Lecture 16: Solving Linear Equations for Sparse Graphs *

Rajat Mittal

IIT Kanpur

We introduced linear algebra by asking a question, how to solve

$$Ax = b.$$

Our last application of spectral methods will be to solve such equations fast when A is a Laplacian matrix (we will call it L). Remember that Gaussian elimination is the standard way to solve such equations. Though, it is expensive when matrix L is sparse.

For most of the lecture, we will talk about *iterative methods* to solve such equations. These methods give approximate solutions, such that, error decreases as you increase the number of iterations (something like Newton iteration).

In the end, we will see that sparse graph approximations help us solve $Lx = b$ really fast. We will also see a sparsification technique. The iterative methods mentioned below will work for positive semidefinite/definite matrices. So, assume that we are given a positive semidefinite matrix A and we are interested in solving $Ax = b$.

For most of these iterative algorithms, the number of iterations depend on a quantity called *condition number* of the matrix A. Condition number, $\kappa(A)$, of a positive definite matrix A is defined to be $\frac{\lambda_n}{\lambda_1}$, where λ_1 is the least eigenvalue and λ_n is the largest eigenvalue for A.

Note 1. Don't worry about the positive semidefinite case yet. We can restrict our attention to the support of the matrix and consider it as positive definite.

The first iterative method we will look at is called *Richardson's iteration.*

1 Richardson's iteration

1.1 Idea behind Richardson's iteration

Let us first look at the intuition behind Richardson's iteration. This subsection is written in an informal manner. More formal approach will be presented in the next subsection.

We know the power series expansion of $(1-x)^{-1}$,

$$\frac{1}{1-x} = 1 + x + x^2 \cdots$$

Solving $Ax = b$ for many b's is similar to the problem of computing A^{-1}. So, if the power series converges,

$$A^{-1} = (I - (I - A))^{-1} = I + (I - A) + (I - A)^2 \cdots$$

When should we expect this power series to converge? If we can show that the norm of $(I - A)^n$ tends to 0 as n tends to infinity, it will hint that the power series converges.

Exercise 1. Can you think of a sequence $s(n)$ tending to 0, such that, $\sum_{i=1}^{n} s(n)$ does not converge?

Remember, from our analysis of random walk, that if the eigenvalues of $(I - A)$ have absolute value less than 1 then the spectral norm of $(I - A)^n$ tends to zero. That is same as saying that the eigenvalues of A lie between 0 and 2.

This is definitely not true for a general positive definite matrix.

* The content of these notes is largely taken from Dan Spielman's course notes

Exercise 2. What can be done to have the eigenvalues of A lie between 0 and 2?

One possible solution might be to scale the matrix A. It is already positive definite; there should exist α, for which, eigenvalues of αA will lie between 0 and 2.

Scaling A means, we want to solve the equation,

$$\alpha A x = \alpha b.$$

Richardson's method is to choose an α intelligently and then find better and better approximation of $(I - (I - \alpha A))^{-1}\alpha b$. In the t-th iteration, Richardson's method will output,

$$x_t = (\sum_{i=0}^{t} (I - \alpha A)^i)\alpha b.$$

It is called an iterative method, because of the easy recurrence,

$$x_t = (I - \alpha A)x_{t-1} + \alpha b.$$

Exercise 3. What will be x_0?

Every iteration needs one multiplication by A and some additions between vectors.

We still haven't answered one question, how to choose α? Taking intuition from the analysis of random walks, we would like to choose α, such that, the norm of $I - \alpha A$ is minimized.

Exercise 4. What are the eigenvalues of $I - \alpha A$ in terms of A?

If $\lambda_1 \leq \lambda_2 \cdots \leq \lambda_n$ were the eigenvalues of A, then $(1 - \alpha\lambda_1) \geq (1 - \alpha\lambda_2) \cdots \geq (1 - \alpha\lambda_n)$ are the eigenvalues of $I - \alpha A$.

The norm will be minimized when the extreme eigenvalues have same absolute value, i.e.,

$$1 - \alpha\lambda_1 = -(1 - \alpha\lambda_n).$$

So, the best α is $\frac{2}{\lambda_1 + \lambda_n}$.

