Distributed Autonomous Systems

Last update: 20 March 2025

Academic Year 2024 – 2025
Alma Mater Studiorum · University of Bologna

Contents

1 Graphs

1.1 Definitions

Directed graph (digraph) Pair $G = (I, E)$ where $I = \{1, \dots, N\}$ is the set of nodes and $E \subseteq I \times I$ is the set of edges. Directed graph

Undirected graph Digraph where $\forall i, j : (i, j) \in E \Rightarrow (j, i) \in E$. Undirected graph

Subgraph Given a graph (I, E), (I', E') is a subgraph of it if $I' \subseteq I$ and $E' \subset E$. Subgraph

Spanning subgraph Subgraph where $I' = I$.

In-neighbor A node $j \in I$ is an in-neighbor of $i \in I$ if $(j, i) \in E$. In-neighbor

Set of in-neighbors The set of in-neighbors of $i \in I$ is the set: Set of in-neighbors

$$\mathcal{N}_i^{\text{IN}} = \{j \in I \mid (j, i) \in E\}$$

In-degree Number of in-neighbors of a node $i \in I$: In-degree

$$\deg_i^{\text{IN}} = |\mathcal{N}_i^{\text{IN}}|$$

Out-neighbor A node $j \in I$ is an out-neighbor of $i \in I$ if $(i, j) \in E$. Out-neighbor

Set of out-neighbors The set of out-neighbors of $i \in I$ is the set: Set of in-neighbors

$$\mathcal{N}_i^{\text{OUT}} = \{j \in I \mid (i, j) \in E\}$$

Out-degree Number of out-neighbors of a node $i \in I$: Out-degree

$$\deg_i^{\text{OUT}} = |\mathcal{N}_i^{\text{OUT}}|$$

Balanced digraph A digraph is balanced if $\forall i \in I : \deg_i^{\text{IN}} = \deg_i^{\text{OUT}}$. Balanced digraph

Periodic graph Graph where there exists a period $k > 1$ that divides the length of any cycle. Periodic graph

Remark. A graph with self-loops is aperiodic.

Strongly connected digraph Digraph where each node is reachable from any node. Strongly connected digraph

Connected undirected graph Undirected graph where each node is reachable from any node. Connected undirected graph

Weakly connected digraph Digraph where its undirected version is connected. Weakly connected digraph

1.2 Weighted digraphs

Weighted digraph Triplet $G = (I, E, \{a_{i,j}\}_{(i,j)\in E})$ where (I, E) is a digraph and $a_{i,j} > 0$ is a weight for the edge (i, j). Weighted digraph

Weighted in-degree Sum of the weights of the inward edges: Weighted in-degree

$$\deg_i^{\text{IN}} = \sum_{j=1}^{N} a_{j,i}$$

Weighted out-degree Sum of the weights of the outward edges: Weighted out-degree

$$\deg_i^{\text{OUT}} = \sum_{j=1}^{N} a_{i,j}$$

Weighted adjacency matrix Non-negative matrix $\boldsymbol{A}$ such that $\boldsymbol{A}_{i,j} = a_{i,j}$: Weighted adjacency matrix

$$\begin{cases} \boldsymbol{A}_{i,j} > 0 & \text{if } (i,j) \in E \\ \boldsymbol{A}_{i,j} = 0 & \text{otherwise} \end{cases}$$

In/out-degree matrix Matrix where the diagonal contains the in/out-degrees: In/out-degree matrix

$$\boldsymbol{D}^{\text{IN}} = \begin{bmatrix} \deg_1^{\text{IN}} & 0 & \cdots & 0 \\ 0 & \deg_2^{\text{IN}} & & \\ \vdots & & \ddots & \\ 0 & \cdots & 0 & \deg_N^{\text{IN}} \end{bmatrix} \qquad \boldsymbol{D}^{\text{OUT}} = \begin{bmatrix} \deg_1^{\text{OUT}} & 0 & \cdots & 0 \\ 0 & \deg_2^{\text{OUT}} & & \\ \vdots & & \ddots & \\ 0 & \cdots & 0 & \deg_N^{\text{OUT}} \end{bmatrix}$$

Remark. Given a digraph with adjacency matrix $\boldsymbol{A}$, its reverse digraph has adjacency matrix $\boldsymbol{A}^T$.

Remark. It holds that:

$$\boldsymbol{D}^{\text{IN}} = \text{diag}(\boldsymbol{A}^T\boldsymbol{1}) \quad \boldsymbol{D}^{\text{OUT}} = \text{diag}(\boldsymbol{A}\boldsymbol{1})$$

where $\boldsymbol{1}$ is a vector of ones.

Remark. A digraph is balanced iff $\boldsymbol{A}^T\boldsymbol{1} = \boldsymbol{A}\boldsymbol{1}$.

1.3 Laplacian matrix

(Out-degree) Laplacian matrix Matrix $\boldsymbol{L}$ defined as: Laplacian matrix

$$\boldsymbol{L} = \boldsymbol{D}^{\text{OUT}} - \boldsymbol{A}$$

Remark. The vector $\boldsymbol{1}$ is always an eigenvector of $\boldsymbol{L}$ with eigenvalue 0:

$$\boldsymbol{L}\boldsymbol{1} = (\boldsymbol{D}^{\text{OUT}} - \boldsymbol{A})\boldsymbol{1} = \boldsymbol{D}^{\text{OUT}}\boldsymbol{1} - \boldsymbol{D}^{\text{OUT}}\boldsymbol{1} = 0$$

In-degree Laplacian matrix Matrix $\boldsymbol{L}^{\text{IN}}$ defined as: In-degree Laplacian matrix

$$\boldsymbol{L}^{\text{IN}} = \boldsymbol{D}^{\text{IN}} - \boldsymbol{A}^T$$

Remark. $\boldsymbol{L}^{\mathrm{IN}}$ is the out-degree Laplacian of the reverse graph.

2 Averaging systems

Distributed algorithm Given a network of N agents that communicate according to a (fixed) digraph G (each agent receives messages from its in-neighbors), a distributed algorithm computes:

Distributed algorithm

$$x_i^{k+1} = \texttt{stf}_i(x_i^k, \{x_j^k\}_{j \in \mathcal{N}_i^{\text{IN}}}) \quad \forall i \in \{1, \dots, N\}$$

where x_i^k is the state of agent i at time k and $\texttt{stf}_i$ is a local state transition function that depends on the current input states.

Remark. Out-neighbors can also be used.

Remark. If all nodes have a self-loop, the notation can be compacted as:

$$x_i^{k+1} = \texttt{stf}_i(\{x_j\}_{j \in \mathcal{N}_i^{\text{IN}}}) \quad \text{or} \quad x_i^{k+1} = \texttt{stf}_i(\{x_j\}_{j \in \mathcal{N}_i^{\text{OUT}}})$$

2.1 Discrete-time averaging algorithm

Linear averaging distributed algorithm (in-neighbors) Given the communication digraph with self-loops $G^{\text{comm}} = (I, E)$ (i.e., $(j, i) \in E$ indicates that j sends messages to i), a linear averaging distributed algorithm is defined as:

Linear averaging distributed algorithm (in-neighbors)

$$x_i^{k+1} = \sum_{j \in \mathcal{N}_i^{\text{IN}}} a_{ij} x_j^k \quad i \in \{1, \dots, N\}$$

where $a_{ij} > 0$ is the weight of the edge $(j, i) \in E$.

Linear time-invariant (LTI) autonomous system By defining $a_{ij} = 0$ for $(j, i) \notin E$, the formulation becomes:

Linear time-invariant (LTI) autonomous system

$$x_i^{k+1} = \sum_{j=1}^{N} a_{ij} x_j^k \quad i \in \{1, \dots, N\}$$

In matrix form, it becomes:

$$\mathbf{x}^{k+1} = \boldsymbol{A}^T \mathbf{x}^k$$

where $\boldsymbol{A}$ is the adjacency matrix of G^{comm}.

Remark. This model is inconsistent with respect to graph theory as weights are inverted (i.e., a_{ij} refers to the edge (j, i)).

