Differentially Private Formation Control: Privacy and Network Co-Design

Calvin Hawkins and Matthew Hale*

***Abstract*—As multi-agent systems proliferate, there is increasing demand for coordination protocols that protect agents' sensitive information while allowing them to collaborate. To help address this need, this paper presents a distributed differentially private formation control framework. Agents' state trajectories are protected using differential privacy, which is a statistical notion of privacy that protects data by adding noise to it. We provide a private formation control implementation and analyze tradeoffs between privacy level, system performance, and connectedness of the network's communication topology. These trade-offs are used to formulate a co-design optimization problem to select the optimal communication topology and privacy parameters for a network running differentially private formation control. We prove that this problem is convex, and thus it can be solved efficiently with standard computational tools. We also calculate a closed form solution for the steady-state error covariance matrix for private formations and analyze how the lack of a central aggregator affects performance of differentially private formation control. Simulation results illustrate the scalability of our proposed privacy/network co-design problem to large multi-agent networks.**

I. Introduction

Multi-agent systems, such as robotic swarms and social networks, require agents to share information to collaborate. In some cases, the information shared between agents may be sensitive. For example, self-driving cars share location data to be routed to a destination. Geo-location data and other data streams can be quite revealing about users and sensitive data should be protected, though this data must still be useful for multi-agent coordination. Thus, privacy in multi-agent control must simultaneously protect agents' sensitive data while guaranteeing that privatized data enables the network to achieve a common task.

This type of privacy has recently been achieved using differential privacy. Differential privacy comes from the computer science literature, where it was originally used to protect sensitive data when databases are queried [1], [2]. Differential privacy is appealing because it is immune to post-processing and robust to side information [1]. These properties mean that privacy guarantees are not compromised by performing operations on differentially private data, and that they are not weakened by much by an adversary with additional information about data-producing agents [3].

Recently, differential privacy has been applied to dynamic systems [4]–[12]. One form of differential privacy in dynamic systems protects sensitive trajectory-valued data, and this is the notion of differential privacy used in this paper. Privacy of this form ensures that an adversary is unlikely to learn much about the state trajectory of a system by observing its outputs. In multi-agent control, this lets an agent share its outputs with other agents while protecting its state trajectory from those agents and eavesdroppers [4]–[7].

In this paper, we develop a framework for private multi-agent formation control using differential privacy. Formation control is a well-studied network control problem that can represent, e.g., robots physically assembling into geometric shapes or non-physical agents maintaining relative state offsets. For differential privacy, agents add privacy noise to their states before sharing them with other agents. The other agents use privatized states in their update laws, and then this process repeats at every time step.

Adding privacy noise makes this problem equivalent to a certain consensus protocol with measurement noises. This paper focuses on private formation control, though the methods presented can be used to design and analyze other private consensus-style protocols, which underlie many multi-agent control and optimization algorithms [13], [14]. The private formation control protocol can be implemented in a completely distributed manner, and, contrary to some other privacy approaches, it does not require a central coordinator.

Privacy is often a post-hoc concern in control systems that is incorporated only after a network and/or a controller is designed, which can make it difficult to implement. Therefore, this paper formulates a co-design problem to design a network topology and a differential privacy implementation together. This problem accounts for (i) the strength of privacy protections, (ii) the formation control error induced by privacy, and (iii) the topology of the network that runs the formation control protocol. It is shown that this problem is convex and thus solvable with conventional methods. The benefits of co-design have been illustrated for problems of security in control systems [15] and the co-design problem in this paper brings these same benefits to problems in privacy.

A preliminary version of this paper appears in [16]. This paper adds the co-design problem, closed-form solution to the steady-state formation control error covariance, new simulations, and proofs of all results. The rest of this paper is organized as follows. Section II gives graph theory and differential privacy background. Section III provides formal problem statements. In Section IV, we implement differential privacy in the formation control protocol and bound formation error in terms of system parameters. Section V compares our decentralized implementation to one with a trusted central coordinator. Section VI finds a closed-form solution for the steady-state error covariance matrix of the private formation

*The authors are with the Department of Mechanical and Aerospace Engineering, Herbert Wertheim College of Engineering, University of Florida. Emails: {calvin.hawkins,matthewhale}@ufl.edu. This work was supported by AFOSR under Grant #FA9550-19-1-0169 and by NSF CAREER grant #1943275.

control protocol. In Section VII, we define, analyze, and provide methods to solve the privacy/network co-design problem. Next, Section VIII provides simulation results, and Section IX provides concluding remarks.

II. Background and Preliminaries

In this section we briefly review the required background on graph theory and differential privacy.

A. Graph Theory Background

A graph $\mathcal{G} = (V, E)$ is defined over a set of nodes V and edges are contained in the set E. For N nodes, V is indexed over $\{1, ..., N\}$. The edge set of $\mathcal{G}$ is a subset $E \subseteq V \times V$, where the pair $(i, j) \in E$ if nodes i and j share a connection and $(i, j) \notin E$ if they do not. This paper considers undirected, weighted, simple graphs. Undirectedness means that an edge $(i, j) \in E$ is not distinguished from $(j, i) \in E$. Simplicity means that $(i, i) \notin E$ for all $i \in V$. Weightedness means that the edge $(i, j) \in E$ has a weight $w_{ij} = w_{ji} > 0$. Of particular interest are connected graphs.

Definition 1 (Connected Graph): A graph $\mathcal{G}$ is connected if, for all $i, j \in \{1, ..., N\}$, $i \neq j$, there is a sequence of edges one can traverse from node i to node j. $\triangle$

This paper uses the weighted graph Laplacian, which is defined with weighted adjacency and weighted degree matrices. The weighted adjacency matrix $A(\mathcal{G}) \in \mathbb{R}^{N \times N}$ of $\mathcal{G}$ is defined element-wise as

$$A(\mathcal{G})_{ij} = \begin{cases} w_{ij} & (i, j) \in E \\ 0 & \text{otherwise} \end{cases}.$$

Because we only consider undirected graphs, $A(\mathcal{G})$ is symmetric. The weighted degree of node $i \in V$ is defined as $d_i = \sum_{j|(i,j)\in E} w_{ij}$. The maximum degree is $d_{max} = \max_i d_i$. The degree matrix $D(\mathcal{G}) \in \mathbb{R}^{N \times N}$ is the diagonal matrix $D(\mathcal{G}) = \text{diag}(d_1, ..., d_N)$. The weighted Laplacian of $\mathcal{G}$ is then defined as $L(\mathcal{G}) = D(\mathcal{G}) - A(\mathcal{G})$.

Let $\lambda_k(\cdot)$ be the k^{th} smallest eigenvalue of a matrix. By definition, $\lambda_1(L(\mathcal{G})) = 0$ for all graph Laplacians and

$$0 = \lambda_1(L(\mathcal{G})) \leq \lambda_2(L(\mathcal{G})) \leq \cdots \leq \lambda_N(L(\mathcal{G})).$$

The value of $\lambda_2(L(\mathcal{G}))$ plays a key role in this paper.

Definition 2 (Algebraic Connectivity [17]): The algebraic connectivity of a graph $\mathcal{G}$ is the second smallest eigenvalue of its Laplacian and $\mathcal{G}$ is connected if and only if $\lambda_2(L(\mathcal{G})) > 0$. $\triangle$

Node i's neighborhood set $N(i)$ is the set of all agents that agent i communicates with, denoted $N(i) = \{j \mid (i, j) \in E\}$.

B. Differential Privacy Background

This section provides a brief description of the differential privacy background needed for the remainder of the paper. More complete expositions can be found in [4], [18]. Overall, the goal of differential privacy is to make similar pieces of data appear approximately indistinguishable from one another. Differential privacy is appealing because its privacy guarantees are immune to post-processing [18]. For example, private data can be filtered without threatening its privacy guarantees [4], [19]. More generally, arbitrary post-hoc computations on private data do not harm differential privacy. In addition, after differential privacy is implemented, an adversary with complete knowledge of the mechanism used to implement privacy has no advantage over another adversary without mechanism knowledge [1], [2].

In this paper we use differential privacy to privatize state trajectories of mobile autonomous agents. We consider vector-valued trajectories of the form $Z = (Z(1), Z(2), ..., Z(k), ...)$, where $Z(k) \in \mathbb{R}^d$ for all k. The ℓ_2 norm of Z is defined as $\|Z\|_{\ell_2} = \left(\sum_{k=1}^{\infty} \|Z(k)\|_2^2\right)^{\frac{1}{2}}$, where $\|\cdot\|_2$ is the ordinary 2-norm on $\mathbb{R}^d$.

We consider privacy over the set of trajectories

$$\tilde{\ell}_2^d = \{Z \mid \|Z(k)\|_2 < \infty \text{ for all } k\}.$$

This set is similar to the ordinary ℓ_2-space, except that the entire trajectory need not have finite ℓ_2-norm. Instead, only each entry of a trajectory must have finite 2-norm in $\mathbb{R}^d$. Thus, the set $\tilde{\ell}_2^d$ contains trajectories that do not converge, which admits a wide variety of trajectories seen in control systems. Consider a network of N agents, where agent i's state trajectory is denoted by x_i. The k^{th} element of agent i's state trajectory is $x_i(k) \in \mathbb{R}^d$ for $d \in \mathbb{N}$, and agent i's state trajectory belongs to $\tilde{\ell}_2^d$.

Differential privacy is defined with respect to an adjacency relation. We provide privacy to single agents' state trajectories (rather than collections of trajectories as in some other works), which enables agents to privatize all information before it is ever shared. Thus, our choice of adjacency relation is defined for single agents.

Definition 3 (Adjacency [5]): Fix an adjacency parameter $b_i > 0$ for agent i. $\text{Adj}_{b_i} : \tilde{\ell}_2^d \times \tilde{\ell}_2^d \to \{0, 1\}$ is defined as

$$\text{Adj}_{b_i}(v_i, w_i) = \begin{cases} 1 & \|v_i - w_i\|_{\ell_2} \leq b_i \\ 0 & \text{otherwise.} \end{cases} \quad \triangle$$

In words, two state trajectories that agent i could produce are adjacent if and only if the ℓ_2-norm of their difference is upper bounded by b_i. This means that every state trajectory within distance b_i from agent i's actual state trajectory must be made approximately indistinguishable from it to enforce differential privacy.

