Messy Notes for Applied

Kexing Ying

May 15, 2020

Contents

1 Circuits

In this section we will discuss the method of using matrices in solving the potiential and conductance problem regarding circuits.

I will assume you can manipulate matrices and use basic linear algebra, I will also assume you can construct the incidence matrix of a graph.

1.1 Null-Spaces

Consider an incidence matrix A representing a circuit.

Then (if we let Φ be the vector representing the potiential at each node) we have $A\Phi = w$ is the potiential difference across each edge.

With this, we see that there is the trivial right null-vector $\Phi_0 = (\mathbf{1})$, as if each node has the same potiential, the the potiential difference across each edge is zero. In fact, this is the **only** linearly independent right null-vector of A (given that the circuit is connected) since if there is two nodes that have different potiential, then of course there is going to be some edge with some potiential difference.

On the other hand, for $-A^T w = f$ where w is the potiential difference across each node, we have f being the vector representing the *sum* of the potientials out of each node.

Then the dimension of the left null-space of A is represented by the number of "loops" of the circuit and we can find the basis of the left null-space by the following method:

1. Let $w_i \in \mathbb{R}^n$ be the vector representing the path of a loop where n is the number of edges of the circuit.
2. For each edge the path traverses, put a 1 in that position if the path follows the directed graph and a -1 if it goes against it.
3. Fill in the rest with 0s.

Thus, by rank-nullity, we can easily verify our previous statement that the trivial right-null vector is the only right-null vector.

1.2 Connected Graphs

The question is what happens when we have two disconnected graphs representing a single graph?

Well, with two disconnected graphs, we can see there are two trivial right-null vectors, i.e. $\Phi_0 = (\mathbf{1}, \mathbf{0})$ and $\Phi_0' = (\mathbf{0}, \mathbf{1})$. We can easily see that our original trivial null-vector is in the span of these two new null-vectors.

1.3 Useful Terms

- A is the *incidence matrix* of the graph
- $K = A^T A$ is the *Laplacian matrix* of the graph
- $C = \text{diag}(c_1, \dots c_n)$ the *conductance* matrix of the graph

- $K_c = A^TCA$ is the *weighted Laplacian matrix* of the graph
- $-A^Tw$ is called the *divergence* of the currents (at the nodes)
- C_e is the *effective conductance* of a circuit

1.4 KCL Cannot Hold at All nodes!

Consider the result from the *fundamental theorem of linear algebra*:

- The null space of A^T is orthogonal to the row space of A, i.e. $\ker A^T \perp \text{csp}A$.

(This is true because if $v \in \ker A$, then the dot product of v and the rows of A must be zero, so it must be perpendicular to the row vectors of A. But for all $w \in \text{rsp}A$, w is a linear combination of the row vectors of A, so $v \cdot w = 0$ and hence, as $\text{rsp}A = \text{csp}A^T$, we have $\ker A \perp \text{csp}A^T$.)

With that, let use suppose *KCL* holds at all the nodes. Then, we have $\mathbf{0} = A^Tw$ for some w representing the potiential difference across each edge. Then, by definition, w is in the null-space of A^T. But this, by our aforementioned lemma must be perpendicular to any vectors in $\text{csp}A = \text{Im}A$ and hence, if $w \neq 0$ (which we are assuming as otherwise we would have a circuit not doing anything!) there does not exist some potiential vector Φ such that $w = A\Phi$.

Physically, this means that under the assumption that *KCL* holds at all nodes, there will be **no** non-trivial potiential that will produce a current through our circuit (but of course the trivial null-vectors of A satisfy our equation $\mathbf{0} = A^TA\Phi$).

1.5 Method for Finding the Potientials

Just solve

$$f = A^TCA\Phi = K_c\Phi$$

and we are done!

Okay, perhaps this is not very helpful so to expand on what I've said above, let us consider two types of questions.