Linear averaging distributed algorithm (out-neighbors) Given a fixed sensing digraph with self-loops $G^{\text{sens}} = (I, E)$ (i.e., $(i, j) \in E$ indicates that j sends messages to i), the algorithm is defined as:

Linear averaging distributed algorithm (out-neighbors)

$$x_i^{k+1} = \sum_{j \in \mathcal{N}_i^{\text{OUT}}} a_{ij} x_j^k = \sum_{j=1}^{N} a_{ij} x_j^k$$

In matrix form, it becomes:

$$\mathbf{x}^{k+1} = \boldsymbol{A}\mathbf{x}^k$$

where $\boldsymbol{A}$ is the weighted adjacency matrix of G^{sens}.

2.1.1 Stochastic matrices

Row stochastic Given a square matrix $\boldsymbol{A}$, it is row stochastic if its rows sum to 1: Row stochastic

$$\boldsymbol{A}\mathbf{1} = \mathbf{1}$$

Column stochastic Given a square matrix $\boldsymbol{A}$, it is column stochastic if its columns sum to 1: Column stochastic

$$\boldsymbol{A}^T\mathbf{1} = \mathbf{1}$$

Doubly stochastic Given a square matrix $\boldsymbol{A}$, it is doubly stochastic if it is both row and column stochastic. Doubly stochastic

Lemma 2.1.1. An adjacency matrix $\boldsymbol{A}$ is doubly stochastic if it is row stochastic and the graph G associated to it is weight balanced and has positive weights.

Lemma 2.1.2. Given a digraph G with adjacency matrix $\boldsymbol{A}$, if G is strongly connected and aperiodic, and $\boldsymbol{A}$ is row stochastic, its eigenvalues are such that:

- $\lambda = 1$ is a simple eigenvalue (i.e., algebraic multiplicity of 1),
- All others μ are $|\mu| < 1$.

Remark. For the lemma to hold, it is necessary and sufficient that G contains a globally reachable node and the subgraph of globally reachable nodes is aperiodic.

2.1.2 Consensus

Theorem 2.1.1 (Discrete-time consensus)**.** Consider a discrete-time averaging system with digraph G and weighted adjacency matrix $\boldsymbol{A}$. Assume G strongly connected and aperiodic, and $\boldsymbol{A}$ row stochastic. Discrete-time consensus

It holds that there exists a left eigenvector $\mathbf{w} \in \mathbb{R}^N$, $\mathbf{w} > 0$ such that the consensus converges to:

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{1}\frac{\mathbf{w}^T\mathbf{x}^0}{\mathbf{w}^T\mathbf{1}} = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix} \frac{\sum_{i=1}^N w_i x_i^0}{\sum_{j=1}^N w_j} = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix} \sum_{i=1}^N \frac{w_i}{\sum_{j=1}^N w_j} x_i^0$$

where $\tilde{w}_i = \frac{w_i}{\sum_{i=j}^N w_j}$ are all normalized and sum to 1 (i.e., they produce a convex combination).

Moreover, if $\boldsymbol{A}$ is doubly stochastic, then it holds that the consensus is the average:

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{1}\frac{1}{N}\sum_{i=1}^N x_i^0$$

Sketch of proof. Let $\boldsymbol{T} = \begin{bmatrix}\mathbf{1} & \mathbf{v}^2 & \cdots & \mathbf{v}^N\end{bmatrix}$ be a change in coordinates that transforms an adjacency matrix into its Jordan form $\boldsymbol{J}$:

$$\boldsymbol{J} = \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}$$

As $\lambda = 1$ is a simple eigenvalue (Lemma 2.1.2), it holds that:

$$\boldsymbol{J} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & \boldsymbol{J}_2 & \\ 0 & & & \end{bmatrix}$$

where the eigenvalues of $\boldsymbol{J}_2 \in \mathbb{R}^{(N-1)\times(N-1)}$ lie inside the open unit disk.
Let $\mathbf{x}^k = \boldsymbol{T}\bar{\mathbf{x}}^k$, then we have that:

$$\begin{aligned} &\mathbf{x}^{k+1} = \boldsymbol{A}\mathbf{x}^k \\ \iff\ &\boldsymbol{T}\bar{\mathbf{x}}^{k+1} = \boldsymbol{A}(\boldsymbol{T}\bar{\mathbf{x}}^k) \\ \iff\ &\bar{\mathbf{x}}^{k+1} = \boldsymbol{T}^{-1}\boldsymbol{A}(\boldsymbol{T}\bar{\mathbf{x}}^k) = \boldsymbol{J}\bar{\mathbf{x}}^k \end{aligned}$$

Therefore:

$$\lim_{k\to\infty} \bar{\mathbf{x}}^k = \bar{x}_1^0 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\bar{x}_1^{k+1} = \bar{x}_1^k \quad \forall k \geq 0$$

$$\lim_{k\to\infty} \bar{x}_i^k = 0 \quad \forall i = 2, \dots, N$$

□

Example (Metropolis-Hasting weights)**.** Given an undirected unweighted graph G with edges of degrees $d_1, \dots, d_n$, Metropolis-Hasting weights are defined as:

$$a_{ij} = \begin{cases} \frac{1}{1+\max\{d_i,d_j\}} & \text{if } (i,j) \in E \text{ and } i \neq j \\ 1 - \sum_{h\in\mathcal{N}_i\smallsetminus\{i\}} a_{ih} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

The matrix $\boldsymbol{A}$ of Metropolis-Hasting weights is symmetric and doubly stochastic.

2.2 Discrete-time averaging algorithm over time-varying graphs

2.2.1 Time-varying digraphs

Time-varying digraph Graph $G = (I, E(k))$ that changes at each iteration k. It can be described by a sequence $\{G(k)\}_{k\geq 0}$. Time-varying digraph

Jointly strongly connected digraph Time-varying digraph that is asymptotically strongly connected: Jointly strongly connected digraph

$$\forall k \geq 0 : \bigcup_{\tau=k}^{+\infty} G(\tau) \text{ is strongly connected}$$

Uniformly jointly strongly/B-strongly connected digraph Time-varying digraph that is strongly connected in B steps: Uniformly jointly strongly/B-strongly connected digraph

$$\forall k \geq 0, \exists B \in \mathbb{N} : \bigcup_{\tau=k}^{k+B} G(\tau) \text{ is strongly connected}$$

Remark. (Uniformly) jointly strongly connected digraph can be disconnected at some time steps k.

Averaging distributed algorithm Given a time-varying digraph $\{G(k)\}_{k\geq 0}$ (always with self-loops), in- and out-neighbors distributed algorithms can be formulated as:

Averaging distributed algorithm over time-varying digraph

$$x_i^{k+1} = \sum_{j \in \mathcal{N}_i^{\text{IN}}(k)} a_{ij}(k) x_j^k \quad x_i^{k+1} = \sum_{j \in \mathcal{N}_i^{\text{OUT}}(k)} a_{ij}(k) x_j^k$$

Linear time-varying (LTV) discrete-time system In matrix form, it can be formulated as:

Linear time-varying (LTV) discrete-time system

$$\mathbf{x}^{k+1} = \boldsymbol{A}(k)\mathbf{x}^k$$

2.2.2 Consensus

Theorem 2.2.1 (Discrete-time consensus over time-varying graphs)**.** Consider a time-varying discrete-time average system with digraphs $\{G(k)\}_{k\geq 0}$ (all with self-loops) and weighted adjacency matrices $\{\boldsymbol{A}(k)\}_{k\geq 0}$. Assume:

Discrete-time consensus over time-varying graphs

- Each non-zero edge weight $a_{ij}(k)$, self-loops included, are larger than a constant $\varepsilon > 0$,
- There exists $B \in \mathbb{N}$ such that $\{G(k)\}_{k\geq 0}$ is B-strongly connected.

It holds that there exists a vector $\mathbf{w} \in \mathbb{R}^N$, $\mathbf{w} > 0$ such that the consensus converges to:

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{1}\frac{\mathbf{w}^T\mathbf{x}^0}{\mathbf{w}^T\mathbf{1}}$$

Moreover, if each $\boldsymbol{A}(k)$ is doubly stochastic, it holds that the consensus is the average:

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{1}\frac{1}{N}\sum_{i=1}^{N} x_i^0$$

2.3 Continuous-time averaging algorithm

2.3.1 Laplacian dynamics

Network of dynamic systems Network described by the ODEs:

Network of dynamic systems

$$\dot{x}_i(t) = u_i(t) \quad \forall i \in \{1, \dots, N\}$$

with states $x_i \in \mathbb{R}$, inputs $u_i \in \mathbb{R}$, and communication following a digraph G.

