# A Beginner's Guide to Physical Networks: Boulder School for Condensed Matter 2026

Aaron Winn
*Department of Physics and Astronomy, University of Pennsylvania, Philadelphia, Pennsylvania 19104, USA*

Eleni Katifori
*Department of Physics and Astronomy, University of Pennsylvania, Philadelphia, Pennsylvania 19104, USA and Center for Computational Biology, Flatiron Institute, New York, NY 10010, USA*

(Dated: July 14, 2026)

Network theory studies entities (nodes) and the relationships between them (edges). While these entities and their relationships may be abstract in general, this tutorial focuses on the special case of **physical networks**: spatially embedded networks that carry physical quantities such as electric charge, fluid volume, or momentum. The underlying structure is described mathematically by a graph (Section 1), with the incidence matrix playing a central role in analysis. Viewing a graph as a special case of a cell complex reveals the topological structure of the incidence matrix and its relation to familiar operators from calculus (Section 2). Because these networks are physical, flows are conserved at each node and often derive from the gradient of a potential. Models satisfying these principles behave analogously to circuits; accordingly, we first review circuit theory (Section 3), then examine related fluid flow (Section 4) and mechanical (Section 5) networks. Analytical examples and Python implementations are included throughout. Exercises are included as footnotes. This tutorial is based on a course first taught at the University of Pennsylvania in Spring 2024, and adapted for the Boulder School for Condensed Matter Physics and should be a useful resource for students and researchers applying graph theory to physical networks. Knowledge of linear algebra is assumed, but no prior knowledge of graph theory or physics is necessary. [1]

## I. GRAPH THEORY

### A. Definitions

A network is represented mathematically as a graph. A **graph** $G$ is a collection of nodes (also known as vertices) $\mathcal{V}$ and edges $\mathcal{E}$. In particular, we will be interested in graphs whose edges have an orientation; such graphs are called **oriented graphs**. Edge orientations establish an arbitrary sign convention for direction, analogous to choosing a coordinate system.[2] As a convenient notation, a node will be identified by a Latin index, $v_i \in \mathcal{V}$. An edge will be identified by a Greek index $e_\mu \in \mathcal{E}$. We may also identify an oriented edge with an ordered pair of nodes: If an edge connects node $v_i$ to node $v_j$, we may write the edge as $e_{ij} \equiv (v_i, v_j)$,[3] in which case $v_i$ is called the tail of edge $e_{ij}$, and $v_j$ is called the head. We will not allow for the possibility of self-loops where an edge leaves and enters the same node. The number of nodes in $\mathcal{V}$ and edges in $\mathcal{E}$ will be denoted $|\mathcal{V}|$ and $|\mathcal{E}|$, respectively.

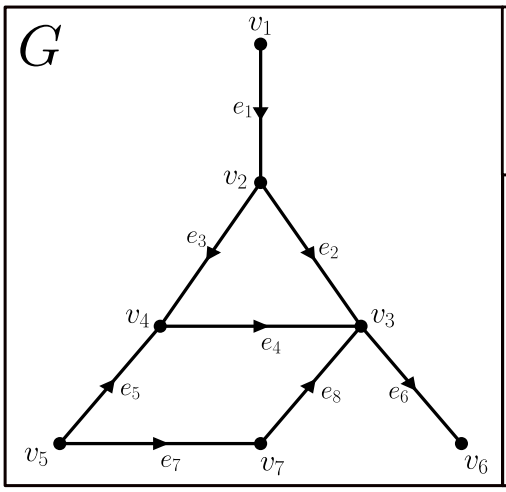
To each abstract node $v_i$ in the set $\mathcal{V}$, we associate a vector $\vec{v}_i$ where $\vec{v}_i$ has a 1 in the $i$-th entry and a 0 in all other $|\mathcal{V}| - 1$ entries. The vectors $\{\vec{v}_i\}$ form a basis for a vector space isomorphic to $\mathbb{R}^{|\mathcal{V}|}$ called the **node space**.[4] Similarly, the vectors $\{\vec{e}_\mu\}$ form a basis for the **edge space** isomorphic to $\mathbb{R}^{|\mathcal{E}|}$. This

[1] A note of caution: the reference list is actually minimal, as we did not intend it to be an exhaustive review of the field: it is virtually certain that we missed some important works, perhaps even your favorite and most educational book or paper; please bring it to our attention.

[2] An oriented graph should not be confused with a **directed graph**. In a directed graph, the edges have a particular (not arbitrary) direction, as will be explained in a later section. As a physical example, an oriented graph can be used to study fluid flow through pipes where the fluid is allowed to flow in either direction. We need a means of specifying when the fluid is transported forward versus backward through the pipe. If the flow is along the edge orientation, we say the flow is positive, and if the flow is against the edge orientation, we say the flow is negative. If, however, the pipes contain valves, then the fluid is only permitted to flow in one direction, so the graph is directed.

[3] This notation becomes problematic for **multigraphs**, graphs with multiple edges between the same two nodes. But, aside from sometimes making the notation cumbersome, the analysis in our review applies perfectly well to multigraphs.

[4] Of course, the node space could instead be defined over complex numbers or any other field, but we will typically denote the node space $\mathbb{R}^{|\mathcal{V}|}$.




$$\mathcal{V} = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$
$$\mathcal{E} = \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_4, v_3), (v_5, v_4), (v_3, v_6), (v_5, v_7), (v_7, v_3)\}$$

$$\hat{\Delta}^T = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \end{array}$$

FIG. 1. Three equivalent representations of the oriented graph $G$ are shown: pictorially, as a set of vertices $\mathcal{V}$ and oriented edges $\mathcal{E}$, and with its incidence matrix $\hat{\Delta}^T$.

vector space structure allows for efficient calculation of physical quantities on graphs using linear algebra. Throughout the paper, vectors in the node or edge space will be distinguish using an arrow (ex: $\vec{\psi} \in \mathbb{R}^{|\mathcal{V}|}$), while spatial vectors will be distinguished using boldface (ex: $\boldsymbol{x} \in \mathbb{R}^3$). Similarly, linear maps between node and edge spaces will be distinguished using a hat (ex: $\hat{\mathcal{L}} : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}|}$) while linear maps between spatial vectors will be distinguished using boldface (ex: $\boldsymbol{\sigma} : \mathbb{R}^3 \to \mathbb{R}^3$).

We can encode the oriented connectivity of the network using the $|\mathcal{V}| \times |\mathcal{E}|$ **incidence matrix**[5] $\hat{\Delta}^T : \mathbb{R}^{|\mathcal{E}|} \to \mathbb{R}^{|\mathcal{V}|}$ given by

$$\Delta^T_{i\nu} = \begin{cases} +1, & \text{if edge } e_\nu \text{ enters node } v_i \\ -1, & \text{if edge } e_\nu \text{ leaves node } v_i \\ 0, & \text{otherwise} \end{cases} . \tag{1}$$

An example of an oriented graph with its incidence matrix is shown in figure 1.

If the graph is weighted, then $G$ is also endowed with a vector of positive **edge weights** $\vec{\kappa} \in \mathbb{R}^{|\mathcal{E}|}$. The notation $\hat{\kappa} = \text{diag}(\vec{\kappa})$ will also be used to denote the $|\mathcal{E}| \times |\mathcal{E}|$ matrix whose diagonal contains the weights of each edge and is otherwise zero. Finally, $\kappa_{ij}$ will be understood as the weight along edge $e_{ij}$. For an unweighted graph, $\kappa_\mu = 1$. For a physical network, the edge weights often correspond to an edge conductance.

The $|\mathcal{V}| \times |\mathcal{V}|$ **adjacency matrix** $\hat{W} : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}|}$ whose elements are

$$W_{ij} = \begin{cases} \kappa_{ij}, & \text{if } (v_i, v_j) \in \mathcal{E} \\ \kappa_{ji}, & \text{if } (v_j, v_i) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

is a symmetric matrix which contains information on the connectivity and weights of the network, but not on the orientation. The **weighted degree** of node $v_i$ is given by summing the weights of the edges adjacent to a node: $d_i = \sum_k W_{ik}$. The $|\mathcal{V}| \times |\mathcal{V}|$ diagonal matrix of weighted degrees is denoted $\hat{D} \equiv \text{diag}(\vec{d})$.

The $|\mathcal{V}| \times |\mathcal{V}|$ **Laplacian matrix** $\hat{\mathcal{L}} : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}|}$ is a symmetric matrix that can be expressed in two different ways[6]:

$$\hat{\mathcal{L}} = \hat{\Delta}^T \hat{\kappa} \hat{\Delta} = \hat{D} - \hat{W}. \tag{3}$$

---

[5] More specifically, this is the **node-edge incidence matrix**. Once we consider more general cell complexes, we will encounter other incidence matrices, but for the case of graphs, the node-edge incidence matrix is the only incidence matrix. So, throughout most of the paper, we will simply refer to the node-edge incidence matrix as *the* incidence matrix.

[6] **EXERCISE**: Prove that these two forms of the Laplacian matrix are equivalent.

From the first representation of the Laplacian, we can see that $\hat{\mathcal{L}}$ is positive semi-definite. However, $\hat{\mathcal{L}}$ always has a zero eigenvalue corresponding to an eigenvector with entries all equal to 1. The number of zero eigenvalues is the number of connected components of the graph, so for a fully connected graph, this is the only zero eigenvalue. The Perron-Frobenius Theorem discussed in the appendix is a useful tool for studying the spectrum of the Laplacian and adjacency matrices. Although $\hat{\mathcal{L}}$ is not invertible, its pseudo-inverse takes a simple form. Note that for a real, symmetric, positive semi-definite matrix, the singular value decomposition is equivalent to the eigendecomposition. If the eigenvectors and eigenvalues of $\hat{\mathcal{L}}$ are $\vec{\xi}_i$ and $\lambda_i$, then we can write $\hat{\mathcal{L}}$ and its pseudo-inverse $\hat{\mathcal{L}}^+$ as

$$\hat{\mathcal{L}} = \sum_i \lambda_i \vec{\xi}_i \vec{\xi}_i^T, \qquad \hat{\mathcal{L}}^+ = \sum_{\{i|\lambda_i \neq 0\}} \lambda_i^{-1} \vec{\xi}_i \vec{\xi}_i^T. \tag{4}$$

The pseudo-inverse will be useful when solving linear systems involving the Laplacian matrix.

### B. Paths, cycles, and trees

Some intuitive terminology is used to describe traversals in a network. These definitions do not require the graph to be oriented. For more definitions and related theorems, see chapter 3 of [1].

A **path** is a sequence of non-repeating edges where any two consecutive edges are connected by a node. Two nodes are **connected** if there exists a path between them; a graph is connected if all nodes are connected. A disconnecting set is any set of edges whose removal increases the number of connected components of the graph. In particular, a **cutset** is a disconnecting set with no proper subset that is a disconnecting set. For example, the set of edges adjacent to a node form a cutset. A **cycle** is a path for which the starting and ending edge are also connected by a node. A **tree** is a graph in which any two nodes are connected by exactly one path. In other words, a tree is a connected graph which contains no cycles. We can use this property of trees to find a convenient basis for the cycles of a graph. A **spanning tree** of a connected graph $G$ is a subgraph which is a tree and which includes all nodes of $G$. Using spanning trees, one can show that the total number of independent cycles in a graph is $|\mathcal{E}| - |\mathcal{V}| + k$, where $k$ is the number of connected components of the graph.[7]

### C. Properties of the incidence matrix

Although it doesn't have a special name, the transpose of the incidence matrix $\hat{\Delta}$ is in some ways more intuitive to work with.[8] Consider a quantity (such as the voltage) with values on each node, that is, a vector $\vec{\psi} \in \mathbb{R}^{|\mathcal{V}|}$. If edge $e_\mu = (v_i, v_j)$, then

$$(\hat{\Delta}\vec{\psi})_\mu = \psi_j - \psi_i \tag{5}$$

measures the voltage drop along edge $e_\mu$. The operator $\hat{\Delta}$ takes the difference of the quantity at the terminal node and the quantity at the starting node. Its image can be thought of as all edge vectors which derive from a difference in node potentials. Note that if $\psi_i = \psi_j$, then $(\hat{\Delta}\vec{\psi})_\mu = 0$. Thus, the vector with all $|\mathcal{V}|$ entries equal to 1 is in the null space of the incidence matrix:

$$\hat{\Delta}\vec{1} = \vec{0}$$
$$\sum_j \Delta_{\mu j} = 0. \tag{6}$$

---

[7] **EXERCISE**: Draw a spanning tree for the graph in figure 1. Show that by adding an edge from $G$ not in the spanning tree, you obtain a graph with one cycle. From this construction, you can form a cycle basis of the graph. Generalize this idea to any connected graph. How many edges are in a spanning tree for a graph with $|\mathcal{V}|$ vertices? How many edges are left in $G$? How many independent cycles must $G$ have? How would the counting change if we had a graph with $k$ connected components?

[8] Warning: We use the symbol $\hat{\Delta}^T$ for the incidence matrix. It's transpose is $\hat{\Delta}$. It's easy to get $\hat{\Delta}$ and $\hat{\Delta}^T$ mixed up, but the easiest way to remember which is which is to remember formula (5) since it is common to use the symbol $\Delta$ to denote a difference in some quantity. Indeed, that is why our group has adopted this notation!

In fact, for a connected graph, this is the only vector in the null space of $\hat{\Delta}$ [2]. To see this, note that if the graph is connected and $\psi_i = \psi_j$ for each $e_{ij} \in \mathcal{E}$, then each $\psi_i$ in the entire network must be the same. For a graph which is not connected, we only require that the $\psi_i$ be the same on each connected component. Therefore, the null space of $\hat{\Delta}$ is spanned by vectors which are equal to 1 on a particular connected component and 0 elsewhere. The nullity of $\hat{\Delta}$ is the number of connected components $k$. By the rank-nullity theorem,[9] the rank of $\hat{\Delta}$ is equal to $|\mathcal{V}| - k$. When a graph is disconnected, $\hat{\Delta}$ can be put in block diagonal form, decoupling whatever problem we wish to solve into $k$ distinct problems.

Now consider an element of the incidence matrix acting on a quantity $\vec{I}$ (for example, the electric current) defined on the edges:

$$(\hat{\Delta}^T \vec{I})_i = \sum_{\mu \to i} I_\mu - \sum_{\nu \leftarrow i} I_\nu. \tag{7}$$

The first sum runs over all edges oriented into node $v_i$ and the second sum runs over all edges oriented out of node $v_i$. The result can be thought of as the net flow into node $v_i$. For many cases of physical interest, this quantity will be zero, corresponding to a conservation law (for example, electric charge).

As is the case with any matrix, the rank of $\hat{\Delta}^T$ is equal to the rank of its transpose, in this case $|\mathcal{V}| - k$. By the rank-nullity theorem, the nullity of $\hat{\Delta}^T$ is $|\mathcal{E}| - (|\mathcal{V}| - k)$. It is no coincidence that this is the same number of cycles we found in the previous section. To see this, it will be useful to recognize the incidence matrix as a boundary operator. The boundary of an edge (or more generally, a path) is the starting and ending node. A cycle is a path which has no boundary. When traversing an edge in a path, the step direction may or may not agree with the orientation of the network. Consider an indicator vector $\vec{e} \in \mathbb{R}^{|\mathcal{E}|}$ whose entries are 1 if a step is taken along the edge orientation, $-1$ if a step is taken against the edge orientation, and 0 if the edge is not in the path. Then, the path is a cycle if and only if $\hat{\Delta}^T \vec{e} = 0$. The cycle space is the null space of $\hat{\Delta}^T$.

It is helpful to think of node quantities as analogous to continuous scalar functions and edge quantities as analogous to continuous vector functions with direction given by the edge orientation. Equation (5) suggests that $\hat{\Delta}$ acts like a discrete derivative (or more precisely, gradient) operator, acting on a node-valued function to give an oriented edge-valued function.[10] Equation (7) suggests that $\hat{\Delta}^T$ acts like the negative of a divergence operator, acting on an edge-valued function to produce a node-valued function.[11] It's quite remarkable that $\hat{\Delta}^T$ acts both as a boundary operator and a derivative operator depending on whether it acts on an indicator vector or a node-valued function. We will explore these ideas further in section II.

Even though our graphs are oriented, it is occasionally useful to consider the **unoriented incidence matrix** which keeps track of which edges and nodes are connected but not the orientation; its entries are the absolute value of the entries of $\hat{\Delta}^T$, so it will be denoted $|\hat{\Delta}|^T$. If $\hat{\Delta}$ acts as a difference operator, then $|\hat{\Delta}|$ acts as a summation operator associating to an edge the sum of its two adjacent node-valued quantities. For example, if the potential $\vec{\psi}$ is defined on the nodes, then the quantity $\frac{1}{2}|\hat{\Delta}|\vec{\psi}$ associates to each edge the average value of the potential on the two adjacent nodes. Similarly, $|\hat{\Delta}|^T$ associates to a node the sum of the adjacent edge-valued quantities. For example, the weighted degree vector can be written $\vec{d} = |\hat{\Delta}|^T \vec{C}$.

### D. Directed graphs

A **directed graph** consists of nodes and directed edges, differing from an oriented graph in that the orientation is not arbitrary, but rather it has some additional physical meaning. In the context of flow networks, a directed edge $e_{ij}$ only permits flow from $v_i$ to $v_j$. If flow is permitted both from $v_i$ to $v_j$ and from $v_j$ to $v_i$, then we will need to include both $e_{ij}$ and $e_{ji}$ in our edge list, though $\kappa_{ij}$ need not equal $\kappa_{ji}$.

---

[9] Recall, the rank-nullity theorem states that for any matrix $A$, the rank (dimension of the image) of $\hat{A}$ plus the nullity (dimension of the kernel, or null space) of $\hat{A}$ is equal to the number of columns of $\hat{A}$.

[10] Unfortunately, in calculus, the symbol $\Delta$ usually refers to the Laplacian. Again, our notation should make you think of $\hat{\Delta}$ as a difference operator, equation (5), not a Laplacian.

[11] Negative because when $(\hat{\Delta}^T \vec{I})_i$ is positive, there is a convergence of flows at node $v_i$.

The adjacency matrix takes a rather intuitive form for a directed graph:

$$W_{ij} = \begin{cases} \kappa_{ij}, & \text{if } (v_i, v_j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}. \tag{8}$$

This encodes the weights across all ordered pairs of vertices, but it will not necessarily be symmetric. We can define two different weighted degrees for each node: the **indegree** $d_{+,i} = \sum_k W_{ki}$ and the **outdegree** $d_{-,i} = \sum_k W_{ik}$.

The incidence matrix can still be defined using (1) with the only caveat being that what we mean by entering versus leaving is not arbitrary anymore; it comes from the direction of the edge. The incidence matrix is not the most fundamental operator needed to describe directed networks. Define the matrices $\hat{\Delta}_+^T$ and $\hat{\Delta}_-^T$ using the following:

$$\Delta_{+,i\nu}^T = \begin{cases} +1, & \text{if edge } e_\nu \text{ enters node } v_i \\ 0, & \text{otherwise} \end{cases}. \tag{9}$$

$$\Delta_{-,i\nu}^T = \begin{cases} +1, & \text{if edge } e_\nu \text{ leaves node } v_i \\ 0, & \text{otherwise} \end{cases}. \tag{10}$$

Clearly, $\hat{\Delta}^T = \hat{\Delta}_+^T - \hat{\Delta}_-^T$, but in some cases, we will only want to consider the incoming or outgoing edges from a node, in which case we'd use $\hat{\Delta}_+^T$ or $\hat{\Delta}_-^T$, respectively.

Certain physics applications require the use of directed graphs (ex: advection of nutrients in a fluid flow), though we will not be working with them in this tutorial. All graphs in this tutorial are undirected, oriented graphs.

### E. NetworkX

The small network examples in this tutorial can be studied by hand, but we recommend reproducing the calculations in Python. The Python package NetworkX allows for the creation, manipulation, and plotting of graphs [3]. For installation instructions and documentation, visit `https://networkx.org/documentation/stable/`. We also recommend importing Numpy, SciPy, and Matplotlib. In this tutorial, we will only review the essentials for getting started using NetworkX to solve physical network problems.

Graphs can be constructed either by using predefined functions or by explicitly adding nodes and edges. Note that NetworkX does not understand the difference between an oriented, undirected graph and a directed graph, so to construct an oriented graph, we actually use `nx.DiGraph()`. In the snippet below, we construct an oriented square graph by explicitly adding nodes and edges.

Listing 1. Drawing a simple graph

```
import networkx as nx
G = nx.DiGraph() #Creates an oriented graph
G.add_node(1, pos=(0,0)) #Adds a node named '1'.
G.add_node(2, pos=(0,1)) #Each node has attribute 'pos' specifying its position.
G.add_node(3, pos=(1,1))
G.add_node(4, pos=(1,0))
G.add_edge(1,2, weight=2) #Adds an edge from node 1 to node 2.
G.add_edge(2,3, weight=4) #Each edge has attribute 'weight' specifying its width.
G.add_edge(3,4, weight=6)
G.add_edge(4,1, weight=10)
positions = nx.get_node_attributes(G,'pos') #Save node positions in a dictionary
weights = [G[i][j]['weight'] for (i, j) in G.edges()] #Save edge weights in a list
nx.draw(G, pos=positions, width=weights, with_labels=True, node_color='orange')
```

Nodes are each given a label when added. It could be anything, but here we use the integers 1 through 4. Edges are added by specifying the starting and ending node label. Nodes and edges can be assigned attributes. Here, we assign the node attribute `pos` for defining the node positions and the edge attribute `weight` for defining the edge weights. Note that `positions` is a dictionary while `weights` is a list.[12]

Once we have defined a graph, the next thing we want to do is obtain its incidence matrix and the Laplacian matrix.