To calibrate differential privacy's protections, agent i selects privacy parameters ϵ_i and δ_i. Typically, $\epsilon_i \in [0.1, \ln 3]$ and $\delta_i \leq 0.01$ for all i [5]. The value of δ_i can be regarded as the probability that differential privacy fails for agent i, while ϵ_i can be regarded as the information leakage about agent i.

The implementation of differential privacy in this work provides differential privacy for each agent individually. This will be accomplished by adding noise to sensitive data directly, an approach called "input perturbation" privacy in the literature [20]. Noise is added by a privacy mechanism, which is a randomized map. We next provide a formal definition of differential privacy. First, fix a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We consider outputs in $\tilde{\ell}_2^d$ and use a σ-algebra over $\tilde{\ell}_2^d$, denoted Σ_2^d [21].

Definition 4 (Differential Privacy): Let $\epsilon_i > 0$ and $\delta_i \in [0, \frac{1}{2})$ be given. A mechanism $M : \tilde{\ell}_2^d \times \Omega \to \tilde{\ell}_2^d$ is (ϵ_i, δ_i)-differentially private if, for all adjacent $x_i, x_i' \in \tilde{\ell}_2^d$, we have

$$\mathbb{P}[M(x_i) \in S] \leq e^{\epsilon_i}\mathbb{P}[M(x_i') \in S] + \delta_i \text{ for all } S \in \Sigma_2^d. \quad \triangle$$

The Gaussian mechanism will be used to implement differential privacy in this work. The Gaussian mechanism adds zero-mean i.i.d. noise drawn from a Gaussian distribution pointwise in time. Stating the required distribution uses the Q-function, defined as $Q(y) = \frac{1}{\sqrt{2\pi}}\int_y^\infty e^{-\frac{z^2}{2}}dz$.

Lemma 1 (Gaussian Mechanism [4]): Let $b_i > 0$, $\epsilon_i > 0$, and $\delta_i \in (0, \frac{1}{2})$ be given, fix the adjacency relation Adj_{b_i}, and let $x_i \in \tilde{\ell}_2^d$. The Gaussian mechanism for (ϵ_i,δ_i)-differential privacy takes the form $\tilde{x}_i(k) = x_i(k) + v_i(k)$. Here v_i is a stochastic process with $v_i(k) \sim \mathcal{N}(0, \Sigma_{v_i})$, where $\Sigma_{v_i} = \sigma_i^2 I_d$ with $\sigma_i \geq \frac{b_i}{2\epsilon_i}(K_{\delta_i} + \sqrt{K_{\delta_i}^2 + 2\epsilon_i})$ and $K_{\delta_i} = Q^{-1}(\delta_i)$. This mechanism provides (ϵ_i,δ_i)-differential privacy to x_i. ■

For convenience, let $\kappa(\delta_i, \epsilon_i) = \frac{1}{2\epsilon_i}(K_{\delta_i} + \sqrt{K_{\delta_i}^2 + 2\epsilon_i})$. We next formally define the problems that are the focus of the rest of the paper.

III. Problem Formulation

In this section we state and analyze the differentially private formation control problem. We begin with the problem statement itself, then elaborate on the underlying technical details.

A. Problem Statement

Problem 1: Consider a network of N agents with communication topology modeled by the undirected, simple, connected, and weighted graph $\mathcal{G}$. Let $x_i(k) \in \mathbb{R}^d$ be agent i's state at time k, $N(i)$ be agent i's neighborhood set, $\gamma > 0$ be a stepsize, and w_{ij} be a positive weight on the edge $(i, j) \in E$. We define $\Delta_{ij} \in \mathbb{R}^d$ for all $(i, j) \in E$ as the desired relative distance between agents i and j. Do each of the following:

i. Implement the formation control protocol
$$x_i(k+1) = x_i(k) + \gamma \sum_{j \in N(i)} w_{ij}(x_j(k) - x_i(k) - \Delta_{ij}), \tag{1}$$
in a differentially private, decentralized manner.
ii. Bound the performance of the network in terms of the privacy parameters of each agent and the algebraic connectivity of the underlying communication topology; use those bounds to quantify trade offs between privacy, connectedness, and network performance.
iii. Analyze how the lack of a central aggregator affects performance and quantify the cost of the absence of a central aggregator.
iv. Find a closed form solution for the steady-state error covariance matrix.
v. Formulate an optimization problem to co-design the communication topology and privacy parameters of the network. $\triangle$

Before solving Problem 1, we give the necessary definitions for formation control. First, we define agent- and network-level dynamics. Then, we detail how each agent will enforce differential privacy. Lastly, we explain how differentially private communications affect the performance of a formation control protocol and how to quantify the quality of a formation.

B. Multi-Agent Formation control

The goal of formation control is for agents in a network to assemble into some geometric shape or set of relative states. Multi-agent formation control is a well-researched problem and there are several mathematical formulations one can use to achieve similar results [13], [22]–[27]. We will define relative distances between agents that communicate and the control objective is for all agents to maintain the relative distances to each of their neighbors. This approach is similar to that of [24] and the translationally invariant formations in [13].

For the formation to be feasible, we require $\Delta_{ij} = -\Delta_{ji}$ for all $(i, j) \in E$. The network control objective is driving $\lim_{k\to\infty}(x_j(k) - x_i(k)) = \Delta_{ij}$ for all $(i, j) \in E$. There is an infinite set of points that can be in formation; the formation can be centered around any point in $\mathbb{R}^d$ and meet the control requirement, i.e., we allow formations to be translationally invariant [13].

Now we define the agents' update law. Let $\{p_1, ..., p_N\}$ be any collection of points in formation such that $p_j - p_i = \Delta_{ij}$ for all $(i, j) \in E$ and let $p = (p_1^T, \ldots, p_N^T)^T \in \mathbb{R}^{Nd}$ be the network-level formation specification. We consider the formation control protocol in (1). At the network level, let $x(k) = (x_1(k)^T, ..., x_N(k)^T)^T \in \mathbb{R}^{Nd}$ and let $\bar{x}(k) = x(k) - p$. Then we analyze

$$\bar{x}(k+1) = ((I_N - \gamma L(\mathcal{G})) \otimes I_d)\,\bar{x}(k). \tag{2}$$

Letting $P = ((I_N - \gamma L(\mathcal{G})) \otimes I_d)$, we may write $\bar{x}(k+1) = P\bar{x}(k)$.

For analysis, we define the Markov chain[1] in dimension d as $P_d = I_N - \gamma L(\mathcal{G})$. Also let $\bar{x}_{i_{[l]}}$ be the l'th scalar element of $\bar{x}_i$. Then

$$\bar{x}_{[l]} = [\bar{x}_{1_{[l]}}, \ldots, \bar{x}_{N_{[l]}}]^T \tag{3}$$

is the vector of each agent's state in the l'th dimension. With $P = P_d \otimes I_d$, the protocol $\bar{x}(k+1) = P\bar{x}(k)$ is equivalent to running the protocol

$$\bar{x}_{[l]}(k+1) = P_d\bar{x}_{[l]}(k) \tag{4}$$

for all $l \in \{1, \ldots, d\}$. In this form, we have the following convergence result.

Lemma 2 ([27], Theorem 2): If $\mathcal{G}$ is connected, P_d is doubly stochastic, and $\gamma \in (0, \frac{1}{d_{max}})$, then the protocol in (4) reaches consensus asymptotically and $\bar{x}_{[l]}(k) \to \mathbb{1}^T\frac{1}{N}\bar{x}_{[l]}(0)\mathbb{1}$ for all $l \in \{1, \ldots, d\}$. ■

Because the protocol in (2) reaches consensus over $\bar{x}_l$ for all l, it solves the translationally invariant formation control problem in $\mathbb{R}^d$ [13].To achieve a formation, each agent must share its state with its neighborhood at every time step, thus revealing its state trajectory to other agents, adversaries, and eavesdroppers. We next describe the means to make this protocol differentially private.

[1]With an abuse of terminology, we will sometimes conflate a Markov chain with its matrix of transition probabilities.

C. Private Communications

Agent j starts by selecting privacy parameters $\epsilon_j > 0$, $\delta_j \in (0, \frac{1}{2})$, and adjacency relation Adj_{b_j} with $b_j > 0$. Agent j then privatizes its state trajectory x_j with the Gaussian mechanism. Let $\tilde{x}_j$ denote the differentially private version of x_j, where, pointwise in time, $\tilde{x}_j(k) = x_j(k) + v_j(k)$, with $v_j(k) \sim \mathcal{N}(0, \Sigma_{v_j})$, where $\Sigma_{v_j} = \sigma_j^2 I_d$ and $\sigma_j \geq \kappa(\delta_j, \epsilon_j) b_j$. Thus agent j keeps the trajectory x_j differentially private. Agent j then shares $\tilde{\tilde{x}}_j(k) = \tilde{x}_j(k) - p_j$, which is also differentially private because subtracting p_j is merely post-processing [18].

D. Private Formation Control

When each agent is sharing differentially private information, the node-level formation control protocol becomes

$$\bar{x}_i(k+1) = \bar{x}_i(k) + \gamma \sum_{j \in N(i)} w_{ij}(\tilde{\tilde{x}}_j(k) - \tilde{\tilde{x}}_i(k)), \quad (5)$$

where agent i uses $\bar{x}_i$ outside of the sum rather than $\tilde{\tilde{x}}_i$ because it always has access to its own unprivatized state, but uses $\tilde{\tilde{x}}_i$ in the sum to ensure there is randomness in its dynamics at every point in time for privacy. Equation (5) solves Problem 1.i. The stochastic nature of this protocol implies that agents no longer exactly reach a formation, and, in particular, their states will never exactly converge to a steady-state value.