Let us look at the error as number of iterations increase in Richardson's iteration.

1.2 Analysis of Richardson's iteration

Let x be a solution of $Ax = b$. Then,

$$x = (I - \alpha A)x + \alpha b.$$

We would like to bound the distance $\|x - x_t\|$ as t increases. Here, x_t is the solution after t iteration of Richardson's method. Then,

$$x - x_t = (I - \alpha A)(x - x_{t-1}).$$

This implies,

$$x - x_t = (I - \alpha A)^t(x - x_0) = (I - \alpha A)^{t+1}x.$$

This equation is very similar to the one obtained in the analysis of random walk. Since, the eigenvalues of $I - \alpha A$ are small, this error should tend to zero.

Assuming α is $\frac{2}{\lambda_1 + \lambda_n}$, we know that the biggest eigenvalue of $I - \alpha A$ is $1 - \frac{2\lambda_1}{\lambda_1 + \lambda_n}$. So,

$$\frac{\|x - x_t\|}{\|x\|} \leq \left\|(I - \alpha A)^{t+1}\right\| = \left(1 - \frac{2\lambda_1}{\lambda_1 + \lambda_n}\right)^{t+1}.$$

Exercise 5. Show that we need to have $O(\frac{\lambda_1 + \lambda_n}{2\lambda_1})$ iterations to bound the error by a constant.

To conclude, we can say that we need to run Richardson's method for around $O(\kappa)$ iterations. Remember that $\kappa = \lambda_n/\lambda_1$ is the condition number of matrix A.

Can we reduce the number of iterations? Next, we will see that we can find an approximation of inverse of A faster (less number of iterations) by using Chebyshev's polynomials.

2 Approximation of inverse using Chebyshev polynomials

In the previous section for Richardson's method, we saw that finding a series of approximate inverses of A was helpful in solving $Ax = b$. Let us try to formally define what we mean by this series of approximate inverses in this case.

In last section, we approximated A^{-1} by the sequence $\{\sum_{i=0}^{t}(I-A)^t\}_t$. Let us call the polynomial in A which we use as inverse in the t-th iteration to be $p^t(A)$.

Let us look at all the conditions we need for $p^t(A)$.

- We want $p^t(A)$ to be a polynomial because we should be able to compute it efficiently.
- We should be able to compute $p^t(A)$ from $p^{t-1}(A)$ quickly.
- Last but not the least, $p^t(A)A$ should be close to identity, i.e.,

$$\left\|p^t(A)A - I\right\| \leq \epsilon_t.$$

  Here, we would like ϵ_t to reach zero quickly.

As mentioned before, we used $p^t(A) = \sum_{i=0}^{t}(I-A)^t$ in Richardson's iteration and got close to the solution in $O(\kappa)$ iterations. Can we do better?

Let us look at the condition $\|Ap^t(A) - I\|$ being small more closely. You will show in the assignment that the eigenvalues of this matrix are $\lambda_i p^t(\lambda_i) - 1$ for eigenvalues λ_i of A.

So, we need to show that $|\lambda_i p^t(\lambda_i) - 1|$ is small. Define $q^t(x) = 1 - xp^t(x)$, where $deg(q^t)$ is $t+1$. If we show that $q^t(0) = 1$ and $|q^t(x)| \leq \epsilon_t$ for all x between λ_1 and λ_n, we can find a good series of approximation for A.

Exercise 6. Show that any polynomial q, such that $q(0) = 1$, can be written as $1 - xp(x)$ where $deg(p)$ is one less than the degree of q.

So, our new target is to find q, s.t., $q(0) = 1$ and $|q(x)|$ is small when x is between λ_1 and λ_n. We will create such polynomials from existing family of polynomials called Chebyshev polynomials.

For us, it is sufficient to note that there exist polynomials (called Chebyshev polynomials) $T_t(x)$ such that,

- $T_{t+1}(x) = 2xT_t(x) - T_{t-1}(x)$, $T_0(x) = 1$ and $T_1(x) = x$. So, the degree of T_t is t and they can be computed iteratively (because of recurrence).
- $T_t(x) \in [-1, 1]$ for $x \in [-1, 1]$ and increase monotonically afterwards.
- $T_t(1 + x) \geq (1 + \sqrt{2x})^t/2$ for $x \geq 0$.

