Distributed Autonomous Systems M

Dante Piotto

spring semester 2024

Contents

Chapter 1

Introduction and scenarios

1.1 Distributed Autonomous System

Each agent $i \in \{1, \dots, N\}$ has

- local physical and/or cyber state x_i
- computational and sensing capabilities
- communication capability: exchange messages with "neighbours"

1.2 Scenarios and applications of distributed systems

- Averaging: distributed estimation, opinion dynamics
- Distrbuted control in cooperative robotics
- Distributed optimization
  - distributed machine learning
  - distributed decision-making in cooperative robotics
  - distributed optimal control in energy systems and cooperative robotics

1.3 Measurement filtering in wireless sensor networks

Consider a network of N sensors with local sensing, computation and communication. Agent $i, i \in \{1, \dots, N\}$, takes a local measurement from the environment (temperature, pressure, etc.). Let $x_{i0} \in \mathbb{R}$ be the scalar local measurement Agents are interested in agreeing on the average of the measurements,

$$x_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} x_{i0}$$

to have a better estimate of the environment quantity

Consider the following "distributed algorithm" based on "local" linear averaging, for each $i \in \{1, \dots, N\}$

$$x_i^0 = x_{i0}$$
$$x_i^{k+1} = \text{average}(x_i^k, \{x_j^k, j \text{ "neighbour" of } i\}), \qquad k \in \mathbb{N}$$

generalizing coefficients of the update:

$$x_i^0 = x_{i0}$$
$$x_i^{k+1} = \sum_{j=1}^{N} a_{ij} x_j^k \qquad k \in \mathbb{N}$$

Remark. $a_{ij} \geq 0$ and $\sum_{j=1}^{N} a_{ij} = 1$

Remark. $a_{ij} = 0$, for some $j \in \{1, \ldots, N\}$, i.e. $a_{ij} = 0$ if i does not have access to the estimate of j

1.4 Parameter Estimation in Wireless Sensor Networks

Consider a network of N sensors with local sensing, computation and communication aiming at estimating a common parameter $\theta^* \in \mathbb{R}$ Each sensor i measures

$$y_i = B_i\theta^* + v_i$$

with $y_i \in \mathbb{R}^{m_1}, B_i$ known matrix and v_i a random measurement noise. Assume $v_1, \ldots, v_N$ independent and Gaussian, with zero mean and covariance $E[v_i v_i^T] = \Sigma_i$. Assume $\sum_{i=1}^{N} m_i \geq m$ and $\begin{bmatrix} B_1 \\ \vdots \\ B_N \end{bmatrix}$ full rank Compute a least-squares estimate

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{N} (y_i - B_i\theta)^T \Sigma_i^{-1} (y_i - B_i\theta)$$

The optimal solution is

$$\begin{aligned} \hat{\theta} &= \left(\sum_{i=1}^{N} B_i^T \Sigma_i^{-1} B_i \right)^{-1} \sum_{i=1}^{N} B_i^T \Sigma_i^{-1} y_i \\ &= \left(\frac{1}{N} \sum_{i=1}^{N} B_i^T \Sigma_i^{-1} B_i \right)^{-1} \frac{1}{N} \sum_{i=1}^{N} B_i^T \Sigma_i^{-1} y_i \end{aligned}$$

The optimal solution can be obtained by computing two averages $\frac{1}{N}\sum_{i=1}^{N} \beta_i$ and $\frac{1}{N}\sum_{i=1}^{N} \beta_i$

1.4.1 Opinion Dynamics in Social Influence Networks

Group of N individuals, with x_i^k being the opinion of individual i at time k. Opinions are updated according to

$$x_i^{k+1} = \sum_{j=1}^{N} a_{ij} x_j^k$$

1.5 Main questions in averaging algorithms

- Do node estimates converge? Do they converge to a common value ("reach consensus")?
- Do they reach consensus to the average ("average consensus")?
- How can we model communication in general networks?
- Can we answer the above questions for general networks and communication protocols?
- What assumptions do we need on the communication network?

1.6 Distributed control in cooperative robotics

Team of N (mobile) robots aiming to execute complex tasks

Basic tasks

- rendevous, containment
- formation, flocking
- coverage

Complex tasks

- pickup and delivery
- surveillance and patrolling
- exploration
- satellite constellation

1.6.1 Main questions in cooperative robotics

- Do robot states asymptotically converge?
- Do the asympototic staes satisfy the global, desired task?
- How can we model communication in (general) robotic networks?
- What assumptions do we need on the communication network?
- Can we answer the above questions for general networks and communication protocols?

1.6.2 Distributed optimal control

$$
\begin{aligned}
&\min_{\substack{x_1,\dots,x_N \\ u_1,\dots,u_N}} \sum_{i=1}^{N} \Big(\sum_{\tau=0}^{T-1} \ell_i(z_{i,\tau}, u_{i,\tau}) + m_i(z_{i,T}) \Big) \\
&\text{subj to} \sum_{i=1}^{N} H_i z_{i,\tau} \leq h, && \tau \in [0,T] \\
&\qquad z_{i,\tau+1} = A_i z_{i,\tau} + B_i u_{i,\tau} && \forall i, \tau \in [0,T] \\
&\qquad z_{i,\tau} \in Z_i, \quad u_{i,\tau} \in U_i, && \forall i, \tau \in [0,T]
\end{aligned}
$$

Chapter 2

Preliminaries on Algebraic Graph Theory

Definition 2.1 (Digraph)
A digraph is a pair $G = (I, E)$ where $I = 1, \ldots, N$ is a set of elements called *nodes* and $E \subset I \times I$ is a set of ordered node pairs called *edges*
Edge: the pair (i, j) denotes an edge from i to j
Self-loop: edge from a node to itself, i.e. (i, i)

Definition 2.2 (Undirected (di)graph)
if for any $(i, j) \in E$ then $(j, i) \in E$

Definition 2.3 (Subgraph)
(I', E') subgraph of (I, E) if $I' \subset I$ and $E' \subset E$. Spanning subgraph if $I' = I$

Definition 2.4 (In-neighbours of i)
$j \in I$ is an in-neighbour of $i \in I$ if $(j, i) \in E$

Definition 2.5 (Set of in-neighbours of i)
$\mathcal{N}_i^{\text{IN}} = \{j \in \{1, \ldots, N\} | (j, i) \in E\}$

Definition 2.6 (Out-neighbours of i)
$j \in I$ is an out-neighbour of $i \in I$ if $(i, j) \in E$