Laplacian dynamics system Consider a network of dynamic systems where u_i is defined as a proportional controller (i.e., only communicating (i, j) have a non-zero weight a_{ij}):

Laplacian dynamics system

$$\begin{aligned} u_i(t) &= -\sum_{j \in \mathcal{N}_i^{\text{OUT}}} a_{ij}\Big(x_i(t) - x_j(t)\Big) \\ &= -\sum_{j=1}^{N} a_{ij}\Big(x_i(t) - x_j(t)\Big) \end{aligned}$$

Remark. With this formulation, consensus can be seen as the problem of minimizing the error defined as the difference between the states of two nodes.

Remark. A definition with in-neighbors also exists.

Theorem 2.3.1 (Linear time invariant (LTI) continuous-time system)**.** With $\mathbf{x} = \begin{bmatrix} x_1 & \dots & x_N \end{bmatrix}^T$, the system can be written in matrix form as:

Linear time invariant (LTI) continuous-time system

$$\dot{\mathbf{x}}(t) = -\boldsymbol{L}\mathbf{x}(t)$$

where $\boldsymbol{L}$ is the Laplacian associated with the communication digraph G.

Proof. The system is defined as:

$$\dot{x}_i(t) = -\sum_{j=1}^{N} a_{ij}\Big(x_i(t) - x_j(t)\Big)$$

By rearranging, we have that:

$$\begin{aligned}\dot{x}_i(t) &= -\left(\sum_{j=1}^{N} a_{ij}\right) x_i(t) + \sum_{j=1}^{N} a_{ij} x_j(t) \\ &= -\deg_i^{\text{OUT}} x_i(t) + (\boldsymbol{A}\mathbf{x}(t))_i\end{aligned}$$

Which in matrix form is:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= -\boldsymbol{D}^{\text{OUT}}\mathbf{x}(t) + \boldsymbol{A}\mathbf{x}(t) \\ &= -(\boldsymbol{D}^{\text{OUT}} - \boldsymbol{A})\mathbf{x}(t)\end{aligned}$$

By definition, $\boldsymbol{L} = \boldsymbol{D}^{\text{OUT}} - \boldsymbol{A}$. Therefore, we have that:

$$\dot{\mathbf{x}}(t) = -\boldsymbol{L}\mathbf{x}(t)$$

□

Remark. By Theorem 2.3.1, row/column stochasticity is not required for consensus. Instead, the requirement is for the matrix to be Laplacian.

2.3.2 Consensus

Lemma 2.3.1. It holds that:

$$\boldsymbol{L}\mathbf{1} = \boldsymbol{D}^{\text{OUT}}\mathbf{1} - \boldsymbol{A}\mathbf{1} = \begin{bmatrix} \deg_1^{\text{OUT}} \\ \vdots \\ \deg_i^{\text{OUT}} \end{bmatrix} - \begin{bmatrix} \deg_1^{\text{OUT}} \\ \vdots \\ \deg_i^{\text{OUT}} \end{bmatrix} = 0$$

Lemma 2.3.2. The Laplacian $\boldsymbol{L}$ of a weighted digraph has an eigenvalue $\lambda = 0$ and all the others have strictly positive real part.

Lemma 2.3.3. Given a weighted digraph G with Laplacian $\boldsymbol{L}$, the following are equivalent:

- G is weight balanced.
- $\mathbf{1}$ is a left eigenvector of $\boldsymbol{L}$: $\mathbf{1}^T\boldsymbol{L} = 0$ with eigenvalue 0.

Lemma 2.3.4. If a weighted digraph G is strongly connected, then $\lambda = 0$ is a simple eigenvalue.

Theorem 2.3.2 (Continuous-time consensus)**.** Consider a continuous-time average system with a strongly connected weighted digraph G and Laplacian $\boldsymbol{L}$. Assume that the system follows the Laplacian dynamics $\dot{\mathbf{x}}(t) = -\boldsymbol{L}\mathbf{x}(t)$ for $t \geq 0$.

Continuous-time consensus

It holds that there exists a left eigenvector $\mathbf{w}$ of $\boldsymbol{L}$ with eigenvalue $\lambda = 0$ such that the consensus converges to:

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{1}\left(\frac{\mathbf{w}^T\mathbf{x}(0)}{\mathbf{w}^T\mathbf{1}}\right)$$

Moreover, if G is weight balanced, then it holds that the consensus is the average:

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{1}\frac{\sum_{i=1}^{N} x_i(0)}{N}$$

Remark. The result also holds for unweighted digraphs as $\mathbf{1}$ is both a left and right eigenvector of $\boldsymbol{L}$.

3 Leader-follower networks

Leader-follower network Consider agents partitioned into N_f followers and $N - N_f$ leaders.

Leader-follower network

The state vector can be partitioned as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_f \\ \mathbf{x}_l \end{bmatrix}$$

where $\mathbf{x}_f \in \mathbb{R}^{N_f}$ are the followers' states and $\mathbf{x}_l \in \mathbb{R}^{N-N_f}$ the leaders'.

The Laplacian can also be partitioned as:

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_f & \boldsymbol{L}_{fl} \\ \boldsymbol{L}_{fl}^T & \boldsymbol{L}_l \end{bmatrix}$$

where $\boldsymbol{L}_f$ is the followers' Laplacian, $\boldsymbol{L}_l$ the leaders', and $\boldsymbol{L}_{fl}$ is the part in common.

Assume that leaders and followers run the same Laplacian-based distributed control law (i.e., an normal averaging system), the system can be formulated as:

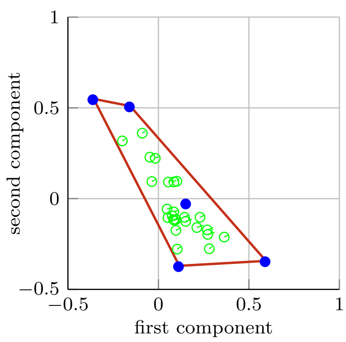
$$\begin{bmatrix} \dot{\mathbf{x}}_f(t) \\ \dot{\mathbf{x}}_l(t) \end{bmatrix} = - \begin{bmatrix} \boldsymbol{L}_f & \boldsymbol{L}_{fl} \\ \boldsymbol{L}_{fl}^T & \boldsymbol{L}_l \end{bmatrix} \begin{bmatrix} \mathbf{x}_f(t) \\ \mathbf{x}_l(t) \end{bmatrix}$$

> **Example.** Consider a path graph with four nodes:
>
> $$0 \leftrightarrow 1 \leftrightarrow 2 \leftrightarrow (3)$$
>
> The nodes $0, 1, 2$ are followers and 3 is a leader. The system is:
>
> $$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \left[\begin{array}{ccc|c} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ \hline 0 & 0 & -1 & 1 \end{array}\right] \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

3.1 Containment

Containment Task where leaders are stationary and the goal is to drive followers within the convex hull enclosing the leaders. Followers can communicate with agents of any type while leaders do not communicate.

Containment



Containment control law Given N_f followers and $N-N_f$ leaders, the control law to solve the containment task have:

Containment control law

- Followers running Laplacian dynamics.
- Leaders being stationary.

The system is:

$$\begin{aligned} \dot{x}_i(t) &= -\sum_{j\in\mathcal{N}_i} a_{ij}\big(x_i(t)-x_j(t)\big) && \forall i \in \{1,\dots,N_f\} \\ \dot{x}_i(t) &= 0 && \forall i \in \{N_f+1,\dots,N\} \end{aligned}$$

In matrix form, it becomes:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= -\boldsymbol{L}\mathbf{x}(t) \\ \begin{bmatrix} \dot{\mathbf{x}}_f(t) \\ \dot{\mathbf{x}}_l(t) \end{bmatrix} &= -\begin{bmatrix} \boldsymbol{L}_f & \boldsymbol{L}_{fl} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_f(t) \\ \mathbf{x}_l(t) \end{bmatrix} \\ \dot{\mathbf{x}}_f(t) &= -\boldsymbol{L}_f\mathbf{x}_f(t) - \boldsymbol{L}_{fl}\mathbf{x}_l \end{aligned}$$

where $\boldsymbol{L}_f$ can be seen as the state matrix and $\boldsymbol{L}_{fl}$ as the input matrix. The input $\mathbf{x}_l = \mathbf{x}_l(0) = \mathbf{x}_l(t)$ is constant.