Listing 2. Constructing the incidence matrix and Laplacian matrix

```
import numpy as np
import scipy.sparse as sp
DeltaT = nx.incidence_matrix(G, oriented = True) #Saved as a SciPy sparse matrix
Delta = DeltaT.T
print(DeltaT.toarray()) #Convert to an array only to visualize the matrix.
Conductance = sp.diags(weights)
Laplacian = DeltaT @ Conductance @ Delta
print(Laplacian)
```

By default, `nx.incidence_matrix(G, oriented = True)` is saved as a sparse matrix. It is a good habit to perform matrix computations using sparse matrices since most of the elements of $\hat{\Delta}^T$ and $\hat{\mathcal{L}}$ are zero. For the tiny graphs in this review, it won't matter, but if your network has thousands of nodes, sparse operations will save you time and memory.

[12] **EXERCISE**: Using NetworkX, generate an unweighted, oriented cycle graph with 10 nodes. Study the incidence matrix and its transpose. How are they related to finite difference operators? How is the Laplacian matrix related to the discretized Laplacian operator from calculus? What about when we include edge weights?

## II. DISCRETE CALCULUS

The type of graphs we are considering (oriented graphs with no self loops) are a special case of a more general mathematical object called a cell complex. By understanding the topology of cell complexes, we can better understand the incidence matrix and its extension to higher dimensions. We will also learn how to study calculus and homology of graphs, as well as how to formally construct a dual graph. We hope to not only give some intuition for common mathematical operations on graphs, but also provide an intuitive introduction to algebraic topology. This section is largely based on the book *Discrete Calculus* by Grady and Polimeni [4] and a short review by Desbrun, et al [5]. A more complete resource for the application of topology to physical problems can be found in [6].

### A. Graphs...with faces

A graph is only concerned with 0-dimensional nodes and 1-dimensional edges. Before considering a completely general cell complex, let's first look in just one higher dimension and consider faces. We will denote the set of faces $\mathcal{F}$. Note that a face is a 2-dimensional object, not to be confused with a cycle which is a closed 1-dimensional path. A face will be identified by a capital Latin index $f_I \in \mathcal{F}$. Much like how we had to assign an orientation to the edges by specifying an order of two nodes, we can assign an orientation to a face by specifying an order to at least three edges (ex: clockwise or counterclockwise). For a given face, the orientation also induces an orientation on the edges bounding the face which may or may not match the actual orientation of the edge. We describe how oriented faces are made up of oriented edges using the **edge-face incidence matrix** $\hat{B}^T : \mathbb{R}^{|\mathcal{F}|} \to \mathbb{R}^{|\mathcal{E}|}$ whose elements are given by

$$B^T_{\nu I} = \begin{cases} 0, & \text{if edge } e_\nu \text{ is not on the boundary of face } f_I \\ +1, & \text{if the orientation of edge } e_\nu \text{ is the orientation induced by face } f_I \\ -1, & \text{if the orientation of edge } e_\nu \text{ is the orientation induced by face } f_I \end{cases}. \tag{11}$$

It is often convenient whenever possible to assign the orientation "counterclockwise" to all faces. Such a notion only makes sense when the graph is planar.

Although cycles and faces are distinct, when we want to measure a physical flux through a loop, we will often fill in each independent cycle with a face. Assigning faces to each such cycle in the graph shown in figure 1 gives the graph-with-faces shown in figure 2. The edge-face incidence matrix of that graph-with-faces is calculated. For example, face $f_1$ is composed of edges $e_2$, $e_3$, and $e_4$. The orientations of edges $e_3$ and $e_4$ agree with the orientation of face $f_1$ (both the edges and the face are oriented counterclockwise), but the orientation of edge $e_2$ disagrees with the orientation of face $f_1$ (the edge is oriented clockwise but the face is oriented counterclockwise). Thus, in the $f_1$ column, we put a $-1$ in the $e_2$ row, a 1 in the $e_3$ and $e_4$ rows, and a 0 in all other rows.

Unlike the node-edge incidence matrix $\hat{\Delta}^T$, the columns of the edge-face incidence matrix need not sum to zero. The fundamental difference is that while each edge must be composed of one tail node and one head node, faces are composed of many edges.

We can also interpret $\hat{B}^T$ as a boundary operator mapping a face to its boundary consisting of edges. Analogous to the calculus result that the boundary of a boundary is zero, we might expect

$$\hat{\Delta}^T \hat{B}^T = 0. \tag{12}$$

To see that this is true, note that the image of $\hat{B}^T$ are cycles of the graph, which when acted on by $\hat{\Delta}^T$ produces the zero vector in the space of vertices.[13] The transposed expression is also true:

$$\hat{B}\hat{\Delta} = 0. \tag{13}$$

---

[13] **EXERCISE**: Check that equation (12) is true for the graph-with-faces shown in figure 2.

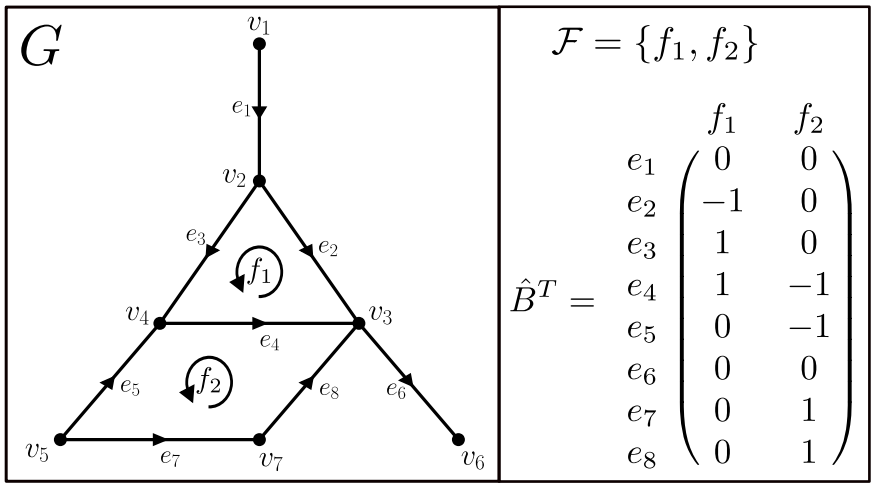



FIG. 2. The oriented graph $G$ from figure 1, now with a set of faces $\mathcal{F}$. The edge-face incidence matrix $\hat{B}^T$ is shown.

To visualize the incidence matrices $\hat{\Delta}^T$ and $\hat{B}^T$ as boundary operators, consider how these matrices act on indicator vectors on the edges and faces. See figure 3.

When acting on face-valued functions and edge-valued functions, $\hat{B}^T$ and $\hat{B}$ act like discrete versions of the curl operator. With this in mind, equations (12) and (13) are the familiar statements $\vec{\nabla} \cdot (\vec{\nabla} \times \vec{F}) = 0$ and $\vec{\nabla} \times (\vec{\nabla} f) = \vec{0}$.

One can also define a face Laplacian matrix $\hat{\mathcal{L}}_2 : \mathbb{R}^{|\mathcal{F}|} \to \mathbb{R}^{|\mathcal{F}|}$

$$\hat{\mathcal{L}}_2 = \hat{B}\hat{\kappa}^{-1}\hat{B}^T \tag{14}$$

analogous to the (node) Laplacian matrix we have already seen. It will become clear later why we are using $\hat{\kappa}^{-1}$ as a weight instead of $\hat{\kappa}$. In many cases, the face Laplacian matrix is invertible.

We now have discrete analogues for common differential operators of vector calculus. Integration on graphs is even more straightforward. If we want to "integrate" a quantity $\vec{\psi}$ defined on the nodes, we need only sum over $\psi_i$ where $v_i$ is in our region of integration. Letting the indicator vector $\vec{v} \in \mathbb{R}^N$ equal 1 for any nodes within the domain of integration and 0 for nodes outside the domain of integration, an integral over nodes is simply the quantity $\langle \vec{\psi}, \vec{v} \rangle = \vec{\psi}^T \vec{v}$. If we pick only a single node $v_i$, then the indicator vector $\vec{v}_i$ is the $i$-th basis vector of the node space, and $\vec{\psi}^T \vec{v}_i = \psi_i$ can be thought of evaluating the scalar function $\vec{\psi}$ on node $v_i$. Integration over edges is similar, only one will need to be more careful keeping track of the direction along which the edge is integrated. To integrate over a path, we define a vector $\vec{e} \in \mathbb{R}^M$ which is equal to $+1$ if the edge is traversed along the edge orientation, $-1$ if the edge is traversed against the edge orientation, and 0 if the edge is not in the path. Then integration over a quantity $\vec{I}$ defined on the edges is $\vec{I}^T \vec{e}$. There's nothing stopping us from assigning other numbers to the components of $\vec{e}$. For example, a 2 would correspond to integrating over the edge twice along the direction of the edge orientation. A summary of the discrete analogues of derivative and integral operators are shown in table I.[14]

In vector calculus, the relationship between derivatives and integrals are summarized in the fundamental theorem of calculus and its relatives. Using the relations in table I, we can write down the discrete versions of these theorems, as summarized in table II. The statements on graphs amount to changing the order

[14] In this section, we are focusing on the topology of graphs, but to fully complete the analogy to calculus, we occasionally need to consider the geometry. The role of the metric tensor is played by the edge weights (and node weights and face weights, though we usually weight all nodes and faces equally). The metric allows us to define a Hodge star operator on complexes and explains why $\hat{\mathcal{L}}$ has a factor of $\hat{\kappa}$ while $\hat{\mathcal{L}}_2$ has a factor of $\hat{\kappa}^{-1}$. We direct the mathematically curious reader to [4]. The form of the node Laplacian and face Laplacian arise naturally when studying physical networks, so we will avoid the unnecessarily technical discussion here.

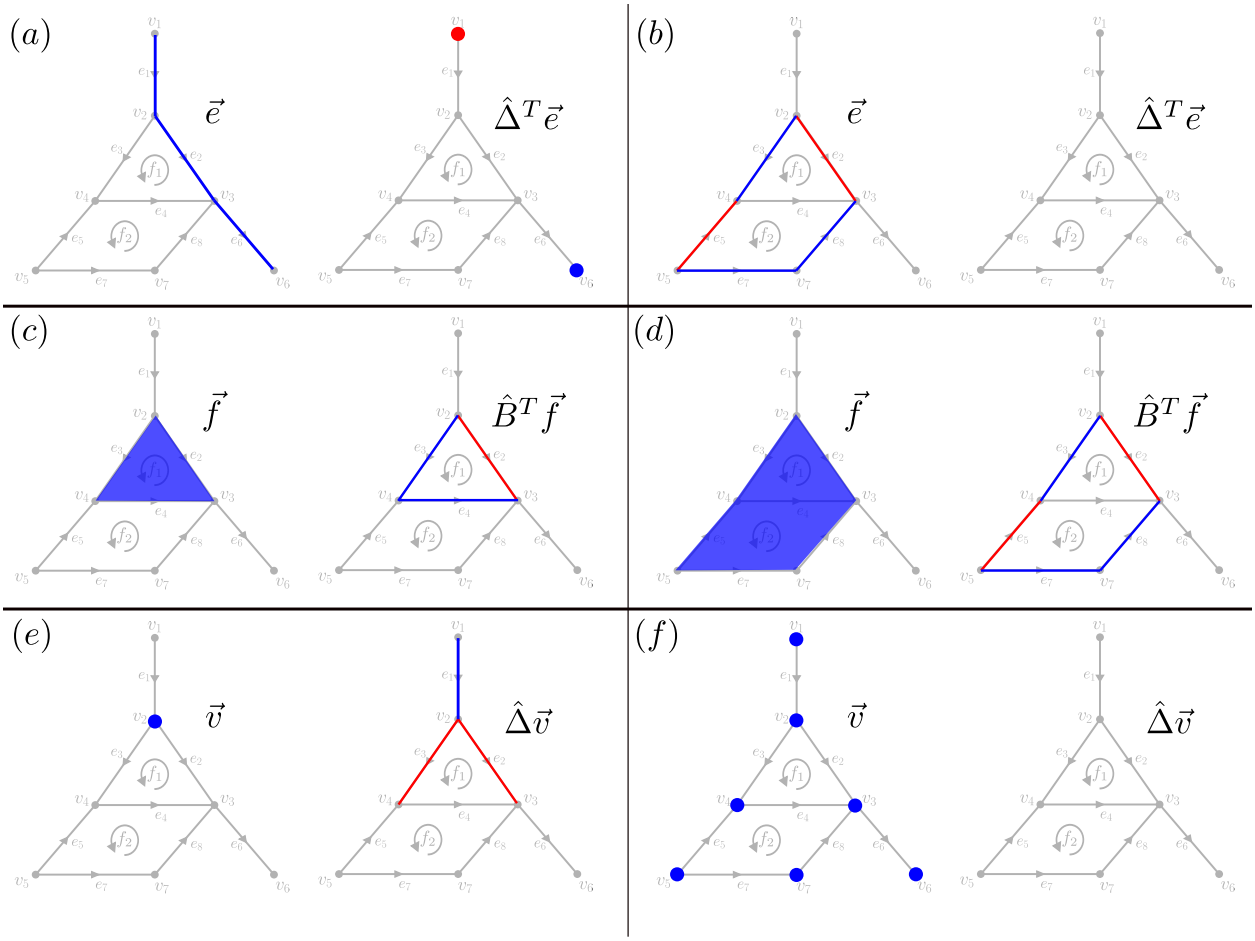



FIG. 3. Examples of the node-edge incidence matrix, edge-face incidence matrix, and the transpose of the node-edge incidence matrix acting on chains. In all cases, rather than a vector representation, a pictoral representation is shown where blue indicates a $+1$ entry and red a $-1$ entry (for example, in $(c)$, $\vec{f} = (1,0)^T$, and $\hat{B}^T\vec{f} = (0,-1,1,1,0,0,0,0)^T$. $(a)$ The boundary of the oriented sequence of edges is one source and one sink. $(b)$ The boundary of a cycle of edges is empty. Note that a red edge in $\vec{e}$ is traversed opposite the edge orientation. $(c)$ The boundary of one face is three edges. $(d)$ The boundary of two adjacent faces is the cycle from $(b)$. $(e)$ The coboundary of a single node describes which edges are entering (blue) or leaving (red) the node. $(f)$ The coboundary of the nodes in the entire graph is empty since each edge enters one node and leaves one node.

of operation of matrix multiplications. When an incidence matrix acts on a physical quantity (think the electric potential $\vec{\psi}$, the voltage $\vec{\Psi}$, and the current $\vec{I}$), it acts like a derivative operator, but when it acts on an indicator vector it acts like a boundary operator. To understand the discrete Fundamental Theorem of Calculus, return to figure 3$(a)$, and suppose we wish to calculate $\langle\hat{\Delta}\psi,\vec{e}\rangle$ for some physical quantity $\vec{\psi}$ defined on the nodes. Physically, this would mean summing up the gradients of $\vec{\psi}$ along the blue edges. By the Fundamental Theorem of Calculus, we can instead calculate $\langle\vec{\psi},\hat{\Delta}^T\vec{e}\rangle$ which amounts to evaluating $\vec{\psi}$ at only two points. When the integral of a gradient is taken over a closed loop, the result is zero regardless of the function. Stokes theorem can be understood in panel $(d)$ as the curl of a function defined on the two faces canceling out in the middle edge, so the curl picks out only the boundary edges. Finally, Gauss's Theorem can be understood as current conservation. For panel $(e)$, if a current exits node $v_2$, it must have entered from the adjacent edges (signed to account for direction with respect to edge orientation). These

| Continuous Operator | Discrete Operator |
|---|---|
| Gradient $\boldsymbol{\nabla}$ | Transposed node-edge incidence matrix $\hat{\Delta} : \mathbb{R}^{\lvert\mathcal{V}\rvert} \to \mathbb{R}^{\lvert\mathcal{E}\rvert}$ |
| Divergence $\boldsymbol{\nabla}\cdot$ | Negative of node-edge incidence matrix $-\hat{\Delta}^T : \mathbb{R}^{\lvert\mathcal{E}\rvert} \to \mathbb{R}^{\lvert\mathcal{V}\rvert}$ |
| Curl 1 $\boldsymbol{\nabla}\times$ | Transposed edge-face incidence matrix $\hat{B} : \mathbb{R}^{\lvert\mathcal{E}\rvert} \to \mathbb{R}^{\lvert\mathcal{F}\rvert}$ |
| Curl 2 $\boldsymbol{\nabla}\times$ | Edge-face incidence matrix $\hat{B}^T : \mathbb{R}^{\lvert\mathcal{F}\rvert} \to \mathbb{R}^{\lvert\mathcal{E}\rvert}$ |
| Laplacian $\nabla^2$ | Negative of Laplacian matrix $-\hat{\mathcal{L}} : \mathbb{R}^{\lvert\mathcal{V}\rvert} \to \mathbb{R}^{\lvert\mathcal{V}\rvert}$ |
| Line integral $\int_l d\boldsymbol{l}\cdot$ | $\langle \cdot\,, \vec{e}\,\rangle : \mathbb{R}^{\lvert\mathcal{E}\rvert} \to \mathbb{R}$ |
| Surface integral $\int_A d\boldsymbol{A}\cdot$ | $\langle \cdot\,, \vec{f}\,\rangle : \mathbb{R}^{\lvert\mathcal{F}\rvert} \to \mathbb{R}$ |
| Volume integral $\int_V dV$ | $\langle \cdot\,, \vec{v}\,\rangle : \mathbb{R}^{\lvert\mathcal{V}\rvert} \to \mathbb{R}$ |

TABLE I. Operators from continuous vector calculus and their discrete analogues.

| Theorem | Continuous | Discrete |
|---|---|---|
| Fundamental Theorem of Calculus | $\int_l \boldsymbol{\nabla} f \cdot d\boldsymbol{l} = f(b) - f(a)$ | $\langle \hat{\Delta}\vec{\psi}\,, \vec{e}\,\rangle = \langle \vec{\psi}\,, \hat{\Delta}^T\vec{e}\,\rangle$ |
| Stokes' Theorem I | $\int_A \boldsymbol{\nabla} \times \boldsymbol{F} \cdot d\boldsymbol{A} = \oint_{\partial A} \boldsymbol{F} \cdot d\boldsymbol{l}$ | $\langle \hat{B}\vec{\Psi}\,, \vec{f}\,\rangle = \langle \vec{\Psi}\,, \hat{B}^T\vec{f}\,\rangle$ |
| Stokes' Theorem II | $\int_A \boldsymbol{\nabla} \times \boldsymbol{F} \cdot d\boldsymbol{A} = \oint_{\partial A} \boldsymbol{F} \cdot d\boldsymbol{l}$ | $\langle \hat{B}^T\vec{i}_{\,\ell}, \vec{e}\,\rangle = \langle \vec{i}_{\,\ell}, \hat{B}\vec{e}\,\rangle$ |
| Gauss's Theorem | $\int_V \boldsymbol{\nabla} \cdot \boldsymbol{F}\, dV = \oint_{\partial V} \boldsymbol{F} \cdot d\boldsymbol{A}$ | $\langle \hat{\Delta}^T\vec{I}, \vec{v}\,\rangle = \langle \vec{I}, \hat{\Delta}\vec{v}\,\rangle$ |

TABLE II. Theorems from calculus and their discrete analogues. Think of $\vec{\psi}$ as the electric potential, $\vec{\Psi}$ as the voltage, $\vec{i}_{\,\ell}$ as the loop current, and $\vec{I}$ as the edge current in an electric circuit.

theorems are closely related to Kirchhoff's Laws in circuits.[15]

There is also a discrete analogue to the Helmholtz Decomposition so that one can decompose an edge-valued function into a gradient term and a curl term. The details are left as an exercise.[16]

---

[15] **EXERCISE**. Show that, for any vector $\vec{I} \in \mathbb{R}^{\lvert\mathcal{E}\rvert}$, if $\vec{v} \in \mathbb{R}^{\lvert\mathcal{V}\rvert}$ is the vector with all node-valued entries equal to 1, then $\langle \hat{\Delta}^T\vec{I}, \vec{v}\,\rangle = 0$. Give a physical interpretation of this equation.