Definition 2.7 (Set of out-neighbours of i)
$\mathcal{N}_i^{\text{IN}} = \{j \in \{1, \ldots, N\} | (i, j) \in E\}$

Definition 2.8 (In-degree $\deg_i^{\text{IN}}$)
number of in-neighbours, i.e. carinality of $\mathcal{N}_i^{\text{IN}} (\deg_i^{\text{IN}} = |\mathcal{N}_i^{\text{IN}}|)$
Out-degree analogous

Definition 2.9 (Balanced digraph)
A digraph G is balanced if $\deg_i^{\text{IN}} = \deg_i^{\text{OUT}}$ for all $i \in \{1, \ldots, N\}$

Definition 2.10 (Complete graph)
Unweighted graph such that $\forall i, j \; \exists \; (i, j), \; (j, i) \in E$

Chapter 3

Averaging Systems

3.1 Distributed algorithm

Given a network of N agents communicating according to a fixed digraph G, i.e. each agent i can receive messages only from in-neighbours in the graph, i.e. from $j \in \mathcal{N}_i^{\text{IN}}$. We start by considering a fixed graph, thus, each agent communicates with the same neighbours at each iteration $k \in \mathbb{N}$

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

where stf_i is a function depending only on state x_i and states $x_j, j \in \mathcal{N}_i^{\text{IN}}$.

Alternative version with out-neighbours:

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

3.2 Discrete-time averaging systems

Let $G^{\text{comm}} = (I, E)$ be a fixed (communication) digraph (self loops included). A linear averaging distributed algorithm can be written as:

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

where $x_i^k \in \mathbb{R}$ is the state of agent i at k and $a_{ij} > 0$ are positive weights.

Remark. The weights a_{ij} are defined only for $(i, j) \in E$

Wach i uses only the states of neighbours $j \in \mathcal{N}_i^{\text{IN}}$, thus distributed algorithm.

For analysis purposes, let us define weights $a_{ij} = 0$ for $(j, i) \notin E$. Thus we can rewrite the distributed algorithm as

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

This is a LTI autonomous system

$$\begin{bmatrix} x_1^{k+1} \\ \vdots \\ x_N^{k+1} \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1^k \\ \vdots \\ x_N^k \end{bmatrix}$$

Which can be compactly written as

$$x^{k+1} 0 A x^k$$

Remark. The matrix A can be seen as the weighted adjacency matrix of the reverse digraph $G^{\text{comm,rev}}$ of the digraph G^{comm}

If instead of in-neighbours we use out-neighbours, we call the digraph a sensing digraph G^{sens}. In this case the notation becomes consistent with graph theory, so we get

$$x^{k+1} = Ax^k$$

where A can be seen as the weighted adjacency matrix of the sensing digraph G^{sens}

3.3 Stochastic matrices

The non-negative square matrix $A \in \mathbb{R}^{N\times N}$ is

- row stochastic if $A\mathbf{1} = \mathbf{1}$ (each row sums to 1)
- column stochastic if $A^\top \mathbf{1} = \mathbf{1}$ (each column sums to 1)
- doubly stochastic if both row and column stochastic.

Lemma. Let A be a row-stochastic matrix and G the associate digraph. If G is strongly connected and aperiodic, then

1. the eigenvalue $\lambda = 1$ is simple;
2. all the other eigenvalues μ satisfy $|\mu| < 1$

Remark. The condition "G contains a globally reachable node and the subgraph of globally reachable noes is aperiodic" is necessary and sufficient

Theorem 3.1 (Consensus)
Consider a (discrete-time) averaging system with associated digraph G and wieghted adjacency matrix A. Assume G is strongly connected and aperiodic, and A is row stochastic. Then

1. there exists a left eigenvector $w \in \mathbb{R}^N, w > 0$ (i.e. with positive components $w_i > 0$ for all $i = 1, \dots, N$) such that
$$\lim_{k\to\infty} x^k = \mathbf{1}\frac{w^\top x^0}{w^\top \mathbf{1}} = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix} \frac{\sum_{i=1}^N w_i x_i^0}{\sum_{i=1}^N w_i}$$
i.e., consensus is reached to $\dfrac{\sum_{i=1}^N w_i x_i^0}{\sum_{i=1}^N w_i}$
2. if additionally A is doubly stochastic, then
$$\lim_{k\to\infty} x^k = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix} \frac{\sum_{i=1}^N x_i^0}{N}$$
i.e., average consensus is reached

3.4 Example: Metropolis-Hastings weights

Given an undirected unweighted graph G with edge set E and degrees $d_1, \dots, d_n$

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

Result: the matrix A is symmetric and doubly-stochastic.

3.5 Time-varying digraphs

A time-varying digraph is a sequence of digraphs $\{G(k)\}_{k\geq 0}$.

Remark. The main definitions of in/out neighbours, in/out degree, adjacency matrix can be generalized by considering time-varying versions, i.e. $\mathcal{N}_i^{\text{IN}}(k)$, $\mathcal{N}_i^{\text{OUT}}(k)$, $\deg_i^{\text{IN}}(k)$,$\deg_i^{\text{OUT}}(k)$, $A(k)$ associated to each graph $G(k)$. Connectivity requires new definitions as assuming each $G(k)$ to be connected is too conservative.

Definition 3.1 (Jointly strongly connected digraph)
if $\bigcup_{\tau=k}^{+\infty} G(\tau)$ is strongly connecetd $\forall k \geq 0$

Definition 3.2 (Uniformly jointly strongly connected (or B-strongly connected) digraph)
if there exists $B \in \mathbb{N}$ such that $\bigcup_{\tau=k}^{k+B} G(\tau)$ is strongly connecetd $\forall k \geq 0$

Remark. The graph can be disconnected at some time k.

3.5.1 Averaging distributed algorithms over time-varying graphs

Let $\{G(k)\}_{k\geq 0}$ be a time-varying digraph (with self loops for each $G(k)$). Consider the distributed algorithm

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

or the out-neighbours version

$$x_i^{k+1} = \sum_{j\in\mathcal{N}_i^{\text{OUT}}(k)} a_{ij}(k)x_j^k \qquad \forall i \in \{1,\dots,N\}$$

where $x_i^k \in \mathbb{R}$ is the state of agent i at k and $a_{ij}(k) > 0$.

