDP Formulation}: \quad \begin{array}{ll} \frac{1}{d}\min_{X \in \mathbb{R}^{n \times n}} & L \bullet X + \lambda G(X) \\ \text{s.t.} & I \bullet X = 1 \\ & J \bullet X = 0 \\ & X \succeq 0, \end{array} \tag{4}$$

where $G(X)$ is an appropriate regularization function that depends on the specific form of the random walk and that (among other things) is strongly convex.

To give an interpretation of what we are doing, consider the eigenvector decomposition of X, where

$$X = \sum_i p_i v_i v_i^T, \quad \text{where} \quad \begin{cases} \forall i \quad p_i \geq 0 \\ \sum_i p_i = 1 \\ \forall i \quad v_i^T \vec{1} = 0 \end{cases} \tag{5}$$

I've actually normalized things so that the eigenvalues sum to 1. If we do this, then the eigenvalues of X define a probability distribution. If we don't regularize in Problem 4, i.e., if we set $\lambda = 0$, then the optimal solution to Problem 4 puts all the weight on the second eigenvector (since $X^* = x^* x^{*T}$). If instead we regularize, then the regularization term ensures that the weight is spread out on all the eigenvectors, i.e., the optimal solution $X^* = \sum_i \alpha_i v_i v_i^T$, for some set of coefficients $\{\alpha_i\}_{i=1}^n$. So, the solution is not rank-one, but it is more stable.

Fact. If we take this optimization framework and put in "reasonable" choices for $G(X)$, then we can recover algorithms that are commonly used in the design of fast algorithms and elsewhere. That the solution is not rank one makes sense from this perspective: if we iterate $t \to \infty$, then all the other eigendirections are washed out, and we are left with the leading direction; but if we only iterate to a finite t, then we still have admixing from the other eigendirections.

To see this in more detail, consider the following three types of random walks. (Recall that $M = AD^{-1}$ and $W = \frac{1}{2}(I + M)$.)

- **Heat Kernel.** $H_t = \exp(-tL) = \sum_{k=0}^{\infty} \frac{(-t)^k}{k!} L^k = \sum_{i=1}^{n} e^{-\lambda_i t} P_i$, where P_i is a projection matrix onto that eigendirection.
- **PageRank.** $R_\gamma = \gamma (I - (1-\gamma) M)^{-1}$. This follows since the PageRank vector is the solution to
$$\pi(\gamma, s) = \gamma s + (1-\gamma) M \pi(\gamma, s),$$
which can be written as
$$\pi(\gamma, s) = \gamma \sum_{t=0}^{\infty} (1-\gamma)^t M^t s = R_\gamma s.$$
- **Truncated Lazy Random Walk.** $W_\alpha = \alpha I + (1-\alpha) M$.

These are formal expressions describing the action of each of those three types of random walks, in the sense that the specified matrix maps the input to the output: to obtain the output vector, compute the matrix and multiply it by the input vector. Clearly, of course, these random walks would not be implemented by computing these matrices explicitly; instead, one would iteratively apply a one-step version of them to the input vector.

Here are the three regularizers we will consider.

- **von Neumann entropy.** Here, $G_H = \mathbf{Tr}(X \log X) = \sum_i p_i \log p_i$.
- **Log-determinant.** Here, $G_D = -\log\det(X)$.
- **Matrix p-norm, $p > 0$.** Here, $G_p = \frac{1}{p}\|X\|_p^p = \frac{1}{p}\mathbf{Tr}(X^p) = \frac{1}{p}\sum_i p_i^p$.

And here are the connections that we want to establish.

- $G = G_H \Rightarrow^{entropy} X^* \sim H_t$, with $t = \lambda$.
- $G = G_D \Rightarrow^{logdet} X^* \sim R_\gamma$, with $\gamma \sim \lambda$.
- $G = G_p \Rightarrow^{p-norm} X^* \sim W_\alpha^t$, with $t \sim \lambda$.

Here is the basic structural theorem that will allow us to make precise this connection between random walks and regularized SDPs. Note that its proof is a quite straightforward application of duality ideas.

Theorem 1. *Recall the regularized SDP of Problem 4, and let $\lambda = 1/\eta$. If G is a connected, weighted, undirected graph, then let L be the normalized Laplacian. The the following are sufficient conditions for X^* to be the solution of the regularized SDP.*

1. $X^* = (\nabla G)^{-1}(\eta(\lambda^* I - L))$, *for some* $\lambda^* \in \mathbb{R}$.
2. $I \bullet X^* = 1$.
3. $X^* \succeq 0$.