In this work we use the total mean square error of the network at steady-state, e_{ss}, to quantify performance. Since we are running d identical copies of (4) we can compute the mean square error in dimension l and then multiply by d to compute e_{ss}. For analysis, let $x_{[l]} = [x_{1_{[l]}}, \ldots, x_{N_{[l]}}]^T$ and $p_{[l]} = [p_{1_{[l]}}, \ldots, p_{N_{[l]}}]^T$. Then to analyze performance, let $\beta_{[l]}(k) := \frac{1}{N}\mathbb{1}^T \bar{x}_{[l]}(k)\mathbb{1} = \frac{1}{N}\mathbb{1}^T x_{[l]}(k)\mathbb{1} + p_{[l]} - \frac{1}{N}\mathbb{1}^T p_{[l]}\mathbb{1}$, which is the state vector the protocol in (4) would converge to with initial state $x_{[l]}(k)$ and without privacy. Also let

$$e_{[l]}(k) = x_{[l]}(k) - \beta_{[l]}(k), \quad (6)$$

which is the distance of the current state to the state the protocol would converge to without differential privacy. To quantify the effects of privacy on the network as a whole, let $e_{\text{agg},l}(k) := \frac{1}{N}\|E[e_{[l]}^2(k)]\|_2^2$ be the aggregate error of the network in dimension l. Then e_{ss} is given by

$$e_{ss} := d \limsup_{k \to \infty} e_{\text{agg},l}(k). \quad (7)$$

Problem 1.ii requires us to quantify the relationship between privacy, encoded by (ϵ_i, δ_i); performance, encoded by e_{ss}; and topology, encoded by λ_2. These tradeoffs are the subject of the next section.

IV. Performance of Differentially Private Formation Control

In this section we solve Problem 1.ii. First, we show how the private formation control protocol can be modeled as a Markov chain with desirable properties. Then, we solve Problem 1.ii by deriving performance bounds that are functions of the underlying graph topology and each agent's privacy parameters.

A. Formation Control as a Markov Chain

The solution to Problem 1.i, given in (5), takes the form of a consensus protocol with Gaussian i.i.d. noise perturbing each agent's state, which has been previously studied in [22]. Work in [22] uses the fact that a consensus protocol subject to noise can be modeled as a Markov chain. We first show that the differentially private formation protocol can be represented as a Markov chain by expanding $\tilde{\tilde{x}}_j(k)$ and $\tilde{\tilde{x}}_i(k)$ in Equation (5), which yields

$$\bar{x}_i(k+1) = \bar{x}_i(k) + \gamma \sum_{j \in N(i)} w_{ij}(\bar{x}_j(k) + v_j(k) - \bar{x}_i(k) - v_i(k)). \quad (8)$$

Let $v(k) = [v_1(k)^T, \ldots, v_N(k)^T]^T$ and let $\mathbb{0}_d \in \mathbb{R}^{d \times d}$ be the matrix of all zeros. Then $v(k) \sim \mathcal{N}(0, \Sigma_v)$, where

$$\Sigma_v = \begin{bmatrix} \Sigma_{v_1} & \mathbb{0}_d & \ldots & \mathbb{0}_d \\ \mathbb{0}_d & \Sigma_{v_2} & \ldots & \mathbb{0}_d \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{0}_d & \mathbb{0}_d & \ldots & \Sigma_{v_N} \end{bmatrix} \in \mathbb{R}^{Nd \times Nd}.$$

For the purposes of analysis, we will consider equivalent network-level dynamics given as follows.

Lemma 3: Let agents use the communication graph $\mathcal{G}$ with weighted Laplacian $L(\mathcal{G})$. Then (8) can be represented at the network level as $\bar{x}(k+1) = P\bar{x}(k) + z(k)$, where $P = (I_N - \gamma L(\mathcal{G})) \otimes I_d$, $z(k) \sim \mathcal{N}(0, \Sigma_z)$, and $\Sigma_z = \gamma^2 (L(\mathcal{G}) \otimes I_d)\Sigma_v (L(\mathcal{G}) \otimes I_d)$.

Proof: See Appendix A. ∎

Notice that Σ_z is not diagonal in general and thus the elements of $z(k)$ are not independent.

For analysis, we use the network-level update law

$$\bar{x}(k+1) = P\bar{x}(k) + z(k), \quad (9)$$

with $P = P_d \otimes I_d$. Before stating our main results on performance, we first define the conditions under which a network modeled by an undirected, weighted, connected graph can be modeled as a Markov chain and establish the properties of this Markov chain.

Lemma 4: For an undirected, weighted, simple, connected graph $\mathcal{G}$, let $\gamma > 0$ be given. If for all i the graph weights are designed such that $\gamma \sum_{j \in N(i)} w_{ij} < 1$, then the matrix $P_d = I_N - \gamma L(\mathcal{G})$ is doubly stochastic, which implies $P = P_d \otimes I_d$ is doubly stochastic.

Proof: See [28, Lemma 4] ∎

Graph Laplacian properties can be used to make stronger statements. In particular, the Laplacian of an undirected graph with symmetric weights is always symmetric, and therefore P_d and P are symmetric. Let the stationary distribution of P_d be denoted by π, which satisfies $\pi^T P_d = \pi^T$. With the symmetry of P_d we have the following explicit form for π.

Lemma 5: If P_d is symmetric, then its stationary distribution is $\pi = \frac{1}{N}\mathbb{1}$.

Proof: See [29, Chapter 4]. ∎

Furthermore, we can make a stronger statement about P_d.

Lemma 6: Let $\gamma \in \left(0, \frac{1}{d_{max}}\right)$. If $\mathcal{G}$ is connected, simple, and finite, then P_d is irreducible, aperiodic, positive recurrent, and reversible.

Proof: See [28, Lemma 6] ■

Lemmas 4-6 allow us to develop bounds on the steady state error in (7) using the following Lemma based on developments in [22]. That work details several specific cases of consensus protocols with noise vectors of the form in (9). Those results are derived in terms of the hitting times of a Markov chain. Given a Markov chain with transition matrix P, the hitting time from node i to j, denoted $H_P(i \to j)$, is the expected time when the chain reaches node j for the first time when starting at node i. For this paper we are only interested in the results when P_d is symmetric, which take the following form.

Lemma 7: If P_d is irreducible, aperiodic, and reversible, then e_{ss} is bounded via

$$e_{\rm ss} \leq (\max_{i,j} s_{ij}) K(P_d^2) d,$$

where $K(P_d^2)$ is the Kemeney constant of the Markov chain with transition matrix P_d^2 and s_{ij} is the $i^{th}j^{th}$ element of Σ_z.

Proof: See Appendix B. ■

Given that $P_d = I_N - \gamma L(\mathcal{G})$, we wish to relate $K(P_d^2)$ to the agents' graph topology encoded in $L(\mathcal{G})$. We do so with the following bound.

Lemma 8: Let P_d be the transition matrix of a finite, irreducible, and reversible Markov chain. Let $\lambda_2(P_d)$ be the second largest eigenvalue of P_d. Then the Kemeney constant $K(P_d)$ is bounded via $\frac{N-1}{2} < K(P_d) \leq \frac{N-1}{1-\lambda_2(P_d)}$, and, for $P_d = I_N - \gamma L(\mathcal{G})$,

$$\frac{N-1}{2} < K(P_d^2) \leq \frac{N-1}{1-(1-\gamma\lambda_2(L(\mathcal{G})))^2}.$$

Proof: See [28, Lemma 8] ■

We are now ready to state our main results. Lemmas 7 and 8 can be used to find a bound on e_{ss} in terms of the privacy parameters ϵ_i, δ_i and the algebraic connectivity of the underlying communication topology, which solves Problem 1.ii.

Theorem 1: Consider the d-dimensional network-level private formation control protocol $\bar{x}(k+1) = P\bar{x}(k) + z(k)$. If $\gamma \sum_{j\in N(i)} w_{ij} < 1$, $\gamma \in \left(0, \frac{1}{d_{max}}\right)$, $\mathcal{G}$ is connected and undirected with symmetric weights, and $\sigma_i \geq \kappa(\delta_i, \epsilon_i) b_i$ for all i, then e_{ss} is upper-bounded by

$$e_{ss} \leq \frac{\gamma(N-1)^2 \max_i \kappa(\delta_i,\epsilon_i)^2 b_i^2 d}{\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))}.$$

Proof: With Lemma 7, $e_{\rm ss} \leq (\max_{i,j} s_{ij}) K(P_d^2) d$. Then using Lemma 8 to upper bound $K(P_d^2)$ gives

$$e_{ss} \leq \frac{(\max_{i,j} s_{ij})\,(N-1)d}{\gamma\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))}. \tag{10}$$

Now consider s_{ij}, which is the $i^{th}j^{th}$ entry of $\Sigma_z = \gamma^2 A(\mathcal{G})\Sigma_v A(\mathcal{G})$ from Lemma 3. It can be shown that the diagonal terms of this product are given by $s_{ii} = \gamma^2 \sum_{j\in N(i)} w_{ij}^2 \sigma_j^2$, and the off-diagonal terms are given as $s_{ij} = \gamma^2 \sum_{l\in N(i), l\neq j} w_{il} w_{jl} \sigma_l^2$. Then, because $w_{ij} \in (0,1)$,

$$\max_{i,j} s_{ij} \leq \gamma^2 \max_i \left(|N(i)| \max_{j\in N(i)} \sigma_j^2 \right).$$

For N agents $|N(i)| \leq N-1$, $\sigma_i^2 \geq \kappa(\delta_i,\epsilon_i)^2 b_i^2$, and $\max_i[\max_{j\in N(i)} \sigma_j^2] = \max_i \sigma_i^2$, which gives

$$\gamma^2 \max_i \left[|N(i)| \max_{j\in N(i)} \sigma_j^2 \right] \leq \gamma^2 (N-1) \max_i \kappa(\delta_i,\epsilon_i)^2 b_i^2. \tag{11}$$

Plugging (13) into (12) completes the proof. ■

We can simplify Theorem 1 when each agent has the same privacy parameters, i.e., $\epsilon_i = \epsilon$, $\delta_i = \delta$, and $b_i = b$ for all i. Consider the case where $\sigma = \kappa(\delta,\epsilon) b$ so that each agent adds the minimum amount of noise needed to attain (ϵ,δ)-differential privacy. Then we have the following.