For analysis purposes, let us define weights $a_{ij}(k) = 0$ for $(i,j) \notin E(k)$. Thus we can rewrite the distributed algorithm as

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

This is a Linear Time-Varying system

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

with state $x := [x_1,\dots,x_N]^\top$ and state matrix

$$A(k) := \begin{bmatrix} a_1 1k & \cdots & a_{1N}(k) \\ \vdots & \ddots & \vdots \\ a_N 1k & \cdots & a_{NN}(k) \end{bmatrix}$$

being a weighted adjacency matrix associated to the digraph $G(k)$.

3.5.2 Discrete-time consensus over time-varying graphs

Theorem 3.2
Let $\{A(k)\}_{k\geq 0}$ be a sequence of row-stochastic matrices with associated digraphs $\{G(k)\}_{k\geq 0}$. Assume

1. each digraph $G(k)$ has a self-loop at each node;
2. each non-zero edge weight $a_{ij}(k)$, including the self-loop wights $a_{ii}(k)$, is larger than a constant $\epsilon > 0$;
3. there exists $B \in \mathbb{N}$ such that, for all times $k \geq 0$, the union digraph $G(k) \cup \cdots \cup G(k+B)$ is strongly connected.

Then

1. there exists a non-negative vector $w \in \mathbb{R}^N$ such that the solution to $x^{k+1} = A(k)x^k$ converges (exponentially) to $\mathbf{1}\frac{w^\top x^0}{w^\top \mathbf{1}}$, i.e.
$$\lim_{k\to\infty} x^k = \mathbf{1}\left(\frac{w^\top x^0}{w^\top \mathbf{1}}\right)$$
2. if additionally each matrix in the sequence is doubly-stochastic, then
$$\lim_{k\to\infty} x^k = \mathbf{1}\frac{1}{N}\sum_{i=1}^{N} x_i^0$$
i.e., average consensus is achieved

3.6 Laplacian dynamics

Consider a network of dynamical systems with dynamics
$$\dot{x}(t) = u_i(t) \qquad i \in \{1, \dots, N\}$$
with states $x_i \in \mathbb{R}$ and inputs $u_i \in \mathbb{R}$, communicating (or interacting) according to a digraph G. Consider a (distributed) "proportional" feedback control
$$u_i(t) = -\sum_{j\in\mathcal{N}_i^{\text{IN}}} a_{ij}(x_i(t) - x_j(t))$$
or the out-neighbour version
$$u_i(t) = -\sum_{j\in\mathcal{N}_i^{\text{OUT}}} a_{ij}(x_i(t) - x_j(t))$$
For analysis purposes, let us define weights $a_{ij}(k) = 0$ for $(i,j) \notin E(k)$. Thus we can rewrite the distributed control systems as
$$\dot{x}_i(t) = -\sum_{j=1}^{N} a_{ij}(x_i(t) - x_j(t)) \qquad \forall i \in \{1, \dots, N\}$$
Defining $x := [x_1 \cdots x_N]^\top$, it can be shown that it can be rewritten as the following Linear Time Invariant continuous-time system
$$\dot{x}(t) = -Lx(t)$$
where L is the (weighted) Laplacian associated to the digraph G with (weighted) adjacency matrix A

Let
$$\dot{x}_i(t) = -\sum_{j=1}^{N} a_{ij}(x_i(t) - x_j(t)) \qquad \forall i \in \{1, \dots, N\}$$
rearranging terms
$$\dot{x}_i(t) = -\left(\sum_{j=1}^{N} a_{ij}\right) x_i(t) + \sum_{j=1}^{N} a_{ij}x_j(t) = -\deg_i^{\text{OUT}}\, x_i(t) + (Ax(t))_i$$
where $(Ax(t))_i$ is the i-th element of $Ax(t)$. Writing the previous dynamics in a compact form
$$\dot{x}(t) = -(D^{\text{OUT}} - A)x(t)$$
where we recall that D^{OUT} is the (weighted) out-degree matrix. Recalling that $L = D^{\text{OUT}} - A$, it holds that
$$\dot{x}(t) = -Lx(t)$$
Remark. if the in-neighbours version is considered, then $\dot{x}(t) = -L^{\text{IN}}x(t)$, where $L^{\text{IN}} = D^{\text{IN}} - A^T$ is the in-degree Laplacian (i.e. the Laplacian of the reverse graph of G)

3.6.1 Properties of the Laplacian matrix

It can be easily verified that

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

i.e., $\lambda = 0$ is an eigenvalue of L and $\mathbf{1}$ is an associated eigenvector.

Lemma. Given a weighted digraph with Laplacian L, then all eigenvalues of L different from zero have strictly positive real part

Lemma. Given a weighted digraph with Laplacian L, the following statements are equivalent:

1. G is weight-balanced, i.e. $D^{\text{IN}} = D^{\text{OUT}}$
2. $\mathbf{1}L = 0$

Theorem 3.3
A weighted digraph with Laplacian L contains a globally reachable node if and only if $\lambda = 0$ is simple.

Corollary. If a weighted digraph is strongly connected, then $\lambda = 0$ is simple

3.6.2 Consensus for Laplacian dynamics

Theorem 3.4
let L be a (weighted) Laplacian matrix with associated strongly connected (weighted) digraph G. Consider the Laplacian dynamics $\dot{x}(t) = -Lx(t),\ t \geq 0$, then

1. $$\lim_{t\to\infty} x(t) = \mathbf{1}\left(\frac{w^\top x(0)}{w^\top \mathbf{1}}\right)$$
   with $w^\top L = 0$, i.e. w is a left eigenvector for the eigenvalue $\lambda = 0$;
2. if additionally G is weight-balanced then
   $$\lim_{t\to\infty} x(t) = \mathbf{1}\frac{\sum_{i=1}^N x_i(0)}{N}$$

Chapter 4

Optimization basics

Convexity and gradient monotonicity

If a convex function ℓ is also differentiable, then its gradient $\nabla \ell : \mathbb{R}^d \to \mathbb{R}^d$ satisfies

$$\left(\nabla \ell(z_A) - \nabla \ell(z_B)\right)^\top (z_A - a_B) \geq 0$$

for all z_A, z_B. That is, the gradient $\nabla \ell$ is a monotone operator

Strict convexity and gradient monotonicity

A function ℓ is strictly convex if for $z_A \neq z_B$ and $\theta \in (0, 1)$

$$\ell(\theta z_A + (1-\theta) z_B) < \theta \ell(z_A) + (1-\theta)\ell(z_B)$$

If the strictly convex function ℓ is also differentiable, then its gradient satisfies

$$\left(\nabla \ell(z_A) - \nabla \ell(z_B)\right)^\top (z_A - z_B) > 0$$

for all z_A, z_B. That is, the gradient $\nabla \ell$ is a strictly monotone operator

