

Distributed Autonomous Systems M

Dante Piotto

spring semester 2024

Contents

1	Introduction and scenarios	5
1.1	Distributed Autonomous System	5
1.2	Scenarios and applications of distributed systems	5
1.3	Measurement filtering in wireless sensor networks	5
1.4	Parameter Estimation in Wireless Sensor Networks	6
1.4.1	Opinion Dynamics in Social Influence Networks	6
1.5	Main questions in averaging algorithms	6
1.6	Distributed control in cooperative robotics	6
1.6.1	Main questions in cooperative robotics	7
1.6.2	Distributed optimal control	7
2	Preliminaries on Algebraic Graph Theory	9
3	Averaging Systems	11
3.1	Distributed algorithm	11
3.2	Discrete-time averaging systems	11
3.3	Stochastic matrices	12
3.4	Example: Metropolis-Hastings weights	12
3.5	Time-varying digraphs	13
3.5.1	Averaging distributed algorithms over time-varying graphs	13
3.5.2	Discrete-time consensus over time-varying graphs	13
3.6	Laplacian dynamics	14
3.6.1	Properties of the Laplacian matrix	15
3.6.2	Consensus for Laplacian dynamics	15
4	Optimization basics	17
4.1	Optimization algorithms	18
4.1.1	A system theoretical perspective to the gradient method	18
4.2	steady-state analysis of the gradient method	18
4.2.1	Gradient method for quadratic programs	19
4.2.2	Gradient flow	19
4.2.3	Nesterov accelerated gradient method	19
5	Parallel Optimization and Federated Learning	21
5.1	Incremental gradient method	21
5.2	Stochastic Gradient Descent	22
5.2.1	Convergence	22
5.3	Beyond SGD: Adaptive Momentum Estimation (Adam)	22
5.4	Federated learning	23
5.5	Distributed learning	23

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
- Distributed 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 $i \in \{1, \dots, N\}$

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

generalizing coefficients of the update:

$$\begin{aligned} x_i^0 &= x_{i0} \\ x_i^{k+1} &= \sum_{j=1}^N a_{ij} x_j^k \quad k \in \mathbb{N} \end{aligned}$$

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

Remark. $a_{ij} = 0$, for some $j \in \{1, \dots, N\}$, 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 $y_i \in \mathbb{R}^{m_i}$, B_i known matrix and v_i a random measurement noise. Assume v_1, \dots, 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

- rendezvous, 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 asymptotic states 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

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

Chapter 2

Preliminaries on Algebraic Graph Theory

Definition 2.1 (Digraph)

A digraph is a pair $G = (I, E)$ where $I = 1, \dots, N$ 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)

$\mathcal{N}_i^{\text{IN}} = \{j \in \{1, \dots, N\} | (j, i) \in E\}$

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)

$\mathcal{N}_i^{\text{OUT}} = \{j \in \{1, \dots, N\} | (i, j) \in E\}$

Definition 2.8 (In-degree \deg_i^{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 $i \in \{1, \dots, N\}$

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 $k \in \mathbb{N}$

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

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:

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

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 $(i, j) \in E$

Each 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

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

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

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

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 nodes is aperiodic" is necessary and sufficient

Theorem 3.1 (Consensus)

Consider a (discrete-time) averaging system with associated digraph G and weighted 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

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

i.e., consensus is reached to $\frac{\sum_{i=1}^N w_i x_i^0}{\sum_{i=1}^N w_i}$

2. if additionally A is doubly stochastic, then

$$\lim_{k \rightarrow \infty} x^k = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \frac{\sum_{i=1}^N x_i^0}{N}$$

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

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

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 connected $\forall k \geq 0$

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 connected $\forall k \geq 0$

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

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

or the out-neighbours version

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

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

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

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

$$A(k) := \begin{bmatrix} a_{11}(k) & \cdots & a_{1N}(k) \\ \vdots & \ddots & \vdots \\ a_{N1}(k) & \cdots & a_{NN}(k) \end{bmatrix}$$

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 weights $a_{ii}(k)$, 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 $G(k) \cup \dots \cup G(k+B)$ 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.

$$\lim_{k \rightarrow \infty} x^k = \mathbf{1} \left(\frac{w^\top x^0}{w^\top \mathbf{1}} \right)$$

2. if additionally each matrix in the sequence is doubly-stochastic, then

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

i.e., average consensus is achieved

3.6 Laplacian dynamics

Consider a network of dynamical systems with dynamics

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

with states $x_i \in \mathbb{R}$ and inputs $u_i \in \mathbb{R}$, communicating (or interacting) according to a digraph $G = (\{1, \dots, N\}, E)$. 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

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

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

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

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.

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

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

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

Chapter 4

Optimization basics

Convexity and gradient monotonicity

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

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

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

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

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

Strong convexity and gradient monotonicity

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

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

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

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

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. We 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

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

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:

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

Consider the change of coordinates $z^k \rightarrow \tilde{z}^k - z_{eq} = z^k - z^*$. 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 inequality¹

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

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

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

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

¹For 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

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

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

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

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)

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

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 $k \in \mathbb{N}$

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

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

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

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

²The 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 $\ell_i : \mathbb{R}^d \rightarrow \mathbb{R}$ 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 optimization 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 unknown 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 exists 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 surely¹ 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

- Construct the descent direction

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

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

- Update of the solution estimate

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

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

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