[16] **EXERCISE**: In continuous space, a Helmholtz decomposition can be performed for any sufficiently smooth and rapidly decaying function $\vec{F}$ by writing $\vec{F}$ as the sum of an irrotational and solenoidal piece: $\vec{F} = \vec{F}_{\,irrot} + \vec{F}_{\,sol} = \boldsymbol{\nabla}\phi + \boldsymbol{\nabla} \times \vec{A}$. Notably, the second term is divergence-less. For an edge-valued vector $\vec{I}$, there exists a $\vec{\psi} \in \mathbb{R}^n$ such that one can perform a Helmholtz decomposition with $\vec{I}_{\,irrot} = -\hat{\Delta}\vec{\psi}$ and $\vec{I}_{\,sol} = \hat{B}^T\vec{i}_{\,\ell}$. Show that $\vec{I}_{\,irrot}$ and $\vec{I}_{\,sol}$ are orthogonal and that such a construction is always possible. Then, demonstrate how to calculate $\vec{\psi}$ when $\hat{\Delta}^T\vec{I}$ is known and $\vec{i}_{\,\ell}$ when $\hat{B}\vec{I}$ is known. Is $\vec{\psi}$ unique? [Hint: If you get stuck, read the next section on circuit theory, and return. How does this relate to the node and loop analyses in a circuit with unit conductances?]

## III. CIRCUIT THEORY

Now that we are familiar with oriented graphs, we are ready to study physical networks. In the next few sections, we will study different examples of linear flow networks. The flows on these networks are driven by a linear response from a potential field and satisfy a conservation law at the nodes. Many examples of physical networks fit into this framework, but we will begin by motivating with perhaps the most familiar example: electronic circuits. The connection between circuits and graph theory can be found in the classic engineering textbook by Strang [7].

A **linear flow network**[17] is a weighted graph with edge currents $\vec{I} \in \mathbb{R}^{|\mathcal{E}|}$ and edge voltages $\vec{\Psi} \in \mathbb{R}^{|\mathcal{E}|}$ satisfying the following laws at steady state:

- **Ohm's Law**

$$\vec{I} = \hat{\kappa}\vec{\Psi} \tag{15}$$

- **Kirchhoff's Current Law (KCL)**

$$\hat{\Delta}^T \vec{I} = \vec{i} \tag{16}$$

- **Kirchhoff's Voltage Law (KVL)**

$$\hat{B}\vec{\Psi} = \vec{\psi}_\ell. \tag{17}$$

The edge weights $\vec{\kappa}$ represent the conductances of the edges (how easily the quantities on the edges flow). Each edge is assumed to have a finite conductance. In circuit theory, you could say that $\mathcal{E}$ is the set of resistors. It is most convenient not to consider voltage sources and current sources as living on the edges, but rather, these are quantities that live either on the nodes or the faces, depending on whether a node analysis or a loop analysis is performed. In a node analysis, the vector $\vec{i} \in \mathbb{R}^{|\mathcal{V}|}$ is used to account for excess outflow ($i_j > 0$) or inflow ($i_j < 0$) at each node, and the edge voltage ("voltage drop") is written as the gradient of a node voltage ("electric potential"): $\vec{\Psi} = -\hat{\Delta}\vec{\psi}$. In a loop analysis, the vector $\vec{\psi}_\ell \in \mathbb{R}^{|\mathcal{F}|}$ is used to account for excess voltage accumulated when traversing a closed loop ("EMF"), and the current is written as the curl of a loop current: $\vec{I} = \hat{B}^T\vec{i}_\ell$. We will demonstrate how to solve linear flow networks using both the node analysis and the loop analysis.

Away from steady state, there are additional linear relationships we can include (ex: capacitors and inductors) that involve the time derivatives of the currents and voltages. We will demonstrate techniques for solving dynamic flow networks using the node analysis.

Of course, in contexts other than circuits, $\vec{I}$ and $\vec{\Psi}$ may be called something else, but the terms "Ohm's Law", "KCL", and "KVL" are still often used for any equations of this form on graphs. Ohm's Law encodes the physical linear response of the network. For most networks, the right-hand-side of equation KCL (KVL) is zero at all but a small number of nodes (faces) where boundary conditions are applied. Aside from these special nodes (faces), KCL (KVL) is a purely topological property of the graph.

### A. KCL as a Continuity Equation

It is useful to see how KCL derives from physics in continuous space. This calculation demonstrates a uniting feature of all physical networks. Throughout the paper, boldface will be used to denote vectors in three-dimensional space, while arrows always denote elements in the vector space of nodes or edges.

---

[17] A nonlinear flow network also satisfies three laws, except Ohm's Law is replaced by a nonlinear current-voltage relationship.

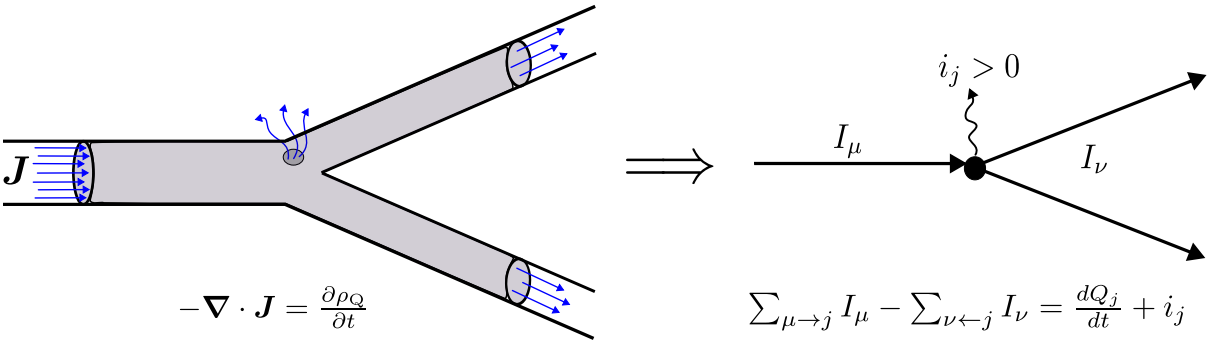



FIG. 4. Visualization of how local charge conservation gives rise to Kirchhoff's Current Law (KCL). A Gaussian surface is drawn around the wires at a junction (node $v_j$). Gauss's theorem converts the integral over the volume to a sum of surface integrals over the wires, thus reproducing KCL.

We begin by identifying a conserved quantity which is being transported in our network. For circuits, this is the electric charge Q.[18] Local charge conservation is expressed in terms of the continuity equation:

$$-\boldsymbol{\nabla} \cdot \boldsymbol{J} = \frac{\partial \rho_{\mathrm{Q}}}{\partial t} \tag{18}$$

where $\boldsymbol{J}$ is the current density, and $\rho_{\mathrm{Q}}$ is the charge density.

We can derive KCL directly from this local statement of charge conservation. We only need to assume that within any edge, the current is spatially uniform. For circuits, this is generally accepted to be the case. The current piercing a cross-section of the wire in edge $e_\mu$ is defined by

$$I_\mu \equiv \pm \int_{S_\mu} \boldsymbol{J} \cdot d\boldsymbol{A}, \tag{19}$$

where the $-$ sign is needed whenever the edge orientation and $d\boldsymbol{A}$ orientation disagree. Now, consider the gray volume $\Omega$ shown in figure 4 enclosing node $v_j$ and bounded by the surface of the wires $S_w$ and the disks $\{S_\mu\}$ which intersect the wires. The current in the wires pierces each of the disks $\{S_\mu\}$, but there may also be some leakage through $S_w$. Let the wire / edge orientation agree with the direction of $\boldsymbol{J}$. Integrating equation (18) and applying Gauss's Law gives:

$$\begin{aligned} -\int_\Omega \boldsymbol{\nabla} \cdot \boldsymbol{J} dV &= -\sum_\mu \int_{S_\mu} \boldsymbol{J} \cdot d\boldsymbol{A} - \int_{S_w} \boldsymbol{J} \cdot d\boldsymbol{A} \\ &= \Big( \sum_{\mu \to j} I_\mu - \sum_{\nu \leftarrow j} I_\nu \Big) - i_j \\ &= (\hat{\Delta}^T \vec{I})_j - i_j, \end{aligned} \tag{20}$$

where we defined the node current

$$i_j \equiv \int_{S_w} \boldsymbol{J} \cdot d\boldsymbol{A} \tag{21}$$

to be the total current leaking out of the walls of the integration region (due to, for example, a current sink). In the second line, we simply applied the definition of current and the assumption that the current is constant along the length of one wire. Note that the orientation of $\boldsymbol{J}$ relative to the boundary normal gives

[18] The transported quantities that appear throughout the paper will be denoted with Roman font. This notation will also be used in the subscript for the associated density.

rise to an orientation of an edge relative to the node, and thus an element of the incidence matrix. Similarly, integrating the right side of the continuity equation gives

$$\int_\Omega \frac{\partial \rho_{\mathrm{Q}}}{\partial t} dV = \frac{d\mathrm{Q}_j}{dt}. \tag{22}$$

The difference between the current entering the node and the current leaving the node can be explained by an accumulation of charge $\mathrm{Q}_j$ at the node. Equating equations (20) and (22) recovers (16), generalized to the non-steady case. Note that we didn't actually apply any physics in this derivation! We only required that there is some conserved scalar quantity being transported. The physics enters the calculation when we say, for example, that the stored charge is related to the node potential by a capacitance.

Given any physical network, we will have a quantity whose continuity equation gives rise to KCL. A similar, though perhaps not as intuitive, calculation can be performed for KVL.[19] Ohm's Law, on the other hand, is entirely a statement about the physics of a linear response. We could derive a discrete Ohm's Law from the continuous Ohm's Law (where $\boldsymbol{J}$ is related to the electric field), but for resistive circuits, the voltage drop occurs not in the wire but only over the discrete resistor element. In this case, (15) can be considered an empirical law. For other physical networks, like fluid flow networks, deriving the linear response equation from conservation laws and established empirical laws in three-dimensional space is an essential step in determining the appropriate network equations. Very often, one will find that no linear response equation exists.

### B. Solving linear flow networks I: Node analysis

The node analysis and loop analysis are two equally valid ways of solving linear flow networks. Depending on how boundary conditions are interpreted, one can use either. In the node analysis, boundary conditions (such as battery voltages) are applied at the nodes. As a consequence, there are no loop voltages: KVL is satisfied at every face with $\vec{\psi}_\ell = 0$. This allows us to write the voltage as the gradient of a potential using (13):

$$\hat{B}\vec{\Psi} = \vec{0} \implies \vec{\Psi} = -\hat{\Delta}\vec{\psi}. \tag{23}$$

Decoupling (23), (15), and (16) gives a single equation on the nodes written in terms of the graph Laplacian $\hat{\mathcal{L}} = \hat{\Delta}^T \hat{\kappa} \hat{\Delta}$:

$$\hat{\mathcal{L}}\vec{\psi} = -\vec{i}. \tag{24}$$

Before solving this equation, let's address the question of why $\hat{\mathcal{L}}$ is called a Laplacian with an example. Let $G$ be a square grid with uniform conductances $\kappa_k = 1$. The nodes are represented with the coordinates on the grid, $(x, y) \in \mathcal{V}$. Equation (24) for the node $(x, y)$ reads

$$[\hat{\mathcal{L}}\psi]_{(x,y)} = -(\psi_{(x,y+\Delta y)} + \vec{\psi}_{(x+\Delta x,y)} + \psi_{(x,y-\Delta y)} + \psi_{(x-\Delta x,y)}) + 4\psi_{(x,y)}.$$

If this graph is embedded in $\mathbb{R}^2$, and the potential changes smoothly over a length scale much larger than the lattice spacing, then we can approximate this expression with a Taylor expansion:

$$[\hat{\mathcal{L}}\vec{\psi}]_{(x,y)} \approx -\Delta x^2 \left( \frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial x^2} \right)_{(x,y)} = -\Delta x^2 \nabla^2 \psi_{(x,y)}.$$

Thus, for a grid, $-\hat{\mathcal{L}}$ approximates the familiar Laplacian operator $\nabla^2$ in continuous space. In general, the topology of $\hat{\mathcal{L}}$ is encoded in the incidence matrix, while the geometry is encoded in the conductance.

---

[19] **EXERCISE**: Try to find an equivalent derivation of KVL from continuum equations. A starting point may be to claim that a change in flux through a surface must be driven by a circulation around the surface's boundary.

Equation (24) is analogous to a Poisson equation in a curved space. Where the node currents are zero, we can take advantage of properties of harmonic functions, as described in the appendix.

Returning to the problem at hand, we wish to solve equation (24). The complication comes from the fact that $\hat{\mathcal{L}}$ is not invertible. If all of the node currents are known, then there are a couple simple computational ways of addressing this problem, such as using the pseudo-inverse described in the first section,[20] but this does not generalize easily when some of the node flows are unknown. The following technique is a general method of inverting the Laplacian for any node boundary conditions. An equivalent description of this method is outlined in Chapter 12 of [8] and in [9] , but we will make use of projection operators which are more computationally explicit.

Denote the set of boundary node voltages $\mathcal{B} \subseteq \mathcal{V}$ and the set of interior nodes $\mathcal{I} = \mathcal{V} \backslash \mathcal{B}$. Suppose the vectors $\{\vec{b}\}$ form a basis for the vector space spanned by boundary nodes. Note that these basis vectors only have $|\mathcal{B}|$ elements. We will define the boundary restriction operator $\hat{\mathbb{P}}_{\mathcal{B}} : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{B}|}$ as

$$\hat{\mathbb{P}}_{\mathcal{B}} = \sum_{i \in \mathcal{B}} \sum_{j \in \mathcal{V}} \vec{b}_i \vec{v}_j^T. \tag{25}$$

Its matrix elements are 1 if the $i$th node in $\mathcal{B}$ is the $j$th node in $\mathcal{V}$ and 0 otherwise. When acting on a vector defined over all of $\mathcal{V}$, it removes the entries of elements in $\mathcal{I}$. The transposed operator $\hat{\mathbb{P}}_{\mathcal{B}}^T$ embeds $\mathcal{B}$ in $\mathcal{V}$. The interior restriction operator $\hat{\mathbb{P}}_{\mathcal{I}}$ can be defined similarly, and the two matrices are orthogonal: $\hat{\mathbb{P}}_{\mathcal{B}} \hat{\mathbb{P}}_{\mathcal{I}}^T = \hat{0}$ and $\hat{\mathbb{P}}_{\mathcal{I}} \hat{\mathbb{P}}_{\mathcal{B}}^T = \hat{0}$. Note that $\hat{\mathbb{P}}_{\mathcal{B}}^T \hat{\mathbb{P}}_{\mathcal{B}} : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}|}$ is a projection operator which sets all components of interior vertices to zero whereas $\hat{\mathbb{P}}_{\mathcal{B}}$ completely removes those entries and will be more useful for our purposes.

Let $\vec{\psi}_{\mathcal{B}} = \hat{\mathbb{P}}_{\mathcal{B}} \vec{\psi} \in \mathbb{R}^{|\mathcal{B}|}$ and $\vec{i}_{\mathcal{I}} = \hat{\mathbb{P}}_{\mathcal{I}} \vec{i} \in \mathbb{R}^{|\mathcal{I}|}$ be known quantities, while $\vec{\psi}_{\mathcal{I}} = \hat{\mathbb{P}}_{\mathcal{I}} \vec{\psi} \in \mathcal{I}$ and $\vec{i}_{\mathcal{B}} = \hat{\mathbb{P}}_{\mathcal{B}} \vec{i} \in \mathcal{B}$ are unknown. Because the flows can be uniquely determined up to a constant shift in pressures, in order to obtain a unique solution, it is necessary that $\mathcal{B}$ be non-empty. When $\vec{i}_{\mathcal{I}} = 0$, current is conserved at all interior nodes, and one is left with **Dirichlet boundary conditions**; in this case, $\mathcal{B}$ must contain at least two elements to get a non-trivial solution. When only one pressure is fixed to zero (the ground node) and the remaining boundary conditions are set on the flow (typically zero for all but a small number of sources and sinks), one is left with **Neumann boundary conditions**. Note that these flow source and sink nodes are, by our definition, elements of $\mathcal{I}$; only the pressure boundary nodes are elements of $\mathcal{B}$.

The advantage to introducing the previous notation for describing boundary conditions is that the operator $\hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}} \equiv \hat{\mathbb{P}}_{\mathcal{I}} \hat{\mathcal{L}} \hat{\mathbb{P}}_{\mathcal{I}}^T : \mathbb{R}^{|\mathcal{I}|} \to \mathbb{R}^{|\mathcal{I}|}$, called the **reduced Laplacian**, is invertible. The reduced Laplacian is simply the Laplacian with the pressure boundary rows and columns removed. We can also define $\hat{\mathcal{L}}_{\mathcal{I}\mathcal{B}} \equiv \hat{\mathbb{P}}_{\mathcal{I}} \hat{\mathcal{L}} \hat{\mathbb{P}}_{\mathcal{B}}^T$, etc. Equation (24) can be decomposed as follows[21]:

$$\begin{pmatrix} \hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}} & \hat{\mathcal{L}}_{\mathcal{I}\mathcal{B}} \\ \hat{\mathcal{L}}_{\mathcal{B}\mathcal{I}} & \hat{\mathcal{L}}_{\mathcal{B}\mathcal{B}} \end{pmatrix} \begin{pmatrix} \vec{\psi}_{\mathcal{I}} \\ \vec{\psi}_{\mathcal{B}} \end{pmatrix} = - \begin{pmatrix} \vec{i}_{\mathcal{I}} \\ \vec{i}_{\mathcal{B}}. \end{pmatrix}. \tag{26}$$

Solving for the unknowns,

$$\vec{\psi}_{\mathcal{I}} = \hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}}^{-1} \left[ -\vec{i}_{\mathcal{I}} - \hat{\mathcal{L}}_{\mathcal{I}\mathcal{B}} \vec{\psi}_{\mathcal{B}} \right] \tag{27}$$

$$\vec{i}_{\mathcal{B}} = (\hat{\mathcal{L}}_{\mathcal{B}\mathcal{I}})(\hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}})^{-1} \vec{i}_{\mathcal{I}} - \left[ (\hat{\mathcal{L}}_{\mathcal{B}\mathcal{B}}) - (\hat{\mathcal{L}}_{\mathcal{B}\mathcal{I}})(\hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}})^{-1} (\hat{\mathcal{L}}_{\mathcal{I}\mathcal{B}}) \right] \vec{\psi}_{\mathcal{B}}. \tag{28}$$

These expressions can often be simplified. For the special case of Neumann boundary conditions, $\vec{\psi}_{\mathcal{B}} = \vec{0}$. For the special case of Dirichlet boundary conditions, $\vec{i}_{\mathcal{I}} = \vec{0}$, and equation (28) looks a lot like equation

---

[20] **EXERCISE**: Prove that $\vec{\psi} = -\hat{\mathcal{L}}^+ \vec{i}$ solves $\hat{\mathcal{L}} \vec{\psi} = -\vec{i}$, where $\hat{\mathcal{L}}^+$ is the pseudo-inverse defined in equation (4). Then, show that $\vec{\psi} = -\hat{\mathcal{L}}^+ \vec{i} + [\hat{\mathbb{1}} - \hat{\mathcal{L}}^+ \hat{\mathcal{L}}] \vec{w}$ also solves $\hat{\mathcal{L}} \vec{\psi} = -\vec{i}$ for any choice of $\vec{w} \in \mathbb{R}^N$.

[21] **EXERCISE**: This equation is most apparent when the nodes in $\mathcal{V}$ are already ordered such that all the interior nodes come first and all the boundary nodes come after, but this form holds true regardless of the ordering in $\mathcal{V}$. Prove this decomposition holds in general. The completeness relation $\hat{\mathbb{P}}_{\mathcal{B}}^T \hat{\mathbb{P}}_{\mathcal{B}} + \hat{\mathbb{P}}_{\mathcal{I}}^T \hat{\mathbb{P}}_{\mathcal{I}} = \hat{\mathbb{1}}$ may be useful.

(24) with the quantity in brackets playing the role of a Laplacian on the boundary nodes.[22] Once the interior potentials have been solved for, we can repackage the potentials back into our original basis: $\vec{\psi} = \hat{\mathbb{P}}_{\mathcal{B}}^T \vec{\psi}_{\mathcal{B}} + \hat{\mathbb{P}}_{\mathcal{I}}^T \vec{\psi}_{\mathcal{I}}$. Then, the edge currents can be obtained using (15).

A step-by-step demonstration of the node analysis in a system with both a current and voltage source is shown in the left column of figure 5. A minimal Python code to solve that network is shown below. In step 1, the graph is constructed. It is important to ensure that a consistent basis is used for nodes and edges. One simple way to do this is to save the nodes and edges in lists (`ordered_nodes` and `ordered_edges`), and pass these lists into `nx.incidence_matrix`. In step 2, we define the physics on the graph by specifying the conductance matrix $\hat{\kappa}$, the boundary potential vector $\vec{\psi}_{\mathcal{B}}$, and the interior current vector $\vec{i}_{\mathcal{I}}$. Note that we have, at this point, fixed a basis for the boundary and interior spaces, and that the sum of the dimensions of these spaces must equal the total number of nodes. In step 3, the reduced Laplacian is inverted to solve for $\vec{\psi}_{\mathcal{I}}$. Note that `PP_I` and `PP_B` are the interior restriction and boundary restriction operators $\hat{\mathbb{P}}_{\mathcal{I}}$ and $\hat{\mathbb{P}}_{\mathcal{B}}$, constructed by taking the first $|\mathcal{I}|$ rows and last $|\mathcal{B}|$ rows of a $|\mathcal{V}| \times |\mathcal{V}|$ identity matrix, respectively. Even if we had not put all of the boundary nodes at the end of `ordered_nodes`, we could select the rows of the identity matrix based on which nodes are in the interior versus boundary. At the end of step 3, the node potentials $\vec{\psi}$ and edge currents $\vec{I}$ are completely solved for in the bases specified by `ordered_nodes` and `ordered_edges`. This algorithm can be adapted to any linear flow network.