Lemma 3.1.1. If the interaction graph G between leaders and followers is undirected and connected, then the followers' Laplacian $\boldsymbol{L}_f$ is positive definite.

Proof. We need to prove that:

$$\mathbf{x}_f^T \boldsymbol{L}_f \mathbf{x}_f > 0 \quad \forall \mathbf{x}_f \neq 0$$

As G is undirected, it holds that:

- The complete Laplacian $\boldsymbol{L}$ is symmetric and thus have real-valued eigenvalues.
- By Lemma 2.3.2, all its non-zero eigenvalues are positive.
- By Lemma 2.3.4, as G is connected, the eigenvalue $\lambda = 0$ is simple.

Therefore:

- $\mathbf{x}^T\boldsymbol{L}\mathbf{x} \geq 0$ as all eigenvalues are non-negative.
- $\mathbf{x}^T\boldsymbol{L}\mathbf{x} = 0 \iff \mathbf{x} = \alpha\mathbf{1}$ for $\alpha \in \mathbb{R}$, as $\lambda = 0$ is simple.

The following two arguments can be made:

1. By choosing $\bar{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_f & 0 \end{bmatrix}^T$, it holds that:

$$\begin{aligned} \bar{\mathbf{x}}^T \boldsymbol{L}\bar{\mathbf{x}} &\geq 0 \quad \forall \bar{\mathbf{x}} \\ \begin{bmatrix} \mathbf{x}_f & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{L}_f & \boldsymbol{L}_{fl} \\ \boldsymbol{L}_{fl}^T & \boldsymbol{L}_l \end{bmatrix} \begin{bmatrix} \mathbf{x}_f \\ 0 \end{bmatrix} &\geq 0 \quad \forall \mathbf{x}_f \\ \mathbf{x}_f^T \boldsymbol{L}_f \mathbf{x}_f &\geq 0 \quad \forall \mathbf{x}_f \end{aligned}$$

2. The only case when $\mathbf{x}^T\boldsymbol{L}\mathbf{x} = 0$ for $\mathbf{x} \neq 0$ is with $\mathbf{x} = \alpha\mathbf{1}$ for $\alpha \neq 0$. As $\forall \mathbf{x}_f : \bar{\mathbf{x}} \neq \alpha\mathbf{1}$, it holds that $\forall \mathbf{x}_f : \mathbf{x}_f^T\boldsymbol{L}_f\mathbf{x}_f \neq 0$.

Therefore, $\boldsymbol{L}_f$ is positive definite as $\forall \mathbf{x}_f \neq 0 : \mathbf{x}_f^T \boldsymbol{L}_f \mathbf{x}_f > 0$. $\square$

Lemma 3.1.2. It holds that $\dot{\mathbf{x}}_f = -\boldsymbol{L}_f \mathbf{x}_f$ is globally exponentially stable (i.e., converges to 0 exponentially).

Proof. As $\boldsymbol{L}_f$ is symmetric and positive definite by Lemma 3.1.1, its eigenvalues are real and positive. Therefore, $-\boldsymbol{L}_f$ have real and negative eigenvalues, which is the condition of a globally exponentially stable behavior. □

Theorem 3.1.1 (Containment optimality)**.** Given a leader-follower network such that:

Containment optimality

- Followers run Laplacian dynamics,
- Leaders are stationary,
- The interaction graph G is fixed, undirected, and connected.

It holds that all followers asymptotically converge to a state (not necessarily the same) within the convex hull containing the leaders.

Proof. The proof is done in two parts:

Unique globally asymptotically stable equilibrium We want to prove that the followers' state $\mathbf{x}_f(t)$ converges to some value $\mathbf{x}_{f,E}$ for any initial state. The equilibrium can be found by solving:

$$0 = -\boldsymbol{L}_f \mathbf{x}_{f,E} - \boldsymbol{L}_{fl} \mathbf{x}_l$$

where $\dot{\mathbf{x}}_f = 0$ (i.e., reached convergence) and $\mathbf{x}_{f,E}$ is the equilibrium state.

By Lemma 3.1.1, $\boldsymbol{L}_f$ is positive definite and thus invertible, therefore, we have that:

$$\mathbf{x}_{f,E} = -\boldsymbol{L}_f^{-1} \boldsymbol{L}_{fl} \mathbf{x}_l$$

Let $\mathbf{e}(t) = \mathbf{x}_f(t) - \mathbf{x}_{f,E}$ (intuitively, the distance to equilibrium). As the rate of change of $\mathbf{e}(t)$ depends only on $\mathbf{x}_f(t)$ (i.e., $\dot{\mathbf{e}}(t) = \dot{\mathbf{x}}_f(t)$), we have that:

$$\begin{aligned}
\dot{\mathbf{e}}(t) &= \dot{\mathbf{x}}_f(t) \\
&= -\boldsymbol{L}_f \mathbf{x}_f(t) - \boldsymbol{L}_{fl} \mathbf{x}_l \\
&= -\boldsymbol{L}_f (\mathbf{e}(t) + \mathbf{x}_{f,E}) - \boldsymbol{L}_{fl} \mathbf{x}_l \\
&= -\boldsymbol{L}_f \mathbf{e}(t) + \cancel{\boldsymbol{L}_f \boldsymbol{L}_f^{-1} \boldsymbol{L}_{fl} \mathbf{x}_l} - \cancel{\boldsymbol{L}_{fl} \mathbf{x}_l}
\end{aligned}$$

Lemma 3.1.3. Any equilibrium or trajectory based on an LTI system enjoys the same stability property of that system.

As Lemma 3.1.2 states that $\dot{\mathbf{x}}_f = -\boldsymbol{L}_f \mathbf{x}_f$ is globally asymptotically stable, by Lemma 3.1.3, it holds that $\dot{\mathbf{e}}(t) = -\boldsymbol{L}_f \mathbf{e}(t)$ is also a globally asymptotically stable system and $\mathbf{x}_{f,E}$ is the unique globally stable equilibrium of the followers' dynamics.

Equilibrium within convex hull We want to prove that each element of $\mathbf{x}_{f,E}$ falls within the convex hull of the leaders.

For simplicity, let us denote the states vector as $\mathbf{x}_E = \begin{bmatrix} \mathbf{x}_{f,E} & \mathbf{x}_l \end{bmatrix}^T$ and its i-th component as $x_{E,i}$.

The dynamics at convergence of the i-th follower is:

$$0 = -\sum_{j=1}^{N} a_{ij}(x_{E,i} - x_{E,j}) \quad \forall i \in \{1, \dots, N_f\}$$

Therefore, we have that:

$$\begin{aligned}
\left(\sum_{j=1}^{N} a_{ij}\right) x_{E,i} &= \sum_{j=1}^{N} a_{ij} x_{E,j} & \forall i \in \{1, \dots, N_f\} \\
x_{E,i} &= \sum_{j=1}^{N} \frac{a_{ij}}{\sum_{k=1}^{N} a_{ik}} x_{E,j} & \forall i \in \{1, \dots, N_f\}
\end{aligned}$$

As $\frac{a_{ij}}{\sum_{k=1}^{N} a_{ik}}$ define a convex combination (i.e., sum of all of them is 1), each follower's equilibrium $x_{E,i}$ belongs to the convex hull of all the other agents (both leaders and followers). As leaders are stationary, they are not affected by this constraint and it can be concluded that followers' equilibria fall within the convex hull of the leaders.

□

Remark (Leader-follower containment weakness)**.** The final part of the proof of Theorem 3.1.1 also shows that if there is an adversarial follower that does not change its state, all others will converge towards it.

3.2 Containment with non-static leaders

Containment with non-static leaders Containment problem where leaders' dynamics is a non-zero constant (i.e., they also move):

Containment with non-static leaders

$$\begin{aligned}
\dot{\mathbf{x}}_f(t) &= -\boldsymbol{L}_f \mathbf{x}_f(t) - \boldsymbol{L}_{fl} \mathbf{x}_l(t) & \mathbf{x}_f(0) &= \mathbf{x}_f^{(0)} \\
\dot{\mathbf{x}}_l(t) &= \mathbf{v}_0 & \mathbf{x}_l(0) &= \mathbf{x}_l^{(0)}
\end{aligned}$$

where $\mathbf{v}_0$ is the leaders' velocity.