1.5.1 Specified Conductance

First, perhaps we are given the net currents at the boundary nodes, i.e. f is known and we are asked to find Φ such that $f = K_c\Phi$. Then, you might just think to do something like $\Phi = K_c^{-1}f$ to determine Φ. But this is no good as K_c is singular and therefore not invertible (recall that we have the trivial null-vector for A, Φ_0 so $K_c\Phi_0 = A^TC(A\Phi_0) = A^TC\mathbf{0} = \mathbf{0}$). So what we do is we set one of the nodes to have potiential zero, i.e. we will have that node as our reference potiential (if you think about this physically what I'm saying is that the potiential at one node doesn't really mean much - its the potiential difference that actually does the work so fixing one node potiential doens't effect what we are really after - the potiential difference).

So what we do have now is some mathematical trickery. If we fix the i-th nodes to have potiential zero, then when we multiply K_c with this potiential vector, you can see that the i-th row and collumn of K_c nolonger contribute to the multiplication so by removing them, we receive a new linear system $K'_c\Phi' = f'$. Now, by repeating this process until its solvable, we should receive some vector $\bar{x}$ that is satisfied by our original system and hence we have a solution space

$$S = \bar{x} + \ker A := \{\bar{x} + x_0 \mid x_0 \in \ker A\}.$$

1.5.2 Specified Potientials

So now what if we are given the potientials at the boundary nodes and are asked to find the potientials at the non-boundary nodes while also finding the net-currents out of the boundary nodes?

Well, thats a lot asked from us, but I think we can manage. So in this case we have $K_c\Phi = f$ where $\Phi = (x_1, x_2, x_3, \dots)^T$ and $f = (f_1, -f_1, 0, \dots)^T$ where x_1, x_2 are given and we are asked to find the rest (if it's not in this form then just rewrite it in this form). And then all we have to do is to use the Schur complement and Wikipedia can explain the rest.

One more comment. The effective conductance of the system $C_e = f_1$.

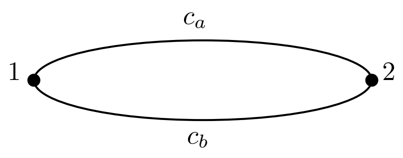
1.6 Calculating the Effective Conductance

We will now present some trickery in calculating the effective conductance of a circuit by simplifying the circuit whenever we see edges in series or parallel.

1.6.1 Edges in Parallel

I'm going to talk about parallel first because it is nicer to work out mathematically.

Suppose we have the following graph represting a particular section of another graph with parallel edges.



Let us write out the incidence matrix and the conductance matrix,

$$A = \begin{pmatrix} -1 & 1 \\ -1 & 1 \end{pmatrix}, \quad C = \begin{pmatrix} c_a & 0 \\ 0 & c_b \end{pmatrix}.$$

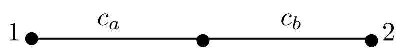
Then, if let the system have unit potential at node 1 and zero potential at node 2, we have

$$f = A^T C A \Phi = \begin{pmatrix} -1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} c_a & 0 \\ 0 & c_b \end{pmatrix} \begin{pmatrix} -1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} c_a + c_b \\ -(c_a + c_b) \end{pmatrix}$$

implying that for edges in parallel, we shall add the conductance.

1.6.2 Edges in Series

It's a very similar story for edges in series but the matrices doesn't multiply as nicely so we will omit it the proof but the result is that given the below graph,



we can treat the system as one edge with conductance $c_a c_b/(c_a + c_b)$.

2 Random Walks

In this section, we shall examine another system - random walks on a graph and explore its similarity to a circuit system. With this, we will conclude that in fact the two systems are the same and are said to be *harmonic* and for each method of solving the specific problem, an equivalent method can be used for the other.

We shall also consider two additional methods for solving problems involving harmonic graphs - *algebraic* and the *method of relaxation*.

2.1 A Stroll Through Central London

Consider the much reduced (an worse) tubemap of central London,

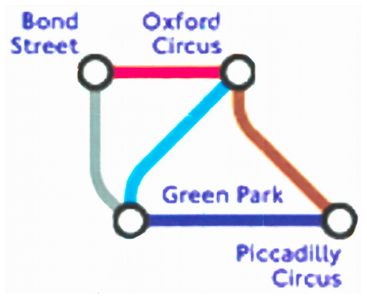


Figure 1: A fictional tubemap of central London.

and suppose, your very drunk friend, Xin, gotten himself wasted at *Station A*. Wanting to get home, Xin takes a random tube at each station and gets off at the next rinse and repeat until he reaches some station we will specify.