Listing 3. Solving a linear flow network computationally

```
import networkx as nx
import numpy as np
import scipy.sparse as sp

#Step 1: Label graph, calculate incidence matrix
G = nx.DiGraph()
ordered_nodes = [1, 2, 3, 4, 5, 6]
ordered_edges = [(5,1),(5,3),(2,6),(4,6),(1,2),(3,4),(1,3),(2,4)]
G.add_nodes_from(ordered_nodes)
G.add_edges_from(ordered_edges)
nx.set_node_attributes(G, {1: (0,1), 2: (1,1), 3: (0,0), 4: (1,0),
                           5: (-np.sqrt(3)/2,1/2), 6: (1+np.sqrt(3)/2,1/2)},
                       "pos")
DeltaT = nx.incidence_matrix(G, oriented=True,
                             nodelist=ordered_nodes, edgelist=ordered_edges)
Delta = DeltaT.T

#Step 2: Identify Knowns and Unknowns
conductance = np.array([1, 1, 1, 1, 1, 1, 1, 1])
psi_B = np.array([1, 0])
i_I = np.array([-1, 1, 0, 0])
N_B = len(psi_B)
N_I = len(i_I)
N = nx.number_of_nodes(G)
M = nx.number_of_edges(G)
assert N_I + N_B == N and N_B > 0, "Invalid boundary conditions"
assert len(conductance) == M, "Invalid conductance assignment"

#Step 3: Solve for unknowns by inverting reduced Laplacian
Conductance = sp.diags(conductance, format="csr")
Laplacian = DeltaT @ Conductance @ Delta
```

[22] Mathematicians would call $(\hat{\mathcal{L}}_{\mathcal{BB}}) - (\hat{\mathcal{L}}_{\mathcal{BI}})(\hat{\mathcal{L}}_{\mathcal{II}})^{-1}(\hat{\mathcal{L}}_{\mathcal{IB}})$ the Schur complement of $\hat{\mathcal{L}}$ with respect to $\hat{\mathcal{L}}_{\mathcal{II}}$. This is a new Laplacian defined only on the boundary obtained by performing Gaussian elimination on the interior nodes [8]. The operator $(\hat{\mathcal{L}}_{\mathcal{BI}})(\hat{\mathcal{L}}_{\mathcal{II}})^{-1}$ maps boundary flows to interior flows, and one can interpret equation (28) as an equation on a reduced network consisting only of the boundary nodes [9].

```
PP_I = sp.eye(N, N, format="csr")[:N_I, :]
PP_B = sp.eye(N, N, format="csr")[-N_B:, :]
L_II = PP_I @ Laplacian @ PP_I.T
L_IB = PP_I @ Laplacian @ PP_B.T
psi_I = sp.linalg.spsolve(L_II, -i_I - L_IB @ psi_B)
psi = PP_I.T @ psi_I + PP_B.T @ psi_B
I = -Conductance @ Delta @ psi

print(psi)
print(I)
```

### C. Solving Linear Flow Networks II: Loop analysis

The loop analysis is more appropriately named the face analysis since its equations are solved on the faces of the graph. In the loop analysis [7, 10], boundary conditions are imposed on the faces, and KCL is everywhere satisfied with $\vec{i} = 0$, which allows us to write the current as the curl of a loop current using (12):

$$\hat{\Delta}^T \vec{I} = \vec{0} \implies \vec{I} = \hat{B}^T \vec{i}_\ell. \tag{29}$$

Notably, if an edge is incident to only a single face, then the loop current is equal to the edge current (if the orientation of the cycle and the edge agree, otherwise the loop current is the negative edge current). Decoupling equations (29), (15), and (17) gives an equation in terms of the face Laplacian $\hat{\mathcal{L}}_2 = \hat{B}\hat{\kappa}^{-1}\hat{B}^T$:

$$\hat{\mathcal{L}}_2 \vec{i}_\ell \equiv \hat{B}\hat{\kappa}^{-1}\hat{B}^T \vec{i}_\ell = \vec{\psi}_l. \tag{30}$$

At this point, the computation parallels that of the node analysis (its even easier in fact), but constructing an appropriate basis and defining the loop currents $\vec{i}_\ell$ and loop voltages $\vec{\psi}_\ell$ can be confusing. Typically, the cycle basis for a planar graph is chosen to form a mesh: cycles in a planar graph which do not contain other cycles, and all cycles are taken to be faces. When a battery supplies a voltage, one can draw a face on the loop containing the battery, and if the orientation of the battery and face agree, then a positive loop voltage equal to the battery voltage is fixed on that face (if the orientations disagree, the loop voltage is equal to minus the battery voltage). The same can be said for loop currents. In the circuit in figure 5, the loop voltage and loop current can be identified intuitively. Carefully choosing a cycle basis can prevent confusion regarding these quantities.[23]

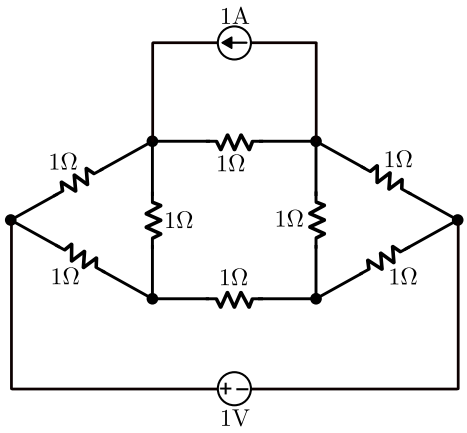
The loop current $\vec{i}_\ell$ is unknown everywhere except along the cycles with current sources, and the loop voltage, or EMF, $\vec{\psi}_\ell$ is known everywhere except along the cycles with current sources. Just as with our node analysis where we had to decompose our space into boundary nodes (where the potential was known) and interior nodes (all other nodes, including node current sources), here we will have to decompose our space into boundary faces (where the loop current is known) and interior faces (all other faces, including voltage sources). Then, the loop currents can be found by inverting $\hat{B}\hat{\kappa}^{-1}\hat{B}^T$ restricted to the interior in equation (30). Unlike the node analysis, the boundary here can be empty since $\hat{B}\hat{\kappa}^{-1}\hat{B}^T$ does not necessarily have a zero eigenvalue. The physical currents can then be found using (29) and the voltages can be found using (15).

When should one use the loop analysis instead of the node analysis? Recall that there are $|\mathcal{E}| - |\mathcal{V}| + 1$ loops and $|\mathcal{V}|$ nodes in a connected graph, so if there are many more edges than nodes (many parallel connections), then a node analysis is more efficient whereas if the number of edges is comparable to the number of nodes (many series connections), a loop analysis is more efficient.[24] However, if computational

[23] **EXERCISE**: Make sure you know how to go from the circuit at the top of figure 5 to the vectors in step 2. Find the edge currents $\vec{I}$ using both the node and loop analyses. Did you get the same answer?

[24] **EXERCISE**: Prove using a loop analysis that for a network with resistors connected in series to a single voltage source, the loop current is related to the source voltage by $\psi_\ell = (\sum_\mu \kappa_\mu^{-1}) i_\ell$. Then, prove using a node analysis that for a network containing two nodes with many resistors in parallel, the potential difference between the two nodes is related to the node current by $\sum_\mu \kappa_\mu(\psi_1 - \psi_2) = i_2$. Comment on the difficulty on trying to prove these relations using the opposite analyses.

Circuit
(Linear Flow Network)



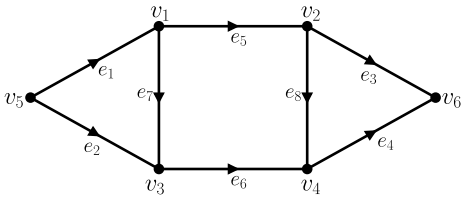
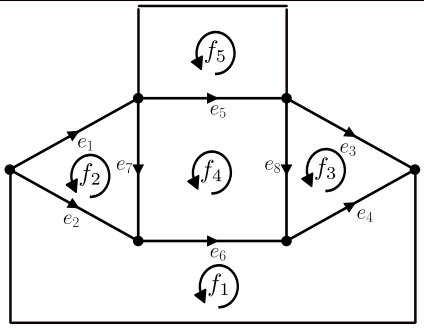

| | Node analysis | Loop analysis |
|---|---|---|
| Step 1: Label graph, calculate incidence matrix |   $\hat{\Delta}^T = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 1 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$ |   $\hat{B}^T = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \end{array} \begin{array}{c} \begin{array}{ccccc} f_1 & f_2 & f_3 & f_4 & f_5 \end{array} \\ \begin{pmatrix} 0 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 \end{pmatrix} \end{array}$ |
| Step 2: Identify knowns and unknowns | $\vec{\imath} = \begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \\ \hdashline i_5 \\ i_6 \end{pmatrix}$ $\vec{\psi} = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \psi_4 \\ \hdashline 1 \\ 0 \end{pmatrix}$ | $\vec{\psi}_\ell = \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ \hdashline \psi_{\ell,5} \end{pmatrix}$ $\vec{\imath}_\ell = \begin{pmatrix} i_{\ell,1} \\ i_{\ell,2} \\ i_{\ell,3} \\ i_{\ell,4} \\ \hdashline 1 \end{pmatrix}$ |
| Step 3: Sove for unknowns by inverting reduced Laplacian | $\hat{\mathcal{L}} = \hat{\Delta}^T \hat{C} \hat{\Delta} = \left(\begin{array}{cccc:cc} 3 & -1 & -1 & 0 & -1 & 0 \\ -1 & 3 & 0 & -1 & 0 & -1 \\ -1 & 0 & 3 & -1 & -1 & 0 \\ 0 & -1 & -1 & 3 & 0 & -1 \\ \hdashline -1 & 0 & -1 & 0 & 2 & 0 \\ 0 & -1 & 0 & -1 & 0 & 2 \end{array}\right)$ $\hat{\mathcal{L}}\vec{\psi} = -\vec{\imath} \implies \vec{\psi}_{\mathcal{I}} = \hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}}^{-1}\left[-\vec{\imath}_{\mathcal{I}} - \hat{\mathcal{L}}_{\mathcal{I}\mathcal{B}}\vec{\psi}_{\mathcal{B}}\right]$ $\vec{I} = -\hat{C}\hat{\Delta}\vec{\psi}$ | $\hat{\mathcal{L}}_2 = \hat{B}\hat{C}^{-1}\hat{B}^T = \left(\begin{array}{cccc:c} 3 & -1 & -1 & -1 & 0 \\ -1 & 3 & 0 & -1 & 0 \\ -1 & 0 & 3 & -1 & 0 \\ -1 & -1 & -1 & 4 & -1 \\ \hdashline 0 & 0 & 0 & -1 & 1 \end{array}\right)$ $\hat{\mathcal{L}}_2\vec{\imath}_\ell = \vec{\imath}_l \implies \vec{\imath}_\ell = \hat{\mathcal{L}}_{2,\mathcal{I}\mathcal{I}}^{-1}\left[\vec{\psi}_{\ell,\mathcal{B}} - \hat{\mathcal{L}}_{2,\mathcal{I}\mathcal{B}}\vec{\imath}_{\ell,\mathcal{B}}\right]$ $\vec{I} = \hat{B}^T\vec{\imath}_\ell$ |

FIG. 5. Review of circuit theory. An example circuit is shown on the top with a 1 Amp current source, 1 Volt DC battery, and eight 1 Ohm resistors. The edge currents $\vec{I}$ can be found by using either the node analysis or loop analysis. In the node analysis, the node-edge incidence matrix $\hat{\Delta}^T$ is used, the interior node currents $\vec{\imath}_{\mathcal{I}}$ and boundary potentials $\vec{\psi}_{\mathcal{B}}$ are known, the unknown potentials $\vec{\psi}_I$ are found by inverting the reduced node Laplacian $\hat{\mathcal{L}}_{\mathcal{I}\mathcal{I}}$, and the edge currents $\vec{I}$ are found using Ohm's Law. In the loop analysis, the edge-face incidence matrix $\hat{B}^T$ is used, the interior loop voltages $\vec{\psi}_{\ell,\mathcal{I}}$ and boundary loop currents $\vec{\imath}_{\ell,\mathcal{B}}$ are known, the unknown loop currents $\vec{\imath}_{\ell,\mathcal{I}}$ are found by inverting the reduced face Laplacian $\hat{\mathcal{L}}_{2,\mathcal{I}\mathcal{I}}$, and the edge currents $\vec{I}$ are found by taking the curl of the loop currents. The dashed lines in the matrices are used to separate interior and boundary elements.

efficiency isn't a concern, then it is best to use whichever method is easiest and most intuitive, which for many of our applications is the node analysis. The loop analysis is also harder to implement in Python. Note that our code for analyzing the node analysis was simple once the graph was constructed because we could use the function `nx.incidence_matrix` to calculate the node-edge incidence matrix. There is no function in NetworkX for calculating the edge-face incidence matrix. It is good to know that the loop analysis exists as it helps to clarify the structure of Kirchhoff's Laws and may greatly improve efficiency of solving certain networks, but we (the authors) can honestly say we never use it. For the remainder of the paper, we will only be using the node analysis.

### D. Power minimization

Physical laws are often the result of energy minimization principles, or in the case of circuits, power. For an electric circuit (in the node analysis where KVL is everywhere satisfied), the solution $\vec{\psi}$ to an equation of the form (24) minimizes the functional[25]

$$E[\vec{\psi}] = \frac{1}{2}\vec{\psi}^T\hat{\mathcal{L}}\vec{\psi} + \vec{\psi}^T\vec{i} = \frac{1}{2}(\hat{\Delta}\vec{\psi})^T\hat{\kappa}\hat{\Delta}\vec{\psi} + \vec{\psi}^T\vec{i}. \tag{31}$$

The physical potentials are the $\vec{\psi}$ which minimize $E$ subject to the constraints given by the boundary conditions. This is just the familiar formula for the power in an electric circuit: The power is the sum of the voltage drops over all edges, weighted by the conductance of each edge. The factor of $1/2$ accounts for the fact that each edge was counted twice. Note also that

$$\frac{\partial E}{\partial\psi_i\partial\psi_j} = \mathcal{L}_{ij}. \tag{32}$$

That is, the Laplacian we have been working with is the Hessian of the power. It tells us about the curvature in the space of node voltages. The zero eigenvalue of the Laplacian suggests (quite intuitively in this case) that uniform shifts in the potential cost no energy. Changes in the potential aligned with a low curvature (small eigenvalue) direction are less energetically costly than those aligned with a high curvature (large eigenvalue) direction.

The loop analysis (assuming KCL is satisfied everywhere) suggests that we should minimize a power function of the form

$$E[\vec{i}_\ell] = \frac{1}{2}\vec{i}_\ell^{\,T}\hat{\mathcal{L}}_2\vec{i}_\ell - \vec{i}_\ell^{\,T}\vec{\psi}_\ell = \frac{1}{2}(\hat{B}^T\vec{i}_\ell)^T\hat{\kappa}^{-1}\hat{B}^T\vec{i}_\ell - \vec{i}_\ell^{\,T}\vec{\psi}_\ell \tag{33}$$

in order to recover equation (30). Recalling that $\hat{B}^T\vec{i}_\ell = \vec{I}$, we see that this expression is another familiar form of the power: a sum of the currents over all edges, weighted by the resistance of each edge. The Laplacian $\hat{\mathcal{L}}_2$ acts as a Hessian in the power landscape of loop currents.

### E. Quantifying networks with the effective resistance

For any connected graph, the **effective resistance between nodes $v_j$ and $v_k$**, denoted $R_{jk}$ is defined as the ratio of the voltage to the current when a current $i$ is injected at node $v_j$ and extracted at node $v_k$:

$$R_{jk} \equiv \frac{\psi_j - \psi_k}{i}. \tag{34}$$

---

[25] **EXERCISE**: Show that if you take the gradient of equation (31) with respect to $\vec{\psi}$, you recover equation (24).

That is, we solve the Neumann problem with $i_j = -i$, $i_k = +i$, and all other node flows are zero, and calculate the above ratio. The result does not depend on the value of $i$ and is always non-negative (zero for $j = k$). The effective graph resistance $R_G$ is defined as

$$R_G \equiv \frac{1}{2} \sum_{j=1}^{|\mathcal{V}|} \sum_{k=1}^{|\mathcal{V}|} R_{jk}. \tag{35}$$

This characterizes the overall difficulty for flows in the graph. At first glance, this seems like a rather complicated quantity to calculate, since each $R_{ij}$ requires us to solve a different flow problem. Remarkably, it has been shown [2, 11] that the effective graph resistance satisfies

$$R_G = |\mathcal{V}| \sum_{\{k|\lambda_k \neq 0\}} \lambda_k^{-1}, \tag{36}$$

where $\lambda_k$ is an eigenvalue of $\hat{\mathcal{L}}$. Thus, the Laplacian not only informs the local energetics of the graph, it can also be used to globally characterize the resistance of the graph.[26]

### F. Solving Dynamic Linear Flow Networks (RLC Circuits)

The techniques presented so far can be applied to any network with steady potential and current sources. In this section, we will demonstrate how to generalize the node analysis to linear networks with dynamics.

Consider a circuit consisting of edge resistors $R$, edge inductors $L$, node capacitors $C$, and dynamic sources.[27] The responses of individual circuit elements are:

$$\Psi = RI \tag{37}$$

$$\Psi = L\frac{dI}{dt} \tag{38}$$

$$\frac{d\mathrm{Q}}{dt} = C\frac{d\psi}{dt}. \tag{39}$$

Thus, many dynamic linear circuits with voltage sources can be solved using the node analysis and the following equations:

$$-\hat{\Delta}\vec{\psi} = \hat{R}\vec{I} + \hat{L}\frac{d\vec{I}}{dt} \tag{40}$$

$$\hat{\Delta}^T\vec{I} = \hat{C}\frac{d\vec{\psi}}{dt} + \vec{i}. \tag{41}$$

This suggests that each edge has an inductor and resistor in series, and each interior node has a capacitor connected to ground plus an exterior current source, though any of these quantities may be zero.

Of course, there are other linear circuits that one could solve. Each edge may consist of lumped elements of resistors, capacitors, and inductors arranged in a particular way, and identifying an appropriate lumped element is often the first step to solving a linear flow network. An example lumped element consisting of two resistors, an inductor, and a capacitor is shown in figure 6. We could have let each of the four circuit elements be its own edge, but that could lead to complications while solving. When the capacitor is fully charged, the impedance of that branch diverges (the conductance goes to zero), but the impedance of the entire lumped element $Z$ remains finite as long as $R_1$ is finite.

---

[26] **EXERCISE**: Calculate $R_{56}$ and $R_G$ for the graph shown in figure 5.

[27] A node capacitor has one of its terminals connected to ground such that the voltage drop across the capacitor is the node potential (minus zero). One could also consider edge capacitors where the edge current is proportional to the time derivative of the edge voltage, though it is the node capacitors which have a clear analogy to other problems studied in this tutorial, so we will introduce them here.

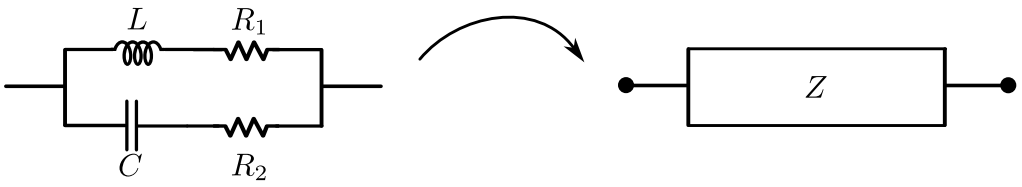



FIG. 6. An example lumped element with complex impedance $Z$ is obtained by grouping four circuit elements together.