Corollary 1 (Homogeneous Privacy Parameters): Let all conditions of Theorem 1 hold, and let each agent in the network have privacy parameters ϵ and δ and the adjacency parameter b. Then

$$e_{ss} \leq \frac{\gamma\kappa(\delta,\epsilon)^2 b^2 (N-1)^2 d}{\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))}.$$

■

V. Cost of No Trust

The solution to Problem 1.i is decentralized, and, in contrast to some other works on privacy, it does not require a central aggregator. The setup in Problem 1.i does not require agents to trust any external entity with sensitive information, and it enables them to privatize all information before it is shared. While the lack of a central aggregator improves privacy in this way, this absence is also generally understood to provide diminished performance. In order to establish and quantify this principle for private formation control, in this section we analyze the performance of a network with a central aggregator introduced; comparisons between this performance and that in Theorem 1 will enable the solution of Problem 1.iv. More specifically, we formally define a privacy mechanism for when a central aggregator is present, show that the central aggregator can improve performance, and find the magnitude of changes required to make a network without an aggregator achieve equal or better performance. We refer to this magnitude as the cost of no trust.

When an aggregator is introduced, agents send their states to the aggregator, the aggregator then computes $q_{\bar{x}}(N(i)) = \sum_{j\in N(i)} w_{ij}\bar{x}_j(k)$ for each i, then adds noise to it to implement privacy, and sends the result to the corresponding agent. This implementation of privacy is referred to as output perturbation. With this definition of $q_{\bar{x}}(N(i))$, the formation control protocol in (1) takes the form

$$\bar{x}_i(k+1) = \bar{x}_i(k) + \gamma \left(q_{\bar{x}}(N(i)) - \sum_{j\in N(i)} w_{ij}\bar{x}_i(k) \right).$$

For the aggregator to implement $(\epsilon,\delta)-$differential privacy, we must first design a privacy mechanism.

A. Privacy Mechanism for Output Perturbation

To formally define an output perturbation differential privacy mechanism, we formulate an appropriate adjacency relationship, determine the sensitivity of the information shared

by the aggregator, and use this information to calibrate the Gaussian mechanism presented in Lemma 1. We first define an adjacency relationship over collections of trajectories.

Definition 5 (Output Perturbation Adjacency): Let $\xi = \{x_1, x_2, \ldots, x_N\}$ and $\xi' = \{x'_1, x'_2, \ldots, x'_N\}$ be two collections of state trajectories. Then, for a fixed adjacency parameter $b > 0$, we define the adjacency relation

$$\text{Adj}_b(\xi, \xi') = \begin{cases} 1 & \exists i \text{ s.t. } x_j = x'_j \ \forall j \neq i \ \ \& \ \ \|x_i - x'_i\|_{\ell_2} \leq b \\ 0 & \text{otherwise} \end{cases}. \qquad \triangle$$

In words, two collections of trajectories are adjacent if they differ in at most one entry, and that difference is bounded in the ℓ_2-norm by b. With this adjacency relation, we define the output perturbation privacy mechanism.

Theorem 2 (Output Perturbation Mechanism): Fix $b > 0$ and consider the adjacency relation in Definition 5. The Gaussian mechanism in Lemma 1 is (ϵ, δ)-differentially private with $\sigma \geq \bar{w} b \kappa(\delta, \epsilon)$, where $\bar{w} \geq w_{ij}$ for all $(i, j) \in E$, i.e., $\bar{w}$ is an upper bound on all weights. Thus, when the aggregator sends $q_{\bar{x}}(N(i))$ to agent i, it is (ϵ, δ)-differentially private.

Proof: See Appendix D. ∎

B. Cost of No Trust

In general, introducing a central aggregator can improve performance while still ensuring the same level of privacy. With output perturbation, the agents first share their sensitive state information with the aggregator, then the aggregator sends noisy functions of those states to the agents. Of course, the agents or network designer must trust that the central aggregator does not have ill intent and will faithfully implement differential privacy.

When using input perturbation, each of the $\bar{x}_j(k)$ terms is privatized individually by each agent and their sum in the formation control protocol in (5) contains several noisy terms. In output perturbation, the sum is computed and noise is added to it once. This difference in the addition of noise results in the difference in performance.

We now analyze the cost of no trust by comparing a network using input perturbation and a network using output perturbation. Consider two graphs, $\mathcal{G}_{IP}$ and $\mathcal{G}_{OP}$, where agents in $\mathcal{G}_{IP}$ use input perturbation privacy and agents in $\mathcal{G}_{OP}$ use output perturbation and a central aggregator. Both networks use identical, homegenous privacy parameters ϵ, δ, and adjacency parameter b. The two networks have identical vertex sets but can have different edge sets and thus different algebraic connectivities. Let $\lambda_2(L(\mathcal{G}_{IP})) = \lambda_{IP}$ and $\lambda_2(L(\mathcal{G}_{OP})) = \lambda_{OP}$. Let e_{IP} be the steady-state error of the input perturbation network and e_{OP} be the steady-state error of the output perturbation network. With Corollary 1, these satisfy

$$e_{IP} \leq \frac{\gamma \kappa(\delta, \epsilon)^2 b^2 (N-1)^2 d}{\lambda_{IP}(2 - \gamma \lambda_{IP})},$$

and

$$e_{OP} \leq \frac{\gamma \kappa(\delta, \epsilon)^2 b^2 (N-1)^2 \bar{w}^2 d}{\lambda_{OP}(2 - \gamma \lambda_{OP})},$$

respectively.

If the two networks were to have identical communication topologies, i.e., where $\lambda_{OP} = \lambda_{IP}$. In this case $e_{OP} < e_{IP}$ because the steady-state errors only vary by a factor of $\bar{w}^2$ and $\bar{w} \in (0, 1)$. Thus, with fixed privacy parameters, the only way that input perturbation will outperform output perturbation is if the input perturbation network is sufficiently connected. Thus, we quantify the "cost of no trust" as how much larger λ_{IP} must be than λ_{OP} to provide the same (or better) performance. The following theorem quantifies the cost of no trust in this sense.

Theorem 3: With $\mathcal{G}_{IP}$ and $\mathcal{G}_{OP}$ defined as above, let $\lambda_{IP} = \lambda_{OP} + \theta$ for some $\theta \geq 0$ Then $e_{IP} \leq e_{OP}$ if θ satisfies

$$\max\left\{0, \frac{\rho_1 - \sqrt{\rho_2}}{2\gamma}\right\} \leq \theta \leq \min\left\{N - \lambda_{OP}, \frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}\right\},$$

where $\rho_1 = 2 - 2\gamma\lambda_{OP}$ and

$$\rho_2 = (2\gamma\lambda_{OP} - 2)^2 + \frac{4\gamma\lambda_{OP}(\gamma\lambda_{OP} - 2)}{\bar{w}^2}.$$

Proof: First, consider $\frac{\rho_1 - \sqrt{\rho_2}}{2\gamma} \leq \theta \leq \frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}$. Note that $\frac{\rho_1 - \sqrt{\rho_2}}{2\gamma}$ and $\frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}$ are the roots of a quadratic equation, after plugging in ρ_1 and ρ_2 and simplifying, this quadratic equation is given by

$$\Upsilon(\theta) = \gamma\theta^2 + (2\gamma\lambda_{OP} - 2)\theta - \frac{(\bar{w}^2 - 1)(\lambda_{OP}(2 - \gamma\lambda_{OP}))}{\bar{w}^2}.$$

The condition $\frac{\rho_1 - \sqrt{\rho_2}}{2\gamma} \leq \theta \leq \frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}$ and $\gamma > 0$ imply that $\Upsilon(\theta) \leq 0$. Simplifying this inequality gives

$$\frac{1}{(\lambda_{OP} + \theta)(2 - \gamma(\lambda_{OP} + \theta))} \leq \frac{\bar{w}^2}{\lambda_{OP}(2 - \gamma\lambda_{OP})},$$

then multiplying both sides by $\gamma\kappa(\delta, \epsilon)^2 b^2 (N-1)^2 d$ gives

$$\frac{\gamma \kappa(\delta, \epsilon)^2 b^2 (N-1)^2 d}{(\lambda_{OP} + x)(2 - \gamma(\lambda_{OP} + \theta))} \leq \frac{\gamma \kappa(\delta, \epsilon)^2 b^2 (N-1)^2 \bar{w}^2 d}{\lambda_{OP}(2 - \gamma\lambda_{OP})},$$

which is precisely $e_{IP} \leq e_{OP}$. With $\lambda_{IP} = \lambda_{OP} + \theta$, we require $\theta \geq 0$ since we are only interested in $\lambda_{IP} \geq \lambda_{OP}$ and $\theta \leq N - \lambda_{OP}$ to ensure that $\lambda_{IP} \leq N$ which must be true for graphs on N nodes. Thus projecting $\frac{\rho_1 - \sqrt{\rho_2}}{2\gamma} \leq \theta \leq \frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}$ to the interval $[0, N - \lambda_{OP}]$ gives $\max\{0, \frac{\rho_1 - \sqrt{\rho_2}}{2\gamma}\} \leq \theta \leq \min\{N - \lambda_{OP}, \frac{\rho_1 + \sqrt{\rho_2}}{2\gamma}\}$. ∎

VI. Closed-Form for Steady-State Error Covariance

In this section we find a closed form solution for the steady-state error covariance matrix of the differentially private formation protocol. The steady-state error covariance matrix, which we denote $\Sigma_{ss} = \lim_{k \to \infty} E[e(k)e(k)^T]$, is useful as it enables a finer-grained analysis of system performance when compared to the scalar e_{ss}, specifically it reveals how the error of each agent is coupled with the rest of the network. Obtaining Σ_{ss} is also useful as it enables the computation of the scalar steady-state error we have been using to quantify performance via $e_{ss} = \frac{1}{N}\text{Tr}(\Sigma_{ss})$.

In [22], results of this kind are derived in terms of the hitting times of the underlying Markov chain. In this section, we provide an alternate solution using the Lyapunov equation whose

coefficients are taken directly from the transition matrix P_d and $L(\mathcal{G})$, thus precluding the need to compute hitting times. In general, Lyapunov equations are easily solved numerically, though we provide an analytical solution. Before presenting this result, we first give a preliminary result from [22].