Strong convexity and gradient monotinicity

A function ℓ is strongly convex with parameter $\mu > 0$ if for $z_A \neq z_B$ and $\theta \in (0, 1)$

$$\ell(\theta z_A + (1-\theta) z_B) < \theta \ell(z_A) + (1-\theta)\ell(z_B) - \mu\theta(1-\theta)\|z_A - z_B\|^2$$

The gradient of a differentiable strongly convex function satisfies

$$\left(\nabla \ell(z_A) - \nabla \ell(z_B)\right)^\top (z_A - z_B) \geq \mu \|z_A - z_B\|^2$$

for all z_A, z_B. That is, the gradient $\nabla \ell$ is a strongly monotone operator

Convexity and Lipschitz continuity of the gradient

Consider a differentiable convex function ℓ with a Lipschitz continuous gradient with parameter $L > 0$, i.e.

$$\|\nabla \ell(z_A) - \nabla \ell(z_B)\| \leq L \|z_A - z_B\|$$

for all z_A, z_B. Then, the following characterization holds

$$\left(\nabla \ell(z_A) - \nabla \ell(z_B)\right)^\top (z_A - z_B) \geq \frac{1}{L} \|\nabla \ell(z_A) - \nabla \ell(z_B)\|^2$$

for all z_A, z_B. That is, the gradient $\nabla \ell$ is a co-coercive operator

Strong convexity and Lipschitz continuity of the gradient

Consider a strongly convex (with parameter $\mu > 0$) function ℓ with Lipschitz continuous gradient (with parameter $L > 0$). The the followin characterization holds

$$(\nabla \ell(z_A) - \nabla \ell(z_B))^\top (z_A - z_B) \geq \frac{\mu L}{\mu + L}\|z_A - z_B\|^2 + \frac{1}{\mu + L}\|\nabla \ell(z_A) - \nabla \ell(z_B)\|^2$$

for all z_A, z_B.

4.1 Optimization algorithms

We consider optimization algorithms based on iterative descent.

Notation. notatioWe denote by $z^k \in \mathbb{R}^d$ the estimate at iteration $k \in \mathbb{N}$ of a local minimum.

The algorithm starts at a given initial guess ecc ecc we know iterative descent

4.1.1 A system theoretical perspective to the gradient method

Let ℓ be μ-strongly convex and with L-Lipschitz continuous gradient. The gradient method is a discrete-time integrator in feedback interconnection with a static map

$$\begin{aligned} z^{k+1} &= z^k - \alpha u^k, \qquad z^0 \text{ given} \\ u^k &= \nabla \ell(z^k) \end{aligned}$$

where u^k is a "control input" obtained through a static (nonlinear) state feedback. The block diagram is

This is known as the *Lur'e problem*

4.2 steady-state analysis of the gradient method

Assume that the state z^k converges to some value z_{eq}. Then, such an equilibrium must satisfy:

$$\begin{aligned} z_{eq} = z_{eq} - \alpha \nabla \ell(z_{eq}) \implies & \qquad z_{eq} : \nabla \ell(z_e q) \\ \implies & \qquad z_{eq} = z^* \end{aligned}$$

Consider the change of coordinates $z^k \to \tilde{z}^k - z_{eq} = z^k - z^*$. Then, the error dynamics is

$$\begin{aligned} \tilde{z}^{k+1} &= \tilde{z}^k - \alpha u^k \\ u^k &= \nabla \ell(\tilde{z}^k + z^*) - \nabla \ell(z^*) \end{aligned}$$

where u^k and $\tilde{z}^k$ satisfy, in light of the assumption on ℓ, the following inequality[1]

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

Consider a Lyapunov function $V : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ given by $V(\tilde{z}) = \|\tilde{z}\|^2$. Then

$$\begin{aligned} V(\tilde{z}^{k+1}) - V(\tilde{z}^k) &= \|\tilde{z}^{k+1}\|^2 - \|\tilde{z}^k\|^2 \\ &= \|\tilde{z}^k\|^2 - 2\alpha (u^k)^\top \tilde{z}^k + \alpha^2 \|u^k\|^2 - \|\tilde{z}^k\|^2 \\ &\leq -2\alpha\gamma_1 \|\tilde{z}^k\|^2 + \alpha(\alpha - 2\gamma_2)\|u^k\|^2 \end{aligned}$$

For a small enough stepsize α (i.e., $\alpha \leq 2\gamma_2$), we can write

$$\begin{aligned} V(\tilde{z}^{k+1}) - V(\tilde{z}^k) < -2\alpha\gamma_1 \|\tilde{z}^k\|^2 \quad \implies \quad \|\tilde{z}^{k+1}\|^2 &\leq (1 - 2\alpha\gamma_1)\|\tilde{z}^k\|^2 \\ &\leq (1 - 2\alpha\gamma_1)^k \|\tilde{z}^0\|^2 \end{aligned}$$

Therefore $\{\tilde{z}^k\}_{k \in N}$ goes exponentially/geometrically fast to zero

[1] For all z_A, z_B it holds that

$$(\nabla \ell(z_A) - \nabla \ell(z_B))^\top (z_A - z_B) \geq \frac{\mu L}{\mu + L}\|z_A - z_B\|^2 + \frac{1}{\mu + L}\|\nabla \ell(z_A) - \nabla \ell(z_B)\|^2$$

4.2.1 Gradient method for quadratic programs

Consider a quadratic program

$$\min_z \frac{1}{2} z^\top Q z + r^\top z$$

With $Q = Q^\top > 0$ The gradient method is an affine linear system

$$\begin{aligned} z^{k+1} &= z^k - \alpha(Qz^k + r) \qquad z^k \text{ given} \\ &= (I - \alpha Q) z^k - \alpha r \end{aligned}$$

For a sufficiently small α, the state matrix $(I - \alpha Q)$ is Schur. Hence, the state trajectory is [2]

$$z^k = (I - \alpha Q)^k z^0 - \alpha \sum_{i=0}^{k-1} (I - \alpha Q)^i r \to^{k\to\infty} -\alpha \left(\sum_{i=0}^{\infty} (I -^{\alpha} Q)^i \right) r = -Q^{-1} r$$