### 1. Method 1: Solving dynamic networks in the frequency domain

We seek the steady steady state solution to a circuit driven by periodic (AC) current and voltage sources. There may still be a constant (DC) component driving a constant current. Let $\omega_0$ denote the fundamental frequency such that the boundary conditions have period $2\pi/\omega_0$. Then, we can expand our functions in a Fourier series:

$$\vec{\psi}(t) = \sum_{m=-\infty}^{\infty} \vec{\tilde{\psi}}^{(m)} e^{im\omega_0 t}, \qquad \vec{I}(t) = \sum_{m=-\infty}^{\infty} \vec{\tilde{I}}^{(m)} e^{im\omega_0 t}, \tag{42}$$

where $\tilde{\psi}_i^{(m)}$ denotes the $m$-th Fourier coefficient of the potential on node $v_i$, and $\tilde{I}_\mu^{(m)}$ denotes the $m$-th Fourier coefficient of the potential on edge $e_\mu$. Defining a complex impedance on edge $e_\mu$ as $Z_\mu(\omega) = R_\mu + i\omega L_\mu$, Fourier transforming (40) and (41) gives

$$\vec{\tilde{I}}^{(m)} = -\hat{Z}(m\omega)^{-1}\hat{\Delta}\vec{\tilde{\psi}}^{(m)}, \tag{43}$$

$$\hat{\Delta}^T \vec{\tilde{I}}^{(m)} = im\omega_0 \hat{C}\vec{\tilde{\psi}}^{(m)} + \vec{\tilde{i}}^{(m)} \tag{44}$$

Decoupling these equations gives a complex Laplacian:

$$\left[im\omega_0\hat{C} + \hat{\Delta}^T\hat{Z}^{-1}(m\omega)\hat{\Delta}\right]\vec{\tilde{\psi}}^{(m)} = -\vec{\tilde{i}}^{(m)}. \tag{45}$$

Solving for $\vec{\tilde{\psi}}^{(m)}$ proceeds exactly as outlined in section III.B. One needs to invert a complex-valued Laplacian for each value of $m$. Note that for $m \neq 0$, the operator in brackets is invertible since time-varying potentials lead to measurable changes in flow, while the $m = 0$ (static) component reduces to the usual Laplacian for a resistive circuit which is not invertible. In practice, the driving currents and voltages typically take a simple form, and one only needs to compute a small number of $m$. For example, if the circuit is driven by an AC voltage at node 1, $\psi_1(t) = \sin(\omega_0 t) = (\frac{1}{2i}e^{i\omega_0 t} - \frac{1}{2i}e^{-i\omega_0 t})$, then only the $m = 1$ and $m = -1$ equations will need to be considered. Also, since $\psi_i(t)$ is real, $\tilde{\psi}_i^{(m)} = \left(\tilde{\psi}_i^{(-m)}\right)^*$, so one does not need to separately solve the $m > 0$ and $m < 0$ cases. Once each of the Fourier coefficients $\{\vec{\tilde{\psi}}^{(m)}\}$ is known, we have successfully solved for $\vec{\psi}(t)$ using (42).[28]

Solving dynamic networks in the frequency domain is simple whenever the boundary conditions are periodic. Depending on the problem, the choice of lumped elements and their arrangement with respect to each other (in series or parallel) may be different, in which case the operator in square brackets will be different, but the general technique of using Fourier series to find the pressures for a periodic drive is the same. The method can be generalized to non-periodic driving, but one would need to perform a Fourier transform rather than a Fourier series, and thus evaluate a continuum of Fourier coefficients, and invert an uncountably infinite number of Laplacians. A better approach is to stick to the time domain.

---

[28] **EXERCISE**: Find the complex impedance of the lumped edge in figure 6. Then, solve for the node potentials in the network shown in figure 5 assuming that the current source is now an AC current source, the battery is now an AC voltage (at the same frequency), and the resistors are now impedances like that in figure 6. You may assume that the node capacitance is zero and set all constants equal to 1.

### 2. Method 2: Solving dynamic networks in the time domain

In many cases, one is interested in transient or non-periodic boundary conditions, in which case the Fourier transform is impractical.[29] Here we will demonstrate the simplest and most intuitive approach to solving circuits in the time domain. For simplicity, assume that $L_\mu = 0$. Then, decoupling equations (40) and (41) gives[30]

$$\frac{d}{dt}\vec{\psi} = -\hat{C}^{-1}\hat{\Delta}^T\hat{R}^{-1}\hat{\Delta}\vec{\psi} - \hat{C}^{-1}\vec{i}. \tag{46}$$

An initial condition $\vec{\psi}\,(0)$ must be specified.

Numerically, one could solve this using a forward difference scheme: $\vec{\psi}\,(t+\delta t) = \vec{\psi}\,(t) + \frac{d\vec{\psi}\,(t)}{dt}\delta t$, and invert a Laplacian at each time step. This is particularly useful when solving networks with nonlinearities. Ensuring that the numerical scheme converges to the physical solution requires taking an appropriately small $\delta t$ and may work better for certain schemes over others. See [12] for details. However, our linear problem can actually be solved without discretizing time. First, project onto the interior nodes (the potential at the boundary nodes is known for all times afterall):

$$\frac{d}{dt}\vec{\psi}_{\mathcal{I}} = -[\hat{C}^{-1}\hat{\Delta}^T\hat{R}^{-1}\hat{\Delta}]_{\mathcal{I}\mathcal{I}}\vec{\psi}_{\mathcal{I}} - [\hat{C}^{-1}\hat{\Delta}^T\hat{R}^{-1}\hat{\Delta}]_{\mathcal{I}\mathcal{B}}\vec{\psi}_{\mathcal{B}} - \hat{C}^{-1}\vec{i}_{\mathcal{I}}. \tag{47}$$

The subscripts $\mathcal{I}\mathcal{I}$ and $\mathcal{I}\mathcal{B}$ are defined using the $\hat{\mathbb{P}}_{\mathcal{I}}$ and $\hat{\mathbb{P}}_{\mathcal{B}}$ operators from section III.B. The above equation is now just a system of first-order linear differential equations [13] . Given an equation of the form

$$\frac{d}{dt}\vec{x} = \hat{A}\vec{x} + \vec{b}, \qquad \vec{x}\,(0) = \vec{x}_{\,0}, \tag{48}$$

the solution is[31]

$$\vec{x}\,(t) = e^{\hat{A}t}\vec{x}_{\,0} + \int_0^t e^{\hat{A}(t-s)}\vec{b}\,(s)ds. \tag{49}$$

For our $RC$ circuit, this suggests that the reciprocal of the eigenvalues of the matrix $[\hat{C}^{-1}\hat{\Delta}^T\hat{R}^{-1}\hat{\Delta}]_{\mathcal{I}\mathcal{I}}$ describe the characteristic $RC$ time scales.

---

[29] Transient effects can also be studied using a third method: the Laplace transform, which conveniently incorporates initial conditions, though may be impractical for general driving forces.

[30] **EXERCISE**: The solution presented in this subsection fails when the capacitance at some of the interior nodes vanishes. Suppose only some interior nodes have capacitors while others have a known node current (either zero or a current source). Generalize the method in this section to account for these combined node boundary conditions. [Hint: Partition the space $\mathcal{I}$ into two spaces, and generalize the method from section III.B.]

[31] The exponential of a matrix is defined by its power series $e^{\hat{A}t} = \sum_{k=0}^{\infty}\frac{(At)^k}{k!}$. If the matrix is diagonalizable with left eignvectors $\{\vec{\xi}_{\,L}\}$, right eigenvectors $\{\vec{\xi}_{\,R}\}$, and eigenvalues $\{\lambda_i\}$, then this reduces to $e^{\hat{A}t} = \sum_i e^{\lambda_i t}\vec{\xi}_{\,R,i}\vec{\xi}^{\,T}_{L,i}$, where we now only have to sum over the finite number of eigenvectors of $\hat{A}$.

## IV. FLUID FLOW NETWORKS

So far, we have studied some general properties of linear flow networks and their application to electrical circuits. Here, we will show that there is a direct mapping between resistive electronic circuits and viscous fluid flow networks. However, unlike circuits, where the voltage drop across an edge occurs over discrete elements, fluid flow networks have a continuous pressure drop along each tube. It is important to develop an understanding of fluids in three spatial dimensions before proceeding to fluids in networks. First, we will review the fundamental equations of fluid mechanics and derive KCL. Second, we will show how steady, fully-developed flow gives rise to the analogue of Ohm's Law, with viscosity playing the role of resistance. Third, we will study flow in compliant vessels where the flow is no longer uniform across an edge. A review of fluid flow networks was written by Basu and Alim [14].

### A. A brief review of fluid dynamics

Before studying flow on networks, let's review the basic equations of fluid dynamics. A compact introduction can be found in the book by Childress [15].

A fluid is described by a velocity field $\boldsymbol{v}(\boldsymbol{r})$, a stress field $\boldsymbol{\sigma}(\boldsymbol{r})$[32], and a density field $\rho(\boldsymbol{r})$. The continuity equation derived from mass conservation is

$$-\boldsymbol{\nabla}\cdot(\rho\boldsymbol{v}) = \frac{\partial\rho}{\partial t}. \tag{50}$$

We could immediately integrate this equation at a node, like we did in section III.A, to obtain KCL for fluid flow networks.[33] In that case, the edge current would be the mass flow rate. However, this is almost never the quantity studied in liquid flows. We will consider the edge current more commonly used for liquid flow networks: the volumetric flow rate.

When studying fluids, it is often necessary to consider the rate of change of a property of a fluid parcel as it is being carried by the flow. Mathematically, this is described by the material derivative:

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \boldsymbol{v}\cdot\boldsymbol{\nabla}. \tag{51}$$

An **incompressible fluid** has a constant density

$$\frac{D\rho}{Dt} = 0. \tag{52}$$

Combining equations (50) and (52) gives that the divergence of an incompressible fluid is zero:

$$\boldsymbol{\nabla}\cdot\boldsymbol{v} = 0. \tag{53}$$

This takes the form of a continuity equation, with conserved quantity volume V. Most liquids are very nearly incompressible, and equation (53) is taken as a starting point. Define the volumetric flow rate, or **flow**:

$$Q \equiv \int_S \boldsymbol{v}\cdot d\boldsymbol{A}. \tag{54}$$

Then, integrating over a volume like that in figure 4 gives rise to a KCL at the nodes:

$$(\hat{\Delta}^T\vec{Q}\,)_j = \frac{d\mathrm{V}_j}{dt} + q_j \tag{55}$$

---

[32] This is the Cauchy stress tensor, a symmetric tensor describing the stress of a deformed configuration. $\sigma_{ij}$ describes the $i$-th component of the stress acting on a surface whose normal is in the $j$-th direction.

[33] **EXERCISE**: Perform this calculation. Show that the conservation of fluid mass gives rise to a mass flow rate conserved at the nodes. The mass flow rate could be used to study the flow of a compressible gas through pipes.

where $\frac{dV_j}{dt}$ describes the accumulation of volume due to an expanding tube, and $q_j$ describes the volume flowing out of node $v_j$.

Fluids also satisfy momentum conservation:[34]

$$\rho \frac{D\boldsymbol{v}}{Dt} = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}. \tag{56}$$

This equation states that the transport of momentum is caused by internal stresses. The stress tensor $\sigma_{ij}$ can be further decomposed into normal $-p\delta_{ij}$ and shear $\tau_{ij}$ components:

$$\sigma_{ij} = -p\delta_{ij} + \tau_{ij}. \tag{57}$$

The scalar function $p$ is called the **pressure**. The pressure will be the potential function studied on the nodes of our network. Typically, we will concern ourselves with **Newtonian** fluids in which the shear stress $\tau_{ij}$ satisfies[35]

$$\tau_{ij} = \mu(\partial_j v_i + \partial_i v_j), \tag{58}$$

where $\mu$ is the viscosity; in which case, equation (56) becomes the **Navier-Stokes equations**, but for now, let's consider a general shear stress. Whereas the pressure describes the isotropic forces acting in the fluid, the shear stress can be thought of as frictional forces arising from fluid layers moving at different velocities. Thus, we expect $\tau_{ij}$ depends only on the gradients of the velocity. Combining equations (56) and (57) gives

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} + \rho(\boldsymbol{v} \cdot \boldsymbol{\nabla})\boldsymbol{v} = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \boldsymbol{\tau}. \tag{59}$$

We are interested in fluid-carrying cylindrical pipes, so let us put our equations in cylindrical coordinates (and assume no $\phi$-dependence). Only one independent component of the shear is nonzero: $\tau \equiv \tau_{rx} = \tau_{xr}$. The mass continuity (53) and momentum conservation (56) equations become:

$$\frac{\partial v_x}{\partial x} + \frac{1}{r}\frac{\partial (r v_r)}{\partial r} = 0 \tag{60}$$

$$\rho \left[ \frac{\partial v_x}{\partial t} + v_x \frac{\partial v_x}{\partial x} + v_r \frac{\partial v_x}{\partial r} \right] = -\frac{\partial p}{\partial x} + \frac{1}{r}\frac{\partial (r\tau)}{\partial r} \tag{61}$$

$$\rho \left[ \frac{\partial v_r}{\partial t} + v_x \frac{\partial v_r}{\partial x} + v_r \frac{\partial v_r}{\partial r} \right] = -\frac{\partial p}{\partial r} + \frac{\partial \tau}{\partial x}. \tag{62}$$

### B. Pipe flow

Consider a cylindrical pipe whose length $\ell_0$ is much larger than its radius $r_0$. The flow is **steady**, meaning that the velocity and stress are independent of time, and **fully-developed**, meaning that the velocity profile is approximately independent of the position along the tube: $\boldsymbol{v}(r, x) = \boldsymbol{v}(r)$, and therefore the flow $Q$ is a constant across the tube. As a consequence of (60), we must have that $v_r(r) = 0$, which also implies that $\tau(r, x) = \tau(r)$. The geometry is shown in figure 7. For steady, fully-developed flow, equations (61) and (62) simplify to

$$\frac{\partial p}{\partial x} = \frac{1}{r}\frac{\partial (r\tau)}{\partial r} \tag{63}$$

---

[34] We are neglecting body forces such as gravity.

[35] This is the most general form for a linear relationship between $\tau_{ij}$ and $\partial_j v_i$ in an isotropic, incompressible fluid. If the fluid is compressible, then an additional term proportional to $\partial_k u_k \delta_{ij}$ is also allowed.

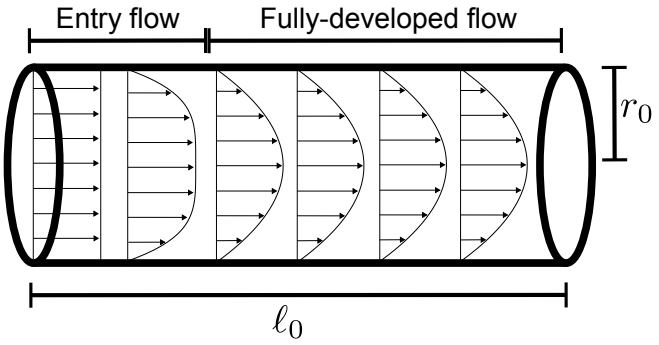



FIG. 7. When a viscous fluid enters a pipe, it initially has a flat profile (entry flow), but as it travels down the pipe, it eventually develops a parabolic profile (fully-developed flow). If $\ell_0 \gg r_0$, then the flow is fully developed throughout almost all of the tube.

$$\frac{\partial p}{\partial r} = 0. \tag{64}$$

The second equation tells us that the $p(x, r) = p(x)$, which allows us to integrate the first equation to obtain:

$$\frac{r}{2}\frac{\partial p}{\partial x} = \tau(r). \tag{65}$$

Note that the right-hand side is not a function of $x$, so it must be that $p$ is a linear function of $x$:

$$\frac{\partial p}{\partial x} = \frac{p(\ell_0) - p(0)}{\ell_0}. \tag{66}$$

Keeping in mind that $\tau$ only depends on $r$ through the gradients of the velocity, all we need to solve the pipe flow problem is a constitutive equation relating the shear stress to the relevant velocity gradient: $\tau(\partial_r v_x)$.[36] This relationship is linear only for Newtonian fluids, equation (58). All other fluids are called non-Newtonian. Fluids that have an effective viscosity that increases with increasing strain rate are called shear thickening, while fluids that have an effective viscosity that decreases with increasing strain rate are called shear thinning. Fluids which have zero strain until a finite yield stress is applied are called plastic. Blood is an example of a Casson fluid, that is, one with both plastic and shear thinning properties. We will first explore pipe flow for a Newtonian fluid, while the Casson fluid is left as an exercise.

In any case, we apply **no-slip** boundary conditions by enforcing that the velocity vanishes on the boundary[37]:

$$\boldsymbol{v}(r_0) = 0. \tag{67}$$

Once a constitutive equation for the shear stress has been specified, one can solve for the flow as a function of pressure. Let's assume the fluid is Newtonian:

$$\tau = \mu\frac{\partial v_x}{\partial r}. \tag{68}$$

The coefficient $\mu$ is called the **viscosity**. With this simple form, integrate equation (65) and apply the no-slip condition (67) to obtain the velocity:

$$v_x(r) = -\frac{r_0^2}{4\mu}\left(1 - \frac{r^2}{r_0^2}\right)\frac{\partial p}{\partial x}. \tag{69}$$

---

[36] Some references prefer to define the **strain rate** $\dot{\gamma} \equiv |\partial_r v_x|$, and specify the magnitude of the shear stress as a function of the strain rate: $|\tau|(\dot{\gamma})$. One can often write $|\tau|(\dot{\gamma}) = |\tau_y| + \mu_{\text{eff}}(\dot{\gamma})\dot{\gamma}$ where $\tau_y$ is the yield stress and $\mu_{\text{eff}}$ defines an effective viscosity.

[37] In general, no-slip boundary conditions dictate that at a fluid-solid boundary, the fluid velocity matches the velocity of the solid.

The velocity as a function of $r$ is parabolic for steady, fully-developed flow of a Newtonian fluid. Integrating equation (69) and applying the definition of the flow (54), we find that the flow is related to the pressure drop across the pipe via

$$Q = -\frac{\pi r_0^4}{8\mu\ell_0}[p(\ell_0) - p(0)]. \tag{70}$$

This is the most commonly used model of pipe flow, commonly known as the **Hagen-Poiseuille Law**. Notice that this equation is exactly of the form (15) provided that we define the conductance as

$$\kappa = \frac{\pi r_0^4}{8\mu\ell_0}. \tag{71}$$

Fluid viscosity behaves mathematically like an electrical resistor, with $Q$ playing the role of electric current and $\Delta p$ playing the role of a voltage drop.[38] Solving for the pressures in the network amounts to solving the equation

$$\hat{\Delta}^T\hat{\kappa}\hat{\Delta}\vec{p} = -\vec{q} \tag{72}$$

where $\vec{q}$ is the vector of node flows. Just as with circuits, we will need to separate the knowns and unknowns and invert the reduced Laplacian to obtain the unknown pressures. Once the pressures at each node are found, the flows are obtained using Ohm's law:

$$\vec{Q} = -\hat{\kappa}\hat{\Delta}\vec{p}. \tag{73}$$

The actual network equations are as simple as solving a resistive circuit. Appreciate the fact that we started from equations in three spatial dimensions (the Navier-Stokes equations), and obtained equations in zero spatial dimensions. Since we carefully applied a series of approximations, we can recover additional spatial information: the shear stress on the walls, the velocity field in the tubes, etc. The simplification from 3D $(r, \phi, x)$ to 1D $(x)$ was essentially obtained by integrating over the cross-section of the pipe. Though the math may get more involved, one can always define the flow as in (54), and consider the cross-sectionally averaged pressure.[39] The reduction from 1D to 0D is more challenging. For steady, fully developed flow, it just so happened that $Q$ was independent of the position along the pipe, and $p$ was a linear function of $x$. In the next section, we will demonstrate how to reduce a PDE in one spatial dimension to a network problem in a more complicated scenario where $Q$ is a function of $x$.

### C. Solving PDEs in Networks (Pulsatile Flow Through Compliant Vessels)

Fluid flow networks are commonly applied to vascular systems in biology. So far we have only studied steady flow, but the flow induced by a pumping heart is certainly not steady! Furthermore, biological vessels can expand to accommodate more volume. In order to take these effects into account, we will first revisit the continuity and momentum equations. Related calculations can be found in [16–18].

Let the tube radius change with space and time. The flow can still be defined by (54) provided that we make the replacement $r_0 \to r_0 + u_r(x,t)$, where $r_0$ is the rest radius of the tube, and $u_r(x,t)$ is the radial displacement. For the case of moving boundaries, the no-slip condition enforces that the fluid velocity equals

---

[38] **EXERCISE**: One can study the steady, fully-developed flow for non-Newtonian fluids as well. Consider blood as an example. The presence of red blood cells gives blood plastic and shear thinning properties. Blood can be modeled as a Casson fluid:

$$\sqrt{\left|\mu\frac{\partial v_x}{\partial r}\right|} = \begin{cases} 0, & |\tau| \le |\tau_y| \\ \sqrt{|\tau|} - \sqrt{|\tau_y|}, & |\tau| > |\tau_y| \end{cases},$$

where $\tau_y$ is the yield stress. For stresses $|\tau|$ below the yield stress $|\tau_y|$, the fluid moves as a rigid body. Above the yield stress, the fluid is shear thinning, approaching a viscosity of $\mu$ for large strain rates. The magnitude of the shear stress is smaller near the center of the tube so define $r_y$ as the radius below which $|\tau| < |\tau_y|$. Find the velocity profile $v_x(r)$ for a Casson fluid, then integrate your expression to obtain the flow as a function of pressure. You will find that $Q(\Delta p)$ is a nonlinear function, as will be the case for any non-Newtonian fluid.

[39] For steady, fully-developed flow in an axisymmetric pipe, we had that $p(r, \phi, x)$ was only a function of $x$, so there was no need to integrate out the transverse coordinates. But much like the 1D flow $Q$ is an integral of the 3D velocity field, we can obtain a 1D pressure $p(x,t)$ by averaging over the radial direction.