Theorem 3.2.1 (Containment with non-static leaders non-equilibrium)**.** Naive containment with non-static leaders do not have an equilibrium.

Proof. Ideally, the equilibria for followers' and leader's dynamics are:

$$\begin{aligned}
0 &= -\boldsymbol{L}_f \mathbf{x}_{f,E} - \boldsymbol{L}_{fl} \mathbf{x}_{l,E} \\
0 &= \mathbf{v}_0
\end{aligned}$$

Let's define the containment error (can also be seen as the error to reach the followers' equilibrium) as:

$$\mathbf{e}(t) = \boldsymbol{L}_f \mathbf{x}_f(t) + \boldsymbol{L}_{fl} \mathbf{x}_l(t)$$

Its dynamics depends on the ones of the followers' and leaders':

$$\begin{aligned}
\dot{\mathbf{e}}(t) &= \boldsymbol{L}_f \dot{\mathbf{x}}_f(t) + \boldsymbol{L}_{fl} \dot{\mathbf{x}}_l(t) \\
&= \boldsymbol{L}_f(-\boldsymbol{L}_f \mathbf{x}_f(t) - \boldsymbol{L}_{fl} \mathbf{x}_l(t)) + \boldsymbol{L}_{fl} \mathbf{v}_0 \\
&= -\boldsymbol{L}_f \mathbf{e}(t) + \boldsymbol{L}_{fl} \mathbf{v}_0
\end{aligned}$$

By inspecting the value of the containment error $\mathbf{e}(t)$ when it reaches equilibrium we have that:

$$\begin{aligned}
& 0 = \dot{\mathbf{e}}(t) \\
\iff & 0 = -\boldsymbol{L}_f \mathbf{e}(t) + \boldsymbol{L}_{fl} \mathbf{v}_0 \\
\iff & \mathbf{e}(t) = \boldsymbol{L}_f^{-1} \boldsymbol{L}_{fl} \mathbf{v}_0
\end{aligned}$$

There are two cases:

$$\mathbf{e}(t) = \begin{cases} 0 & \text{if } \mathbf{v}_0 = 0 \text{ (i.e., same case of Theorem 3.1.1)} \\ \boldsymbol{L}_f^{-1}\boldsymbol{L}_{fl}\mathbf{v}_0 & \text{if } \mathbf{v}_0 \neq 0 \end{cases}$$

Therefore, when leaders are non-static, the containment error converges to a non-zero constant. Thus, followers' equilibrium is never reached (i.e., they keep moving) and the containment problem cannot be solved. □

3.3 Containment with non-static leaders and integral action

Containment with non-static leaders and integral action

Containment with non-static leaders and integral action Leader-follower dynamics defined as:

$$\begin{aligned} \dot{\mathbf{x}}_f(t) &= -\boldsymbol{L}_f\mathbf{x}_f(t) - \boldsymbol{L}_{fl}\mathbf{x}_l(t) + \mathbf{u}_f(t) \qquad & \mathbf{x}_f(0) = \mathbf{x}_f^{(0)} \\ \dot{\mathbf{x}}_l(t) &= \mathbf{v}_0 & \mathbf{x}_l(0) = \mathbf{x}_l^{(0)} \end{aligned}$$

where $\mathbf{u}_f(t)$ is a distributed control action (can be seen as a correction) that processes the containment error $\mathbf{e}(t)$. It is composed of a proportional controller (i.e., value proportional to the error) and an integral controller (i.e., value proportional to the integral to the error):

$$\mathbf{u}_f(t) = \boldsymbol{K}_P\mathbf{e}(t) + \boldsymbol{K}_I \int_0^t \mathbf{e}(\tau)\, d\tau$$

where $\boldsymbol{K}_P$ and $\boldsymbol{K}_I$ are coefficients for the proportional and integral controller, respectively.

By defining a proxy ξ for the integral of the error (i.e., sort of accumulator) as follows:

$$\begin{aligned} \dot{\xi}(t) &= \mathbf{e}(t) \\ &= \boldsymbol{L}_f\mathbf{x}(t) + \boldsymbol{L}_{fl}\mathbf{x}_l(t) \qquad \xi(0) = \xi^{(0)} \end{aligned}$$

The control action can be defined as:

$$\mathbf{u}_f(t) = \boldsymbol{K}_P\mathbf{e}(t) + \boldsymbol{K}_I\xi(t)$$

In the simplest case, $\mathbf{u}_f(t)$ is a pure integral control where $\boldsymbol{K}_I = -\kappa_I\boldsymbol{I}, \kappa_I > 0$ is a sparse matrix (e.g., diagonal) and $\boldsymbol{K}_P = 0$. The overall system can be defined in matrix form as:

$$\begin{bmatrix} \dot{\mathbf{x}}_f(t) \\ \dot{\mathbf{x}}_l(t) \\ \dot{\xi}(t) \end{bmatrix} = \begin{bmatrix} -\boldsymbol{L}_f & -\boldsymbol{L}_{fl} & \boldsymbol{K}_I \\ 0 & 0 & 0 \\ \boldsymbol{L}_f & \boldsymbol{L}_{fl} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_f(t) \\ \mathbf{x}_l(t) \\ \xi(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{I} \\ 0 \end{bmatrix} \mathbf{v}_0$$

Remark. The value of this formulation of the control action for an agent i is:

$$u_{F_i}(t) = \kappa_I\xi_i(t)$$

It can be seen that it is computable as a distributed system as κ_I is constant and $\xi_i(t)$ is based on the Laplacian (i.e., it is sufficient to look up the neighbors' states).

Theorem 3.3.1 (Containment with non-static leaders and integral action optimality)**.** With the integral action, containment with non-static leaders converges to a valid solution.

3.4 Containment with discrete-time

Containment with discrete-time Containment can be discretized using the forward-Eurler discretization. Its dynamics is defined as:

Containment with discrete-time

$$\begin{aligned}
\dot{\mathbf{x}}_i(t) &= -\sum_{j \in \mathcal{N}_i} a_{ij}(x_i(t) - x_j(t)) && \forall i \in \{1, \dots, N_f\} \\
\dot{\mathbf{x}}_i(t) &= 0 && \forall i \in \{N_f + 1, \dots, N\}
\end{aligned}$$

And the followers' states are sampled with a time-step $\varepsilon > 0$ while the leaders' is constant:

$$\begin{aligned}
x_i^{k+1} &= x_i(t)|_{t=(k+1)\varepsilon} \\
&= x_i^k + \varepsilon\, \dot{x}_i(t)|_{t=k\varepsilon} \\
&= \left(1 - \varepsilon \sum_{j \in \mathcal{N}_i} a_{ij}\right) x_i^k + \varepsilon \sum_{j \in \mathcal{N}_i} a_{ij} x_j^k \qquad \forall i \in \{1, \dots, N_f\} \\
x_i^{k+1} &= x_i^k \qquad \forall i \in \{N_f + 1, \dots, N\}
\end{aligned}$$

In matrix form, it can be defined as:

$$\begin{bmatrix} \mathbf{x}_f^{k+1} \\ \mathbf{x}_l^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} - \varepsilon \boldsymbol{L}_f & -\varepsilon \boldsymbol{L}_{fl} \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_f^k \\ \mathbf{x}_l^k \end{bmatrix}$$

3.5 Containment with multivariate states

Containment with multivariate states With multivariate states, it can be shown that the dynamics is described as:

Containment with multivariate states

$$\dot{\mathbf{x}}(t) = -\boldsymbol{L} \otimes \boldsymbol{I}_d \mathbf{x}(t)$$

where $\otimes$ is the Kronecker product.

4 Optimization

4.1 Definitions

4.1.1 Unconstrained optimization

Unconstrained optimization Problem of form: Unconstrained optimization

$$\min_{\mathbf{z} \in \mathbb{R}^d} l(\mathbf{z})$$

where $l : \mathbb{R}^d \to \mathbb{R}$ is the cost function and $\mathbf{z}$ the decision variables.