Proof. Write the Lagrangian $\mathcal{L}$ of the above SDP as

$$\mathcal{L} = L \bullet X + \frac{1}{\eta} G(X) - \lambda \left(I \bullet X - 1 \right) - U \bullet X,$$

where $\lambda \in \mathbb{R}$ and where $U \succeq 0$. Then, the dual objective function is

$$h\left(\lambda, U\right) = \min_{X \succeq 0} \mathcal{L}(X, \lambda, U).$$

Since $G(\cdot)$ is strictly convex, differentiable, and rotationally invariant, the gradient of G over the positive semi-definite cone is invertible, and the RHS is minimized when

$$X = \left(\nabla G\right)^{-1} \left(\eta \left(-L + \lambda^* I + U \right) \right),$$

where λ^* is chosen s.t. $I \bullet X^* = 1$. Hence,

$$\begin{aligned} h\left(\lambda^*, 0\right) &= L \bullet X^* + \frac{1}{\eta} G(X^*) - \lambda^* \left(I \cdot X^* - 1 \right) \\ &= L \bullet X^* + \frac{1}{\eta} G(X^*). \end{aligned}$$

By Weak Duality, this implies that X^* is the optimal solution to the regularized SDP. □

17.4 Deriving different random walks from Theorem 1

To derive the Heat Kernel random walk from Theorem 1, let's do the following. Since

$$G_H(X) = \mathbf{Tr}\left(X \log(X)\right) - \mathbf{Tr}\left(X\right),$$

it follows that $\left(\nabla G\right)(X) = \log(x)$ and thus that $\left(\nabla G\right)^{-1}(Y) = \exp(Y)$, from which it follows that

$$\begin{aligned} X^* &= \left(\nabla G\right)^{-1} \left(\eta \left(\lambda I - L \right) \right) \\ &= \exp\left(\eta \left(\lambda I - L \right) \right) \quad \text{for an appropriate choice of } \eta, \lambda \\ &= \exp\left(-\eta L \right) \exp\left(\eta \lambda \right) \\ &= \frac{H_\eta}{\mathbf{Tr}\left(H_\eta\right)}, \end{aligned}$$

where the last line follows if we set $\lambda = \frac{-1}{\eta} \log\left(\mathbf{Tr}\left(\exp\left(-\eta L\right)\right)\right)$

To derive the PageRank random walk from Theorem 1, we follow a similar derivation. Since

$$G_D(X) = -\log\det\left(X\right),$$

it follows that $\left(\nabla G\right)(X) = -X^{-1}$ and thus that $\left(\nabla G\right)^{-1}(Y) = -Y^{-1}$, from which it follows that

$$\begin{aligned} X^* &= \left(\nabla G\right)^{-1} \left(\eta \left(\lambda I - L \right) \right) \\ &= -\left(\eta \left(\lambda I - L \right) \right)^{-1} \quad \text{for an appropriate choice of } \eta, \lambda \\ &= \frac{D^{-1/2} R_\gamma D^{-1/2}}{\mathbf{Tr}\left(R_\gamma\right)}, \end{aligned}$$

for η, λ chosen appropriately.

Deriving the truncated iterated random walk or other forms of diffusions is similar.

We will go into more details on the connection with PageRank next time, but for now we just state that the solution can be written in the form

$$x^* = c\,(L_G - \alpha L_{K_n})^+ Ds,$$

for a constant c and a parameter γ. That is, it is of the form of the solution to a linear equation, i.e., $L_G^+ s$, except that there is a term that moderates the effect of the graph by adding the Laplacian of the complete graph. This is essentially a regularization term, although it is not usually described as such. See the Gleich article for more details on this.

17.5 Interpreting Heat Kernel random walks in terms of stability

Here, we will relate the two previous results for the heat kernel. Again, for simplicity, assume that G is d-regular. Recall that the Heat Kernel random walk is a continuous time Markov chain, modeling the diffusion of heat along the edges of G. Transitions take place in continuous time t with an exponential distribution:

$$\frac{\partial \rho(t)}{\partial t} = -L\frac{\rho(t)}{d} \Rightarrow \rho(t) = \exp\left(-\frac{t}{d}L\right)\rho(0).$$

That is, this describes the way that the probability distribution changes from one step to the next and how it is related to L. In particular, the Heat Kernel can be interpreted as a Poisson distribution over the number of steps of the natural random walk $W = AD^{-1}$, where we get the following:

$$e^{-\frac{t}{d}L} = e^{-t}\sum_{k=1}^{\infty}\frac{t^k}{k!}W^k.$$

What this means is: pick a number of steps from the Poisson distribution; and then perform that number of steps of the natural random walk.