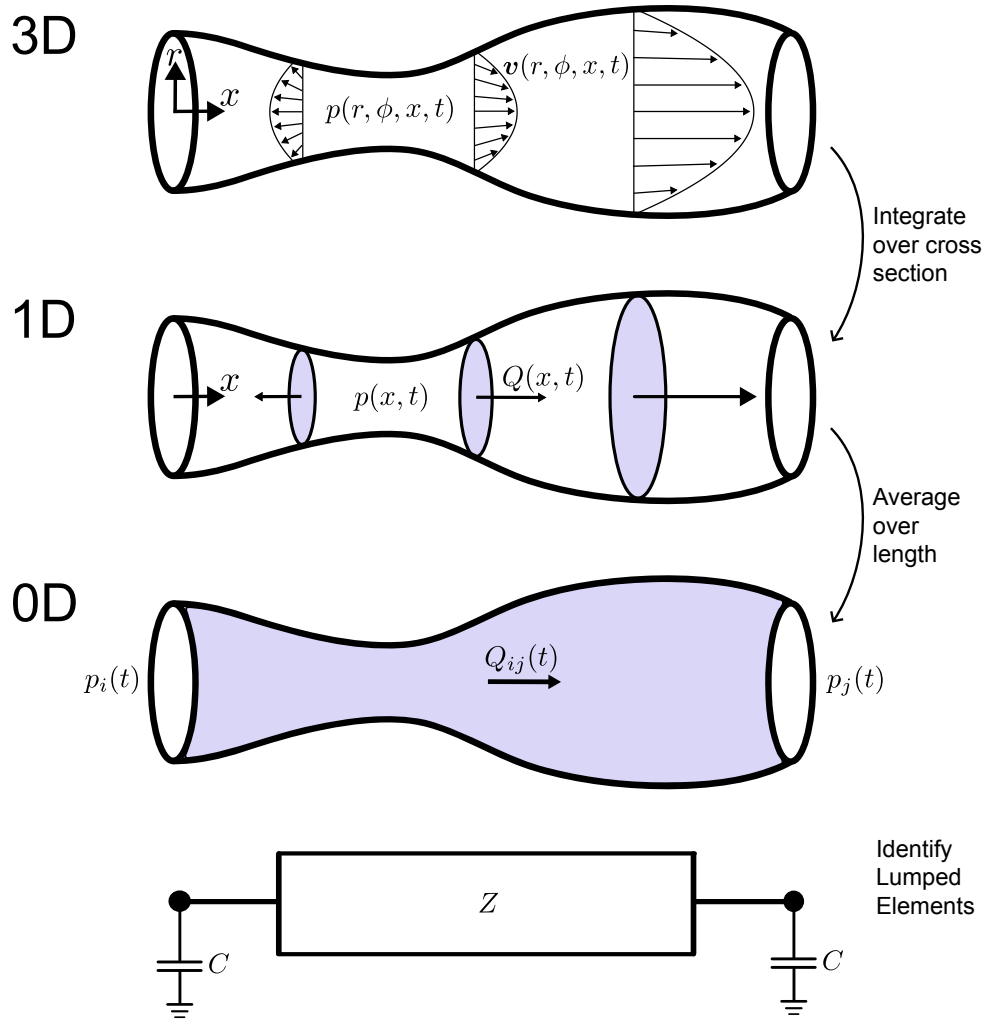



FIG. 8. Pulsatile flow in three different descriptions. Developing a network model from fundamental physics often involves integrating over dimensions. For a fluid flow network, one begins with 3D equations for mass continuity and momentum conservation in a single tube, integrates over the cross section to obtain 1D equations for the pressure and flow, and finally integrates over the length of the tube to obtain 0D equations for edge flow $Q_{ij}$ and node pressures $p_i, p_j$. For the case of pulsatile flow, the dynamics of the 3D fluid and tube are characterized by the viscosity, density, and vessel stiffness. Integrating out the spatial dynamics introduces lumped elements: the impedance (which accounts for viscous and inertial effects) and the node capacitance (which accounts for vessel compliance).

the wall velocity. We'll assume that the wall only moves in the radial direction so the length remains $\ell_0$. The no-slip condition reads:

$$v_x|_{r=r_0+u_r(x,t)} = 0, \qquad v_r|_{r=r_0+u_r(x,t)} = \frac{\partial u_r(x,t)}{\partial t}. \tag{74}$$

The flow is now a function of space and time. Multiplying equation (60) by $2\pi r$, integrating from 0 to $r_0 + u_r(x,t)$, and applying no-slip boundary conditions gives

$$\frac{\partial Q}{\partial x} + \frac{\partial}{\partial t}\left(\pi(r_0+u_r)^2\right) = 0. \tag{75}$$

For an elastic pipe, the radius will increase as the pressure of the fluid in the pipe increases. For small

deformations, this will typically look like Laplace's law [19]:

$$p(x,t) = \frac{Eh}{r_0^2} u_r(x,t), \tag{76}$$

where $E$ is the Young's modulus of the tube, $h$ is the tube thickness, and $r_0$ is the radius of the stress-free tube. Combining these equations and keeping only terms linear in the deformation, we have

$$\frac{\partial Q}{\partial x} + \frac{2\pi r_0^3}{Eh}\frac{\partial p}{\partial t} = 0. \tag{77}$$

Next, we'd like a simple form of the momentum equation (61). Assume that the rest radius of an edge $r_0$ remains much less than the edge length $\ell_0$ and the wavelength of the deforming pipe $\lambda$. When the transverse length scale is much less than the axial length scale, we can apply lubrication theory: A careful scaling analysis for $r_0/\ell_0 \ll 1$ (see [20]) reveals that even though the flow is not steady, the pressure still only depends on $x$, and the gradient $\partial_r v_x$ dominates the shear terms. Under the lubrication approximation (and assuming a Newtonian fluid), we have

$$\rho\left[\frac{\partial v_x}{\partial t} + v_x\frac{\partial v_x}{\partial x} + v_r\frac{\partial v_x}{\partial r}\right] = -\frac{\partial p}{\partial x} + \mu\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v_x}{\partial r}\right) \tag{78}$$

$$0 = \frac{\partial p}{\partial r}. \tag{79}$$

Finally, we will linearize our solutions about $Q = 0$, $u_r = 0$. Under these approximations, the integrated momentum equation becomes

$$\frac{\partial p}{\partial x} + \frac{\rho}{\pi r_0^2}\frac{\partial Q}{\partial t} + \frac{8\mu}{\pi r_0^4}Q = 0. \tag{80}$$

These equations resemble the equations for transmission lines, with vessel compliance $2\pi r_0^3/Eh$ playing the role of capacitance per length, inertia $\rho/\pi r_0^2$ playing the role of inductance per length, and viscosity $8\mu/\pi r_0^4$ playing the role of resistance per length. We wish to solve equations (77) and (80) in a network. There are two ways to proceed, as will be discussed in the next two subsections.

#### *1. Method 1: Solving Network PDEs by Discretizing Each Edge*

One simple way of solving PDEs on networks is to further discretize the network such that each edge in the original network is decomposed into many smaller edges satisfying simpler equations. This is, in practice, how general PDEs are solved computationally. Here, we will be taking full advantage of the correspondence between the incidence matrix and the discrete finite difference operator. Throughout this subsection, an edge will be understood to be an infinitesimal edge in the discretized graph.

On each edge, we can define the edge-averaged flow:

$$\bar{Q}_\mu(t) \equiv \frac{1}{\ell_\mu}\int_0^{\ell_\mu} Q_\mu(x,t)dx. \tag{81}$$

Furthermore, we can associate the spatial pressure at an endpoint of an edge with the pressure at a node. In particular,

$$p_\mu(\ell_\mu, t) - p_\mu(0,t) = (\hat{\Delta}\vec{p}(t))_\mu. \tag{82}$$

Integrating equation (80) over an edge and applying the previous two equations gives

$$-(\hat{\Delta}\vec{p})_\mu = \frac{\rho\ell_\mu}{\pi r_{0,\mu}^2}\frac{\partial \bar{Q}_\mu}{\partial t} + \frac{8\mu\ell_\mu}{\pi r_{0,\mu}^4}\bar{Q}_\mu. \tag{83}$$

Next, we want an integrated continuity equation. Equation (55) still holds, though we have to be careful what we mean. Because the flow now depends on $x$, KCL will depend on how we draw the Gaussian surface! Here's where our approximation comes in. We can place the discs of the Gaussian surface (see figure 4) at the midpoint of the edges. In general, the flow at the midpoint need not equal the edge-averaged flow. However, if the edges are infinitesimally small, then the flow at the midpoint of the tube is equal to the edge-averaged flow (with corrections that scale with the infinitesimal edge length). Thus, equation (55) holds with $\vec{Q}$ representing the edge-averaged flows $\vec{\bar{Q}}$. Note that, by our convention of enclosing half of each edge adjacent to node $j$ in our Gaussian surface, $\mathrm{V}_j$ is equal to half the sum of all edge volumes adjacent to node $j$:

$$\mathrm{V}_j = \frac{1}{2}\sum_\mu |\Delta^T|_{j\mu}\mathrm{V}_\mu. \tag{84}$$

Recall that the unoriented incidence matrix $|\hat{\Delta}^T|$ sums over all edge-valued quantities adjacent to a node. Lastly, we use Laplace's law (76) to relate the change in volume to the pressure. Although this relation holds at each $x$, we can replace $p_\mu(x,t)$ with $p_j(t)$ since any corrections will be higher order in the infinitesimal edge length:

$$\mathrm{V}_j = \frac{1}{2}\sum_\mu |\Delta^T|_{j\mu}\frac{2\pi r_{0,\mu}^3\ell_\mu}{Eh_\mu}p_j. \tag{85}$$

Combining this result with equation (55) gives

$$(\hat{\Delta}^T\vec{\bar{Q}})_j = \frac{1}{2}\sum_\mu |\Delta^T|_{j\mu}\frac{2\pi r_{0,\mu}^3\ell_\mu}{Eh_\mu}\frac{dp_j}{dt} + \vec{q}. \tag{86}$$

Defining

$$R_\mu = \frac{8\mu\ell_\mu}{\pi r_{0,\mu}^4}, \qquad L_\mu = \frac{\rho\ell_\mu}{\pi r_{0,\mu}^2}, \qquad C_i = \frac{1}{2}\sum_\mu |\Delta^T|_{j\mu}\frac{2\pi r_{0,\mu}^3\ell_\mu}{Eh_\mu}, \tag{87}$$

our equations reduce to the equations for RLC circuits, equations (40) and (41). This simple correspondence only holds on infinitesimal edges. Although equation (82) is exact for any edge length (given our convention that the edge flow is the edge-averaged flow), the continuity equation will, in general, be more complicated.

This method can be generalized to other PDEs by finding the equations to leading-order in small edge length. This is a practical method in cases where analytic solutions are unnecessary and runtime is not a concern. However, one must check that the edge lengths are sufficiently small that the finite difference scheme converges to the physical solution.

### *2. Method 2: Solving Network PDEs Explicitly*

For some simple 1D PDEs, we can explicitly solve for $p_\mu(x,t)$ and $Q_\mu(x,t)$. We will still define the 0D edge flow $\bar{Q}_\mu(t)$ as in equation (81), but $Q_\mu(x,t)$ may have some complicated spatial structure. For simplicity, we will assume that the viscous losses are negligible for the rest of this section (just to simplify our calculations). Further, assume that the system is driven sinusoidally at a single frequency $\omega$, so that we can Fourier transform the equations and look for the steady state. Tildes will denote functions in the frequency domain. Equation (82) still holds, which with our current simplifications reads:

$$\vec{\tilde{\bar{Q}}}(\omega) = -\hat{Z}(\omega)^{-1}\hat{\Delta}\vec{\tilde{p}}(\omega), \qquad Z_\mu(\omega) \equiv \frac{i\omega\rho\ell_\mu}{\pi r_{0,\mu}^2}. \tag{88}$$

We can attempt to calculate KCL as we did in the previous subsection. Using a Gaussian surface with discs located at the midpoint of each edge results in a KCL relating flows at the midpoints of the edges

$\tilde{Q}_\mu(\ell_\mu/2,\omega)$ incident to a node. However, since our edges are no longer infinitesimally small, we cannot equate $\tilde{Q}_\mu(\ell_\mu/2,\omega)$ with the edge-averaged flow $\tilde{\bar{Q}}_\mu(\omega)$. Along edge $e_\mu$, let $x=0$ be the tail node and $x=\ell_\mu$ be the head node. It is still true that fluid flow is conserved at a junction (assuming no flow sources/sinks):

$$\sum_{\mu\to j}\tilde{Q}_\mu(\ell_\mu,\omega)-\sum_{\nu\leftarrow j}\tilde{Q}_\nu(0,\omega)=0, \tag{89}$$

but the edge-averaged flow is NOT conserved! We need to express $\tilde{Q}_\mu(\ell_\mu,\omega)$ and $\tilde{Q}_\nu(0,\omega)$ in terms of $\tilde{\bar{Q}}_\mu(\omega)$ and $\tilde{\bar{Q}}_\nu(\omega)$. Finding an appropriate KCL for $\tilde{\bar{Q}}$ will require us to solve for the pressure and flow along the edge.

Let's forget about our network, for a moment, and solve our system of PDEs with Dirichlet boundary conditions. Decoupling equations (77) and (80) and dropping the viscous term gives a wave equation with speed $c$:

$$\frac{\partial^2 p}{\partial x^2}-\frac{1}{c^2}\frac{\partial^2 p}{\partial t^2}=0, \qquad c=\sqrt{\frac{Eh}{2\rho r_0}}. \tag{90}$$

This wave speed, called the Moens-Korteweg formula, has a long history in biomechanics [19]. The Fourier transformed pressure in a single tube satisfies

$$\tilde{p}(x,\omega)=\tilde{p}(0,\omega)\frac{\sin\left(\frac{\omega}{c}(\ell_0-x)\right)}{\sin\left(\frac{\omega}{c}\ell_0\right)}+\tilde{p}(\ell_0,\omega)\frac{\sin\left(\frac{\omega}{c}x\right)}{\sin\left(\frac{\omega}{c}\ell_0\right)}. \tag{91}$$

This is a promising equation for applying to networks. It involves the pressure at the endpoints of the tube which for a network would be the node pressures. Next, calculate the flows,

$$\tilde{Q}(x,\omega)=-\frac{\ell_0}{Z(\omega)}\frac{\partial\tilde{p}}{\partial x}=\frac{\omega\ell_0}{cZ(\omega)}\left[\tilde{p}(0,\omega)\frac{\cos\left(\frac{\omega}{c}(\ell_0-x)\right)}{\sin\left(\frac{\omega}{c}\ell_0\right)}-\tilde{p}(\ell_0,\omega)\frac{\cos\left(\frac{\omega}{c}x\right)}{\sin\left(\frac{\omega}{c}\ell_0\right)}\right] \tag{92}$$

Returning to equation (89),

$$\begin{aligned}
0&=\sum_{\mu\to i}\tilde{Q}_\mu(\ell_\mu,\omega)-\sum_{\nu\leftarrow i}\tilde{Q}_\nu(0,\omega)\\
&=\sum_{\mu\to i}\frac{\omega\ell_\mu/c_\mu}{\sin\left(\frac{\omega\ell_\mu}{c_\mu}\right)}\left[\frac{\tilde{p}_\mu(0,\omega)}{Z_\mu(\omega)}-\frac{\tilde{p}(\ell_\mu,\omega)}{Z_\mu(\omega)}\cos\left(\frac{\omega\ell_\mu}{c_\mu}\right)\right]-\sum_{\nu\leftarrow i}\frac{\omega\ell_\nu/c_\nu}{\sin\left(\frac{\omega\ell_\nu}{c_\nu}\right)}\left[\frac{\tilde{p}_\nu(0,\omega)}{Z_\nu(\omega)}\cos\left(\frac{\omega\ell_\nu}{c_\nu}\right)-\frac{\tilde{p}_\nu(\ell_\nu,\omega)}{Z_\nu(\omega)}\right]\\
&=\sum_{\mu\to i}\frac{\omega\ell_\mu/c_\mu}{\sin\left(\frac{\omega\ell_\mu}{c_\mu}\right)}\left[\tilde{\bar{Q}}_\mu(\omega)+\frac{\tilde{p}_i(\omega)}{Z_\mu(\omega)}\left(1-\cos\left(\frac{\omega\ell_\mu}{c_\mu}\right)\right)\right]-\sum_{\nu\leftarrow i}\frac{\omega\ell_\nu/c_\nu}{\sin\left(\frac{\omega\ell_\nu}{c_\nu}\right)}\left[\tilde{\bar{Q}}_\nu(\omega)-\frac{\tilde{p}_i(\omega)}{Z_\nu(\omega)}\left(1-\cos\left(\frac{\omega\ell_\nu}{c_\nu}\right)\right)\right]\\
&=\sum_\mu\hat{\Delta}^T_{i\mu}\frac{\omega\ell_\mu/c_\mu}{\sin\left(\frac{\omega\ell_\mu}{c_\mu}\right)}\tilde{\bar{Q}}_\mu(\omega)+\sum_\mu|\hat{\Delta}|^T_{i\mu}\frac{\omega\ell_\mu/c_\mu}{Z_\mu(\omega)\sin\left(\frac{\omega\ell_\mu}{c_\mu}\right)}\left(1-\cos\left(\frac{\omega\ell_\mu}{c_\mu}\right)\right)\tilde{p}_i(\omega)
\end{aligned} \tag{93}$$

In the third line, we applied equation (88) to express $\tilde{p}_\mu(0,\omega)$ in terms of $\tilde{\bar{Q}}_\mu(\omega)$ and $\tilde{p}_\mu(\ell_\mu,\omega)=\tilde{p}_i(\omega)$, and to express $\tilde{p}_\nu(\ell_\nu,\omega)$ in terms of $\tilde{\bar{Q}}_\nu(\omega)$ and $\tilde{p}_\nu(0,\omega)=\tilde{p}_i(\omega)$. Equation (93) is a continuity equation at the node purely in terms of the node pressures and edge-averaged flows. It is the appropriate generalization of KCL from the previous subsection to the case with finite edge lengths. The above equation, along with equation (88), define the network equations.[40]

---

[40] **EXERCISE**: By defining appropriate matrices on the edges and nodes, write this as a matrix equation. How would you solve for the pressures? Show that in the limit $\omega\ell_\mu/c_\mu\to 0$, your equation recovers the form from the previous subsection.

There are two advantages to solving the spatial dynamics of an edge by hand. One, it's computationally more efficient, at least in this case. By not having to employ finite differences, the Laplacian that we have to invert is much smaller. However, in some cases, the coefficients may involve complicated integrals with no analytic solution, and it's not clear that solving those integrals computationally saves runtime as compared with just employing finite differences. The second advantage is that we can probe the physics of a single edge. In this example, we notice something interesting. Whenever $\omega \ell_\mu / c_\mu$ is an integer multiple of $\pi$, there is a resonance in the pressure and flow. Although this result would have been observed computationally using the finite difference scheme, here we have an analytic form for the resonances. Furthermore, once the node pressures and edge-averaged flows are found, we can write down the full 1D dynamics of the edges using equations (91) and (92). The main drawback of this method is that even linear PDEs can be difficult to solve. Had we included the viscosity, the wave number becomes complex, and one has to take care handling the square roots of complex numbers [21]. Furthermore, dropping the lubrication assumption leads to flatter velocity profiles which can be expressed in terms of Bessel functions [18].

The bottom line is that if you can write down linear PDEs for the pressure and flow, there is probably a clever way to integrate the 1D equations into 0D network equations, using either of the two methods presented in this section. A visual summary is shown in figure 8. One can apply this technique to other linear PDEs like the heat equation.

## V. MECHANICAL NETWORKS

Mechanical networks come in many forms: balls attached by springs, lattice vibrations, filamentous networks, truss structures, etc. In any case, we wish to determine the displacement of the material elements when forces or displacements are applied. In the previous section, we saw that viscous fluid flow networks map directly onto resistive electric circuits. Mechanical networks possess a similar mapping, but with one key difference: each node possesses several degrees of freedom. Wherein an electrical (fluid) network, each node is characterized by its electric potential (pressure), a mechanical network is characterized by its node position or node velocity, one number for each dimension of the embedding space.

First, we will consider how momentum conservation gives rise to a KCL only if we use the Lagrangian coordinates of the edges. Next, we will see how static mechanical networks in one dimension map directly onto electronic circuits, but more interesting behavior can arise in higher dimensions. Lastly, we will show how dynamic networks can be mapped to RLC circuits. A complete discussion of mechanical networks and how they relate to general linear flow networks can be found in a new review paper [22], while their application to condensed matter physics is summarized in [23].

### A. Physical Networks are Lagrangian

A continuum elastic solid is described by its displacement field $\boldsymbol{u}(\boldsymbol{R}, t)$ from a rest configuration to the current configuration. In the context of elasticity theory, the coordinates $\boldsymbol{R}$ parameterizing the rest configuration are called material coordinates because they label a material element in the rest configuration. In general, coordinates that permanently label the particles of a material are called **Lagrangian**. This is to be contrasted with Eulerian coordinates $\boldsymbol{r}$ which label a particular point in space. For example, often when we describe fluids, we specify the velocity field $\boldsymbol{v}$ at a particular Eulerian point in space $\boldsymbol{r}$, not the velocity of individual parcels of fluid.