These are the polynomials which stay in $[-1, 1]$ when $x \in [-1, 1]$ and grow rapidly afterwards. Given the conditions on q, we would like to map λ_1 to 1 and λ_n to -1 using a linear map.

Exercise 7. Why did we not map λ_1 to -1 and λ_n to 1?

Solving $a + b\lambda_1 = 1$ and $a + b\lambda_n = -1$, we get the linear map

$$f(x) := \frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1} - \frac{2}{\lambda_n - \lambda_1}x.$$

Since we want $q^t(0) = 1$, define,

$$q^t(x) := \frac{T_t(f(x))}{T_t(f(0))}.$$

We now know that between λ_1 and λ_n, $|q^t(x)|$ is less than $|1/T_t(f(0))|$. To establish the rate of convergence, we only need to lower bound the value of $T_t(f(0))$.

We know that $f(0) = \frac{\lambda_n+\lambda_1}{\lambda_n-\lambda_1} \geq 1 + 2/\kappa$. Using the bound on Chebyshev polynomials after 1,

$$T_t(f(0)) \geq (1 + 2/\sqrt{\kappa})^t/2.$$

This implies that we will hit constant epsilon when $t = O(\sqrt{\kappa})$.

We crucially needed the fact that $\lambda_1 > 0$ for Chebyshev's method to work. We also know, $\lambda_1 = 0$ for a Laplacian. This would imply that Chebyshev polynomial method will not work for Laplacian.

Though, a small change can make the above method work. We restrict our attention to the non zero eigenspaces of L. L is positive definite there and we can find an approximate inverse in that space.

3 Preconditioning

For this section, the error in our algorithm will be measured in A-norm,

$$\|x\|_A = \sqrt{x^T A x},$$

instead of the 2-norm used in the previous section.

We want to improve the running time of our iterative algorithms. Our next idea is to find a matrix C, s.t., we will solve $CAx = Cb$ instead of $Ax = b$. In this case, it should be easier to solve linear equations in CA as compared to linear equations in A. Matrices C's, such that, it is easy to solve linear equations in CA are called *preconditioners*.

We saw in the last section that the time required to solve linear equations $Ax = b$ depends on the condition number of A. If the condition number is big, we want to find faster ways to solve $Ax = b$. So, one strategy might be to find a preconditioner C, s.t., the condition number of CA is small.

What does it mean that the condition number is small (close to one)? This implies that all the eigenvalues are close to 1 under some scaling. In other words, matrix is close to identity. To formalize the strategy mentioned above, we first need the definition of *close*.

3.1 Approximation

A symmetric matrix A is ϵ approximated by a symmetric matrix B iff

$$(1 - \epsilon)B \preceq A \preceq (1 + \epsilon)B. \tag{1}$$

Note 2. This is an extension of the notion of graph approximation studied before.

This is a pretty strong notion of approximation, because it implies that for every vector x,

$$(1 - \epsilon)x^T Bx \preceq x^T Ax \preceq (1 + \epsilon)x^T Bx.$$

Since the quadratic form captures many properties of matrices, all those properties of A are approximated by B. For one, if λ_i, μ_i are the i-th eigenvalue of A, B respectively, then

$$(1 - \epsilon)\mu_i \leq \lambda_i \leq (1 + \epsilon)\mu_i.$$

You will prove this in the assignment. Next, we assume that A and B are positive definite.

Exercise 8. From approximation between A and B (Eqn. 1), prove that,

$$(1 - \epsilon)I \preceq B^{-1/2} A B^{-1/2} \preceq (1 + \epsilon)I.$$

The exercise implies that the eigenvalues of $B^{-1/2} A B^{-1/2}$ are very close to identity and hence the condition number is pretty small.

Exercise 9. Show that $B^{-1/2}AB^{-1/2}$ is similar to $B^{-1}A$.

