UNIVERSITY OF CALIFORNIA
SANTA CRUZ

GRAPH CURVATURE FOR COVID-19 NETWORK RISK ANALYTICS

A thesis submitted in partial satisfaction of the
requirements for the degree of

MASTER OF SCIENCE

in

SCIENTIFIC COMPUTING AND APPLIED MATHEMATICS

by

Qingyuan Cui

September 2021

The thesis of Qingyuan Cui
is approved:

Professor Abhishek Halder

Professor Qi Gong

Professor Seshadhri Comandur

Peter F. Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Graph Curvature for COVID-19 Network Risk Analytics

by

Qingyuan Cui

Curvature of a smooth manifold is quite intuitive, and has been studied in differential geometry for a long time. However, the notion of curvature for metric spaces in general, and for graphs in particular, is a relatively recent idea. In 2015, graph Ricci curvature was introduced as a framework to consider neighborhood to neighborhood interactions within a weighted undirected graph. In this thesis, we generalize graph Ricci curvature for weighted directed graphs, and apply this notion to analyze the spread of the Coronavirus disease 2019 (COVID-19) across the counties in the state of California.

We use real data for the daily traffic across different counties in California, and the daily COVID-19 case counts from March 2020 to March 2021. We demonstrate that graph Ricci curvature, and curvatures derived from it–such as graph scalar curvature–are particularly suited to dynamically predict and locate the onset and intensity of virus spread. The outcome of this thesis is a novel geometric data-driven risk analytics methodology to identify time-varying network-level risks for a virus spread. We envisage that our ideas will be useful for designing dynamic nonpharmaceutical intervention (NPI) strategies across the network to optimally mitigate the spread of the virus.

Acknowledgments

I thank Professor Abhishek Halder for his detailed and patient guidance. I also thank Karthik Sivaramakrishnan for discussion during the early phase of this research.

Chapter 1

Preliminaries on Graphs

In this introductory Chapter, we review some preliminaries on graphs, and introduce the county-level traffic graph that will be useful in the sequel.

1.1 Graphs

A *graph* $\mathcal{G}$ is specified by pair of sets: a set of vertices $\mathcal{V}$, and a set of edges $\mathcal{E}$. We write $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Figure 1.1 shows an unweighted undirected graph with the vertex set $\mathcal{V} = \{a, b, c\}$ and the edge set $\mathcal{E} = \{ab, ac, bc\}$. Because this graph is undirected, meaning the edges have no directionality associated with them, hence ab and ba are referring to the same edge, that is, $ab \equiv ba$.

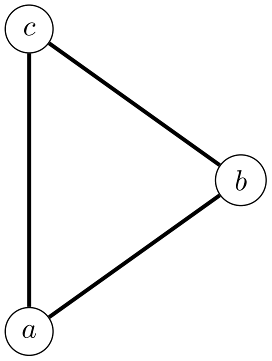


Figure 1.1: An unweighted undirected graph with three vertices a, b, c.

A *directed graph* is a graph whose all edges are directed. Each directed edge encodes two information: the connectivity between the two associated vertices, and the direction of that connectivity. Figure 1.2 shows an unweighted directed graph with the vertex set $\mathcal{V} = \{a, b, c\}$ and the edge set $\mathcal{E} = \{ab, ac, bc, cb\}$.

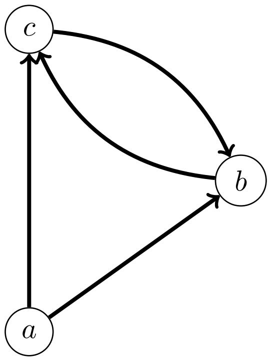


Figure 1.2: An unweighted directed graph with three vertices a, b, c.

A *weighted graph* $\mathcal{G}$ is a triple: a set of vertices $\mathcal{V}$, a set of edges $\mathcal{E}$, and a weight function $w : \mathcal{E} \mapsto \mathbb{R}$. For a weighted graph $\mathcal{G}$, we write $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$. Figure 1.3 shows a weighted undirected graph with the vertex set $\mathcal{V} = \{a, b, c\}$, the edge set $\mathcal{E} = \{ab, ac, bc\}$, and the weights $w(ab) = 4, \quad w(bc) = 6, \quad w(ac) = 12.$

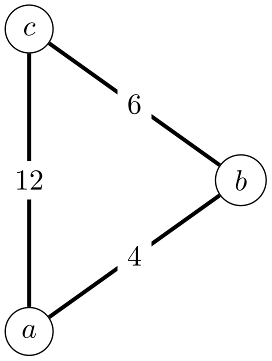


Figure 1.3: A weighted undirected graph with three vertices a, b, c.

A *weighted directed graph* is a weighted graph whose all edges are directed. Figure 1.4 shows a weighted directed graph with the vertex set $\mathcal{V} = \{a, b, c\}$, the edge set $\mathcal{E} = \{ab, bc, ac, cb\}$, and the weights $w(ab) = 4, \quad w(bc) = 6, \quad w(ac) = 12, \quad w(cb) = 23$.

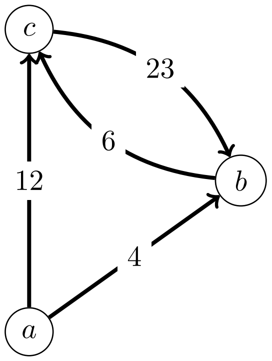


Figure 1.4: A weighted directed graph with three vertices a, b, c.

1.2 Adjacency Matrices for Weighted Directed Graphs

The *adjacency matrix* of a weighted directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, w)$ with $\mathcal{V} = \{v_1, v_2, v_3, ..., v_{|\mathcal{V}|}\}$, is denoted as $A \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ where $|\mathcal{V}|$ denotes the cardinality of the vertex set $\mathcal{V}$. Specifically, the adjacency matrix A is defined as

$$A_{ij} := \begin{cases} w(v_i v_j) & \text{if } v_i v_j \in \mathcal{E}, \\ 0 & \text{otherwise}, \end{cases} \tag{1.1}$$

for all $i, j = 1, \ldots, |\mathcal{V}|$. Consider, for instance, the graph shown in Fig. 1.4. Its adjacency matrix is

$$\boldsymbol{A} = \begin{pmatrix} 0 & 4 & 12 \\ 0 & 0 & 6 \\ 0 & 23 & 0 \end{pmatrix}.$$

1.3 County-level Traffic Graph

In this work, we model the county level traffic of the United States on any given day (say January 1st, 2020) using a weighted directed graph where each county is represented as a vertex, and there exists a directed edge ab between two counties a and b provided people traveled from county a to county b on that day. Then, the weight

$$w(ab) = \text{ the traffic count from county } a \text{ to county } b \text{ on that day.}$$

To exemplify this idea, suppose that we only have three counties: Santa Clara County (SC), San Francisco County (SF), and Los Angeles County (LA), and consider the following table showing traffic between these three counties on a particular day. The weighted directed graph associated with Table 1.1 below, is shown in Fig. 1.5.

From county	To county	Traffic count
SC	SF	50
SC	LA	5
SF	SC	0
SF	LA	2
LA	SC	0
LA	SF	60

Table 1.1: Example single day traffic between three counties SC, SF, and LA.

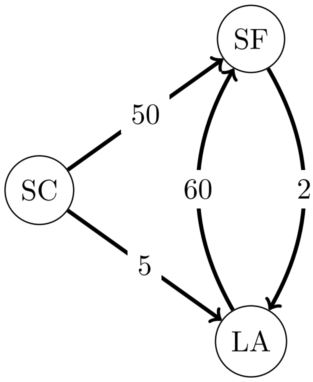


Figure 1.5: County-level traffic graph as per Table 1.1.

The corresponding adjacency matrix is given by

$$A = \begin{pmatrix} 0 & 50 & 5 \\ 0 & 0 & 2 \\ 0 & 60 & 0 \end{pmatrix}.$$

In the 3 county example above, we observe a significant cross-county traffic to SF, and in the context of COVID-19, hinting that implementing non-pharmaceutical interventions (NPIs) such as shelter-in-place orders, social distancing, closure of schools and daycares, should become important in SF. This suggests that the cross-county traffic graph may, in general, be useful to identify vulnerable edges, thus helping us determine exactly which edges should the NPI measures be directed at. This will in turn, help the county-level decision making by determining when, where, and for how long to implement the NPI measures, by looking at the time-varying county-level traffic graphs.

In the ensuing chapters, we will propose an algorithm to compute the distribution of the graph Ricci curvature over the directed edges of the daily county-level traffic graphs, to identify the spatio-temporal variation of the edge vulnerabilities.

Chapter 2

Optimal Transport on Directed Graphs

In this chapter, we introduce the optimal transport between discrete probability measures on a directed graph. To this end, we first recall the notion of directed path, its length, and the hop distance. Then we set up the optimal transport problem, and define the 1-Wasserstein distance. The notion of 1-Wasserstein distance will be needed in the subsequent chapters to compute the graph Ricci curvature.