Similar to the previous sections, we opt to analyze the protocol that is being run in each dimension of $\mathbb{R}^d$ and then make conclusions about the steady-state error covariance for the protocol in $\mathbb{R}^d$. Recall that to implement (9), each agent runs the protocol

$$\bar{x}_{[l]}(k+1) = P_d \bar{x}_{[l]}(k) + z_{[l]}(k),$$

for $l \in \{1, \dots, d\}$, with $P_d = I_N - \gamma L(\mathcal{G})$, $\bar{x}_{[l]}$ defined by (3), $\Sigma_{z_{[l]}} = \gamma^2 L(\mathcal{G}) \Sigma_{v_{[l]}} L(\mathcal{G})$ from Lemma 3, and error in the l^{th} dimension, $e_{[l]}(k)$, defined by (6). Since $e_{[l]}(k)$ is a zero mean random variable, the error covariance matrix in the l^{th} dimension at time k is defined by $\Sigma_{e_{[l]}}(k) = E[e_{[l]}(k) e_{[l]}(k)^T]$. Let $J = \frac{1}{N} \mathbb{1}\mathbb{1}^T$. Then we have the following.

Lemma 9 (Error Dynamics [22]): The matrix $P_d - J$ has eigenvalues strictly less than 1. Moreover, $e_{[l]}(k)$ evolves via

$$e_{[l]}(k+1) = (P_d - J) e_{[l]}(k) + (I_N - J) z_{[l]}(k),$$

$\Sigma_{e_{[l]}}(k)$ evolves according to

$$\Sigma_{e_{[l]}}(k+1) = (P_d - J) \Sigma_{e_{[l]}}(k) (P_d - J) + (I_N - J) \Sigma_z (I_N - J),$$

and $\Sigma_{e_{[l]}}(k)$ can be determined recursively by

$$\Sigma_{e_{[l]}}(k) = \sum_{i=0}^{k-1} (P_d - J)^i (I_N - J) \Sigma_z (I_N - J) (P_d - J)^i. \quad \blacksquare$$

We now show that the steady-state error covariance matrix is a solution to a discrete-time Lypaunov equation. Let $\Sigma_{ss_{[l]}} = \lim_{k \to \infty} \Sigma_{e_{[l]}}(k)$ be the steady-state error covariance matrix in dimension l. An analytical solution to the Lyapunov equation can be found using the vectorization operator defined by $\text{vec}(B) = [b_{11}, \dots, b_{m1}, b_{12}, \dots, b_{m2}, \dots, b_{1n}, \dots, b_{mn}]^T$ where $B \in \mathbb{R}^{m \times n}$ and the i^{th} j^{th} entry is b_{ij} [30].

Theorem 4 (Solving for $\Sigma_{ss_{[l]}}$): Let

$$Q_{[l]} = \gamma^2 (I_N - J) L(\mathcal{G}) \Sigma_{v_{[l]}} L(\mathcal{G}) (I_N - J),$$

with $\Sigma_{v_{[l]}} = \text{diag}(\sigma_1^2, \dots, \sigma_n^2)$ and $\sigma_j \geq \kappa(\delta_j, \epsilon_j) b_j$. Then, $\Sigma_{ss_{[l]}}$ is a solution to the discrete time Lyapunov equation.

$$\Sigma_{ss_{[l]}} = Q_{[l]} + (P_d - J) \Sigma_{ss_{[l]}} (P_d - J).$$

Solving using the vectorization operator gives

$$\text{vec}(\Sigma_{ss_{[l]}}) = \gamma^2 R^{-1} M \text{vec}\left(\Sigma_{v_{[l]}}\right),$$

where $R = I_{N^2} - (P_d - J) \otimes (P_d - J)$, and $M = ((I_N - J) \otimes (I_N - J)) (L(\mathcal{G}) \otimes L(\mathcal{G}))$.

Proof: See Appendix E. ∎

Since agents run an identical copy of the same controller in each dimension, we have $\Sigma_{ss} = \Sigma_{e_{ss_{[l]}}} \otimes I_d$. In the previous sections we have been using the scalar e_{ss} to quantify performance, and here we note that $e_{ss} = \frac{1}{N} \text{Tr}(\Sigma_{e_{ss_{[l]}}}) d$. This equality means that we have found a series of matrix operations that can be used to compute the steady-state error covariance matrix of a network executing differentially private formation control. Theorem 4 explicitly shows that the error in the network depends on the underlying communication topology through $L(\mathcal{G})$. This implies that one can exploit this dependence to design a topology that promotes lower error, and we explore this in the next section.

VII. Privacy and Network Co-design

Given the aforementioned bounds on formation error, we now focus on designing networks for performing private formation control. Our goal is to design a network and privacy scheme that meet global and network-level performance requirements, agent-level privacy requirements, and other constraints. The key tradeoff is balancing the agent-level privacy requirements with global performance. For example, if some agents use very strong privacy, global performance will be poor, even if many other agents have only weak privacy requirements. Network design thus requires balancing these tradeoffs while designing a weighted, undirected graph.

In this section, we formulate an optimization problem that takes an undirected, unweighted graph topology and several constraints as inputs and outputs the privacy parameters and edge weights that minimize formation control error. In particular, agents' privacy parameters and their graph Laplacian are the decision variables over which we optimize. Using optimization to design a network has appeared in several different contexts. In [31], the authors formulate the optimal allocation of edge weights in a network to minimize effective resistance for a given input network. In [32], an initial topology is given and an optimization problem is formulated to determine what edges should be added to optimize network coherence. There are many ways to formulate an optimal network design problem and in this work we use an approach similar to [31]: we fix an unweighted input topology and determine the allocation of weights to optimize network-level criteria while satisfying constraints.

Other works, such as [33], explore using graph Laplacian eigenvalue constraints to ensure a minimal level of connectivity in the designed graph and show that these constraints are typically convex and, we use similar methods here. Overall, this privacy and network co-design problem is formulated to incorporate constraints on privacy and performance, and we show that this leads to problems that are convex in the graph Laplacian and privacy parameters, enabling them to be solved with conventional computational tools.

Throughout this section we assume an input undirected, unweighted, simple graph Laplacian L_0 is given and we are left with choosing the edge weights and privacy parameters to minimize an objective subject to constraints. We define $\mathcal{L}(L_0)$ as the space of all weighted graph Laplacians where there is no edge between between agents i and j if there is no edge between agents i and j in L_0, and we optimize over the set $\mathcal{L}(L_0)$.

The three following subsections discuss (i) network-level design goals, which give the objective function and some constraints, (ii) agent-level requirements, which add constraints, and (iii) the convexity of the overall problem that results.

A. Network-Level Considerations

The first network-level constraint we consider is performance, measured by the steady-state formation error of the network. Mathematically, the steady-state error of the network must not exceed some specified level of steady state error e_R, i.e., we require $e_{ss} \leq e_R$. We use the upper bound on e_{ss} from Theorem 1 to provide a sufficient condition for this constraint. Applying Theorem 1, we see that

$$\frac{\gamma(N-1)^2 \max_i \kappa(\delta_i, \epsilon_i)^2 b_i^2 d}{\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))} \leq e_R$$

ensures that $e_{ss} \leq e_R$. This further incorporates each agent's privacy parameters and the underlying topology, which one expects to impact performance.

Next, we must ensure that the network's communication topology is sufficiently connected. Simply providing connectivity can be achieved by requiring that the algebraic connectivity of the graph be positive. However, one can require a greater degree of connectedness by requiring that the algebraic connectivity be larger. While connected weighted graphs can have values of algebraic connectivity that are arbitrarily small, we require a graph's algebraic connectivity to be bounded below by λ_{2_L}. One example would be to let λ_{2_L} be the alegraic connecitivity of the unweighted line graph on N nodes,the line graph is the least connected graph among connected graphs [17], and thus this constraint ensures that the performance of the weighted graphs we design is no worse than the worst-performing unweighted graph. The connectivity constraint thus takes the form $\lambda_2(L(\mathcal{G})) \geq \lambda_{2_L}$, or $-\lambda_2(L(\mathcal{G})) \leq -\lambda_{2_L}$.

With network-level constraints defined, we now define a network-level objective function. This objective function should encode the fact that a dense graph costs more. This cost may be monetary, e.g., when building a network, or it may simply encode the desire to reduce communications, e.g., when bandwidth is limited. To encode this, the cost should be small when the degree of each agent is small and large when the degrees are large. We therefore choose the cost $\vartheta \mathrm{Tr}(L(\mathcal{G}))$, where $\vartheta \in \mathbb{R}$ is a weighting factor that can be tuned.

The objective function should also depend on the privacy parameters. As the strength of privacy in the network decreases, performance increases, and this weaker privacy should result in a lower cost (constraints on privacy are addressed in the next sub-section). Weaker privacy protections come from increasing ϵ_i for all i and thus the objective function should be decreasing in ϵ_i. Therefore, we can achieve our goals of minimizing the cost of the communication and maximizing performance by minimizing the objective function

$$\Gamma(\{\epsilon_i\}_{i\in[N]}, L(\mathcal{G})) = \vartheta \mathrm{Tr}(L(\mathcal{G})) + \sum_{i\in[N]} \frac{1}{\epsilon_i^2}$$

over $L(\mathcal{G}) \in \mathcal{L}(L_0)$ and ϵ_i for all i.

If we were to minimize $\Gamma(\{\epsilon_i\}_{i\in[N]}, L(\mathcal{G}))$ with the current constraints, the optimal solution will have large ϵ_i for all i. This results in weak privacy, and implies that the network designer is forcing each agent to share their sensitive information with minimal protections. However, individual agents may have their own privacy requirements, which we incorporate in the next section.

B. Agent-Level Constraints

We let each agent specify a maximum leakage of private information that it will tolerate, which is formalized as a maximum value of the privacy parameter ϵ_i. Thus, agent i specifies some ϵ_i^{max}, and we incorporate the constraint $\epsilon_i \leq \epsilon_i^{max}$ for all i.