We would now like to pose three questions:

1. What is the probability that, if Xin starts at *Oxford Circus*, he reaches *Bond Street* before reaching *Green Park*?
2. What is the probability if Xin starts at *Piccadilly Circus* instead?
3. What is the probability that if Xin starts as *Bond Steet* and leaves, he reaches *Green Park* rather than returning to *Bond Street*?

2.1.1 An Recursive Algorithm for Question 1

Often times, it is helpful to first simulate our problem with a computer programme so we can see perhaps there is a pattern to our answer. To achieve this however, there are many different methods. Here I have prepared a pseudocode representation for a recursive implementation of simulating one random walk however there are different ways of achieving the same without recursions.

Algorithm 1: An recursive method of simulating random walks

```
Class stations(name, connected):
    let self.name = name;
    let self.connected = connected;
Function walkNext(currentStation):
    if currentStation $\neq$ bondStreet $\vee$ greenPark then
        walkNext(choose from currentStation.connected);
    else
        return currentStation
    end
```

By making each station a data structure containing its name and its adjacent stations, we can recursively choose the next station until we reach Bond Street or Green Park.

Algorithm 2: Finding the probability given our method of simulating one random walk

```
let reachedBondStreet = 0;
while runs $\leq$ n do
    if walkNext(oxfordCircus) = bondStreet then
        bondStreet += 1;
    else
        Pass
    end
    runs += 1;
end
let probability = reachedBondStreet/n
```

Thus, by doing sufficiently large number of such random walks, we can compare the ratio of the number times Xin reached Bond Street against the total number of runs.

By implementing this algorithm in an actual programming language, it seems that this probability appears to be around 2/5 which is exactly the same value if we attempted to compute this as if this was a circuit with unit conductances on each edge.

This is surprising as on the surface, neither systems seemed to have anything to do with each other, but however, later we will find out that both systems are *harmonic* and therefore are essentially the same.

2.2 An Algebraic Method

As promised, I will now present an algebraic method in solving the random walker problem.

Let use first rephrase the question so we can refer to it easier. Consider the graph drawn below. We can easily see that the graph and the tubemap are infact the same with Bond Street being node 1, Green Park as node 2 and so on.

With that, we can rephrase the question such that we are asked to find p_3 where p_i represents the probability of reaching node 1 before reaching node 4.

Consider the *law of total probability*:

- Given a event $E \in \mathcal{F}$ and $B = \{B_i | i \in \mathcal{I}\} \subseteq \mathcal{F}$ for some index set $\mathcal{I}$, if B forms a partition of the sample space Ω, then

$$P(E) = \sum_{i \in \mathcal{I}} P(E|B_i)P(B_i).$$

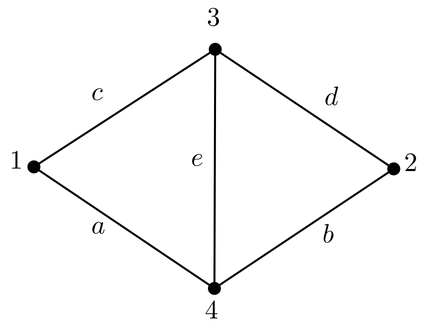


Figure 2: A graph representation of the tubemap.

Then, starting at node 3, we see that Xin can either go to node 1, 2 or 4, so the events of Xin going to node 1, node 2 and goint to node 4 partitions the sample space. Therefore by the law of total probability, we have

$$p_3 = \frac{1}{3} \times p_1 + \frac{1}{3} \times p_2 + \frac{1}{3} \times p_4.$$

Similarily, we see that going to node 3 or node 4 from node 2 partitions the sample space, so

$$p_2 = \frac{1}{2} \times p_3 + \frac{1}{2} \times p_4.$$

Now, if we started from node 4, then there is of course no change of reaching node 1 before node 4 (as you are aleady at node 4!) so $p_4 = 0$. Similarily if we started from node 1, then the probability of reaching 1 before 4 is 1, so $p_1 = 1$.