2.1 Directed Path

Let $[n] := \{1, 2, \ldots, n\}$. Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a *directed path of length* n is a sequence of distinct directed edges $\{e_1, e_2, e_3, ..., e_n\}$ in $\mathcal{G}$, for which there exist a sequence of distinct vertices $\{v_1, v_2, v_3, ..., v_{n+1}\}$ in $\mathcal{G}$ such that

$$e_i = v_i v_{i+1} \quad \text{for all } i \in [n], \quad e_i \in \mathcal{E}, \quad v_i \in \mathcal{V}.$$

Thus, a directed path is a sequence of distinct directed edges joining a sequence of distinct vertices. We denote the *length of a directed path* as $l_{\{e_1,e_2,\ldots,e_n\}}(v_1, v_{n+1}) =$

n.

Notice that for any pair of nonidentical vertices in $\mathcal{G}$, there may exist zero, one or multiple directed paths connecting them. If there exist multiple directed paths between two distinct vertices, it is further possible that some of them may have equal lengths. This is why, in the notation for the length of a directed path, we put the edge sequence as the subscript of l.

2.2 Vertex Reachability

We say that the vertex $v_1 \in \mathcal{V}$ *connects* to $v_2 \in \mathcal{V}$, $v_1 \neq v_2$, if there exists at least one directed path from v_1 to v_2. In other words, v_2 is *reachable* from v_1. Symbolically, we write $v_1 \rightsquigarrow v_2$.

Likewise, we say that the vertex $v_1 \in \mathcal{V}$ *does not connect* to the vertex $v_2 \in \mathcal{V}$, $v_1 \neq v_2$, if there does not exist a directed path from v_1 to v_2, i.e., v_2 is *not reachable* from v_1. Symbolically, we write $v_1 \not\rightsquigarrow v_2$.

We can extend the notion of reachability from vertex-to-vertex level to subset-to-subset level. For instance, given two nonidentical subsets $\mathcal{V}_1, \mathcal{V}_2 \subset \mathcal{V}$, we say $\mathcal{V}_1 \rightsquigarrow \mathcal{V}_2$ if $v_1 \rightsquigarrow v_2$ for all vertex pairs $(v_1, v_2) \in \mathcal{V}_1 \times \mathcal{V}_2$. In words, $\mathcal{V}_1$ *connects to* $\mathcal{V}_2$, or equivalently, the set $\mathcal{V}_2$ is *reachable* from the set $\mathcal{V}_1$. Likewise, $\mathcal{V}_1 \not\rightsquigarrow \mathcal{V}_2$ if there exists a vertex pair $(v_1, v_2) \in \mathcal{V}_1 \times \mathcal{V}_2$ such that $v_1 \not\rightsquigarrow v_2$.

2.3 Hop Distance

Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we define the *hop distance*, also known as the *combinatorial graph distance*, as the mapping $d_{\text{Hop}} : \mathcal{V} \times \mathcal{V} \mapsto \mathbb{R}$ as follows.

$$d_{\text{Hop}}(v_1, v_2) := \begin{cases} \min\limits_{s \in \mathcal{S}} l_s(v_1, v_2) & \text{if } v_1 \rightsquigarrow v_2, \\ \infty & \text{if } v_1 \not\rightsquigarrow v_2, \\ 0 & \text{if } v_1 = v_2, \end{cases}$$

where $\mathcal{S}$ is the set of all directed paths from the vertex v_1 to the vertex v_2. In words, the hop distance is the length of the *shortest directed path* from a vertex to another.

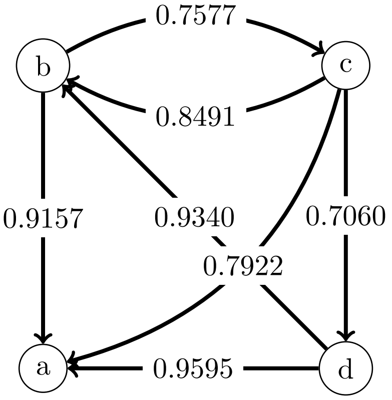


Figure 2.1: An example directed graph to illustrate the hop distance.

To illustrate the ideas above, consider the directed graph shown in Fig. 2.3. Here, $d_{\text{Hop}}(a, c) = 0$ because the vertex a does not connect to the vertex c. On the other hand, $d_{\text{Hop}}(d, c) = 2$ because the shortest directed path from the vertex d to the vertex c is realized by the edge sequence $\{db, bc\}$, and the length of this directed path is $l_{\{db,bc\}}(d, c) = 2$.

2.4 Optimal Transport and the 1-Wasserstein Distance

Given a directed graph $\mathcal{G}\left(\mathcal{V},\mathcal{E}\right)$, consider two subsets $\mathcal{X},\mathcal{Y}\subseteq\mathcal{V}$. Suppose that we are given two probability distributions (i.e., discrete probability measures) $\boldsymbol{\mu},\boldsymbol{\nu}$ over the vertex sets $\mathcal{X},\mathcal{Y}$, respectively. We represent $\boldsymbol{\mu},\boldsymbol{\nu}$ as column probability vectors, i.e., elements of the standard simplices of dimensions[1] $|\mathcal{X}|$ and $|\mathcal{Y}|$, respectively. In other words,

$$\mathbf{1}_{|\mathcal{X}|}^{\top}\boldsymbol{\mu}=\mathbf{1}_{|\mathcal{Y}|}^{\top}\boldsymbol{\nu}=1,\quad \boldsymbol{\mu}\geq\mathbf{0}_{|\mathcal{X}|}\text{ (elementwise)},\quad \boldsymbol{\nu}\geq\mathbf{0}_{|\mathcal{Y}|}\text{ (elementwise)},$$

where the symbols $\mathbf{1}$ and $\mathbf{0}$ denote column vectors with all ones and all zeros, respectively, with their lengths denoted by their subscripts.

The problem of optimal transport [18, 22] concerns with computing the optimal plan to transport the measure $\boldsymbol{\mu}$ on $\mathcal{X}$ to the measure $\boldsymbol{\nu}$ on $\mathcal{Y}$. For this purpose, suppose that the cost of transporting unit amount of probability mass from the vertex $x\in\mathcal{X}$ to the vertex $y\in\mathcal{Y}$ is equal to $d_{\text{Hop}}(x,y)$. Then, the optimal transportation plan $\boldsymbol{\pi}$ solves

$$\underset{\boldsymbol{\pi}}{\arg\min}\quad\sum_{x\in\mathcal{X},y\in\mathcal{Y}}d_{\text{Hop}}(x,y)\boldsymbol{\pi}(x,y) \tag{2.1a}$$

$$\boldsymbol{\pi}(x,y)\geq0\quad\text{(elementwise)}, \tag{2.1b}$$

$$\sum_{y\in\mathcal{Y}}\boldsymbol{\pi}(x,y)=\boldsymbol{\mu}, \tag{2.1c}$$

$$\sum_{x\in\mathcal{X}}\boldsymbol{\pi}(x,y)=\boldsymbol{\nu}. \tag{2.1d}$$

Notice that the objective in (2.1a) is the total transportation cost. Also observe that (2.1c)-(2.1d) together imply $\sum\limits_{x\in\mathcal{X},y\in\mathcal{Y}}\boldsymbol{\pi}(x,y)=1$, which combined with (2.1b) ensure that $\boldsymbol{\pi}$ is a joint probability measure with $\mathcal{X}$ marginal $\boldsymbol{\mu}$, and $\mathcal{Y}$ marginal $\boldsymbol{\nu}$.

[1] Here, $|\cdot|$ denotes the cardinality of the set argument, and the superscript $^{\top}$ denotes the transpose operator.

Since the objective and constraints in (2.1) are all linear in $\boldsymbol{\pi} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{Y}|}$, computing the same amounts to solving a linear program (LP) in the matrix variable $\boldsymbol{\pi}$. To explicitly write this LP, let $\boldsymbol{d}_{\text{Hop}} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{Y}|}$ be the associated Hop distance matrix, and let $\langle \cdot, \cdot \rangle$ denote the Frobenius inner product. Then (2.1) can be rewritten as

$$\underset{\boldsymbol{\pi}}{\arg\min} \quad \langle \boldsymbol{d}_{\text{Hop}}, \boldsymbol{\pi} \rangle \tag{2.2a}$$

$$\boldsymbol{\pi} \geq 0 \quad \text{(elementwise)}, \tag{2.2b}$$

$$\mathbf{1}_{|\mathcal{Y}|}^{\top} \boldsymbol{\pi} = \boldsymbol{\mu}, \tag{2.2c}$$

$$\mathbf{1}_{|\mathcal{X}|}^{\top} \boldsymbol{\pi} = \boldsymbol{\nu}. \tag{2.2d}$$

The minimum value of (2.1), or equivalently of (2.2), is called the *1-Wasserstein distance*, or the *Earth mover's distance*, denoted as $W_1(\boldsymbol{\mu}, \boldsymbol{\nu})$. It can be shown that W_1 is, in fact, a metric on the space of probability measures, i.e., $W_1(\boldsymbol{\mu}, \boldsymbol{\nu}) = 0$ if and only if $\boldsymbol{\mu} = \boldsymbol{\nu}$, $W_1(\boldsymbol{\mu}, \boldsymbol{\nu}) = W_1(\boldsymbol{\nu}, \boldsymbol{\mu})$, and it satisfies the triangle inequality. This quantity W_1 will be an important ingredient in the next Chapter.