We also wish to allow agents that have larger degrees to be able to use stronger privacy. To interpret this, if an agent improves the connectivity of the network by having more edges, then that agent is contributing to making the graph more connected, which increases λ_2 and decreases steady-state error. In exchange for doing so, they can be provided with the flexibility to make their information more private. To formalize this, let $q_d^i : \mathbb{R} \to \mathbb{R}$ be an increasing, convex function of agent i's degree d_i, and let $q_\epsilon^i : \mathbb{R} \to \mathbb{R}$ be an increasing, convex function of agent i's privacy parameter ϵ_i; because smaller ϵ_i implies stronger privacy, the function q_ϵ^i is decreasing in the strength of agent i's privacy. Then, we implement the constraint $q_d^i(d_i) + q_\epsilon^i(\epsilon_i) \leq \rho_i$, where ρ_i is a tunable parameter.

C. Co-Design Problem and Convexity

Assembling the aforementioned objective and constraints gives the following problem.

Problem 2 (Privacy and Network Co-Design): Given an input undirected, unweighted, simple graph L_0, to co-design privacy and agents' communication topology, solve:

$$\begin{aligned}
\min_{L(\mathcal{G})\in\mathcal{L}(L_0),\{\epsilon_i\}_{i\in[N]}} \quad & \vartheta \mathrm{Tr}(L(\mathcal{G})) + \sum_{i\in[N]} \frac{1}{\epsilon_i^2} \\
\text{subject to} \quad & \frac{\gamma(N-1)^2 \max_i \kappa(\delta_i, \epsilon_i)^2 b_i^2 d}{\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))} \leq e_R \\
& q_d^i(d_i) + q_\epsilon^i(\epsilon_i) \leq \rho_i \quad \text{for all } i \\
& \epsilon_i \leq \epsilon_i^{max} \quad \text{for all } i \\
& -\lambda_2(L(\mathcal{G})) \leq -\lambda_{2_L}.
\end{aligned}$$

Now we state our main results on the convexity of Problem 2.

Theorem 5: Fix λ_{2_L} and δ_i, b_i, ϵ_i^{max}, and ρ_i for all i. Then Problem 2 is a convex optimization problem.

Proof: See Appendix F. ∎

D. Numerically Solving Problem 2

We have shown that Problem 2 is convex. This implies that common off-the-shelf optimization algorithms can solve it efficiently.

We have built a MATLAB program to solve Problem 2, which is available on GitHub [34]. Due to the non-linearities in the problem, i.e., some of our constraints are in terms of the second largest eigenvalue of one of our decision variables, we found MATLAB and `fmincon` to perform well for problems of this kind. When optimizing over $L(\mathcal{G}) \in \mathcal{L}(L_0)$, since we only consider undirected, symmetrically weighted graphs, we need only find the upper triangular elements of $L(\mathcal{G})$, which helps reduce computation time.

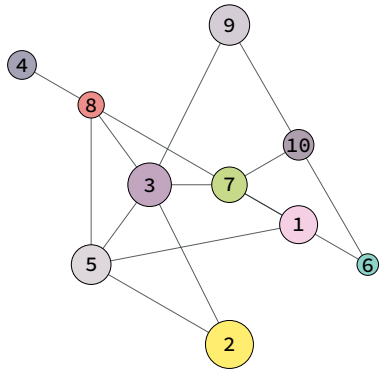


Fig. 1: The fixed input topology used for the simulation results presented in Section VIII. This topology contains 10 nodes and specifies the edges that must be present, though the edge weights are not specified and will be optimized over.

Agent \ Parameter	η_i	τ_i	ρ_i	ϵ_i^{max}
1	0.062	0.041	14.79	0.681
2	0.148	0.008	15.27	0.862
3	0.157	0.345	15.09	0.782
4	0.735	0.176	15.16	0.516
5	0.323	0.022	14.84	0.713
6	0.452	0.174	14.75	0.387
7	0.403	0.287	14.84	0.637
8	0.066	0.272	14.95	0.472
9	0.333	0.213	14.78	0.725
10	0.129	0.195	15.23	0.513

TABLE I: The agent-level parameters used in the privacy and network co-design simulation results presented in Section VIII. The parameters $\eta_i,\ \tau_i,\ \rho_i$ are used to define the constraint $\tau_i\epsilon_i + \eta_i d_i \leq \rho_i$ for all i, and ϵ_i^{max} defines the minimum level of privacy for each agent. The values $\epsilon_i^{max} \in [0.387,\ 0.862]$ correspond to each agent requiring relatively strong privacy. Agent 6 requires the highest level of privacy and agent 2 requires the weakest privacy among the agents in the network.

VIII. Simulation Results

Here we provide simulation results for optimal privacy and network co-design. We define an input topology and then fix all of the constraint parameters other than the steady state error requirement, and we vary the maximum permissible steady state error to illustrate how the output of the problem changes.

Consider a network of $N = 10$ agents. First, we fix the input topology to be the graph shown in Figure 1, which is an undirected, unweighted graph with 10 nodes. Then, we let $q_\epsilon^i = \tau_i\epsilon_i$ and $q_d^i = \eta_i d_i$ for all i, where $\eta_i,\ \rho_i \in \mathbb{R}$ are weighting factors. The constraint $q_d^i(d_i) + q_\epsilon^i(\epsilon_i) \leq \rho_i$ takes the form $\tau_i\epsilon_i + \eta_i d_i \leq \rho_i$. Fix $\lambda_{2_L} = 0.3$, $\gamma = \frac{1}{20}$, $b_i = 1$ for all i, and $\delta_i = 0.05$ for all i. The values of $\eta_i,\ \tau_i,\ \rho_i$, and ϵ_i^{max} are specified in Table I.

Figure 2 shows the output topologies for $e_R \in \{50,\ 100,\ 150\}$. In the figure it can be seen that as e_R increases, the weights in the network become smaller, depicted in the figure by the labeled weights and the decreased thickness of the edges. This shows that as the performance requirement is relaxed, the resulting optimal network is less connected. As the edge weights change with varying e_R, we can see that the optimization problem is changing some weights more than others, which is primarily controlled by the constraint $q_d^i(d_i) + q_\epsilon^i(\epsilon_i) \leq \rho_i$. For example, in Figure 2 the weight of the edge connecting agents 5 and 8 does not change much with e_R much, but the other edge weights have larger variation with e_R. Figures 1 & 2 were generated using [35].

IX. Conclusions

In this paper we have studied the problem of differentially private formation control. This work enables agents to collaboratively assemble into formations with bounded steady state error and provides methods for solving for the error covariance matrix at steady-state. This work also develops and solves an optimization problem to design the optimal network and privacy parameters for differentially private formation control. Future work includes generalizing to other privacy/performance co-design problems and implementation on mobile robots.

References

[1] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[2] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[3] S. P. Kasiviswanathan and A. Smith, "On the'semantics' of differential privacy: A bayesian formulation," *Journal of Privacy and Confidentiality*, vol. 6, no. 1, 2014.

[4] J. Le Ny and G. J. Pappas, "Differentially private filtering," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 341–354, 2013.

[5] K. Yazdani, A. Jones, K. Leahy, and M. Hale, "Differentially private lq control," *arXiv preprint arXiv:1807.05082*, 2018.

[6] M. T. Hale and M. Egerstedt, "Cloud-enabled differentially private multiagent optimization with constraints," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1693–1706, 2017.

[7] J. Le Ny and M. Mohammady, "Differentially private mimo filtering for event streams," *IEEE Transactions on Automatic Control*, vol. 63, no. 1, pp. 145–157, 2017.

[8] A. Jones, K. Leahy, and M. Hale, "Towards differential privacy for symbolic systems," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 372–377.

[9] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society*, ser. WPES '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 81–90. [Online]. Available: https://doi.org/10.1145/2381966.2381978

[10] Y. Wang, Z. Huang, S. Mitra, and G. E. Dullerud, "Differential privacy in linear distributed control systems: Entropy minimizing mechanisms and performance tradeoffs," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 118–130, 2017.

[11] Z. Xu, K. Yazdani, M. T. Hale, and U. Topcu, "Differentially private controller synthesis with metric temporal logic specifications," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 4745–4750.

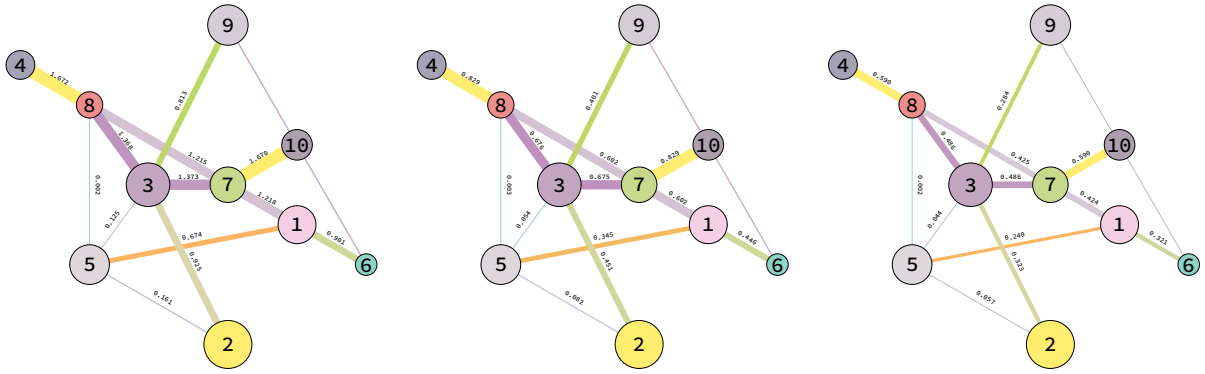
[12] Y. Wang, M. Hale, M. Egerstedt, and G. E. Dullerud, "Differentially private objective functions in distributed cloud-based optimization," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 3688–3694.

[13] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[14] A. Nedić, A. Olshevsky, and W. Shi, *Decentralized Consensus Optimization and Resource Allocation*, 2018, pp. 247–287.

[15] N. Hashemi and J. Ruths, "Co-design for security and performance: Geometric tools," *arXiv e-prints*, pp. arXiv–2006, 2020.

[16] C. Hawkins and M. Hale, "Differentially private formation control," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 6260–6265.