Physical networks are inherently Lagrangian. We measure quantities at a node or edge, not at a particular location in space. For a mechanical network, where the nodes are changing position with time, this is important to keep in mind.[41] Ideally, the higher-dimensional models used to derive the network equations should also be expressed in Lagrangian coordinates. In terms of the Lagrangian coordinates of the solid, momentum conservation takes the form

$$\boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \frac{\partial \boldsymbol{\rho}_{\mathbf{p}}}{\partial t}, \tag{94}$$

where $\boldsymbol{\rho}_{\mathbf{p}}$ is the momentum density.[42] Integrating this equation over a volume enclosing a junction at node $v_j$ in the rest configuration (fixed Lagrangian coordinates) gives rise to an integrated form of momentum conservation that functions as our KCL for a mechanical network. This is a fancy way of saying Newton's second law is satisfied at a node in a mechanical network. The "edge current" in this case is the axial tension

$$F_\mu \equiv \int_{S_\mu} \boldsymbol{n}_\mu \cdot \boldsymbol{\sigma} \cdot d\boldsymbol{A}. \tag{95}$$

Here, $S_\mu$ is a fixed cross-section in the rest configuration and $\boldsymbol{n}_\mu$ is the vector orientation of the edge.

[41] It may seem that this problem goes away for a network like an electric circuit where we are measuring flows through the network rather than flows of the network itself, but actually, the problem arises even there. When we integrated the continuous current density $\boldsymbol{J}$ across the Eulerian volume $\Omega$ in figure 4, we secretly assumed that the circuit was not moving. But, there is no reason we can't analyze a circuit as it is being transported across a lab. In that case, our Gaussian surface should follow the circuit as it is being transported. We only care about current through the wires. We do not care that charge is being transported across the lab as we move the circuit. The relevant current $I$ is to be measured with respect to fixed Lagrangian coordinates of the wire, not at fixed coordinates in space. This may all sound silly and pedantic, but there are cases where this subtlety is quite important, such as the fluid flow through a network undergoing longitudinal stretching [24].

[42] The stress that appears in equation (56) is the Cauchy stress tensor relating area elements in the current configuration to forces in the current configuration. The stress appearing in equation (94) is the first Piola-Kirchhoff stress tensor relating area elements in the rest configuration to forces in the current configuration. To linear order in small displacements, which is all we will consider here, the Cauchy stress and first Piola-Kirchhoff stress tensors are identical.

### B. Static Mechanical Networks in One Dimension

Before entering the uncharted territory of $d$-dimensional quantities on each node, let's formulate the problem in 1D, which maps directly to resistive circuits or viscous fluid flow networks.

One of the most famous equations in physics is Hooke's Law which states that the force in a spring is proportional to the change in length of the spring. Denote the rest length of the spring $l_0$, the current length $l$, the stiffness $K$, and the tension $F$. Then, Hooke's law on edge $e_\mu$ states

$$F_\mu = -K_{\mu\nu}(l_\nu - l_{0,\nu}). \tag{96}$$

In 1D, we can write the change in edge length as the difference in node displacements. If displacement of node $v_i$ is denoted $u_i$, then

$$l_{ij} - l_{0,ij} = u_j - u_i. \tag{97}$$

Thus, we have our linear response law on the edges:[43]

$$\vec{F} = -\hat{K}\hat{\Delta}\vec{u}\,. \tag{98}$$

KCL takes the form of force balance. Imagine that at the nodes, we apply some external force $\vec{f}$. Then, the force balance on the nodes has contributions from the external force and the adjacent springs:

$$\hat{\Delta}^T\vec{F} = \vec{f}. \tag{99}$$

At this point, you know what to do: decouple force balance and Hooke's Law, fix the position of at least one node, and invert the reduced Laplacian $(\hat{\Delta}^T\hat{K}\hat{\Delta})_{\mathcal{I}\mathcal{I}}$ to obtain the unknown node displacements. The null space of $\hat{\Delta}$ is one dimensional, and it is spanned by the vector with constant entries. Much like how we could shift the electric potential of all nodes without introducing any currents, we can uniformly translate all nodes without altering any edge tensions. In higher dimensions, there are more ways in which we can displace nodes without changing the edge lengths.

### C. Static Mechanical Networks in $d$ Dimensions

In higher dimensions, Hooke's law (96) is still true. However, it is no longer so simple to relate the edge lengths to the displacements. If the rest position vector of node $v_i$ is denoted $\boldsymbol{r}_i$, and the displacement is denoted $\boldsymbol{u}_i$, then we can write[44]

$$l_{0,ij} \equiv |\boldsymbol{r}_j - \boldsymbol{r}_i| = \sqrt{(\boldsymbol{r}_j - \boldsymbol{r}_i)\cdot(\boldsymbol{r}_j - \boldsymbol{r}_i)}. \tag{100}$$

$$l_{ij} \equiv |(\boldsymbol{r}_j + \boldsymbol{u}_j) - (\boldsymbol{r}_i + \boldsymbol{u}_i)| = \sqrt{(\boldsymbol{r}_j - \boldsymbol{r}_i + \boldsymbol{u}_j - \boldsymbol{u}_i)\cdot(\boldsymbol{r}_j - \boldsymbol{r}_i + \boldsymbol{u}_j - \boldsymbol{u}_i)}. \tag{101}$$

Taylor expanding to linear order in $\boldsymbol{u}$, we get

$$\begin{aligned} l_{ij} - l_{0,ij} &= \sqrt{l_{0,ij}^2 + 2(\boldsymbol{r}_j - \boldsymbol{r}_i)\cdot(\boldsymbol{u}_j - \boldsymbol{u}_i) + (\boldsymbol{u}_j - \boldsymbol{u}_i)\cdot(\boldsymbol{u}_j - \boldsymbol{u}_i)} - l_{0,ij} \\ &\approx \frac{(\boldsymbol{r}_j - \boldsymbol{r}_i)}{l_{0,ij}}\cdot(\boldsymbol{u}_j - \boldsymbol{u}_i). \end{aligned} \tag{102}$$

---

[43] It may be surprising that Hooke's law is the analogue to Ohm's Law when both resistance and viscosity represent dissipative physics, but Hooke's law does not. Indeed, the analogy to circuits is most apparent when considering velocity, not position, as the node degree of freedom, in which case the dashpot (which is a dissipative element) plays the role of resistance. We will elaborate on this analogy in a later section.

[44] Recall that boldface denotes vectors with Cartesian components. An arrowhead is reserved for vectors whose elements correspond to a particular node or edge. Using a boldface and an arrowhead simultaneously indicates a node/edge-valued quantity with Cartesian components.

Thus, for small displacements, Hooke's law can be written as

$$F_{ij} = -K_{ij}\frac{(\boldsymbol{r}_j - \boldsymbol{r}_i)}{l_{0,ij}} \cdot (\boldsymbol{u}_j - \boldsymbol{u}_i). \tag{103}$$

The node displacements are now vector quantities! Recognize that

$$\boldsymbol{n}_{ij} = \frac{(\boldsymbol{r}_j - \boldsymbol{r}_i)}{l_{0,ij}} \tag{104}$$

is just the unit vector along edge $(v_i, v_j)$ at rest. As an edge vector,

$$\vec{\boldsymbol{n}} = \hat{l}_0^{-1}\hat{\Delta}\vec{\boldsymbol{r}}, \qquad l_{0,\mu} = \sqrt{(\hat{\Delta}\vec{\boldsymbol{r}})_\mu \cdot (\hat{\Delta}\vec{\boldsymbol{r}})_\mu}, \tag{105}$$

where in the last expression, there is no summation over $\mu$. We will let $\hat{\boldsymbol{n}} = \mathrm{diag}(\vec{\boldsymbol{n}})$. Returning to equation (103), we have

$$F_\mu = -K_{\mu\nu}\boldsymbol{n}_{\nu\rho} \cdot (\hat{\Delta}\boldsymbol{u})_\rho = -K_{\mu\nu}\boldsymbol{n}_{\nu\rho}\Delta_{\rho j} \cdot \boldsymbol{u}_j. \tag{106}$$

Thus, the relevant matrix describing the topology of our mechanical network is $\hat{\boldsymbol{\Delta}} \equiv \hat{\boldsymbol{n}}\hat{\Delta}$. Throughout the mechanical networks literature, this matrix is known as the **compatibility matrix**.[45] With this definition, we can finally write the desired network equations

$$\vec{F} = -\hat{K}\hat{\boldsymbol{\Delta}} \cdot \vec{\boldsymbol{u}} \tag{107}$$

$$\hat{\boldsymbol{\Delta}}^T\vec{F} = \vec{\boldsymbol{f}}. \tag{108}$$

By now, you may be annoyed that we are treating $\hat{\boldsymbol{\Delta}}$ as a matrix when it is really a three-index tensor, where one index runs over the $d$ spatial indices, but we will now see how it can be handled as a matrix. Rather than thinking of $\vec{\boldsymbol{u}}$ as a vector of vectors, let's flatten the nested vectors into one big vector of size $d|\mathcal{V}|$. There are two convenient bases to work with: We may either write out all $d$ components of node 1 then all $d$ components of node 2..., or we may write the $x$-components of all $|\mathcal{V}|$ nodes then all $y$-components of all $|\mathcal{V}|$ nodes... We have found the latter to be more convenient. For example, in 2D, the compatibility matrix acting on the displacement vector can be simply written

$$\hat{\boldsymbol{\Delta}} \cdot \vec{\boldsymbol{u}} = \begin{bmatrix} \hat{n}_x\hat{\Delta} & \hat{n}_y\hat{\Delta} \end{bmatrix} \begin{bmatrix} \vec{u}_x \\ \vec{u}_y \end{bmatrix}. \tag{109}$$

Unlike the mechanical 1D network, the null space of $\hat{\boldsymbol{\Delta}}$ in 2D is not one dimensional! Physically, we have, at least, uniform translations in $x$, uniform translations in $y$, and rotations through the origin that do not alter the edge lengths.[46]. These are the trivial rigid body motions; however, we may have additional displacements that leave our edge lengths invariant, something which was impossible in 1d.

Let's now return to the general $d$-dimensional case. The compatibility matrix is a $|\mathcal{E}| \times d|\mathcal{V}|$ matrix. A **zero mode** is any solution $\vec{\boldsymbol{u}}$ to the equation[47]

$$\hat{\boldsymbol{\Delta}} \cdot \vec{\boldsymbol{u}} = \vec{0}. \tag{110}$$

If a zero mode is not one of the $d$ rigid-body translations or $d(d-1)/2$ rigid-body rotations, then we call the zero mode a **floppy mode**. Systems which do not have any floppy modes are called rigid. While this

---

[45] The transpose of the compatibility matrix $\hat{\boldsymbol{\Delta}}^T$ is known as the equilibrium matrix.

[46] **EXERCISE**: Show that $\vec{\boldsymbol{u}} = \begin{bmatrix} \vec{1} \\ \vec{0} \end{bmatrix}, \begin{bmatrix} \vec{0} \\ \vec{1} \end{bmatrix}$, and $\begin{bmatrix} -\vec{y} \\ +\vec{x} \end{bmatrix}$ all satisfy $\hat{\boldsymbol{\Delta}} \cdot \vec{\boldsymbol{u}} = \vec{0}$.

[47] More precisely, these are the linear zero modes. Recall, the form of the edge lengths we are using is correct only to linear order in the displacements. It may be that the node displacements satisfy (110), but the edge lengths change at a higher order in $\vec{\boldsymbol{u}}$. Similarly, when we use the term "rigid", we really mean "first-order rigid".

analysis holds for crystal structures in equilibrium, crystals are always rigid, so working in the static limit doesn't say much; rather, one would see how the solid responds to perturbations by finding the normal modes (phonons) of the system. For "soft" materials such as packed spheres, biopolymer networks, and glasses, the simple question of whether the system is rigid or not is an interesting one.

Alongside the zero modes, we can also define a **state of self stress** as a set of nonzero tensions which exerts no net forces on the nodes, that is a solution $\vec{F}$ to the equation

$$\hat{\mathbf{\Delta}}^T \vec{F} = \vec{\mathbf{0}}. \tag{111}$$

One can think of these states of self stress as cases where the system is over-constrained, so that even though the edges may be under tension, the excess bonds prevent the system from relaxing to a zero energy configuration. The most intuitive network that has a state of self stress is a 1D lattice with periodic boundary conditions. If each edge has an identical tension, then each node will be pulled equal amounts to the left and right such that the net force at each node is zero.

One can decouple equations (107) and (108):

$$\hat{\boldsymbol{\mathcal{L}}} \cdot \vec{\boldsymbol{u}} \equiv \hat{\mathbf{\Delta}}^T \hat{K} \hat{\mathbf{\Delta}} \cdot \vec{\boldsymbol{u}} = -\vec{\boldsymbol{f}}. \tag{112}$$

The operator $\hat{\boldsymbol{\mathcal{L}}}$ that resembles the Laplacian is called the dynamical matrix.[48] Inverting this matrix is the key step to solving mechanical networks. One must apply enough constraints to eliminate the zero modes. For 2d, this means fixing at least 3 displacements (ex: the $x$ and $y$ component of one node, and the $y$ component of a different node), and constraining any floppy modes.

Much like our analysis of circuits, we can write down an energy function

$$E = \frac{1}{2} \vec{\boldsymbol{u}}^T \cdot \hat{\boldsymbol{\mathcal{L}}} \cdot \vec{\boldsymbol{u}} + \vec{\boldsymbol{u}}^T \cdot \vec{\boldsymbol{f}} \tag{113}$$

whose minimization gives the physical displacements. We can interpret the Laplacian for this problem as a Hessian of the elastic energy. Furthermore, we can now interpret our zero modes as either vectors in the null space of $\hat{\mathbf{\Delta}}$ or as displacements which cost zero energy.

The zero modes and states of self stress can be counted using the Maxwell-Calladine index theorem [23]. We can prove the theorem just by applying the rank-nullity theorem twice, once to $\hat{\mathbf{\Delta}}$ and once to $\hat{\mathbf{\Delta}}^T$:

$$\text{rank}(\hat{\mathbf{\Delta}}) + \text{nullity}(\hat{\mathbf{\Delta}}) = 2|\mathcal{V}| \tag{114}$$

$$\text{rank}(\hat{\mathbf{\Delta}}^T) + \text{nullity}(\hat{\mathbf{\Delta}}^T) = |\mathcal{E}|. \tag{115}$$

Recalling that the rank of a matrix is equal to that of its transpose, we can subtract these two equations to get

$$N_0 - N_{ss} = 2|\mathcal{V}| - |\mathcal{E}|, \tag{116}$$

where $N_0$ is the number of zero modes and $N_{ss}$ is the number of states of self stress.

An example 2d static mechanical network is analyzed in figure 9. The Python code to generate the compatibility matrix and find its null space is shown below. Note that the basis vectors of the null space that Python returns may not be the intuitive vectors shown in figure 9.[49]

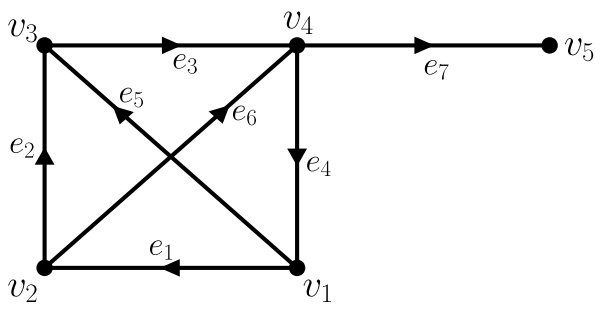
Listing 4. Analyzing a Mechanical Network

```
import networkx as nx
import numpy as np
```

[48] The name makes more sense when we actually study dynamics.

[49] **EXERCISE**: Calculate the zero modes for the left network shown in figure 5, and classify them physically. Are there any states of self stress?

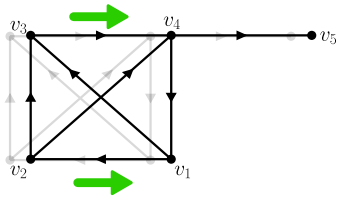
Static Mechanical Network



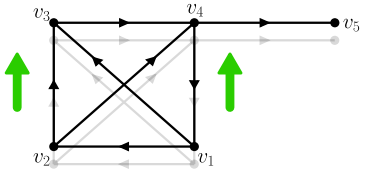

$$\hat{n}_x = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{array} \\ \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}, \quad \hat{n}_y = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}, \quad \hat{\Delta} = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{pmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix} \end{array}$$

$$\hat{\mathbf{\Delta}} = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{array} \begin{array}{c} \begin{array}{cccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & y_1 & y_2 & y_3 & y_4 & y_5 \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

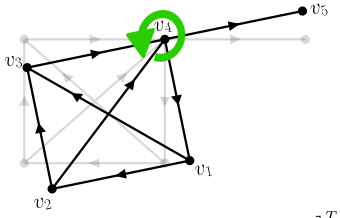
Trivial zero modes (rigid body motion)




$$\vec{\mathbf{u}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

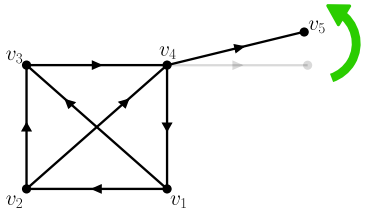



$$\vec{\mathbf{u}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$$



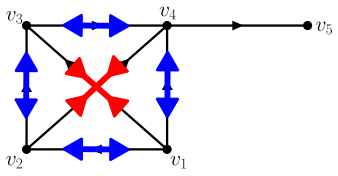

$$\vec{\mathbf{u}} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 \end{bmatrix}^T$$

Nontrivial zero mode (floppy mode)




$$\vec{\mathbf{u}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T$$

State of Self Stress




$$\vec{\mathbf{f}} = \begin{bmatrix} 1 & 1 & 1 & 1 & -\sqrt{2} & -\sqrt{2} & 0 \end{bmatrix}^T$$

FIG. 9. Mechanical network example. The compatibility matrix $\hat{\mathbf{\Delta}}$ is calculated from the incidence matrix $\hat{\Delta}$ along with the matrices $\hat{n}_x$ and $\hat{n}_y$ encoding edge directions: $\begin{bmatrix} \hat{n}_x\hat{\Delta} & \hat{n}_y\hat{\Delta} \end{bmatrix}$. The zero modes satisfy $\hat{\mathbf{\Delta}} \cdot \vec{\boldsymbol{u}} = \vec{0}$ while the state of self stress satisfies $\hat{\mathbf{\Delta}}^T \vec{F} = \vec{\mathbf{0}}$. In two dimensions, there are three trivial zero modes: two translations and one rotation. The floppy mode corresponds to independently rotating node 5. The state of self stress occurs when the edges on the perimeter of the square are in extension while the crossbars are in compression. Note that (116) is satisfied.