4.2.2 Gradient flow

Let us swap the roles of the plant (the static nonlinearity) and the controller (the integrator) We obtain the so-called gradient flow (continuous-time dynamics)

$$\dot{z}(t) = -\nabla \ell(z(t)) \qquad z(0) = z_0$$

Remark. A solution to the ODE exists if the vector field is Lipschitz continuous

4.2.3 Nesterov accelerated gradient method

Consider the following two-step algorithm: for all $k \in \mathbb{N}$

$$\zeta^{k+1} = \zeta^k + \alpha_1(\zeta^k - \zeta^{k+1}) - \alpha_2 \nabla \ell \left(\zeta^k + \alpha_1(\zeta^k - \zeta^{k+1}) \right), \quad \zeta^0, \zeta^{-1} \text{ given}$$

for some $\alpha_1, \alpha_2 > 0$. It admits the state-space representation More general updates can also be considered: for all $k \in \mathbb{N}$

$$\zeta^{k+1} = \zeta^k + \alpha_1(\zeta^k - \zeta^{k+1}) - \alpha_2 \nabla \ell \left(\zeta^k + \alpha_3(\zeta^k - \zeta^{k-1}) \right), \quad \zeta^0, \zeta^{-1} \text{ given}$$

for some $\alpha_1, \alpha_2, \alpha_3 > 0$

[2] The geometric series $\sum_{i=0}^{\infty} \rho^i$ is equal to $(1-\rho)^{-1}$ for all $\rho < 1$

Chapter 5

Parallel Optimization and Federated Learning

Cost-coupled optimization for learning

In learning applications, we usually consider optimization problems in the form

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

where, for all $i = 1, \dots, N$, the cost function $\ell_i : \mathbb{R}^d \to \mathbb{R}$ is local and private

(Batch) gradient method for learning

Consider the optimization problem

$$\min_{z} \sum_{i=1}^{N} \ell_i(z)$$

The (batch) gradient method is: for each iteration $k \in \mathbb{N}$

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

Remark. computation can be expensive

5.1 Incremental gradient method

Consider the optimizatio problem

$$\min_{z} \sum_{i=1}^{N} \ell_i(z)$$

Idea: rather than using the whole batch gradient at each $k \in \mathbb{N}$, just select one single "sample" per iteration.

The *incremental gradient method* is: for each iteration $k \in \mathbb{N}$

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

where $i^k \in \{1, \dots, N\}$ Two rules for choosing index i^k at iteration k:

- Cyclic rule
- Randomized rule

5.2 Stochastic Gradient Descent

consider the stochastic optimization problem

$$\min_z \mathbb{E}_{\mathcal{W}}[\ell(z, \mathcal{W})]$$

where $\mathbb{E}_{\mathcal{W}}[\cdot]$ denotes the expected value with respect to the random variable $\mathcal{W}$ (possibly having an uknown probability distribution $p_{\mathcal{W}}(w)$)

Remark. for all z, also $\ell(z, \mathcal{W})$ is a random variable, whose probability distribution depends on $p_{\mathcal{W}}$ and ℓ. Moreover, the gradient $\nabla\ell(z, \mathcal{W})$ at each z is a random quantity

Assumption: There exsists an oracle that, given a realization $\bar{w}$ of $\mathcal{W}$, returns the corresponding realization of the gradient $\nabla\ell(\bar{z}, \bar{w})$ at any query point $\bar{z}$

The stochastic gradient descent is: for each iteration $k \in \mathbb{N}$ draw a realization w^k of $\mathcal{W}$ and update

$$z^{k+1} = z^k - \alpha\nabla\ell(z^k, w^k)$$

5.2.1 Convergence

Proposition 5.1
Assume:

- ℓ is a μ-strongly convex function with L-Lipschitz continuous gradient (uniformly in its second argument)
- $\nabla\ell(z, \mathcal{W})$ is an unbiased estimatee of $\nabla_z\mathbb{E}_{\mathcal{W}}[\ell(z, \mathcal{W})]$
- $\|\nabla\ell(z, \mathcal{W})\| \leq M$ almost surely[1] for some $M > 0$

5.3 Beyond SGD: Adaptive Momentum Estimation (Adam)

The ADAM algorithm reads as follows

- Mean and Variance (first momentum and second momentum)

$$\begin{aligned} m^{k+1} &= \beta_1 m^k + (1-\beta_1)\nabla\ell(z^k, w^k), && \text{for some } \beta \in (0,1) \\ v^{k+1} &= \beta_2 v^k + (1-\beta_2)[\nabla\ell(z^k, w^k)]^2, && \text{for some } \beta_2 \in (0,1) \end{aligned}$$

  where the square operation $[\cdot]^2$ is meant component-wise

- Construnct the descent direction

$$\begin{aligned} \hat{m} &= \frac{1}{1-\beta_1^{k+1}} m^{k+1} \\ \hat{v} &= \frac{1}{1-\beta_2^{k+1}} v^{k+1} \\ d^k &= -\frac{\hat{m}}{\sqrt{\hat{v}} + \epsilon}, \qquad \text{for some } \epsilon > 0 \end{aligned}$$

  where the division in the last equation is meant element-wise

- Update of the solution estimate

$$z^{k+1} = z^k + \alpha d^k$$

[1] A sequence of random variables $\{\mathcal{X}_k\}_{k\in\mathbb{N}}$ converges almost surely to the rv $\mathcal{X}$ if $\mathbb{P}(\lim_{k\to\infty} \mathcal{X}_k = \mathcal{X}) = 1$

5.4 Federated learning

Consider the optimization problem

$$\min_z \sum_{i=1}^{N} \ell(z; \mathcal{D}^i, p^i)$$

Paradigm:

- local private data $\mathcal{D}^i = ([\mathcal{D}\rangle]_1, [\mathcal{D}\rangle]_2, \ldots, [\mathcal{D}\rangle]_d)$ and p^i
- learn common parameters $z^* \in \mathbb{R}^d$ (common neural network)
- communication with a parameter server

5.5 Distributed learning

Consider the optimization problem

$$\min_z \sum_{i=1}^{N} \ell(z; \mathcal{D}^i, p^i)$$

Paradigm:

- local private data $\mathcal{D}^i = ([\mathcal{D}\rangle]_1, [\mathcal{D}\rangle]_2, \ldots, [\mathcal{D}\rangle]_d)$ and p^i
- learn common parameters $z^* \in \mathbb{R}^d$ (common neural network)
- communication with neighbours only