We remark here that the Wasserstein metric and optimal transport have garnered significant recent interests in control design [1,11], uncertainty propagation and filtering [2,9], probabilistic validation [6–8] and verification [8,13].

Notice that if the subsets $\mathcal{X}, \mathcal{Y}$ are such that a vertex $x \in \mathcal{X}$ does not connect to a vertex $y \in \mathcal{Y}$, then the corresponding $d_{\text{Hop}}(x, y) = \infty$, and therefore, $W_1 = \infty$. Thus, to have $W_1 < \infty$, each vertex in $\mathcal{X}$ should connect (over possibly multi hop) to every other vertex in $\mathcal{Y}$.

We note that the LP (2.1), or equivalently (2.2), has $|\mathcal{X}| \times |\mathcal{Y}|$ unknowns, and $|\mathcal{X}| \times |\mathcal{Y}| + |\mathcal{X}| + |\mathcal{Y}|$ constraints. The state-of-the-art method to solve this exact LP involves recasting the same as a network flow problem and then to solve it in $\tilde{O}\left(|\mathcal{X}| \times |\mathcal{Y}|\sqrt{|\mathcal{X}| + |\mathcal{Y}|}\right)$ time [14] wherein as usual, $\tilde{O}(\cdot)$ hides polylogarithmic factors in $|\mathcal{X}|, |\mathcal{Y}|$.

Chapter 3

Curvature of Weighted Directed Graphs

As the intuition suggests, the concept of curvature quantifies the degree to which a surface, or smooth Riemannian manifold, differs from being "flat" or Euclidean. Perhaps less intuitive is the idea that the notion of curvature can be extended to nonsmooth discrete spaces such as graphs, and even if such an extension is possible, it is far from obvious what information can be gleaned about a dynamics that is evolving on the underlying network. Progress in this direction came quite recently: a 2009 paper by Ollivier [16] extended the idea of Ricci curvature to Markov chains on general metric spaces, and a 2011 follow-up by Lin, Lu and Yau [15] extended the same to the graphs. In the following, we review these ideas and then introduce the Ricci curvature for weighted directed graphs.

3.1 Ricci Curvature on a Riemannian Manifold

In Riemannian geometry, the notion of curvature measures the deviation of the manifold from being locally Euclidean. Ricci curvature quantifes that deviation for tangent directions. It controls the average dispersion of geodesics around that direction. It also controls the growth of the volume of distance balls and spheres.

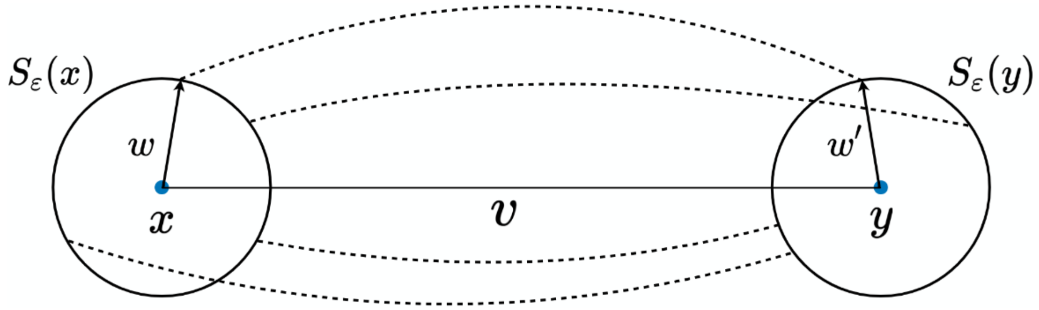


Figure 3.1: Two spheres $S_\varepsilon(x)$ and $S_\varepsilon(y)$ in a Riemannian manifold $\mathcal{M}$. The tangent vector at $x \in \mathcal{M}$ is the vector v with endpoint $y \in \mathcal{M}$. The dashed lines show the "inter-sphere" geodesics.

Let $\varepsilon, \delta > 0$, and as shown in Fig. 3.1, consider two points x, y in a smooth Riemannian manifold $\mathcal{M}$ of dimension n. Suppose that the points x, y are at a geodesic distance $d(x, y) = \delta$ apart. Let v be a tangent vector at x with endpoint y. Suppose the sphere $S_\varepsilon(x)$ with radius ε centered at x get parallel transported to the sphere $S_\varepsilon(y)$ with radius ε centered at y. The points on these two spheres are, on average, at a distance $\delta(1 - \frac{\varepsilon^2}{2n}\mathbf{Ric}(v))$, where $\mathbf{Ric}(v)$ denotes the Ricci curvature in the tangent direction v; see e.g., [17, Corollary 10], [19].

From above, it is clear that in the context of Riemannian geometry, a positive Ricci curvature implies that small spheres are closer (in transport distance) than their centers. Specifically, if the Ricci curvature is negative, small spheres are further than their centers. Replacing the *spheres* by *balls* replaces the factor $\frac{\varepsilon^2}{2n}$

by $\frac{\varepsilon^2}{2(n+2)}$; see [17, Corollary 10]. In other words, in a Riemannian manifold $\mathcal{M}$ of dimension n, two balls of radius ε centered at $x, y \in \mathcal{M}$ respectively, are located at an average distance

$$\delta \left(1 - \frac{\varepsilon^2}{2(n+2)} \mathbf{Ric}(v) + O\left(\varepsilon^3 + \varepsilon^2 \delta\right) \right).$$

Here, the average is with respect to the uniform Riemannian volume measure on the respective balls. Neglecting the higher order (i.e., $O(\cdot)$) term, gives us the stated relation among the average distance between the two balls, the distance between the centers of the two balls, and the Ricci curvature.

3.2 Ricci Curvature on a Metric Space

The 2009 paper by Ollivier [16] extended the notion of Ricci curvature to general metric spaces. The key idea was to equip the metric space with a suitable (possibly discrete) measure, and to extend the geometric intuition that in a positively (resp. negatively) curved space, balls are closer (farther) than their centers. Balls are defined by their volume measures. So to define Ricci curvature in a metric space, what is needed is a way to compare the distance between suitably defined "ball measures" with the distance between those balls' centers (points in that metric space). In case of a smooth Riemannian manifold, it is natural to take arbitrarily small balls (hence the small radius $\varepsilon > 0$ in the previous subsection). In case of a graph, it is natural to take unit balls.

Thus, characterizing the Ricci curvature in a metric space requires two ingredients:

- a way to compute distance between the volume measure of the balls.

- a way to compute distance between the "points".

When the metric space is a graph, the "points" are the nodes or the vertices of the graph, and a natural notion of distance between the nodes is the "hop distance", i.e., the minimum number of edges (geodesic length) from one node to the other.

To compute distance between the balls, one needs a notion of distance between the (normalized) measures, and in case of a graph, this is naturally given by the 1-Wasserstein distance W_1 introduced in Chapter 2. Thus, computing the distance between *unit balls* centered on two different nodes of a graph reduces to the following: consider the pair of normalized discrete measures in the respective (one hop) neighborhood of the two nodes under consideration, and then compute the 1-Wasserstein distance between these two discrete measures. In the graph context, unit ball $\equiv$ "one hop neighborhood". To compute the Ricci curvature in a graph, we are thus led to comparing the transportation distance between the discrete measures in the one hop neighborhood of two given points to the distance between those points.

To formalize the ideas above, let $(\mathcal{X}, d, m)$ be a polish (i.e., complete separable) metric space equipped with measure m, and distance metric d. Let $x,\, y \in \mathcal{X}$ be two distinct points. The Ricci curvature $\kappa(x, y)$ of $(\mathcal{X}, d, m)$ along the path xy is

$$\kappa(x, y) = 1 - \frac{W_1(m_x, m_y)}{d(x, y)}, \tag{3.1}$$

where $W_1(m_x, m_y)$ is the 1-Wasserstein distance between two measures m_x and m_y over the unit balls centered around the points x, y, respectively. In (3.1), the $d(x, y)$ denotes the length of the geodesic distance between $x,\, y \in \mathcal{X}$. Often (3.1) is referred to as the "coarse Ricci curvature" [17, Def. 18]. We point out that (3.1) tacitly assumes that both the measures m_x, m_y have finite first moments.