Theorem 4.1.1 (First-order necessary condition of optimality)**.** Given a point $\mathbf{z}^*$ and a cost function $l : \mathbb{R}^d \to \mathbb{R}$ such that $l \in C^1$ in $B(\mathbf{z}^*, \varepsilon)$ (i.e., neighbors of $\mathbf{z}^*$ within a radius ε), it holds that: First-order necessary condition of optimality

$$\mathbf{z}^* \text{ is local minimum } \Rightarrow \nabla l(\mathbf{z}^*) = 0$$

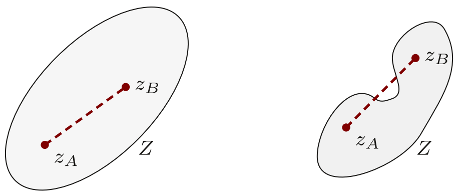
Theorem 4.1.2 (Second-order necessary condition of optimality)**.** Given a point $\mathbf{z}^*$ and a cost function $l : \mathbb{R}^d \to \mathbb{R}$ such that $l \in C^2$ in $B(\mathbf{z}^*, \varepsilon)$, it holds that: Second-order necessary condition of optimality

$$\mathbf{z}^* \text{ is local minimum } \Rightarrow \nabla^2 l(\mathbf{z}^*) \geq 0 \text{ (i.e., positive semidefinite)}$$

4.1.2 Convexity

Convex set A set $Z \subseteq \mathbb{R}^d$ is convex if it holds that: Convex set

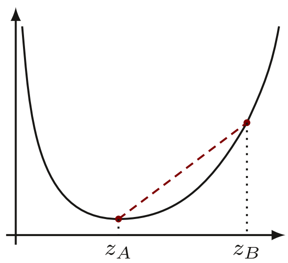
$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z : \Big(\exists \alpha \in [0, 1] : (\alpha \mathbf{z}_A + (1 - \alpha)\mathbf{z}_B) \in Z \Big)$$



Convex set Nonconvex set

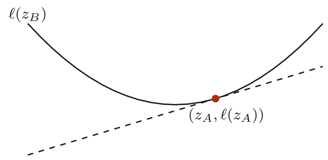
Convex function Given a convex set $Z \subseteq \mathbb{R}^d$, a function $l : Z \to \mathbb{R}$ is convex if it holds that: Convex function

$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z : \Big(\exists \alpha \in [0, 1] : l(\alpha \mathbf{z}_A + (1 - \alpha)\mathbf{z}_B) \leq \alpha l(\mathbf{z}_A) + (1 - \alpha) l(\mathbf{z}_B) \Big)$$



Remark. Given a differentiable and convex function $l : Z \to \mathbb{R}$, it holds that any of its points lie above all its tangents:

$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z : l(\mathbf{z}_B) \geq l(\mathbf{z}_A) + \nabla l(\mathbf{z}_A)^T(\mathbf{z}_B - \mathbf{z}_A)$$



Strongly convex function

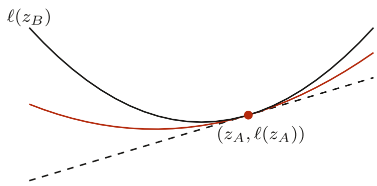
Strongly convex function Given a convex set $Z \subseteq \mathbb{R}^d$, a function $l : Z \to \mathbb{R}$ is strongly convex with parameter $\mu > 0$ if it holds that:

$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z, \mathbf{z}_A \neq \mathbf{z}_B : \Big(\exists \alpha \in (0,1) : l(\alpha \mathbf{z}_A + (1-\alpha)\mathbf{z}_B) < \alpha l(\mathbf{z}_A) + (1-\alpha) l(\mathbf{z}_B) - \frac{1}{2}\mu\alpha(1-\alpha)\|\mathbf{z}_A - \mathbf{z}_B\|^2\Big)$$

Intuitively, it is strictly convex and grows as fast as a quadratic function.

Remark. Given a differentiable and μ-strongly convex function $l : Z \to \mathbb{R}$, it holds that any of its points lie above all the paraboloids with curvature determined by μ and tangent to a point of the function:

$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z : l(\mathbf{z}_B) \geq l(\mathbf{z}_A) + \nabla l(\mathbf{z}_A)^T(\mathbf{z}_B - \mathbf{z}_A) + \frac{\mu}{2}\|\mathbf{z}_B - \mathbf{z}_A\|^2$$



A geometric interpretation is that strong convexity imposes a quadratic lower-bound to the function.

Convexity and gradient monotonicity

Lemma 4.1.1 (Convexity and gradient monotonicity)**.** Given a differentiable and convex function l, its gradient ∇l is a monotone operator, which means that it satisfies:

$$\forall \mathbf{z}_A, \mathbf{z}_B : \big(\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\big)^T(\mathbf{z}_A - \mathbf{z}_B) \geq 0$$

Strict convexity and gradient monotonicity

Lemma 4.1.2 (Strict convexity and gradient monotonicity)**.** Given a differentiable and strictly convex function l, its gradient ∇l is a strictly monotone operator, which means that it satisfies:

$$\forall \mathbf{z}_A, \mathbf{z}_B : \big(\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\big)^T(\mathbf{z}_A - \mathbf{z}_B) > 0$$

Strong convexity and gradient monotonicity

Lemma 4.1.3 (Strong convexity and gradient monotonicity)**.** Given a differentiable and μ-strongly convex function l, its gradient ∇l is a strongly monotone operator, which means that it satisfies:

$$\forall \mathbf{z}_A, \mathbf{z}_B : \big(\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\big)^T(\mathbf{z}_A - \mathbf{z}_B) \geq \mu\|\mathbf{z}_A - \mathbf{z}_B\|^2$$

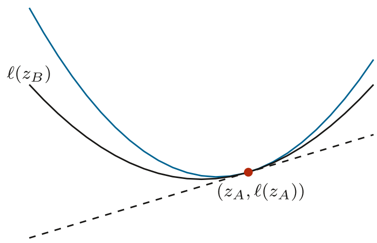
Lipschitz continuity

Lipschitz continuity Given a function l, it is Lipschitz continuous with parameter $L > 0$ if:

$$\forall \mathbf{z}_A, \mathbf{z}_B : \|l(\mathbf{z}_A) - l(\mathbf{z}_B)\| \leq L\|\mathbf{z}_A - \mathbf{z}_B\|$$

Remark. Given a differentiable function l with L-Lipschitz continuous gradient ∇l, it holds that any of its points lie below all the paraboloids with curvature determined by L and tangent to a point of the function:

$$\forall \mathbf{z}_A, \mathbf{z}_B \in Z : l(\mathbf{z}_B) \leq l(\mathbf{z}_A) + \nabla l(\mathbf{z}_A)^T (\mathbf{z}_B - \mathbf{z}_A) + \frac{L}{2} \|\mathbf{z}_B - \mathbf{z}_A\|^2$$



A geometric interpretation is that Lipschitz continuity of the gradient imposes a quadratic upper-bound to the function.

Lemma 4.1.4 (Convexity and Lipschitz continuity of gradient)**.** Given a differentiable convex function l with L-Lipschitz continuous gradient ∇l, its gradient is a co-coercive operator, which means that it satisfies:

Convexity and Lipschitz continuity of gradient

$$\forall \mathbf{z}_A, \mathbf{z}_B : \Big(\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\Big)^T (\mathbf{z}_A - \mathbf{z}_B) \geq \frac{1}{L} \|\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\|^2$$

Lemma 4.1.5 (Strong convexity and Lipschitz continuity of gradient)**.** Given a differentiable μ-strongly convex function l with L-Lipschitz continuous gradient ∇l, its gradient is a strongly co-coercive operator, which means that it satisfies:

Strong convexity and Lipschitz continuity of gradient

$$\forall \mathbf{z}_A, \mathbf{z}_B : \Big(\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\Big)^T (\mathbf{z}_A - \mathbf{z}_B) \geq \underbrace{\frac{\mu L}{\mu + L}}_{\gamma_1} \|\mathbf{z}_A - \mathbf{z}_B\|^2 + \underbrace{\frac{1}{\mu + L}}_{\gamma_2} \|\nabla l(\mathbf{z}_A) - \nabla l(\mathbf{z}_B)\|^2$$

4.2 Iterative descent methods

Theorem 4.2.1. Given a convex function l, it holds that a local minimum of l is also global.
Moreover, in the unconstrained optimization case, the first-order necessary condition of optimality is sufficient for a global minimum.

Theorem 4.2.2. Given a convex function l, it holds that $\mathbf{z}^*$ is a global minimum if and only if $\nabla f(\mathbf{z}^*) = 0$.