Chapter 6

Leader Follower networks: Formation control

6.1 analogy with mass-spring systems

Consider a platoon of N masses such that each mass i is connected with mass $i-1$ and $i+1$ through a spring with elastic constants respectively $a_{i-1,i} = a_{i,i-1} > 0$ and $a_{i+1,i} = a_{i,i+1} > 0$. Let $x_i \in \mathbb{R}$ be the position of mass i

The elastic force at mass i, $F_{e,i}(x)$ is given by

$$F_{e,i}(x) = -a_{i,i-1}(x_i - x_{i-1}) - a_{i,i+1}(x_i - x_{i+1})$$

For each spring, we can write the associated elastic force as the negative gradient of the elastic (potential) energy, so that

$$F_{e,i}(x) = -\frac{\partial}{\partial x_i}(\frac{1}{2}a_{i,i-1}\|x_i - x_{i-1}\|^2 + \frac{1}{2}a_{i,i+1}\|x_i - x_{i+1}\|^2)$$

Let us suppose that each spring can be written as the parallel of two springs with elastic constants respectively $\frac{1}{2}a_{i,i-1} > 0$ and $\frac{1}{2}a_{i,i+1} > 0$.

Let $x_i \in \mathbb{R}$ be the positino of mass i. The elastic force at mass i, $F_{e,i}(x)$ is given by

$$F_{e,i} = -(\frac{1}{2}a_{i,i-1} + \frac{1}{2}a_{i,i-1})(x_i - x_{i-1}) - (\frac{1}{2}a_{i,i+1} + \frac{1}{2}a_{i,i+1})(x_i - x_{i+1})$$

As before, we can write the elastic force as the negative gradient of the elastic (potential) energy, i.e.

$$F_{e,i}(x) = -\frac{\partial}{\partial x_i}(\frac{1}{2}\frac{a_{i,i-1}}{2}\|x_i - x_{i-1}\|^2 + \frac{1}{2}\frac{a_{i,i-1}}{2}\|x_i - x_{i-1}\|^2 + \frac{1}{2}\frac{a_{i,i+1}}{2}\|x_i - x_{i+1}\|^2 + \frac{1}{2}\frac{a_{i,i+1}}{2}\|x_i - x_{i+1}\|^2)$$

The total elastic (potential) energy of the mass-spring system can be written as

$$\begin{aligned} V(x) &= \sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\frac{1}{2}\frac{a_{i,j}}{2}\|x_i - x_j\|^2 \\ &= \sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} V_{ij}(x_i, x_j) \end{aligned}$$

where we have defined $\mathcal{N}_i := \{i-1, i+1\}$ and $V_{ij}(x_i, x_j) := \frac{1}{2}\frac{a_{i,j}}{2}\|x_i - x_j\|^2$.

Thus, the elastic force at mass i can be seen as the negative gradient of the energy, i.e.

$$\begin{aligned} F_{e,i}(x) &= -\frac{\partial}{\partial x_i}\sum_{j\in\mathcal{N}_i}(V_{ij}(x_i, x_j) + V_{ji}(x_j, x_i)) \\ &= -\frac{\partial}{\partial x_i}V(x) \end{aligned}$$

This formulation can be extended to more general systems in which masses are interconnected according to a topology described by an undirected graph $G = (\{1, \dots, N\}, E)$

By adding a damping term on each mass, the system dynamics can be written as

$$\begin{aligned} \dot{x}_i &= v_i \\ m_i \dot{v}_i &= -v_i - \frac{\partial}{\partial x_i} V(x) \quad \forall i \in \{1, \dots, N\} \end{aligned}$$

where we have considered the damping coeffiecient equal to one.

If we assume that masses are small, we may written

$$v_i \approx -\frac{\partial}{\partial x_i} V(x)$$

so that the dynamics may be approximated by the following firt order dynamics

$$\dot{x}_i = -\frac{\partial}{\partial x_i} V(x) \quad \forall i \in \{1, \dots, N\}$$

Consider a network of N agents communicating/interacting according to a fixed, undirected graph G. Let $x_i(t) \in \mathbb{R}^d$ be the state of agent i. Let agents run a Laplacian dynamics

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i} a_{ij}(x_i - x_j) \quad \forall i \in \{1, \dots, N\}$$

We can rewrite it as

$$\dot{x}_i(t) = -\sum_{j \in \mathcal{N}_i} \frac{\partial}{\partial x_i} \left(V_{ij}(x_i, x_j) + V_{ji}(x_j, x_i) \right) \quad \forall i \in \{1, \dots, N\}$$

with $V_{ij}(x) = \frac{1}{2} \frac{a_{i,j}}{2} \|x_i - x_j\|^2$.

By recalling the definition of the total energy

$$V(x) = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} V_{ij}(x_i, x_j)$$

the Laplacian dynamics

$$\dot{x} = -Lx$$

can be seen as a "gradient flow", i.e.

$$\dot{x} = -\nabla V(x)$$

Thus, the consensus configuration can be seen as a stationary point of V. This idea can be extended to general potential functions and applied to distributed control systems.

Consider a network of N autonomous agents communicating/interacting according to a fixed, undirected graph. Let $x_i(t) \in \mathbb{R}^d$ be the state of agent i. Consider a global potential function defined as

$$V(x) = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} V_{ij}(x_i, x_j)$$

such that (local) minima of the potential correspond to desired configurations of the team. The gradient flow dynamics $\dot{x} = -\nabla V(x)$ turns out to be distributed. That is,

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i} \frac{\partial}{\partial x_i} \left(V_{ij}(x_i, x_j) + V_{ji}(x_j, x_i) \right) \qquad \forall i \in \{1, \dots, N\}$$

We can define a desired formation by assigning a set of distances, d_{ij}, between neighbouring agents i and J in a suitable graph

The main idea for formation control is to define a potential function matching the sparsity of G, $V^{\text{form}}(x) = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} V_{ij}^{\text{form}}(i,j)$, such that a configuration $x^{textform}$ satisfying

$$\|x_i^{\text{form}} - x_j^{\text{form}}\| = d_i j \qquad \forall (i,j) \in E$$

is a minimum of V.

In order to reach a formation with assigned distances d_{ij}, let us define

$$V_{ij}^{\text{form}}(x) = \frac{1}{8} \left(\|x_i - x_j\|^2 - d_{ij}^2 \right)^2$$

with corresponding (global) potential function $V^{\text{form}}(x) = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} V_{ij}^{\text{form}}(x_i, x_j)$.