3.3 Unit Ball in A Weighted Directed Graph

Consider a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$. For any $x \in \mathcal{V}$, let $\mathcal{N}(x) := \{x_1, x_2, \ldots, x_k\}$ be the one hop out-neighborhood of x, meaning that $d_{\text{Hop}}(x, x_i) = 1$ for $i = 1, 2, ..., k$. Clearly, $\mathcal{N}(x) \subset \mathcal{V}$ for all $x \in \mathcal{V}$. We define the unit ball of any node $x \in \mathcal{V}$ as a discrete probability measure

$$m_x = \{\mu_x(x_1), \mu_x(x_2), \ldots, \mu_x(x_k)\}$$

supported on $\{x_1, x_2, \ldots, x_k\}$, where

$$\mu_x(x_i) := \begin{cases} \frac{w(xx_i)}{\sum_{j=1}^{k} w(xx_j)} & \text{if } x_i \in \mathcal{N}(x), \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

As an example, consider the node b in Fig. 2.1. Its out-neighborhood $\mathcal{N}(b) = \{a, c\}$. The unit ball centered at the node b is given by the (normalized) discrete probability measure

$$m_b = \left\{ \frac{0.9157}{0.7577 + 0.9157}, \frac{0.7577}{0.7577 + 0.9157} \right\} = \{0.5472, 0.4528\}$$

supported on $\mathcal{N}(b) = \{a, c\}$.

3.4 W_1 Distance between Unit Balls in A Weighted Directed Graph

Recall from Chapter 2 that the 1-Wasserstein distance W_1 is defined between two normalized measures. Following Sec. 3.2, the 1-Wasserstein distance between two unit balls centered at $x, y \in \mathcal{V}$ in a weighted directed graph $\mathcal{G}\,(\mathcal{V}, \mathcal{E}, w)$, is then $W_1(m_x, m_y)$ where the normalized measures m_x, m_y are defined as in (3.2), and are supported on $\mathcal{N}(x)$ and $\mathcal{N}(y)$, respectively. Thus, each choice of distinct nodes $x, y \in \mathcal{V}$ such that x connects to y (i.e., node y is reachable from node x via some directed path) yields a value of W_1 associated with that pair of nodes. It is obvious that the W_1 distance between a ball to itself must be zero. Hence, we can element-wise define a 1-Wasserstein matrix $\boldsymbol{W} = [W_{ij}] \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ where $|\mathcal{V}|$ denotes the cardinality of the vertex set $\mathcal{V} = \{v_1, v_2, v_3, ..., v_{|\mathcal{V}|}\}$, as

$$W_{ij} := \begin{cases} 0 & \text{if } i = j, \\ \text{undefined} & \text{if } (i \neq j) \bigwedge ((v_i \not\rightsquigarrow v_j) \bigvee (\mathcal{N}(v_i) = \varnothing) \bigvee (\mathcal{N}(v_j) = \varnothing)), \\ W_1\left(m_{v_i}, m_{v_j}\right) & \text{otherwise,} \end{cases} \tag{3.3}$$

where $i, j = 1, \ldots, |\mathcal{V}|$, and following Ch. 2.2, the notation "$v_i \not\rightsquigarrow v_j$" stands for "$v_i$ does not connect to v_j". In (3.3), the symbol $\bigwedge$ denotes the logical AND. The symbol $\bigvee$ denotes the logical OR. We point out that the condition given the third case in (3.3), includes the possibility that the W_1 value equals ∞.

To exemplify (3.3), consider again the weighted directed graph shown in Fig. 2.1. Its 1-Wasserstein matrix $\boldsymbol{W} \in \mathbb{R}^{4\times4}$ is

$$\boldsymbol{W} = \begin{pmatrix} 0 & \text{undefined} & \text{undefined} & \text{undefined} \\ \text{undefined} & 0 & \infty & \infty \\ \text{undefined} & 0.6293 & 0 & 0.2852 \\ \text{undefined} & 0.3801 & \infty & 0 \end{pmatrix}$$

where the row and column indices follow the sequence $\{a, b, c, d\}$. The off-diagonal finite values in the above matrix were computed by solving the LP (2.1), or equivalently (2.2) from Chapter 2.3, using the `cvx` package [3] in MATLAB. The procedure for the W_1 computation using `cvx` is outlined in Algorithm 1.

Algorithm 1 Compute the Earth mover's distance (EMD) W_1

```
procedure EMD($\boldsymbol{C}$, $\boldsymbol{\mu}$, $\boldsymbol{\nu}$)    ▷ $\boldsymbol{C}$ is the ground cost matrix for optimal transport between the supports of the discrete probability mass function vectors $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$
    $m = \text{count}(\boldsymbol{\mu})$
    $n = \text{count}(\boldsymbol{\nu})$
    $\boldsymbol{A} = \begin{pmatrix} \mathbf{1}_n^\top \otimes \boldsymbol{I}_m \\ \boldsymbol{I}_n \otimes \mathbf{1}_m^\top \end{pmatrix}$    ▷ $\mathbf{1}_m$ denotes the $m \times 1$ column vector of ones, and $\boldsymbol{I}_m$ denotes the identity matrix of size $m \times m$, the symbol $\otimes$ denotes the Kronecker product
    $\boldsymbol{b} = \begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{\nu} \end{pmatrix} \in \mathbb{R}_{\geq 0}^{m+n}$
    $\boldsymbol{c} = \text{vec}(\boldsymbol{C})$    ▷ vectorize the cost matrix $\boldsymbol{C} \in \mathbb{R}_{\geq 0}^{m \times n}$ to a column vector $\boldsymbol{c} \in \mathbb{R}_{\geq 0}^{mn}$
    cvx_begin
        variable $\boldsymbol{x} \in \mathbb{R}^{mn}$
        minimize $\langle \boldsymbol{c}, \boldsymbol{x} \rangle$
        subject to
            $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$    ▷ Equality constraints for the LP
            $\boldsymbol{x} \geq \mathbf{0}$    ▷ Elementwise vector inequality constraint
    cvx_end
end procedure
```

3.5 Ricci Curvature for a Weighted Directed Graph

Given a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$, to define its Ricci curvature, we generalize (3.1) as follows. For $v_i, v_j \in \mathcal{V}$, we define the Ricci curvature

$\kappa(v_i, v_j)$ along a weighted directed path $v_i v_j$ as

$$\kappa\left(v_i, v_j\right) := 1 - \frac{W_{ij}}{d_{\text{Hop}}\left(v_i, v_j\right)}. \tag{3.4}$$

where W_{ij} is defined as in (3.3). Then, using (3.4), we can define a *Ricci curvature matrix* $\boldsymbol{K} = [K_{ij}] \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ having elements

$$K_{ij} := \kappa\left(v_i, v_j\right), \quad \text{where } i, j = 1, \ldots, |\mathcal{V}|, \tag{3.5}$$

associated with the given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$.

Notice that (3.3) permits the possibilities that W_{ij} may be undefined or equals to 0, or equals to ∞. Since $W_{ij} = 0$ for $i = j$, and in that case, $d_{\text{Hop}}\left(v_i, v_j\right) = 0$, so (3.4) evaluates to $1 - \frac{0}{0}$, and therefore, K_{ij} is undefined. If W_{ij} itself is undefined, then so is the corresponding K_{ij}. If W_{ij} equals ∞ but $d_{\text{Hop}}\left(v_i, v_j\right) < \infty$, then by (3.4), we get $K_{ij} = -\infty$.

Algorithm 2 Compute the Ricci curvature matrix $\boldsymbol{K}$

```
procedure RicciOllivier_Curvature($\mathcal{G}$)
    for $i = 1$ to $|\mathcal{V}|$ do
        $\boldsymbol{\mu}\{i\} = \frac{\boldsymbol{e}\{i\}}{\text{sum}(\boldsymbol{e}\{i\})}$    ▷ $\boldsymbol{e}\{i\}$ denotes the vector of edge weights going out from $v_i$
    end for
    for $i = 1$ to $|\mathcal{V}|$ do
        for $j = 1$ to $|\mathcal{V}|$ do
            if $(v_i \rightsquigarrow v_j) \bigwedge (\mathcal{N}(v_j) = \varnothing)$ then
                $W_{ij} = \text{EMD}(\boldsymbol{C}, \boldsymbol{\mu}\{i\}, \boldsymbol{\mu}\{j\})$    ▷ $\boldsymbol{C}$ is the ground cost matrix for optimal transport between $\boldsymbol{\mu}\{i\}$ and $\boldsymbol{\mu}\{j\}$. The procedure EMD follows Algorithm 1.
                $K_{ij} = 1 - \frac{W_{ij}}{d_{\text{Hop}}(v_i,v_j)}$
            else
                $K_{ij}$ is undefined
            end if
        end for
    end for
return $\boldsymbol{K}$                    ▷ return the Ricci curvature matrix
end procedure
```

To give an example for the computation of $\boldsymbol{K}$, consider again the weighted directed graph $\mathcal{G}$ shown in Fig. 2.1. Its Ricci curvature matrix $\boldsymbol{K} \in \mathbb{R}^{4\times 4}$ is

$$\boldsymbol{K} = \begin{pmatrix} \text{undefined} & \text{undefined} & \text{undefined} & \text{undefined} \\ \text{undefined} & \text{undefined} & -\infty & -\infty \\ \text{undefined} & 0.2588 & \text{undefined} & 0.5960 \\ \text{undefined} & 0.5930 & -\infty & \text{undefined} \end{pmatrix},$$

where the row and column indices follow the vertex sequence $\{a, b, c, d\}$.