So, if we have two graphs G and G' and they are close, say in an ℓ_∞ norm sense, meaning that the edges only change a little, then we can show the following. (Here, we will normalize the two graphs so that their respective eigenvalues sum to 1.) The statement analogous to Statement 3 is the following.

$$\|G - G'\|_\infty \Rightarrow \|\frac{H_G^t}{I \bullet H_G^t} - \frac{H_{G'}^t}{I \bullet H_{G'}^t}\|_1 \leq t\delta.$$

Here, $\|\cdot\|_1$ is some other norm (the ℓ_1 norm over the eigenvalues) that we won't describe in detail. Observe that the bound on the RHS depends on how close the graphs are (δ) as well as how long the Heat Kernel random walk runs (t). If the graphs are far apart (δ is large), then the bound is weak. If the random walk is run for a long time ($t \to \infty$), then the bound is also very weak. But, if the walk is nor run too long, then we get a robustness result. And, this follows from the strong convexity of the regularization term that the heat kernel is implicitly optimizing exactly.

17.6 A statistical interpretation of this implicit regularization result

Above, we provided three different senses in which early-stopped random walks can be interpreted as providing a robust or regularized notion of the leading eigenvectors of the Laplacian: e.g., in

the sense that, in addition to approximating the Rayleigh quotient, they also exactly optimize a regularized version of the Rayleigh quotient. Some people interpret regularization in terms of statistical priors, and so let's consider this next.

In particular, let's now give a statistical interpretation to this implicit regularization result. By a "statistical interpretation," I mean a derivation analogous to the manner in which ℓ_2 or ℓ_1 regularized ℓ_2 regression can be interpreted in terms of a Gaussian or Laplace prior on the coefficients of the regression problem. This basically provides a Bayesian interpretation of regularized linear regression. The derivation below will show that the solutions to the Problem 4 that random walkers implicitly optimize can be interpreted as a regularized estimate of the pseudoinverse of the Laplacian, and so in some sense it provides a Bayesian interpretation of the implicit regularization provided by random walks.

To start, let's describe the analogous results for vanilla linear regression. For some (statistics) students, this is well-known; but for other (non-statistics) students, it likely is not. The basic idea should be clear; and we cover it here to establish notation and nomenclature. Let's assume that we see n predictor-response pairs in $\mathbb{R}^p \times \mathbb{R}$, call them $\{(x_i, y_i)\}_{i=1}^n$, and the goal is to find a parameter vector $\beta \in \mathbb{R}^p$ such that $\beta^T x_i \approx y_i$. A common thing to do is to choose β by minimizing the RSS (residual sum of squares), i.e., choosing

$$F(\beta) = RSS(\beta) = \sum_{i=1}^{n} \|y_i - \beta^T x_i\|_2^2.$$

Alternatively, we could optimize a regularized version of this objective. In particular, we have

$$\begin{aligned} \text{Ridge regression:} \quad & \min_{\beta} F(\beta) + \lambda \|\beta\|_2^2 \\ \text{Lasso regression:} \quad & \min_{\beta} F(\beta) + \lambda \|\beta\|_1. \end{aligned}$$

To derive these two versions of regularized linear regression, let's model y_i as independent random variables with distribution dependent on β as follows:

$$y_i \sim N\left(\beta^T x, \sigma^2\right), \tag{6}$$

i.e., each y_i is a Gaussian random variable with mean $\beta^T x_i$ and known variance σ^2. This induces a conditional density for y as follows:

$$p\left(y|\beta\right) \sim \exp\{\frac{-1}{2\sigma^2} F(\beta)\}, \tag{7}$$

where the constant of proportionality depends only on y and σ. From this, we can derive the vanilla least-squares estimator. But, we can also assume that β is a random variable with distribution $p(\beta)$, which is known as a prior distribution, as follows:

$$p(\beta) \sim \exp\{-U(\beta)\}, \tag{8}$$

where we adopt that functional form without loss of generality. Since these two random variables are dependent, upon observing y, we have information on β, and this can be encoded in a posterior density, $p\left(\beta|y\right)$, which can be computed from Bayes' rule as follows:

$$\begin{aligned} p\left(\beta|y\right) &\sim p\left(y|\beta\right) p(\beta) \\ &\sim \exp\{\frac{-1}{2\sigma^2} F(\beta) - U(\beta)\}. \end{aligned} \tag{9}$$

We can form the MAP, the maximum a posteriori, estimate of β by solving

$$\max_{\beta} p(\beta|y) \;\; \textbf{iff} \;\; \min_{\beta} -\log p(\beta|y).$$

From this we can derive ridge regression and Lasso regression:

$$\begin{aligned} U(\beta) &= \frac{\lambda}{2\sigma^2}\|\beta\|_2^2 \quad \Rightarrow \text{Ridge regression} \\ U(\beta) &= \frac{\lambda}{2\sigma^2}\|\beta\|_1 \quad \Rightarrow \text{Lasso regression} \end{aligned}$$

To derive the analogous result for regularized eigenvectors, we will follow the analogous setup. What we will do is the following. Given a graph G, i.e., a "sample" Laplacian L, assume it is a random object drawn from a "population" Laplacian $\mathcal{L}$.