Iterative descent Given a function l and an initial guess $\mathbf{z}^0$, an iterative descent algorithm iteratively moves to new points $\mathbf{z}^k$ such that:

Iterative descent

$$\forall k \in \mathbb{N} : l(\mathbf{z}^{k+1}) < l(\mathbf{z}^k)$$

4.2.1 Gradient method

Gradient method Algorithm that given the function l to minimize and the initial guess $\mathbf{z}^0$, computes the update as:

Gradient method

$$\mathbf{z}^{k+1} = \mathbf{z}^k - \alpha^k \nabla l(\mathbf{z}^k)$$

where $\alpha^k > 0$ is the step size and $-\nabla l(\mathbf{z}^k)$ is the step direction.

Theorem 4.2.3. For a sufficiently small $\alpha^k > 0$, the gradient method is an iterative descent algorithm:

$$l(\mathbf{z}^{k+1}) < l(\mathbf{z}^k)$$

Proof. Consider the first-order Taylor approximation of $l(\mathbf{z}^{k+1})$ about $\mathbf{z}^k$:

$$\begin{aligned} l(\mathbf{z}^{k+1}) &= l(\mathbf{z}^k) + \nabla l(\mathbf{z}^k)^T(\mathbf{z}^{k+1} - \mathbf{z}^k) + o(\|\mathbf{z}^{k+1} - \mathbf{z}^k\|) \\ &= l(\mathbf{z}^k) - \alpha^k \|\nabla l(\mathbf{z}^k)\|^2 + o(\alpha^k) \end{aligned}$$

Therefore, $l(\mathbf{z}^{k+1}) < l(\mathbf{z}^k)$ for some α^k. □

Remark (Step size choice)**.** Possible choices for the step size are: Step size choice

Constant $\forall k \in \mathbb{N} : \alpha^k = \alpha > 0$.

Diminishing $\alpha^k \overset{k\to\infty}{\longrightarrow} 0$. To avoid decreasing the step too much, a typical choice is an α^k such that:

$$\sum_{k=0}^{\infty} \alpha^k = \infty \qquad \sum_{k=0}^{\infty} (\alpha^k)^2 < \infty$$

Line search Algorithmic methods such as the Armijo rule.

Generalized gradient method Gradient method where the update rule is generalized as: Generalized gradient method

$$\mathbf{z}^{k+1} = \mathbf{z}^k - \alpha^k \boldsymbol{D}^k \nabla l(\mathbf{z}^k)$$

where $\boldsymbol{D}^k \in \mathbb{R}^{d\times d}$ is uniformly positive definite (i.e., $\delta_1 \boldsymbol{I} \leq \boldsymbol{D}^k \leq \delta_2 \boldsymbol{I}$ for some $\delta_2 \geq \delta_1 > 0$).

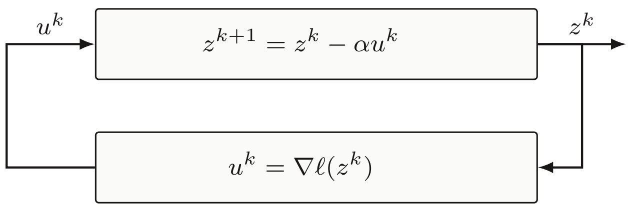
Possible choices for $\boldsymbol{D}^k$ are:

- Steepest descent: $\boldsymbol{D}^k = \boldsymbol{I}$.
- Newton's method: $\boldsymbol{D}^k = (\nabla^2 l(\mathbf{z}^k))^{-1}$.
- Quasi-Newton method: $\boldsymbol{D}^k = (H(\mathbf{z}^k))^{-1}$, where $H(\mathbf{z}^k) \approx \nabla^2 l(\mathbf{z}^k)$.

Gradient method as discrete-time integrator with feedback The gradient method can be interpreted as a discrete-time integrator with a feedback loop. This means that it is composed of: Gradient method as discrete-time integrator with feedback

Integrator A linear system that defines the update: $\mathbf{z}^{k+1} = \mathbf{z}^k - \alpha \mathbf{u}^k$.

Plant A non-linear (bounded) function whose output is re-injected into the integrator. In this case, it is the gradient: $\mathbf{u}^k = \nabla l(\mathbf{z}^k)$.



Theorem 4.2.4 (Gradient method convergence)**.** Consider a function l such that: Gradient method convergence

- ∇l is L-Lipschitz continuous,

- The step size is constant or diminishing.

Let $\{\mathbf{z}^k\}_{k\in\mathbb{N}}$ be the (bounded) sequence generated by the gradient method. It holds that every limit point $\bar{\mathbf{z}}$ of the sequence $\{\mathbf{z}^k\}_{k\in\mathbb{N}}$ is a stationary point (i.e., $\nabla l(\bar{z}) = 0$).
In addition, if l is μ-strongly convex and the step size is constant, then the convergence rate of the sequence $\{\mathbf{z}^k\}_{k\in\mathbb{N}}$ is exponential (also said geometric or linear):

$$\|\mathbf{z}^k - \mathbf{z}^*\| \leq M\rho^k$$

where $\rho \in (0, 1)$ and $M > 0$ depends on μ, L, and $\|\mathbf{z}^0 - \mathbf{z}^*\|$.

Proof. We need to prove the two parts of the theorem:

1. We want to prove that any limit point of the sequence generated by the gradient method is a stationary point.

   In other words, by considering the gradient method as an integrator with feedback, we want to analyze the equilibrium of the system. Assume that the system converges to some equilibrium $\mathbf{z}_E$. To be an equilibrium, it must be that the feedback loop stopped updating the system (i.e., $\mathbf{u}^k = 0$ for k after some threshold) so that:

   $$\mathbf{z}_E = \mathbf{z}_E - \alpha\nabla l(\mathbf{z}_E)$$

   Therefore, an equilibrium point is necessarily a stationary point of l as it must be that $\nabla l(\mathbf{z}_E) = 0$.

2. We want to prove that if l is μ-strongly convex and the step size is constant, the sequence converges exponentially.

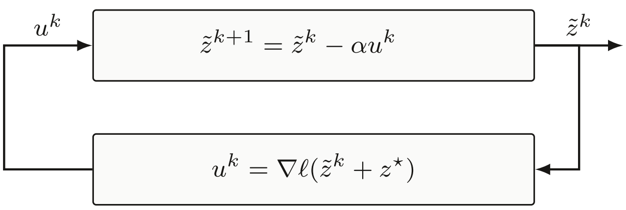
   Remark. As l is convex, its equilibrium is also the global minimum $\mathbf{z}^*$.

   Consider the following change in coordinates (i.e., a translation):

   $$\mathbf{z}^k \mapsto \tilde{\mathbf{z}}^k$$
   $$\text{with } \tilde{\mathbf{z}}^k = \mathbf{z}^k - \mathbf{z}_E = \mathbf{z}^k - \mathbf{z}^*$$

   The system in the new coordinates becomes:

   $$\begin{aligned}\tilde{\mathbf{z}}^{k+1} &= \tilde{\mathbf{z}}^k - \alpha\mathbf{u}^k \\ \mathbf{u}^k &= \nabla l(\mathbf{z}^k) \\ &= \nabla l(\tilde{\mathbf{z}}^k + \mathbf{z}^*) \\ &= \nabla l(\tilde{\mathbf{z}}^k + \mathbf{z}^*) - \nabla l(\mathbf{z}^*) \qquad \nabla l(\mathbf{z}^*) = 0\text{, but useful for Lemma 4.1.5}\end{aligned}$$



   Remark. As l is strongly convex and its gradient Lipschitz continuous, by Lemma 4.1.5 it holds that:

   $$-(\mathbf{u}^k)^T\tilde{\mathbf{z}}^k \leq -\gamma_1\|\tilde{\mathbf{z}}^k\|^2 - \gamma_2\|\tilde{\mathbf{u}}^k\|^2$$