Some properties of the Ricci curvature matrix $\boldsymbol{K}$ are immediate from (3.5). For instance, $K_{ij} \geq 0$ if and only if $W_{ij} \leq d_{\text{Hop}}(v_i, v_j)$. This matches with the geometric intuition that a positive curvature implies that unit balls (here, discrete measures supported on one hop out-neighbors) are closer than their centers (here, graph vertices), as we explained in Ch. 3.1 and 3.2. Likewise, $K_{ij} < 0$ if and only if $W_{ij} > d_{\text{Hop}}(v_i, v_j)$. This again confirms the geometric insight that a negative curvature implies that unit balls are farther than their centers.

The above observation can phrased equivalently as follows. A positive (resp. negative) Ricci curvature between two vertices implies that their one hop out-neighborhoods are on average “well connected” (resp. “weakly connected”) than those vertices. In our COVID-19 spread context, the vertices stand for the counties, and the directed edges give the inter-county daily traffic flow directions with edge-weights being indicative of the corresponding traffic volume. Thus, if the outgoing edges from a county has mostly large positive curvatures, then we can expect that county to be a "super-spreader" of the virus.

3.6 Scalar Curvature for a Weighted Directed Graph

Recall that the Ricci curvature is defined over paths. However, we sometimes want to focus on the vertices of a graph. For this purpose, we next define the *scalar curvature* as the weighted average of the Ricci curvature. This is inspired by the analogous definition in the Riemannian geometry context. Specifically, in a Riemannian manifold $\mathcal{M}$, the scalar curvature at a point $x \in \mathcal{M}$ is the average of the Ricci curvature $\mathbf{Ric}(v)$ over all unit vectors v around x. Then, it is natural to generalize the scalar curvature for a metric space $(\mathcal{X}, d, m)$ as follows. Suppose that $\kappa(x, y)$ is the Ricci curvature of $(\mathcal{X}, d, m)$ along the path xy, as defined in (3.1). Then the scalar curvature at $x \in \mathcal{X}$ is $\int \kappa(x, y) \mathrm{d}m(y)$, i.e., the average of the Ricci curvature $\kappa(x, y)$ w.r.t. the volume measure m.

Motivated by the above, we are now ready to define the scalar curvature in a weighted directed graph. Given a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$ and its Ricci curvature matrix $\boldsymbol{K}$, we define the *scalar curvature* $s\left(v_i\right)$ of a vertex v_i as

$$s(v_i) := \sum_{j \in \mathcal{N}_{v_i}} K_{ij} \mu_{v_i}\left(v_j\right), \quad i = 1, \ldots, |\mathcal{V}|, \tag{3.6}$$

where $\mu_{v_i}\left(v_j\right)$ was introduced in Ch. 3.3. In (3.6), we tacitly assume that the summation indices $j \in \mathcal{N}_{v_i}$ are such that K_{ij} is finite and well-defined. Using (3.6), we then define a *scalar curvature vector* $\boldsymbol{s} \in \mathbb{R}^{|\mathcal{V}|}$ with entries $s_i := s(v_i)$, $i = 1, \ldots, |\mathcal{V}|$. For completeness, we outline the computation of the vector $\boldsymbol{s}$ in

Algorithm 3.

Algorithm 3 Compute the scalar curvature vector $\boldsymbol{s}$

```
procedure ScalarCurvature($\mathcal{G}$, $\boldsymbol{K}$, $\boldsymbol{\mu}$)
    for $i = 1$ to $|\mathcal{V}|$ do
        $s_i = \sum_{j \in \mathcal{N}_{v_i}} K_{ij} \mu_{v_i}(v_j)$
    end for
return $\boldsymbol{s} \in \mathbb{R}^{|\mathcal{V}|}$                ▷ return the scalar curvature vector $\boldsymbol{s}$
end procedure
```

Chapter 4

Numerical Simulations

In this Chapter, we apply the graph curvature related ideas and algorithms introduced in the preceding chapters to analyze the spread of COVID-19 based on the inter-county traffic data for the state of California. Based on the computation performed on real data, our intent is to demonstrate that the graph Ricci curvature and scalar curvature are indeed suited for network-level risk analytics.

4.1 Simulation Set Up

We obtained the daily inter-county traffic count data for the state of California from March 1, 2020 to March 31, 2021. These data were obtained from SafeGraph [12] dataset: "Social Distancing Metrics". The obtained anonymous daily commute data were differentially private [4, 5] and were collected based on cellular pings and social network usage. Using this dataset, we computed the county level traffic graphs $\mathcal{G}\left(\mathcal{V},\mathcal{E},w\right)$ and the associated adjacency matrices, as detailed in Ch. 1.3. The vertex set $\mathcal{V}$ in our context denotes the collection of counties in the state of California with $|\mathcal{V}| = 58$.

We emphasize here that since the inter-county daily traffic in any given day is different from the other, hence the graphs and adjacency matrices are time-varying. In other words, we have a different county level traffic graph and thus different adjacency matrix for each day because the edge weights for the same directed edge for different days are different.

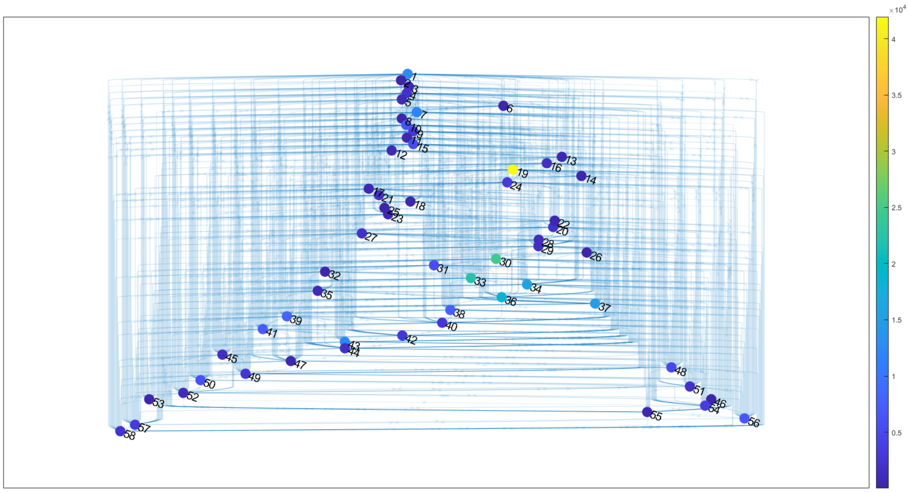


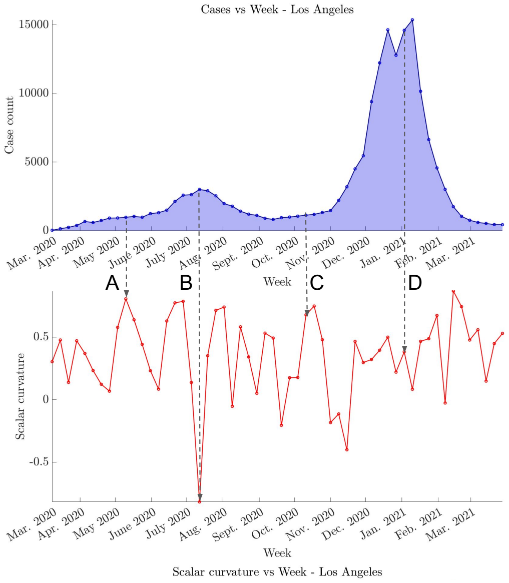
Figure 4.1: County level traffic graph for all the 58 counties of the state of California corresponding to January 1st, 2021. The above does not show the edge weights but instead shows the weighted out-degrees for all the vertices (i.e., counties) as the colormap. This helps depict which counties have more outgoing traffic than others for that particular day.

Fig. 4.1 shows the county-level traffic graph for the state of California corresponding to January 1, 2021, wherein the Colors of the vertices (see colormap) denote the associated weighted out-degrees, i.e., the sum of edge weights going out from that vertex for that day. After computing all the county-level traffic graphs and the corresponding adjacency matrices, we applied the Algorithm 2 to compute the Ricci curvature matrices $\boldsymbol{K}$ for each week. Recall from Ch. 3 that Algorithm 2 in turn uses Algorithm 1. Having obtained the Ricci curvature matrices at the weekly time scale, we finally computed the scalar curvature vectors $\boldsymbol{s}$ using Algorithm 3, again at the weekly timescale.