- This induces a conditional density for L, call it $p(L|\mathcal{L})$.
- Then, we can assume prior information about the population Laplacian $\mathcal{L}$ in the form of $p(\mathcal{L})$.
- Then, given the observed L, we can estimate the population Laplacian by maximizing its posterior density $p(\mathcal{L}|L)$.

While this setup is analogous to the derivation for least-squares, there are also differences. In particular, one important difference between the two approaches is that here there is one data point, i.e., the graph/Laplacian is a single data point, and so we need to invent a population from which it was drawn. It's like treating the entire matrix X and vector y in the least-squares problem as a single data point, rather than n data points, each of which was drawn from the same distribution. (That's not a minor technicality: in many situations, including the algorithmic approach we adopted before, it is more natural to think of a graph as a single data point, rather than as a collection of data points, and a lot of statistical theory breaks down when we observe $N = 1$ data point.)

In more detail, recall that a Laplacian is an SPSD matrix with a very particular structure, and let's construct/hypothesize a population from which it was drawn. To do so, let's assume that nodes n in the population and in the sample have the same degrees. If $d = (d_1, \ldots, d_n)$ is the degree vector, and $D = \deg(d_1, \ldots, d_n)$ is the diagonal degree matrix, then we can define the set

$$\chi = \{X : X \succeq 0, XD^{1/2}\vec{1} = 0, \text{rank}(X) = n - 1\}.$$

So, the population Laplacian and sample Laplacian are both members of χ. To model L, let's use a scaled Wishart matrix with expectation $\mathcal{L}$. (This distribution plays the role of the Gaussian distribution in the least-squares derivation. Note that this is a plausible thing to assume, but other assumptions might be possible too.) Let $m \geq n - 1$ be a scale parameter (analogous to the variance), and suppose that L is distributed over χ as $\frac{1}{m}\text{Wishart}(\mathcal{L}, m)$. Then $\mathbb{E}[L|\mathcal{L}] = \mathcal{L}$ and L has the conditional density

$$p(L|\mathcal{L}) \sim \frac{1}{\det(\mathcal{L})^{m/2}} \exp\{\frac{-m}{2}\textbf{Tr}\left(L\mathcal{L}^{+}\right)\}. \tag{10}$$

This is analogous to Eqn (7) above. Next, we can say that $\mathcal{L}$ is a random object with prior density $p(\mathcal{L})$, which without loss of generality we can take to be of the following form:

$$p(\mathcal{L}) \sim \exp\{-U(\mathcal{L})\},$$

where U is supported on a subset $\bar{\chi} \subseteq \chi$. This is analogous to Eqn (8) above. Then, observing L, the posterior distribution for $\mathcal{L}$ is the following:

$$\begin{aligned} p(\mathcal{L}|L) &\sim p(L|\mathcal{L})\, p(\mathcal{L}) \\ &\sim \exp\{\frac{m}{2}\mathbf{Tr}\left(L\mathcal{L}^{+}\right) + \frac{m}{2}\log\det\left(\mathcal{L}^{+}\right) - U(\mathcal{L})\}, \end{aligned}$$

with support determined by $\bar{\chi}$. This is analogous to Eqn (9) above.

If we denote by $\hat{\mathcal{L}}$ the MAP estimate of $\mathcal{L}$, then it follows that $\hat{\mathcal{L}}^{+}$ is the solution of the following optimization problem.

$$\begin{array}{ll} \min_X & \mathbf{Tr}(L \bullet X) + \dfrac{2}{m}U\left(X^{+}\right) - \log\det(X) \\ \text{s.t.} & X \in \bar{\chi} \subseteq \chi. \end{array} \tag{11}$$

If $\bar{\chi} = \{X : \mathbf{Tr}(X) = 1\} \cap \chi$, then Problem 11 is the same as Problem 4, except for the factor of $\log\det(x)$.

This is almost the regularized SDP we had above.