This implies that the eigenvalues of $B^{-1}A$ are close to 1 and hence condition number is small. So, if B approximates A well, then we can use B^{-1} as a preconditioner.

To conclude, matrix approximation gives us a strategy to solve $Ax = b$ fast. We need to find a matrix B, s.t.,

- B is an ϵ approximation of A,
- B^{-1} can be computed fast,
- We can find B fast.

In the following subsections, we will see, how these B's can help us solve linear equations fast. In the next section, we will construct approximations for a Laplacian.

3.2 Preconditioning with Richardson's method

Let us look at one way in which such an approximation can be helpful ($B^{-1}A$ has eigenvalues close to 1). To solve $Ax = b$, we will essentially solve $B^{-1}Ax = B^{-1}b$.

Our first guess for the solution will be $x_0 = B^{-1}b$. The error obtained is $\left\|x - B^{-1}b\right\|_A = \left\|(I - B^{-1}A)x\right\|_A$, where x is the solution. Analyzing,

$$\begin{aligned}
\left\|(I - B^{-1}A)x\right\|_A &= \left\|x - B^{-1}Ax\right\|_A \\
&= \left\|A^{1/2}x - A^{1/2}B^{-1}Ax\right\| \\
&= \left\|(I - A^{1/2}B^{-1}A^{1/2})A^{1/2}x\right\| \\
&\leq \left\|I - A^{1/2}B^{-1}A^{1/2}\right\| \|x\|_A \\
&\leq \epsilon\|x\|_A.
\end{aligned}$$

Here, the last equality follows because the matrix $I - A^{1/2}B^{-1}A^{1/2}$ and $I - B^{-1}A$ are similar.

If we want to improve the approximation, we go to the next iteration of Richardson's method. That means, our next attempt at solution will be,

$$x_1 = B^{-1}b + (I - B^{-1}A)x_0.$$

Again, calculating the error,

$$\|x - x_1\|_A = \left\|x - x_0 - B^{-1}(b - Ax_0)\right\|_A.$$

Exercise 10. Show that $\|x - x_1\|_A$ is less than $\epsilon^2\|x\|_A$.

In general, $x_t = B^{-1}b + (I - B^{-1}A)x_{t-1}$ and you will show in the assignment that $\|x - x_t\|_A \leq \epsilon^t\|x\|_A$.

3.3 Preconditioning on Chebyshev's method

Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of $B^{-1}A$.

From the section on Chebyshev's method, we know that there exist polynomial $q(x)$, s.t.,

- $q(0) = 1$,
- $q(x) \leq \epsilon$, when x is between λ_1 and λ_n,
- degree of q is $O(\sqrt{\frac{\lambda_n}{\lambda_1}}\log(\frac{1}{\epsilon}))$.

Note 3. We ran Chebyshev method till the error became constant, it can be run till the error becomes ϵ.

Since $q(0) = 1$, set $q(x) = 1 - xp(x)$ like before. Then, with preconditioning, the solution given by Chebyshev's method would be,

$$x_c = p(B^{-1}A)B^{-1}b.$$

Let x be the vector, s.t., $Ax = b$. We will show that $\|x - x_c\|_A \leq \epsilon \|x\|_A$. Looking at the A-norm,

$$\begin{aligned}
\|x - x_c\|_A &= \left\| A^{1/2}(x - x_c) \right\| \\
&= \left\| (I - A^{1/2}p(B^{-1}A)B^{-1}A^{1/2})A^{1/2}x \right\| \\
&\leq \left\| I - A^{1/2}p(B^{-1}A)A^{-1/2}A^{1/2}B^{-1}A^{1/2} \right\| \|x\|_A \\
&= \left\| I - p(A^{1/2}B^{-1}A^{1/2})A^{1/2}B^{-1}A^{1/2} \right\| \|x\|_A \\
&= \left\| q(A^{1/2}B^{-1}A^{1/2}) \right\| \|x\|_A \\
&\leq \epsilon \|x\|_A.
\end{aligned}$$

The last inequality follows because $A^{1/2}B^{-1}A^{1/2}$ is similar to $B^{-1}A$. The third last inequality follows from the following exercise.