Thus, by plugging in $p_1 = 1$ and $p_4 = 0$, we have a system of equations with two unknowns,

$$\begin{aligned} p_3 &= \frac{1}{3} + \frac{1}{3} \times p_2 \\ p_2 &= \frac{1}{2} \times p_3 \end{aligned}$$

which is easily solvable resulting in $p_3 = 2/5$ and $p_2 = 1/5$ would you imagine it! It's exactly what we expected from the simulated solution (and the same solution as the matrix method).

2.3 Connection Between the Two Systems

Recall our equation for circuits, $f = K\Phi$ where in which at the non-boundary nodes, we had the enforcement of *KCL*. So suppose ϕ_i is the potiential at a non-boundary node. As *KCL* is upheld at this node, we have the i-th row of $K\Phi$ being zero. So by denoting $\deg(i)$ by the number of edges connected to node i, we have

$$\deg(i)x_i - \sum_{\substack{j\in\text{connected}\\ \text{nodes}}} x_j = 0,$$

so by rearranging, we have,

$$x_i = \sum_{\substack{j\in\text{connected}\\ \text{nodes}}} x_j / \deg(i), \tag{1}$$

which is exactly what we have done for the algebraic method for random walks.

2.3.1 Intuitive Interpretation

Intuitively we can interpret both system as graphs with some variables associated with the nodes. Furthermore, as demonstrated by the above equation, we see that in both system, the node variable of each node is the **average** of the its neighbouring node variables. In the case of circuits, the potientials (the node variable) of each node is the average of its neighbours (assuming uniform conductance) and similarly in the case of the random walk, the probability (the node variable) is the average of its neighbours (assuming each path is equally likely).

With that, we can abstract graph systems such as these in which each nodes of the graph is associated with a node variable that is the average of its neighbouring nodes and we will call such systems *harmonic*.

And thus, as these systems are essentially the same in our framework, we can utilize a method of one system interchangably with another.

2.3.2 Weighted Harmonic Graphs

However, what if our conductances/ probabilites of going to each neighbouring stations are not the same? Then we shall call the harmonic system *weighted*.

As an extention of the regular harmonic system, we see that we do not need to change our original framework much but to weight each node variable according to the conductances/ probabilites or whatever other system we are using resulting in

$$x_i = \sum_{\substack{j\in\text{connected}\\ \text{nodes}}} (x_j / c_j), \tag{2}$$

where c_j denotes the weight we have assigned to the edge connecting the i-th node and the j-th node.

2.4 Method of Relaxation

With our interpretation in mind, we can now construct another algorithm that converges to our desired node variables by averaging the each node variables with the adjacent ones (with weighing dependent one the edge variables if the edge variables are not uniform) over and over agains for sufficiently large number of times.

Below are some pseudocode describing this method.

We first create a object called `Node` containing information about its neighbours and it's initial potiential and we will let a `graph` to be a collection of `node` : `Node`.

Algorithm 3: Constructing the object of nodes and creating a graph as a set of nodes

```
Class Node(name, nodeVariable, connected, weighing):
    let self.name = name;
    let self.connected = connected;
    let $self.weighing_i = weighing_i$;
    if declared nodeVariable then
        let self.nodeVariable = nodeVariable;
        let self.boundary = True
    else
        let self.nodeVariable = 0;
        let self.boundary = False
    end
let $graph = \{node_1, node_2, \cdots, node_k\}$;
```

Then by looping over all the non-boundary nodes for sufficient amount of times the node variables of each node converges to the true value.

Algorithm 4: Constructing the object of nodes and creating a graph as a set of nodes

```
while $runs \leq n$ do
    for $node \in graph \land node.boundary = False$ do
        node.nodeVariable = $\sum_{i \in node.connected} weighing_i \times node_i.nodeVariable$
    end
    runs += 1
end
```

3 Harmonic Graphs

In this section we shall examine some properties of harmonic graphs.

3.1 Maximum Principle

Given a harmonic graph, it is rather natural to seperate its nodes into *boundary nodes* and *internal nodes* (all other nodes). Then the *maximum principle* states the following:

- For a harmonic graph, the node variables attains maximum and minimum at the boundary nodes.