Next, we present a prior that will be related to the PageRank procedure. This will make the connection with the regularized SDP more precise. In particular, we present a prior for the population Laplacian that permits us to exploit the above estimation framework to show that the MAP estimate is related to a PageRank computation.

The criteria for the prior are so-called neutrality and invariance conditions. It is to be supported on χ; and in particular, for any $X \in \chi$, it will have rank $n-1$ and satisfy $XD^{1/2}1 = 0$. The prior will depend only on the eigenvalues of the Laplacian (or equivalently of the inverse Laplacian). Let $\mathcal{L}^{+} = \tau O \Lambda O$ be the spectral decomposition of the inverse Laplacian, where τ is a scale parameter. We will require that the distribution for $\lambda = (\lambda_1, \ldots, \lambda_n)$ be exchangeable (i.e., invariant under permutations) and neutral (i.e., $\lambda(v)$ is independent of $\frac{\lambda(u)}{1-\lambda(v)}$, for $u \neq v$, for all v). The only non-degenerate possibility is that λ is distributed as a Dirichlet distribution as follows:

$$p(\mathcal{L}) \sim p(\tau) \prod_{v=1}^{n-1} \lambda(v)^{\alpha-1}, \tag{12}$$

where α is a so-called shape parameter. Then, we have the following lemma.

Lemma 1. *Given the conditional likelihood for L given $\mathcal{L}$ in Eqn. (10) and the prior density for $\mathcal{L}$ given in Eqn. (12); if $\hat{\mathcal{L}}$ is the MAP estimate of $\mathcal{L}$, then*

$$\frac{\hat{\mathcal{L}}^{+}}{\mathbf{Tr}\left(\hat{\mathcal{L}}^{+}\right)}$$

solves the regularized SDP, with $G(X) = --\log\det(X)$ and with the value of η given in the proof below.

Proof. For $\mathcal{L}$ in the support set of the posterior, we can define $\tau = \mathbf{Tr}(\mathcal{L}^{+})$ and $\Theta = \frac{1}{\tau}\mathcal{L}^{+}$, so that $\mathrm{rank}(\Theta) = n-1$ and $\mathbf{Tr}(\Theta) = 1$. Then, $p(\mathcal{L}) \sim \exp\{-U(\mathcal{L})\}$, where

$$U(\mathcal{L}) = -\log\{p(\tau)\det(\Theta)^{\alpha-1}\} = -(\alpha-1)\log\det(\Theta) - \log(p(\tau)).$$

Thus,

$$\begin{aligned} p\left(\mathcal{L}|L\right) &\sim \exp\{-\frac{m}{2}\mathbf{Tr}\left(L\mathcal{L}^{+}\right)+\frac{m}{2}\log\det\left(\mathcal{L}^{+}\right)-U\left(\mathcal{L}\right)\} \\ &\sim \exp\{\frac{-m\tau}{2}\mathbf{Tr}\left(L\Theta\right)+\frac{m+2(\alpha-1)}{2}\log\det\left(\Theta\right)+g(\tau)\}, \end{aligned}$$

where the second line follows since $\det\left(\mathcal{L}^{+}\right)=\tau^{n-1}\det\left(\Theta\right)$, and where $g(\tau)=\frac{m(n-1)}{2}\log(\tau)+\log p(\tau)$. If $\hat{\mathcal{L}}$ maximizes the posterior likelihood, then define $\hat{\tau}=\mathbf{Tr}\left(\hat{\mathcal{L}^{+}}\right)$ and $\hat{\Theta}=\frac{1}{\tau}\hat{\mathcal{L}}^{+}$, and so $\hat{\Theta}$ must minimize $\mathbf{Tr}\left(L\hat{\Theta}\right)-\frac{1}{\eta}\log\det\left(\hat{\Theta}\right)$, where

$$\eta=\frac{m\hat{\tau}}{m+2(\alpha-1)}$$

This $\hat{\Theta}$ solves the regularized SDP with $G(x)=-\log\det\left(X\right)$. □

Remark. Lemma 1 provides a statistical interpretation of the regularized problem that is optimized by an approximate PageRank diffusion algorithm, in the sense that it gives a general statistical estimation procedure that leads to the Rayleigh quotient as well as statistical prior related to PageRank. One can write down priors for the Heat Kernel and other random walks; see the two references if you are interested. Note, however, that the prior for PageRank makes things particularly simple. The reason is that the extra term in Problem 11, i.e., the $\log\det\left(X\right)$ term, is of the same form as the regularization function that the approximate PageRank computation implicitly regularizes with respect to. Thus, we can choose parameters to make this term cancel. Otherwise, there are extra terms floating around, and the statistical interpretation is more complex.