Exercise 11. Let f be a polynomial, show that,

$$A^{1/2}f(CA)A^{-1/2} = f(A^{1/2}CA^{1/2}).$$

We will run Chebyshev's method on $B^{-1}A$ for $O(\sqrt{\kappa(B^{-1}A)}\log(\frac{1}{\epsilon}))$ iterations and get a solution which is ϵ close to actual solution x. This might not seems like an improvement if we just look at the expression. Though, the new κ is significantly smaller and hence the number of iterations have reduced.

4 Graph Sparsification

Suppose we want to solve $L_G x = b$ where L_G is a Laplacian of a graph G. From previous section, we need a matrix K, s.t.,

- K is an ϵ approximation of L_G,
- K^{-1} can be computed fast,
- We can find H fast.

A graph H is called a sparsifier of G if it has few edges and is a good approximation of G. Since H has few edges (close to linear), it is easy to find its inverse. Then, we can precondition L_G with L_H. We will later have a small discussion on the third property, how to find H fast. Then, we can use L_H to precondition L_G

In this section, we will show the existence of almost linear sized sparsifiers. Remember, a graph can have $O(n^2)$ edges in worst case.

A very simple idea to sparsify a graph is to sample $O(n)$ edges from some probability distribution. It is not at all clear that such a graph will be a good approximation of the original graph.

In many cases, Chernoff bounds are used to prove the effectiveness of sampling.

Theorem 1 (Chernoff bound). *Given $X_1, X_2, \cdots, X_n$, n independent binary random variables, define $X = \sum_i X_i$. Then,*

$$Pr(X \geq (1 + \delta)\mathbb{E}[X]) \leq e^{O(-\delta^2 \mathbb{E}[X])}.$$

Simply, Chernoff bounds states that the behavior of sum of *independent* random variables is close to their expectation.

For our case, our random variables are matrices. Every edge sampled can be represented by its Laplacian and we would like to get close to the complete Laplacian L_G. Similar results (to Chernoff bounds) have been proved in the case of matrices too and are called matrix Chernoff bounds. We will use the version by Tropp presented in Dan Spielman's notes.

Theorem 2 (Tropp). *Suppose $X_1, X_2, \cdots, X_n$, n independent positive definite matrix valued random variables with $\|X_i\| \leq R$. Define the random variable $X = \sum_i X_i$ and say that μ_1 and μ_n are the smallest and largest eigenvalues of $\mathbb{E}[X]$. Then,*

$$Pr(\lambda_n(X) \geq (1+\delta)\mu_n) \leq ne^{O(-\delta^2\mu_n/R)},$$

and

$$Pr(\lambda_1(X) \leq (1-\delta)\mu_1) \leq ne^{O(-\delta^2\mu_n/R)},$$

Where $\lambda_n(M)$ ($\lambda_1(M)$) is the largest (smallest) eigenvalue of M.

So, we need to find suitable probability distribution on edges, s.t., their Laplacian's expectation is close to L_G and expected number of edges sampled are small.

Exercise 12. Why is it enough to prove that the expected number of edges sampled is small?

One such measure turns out to be *effective resistance* of the edges.

4.1 Effective resistance

For the rest of this lecture, we will consider weighted graphs. So, there will be a weight $w_{i,j}$ associated with every edge i, j. The Laplacian for such a graph is,

$$L_G = \sum_{(i,j)\in E} w_{i,j} L_{i,j}.$$

Where $L_{i,j}$ is the Laplacian of an edge (i, j) discussed before. We look at the action of L only on the positive eigenspace (where it is positive definite and has full rank). On that space, we can define its inverse L^{-1}, s.t., $LL^{-1} = I$. Here, I is only over the positive eigenspace.

Intuitively, if the weight is large, we should think of this edge being strongly present, otherwise it is weakly present. This intuition will help us in providing physical intuition to our strategy.

A weighted graph can also be considered as an *electrical network*, with edges having resistance $r_{i,j} = 1/w_{i,j}$. If there is more weight then edge is strongly connected and hence resistance should be low and vice versa.