To see why this is true recall that for a (weighted) harmonic graph, each internal node has node variable that is a weighted average of its neighbouring node variables. So if a harmonic graph attains its maximum node variable at a internal node, then all nodes adjacent must be less equal to it. However, as the node variable is the weighted average of the adjacent node variables, if there is one adjacent node variable strictly less than it, the weighted average will be smaller resulting in a contradiction. Therefore, all adjacent nodes must have the same node variable as the maximum. Now repeat this until it reaches a boundary node, resulting in our original statement.

The same argument works for the minimum.

3.2 Uniqueness Principle

Given two harmonic graphs, the uniqueness principle states that if the graphs have the same boundary conditions (and the same edge variables if it is not uniform), then they also have the same node variables everywhere.

This is true because if we let x_i and p_i denote the i-th node variable of the two graphs, then the graph with node variables defined by $d_i = x_i - p_i$ attains maximum and minimum at one of its boundary nodes. But at the boundary nodes, $x_i = p_i$ implying $\max_j d_j = \min_j dj = 0$ and hence has node variable zero everywhere.

This is a very important principle as this ensures that our solution for the random walk problem is the same as the circuit and is the same for any other harmonic graphs for that matter!

3.3 Dirichlet's Principle

The energy dissipation associated with any circuit with two boundary nodes with potiential 1 and 0 is defined to be

$$\xi(\Phi) = \Phi^T K \Phi$$

where K is the (weighted) Laplacian and Φ is any potiential vector.

Dirichlet's principle states that the potential vector Φ_* satisfying KCL at all internal nodes minimizes the energy dissipation, i.e.

$$\xi(\Phi_*) = \min_{\Phi} \xi(\Phi).$$

3.4 Thomson's Principle

By mathematical manipulation the energy dissipation from a cirucit can also be written as

$$\begin{aligned}\xi(\Phi) = \Phi^T K \Phi &= \Phi^T A^T C A \Phi \\ &= (-A\Phi)^T(-CA\Phi) \\ &= \sum_{k \in \text{edges}} w_k \left(\frac{w_k}{c_k}\right) \\ &= \hat{\xi}(w),\end{aligned}$$

where w_k is the current in the k-th edge and c_k is the conductance in the k-th edge.

Notice that this new expression of energy dissipation does not have potiential as it's variable but the edge currents. So as a corollary of Dirichlet's principle, the current vector such that all the internal nodes satisfies *KCL* also minimizes energy dissipation.

3.5 Tellegen's Theorem

Given a circuit with incidence matrix A and any vector w satisfying *KCL* at all nodes (w does not need to equal some $A\Phi$ for some potiential vector Φ), then for any e a potiential difference vector, Tellegen's theorem states,

$$e^T w = \mathbf{0}.$$

Proof. Since e is some vector of potiential difference, then there exist some $\Phi \in \mathbb{R}^n$, a potiential vector, such that $e = -A\Phi$. Thus

$$\begin{aligned} e^T w = (-A\Phi)^T w &= -\Phi^T A^T w \\ &= -\Phi^T (w^T A)^T \\ &= -\Phi^T \mathbf{0} = \mathbf{0} \end{aligned}$$

as required. □

Now suppose that w is some vector that satisfies $-A^T w = f$ then for all e a potiential difference vector such that $e = -A\Phi$, we have

$$e^T w = x^T f.$$

Proof.

$$\begin{aligned} e^T w = (-A\Phi)^T w &= -\Phi^T A^T w \\ &= -\Phi^T (-(-A^T w)) \\ &= \Phi^T f. \end{aligned}$$

□

Corollary 0.1. *Suppose we have two boundary nodes with potiential 1 and 0. Let w be the current in the cirucit and let e be the potiential difference of the cirucit, i.e. $e = -A\Phi$ where Φ is the potiential vector of the circuit. Then the effective conductance is the energy dissipation.*

Proof. By considering Tellegen's theorem and the fact that other than the two boundary nodes, all other nodes satisfy *KCL*, we have

$$e^T w = x^T f = x_1 C_e + x_0(-C_e) = C_e.$$

But we also have,

$$\begin{aligned} e^T w &= (-A\Phi)^T (-CA\Phi) \\ &= \Phi^T A^T C A \Phi \\ &= \Phi^T K \Phi = \xi(\Phi). \end{aligned}$$

So therefore $C_e = \xi(\Phi)$ as required. □