4.2 California COVID-19 Cases and the Scalar Curvature

Recall from Ch. 3.6 that the scalar curvature of a vertex v is the weighted average of Ricci curvatures going out from v. Recall also that in our county-level traffic graphs, the vertices represent the counties. Therefore, a high scalar curvature for a county indicates that the average outgoing Ricci curvatures from that county is high. If we postulate that inter-county traffic may contribute to the spread of COVID-19, then it follows that a high scalar curvature can be indicative of an acceleration of COVID-19 case counts. Likewise, a small scalar curvature for a particular county implies that the average outgoing Ricci curvatures from that county remains small, i.e., a small scalar curvature may predict a deceleration of COVID-19 cases.

In Figs. 4.2–4.5, we plot how the COVID-19 case counts and the scalar curvatures varied over the weeks for 4 counties: Los Angeles, Sacramento, San Diego, and Santa Clara.



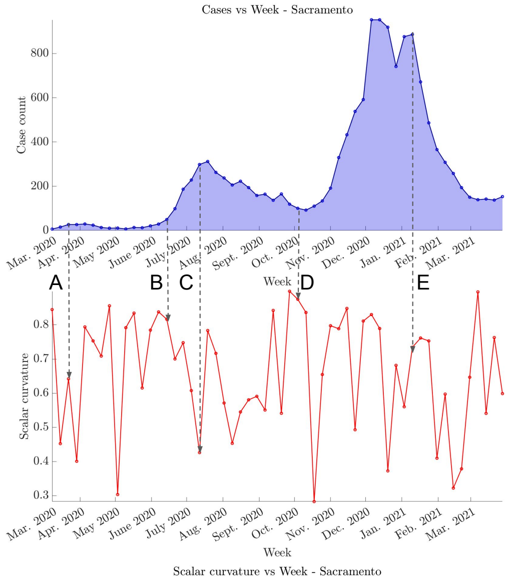
A. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up.

B. Low Scalar Curvature, harder for COVID-19 to spread, and the number of new cases has a negative acceleration. Cases vs Week graph concave down.

C. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up again.

D. COVID-19 vaccine comes out, number of new cases drops significantly as more people were vaccinated.

Figure 4.2: Scalar curvature and cases vs week graph - Los Angeles county.



A. Low Scalar Curvature, harder for COVID-19 to spread, and the number of new cases has a negative acceleration. Cases vs Week graph concave down.

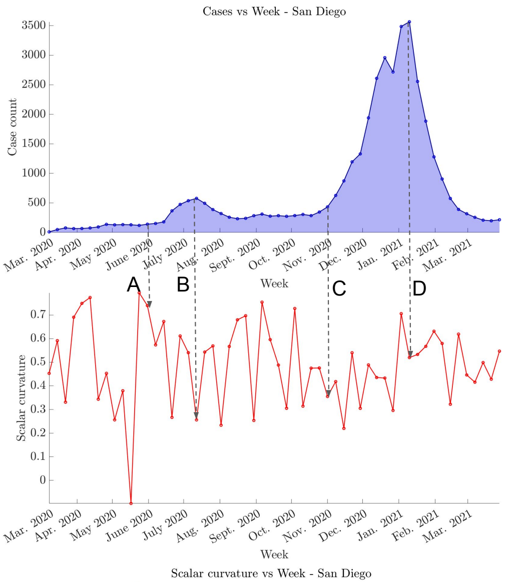
B. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up again.

C. Low Scalar Curvature, harder for COVID-19 to spread, and the number of new cases has a negative acceleration. Cases vs Week graph concave down.

D. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up again.

E. COVID-19 vaccine comes out, number of new cases drops significantly as more people were vaccinated.

Figure 4.3: Scalar curvature and cases vs week graph - Sacramento county.



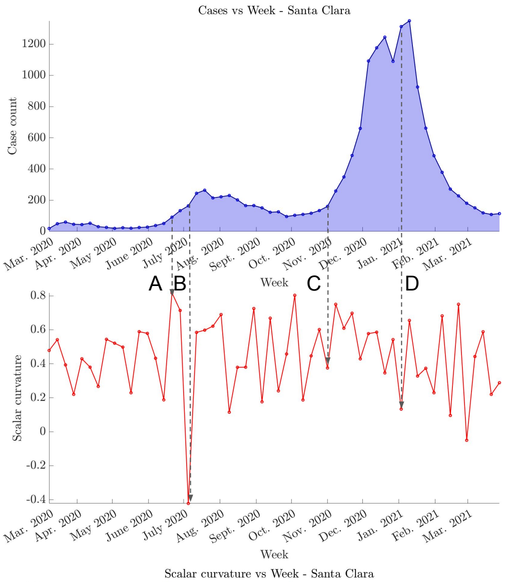
A. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up.

B. Low Scalar Curvature, harder for COVID-19 to spread, and the number of new cases has a negative acceleration. Cases vs Week graph concave down.

C. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up again.

D. COVID-19 vaccine comes out, number of new cases drops significantly as more people were vaccinated.

Figure 4.4: Scalar curvature and cases vs week graph - San Diego county.



A. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up.

B. Low Scalar Curvature, harder for COVID-19 to spread, and the number of new cases has a negative acceleration. Cases vs Week graph concave down.

C. High Scalar Curvature, easier for COVID-19 to spread, and the number of new cases has a positive acceleration. Cases vs Week graph concave up again.

D. COVID-19 vaccine comes out, number of new cases drops significantly as more people were vaccinated.

Figure 4.5: Scalar curvature and cases vs week graph - Santa Clara county.

From Fig. 4.2–4.5, comparing the time histories of the COVID-19 case counts and scalar curvature, we observe that the scalar curvature appears to be a predictor of new cases. When scalar curvature drops, case count is about to go down, and when scalar curvature rises, case count is about to go up.

We have observed the trends similar to the Figs. 4.2–4.5 for all the counties. This suggests that future trend of case counts can be reliably based on scalar curvature, and the NPIs may be implemented in an data-driven adaptive manner, for instance, by issuing more regulations to counties with growing scalar curvature.

4.3 California COVID-19 Cases and the Ricci Curvature

The graph Ricci curvature contains more information than the graph scalar curvature. This is because the graph Ricci curvature not only considers how well a particular vertex is connected with other vertices, but also considers which vertices are best connected to or path connected by this particular vertex.

Ricci curvature is directed. For a county-level traffic graph of California, outward Ricci curvature of a county measures the wellness of connection from that county to all its reachable counties. Similarly, the inward Ricci curvature of a county measures the wellness of connection from its reachable counties to itself. In summary, the Ricci curvatures help answer more fine-grained questions such as from which external counties is the virus spreading from, and to which counties is the virus spreading to.

4.3.1 Outward Ricci Curvature

We computed the outward Ricci curvatures of two populated counties with a high number of COVID-19 cases: Los Angeles(LA) and Santa Clara(SC), to examine which counties were largely affected by LA and SC. Figures 4.6 and 4.7 show the heatmaps of the outward Ricci curvatures from LA and SC, respectively.

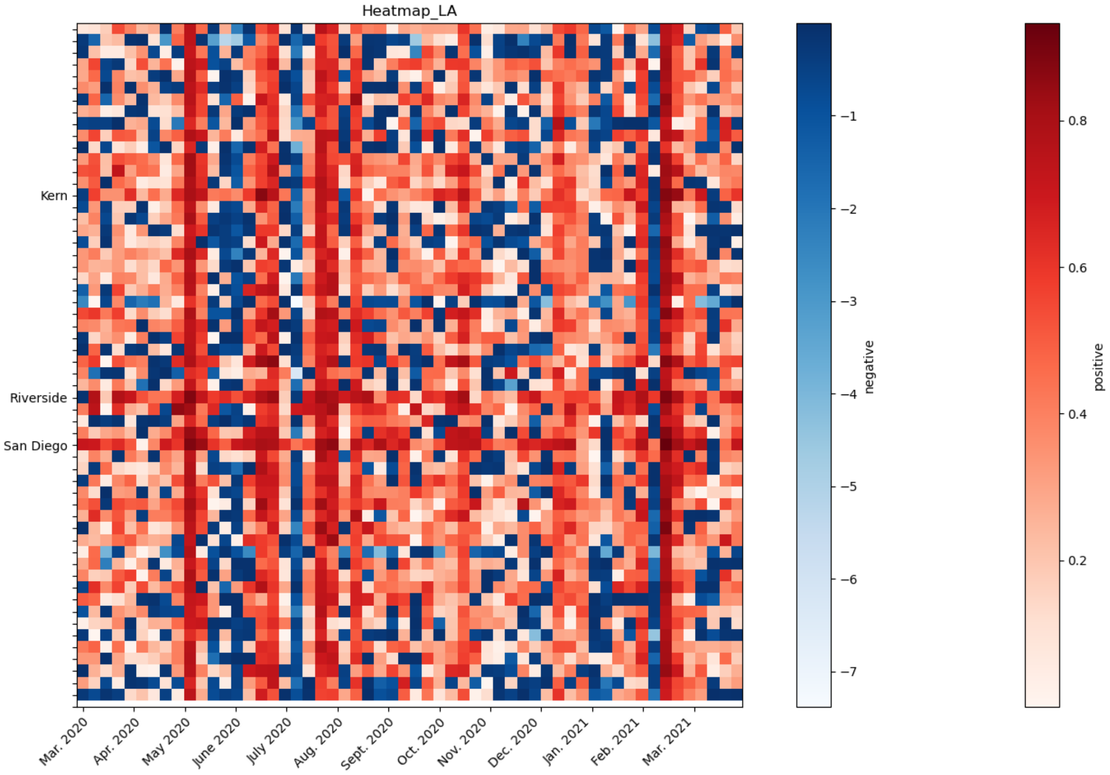


Figure 4.6: Heatmap of the outward Ricci curvature from the Los Angeles county.