The effective resistance between an edge i, j is the voltage difference between node i and node j, when 1 unit of current is put at node i and taken out at node j. Simply put, we consider the entire network as one big resistor and apply Ohm's law to get the resistance between i and j.

Note 4. In this case, the current between node i and j need not be 1 unit.

It is good to have this intuition, but is not required for our sparsification strategy. We will assume, without proof, that the effective resistance between i, j is,

$$Re_{i,j} = (e_i - e_j)^T L^{-1} (e_i - e_j).$$

Note 5. For proof of this and other interesting properties of effective resistance, please look at Dan Spielman's course notes.

We are going to use effective resistance to sample edges. Intuitively, effective resistance captures the importance of an edge in keeping the graph connected. More formally, it is known that effective resistance of an edge is the probability that the edge is present in a random spanning tree. So, effective resistance seems like a good measure to decide the fate of an edge.

4.2 Sampling using effective resistance

Define $p_{i,j} = C \log(n)\epsilon^{-2} w_{i,j} Re_{i,j}$, where C is some constant. We will assume that $p_{i,j}$ is less than 1, if not, the edge can be broken down into many parts with weight divided equally between them. The reason for choosing this probability will be clear with the proof. At this point, notice that the probability is proportional to the effective resistance.

The sampling procedure is pretty simple; we pick an edge i, j with probability $p_{i,j}$ and give it a weight $w_{i,j}/p_{i,j}$.

Suppose we get a random matrix H. We need to show:

1. G is an ϵ approximation of H with high probability.
2. The expected number of edges in H are $O(n \log n)$. This will imply that the number of edges in H are $O(n \log n)$ with high probability.

For the first part, we need to show,

$$(1-\epsilon)L_G \preceq L_H \preceq (1+\epsilon)L_G.$$

It is equivalent to showing,

$$(1-\epsilon)I \preceq L_G^{-1/2} L_H L_G^{-1/2} \preceq (1+\epsilon)I.$$

Which is same as showing that the eigenvalues of $L_G^{-1/2} L_H L_G^{-1/2}$ are between $1-\epsilon$ and $1+\epsilon$. We will use Thm. 2 to show this fact.

From the sampling procedure, L_H is the sum of random matrices $X_{i,j}$ where $X_{i,j}$ is $(w_{i,j}/p_{i,j})L_{i,j}$ with probability $p_{i,j}$ and 0 otherwise. This implies, $L_G^{-1/2} L_H L_G^{-1/2}$ is the sum of random matrices $X'_{i,j}$ where $X'_{i,j}$ is $(w_{i,j}/p_{i,j})L_G^{-1/2} L_{i,j} L_G^{-1/2}$ with probability $p_{i,j}$ and 0 otherwise.

Let us denote an edge i, j by e for simplicity of notation. So, the expectation of $L = L_G^{-1/2} L_H L_G^{-1/2}$ is,

$$\begin{aligned}
\mathbb{E}[L] &= \sum_{e \in E} \mathbb{E}[X'_e] \\
&= \sum_e L_G^{-1/2} \mathbb{E}[X_e] L_G^{-1/2}] \\
&= L_G^{-1/2} \left(\sum_e p_e \frac{w_e}{p_e} L_e \right) L_G^{-1/2} \; L_G \text{ is fixed} \\
&= L_G^{-1/2} \left(\sum_e w_e L_e \right) L_G^{-1/2} \\
&= I
\end{aligned}$$

For applying Thm. 2, we need to bound $\left\| X'_{i,j} \right\|$.

Exercise 13. Show that $L_{i,j} = (e_i - e_j)(e_i - e_j)^T$.

Using the exercise,

$$\left\| X'_{i,j} \right\| = \frac{w_{i,j}}{p_{i,j}} \left\| L_G^{-1/2} (e_i - e_j)(e_i - e_j)^T L_G^{-1/2} \right\|.$$

The quantity inside the norm is a rank one positive semidefinite matrix and hence norm is equal to trace (you will prove this statement in the assignment). So,

$$\left\| X'_{i,j} \right\| = \frac{w_{i,j}}{p_{i,j}} Tr(L_G^{-1/2} (e_i - e_j)(e_i - e_j)^T L_G^{-1/2}).$$

We can use the cyclicity of trace,

$$\left\|X'_{i,j}\right\| = \frac{w_{i,j}}{p_{i,j}} Re_{i,j} = \frac{\epsilon^2}{C \log n}.$$

We have the expectation of the random variable and the norm of the individual random variable.