(a) $e_{ss} \leq 50$ (b) $e_{ss} \leq 100$ (c) $e_{ss} \leq 150$

Fig. 2: The output of the optimal privacy and network co-design solver with a fixed input topology defined in Figure 1, agent level parameters defined by Table I, and with the remaining parameters defined in Section VIII. For these results, we hold everything but the e_{ss} constraint fixed and then run the solver for $e_R \in \{50, 100, 150\}$. In the graphs above, the edges are drawn with thickness proportional to their weight, the edges are labeled with their weight, and the smaller a node is drawn, the more private it is; that is a smaller node corresponds to a smaller ϵ_i. As we increase e_R, we are allowing weaker performance for the resulting network. The sample outputs here illustrate that as we loosen the network level performance constraint, much less edge weight is used throughout the network and thus the network becomes less connected. This shows that the privacy and network co-design problem successfully encodes the trade-offs between privacy, performance, and communication topology given design requirements.

[17] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973.

[18] C. Dwork, "Differential privacy," *Automata, languages and programming*, pp. 1–12, 2006.

[19] K. Yazdani and M. Hale, "Error bounds and guidelines for privacy calibration in differentially private kalman filtering," in *2020 American Control Conference (ACC)*, 2020, pp. 4423–4428.

[20] J. Le Ny, *Differential Privacy for Dynamic Data*. Springer, 2020.

[21] B. Hajek, *Random processes for engineers*. Cambridge university press, 2015.

[22] A. Jadbabaie and A. Olshevsky, "Scaling laws for consensus protocols subject to noise," 2015.

[23] L. Krick, M. E. Broucke, and B. A. Francis, "Stabilisation of infinitesimally rigid formations of multi-robot networks," *International Journal of control*, vol. 82, no. 3, pp. 423–439, 2009.

[24] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control systems magazine*, vol. 27, no. 2, pp. 71–82, 2007.

[25] W. Ren, "Consensus strategies for cooperative control of vehicle formations," *IET Control Theory & Applications*, vol. 1, no. 2, pp. 505–512, 2007.

[26] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE transactions on automatic control*, vol. 49, no. 9, pp. 1465–1476, 2004.

[27] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[28] C. Hawkins and M. Hale, "Differentially private formation control," *arXiv preprint arXiv:2004.02744*, 2020.

[29] M. Pinsky and S. Karlin, *An introduction to stochastic modeling*. Academic press, 2010.

[30] K. Schacke, "On the kronecker product," *Master's thesis, University of Waterloo*, 2004.

[31] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM review*, vol. 50, no. 1, pp. 37–66, 2008.

[32] T. Summers, I. Shames, J. Lygeros, and F. Dörfler, "Topology design for optimal network coherence," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 575–580.

[33] S. Y. Shafi, M. Arcak, and L. El Ghaoui, "Designing node and edge weights of a graph to meet laplacian eigenvalue constraints," in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2010, pp. 1016–1023.

[34] C. Hawkins, "Differentially-private-formation-control-privacy-network-co-design," https://github.com/fx0641/Differentially-Private-Formation-Control-Privacy-Network-Co-Design, 2021.

[35] T. P. Peixoto, "The graph-tool python library," 2014.

[36] A. Ghosh and S. Boyd, "Upper bounds on algebraic connectivity via convex optimization," *Linear algebra and its applications*, vol. 418, no. 2-3, pp. 693–707, 2006.

[37] A. Agrawal and S. Boyd, "Disciplined quasiconvex programming," *Optimization Letters*, vol. 14, no. 7, pp. 1643–1657, 2020.

Calvin Hawkins is a PhD student at the University of Florida and is a recipient of the Graduate School Preeminence Award. He received his Bachelor's degree in Mechanical Engineering *summa cum laude* from Wayne State University in May, 2019. His current research interests are broadly in the area of privacy in control, with an emphasis on bringing differential privacy to new classes of data typically used by control systems and quantifying the effects of privacy upon feedback.

Matthew Hale is an Assistant Professor of Mechanical and Aerospace Engineering at the University of Florida. He received his BSE in Electrical Engineering *summa cum laude* from the University of Pennsylvania in 2012, and his MS and PhD in Electrical and Computer Engineering from the Georgia Institute of Technology in 2015 and 2017, respectively. His research interests include multi-agent systems, mobile robotics, privacy in control, and distributed optimization. He was the Teacher of the Year in the UF Department of Mechanical and Aerospace Engineering for the 2018-2019 school year, and he received an NSF CAREER Award in 2020 for his work on privacy in control systems.

APPENDIX

A. Proof of Lemma 3

We have the node level protocol

$$\bar{x}_i(k+1) = \bar{x}_i(k)+\gamma \sum_{j\in N(i)} w_{ij}(\bar{x}_j(k)+v_j(k)-\bar{x}_i(k)-v_i(k)),$$

which we factor as

$$\begin{aligned}\bar{x}_i(k+1) = \bar{x}_i(k) &+ \gamma \sum_{j\in N(i)} w_{ij}(\bar{x}_j(k) - \bar{x}_i(k)) \\ &+\gamma \sum_{j\in N(i)} w_{ij}(v_j(k) - v_i(k)).\end{aligned}$$

Let $z_i(k) = \gamma \sum_{j\in N(i)} w_{ij}(v_j(k) - v_i(k))$, and note that

$$z_i(k) = \gamma[A(\mathcal{G}) \otimes I_d]_{row\ i} v(k) - \gamma[D(\mathcal{G}) \otimes I_d]_{row\ i} v(k).$$

Hence if we let $z(k) = [z_1(k)^T, \dots, z_N(k)^T]^T$ we have $z(k) = -\gamma(L(\mathcal{G}) \otimes I_d)v(k)$, since $L(\mathcal{G}) = D(\mathcal{G}) - A(\mathcal{G})$. Since $v(k)$ and $z(k)$ are zero mean, by definition $\Sigma_v = E[v(k)v(k)^T]$, and we have

$$\begin{aligned}\Sigma_z =&E[z(k)z(k)^T] \\ =&E[\gamma(L(\mathcal{G}) \otimes I_d)v(k)(\gamma(L(\mathcal{G}) \otimes I_d)v(k))^T],\end{aligned}$$

then since $L(\mathcal{G})$ is symmetric, the linearity of expectation gives

$$\begin{aligned}\Sigma_z &= \gamma^2(L(\mathcal{G}) \otimes I_d)E[v(k)v(k)^T](L(\mathcal{G}) \otimes I_d) \\ &= \gamma^2(L(\mathcal{G}) \otimes I_d)\Sigma_v(L(\mathcal{G}) \otimes I_d).\end{aligned}$$

Overall, this gives $z(k) \sim \mathcal{N}(0, \gamma^2(L(\mathcal{G})\otimes I_d)\Sigma_v(L(\mathcal{G})\otimes I_d))$.

B. Proof of Lemma 7

First, recall our definition of network level steady-state error in (7): we let $e_d = \limsup_{k\to\infty} e_{\text{agg}}(k)$ and then set $e_{ss} = e_d d$. From [22], (17) states that

$$\begin{aligned}e_d = &\frac{1}{N^3}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} H_{P_d^2}(k \to l)s_{li} \\ &- \frac{1}{N^2}\sum_{i<j} s_{ij}\left(H_{P_d^2}(i \to j) + H_{P_d^2}(j \to i)\right).\end{aligned}$$

By dropping the negative term and taking the max over s_{li}, it follows that

$$\begin{aligned}e_d &\le \frac{1}{N^3}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} H_{P_d^2}(k \to l)s_{li} \\ &\le \frac{\max_{i,j} s_{ij}}{N^3}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} H_{P_d^2}(k \to l).\end{aligned}$$

The Markov chain evolving according to P_d has stationary distribution $\pi = \frac{1}{N}\mathbb{1}$, and thus

$$\begin{aligned}&\frac{\max_{i,j} s_{ij}}{N^3}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} H_{P_d^2}(k \to l) \\ &\quad= \frac{(\max_{i,j} s_{ij})}{N}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} \frac{1}{N^2} H_{P_d^2}(k \to l) \\ &\quad= \frac{(\max_{i,j} s_{ij})}{N}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} \pi_k\pi_l H_{P_d^2}(k \to l).\end{aligned}$$

By definition, $K(P_d^2) = \sum_{k=1}^{N}\sum_{l=1}^{N} \pi_k\pi_l H_{P_d^2}(k \to l)$, thus

$$\begin{aligned}&\frac{\max_{i,j} s_{ij}}{N^3}\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} H_{P_d^2}(k \to l) \\ &\quad= \frac{(\max_{i,j} s_{ij})}{N}\sum_{i=1}^{N} K(P_d^2) = (\max_{i,j} s_{ij})K(P_d^2)\end{aligned}$$

giving $e_d \le (\max_{i,j} s_{ij})K(P_d^2)$, and finally

$$e_{ss} \le (\max_{i,j} s_{ij})K(P_d^2)d.$$

C. Proof of Theorem 1

With Lemma 7, $e_{\text{ss}} \le (\max_{i,j} s_{ij})K(P_d^2)d$. Then using Lemma 8 to upper bound $K(P_d^2)$ gives

$$e_{ss} \le \frac{(\max_{i,j} s_{ij})\,(N-1)d}{\gamma\lambda_2(L(\mathcal{G}))(2-\gamma\lambda_2(L(\mathcal{G})))}. \tag{12}$$

Now consider s_{ij}, which is the $i^{th}j^{th}$ entry of $\Sigma_z = \gamma^2 A(\mathcal{G})\Sigma_v A(\mathcal{G})$ from Lemma 3. It can be shown that the diagonal terms of this product are given by $s_{ii} = \gamma^2\sum_{j\in N(i)} w_{ij}^2\sigma_j^2$, and the off-diagonal terms are given as $s_{ij} = \gamma^2\sum_{l\in N(i), l\neq j} w_{il}w_{jl}\sigma_l^2$. Then, because $w_{ij} \in (0,1)$,

$$\max_{i,j} s_{ij} \le \gamma^2 \max_i \left(|N(i)| \max_{j\in N(i)} \sigma_j^2\right).$$

For N agents $|N(i)| \le N-1$, $\sigma_i^2 \ge \kappa(\delta_i,\epsilon_i)^2 b_i^2$, and $\max_i[\max_{j\in N(i)} \sigma_j^2] = \max_i \sigma_i^2$, which gives

$$\gamma^2 \max_i \left[|N(i)| \max_{j\in N(i)} \sigma_j^2\right] \le \gamma^2(N-1)\max_i \kappa(\delta_i,\epsilon_i)^2 b_i^2. \tag{13}$$

Plugging (13) into (12) completes the proof.