Focusing on columns of figure 4.6, the heatmap is mostly red(positive) from March 2020 to August 2020, which matches with our scalar curvature graph. Perhaps, the more interesting to examine is a horizontal direction in the heatmap. Each row of the heatmap shown in Fig. 4.6 represents Ricci curvature from LA to a particular county. If a row is particularly red, it means that LA might be a important source of the virus, i.e., “spreader” for that county. For instance, Fig. 4.6 shows that LA is well connected with Riverside, Kern, and San Diego countioes, and implies that LA could be the main source of the virus spread in these three counties.

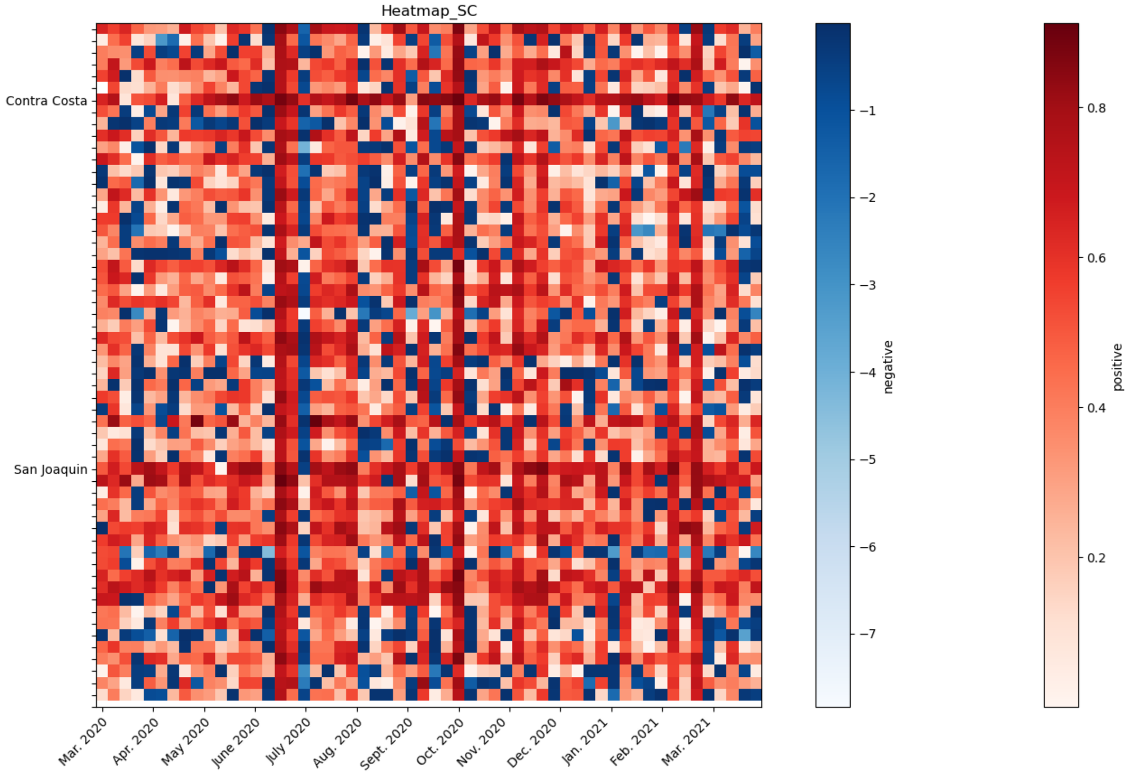


Figure 4.7: Heatmap of the outward Ricci curvature from the Santa Clara county.

Similarly, for Santa Clara, the rows for Contra Costa and San Joaquin counties shown in Fig. 4.7, are particularly red, and this means Santa Clara could county be a major source of virus spread for them.

4.3.2 Inward Ricci Curvature

We computed the inward Ricci curvatures of two counties with relatively low number of COVID-19 cases: Butte and Ventura, to examine where did the virus came from in those two counties..

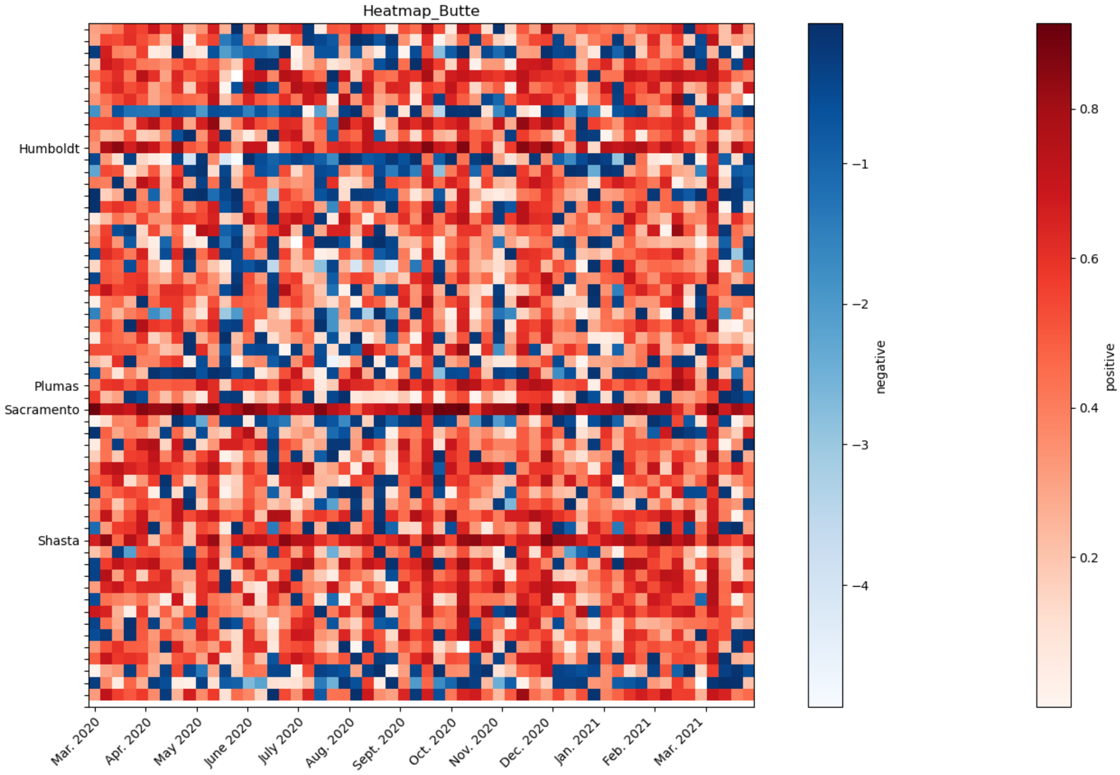


Figure 4.8: Heatmap of the inward Ricci curvature to the Butte county.

In Fig. 4.8, each row of the heatmap represents Ricci curvature from a particular county to Butte. If a row is particularly red, it means that county connects to Butte well, and might be responsible for the virus in Butte. As shown in Fig. 4.8, our analysis finds that the counties Humboldt, Plumas, Sacramento, and Shasta connect to Butte well, and out of these four counties, only Sacramento has a lot of COVID-19 cases. So we can conclude that cases in Butte will be more likely to come from Sacramento than from Los Angeles. This conclusion also makes sense geographically because Butte is located only 80 miles away from Sacramento but about 500 miles away from Los Angeles.