Exercise 14. Using Thm. 2, show that the eigenvalues of $L_G^{-1/2} L_H L_G^{-1/2}$ are between $1-\epsilon$ and $1+\epsilon$.

This proves that the matrix H is close to G. We only need to show that the expected number of edges in our sampling procedure is $O(n \log n)$.

The idea for the proof has been given before. If we choose an edge i, j with probability $p_{i,j}$, then the expected number of edges is $\sum_{(i,j)\in E} p_{i,j}$. Since effective resistance is the probability of choosing an edge in a random spanning tree, we would expect $Re_{i,j}$ to sum to $O(n)$. This will imply that $p_{i,j}$ will sum to $O(n \log n)$.

Let us prove this formally. Remember that $p_{i,j} = C \log(n) \epsilon^{-2} w_{i,j} Re_{i,j}$. Let us define $C' = C \log n \epsilon^{-2}$. Then,

$$\sum_{(i,j)\in E} p_{i,j} = C' \sum_{(i,j)\in E} w_{i,j} Re_{i,j}.$$

We know that, $Re_{i,j} = (e_i - e_j)^T L_G^{-1} (e_i - e_j)$. Substituting,

$$\begin{aligned}
\sum_{(i,j)\in E} p_{i,j} &= C' \sum_{(i,j)\in E} w_{i,j} (e_i - e_j)^T L_G^{-1} (e_i - e_j) \\
&= C' \sum_{(i,j)\in E} Tr(w_{i,j} (e_i - e_j)^T L_G^{-1} (e_i - e_j)) \\
&= C' \sum_{(i,j)\in E} Tr((w_{i,j} (e_i - e_j)(e_i - e_j)^T) L_G^{-1}) \\
&= C' \sum_{(i,j)\in E} Tr(L_G L_G^{-1}).
\end{aligned}$$

Exercise 15. Prove that $Tr(L_G L_G^{-1}) = n - 1$.

Substituting the value of C' back, $\sum_{(i,j)\in E} p_{i,j} = O(n \log n)$. So, the expected number of edges are $O(n \log n)$ as desired.

We learnt how to sparsify and potentially use this sparsification to solve $Lx = b$. Though, to calculate the sparsification, we need effective resistance and that needs L^{-1}. It seems we have gone into a loop. But, there are methods to calculate the effective resistance approximately, without finding L_G^{-1}. These approximate effective resistances are good enough to sparsify a graph.

We will not cover those techniques here. The article, "A fast solver for a class of linear systems", by Koutis, Miller and Peng is an excellent survey on these techniques.

5 Assignment

Exercise 16. Suppose $\{\lambda_i\}_{i=1}^n$ are eigenvalues of A. Show that $\{\lambda_i p(\lambda_i) - 1\}_{i=1}^n$ are eigenvalues of $Ap(A) - I$.

Exercise 17. Read about Chebyshev polynomials, their recurrence and their properties.

Exercise 18. Let A be ϵ approximated by B. Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be eigenvalues of A and $mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$ be eigenvalues of B. Using Courant-Fischer theorem, prove that for all i,

$$(1-\epsilon)\mu_i \leq \lambda_i \leq (1+\epsilon)\mu_i.$$

Exercise 19. Let $x_t = B^{-1}b + (I - B^{-1}A)x_{t-1}$ be the t-th iteration solution of Richardson's method with preconditioning. Show that $\|x - x_t\|_A \leq \epsilon^t \|x\|_A$.

Exercise 20. Prove that for a rank one positive semidefinite matrix, trace is equal to its norm.

Exercise 21. Suppose, for the purpose of sparsification, we divide an edge into multiple parts with weight divided equally between them. Show that the approximation analysis and expected number of edges does not change.