D. Proof of Theorem 2

First we establish a sensitvity bound for two adjacent collections of trajectories, ξ and ξ'. We consider

$$\begin{aligned}&\|q_\xi(N(i)) - q_{\xi'}(N(i))\|_{\ell_2} \\ &\qquad= \left(\sum_{k=1}^{\infty}\left[\sum_{j\in N(i)} w_{ij}\big(\xi_j(k) - \xi_j'(k)\big)\right]^2\right)^{\frac{1}{2}}.\end{aligned}$$

Using the adjacency relationship, suppose that ξ and ξ' differ in entry l. Then

$$\|q_\xi(N(i)) - q_{\xi'}(N(i))\|_{\ell_2} = \left(\sum_{k=1}^{\infty}\left[w_{il}\big(\xi_l(k) - \xi_l'(k)\big)\right]^2\right)^{\frac{1}{2}}.$$

By factoring out w_{ij} and then using $w_{ij} \le \bar{w}$ and upper bounding the remaining terms in the sum with the adjacency relationship, we arrive at the sensitivity bound $\|q_\xi(N(i) - q_{\xi'}(N(i))\|_{\ell_2} \le \bar{w}b$. Then using Lemma 1, the Gaussian mechanism is differentially private for $\sigma \ge \bar{w}b\kappa(\delta,\epsilon)$.

E. Proof of Theorem 4

From Lemma 9, and with $Q_{[l]}$ defined above, we have

$$\Sigma_{ss_{[l]}} = \sum_{i=0}^{\infty}(P_d - J)^i Q_{[l]}(P_d - J)^i,$$

and taking the first term out of the sum gives

$$\Sigma_{ss_{[l]}} = Q_{[l]} + \sum_{i=1}^{\infty}(P_d - J)^i Q_{[l]}(P_d - J)^i.$$

Factoring out $(P_d - J)$ on both sides of the sum gives

$$\Sigma_{ss_{[l]}} = Q_{[l]} + (P_d - J)\left(\sum_{i=1}^{\infty}(P_d - J)^{i-1} Q_{[l]}(P_d - J)^{i-1}\right)(P_d - J).$$

The remaining infinite sum is precisely $\Sigma_{ss_{[l]}}$. Thus, we arrive at the equation $\Sigma_{ss_{[l]}} = Q_{[l]} + (P_d - J)\Sigma_{ss_{[l]}}(P_d - J)$, which is the discrete-time Lyapunov equation. One analytical solution to it can be found using the vectorization operator [30], and it takes the form

$$\text{vec}(\Sigma_{ss_{[l]}}) = (I_{N^2} - (P_d - J) \otimes (P_d - J))^{-1}\text{vec}(Q_{[l]}).$$

For convenience let $R = I_{N^2} - (P_d - J) \otimes (P_d - J)$. R is invertible because $(P_d - J)$ has eigenvalues strictly less than 1 from Lemma 9. Then simplifying further by plugging in $Q_{[l]} = (I - J)\Sigma_{z_{[l]}}(I - J)$ and using $\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$, we have

$$\text{vec}(\Sigma_{ss_{[l]}}) = R^{-1}\left((I_N - J) \otimes (I_N - J)\right)\text{vec}\left(\Sigma_{z_l}\right).$$

Then, plugging in $\Sigma_{z_{[l]}} = \gamma^2 L(\mathcal{G})\Sigma_{v_{[l]}}L(\mathcal{G})^T$ gives us

$$\text{vec}(\Sigma_{ss_{[l]}}) = \gamma^2 R^{-1} M \text{vec}\left(\Sigma_{v_{[l]}}\right).$$

F. Proof of Theorem 5

We begin by analyzing the objective function $\vartheta\text{Tr}(L(\mathcal{G})) + \sum_{i\in[N]}\frac{1}{\epsilon_i^2}$. The trace is a linear operator over the set of $N \times N$ matrices and hence convex, and the function $\frac{1}{\epsilon_i^2}$ is easily shown to be convex for all i. The objective function is thus convex by virtue of being a sum of convex functions.

The constraint $q_d(d_i) + q_\epsilon(\epsilon_i) \leq \rho_i$ is convex since q_d^i and q_ϵ^i are convex for all i. The constraint $\epsilon_i \leq \epsilon_i^{max}$ is also trivially convex. Lastly, we consider the constraint on λ_2. We note that, for a graph Laplacian L, $\lambda_2(L) = \inf\{x^T Lx : \|x\|_2 = 1, 1^T x = 0\}$ [36], which is a concave function because it is the infimum over a set of linear maps. The constraint above contains $-\lambda_2(L(\mathcal{G}))$, which is a convex function, upper bounded by a constant, which gives a convex constraint.

To analyze the constraint $e_{ss} \leq e_R$, we use the following Lemma.

Lemma 10 ([37]): Suppose h is a quasiconvex mapping of a subset C of $\mathbb{R}^k$ into $\mathbb{R} \cup \infty$, and $\{I_1, I_2, I_3\}$ is a partition of $\{1, 2, \ldots, k\}$ such that h is nondecreasing in the arguments indexed by I_1 and nonincreasing in the arguments indexed by I_2, and g maps a subset of $\mathbb{R}^n$ into $\mathbb{R}^k$ in such a way that its components g_i are convex for $i \in I_1$, concave for $i \in I_2$, and affine for $i \in I_3$. Then the composition

$$f = h \circ g$$

is quasiconvex. If additionally h is convex, then f is convex as well. ■

We will now show that $e_{ss} \leq e_R$ is a convex constraint. As written, e_{ss} is a mapping from $L \in \mathcal{L}(L_0)$ and $\{\epsilon_i\}_{i\in[N]} \in \mathbb{R}_+^N$ to $\mathbb{R}_+$. However, for the purpose of analysis, we will consider e_{ss} as a function of $\lambda_2(L) \in [0, N]$ and $\{\epsilon_i\}_{i\in[N]} \in \mathbb{R}_+^N$, where λ_2 is a function of L. Then, use the composition of e_{ss} and $\lambda_2(L)$ to analyze e_{ss} as a function of $L \in \mathcal{L}$ and $\{\epsilon_i\}_{i\in[N]} \in \mathbb{R}_+^N$.

First we rewrite the constraint $e_{ss} \leq e_R$ as

$$\frac{\gamma N d(N-1)^2}{e_R}\max_i \kappa(\delta_i, \epsilon_i)^2 b_i^2 \leq N\lambda_2(L(\mathcal{G}))(2 - \gamma\lambda_2(L(\mathcal{G}))).$$

To simplify the left hand side, group the constants as $c = \frac{\gamma N d(N-1)^2}{e_R}$ which is independent of the topology and privacy, and let $g_{1_i}(\epsilon_i) = \kappa(\delta, \epsilon_i)^2 b_i^2$. Then for the right hand side we treat $z = \lambda_2(L(\mathcal{G}))$ as a scalar giving $g_2(z) = Nz(2 - \gamma z)$, and $h(L) = \lambda_2(L)$ is the mapping from L to $\lambda_2(L)$ Then the constraint can be reduced to $c\max_i g_{1_i} \leq g_2 \circ h$

Now we analyze the convexity of g_{1_i}. It is only a function of ϵ_i, and we only need to consider the convexity over the space of privacy parameters $\epsilon_i > 0$. Differentiating g_{1_i} with respect to ϵ_i twice gives

$$\frac{\partial^2 g_{1_i}}{\partial \epsilon_i^2} = \frac{1}{2\epsilon^2(2\epsilon + K_\delta^2)} - \frac{K_\delta + \sqrt{2\epsilon + K_\delta^2}}{2\epsilon^2(2\epsilon + + K_\delta^2)^{\frac{3}{2}}}$$
$$- \frac{2(K_\delta + \sqrt{2\epsilon + K_\delta^2})}{\epsilon^3\sqrt{2\epsilon + + K_\delta^2}} + \frac{3(K_\delta + \sqrt{2\epsilon + K_\delta^2})^2}{2\epsilon^4}.$$

This expression is greater than or equal to 0 if $K_\delta \geq 0$ and $\epsilon_i > 0$. Since we only consider $\epsilon_i > 0$ and $K_\delta > 0$ because $\delta \in (0, \frac{1}{2})$, the second derivative is always greater than 0 and thus g_{1_i} is a convex function. In the constraint, we are taking the the maximum of g_{1_i} over i, which is the maximum over a collection of convex functions, which is also convex. Thus, $g_1 = \max_i g_{1_i}$ is convex.

Now we shift our attention to g_2. We have $\frac{\partial^2 g_2}{\partial z^2} = -2N\gamma$, and since $\gamma > 0$, we have $\frac{\partial^2 g_2}{\partial z^2} < 0$ and thus g_2 is concave in λ_2. Taking the first derivative with respect to z gives, $\frac{\partial g_2}{\partial z} = 2N - 2\gamma z$, simplifying $\frac{\partial g_2}{\partial z} \geq 0$ gives $\gamma z \leq N$, where this inequality always holds because we only consider connected graphs, such that $z \in [0, N]$, and we require that $\gamma \leq \frac{1}{N}$. Thus, g_2 is non-decreasing

Now we apply Lemma 10 to $g_2 \circ h$. We consider $h(L) = \lambda_2(L) = \inf\{x^T Lx : \|x\|_2 = 1, 1^T x = 0\}$, which is an infimum over a set of linear maps and is thus concave. We have shown that g_2 is concave and non-decreasing, which implies that $-g_2$ is convex and non-increasing in its argument z. This allows us to use Lemma 10 to conclude that the composition $-g_2 \circ \lambda_2$ is also convex, because $-g_2$ is non-increasing in all of its arguments and h is concave. All of this implies $c\max_i g_{1_i} - g_2 \circ h$ is a convex function and thus $c\max_i g_{1_i} - g_2 \circ h \leq 0$ is a convex constraint. Since the cost function and all other constraints are also convex, Problem 2 is a convex optimization problem.