The gradient flow dynamics of each agent i is given by

$$\dot{x}_i = - \sum_{j \in \mathcal{N}_i} \frac{\partial}{\partial x_i} \left(V_{ij}^{\text{form}}(x_i, x_j) + V_{ji}(x_j, x_i) \right) \qquad \forall i \in \{1, \dots, N\}$$

which reads as

$$\dot{x}_i = - \sum_{j \in \mathcal{N}_i} \left(\|x_i - x_j\|^2 - d_{ij}^2 \right) (x_i - x_j) \qquad \forall i \in \{1, \dots, N\}$$

This dynamics has multiple equilibrium points, including the desired formation in which the agents are at the assigned distances. In particular, the consensual solution $x_1 = x_2 = \cdots = x_N$ is an (undesired) equilibrium.

Such a "degenerate" equilibrium can be avoided by means of additional "collision avoidance" potential functions $V_{ij}^{\text{ca}}(x_i, x_j)$ such that

$$\lim_{\|x_i - x_j\| \to 0} V_{ij}^{\text{ca}}(x_i, x_j) = +\infty$$

A possible solution is a barrier function given

$$V_{ij}^{\text{ca}} = - \log(x_i - x_j)$$

Similarly, barrier potential functions $V^{\text{obs}}(x_i)$, depending only on the state of agent x_i, can be used to avoid obstacles.

The formation control dynamics becomes

$$\dot{x}_i = - \frac{\partial V^{\text{form}}}{\partial x_i} - \frac{\partial V^{\text{ca}}(x)}{\delta x_i} - \nabla V^{\text{obs}}(x) \qquad \forall i \in \{1, \dots, N\}$$

Chapter 7

Distributed Aggregative Optimization

Consider N robots in the plane that want to optimize their positions $z_i \in \mathbb{R}^2$, for all $i = 1, \dots, N$ to perform multi-robot surveillance. Let:

- $r_0 \in \mathbb{R}^2$ be a target to protect
- $r_i \in \mathbb{R}^2$ be the intruder associated to robot i
- $\sigma(z) = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} z_i$ is the barycenter of the robots
- Local cost function of robot i

$$\ell_i(z_i, \sigma(z)) = \gamma_i \|z_i - r_i\|^2 + \|\sigma(z) - r_0\|^2$$

with $z \in \mathbb{R}^{2N}$ the stack of $z_1, \dots, z_N$ and $\gamma_i > 0$ being a tradeoff parameter.

7.1 Aggregative optimization

Let us consider aggregative optimization problems in the form

$$\min_{z_1,\dots,z_N} \sum_{i=1}^{N} \ell_i(z_i, \sigma(z))$$

where the aggregative variable $\sigma(z)$ is defined as

$$\sigma(z) = \frac{1}{N}\sum_{i=1}^{N} \phi_i(z_i)$$

where

- $z = (z_1, \dots, z_N)$, with each $z_i \in \mathbb{R}^{n_i}$, for all $i = 1, \dots, N$
- $\ell_i : \mathbb{R}^{n_i} \times \mathbb{R}^d \to \mathbb{R}$ and $\phi_i \mathbb{R}^{n_i} \to \mathbb{R}^d$, for all $i = 1, \dots, N$

For scalar states, $z_i \in \mathbb{R}$, the *centralized gradient* method at iteration k reads as

$$z_i^{k+1} = z_i^k - \alpha \frac{\partial}{\partial z_i} \left(\sum_{j=1}^{N} \ell_j(z_j, \sigma(z_1, \dots, z_N)) \right) \Bigg|_{z_1 = z_1^k, \dots, z_N = z_N^k}$$

for all $i = 1, \dots, N$ where $\alpha > 0$ is the stepsize

Gradient computation (scalar case)

Since the cost function is a composite function, we need the chain rule to compute its derivative with respect to z_i

$$\begin{aligned}
&\frac{\partial}{\partial z_i}\left(\sum_{j=1}^{N}\ell_j(z_j,\sigma(z_1,\dots,z_N))\right)\Bigg|_{z_1=z_1^k,\dots,z_N=z_N^k}\\
&=\frac{\partial}{\partial z_i}\ell_i(z_i,\sigma)\Big|_{z_i=z_i^k,\sigma=\frac{1}{N}\sum_{j=1}^{N}\phi_j(z_j^k)}\\
&\quad+\sum_{j=1}^{N}\frac{\partial}{\partial\sigma}l_j(z_j,\sigma)\Bigg|_{z_j=z_j^k,\sigma=\frac{1}{N}\sum_{j=1}^{N}\phi_j(z_j^k)}\cdot\frac{\partial\sigma(z_1,\dots,z_N)}{\partial z_i}\Bigg|_{z_1=z_1^k,\dots,z_N=z_N^k}
\end{aligned}$$

Notice that $\dfrac{\partial\sigma(z_1,\dots,z_N)}{\partial z_i}=\dfrac{1}{N}\dfrac{d}{dz_i}\phi_i(z_i)$ can be computed locally

As in the scalar case, we use the chain rule to compute the gradient of the composite function. The i-th block of the gradient, denoted as $\left[\nabla\left(\sum_{j=1}^{N}\ell_j(z_j,\sigma(z_1,\dots,z_N))\right)\right]_i\in\mathbb{R}^{n_i}$, is given by

$$\begin{aligned}
&\left[\nabla\left(\sum_{j=1}^{N}\ell_j(z_j,\sigma(z_1,\dots,z_N))\right)\right]_i\\
&=\nabla_1\ell_i(z_i,\sigma)|_{z_j=z_j^k,\sigma=\frac{1}{N}\sum_{j=1}^{N}\phi_j(z_j^k)}\\
&\quad+\frac{1}{N}\nabla\phi_i(z_i)\Big|_{z_i=z_i^k}\cdot\sum_{j=1}^{N}\nabla_2\ell_j(z_j,\sigma)\Bigg|_{z_j=z_j^k,\sigma=\frac{1}{N}\sum_{j=1}^{N}\phi_j(z_j^k)}
\end{aligned}$$

7.2 Distributed aggregative optimization

In a distributed contex, each agent i

- knows only ℓ_i and ϕ_i
- maintains an estimate z_i^k of $z_i^\star$
- maintains an estimate s_i^k of $\phi(z^k)=\dfrac{1}{N}\sum_{j=1}^{N}\phi_j(z_j^k)$
- maintains an estimate v_i^k of $\sum_{j=1}^{N}\nabla_2\ell_j(z_j^k,\sigma(z^k))$