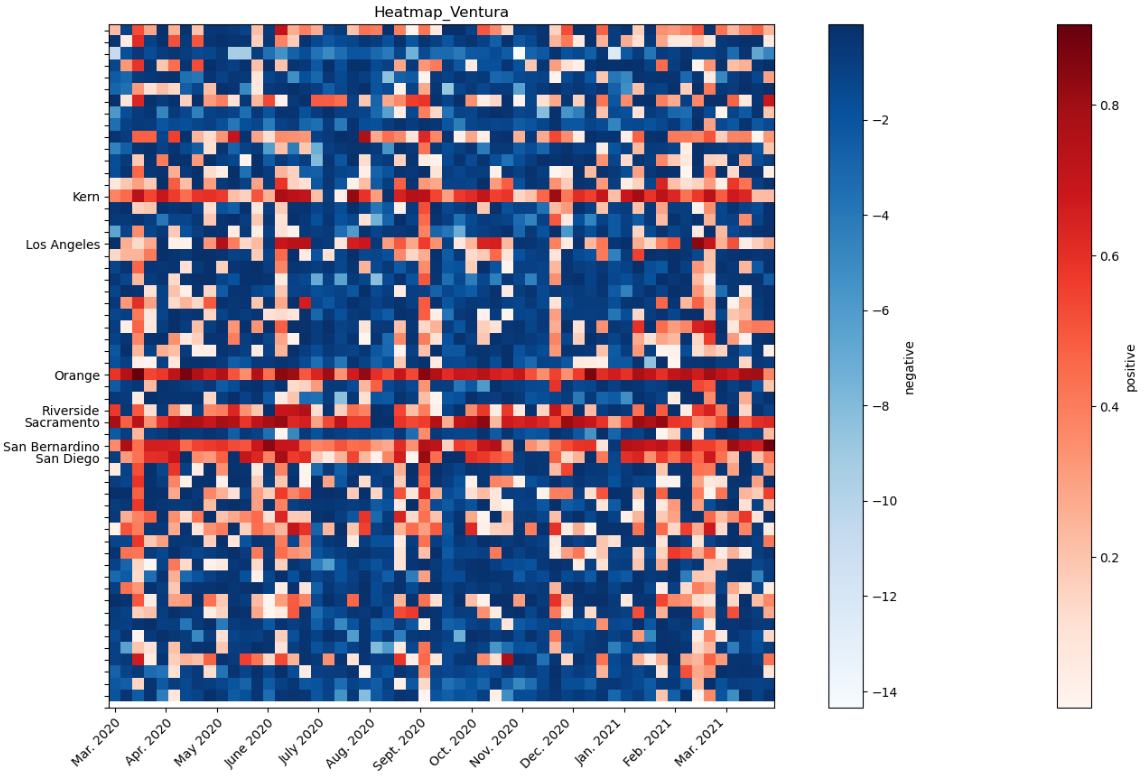


Figure 4.9: Heatmap of the inward Ricci curvature to the Ventura county.

In Fig. 4.9, we observe that the heatmap for Ventura is dominantly blue, meaning that Ventura is a comparatively isolated county. But it still has great connection with counties such as Los Angeles and San Bernardino. The heatmap shown in Fig. 4.9 implies that LA is more likely to be the main virus source for Ventura county.

Chapter 5

Conclusions

In this thesis, we posit that graph curvatures are well-suited for analyzing the spatio-temporal dynamics, such as the spread of the COVID-19 virus, evolving on graphs. In particular, we defined the Ricci curvature and the scalar curvature for weighted directed graphs. We used the inter-county daily traffic data during March 1, 2020–March 31, 2021, for the state of California to highlight that the Ricci curvature and the scalar curvature can serve as a predictor of the surge and reduction of the case-counts. Specifically, the graph Ricci curvature is shown to encode fine-grained interaction information helping to locate the origin and destination of virus spread.

We emphasize that unlike the notions such as network centrality, the Ricci curvature provides pairwise information over all possible pathways in the network, i.e., returns a distribution of the curvature over the edges at any given time. In our application, this is particularly significant to capture the county-to-county interactions contributing to the resilience of the COVID-19 spread. In other words, unlike the nodal measures, the notion of network Ricci curvature does not "average out" the pairwise interaction information. This particular realization has prompted researchers to investigate the implication of curvature in different applications: robustness of cancer genomic networks [20], risk in financial networks [21], and congestion management in wireless networks [23] have all appeared in the last few years. In the proposed research, the Ricci curvature computation will reveal which edges on the network, at any given time, are most robust (having large positive Ricci curvature), i.e., most resistant to NPI measures, and which may be more fragile (having large negative Ricci curvature), hence more responsive to immediate NPI measures but less effective in the long network range (i.e., for large hop distances).

We remark that the numerical simulations reported in this thesis directly generalizes to larger spatial (e.g., entire country) and temporal (e.g., longer duration, daily level analysis etc.) scales. We decided to restrict our analysis to the state of California only for the illustration purpose.

One possible direction of future research is to design control mechanisms to optimally (e.g., minimum effort) steer the spatial distribution of the graph Ricci curvatures over time. This can be of interest to design optimal time-varying mitigation strategies for abating the virus spread. Another potential direction is to generalize the notions of Ricci and scalar curvatures from graphs to higher dimensional simplical complexes, to capture more than two-site interactions.

Bibliography

[1] Kenneth Caluya and Abhishek Halder. Wasserstein proximal algorithms for the Schrödinger bridge problem: Density control with nonlinear drift. *IEEE Transactions on Automatic Control*, 2021.

[2] Kenneth F Caluya and Abhishek Halder. Gradient flow algorithms for density propagation in stochastic systems. *IEEE Transactions on Automatic Control*, 65(10):3991–4004, 2019.

[3] Inc. CVX Research. CVX: Matlab software for disciplined convex programming, version 2.0. `http://cvxr.com/cvx`, August 2012.

[4] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer, 2006.

[5] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[6] Abhishek Halder and Raktim Bhattacharya. Model validation: A probabilistic formulation. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pages 1692–1697. IEEE, 2011.

[7] Abhishek Halder and Raktim Bhattacharya. Further results on probabilistic model validation in Wasserstein metric. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5542–5547. IEEE, 2012.

[8] Abhishek Halder and Raktim Bhattacharya. Probabilistic model validation for uncertain nonlinear systems. *Automatica*, 50(8):2038–2050, 2014.

[9] Abhishek Halder and Tryphon T Georgiou. Gradient flows in uncertainty propagation and filtering of linear Gaussian systems. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 3081–3088. IEEE, 2017.

[10] Abhishek Halder, Kooktae Lee, and Raktim Bhattacharya. Optimal transport approach for probabilistic robustness analysis of F-16 controllers. *Journal of Guidance, Control, and Dynamics*, 38(10):1935–1946, 2015.

[11] Abhishek Halder and Eric DB Wendel. Finite horizon linear quadratic gaussian density regulator with Wasserstein terminal cost. In *2016 American Control Conference (ACC)*, pages 7249–7254. IEEE, 2016.

[12] SafeGraph Inc. `https://www.safegraph.com/`.

[13] Kooktae Lee, Abhishek Halder, and Raktim Bhattacharya. Performance and robustness analysis of stochastic jump linear systems using Wasserstein metric. *Automatica*, 51:341–347, 2015.

[14] Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: Solving linear programs in $\tilde{O}(\sqrt{\text{rank}})$ iterations and faster algorithms for maximum flow. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 424–433. IEEE, 2014.

[15] Yong Lin, Linyuan Lu, and Shing-Tung Yau. Ricci curvature of graphs. *Tohoku Mathematical Journal, Second Series*, 63(4):605–627, 2011.

[16] Yann Ollivier. Ricci curvature of Markov chains on metric spaces. *Journal of Functional Analysis*, 256(3):810–864, 2009.

[17] Yann Ollivier. A survey of ricci curvature for metric spaces and markov chains. In *Probabilistic approach to geometry*, pages 343–381. Mathematical Society of Japan, 2010.

[18] Gabriel Peyré and Marco Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[19] Areejit Samal, R.P. Sreejith, Jiao Gu, Shiping Liu, Emi Saucan, and Jürgen Jost. Comparative analysis of two discretizations of Ricci curvature for complex networks. *Scientific Reports*, 8(8560):2–3, 2018.

[20] Romeil Sandhu, Tryphon Georgiou, Ed Reznik, Liangjia Zhu, Ivan Kolesov, Yasin Senbabaoglu, and Allen Tannenbaum. Graph curvature for differentiating cancer networks. *Scientific reports*, 5(1):1–13, 2015.

[21] Romeil S Sandhu, Tryphon T Georgiou, and Allen R Tannenbaum. Ricci curvature: An economic indicator for market fragility and systemic risk. *Science advances*, 2(5):e1501495, 2016.

[22] Cédric Villani. *Topics in optimal transportation.* Number 58. American Mathematical Soc., 2003.

[23] Chi Wang, Edmond Jonckheere, and Reza Banirazi. Wireless network capacity

versus Ollivier-Ricci curvature under heat-diffusion (HD) protocol. In *2014 American Control Conference*, pages 3536–3541. IEEE, 2014.