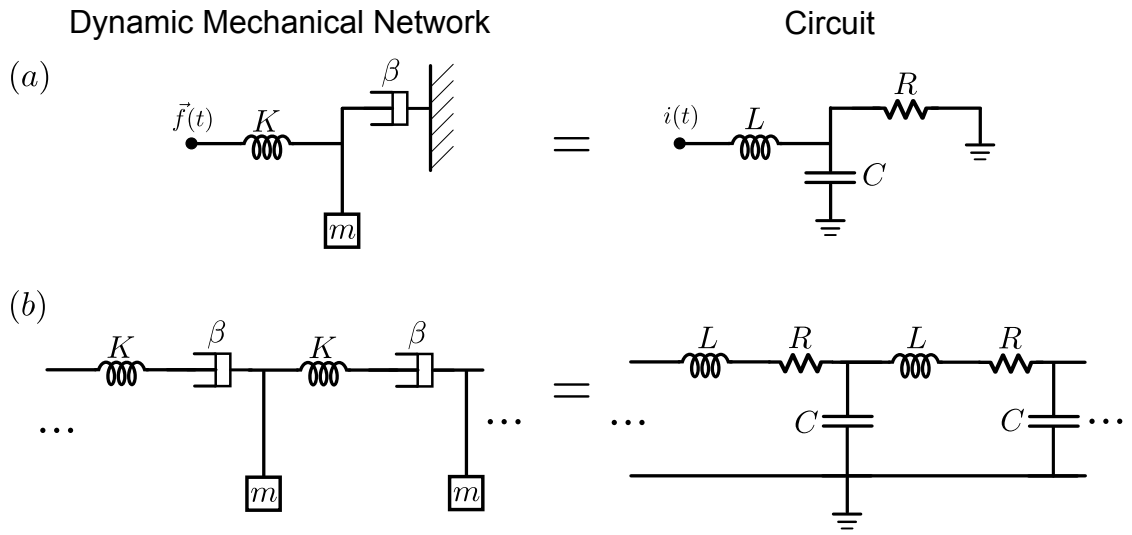



FIG. 10. Demonstration of the correspondence between an electrical circuit and a dynamic mechanical network with two examples. In the mapping described by equations (117), (118), and (119), the dashpot $\beta$ plays the role of the resistor $R$, the spring $K$ plays the role of the inductor $L$, and the mass plays the role of the capacitor $C$. Many other such mappings exist [22].

```
import scipy.sparse as sp
from scipy.linalg import null_space

G = nx.DiGraph()
ordered_nodes = [1, 2, 3, 4, 5]
ordered_edges = [(1,2),(2,3),(3,4),(4,1),(1,3),(2,4),(4,5)]
G.add_nodes_from(ordered_nodes)
G.add_edges_from(ordered_edges)
restPos_dict = {1: (0,0), 2: (-1,0), 3: (-1,1), 4: (0,1), 5: (1,1)}
nx.set_node_attributes(G, restPos_dict, "restPos")
DeltaT = nx.incidence_matrix(G, oriented=True,
                             nodelist=ordered_nodes, edgelist=ordered_edges)
Delta = DeltaT.T
restPos = np.array([restPos_dict[n] for n in G.nodes()])
edgeLengthVector = Delta @ restPos
DXVec = edgeLengthVector[:,0]
DYVec = edgeLengthVector[:,1]
l0Vec = np.sqrt(DXVec**2 + DYVec**2)
nXVec = DXVec/l0Vec
nYVec = DYVec/l0Vec
CompatibilityMatrix = sp.hstack([sp.diags(nXVec)@Delta, sp.diags(nYVec)@Delta])
print(CompatibilityMatrix.toarray())
print(null_space(CompatibilityMatrix.toarray()))
print(null_space(CompatibilityMatrix.toarray().T))
```

### D. Dynamic Mechanical Networks

There are many ways in which we can map mechanical networks to circuits [22], though when dynamics are involved, the analogy is most apparent if we study the velocity of the nodes rather than the displacement:

$$\psi_i - \psi_j = R_{ij} I_{ij} \iff v_i - v_j = \frac{1}{\beta_{ij}} F_{ij} \tag{117}$$

$$\psi_i - \psi_j = -L_{ij} \frac{dI_{ij}}{dt} \iff v_i - v_j = -\frac{1}{K_{ij}} \frac{dF_{ij}}{dt} \tag{118}$$

$$i_j = C_j \frac{d\psi_j}{dt} \iff m_j \frac{dv_j}{dt} = f_j \tag{119}$$

The first equation describes frictional damping along an edge. The second equation describes Newton's second law at a node. The third equation is the time derivative of Hooke's law from the previous sections. Some example mappings are shown in figure 10. Of course, this analogy only works in 1D where velocity is a scalar. Combining the techniques for RLC circuits introduced in section III.F with those to analyze vector-valued node quantities in the previous section, you can analyze dynamic mechanical networks. Furthermore, spatial dynamics such as wave propagation can be accounted for by using the technique described in section IV.C [25].

## VI. RANDOM WALKS ON GRAPHS

The theory of Markov chains and random walks is a thoroughly studied branch of mathematics that we will briefly review here. Some reviews on the topic, as applied to graphs, can be found in [26, 27]. In particular, the connection to circuit theory was studied by Doyle and Snell in [28].

A Markov chain is any process which can be described as a sequence of states such that the current state only depends on the previous state. We can represent each state with a node and each transition with a directed weighted edge. If the weight of edge $e_{ij}$ is $w_{ij}$, then the **transition probability** from $i$ to $j$ is

$$P_{ij} = \frac{w_{ij}}{\sum_j w_{ij}}, \tag{120}$$

where the denominator ensures that the probability to transition from $i$ to somewhere including $i$ is 1:

$$\sum_j P_{ij} = 1. \tag{121}$$

Thus, much like the Laplacian matrix, a probability matrix always has a $\vec{1}$ right eigenvector. Note that even if $\hat{W}$ is symmetric, $\hat{P}$ will typically not be symmetric, so the left and right eigenvectors are not the same.

We will specifically consider the problem of random walkers, though the math can easily be adapted to other Markov processes. Consider a random walker that when released, moves randomly along the edges of a graph. Let $p_i(k)$ denote the probability that a random walker is present at node $i$ at time step $k$. We have a recursion relation relating $p_i(k)$ at different time steps:

$$p_j(k+1) = \sum_i p_i(k) P_{ij}. \tag{122}$$

In matrix form,

$$\vec{p}^{\,T}(k+1) = \vec{p}^{\,T}(k) \hat{D}_-^{-1} \hat{W} \tag{123}$$

where $\hat{D}_- = \text{diag}\{\sum_j w_{ij}\}$ is the diagonal matrix of outdegrees and $\hat{W}$ is the adjacency matrix with elements $w_{ij}$.[50] Given some initial condition $\vec{p}(0)$, we can solve this equation iteratively to obtain:

$$\vec{p}^{\,T}(k) = \vec{p}^{\,T}(0) \left(\hat{D}_-^{-1} \hat{W}\right)^k. \tag{124}$$

For each iteration, the operator $\hat{D}_-^{-1}\hat{W}$ causes the solution to spread out. Intuitively, the distribution of walkers diffuses until reaching a steady state $\vec{p}_*^{\,T}$ satisfying

$$\vec{p}_*^{\,T} \hat{P} = \vec{p}_*^{\,T}. \tag{125}$$

For a strongly connected graph,[51] there is a unique steady state.[52] That's not to say that the walker actually reaches this steady state at late times, as the walker may get trapped in a cycle. The exact late-time behavior may depend on the initial conditions.

---

[50] It may look weird that we are left-multiplying our matrices, but this is essential to be consistent with how we defined $P_{ij}$. Some authors define the transition matrix to be the transpose of this which has the advantage that we can multiple probability vectors from the right, but it has the disadvantage that we need to read our expressions from right to left. Be sure to check which convention is being used when encountering such matrices!

[51] A **strongly connected** directed graph has the property that every node can be reached by every other node along directed edges. A connected undirected graph is also strongly connected, but a connected directed graph may not be strongly connected.

[52] **EXERCISE**: Show that for an undirected network, $p_{*,i} = d_i / \sum_i d_i$, where $d_i = \sum_j w_{ij}$ is the weighted degree of the node.

### A. Escape probability

What is the probability $x_i$ that a random walker (drunken sailor) starting at node $i$ reaches node $N$ (home) before it reaches node 1 (bar)? It is easy to see that $x_N = 1$, as the sailor starts the random walk already home, and $x_1 = 0$, as the walker is already at the bar and has no chance to get home.

As a self-consistency condition,

$$x_i = \sum_j P_{ij} x_j \tag{126}$$

where the right-handside encodes all of the locations $j$ the walker could be at after one step, weighted by the probability of hopping from $i$ to $j$. In matrix form, this can be written

$$\vec{x} = \hat{P}\vec{x} = \hat{D}_-^{-1}\hat{W}\vec{x} \implies (\hat{D}_- - \hat{W})\vec{x} = \vec{0}\,. \tag{127}$$

If our network is undirected, then this is equivalent to solving the circuit/flow problem with Dirichlet boundary conditions at 1 and $N$.

An absorbing state in a directed graph is a node whose outdegree is zero. Consider a graph which has the property that from any initial node, it is possible for our walker to reach an absorbing state.[53] Generalizing the idea of the drunken sailor, we see that absorbing states can be incorporated as boundary conditions to a Dirichlet problem. The interior nodes are all non-absorbing nodes. Using the method and notation introduced in section 3, we have that

$$\vec{x}_{\mathcal{I}} = -(\hat{\mathbb{P}}_{\mathcal{I}}(\hat{D}_- - \hat{W})\hat{\mathbb{P}}_{\mathcal{I}}^T)^{-1}\hat{\mathbb{P}}_{\mathcal{I}}(\hat{D}_- - \hat{W})\hat{\mathbb{P}}_{\mathcal{B}}^T\vec{x}_{\mathcal{B}}. \tag{128}$$

Plugging in $x_{B,j} = \delta_{jk}$ gives the probability to reach node $k$ (home) before reaching the other absorbing boundaries (bars).[54]

### B. Travel time

A related problem is to find the mean first-passage time. Define the mean first-passage time $h_i$ as the expected number of steps to go from node $i$ to node 1. The time $h_i$ is equal to the sum of the neighbor's mean first-passage times weighted by the transition probability $P_{ij}$ plus 1 for the extra time step:

$$h_i = \sum_j P_{ij} h_j + 1 \tag{129}$$

or in matrix form

$$\vec{h} = \hat{D_-}^{-1}\hat{W}\vec{h} + \vec{1} \Rightarrow (\hat{D_-} - \hat{W})\vec{h} = \vec{d}_-. \tag{130}$$

This is identical to a flow problem with distributed node currents $-\vec{d}$ and node 1 grounded. Three other methods for calculating the mean first-passage times are summarized in [27] and extensively studied in [29].

## VII. CONCLUDING REMARKS

In this tutorial, we showed how electric circuits, fluid flow networks, and mechanical networks are united by conservation laws and linear response. Whenever a transported quantity satisfies a continuity equation,

---

[53] An ergodic set as consisting of nodes for which our walker can completely explore all nodes in the set, but cannot leave the set [27]. Absorbing states are ergodic sets consisting of a single node. We are excluding graphs which have ergodic sets consisting of multiple nodes for simplicity.

[54] **EXERCISE**: Show that the operator in front of $\vec{x}_{\mathcal{B}}$ can be rewritten as $\sum_{n=0}^{\infty}(\hat{\mathbb{P}}_{\mathcal{I}}\hat{P}\hat{\mathbb{P}}_{\mathcal{I}}^T)^n\hat{\mathbb{P}}_{\mathcal{I}}\hat{P}\hat{\mathbb{P}}_{\mathcal{B}}^T$. Each operator in this sum can be interpreted as taking $n$ steps within the interior, then one step from the interior to the boundary.

we can write down a law analogous to Kirchhoff's Current Law. For most of the tutorial, we worked with the node analysis where the edge current is directly proportional to the gradient of a potential function on the nodes. This combines Kirchhoff's Voltage Law, which concerns the existence of such a potential function, with Ohm's Law, which encodes the physics of linear response (conductance, viscosity, damping, etc.). A summary of the connection between electrical networks, fluidic networks, and mechanical networks is given in table III. Although not covered in this tutorial, it has long been appreciated that problems concerning random walkers can also be mapped onto linear flow networks [28].

We showed that the node-edge incidence matrix $\hat{\Delta}^T$ is fundamental to understanding both the topology of oriented graphs and the physics defined on them. The incidence matrix functions both as a boundary operator (when acting on "chains") and a derivative operator (when acting on "cochains"). Physical networks have edge weights which encode the conductance of a linear response. From the incidence matrix and the conductance, one can build a Laplacian matrix $\hat{\mathcal{L}} = \hat{\Delta}^T \hat{\kappa} \hat{\Delta}$. Solving linear flow networks usually amounts to inverting a Laplacian. In its simplest form, the Laplacian is not invertible, but once we apply constraints (boundary conditions) on the nodes, we can uniquely solve for the potential on all interior nodes. This technique can even be generalized to cases where vector quantities are defined on the nodes by using the compatibility matrix $\hat{\mathbf{\Delta}}^T$ rather than the incidence matrix. For more information on how linear flow networks are united by linear algebra with particular focus on mechanical networks, we highly recommend a recent review paper by Ortiz-Tavárez, et al. [22]. We hope our work supplements this and other existing work by providing explicit examples and Python code for students and researchers beginning to study physical networks.

We stressed that one should understand the underlying (often higher-dimensional) physics of the edges before solving the network equations. Begin with fundamental conservation laws, incorporate constitutive relations, integrate out spatial dimensions, and hope that you are left with something linear. In many cases, there is a nonlinear response between the current and potential difference across an edge (ex: diodes, turbulent flow, non-Hookean solids). Although the graph topology is still completely encoded by the incidence matrix, we can no longer invert a Laplacian to solve the problem. Perhaps the most commonly used method for solving such systems is to use Newton's method. When applied to networks, this amounts to iteratively solving many linear problems, each step of which amounts to inverting the Jacobian of the nonlinear function (analogous to the Laplacian). Kirchhoff's Laws are unchanged. Thus, the techniques in this paper can even be used as a starting point for understanding how to solve nonlinear networks.

| | **Electrical** | **Fluidic** | **Mechanical** |
|---|---|---|---|
| **Transported quantity** | Charge Q | Volume V | Momentum $\mathbf{p}$ |
| **Edge Current** | Current $I$ | Flow $Q$ | Tension $F$ |
| **Node Potential** | Potential $\psi$ | Pressure $p$ | Node velocity $\boldsymbol{v}$ |
| **Edge Linear Response I** | Resistance $\sim \Delta\psi/I$ | Viscosity $\sim \Delta p/Q$ | Damping $\sim \Delta\boldsymbol{v}/F$ |
| **Edge Linear Response II** | Inductance $\sim \Delta\psi/\frac{dI}{dt}$ | Inertia $\sim \Delta p/\frac{dQ}{dt}$ | Elasticity $\sim \Delta\boldsymbol{v}/\frac{dF}{dt}$ |
| **Node Linear Response** | Capacitance $\sim$ Q$/\psi$ | Compliance $\sim$ V$/p$ | Inertia $\sim \mathbf{p}/v$ |

TABLE III. Summary of physical quantities relevant for three types of networks studied in this tutorial. In the case of mechanical networks, only the dynamical networks considered in the last section are displayed.

## ACKNOWLEDGMENTS

We'd like to thank Phil Nelson, Felipe Martins, Niranjan Sarpangala, Adam Kline, Marcelo Guzman, Suman Kulkarni, and the PHYS 5570 Spring 2024 class for useful discussions and feedback on the draft.

This is hopefully the first of multiple iterations of this tutorial, so we welcome comments.

# VIII. APPENDIX

## A. Perron-Frobenius Theory

The Perron-Frobenius Theorem gives us some useful information for the eigenvalues and eigenvectors of the sorts of matrices we encounter in graph theory. We will present the theorem in a way that is most intuitive for those studying networks [30]. A more thorough review of the spectral theory of graphs can be found in [2].

First, note that we can associate any $N \times N$ matrix with positive coefficients $\hat{W}$ with the adjacency matrix of a directed graph $G(\hat{W})$.

The **Perron-Frobenius Theorem** states that for any matrix $\hat{W}$ with non-negative entries, if $G(\hat{W})$ is strongly connected, there exists a positive real eigenvalue $\lambda_1$ such that for any eigenvalue $\lambda_i$ of $\hat{W}$, $|\lambda_i| \leq \lambda_1$. The algebraic multiplicity of $\lambda_1$ is one, and the eigenvector $\vec{v}_1$ corresponding to the eigenvalue $\lambda_1$ can be made to have purely positive entries. Any other eigenvector which has purely non-negative entries is a scalar multiple of $\vec{v}_1$.

If in addition, $\hat{W}$ is symmetric (or equivalently, $G$ is undirected), we know that the eigenvalues are real, so we can order the eigenvalues $\lambda_1 > \lambda_2 \geq \lambda_3 ... \geq \lambda_N \geq -|\lambda_1|$. In particular, there is always a spectral gap $\lambda_1 - \lambda_2 > 0$.

For a matrix whose only negative entries lie on the diagonal, we can simply add a sufficiently large positive number times the identity matrix, then apply Perron-Frobenius. For example, the negative of the Laplacian matrix for a connected undirected graph has only negative entries on its diagonal, so we can add the maximum degree $d_{max}$: $-\hat{\mathcal{L}} + d_{max}\hat{\mathbb{1}} = \hat{W} + d_{max}\hat{\mathbb{1}} - \hat{D} > 0$. The unique positive eigenvector of $-\hat{\mathcal{L}} + d_{max}\hat{\mathbb{1}}$ is $\vec{1}$ with eigenvalue $d_{max}$ which is the largest eigenvalue of this matrix. Thus, the smallest eigenvalue of $\hat{\mathcal{L}}$ is zero with eigenvector $\vec{1}$, which is the only purely positive eigenvector.

## B. Harmonic functions

Suppose we wish to solve the linear flow network problem with $i_j = 0$ for each $v_j \in \mathcal{I}$:

$$(\hat{\mathcal{L}}\vec{\psi})_j = 0. \tag{131}$$

This is the discrete analogue to the continuous Laplace's equation:

$$\nabla^2 \psi(\vec{x}) = 0. \tag{132}$$

Functions satisfying Laplace's equation are called **harmonic functions**. Harmonic functions have several useful identities with direct analogues in the discrete case [4, 28]. The **mean-value property** equates the value of a harmonic function averaged on the surface of a sphere to its value at the center of the sphere. Its discrete analogue can be written[55]

$$\psi_i = \frac{1}{d_i} \sum_{j \in \mathcal{V}} W_{ij} \psi_j, \tag{133}$$

valid for any $\psi_i \in \mathcal{I}$. In words, the value of the potential at a node is equal to the weighted average of the potential at all of the adjacent nodes. Harmonic functions also satisfy the **minimum (maximum) principle** which states that $\psi$ takes its minimum (maximum) value on the boundary: For each $i \in \mathcal{I}$,

$$\min_{j \in \mathcal{B}} \psi_j \leq \psi_i \leq \max_{j \in \mathcal{B}} \psi_j. \tag{134}$$

Furthermore, equality in either of these expressions implies that $\psi_i$ is constant throughout $\mathcal{I}$.

---

[55] **EXERCISE**: Prove the mean-value property for networks using the definition of $\hat{\mathcal{L}}$. Then, use it to prove the minimum (maximum) principle for networks.

[1] Robin J. Wilson. *Introduction to Graph Theory Fourth edition*. Addison Wesley Longman Limited, fourth edition edition, 1996.

[2] Piet Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, Cambridge, UK; New York, 2011.

[3] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[4] Leo J. Grady and Jonathan Polimeni. *Discrete calculus: applied analysis on graphs for computational science*. Springer, London; New York, 2010.

[5] Mathieu Desbrun, Eva Kanso, and Yiying Tong. Discrete Differential Forms for Computational Modeling. Technical report, Applied Geometry Lab, Caltech, 2006.

[6] Mikio Nakahara. *Geometry, Topology and Physics*. Taylor & Francis, 2nd edition, 2003.

[7] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, 1986.

[8] Daniel Spielman. *Spectral and Algebraic Graph Theory (Draft)*. Yale University, 2019.

[9] Florian Dörfler and Francesco Bullo. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(1):150–163, 2013.

[10] J David Irwin and R Mark Nelms. *Basic engineering circuit analysis*. John Wiley & Sons, 2020.

[11] D. J. Klein and M. Randić. Resistance distance. *Journal of Mathematical Chemistry*, 12(1):81–95, December 1993.

[12] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, October 1992.

[13] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. Westview Press, 2nd edition, 2015.

[14] Swarnavo Basu and Karen Alim. Les houches lectures on flow networks in biology. *SciPost Physics Lecture Notes (preprint)*, 2025. Preprint version scipost_202508_00076v1.

[15] Stuart Childress. *An Introduction to Theoretical Fluid Mechanics*, volume 19 of *Courant Lecture Notes in Mathematics*. American Mathematical Society, 2009.

[16] David A. Rubenstein, Wei Yin, and Mary D. Frame. *Biofluid mechanics: an introduction to fluid mechanics, macrocirculation, and microcirculation*. Academic press series in biomedical engineering. Elsevier/AP, Academic Press is an imprint of Elsevier, Amsterdam; Boston, second edition edition, 2015.

[17] Yuan-Cheng Fung. *Biomechanics: Motion, Flow, Stress, and Growth*. Springer, 2nd edition, 1997.

[18] M. Zamir. *The Physics of Pulsatile Flow*. Springer New York, 2000.

[19] Yuan-Cheng Fung. *Biomechanics: Circulation*. Springer, 2nd edition, 1997.

[20] Behrouz Tavakol, Guillaume Froehlicher, Douglas P. Holmes, and Howard A. Stone. Extended lubrication theory: Improved estimates of flow in channels with variable geometry. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2206), 10 2017.

[21] Sean Fancher and Eleni Katifori. Mechanical response in elastic fluid flow networks. *Physical Review Fluids*, 7(1), 1 2022.

[22] José M. Ortiz-Tavárez, William Stephenson, and Xiaoming Mao. A unified model for linear responses of physical networks. arXiv preprint, 2025.

[23] T. C. Lubensky, C. L. Kane, Xiaoming Mao, A. Souslov, and Kai Sun. Phonons and elasticity in critically coordinated lattices. *Reports of Progress in Physics*, 78(7):073901, 2015.

[24] Aaron Winn and Eleni Katifori. The geometry of contraction-induced flows. *In progress*, 2025.

[25] Sean Fancher, Niranjan Sarpangala, Prashant Purohit, and Eleni Katifori. An efficient spectral method for the dynamic behavior of truss structures. *ArXiv*, 2025.

[26] D. Aldous and J. A. Fill. *Reversible Markov Chains and Random Walks on Graphs*. Unfinished Monograph, 2014.

[27] Naoki Masuda, Mason A. Porter, and Renaud Lambiotte. Random walks and diffusion on networks. *Physics Reports*, 716–717:1–58, November 2017.

[28] Peter G. Doyle and J. Laurie Snell. *Random walks and electric networks*. Carus mathematical monographs. Mathematical Association of America, Washington, D.C., 1984.

[29] Sidney Redner. *A Guide to First-Passage Processes*. Cambridge University Press, 2001.

[30] John C. Baez and Jacob D. Biamonte. *Quantum techniques in stochastic mechanics*. World Scientific Publishing Co., 2018.