Consider a Lyapunov function $V : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ defined as:

$$V(\tilde{\mathbf{z}}) = \|\tilde{\mathbf{z}}\|^2$$

It holds that:

$$\begin{aligned}
V(\tilde{\mathbf{z}}^{k+1}) - V(\tilde{\mathbf{z}}^k) &= \|\tilde{\mathbf{z}}^{k+1}\|^2 - \|\tilde{\mathbf{z}}^k\|^2 \\
&= \cancel{\|\tilde{\mathbf{z}}^k\|^2} - 2\alpha(\mathbf{u}^k)^T\tilde{\mathbf{z}}^k + \alpha^2\|\mathbf{u}^k\|^2 - \cancel{\|\tilde{\mathbf{z}}^k\|^2} && \text{Lemma 4.1.5} \\
&\leq -2\alpha\gamma_1\|\tilde{\mathbf{z}}^k\|^2 + \alpha(\alpha - 2\gamma_2)\|\mathbf{u}^k\|^2
\end{aligned}$$

By choosing $\alpha \leq 2\gamma_2$, we have that:

$$\begin{aligned}
&V(\tilde{\mathbf{z}}^{k+1}) - V(\tilde{\mathbf{z}}^k) \leq -2\alpha\gamma_1\|\tilde{\mathbf{z}}^k\|^2 \\
\iff &\|\tilde{\mathbf{z}}^{k+1}\|^2 - \|\tilde{\mathbf{z}}^k\|^2 \leq -2\alpha\gamma_1\|\tilde{\mathbf{z}}^k\|^2 \\
\iff &\|\tilde{\mathbf{z}}^{k+1}\|^2 \leq (1 - 2\alpha\gamma_1)\|\tilde{\mathbf{z}}^k\|^2
\end{aligned}$$

Finally, as the gradient method is an iterative descent algorithm, it holds that:

$$\begin{aligned}
\|\tilde{\mathbf{z}}^{k+1}\|^2 &\leq (1 - 2\alpha\gamma_1)\|\tilde{\mathbf{z}}^k\|^2 \\
&\leq \dots \\
&\leq (1 - 2\alpha\gamma_1)^k\|\tilde{\mathbf{z}}^0\|^2
\end{aligned}$$

Therefore, the sequence $\{\tilde{\mathbf{z}}^k\}_{k\in\mathbb{R}}$ goes exponentially fast to zero and we have shown that:

$$\begin{aligned}
\|\mathbf{z}^{k+1} - \mathbf{z}^*\|^2 &\leq (1 - 2\alpha\gamma_1)^k\|\mathbf{z}^0 - \mathbf{z}^*\|^2 \\
&= \rho^k M
\end{aligned}$$

□

Remark (Gradient method for a quadratic function)**.** Given the problem of minimizing a quadratic function:

Gradient method for a quadratic function

$$\min_{\mathbf{z}} \frac{1}{2}\mathbf{z}^T\boldsymbol{Q}\mathbf{z} + \mathbf{r}^T\mathbf{z} \qquad \nabla l = \boldsymbol{Q}\mathbf{z}^k + \mathbf{r}$$

The gradient method can be reduced to an affine linear system:

$$\begin{aligned}
\mathbf{z}^{k+1} &= \mathbf{z}^k - \alpha(\boldsymbol{Q}\mathbf{z}^k + \mathbf{r}) \\
&= (\boldsymbol{I} - \alpha\boldsymbol{Q})\mathbf{z}^k - \alpha\mathbf{r}
\end{aligned}$$

For a sufficiently small α, the matrix $(\boldsymbol{I} - \alpha\boldsymbol{Q})$ is Schur (i.e., $\forall\boldsymbol{\rho}, |\boldsymbol{\rho}| < 1 : \sum_{i=0}^{\infty}\boldsymbol{\rho}^i = (1 - \boldsymbol{\rho})^{-1}$). Therefore, the solution can be computed in closed form as:

$$\begin{aligned}
\mathbf{z}^k &= (\boldsymbol{I} - \alpha\boldsymbol{Q})^k\mathbf{z}^0 - \alpha\sum_{\tau=0}^{k-1}(\boldsymbol{I} - \alpha\boldsymbol{Q})^\tau\mathbf{r} \\
&\xrightarrow{k\to\infty} -\alpha\left(\sum_{\tau=0}^{\infty}(\boldsymbol{I} - \alpha\boldsymbol{Q})^\tau\right)\mathbf{r} = -\boldsymbol{Q}^{-1}\mathbf{r}
\end{aligned}$$

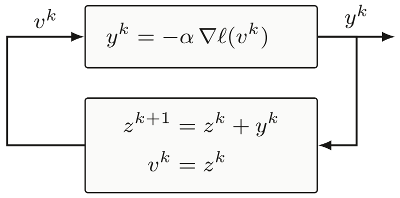
Remark (Gradient flow)**.** By inverting the integrator and plant of the discrete-time integrator of the gradient method, and considering the continuous-time case, the result is the

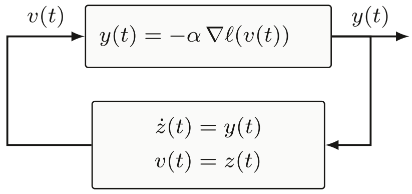
Gradient flow

gradient flow:

$$\dot{\mathbf{z}}(t) = -\nabla l(\mathbf{z}(t))$$

which has a solution if the vector field is Lipschitz continuous.





4.2.2 Accelerated gradient methods

Heavy-ball method

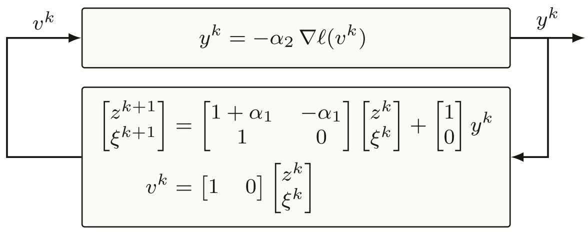
Heavy-ball method Given η^0 and η^{-1}, the algorithm is defined as:

$$\eta^{k+1} = \eta^k + \alpha_1(\eta^k - \eta^{k-1}) - \alpha_2 \nabla l(\eta^k)$$

with $\alpha_1, \alpha_2 > 0$.

Remark. With $\alpha_1 = 0$, the algorithm is reduced to the gradient method with step size α_2.

Remark. The algorithm admits a state-space representation as a discrete-time integrator with a feedback loop:



Note that the matrix $\begin{bmatrix} 1+\alpha_1 & -\alpha_1 \\ 1 & 0 \end{bmatrix}$ is row stochastic.

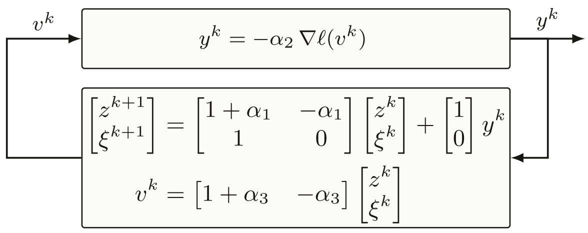
Generalized heavy-ball method

Generalized heavy-ball method Given ζ^0 and ζ^{-1}, the algorithm is defined as:

$$\zeta^{k+1} = \zeta^k + \alpha_1(\zeta^k - \zeta^{k-1}) - \alpha_2 \nabla l(\zeta^k + \alpha_3(\zeta^k - \zeta^{k-1}))$$

with $\alpha_1, \alpha_2, \alpha_3 > 0$.

Remark. The algorithm admits a state-space representation as a discrete-time integrator with a feedback loop:



4.3 Parallel optimization

Cost-coupled optimization Problem of minimizing N cost functions $l_i : \mathbb{R}^d \rightarrow \mathbb{R}$, each local and private to an agent:

Cost-coupled optimization

$$\min_{\mathbf{z} \in \mathbb{R}^d} \sum_{i=1}^{N} l_i(\mathbf{z})$$

Batch gradient method Compute the gradient method direction by considering all the losses:

Batch gradient method

$$\mathbf{z}^{k+1} = \mathbf{z}^k - \alpha \sum_{i=1}^{N} \nabla l_i(\mathbf{z}^k)$$

Remark. Computation in this way can be expensive.

Incremental gradient method At each iteration k, compute the direction by considering the loss of a single agent i^k:

Incremental gradient method

$$\mathbf{z}^{k+1} = \mathbf{z}^k - \alpha \nabla l_{i^k}(\mathbf{z}^k)$$

Remark. Two possible rules to select the agent at each iteration are:

Cyclic $i^k = 1, 2, \dots, N, 1, 2, \dots, N, \dots$

Randomized Draw i^k from a uniform distribution.