The "tracking" idea of gradient tracking algorithm is applied to aggregative optimization

$$\begin{aligned}
z_i^{k+1}&=z_i^k-\alpha\left(\nabla_1\ell_i(z_i^k,s_i^k)+\nabla\phi_i(z_i^k)v_i^k\right) && z_i^0\in\mathbb{R}^{n_i}\\
s_i^{k+1}&=\sum_{j\in\mathcal{N}_i}a_{ij}s_j^k+\phi_i(z_i^{k+1})-\phi_i(z_i^k) && s_i^0=\phi_i(z_i^0)\\
v_i^{k+1}&=\sum_{j\in\mathcal{N}_i}a_{ij}v_j^k+\nabla_2\ell_i(z_i^{k+1},s_i^{k+1})-\nabla_2\ell_i(z_i^k,s_i^k) && v_i^0=\nabla_2\ell_i(z_i^0,s_i^0)
\end{aligned}$$

Theorem 7.1 (aggregative tracking distributed optimization algorithm: convergence)
Assume G is a strongly connected and aperiodic digraph, and A is doubly stochastic. Assume that each function ℓ_i is strongly convex, the gradients $\nabla_1\ell_i$ and $\nabla_2\ell_i$ are Lipschitz continuous, and ϕ_i is differentiable and Lipschitz continuous.

Then, there exists $\alpha^\star$ such that for all $\alpha \in (0, \alpha^\star)$ the sequences of local solution estimates $\{z_1^k, \dots, z_N^k\}_{k\in\mathbb{N}}$ generated by the aggregative tracking distributed opitmization algorithm satisfy

$$\lim_{k\to\infty} \|z_i^k - z_i^\star\| = 0$$

at a linear rate, for all $i = 1, \dots, N$

7.2.1 Extension to online aggregative optimization

Consider a time-varying instance of the problem

$$\min_z \sum_{i=1}^N \ell_i^k(z_i, \sigma^k(z))$$
$$\text{subj. to } z_i \in Z_i^k, \qquad \forall i = 1, \dots, N$$

where $Z_i^k \subset \mathbb{R}^{n_i}$ are local constraint sets.

The goal is to design an algorithm generating a sequence $\{z_i^k\}_{k\in\mathbb{N}}$ that "tracks" the solution $z^{k,\star} = (z_1^{k,\star}, \dots, z_N^{k,\star})$ of the $k-th$ problem instance.

Remark. A regret R_K can be introduced for the analysis. Under suitable assumptions, it can be proven that

Chapter 8

Learning with Neural Networks

In supervised learning we have labeled data whose generic element(sample) is composed by

- an input point $\mathcal{D} \in \mathbb{R}^d$, i.e. a vector $\mathcal{D} = ([\mathcal{D}]_1, \dots, [\mathcal{D}]_d)$
- a lebel $p \in \mathbb{R}$ associated to the input point

A dataset is usually made by $\mathcal{M}$ samples, i.e. $((\mathcal{D}^1, p^1), \dots, (\mathcal{D}^{\mathcal{M}}, p^{\mathcal{M}}))$
Goal: reconstruct the uknown input-output map between inputs and labels
Strategy: approximate the map as a nonlinear function $\phi(\approxeq; \cdot) : \mathbb{R}^d \to \mathbb{R}$ parametrized byh $\approxeq$
Problem: find the best parameters $\approxeq^\star$ based on data

8.1 Basic element: the neuron model

A generic neuron h is a computational unit that

- has a set of weights $u_h \in \mathbb{R}^d$
- elaborates a vector $x \in \mathbb{R}^d$
- computes a scalar quantity
$$x_h^+ = \phi(x^\top u_h)$$
with $\sigma : \mathbb{R} \to \mathbb{R}$ being the *activation function*

Chapter 9

Multi-Robot Safety Controllers

Consider a nonlinear system

$$\dot{x}(t) = f(x(t), u(t)), \qquad x \in \mathbb{R}^n \quad u \in U \subset \mathbb{R}^m$$

Let us define a safe (state) set

$$X^s := \{x \in \mathbb{R}^n | V^s(x) \geq 0\}$$

for some sufficiently regular function $V^s : \mathbb{R}^n \to \mathbb{R}$

Goal: design a feedback control law $\kappa(x), \kappa : \mathbb{R}^n \to \mathbb{R}^m$ such that the set X^s is forward invariant

9.1 Control Barrier Functions

Consider the time derivative of $V^s(x(t))$ along the system trajectories

$$\begin{aligned}\frac{d}{dt}V^s(x(t)) &= \nabla V^s(x(t))^\top f(x(t)) + \sum_{h=1}^{m} \nabla V^s(x(t))^\top g_h(x(t))u_h(x(t)) \\ &= L_f V^s(x(t)) + L_g V^s(x(t))u(t)\end{aligned}$$

where L_f and L_g represent Lie derivatives. We say that V^s is a Control Barrier Function (CBF) if there exists a continuous, strictly increasing function $\gamma : \mathbb{R} \to \mathbb{R}$, with $\gamma(0) = 0$, such that

$$\sup_{u \in U} \{L_f V^s(x) + L_g V^s(x)u + \gamma(V^s(x)) \geq 0\}$$

for all $x \in \mathbb{R}^n$.

The above inequality induced by a CBF is called a *control barrier certificate*.

9.1.1 Admissible Control Space

We can define the set of admissible (safe) controllers for a given state x as

$$U^s(x) = \{u \in \mathbb{R}^m | L_f V^s(x) + L_g V^s(x)u + \gamma(V^s(x)) \geq 0\}$$

i.e., the set of inputs satisfying the control barrier certificates.

Remark. It can be shown that (sufficiently regular) feedback controllers satisfying the above condition render X^s forward invariant and asymptotically stable.

9.1.2 Safety Filters via Control Barrier Certificates

Let $u^{\text{ref}} \in \mathbb{R}^m$ be a (possibly unsafe) reference input, e.g., obtained by a higher-level controller. The safety controller is designed to be "minimally invasive", i.e. it aims at altering the reference controller as little as possible. A safe policy $u = \kappa(x)$ can be designed as

$$\begin{aligned}\kappa(x) = \arg\min_{u \in \mathbb{R}^m} \; & \|u - u^{\text{ref}}(x)\|^2 \\ \text{subj. to } & -L_f V^s(x) - L_g V^s(x)u - \gamma(V^s(x)) \leq 0\end{aligned}$$